Chemical Kinetics Library

Harvard University

CS 207 project

Hongxiang Qiu
Riddhi Shah
Yijun Shen
Yuyue Wang

November 20, 2017

Contents

Chapter 1

Introduction

Chemical kinetics is the study of rates of chemical processes. Different experiment settings and conditions, such as temperature or concentration of reactants, can influence the speed of chemical reactions as well as the yield of products.

The library *chemkin* we wrote aims to make such calculation in chemical kinetics easier.

1 Reaction Rate Coefficients

There are several kinds of reaction rates, and here we are introducing three types of reaction rate coefficients:

1.1 Constant Reaction Rate Coefficient

$$k_{const} = k$$

1.2 Arrhenius Reaction Rate Coefficient

$$k_{arr} = A \exp\left(-\frac{E}{RT}\right)$$

1.3 Modified Arrhenius Reaction Rate Coefficient

$$k_{modarr} = AT^b \exp\left(-\frac{E}{RT}\right)$$

Note:

- The Arrhenius prefactor A is strictly positive.
- The modified Arrhenius parameter b must be real.
- R=8.314 is the ideal gas constant. It should never be changed (except to convert units)
- The temperature T must be positive (assuming a Kelvin scale)

2 Irreversible Reactions

Consider a system consisting of N species undergoing M irreversible, elementary reactions of the form

$$\textstyle\sum_{i=1}^{N} \nu'_{ij}\mathcal{S}_i \text{ −> } \sum_{i=1}^{N} \nu''_{ij}\mathcal{S}_i, \qquad \text{j = 1, \ldots, M}$$

the rate of change of specie i (**the reaction rate**) can be written as

$$\textstyle f_i = \sum_{j=1}^{M} \nu_{ij}\omega_j, \qquad \text{i = 1, \ldots, N}$$

where **the progress rate** for each reaction is given by

$$\omega_j = k_j \prod_{i=1}^{N} x_i^{\nu'_{ij}}, \qquad \text{j = 1, \ldots, M}$$

and k_j is **the forward reaction rate coefficient**.

3 Reversible Reactions

In reality, it is often the case that the products can react and produce the reactants. This is called a reversible reaction. It may have the form

$$\sum_{i=1}^{N} \nu'_{ij} \mathcal{S}_i \rightleftharpoons \sum_{i=1}^{N} \nu''_{ij} \mathcal{S}_i \qquad j = 1, \ldots, M$$

where $\rightleftharpoons$ indicts that the forward and backward reactions occur.

The total progress rate is now given by

$$\omega_j = k_j^{(f)} \prod_{i=1}^{N} x_i^{\nu'_{ij}} - k_j^{(b)} \prod_{i=1}^{N} x_i^{\nu''_{ij}}, \qquad j = 1, \ldots, M.$$

where $k_j^{(f)}$ is **the forward reaction rate coefficient**, and $k_j^{(b)}$ is **the backward reaction rate coefficient**.

For an elementary reaction (and only elementary reactions), we have

$$k_j^{(b)} = \frac{k_j^{(f)}}{k_j^e}, \qquad j = 1, \ldots, M$$

where k_j^e is **the equilibrium coefficient** for reaction j.

The final expression for **the equilibrium coefficient** is

$$k_j^e = \left(\frac{p_0}{RT}\right)^{\gamma_j} \exp\left(\frac{\Delta S_j}{R} - \frac{\Delta H_j}{RT}\right), \qquad j = 1, \ldots, M$$

where $\gamma_j = \sum_{i=1}^{N} \nu_{ij}$ and p_0 is the pressure of the reactor (take it to be 10^5 Pa).

We call ΔS_j **the entropy change** of reaction j and ΔH_j **the enthalpy change** of reaction j. We have:

$$\Delta S_j = \sum_{i=1}^{N} \nu_{ij} S_i \quad \text{and} \quad \Delta H_j = \sum_{i=1}^{N} \nu_{ij} H_i, , \qquad j = 1, \ldots, M.$$

To calculate these terms, we use **the NASA Polynomials**.

The 7**th order NASA polynomials** are given by

$$\frac{C_{p,i}}{R} = a_{i1} + a_{i2}T + a_{i3}T^2 + a_{i4}T^3 + a_{i5}T^4$$
$$\frac{H_i}{RT} = a_{i1} + \frac{1}{2}a_{i2}T + \frac{1}{3}a_{i3}T^2 + \frac{1}{4}a_{i4}T^3 + \frac{1}{5}a_{i5}T^4 + \frac{a_{i6}}{T}$$
$$\frac{S_i}{R} = a_{i1}\ln(T) + a_{i2}T + \frac{1}{2}a_{i3}T^2 + \frac{1}{3}a_{i4}T^3 + \frac{1}{4}a_{i5}T^4 + a_{i7}$$

for $i = 1, \ldots, N$.

The coefficients a_{ik} are called **the NASA Polynomial Coefficients**. They are given in databases for multiple temperature ranges. For example, the NASA polynomial coefficients for H_2 are given as (T in Kelvin)

k	Low Range: $300 < T < 1000$	High Range: $1000 \leq T < 5000$
1	3.33727920	2.344331122
2	$-4.94024731 \times 10^{-5}$	$7.98052075 \times 10^{-3}$
3	$4.99456778 \times 10^{-7}$	$-1.94781510 \times 10^{-5}$
4	$-1.79566394 \times 10^{-10}$	$2.01572094 \times 10^{-8}$
5	$2.00255376E \times 10^{-14}$	$-7.37611761 \times 10^{-12}$
6	$-9.50158922 \times 10^{2}$	$-9.17935173 \times 10^{2}$
7	-3.20502331	$6.83010238 \times 10^{-1}$

Symbol Notes

Symbol	Meaning
$\mathcal{S}_i$	Chemical symbol of specie i
ν'_{ij}	Stoichiometric coefficients of reactants
ν''_{ij}	Stoichiometric coefficients of products
N	Number of species in system
M	Number of elementary reactions
f_i	Rate of consumption or formation of specie i (reaction rate)
ω_j	Progress rate of reaction j
x_i	Concentration of specie i
k_j	Reaction rate coefficient for reaction j

Chapter 2

Installation

- Clone the repository (git clone https://github.com/cs207G6/cs207-FinalProject.git)
- Change working directory to the root directory of the cloned repository
- Install using “pip install .” or “python setup.py install”
- If desired, run tests using “python setup.py test”

Chapter 3

Basic Usage and Examples

- Get Progress Rate

```
import chemkin.parser
import chemkin.nasa

nasa = chemkin.nasa.NASACoeffs()
# create a data parser class
data_parser = chemkin.parser.DataParser()
# parse the data file and return an instance of ReactionData class
reaction_data = data_parser.parse_file("chemkin/example_data/rxns.xml",
    nasa)
progress_rates = reaction_data.get_progress_rate([1, 2, 3, 4, 5, 6], 100)
print(progress_rates)
```

where the arguments of *get_progress_rate* are chemical concentrations for each of the six species, ordered as specified in the *.xml* file ([1,2,3,4,5,6] in the example above) and reaction temperature (100 in the example above).

output:

```
[  1.06613928e-26   1.85794997e-09   1.20000000e+04]
```

where each value above is corresponding to the progress rate of each elementary reaction.

- Get Reaction Rate

```
reaction_rates = reaction_data.get_reaction_rate(progress_rates)
print(reaction_rates)
```

where the arguments of *get_reaction_rate* is the result of *get_progress_rate* we got above. output:

```
[  1.20000000e+04  -1.85794997e-09  -1.20000000e+04  -1.20000000e+04
   1.20000000e+04  -1.06613928e-26]
```

where each value above is the reaction rate of each species in the order given in the *.xml* file.

- Get Reaction Rate Coefficients

```
ks = reaction_data.get_k(20)
print(ks)
```

where the argument of *get_k* is the temperature (20 in the example above) of the reaction. output:

```
[  1.15383261e-176   3.35659594e-067   1.00000000e+003]
```

where each value above is the reaction rate coefficient calculated based on the type specified in the *.xml* file.

Chapter 4

Future Features

1 Feature 1: UI & Web API

1.1 Motivation

Chemical kinetics can be intimidating for those who are new to this area, with all the complex equations and coefficients. We want to propose a user-friendly interface where the user can access all our package functionality without needing to actually install the package or pass in a file.

Instead of a single answer, some plots may help user to visualize the chemical process. This feature can be useful becasue the UI will display visualizations in various type of plots for example how k changes over different temperatures for a particular reaction. Lastly, we will have a Web API for our package so user can easily integrate out database into there code.

1.2 Discuss the modules that you will write to realize your feature

We will write a module for visualizing the various factor i.e reaction rate over different temperatures and other factors. Once the user inputs the species, their concentrations and all other required information, we will not only provide the reaction's progress rate or reaction rate, but also plots of how respective rate changes across factors i.e temperature. In this way, the user can get an idea of how various factor like temperature effect a reaction's progress or reaction rate.

1.3 Map out the to-dos you plan on implementing

1. Write the Python module that implements the plotting process described above.
2. Build up the website using html/css/javascript depending on the specific features we want on the web page.
3. Create the web API using Python.
4. Integrate the web API and web page with the chemkin file we already have and the plotting model created in step 1
5. Test for unexpected inputs and any other possible errors.

1.4 Overview how you envision the user to use your new feature

On the UI, the user can enter the necessary inputs including reactant species, product species, and concentrations etc. to get the final answer of reaction and progress rate as well as see plots with changing temperature (or other factors), which may help the user better understand/learn chemical kinetics better. In addition,

1.5 External dependencies that your feature will require

Our feature will require an implementation of a webpage which requires coding in html/css/javascript or python GUI and a Web API.

2 Feature 2: Non-Elementary Reaction

2.1 Motivation

Based on the idea that one complete non-elementary reaction consists of a set of elementary reactions, it would be useful if we provide the user with the complete non-elementary reaction given the elementary reactions in the .xml file that the user passes in.

2.2 Discuss the modules that you will write to realize your feature

We will write a module adding up all elementary reactions considering the stoichiometric coefficients given in the input .xml file and clean them into a balanced non-elementary reaction equation.

2.3 Map out the to-dos you plan on implementing

1. Get the reactants and products for each elementary reactions
2. Get the stoichiometric coefficients for each chemical in each elementary reaction
3. Add up all reactions considering all coefficients
4. Clean up the summed equation and return the non-elementary overall reaction equation
5. Add a method in ReactionData that calls the model created above and returns the overall non-elementary reaction
6. Edit the web page/web API so that showing the non-elementary equation becomes a option for the user.

2.4 Overview how you envision the user to use your new feature

If the user is using the web page, they just need to select the "show overall reaction" option to see the overall non-elementary reaction. If the user installs our package, they need to call the method we implement in step 2.3.5

2.5 External dependencies that your feature will require

We need to look up literatures about how elementary reactions are combined into one complete non-elementary reaction. Whether certain factors need special consideration during the process of combination is the question we need to investigate a bit before we implement the model.