UNIVERSITY OF OTTAWA

uOttawa

MASTERS THESIS

Modeling Simplified Reaction Mechanisms using Continuous Thermodynamics for Hydrocarbon Fuels

---

by

Clayton D.L. FOX, P.ENG.

A thesis submitted to the Faculty of Graduate and Postdoctoral Studies in partial fulfillment of the requirements for the degree of

Master of Applied Science (MASc)

in Mechanical Engineering
Ottawa-Carleton Institute for Mechanical and Aerospace Engineering

April 25, 2018

"Fall in love with some activity, and do it! Nobody ever figures out what life is all about, and it doesn't matter. Explore the world. Nearly everything is really interesting if you go into it deeply enough. Work as hard and as much as you want to on the things you like to do the best. Don't think about what you want to be, but what you want to do. Keep up some kind of a minimum with other things so that society doesn't stop you from doing anything at all. "

- Richard P. Feynman

Abstract

Commercial fuels are mixtures with large numbers of components. Continuous thermodynamics is a technique for modelling fuel mixtures using a probability density function rather than dealing with each discreet component. The mean and standard deviation of the distribution are then used to model the chemical reactions of the mixture.

This thesis develops the necessary theory to apply the technique of continuous thermodynamics to the oxidation reactions of hydrocarbon fuels. The theory is applied to three simplified models of hydrocarbon oxidation: a global one-step reaction, a two-step reaction with CO as the intermediate product, and the four-step reaction of Müller et al. (1992), which contains a high- and a low-temperature branch. These are all greatly simplified models of the complex reaction kinetics of hydrocarbons, and in this thesis they are applied specifically to n-paraffin hydrocarbons in the range from n-heptane to n-hexadecane.

The model is tested numerically using a simple constant pressure homogeneous ignition problem using Cantera and compared to simplified and detailed mechanisms for n-heptane. The continuous thermodynamics models are able not only to predict ignition delay times and the development of temperature and species concentrations with time, but also changes in the mixture composition as reaction proceeds as represented by the mean and standard deviation of the distribution function. Continuous thermodynamics is therefore shown to be a useful tool for reactions of multicomponent mixtures, and an alternative to the "surrogate fuel" approach often used at present.

Acknowledgements

I would like to dedicate this work to my wife Sarah, our son Elijah, and our family, for their motivation, support and patience while I completed courses and prepared this thesis.

I would also like to express my sincere gratitude for the opportunity to work with Dr. William L.H. Hallett. I am thankful for his expertise in mathematical modelling, his support with trouble-shooting, and for his time and assistance with editing this manuscript. It was an honour to work together and I will always be grateful.

Lastly, I acknowledge and greatly appreciate the funding for this research provided by the Natural Sciences and Engineering Research Council of Canada (**NSERC**).

Contents

List of Figures

List of Tables

Listings

List of Symbols

Symbol	**Name**	**Unit**
A	pre-exponential factor	kmol/m^3 s
	or frequency factor	1/s
$\breve{a}$	coefficient NASA polynomial	
c	molar density	kmol/m^3
c_p	molar specific heat	kJ/kmol K
$\bar{c}_p$	mixture specific heat	kJ/kmol K
C_nH_{2n+2}	fuels (n-paraffin)	
D_{im}	diffusion of species i in mixture	m^2/s
$\bar{D}$	average diffusion of species	m^2/s
$\hat{D}$	average diffusion of species w.r.t. θ	m^2/s
$\tilde{D}$	average diffusion of species w.r.t. Ψ	m^2/s
dm	Detailed mechanism	
E_a	activation energy	kJ/gmol
E_{EFF}	effective activation energy	kJ/gmol
F	Fuel	
$f(I)$	fuel probability density function	
$f\,'(I)$	intermediate probability density function	
$\bar{g}_f^\circ$	Gibbs energy of formation	kJ/kmol
$\bar{h}$	molar enthalpy	kJ/kmol
$\bar{h}_f^\circ$	enthalpy of formation	kJ/kmol
I	distribution variable (molecular mass)	kg/kmol
I, **X**	intermediate species	
k	reaction rate	gmol/cm^{3}s
K	pre-exponential factor	kmol/m^3 s
	or frequency factor	1/s
K_{EFF}	effective pre-exponetial factor	kmol/m^3 s
M	molecular mass	kg/kmol
δM	change in molecular mass	kg/kmol
N_{GEN}	moles generated	kmol
N_P	number of product moles	kmol

N_R	number of reactant moles	kmol
ΔN	difference of products and reactants	kmol
P	pressure	Pa
$\bar{R}$	universal gas constant	kJ/gmolK
R_i	source term, rate of creation species i	$\mathrm{kmol/m^3\,s}$
$\bar{s}$	molar entropy	kJ/kmol K
$\bar{s}^\circ$	standard entropy	kJ/kmol K
t	time	s
δt	change in time	s
T	temperature	K
δT	change in temperature	K
v	velocity	m/s
v^*	molar average velocity	m/s
W_i	reaction rate species i	$\mathrm{kmol/m^3\,s}$
$\bar{W}$	average reaction rate	$\mathrm{kmol/m^3\,s}$
$\widehat{W}$	average reaction rate w.r.t. θ	$\mathrm{kmol/m^3\,s}$
$\widetilde{W}$	average reaction rate w.r.t. Ψ	$\mathrm{kmol/m^3\,s}$
X_i	mole fraction of species i	
Y_i	mass fraction of species i	
$y(\theta)$	linear equation as function of θ	

Greek Letters:

α	fuel gamma distribution parameter	
β	fuel gamma distribution parameter	
γ	fuel origin of gamma distribution	
$\alpha\,'$	intermediate gamma distribution parameter	
$\beta\,'$	intermediate gamma distribution parameter	
$\gamma\,'$	intermediate origin of gamma distribution	
$\Gamma(\alpha)$	gamma function	
θ	fuel distribution mean (molecular mass)	kg/kmol
σ	fuel distribution standard deviation	kg/kmol
Ψ	fuel second moment about distribution	$\mathrm{kg^2/kmol^2}$
$\theta\,'$	intermediate distribution mean (molecular mass)	kg/kmol
$\sigma\,'$	intermediate distribution standard deviation	kg/kmol
$\Psi\,'$	intermediate second moment about distribution	$\mathrm{kg^2/kmol^2}$
ρ	mass density	$\mathrm{kg/m^3}$
ζ	inverse of temperature	1/K
κ	factor for equalizing units	
ν'_j	stoichiometric coefficient	mol

Subscripts:

$_1$	reaction step 1
$_2$	reaction step 2
$_3$	reaction step 3
$_4$	reaction step 4
$_B$	backward
$_F$	forward
$_\mathbf{F}$	fuel
$_\mathbf{I}$, $_\mathbf{X}$	intermediate species
i	index for chemical species i
n	number of carbon atoms
$_O$	oxygen

Superscripts:

m, n	species concentration exponents

Linearization Coefficients:

a_E, b_E	coefficient in the equation for $E(I)$
a_k, b_k	coefficient in the equation for $K(I)$

Chapter 1

Introduction

1.1 Motivation

Commercial fuels are mixtures containing hundreds or even thousands of discreet components. Detailed kinetic models are challenging enough for pure fuels, and means of modelling reactions for complex mixtures are not available. Typically surrogate fuel models are used, where a surrogate fuel is composed of a small number of pure fuel components which have been selected to simulate the behaviour of a real mixture.

The motivation of this work is to produce a model for predicting the behaviour of fuel mixtures that uses reduced reaction mechanisms for hydrocarbon fuels, and to apply it to the simple test problem of homogeneous constant pressure ignition. In addition, instead of dealing with a large number of discreet fuel components, a model called continuous thermodynamics is used, which treats multi-component fuels as "continuous" mixtures, for which the composition is approximated by a probability density function and characterized by the mean molecular mass and standard deviation of the mixture.

The components present within the mixtures of interest in this work are n-paraffin hydrocarbons which make up a large majority of commercial liquid fuels used today, ranging from n-heptane to n-hexadecane (C_7H_{16} to $C_{16}H_{34}$) in varying concentrations.

1.2 Continuous Thermodynamics

In continuous thermodynamics, a fuel mixture is represented by a probability density function with species molecular mass as the distribution variable, so that the distribution mean is the average molecular mass of the fuel. The standard deviation represents the width of the distribution: a small value approximates a pure fuel, while a large value would be a mixture with many components.

Reduced simplified reaction mechanisms are incorporated based on single, two and four step mechanisms from the literature on pure fuels and have been written in continuous thermodynamics form to represent mixtures, where the stoichiometric coefficients for the oxygen in the reactants and the species in the products are written as a function of fuel mean molecular mass. The governing conservation and reaction rate equations are also written in a continuous thermodynamics form.

To model the thermodynamic properties of the fuel as a mixture, continuous thermodynamics is incorporated which provides specific heat, enthalpy and entropy formulations written as functions of the mean molecular mass. This is done by writing the NASA polynomials for a general n-paraffin fuel in continuous thermodynamics form.

1.3 Simplified Reaction Mechanisms

Reduced, skeletal, and simplified are common terms for reaction mechanisms that have been reduced from a detailed reaction mechanism which contains many more steps and species. In this thesis the terms reduced or simplified will refer to reaction mechanisms containing between one and four steps.

Simplified reaction mechanisms typically contain a minimal number of steps essential to predict reaction details of interest. If more details of the reaction are required, then more steps are needed. For example, a single-step mechanism may be sufficient for predicting ignition delay time within a narrow initial temperature range; however, generally it can not accurately predict final gas temperatures, since it does not include equilibrium reactions for the products. If final gas temperature is not of interest then the single-step model will suffice. Adding more reaction steps adds to the computational cost of the mechanism and depending on the application may be unwarranted.

While detailed reaction mechanisms for lower carbon number hydrocarbons, particularly n-methane, n-heptane, and n-octane have been developed to a great extent, there are a very few multi-component fuel models available.

Pure fuel detailed reaction mechanisms can be quite complex: for example the GRI-3.0 from Smith et al. (2000) for pure methane (CH_4) has 325 reaction steps and 53 species and is widely cited for methane simulations. There is also the detailed mechanism from Mehl et al. (2011) for n-heptane and n-octane which is used in this work for a basis of comparison.

The focus of this research is to produce continuous thermodynamic models for fuel mixtures that can be used to predict the behaviour of simplified reaction mechanisms such as single-step, two-step, and four-step models of mixtures. The models in this work are based on published mechanisms from Westbrook and Dryer (1981), Howard et al. (1973), and Müller et al. (1992) for pure fuels, which have been experimentally verified. They are modified in this work to incorporate continuous thermodynamics.

1.4 Objectives

The objective of this work is to produce a model for mixtures of n-paraffin hydrocarbons which are treated as a probability density function with a mean and standard deviation, represent the reaction kinetics (specifically the rate parameters), reaction mechanisms, and thermal properties in a simplified form using continuous thermodynamics, and test the model within a simple constant pressure perfectly mixed reactor.

1.5 Cantera

Cantera is an open source software designed to be used as a solver or tool kit with programming languages such as Python, Matlab, C++, or FORTRAN 90, for the purpose of solving chemical kinetics, thermodynamics, and transport processes. Cantera is used in this work to assist with solving the thermodynamic and chemical kinetic properties for the constant pressure reactor described in the motivation and objective for this work.

Chapter 2

Literature Survey

2.1 Introduction

This chapter provides a brief overview of the background and theory around research in continuous thermodynamics and simplified reaction mechanisms. It also identifies the original contributions of the present research which build on these foundational works.

2.2 Continuous Thermodynamics

The principle of continuous thermodynamics describes fuel mixtures with a probability density function $f(I)$ rather than as discrete fuel components, where I is the distribution variable representing any chemical property, e.g., molecular mass, boiling point, and carbon number. With this principle, only the distribution parameters (mean θ and standard deviation σ) of the distribution are required to characterize the fuel as a mixture.

The main objective here is to develop continuous thermodynamic formulations for the reaction rates of fuel mixtures. The earliest work on reaction rates in continuous mixtures is that of Aris and Gavalas (1966), who developed a mathematical model and applied it to a simple decomposition reaction $A \to B$. These ideas were followed by Astarita and Ocone (1992), who also dealt with a simple decomposition, which is not directly applicable to combustion problems. Chou and Ho (1988), and recently Ho and White (2015) also applied a theory of reaction rates; however, they defined their rate constant as the distribution variable, which is difficult to integrate into a continuous mixture problem, since the distribution variable cannot easily be linked to the transport properties needed to develop the diffusion and convection terms for the continuity equations.

A significant amount of literature on continuous thermodynamics focuses on droplet evaporation. Tamim and Hallett (1995) were the first to present a full transient model for the evaporation of a single droplet made up of a multicomponent fuel using continuous thermodynamics. This model described the evolution of the vapour composition in space and time as evaporation of a spherical droplet progressed. The gamma distribution function (not to be confused with the gamma function) commonly used in the literature was chosen as the distribution function $f(I)$ to represent a hydrocarbon mixture with an indefinite number of components, and molecular mass was chosen as the distribution variable I. They were also the first to study transport processes (diffusion and convection), and developed transport equations for the distribution function in the vapour phase, written in a spherical coordinate system. They could successfully model the evaporation of real mixtures by selecting the appropriate distribution parameters and applied the model to gasoline and diesel fuels.

The model by Tamim and Hallett (1995) was later simplified by Hallett (2000), since the full transient model would be difficult to include in spray evaporation or combustion calculations. Abdel-Qader and Hallett (2005) investigated evaporation and the role of internal liquid mixing within a continuous droplet model, where liquid phase transport and diffusivity relations were developed. Another real fuel mixture application of continuous thermodynamics is the work by Tolonen et al. (2015), who applied the methods to model evaporating droplets of liquid fuel mixtures derived through fast pyrolysis of chicken litter, and found the model to give accurate predictions of droplet temperature and diameter histories during evaporation.

The first attempt to add chemical reaction in continuous thermodynamics form to a droplet model was that of Sabourin et al. (2016), who conducted auto-ignition experiments on single droplets of liquid mixtures and developed an ignition model by adding a continuous thermodynamics formulation for a single-step global reaction rate expression to the droplet evaporation models developed by Tamim and Hallett (1995) and Abdel-Qader and Hallett (2005). The reaction rate expressions developed were represented as functions of the distribution variable I (molecular mass). For example, rate parameters were experimentally fitted and written as functions of the distribution variable I, and the concentration exponents were set to unity before numerically integrating the reaction rate equations over the distribution function $f(I)$, which is required to get the reaction rate into a continuous thermodynamics form.

Sabourin et al. (2016) found that unlike binary mixtures, ignition is not primarily controlled by the most volatile component, since other components are present in significant concentrations in the vapour phase. They concluded that the models could reproduce ignition delay times of experiments quite well despite the fact they used single-step reaction kinetics.

The most important simplification of the Sabourin et al. (2016) model was that reactant consumption prior to ignition was neglected so that the chemical reaction only appeared in the energy equation, not in the conservation equations for species. In addition, the simplified rate model was a one-step reaction and was not capable of simulating the negative temperature coefficient region (increasing ignition delay for increasing temperature), or two-stage ignition. This present work builds on that of Tamim and Hallett (1995) and Sabourin et al. (2016) to include formulations for multi-step reaction mechanisms to study this phenomena. However, instead of dealing with droplet ignition, the much simpler test case of constant pressure homogeneous ignition is dealt with.

2.3 Simplified Reaction Mechanisms

In real commercial fuel mixtures, the combustion process involves many species and reactions. For example, the widely used detailed mechanism by Mehl et al. (2011) describes the reaction of n-heptane with air. It consists of more than a thousand reactions and involves hundreds of species. Such a mechanism is obviously very complex. Not only would it be unreasonable to develop this kind of detailed mechanism for a myriad of mixtures, it would be unrealistic to write each of these reaction steps in a continuous thermodynamics form. On the other hand, it is very reasonable to apply continuous thermodynamics to simplified reaction mechanisms, since there are quite a few simplified mechanisms that reproduce the essential combustion features (ignition, adiabatic temperature, flame speed, etc.) for hydrocarbon fuels.

2.3.1 Single-step Reaction Mechanisms

The simplest reaction mechanisms are the single-step mechanisms such as those developed by Westbrook and Dryer (1981), who applied them to a pre-mixed laminar flame speed model, and experimentally developed reaction rate parameters by comparing computed and observed flame speeds in selected mixtures of fuel and air. This is a significant paper on simplified mechanisms as it encompasses a large number of fuels. Some of the fuels studied included n-paraffins (methane to n-decane), aromatic hydrocarbons and others.

They found that first order fuel and oxidizer concentration exponents in the reaction rate expressions cannot yield the correct dependence of flame speed on equivalence ratio or on pressure, and cannot predict flammability limit. Thus, they developed reaction rate parameters with non-integer concentration exponents to fit their model with observations for all the fuels they examined. They also made significant simplifications to the transport coefficients for thermal diffusivity and molecular species diffusifities, and they concluded that their experimentally developed parameters should act as initial estimates only, especially if used in other numerical models, and cautioned that further calibration would be required for problems other than laminar flame propagation.

In the droplet ignition model mentioned earlier, Sabourin et al. (2016) incorporated the single-step mechanism like that of Westbrook and Dryer (1981). However, modifications were required to make it compatible with the continuous thermodynamic formulations and the reactant consumption prior to ignition was neglected.

2.3.2 Two-step Reaction Mechanisms

It is well known that the rate controlling step for hydrocarbon combustion at higher temperatures is the oxidation of carbon monoxide (Turns 2012). Hydrocarbon combustion can be modelled as a two-step mechanism. The first step involves a fuel breakdown to fuel and carbon monoxide, the second step involves the final oxidation of carbon monoxide to carbon dioxide. In many two-step models, the second step is an equilibrium reaction, where the forward rate is CO oxidation, and the reverse is CO_2 dissociation. The two-step model by Westbrook and Dryer (1981) for pure n-heptane (C_7H_{16}) is

(R1)
$$C_7H_{16} + 7.5\,(O_2 + 3.76N_2) \rightarrow 7\,CO + 8\,H_2O + 28.2\,N_2 \tag{2.1}$$

(R2)
$$CO + \frac{1}{2}\,O_2 \Leftrightarrow CO_2 \tag{2.2}$$

Adding a second step improves the accuracy of the model. For example a single-step model will over-predict the final adiabatic flame temperature because it does not include the dissociation of the $CO \Leftrightarrow CO_2$ balance, represented by the equilibrium reaction **R2**. Howard et al. (1973) also developed a single-step mechanism to describe the oxidation of carbon monoxide as a forward reaction only, which is an alternative to the equilibrium reaction **R2**.

2.3.3 Three-step Reaction Mechanisms

There are also three-step reaction mechanisms, such as those developed by Gray and Yang (1965) and later by Gonçalves de Azevedo et al. (2014), who successfully applied it to establishing explosion limit criteria. They modelled a three-step reaction mechanism for a gas in a closed spherical vessel. The first step was the initiation of reactant species **P** to intermediates **A**, the second step was a branching step of intermediate species, and the final step was the termination step from intermediates to products **B**.

2.3.4 Four-step Reaction Mechanisms

A four-step reaction gives an ignition delay with a low and a high temperature reaction path. In each path, fuel goes to intermediates, then intermediate species go to products; hence, the term two-stage ignition. Two-stage ignition in real fuels is also the reason for a region called the negative temperature coefficient region, where ignition delay time increases, and the reaction rate slows down with increasing temperature during mid range initial gas temperatures. This behaviour is called the negative temperature coefficient region, and it is also associated with the phenomenon of cool flames. The term cool flames characterizes the phenomena of heat being released during the first stage of ignition without accelerating the reaction as the temperature increases.

A four-step reaction mechanism was developed by Müller et al. (1992), which was simplified from a detailed mechanism for n-heptane which had over a thousand reaction steps, and over 100 chemical species. It was developed numerically by comparison with the detailed kinetics and experimental data, by adjusting the rate coefficients of the four reaction steps to describe constant pressure ignition delay time at high pressures (around 40 atm). They also made significant simplifications when writing the conservation equations by assuming constant mean molecular mass and specific heats. This simplified mechanism is capable of simulating the negative temperature coefficient region, and two-stage ignition. The first two steps account for the high temperature branch, and the other two steps account for the low temperature branch. The transition between these two branches is the negative temperature coefficient region which was accounted for by a reversible third reaction step, an equilibrium reaction.

They reported that their four-step model was in agreement with experimental data for ignition delay, and the kinetic model, with a maximum error of 20% of the kinetic model occurring in the negative temperature coefficient region (NTC). They concluded that the four-step mechanism contains the general features of the full kinetic mechanism.

The limitation here is that it does not predict a true NTC region, because with increasing temperature, ignition delay time and reaction rate still increase, although only very slightly. As mentioned earlier, in the NTC region of a real fuel, the ignition delay time and reaction rate slow down with increasing temperature.

2.3.5 Seven-step Reaction Mechanisms

A seven-step model was adapted by Vandersickel et al. (2013), who noted that the four-step model by Müller et al. (1992) could not model true negative temperature coefficient behaviour, due to the lack of chain branching species and the temperature dependency of the model. The seven-step model presented could predict the true behaviour of the negative temperature coefficient region and the phenomenon of cool flames. In addition to modelling ignition delays, the model could describe the evolution of intermediate species and heat release rates for a variety of full boiling range fuels. They concluded that based on the predictive capability observed for the seven-step model, that in principle, it could be applied to any hydrocarbon fuel, providing suitable adjustments to the model parameters are made.

2.4 Conclusion

The conclusions drawn from the literature survey are:

- Continuous thermodynamics has been shown to successfully model complex mixtures in the area of droplet evaporation and ignition models, hence the principle should extend to other models.
- There are no models currently available to describe simplified multi-step reaction mechanisms for fuel mixtures in a continuous thermodynamics form, or even one-step reactions if one does not assume negligible reactant consumption.

Chapter 3

Model Theory

3.1 Introduction

This chapter develops the continuous thermodynamics theory and equations written for single-, two- and four-step reaction mechanisms, beginning with single-step stoichiometry, conservation equations and reaction rate as applied to the test case of constant pressure ignition. Then multi-step reactions are presented and lastly thermodynamic properties (specific heat, enthalpy, entropy) are developed in continuous thermodynamics form for reactant, product, and intermediate chemical species.

3.2 Continuous Thermodynamics

Commercial fuels are mixtures of a large number of components. The basic idea of continuous thermodynamics is that a multi-component mixture can be approximated using a probability density function $f(I)$. Instead of multiple discrete fuel components, the mixture can then be approximated based on a few properties, namely the moments (mean, standard deviation) of the distribution function. The mole fraction of a particular species of (say) molecular mass I is then

$$X_i = f(I)\delta I \tag{3.1}$$

If the overall mole fraction of fuel in a fuel/air/products mixture is $X_{\mathbf{F}}$, then the mole fraction of an individual component of the fuel becomes

$$X_{\mathbf{F},i} = X_{\mathbf{F}} f(I)\delta I \tag{3.2}$$

Allowing $\delta I \rightarrow dI$, the distribution function $f(I)$ has the property that

$$\int_0^\infty f(I)dI = 1 \tag{3.3}$$

$$\int_0^\infty f(I)IdI = \theta \tag{3.4}$$

$$\int_0^\infty f(I)I^2dI = \Psi \tag{3.5}$$

The gamma distribution (Eq. 3.6) is commonly used in continuous thermodynamics for $f(I)$, and is the basis for this work.

$$f(I) = \frac{(I-\gamma)^{\alpha-1}}{\beta^\alpha\Gamma(\alpha)} \exp\left[-\frac{I-\gamma}{\beta}\right] \tag{3.6}$$

where $\Gamma(\alpha)$ is the gamma function.

The gamma distribution parameters α, β and γ are related by the moments, which are the mean θ and standard deviation σ (or variance σ^2) of the gamma distribution, by

$$\theta = \alpha\beta + \gamma \tag{3.7}$$

$$\sigma^2 = \alpha\beta^2 \tag{3.8}$$

The second moment Ψ, as derived in Tamim and Hallett (1995), is about the origin of the probability distribution function:

$$\Psi = \theta^2 + \sigma^2 \tag{3.9}$$

The distribution variables α and β control the shape of the gamma distribution, and γ is called the distribution origin which is the minimum value of I. The distribution origin γ is set to zero for this work

$$\gamma = 0 \tag{3.10}$$

In order to model a multi-component mixture, the mean of the distribution θ is set to the mean molecular mass of the mixture, i.e. the combined molecular mass of a mixture of fuels ranging from n-heptane to n-hexadecane, and a value of standard deviation σ is selected which best represents the mixture. A pure compound can be modelled with a small standard deviation ($\sigma = 2$, or 3), while a mixture with a large number of components may have a larger variation ($\sigma = 20$, or 30 (Hallett 2017)).

From these formulations the mixture can be modelled based on three properties; the overall mole fraction of the distribution, the mean θ, and standard deviation σ of the mixture.

3.3 Single-step Continuous Thermodynamic Model

3.3.1 Stoichiometry

The derivation given here originated with Hallett (2017). For a single-step reaction involving n-paraffin fuels, a premixed fuel is stoichiometric in fuel and oxidizer if the reactants in the mixture are in exact proportion so that all the fuel could potentially react with and consume all the oxidizer. In most combustion systems the oxidizer is air, which is composed of 21% O_2 and 79% N_2 by mol, which gives the molar ratio of 3.762 mol of N_2 to 1 mol of O_2. In a fuel mixture, the fuel can be represented as $\sum X_i C_{n_i} H_{2n_i+2}$, where X_i is the mole fraction of each fuel species. The overall stoichiometric equation can then be broken into an equation for each fuel species. Thus the balanced single-step stoichiometric reaction equation for a single species i becomes:

$$\sum X_i C_{n_i} H_{2n_i+2} + X_i \left(\frac{3n}{2} + \frac{1}{2} \right) (O_2 + 3.762 N_2) \rightarrow \ldots \tag{3.11}$$

$$\ldots X_i n CO_2 + X_i (n+1) H_2O + 3.762 X_i \left(\frac{3n}{2} + \frac{1}{2} \right) N_2$$

The fuel species molecular mass is chosen as the distribution variable and given the symbol I. For a single species, I is the sum of the carbon and hydrogen molecular masses:

$$I = 12.011n + 1.00794(2n+2) = 14.027n + 2.016 \tag{3.12}$$

Solving,

$$\begin{aligned} I &= 14.027n + 2.016 \\ n &= (I - 2.016)/14.027 \end{aligned} \tag{3.13}$$

Writing the mole fractions X_i in terms of the continuous thermodynamics distribution function:

$$X_i = f(I)\delta I \tag{3.14}$$

The stoichiometric equation (Eq. 3.11) becomes

$$f(I)\;\delta I\;\mathbf{F} + \frac{(3I + 7.979)}{28.054} f(I)\;\delta I\;(O_2 + 3.762N_2) \rightarrow \ldots \tag{3.15}$$

$$\ldots \frac{(I - 2.016)}{14.027} f(I)\;\delta I\;CO_2 + \frac{(I + 12.011)}{14.027} f(I)\;\delta I\;H_2O + \frac{(3I + 7.979)}{28.054} f(I)\;\delta I\;3.762N_2$$

where $\mathbf{F} = C_{n_i}H_{2n_i+2}$. Allowing $\delta I \rightarrow 0$, and integrating over the distribution function from 0 to ∞ gives the continuous mixture stoichiometric equation for complete combustion:

$$\textbf{(R1)} \qquad \mathbf{F} + \frac{(3\theta + 7.979)}{28.054}(O_2 + 3.762N_2) \rightarrow \ldots \tag{3.16}$$

$$\ldots \frac{(\theta - 2.016)}{14.027} CO_2 + \frac{(\theta + 12.011)}{14.027} H_2O + \frac{(3\theta + 7.979)}{28.054} 3.762N_2$$

where θ is the distribution mean, and **R1** represents the first step of the reduced reaction mechanism, and in this case the only step.

3.3.2 Transport Equations

For a single fuel species i, the transport equation for convection, diffusion and reaction of that species is as follows (Tamim and Hallett 1995),

$$\underbrace{\frac{\partial}{\partial t}(cX_i)}_{\text{Transient}} + \underbrace{\nabla \cdot (cv^* X_i)}_{\text{Convection}} = \underbrace{\nabla \cdot (cD_{im} \nabla X_i)}_{\text{Diffusion}} + \underbrace{R_i}_{\text{Source Term}} \tag{3.17}$$

where c is molar density $[\mathrm{kmol/m^3}]$, X_i is mole fraction, v^* is molar average velocity $[\mathrm{m/s}]$, D_{im} is the effective diffusivity of species i in the mixture $[\mathrm{m^2/s}]$, and R_i is the rate of creation of species i by chemical reaction in $[\mathrm{kmol/m^3\,s}]$. This equation is written on a mol basis because it was originally formulated for droplet vaporization. If species i is fuel, R_i will be negative. This equation is put into a continuous thermodynamics form by substituting for the mole fraction in terms of the distribution function:

$$X_i = X_{\mathbf{F}} f(I)_i \Delta I_i \tag{3.18}$$

where ΔI_i is the interval in I centred about the value of I corresponding to species i. The distribution variable I may be any physical property, such as molecular mass, boiling point or carbon number. The molecular mass is chosen for this work. This definition is substituted into the transport equation, the limit as $\Delta I \rightarrow 0$ is taken, and the equation is integrated over the distribution function from $I = 0 \rightarrow \infty$. This integration is done using weighting functions of 1, I and I^2, resulting in transport equations for the fuel mole fraction $X_{\mathbf{F}}$, the distribution mean θ, and standard deviation σ:

$$\frac{\partial}{\partial t}(cX_{\mathbf{F}}) + \nabla \boldsymbol{\cdot} (cv^{*}X_{\mathbf{F}}) = \nabla \boldsymbol{\cdot} (c\bar{D}\nabla X_{\mathbf{F}}) + \bar{W} \tag{3.19}$$

$$\frac{\partial}{\partial t}(cX_{\mathbf{F}}\theta) + \nabla \boldsymbol{\cdot} (cv^{*}X_{\mathbf{F}}\theta) = \nabla \boldsymbol{\cdot} (c\widetilde{D}\nabla X_{\mathbf{F}}\theta) + \widetilde{W} \tag{3.20}$$

$$\frac{\partial}{\partial t}(cX_{\mathbf{F}}\Psi) + \nabla \boldsymbol{\cdot} (cv^{*}X_{\mathbf{F}}\Psi) = \nabla \boldsymbol{\cdot} (c\widehat{D}\nabla X_{\mathbf{F}}\Psi) + \widehat{W} \tag{3.21}$$

where the second central moment $\Psi = \theta^2 + \sigma^2$. Two small diffusion terms have been dropped from the full equation as described in Tamim and Hallett (1995). There are three averaged diffusivities and three averaged reaction rates; finding the latter is now our task.

The corresponding energy equation, as developed by Tamim and Hallett (1995) and Sabourin et al. (2016) is

$$\bar{C}_p\frac{\partial}{\partial t}(cT) + \bar{C}_p\nabla \boldsymbol{\cdot} (cv^{*}T) = \nabla \boldsymbol{\cdot} \lambda\nabla T + \left[(a_C - C_{PA})c\bar{D} + b_c\theta c\widetilde{D}\right]\nabla_F - \bar{W}\Delta H_R \tag{3.22}$$

where ΔH_R is the enthalpy of reaction (assumed the same for all fuel species, in accordance with data for n-paraffin hydrocarbons Sabourin et al. (2016), $\bar{C}_p$ is the mixture specific heat, and the second-last term represents inter-diffusion of species, with notation as defined in Tamim and Hallett (1995).

3.3.3 Reaction Rates

The three reaction rate terms ($\bar{W}$, $\widetilde{W}$ and $\widehat{W}$ in Eq. 3.19-3.21) are the averaged rates for their respective transport equations, and must now be derived. The reaction rate equation for a single-step reaction for a single species is

$$W_i = -K_i c^{m+n} X_i^m X_O^n \exp(-E_i/\bar{R}T) \tag{3.23}$$

where the fuel and oxygen mole fractions have been given exponents m and n for generality. Note that a single step rate model such as this is not capable of simulating the negative temperature coefficient region of reaction, two-stage ignition, or the phenomena of cool flames which accompany it. The negative sign results from the fact that fuel is consumed (a positive rate means that a species is generated). Units are as follows: X_i and X_O are in $[\mathrm{kmol}]$ of i or O per $[\mathrm{kmol}]$ mixture (therefore unit-less), c is the molar density in $[\mathrm{kmol}$ mixture$/\mathrm{m}^3]$, W_i is in $[\mathrm{kmol}$ of fuel species $i/\mathrm{m}^3\,\mathrm{s}]$, and K_i is in whatever units are required to make the left and right hand sides agree. In this case, K_i has units of $\left[\mathrm{m}^{3(m+n-1)}(\mathrm{kmol}\ i\ )^{(1-m)}(\mathrm{kmol}\ O)^{(-n)}\ \mathrm{s}^{-1}\right]$. For the special case of $m = n = 1$, K_i is in $\left[\ \mathrm{m}^3/\mathrm{kmol}\ O\ /\mathrm{s}\right]$ (Sabourin et al. 2016). The molar density is calculated from the ideal gas law:

$$c = P/\bar{R}T \tag{3.24}$$

To get the averaged rates ($\bar{W}$) in continuous thermodynamics form, we introduce the distribution function for X_i (Eq. 3.18), and the expressions for K_i and E_i (Eq. 3.25-3.26) as functions of the distribution variable I, and substitute them into Eq. 3.23. The specific forms of Eq. 3.25 & 3.26 were developed by Sabourin et al. (2016) as fits for the rate constants of n-paraffins:

$$K(I) = \kappa \exp(a_K + b_K I) \tag{3.25}$$

$$E(I) = a_E + b_E I \tag{3.26}$$

Here a_K is dimensionless, b_k has dimensions of [kmol/kg] - i.e. $1/I$ - and κ is a constant equal to unity that gives K the correct units, since the exponential is dimensionless.

To get the averaged rates ($\bar{W}$), we also introduce the gamma distribution function for $f(I)$ into Eq. 3.23. We then integrate over the distribution. As noted in Sabourin et al. (2016), it is impossible to carry out the integration with a value of m other than an integer; therefore we assume $m = 1$. The expression to be integrated is then

$$\bar{W} = \int_0^\infty -\kappa c^{1+n} \exp(a_K + b_K I) X_{\mathbf{F}} X_O^n \frac{(I-\gamma)^{\alpha-1}}{\beta^\alpha \Gamma(\alpha)} \exp\left[-\frac{(I-\gamma)}{\beta}\right] \exp\left[-\frac{(a_E + b_E I)}{\bar{R}T}\right] dI \tag{3.27}$$

where $\Gamma(\alpha)$ is the gamma function (not the same as the gamma distribution). This can be simplified to

$$\bar{W} = \frac{-\kappa c^{1+n} X_{\mathbf{F}} X_O^n}{\beta^\alpha \Gamma(\alpha)} \int_0^\infty (I-\gamma)^{\alpha-1} \exp[A_1 - A_2(I-\gamma)] dI \tag{3.28}$$

or

$$\bar{W} = \frac{-\kappa c^{1+n} X_{\mathbf{F}} X_O^n \exp(A_1)}{\beta^\alpha \Gamma(\alpha) A_2^\alpha} \int_0^\infty A_2^{\alpha-1} (I-\gamma)^{\alpha-1} \exp[-A_2(I-\gamma)] d[A_2(I-\gamma)] \tag{3.29}$$

where A_1 and A_2 are as defined by Sabourin et al. (2016):

$$A_1 = a_K + \gamma b_K - (a_E + \gamma b_E)/\bar{R}T \tag{3.30}$$

$$A_2 = -b_K + b_E/\bar{R}T + 1/\beta \tag{3.31}$$

Using the definition of a gamma function

$$\Gamma(\alpha) = \int\limits_0^\infty t^{\alpha-1}\exp(-t)dt \tag{3.32}$$

this integrates to

$$\bar{W} = \frac{-\kappa c^{1+n} X_{\mathbf{F}} X_O^n \exp(A_1)}{(\beta A_2)^\alpha} \tag{3.33}$$

The other two average rates ($\widetilde{W}$ and $\widehat{W}$) are generated by multiplying by I and I^2 respectively before integration:

$$\widetilde{W} = \frac{-\kappa c^{1+n} X_{\mathbf{F}} X_O^n \exp(A_1)}{\beta^\alpha \Gamma(\alpha)} \int\limits_0^\infty (I-\gamma)^{\alpha-1} I \exp[-A_2(I-\gamma)]dI \tag{3.34}$$

or, expanding so as to change the variable to $(I-\gamma)$,

$$\widetilde{W} = \frac{-\kappa c^{1+n} X_{\mathbf{F}} X_O^n \exp(A_1)}{\beta^\alpha \Gamma(\alpha)} \int\limits_0^\infty (I-\gamma)^{\alpha-1}[(I-\gamma)+\gamma] I \exp[-A_2(I-\gamma)]dI \tag{3.35}$$

which integrates to

$$\widetilde{W} = \frac{-\kappa c^{1+n} X_{\mathbf{F}} X_O^n \exp(A_1)}{(\beta A_2)^\alpha} \left(\frac{\alpha}{A_2} + \gamma \right)$$

$$\widetilde{W} = \bar{W} \left(\frac{\alpha}{A_2} + \gamma \right) \tag{3.36}$$

where use has been made of the identity $\Gamma(\alpha + 1) = \alpha\Gamma(\alpha)$. This is identical to Eq. 9 in Sabourin et al. (2016).

Finally, with weighting I^2:

$$\widehat{W} = \frac{-\kappa c^{1+n} X_{\mathbf{F}} X_O^n \exp(A_1)}{\beta^\alpha \Gamma(\alpha)} \int_0^\infty (I-\gamma)^{\alpha-1} \left[(I-\gamma)^2 + 2\gamma(I-\gamma) + \gamma^2 \right] \exp[-A_2(I-\gamma)] dI \tag{3.37}$$

which integrates to

$$\widehat{W} = \frac{-\kappa c^{1+n} X_{\mathbf{F}} X_O^n \exp(A_1)}{(\beta A_2)^\alpha} \left(\frac{\alpha(\alpha+1)}{A_2^2} + \frac{2\gamma\alpha}{A_2} + \gamma^2 \right)$$

$$\widehat{W} = \bar{W} \left(\frac{\alpha(\alpha+1)}{A_2^2} + \frac{2\gamma\alpha}{A_2} + \gamma^2 \right) \tag{3.38}$$

3.3.4 Rate Equation in Arrhenius form

In order to use this theory with Cantera (a chemical kinetic solver tool), the rate must be put into the form of an Arrhenius rate equation. The general equation for the

Arrhenius forward rate coefficient K_f is

$$K_f = AT^b \exp\left(-\frac{E_a}{\bar{R}T}\right) \tag{3.39}$$

where A is the pre-exponential coefficient, b is the dimensionless temperature exponent, and E_a is the activation energy.

For our calculations in continuous thermodynamics this equation looks like this:

$$\bar{W} = -K_{EFF}\, c^{1+n} X_{\mathbf{F}} X_O^n \exp\left(-\frac{E_{EFF}}{\bar{R}T}\right) \tag{3.40}$$

where K_{EFF} is the effective pre-exponential coefficient, E_{EFF} is the effective activation energy. The dimensionless temperature exponent is set to $b = 0$, which is consistent with the assumptions made by both Westbrook and Dryer (1981) and Müller et al. (1992). Our task is to formulate expressions for E_{EFF} and K_{EFF}.

If we let

$$A_E = a_E + \gamma b_E \tag{3.41}$$

$$A_K = a_K + \gamma b_K \tag{3.42}$$

then we can re-write Eq. 3.33 as

$$\bar{W} = -\frac{\exp(A_K)}{(\beta A_2)^\alpha} \kappa c^{1+n} X_{\mathbf{F}} X_O^n \exp(-A_E/\bar{R}T) \tag{3.43}$$

The problem with this is that the term A_2 contains a $(1/T)$ dependence (see Eq. 3.31) and hence contributes to the effective activation energy of the reaction. In the limit as the distribution becomes very narrow ($\sigma \rightarrow 0$) the pre-exponential and activation energy should approach the values obtained by substituting $I = \theta$ into the expressions for $K(I)$ and $E(I)$ (Eq. 3.25, 3.26), but the expression above does not give this. The following leads to an expression which satisfies this requirement.

The purpose is to find the effective activation energy, E_{EFF}, and the effective pre-exponential, K_{EFF}, such that

$$\frac{\bar{W}}{-c^{1+n} X_{\mathbf{F}} X_O^n} = R = \frac{\kappa \exp(A_1)}{(\beta A_2)^\alpha} = K_{EFF} \exp(-E_{EFF}/\bar{R}T) \tag{3.44}$$

where R is the reaction rate without concentration of species. This gives a linear relationship between $\ln R$ and $\zeta = 1/T$, whose slope is E_{EFF}:

$$\ln R = A_1 - \alpha \ln(\beta A_2)$$

$$\frac{d}{d\zeta}(\ln R) = \frac{dA_1}{d\zeta} - \frac{\alpha}{\beta A_2}\frac{d}{d\zeta}(\beta A_2) = \frac{-E_{EFF}}{\bar{R}} \tag{3.45}$$

which when simplified gives

$$E_{EFF} = a_E + \frac{b_E \alpha}{A_2} + \gamma b_E \tag{3.46}$$

As $\sigma \rightarrow 0$, $\beta \rightarrow$ small, and $1/\beta$ becomes the dominant term in (Eq. 3.31), so that $(\alpha/A_2) \rightarrow \alpha\beta = (\theta - \gamma)$, and $E_{EFF} \rightarrow a_E + b_E\theta$ as required (compare to Eq. 3.26). The term A_2 is still dependent on temperature, but the dependency is much weaker than in the original rate equation, where A_2 was raised to the power α. The maximum value of σ required for a distribution function resembling a commercial fuel is about 30 $\mathrm{kg/kmol}$; for this value, E_{EFF} varies by about 10% between 500 K and 2000 K, a small variation.

To find K_{EFF}, we substitute E_{EFF} into the original equation (Eq. 3.44):

$$K_{EFF} = \frac{\kappa \exp(A_1)}{(\beta A_2)^{\alpha} \exp(-E_{EFF}/\bar{R}T)} \tag{3.47}$$

which yields

$$K_{EFF} = \kappa(\beta A_2)^{-\alpha} \exp\left(a_K + \gamma b_K + \frac{b_E \alpha}{A_2 \bar{R}T}\right) \tag{3.48}$$

which can be re-written as

$$K_{EFF} = \kappa \exp(a_K + b_K\theta)(\beta A_2)^{-\alpha} \exp\left[-(\theta - \gamma)b_K + \frac{b_E \alpha}{A_2 \bar{R}T}\right] \tag{3.49}$$

The first term is the pre-exponential from the correlation (Eq. 3.25), while the second term can be seen as a correction to it. Again, as $\sigma \to 0$, $(\alpha/A_2) \to \alpha\beta = (\theta - \gamma)$, and this equation can be approximated as

$$K_{EFF} \approxeq \kappa \exp(a_K + b_K\theta)(\beta A_2)^{-\alpha} \exp\left[-(\theta - \gamma)\left(b_K - \frac{b_E}{\bar{R}T}\right)\right] \tag{3.50}$$

It can be shown that

$$K_{EFF} \to \exp(a_K + b_K\theta) \tag{3.51}$$

as $\sigma \to$ small.

Using (Eq. 3.31):

$$(\beta A_2)^{-\alpha} \exp\left[-(\theta - \gamma)\left(b_K - \frac{b_E}{\bar{R}T}\right)\right] = (\beta A_2)^{-\alpha} \exp\left[-\alpha\beta(\frac{1}{\beta} - A_2)\right] \tag{3.52}$$

In the limit as $\sigma \to 0$, $\beta \to$ small, and $A_2 \to 1/\beta$, so that

$$(\beta A_2)^{-\alpha} \exp\left[-(\theta - \gamma)\left(b_K - \frac{b_E}{\bar{R}T}\right)\right] \to 1 \tag{3.53}$$

For $\sigma = 30$, K_{EFF} roughly triples in going from $500 - 2000$ K.

3.3.5 Simplified Rate Expressions

In this thesis, as in most earlier work, the distribution origin will be set to zero: $\gamma = 0$. For this special case, Eq. 3.30 & 3.31 simplify to

$$A_1 = a_K - a_E/\bar{R}T \tag{3.54}$$

$$A_2 = -b_K - b_E/\bar{R}T + 1/\beta \tag{3.55}$$

and Eq. 3.44 becomes

$$\bar{W} = \frac{-\kappa c^{1+n} X_{\mathbf{F}} X_O^n \exp(A_1)}{(\beta A_2)^\alpha} = -K_{EFF} c^{1+n} X_{\mathbf{F}} X_O^n \exp(-E_{EFF}/\bar{R}T) \tag{3.56}$$

where $n = 1$ for the present work, and

$$E_{EFF} = a_E + b_E\alpha/A_2 \tag{3.57}$$

$$\begin{aligned} K_{EFF} &= \kappa(\beta A_2)^{-\alpha} \exp\left(a_K + \frac{b_E\alpha}{A_2\bar{R}T}\right) \\ &= \kappa \exp(a_K + b_K\theta)(\beta A_2)^{-\alpha} \exp\left(-\theta b_K + \frac{b_E\alpha}{A_2\bar{R}T}\right) \end{aligned} \tag{3.58}$$

and

$$\widetilde{W} = \bar{W}\left(\frac{\alpha}{A_2}\right) \tag{3.59}$$

$$\widehat{W} = \bar{W}\left(\frac{\alpha(\alpha+1)}{A_2^2}\right) \tag{3.60}$$

The reaction rate parameters a_K, and a_E are from Table 3.1. These values are from Sabourin et al. (2016) and were fitted to experimental data for droplet ignition while neglecting reactant consumption. They will be adjusted to fit results from a detailed reaction mechanism in the present problem. This process is carried out in Chapter 5.

TABLE 3.1: Correlations for reaction rate parameters n-paraffins (Linearization Coefficients)

	a_K	b_K	a_E	b_E
n-paraffins	13.253	0.03048	73.529	0.2270

(Sabourin et al. 2016)

3.3.6 Application to Constant Pressure Ignition

Our test case is spontaneous ignition at constant pressure in a mixture of uniform composition. For this special case, there is no diffusion, since the composition is uniform, and the conservation equations simplify to

$$\frac{\partial}{\partial t}(cX_{\mathbf{F}}) + \nabla \cdot (cv^* X_{\mathbf{F}}) = \bar{W} \tag{3.61}$$

$$\frac{\partial}{\partial t}(cX_{\mathbf{F}}\theta) + \nabla \cdot (cv^* X_{\mathbf{F}}\theta) = \widetilde{W} \tag{3.62}$$

$$\frac{\partial}{\partial t}(cX_{\mathbf{F}}\Psi) + \nabla \cdot (cv^* X_{\mathbf{F}}\Psi) = \widehat{W} \tag{3.63}$$

Note that the averaged reaction rates, $\bar{W}$, $\widetilde{W}$, and $\widehat{W}$ are all negative, since fuel is being consumed. The convection terms are not zero, because the gas expands as the temperature rises at constant pressure.

The corresponding energy equation is likewise simplified from the full equation (Eq. 3.22):

$$\bar{C}_p \frac{\partial}{\partial t}(cT) + \bar{C}_p \nabla \cdot (cv^* T) = -\bar{W}\Delta H_R \tag{3.64}$$

A further equation is required to solve for the convection (or expansion) term:

$$\frac{\partial c}{\partial t} + \nabla \cdot (cv^*) = N_{GEN} \tag{3.65}$$

This equation results from writing continuity equations for all of the mixture components (including air and product species) and summing them. N_{GEN} is the net rate of production of mols by all of the chemical reactions involved. An expression for

N_{GEN} can be derived by comparing the mass and mol forms of the overall continuity equation:

$$\frac{\partial c}{\partial t} + \nabla \cdot (cv^*) = N_{GEN} \tag{3.65}$$

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho v) = 0 \tag{3.66}$$

where the latter equates to zero because mass cannot be created or destroyed, whereas mols can. The difference between the molar average and mass average velocities v and v^* is a function of the rates of diffusion of different species; however, since the mixture composition is uniform, there is no diffusion, and these velocities are equal (Bird et al. 1961). Substituting $\rho = cM$, where M is the mixture molecular mass, into the mass version of the equation gives

$$M\left[\frac{\partial c}{\partial t} + \nabla \cdot (cv)\right] = -c\left[\frac{\partial M}{\partial t} + v \cdot \nabla M\right] \tag{3.67}$$

Since concentrations are uniform, the last term is again zero, and we have

$$\frac{\partial c}{\partial t} + \nabla \cdot (cv) = -\frac{c}{M}\frac{\partial M}{\partial t} = N_{GEN} \tag{3.68}$$

The expansion term can then be solved for as

$$\nabla \cdot (cv^*) = N_{GEN} - \frac{\partial c}{\partial t} \tag{3.69}$$

From the ideal gas law, assuming constant pressure

$$c = P/\bar{R}T \tag{3.24}$$

$$\frac{dc}{dt} = \frac{P}{\bar{R}}\frac{d}{dt}\left(\frac{1}{T}\right) = -\frac{c}{T}\frac{dT}{dt} \tag{3.70}$$

The expansion term then becomes

$$\nabla \cdot (cv^*) = -\frac{c}{M}\frac{\partial M}{\partial t} - \frac{\partial c}{\partial t} = -\frac{c}{M}\frac{\partial M}{\partial t} + \frac{c}{T}\frac{\partial T}{\partial t} \tag{3.71}$$

The physical interpretation of this term is that when it is positive, it indicates expansion when temperature increases and when the molecular mass goes down. The latter results when mols are produced, leading to more mols for the same mass.

With the expansion term found, the transport equations can be solved for the changes in $(cX_{\mathbf{F}})$, $(cX_{\mathbf{F}}\theta)$, and $(cX_{\mathbf{F}}\Psi)$:

$$\frac{\partial}{\partial t}(cX_{\mathbf{F}}) = \bar{W} - \nabla \cdot (cv^* X_{\mathbf{F}}) = \bar{W} - X_{\mathbf{F}}\nabla \cdot (cv^*) - cv^* \cdot \nabla X_{\mathbf{F}} \tag{3.72}$$

The last term is 0, because the gradient of $X_{\mathbf{F}}$ is 0 (uniform concentration assumed), and therefore

$$\frac{\partial}{\partial t}(cX_{\mathbf{F}}) = \bar{W} - X_{\mathbf{F}}\nabla \cdot (cv^*) \tag{3.73}$$

Likewise

$$\frac{\partial}{\partial t}(cX_{\mathbf{F}}\theta) = \widetilde{W} - X_{\mathbf{F}}\theta\nabla \cdot (cv^*) \tag{3.74}$$

$$\frac{\partial}{\partial t}(cX_{\mathbf{F}}\Psi) = \widehat{W} - X_{\mathbf{F}}\Psi\nabla \cdot (cv^*) \tag{3.75}$$

Note once again that the averaged reaction rates, $\bar{W}$, $\widetilde{W}$, and $\widehat{W}$ all have negative values.

3.4 Multi-Step Reaction Models

3.4.1 Two-step Continuous Thermodynamic Model

3.4.1.1 Stoichiometry

The Westbrook and Dryer (1981) two-step reaction, for n-paraffin hydrocarbons, reaction of a single species with mole fraction X_i:

$$\textbf{(R1)} \qquad \sum X_i C_{n_i} H_{2n_i+2} + X_i\left(n + \frac{1}{2}\right)(O_2 + 3.762N_2) \rightarrow \ldots \tag{3.76}$$

$$\ldots X_i nCO + X_i(n+1)H_2O + 3.762X_i\left(n + \frac{1}{2}\right)N_2$$

$$\textbf{(R2)} \qquad CO + \frac{1}{2}O_2 \Leftrightarrow CO_2 \tag{3.77}$$

Selecting molecular mass as the distribution variable I and using (Eq. 3.13) and (Eq. 3.14) in equation (Eq. 3.76) and integrating over the distribution function from 0 to ∞ gives the continuous stoichiometric equation for the first step:

$$\textbf{(R1)} \qquad \mathbf{F} + \frac{(\theta + 4.998)}{14.027}(O_2 + 3.762N_2) \rightarrow \ldots \tag{3.78}$$

$$\ldots \frac{(\theta - 2.016)}{14.027}CO + \frac{(\theta + 12.011)}{14.027}H_2O + \frac{(\theta + 4.998)}{14.027}3.762N_2$$

$$\textbf{(R2)} \qquad CO + \frac{1}{2}\,O_2 \Leftrightarrow CO_2 \tag{3.77}$$

where **F** $= C_nH_{2n+2}$, θ is the distribution mean, and **R1** represents the first step of the reduced reaction mechanism, and in this case the first of two steps.

3.4.1.2 Reaction Rates

The two-step scheme reacts fuel to CO and H_2O, then reacts the CO to CO_2. Westbrook and Dryer (1981) give this scheme for a number of hydrocarbons. Continuous thermodynamics can be used for the first step, while the second step would be the same for all fuels. However, Westbrook and Dryer's first step can not be used without modification, because it has non-integer reaction orders for all fuels, whereas the continuous thermodynamics formulation just developed requires a whole number (preferably unity) reaction order for the fuel. The first step will therefore be assumed first order in fuel and oxygen where Eq. 3.23 becomes:

$$W_i = -K_i c^2 X_i X_O \exp(-E_i/\bar{R}T) \tag{3.79}$$

The simplified continuous thermodynamics expressions (Sec 3.3.5) already developed for the rate of a single-step reaction will hold here as well.

The forward reaction rate for the second step **R2** (Eq. 3.77) from Westbrook and Dryer (1981) is then:

$$k_F = \frac{d[CO]}{dt} = 10^{14.6}[CO][H_2O]^{0.5}[O_2]^{0.25}\exp(-40/\bar{R}T) \;\; (\mathrm{gmol/cm^3 s}) \tag{3.80}$$

where concentrations are in $\mathrm{gmol/cm^3}$ and the activation energy is in $\mathrm{kcal/gmol}$. The reverse reaction from Westbrook and Dryer (1981) is also added to this to produce equilibrium:

$$k_B = \frac{d[CO_2]}{dt} = 5.0 \times 10^8[CO_2]\exp(-40/\bar{R}T) \;\; (\mathrm{gmol/cm^3 s}) \tag{3.81}$$

Converting the concentration units to $\mathrm{kmol/m^3}$ and the activation energies to $\mathrm{kJ/gmol}$:

(Rate: **R2**$_F$)

$$\begin{aligned} k_F = \frac{d[CO]}{dt} &= 10^{12.35}[CO][H_2O]^{0.5}[O_2]^{0.25}\exp(-167/\bar{R}T) \;\; (\mathrm{kmol/m^3\,s}) \qquad (3.80) \\ &= 22.39 \times 10^{11}[CO][H_2O]^{0.5}[O_2]^{0.25}\exp(-20\,138/T) \;\; (\mathrm{kmol/m^3\,s}) \end{aligned}$$

(Rate: **R2**$_B$)

$$k_B = \frac{d[CO_2]}{dt} = 5.0 \times 10^8[CO_2]\exp(-167/\bar{R}T) \;\; (\mathrm{kmol/m^3\,s}) \qquad (3.81)$$

An **alternative** rate expression for this reaction is that of Howard et al. (1973):

$$k_F = \frac{d[CO]}{dt} = 1.3 \times 10^{11}[CO][H_2O]^{0.5}[O_2]^{0.5}\exp(-15\,105/T) \;\; (\mathrm{kmol/m^3\,s}) \tag{3.82}$$

with a reverse reaction as devised by Cooper and Hallett (2000):

$$k_B = \frac{d[CO_2]}{dt} = 7.5 \times 10^{11}[CO_2]\exp(-46\,500/T) \;\; (\mathrm{kmol/m^3\,s}) \tag{3.83}$$

Concentrations in these two expressions (Eq. 3.80-3.81) are in $\mathrm{kmol/m^3}$. The presence of an H_2O term in the forward rate reflects the fact that water vapour can substantially accelerate CO oxidation, even though it does not participate in the overall reaction.

We shall take the second step CO reaction (Eq. 3.82-3.83) as a given, then fit the parameters of the continuous thermodynamics first step so as to reproduce the ignition times of the full kinetic scheme for n-heptane.

3.4.2 Four-step Continuous Thermodynamic Model

There are a number of models which feature two parallel reaction pathways, one favoured at high temperature and the other at low. This sort of model is necessary to reproduce the negative temperature coefficient region, the phenomena of two-stage ignition and cool flames. Examples are the four-step model of Müller et al. (1992) and the seven-step model of Vandersickel et al. (2013). These models have two separate parallel reactions which consume fuel, and therefore there will be at least two reactions to be modelled using continuous thermodynamics.

The four-step mechanism by Müller et al. (1992) for n-heptane was originally reduced from a detailed mechanism by adjusting the rate coefficients for pressures close to 40 atm. The first two steps of the mechanism account for the high-temperature (HT) branch of the ignition delay curve, while the other two steps account for the

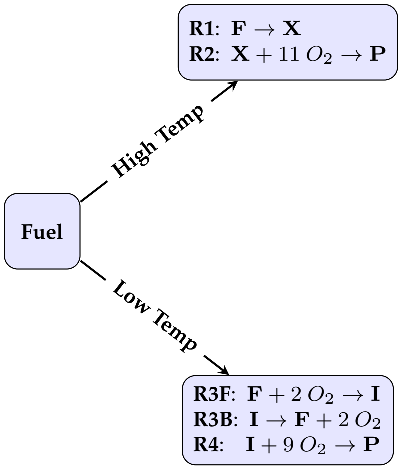


FIGURE 3.1: Four-step model high and low temperature branches

low-temperature (LT) branch shown in Figure 3.1. A minimum of a four-step mechanism is required to model the ignition delay time as it passes through the negative temperature coefficient region during transition between the high-temperature and low-temperature branches. This is achieved in the four-step model by incorporating a reversible third reaction step (Reaction: $\mathbf{R3}_F$ & $\mathbf{R3}_B$).

The reaction steps from Müller et al. (1992) are shown in Eq 3.84 - 3.87:

$$(\mathbf{R1}) \qquad \mathbf{F} \rightarrow \mathbf{X} \tag{3.84}$$

$$(\mathbf{R2}) \qquad \mathbf{X} + 11\,O_2 \rightarrow \mathbf{P} \tag{3.85}$$

$$(\mathbf{R3}_F \;\&\; \mathbf{R3}_B) \qquad \mathbf{F} + 2\,O_2 \Leftrightarrow \mathbf{I} \tag{3.86}$$

$$(\mathbf{R4}) \qquad \mathbf{I} + 9\,O_2 \rightarrow \mathbf{P} \tag{3.87}$$

where **F** stands for the fuel (n-heptane in Müller et al. 1992), **X** and **I** represent the combined intermediates and **P** represents the combination of products:

$$\mathbf{F} = C_7H_{16} \tag{3.88}$$

$$\mathbf{X} = 3\,C_2H_4 + CH_3 + H \tag{3.89}$$

$$\mathbf{I} = HO_2R''O + H_2O \tag{3.90}$$

$$\mathbf{P} = 7\,CO_2 + 8\,H_2O \tag{3.91}$$

For n-heptane $R'' = C_7H_{13}$ and thus the $HO_2R''O$ species ($HO_2 - C_7H_{13} - O$) has 14 H, 7 C and 3 O atoms.

Unlike the one- and two-step models, the reaction rate parameters for the four-step model were not changed from those published by Müller et al. (1992), which are shown in Table 3.2:

TABLE 3.2: Original four-step rate parameters **without** continuous thermodynamics

Reactions	**Pre-exponential**	**Activation Temperatures**
Reaction: **R1**	$A_1 = 1 \times 10^{10}$ (s^{-1})	$E_1/\bar{R} = 21\,650$ K
Reaction: **R2**	$A_2 = 2 \times 10^{12}$ (cm^3 mol^{-1} s^{-1})	$E_2/\bar{R} = 7\,217$ K
Reaction: **R3**$_F$	$A_{3_f} = 3 \times 10^{18}$ (cm^3 mol^{-1} s^{-1})	$E_{3_f}/\bar{R} = 21\,650$ K
Reaction: **R3**$_B$	$A_{3_b} = 4 \times 10^{22}$ (s^{-1})	$E_{3_b}/\bar{R} = 37\,285$ K
Reaction: **R4**	$A_4 = 5 \times 10^{13}$ (cm^3 mol^{-1} s^{-1})	$E_4/\bar{R} = 13\,230$ K

Müller et al. (1992)

The original reaction rates **without** continuous thermodynamics become:

$$\text{(Rate: }\mathbf{R1}\text{)} \qquad k_{R1} = 1 \times 10^{10} [\mathbf{F}]^1 \exp(-21\,650/T) \tag{3.92}$$

$$\text{(Rate: }\mathbf{R2}\text{)} \qquad k_{R2} = 2 \times 10^{12} [\mathbf{X}]^1 [O_2]^1 \exp(-7\,217/T) \tag{3.93}$$

$$\text{(Rate: }\mathbf{R3}_F\text{)} \qquad k_{R3_F} = 3 \times 10^{18} [\mathbf{F}]^1 [O_2]^1 \exp(-21\,650/T) \tag{3.94}$$

$$\text{(Rate: }\mathbf{R3}_B\text{)} \qquad k_{R3_B} = 4 \times 10^{22} [\mathbf{I}]^1 \exp(-37\,285/T) \tag{3.95}$$

$$\text{(Rate: }\mathbf{R4}\text{)} \qquad k_{R4} = 5 \times 10^{13} [\mathbf{I}]^1 [O_2]^1 \exp(-13\,230/T) \tag{3.96}$$

Note that in each case the units work out such that the reaction rates k_{Ri} are in $\mathrm{mol/cm^3\,s}$.

3.4.2.1 Form of reaction equations

The reaction equations for this model are presented by Müller et al. (1992) in terms of mole fraction rather than concentration. Concentration, however, is the fundamentally correct form for reaction equations. Consider the first step of the low-temperature path (Reaction: **R3**$_F$ & **R3**$_B$), and for simplicity assume this is the only reaction consuming fuel:

$$(\mathbf{R3}_F \ \&\ \mathbf{R3}_B) \qquad \mathbf{F} + 2\,O_2 \Leftrightarrow \mathbf{I} \tag{3.97}$$

For constant pressure homogeneous ignition, the conservation equation for fuel simplifies to

$$\frac{\partial}{\partial t}(cX_{\mathbf{F}}) + \nabla \boldsymbol{\cdot} (cv^* X_{\mathbf{F}}) = W \tag{3.98}$$

with

$$\begin{aligned} W &= -A_{3f}[\mathbf{F}][O_2]\exp(-E_{3f}/\bar{R}T) \\ &= -A_{3f}\, c^2 X_{\mathbf{F}} X_O \exp(-E_{3f}/\bar{R}T) \end{aligned} \tag{3.99}$$

where W is the reaction rate of a pure fuel, and where A_{3_f} and E_{3_f} are from Table 3.2. The units are $\mathrm{kmol}\,\mathbf{F}/\mathrm{m}^3\mathrm{s}$. Expanding the derivatives

$$c\frac{\partial X_{\mathbf{F}}}{\partial t} + cv^* \boldsymbol{\cdot} \nabla X_{\mathbf{F}} + X_{\mathbf{F}}\frac{\partial c}{\partial t} + X_{\mathbf{F}}\nabla \boldsymbol{\cdot} (cv^*) = W \tag{3.100}$$

The third and fourth terms are the continuity equation; they sum to the net number of mols N_{GEN} produced by the reaction (Eq. 3.65 above). Müller et al. (1992) assume as an approximation that the mean molecular mass is constant, which is equivalent to assuming that the reaction is equimolal; this makes these two terms zero. The second term is also zero because the concentration is uniform, and the equation then simplifies to

$$\begin{aligned} \frac{\partial X_{\mathbf{F}}}{\partial t} &= -A_{3f}\, cX_{\mathbf{F}} X_O \exp(-E_{3f}/\bar{R}T) \\ &= -A_{3f}\, \frac{P}{\bar{R}T} X_{\mathbf{F}} X_O \exp(-E_{3f}/\bar{R}T) \end{aligned} \tag{3.101}$$

This is the form that Müller et al. (1992) give for the **Reaction R3**$_F$. Note that their rate constant now becomes $(A_{3f}P/\bar{R}T)$. If, however, one is calculating the rate using concentrations (i.e. in the form of Eq. 3.99), as one usually does, and as Cantera does, the factor $(P/\bar{R}T)$ should be omitted and only the pre-exponential A_{3f} should be used. Note that the coefficients A_{3f}, etc. mean exactly the same thing here as they do in the original paper by Müller et al. (1992).

3.4.2.2 Stoichiometry

The high temperature path is

$$(\mathbf{R1}) \qquad \mathbf{F} \rightarrow \mathbf{X} \tag{3.102}$$

$$(\mathbf{R2}) \qquad \mathbf{X} + 11\ O_2 \rightarrow \mathbf{P} \tag{3.103}$$

The fuel is n-heptane, so $\mathbf{P} = 7\ CO_2 + 8\ H_2O$. **X** represents three (3) intermediate species, but the stoichiometric equation says that 1 mol of **F** produces 1 mol of **X**, not the 5 mol of individual species that are lumped together in **X** (see Eq. 3.89). In writing the rate equation we therefore treat **X** as a unit, and the rate of **Reaction R2** is

$$(\text{Rate: }\mathbf{R2}) \qquad \frac{\partial[\mathbf{X}]}{\partial t} = -A_2[\mathbf{X}][O_2]\exp(-E_2/\bar{R}T) \tag{3.104}$$

As noted in the previous section, the factor $(P/\bar{R}T)$ which appears in Müller et al. (1992), does not appear when the equation is written in terms of concentrations.

For a continuous thermodynamics formulation, the stoichiometry of **X** is the same as for n-heptane. For simplicity, we will assume that the properties of **X** do not change when the fuel becomes a distribution. **Reaction R1** says that 1 mol of **F** becomes 1

mol of **X**, so there is no possibility of variation there, and the stoichiometry does not change with the continuous reaction rate; the first reaction therefore, does not need to be written in a continuous form. However, in **Reaction R2** the stoichiometry must vary to give the correct products when the fuel changes, hence there must be a theta-dependent stoichiometry:

$$(\mathbf{R2}) \qquad \mathbf{X} + \frac{(3\theta+8)}{28}O_2 \rightarrow \frac{(\theta-2)}{14}CO_2 + \frac{(\theta+12)}{14}H_2O \tag{3.105}$$

the same as for an n-paraffin in the one-step reaction.

For the low temperature path, it will again be assumed that the oxygen requirement for the first step (Reaction: $\mathbf{R3}_F$ & $\mathbf{R3}_B$) remains the same as fuel molecular mass changes. The justification for this is that the intermediate **I** changes as the fuel changes only in the nature of $\mathbf{R}''$. We shall again assume that **I** remains the same in properties as the fuel changes. The stoichiometry reflected in continuous form for **Reaction 4** becomes

$$(\mathbf{R4}) \qquad \mathbf{I} + \frac{(3\theta-20)}{28}O_2 \rightarrow \frac{(\theta-2)}{14}CO_2 + \frac{(\theta+12)}{14}H_2O \tag{3.106}$$

3.4.2.3 Rate Expressions

The initial step in a reaction where fuel breaks down into intermediates is unique to a particular fuel and therefore the reactions that convert fuel to intermediates should be the ones whose rates are dealt with using continuous thermodynamics. Therefore, in the Müller et al. (1992) four-step mechanism, the two reactions that get continuous reaction rate expressions will be **Reaction R1** and **Reaction** $\mathbf{R3}_F$. We therefore need to apply expressions for rate constants as a function of the distribution variable I in the form $a + bI$ to each of these two reactions as we did in Eq. 3.25 & 3.26. In addition we do not know the dependence of rate on the distribution variable I for a continuous fuel **F** because we have no information on rates for fuels other than n-heptane using this scheme, so we shall have to assume that the b coefficient in each case is the same as it is in the Sabourin et al. (2016) model. The a coefficients would then be chosen to duplicate the values of pre-exponential and activation energy given by Müller et al. (1992). See Table 3.3 for the simplified four-step rate parameters with continuous thermodynamics.

TABLE 3.3: Four-step rate parameters **with** continuous thermodynamics

	Pre-exponential		**Activation Energy**	
	$K(I) = \exp(a_K + b_K I)$		$E(I) = a_E + b_E I$	
Refer to: Eq. 3.57-3.58	$K_{i_{EFF}} = \frac{\exp(a_{K_i}+b_{K_i}\theta)}{(\beta A_{2_i})^{\alpha}} \exp(-\theta b_{K_i} + \frac{b_{E_i}\alpha}{A_{2_i}\bar{R}T})$		$E_{i_{EFF}} = a_{E_i} + b_{E_i}\alpha / A_{2_i}$	
Reactions	a_{K_i}	b_{K_i}	a_{E_i}	b_{E_i}
R1 $i = 1$	19.972	0.03048	157.252	0.2270
R3$_F$ $i = 3_F$	32.583	0.03048	157.252	0.2270

Note: For the other reactions **R2**, **R3**$_B$ and **R4**, the pre-exponential and activation energy are the same as those in Müller et al. (1992).

3.4.2.4 Reverse Reaction R3$_B$

Since the reverse reaction produces fuel, a positive source term needs to be accounted for in the transport equations. For a single fuel species i, the basic conservation equation is

$$\frac{\partial}{\partial t}(cX_i) + \nabla \cdot (cv^* X_i) = R_{1_i} + R_{3F_i} + R_{3B_i} \tag{3.107}$$

where the diffusion term has been omitted since we are dealing with a homogeneous mixture. Substituting $X_i = X_{\mathbf{F}} f(I) dI$:

$$\begin{aligned}\frac{\partial}{\partial t}(cX_{\mathbf{F}} f(I)dI) + \nabla \cdot (cv^* X_{\mathbf{F}} f(I)dI) = {} & A_1 cX_{\mathbf{F}} f(I)dI \exp(-E_1/\bar{R}T) \quad + \dots \\ & \dots + A_{3f} c^2 X_O X_{\mathbf{F}} f(I)dI \exp(-E_{3f}/\bar{R}T) \quad + \dots \\ & \dots + A_{3b} cX_{\mathbf{I}} f\,'(I)dI \exp(-E_{3b}/\bar{R}T)\end{aligned} \tag{3.108}$$

The **Reaction R3**$_B$ has been given a different distribution function $f\,'(I)$: since intermediate $\mathbf{I}$ is produced and consumed by different reactions than those affecting the fuel, we do not know that it has the same distribution. Integrating over I from 0 to ∞ gives

$$\frac{\partial}{\partial t}(cX_{\mathbf{F}}) + \nabla \cdot (cv^* X_{\mathbf{F}}) = \bar{W}_1 + \bar{W}_{3F} + \bar{W}_{3B} \tag{3.109}$$

The first two rates are given by the same sort of expression as in the single- and two-step models:

$$\bar{W}_1 = -cX_{\mathbf{F}} K_{1_{EFF}} \exp(-E_{1_{EFF}}/\bar{R}T) \tag{3.110}$$

where

$$E_{1_{EFF}} = a_{E_1} + b_{E_1}\alpha/A_{2_1} + \gamma b_{E_1} \tag{3.111}$$

$$K_{1_{EFF}} = \kappa \exp(a_{K_1} + b_{K_1}\theta)(\beta A_{2_1})^{-\alpha} \exp\left(-(\theta - \gamma) b_{K_1} + \frac{b_{E_1}\alpha}{A_{2_1}\bar{R}T}\right) \tag{3.112}$$

and **Reaction R3**$_F$

$$\bar{W}_{3F} = -c^2 X_{\mathbf{F}} X_O K_{3F_{EFF}} \exp(-E_{3F_{EFF}}/\bar{R}T) \tag{3.113}$$

with similar expressions for K_{EFF} and E_{EFF}. Note that the A_2 variables here are the expressions defined in the continuous thermodynamics model (Eq. 3.31 above) and are not the same as the pre-exponential factors given by Müller et al. (1992).

For the reverse **Reaction R3**$_B$,

$$\bar{W}_{3B} = +cX_{\mathbf{I}} K_{3B_{EFF}} \exp(-E_{3B_{EFF}}/\bar{R}T) \tag{3.114}$$

Note the change in sign. This reaction is producing fuel, so it must be positive. The effective rate parameters are

$$E_{3B_{EFF}} = a_{E_{3B}} + b_{E_{3B}}\alpha\,'/A_{2_{3B}} + \gamma b_{E_{3B}} \tag{3.115}$$

$$K_{3B_{EFF}} = \kappa \exp(a_{K_{3B}} + b_{K_{3B}}\theta\,')(\beta\,' A_{2_{3B}})^{-\alpha\,'} \exp\left(-(\theta\,' - \gamma\,')b_{K_{3B}} + \frac{b_{E_{3B}}\alpha\,'}{A_{2_{3B}}\bar{R}T}\right) \tag{3.116}$$

where the primes ($'$) refer to the distribution function $f\,'(I)$ for intermediate **I**.

Multiplying by weighting factors of I and I^2 results in conservation equations for the fuel parameters:

$$\frac{\partial}{\partial t}(cX_{\mathbf{F}}\theta) + \nabla \cdot (cv^* X_{\mathbf{F}}\theta) = \widetilde{W}_1 + \widetilde{W}_{3F} + \widetilde{W}_{3B} \tag{3.117}$$

$$\frac{\partial}{\partial t}(cX_{\mathbf{F}}\Psi) + \nabla \cdot (cv^* X_{\mathbf{F}}\Psi) = \widehat{W}_1 + \widehat{W}_{3F} + \widehat{W}_{3B} \tag{3.118}$$

where

$$\widetilde{W}_1 = \bar{W}_1\left(\frac{\alpha}{A_{2_1}} + \gamma\right) \tag{3.119}$$

$$\widetilde{W}_{3F} = \bar{W}_{3F}\left(\frac{\alpha}{A_{2_{3F}}} + \gamma\right) \tag{3.120}$$

$$\widetilde{W}_{3B} = \bar{W}_{3B}\left(\frac{\alpha\,'}{A_{2_{3B}}} + \gamma\,'\right) \tag{3.121}$$

and

$$\widehat{W}_1 = \bar{W}_1\left(\frac{\alpha(\alpha+1)}{A^2_{2_1}} + \frac{2\gamma\alpha}{A_{2_1}} + \gamma^2\right) \tag{3.122}$$

$$\widehat{W}_{3F} = \bar{W}_{3F}\left(\frac{\alpha(\alpha+1)}{A^2_{2_{3F}}} + \frac{2\gamma\alpha}{A_{2_{3F}}} + \gamma^2\right) \tag{3.123}$$

$$\widehat{W}_{3B} = \bar{W}_{3B}\left(\frac{\alpha\,'(\alpha\,'+1)}{A^2_{2_{3B}}} + \frac{2\gamma\,'\alpha\,'}{A_{2_{3B}}} + \gamma\,'^{\,2}\right) \tag{3.124}$$

Note that the $\bar{W}_1$ and $\bar{W}_{3F}$ terms will be negative, since fuel is consumed in **Reaction R1** and **Reaction 3F**, but $\bar{W}_{3B}$ will be positive, since **Reaction 3B** is producing fuel.

3.4.2.5 Reverse Reaction - Simplified

Two simplifications will be made for **Reaction 3B**. First, we have no means of determining the dependence of the rate of the reverse reaction on the distribution variable I, so we shall assume that $b_{K_{3B}} = 0$ and $b_{E_{3B}} = 0$. Secondly, as in all previous work, we shall assume that the distribution origin $\gamma = 0$. These reduce the equations to the following:

$$A_{2_{3B}} = \frac{1}{\beta\,'} \tag{3.125}$$

$$E_{3B_{EFF}} = a_{E_{3B}} \tag{3.126}$$

$$K_{3B_{EFF}} = \kappa \exp(a_{K_{3B}}) \tag{3.127}$$

$$\widetilde{W}_{3B} = \bar{W}_{3B}\left(\frac{\alpha\,'}{A_{2_{3B}}}\right) = \bar{W}_{3B}(\alpha\,'\beta\,') \tag{3.128}$$

$$\widehat{W}_{3B} = \bar{W}_{3B}\left(\frac{\alpha\,'(\alpha\,'+1)}{A_{2_{3B}}^2}\right) = \bar{W}_{3B}(\alpha\,'(\alpha\,'+1)\beta\,'^{\,2}) \tag{3.129}$$

Note that the effective E and K are simply those for the rate of this reaction for a pure fuel - in other words, the values from the Müller et al. (1992) paper.

3.4.2.6 Reverse Reaction - Finding the parameters of $f\,'(I)$

The distribution function for intermediate **I** is found from a balance on **I**, derived using the same steps as for fuel:

$$\frac{\partial}{\partial t}(cX_{\mathbf{I}}) + \nabla \cdot (cv^{*}X_{\mathbf{I}}) = \bar{W}_4 - \bar{W}_{3F} - \bar{W}_{3B} \tag{3.130}$$

Note that the signs for $\bar{W}_{3F}$ and $\bar{W}_{3B}$ have changed, because the balance equation is now for the intermediate **I** rather than the fuel **F**. The rate for **Reaction R4** is

$$\bar{W}_4 = -c^2 X_{\mathbf{I}} X_O K_{4_{EFF}} \exp(-E_{4_{EFF}}/\bar{R}T) \tag{3.131}$$

with a negative sign because **Reaction R4** consumes **I**. The rate parameters are

$$A_{2_4} = \frac{1}{\beta\,'} \tag{3.132}$$

$$E_{4_{EFF}} = a_{E_4} \tag{3.133}$$

$$K_{4_{EFF}} = \kappa \exp(a_{K_4}) \tag{3.134}$$

Again, the effective rate parameters are simply those for a pure fuel - in other words, the values from the Müller et al. (1992) paper.

The equations for the distribution parameters are

$$\frac{\partial}{\partial t}(cX_{\mathbf{I}}\theta\,') + \nabla \cdot (cv^* X_{\mathbf{I}}\theta\,') = \widetilde{W}_4 - \widetilde{W}_{3F} - \widetilde{W}_{3B} \tag{3.135}$$

$$\frac{\partial}{\partial t}(cX_{\mathbf{I}}\Psi\,') + \nabla \cdot (cv^* X_{\mathbf{I}}\Psi\,') = \widehat{W}_4 - \widehat{W}_{3F} - \widehat{W}_{3B} \tag{3.136}$$

$$\widetilde{W}_4 = \bar{W}_4\left(\frac{\alpha\,'}{A_{2_4}}\right) = \bar{W}_4(\alpha\,'\beta\,') \tag{3.137}$$

$$\widehat{W}_4 = \bar{W}_4\left(\frac{\alpha\,'(\alpha\,'+1)}{A_{2_4}^2}\right) = \bar{W}_4(\alpha\,'(\alpha\,'+1)\beta\,'^{\,2}) \tag{3.138}$$

The transport equations for $\theta\,'$ and $\Psi\,'$ are stepped forward in time just as those for θ and Ψ are. For initial values one should assume $\theta\,' = \theta$ and $\Psi\,' = \Psi$.

In summary, comparing the continuous thermodynamics model to the Müller et al. (1992) model for pure n-heptane, the reaction rates $\bar{W}_{3B}$ and $\bar{W}_4$ are identical, but the rates $\widetilde{W}_{3B}$, $\widetilde{W}_4$, $\widehat{W}_{3B}$ and $\widehat{W}_4$ are modified by factors from the distribution function $f\,'(I)$.

3.5 Thermodynamic Properties - NASA Polynomials

To solve the conservation equations (Eq. 3.61-3.65), thermodynamic properties are required. Since homogeneous mixture ignition is assumed, transport properties such as diffusivities are not required. For thermodynamic properties, the 7-coefficient NASA polynomial parameterization is used by Cantera, where seven coefficients ($\breve{a}_0$ to $\breve{a}_6$) can be specified for one or more temperature ranges to calculate specific heat $\bar{c}_p$, enthalpy $\bar{h}$ and entropy $\bar{s}$ in the following forms:

$$\frac{\bar{c}_p(T)}{\bar{R}} = \breve{a}_0 + \breve{a}_1 T + \breve{a}_2 T^2 + \breve{a}_3 T^3 + \breve{a}_4 T^4 \tag{3.139}$$

$$\frac{\bar{h}(T)}{\bar{R}T} = \breve{a}_0 + \frac{\breve{a}_1}{2} T + \frac{\breve{a}_2}{3} T^2 + \frac{\breve{a}_3}{4} T^3 + \frac{\breve{a}_4}{5} T^4 + \frac{\breve{a}_5}{T} \tag{3.140}$$

$$\frac{\bar{s}(T)}{\bar{R}} = \breve{a}_0 \ln T + \breve{a}_1 T + \frac{\breve{a}_2}{2} T^2 + \frac{\breve{a}_3}{3} T^3 + \frac{\breve{a}_4}{4} T^4 + \breve{a}_6 \tag{3.141}$$

3.5.1 Continuous Thermodynamics Specific Heat

The form for fuel vapour specific heat used in the previous continuous thermodynamics models is slightly modified from the NASA parameterization, where $\breve{a}_0$ becomes $(a_0 + b_0\theta)$, and so on:

$$\frac{\bar{c}_p(T,\theta)}{\bar{R}} = (a_0 + b_0\theta) + (a_1 + b_1\theta)T + (a_2 + b_2\theta)T^2 + (a_3 + b_3\theta)T^3 \tag{3.142}$$

where T is in K, and θ is the distribution mean in $\mathrm{kg/kmol}$. Note that there is no T^4 term present as in Eq. 3.139 because a_4 and b_4 are both zero (Sabourin et al. 2016). The curve-fitted coefficients are for a temperature range of 200-1100 K, and are listed in Table 3.4. These are slightly different from the ones given by Sabourin et al. (2016), because they were fitted by Hallett (2017) for the carbon number range from $C_7 \rightarrow C_{16}$, whereas the former were fitted for $C_7 \rightarrow C_{20}$.

TABLE 3.4: Continuous Thermodynamic NASA Coefficients for Fuel Specific Heat

Coefficient	$i=0$	$i=1$	$i=2$	$i=3$	$i=4$
a_i	2.473e-01	1.092e-02	-3.124e-06	-3.757e-11	0
b_i	-8.328e-03	8.704e-04	-4.984e-07	1.090e-10	0

(Hallett 2017)

These expressions can then be substituted for the NASA polynomial coefficients, with $(a_0 + b_0\theta)$ becoming the NASA $\breve{a}_0$, etc, and updated as temperature T and the distribution mean θ change.

3.5.2 Continuous Thermodynamics Enthalpy

The relation between enthalpy and specific heat for an ideal gas is $dh = c_p dT$. The enthalpy is calculated by integrating the specific heat from Eq. 3.139, with $(a_0 + b_0\theta)$

becoming the NASA $\breve{a}_0$, etc:

$$\frac{\bar{h}}{\bar{R}T} = \frac{1}{T}\left[\int_0^T \frac{\bar{c}_p}{\bar{R}} dT + (a_5 + b_5\theta)\right] = \dots \tag{3.143}$$

$$\dots = (a_0 + b_0\theta) + \frac{(a_1 + b_1\theta)}{2}T + \frac{(a_2 + b_2\theta)}{3}T^2 + \frac{(a_3 + b_3\theta)}{4}T^3 + \frac{(a_5 + b_5\theta)}{T}$$

where $(a_5+b_5\theta)$ accounts for the enthalpy of formation $\bar{h}_f^\circ$. By convention, the sensible heat part of the enthalpy (i.e. the $\int c_p dT$ part) is set to zero at the reference temperature of $T_r = 298.15$ K, which means that at the reference temperature the enthalpy $\bar{h}$ must equal the enthalpy of formation $\bar{h}_f^\circ$:

$$\frac{\bar{h}}{\bar{R}T_r} = \frac{\bar{h}_f^\circ}{\bar{R}T_r} \tag{3.144}$$

This fact is used to set the value of $(a_5 + b_5\theta)$:

$$\frac{(a_5 + b_5\theta)}{T_r} = \dots \tag{3.145}$$

$$\dots = \frac{\bar{h}_f^\circ}{\bar{R}T_r} - \left[(a_0 + b_0\theta) + \frac{(a_1 + b_1\theta)}{2}T_r + \frac{(a_2 + b_2\theta)}{3}T_r^2 + \frac{(a_3 + b_3\theta)}{4}T_r^3\right]$$

For a continuous thermodynamics formulation, $\bar{h}_f^\circ$ is a function of molecular mass and therefore the expression contains a θ term. Values of $\bar{h}_f^\circ$ for a number of n-paraffins are listed in Table 3.5, and are used with the fuel coefficients (a_0, b_0) to (a_3, b_3) from Table 3.4 to solve for $(a_5 + b_5\theta)$.

TABLE 3.5: Continuous Thermodynamic NASA Coefficients for Enthalpy

Formula	**Name**	θ_i kg/kmol	$\bar{h}_f^\circ$ kJ/kmol	$(a_5 + b_5\theta)$
C_7H_{16}	n-heptane	100.204	-187800	-26327
C_8H_{18}	n-octane	114.231	-208750	-29295
C_9H_{20}	n-nonane	128.258	-228860	-32163
$C_{10}H_{22}$	n-decane	142.285	-249530	-35098
$C_{11}H_{24}$	n-undecane	156.312	-270160	-38028
$C_{12}H_{26}$	n-dodecane	170.338	-290790	-40959
$C_{13}H_{28}$	n-tridecane	184.365	-311420	-43889
$C_{14}H_{30}$	n-tetradecane	198.392	-332050	-46819
$C_{15}H_{32}$	n-pentadecane	212.419	-352680	-49750
$C_{16}H_{34}$	n-hexadecane	226.446	-373310	-52680

Enthalpy of formation (Poling et al. 2001)

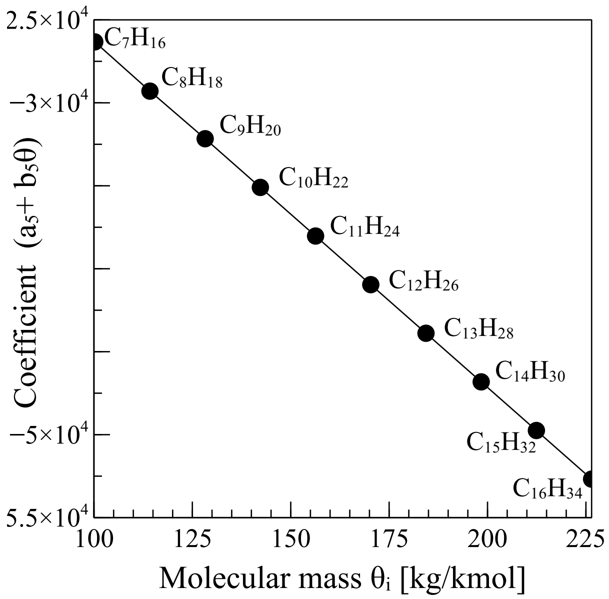


FIGURE 3.2: NASA Polynomial Fuel Coefficients $(a_5 + b_5\theta)$ vs. Molecular Mass θ

This is fitted as the equation (Figure. 3.2):

$$y(\theta) = (a_5 + b_5\theta) = -5420.438 - 208.666(\theta) \tag{3.146}$$

3.5.3 Continuous Thermodynamics Entropy

Although the entropy is not used directly for the present calculations, it must be furnished from Eq. 3.141 as an input to Cantera. The form for vapour entropy used in the continuous thermodynamics model from is

$$\frac{\bar{s}}{\bar{R}} = \left[\int_0^T \frac{\bar{c}_p}{\bar{R}T} dT + (a_6 + b_6\theta) \right] = \dots \tag{3.147}$$

$$\dots = (a_0 + b_0\theta)\ln T + (a_1 + b_1\theta)T + \frac{(a_2 + b_2\theta)}{2}T^2 + \frac{(a_3 + b_3\theta)}{3}T^3 + (a_6 + b_6\theta)$$

In a similar fashion as Eq. 3.145, $(a_6 + b_6\theta)$ can be solved at the reference temperature of $T_r = 298.15$ K:

$$(a_6 + b_6\theta) = \dots \tag{3.148}$$

$$\dots = \frac{\bar{s}^\circ}{\bar{R}} - \left[(a_0 + b_0\theta)\ln T_r + (a_1 + b_1\theta)T_r + \frac{(a_2 + b_2\theta)}{2}T_r^2 + \frac{(a_3 + b_3\theta)}{3}T_r^3 \right]$$

The fuel coefficients (a_0, b_0) to (a_3, b_3) from Table 3.4, and the standard entropy $\bar{s}^\circ$ for each fuel are required; however, for consistency the same sources used for enthalpy $\bar{h}_f^\circ$ in Table 3.5 should be used, and the source (Poling et al. 2001) only lists the Gibbs formation function $\bar{g}_f^\circ$. The relationship between the fuel Gibbs energy of formation and the fuel standard entropy is:

$$\bar{g}_{f,i}^\circ(T_r) = \bar{h}_{f,i}^\circ(T_r) - T_r\bar{s}_i^\circ(T_r) - \sum_j \nu_j'[-T_r\bar{s}_j^\circ(T_r)] \tag{3.149}$$

The summation combines the elements j in their natural states with the appropriate stoichiometric coefficients ν_j' from the reaction which forms the fuel from its elements. In addition, the summation includes all species in the reaction of formation. Here j represents the fuel elements or species in their natural state - i.e. C, H_2, and ν_j' is the number of moles for the stoichiometric equation for the formation of the compound from those natural states respectively.

Specifically for an n-paraffin hydrocarbon fuel, $\bar{s}^\circ(T_r)$ is found from

$$\bar{s}_{fuel}^\circ(T_r) = \frac{\bar{g}_{f,fuel}^\circ(T_r) - \bar{h}_{f,fuel}^\circ(T_r)}{T_r} + \nu_C'\bar{s}_C^\circ + \nu_{H_2}'\bar{s}_{H_2}^\circ \tag{3.150}$$

TABLE 3.6: Species Absolute Entropy at Standard Conditions

Formula	**Name**	$\bar{s}_j^\circ(T_r)$ kJ/kmol K
C	Carbon (graphite)	5.740
H_2	Hydrogen	130.678

(Sonntag et al. 2002)

The Gibbs function $\bar{g}_f^\circ$, entropy $\bar{s}^\circ$ and the coefficient $(a_6 + b_6\theta)$ are tabulated in Table 3.7.

TABLE 3.7: Continuous Thermodynamic NASA Coefficients for Entropy

Formula	**Name**	$\bar{g}_f^\circ$ kJ/kmol	$\bar{s}^\circ$ kJ/kmol K	$(a_6 + b_6\theta)$
C_7H_{16}	n-heptane	8200	428	27.9
C_8H_{18}	n-octane	16270	467	29.9
C_9H_{20}	n-nonane	25000	507	31.9
$C_{10}H_{22}$	n-decane	33300	546	34.1
$C_{11}H_{24}$	n-undecane	41250	587	36.2
$C_{12}H_{26}$	n-dodecane	49530	626	38.3
$C_{13}H_{28}$	n-tridecane	57810	666	40.4
$C_{14}H_{30}$	n-tetradecane	66090	705	42.4
$C_{15}H_{32}$	n-pentadecane	74370	745	44.5
$C_{16}H_{34}$	n-hexadecane	82650	784	46.6

Gibbs function of formation (Poling et al. 2001), Entropy calculated from Gibbs

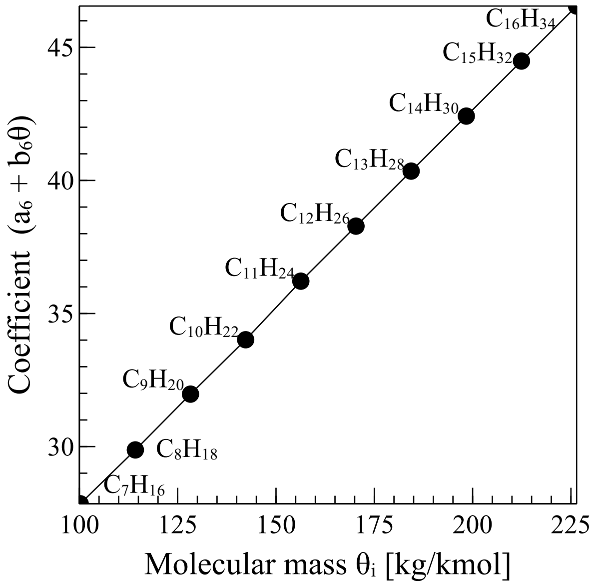


FIGURE 3.3: NASA Polynomial Fuel Coefficients $(a_6 + b_6\theta)$ vs. Molecular Mass θ

This is fitted as the equation (Figure. 3.3):

$$y(\theta) = (a_6 + b_6\theta) = 12.931 + 0.149(\theta) \tag{3.151}$$

TABLE 3.8: Summary of Continuous Thermodynamic NASA Coefficients for the fuel

Coefficient	$i = 0$	$i = 1$	$i = 2$	$i = 3$	$i = 4$	$i = 5$	$i = 6$
a_i	2.473e-01	1.092e-02	-3.124e-06	-3.757e-11	0	-5420.438	12.931
b_i	-8.328e-03	8.704e-04	-4.984e-07	1.090e-10	0	-208.666	0.149

(Hallett 2017)

3.5.4 Thermal Properties for Intermediate Species

The four-step model has intermediate species, therefore NASA polynomials for the two intermediates **X** and **I** are required. For an ideal gas, the enthalpy, entropy and specific heats are linear combinations of the individual species values by mole fraction. One “mol” of intermediate **X** contains ($3\ C_2H_4 + CH_3 + H$), which has a total of 5 mols of species. The specific heat of one “mol” of **X** is therefore

$$c_{p\ \mathbf{X}} = 3\ c_{p\ C_2H_4} + c_{p\ CH_3} + c_{p\ H} \tag{3.152}$$

For the NASA polynomials, this means that the individual coefficient terms will also be combined by mol fraction. For example,

$$a_{1\ \mathbf{X}} = 3\ a_{1\ C_2H_4} + a_{1\ CH_3} + a_{1\ H} \tag{3.153}$$

where the NASA polynomials for C_2H_4, CH_3 and H are already available in Cantera.

Similarly for the other intermediate, one "mol" of **I** contains ($HO_2R''O + H_2O$), which has a total of 2 mols of species. The specific heat of one "mol" of **I** is therefore

$$c_{p\ \mathbf{I}} = c_{p\ HO_2R''O} + c_{p\ H_2O} \tag{3.154}$$

For the NASA polynomials, the individual coefficient terms are also combined by mol fraction:

$$a_{1\ \mathbf{I}} = a_{1\ HO_2R''O} + a_{1\ H_2O} \tag{3.155}$$

The NASA polynomials for H_2O are already available in Cantera; however, the NASA polynomials for $HO_2R''O$ are from Stagni et al. (2016).

Chapter 4

Methods and Procedures

4.1 Introduction

As discussed in Section 3.3.6, the simplest test case that can be applied to the continuous thermodynamics theory is the perfectly mixed constant pressure reactor, for which the diffusion term can be dropped from the conservation equations. The task in this chapter is to present the methods and procedures to solve these equations (Eq. 3.61-3.63) using the chemical kinetics and thermodynamics solver called Cantera.

4.2 Cantera

Cantera is an object-oriented open source software designed to be used as a solver or tool kit with a programming language such as Python, Matlab, C++, or FORTRAN 90, for the purpose of solving chemical kinetics, thermodynamics and transport processes. The Cantera tools, solvers and functions are in the form of C++ libraries that can be called through functions via a script written in an object-oriented programming language (i.e. Python) to assist with modelling and solving many combustion problems.

Cantera can be installed on Windows, Mac OS, and Linux platforms such as ubuntu. This research was performed using Cantera 2.3.0, Python 2.7.12 and ubuntu 16.04 LTS, installed on a Lenovo ThinkPad Edge with a 64-bit, Intel® Core^{TM} i5-3230M CPU @ 2.60GHz x 4 cores and 4GB RAM.

This chapter focuses on applying Cantera with the Python programming language for the topic of this thesis. It is not meant to be an authority on how Cantera works or a user guide, and will only provide enough information to describe this work. For a complete user guide and Cantera information see Goodwin et al. (2017).

4.3 Model criteria

One of the main objectives of this thesis is to show that Cantera can be used for modelling simplified reaction mechanisms using continuous thermodynamics.

What is unique about our continuous thermodynamic problem is that the rate constants and fuel properties can change. The nature of continuous thermodynamics requires updating multiple chemical kinetic, thermodynamic, and transport process variables that are a function of the distribution mean (molecular mass) θ and standard deviation σ, both of which may change as the reaction progresses and the composition of the fuel mixture changes.

To achieve our objective it was decided to select a test case (homogeneous ignition) that avoids modifying transport properties, so as to focus on the thermodynamics and chemical kinetics. A suitable Cantera function is `IdealGasConstPressureReactor`, which is a homogeneous, constant pressure, zero-dimensional reactor for ideal gas mixtures. The volume of the reactor changes as a function of time in order to keep the pressure constant. In other words, this is a calculation for constant pressure auto-ignition of a premixed mixture. Since this type of reactor is homogeneous, there are no gradients of properties and therefore no transport properties.

Therefore the main objective for this model is to use Cantera with a reduced scheme for a continuous thermodynamic mixture. It was determined that for a systematic approach to modelling a reduced scheme, the best course of action would be to start with a single-step mechanism, analyze the results, and then incrementally move to two- and four-step mechanisms. The purpose is to find a simple kinetic scheme that will still reproduce the essential phenomena, such as two-stage ignition.

The fuels of interest are mixtures of volatile liquid n-paraffin fuels such as C_nH_{2n+2}. The tabulated fuel formula, molecular mass and autoignition temperatures are useful for modelling, Table 4.1. For example, in Cantera a mixture of pure fuel and air is imported, and the initial pressure, temperature and stoichiometry are set. However, if the initial temperature is too low, the reaction becomes so slow that ignition does not occur within a reasonable length of time.

TABLE 4.1: Auto Ignition Temperatures for n-paraffin Fuels, (Zabetakis et al. 1954)

Formula	**Name**	**Molecular mass M_i or θ_i $\mathrm{kg/kmol}$**	**Auto ignition temp K**
C_7H_{16}	n-heptane	100.204	496
C_8H_{18}	n-octane	114.231	493
C_9H_{20}	n-nonane	128.258	479
$C_{10}H_{22}$	n-decane	142.285	481
$C_{11}H_{24}$	n-undecane	156.312	-
$C_{12}H_{26}$	n-dodecane	170.338	477
$C_{13}H_{28}$	n-tridecane	184.365	-
$C_{14}H_{30}$	n-tetradecane	198.392	475*
$C_{15}H_{32}$	n-pentadecane	212.419	-
$C_{16}H_{34}$	n-hexadecane	226.446	478

*(Egolf and Jurs 1992)

4.4 Working with Cantera

This section is a brief summary of the procedures used to compile a Cantera constant pressure reactor program in Python. See Appendix A for a detailed discussion, including Python scripts and example codes using Cantera.

4.4.1 Python and Cantera

Python is an open source object-oriented programming language. It is also one of the most documented and supported programing languages for Cantera, and has therefore been selected for the continuous thermodynamics model in this thesis.

There are four essential operations to run a constant pressure reactor with Python and Cantera:

- Import Cantera as a Python module
- Input gas phase and set initial conditions

- Define a reactor containing the gas phase
- Print the states of the gas as it advances in time through the reactor simulation

The thermodynamic states of the gas (T, $X_{\mathbf{F}}$, X_O, ..., etc.) are obtained at each internal time-step from the Cantera solver, and are used to update the continuous thermodynamic rate parameters for the next time-step. Each update will change how Cantera calculates the properties for the next time-step, which is the essential feature of the model.

4.4.2 Cantera input file

Along with the Python program which contains the function call of Cantera and the numerical calculations to solve the continuous thermodynamics model, an additional chemistry file called a Cantera input file is required. The Cantera input file is made up of chemical phases, species, and reaction data. The phases section contains the chemical elements and the states (solid, liquid, gas, surface interfaces, and multi-states) and the chemical species that are composed of those elements. The thermodynamic properties of each species present are represented by the NASA polynomial parameterization and the final section of the input file contains all the required reaction equations and the associated rates.

4.5 Numerical Solution for Constant Pressure Test Case

The calculation sequence looks like this:

1. Set up initial conditions.

2. Using the initial values of distribution mean θ (in this case initial molecular weight of fuel) and a chosen standard deviation σ, get the initial values of distribution parameters α and β, (Eq. 3.7, 3.8). Determine effective rate parameters E_{EFF} and K_{EFF}, (Eq. 3.57, 3.58). Use these as the rate parameter inputs to Cantera.

3. To solve the transport equations (Eq. 3.61-3.63), run the Cantera constant pressure reactor model for the first time-step δt, using these parameters. From Cantera get the rate $\bar{W}$ and the new value of fuel mol fraction $X_{\mathbf{F}}$ at the end of δt, as well as the new temperature T, molar density c, and the new mixture molecular mass M. After this, the new distribution parameters are calculated as in the next two steps.

4. From Eq. 3.71 the expansion term for a small time-step is calculated as

   $$\nabla \cdot (cv^*)\delta t = -\frac{c}{M}\delta M + \frac{c}{T}\delta T \tag{4.1}$$

   The c, M and T values can be either old ones (explicit method) or new ones (implicit). Both are available from the output information from Cantera, and were tested with very little difference. The explicit method was chosen in the end for this work.

5. From Eq. 3.74 and 3.75, update the values of $(cX_{\mathbf{F}}\theta)$ and $(cX_{\mathbf{F}}\Psi)$ as follows:

   $$(cX_{\mathbf{F}}\theta)_{t+\delta t} = (cX_{\mathbf{F}}\theta)_t + \widetilde{W}\delta t - (X_{\mathbf{F}}\theta)_t \nabla \cdot (cv^*)\delta t \tag{4.2}$$

   $$(cX_{\mathbf{F}}\Psi)_{t+\delta t} = (cX_{\mathbf{F}}\Psi)_t + \widehat{W}\delta t - (X_{\mathbf{F}}\Psi)_t \nabla \cdot (cv^*)\delta t \tag{4.3}$$

then divide each of these by $(cX_{\mathbf{F}})_{t+\delta t}$ to get the new θ and Ψ. There is no need to calculate a new $X_{\mathbf{F}}$ because Cantera has already done this. Note that $(cX_{\mathbf{F}})$ have been kept together in this formulation; this is because the transport equations are written for conservation of $(cX_{\mathbf{F}})$ rather than $X_{\mathbf{F}}$.

6. Revise the effective rate parameters E_{EFF} and K_{EFF} and the stoichiometric coefficients for the next time step.
7. Call Cantera for the next time step (i.e. return to step 2). Keep going until ignition is achieved.

The detailed process is illustrated in a flow chart for the single-step continuous thermodynamics reactor model in Figure 4.1, and the associated code in Listing C.1 can be found in Appendix C.

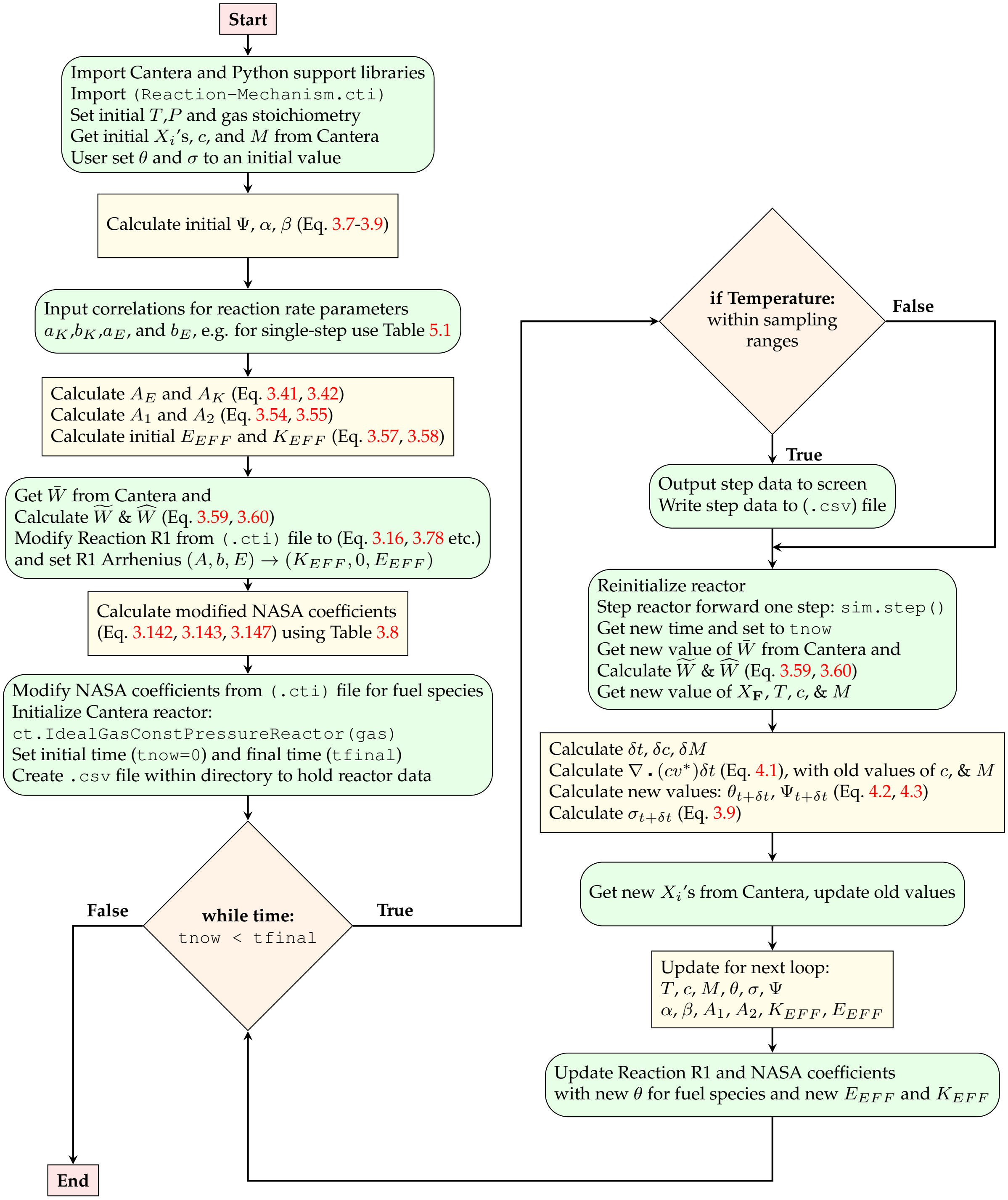


FIGURE 4.1: Continuous Thermodynamics Reactor Model Flow chart

4.6 Model Testing

To test the model, a reactor program was run with a pure fuel without continuous thermodynamics and the results were compared to a distribution with a small σ and the appropriate continuous thermodynamic calculations as in Figure 4.1. As stated earlier, a small σ should provide similar results to a pure fuel. The continuous thermodynamics model was then run with increasing values of σ in order to determine the effect of σ. These steps were followed:

1. N-heptane was selected as a pure fuel, and the Arrhenius rate constants (A, b, E) were set as $(K_{EFF}, 0, E_{EFF})$ where the effective activation energy E_{EFF}, and the effective pre-exponential coefficient K_{EFF}, were calculated from Eq. 3.57 and 3.58 respectively. These rate constants were entered into the Cantera input file `(.cti)` as in Appendix B.1 and the reactor program (Listing C.2) without continuous thermodynamics was run. For this case, $(K_{EFF}, 0, E_{EFF})$ will remain constant throughout the calculation since they are not updated in the program.

2. N-heptane was modelled as a continuous distribution with $\theta = \textit{fuel mol mass}$ and with a small standard deviation $\sigma = 2$ kg/kmol. The reactor program in Listing C.1 (Appendix C) was run, this time with the appropriate continuous thermodynamic calculations and revision of K_{EFF} and E_{EFF} after each time-step. The constants should change very little, since a narrow distribution will have very little change in θ or σ with time, and should give the same results as the pure fuel.

3. The model was run again with the same mean θ and increasing standard deviations σ. As σ increases, there should be changes in the ignition time.

Chapter 5

Results and Discussion

5.1 Introduction

This chapter presents the results and discussions for the single-, two- and four-step reaction mechanisms in continuous thermodynamics form, applied to the simple test case of a constant pressure homogeneous reactor. For each mechanism, the non-continuous models are compared to the continuous ones in terms of ignition delay, adiabatic flame temperature, chemical species mol fraction development and changes in the mean and standard deviation during ignition.

5.2 Simulations with Single-step Mechanisms

5.2.1 Single-step Reaction Without Continuous Thermodynamics

A detailed chemical kinetic reaction mechanism for n-heptane from Mehl et al. (2011) was selected as a benchmark with which to compare our results. It was written in the Chemkin format (`.ck`), and converted to a Cantera input file (`.cti`). The detailed

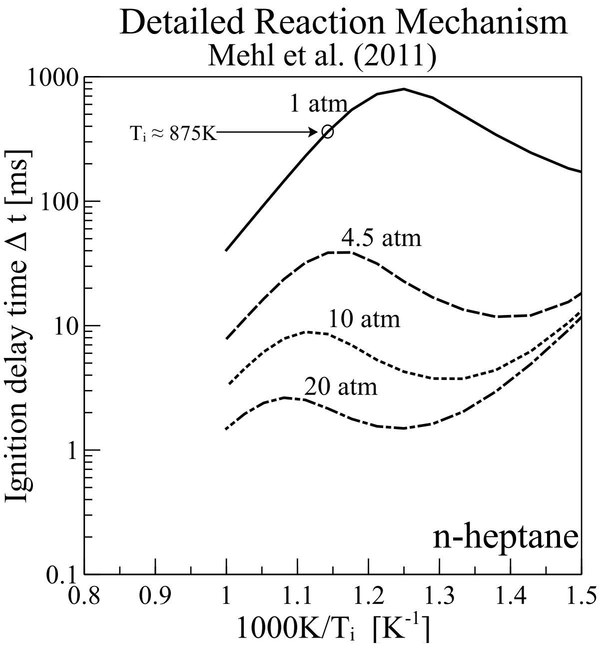


FIGURE 5.1: Ignition delay times as a function of $(1/T_i)$ for multiple pressures using the detailed mechanism for n-heptane from Mehl et al. (2011). Curves are detailed reaction model predictions using Cantera.

mechanism was modelled in a Cantera constant pressure ideal gas reactor at multiple pressures, and the corresponding ignition delay times were plotted in Figure 5.1.

The reaction rate parameters a_K and a_E used in Eq. 3.57 and 3.58 were adjusted in order to fit the ignition delay times of the single-step reaction mechanism without continuous thermodynamics to the detailed reaction mechanism (Fig. 5.2). The ignition delay time as a function of $(1/T_i)$ for the single-step reaction mechanism produces a straight line. Modification to a_K adjusts the vertical position of the line, while modification to a_E adjusts the slope. Initial values for slope and vertical positions were matched from the straight portion of the detailed mechanism. The parameters b_K and b_E were kept constant at the values from Sabourin et al. (2016) (see Table 3.1); in other words, the dependence of rate on fuel molecular mass was assumed the same. The

TABLE 5.1: Single-step fitted correlations for reaction rate parameters n-paraffins. a_K and a_E have been fitted to the detailed reaction mechanism in Figure 5.2.

	a_K	b_K*	a_E	b_E*
n-paraffins	21.943	0.03048	137.315	0.2270

b_K* and b_E* from Sabourin et al. (2016) as in Table 3.1

final rate parameters for the single-step mechanism are shown in Table 5.1.

The reaction equation **R1**, and the associated reaction rate for single-step n-heptane oxidation without continuous thermodynamic correlations become

(Reaction: **R1**) $$C_7H_{16} + 11\ O_2 \rightarrow 7\ CO_2 + 8\ H_2O \tag{5.1}$$

(Rate: **R1**) $$K_{R1} = 7.181 \times 10^{10} [C_7H_{16}]^1 [O2]^1 \exp(-19\,251/T) \tag{5.2}$$

where the pre-exponential coefficient and the activation energy are found using the rate parameters in Table 5.1 with Eq. 3.25 and 3.26, and the molecular mass of n-heptane.

The rate is written here with concentrations in $\mathrm{kmol/m^3}$, pre-exponential factor in $[\mathrm{kmol/m^3}]^{1\,-\,\mathrm{m}\,-\,\mathrm{n}}/\mathrm{s}$, where m, n are the concentration exponents, and the activation energy is expressed in K.

The fitted single-step reaction mechanism <u>without</u> continuous thermodynamics is shown in Fig. 5.2 and agrees very well with the published detailed mechanism for initial temperatures $T_i > 875$ K. Below this temperature the detailed mechanism enters the negative temperature coefficient region, which of course a single step reaction is incapable of modelling.

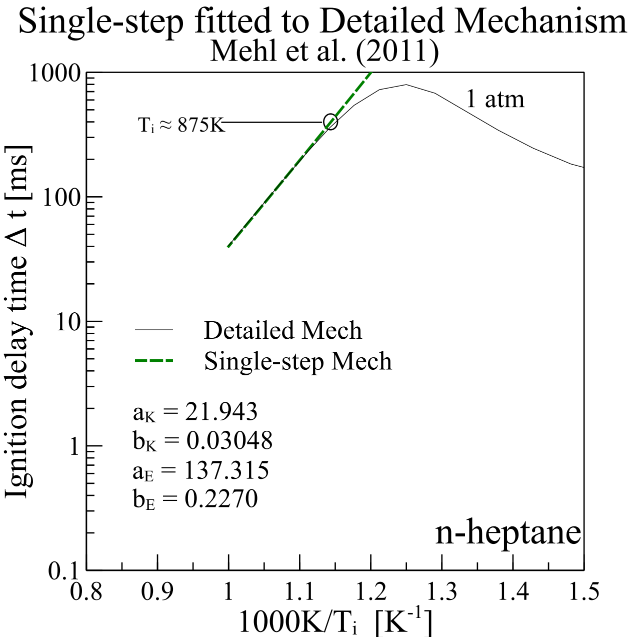


FIGURE 5.2: Ignition delay time as a function of $(1/T_i)$. Single-step reaction mechanism without continuous thermodynamics, fitted to detailed mechanism from Mehl et al. (2011), using the rate constant correlations of Table 5.1.

5.2.2 Single-step Reaction With Continuous Thermodynamics

The reaction rate parameters a_K and a_E used in Table 5.1 were applied to the single-step continuous thermodynamics model and the corresponding ignition delay times as a function of $(1/T_i)$ for each mechanism are shown in Table 5.2.

For initial temperatures $T_i \lessapprox 875$ K, the single-step models over-predict the ignition delay times compared to the detailed model, since the non-continuous single-step model does not quite fit the curve of the detailed model at this location (Fig. 5.2).

For higher initial temperatures, the non-continuous single-step model has a very slight difference in ignition delay time, since the detailed mechanism is not actually a straight line in this region, but curves slightly. In addition, the single-step model

TABLE 5.2: Ignition delay times at multiple initial temperatures T_i. Model predictions for single-step, with and without continuous thermodynamics ($\sigma = 2$), and the detailed reaction mechanism from Mehl et al. (2011).

Initial Temp T_i	**Single-step no CT**	**Single-step with CT** $\sigma = 2$	**Detailed mechanism**
875 K	0.393 s	0.404 s	0.362 s
900 K	0.235 s	0.242 s	0.230 s
925 K	0.145 s	0.149 s	0.145 s
950 K	0.092 s	0.095 s	0.093 s
975 K	0.060 s	0.062 s	0.061 s
1000 K	0.040 s	0.041 s	0.040 s

Model pridictions for pure n-heptane

with continuous thermodynamics very slightly over-predicts the ignition times of the non-continuous single-step model. This is because the effective activation energy E_{EFF}, and the effective pre-exponential coefficient K_{EFF} change slightly as they are updated in the continuous model at each time step with a new temperature for Eq. 3.31, 3.57 and 3.58. This updating of rate parameters was not incorporated in the non-continuous model which was used to fit the rate parameters a_K and a_E to the detailed model.

Additionally, θ and σ in Eq. 3.7 and 3.8 change slightly as shown in Figure 5.5 and for this reason also add to the delay compared to the non-continuous single-step model.

The rate parameters a_K and a_E could be re-fitted to the detailed mechanism using the continuous single-step model for any initial temperature T_i if greater accuracy was required. This was not done here because the purpose is to show the effects of the continuous thermodynamics model as opposed to reproducing the same results

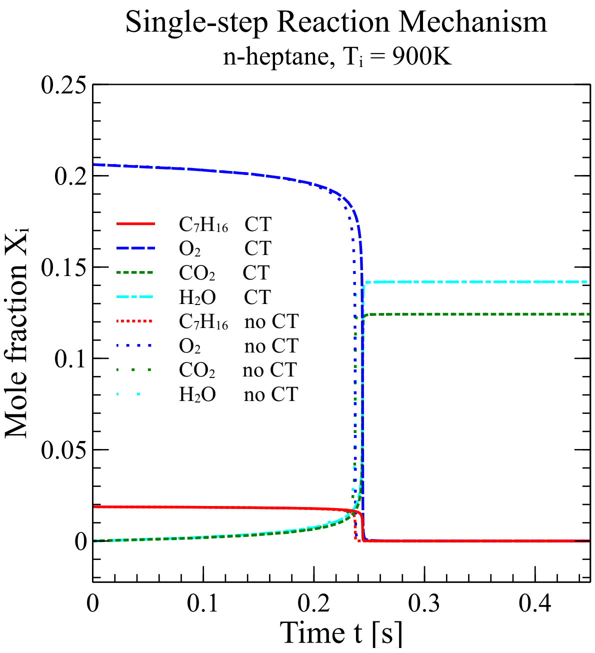


FIGURE 5.3: Species mol fraction as a function of t. Species shown with **CT** is the model prediction with continuous thermodynamics, $\theta =$ molecular mass of n-heptane and $\sigma = 2$; while species shown with **no CT** is the model prediction without continuous thermodynamics. Initial temperature of $T_i = 900$ K for both models.

from the pure fuel detailed model.

The changes in mol fraction X_i and temperature T for both single-step mechanisms are shown in Figure 5.3 and 5.4 respectively, and are almost identical to each other. The temperature rise for the detailed mechanism (dm) from Mehl et al. (2011) was included in the latter for reference.

The mol fractions predicted in the reactants and products by the continuous single-step model are identical to the non-continuous model (Fig. 5.3), but there is a slight delay in the consumption of reactants and production of product species during ignition due to the characteristics of the continuous model discussed earlier.

The final temperature rise predicted by the continuous single-step model is slightly

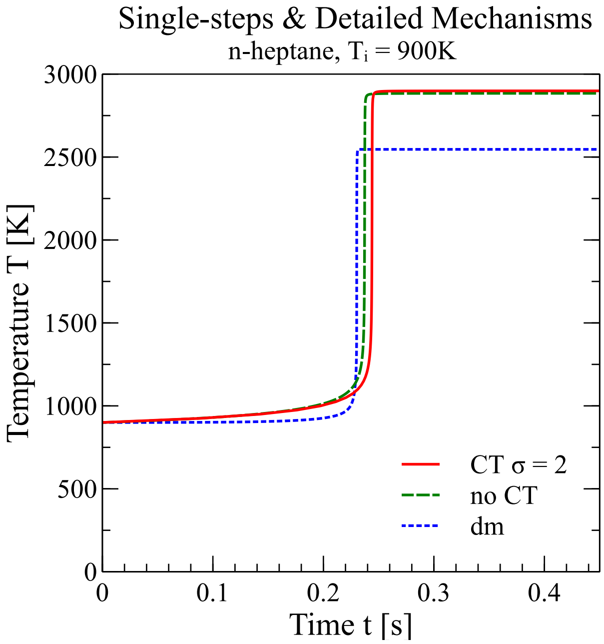


FIGURE 5.4: Temperature as a function of t. Curve shown with **CT** is the model prediction with continuous thermodynamics, $\theta =$ molecular mass of n-heptane and $\sigma = 2$; curve shown with **no CT** is the model prediction without continuous thermodynamics; and curve shown with **dm** is the detailed mechanism from Mehl et al. (2011). Initial temperature of $T_i = 900$ K for each model.

higher but almost identical to the non-continuous model (Fig. 5.4), again with the slight delay due to the characteristics of the continuous model discussed earlier.

The purpose here is to show the difference between the single-step models. If more accuracy between the single-step models were required, a new set of rate parameters a_K and a_E (unique to the continuous model), could be fitted to the non-continuous model.

The detailed mechanism was also shown in Figure 5.4 for discussion. The single step mechanisms over-predict the final gas temperatures because they do not include dissociation of $CO_2 \rightarrow CO$ and of $H_2O \rightarrow OH + H^+$, which reduce the final gas temperature considerably, whereas the detailed mechanism includes this dissociation.

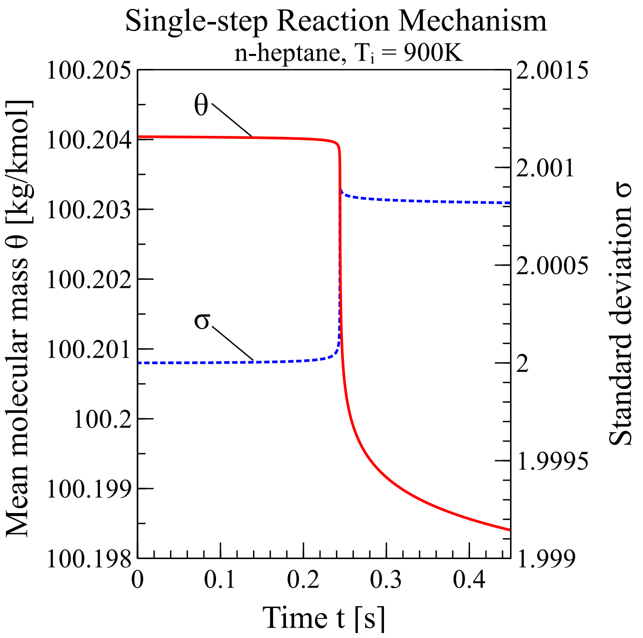


FIGURE 5.5: Distribution mean and standard deviation as a function of t, for the single-step reaction mechanism with continuous thermodynamics. Initial θ_i = molecular mass of n-heptane and initial $\sigma_i = 2$.

The distribution mean θ and standard deviation σ for the continuous single-step model are shown in Figure 5.5. θ decreases very slightly and σ increases very slightly.

The reaction rate constant expressions used (Eq. 3.57, 3.58) give a reaction rate which decreases as molecular mass increases at lower initial temperatures T_i. Lower molecular mass components will therefore be consumed more quickly, leading to a gradual shift of the distribution to higher molecular masses. However, at higher initial temperatures this trend reverses: the reaction rate begins to increase as molecular mass increases and higher molecular mass components are consumed more quickly, shifting the distribution back toward lower molecular masses instead (see Fig. 5.6).

Figures 5.7 and 5.8 show the distribution mean θ and the standard deviation σ throughout the combustion process for initial temperatures of 700 K, 750 K, 900 K,

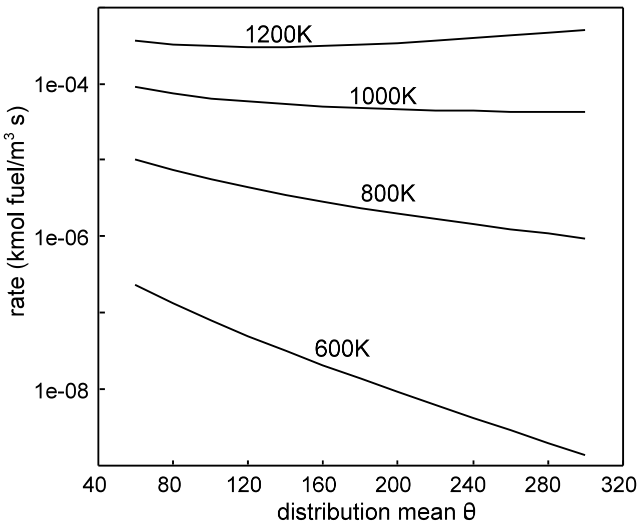


FIGURE 5.6: Reaction rates using n-paraffin correlations of Sabourin et al. 2016 (Eq. 3.57, 3.58) for a stoichiometric mixture

Source: (Hallett 2017)

950 K, and 1000 K.

The argument presented in the previous paragraph and the rates in Figure 5.6 show that for temperatures up to about 1000 K the distribution mean θ **increases** during combustion. Note the model is best fitted for $T_i \approx 900$ K and higher. For $T_i \gtrapprox 1000$ K, θ **decreases** during combustion as shown in Figure 5.7. In reality, the change in behaviour from θ increasing with time to decreasing occurs at around 900 K rather than 1000 K; the difference is due to the fact that the rate parameters are changing as reaction proceeds, a fact which is not reflected in Figure 5.6.

In general, the rate of decrease in θ gets larger as initial temperature increases, which only occurs for higher initial temperatures. This trend reverses for lower initial temperatures. The same behaviour in σ is shown in Figure 5.8, which confirms the region where this reversal occurs also affects σ, as it is dependent on θ since in the continuous model it is calculated from Ψ with Eq. 4.1, 4.3 and 3.9.

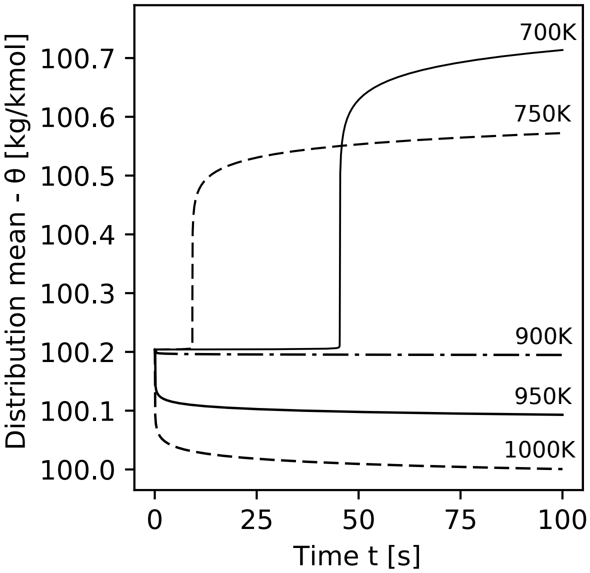


FIGURE 5.7: Distribution mean θ as a function of t for multiple, and low initial temperatures T_i, for the continuous single-step reaction mechanism. Initial $\theta_i =$ molecular mass of n-heptane and initial $\sigma_i = 2$, 1 atm. At lower temperatures θ's behaviour reverses.

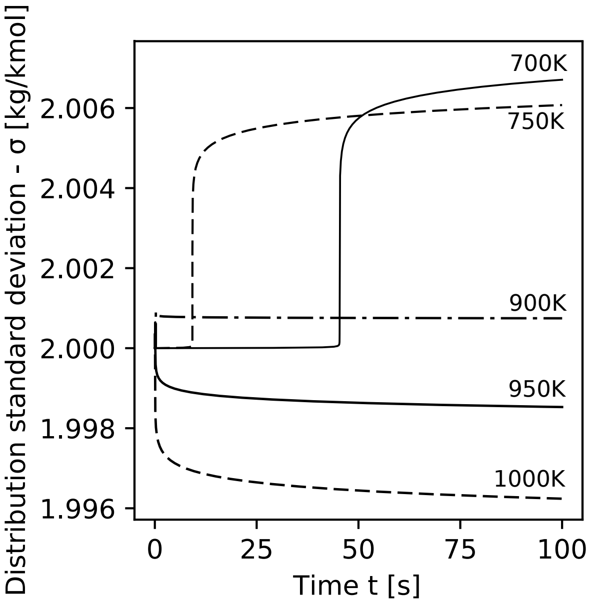


FIGURE 5.8: Distribution standard deviation σ as a function of t for multiple, and low initial temperatures T_i, for the continuous single-step reaction mechanism. Initial $\theta_i =$ molecular mass of n-heptane and initial $\sigma_i = 2$, 1 atm. At lower temperatures σ's behaviour reverses.

The ignition delay times as a function of initial gas temperature ($1/T_i$) for increasing distribution standard deviations σ are shown in Table 5.3 and Figure 5.9.

TABLE 5.3: Ignition delay times for multiple σ's at multiple temperatures, for the single-step reaction mechanism with continuous thermodynamics, initial θ_i = molecular mass of n-heptane.

Initial Temp T_i	$\sigma = 2$	$\sigma = 10$	$\sigma = 20$	$\sigma = 40$
875 K	0.40385 s	0.40388 s	0.40400 s	0.40447 s
900 K	0.24189 s	0.24188 s	0.24186 s	0.24179 s
925 K	0.14932 s	0.14928 s	0.14919 s	0.14881 s
950 K	0.09475 s	0.09470 s	0.09458 s	0.09407 s
975 K	0.06166 s	0.06162 s	0.06148 s	0.06095 s
1000 K	0.04108 s	0.04104 s	0.04091 s	0.04040 s

$\theta_i = 100.204$ kg/kmol, for n-heptane.

For $T_i > 875$ K as the distribution standard deviation σ increases, the ignition time decreases very slightly with θ held constant.

It must be noted that the insensitivity of ignition time to σ shown here is dependent on the particular variation of rate constants with θ chosen (Eq. 3.25, 3.26 and Table 3.1). A different dependence on species molecular mass could produce a larger change with σ.

5.3 Simulations with Two-step Mechanisms

5.3.1 Two-step Reaction Without Continuous Thermodynamics

In a similar manner as Section 5.2.1, the reaction rate parameters a_K and a_E used in Eq. 3.57 and 3.58 were adjusted for the first step of a two-step reaction mechanism in

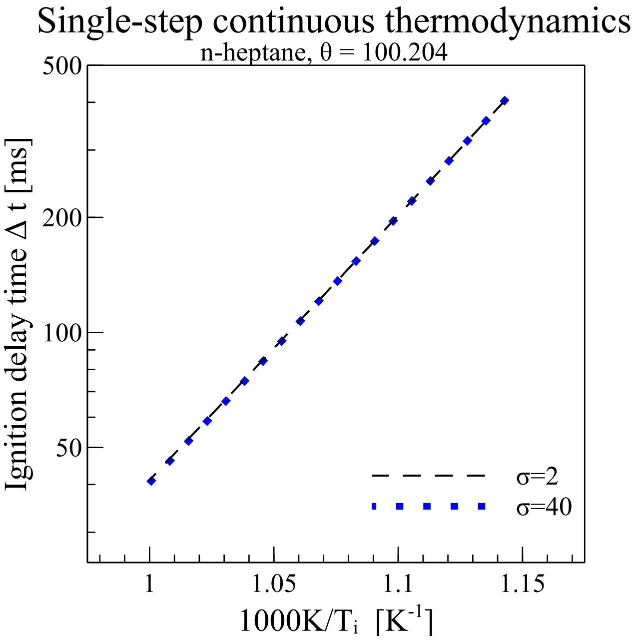


FIGURE 5.9: Ignition delay times as a function of $(1/T_i)$, at multiple distribution standard deviations σ; model predictions from Table 5.2. The ignition time is not a strong function of σ.

order to fit the ignition delay times with the detailed reaction mechanism. The other two parameters, b_K and b_E, which give the dependence of the rate on fuel molecular mass, were assumed to be the same as in the single-step model. The final rate parameters for the two-step mechanism are shown in Table 5.4.

The rate parameters in Table 5.4 apply only to the first step of the mechanism for n-heptane oxidation (**R1**), while the remaining part of the mechanism is a reversible reaction with different forward (**R2**$_F$) and backward (**R2**$_B$) reaction rates which accounts for CO oxidation and dissociation of CO_2 . Reaction **R2**$_F$ was modelled using the rate equation for oxidation of $CO \rightarrow CO_2$ from Howard et al. (1973), and the reaction **R2**$_B$ was modelled using the rate equation for dissociation of $CO_2 \rightarrow CO$ from Cooper and Hallett (2000) which was developed explicitly as a reverse reaction for

TABLE 5.4: Two-step fitted correlations for reaction rate parameters n-paraffins. a_K and a_E have been fitted to the detailed reaction mechanism in a similar fashion as the single step mechanism.

	a_K	b_K*	a_E	b_E*
n-paraffins	21.998	0.03048	134.875	0.2270

b_K* and b_E* from Sabourin et al. (2016) as in Table 3.1

the Howard et al. (1973) forward reaction.

The reaction equations **without** continuous thermodynamics become

$$\text{(Reaction: } \mathbf{R1}) \qquad C_7H_{16} + 7.5\, O_2 \rightarrow 7\, CO + 8\, H_2O \tag{5.3}$$

$$\text{(Reaction: } \mathbf{R2}_F) \qquad CO + 0.5\, O_2 \rightarrow CO_2 \tag{5.4}$$

$$\text{(Reaction: } \mathbf{R2}_B) \qquad CO_2 \rightarrow CO + 0.5\, O_2 \tag{5.5}$$

and the associated rates **without** continuous thermodynamic correlations become

$$\text{(Rate: } \mathbf{R1}) \qquad k_{R1} = 7.587 \times 10^{10} [C_7H_{16}]^1 [O2]^1 \exp(-18\,958/T) \tag{5.6}$$

$$\text{(Rate: } \mathbf{R2}_F) \qquad k_{R2_F} = 1.3 \times 10^{11} [CO][O_2]^{1/2} [H_2O]^{1/2} \exp(-15\,105/T) \tag{5.7}$$

$$\text{(Rate: } \mathbf{R2}_B) \qquad k_{R2_B} = 7.5 \times 10^{11} [CO_2] \exp(-46\,500/T) \tag{5.8}$$

The reaction equations and associated rates produce the following change in mol fractions shown in Figure 5.10.

The major difference between the two-step and detailed mechanisms is that the two-step does not include dissociation of H_2O. The two-step mechanism slightly over-predicts CO_2 and CO and slightly under-predicts O_2 in the products compared

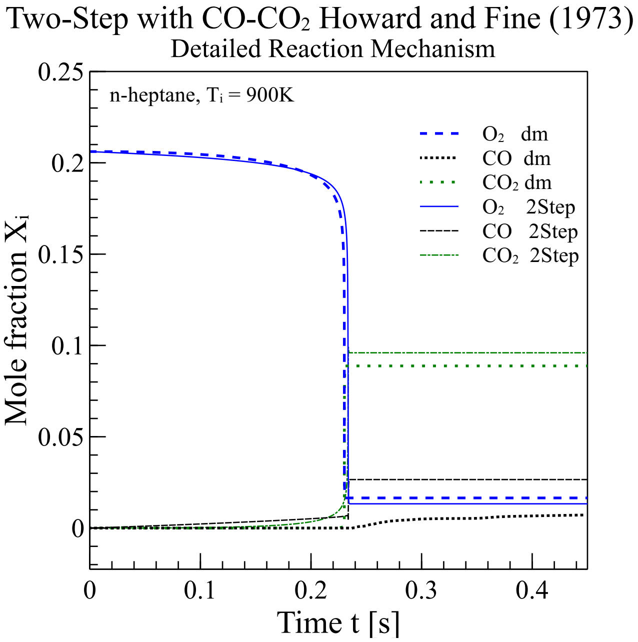


FIGURE 5.10: Species mol fraction as a function of t. Initial temperature of $T_i = 900$ K, Two-step mechanism without CT, $CO \rightarrow CO_2$ oxidation and $CO_2 \rightarrow CO$ dissociation from Howard et al. (1973) compared to the detailed mechanism (dm) from Mehl et al. (2011)

to the detailed mechanism. The solution of the detailed mechanism shows that CO is still rising; however, the reactor was stopped after 0.45 s instead of allowing it to run until a full equilibrium state was reached, since only the ignition process is of interest here.

5.3.2 Two-step Reaction With Continuous Thermodynamics

The mol fractions predicted in the reactants and products by the continuous two-step model are identical to the non-continuous model (Fig. 5.11), but there is a slight delay in the consumption of reactants and production of product species during ignition due to the characteristics of the continuous model discussed earlier.

The final temperature rise predicted by the continuous two-step model is slightly higher but almost identical to the non-continuous model (Fig. 5.12), again with the slight delay due to the characteristics of the continuous model discussed earlier. In addition, the two-step mechanism again over-predicts the final temperature since it does not include dissociation of H_2O.

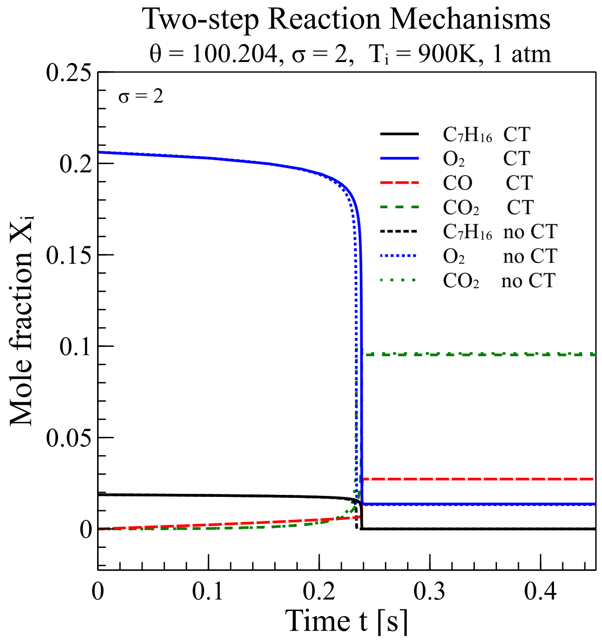


FIGURE 5.11: Species mol fraction as a function of t. Two-step model prediction with and without continuous thermodynamics, $\theta =$ molecular mass of n-heptane and $\sigma = 2$. Initial temperature of $T_i = 900$ K for both models. The two-step model with $CO \rightarrow CO_2$ oxidation and $CO_2 \rightarrow CO$ dissociation

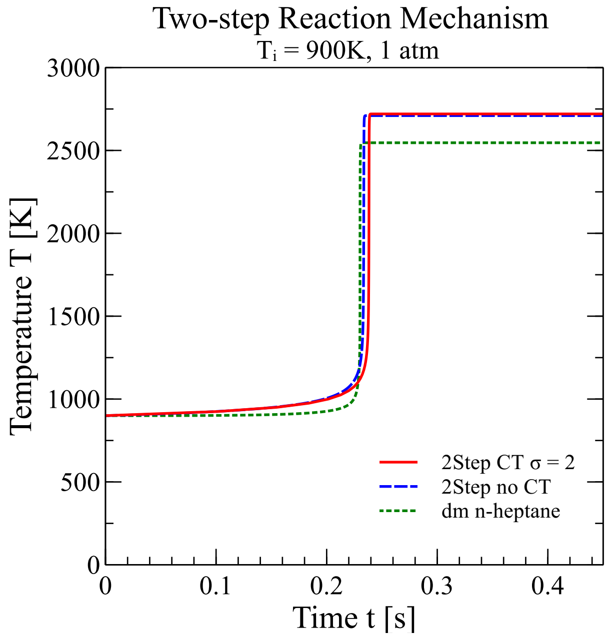


FIGURE 5.12: Temperature as a function of t. Two-step model prediction with and without continuous thermodynamics, $\theta =$ molecular mass of n-heptane and $\sigma = 2$; and curve shown with **dm** is the detailed mechanism from Mehl et al. (2011). Initial temperature of $T_i = 900$ K for each model.

The ignition delay times predicted by the continuous two-step model are almost identical to the non-continuous model, with a very slight delay seen in the high temperature branch region as shown in Figure 5.13. The detailed mechanism (dm) for n-heptane is shown for reference.

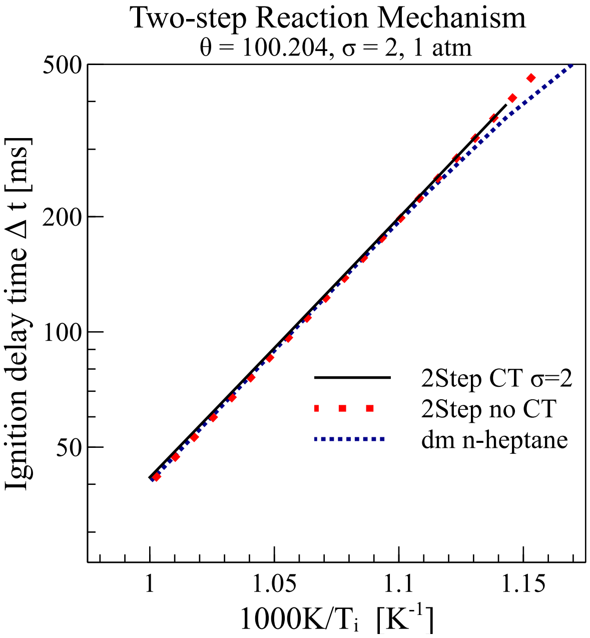


FIGURE 5.13: Ignition delay times as a function of ($1/T_i$), Two-step model prediction with and without continuous thermodynamics, $\theta =$ molecular mass of n-heptane and $\sigma = 2$. Detailed mechanism (dm) from Mehl et al. (2011) shown for reference. Initial temperature of $T_i = 900$ K for each model.

TABLE 5.5: Ignition delay times at multiple initial temperatures T_i. Model predictions for two-step, with and without continuous thermodynamics, and the detailed reaction mechanism

Initial Temp T_i	**Two-step no CT**	**Two-step with CT** $\sigma = 2$	**Detailed mechanism**
875 K	0.390 s	0.391 s	0.362 s
900 K	0.234 s	0.237 s	0.230 s
925 K	0.145 s	0.148 s	0.145 s
950 K	0.092 s	0.095 s	0.093 s
975 K	0.060 s	0.062 s	0.061 s
1000 K	0.040 s	0.042 s	0.040 s

Model predictions for pure n-heptane

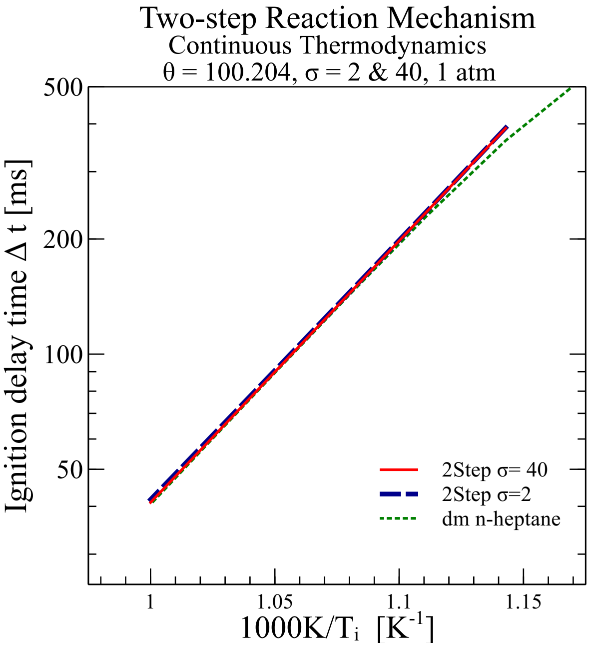


FIGURE 5.14: Ignition delay times as a function of $(1/T_i)$, two-step model prediction at multiple distribution standard deviations σ.

The ignition delay time for the continuous model at two distribution standard deviations $\sigma = 2$ and $\sigma = 40$ is shown in Figure 5.14 and Table 5.6.

TABLE 5.6: Ignition delay at multiple distribution standard deviations

Temp T_i	$\sigma = 2$	$\sigma = 40$
875 K	0.391 s	0.390 s
900 K	0.237 s	0.236 s
925 K	0.148 s	0.147 s
950 K	0.095 s	0.094 s
975 K	0.062 s	0.061 s
1000 K	0.042 s	0.041 s

The ignition delay time very slightly decreases with increasing σ, and overall the ignition delay time is not a strong function of the distribution standard deviation σ for the two-step model.

5.4 Simulations with Four-step Mechanisms

5.4.1 Four-step Reaction Without Continuous Thermodynamics

The detailed mechanism of Mehl et al. (2011) was modelled at 40 atm and compared to the four-step mechanism from Müller et al. (1992) as shown in Figure 5.15. The four-step model agrees very well with the detailed mechanism without making any modifications to the reaction rate parameters in Table 3.2.

Some of the deviation from the detailed mechanism may occur because Müller et al. (1992) developed the reaction rate parameters with the assumption of constant specific heats, which is not reproducible in Cantera and therefore contributes to a deviation from the detailed mechanism. It would be straightforward to modify the reaction rate parameters of Müller et al. (1992) to fit the detailed mechanism; however,

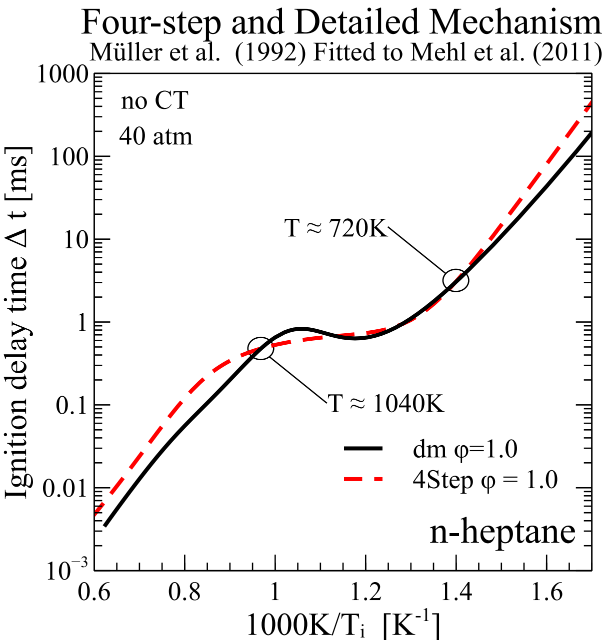


FIGURE 5.15: Ignition delay times as a function of $(1/T_i)$, four-step model prediction without continuous thermodynamics.

that is not the objective here, since the detailed model is for reference only and our main objective is to compare a four-step model with and without continuous thermodynamics.

5.4.2 Four-step Reaction With Continuous Thermodynamics

The four-step continuous thermodynamic model is compared to simplified and detailed models for pure n-heptane by setting the mean θ of the distribution to the molecular mass of n-heptane, and the standard deviation to a narrow distribution ($\sigma = 2$). The ignition delay times for the four-step models with and without continuous thermodynamics are shown in Figure 5.16 with the detailed mechanism of Mehl et al. (2011) included for reference. The continuous and non-continuous models agree quite well. The ignition delay time was not calculated for initial temperatures greater than $T_i > 1300$ K with the continuous model since there was a performance decrease with Cantera's internal reactor integrator (CVODES). Applying continuous thermodynamics in Cantera requires that the rate parameters, stoichiometry and thermodynamic properties be updated regularly. When this is done, the reactor integrator in Cantera needs to re-initialize, which results in very small time-steps. The time-steps became too small for practical calculations beyond this point. A proposed way to overcome this would be to write the governing equations for the constant pressure reactor and solve without using Cantera.

The temperature as a function of time was plotted for initial temperatures of $T_i = 720$ K and $T_i = 1040$ K since these temperatures occur where both four-step models intersect the detailed mechanism. They are shown for reference in Figure 5.17. The continuous and non-continuous models agree quite well with each other. Figure 5.17 (a)

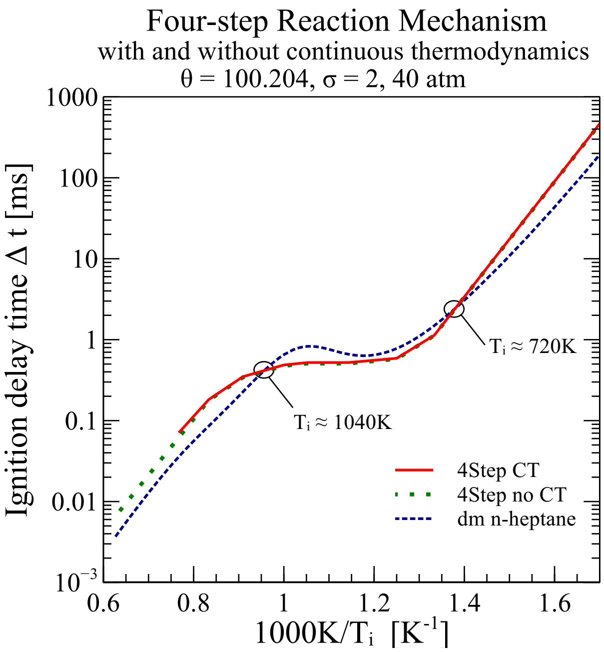
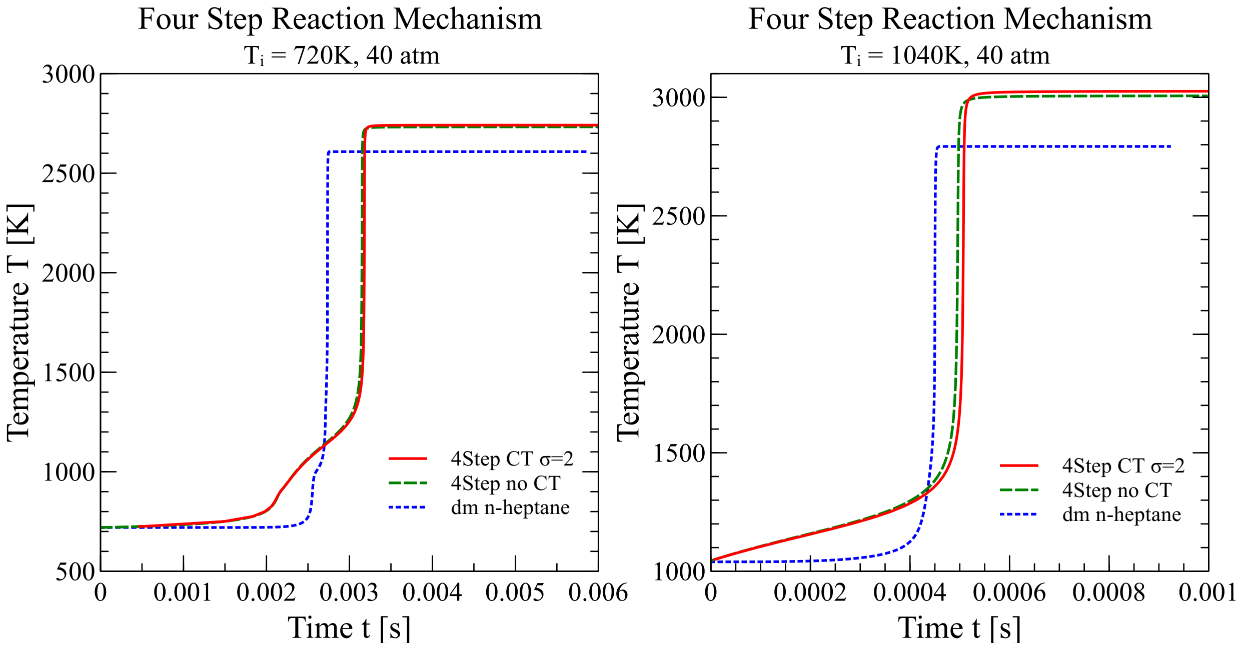


FIGURE 5.16: Ignition delay times as a function of $(1/T_i)$, four-step model prediction with and without continuous thermodynamics.

shows where the four-step models enter the negative temperature coefficient region of the ignition delay curve (Fig. 5.16). The effects of two-stage ignition and cool flame phenomena can be seen by the temperature behaviour at the beginning of ignition, which is more accurately predicted by the detailed model as expected. Figure 5.17 (b) shows the end of the negative temperature region where there is no longer two-stage ignition effects on the temperature curves.

The ignition delay time for the continuous model at two distribution standard deviations $\sigma = 2$ and $\sigma = 40$ is shown in Figure 5.18. The ignition delay time slightly increases with increasing σ in the high temperature (HT) branch, and slightly decreases in the low temperature (LT) branch, but overall the ignition delay time is not a strong function of the distribution standard deviation σ. Note, however, that as with the other rate models this conclusion only applies for the particular dependence of



(a) Two stage ignition within the negative temperature coefficient region

(b) Not within the negative temperature coefficient region

FIGURE 5.17: Temperature as a function of t. Four-step model prediction with and without continuous thermodynamics, $\theta =$ molecular mass of n-heptane and $\sigma = 2$. Initial temperatures: (a) $T_i = 720$ K, and (b) $T_i = 1040$ K

K_{EFF} and E_{EFF} on the distribution variable I chosen in this work.

The species mol fractions as a function of time for the four-step continuous and non-continuous models are shown in Figure 5.19, where the continuous and non-continuous models agree quite well with each other.

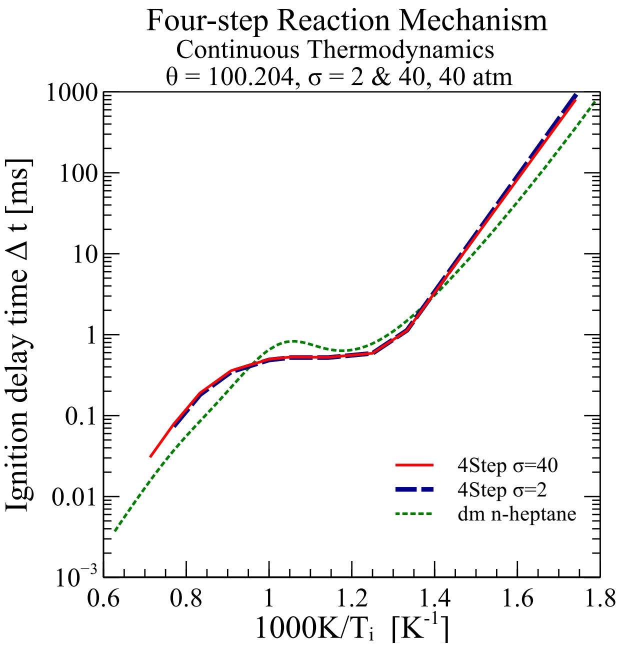


FIGURE 5.18: Ignition delay times as a function of $(1/T_i)$, four-step model prediction at multiple distribution standard deviations σ. The ignition time is not a strong function of σ.

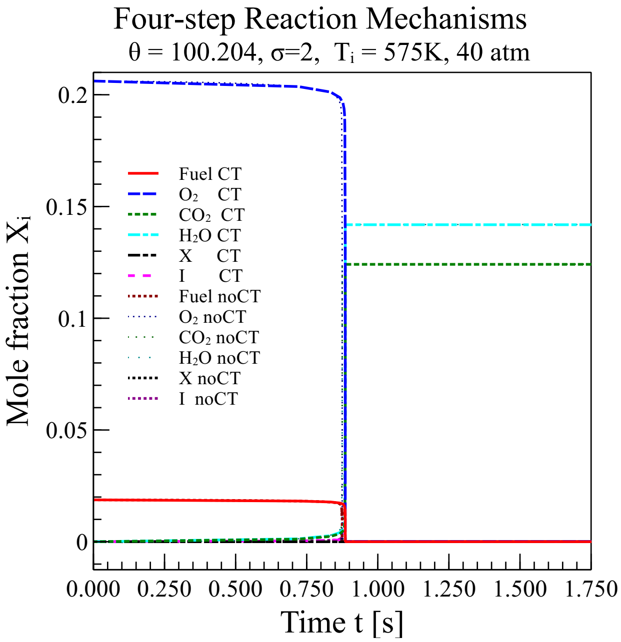
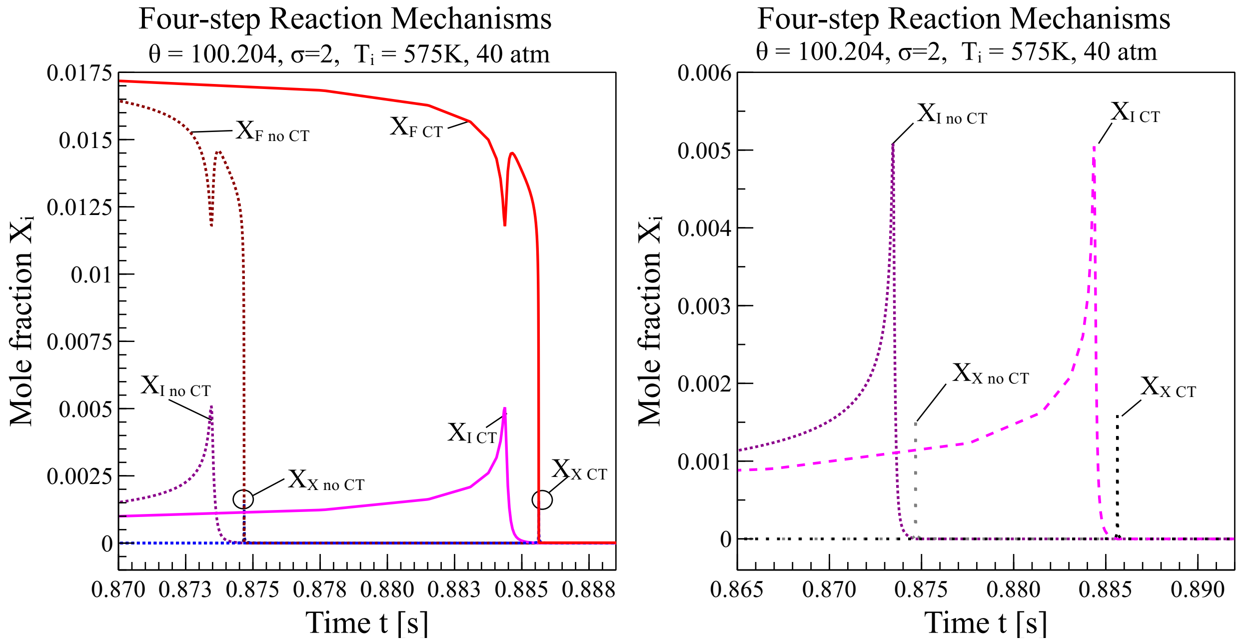


FIGURE 5.19: Species mol fraction as a function of t. Four-step model prediction with and without continuous thermodynamics.

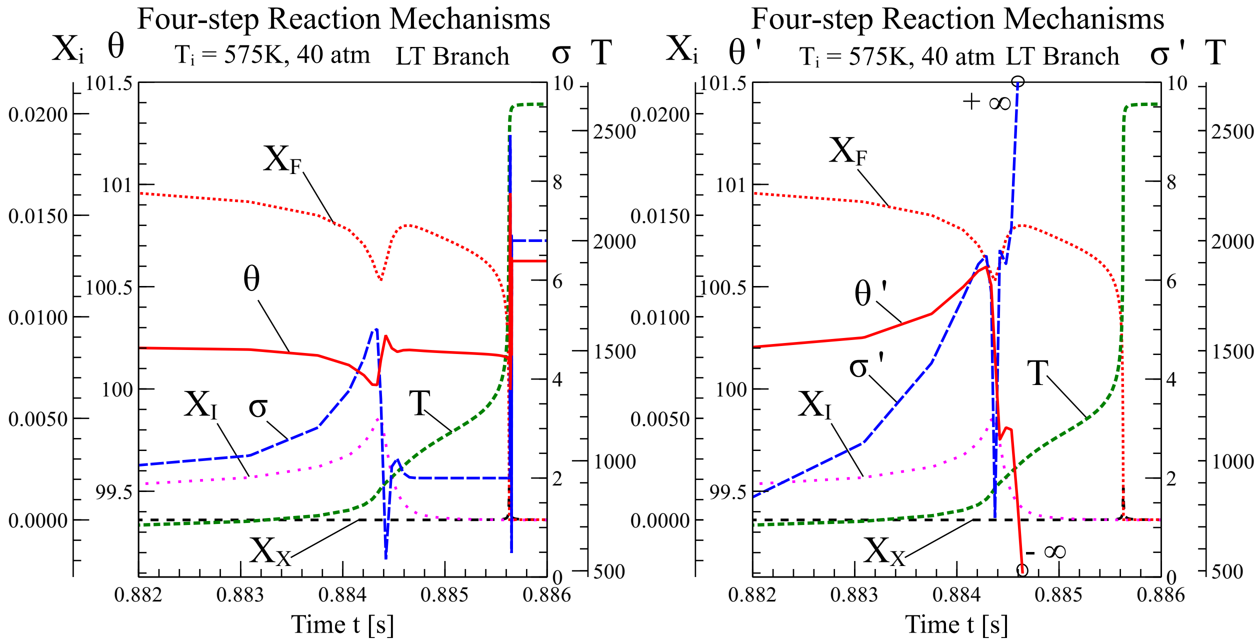


(a) X_F, X_I & X_X

(b) X_I & X_X

FIGURE 5.20: Four-step model prediction with and without continuous thermodynamics. (a) Fuel and intermediate species mol fraction as a function of t. (b) Intermediates only (expanded time scale).

An expanded time scale is needed to see the details during ignition and the changes to mol fractions of the fuel and intermediate species **I** and **X** as shown in Figure 5.20. There is a very slight delay in the ignition event of the continuous model. The behaviour of the fuel **F** and intermediates **I** and **X** mol fractions are duplicated very well by the continuous model. Fuel is initially consumed as intermediate **I** is produced, which begins in the LT-branch with $\mathbf{R3}_F$. The LT-branch is considered the first stage of ignition. After the concentration of **I** peaks, the reverse reaction $\mathbf{R3}_B$ produces fuel **F** again, and the remainder of intermediate **I** is consumed with $\mathbf{R3}_B$ and **R4**. As the temperature increases, the second stage of ignition begins when the HT-branch is activated. The concentration of intermediate **X** peaks, and **X** is consumed within the



(a) Fuel **F** distribution parameters: θ and σ. $\sigma_i = 2$

(b) Intermediate **I** distribution parameters: $\theta\,'$ and $\sigma\,'$, and $\sigma_i\,' = 2$

FIGURE 5.21: Four-step model predictions, expanded time scale of the LT-branch, mol fractions, temperature and continuous thermodynamic properties as a function of time t. (a) θ and σ. (b) $\theta\,'$ and $\sigma\,'$.

second stage of ignition with **R1** and **R2** (Fig.5.20 (b)).

The change in the fuel **F** gamma distribution $f(I)$, mean θ and standard deviation σ, and the change in the intermediate **I** gamma distribution $f\,'(I)$, mean $\theta\,'$ and standard deviation $\sigma\,'$, the latter which corresponds to the reverse reaction of $\mathbf{R3}_B$, are shown in Figure 5.21 (a) and (b) respectively. The behaviour of θ and σ (Fig.5.21 (a)) during the first stage ignition is related to the variation in rate with θ provided by the rate constant expressions from Sabourin et al. (2016) as shown in Figure 5.6. The behaviour is also due to the transition from the LT to HT branch as the temperature rises as well as the contribution of the reverse low temperature reaction $\mathbf{R3}_B$. It is also clear that during the first stage of ignition within the LT-branch θ and σ are related to the behaviour of the fuel mol fraction $X_{\mathbf{F}}$ and stabilize before entering the HT-branch, at which point the fuel is completely consumed and the values of θ and σ no longer have any significance.

There is a similar behaviour with $\theta\,'$ and $\sigma\,'$ (Fig.5.21 (b)). However, instead they are related to the behaviour of the intermediate **I**, and become undefined as $X_{\mathbf{I}}$ approaches zero (i.e. the values shown after $X_I \to 0$ are meaningless). In addition $\theta\,'$ and $\sigma\,'$ do not participate in the second stage of ignition in the HT-branch since **I** no longer exists at this point. θ and $\theta\,'$ are linked as follows: The trend in θ shows that fuel species of higher molecular weight are being consumed faster; this leads to more high molecular mass intermediate **I** being formed, so that $\theta\,'$ goes up. Refer to Figure 5.7, where it is shown that θ starts to decrease with time for $T > 900$ K because of the molecular mass dependence of reaction rate in Figure 5.6.

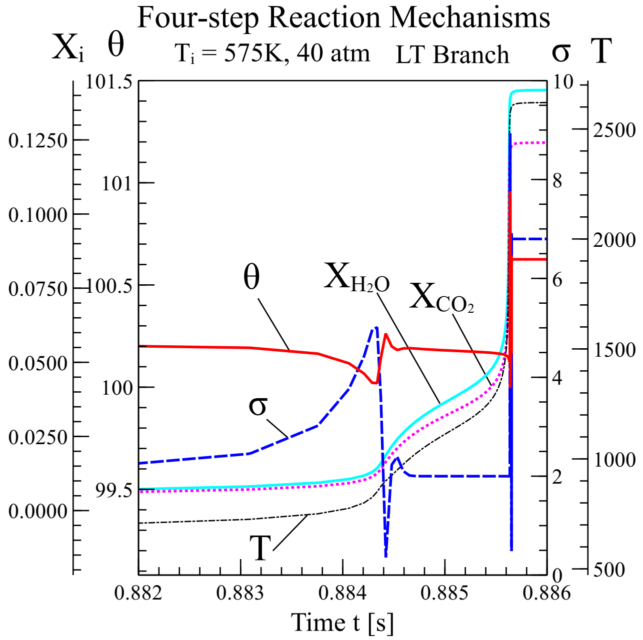
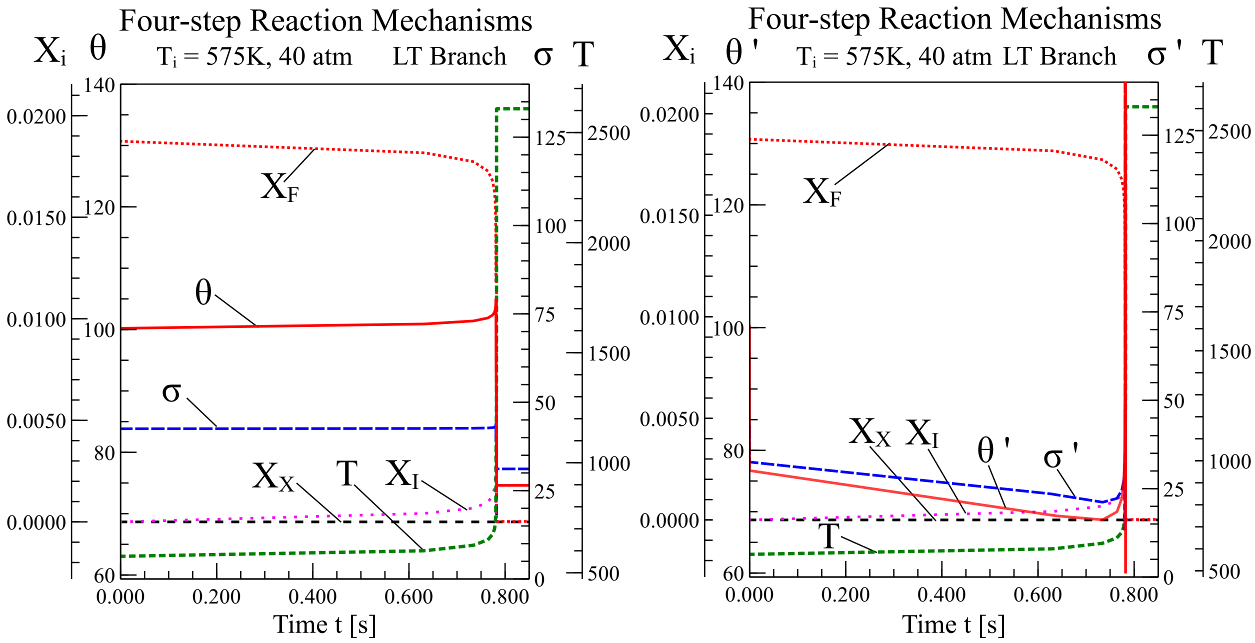


FIGURE 5.22: Four-step model predictions, expanded time scale of the LT-branch, product mol fractions, temperature and continuous thermodynamic properties as a function of time t.

The development of the product species CO_2 and H_2O during the first stage of combustion is shown in Figure 5.22.

The four-step model was also tested with an initial standard deviation of $\sigma = 40$, and the change in θ, σ, $\theta\,'$ and $\sigma\,'$ are shown in Figure 5.23 (a) and (b) respectively. In Fig 5.23 (a) θ is rising, because light components are being consumed more quickly than heavy ones as indicated by Figure 5.6, and the consequence is that the intermediate $X_{\mathbf{I}}$ produced is lighter, and gets progressively lighter as the reaction proceeds, as shown in Fig 5.23 (b) where $\theta\,'$ decreases.

The behaviour of the fuel $X_{\mathbf{F}}$ and intermediates $X_{\mathbf{I}}$ and $X_{\mathbf{X}}$ can be seen with an expanded time scale during the first stage of ignition in Figure 5.24 (a) and (b). The behaviour of the fuel and intermediates at $\sigma = 40$, is essentially the same as with a



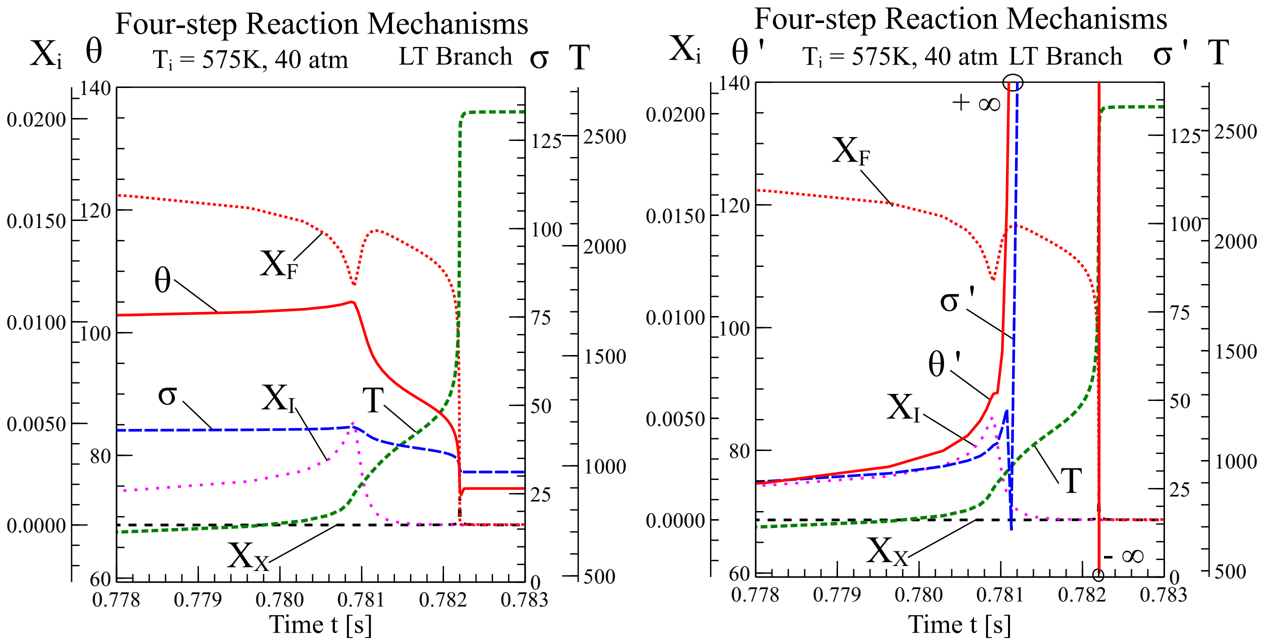
(a) Fuel **F** distribution parameters: θ and σ.

(b) Intermediate **I** distribution parameters: $\theta\,'$ and $\sigma\,'$.

FIGURE 5.23: Four-step model predictions, expanded time scale of the LT-branch, mol fractions, temperature and continuous thermodynamic properties as a function of time t. $\sigma_i = 40$ (a) θ and σ. (b) $\theta\,'$ and $\sigma\,'$.

narrow distribution $\sigma = 2$; however, there is a decrease in ignition delay time since the initial gas temperature ($T_i = 575$ K) corresponds to the LT-branch (Fig 5.18). The mean θ and standard deviation σ decrease overall during the first stage of ignition before stabilizing, which is similar to the behaviour of the model at $\sigma = 2$ Fig 5.21 (a).

When $X_{\mathbf{I}}$ gets very small, $\theta\,'$ and $\sigma\,'$ are essentially undefined Fig 5.21 (b), because there is no longer any material there to assign properties to.



(a) Fuel **F** distribution parameters: θ and σ. $\sigma_i = 40$

(b) Intermediate **I** distribution parameters: $\theta\,'$ and $\sigma\,'$, where $\sigma_i\,' = 40$

FIGURE 5.24: Four-step model predictions, expanded time scale of the LT-branch, mol fractions, temperature and continuous thermodynamic properties as a function of time t. $\sigma_i = 40$ (a) θ and σ. (b) $\theta\,'$ and $\sigma\,'$.

Chapter 6

Conclusions and Recommendations

6.1 Conclusions

Simplified reaction mechanisms normally designed for pure fuels can be adapted for mixtures which are modelled as probability density functions. This is the essential objective of applying continuous thermodynamics in this work. To achieve this, the rate parameters, reaction mechanisms, and thermal properties are represented as simple functions of species molecular mass so that they can be integrated over the probability density function and then be incorporated into the simplified reaction mechanisms.

A continuous thermodynamics model can very accurately predict the behaviour of a pure fuel by using a narrow distribution with a small standard deviation σ and selecting the mean θ to match the pure fuel mean molecular mass. It follows that the models presented in this work could predict the ignition of mixtures of multi-component fuels by selecting an appropriate mean molecular mass θ and a wider distribution with a standard deviation such as $\sigma = 20, 40...$ etc. Future work, either with a detailed mechanism for a mixture, or results obtained from experiments should be studied for comparison.

When a larger standard deviation σ is selected to represent a mixture, there will be a slight change in the ignition delay time as a result. This is expected as a larger standard deviation represents a wider range of fuel components. The amount by which changing σ affects ignition however is totally dependent on the way the effective pre-exponential coefficient (K_{EFF}) and activation energy (E_{EFF}) are related to the distribution variable I. A different set of linearization coefficients (b_K and b_E) could produce a lot more change in the average reaction rates $\bar{W}$ with changing σ.

The single-, two- and four-step continuous thermodynamic models were successfully tested with a simple case of a constant pressure homogeneous reactor. The software Cantera was used to assist in solving the continuous thermodynamic equations. Cantera simplified the work because it provided a solution for constant pressure ignition which could be adapted to a continuous mixture.

6.2 Recommendations

The first recommendation for future work is to develop a continuous model with more steps, such as the seven-step model by Vandersickel et al. (2013). Such a model would more accurately predict the behaviour of the negative temperature coefficient region, two-stage ignition and cool flames.

Another recommendation is to develop a droplet ignition model using Cantera, because it has tools for surface interfaces and multiple fuel phases (liquid and gas) which could assist in modelling evaporation and ignition of continuous droplet mixtures.

Another recommendation is, with further development of the models presented in this thesis, laminar flame propagation could be incorporated into these continuous simplified mechanisms. The diffusion terms in the transport equations would need to be developed that incorporate the continuous variables, and cases such as 1-D premixed flame speed could be compared to experimental data.

A final recommendation is to develop a solver for the governing equations without the use of Cantera. One reason is that Cantera does not take well to updating each reaction equation at every time-step of the reactor solver, which is needed to solve the continuous thermodynamic equations, with varying rate coefficients; stoichiometry; and thermodynamic properties. This is particularly noticeable with more reaction steps. Some of the difficulties include controlling the reactor solver time-step size. When properties are modified in the Cantera reactor solver, it re-initializes automatically, resulting in very small time-steps at the beginning of each loop, and therefore a large amount of data is generated to predict short ignition events.

References

Abdel-Qader, Zainab and William L.H. Hallett (2005). “The role of liquid mixing in evaporation of complex multicomponent mixtures: modelling using continuous thermodynamics”. *Chemical Engineering Science* 60, pp. 1629–1640.

Aris, R. and G. R. Gavalas (1966). “On the Theory of Reactions in Continuous Mixtures”. *Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences (1934-1990)* 260, pp. 351–393.

Astarita, Gianni and Rafaella Ocone (1992). “Chemical reaction engineering of complex mixtures”. *Chemical Engineering Science* 47, pp. 2135–2147.

Bird, R.B., W.E. Stewart, and E.N. Lightfoot (1961). *Transport Phenomena*. First Edition. Wiley.

Chou, M.Y. and T.C. Ho (1988). “Continuum theory for lumping nonlinear reactions”. *American Institute of Chemical Engineers Journal* 34, pp. 1519–1527.

Cooper, J. and W.L.H. Hallett (2000). “A numerical model for packed-bed combustion of char particles”. *Chemical Engineering Science* 55, pp. 4451 –4460.

Egolf, Leanne M. and Peter C. Jurs (1992). “Estimation of autoignition temperatures of hydrocarbons, alcohols, and esters from molecular structure”. *Industrial & Engineering Chemistry Research* 31, pp. 1798–1807.

Gonçalves de Azevedo, Filipa, John F. Griffiths, and Silvana S.S. Cardoso (2014). “Explosion criteria for a three-step Gray and Yang, non-isothermal branching reaction scheme in fluids with natural convection". *Combustion and Flame* 161, pp. 1752–1755.

Goodwin, David G., Harry K. Moffat, and Raymond L. Speth (2017). *Cantera: An Object-oriented Software Toolkit for Chemical Kinetics, Thermodynamics, and Transport Processes*. Version 2.3.0.

Gordon, S. and B.J. McBride (1971). *Computer Program for Calculation of Complex Chemical Equilibrium Composition, Rocket Performance, Incident and Reflected Shocks and Chapman-Jouguet Detonations*.

Gray, B.F. and C.H. Yang (1965). "On the unification of the thermal and chain theories of explosion limits". *Journal of Physical Chemistry* 69, pp. 2747–2750.

Hallett, William L.H. (2000). "A simple model for the vaporization of droplets with large numbers of components". *Combustion and Flame* 121, pp. 334–344.

Hallett, William L.H. (2017). Personal communication, Thesis supervisor. Professor, Department of Mechanical Engineering, University of Ottawa.

Ho, T.C. and Benjamin S. White (2015). "Continuum approximation of large reaction mixtures in reactors with backmixing". *American Institute of Chemical Engineers Journal* 61, pp. 159–165.

Howard, J.B., G.C. Williams, and D.H. Fine (1973). "Kinetics of carbon monoxide oxidation in postflame gases". *Symposium (International) on Combustion* 14. Fourteenth Symposium (International) on Combustion, pp. 975 –986.

McBride, B.J., S. Gordon, and M.A. Reno (1993). *Coefficients for Calculating Thermodynamic and Transport Properties of Individual Species*.

Mehl, Marco et al. (2011). "Kinetic modeling of gasoline surrogate components and mixtures under engine conditions". *Proceedings of the Combustion Institute* 33, pp. 193 –200.

Müller, U.C., N. Peters, and A. Liñán (1992). "Global kinetics for n-heptane ignition at high pressures". *Symposium (International) on Combustion* 24. Twenty-Fourth Symposium on Combustion, pp. 777 –784.

Poling, Bruce E., John M. Prausnitz, and John P. O'Connell (2001). *The properties of gases and liquids*. New York: McGraw-Hill.

Sabourin, Shaun W., Claire I. Boteler, and William L.H. Hallett (2016). "Droplet ignition of approximately continuous liquid mixtures of n-paraffins and n-alkyl aromatics". *Combustion and Flame* 163, pp. 326 –336.

Smith, Gregory P. et al. (2000). "GRI-Mech". Version 3.0.

Sonntag, R.E., C. Borgnakke, and G.J. Van Wylen (2002). *Fundamentals of Thermodynamics, Sixth Edition*. Wiley.

Stagni, A. et al. (2016). "Skeletal mechanism reduction through species-targeted sensitivity analysis". *Combustion and Flame* 163, pp. 382–393.

Tamim, J. and W.L.H. Hallett (1995). "A continuous thermodynamics model for multicomponent droplet vaporization". *Chemical Engineering Science* 50, pp. 2933 –2942.

Tolonen, Erik, William L.H. Hallett, and Carlos M. Monreal (2015). "Droplet evaporation behaviour of a liquid fuel from chicken litter". *Fuel* 139, pp. 26–34.

Turns, Stephen R. (2012). *An Introduction to Combustion: Concepts and Applications, Third Edition*. McGraw-Hill Companies Inc.

Vandersickel, A., Y. M. Wright, and K. Boulouchos (2013). "Global reaction mechanism for the auto-ignition of full boiling range gasoline and kerosene fuels". *Combustion Theory and Modelling* 17, pp. 1020–1052.

Westbrook, Charles K. and Frederick L. Dryer (1981). "Simplified Reaction Mechanisms for the Oxidation of Hydrocarbon Fuels in Flames". *Combustion Science and Technology* 27, pp. 31–43.

Zabetakis, M.G., A.L. Furno, and G.W. Jones (1954). "Minimum spontaneous ignition temperatures of combustibles in air". *Industrial and Engineering Chemistry* 46, pp. 2173–2178.

Appendix A

Detailed Methods and Procedures - Cantera and Python

Appendix A is comprised of the detailed methods and procedures for building a Cantera constant pressure reactor and running it with a general continuous thermodynamics model. It should be read along with Chapter 4 and is included here as reference information.

A.1 Python and Cantera

Python is an open source object-oriented programming language. It is also one of the most documented and supported programing languages for Cantera, and has therefore been selected for the continuous thermodynamics model in this thesis. The Python script `reactor1_CF.py` in Listing A.1, shows an example that comes with the Cantera source files. It has been modified slightly to use a n-heptane (C_7H_{16}) single-step reaction mechanism, and does not include continuous thermodynamics. It is shown here because it is a short program and essentially has the same structure required for the detailed continuous thermodynamics model. It will be used to discuss the basic structure of the model, then the detailed model will be presented.

LISTING A.1: Simple Python Cantera reactor - reactor1_CF.py

```
import cantera as ct

ct.add_directory('/home/foxc/droplet/input_cti') # file location
gas = ct.Solution('1step_C7H16.cti')
gas.TPX = 496.15, ct.one_atm, 'C7H16:1,O2:11,N2:41.382'

r = ct.IdealGasConstPressureReactor(gas)
sim = ct.ReactorNet([r])
time = 0.0
states = ct.SolutionArray(gas, extra=['t'])

print('%10s %10s %10s %14s' % ('t [s]','T [K]','P [Pa]','h [J/kg]'))
for n in range(400):
   time += 20
   sim.advance(time)
   states.append(r.thermo.state, t=time)
   print('%10.3e %10.3f %10.3f %14.6e' % (sim.time, r.T,
      r.thermo.P, r.thermo.h))
```

(Goodwin et al. 2017)

There are essentially four sections to this example program:

- Import Cantera as a Python module
- Input gas phase and set initial conditions
- Define a reactor containing the gas phase
- Print the states of the gas as it advances in time through the reactor simulation

Cantera is used as a module which is imported into Python via: `import cantera as ct`. Once this command has been entered, Cantera functions can be called within the Python script to perform thermodynamic, chemical kinetic, and transport process calculations.

Following this is: `gas = ct.Solution('1step_C7H16.cti')`, which is used for importing the Cantera solution for the phase `1step_C7H16.cti`, and setting it equal to a variable, in this case `gas`. A Cantera solution is defined as a mixture of two or more constituents (chemical species) that are completely mixed on molecular length scales (Goodwin et al. 2017). The `ct.` in front of the `Solution` class tells Python to use a Cantera function. The imported phase could be a solid, liquid, gas or a surface interface. The `1step_C7H16.cti` is a **C**an**T**era **I**nput file or `(.cti)`. The input file must be located in the same working file directory as the Python script file, i.e. `reactor1_CF.py`, or you can tell Cantera to look elsewhere ie: `ct.add_directory()`.

The input file is very important and is required for defining the phase of `gas`. It contains a list of all possible phases, chemical species, and equilibrium reaction equations, with the associated rate and thermodynamic data. The input file cannot directly be modified by Cantera; however, for certain properties (rate and thermodynamic data) the input file is only needed to initialize a process such as a reactor, and it is possible to feed Cantera with new input data for certain properties from functions written in Python. For example, thermodynamic and kinetic properties can be formulated as functions of variables in Python, and then fed into Cantera after the reactor solver has been initialized.

Following the input file, the initial conditions are set using: `gas.TPX = ....` The `.TPX` at the end of `gas` is used to initialize gas temperature, pressure and moles of reactant chemical species. It is also possible to change one or two properties at a time simply by `gas.T` or `gas.TP` respectively. The default input units Cantera is expecting for temperature is K, and for pressure Pa. The number of moles of the reactants are also set here, and in this case are set as follows:

$$C_7H_{16} + 11(O_2 + 3.762N_2) \rightarrow Products \tag{A.1}$$

The chemical species, and composition of the products will depend on the species and reaction equations within the input file: `1step_C7H16.cti`. See section A.2. Once the program runs, Cantera will solve over time the thermodynamic and kinetic properties for all the reactants and products.

The next section of `reactor1_CF.py`, shown in Listing A.2, is used to initialize and build a single ideal gas constant pressure reactor: `r`, and fill it with `gas` at the initial conditions previously defined.

LISTING A.2: Building a reactor network - reactor1_CF.py

```
r = ct.IdealGasConstPressureReactor(gas)
sim = ct.ReactorNet([r])
time = 0.0
states = ct.SolutionArray(gas, extra=['t'])
```

(Goodwin et al. 2017)

This section only defines the state of the reactor and it is not running a simulation. First the reactor is incorporated into a network `sim = ct.ReactorNet([r])`. In this case there is only one reactor in the network, represented by the array `[r]`.

Following this the initial time is set, and a new array `states` is defined. Later as the reactor simulation progresses, the `states` array will fill up with data (chemical kinetic, thermodynamic and transport properties). The size of this array is set to allow all the possible data available in `ct.SolutionArray` for `gas`.

The final section shown in Listing A.3 prints the results to screen, and advances the reactor incrementally through time. The results can also be saved to a `(.csv)` file and/or plotted from Python which will be discussed later.

LISTING A.3: Printing results of reactor network - reactor1_CF.py

```
print('%10s %10s %10s %14s' % ('t [s]','T [K]','P [Pa]','h [J/kg]'))
for n in range(400):
   time += 20
   sim.advance(time)
   states.append(r.thermo.state, t=time)
   print('%10.3e %10.3f %10.3f %14.6e' % (sim.time, r.T,
      r.thermo.P, r.thermo.h))
```

(Goodwin et al. 2017)

The for loop is used to define the number of outputs, in this case lines to display on the screen, and the time step between those displayed outputs is set to increment in this case by 20 s, using `time += 20`. Recall that `sim` contains the reactor. The command `sim.advance(time)` advances the state of the reactor network from the current time to the new incremented `time`, taking as many internal Cantera integrator time-steps as necessary but only returning the results at the new `time`.

This causes an issue with the continuous thermodynamics model because the properties (such as mole fractions, molecular mass etc.) at the internal Cantera integrator time steps are required for the model to work. It will be shown later that in order to obtain results from the internal integrator time steps, the `sim.step()` function will be used, which takes a single internal time-step toward the end time of the reaction.

After the reactor advances one time step, all thermodynamic states within it are updated, and stored in the `states` array using `states.append(r.thermo.state, t = time)` and finally the time, temperature, pressure and enthalpy in this case are printed to screen as the for-loop progresses from `n=0`, to `n=400`. Note there is a myriad of properties that can be printed from the object `.thermo`, and those related to the continuous thermodynamic model will be presented in later sections. For further details see Goodwin et al. (2017).

The example of Listing A.1 is essentially the same structure required for the detailed continuous thermodynamics model, and the following sections will expand on this to present the detailed workings of the model.

A.2 Cantera Input File (`.cti`)

Cantera input files (`.cti`) are made up of three sections: phases (Listing A.4), species (Listing A.5), and reaction data (Listing A.6). See Appendix B for the full Cantera input file.

Before the phase is specified it is recommended to set the units within the `(.cti)` file. Note that the units set here will only apply within this `(.cti)` file, and will be converted to the Cantera default units, which are listed in Goodwin et al. (2017). In addition, if they are not set here, then the default units will simply apply.

Listing A.4: Phase data - 1step_C7H16.cti

```
#-------------------------------------------------------------------
# Phase data - # n-heptane
#-------------------------------------------------------------------
units(length=’m’, quantity=’kmol’, time=’s’, act_energy=’K’)

ideal_gas(name = "1step",
     elements = " C H O N ",
     species = """ C7H16 O2 N2 CO2 H2O """,
     reactions = "all",
     initial_state = state(temperature = 300.0, pressure = OneAtm))
```

(Goodwin et al. 2017)

The phase could be a solid, liquid, gas or a surface interface. In this case the phase imported to Cantera is an `ideal_gas` named `"1step"`. Multiple phases can be defined here.

The chemical elements present within each phase need to be specified. For example: `elements = " C H O N "` represents all the elements present within a typical n-paraffin reaction. The names of these elements are strings that Cantera uses to reference element properties from a C++ library file where it finds the defined chemical properties such as atomic weights, full name and symbol.

The chemical species that are composed of those elements within the phase also need to be specified; for example, `species = """ C7H16 O2 N2 CO2 H2O """`. These represent all the species present within a typical single-step n-paraffin reaction. The names of these species are also strings; however, the thermodynamic properties of the species are defined within the same `(.cti)` file below.

The next line `reactions="all"` specifies which reactions within this `(.cti)` file apply to this phase, and in this case all are set to apply. The initial state of the phase, temperature, and pressure are also set here, and can be changed after importing to Cantera, as presented before.

The next section of the input file defines the properties of every species within the phase, shown in Listing A.5. In this case only `C7H16` is shown, but the full listing can be seen in Appendix A. The species data are from Gordon and McBride (1971) & McBride et al. (1993), who have provided a full database of almost every species of typical interest in combustion problems.

LISTING A.5: Species data - 1step_C7H16.cti

```
#----------------------------------------------------------------------
#  Species data
#----------------------------------------------------------------------
# Use one NASA parameterization for T < 1000 K, and another for
# T > 1000 K upto 3500.00 K.
# There are seven(7) coefficents used for molar heat capacity,
# molar enthalpy and absolute molar entropy CFOX

species(name = "C7H16",
   atoms = " C:7 H:16",
   thermo = (
     NASA( [ 200.00, 1000.00], [ 1.115324840E+01, -9.494154330E-03,
            1.955711810E-04, -2.497525200E-07, 9.848732130E-11,
           -2.677117350E+04, -1.590961100E+01] ),
     NASA( [ 1000.00, 6000.00], [ 1.853547040E+01, 3.914204680E-02,
           -1.380302680E-05, 2.224038740E-09, -1.334525800E-13,
           -3.195007830E+04, -7.019028400E+01] )
```

(Gordon and McBride 1971) & (McBride et al. 1993)

For thermodynamic properties, the 7-coefficient NASA polynomial parameterization is used, where seven coefficients (a_0 to a_6) for two temperature ranges are specified to solve the species reference state properties for specific heat $\bar{c_p}^{\circ}$ (Eq. 3.139), enthalpy $\bar{h}^{\circ}$ (Eq. 3.140), and entropy $\bar{s}^{\circ}$ (Eq. 3.141).

These coefficients cannot be changed within the input `(.cti)` file; however, in the continuous thermodynamics model, these coefficients are functions of the distribution mean θ, and therefore require updating as the reaction progresses. It will be shown later that after the coefficients are used to initialize the properties of the the species within Cantera, the coefficients that Cantera uses can be modified without changing the values within the input `(.cti)` file.

The final section of the input file contains all the required reaction equations. In this example there is only one equation for a single-step reaction mechanism (Eq. 3.11).

However as described earlier, the main criterion for this model is to use Cantera with a reduced scheme, first starting with a single-step mechanism, analyzing the results, and then incrementally moving up to second, and then a four-step mechanism. It is in this section (Listing A.6) where those additional reaction mechanism equations are required.

The line `reaction("C7H16 + 11 O2 => 7 CO2 + 8 H2O "` defines the forward rate equation, followed by three numbers `[A,b,E]` which represent the Arrhenius rate parameters from (Eq. 3.39, 3.40), and then followed by the reactant reaction orders `order = " C7H16:1 O2:1 " ).`

LISTING A.6: Reaction data - 1step_C7H16.cti

```
#-------------------------------------------------------------------
#  Reaction data
#-------------------------------------------------------------------
# R1 - Single-step

# Sabourin, Boteler, and Hallett, 2016
reaction("C7H16 + 11 O2 => 7 CO2 + 8 H2O",
      [12069825.543005057, 0.0, 11572.6658801394],
       order="C7H16:1 O2:1" )
```

(Sabourin et al. 2016)

Again, the reaction equation species coefficients and the Arrhenius rate parameters cannot be changed within the input `(.cti)` file; however, in the continuous thermodynamics model they are also functions of the distribution mean θ, and require updating as the reaction progresses through the constant pressure reactor. The reaction equation species coefficients and Arrhenius rate parameters are used to initialize the Cantera reactor, the values Cantera uses can then be modified without changing the values within the input `(.cti)` file. The values of `[A,b,E]` shown in Listing A.6

are taken from correlations for the pre-exponential coefficient $A(\theta)$, and the activation energy $E(\theta)$, produced by Sabourin et al. (2016), for $\theta = 100.2040 \text{ kg/kmol}$ which represents n-heptane (C_7H_{16}).

Appendix B

Cantera Input Files `(.cti)`

Appendix B includes the chemical kinetic and thermodynamic Cantera input files for single, two and four-step reduced mechanisms. Note these mechanisms are not written in continuous thermodynamics form here within the `(.cti)` file, however they are used here to initialize the constant pressure reactor, and the relevant reaction steps are modified within the Cantera reactor program written in Python, shown in Appendix C.

B.1 Single-step Cantera input file (1Step_Reactor_CxHy.cti)

LISTING B.1: 1Step_Reactor_CxHy.cti

```
##########################################################################################################################
# 1step. cti
# Input  file   for  One−Step Reaction Mechanism
#########################################################################################################################

# N−Paraffin Generic CxHy Fuel but initiated as  n−heptane C7H16

units(length=’m’, quantity=’kmol’, time=’s’,  act_energy=’K’)

#−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
#   Phase data
#−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
ideal_gas(name = "1step",
      elements = " C  H  O  N ",
      species  =  """  CxHy O2  N2  CO2 H2O """,
      reactions  = " all " ,
       initial_state   = state (temperature = 300.0,
                              pressure = OneAtm) )

#−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
#   Species data
#−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
# Use one NASA parameterization for T < 1000 K, and another for
# T > 1000 K upto 3500.00 K.
# There are  seven(7)  coefficents  used for  molar heat capacity,  molar enthalpy and absolute molar entropy CFOX

species(name = "CxHy",
    atoms = " C:7   H:16",
    thermo=(NASA([300.00, 1391.00],
                  [−1.26836187E+00, 8.54355820E−02, −5.25346786E−05,
                    1.62945721E−08, −2.02394925E−12, −2.56586565E+04,
                    3.53732912E+01]),
              NASA([1391.00, 5000.00],
                  [ 2.22148969E+01, 3.47675750E−02, −1.18407129E−05,
                    1.83298478E−09, −1.06130266E−13, −3.42760081E+04,
                   −9.23040196E+01])),
        transport=gas_transport(geom=’nonlinear’,
                                diam=6.253,
                                well_depth=459.6,
                                rot_relax=1.0) ,
        note=’7/19/0THERM’)

species(name = "CO2",
    atoms = " C:1   O:2 ",
    thermo=(NASA([300.00, 1380.00],
                  [ 2.57930490E+00, 8.24684987E−03, −6.42716047E−06,
                    2.54637024E−09, −4.12030443E−13, −4.84162830E+04,
                    8.81141041E+00]),
              NASA([1380.00, 5000.00],
                  [ 5.18953018E+00, 2.06006476E−03, −7.33575324E−07,
                    1.17004374E−10, −6.91729215E−15, −4.93178953E+04,
                   −5.18289303E+00])),
        transport=gas_transport(geom=’linear’,
                                diam=3.763,
```

```
                                well_depth=244.0,
                                polar=2.65,
                                rot_relax=2.1),
        note= ’29/11/04’)

species(name = "H2O",
    atoms = " H:2  O:1 ",
    thermo=(NASA([300.00, 1000.00],
                 [ 3.38684200E+00, 3.47498200E−03, −6.35469600E−06,
                   6.96858100E−09, −2.50658800E−12, −3.02081100E+04,
                   2.59023300E+00]),
            NASA([1000.00, 5000.00],
                 [ 2.67214600E+00, 3.05629300E−03, −8.73026000E−07,
                   1.20099600E−10, −6.39161800E−15, −2.98992100E+04,
                   6.86281700E+00])),
        transport=gas_transport(geom=’nonlinear’,
                                diam=2.605,
                                well_depth=572.4,
                                dipole=1.844,
                                rot_relax=4.0),
        note= ’20387’)

species(name = "O2",
    atoms = " O:2 ",
    thermo=(NASA([300.00, 1000.00],
                 [ 3.21293600E+00, 1.12748600E−03, −5.75615000E−07,
                   1.31387700E−09, −8.76855400E−13, −1.00524900E+03,
                   6.03473800E+00]),
            NASA([1000.00, 5000.00],
                 [ 3.69757800E+00, 6.13519700E−04, −1.25884200E−07,
                   1.77528100E−11, −1.13643500E−15, −1.23393000E+03,
                   3.18916600E+00])),
        transport=gas_transport(geom=’linear’,
                                diam=3.458,
                                well_depth=107.4,
                                polar=1.6,
                                rot_relax=3.8),
        note= ’121386’)


species(name = "N2",
    atoms = " N:2 ",
    thermo=(NASA([300.00, 1000.00],
                 [ 3.29867700E+00, 1.40824000E−03, −3.96322200E−06,
                   5.64151500E−09, −2.44485500E−12, −1.02090000E+03,
                   3.95037200E+00]),
            NASA([1000.00, 5000.00],
                 [ 2.92664000E+00, 1.48797700E−03, −5.68476100E−07,
                   1.00970400E−10, −6.75335100E−15, −9.22797700E+02,
                   5.98052800E+00])),
        transport=gas_transport(geom=’linear’,
                                diam=3.621,
                                well_depth=97.53,
                                polar=1.76,
                                rot_relax=4.0),
        note= ’121286’)

#−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
#   Reaction Equation data
#−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
# R1 Single step
```

```
# A and E Fitted to detailed mech                        A=Keff       b           E/R= Eeff/R
# From Mehl et al. (2011)
reaction("CxHy + 11 O2 => 7 CO2 + 8 H2O", [71806117581.4224, 0.0,  19250.9674596462], order="CxHy:1 O2:1")

# With:
# ak 21.943
# ae 137.315
# bk 0.03048
# be 0.2240

# Commented out
# Sabourin et al. 2016                                        A                  b               E
#reaction("CxHy + 11 O2 => 7 CO2 + 8 H2O", [12081886.6752278, 0.0,  11579.2587472375], order="CxHy:1 O2:1")
# With:
# ak 13.253
# ae 73.529
# bk 0.03048
# be 0.2240
######################################################################################################################################
#                                                       End of Input File
######################################################################################################################################
```

B.2 Two-step Cantera input file (2Step_Reactor_CxHy.cti)

LISTING B.2: 2Step_Reactor_CxHy.cti

```
#############################################################################################
# 2step. cti
# Input file for Two−Step Reaction Mechanism
############################################################################################

units(length=’m’, quantity=’kmol’, time=’s’, act_energy=’K’)

#−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
#   Phase data
#−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
ideal_gas(name = "2step",
        elements = " C  H  O  N ",
        species  = """  CxHy O2 N2  CO CO2 H2O """,
        reactions  = " all ",
         initial_state   = state (temperature = 300.0,
                                 pressure = OneAtm) )

#−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
#   Species data
#−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−

# Use one NASA parameterization for T < 1000 K, and another for
# T > 1000 K upto 3500.00 K.
# There are seven(7) coefficents  used for  molar heat capacity,  molar enthalpy and absolute molar entropy CFOX

species(name = "CxHy",
     atoms = " C:7   H:16",
     thermo=(NASA([300.00, 1391.00],
                        [−1.26836187E+00, 8.54355820E−02, −5.25346786E−05,
                          1.62945721E−08, −2.02394925E−12, −2.56586565E+04,
                          3.53732912E+01]),
                 NASA([1391.00, 5000.00],
                        [ 2.22148969E+01, 3.47675750E−02, −1.18407129E−05,
                          1.83298478E−09, −1.06130266E−13, −3.42760081E+04,
                         −9.23040196E+01])),
          transport=gas_transport(geom=’nonlinear’,
                                 diam=6.253,
                                 well_depth=459.6,
                                 rot_relax=1.0) ,
          note=’7/19/0THERM’)

species(name= "CO",
          atoms= "C:1 O:1",
          thermo=(NASA([300.00, 1429.00],
                        [ 3.19036352E+00, 8.94419972E−04, −3.24927563E−08,
                         −1.04599967E−10, 2.41965693E−14, −1.42869054E+04,
                          5.33277914E+00]),
                 NASA([1429.00, 5000.00],
                        [ 3.11216890E+00, 1.15948283E−03, −3.38480362E−07,
                          4.41403098E−11, −2.12862228E−15, −1.42718539E+04,
                          5.71725177E+00])),
          transport=gas_transport(geom=’linear’,
                                 diam=3.65,
                                 well_depth=98.1,
```

```
                                        polar=1.95,
                                        rot_relax=1.8),
          note= ’29/11/04’)

species(name = "CO2",
     atoms = " C:1  O:2 ",
     thermo=(NASA([300.00, 1380.00],
                        [ 2.57930490E+00, 8.24684987E−03, −6.42716047E−06,
                          2.54637024E−09, −4.12030443E−13, −4.84162830E+04,
                          8.81141041E+00]),
                  NASA([1380.00, 5000.00],
                        [ 5.18953018E+00, 2.06006476E−03, −7.33575324E−07,
                          1.17004374E−10, −6.91729215E−15, −4.93178953E+04,
                         −5.18289303E+00])),
          transport=gas_transport(geom=’linear’,
                                        diam=3.763,
                                        well_depth=244.0,
                                        polar=2.65,
                                        rot_relax=2.1),
          note= ’29/11/04’)

species(name = "H2O",
     atoms = " H:2  O:1 ",
     thermo=(NASA([300.00, 1000.00],
                        [ 3.38684200E+00, 3.47498200E−03, −6.35469600E−06,
                          6.96858100E−09, −2.50658800E−12, −3.02081100E+04,
                          2.59023300E+00]),
                  NASA([1000.00, 5000.00],
                        [ 2.67214600E+00, 3.05629300E−03, −8.73026000E−07,
                          1.20099600E−10, −6.39161800E−15, −2.98992100E+04,
                          6.86281700E+00])),
          transport=gas_transport(geom=’nonlinear’,
                                        diam=2.605,
                                        well_depth=572.4,
                                        dipole=1.844,
                                        rot_relax=4.0),
          note= ’20387’)

species(name = "O2",
     atoms = " O:2 ",
     thermo=(NASA([300.00, 1000.00],
                        [ 3.21293600E+00, 1.12748600E−03, −5.75615000E−07,
                          1.31387700E−09, −8.76855400E−13, −1.00524900E+03,
                          6.03473800E+00]),
                  NASA([1000.00, 5000.00],
                        [ 3.69757800E+00, 6.13519700E−04, −1.25884200E−07,
                          1.77528100E−11, −1.13643500E−15, −1.23393000E+03,
                          3.18916600E+00])),
          transport=gas_transport(geom=’linear’,
                                        diam=3.458,
                                        well_depth=107.4,
                                        polar=1.6,
                                        rot_relax=3.8),
          note= ’121386’)

species(name = "N2",
     atoms = " N:2 ",
     thermo=(NASA([300.00, 1000.00],
                        [ 3.29867700E+00, 1.40824000E−03, −3.96322200E−06,
                          5.64151500E−09, −2.44485500E−12, −1.02090000E+03,
                          3.95037200E+00]),
```

```
            NASA([1000.00, 5000.00],
                 [ 2.92664000E+00, 1.48797700E−03, −5.68476100E−07,
                   1.00970400E−10, −6.75335100E−15, −9.22797700E+02,
                   5.98052800E+00])),
    transport=gas_transport(geom=’linear’,
                            diam=3.621,
                            well_depth=97.53,
                            polar=1.76,
                            rot_relax=4.0),
    note= ’121286’)

#−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
#   Reaction Equation data − Two Step Mechanism
#−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−

# Two Step mechanism with reversible CO−CO2 oxidation

# Note below use different units  units(length=’m’, quantity=’kmol’, time=’s’,  act_energy=’K’)

# CO−CO2 Oxidation from Cooper and Hallett (2000, p4454) based on Howard and Fine (1973)
# R1
reaction("CxHy + 7.5 O2 => 7 CO + 8 H2O", [715866079606.9368,0,18957.5028950534], order="CxHy:1 O2:1")

# With
# ae = 134.875
# ak = 21.998
# be = 0.227
# bk = 0.03048

# R2f Two step
reaction("CO + 0.5 O2 => CO2",  [1.3e+11,  0.0,  15105],  order="CO:1 O2:0.5 H2O:0.5", options=[’nonreactant_orders’])

# R2r Two step
reaction("CO2 => CO + 0.5 O2",  [7.5e+11,  0.0,  46500],  order="CO2:1")
##################################################################################################
# Commented Out
# CO−CO2 Oxidation from Westbrook and Dryer (1981)
# R1
#reaction("CxHy + 7.5 O2 => 7 CO + 8 H2O", [57568162211905.3,0,37.6724281385], order="CxHy:1 O2:1")

# R1 Fitted  With
# ae = 134.875
# ak = 21.722
# be = 0.227
# bk = 0.03048

# R2f Two step
#reaction("CO + 0.5 O2 => CO2",  [3.98107e14,  0.0,  40],  order="CO:1 H2O:0.5 O2:0.25", options=[’nonreactant_orders’])

# R2r Two step
#reaction("CO2 => CO + 0.5 O2",  [5.0e8,  0.0,  40],  order="CO2:1")
###############################################################################################
#                                      End of Input File
###############################################################################################
```

B.3 Four-step Cantera input file (4Step_Reactor_CxHy.cti)

LISTING B.3: 4Step_Reactor_CxHy.cti

```
########################################################################################################
# 4step. cti
# Input  file  for  Four−Step Reaction Mechanism
#######################################################################################################

# N−Paraffin Generic CxHy Fuel but initiated as  n−heptane C7H16

units(length='cm', quantity='mol', time='s',  act_energy='K')

#−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
#   Phase data
#−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
ideal_gas(name = "4step",
      elements = " C  H  O  N ",
      species  =  """  CxHy O2  N2 CO2 H2O X I """,
      reactions  = " all ",
       initial_state   = state (temperature = 300.0,
                          pressure = OneAtm) )

#−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
#   Species  data
#−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
# Use one NASA parameterization for T < 1000 K, and another for
# T > 1000 K upto 3500.00 K.
# There are  seven(7)  coefficents  used for  molar heat capacity,  molar enthalpy and absolute molar entropy CFOX

species(name = "CxHy",
    atoms = " C:7   H:16",
    thermo=(NASA([300.00, 1391.00],
                     [−1.26836187E+00, 8.54355820E−02, −5.25346786E−05,
                       1.62945721E−08, −2.02394925E−12, −2.56586565E+04,
                       3.53732912E+01]),
                NASA([1391.00, 5000.00],
                     [ 2.22148969E+01,  3.47675750E−02, −1.18407129E−05,
                       1.83298478E−09, −1.06130266E−13, −3.42760081E+04,
                      −9.23040196E+01])),
        transport=gas_transport(geom='nonlinear',
                                diam=6.253,
                                well_depth=459.6,
                                rot_relax=1.0) ,
        note='7/19/0THERM')

species(name = "CO2",
    atoms = " C:1   O:2 ",
    thermo=(NASA([300.00, 1380.00],
                     [ 2.57930490E+00,  8.24684987E−03, −6.42716047E−06,
                       2.54637024E−09, −4.12030443E−13, −4.84162830E+04,
                       8.81141041E+00]),
                NASA([1380.00, 5000.00],
                     [ 5.18953018E+00,  2.06006476E−03, −7.33575324E−07,
                       1.17004374E−10, −6.91729215E−15, −4.93178953E+04,
                      −5.18289303E+00])),
        transport=gas_transport(geom='linear',
                                diam=3.763,
```

```
                                well_depth=244.0,
                                polar=2.65,
                                rot_relax=2.1),
        note= '29/11/04')

species(name = "H2O",
    atoms = " H:2  O:1 ",
    thermo=(NASA([300.00, 1000.00],
                 [ 3.38684200E+00, 3.47498200E−03, −6.35469600E−06,
                   6.96858100E−09, −2.50658800E−12, −3.02081100E+04,
                   2.59023300E+00]),
            NASA([1000.00, 5000.00],
                 [ 2.67214600E+00, 3.05629300E−03, −8.73026000E−07,
                   1.20099600E−10, −6.39161800E−15, −2.98992100E+04,
                   6.86281700E+00])),
        transport=gas_transport(geom='nonlinear',
                                diam=2.605,
                                well_depth=572.4,
                                dipole=1.844,
                                rot_relax=4.0),
        note= '20387')

species(name = "O2",
    atoms = " O:2 ",
    thermo=(NASA([300.00, 1000.00],
                 [ 3.21293600E+00, 1.12748600E−03, −5.75615000E−07,
                   1.31387700E−09, −8.76855400E−13, −1.00524900E+03,
                   6.03473800E+00]),
            NASA([1000.00, 5000.00],
                 [ 3.69757800E+00, 6.13519700E−04, −1.25884200E−07,
                   1.77528100E−11, −1.13643500E−15, −1.23393000E+03,
                   3.18916600E+00])),
        transport=gas_transport(geom='linear',
                                diam=3.458,
                                well_depth=107.4,
                                polar=1.6,
                                rot_relax=3.8),
        note= '121386')

species(name = "N2",
    atoms = " N:2 ",
    thermo=(NASA([300.00, 1000.00],
                 [ 3.29867700E+00, 1.40824000E−03, −3.96322200E−06,
                   5.64151500E−09, −2.44485500E−12, −1.02090000E+03,
                   3.95037200E+00]),
            NASA([1000.00, 5000.00],
                 [ 2.92664000E+00, 1.48797700E−03, −5.68476100E−07,
                   1.00970400E−10, −6.75335100E−15, −9.22797700E+02,
                   5.98052800E+00])),
        transport=gas_transport(geom='linear',
                                diam=3.621,
                                well_depth=97.53,
                                polar=1.76,
                                rot_relax=4.0),
        note= '121286')

########################################### below added for 4step Made up NASA Species for X and I

species(name="X",
        atoms= "C:7  H:16",
        thermo=(NASA9([200.00, 1000.00],
```

```
                    [ −3.778568421E+05, 8.174087167E+03, −4.559343594E+01,
                      2.124132427E−01, −2.508981809E−04, 1.638963875E−07,
                      −4.413859269E−11, 2.105598713E+04, 3.477892929E+02]),
               NASA9([1000.00, 6000.00],
                    [ 1.298714042E+07, −5.058182733E+04, 8.835374302E+01,
                     −8.710536021E−03, 1.573780467E−06, −1.528145976E−10,
                      6.165154740E−15, 3.649332021E+05, −4.910237796E+02]) ) )

species(name="I",
        atoms= "C:7  H:16  O:4",
        thermo=(NASA([300, 1630],
                    [ 6.285282710E+00, 9.100857497E−02, −5.905027386E−05,
                      1.966186280E−08, −2.662521410E−12, −8.296147480E+04,
                      2.733420836E+01]),
               NASA([1630, 3500],
                    [ 2.506781673E+01, 4.622846467E−02, −1.902483971E−05,
                     3.765918903E−09, −2.958251483E−13, −8.926245570E+04,
                      −7.299751213E+01]) ) )

#−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
#   Reaction Equation data 4−Steps − n−heptane
#   Note there is  no CO
#−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
# Muller et  al.  (1992)
reaction("CxHy => X",                 [(1e10,’1/s’),  0.0,  21650], order="CxHy:1")               # R1 HT
reaction("X + 11  O2 => 7 CO2 + 8 H2O", [(2e12,  ’cm3/mol/s’), 0.0, 7216.666667], order="X:1 O2:1") # R2 HT
reaction("CxHy + 2 O2 => I",          [(3e18,  ’cm3/mol/s’), 0.0, 21650], order="CxHy:1 O2:1")  # R3f LT
reaction("I  => CxHy + 2 O2",         [(4e22,  ’1/s’),  0.0,  37285], order="I:1")              # R3b LT
reaction("I  + 9  O2 => 7 CO2 + 8 H2O", [(5e13,  ’cm3/mol/s’), 0.0, 13230], order="I:1  O2:1")     # R4 LT

# Below commented out
# Muller et  al.  (1992) # units  not  specified
#reaction("CxHy => X",                  [1e10,  0.0,  21650], order="CxHy:1")             # R1 HT
#reaction("X + 11 O2 => 7 CO2 + 8 H2O", [2e12,  0.0,  7216.666667], order="X:1 O2:1")     # R2 HT
#reaction("CxHy + 2 O2 => I",           [3e18,  0.0,  21650], order="CxHy:1 O2:1")        # R3f LT
#reaction("I  => CxHy + 2 O2",          [4e22,  0.0,  37285], order="I:1")                # R3b LT
#reaction("I  + 9  O2 => 7 CO2 + 8 H2O", [5e13,  0.0,  13230], order="I:1  O2:1")         # R4 LT

# Muller et  al.  (1992)  Best  Fit,  Modified R1 and R3f reactions.
#reaction("CxHy => X",                  [(3.5e9,’1/s’),  0.0,  18875], order="CxHy:1") # modified
#reaction("X + 11 O2 => 7 CO2 + 8 H2O", [(2e12,  ’cm3/mol/s’), 0.0, 7217], order="X:1 O2:1")
#reaction("CxHy + 2 O2 => I",           [(8.5e16,  ’cm3/mol/s’), 0.0, 18875], order="CxHy:1 O2:1") # modified
#reaction("I  => CxHy + 2 O2",          [(4e22,  ’1/s’),  0.0,  37285], order="I:1")
#reaction("I  + 9  O2 => 7 CO2 + 8 H2O", [(5e13,  ’cm3/mol/s’), 0.0, 13230], order="I:1  O2:1")

####################################################################################################
#                                        End of Input File
####################################################################################################
```

Appendix C

Constant Pressure Reactor Program Files `(.py)`

Appendix C includes the reactor programs written in Python for single, two and four-step reactors in continuous thermodynamics form. A non-continuous reactor is also presented here for the single step mechanism only.

C.1 Single-step Reactor Program (1Step_Reactor_CxHy.py)

C.1.1 Single-step With Continuous Thermodynamics

LISTING C.1: 1Step_Reactor_CxHy.py - with CT

```
####################################################################################################
"""
C.D.L.Fox & Dr.W.L.H.Hallett

Single−Step Mechanism 1Step_Reactor_CxHy.py

Default Units:  length m, mass kg, specify  number of molecules kmol, time s, energy J,  act_energy K, pressure Pa.

"""
####################################################################################################
import cantera as  ct
import math
import csv

####################################################################################################

ct.add_directory(’/home/foxc/droplet/input_cti’)

gas         = ct.Solution(’1step_CxHy.cti’)

phi         = 1.0                       # equivalence ratio
x           = gas.n_atoms(’CxHy’, ’C’) # of  C atoms in fuel
y           = gas.n_atoms(’CxHy’, ’H’) # of  H atoms in fuel
a           = x + y/4
P           = ct.one_atm
T_i         = 900                       # Set   Initial   Temp
gas.TPX     = T_i,  P,  {’CxHy’: 1,  ’O2’:  a/phi,  ’N2’:  a∗3.76/phi}  # set  temp, pressure,  fuel  mixture stoichiometry

####################################################################################################
# Get  initial   mole fractions  from cantera
# Single−step Mechanism only
Xf          = gas.X[0]                   # mole fraction  of  fuel
Xo          = gas.X[1]                   # mole  fraction  of  oxygen
X_N2        = gas.X[2]                   # mole fraction  of  N2
X_CO2       = gas.X[3]                   # mole  fraction  of  CO2
X_H2O       = gas.X[4]                   # mole fraction  of  H2O

####################################################################################################

# Get  initial   gas properties  and theta  from cantera
C           = gas.density_mole           # default  density  [kmol/m^3]
M           = gas.mean_molecular_weight # mix molar mass default [kg/kmol]
R           = (ct.gas_constant/10∗∗6)   # default  R [J/kmol−K] convert to [kJ/gmol−K]
RT          = R∗gas.T                    #  [kJ/gmol]
T           = gas.T                      #  starts   at   initial   temp
theta       = gas.molecular_weights[0]  # fuel  n−heptane = 100.2 [kg/kmol]

sigma       = 2                          # User can set  sigma here
```

```
# Calculate  initial  Psi,  alpha, beta,  and Keff and Eeff
Psi            = theta∗∗2 + sigma∗∗2
alpha          = (theta∗∗2)/(sigma∗∗2)
beta           = (sigma∗∗2)/theta
gamma          = 0                       # orgin of the distribution

ak             = 21.943                  # unit  less  Fitted  to  Mehl et al.  2011
bk             = 0.03048                 # unit  less  (Sabourin et al.,  2016)
ae             = 137.315                 # [kJ/mol]  Fitted  to  Mehl et al.  2011
be             = 0.2270                  # unit  less  (Sabourin et al.,  2016)

Ae             = ae + gamma∗be           # constant
Ak             = ak + gamma∗bk           # constant
Aa1            = ak − ae/RT              # Aa1 to manually calculate W_bar
Aa2            = − bk + be/RT + 1/beta
Keff           = math.exp(ak)∗(1/((beta∗Aa2)∗∗alpha)∗math.exp(be∗alpha/(Aa2∗RT)))

dt             = 0                       #  Initialize   for  lower while loop

# Calculate Arrhenius {A, b,  E},  default  b = 0
A              = Keff          # Sabourin et al.  2016 expression A = K(theta) = math.exp(ak + bk∗theta) # [1/s]
Eeff           = ae + be∗alpha/Aa2 # [kJ/gmol]

##################################################################################################
# Get  initial   W_bar from cantera and calculate initial  twiddle and hat
W_bar          = − gas.forward_rates_of_progress[0] # default  [kmol/m^3/s] Note the negative
W_twiddle      = W_bar∗(alpha/Aa2)                  # needed to calculate  theta
W_hat          = W_bar∗(alpha∗(alpha + 1)/(Aa2∗∗2)) # needed to calculate Psi

##################################################################################################

# Get reaction  from Cantera − Inport R1 from CTI file
# Reaction 1
R1             = gas.reaction(0)
A1             = R1.rate.pre_exponential_factor     # default  [kmol/m3−s]
b1             = R1.rate.temperature_exponent       # unit−less
E1             = R1.rate.activation_energy          # default  [J/Kmol]

# Modify Reaction R1 − Single−step mechanism
R1.reactants   = {'CxHy': 1, 'O2':  (3∗theta+7.979)/28.054}
R1.products    = {'CO2': (theta−2.016)/14.027, 'H2O': (theta+12.011)/14.027}
A_1            = A                        #  [kmol/m3−s] needs to be in those units
b_1            = b1
E_1            = Eeff∗10∗∗6               # convert from [kJ/gmol] to [J/Kmol]. Needed for Cantera
R1.rate        = ct.Arrhenius(A_1, b_1, E_1)
gas.modify_reaction(0, R1)                # update first  reaction  equation '0'

##################################################################################################

# Set  NASA "FUEL" species polynomial coefficients for specific heat  [Cp^o], Enthalpy [h^o] and entropy [s^0].
A00 =  2.473e−01        # (Hallett,  2017)
A01 =  1.092e−02        # (Hallett,  2017)
A02 = −3.124e−06        # (Hallett,  2017)
A03 = −3.757e−11        # (Hallett,  2017)
A04 =  0                # (Hallett,  2017)
A05 = −5420.4379217007 # Curve Fit Values (Fox and Hallett, 2017)
A06 =  12.9312135308    # Curve Fit Values (Fox and Hallett,  2017)

B00 = −8.328e−03        # (Hallett,  2017)
B01 =  8.704e−04        # (Hallett,  2017)
```

```
B02 = −4.984e−07       # (Hallett, 2017)
B03 =  1.090e−10       # (Hallett, 2017)
B04 =  0               # (Hallett, 2017)
B05 = −208.6660658572 # Curve Fit Values (Fox and Hallett, 2017)
B06 =  0.14860092      # Curve Fit Values (Fox and Hallett, 2017)

fuel_coeffs  = [2350,
               (A00+B00∗theta), (A01+B01∗theta), (A02+B02∗theta),
               (A03+B03∗theta), (A04+B04∗theta), (A05+B05∗theta),
               (A06+B06∗theta), (A00+B00∗theta), (A01+B01∗theta),
               (A02+B02∗theta), (A03+B03∗theta),(A04+B04∗theta),
               (A05+B05∗theta), (A06+B06∗theta) ]

# Modify NASA polynomial coefficients imported from .cti file.
S1 = gas.species(0)
S1.thermo = ct.NasaPoly2(300, 5000, ct.one_atm, fuel_coeffs)
gas.modify_species(0, S1) # update first species coeffs

###############################################################################################

# Build Reactor r
r           = ct.IdealGasConstPressureReactor(gas)  # ideal gas, constant pressure reactor filled with gas
tnow          = 0                                   # start time [s]
 tfinal       = 0.45                                # End after  Time [s]
 states       = ct.SolutionArray(gas, extra=[’t’])  # array to hold solution of gas (used for plotting)
sim           = ct.ReactorNet([r])                  # prepare the simulation with a ReactorNet object

###############################################################################################

# print to screen a header for output values
print(’%10s %10s %10s %10s %10s %10s %10s %10s %10s %10s %10s %10s %10s %10s %10s %10s %10s %10s %10s %10s %10s’
      ’%10s %10s %10s’%
    (’t’,’T’,’R’,’P’,’C’,’M’,’Xf’,’Xo’,’Xn2’,’Xco2’,’Xh2o’,’Aa1’,’Aa2’,’alpha’,’beta’,
     ’W_bar’,’W_twiddle’, ’W_hat’,’theta’,’Psi’,’sigma’, ’A1’, ’E1’, ’R1’))

# Define storage arrays (Use Capitals to keep tract) to hold data (Only used for plotting with matplotlib.pylab)
Temp            = []  # Temperature [K]
Temp_invers     = []  # Inverse Temperature [1/K]
TIME            = []  # Time [s]
THETA           = []
PSI             = []
SIGMA           = []
WBAR            = []  # [kmol/m^3/s]
KEFF            = []  # [1/s]
EEFF            = []  # [kJ/gmol]

###############################################################################################

# Write output to CSV file for importing into Excel
 csv_file  = ’1Step_Reactor_CxHy.csv’

with open(csv_file, ’w’) as outfile:
    writer = csv.writer(outfile)
    writer.writerow([’t [s] c1’,’T [K] c1’, ’C [kmol/m^3] c1’,’M [kg/kmol] c1’,
                     ’Xf c1’,’Xo c1’,’Xn2 c1’,’Xco2 c1’,’Xh2o c1’, ’Aa1 c1’,’Aa2 c1’,’alpha c1’,’beta c1’,
                     ’W_bar [kmol/m^3/s] c1’, ’W_twiddle [kmol/m^3/s] c1’, ’W_hat [kmol/m^3/s] c1’,’theta c1’,’Psi
                          c1’,’sigma c1’,
                     ’pre exponential: Keff [kmol/m3−s] c1’, ’activation energy: Eeff [J/Kmol] c1’, ’R1.equation c1’])

    # Begin while loop (note its nested within a "with open" statement)
    while tnow < tfinal:   # Note, in first step, initial values are displayed
```

```
if   850 <= T <= 850.01  or  875  <= T <= 875.01  or  900  <= T <= 900.01  or  925  <= T <= 925.01  or  \
     950 <= T <= 950.01  or  975  <= T <= 975.01  or  1000 <= T <= 1000.01 or  1025 <= T <= 1025.01 or  \
    1050 <= T <= 1050.01 or  1075 <= T <= 1075.01 or  1100 <= T <= 1100.01 or  1125 <= T <= 1125.01 or  \
    1150 <= T <= 1150.01 or  1175 <= T <= 1175.01 or  1200 <= T <= 1200.01 or  1225 <= T <= 1225.01 or  \
    1250 <= T <= 1250.01 or  1275 <= T <= 1275.01 or  1300 <= T <= 1300.01 or  1325 <= T <= 1325.01 or  \
    1350 <= T <= 1350.01 or  1375 <= T <= 1375.01 or  1400 <= T <= 1400.01 or  1425 <= T <= 1425.01 or  \
    1450 <= T <= 1450.01 or  1475 <= T <= 1475.01 or  1500 <= T <= 1500.01 or  1525 <= T <= 1525.01 or  \
    1550 <= T <= 1550.01 or  1575 <= T <= 1575.01 or  1600 <= T <= 1600.01 or  1625 <= T <= 1625.01 or  \
    1650 <= T <= 1650.01 or  1675 <= T <= 1675.01 or  1700 <= T <= 1700.01 or  1725 <= T <= 1725.01 or  \
    1750 <= T <= 1750.01 or  1775 <= T <= 1775.01 or  1800 <= T <= 1800.01 or  1825 <= T <= 1825.01 or  \
    1850 <= T <= 1850.01 or  1875 <= T <= 1875.01 or  1900 <= T <= 1900.01 or  1925 <= T <= 1925.01 or  \
    1950 <= T <= 1950.01 or  1975 <= T <= 1975.01 or  2000 <= T <= 2000.01 or  2025 <= T <= 2025.01 or  \
    2050 <= T <= 2050.01 or  2075 <= T <= 2075.01 or  2100 <= T <= 2100.01 or  2125 <= T <= 2125.01 or  \
    2150 <= T <= 2150.01 or  2175 <= T <= 2175.01 or  2200 <= T <= 2200.01 or  2225 <= T <= 2225.01 or  \
    2250 <= T <= 2250.01 or  2275 <= T <= 2275.01 or  2300 <= T <= 2300.01 or  2325 <= T <= 2325.01 or  \
    2350 <= T <= 2350.01 or  2375 <= T <= 2375.01 or  2400 <= T <= 2400.01 or  2425 <= T <= 2425.01 or  \
    2450 <= T <= 2450.01 or  2475 <= T <= 2475.01 or  2500 <= T <= 2500.01 or  2525 <= T <= 2525.01 or  \
    2550 <= T <= 2550.01 or  2575 <= T <= 2575.01 or  2600 <= T <= 2600.01 or  2625 <= T <= 2625.01 or  \
    2650 <= T <= 2650.01 or  2675 <= T <= 2675.01 or  2700 <= T <= 2700.01 or  2725 <= T <= 2725.01 or  \
    2750 <= T <= 2750.01 or  2775 <= T <= 2775.01 or  2800 <= T <= 2800.01 or  2825 <= T <= 2825.01 or  \
    2850 <= T <= 2850.01 or  2875 <= T <= 2875.01 or  2880 <= T:


     ''' <= 2880.01 or  2882 <= T <= 2882.01 or  \
    2884 <= T <= 2884.01 or 2886 <= T <= 2886.01 or 2888 <= T <= 2888.01 or 2890 <= T <= 2890.01 or \
    2892 <= T <= 2892.01 or 2894 <= T <= 2894.01 or 2896 <= T <= 2896.01 or 2898 <= T <= 2898.01 or \
    2899 <= T <= 2899.01 or 2899.1  <= T: '''


    # output to screen
    print('%10.1e %10.2f %10.3e %10.0f %10.5f %10.5f %10.5f %10.5f %10.5f %10.5f %10.5f %10.2f %10.2f %10.2f %10.2f'
          '%10.1e %10.1e %10.1e %10.2f %10.2f %10.2f %10s %10s %10s'%
           (sim.time,  T,  R,  P,  C,  M, Xf,  Xo,  X_N2,  X_CO2,X_H2O,  Aa1,  Aa2,  alpha,  beta,  W_bar,
            W_twiddle, W_hat, theta, Psi, sigma,
            R1.rate.pre_exponential_factor,  R1.rate.activation_energy,  R1.equation))

    # write output to CSV
    writer.writerow([sim.time, T,  C,  M, Xf,  Xo, X_N2, X_CO2,X_H2O, Aa1, Aa2, alpha, beta,
                     W_bar, W_twiddle, W_hat, theta, Psi, sigma,
                     R1.rate.pre_exponential_factor, R1.rate.activation_energy, R1.equation])

sim. reinitialize  ()                  # Comment out to speed up, but CT results are no good
t1       = sim.time
tnow     = sim.step()                  # take  single  steps  towards  tfinal
states .append(r.thermo.state, t=tnow) # update the array of  states  at  tnow (for  plotting  to  matplotlib.pylab)
t2       = sim.time
dt       = t2-t1                       # Need delta t  for  computations (Should it be zero at  step  1?)

# Get properties  from cantera:  W_bar, Xf_new, T, C, M_mix
W_bar          = - gas.forward_rates_of_progress[0] # default  [kmol/m^3/s] Note the negative
W_twiddle      = W_bar*(alpha/Aa2)                  # needed to calculate  theta
W_hat          = W_bar*(alpha*(alpha + 1)/(Aa2**2)) # needed to calculate Psi
Xf_new         = r.thermo.X[0]                      # mole fraction  of  fuel
T_new          = r.T                                # temp [K]
dT             = T_new - T                          # change in temp [k]
C_new          = gas.density_mole                   # update default molar density [kmol/m^3]
M_new          = gas.mean_molecular_weight          # molecular mass [kg/kmol]
del_C          = C_new - C                          # change in molar density [kmol/m^3]
del_M          = M_new - M                          # change in mixture molecular mass [kg/kmol]
```

```
    # Convection expansion term
    del_Cvdt        = − (C/M)∗del_M − del_C          # use old values (M,C,T)

    # Calculate theta
    RHS_theta       = C∗Xf∗theta + W_twiddle∗dt − Xf∗theta∗del_Cvdt
    theta_new       = RHS_theta/C_new/Xf_new

    # Calculate Psi
    RHS_psi         = C∗Xf∗Psi + W_hat∗dt − Xf∗Psi∗del_Cvdt
    Psi_new         = RHS_psi/(C_new∗Xf_new)

    # Calculate sigma
    x_root          = Psi_new − (theta_new∗∗2)
    sigma_new       = math.sqrt(abs(x_root))     # careful remove abs(), then this breaks if root is negative


    # used for making plots (matplotlib.pylab)
    TIME.append(float(sim.time))
    Temp_invers.append(float(1/r.T))
    WBAR.append(float(gas.forward_rates_of_progress[0]))
    THETA.append(float(theta))
    SIGMA.append(float(sigma))

################################################################################################################
    # Update for next run of loop

    # Single−step Mechanism only
    Xf              = gas.X[0]  # mole fraction of fuel
    Xo              = gas.X[1]  # mole fraction of O2
    X_N2            = gas.X[2]  # mole fraction of N2
    X_CO2           = gas.X[3]  # mole  fraction  of CO2
    X_H2O           = gas.X[4]  # mole fraction of H2O

    T               = r.T
    R               = (ct.gas_constant/10 ∗∗ 6)  # default R [J/kmol−K] convert to [kJ/gmol−K]
    P               = gas.P                      # [pa]
    C               = gas.density_mole # default density [kmol/m^3]
    M               = gas.mean_molecular_weight # mixture molar mass default [kg/kmol]
    theta           = theta_new
    Psi             = Psi_new
    sigma           = sigma_new
    alpha           = (theta∗∗2)/(sigma∗∗2) # theta/beta
    beta            = (sigma∗∗2)/theta
    Aa1             = ak − ae/RT            # Aa1 to manually calculate W_bar
    Aa2             = − bk + be/RT + 1/beta
    Keff            = math.exp(ak)∗(1/((beta∗Aa2)∗∗alpha)∗math.exp(be∗alpha/(Aa2∗RT)))

    # Calculate Arrhenius {A, b, E}, default b = 0
    A              = Keff                # [1/s]
    Eeff           = ae + be∗alpha/Aa2 # [kJ/gmol]

    # Modify Reaction equations
    # Reaction 1 (Single step only !!!)
    R1.reactants   = {’CxHy’: 1, ’O2’: (3∗theta + 7.979)/28.054}
    R1.products    = {’CO2’: (theta − 2.016)/14.027, ’H2O’: (theta + 12.011)/14.027}
    A_1            = A           # [kmol/m3−s] needs to be in those units
    b_1            = b1
    E_1            = Eeff∗10∗∗6  # convert from [kJ/gmol] to [J/Kmol]. Needed for Cantera
    R1.rate        = ct.Arrhenius(A_1, b_1, E_1)
    gas.modify_reaction(0, R1)  # update first reaction equation ’0’
```

```
        # Modify "Fuel" species NASA coefficients
        fuel_coeffs  = [2350,
            (A00+B00*theta), (A01+B01*theta), (A02+B02*theta),
            (A03+B03*theta), (A04+B04*theta), (A05+B05*theta),
            (A06+B06*theta), (A00+B00*theta), (A01+B01*theta),
            (A02+B02*theta), (A03+B03*theta),(A04+B04*theta),
            (A05+B05*theta), (A06+B06*theta) ]

        S1 = gas.species(0)
        S1.thermo = ct.NasaPoly2(300, 5000, ct.one_atm, fuel_coeffs)
        gas.modify_species(0, S1)  # update first species coeffs

##########################################################################################################
#                                          THE END OF PROGRAM
##########################################################################################################
```

C.1.2 Single-step Without Continuous Thermodynamics

LISTING C.2: 1Step_Reactor_n-heptane_no_CT.py

```
"""
Constant-pressure, adiabatic kinetics simulation.
"""


import cantera as ct
import csv

ct.add_directory('/home/foxc/droplet/input_cti') # use to find input .cti file in other than current directory

gas         = ct.Solution('1step_CxHy.cti')    # load chemical kinetic reaction mechanism into cantera (.cti)

phi         = 1.0                          # equivalence ratio
x           = gas.n_atoms('CxHy', 'C') # of C atoms in fuel
y           = gas.n_atoms('CxHy', 'H') # of H atoms in fuel
a           = x + y/4

P                   = ct.one_atm            # [pa]
initial_composition = {'CxHy': 1, 'O2': a/phi, 'N2': a*3.76/phi}
gas.TPX             = 900, P, initial_composition  # Initial temp > 850K for results to be accurate

Xf          = gas.X[gas.species_index('CxHy')]   # mole fraction of fuel
Xo          = gas.X[gas.species_index('O2')]     # mole fraction of oxygen
X_N2        = gas.X[gas.species_index('N2')]     # mole fraction of N2
X_CO2       = gas.X[gas.species_index('CO2')]    # mole fraction of CO2
X_H2O       = gas.X[gas.species_index('H2O')]    # mole fraction of H2O

# Get initial gas properties and theta from cantera
C           = gas.density_mole              # default density [kmol/m^3]
M           = gas.mean_molecular_weight # mix molar mass default [kg/kmol]
R           = (ct.gas_constant/10**6)     # default R [J/kmol-K] convert to [kJ/gmol-K]
RT          = R*gas.T                       # [kJ/gmol]
T           = gas.T                         # starts at initial temp
W_bar       = - gas.forward_rates_of_progress[gas.species_index('CxHy')] # default [kmol/m^3/s] Note the negative

# Get reaction from Cantera - Inport R1 from CTI file
# Reaction 1
R1          = gas.reaction(0)
A1          = R1.rate.pre_exponential_factor     # default [kmol/m3-s]
b1          = R1.rate.temperature_exponent       # unit-less
E1          = R1.rate.activation_energy          # default [J/Kmol]

###########################################################################################################

# Build Reactor r
r           = ct.IdealGasConstPressureReactor(gas) # ideal gas, constant pressure reactor filled with gas
tnow        = 0                                    # start time [s]
 tfinal     = 0.45                                 # end time [s]
 states     = ct.SolutionArray(gas, extra=['t'])   # array to hold solution (only use if plotting)

sim         = ct.ReactorNet([r])                   # prepare the simulation with a ReactorNet object
sim.set_initial_time(0)
#sim.set_max_time_step(1)                          # Seems not to be needed
```

```
##########################################################################################################
dt              = 0                                 #  Initialize   for  lower while loop
t1              = 0
t2              = 0

print(’%10s %10s %10s %10s %10s %10s %10s %10s %10s %10s’ % (’t’,’T’,’C’,’M’,’Xf’,’Xo’,’Xn2’,’Xco2’,’Xh2o’, ’W_bar’))

# Write output to CSV file  for  importing into excel  etc ...
 csv_file  = ’1Step_Reactor_n−heptane_no_CT.csv’

with open(csv_file,  ’w’) as  outfile :
    writer  = csv.writer( outfile )
    writer.writerow([’t  [s]’ , ’T [K]’,  ’C [kmol/m^3]’,’M [kg/kmol]’,
                    ’Xf  [Fule]’ , ’Xo [O2]’, ’Xn2 [N2]’,’Xco2 [CO2]’,’Xh2o [H2O]’, ’W_bar [kmol/m^3/s]’,
                    ’pre  exponential:  Keff  [kmol/m3−s]’, ’activation energy: Eeff  [J/Kmol]’])

    while tnow < tfinal :   # Used to "Sample" the date to  reduce output  file  size

        # output to screen
        print( ’%10.1e %10.2f %10.3e %10.0f %10.5f %10.5f %10.5f %10.5f %10.5f %10.5f’ %
             (sim.time,  T,  C,  M, Xf,  Xo, X_N2, X_CO2, X_H2O, W_bar))

        # write output to CSV
        writer.writerow([sim.time, T,  C,  M, Xf,  Xo, X_N2, X_CO2, X_H2O, W_bar, A1, E1])

        # sim. reinitialize  ()    # Not required for  single  step
        tnow = sim.step()     # take  single  steps  towards  tfinal
         states .append(r.thermo.state, t=tnow) # update the array of  states   at  tnow (for  plotting   to  matplotlib.pylab)

        # Update
        Xf = gas.X[gas.species_index(’CxHy’)]  # mole fraction  of  fuel
        Xo = gas.X[gas.species_index(’O2’)]    # mole   fraction   of  oxygen
        X_N2 = gas.X[gas.species_index(’N2’)]  # mole fraction  of  N2
        X_CO2 = gas.X[gas.species_index(’CO2’)] # mole  fraction   of  CO2
        X_H2O = gas.X[gas.species_index(’H2O’)] # mole fraction of  H2O

        T = gas.T
        R = (ct.gas_constant / 10 ∗∗ 6)    # default  R [J/kmol−K] convert to [kJ/gmol−K]
        P = gas.P   # [pa]
        C = gas.density_mole # default  density  [kmol/m^3]
        M = gas.mean_molecular_weight # mixture molar mass default [kg/kmol]

        # In  this  case  we do this  just   to  show the following do not change
        R1 = gas.reaction (0)
        A1 = R1.rate.pre_exponential_factor  # default  [kmol/m3−s]
        b1 = R1.rate.temperature_exponent # unit−less
        E1 = R1.rate.activation_energy   # default  [J/Kmol]

##########################################################################################################
#                                                    THE END OF PROGRAM
##########################################################################################################
```

C.2 Two-step Reactor Program (2Step_Reactor_CxHy.py)

With Continuous Thermodynamics

LISTING C.3: 2Step_Reactor_CxHy.py - with CT

```
##############################################################################################################
"""
C.D.L.Fox & Dr.W.L.H.Hallett (2017)

Two−Step Mechanism 2Step_Reactor_CxHy.py

This Program prints the results to the screen, stores data in a CSV file, and can plot graphs

Default Units: length m, mass kg, specify number of molecules kmol, time s, energy J, act_energy K, pressure Pa.
"""
##############################################################################################################
import cantera as ct
import math
import csv
##############################################################################################################

ct.add_directory('/home/foxc/droplet/test/reactors/reactor1/input_cti')

gas         = ct.Solution('2step_CxHy.cti')

phi         = 1.0                         # equivalence ratio
x           = gas.n_atoms('CxHy', 'C') # of C atoms in fuel
y           = gas.n_atoms('CxHy', 'H') # of H atoms in fuel
a           = x + y/4
P           = ct.one_atm
T_i         = 900                         # Set Initial Temp
gas.TPX     = T_i, P, {'CxHy': 1, 'O2': a/phi, 'N2': a*3.76/phi}  # set temp, pressure, fuel mixture stoichiometry

##############################################################################################################
# Get initial mole fractions from cantera
Xf          = gas.X[gas.species_index('CxHy')]  # mole fraction of fuel
Xo          = gas.X[gas.species_index('O2')]    # mole fraction of oxygen
X_N2        = gas.X[gas.species_index('N2')]    # mole fraction of N2
X_CO        = gas.X[gas.species_index('CO')]    # mole fraction of CO
X_CO2       = gas.X[gas.species_index('CO2')]   # mole fraction of CO2
X_H2O       = gas.X[gas.species_index('H2O')]   # mole fraction of H2O
##############################################################################################################

# Get initial gas properties and theta from cantera
C           = gas.density_mole            # default density [kmol/m^3]
M           = gas.mean_molecular_weight # mix molar mass default [kg/kmol]
R           = (ct.gas_constant/10**6)     # default R [J/kmol−K] convert to [kJ/gmol−K]
RT          = R*gas.T                     # [kJ/gmol]
T           = gas.T                       # starts at initial temp
theta       = gas.molecular_weights[gas.species_index('CxHy')] # fuel n−heptane = 100.2 [kg/kmol]

sigma       = 2                           # User can set sigma here

# Calculate initial Psi, alpha, beta, and Keff and Eeff
```

```
Psi          = theta∗∗2 + sigma∗∗2
alpha        = (theta∗∗2)/(sigma∗∗2)
beta         = (sigma∗∗2)/theta
gamma        = 0                         # orgin of the distribution

ak           = 21.998                    # unit less Fitted to Mehl et al . 2011
bk           = 0.03048                   # unit less (Sabourin et al ., 2016)
ae           = 134.875                   # [kJ/mol] Fitted to Mehl et al . 2011
be           = 0.2270                    # unit less (Sabourin et al ., 2016)

Ae           = ae + gamma∗be             # constant
Ak           = ak + gamma∗bk             # constant
Aa1          = ak − ae/RT                # Aa1 to manually calculate W_bar
Aa2          = − bk + be/RT + 1/beta
Keff         = math.exp(ak)∗(1/((beta∗Aa2)∗∗alpha)∗math.exp(be∗alpha/(Aa2∗RT)))

dt           = 0                         #  Initialize  for lower while loop

# Calculate Arrhenius {A, b, E}, default b = 0
A            = Keff           # Sabourin et al . 2016 expression A = K(theta) = math.exp(ak + bk∗theta) # [1/s]
Eeff         = ae + be∗alpha/Aa2 # [kJ/gmol]

#######################################################################################################################
# Get initial W_bar from cantera and calculate initial twiddle and hat
W_bar        = − gas.forward_rates_of_progress[gas.species_index(’CxHy’)] # default [kmol/m^3/s] Note the negative
W_twiddle    = W_bar∗(alpha/Aa2)                     # needed to calculate theta
W_hat        = W_bar∗(alpha∗(alpha + 1)/(Aa2∗∗2)) # needed to calculate Psi

#######################################################################################################################

# Get reaction from Cantera − Inport R1 from CTI file
# Reaction 1
R1           = gas.reaction(0)
A1           = R1.rate.pre_exponential_factor     # default [kmol/m3−s]
b1           = R1.rate.temperature_exponent       # unit−less
E1           = R1.rate.activation_energy          # default [J/Kmol]

# Modify Reaction R1 − Two−step mechanism
R1.reactants = {’CxHy’: 1, ’O2’: (theta+4.998)/14.027}
R1.products  = {’CO’: (theta−2.016)/14.027, ’H2O’: (theta+12.011)/14.027}
A_1          = A                          # [kmol/m3−s] needs to be in those units
b_1          = b1
E_1          = Eeff∗10∗∗6                 # convert from [kJ/gmol] to [J/Kmol]. Needed for Cantera
R1.rate      = ct.Arrhenius(A_1, b_1, E_1)
gas.modify_reaction(0, R1)                # update first reaction equation ’0’

#######################################################################################################################

# Set NASA "FUEL" species polynomial coefficients for specific heat [Cp^o], Enthalpy [h^o] and entropy [s^0].
A00 =  2.473e−01        # (Hallett, 2017)
A01 =  1.092e−02        # (Hallett, 2017)
A02 = −3.124e−06        # (Hallett, 2017)
A03 = −3.757e−11        # (Hallett, 2017)
A04 =  0                # (Hallett, 2017)
A05 = −5420.4379217007 # Curve Fit Values (Fox and Hallett, 2017)
A06 =  12.9312135308    # Curve Fit Values (Fox and Hallett, 2017)

B00 = −8.328e−03        # (Hallett, 2017)
B01 =  8.704e−04        # (Hallett, 2017)
B02 = −4.984e−07        # (Hallett, 2017)
B03 =  1.090e−10        # (Hallett, 2017)
```

```
B04 =  0                   # (Hallett, 2017)
B05 = −208.6660658572 # Curve Fit Values (Fox and Hallett, 2017)
B06 =  0.14860092        # Curve Fit Values (Fox and Hallett, 2017)

fuel_coeffs  = [2350,
               (A00+B00∗theta), (A01+B01∗theta), (A02+B02∗theta),
               (A03+B03∗theta), (A04+B04∗theta), (A05+B05∗theta),
               (A06+B06∗theta), (A00+B00∗theta), (A01+B01∗theta),
               (A02+B02∗theta), (A03+B03∗theta),(A04+B04∗theta),
               (A05+B05∗theta), (A06+B06∗theta) ]

# Modify NASA polynomial coefficients imported from .cti file.
S1 = gas.species(gas.species_index(’CxHy’))
S1.thermo = ct.NasaPoly2(300, 5000, ct.one_atm, fuel_coeffs)
gas.modify_species(gas.species_index(’CxHy’), S1) # update first species coeffs

###########################################################################################

# Build Reactor r
r             = ct.IdealGasConstPressureReactor(gas)  # ideal gas, constant pressure reactor filled with gas
tnow            = 0                                   # start time [s]
tfinal          = 0.45                                # End after Time [s]
states          = ct.SolutionArray(gas, extra=[’t’])  # array to hold solution of gas (used for plotting)
sim             = ct.ReactorNet([r])                  # prepare the simulation with a ReactorNet object

###########################################################################################

# print to screen a header for output values
print(’%10s %10s %10s %10s %10s %10s %10s %10s %10s %10s %10s %10s %10s %10s %10s %10s %10s %10s %10s %10s %10s
       %10s’
       ’%10s %10s %10s’%
      (’t’,’T’,’R’,’P’,’C’,’M’,’Xf’,’Xo’,’Xn2’,’Xco’, ’Xco2’,’Xh2o’,’Aa1’,’Aa2’,’alpha’,’beta’,
       ’W_bar’,’W_twiddle’, ’W_hat’,’theta’,’Psi’,’sigma’, ’A1’, ’E1’, ’R1’))

# Define storage arrays (Use Capitals to keep tract) to hold data (Only used for plotting with matplotlib.pylab)
Temp            = []   # Temperature [K]
Temp_invers     = []   # Inverse Temperature [1/K]
TIME            = []   # Time [s]
THETA           = []
PSI             = []
SIGMA           = []
WBAR            = []   # [kmol/m^3/s]
KEFF            = []   # [1/s]
EEFF            = []   # [kJ/gmol]

###########################################################################################

# Write output to CSV file for importing into Excel
csv_file  = ’2Step_Reactor_CxHy.csv’

with open(csv_file, ’w’) as outfile:
    writer = csv.writer(outfile)
    writer.writerow([’t [s] c2’,’T [K] c2’, ’C [kmol/m^3] c2’,’M [kg/kmol] c2’,
                     ’Xf c2’,’Xo c2’,’Xn2 c2’,’Xco c2’, ’Xco2 c2’,’Xh2o c2’, ’Aa1 c2’,’Aa2 c2’,’alpha c2’,’beta c2’,
                     ’W_bar [kmol/m^3/s] c2’, ’W_twiddle [kmol/m^3/s] c2’, ’W_hat [kmol/m^3/s] c2’,’theta c2’,’Psi
                          c2’,’sigma c2’,
                     ’pre exponential: Keff [kmol/m3−s] c2’, ’activation energy: Eeff [J/Kmol] c2’, ’R1.equation c2’])

    # Begin while loop (note its nested within a "with open" statement)
    while tnow < tfinal:   # Note, in first step, initial values are displayed
```

```
if   850 <= T <= 850.01  or 875  <= T <= 875.01  or 900  <= T <= 900.01  or 925  <= T <= 925.01  or \
     950 <= T <= 950.01  or 975  <= T <= 975.01  or 1000 <= T <= 1000.01 or 1025 <= T <= 1025.01 or \
    1050 <= T <= 1050.01 or 1075 <= T <= 1075.01 or 1100 <= T <= 1100.01 or 1125 <= T <= 1125.01 or \
    1150 <= T <= 1150.01 or 1175 <= T <= 1175.01 or 1200 <= T <= 1200.01 or 1225 <= T <= 1225.01 or \
    1250 <= T <= 1250.01 or 1275 <= T <= 1275.01 or 1300 <= T <= 1300.01 or 1325 <= T <= 1325.01 or \
    1350 <= T <= 1350.01 or 1375 <= T <= 1375.01 or 1400 <= T <= 1400.01 or 1425 <= T <= 1425.01 or \
    1450 <= T <= 1450.01 or 1475 <= T <= 1475.01 or 1500 <= T <= 1500.01 or 1525 <= T <= 1525.01 or \
    1550 <= T <= 1550.01 or 1575 <= T <= 1575.01 or 1600 <= T <= 1600.01 or 1625 <= T <= 1625.01 or \
    1650 <= T <= 1650.01 or 1675 <= T <= 1675.01 or 1700 <= T <= 1700.01 or 1725 <= T <= 1725.01 or \
    1750 <= T <= 1750.01 or 1775 <= T <= 1775.01 or 1800 <= T <= 1800.01 or 1825 <= T <= 1825.01 or \
    1850 <= T <= 1850.01 or 1875 <= T <= 1875.01 or 1900 <= T <= 1900.01 or 1925 <= T <= 1925.01 or \
    1950 <= T <= 1950.01 or 1975 <= T <= 1975.01 or 2000 <= T <= 2000.01 or 2025 <= T <= 2025.01 or \
    2050 <= T <= 2050.01 or 2075 <= T <= 2075.01 or 2100 <= T <= 2100.01 or 2125 <= T <= 2125.01 or \
    2150 <= T <= 2150.01 or 2175 <= T <= 2175.01 or 2200 <= T <= 2200.01 or 2225 <= T <= 2225.01 or \
    2250 <= T <= 2250.01 or 2275 <= T <= 2275.01 or 2300 <= T <= 2300.01 or 2325 <= T <= 2325.01 or \
    2350 <= T <= 2350.01 or 2375 <= T <= 2375.01 or 2400 <= T <= 2400.01 or 2425 <= T <= 2425.01 or \
    2450 <= T <= 2450.01 or 2475 <= T <= 2475.01 or 2500 <= T <= 2500.01 or 2525 <= T <= 2525.01 or \
    2550 <= T <= 2550.01 or 2575 <= T <= 2575.01 or 2600 <= T <= 2600.01 or 2625 <= T <= 2625.01 or \
    2650 <= T <= 2650.01 or 2675 <= T <= 2675.01 or 2700 <= T :



    # output to screen
    print('%10.1e %10.2f %10.3e %10.0f %10.5f %10.5f %10.5f %10.5f %10.5f %10.5f %10.5f %10.5f %10.2f %10.2f %10.2f
          %10.2f'
          '%10.1e %10.1e %10.1e %10.2f %10.2f %10.2f %10s %10s %10s'%
           (sim.time, T, R, P, C, M, Xf, Xo, X_N2, X_CO, X_CO2, X_H2O, Aa1, Aa2, alpha, beta, W_bar,
            W_twiddle, W_hat, theta, Psi, sigma,
            R1.rate.pre_exponential_factor, R1.rate.activation_energy, R1.equation))

    # write output to CSV
    writer.writerow([sim.time, T, C, M, Xf, Xo, X_N2, X_CO, X_CO2,X_H2O, Aa1, Aa2, alpha, beta,
                     W_bar, W_twiddle, W_hat, theta, Psi, sigma,
                     R1.rate.pre_exponential_factor, R1.rate.activation_energy, R1.equation])

sim. reinitialize ()                   # Comment out to speed up, but CT results are no good
t1      = sim.time
tnow    = sim.step()                   # take single steps towards tfinal
states.append(r.thermo.state, t=tnow)  # update the array of states at tnow (for plotting to matplotlib.pylab)
t2      = sim.time
dt      = t2-t1                        # Need delta t for computations (Should it be zero at step 1?)

# Get properties from cantera: W_bar, Xf_new, T, C, M_mix
W_bar       = - gas.forward_rates_of_progress[gas.species_index('CxHy')] # default [kmol/m^3/s] Note the
        negative
W_twiddle   = W_bar*(alpha/Aa2)                    # needed to calculate theta
W_hat       = W_bar*(alpha*(alpha + 1)/(Aa2**2)) # needed to calculate Psi
Xf_new      = r.thermo.X[gas.species_index('CxHy')]                       # mole fraction of fuel
T_new       = r.T                                  # temp [K]
dT          = T_new - T                            # change in temp [k]
C_new       = gas.density_mole                     # update default molar density [kmol/m^3]
M_new       = gas.mean_molecular_weight            # molecular mass [kg/kmol]
del_C       = C_new - C                            # change in molar density [kmol/m^3]
del_M       = M_new - M                            # change in mixture molecular mass [kg/kmol]

# Convection expansion term
del_Cvdt    = - (C/M)*del_M - del_C                # use old values (M,C,T)

# Calculate theta
RHS_theta   = C*Xf*theta + W_twiddle*dt - Xf*theta*del_Cvdt
theta_new   = RHS_theta/C_new/Xf_new
```

```
    # Calculate Psi
    RHS_psi          = C*Xf*Psi + W_hat*dt - Xf*Psi*del_Cvdt
    Psi_new          = RHS_psi/(C_new*Xf_new)

    # Calculate sigma
    x_root           = Psi_new - (theta_new**2)
    sigma_new        = math.sqrt(abs(x_root))      # careful remove abs(), then this breaks if root is negative


    # used for making plots (matplotlib.pylab)
    TIME.append(float(sim.time))
    Temp_invers.append(float(1/r.T))
    WBAR.append(float(gas.forward_rates_of_progress[gas.species_index('CxHy')]))
    THETA.append(float(theta))
    SIGMA.append(float(sigma))

##########################################################################################################
    # Update for next run of loop
    Xf = gas.X[gas.species_index('CxHy')]     # mole fraction of fuel
    Xo = gas.X[gas.species_index('O2')]       # mole fraction of oxygen
    X_N2 = gas.X[gas.species_index('N2')]     # mole fraction of N2
    X_CO = gas.X[gas.species_index('CO')]     # mole fraction of CO
    X_CO2 = gas.X[gas.species_index('CO2')] # mole fraction of CO2
    X_H2O = gas.X[gas.species_index('H2O')] # mole fraction of H2O

    T                = r.T
    R                = (ct.gas_constant/10 ** 6)  # default R [J/kmol-K] convert to [kJ/gmol-K]
    RT               = R*T                        # [kJ/gmol]
    P                = gas.P                      # [pa]
    C                = gas.density_mole           # default density [kmol/m^3]
    M                = gas.mean_molecular_weight # mixture molar mass default [kg/kmol]
    theta            = theta_new
    Psi              = Psi_new
    sigma            = sigma_new
    alpha            = (theta**2)/(sigma**2) # theta/beta
    beta             = (sigma**2)/theta
    Aa1              = ak - ae/RT             # Aa1 to manually calculate W_bar
    Aa2              = - bk + be/RT + 1/beta
    Keff             = math.exp(ak)*(1/((beta*Aa2)**alpha)*math.exp(be*alpha/(Aa2*RT)))

    # Calculate Arrhenius {A, b, E}, default b = 0
    A              = Keff                # [1/s]
    Eeff           = ae + be*alpha/Aa2 # [kJ/gmol]

    # Modify Reaction equations
    # Reaction 1 (Two step only !!!)
    R1.reactants   = {'CxHy': 1, 'O2': (theta + 4.998)/14.027}
    R1.products    = {'CO': (theta - 2.016)/14.027, 'H2O': (theta + 12.011)/14.027}
    A_1            = A            # [kmol/m3-s] needs to be in those units
    b_1            = b1
    E_1            = Eeff*10**6  # convert from [kJ/gmol] to [J/Kmol]. Needed for Cantera
    R1.rate        = ct.Arrhenius(A_1, b_1, E_1)
    gas.modify_reaction(0, R1)  # update first reaction equation '0'

    # Modify "Fuel" species NASA coefficients
    fuel_coeffs  = [2350,
        (A00+B00*theta), (A01+B01*theta), (A02+B02*theta),
        (A03+B03*theta), (A04+B04*theta), (A05+B05*theta),
        (A06+B06*theta), (A00+B00*theta), (A01+B01*theta),
        (A02+B02*theta), (A03+B03*theta),(A04+B04*theta),
```

```
            (A05+B05*theta), (A06+B06*theta) ]

    S1 = gas.species(gas.species_index('CxHy'))
    S1.thermo = ct.NasaPoly2(300, 5000, ct.one_atm, fuel_coeffs)
    gas.modify_species(gas.species_index('CxHy'), S1) # update first species coeffs

################################################################################################################
#                                              THE END OF PROGRAM
################################################################################################################
```

C.3 Four-step Reactor Program (4Step_Reactor_CxHy.py)

With Continuous Thermodynamics

LISTING C.4: 4Step_Reactor_CxHy.py - with CT

```
####################################################################################################
# C.D.L.Fox & Dr.W.L.H.Hallett
# Four-Step Mechanism 4Step_Reactor_CxHy.py
# Continuous Thermodynamics Model
# Default Units: length m, mass kg, specify number of molecules kmol, time s, energy J, act_energy K, pressure Pa
####################################################################################################
from __future__ import division
import cantera as ct
from mpmath import *
mp.dps = 20.0       # Set mpmath global number of decimal places
import csv
####################################################################################################
ct.add_directory('/home/foxc/droplet/test/reactors/reactor1/input_cti')
gas                 = ct.Solution('4step_CxHy_40atm_NASA_Modified.cti') # n-heptane 100.204 [kg/kmol]
phi                 = 1.0                         # Equivalence ratio
x                   = gas.n_atoms('CxHy', 'C')    # of C atoms in fuel
y                   = gas.n_atoms('CxHy', 'H')    # of H atoms in fuel
a                   = float (x + y/4.0)
P                   = 40.0*ct.one_atm             # [pa]
initial_composition = {'CxHy': 1.0, 'O2': a/phi, 'N2': a*3.76/phi}
gas.TPX             = 575, P, initial_composition # T =575K, LT=720K, HT = 1040K

# Species = CxHy O2 N2 CO2 H2O X I
Xf                  = gas.X[gas.species_index('CxHy')]    # mole fraction of fuel
Xo                  = gas.X[gas.species_index('O2')]      # mole fraction of oxygen
X_CO2               = gas.X[gas.species_index('CO2')]     # mole fraction of CO2
X_H2O               = gas.X[gas.species_index('H2O')]     # mole fraction of H2O
X_X                 = gas.X[gas.species_index('X')]       # mole fraction of X intermediate species
X_I                 = gas.X[gas.species_index('I')]       # mole fraction of I intermediate species

# Get initial gas properties and theta from cantera
C                   = gas.density_mole             # default density [kmol/m^3]
M                   = gas.mean_molecular_weight # mix molar mass default [kg/kmol]
R                   = (ct.gas_constant/1000000.0)   # default R [J/kmol-K] convert to [kJ/gmol-K]
RT                  = R*gas.T                      # [kJ/gmol]
T                   = gas.T                        # starts at initial temp
dt                  = 0.0                          # Initialize for lower while loop

theta               = gas.molecular_weights[gas.species_index('CxHy')] # n-heptane = 100.204 [kg/kmol]
sigma               = 2 #42.5                                    # User can set sigma here

# Calculate initial Psi, alpha, beta, and Keff and Eeff
Psi                 = theta*theta + sigma*sigma
alpha               = (theta*theta)/(sigma*sigma)
beta                = (sigma*sigma)/theta
gamma               = 0.0                          # origin of the distribution

# Initialize distribution variables for I
```

```
theta_p                = theta  + 64
#theta_p                = theta

sigma_p                = sigma

Psi_p                  = Psi  +128*theta + 64*64
#Psi_p                  = Psi

alpha_p                = (theta_p*theta_p)/(sigma_p*sigma_p)
beta_p                 = (sigma_p*sigma_p)/theta_p

# Rate parameters for R1 and R3f
bk                     = 0.03048  # kmol/kg          # Sabourin et al, 2016
be                     = 0.2270   # kmol-kJ/kg-gmol # Sabourin et al, 2016
ak                     = 19.972   # unitless         # Derived from Muller et al. 1992
ae                     = 157.252  # [kJ/gmol]        # Derived from Muller et al. 1992
ak3                    = 32.583   # unitless         # Derived from Muller et al. 1992
ae3                    = 157.252  # [kJ/gmol]        # Derived from Muller et al. 1992

# Reaction R1
Ae                     = ae + gamma*be # constant
Ak                     = ak + gamma*bk # constant
Aa2                    = - bk + be/RT + 1.0/beta
Keff                   = float(mp.exp(ak)*(1.0/(mp.power(beta*Aa2,alpha))*mp.exp(be*alpha/(Aa2*RT))))
Eeff                   = ae + be*alpha/Aa2                                             # [kJ/gmol]
R1                     = gas.reaction(0)
R1.rate                = ct.Arrhenius(Keff, 0.0,  Eeff*1000000.0)  # convert Eeff  [kJ/gmol] to [J/Kmol]
gas.modify_reaction(0, R1)                                         # update first  reaction equation '0'

# Modify Reactions R2
R2                     = gas.reaction(1)
Keff2                  = R2.rate.pre_exponential_factor  # [1/s]
Eeff2                  = R2.rate.activation_energy * 0.000001
R2.reactants           = {'X':  1.0,  'O2':  (3.0*theta  + 8.0)/28.0}
R2.products            = {'CO2': (theta  - 2.0)/14.0,  'H2O': (theta  + 12.0)/14.0}
gas.modify_reaction(1, R2)

# Reaction 3f
Ae3                    = ae3 + gamma*be # constant
Ak3                    = ak3 + gamma*bk # constant
Aa23                   = - bk + be/RT + 1.0/beta
Keff3                  = float(mp.exp(ak3)*(1.0/(mp.power(beta*Aa23,alpha))*mp.exp(be*alpha/(Aa23*RT))))
Eeff3                  = ae3+be*alpha/Aa23                         # [kJ/gmol]
R3                     = gas.reaction(2)
R3.rate                = ct.Arrhenius(Keff3, 0.0,  Eeff3*1000000.0)    # convert Eeff  [kJ/gmol] to [J/Kmol]
gas.modify_reaction(2, R3)

# Reaction 3b
R3b                    = gas.reaction(3)
Keff3b                 = R3b.rate.pre_exponential_factor               # [1/s]
Eeff3b                 = R3b.rate.activation_energy*0.000001           # [J/kmol] to [kJ/gmol]
R3b.rate               = ct.Arrhenius(Keff3b, 0.0,  Eeff3b*1000000.0)  # convert Eeff  from [kJ/gmol] to [J/Kmol]
gas.modify_reaction(3, R3b)

# Reaction 4
R4                     = gas.reaction(4)
Keff4                  = R4.rate.pre_exponential_factor                # [m3/kmol/s]
Eeff4                  = R4.rate.activation_energy*0.000001            # [J/kmol] to [kJ/gmol]
R4.rate                = ct.Arrhenius(Keff4, 0.0,  Eeff4*1000000.0)    # convert Eeff  [kJ/gmol] to [J/Kmol]

R4.reactants           = {'I':  1.0,  'O2':  (3.0*theta_p - 240)/28.0}
```

```
R4.products          = {’CO2’: (theta_p − 66)/14.0, ’H2O’: (theta_p − 52)/14.0}
#R4.reactants         = {’ I ’:  1.0,  ’O2’:  (3.0∗theta−20.0)/28.0}
#R4.products          = {’CO2’: (theta−2.0)/14.0, ’H2O’: (theta+12.0)/14.0}

gas.modify_reaction(4, R4)

###########################################################################################################
W_bar                = − gas.forward_rates_of_progress[0] # R1
W_bar2               = − gas.forward_rates_of_progress[1] # R2
W_bar3               = − gas.forward_rates_of_progress[2] # R3
W_bar3b              = + gas.forward_rates_of_progress[3] # R3b
W_bar4               = − gas.forward_rates_of_progress[4] # R4

# These give the same result as Cantera
#W_bar               = − Keff∗C∗Xf∗mp.exp(−Eeff/(RT))          # [kmol/m^3/s] Note the negative # R1
#W_bar3              = − Keff3∗(C∗∗2)∗Xf∗Xo∗mp.exp(−Eeff3/(RT)) # [kmol/m^3/s] Note the negative # R3
#W_bar3b             = + Keff3b∗C∗X_I∗mp.exp(−Eeff3b/(RT))     # [kmol/m^3/s] Note the positive # R3b
#W_bar4              = − Keff4∗(C∗∗2)∗X_I∗Xo∗mp.exp(−Eeff4/(RT)) # [kmol/m^3/s] Note the negative # R4

W_twiddle            = W_bar∗(alpha/Aa2)                     # needed to calculate theta              # R1
W_twiddle3           = W_bar3∗(alpha/Aa23)                   # needed to calculate theta              # R3
W_twiddle3b          = W_bar3b∗(alpha_p∗beta_p)              # needed to calculate theta and theta_p  # R3b
W_twiddle4           = W_bar4∗(alpha_p∗beta_p)               # needed to calculate theta and theta_p  # R4

W_hat                = W_bar∗(alpha∗(alpha + 1.0)/(Aa2∗Aa2))           # needed to calculate Psi            # R1
W_hat3               = W_bar3∗(alpha∗(alpha + 1.0)/(Aa23∗Aa23))        # needed to calculate Psi            # R3
W_hat3b              = W_bar3b∗(alpha_p∗(alpha_p + 1.0)∗beta_p∗beta_p) # needed to calculate theta and theta_p # R3b
W_hat4               = W_bar4∗(alpha_p∗(alpha_p + 1.0)∗beta_p∗beta_p) # needed to calculate theta and theta_p  # R4

###########################################################################################################

# Set NASA "FUEL" species polynomial coefficients for specific heat [Cp^o], Enthalpy [h^o] and entropy [s^0].
A00                  =  2.473e−01       # (Hallett, 2017)
A01                  =  1.092e−02       # (Hallett, 2017)
A02                  = −3.124e−06       # (Hallett, 2017)
A03                  = −3.757e−11       # (Hallett, 2017)
A04                  =  0.0             # (Hallett, 2017)
A05                  = −5420.4379217007 # Curve Fit Values (Fox and Hallett, 2017)
A06                  =  12.9312135308   # Curve Fit Values (Fox and Hallett, 2017)

B00                  = −8.328e−03       # (Hallett, 2017)
B01                  =  8.704e−04       # (Hallett, 2017)
B02                  = −4.984e−07       # (Hallett, 2017)
B03                  =  1.090e−10       # (Hallett, 2017)
B04                  =  0.0             # (Hallett, 2017)
B05                  = −208.6660658572  # Curve Fit Values (Fox and Hallett, 2017)
B06                  =  0.14860092      # Curve Fit Values (Fox and Hallett, 2017)

fuel_coeffs  = [2350.0,
                (A00+B00∗theta), (A01+B01∗theta), (A02+B02∗theta),
                (A03+B03∗theta), (A04+B04∗theta), (A05+B05∗theta),
                (A06+B06∗theta), (A00+B00∗theta), (A01+B01∗theta),
                (A02+B02∗theta), (A03+B03∗theta),(A04+B04∗theta),
                (A05+B05∗theta), (A06+B06∗theta) ]

# Modify NASA polynomial coefficients imported from .cti file.
S1                  = gas.species(gas.species_index(’CxHy’))
S1.thermo           = ct.NasaPoly2(300.0, 5000.0, ct.one_atm, fuel_coeffs)
gas.modify_species(gas.species_index(’CxHy’), S1) # update first species coeffs

###########################################################################################################
```

```
# Build Reactor r
r               = ct.IdealGasConstPressureReactor(gas) # ct.ConstPressureReactor(gas)
tnow            = 0.0                                  # start time [s]
 tfinal         = 1.75                                 # End after  Time [s] 1.75s, # LT 0.0063s, HT 0.001s
sim             = ct.ReactorNet([r])                   # prepare the simulation with a ReactorNet object
time            = 0.0
count           = 0.0
runonce         = 0.0
error_count     = 0.0

sim.set_max_time_step(0.00001)  #1e-5
sim.max_err_test_fails = 20

##########################################################################################################
# print to screen a header for output values
print(’%10s %10s %10s %10s %10s %10s %10s’% (’t’,’T’,’sigma’,’theta’,’Psi’ , ’C’, ’M’))
# print(’%10s %10s %10s %10s %10s %10s %10s %10s %10s %10s’ % (’t’,’T’,’Xf’,’Xo’,’Xco2’,’Xh2o’,’X_X’,’X_I’,’cX’,  ’cI ’) )

# Write output to CSV file for importing into Excel
 csv_file  = ’/home/foxc/droplet/test/reactors/reactor1/!4Step_Reactor_CxHy/4Step_Reactor_CxHy_LT_1040K.csv’

# species = """  CxHy O2 N2 CO2 H2O X I """,
# LT = 720K
# HT = 1040K
with open(csv_file, ’w’) as outfile:
    writer = csv.writer(outfile)
    writer.writerow([’t c4_sigma2_I’,’T c4_sigma2_I’,’C c4_sigma2_I’,’M c4_sigma2_I’, ’Xf c4_sigma2_I’,’Xo c4_sigma2_I’,
                     ’Xco2 c4_sigma2_I’,’Xh2o c4_sigma2_I’,’X_X c4_sigma2_I’, ’X_I c4_sigma2_I’,
                     ’W_bar c4_sigma2_I’,’W_bar2 c4_sigma2_I’, ’W_bar3 c4_sigma2_I’, ’W_bar3b c4_sigma2_I’,’W_bar4
                          c4_sigma2_I’,
                     ’W_twiddle c4_sigma2_I’,’W_twiddle3 c4_sigma2_I’, ’W_twiddle3b c4_sigma2_I’,’W_twiddle4
                          c4_sigma2_I’,
                     ’W_hat c4_sigma2_I’, ’W_hat3 c4_sigma2_I’, ’W_hat3b c4_sigma2_I’, ’W_hat4 c4_sigma2_I’,
                     ’theta  c4_sigma2_I’,’Psi  c4_sigma2_I’,’sigma c4_sigma2_I’,’theta_p c4_sigma2_I’,’Psi_p c4_sigma2_I’,
                     ’sigma_p c4_sigma2_I’,
                     ’R1.rate  c4_sigma2_I’, ’R2.rate  c4_sigma2_I’, ’R3.rate  c4_sigma2_I’, ’R3b.rate c4_sigma2_I’,
                     ’R4.rate  c4_sigma2_I’,
                     ’Keff c4_sigma2_I’, ’Keff2 c4_sigma2_I’, ’Keff3 c4_sigma2_I’, ’Keff3b c4_sigma2_I’, ’Keff4
                          c4_sigma2_I’,
                     ’Eeff  c4_sigma2_I’, ’Eeff2  c4_sigma2_I’, ’Eeff3  c4_sigma2_I’, ’Eeff3b c4_sigma2_I’, ’Eeff4
                          c4_sigma2_I’,
                     ’R1.equation c4_sigma2_I’, ’R2.equation c4_sigma2_I’, ’R3.equation c4_sigma2_I’,
                     ’R3b.equation c4_sigma2_I’,’R4.equation c4_sigma2_I’])

    # Begin while loop (note its  nested within a  "with open" statement)

    while sim.time < tfinal:   # Note, in  first  step,  initial  values are  displayed

        # This if statement is used to reduce the amount of output to screen and CSV file normally works perfect.
        if   575 <= T <=  575.000001 or 600   <= T <= 601  or 625   <= T <= 626  or \
             650 <= T <=  651 or 675   <= T <= 676  or 700   <= T <= 701  or 725   <= T <= 726  or \
             750 <= T <=  751 or 775   <= T <= 776  or 800   <= T <= 801  or 825   <= T <= 826  or \
             850 <= T <=  851 or 875   <= T <= 876  or 900   <= T <= 901  or 925   <= T <= 926  or \
             950 <= T <=  951 or 975   <= T <= 976  or 1000 <= T <= 1001 or 1025 <= T <= 1026 or \
            1050 <= T <= 1051 or 1075 <= T <= 1076 or 1100 <= T <= 1101 or 1125 <= T <= 1126 or \
            1150 <= T <= 1151 or 1175 <= T <= 1176 or 1200 <= T <= 1201 or 1225 <= T <= 1226 or \
            1250 <= T <= 1251 or 1275 <= T <= 1276 or 1300 <= T <= 1301 or 1325 <= T <= 1326 or \
            1350 <= T <= 1351 or 1375 <= T <= 1376 or 1400 <= T <= 1401 or 1425 <= T <= 1426 or \
            1450 <= T <= 1451 or 1475 <= T <= 1476 or 1500 <= T <= 1501 or 1525 <= T <= 1526 or \
            1550 <= T <= 1551 or 1575 <= T <= 1576 or 1600 <= T <= 1601 or 1625 <= T <= 1626 or \
            1650 <= T <= 1651 or 1675 <= T <= 1676 or 1700 <= T <= 1701 or 1725 <= T <= 1726 or \
```

```
   1750 <= T <= 1751 or 1775 <= T <= 1776 or 1800 <= T <= 1801 or 1825 <= T <= 1826 or \
   1850 <= T <= 1851 or 1875 <= T <= 1876 or 1900 <= T <= 1901 or 1925 <= T <= 1926 or \
   1950 <= T <= 1951 or 1975 <= T <= 1976 or 2000 <= T <= 2001 or 2025 <= T <= 2026 or \
   2050 <= T <= 2051 or 2075 <= T <= 2076 or 2100 <= T <= 2101 or 2125 <= T <= 2126 or \
   2150 <= T <= 2151 or 2175 <= T <= 2176 or 2200 <= T <= 2201 or 2225 <= T <= 2226 or \
   2250 <= T <= 2251 or 2275 <= T <= 2276 or 2300 <= T <= 2301 or 2325 <= T <= 2326 or \
   2350 <= T <= 2351 or 2375 <= T <= 2376 or 2400 <= T <= 2401 or 2425 <= T <= 2426 or \
   2450 <= T <= 2451 or 2475 <= T <= 2476 or 2500 <= T <= 2501 or 2525 <= T <= 2526 or \
   2550 <= T <= 2551 or 2575 <= T <= 2576 or 2600 <= T <= 2601 or 2625 <= T <= 2626 or \
   2650 <= T <= 2651 or 2675 <= T <= 2676 or 2700 <= T <= 2701 or 2725 <= T <= 2726 or \
   2750 <= T <= 2751 or 2775 <= T <= 2776 or 2800 <= T <= 2801 or 2825 <= T <= 2826 or \
   2850 <= T <= 2851 or 2875 <= T <= 2876 or 2900 <= T <= 2901 or 2925 <= T <= 2926 or \
   2950 <= T <= 2951 or 2975 <= T <= 2976 or 3000 <= T <= 3001 or 3025 <= T:
        # output to screen
    print('%10.8e %10.2f %10.1e %10.3f %10.5f'
          '%10.5f %10.5f'%
            (sim.time, T, sigma, theta, Psi, C, M))

    # Note Monitor sigma, sigma= 2 as long as "sim. reinitialize " is being used

    # write output to CSV for plotting
    writer.writerow([sim.time, T, C, M, Xf, Xo, X_CO2,X_H2O, X_X, X_I,
                     W_bar, W_bar2, W_bar3, W_bar3b, W_bar4,
                     W_twiddle, W_twiddle3, W_bar3b, W_twiddle4,
                     W_hat, W_hat3, W_hat3b, W_hat4,
                     theta, Psi, sigma, theta_p, Psi_p, sigma_p,
                     R1.rate,R2.rate,R3.rate, R3b.rate, R4.rate,
                     Keff, Keff2, Keff3, Keff3b, Keff4,
                     Eeff, Eeff2, Eeff3, Eeff3b, Eeff4,
                     R1.equation, R2.equation, R3.equation, R3b.equation, R4.equation])
count +=1.0
if count >= 1e5:                         #1e5
    sim. reinitialize ()
    count = 0.0

if T > 2500 and runonce == 0:    # T > 2500.60, LT > 2625.80, HT > 2900.0 cuts down output larger time step
   sim.set_max_time_step(0.001)
   runonce = 1

t1      = sim.time
tnow    = sim.step()
t2      = sim.time
dt      = t2−t1 # Assume tiny time step t2−t1    # Need delta t for computations

# states .append(r.thermo.state, t=tnow) # update the array of states at tnow (for plotting to matplotlib.pylab)
W_bar      = − gas.forward_rates_of_progress[0]
W_bar2     = − gas.forward_rates_of_progress[1]
W_bar3     = − gas.forward_rates_of_progress[2]
W_bar3b    = + gas.forward_rates_of_progress[3]
W_bar4     = − gas.forward_rates_of_progress[4]

W_twiddle  = W_bar∗(alpha/Aa2)                       # needed to calculate theta              # R1
W_twiddle3 = W_bar3∗(alpha/Aa23)                     # needed to calculate theta              # R3

W_twiddle3b = W_bar3b∗(alpha_p∗beta_p)               # needed to calculate theta and theta_p  # R3b
W_twiddle4 = W_bar4∗(alpha_p∗beta_p)                 # needed to calculate theta and theta_p  # R4

W_hat  = W_bar∗(alpha∗(alpha + 1.0)/(Aa2∗Aa2))           # needed to calculate Psi                # R1
W_hat3 = W_bar3∗(alpha∗(alpha + 1.0)/(Aa23∗Aa23))        # needed to calculate Psi                # R3

W_hat3b = W_bar3b∗(alpha_p∗(alpha_p + 1.0)∗(beta_p∗beta_p)) # needed to calculate theta and theta_p  # R3b
```

```
W_hat4 = W_bar4∗(alpha_p∗(alpha_p + 1.0)∗(beta_p∗beta_p)) # needed to calculate theta and theta_p   # R4

# Get properties from cantera: Xf_new, T, C, M_mix
Xf_new          = r.thermo.X[gas.species_index(’CxHy’)]       # mole fraction of fuel
X_I_new         = r.thermo.X[gas.species_index(’I’)]
T_new           = r.T                                   # temp [K]
dT              = T_new − T                             # change in temp [k]
C_new           = gas.density_mole                      # update default molar density [kmol/m^3]
M_new           = gas.mean_molecular_weight             # molecular mass [kg/kmol]
del_C           = C_new − C                             # change in molar density [kmol/m^3]
del_M           = M_new − M                             # change in mixture molecular mass [kg/kmol]

# Convection expansion term
del_Cvdt        = − (C/M)∗del_M − del_C                 # use old values (M,C,T)

# Calculate theta
RHS_theta       = C∗Xf∗theta + (W_twiddle + W_twiddle3 + W_twiddle3b)∗dt − (Xf)∗theta∗del_Cvdt
theta_new       = RHS_theta/C_new/(Xf_new)

# Calculate Psi
RHS_psi         = C∗Xf∗Psi + (W_hat + W_hat3 + W_hat3b)∗dt − (Xf)∗Psi∗del_Cvdt
Psi_new         = RHS_psi/(C_new∗(Xf_new))

# Calculate sigma
x_root          = Psi_new − (theta_new∗theta_new)
sigma_new       = mp.sqrt(abs(x_root))      # careful remove abs(), then this breaks if root is negative

theta           = theta_new
Psi             = Psi_new
sigma           = sigma_new                 # Should stay constant with sim. reinitialize ()
alpha           = (theta∗theta)/(sigma∗sigma)
beta            = (sigma∗sigma)/theta

# Calculate theta_p
RHS_theta_p     = C∗X_I∗theta_p + (W_twiddle4 − W_twiddle3 − W_twiddle3b)∗dt − (X_I)∗theta_p∗del_Cvdt
theta_p_new     = RHS_theta_p/C_new/(X_I_new)

# Calculate Psi_p
RHS_psi_p       = C∗X_I∗Psi_p + (W_hat4 − W_hat3 − W_hat3b)∗dt − (X_I)∗Psi_p∗del_Cvdt
Psi_p_new       = RHS_psi_p/(C_new ∗(X_I_new))

# Calculate sigma_p
x_root_p        = Psi_p_new − (theta_p_new∗theta_p_new)
sigma_p_new     = mp.sqrt(abs(x_root_p)) # careful remove abs(), then this breaks if root is negative

theta_p         = theta_p_new
Psi_p           = Psi_p_new
sigma_p         = sigma_p_new

# This is done because the values of theta_p, sigma_p and Psi_p will not converge with Cantera
# unless the time step is extremly smaller than (1e−7) # not worth the computation time.



#alpha_p         = (theta_p∗theta_p)/(sigma_p∗sigma_p)
#beta_p          = theta_p/alpha_p


# Update Species = CxHy O2 N2 CO2 H2O X I
Xf              = gas.X[gas.species_index(’CxHy’)]    # mole fraction of fuel (note this drops which is why I
    predict sigma deviates)
```

```
Xo              = gas.X[gas.species_index(’O2’)]       # mole fraction of oxygen
X_CO2           = gas.X[gas.species_index(’CO2’)]      # mole fraction of CO2
X_H2O           = gas.X[gas.species_index(’H2O’)]      # mole fraction of H2O
X_X             = gas.X[gas.species_index(’X’)]        # mole fraction of X intermediate species
X_I             = gas.X[gas.species_index(’I’)]        # mole fraction of I intermediate species

T               = gas.T
R               = (ct.gas_constant/1000000.0)
RT              = R∗T
P               = gas.P                      # [pa]
C               = gas.density_mole           # default density [kmol/m^3]
M               = gas.mean_molecular_weight # mixture molar mass default [kg/kmol]

# Reaction R1
Aa2   = − bk + be/RT + 1.0/beta
if  T> 2500:                              # T> 2500.0, LT >2600, HT > 2919.0
    Keff = 0.0 #R1.rate.pre_exponential_factor # Turn off this Reaction
    Eeff = R1.rate.activation_energy∗0.000001
else:
    Keff  = float(mp.exp(ak + (be∗alpha)/(Aa2∗RT))/(mp.power(beta∗Aa2,alpha))) # [1/s]
    Eeff  = ae + be∗alpha/Aa2 # [kJ/gmol]
R1.rate = ct.Arrhenius(Keff, 0.0, Eeff∗1000000.0) # convert Eeff from [kJ/gmol] to [J/Kmol]. Needed for Cantera

# Reactions R2
Keff2 = R2.rate.pre_exponential_factor # [1/s]
Eeff2 = R2.rate.activation_energy ∗ 0.000001
R2.reactants = {’X’: 1.0, ’O2’: (3.0∗theta+8.0)/28.0}
R2.products = {’CO2’: (theta−2.0)/14.0, ’H2O’: (theta+12.0)/14.0}

# Reaction R3f

Aa23 = − bk + be/RT + 1.0/beta
if  T> 2500:                              # T> 2500.0, LT >, HT > 2919.0
    Keff3 = 0.0
    Eeff3 = R3.rate.activation_energy∗0.000001
else:
    Keff3 = float(mp.exp(ak3 + (be ∗ alpha) / (Aa23 ∗ RT)) / (mp.power(beta ∗ Aa23, alpha)))
    Eeff3 = ae3 + be∗alpha/Aa23 # [kJ/gmol]
R3.rate = ct.Arrhenius(Keff3, 0.0, Eeff3 ∗1000000.0) # convert Eeff from [kJ/gmol] to [J/Kmol]. Needed for
       Cantera

# Reaction R3b
Keff3b = R3b.rate.pre_exponential_factor # [1/s]
Eeff3b = R3b.rate.activation_energy∗0.000001 # [J/kmol] to [kJ/gmol]
R3b.rate = ct.Arrhenius(Keff3b, 0.0, Eeff3b ∗1000000.0) # convert Eeff from [kJ/gmol] to [J/Kmol]. Needed for
       Cantera

# Reaction R4
R4.reactants = {’I’: 1.0, ’O2’: (3.0 ∗ theta_p − 240) / 28.0}
R4.products = {’CO2’: (theta_p − 66) / 14.0, ’H2O’: (theta_p − 52) / 14.0}
#R4.reactants = {’I’: 1.0, ’O2’: (3.0∗theta−20.0)/28.0}
#R4.products = {’CO2’: (theta−2.0)/14.0, ’H2O’: (theta+12.0)/14.0}
Keff4 = R4.rate.pre_exponential_factor # [m3/kmol/s]
Eeff4 = R4.rate.activation_energy∗0.000001 # [J/kmol] to [kJ/gmol]
R4.rate = ct.Arrhenius(Keff4, 0.0, Eeff4∗1000000.0) # convert Eeff from [kJ/gmol] to [J/Kmol]. Needed for Cantera

gas.modify_reaction(0, R1) # Update all Reactions
gas.modify_reaction(1, R2)
gas.modify_reaction(2, R3)
gas.modify_reaction(3, R3b)
gas.modify_reaction(4, R4)
```

```
# Modify "Fuel" species NASA coefficients
fuel_coeffs  =  [2350.0,
```

Appendix D

Four-step Ignition Delay Program File (`.py`)

Appendix D includes the ignition delay program written in Python for the four-step reactor in continuous thermodynamics form only. The single- and two-step ignition delay programs are not included since they are similar and the code required inside the functions is the same as that presented in Appendix C.

D.1 Four-step Ignition Delay Program

D.1.1 Main Program

Listing D.1: 4Step_Mech_n-heptane_Ignition_Time_CT.py - with CT

```
from __future__ import division
import four_step_Ignition_delay_CT
import csv

initial_temperatures = [575, 750, 800, 875, 950, 1000, 1100, 1200, 1300, 1400, 1500, 1600]

 csv_file = '4Step_Mech_n-heptane_Ignition_Time_CT_Sigma0_I.csv'

with open(csv_file, 'w') as outfile:
    writer = csv.writer(outfile)
    writer.writerow([' Initial temp T_i [K] 4CT_Sigma40_I', 'Ignition delay t [s] 4CT_Sigma40_I', '1/T_i [1/K]
        4CT_Sigma0_I'])

    for T0 in initial_temperatures:

        time = four_step_Ignition_delay_CT.ignition_delay(T0)

        print('The ignition delay for {} K is {}'.format(T0, time))
        writer.writerow([T0, time, float(1)/float(T0)])

####################################################################################################
#                                          THE END OF PROGRAM
####################################################################################################
```

D.1.2 Function for Main Program

LISTING D.2: four_step_Ignition_delay_CT.py - with CT

```
from __future__ import division

def ignition_delay(T0):

    import cantera as ct
    from mpmath import *
    mp.dps = 20.0 # Set mpmath global number of decimal places

    ct.add_directory('/home/foxc/droplet/test/reactors/reactor1/input_cti')
    gas = ct.Solution('4step_CxHy_40atm_NASA_Modified.cti') # n-heptane 100.204 [kg/kmol]
    phi = 1.0   # Equivalence ratio
    x = gas.n_atoms('CxHy', 'C')  # of C atoms in fuel
    y = gas.n_atoms('CxHy', 'H')  # of H atoms in fuel
    a = float(x + y / 4.0)
    P = 40.0 * ct.one_atm # [pa]
    initial_composition = {'CxHy': 1.0, 'O2': a / phi, 'N2': a * 3.76 / phi}
    gas.TPX = T0, P, initial_composition

    # Species = CxHy O2 N2 CO2 H2O X I
    Xf = gas.X[gas.species_index('CxHy')] # mole fraction of fuel
    X_I = gas.X[gas.species_index('I')]   # mole fraction of I intermediate species

    # Get initial gas properties and theta from cantera
    C = gas.density_mole # default density [kmol/m^3]
    M = gas.mean_molecular_weight # mix molar mass default [kg/kmol]
    R = (ct.gas_constant / 1000000.0)  # default R [J/kmol-K] convert to [kJ/gmol-K]
    RT = R * gas.T  # [kJ/gmol]
    T = gas.T  # starts at initial temp
    dt = 0.0  # Initialize for lower while loop

    theta = gas.molecular_weights[gas.species_index('CxHy')] # n-heptane = 100.204 [kg/kmol]
    sigma = .1  # User can set sigma here #42.5

    # Calculate initial Psi, alpha, beta, and Keff and Eeff
    Psi = theta * theta + sigma * sigma
    alpha = (theta * theta) / (sigma * sigma)
    beta = (sigma * sigma) / theta
    gamma = 0.0 # origin of the distribution

    # Initialize distribution variables for I
    theta_p = theta + 64
    #theta_p = theta
    sigma_p = sigma
    Psi_p = Psi + 128 * theta + 64 * 64
    #Psi_p = Psi
    alpha_p = (theta_p * theta_p) / (sigma_p * sigma_p)
    beta_p = (sigma_p * sigma_p) / theta_p

    # Rate parameters for R1 and R3f
    bk = 0.03048  # kmol/kg          # Sabourin et al, 2016
    be = 0.2270  # kmol-kJ/kg-gmol # Sabourin et al, 2016
    ak = 19.972  # unitless           # Derived from Muller et al. 1992
    ae = 157.252  # [kJ/gmol]         # Derived from Muller et al. 1992
```

```
ak3 = 32.583   #  unitless              # Derived from Muller et al.  1992
ae3 = 157.252  #  [kJ/gmol]             # Derived from Muller et al.  1992

# Reaction R1
Ae = ae + gamma ∗ be # constant
Ak = ak + gamma ∗ bk # constant
Aa2 = − bk + be / RT + 1.0 / beta
Keff = float(mp.exp(ak) ∗ (1.0 / (mp.power(beta ∗ Aa2, alpha)) ∗ mp.exp(be ∗ alpha / (Aa2 ∗ RT))))
Eeff = ae + be ∗ alpha / Aa2  # [kJ/gmol]
R1 = gas.reaction(0)
R1.rate = ct.Arrhenius(Keff, 0.0, Eeff ∗ 1000000.0)  # convert Eeff [kJ/gmol] to [J/Kmol]
gas.modify_reaction(0, R1)  # update first reaction equation ’0’

# Modify Reactions R2
R2 = gas.reaction(1)

R2.reactants = {’X’: 1.0, ’O2’: (3.0 ∗ theta + 8.0) / 28.0}
R2.products = {’CO2’: (theta − 2.0) / 14.0, ’H2O’: (theta + 12.0) / 14.0}
gas.modify_reaction(1, R2)

# Reaction 3f
Ae3 = ae3 + gamma ∗ be # constant
Ak3 = ak3 + gamma ∗ bk # constant
Aa23 = − bk + be / RT + 1.0 / beta
Keff3 = float(mp.exp(ak3) ∗ (1.0 / (mp.power(beta ∗ Aa23, alpha)) ∗ mp.exp(be ∗ alpha / (Aa23 ∗ RT))))
Eeff3 = ae3 + be ∗ alpha / Aa23  # [kJ/gmol]
R3 = gas.reaction(2)
R3.rate = ct.Arrhenius(Keff3, 0.0, Eeff3 ∗ 1000000.0)  # convert Eeff [kJ/gmol] to [J/Kmol]
gas.modify_reaction(2, R3)

# Reaction 3b
R3b = gas.reaction(3)
Keff3b = R3b.rate.pre_exponential_factor  # [1/s]
Eeff3b = R3b.rate.activation_energy ∗ 0.000001  # [J/kmol] to [kJ/gmol]
R3b.rate = ct.Arrhenius(Keff3b, 0.0, Eeff3b ∗ 1000000.0)  # convert Eeff from [kJ/gmol] to [J/Kmol]
gas.modify_reaction(3, R3b)

# Reaction 4
R4 = gas.reaction(4)
Keff4 = R4.rate.pre_exponential_factor  # [m3/kmol/s]
Eeff4 = R4.rate.activation_energy ∗ 0.000001  # [J/kmol] to [kJ/gmol]
R4.rate = ct.Arrhenius(Keff4, 0.0, Eeff4 ∗ 1000000.0)  # convert Eeff [kJ/gmol] to [J/Kmol]
R4.reactants = {’I’: 1.0, ’O2’: (3.0 ∗ theta_p − 240) / 28.0}
R4.products = {’CO2’: (theta_p − 66) / 14.0, ’H2O’: (theta_p − 52) / 14.0}
#R4.reactants = {’I’: 1.0, ’O2’: (3.0 ∗ theta − 20.0) / 28.0}
#R4.products = {’CO2’: (theta − 2.0) / 14.0, ’H2O’: (theta + 12.0) / 14.0}
gas.modify_reaction(4, R4)

W_bar = − gas.forward_rates_of_progress[0] # R1
W_bar2 = − gas.forward_rates_of_progress[1] # R2
W_bar3 = − gas.forward_rates_of_progress[2] # R3
W_bar3b = + gas.forward_rates_of_progress[3] # R3b
W_bar4 = − gas.forward_rates_of_progress[4] # R4

W_twiddle = W_bar ∗ (alpha / Aa2) # needed to calculate theta                    # R1
W_twiddle3 = W_bar3 ∗ (alpha / Aa23) # needed to calculate theta                 # R3
W_twiddle3b = W_bar3b ∗ (alpha_p ∗ beta_p) # needed to calculate theta and theta_p   # R3b
W_twiddle4 = W_bar4 ∗ (alpha_p ∗ beta_p) # needed to calculate theta and theta_p   # R4

W_hat = W_bar ∗ (alpha ∗ (alpha + 1.0) / (Aa2 ∗ Aa2))  # needed to calculate Psi          # R1
W_hat3 = W_bar3 ∗ (alpha ∗ (alpha + 1.0) / (Aa23 ∗ Aa23))  # needed to calculate Psi      # R3
```

```
W_hat3b = W_bar3b ∗ (alpha_p ∗ (alpha_p + 1.0) ∗ beta_p ∗ beta_p) # needed to calculate theta and theta_p # R3b
W_hat4 = W_bar4 ∗ (alpha_p ∗ (alpha_p + 1.0) ∗ beta_p ∗ beta_p) # needed to calculate theta and theta_p # R4


# Set NASA "FUEL" species polynomial coefficients for specific heat [Cp^o], Enthalpy [h^o] and entropy [s^0].
A00 = 2.473e−01 # (Hallett, 2017)
A01 = 1.092e−02 # (Hallett, 2017)
A02 = −3.124e−06 # (Hallett, 2017)
A03 = −3.757e−11 # (Hallett, 2017)
A04 = 0.0  # (Hallett, 2017)
A05 = −5420.4379217007 # Curve Fit Values (Fox and Hallett, 2017)
A06 = 12.9312135308 # Curve Fit Values (Fox and Hallett, 2017)

B00 = −8.328e−03 # (Hallett, 2017)
B01 = 8.704e−04 # (Hallett, 2017)
B02 = −4.984e−07 # (Hallett, 2017)
B03 = 1.090e−10 # (Hallett, 2017)
B04 = 0.0  # (Hallett, 2017)
B05 = −208.6660658572 # Curve Fit Values (Fox and Hallett, 2017)
B06 = 0.14860092 # Curve Fit Values (Fox and Hallett, 2017)

fuel_coeffs = [2350.0,
               (A00 + B00 ∗ theta), (A01 + B01 ∗ theta), (A02 + B02 ∗ theta),
               (A03 + B03 ∗ theta), (A04 + B04 ∗ theta), (A05 + B05 ∗ theta),
               (A06 + B06 ∗ theta), (A00 + B00 ∗ theta), (A01 + B01 ∗ theta),
               (A02 + B02 ∗ theta), (A03 + B03 ∗ theta), (A04 + B04 ∗ theta),
               (A05 + B05 ∗ theta), (A06 + B06 ∗ theta)]

# Modify NASA polynomial coefficients imported from .cti file.
S1 = gas.species(gas.species_index(’CxHy’))
S1.thermo = ct.NasaPoly2(300.0, 5000.0, ct.one_atm, fuel_coeffs)
gas.modify_species(gas.species_index(’CxHy’), S1) # update first species coeffs

################################################################################################
# Build Reactor r
r = ct.IdealGasConstPressureReactor(gas) # ct.ConstPressureReactor(gas)
tnow = 0.0  # start time [s]
tfinal = 1.75  # End after Time [s]
sim = ct.ReactorNet([r])  # prepare the simulation with a ReactorNet object
time = 0.0
count = 0.0
runonce = 0.0
error_count = 0.0

#sim.set_max_time_step(0.00001) # 1e−5
sim.max_err_test_fails = 30
################################################################################################


# Begin while loop (note its nested within a "with open" statement)
while r.T < T0 + 300:  # Note, in first step, initial values are displayed

    count += 1.0
    if count >= 1e5:  # 1e5
        sim. reinitialize ()
        count = 0.0

    ’’’
    if T > 2500.60 and runonce == 0:  # This cuts down date output at the end by using a larger time step
        sim.set_max_time_step(0.001)
        runonce = 1
```

```
’’’

t1  = sim.time
tnow = sim.step()
t2  = sim.time
dt = t2  − t1  # Assume tiny time step t2−t1     # Need delta t  for  computations

# states .append(r.thermo.state, t=tnow) # update the array of  states  at  tnow (for  plotting  to  matplotlib.pylab)
W_bar = − gas.forward_rates_of_progress[0]
W_bar2 = − gas.forward_rates_of_progress[1]
W_bar3 = − gas.forward_rates_of_progress[2]
W_bar3b = + gas.forward_rates_of_progress[3]
W_bar4 = − gas.forward_rates_of_progress[4]

W_twiddle = W_bar ∗ (alpha / Aa2) # needed to calculate theta                        # R1
W_twiddle3 = W_bar3 ∗ (alpha / Aa23) # needed to calculate theta                        # R3

W_twiddle3b = W_bar3b ∗ (alpha_p ∗ beta_p) # needed to calculate theta and theta_p   # R3b
W_twiddle4 = W_bar4 ∗ (alpha_p ∗ beta_p) # needed to calculate theta and theta_p    # R4

W_hat = W_bar ∗ (alpha ∗ (alpha + 1.0) / (Aa2 ∗ Aa2))  # needed to calculate  Psi                           # R1
W_hat3 = W_bar3 ∗ (alpha ∗ (alpha + 1.0) / (Aa23 ∗ Aa23))  # needed to calculate  Psi                           # R3

W_hat3b = W_bar3b ∗ (
alpha_p ∗ (alpha_p + 1.0)  ∗  (beta_p ∗ beta_p))   # needed to calculate  theta  and theta_p    # R3b
W_hat4 = W_bar4 ∗ (
alpha_p ∗ (alpha_p + 1.0)  ∗  (beta_p ∗ beta_p))   # needed to calculate  theta  and theta_p    # R4

# Get properties  from cantera:  Xf_new, T, C, M_mix
Xf_new = r.thermo.X[gas.species_index(’CxHy’)] # mole fraction of  fuel
X_I_new = r.thermo.X[gas.species_index(’I’)]
T_new = r.T  # temp [K]
dT = T_new − T # change in temp [k]
C_new = gas.density_mole # update default molar density [kmol/m^3]
M_new = gas.mean_molecular_weight # molecular mass [kg/kmol]
del_C = C_new − C # change in molar density [kmol/m^3]
del_M = M_new − M # change in mixture molecular mass [kg/kmol]

# Convection expansion term
del_Cvdt = − (C / M) ∗ del_M − del_C # use old values (M,C,T)

# Calculate  theta
RHS_theta = C ∗ Xf ∗ theta + (W_twiddle + W_twiddle3 + W_twiddle3b) ∗ dt − (Xf) ∗ theta ∗ del_Cvdt
theta_new = RHS_theta / C_new / (Xf_new)

# Calculate  Psi
RHS_psi = C ∗ Xf ∗ Psi + (W_hat + W_hat3 + W_hat3b) ∗ dt − (Xf) ∗ Psi ∗ del_Cvdt
Psi_new = RHS_psi / (C_new ∗ (Xf_new))

# Calculate  sigma
x_root = Psi_new − (theta_new ∗ theta_new)
sigma_new = mp.sqrt(abs(x_root)) # careful  remove abs(), then  this  breaks  if  root  is  negative

theta  = theta_new
Psi  = Psi_new
sigma = sigma_new # Should stay constant with sim. reinitialize ()
alpha = (theta  ∗  theta)  / (sigma ∗ sigma)
beta  = (sigma ∗ sigma) / theta

# Calculate  theta_p
RHS_theta_p = C ∗ X_I ∗ theta_p + (W_twiddle4 − W_twiddle3 − W_twiddle3b) ∗ dt − (X_I) ∗ theta_p ∗ del_Cvdt
```

```
theta_p_new = RHS_theta_p / C_new / (X_I_new)

# Calculate Psi_p
RHS_psi_p = C * X_I * Psi_p + (W_hat4 - W_hat3 - W_hat3b) * dt - (X_I) * Psi_p * del_Cvdt
Psi_p_new = RHS_psi_p / (C_new * (X_I_new))

# Calculate sigma_p
x_root_p = Psi_p_new - (theta_p_new * theta_p_new)
sigma_p_new = mp.sqrt(abs(x_root_p)) # careful remove abs(), then this breaks if root is negative

theta_p = theta_p_new
Psi_p = Psi_p_new
sigma_p = sigma_p_new

# This is done because the values of theta_p, sigma_p and Psi_p will not converge with Cantera
# unless the time step is extremly smaller than (1e-7) # not worth the computation time.
'''
alpha_p         = (theta_p*theta_p)/(sigma_p*sigma_p)
beta_p          = theta_p/alpha_p
'''

# Update Species = CxHy O2 N2 CO2 H2O X I
Xf = gas.X[gas.species_index('CxHy')] # mole fraction of fuel (note this drops which is why I predict sigma deviates)

T = gas.T
R = (ct.gas_constant / 1000000.0)
RT = R * T
P = gas.P  # [pa]
C = gas.density_mole # default density [kmol/m^3]
M = gas.mean_molecular_weight # mixture molar mass default [kg/kmol]

# Reaction R1
Aa2 = - bk + be / RT + 1.0 / beta
if T > 2500.0:
    Keff = 0.0   # R1.rate.pre_exponential_factor # Turn off this Reaction
    Eeff = R1.rate.activation_energy * 0.000001
else:
    Keff = float(mp.exp(ak + (be * alpha) / (Aa2 * RT)) / (mp.power(beta * Aa2, alpha))) # [1/s]
    Eeff = ae + be * alpha / Aa2 # [kJ/gmol]
R1.rate = ct.Arrhenius(Keff, 0.0,
                       Eeff * 1000000.0)  # convert Eeff from [kJ/gmol] to [J/Kmol]. Needed for Cantera

# Reactions R2
R2.reactants = {'X': 1.0, 'O2': (3.0 * theta + 8.0) / 28.0}
R2.products = {'CO2': (theta - 2.0) / 14.0, 'H2O': (theta + 12.0) / 14.0}

# Reaction R3f
Aa23 = - bk + be / RT + 1.0 / beta
Keff3 = float(mp.exp(ak3 + (be * alpha) / (Aa23 * RT)) / (mp.power(beta * Aa23, alpha)))
Eeff3 = ae3 + be * alpha / Aa23 # [kJ/gmol]
R3.rate = ct.Arrhenius(Keff3, 0.0, Eeff3 * 1000000.0)  # convert Eeff from [kJ/gmol] to [J/Kmol]. Needed for Cantera

# Reaction R3b
Keff3b = R3b.rate.pre_exponential_factor  # [1/s]
Eeff3b = R3b.rate.activation_energy * 0.000001  # [J/kmol] to [kJ/gmol]
R3b.rate = ct.Arrhenius(Keff3b, 0.0, Eeff3b * 1000000.0)  # convert Eeff from [kJ/gmol] to [J/Kmol]. Needed for Cantera

# Reaction R4
```

```
R4.reactants = {’I’: 1.0, ’O2’: (3.0 ∗ theta_p − 240) / 28.0}
R4.products = {’CO2’: (theta_p − 66) / 14.0, ’H2O’: (theta_p − 52) / 14.0}
#R4.reactants = {’I’: 1.0, ’O2’: (3.0 ∗ theta − 20.0) / 28.0}
#R4.products = {’CO2’: (theta − 2.0) / 14.0, ’H2O’: (theta + 12.0) / 14.0}
Keff4 = R4.rate.pre_exponential_factor # [m3/kmol/s]
Eeff4 = R4.rate.activation_energy ∗ 0.000001 # [J/kmol] to [kJ/gmol]
R4.rate = ct.Arrhenius(Keff4, 0.0,
                       Eeff4 ∗ 1000000.0) # convert Eeff from [kJ/gmol] to [J/Kmol]. Needed for Cantera

gas.modify_reaction(0, R1) # Update all Reactions
gas.modify_reaction(1, R2)
gas.modify_reaction(2, R3)
gas.modify_reaction(3, R3b)
gas.modify_reaction(4, R4)

# Modify "Fuel" species NASA coefficients
fuel_coeffs = [2350.0,
               (A00 + B00 ∗ theta), (A01 + B01 ∗ theta), (A02 + B02 ∗ theta),
               (A03 + B03 ∗ theta), (A04 + B04 ∗ theta), (A05 + B05 ∗ theta),
               (A06 + B06 ∗ theta), (A00 + B00 ∗ theta), (A01 + B01 ∗ theta),
               (A02 + B02 ∗ theta), (A03 + B03 ∗ theta), (A04 + B04 ∗ theta),
               (A05 + B05 ∗ theta), (A06 + B06 ∗ theta)]
S1 = gas.species(gas.species_index(’CxHy’))
S1.thermo = ct.NasaPoly2(300.0, 5000.0, ct.one_atm, fuel_coeffs)
gas.modify_species(gas.species_index(’CxHy’), S1) # update Fuel species NASA coeffs
```