UConnRCMPy: Python-based Data Analysis for Rapid Compression Machines

Bryan Weber
Chih-Jen Sung
University of Connecticut
Rapid Compression Machines

• High pressure and low temperature conditions
• Minimize effects of fluid mechanics and inhomogeneity
• 25+ RCMs in use around the world
What do we measure?

A piezo-transducer outputs a charge signal as pressure changes→ converted to voltage, recorded by computer.
Voltage Trace

Noisy!
Voltage Trace

Offset from 0 V
What do we measure?

• A dynamic pressure transducer produces a charge output that is converted to a 0–10 V output
• Nominally, the initial voltage before compression is 0 V
• Ideally, the signal will be free of noise
• The voltage must be processed to compute the pressure, temperature, and ignition delay
Problems

• The signal is noisy → Error in $P_C$
• There is an offset in the initial voltage → Error in $P_0, T_C$
• There are 25+ RCMs in the world, and everyone uses a different processing procedure
• Reproducibility is important!
Let’s use Python to write a data analysis framework with the following goals:

1. Reproducible analysis across researchers
2. Documented design choices for filter criteria, etc.
3. Citable, open-source publication of code

**UConnRCMPy**

[https://github.com/bryanwweber/UConnRCMPy](https://github.com/bryanwweber/UConnRCMPy)
Features of UConnRCMPy

• Low-pass filtering the raw voltage trace
  • Automatic filter cutoff frequency selection

• Converting the voltage trace into a pressure trace

• Processing the pressure trace to determine
  • ignition delay(s)
  • machine-specific effects on the experiment

• Calculating $T_C$ from experiments
Filter Cutoff Frequency affects residuals

Construct low-pass filters with varying cutoff frequencies

Filter the voltage trace and calculate the root mean square residual of the filtered signal relative to the original signal.
Linear fit to residuals to select optimum

Duarte (2014) URL: https://goo.gl/GoCGfq
Yu et al. (1999) DOI: 10.1123/jab.15.3.318
Linear fit to residuals to select optimum

Duarte (2014) URL: [https://goo.gl/GoCGfq](https://goo.gl/GoCGfq)
Yu et al. (1999) DOI: 10.1123/jab.15.3.318
Optimum where $y$-intercept crosses residuals

![Graph showing root mean square residuals against cutoff frequency with marked interception point and optimal frequency](image)

Duarte (2014) URL: [https://goo.gl/GoCGfq](https://goo.gl/GoCGfq)

Yu et al. (1999) DOI: 10.1123/jab.15.3.318
Filtering the Voltage Trace
Correcting the Offset

\[ V = \overline{V} - \overline{V}(0) \]
Computing Ignition Delay

\[
P = V \ast F + P_0
\]

\[
\tau = \max \left( \frac{dP}{dt} \right)
\]

\[
\frac{dP}{dt} \rightarrow \text{Second order forward difference}
\]
Modeling facility effects

- Replace oxygen with nitrogen and run the experiment again
Modeling Facility Effects

- Need the volume of the reactor as a function of time (not measured)
- Reaction chamber modeled as isentropic compression followed by isentropic expansion
- Volume trace calculated from pressure trace

Experimental Pressures
Modeling Facility Effects

- The temperature at the end of compression is found by applying the compression/expansion process to the law of conservation of energy

\[ c_v \frac{dT}{dt} = -P \frac{dv}{dt} - \sum_k u_k \frac{dY_k}{dt} \]

- Non-reactive and reactive temperatures agree at the end of compression
Outputs from UConnRCMPy

• We output the volume as a function of time for use in simulations
  • volume.csv

• We output the pressure as a function of time for comparison
  • Tc_P0_T0_pressure.txt

• We output the choices of important parameters relevant to reproducing the analysis
  • volume_trace.yaml

• We output the values of $P_C$, $P_0$, $T_C$, $T_0$, $\tau$, and the optimal filter frequency for reporting
Modular Design

• Enables modifications for different file formats with consistent choices of filtering criteria, etc.
Scientific Python Software

• SciPy ([https://github.com/scipy/scipy](https://github.com/scipy/scipy)) for filter construction and convolution

• Cantera ([https://github.com/Cantera/cantera](https://github.com/Cantera/cantera)) to calculate thermodynamic information about the reactor

• Matplotlib ([https://github.com/matplotlib/matplotlib](https://github.com/matplotlib/matplotlib)) for plots

• Documentation is available online ([http://bryanwweber.github.io/UConnRCMPy/](http://bryanwweber.github.io/UConnRCMPy/)), generated by Sphinx
Demo
10th NCM 2017 UConnRCMPy demo

```
In [1]:
import uconnrcmpy as ucr
import os
from pathlib import Path
import yaml
print(ucr.__version__)
%matplotlib qt5
print(os.listdir('.'))
```

```
3.0.1
['.ipynb_checkpoints', '00_in_02_mm_373K-1282t-100x-19-Jul-15-1626.txt', '00_in_02_mm_373K-1282t-100x-19-Jul-15-1633.txt', '00_in_02_mm_373K-1282t-100x-19-Jul-15-1640.txt', '00_in_02_mm_373K-1282t-100x-19-Jul-15-1646.txt', '00_in_02_mm_373K-1285t-100x-19-Jul-15-1620.txt', 'demo.ipynb', 'NR_00_in_02_mm_373K-1278t-100x-19-Jul-15-1652.txt', 'species.cti', 'Untitled.ipynb']
```
Create the Condition

In [2]:
cond_00_in_02_mm = ucr.Condition(.cti_file='./species.cti')

Start adding experiments using the input field

In [3]:
cond_00_in_02_mm.add_experiment()

Filename: 00_in_02_mm_373K-1282t-100x-19-Jul-15-1626.txt

Finish the reactive experiments using the argument to
add_experiment()

In [4]:
cond_00_in_02_mm.add_experiment('00_in_02_mm_373K-1282t-100x-19-Jul-15-1620.txt')
cond_00_in_02_mm.add_experiment('00_in_02_mm_373K-1282t-100x-19-Jul-15-1633.txt')
cond_00_in_02_mm.add_experiment('00_in_02_mm_373K-1282t-100x-19-Jul-15-1640.txt')
cond_00_in_02_mm.add_experiment('00_in_02_mm_373K-1282t-100x-19-Jul-15-1646.txt')

Determine which is the representative experiment and add it to the
Condition instance

In []:
reacfile = '00_in_02_mm_373K-1282t-100x-19-Jul-15-1633.txt'
cond_00_in_02_mm.reactive_file = reacfile

Add the non-reactive experiment, still using add_experiment()

In []:
cond_00_in_02_mm.add_experiment(Path('NR_00_in_02_mm_373K-1278t-100x-19-Jul-15-1652.txt'))

This non-reactive trace matches well, so we can proceed. First, define the
remaining quantities in the Condition instance

In []:
nonfile = 'NR_00_in_02_mm_373K-1278t-100x-19-Jul-15-1652.txt'
cond_00_in_02_mm.nonreactive_file = nonfile
cond_00_in_02_mm.compression_time = 36 # ms
cond_00_in_02_mm.nonreactive_end_time = 400 # ms

All experiments

Single Experiment
Non-reactive comparison

Full simulation

New output files

Add the non-reactive experiment, still using add_experiment()

In [6]: cond_00_in_02_mm.add_experiment(Path('NR_00_in_02_mm_373K-1278t-100x-19-Jul-15-1652.txt'))

This non-reactive trace matches well, so we can proceed. First, define the file containing the non-reactive experiment

In [7]: nonfile = 'NR_00_in_02_mm_373K-1278t-100x-19-Jul-15-1652.txt'
cond_00_in_02_mm.nonreactive_file = nonfile

Then we need to create the volume trace used for modeling

In [8]: cond_00_in_02_mm.create_volume_trace()

Specify a value for the nonreactive end time: 400
Specify a value for the reactive end time: 80
Specify a value for the reactive compression time: 36

Then run the simulation to determine $T_C$

In [9]: cond_00_in_02_mm.compare_to_sim(run_reactive=True, run_nonreactive=True)

...
Installation

conda install -c bryanwweber uconnrcmpy

pip install uconnrcmpy
Future Work

• Improved detection of the EOC
• Improved detection of two-stage ignition
• (More) unit testing!
• See https://github.com/bryanwweber/UConnRCMPy/issues
Acknowledgements

This work was funded by the National Science Foundation under Grant No. CBET-1402231

Email: bryan.weber@uconn.edu
Github: @bryanwweber
Web: bryanwweber.com
This work is licensed under the Creative Commons Attribution 4.0 International License. To view a copy of this license, visit http://creativecommons.org/licenses/by/4.0/.