Adaptive Kernel Functions and Optimization Over a Space of Rank-One Decompositions

Roy Chih Chung Wang

A thesis submitted in partial fulfillment of the requirements for the
Doctor of Philosophy degree in Electrical Engineering

School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa

Acknowledgements

I am grateful to my mother, Su Fang Wang-Hu, my father Wen Yun Wang, and my brother, Louis Chih Yoa Wang. They have provided me with moral and emotional support throughout my life. I am grateful to my grandfather, Hu Ming Fung, for playing chess with me and discussing science ever since I started grade school. I am also grateful to my thesis adviser Prof. Eric Dubois for providing me with great guidance over these past few years. I will always remember his optimism, patience, and kindness.

I want to express my gratitude to Prof. Jochen Lang and Prof. Richard Dansereau for providing constructive feedback over the years; I have learned a lot from their positive attitude towards their work. I want to thank Prof. Jiying Zhao for being a member of my defence committee, Prof. Bertrand Jodoin for chairing my defence committee, and the external examiner Prof. Sri Krishnan for attending my defence in person.

I have a great deal of respect for Prof. Gilbert Strang, Prof. Stephen Boyd, Prof. Daphne Koller, Prof. Fredric Schuller, and any researcher who took the time to make their lectures available online. Their videos and my discussions with Prof. Dubois, really helped me throughout my studies. I want to also thank Dr. Nicolas Chenouard and Prof. Dimitri Van De Ville for answering my questions on their implementation of the Riesz transform.

A special shout-out goes out to the following people: Chris Whiten, Dr. Andrés Solís Montero, Dr. Mina Rafinazari, Dr. Navid Tadayon, Dr. Ahmad Al-Kabbany, Mohammad Esmaeel Mousa Pasandi, Po Kong Lai, Hamid Bazargani Gilani, Mehdi Arezoomand Ershadi, Dr. Houman Rastgar, Prof. Aaron Smith, Prof. Howard Schwartz, Prof. Rolf Clackdoyle, Samuel Witherspoon, Sean Benjamin, and Dr. Jean-Paul DeCruyenaere. I enjoyed insightful and fun discussions with them during my time as a doctoral student.

Finally, I want to thank my past mentors: Keith Thompson, Dr. Bob Ebrahim, Prof. Nicola Nicolici, Gord Harris, Prof. Joseph Hayward, Prof. Jamal Deen, and Prof. Qiyin Fang. My experiences with them encouraged me to pursue a doctoral degree.

Abstract

The representer theorem from the reproducing kernel Hilbert space theory is the origin of many kernel-based machine learning and signal modelling techniques that are popular today. Most kernel functions used in practical applications behave in a homogeneous manner across the domain of the signal of interest, and they are called *stationary kernels*. One open problem in the literature is the specification of a non-stationary kernel that is computationally tractable. Some recent works solve large-scale optimization problems to obtain such kernels, and they often suffer from non-identifiability issues in their optimization problem formulation. Many practical problems can benefit from using application-specific prior knowledge on the signal of interest. For example, if one can adequately encode the prior assumption that edge contours are smooth, one does not need to learn a finite-dimensional dictionary from a database of sampled image patches that each contains a circular object in order to up-convert images that contain circular edges.

In the first portion of this thesis, we present a novel method for constructing non-stationary kernels that incorporates prior knowledge. A theorem is presented that ensures the result of this construction yields a symmetric and positive-definite kernel function. This construction does not require one to solve any non-identifiable optimization problems. It does require one to manually design some portions of the kernel while deferring the specification of the remaining portions to when an observation of the signal is available. In this sense, the resultant kernel is adaptive to the data observed. We give two examples of this construction technique via the grayscale image up-conversion task where we chose to incorporate the prior assumption that edge contours are smooth. Both examples use a novel local analysis algorithm that summarizes the p-most dominant directions for a given grayscale image patch. The non-stationary properties of these two types of kernels are empirically demonstrated on the Kodak image database that is popular within the image processing research community.

Tensors and tensor decomposition methods are gaining popularity in the signal processing and machine learning literature, and most of the recently proposed tensor decomposition methods are based on the tensor power and alternating least-squares algorithms, which were both originally devised over a decade ago. The algebraic approach for the canonical polyadic (CP) symmetric tensor decomposition problem is an exception. This approach exploits the bijective relationship between symmetric tensors and homogeneous polynomials. The solution of a CP symmetric tensor decomposition problem is a set of p rank-one tensors, where p is fixed. In this thesis, we refer to such a set of tensors as a *rank-one decomposition with cardinality* p. Existing works show that the CP symmetric tensor decomposition problem is non-unique in the general case, so there is no bijective mapping between a rank-one decomposition and a symmetric tensor. However, a proposition in this thesis shows that a particular space of rank-one decompositions, $\mathcal{S}_{\mathrm{E}}$, is isomorphic to a space of moment matrices that are called *quasi-Hankel matrices* in the literature.

Optimization over Riemannian manifolds is an area of optimization literature that is also gaining popu-

larity within the signal processing and machine learning community. Under some settings, one can formulate optimization problems over differentiable manifolds where each point is an equivalence class. Such manifolds are called quotient manifolds. This type of formulation can reduce or eliminate some of the sources of non-identifiability issues for certain optimization problems. An example is the learning of a basis for a subspace by formulating the solution space as a type of quotient manifold called the *Grassmann manifold*, while the conventional formulation is to optimize over a space of full column rank matrices.

The second portion of this thesis is about the development of a general-purpose numerical optimization framework over $\mathcal{S}_{\mathrm{E}}$. A general-purpose numerical optimizer can solve different approximations or regularized versions of the CP decomposition problem, and they can be applied to tensor-related applications that do not use a tensor decomposition formulation. The proposed optimizer uses many concepts from the Riemannian optimization literature. We present a novel formulation of $\mathcal{S}_{\mathrm{E}}$ as an embedded differentiable submanifold of the space of real-valued matrices with full column rank, and as a quotient manifold. Riemannian manifold structures and tangent space projectors are derived as well. The CP symmetric tensor decomposition problem is used to empirically demonstrate that the proposed scheme is indeed a numerical optimization framework over $\mathcal{S}_{\mathrm{E}}$. Future investigations will concentrate on extending the proposed optimization framework to handle decompositions that correspond to non-symmetric tensors.

Contents

List of Figures

List of Algorithms

List of Tables

Notations

General

Symbol	Description
$\mathbf{0}$	The neutral element of some vector space that should be clear from context.
$A \cong B$	This means A is isomorphic to B. A and B can be sets, spaces, or manifolds.
$\pi^{-1}(S)$	$\pi^{-1}(S) := \{x \in X \,\vert\, \pi(x) \in S\} \subset X$ denotes the preimage of S under some map $\pi : X \to Y$, where X and Y are some set such that $S \subset Y$.
Proj_V	An orthogonal projector that maps the input to a vector in the vector space V. The details of this map are always discussed in text when it appears.
$[N]$	The set of consecutive integers $\{1,\, 2, \cdots,\, N\}$. N must be a positive interger.
$(X)_{\mathcal{B},\, i}$	The ith component of the tensor X with respect to the basis $\mathcal{B}$. i could be a multi-index; see Definition 2.26.
$\mathcal{E}(V)$	The elementary basis for V. We write $\mathcal{E}$ when V is clear from context.
$[X]_\sim$	The equivalence class of X under the equivalence relation $\sim$.
$[X]$	The equivalence class of a non-integer X. This is used when the equivalence relation is clear from context.
$\mathbb{Z}_{++}$	The space of strictly positive integers.
$\mathbb{Z}_{+}$	The space of non-negative integers
$L_2(\mathbb{R}^D)$	The space of square-integratable real-valued functions that have the D-dimensional real vector space as its domain.
$\hat{f}(\omega)$	Multi-dimensional Fourier transform of a square-integratable function f.
δ_{ij}	The Kronecker delta function.
$N : C : M$	The set of integers $\big\{N,\, N + C,\, N + 2C, \cdots,\, N + \text{floor}\big(\frac{M-N}{C}\big)C\big\}$. N, M, C are positive integers, with $C,\, N \leq M$.
$N : M$	The set of consecutive integers from N to M. This is the empty set if $N > M$.
$\text{size}(X)$	The tuple that contains the number of elements in each dimension of the multi-dimensional array X.

$\mathbb{R}_{\text{s}}^L$	The set $\underbrace{\mathbb{R} \times \mathbb{R} \times \cdots \times \mathbb{R}}_{L \text{ times}}$. 1-dimensional arrays are treated as an element from this set.
$\mathbb{R}^L$	The vector space induced by $\mathbb{R}_s^L$.
$\mathbb{R}_{\text{s}}^{D_1 \times D_2 \times \cdots \times D_L}$	The set $\underbrace{\mathbb{R}^{D_1} \times \mathbb{R}^{D_2} \times \cdots \times \mathbb{R}^{D_L}}_{L \text{ entries}}$. L-dimensional arrays are treated as an element from this set.
$\mathbb{R}^{D_1 \times D_2 \times \cdots \times D_L}$	The vector space induced by $\mathbb{R}_s^{D_1 \times D_2 \times \cdots \times D_L}$.
S^L	The space $\underbrace{S \times S \times \cdots \times S}_{L \text{ times}}$, where S is a set.
$\{S_i\}_{i=1}^L$	The space $S_1 \times S_2 \times \cdots \times S_L$, where each S_i is a set.
$y_{N:M}$	The ordered subset $\{y_N, y_{N+1}, \cdots, y_M\}$ of the ordered set $\{y_i\}_{i=1}^M$. This is the empty set if $N > M$.
a_{-k}	Identical to the multi-index a, but with the kth element removed.
$\lvert\mathcal{X}\rvert$	Cardinality of the set $\mathcal{X}$.
$\dim(V)$	Dimension of V.
$\lvert n\rvert$	The sum of the entries of the multi-index n.
$\mathbb{P}(S)$	The power set of S.
$\lVert v\rVert_p$	The p-norm of the finite-dimensional vector v.
$f : A \xrightarrow{\sim} B$	f is a map from A to B that is linear with respect to elements in A.

Matrices

Symbol	Description
I_n	The identity matrix of size $n \times n$.
$I_{m \times n}$	The matrix where the first m rows and columns is the identity matrix of size $m \times m$, and the rest of the entires being zero.
$\text{colspan}(A)$	The vector space that is spanned by the set of vector components that are stored in the columns of the matrix A.
$\text{diagm}(\mathcal{A})$	The diagonal matrix of size $\lvert\mathcal{A}\rvert \times \lvert\mathcal{A}\rvert$ that has the ith element in $\mathcal{A}$ as the (i, i)th entry of the matrix. Here, $\mathcal{A}$ must be an ordered set or a tuple of numbers.
$\mathbb{D}_{++}^p$	The set of diagonal matrices of size $p \times p$ with strictly positive numbers on its diagonal.
$\mathbb{V}(D, L, p)$	The space of quasi-Vandermonde matrices with p distinct roots, and maximum degree L. Each root is a D-tuple of real numbers, where the first entry is fixed to 1. Each root cannot be a tuple of all zeros. Defined in Equation 4.12.
$\mathbb{O}(p)$	The space of orthogonal matrices: $Q^T Q = I_p$ if $Q \in \mathbb{O}(p)$.
$\mathbb{K}(p)$	The space of $p \times p$ skew-symmetric matrices: $K^T + K = 0_p$ if $K \in \mathbb{K}(p)$.
$\mathbb{S}_{++}^n$	The space of symmetric positive-definite matrices of size $n \times n$.
$\mathbb{S}^n$	The space of symmetric matrices of size $n \times n$.
$A^\dagger$	Left pseudoinverse of A.
$A^{\dagger,T}$	The transpose of $A^\dagger$.

$\mathtt{H}(\mathbb{B}_\succ, \mathfrak{z}, c)$	The quasi-Hankel matrix with roots specified by $\mathfrak{z}$, weights c, and monomial exponents specified by the ordered set $\mathbb{B}_\succ$.
$\mathtt{H}$	A quasi-Hankel matrix.
$\mathtt{V}(\mathbb{B}_\succ, \mathfrak{z})$	The quasi-Vandermonde matrix with roots specified by $\mathfrak{z}$, monomial exponents specified by the ordered set $\mathbb{B}_\succ$.
$\mathtt{V}$	A quasi-Vandermonde matrix.
$\circ_{\mathrm{F}}$	Frobenius product; see Equation 2.50.

Manifolds and $(p\,, q)$-Tensors

Symbol	Description
∂_i	The chart-induced basis $\partial_i := \frac{\partial}{\partial x^i}$, where x is the chart map.
dx^i	The chart-induced dual basis, where x is the chart map.
x^i	The ith component of the chart map x.
$S^{\otimes L}$	The tensor (or vector space) $\underbrace{S \otimes S \otimes \cdots \otimes S}_{L \text{ times}}$, where S is a tensor (or vector space).
$\{S_i\}_{i=1}^{\otimes L}$	The tensor (or vector space) $S_1 \otimes S_2 \otimes \cdots \otimes S_L$, where each S_i is a tensor (or vector space).
$\mathcal{B}^{\otimes p} \otimes (\mathcal{B}^*)^{\otimes q}$	The tensor product basis for the tensor space $\mathcal{T}^p_q V$ if $\mathcal{B}$ is a basis for the vector space V, and $\mathcal{B}^*$ is the associated dual basis for the dual vector space V^*.
$\Gamma(E)$	The space of all sections of the bundle where E is the total space.
$\Gamma^k(E)$	The space of k-times differentiable sections of the bundle where E is the total space.
$\mathcal{T}^p_q V$	The space of $(p,\, q)$-tensors over the vector space V.

Symmetric Tensors and Polynomials

All polynomials and monomial spaces denoted here are constructed by monomials that have non-negative integer exponents.

Symbol	Description
$\mathbb{A}(D,\, L)$	The space of multi-indices for the distinct components of a D-dimensional, Lth-order symmetric tensor. This is the same as the space of multi-indices that correspond to the exponents of a D-variate, Lth-degree homogeneous polynomial. See Equation 2.3.
$\mathbb{A}(D,\, L)_{\succ_{\mathrm{V}}}$	An ordering of $\mathbb{A}(D,\, L)$ from Definition 2.123.
$\mathcal{J}(D,\, L)$	The space of rank-one tensors with construction detailed in Example 2.28.
$\mathbb{M}_D$	The space of D-variate monomials.
$\mathbb{M}_{D,\,L}$	The space of D-variate monomials of degrees that are at most L.
$\mathbb{M}_{D,\,L,\,\mathrm{hom}}$	The space of D-variate monomials of degree L.
$\mathbb{P}_{D,\,L}$	The space of D-variate, Lth-degree polynomials.
$\mathbb{H}_{D,\,L}$	The space of D-variate, Lth-degree homogeneous polynomials.

$\mathcal{S}(D,\, L)$	The space of D-dimensional, Lth-order, real-valued symmetric tensors that can be written as Equation 2.88 for some p.
$\mathcal{S}_{\mathrm{E}}(D,\, L,\, p)$	The subset of $\mathcal{S}(D,\, L)$ that is defined in Equation 1.1.
$\mathcal{B}^{\otimes L}$	The tensor product basis for the tensor space $\mathcal{S}(D,\, L)$ if $\mathcal{B}$ is a basis for $\mathbb{R}^D$.
$\mathfrak{Z}$	The space of all sets of p D-tuples of real numbers where the first entry is one. The tuples are not multiples of each other. See Equation 4.21 for details.
$\mathcal{V}$	See Equation 4.15.
$\mathcal{P}$	See Equation 4.30.
$\mathcal{N}$	See Equation 4.35.

CHAPTER 1

Introduction

This thesis is about the modelling of real-valued functions on $\mathbb{R}^D$ from a set of samples, and a general-purpose optimization scheme over a space of variables that can be each be mapped to a symmetric tensor. In this introductory chapter, we briefly discuss each of these works, and conclude with an overview of the structure of this thesis.

1.1 Construction of Non-Stationary Kernels

Models are approximations to real world phenomena, and a suitable model for a given application provides enough details to produce a good solution to the given application. There are different philosophies on how complex a model should be, and complexity is often measured in terms of how many *parameters* (or *degrees-of-freedom*) a model requires for its complete characterization. One such philosophy is *Occam's razor*, which prefers the simplest model that produces a good solution to a given application. This preference is often formulated as choosing the model with the least amount of parameters that produces a good solution to a given application. During model construction, *parametric models* have their number of parameters given to them. *Non-parametric models* are methods that specify a family $\mathcal{F}$ of parametric models where each model $m \in \mathcal{F}$ has an arbitrary finite number of parameters and satisfies some given conditions. One defers the choice of a $m \in \mathcal{F}$ until an observation is available. It follows that the specifications of non-parametric models are invariant to the cardinality of the parameter set. In a practical implementation of a non-parametric model, one still needs to upper bound the maximum number of parameters allowed to avoid memory storage issues.

Let D be an arbitrary finite positive integer. One example of a non-parametric model is to treat any D-dimensional observation as a realization of the random variables $X_{1:D} := \{X_d\}_{d=1}^{D}$, where: 1) $p(X_{1:D})$ is the finite-dimensional distribution of the stochastic process $X_t := X(t)$ at time $t = 1, 2, \cdots, D$, and 2) $p(X_{1:D})$ has to satisfy some conditions, for all finite positive values of D. An example of such a condition is to force $p(X_{1:D})$ to be a D-variate normal distribution, which implies $X(t)$ is a Gaussian process. This emphasis on specifying models in terms of characteristics that remain invariant for any finite number of parameters makes non-parametric models suitable when a model complexity control scheme is required. This is because one does not have to worry about model characteristics not being consistent across all possible solutions of the set of model parameters, which may have different cardinalities. The stochastic process example is the foundation of the *Gaussian process prior* model [2] from the Bayesian non-parametric research

community. This and other similar methods [3, 4] can be used under a Bayesian inference setup to fit a stochastic process to observed realizations. The Gaussian process prior is attractive because its posterior distribution can be completely characterized by analytic expressions, by using some identities for multivariate normal distributions (e.g., see Sections 4.3.1 and 4.4 of [5]).

The posterior distribution of the mean process of a Gaussian process can also be interpreted as the solution of a *reproducing kernel Hilbert space* (RKHS) regularization problem. This suggests that a RKHS regularization problem can be interpreted as a deterministic counterpart of a Bayesian inference scheme that uses a Gaussian process prior. These regularization problems are based on the representer theorem [6, 7], where the regularizer is specified by a *kernel* function. Since the regularizer is treated as another function in the RKHS setting, it also requires parameters for the scheme to be completely specified. These parameters are commonly referred to as the *hyperparameters* of the model. In this thesis, we only consider function regularization problems that are over real-valued functions that have a domain in $\mathbb{R}^D$. Under this setting, the kernel function is a function that takes two inputs from $\mathbb{R}^D$ to a real number. *Stationary kernels* only use the displacement information between the two inputs, and they have many computational benefits. However, many problems in the applied sciences deal with phenomena that exhibit non-homogeneous patterns. *Non-stationary kernels* are kernels that do not have this constraint, and are more suitable when non-homogeneous patterns are exhibited across the function's domain.

1.1.1 Open Problems

The specification of a computationally tractable non-stationary kernel is an active area of research. The works [8, 9] are analytical methods for constructing such kernels. They model kernels as linear systems that change white Gaussian noise to some other stochastic process. It is difficult for such methods to incorporate prior knowledge about the function into the constructed kernels because the application at hand may not have a stochastic process interpretation. The works [10, 11, 12, 13] specify the kernel hyperparameters as the solution of another optimization problem, and they suffer from computational issues that are caused by the existence of equivalent solutions. In this thesis, an optimization problem that suffers from *non-identifiability* issues is an optimization problem that has reduced convergence rates or increased computational loads due to the presence of philosophically equivalent solutions. One should keep in mind that the meaning of equivalence is always application-dependent.

1.1.2 Objectives

The primary objective of the first part of this thesis is to devise non-stationary kernels that can 1) capture non-homogeneous patterns across a function's domain, and 2) intuitively incorporate prior knowledge into the constructed kernel. The construction of such kernels must always yield a symmetric and positive-definite kernel, otherwise the representer theorem cannot be used. The hyperparameters of such kernels must not require one to solve numerically-intensive optimization problems, or one that is non-identifiable. For example, an optimization problem over the space of a mixture of N Gaussian functions is a non-identifiable optimization problem, since any permutation of the mixture components of a solution is itself an equivalent solution. However, if one can devise a pre-processing step such that this type of permutation does not yield an equivalent solution, then the optimization problem with this pre-processing step is identifiable. The secondary objective of this work is to give examples of how one can construct a non-stationary kernel so that prior knowledge on the function of interest is incorporated into the resultant kernel. Although this thesis is not about the applications of the Gaussian process prior, the kernel construction method proposed in this thesis can be used in similar Bayesian inference-based estimation schemes.

1.2 Optimization over a Space of Rank-One Decompositions

The second portion of this thesis is about developing a general-purpose optimization framework over variables that map to symmetric tensors. We now briefly review tensors before presenting the nature of these optimization variables.

Tensors can be interpreted as multilinear maps, and they can be represented in component-form as multi-dimensional arrays with respect to an appropriate basis. We only consider tensors with real-valued components in this thesis. Symmetric tensors have the symmetry property where any permutation of its component indices corresponds to the same component value, e.g., $T_{(1,2,3)} = T_{(2,1,3)}$ if T is the component-form of a third order symmetric tensor with respect to some basis. Tensor components and their basis are reviewed in more detail in Section 2.3. Symmetric tensors arise when there are polynomial relationships between the components of a set of vectors [14, 15], when a model utilizes higher-order partial derivatives of a function that satisfy the symmetry condition in the Clairaut-Schwarz theorem [16], or in differential manifold topics such as the Riemannian metric tensor and the Ricci tensor [17].

Example applications of tensors include: computer vision and image processing [18, 19, 20, 21], machine learning [22, 23], signal processing [24, 25, 26], medical imaging [14, 15], and fluorescence spectroscopy [27]. One application of tensors is to pose a dimensionality reduction problem as a tensor decomposition problem. In this thesis, we only consider the *canonical polyadic* (CP) tensor decomposition problem (see [28, 29] for a survey) when we discuss tensor decompositions. It decomposes a tensor into a sum of *rank-one tensors*, which are tensors that can be completely described by a vector; more details are reviewed in Example 2.28. In this thesis, we refer to a set of p rank-one tensors where each tensor can be added with every other tensor in the set (i.e., a set of rank-one tensors of compatible dimension and tensor order) as a *rank-one decomposition with cardinality* p. We do not mention p when it is clear from context, or unimportant to the discussion at hand. The symmetric tensor that corresponds to the sum of these p rank-one tensors is called the *corresponding symmetric tensor of the rank-one decomposition*. The cardinality p here is also called the *canonical polyadic rank* (CP rank) in the literature. It was shown in [30] that the sum operation over these p rank-one tensors is a non-injective map to the space of D-variate, homogeneous polynomials of degree L that have p terms, or equivalently, to the space of D-dimensional, Lth-order, symmetric tensors.

For technical reasons that we briefly discuss in the following subsections, I opted to formulate this task as optimization over a space of rank-one decompositions that are isomorphic to a space of quasi-Hankel matrices. This isomorphism is shown in Proposition 4.1. We go over these reasons in greater detail in Section 4.1. Quasi-Hankel matrices are a subset of moment matrices, and we review them in Sections 2.13.4 and 2.14.5.

1.2.1 Open Problems in the CP Decomposition Problem

References [29, 28] are the only works that I am aware of that exploit the symmetry property of symmetric tensors to perform CP decomposition. Their approach is called the *algebraic approach*, and is known to not scale well for higher dimensions and orders [31, 32]. However, their approach makes it clear as to when symmetric tensor decompositions are unique [29, 30]. Other methods, such as the tensor power method and the alternating least-squares method, do not exploit the symmetry property when they operate on symmetric tensors [33].

Consider a numerical scheme that solves the symmetric tensor decomposition problem where the returned solution is solely treated as an *ordered* set of rank-one tensors. Such a scheme suffers from non-identifiability issues that arise from re-ordering the returned tensors; we refer to any such non-identifiability issues as *permutation-related non-identifiability issues* in this thesis. Philosophically, one should solve for an *unordered* set of rank-one tensors in order to avoid this problem, yet most numerical schemes are formulated without treating a re-ordering of the solution tensors as equivalent solutions. For example, if the vectors

that correspond to the rank-one tensors are represented as columns of a matrix, then any permutation of the columns of that matrix is an equivalent solution. This issue is well-known to the CP decomposition research community [33, 26], and caused some problems in the evaluation of the different numerical schemes for performing CP decomposition (e.g., see Section 5.1 of [34]).

Works such as [35, 36, 37] that focus on applying CP decompositions to real-life problems also report instabilities in the existing CP decomposition algorithms. This is because the CP decomposition is not unique for an arbitrary combination of dimension and tensor orders. A survey of some uniqueness results for the CP decomposition for tensors can be found in the references [33, 37], and we do not review them in this thesis. We focus on symmetric tensors in this thesis, and we discuss their uniqueness issues further in Section 4.1.2.

1.2.2 General-Purpose Optimization vs. the CP Decomposition Problem

We do not focus on solving the CP decomposition problem, because of three reasons. First, the exact CP decomposition problem for symmetric tensors is non-unique for arbitrary dimensions and tensor orders [30]. It should be modified to reduce the effect of the ill-posed nature of the original decomposition problem. Second, it is more realistic in practice to solve for a set of rank-one tensors whose sum approximates the given tensor that is constructed from data, as opposed to solving for an exact decomposition. This is because the data is unlikely to perfectly describe a multilinear model that has a known number (i.e., p) of rank-one tensors as its CP decomposition. Third, tensors require a lot of storage, but a low CP rank tensor only needs to store a rank-one decomposition (although this decomposition is non-unique as we just mentioned). Therefore, many works in the literature (e.g., [35, 36]) have proposed using some form of CP decomposition scheme as a form of compression. These reasons suggest that, for a dimensionality reduction application, a relaxed or regularized version of the CP decomposition problem should be solved instead of an exact decomposition. There are also tensor applications that do not use a tensor decomposition framework: for example, the bioinformatics work [38] uses an optimization scheme over tensors to identify network relationships of graphical models. A general-purpose optimizer over a space of tensors can be used for these problems, but a tensor decomposition algorithm could not.

In this thesis, we approach the task of optimizing symmetric tensors as an optimization over a space of rank-one decompositions of a fixed cardinality p. We denote this space of rank-one decompositions by $\mathcal{S}_{\mathrm{E}}$ throughout this thesis, and we give an explicit formulation of it in Section 1.2.3. We now briefly discuss the motivation behind our decision of working with this setup, and continue this discussion in more detail in Section 4.1. Due to the non-uniqueness of the CP decomposition problem for symmetric tensors [30], $\mathcal{S}_{\mathrm{E}}$ is not isomorphic to a space of symmetric tensors. Proposition 4.1 was created based on Theorem 6.3 from [29], which shows that $\mathcal{S}_{\mathrm{E}}$ is isomorphic to a space of quasi-Hankel matrices when certain conditions are met. This motivates us to work with quasi-Hankel matrices instead of symmetric tensors, which requires us to solve a quasi-Hankel matrix-completion problem when only a symmetric tensor is given. The works [29, 39, 40, 41, 42, 43] discuss methods to accomplish this matrix-completion problem, so we do not solve this problem in this thesis.

The space of quasi-Hankel matrices that corresponds to $\mathcal{S}_{\mathrm{E}}$ is a differentiable manifold, so the proposed optimization framework is an optimization scheme over a differentiable manifold. A type of submanifold called the *quotient manifold* can be used to eliminate permutation-related non-identifiability issues [1, 44].

1.2.2.1 Open Problems in Optimization over Differentiable Manifolds

Although optimization over manifolds is an area that is gaining popularity within the signal processing and machine learning research communities, it is presently still considered a non-standard topic within these communities. Aside from conference workshops and the reference [1], there is a lack of accessible learning aids for a researcher to learn about differentiable manifolds, Riemannian manifold structures, and opti-

mization over manifolds. Riemannian structures are attractive since most convergence results on manifold optimization require the use of them [1]. However, Riemannian structures could be computationally expensive to compute, and there is a need to discover approximations to these structures such that the empirical performance of an optimizer that uses such approximations remains acceptable. An example of such an approximation is the *retraction map* [1, 45], which is an approximation of the *exponential map*. Retractions are maps that take a search direction and a source point on a manifold, and returns another point on the manifold [1]. Since a single evaluation of the exponential map requires solving a systems of differential equations, most of the Riemannian optimization methods in the reference [1] are based on the retraction map for computational reasons. We review retractions in Section 2.8.2.

1.2.3 Objectives

Our goal in the second portion of this thesis is to develop a general-purpose numerical optimization framework over the space of rank-one decompositions $\mathcal{S}_{\text{E}}$, and a solution from this space can be mapped to a symmetric tensor. This is equivalent to an optimization problem over a space of quasi-Hankel matrices when the conditions in Proposition 4.1 are met, which means it can be formulated as an optimization problem over a differentiable manifold.

The primary objectives of the second part of this thesis are 1) to pose $\mathcal{S}_{\text{E}}$ as an embedded submanifold $\mathcal{P}$ and a quotient manifold $\mathcal{M}$, and 2) to devise a numerical optimization scheme over the embedded submanifold $\mathcal{P}$. Given an even order L, dimension D, and CP rank p that satisfies the conditions in Proposition 4.1, the explicit form of the space of rank-one decompositions that we work with is

$$\begin{aligned}\mathcal{S}_{\text{E}}(D,\, L,\, p) = \{(\mathfrak{z},\, c)\}, \qquad & (1.1)\\ \mathfrak{z} \in \left\{z^{(j)} \in \mathbb{R}_{\text{s}}^{D}\right\}_{j=1}^{p},\, \left(z^{(j)}\right)_1 = 1,\, z^{(j)} \neq b z^{(i)},\, \forall i,\, j \in [p],\, \forall b \in \mathbb{R},\\ c \in \mathbb{R}_{++}^{p},\, L \text{ is even.}\end{aligned}$$

This particular space of rank-one decompositions is chosen because it is isomorphic to a space of quasi-Hankel matrices that have a particular diagonal matrix factorization that is useful in deriving $\mathcal{P}$. This factorization is reviewed in the discussion leading up to Equation 2.82. The approach taken in this work is to treat the resultant matrix factors as the embedded submanifold $\mathcal{P}$, and perform numerical optimization over it. Riemannian structures for $\mathcal{P}$ were derived for this purpose, and the most important structure being a *Riemannian metric* $G_{\mathcal{P}}$ for $\mathcal{P}$. Riemannian metrics can be thought of as a collection of inner products that vary *smoothly* (in some sense) across the manifold. We review manifolds and Riemannian metrics in Section 2.8. The quotient manifold $\mathcal{M}$ and some Riemannian structures for it were also derived from the matrix factorization, but optimization over $\mathcal{M}$ is not covered in this thesis because a suitable retraction for $\mathcal{M}$ was not derived. Retractions are essential for numerical optimization over differentiable manifolds [1].

There are two secondary objectives for this part of the thesis. Although optimization over $\mathcal{M}$ is not covered, an approximation to a Riemannian metric on $\mathcal{P}$ that mimics some behaviour from $\mathcal{M}$ was derived in Section 4.5.5. This approximation, denoted by $G_{\diamond}$, is a bilinear form that is not a proper Riemannian metric on $\mathcal{P}$. One of the secondary objectives of this work is to compare the effectiveness of the version of the proposed optimizer that uses $G_{\diamond}$ against the version that uses $G_{\mathcal{P}}$. The motivation for this is because the derived quotient manifold $\mathcal{M}$ treats a rank-one decomposition (i.e., a point on $\mathcal{M}$) as an unordered set of rank-one tensors as opposed to an ordered set of rank-one tensors. This effectively eliminates the permutation-related non-identifiability issues that we discussed earlier. An approximation of this behaviour on the non-quotient manifold $\mathcal{P}$ might reduce the severity of the non-identifiability issues that arises from permutations.

To the best of my knowledge, the use of the manifolds $\mathcal{P}$ and $\mathcal{M}$ to represent a space of rank-one decompositions (i.e., a space where each element is a set of rank-one tensors) is not reported in the existing

literature. The final secondary objective is to provide a detailed derivation of the manifolds, necessary background material, and proper references in this thesis so that this thesis document can be used as a learning aid for those who are interested in learning about differentiable manifolds.

1.3 Contributions

The main contributions of this thesis are:

1. The construction and demonstration of a non-stationary kernel, and the interpretation of the RKHS regularization problem as an instance of the *sparse recovery* method from the signal processing literature.

2. The formulations of $\mathcal{S}_{\mathrm{E}}$ as a differentiable embedded submanifold, and as a quotient manifold.

3. The specification and demonstration of an optimizer over $\mathcal{S}_{\mathrm{E}}$. The CP decomposition problem was used as demonstration.

Two versions of the optimizer over $\mathcal{S}$ are proposed in the third contribution. The first version does not use any quotient space machinery. The second version uses an approximation $G_{\diamond}$ in place of a proper Riemannian metric to partially alleviate non-identifiability issues.

There are some other minor contributions that appear as a by-product of my work on these three contributions, but they require more technical background to understand. We revise our list of contributions in the concluding chapter of this thesis to include these contributions.

1.4 Thesis Organization and Overview

1.4.1 Chapter 2: Background

We review the fundamentals of sets, arrays, and tensors. In this thesis, the component-form of a tensor is treated as a multi-dimensional array, and this is clarified in Chapter 2. We review the theory of RKHS regularization, Riesz-wavelet transform, differentiable manifolds, Riemannian structures, quotient manifolds, and some existing Riemannian optimization methods. We conclude our review with my interpretation of the relationship between rank-one decompositions and homogeneous polynomials. A reference for this relationship is [29], and it leads to the setup of the work done in Chapter 4.

1.4.2 Chapter 3: Adaptive Kernels

In Chapter 3, we present a novel family of non-stationary kernels that we call *adaptive kernels*. To avoid non-identifiability issues in obtaining kernel hyperparameters, one is required to define a *warp map* that induces a sparse representation of the signal of interest. The main contribution of Chapter 3 is Theorem 3.1, which ensures that adaptive kernels are positive-definite. The warp map can be heuristically specified based on the application at hand. In Section 3.4, we demonstrate the effectiveness of two exemplar warp maps for the grayscale image up-conversion problem. We chose the prior assumption that contours along edges in the up-converted image are smooth, and present two warp map construction techniques in Sections 3.4.2 and 3.4.3 that implement this behaviour. One of these construction techniques is based on the Riesz transform for grid-sampled signals [46], so it is not applicable for signals that are not grid-sampled. The focus of this demonstration is not to showcase a competitive grayscale image up-conversion scheme, but to validate that the resultant RKHS regularization problem did in fact incorporate the chosen prior assumption.

1.4.3 Chapter 4: Optimization over a Space of Rank-One Decompositions

The motives behind this portion of the thesis work are explained in greater detail in Section 4.1, and the objectives are reviewed again in Section 4.2. We review similar works in Section 4.3.

Proposed Manifolds and Riemannian Structures

In Section 4.4, we review some concepts and notations from Section 2.14 that are commonly encountered in Chapter 4. In this thesis, the $\cong$ symbol means the left-hand side object is isomorphic to the right-hand side object. In Section 4.5.1, we derive the embedded submanifold $\mathcal{V} \cong \mathcal{S}_{\text{E}}$. The issue with $\mathcal{V}$ is that existing retractions for full-rank matrices (e.g., see Example 2.87) cannot be used on $\mathcal{V}$, so we derive the manifolds $\mathcal{P} \cong \mathcal{N}$ in Section 4.5.2, and the Riemannian metric $G_{\mathcal{P}}$ for $\mathcal{P}$ in Section 4.5.3. All metrics for the derived manifolds in Chapter 4 are inherited from the Euclidean metric (i.e., dot product) on the vector space over the real matrices.

In Section 4.5.4, we derive the quotient manifold formulation of $\mathcal{S}_{\text{E}}$, and denote this manifold by $\mathcal{M}$ throughout Chapter 4. A retraction for $\mathcal{M}$ was not devised in this thesis. It is impossible to devise an inherited metric from $\mathcal{M}$ for the manifold $\mathcal{P}$, which has a retraction. Therefore, in Section 4.5.5, we introduce an approximation $G_{\diamond}$ of a Riemannian metric for $\mathcal{P}$ that uses some of the quotient space machinery that was derived in Section 4.5.4.

Proposed Optimizer

The proposed optimizer over $\mathcal{S}_{\text{E}}$ is actually made of four separate optimizers. Two of them are existing Riemannian optimization methods from [1], which are reviewed in Section 2.12; the conjugate-gradient and the gradient descent methods. One of them is a trust-region optimization method that I modified from [1] so that it also uses the Rendl-Wolkowicz algorithm [47] to solve the trust-region subproblem. We discuss this modification in Section 4.6.1. The last of the four optimizers is a direct-search optimizer that I devised based on an extension of the line-search equation of the Riemannian optimization methods from [1]. This optimizer uses the Nelder-Mead solver to do the actual computation, and is presented in Section 4.6.2. The final proposed optimizer over $\mathcal{P} \cong \mathcal{S}_{\text{E}}$ is represented in Section 4.6.3. There are two versions of this optimizer that are used in the demonstrations. One uses $G_{\mathcal{P}}$ as the Riemannian metric, and the other uses $G_{\diamond}$ as an approximation to the Riemannian metric that is used in the proposed optimizer.

The gradient of the cost function that is used in our demonstrations is derived in Section 4.7.1. In Section 4.7.2.1, we compare the results of the optimization over 3 and 4-dimensional symmetric tensors of orders 4 and 6. Any benchmarks that uses larger orders or dimensions took too long to run. This is expected since the work done in this chapter is based on the algebraic approach to CP decomposition, which does not scale well [31]. In Section 4.7.2.2, we compare the performance from both versions of the proposed optimizer to the alternating least-squares algorithm.

1.4.4 Chapter 5: Conclusion

In Chapter 5, we summarize the results of the work presented in this thesis, and recommend future directions for the proposed work.

1.4.5 Appendix A: Preparation of Reference Images

We describe the preparation of the images that are used for the demonstrations in Chapter 3.

1.4.6 Appendix B: The Multi-Index Set $\mathbb{A}_{\mathbf{de}}(D,\, L) \cong \mathbb{A}(D,\, L)$

The multi-index set $\mathbb{A}(D,\, L)$ is introduced in Equation 2.3, and it is used as a space of integer monomial exponents throughout this thesis. The implementation used in this thesis of a particular ordering of monomials is better described using a space of multi-indices that is isomorphic to $\mathbb{A}(D,\, L)$. This space of indices is denoted by $\mathbb{A}_{\text{de}}(D,\, L)$, and discussed in detail in this appendix.

1.4.7 Appendix C: Synthetic Benchmark Setup

We describe the ground truth rank-one decompositions from $\mathcal{S}_{\text{E}}$ and initial solutions on $\mathcal{P}$ that were used in the benchmarks in Section 4.7.2.

CHAPTER 2

Background

In this review chapter, we discuss the notations and concepts that are frequently used throughout this thesis. We present frequently used notations and terminologies in Section 2.1. The theory of the representer theorem when applied to kernels is reviewed in Section 2.2. We review the basics of multilinear algebra in Section 2.3, and linear maps between vector spaces in Section 2.4. The Riesz transform is reviewed in Section 2.5. At this point, we have reviewed all the background required to read Chapter 3. Manifold-related topics are reviewed from Section 2.6 to 2.12. The remainder of this chapter reviews the relationship between symmetric tensors and homogeneous polynomials. More background material about symmetric tensors and quasi-Vandermonde matrices are reviewed in Chapter 4.

2.1 Preliminaries

We start by discussing our notation for ordered and unordered sets. We then review equivalence relations, and my decision to treat real-valued arrays and vectors from a vector space with field $\mathbb{R}$ as separate objects. We discuss our observation model setup at the end of this section.

2.1.1 Indexed Sets

We write $X = \{x_i\}_{i=1}^N$ if X is an ordered set with N elements, and that it uses integer indices to specify the order. When the indices are no longer integers, it is common practice to use the index set notation. We are motivated to discuss this notation in more detail because it can be used to succinctly express equivalence relations that involve orderings of a set.

Definition 2.1. Index family of sets. Let $\mathcal{I}$ and $\mathcal{X}$ be two unordered sets. An *index family of sets* is a function

$$\begin{aligned} \iota_{\mathcal{I},\mathcal{X}} :& \mathcal{I} \to \mathcal{X} \\ & i \mapsto x \end{aligned}$$

that maps an element from $\mathcal{I}$, called a *label* or *index*, to an element in $\mathcal{X}$. $\mathcal{I}$ is called *the index set of* $\mathcal{X}$. The range of $\iota_{\mathcal{I},\mathcal{X}}$ is denoted range$\{\iota_{\mathcal{I},\mathcal{X}}\}$. When $\iota_{\mathcal{I},\mathcal{X}}$ is the only index family of sets in the discussion at hand, we often use the following shorthand notation:

$$\mathcal{X}_{\mathcal{I}} := \{x_i \in \mathcal{X}\}_{i\in\mathcal{I}} := \text{range}\{\iota_{\mathcal{I},\mathcal{X}}\}.$$

We sometimes treat the finite set of sampling locations as an unordered set $\mathcal{X} \subseteq \mathbb{R}^D$, with an accompanying index family of sets $\mathcal{I}$. In these situations, we say the ordered set $\mathcal{X}_\mathcal{I}$ is an ordered version of $\mathcal{X}$ if the specification of $\mathcal{I}$ does not matter to the discussion at hand.

Definition 2.2. Ordering of a finite set. Let $\mathcal{X}$ be a finite set, and $\mathcal{I}$ be a set of positive integers. An index family of sets $\iota_\mathcal{X} := \iota_{[|\mathcal{X}|],\mathcal{X}}$ is an *ordering of* $\mathcal{X}$ if it is bijective. Since $\mathcal{I}$ is a subset of $\mathbb{Z}$, we have a natural definition of ascending order. We define the notations

$$\{x_i \in \mathcal{X}\}_{i=1}^{|\mathcal{X}|} := \mathcal{X}_{[|\mathcal{X}|]} = \{x_i \in \mathcal{X}\}_{i \in [|\mathcal{X}|]} = \text{range}\{\iota_\mathcal{X}\}$$

to all mean the range of $\iota_\mathcal{X}$. These notations all denote the same ordered set. When $\mathcal{X}$ and its cardinality is clear from context, we write ι to mean $\iota_\mathcal{X}$.

When an index family of sets creates an ordered set $\mathcal{Y}_{[|\mathcal{Y}|]}$ from the unordered set $\mathcal{Y}$, we say $\mathcal{Y}_{[|\mathcal{Y}|]}$ is *an ordered version of* $\mathcal{Y}$. When a set is denoted $\{y_i\}_{i=1}^{|\mathcal{Y}|}$, we do not explicitly specify that it is ordered because it is clear that there is an ordering assigned to the unordered version of $\{y_i\}_{i=1}^{|\mathcal{Y}|}$. The ordering of an unordered set $\mathcal{X}$ is a bijective function that enumerates each element to an integer $i \in [|\mathcal{X}|]$. This allows us to sensibly talk about the ith sample from $\{x_i \in \mathcal{X}\}_{i \in [N]}$, given some specified ordering $\iota_\mathcal{X}$. Our emphasis on modelling sampled observations as an unordered set as opposed to an ordered set is useful when we want to emphasize that a certain result over a set holds regardless of the enumeration scheme used; i.e., when order does not matter. In practice, one can choose an ordering that facilitates implementation advantages on the computational hardware at hand.

Example 2.3. Orderings for 2-dimensional arrays. Consider the case where we wish to label the ith element from a sampled grayscale image that is treated as a 2-dimensional array. We can linearize the 2-dimensional pixel index by doing row-major ordering or column-major ordering. One ordering may be more computationally more efficient than another depending on how the computer stores the image in memory. We discuss column-major ordering for multi-dimensional arrays in Section 2.1.4.

When we define a custom comparison operator $\prec$ to induce an order on a set $\mathcal{C}$, we write $\mathcal{C}_\prec$ to mean the ordered version of $\mathcal{C}$. Unless otherwise specified, the element on the left-hand side of a custom comparison operator is placed in the resultant ordered set before the element on the right-hand side. For example, if $\mathcal{C}$ is a set of real numbers and the comparison operator is the usual $>$ comparison operator for real numbers, then $\mathcal{C}_>$ is a sorted version of $\mathcal{C}$ that is sorted in descending order.

2.1.2 Equivalence Relations and Permutations

Definition 2.4. Equivalence relations and equivalence classes [48]. Consider the set $\mathcal{A}$, and R a subset of $\mathcal{A} \times \mathcal{A}$. If $\forall a, b, c \in \mathcal{A}$,

$$\begin{aligned} (a, a) &\in R \\ (a, b) \in R &\Rightarrow (b, a) \in R \\ (a, b) \in R \text{ and } (b, c) \in R &\Rightarrow (a, c) \in R \end{aligned}$$

holds, then R is called an equivalence relation. We write $a \sim b$ if $(a, b) \in R$. Let $b \in \mathcal{A}$, and call the set

$$[b]_\sim = \{a \in \mathcal{A} \,|\, a \sim b\}$$

the *equivalence class containing b under the equivalence relation $\sim$*. The element b is called the *representative element* of $[b]_\sim$. For any $c \in \mathcal{A}$, this means that either

$$[b]_\sim = [c]_\sim, \text{ or}$$
$$[b]_\sim \cap [c]_\sim = \emptyset,$$

which means $\sim$ induces a partition on $\mathcal{A}$.

Any element $z \in [b]_\sim$ can be a representative element for the equivalence class $[b]_\sim$. If $b \in \mathcal{A}$, and $\mathcal{A}$ is not the set of non-negative integers, we define the shorthand notation $[b] := [b]_\sim$ when the equivalence relationship $\sim$ is clear from context. This does not conflict with our other shorthand notation $[N] = \{1, 2, \cdots, N\}$, since N needs to be a non-negative integer for this shorthand to be used.

Given a set $\mathcal{A}$, a bijective map $\varsigma : \mathcal{A} \to \mathcal{A}$ is called a *permutation map*, and its evaluation at any $a \in \mathcal{A}$ is called a *permutation of a*.

Example 2.5. Let ι and γ be two orderings for the finite set $\mathcal{X}$. Define $\sim$ such that $\iota \sim \gamma$ if $\text{range}\{\iota\} = \varsigma(\text{range}(\gamma))$, for some permutation map ς. The equivalence class of ι is $[\iota]_\sim = \{\gamma \in \mathcal{O}(\mathcal{X})\}$, where $\mathcal{O}(\mathcal{X})$ here denotes the space of all orderings of $\mathcal{X}$.

2.1.3 Multi-Index Notation

In this thesis, we often use multi-indices to index multi-dimensional arrays. A reference for such an indexing scheme can be found in Section 2 from [46]. A D-dimensional multi-index n is a D-tuple of non-negative integers where for each $d \in [D]$, the dth entry n_d of this tuple is a non-negative integer that satisfies $n_d \leq N_d$, for some given N_d. We do not use special mark-up to differentiate a multi-index n from a 1-dimensional integer index when there is no confusion. Given non-negative integers D and N_d, $d \in [D]$, we denote the space of D-dimensional multi-indices that start with zero by $\{0 : N_d\}_{d=1}^D$, and denote the space of D-dimensional multi-indices that start with one by $[N_d]_{d=1}^D$. In the special case when $N = N_d$ for all $d \in [D]$, we use the notations $\{0 : N\}^D$ and $[N]^D$ to denote the space of multi-indices that start with zero and one, respectively.

The following are common operations on multi-indices that we use throughout this thesis:

1. Addition between two indices: this is point-wise addition on each slot, i.e.,
$$n + m := (n_1 + m_1, n_2 + m_2, \cdots, n_D + m_D).$$
2. Sum of an index: $|n| := \sum_{i=1}^D n_i$.
3. Factorial: $n! := \prod_{i=1}^D n_i!$.
4. Exponentiation: If z is a D-tuple of complex numbers, then $z^n := \prod_{i=1}^D z_i^{n_i}$.
5. Multinomial theorem: Let x be a D-tuple of real numbers and $L \in \mathbb{Z}_{++}$. The multinomial theorem can then be expressed as
$$\begin{aligned}\left(\sum_{i=1}^D x_i\right)^L &= \sum_{|a|=L} \frac{L!}{a!} x^a,\ a \in [0 : L]^D, \\ &= \sum_{|a|=L} \binom{L}{a} x^a,\ a \in [0 : L]^D, \qquad (2.1) \\ &= \sum_{b \in [D]^L} \prod_{l=1}^L x_{b_l}, \qquad (2.2)\end{aligned}$$

and we have $\binom{L+D-1}{D-1}$ monomial terms in each of the sums that involve $a \in [0:L]^D$. This implies that although there are D^L monomial terms in the sum that involves $b \in [D]^L$, only $\binom{L+D-1}{D-1}$ of them are distinct. The space in which a from Equation 2.1 resides in is frequently encountered in this thesis, so we define the following notation for this space of multi-indices:

$$\mathbb{A}(D,\, L) := \left\{ a \in [0:L]^D \;\middle|\; |a| = L \right\} \subset [0:L]^D. \tag{2.3}$$

Depending on the context, we use the notation $\binom{\cdot}{\cdot}$ to mean either the binomial coefficient or the multinomial coefficient throughout this thesis. If n and k are non-negative integers, then

$$\binom{n}{k} := \frac{n!}{(n-k)!k!}$$

means the binomial coefficient. When $n \in \mathbb{A}(D,\, L)$, then

$$\binom{L}{n} := \frac{L!}{n_1!n_2!\cdots n_D!} = \frac{L!}{n!}$$

means the multinomial coefficient.

2.1.4 Arrays

In this thesis, we treat arrays of real numbers and vectors from a vector space with field $\mathbb{R}$ as separate objects, and similarly for the space of arrays and vector spaces. This is because the present-day computing framework is based on arrays, and having a dedicated notation for these objects makes it easier to implement the mathematical concepts in this thesis. Another reason is that arrays alone are not equipped with the concept of a basis, while a vector in numerical form requires a pair of objects for representation; a specified basis and its corresponding basis coefficients. In this thesis, we use letters, tuples, or column matrices to represent 1-dimensional arrays. The column matrix representation is used when we need to express a formula that contains 1-dimensional arrays using matrix notation. When D is a finite and non-negative number, we use the symbol $\mathbb{R}_{\mathrm{s}}^D$ to mean the set of 1-dimension arrays of length D, and we use the symbol $\mathbb{R}^D := (\mathbb{R}_{\mathrm{s}}^D,\, +,\, \cdot)$ to denote the vector space, where the vector addition $+$ and scalar-multiplication $\cdot$ here are the per-element addition and multiplication for real numbers, respectively.

We now discuss some data-structure conversion operations that are used in this thesis. The simplest is the *identity map*:

$$\begin{aligned} \mathbf{id} : \mathbb{R}_s^D &\to \mathbb{R}_s^D \\ (x_1,\, x_2,\, \cdots,\, x_D) &\mapsto (x_1,\, x_2,\, \cdots,\, x_D). \end{aligned}$$

We also overload the symbol **id** to mean the map

$$\begin{aligned} \mathbf{id} : \mathbb{R}^D &\to \mathbb{R}_{\mathrm{s}}^D \\ y &\mapsto (y_1,\, y_2,\, \cdots,\, y_D), \end{aligned}$$

where $\{y_i\}_{i=1}^D$ is the set of basis coefficients of y under the elementary basis of $\mathbb{R}^D$. We do not explicitly specify which version of **id** is used when it is clear from context. We also define the reverse of this map as follows

$$\begin{aligned} \mathbf{id}_{\mathrm{v}} : \mathbb{R}_s^D &\to \mathbb{R}^D \\ (y_1,\, y_2,\, \cdots,\, y_D) &\mapsto \sum_{i=1}^{D} y_i e_i, \end{aligned}$$

where $\{e_i\}$ is the elementary basis of $\mathbb{R}^D$. In this thesis we write $\mathcal{E}(V)$ to mean the elementary basis of the vector space V. We omit V when it is unimportant in the discussion at hand.

Consider the finite positive integers L and D_l, $l \in [L]$. The set of L-dimensional arrays of real numbers that has size $D_1 \times D_2 \times \cdots \times D_L$ is denoted as $\mathbb{R}_{\text{s}}^{D_1 \times D_2 \times \cdots \times D_L}$. In this thesis, when we construct an $M \times N$ matrix in an entry-wise manner, we treat it as an element in $\mathbb{R}_{\text{s}}^{M \times N}$. If n is an L-dimensional multi-index, the nth entry is denoted by $(X)_n$. Consider the *column-major ordering map*

$$\begin{aligned} \textbf{colmaj} : \mathbb{R}_{\text{s}}^{D_1 \times D_2 \times \cdots \times D_L} &\to \mathbb{R}_{\text{s}}^{\prod_l D_l} \\ \breve{X} &\mapsto \textbf{colmaj}\Big(\breve{X}\Big), \end{aligned}$$

where the nth entry of $\breve{X}$ is mapped to the kth entry of $\textbf{colmaj}\Big(\breve{X}\Big)$ in a way that satisfies the following:

$$k = 1 + \sum_{l=1}^{L} (n_l - 1) \prod_{i=1}^{l-1} D_i. \tag{2.4}$$

Here, the multi-index $n \in [D_l]_{l=1}^{\times L}$ is used to index $\breve{X}$, and the linear index $k \in \mathbb{Z}_{++}$ is used to index $\textbf{colmaj}\Big(\breve{X}\Big)$. This map is sometimes denoted by $\text{vec}(\cdot)$ in the literature. $\mathbb{R}_{\text{s}}^{D_1 \times D_2 \times \cdots \times D_L}$ can be made into a vector space using per-element addition $+$ and multiplication for real numbers $\cdot$, and we denote this vector space by

$$\mathbb{R}^{D_1 \times D_2 \times \cdots \times D_L} := \big(\mathbb{R}_{\text{s}}^{D_1 \times D_2 \times \cdots \times D_L}, +, \cdot\big).$$

The elementary basis of this vector space $\mathcal{E}(\mathbb{R}^{D_1 \times D_2 \times \cdots \times D_L})$ is the set of arrays that have one entry set to one, and the rest of the entries set to zero. We reuse the symbol **colmaj** to mean the map

$$\begin{aligned} \textbf{colmaj} : \mathbb{R}^{D_1 \times D_2 \times \cdots \times D_L} &\to \mathbb{R}_{s}^{\prod_l D_l} \\ X &\mapsto \textbf{colmaj}\Big(\breve{X}\Big), \end{aligned}$$

where $\breve{X}$ is the multi-dimensional array that stores the basis coefficients of X under the $\mathcal{E}(\mathbb{R}^{D_1 \times D_2 \times \cdots \times D_L})$. We do not explicitly specify which version of **colmaj** is used when it is clear from context.

To convert an appropriately sized 1-dimensional array to a D-dimensional array, we use the *inverse column-major ordering map*, which is defined as

$$\begin{aligned} \textbf{colmaj}^{-1} : \mathbb{R}_{\text{s}}^{\prod_l D_l} &\to \mathbb{R}_{\text{s}}^{D_1 \times D_2 \times \cdots \times D_L} \\ X_c &\mapsto \breve{X}, \end{aligned}$$

where, $\forall n \in [D_l]_{l=1}^{L}$, the nth entry of $\breve{X}$ is mapped to the kth entry of X_c as specified in Equation 2.4.

2.1.5 Restriction of a Function to a Set

Consider a function $f : \mathcal{A} \to \mathbb{R}$, where $\mathcal{A}$ is an arbitrary set, and a subset $\mathcal{U} \subset \mathcal{A}$. The *restriction of f to $\mathcal{U}$* is a new function denoted $f|_{\mathcal{U}} : \mathcal{U} \to \mathbb{R}$. The unordered set $f(\mathcal{U}) = \{f(x)\}_{x \in \mathcal{U}}$ is called *the image of $\mathcal{U}$ under f*. Let $\mathcal{U}_{[|\mathcal{U}|]}$ be an ordered version of $\mathcal{U}$. We extend this notation so that when we write $f\big(\mathcal{U}_{[|\mathcal{U}|]}\big)$, we mean the following ordered set:

$$f\big(\mathcal{U}_{[|\mathcal{U}|]}\big) := \{f(u_i)\}_{i=1}^{|\mathcal{U}|},$$

where u_i is the ith element in $\mathcal{U}_{[|\mathcal{U}|]}$.

Consider now a finite subset $\mathcal{X} \subset \mathcal{A}$, and $N = |\mathcal{X}|$. In this thesis, we often work with the function $f|_{\mathcal{X}}$ under a finite-dimensional vector space setting. We write $\boldsymbol{f}_{\mathcal{X}_{[N]}}$ to mean the vector in $\mathbb{R}^N$ where its ith

component with respect to $\mathcal{E}(\mathbb{R}^N)$ is $f(x_i)$, x_i is the ith entry in $\mathcal{X}_{[N]}$, $\forall i \in [N]$. We call $\boldsymbol{f}_{\mathcal{X}_{[N]}}$ the *vector space version of $f|_{\mathcal{X}}$ under the ordering implied in $\mathcal{X}_{[N]}$*. When $\mathcal{A} = \mathbb{R}^D$, one can think of f as a continuous signal, $f|_{\mathcal{X}}$ as the sampled signal at locations in $\mathcal{X}$, $\boldsymbol{f}_{\mathcal{X}_{[N]}}$ as $f|_{\mathcal{X}}$ but represented as a vector in a vector space, and $\left(\boldsymbol{f}_{\mathcal{X}_{[N]}}\right)_{\mathcal{E}}$ or $\left(f|_{\mathcal{X}_{[N]}}\right)$ the 1-dimensional array that contains the values of $f\left(\mathcal{X}_{[N]}\right)$.

When the elements of $\mathcal{X}$ form a D-dimensional grid that has N_i number of samples in the ith dimension, $i \in [D]$, we say $\mathcal{X}$ is *on a grid*, and $f|_{\mathcal{X}}$ is *sampled on a grid*. In this case, we want $\boldsymbol{f}_{\mathcal{X}_{[N]}}$ to be a vector in $\mathbb{R}^{N_1 \times N_2 \times \cdots \times N_D}$ instead of a vector in $\mathbb{R}^{\prod_i N_i}$, so that the grid structure is preserved. Therefore, when $\mathcal{X}$ is explicitly stated to be on a grid, we write $\boldsymbol{f}_{\mathcal{X}_{[N]}}$ to be the vector in $V = \mathbb{R}^{N_1 \times N_2 \times \cdots \times N_D}$. Its set of basis coefficients with respect to $\mathcal{E}(V)$ is

$$\{f(x_a)\}_{a \in \{N_i\}_{i=1}^D},$$

where a is a multi-index.

2.1.6 Observation Models

We use the term *signal* instead of the term *function* when we are emphasizing that a function is used to model a phenomenon in real-life, and that sampling occurs somewhere in the discussion at hand. In this thesis, we focus on signals that have a domain of $\mathbb{R}^D$ and a codomain of $\mathbb{R}$. The space of such signals that are also square-integrable is denoted by $L_2(\mathbb{R}^D)$. Here is the general framework of the observation models that we use in this thesis:

1. We assume there is an underlying continuous function $f : \mathbb{R}^D \to \mathbb{R}$ that generated our observation.

2. During the observation process, f undergoes degradation. We assume that the degradation process operates on the continuous function f to generate the continuous function f_{ob}; i.e., the degradation takes place before the sampling procedure.

3. Given a set of sampling locations $\mathcal{X}$, we formulate the sampling procedure as the restriction of f_{ob} to $\mathcal{X}$. $f_{\text{ob}}|_{\mathcal{X}}$ is called the *observed signal*, and the elements in $\mathcal{X}$ are called the observed inputs.

4. Given a set of different sampling locations $\mathcal{X}_q$, our goal is to recover $\tilde{f}\Big|_{\mathcal{X}_q}$, where $\tilde{f} : \mathbb{R}^D \to \mathbb{R}$ is such that $\tilde{f}\Big|_{\mathcal{X}_q} \approx f|_{\mathcal{X}_q}$ in some specified sense. The elements in $\mathcal{X}_q$ are called *query inputs*, and $\tilde{f}$ is called the *query signal.*

In this thesis, an *observation model of f* refers to how f_{ob} is related to f. The *sample-without-prefiltering observation model* is when $f_{\text{ob}}(x) = f(x), \forall x \in \mathcal{X}$. By a *prefilter*, we mean a low-pass filter that outputs an approximately band-limited signal, with the cut-off frequency at or below the assumed Nyquist frequency of the input signal. The prefilter is also called the *anti-aliasing filter* in some literature. Another observation model is when $f_{\text{ob}}(x) = f(x) + \epsilon$, where ϵ is a realization of a white Gaussian random variable with its standard deviation σ specified as part of the model; we call this the *additive white Gaussian noise observation model*. A popular observation model used in the signal processing community is to prefilter f before sampling; we call this the *sample-with-prefiltering observation model*. If the prefilter causes f_{ob} to be perfectly band-limited with the cut-off frequency at or below the Nyquist frequency, then one can use the Nyquist-Shannon sampling theorem to get an exact reconstruction of the observed signal: i.e., $\tilde{f}\Big|_{\mathcal{X}_q} = f_{\text{ob}}|_{\mathcal{X}_q}$.

Our observation model setup requires the specification of the set of sampling locations $\mathcal{X}$ and their corresponding signal values $f_{\text{ob}}(\mathcal{X})$. We refer to the set $\mathcal{D}$ that contains these two objects as an *observation of f*. We define the following map so that these objects can be expressed in a more compact manner:

Definition 2.6. Projection map for tuple-membered sets. Let S be an ordered set made up of the Cartesian product of N other sets; $S = S_1 \times S_2 \times \cdots \times S_N$. Given a $k \in [N]$, the *kth tuple projection map* is defined as

$$\begin{aligned} \pi_k : S &\to S_k \\ (s_1,\, s_2, \cdots,\, s_N) &\mapsto s_k. \end{aligned}$$

Here is an example that splits an ordered set into two ordered sets that preserves the ordering.

Example 2.7. An observation of f. Suppose $\mathcal{D}_{[N]} := \{(x_i,\, y_i)\}_{i=1}^N$ is an ordered set of N sampling locations $x_i \in \mathbb{R}^D$ and its corresponding signal values $y_i = f(x_i) \in \mathbb{R}$. The decomposition of $\mathcal{D}_{[N]}$ into its tuple members are

$$\begin{aligned} \mathcal{X}_{[N]} &:= \pi_1\big(\mathcal{D}_{[N]}\big) = \{x_i\}_{i=1}^N, \text{ and} \\ \mathcal{Y}_{[N]} &:= \pi_2\big(\mathcal{D}_{[N]}\big) = \{y_i\}_{i=1}^N. \end{aligned}$$

Note that the order is preserved; i.e., the ith element of $\pi_2\big(\mathcal{D}_{[N]}\big)$ is the observation at the sampling location specified by the ith element of $\pi_1\big(\mathcal{D}_{[N]}\big)$.

2.2 Reproducing Kernel Hilbert Space

Our presentation here is based on Sections 2.2.1 and 2.2.3 of [49]. We focus on kernels that take inputs from a vector space with field $\mathbb{R}$.

Definition 2.8. Symmetric and positive-definite kernel functions. A two-argument, bounded, and continuous function $k : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$ is a *kernel function in D dimensions*. k is a *symmetric kernel* if the order of the arguments do not matter, i.e., $k(x,\, t) = k(t,\, x)$, $\forall x,\, t \in \mathbb{R}^D$. k is a *positive-definite kernel* if and only if

$$\sum_{i=1}^N \sum_{j=1}^N y_i k(x_i,\, x_j) y_j > 0$$

for each bounded $x_i \in \mathbb{R}^D$ and each bounded $y_i \in \mathbb{R}$, $\forall i,\, j \in [N]$, for any finite and positive integer N.

The parameters that are required to specify a kernel function are called the *hyperparameters* of the kernel function. The definition of a positive-definite kernel is equivalent to requiring the following family of Gramian matrices to be positive-definite:

Definition 2.9. Kernel matrix. Let $\mathcal{X} \subset \mathbb{R}^D$ be a finite unordered subset, and denote by $\mathcal{O}(\mathcal{X})$ the space of all orderings of $\mathcal{X}$. Define the *kernel matrix map*

$$\begin{aligned} K_{\mathcal{X},\,k} :& \mathcal{O}(\mathcal{X}) \to \mathbb{R}_s^{|\mathcal{X}|\times|\mathcal{X}|} \\ & \iota \mapsto K_{\mathcal{X}}(\iota) \end{aligned}$$

in an entry-wise fashion:

$$\big(K_{\mathcal{X},\,k}(\iota)\big)_{(i,\,j)} = k(x_i,\, x_j),$$

where $\{x_i \in \mathbb{R}^D\}_{i=1}^{|\mathcal{X}|}$ is the range of the ordering $\iota \in \mathcal{O}(\mathcal{X})$. k is a symmetric positive-definite kernel if and only if $K_{\mathcal{X},\,k}(\iota)$ is a symmetric positive-definite matrix, for any finite non-empty subset $\mathcal{X} \subset \mathbb{R}^D$ and for any

ordering $\iota \in \mathcal{O}(\mathcal{X})$. We call $K_{\mathcal{X},k}(\iota)$ a *kernel matrix of kernel k, on the set $\mathcal{X}$, under the ordering ι*. We use the short-hand notation $K_{\mathcal{X}_{[|\mathcal{X}|]},k} := K_{\mathcal{X},k}(\iota)$ when $\mathcal{X}_{[|\mathcal{X}|]}$ is defined in text to be the ordered version of X, i.e., $\mathcal{X}_{[|\mathcal{X}|]} := \mathcal{X}_\iota$. We also use this notation when it is clear that $\mathcal{X}_{[|\mathcal{X}|]}$ is an ordered set, and that its ordering does not matter to the discussion at hand.

This definition of a kernel matrix ensures that it is symmetric if $k(\cdot,\cdot)$ is symmetric, which implies the rows and columns of K correspond to points in $\{x_i \in \mathbb{R}^D\}_{i=1}^{|\mathcal{X}|}$. In this thesis, we do not specify the ordering of a kernel matrix when it does not affect the outcome of the discussion at hand.

A *Hilbert space* is a vector space equipped with an inner product. We consider infinite-dimensional Hilbert spaces that have elements that are continuous functions. Symmetric positive-definite kernels are related to infinite-dimensional Hilbert spaces in the following way:

Definition 2.10. (Definition 2.9 of [49]) Reproducing Kernel Hilbert Space. Let $\mathcal{H}$ be a Hilbert space of functions $f : \mathbb{R}^D \to \mathbb{R}$. $\mathcal{H}$ is a *reproducing kernel Hilbert space* (RKHS) endowed with inner product $\langle\cdot,\cdot\rangle$ if there exists a symmetric positive-definite kernel function $k : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$ with the following properties:

1. $\langle f,\, k(x,\,\cdot)\rangle = f(x), \forall f \in \mathcal{H}$; in particular, $\langle k(x,\,\cdot),\, k(t,\,\cdot)\rangle = k(x,\,t), \forall x,\, t \in \mathcal{X}$.
2. $\mathcal{H}$ is obtained by adding the limits of all Cauchy sequences of the pre-Hilbert space

$$\mathcal{H}_0 = \text{span}\,\{k(x,\,\cdot)\,|\,x \in \mathcal{X}\}\,.$$

The RKHS $\mathcal{H}$ uniquely determines k, which is called the *reproducing kernel* of $\mathcal{H}$. The norm induced by the inner product is denoted by $\|\cdot\|_{\mathcal{H}}$.

The following theorem makes the RKHS an attractive solution space for regularization problems over continuous functions:

Theorem 2.11. *(Theorem 4.2 of [49]) Representer Theorem. Let $\Omega : [0,\,\infty) \to \mathbb{R}$ be a strictly monotonic increasing function, $c : (\mathbb{R}^D \times \mathbb{R} \times \mathbb{R})^N \to \mathbb{R}\cup\{\infty\}$ an arbitrary cost function, $\mathcal{H}$ a RKHS with reproducing kernel k, $\{x_i\}_{i=1}^N$ a set of N vectors of $\mathbb{R}^D$, and $\{y_i\}_{i=1}^N$ a set of N real numbers. Then each minimizer $f \in \mathcal{H}$ of the objective function*

$$c((x_1,\,y_1,\,f(x_1)),,\ldots,(x_N,\,y_N,\,f(x_N))) + \Omega(\|f\|_{\mathcal{H}})$$

admits a representation of the form

$$f(x) = \sum_{i=1}^{N} \alpha_i k(x_i,\,x). \tag{2.5}$$

Proof. See [49]. □

Unfortunately, this theorem alone does not tell us how to solve for $\{\alpha_i\}$. Most results in literature are for the case when Ω is a quadratic function with a positive scaling factor. We present a reworded result from [7], which was originally for the case when f is vector-valued. Since we only deal with $\mathbb{R}$-valued functions in this thesis, their proof covers our reworded theorem.

Theorem 2.12. *RKHS regularization problem (modified from Theorem 4 of [7] for the case when f is $\mathbb{R}$-valued). Consider the objective function $E(f) = \sum_{i=1}^N \|y_i - f(x_i)\|^2 + \mu\|f\|^2$, where $\mu \geq 0$ is a fixed constant, $x_i \in \mathbb{R}^D$, and $y_i \in \mathbb{R}$. Let $\tilde{f}$ be the unique minimizer of E over the solution space $\mathcal{H}$ that is a RKHS with reproducing kernel k. Then $\tilde{f}(\cdot) = \sum_{i=1}^N k(x_i,\,\cdot)\alpha_i$, and the coefficients $\{\alpha_i\} \in \mathbb{R}$ are the unique solution of the linear equations*

$$\sum_{l=1}^{N}(k(x_i,\, x_l) + \mu\delta_{il})\alpha_l = y_i. \tag{2.6}$$

Proof. See [7]. □

Recall from Section 2.1.6 that our goal in modelling a continuous signal f is so that we can estimate the values of f at some query locations that collectively make up the set $\mathcal{X}_q$. The RKHS regularization problem can be used to obtain an approximation for $f(\mathcal{X}_q)$ given any finite set $\mathcal{X}_q$, and we call this estimation approach the *RKHS regularization framework*. Let $\mathcal{H}$ be a RKHS for real-valued functions with domain $\mathbb{R}^D$. Consider $f_{\text{ob}}|_{\mathcal{X}}$, an observed signal of $f \in \mathcal{H}$, where the degradation is in the form of additive white Gaussian noise of a given standard deviation σ. Denote by $\mathcal{D} \in (\mathbb{R}^D, \mathbb{R})^N$ the observation of f at $\mathcal{X}$. Given an ordering ι for $\mathcal{D}$, we make $\mathcal{D}$ into an ordered set $\mathcal{D}_{[N]}$. We use the first tuple projection map π_1 to split the ordered set $\mathcal{D}_{[N]}$ into the ordered set of sampling locations $\mathcal{X}_{[N]} = \{x_i\}_{i=1}^N$, and similarly with π_2 for the corresponding observations $\mathcal{Y}_{[N]} = \{y_i\}_{i=1}^N$ at those locations. In discussions that involve the RKHS regularization framework, this construction of $\mathcal{X}_{[N]}$ and $\mathcal{Y}_{[N]}$ is implied when we reference an ordered version of $\mathcal{X}$ and $\mathcal{Y}$.

If we solve the regularization problem in Theorem 2.12 under this setup, we get a solution $\tilde{f}$ that is the sum of N kernel evaluations (see Equation 2.5). It is clear that the solution increases in complexity as the number of sampling locations N increases. The RKHS regularization framework is thus an implementation of the non-parametric modelling strategy described in the introductory chapter of this thesis.

Remark 2.13. An extension of the RKHS regularization framework to the estimation of stochastic processes is to formulate the estimation approach as a Bayesian inference. The prior over the stochastic process is typically a Gaussian process, and the kernel matrix as defined in Definition 2.9 is set as this prior's correlation matrix. The prior's mean process is usually taken to be the zero function, but other functions can be used. Under this scenario, μ from Theorem 2.12 can be interpreted as the variance of the additive white Gaussian noise observation model [2]. Equation 2.6 shows that the optimal solution is allowed to deviate more from the observed signal values if there is more additive uncertainty (i.e., larger regularization constant μ). Kernels are also used in non-Gaussian prior processes such as elliptical stochastic processes and copula processes [2, 3, 4].

Remark 2.13 suggests that the additive white Gaussian noise observation model is a suitable signal acquisition model for the RKHS regularization problem. The sample-without-prefiltering model is a special case of the sample with additive white noise model when we specify $\mu \to 0$. For a robust practical implementation of the RKHS regularization problem that uses this observation model, I believe it is a good idea to add a small μ in case the kernel matrix is numerically close to being singular.

When the observation model is neither the additive white Gaussian noise model nor the sample-without-prefiltering model, I believe the performance of the RKHS regularization framework depends on whether one can use the information in $\mathcal{D}$ to 1) specify a kernel function k that sufficiently describes f via Equation 2.5, and 2) replace the set $\{y_i\}$ in Theorem 2.12 with another set of real numbers, $\{w_i\}_{i=1}^N$, such that w_i is more similar to $f(x_i)$ than the observed y_i. These two tasks depends on the nature and severity of the degradation in the observation model, and their solutions are likely to be application-dependent.

2.2.1 Stationary and Isotropic Kernels

We only consider the Euclidean distance function for use with isotropic kernels in this thesis.

Definition 2.14. Isotropic kernels and univariate radial functions. An *isotropic* kernel k_{iso} is a family of kernels that is specified by a *univariate radial function* $h : \mathbb{R}_+ \to \mathbb{R}$. An *isotropic kernel in D dimensions* is denoted $k_{D\text{-iso}}$, and is also called the *instantiation of k_{iso} in D dimensions*. It is defined point-wise by

$$k(x_i, x_j) = h\big(\|x_i - x_j\|_2\big), \qquad \forall x_i, x_j \in \mathbb{R}^D. \tag{2.7}$$

From Remark 2.13, the kernel matrix can be interpreted as the correlation matrix of a stochastic process prior under a Bayesian inference framework. Under this interpretation, it makes sense to call a kernel that stays invariant when its inputs are both shifted by the same displacement vector a *stationary kernel*, since such a stochastic process is wide-sense stationary. Bochner's theorem (see Theorem 4.1 of [2]) is sometimes used to place constraints on the space of possible stationary kernels, since it provides a certain result about the Fourier transform of the covariance functions for wide-sense stationary stochastic processes. We do not go into the details because this approach is not used in this thesis.

The following isotropic kernel is an isotropic Gaussian function:

Example 2.15. Square exponential kernel. The square exponential kernel with *hyperparameter* s is specified by

$$h(\tau) = \exp\left\{-\frac{\tau^2}{2s^2}\right\}, \forall \tau \in \mathbb{R}_+.$$

An isotropic kernel is not necessarily stationary; a kernel k such that $k(x_0, \cdot)$ is an isotropic Gaussian function with bandwidth that varies according to x_0 is a non-stationary kernel. The kernel from Example 2.15 has infinite support, so its kernel matrix is known to be ill-conditioned [50] when the number of samples in the input set $\mathcal{X}$ is large.

Kernels with compact support produce sparse kernel matrices. Depending on the level of sparsity, this may have some computational benefits. Wendland summarized in [51] many results concerning the positive-definite property for univariate radial functions. One important result is that a given compactly supported isotropic kernel k_{iso} cannot instantiate a positive-definite kernel function on an arbitrary dimension $D \in \mathbb{Z}_{++}$, but $k_{D\text{-iso}}$ could be a positive-definite kernel function for some fixed D (see Theorem 9.2 and Corollary 9.3 of [51]). If $k_{D\text{-iso}}$ is not a positive-definite kernel function, then $k_{M\text{-iso}}$ cannot be a positive-definite kernel function, $\forall M > D$ (see Section 9.1 of [51]). Table 9.1 of [51] contains some compactly supported positive-definite kernel functions for dimensions up to 5, and more general constructions are described in Chapter 9 of [51].

Definition 2.16. Truncated power function. The following is the truncated power function $(1 - \cdot)_+^n$ of degree $n \in \mathbb{Z}_{++}$:

$$(1 - \tau)_+^n := \begin{cases} (1 - \tau)^n & \text{if } 1 - \tau > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{2.8}$$

Example 2.17. Compactly supported spline kernel (modified from Table 9.1 of [51]). We modified a kernel from Table 9.1 of [51] such that τ was replaced by $a\tau$, where $a \in \mathbb{R}_{++}$ is a hyperparameter. The original kernel from that table is a positive-definite kernel for dimensions up to 3, and is continuously differentiable up to the 4th order. Our modification does not change these results, and still preserves the compact support property. Our version of this kernel is specified by

$$h(\tau) = \frac{1}{3}(1 - a\tau)_+^6\Big(35(a\tau)^2 + 18a\tau + 3\Big), \ \forall \tau \in \mathbb{R}_+. \tag{2.9}$$

Example 2.17 is used in the grayscale image up-conversion demonstration in Section 3.4, because its original version in Table 9.1 of [51] is one of the kernels there that has the lowest differentiability order, and it has a

low computational complexity relative to the others in that table. Kernels with a high differentiability order is unsuitable for modelling signals that have more power at high frequencies (see Section 4.3 of [2]), so they are not used in our demonstrations.

2.2.2 Scaling the RKHS Regularization Framework

Let $\mathcal{X}$ be the set of observed inputs in a RKHS regularization framework, and $N = |\mathcal{X}|$. The RKHS regularization problem requires us to solve a linear system with N equations and N unknowns in order to obtain an approximation $\tilde{f}\Big|_{\mathcal{X}_q}$ for any set of query inputs $\mathcal{X}_q$. For applications like digital image processing where the number of pixels in the input image is very large, a direct application of the RKHS framework that treats the entire input image as the observed signal $f_{\text{ob}}|_{\mathcal{X}}$ is computationally infeasible. One solution is to approximately solve the linear system by using only a subset of the observation $\mathcal{D} = \{x_i \in \mathcal{X}_{[N]},\ f_{\text{ob}}(x_i)\}$. In digital image processing, this implies using only some of the pixels in the input image for the estimation of $\tilde{f}\Big|_{\mathcal{X}_q}$. I took a different approach that is based on common patch-based image processing methods. We present this solution here, and focus on the more important contributions in Chapter 3.

We segment the domain of f into smaller overlapping sets, then apply the RKHS regularization framework to each of these sets. Instead of solving one large RKHS regularization problem, we solve many smaller RKHS regularization problems that each use a small subset of observations.

Definition 2.18. Query regions, local models, and local observations. Let $\mathcal{H}$ be a RKHS for real-valued functions with domain $\mathbb{R}^D$, $f \in \mathcal{H}$ be the non-degraded signal, and $\mathcal{U}$ be a partition of $\mathbb{R}^D$. An element of $\mathcal{U}$ is called a *query region*, and the RKHS framework for f over a query region is called a *local model*. Let $\mathcal{D}$ be an observation that corresponds to $f_{\text{ob}}|_{\mathcal{X}}$. Given a method to specify the neighbourhoods of each query region's border, the *local observation* of a local model is a restriction of $\mathcal{D}$ to sampling locations that are 1) within the local model's query region, or 2) within the neighbourhood of the local model's query region's border.

This definition implies that a query input for a local model must be an element of its query region. The set of all query regions is defined to be a partition of the domain of f, but the set of all local observations does not have to be a partition of $\mathcal{D}$; i.e., the observed image patches can have overlaps. This is because a point in the domain of f could be in the neighbourhood of a query region, but not in the query region. We now describe a particular setup of these ideas when the inputs in $\mathcal{X}$ are on a grid.

2.2.2.1 Patch-RKHS regularization framework

Let the domain of f be measured in units of the grid spacing, e.g., pixels are a valid choice when f is a grayscale image. The portion of $\mathbb{R}^D$ that is covered by the sampling locations in $\mathcal{X}$ is called the *signal canvas of* $f_{\text{ob}}|_{\mathcal{X}}$ in this thesis, and denoted by $\text{canvas}(\mathcal{X})$.

The demonstration for Chapter 3 has the setup where the observed inputs in $\mathcal{X}$ are on a uniform grid and the query points in $\mathcal{X}_q$ all lie inside or on the boundary of $\text{canvas}(\mathcal{X})$. Since we do not care about the value of f at inputs that are outside of the signal canvas, we can model the set $\mathcal{A} = \mathbb{R}^D \setminus \text{canvas}(\mathcal{X})$ as a single query region where we force $f(\mathcal{A}) := \{c\}$, for some arbitrary $c \in \mathbb{R}$. We can use rectangular patches to partition $\text{canvas}(\mathcal{X})$ as shown in Figure 2.1. This partitioning scheme for the query regions is specified by the local observations, which in turn requires four types of information to be specified; the *maximum rectangular patch size* $m \times n$, the *position of the first patch*, the *choice of traversing the first or last dimension of the grid in* $\mathcal{X}$, and the *overlap distance* p. Each are measured in terms of the spacing of the uniform sampling grid. Each local observation is generated by a sequential scheme where it tries to make each patch as large as possible without violating these four pieces of given information. We then generate each query region by

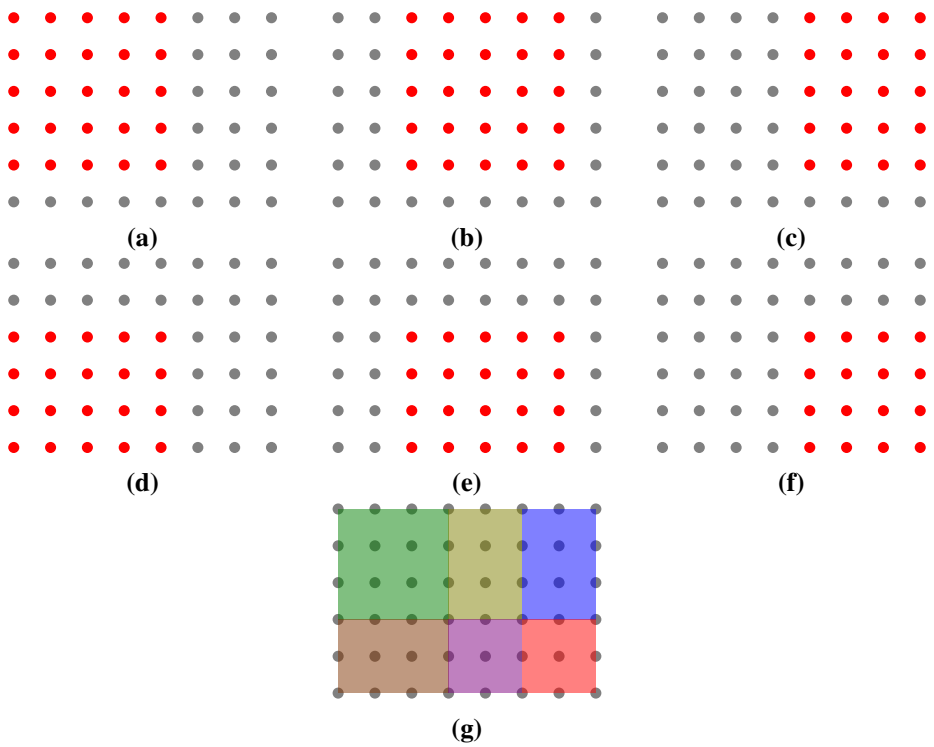


Figure 2.1: The partition system from Example 2.19 for f. The following are sampling locations for local models that have origins at: (a) $(1,\ 1)$, (b) $(1,\ 3)$, (c) $(1,\ 5)$, (d) $(3,\ 1)$, (e) $(3,\ 3)$, and (f) $(3,\ 5)$. (g) The query regions for each local model. The local model for the query region $\mathbb{R}^2 \setminus \text{canvas}(\mathcal{X})$ is not shown.

reducing the region covered by its local observations such that the overlapping regions are split evenly with its neighbours.

Example 2.19. Patch-RKHS regularization framework for a grayscale image. We us the term *image canvas* instead of signal canvas here, and the length of one grid spacing unit is defined to be a pixel. Each local model is called a patch in this example. Figure 2.1 illustrates the setup here for an observed image that has 6×8 pixels, an overlap of $p = 3$ pixels, and a maximum patch size of $m \times n = 5 \times 5$ pixels. The origin of the first patch is positioned at the origin of the sampling grid, which is taken to be the top-left corner of the image canvas. Each of the 6 query regions are represented as the different coloured rectangular patches in Figure 2.1(g), and the gray dots are the sampling locations from $\mathcal{X}$. The rest of the subfigures each describe the sampling locations in $\mathcal{X}$ that are used by a local observation (red dots), and the unused sampling locations in $\mathcal{X}$ with respect to the corresponding local model of that subfigure (gray dots).

Since $\mathcal{U}$ is a partition of $\mathbb{R}^D$, there is no overlap for the query region in Figure 2.1(g). This allows us to solve all the RKHS regularization problems in parallel. A tie-breaking strategy needs to be specified if there are query points that lie on the boundary between two query regions. However, the solution of the RKHS regularization problem is discontinuous at the query region borders. This is noticeable when no overlap between the local observations is used (i.e., when $p = 0$), as shown in Figure 2.2. Since our query regions are rectangular, we can reduce the severity of the border artefacts by increasing the overlap between neighbouring local observations. As shown in Figures 2.2 and 2.3, the border artefacts become less noticeable as the overlap parameter is increased. These two figures were generated by the grayscale up-conversion demonstration from Section 3.4, but with an overlap of 0 and 1 pixels. An overlap of 4 pixels was used for the results presented in Section 3.4.

Figure 2.2: The RKHS regularization solution for no overlapping local observations.

Figure 2.3: The RKHS regularization solution for local observations that have 1 overlapping pixel with each neighbour.

2.3 Multilinear Algebra

We mentioned that we treat matrices as 2-dimensional arrays in Section 2.1.4, but many matrix theories require the interpretation of a matrix as a linear map between vector spaces, or a bilinear map from two vector spaces to a field. In this thesis, we treat the matrices as an array that numerically represents these types of objects once a basis or coordinate system is given, but we often use the coordinate-free version of such objects in our discussions. We are only interested in the finite-dimensional case and vector spaces with field $\mathbb{R}$, which has the property that each such space is isomorphic to $\mathbb{R}^D$ for some finite positive integer D. Vector space isomorphisms are reviewed in Definition 2.31. Objects such as a function in the induced RKHS of a symmetric positive-definite kernel are excluded from our review in this section because we focus on finite-dimensional vector spaces here. The books [52, 53] are references that present multilinear algebra topics in a similar fashion as our presentation here.

2.3.1 Finite-Dimensional Multilinear Maps over V and V^*

Given the vectors spaces V and W, we denote $\text{Hom}(V, W)$ to be the space of linear maps from V to W.

Definition 2.20. The dual space. If $V = \mathbb{R}^N$ is a N-dimensional vector space with field $\mathbb{R}$, then its dual space is $V^* := \text{Hom}(V, \mathbb{R})$. $\omega \in V^*$ is called a *covector of* V or a *functional*.

In the finite-dimensional setting, a vector space V and its dual V^* are both vector spaces of the same dimension. Given a basis $\mathcal{B}$ for V, this thesis uses a particular basis for V^* that is related to $\mathcal{B}$ in the following way:

Definition 2.21. Associated dual basis. Let $\mathcal{B} = \{b_i\}_{i=1}^N$ be an ordered basis for $V = \mathbb{R}^N$. The *associated dual basis* for V^*, $\mathcal{B}^* = \{u_i\}_{i=1}^N$, must satisfy $u_j b_i = \delta_{ij}$, $\forall i, j \in [N]$. We write *the dual basis of* V^* to mean *the associated dual basis for* V^* when there is no confusion.

Example 2.22. Row matrices as coordinates to the dual vector space. Let $\mathcal{B} = \{b_i\}_{i=1}^N$ be a basis of $\mathbb{R}^N$, $x \in \mathbb{R}^N$, $\mathcal{U} = \{u_i\}_{i=1}^N$ be the dual basis of $(\mathbb{R}^N)^*$, and $\omega \in (\mathbb{R}^N)^*$. Since ω is a linear map, ω evaluated at x can be written as

$$\omega x = \sum_{i=1}^{N} \omega_i u_i x_i b_i \in \mathbb{R},$$

where $\{x_i\}$ is the set of basis coefficients for x under $\mathcal{B}$, and $\{\omega_i\}$ is the set of basis coefficients for ω under basis $\mathcal{U}$. We can think of this summation as the matrix product of a row matrix containing $\{\omega_i\}$ and a column matrix containing $\{x_i\}$.

Note that when V is the tangent space of a point on a manifold, we use a superscript to index a dual basis or a basis coefficient for a vector on V. We shall review this notation in Section 2.7. We do not use that convention when V is not specified to be such a vector space, and such is the case with all discussions in this section.

The dual vector space is used to define linear maps of the type $V \to \mathbb{R}$. The following is an extension of this idea:

Definition 2.23. (p, q)-Tensors. Let $V = \mathbb{R}^N$ be finite-dimensional. Consider a function f that take p covectors from V^* and q vectors from V as inputs. If, for every $i \in [p + q]$, f is linear when all but the ith input slot is fixed, then f is a *multilinear map*. We write $\overset{\sim}{\to}$ instead of $\to$ when specifying the domain and codomain of such functions. The (p, q)*-tensor over* V is a multilinear map of the form

$$T : \underbrace{V^* \times V^* \times \cdots \times V^*}_{p \text{ copies}} \times \underbrace{V \times V \times \cdots \times V}_{q \text{ copies}} \overset{\sim}{\to} \mathbb{R}, \tag{2.10}$$

and the components of this map can be treated as a multi-dimensional array of size

$$[N]^{p+q} := \underbrace{[N] \times [N] \times \cdots \times [N]}_{p+q \text{ copies}}.$$

The tuple $(p,\ q)$ is the *valence of* T. If both $p,\ q \geq 1$, then T is a *mixed tensor*.

Note that some authors define $(p,\ q)$-tensor with p and q reversed in Definition 2.23.

Definition 2.24. The space of $(p,\ q)$-tensors. Let us denote the space of all (p, q)-tensors over V as $\mathcal{T}^p_q V$. We write

$$\mathcal{T}^p_q V := \left\{ T \,|\, T : \underbrace{V^* \times V^* \times \cdots \times V^*}_{p \text{ copies}} \times \underbrace{V \times V \times \cdots \times V}_{q \text{ copies}} \xrightarrow{\sim} \mathbb{R} \right\} \tag{2.11}$$

$$\cong \underbrace{V \otimes V \otimes \cdots \otimes V}_{p \text{ copies}} \otimes \underbrace{V^* \otimes V^* \otimes \cdots \otimes V^*}_{q \text{ copies}}, \tag{2.12}$$

$$:= V^{\otimes p} \otimes (V^*)^{\otimes q},$$

and we treat the entire right-hand side of the second line as a single vector space called a *tensor product of vector spaces*.

We defined the tensor product of vector spaces as a vector space that is isomorphic to a set of multilinear maps. In this thesis, we abuse the notation and still write $\mathcal{T}^p_q V$ when we mean $\mathcal{T}^p_q V$ as a vector space through this isomorphism. This implies that we treat tensors as both a multilinear map and a vector in a vector space; our meaning should be clear from context. We did not define tensors as an element of a tensor product of vector spaces because it requires the review of more technical material, and that the approach taken here is sufficient to understand how tensors are used in this thesis. For the rest of this section, we review how one can obtain components and bases for this vector space by using the multilinear map interpretation of tensors.

To make $\mathcal{T}^p_q V$ into a vector space, we equip $\mathcal{T}^p_q V$ with point-wise addition between tensors and scalar-multiplication between a real number and a tensor; i.e., if $T,\ S \in \mathcal{T}^p_q V$, then $\forall \omega_i \in V^*$ and $\forall v_j \in V$, $i \in [p]$, $j \in [q]$, we have

$$\begin{aligned} (T + S)(\omega_{1:p},\ v_{1:q}) &:= T(\omega_{1:p},\ v_{1:q}) + S(\omega_{1:p},\ v_{1:q}), \\ (a \cdot T)(\omega_{1:p},\ v_{1:q}) &:= a \cdot T(\omega_{1:p},\ v_{1:q}). \end{aligned} \tag{2.13}$$

Before we specify a basis, we first define the tensor product between tensors.

Definition 2.25. Tensor product of $(p,\ q)$-tensors. Let $T \in \mathcal{T}^p_q V$ and $S \in \mathcal{T}^r_s V$, V a vector space with field $\mathbb{R}$. We define the *tensor product* $T \otimes S \in \mathcal{T}^{p+r}_{q+s} V$ in a point-wise manner; $\forall \omega_i \in V^*$, $\forall i \in [p + r]$, and $\forall v_j \in V$, $\forall j \in [q + s]$, we have

$$(T \otimes S)(\omega_{1:p+r},\ v_{1:q+s}) := T(\omega_{1:p},\ v_{1:q}) \cdot S(\omega_{p+1:p+r},\ v_{q+1:q+s}), \tag{2.14}$$

where the $\cdot$ operator is the scalar multiplication on $\mathbb{R}$.

Let $\{\phi_i\}$ be a given basis for V, and $\{\psi_j\}$ be a given basis for V^*. To get a basis for $\mathcal{T}^p_q V$ (as a vector space), we first treat each basis vector ϕ_i and basis covector ψ_j as a $(1,\ 0)$-tensor from $\mathcal{T}^1_0 V$ and $(0,\ 1)$-tensor from $\mathcal{T}^0_1 V$, respectively. For a given $a \in [N]^p$ and $b \in [N]^q$ multi-indices, we then use the tensor product of these basis vectors and covectors to define the $(a,\ b)$th basis vector for $\mathcal{T}^p_q V$, and denote it as

$$\{\phi_{a_i}\}_{i=1}^{\otimes p} \otimes \{\psi_{b_i}\}_{i=1}^{\otimes q} := \phi_{a_1} \otimes \phi_{a_2} \otimes \cdots \otimes \phi_{a_p} \otimes \psi_{b_1} \otimes \psi_{b_2} \otimes \cdots \otimes \psi_{b_q} \in \mathcal{T}^p_q V. \tag{2.15}$$

This process is repeated for all $a \in [N]^p$ and all $b \in [N]^q$.

When the bases for V and V^* are denoted by a single symbol, e.g., $\mathcal{B} = \{\phi_i\}$ and $\mathcal{B}^* = \{\psi_i\}$, and $N = \dim V$, we define the short-hand notation

$$\mathcal{B}^{\otimes p} \otimes (\mathcal{B}^*)^{\otimes q} := \left\{ \{\phi_{a_i}\}_{i=1}^{\otimes p} \otimes \{\psi_{b_i}\}_{i=1}^{\otimes q} \right\}_{a \in [N]^p,\, b \in [N]^q}.$$

We call $\mathcal{B}^{\otimes p} \otimes (\mathcal{B}^*)^{\otimes q}$ the *tensor product basis for $\mathcal{T}_q^p V$ from $\mathcal{B}$.*

We are now ready to define the basis coefficients of a tensor, also called the components of a tensor.

Definition 2.26. Components of a finite-dimensional tensor and basis vector. Let $V = \mathbb{R}^N$, $\mathcal{B} = \{\phi_i\}$ is an ordered basis for V, $\mathcal{B}^* = \{\psi_i\}$ is the dual basis of V^*, and $T \in \mathcal{T}_q^p V$. Let $a \in [N]^p$ and $b \in [N]^q$ be multi-indices. The $(a,\, b)$*th component of T with respect to $\mathcal{B}^{\otimes p} \otimes (\mathcal{B}^*)^{\otimes q}$* is defined as

$$(T)_{\mathcal{B}^{\otimes p} \otimes (\mathcal{B}^*)^{\otimes q},\, a,\, b} := T\big(\psi_{1,\, a_1},\, \psi_{2,\, a_2}, \cdots,\, \psi_{p,\, a_p},\, \phi_{1,\, b_1},\, \phi_{2,\, b_2}, \cdots,\, \phi_{q,\, b_q}\big). \tag{2.16}$$

To reduce notation clutter, if S is in $\mathcal{T}_q^0 V$ and $c \in [N]^q$, we write $(S)_{(\mathcal{B}^*)^{\otimes q},\, c}$ to mean $(S)_{(\mathcal{B}^*)^{\otimes q},\, 0,\, c}$. If S is in $\mathcal{T}_0^p V$ and $c \in [N]^p$, we write $[S]_{\mathcal{B}^{\otimes p},\, c}$ to mean $(S)_{\mathcal{B}^{\otimes p},\, c,\, 0}$.

We can express any $T \in \mathcal{T}_q^p V$ by

$$T = \sum_{a \in [N]^p} \sum_{b \in [N]^q} T_b^a \cdot \{\phi_{a_i}\}_{i=1}^{\otimes p} \otimes \{\psi_{b_i}\}_{i=1}^{\otimes q},$$

where

$$T_b^a := (T)_{\mathcal{B}^{\otimes p} \otimes (\mathcal{B}^*)^{\otimes q},\, a,\, b}.$$

In this thesis, we write $(T)_{\mathcal{U}}$ to mean the multi-dimensional array that contains the components of $T \in \mathcal{T}_q^p V$ with respect to some basis $\mathcal{U}$ for $\mathcal{T}_q^p V$, and the structure of such an array corresponds to the ordering used for $\mathcal{U}$. When $p + q = 2$, we call $(T)_{\mathcal{U}}$ the *matrix-form of T* because it is stored in a matrix. The matrix storage layout for representing a $T \in \mathcal{T}_2^0 V$ is to store the component $(T)_{(\mathcal{B}^*)^{\otimes 2},\, (i,\, j)}$ in the ith row and jth column of $(T)_{(\mathcal{B}^*)^{\otimes 2}}$. We sometimes drop the symbol for the basis when it is clear from context.

Example 2.27. Tensor product of two tensors from the same tensor space. Let $V = \mathbb{R}^N$ be a finite-dimensional vector space, and $T,\, S \in \mathcal{T}_q^p V$. Let $a \in [N]^p$ and $b \in [N]^q$ be multi-indices for T and S. Let $c \in [N]^{2p}$ and $d \in [N]^{2q}$ be multi-indices for $W = T \otimes S$. Given the components of T and S with respect to a basis $\mathcal{B}$ for V, the $(c,\, d)$th component of W with respect to $\mathcal{B}^{\otimes p} \otimes (\mathcal{B}^*)^{\otimes q}$ is

$$W_d^c = T_{d_{1:p}}^{c_{1:p}} S_{d_{p+1:2p}}^{c_{p+1:2p}}.$$

When $p = 1$, $q = 0$, and $V = \mathbb{R}^N$, we have $T,\, S \in \mathbb{R}^N$. We store the components of T into the column matrix t, similarly for S and s, and the components of their tensor product are stored in the $N \times N$ matrix w. Then $w = ts^T$ is the outer product of two column matrices.

Example 2.28. Rank-one tensors. Given a D-dimensional vector $v \in \mathcal{T}_0^1 \mathbb{R}^D$ and a finite positive integer L, a rank-one tensor of valence $(L,\, 0)$ can be constructed as follows:

$$v^{\otimes L} := \underbrace{v \otimes v \otimes \cdots \otimes v}_{L \text{ times}}. \tag{2.17}$$

If the ith component of v with respect to $\mathcal{E}(\mathbb{R}^D)$ is denoted by v_i, $\forall i \in [D]$, then

$$\big(v^{\otimes L}\big)_{\mathcal{E}(\mathbb{R}^D)^{\otimes L},\, a} = v_{a_1} v_{a_2} \cdots v_{a_L} = \prod_{l=1}^{L} v_{a_l},\ \forall a \in [D]^L.$$

From Equation 2.2, we have

$$\left(\sum_{i=1}^{D} v_i\right)^L = \sum_{b\in[D]^L} \prod_{l=1}^{L} v_{b_l}.$$

This implies that the components of a rank-one tensor constructed in this fashion, with respect to some basis $\mathcal{B}^{\otimes L}$ for $\mathcal{T}_0^L\mathbb{R}^D$, can be mapped to a D-variate polynomial of the form $\left(\sum_{d=1}^{D} v_d x_d\right)^L$. For each $d \in [D]$, the $x_d \in \mathbb{R}$ here is treated as a real-valued variable for the polynomial, and v_d is the dth component of v with respect to the basis $\mathcal{B}$ for $\mathbb{R}^D$. In this thesis, the space of tensors constructed this way is denoted by

$$\mathcal{J}(D,\, L) := \left\{v^{\otimes L} \in \mathcal{T}_0^L\mathbb{R}^D \;\middle|\; v \in \mathcal{T}_0^1\mathbb{R}^D\right\}. \tag{2.18}$$

2.3.2 Bilinear Forms and Linear Maps as Tensors

An important type of tensor is the $(0,\, 2)$-tensor of $\mathbb{R}^N$. Let $V = \mathbb{R}^N$, $\mathcal{B} = \{b_i\}_{i=1}^N$ be a basis of V, and $T \in \mathcal{T}_2^0 V = V^* \otimes V^*$. If we use the multilinear map interpretation, then $T : V \times V \xrightarrow{\sim} \mathbb{R}$ is a bilinear form. Let the dual basis for V^* be $\mathcal{B}^* = \{u_i\}_{i=1}^N$, so that the induced tensor product basis for $\mathcal{T}_2^0 V$ is

$$(\mathcal{B}^*)^{\otimes 2} = \left\{\{u_{a_i}\}_{i=1}^{\otimes 2}\right\}_{a\in[N]^2} = \{u_i \otimes u_j\}_{(i,\,j)\in[N]^2}$$

Let the component-form of T with respect to this basis be treated as the 2-dimensional array $(T)_{(\mathcal{B}^*)^{\otimes 2}}$. For any $v,\, w \in V$, we can express the evaluation of the map T as

$$\begin{aligned} T(v, w) &= \sum_{a\in[N]^2} (T)_a u_{a_1} u_{a_2}(v)_{a_1} b_{a_1}(w)_{a_2} b_{a_2} && (2.19)\\ &= \sum_{i,\,j} (T)_{(i,\,j)} u_i u_j(v)_i b_i(w)_j b_j \\ &= \sum_{i,\,j} (T)_{(i,\,j)} (v)_i (w)_j \\ &= (v)_{\mathcal{B}}^T (T)_{(\mathcal{B}^*)^{\otimes 2}} (w)_{\mathcal{B}}, && (2.20) \end{aligned}$$

where $= \{u_i \otimes u_j\}_{(i,\,j)\in[N]^2}$. We dropped the subscript for $\mathcal{B}$ and $(\mathcal{B}^*)^{\otimes 2}$ for the components of v, w, and T in the first three lines. The last line is the matrix-form of this expression where matrix product and matrix transpose are used on the arrays.

Similarly, a linear map from a vector space with field $\mathbb{R}$ to itself can be treated as a $(1,\, 1)$-tensor. This is possible by using the isomorphism from the following proposition:

Proposition 2.29. *(Modified from Proposition C.4.7 of [53]). Let V and W be finite-dimensional vector spaces over a field K. The space $W \otimes V^*$ is isomorphic to Hom$(V,\, W)$.*

Proof. The original proposition had $W^* \otimes V$ instead of the $W \otimes V^*$ here. For finite-dimensional vector spaces, $V^* \otimes W$ and $W \otimes V^*$ are isomorphic as vector spaces with the isomorphism being the matrix transpose operator. Thus, the proof presented in [53] for $V^* \otimes W$ applies here. □

The above proposition implies that the tensor space $\mathcal{T}_1^1 V$ is isomorphic to $\text{Hom}(V,\, V) := \left\{f : V \xrightarrow{\sim} V\right\}$; i.e., we can find a unique linear map in $\text{Hom}(V,\, V)$ from a given $T \in \mathcal{T}_1^1 V$, and the converse is true. In this thesis, if $A : \mathbb{R}^m \xrightarrow{\sim} \mathbb{R}^m$, we use the notation $(A)_{\mathcal{B}}$ to mean the matrix-form of the $(1,\, 1)$-tensor that is isomorphic to $A \in \text{Hom}(\mathbb{R}^m,\, \mathbb{R}^m)$. The reason for the modification in Proposition 2.29 is so that we remain

consistent with our tensor product of vector spaces notation for $\mathcal{T}_1^1 V$. Another reason is that the matrix-form of a $T \in W \otimes V^*$ gives rise to a matrix K such that Ku implements $T(u) \in W$ (T treated as a linear map from V to W), where u is the column matrix-form of a vector in V. The matrix-form of the same linear map but as an element of $V^* \otimes W$ would give K^T instead of K, which requires us to use $u^T K$ to implement $T(u)$; this is not the common convention used in the signal processing literature.

By relating bilinear forms and linear maps between finite-dimensional vector spaces to tensors, we are forced to distinguish abstract objects such as bilinear maps and elements from $\mathrm{Hom}(V,\, V)$ as separate objects that have different bases. For example, let $\mathcal{E} = \{e_i\}_{i=1}^{\dim V}$ and $\mathcal{E}^* = \{e_i^*\}_{i=1}^{\dim V}$ be the elementary basis for V and V^*, respectively. The elementary basis for the space of linear maps from V to itself is

$$\left\{e_i \otimes e_j^*\right\}_{(i,\, j) \in [\dim V] \times [\dim V]},$$

This is different from the elementary basis for the space of bilinear forms on V, which is

$$\left\{e_i^* \otimes e_j^*\right\}_{(i,\, j) \in [\dim V]^2}.$$

The inverse of a bilinear form on V is a $(2,\, 0)$-tensor, and it has the following elementary basis:

$$\left\{e_i \otimes e_j\right\}_{(i,\, j) \in [\dim V]^2}.$$

We see that although the components of these types of objects can be stored in a matrix, they are objects from different spaces, as their basis suggests.

2.3.3 Inner Product and Musical Isomorphism

Here, we review an important concept that converts a finite-dimensional vector to a covector, and vice versa.

Definition 2.30. Inner product of finite-dimensional vector spaces. Let $V = \mathbb{R}^N$, and $G \in \mathcal{T}_2^0 V$. We denote $G(\cdot,\, \cdot) := \langle \cdot,\, \cdot \rangle$ and call G an inner product if it satisfies the following: $\forall u,\, v,\, w \in V, \forall a \in \mathbb{R}$,

1. Symmetric: $G(v,\, w) = G(w,\, v)$.
2. Positive-definite: $G(v,\, v) \geq 0$, and equality if and only if $v = 0 \in V$, i.e., v is the neutral element of V.

Inner products are useful for quantifying the concept of angles between two elements of a vector space. Given an inner product G, the *norm induced from* G is the function $\sqrt{G(\cdot,\, \cdot)}$.

Definition 2.31. Vector space isomorphisms. Two vector spaces V and W over the same field F are *isomorphic as vector spaces* if there is a bijection $f : V \to W$, called a *vector space isomorphism*, which preserves addition and scalar multiplication; i.e., $\forall u,\, v \in V$ and $\forall a \in \mathbb{R}$, we have

1. $f(u + v) = f(u) + f(v)$.
2. $f(av) = af(v)$.

If two vector spaces are isomorphic to each other, then we can interpret them as being equivalent in terms of the information provided to us, since having complete information in one space means having complete information in the other.

Example 2.32. Given finite positive integers p and q, and a finite-dimensional vector space V, $\mathcal{T}_q^p V$ is isomorphic to $\mathbb{R}^D$, where $D = \dim(V)^{p+q}$ is the dimension of $\mathcal{T}_q^p V$ as a vector space.

Let V be a finite-dimensional vector space with field $\mathbb{R}$ that is equipped with any bilinear form $T(\cdot, \cdot)$, then

$$\begin{aligned} V &\xrightarrow{\sim} V^* \\ v &\mapsto \omega := T(v, \cdot) = (u \mapsto T(v, u)) \end{aligned}$$

is a linear map from V to V^*. If T is an inner product G, then this linear map is a vector space isomorphism. To see this, let $V = \mathbb{R}^N$ be equipped with an inner product G, and $i, j \in [N]$. We then have

$$\begin{aligned} V &\xrightarrow{\sim} V^* \\ v &\mapsto \omega := G(v, \cdot). \end{aligned}$$

For given bases $\mathcal{B} = \{b_i\}$ for V, $\mathcal{B}^* = \{b_i^*\}$ for V^*, and $(\mathcal{B}^*)^{\otimes 2}$ for $\mathcal{T}_2^0 V$, we have

$$\begin{aligned} G(v, u) &= \sum_{i,j} (G)_{(i,j)} (v)_i (u)_j \\ \implies \omega u &= \sum_{i,j} (G)_{(i,j)} (v)_i (u)_j \\ &= \sum_j (\omega)_j b_j^*(u)_j b_j \end{aligned}$$

We get the last line from the fact that a covector is a linear map from V to $\mathbb{R}$. Since $\omega = \sum_j (\omega)_j b_j^*$, we have an expression of each component

$$(\omega)_{\mathcal{B}} = (G)_{(\mathcal{B}^*)^{\otimes 2}} (v)_{\mathcal{B}}. \tag{2.21}$$

Given an inner product and a basis for V, we just specified a covector ω from a vector v. The reverse direction follows a similar development. Let

$$\begin{aligned} V^* &\xrightarrow{\sim} V \\ \omega := G(v, \cdot) &\mapsto v, \end{aligned}$$

and for some basis $\mathcal{B} = \{b_i\}$ for V , we have

$$\begin{aligned} v &= \sum_k (v)_k b_k \\ &= \sum_{k,i} \delta_{ki} (v)_i b_k \\ &= \sum_{k,j} (G)^{-1}_{(j,k)} \sum_i (G)_{(i,j)} (v)_i b_k \\ &= \sum_k \sum_j (G)^{-1}_{(j,k)} (\omega)_j b_k \\ \implies (v)_{\mathcal{B}} &= (G)^{-1}_{(\mathcal{B}^*)^{\otimes 2}} (\omega)_{\mathcal{B}}. \end{aligned} \tag{2.22}$$

We write $(G)^{-1}_{(\mathcal{B}^*)^{\otimes 2}}$ to mean the matrix inverse of $(G)_{(\mathcal{B}^*)^{\otimes 2}}$. Note that because G is an inner product, the matrices $(G)_{(\mathcal{B}^*)^{\otimes 2}}$ and $(G)^{-1}_{(\mathcal{B}^*)^{\otimes 2}}$ are symmetric positive-definite matrices. We get the third line from the definition of matrix inverse, and the fourth line is from Equation 2.21. Equation 2.21 maps a vector v to a covector ω, and we call the output covector ω the $\flat$-*isomorphism of vector* v, or $v^\flat$. Equation 2.22 maps a covector ω to a vector v, and we call the output vector v the $\sharp$-*isomorphism of covector* ω, or $\omega^\sharp$. These two equations are commonly referred to as the *musical isomorphisms of* V *via* G.

2.4 Matrices and Linear Algebra

In this section, we present our notations on some matrix spaces, and some facts for linear maps between vector spaces. We use the short-hand notations

$$\mathbb{R}^{m\times p}_{\text{full, s}} := \left\{A \in \mathbb{R}^{m\times p}_{\text{s}} \mid \text{rank}(A) = \min(m, p)\right\}$$

to mean the space of full-rank $m \times p$ matrices,

$$\mathbb{R}^{m\times p}_{\text{col, s}} := \left\{A \in \mathbb{R}^{m\times p}_{\text{s}} \mid \text{rank}(A) = p\right\}$$

when $m \geq p$ to mean the space of full column rank $m \times p$ matrices, and

$$\mathbb{R}^{m\times p}_{\text{row, s}} := \left\{A \in \mathbb{R}^{m\times p}_{\text{s}} \mid \text{rank}(A) = m\right\}$$

when $m \leq p$ to mean the space of full row-rank $m \times p$ matrices.

When $A \in \mathbb{R}^{m\times p}_{\text{col, s}}$, its columns can be interpreted as the components of the basis vectors for a p-dimensional subspace of $\mathbb{R}^m$; this subspace is the *column span of* A, and is denoted by $\text{span}(A)$. The components are expressed in terms of some basis $\mathcal{B}$ of $\mathbb{R}^m$, which is commonly taken to be the elementary basis of $\mathbb{R}^m$ if unspecified. The orthogonal projector onto the column span of A is given by

$$\begin{aligned}\text{Proj}_{\text{span}(A)} : \mathbb{R}^m_{\text{s}} &\xrightarrow{\sim} \mathbb{R}^m_{\text{s}}\\ v &\mapsto A\big(A^TA\big)^{-1}A^Tv,\end{aligned}$$

where the input v is the component-form of a vector in $\mathbb{R}^m$ with respect to the basis $\mathcal{B}$. The output is an array of components of a vector in $\mathbb{R}^p$ that is embedded in $\mathbb{R}^m$. These components are represented in $\mathbb{R}^m$ with respect to the basis $\mathcal{B}$. The orthogonal projector onto the orthogonal complement of the column span of A is

$$\begin{aligned}\text{Proj}_{\text{span}(A)^\perp} : \mathbb{R}^m_{\text{s}} &\xrightarrow{\sim} \mathbb{R}^m_{\text{s}}\\ v &\mapsto v - \text{Proj}_{\text{span}(A)}v,\end{aligned}$$

where the output is the component-form of a vector in $\mathbb{R}^{m-p}$ that is embedded in $\mathbb{R}^m$. These components are represented in $\mathbb{R}^m$ with respect to the basis $\mathcal{B}$.

Suppose $P \in \mathbb{R}^{n\times n}_{\text{full, s}}$ is a matrix whose columns are the components of the basis vectors of $\mathcal{B}$ for $\mathbb{R}^n$ with respect to another basis $\mathcal{U}$ for $\mathbb{R}^n$. Let $(v)_{\mathcal{B}}$ denote the column matrix that contains the components of a $v \in \mathbb{R}^n$ with respect to $\mathcal{B}$. Since such a P is the matrix-form of a $(1, 1)$-tensor with respect to the basis $\mathcal{U} \otimes \mathcal{B}^*$, we have

$$(v)_{\mathcal{U}} = P(v)_{\mathcal{B}} \tag{2.23}$$

where the right-hand side is a matrix multiplication. We also have

$$(v)_{\mathcal{B}} = P^{-1}(v)_{\mathcal{U}}. \tag{2.24}$$

Equation 2.23 also holds when $\mathcal{B}$ is a set of n linearly independent vectors in $\mathbb{R}^m$ and $\mathcal{U}$ is a basis for $\mathbb{R}^n$, where $m > n$. In this case, Equation 2.24 holds if the inverse is replaced by the pseudo-inverse.

Given an inner product $\langle\cdot, \cdot\rangle$, the Gram-Schmidt orthogonalization procedure refines a set of linearly independent vectors to a set of orthogonal (with respect to $\langle\cdot, \cdot\rangle$) vectors that also have unit norm. The norm is taken to be the induced-norm of the inner product. An algorithm for performing the Gram-Schmidt orthogonalization procedure is as follows:

Algorithm 2.1 Gram-Schmidt orthogonalization procedure

Require: A set of linearly independent vectors $\{a_k\}_{k=1}^n$ and an inner product $\langle \cdot, \cdot \rangle$.

1: $q_1 = a_1 ./ \sqrt{\langle a_k, a_k \rangle}$. ▷ ./ is per-element division for an array.
2: **for** $k \in [2 : D]$ **do**
3: $\quad t_k = a_k - \sum_{j=1}^{k-1} \frac{\langle a_k, q_j \rangle}{\langle q_j, q_j \rangle} q_j$.
4: $\quad q_k = t_k ./ \sqrt{\langle t_k, t_k \rangle}$.
5: **end for**
6: Return $\{q_k\}_{k=1}^n$.

2.5 Riesz Transform

The Riesz transform as described in [54, 55, 46, 56] is an extension of the Hilbert transform for functions in $L_2(\mathbb{R})$ to functions on $L_2(\mathbb{R}^D)$. In this section, we consider a $f \in L_2(\mathbb{R}^D)$, and its Fourier transform $\hat{f}$. The frequency index of $\hat{f}$ is denoted by ω here, and j denotes the imaginary number. The material here is based on the presentation from [46]. Their reference implementation of the 2-dimensional and 3-dimensional versions of the Riesz-wavelet transform can be obtained at their research group's website (http://bigwww.epfl.ch/, retrieved August 2017).

The Riesz-transform can be interpreted as a covector field, but we treat it as a D-tuple-valued function because we do not explore its covector properties in this thesis.

Definition 2.33. (Section B of [46]) The Riesz operator. Given a finite positive integer D, the Riesz transform operator $\mathcal{R} : L_2(\mathbb{R}^D) \to L_2^D(\mathbb{R}^D)$ is defined point-wise as

$$\mathcal{R}f(x) := \begin{bmatrix} \mathcal{R}_1 f(x) \\ \vdots \\ \mathcal{R}_D f(x) \end{bmatrix}. \tag{2.25}$$

The *ith Riesz component operator* $\mathcal{R}_i : L_2(\mathbb{R}^D) \to L_2(\mathbb{R}^D)$ is defined to have the following frequency response:

$$\hat{\mathcal{R}}_i(\omega) = \frac{-j\omega_i}{\|\omega\|}, \ \forall i \in [D], \tag{2.26}$$

where ω_i is the ith slot of the frequency index ω.

The second-order extension of $\mathcal{R}f$ is obtained by applying the Riesz operator on f twice. The Riesz component operators are linear and space-invariant, so we have $\mathcal{R}_1\mathcal{R}_2 f = \mathcal{R}_2\mathcal{R}_1 f$. This is one of the three distinct terms for the second-order case. For the Lth-order cases, there are $M = \binom{L+D-1}{D-1}$ distinct terms. The Lth-order Riesz-transform can be interpreted as a symmetric tensor field, but we treat it as a M-tuple-valued function in this thesis because its tensorial properties are not needed.

Definition 2.34. (Section B and Equations 4 to 6 of [46]) The higher-order Riesz transform. Given finite positive integers D and L, $\forall f \in L_2(\mathbb{R}^D)$, the Lth-order Riesz transform operator $\mathcal{R}^{(L)} : L_2(\mathbb{R}^D) \to L_2^M(\mathbb{R}^D)$ is defined point-wise as

$$\mathcal{R}^{(L)}f(x) := \begin{bmatrix} \mathcal{R}^{(L,0,\cdots,0)}f(x) \\ \vdots \\ \mathcal{R}^{n}f(x) \\ \vdots \\ \mathcal{R}^{(0,0,\cdots,L)}f(x) \end{bmatrix}, \ \forall n \in \mathcal{I}(D,\, L), \tag{2.27}$$

where $M = \binom{L+D-1}{D-1}$, and n is a multi-index. The *nth Riesz component operator* $\mathcal{R}^n : L_2(\mathbb{R}^D) \to L_2(\mathbb{R}^D)$ is defined as

$$\mathcal{R}^n := \sqrt{\binom{L}{n}}\mathcal{R}_1^{n_1}\mathcal{R}_2^{n_2}\cdots\mathcal{R}_D^{n_D}, \tag{2.28}$$

and the Fourier transform of $\mathcal{R}^n f$ is

$$\hat{\mathcal{R}^n} f(\omega) = \sqrt{\binom{L}{n}}\frac{(-j\omega)^n}{\|\omega\|^M}\hat{f}(\omega). \tag{2.29}$$

Although $\mathcal{R}^{(L)}f(x)$ is a 1-dimensional array in this thesis, we use a multi-index $n \in \mathcal{I}(D,\, L)$ to index its entries. The notation defined in Definition 2.34 allows us to write $\mathcal{R}^n f$ instead of $\left(\mathcal{R}^{(L)}f(x)\right)_n$ when we want to reference the entry of $\mathcal{R}^{(L)}f$ that corresponds to the multi-index n.

Here are some properties of the Riesz transform, taken from Property 1 of [55]:

1. Translation invariance: $\forall x_0 \in \mathbb{R}^D$, $\mathcal{R}\{f(\cdot - x_0)\}(x) = \mathcal{R}\{f(\cdot)\}(x - x_0)$.
2. Scale-invariance: $\forall a \in \mathbb{R}^D$, $\mathcal{R}\left\{f\left(\frac{\cdot}{a}\right)\right\}(x) = \mathcal{R}\{f(\cdot)\}\left(\frac{x}{a}\right)$.

The Riesz transform is also steerable [57], in the sense that for any rotation matrix $R \in \mathbb{R}_s^{D\times D}$ and input $x \in \mathbb{R}^D$, any entry of $\mathcal{R}^{(L)}f(Rx)$ can be expressed as some linear combination of the entries in $\mathcal{R}^{(L)}f(x)$. We need to set up a representation of the rotation matrix before we can present this result from [55]. Let $u \in \mathbb{R}^D$ be a given unit vector, and we use the elementary basis for its representation. From Definition 2.33, we see that for each $i \in [D]$, the Riesz component operator $\mathcal{R}_i$ is isotropic on the hyperplane that is perpendicular to the ith elementary basis vector e_i. This means that given x, $\mathcal{R}_i(R_u x)$ evaluates to the same number for any $D \times D$ rotation matrix R_u that has the components of u as the ith row. This means that $\mathcal{R}_i(Ru)$ is a function that is invariant over the equivalence class of rotation matrices$[R_u]$, where the equivalence relation is that two rotation matrices in $\mathbb{R}^{D\times D}$ are equivalent if their ith row corresponds to the components of u.

Proposition 2.35. *(Modified from Property 2 from [55]) Steerability of the impulse response of Riesz component operators. The component operators here are with respect to the elementary basis of the Riesz transform tensor. Given a unit vector $u \in \mathbb{R}^D$, consider the equivalence relation $\sim$ that specifies two rotation matrices in $\mathbb{R}^{D\times D}$ to be equivalent if their their ith row corresponds to the components of u. If $i \in [D]$, x, $u \in \mathbb{R}^D$, and $R_u \in [R_u]_\sim$, then $\mathcal{R}_i(R_u x) = \sum_{d=1}^D u_d \mathcal{R}_d(x)$, where u_d are the components of u with respect to $\mathcal{E}(\mathbb{R}^D)$.*

Proof. (Taken from [55]) From Definition 2.33, the frequency response of $\mathcal{R}_i(R_u x)$ is □

$$\hat{\mathcal{R}}(R_u\omega) = -j\frac{(R_u\omega)_i}{\|R_u\omega\|} = \sum_{d=1}^{D} -j\frac{u_d\omega_d}{\|\omega\|} = \sum_{d=1}^{D} u_d\hat{\mathcal{R}}_d(\omega).$$

If R is a D-dimensional rotation matrix that has the kth row set to the elementary basis vector e_i, for a given i, $k \in [D]$, then the steerability property implies that the impulse response of the ith Riesz component operator is just the impulse response of the kth Riesz operator under the argument Rx.

Proposition 2.36. *(Reworded from Property 2 of [55]) Component impulse responses can be derived from a single component operator. The component operators here are with respect to the elementary basis of the Riesz transform tensor. Let us choose $\mathcal{R}_k$ as the component operator that we shall derive the other component operators from. The impulse response from the ith Riesz operator is then $\mathcal{R}_i(x) = \mathcal{R}_k(R_{e_i}x)$, $\forall x \in \mathbb{R}^D$, where R_{e_i} is the D-dimensional rotation matrix that has the kth row set to the elementary basis vector e_i.*

Proof. See [55]. □

When one applies the above to interpret a component operator, the choice of k is arbitrary; the authors of [55] arbitrarily set $k = 1$. They also defined a directional analysis based on the steerability property.

Definition 2.37. Directional Hilbert transform (equation 6 of [55]). The directional Hilbert transform of $f \in L_2(\mathbb{R}^D)$ in the direction of the unit vector $u \in \mathbb{R}^D$ is defined as follows:

$$\mathcal{H}_u f(x) = \sum_{d=1}^{D} u_d \mathcal{R}_d f(x) = \langle u,\, \mathcal{R}f(x) \rangle, \tag{2.30}$$

where $\langle \cdot, \cdot \rangle$ is the dot product.

Unser et. al. showed this directional analysis can be interpreted as evaluating a directional derivative (Sections II.C and II.F of [55]), if the input signal f has "zero-mean" (Sections III.B of [54]). I took their meaning of "zero-mean" as having zero moment; i.e., $\int f(x)dx = 0$. In this thesis, we use the Riesz transform on grayscale images. In order to interpret our Riesz transformed outputs as derivatives, we shifted the gray levels of each image by half of its possible dynamic range. For example, if f is a grayscale image that takes values between 0 and 1, then Riesz transform operations done in this thesis would work with the signal g, with $g(x) = f(x) - 0.5$, instead of f.

The generalized Riesz transform [46, 58] is an extension of the Riesz transform where the transform was defined to have some given analysis direction "baked" into the Riesz transform. The experiments in [46, 58] show that when this direction is chosen to coincide with the local dominant orientation, their grayscale image denoising demonstration achieved good performance. It is also possible to construct perfect-reconstruction wavelet frames from certain existing frames using this theory. These results were reported in greater details in [46]. The generalized Riesz transform can also be thought of as allowing a change-of-basis matrix to be used with the Riesz transform reviewed here, but we do not go into the details because these results are not used in this thesis.

One application of the Riesz transform is to perform *instantaneous amplitude* and *instantaneous phase* analysis on f. This was explored for a colour image representation scheme in [56]. This type of analysis was originally developed for real-valued functions in $L_2(\mathbb{R})$ that are locally narrow-band. A signal with this property contains only a narrow-band of frequencies when viewed in a small neighbourhood centered at any point on its domain. The chirp signal is such a signal. In this setting, one can define an instantaneous amplitude $a(t)$ and instantaneous phase $\phi(t)$ such that $s(t) \approx a(t)\cos\phi(t)$ approximately holds at all points t in the domain of $s \in L_2(\mathbb{R})$. One popular approach is to use a complex-valued version of s, called the *complexified signal* s_C, where the real part is s. There are infinitely many ways to define a and ϕ to meet this constraint, since the imaginary part can be chosen arbitrarily; i.e., any $s_C = s + jv$ with v being an arbitrary real-valued function would be valid. Once a choice on v is made, one can define a and ϕ at a point t on the domain of s to be the magnitude and phase of the complex number $s_C(t)$, respectively. Here is our extension of this idea for functions with $\mathbb{R}^D$ as its domain:

Definition 2.38. Instantaneous amplitude and phase for functions in $L_2(\mathbb{R}^D)$. Given a function $v \in L_2(\mathbb{R}^D)$, define the *complexified signal* s_C of a function $s \in L_2(\mathbb{R}^D)$ to be $s_C = s + jv$. For a given $x \in \mathbb{R}^D$, the

magnitude (and phase) of the complex number $s_C(x)$ is assigned the values of the instantaneous amplitude (and instantaneous phase) of s under s_C at x, respectively.

Different choices of v leads to different definitions of the instantaneous amplitude and phase. For functions in $L_2(\mathbb{R})$, a popular choice is to use the Hilbert transform of s, $\mathcal{H}(s)$, as v. The resultant complexified signal is called the *analytic signal* s_A of s. $\mathcal{H}$ here is the conventional Hilbert transform, and it is defined by its frequency response $H(\omega) = -j\text{sgn}(\omega)$. The term *monogenic signal* is used by several authors as an extension of the analytic signal to functions in $L_2(\mathbb{R}^2)$. In this thesis, we use the following extension of this idea to functions in $L_2(\mathbb{R}^D)$ as follows:

Definition 2.39. Higher-order monogenic signal and analysis. Given a function $s \in L_2(\mathbb{R}^D)$, finite positive integers L and D, $M = \binom{L+D-1}{D-1}$, and a map $H : \mathbb{R}_s^M \to \mathbb{R}$, define its *Lth-order monogenic signal* $s_{\text{mon},\,L}$ to be the tuple

$$s_{\text{mon},\,L} := \big(s,\, \mathcal{R}^{(L)}s,\, H\big). \tag{2.31}$$

$\forall x \in \mathbb{R}^D$, the instantaneous amplitude is

$$a_{\text{mon},\,L}(x) := \sqrt{[s(x)]^2 + \left(H \circ \mathcal{R}^{(L)}s(x)\right)^2},$$

and the instantaneous phase is

$$\theta_{\text{mon},\,L}(x) := \arg\big\{s + j \cdot H \circ \mathcal{R}^{(L)}s(x)\big\}.$$

The instantaneous amplitude and phase analysis of s are referred to as the *Lth-order monogenic analysis* of s using H.

Different choices of the map H induce different behaviours in the monogenic analysis that they instantiate, so its choice is application-dependent. We discuss an example of H in Section 3.4.3.

2.6 Manifolds

Part of the motivation for the next few sections of this chapter is to provide a learning aid to researchers in signal processing and machine learning who are interested in learning about differentiable manifolds. Therefore, we present the background required to understand some of the key results from the literature that allows one to define differentiable manifolds, embedded submanifolds, quotient manifolds, and Riemannian structures. Although some of this background is required to understand these results, they are not used frequently in the derivation of the proposed manifolds of this thesis work.

The theory of differentiable manifolds allows one to do calculus on abstract sets. Differentiation on open sets of vector spaces with field $\mathbb{R}$ is well-studied, so we first review the idea of an open set for abstract sets in this section.

Definition 2.40. Topologies of a set. Let M be an abstract set. A subset $\mathcal{O}$ of the collection of all subsets of M (i.e., the power set $\mathbb{P}(M)$) is a topology of M if it satisfies the following:

1. The empty set and M are both in $\mathcal{O}$.
2. If U, $V \in \mathcal{O}$, then $U \cap V$ is in $\mathcal{O}$.
3. Any finite or infinite union of members of $\mathcal{O}$ is in $\mathcal{O}$.

Let us call the following object a *D-dimensional open ball of radius r centered at x*:

$$\text{ball}_r(x) := \left\{ y \in \mathbb{R}^D_s \mid \|y - x\|_2 < r \right\}.$$

Example 2.41. Standard topology on $\mathbb{R}^D$. $\mathcal{O}$ is *the standard topology on* $\mathbb{R}^D_s$ if $U \in \mathcal{O}$, then $\forall p \in U$, $\exists r \in \mathbb{R}_{++}$ such that $\text{ball}_r(p) \subseteq U$.

Example 2.42. Standard topology on $\mathbb{R}$. $\mathcal{O}$ is *the standard topology on* $\mathbb{R}$ if $U \in \mathcal{O}$, then $\forall p \in U$, $\exists \epsilon \in \mathbb{R}_{++}$ such that $(p - \epsilon,\, p + \epsilon) \subseteq u$.

In this thesis, when we treat $\mathbb{R}$, $\mathbb{R}^D_s$, or $\mathbb{R}^D$ as a topological space without specifying the topology, we mean to use their standard topology.

The elements of $\mathcal{O}$ are called *open subsets of* M, and a subset $A \subset M$ is called *closed* if $M \setminus A$ is in $\mathcal{O}$. Any open set that contains a given $p \in M$ is called a *neighbourhood of p under the topology* $\mathcal{O}$. The pair $(M,\, \mathcal{O})$ is called *a topological space*.

Given a topology $\mathcal{O}$ for M, we can derive a topology $\mathcal{O}|_W$ for a subset $W \subseteq M$ by calling any subset $A \subseteq W$ open if and only if there exists an open subset $U \subseteq M$, such that $A = W \cap U$. In notation, this means

$$\mathcal{O}|_W := \{W \cap U \mid U \in \mathcal{O}\} \subseteq \mathbb{P}(W),$$

and $\mathcal{O}|_W$ is called *an induced topology of $\mathcal{O}$ on W*. This topology is also called a *subspace topology* in the literature. Given two topological spaces $(X,\, \mathcal{O}_X)$ and $(Y,\, \mathcal{O}_Y)$, we can also derive a topology $\mathcal{O}_{X\times Y}$ for $Z = X \times Y$ by requiring any $A \in \mathcal{O}_{X\times Y}$ to satisfy the following:

$$\forall (a,\, b) \in U,\ \exists S \in \mathcal{O}_A,\, \exists T \in \mathcal{O}_B,$$

such that S is a neighbourhood of $a \in A$, T is a neighbourhood of $b \in B$, and $S \times T \subseteq U$. The topology constructed this way is called a *product topology*. This definition can be extended to involve the product of more than two topological spaces.

Example 2.43. Alternative formulation for the standard topology of $\mathbb{R}^D_s$. $\mathcal{O}$ is *the standard topology on* $\mathbb{R}^D_s$ if it is the product topology of D copies of the standard topology on $\mathbb{R}$. For reasons that we do not review here, this formulation of the standard topology for $\mathbb{R}^D$ is the same as the one from Example 2.41.

Consider two sets X and Y that are isomorphic to each other via the bijective map ϕ, and X has a topology $\mathcal{O}$. We can derive a topology for Y by mapping any points we need from Y to X, and then use $\mathcal{O}$. This topology construction method for Y is called an *inherited topology from X* in this thesis. Given a topological space $(W,\, \mathcal{O}_W)$, we can equip it with an equivalence relation $\sim$. The set of equivalence classes

$$M := W/\sim := \{[z] \mid z \in W\}$$

is called the *quotient of W by* $\sim$, and each element of this set is a subset of W. This set is called a *quotient space*. When W is a vector space, $W/\sim$ is a *quotient vector space*. The *quotient map of* $W/\sim$ is the following surjective map

$$\begin{aligned} \pi : W &\to M = W/\sim \\ u &\mapsto \pi(u) := [u]. \end{aligned}$$

The topology

$$\mathcal{O}_q = \left\{ A \subseteq M \mid \pi^{-1}(A) \in \mathcal{O}_X \right\}$$

is a topology for M, and it is called the *quotient topology from W using* $\sim$.

The notion of a continuous function on $\mathbb{R}$ can be extended to maps between topological spaces.

Definition 2.44. Continuous maps (Proposition A.3.23 of [53]). Let $(M_1,\, \mathcal{O}_1)$ and $(M_2,\, \mathcal{O}_2)$ be topological spaces, and let $f : M_1 \to M_2$ be a function. f is *continuous* if and only if for all $f(x) \in M_2$ and any neighbourhood V of $f(x)$, there exist a neighbourhood U of x such that $f(U) \subset V$.

Definition 2.45. Homeomorphisms and isomorphic topological spaces. Let $(M_1,\, \mathcal{O}_1)$ and $(M_2,\, \mathcal{O}_2)$ be topological spaces. A *homeomorphism* is a bijective and continuous map $f : M_1 \to M_2$ where f^{-1} is also continuous. If such a map exists, then we write $(M_1,\, \mathcal{O}_1) \cong (M_2,\, \mathcal{O}_2)$ and call them *isomorphic as topological spaces* or *homeomorphic to each other*. A map $g : M_1 \to g(M_1) \subseteq M_2$ is called a *topological embedding* if it is a homeomorphism between the topological spaces $(M_1,\, \mathcal{O}_1)$ and $\Big(g(M_1),\, \mathcal{O}_2|_{g(M_1)}\Big)$.

Most manifold literature works only with topological spaces that have the following property:

Definition 2.46. Paracompact Hausdorff topological spaces. A topological space $(M,\, \mathcal{O})$ is a *Hausdorff topological space* if given any distinct elements $p_1,\, p_2 \in M$, there exists $U_1,\, U_2 \in \mathcal{O}$ such that 1) U_i is a neighbourhood of p_i, $i \in \{1,\, 2\}$, and 2) $U_1 \cap U_2 = \emptyset$. If $(M,\, \mathcal{O})$ is Hausdorff and there is a countable collection $\mathcal{B}$ of open sets, such that every open set is the union of some subcollection of $\mathcal{B}$, then it is a *paracompact Hausdorff topological space*.

Hausdorff topological spaces are useful because distinct points in M have disjoint neighbourhoods, which means a convergent sequence on such spaces should have a unique limit point.

Definition 2.47. Topological manifolds. A paracompact Hausdorff topological space $(M,\, \mathcal{O})$ is called a *D-dimensional topological manifold* if:

1. $\forall p \in M$, there exists a U such that $p \in U \in \mathcal{O}$.
2. For at least one such U, there exists a homeomorphism $\varphi : U \to \varphi(U) \subseteq \mathbb{R}^D_{\text{s}}$.

We often abuse the notation and write M to mean a topological manifold $(M,\, \mathcal{O})$ when the topology is clear from context, or unimportant to the discussion at hand.

We can also derive new topological manifolds from subsets of an existing one.

Definition 2.48. Embedded topological submanifolds. Let $(M,\, \mathcal{O})$ be a topological manifold, $W \subset M$, and $\mathcal{O}|_W$ be the induced topology for W. If $(W,\, \mathcal{O}|_W)$ is also a topological manifold, then it is called a *embedded topological submanifold of* $(M,\, \mathcal{O})$.

We continue our review of embedded submanifolds in Section 2.9.

2.6.1 Differentiable Manifolds

To refine topological manifolds to differentiable manifolds, we need to first introduce the idea of coordinate systems.

Definition 2.49. Charts. Let M be an abstract set, D a finite positive integer, and $U \subseteq M$. The tuple $(U,\, \varphi)$ is a *chart on* M if $\varphi : U \to V \subseteq \mathbb{R}^D_s$ is a homeomorphism. We call φ a *chart map,* and U its corresponding *coordinate patch*. It is common to abuse this terminology, and call φ a chart when U is clear from context, or unimportant in the discussion at hand. We call φ^i the *ith component of the chart map* φ, and denote its entries at any $p \in U$ as:

$$\varphi(p) := \big(\varphi^1(p),\, \varphi^2(p), \cdots,\, \varphi^D(p)\big). \tag{2.32}$$

The entries of the 1-dimensional array $\varphi(p)$ are called the *coordinates of p under the chart* (U, φ).

Note that we use raised indices to reference a component of a chart map. This is a convention used in many differentiable manifold textbooks, e.g., [59, 52, 53]. We defined φ as a map from U to the space of tuples $\mathbb{R}_{\text{s}}^D$ because we want to split φ into the set of maps $\{\varphi^i : p \mapsto (\varphi(p))_i\}_{i=1}^D$ without having to specify any basis, where $(\varphi(p))_i$ is the ith entry of $\varphi(p)$, $p \in U$. We sometimes abuse this definition and interpret φ as a map from U to the vector space $\mathbb{R}^D$ under the condition that all component-forms of $\varphi(p)$ are always expressed with respect to the elementary basis. When we use this abuse of definition, it means that we use the short-hand $\varphi := \text{id}_{\text{v}} \circ \varphi$, where id_{v} was defined in Section 2.1.4. This is done to reduce notation clutter.

When the abstract set is the space of 1-dimensional arrays $\mathbb{R}_{\text{s}}^D$ or the vector space $\mathbb{R}^D$, we sometimes use the *identity chart* $(\mathbb{R}_{\text{s}}^D, \textbf{id})$ or $(\mathbb{R}^D, \textbf{id})$, where **id** is the identity map from Section 2.1.4. By the convention specified in Definition 2.49, we denote this chart by **id** when the abstract set is clear from context. When the abstract set is the space of L-dimensional arrays $\mathbb{R}_{\text{s}}^{D_1 \times D_2 \times \cdots \times D_L}$ or its induced vector space $\mathbb{R}^{D_1 \times D_2 \times \cdots \times D_L}$, we sometimes use the *column-major ordering chart* $(\mathbb{R}^D, \textbf{colmaj})$, where **colmaj** is the column-major ordering map from Section 2.1.4. We denote this chart by **colmaj** when the abstract set is clear from context.

Although presenting concepts without specifying a particular choice of a chart on M is often useful, one must choose a specific chart when numerical computation is required.

Example 2.50. Coordinate-specific definition of a map via a chart. Let $f : U \to W$, where U and W are an arbitrary sets. Let φ denote a chart of the form $(W, \varphi : \mathbb{R}^D \to \mathbb{R}^D)$ on U. In this thesis, the phrase "f is defined in a point-wise manner using the chart φ" means that $\forall p \in U$, f is defined by the coordinates in $f(p)$ that uses φ as the chart map. We can think of this as using $f \circ \varphi$ to specify the point-wise behaviour of f. When $W = \mathbb{R}^D$ and $\varphi = \textbf{id}$, this phrase means that $f(p)$ is specified by its basis coefficients with respect to the elementary basis for W. When $W = \mathbb{R}^{M \times N}$ and $\varphi = \textbf{colmaj}$, this phrase means that $f(p)$ is specified by its basis coefficients with respect to the elementary basis for $\mathbb{R}^{M \times N}$, and the ordering used to identify these coefficients is the column-major ordering map.

Definition 2.51. Change-of-coordinates. Let (U_a, φ_a) and (U_b, φ_b) be charts on M, where $U_a \cap U_b \neq \emptyset$. The *transition map from φ_b to φ_a*, or the *change-of-coordinate map from φ_b to φ_a* is defined by

$$\varphi_a \circ \varphi_b^{-1} : \varphi_b(U_a \cap U_b) \to \varphi_a(U_a \cap U_b). \tag{2.33}$$

In this thesis, we call the map $\Psi_{a,b} := \textbf{id}_{\text{v}} \circ \varphi_a \circ \varphi_b^{-1} \circ \textbf{id}$ the *vector space version of $\varphi_a \circ \varphi_b^{-1}$*.

Definition 2.52. Atlas. Consider an indexable collection of charts $\mathcal{A} := \{(U_a, \varphi_a)\}_{a \in \mathcal{I}}$ on the abstract set M. $\mathcal{I}$ is the index set, and the range of each φ_a is a subset of $\mathbb{R}_s^D$, for some fixed positive integer D. If $\mathcal{A}$ satisfies the following conditions, then it is a C^k *atlas*:

1. $M = \bigcup_a U_a$.
2. $\forall a, b \in \mathcal{I}$, $\varphi_a(U_a \cap U_b)$ is an open set.
3. $\forall a, b \in \mathcal{I}$, $\Psi_{a,b}$ is a k-times differentiable function, where $\Psi_{a,b} : \mathbb{R}^D \to \mathbb{R}^D$ is the vector space version of $\varphi_a \circ \varphi_b^{-1}$. This condition is called the *C^k-compatibility condition between charts*.

We do not review the definition of differentiability for a map between vector spaces with field $\mathbb{R}$, because such information can be found in most vector calculus textbooks, e.g., [60, 61].

Definition 2.53. Maximal atlas and differentiable manifolds. Let $\mathcal{A}$ be a C^k atlas for a manifold M. If any chart that is C^k-compatible with every chart in $\mathcal{A}$ is already in $\mathcal{A}$, then $\mathcal{A}$ is called a *maximal atlas* or a *differentiable structure on M*. A C^k *manifold* $(M, \mathcal{A})$ is a topological manifold where its C^k atlas, $\mathcal{A}$, is a differentiable structure on M. If the atlas is infinitely differentiable, then such a manifold is a *smooth manifold*. The dimension of M is taken to be the dimension of the vector space that M is homeomorphic to.

For Definitions 2.47 and 2.53, it is important to note that the dimension of a differentiable manifold is not necessarily the dimension of the range of a chart map φ. This is because $\varphi(U)$ might be a subspace of the range of φ.

The following proposition allows us to use a differentiable atlas to specify a differentiable structure for a topological manifold:

Proposition 2.54. *(Proposition 1.17 from [59]). Let M be a topological manifold.*

1. *Every smooth atlas $\mathcal{A}$ for M is contained in a unique maximal smooth atlas, called the smooth structure determined by $\mathcal{A}$.*
2. *Two smooth atlases for M determine the same smooth structure if and only if their union is a smooth atlas.*

Proof. See [59]. □

The consequence of the above proposition is that if a topological manifold M is covered by a single smooth chart, then it determines a smooth structure on M. We can then treat M as a differentiable manifold. We often write M to mean $(M, \mathcal{A})$ and omit the topology when there is no confusion. We only consider finite-dimensional differentiable manifolds in this thesis. If we do not specify whether a manifold is a non-differentiable manifold or a differentiable one, then we mean the manifold is a differentiable manifold.

Example 2.55. Vector spaces as manifolds (Section 3.1.4 from [1]). The spaces

$$(\mathbb{R}^m_{\mathrm{s}}, \mathrm{id}),$$
$$(\mathbb{R}^m, \mathrm{id}),$$
$$\big(\mathbb{R}^{m\times n}, \mathrm{colmaj}\big), \text{ and}$$
$$\big(\mathbb{R}^{m\times n}_{\mathrm{s}}, \mathrm{colmaj}\big)$$

with their standard topology are all differentiable manifolds. In this thesis, when we interpret these sets as manifolds but provide no atlas, we mean their corresponding differentiable manifolds here.

Let M_1 and M_2 be m_1-dimensional and m_2-dimensional differentiable manifolds with charts (U_1, φ_1) and (U_2, φ_2), respectively. The set $M_1 \times M_2$ can be made into a differentiable manifold with

$$(U_1 \times U_2, \varphi_1 \times \varphi_2)$$

as its chart, and its manifold topology is the product topology of M_1 and M_2. The differentiable manifold $M_1 \times M_2$ is called the *product manifold of M_1 and M_2, and it has dimension $m_1 + m_2$.*

Definition 2.56. Functions on a manifold. Let M be a D-dimensional manifold with $\{\varphi_a : U_a \to V_a \subseteq \mathbb{R}^D\}$ as its set of chart maps in its atlas. Note that we interpret φ as a map to a vector space here. A function $f : M \to \mathbb{R}$ that generates the set

$$\big\{f \circ \varphi_a^{-1} : V_a \to \mathbb{R}\big\},$$

where each element is a real-valued C^k function on $\mathbb{R}^D$, is called a *C^k function on M*. We write $f \in C^k(M)$ if this is the case. If $f \in C^\infty(M)$, then it is a *smooth function on M*.

If a function on M is smooth in one chart, it is smooth also in any other charts. This is because the composition of smooth functions defined on an open set of a vector space with field $\mathbb{R}$ is also smooth. To reduce notation clutter, our notation treats all real-valued functions on a manifold M as being in $C^\infty(M)$, but we mean that one should replace $C^\infty(M)$ with $C^k(M)$ for some suitable k if the function in the discussion at hand is not infinitely-differentiable but at least once-differentiable.

Definition 2.57. Maps between manifolds (Definition 3.2.1 of [53]). Let M and N be differentiable manifolds of dimensions m and n, respectively. Consider a map $\phi : M \to N$. If for any chart $y : V \to \mathbb{R}^n$ on N and for any chart $x : U \to \mathbb{R}^m$, the map

$$y \circ f \circ x^{-1} : x\big(U \cap \phi^{-1}(V)\big) \subset \mathbb{R}^m \to y(V) \subset \mathbb{R}^n,$$

is a k-times differentiable function, then we write $\phi \in C^k(M,\, N)$.

Definition 2.58. Diffeomorphism. Let M and N be smooth manifolds. A smooth bijective map $\phi : M \to N$ that has a smooth inverse is called a diffeomorphism, and the manifolds M and N are said to be *diffeomorphic* to each other if a diffeomorphism exists between them. We write $M \cong N$ if they are diffeomorphic to each other.

Theorem 2.59. *Diffeomorphism invariance of dimension (Theorem 2.17 from [59]). A non-empty, m-dimensional smooth manifold cannot be diffeomorphic to a n-dimensional smooth manifold unless $m = n$.*

Proof. See [59]. □

2.6.2 Bundles of Topological Manifolds

The Cartesian product and the product topology is a good way to create new topological spaces or manifolds, but sometimes it may be too restrictive. The following object provides an extension of the Cartesian product for manifolds:

Definition 2.60. Bundles. Let E and M be two topological manifolds, and $\pi : E \to M$ a continuous, surjective map. The triplet $(E,\, \pi,\, M)$ is called a *bundle*, E is called the *total space*, and M is called the *base space*. Given a $p \in M$, the *fibre at* p is the set $\pi^{-1}(\{p\})$, and is denoted $\text{fib}_p(E)$. We sometimes write $E \xrightarrow{\pi} M$ to emphasize that $(E,\, \pi,\, M)$ is a bundle. The map π is called the *projection map* of $(E,\, \pi,\, M)$. A map $\sigma : M \to E$ is called a *section* of $(E,\, \pi,\, M)$ if $\pi \circ \sigma(p) = p, \forall p \in M$.

It is common to denote bundles by just their total space when its meaning is clear from context, i.e., $E := (E,\, \pi,\, M)$. When the discussion at hand involves a bundle $(E,\, \pi,\, M)$, we reserve the symbol $\Gamma(E)$ to mean the space of all sections of $(E,\, \pi,\, M)$, and we use the notation $\Gamma^k(E)$ to denote the space of k-times differentiable sections on this bundle if E and M are C^k manifolds. Note that $\Gamma^k(E) \subset \Gamma(E)$ for any given positive integer k. Unless stated otherwise, if a definition involves $\Gamma^\infty(E)$ of some bundle E, we mean to apply this definition to any positive integer k so that a section that is k-times differentiable can still use this definition, for any valid k. When we work with a section $\sigma \in \Gamma(E)$, we use the short-hand notation $\sigma_p := \sigma(p)$.

When we restrict our attention to a subset of the base space, we have the following bundle:

Definition 2.61. Restricted bundle. If $E \xrightarrow{\pi} M$ be a bundle and $W \subset M$ is an embedded topological submanifold, then the following is called the *restricted bundle to* W:

$$\Big(\pi^{-1}(W),\ \pi|_{\pi^{-1}(W)},\ W\Big).$$

We can also define maps between bundles; we need a map between the total spaces, and another between the base spaces.

Definition 2.62. Bundle isomorphisms. Let $E \xrightarrow{\pi} M$ and $E' \xrightarrow{\pi'} M'$ be two bundles, $a : E \to E'$, and $b : M \to M'$. If the following commutes:

$$\begin{array}{ccc} E & \xrightarrow{a} & E' \\ \downarrow{\scriptstyle \pi} & & \downarrow{\scriptstyle \pi'} \\ M & \xrightarrow{b} & M' \end{array},$$

i.e., $\pi' \circ a = b \circ \pi$, then $(a,\, b)$ is called a *bundle morphism from* $E \xrightarrow{\pi} M$ *to* $E' \xrightarrow{\pi'} M'$.

If a^{-1} and b^{-1} exists and forms a bundle morphism from $E' \xrightarrow{\pi'} M'$ to $E \xrightarrow{\pi} M$, then $(a,\, b)$ are *bundle isomorphisms*. In this case, we say $E \xrightarrow{\pi} M$ and $E' \xrightarrow{\pi'} M'$ are *isomorphic as bundles*.

The idea of a differentiable manifold is that it locally behaves like $\mathbb{R}^D_{\text{s}}$. We can similarly introduce the idea that bundles can locally behave like another bundle.

Definition 2.63. Locally isomorphic bundles. Let $E \xrightarrow{\pi} M$ and $E' \xrightarrow{\pi'} M'$ be two bundles. If, $\forall p \in M$, there exists a neighbourhood $U \subseteq M$ of p such that the restricted bundle

$$\left(\pi^{-1}(U),\ \pi|_{\pi^{-1}(U)},\ U\right)$$

is isomorphic to $E' \xrightarrow{\pi'} M'$, then $E \xrightarrow{\pi} M$ and $E' \xrightarrow{\pi'} M'$ are said to be *locally isomorphic as bundles*.

When the fibres at all points on the base space M of a base space is homeomorphic to a common topological manifold, it has the following name:

Definition 2.64. Fibre bundle. If $E \xrightarrow{\pi} M$ satisfies $F \cong \text{fib}_p(E)$, $\forall p \in M$, and F is a topological manifold, then $E \xrightarrow{\pi} M$ is called a *fibre bundle with typical fibre* F.

The simplest fibre bundle is the *product bundle*, which has the form $M \times F \xrightarrow{\pi_1} M$; F is some topological manifold, and π_1 is the map that projects the first slot of the input (see Definition 2.6). For a product bundle, a section must take the form

$$\begin{aligned} \sigma : M &\to M \times F \\ p &\mapsto (p,\, s(p)), \end{aligned}$$

where $s : M \to F$ is a map that can completely represent σ in this special case.

Definition 2.65. Trivial bundles, locally trivial bundles, and vector bundles. A bundle $E \xrightarrow{\pi} M$ is called the trivial bundle if it is isomorphic to a product bundle. It is called a *locally trivial bundle* if it is locally isomorphic to a product bundle. It is called a *vector bundle* if it is locally isomorphic to a product bundle with a typical fibre that is isomorphic to a vector space with field $\mathbb{R}$.

Any section of a trivial bundle can be represented by some map from M to the typical fibre F. The implication is more interesting for a locally trivial bundle; any section can be locally represented by a map from M to the typical fibre F. This map varies slowly as we move from neighbourhood to neighbourhood on the base space M. We can interpret the construction of a vector bundle as attaching a vector space with field $\mathbb{R}$ to each point on a differentiable manifold, where the local triviality requirement ensures that the vector space varies slowly as we move around on the base space.

We can define a set of sections on a vector bundle that serves as a point-wise basis of all sections (point-wise) on that bundle.

Definition 2.66. Frame field. Let $E \xrightarrow{\pi} M$ be a vector bundle. If the set of sections $\{s_k\}_{k=1}^{\dim E}$ evaluated at any given $p \in M$, denoted $\{s_k(p)\}$, is a basis of the fibre E_p, then the set $\{s_k\}_{k=1}^{\dim E}$ is called a *frame field for* E.

Having a vector space available at any point on the base space allows us to use tensors on the base space. We continue our review of this topic in Section 2.7.2 after we define the vectors and covectors that such tensors take in as inputs.

2.7 Tangent Vectors and Tensor Fields

In this section, we review material that are used throughout Chapter 4. Of particular interest is the differential, which is introduced in Section 2.7.1. Differentiable manifolds are sets that locally look like the real Euclidean vector space, and an extension of the directional derivative from a vector space with field $\mathbb{R}$ can be defined on such manifolds. Let M be a differentiable manifold. A *differentiable curve on* M, $\gamma : \mathbb{R} \to M$, is a map between the manifold $\mathbb{R}$ to M, where $\mathbb{R}$ is treated as a differentiable manifold so that the differentiability in the sense of Definition 2.57 can be used. The combination of direction and derivatives on a differentiable manifold results in the following coordinate-free object:

Definition 2.67. Tangent vectors. Consider a differentiable curve γ on a manifold M, such that $\gamma(0) = p$, for some given $p \in M$. For any differentiable function f on M, the *directional derivative of* f *at* p *along* γ is defined as

$$\begin{aligned} X_{\gamma,\,p} : C^\infty(M) &\xrightarrow{\sim} \mathbb{R} \\ f &\mapsto X_{\gamma,\,p} f := \frac{d}{dt}\left(f \circ \gamma(t)\right)\big|_{t=0}. \end{aligned} \tag{2.34}$$

Equivalently, the equivalence class of curves $X_{\gamma,\,p} := [\gamma]_\sim$ under the equivalence relation

$$\gamma_1 \sim \gamma_2 \implies X_{\gamma_1,\,p} f = X_{\gamma_2,\,p} f,\ \forall f \in C^k(M),\ \forall k \in \mathbb{Z}_{++}$$

is called *the tangent vector to* γ *at* p. These two definitions of $X_{\gamma,\,p}$ are equivalent. For this reason, a tangent vector can be interpreted as an equivalence class of differentiable curves on M, where each of these curves must evaluate to p at 0. The expression $X_{\gamma,\,p} f$ is called the *action of* $X_{\gamma,\,p}$ *on* f.

It is common to drop the dependency on γ or p from the notation of a tangent vector when it is unimportant to the discussion at hand. We refer to an element of the equivalence class of curves that make up a given tangent vector X_p as *a curve of* X_p.

To numerically specify a tangent vector, one requires a coordinate system, i.e., a chart. Tangent vectors are elements of a vector space, so we need a basis. When $(U,\, x)$ is a chart for M, $p \in U$, and $f : U \to \mathbb{R}$ is a differentiable on M, we use the following short-hand notation throughout this thesis:

$$\left.\frac{\partial f}{\partial x^i}\right|_p := \left.\frac{\partial (f \circ x^{-1})}{\partial x^i}\right|_{x(p)}. \tag{2.35}$$

Proposition 2.68. *(Based on Proposition 3.3.2 of [53]) Chart-induced basis. Let* M *be a differentiable manifold of dimension* m*,* $p \in M$*,* x *a chart map for* M*. The set of all tangent vectors to* M *at* p *is a vector space* T_pM *of dimension* m *with basis*

$$\left\{ \left.\frac{\partial}{\partial x^i}\right|_p \right\}_{i=1}^{M}.$$

This basis is called the chart-induced basis at p*. The vector space* T_pM *is called the tangent space of* M *at* p*.*

Proof. See [53]. □

The set of functions $\left\{\frac{\partial}{\partial x^i}\right\}_{i=1}^{M}$ is used frequently, so a common short-hand notation when the chart map is clear from context or unimportant to the discussion at hand is

$$\partial_i := \frac{\partial}{\partial x^i}. \tag{2.36}$$

This notation is commonly used because the convention in most differentiable manifold literature is to use a raised index for a tangent covector or a component of a tangent vector, and a lowered index for a tangent vector or a component of a tangent covector. The ith component function of $X_{\gamma,\,p}$ is

$$X^i_{\gamma,\,p} := \left.\frac{dx^i(\gamma(t))}{dt}\right|_{t=0}; \tag{2.37}$$

see the proof of Proposition 3.3.2 from [53] for its derivation. We say a vector $X_{\gamma,\,p}$ *acts* on a function f to mean the evaluation of the directional derivative in Equation 2.34:

$$\begin{aligned} X_{\gamma,\,p} f &= \sum_{i=1}^{m} X^i_{\gamma,\,p} \partial_i f \\ &= \sum_{i=1}^{m} \left.\frac{\partial f}{\partial x^i}\right|_p \left.\frac{dx^i(\gamma(t))}{dt}\right|_{t=0}. \end{aligned} \tag{2.38}$$

Equation 2.34 shows that we can interpret a tangent vector as a linear map, and we can treat it as a $(0,\,1)$-tensor. The dual vector space of the tangent space T_pM is called the cotangent space, and is denoted by T^*_pM. An element of this dual space is called a *cotangent vector*, or a *covector* when it is clear from context that the discussion at hand involves a cotangent space.

Definition 2.69. The *gradient operator at* $p \in M$, d_p, is defined as

$$\begin{aligned} d_p : C^\infty(M) &\xrightarrow{\sim} T^*_pM \\ f &\mapsto d_p f, \end{aligned} \tag{2.39}$$

where the cotangent vector d_pf is defined point-wise by

$$(d_pf)X := Xf, \;\; \forall X \in T^*_pM.$$

The covector d_pf is called *the gradient of* f *at* p, and we often use the notation $(df)_p := (d_pf)$ in this thesis.

The *chart-induced covector basis at* p

$$\left\{d_px^i\right\}_{i=1}^m$$

is a basis for T^*_pM, but the notation $\{dx^i\}_{i=1}^m$ is often used instead when p is not provided. Let $\omega := (d_pf)$, then

$$\omega X = \sum_{i=1}^{m} \omega_i d_p x^i X^i \partial_i. \tag{2.40}$$

Since $\omega X = Xf$, then it follows from Equations 2.38 and 2.40 that the ith component of the covector d_pf is

$$\omega_i = (d_pf)_i = \left.\frac{\partial f}{\partial x^i}\right|_p.$$

When we denote the components of vectors and covectors under a tangent space setting, we emphasize their difference by using a subscript for covector components, and a superscript for vector components. We use a superscript for a basis vector of the dual tangent space, and a subscript for a basis vector of the tangent space. This results in the same index occurring once in both the subscript and the superscript when we write the linear expansion of a vector or covector in terms of its components and basis vectors. This notation is used in most other literature on differentiable manifolds, e.g., [59, 53, 17]. We do not use this notation when vectors and covectors are not discussed under a tangent space setting.

2.7.1 Differentials (Push-Forwards)

We often wish to map a directional derivative operator from one manifold to another. This requires a map between the tangent spaces of the two manifolds.

Definition 2.70. The differential (or push-forward) of a map between manifolds. Let $\phi : M \to W$ be a smooth map between differentiable manifolds M and W, with $m = \dim M$ and $w = \dim W$. The *differential (or push-forward) of ϕ at p* is defined as

$$\begin{aligned} (d\phi)_p : T_pM &\xrightarrow{\sim} T_{\phi(p)}W \\ X &\mapsto (d\phi)_p X, \end{aligned} \tag{2.41}$$

where the vector $(d\phi)_p X$ is defined point-wise by

$$(d\phi)_p Xf := X(f \circ \phi), \ \forall f \in C^\infty(W).$$

The above definition is coordinate-free, and we need a basis before we proceed to a coordinate-specific definition. Since $(d\phi)_p \in \mathrm{Hom}\big(T_pM, T_{\phi(p)}W\big)$ is isomorphic to an element of the vector space $T_{\phi(p)}W \otimes T_p^*M$ (see Proposition 2.29), we can use the tensor product basis

$$\mathfrak{B}\big(T_{\phi(p)}W \otimes T_p^*M\big) := \left\{ d_p y^i \otimes \frac{\partial}{\partial x^j}\bigg|_p \right\}_{(i,\,j)\in[w]\times[m]}. \tag{2.42}$$

In this thesis, when V is a finite-dimensional vector space that is made from the tensor product between tangent and cotangent spaces, we write $\mathfrak{B}(V)$ to mean the basis of V. We write $\mathfrak{B}$ when the meaning of V is clear from context. Tensor product of basis is reviewed in the discussion leading up to Equation 2.15. When each of the input vector spaces are not from the same manifold, such as the case for $(d\phi)_p$, we need to specify charts from all the manifolds that are involved. In Equation 2.42, we used the chart x for the manifold M, and the chart y for the manifold W.

Given a chart x for M and a chart y for W, a smooth map $\phi : M \to W$, and a $p \in M$, we have, $\forall X \in T_pM$ and any $f \in C^\infty(W)$,

$$\begin{aligned} (d\phi)_p Xf &= \sum_{i=1}^{w} \frac{\partial f}{\partial y^i} \frac{d(y^i \circ \phi \circ \gamma)}{dt}\bigg|_{t=0} \\ &= \sum_{i=1}^{w} \frac{\partial f}{\partial y^i} \left(\sum_{i=1}^{m} \frac{\partial (y^i \circ \phi \circ x^{-1})}{\partial x^j} \frac{d(x^j \circ \gamma)}{dt}\bigg|_{t=0} \right), \end{aligned} \tag{2.43}$$

where γ is a curve of X, $m = \dim M$, and $w = \dim W$. It follows from Equations 2.37 and 2.45 that we can use a matrix $\Big((d\phi)_p\Big)_{\mathfrak{B}}$ to represent $(d\phi)_p$

$$\Big((d\phi)_p\Big)_{\mathfrak{B},\,(i,\,j)} = \frac{\partial (y^i \circ \phi \circ x^{-1})}{\partial x^j}\bigg|_{t=0}. \tag{2.44}$$

This matrix is the Jacobian matrix when x and y are the identity chart on their respective manifolds. The following is the coordinate-specific matrix equation that describes Equation 2.43 using the chart-induced basis for M and W:

$$\begin{aligned} (d\phi)_p\big(X,\, d_{\phi(p)}f\big) &= \big(d_{\phi(p)}f\big)^T_{\mathfrak{B}\left(T^*_{\phi(p)}W\right)} \Big((d\phi)_p\Big)_{\mathfrak{B}\left(T_{\phi(p)}W \otimes T_p^*M\right)} (X)_{\mathfrak{B}\left(T_pM\right)} \\ &= (d\phi)_p Xf, \end{aligned} \tag{2.45}$$

where the notation $(d\phi)_p$ on left-hand side is abused to mean the element of $T_{\phi(p)}W \otimes T_p^*M$ that corresponds to $(d\phi)_p$ under the canonical isomorphism from Proposition 2.29, i.e., $(d\phi)_p$ on the left-hand side is being treated like an element of $T_{\phi(p)}W \otimes T_p^*M$. The object $(d_{\phi(p)}f)_{\mathfrak{B}(T_p^*M)}$ here is a column matrix that contains the components of the gradient of f with respect to $\{dy^j\}_{j=1}^w$. Equation 2.45 implies that the coordinate-form of the right-hand side of Equation 2.41 is

$$\Big((d\phi)_p X\Big)_{\mathfrak{B}(T_{\phi(p)}W)} = \Big((d\phi)_p\Big)_{\mathfrak{B}(T_{\phi(p)}W \otimes T_p^*M)} (X)_{\mathfrak{B}(T_pM)}.$$

We call the 2-dimensional array $\Big((d\phi)_p\Big)_{\mathfrak{B}(T_p^*M \otimes T_{\phi(p)}W)}$ the *matrix-form of* $(d\phi)_p$.

The differential has the following property:

$$(d\phi)_p X_{\gamma,\,p} = Y_{\phi\circ\gamma,\,\phi(p)},$$

where X denotes a vector on T_pM, and Y denotes a vector on $T_{\phi(p)}W$.

2.7.2 Tensor Fields as Sections

Given a differentiable manifold M, the *tangent bundle as a set* is

$$TM := \bigcup_{p\in M} T_pM,$$

and the tangent bundle projection map is

$$\begin{aligned} \pi : TM &\to M \\ X &\mapsto p, \end{aligned}$$

where p is the point for which $X \in T_pM$. We do not go into the details, but it is possible to specify a topology and a differentiable atlas for TM, such that TM is a differentiable manifold and that $TM \xrightarrow{\pi} M$ is a smooth bundle of differentiable manifolds [59].

The fibres of $TM \xrightarrow{\pi} M$ are all different tangent spaces of M. Since each T_pM is isomorphic to $\mathbb{R}^m$ for any $p \in M$, $TM \xrightarrow{\pi} M$ is a vector bundle. Note that a $(\dim M)$-dimensional vector space with field $\mathbb{R}$ is isomorphic to the space of $(1,\,0)$-tensors, $\mathcal{T}_0^1(T_pM)$. We do not go into the details, but given finite positive integers p and q, we can similarly construct a vector bundle where the fibres are all different tensor products of tangent spaces and cotangent spaces of M. This bundle is a vector bundle because the fibres are all isomorphic to the vector space $\mathcal{T}_q^p(T_pM)$, which is isomorphic to a vector space with field $\mathbb{R}$ (see Example 2.32). We denote the total space of this bundle by

$$TM^{\otimes p} \otimes T^*M^{\otimes q} := \underbrace{TM \otimes TM \otimes \cdots \otimes TM}_{p \text{ copies}} \otimes \underbrace{T^*M \otimes T^*M \otimes \cdots \otimes T^*M}_{q \text{ copies}}. \tag{2.46}$$

The TM bundle is a special case of this bundle when $p = 1$ and $q = 0$.

The $TM^{\otimes p} \otimes T^*M^{\otimes q}$ vector bundle is very useful, and it is called a *tensor bundle* in the literature. We can interpret a section of this bundle as a multilinear map that is indexed by points on the manifold, but it is required to be continuous on the topology of M so that we can do calculus on it. A section of the TM bundle is called a *vector field*, a section of the T^*M bundle is called a *covector field*, and a section of the $TM^{\otimes p} \otimes T^*M^{\otimes q}$ bundle is called a *tensor field*. In this thesis, when we drop the subscript $p \in M$ on a differential $(d\phi)_p$ or a gradient covector $(d_pf) = (df)_p$, we mean to treat it as a $(1,\,1)$-tensor field $d\phi \in \Gamma(TM \otimes T^*M)$ (i.e., point-wise isomorphic to $\mathrm{Hom}(T_pM,\,T_pM)$) or covector field $df \in \Gamma(T^*M)$, respectively. We also use the terms vector fields, covector fields, and tensor fields to describe sections on a generic vector bundle when appropriate. We add the word *smooth* to the front of vector, covector, and tensor fields, if they represent a smooth section.

2.7.3 Pull-Backs

When we want to map a linear functional from one manifold to another, we need a map between the cotangent spaces of the two manifolds.

Definition 2.71. The pull-back of a map between manifolds. Let $\phi : M \to W$ be a smooth map between differentiable manifolds. The *pull-back of ϕ at p* is defined as

$$\begin{aligned} \phi_p^* : T^*_{\phi(p)}W &\xrightarrow{\sim} T^*_p M \\ \omega &\mapsto \phi_p^* \omega, \end{aligned} \tag{2.47}$$

where the vector $\phi_p^*\omega$ is defined point-wise by

$$\phi_p^*(\omega)(X) := \omega\Big((d\phi)_p X\Big), \ \forall X \in T_pM.$$

This idea can be extended to sections and bundles.

Definition 2.72. Pull-back bundle. Let $E \xrightarrow{\pi} M$ be a bundle, M' a topological manifold, and $\phi : M' \to M$. Define

$$\phi^* E := \{(m',\, e) \in M' \times E \mid \pi(e) = \phi(m')\} \tag{2.48}$$

and

$$\begin{aligned} \pi' : \phi^* E &\to M' \\ (m',\, e) &\mapsto m'. \end{aligned}$$

The bundle $\phi^* E \xrightarrow{\pi'} M'$ is called the *pull-back bundle of E by ϕ*. If σ is a section for $E \xrightarrow{\pi} M$, the map $\phi^*\sigma := \sigma \circ \phi$ is a section for $\phi^* E \xrightarrow{\pi'} M'$ that is called the *pull-back of σ by ϕ*.

The pull-back bundle takes the following setup on the left to the setup on the right:

$$\begin{array}{ccc} & & E \\ & {\scriptstyle\sigma}\uparrow\ \downarrow{\scriptstyle\pi} & \\ M' & \xrightarrow{\phi} & M \end{array} \qquad \begin{array}{ccc} \phi^* E & & E \\ {\scriptstyle\phi^*\sigma}\uparrow\ \downarrow{\scriptstyle\pi'} & & {\scriptstyle\sigma}\uparrow\ \downarrow{\scriptstyle\pi} \\ M' & \xrightarrow{\phi} & M \end{array}.$$

We can interpret it as something similar to a restricted bundle to $\phi^{-1}(M)$.

Given a map between manifolds ϕ, here are some hints that I find helpful in remember the mapping directions of differentials (i.e., push-forwards) and pull-backs:

- If we want to linearly transport an object from the domain of ϕ to the range of ϕ, then a push-forward might be useful.
- If we want to linearly transport an object from the range of ϕ to the domain of ϕ, then a pull-back might be useful.
- Push-forwards tend to act on vectors.
- Pull-backs tend to act on covectors or sections of $(0,\, q)$-tensor fields.

2.8 Riemannian Metrics and Affine Connections

In this section, we review the Riemannian metric, retractions, and affine connections. Riemannian metrics and retractions are used throughout this thesis, but affine connections are only required to understand the Riemannian Hessian that is defined later in this section. The implementation of the Riemannian Hessian used in Chapter 4 are approximations, which require the concepts of retractions.

On a finite-dimensional vector space with field $\mathbb{R}$, an inner product can induce a norm, which can be used as a measure of discrepancy between two vectors. Although not all norms can be constructed this way, this method allows for an easy extension to differentiable manifolds, since it only requires the specification of a type of bilinear map at each point on the manifold.

Definition 2.73. Riemannian Metric. A section g of the vector bundle $T^*M^{\otimes 2} \xrightarrow{\pi} M$ is a *Riemannian metric* if for each $p \in M$, $G_p := G(p)$ is an inner product for T_pM. This means, $\forall u,\, v \in T_pM$, G_p satisfies

1. $G(v,\, w) = G(w,\, v)$.
2. $G(v,\, v) \geq 0$, and equality if and only if $v = 0 \in V$, i.e., v is the neutral element of V.

When a chart x is specified, the $(i,\, j)$th component of G is $G_{i,\,j} := G\big(\frac{\partial}{\partial x^i},\, \frac{\partial}{\partial x^j}\big)$.

A Riemannian metric evaluated at some $p \in M$ is a symmetric $(0,\, 2)$-tensor, and we refer to such a tensor as a *metric tensor* in this thesis. Given a basis $\mathcal{B}$ of T_pM, we have

$$G_p(X_p,\, Y_p) = (X_p)_{\mathcal{B}}^T (G_p)_{(\mathcal{B}^*)^{\otimes 2}} (Y_p)_{\mathcal{B}}. \tag{2.49}$$

Since a Riemannian metric is a section of a bundle of smooth manifolds, it is smooth in the sense that for any smooth vector fields $X,\, Y \in \Gamma(TM)$, the function $p \mapsto G_p(X_p,\, Y_p)$ is smooth. This is equivalent to requiring the component functions of G being smooth in some chart. Given a basis for T_pM, the *matrix-form of* G is the 2-dimensional array where the $(i,\, j)$th entry is $G_{i,\,j}$ evaluated at p.

Example 2.74. The Euclidean metric and Euclidean space. For every tangent space in the tangent bundle, the Euclidean metric is the dot product of two vectors. We denote this metric by G_{id}, and the Riemannian manifold $(\mathbb{R}^m,\, G_{\text{id}})$ is called the *m-dimensional Euclidean space*.

We use the Frobenius product, which is the dot product for multi-dimensional arrays, when we need to compute the Euclidean metric. Throughout this thesis, we sometimes use the following short-hand notation for the Frobenius product for same-sized matrices A and B:

$$A \circ_{\text{F}} B = \sum_{i \in |A|} A_i B_i. \tag{2.50}$$

When A and B are matrices of the same size, the Frobenius product is

$$A \circ_{\text{F}} B \, = \text{tr}\big(A^T B\big). \tag{2.51}$$

A differentiable manifold M that is equipped with a Riemannian metric G is called a *Riemannian manifold*, and we denote it by $(M,\, G)$.

Proposition 2.75. *(Proposition 5.1.8 of [53]) Every smooth manifold has a Riemannian metric.*

Proof. See [53]. □

Recall from Section 2.3.3 that the inner product G_p can induce the $\sharp$-isomorphism that maps a covector ω to a vector $\omega^\sharp$ for T_pM; see Equation 2.22 for the component-wise specification of this isomorphism. Given some basis $\mathcal{B}$ for T_pM, the matrix-form of the $\sharp$-isomorphism is

$$\left(\omega^\sharp\right)_{\mathcal{B}} = (G_p)^{-1}_{(\mathcal{B}^*)^{\otimes 2}}(\omega)_{\mathcal{B}^*},$$

where $(G_p)_{(\mathcal{B}^*)^{\otimes 2}}$ is the matrix-form (see Example 2.3.2) of the metric tensor G_p. It follows that

$$\begin{aligned} Xf &= (d_pf)X \\ &= (d_pf)^T_{\mathcal{B}^*}(X)_{\mathcal{B}} \\ &= \left((d_pf)^\sharp\right)^T_{\mathcal{B}}(G_p)_{(\mathcal{B}^*)^{\otimes 2}}(Y)_{\mathcal{B}} \\ &= G_p\Big((d_pf)^\sharp,\, X\Big),\ \forall X \in \Gamma^\infty(TM) \end{aligned} \tag{2.52}$$

is a way to evaluate the directional derivative of f at p along a curve of X.

The Riemannian metric is not actually a metric, since it does not specify a distance. To see its relationship with metrics, we need to first define the velocity of a differentiable curve.

Definition 2.76. Velocity of a curve. Given a differentiable curve γ on a differentiable manifold M and a chart x, we define the *velocity of* γ to be the map

$$\begin{aligned} \dot\gamma : \mathbb{R} &\to \mathbb{R} \\ t_0 &\mapsto \frac{d}{dt}(x\circ\gamma)\Big|_{t=t_0}. \end{aligned}$$

The induced arc length of a curve from a Riemannian metric is

$$L(\gamma) = \int_a^b \sqrt{G_{\gamma(t)}(\dot\gamma(t),\, \dot\gamma(t))}dt,$$

and a notion of a metric can be defined based on this arc length.

Consider a map $\phi : M \to W$ between manifolds. Recall $d\phi$ is a linear map between T_pM and T_pW for some given $p \in M$. The pull-back bundle of TW in this case is the tangent bundle for M, and it is very useful when one wants to inherit $(0,\, q)$-tensor fields (i.e., maps that take vectors as inputs) from W, for any finite positive integer q. Below is a diagram of the pull-back bundle $TM := \phi^*TW$ that we used here.

$$\begin{array}{ccc} TM := \phi^*M & & TW \\ {\scriptstyle \phi^*\sigma}\uparrow \quad \downarrow{\scriptstyle \pi'} & & {\scriptstyle \sigma}\uparrow \quad \downarrow{\scriptstyle \pi} \\ M & \xrightarrow{\ \phi\ } & W \\ & \xleftarrow[\ \phi^{-1}\ ]{} & \end{array}.$$

When $q = 2$, we have the metric tensor. If $(W,\, G)$ is a Riemannian manifold, then the *pull-back metric of* G *to* TM *using* ϕ is denote by ϕ^*G. It is defined in a point-wise manner:

$$(\phi^*G)_p(X,\, Y) = G_{\phi(p)}((d\phi)X,\, (d\phi)Y),\ \forall X,\, Y \in T_pM,\ \forall p \in M. \tag{2.53}$$

The inner product is a symmetric positive-definite bilinear form. The condition of positive-definite for bilinear forms can be weakened to just requiring its matrix-form (with respect to some basis) to be invertible.

Definition 2.77. Pseudo-inner product. Let V be a finite-dimensional vector space. The bilinear form $B : V \times V \xrightarrow{\sim} \mathbb{R}$ is a pseudo-inner product for V if the following holds for all $w \in V$:

1. $(w,\, v) = (v,\, w)$, for all $v \in V$.
2. $(v,\, w) = 0 \implies v = 0$. This is equivalent to $\det(B)_{\mathcal{B}} \neq 0$ for some basis $\mathcal{B}$ of $\mathcal{T}_1^1 V$.

A pseudo-Riemannian metric can be formulated based on the pseudo-inner product, but we do not use such an object in this thesis.

2.8.1 Affine Connections

An affine connection is a structure that one can impose on a vector bundle.

Definition 2.78. Affine Connections and covariant derivatives (Definition 5.2.1 of [53]). Let M be a differentiable manifold, and let $(E,\, \pi,\, M)$ be a vector bundle. The map

$$\begin{aligned} \nabla : \Gamma^\infty(TM) \times \Gamma^\infty(E) &\to \Gamma^\infty(E) \\ (X,\, \sigma) &\mapsto \nabla_X \sigma, \end{aligned}$$

that satisfies the following is an *affine connection of* M:

1. Linear in first slot: $\forall \sigma \in \Gamma^\infty(E), \forall f\, g \in C^\infty(M)$,

$$\nabla_{fX_1 + gX_2}\sigma = f\nabla_{X_1}\sigma + g\nabla_{X_2}\sigma,\ \forall X_1,\, X_2 \in \Gamma^\infty(TM).$$

2. Linear in second slot: $\forall a,\, b \in \mathbb{R}$,

$$\nabla_X(a\sigma_1 + b\sigma_2) = a\nabla_X\sigma_1 + b\nabla_X\sigma_2,\ \forall \sigma_1,\, \sigma_2 \in \Gamma^\infty(E).$$

3. Product rule: $\forall X \in \Gamma^\infty(TM), \forall \sigma \in \Gamma^\infty(E)$,

$$\nabla_X f\sigma = (Xf)\sigma + f\nabla_X\sigma,\ \forall f \in C^\infty(M).$$

The section $\nabla_X\sigma \in \Gamma^\infty(E)$ is called the *covariant derivative of* σ *in the direction of* X.

There are others, such as the Lie derivative, but we do not use them in this thesis. The covariant derivative is a type of derivative one can define for differentiable sections of a vector bundle. One can also specify an affine connection with additional properties for tensor bundles, so that we can compute covariant derivatives of sections of a tensor bundle. We do not review it in this thesis because the work done in this thesis only requires covariant derivatives of sections of the tangent bundle.

An affine connection is specified by the following set of functions.

Definition 2.79. Christoffel symbols of an affine connection. Let $(E,\, \pi,\, M)$ be a vector bundle, ∇ an affine connection for E, $\{s_j\}_{j=1}^{\dim E}$ a frame field for E, and $\{\partial_i\}_{i=1}^{\dim M}$ the chart-induced basis for TM. *The Christoffel symbols of* ∇ *with respect to* $\{s_j\}$ *and* $\{\partial_i\}$ are the elements in the set $\left\{\Gamma^k_{i,\,j}\right\}_{i,\,j,\,k\in[\dim E]}$, and must satisfy

$$\nabla_{\partial_i} s_j = \sum_k \Gamma^k_{i,\,j}\partial_k, \tag{2.54}$$

where each $\Gamma^k_{i,j}$ is some function in $C^\infty(M)$. Given any $X = \sum_i X^i \partial_i \in \Gamma^\infty(TM)$ and any $Y = \sum_j Y^j s_j \in \Gamma^\infty(E)$, the Christoffel symbols of ∇ are used in the following way to evaluate the connection at X and Y:

$$\nabla_X Y = \sum_{i,k} X^i \left(\frac{\partial Y^k}{\partial x^i} + \sum_j Y^j \Gamma^k_{i,j} \right) s_k, \ i \in [\dim M], \ j, \ k \in [\dim E]. \tag{2.55}$$

There is a bijective correspondence between an affine connection and its Christoffel symbols, so specifying one is equivalent to specifying the other. The construction of an affine connection does not require a Riemannian metric to be specified. However, given a Riemannian metric, one can use the following result to specify a special affine connection that is unique:

Theorem 2.80. *Levi-Civita Theorem (Theorem 5.2.11, Equation 5.17, and Definition 4.2.10 of [53]). Let (M, G) be a Riemannian manifold. The Levi-Civita connection exists on such a manifold, and it is the unique affine connection ∇ that satisfies the following two conditions:*

1. Compatibility: for all $X \in \Gamma^\infty(TM)$ and all $Y, Z \in \Gamma^\infty(E)$, we have

$$\nabla_X G(Y, Z) = G(\nabla_X Y, Z) + G(Y, \nabla_X Z).$$

2. Symmetry: for all $X \in \Gamma^\infty(TM)$ and all $Y \in \Gamma^\infty(E)$, we have

$$\nabla_X Y - \nabla_Y X = XY - YX.$$

Proof. See [53]. □

The Levi-Civita connection can be specified by the following result:

Proposition 2.81. *(Proposition 5.2.13 of [53]) The Christoffel symbols of the Levi-Civita connection. Let (M, G) be a smooth Riemannian manifold with dimension m. Then over a coordinate patch of M with chart x, the Christoffel symbols of the Levi-Civita connection are given by*

$$\Gamma^k_{i,j} = \frac{1}{2} \sum_{l=1}^m G^{kl} \left(\frac{\partial G_{jl}}{\partial x^i} + \frac{\partial G_{li}}{\partial x^j} - \frac{\partial G_{ij}}{\partial x^l} \right),$$

where G^{kl} is the (k, l)th entry of the inverse of the Riemannian metric, i.e., its matrix-form is the matrix inverse of the matrix-form of G.

Proof. See [53]. □

The Levi-Civita connection is the only type of affine connection that we work with in this thesis.

Example 2.82. Levi-Civita connection of a Euclidean space. The directional derivative for vector fields on $\mathbb{R}^m$ is the Levi-Civita connection ∇ of $(\mathbb{R}^m, G_{\text{id}})$. It is defined by

$$(\nabla_X Y)_p := \lim_{t \to 0} \frac{Y_{p+tX_p} - Y_p}{t}.$$

2.8.2 Retractions

To discuss the properties of a given differentiable curve on a differentiable manifold M, we need a sense of covariant derivative for this curve. First, we review some ideas about curves.

Definition 2.83. Smooth vector fields along a curve (Definition 5.3.1 from [53]). Let M be a differentiable manifold, and $\gamma : I \to M$ be a differentiable curve in M, where I is an open interval in $\mathbb{R}$. We call V a *vector field along* γ if the following are true:

1. $\forall t \in I$, $V(t)$ is a tangent vector in $T_{\gamma(t)}M$.
2. V is a smooth map $I \to TM$.

We denote by $\Gamma^\infty_\gamma(TM)$ the space of smooth vector fields on M along γ.

If V is in fact the restriction of a vector field Y to γ, i.e., γ does not self-intersect, then we say V *is induced from* Y or that V *extends to* Y. Let $V(t)$ be a smooth vector field along a smooth curve γ on M, x a chart for M. If $V(t)$ extends to some smooth vector field on TM, then we often use the following notation for such a $V = V(t)$ in order to emphasize that its components at a point are derivatives of γ:

$$V(t) = \sum_i V^i(t)\partial_i := \sum_i \dot{\gamma}^i(t)\partial_i := \dot{\gamma}(t).$$

We can define an affine connection that operates on vector fields along a curve, that becomes identical to our previously defined affine connection for vector fields of M when the curve does not self-intersect.

Definition 2.84. Covariant Derivative along a curve (based on Proposition 5.3.2 and Definition 5.3.3 from [53]). Let M be a smooth manifold with an affine connection ∇, and let $\gamma : I \to M$ be a smooth curve on M. We define the map $D_t : \Gamma^\infty_\gamma(TM) \to \Gamma^\infty_\gamma(TM)$ to satisfy the following:

1. $D_t(V + W) = D_tV + D_tW$ for all V, $W \in \Gamma^\infty_\gamma(TM)$.
2. $D_t(fV) = \frac{d(f\circ\gamma)}{dt}V + fD_tV$ for all $V \in \Gamma^\infty_\gamma(TM)$ and all $f \in C^\infty(M)$.
3. If V extends to a vector field $Y \in \Gamma^\infty_\gamma(TM)$, then $D_tV = \nabla_{\dot{\gamma}(t)}Y$.

D_t is called a *covariant derivative along* γ, and it is unique. Its existence is guaranteed by Proposition 5.3.2 from [53].

The Levi-Civita connection has the property that

$$\frac{d}{dt}G_{\gamma(t)}(V,\, W) = G_{\gamma(t)}(D_tV,\, W) + G_{\gamma(t)}(V,\, D_tW).$$

A retraction is a map that takes a source point on a manifold and a tangent vector in the source's tangent space to a destination point on the manifold. It is an approximation to the more computationally expensive *exponential map* [1]. A *geodesic* is a curve on a manifold that share many properties with straight lines in Euclidean space [53]. Since the points from a smooth curve and their tangent vectors map to a geodesic under the exponential map, the retraction map is a method to approximately produce curves that behave like geodesics. In this thesis, we reserve the symbol $\mathfrak{R}$ to mean a retraction.

Definition 2.85. First-order retractions (Definition 4.1.1 from [1]). Let M be a differentiable manifold. Consider the map $\mathfrak{R} : TM \to M$, and define the short-hand notation

$$\mathfrak{R}_p := \mathfrak{R}|_{T_pM} : T_pM \to M.$$

If $\mathfrak{R}$ satisfies the following for any given tangent vector $X \in TM$ and any $p \in M$, then it is called a *first-order retraction*:

1. $\mathfrak{R}_p(\mathbf{0}) = p$, where $\mathbf{0}$ is the neutral element of T_pM.
2. The differential of $\mathfrak{R}$ at the neutral element $\mathbf{0}$ of T_pM,

$$(d\mathfrak{R}_p)_{\mathbf{0}} : T_{\mathbf{0}}(T_pM) \xrightarrow{\sim} T_{\mathfrak{R}_p(\mathbf{0})}M = T_pM,$$

is the identity map on T_pM. To specify the components of this linear map, we need a basis for the input and output vector spaces (see Section 2.7.1). Let $\mathcal{B}$ be the basis of $T_{\mathbf{0}}(T_pM)$ that is the chart-induced basis of T_pM; this is valid since $T_{\mathbf{0}}(T_pM)$ and T_pM are isomorphic as vector spaces. Then this condition means that the matrix-form of $(d\mathfrak{R}_p)_{\mathbf{0}}$ is the identity matrix with respect to $\mathcal{B}$ and the chart-induced basis of T_pM.

The second condition implies that the curve $\gamma : t \mapsto \mathfrak{R}_p(tX)$ for some given $X \in T_pM$ satisfies $\dot{\gamma}(0) = X$. If we think of the input to γ as time, we can interpret this behaviour as "X is the velocity of the trajectory traced out by γ at time zero". The following type of retractions satisfy a sense of "having no acceleration":

Definition 2.86. Second-order retractions (Equation 5.33 from [1]). Let M be a Riemannian manifold equipped with the Levi-Civita connection. A first-order retraction $\mathfrak{R}$ on TM that satisfies $D_t\dot{\gamma}|_{t=0} = 0$, where $\gamma : t \mapsto \mathfrak{R}_p(tX)$, $\forall X \in T_pM$, is called a *second-order retraction*.

Research on retractions have increased over the past two decades, because when f is a cost function over a complication differentiable M, for a given $p \in M$, we can work with the function $f \circ \mathfrak{R}_p : T_pM \to \mathbb{R}$ where well-studied Euclidean optimization techniques can be used. Some techniques for specifying retractions are found in [1, 45], and we do not go into the details here. The following retraction is used in our thesis:

Example 2.87. Orthographic retraction on the set of fixed-rank matrices (Proposition 24 from [45]). The orthographic retraction $\mathfrak{R}$ on the set of fixed-rank matrices $M = \{K \in \mathbb{R}_{\text{s}}^{n\times m} \mid \text{rank}(K) = r\}$ is defined by

$$\mathfrak{R}_K(Z) = U\begin{bmatrix}\Sigma_0 + A & C \\ B & B(\Sigma_0 + A)^{-1}C\end{bmatrix}V^T,\ \forall K \in M,$$

where

$$K = U\begin{bmatrix}\Sigma_0 & 0 \\ 0 & 0\end{bmatrix}V^T,\ \Sigma_0 \in \mathbb{R}_{\text{s}}^{r\times r}$$

is the singular value decomposition of $K \in M$, and

$$Z = K = U\begin{bmatrix}A & C \\ B & 0\end{bmatrix}V^T,\ A \in \mathbb{R}_{\text{s}}^{r\times r}$$

is the singular value decomposition of the tangent vector $Z \in T_KM \cong \mathbb{R}_{\text{s}}^{n\times m}$. The proof for Proposition 24 in [45] shows that this is a second-order retraction.

Example 2.88. Straight-line retraction. The function $\mathfrak{R}_p(X) = p + X$ is a retraction on the real number line $\mathbb{R}$, where $p \in \mathbb{R}$, $X \in T_p\mathbb{R} \cong \mathbb{R}$.

2.8.3 Riemannian Hessian

The Hessian operator for any function $f \in C^k(\mathbb{R}^m)$, where $k \geq 2$, is a linear map from $\mathbb{R}^m$ to itself that is defined point-wise $\forall p \in \mathbb{R}^m$, by:

$$(\text{Hess}_X f(p))_{\mathcal{E}} := \sum_{i,\,j=1}^{m} \frac{\partial^2 f}{\partial x^i \partial x^j}\bigg|_p X^i b_i,\ \forall X = \sum_{i=1}^{m} X^i b_i \in T_p\mathbb{R}^m, \tag{2.56}$$

where $\{b_i\}_{i=1}^m$ can be any orthonormal basis of $T_p\mathbb{R}^m$, and x here is the identity chart of $\mathbb{R}^m$. The orthogonality of $\{b_i\}$ is defined by some given inner product of $\mathbb{R}^m$. Its definition can be extended to a Riemannian manifold with the Levi-Civita connection as follows:

Definition 2.89. Riemannian Hessian (based on Definition 5.5.1 from [1]). Let (M, G) be a Riemannian manifold equipped with the Levi-Civita connection ∇. Given any $p \in M$, any $f \in C^k(M)$, where $k \geq 2$, the Riemannian Hessian of f at p is

$$\begin{aligned} \text{Hess} f : \Gamma^\infty(TM) &\xrightarrow{\sim} \Gamma^\infty(TM) \\ X &\mapsto \text{Hess}_X f := \nabla_X (df)^\sharp, \end{aligned}$$

where the linearity means that $\text{Hess}_X f(p) : T_pM \xrightarrow{\sim} T_pM, \forall p \in M$.

The Riemannian Hessian has the following properties [1]:

1. $G(\text{Hess}_X f, Y) = X(Yf) - (\nabla_X Y)f$, for all $X, Y \in \Gamma^\infty(TM)$.
2. $G(\text{Hess}_X f, Y) = G(X, \text{Hess}_Y f)$, for all $X, Y \in \Gamma^\infty(TM)$.
3. If $\mathfrak{R}_z$ is a second-order retraction, then $\text{Hess} f(z) = \text{Hess}(f \circ \mathfrak{R}_z(\mathbf{0}))$, for all $z \in M$. The right-hand side is the Euclidean Hessian from Equation 2.56 with the substitution $p \leftarrow \mathbf{0} \in T_zM \cong \mathbb{R}^{\dim M}$, the neutral element of T_zM, and orthogonality determined by G_z.

The last property is usually approximated in practice. Given a retraction $\mathfrak{R}_z$ that is not necessarily a second-order retraction, Section 7.5.1 of [1] is about how

$$\text{Hess} f(z) \approx \tilde{H} := \text{Hess}(f \circ \mathfrak{R}_z(\mathbf{0})) \tag{2.57}$$

is a valid approximate Hessian that meets the convergence theory of the trust-region method in that book. We review the trust-region algorithm later, but we do not go into further details about the approximation of Hessians.

2.9 Embedded Submanifolds

It is often useful to inherit properties from a simpler manifold that contains the manifold of interest. In this section, we review construction techniques that accomplish this task. The material here is used in Section 4.5 to show the derived spaces $\mathcal{P}$, $\mathcal{N}$, and $\mathcal{V}$ are indeed manifolds.

Definition 2.90. Inclusion map. Let $X \subset Y$ be a subset of an arbitrary set Y. We write $X \hookrightarrow Y$ to mean the *inclusion* map from X to Y, which is defined as

$$\begin{aligned} X &\to Y \\ x &\mapsto x. \end{aligned}$$

Definition 2.91. Immersions, submersions, smooth embeddings, and standard embedding. Let $\phi : N \to W$ be a differentiable map between differentiable manifolds.

1. If $d\phi$ is surjective at all $p \in N$, then ϕ is called an *submersion*.
2. If $d\phi$ is injective at all $p \in N$, then ϕ is called an *immersion*.
3. A *smooth embedding* is a smooth immersion that is also a topological embedding (see Definition 2.48).

4. Given a chart map x for $\mathbb{R}^k \subset \mathbb{R}^n$, the map

$$\begin{aligned} \mathbb{R}^k &\underset{\text{em}}{\longrightarrow} \mathbb{R}^n \\ \left(x^1, x^2, \cdots, x^k\right) &\mapsto \left(x^1, x^2, \cdots, x^k, \underbrace{0, 0, \cdots, 0}_{n-k \text{ times}}\right) \end{aligned}$$

is called the *standard embedding of* $\mathbb{R}^k$ *to* $\mathbb{R}^n$.

Example 2.92. Smooth embeddings. The standard embedding and the inclusion map between smooth manifolds are both smooth embeddings.

Definition 2.93. Embedded submanifolds of smooth manifolds. Let M be a smooth manifold, and $S \subseteq M$ be a subset. If S inherits the topology from M, and is equipped with a smooth structure such that the inclusion map $S \hookrightarrow M$ is a smooth embedding under this structure, then S is called *an embedded submanifold of* M.

2.9.1 Construction Methods

We now review some methods for constructing embedded differentiable submanifolds.

2.9.1.1 Construction from Embeddings

The following proposition shows that embedded submanifolds are the images of smooth embeddings:

Proposition 2.94. *Images of embeddings as submanifolds (reworded based on Proposition 5.2 from [59]). Let M and N be smooth manifolds, and $\phi : N \to M$ be a smooth embedding. Then $S = \phi(N)$ is a topological manifold with the inherited topology from M, and it has a unique smooth structure that makes it into an embedded smooth submanifold of M. ϕ in this case is a diffeomorphism between N and S.*

Proof. See [59]. □

If (U, φ) is a chart for N, then $(\phi(U), \varphi \circ \phi^{-1})$ is its induced chart for $S \subset N$. The collection of such charts for S is the unique smooth structure for S. The inclusion map $S \hookrightarrow M$ is a smooth embedding since it is given by

$$\begin{aligned} S &\to M \\ s &\mapsto \phi^{-1} \circ \phi(s) = s. \end{aligned}$$

Therefore, we can interpret an embedded submanifold as an image of a smooth embedding.

2.9.1.2 Construction from Open Subsets

We write m-manifold to mean a manifold that is m-dimensional. Let $(M, \{(W_i, \varphi_i)\})$ be a smooth m-manifold, and let $U \subseteq M$ be any open subset. Then the atlas for U

$$\mathcal{A} = \left\{ \left(W_i \cap U,\ \varphi_i|_{W_i \cap U} \right) \mid V \subseteq U \right\}$$

is a smooth atlas for U. This means any open subset of M is an embedded submanifold of M if it uses an atlas of this construction.

Example 2.95. Full-rank matrices as embedded submanifolds. Consider the set of full-rank matrices

$$\mathbb{R}^{D\times p}_{\text{full, s}} := \begin{cases} \left\{A \in \mathbb{R}^{D\times p}_{\text{col, s}}\right\} & \text{if } D \geq p, \\ \left\{A \in \mathbb{R}^{D\times p}_{\text{row, s}}\right\} & \text{otherwise.} \end{cases}$$

$\mathbb{R}^{m\times n}_{\text{col, s}}$ is an open subset of $\mathbb{R}^{m\times n}_{\text{s}}$ since its complement

$$\left\{A \in \mathbb{R}^{m\times n}_{\text{s}} \mid \det\left(A^T A\right) = 0\right\}$$

is a closed subset. $\mathbb{R}^{m\times n}_{\text{row, s}}$ is also an open subset of $\mathbb{R}^{m\times n}_{\text{s}}$ by a similar development. Therefore, $\mathbb{R}^{m\times n}_{\text{col, s}}$, $\mathbb{R}^{m\times n}_{\text{row, s}}$, and $\mathbb{R}^{m\times n}_{\text{full, s}}$ are all embedded submanifolds of $\mathbb{R}^{m\times n}_{\text{s}}$.

2.9.1.3 Construction from Graphs

Proposition 2.96. *Graphs as submanifolds (Proposition 5.4 from [59]). Suppose M is a smooth m-manifold and N is a n-manifold, $U \subseteq M$ is open, and $f : U \to N$ is a smooth map. The graph of f is the following subset:*

$$\mathcal{G}(f) := \{(x,\, y) \in M \times N \mid x \in U,\, y = f(x)\}.$$

Then $\mathcal{G}(f)$ is an embedded m-dimensional submanifold of $M \times N$.

Proof. See [59]. □

2.9.1.4 Construction from Slices of Charts

Definition 2.97. Slices of charts (taken from [59]). Let M be a smooth m-manifold, and let $(U,\, \varphi)$ be a smooth chart on M. Given a $k \in [0 : m]$, any subset of the form

$$\left\{\left(x^1, \cdots, x^k, x^{k+1}, \cdots, x^m\right) \in U \mid x^{k+1} = c_{k+1}, \cdots,\, x^n = c_n\right\}$$

for some constants c_j, $\forall j \in [k+1 : m]$, is called a *k-dimensional slice of U*.

The following is a method that verifies whether a manifold is an embedded submanifold of another:

Theorem 2.98. *Local slice criterion for embedded submanifolds (reworded based on Theorem 5.8 from [59]). Let $k \in [0 : m]$ and denote by M a m-dimensional smooth manifold. If each point of $S \subseteq M$ is contained in the domain of a smooth chart $(U,\, \varphi)$ for M such that $S \cap U$ is a k-dimensional slice of U, then S is a k-dimensional topological manifold with the inherited topology from M, and it has a smooth structure that makes it into a k-dimensional embedded submanifold of M. The converse is also true.*

Proof. See [59]. □

2.9.1.5 Construction from Level-Sets

Definition 2.99. Regular values. Let $\phi : M_1 \to M_2$ be submersion between two manifolds of dimension m_1 and m_2, $m_1 > m_2$, and let $p \in M_2$. If $\forall x \in \phi^{-1}(\{p\})$, any matrix-form of $(d\phi)_x$ has a row rank of m_2, then p is called a *regular value* of ϕ.

Proposition 2.100. *Submersion theorem (Proposition 3.3.3 from [1]). Let $\phi : M_1 \to M_2$ be a submersion between two manifolds of dimension m_1 and m_2, $m_1 > m_2$, and let $p \in M_2$. If p is a regular value of ϕ, then $\phi^{-1}(\{p\})$ is a closed, $(m_1 - m_2)$-dimensional, embedded differentiable submanifold of M_1.*

2.9.2 Tangent Space Projector

Consider M to be an embedded submanifold of a Riemannian manifold $(W,\ G)$. Since every tangent space of M is a subspace of a tangent space of W, the restriction of G to TM is a valid Riemannian metric for M. On the Riemannian manifold $(M,\ G|_{TM})$, the following relationship holds for any differentiable function $f : W \to \mathbb{R}$:

$$(df|_M)_p^\sharp = \text{Proj}_{T_pM}(df)_p^\sharp,\ \forall p \in M \subset W, \tag{2.58}$$

where the map $\text{Proj}_{T_pM} : T_pW \xrightarrow{\sim} T_pM$ for any $p \in M$ is the orthogonal projector onto T_pM. Denote by $\text{Proj}_{T_pM}^\perp$ the projection operator onto the orthogonal complement of T_pM. It follows that any tangent vector $X \in T_pW$ can be uniquely decomposed by these two projectors:

$$X = \text{Proj}_{T_pM}X + \text{Proj}_{T_pM}^\perp X.$$

Given any affine connection ∇ for the tangent bundle $TW \xrightarrow{\pi} W$, we can map the resultant section from $\Gamma^\infty(TW)$ to $\Gamma^\infty(TM)$ in a point-wise manner: $\forall p \in M$,

$$\left((\nabla_{TM})_X \mu\right)_p = \text{Proj}_{T_pM}(\nabla_X \mu)_p \in T_pM. \tag{2.59}$$

When ∇ is the Levi-Civita connection of $(M,\ G|_{TM})$, this is used to derive an expression for the point-wise action of the Riemannian Hessian with any input vector field $X \in \Gamma^\infty(TM)$:

$$\begin{aligned}(\text{Hess}_X f)_p &= \left((\nabla_{TM})_{X_p}(df|_M)^\sharp\right)_p \\ &= \text{Proj}_{T_pM}\left(\nabla_X (df|_M)^\sharp\right)_p \in T_pM,\ \forall p \in M.\end{aligned} \tag{2.60}$$

2.10 Lie Groups

In this thesis, we are primarily interested in equivalence relations that are in the set called Lie groups. The material covered here is required to understand Theorem 2.108 that is reviewed later in Section 2.11. This theorem is used to show the manifold from Section 4.5.4 is indeed a quotient manifold.

Definition 2.101. Groups and subgroups. Let G be a non-empty set and $\circ_G$ be a product denoted by

$$\begin{aligned}\circ_G : G \times G &\to G \\ (g_1,\ g_2) &\mapsto g_1 \circ_G g_2\end{aligned}$$

that satisfies the following properties:

1. $\circ_G$ is associative: $\forall g_1,\ g_2,\ g_3 \in G,\ (g_1 \circ_G g_2) \circ_G g_3 = g_1 \circ_G (g_2 \circ_G g_3)$.
2. There exists a unique element $e \in G$ such that $e \circ_G g = g \circ_G e$ for all $g \in G$. e is called the *identity element*.
3. $\forall g \in G$, there exists a unique element g^{-1} such that $g^{-1} \circ_G g = g \circ_G g^{-1} = e$ is the identity element. g^{-1} is called the *inverse of* g.

The pair $(G,\ \circ_G)$ is called a *group*, and $\circ_G$ is called a *group product*. Given a subset $H \subset G$, if the product $\circ_G|_{H\times H}$ also satisfies the above three properties for elements in H, then the group $\left(H,\ \circ_G|_{H\times H}\right)$ is called a *subgroup of* $(G,\ \circ_G)$. We also write $H \subset G$ to mean H is a subgroup of G when there is no confusion.

When the group product $\circ_G$ is clear from context, we write G to mean $(G, \circ_G)$. If the group product satisfies $g_1 \circ_G g_2 = g_2 \circ_G g_1$ for all $g_1, g_2 \in G$, then G is a *commutative group*. It is common to write $g_1 g_2$ to mean $g_1 \circ_G g_2$ when its meaning is clear from context. When G is a commutative group, it is common convention to write $g_1 + g_2$ to mean $g_1 \circ_G g_2$, write 0 to mean the identity element, and write $-g$ to mean the inverse element of g. The addition and multiplication on vector spaces with field $\mathbb{R}$ can be formulated in terms of groups, but we do not use the group product notation for them. In this thesis, we only use group-specific notations when we work with the following type of groups:

Definition 2.102. Lie group. A group $(G, \circ_G)$ is called a *Lie group* if it satisfies the following:

1. G is a smooth manifold.
2. The following smooth map exists:

$$\begin{aligned} \upsilon : G &\to G \\ g &\mapsto g^{-1}. \end{aligned}$$

Example 2.103. Translation group. Let $+$ be the vector space addition of $\mathbb{R}^m$. The commutative Lie group $(\mathbb{R}^m, +)$ is called the *m-dimensional translation group*.

Example 2.104. General linear group. For any basis $\mathcal{B}$ for the tensor space $\mathcal{T}_1^1\mathbb{R}^m$, if

$$G = \left\{ A : \mathbb{R}^m \xrightarrow{\sim} \mathbb{R}^m \mid \det(A)_{\mathcal{B}} \neq 0 \right\}$$

and $\circ_{\text{mat}}$ is the matrix multiplication of the matrix-form of two group elements, then $(G, \circ_{\text{mat}})$ is a Lie group that is called the *m-dimensional real general linear group*. It is denoted by $\text{GL}(m, \mathbb{R})$.

Example 2.105. Orthogonal group. Let $\langle \cdot, \cdot \rangle$ be an inner product for $\mathbb{R}^m$. If

$$G = \left\{ A : \mathbb{R}^m \xrightarrow{\sim} \mathbb{R}^m \mid \langle A(v), A(w) \rangle = \langle v, w \rangle, \forall v, w \in \mathbb{R}^m \right\},$$

then $(G, \circ_{\text{mat}})$ is a Lie group that is called the *orthogonal group*. It is a subgroup of the general linear group, and is denoted by $\text{O}(m, \mathbb{R})$.

Let $(G, \circ_G)$ be a Lie group and M a differentiable manifold. The *left group action on M associated with G* is defined as the map

$$\begin{aligned} \triangleright : G \times M &\to M \\ (g, p) &\mapsto g \triangleright p \end{aligned}$$

such that the following are satisfied for all $p \in M$:

1. $e \triangleright p = p$ where $e \in G$ is the identity element.
2. $g_2 \triangleright (g_1 \triangleright p) = (g_2 \circ_G g_1) \triangleright p, \forall g_1, g_2 \in G$.

Example 2.106. The natural left action of the general linear group and its subgroups on $\mathbb{R}^m$. The left group action $\triangleright : \text{GL}(m, \mathbb{R}) \times M \to M$ on $M = \mathbb{R}^m$ that is given by the point-wise specification

$$(A \triangleright p)_{\mathcal{B}} = (A)_{\mathcal{B} \otimes \mathcal{B}^*} (p)_{\mathcal{B}}, \ A \in \text{GL}(m, \mathbb{R}), \ x \in M = \mathbb{R}^m,$$

is called the *natural left action of GL*$(m, \mathbb{R})$. This is a continuous group action. Given a subgroup $G \subset \text{GL}(m, \mathbb{R})$, its natural left group action is the restriction of the natural left action of $\text{GL}(m, \mathbb{R})$ to $G \times \mathbb{R}^m \to \mathbb{R}^m$.

The *right group action on M associated with G* is similarly defined as the map

$$\begin{aligned} \triangleleft : M \times G &\to M \\ (p,\, g) &\mapsto p \triangleleft g \end{aligned}$$

such that the following are satisfied for all $p \in M$:

1. $p \triangleleft e = p$ where $e \in G$ is the identity element.
2. $(p \triangleleft g_1) \triangleleft g_2 = (g_2 \circ_G g_1) \triangleright p, \forall g_1,\, g_2 \in G$.

Given a left action $\triangleright$, a right action can be constructed from it in the following manner:

$$\begin{aligned} \triangleleft : M \times G &\to M \\ (p,\, g) &\mapsto p \triangleleft g := g^{-1} \triangleright p. \end{aligned}$$

We can similarly construct a left group action from a given right group action. When a left and right group action have this relationship, we say one is *induced* from another, or that they are *canonically related*.

Example 2.107. When G is a Lie group that is also a set of bijective maps from M to itself, G is called a *Lie transformation group*. In this case, if we define $g \triangleright p := g(p)$, then the induced right group action is a way to write the inverse map of g:

$$p \triangleleft g := g^{-1} \triangleright p = g^{-1}(p).$$

Here are some standard terminologies regarding group actions. Let $p \in M$, and $\triangleright$ be a left group action on M.

- The following set is called an *orbit relative to G under $\triangleright$ at p*:

$$\text{Orb}_p := \{u \in M \mid \exists g \in G : g \triangleright p = u\} \subseteq M.$$

- The following set is called a *stabilizer relative to G under $\triangleright$ at p* or an *isotropy group at p*:

$$\text{Stab}_p := \{g \in G \mid g \triangleright p = p\} \subseteq G.$$

- The following set is the *quotient of M by G*, also called the *orbit space*:

$$M/G := \{\text{Orb}_p \mid p \in M, \}.$$

- If $\forall p \in M$, $\text{Stab}_p = \{e\}$, where e is the identity element of G, then the $\triangleright$ associated with the stabilizers is called a *free action*.

- A map is a *proper map* if the preimage of any compact set is compact. If the map

$$\begin{aligned} G \times M &\to M \times M \\ (g,\, p) &\mapsto (g \triangleright p,\, m) \end{aligned}$$

  is a proper map, then $\triangleright$ is a *proper action*.

- A left group action $\triangleright$ (or right group action $\triangleleft$) is said to be *transitive* if $\forall p,\, u \in M$, $\exists g \in G$ such that $g \triangleright p = q$ (or $p \triangleleft g = u$). This is equivalent to requiring $\text{Orb}_p = M, \forall p \in M$. In this case, M is called a *homogeneous space with action* $\triangleright$ (or $\triangleleft$).

For a fixed $p \in M$, we can interpret an orbit or stabilizer as a subset of M or G, respectively, that leaves $g \triangleright p$ invariant in some sense. When a left group action is free, then every point on M is *stabilized* by the identity element of G. Orbit spaces are the type of quotient spaces that we focus on in this thesis. Later, we shall treat a homogeneous space as a space of equivalence classes, where the equivalence relation is given by the transitive group action associated with the homogeneous space.

2.11 Quotient Manifolds

The concepts reviewed in this section is required to understand Section 4.5, and Theorem 2.108 is used to show that the manifold proposed there is a quotient manifold. The example at the end of this section is used to give a concrete example of the tangent space projectors for such manifolds, but it is not used elsewhere in this thesis.

Given a smooth manifold W, we can equipped it with an equivalence relation $\sim$ to construct a new manifold. This section of our review is based on the presentation found in [1]. Consider the quotient of W (as a set) by a given $\sim$;

$$W/\sim := \{[z] \mid z \in W\}.$$

Recall from Section 2.6 that the quotient map is

$$\begin{aligned} \pi : W &\to W/\sim \\ u &\mapsto \pi(u) := [u]. \end{aligned} \tag{2.61}$$

Each element of this set is a subset of W. Given a $u \in W$, we use the following convention to distinguish a point in $W/\sim$ and its corresponding subset in W:

- The point in $W/\sim$ that corresponds to $[u]$ is denoted by $\pi(u)$ as opposed to $[u]$ when we want to emphasize that the points are from $W/\sim$.
- The expressions on both sides of the equation

  $$[u] = \pi^{-1} \circ \pi(u)$$

  are both used to denote the subset of points on W that is the equivalence class of u.

Our goal is to make $W/\sim$ into a differentiable manifold, and this requires π to be a submersion [1]. Propositions 3.4.2 to 3.4.4 from [1] claims that this is ensured by imposing some constraints on the graph of the equivalence relation $\sim$. We use the following theorem since we work with orbit spaces instead of generic quotient spaces:

Theorem 2.108. *Quotient manifold theorem (Theorem 21.10 from [59]). Suppose G is a Lie group acting smoothly, freely, and properly on a smooth manifold W. Then the orbit space W/G is a topological manifold of dimension equal to $\dim W - \dim G$, and has a unique smooth structure with the property that the quotient map $\pi : W \to W/G$ is a smooth submersion.*

Proof. See [59]. □

The manifold W/G that satisfies Theorem 2.108 is called a *quotient manifold*. There are other quotient manifolds that do not require the equivalence relation to be specified by a Lie group action, but we do not discuss them here. In the case of Theorem 2.108, it is common to treat the objects W, π, and W/G collectively as the fibre bundle $W \xrightarrow{\pi} W/G$. This bundle and any other bundle that is isomorphic to it, is called a *principal G-bundle*. Therefore, the space W in a quotient manifold setup that is obtained using Theorem 2.108 is also called the *total space*. Note that in most differentiable manifold theory literature, principal G-bundles are usually specified by a right group action. This right action is induced from the left group action that constructed the orbit space W/G in the setup presented here.

In this thesis, we focus on the case when $\dim(W/G) < \dim W$, which has the property that each equivalence class $\pi^{-1} \circ \pi(u)$, $u \in W$, is an embedded submanifold of W, since it is the fibre of the principal G-bundle $W \xrightarrow{\pi} W/G$. If we treat each equivalence class as a homogeneous space, we see that we can map

any element of $[u]$, u_1, to any element in $[u]$, u_2, by $g \triangleright u_1 = u_2$, where $g \in G$ is guaranteed to exist. We can think of this as a way to move within $[u]$; i.e., move along the fibre

$$\text{fib}_p(W) = \pi^{-1} \circ \pi(u).$$

Some orbit spaces such as $\mathbb{R}^m/\text{GL}(m, \mathbb{R})$ and $\mathbb{R}^m/\text{O}(m, \mathbb{R})$ with its natural group action are not manifolds (for example, see Example 21.2 from [59]). However, when the origin is removed from $\mathbb{R}^m$, the orbit spaces $\{\mathbb{R}^m \setminus \{0\}\}/\text{GL}(m, \mathbb{R})$ and $\{\mathbb{R}^m \setminus \{0\}\}/\text{O}(m, \mathbb{R})$ with its natural group action are manifolds.

Example 2.109. Grassmann manifolds (based on Section 3.4.4 from [1]). Recall from Example 2.95 that the manifold $\mathbb{R}^{m\times n}_{*,\text{s}}$, $m \geq n$, does not contain the origin. Consider the orbit space $\mathbb{R}^{m\times n}_{*,\text{s}} \xrightarrow{\pi} \mathbb{R}^{m\times n}_{*,\text{s}}/\text{GL}(n, \mathbb{R})$ that uses the natural left group action of $\text{GL}(n, \mathbb{R})$. It describes the set of full column rank matrices that span the same subspace, which is just a singleton set if $n = m$. It follows from Proposition 3.4.6 and its proof from [1] that $\big(\mathbb{R}^{m\times n}_{*,\text{s}}/\text{GL}(n, \mathbb{R}), \text{colmaj}\big)$ is a quotient manifold. This manifold is called the *Grassmann manifold of size* $m \times n$.

For the rest of this section, we consider the principal G-bundle setup $W \xrightarrow{\pi} M = W/G$ where the orbit space W/G was constructed using some smooth, free, and proper left Lie group action $\triangleright$. The representation of a point $p \in M$ and a tangent vector in T_pM is a tricky issue. If M was isomorphic as a manifold to another manifold K that is not a collection of equivalence classes, then we can simply work with K instead of M. For example, the Stiefel manifold is presented as an embedded submanifold that does not use a quotient manifold setup in [1], and presented as a quotient manifold in [62]. In this case, one can simply work with the embedded submanifold version of the Stiefel manifold for simplicity.

Quotient manifolds are used when its the isomorphism to a simpler non-quotient manifold is unknown. In this case, we need to choose a representative element $u \in \text{fib}_p(W)$ to represent p. This implies that we need to work with T_uW to represent vectors from T_pM. Given a sense of orthogonality on T_uW, a unique subspace $\text{Hor}_uW \subset T_uW$ can be identified such that vectors in Hor_uW are isomorphic to vectors in T_pM. When Hor_uW and T_pM are isomorphic as vector spaces, there exists a unique $Y \in \text{Hor}_uW$ such that the push-forward of π at u acting on Y is X; i.e., $(d\pi)_u Y = X$. Therefore, given u, we can derive Y to be used as the representative element for X. This subspace Hor_uW is called the *horizontal space of* TW *at* u. Its orthogonal complement is called the *vertical space of* TW *at* u, and is denoted by Vert_uW. In this thesis, we focus on the scenario when (W, H) is a Riemmanian manifold, and its metric tensor at u provides the orthogonality that is used to specify the horizontal and vertical spaces. This implies that different Riemannian metrics induce different representative elements from T_uW of a given $X \in T_pM$, provided that a representative element u is already chosen for p.

The vertical space of TW does not require a Riemannian metric on W for its definition, but requires a point $p \in M$ from the quotient manifold. It is given by

$$\text{Vert}_uW := T_u\big(\pi^{-1}(\{p\})\big). \tag{2.62}$$

Equivalently, we have

$$(d\pi)_u X = 0 \iff X \in \text{Vert}_uW \subset T_uW.$$

From this definition, we can interpret tangent vectors in Vert_uW to be vectors that are tangent to the group G. Given a Riemannian metric H for W, the horizontal space of TW at $u \in \text{fib}_p(W) \subset W$ is defined as:

$$\text{Hor}_uW := \{Y \in T_uW \mid H_u(\eta, Y) = 0, \forall \eta \in \text{Vert}_uW\}. \tag{2.63}$$

One should be aware that using the horizontal space without first specifying a way to pick a representative element for $[u]$ does not produce a unique representative element for X. This is because if we change our

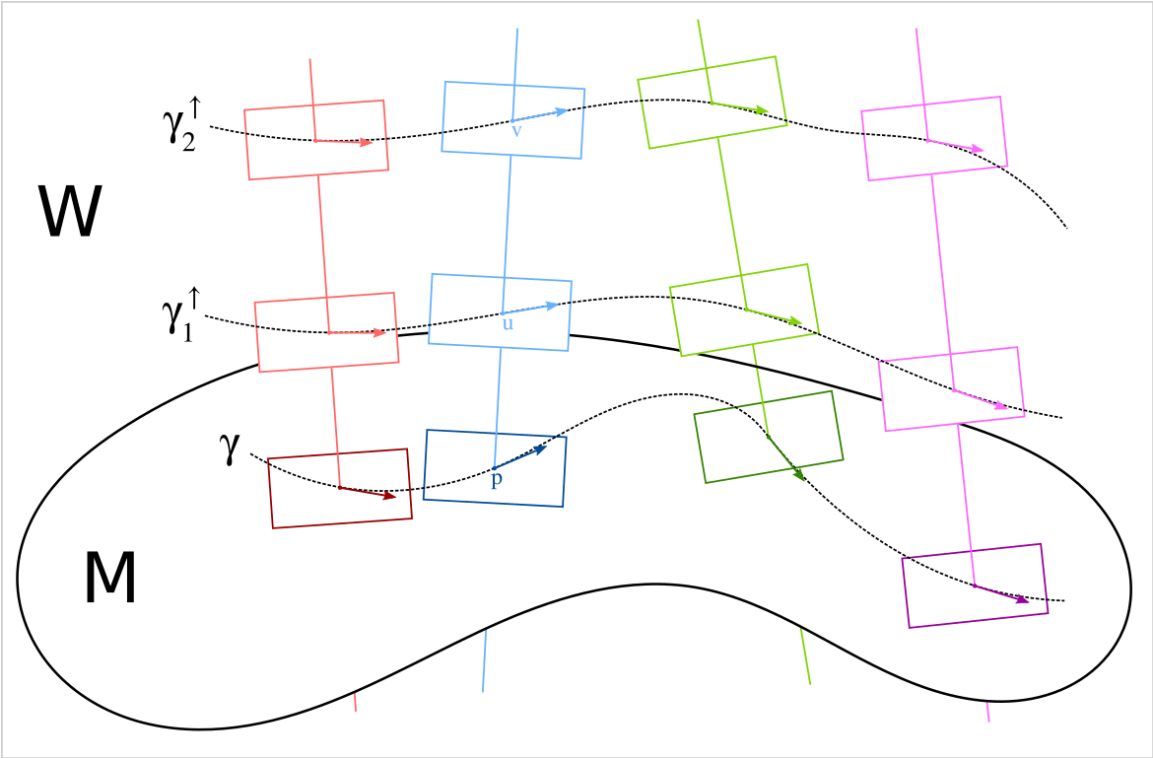


Figure 2.4: $\gamma_1^{\uparrow}$ and $\gamma_2^{\uparrow}$ are horizontal lifts of the curve $\gamma : \mathbb{R} \to M$. The tangent spaces of four points on M are coloured in a darker version of its corresponding horizontal spaces along its corresponding fibres.

representative element from u to another point $u_2 \in \text{fib}_p(W)$, then our representative element of X is not even in T_uW, but in some subspace of $T_{u_2}W$. In other words, if we do not keep track of which tangent space we are in, or equivalently, if we do not keep track of which element of $\text{fib}_{\pi(p)}(W)$ is used to represent p, then we do not have enough information to select a representative element for X. This behaviour is important during numerical implementation of a scheme that deals with tangent vectors from a quotient manifold, especially when we want to characterize a curve γ on M. This is summarized in Figure 2.4: the tangent space at a point on M (shown in darker colours) can be represented by any horizontal space along the fibre that contains that point on M (shown in brighter colours). For example, let the blue tangent vector be $X \in T_pM$, and it can be represented by any of the bright blue tangent vectors in TW (only two are shown). Specifying a way to pick a representative element from fib_pW is equivalent to specifying a way to ignore all but one bright blue tangent vector, and we use that vector as our representative element of the blue vector in TM. If we choose u as the representative element, then the resultant representative vector is called the *horizontal lift of X at u*, denoted by $X_u^{\uparrow} \in \text{Hor}_uW$. If we choose v as the representative element, then we call the resultant representative vector *the horizontal lift of X at v*, denoted by $X_v^{\uparrow} \in \text{Hor}_vW$. The curves $\gamma_1^{\uparrow}$ and $\gamma_2^{\uparrow}$ are two possible representative elements of the curve γ, because the vector field along those two curves are representative elements of the vectors in the vector field along γ. The curves $\gamma_1^{\uparrow}$, $\gamma_2^{\uparrow}$, and any other curves in W with this property are each called a *horizontal lift of γ*.

The quotient manifold M cannot inherit the Riemannian metric H on W unless H satisfies the following $\forall p \in M$:

$$H_{u_1}\left(X_{u_1}^{\uparrow}, Y_{u_1}^{\uparrow}\right) = H_{u_2}\left(X_{u_2}^{\uparrow}, Y_{u_2}^{\uparrow}\right), \ \forall X, Y \in T_pM, \ \forall u_1, u_2 \in \text{fib}_p(W), \tag{2.64}$$

where the horizontal space used to perform the horizontal lift is specified by H. If H meets this requirement,

then

$$G_p(X,\,Y) := H_u\big(X_u^{\uparrow},\,Y_u^{\uparrow}\big),\ u \in \text{fib}_p(W),\ \forall p \in M \tag{2.65}$$

is a point-wise definition of a Riemannian metric G for M, and is called the *induced Riemannian metric from* H.

Let ∇ be the Levi-Civita connection for the Riemannian manifold $(W,\,H)$, and $(M,\,G)$ be a Riemannian quotient manifold where G is induced from H. The Levi-Civita connection $\tilde{\nabla}$ for the tangent bundle TM is given by Proposition 5.3.3 from [1] as follows:

$$\begin{aligned}\tilde{\nabla} : \Gamma^\infty(TM) \times \Gamma^\infty(TM) &\to \Gamma^\infty(TM)\\ (X,\,Y) &\mapsto \tilde{\nabla}_X Y,\ \text{such that } \Big(\tilde{\nabla}_X Y\Big)^{\uparrow} := \text{Proj}_{\mathfrak{H}_A}\big(\nabla_{X^\uparrow} Y^\uparrow\big) \in \Gamma^\infty(TW),\end{aligned}$$

where $\mathfrak{H}_A := \text{Hor}_A W$ here.

Let K be a differentiable manifold. A map $f : M \to K$ can induce a map $h : W \to K$ via the pull-back π^* in the following way:

$$h = \pi^* f := f \circ \pi.$$

Such a function h is invariant on $\text{fib}_p(W)$ for every $p \in M$. Since π is a smooth map, such a f is smooth if and only if $\pi^* f$ is a smooth map between manifolds. When $K = \mathbb{R}$ and f is differentiable, we can take represent the $\sharp$-isomorphism of the gradient covector of f at any $p \in M$ by

$$(df)_p^\sharp = (dh)_u^\sharp \in \text{Hor}_u W,\ u \in \text{fib}_p(W). \tag{2.66}$$

In numerical optimization, one sometimes need to modify $(df)^\sharp$, so having a projection that map vectors from $T_u W$ to $\text{Hor}_u W$ is useful in these situations.

Example 2.110. The horizontal space projector for Grassmann manifolds of a particular Riemannian metric (Example 3.6.4 from [1]). Let $m > n$, and take the Riemannian metric H for $\mathbb{R}_{*,\,\text{s}}^{m\times n}$ to be

$$H_A(X,\,Y) = \text{tr}\Big(\big(A^TA\big)^{-1}X^TY\Big),\ \forall A \in \mathbb{R}_{*,\,\text{s}}^{m\times n}.$$

The vertical space at a given $A \in \mathbb{R}_{*,\,\text{s}}^{m\times n}$ is the tangent space of $\text{fib}_A\big(\mathbb{R}_{*,\,\text{s}}^{m\times n}\big)$, which is

$$\text{Vert}_A\mathbb{R}_{*,\,\text{s}}^{m\times n} = \big\{AB \ \mid\ B \in \mathbb{R}_{\text{s}}^{n\times n}\big\}.$$

This means that the horizontal space is

$$\text{Hor}_A\mathbb{R}_{*,\,\text{s}}^{m\times n} = \big\{B \in \mathbb{R}_{*,\,\text{s}}^{m\times n} \ \mid\ A^TB = 0\big\}.$$

Let $U \in \mathbb{R}_{*,\,\text{s}}^{m\times n}$ and $V \in \mathbb{R}_{\text{s}}^{m\times(m-n)}$ be such that $U^TV = 0$ and $V^TV = I_{m-n}$. This means V is an orthonormal basis for the orthogonal complement of colspan(U), i.e., colspan$(U)^\perp$ or colspan(V). We then have

$$\begin{aligned}\text{Proj}_{\text{colspan}(U)} : \mathbb{R}_{*,\,\text{s}}^{m\times n} &\xrightarrow{\sim} \text{colspan}(U) \subset \mathbb{R}^{m\times m}\\ B &\mapsto \text{Proj}_{\text{span}(U)}B = Z,\ (Z)_{\mathcal{E}} := U\big(U^TU\big)^{-1}U^TB,\end{aligned}$$

and

$$\begin{aligned}\text{Proj}_{\text{colspan}(U)^\perp} : \mathbb{R}_{*,\,\text{s}}^{m\times n} &\xrightarrow{\sim} \text{colspan}(U)^\perp \subset \mathbb{R}^{m\times m}\\ B &\mapsto \text{Proj}_{\text{span}(U)^\perp}B = Z,\ (Z)_{\mathcal{E}} := \Big(I_m - U\big(U^TU\big)^{-1}U^T\Big)B = VV^TB.\end{aligned}$$

Let $\mathfrak{H}_A := \text{Hor}_A\mathbb{R}_{*,\,\text{s}}^{m\times n}$. It follows that

$$\begin{aligned}\text{Proj}_{\mathfrak{H}_A} : T_A\mathbb{R}_*^{m\times n} &\xrightarrow{\sim} \big(\text{Vert}_A\mathbb{R}_{*,\,\text{s}}^{m\times n}\big)^\perp \subset T_A\mathbb{R}_{*,\,\text{s}}^{m\times n}\\ X &\mapsto \text{Proj}_{\mathfrak{H}_A}X = Z,\ (Z)_{\mathcal{E}} := \Big(I_m - A\big(A^TA\big)^{-1}A^T\Big)(X)_{\mathcal{E}}.\end{aligned}$$

2.12 Riemannian Optimization

In this section, we review the gradient descent, conjugate-gradient, and trust-region algorithms that are described in [1]. These topics are required to understand Section 4.6.3. Given a search direction η_k for iteration k, we modify the gradient descent algorithm from Algorithm 1 of [1] to use an exact line-search method:

$$\alpha_k = \underset{\tilde{\alpha}_k \in \mathbb{R}}{\arg\min} . f \circ \mathfrak{R}_{p_k}(\tilde{\alpha}_k \eta_k). \tag{2.67}$$

The following is the gradient descent algorithm that we use in this thesis:

Algorithm 2.2 Gradient descent, based on Algorithm 1 of [1].

Require: p_0 and a way to compute α_k.

1: $k = 0$.
2: **while** Stopping criteria not reached **do**
3: $\quad \eta_k = -(df)^\sharp_{p_k}$.
4: $\quad$ Compute a step size α_n by minimizing the exact line-search cost function from Equation 2.67.
5: $\quad p_{n+1} = R_{p_k}(\alpha_k \eta_k)$.
6: $\quad k = k + 1$.
7: **end while**
8: Return p_k.

For the rest of this section, we review some other Riemannian optimization methods from [1].

2.12.1 The Conjugate-Gradient Algorithm

The conjugate-gradient algorithm requires us to combine tangent vectors from different iterations together. This combination is vector addition under the Euclidean optimization setting. Special treatment is required under the manifold optimization setting, since the tangent spaces of these vectors are different in general. The *vector transport map* that is described in Equation 8.6 of [1] is used in this thesis to treat this situation. For a given $p \in M$ and a given $X \in T_pM$, it is defined as

$$\begin{aligned} \mathrm{VT}_{\mathfrak{R}_p, X} : T_pM &\to T_{\mathfrak{R}_p(X)}M \\ Y &\mapsto \frac{d}{dt}\mathfrak{R}_p(X + tY)\Big|_{t=0}, \end{aligned} \tag{2.68}$$

where $\mathfrak{R}_p$ is a retraction for the manifold M at point $p \in M$. Any vector transport map can be interpreted as an approximation to the *parallel transport map* [1], and we do not review its theory in this thesis. The conjugate-gradient algorithm that is used in this thesis is shown in Algorithm 2.3.

This algorithm requires a line-search method to find α_n. We do not review line-search methods since any univariate optimization algorithms can be used. We use two methods for updating β_n, and they are both from [1]. The *Fletcher-Reeves* method is

$$\beta_{k+1} = \frac{G_{p_{k+1}}\left((df)^\sharp_{p_{k+1}}, (df)^\sharp_{p_{k+1}}\right)}{G_{p_k}\left((df)^\sharp_{p_k}, (df)^\sharp_{p_k}\right)}, \tag{2.69}$$

and the *Polak-Ribière* method is

$$\beta_{k+1} = \frac{G_{p_{k+1}}\left((df)^\sharp_{p_{k+1}}, (df)^\sharp_{p_{k+1}} - \mathrm{VT}_{\mathfrak{R}_{p_k}, \alpha_k \eta_k}\left((df)^\sharp_{p_k}\right)\right)}{G_{p_k}\left((df)^\sharp_{p_k}, (df)^\sharp_{p_k}\right)}. \tag{2.70}$$

Algorithm 2.3 Conjugate-gradient, taken from Algorithm 13 of [1].

Require: p_0 and a way to compute $\eta_0 = -(df)^\sharp$, $\mathrm{VT}_{\mathfrak{R}_{p,\,\alpha_k\eta_k}}(\eta_k)$, α_k, and β_k.

$\eta_0 = -(df)^\sharp_{p_0}$.
$k = 0$.
while Stopping criteria not reached **do**
    Compute a step size α_n.
    $p_{n+1} = R_{p_k}(\alpha_k\eta_k)$.
    Compute $beta_{k+1}$.
    $\eta_{k+1} = -(df)^\sharp_{p_{k+1}} + \beta_{k+1}\mathrm{VT}_{\mathfrak{R}_{p,\,\alpha_k\eta_k}}(\eta_k)$.
    $k = k + 1$.
end while

Return p_k.

2.12.2 The Trust-Region Algorithm

The trust-region method that is used in this thesis is presented in Algorithm 2.4. Consider the function $\mathcal{L}_k : T_{p_k}M \cong \mathbb{R}^{\dim M} \to \mathbb{R}$ that is given by

$$\mathcal{L}_k(\eta) = f(p_k) + G_{p_k}\Big((df)^\sharp_{p_k},\, \eta\Big) + \frac{1}{2}G_{p_k}(B_k\eta,\, \eta), \tag{2.71}$$

where $B_k : T_{p_k}M \xrightarrow{\sim} T_{p_k}M$ is a symmetric linear map. For each iteration k, this algorithm computes a $s_k \in \mathbb{R}_{++}$ called the *trust-region radius for iteration* k, and it is used to define an inner optimization problem

$$\begin{aligned} &\min_{\eta \in T_{p_k}M} \mathcal{L}_k(\eta) \\ &\text{subject to } G_{p_k}(\eta,\, \eta) \leq s_k^2. \end{aligned} \tag{2.72}$$

This optimization problem is called the trust-region subproblem, and we can use any conventional optimization algorithm for Euclidean optimization to solve it. Let η_k be an approximate solution of Equation 2.72. After computing η_k, Algorithm 2.4 decides whether to accept the computed solution for η and update $p_{k+1} = \mathfrak{R}_{p_k}(\eta_k)$. It does this by comparing the following ratio against given threshold values:

$$\rho_k := \frac{f(p_k) - f \circ \mathfrak{R}_{p_k}(\eta_k)}{\mathcal{L}_k(0) - \mathcal{L}_k(\eta_k)}. \tag{2.73}$$

We call this ratio the *trust-region score for iteration* k in this thesis. In a practical implementation, one should return an error flag to the calling function if division by zero is encountered. In this thesis, we use $B_k = \tilde{H}_k$, where $\tilde{H}_k$ is the approximate Hessian from Equation 2.57 for iteration k.

One can numerically solve Equation 2.72 by using a general-purpose solver for Euclidean spaces, or re-formulate the subproblem as a simpler equivalent problem; [63] provides a survey of these approaches. One can also compute a η_k that is not the optimal solution of Equation 2.72, but still ensures that Algorithm 2.4 satisfies some convergence results [1]. We used the truncated conjugate-gradient method from Algorithm 11 of [1] (reproduced in Algorithm 2.5) to compute such an η_k in this thesis. This algorithm requires us to solve for the positive root of the following quadratic equation, where $\tau \in \mathbb{R}$ is the variable:

$$\tau^2 G_{p_k}(X_j,\, X_j) + 2\tau G_{p_k}(\eta_j,\, X_j) = s_k^2 - G_{p_k}(\eta_j,\, \eta_j). \tag{2.74}$$

Algorithm 2.4 High-level trust-region algorithm. Taken from Algorithm 10 of [1].

Require: $s_{\max} \in \mathbb{R}_{++}$, $0 \leq \rho_{\text{accept}} < 0.25$.

$k = 0$.
while termination condition not met **do**
  Solve η_k so that it is an approximate solution of Equation 2.72.
  Compute ρ_k.
  if $\rho_k < 0.25$ **then**
    $s_{k+1} = 0.25 s_k$.
  else
    if $\rho_k > 0.75$ and $G_{p_k}(\eta_k,\, \eta_k) = s_k^2$ **then**
      $p_{k+1} = \mathfrak{R}_{p_k}(\eta_k)$.
    else
      $p_{k+1} = p_k$.
    end if
  end if
  $k = k + 1$.
end while
return p_k.

Algorithm 2.5 Truncated conjugate-gradient algorithm.

Require: Current solution iterate p_k, Hessian approximate .

$\eta_0 = 0$.
$Y_0 = (df)^{\sharp}_{p_k}$.
$X_0 = -Y_0$.
$j = 0$.
while maximum allowed iterations not reached **do**
  if $G_{p_k}\left(X_j,\, \tilde{H}_k X_j\right) \leq 0$ **then**
    Solve for the positive root in Equation 2.74 for τ.
    return $\eta_k = \eta_j + \tau X_j$.
  end if
  $\alpha = \frac{G_{p_k}\left(Y_j, Y_j\right)}{G_p\left(X_j, \tilde{H}_k X_j\right)}$.
  $\eta_{j+1} = \eta_j + \alpha X_j$.
  if $G_{p_k}(\eta_{j+1},\, \eta_{j+1}) \geq s_k^2$ **then**
    Solve for the positive root in Equation 2.74 for τ.
    return $\eta_k = \eta_j + \tau X_j$.
  end if
  $Y_{j+1} = Y_j + \alpha \tilde{H}_k X_j$.
  $\beta_{j+1} = \frac{G_{p_k}\left(Y_{j+1}, Y_{j+1}\right)}{G_{p_k}\left(Y_j, Y_j\right)}$.
  $X_{j+1} = -Y_{j+1} + \beta_{j+1} X_j$.
  $j = j + 1$.
end while
Notify the calling function that this algorithm failed to find a solution.

The Rendl-Wolkowicz algorithm [47] re-formulates the trust-region subproblem as the following univariate optimization problem:

$$\max_{t\in\mathbb{R}} \left(s_k^2+1\right)\lambda_{\min}(t) - t, \tag{2.75}$$

where $\lambda_{\min}(t^*)$ is the minimum eigenvalue of

$$D(t) := \begin{bmatrix} t & -a^T \\ -a & A \end{bmatrix}.$$

The matrix A and column matrix a is from the following form of the trust-region subproblem that [47] focused on:

$$\begin{aligned} &\min_{x\in\mathbb{R}^m_s} x^T A x - 2a^T x \\ &\text{subject to } x^T x \le s_k^2. \end{aligned} \tag{2.76}$$

Here, A is replaced by $\frac{(A+A^T)}{2}$ if it is not a symmetric matrix. Let x^*be the solution to Equation 2.76. Denote by $v_{\min}(t^*) \in \mathbb{R}^{m+1}$ the normalized eigenvector for $\lambda_{\min}(t^*)$. By Theorem 3.6 of [63], if $\lambda_{\min}(t^*) < 0$, we have

$$x^* = \frac{1}{(v_{\min}(t^*))_1} \begin{bmatrix} (v_{\min}(t^*))_2 \\ (v_{\min}(t^*))_3 \\ \vdots \\ (v_{\min}(t^*))_m \end{bmatrix}, \tag{2.77}$$

otherwise we have

$$x^* = A^{-1}a. \tag{2.78}$$

The x^* here is the component-form of η_k with respect to some basis $\mathcal{B}$. The matrix A and column matrix a need to be specified according to $\mathcal{B}$. In Section 4.6.1, we discuss the details of converting the trust-region subproblem in Equation 2.72 to the subproblem in Equation 2.76.

2.13 Multivariate Polynomials and Quasi-Hankel Matrices

In this section, we develop notations for polynomials that are used in our presentation. We also introduce our definition of a quasi-Hankel matrix. This definition is more restrictive than others in the literature, but it should be less confusing to readers who are unfamiliar with this topic. These matrices are used frequently in Chapter 4. Additional details of the material reviewed here can be found in the references [64, 39, 29].

Given a tuple $x := (x_1, x_2, \cdots, x_D)$ of real-valued variables, a *polynomial term* of these variables has the form

$$cx^\alpha,\ c \in \mathbb{R}\setminus\{0\},\ \alpha \in \mathbb{Z}_+^D,$$

Here, we use the multi-index notation where $x^\alpha = \prod_{d=1}^D x_d^{\alpha_d}$. The object x^α is a *monomial*, and its *degree* is the sum of the entries in α. We reserve the following symbol to mean the set of D-variate monomials:

$$\mathbb{M}_D := \left\{ x^\alpha = \prod_{d=1}^D x_d^{\alpha_d} \ \mid\ \alpha \in \mathbb{Z}_+^D \right\}.$$

We reserve the following symbol to mean the set of D-variate monomials with degrees that are at most L:

$$\mathbb{M}_{D,\,L} := \left\{ x^\alpha = \prod_{d=1}^D x_d^{\alpha_d} \ \mid\ \alpha \in [0:L]^D \subset \mathbb{Z}_+^D,\ |\alpha| = \sum_{d=1}^D \alpha_d \le L \right\}.$$

We also reserve the following symbol to mean the set of D-variate monomials with degree L:

$$\mathbb{M}_{D,\,L,\,\text{hom}} := \left\{ x^\alpha = \prod_{d=1}^{D} x_d^{\alpha_d} \;\middle|\; \alpha \in [0:L]^D \subset \mathbb{Z}_+^D,\, |\alpha| = \sum_{d=1}^{D} \alpha_d = L \right\}$$
$$= \left\{ x^\alpha = \prod_{d=1}^{D} x_d^{\alpha_d} \;\middle|\; \alpha \in \mathbb{A}(D,\,L) \right\}.$$

See Equation 2.3 for the definition of $\mathbb{A}(D,\,L)$.

2.13.1 Polynomials as Vectors

A *polynomial* is a finite sum of polynomial terms. The degree of a polynomial is the largest monomial degree out of each of its terms. In this thesis, we denote by $\mathbb{P}_{D,\,L}$ the space of D-variate polynomials of degree L. Elements of $\mathbb{M}_D$ can be used to represent a D-variate polynomial f in a linear fashion:

$$f = \sum_{\alpha \in \mathbb{M}_D} c_\alpha x^\alpha. \tag{2.79}$$

If we replace $\mathbb{M}_D$ with $\mathbb{M}_{D,\,L}$ in the above equation, then f is a D-variate polynomial of degree L. The space $\mathbb{P}_{D,\,L}$ can be made into the vector space that uses $\mathbb{M}_{D,\,L}$ as a basis, so that $f \in \mathbb{P}_{D,\,L}$ can be represented as a vector with components

$$(f)_{\mathbb{M}_{D,\,L},\,\alpha} = c_\alpha,\, \forall \alpha \in \alpha \in [0:L]^D \subset \mathbb{Z}_+^D,\, |\alpha| = \sum_{d=1}^{D} \alpha_d \leq L.$$

When our meaning is clear from context, we use the same symbol $\mathbb{P}_{D,\,L}$ to denote the vector space associated with the polynomial space $\mathbb{P}_{D,\,L}$. In this thesis, if f is a polynomial, we also write f to mean the vector that is isomorphic to that polynomial. Since there are $|\mathbb{M}_{D,\,L}|$ components, the dimension of $\mathbb{P}_{D,\,L}$ is $|\mathbb{M}_{D,\,L}|$. Although we use the raised-index notation for vector components in this thesis when we discuss tangent space vectors under a differentiable manifold setting, this notation is not used when we treat polynomials as vectors, because of possible confusion from the non-linear object x^α in Equation 2.79.

2.13.2 Moment Matrices

Our definition of a quasi-Hankel matrix later requires us to first define the following type of matrices:

Definition 2.111. Moment matrices (based on Section 1.1 from [64]). Let H be a square and real-valued matrix. It is a *moment matrix* if:

1. If the entries of H can be indexed by a subset $\mathcal{C} \subseteq \mathbb{M}_D$.
2. If $H_{a,\,b} = H_{a',\,b'}$ for all $a,\, b,\, a',\, b' \in \mathcal{C}$ that satisfies $ab = a'b'$. The multiplication here is the multiplication between monomials.

This is equivalent to the conditions:

1. If the entries of H can be indexed by the subset $\mathbb{B} \subseteq \mathbb{Z}_+^D$ that is the space of exponents of each monomial in $\mathcal{C}$.
2. If $H_{\alpha,\,\beta} = H_{\alpha',\,\beta'}$ for all $\alpha,\, \beta,\, \alpha',\, \beta' \in \mathbb{B}$ that satisfies $\alpha + \beta = \alpha' + \beta'$. The addition here is per-element addition between tuples.

Consider the product between sets

$$\mathcal{C} \cdot \mathcal{C} = \{\alpha\beta \mid \alpha\beta \in \mathcal{C}\},$$

and the *Minkowski addition* of sets

$$2\mathbb{B} := \mathbb{B} + \mathbb{B} = \{\alpha + \beta \mid \alpha \in \mathbb{B},\, \beta \in \mathbb{B}\}.$$

It follows that the unique entries of H are from the set $\{h_c\}_{c \in \mathcal{C} \cdot \mathcal{C}}$, or equivalently $\{h_\alpha\}_{\alpha \in 2\mathbb{B}}$ because we can choose to index h by monomials c or monomial exponents α. If $\mathcal{C} = \mathbb{M}_{D,\,L}$, then the unique entries of H are from the set $\{h_c\}_{c \in \mathbb{M}_{D,\,2L}}$. If $\mathcal{C} = \mathbb{M}_{D,\,L,\,\text{hom}}$, then the unique entries of H are from the set $\{h_\alpha\}_{\alpha \in \mathbb{A}(D,\,2L)}$.

2.13.3 Monomials Connected to 1

Definition 2.112. The *closure* of a set of monomials (taken from Section 1.3 of [64]). Let x be a D-tuple of real-valued variables. The closure of $\mathcal{C} \subseteq \mathbb{M}_D$ is defined as

$$\begin{aligned}\mathcal{C}^+ &:= \mathcal{C} \cup \bigcup_{d=1}^{D} x_i \mathcal{C} \\ &= \{a,\, x_1 a,\, x_2 a, \cdots,\, x_D a \mid a \in \mathcal{C}\}.\end{aligned}$$

Definition 2.113. Monomials connected to 1 (taken from Section 1.3 of [64]). A set of monomials $\mathcal{C} \subseteq \mathbb{M}_D$ is said to be *connected to* 1 if $1 \in \mathcal{C}$ and every monomial $m \in \mathcal{C} \setminus \{1\}$ can be written as

$$m = \prod_{d=1}^{k} x_{i_d},$$

for some $k \leq D$, and the index set $\{i_d\}_{d=1}^{k}$ must satisfy

$$\prod_{j=1}^{d} x_{i_j} \in \mathcal{C},\, \forall d \in [k].$$

I derived the following property for the proof of Proposition 4.1:

Proposition 2.114. *Let x be a D-tuple of real-valued variables, and $\mathcal{C}$ be a set of monomials that contains*

$$\mathcal{X} = \{1,\, x_1,\, x_2, \cdots,\, x_D\}.$$

If $\mathcal{A} \subset \mathcal{C}$, then $\mathcal{A}^+ \subset \mathcal{C} \cdot \mathcal{C}$.

Proof. The multi-index notation is used here.

$$\begin{aligned}\mathcal{A}^+ &= \{x^\eta,\, x_1 x^\eta,\, x_2 x^\eta, \cdots,\, x_D x^\eta \mid x^\eta \in \mathcal{A} \subset \mathcal{C}\} \\ &\subset \{x^\alpha,\, x_1 x^\alpha,\, x_2 x^\alpha, \cdots,\, x_D x^\alpha \mid x^\alpha \in \mathcal{C}\} \\ &= \left\{x^{\alpha+e} \mid x^\alpha \in \mathcal{C},\, e \in \{0,\,1\}^D,\, |e| \in \{0,\,1\}\right\}, \\ &\subset \left\{x^{\alpha+\beta} \,|\, x^\alpha,\, x^\beta \in \mathcal{C}\right\} \\ &= \mathcal{C} \cdot \mathcal{C}.\end{aligned}$$

Note that the multi-index e is such that $x^e \in \mathcal{X}$. □

2.13.4 Quasi-Vandermonde and Quasi-Hankel Matrices

The material presented here is my modification of the presentation from the references [39, 29]. Quasi-Hankel matrices are used in [29] to specify the uniqueness conditions of the CP symmetric tensor decomposition problem. Quasi-Hankel matrices and quasi-Vandermonde matrices are both used to describe the work done in Chapter 4.

Definition 2.115. Quasi-Hankel Matrices. Let L, D, and p be finite positive integers, x a D-tuple of real-valued variables, $\mathcal{C}$ a set of monomials, $\mathbb{B} \subset \mathbb{Z}_+^D$ a set of multi-indices such that $x^\alpha \in \mathcal{C}$ for any $\alpha \in \mathbb{B}$, and $M = |\mathcal{C}|$. A *quasi-Hankel matrix* is a moment matrix that has unique entries from the set $\{h_\eta\}_{\eta \in 2\mathbb{B}}$, where if given a $\eta = \alpha + \beta$ (point-wise addition of multi-indices) for some $\alpha,\ \beta \in \mathbb{B} \subset \mathbb{N}^D$ implies that we have $x^\alpha,\ x^\beta \in \mathcal{C}$, then

$$h_\eta = \sum_{n=1}^{p} c_n \big(z^{(n)}\big)^\alpha \big(z^{(n)}\big)^\beta,$$

for some $c \in \mathbb{R}_{\text{s}}^p$ and $z^{(n)} \in \mathbb{R}_{\text{s}}^D$. Given a monomial ordering $\succ$ for $\mathbb{B}$, we can concretely define a quasi-Hankel matrix in an entry-wise fashion: $\forall i,\ j \in [M], \forall \alpha^{(i)},\ \alpha^{(j)} \in \mathbb{A}(D,\ L)$,

$$\big(\mathtt{H}(\mathbb{B}_\succ,\ \mathfrak{z},\ c)\big)_{i,\ j} := \sum_{n=1}^{p} c_n \big(z^{(n)}\big)^{\alpha^{(i)}} \big(z^{(n)}\big)^{\alpha^{(j)}}, \tag{2.80}$$

where $\alpha^{(i)}$ is the ith element in $\mathbb{B}_\succ$.

We write $\mathtt{H}$ to mean $\mathtt{H}(\mathbb{B}_\succ,\ \mathfrak{z},\ c)$ when $\mathbb{B}_\succ$, $\mathfrak{z}$, and c are clear from context, or unimportant to the discussion at hand. The definition of quasi-Hankel matrices in [39] is more general than ours. The motivation of our definition here becomes clear in Sections 2.14 and 4.1.3.1.

The following type of structured matrices is used to factorize our definition of a quasi-Hankel matrix:

Definition 2.116. Quasi-Vandermonde Matrices. Let D, and p be finite positive integers, x a D-tuple of real-valued variables, $\mathcal{C}$ a set of monomials, $\mathbb{B} \subset \mathbb{Z}_+^D$ a set of multi-indices such that $x^\alpha \in \mathcal{C}$ for any $\alpha \in \mathbb{B}$, $\succ$ an ordering for $\mathbb{B}$, and $M = |\mathcal{C}|$. A *quasi-Vandermonde matrix* is an $M \times p$ matrix that is defined entry-wise as:

$$\big(\mathtt{V}(\mathbb{B}_\succ,\ \mathfrak{z})\big)_{i,\ j} := \big(z^{(j)}\big)^{\alpha^{(i)}} = \prod_{d=1}^{D} \big(z^{(j)}\big)_d^{\left(\alpha^{(i)}\right)_d}, \tag{2.81}$$

for some $c \in \mathbb{R}_{\text{s}}^p$ and $z^{(n)} \in \mathbb{R}_{\text{s}}^D$, $\forall n \in [p]$. The multi-index $\alpha^{(i)}$ is the ith element in $\mathbb{B}_\succ$.

We write $\mathtt{V}$ to mean $\mathtt{V}(\mathbb{B}_\succ,\ \mathfrak{z},\ c)$ when $\mathcal{C}_\succ$ and $\mathfrak{z}$ are clear from context, or unimportant to the discussion at hand. There are more general definitions for quasi-Vandermonde matrices in the literature, but the one presented here is sufficient for this thesis. Each column of $\mathtt{V}(\mathbb{B}_\succ,\ \mathfrak{z},\ c)$ corresponds to the coefficients of a D-variate polynomial as described by the monomials in $\mathcal{C}$. This means when $M \geq p$, $\text{Vand}(\mathfrak{z}) \in \mathbb{R}_{*,\text{s}}^{M \times p}$ is a full column rank matrix, and it is an element of the manifold described in Example 2.95.

Our definition of quasi-Hankel matrices and quasi-Vandermonde matrices implies that the following decomposition is always possible:

$$\mathtt{H}(\mathbb{B}_\succ,\ \mathfrak{z},\ c) = \mathtt{V}(\mathbb{B}_\succ,\ \mathfrak{z})\text{diagm}(c)\mathtt{V}(\mathbb{B}_\succ,\ \mathfrak{z})^T, \tag{2.82}$$

where $\text{diagm}(c)$ is the $p \times p$ diagonal matrix that have c_i assigned to the $(i,\ i)$th entry. Each matrix in Equation 2.82 has rank p when 1) each entry of $c \in \mathbb{R}_{\text{s}}^D$ is non-zero, 2) $\mathfrak{z}$ is a set of D-tuples that are not multiples of each other, and 3) $|\mathbb{B}| \geq p$.

2.13.5 Homogeneous Polynomials

Elements of $\mathbb{M}_{D,\,L,\,\text{hom}}$ can be used to represent a D-variate *homogeneous polynomial of degree* L, which we denote by f, that has the form

$$f = \sum_{\alpha \in \mathbb{A}(D,\,L)} \binom{L}{\alpha} c_\alpha x^\alpha, \tag{2.83}$$

which implies that each $x^\alpha \in \mathbb{M}_{D,\,L,\,\text{hom}}$. In this thesis, we denote by $\mathbb{H}_{D,\,L}$ the space of D-variate homogeneous polynomials of degree L. The multi-index notation and multinomial coefficient notation are both used here. The vector space that corresponds to $\mathbb{H}_{D,\,L}$ is an abstract vector space of homogeneous polynomials. If the set $\mathbb{H}_{D,\,L}$ is used as a basis, then a homogeneous polynomial $f \in \mathbb{H}_{D,\,L}$ can be represented as a vector with components

$$(f)_{\mathbb{H}_{D,\,L},\,\alpha} = \binom{L}{\alpha} c_\alpha,\ \forall \alpha \in \mathbb{A}(D,\,L).$$

Since there are $|\mathbb{A}(D,\,L)|$ components, the dimension of $\mathbb{H}_{D,\,L}$ is $|\mathbb{A}(D,\,L)|$.

Dehomogenization of a Homogeneous Polynomial

Consider D-variate polynomials of the following form:

$$f(x) = \sum_{j=1}^{p} c_j x^{a_j},\ c_j \in \mathbb{R} \setminus \{0\},\ a_j \in \mathbb{Z}_+,\ j \in [p],\ p \in \mathbb{Z}_+,$$

where x is a D-tuple of real-valued variables. The degree of polynomials of this form is $L = \max_{j \in [p]}\{a_j\}$. This is called a *p-sparse polynomial* in the literature, because the dimension of $\mathbb{P}_{D,\,L}$ as a vector space is large if D and L are large. Let $x_{D+1} \in \mathbb{R}$ be another real-valued variable. The procedure where each term of f (i.e., $c_j x^{a_j}$) is multiplied by $x_{D+1}^{b_j}$ is called a *homogenization of* f if $b_j \in \mathbb{Z}_+$ is such that the resultant polynomial produced by this procedure is a homogeneous polynomial of degree L. This is a reparameterization of $f(x_{1:D})$ with the variables $x_{1:D+1}$. The monomial exponents in the resultant homogeneous polynomial correspond to multi-indices in $\mathbb{A}(D+1,\,L)$.

Now, consider D-variate, Lth-degree, homogeneous polynomials (i.e., an element of $\mathbb{H}_{D,\,L}$) of the following form:

$$g(x) = \sum_{j=1}^{p} c_j \prod_{l=1}^{L} x_{b_l},\ c_j \in \mathbb{R} \setminus \{0\},\ j \in [p],\ p \in \mathbb{Z}_+,\ b_l \in [D],\ l \in [L],$$

where x is a D-tuple of real-valued variables. The procedure where we pick one of the D variables to be fixed to 1, then divide each term of g by that variable is called the *dehomogenization of* g, and it produces a non-homogeneous $(D-1)$-variate polynomial. This is a reparameterization of $g(x_{1:D})$ with the variables $\{x_d\}_{d \in [D],\,d \neq k}$, where x_k is the variable chosen to be fixed to 1. Without loss of generality, most works in the literature usually work with a permuted version of x, so that x_1 or x_D is chosen to be fixed to 1. The permutation of the other variables do not matter.

2.14 Symmetric Tensors

In this section, we review rectangular tensors, and their relationship with homogeneous polynomials and quasi-Hankel matrices. The material reviewed here is required to understand Chapter 4, and is discussed in greater detail in [29, 40, 39].

Recall from Section 2.3 that the $(p,\, q)$-tensors are multilinear maps that take inputs from a vector space V and its dual V^*. We can generalize $(p,\, q)$-tensors to multilinear maps that take inputs from any finite-dimensional vector spaces with field $\mathbb{R}$. The components of these multilinear maps are stored in a multi-dimensional array that is not necessarily a hypercube.

Definition 2.117. Rectangular tensors and cubical tensors. Let L be a finite positive integer. For each $l \in [L]$, let V_l be the D_l-dimensional vector space with field $\mathbb{R}$. A multilinear map T of the form

$$T : V_1 \times V_2 \times \cdots \times V_L \xrightarrow{\sim} \mathbb{R}$$

is called a *cubical tensor* if all input vector spaces V_l have the same dimension, otherwise is called a *rectangular tensor*. L is the *order* of such a tensor.

Definition 2.117 characterizes multilinear maps based on their input dimensions instead of the relationship between a vector space and its dual. All cubical tensors are rectangular tensors. All $(p,\, q)$-tensors are cubical tensors, but the converse is not true. One can define tensor space, tensor product, and tensor basis for rectangular tensors in a manner similar to our review in Section 2.3, so we do not go over this explicitly here.

The differentiable manifolds literature often uses $(p,\, q)$-tensors, since the concept of a vector and its dual is important there. Dimensionality reduction schemes for data sets tend to use generic rectangular tensors, since the flexibility on the input space dimensions is important in this case. There are a total of $\prod_{l=1}^{L} D_l$ components for a rectangular tensor. For an application to be computationally feasible that uses rectangular tensors, L can be large as long as not too many D_l's are large. There are D^L components for a cubical tensor, where D is the dimension of all the input vector spaces. For an application to be computationally feasible that uses cubical tensors, both L and D should be small.

We focus on the following type of cubical tensor in Chapter 4:

Definition 2.118. Symmetric tensor. A *symmetric tensor* T is a $(p,\, q)$-tensor over the vector space V from Definition 2.23 where its components with respect to any basis $\mathcal{B}$ of $\mathcal{T}_q^p V$ must satisfy

$$(T)_{\mathcal{E},\, a} = (T)_{\mathcal{E},\, \varsigma(a)},\ \forall a \in [\dim V]^{p+q}$$

for any permutations $\varsigma(a)$ of the multi-index a; see Section 2.1.2 for details on ς. $L := p + q$ is the *order* of a symmetric tensor.

A symmetric tensor can be used as a multilinear map that represents the iterated partial derivative operator on a function that satisfy Clairaut's theorem. When this theorem applies, the symmetry between its partial derivatives yields a symmetric tensor. Without a loss of generality, when the discussion at hand does not utilize any vector-covector relationship, we treat a symmetric tensor as a $(L,\, 0)$-tensor. We use the shorthand notation

$$\mathcal{S}(D,\, L) \subset \mathcal{T}_0^L \mathbb{R}^D$$

to mean the space of real-valued symmetric tensors. Unless specified otherwise, we describe the components of symmetric tensors with respect to the elementary basis of its tensor space.

Example 2.119. A rank-one tensor from Example 2.28 is an element of $\mathcal{S}(D,\, L)$.

2.14.1 Symmetric Tensors as Homogeneous Polynomials

An Lth-order, D-dimensional symmetric tensor is a cubical tensor that has D^L entries, but only $\binom{L+D-1}{D-1}$ distinct entries in the general case. It follows from the multinomial theorem in Equations 2.1 and 2.2 that

one can index such tensors using multi-indices from $[D]^L$ or $\mathbb{A}(D, L) \subset [0 : L]^D$. Let x be a D-tuple of real-valued variables. We can represent a monomial of degree L (i.e., an element of $\mathbb{M}_{D,L,\text{hom}}$) using the multi-index notation

$$x^a := \prod_{d=1}^{D} x_d^{a_d},\ a \in \mathbb{A}(D, L) \subset [0 : L]^D. \tag{2.84}$$

Another way is by

$$x_{b_1} x_{b_2} \cdots x_{b_L} = \prod_{l=1}^{L} x_{b_l}, \tag{2.85}$$

for some $b \in [D]^L$.

There is an isomorphism between symmetric tensors and homogeneous polynomials [28, 29, 31], but its proof is out of the scope of this thesis. If a homogeneous polynomial $f \in \mathbb{H}_{D,L}$ is isomorphic to a symmetric tensor T, then we say *T is the associated tensor to f*, or *f is the associated homogeneous polynomial to T*.

Example 2.120. The associated homogeneous polynomial of a symmetric tensor. Let A be a $2 \times 2 \times 2$ array with entries given by

$$A_{:,:,1} = \begin{bmatrix} -2 & 0 \\ 0 & 1 \end{bmatrix},$$
$$A_{:,:,2} = \begin{bmatrix} 0 & 1 \\ 1 & -2.3 \end{bmatrix}.$$

Consider the tensor with the values

$$B = \sum_{b \in [2]^3} A_b e_{b_1} \otimes e_{b_2} \otimes e_{b_3}.$$

as its components with respect to the elementary basis

$$\mathcal{E} = \{e_{b_1} \otimes e_{b_2} \otimes e_{b_3}\}_{b \in [2]^3}.$$

Let $x := \{x_1, x_2\}$ be a set of real-valued polynomial variables. The associated homogeneous polynomial to B is

$$\begin{aligned} &- 2x_1^3 + x_2 x_2 x_1 + x_2 x_1 x_2 + x_1 x_2 x_2 - 2.3 x_2^3 \\ &= -2x_1^3 + 3x_1 x_2^2 - 2.3 x_2^3. \end{aligned}$$

2.14.2 The CP Decomposition: Non-Uniqueness Issues

Since $[D]^L$ is the space of multi-indices to index monomials of degree L, it follows that a homogeneous polynomials from the space

$$\mathbb{J}(D, L) := \left\{ \left(\sum_{d=1}^{D} c_d x_d \right)^L \mid c_d \in \mathbb{R},\ d \in [D] \right\} \subset \mathbb{H}(D, L) \tag{2.86}$$

maps to a rank-one tensor. Recall from Example 2.28 and Equation 2.18 that $\mathcal{J}(D, L)$ is the space of rank-one tensors of dimension D and order L. Given a $\left(\sum_{d=1}^{D} c_d x_d\right)^L \in \mathbb{J}(D, L)$, define a change of variables

$y_d := c_d x_d$, $\forall d \in [D]$, and apply the multinomial theorem from Equations 2.1 and 2.2:

$$\begin{aligned}\left(\sum_{d=1}^{D} c_d x_d\right)^L &= \left(\sum_{i=1}^{D} y_i\right)^L \\ &= \sum_{a \in \mathbb{A}(D,\, L)} \binom{L}{a} y^a \\ &= \sum_{b \in [D]^L} \prod_{l=1}^{L} y_{b_l} \\ &= \sum_{b \in [D]^L} \prod_{l=1}^{L} c_{b_l} x_{b_l}. \end{aligned} \tag{2.87}$$

If we treat $\prod_{l=1}^{L} c_{b_l}$ as the component for the bth basis of $\mathcal{S}(D,\, L)$, then the last line together with Example 2.28 implies that the rank-one tensor $v^{\otimes L} \in \mathcal{J}(D,\, L)$ corresponds to $\left(\sum_{d=1}^{D} c_d x_d\right)^L \in \mathbb{J}(D,\, L)$, where $v \in \mathcal{T}_0^1 \mathbb{R}^D$ is such that $c_d = (v)_{\mathcal{B},\, d}$ is the dth component of v with respect to some basis $\mathcal{B}$ of $\mathbb{R}^D$. This relationship extends further; one can decompose any symmetric tensor into a sum of rank-one tensors where each tensor is from $\mathcal{J}(D,\, L)$ [30]. However, this decomposition is non-unique for general combinations of D and L. It follows that one can non-injectively map an element $T \in \mathcal{S}(D,\, L)$ to a sum of elements in $\mathcal{J}(D,\, L)$. Therefore, for a general D and L, a homogeneous polynomial in $\mathbb{H}_{D,\, L}$ can have multiple rank-one decompositions where each decomposition corresponds to a different element in $\mathbb{J}(D,\, L)$. We continue this discussion in Section 4.1.

2.14.3 Symmetric Tensors as Dehomogenized Polynomials

Given a tensor $T \in \mathcal{S}(D,\, L)$, the *canonical polyadic tensor* (CP) decomposition problem for symmetric tensors is to find a decomposition of the following form with the smallest p:

$$T = \sum_{j=1}^{p} \lambda_j \left(v^{(j)}\right)^{\otimes L},$$

for some $\lambda_j \in \mathbb{R}$, $v^{(j)} \in \mathbb{R}^D$, $\forall j \in [p]$. The reference [30] shows that such a decomposition always exists, and its authors call p the *symmetric tensor rank of* T. We have been referring to this sense of tensor rank as the *CP rank* since Chapter 1. Let $g(x)$ be the homogeneous polynomial associated with T, where x is a D-tuple of real-valued variables for g. The *algebraic approach* [29, 28] to the CP symmetric tensor decomposition problem computes the following form of g:

$$g(x) = \sum_{j=1}^{p} \lambda_j \left(\sum_{d=1}^{D} u_d^{(j)} x_d\right)^L \in \mathbb{H}_{D,\, L}, \tag{2.88}$$

for some $u_d \in \mathbb{R}$, $\forall d \in [D]$. This representation is called a *sum of powers of linear forms* [29]. Now, let us dehomogenize g by setting x_k to 1, and denote by $\boldsymbol{g}$ the result. We choose $k \in [D]$ such that

$$\left(u^{(j)}\right)_k \neq 0,\ \forall j \in [p]. \tag{2.89}$$

We see that $\exists k \in [D]$ such that Equation 2.89 holds, otherwise g cannot be a homogeneous polynomial of degree L. For this reason, a dehomogenized version of g always exists. Without loss of generality, let us work

with a permuted version of the previously defined set of polynomial variables $\{x_d\}$, such that Equation 2.89 holds when k is 1. The permutation of the other variables do not matter. Denote by $\boldsymbol{g}$ the dehomogenized version of g. The sum of powers of linear forms for $\boldsymbol{g}$ that corresponds to Equation 2.88 is then

$$\begin{aligned} g(x) &= \sum_{j=1}^{p} \lambda_j \left(u_1^{(j)} x_1 + \sum_{d=2}^{D} u_d^{(j)} x_d \right)^L \\ &= \sum_{j=1}^{p} \lambda_j \left(u_1^{(j)} x_1 \right)^L \left(1 + \sum_{d=2}^{D} \frac{u_d^{(j)}}{u_1^{(j)} x_1} x_d \right)^L \\ \implies \boldsymbol{g}(x_{2:D}) &= g(x)|_{x_1=1} \\ &:= \sum_{j=1}^{p} c_j \left(1 + \sum_{d=2}^{D} z_d^{(j)} x_d \right)^L . \end{aligned} \tag{2.90}$$

The new polynomial coefficients $\left\{ z_d^{(j)} \right\}$ and the new rank-one tensor scale factors $\{c_j\}$ are defined as

$$z_d^{(j)} := \left. \frac{u_d^{(j)}}{u_1^{(j)} x_1} \right|_{x_1=1} = \frac{u_d^{(j)}}{u_1^{(j)}}, \forall d \in [D], \forall j \in [p], \tag{2.91}$$

and

$$c_j := \lambda_j \left(u_1^{(j)} x_1 \right)^L \Big|_{x_1=1} = \lambda_j \left(u_1^{(j)} \right)^L, \forall d \in [D]. \tag{2.92}$$

It follows that $z_1^{(j)} = 1$ for all $j \in [p]$.

2.14.4 A Monomial Ordering for $\mathbb{H}_{D,\,L}$

In Section 2.13.2, we discussed how the unique entries of a moment matrix are indexed by either monomials, or multi-indices that specify the exponents of monomials. An ordering is required for practical implementation of monomial-related schemes, and we present a particular ordering of monomial exponents in this subsection. We first review another ordering:

Definition 2.121. Graded reverse lexicographic monomial exponent ordering $\succ_{\text{R}}$. Let $\alpha, \beta \in \{0 : L\}^D$ be two multi-indices in $\mathbb{A}(D,\, L)$. Let the comparison relation $\succ_{\text{R}}$ be such that:

- If $|\alpha| > |\beta|$, then $\alpha \succ_{\text{R}} \beta$.
- If $|\alpha| = |\beta|$ and $\alpha_i - \beta_i < 0$ where i is the largest element in $[D]$ such that $\alpha_i - \beta_i \neq 0$, then $\alpha \succ_{\text{R}} \beta$.

Example 2.122. Given $D = 3$, $L = 4$, we have

$$\begin{aligned} \mathbb{A}(D,\, l)_{\succ_{\text{R}}} = &\{(4,\, 0,\, 0),\, (3,\, 1,\, 0),\, (2,\, 2,\, 0),\, (1,\, 3,\, 0),\, (0,\, 4,\, 0),\, (3,\, 0,\, 1),\, (2,\, 1,\, 1), \\ &(1,\, 2,\, 1),\, (0,\, 3,\, 1),\, (2,\, 0,\, 2),\, (1,\, 1,\, 2),\, (0,\, 2,\, 2),\, (1,\, 0,\, 3),\, (0,\, 1,\, 3),\, (0,\, 0,\, 4)\}. \end{aligned}$$

Present-day computing architectures are more efficient when data is stored in an array as opposed to hash-tables or tree-structures, so we store the distinct components of a symmetric tensor in a 1-dimensional array. The multi-index set $\mathbb{A}(D,\, L)$ can be used to index these distinct components (see Section 2.13.5). I devised the following ordering for $\mathbb{A}(D,\, L)$:

Definition 2.123. The monomial exponent ordering $\succ_{\text{V}}$ for $\mathbb{A}(D,\, L)$. Let $\alpha,\, \beta \in \{0 : L\}^D$ be two multi-indices in $\mathbb{A}(D,\, L)$. Let the comparison relation $\succ_{\text{V}}$ be such that:

- If $\alpha_1 > \beta_1$, then $\alpha \succ_{\text{V}} \beta$.
- If $\sum_{d=2}^{D} \alpha_d = \sum_{d=2}^{D} \alpha_d$ and

$$(\alpha_2,\, \alpha_3, \cdots,\, \alpha_D) \succ_{\text{R}} (\beta_2,\, \beta_3, \cdots,\, \beta_D)$$

  then $\alpha \prec_{\text{V}} \beta$.

Example 2.124. Given $D = 3$, $L = 4$, we have

$$\begin{aligned}\mathbb{A}(D,\, L)_{\succ_{\text{V}}} =\{&(4,\, 0,\, 0),\, (3,\, 1,\, 0),\, (3,\, 0,\, 1),\, (2,\, 2,\, 0),\, (2,\, 1,\, 1),\, (2,\, 0,\, 2),\, (1,\, 3,\, 0),\\ &(1,\, 2,\, 1),\, (1,\, 1,\, 2),\, (1,\, 0,\, 3),\, (0,\, 4,\, 0),\, (0,\, 3,\, 1),\, (0,\, 2,\, 2),\, (0,\, 1,\, 3),\, (0,\, 0,\, 4)\}.\end{aligned}$$

To the best of my knowledge, I have not come across this ordering in the literature. The ordering $\succ_{\text{V}}$ is used with dehomogenized polynomials in the implementation of this thesis work. In Appendix B, we introduce a multi-index set $\mathbb{A}_{\text{de}}(D,\, L)$ over the actual $D-1$ variables of a dehomogenized polynomial, where the first slot of the D-tuple of variables x is set to 1. This facilitates efficient numerical computation, and is used in all numerical implementations for the work done in Chapter 4. An interpretation of the monomial exponent ordering $\succ_{\text{V}}$ for $\mathbb{A}(D,\, L)$ using $\mathbb{A}_{\text{de}}(D,\, L)$ is also presented in Appendix B. The set $\mathbb{A}_{\text{de}}(D,\, L)$ is isomorphic to $\mathbb{A}(D,\, L)$, so either one can be used to present the work done on dehomogenized polynomials in this thesis.

2.14.5 Symmetric Tensors and Quasi-Hankel Matrices

Our goal in this subsection is to show that a column of a quasi-Hankel matrix corresponds to the unique components of a symmetric tensor. The material covered in this subsection is my interpretation of existing knowledge in the literature.

Let x be a D-tuple of real-valued variables, p a positive integer, and L chosen such that $\binom{L+D-1}{L} > p$. Recall from Equations 2.88 and 2.90 that we can represent the components of a symmetric tensor $T \in \mathcal{S}(D,\, L)$ as a homogeneous polynomial g that is parameterized as

$$g(x) = \sum_{j=1}^{p} \lambda_j \left(\sum_{d=1}^{D} u_d^{(j)} x_d \right)^L,$$

or its dehomogenized parameterization

$$\boldsymbol{g}(x_{2:D}) = \sum_{j=1}^{p} c_j \left(1 + \sum_{d=2}^{D} z_d^{(j)} x_d \right)^L,$$

for some $\lambda_j \in \mathbb{R}$, $\forall j \in [p]$, and some $u_d \in \mathbb{R}$, $\forall d \in [D]$. We also have

$$z_d^{(j)} := \frac{u_d^{(j)}}{u_1^{(j)}},\ \forall d \in [D],\ \forall j \in [p],$$

and

$$c_j := \lambda_j \left(u_1^{(j)} \right)^L,\ \forall d \in [D].$$

It follows that $z_1^{(j)} = 1$ for all $j \in [p]$. If we apply the multinomial theorem (Equation 2.1) to $\boldsymbol{g}$, we have

$$\sum_{j=1}^{p} c_j \left(1 + \sum_{d=2}^{D} z_d^{(j)} x_d\right)^L = \sum_{j=1}^{p} c_j \sum_{a \in \mathbb{A}(D,\, L)} \binom{L}{a} 1^{a_1} \prod_{d=2}^{D} \left(z_d^{(j)}\right)^{a_d} x_d^{a_d}$$

We see that

$$(T)_a := \sum_{j=1}^{p} c_j \binom{L}{a} \prod_{d=2}^{D} \left(z_d^{(j)}\right)^{a_d}$$

is the ath component of a symmetric tensor T with respect to some basis.

Let

$$\mathfrak{z} := \left\{z^{(j)} \in \mathbb{R}_{\mathrm{s}}^D \mid \left(z^{(j)}\right)_1 = 1\right\}_{j=1}^{p}$$
$$c := \{c_j \in \mathbb{R} \setminus \{0\}\}_{j=1}^{p}.$$

Equation 2.80 shows that the first column of the quasi-Hankel matrix $\mathtt{H}\left(\mathbb{A}(D,\, L)_{\succ_{\mathrm{V}}},\, \mathfrak{z},\, c\right)$ is given in an entry-wise manner by, $\forall k \in [|\mathbb{A}(D,\, L)|]$,

$$\begin{aligned}
\left(\mathtt{H}\left(\mathbb{A}(D,\, L)_{\succ_{\mathrm{V}}},\, \mathfrak{z},\, c\right)\right)_{1,\,k} &= \mathfrak{h}\left(\beta^{(1)} + \beta^{(k)},\, \mathfrak{z},\, c\right) \\
&= \sum_{j=1}^{p} c_j \left(z^{(j)}\right)^{\beta^{(1)}} \left(z^{(j)}\right)^{\beta^{(k)}} \\
&= \sum_{j=1}^{p} c_j \left(z^{(j)}\right)^{\beta^{(k)}} \\
&= \frac{T_{\beta^{(k)}}}{\binom{L}{\beta^{(k)}}},
\end{aligned} \tag{2.93}$$

where $\beta^{(k)}$ is the kth entry of $\mathbb{A}(D,\, L)_{\succ_{\mathrm{V}}}$. The second last line follows from the fact the first multi-index in the ordered set $\mathbb{A}(D,\, L)_{\succ_{\mathrm{V}}}$ is the D-tuple $(L,\, 0,\, 0,\, \cdots,\, 0)$; see Example 2.124. This shows that the first column of $\mathtt{H}\left(\mathbb{A}(D,\, L)_{\succ_{\mathrm{V}}},\, \mathfrak{z},\, c\right)$ corresponds to the distinct tensor components of T that are ordered by $\succ_{\mathrm{V}}$ and scaled by a factor of

$$\frac{T_a}{\binom{L}{a}},\ a \in \mathbb{A}(D,\, L)_{\prec_{\mathrm{V}}}.$$

A similar result holds for the first row of $\mathtt{H}\left(\mathbb{A}(D,\, L)_{\succ_{\mathrm{V}}},\, \mathfrak{z},\, c\right)$, since it is a symmetric matrix. Using this relationship, we can convert between D-dimensional, Lth-order, symmetric tensors and the first row of a quasi-Hankel matrix of the form $\mathtt{H}\left(\mathbb{A}(D,\, L)_{\succ_{\mathrm{V}}},\, \mathfrak{z},\, c\right)$. The following example uses less notation than the general formulae presented in this section because we fixed D, L, and p to 2, 4, and 2, respectively:

Example 2.125. The symbolic expression for a 2-dimensional, 4th-order rank-one decomposition with cardinality 2 and its quasi-Hankel matrix representation. Consider the non-collinear and non-zero tuples $a,\, b \in \mathbb{R}_{\mathrm{s}}^D$ where the first entry is non-zero, and non-zero real numbers λ_1 and λ_2. If a and b are the component-forms (with respect to some basis $\mathcal{B}$) for two vectors in $\mathbb{R}^D$, then we have a rank-one decomposition in homogeneous form if we make the substitution $u^{(1)} \leftarrow a$ and $u^{(2)} \leftarrow b$ in Equation 2.88. Its

dehomogeneous form is

$$\begin{aligned} c_1 &\leftarrow \lambda_1 a_1^4 \\ c_2 &\leftarrow \lambda_2 b_1^4 \\ \mathfrak{z}^{(1)} &\leftarrow \left(1, \frac{a_2}{a_1}\right) \\ \mathfrak{z}^{(2)} &\leftarrow \left(1, \frac{b_2}{b_1}\right). \end{aligned}$$

The components of the symmetric tensor J with respect to $\mathcal{B}^{\otimes L}$ of this rank-one decomposition are given by

$$\begin{aligned} J_{:,:,1,1} &= \begin{bmatrix} \lambda_1 a_1^4 + \lambda_2 b_1^4 & \lambda_1 a_1^3 a_2 + \lambda_2 b_1^3 b_2 \\ \lambda_1 a_1^3 a_2 + \lambda_2 b_1^3 b_2 & \lambda_1 a_1^2 a_2^2 + \lambda_2 b_1^2 b_2^2 \end{bmatrix} \\ J_{:,:,2,1} &= \begin{bmatrix} \lambda_1 a_1^3 a_2 + \lambda_2 b_1^3 b_2 & \lambda_1 a_1^2 a_2^2 + \lambda_2 b_1^2 b_2^2 \\ \lambda_1 a_1^2 a_2^2 + \lambda_2 b_1^2 b_2^2 & \lambda_1 a_1 a_2^3 + \lambda_2 b_1 b_2^3 \end{bmatrix} \\ J_{:,:,1,2} &= \begin{bmatrix} \lambda_1 a_1^3 a_2 + \lambda_2 b_1^3 b_2 & \lambda_1 a_1^2 a_2^2 + \lambda_2 b_1^2 b_2^2 \\ \lambda_1 a_1^2 a_2^2 + \lambda_2 b_1^2 b_2^2 & \lambda_1 a_1 a_2^3 + \lambda_2 b_1 b_2^3 \end{bmatrix} \\ J_{:,:,2,2} &= \begin{bmatrix} \lambda_1 a_1^2 a_2^2 + \lambda_2 b_1^2 b_2^2 & \lambda_1 a_1 a_2^3 + \lambda_2 b_1 b_2^3 \\ \lambda_1 a_1 a_2^3 + \lambda_2 b_1 b_2^3 & \lambda_1 a_2^4 + \lambda_2 b_2^4 \end{bmatrix}. \end{aligned}$$

The tensor product basis notation $\mathcal{B}^{\otimes L}$ is discussed in Section 2.3. There are 2^4 entries, but only $|\mathbb{A}_{\text{de}}(2,\ 4)| = 5$ entries are unique. Recall from our discussion near the end of Section 2.14.4 that the multi-index set $\mathbb{A}_{\text{de}}(D,\ L)$ is isomorphic to $\mathbb{A}(D,\ L)$. More details on $\mathbb{A}_{\text{de}}$ are discussed in Appendix B, and the formula for $|\mathbb{A}_{\text{de}}(D,\ L)|$ is given by Equation B.2. The unique entries are

$$\begin{aligned} h_{40} &:= \lambda_1 a_1^4 + \lambda_2 b_1^4 \\ h_{31} &:= \lambda_1 a_1^3 a_2 + \lambda_2 b_1^3 b_2 \\ h_{22} &:= \lambda_1 a_1^2 a_2^2 + \lambda_2 b_1^2 b_2^2 \\ h_{13} &:= \lambda_1 a_1 a_2^3 + \lambda_2 b_1 b_2^3 \\ h_{04} &:= \lambda_1 a_2^4 + \lambda_2 b_2^4. \end{aligned}$$

The quasi-Vandermonde matrix $\mathtt{V}\left(\mathbb{A}_{\text{de}}(D,\ L)_{\succ_{\mathtt{V}}},\ \mathfrak{z}\right)$ is given by

$$\begin{bmatrix} 1 & 1 \\ \frac{a_2}{a_1} & \frac{b_2}{b_1} \\ \frac{a_2^2}{a_1^2} & \frac{b_2^2}{b_1^2} \\ \frac{a_2^3}{a_1^3} & \frac{b_2^3}{b_1^3} \\ \frac{a_2^4}{a_1^4} & \frac{b_2^4}{b_1^4} \end{bmatrix},$$

and the quasi-Hankel matrix $\mathtt{H}\left(\mathbb{A}_{\text{de}}(D,\ L)_{\succ_{\mathtt{V}}},\ \mathfrak{z},\ c\right)$ is given by

$$\begin{bmatrix} h_{04} & h_{31} & h_{22} & h_{13} & h_{04} \\ h_{31} & h_{22} & h_{13} & h_{04} & \frac{\lambda_1 a_2^5}{a_1} + \frac{\lambda_2 b_2^5}{b_1} \\ h_{22} & h_{13} & h_{04} & \frac{\lambda_1 a_2^5}{a_1} + \frac{\lambda_2 b_2^5}{b_1} & \frac{\lambda_1 a_2^6}{a_1^2} + \frac{\lambda_2 b_2^6}{b_1^2} \\ h_{13} & h_{04} & \frac{\lambda_1 a_2^5}{a_1} + \frac{\lambda_2 b_2^5}{b_1} & \frac{\lambda_1 a_2^6}{a_1^2} + \frac{\lambda_2 b_2^6}{b_1^2} & \frac{\lambda_1 a_2^7}{a_1^3} + \frac{\lambda_2 b_2^7}{b_1^3} \\ h_{04} & \frac{\lambda_1 a_2^5}{a_1} + \frac{\lambda_2 b_2^5}{b_1} & \frac{\lambda_1 a_2^6}{a_1^2} + \frac{\lambda_2 b_2^6}{b_1^2} & \frac{\lambda_1 a_2^7}{a_1^3} + \frac{\lambda_2 b_2^7}{b_1^3} & \frac{\lambda_1 a_2^8}{a_1^4} + \frac{\lambda_2 b_2^8}{b_1^4} \end{bmatrix}.$$

It is clear from Example 2.125 that the entries from a symmetric tensor alone cannot produce every entry in the quasi-Hankel matrix that corresponds to any rank-one decomposition of the tensor, denoted there by $(\lambda_1, \lambda_2, a, b)$. Therefore, rank-one decompositions and symmetric tensors are treated as different objects in this thesis.

CHAPTER 3

Adaptive Kernels

The reproducing kernel Hilbert space (RKHS) regularization framework introduced in Section 2.2 is often used to parameterize a continuous function from discrete data, but its performance is limited when paired with a stationary kernel. Non-stationary kernels can vary their behaviour across the input domain, which means it is possible for them to adapt to a signal's local patterns. The construction of non-stationary kernels that perform better than stationary kernels for a given application is an open problem in the literature. My main contribution of this chapter is a method that constructs non-stationary kernels from ones that are both stationary and isotropic. It is motivated by the *warp Gaussian process* method [10, 11], which we review in Section 3.1. We discuss in Section 3.2 my interpretation of the RKHS regularization framework as a sparse recovery method [65, 66] from the signal processing literature. This comparison gives us hints as to how kernels should be chosen to achieve good performance.

We present the proposed kernel construction method through Theorem 3.1 in Section 3.3. Given some chosen prior assumptions on the idealized signal f, different strategies can be used with this theorem to implement these prior assumptions as non-stationary kernels. In Section 3.4, we demonstrate such strategies when f is a grayscale image. Our only prior assumption chosen for f is that edges in f have smooth contours. We chose this particular prior assumption on f because it is easy to see how our different strategies implement this assumption, and their implications on performance. For reference purposes, we provide the grayscale image up-conversion results from a separable cubic-spline interpolator in our demonstrations as well. The goal of our demonstrations is to give examples of how one could use Theorem 3.1 to construct non-stationary kernels that implement a particular assumption on the desired signal. The techniques developed in our demonstrations are not meant to compete against the state-of-the-art image up-conversion schemes. In an practical application, one may fare better by using a combination of the state-of-the-art grayscale image up-conversion schemes as pre-processing or post-processing steps, but that is not the goal of our demonstrations.

Our visual demonstrations use the sample-without-prefiltering observation model as specified in Section 2.1.6. Our quantitative demonstrations use this and the sample-with-prefiltering observation models. The additive white Gaussian noise model from Section 2.1.6 is not used in this thesis because:

1. It evaluates the RKHS regularization framework's ability to handle this kind of white noise corruption, instead of the goal we discussed.

2. The exemplar strategies we used to implement our chosen prior assumption on f do not assume this noise model.

3. I believe the question of robustness to realistic noise models for any extensions of the RKHS regularization framework is an application-specific and open problem. Therefore, one cannot perform a proper investigation on noise models without a specific application in mind.

We use the concepts laid out in the RKHS regularization framework from Section 2.1.6 as our signal model. Given a positive integer D, we denote the continuous-domain signal of interest as $f \in L_2(\mathbb{R}^D)$, the continuous-domain degraded signal as $f_{\text{ob}} \in L_2(\mathbb{R}^D)$, the set of observed inputs as $\mathcal{X} \subset \mathbb{R}^D$, the set of query inputs (i.e., the up-conversion locations) as $\mathcal{X}_q \subset \mathbb{R}^D$. When all elements in $\mathcal{X}$ form a finite D-dimensional grid, we say the observed signal is *grid-sampled*. We write $f_{\text{ob}}|_{\mathcal{X}}$ to mean the observed signal. We write $\boldsymbol{f}_{\text{ob},\mathcal{X}}$ to mean the observed signal as a vector in $\mathbb{R}^{|\mathcal{X}|}$, or $\mathbb{R}^{N_1 \times \cdots \times N_D}$ if it is grid-sampled, where $\forall i \in [D]$, N_i is the number of elements along the ith dimension of this grid. The mapping from $f_{\text{ob}}|_{\mathcal{X}}$ to the components of $\boldsymbol{f}_{\text{ob},\mathcal{X}}$ with respect to the elementary basis is under some fixed arbitrary ordering $\mathcal{I}$, which is unimportant to most of our discussions unless we mention it. The particular choice of $\mathcal{I}$ used in our demonstrations is the inverse column-major ordering map. The continuous-domain query signal is denoted by $\tilde{f}$, but we mostly deal with the sampled query function $\tilde{f}\big|_{\mathcal{X}_q}$.

When the observed signal is grid-sampled, we write $\left(\boldsymbol{f}_{\text{ob},\mathcal{X}}\right)_{\mathcal{E}}$ to mean the D-dimensional array of the components of $\boldsymbol{f}_{\text{ob},\mathcal{X}}$ with respect to the elementary basis of $\mathbb{R}^{N_1 \times \cdots \times N_D}$. Both $\left(\boldsymbol{f}_{\text{ob},\mathcal{X}}\right)_{\mathcal{E}}$ and $\boldsymbol{f}_{\text{ob},\mathcal{X}}$ are used to reference the observed signal. $\left(\boldsymbol{f}_{\text{ob},\mathcal{X}}\right)_{\mathcal{E}}$ is often used in the algorithms presented in this chapter, and $\boldsymbol{f}_{\text{ob},\mathcal{X}}$ is used when the array interpretation of the grid-sampled signal is not central to the discussion at hand. We define $\tilde{\boldsymbol{f}}_{\mathcal{X}_q}$ and $\left(\tilde{\boldsymbol{f}}_{\mathcal{X}_q}\right)_{\mathcal{E}}$ in a similar fashion with respect to $\mathbb{R}^{|\mathcal{X}_q|}$, or $\mathbb{R}^{M_1 \times \cdots \times M_D}$ when $\mathcal{X}_q$ is on a grid. Here, $\forall i \in [D]$, M_i is the number of elements along the ith dimension of this grid.

3.1 Related Works

Past works on constructing non-stationary kernels tend to use theoretical results from stationary random processes to ensure the result is a symmetric positive-definite kernel, or pose the kernel as the solution of an optimization problem. Examples of the first approach are [8, 9], where non-stationary kernels are posed as linear systems that operate on white Gaussian processes. It is unintuitive to use this type of kernel construction method to implement prior assumptions on the signal of interest into the kernel design, since it may not make sense to interpret the application at hand as the modification of a white Gaussian process. This obscures the relationship between the prior knowledge one wishes to encode and the hyperparameters of the resultant kernel. Optimization-based non-stationary kernels usually use some form of likelihood function as the cost function; an example is [12]. These approaches typically require solving a non-convex optimization problem.

The warp Gaussian process method [10, 11] treats the observations in the RKHS regularization problem as random variables, and it requires multiple realizations of these random variables. Let k be a stationary kernel, f be the signal that generated the observations, and x the input variable to f. A non-linear function w is used, such that $k \circ w$ is used as the kernel instead of k. However, the forms of w that give rise to positive-definite kernels were not discussed in these works, and a Markov Chain Monte Carlo (MCMC) algorithm was used to characterize the behaviour of $k \circ w$. MCMC algorithms are known to have slow convergence rates in practical settings. Another issue is that multiple instances of the observable random variables are required, so these works are not applicable to problems where only one instance of the observable variable is available. An example of such a problem is the grayscale image up-conversion task where only one lower-resolution image of the scene is provided.

In [12], a hierarchical network of Gaussian processes that each have a stationary kernel was used to model a signal that exhibits non-homogeneous patterns. The network can be collectively viewed as a single non-stationary kernel. Their work did not require multiple realizations of the observation variables, because

all quantities were estimated using a Bayesian inference setup. However, like most hierarchical network-learning schemes, there are significant non-identifiability issues that arise from a permutation of the nodes (e.g., see Section 16.5.5 of [5]). It is also unclear how one can incorporate application-specific prior information about the signal of interest. The non-stationary kernel construction method proposed in this chapter is designed for the scenario where large-scale observations are unavailable but prior knowledge does exist for the application at hand. The proposed method ensures that the result is a positive-definite kernel, and computationally-intensive algorithms are not required in the process.

The image processing work [67] is based on a ridge regression formulation, which is known to be closely related to kernel methods [49, 2]. Using the gradient of an image, they posed some image processing problems as ridge regression regularization problems. Their derivations are based on the bilateral filter from image processing and ridge regression theory, while the work done in this chapter is based on the theory of regularization over a RKHS and sparse recovery methods. The focus of this chapter is on specifying a construction method for non-stationary kernels, with applications taking a secondary priority, whereas their work is focused on some specific image processing problems. Their theoretical contributions consist of modifications to the ridge regression equations so that it is a function of the gradient of an image signal, while our theoretical contribution is in the form of a theorem that guarantees the constructed kernel is symmetric and positive-definite. The significance is that the representer theorem can be used to solve a regularization problem over the RKHS induced by such a kernel. Their main contributions are the documented applications for which they have shown their method to be empirically successful. In this chapter, we have a demonstration on real-life data for the proposed kernel construction method. More detailed investigations into applying the proposed methods to present-day relevant problems are intended for future works.

3.2 Sparse Representation of Continuous Functions

Recall from Section 2.1.6 that the underlying assumption of our observation model is that a continuous-domain function f is assumed to have generated the observed discrete-domain signal under some degrade-then-sample procedure. In this section, we discuss how the sparse recovery method [65, 66] and the RKHS regularization framework are related. We present a modified version of the sparse recovery method that does not use an orthonormal basis for an infinite-dimensional Hilbert space, so that we do not need to introduce additional background. We focus our attention on the case where no degradation takes place, which corresponds to the case when $f_{\text{ob}} = f$ in Section 2.1.6.

Let $b : \mathbb{R}^D \times \mathbb{R}^E \to \mathbb{R}$ be a function such that $b(\cdot, a) \in L_2(\mathbb{R}^D)$, $\forall a \in \mathbb{R}^D$, for some given positive integer D. Let $b_a := b(\cdot, a)$ be a short-hand notation for a function made from fixing one of the inputs of a two-input function to the object a. We want to build a basis for a finite-dimensional vector space by using evaluations of b_a for different a. Let $c : \mathbb{R}^D \to \mathbb{R}$ be a bounded map that satisfies

$$\sum_{i=1}^{P} c(a_i) b_{a_i}(x) = 0\,, \forall x \in \mathbb{R}^D \quad \Longrightarrow \quad c(a_i) = 0,\ \forall a_i \in \mathbb{R}^D,\ \forall i \in [P], \tag{3.1}$$

for every finite positive integer P. We call each $c(a_i) \in \mathbb{R}$ a *linear expansion coefficient with respect to* $\mathcal{B}$. Given a finite set of inputs $\mathcal{U} \subset \mathbb{R}^D$, an ordered version of it $\mathcal{U}_{[|\mathcal{U}|]} := \{u_i\}_{i=1}^{|\mathcal{U}|}$, and a $p \in [|\mathcal{U}|]$, Equation 3.1 implies that the set of $|\mathcal{U}|$-dimensional vectors $\boldsymbol{\mathcal{B}}_{\mathcal{U},\,p} := \{\boldsymbol{b}_{a_i,\mathcal{U}}\}_{i=1}^{p}$ is a set of linearly independent vectors. Recall from our notation defined in Section 2.1.5, the vector $\boldsymbol{b}_{a_i,\mathcal{U}} \in \mathbb{R}^{|\mathcal{U}|}$ is defined component-wise with

respect to the elementary basis as

$$(\boldsymbol{b}_{a_i,\mathcal{U}})_{\mathcal{E}} := \begin{bmatrix} b_{a_i}(u_1) \\ b_{a_i}(u_2) \\ \vdots \\ b_{a_i}(u_{|\mathcal{U}|}) \end{bmatrix} \in \mathbb{R}_{\text{s}}^{|\mathcal{U}|}, \ \forall i \in p. \tag{3.2}$$

By choosing different $p \in [|\mathcal{U}|]$, we have $\boldsymbol{\mathcal{B}}_{\mathcal{U},\,p}$ is a basis for a p-dimensional subspace of $\mathbb{R}^{|\mathcal{U}|}$.

Consider our observation model setup that was reviewed in Section 2.1.6 and at the beginning of this chapter, under the special case when there is no observation degradation, i.e., $f_{\text{ob}} = f$. We assume that the set $\{a_i\}_{i=1}^{|\mathcal{U}|}$ and the map $c : \mathbb{R}^D \to \mathbb{R}$ is such that

$$f_{\text{ob}} \approx \sum_{i=1}^{|\mathcal{U}|} c(a_i) b_{a_i}(x),$$

where the quality of this approximation depends on the application at hand. The estimation of $\tilde{\boldsymbol{f}}_{\mathcal{X}_q}$ such that $\left(\tilde{\boldsymbol{f}}_{\mathcal{X}_q}\right)_{\mathcal{E}} \approx f|_{\mathcal{X}_q}$ is the goal of the sparse recovery method. It assumes that there exists a $\boldsymbol{\mathcal{B}}_{\mathcal{X}_q,\,|\mathcal{U}|}$ such that its corresponding $\left\{b_{a_i} \in \mathbb{R}^{|\mathcal{X}_q|}\right\}_{i=1}^{|\mathcal{U}|}$ and $c : \mathbb{R}^D \to \mathbb{R}$ gives a good enough approximation

$$f|_{\mathcal{X}_q} \approx \sum_{i=1}^{|\mathcal{U}|} c(a_i) b_{a_i}(x) \tag{3.3}$$

for the application at hand.

Consider the collection of functions

$$\mathcal{B} := \left\{b_a(\cdot) \ \middle| \ a \in \mathbb{R}^D\right\}.$$

Given $\mathcal{X} \subseteq \mathcal{X}_q$ and a finite positive integer N, the sparse recovery method computes $\tilde{\boldsymbol{f}}_{\mathcal{X}_q}$ using the following procedure:

1. Use $f|_{\mathcal{X}}$ to estimate the N most significant elements of $\mathcal{B}$ to approximating $f|_{\mathcal{X}_q}$. Order these elements such that they are identified by $\{a_i\}_{i=1}^{N}$, so we can construct $\boldsymbol{\mathcal{B}}_{\mathcal{X},\,N} := \left\{\boldsymbol{b}_{a_i,\,\mathcal{X}} \in \mathbb{R}^{|\mathcal{X}|}\right\}_{i=1}^{N}$.
2. Compute the best-fit linear expansion coefficients of $\boldsymbol{f}_{\text{ob},\,\mathcal{X}}$ with respect to $\boldsymbol{\mathcal{B}}_{\mathcal{X},\,|\mathcal{X}|}$, according to some given criterion. These are $\{c(a_i)\}_{i=1}^{|\mathcal{X}|}$.
3. Construct $\tilde{f}\Big|_{\mathcal{X}_q} = \sum_{i=1}^{N} c(a_i) b_{a_i}(x)$, where $x \in \mathcal{X}$. The function $\tilde{f}$ is an element of the space of functions
$$\mathcal{F}_{\mathcal{B},\,N} = \left\{\sum_{i=1}^{N} c(a_i) b(x,\, a_i) \ \middle| \ c(a_i) \in \mathbb{R},\, |c(a_i)| < \infty\right\}.$$

To get the vector space version $\tilde{\boldsymbol{f}}_{\mathcal{X}_q}$ of $\tilde{f}\Big|_{\mathcal{X}_q}$, we go through a similar development as Equation 3.2, but for the set of inputs $\mathcal{X}_q$. Define the set of vectors

$$\boldsymbol{\mathcal{B}}_{\mathcal{X}_q,\,N} = \left\{\boldsymbol{b}_{a_i,\,\mathcal{X}_q} \in \mathbb{R}^{|\mathcal{X}_q|}\right\}_{i=1}^{N}$$

where $\forall i \in [N]$, we have

$$\left(\boldsymbol{b}_{a_i,\,\mathcal{X}_q}\right)_{\mathcal{E}} = \begin{bmatrix} b_{a_i}(x_1) \\ \vdots \\ b_{a_i}\left(x_{|\mathcal{X}_q|}\right) \end{bmatrix} \in \mathbb{R}_s^{|\mathcal{X}_q|}. \tag{3.4}$$

In this thesis, we only consider the case when $|\mathcal{X}_q| \geq N$, so that $\boldsymbol{\mathcal{B}}_{\mathcal{X}_q,\,N}$ is a basis for a N-dimensional subspace of $\mathbb{R}^{|\mathcal{X}_q|}$. Since $\{a_i\}_{i=1}^N$ and the map b specifies the basis $\boldsymbol{\mathcal{B}}_{\mathcal{X}_q,\,N}$, they control which subspace we think $\tilde{\boldsymbol{f}}_{\mathcal{X}_q}$ is in. The map c specifies the element in this subspace that we assign as $\tilde{\boldsymbol{f}}_{\mathcal{X}_q}$. When the approximation is good enough for the application at hand, we refer to such a $\boldsymbol{\mathcal{B}}_{\mathcal{X}_q,\,N}$ as a *sufficiently sparse dictionary for* f on $\mathcal{X}_q$. Other authors [65] typically define a dictionary as a sampled version of a Schauder basis for f, but it requires additional technical background that we do not review here. Instead, we just use the informal interpretation that the infinite set $\mathcal{B}$ is capable of representing the important parts of f, in the sense that only a few elements of $\mathcal{B}$ are required to produce a good approximation to any restrictions of f to the finite set $\mathcal{X}_q$.

In the literature, the elements of a dictionary are called *atoms*. When $\boldsymbol{\mathcal{B}}_{\mathcal{X},\,N}$ has more than $|\mathcal{X}|$ elements, it is an *overcomplete dictionary* for $\mathbb{R}^{|\mathcal{X}|}$. When $\boldsymbol{\mathcal{B}}_{\mathcal{X},\,N}$ has less than $|\mathcal{X}|$ elements, it is an *undercomplete dictionary* for $\mathbb{R}^{|\mathcal{X}|}$. In practical applications, one usually does not know the form of the function b such that the set $\boldsymbol{\mathcal{B}}_{\mathcal{X}_q,\,N}$ is a sufficiently sparse dictionary for the function one wants to model. The complete specification of $\boldsymbol{\mathcal{B}}_{\mathcal{X},\,N}$ could be delayed until after seeing some observation of f, and we call this approach to specifying $\boldsymbol{\mathcal{B}}_{\mathcal{X},\,N}$ the *adaptive approach*. Some examples of this approach are the bandlet and the grouplet transforms [68]. Some other schemes for specifying $\boldsymbol{\mathcal{B}}_{\mathcal{X},\,N}$ are the compressed sensing and dictionary learning family of techniques. The compressed sensing approach obtains $\boldsymbol{\mathcal{B}}_{\mathcal{X},\,N}$ by treating it as a realization from a random process, and this yields some useful properties [66]. The research area of finite-dimensional dictionary learning [69] uses some offline data set $\mathcal{D}$ to estimate overcomplete dictionaries, where elements in $\mathcal{D}$ are assumed to be sampled versions of f. Since $\boldsymbol{\mathcal{B}}_{\mathcal{X},\,N}$ for these works is a set of linearly-dependent vectors, this approach suffers from non-identifiability issues. One can circumvent these issues by fixing the overcomplete dictionaries, i.e., seeing any observations of f does not affect its specification [70, 71]. However, such a dictionary cannot utilize information in the observed f to improve the level of sparsity.

Here, we focus on the case when $\boldsymbol{\mathcal{B}}_{\mathcal{X},\,N}$ is a basis, which is when $N = |\mathcal{X}|$. Using $\boldsymbol{\mathcal{B}}_{\mathcal{X},\,|\mathcal{X}|}$, we can uniquely decompose $\boldsymbol{f}_{\text{ob},\,\mathcal{X}}$ as

$$\left(\boldsymbol{f}_{\text{ob},\,\mathcal{X}}\right)_{\mathcal{E}} = Bc = \begin{bmatrix} \boldsymbol{b}_1 & \cdots & \boldsymbol{b}_N \end{bmatrix} \begin{bmatrix} c_1 \\ \vdots \\ c_N \end{bmatrix}, \tag{3.5}$$

where $c = \begin{bmatrix} c_1 & \cdots & c_N \end{bmatrix}^T$ is the column matrix that stores the basis coefficients of $\boldsymbol{f}_{\text{ob},\,\mathcal{X}}$ under $\boldsymbol{\mathcal{B}}_{\mathcal{X},\,|\mathcal{X}|}$, and B is the $\left|\boldsymbol{\mathcal{B}}_{\mathcal{X},\,|\mathcal{X}|}\right| \times |\mathcal{X}|$ matrix where the columns are the vectors in $\boldsymbol{\mathcal{B}}_{\mathcal{X},\,|\mathcal{X}|}$. We used $\boldsymbol{b}_i := \boldsymbol{b}_{a_i,\,\mathcal{X}}$ and $c_i := c(a_i)$ here to reduce notation clutter. To estimate the basis coefficients $\{c_i\}$ from $\boldsymbol{f}_{\text{ob},\,\mathcal{X}}$, one could use some specified cost function that involves $\boldsymbol{f}_{\text{ob},\,\mathcal{X}}$, such as the least-squares objective function, to solve for c from Equation 3.5. One can also jointly solve for $\boldsymbol{\mathcal{B}}_{\mathcal{X},\,|\mathcal{X}|}$ and c , but is likely to run into computational issues that stem from non-identifiability issues. One popular approximation is to solve for a convex relaxation of the original cost function [66], but we do not use this approach in this thesis.

Let $M = |\mathcal{X}_q|$, and $\boldsymbol{\mathcal{U}} = \{\boldsymbol{u}_i\}_{i=1}^N$ be an ordered set of linearly independent vectors in $\mathbb{R}^M$ that has $N = |\mathcal{X}|$ elements. The final step of the sparse recovery method is to obtain a basis

$$\boldsymbol{\mathcal{B}}_{\mathcal{X}_q,\,|\mathcal{X}|} = \left\{\boldsymbol{u}_i \in \mathbb{R}^M\right\}_{i=1}^N$$

for the subspace in $\mathbb{R}^M$ that is spanned by $\boldsymbol{\mathcal{B}}_{\mathcal{X}_q,\,N}$. One can then specify $\tilde{\boldsymbol{f}}_{\mathcal{X}_q}$ in component-form by

$$\left(\tilde{\boldsymbol{f}}_{\mathcal{X}_q}\right)_{\mathcal{E}} := \begin{bmatrix}\boldsymbol{u}_1 & \cdots & \boldsymbol{u}_N\end{bmatrix}\begin{bmatrix}c_1\\ \vdots \\ c_N\end{bmatrix}. \tag{3.6}$$

If $\boldsymbol{\mathcal{B}}_{\mathcal{X},\,|\mathcal{X}|}$ was estimated in a way such that Equations 3.4 and 3.3 are satisfied, then one can choose

$$\boldsymbol{u}_i = \begin{bmatrix} b_{a_i}(z_1)\\ \vdots \\ b_{a_i}(z_M)\end{bmatrix},\ \forall z_i \in \mathcal{X}_{q,\,[M]}. \tag{3.7}$$

This means $\boldsymbol{u}_i = \boldsymbol{b}_{a_i,\,\mathcal{X}_q}$, and this choice of $\boldsymbol{\mathcal{B}}_{\mathcal{X}_q,\,|\mathcal{X}|}$ means that $\tilde{f} \in \mathcal{F}_{\mathcal{B},\,N}$. We see that the sparse recovery method yields a good approximation $\tilde{f}\big|_{\mathcal{X}_q} \approx f|_{\mathcal{X}_q}$ if 1) $\boldsymbol{\mathcal{B}}_{\mathcal{X}_q,\,|\mathcal{X}|}$ is a sparse dictionary of f over $\mathcal{X}_q$, and 2) one can obtain $\boldsymbol{\mathcal{B}}_{\mathcal{X},\,|\mathcal{X}|}$ such that Equations 3.4 and 3.3 are sufficiently satisfied for the application at hand. In summary, the sparse recovery method obtains $\{c_i\}$ from $f|_{\mathcal{X}}$, then uses only $|\mathcal{X}|$ basis vectors to produce $\tilde{f}\big|_{\mathcal{X}_q}$ by using those same $\{c_i\}$ with some basis $\boldsymbol{\mathcal{B}}_{\mathcal{X}_q,\,|\mathcal{X}|}$ for the $|\mathcal{X}|$-dimensional subspace of $\mathbb{R}^{|\mathcal{X}_q|}$.

The RKHS regularization problem reviewed in Section 2.2 with the sample-without-prefiltering observation model (i.e, $f_{\text{ob}} = f$) can be interpreted as a version of the sparse recovery method where an additional constraint is imposed on $\boldsymbol{\mathcal{B}}_{\mathcal{X},\,N}$, where $N = |\mathcal{X}|$. If the matrix $B = \begin{bmatrix}\boldsymbol{b}_1 & \cdots & \boldsymbol{b}_N\end{bmatrix}$ is symmetric and positive-definite, then each component of $\boldsymbol{b}_i$ with respect to the elementary basis of $\mathbb{R}^N$ can be modelled as an evaluation of a symmetric positive-definite kernel function $k(\cdot,\, a_i)$ on $\mathbb{R}^D$ at the observed input x_i, where $a_i \in \mathbb{R}^D$ is given, $\forall i \in [N]$. To see this, let $\mathcal{I}$ denote the order that generated $\mathcal{X}_{[N]}$ from $\mathcal{X}$. The kernel matrix K of k, on the input set $\mathcal{X}$, under the ordering $\mathcal{I}$ is the matrix B; see Definition 2.9 for a review on our definition of a kernel matrix. Therefore, the set of basis coefficients $\{c_i\}$ of $\boldsymbol{f}_{\text{ob},\,\mathcal{X}}$ with respect to $\boldsymbol{\mathcal{B}}_{\mathcal{X},\,|\mathcal{X}|}$ is the set $\{\alpha_i\}$ in Theorem 2.12. This means we solve the optimization problem in Theorem 2.12 to get our basis vector coefficients under the sparse recovery framework. Since this problem is equivalent to solving a linear system that involves a positive-definite matrix, it must have a unique solution.

Definition 2.9 states that each row of a kernel matrix corresponds to a point in $\mathcal{X}$ (under its accompanying ordering $\mathcal{I}$), and a similar correspondence holds for each column, which makes the kernel matrix symmetric. This is because $\mathcal{I}$ is used to map an input $x \in \mathcal{X}$ to an element $\boldsymbol{b}_{\mathcal{I}(x)} \in \boldsymbol{\mathcal{B}}_{\mathcal{X},\,|\mathcal{X}|}$ (recall $\boldsymbol{b}_i := \boldsymbol{b}_{a_i,\,\mathcal{X}}$) and the functions b that make up $\mathcal{B}$ is given by

$$b\big(\cdot,\, a_{\mathcal{I}(x)}\big) = k\big(\cdot,\, a_{\mathcal{I}(x)}\big),\ \forall x \in \mathcal{X}. \tag{3.8}$$

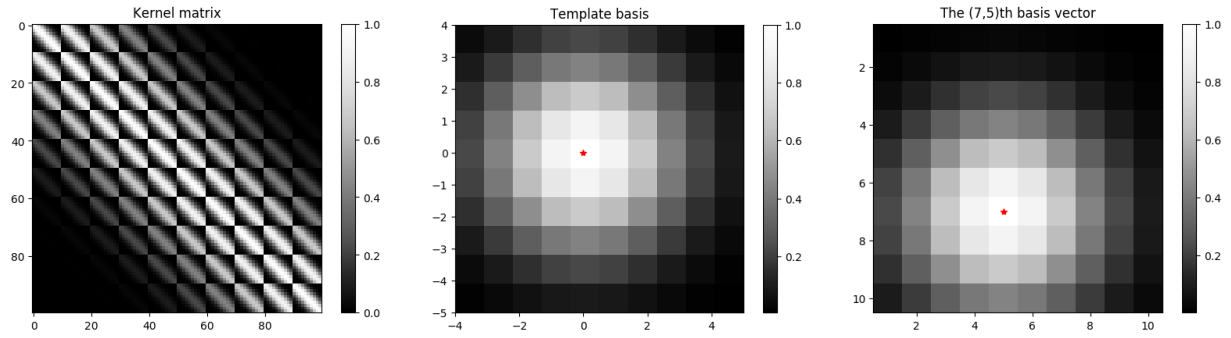
It follows that the vectors in $\boldsymbol{\mathcal{B}}_{\mathcal{X}_q,\,|\mathcal{X}|}$ are related to the kernel by

$$\boldsymbol{u}_i = \begin{bmatrix} k(z_1,\, x_i)\\ \vdots \\ k(z_M,\, x_i)\end{bmatrix},\ \forall x_i \in \mathcal{X}_{[N]},\ \forall z_i \in \mathcal{X}_{q,\,[M]}, \tag{3.9}$$

and the vectors in $\boldsymbol{\mathcal{B}}_{\mathcal{X},\,|\mathcal{X}|}$ are related to k by

$$\boldsymbol{b}_i = \begin{bmatrix} k(x_1,\, x_i)\\ \vdots \\ k(x_N,\, x_i)\end{bmatrix},\ \forall x_i \in \mathcal{X}_{[N]}. \tag{3.10}$$

It also follows that the RKHS regularization solution must an element of $\mathcal{F}_{\mathcal{B},\,N}$, even though the solution space of that regularization problem is the RKHS of the kernel k; see the form of $\tilde{f}$ in Theorem 2.12 for



(a) **(b)** **(c)**

Figure 3.1: A column of a kernel matrix is depicted as a basis vector for $\mathbb{R}^{10\times 10}$. (a) The kernel matrix on $\mathcal{X} = \{1:10\}^2$, under the column-major ordering. (b) The template $\boldsymbol{b}_0$. (c) The 47th column of the kernel matrix, rearranged to make a basis vector. The red star marks the peak of the kernel.

details. The significance of our discussion in this section is that we do not require any infinite-dimensional Hilbert space theory to define the space $\mathcal{F}_{\mathcal{B},\,N}$.

If a stationary kernel is used, then the vectors in $\boldsymbol{\mathcal{B}}_{\mathcal{X},\,|\mathcal{X}|}$ can be interpreted as being parameterized by a translation parameter $x_i \in \mathcal{X}$. Let $\boldsymbol{b}_0 \in \mathbb{R}^N$ be the vector with its components with respect to the elementary basis of $\mathbb{R}^N$, given component-wise by

$$(\boldsymbol{b}_0)_{\mathcal{E},\,i} := k(x_i,\, x_0),\ \forall x_i \in \mathcal{X}_{[N]},$$

where x_0 is the zero vector of $\mathbb{R}^D$. We can interpret the ith basis vector $\boldsymbol{b}_i$ as a translation of $\boldsymbol{b}_0$ such that its center is moved from $\mathbf{0} \in \mathbb{R}^D$ to the location x_i in $\mathcal{X}_{[N]}$. We can take any column of the kernel matrix, e.g., the column that corresponds to the input $x \in \mathcal{X}$, and reshape this column matrix into a basis vector for the observed signal, $\boldsymbol{b}_{\mathcal{I}(x)}$. Here, $\mathcal{I}(x) \in [N]$ is the column index of the kernel matrix that corresponds to x. Figure 3.1 illustrates this idea for the inputs on a uniform 2-dimensional grid from position $(1,\,1)$ to position $(10,\,10)$. Figure 3.1(c) is the $\mathcal{I}(x) = (5-1)\times 10 + 7 = 47$th basis vector, where $x = (7,\,5)$ is the position of the red marker, and $\mathcal{I}$ is the column-major ordering. It is a translated version of $\boldsymbol{b}_0$.

If our definition of a kernel function in Definition 2.8 did not require the kernel matrix to be positive-definite, then the solution of the optimization problem in Theorem 2.12 is not guaranteed to be unique. We conclude that the RKHS regularization framework can be interpreted as a type of sparse recovery scheme where 1) the computation of the basis vector coefficients $\{c(a_i)\}_{i=1}^{|\mathcal{X}|}$ and the functions $\{k(\cdot,\, a_i)\}_{i=1}^{|\mathcal{X}|}$ (which fully specifies the bases $\boldsymbol{\mathcal{B}}_{\mathcal{X},\,|\mathcal{X}|}$ and $\boldsymbol{\mathcal{B}}_{\mathcal{X}_q,\,|\mathcal{X}|}$) is not done jointly with respect to a common optimization objective function, and 2) the solution of the basis vector coefficients is unique. We see that the kernel should be chosen so that its induced infinite collection $\mathcal{B}$ gives rise to a sparse dictionary $\boldsymbol{\mathcal{B}}_{\mathcal{X}_q,\,|\mathcal{X}|}$ for any $\mathcal{X}_q$ that is encountered in the application at hand. This framework does not use any relaxation approach to solve some computationally-expensive optimization objective when solving for the basis coefficients. It is well-established in the literature (e.g., [2, 49, 51]), with extensions to vector-valued function spaces [7, 6].

3.3 Adaptive Kernels

Our plan is to adapt a given stationary and isotropic kernel on $\mathbb{R}^D$, $s : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$, where $D > D_0$, to the local patterns of the target signal f such that the resultant adaptive kernel $k : \mathbb{R}^{D_0} \times \mathbb{R}^{D_0} \to \mathbb{R}$ induces a sparse dictionary. We model k as a composition of s with a map $w : \mathbb{R}^{D_0} \to \mathbb{R}^D$. We call s the *canonical kernel*, and w the *warp map*. The works [10, 11] also took a function composition approach, where w is

fitted from the observations via some optimization objective function. We take a more constrained approach that forces w to be the identity map for the first D_0 dimensions, so that one only needs to specify $D - D_0$ dimensions.

Consider the case where $D = 3$ and $D_0 = 2$. We can interpret the identity map constraint on w in the following way: w augments each signal location $x \in \mathbb{R}^2$ an extra dimension that takes on the value $\phi(x) \in \mathbb{R}$. The input that the canonical kernel s sees is $\left[\begin{smallmatrix} x \\ \phi(x) \end{smallmatrix}\right] \in \mathbb{R}^3$. Since s is a stationary and isotropic kernel, it only cares about the distance between its two inputs. We could increase the effective distance seen by s from two inputs by making w assign very different values for these two locations; i.e., we choose w so that we get a large distance between input pairs that have very different signal values, even if they are very close together on the signal's input space of $\mathbb{R}^2$. Now, let us extend this idea to the general case of $D > D_0$.

Theorem 3.1. *Let $s : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$ be a symmetric positive-definite and isotropic kernel that has constant kernel parameters across its domain. Let $D > D_0$. Define $w : \mathbb{R}^{D_0} \to \mathbb{R}^D$, with its components with respect to the elementary basis for $\mathbb{R}^D$ being*

$$(w(x)) = \begin{bmatrix} (x) \\ \phi(x) \end{bmatrix},$$

where $\phi : \mathbb{R}^D \to \mathbb{R}_s^{D-D_0}$ is a map with bounded output. The function

$$\begin{aligned} k : \mathbb{R}^{D_0} \times \mathbb{R}^{D_0} &\to \mathbb{R} \\ (a, b) &\mapsto s(w(a),\, w(b)) \end{aligned}$$

is then a symmetric positive-definite kernel on $\mathbb{R}^D$.

Proof. k is a symmetric kernel because s is symmetric. The positive-definiteness portion of this proof is essentially the proof for the fact that any principal submatrix of a positive-definite matrix is positive-definite. Consider an arbitrary finite set $\mathcal{X}$ of N unique points in $\mathbb{R}^{D_0}$, and any set $\mathcal{Y}$ of $M \geq N$ unique points in $\mathbb{R}^D$ that contain the set $\mathcal{P} = w(\mathcal{X})$. Let $S(\mathcal{Y})$ be the kernel matrix (see Definition 2.9) that corresponds to the evaluation of s at each of the inputs $y \in \mathcal{Y} \subseteq \mathbb{R}^D$. The ordering of inputs in $\mathcal{Y}$ that is used to generate $S(\mathcal{Y})$ does not affect this result. Similarly define the kernel matrix K for k and $\mathcal{X}$ under a fixed arbitrary ordering.

Define the map between column matrices $u : \mathbb{R}_s^{D_0} \to \mathbb{R}_s^D$ in an entry-wise manner: $\forall j \in [D], i \in [D_0]$,

$$u_j(z) := \begin{cases} z_i & \text{if the } j\text{th column of S}(\mathcal{Y}) \text{ corresponds to } w(z_i) \text{ in } \mathcal{P} \\ 0 & \text{otherwise.} \end{cases}$$

We then have

$$z^T K(\mathcal{X}) z = u(z)^T S(\mathcal{Y}) u(z) > 0$$

for any non-zero $z \in \mathbb{R}_s^{D_0}$, which shows that k is positive-definite. The inequality comes from the facts that 1) $u(z)$ is non-zero if z is non-zero, and 2) S is a $D \times D$ positive-definite matrix since s is a positive-definite kernel on $\mathbb{R}^D$. □

Since an adaptive kernel k on $\mathbb{R}^{D_0}$ is essentially a restriction of the canonical kernel s on $\mathbb{R}^D$ to the set $w(\mathbb{R}^{D_0})$, one should use a canonical kernel that has good numerical conditioning in practical applications. With a suitable choice of w, it is possible for an adaptive kernel to have oscillatory behaviour even when its canonical kernel is unimodal. An uncomplicated canonical kernel s_1 is a better choice than a more sophisticated kernel s_2 if the kernel matrix of s_1 is significantly better conditioned than the kernel matrix of s_2 over the given set of inputs. The compactly supported, unimodal, spline kernel from Example 2.17 is used for the demonstration in Section 3.4.

The adaptive kernel requires $\phi : \mathbb{R}^D \to \mathbb{R}^{D-D_0}$ to be specified on the input domain $\mathbb{R}^D$. In most applications, one can justify some estimation scheme for the set of points $\{w(x)\}_{x\in\mathcal{X}}$ from $f_{\text{ob}}|_{\mathcal{X}}$, but this does not completely define the map w for all inputs on its domain. One simple specification when $D = D_0+1$ is to simply assign $\phi(\mathcal{X}) = f_{\text{ob}}|_{\mathcal{X}}$, the use some low-complexity interpolator to get the values of $\phi(x)$, $\forall x \in \mathbb{R}^2 \setminus \mathcal{X}$. We can interpret the adaptive kernel as a scheme that uses a low-complexity interpolator as an intermediate step to up-convert a signal via the sparse recovery framework that was discussed in Section 3.2. An interesting future direction is to explore whether some hyperparameters of ϕ could be optimized in an offline training setup.

3.4 Demonstration: Grayscale Image Up-Conversion

For the rest of this chapter, we focus on a demonstration of how one can come up with a warp map that removes aliasing artefacts along the edges of a grayscale image. Unless otherwise stated, f is a non-degraded, continuous-domain grayscale image, and f_{obs} is the observed continuous-domain image just before sampling. The Kodak image set from the *Stanford center for image systems engineering* (SCIEN, https://scien.stanford.edu/, retrieved August 2017) was used. For evaluation purposes, we defined the reference images to be the grayscale versions of the images from this data set, i.e., $f_{\text{ref}} := f|_{\mathcal{X}_q}$. The images $(\boldsymbol{f}_{\text{ref}})_{\mathcal{E}}$ and $\left(\boldsymbol{f}_{\text{ob},\,\mathcal{X}}\right)_{\mathcal{E}}$, and the set of pixel positions $\mathcal{X}$ and $\mathcal{X}_q$ are constructed from the grayscale versions of this image set; the details are reported in Appendix A.

The observation model (see Section 2.1.6) that we focus on is sample-without-prefiltering, but we include image similarity scores for the sample-with-prefiltering model as well. The goal of this demonstration is to show how one can use an adaptive kernel to encode prior assumptions about the signal of interest. The prior assumption used here is that edges should have smooth contours. Our challenge is to come up with a warp map that reduces the effect of the adaptation when the prior assumption is invalid.

$D = 3$ and $D_0 = 2$ were used for this demonstration. The compactly supported, unimodal, spline kernel from Example 2.17 was used, and all interpolation tasks on a grid were performed by the cubic-spline interpolator from the *Grid operations for the Julia language* software package (https://github.com/timholy/Grid.jl, retrieved August 2017). The patch-based RKHS framework from Example 2.19 was used, with the maximum patch size set to 20×20 pixels, and an overlap of 4 pixels. In this section, a *pixel unit* refers to the length of a pixel in the observed image $\left(\boldsymbol{f}_{\text{ob},\,\mathcal{X}}\right)_{\mathcal{E}}$, unless explicitly stated otherwise.

3.4.1 Local Analysis

We want to reduce the adaptation effect when we encounter textured regions or corners. These image features tend to have multiple edges concentrated in a small region. To detect the presence of these patterns, a local analysis of the estimated gradient field of f_{ob} was used. A sliding window scheme that extracts the p-most dominant gradient directions via k-means clustering was used as a pre-processing step. This reduces the number of gradient directions that we have to consider. The *Clustering.jl* software package (https://github.com/JuliaStats/Clustering.jl, retrieved August 2017) was used to perform k-means clustering. The dominant directions are represented as vectors on $\mathbb{R}^2$, where the magnitude specifies the degree of their dominance in the analysis window.

Consider a positive integer $M \geq p$. The idea here is to use a function that 1) maps a set of p non-collinear vectors in $\mathbb{R}^D$ to a set of linearly independent vectors in $\mathbb{R}^M$, and 2) maps collinear vectors in $\mathbb{R}^D$ to linearly dependent vectors in $\mathbb{R}^M$. An example is the map that takes an ordered set of vectors as inputs:

$$\text{Vand}_{D,\,L,\,p}(\cdot) : \left\{\mathbb{R}^D\right\}^p \to \mathbb{R}_s^{M\times p},$$

where L is a finite positive integer, and $M = \sum_{l=0}^{L} \binom{l+D-1}{l}$. The range of this map is a matrix where each column corresponds to the components of a basis vector, with respect to the elementary basis of $\mathbb{R}^M$, that

collectively form a particular multivariate basis for $\mathbb{R}^M$. The general specification of this map is denoted by $\mathtt{V}(\mathcal{C}_{\succ}, \mathfrak{z}, c)$, and the objects $\mathcal{C}_{\succ}$, $\mathfrak{z}$, and c are explained in greater detail in Section 2.13.4. The case $D = 2$, $L = 6$, and $p = L - 1$ was empirically found to give good performance, and was used in our benchmarks. The corresponding value for M is 28. This is valid for our purposes here because it is possible for a 28×5 matrix to have full column rank; we can determine the presence of nearly collinear vectors by checking if any of the p singular values are close to zero. For any ordered set of vectors in $\mathbb{R}^2$, $\mathcal{V}_{[p]} = \{v_i\}_{i=1}^{p}$, we define the map $\text{Vand}_{2,\,6,\,5}(\cdot) : \{\mathbb{R}^2\}^5 \to \mathbb{R}_s^{28\times 5}$ in a point-wise manner:

$$\left(\text{Vand}_{2,\,6,\,5}\left(\mathcal{V}_{[p]}\right)\right)_{i,\,j} := (v_j)_{\mathcal{E},\,1}^{\alpha_{i,\,1}} (v_j)_{\mathcal{E},\,2}^{\alpha_{i,\,2}},\ \forall i \in [28],\ \forall j \in [5], \tag{3.11}$$

where

$$\begin{aligned}\alpha :=&((0,0),\,(1,0),\,(0,1),\,(2,0),\,(1,1),\,(0,2),\,(3,0),\,(2,1),\,(1,2),\,(0,3),\\ &(4,0),\,(3,1),\,(2,2),\,(1,3),\,(0,4),\,(5,0),\,(4,1),\,(3,2),\,(2,3),\,(1,4),\\ &(0,5),\,(6,0),\,(5,1),\,(4,2),\,(3,3),\,(2,4),\,(1,5),\,(0,6)) \in \left\{\mathbb{Z}_s^2\right\}^{28}\end{aligned} \tag{3.12}$$

is a 28-tuple of 2-dimensional multi-indices where each dimension takes on a value from 0 to 6. We explicitly gave the polynomial ordering here as the object α, but the general method is discussed in Section 2.14.4 and Appendix B.

The closer the original vectors in $\mathbb{R}^D$ are to being collinear, the closer the resultant vectors in $\mathbb{R}^M$ are to being linearly dependent; i.e., the matrix $\text{Vand}_{2,\,6,\,5}\left(\mathcal{V}_{[p]}\right)$ gets closer to being singular. A score can now be formulated based on the singular values of this matrix, and we call it the *collinearity score*. The resultant effect is that the "busier" the local neighbourhood around a pixel location is, the larger the singular values of this matrix. Algorithm 3.1 describes the local analysis and the collinearity score. Note that $\mathtt{V} := \text{Vand}_{2,\,6,\,5}\left(\mathcal{V}_{[p]}\right)$ in this algorithm.

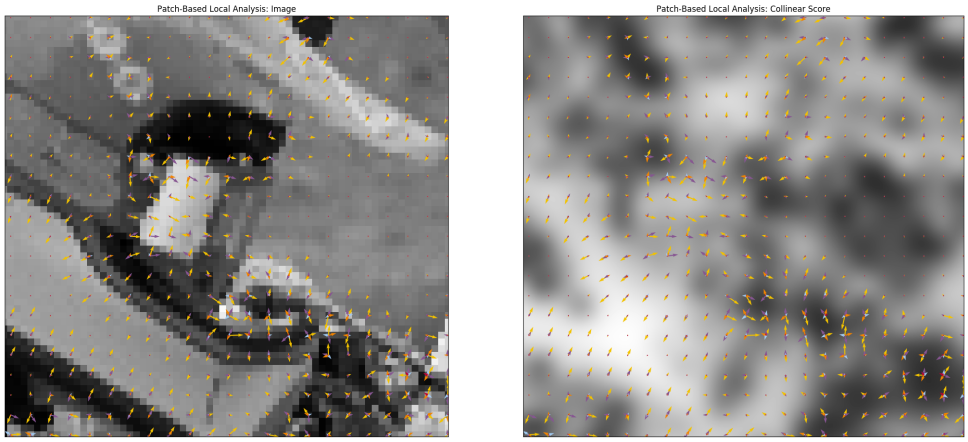
Algorithm 3.1 Collinearity score of a set of vectors

Require: $\mathcal{V}_{[p]} = \{v_j\}_{i=1}^{p}$ be sorted in descending Euclidean norm, and $\xi \in [0, 1]$

for $j \in [p]$ **do**
  $w_j := \frac{v_j}{\|v_1\|}$ ▷ Normalize each $v_j \in \mathcal{V}_{[p]}$ so that $\|w_1\| = 1$.
end for
Compute $\mathtt{V}$ from $\mathcal{V}_{[p]}$, with $L = p + 1$.
Store in the array s the singular values of $\mathtt{V}$ that are normalized so that the largest value is 1.
Sort s in descending magnitude.
for $i \in \{2 : p\}$ **do**
  $u_i = \xi - s_i$
  Clamp each u_i to a minimum of zero, and a maximum of ξ.
end for
Return $\boldsymbol{\kappa} = \sum_{i=2}^{p} \frac{u_i}{\xi(p-1)}$.

Let $\mathcal{V}_{[p]} = \{v_i\}_{i=1}^{p}$ be the set of the p-most dominant gradient directions, sorted in descending Euclidean norm. In Algorithm 3.1, the collinearity score is the sum of $\mathtt{V}$'s normalized singular values that are less than or equal to ξ. Each s_i can have a maximum contribution of $\frac{1}{p-1}$, which is when $s_i = 0$. If $s_i > \xi$, then the ith vector is deemed too far from being collinear, and the contribution from the ith dominant direction to the score is zero. A large value of ξ allows vectors that are not close to being collinear to contribute to the collinearity score. The opposite is true for a small value of ξ. The dominant direction is excluded from the sum. I found that the strategies that we present in the following sections performed well if $\xi \in [0.15,\ 0.3]$.

I implemented two versions of this local analysis idea. The *dense local analysis* uses a sliding window to compute a $\mathcal{V}_{[p]}$ centered at each location in $\mathcal{X}$, then computes the collinearity score via Algorithm 3.1



(a) **(b)**

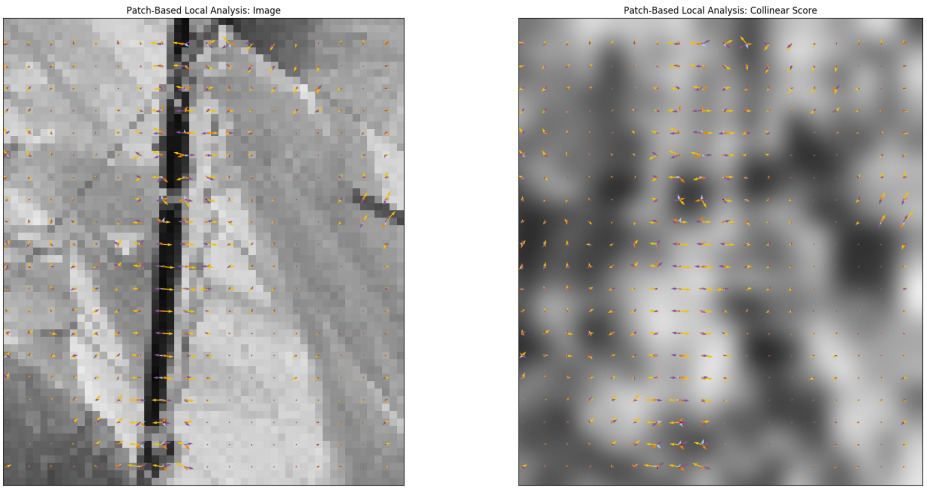
Figure 3.2: Patch-based local analysis on a portion of Kodak scene 5. For each image patch in the observed image $\boldsymbol{f}_{\text{ob}}$, a set of five colour-coded orientation vectors are placed such that their origin is at the center of the patch. The colour-code order is yellow, purple, orange, cyan, and red. It sorts the set of vectors according to descending Euclidean norm. The image background is $\boldsymbol{f}_{\text{ob}}$ for (a), and the collinear scores $\boldsymbol{\kappa}$ that corresponds to the image patches of $\boldsymbol{f}_{\text{ob}}$ for (b). The arrows are more visually discernible when zoomed in.

at each location. The *patch-based local analysis* first segments $\mathcal{X}$ into patches in a manner similar to the patch system in Example 2.19, and then $\mathcal{V}_{[p]}$ is computed with the analysis window being the specified patch size. Note that the patch size used here to compute $\mathcal{V}_{[p]}$ is separately defined from the patch size used for the patch-based RKHS framework. The collinear score is then computed via Algorithm 3.1 for each patch center, and then interpolated for the rest of the locations in $\mathcal{X}$. The patch-based local analysis tends to be less sensitive to tuning parameters such as ξ and p, and is faster to compute than the dense version. The dense local analysis is more responsive in detecting sudden changes in scene content.

Let the *collinearity score map* $\kappa : \mathbb{R}^D \to \mathbb{R}_+$ be the map that takes any input on the domain of f to a collinearity score. We treat the collinearity score that is returned from Algorithm 3.1 as $\kappa(x)$ at some input $x \in \mathcal{X}$ if a dense local analysis is used. We have $x \in \mathcal{P}$ if a patch-based local analysis is used, where $\mathcal{P} \subset \mathcal{X}$ is the space of patch centers. We denote by $\kappa|_{\mathcal{X}_{[N]}}$ the *observed collinearity score*, and denote by $\boldsymbol{\kappa}$ the 2-dimensional array representation of $\kappa|_{\mathcal{X}_{[N]}}$. In the patch-based scenario, we interpolate $\kappa|_{\mathcal{P}_{[|\mathcal{P}|]}}$ to get $\kappa|_{\mathcal{X}_{[N]}}$, before proceeding to storing $\kappa|_{\mathcal{X}_{[N]}}$ as the array $\boldsymbol{\kappa}$.

The following are some examples of $\boldsymbol{\kappa}$ from a patch-based local analysis with $p = 5$, $\xi = 0.23$. It takes longer to process a large patch, but there are less patches to process for a given image. We chose a patch size of 6×6 pixel units with a 3 pixel overlap for our patch-based local analysis, measured with respect to the grid that $\mathcal{X}$ is on. The gradient field of f_{ob} was estimated from $\left(\boldsymbol{f}_{\text{ob},\,\mathcal{X}}\right)_{\mathcal{E}}$ by a separable cubic-spline derivative filter of size 5×5 pixel units. These values were chosen empirically. In Figures 3.2(a), 3.3(a), and 3.4(a), the background of the figures are set to be the array version of the observed images $\left(\boldsymbol{f}_{\text{ob},\,\mathcal{X}}\right)_{\mathcal{E}}$ of three different scenes, under the sample-without-prefiltering observation model. $\left(\boldsymbol{f}_{\text{ob},\,\mathcal{X}}\right)_{\mathcal{E}}$ was generated from the grayscale versions of the Kodak image set as specified in Appendix A with the downsample factor set to 5.

In Figures 3.2(b), 3.3(b), and 3.4(b), the background of the figures are set to be the $\boldsymbol{\kappa}$ arrays of three different scenes. Kodak scene 5 contains many high-contrast edges, Kodak scene 9 contains softer edges, and Kodak scene 13 features some rocks that are severely aliased. The collinearity score is usually high near high-contrast edges in these scenes. We see that the corners of the white rectangular goggles in Figure 3.2(a)



(a) **(b)**

Figure 3.3: Patch-based local analysis on a portion of Kodak scene 9. For each image patch in the observed image $\boldsymbol{f}_{\text{ob}}$, a set of five colour-coded orientation vectors are placed such that their origin is at the center of the patch. The colour-code order is yellow, purple, orange, cyan, and red. It sorts the set of vectors according to descending Euclidean norm. The image background is $\boldsymbol{f}_{\text{ob}}$ for (a), and the collinear scores $\boldsymbol{\kappa}$ that corresponds to the image patches of $\boldsymbol{f}_{\text{ob}}$ for (b). The arrows are more visually discernible when zoomed in.

have spread out dominant directions, and thus have a lower collinearity score. The masts and edges of the boat are properly analyzed in Figure 3.3, but the downsample amount is too large for the rigging lines to show up as edges in our local analysis. The rocks in Figure 3.4 are varying too fast in signal intensity relative to the local analysis window, so most of the regions have low collinear scores. We use collinearity scores in our warp map construction strategies because they quantitatively identify regions that should receive less adaptation.

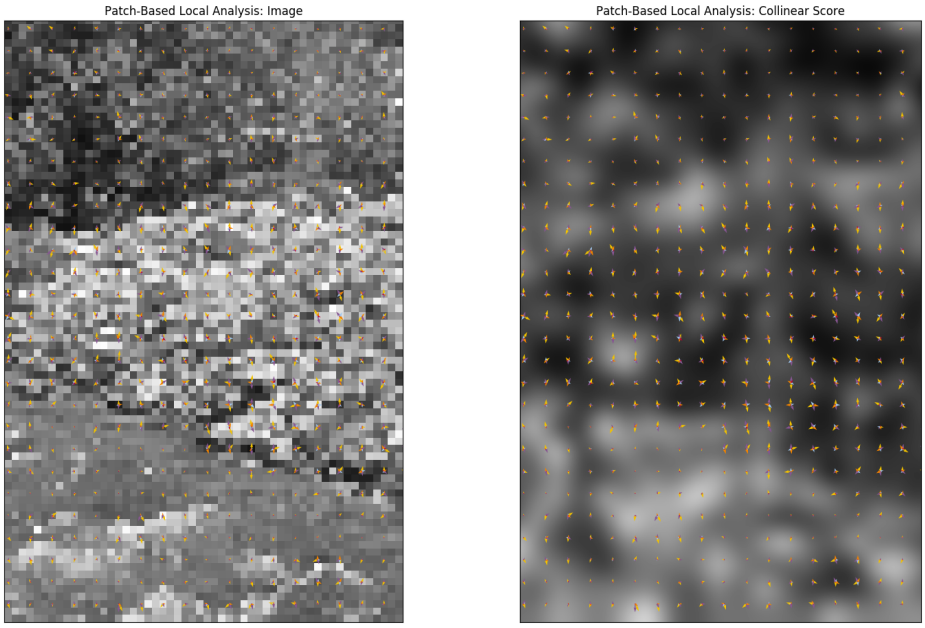
We do not show illustrations of the dense local analysis because of the large amount of arrows that are present in such an illustration. The dense local analysis looks similar to the patch-based local analysis, except the a collinearity score and a set of dominant directions is computed for each pixel location in the observed image.

3.4.2 The Adaptive Smoothing Strategy

Our chosen prior assumption on the signal implies that the dictionary atom functions $k(\cdot, x_i)$, $i \in [N]$, for each image patch should be smoother along edges than the observed image patch. For this subsection, we write f to mean a patch in the observed image. The strategy discussed here sets $\phi|_{\mathcal{X}}$ to be a smoothed version of $f_{\text{ob}}|_{\mathcal{X}}$, where high-contrast edges are smoothed more. The rest of ϕ is constructed by using an interpolator on $\phi|_{\mathcal{X}}$. The warped distances seen by the canonical kernel on $\mathbb{R}^3$ changes in a smoother manner around high-contrast edges, which leads to a smoother contour in the atom function that it instantiates on $\mathbb{R}^2$. The patch-based local analysis was used to get $\kappa : \mathbb{R}^D \to \mathbb{R}_+$ because the dense local analysis was too sensitive to the analysis window size and ξ.

Algorithm 3.2 is used to non-homogeneously smooth a D-dimensional array $\boldsymbol{y} := \left(\boldsymbol{f}_{\text{ob},\,\mathcal{X}}\right)_{\mathcal{E}}$ according to the set of smoothing radii which we take to be $\boldsymbol{\kappa}$. We assume both $\boldsymbol{y}$ and $\boldsymbol{\kappa}$ have the same size.

Since ϕ is based on $f_{\text{ob}}|_{\mathcal{X}}$ without any pre-processing, it has a wide dynamic range. This makes it hard for us to control the amount of adaptation we want to set.



(a) **(b)**

Figure 3.4: Patch-based local analysis on a portion of Kodak scene 13. For each image patch in the observed image $\boldsymbol{f}_{\text{ob}}$, a set of five colour-coded orientation vectors are placed such that their origin is at the center of the patch. The colour-code order is yellow, purple, orange, cyan, and red. It sorts the set of vectors according to descending Euclidean norm. The image background is $\boldsymbol{f}_{\text{ob}}$ for (a), and the collinear scores $\boldsymbol{\kappa}$ that corresponds to the image patches of $\boldsymbol{f}_{\text{ob}}$ for (b). The arrows are more visually discernible when zoomed in.

Algorithm 3.2 Variable spatial low-pass filter. The operator $.*$ is per-element multiplication of two arrays.

Require: maximum smoothing radius a, collinear threshold $t \in \mathbb{R}_+$, the signal $\boldsymbol{y} \in \mathbb{R}^N$ to be smoothed, and the interpolated weight function κ.

for $x_i \in \mathcal{X}$ **do**
  $u_i := \kappa(x_i)$ ▷ obtain the interpolated evaluation of κ at x_i.
end for
for $i \in [N]$ **do**
  if $u_i > ta$ **then**
    $W_i = \text{getGaussianweights}(u_i)$ ▷ sample weights from an isotropic 2-D Gaussian of radius u_i.
    $\tilde{\boldsymbol{y}}_i := \boldsymbol{y}_i . * W_i$
  else
    $\tilde{\boldsymbol{y}}_i := \boldsymbol{y}_i$
  end if
end for
Return $\tilde{\boldsymbol{y}}$.

3.4.3 The Monogenic Adaptive Smoothing Strategy

This strategy uses a pre-processing step on $f_{\text{ob}}|_{\mathcal{X}}$ to reduce its dynamic range before applying adaptive smoothing, such that ϕ is close to zero except near edges. Here, f refers to the entire image. Let L and D be finite positive integers, $\eta_i f_{\text{ob}}$ be an isotropically band-passed version of f_{ob} at radial frequency ν_i; i.e., the filter passbands are ring-shaped on the 2-dimensional plane. Let $H : \mathbb{R}_s^M \to \mathbb{R}$ be defined as

$$H \circ \mathcal{R}^{(L)} \eta_i f_{\text{ob}} = \sum_{n \in \mathcal{I}(2,\, L)} \binom{L}{n}^2 \mathcal{R}^n \eta_i f_{\text{ob}},$$

where $\mathcal{R}^n$ is the nth-order Riesz transform; see Section 2.5 for details. Let θ denote the monogenic phase of the signal $\eta_i f_{\text{ob}}$. Here, we denote by $J_i : L_2(\mathbb{R}^D) \to L_2(\mathbb{R}^D)$ as the ith portion of the pre-processing step, and define it as

$$\begin{aligned} J_i(\eta_i f_{\text{ob}}) : &= \cos\theta \cdot H \circ \mathcal{R}^{(L)} \eta_i f_{\text{ob}} \\ &= \frac{\sin\theta}{\tan\theta} \cdot H \circ \mathcal{R}^{(L)} \eta_i f_{\text{ob}} \\ &= \sin\theta \cdot \eta_i f_{\text{ob}}, \end{aligned} \tag{3.13}$$

where the following relationship is used in the last line:

$$\tan\theta = \frac{H \circ \mathcal{R}^{(L)} \eta_i f_{\text{ob}}}{\eta_i f_{\text{ob}}}.$$

The entire pre-processing step is defined as

$$\begin{aligned} J : L_2\big(\mathbb{R}^D\big) \times \mathbb{R}_s^M &\to L_2\big(\mathbb{R}^D\big) \\ \big(f_{\text{ob}},\, (\nu_1,\, \nu_2, \cdots,\, \nu_M)\big) &\mapsto \sum_{i=1}^{M} J_i(\eta_i f_{\text{ob}}), \end{aligned}$$

and it requires the specification of a set of analysis frequencies $\nu_{\text{set}} = \{\nu_i\}_{i=1}^M$. The pre-processing step is summarized in Algorithm 3.3.

In practice, one computes the Riesz transform on discrete signals such as $\big(\boldsymbol{f}_{\text{ob},\,\mathcal{X}}\big)_{\mathcal{E}}$, and thus we implement a version of J that operates on such discrete signals as opposed to functions in $L_2(\mathbb{R}^D)$. In Algorithm 3.3, we abuse the notation on J and write $J(\eta_i\, f_{\text{ob}}|_{\mathcal{X}},\, \nu_{\text{set}})$ when we mean the version of J that operates on the discrete signal form of $f_{\text{ob}}|_{\mathcal{X}}$. Similarly, we write $\eta_i\, f_{\text{ob}}|_{\mathcal{X}}$ to mean bandpass filtering is done on the discrete signal form of $f_{\text{ob}}|_{\mathcal{X}}$. Since $f_{\text{ob}}|_{\mathcal{X}} : \mathcal{X} \to \mathbb{R}$, the outputs of $J(\eta_i\, f_{\text{ob}}|_{\mathcal{X}},\, \nu_{\text{set}})$ and $\eta_i\, f_{\text{ob}}|_{\mathcal{X}}$ are also maps from $\mathcal{X}$ to $\mathbb{R}$.

The choice of this H quickly saturates θ to be close to $\frac{\pi}{2}$ when the order L of the Riesz transform is increased, and so $J(\eta_i f_{\text{ob}}) \approx \eta_i f_{\text{ob}}$ at high orders (see Figure 3.5). Since the bandpassed version of f_{ob} is sparse in areas without edges, it is easier to come up with schemes that set ϕ to zero in regions where we want to reduce the adaptation; i.e., where the collinearity score map κ take on low values. One can set the order to a low number if a sparse ϕ is desired. Under this scheme for generating ϕ, the even orders of the Riesz transform yielded a smoother ϕ with less ringing artefacts than the odd orders. This empirical observation might be a consequence of the relationship between the higher-order Riesz transform and higher-order derivatives as discussed in Section 2 of [46]. We used only the even orders in our illustrations and benchmarks.

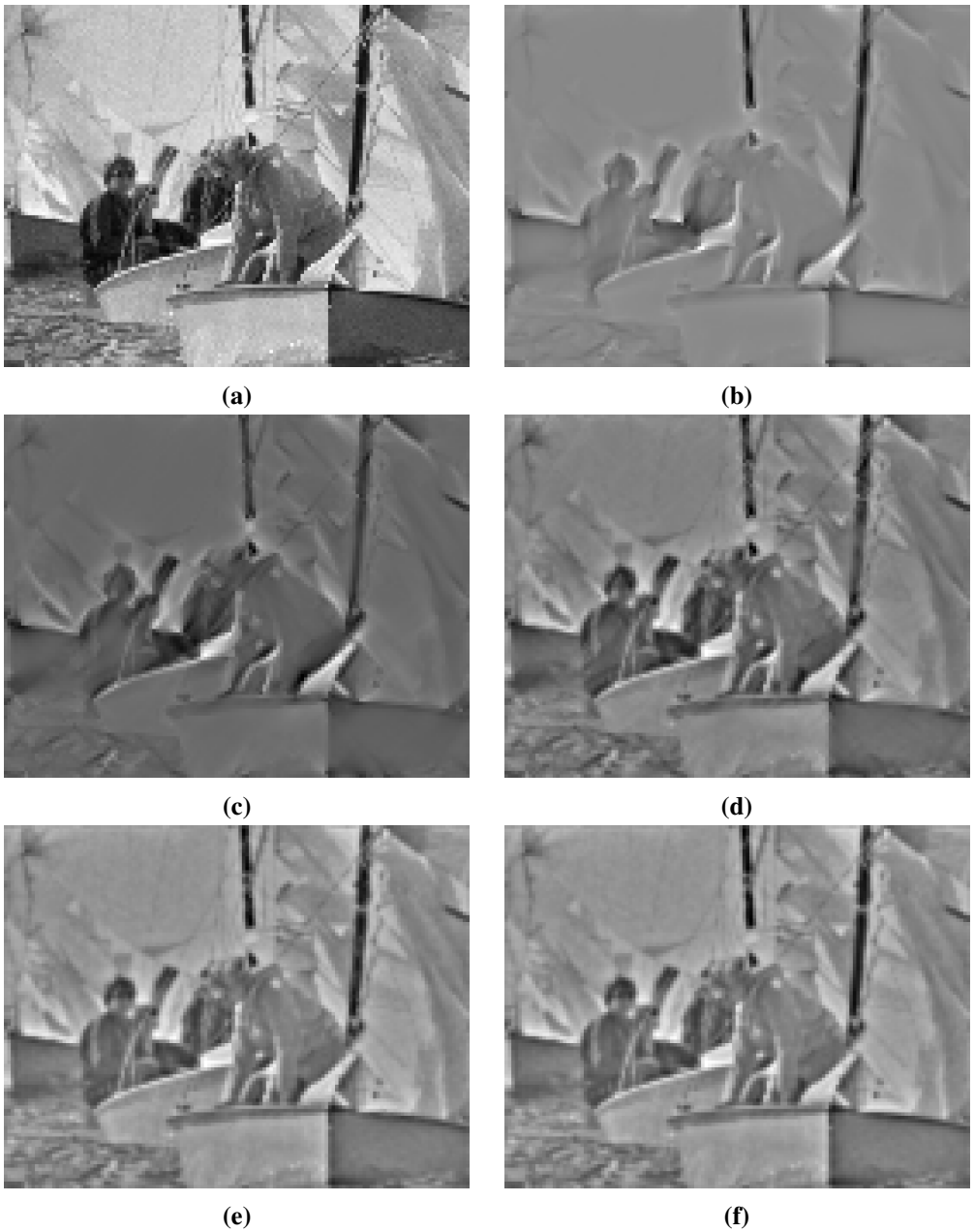


Figure 3.5: Comparison between different values of L for the pre-processed image $J(\eta_i f_{\text{ob}}|_{\mathcal{X}}, \nu_{\text{set}})$ from Algorithm 3.3. The pre-processed images are normalized such that values are between 0 and 1, and they all used the set of cut-off frequencies $\nu_{\text{set}} = \{0.1 : 0.1 : 0.5 \,\text{cycles/pixel}\}$ in their construction. (a) The observed image $\boldsymbol{f}_{\text{ob},\,\mathcal{X}}$ is a portion Kodak scene 9. (b) Pre-processed result when $L = 2$, (c) $L = 6$, (d) $L = 10$, (e) and $L = 14$. (f) The average of the bandpassed version of the observed image f_{ob} to the frequencies in ν_{set}, i.e., $\frac{1}{5}\sum_{i=1}^{5}(\nu_i f_{\text{ob}}|_{\mathcal{X}})$.

Algorithm 3.3 Monogenic adaptive strategy for constructing the warp map. The operator $.*$ is per-element multiplication of two arrays.

Require: $\nu_{\text{set}} = \{\nu_i\}_{i=1}^{M}$, order L, and low-pass frequency ν_{LP}.

1: Compute $J(\eta_i\, f_{\text{ob}}|_{\mathcal{X}},\, \nu_{\text{set}})$ from ν_{set}.
2: Get the dense collinear scores κ from f.
3: Treat κ as a discrete-domain signal, and low-pass filter it with cut-off frequency ν_{LP}. Overwrite κ with the filtered result.
4: $\phi = J(\eta_i\, f_{\text{ob}}|_{\mathcal{X}},\, \nu_{\text{set}}).* \kappa$.
5: Replace ϕ with the low-pass filtered version of ϕ at cut-off frequency ν_{LP}.
6: Apply the variable low-pass filter from Algorithm 3.2 to ϕ with: radius one, ϕ as the signal to be smoothed, and the cubic-spline interpolated κ as the weight function. Overwrite ϕ with the filtered result.
7: Normalize ϕ so that the minimum is at 0 and the maximum is at 1.
8: Return ϕ.

3.4.4 Results

We now present visual inspections and quantitative evaluations of our strategies for specifying $\phi(\mathcal{X})$. For all strategies, the values of $\phi(x)$, $\forall x \in \mathbb{R}^2 \setminus \mathcal{X}$, were obtained via the cubic-spline interpolation of $\phi(\mathcal{X})$. Our chosen problem for demonstrating the adaptive kernel construction method is the grayscale up-conversion problem. The up-conversion result from cubic-spline interpolation of the observed image f_{ob} is also included for reference. Four strategies are reported:

1. The strategy denoted by $\phi = 0$ is the one where $\phi(x) = 0$, $\forall x \in \mathbb{R}^2$. This is simply the restriction of the canonical kernel on $\mathbb{R}^3$ to the plane where the third coordinate is zero, which is the same type of isotropic kernel but on $\mathbb{R}^2$. We refer this strategy as the *isotropic strategy*, and it serves as a control for the other strategies.

2. The strategy denoted by $\phi \leftarrow \boldsymbol{f}$ is the one where $\phi(\mathcal{X}) = f_{\text{ob}}|_{\mathcal{X}}$. We refer to it as the *interpolated observation strategy* because we interpolate the observed image using an cubic-spline interpolator to get the map ϕ.

3. The strategy denoted by ϕ_{a} is the *adaptive smoothing strategy* from Section 3.4.2 that uses patch-based local analysis, with $\xi = 0.23$, $p = 5$, and a local analysis patch size of of 6×6 pixels with 3 pixel overlap.

4. The strategy denoted by ϕ_{m} is the *monogenic adaptive smoothing strategy* from Section 3.4.3 that uses dense local analysis, with $\xi = 0.23$, $p = 5$, and a local analysis window of of 7×7 pixels. Isotropic $\mathbb{R}^2$ Gaussian low-pass and complementary high-pass filter pairs were used to implement the bandpass filter (i.e., the η_i operator from Section 3.4.3) that operates on discrete signals. The gain at the cut-off frequency of the Gaussian low-pass filter is set to 0.5, and the bandpass band is set to be 0.1 cycles per pixel. The Riesz transform order is set to 14. The set of bandpass frequencies are

$$\nu_{\text{set}} = \{0.1 : 0.1 : 0.5\,\text{cycles/pixel}\}.$$

The sample-without-prefiltering observation model satisfies the underlying assumption of the RKHS regularization framework, but the conventional sample-with-prefiltering observation model used in signal processing does not. For the sample-with-prefiltering model, it is difficult for one to visually distinguish how our strategies for constructing ϕ are different from each other because there is little aliasing in the observed

image under this model. For this reason, results from both observation models are used for the quantitative evaluation, but only the sample-without-prefiltering observation model is used for the visual inspections. Although the additive white Gaussian noise model satisfies the underlying assumption of the RKHS regularization framework, it was not used because none of the up-conversion methods compared here, including cubic-spline interpolation, was designed for that type observation model.

The reference images for each scene were constructed from the full size (3072×2048 pixels) colour Kodak data set that was obtained from the SCIEN website (https://scien.stanford.edu/, retrieved August 2017). It contains 24 images, and we refer to each image from this data set as a *Kodak scene* in this thesis. The Kodak scenes that have an image canvas that is longer in the horizontal direction have a gray border of 16 pixels along the horizontal border, and a gray border of 10 pixels along the vertical border. The Kodak scenes that have an image canvas that is longer in the vertical direction have a gray border of 10 pixels along the horizontal border, and a gray border of 16 pixels along the vertical border. We removed the border accordingly, then converted the result to grayscale by using the *Images.jl* package for the Julia programming language at https://github.com/JuliaImages/Images.jl (retrieved in July 2017). This package computes grayscale representations of colour images by using Rec. 601 coefficients while keeping luma range unchanged; an introduction for colour space conversions and the Rec. 601 standard can be found in [72]. For the quantitative benchmarks, we denote by $\boldsymbol{f}_{\text{kodg}}$ such a grayscale image for a particular Kodak scene. For the visual inspection benchmarks, we use $\boldsymbol{f}_{\text{kodg}}$ again to denote a cropped version of such a grayscale image for a particular Kodak scene. The full image was not used for visual inspections because they do not fit in this document at a large enough magnification for one to make out the details.

For the sample-without-prefiltering observation model, the scheme described in Appendix A was used to construct the reference image $\boldsymbol{f}_{\text{ref}}$ where $\boldsymbol{f}_{\mathcal{A}} = \boldsymbol{f}_{\text{kodg}}$ is used there. The same scheme is used for the sample-with-prefiltering observation model, but $\boldsymbol{f}_{\mathcal{A}}$ is a discrete-domain low-pass filtered version of $\boldsymbol{f}_{\text{kodg}}$. The low-pass filter is a Gaussian filter with a gain of $\frac{1}{3}$ at the Nyquist frequency that corresponds to the downsampling factor; a downsampling factor of c corresponds to a Nyquist frequency of $\frac{1}{2c}$ cycles per pixel unit, where this pixel unit is measured with respect to the length of a pixel on $(\boldsymbol{f}_{\mathcal{A}})_{\mathcal{E}}$. We assume square pixels in this thesis. The downsample factor of 5 is used in the visual inspection benchmarks, while factors 4, 5, and 6 were used for the quantitative benchmarks.

The patch-based RKHS regularization framework from Example 2.19 was used. A maximum RKHS patch size of 20×20 is used, with an overlap of 4 pixels. The isotropic kernel from Example 2.17 on $\mathbb{R}^3$ is used as the canonical kernel, with the hyperparamter $a = 0.048$. These tuning parameters were used for all the scenes for all of the compared warp map construction strategies because we want to see how each strategy fails when the scene content does disagrees too much with our chosen assumption.

3.4.4.1 Visual Inspection of Adapted Kernels

For our discussion here, we write f to mean an image patch. Consider our specification of the set $\mathcal{B}$ from Section 3.2, where $k(\cdot, x_0)$ can be interpreted as an element from $\mathcal{B}$ (see Equation 3.8). Let $\mathcal{X}_{\text{patch}}$ be the pixel locations in $\mathcal{X}$ for the current patch, and $\mathcal{I}$ an ordering for $\mathcal{X}_{\text{patch}}$. A RKHS patch size of 20×20 pixels was used to generate the illustrations here.

Denote by $\boldsymbol{b}_{\mathcal{I}(x)}$ the $\mathcal{I}(x)$th basis vector for $\mathbb{R}^{20\times 20}$, and denote by x the point of interest. Figure 3.6(a) shows the observed image $\left(\boldsymbol{f}_{\text{ob},\,\mathcal{X}}\right)_{\mathcal{E}}$, and the rest of the subfigures illustrate the various $\phi(\mathcal{X}_{\text{patch}})$'s or the various $\boldsymbol{b}_{\mathcal{I}(x)}$'s for the patch that contains x, which is marked with a red star in all the subfigures. The strategy $\phi \leftarrow \boldsymbol{f}$ was used in Figure 3.6(c), and we see that the contours of this strategy's adapted kernel are not smooth along edges in the scene. Therefore, this strategy does not implement our choice of prior assumption on f. The ϕ_{m} strategy gave basis vectors that are smoother along the contours of the edge.

In Figure 3.7, the contrast between the stem and the background is the greatest in the ϕ_{a} and the $\phi \leftarrow \boldsymbol{f}$ strategies (Figures 3.7(c) and (d)), but we shall see later that the contour of the stem for both of these adapted

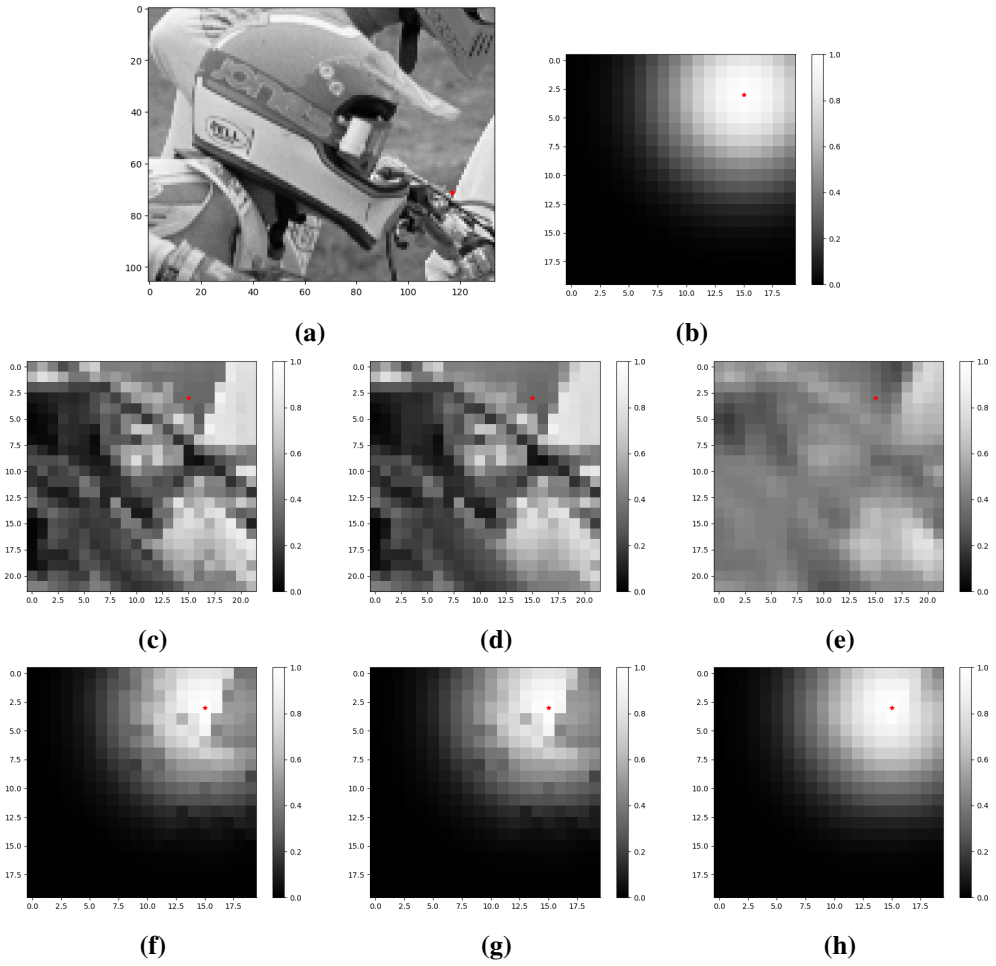


Figure 3.6: A portion of Kodak scene 5: visualization of adapted basis vector $\boldsymbol{b}_{\mathcal{I}(x)}$ and the warp map $\phi(\mathcal{X}_{\text{patch}})$ for the different strategies. (a) The reference image patch that contains x, which is marked by a red star. (b) The image representation of $\phi|_{\mathcal{X}}$ for this patch that corresponds to the isotropic strategy $\phi = 0$. (c) The image representation of $\phi|_{\mathcal{X}}$ for this patch that corresponds to the interpolated observation strategy $\phi \leftarrow \boldsymbol{f}$. (d) The image representation of $\phi|_{\mathcal{X}}$ for this patch that corresponds to the adaptive smoothing strategy ϕ_{a}. (e) The image representation of $\phi|_{\mathcal{X}}$ for this patch that corresponds to the monogenic adaptive strategy ϕ_{m}. (f) The image representation of $\boldsymbol{b}_{\mathcal{I}(x)}$ for this patch that corresponds to the $\phi \leftarrow \boldsymbol{f}$ strategy. (g) The image representation of $\boldsymbol{b}_{\mathcal{I}(x)}$ for this patch that corresponds to the ϕ_{a} strategy. (h) The image representation of $\boldsymbol{b}_{\mathcal{I}(x)}$ for this patch that corresponds to the ϕ_{m} strategy.

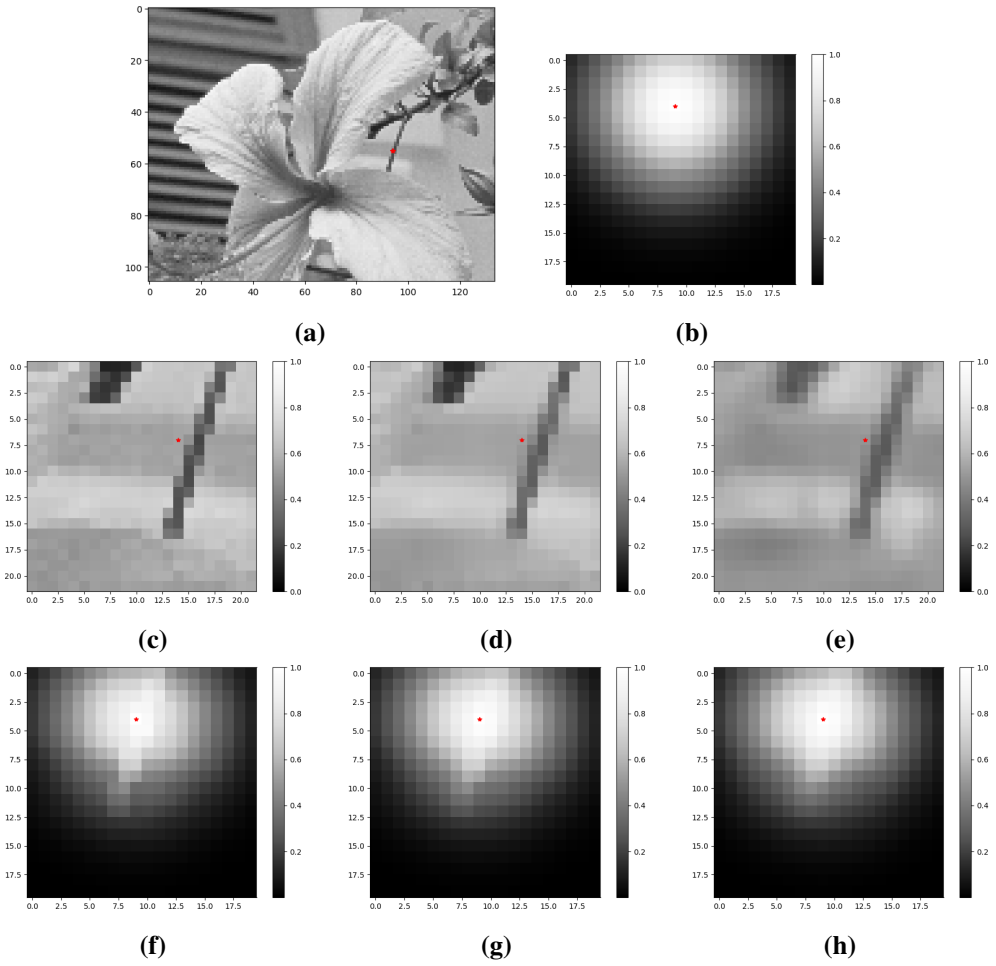


Figure 3.7: A portion of Kodak scene 7: visualization of adapted basis vector $\boldsymbol{b}_{\mathcal{I}(x)}$ and the warp map $\phi(\mathcal{X}_{\text{patch}})$ for the different strategies. (a) The reference image patch that contains x, which is marked by a red star. (b) The image representation of $\phi|_{\mathcal{X}}$ for this patch that corresponds to the isotropic strategy $\phi = 0$. (c) The image representation of $\phi|_{\mathcal{X}}$ for this patch that corresponds to the interpolated observation strategy $\phi \leftarrow \boldsymbol{f}$. (d) The image representation of $\phi|_{\mathcal{X}}$ for this patch that corresponds to the adaptive smoothing strategy ϕ_{a}. (e) The image representation of $\phi|_{\mathcal{X}}$ for this patch that corresponds to the monogenic adaptive strategy ϕ_{m}. (f) The image representation of $\boldsymbol{b}_{\mathcal{I}(x)}$ for this patch that corresponds to the $\phi \leftarrow \boldsymbol{f}$ strategy. (g) The image representation of $\boldsymbol{b}_{\mathcal{I}(x)}$ for this patch that corresponds to the ϕ_{a} strategy. (h) The image representation of $\boldsymbol{b}_{\mathcal{I}(x)}$ for this patch that corresponds to the ϕ_{m} strategy.

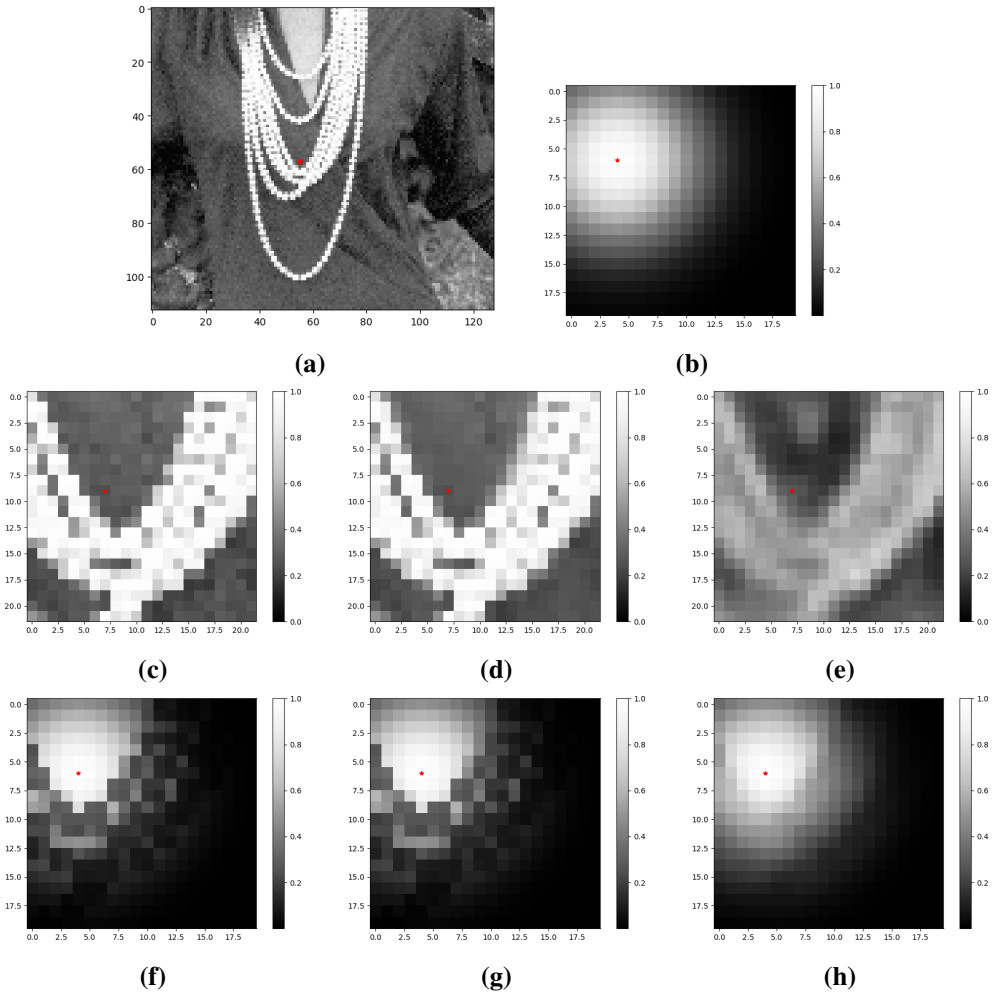


Figure 3.8: A portion of Kodak scene 18: visualization of adapted basis vector $\boldsymbol{b}_{\mathcal{I}(x)}$ and the warp map $\phi(\mathcal{X}_{\text{patch}})$ for the different strategies. (a) The reference image patch that contains x, which is marked by a red star. (b) The image representation of $\phi|_{\mathcal{X}}$ for this patch that corresponds to the isotropic strategy $\phi = 0$. (c) The image representation of $\phi|_{\mathcal{X}}$ for this patch that corresponds to the interpolated observation strategy $\phi \leftarrow \boldsymbol{f}$. (d) The image representation of $\phi|_{\mathcal{X}}$ for this patch that corresponds to the adaptive smoothing strategy ϕ_{a}. (e) The image representation of $\phi|_{\mathcal{X}}$ for this patch that corresponds to the monogenic adaptive strategy ϕ_{m}. (f) The image representation of $\boldsymbol{b}_{\mathcal{I}(x)}$ for this patch that corresponds to the $\phi \leftarrow \boldsymbol{f}$ strategy. (g) The image representation of $\boldsymbol{b}_{\mathcal{I}(x)}$ for this patch that corresponds to the ϕ_{a} strategy. (h) The image representation of $\boldsymbol{b}_{\mathcal{I}(x)}$ for this patch that corresponds to the ϕ_{m} strategy.

kernels are not smooth enough to induce a visual sensation of a stem with smooth edges (Figures 3.7(e) and (f)). Our strategies do not perform well when there is insufficient information in $f_{\text{ob}}|_{\mathcal{X}}$ to characterize the pattern in the image. An example is in Figure 3.8, where the necklace in Kodak scene 18 is subjected to a downsampling factor of 5. The ϕ_{m} strategy applied too much smoothing to the beads' outline, to the point that the basis induced by the adapted kernel does not resemble the contours of the beads.

3.4.4.2 Visual Inspection of Up-Converted Images

Here, we show the up-conversion results of some of the scenes from the data set. These scenes were chosen to showcase scenarios when our prior assumption is mostly valid (scenes 5, 7, and 20), sometimes valid (scenes 9), and invalid (scene 4, 13, and 18). In Figure 3.9, we see that the ϕ_{a} and ϕ_{m} strategies removed most of the aliasing artefacts along high-contrast edges that is marked by the red circle. The detailed portion of the scene that is marked by the green circle were not affected by our strategies. In Figure 3.10, we see that both ϕ_a and ϕ_m removed most of the aliasing on the high-contrast edges of the propeller blade that is shown. In Figure 3.11, the thin vertical stem that is circled in red is smoothest in the up-converted result that uses the ϕ_m strategy, but it has some minor banding artefacts near the border of the bright flower pedals that is circled in green. In Figure 3.12, we have texture patterns that were sufficiently captured in the observed image $f_{\text{ob}}|_{\mathcal{X}}$ for our strategies to infer its edge contours correctly.

Figures 3.13, 3.15, and 3.17 illustrate scenarios when our strategies begin to fail. The rigging lines of the sail and shadows on the sail fabric in Figure 3.13 are details that were not sufficiently captured in $f_{\text{ob}}|_{\mathcal{X}}$, since some lines and shadows are emphasized more than others. This creates an unnatural look on the up-converted result. The scene in Figure 3.15 has a lot of detail, but the sampling period of $\mathcal{X}$ was not short enough for our strategies to obtain enough information to recreate the edges accurately. As a result, the ϕ_{m} strategy produced adapted basis functions from what it thought were edges, and led to an up-converted result where the rocks look like strips. A similar scenario is present in Figure 3.17, where certain beads on the necklace are too close together such that $f_{\text{ob}}|_{\mathcal{X}}$ was unable to capture enough details. To make matters worse, the high-contrast between the beads and the background clothing prompted the ϕ_{a} and ϕ_{m} strategies to over-emphasize the contour of the beads.

These problems were reduced when the observed image was generated using a downsample factor of 4 instead of 5. Figure 3.14 shows that the rigging lines have improved over the lines in Figure 3.13. There are still a lot of artefacts in Figure 3.16. We could have tried scene-specific tuning for parameters such as ξ to improve the performance, but we did not pursue this in this thesis. Figure 3.18 shows that our strategies produced a more realistic-looking up-converted result when the downsample factor was set to 4 instead of 5.

3.4.4.3 Similarity Scores

Here, we present a quantitative evaluation of the up-conversion results from our strategies. We measure the image similarity between the reference image $\boldsymbol{f}_{\text{ref}}$ and the up-converted result $\widetilde{\boldsymbol{f}}_{\mathcal{X}_q}$. The similarity measure used here is the information-weighted structural similarity information measure (IW-SSIM) [73], and we used the implementation from its author's website [74]. The IW-SSIM score is the most recent extension of the popular structural similarity information measure score that is widely used across the image processing literature. This extension is such that patterns of different magnifications from the scene all contribute in a perceptually-uniform manner to the measurement score [73].

We see from Table 3.1 that the scenes with more high-contrast edges have a higher IW-SSIM score relative to its score from cubic-spline interpolation. I believe the superiority of the ϕ_{m} strategy over the ϕ_{a} strategy in many scenes can be explained by its use of multiple frequency bandpass filters. However, the ϕ_{m} strategy failed for some scenes. One example is scene 18, which contains the necklace from Figure 3.17. This means a more robust method to decrease adaptation is required in high-contrast areas that have

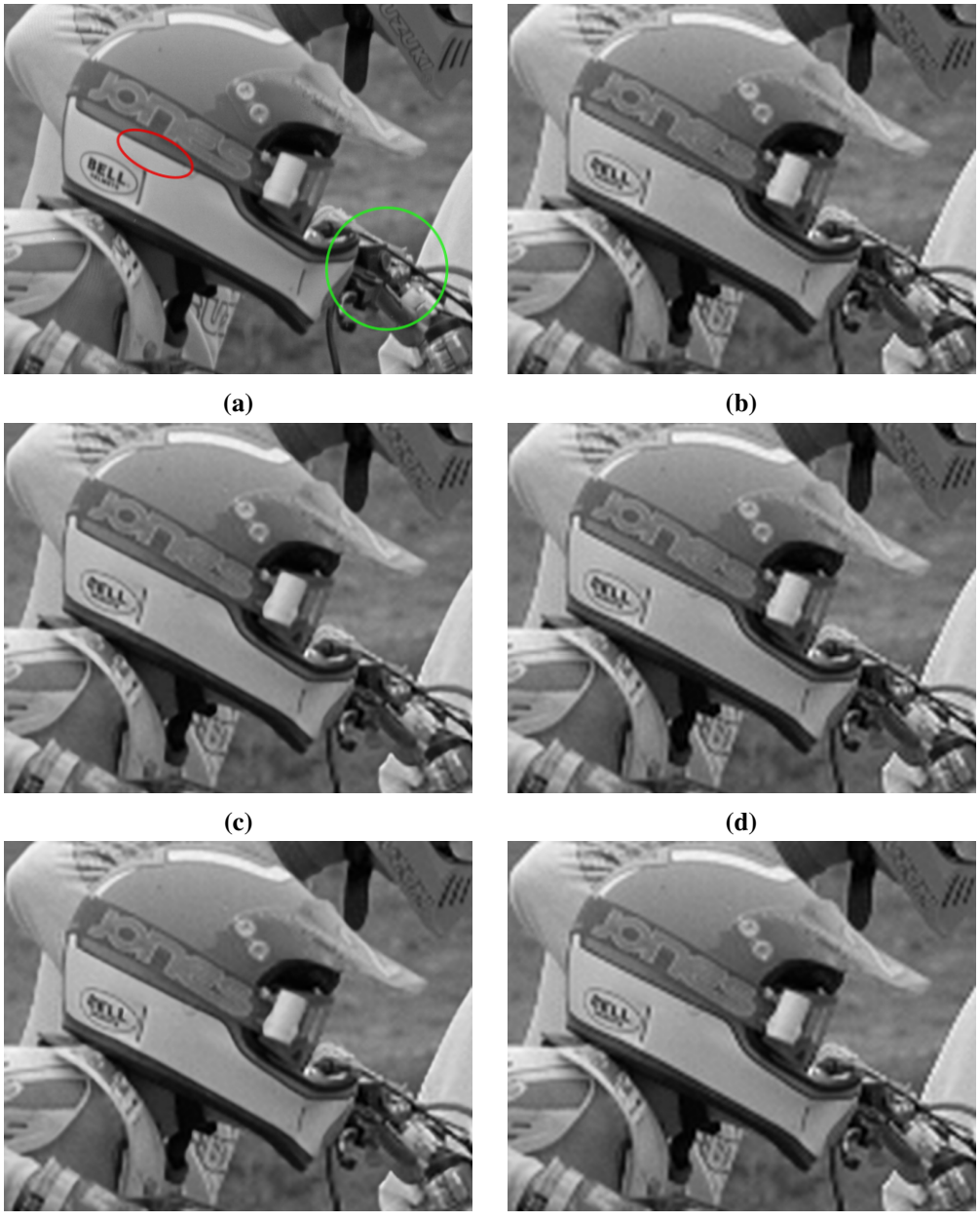
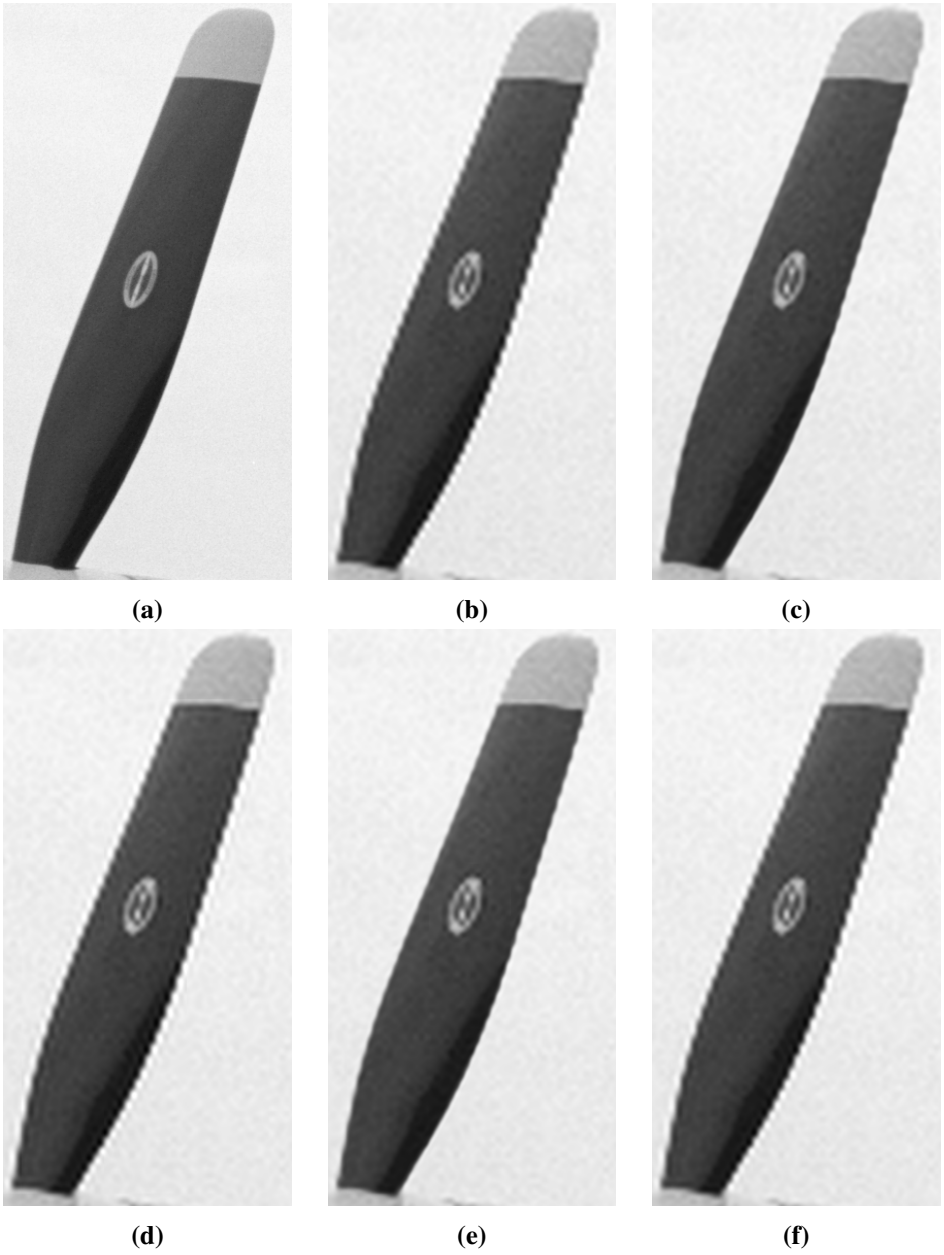


Figure 3.9: A portion of Kodak scene 5. The observed image $f_{\text{ob}}|_{\mathcal{X}}$ was generated by downsample-without-prefiltering by a factor of 5 in both spatial directions, and the up-conversion is by a factor of 5 in both spatial directions. (a) Reference image with circled regions of interest, (b) cubic-spline up-converted result, (c) the up-converted result from the monogenic adaptive smoothing strategy ϕ_{m}, (d) the up-converted result from the isotropic strategy $\phi = 0$, (e) the up-converted result from the adaptive smoothing strategy ϕ_a, (f) the up-converted result from the interpolated observation strategy $\phi \leftarrow \boldsymbol{f}$.



(a) **(b)** **(c)**

(d) **(e)** **(f)**

Figure 3.10: A portion of Kodak scene 20. The observed image $f_{\mathrm{ob}}|_{\mathcal{X}}$ was generated by downsample-without-prefiltering by a factor of 5 in both spatial directions, and the up-conversion is by a factor of 5 in both spatial directions. (a) Reference image, (b) cubic-spline up-converted result, (c) the up-converted result from the monogenic adaptive smoothing strategy ϕ_{m}, (d) the up-converted result from the isotropic strategy $\phi = 0$, (e) the up-converted result from the adaptive smoothing strategy ϕ_a, (f) the up-converted result from the interpolated observation strategy $\phi \leftarrow \boldsymbol{f}$.

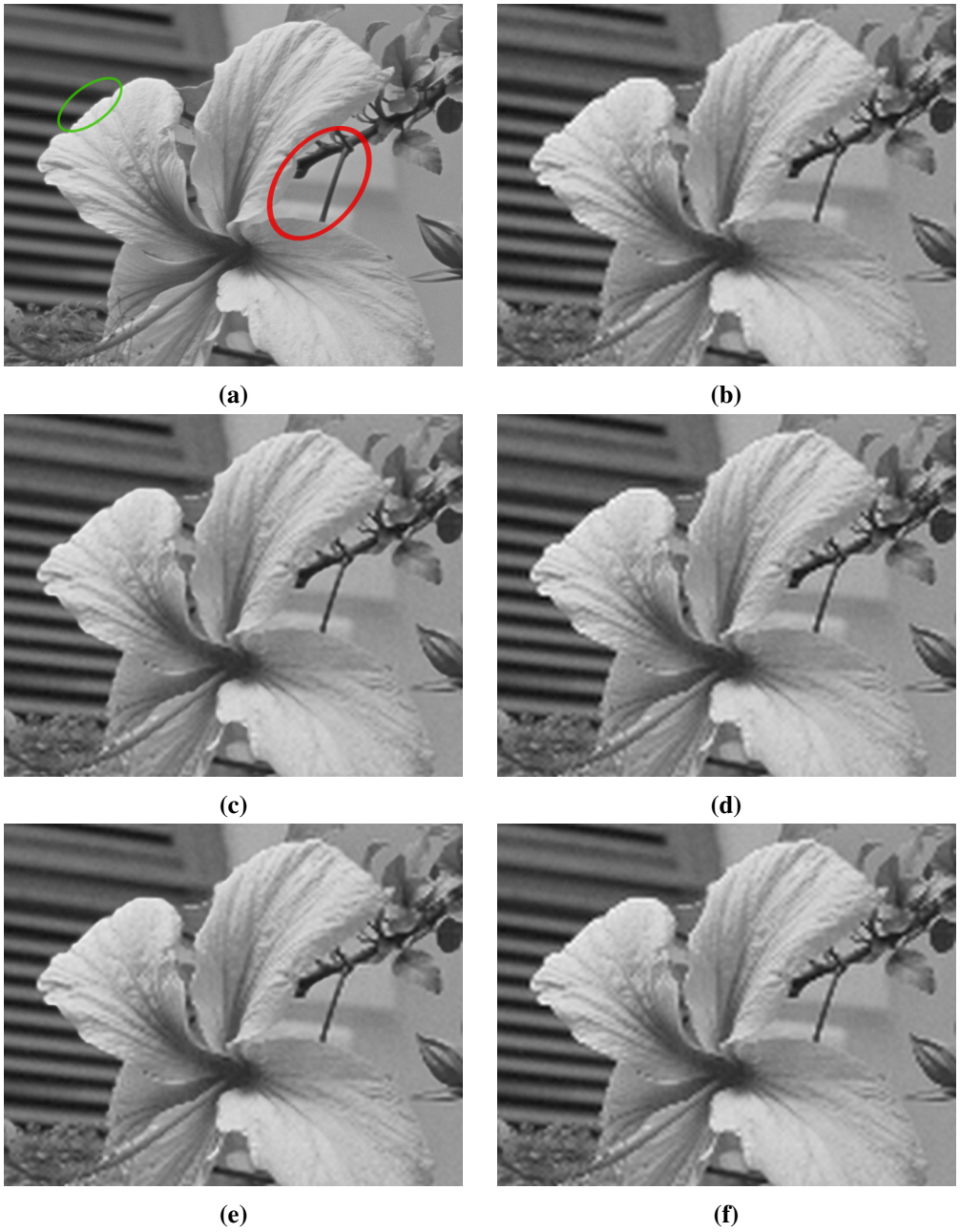


Figure 3.11: A portion of Kodak scene 7. The observed image $f_{\text{ob}}|_{\mathcal{X}}$ was generated by downsample-without-prefiltering by a factor of 5 in both spatial directions, and the up-conversion is by a factor of 5 in both spatial directions. (a) Reference image with circled regions of interest, (b) cubic-spline up-converted result, (c) the up-converted result from the monogenic adaptive smoothing strategy ϕ_{m}, (d) the up-converted result from the isotropic strategy $\phi = 0$, (e) the up-converted result from the adaptive smoothing strategy ϕ_a, (f) the up-converted result from the interpolated observation strategy $\phi \leftarrow \boldsymbol{f}$.

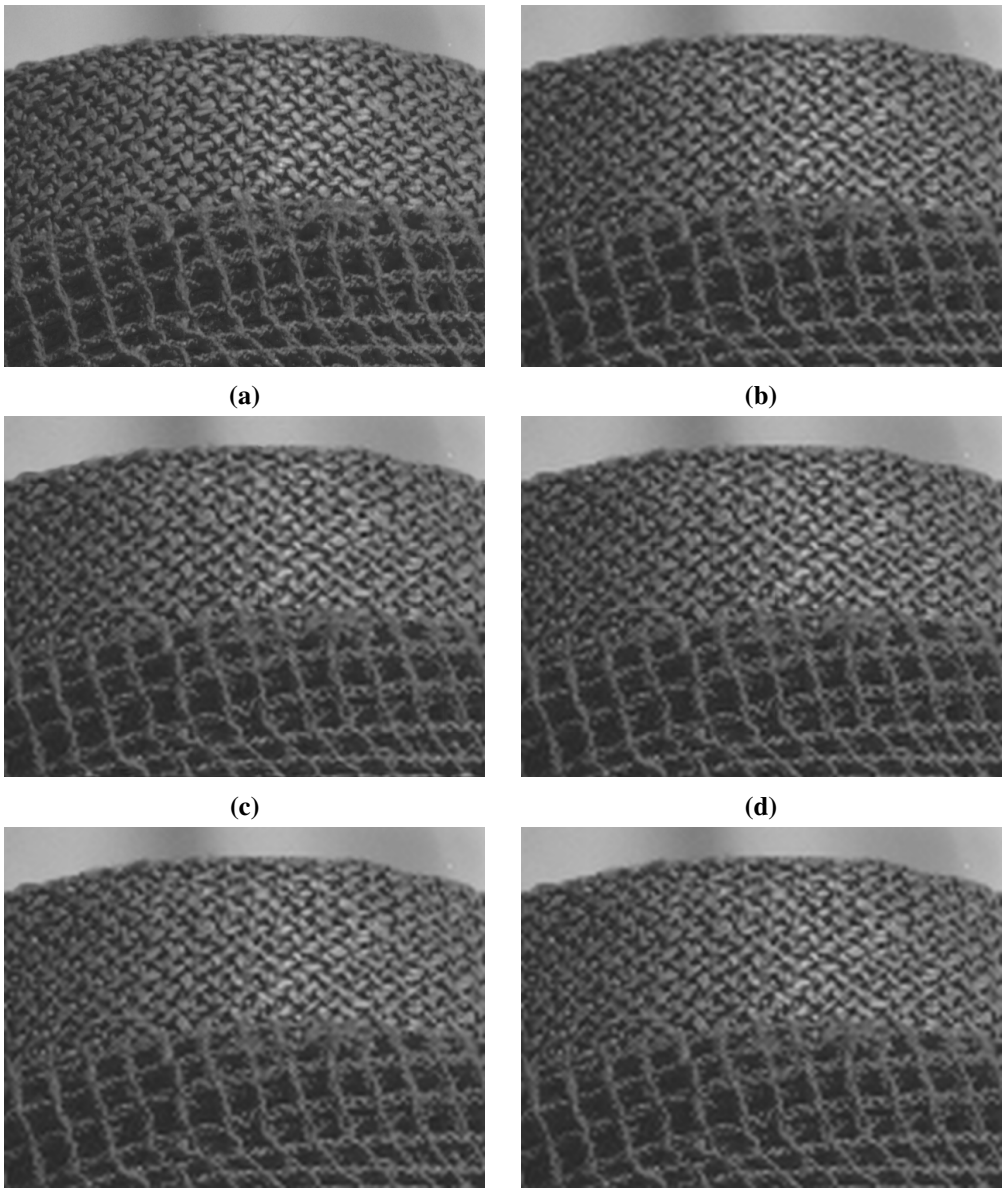
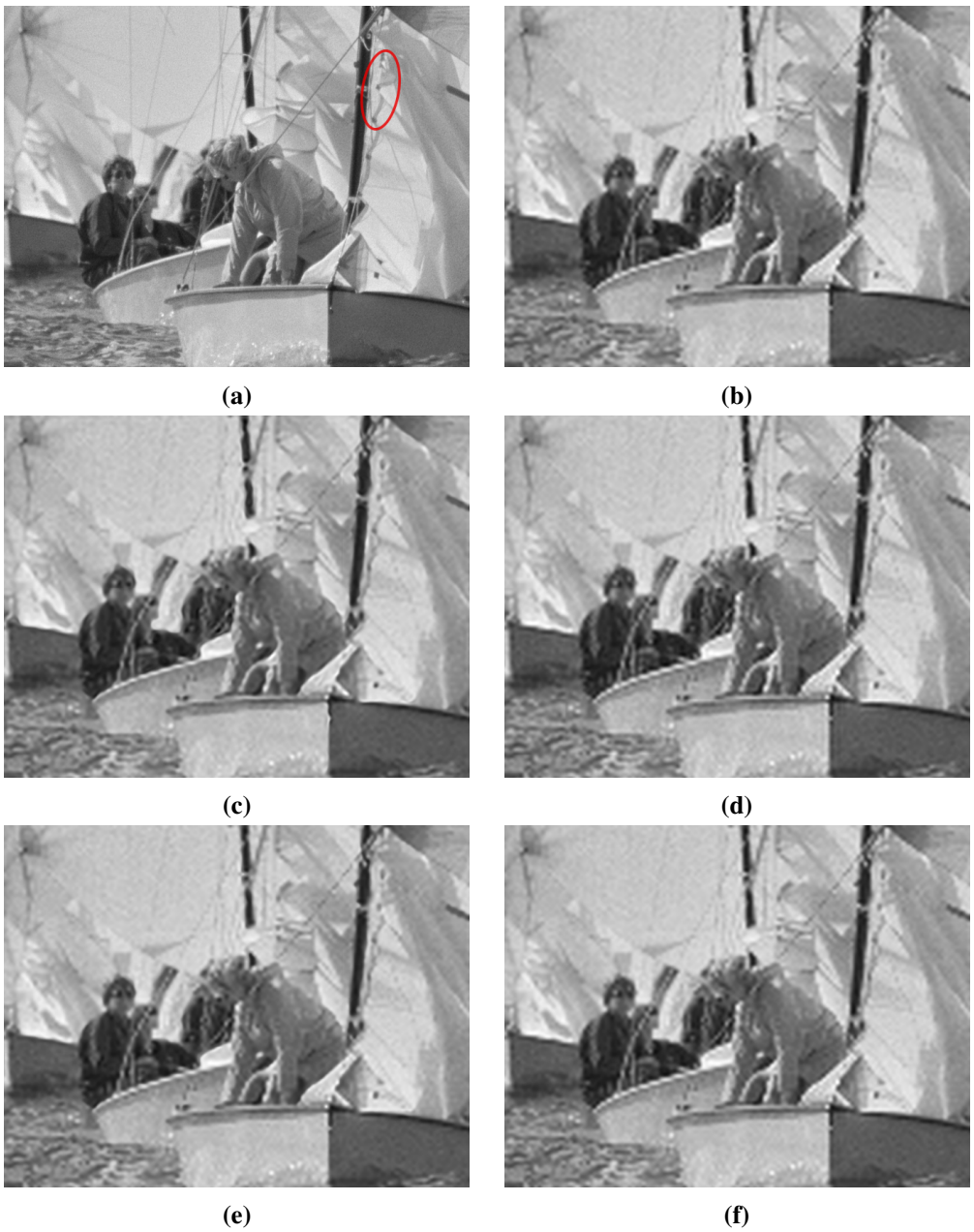
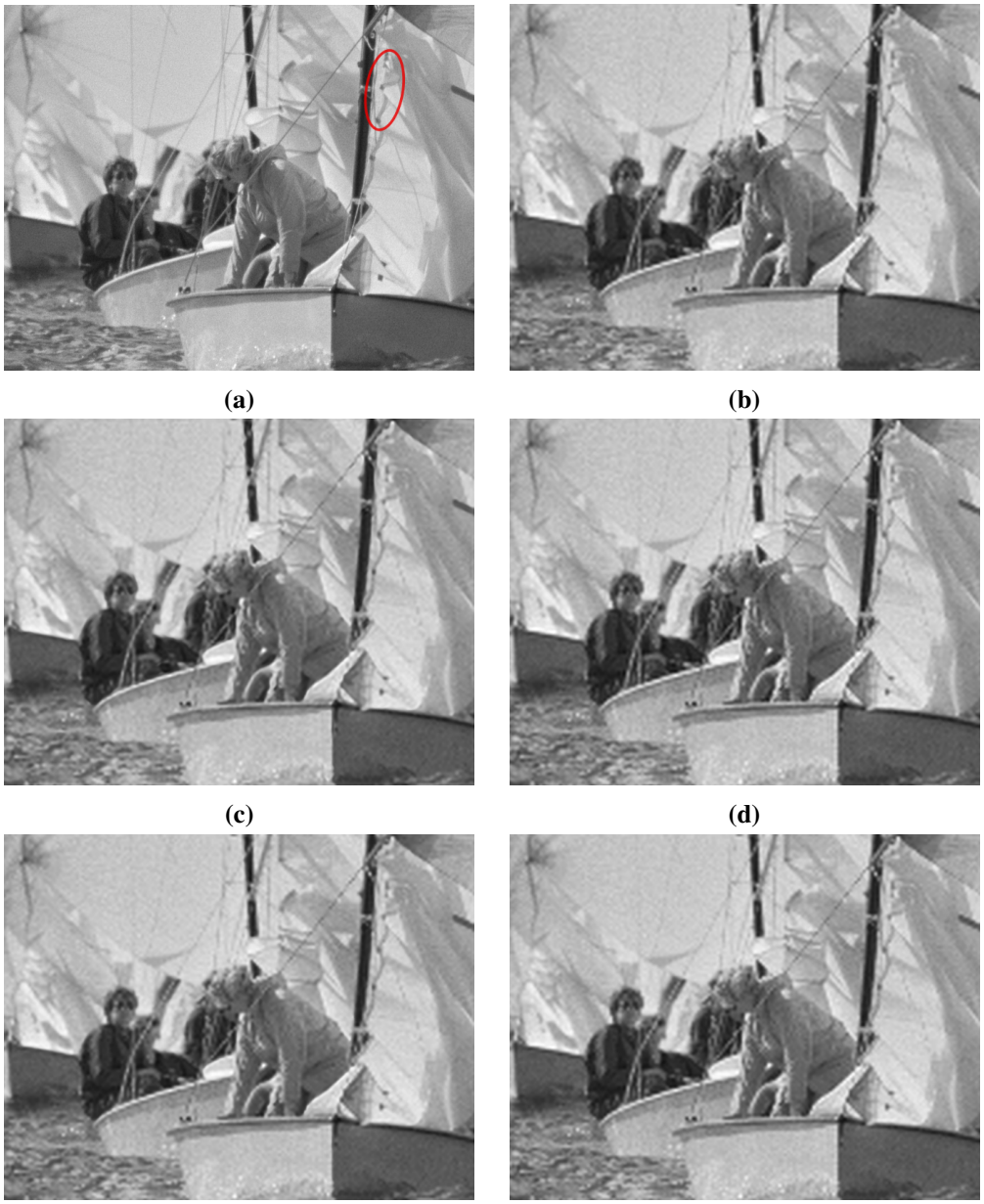


Figure 3.12: A portion of Kodak scene 4. The observed image $f_{\text{ob}}|_{\mathcal{X}}$ was generated by downsample-without-prefiltering by a factor of 5 in both spatial directions, and the up-conversion is by a factor of 5 in both spatial directions. (a) Reference image, (b) cubic-spline up-converted result, (c) the up-converted result from the monogenic adaptive smoothing strategy ϕ_{m}, (d) the up-converted result from the isotropic strategy $\phi = 0$, (e) the up-converted result from the adaptive smoothing strategy ϕ_a, (f) the up-converted result from the interpolated observation strategy $\phi \leftarrow \boldsymbol{f}$.



(a) **(b)** **(c)** **(d)** **(e)** **(f)**

Figure 3.13: A portion of Kodak scene 9. The observed image $f_{\text{ob}}|_{\mathcal{X}}$ was generated by downsample-without-prefiltering by a factor of 5 in both spatial directions, and the up-conversion is by a factor of 5 in both spatial directions. (a) Reference image with circled region of interest, (b) cubic-spline up-converted result, (c) the up-converted result from the monogenic adaptive smoothing strategy ϕ_{m}, (d) the up-converted result from the isotropic strategy $\phi = 0$, (e) the up-converted result from the adaptive smoothing strategy ϕ_a, (f) the up-converted result from the interpolated observation strategy $\phi \leftarrow \boldsymbol{f}$.



(a) **(b)** **(c)** **(d)** **(e)** **(f)**

Figure 3.14: A portion of Kodak scene 9. The observed image $f_{\text{ob}}|_{\mathcal{X}}$ was generated by downsample-without-prefiltering by a factor of 4 in both spatial directions, and the up-conversion is by a factor of 4 in both spatial directions. (a) Reference image with circled region of interest, (b) cubic-spline up-converted result, (c) the up-converted result from the monogenic adaptive smoothing strategy ϕ_{m}, (d) the up-converted result from the isotropic strategy $\phi = 0$, (e) the up-converted result from the adaptive smoothing strategy ϕ_a, (f) the up-converted result from the interpolated observation strategy $\phi \leftarrow \boldsymbol{f}$.

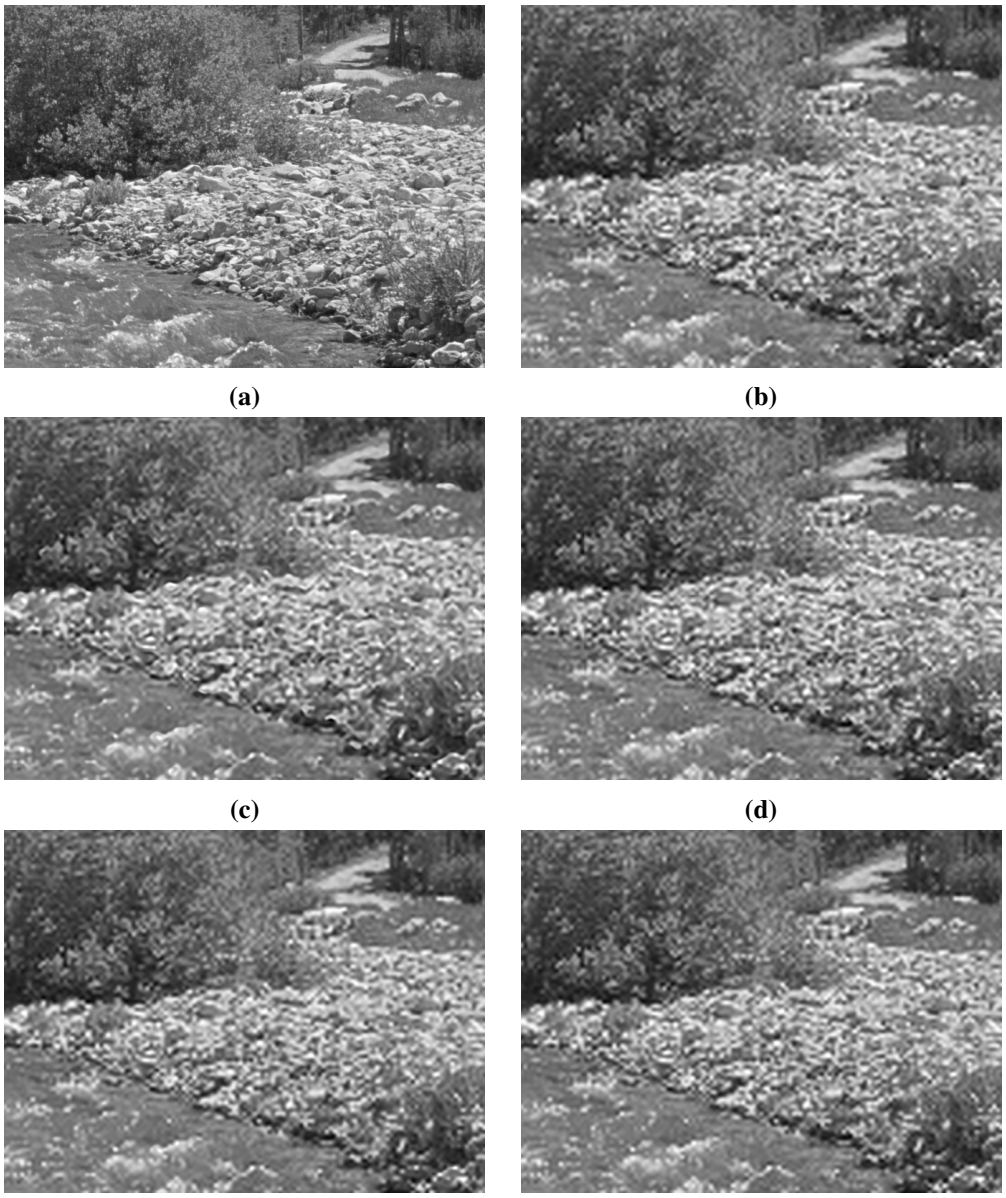


Figure 3.15: A portion of Kodak scene 13. The observed image $f_{\mathrm{ob}}|_{\mathcal{X}}$ was generated by downsample-without-prefiltering by a factor of 5 in both spatial directions, and the up-conversion is by a factor of 5 in both spatial directions. (a) Reference image, (b) cubic-spline up-converted result, (c) the up-converted result from the monogenic adaptive smoothing strategy ϕ_{m}, (d) the up-converted result from the isotropic strategy $\phi = 0$, (e) the up-converted result from the adaptive smoothing strategy ϕ_a, (f) the up-converted result from the interpolated observation strategy $\phi \leftarrow \boldsymbol{f}$.

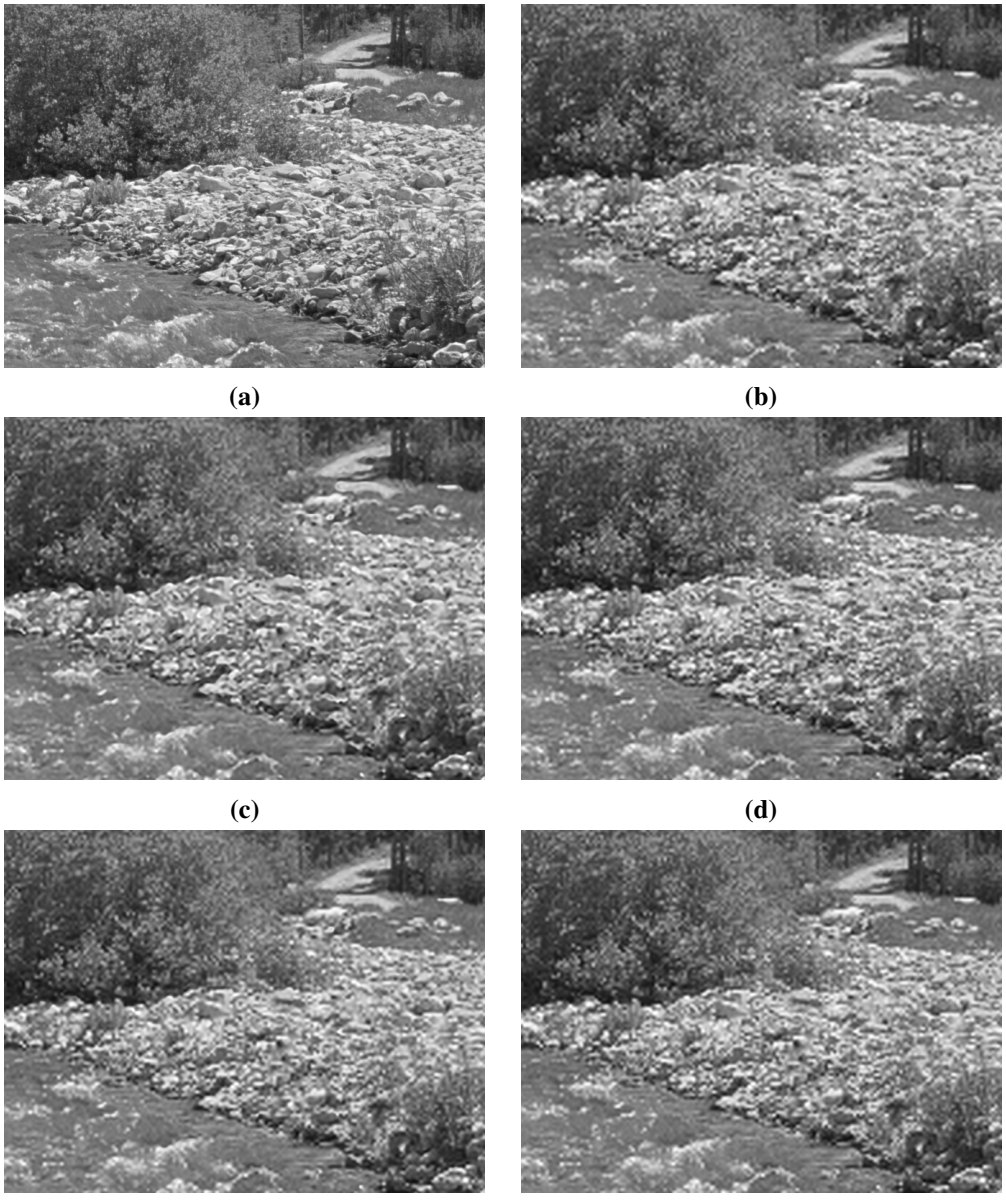


Figure 3.16: A portion of Kodak scene 13. The observed image $f_{\mathrm{ob}}|_{\mathcal{X}}$ was generated by downsample-without-prefiltering by a factor of 4 in both spatial directions, and the up-conversion is by a factor of 4 in both spatial directions. (a) Reference image, (b) cubic-spline up-converted result, (c) the up-converted result from the monogenic adaptive smoothing strategy ϕ_{m}, (d) the up-converted result from the isotropic strategy $\phi = 0$, (e) the up-converted result from the adaptive smoothing strategy ϕ_a, (f) the up-converted result from the interpolated observation strategy $\phi \leftarrow \boldsymbol{f}$.

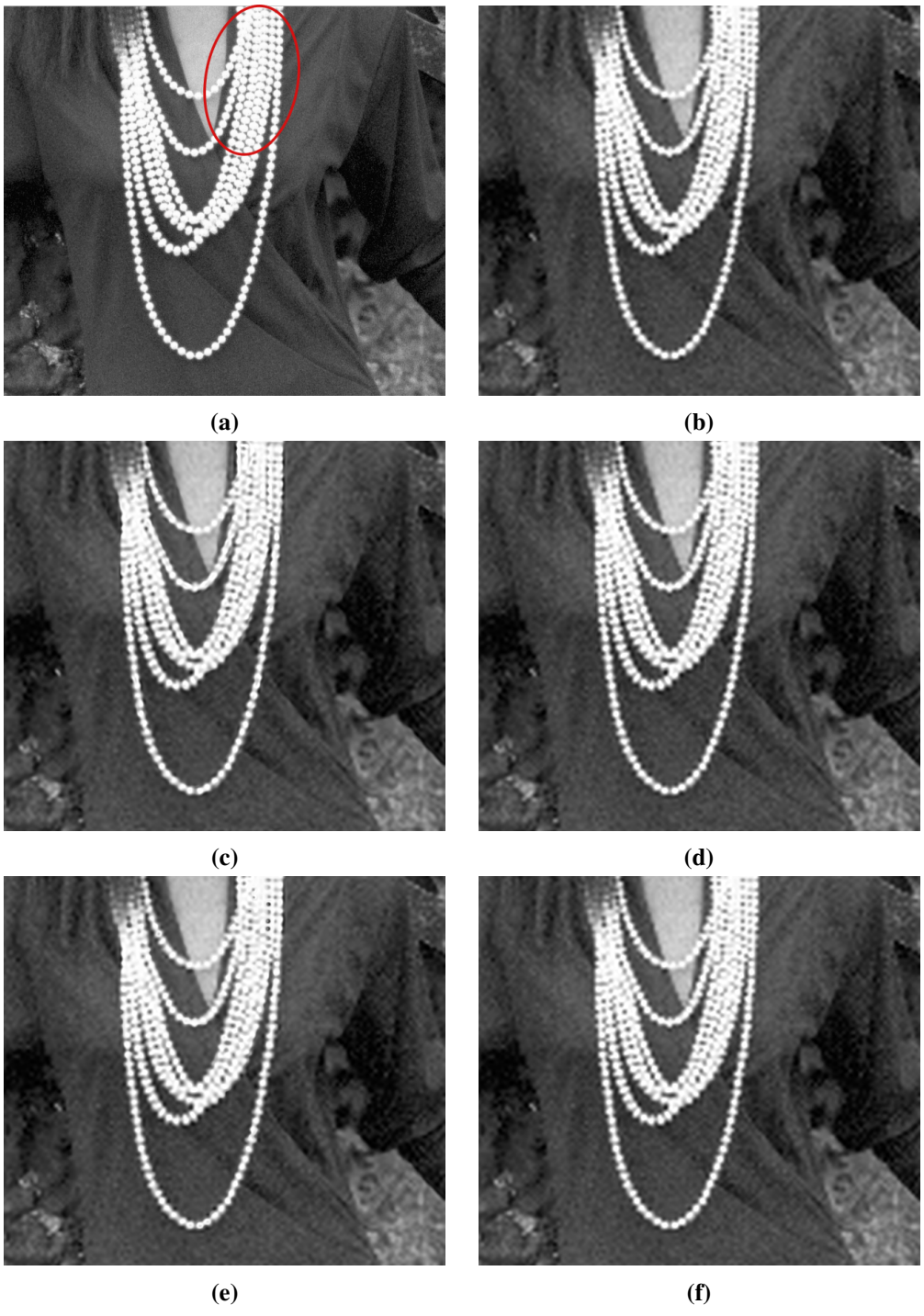


Figure 3.17: A portion of Kodak scene 18. The observed image $f_{\text{ob}}|_{\mathcal{X}}$ was generated by downsample-without-prefiltering by a factor of 5 in both spatial directions, and the up-conversion is by a factor of 5 in both spatial directions. (a) Reference image with circled region of interest, (b) cubic-spline up-converted result, (c) the up-converted result from the monogenic adaptive smoothing strategy ϕ_{m}, (d) the up-converted result from the isotropic strategy $\phi = 0$, (e) the up-converted result from the adaptive smoothing strategy ϕ_a, (f) the up-converted result from the interpolated observation strategy $\phi \leftarrow \boldsymbol{f}$.

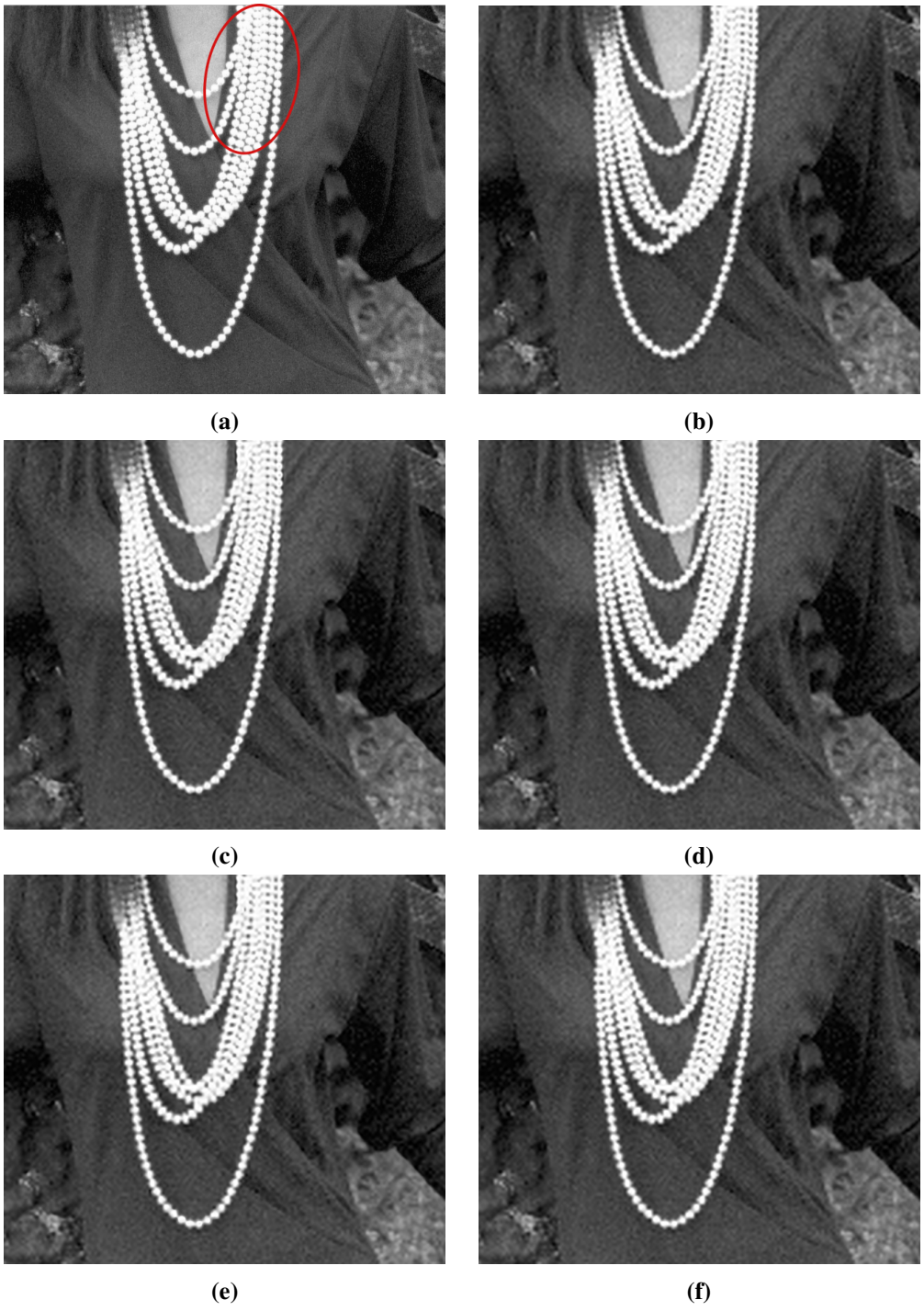


Figure 3.18: A portion of Kodak scene 18. The observed image $f_{\text{ob}}|_{\mathcal{X}}$ was generated by downsample-without-prefiltering by a factor of 4 in both spatial directions, and the up-conversion is by a factor of 4 in both spatial directions. (a) Reference image with circled region of interest, (b) cubic-spline up-converted result, (c) the up-converted result from the monogenic adaptive smoothing strategy ϕ_{m}, (d) the up-converted result from the isotropic strategy $\phi = 0$, (e) the up-converted result from the adaptive smoothing strategy ϕ_a, (f) the up-converted result from the interpolated observation strategy $\phi \leftarrow \boldsymbol{f}$.

insufficient information.

Our strategies have a lower IW-SSIM score relative to its score from cubic-spline interpolation when the downsample factor is lowered to 4. When the downsample factor was increased from 5 to 6, the IW-SSIM scores were more favourable for the ϕ_{m} strategy. I believe this is because our chosen prior assumption on f is more helpful when there are more aliasing artefacts in the observation.

From Tables 3.1, 3.2, and 3.3, we see that the RKHS regularized solution of the $\phi \leftarrow \boldsymbol{f}$ and $\phi = 0$ strategies yielded similar IW-SSIM scores to the cubic-spline up-conversion method for most of the scenes. For many of these scenes, it was difficult for me to visually pick out any differences between these three methods.

In Table 3.4, the $\phi = 0$ strategy has higher IW-SSIM scores than its score from cubic-spline interpolation for all the Kodak scenes. This trend continues in Table 3.5, where the downsample factor is lowered to 4 from 5. In Table 3.6, we see that the ϕ_{m} strategy performed better when the downsample factor was increased to 6 from 5.

Tables 3.4, 3.5, and 3.6 show a trend where the $\phi = 0$ is preferred, unless a high downsample factor is used. The sample-with-prefiltering observation model was used to generate these three tables, and the prefilter used was an isotropic Gaussian low-pass filter. The cubic-spline interpolation used is a separable interpolator. I believe the superior performance of the $\phi = 0$ over the cubic-spline interpolator might be related to the fact that an isotropic kernel is used for an isotropic low-pass filtered observation. Even with the sample-with-prefiltering model, we continue to see the trend that the ϕ_{m} strategy is more suitable for scenarios with high downsample factors.

3.4.4.4 Discussion

We saw in Section 3.4.4.1 the adaptive kernel modified isotropic basis vectors to non-isotropic basis vectors for $\mathbb{R}^N$, and that different strategies for specifying ϕ gave rise to different non-isotropic basis vectors for a patch of $\boldsymbol{f}_{\mathrm{ob},\,\mathcal{X}}$. These basis vectors in component-form (with respect to the elementary basis) are the columns of the kernel matrix of k on the inputs in $\mathcal{X}$. The induced set of $|\mathcal{X}|$ linearly independent vectors for the signal space of $\left(\tilde{\boldsymbol{f}}_{\mathcal{X}_q}\right)_{\mathcal{E}} \in \mathbb{R}^M$ is

$$\boldsymbol{\mathcal{B}}_{\mathcal{X}_q} := \{k(x_q,\, x)\}_{x \in \mathcal{X},\, x_q \in \mathcal{X}_q},$$

which is the evaluation of the adapted kernel at the observed inputs and the queried inputs. If we take the sparse recovery interpretation of the RKHS regularization framework with $f = f_{\mathrm{ob}}$ that was discussed in Section 3.2, then we have $\boldsymbol{\mathcal{U}} \leftarrow \boldsymbol{\mathcal{B}}_{\mathcal{X}_q}$ and $f \leftarrow f_{\mathrm{ob}}$ if we use the notation defined there. Since $M \geq |\mathcal{X}|$ in the signal up-conversion problem, $\boldsymbol{\mathcal{B}}_{\mathcal{X}_q}$ is a basis for a $|\mathcal{X}|$-dimensional subspace of $\mathbb{R}^M$.

When $\mathcal{X}$ is on a grid, it is hard for one to directly specify a value for ϕ at points that are not in $\mathcal{X}$ because most discrete-domain signal processing methods only operate on a grid. This requires the use of an interpolator so that we can obtain $\phi(x)$ from $\phi(\mathcal{X})$, $\forall x \in \mathbb{R}^{D_0} \setminus \mathcal{X}$. An interpretation of the adaptive RKHS regularization framework is that we are doing RKHS regularization with the canonical kernel s where the input space is $\mathbb{R}^D$ instead of $\mathbb{R}^{D_0}$. When we use an interpolator to specify ϕ in its entirety, the pair of query points that we input to the canonical kernel s each has its last $D - D_0$ coordinates interpolated from $\phi(\mathcal{X})$. Our demonstration shows that this interpolator need not be sophisticated for the result of the RKHS regularization to make a difference. I believe this interpretation of the adaptive kernel should be reviewed when one applies this framework to a real-life application, because it forces one to address the question of whether the specified $\phi(\mathcal{X})$ and the interpolator that constructs ϕ from $\phi(\mathcal{X})$ yields sufficient improvement over a low-complexity up-conversion scheme for the application at hand. From Tables 3.4, 3.5, and 3.6, we see that $\phi(\mathcal{X})$ is better left to take on a value of zero everywhere when the application at hand is to up-convert grayscale images that are prefiltered and downsampled by $\lessapprox 5$ pixels in each dimension.

Scene	cubic-spline	$\phi = 0$	$\phi \leftarrow \boldsymbol{f}$	ϕ_{a}	ϕ_{m}
K01	0.85600	0.85175	0.85611	0.85525	**0.85728**
K02	0.84661	0.84185	0.84652	0.84585	**0.84691**
K03	0.91334	0.91041	0.91336	0.91361	**0.91421**
K04	**0.88938**	0.88595	0.88936	0.88920	0.88928
K05	0.94076	0.93960	0.94092	0.94392	**0.94774**
K06	0.86203	0.85761	0.86209	0.86173	**0.86306**
K07	0.92913	0.92636	0.92921	0.93024	**0.93143**
K08	0.80842	0.80267	0.80842	0.80698	**0.80956**
K09	**0.78972**	0.78314	0.78969	0.78941	0.78968
K10	0.82892	0.82329	0.82889	0.82845	**0.82917**
K11	0.86573	0.86095	0.86574	0.86566	**0.86771**
K12	0.91175	0.90840	0.91179	0.91232	**0.91255**
K13	0.81912	0.81459	0.81910	0.81881	**0.81947**
K14	0.90390	0.90095	0.90386	0.90451	**0.90714**
K15	**0.89451**	0.89083	0.89444	0.89412	0.89392
K16	0.84248	0.83750	0.84250	0.84202	**0.84266**
K17	0.83770	0.83183	0.83765	0.83757	**0.83797**
K18	**0.81158**	0.80667	0.81151	0.81101	0.81093
K19	0.86038	0.85580	0.86044	0.86166	**0.86274**
K20	0.86879	0.86436	0.86884	**0.87043**	0.87028
K21	0.83600	0.83107	0.83601	0.83575	**0.83617**
K22	**0.88603**	0.88221	0.88602	0.88548	0.88544
K23	0.92086	0.91793	0.92085	**0.92092**	0.92077
K24	0.91915	0.91706	0.91932	0.91903	**0.91987**

Table 3.1: IW-SSIM scores with respect to $\boldsymbol{f}_{\mathrm{ref}}$ under a downsample-without-prefiltering model, where 1 out of every 5 samples of $(\boldsymbol{f}_{\mathrm{ref}})_{\mathcal{E}}$ is retained (per dimension) to form $\left(\boldsymbol{f}_{\mathrm{ob},\mathcal{X}}\right)_{\mathcal{E}}$. Bold is the highest score for a scene.

Scene	cubic-spline	$\phi = 0$	$\phi \leftarrow \boldsymbol{f}$	ϕ_{a}	ϕ_{m}
K01	0.92162	0.91792	0.92171	0.92126	**0.92207**
K02	**0.90175**	0.89875	0.90169	0.90120	0.90172
K03	0.94579	0.94412	0.94579	0.94589	**0.94613**
K04	**0.93030**	0.92830	0.93029	0.93012	0.93005
K05	0.97127	0.97091	0.97135	0.97272	**0.97448**
K06	0.92242	0.92024	0.92246	0.92222	**0.92315**
K07	0.95188	0.95002	0.95190	0.95239	**0.95259**
K08	0.87209	0.86803	0.87210	0.87107	**0.87321**
K09	**0.82659**	0.82122	0.82655	0.82628	0.82610
K10	**0.86141**	0.85671	0.86137	0.86116	0.86122
K11	0.92191	0.91935	0.92192	0.92193	**0.92314**
K12	0.94910	0.94744	0.94913	**0.94938**	0.94937
K13	0.89290	0.89033	0.89288	0.89247	**0.89309**
K14	0.94492	0.94347	0.94487	0.94534	**0.94678**
K15	**0.93997**	0.93860	0.93991	0.93992	0.93979
K16	0.90158	0.89877	0.90158	0.90131	**0.90165**
K17	**0.87169**	0.86701	0.87163	0.87137	0.87146
K18	**0.85515**	0.85078	0.85506	0.85478	0.85455
K19	0.91475	0.91180	0.91483	0.91533	**0.91619**
K20	0.91589	0.91302	0.91594	**0.91649**	0.91636
K21	0.89106	0.88779	**0.89108**	0.89082	0.89097
K22	**0.93140**	0.92921	**0.93140**	0.93110	0.93111
K23	0.94830	0.94653	0.94829	**0.94832**	0.94810
K24	0.96210	0.96107	0.96220	0.96211	**0.96267**

Table 3.2: IW-SSIM scores with respect to $\boldsymbol{f}_{\mathrm{ref}}$ under a downsample-without-prefiltering model, where 1 out of every 4 samples of $(\boldsymbol{f}_{\mathrm{ref}})_{\mathcal{E}}$ is retained (per dimension) to form $\left(\boldsymbol{f}_{\mathrm{ob},\mathcal{X}}\right)_{\mathcal{E}}$. Bold is the highest score for a scene.

Scene	cubic-spline	$\phi = 0$	$\phi \leftarrow \boldsymbol{f}$	ϕ_{a}	ϕ_{m}
K01	0.79171	0.78529	0.79186	0.79105	**0.79488**
K02	0.79431	0.78822	0.79418	0.79390	**0.79533**
K03	0.88532	0.88176	0.88536	0.88590	**0.88671**
K04	0.84944	0.84500	0.84945	0.84957	**0.84966**
K05	0.89773	0.89537	0.89800	0.90238	**0.90951**
K06	0.80009	0.79398	0.80017	0.79959	**0.80208**
K07	0.91242	0.90912	0.91255	0.91440	**0.91663**
K08	0.74572	0.73795	0.74577	0.74475	**0.74741**
K09	0.75778	0.75053	0.75774	0.75745	**0.75866**
K10	0.80214	0.79586	0.80216	0.80197	**0.80333**
K11	0.81270	0.80622	0.81270	0.81313	**0.81522**
K12	0.87742	0.87282	0.87745	0.87809	**0.87886**
K13	0.74594	0.73999	0.74592	0.74539	**0.74680**
K14	0.85891	0.85460	0.85887	0.85960	**0.86369**
K15	**0.85792**	0.85368	0.85780	0.85763	0.85772
K16	0.78761	0.78140	0.78763	0.78733	**0.78839**
K17	0.81080	0.80445	0.81073	0.81061	**0.81204**
K18	**0.76902**	0.76340	0.76894	0.76852	0.76821
K19	0.80726	0.80047	0.80738	0.80837	**0.81033**
K20	0.82786	0.82207	0.82793	0.83020	**0.83040**
K21	0.78992	0.78365	**0.78998**	0.78937	0.78997
K22	0.**84270**	0.83747	**0.84270**	0.84222	0.84236
K23	0.90045	0.89687	0.90045	0.90072	**0.90094**
K24	0.86907	0.86474	0.86927	0.86898	**0.87013**

Table 3.3: IW-SSIM scores with respect to $\boldsymbol{f}_{\mathrm{ref}}$ under a downsample-without-prefiltering model, where 1 out of every 6 samples of $(\boldsymbol{f}_{\mathrm{ref}})_{\mathcal{E}}$ is retained (per dimension) to form $\left(\boldsymbol{f}_{\mathrm{ob},\mathcal{X}}\right)_{\mathcal{E}}$. Bold is the highest score for a scene.

Scene	cubic-spline	$\phi = 0$	$\phi \leftarrow \boldsymbol{f}$	ϕ_{a}	ϕ_{m}
K01	0.82813	0.83430	0.82855	0.82965	**0.83439**
K02	0.87206	**0.87652**	0.87208	0.87246	0.87498
K03	0.93401	**0.93691**	0.93416	0.93505	0.93674
K04	0.90263	**0.90588**	0.90273	0.90311	0.90506
K05	0.89296	0.89833	0.89320	0.89631	**0.90016**
K06	0.84973	**0.85535**	0.85001	0.85059	0.85368
K07	0.94769	0.94987	0.94788	0.94915	**0.95110**
K08	0.81217	0.81836	0.81269	0.81443	**0.81838**
K09	0.91985	0.92235	0.92010	0.92117	**0.92278**
K10	0.92410	0.92687	0.92430	0.92506	**0.92697**
K11	0.86616	**0.87076**	0.86639	0.86780	0.87066
K12	0.93123	**0.93422**	0.93142	0.93218	0.93393
K13	0.79233	**0.79845**	0.79248	0.79285	0.79644
K14	0.88306	0.88758	0.88321	0.88429	**0.88776**
K15	0.90751	0.91040	0.90760	0.90807	**0.90961**
K16	0.86183	**0.86614**	0.86197	0.86240	0.86443
K17	0.90982	0.91249	0.90992	0.91083	**0.91290**
K18	0.85693	**0.86112**	0.85703	0.85735	0.86008
K19	0.87064	0.87486	0.87108	0.87301	**0.87553**
K20	0.90526	0.90844	0.90540	0.90689	**0.90902**
K21	0.87183	**0.87594**	0.87203	0.87296	0.87520
K22	0.88902	**0.89272**	0.88913	0.88974	0.89172
K23	0.95184	**0.95376**	0.95189	0.95234	0.95353
K24	0.88279	**0.88769**	0.88303	0.88360	0.88666

Table 3.4: IW-SSIM scores with respect to $\boldsymbol{f}_{\mathrm{ref}}$ under a downsample with prefilter model, where 1 out of every 5 samples of $(\boldsymbol{f}_{\mathrm{ref}})_{\mathcal{E}}$ is retained (per dimension) to form $\left(\boldsymbol{f}_{\mathrm{ob},\,\mathcal{X}}\right)_{\mathcal{E}}$. Bold is the highest score for a scene.

Scene number	cubic-spline	$\phi = 0$	$\phi \leftarrow \boldsymbol{f}$	ϕ_{a}	ϕ_{m}
K01	0.89470	**0.89930**	0.89497	0.89546	0.89856
K02	0.91931	**0.92300**	0.91935	0.91968	0.92168
K03	0.96040	**0.96244**	0.96049	0.96086	0.96208
K04	0.93823	**0.94097**	0.93830	0.93846	0.94012
K05	0.94132	0.94507	0.94146	0.94295	**0.94544**
K06	0.90766	**0.91250**	0.90786	0.90829	0.91080
K07	0.96794	0.96946	0.96804	0.96879	**0.96989**
K08	0.87890	**0.88422**	0.87928	0.88029	0.88404
K09	0.94469	**0.94662**	0.94483	0.94555	**0.94662**
K10	0.94896	**0.95089**	0.94908	0.94947	0.95078
K11	0.91657	**0.92059**	0.91673	0.91758	0.92002
K12	0.95845	**0.96078**	0.95857	0.95903	0.96033
K13	0.86807	**0.87358**	0.86818	0.86846	0.87147
K14	0.92873	**0.93234**	0.92882	0.92952	0.93194
K15	0.94196	**0.94478**	0.94201	0.94227	0.94375
K16	0.91113	**0.91501**	0.91124	0.91151	0.91329
K17	0.93778	**0.93993**	0.93783	0.93857	0.93989
K18	0.90313	**0.90642**	0.90320	0.90347	0.90559
K19	0.91829	**0.92203**	0.91861	0.91972	0.92188
K20	0.93938	0.94193	0.93947	0.94036	**0.94194**
K21	0.91696	**0.92046**	0.91709	0.91765	0.91970
K22	0.92977	**0.93289**	0.92985	0.93009	0.93210
K23	0.96973	**0.97101**	0.96976	0.97002	0.97074
K24	0.93213	**0.93602**	0.93227	0.93258	0.93489

Table 3.5: IW-SSIM scores with respect to $\boldsymbol{f}_{\mathrm{ref}}$ under a downsample with prefilter model, where 1 out of every 4 samples of $\boldsymbol{f}_{\mathrm{ref}}$ is retained (per dimension) to form $\boldsymbol{f}_{\mathcal{X}}$. Bold is the highest score for a scene.

Scene	cubic-spline	$\phi = 0$	$\phi \leftarrow \boldsymbol{f}$	ϕ_{a}	ϕ_{m}
K01	0.76092	0.76697	0.76152	0.76262	**0.76866**
K02	0.82392	0.82868	0.82392	0.82438	**0.82715**
K03	0.90516	0.90851	0.90535	0.90683	**0.90887**
K04	0.86626	**0.86983**	0.86640	0.86703	0.86920
K05	0.83652	0.84240	0.83686	0.83985	**0.84650**
K06	0.79050	**0.79615**	0.79087	0.79194	0.79516
K07	0.92452	0.92720	0.92479	0.92681	**0.92924**
K08	0.74493	0.75082	0.74560	0.74732	**0.75245**
K09	0.89344	0.89650	0.89381	0.89544	**0.89734**
K10	0.89619	0.89958	0.89648	0.89794	**0.90023**
K11	0.81552	0.82023	0.81582	0.81759	**0.82066**
K12	0.90196	0.90522	0.90219	0.90313	**0.90533**
K13	0.71744	**0.72333**	0.71766	0.71826	0.72175
K14	0.83409	0.83895	0.83427	0.83618	**0.83978**
K15	0.87523	**0.87854**	0.87534	0.87626	0.87805
K16	0.81295	**0.81748**	0.81314	0.81372	0.81575
K17	0.88118	0.88424	0.88131	0.88262	**0.88515**
K18	0.80958	**0.81404**	0.80969	0.81024	0.81284
K19	0.82303	0.82694	0.82359	0.82562	**0.82840**
K20	0.87134	0.87484	0.87153	0.87404	**0.87623**
K21	0.82676	**0.83108**	0.82700	0.82771	0.83056
K22	0.84702	0.85109	0.84716	**0.84788**	0.85026
K23	0.93145	**0.93389**	0.93154	0.93227	0.93388
K24	0.83051	**0.83563**	0.83080	0.83169	0.83518

Table 3.6: IW-SSIM scores with respect to $\boldsymbol{f}_{\mathrm{ref}}$ under a downsample with prefilter model, where 1 out of every 6 samples of $(\boldsymbol{f}_{\mathrm{ref}})_{\mathcal{E}}$ is retained (per dimension) to form $\left(\boldsymbol{f}_{\mathrm{ob},\,\mathcal{X}}\right)_{\mathcal{E}}$. Bold is the highest score for a scene.

From our visual inspection benchmarks, we often see that two different adapted kernels produce a similar result. This was shown in Figure 3.9, where the up-conversion result of the $\phi = 0$ and the $\phi \leftarrow \boldsymbol{f}$ strategies look similar. Their adapted basis vectors are shown in Figure 3.6 to be very different. This is a sensible observation because there are likely to be different sparse dictionaries for the application at hand that give similar results, where "similar" is application-dependent. This suggests that a construction strategy for ϕ that is based on free-form optimization is unsuitable, because there are many "almost equivalent" solutions for ϕ such that it induces a "sparse enough" adapted basis.

The monogenic adaptive smoothing ϕ_{m} strategy fails when there is too little information in $f_{\mathrm{ob}}|_{\mathcal{X}}$ to specify a ϕ that induces a sparse basis for $\left(\tilde{\boldsymbol{f}}_{\mathcal{X}_q}\right)_{\mathcal{E}}$. For scene 18, the high-contrast edges from the contours of the necklace beads along with severe aliasing artefacts gave an unrealistic up-converted result (see Figure 3.17). A practical deployment of the adaptive kernel needs improved local analysis schemes that look for crippling cases like this. If one is unable to devise a robust adaptive kernel that adequately handles such scenarios, a pre-processing step that decides whether or not to adapt the kernel could help. I think an interesting future direction is to train different presets for the tuning parameters of construction strategies for ϕ using an offline data set. Another idea is to perform scene-content analysis on the observed image, then specify different tuning parameters for the different patches of the image. However, the ideal patch shapes and locations in this case should probably be non-parametric and adapted from the observation, as opposed to the fixed rectangular RKHS patch system that we used in the demonstrations.

The isotropic strategy $\phi = 0$ was very favourable in our downsample-with-prefiltering benchmarks, and I think the reason is because an isotropic low-pass filter was used as the prefilter. The task of extending the RKHS regularization framework to cover observation models other than the ones investigated here is a possible extension of the work proposed in this chapter.

In summary, the goal of our demonstrations was to present exemplar strategies for specifying warp maps that remove aliasing artefacts along the edges of grayscale images. We chose the assumption that edges in f should have smooth contours, and presented the adaptive smoothing ϕ_{a} and the monogenic adaptive smoothing ϕ_{m} strategies. The reference strategies are the interpolated observation strategy $\phi \leftarrow \boldsymbol{f}_{\mathcal{X}}$, which assigns $\phi(\mathcal{X}) = f_{\mathrm{ob}}|_{\mathcal{X}}$, and the isotropic strategy $\phi = 0$, which assigns ϕ to zero for all inputs. Visual inspections of the adapted basis vectors from each strategy were presented, as well as up-conversion results of selected cropped scenes. IW-SSIM image similarity scores against the reference images were presented for both the downsample-with-prefiltering and downsample-without-prefiltering observation models. Scenes with lots of high-contrast edges consistently had higher IW-SSIM scores under the ϕ_{a} and ϕ_{m} strategies. These two strategies did not perform well when edge contours were not accurately inferred. For the adaptive RKHS regularization framework to be effective for an application, one should clearly state their chosen prior assumptions on f, then try different strategies for constructing a warp map that implements these assumptions. The challenge is to design a scheme that is robust to situations when these assumptions do not hold.

CHAPTER 4

Optimization over a Space of Rank-One Decompositions

Recall from Chapter 1 that a *rank-one decomposition of cardinality* p (or CP rank p) is a set of p rank-one tensors that have the same given dimension D and tensor order L. We do not mention p when it is clear from context, or unimportant to the discussion at hand. In this chapter, we present a novel approach to optimize any differentiable cost function over the following space of rank-one decompositions of cardinality p:

$$\begin{aligned}\mathcal{S}_{\mathrm{E}}(D,\, L,\, p) = \{(\mathfrak{z},\, c)\}, \qquad & (4.1)\\ \mathfrak{z} \in \left\{z^{(j)} \in \mathbb{R}_{\mathrm{s}}^{D}\right\}_{j=1}^{p},\ & \left(z^{(j)}\right)_1 = 1,\ z^{(j)} \neq b z^{(i)},\ \forall i,\, j \in [p],\, \forall b \in \mathbb{R},\\ c \in \mathbb{R}_{++}^{p},\ & L \text{ is even.}\end{aligned}$$

When the given dimension D, even order L, and CP rank p is clear from context, or unimportant in the discussion at hand, we simply write $\mathcal{S}_{\mathrm{E}}$ to mean $\mathcal{S}_{\mathrm{E}}(D,\, L,\, p)$. We shall see in Proposition 4.1 that this space of rank-one decompositions is isomorphic to a space of quasi-Hankel matrices, which motivates us to work with this quasi-Hankel matrix space as opposed to the space of symmetric tensors. The work done here assumes that a quasi-Hankel matrix is given from data, or obtained by some matrix-completion problem as a pre-processing step.

The motives of the work presented in this chapter are explained in more detail in Section 4.1, and the objectives are reviewed in Section 4.2. Related works are reviewed in Section 4.3. In Section 4.4, we summarize the notation and setup used in this chapter. We pose $\mathcal{S}_{\mathrm{E}}$ as differentiable manifolds in Section 4.5, and the proposed numerical optimization scheme over $\mathcal{S}_{\mathrm{E}}$ is presented in Section 4.6. Results on synthetically-generated data are reported in Section 4.7.

4.1 Motives

We discuss the reasons behind my decision to work with $\mathcal{S}_{\mathrm{E}}$ in this section. We first start with the formulation of rank-one decompositions that is used in this thesis. In Section 4.1.2, we clarify the uniqueness issue of the CP decomposition problem for symmetric tensors. This provides motivation for developing a general-purpose optimizer over a space of tensors, so that modified versions of the CP decomposition problem that are not ill-posed can be solved instead. This also motivates us to seek a space of objects that are isomorphic to rank-one decompositions, so that we can work with those objects instead of symmetric tensors. It was implied in the results from [29] that such objects are quasi-Hankel matrices. In Section 4.1.3.1, we discuss

my interpretation of Theorem 6.3 from [29] in the form of Proposition 4.1, which clarifies the relationship between quasi-Hankel matrices and rank-one decompositions. We then discuss in Section 4.1.3.2 how one can construct a quasi-Hankel matrix from data, and provide references to existing quasi-Hankel matrix-completion schemes. This provides motivation for working with a space of quasi-Hankel matrices instead of symmetric tensors.

In Section 4.1.4, we discuss the motivation behind formulating a quotient manifold $\mathcal{M}$ that treats a rank-one decomposition as an unordered set (as opposed to an ordered set). The details of the derivation of $\mathcal{M}$ are deferred to Section 4.5.4.

4.1.1 Representation of Rank-One Decompositions

We review the results of Section 2.14.3 before defining our representations of rank-one decompositions. Consider the space of D-variate, Lth-degree, homogeneous polynomials:

$$\left\{ \sum_{j=1}^{p} \lambda_j \left(\sum_{d=1}^{D} u_d^{(j)} x_d \right)^L \;\middle|\; \lambda_j \in \mathbb{R} \setminus \{0\} \right\}, \tag{4.2}$$

where x is a D-tuple of real-valued variables, and $\left\{u_d^{(j)} \in \mathbb{R}\right\}_{d=1,\,j=1}^{D,\,p}$ is a set of bounded real numbers where $u_1^{(j)} \neq 0$ for all $j \in [p]$. In general, there exists at least one $k \in [D]$ such that $u_k^{(j)} \neq 0$ for all $j \in [p]$, otherwise we do not have a homogeneous polynomial of degree L, so we can always re-order the variables in x so that $u_1^{(j)} \neq 0$ for all $j \in [p]$. The objects u_d and λ_j together describe the components of each of the rank-one tensors with respect to some basis in an element of this space.

We can alternatively describe this space of polynomials by the dehomogenized version of each D-variate, Lth-degree, homogeneous polynomial:

$$\left\{ \sum_{j=1}^{p} c_j \left(1 + \sum_{d=2}^{D} z_d^{(j)} x_d \right)^L \;\middle|\; c_j \in \mathbb{R} \setminus \{0\},\, z_d \in \mathbb{R}^D \right\}$$

The new polynomial coefficients $\left\{z_d^{(j)}\right\}$ and the new scale factors $\{c_j\}$ are defined as

$$z_d^{(j)} := \left. \frac{u_d^{(j)}}{u_1^{(j)} x_1} \right|_{x_1=1} = \frac{u_d^{(j)}}{u_1^{(j)}},\ \forall d \in [D],\ \forall j \in [p],$$

and

$$c_j := \lambda_j \left(u_1^{(j)} x_1 \right)^L \Big|_{x_1=1} = \lambda_j \left(u_1^{(j)} \right)^L,\ \forall d \in [D].$$

It follows that $z_1^{(j)} = 1$ for all $j \in [p]$. These two polynomial spaces are isomorphic to each other, so we can represent a rank-one decomposition by either $(u,\, \lambda)$ or $(\mathfrak{z},\, c)$, where $u = \{u_d\}$, $\lambda = \{\lambda_j\}$,

$$\mathfrak{z} := \left\{ z^{(j)} \in \mathbb{R}_{\text{s}}^D \;\middle|\; \left(z^{(j)}\right)_1 = 1 \right\}_{j=1}^{p},$$

and $c \in \{\mathbb{R} \setminus \{0\}\}^p$. We can see that the space $\mathcal{S}_{\text{E}}(D,\, L,\, p)$ uses the dehomogenized polynomial representation.

4.1.2 Uniqueness of the CP Decomposition Problem for Symmetric Tensors

Our discussion here is based on the material from reference [30]. The uniqueness results reviewed here are separate from the permutation-related non-identifiability issues of the CP decomposition problem, so we do not discuss the latter in this subsection.

The key result of Section 7.2 from [30] is that, for only some combinations of dimension D and order L, the rank-one decomposition of a given symmetric tensor is unique. Another key result from this reference is that a CP decomposition for any given symmetric tensor always exists. This means that whilst a symmetric tensor is guaranteed to map to a rank-one decomposition (with some unknown CP rank p), this is only an injective map for certain combinations of D and L. For the general case, it is a one-to-many map. Therefore, the space of rank-one decompositions $\mathcal{S}_{\text{E}}$ is not isomorphic to a space of symmetric tensors. Similar uniqueness issues also exist for rectangular tensors [33]. Some works [33, 37, 36] attribute the unstable performance of CP existing decomposition algorithms to this non-uniqueness issue.

Some of the existing uniqueness results require knowledge of the underlying CP rank r or other objects that may be unrealistic to obtain in practice. Most existing approaches that pose dimensionality reduction problems as a tensor decomposition problem construct a tensor T from data [21, 24, 18, 26]. These approaches then fix a p to be the CP rank of the decomposition solution, and attempt to decompose T into $p \ll r$ rank-one tensors, where r is the true CP rank of T. Although uniqueness results are used to explain problems such as convergence to different solutions with different initial guesses, it is not obvious how they can be applied to improve numerical performance. Some recent works [32, 38] solve a relaxed version of the CP decomposition to reduce the effects of these issues.

4.1.3 Quasi-Hankel Matrices and Rank-One Decompositions

In this thesis, a quasi-Hankel matrix is a symmetric, real-valued, and square moment matrix that is defined in Definition 2.115. Given a basis of a symmetric tensor space, a quasi-Hankel matrix constructed from the ordered set of monomial exponents $\mathbb{A}(D,\, L)_{\succ_{\text{V}}}$ has the property that its first row and column are bijective to the unique components of a symmetric tensor under some basis. Components of tensors are reviewed in Section 2.3, and the relationship between quasi-Hankel matrices and symmetric tensors is reviewed in Section 2.14.5. The topics discussed in this subsection provide motives for one to work with cost functions that take a quasi-Hankel matrix as input, as opposed to a symmetric tensor as input. An example of a rank-one decomposition and its corresponding quasi-Hankel matrix is discussed in Example 2.125.

Symmetric tensors do not have a unique CP decomposition in the general case [30]. There is no uniqueness problem (excluding the permutation-related non-identifiability issues) if one were to decompose a quasi-Hankel matrix that satisfies certain conditions [29]. We explicitly state this as Proposition 4.1 in Section 4.1.3.1. This can be informally interpreted as: the entries other than the first row and column of a quasi-Hankel matrix are used to "stabilize" the decomposition problem by providing more information to fit to. This in turn decreases the space of equivalent rank-one decomposition solutions to a single solution.

There are unknown entries in a quasi-Hankel matrix if one is given only a symmetric tensor that is constructed from data; see Example 2.125 and the discussion that follows it. For some applications, it is possible to directly construct a quasi-Hankel matrix from data. Otherwise, one needs to solve a matrix-completion problem using a symmetric tensor that is constructed from data. We give references for works that deal with the quasi-Hankel matrix-completion problem in Section 4.1.3.2.

4.1.3.1 Uniqueness of Rank-One Decomposition of Quasi-Hankel Matrices

Our discussion here requires the background reviewed in Section 2.13. I constructed the following proposition based on the result of Theorem 6.3 from [29]:

Proposition 4.1. *Let D and p be given finite positive integers, and $c \in \mathbb{R}_{++}^p$ a given p-tuple of strictly positive real numbers. Let L be a given finite, positive, even integer such that $|\mathbb{A}(D, L)| > p$. Let $\mathfrak{z} \in \left\{z^{(n)} \in \mathbb{R}_s^D\right\}_{n=1}^p$ be given such that $z_1^{(n)} = 1$ and each $z^{(n)}$ is not a multiple of $z^{(m)}$, $\forall m, n \in [p]$, $m \neq n$. Then the quasi-Hankel matrix $\mathtt{H}\left(\mathbb{A}(D, L)_{\succ_V}, \mathfrak{z}, c\right)$ is uniquely generated by $\mathfrak{z}$ and c.*

Proof. In this proof, x is a D-tuple of real-valued variables where the first slot is fixed to 1. We use this dehomogenized version of x since we are working with the dehomogenized representation $(\mathfrak{z}, c)$ of a rank-one decomposition instead of the homogeneous polynomial representation (u, λ); see Section 4.1.1 for more details. In this proof, if a multi-index set $\mathbb{B} \subset \mathbb{A}(D, L)$ is such that the set of monomials

$$\mathcal{C} = \{x^\alpha \mid \alpha \in \mathbb{B}\}$$

is connected by 1 (Definition 2.113), then we define the multi-index set $\mathbb{B}^+$ by the condition

$$\beta \in \mathbb{B}^+ \iff x^\beta \in \mathcal{C}^+. \tag{4.3}$$

We denote by $\succ$ and $\succ_+$ any ordering on $\mathbb{B}$ and $\mathbb{B}^+$, respectively.

Recall from Section 2.14.5 that the first row or column of a quasi-Hankel matrix as specified in Definition 2.115 can be bijectively mapped to the components of a symmetric tensor (with respect to some basis). Consider the homogeneous polynomial f that corresponds to the first column (or row) of $\mathtt{H}\left(\mathbb{A}(D, L)_{\succ_V}, \mathfrak{z}, c\right)$. If the quasi-Hankel matrices $\mathtt{H}(\mathbb{B}_\succ, \mathfrak{z}, c)$ and $\mathtt{H}\left(\mathbb{B}_{\succ_+}^+, \mathfrak{z}, c\right)$ both have rank p, then by Theorem 6.3 from [29] and its proof that is published there, f has a unique rank-one decomposition of cardinality p that also generates $\mathtt{H}\left(\mathbb{B}_{\succ_+}^+, \mathfrak{z}, c\right)$.

The following set of monomials is connected to 1:

$$\{x^\alpha \,|\, \alpha \in \mathbb{A}(D, L)\}.$$

This is due to the definition of $\mathbb{A}(D, L)$ (Equation 2.3), and is easier to see if we use $\mathbb{A}_{\text{de}}(D, L)$ defined in Appendix B. The quasi-Hankel matrix $\mathtt{H}\left(\mathbb{A}(D, L)_{\succ_V}, \mathfrak{z}, c\right)$ is positive-definite with rank p, so we can always choose a $\mathbb{B}$ such that 1)

$$\{x^\alpha \mid \alpha \in \mathbb{B}(D, L)\}$$

is a set of monomials connected to 1, and 2) $\mathtt{H}(\mathbb{B}_\succ, \mathfrak{z}, c)$ is a rank-p principal submatrix of $\mathtt{H}\left(\mathbb{A}(D, L)_{\succ_V}, \mathfrak{z}, c\right)$.

Construct $\mathbb{B}^+$ from this choice of $\mathbb{B}$ using Equation 4.3. Since we required $|\mathbb{A}(D, L)| > p$, the matrix $\mathtt{H}\left(\mathbb{B}_{\succ_+}^+, \mathfrak{z}, c\right)$ is also another principal submatrix of $\mathtt{H}\left(\mathbb{A}(D, L)_{\succ_V}, \mathfrak{z}, c\right)$. This is due to Proposition 2.114, Definition 2.111 (moment matrices) and the discussion following it, Definition 2.115 (quasi-Hankel matrices), and Definition 2.112 (the closure of $\mathcal{C}$, $\mathcal{C}^+$). Since $\mathcal{C} \subset \mathcal{C}^+$, we have

$$\mathtt{H}(\mathbb{B}_\succ, \mathfrak{z}, c) \subset \mathtt{H}\left(\mathbb{B}_{\succ_+}^+, \mathfrak{z}, c\right) \subset \mathtt{H}\left(\mathbb{A}(D, L)_{\succ_V}, \mathfrak{z}, c\right),$$

where $A \subset B$ here means that A can be re-arranged to be a principal submatrix of B. It follows that the rank of $\mathtt{H}\left(\mathbb{B}_{\succ_+}^+, \mathfrak{z}, c\right)$ is p. Therefore, the first column of $\mathtt{H}\left(\mathbb{A}(D, L)_{\succ_V}, \mathfrak{z}, c\right)$ is uniquely generated by the given rank-one decomposition $(\mathfrak{z}, c)$.

If a portion of a matrix is uniquely generated, then the entire matrix as a single object is uniquely generated. It follows that $\{c, \mathfrak{z}\}$ uniquely generates $\mathtt{H}\left(\mathbb{A}(D, L)_{\succ_V}, \mathfrak{z}, c\right)$. □

The consequence of this proposition is that, whilst the problem of recovering a rank-one decomposition that generated a given symmetric tensor with arbitrary dimension and order is not a well-defined problem (i.e., more than one solution may exist), the problem of recovering a rank-one decomposition that generated a given quasi-Hankel matrix is well-defined if the conditions described in Proposition 4.1 are met. More general conditions probably exist, but we do not pursue this topic in this thesis.

Under an optimization setting where the rank-one decomposition $\{\mathfrak{z},\, c\}$ is the optimization variable, one can freely choose the quasi-Hankel order L and p such that $|\mathbb{A}(D,\, L_2)| > p$. As specified in Proposition 4.1, our optimization framework needs to force every update to the variable $\mathfrak{z}$ to be a set of non-collinear vector components, and this thesis work used a differentiable manifold formulation for $\{\mathfrak{z},\, c\}$, so that such requirements are taken care of by retraction maps. In conclusion, by working with $\{\mathfrak{z},\, c\}$ with a cost function that takes a quasi-Hankel matrix as input, we are certain that $\{\mathfrak{z},\, c\}$ is isomorphic to a quasi-Hankel matrix that satisfies the conditions in Proposition 4.1.

4.1.3.2 Quasi-Hankel Matrices from Data

When one wants to solve a quasi-Hankel decomposition problem when only a symmetric tensor T of order L_1 is given, one needs to convert T to a quasi-Hankel matrix $\mathtt{H}\Big(\mathbb{A}(D,\, L_2)_{\succ_{\mathrm{V}}},\, \mathfrak{z},\, c\Big)$, where $L_2 \geq L_1$ is chosen such that L_2 and p meet the requirement $|\mathbb{A}(D,\, L_2)| > p$. This is done by solving a quasi-Hankel matrix-completion problem, which implicitly chooses a $(\mathfrak{z},\, c)$ by the regularization scheme used by the matrix-completion scheme. One then solves for $(\mathfrak{z},\, c)$, which is unique. Existing works on solving this matrix-completion problem can be found in [29, 39, 40, 41, 42, 43].

It is possible to construct a quasi-Hankel matrix directly from data in some applications. Symmetric tensors are used when modelling the Lth-order random moment of a random vector [31], and if the moment is empirically estimated from realizations of the random vector as opposed to realizations of the moment itself, then one is free to compute empirical estimates for moments of different orders, as opposed to moments of only the Lth-order. Under this setting, the entries of the quasi-Hankel matrix $\mathtt{H}\Big(\mathbb{A}(D,\, L_2)_{\succ_{\mathrm{V}}},\, \mathfrak{z},\, c\Big)$ correspond to random moments of different orders, so they can be computed directly from data. One can use Monte-Carlo estimation or importance sampling to estimate the random moments of arbitrary orders. One example of this in use in the literature is [16], where the given tensor used in the demonstrations was computed from Monte-Carlo estimation. The diffusion tensor of magnetic resonance imaging [15] and the Raman tensor from biophysics [75] are both tensors that are constructed from empirical measurements. For these type of applications, I suspect that it is possible to work out a list of additional measurements that can be used to compute the entries of $\mathtt{H}\Big(\mathbb{A}(D,\, L_2)_{\succ_{\mathrm{V}}},\, \mathfrak{z},\, c\Big)$ that does not correspond to a symmetric tensor.

In the event that a quasi-Hankel matrix cannot be obtained, one can still use the proposed work in this chapter by ignoring the entries of the quasi-Hankel matrix that is not in the first row or column. For example, consider the cost function f, given by

$$f(B) = (S. * (\mathtt{H} - B)) \circ_{\mathrm{F}} (S. * (\mathtt{H} - B)), \tag{4.4}$$

where the *mask matrix* S is a given matrix of the same size as $\mathtt{H}$, the matrix variable B is the quasi-Hankel matrix that is isomorphic to the optimization variable $\{\mathfrak{z},\, c\}$, the operator $.*$ is the per-element multiplication operator between two matrices, and the operator $\circ_{\mathrm{F}}$ is the Frobenius product between matrices. One can avoid having to specify a value for $\mathtt{H}$ in the entries that do not correspond to the given tensor if the entries in S are zeros except in the first row and column. The disadvantage of this approach is that the optimizer will ignore the uniqueness property discussed in Proposition 4.1.

An interesting approach is to combine the matrix-completion and mask matrix approaches, so that a quasi-Hankel matrix $\tilde{H}$ that is inexpensive to compute is set as $\mathtt{H}$, and S is chosen such that it has a value close to zero for entries that are not in the first row nor column. This way, the entries in $\tilde{H}$ that do not correspond to a symmetric tensor are not as emphasized in the fit to data as the entries of $\tilde{H}$ that do. These entries are still used to stabilize the rank–one solution so that the optimization problem is well-posed (ignoring permutation-related non-identifiability issues). This approach is intended for future works, since a survey of computationally inexpensive quasi-Hankel matrix-completion schemes is out of the scope of this thesis.

4.1.4 Quotient Manifolds and Riemannian Structure Approximation

The topics discussed here are motives for using an approximation $G_\diamond$ (introduced in Section 4.5.5) that mimics the behaviour of a Riemannian metric on a quotient manifold. In Chapter 1, we introduced the idea of non-identifiability issues for an optimization problem. An example of an optimization problem that suffers from non-identifiability issues is the unsupervised clustering task when it is posed as the task: "simultaneously learn a set of labels and its correspondence to data points". If the set of possible labels is $\{A, B\}$ and there are three data points X, Y, and Z, then the solution where X and Y corresponds to A and Z corresponds to B is equivalent to the solution where X and Y corresponds to B and Z corresponds to A. It follows that one can permutate the labels of the clusters to get an equivalent solution. For this task, a more sensible philosophical question to ask is "which of the data points belong together?", which requires one to pose each candidate solution as a partition of the set of data points. For example, the two equivalent solutions just discussed both describe the same partition $\{\{X, Y\}, \{Z\}\}$ of the set $\{X, Y, Z\}$. The space of all equivalence classes is called a *quotient space*, and Section 2.11 reviews this topic in the context of differentiable manifolds. The key in the second formulation is that a partition is an unordered set instead of an ordered set. Therefore, the clustering example we just discussed is an example of a *permutation-related non-identifiability issue*, which we briefly introduced in Section 1.2.1. These issues are troublesome because they give rise to massive numbers of equivalent solutions when the problem scales up.

Non-identifiability issues could slow down the convergence rate in optimization algorithms, because the search space at each iteration may produce an update direction that has a significant component that points towards an equivalent solution. This problem is amplified when heuristic strategies for traversing between modes of the objective function are used, since these strategies may think they are moving between local modes when they might be modes formed by different equivalent solutions. Local search derivative-based methods also suffer from these issues because the search direction may have a significant portion that points towards an equivalent solution.

Riemannian optimization problems (e.g., see [1]) avoid this issue by using a *Riemannian metric* to define a vector subspace of the tangent space, called the *horizontal tangent space*, that does not have any component towards an equivalent solution. These concepts are reviewed in more detail in Sections 2.11 and 2.12. If a local search direction is based on the gradient $\sharp$-isomorphism vector, then it is possible to eliminate the direction component that corresponds to another equivalent solution by using a Riemannian metric on a quotient manifold to induce the $\sharp$-isomorphism. The space of equivalent solutions is characterized by the stabilizer group of the quotient manifold. Stabilizers and quotient manifolds are reviewed in Sections 2.10 and 2.11.

A retraction map is required for optimization on a differentiable manifold [1]. I was unable to devise a suitable retraction for the quotient manifold $\mathcal{M}$ proposed in this chapter. I decided to use an approximation $G_\diamond$ in lieu of a Riemannian metric on the non-quotient manifold $\mathcal{P}$ to see if it can reduce the effects of permutation-related non-identifiability issues. However, I do not know of any existing convergence theory for using such an approximation to a Riemannian metric, and this topic was not pursued in this thesis. The empirical performance of $G_\diamond$ and that of a proper Riemannian metric $G_\mathcal{P}$ are both reported in Section 4.7.2.1. I believe such approximations to Riemannian structures of a differentiable manifold should be explored more in the literature. This is because Riemannian metrics and related structures can be computationally expensive to evaluate, so approximations may offer a good compromise between convergence properties and computational burden.

4.2 Objectives

One of the primary objectives for the work done in this chapter is to develop a general-purpose numerical optimization framework over $\mathcal{S}_{\text{E}}$. Investigations for applications that use symmetric tensors are outside the

scope of this thesis, but are intended for future work. For this reason, the demonstrations in this chapter focus on evaluating the numerical ability for the proposed optimization framework to minimize a given objective function. The test objective function is mentioned in Equation 4.4, and it was chosen to implement the CP decomposition problem. For every optimization problem tested, the given quasi-Hankel matrix $\mathtt{H}$ is constructed from a set of rank-one decompositions with cardinality p. The number p is known to the optimizer, so that it can recover these ground truth rank-one decompositions. This synthetic data setup was chosen because a minimum cost of zero is attained when the CP decomposition is successful. We report the results in Section 4.7.2. The goals of the demonstrations here are to verify the proposed optimization framework, and to deduce whether $G_\diamond$ had any clear advantages over $G_\mathcal{P}$. Although we compare our demonstration results against a CP decomposition method in Section 4.7.2.2, our optimizer was not designed to be a competitive CP decomposition method.

Another primary objective in this chapter is to derive a differentiable submanifold $\mathcal{P}$ and quotient manifold $\mathcal{M}$ that correspond to $\mathcal{S}_\mathrm{E}$. One can swap the proposed retractions and Riemannian structures with their own without affecting the theoretical results presented in this chapter. One of our secondary objectives is to present theoretical formulation of these manifold-related objects in a detailed manner, so that others who are interested in learning about differentiable manifolds may use this chapter as a learning aid. A suitable retraction for $\mathcal{M}$ was not devised in this chapter, so an optimizer over $\mathcal{M}$ was not proposed. Another of our secondary objectives is to evaluate the effectiveness of an approximate Riemannian metric $G_\diamond$ for the manifold $\mathcal{P}$ that mimics the proposed Riemannian metric for $\mathcal{M}$. The results of $G_\diamond$ against the proper Riemannian metric $G_\mathcal{P}$ for $\mathcal{P}$ are shown in Section 4.7.2.1.

4.3 Related Works

In the recent work [76], a general-purpose optimizer over the space of hierarchical Tucker decompositions was developed. In was shown that such a tensor decomposition was unique, given a fixed hierarchical Tucker decomposition tree structure. In this thesis, we focus on the more mature but problematic rank-one decompositions. I am unaware of any existing work on general-purpose optimization over rank-one decompositions, symmetric tensor spaces, nor over moment matrix spaces.

The recent works [77, 78, 44, 79] are about using an optimization framework over a matrix space to solve a tensor-completion problem. They represent the component-form of a tensor, which is a multi-dimensional array, as a set of matrices that corresponds to a set of slices of this array. This is a popular representation scheme in the tensor decomposition research community, and we refer to this tensor representation method as the *mode-matrix representation* for the rest of our discussion here. A reference for this representation method and its formulation to the CP and Tucker tensor decomposition problems can be found in the survey article [33].

The space of matrices under the mode-matrix representation of tensors is very different from the quasi-Hankel matrix space that we consider in this thesis. The critical advantage of using mode-matrix setup is that conventional linear algebra operations such as least-squares optimization via solving linear systems and the singular-value decomposition can be used on this set of matrices. This approach is taken by the majority of the existing tensor decomposition works in the literature. The disadvantage is that the underlying tensor structure is lost, since only slices of the array of tensor components are processed at any given time. The symmetry property of symmetric tensors are not exploited under this representation. For this reason, some works that focus on the uniqueness and existence of the symmetric tensor decomposition problem use other representations of tensors, such as a multilinear setup in [37], or a polynomial setup in [30, 28]. These uniqueness results are readily understood using conventional multilinear algebra theory [30, 37, 28], whereas the works that use a mode-matrix representation have less obvious interpretations from the perspective of multilinear algebra.

The tensor-completion works [77, 78, 44] use a differentiable manifold approach on their matrix variables. Retractions and quotient manifolds were used in the tensor-completion work in [44]. Their matrix space follows the mode-matrix representation of a tensor, and they formulated that space to characterize the Tucker decomposition, also known as the *higher-order singular value decomposition* in the literature. They devised a quotient manifold and accompanying retraction to address non-identifiability issues. The quotient manifold portion of the work done in this chapter follows the same motivation, except the factorization from Equation 2.82 on a quasi-Hankel matrix is used before we can formulate the quotient manifold $\mathcal{M}$ that removes the group action of orthogonal matrices. Since the group of orthogonal matrices contains the set of permutation matrices, optimization over $\mathcal{M}$ does not suffer permutation-related non-identifiability issues.

The proposed numerical optimization framework in this chapter is compared to a conventional CP decomposition scheme called the *alternating least-squares* in Section 4.7.2.2. This method is based on an alternating optimization scheme, and its application to both the CP and Tucker decomposition problem for rectangular tensors can be found in the survey article [33]. We use the version of the alternating least-squares algorithm from [80, 81] for our demonstrations because it is widely used in the tensor decomposition literature. If the given tensor J is an order-L, D-dimensional, symmetric tensor with a CP rank of p, and its component-form in some basis is denoted by $\mathtt{J}$, then the alternating least-squares optimization problem is equivalent to the following:

$$\min_{\{\lambda_i \in \mathbb{R},\, z_i \in \mathbb{R}^D_{\mathrm{s}}\}_{i=1}^{p}} \left(\sum_{i=1}^{p} \lambda_i z_i^{\otimes L} - \mathtt{J} \right) \circ_{\mathrm{F}} \left(\sum_{i=1}^{p} \lambda_i z_i^{\otimes L} - \mathtt{J} \right),$$

where $\circ_{\mathrm{F}}$ is the Frobenius product between tensors here, which is the sum of the per-element multiplication of the components of two tensors under some basis. This decomposition method requires the user to know the correct CP rank of J before performing the decomposition. A symmetric tensor decomposition method based on a gradient-based optimization approach was presented in [32]. That work used a similar cost function with an additive regularization term, and was solved using a gradient algorithm instead of an alternating least-square algorithm. Their result is only applicable to the CP tensor decomposition problem. The work done in this chapter is applicable to any differentiable, real-valued cost function, but we are limited to symmetric tensors that correspond to the space $\mathcal{S}_{\mathrm{E}}$.

4.4 Setup and Notations

In this section, we briefly discuss the notation, setup, and identities used throughout this chapter. The technical topics here are reviewed in greater detail in Sections 2.13, 2.14, and 4.1.

- D, L, and p are given finite positive integers. L is even.
- Every symmetric tensor can be mapped to a sum of powers of linear forms [30, 29]. This means that we can write any $T \in \mathcal{S}(D,\, L)$ as

$$T = \sum_{j=1}^{p} \lambda_j \left(u^{(j)}\right)^{\otimes L},\ u^{(j)} \in \mathcal{J}(D,\, 1),\, \forall j \in [p]. \tag{4.5}$$

  for some rank-one tensors that can be constructed in the following way (see Example 2.28):

$$\left(u^{(j)}\right)^{\otimes L} = \underbrace{u^{(j)} \otimes u^{(j)} \otimes \cdots \otimes u^{(j)}}_{L \text{ times}}. \tag{4.6}$$

If the ith component of $u^{(j)}$ with respect to the elementary basis of $\mathbb{R}^D$ is denoted by $\left(u^{(j)}\right)_i$, $\forall i \in [D]$, then

$$\left(\left(u^{(j)}\right)^{\otimes L}\right)_{\mathcal{B}^{\otimes L},\, a} = \left(u^{(j)}\right)_{a_1}\left(u^{(j)}\right)_{a_2}\cdots\left(u^{(j)}\right)_{a_L} = \prod_{l=1}^{L}\left(u^{(j)}\right)_{a_l},\ \forall a \in [D]^L.$$

Keep in mind that here, we assume that u is specified such that $u_1 \neq 0$, otherwise we re-label our dimensions so that this is true; see the discussion surrounding Equation 2.89 for more details. We discuss this issue later in this chapter.

- There are $m = \binom{L+D-1}{L}$ elements in $\mathbb{A}(D, L)$. We do not consider the case when $m < p$ in this thesis, because we can always choose a larger L so that $m > p$.

- The associated homogeneous polynomial of T from Equation 4.5 is given by

$$g(x) = \sum_{j=1}^{p}\lambda_j\left(\sum_{d=1}^{D}u_d^{(j)}x_d\right)^L,$$

for some $\lambda_j \in \mathbb{R}$, $\forall j \in [p]$, and some $u_d \in \mathbb{R}$, $\forall d \in [D]$. Here, x_d is a real-valued variable for a D-variate polynomial. The dehomogenized version of g where $x_1 = 1$ is

$$\boldsymbol{g}(x_{2:D}) := \sum_{j=1}^{p}c_j\left(1+\sum_{d=2}^{D}z_d^{(j)}x_d\right)^L, \tag{4.7}$$

where the polynomial coefficients $\mathfrak{z} := \left\{z^{(j)} \in \mathbb{R}_{\mathrm{s}}^D \mid \left(z^{(j)}\right)_1 = 1\right\}_{j=1}^{p}$ and the scale factors array $c \in \mathbb{R}_{\mathrm{s}}^p$ are defined as

$$z_d^{(j)} := \frac{u_d^{(j)}}{u_1^{(j)}},\ \forall d \in [D],\ \forall j \in [p],$$

and

$$c_j := \lambda_j\left(u_1^{(j)}\right)^L,\ \forall d \in [D].$$

It follows that

$$\left(z^{(j)}\right)^a := \prod_{d=1}^{D}\left(z_d^{(j)}\right)^{a_d} \tag{4.8}$$

$$= \prod_{d=2}^{D}\left(z_d^{(j)}\right)^{a_d} \tag{4.9}$$

for any multi-index $a \in [L]^D$, since $\left(z^{(j)}\right)_1 = 1$. The products in Equations 4.8 and 4.9 are both used to represent $\left(z^{(j)}\right)^a$ throughout this chapter.

- We denote by $(\mathfrak{z},\, c)$ a rank-one decomposition of a dehomogenized polynomial.

- The quasi-Hankel matrix of $\mathfrak{z}$ and c under the ordered set of monomial exponents $\mathbb{A}(D,\, L)_{\succ_{\mathrm{V}}}$ (see Section 2.13 with $\mathbb{B}_{\succ} \leftarrow \mathbb{A}(D,\, L)_{\succ_{\mathrm{V}}}$) is specified in an entry-wise fashion (see Definition 2.115): $\forall \beta^{(i)},\, \beta^{(j)} \in \mathbb{A}(D,\, L)$,

$$\left(\mathtt{H}\left(\mathbb{A}(D,\, L)_{\succ_{\mathrm{V}}},\, \mathfrak{z},\, c\right)\right)_{i,\,j} = \mathfrak{h}\left(\beta^{(i)}+\beta^{(j)},\, \mathfrak{z},\, c\right)$$

$$= \sum_{n=1}^{p}c_n\left(z^{(n)}\right)^{\beta^{(i)}}\left(z^{(n)}\right)^{\beta^{(j)}}. \tag{4.10}$$

This is an $m \times m$ symmetric matrix. The first row or column of this matrix corresponds to the distinct components of a symmetric tensor T, with the ordering of these components determined by the monomial exponent multi-index that follows the ordering $\mathbb{A}(D,\, L)_{\succ_{\mathrm{V}}}$.

- The quasi-Vandermonde matrix of $\mathfrak{z}$ is specified in an entry-wise fashion (see Definition 2.116):

$$\Big(\mathtt{V}\Big(\mathbb{A}(D,\, L)_{\succ_{\mathrm{V}}},\, \mathfrak{z}\Big)\Big)_{i,\,j} := \big(z^{(j)}\big)^{\alpha^{(i)}} = \prod_{d=1}^{K} \big(z^{(j)}\big)_d^{\left(\alpha^{(i)}\right)_d}, \tag{4.11}$$

where $\alpha^{(i)}$ is the ith element in $\mathbb{A}(D,\, L)_{\succ_{\mathrm{V}}}$. We have

$$\mathtt{H}\Big(\mathbb{A}(D,\, L)_{\succ_{\mathrm{V}}},\, \mathfrak{z},\, c\Big) = \mathtt{V}\Big(\mathbb{A}(D,\, L)_{\succ_{\mathrm{V}}},\, \mathfrak{z}\Big)\mathrm{diagm}(c)\mathtt{V}\Big(\mathbb{A}(D,\, L)_{\succ_{\mathrm{V}}},\, \mathfrak{z}\Big)^T.$$

We denote by $\mathbb{V}(D,\, L,\, p)$ the following space of quasi-Vandermonde matrices:

$$\mathbb{V}(D,\, L,\, p) = \Big\{\mathtt{V}\Big(\mathbb{A}(D,\, L)_{\succ_{\mathrm{V}}},\, \mathfrak{z}\Big) \;\mid\; \mathfrak{z} \text{ is the first slot of an element from } \mathcal{S}_{\mathrm{E}}(D,\, L,\, p)\Big\} \tag{4.12}$$

- To reduce notation clutter, let us use the following short-hand notation when c is clear from context, or unimportant to the discussion at hand:

$$\bar{\mathtt{C}} := \mathrm{diagm}\Big(\big\{\sqrt{c_j}\big\}_{j=1}^{p}\Big). \tag{4.13}$$

We also use $\bar{c}_j := \sqrt{c_j}$.

For the rest of this chapter, we continue to use the symbols and notations reviewed above, unless specified otherwise. We also write $\mathtt{H}$ to mean $\mathtt{H}\Big(\mathbb{A}(D,\, L)_{\succ_{\mathrm{V}}},\, \mathfrak{z},\, c\Big)$, $\mathtt{V}$ to mean $\mathtt{V}\Big(\mathbb{A}(D,\, L)_{\succ_{\mathrm{V}}},\, \mathfrak{z}\Big)$, and $\mathtt{C}$ to mean $\mathrm{diagm}(c)$ when $\mathbb{A}(D,\, L)_{\succ_{\mathrm{V}}}$, $\mathfrak{z}$, and c are clear from context, or unimportant to the discussion at hand. We use the identity chart for all our matrix manifolds and the elementary basis for components of a tensor or matrix, unless otherwise specified.

4.5 The Proposed Manifolds

We are interested in the space of quasi-Hankel matrices that meet the conditions of Proposition 4.1. The first row of a matrix from such a space is isomorphic to the space S_{E}, which in turn is isomorphic to the space of even-ordered symmetric tensors that admit a CP decomposition to $(u,\, \lambda)$, where $\forall i \in [p]$, $\lambda_i \in \mathbb{R}^p_{++}$. This corresponds to $c \in \mathbb{R}^p_{++,\,\mathrm{s}}$ in the dehomogenized parameterization. We can factorize a positive-definite quasi-Hankel matrix $\mathtt{H}$ as

$$\begin{aligned} \mathtt{H} &= \mathtt{V}\mathtt{C}\mathtt{V}^T \\ &= \mathtt{V}\bar{\mathtt{C}}\bar{\mathtt{C}}\mathtt{V}^T \\ &= AA^T \end{aligned} \tag{4.14}$$

This parameterization of $\mathtt{H}$ works with a point A from the set

$$\mathcal{V} := \Big\{\mathtt{V}\Big(\mathbb{A}(D,\, L)_{\succ_{\mathrm{V}}},\, \mathfrak{z}\Big)\mathrm{diagm}(\bar{c}) \in \mathbb{R}^{m\times p}_{\mathrm{s}} \;\mid\; (\mathfrak{z}\,, c) \in \mathcal{S}_{\mathrm{E}},\, \bar{c}_j = \sqrt{c_j},\, \forall j \in [p]\Big\}, \tag{4.15}$$

$$\cong \mathbb{V}(D,\, L,\, p) \times \mathbb{D}^p_{++} \tag{4.16}$$

where $\mathbb{V}(D,\, L,\, p)$ is the space of quasi-Vandermonde matrices, and $\mathbb{D}^p_{++}$ is the space of $p \times p$ diagonal matrices with strictly positive diagonals. To recover a quasi-Vandermonde matrix $\mathtt{V}$ and $\bar{\mathtt{C}}$, we set the first row of $A = \mathtt{V}\bar{\mathtt{C}}$ to get the diagonals of $\bar{\mathtt{C}}$, and divide every entry of the jth column of $\mathtt{V}\bar{\mathtt{C}}$ by $\bar{\mathtt{C}}_{j,\,j}$to get $\mathtt{V}$, for each $j \in [p]$. Note that the map $A \mapsto AA^T$ is not a surjective map to the space of all quasi-Hankel matrices, since we require the entries of c to be strictly positive. This means that our solution space consists of only symmetric tensors that can be decomposed into rank-one tensors that have a strictly positive weight c_j, which requires L to be even and λ_i to be strictly positive.

4.5.1 The Manifold $\mathcal{V}$

We now show that $\mathcal{V}$ can be treated as a differentiable manifold. Consider the subset of full-rank matrices

$$\mathcal{Z}(D,\, p) := \left\{A \in \mathbb{R}^{D\times p}_{\text{full, s}} \;\middle|\; A_{1,\,j} = 1,\, \forall j \in [p]\right\}. \tag{4.17}$$

It follows from Theorem 2.98 that $\mathcal{Z}(D,\, p)$ is a $(D-1)p$-dimensional, embedded submanifold of $\mathbb{R}^{D\times p}_{\text{full, s}}$. Given a $L \in \mathbb{Z}_{++}$, let $m = \binom{L+D-1}{L}$. Consider the following map:

$$\begin{aligned} \omega_{\mathcal{Z}} : \mathcal{Z}(D,\, p) &\to \mathbb{R}^{(m-D)\times p}_{\text{s}} \\ A &\mapsto \omega_{\mathcal{Z}}(A), \\ (\omega_{\mathcal{Z}}(A))_{i,\,j} &:= \prod_{d=1}^{D} A_{d,\,j}^{\alpha_d^{(D+i)}}, \end{aligned} \tag{4.18}$$

where $\alpha^{(i)}$ is the ith element in $\mathbb{A}(D,\, L)_{\succ \text{V}}$. Note that $\omega_{\mathcal{Z}}(A)$ consists of the last $m-D$ rows of a $m\times p$ quasi-Vandermonde matrix, and A consists of the first D rows of that quasi-Vandermonde matrix. By Proposition 2.96, the set

$$\mathcal{G}(\omega_{\mathcal{Z}}) = \left\{(A,\, \omega_{\mathcal{Z}}(A)) \in \mathcal{Z}(D,\, p) \times \mathbb{R}^{(m-D)\times p}_{\text{s}}\right\} \cong \mathbb{V}(D,\, L,\, p)$$

is an embedded, $(D-1)p$-dimensional, submanifold of $\mathcal{Z}(D,\, p) \times \mathbb{R}^{(m-D)\times p}_{\text{s}}$. Since

$$\mathcal{Z}(D,\, p) \times \mathbb{R}^{(m-D)\times p}_{\text{s}} \subset \mathbb{R}^{D\times p} \times \mathbb{R}^{(m-D)\times p} \cong \mathbb{R}^{m\times p}_{\text{s}}$$

and $\mathcal{G}(\omega_{\mathcal{Z}}) \cong \mathbb{V}(D,\, L,\, p)$, we see that the space of all quasi-Vandermonde matrices $\mathbb{V}(D,\, L,\, p)$ is a $(D-1)p$-dimensional embedded submanifold of $\mathbb{R}^{m\times p}_{\text{s}}$. Since the left-hand side of

$$\mathbb{V}(D,\, L,\, p) \times \mathbb{D}^p_{++} \cong \mathcal{V} \subset \mathbb{R}^{m\times p}_{\text{s}}$$

is a product manifold, we can treat $\mathcal{V}$ as an embedded submanifold of dimension Dp of $\mathbb{R}^{m\times p}_{\text{s}}$, where the dimension is so because $\dim \mathbb{V}(D,\, L,\, p) = (D-1)p$ and $\dim \mathbb{R}^p_{++} = p$.

We now derive an orthogonal projector onto the tangent space of $\mathcal{V}$. For any $A \in \mathcal{V}$, we have

$$A = \begin{bmatrix} \bar{c}_1 & \bar{c}_2 & \cdots & \bar{c}_p \\ \bar{c}_1\mathtt{V}_{2,\,1} & \bar{c}_2\mathtt{V}_{2,\,2} & \cdots & \bar{c}_p\mathtt{V}_{2,\,p} \\ \bar{c}_1\mathtt{V}_{3,\,1} & \bar{c}_2\mathtt{V}_{3,\,2} & \cdots & \bar{c}_p\mathtt{V}_{3,\,p} \\ \vdots & \vdots & \ddots & \vdots \\ \bar{c}_1\mathtt{V}_{m,\,1} & \bar{c}_2\mathtt{V}_{m,\,2} & \cdots & \bar{c}_p\mathtt{V}_{m,\,p} \end{bmatrix}, \tag{4.19}$$

$$= \begin{bmatrix} \bar{c}_1 & \bar{c}_2 & \cdots & \bar{c}_p \\ \bar{c}_1\left(z^{(1)}\right)_2 & \bar{c}_2\left(z^{(2)}\right)_2 & \cdots & \bar{c}_p\left(z^{(p)}\right)_2 \\ \bar{c}_1\left(z^{(1)}\right)_3 & \bar{c}_2\left(z^{(2)}\right)_3 & \cdots & \bar{c}_p\left(z^{(p)}\right)_3 \\ \vdots & \vdots & \ddots & \vdots \\ \bar{c}_1\left(z^{(1)}\right)_D & \bar{c}_2\left(z^{(2)}\right)_D & \cdots & \bar{c}_p\left(z^{(p)}\right)_D \end{bmatrix} \tag{4.20}$$

for some $\mathtt{V} \in \mathbb{V}(D,\, L,\, p)$ and some $\bar{c} \in \mathbb{R}^p_{++} \cong \mathbb{D}^p_{++}$. Note that $\mathtt{V}_{1,\,j} = 1,\, \forall j \in [p]$. Let us define a set that is isomorphic to $\mathcal{Z}$ that is easier to work with in our upcoming discussion:

$$\mathfrak{Z} := \left\{ \left\{ z^{(j)} \in \mathbb{R}^D_{\mathrm{s}} \right\}_{j=1}^{p} \right\}, \tag{4.21}$$

where

$$\left(z^{(j)}\right)_1 = 1,\, z^{(j)} \neq b z^{(i)},\, \forall i,\, j \in [p],\, \forall b \in \mathbb{R}.$$

We can interpret this space as a space of dehomogenized polynomial roots. The entry-wise formula for A is

$$A_{i,\,j} = \bar{c}_j \cdot \left(z^{(j)}\right)^{\alpha^{(i)}} \tag{4.22}$$

for some $\mathfrak{z} := \left\{ z^{(j)} \right\}_{j=1}^{p} \in \mathfrak{Z}$.

To find the orthogonal projector onto

$$T_A\mathcal{V} \cong \mathbb{R}^{Dp} \subset \mathbb{R}^{mp} \cong T_A\mathbb{R}^{m\times p}_{\mathrm{s}},$$

our strategy is to identify a set of linearly independent vectors as a basis for the vector subspace $T_A\mathcal{V} \subset T_A\mathbb{R}^{m\times p}_{\mathrm{s}}$, orthogonalize it with respect to the dot product, then apply the orthogonal projector equation. The dot product is used here since we treat the vector space $\mathbb{R}^{m\times p}_{\mathrm{s}}$ as a Riemannian manifold with the Euclidean metric in this thesis. Recall from Definition 2.67 that tangent vectors can be interpreted as a set of curves on the manifold. For the remainder of this discussion, let us use the short-hand notations $z := z^{(j)}$, $\dot{z} := \frac{dz(t)}{dt}$, $\dot{\bar{c}} := \frac{d\bar{c}(t)}{dt}$, and $\alpha := \alpha^{(i)}$ when we have a given $i \in [m]$ and $j \in [p]$. Here, $\alpha^{(i)}$ is the ith element in $\mathbb{A}(D,\, L)_{\succ_{\mathrm{V}}}$. A valid tangent vector in $T_A\mathbb{R}^{m\times p}_{\mathrm{s}}$ is a derivative of $A_{i,\,j}$ with respect to a curve parameter t that is evaluated at 0. We drop the notation for evaluation at $t = 0$ to reduce clutter in our presentation. Here, $A_{i,\,j}(t)$ denotes a curve on $\mathbb{R}^{m\times p}_{\mathrm{s}}$ that passes through $A_{i,\,j} \in \mathbb{R}$ at $t = 0$, $\bar{c}(t)$ denotes a curve on $\mathbb{R}^p_{++}$ that passes through $\bar{c}$ at $t = 0$, and $z(t)$ denotes a curve on $\mathbb{R}^D_{\mathrm{s}}$ that passes through z at $t = 0$. The $(i,\, j)$th entry of a tangent vector of A is then

$$\dot{A}_{i,\,j} := \frac{d}{dt} A_{i,\,j}(t) \tag{4.23}$$

$$= \dot{\bar{c}}\left(z^{(j)}\right)^{\alpha^{(i)}} + \bar{c} \sum_{d=2}^{D} \left(z^{(j)}\right)^{\alpha^{(i)}}_{-d} \alpha^{(i)}_d \left(\dot{z}^{(j)}\right)_d \tag{4.24}$$

$$= \dot{\bar{c}}(z)^{\alpha} + \bar{c} \sum_{d=2}^{D} z^{\alpha}_{-d} \alpha_d \dot{z}_d \tag{4.25}$$

where we define

$$z^{\alpha}_{-d} := \begin{cases} z_1^{\alpha_1} z_2^{\alpha_2} \cdots z_{i-1}^{\alpha_{d-1}} z_i^{\alpha_d - 1} z_{i+1}^{\alpha_{d+1}} \cdots z_D^{\alpha_D} & \text{if } \alpha_d > 0, \\ 0 & \text{otherwise,} \end{cases}$$

for all $d \in [2 : D]$. It follows that $z^{\alpha}_{-d} z_d \alpha_d = z^{\alpha} \alpha_d,\, \forall \alpha_d \geq 0$. The second line is obtained by taking the total derivative of $A_{i,\,j}$ (see Equation 4.22) over the variables w and $\{z_d\}$. The third line applies our short-hand notation for z.

For a fixed $j \in [p]$, there are D independent derivatives in Equation 4.25 that can act as D elements of a basis $\mathcal{B}$ for $T_A\mathcal{V}$. To see this, consider the collection of D objects $\left\{ y^{(j)}_k \right\}_{k=1}^{D}$, where $y^{(j)}_k := \dot{A}_{k,\,j}$. We use

the short-hand notation $y_k := y_k^{(j)}$ when j is fixed to reduce notation clutter. It follows that if $i = 1$, then $y_1 = \dot{c}$. For a $k \in [2 : D]$, we have

$$\begin{aligned} y_k &= y_1 z_k + \bar{c}\dot{z}_k \\ \implies \bar{c}\dot{z}_k &= y_k - y_1 z_k \\ \implies \dot{z}_k &= \frac{y_k}{\bar{c}} - \frac{y_1 z_k}{\bar{c}}, \forall k \in [D]. \end{aligned}$$

If a $i \in [D]$ is also given, we can use the short-hand $\alpha := \alpha^{(i)}$ to express the derivative from Equation 4.25 as a sum of D independent variables $\{y_k\}$:

$$\begin{aligned} \dot{A}_{i,\,j} &= y_1 z^\alpha + \bar{c}\sum_{d=2}^{D} z_{-d}^{\alpha}\alpha_d \dot{z}_i \\ &= y_1 z^\alpha + \sum_{d=2}^{D} z_{-d}^{\alpha}\alpha_d (y_d - y_1 z_d) \\ &= y_1 z^\alpha + \sum_{k=2}^{D} z_{-k}^{\alpha}\alpha_k y_k - \sum_{d=2}^{D} z^\alpha \alpha_d y_1 \\ &= y_1 \left(z^\alpha - z^\alpha \sum_d \alpha_d \right) + \sum_k z_{-k}^{\alpha}\alpha_k y_k \\ &= y_1 z^\alpha \left(1 - \sum_d \alpha_d \right) + \sum_k z_{-k}^{\alpha}\alpha_k y_k. \end{aligned} \tag{4.26}$$

Note that j is fixed in the above expression.

We now treat each $\dot{A}_{i,\,j}$ as a tangent vector on $T_A\mathbb{R}_{\text{s}}^{m\times p}$ and each $y_k^{(j)}$ as a tangent vector on $T_A\mathcal{V}$. For a given $j \in [p]$, consider the matrix $U^{(j)} \in \mathbb{R}_{\text{s}}^{m\times D}$ that is specified by: $\forall i \in [m]$,

$$\begin{aligned} U_{i,\,1}^{(j)} &:= \left(z^{(j)}\right)^{\alpha^{(i)}} \left(1 - \sum_{d=2}^{D} \alpha_d^{(i)} \right) \\ U_{i,\,k}^{(j)} &:= \left(z^{(j)}\right)_{-k}^{\alpha^{(i)}} \alpha_k^{(i)}, \forall k \in [2 : D]. \end{aligned} \tag{4.27}$$

For the current discussion, we denote by $B \in \mathbb{R}^{mp\times Dp}$ the matrix-form of a basis for the Dp-dimensional subspace with respect to the basis $\left\{y_k^{(j)}\right\}_{k\in[D],\,j\in[p]}$, and it is a block diagonal matrix. A vector in this basis is treated as a vector in the mp-dimensional vector space with respect to the basis $\left\{\dot{A}_{i,\,j}\right\}_{i\in[m],\,j\in[p]}$; i.e., the vector space spanned by $\left\{y_k^{(j)}\right\}_{k\in[D],\,j\in[p]}$ is treated as a Dp-dimensional subspace of a mp-dimensional vector space.

The diagonal blocks of B are given by

$$B_{j(m-1)+1:jm,\,j(D-1)+1:jD} = U^{(j)}. \tag{4.28}$$

Once B is constructed, we apply the Gram-Schmidt procedure to orthonormalize the columns of B with respect to the Euclidean metric; see Algorithm 2.1 for the Gram-Schmidt algorithm used in this thesis. The resultant matrix is denoted by B_{GS}, and its columns correspond to orthonormal vectors that describes the subspace $T_A\mathcal{V} \subset T_A\mathbb{R}_{\text{s}}^{m\times p}$. Let $\mathcal{B}$ be the basis for $T_A\mathcal{P}$ but represented using a chart that maps to a Dp-dimensional subspace of $\mathbb{R}^{mp}$. The orthogonal projector is then

$$\begin{aligned} \text{Proj}_{T_A\mathcal{V}} : T_A\mathbb{R}^{mp} &\xrightarrow{\sim} T_A\mathcal{V} \\ v &\mapsto \text{Proj}_{T_A\mathcal{V}}(v), \forall A \in \mathcal{V}. \end{aligned} \tag{4.29}$$

We have

$$\Big(\text{Proj}_{T_A\mathcal{V}}(v)\Big)_{\mathcal{B}} = B_{\text{GS}}\big(B_{\text{GS}}^T B_{\text{GS}}\big)^{-1} B_{\text{GS}}^T (v)_{\mathcal{B}} = B_{\text{GS}} B_{\text{GS}}^T (v)_{\mathcal{B}}.$$

The last equality holds because $B_{\text{GS}}^T B_{\text{GS}} = I_{Dp}$ due to our Gram-Schmidt post-processing step. Since B_{GS} has more rows than columns, $B_{\text{GS}} B_{\text{GS}}^T \neq I_{mp}$. We can also have

$$\Big(\text{Proj}_{T_A\mathcal{V}}(v)\Big)_{\mathcal{B}} = B\big(B^T B\big)^{-1} B^T (v)_{\mathcal{B}},$$

since the choice of using B_{GS} or B does not theoretically affect the definition of the orthogonal projection matrix. The approach that uses B_{GS} requires a Gram-Schmidt procedure, which is recursive in nature. The approach that uses B requires a matrix inverse. Whichever approach that favours the implementation hardware at hand should be used.

4.5.2 The Manifolds $\mathcal{N}$ and $\mathcal{P}$

The manifold $\mathcal{V}$ is used to project the gradient $\sharp$-isomorphism vector from $T_A\mathbb{R}_{\text{s}}^{m\times p} \cong T_A\mathbb{R}^{mp}$ to $T_A\mathcal{V}$ via Equation 4.29. However, we need to work with a different manifold that is diffeomorphic to $\mathcal{V}$ to define a retraction.

It is simpler to work with the unique portion of $\mathcal{V}$, which are the first D rows of elements from $\mathcal{V}$:

$$\mathcal{P} := \big\{A_{1:D,:} \in \mathbb{R}_{\text{s}}^{D\times p} \mid A \in \mathcal{V}\big\} \cong \mathcal{V}. \tag{4.30}$$

This is an embedded submanifold since it is the Dp-coordinate slice of $\mathcal{V}$. The entries of $\mathcal{P}$ do not separate the product between $\bar{c}_j$ and $\big(z^{(j)}\big)_i$ in Equation 4.19 for any $i \in [2:D]$, $j \in [p]$. We need to separate this product since $\bar{c}_j$ is strictly positive; it is on a different space than $z^{(j)}$.

Consider the map

$$\begin{aligned}\phi_{\mathcal{P},\mathcal{N}} : \mathcal{P} &\to \mathbb{R}_{\text{s}}^{D\times p}\\ A &\mapsto \phi_{\mathcal{P},\mathcal{N}}(A),\end{aligned} \tag{4.31}$$

where for all $i \in [D]$ and $j \in [p]$:

$$\big(\phi_{\mathcal{P},\mathcal{N}}(A)\big)_{i,j} = \begin{cases} A_{1,j} & \text{if } i = 1\\ \frac{A_{i,j}}{A_{1,j}} & \text{otherwise.}\end{cases}$$

We re-use the symbol A here to mean a point on $\mathcal{P}$ as opposed to a point on $\mathcal{V}$. If A is the first D rows of a matrix that has the form of Equation 4.19, i.e.,

$$\begin{aligned}A_{1,j} &= \bar{c}_j\\ A_{i,j} &= \bar{c}_j\big(z^{(j)}\big)_i,\end{aligned} \tag{4.32}$$

then $B = \phi_{\mathcal{P},\mathcal{N}}(A)$ is given by

$$B = \begin{bmatrix} \bar{c}_1 & \bar{c}_2 & \cdots & \bar{c}_p\\ \big(z^{(1)}\big)_2 & \big(z^{(2)}\big)_2 & \cdots & \big(z^{(p)}\big)_2\\ \big(z^{(1)}\big)_3 & \big(z^{(2)}\big)_3 & \cdots & \big(z^{(p)}\big)_3\\ \vdots & \vdots & \ddots & \vdots\\ \big(z^{(1)}\big)_D & \big(z^{(2)}\big)_D & \cdots & \big(z^{(p)}\big)_D\end{bmatrix} \tag{4.33}$$

$$= \begin{bmatrix} \bar{c}_1 & \bar{c}_2 & \cdots & \bar{c}_p\\ \mathtt{V}_{2,1} & \mathtt{V}_{2,2} & \cdots & \mathtt{V}_{2,p}\\ \mathtt{V}_{3,1} & \mathtt{V}_{3,2} & \cdots & \mathtt{V}_{3,p}\\ \vdots & \vdots & \ddots & \vdots\\ \mathtt{V}_{D,1} & \mathtt{V}_{D,2} & \cdots & \mathtt{V}_{D,p}\end{bmatrix}. \tag{4.34}$$

The set

$$\mathcal{N} := \phi_{\mathcal{P},\mathcal{N}}(\mathcal{P}) \subset \mathbb{R}^{(D-1)\times p}_{\text{full, s}} \times \mathbb{R}^{p}_{++,\text{ s}} \tag{4.35}$$

is a Dp-coordinate slice of $\mathbb{R}^{(D-1)\times p}_{\text{full, s}} \times \mathbb{R}^{p}_{\text{s}}$. By Theorem 2.98, it is a Dp-dimensional embedded submanifold of $\mathbb{R}^{(D-1)\times p}_{\text{full, s}} \times \mathbb{R}^{p}_{\text{s}}$.

In this thesis, the retraction from Example 2.87 is used as a retraction for $\mathbb{R}^{(D-1)\times p}_{\text{full, s}}$. I devised two retractions for $\mathbb{R}^{p}_{++,\text{ s}}$. The first is

$$\mathfrak{R}_{\bar{c}_j}(X) = \bar{c}_j e^{\left(\frac{X}{\bar{c}_j}\right)}. \tag{4.36}$$

To verify that this is a first-order retraction, we need to verify the conditions from Definition 2.85. $\mathfrak{R}_{\bar{c}_j}(0) = \bar{c}_j$ clears the first condition. For the second condition, we treat $\mathfrak{R}_{\bar{c}_j} : T_{\bar{c}_j}\mathbb{R}_{++} \to \mathbb{R}_{++}$ as a map between the manifolds $T_{\bar{c}_j}\mathbb{R}_{++}$ and $\mathbb{R}_{++}$. Just for this discussion, let x be the only component of X with respect to the elementary basis $\mathcal{E}$ of $\mathbb{R}$. The differential $d\mathfrak{R}_{\bar{c}_j}$ in component-form is

$$\frac{d\mathfrak{R}_{\bar{c}_j}(x)}{dx} = e^{\left(\frac{x}{\bar{c}_j}\right)}, \tag{4.37}$$

which implies the 1×1 matrix-form of the differential, $\left(\left(d\mathfrak{R}_{\bar{c}_j}\right)_0\right)_{\mathcal{E}}$, is the identity matrix. This clears the second condition. The second retraction is

$$\mathfrak{R}_{\bar{c}_j}(X) = \bar{c}_j e^{\left(-\frac{1}{2}\left(\frac{X}{\bar{c}_j}\right)^2 + \frac{X}{\bar{c}_j}\right)}. \tag{4.38}$$

$\mathfrak{R}_{\bar{c}_j}(0) = \bar{c}_j$ clears the first condition. Using x, we have the differential $d\mathfrak{R}_{\bar{c}_j}$ in component-form as

$$\begin{aligned}\frac{d\mathfrak{R}_{\bar{c}_j}(x)}{dx} &= \bar{c}_j e^{\left(-\frac{1}{2}\left(\frac{x}{\bar{c}_j}\right)^2 + \frac{x}{\bar{c}_j}\right)}\left(-\frac{x}{\bar{c}_j} + \frac{1}{\bar{c}_j}\right) \\ &= e^{\left(-\frac{1}{2}\left(\frac{x}{\bar{c}_j}\right)^2 + \frac{x}{\bar{c}_j}\right)}(1 - x).\end{aligned}$$

which implies the 1×1 matrix-form of the differential, $\left(\left(d\mathfrak{R}_{\bar{c}_j}\right)_0\right)_{\mathcal{E}}$, is the identity matrix. This clears the second condition.

The practical issue of using these two retractions is that the input tangent vector X still needs to be bounded, otherwise the exponential function for real numbers would numerically overflow. In light of this, I simply used the truncation function

$$\tilde{\mathfrak{R}}_{\bar{c}_j}(X) = \begin{cases} \bar{c}_j + X & \text{if } \bar{c}_j + X > \epsilon \\ \epsilon & \text{otherwise,} \end{cases} \tag{4.39}$$

where ϵ is a small positive value to make sure $\bar{c}_j$ is large enough. This is not a retraction, since its range is not an open set. When the function in Equation 4.39 is used in place of a proper retraction such as the ones in Equations 4.36 and 4.38, the proposed numerical optimization framework was more stable. For this reason, the function in Equation 4.39 was used in place of a retraction for $\mathbb{R}_{++,\text{ s}}$ for the $\bar{c}_j$'s in my implementation. A retraction for $\mathcal{P}$ can be obtained by pushing any tangent vectors in $T\mathcal{P}$ to $T\mathcal{N}$ via $d\phi_{\mathcal{P},\mathcal{N}}$, then apply a retraction on $\mathcal{N}$.

4.5.3 Metrics for $\mathcal{P}$ and $\mathcal{N}$

In Section 4.5.1, we identified a method to project vectors from a tangent space of $\mathbb{R}^{m\times p}_{\text{col, s}}$ to a tangent space on $\mathcal{V}$. In Section 4.5.2, we discussed retractions on $\mathcal{N} \cong \mathcal{P} \cong \mathcal{V}$. In this subsection, we pull-back the Euclidean metric on $T\mathbb{R}^{m\times p}_{\text{col, s}}$ to $T\mathcal{P}$. The pull-back metric was introduced in Equation 2.53. The use of the Euclidean metric on $T\mathbb{R}^{m\times p}_{\text{col, s}}$ induces the following behaviours:

1. At any given point on $\mathbb{R}^{m\times p}_{\text{col, s}}$, the gradient $\sharp$-isomorphism has the same components as the gradient covector.

2. The Riemannian gradient $\sharp$-isomorphism points in the direction of most increase out of all unit vectors that use the induced Riemannian norm [1]. The Euclidean metric is a constant Riemannian metric on $\mathbb{R}^{m\times p}_{\text{col, s}}$, which means that the induced Riemannian norm is unaffected by where one is at in $\mathbb{R}^{m\times p}_{\text{col, s}}$.

To perform a change-of-coordinates from a vector $v \in T_{A_{1:D,:}}\mathcal{P}$ to its corresponding component-form representation in $T_A\mathcal{V}$ with respect to the basis from Equation 4.27, one can pre-multiply the components of v by the block-diagonal matrix B from Equation 4.28; see Section 2.4 for a review of this linear algebra property.

We use the independent variables $\{y_k\}_{k=1}^{D}$ from Equation 4.26 to construct our chosen set of basis for $T_{A_{1:D,:}}\mathcal{P}$. Similar to our development of the block-diagonal B from Equation 4.28, we can store the component-form of our chosen basis for $T_{A_{1:D,:}}\mathcal{P}$ as a block-diagonal matrix $B_{\mathcal{P}}$. The diagonal blocks of B are given by

$$B_{j(m-1)+1:jm,\, j(D-1)+1:jD} = U^{(j)}, \tag{4.40}$$

where $U^{(j)}$is given in Equation 4.27.

To describe the differential that is used in this pull-back metric, we first push-forward an input $X \in T_A\mathcal{P}$ via $d\phi_{\mathcal{P},\mathcal{N}}$ to $T_{\phi_{\mathcal{P},\mathcal{N}}(A)}\mathcal{N}$, then push-forward to $T_{\phi_{\mathcal{N},\mathcal{V}}(A)}\mathcal{V}$. We define $\phi_{\mathcal{N},\mathcal{V}}$ after we present $d\phi_{\mathcal{P},\mathcal{N}}$.

Let A be a point in $\mathcal{P}$. The differential $d\phi_{\mathcal{P},\mathcal{N}}$ requires us to work with

$$\mathbb{R}^{D\times p} \cong \mathbb{R}^{Dp} \cong T_A\mathcal{P} \cong T_{\phi_{\mathcal{P},\mathcal{N}}(A)}\mathcal{N},$$

and we use the column-major ordering that takes an element from $\mathbb{R}^{D\times p}$ to $\mathbb{R}^{Dp}$. Let $\boldsymbol{A} := \textbf{colmaj}(A)$ and $\boldsymbol{B} := \textbf{colmaj}(\phi_{\mathcal{P},\mathcal{N}}(A))$, and we treat $\boldsymbol{B}$ as a function of $\boldsymbol{A}$. Since we're using their respective identity charts for both $\mathcal{N}$ and $\mathcal{P}$, the differential $d\phi_{\mathcal{P},\mathcal{N}}$ is the Jacobian matrix of $\boldsymbol{B}$. We denote this Jacobian matrix by $J_{\mathcal{P},\mathcal{N}} := \left(d(\phi_{\mathcal{P},\mathcal{N}})_A\right)_{\mathcal{B}\otimes\mathcal{B}^*}$, where $\mathcal{B}$ is some basis for $\mathbb{R}^{Dp}$ and $\mathcal{B}^*$ is the dual basis for $(\mathbb{R}^{Dp})^*$. We have

$$\begin{aligned}(J_{\mathcal{P},\mathcal{N}})_{a,\,b} &:= \frac{\partial \boldsymbol{B}_a}{\partial \boldsymbol{A}_b} \\ &= \frac{\partial B_{k,\,l}}{\partial A_{i,\,j}},\end{aligned}$$

where $(k,\, l) \in [D] \times [p]$ is the multi-index that corresponds to the column-major ordered (linearized) index $a \in [Dp]$, and similarly for $(i,\, j) \in [D] \times [p]$ and $b \in [Dp]$. Using Equations 4.32, 4.31, and 4.33, we have the following derivatives:

$$\begin{aligned}\frac{\partial B_{1,\,j}}{\partial A_{1,\,j}} &= \frac{\partial \bar{c}_j}{\partial \bar{c}_j} = 1,\ \forall j \in [p], \\ \frac{\partial B_{i,\,j}}{\partial A_{i,\,j}} &= \frac{\partial\left(\frac{A_{i,\,j}}{\bar{c}_j}\right)}{\partial A_{i,\,j}} = \frac{1}{\bar{c}_j},\ \forall i \in [2:D], \\ \frac{\partial B_{k,\,j}}{\partial A_{1,\,j}} &= \frac{\partial\left(\frac{A_{k,\,j}}{\bar{c}_j}\right)}{\partial \bar{c}_j} = \frac{A_{k,\,j}}{\bar{c}_j^2},\ \forall k \in [2:D],\end{aligned}$$

and all other entries of $J_{\mathcal{P},\mathcal{N}}$ are zero. It follows that the matrix-form of $J_{\mathcal{P},\mathcal{N}}$ that is indexed by $(a,\, b)$ is a block-diagonal matrix, and the non-zero entries of each block are given by the above expressions.

The map from $\mathcal{N}$ to $\mathcal{V}$

$$\begin{aligned}\phi_{\mathcal{N},\mathcal{V}} : \mathcal{N} &\to \mathcal{V} \\ B &\mapsto \phi_{\mathcal{N},\mathcal{V}}(B),\end{aligned} \tag{4.41}$$

is defined entry-wise as

$$\left(\phi_{\mathcal{N},\mathcal{V}}(B)\right)_{i,\,j} := B_{1,\,j}\prod_{d=1}^{D} B_{d,\,j}^{\alpha_d^{(i)}},\ \forall i \in [m]$$

which becomes

$$\left(\phi_{\mathcal{N},\mathcal{V}}(B)\right)_{i,\,j} := \bar{c}_j\prod_{d=1}^{D}\left(z^{(j)}\right)_d^{\alpha_d^{(i)}} \tag{4.42}$$

if B is of the form in Equation 4.33. This map is very similar to the map $\omega_{\mathcal{Z}}$ from Equation 4.18 that we used to show that quasi-Vandermonde matrices are a submanifold, different by a factor of $\bar{c}_j$. Similar to before, the differential $d\phi_{\mathcal{N},\mathcal{V}}$ requires us to work with $\mathbb{R}^{Dp} \cong T_B\mathcal{N}$ and $\mathbb{R}^{mp} \cong T_{\phi_{\mathcal{N},\mathcal{V}}(B)}\mathcal{V}$. We use the column-major ordering identity chart for $\mathcal{P}$ and $\mathcal{V}$. Let $\boldsymbol{B} := \textbf{colmaj}(B)$ and $\boldsymbol{C} := \textbf{colmaj}(\phi_{\mathcal{N},\mathcal{V}}(B))$, and we treat $\boldsymbol{C}$ as a function of $\boldsymbol{B}$. The differential is the Jacobian matrix of $\boldsymbol{C}$, and we denote it by $J_{\mathcal{N},\mathcal{V}} := \left(d(\phi_{\mathcal{N},\mathcal{V}})_A\right)_{\mathcal{U}\otimes\mathcal{B}^*}$; $\mathcal{U}$ is some basis for $\mathbb{R}^{mp}$. We have

$$\begin{aligned}\left(J_{\mathcal{N},\mathcal{V}}\right)_{a,\,b} &:= \frac{\partial \boldsymbol{C}_a}{\partial \boldsymbol{B}_b} \\ &= \frac{\partial C_{k,\,l}}{\partial B_{i,\,j}},\end{aligned}$$

where $(k,\, l) \in [m] \times [p]$ is the multi-index that corresponds to the column-major ordered (linearized) index $a \in [mp]$, and similarly for $(i,\, j) \in [D] \times [p]$ and $b \in [Dp]$. Using Equations 4.42 and 4.33, we have the following derivatives:

$$\begin{aligned}\frac{\partial C_{1,\,j}}{\partial B_{1,\,j}} &= \frac{\partial \bar{c}_j}{\partial \bar{c}_j} = 1,\ \forall j \in [p], \\ \frac{\partial C_{k,\,j}}{\partial B_{1,\,j}} &= \frac{\partial C_{k,\,j}}{\partial \bar{c}_j} = \prod_{d=1}^{D}\left(z^{(j)}\right)_d^{\alpha_d^{(i)}},\ \forall k \in [2:m], \\ \frac{\partial C_{k,\,j}}{\partial B_{i,\,j}} &= \frac{\partial C_{k,\,j}}{\partial \left(z^{(j)}\right)_i} = \alpha_i^{(k)}\left(z^{(j)}\right)_i^{\alpha_i^{(k)}-1}\bar{c}_j\prod_{d\neq i}^{D}\left(z^{(j)}\right)_d^{\alpha_d^{(k)}},\ \forall i \in [2:D],\end{aligned}$$

and all other entries of $J_{\mathcal{P},\mathcal{V}}$ are zero. It follows that the matrix-form of $J_{\mathcal{P},\mathcal{V}}$ that is indexed by $(a,\, b)$ is a block-diagonal matrix, and the non-zero entries of each block are given by the above expressions.

The differential from $T_A\mathcal{P}$ to $T_{\phi_{\mathcal{N},\mathcal{V}}\circ\phi_{\mathcal{P},\mathcal{N}}(A)}\mathcal{V}$ is

$$d(\phi_{\mathcal{N},\mathcal{V}} \circ \phi_{\mathcal{P},\mathcal{N}})_A = d(\phi_{\mathcal{N},\mathcal{V}})_{\phi_{\mathcal{P},\mathcal{N}}(A)} \circ d(\phi_{\mathcal{P},\mathcal{N}})_A,$$

and its matrix-form using the charts and basis we discussed is the matrix product

$$\left(\textbf{colmaj}^{-1} \circ d(\phi_{\mathcal{N},\mathcal{V}} \circ \phi_{\mathcal{P},\mathcal{N}})_A \circ \textbf{colmaj}\right)_{\mathcal{U}\otimes\mathcal{B}^*} = J_{\mathcal{N},\mathcal{V}}J_{\mathcal{P},\mathcal{N}}. \tag{4.43}$$

For any given $X,\, Y \in T_A\mathcal{P}$ at any $A \in \mathcal{P}$, we can compute the pull-back metric $G_{\mathcal{P}}$ for $T\mathcal{P}$ by

$$\begin{aligned}G_{\mathcal{P},\,A}(X,\, Y) &:= \left((\phi_{\mathcal{N},\mathcal{V}} \circ \phi_{\mathcal{P},\mathcal{N}})^* G_{\text{Euclidean}}\right)_A(X,\, Y) &(4.44)\\ &= J_{\mathcal{N},\mathcal{V}}J_{\mathcal{P},\mathcal{N}}\textbf{colmaj}((X)_{\mathcal{B}}) \circ_{\text{F}} J_{\mathcal{N},\mathcal{V}}J_{\mathcal{P},\mathcal{N}}\textbf{colmaj}((Y)_{\mathcal{B}}) &(4.45)\end{aligned}$$

where $G_{\text{Euclidean}}$ is the Euclidean metric on $T\mathcal{V}$, and $\circ_{\text{F}}$ here is the Frobenius product between two matrices. It follows that the metric tensor at the point $A \in \mathcal{P}$, in matrix-form with respect to $\mathcal{B}^* \otimes \mathcal{B}^*$, is

$$\left(\left((\phi_{\mathcal{N},\mathcal{V}} \circ \phi_{\mathcal{P},\mathcal{N}})^* G_{\text{Euclidean}}\right)_A\right)_{\mathcal{B}^* \otimes \mathcal{B}^*} := \left(J_{\mathcal{N},\mathcal{V}} J_{\mathcal{P},\mathcal{N}}\right)^T J_{\mathcal{N},\mathcal{V}} J_{\mathcal{P},\mathcal{N}}. \tag{4.46}$$

One simply remove $J_{\mathcal{P},\mathcal{N}}$ from the above equation to get the pull-back metric for $\mathcal{N}$.

4.5.4 The Quotient Manifold $\mathcal{M}$

Optimization problems over any manifold that is isomorphic to $\mathcal{V}$ suffer from non-identifiability issues that arise from any permutation on the columns of an optimal solution $A_{\text{opt}} \in \mathcal{P}$. This follows from the fact that the factor in Equation 4.14 is unique up to a multiplication of a orthogonal matrix:

$$\begin{aligned} \mathtt{H} &= \mathtt{V}\mathtt{C}\mathtt{V}^T \\ &= \mathtt{V}\bar{\mathtt{C}}QQ^T\bar{\mathtt{C}}\mathtt{V}^T \\ &= BB^T, \end{aligned} \tag{4.47}$$

where $B = \mathtt{V}\bar{\mathtt{C}}Q$ and $Q \in \mathbb{O}(p)$ is any orthogonal matrix of size $p \times p$; i.e., $Q^TQ = QQ^T = I_p$. This parameterization of $\mathtt{H}$ works with a point from the space

$$\mathcal{W} := \left\{AQ \in \mathbb{R}_{\text{s}}^{m\times p} \mid A \in \mathcal{V},\, Q \in \mathbb{O}(p)\right\}. \tag{4.48}$$

The space of permutation matrices of size $p \times p$ are contained in $\mathbb{O}(p)$, so a point from $[B]_\sim \in \mathcal{W}/\mathbb{O}(p)$ should not suffer from permutation-related non-identifiability issues. We need to work with $\mathcal{W}$ instead of $\mathcal{V}$ since the set of permutation matrices do not form a Lie group, but $\mathbb{O}(p)$ is a Lie group. In this subsection, we derive a quotient manifold from $\mathcal{W}$, and a horizontal projector for this quotient manifold. The background required for this discussion is reviewed in Section 2.11. Here, A is a point on $\mathcal{W}$ or $\mathcal{V} \subset \mathcal{W}$, which is a full column rank matrix of size $m \times p$.

The space $\mathcal{W}$ is in $\mathbb{R}_{\text{col, s}}^{m\times p}$ since for any $A \in \mathcal{V}$ and any $Q \in \mathbb{O}(p)$; AQ is just an orthogonal change-of-basis of a full-column matrix. The product manifold $\mathcal{V} \times \mathbb{O}(p)$ is a differentiable manifold because $\mathbb{O}(p)$ is a Lie group (i.e., a smooth manifold), and we showed $\mathcal{V}$ is a differentiable manifold in Section 4.5.1. Define the matrix multiplication map

$$\begin{aligned} F : \mathcal{V} \times \mathbb{O}(p) &\to \mathbb{R}_{\text{col, s}}^{m\times p} \\ (A,\, Q) &\mapsto AQ, \end{aligned}$$

then we have $\mathcal{W} = F(\mathcal{V} \times \mathbb{O}(p))$. By Proposition 2.94, $\mathcal{W}$ is a differentiable submanifold of $\mathbb{R}_{\text{col, s}}^{m\times p}$. By Theorem 2.108, the space

$$\mathcal{M} := \mathcal{W}/\mathbb{O}(p)$$

is a quotient manifold, and we equip $\mathcal{W}$ with the Euclidean metric. The quotient map is

$$\begin{aligned} \pi : \mathcal{W} &\to \mathcal{M} \\ A &\mapsto \pi(A) := [A]_\sim, \end{aligned}$$

where the equivalence relation $\sim$ is defined as

$$A \sim C \text{ if } \exists Q \in \mathbb{O}(p) \text{ such that } C = AQ. \tag{4.49}$$

It follows that the equivalence class that contains a given $A \in \mathcal{W}$ is

$$[A]_\sim = \{AQ \mid Q \in \mathbb{O}(p)\}.$$

Recall from our review in Section 2.11 on quotient manifolds that, for a given $A_{\mathcal{M}} \in \mathcal{M}$, the vertical tangent subspace in $T_A\mathcal{W}$, with

$$A \in \pi^{-1}(\{A_{\mathcal{M}}\}) = [A]_{\sim} \subset \mathcal{W},$$

is

$$\text{Vert}_A\mathcal{W} := T_A\big(\pi^{-1}(A_{\mathcal{M}})\big).$$

We write $\mathbb{K}(p)$ to mean the space of $p \times p$ skew-symmetric matrices in this thesis. Let us also define the short-hand notation $\dot{Q} := \frac{d}{dt}Q(t)$ for our current discussion. We have

$$\begin{aligned} & Q^TQ = I_p,\ \forall Q \in \mathbb{O}(p) \\ \implies\ & \dot{Q}^TQ + Q^T\dot{Q} = 0 \\ & \dot{Q}^TQ = -Q^T\dot{Q} \in \mathbb{K}(p) \\ \implies\ & \dot{Q} = QK, \end{aligned}$$

for some $K \in \mathbb{K}(p)$. Recall from Section 2.11 that the vertical space contains only tangent vectors that move along the fibre defined by the Lie group action; here, we interpret tangent vectors as differentiable curves. For any $Q \in \mathbb{O}(p)$, it follows that we have

$$\begin{aligned} \text{Vert}_{AQ}\mathcal{W} &= \Big\{A\dot{Q} \mid K \in \mathbb{K}(p)\Big\},\ \forall A \in \mathcal{P} \\ &= \{AQK \mid K \in \mathbb{K}(p)\} \\ \implies \text{Vert}_A\mathcal{W} &= \{AK \mid K \in \mathbb{K}(p)\},\ \forall A \in \mathcal{W} \end{aligned} \tag{4.50}$$

For a given $A \in \mathcal{W}$, recall from Equation 2.63 that the horizontal tangent subspace is defined to be

$$\text{Hor}_A\mathcal{W} := \{Y \in T_A\mathcal{W} \mid H_A(\eta, Y) = 0,\ \forall \eta \in \text{Vert}_A\mathcal{W}\},$$

and we have the Riemannian metric H for $\mathcal{W}$ set to the Euclidean metric here. This means that

$$Y \in \text{Hor}_A\mathcal{W} \iff (X)_{\mathcal{B}} \circ_{\text{F}} (Y)_{\mathcal{B}} = \text{tr}\Big((X)_{\mathcal{B}}^T(Y)_{\mathcal{B}}\Big) = 0,\ \forall X \in \text{Vert}_A\mathcal{W},$$

where $\mathcal{B}$ is some basis of $T_A\mathcal{W}$. If $\mathcal{B}$ is the identity chart-induced tangent basis, then we can write $(X)_{\mathcal{B}} = AK$ for some $K \in \mathbb{K}(p)$ as the matrix-form of any $X \in \text{Vert}_A\mathcal{W}$. The following holds for any $A \in \mathbb{R}^{m\times p}_{\text{full, s}}$, $m \geq p$:

$$\begin{aligned} 0 &= \text{tr}\big(SA^\dagger AK\big),\ \forall S \in \mathbb{S}^p \\ &= \text{tr}\Big(\Big(\big(A^\dagger\big)^T S\Big)^T AK\Big) \\ &= \big(A^\dagger\big)^T S \circ_{\text{F}} AK. \end{aligned}$$

Since we work only with combinations of D, L, and p that satisfy Proposition 4.1, we do not deal with the case when $m < p$. Therefore, we have $Y \in \text{Hor}_A\mathcal{W}$ if $(Y)_{\mathcal{B}} = \big(A^\dagger\big)^T S$ for some $S \in \mathbb{S}^p$. In this thesis, we always use the left pseudoinverse

$$B^\dagger := \big(B^TB\big)^{-1}B^T$$

when B here is any $m \times p$, full column rank matrix that satisfies $m \geq p$. We also sometimes use the short-hand notation

$$B^{\dagger,\,T} := \big(B^\dagger\big)^T$$

to reduce the number of nested round brackets in our presentation.

Since a vector from $\text{Hor}_A\mathcal{W}$ has the form $Y = \left(A^\dagger\right)^T S$ for some $S \in \mathbb{S}^p$, performing the orthogonal projection from $T_A\mathcal{W}$ to $\text{Hor}_A\mathcal{W}$ is equivalent to solving for a unique $S \in \mathbb{S}^p$. The method used here is based on the solution in Lemma 3.2 from [82], and we now describe our adaptation of their strategy. First, we express any tangent vector that can be written as $\eta = AW \in T_A\mathcal{W}$, where W is some $p \times p$ matrix, as a sum of a horizontal vector $Y \in \text{Hor}_A\mathcal{W}$ and a vertical vector $X \in \text{Vert}_A\mathcal{W}$.

$$\begin{aligned} Y &= \eta - X \\ A^{\dagger,T}S &= \eta - AK, \end{aligned} \tag{4.51}$$

where $Y = A^{\dagger,T}S$ for some $S \in \mathbb{S}^p$, and $X = AK$ for some $K \in \mathbb{K}(p)$. Next, we multiply Equation 4.51 by $A^\dagger$ from the left to get

$$\begin{aligned} A^\dagger A^{\dagger,T} S &= A^\dagger\eta - A^\dagger AK \\ \left(A^TA\right)^{-1}A^T\left(A\left(A^TA\right)^{-T}\right)S &= A^\dagger\eta - A^\dagger AK \\ \left(A^TA\right)^{-1}S &= A^\dagger\eta - K \end{aligned}$$

We then add the transpose of the above result to itself to get

$$\begin{aligned} \left(A^TA\right)^{-1}S + S\left(A^TA\right)^{-T} &= A^\dagger\eta - K + \eta^TA^{\dagger,T} - K^T \\ \left(A^TA\right)^{-1}S + S\left(A^TA\right)^{-T} &= A^\dagger\eta + \eta^TA^{\dagger,T}. \end{aligned} \tag{4.52}$$

The generalized continuous-domain Lyapunov equation [83] of a matrix-valued variable U has the form

$$B^TUE + E^TUB = -M,$$

where B, E, and M are given $p \times p$ matrices. If M and U are symmetric $p \times p$ matrices, then the solution for U is unique [83]. With the substitutions $U \leftarrow S$, $E \leftarrow I_p$, $B \leftarrow (A^TA)^{-T}$, and

$$M \leftarrow -\left(A^\dagger\eta + \eta^TA^{\dagger,T}\right),$$

we can use a Lyapunov solver to uniquely determine S. We do not discuss the numerical implementation of such a solver in this thesis, because it comes standard in many present-day numerical linear algebra libraries.

The above computation of S is a direct-application of the strategy from [82]. However, we need to add the following post-processing step to get the horizontal projection $\eta_{\text{hor}} \in \text{Hor}_A\mathcal{W}$ from any given $\eta \in T_A\mathcal{W}$:

$$\left(\eta_{\text{hor}}\right)_{\mathcal{B}} = \text{Proj}_{\text{span}(A)^\perp}\left(\eta - A^{\dagger,T}S\right) + A^{\dagger,T}S \tag{4.53}$$

This step is required because A is a rectangular matrix, so not all vectors in $T_A\mathcal{W}$ can be written as AW, for some $p \times p$ matrix W. This violates the assumption that underpins the approach we just discussed. Since all vectors from $\text{Vert}_A\mathcal{W}$ are in the column span of A, we needed to add the portion of the residual $\eta - A^{\dagger,T}S$ that lies in $\text{span}(A)^\perp$ to get the horizontal projection of η. The projection onto the horizontal space is then given by

$$\begin{aligned} \text{Proj}_{\text{Hor}_A\mathcal{W}} : T_A\mathcal{W} &\xrightarrow{\sim} \text{Hor}_A\mathcal{W} \\ \eta &\mapsto \eta_{\text{hor}}, \end{aligned}$$

where η_{hor} is given by Equation 4.53.

The metric for $T\mathcal{M}$ that we use in this thesis is the metric that is inherited from the Euclidean metric on $T\mathcal{W}$. It follows from Equation 2.65 that this metric is

$$G_{\pi(A)}(X, Y) := \left(X^\uparrow\right)_{\mathcal{B}} \circ_{\text{F}} \left(Y^\uparrow\right)_{\mathcal{B}}, \forall\pi(A) \in M, \forall X, Y \in T_{\pi(A)}\mathcal{M}, \tag{4.54}$$

where $X^{\uparrow} \in \text{Hor}_A\mathcal{W}$ denotes the horizontal lift of X, $\circ_{\text{F}}$ is the Frobenius product between matrices, $\pi(A) := [A]_{\sim}$, and $\mathcal{B}$ is some basis of $\text{Hor}_A\mathcal{W}$.

Suppose now $X \in \Gamma^{\infty}(T\mathcal{M})$ and $B \in \mathcal{M}$. Let us choose $A \in \text{fib}_p\mathcal{W} \cong [A]_{\sim}$ as the representative element of B, and let $X^{\uparrow}_A \in \text{Hor}_A\mathcal{W}$ be the horizontal lift of an arbitrary vector $X_B \in T_B\mathcal{M}$ to the tangent space $\text{Hor}_A\mathcal{W} \subset T_A\mathcal{W}$. Since $\mathcal{W} \to \mathcal{M}$ is a principal-G bundle with the group being $\mathbb{O}(p)$, we have the property

$$X^{\uparrow}_A Q = X^{\uparrow}_{AQ} \in \text{Hor}_{AQ}\mathcal{W} \subset T_{AQ}\mathcal{W},\ \forall Q \in \mathbb{O}(p).$$

This allows us to switch the representation of a tangent vector for a point in $\mathcal{M}$ from $A \in [A]_{\sim}$ to $AQ \in [A]_{\sim}$ in a convenient manner. The main application of this in a gradient-based optimization algorithm is to set the vector field X to be the $\sharp$-isomorphism of the gradient covector field of a cost function $f \in C^{\infty}(\mathcal{M})$. Under this setting, we have

$$\left((df)^{\sharp}\right)^{\uparrow}_A Q = \left((df)^{\sharp}\right)^{\uparrow}_{AQ}. \tag{4.55}$$

This property does not require us to re-compute the gradient nor its horizontal lift if we switch the representative element. I did not pursue an application for this property in this thesis work, but it could be useful when one needs to switch the representative element of $\pi(A) = [A]_{\sim}$ from A to an element in $[A]_{\sim}$ that has better numerical conditioning or some implementation-related advantages. The work [82] exploited a similar trick to achieve computational advantages in the evaluation of their horizontal space projector.

4.5.5 A Bilinear Form for $T\mathcal{P}$

We defined a pull-back metric $G_{\mathcal{P}}$ for $T\mathcal{P}$ from a metric for $T\mathcal{V}$ in Section 4.5.3. We cannot inherit the metric on $T\mathcal{M}$ from Equation 4.54 for $T\mathcal{P}$. This is because $\mathcal{P}$ is diffeomorphic to $\mathcal{V}$, which is a submanifold of $\mathcal{W}$. We cannot pull-back a metric from $T\mathcal{M}$ to $T\mathcal{W}$ since the map from $\mathcal{W}$ to $\mathcal{M}$ is not injective. A similar argument follows for $\mathcal{N}$, since it is also diffeomorphic to $\mathcal{V}$. Since I only devised retractions for $\mathcal{N}$ and $\mathcal{P}$, the proposed optimization framework can only be over these manifolds.

We devise a bilinear form that is not required to be positive-definite to induce an effect similar to the metric in Equation 4.54. The motivation behind this bilinear form is to use it in place of a Riemannian metric, and observe whether it lessens the harmful effects of having permutation-related non-identifiability issues. Define the short-hand $\phi := \phi_{\mathcal{N},\mathcal{V}} \circ \phi_{\mathcal{P},\mathcal{N}}$. Given any $X, Y \in T_A\mathcal{P}$ at any $A \in \mathcal{P}$, we first push-forward these vectors to $T_{\phi(A)}\mathcal{V} \subset T\mathcal{W}$, followed by a projection to $\text{Hor}_{\phi(A)}\mathcal{W}$, and then apply the Euclidean metric on $\text{Hor}_{\phi(A)}\mathcal{W}$. The resultant bilinear form $G_{\diamond}$ is defined point-wise as

$$G_{\diamond,A}(X, Y) = \left(\text{Proj}_{\text{Hor}_A\mathcal{W}} \circ (d\phi)_A X\right)_{\mathcal{B}} \circ_{\text{F}} \left(\text{Proj}_{\text{Hor}_A\mathcal{W}} \circ (d\phi)_A Y\right)_{\mathcal{B}}. \tag{4.56}$$

The evaluation of $G_{\diamond}$ requires a Lyapunov solver while $G_{\mathcal{P}}$ only requires matrix multiplication. We use this bilinear form in place of $G_{\mathcal{P}}$ in some of our benchmarks later.

4.6 Numerical Optimization over $\mathcal{S}_{\text{E}}$

The Hessian approximation from Equation 2.57 allows us to use the Euclidean Hessian from Equation 2.56 to approximate the Riemannian Hessian. The reference [1] of that equation did not go through in detail how one would implement this approximation, and so we discuss this before presenting the proposed optimization scheme. For the rest of this chapter, we denote $f \in C^{\infty}(\mathcal{P})$ as the function of interest, $\tilde{H}$ as the approximated Hessian, and A_k as the solution update at iteration k unless otherwise specified. Our presentation in this section uses the manifold $\mathcal{P}$ with $G_{\mathcal{P}}$ as its Riemannian metric, but our schemes stay the same when the bilinear form $G_{\diamond}$ from Equation 4.56 is used in its place. Note that $G_{\diamond}$ does not induce a norm because it is not a proper Riemannian metric.

The Euclidean Hessian requires an orthonormal basis, so the first step is to perform a change-of-basis from the chart-induced basis to a basis that is orthonormal with respect to $G_{\mathcal{P}}$. Given an $A \in \mathcal{P}$, Algorithm 2.1 was used to orthonormalize the elementary basis of $\mathbb{R}^{Dp} \cong T_A\mathcal{P}$ with respect to $G_{\mathcal{P},\,A}$. Let the resultant basis be denoted by $\mathcal{U}$, and its components with respect to the chart-induced-basis $\mathfrak{B} = \{\partial_i\}_{i\in[\dim \mathcal{P}]}$ are stored as the columns of a matrix P. We then compute a finite-difference approximation of the Hessian of the function

$$\begin{aligned} h : \mathbb{R}^{Dp} &\to \mathbb{R}^{Dp} \\ X &\mapsto h(X) := f \circ \mathfrak{R}_{A,\mathcal{U}}((X)_{\mathcal{U}}), \end{aligned}$$

to get a temporary object H_{tmp}. The map $\mathfrak{R}_{A,\mathcal{U}} : \mathbb{R}^{Dp}_{\text{s}} \to \mathcal{P}$ is an implementation of a retraction with respect to the basis $\mathcal{U}$.

H_{tmp} might be numerically non-symmetric due to finite-precision arithmetic and floating-point representation. Therefore, we overwrite $H_{\text{tmp}} \leftarrow \left(H_{\text{tmp}} + H_{\text{tmp}}^T\right)/2$. If left uncorrected, this may trigger some errors in standard numerical libraries later in the algorithm. The Newton vector cannot be computed if H_{tmp} is singular. We prevent this by clamping the magnitude of each singular value s_i of H_{tmp} to some specified minimum value when the ratio $\frac{s_i}{\max\{s_i\}}$ drops below a specified threshold. Finally, we compute

$$\tilde{H} = PH_{\text{tmp}}P^{-1}, \tag{4.57}$$

where $\tilde{H}$ is the approximate Hessian from Equation 2.57. The multiplications with P and P^{-1} are required since $\tilde{H}$ is the matrix-form of a linear transformation (i.e., $(1,\,1)$-tensor) with respect to the basis $\mathfrak{B} \otimes \mathfrak{B}^*$, but H_{tmp} is the matrix-form of a linear transformation with respect to the basis $\mathcal{U} \otimes \mathcal{U}^*$.

The goal of this section is to specify an optimizer over $\mathcal{S}_{\text{E}}$. This optimizer is a cascade combination of four smaller optimization schemes; a trust-region method, a direct-search method based on the Nelder-Mead algorithm, the conjugate-gradient method from Algorithm 2.3, and the gradient-descent method from Algorithm 2.2. The trust-region method here is a modification of the one from [1] so that it uses the Rendl-Wolkowicz algorithm [47], and the direct-search method is my own adaptation of the line-search problem from [1] that uses the Nelder-Mead algorithm for the computations.

4.6.1 The Trust-Region Scheme

In this subsection, we discuss the details of the trust-region algorithm that is used in the quasi-Hankel optimizer. The material here builds on the theory from Section 2.12.2. At every iteration of the trust-region solver, we obtain solutions for η_k from both the Rendl-Wolkowicz algorithm [47] and the truncated conjugate-gradient algorithm [1]. We use the same the truncated conjugate-gradient algorithm as in [1], which is described in Algorithm 2.5. To use the Rendl-Wolkowicz algorithm, we first need to convert between the Riemannian trust-region subproblem

$$\begin{aligned} &\underset{\eta\in T_{A_k}\mathcal{P}}{\text{min.}} G_{A_k}\Big((df)^\sharp_{A_k},\,\eta\Big) + \frac{1}{2}G_{A_k}\Big(\tilde{H}_k\eta,\,\eta\Big) \\ &\text{subject to } G_{A_k}(\eta,\,\eta) \leq s_k^2, \end{aligned} \tag{4.58}$$

and the trust-region subproblem from Equation 2.76

$$\begin{aligned} &\underset{x\in\mathbb{R}^{Dp}_{\text{s}}}{\text{min.}} x^TAx - 2a^Tx \\ &\text{subject to } x^Tx \leq s_k^2, \end{aligned} \tag{4.59}$$

which is what the Rendl-Wolkowicz algorithm was designed to solve. Here, s_k is the trust-region radius for iteration k, and we make the replacement $A \leftarrow (A + A^T)/2$ if it is not a symmetric matrix.

Let $\mathcal{B}$ be a basis of $T_{A_k}\mathcal{P}$. For the rest of this subsection, we write $(X)_{\mathcal{B}}$ to mean $\textbf{colmaj}(X)_{\mathcal{B}}$ when $(X)_{\mathcal{B}}$ is involved in matrix multiplications. Let $\mathtt{G}$ be the metric tensor at A_k under some basis $\mathcal{B}^* \otimes \mathcal{B}^*$ (see Equation 4.46), so that

$$G_{A_k}(X, Y) = (X)_{\mathcal{B}}^T \mathtt{G} (Y)_{\mathcal{B}}, \forall X, Y \in T_{A_k}\mathcal{P}.$$

To reduce notation clutter, let us use the short-hand notations $(\boldsymbol{df})^\sharp := \left((df)^\sharp_{A_k}\right)_{\mathcal{B}}$ and $\boldsymbol{\eta} := (\eta)_{\mathcal{B}}$ for the component-forms of $(df)^\sharp_{A_k}$ and η. We make the substitution $x \leftarrow \boldsymbol{\eta}$ in Equation 4.59 to convert the metric G in Equation 4.58 to the Euclidean metric (i.e., dot product) in Equation 4.59. We then have

$$\begin{aligned} -2\boldsymbol{\eta}^T a &:= G_{A_k}\left((df)^\sharp_{A_k}, \eta\right) \\ &= \left((\boldsymbol{df})^\sharp\right)^T \mathtt{G}\boldsymbol{\eta} \\ \implies a &= -\frac{1}{2}\mathtt{G}(\boldsymbol{df})^\sharp \end{aligned}$$

and

$$\begin{aligned} \boldsymbol{\eta}^T A \boldsymbol{\eta} &:= \frac{1}{2} G_{A_k}\left(\tilde{H}_k \eta, \eta\right) \\ &= \frac{1}{2}\boldsymbol{\eta}^T \tilde{H}\mathtt{G}\boldsymbol{\eta} \\ \implies A &= \frac{1}{2}\tilde{H}\mathtt{G}. \end{aligned}$$

To convert the norm constraint, we apply the change-of-variables $y \leftarrow \mathtt{G}^{\frac{1}{2}} x$ in Equation 4.59. Here, the matrix square-root $\mathtt{G}^{\frac{1}{2}}$ is such that if $\mathtt{G} = Q\Lambda Q^T$ is the eigenvalue decomposition of $\mathtt{G}$, then

$$\mathtt{G}^{\frac{1}{2}} := Q\Lambda^{\frac{1}{2}}Q^T.$$

From Equation 4.58, we have

$$\begin{aligned} &\min_{y \in \mathbb{R}_s^{Dp}} y^T B y - 2b^T y \\ &\text{subject to } y^T y \leq s_k^2, \end{aligned} \tag{4.60}$$

where

$$\begin{aligned} B &= \mathtt{G}^{-\frac{1}{2}} A \mathtt{G}^{-\frac{1}{2}} = \frac{1}{2}\mathtt{G}^{-\frac{1}{2}}\tilde{H}\mathtt{G}^{\frac{1}{2}}, \\ b &= \mathtt{G}^{-\frac{1}{2}} a = -\frac{1}{2}\mathtt{G}^{\frac{1}{2}}(\boldsymbol{df})^\sharp, \\ y &= \mathtt{G}^{\frac{1}{2}}\boldsymbol{\eta}. \end{aligned}$$

We then overwrite $B \leftarrow (B^T + B)/2$ to make it symmetric. This produces an equivalent optimization problem to the one in Equation 4.60. We can now use the Rendl-Wolkowicz algorithm.

The trust-region algorithm that we used was modified from Algorithm 2.4 (taken from [1]) so that both the Rendl-Wolkowicz and the truncated conjugate-gradient algorithms were performed at each iteration. The solution that decreases the objective function the most is used. We also made minor modifications to some decision-making logic. The final algorithm that was used in our demonstrations is presented in Algorithm

4.1. The parameter c is the counter for consecutive iterations when this trust-region algorithm failed, T is a small number that makes sure the trust-region updates are making large-enough improvements to the objective function, and s_{max} ensures our allowed trust-region radius does not go unbounded. This could happen when the trust-region model is very inaccurate for many consecutive iterations. We reset the trust-region parameters once c is larger than a certain specified threshold.

Algorithm 4.1 High-level trust-region algorithm that combines the truncated conjugate-gradient and the Rendl-Wolkowicz algorithms.

Require: $s_{\text{min}}, s_{\text{max}} \in \mathbb{R}_{++}$, $s_{\text{min}} < s_{\text{max}}$, $\rho_{\text{min}} \leq \rho_{\text{accept}} \leq \rho_{\text{max}}$, $c \in \mathbb{Z}_{++}$, and $T \in \mathbb{R}$.

$k = 0$.
while termination condition not met **do**
  Compute η_{tCG} via Algorithm 2.5 and η_{RW} via Equations 2.77 and 2.78.
  if $f\left(\mathfrak{R}_{A_k}(\eta_{\text{tCG}})\right) > f(A_k)$ **then**
    Reject η_{gCG}
  end if
  if $f\left(\mathfrak{R}_{A_k}(\eta_{\text{RW}})\right) > f(A_k)$ **then**
    Reject η_{RW}
  end if
  if both algorithms are rejected **then**
    return A_k, maximum$(0.25s_k,\ s_{\text{min}})$, $c+1$.
  end if
  Compute the trust-region scores, $\rho_{k,\,\text{tCG}}$ and $\rho_{k,\,\text{RW}}$.
  Accept a solution if its trust-region score is higher than ρ_{accept}. If both solutions satisfy this condition, then choose the solution that decreases the objective cost the most. Denote this solution by η^*.
  if $\rho_k < 0.25$ **then**
    $s_{k+1} = \text{maximum}(0.25s_k,\ s_{\text{min}})$.
  end if
  if $\rho_k > 0.75$ **then**
    $s_{k+1} = \text{minimum}(2s_k,\ s_{\text{max}})$.
  end if
  if $\sqrt{G_{A_k}(\eta^* - \eta_k,\ \eta^* - \eta_k)} < T$ **then**
    return A_k, s_{k+1}, $c+1$.
  else
    return A_k, s_{k+1}, 0.
  end if
  $k = k+1$.
end while
return p_k.

4.6.2 The Nelder-Mead Scheme

Many Riemannian optimization methods are based on gradient-based Euclidean optimization algorithms so that their convergence analysis follows a similar development as the analysis for their Euclidean counterparts. In this thesis, we are not focused on proving convergence of our optimizer. We are focused on the empirical performance of our proposed solver, so we also take a direct-search approach to optimization on a manifold. Direct-search algorithms have the advantage that a gradient is not required, which means a Riemannian metric is not required. This is useful when the bilinear form $G_\diamond$ is used instead of the proper Riemannian

metric $G_{\mathcal{P}}$, when the current solution is ill-conditioned, when we have little confidence over our Hessian estimate $\tilde{H}$, or when $\tilde{H}$ does not provide a good search strategy (e.g., when near a saddle point [84]).

The gradient and Newton optimization algorithms in [1] identify a 1-dimensional subspace of $T_{p_k}M$ that is spanned by a vector, η_s, then performs line-search on the function $f \circ \mathfrak{R}_{A_k}(t\eta_s)$, with t being the line-search variable. The search direction η_s is usually $-(df)^{\sharp}_{A_k}$ or a direction based on the Newton vector. The Rendl-Wolkowicz approach to the trust-region subproblem also solves an equivalent univariate optimization problem (see Equation 2.75) to the original trust-region subproblem (see Equation 2.76). This motivated me to use the Nelder-Mead direct-search algorithm to minimize the following optimization subproblem over a vector space:

$$\min_{\eta \in T_{p_k}\mathcal{P}} f \circ \mathfrak{R}_{A_k}(\eta). \tag{4.61}$$

In this thesis, we call Equation 4.61 the *full-space search subproblem*. In the general case, this optimization problem is multi-modal, but a direct-search algorithm may allow us to escape from getting stuck around a local stationary point that is not the correct solution. Empirically, I found the presence of this solver in the proposed optimizer over $\mathcal{S}_E$ to be beneficial in accelerating the convergence rate.

To alleviate computational costs and to focus on a lower dimensional subspace of $T_{p_k}\mathcal{P}$ that is likely to contain a good update, I also used Nelder-Mead to solve optimization subproblems of the following form:

$$\min_{\{\alpha_i \in \mathbb{R}\}_i} f \circ \mathfrak{R}_{A_k}\left(\sum_{i=1}^{N} \alpha_i v_i\right), \tag{4.62}$$

where $N \leq \dim \mathcal{P}$, and $\{v_i \in T_{p_k}\mathcal{P}\}$ is a given set of independent vectors that changes at every iteration of the optimizer. These vectors can be based on the gradient of the updates. The optimization problem in Equation 4.62 is equivalent to the one in Equation 4.61 when $N = \dim \mathcal{P}$, and is equivalent to the line-search problem in Equation 2.67 when $N = 1$ and $v_1 = -(df)^{\sharp}_{A_k}$. The problem in Equation 4.62 is called a *subspace search subproblem* in this thesis. We use the following short-hand notation for the Newton vector:

$$\boldsymbol{\xi}_{Newton} := -\tilde{H}^{-1}\left((df)^{\sharp}_{A_k}\right)_{\mathcal{B}}.$$

Note that $\tilde{H}$ is always invertible due to our specification of it in the text surrounding Equation 4.57.

Let $\boldsymbol{\xi}$ be the normalized (using the Euclidean norm) version of $\left((df)^{\sharp}_{A_k}\right)_{\mathcal{B}} \in \mathbb{R}_s^{D \times p}$, $\boldsymbol{\xi}_{\text{fd}}$ be the normalized version of the finite-difference approximation of $\left((df)^{\sharp}_{A_k}\right)_{\mathcal{B}}$, and $\boldsymbol{\eta}_{\text{cg}}$ be the normalized version of the Fletcher-Reeves conjugate-gradient search direction, as computed by the object η in Algorithm 2.3 and β computed by Equation 2.69. Given a target CP rank p, I used the following versions of the subspace search subproblem in the demonstrations:

1. Subspace search subproblem #1, $N = p + 2$: Let the first p vectors in $\{v_k\}_{k=1}^{p+2}$ be such that

$$(v_k)_{i,j} = \begin{cases} \boldsymbol{\xi}_{i,k} & \text{if } k = j \\ 0 & \text{otherwise.} \end{cases}$$

   This means that each of the v_k vectors in component-form is a matrix that is zero everywhere except the kth column, which is set to be the normalized version of the component-form of the gradient $\sharp$-isomorphism vector. The motivation behind this setup is to allow different α_k's to scale the portion of the gradient for each of the p rank-one tensors differently. The last two vectors are $v_{p+1} = \boldsymbol{\xi}_{Newton}$ and $v_{p+2} = \boldsymbol{\xi}_{\text{cg}}$.

2. Subspace search subproblem #2, $N = p + 2$: Same as subproblem #1, but the first p vectors are from $\boldsymbol{\xi}_{\text{fd}}$ instead of $\boldsymbol{\xi}$.
3. Subspace search subproblem #3, $N = 3$: $v_1 = \boldsymbol{\xi}_{\text{cg}}$, $v_2 = \boldsymbol{\xi}$, and $v_3 = \boldsymbol{\xi}_{\boldsymbol{Newton}}$.
4. Subspace search subproblem #4, $N = 2$: $v_1 = \boldsymbol{\xi}_{\text{fd}}$ and $v_2 = \boldsymbol{\xi}_{\boldsymbol{Newton}}$.

The reason $\boldsymbol{\xi}$, $\boldsymbol{\xi}_{\text{fd}}$, and $\boldsymbol{\eta}_{\text{cg}}$ are normalized with respect to the Euclidean norm is that normalization by the induced norm from the Riemannian metric did not improve the average performance, but it did increase the computational cost.

4.6.3 The Proposed Optimizer Over $\mathcal{S}_{\mathbf{E}}$

We now combine the trust-region, Nelder-Mead, conjugate-gradient, and gradient descent methods into a single optimization scheme. We reject a candidate update A_k^* if it increases the objective function, or if the ratio of the largest and smallest magnitudes of the singular values of the matrix $\phi_{\mathcal{N},\mathcal{V}} \circ \phi_{\mathcal{P},\mathcal{N}}(A_k) \in \mathcal{V}$ exceed a given threshold ϵ_{gap}. This prevents our updates from becoming severely ill-conditioned. The maps $\phi_{\mathcal{P},\mathcal{N}}$ and $\phi_{\mathcal{N},\mathcal{V}}$ were defined in Equations 4.31 and 4.41, respectively. In our implementation, we used finite-difference differentiation from an open source library (Julia Calculus: https://github.com/johnmyleswhite/Calculus.jl, retrieved July 2017) to approximate the derivatives for the vector transport (the right-hand side of Equation 2.68) and the Hessian approximation $\tilde{H}$. The termination conditions are maximum iterations N_{max}, maximum allowed consecutive no-updates N_{stuck}, and a tolerance criterion ϵ_{grad} on the gradient $\sharp$-isomorphism vector's length using G. There are two versions of the proposed optimizer: one uses $G \leftarrow G_{\mathcal{P}}$, and the other uses $G \leftarrow G_{\diamond}$. We reset the trust-region algorithm's parameters when it gives more than N_{TRS} unaccepted candidate updates. We reset the conjugate-gradient parameters every N_{CG} iterations.

The exit conditions of the proposed optimizer are (with the exit code in brackets):

- Gradient tolerance reached (C): If $G_{A_k}\left((df)^{\sharp}_{A_k}, (df)^{\sharp}_{A_k}\right) < \epsilon_{\text{grad}}$.
- Stuck (R): If the number of consecutive iterations without a valid update exceeds N_{stuck}.
- Maximum iteration (T): If the maximum number of iterations is reached without reaching the gradient tolerance.

The counters k, i_{stuck}, and i_{TRS} are initialized to 0, and A_1 set to the initial solution. The optimizer performs the following sequence of actions:

1. Increment iteration counter $k \leftarrow k + 1$. Check if k exceeds N_{max}. If so, return with flag T.
2. Compute the Hessian approximation $\tilde{H}$ and the gradient $\sharp$-isomorphism vector $(df)^{\sharp}_{A_k}$. Check if $G_{A_k}\left((df)^{\sharp}_{A_k}, (df)^{\sharp}_{A_k}\right)$ is smaller than ϵ_{grad}. If so, return with flag C.
3. Increment $i_{\text{CG}} \leftarrow i_{\text{CG}} + 1$. Check if i_{CG} exceeds N_{CG}. If so, reset conjugate-gradient parameters, and set $i_{\text{CG}} \leftarrow 0$.
4. Solve the trust-region subproblem by using Algorithm 4.1. Check if the returned solution A_k^* is acceptable. If so, update $A_{k+1} = A_k^*$, set $i_{\text{stuck}} \leftarrow 0$, set $i_{\text{TRS}} \leftarrow 0$, and go to step 1. If not, increment $i_{\text{TRS}} \leftarrow i_{\text{TRS}} + 1$.
5. If $i_{\text{TRS}} > N_{\text{TRS}}$, then reset all given parameters of Algorithm 4.1, and set $i_{\text{TRS}} \leftarrow 0$.
6. Solve the full-space search subproblem in Equation 4.61 by using the Nelder-Mead algorithm. Check if the returned solution A_k^* is acceptable. If so, update $A_{k+1} = A_k^*$, set $i_{\text{stuck}} \leftarrow 0$, and go to step 1.

7. Solve all of the subspace search subproblems from Section 4.6.2 by using the Nelder-Mead algorithm on the subproblems of the form in Equation 4.62. Check if the returned solution A_k^* is acceptable. If so, update $A_{k+1} = A_k^*$, set $i_{\text{stuck}} \leftarrow 0$, and go to step 1.

8. Compute a conjugate-gradient update A_k^* by using Algorithm 2.3. Check if the returned solution A_k^* is acceptable. If so, update $A_{k+1} = A_k^*$, set $i_{\text{stuck}} \leftarrow 0$, and go to step 1.

9. Compute a gradient descent update A_k^* by using Algorithm 2.2. Check if the returned solution A_k^* is acceptable. If so, update $A_{k+1} = A_k^*$, set $i_{\text{stuck}} \leftarrow 0$, and go to step 1.

10. Increment $i_{\text{stuck}} \leftarrow i_{\text{stuck}} + 1$, and check if $i_{\text{stuck}} > N_{\text{stuck}}$. If so, exit with flag R.

When our trust-region method (Algorithm 4.1) is inaccurate, we try the full-space and subspace search methods before the conjugate-gradient algorithms. From my observations, the full-space updates often select a more substantial update than the trust-region methods when the trust-region methods start to converge slowly, but the trust-region method is faster to compute. Therefore, the steps of this cascading scheme between the trust-region and a direct-search method complement each other. I also observed that the conjugate-gradient and gradient descent algorithms implemented here have inferior convergence rates than the other methods, but they are the only solvers here that provide reliable updates when the Hessian approximation $\tilde{H}$ or the metric tensor G is ill-conditioned. For this reason, they are included as a last-resort when all other methods fail.

4.7 Demonstrations

For the remainder of this chapter, let us use the following notation unless otherwise stated:

- $A \in \mathcal{P}$ and $B = \phi(A) = \phi_{\mathcal{N},\mathcal{V}} \circ \phi_{\mathcal{P},\mathcal{N}}(A)$, where $\phi_{\mathcal{P},\mathcal{V}}$ and $\phi_{\mathcal{P},\mathcal{N}}$ are defined in Equations 4.31 and 4.41, respectively. It follows that BB^T is a quasi-Hankel matrix.

- $A^* \in \mathcal{P}$ is the returned solution from an optimizer.

- The ground-truth rank-one decompositions are of cardinality p, i.e., they are a set of p rank-one tensors. Their exact values are reported in Appendix C. The rank-one decomposition $(\mathfrak{z},\ c)$ that we solve for is also of cardinality p.

We demonstrate the proposed optimizer by solving the optimization problem

$$\min_{A \in \mathcal{P}} \left(S. * \left(\mathtt{H} - \phi(A)\phi(A)^T \right) \right) \circ_{\text{F}} \left(S. * \left(\mathtt{H} - \phi(A)\phi(A)^T \right) \right), \tag{4.63}$$

where $.*$ denotes the per-element multiplication of two matrices, and the entries of S are non-negative. The matrix S acts as a mask, or a set of weights, so that we can control which entries of $\mathtt{H}$ we wish to emphasize in the fit. This problem was chosen because its optimal solution has an objective score of zero, which makes it easy to evaluate performance. The focus here is to validate the numerical performance of our optimizer. In practice, a quasi-Hankel matrix-completion problem needs to be solved to obtain $\mathtt{H}$, unless $\mathtt{H}$ can be computed directly from data; this was discussed in Section 4.1.3.2. We do not cover investigations into tensor-completion problems or applications of general-purpose optimization over tensors in this thesis.

It follows from Section 2.14 that not all entries in a quasi-Hankel matrix, such as $\mathtt{H}$, are distinct. In our benchmarks, we set S to be such that $S_{i,j} = \frac{1}{N_{i,j}}$ if $N_{i,j}$ is the number of entries of $\mathtt{H}$ that have the same value as $\mathtt{H}_{i,j}$. This choice of S allows each distinct entry of $\mathtt{H}$ to make a contribution to the cost function that is invariant to how many times that distinct entry appears in $\mathtt{H}$.

4.7.1 Gradient of Cost Function on $\mathbb{R}^{m\times p}_{\mathbf{s}}$

We discussed how to project a tangent vector from $T_B\mathbb{R}^{m\times p}_{\mathrm{s}}$ to $T_B\mathcal{V}$ in Equation 4.29, but we did not discuss how to compute the gradient $\sharp$-isomorphism vector for $\mathcal{V}$ if a $f \in C^\infty(\mathcal{V})$ was given. Here, we derive an analytical expression for this vector when

$$\begin{aligned} f(B) &= \big(S. * \big(\mathtt{H} - BB^T\big)\big) \circ_{\mathrm{F}} \big(S. * \big(\mathtt{H} - BB^T\big)\big) \\ &= \Big(S. * \Big(\mathtt{H} - \phi(A)\phi(A)^T\Big)\Big) \circ_{\mathrm{F}} \Big(S. * \Big(\mathtt{H} - \phi(A)\phi(A)^T\Big)\Big), \end{aligned}$$

which is a re-parameterization of the cost function f in Equation 4.63 from $A \in \mathcal{P}$ to $B = \phi(A) \in \mathcal{V}$. Our strategy is to first compute the gradient covector on $\mathbb{R}^{m\times p}$, which is also the gradient $\sharp$-isomorphism vector since we are using the Euclidean metric on $\mathbb{R}^{m\times p}$. We project to $\mathcal{V}$ via the projector defined in Equation 4.29. From there, one can use the matrix $\Big(\big(d\phi\big)_{\phi(A)}\Big)^{-1}_{\mathcal{B}}$ as the push-forward map in matrix-form (under some appropriate basis) from $T_{\phi(A)}\mathcal{V}$ to $T_A\mathcal{P}$ to get the gradient $\sharp$-isomorphism on $\mathcal{P}$. Here, we only discuss the computation of the gradient covector for B on $\mathbb{R}^{m\times p}$, since the other steps were detailed in earlier discussions.

We use the formulae for matrix differentials from Equations 33 to 45 of [85] to express the total derivative of this cost function $f(B)$, then infer the derivative with respect to B. This is done by re-arranging the expressions so that if we have $df = C \circ_{\mathrm{F}} dB$, then C is the matrix derivative of the differentiable function $f : \mathbb{R}^{m\times m}_{\mathrm{s}} \to \mathbb{R}$ with respect to the variables $\{B_{i,\,j}\}^{m,\,m}_{i=1,\,j=1}$; i.e.,

$$C_{i,\,j} := \frac{\partial f}{\partial B_{i,\,j}}.$$

Throughout this derivation, we frequently use the cyclic property of trace, i.e.,

$$\mathrm{tr}(XYZ) = \mathrm{tr}(ZXY) = \mathrm{tr}(YZX)$$

for similarly sized square matrices X, Y, and Z. This is useful because the Frobenius product of matrices can be written as

$$X \circ_{\mathrm{F}} Y = \mathrm{tr}\big(X^TY\big) = \mathrm{tr}\big(Y^TX\big)$$

for any similarly sized square matrices X and Y.

Taking the total derivative of $f(B)$, we have

$$\begin{aligned} df &= d\big(S. * \big(\mathtt{H} - BB^T\big)\big) \circ_{\mathrm{F}} \big(S. * \big(\mathtt{H} - BB^T\big)\big) + \big(S. * \big(\mathtt{H} - BB^T\big)\big) \circ_{\mathrm{F}} d\big(S. * \big(\mathtt{H} - BB^T\big)\big) \\ &= \big(dS. * \big(BB^T - H\big) + S. * d\big(BB^T - H\big)\big) \circ_{\mathrm{F}} \big(S. * \big(BB^T - H\big)\big) + \\ &\quad \big(S. * \big(BB^T - H\big)\big) \circ_{\mathrm{F}} \big(dS. * \big(BB^T - H\big) + S. * d\big(BB^T - H\big)\big), \end{aligned}$$

where $.*$ is the per-element multiplication operator.

Since S is constant with respect to B, $dS = 0$. The above becomes

$$\begin{aligned} df &= \big(S. * d\big(BB^T - \mathtt{H}\big)\big) \circ_{\mathrm{F}} \big(S. * \big(BB^T - \mathtt{H}\big)\big) + \big(S. * \big(BB^T - \mathtt{H}\big)\big) \circ_{\mathrm{F}} \big(S. * d\big(BB^T - \mathtt{H}\big)\big). \\ &= 2\big(S. * d\big(BB^T - \mathtt{H}\big)\big) \circ_{\mathrm{F}} \big(S. * \big(BB^T - \mathtt{H}\big)\big). \end{aligned} \tag{4.64}$$

For the first term, we have

$$\begin{aligned} d\big(BB^T - \mathtt{H}\big) &= d\big(BB^T\big) \\ &= (dB)B^T + B\big(dB^T\big) \\ \implies S. * d\big(BB^T - \mathtt{H}\big) &= S. * \big((dB)B^T + B\big(dB^T\big)\big). \end{aligned} \tag{4.65}$$

Note that for any $m \times p$ matrices X and Y, and any $m \times m$ matrix Z, we have

$$\begin{aligned}\big(S. * \big(XY^T\big)\big) \circ_{\mathrm{F}} Z &= Z \circ_{\mathrm{F}} \big(S. * \big(XY^T\big)\big) \\ &= \mathrm{tr}\big(Z^T\big(S. * \big(XY^T\big)\big)\big) \qquad (4.66) \\ &= \sum_{i,\,j \in [m]^2} \big(Z. * S. * \big(XY^T\big)\big)_{i,\,j} \\ &= \mathrm{tr}\Big((Z. * S)^T XY^T\Big) \\ &= (Z. * S) \circ_{\mathrm{F}} \big(XY^T\big). \qquad (4.67)\end{aligned}$$

Let $U := S. * (BB^T - \mathtt{H})$ to reduce notation clutter. Now, we use the expressions in Equations 4.65 and 4.67 to rewrite Equation 4.64 as

$$\begin{aligned} df &= 2\big(S. * \big((dB)B^T + B\big(dB^T\big)\big)\big) \circ_{\mathrm{F}} U \\ &= 2(U. * S) \circ_{\mathrm{F}} (dB)B^T + 2(U. * S) \circ_{\mathrm{F}} B\big(dB^T\big) \\ &= 2(U. * S)B \circ_{\mathrm{F}} dB + 2B^T(U. * S) \circ_{\mathrm{F}} dB^T \\ &= 2(U. * S)B \circ_{\mathrm{F}} dB + 2(U. * S)^T B \circ_{\mathrm{F}} dB \\ &= 2\Big((U. * S) + (U. * S)^T\Big)B \circ_{\mathrm{F}} dB. \end{aligned}$$

It follows that the gradient covector in matrix-form with respect to the chart-induced dual basis $\mathfrak{B}^*$ is

$$\begin{aligned}(df)_{\mathfrak{B}} &= 2\Big((U. * S) + (U. * S)^T\Big)B, \\ &= 2\big(S. * S. * \big(BB^T - \mathtt{H}\big)\big)B + 2\big(S. * S. * \big(BB^T - \mathtt{H}\big)\big)^T B. \qquad (4.68)\end{aligned}$$

Since the Euclidean metric is used on $\mathbb{R}^{m \times p}_{\mathrm{s}}$ in this thesis, we have $(df)^{\sharp}_{\mathfrak{B}} = (df)_{\mathfrak{B}^*}$, where $\mathfrak{B}$ is the chart-induced basis. In our implementation, the identity chart for $m \times p$ matrices is used, which means the gradient and its $\sharp$-isomorphism vector at $B = \phi(A)$ is expressed in the elementary basis of $T_{\phi(A)}\mathbb{R}^{m \times p}_{\mathrm{s}} \cong \mathbb{R}^{m \times p}$.

4.7.2 Results

In Section 4.7.2.1, we report results from both versions of the proposed optimizer. In Section 4.7.2.2, we report results from both versions of the proposed optimizer and the alternating least-squares (CP-ALS) algorithm from [80, 81]. Since we have a cost function that implements the CP decomposition problem, we also include a *relative error* performance measure [32], which is often used in the tensor decomposition literature. We modified it for use with quasi-Hankel matrices, and the details are discussed in Sections 4.7.2.1 and 4.7.2.2.

Our optimizer only accepts an update if it is in the descent direction, so it can only yield the same or a better solution than any CP decomposition scheme if the initial guess for our optimizer is set to be the solution returned by that CP decomposition scheme. For this reason, we use the same initial guesses for our optimizer as the CP-ALS algorithm in order to get a fair comparison between the two methods.

Denote by A_k the returned solution from an optimizer at iteration k. We have

$$f(A_k) = \Big(S. * \Big(\mathtt{H} - \phi(A_k)\phi(A_k)^T\Big)\Big) \circ_{\mathrm{F}} \Big(S. * \Big(\mathtt{H} - \phi(A_k)\phi(A_k)^T\Big)\Big), \qquad (4.69)$$

where $\phi = \phi_{\mathcal{N},\,\mathcal{V}} \circ \phi_{\mathcal{P},\,\mathcal{N}}$. We also write $\mathrm{GM}(X)$ to mean the following method of approximating the length of a tangent vector X at A_k:

$$\mathrm{GM}(X) := \sqrt{G(X,\,X)}, \qquad (4.70)$$

where G is $G_{\mathcal{P}}$ or $G_{\diamond}$. Note that $\text{GM}(X)$ is not a norm when $G_{\diamond}$ is used, since $G_{\diamond}$ at A_k is not necessarily a positive-definite bilinear form.

For all our benchmarks in this chapter, we use ϵ_{grad} =1E-08, $N_{\text{stuck}} = 100$, ϵ_{gap} =1E-05, $N_{\text{max}} = 20000$, $N_{\text{TRS}} = 50$, and $N_{\text{CG}} = 30$. Recall from Section 4.6.3 that the exit code C means gradient tolerance reached, R means the optimizer is stuck, and T means gradient tolerance was not reached within N_{max} iterations. For the trust-region method from Algorithm 4.1, we have s_{max} =1E15, s_{min} =1E-12, T =1E-10. We set the default trust-region radius upon reset to be $s_{\text{reset}} = 1$. For the truncated conjugate-gradient method (Algorithm 2.5) portion of Algorithm 4.1, we limit the solver to only 10 iterations. For the Rendl-Wolkowicz method portion of Algorithm 4.1, the version of Brent's method that was implemented in the Julia Optimization toolbox (https://github.com/JuliaNLSolvers/Optim.jl, retrieved July 2017) was used with an initial solution of $t \leftarrow 1$ to solve Rendl-Wolkowicz's univariate optimization problem in Equation 2.75, which is equivalent to Equation 4.60. These parameter values were empirically found to give good performance with the proposed scheme.

We use the following abbreviation in our table headings to identify specific *configurations* of D, L, and p: Dx-py-Lz means $D = x$, $p = y$, and $L = z$. All objective scores, relative errors, and GM$(\cdot)$ lengths are rounded to three significant figures. The configurations of the benchmarks used for the demonstrations here have $D \in \{3, 4\}$ and $L \in \{4, 6\}$; recall that $\mathcal{S}_{\text{E}}$ allows only even values of L. Any higher orders or dimensions took too long to run on the test machine. The test computer has an AMD Ryzen 1700 CPU (8 cores, each at 3 Ghz) and 16 gigabytes of RAM. No GPU acceleration was used. Julia 0.5.2 was used to implement all thesis work, and the operating system used was 64-bit Ubuntu 16.04.2.

We now define some terminology to address the benchmark that generated the results reported here. Each *instantiated problem* is generated by a ground truth rank-one decomposition. Each *test run* is a pairing of an instantiated problem with an *initial solution*. The results reported in Sections 4.7.2.1 and 4.7.2.2 came from a benchmark where there are:

1. Six configurations: D3-p2-L4, D3-p4-L6, D3-p7-L4, D4-p3-L4, D4-p4-L4, and D4-p4-L6.
2. Seven instantiated problems for each configuration.
3. Five initial solutions for each problem. The same initial solutions were used within the same configuration.

4.7.2.1 Decomposing Quasi-Hankel Matrices

Here, we report the results of both versions of the proposed optimizer for the configurations D3-p2-L4, D3-p4-L6, D3-p7-L4, D4-p3-L4, D4-p4-L4, and D4-p4-L6 on the optimization problem from Equation 4.63. Higher values of p for the 4-dimensional configurations were too slow to compute on the test machine. The primary performance measure here is the objective score from Equation 4.69. The secondary performance measure is the relative error [32] given by

$$\frac{\left\| \phi(A)\phi(A)^T - \mathtt{H} \right\|_2}{\left\| \mathtt{H} \right\|_2}, \tag{4.71}$$

where $\mathtt{H}$ is the given quasi-Hankel matrix for the optimization problem.

The relative error, objective score upon termination, the approximated gradient length (GM from Equation 4.70), iterations taken, and termination conditions are reported in Tables 4.1, 4.2, 4.3, 4.4, 4.5, and 4.6. These tables are for configurations D3-p2-L4, D3-p4-L6, D3-p7-L4, D4-p3-L4, D4-p4-L4, D4-p4-L6, respectively. When the numerical gradient tolerance was achieved, it implies that we are likely to have converged to a stationary point of the objective function, which may not be the correct solution.

Since our setup is based on a dehomogenized polynomial, we can expect some numerical issues when the initial solutions or ground truth rank-one decompositions have a near-zero value in entries that correspond to c. This is because those entries are involved in the division operation of the dehomogenization procedure, e.g., in the map $\phi_{\mathcal{P},\mathcal{N}}$ (Equation 4.31). The implementation of both versions of the proposed optimizer treats a rank-one decomposition $(\mathfrak{z}, c)$ as a point $A^{\mathcal{P}}$ on the manifold $\mathcal{P}$, and the first row of $A^{\mathcal{P}}$ corresponds to $\bar{c} := \sqrt{c}$; see Section 4.5.2 for the derivation of $\mathcal{P}$. To avoid distraction to our presentation here, the initial solutions $A_0^{\mathcal{P}}$ and ground truth rank-one decompositions used are reported in Appendix C.

Table 4.7 summarizes the results presented in Tables 4.1 to 4.6. Each row corresponds to one of the proposed schemes on a particular configuration, which corresponds to a pair of rows that are vertically connected in this table. In Table 4.7, the single bold font for each column indicates the scheme that performed better; a lower relative error or a higher number of runs that achieves a certain level of relative error makes a scheme better than the others.

	$G_{\mathcal{P}}$					$G_{\diamond}$				
	Error	Score	GM	Iter.	Exit	Error	Score	GM	Iter.	Exit
P1-S1	3.82E-15	1.01E-28	2.55E-14	56	C	1.38E-15	3.21E-29	3.12E-14	69	C
P1-S2	1.00E+00	8.43E+00	5.89E-12	10	C	1.00E+00	8.43E+00	1.00E-11	10	C
P1-S3	1.00E+00	8.43E+00	1.06E-11	16	C	1.00E+00	8.43E+00	5.54E-08	128	R
P1-S4	1.02E-11	4.55E-22	2.00E-11	16	C	6.28E-15	3.01E-28	4.33E-14	13	C
P1-S5	2.45E-12	2.71E-23	4.46E-12	18	C	1.00E+00	8.43E+00	2.65E-11	14	C
P2-S1	8.68E-16	2.81E-29	6.12E-14	16	C	9.78E-15	3.19E-28	5.98E-14	16	C
P2-S2	1.09E-14	4.11E-28	4.66E-14	10	C	2.24E-13	1.90E-25	9.78E-13	11	C
P2-S3	6.17E-15	1.30E-28	5.98E-14	27	C	2.90E-15	3.55E-29	5.13E-14	20	C
P2-S4	1.01E-14	3.91E-28	1.84E-13	20	C	9.87E-15	3.17E-28	1.37E-13	20	C
P2-S5	6.98E-16	6.84E-30	1.86E-14	14	C	3.58E-15	7.09E-29	1.22E-13	14	C
P3-S1	1.04E-14	1.85E-27	1.59E-13	25	C	1.36E-15	1.81E-29	2.81E-14	14	C
P3-S2	1.87E-15	7.64E-29	3.73E-14	30	C	1.65E-15	6.88E-29	2.63E-14	30	C
P3-S3	1.97E-14	7.48E-27	6.03E-13	19	C	9.70E-16	7.24E-30	1.06E-14	19	C
P3-S4	1.51E-12	5.30E-23	2.00E-11	12	C	4.10E-16	2.37E-30	9.43E-15	12	C
P3-S5	1.30E-12	4.78E-23	1.97E-11	23	C	1.33E-15	3.39E-29	3.84E-14	32	C
P4-S1	5.43E-10	7.17E-19	3.36E-11	239	C	6.09E-09	9.00E-17	6.14E-11	125	C
P4-S2	5.22E-15	1.03E-27	5.11E-11	243	C	4.85E-09	5.71E-17	4.30E-11	118	C
P4-S3	3.22E-08	2.52E-15	8.24E-11	125	C	8.86E-10	1.91E-18	1.34E-11	179	C
P4-S4	1.23E-08	3.66E-16	3.61E-11	142	C	1.62E-07	6.39E-14	7.40E-11	134	C
P4-S5	2.19E-15	6.00E-26	7.47E-11	227	C	4.30E-08	4.49E-15	4.97E-11	112	C
P5-S1	2.16E-15	6.45E-29	3.27E-14	12	C	2.70E-13	5.47E-24	1.31E-11	18	C
P5-S2	7.24E-16	1.85E-29	2.52E-14	13	C	1.51E-15	5.11E-29	4.05E-14	13	C
P5-S3	1.04E-15	4.14E-29	2.84E-14	18	C	1.53E-15	4.11E-29	3.02E-14	19	C
P5-S4	3.01E-15	1.44E-28	4.85E-14	50	C	1.55E-15	3.11E-29	2.24E-14	31	C
P5-S5	1.58E-15	3.60E-29	2.59E-14	12	C	2.97E-16	1.99E-30	6.06E-15	12	C
P6-S1	2.13E-13	1.32E-25	1.13E-13	24	C	5.15E-09	7.49E-17	5.22E-11	30	C
P6-S2	5.26E-13	7.83E-25	7.97E-14	23	C	8.05E-11	1.89E-20	9.67E-13	17	C
P6-S3	7.53E-14	1.74E-26	2.60E-14	26	C	9.13E-11	2.35E-20	1.28E-12	20	C
P6-S4	1.39E-11	7.35E-22	2.62E-12	15	C	1.81E-10	9.57E-20	1.28E-11	26	C
P6-S5	6.80E-14	1.43E-26	5.63E-14	15	C	5.69E-12	9.16E-23	2.64E-13	12	C
P7-S1	3.74E-15	7.45E-28	1.50E-13	17	C	1.41E-11	1.06E-21	9.67E-12	159	C
P7-S2	1.32E-12	1.48E-23	1.18E-12	13	C	3.81E-15	1.55E-28	6.10E-15	14	C
P7-S3	4.68E-14	1.48E-26	7.68E-14	76	C	6.32E-14	3.34E-26	2.96E-13	22	C
P7-S4	9.70E-15	6.90E-28	1.06E-14	13	C	3.44E-15	6.12E-29	1.75E-15	14	C
P7-S5	3.41E-15	1.27E-28	1.18E-14	25	C	5.32E-15	1.36E-28	4.24E-15	18	C
Avg.	5.71E-02	4.82E-01	1.04E-11	46.9	-	8.57E-02	7.23E-01	1.59E-09	43.3	-
Med.	1.09E-14	7.48E-27	1.13E-13	19	-	9.87E-15	3.19E-28	1.37E-13	19	-

Table 4.1: Optimization results for the problem in Equation 4.63, using configuration D3-p2-L4. The test run label format "Px-Sy" refers to the xth instantiated problem with the yth initial solution. "Error" is the relative error as defined by Equation 4.71. "Score" and "GM" are given by Equations 4.69 and 4.70, respectively. "Iter." is the number of iterations ran. "Exit" is the exit state, which can take on the values: "C" to mean gradient tolerance was reached, "R" to mean no valid updates after N_{stuck} consecutive iterations, and "T" to mean maximum iterations reached. "Avg." and "Med." are the average and median of the column, respectively.

	$G_{\mathcal{P}}$					$G_{\diamond}$				
	Error	Score	GM	Iter.	Exit	Error	Score	GM	Iter.	Exit
P1-S1	2.69E-14	6.73E-26	2.12E-12	90	C	2.54E-14	2.36E-26	5.90E-11	755	C
P1-S2	4.35E-14	2.22E-25	1.43E-11	320	C	1.24E-12	7.24E-24	4.71E-11	566	C
P1-S3	1.82E-14	5.60E-26	3.38E-11	212	C	8.35E-13	9.86E-24	8.42E-11	722	C
P1-S4	2.80E-14	2.90E-26	2.36E-11	192	C	5.96E-13	9.56E-23	3.21E-11	586	C
P1-S5	5.81E-14	3.06E-25	3.38E-11	72	C	6.40E-14	2.26E-25	2.04E-13	466	C
P2-S1	7.08E-15	6.58E-28	2.88E-15	227	C	3.72E-14	1.32E-26	2.98E-14	115	C
P2-S2	1.46E-14	8.56E-26	2.95E-13	216	C	3.12E-14	5.16E-26	5.00E-14	133	C
P2-S3	8.98E-14	5.80E-24	6.62E-12	451	C	1.04E-13	3.97E-25	2.63E-12	124	C
P2-S4	5.57E-14	2.57E-26	5.36E-14	1347	C	3.69E-14	1.29E-26	9.89E-15	115	C
P2-S5	1.04E-13	1.00E-25	1.01E-13	115	C	1.41E-14	5.25E-27	1.80E-13	115	C
P3-S1	1.33E-13	2.28E-23	4.17E-12	114	C	4.53E-04	4.22E-05	3.98E-12	518	C
P3-S2	2.99E-15	9.28E-29	2.13E-14	185	C	1.14E-14	4.20E-25	1.04E-12	75	C
P3-S3	4.53E-04	4.22E-05	4.66E-11	676	C	4.53E-04	4.22E-05	1.83E-12	258	C
P3-S4	4.53E-04	4.22E-05	5.36E-14	397	C	1.55E-12	8.11E-21	3.30E-11	114	C
P3-S5	4.53E-04	4.22E-05	6.42E-12	180	C	2.83E-15	6.62E-29	1.23E-14	115	C
P4-S1	2.00E-11	1.33E-21	4.53E-13	133	C	1.33E-06	4.83E-12	7.82E-11	3034	C
P4-S2	1.00E+00	8.40E+00	7.34E-09	7178	R	1.00E+00	8.40E+00	1.46E-08	6602	R
P4-S3	1.75E-07	8.46E-14	1.29E-11	565	C	7.62E-07	1.61E-12	7.20E-11	457	C
P4-S4	1.00E+00	8.40E+00	8.71E-09	7798	R	1.00E+00	8.40E+00	2.42E-09	4375	R
P4-S5	1.97E-08	1.20E-15	6.31E-12	226	C	3.07E-10	3.04E-19	6.86E-12	1351	C
P5-S1	4.46E-05	4.46E-06	1.34E-09	166	R	3.59E-05	4.47E-06	5.83E-08	113	R
P5-S2	4.46E-05	4.46E-06	6.80E-11	11552	C	2.26E-14	1.06E-25	9.78E-11	64	C
P5-S3	5.23E-15	1.34E-27	2.02E-12	197	C	2.78E-14	3.56E-26	4.73E-11	93	C
P5-S4	4.46E-05	4.46E-06	8.37E-11	326	C	4.46E-05	4.46E-06	1.73E-10	292	R
P5-S5	4.46E-05	4.46E-06	1.12E-07	141	R	1.97E-15	2.87E-28	9.14E-13	136	C
P6-S1	5.08E-14	1.94E-24	9.21E-12	41	C	9.20E-05	8.59E-05	7.48E-12	108	C
P6-S2	5.44E-02	4.60E+00	5.13E-08	226	R	9.20E-05	8.59E-05	1.64E-08	134	R
P6-S3	5.61E-05	6.70E-05	3.17E-08	114	R	5.61E-05	6.70E-05	4.78E-08	295	R
P6-S4	5.61E-05	6.70E-05	2.84E-11	604	C	8.27E-05	8.72E-05	9.33E-11	10700	C
P6-S5	5.86E-02	5.89E+00	2.10E-07	145	R	5.61E-05	6.70E-05	2.11E-08	128	R
P7-S1	5.45E-14	3.25E-26	8.07E-13	59	C	2.91E-05	1.50E-05	5.27E-12	63	C
P7-S2	2.16E-12	2.08E-21	6.80E-11	57	C	6.75E-05	1.59E-05	6.33E-12	72	C
P7-S3	5.12E-05	1.61E-05	6.31E-11	105	C	5.12E-05	1.61E-05	1.43E-11	123	C
P7-S4	6.75E-05	1.59E-05	4.61E-11	63	C	3.55E-13	2.89E-24	8.81E-12	53	C
P7-S5	1.98E-12	1.42E-21	7.20E-11	61	C	5.12E-05	1.61E-05	5.66E-11	91	C
Avg.	6.04E-02	7.80E-01	1.21E-08	987.17	-	5.72E-02	4.80E-01	4.61E-09	944.6	-
Med.	2.00E-11	2.08E-21	2.36E-11	192	-	3.07E-10	3.04E-19	3.21E-11	133	-

Table 4.2: Optimization results for the problem in Equation 4.63, using configuration D3-p4-L6. The test run label format "Px-Sy" refers to the xth instantiated problem with the yth initial solution. "Error" is the relative error as defined by Equation 4.71. "Score" and "GM" are given by Equations 4.69 and 4.70, respectively. "Iter." is the number of iterations ran. "Exit" is the exit state, which can take on the values: "C" to mean gradient tolerance was reached, "R" to mean no valid updates after N_{stuck} consecutive iterations, and "T" to mean maximum iterations reached. "Avg." and "Med." are the average and median of the column, respectively.

	$G_{\mathcal{P}}$					$G_{\diamond}$				
	Error	Score	GM	Iter.	Exit	Error	Score	GM	Iter.	Exit
P1-S1	6.85E-12	1.98E-22	1.75E-12	227	C	1.50E-11	9.51E-22	2.05E-11	226	C
P1-S2	6.85E-12	3.43E-22	3.10E-12	233	C	7.03E-12	1.96E-21	6.72E-12	227	C
P1-S3	3.18E-11	2.86E-21	2.13E-12	339	C	2.29E-12	2.22E-23	3.55E-13	159	C
P1-S4	2.92E-10	5.92E-19	9.51E-11	250	C	1.04E-11	5.79E-22	3.40E-12	563	C
P1-S5	4.24E-10	4.04E-19	7.35E-12	451	C	2.39E-11	2.73E-21	5.99E-12	229	C
P2-S1	3.21E-14	3.85E-26	5.55E-13	5287	C	2.05E-14	4.70E-27	3.02E-14	14269	C
P2-S2	5.75E-14	1.45E-25	9.06E-13	9542	C	4.11E-14	2.16E-26	2.77E-13	14619	C
P2-S3	2.42E-13	1.89E-23	1.27E-12	758	C	9.65E-15	1.60E-27	3.29E-14	9347	C
P2-S4	1.54E-14	4.21E-26	5.08E-13	616	C	1.76E-13	4.63E-22	7.25E-11	2914	C
P2-S5	2.47E-14	8.44E-26	5.56E-13	6411	C	3.89E-14	1.50E-26	2.90E-13	13565	C
P3-S1	2.87E-11	3.17E-21	3.08E-11	233	C	5.99E-04	4.46E-05	2.61E-07	7873	R
P3-S2	1.28E-12	1.36E-20	7.47E-11	18256	C	1.01E-04	5.52E-05	3.09E-11	340	C
P3-S3	1.01E-04	5.52E-05	1.16E-11	439	C	5.99E-11	7.18E-21	1.67E-11	258	C
P3-S4	5.11E-11	1.43E-20	7.60E-11	408	C	1.01E-04	5.52E-05	6.48E-12	6062	C
P3-S5	1.11E-04	5.54E-05	5.24E-11	180	C	6.23E-12	2.01E-22	8.15E-12	924	C
P4-S1	4.12E-07	3.49E-11	9.39E-11	144	C	4.16E-07	3.49E-11	2.11E-12	245	C
P4-S2	2.26E-08	2.40E-13	5.85E-11	1323	C	1.04E-07	7.22E-13	5.38E-11	337	C
P4-S3	4.13E-07	3.50E-11	5.77E-11	274	C	8.69E-09	2.92E-14	4.04E-11	220	C
P4-S4	2.66E-08	3.17E-13	7.81E-11	626	C	2.48E-07	2.24E-11	7.93E-11	225	C
P4-S5	2.06E-08	1.88E-13	2.89E-11	785	C	8.77E-09	2.43E-14	5.93E-11	404	C
P5-S1	1.40E-04	3.04E-05	1.45E-08	1542	R	7.47E-06	1.22E-07	4.71E-08	3196	R
P5-S2	6.96E-06	1.23E-07	4.35E-11	3525	C	1.41E-05	5.10E-08	2.21E-07	20000	T
P5-S3	3.11E-11	1.01E-20	1.89E-11	3865	C	6.21E-05	3.21E-05	2.74E-08	3143	R
P5-S4	4.60E-12	2.94E-22	1.39E-11	2948	C	6.90E-06	1.24E-07	7.97E-11	2469	C
P5-S5	7.07E-06	1.23E-07	5.40E-11	663	C	6.49E-06	1.25E-07	8.99E-11	11077	C
P6-S1	7.85E-02	2.72E+00	3.33E-01	481	R	3.10E-14	3.48E-24	7.11E-11	898	C
P6-S2	3.17E-14	1.23E-26	4.62E-13	675	C	2.27E-12	8.94E-22	9.14E-11	1074	C
P6-S3	6.02E-13	3.22E-23	9.52E-12	463	C	1.23E-13	2.08E-24	4.63E-11	9231	C
P6-S4	7.82E-15	5.86E-27	1.79E-13	9191	C	9.81E-13	1.01E-22	1.75E-11	9253	C
P6-S5	1.88E-12	3.16E-22	4.99E-11	710	C	7.80E-14	1.54E-24	8.07E-11	9233	C
P7-S1	3.24E-12	4.04E-23	8.57E-11	296	C	5.24E-12	1.11E-22	5.00E-11	274	C
P7-S2	6.49E-13	2.64E-24	3.20E-11	261	C	7.92E-13	1.29E-24	5.15E-12	219	C
P7-S3	5.70E-03	6.08E-03	2.80E-03	872	R	1.13E-12	3.84E-24	1.91E-12	294	C
P7-S4	5.70E-03	6.08E-03	2.80E-03	806	R	5.70E-03	6.08E-03	2.03E-03	464	R
P7-S5	7.81E-15	1.52E-25	1.07E-12	199	C	2.54E-14	5.08E-26	4.78E-12	227	C
Avg.	2.58E-03	7.80E-02	9.67E-03	2093.69	-	1.88E-04	1.79E-04	5.81E-05	4115.94	-
Med.	3.11E-11	1.01E-20	3.08E-11	626	-	1.04E-11	9.51E-22	3.09E-11	898	-

Table 4.3: Optimization results for the problem in Equation 4.63, using configuration D3-p7-L4. The test run label format "Px-Sy" refers to the xth instantiated problem with the yth initial solution. "Error" is the relative error as defined by Equation 4.71. "Score" and "GM" are given by Equations 4.69 and 4.70, respectively. "Iter." is the number of iterations ran. "Exit" is the exit state, which can take on the values: "C" to mean gradient tolerance was reached, "R" to mean no valid updates after N_{stuck} consecutive iterations, and "T" to mean maximum iterations reached. "Avg." and "Med." are the average and median of the column, respectively.

	$G_{\mathcal{P}}$					$G_{\diamond}$				
	Error	Score	GM	Iter.	Exit	Error	Score	GM	Iter.	Exit
P1-S1	3.12E-15	6.97E-27	1.52E-11	185	C	9.76E-16	1.57E-27	1.09E-11	138	C
P1-S2	1.03E-15	1.57E-26	3.60E-11	115	C	4.75E-15	4.46E-26	5.87E-11	81	C
P1-S3	1.04E-14	6.97E-27	2.12E-11	166	C	4.54E-16	3.80E-28	9.64E-13	116	C
P1-S4	2.95E-16	3.43E-26	5.41E-11	115	C	1.68E-01	7.68E+00	1.76E+00	263	R
P1-S5	1.12E-15	9.11E-27	2.78E-11	26	C	1.91E-15	5.82E-27	2.16E-11	44	C
P2-S1	7.94E-15	1.80E-27	1.80E-14	339	C	4.05E-13	6.32E-23	6.81E-12	194	C
P2-S2	4.63E-15	4.89E-28	1.42E-14	126	C	5.60E-14	4.94E-26	6.83E-14	115	C
P2-S3	3.87E-12	3.19E-21	4.39E-11	402	C	9.95E-14	1.64E-25	1.48E-13	115	C
P2-S4	9.38E-15	1.41E-27	8.96E-15	119	C	1.02E-14	3.54E-27	2.49E-14	227	C
P2-S5	8.38E-12	3.59E-21	7.37E-11	226	C	2.72E-12	2.43E-20	8.52E-11	114	C
P3-S1	4.04E-14	2.11E-26	1.67E-12	57	C	9.84E-01	1.51E+01	7.02E-07	118	R
P3-S2	9.56E-14	9.86E-25	1.53E-11	43	C	1.29E-13	2.18E-24	2.28E-11	24	C
P3-S3	1.24E-01	1.28E+01	4.32E-08	94	R	6.15E-02	1.57E+01	6.78E-12	24	C
P3-S4	1.02E-01	1.31E+01	4.62E-08	114	R	9.05E-02	1.56E+01	9.90E-11	37	C
P3-S5	2.51E-13	3.53E-24	2.78E-11	57	C	1.05E-01	1.44E+01	2.71E-12	38	C
P4-S1	8.18E-12	1.00E-20	7.77E-11	114	C	7.94E-15	1.45E-27	2.82E-14	115	C
P4-S2	4.69E-15	2.41E-28	6.85E-15	23	C	2.97E-12	1.52E-21	3.12E-11	22	C
P4-S3	7.78E-12	1.31E-20	8.88E-11	23	C	3.81E-12	3.11E-21	4.34E-11	23	C
P4-S4	1.06E-13	9.10E-25	2.24E-13	80	C	2.26E-14	1.87E-26	1.49E-13	108	C
P4-S5	1.00E-14	1.23E-26	7.51E-14	23	C	2.99E-14	1.48E-25	2.54E-13	23	C
P5-S1	3.03E-16	5.89E-28	3.57E-12	98	C	6.65E-03	5.39E-03	6.79E-07	377	R
P5-S2	5.67E-16	1.20E-28	1.18E-12	58	C	3.84E-16	2.14E-27	1.01E-11	115	C
P5-S3	6.65E-03	5.39E-03	2.29E-09	108	R	6.25E-03	6.83E-03	3.32E-02	262	R
P5-S4	2.69E-13	3.95E-24	5.76E-11	173	C	6.09E-16	9.49E-28	3.76E-12	258	C
P5-S5	5.53E-16	1.18E-28	4.89E-13	39	C	3.39E-16	6.09E-29	4.39E-13	227	C
P6-S1	7.94E-02	1.13E+01	3.47E-12	63	C	1.20E-15	1.90E-28	2.77E-13	82	C
P6-S2	2.33E-01	1.02E+01	3.03E-12	42	C	2.33E-01	1.02E+01	4.10E-13	38	C
P6-S3	7.29E-15	2.89E-27	1.16E-12	20	C	1.00E+00	2.31E+01	8.35E-07	87	R
P6-S4	1.03E-12	2.05E-23	3.42E-11	100	C	3.94E-15	1.39E-27	1.08E-12	62	C
P6-S5	9.43E-15	9.82E-27	2.75E-12	29	C	5.35E-15	2.06E-27	1.31E-12	29	C
P7-S1	2.31E-02	1.19E+00	1.16E-11	202	C	2.31E-02	1.19E+00	1.39E-07	3932	R
P7-S2	2.70E-15	1.14E-26	3.66E-12	49	C	2.31E-02	1.19E+00	3.14E-07	85	R
P7-S3	1.23E-14	6.56E-27	3.51E-12	116	C	6.96E-15	8.11E-28	1.59E-13	203	C
P7-S4	2.52E-01	1.15E+01	2.97E-01	3072	R	2.91E-14	7.69E-25	9.83E-12	1054	C
P7-S5	2.31E-02	1.19E+00	5.85E-12	116	C	4.62E-15	4.60E-26	1.08E-11	109	C
Avg.	2.41E-02	1.75E+00	8.47E-03	192.3	-	7.72E-02	2.97E+00	5.11E-02	253.1	-
Med.	4.04E-14	3.43E-26	1.16E-11	100	-	5.60E-14	1.64E-25	9.83E-12	114	-

Table 4.4: Optimization results for the problem in Equation 4.63, using configuration D4-p3-L4. The test run label format "Px-Sy" refers to the xth instantiated problem with the yth initial solution. "Error" is the relative error as defined by Equation 4.71. "Score" and "GM" are given by Equations 4.69 and 4.70, respectively. "Iter." is the number of iterations ran. "Exit" is the exit state, which can take on the values: "C" to mean gradient tolerance was reached, "R" to mean no valid updates after N_{stuck} consecutive iterations, and "T" to mean maximum iterations reached. "Avg." and "Med." are the average and median of the column, respectively.

	$G_{\mathcal{P}}$					$G_{\diamond}$				
	Error	Score	GM	Iter.	Exit	Error	Score	GM	Iter.	Exit
P1-S1	2.31E-14	1.45E-26	1.54E-12	127	C	2.02E-14	2.95E-27	2.42E-13	136	C
P1-S2	4.16E-02	3.75E+00	1.62E-07	137	R	4.53E-15	1.71E-26	1.63E-12	41	C
P1-S3	4.87E-02	2.88E+00	3.73E-11	96	C	4.87E-02	2.88E+00	6.11E-12	78	C
P1-S4	4.87E-02	2.88E+00	6.53E-11	115	C	4.87E-02	2.88E+00	1.40E-12	77	C
P1-S5	3.91E-16	1.27E-28	8.96E-14	47	C	3.84E-14	1.16E-24	1.13E-11	51	C
P2-S1	4.16E-03	6.94E-04	6.56E-06	424	R	7.46E-11	1.65E-19	2.61E-11	143	C
P2-S2	3.02E-13	4.76E-24	1.42E-13	151	C	8.94E-12	3.48E-21	3.67E-13	214	C
P2-S3	7.79E-12	2.25E-21	3.40E-13	198	C	3.95E-03	7.02E-04	8.59E-06	20000	T
P2-S4	7.29E-13	1.49E-23	1.62E-13	7812	C	6.19E-12	2.56E-21	6.96E-14	117	C
P2-S5	1.91E-10	8.49E-19	6.75E-12	229	C	3.42E-03	6.93E-04	2.13E-11	128	C
P3-S1	4.72E-15	4.50E-28	6.99E-14	177	C	2.92E-03	1.90E-01	5.29E-09	319	R
P3-S2	2.98E-11	1.58E-20	1.66E-11	115	C	2.42E-13	1.18E-24	1.62E-13	138	C
P3-S3	3.96E-09	3.42E-16	8.34E-11	129	C	2.92E-03	1.90E-01	8.05E-09	201	R
P3-S4	3.67E-10	2.55E-18	4.02E-11	115	C	2.92E-03	1.90E-01	9.77E-11	137	C
P3-S5	9.32E-10	9.19E-18	1.44E-11	96	C	2.96E-09	1.68E-16	4.75E-11	483	C
P4-S1	2.13E-14	3.79E-26	6.42E-13	98	C	5.86E-15	2.81E-27	2.00E-13	179	C
P4-S2	9.84E-15	2.00E-26	5.19E-13	59	C	5.43E-03	3.42E-02	1.62E-04	574	R
P4-S3	9.05E-15	1.26E-26	2.28E-13	281	C	3.06E-14	8.47E-25	1.66E-12	64	C
P4-S4	5.43E-03	3.42E-02	1.72E-04	314	R	6.40E-15	9.50E-26	1.11E-12	111	C
P4-S5	5.43E-03	3.42E-02	1.69E-04	163	R	6.75E-13	9.46E-22	3.30E-11	108	C
P5-S1	1.00E+00	1.82E+01	5.88E-07	204	R	9.09E-14	2.14E-25	1.16E-12	269	C
P5-S2	9.14E-14	2.63E-24	4.49E-11	228	C	1.00E+00	1.81E+01	7.10E-07	135	R
P5-S3	6.73E-14	2.20E-25	2.22E-12	137	C	5.13E-15	2.83E-26	1.32E-12	142	C
P5-S4	1.99E-03	4.87E-01	1.95E-09	20000	T	1.80E-13	1.51E-24	3.06E-12	139	C
P5-S5	8.85E-15	1.48E-26	1.08E-12	229	C	4.59E-15	5.36E-27	4.49E-13	368	C
P6-S1	2.15E-15	7.46E-28	1.00E-13	130	C	4.14E-15	1.49E-28	2.48E-14	115	C
P6-S2	6.09E-15	4.77E-28	2.63E-14	92	C	9.18E-13	2.51E-22	1.55E-11	37	C
P6-S3	9.47E-14	5.74E-24	5.34E-12	85	C	5.55E-12	2.89E-21	5.51E-11	114	C
P6-S4	3.91E-11	1.17E-20	5.84E-11	123	C	3.95E-14	1.16E-26	1.12E-13	115	C
P6-S5	1.53E-15	3.17E-28	5.18E-14	69	C	9.57E-15	1.73E-27	1.22E-13	52	C
P7-S1	1.33E-15	2.33E-27	1.41E-12	253	C	1.11E-15	5.57E-28	4.21E-13	219	C
P7-S2	4.70E-14	6.19E-25	2.45E-12	97	C	3.07E-13	2.68E-23	1.02E-11	28	C
P7-S3	1.02E-15	1.73E-27	1.35E-12	259	C	4.47E-16	2.21E-28	5.97E-13	34	C
P7-S4	7.42E-16	1.46E-27	1.07E-12	114	C	9.95E-13	2.75E-22	3.09E-11	328	C
P7-S5	3.91E-13	4.26E-23	1.31E-11	60	C	9.17E-16	2.34E-27	1.48E-12	285	C
Avg.	3.30E-02	8.08E-01	9.96E-06	941.8		3.20E-02	7.00E-01	4.91E-06	733.69	
Med.	3.02E-13	5.74E-24	2.45E-12	130		3.07E-13	2.68E-23	1.66E-12	136	

Table 4.5: Optimization results for the problem in Equation 4.63, using configuration D4-p4-L4. The test run label format "Px-Sy" refers to the xth instantiated problem with the yth initial solution. "Error" is the relative error as defined by Equation 4.71. "Score" and "GM" are given by Equations 4.69 and 4.70, respectively. "Iter." is the number of iterations ran. "Exit" is the exit state, which can take on the values: "C" to mean gradient tolerance was reached, "R" to mean no valid updates after N_{stuck} consecutive iterations, and "T" to mean maximum iterations reached. "Avg." and "Med." are the average and median of the column, respectively.

	$G_{\mathcal{P}}$					$G_{\diamond}$				
	Error	Score	GM	Iter.	Exit	Error	Score	GM	Iter.	Exit
P1-S1	1.76E-02	5.37E+00	1.01E-02	518	R	1.76E-02	5.37E+00	1.02E-02	799	R
P1-S2	1.66E-02	5.25E+00	9.56E-03	201	R	1.72E-01	9.75E+00	1.64E-06	324	R
P1-S3	1.26E-01	7.74E+00	3.22E-06	1752	R	4.01E-14	1.27E-25	3.40E-12	187	C
P1-S4	4.52E-14	3.41E-26	2.44E-12	126	C	3.17E-01	3.19E+01	9.98E-02	11598	R
P1-S5	1.52E-01	1.64E+01	8.79E-07	234	R	3.07E-01	2.46E+01	7.24E-02	558	R
P2-S1	8.28E-08	1.02E-13	8.23E-11	2952	C	3.58E-07	6.42E-12	4.94E-11	2966	C
P2-S2	7.04E-07	1.80E-11	9.56E-11	2707	C	3.36E-07	7.60E-12	8.19E-11	4677	C
P2-S3	5.62E-07	6.49E-12	6.41E-11	2849	C	1.76E-07	8.83E-12	8.61E-11	675	C
P2-S4	6.12E-07	1.54E-11	7.52E-11	1914	C	3.71E-07	2.92E-11	6.17E-11	3992	C
P2-S5	6.68E-07	1.67E-11	9.52E-11	13498	C	9.97E-07	4.84E-11	8.31E-11	1482	C
P3-S1	9.97E-01	3.66E+01	1.46E-02	670	R	9.97E-01	3.67E+01	1.21E-02	1076	R
P3-S2	9.97E-01	3.59E+01	1.27E-02	20000	T	9.97E-01	3.67E+01	1.29E-02	401	R
P3-S3	9.97E-01	3.59E+01	1.30E-02	484	R	9.97E-01	3.68E+01	1.22E-02	572	R
P3-S4	9.97E-01	3.59E+01	1.30E-02	1073	R	9.97E-01	3.59E+01	1.30E-02	2175	R
P3-S5	9.97E-01	3.66E+01	1.44E-02	6833	R	9.97E-01	4.35E+01	1.68E-02	1287	R
P4-S1	1.12E-12	2.22E-20	6.37E-11	2155	C	6.80E-14	1.74E-24	9.09E-12	424	C
P4-S2	5.56E-15	3.90E-25	8.35E-13	2163	C	4.85E-02	8.79E+00	2.13E-07	5395	R
P4-S3	3.24E-14	2.50E-25	3.29E-13	227	C	4.85E-02	8.79E+00	1.08E-06	2106	R
P4-S4	5.78E-15	4.84E-26	1.52E-12	339	C	5.30E-02	1.04E+01	8.32E-08	20000	T
P4-S5	7.60E-16	1.42E-27	2.39E-13	341	C	9.12E-14	2.07E-24	1.26E-11	269	C
P5-S1	1.47E-12	1.83E-21	2.44E-10	449	R	1.95E-09	3.30E-17	5.51E-11	491	C
P5-S2	1.16E-03	5.97E-01	4.42E-08	20000	T	5.69E-02	1.30E+01	2.23E-06	20000	T
P5-S3	1.35E-09	1.58E-17	6.03E-11	495	C	1.93E-10	3.23E-19	3.42E-12	1433	C
P5-S4	1.33E-11	1.57E-21	4.90E-11	303	C	4.91E-09	2.09E-16	5.80E-11	386	C
P5-S5	5.00E-03	5.97E-01	8.64E-09	1148	R	2.95E-10	7.50E-19	9.51E-11	1525	C
P6-S1	1.20E-14	6.04E-27	4.23E-14	100	C	7.84E-03	7.56E-01	1.65E-08	181	R
P6-S2	6.65E-14	4.08E-26	8.11E-14	54	C	1.01E-13	9.68E-26	1.16E-13	37	C
P6-S3	1.58E-14	1.90E-27	4.30E-14	140	C	2.01E-14	6.69E-27	8.98E-14	115	C
P6-S4	4.63E-02	2.46E+00	1.99E-02	145	R	2.40E-14	6.39E-27	7.33E-14	99	C
P6-S5	6.93E-14	4.04E-26	8.19E-14	193	C	6.90E-14	4.03E-26	6.75E-14	55	C
P7-S1	3.48E-02	2.51E+00	8.92E-06	378	R	1.51E-02	2.48E+00	2.09E-05	477	R
P7-S2	1.09E-14	2.69E-26	4.34E-11	466	C	1.26E-03	1.22E-03	2.52E-06	1244	R
P7-S3	4.57E-15	1.31E-26	3.60E-11	252	C	7.73E-04	1.63E-03	8.58E-06	505	R
P7-S4	1.29E-03	1.68E-03	9.79E-04	20000	T	1.26E-03	1.22E-03	3.07E-07	297	R
P7-S5	1.28E-03	1.67E-03	6.13E-07	347	R	4.77E-15	1.69E-26	2.31E-11	251	C
Avg.	1.54E-01	6.34E+00	3.09E-03	3014.46		1.72E-01	8.72E+00	7.12E-03	2516.97	
Med.	6.12E-07	1.54E-11	9.52E-11	484		7.73E-04	1.22E-03	1.65E-08	572	

Table 4.6: Optimization results for the problem in Equation 4.63, using configuration D4-p4-L6. The test run label format "Px-Sy" refers to the xth instantiated problem with the yth initial solution. "Error" is the relative error as defined by Equation 4.71. "Score" and "GM" are given by Equations 4.69 and 4.70, respectively. "Iter." is the number of iterations ran. "Exit" is the exit state, which can take on the values: "C" to mean gradient tolerance was reached, "R" to mean no valid updates after N_{stuck} consecutive iterations, and "T" to mean maximum iterations reached. "Avg." and "Med." are the average and median of the column, respectively.

Configuration	Runs with qH relative errors			Median qH	Average qH	Minimum qH	Maximum qH
	$< 1.0 \times 10^{-1}$	$< 1.0 \times 10^{-2}$	$< 1.0 \times 10^{-5}$	relative error	relative error	relative error	relative error
D3-L4-p2-$G_{\mathcal{P}}$	**33**	**33**	**33**	1.09E-14	**5.71E-02**	6.98E-16	**1.00**
D3-L4-p2-$G_{\diamond}$	32	32	32	**9.87E-15**	8.57E-02	**2.97E-16**	**1.00**
D3-L6-p4-$G_{\mathcal{P}}$	**33**	31	**20**	**2.00E-11**	6.04E-02	2.99E-15	**1.00**
D3-L6-p4-$G_{\diamond}$	**33**	**33**	**20**	3.07E-10	**5.72E-02**	**1.97E-15**	**1.00**
D3-L4-p7-$G_{\mathcal{P}}$	**35**	34	**29**	3.11E-11	2.58E-03	**7.81E-15**	7.85E-02
D3-L4-p7-$G_{\diamond}$	**35**	**35**	**29**	**1.04E-11**	**1.89E-04**	9.65E-15	**5.70E-03**
D4-L4-p3-$G_{\mathcal{P}}$	**31**	**28**	**27**	**4.04E-14**	**2.41E-02**	**2.95E-16**	**2.52E-01**
D4-L4-p3-$G_{\diamond}$	30	26	24	5.60E-14	7.72E-02	3.39E-16	1.00
D4-L4-p4-$G_{\mathcal{P}}$	**34**	31	**27**	**3.02E-13**	3.30E-02	**3.91E-16**	**1.00**
D4-L4-p4-$G_{\diamond}$	**34**	**32**	26	3.07E-13	**3.20E-02**	4.47E-16	**1.00**
D4-L6-p4-$G_{\mathcal{P}}$	**28**	**24**	**20**	**6.12E-07**	**1.54E-01**	**7.60E-16**	**9.97E-01**
D4-L6-p4-$G_{\diamond}$	27	21	17	7.73E-04	1.72E-01	4.77E-15	**9.97E-01**

Table 4.7: Summary of the numerical results on the optimization problem in Equation 4.63. All errors reported here are relative errors for quasi-Hankel matrices (defined in Equation 4.71); the abbreviation "qH" in the column labels stands for "quasi-Hankel matrix". The bold entry in each column within a configuration indicate the best performing optimization scheme (one per row within a configuration) for the corresponding performance metric in the column. The schemes are: $G_{\mathcal{P}}$ for the proposed optimizer that uses the Riemannian metric $G_{\mathcal{P}}$, and $G_{\diamond}$ for the proposed optimizer that uses the approximation bilinear form $G_{\diamond}$.

4.7.2.2 Comparison with the CP-ALS Algorithm

Here, we compare the results of the proposed schemes with the CP-ALS algorithm, which is a well-known CP tensor decomposition method [33, 26]. This comparison is possible for the proposed schemes since our chosen optimization problem for this demonstration can be interpreted as performing a CP tensor decomposition problem. However, the CP-ALS algorithm requires a tensor as input instead of a quasi-Hankel matrix, so the optimization problem stated in Equation 4.63 cannot be solved by such an algorithm. Instead, we constructed a symmetric tensor J_{ALS} for each test run by taking only the first row of the quasi-Hankel matrix $\mathtt{H}$ that is isomorphic to the ground truth rank-one decomposition.

The same configurations from Section 4.7.2.1 were used for this comparison. Since the same ground truth rank-one decompositions and initial solutions from Appendix C were used, the recovered rank-one decomposition solutions for both versions of the proposed optimizer were the same as the ones recovered in Section 4.7.2.1. However, their relative errors reported here are not the same as the ones reported in that section because they are computed differently. Here, we used the tensor J_{qH} that corresponds to the first column of the recovered rank-one decomposition $A^* \cong (\mathfrak{z}, c)$ to compute a relative error for a test run in this section, whereas the quasi-Hankel matrix $\phi(A^*)\phi(A^*)^T$ was used to compute the relative error in Section 4.7.2.1. This definition of relative error is used here, so that a sensible comparison between the proposed optimizer and the CP-ALS algorithm can be made.

The relative errors for the results in Tables 4.8 and 4.9 from both versions of the proposed optimizer are given by

$$\frac{\|J_{\text{qH}} - J\|_2}{\|J\|_2} \tag{4.72}$$

where J is the true tensor in component-form (with respect to the elementary basis), and is obtained from the given quasi-Hankel matrix $\mathtt{H}$ by using Equation 2.93 on the first column of the quasi-Hankel matrix $\phi(A^*)\phi(A^*)^T$ to get J_{qH}. The relative errors for the results in Tables 4.8 and 4.9 from the CP-ALS algorithm are given by

$$\frac{\|J_{\text{ALS}} - J\|_2}{\|J\|_2}. \tag{4.73}$$

The CP-ALS algorithm is significantly faster than the proposed optimizer, but it could converge to a non-stationary point [33]. The goal of the comparison here is to investigate whether the two versions of the proposed optimizer are more accurate than CP-ALS, since they should not get stuck as often as the CP-ALS. The bold entries in Tables 4.8 and 4.9 correspond to the method that produced the lowest relative error for a given test run.

We see from Table 4.8 that for most cases, the relative errors that correspond to both versions of the proposed optimizer are significantly lower than the ones that corresponds to the CP-ALS algorithm. The situation is less favourable for both versions of the proposed optimizers when D is increased to 4, or when L is increased to 6; for example, see the last three columns of Table 4.6. I suspect numerical issues with the proposed optimizer may be the reason behind this behaviour. However, there are still many test runs in the $D = 4$ or $L = 6$ configurations where the proposed optimizer produced a significantly more accurate solution than the one offered by CP-ALS.

Table 4.10 summarizes the results presented in Tables 4.8 and 4.9. Each row corresponds to one of the proposed schemes on a particular configuration, which corresponds to a triplet of rows that are vertically connected in this table. In Table 4.10, the single bold font for each column indicates the scheme that performed better; a lower relative error or a higher number of runs that achieves a certain level of relative error makes a scheme better than the others.

	D3-p2-L4			D3-p4-L6			D3-p7-L4		
	$G_{\mathcal{P}}$	$G_{\diamond}$	CP-ALS	$G_{\mathcal{P}}$	$G_{\diamond}$	CP-ALS	$G_{\mathcal{P}}$	$G_{\diamond}$	CP-ALS
P1-S1	1.31E-15	**7.25E-16**	4.27E-05	5.89E-14	**2.34E-14**	1.02E-02	**6.70E-13**	3.62E-12	1.59E-02
P1-S2	9.02E-01	9.02E-01	**2.18E-05**	**7.61E-14**	1.76E-13	1.07E-02	**6.06E-13**	8.08E-12	5.29E-03
P1-S3	9.02E-01	9.02E-01	**4.24E-05**	**8.11E-14**	3.91E-13	2.76E-02	5.40E-12	**3.52E-13**	5.25E-03
P1-S4	7.02E-12	**6.92E-16**	5.65E-05	**4.06E-14**	1.01E-12	7.53E-03	1.75E-10	**1.41E-12**	6.13E-04
P1-S5	**1.79E-12**	9.02E-01	4.11E-05	1.30E-13	**8.57E-14**	9.76E-03	8.14E-11	**2.21E-12**	3.66E-03
P2-S1	1.70E-15	**4.01E-16**	7.89E-02	**3.62E-15**	1.25E-14	2.02E-03	9.51E-15	**8.79E-15**	2.40E-02
P2-S2	**4.34E-16**	1.88E-14	2.28E-06	4.31E-14	**2.38E-14**	1.01E-02	4.49E-14	**2.84E-14**	1.30E-02
P2-S3	**4.45E-16**	7.77E-16	2.23E-06	2.88E-13	**6.12E-14**	3.48E-03	1.10E-12	**7.54E-15**	2.54E-02
P2-S4	1.56E-15	**7.98E-16**	8.74E-07	1.81E-14	**1.15E-14**	6.62E-03	**9.67E-15**	3.15E-12	4.99E-03
P2-S5	9.30E-16	**5.36E-16**	7.84E-02	4.13E-14	**2.08E-14**	2.82E-03	3.79E-14	**1.47E-14**	1.22E-02
P3-S1	1.12E-14	**8.78E-16**	7.72E-06	**6.16E-13**	1.44E-03	9.31E-03	**7.16E-13**	2.94E-03	6.51E-03
P3-S2	**2.41E-15**	2.92E-15	9.56E-06	**2.16E-15**	1.29E-13	2.88E-02	**1.62E-11**	3.73E-03	3.97E-03
P3-S3	1.73E-14	**9.09E-16**	2.34E-01	**1.44E-03**	**1.44E-03**	2.98E-03	3.73E-03	**1.58E-12**	2.91E-02
P3-S4	2.74E-12	**1.84E-16**	1.75E-05	1.44E-03	**1.72E-11**	4.70E-03	**1.88E-12**	3.73E-03	4.41E-03
P3-S5	2.61E-12	**2.87E-15**	1.26E-05	1.44E-03	**1.58E-15**	5.60E-03	3.74E-03	**1.64E-13**	3.24E-02
P4-S1	**4.77E-11**	5.34E-10	2.40E-04	**2.35E-12**	9.59E-08	1.52E-03	**3.02E-06**	3.03E-06	6.45E-03
P4-S2	**2.21E-15**	4.26E-10	2.31E-04	9.13E-02	9.13E-02	**1.54E-02**	**1.77E-07**	3.14E-07	1.01E-02
P4-S3	2.83E-09	**7.78E-11**	2.40E-04	**1.20E-08**	5.21E-08	8.16E-04	3.02E-06	**8.50E-08**	2.84E-03
P4-S4	**1.08E-09**	1.42E-08	2.44E-04	9.13E-02	9.13E-02	**2.41E-04**	**1.78E-07**	2.62E-06	1.02E-02
P4-S5	**1.02E-13**	3.77E-09	2.40E-04	**1.19E-09**	1.83E-11	2.38E-03	1.87E-07	**8.01E-08**	1.18E-02
P5-S1	**7.33E-16**	2.00E-13	4.00E-09	**1.57E-03**	**1.57E-03**	2.51E-02	3.08E-03	**5.64E-05**	5.07E-03
P5-S2	**4.24E-16**	7.59E-16	8.31E-01	1.57E-03	**8.46E-15**	3.65E-02	5.68E-05	**3.23E-05**	4.48E-03
P5-S3	**8.44E-16**	1.32E-15	1.81E-11	**1.50E-14**	6.72E-14	3.61E-03	**1.42E-11**	3.08E-03	4.66E-03
P5-S4	9.48E-16	**7.58E-16**	5.60E-09	**1.57E-03**	**1.57E-03**	5.74E-03	**3.01E-12**	5.68E-05	4.27E-03
P5-S5	5.52E-16	**3.19E-16**	5.57E-01	1.57E-03	**2.70E-15**	3.08E-02	**5.67E-05**	5.70E-05	4.33E-03
P6-S1	**2.26E-14**	6.88E-10	1.79E-06	**5.38E-14**	4.78E-04	8.65E-03	2.82E-01	**3.42E-13**	1.26E-02
P6-S2	**8.19E-14**	1.51E-11	2.62E-06	7.61E-02	**4.78E-04**	7.56E-03	**1.71E-14**	1.14E-11	1.26E-02
P6-S3	**5.60E-15**	1.26E-11	2.68E-06	**1.83E-04**	**1.83E-04**	6.71E-03	2.03E-12	**3.49E-13**	2.15E-02
P6-S4	**6.14E-12**	4.52E-11	9.97E-06	**1.83E-04**	4.45E-04	1.74E-03	**2.26E-14**	3.40E-12	9.31E-03
P6-S5	**1.76E-14**	6.92E-13	9.06E-06	6.64E-01	**1.83E-04**	2.94E-03	6.02E-12	**8.49E-14**	3.73E-02
P7-S1	**4.45E-15**	7.43E-12	8.56E-03	**2.20E-15**	1.76E-04	3.87E-03	**1.12E-14**	1.71E-12	1.15E-02
P7-S2	9.50E-13	**1.57E-15**	8.59E-03	**3.09E-12**	3.06E-04	7.03E-03	**6.22E-15**	8.88E-14	2.40E-02
P7-S3	**2.06E-14**	4.47E-14	8.67E-03	**3.23E-04**	**3.23E-04**	4.22E-03	1.46E-02	**1.36E-14**	9.09E-03
P7-S4	3.09E-15	**1.32E-15**	8.46E-03	3.06E-04	**4.47E-14**	1.60E-03	1.46E-02	1.46E-02	**3.81E-03**
P7-S5	**1.17E-15**	1.78E-15	8.79E-03	**2.55E-12**	3.23E-04	9.03E-03	8.94E-14	**5.00E-14**	2.77E-02
Avg.	**5.15E-02**	7.73E-02	5.21E-02	2.67E-02	**5.47E-03**	9.08E-03	9.18E-03	**8.07E-04**	1.20E-02
Med.	1.12E-14	**2.92E-15**	4.24E-05	**3.09E-12**	1.83E-11	6.62E-03	6.02E-12	**3.40E-12**	9.31E-03

Table 4.8: The relative errors of both versions of the proposed optimizer and the CP-ALS algorithm, for the configurations D3-p2-L4, D3-p4-L6, and D3-p7-L4. The test run label format "Px-Sy" refers to the xth instantiated problem with the yth initial solution. "Error" is the relative error of tensors as defined by Equation 4.72. "Avg." and "Med." are the average and median of the column, respectively. The bold entries correspond to the method that produced the lowest relative error for a test run.

	D4-p3-L4			D4-p4-L4			D4-p4-L6		
	$G_{\mathcal{P}}$	$G_{\diamond}$	CP-ALS	$G_{\mathcal{P}}$	$G_{\diamond}$	CP-ALS	$G_{\mathcal{P}}$	$G_{\diamond}$	CP-ALS
P1-S1	5.71E-15	**1.37E-15**	2.04E-02	**1.19E-15**	1.40E-15	2.04E-03	6.14E-01	6.14E-01	**8.98E-04**
P1-S2	**1.08E-15**	1.23E-15	2.02E-02	3.21E-01	**8.64E-16**	5.88E-03	6.15E-01	4.18E-01	**1.23E-01**
P1-S3	1.42E-15	**1.08E-15**	1.33E-02	3.98E-01	3.98E-01	**2.45E-03**	7.25E-02	**2.85E-14**	1.24E-03
P1-S4	**5.23E-16**	2.92E-01	6.94E-02	3.98E-01	3.98E-01	2.02E-03	**1.02E-14**	4.26E-01	9.27E-04
P1-S5	**5.34E-16**	6.42E-16	3.10E-03	**1.17E-15**	1.18E-13	2.36E-03	1.01E-01	4.35E-01	**1.02E-03**
P2-S1	**2.63E-15**	8.21E-13	4.24E-04	2.50E-03	**6.86E-11**	2.76E-02	**3.43E-08**	3.83E-07	3.41E-03
P2-S2	**1.70E-15**	1.81E-14	1.54E-04	**3.84E-13**	1.27E-11	2.97E-03	5.93E-07	**4.00E-07**	5.09E-03
P2-S3	6.49E-12	**3.32E-14**	4.26E-04	**1.10E-11**	2.52E-03	1.03E-03	**3.06E-07**	5.12E-07	2.57E-03
P2-S4	**4.46E-15**	4.93E-15	1.59E-04	**4.79E-13**	1.19E-11	2.76E-02	**5.66E-07**	9.25E-07	9.20E-03
P2-S5	**1.47E-11**	1.58E-11	2.03E-04	**1.89E-10**	2.67E-03	8.31E-04	**5.66E-07**	1.02E-06	5.04E-03
P3-S1	**1.63E-15**	3.58E-01	9.86E-05	**7.45E-16**	9.87E-02	7.84E-02	2.38E-01	2.09E-01	**8.60E-02**
P3-S2	**1.78E-14**	1.89E-14	7.26E-05	4.79E-13	**3.67E-15**	5.59E-02	1.02E-01	2.05E-01	**1.87E-02**
P3-S3	9.04E-01	5.31E-01	**9.23E-02**	**1.60E-10**	9.87E-02	1.30E-02	1.02E-01	2.44E-01	**1.64E-02**
P3-S4	8.95E-01	6.41E-01	**7.62E-05**	**1.76E-11**	9.87E-02	8.64E-03	1.02E-01	1.02E-01	**1.83E-02**
P3-S5	**8.57E-15**	9.78E-01	8.79E-02	**4.82E-11**	1.56E-10	3.73E-03	2.39E-01	9.92E-01	**1.28E-03**
P4-S1	2.84E-11	**3.00E-15**	2.27E-03	6.83E-15	**1.94E-15**	1.72E-01	6.58E-12	**1.40E-14**	3.97E-05
P4-S2	**1.52E-15**	1.24E-11	5.68E-03	**2.30E-15**	5.49E-03	1.55E-03	**2.61E-14**	4.23E-01	3.72E-05
P4-S3	3.56E-11	**1.73E-11**	9.87E-03	**6.78E-15**	4.76E-14	8.71E-04	**2.37E-14**	4.23E-01	1.38E-04
P4-S4	1.84E-13	**3.48E-14**	3.10E-03	5.49E-03	**1.59E-14**	1.06E-03	**1.89E-15**	8.17E-01	6.94E-05
P4-S5	**3.32E-14**	1.18E-13	1.54E-03	5.49E-03	1.**72E-12**	9.04E-04	**6.62E-16**	2.19E-14	1.02E-04
P5-S1	**3.82E-16**	1.02E-02	3.70E-03	8.68E-01	**7.91E-14**	2.55E-02	**3.26E-12**	3.10E-10	1.39E-02
P5-S2	5.70E-16	**5.64E-16**	4.07E-03	**4.61E-14**	8.69E-01	2.41E-02	1.60E-02	2.46E-01	**1.42E-02**
P5-S3	1.02E-02	8.88E-03	**3.67E-03**	6.42E-14	**1.60E-14**	2.36E-02	2.15E-10	3.04E-11	1.37E-02
P5-S4	2.68E-13	**8.09E-16**	4.74E-03	2.76E-02	**3.48E-13**	3.07E-02	**1.80E-12**	7.82E-10	6.94E-03
P5-S5	6.45E-16	**3.93E-16**	1.70E-03	8.06E-15	**6.26E-15**	2.46E-02	1.48E-02	**4.53E-11**	1.31E-02
P6-S1	1.63E-01	**2.40E-15**	3.67E-04	1.99E-15	**7.89E-16**	8.76E-02	**9.76E-16**	8.45E-02	2.11E-02
P6-S2	8.64E-01	8.64E-01	**3.77E-04**	**3.68E-16**	3.13E-12	2.32E-02	**1.69E-15**	2.44E-15	7.08E-02
P6-S3	**2.20E-15**	9.84E-01	1.52E-01	**2.92E-13**	1.40E-11	4.02E-02	**8.38E-16**	2.87E-15	7.73E-02
P6-S4	1.26E-12	**8.76E-16**	1.52E-01	1.11E-11	**4.41E-15**	8.30E-03	1.37E-01	**4.23E-15**	6.80E-04
P6-S5	2.39E-15	**1.12E-15**	3.75E-04	2.61E-15	**1.78E-15**	3.75E-02	4.23E-15	**1.47E-15**	7.04E-02
P7-S1	1.18E-01	1.18E-01	**7.00E-04**	**6.44E-16**	1.27E-15	4.25E-03	3.79E-02	**1.14E-02**	1.72E-02
P7-S2	**2.48E-15**	1.18E-01	1.02E-03	**5.52E-14**	3.64E-13	4.26E-03	**1.28E-14**	1.67E-03	1.85E-02
P7-S3	**1.69E-15**	3.02E-15	1.03E-03	6.44E-16	**4.52E-16**	4.24E-03	**4.25E-15**	1.01E-03	1.68E-02
P7-S4	4.24E-01	**4.37E-14**	8.90E-04	**9.85E-16**	1.18E-12	4.24E-03	**1.27E-03**	1.67E-03	1.85E-02
P7-S5	1.18E-01	**1.20E-15**	1.00E-03	4.62E-13	**5.18E-16**	4.25E-03	1.25E-03	**5.75E-15**	7.93E-04
Avg.	9.99E-02	1.40E-01	**1.88E-02**	5.79E-02	5.64E-02	**2.16E-02**	6.84E-02	1.62E-01	**1.91E-02**
Med.	**8.57E-15**	3.48E-14	1.70E-03	3.84E-13	**3.64E-13**	5.88E-03	**5.66E-07**	1.01E-03	9.20E-03

Table 4.9: The relative errors of both versions of the proposed optimizer and the CP-ALS algorithm, for the configurations D4-p3-L4, D4-p4-L4, and D4-p4-L6. The test run label format "Px-Sy" refers to the xth instantiated problem with the yth initial solution. "Error" is the relative error of tensors as defined by Equation 4.72. "Avg." and "Med." are the average and median of the column, respectively. The bold entries correspond to the method that produced the lowest relative error for a test run.

Configuration	Runs with tensor relative errors			Median tensor relative error	Average tensor relative error	Minimum tensor relative error	Maximum tensor relative error
	$< 1.0 \times 10^{-1}$	$< 1.0 \times 10^{-2}$	$< 1.0 \times 10^{-5}$				
D3-L4-p2-$G_{\mathcal{P}}$	**33**	**33**	**33**	1.12E-14	**5.15E-02**	4.24E-16	9.02E-01
D3-L4-p2-$G_{\diamond}$	32	32	32	**2.92E-15**	7.73E-02	**1.84E-16**	9.02E-01
D3-L4-p2-ALS	32	30	13	4.24E-05	5.21E-02	1.81E-11	**8.31E-01**
D3-L6-p4-$G_{\mathcal{P}}$	34	31	**20**	**3.09E-12**	2.67E-02	2.16E-15	6.64E-01
D3-L6-p4-$G_{\diamond}$	**35**	**33**	**20**	1.83E-11	**5.47E-03**	**1.58E-15**	9.13E-02
D3-L6-p4-ALS	**35**	26	0	6.62E-03	9.08E-03	2.41E-04	**3.65E-02**
D3-L4-p7-$G_{\mathcal{P}}$	34	32	**27**	6.02E-12	9.20E-03	**6.22E-15**	2.82E-01
D3-L4-p7-$G_{\diamond}$	**35**	**34**	26	**3.4E-12**	**8.08E-04**	7.54E-15	**1.46E-02**
D3-L4-p7-ALS	**35**	18	0	9.31E-03	1.20E-02	6.13E-04	3.73E-02
D4-L4-p3-$G_{\mathcal{P}}$	28	**27**	**27**	**7.14E-15**	9.94E-02	**3.82E-16**	9.04E-01
D4-L4-p3-$G_{\diamond}$	25	24	23	3.92E-14	1.44E-01	3.93E-16	9.84E-01
D4-L4-p3-ALS	**32**	26	0	1.98E-03	**1.93E-02**	7.26E-05	**1.52E-01**
D4-L4-p4-$G_{\mathcal{P}}$	30	**29**	**26**	**3.38E-13**	5.96E-02	**3.68E-16**	8.68E-01
D4-L4-p4-$G_{\diamond}$	31	28	25	7.72E-13	5.80E-02	4.52E-16	8.69E-01
D4-L4-p4-ALS	**33**	19	0	7.09E-03	**2.22E-02**	8.31E-04	**1.72E-01**
D4-L6-p4-$G_{\mathcal{P}}$	25	**21**	**20**	**4.36E-07**	**7.04E-02**	**6.62E-16**	6.15E-01
D4-L6-p4-$G_{\diamond}$	21	19	16	1.34E-03	1.66E-01	1.47E-15	9.92E-01
D4-L6-p4-ALS	**34**	18	0	1.12E-02	1.96E-02	3.72E-05	**1.23E-01**

Table 4.10: Summary of the numerical results on the optimization problem in Equation 4.63 with the numerical results from CP-ALS on the symmetric tensor decomposition problem. All errors reported here are relative errors for symmetric tensors (defined in Equation 4.72). The bold entry in each column within a configuration indicate the best performing optimization scheme (one per row within a configuration) for the corresponding performance metric in the column. The schemes are: $G_{\mathcal{P}}$ for the proposed optimizer that uses the Riemannian metric $G_{\mathcal{P}}$, $G_{\diamond}$ for the proposed optimizer that uses the approximation bilinear form $G_{\diamond}$, and the CP-ALS method as a control method.

4.7.3 Discussion

The tables of Section 4.7.2.1 show many cases where one version of the proposed optimizer significantly outperformed the other. This suggests that an adaptive scheme that switches between the two versions may offer a good compromise. Some examples of when $G_\diamond$ significantly outperformed $G_\mathcal{P}$ are the P2-S4 and P5-S2 runs from Table 4.2, the P5-S4 run from Table 4.5, and the P2-S5 run from Table 4.6. From Table 4.7, we see that the version that uses $G_\diamond$ had a similar level of performance as the version that uses $G_\mathcal{P}$ for the three-dimensional configurations, despite $G_\diamond$ not being a proper Riemannian metric. However, we can see the stress of solving larger Lyapunov equations had lowered the performance $G_\diamond$ when D or L is increased. For this reason, we conclude that $G_\mathcal{P}$ should be used for general-purpose optimization over $\mathcal{S}_\text{E}$.

There are also a few instances where the gradient tolerance was reached, but the correct answer was not obtained; i.e., the objective score at termination was not close to zero. Some examples are the P1-S2 and P1-S3 runs from Table 4.1, the P3-S3 and P3-S4 runs from Table 4.4, P1-S3 and P1-S4 runs from Table 4.5, and P1-S1 and P1-S5 and P7-S1 runs from Table 4.6. This suggests that the optimizer converged to a stationary-point that is not the global minimum. Saddle points are known to cause problems in derivative-based optimization schemes [84]. Another possibility is that there are numerical errors in the computation of our definition of the gradient length, which is not a true distance metric when $G_\diamond$ is used. Numerical errors are also likely to arise when the magnitude of the gradient approaches zero, since the gradient is likely to be ill-conditioned. One can adjust the gradient tolerance to reduce the chances of a false-convergence, or devise a more appropriate termination conditions that rely on the history of past gradients or solution updates to reduce the effect of numerical errors. Another idea is to devise saddle point detection schemes based on the presence of negative and positive eigenvalues of the Hessian approximation $\tilde{H}$ [84]. If we were designing an optimizer that only solves CP decomposition problems, we could modify the proposed optimizer such that it does not terminate until both the gradient length and the objective score is near zero. This is because we know *a priori* that a solution to a CP decomposition problem has to attain a cost of zero. We did not do this in our benchmarks because in a general-purpose setting, we do not know the optimal objective score. These potential remedies are interesting directions for future explorations.

The demonstrations in this chapter show a lot of variability across different test runs for a given configuration. Recall from Equations 4.13 and 4.7 that $\bar{c}_j$ is proportional to the coordinate $u_1^{(j)}$ that was used to dehomogenize the homogeneous polynomial in Equation 4.5. I suspect that there are some numerical issues with the proposed optimizer when it has to work with solution updates $A \in \mathcal{P}$ that have entries in the first row that are close to zero. This affects the numerical conditioning of computations that involve $\phi_{\mathcal{P},\mathcal{N}}$, since numbers are being divided by the entries in the first row of A. These division operations originate from the dehomogenization procedure, which is briefly reviewed in Section 4.1.1, and in more detail in Section 2.14.3. Recall that this procedure is a division by the first coordinate of u from Equation 2.91. It is interesting to see whether one can avoid having to dehomogenize a homogeneous polynomial in our setup to avoid this problem. Another approach is to find a manifold that is isomorphic to $\mathcal{P}$ such that one avoids this division operation. I recommend further investigation on this issue for future works.

The proposed optimizer could be used to numerically solve modified versions of the CP decomposition problem that are well-posed. From our benchmarks, we see that the performance of the proposed optimizer is sensitive to the initial solution. The results in Section 4.7.2.2 suggest that the proposed optimizer requires additional tweaking if it were to be used as an CP decomposition algorithm. Since it only accepts updates that are better than the current solution, one should initialize the proposed optimizer with CP-ALS or another computationally inexpensive CP decomposition scheme. This was not done in our demonstrations because we wanted a fair comparison between the proposed optimizer and the CP-ALS algorithm, so the same initial solutions were used for both schemes. Since the CP-ALS algorithm runs faster than the proposed optimizer, an interesting extension of the work here is to compute a few iterations of the proposed optimizer when the CP-ALS algorithm is stuck before restarting the CP-ALS, or devise some adaptive scheme that switches

between the CP-ALS algorithm and the proposed optimizer.

In summary, we empirically demonstrated that the manifold-related derivations from Section 4.5 can be used by the proposed optimizer in Section 4.6 to minimize a cost function over $\mathcal{S}_{\mathrm{E}}$. The quotient manifold-inspired bilinear form $G_{\diamond}$ was not as successful as I expected, but it did significantly out-perform the proper Riemannian metric $G_{\mathcal{P}}$ on several test runs. The fact that the version that used $G_{\diamond}$ still converged on many cases suggests that one might be able to achieve good empirical performance by using approximations of Riemannian structures in combination with direct-search optimization methods. This may yield computational savings, though we did not demonstrate it here. More work is required to adapt the optimizer proposed here to real-life applications.

CHAPTER 5

Conclusion

In this thesis, we covered two novel works. The first is a method to construct non-stationary kernels for RKHS regularization problems, and we referred to it as the *adaptive kernel method* in this thesis. It is designed for problems where prior knowledge on the signal of interest is available. It does not require one to solve large-scale optimization problems, but it does require one to specify a warp map that describes the prior assumptions that one makes on the signal. This accomplishes the primary objective for the first portion of this thesis, as laid out in Section 1.1.2.

In Section 3.4, the adaptive smoothing and monogenic adaptive smoothing strategies were presented as warp map examples. These strategies were designed to incorporate our chosen prior assumption that edge contours are smooth. These two strategies are both based on the collinearity analysis from Section 3.4.1, which returns the p-most dominant directions in some defined local neighbourhood of a pixel, for a given positive integer p. This accomplishes the secondary objective for the first portion of this thesis, as laid out in Section 1.1.2. When the observed images had enough information for the strategies to properly infer the presence of edge contours, the results from Section 3.4.4 indicate that these two strategies indeed produce up-converted images that have smooth edge contours. More work is required to reduce the effect of the warp map when there is high uncertainty on the inferred nature of the pattern in the image scene from the observed image.

We covered the second novel work in Chapter 4. We introduced a numerical optimization scheme over the space

$$\begin{aligned}\mathcal{S}_{\mathrm{E}}(D,\, L,\, p) = \{(\mathfrak{z},\, c)\},& \\ \mathfrak{z} \in \left\{z^{(j)} \in \mathbb{R}^D_{\mathrm{s}}\right\}_{j=1}^{p},\, \left(z^{(j)}\right)_1 = 1,\, z^{(j)} \neq bz^{(i)},\, \forall i,\, j \in [p],\, \forall b \in \mathbb{R},& \\ c \in \mathbb{R}^p_{++},\, L \text{ is even.}&\end{aligned}$$

The motives for this work were presented in Section 4.1. In Section 4.5, we showed that this space is isomorphic to the differentiable manifold $\mathcal{P}$, which is an embedded submanifold of the manifold of full-rank matrices. We also derived a quotient manifold $\mathcal{M}$ based on $\mathcal{S}_{\mathrm{E}}$, so that permutation-related non-identifiability issues do not exist there. Riemannian metrics and tangent space projectors were also derived. This accomplishes both of the primary objectives for the second portion of this thesis, as laid out in Section 1.2.3.

Two versions of an optimization scheme over $\mathcal{P}$ that uses these manifold-related objects were presented in Section 4.6: one uses the Riemannian metric $G_{\mathcal{P}}$, and the other uses a bilinear form $G_{\diamond}$ that approximates a Riemannian metric. A new direct-search optimization algorithm over any differentiable manifold that has

a retraction was presented in Section 4.6.2. This direct-search optimizer is based on the line-search equation from [1] and the Nelder-Mead algorithm. The empirical results from Section 4.7.2 suggest that both versions of the proposed optimization scheme over $\mathcal{S}_{\text{E}}$ are indeed numerical algorithms that can minimize the test optimization problem

$$\min_{A \in \mathcal{P}} \left(S. * \left(\mathtt{H} - \phi(A)\phi(A)^T \right) \right) \circ_{\text{F}} \left(S. * \left(\mathtt{H} - \phi(A)\phi(A)^T \right) \right),$$

where $.*$ denotes the per-element multiplication of two matrices, and the entries of S are non-negative. Here, $\mathtt{H}$ and $\phi(A) = \phi_{\mathcal{N},\mathcal{V}} \circ \phi_{\mathcal{P},\mathcal{N}}(A)$ are quasi-Hankel matrices, and $\phi_{\mathcal{P},\mathcal{N}}$ and $\phi_{\mathcal{P},\mathcal{V}}$ are defined in Equations 4.31 and 4.41, respectively. The optimal solution of this optimization problem coincides with the solution of the CP decomposition. An overview of the CP decomposition problem can be found in Sections 2.14, 4.1, and 4.3. This accomplishes the secondary portion of this thesis, as laid out in Section 1.2.3.

Contributions

The following is the list of contributions made in this thesis work, listed in the order that they appear in the thesis:

1. Adaptive kernel construction method (Section 3.3): The warp kernel family of kernel construction methods [10, 11] required solving numerically-intensive problems, and they did not specify the constraints under which the resultant kernel is symmetric and positive-definite. However, they promoted the idea that kernels can be constructed from other kernels by using function composition. The proposed adaptive kernel construction method of creating non-stationary kernels is based on this idea, but does not require solving numerically-intensive problems. The conditions under which the resultant kernel is symmetric and positive-definite is presented in Theorem 3.1.

2. Collinearity analysis for a given image patch (Section 3.4.1): This is a method to select p-most dominant directions for a given grayscale image patch, where p is a given finite positive integer. A similar work in the literature is the *higher-order structure tensor* framework from the works [14, 15], but it requires the computation of a rank-one decomposition of a symmetric tensor. The collinearity analysis in Section 3.4.1 only requires solving a k-means clustering algorithm of $k \leftarrow p$ clusters, thus it is faster to compute than the higher-order structure tensor-based schemes.

3. Two methods to do non-homogeneous image up-conversion by using adaptive kernels (Sections 3.4.2 and 3.4.3): These two methods are not competitive to existing grayscale image up-conversion problems in terms of robustness to scene content and computational speed. However, they produced remarkable up-conversion results that have smooth edge contours for image scenes that have lots of coarse high-contrast edges, such as the scene from Figure 3.9. For this reason, these two methods could serve as stepping-stones for adaptive kernel applications that deal with colour images and videos.

4. It was implied in Theorem 6.3 from [29] that rank-one decompositions are isomorphic to some spaces of quasi-Hankel matrices. I have identified one such space through Proposition 4.1.

5. The formulations of $\mathcal{S}_{\text{E}}$ as differentiable manifolds (Section 4.5): We showed how the space $\mathcal{S}_{\text{E}}$ can be formulated as the embedded differentiable submanifold $\mathcal{P}$, and as the quotient manifold $\mathcal{M}$. Riemannian structure and tangent space projectors were derived for $\mathcal{P}$ and $\mathcal{M}$, and a retraction was presented for $\mathcal{P}$ that is based on a retraction from [45] (which was reviewed in Example 2.87). To the best of my knowledge, no other treatment of the space of rank-one decompositions $\mathcal{S}_{\text{E}}$ as a manifold is reported in the literature. My derivation of these manifolds can serve as a learning aid for those who are unfamiliar with these subjects.

6. The extension of the line-search equation from [1] into a direct-search optimizer for optimization problems over differentiable manifolds that are equipped with retractions (Section 4.6.2): We used the Nelder-Mead direct-search optimization algorithm on an Euclidean optimization subproblem where the cost function is an extension of the line-search equation from Equation 2.67 (taken from Algorithm 1 of [1]). One can use the full-space search method from Section 4.6.2 without defining a Riemannian metric.

7. An optimization scheme over $\mathcal{S}_{\mathrm{E}}$ (Section 4.6.3): To the best of my knowledge, the most similar existing work on this subject is [44], but the mode-matrix representation of tensors was used there. The advantages and disadvantages of this representation over the one used in this thesis were discussed in Section 4.3. The proposed optimization scheme was over the non-quotient manifold $\mathcal{P}$ instead of $\mathcal{M}$ because a suitable retraction for $\mathcal{M}$ was not devised in this thesis. The empirical results from Section 4.7.2 suggest that an approximation bilinear form $G_{\diamond}$ to the Riemannian metric for $\mathcal{M}$ seems to perform well for low dimensions and tensor orders, but it suffers from numerical issues for larger dimensions and order.

Recommended Future Works

Future works on applying adaptive kernels to image processing should focus on warp map construction strategies that reduce the effect of the warp map when the chosen prior assumptions are in doubt. One possible extension is to include the case of non-trivial degradation of the observed signal to the observation model, so that adaptive kernels can be applied to deconvolution problems. Another direction is to extend the adaptive kernel method to colour images. This is possible since there are existing investigations into vector-valued kernels [7, 6]. The warp map construction strategies presented in this thesis were favourable for the sample-without-prefiltering signal model, so one potential application of the adaptive kernel method is to reduce colour-aliasing artefacts across time in colour video processing tasks. This is because the temporal dimension is usually not prefiltered during the acquisition of colour videos.

The proposed optimizer over $\mathcal{S}_{\mathrm{E}}$ experienced bad performance when the ground truth rank-one decomposition or initial solutions have weights (i.e., the λ variable in the (u, λ) parameterization as described in Section 4.1.1) that are close to zero, so future investigations into this issue might improve the performance of the proposed optimization framework. The evaluation of the proposed Riemannian metric and Hessian do not scale for large dimensions or tensor orders. Approximations of the Riemannian metric and Hessian may offer a solution. The results in Section 4.7.2.1 hint that bilinear forms that approximate Riemannian metrics, such as $G_{\diamond}$ from Section 4.5.5, may be a valid substitute for proper Riemannian metrics in numerical optimization under certain conditions. Future efforts on this issue may improve the applicability of the proposed methods to a wider range of problems.

Another interesting direction is to investigate hybrid techniques that combine the work done in Chapter 4 with conventional CP decomposition algorithms. For example, when a conventional CP tensor decomposition method gets stuck, one could run a few iterations of the proposed optimizer to obtain an initial solution for a restart of the decomposition method. The tricky task is to ensure such heuristics-based strategies achieve better performance than a scheme that does not use such restarts, and that they have a reasonable computational cost.

Our discussion in Section 4.1 promotes the idea that one should be working with quasi-Hankel matrices instead of symmetric tensors when one wants a unique solution and low computational storage. This is my motivation for working with rank-one decompositions. A useful future direction is to investigate how one can directly construct quasi-Hankel matrices from data in practical applications, as opposed to relying on existing quasi-Hankel matrix-completion schemes, e.g., [29, 39, 40, 41, 42, 43].

APPENDIX A

Preparation of Reference Images

Here, we work with the D-dimensional arrays counterparts $(\boldsymbol{f}_{\mathcal{A}})_{\mathcal{E}}$ of D-dimensional, grid-sampled signals $\boldsymbol{f}_{\mathcal{A}}$. $\boldsymbol{f}_{\mathcal{A}}$ is a Kodak scene grayscale image that is treated as a finite-dimensional vector. Let the number of samples for the dth dimension be given by the function $\text{size}(\cdot)_d$. Algorithm A.1 is used to crop a D-dimensional grid-sampled signal $(\boldsymbol{f}_{\mathcal{A}})_{\mathcal{E}} \in \mathbb{R}_s^{N_1 \times \cdots \times N_D}$ to generate a reference signal $(\boldsymbol{f}_{\text{ref}})_{\mathcal{E}}$ that is sized such that the number $\text{size}(\boldsymbol{f}_{\text{ref}})_d - 1$ is divisible by the given integer downsample factor $c > 1$, $\forall d \in [D]$. The observed signal in D-dimensional array form, $(\boldsymbol{f}_{\mathcal{X}})_{\mathcal{E}}$, is generated by keeping the first and last samples along each dimension of $\boldsymbol{f}_{\text{ref}}$, and keeping every c samples along each dimension. $\mathcal{X}$ is defined as the set of inputs that corresponds to our construction of $\boldsymbol{f}_{\text{ob},\mathcal{X}}$, and $\mathcal{X}_q$ is defined as the set of inputs that corresponds to our construction of $\boldsymbol{f}_{\text{ref}}$.

Algorithm A.1 Crop array for downsampling

Require: The D-dimensional array $(\boldsymbol{f}_{\mathcal{A}})_{\mathcal{E}}$, and $c > 1$.

for $d \in [D]$ **do**
  if $\text{rem}(s_d, c)$ is non-zero **then**
    $m_d := c \cdot \text{floor}\left(\frac{s_d}{c}\right) + 1$
  else
    $m_d := s_d - (c - 1)$
  end if
end for
Return $\left(\boldsymbol{f}_{\text{ob},\mathcal{X}}\right)_{\mathcal{E}} := (\boldsymbol{f}_{\mathcal{A}})_{\mathcal{E}}[1 : c : m_1, 1 : c : m_2, \cdots, 1 : c : m_D]$, and $(\boldsymbol{f}_{\text{ref}})_{\mathcal{E}} := (\boldsymbol{f}_{\mathcal{A}})_{\mathcal{E}}[1 : m_1, 1 : m_2, \cdots, 1 : m_D]$

Example A.1. Consider the case when $\boldsymbol{f}_{\mathcal{A}}$ is a grayscale grid-sampled image with 6×8 pixels, and $c = 3$ is specified. D is 2, and $\text{size}(\boldsymbol{f}_{\mathcal{A}}) = (6, 8)$. The gray dots in Figure A.1 are the sampling locations, the 6 green dots are the sampling locations for the 2×3 pixels that make up the downsampled image $\boldsymbol{f}_{\text{ob},\mathcal{X}}$.

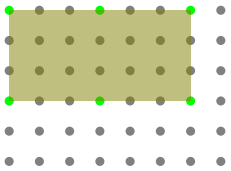


Figure A.1: Reference image cropping scheme. The gray dots represent $\boldsymbol{f}_a$, the shaded area represents the image canvas of $\boldsymbol{f}_{\text{ref}}$, and the green dots are the sampling locations for the downsampled image $\boldsymbol{f}$.

APPENDIX B

The Multi-Index Set $\mathbb{A}_{\text{de}}(D,\, L) \cong \mathbb{A}(D,\, L)$

For a dehomogenized polynomial, one of the variables is fixed to 1. In this thesis, this is taken to be the first variable when we discuss dehomogenized polynomials. This means any monomial ordering over a D-tuple of variables for a dehomogenized polynomials can safely ignore the first slot. In this appendix, we discuss the following index-set that uses only the last $D-1$ variables that serves as the monomial exponents for such a dehomogenized polynomial:

Definition B.1. The $\mathbb{A}_{\text{de}}(D,\, L)$ spaces of multi-indices. Given finite positive integers D and L, we define the following subset of $(D-1)$-dimensional multi-indices in $[0 : L]^{D-1}$.

$$\mathbb{A}_{\text{de}}(D,\, L) := \bigcup_{l=0}^{L} \mathbb{A}(D-1,\, l). \tag{B.1}$$

It follows that

$$|\mathbb{A}_{\text{de}}(D,\, L)| = \sum_{l=0}^{L} \binom{l + (D-1) - 1}{l} = \binom{L + D - 1}{L}, \tag{B.2}$$

B.1 Isomorphism to $\mathbb{A}(D,\, L)$

We can bijectively map a multi-index from $\mathbb{A}(D,\, L)$ to $\mathbb{A}_{\text{de}}(D,\, L)$ by the following map:

$$\begin{aligned} \mathfrak{a}_{D,\,L} : \mathbb{A}_{\text{de}}(D,\, L) &\to \mathbb{A}(D,\, L) \\ \alpha &\mapsto (L - |\alpha|,\, \alpha_1,\, \alpha_2, \cdots,\, \alpha_{D-1}), \end{aligned} \tag{B.3}$$

where α_i is the ith entry of α, $\forall i \in [D-1]$. Its inverse is

$$\begin{aligned} \mathfrak{a}^{-1}_{D,\,L} : \mathbb{A}(D,\, L) &\to \mathbb{A}_{\text{de}}(D,\, L) \\ \beta &\mapsto (\beta_2,\, \beta_3, \cdots,\, \beta_D), \end{aligned} \tag{B.4}$$

where β_i is the ith entry of β, $\forall i \in [D]$.

B.2 An Interpretation for $\mathbb{A}(D,\, L)_{\succ_{\mathbf{V}}}$

The monomial ordering $\succ_{\mathrm{V}}$ for $\mathbb{A}(D,\, L)$ from Definition 2.123 are related to the maps $\mathfrak{a}_{D,\,L}$ and $\mathfrak{a}^{-1}_{D,\,L}$. Recall from Equation B.1 that

$$\mathbb{A}_{\mathrm{de}}(D,\, L) := \bigcup_{l=0}^{L} \mathbb{A}(D-1,\, l).$$

Since we can map an element of $\mathbb{A}(D,\, L)$ to $\mathbb{A}_{\mathrm{de}}(D,\, L)$ using the map from Equation B.4, an intuitive way to visualize the ordering $\succ_{\mathrm{V}}$ is to follow the ordering of $\mathbb{A}(D-1,\, 0)_{\succ_R}$, then the ordering of $\mathbb{A}(D-1,\, l)_{\succ_R}$ for incremental values of $l \in [0 : L]$, until we reach $\mathbb{A}(D-1,\, L)_{\succ_R}$. Each element returned is a multi-index in $\mathbb{A}(D-1,\, l)$ for some corresponding $l \in [0 : L]$, and we use $\mathfrak{a}_{D,\,L}$ from Equation B.3 to convert them to an element of $\mathbb{A}(D,\, L)$. This was used in the numerical implementation of the monomial ordering $\succ_{\mathrm{V}}$ for $\mathbb{A}(D,\, L)$ in this thesis work.

Example B.2. The elements of the ordered set $\mathbb{A}_{\mathrm{de}}(3,\, 4)_{\succ_{\mathrm{V}}}$. If we apply the algorithm discussed in this appendix, we have

$$\begin{aligned}\mathbb{A}_{\mathrm{de}}(3,\, 4)_{\succ_{\mathrm{V}}} :=\{&(0,\, 0),\, (1,\, 0),\, (0,\, 1),\, (2,\, 0),\, (1,\, 1),\, (0,\, 2),\, (3,\, 0),\\ &(2,\, 1),\, (1,\, 2),\, (0,\, 3),\, (4,\, 0),\, (3,\, 1),\, (2,\, 2),\, (1,\, 3),\, (0,\, 4)\}.\end{aligned}$$

This is isomorphic to

$$\begin{aligned}\mathbb{A}(3,\, 4)_{\succ_{\mathrm{V}}} :=\{&(4,\, 0,\, 0),\, (3,\, 1,\, 0),\, (3,\, 0,\, 1),\, (2,\, 2,\, 0),\, (2,\, 1,\, 1),\, (2,\, 0,\, 2),\, (1,\, 3,\, 0),\\ &(1,\, 2,\, 1),\, (1,\, 1,\, 2),\, (1,\, 0,\, 3),\, (0,\, 4,\, 0),\, (0,\, 3,\, 1),\, (0,\, 2,\, 2),\, (0,\, 1,\, 3),\, (0,\, 0,\, 4)\},\end{aligned}$$

which is exactly what we have in Example 2.124.

APPENDIX C

Synthetic Benchmark Setup

In this appendix, we provide the numerical values of the set of ground truth rank-one decompositions and initial solutions that were used to generate the results in Section 4.7.2. Each instantiated problem is generated by a ground truth. Each test run is an instantiated problem paired with an initial solution. For the results reported in Section 4.7.2, there are:

1. Six configurations: D3-p2-L4, D3-p4-L6, D3-p7-L4, D4-p3-L4, D4-p4-L4, and D4-p4-L6.

2. Seven instantiated problems for each configuration.

3. Five initial solutions for each configuration.

C.1 Ground Truth Rank-One Decompositions

The ground truth rank-one decomposition for each instantiated problem for a given configuration is represented using the homogeneous polynomial representation of a symmetric tensor that was reviewed in Equation 2.88:

$$g(x) = \sum_{j=1}^{p} \lambda_j \left(\sum_{d=1}^{D} u_d^{(j)} x_d \right)^L,$$

where x is a D-tuple of variables. A rank-one decomposition is represented by the tuple $(\lambda,\, u)$. Each entry of u is generated by an independent draw from a standard normal distribution. Each entry of λ is generated by an independent draw from an uniform distribution over the interval between 0 and 1. All values are reported up to four significant figures.

For the D3-p2-L4 configuration, problem 1:

$$u = \left\{ \begin{bmatrix} 0.4237 \\ 0.8536 \\ 2.157 \end{bmatrix}, \begin{bmatrix} -1.432 \\ 1.868 \\ 1.141 \end{bmatrix} \right\}$$
$$\lambda = \{0.3554,\, 0.1841\}.$$

For the D3-p2-L4 configuration, problem 2:

$$u = \left\{ \begin{bmatrix} -0.9171 \\ 0.9454 \\ 0.09979 \end{bmatrix}, \begin{bmatrix} 0.1647 \\ -0.2150 \\ -1.046 \end{bmatrix} \right\}$$
$$\lambda = \{0.7165,\ 0.1284\}.$$

For the D3-p2-L4 configuration, problem 3:

$$u = \left\{ \begin{bmatrix} 0.9531 \\ 1.070 \\ -0.5916 \end{bmatrix}, \begin{bmatrix} -1.859 \\ 0.4636 \\ -0.1673 \end{bmatrix} \right\}$$
$$\lambda = \{0.9854,\ 0.1067\}.$$

For the D3-p2-L4 configuration, problem 4:

$$u = \left\{ \begin{bmatrix} -0.02076 \\ -2.041 \\ -0.02245 \end{bmatrix}, \begin{bmatrix} 0.8328 \\ 0.5530 \\ -0.2448 \end{bmatrix} \right\}$$
$$\lambda = \{0.6479,\ 2.710\text{E-}03\}.$$

For the D3-p2-L4 configuration, problem 5:

$$u = \left\{ \begin{bmatrix} 1.014 \\ -1.077 \\ -0.9779 \end{bmatrix}, \begin{bmatrix} 1.007 \\ 0.1530 \\ 0.9792 \end{bmatrix} \right\}$$
$$\lambda = \{0.5895,\ 0.9808\}.$$

For the D3-p2-L4 configuration, problem 6:

$$u = \left\{ \begin{bmatrix} 0.06516 \\ -1.726 \\ 0.6256 \end{bmatrix}, \begin{bmatrix} -1.729 \\ 0.6332 \\ 0.1183 \end{bmatrix} \right\}$$
$$\lambda = \{0.9999,\ 0.7757\}.$$

For the D3-p2-L4 configuration, problem 7:

$$u = \left\{ \begin{bmatrix} -0.4245 \\ 1.750 \\ -2.438 \end{bmatrix}, \begin{bmatrix} 0.9712 \\ -0.6773 \\ 0.3760 \end{bmatrix} \right\}$$
$$\lambda = \{0.4884,\ 0.1949\}.$$

For the D3-p4-L6 configuration, problem 1:

$$u = \left\{ \begin{bmatrix} 0.6500 \\ 4.794\text{E-}03 \\ 1.014 \end{bmatrix}, \begin{bmatrix} -1.077 \\ -0.9779 \\ 1.007 \end{bmatrix}, \begin{bmatrix} 0.1530 \\ 0.9792 \\ 0.6933 \end{bmatrix}, \begin{bmatrix} -0.7347 \\ 0.06516 \\ -1.726 \end{bmatrix} \right\}$$
$$\lambda = \{0.3477,\ 0.8417,\ 0.4844,\ 0.06080\}.$$

For the D3-p4-L6 configuration, problem 2:

$$u = \left\{ \begin{bmatrix} -0.4245 \\ 1.750 \\ -2.348 \end{bmatrix}, \begin{bmatrix} 0.9712 \\ -0.6773 \\ 0.3760 \end{bmatrix}, \begin{bmatrix} 1.053 \\ -0.2495 \\ 0.4083 \end{bmatrix}, \begin{bmatrix} -0.4584 \\ 0.6186 \\ 2.112 \end{bmatrix} \right\}$$
$$\lambda = \{0.9333,\ 0.1257,\ 0.8004,\ 0.9423\}.$$

For the D3-p4-L6 configuration, problem 3:

$$u = \left\{ \begin{bmatrix} -2.146 \\ 0.8898 \\ 0.05702 \end{bmatrix}, \begin{bmatrix} -0.7324 \\ 0.03222 \\ 0.4204 \end{bmatrix}, \begin{bmatrix} 0.92334 \\ 1.369 \\ -0.8699 \end{bmatrix}, \begin{bmatrix} 0.9475 \\ -0.01474 \\ -0.7970 \end{bmatrix} \right\}$$
$$\lambda = \{0.09946,\ 0.5046,\ 0.6718,\ 0.6489\}.$$

For the D3-p4-L6 configuration, problem 4:

$$u = \left\{ \begin{bmatrix} 0.03000 \\ -0.3769 \\ 1.472 \end{bmatrix}, \begin{bmatrix} 1.288 \\ 0.2806 \\ -0.4720 \end{bmatrix}, \begin{bmatrix} 1.413 \\ 0.8313 \\ 1.182 \end{bmatrix}, \begin{bmatrix} -0.06724 \\ -0.9143 \\ -0.5966 \end{bmatrix} \right\}$$
$$\lambda = \{0.8208,\ 0.2923,\ 0.6032,\ 0.3848\}.$$

For the D3-p4-L6 configuration, problem 5:

$$u = \left\{ \begin{bmatrix} -1.376 \\ -0.7087 \\ 0.1848 \end{bmatrix}, \begin{bmatrix} 0.5620 \\ 1.9516 \\ 0.2276 \end{bmatrix}, \begin{bmatrix} 0.3476 \\ -0.6687 \\ -6.535\text{E-}03 \end{bmatrix}, \begin{bmatrix} -0.6920 \\ -0.3460 \\ 0.6595 \end{bmatrix} \right\}$$
$$\lambda = \{0.9549,\ 0.3179,\ 0.2397,\ 0.4249\}.$$

For the D3-p4-L6 configuration, problem 6:

$$u = \left\{ \begin{bmatrix} 1.923 \\ 0.8408 \\ -0.9131 \end{bmatrix}, \begin{bmatrix} -0.2391 \\ 0.09132 \\ -0.3066 \end{bmatrix}, \begin{bmatrix} -0.1706 \\ 0.1563 \\ 0.4877 \end{bmatrix}, \begin{bmatrix} 0.5712 \\ 2.176 \\ 0.3352 \end{bmatrix} \right\}$$
$$\lambda = \{0.2092,\ 0.6221,\ 0.6889,\ 0.2630\}.$$

For the D3-p4-L6 configuration, problem 7:

$$u = \left\{ \begin{bmatrix} -2.344 \\ 1.171 \\ -0.7633 \end{bmatrix}, \begin{bmatrix} 1.853 \\ 0.8778 \\ 0.01498 \end{bmatrix}, \begin{bmatrix} -0.7477 \\ 0.1279 \\ 0.4847 \end{bmatrix}, \begin{bmatrix} 0.1484 \\ 0.5823 \\ 0.9334 \end{bmatrix} \right\}$$
$$\lambda = \{0.02916,\ 0.5157,\ 0.01386,\ 0.6300\}.$$

For the D3-p7-L4 configuration, problem 1:

$$u = \left\{ \begin{bmatrix} 0.6425 \\ -1.027 \\ -1.257 \end{bmatrix}, \begin{bmatrix} 0.03000 \\ -0.3769 \\ 1.472 \end{bmatrix}, \begin{bmatrix} 1.288 \\ 0.2806 \\ -0.4720 \end{bmatrix}, \begin{bmatrix} 1.413 \\ 0.8313 \\ 1.182 \end{bmatrix}, \begin{bmatrix} -0.06724 \\ -0.9143 \\ -0.5966 \end{bmatrix}, \begin{bmatrix} -1.299 \\ 0.6208 \\ 0.6100 \end{bmatrix}, \begin{bmatrix} -0.3637 \\ -1.376 \\ -0.7087 \end{bmatrix} \right\}$$
$$\lambda = \{0.1392,\ 0.4808,\ 0.9634,\ 0.2216,\ 0.1932,\ 0.3086,\ 0.9009\}.$$

For the D3-p7-L4 configuration, problem 2:

$$u = \left\{ \begin{bmatrix} -0.9983 \\ -6.535\text{E-}03 \\ -0.6920 \end{bmatrix}, \begin{bmatrix} -0.3460 \\ 0.6600 \\ -1.395 \end{bmatrix}, \begin{bmatrix} -0.4155 \\ -0.2784 \\ -0.5322 \end{bmatrix}, \begin{bmatrix} 1.923 \\ 0.8408 \\ -0.9131 \end{bmatrix}, \begin{bmatrix} -0.2391 \\ 0.09132 \\ -0.3066 \end{bmatrix}, \begin{bmatrix} -0.1706 \\ 0.1563 \\ 0.4877 \end{bmatrix}, \begin{bmatrix} 0.5712 \\ 2.176 \\ 0.3352 \end{bmatrix} \right\}$$
$$\lambda = \{0.2092,\ 0.6221,\ 0.6889,\ 0.2630,\ 0.8203,\ 0.9895,\ 0.5152\}.$$

For the D3-p7-L4 configuration, problem 3:

$$u = \left\{ \begin{bmatrix} 1.853 \\ 0.8778 \\ 0.01498 \end{bmatrix}, \begin{bmatrix} -0.7477 \\ 0.1279 \\ 0.4847 \end{bmatrix}, \begin{bmatrix} 0.1484 \\ 0.5823 \\ 0.9334 \end{bmatrix}, \begin{bmatrix} -0.02926 \\ 1.336 \\ 09.883\text{E-}03 \end{bmatrix}, \begin{bmatrix} 1.631 \\ 1.257 \\ -1.577 \end{bmatrix}, \begin{bmatrix} 0.8125 \\ 0.1540 \\ 0.3028 \end{bmatrix}, \begin{bmatrix} 1.397 \\ -1.032 \\ -1.046 \end{bmatrix} \right\}$$
$$\lambda = \{0.7583,\ 0.9971,\ 0.9614,\ 0.9034,\ 0.6022,\ 0.9601,\ 0.4270\}.$$

For the D3-p7-L4 configuration, problem 4:

$$u = \left\{ \begin{bmatrix} 1.161 \\ -0.04652 \\ 1.539 \end{bmatrix}, \begin{bmatrix} 1.451 \\ -0.9585 \\ -0.6686 \end{bmatrix}, \begin{bmatrix} -0.1202 \\ 1.521 \\ 1.941 \end{bmatrix}, \begin{bmatrix} -0.7928 \\ 0.4845 \\ -0.6949 \end{bmatrix}, \begin{bmatrix} 0.4142 \\ -0.9770 \\ -0.2651 \end{bmatrix}, \begin{bmatrix} -1.593 \\ 1.261 \\ -1.194 \end{bmatrix}, \begin{bmatrix} 0.1820 \\ -0.9176 \\ 0.0259 \end{bmatrix} \right\}$$

$\lambda = \{0.07628,\ 0.2894,\ 0.9854,\ 0.1997,\ 0.4703,\ 0.6370,\ 0.7872\}$.

For the D3-p7-L4 configuration, problem 5:

$$u = \left\{ \begin{bmatrix} 0.2933 \\ 0.03563 \\ 0.3851 \end{bmatrix}, \begin{bmatrix} -0.7877 \\ -0.1459 \\ -0.7736 \end{bmatrix}, \begin{bmatrix} 0.4419 \\ 0.3815 \\ -0.2652 \end{bmatrix}, \begin{bmatrix} 0.04312 \\ 0.9431 \\ -0.2150 \end{bmatrix}, \begin{bmatrix} 0.01637 \\ 0.3724 \\ -0.3557 \end{bmatrix}, \begin{bmatrix} 0.4023 \\ 1.683 \\ 0.2794 \end{bmatrix}, \begin{bmatrix} -1.975 \\ -0.2434 \\ -1.522 \end{bmatrix} \right\}$$

$\lambda = \{0.7273,\ 0.5302,\ 0.5370,\ 0.8856,\ 0.7608,\ 0.1218,\ 0.7669\}$.

For the D3-p7-L4 configuration, problem 6:

$$u = \left\{ \begin{bmatrix} 1.5629 \\ 1.354 \\ -0.3012 \end{bmatrix}, \begin{bmatrix} 2.313 \\ -0.8432 \\ 0.1662 \end{bmatrix}, \begin{bmatrix} -1.958 \\ 0.8908 \\ 0.2657 \end{bmatrix}, \begin{bmatrix} 0.6159 \\ -1.079 \\ 0.6849 \end{bmatrix}, \begin{bmatrix} -1.466 \\ 0.6744 \\ -0.05636 \end{bmatrix}, \begin{bmatrix} -0.6728 \\ -2.118 \\ -0.06178 \end{bmatrix}, \begin{bmatrix} -0.1721 \\ -0.3216 \\ -0.7321 \end{bmatrix} \right\}$$

$\lambda = \{0.1678,\ 0.9400,\ 0.3354,\ 0.9711,\ 0.4438,\ 0.4714,\ 0.6187\}$.

For the D3-p7-L4 configuration, problem 7:

$$u = \left\{ \begin{bmatrix} -0.6733 \\ -0.3630 \\ -1.185 \end{bmatrix}, \begin{bmatrix} 0.4452 \\ 0.7549 \\ 0.06958 \end{bmatrix}, \begin{bmatrix} -0.1814 \\ 0.3906 \\ -0.5499 \end{bmatrix}, \begin{bmatrix} -1.049 \\ 1.963 \\ -0.08115 \end{bmatrix}, \begin{bmatrix} 0.7882 \\ -0.7214 \\ 1.928 \end{bmatrix}, \begin{bmatrix} -0.01583 \\ 0.1057 \\ -0.3365 \end{bmatrix}, \begin{bmatrix} -1.342 \\ -1.159 \\ -1.127 \end{bmatrix} \right\}$$

$\lambda = \{0.7907,\ 0.3965,\ 0.5452,\ 0.08832,\ 0.1685,\ 0.9194,\ 0.2741\}$.

For the D4-p3-L4 configuration, problem 1:

$$u = \left\{ \begin{bmatrix} 0.6500 \\ 4.794\text{E-}03 \\ 1.014 \\ -1.077 \end{bmatrix}, \begin{bmatrix} -0.9779 \\ 1.007 \\ 0.1530 \\ 0.9792 \end{bmatrix}, \begin{bmatrix} 0.6933 \\ -0.7347 \\ 0.06516 \\ -1.726 \end{bmatrix} \right\}$$

$\lambda = \{0.347714,\ 0.841742,\ 0.484449\}$.

For the D4-p3-L4 configuration, problem 2:

$$u = \left\{ \begin{bmatrix} 0.1183 \\ -0.4245 \\ 1.750 \\ -2.438 \end{bmatrix}, \begin{bmatrix} 0.9712 \\ -0.6773 \\ 0.3760 \\ 1.053 \end{bmatrix}, \begin{bmatrix} -0.2495 \\ 0.4083 \\ -0.4584 \\ 0.6186 \end{bmatrix} \right\}$$

$\lambda = \{0.8088,\ 0.9333,\ 0.1257\}$.

For the D4-p3-L4 configuration, problem 3:

$$u = \left\{ \begin{bmatrix} 0.7139 \\ -1.415 \\ -2.146 \\ 0.8898 \end{bmatrix}, \begin{bmatrix} 0.05702 \\ -0.7324 \\ 0.03222 \\ 0.4204 \end{bmatrix}, \begin{bmatrix} 0.9234 \\ 1.369 \\ -0.8699 \\ 0.9475 \end{bmatrix} \right\}$$

$\lambda = \{0.01089,\ 0.4061,\ 0.09946\}$.

For the D4-p3-L4 configuration, problem 4:

$$u = \left\{ \begin{bmatrix} 0.6425 \\ -1.027 \\ -1.257 \\ 0.03000 \end{bmatrix}, \begin{bmatrix} -0.3769 \\ 1.472 \\ 1.288 \\ 0.2806 \end{bmatrix}, \begin{bmatrix} -0.4720 \\ 1.413 \\ 0.8313 \\ 1.182 \end{bmatrix} \right\}$$

$\lambda = \{0.01949,\ 0.8703,\ 0.5557\}$.

For the D4-p3-L4 configuration, problem 5:

$$u = \left\{ \begin{bmatrix} -1.299 \\ 0.6208 \\ 0.6100 \\ -0.3637 \end{bmatrix}, \begin{bmatrix} -1.376 \\ -0.7087 \\ 0.1848 \\ 0.5620 \end{bmatrix}, \begin{bmatrix} 1.952 \\ 0.2276 \\ 0.3476 \\ -0.6687 \end{bmatrix} \right\}$$
$$\lambda = \{0.9009,\ 0.7883,\ 0.005660\}.$$

For the D4-p3-L4 configuration, problem 6:

$$u = \left\{ \begin{bmatrix} -0.6920 \\ -0.3460 \\ 0.6595 \\ -1.395 \end{bmatrix}, \begin{bmatrix} -0.4155 \\ -0.2784 \\ -0.5322 \\ 1.923 \end{bmatrix}, \begin{bmatrix} 0.8408 \\ -0.9131 \\ -0.2391 \\ 0.09132 \end{bmatrix} \right\}$$
$$\lambda = \{0.1029,\ 0.1578,\ 0.1919\}.$$

For the D4-p3-L4 configuration, problem 7:

$$u = \left\{ \begin{bmatrix} 0.4877 \\ 0.5712 \\ 2.176 \\ 0.3352 \end{bmatrix}, \begin{bmatrix} 0.443 \\ -0.8966 \\ 1.389 \\ 0.5232 \end{bmatrix}, \begin{bmatrix} -2.344 \\ 1.171 \\ -0.7633 \\ 1.853 \end{bmatrix} \right\}$$
$$\lambda = \{0.9374,\ 8.800\text{E-}03,\ 0.5662\}.$$

For the D4-p4-L4 and D4-p4-L6 configurations, problem 1:

$$u = \left\{ \begin{bmatrix} -0.6773 \\ 0.3760 \\ 1.053 \\ -0.2495 \end{bmatrix}, \begin{bmatrix} 0.4083 \\ -0.4584 \\ 0.6186 \\ 2.112 \end{bmatrix}, \begin{bmatrix} -1.891 \\ 0.05501 \\ 0.7139 \\ -1.415 \end{bmatrix}, \begin{bmatrix} -2.146 \\ 0.8898 \\ 0.05702 \\ -0.7324 \end{bmatrix} \right\}$$
$$\lambda = \{0.04441,\ 0.2890,\ 0.9683,\ 0.7740\}.$$

For the D4-p4-L4 and D4-p4-L6 configurations, problem 2:

$$u = \left\{ \begin{bmatrix} -0.8699 \\ 0.9475 \\ -0.01474 \\ -0.7970 \end{bmatrix}, \begin{bmatrix} -0.09568 \\ 0.6425 \\ -1.027 \\ -1.257 \end{bmatrix}, \begin{bmatrix} 0.03000 \\ -0.3770 \\ 1.472 \\ 1.288 \end{bmatrix}, \begin{bmatrix} 0.2806 \\ -0.4720 \\ 1.413 \\ 0.8313 \end{bmatrix} \right\}$$
$$\lambda = \{0.8126,\ 0.01949,\ 0.8703,\ 0.5557\}.$$

For the D4-p4-L4 and D4-p4-L6 configurations, problem 3:

$$u = \left\{ \begin{bmatrix} -1.299 \\ 0.6208 \\ 0.6100 \\ -0.3637 \end{bmatrix}, \begin{bmatrix} -1.376 \\ -0.7087 \\ 0.1848 \\ 0.5620 \end{bmatrix}, \begin{bmatrix} 1.952 \\ 0.2276 \\ 0.3476 \\ -0.6687 \end{bmatrix}, \begin{bmatrix} -6.535\text{E-}03 \\ -0.6920 \\ -0.3460 \\ 006595 \end{bmatrix} \right\}$$
$$\lambda = \{0.9549,\ 0.3179,\ 0.2397,\ 0.4249\}.$$

For the D4-p4-L4 and D4-p4-L6 configurations, problem 4:

$$u = \left\{ \begin{bmatrix} 1.923 \\ 0.8408 \\ -0.9131 \\ -0.2391 \end{bmatrix}, \begin{bmatrix} 0.09132 \\ -0.3066 \\ -0.1706 \\ 0.1563 \end{bmatrix}, \begin{bmatrix} 0.4877 \\ 0.5712 \\ 2.176 \\ 0.3352 \end{bmatrix}, \begin{bmatrix} 0.4443 \\ -0.8966 \\ 1.389 \\ 0.5232 \end{bmatrix} \right\}$$
$$\lambda = \{0.8203,\ 0.9895,\ 0.5152,\ 0.6402\}.$$

For the D4-p4-L4 and D4-p4-L6 configurations, problem 5:

$$u = \left\{ \begin{bmatrix} 0.8778 \\ 0.01498 \\ -0.7477 \\ 0.1279 \end{bmatrix}, \begin{bmatrix} 0.4847 \\ 0.1484 \\ 0.5823 \\ 0.9334 \end{bmatrix}, \begin{bmatrix} -0.02926 \\ 1.336 \\ 9.883\text{E-}03 \\ 1.631 \end{bmatrix}, \begin{bmatrix} 1.257 \\ -1.578 \\ 0.8125 \\ 0.1540 \end{bmatrix} \right\}$$
$$\lambda = \{0.2782,\ 0.4051,\ 0.9686,\ 0.4605\}.$$

For the D4-p4-L4 and D4-p4-L6 configurations, problem 6:

$$u = \left\{ \begin{bmatrix} 1.462 \\ 1.363 \\ -1.156 \\ 1.788 \end{bmatrix}, \begin{bmatrix} 0.5319 \\ 1.161 \\ -0.0465 \\ 1.539 \end{bmatrix}, \begin{bmatrix} 1.451 \\ -0.9585 \\ -0.6686 \\ -0.1202 \end{bmatrix}, \begin{bmatrix} 1.521 \\ 1.945 \\ -0.7928 \\ 0.4845 \end{bmatrix} \right\}$$
$$\lambda = \{0.5429,\ 0.2254,\ 0.6172,\ 0.1726\}.$$

For the D4-p4-L4 and D4-p4-L6 configurations, problem 7:

$$u = \left\{ \begin{bmatrix} -1.593 \\ 1.261 \\ -1.194 \\ 0.1820 \end{bmatrix}, \begin{bmatrix} -0.9176 \\ 0.02589 \\ 0.1321 \\ -0.4702 \end{bmatrix}, \begin{bmatrix} 1.012 \\ -0.1563 \\ -0.3234 \\ -0.7030 \end{bmatrix}, \begin{bmatrix} -0.4172 \\ 0.2933 \\ 0.03563 \\ 0.3851 \end{bmatrix} \right\}$$
$$\lambda = \{0.5676,\ 0.06165,\ 0.4173,\ 0.5359\}.$$

C.2 Initial Solutions

An initial solution $A_0^{\mathcal{P}} := \phi_{\mathcal{P},\mathcal{N}}(A_0) \in \mathbb{R}_{\text{s}}^{D\times p}$ for optimization problems over $\mathcal{P}$ for a given configuration is the first D rows of the matrix $A_0 = \mathtt{V}_0\bar{\mathtt{C}}_0 \in \mathcal{V}$, where $\mathtt{V}_0$ is a quasi-Vandermonde matrix and $\bar{C}_0$ is a diagonal matrix with strictly positive entries. Here, $\mathcal{V}$ is the manifold proposed in Section 4.5.1, and the map $\phi_{\mathcal{P},\mathcal{N}}$ is explained in greater detail in Section 4.5.2.

The first row of each $A_0^{\mathcal{P}}$ corresponds to the diagonals of $\bar{\mathtt{C}}_0$, and must be strictly positive. The proposed optimizer frequently convert between tangent spaces during its operation, so division by an entry from the first row of $\phi_{\mathcal{P},\mathcal{N}}(A)$ for some $A \in \mathcal{V}$ is frequently encountered. For this reason, the entry of $\phi_{\mathcal{P},\mathcal{N}}(A)$ is bounded to between positive real numbers $\bar{c}_{\text{min}}$ and $\bar{c}_{\text{max}}$ to avoid numerical issues, and we used the following heuristic strategy to clamp a randomly generated $A_0^{\mathcal{P}}$ so that it entries in the first row that are between $\bar{c}_{\text{min}}$ and $\bar{c}_{\text{max}}$. First, each entry in $A_0^{\mathcal{P}}$ is generated by independent draws to the standard normal distribution. Let $d := \bar{c}_{\text{max}} - \bar{c}_{\text{min}}$. Then, we make the replacement

$$\left(A_0^{\mathcal{P}}\right)_{1,\,j} \leftarrow \bar{c}_{\text{min}} + 0.1d$$

if the jth entry in the first row, $(A_0^{\mathcal{P}})_{1,\,j}$, is smaller than $\bar{c}_{\text{min}}$. We make the replacement

$$\left(A_0^{\mathcal{P}}\right)_{1,\,j} \leftarrow \bar{c}_{\text{max}} - 0.1d$$

if the jth entry in the first row, $(A_0^{\mathcal{P}})_{1,\,j}$, is larger than $\bar{c}_{\text{max}}$.

We denote by $A_0^{\mathcal{P},(i)}$ the ith initial solution for a given configuration, and there are five initial solutions used for each configuration for the results reported in Section 4.7.2. The values $\bar{c}_{\text{min}}$ =1E-4 and $\bar{c}_{\text{max}} = 9$ were used because all benchmarks ran in a stable manner with this choice. Smaller values of $\bar{c}_{\text{min}}$ caused some numerical issues for some of the optimization problems in the configurations that have higher tensor

orders (i.e., when L is 6 as opposed to 4). Changing the value of $\bar{c}_{\text{max}}$ did not have a noticeable effect on performance. A value that appears frequently is

$$\begin{aligned}\bar{c}_{\text{min}} + 0.1d &= 0.0001 + 0.1(9 - 0.0001) \\ &= 0.90009 \\ &\approx 0.9001.\end{aligned}$$

This value was used as a replacement to an entry from the first row of an $A_0^{\mathcal{P}}$, if the original randomly generated entry is smaller than the tolerance of 1E-4.

For the D3-p2-L4 configuration:

$$A_0^{\mathcal{P},(1)} = \begin{bmatrix} 0.9001 & 0.9001 \\ -0.5118 & -0.1509 \\ -0.9280 & -2.067 \end{bmatrix}$$

$$A_0^{\mathcal{P},(2)} = \begin{bmatrix} 0.9001 & 1.054 \\ 0.3176 & 0.5118 \\ 0.04079 & 1.626 \end{bmatrix}$$

$$A_0^{\mathcal{P},(3)} = \begin{bmatrix} 1.035 & 0.9001 \\ -0.9246 & -1.775 \\ -1.090 & -1.557 \end{bmatrix}$$

$$A_0^{\mathcal{P},(4)} = \begin{bmatrix} 0.7028 & 1.527 \\ 0.8348 & -0.05500 \\ -1.451 & 1.023 \end{bmatrix}$$

$$A_0^{\mathcal{P},(5)} = \begin{bmatrix} 1.547 & 2.610 \\ -0.7160 & 1.309 \\ -1.600 & -1.050 \end{bmatrix}.$$

For the D3-p4-L6 configuration:

$$A_0^{\mathcal{P},(1)} = \begin{bmatrix} 0.9001 & 0.9001 & 0.9001 & 1.054 \\ -0.5118 & -0.1509 & 0.3176 & 0.5118 \\ -0.9280 & -2.067 & 0.04079 & 1.626 \end{bmatrix}$$

$$A_0^{\mathcal{P},(2)} = \begin{bmatrix} 1.035 & 0.9001 & 0.7028 & 1.527 \\ -0.9246 & -1.775 & 0.8348 & -0.05500 \\ -1.090 & -1.557 & -1.451 & 1.023 \end{bmatrix}$$

$$A_0^{\mathcal{P},(3)} = \begin{bmatrix} 1.547 & 2.610 & 0.4237 & 0.9001 \\ -0.7160 & 1.309 & 0.8536 & 1.868 \\ -1.600 & -1.050 & 2.157 & 1.141 \end{bmatrix}$$

$$A_0^{\mathcal{P},(4)} = \begin{bmatrix} 0.9001 & 0.9454 & 0.9001 & 0.9001 \\ 0.2331 & 0.09979 & -1.046 & 0.9531 \\ -0.9171 & 0.1647 & -1.284 & 1.070 \end{bmatrix}$$

$$A_0^{\mathcal{P},(5)} = \begin{bmatrix} 0.9001 & 0.9001 & 0.9001 & 0.8328 \\ -1.859 & -1.859 & -2.041 & 0.5530 \\ 0.4636 & -0.09514 & -0.02245 & -0.2448 \end{bmatrix}.$$

For the D3-p7-L4 configuration:

$$A_0^{\mathcal{P},(1)} = \begin{bmatrix} 0.9001 & 0.9001 & 0.9001 & 1.054 & 1.035 & 0.9001 & 0.7028 \\ -0.5118 & -0.1509 & 0.3176 & 0.5118 & -0.9246 & -1.775 & 0.8348 \\ -0.9280 & -2.067 & 0.04079 & 1.626 & -1.090 & -1.557 & -1.451 \end{bmatrix}$$

$$A_0^{\mathcal{P},(2)} = \begin{bmatrix} 1.527 & 1.547 & 2.610 & 0.4237 & 0.9001 & 0.9001 & 0.9454 \\ -0.05500 & -0.7160 & 1.309 & 0.8536 & 1.868 & 0.2331 & 0.09979 \\ 1.023 & -1.600 & -1.050 & 2.157 & 1.141 & -0.9171 & 0.1647 \end{bmatrix}$$

$$A_0^{\mathcal{P},(3)} = \begin{bmatrix} 0.9001 & 0.9001 & 0.9001 & 0.9001 & 0.9001 & 0.8328 & 0.6500 \\ -1.046 & 0.9531 & -1.859 & -1.859 & -2.041 & 0.5530 & 0.004794 \\ -1.284 & 1.070 & 0.4636 & -0.09514 & -0.02245 & -0.2448 & 1.014 \end{bmatrix}$$

$$A_0^{\mathcal{P},(4)} = \begin{bmatrix} 0.9001 & 0.1530 & 0.9001 & 0.6256 & 0.1183 & 0.9001 & 0.3760 \\ -0.9779 & 0.9792 & 0.06516 & -1.729 & -0.4245 & 0.9712 & 1.053 \\ 1.007 & 0.6933 & -1.726 & 0.6332 & 1.750 & -0.6773 & -0.2495 \end{bmatrix}$$

$$A_0^{\mathcal{P},(5)} = \begin{bmatrix} 0.4083 & 2.112 & 0.7138 & 0.8898 & 0.03222 & 1.369 & 0.9001 \\ -0.4584 & -1.891 & -1.415 & 0.05702 & 0.4204 & -0.8699 & -0.7970 \\ 0.6186 & 0.05501 & -2.146 & -0.7324 & 0.9233 & 0.9475 & -0.09568 \end{bmatrix}.$$

For the D4-p3-L4 configuration:

$$A_0^{\mathcal{P},(1)} = \begin{bmatrix} 0.9001 & 0.9001 & 0.04079 \\ -0.5118 & -2.067 & 1.054 \\ -0.9280 & -1.659 & 0.5118 \\ -1.198 & 0.3176 & 1.626 \end{bmatrix}$$

$$A_0^{\mathcal{P},(2)} = \begin{bmatrix} 1.035 & 0.9001 & 0.9001 \\ -0.9246 & -1.557 & 1.527 \\ -1.090 & 0.7028 & -0.05500 \\ -0.7709 & 0.8348 & 1.023 \end{bmatrix}$$

$$A_0^{\mathcal{P},(3)} = \begin{bmatrix} 1.547 & 1.309 & 2.157 \\ -0.7160 & -1.050 & -1.432 \\ -1.600 & 0.4237 & 1.868 \\ 2.610 & 0.8536 & 1.141 \end{bmatrix}$$

$$A_0^{\mathcal{P},(4)} = \begin{bmatrix} 0.9001 & 0.09979 & 0.9001 \\ 0.2331 & 0.1647 & -0.2590 \\ -0.9171 & -0.2150 & 0.9531 \\ 0.9454 & -1.046 & 1.070 \end{bmatrix}$$

$$A_0^{\mathcal{P},(5)} = \begin{bmatrix} 0.9001 & 0.9001 & 0.9001 \\ -1.859 & -0.09514 & 0.8328 \\ 0.4636 & -0.02076 & 0.5530 \\ -0.1673 & -2.041 & -0.245 \end{bmatrix}.$$

For the D4-p4-L4 and D4-p4-L6 configurations:

$$A_0^{\mathcal{P},(1)} = \begin{bmatrix} 0.9001 & 0.9001 & 0.04079 & 1.035 \\ -0.5118 & -2.067 & 1.054 & -0.9246 \\ -0.9280 & -1.659 & 0.5118 & -1.090 \\ -1.198 & 0.3176 & 1.626 & -0.7709 \end{bmatrix}$$

$$A_0^{\mathcal{P},(2)} = \begin{bmatrix} 0.9001 & 0.9001 & 1.547 & 1.309 \\ -1.557 & 1.527 & -0.7160 & -1.050 \\ 0.7028 & -0.05500 & -1.600 & 0.4237 \\ 0.8348 & 1.023 & 2.610 & 0.8536 \end{bmatrix}$$

$$A_0^{\mathcal{P},(3)} = \begin{bmatrix} 2.157 & 0.9001 & 0.09979 & 0.9001 \\ -1.432 & 0.2331 & 0.1647 & -0.2590 \\ 1.868 & -0.9171 & -0.2150 & 0.9531 \\ 1.141 & 0.9454 & -1.046 & 1.070 \end{bmatrix}$$

$$A_0^{\mathcal{P},(4)} = \begin{bmatrix} 0.9001 & 0.9001 & 0.9001 & 0.6500 \\ -1.859 & -0.09514 & 0.8328 & 0.004794 \\ 0.4636 & -0.02076 & 0.5530 & 1.014 \\ -0.1673 & -2.041 & -0.2448 & -1.077 \end{bmatrix}$$

$$A_0^{\mathcal{P},(5)} = \begin{bmatrix} 0.9001 & 0.6933 & 0.6256 & 0.9001 \\ 1.007 & -0.7347 & -1.729 & 1.750 \\ 0.1530 & 0.06516 & 0.6332 & -2.438 \\ 0.9792 & -1.726 & 0.1183 & 0.9712 \end{bmatrix}.$$

Bibliography

1. P.-A. Absil, R. Mahony, and R. Sepulchre, *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, 2007. xii, 4, 5, 7, 37, 49, 50, 51, 53, 57, 58, 60, 61, 62, 63, 121, 131, 136, 137, 138, 140, 161, 162

2. C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. MIT Press, 2006. 1, 17, 18, 19, 79, 83

3. A. Shah, A. G. Wilson, and Z. Ghahramani, "Student-t processes as alternatives to gaussian processes," in *Proceedings of the 17th International Conference on Artificial Intelligence and Statistics*, 2014. 2, 17

4. A. Wilson and Z. Ghahramani, "Copula processes," in *Advances in Neural Information Processing Systems 23* (J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, eds.), pp. 2460–2468, Curran Associates, Inc., 2010. 2, 17

5. K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012. 2, 79

6. H. Q. Minh, L. Bazzani, and V. Murino, "A unifying framework for vector-valued manifold regularization and multi-view learning," in *Proceedings of the 30th International Conference on Machine Learning*, 2013. 2, 83, 162

7. C. A. Micchelli and M. A. Pontil, "On learning vector-valued functions," *Neural Computation*, vol. 17, pp. 177–204, Jan. 2005. 2, 16, 17, 83, 162

8. D. Higdon, "Space and space-time modeling using process convolutions," in *Quantitative Methods for Current Environmental Issues* (C. W. Anderson, V. Barnett, P. C. Chatwin, and A. H. El-Shaarawi, eds.), Springer, 2002. 2, 78

9. C. Paciorek, *Nonstationary Gaussian processes for regression and spatial modelling*. PhD thesis, Carnegie Mellon University, 2003. 2, 78

10. P. D. Sampson, D. Damian, and P. Guttorp, *Advances in Modeling and Inference for Environmental Processes with Nonstationary Spatial Covariance*, pp. 17–32. Dordrecht: Springer Netherlands, 2001. 2, 77, 78, 83, 161

11. D. Damian, P. D. Sampson, and P. Guttorp, "Variance modeling for nonstationary spatial processes with temporal replications," *Journal of Geophysical Research: Atmospheres*, vol. 108, no. D24, 2003. 8778. 2, 77, 78, 83, 161

12. A. G. Wilson, D. A. Knowles, and Z. Ghahramani, "Gaussian process regression networks," in *Proceedings of the 29th International Conference on Machine Learning*, 2012. 2, 78

13. A. C. Damianou and N. D. Lawrence, "Deep gaussian processes," in *Proceedings of the 16th International Conference on Artificial Intelligence and Statistics*, 2013. 2

14. T. Schultz, J. Weickert, and H.-P. Seidel, *Visualization and Processing of Tensor Fields*, ch. A Higher-Order Structure Tensor, pp. 263–279. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. 3, 161

15. T. Schultz, A. Fuster, A. Ghosh, R. Deriche, L. Florack, and L.-H. Lim, *Higher-Order Tensors in Diffusion Imaging*, ch. Visualization and Processing of Tensors and Higher Order Descriptors for Multi-Valued Data, pp. 129–161. Mathematics and Visualization, Springer, 2014. 3, 120, 161

16. N. Goyal, S. Vempala, and Y. Xiao, "Fourier pca and robust tensor decomposition," in *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pp. 584–593, 2014. 3, 120

17. J. Jost, *Riemannian Geometry and Geometric Analysis*. Springer-Verlag Berlin Heidelberg, 6 ed., 2011. 3, 41

18. M. Alex and O. Vasilescu, "Multilinear projection for face recognition via canonical decomposition," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 184–191, June 2010. 3, 118

19. M. A. O. Vasilescu and D. Terzopoulos, "TensorTextures: Multilinear image-based rendering," in *ACM SIGGRAPH 2004 Papers*, pp. 336–342, 2004. 3

20. S. Aja-Fernández, R. de Luis García, D. Tao, and X. Li, eds., *Tensors in Image Processing and Computer Vision*. Springer, 2009. 3

21. A. Shashua and T. Hazan, "Non-negative tensor factorization with applications to statistics and computer vision," in *Proceedings of the 22nd International Conference on Machine Learning*, ICML 2005, (New York, NY, USA), pp. 792–799, ACM, 2005. 3, 118

22. A. Cichocki, N. Lee, I. Oseledets, A. H. Phan, Q. Zhao, and D. P. Mandic, "Tensor networks for dimensionality reduction and large-scale optimization: Part 1 low-rank tensor decompositions," *Foundations and Trends in Machine Learning*, vol. 9, no. 4-5, pp. 249–429, 2016. 3

23. A. Cichocki, A. H. Phan, Q. Zhao, N. Lee, I. Oseledets, M. Sugiyama, and D. P. Mandic, "Tensor networks for dimensionality reduction and large-scale optimization: Part 2 applications and future perspectives," *Foundations and Trends in Machine Learning*, vol. 9, no. 6, pp. 431–673, 2017. 3

24. A. Cichocki, D. Mandic, L. D. Lathauwer, G. Zhou, Q. Zhao, C. Caiafa, and H. A. Phan, "Tensor decompositions for signal processing applications: From two-way to multiway component analysis," *IEEE Signal Processing Magazine*, vol. 32, no. 2, pp. 145–163, 2015. 3, 118

25. Q. Zhao, G. Zhou, T. Adali, L. Zhang, and A. Cichocki, "Kernelization of tensor-based models for multiway data analysis: Processing of multidimensional structured data," *IEEE Signal Processing Magazine*, vol. 30, pp. 137–148, July 2013. 3

26. N. D. Sidiropoulos, L. D. Lathauwer, X. Fu, K. Huang, E. E. Papalexakis, and C. Faloutsos, "Tensor decomposition for signal processing and machine learning," *IEEE Transactions on Signal Processing*, vol. 65, no. 13, pp. 3551–3582, 2017. 3, 4, 118, 154

27. A. Smilde, R. Bro, and P. Geladi, *Multi-Way Analysis: Applications in the Chemical Sciences*. John Wiley & Sons, 2004. 3

28. P. Comon and B. Mourrain, "Decomposition of quantics in sums of powersof linear forms," *Signal Processing*, vol. 53, no. 2-3, pp. 93–107, 1996. 3, 70, 71, 122

29. J. Brachat, P. Comon, and B. Mourrain, "Symmetric tensor decomposition," *Linear Algebra and its Applications*, vol. 433, no. 11–12, pp. 1851–1872, 2010. 3, 4, 6, 64, 67, 68, 70, 71, 116, 117, 118, 119, 120, 123, 161, 162

30. P. Comon, G. Golub, L.-H. Lim, and B. Mourrain, "Symmetric tensors and symmetric tensor rank," *SIAM Journal on Matrix Analysis and Applications*, vol. 30, no. 3, pp. 1254–1279, 2008. 3, 4, 71, 118, 122, 123

31. L. D. Lathauwer, *Handbook of Blind Source Separation: Independent Component Analysis and Applications*, ch. 5: Algebraic methods after prewhitening, pp. 155–177. Elsevier Ltd., 2010. 3, 7, 70, 120

32. T. G. Kolda, "Numerical optimization for symmetric tensor decomposition," *Mathematical Programming*, vol. 151, no. 1, pp. 225–248, 2015. 3, 118, 123, 144, 145

33. T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Review*, vol. 51, no. 3, pp. 455–500, 2009. 3, 4, 118, 122, 123, 154

34. P. Comon, X. Luciani, and A. D. Almeida, "Tensor decompositions, alternating least squares and other tales," *Journal of Chemometrics*, vol. 23, pp. 393–405, July - August 2009. 4

35. E. L. Denton, W. Zaremba, J. Bruna, Y. Lecun, and R. Fergus, "Exploiting linear structure within convolutional networks for efficient evaluation," in *Advances in Neural Information Processing Systems 27*, pp. 1269–1277, 2014. 4

36. M. Astrid and S.-I. Lee, "CP-decomposition with tensor power method for convolutional neural networks compression," in *2017 IEEE International Conference on Big Data and Smart Computing*, pp. 115–118, 2017. 4, 118

37. V. de Silva and L.-H. Lim, "Tensor rank and the ill-posedness of the best low-rank approximation problem," *SIAM Journal on Matrix Analysis and Applications*, vol. 30, no. 3, pp. 1084–1127, 2008. 4, 118, 122

38. W. Li, C.-C. Liu, T. Zhang, H. Li, M. S. Waterman, and X. J. Zhou, "Integrative analysis of many weighted co-expression networks using tensor computation," *PLOS Computational Biology*, vol. 7, no. 6, p. e1001106, 2011. 4, 118

39. K. Usevich and P. Comon, "Quasi-hankel low-rank matrix completion: a convex relaxation." https://arxiv.org/abs/1505.07766, 2015. 4, 64, 67, 68, 120, 162

40. K. Usevich and P. Comon, "Hankel low-rank matrix completion: Performance of the nuclear norm relaxation," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, pp. 637–646, June 2016. 4, 68, 120, 162

41. M. Laurent, *Sums of Squares, Moment Matrices and Optimization Over Polynomials*, vol. 149 of *The IMA Volumes in Mathematics and its Applications*, ch. Sums of Squares, Moment Matrices and Optimization Over Polynomials, pp. 157–270. Spinger, 2009. 4, 120, 162

42. M. Laurent, "Revisiting two theorems of Curto and Fialkow on moment matrices," in *Proceedings of the American Mathematical Society*, vol. 133, pp. 2965–2976, 2005. 4, 120, 162

43. R. E. Curto and L. A. Fialkow, "Recursiveness, positivity, and truncated moment problems," *Houston Journal of Mathematics*, vol. 17, no. 4, pp. 603–635, 1991. 4, 120, 162

44. H. Kasai and B. Mishra, "Low-rank tensor completion: a Riemannian manifold preconditioning approach," in *Proceedings of the 33rd International Conference on Machine Learning*, 2016. 4, 122, 123, 162

45. P.-A. Absil and J. Malick, "Projection-like retractions on matrix manifolds," *SIAM Journal on Optimization*, vol. 22, no. 1, pp. 135–158, 2012. 5, 50, 161

46. M. Unser, N. Chenouard, and D. V. D. Ville, "Steerable pyramids and tight wavelet frames in $L_2(\mathbb{R}^d)$," *IEEE Transactions on Image Processing*, vol. 20, pp. 2705–2721, Oct 2011. 6, 11, 30, 32, 90

47. F. Rendl and H. Wolkowicz, "A semidefinite framework for trust region subproblems with applications to large scale minimization," *Mathematical Programming*, vol. 77, no. 1, pp. 273–299, 1997. 7, 64, 137

48. S. Axler, *Linear Algebra Done Right*. Undergraduate Texts in Mathematics, Springer, 3 ed., 2015. 10

49. B. Scholköpf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT, 2002. 15, 16, 79, 83

50. G. E. Fasshauer and M. J. McCourt, "Stable evaluation of gaussian radial basis function interpolants," *SIAM Journal on Scientific Computing*, vol. 34, no. 2, pp. A737–A762, 2012. 18

51. H. Wendland, *Scattered Data Approximation*. Cambridge Monographs on Applied and Computational Mathematics, Cambridge University Press, 2004. 18, 83

52. R. H. Wasserman, *Tensors and Manifolds: With Applications to Physics*. Oxford University Press, 2 ed., 2009. 23, 36

53. S. T. Lovett, *Differential Geometry of Manifolds*. A K Peters/CRC Press, 2010. 23, 26, 35, 36, 38, 40, 41, 45, 47, 48, 49

54. M. Unser, D. Sage, and D. V. D. Ville, "Multiresolution monogenic signal analysis using the riesz-laplace wavelet transform," *IEEE Transactions on Image Processing*, vol. 18, pp. 2402–2418, Nov 2009. 30, 32

55. M. Unser and D. V. D. Ville, "Wavelet steerability and the higher-order riesz transform," *IEEE Transactions on Image Processing*, vol. 19, pp. 636–652, March 2010. 30, 31, 32

56. R. Soulard, P. Carré, and C. Fernandez-Maloigne, "Vector extension of monogenic wavelets for geometric representation of color images," *IEEE Transactions on Image Processing*, vol. 22, pp. 1070–1083, March 2013. 30, 32

57. P. C.-S. Teo, *Theory and Applications of Steerable Functions*. PhD thesis, Stanford University, 1998. 31

58. M. Unser and N. Chenouard, "A unifying parametric framework for 2D steerable wavelet transforms," *SIAM Journal on Imaging Sciences*, vol. 6, no. 1, pp. 102–135, 2013. 32

59. J. M. Lee, *Introduction to Smooth Manifolds*. Springer New York, 2012. 36, 37, 38, 41, 43, 52, 53, 57, 58

60. A. Galbis and M. Maestre, *Vector Analysis Versus Vector Calculus*. Universitext, Springer, 2012. 36

61. R. Coleman, *Calculus on Normed Vector Spaces*. Universitext, Springer, 2012. 36

62. A. Edelman, T. A. Arias, and S. T. Smith, "The geometry of algorithms with orthogonality constraints," *SIAM Journal on Matrix Analysis and Applications*, vol. 20, no. 2, pp. 303–353, 2006. 58

63. H. Ye, "Efficient trust region subproblem algorithms," Master's thesis, University of Waterloo, 2011. 62, 64

64. M. Laurent and B. Mourrain, "A generalized flat extension theorem for moment matrices," *Archiv der Mathematik*, vol. 93, no. 1, pp. 87–98, 2009. 64, 65, 66

65. S. Mallat and G. Peyré, *A Wavelet Tour of Signal Processing: The Sparse Way*. Academic Press, 2009. 77, 79, 81

66. Y. C. Eldar and G. Kutyniok, eds., *Compressed Sensing: Theory and Applications Hardcover*. Combridge University Press, 2012. 77, 79, 81

67. K. He, J. Sun, and X. Tang, "Guided image filtering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 6, pp. 1397–1409, 2013. 79

68. S. Mallat and G. Peyré, "A review of bandlet methods for geometrical image representation," *Numerical Algorithms*, vol. 44, no. 3, pp. 205–234, 2007. 81

69. A. Hyvärinen, J. Hurri, and P. O. Hoyer, *Natural Image Statistics: A Probabilistic Approach to Early Computational Vision*. Springer-Verlag, 2009. 81

70. J. Kovacevic and A. Chebira, "Life beyond bases: The advent of frames (part I)," *IEEE Signal Processing Magazine*, vol. 24, no. 4, pp. 86–104, 2007. 81

71. J. Kovacevic and A. Chebira, "Life beyond bases: The advent of frames (part II)," *IEEE Signal Processing Magazine*, vol. 24, no. 5, pp. 115–125, 2007. 81

72. E. Dubois, *The Structure and Properties of Color Spaces and the Representation of Color Images*. Morgan & Claypool Publishers, 2009. 93

73. Z. Wang and Q. Li, "Information content weighting for perceptual image quality assessment," *IEEE Transactions on Image Processing*, vol. 20, no. 5, pp. 1185–1198, 2011. 97

74. Zhou, "IW-SSIM: Information content weighted structural similarity index for image quality assessment," May 2017. 97

75. M. Tsuboi, J. M. Benevides, and G. J. Thomas, "Raman tensors and their application in structural studies of biological systems," *Proceedings of the Japan Academy, Series B*, vol. 85, no. 3, pp. 83–97, 2009. 120

76. A. Uschmajew and B. Vandereycken, "The geometry of algorithms using hierarchical tensors," *Linear Algebra and its Applications*, vol. 439, no. 1, pp. 133 – 166, 2013. 122

77. J. Girson and S. Aeron, "Tensor completion via optimization on the product of matrix manifolds," in *IEEE 6th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing*, 2015. 122, 123

78. D. Kressner, M. Steinlechner, and B. Vandereycken, "Low-rank tensor completion by Riemannian optimization," *BIT Numerical Mathematics*, vol. 54, no. 2, pp. 447–468, 2014. 122, 123

79. S. Gandy, B. Recht, and I. Yamada, "Tensor completion and low-n-rank tensor recovery via convex optimization," *Inverse Problems*, vol. 27, no. 2, p. 025010, 2011. 122

80. B. W. Bader, T. G. Kolda, J. Sun, D. M. Dunlavy, E. Acar, J. Mayo, E. C. Chi, and S. Hansen, "MATLAB tensor toolbox version 2.6." http://www.sandia.gov/~tgkolda/TensorToolbox/index-2.6.html, February 2015. 123, 144

81. B. W. Bader and T. G. Kolda, "Algorithm 862: MATLAB tensor classes for fast algorithm prototyping," *ACM Transactions on Mathematical Software*, vol. 32, pp. 635–653, December 2006. 123, 144

82. B. Vandereycken, P.-A. Absil, and S. Vandewalle, "A Riemannian geometry with complete geodesics for the set of positive semidefinite matrices of fixed rank," *IMA Journal of Numerical Analysis*, vol. 33, no. 2, pp. 481–514, 2013. 135, 136

83. T. Penzl, "Numerical solution of generalized Lyapunov equations," *Advances in Computational Mathematics*, vol. 8, no. 1-2, pp. 33–48, 1998. 135

84. J. Sun, *When Are Nonconvex Optimization Problems Not Scary?* PhD thesis, Columbia University, 2016. 140, 158

85. K. B. Petersen and M. S. Pedersen, "The matrix cookbook." Technical University of Denmark: http://www2.imm.dtu.dk/pubdb/views/publication_details.php?id=3274, November 2012. 143