The Onyx Program

Alexander C. Brueckner,[†] O. Maduka Ogba,[‡] Sebastian L. Cevallos,[‡] Daniel W. Walden,[†] Daniel J. O'Leary,[‡] Paul H.–Y. Cheong[†]

[†] Department of Chemistry, Oregon State University, 153 Gilbert Hall, Corvallis, Oregon 97331, United States
[‡] Department of Chemistry, Pomona College, Claremont, California 91711, United States

1: Introduction and Overview

Computational chemists often use experimental isotope effect values to validate theoretical models (i.e. exact atomic motions and positions) during chemical transformations. Unfortunately, the steps required to procure computationally determined isotope effect values are arduous. Chemists must manually parse large quantities of information from files after quantum mechanical (QM) computations and manipulate the data using spreadsheets with several complex equations. To simplify the process, we have developed a user-friendly "black box" program, appropriately named Onyx, using the Python coding language[1] that parses and manipulates data for the user with minimal guidance.

To prepare necessary information for the Onyx program, the chemist first runs QM optimization jobs using the Gaussian 09 computational package.[2] For kinetic isotope effect (KIE) cases, the optimizations should be for one transition state with the corresponding ground state structures. For equilibrium isotope effect (EIE) cases, the optimization should be for two ground state structures. Upon completion, the user can either provide the Onyx program with the formatted checkpoint (.fchk) file created during the computation (Figure 1, left) or compute frequency values using the freq=readisotopes keyword[3] and running four separate computations (i.e. for KIE: 2 ground states; one with the labeled isotope, one unlabeled and 2 transition states; one labeled, one unlabeled). The user can then provide the Onyx program with the output (.log or .out) files created during the vibrational frequency computation (Figure 1, right).

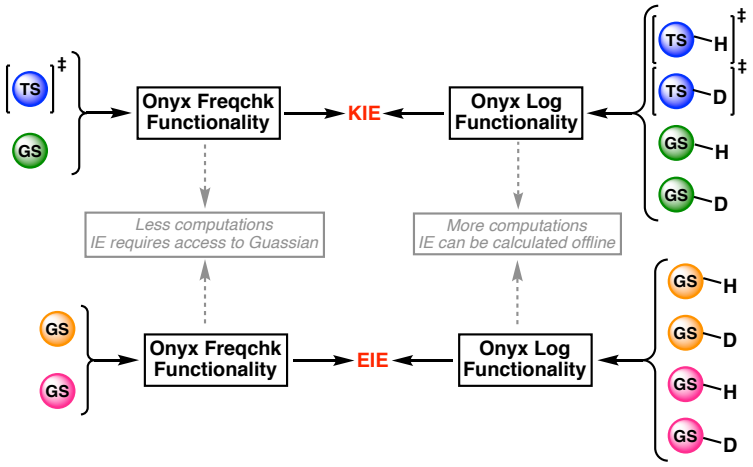


Figure 1. Visual representation of ways to interact with the Onyx program. Utilizing the freqchk or log functionalities to calculate (top) kinetic isotope effects, or (bottom) equilibrium isotope effects.

The user has two ways of interacting with the Onyx program. The first way utilizes the Python module Tkinter[1] to provide an intuitive graphical user interface (GUI). All required information is clearly marked with text boxes and buttons that allow the user to customize the isotope effect calculation to their needs. This option is ideal for users without extensive coding background. The second way to interact with the Onyx program is through the command line with any terminal interface. The user is prompted with several required input fields to set up a simple calculation. For those wanting more customization, several advanced options are also provided (for an example, see Section 4: Advanced Functionality Tutorial). This method is ideal for users with a better understanding of command-line interfaces or for those who want more control over the Onyx program.

The Onyx program accelerates research by rapidly calculating isotope effect values for theoretical models. The Onyx program spends ~15 seconds per calculation while the average person spends upwards of 300 seconds per calculation. This amounts to a ~20x increase in speed compared to manual prediction of isotope effects from vibrational frequencies. This not only streamlines the research process, but also enables researchers to enter areas that were previously unavailable due to large and unmanageable amounts of data (i.e. isotope effect experiments in large biological systems).

2: Installation

2.1: Unix

2.1.1: Xcode

- Xcode is found in the App Store. This is a big download (~5GB)!
- Once installed, open your favorite terminal app and type: `sudo xcodebuild -license`

2.1.2: MacPorts

- Download here: https://www.macports.org/install.php
- Follow on-screen installation instructions

2.1.3: Python 2.7

- In your favorite terminal app, copy and paste each of the following lines one at a time:

```
sudo port install python27
sudo port select python python27
```

2.1.4: NumPy

- In your favorite terminal app, copy and paste the following line:

```
sudo port install py27-numpy
```

2.1.5: Tkinter

- In your favorite terminal app, copy and paste the following line:

```
sudo port install py27-tkinter
```

2.1.6: ttk

- Download the tar.gz file here: https://pypi.python.org/pypi/pyttk
- In your favorite terminal app, copy and paste each of the following lines one at a time:

```
cd ~/Downloads (assuming package was downloaded into Downloads directory)
tar -xzvf pyttk-0.X.X.tar.gz (where X.X = version number)
cd pyttk-0.X.X
sudo python setup.py install
```

2.1.7: argparse

- Download the tar.gz file here: https://pypi.python.org/pypi/argparse
- In your favorite terminal app, copy and paste each of the following lines one at a time:

```
cd ~/Downloads (assuming package was downloaded into Downloads directory)
tar -xzvf argparse-X.X.tar.gz (where X.X = current version)
cd argparse-X.X.X
sudo python setup.py install
```

2.2: Windows

2.2.1: Python 2.7

- Download Python 2.7.13 from https://www.python.org/downloads/release/python-2713
- Follow on-screen installation instructions

2.2.2: NumPy

- Download "numpy-1.13.1-cp27-none-win_amd64.whl" to Downloads directory from https://pypi.python.org/pypi/numpy#downloads
- Click and save "get-pip" to Downloads directory (should be second answer) from https://stackoverflow.com/questions/11200137/installing-numpy-on-64bit-windows-7-with-python-2-7-3
- In your favorite terminal app, copy and paste each of the following lines one at a time:

```
cd C:\Users\%USERNAME%\Downloads (assuming files in Downloads directory)
```

```
get-pip.py
copy numpy-1.13.1-cp27-none-win_amd64.whl C:\Python27\Scripts
cd C:\Python27\Scripts
pip2.7.exe install "numpy-1.13.1-cp27-none-win_amd64.whl"
```

3: Basic Functionality Tutorial

3.1: Introduction

In this tutorial, we examine a kinetic isotope effect (KIE) for the C-N rotation of formamide which we previously used as a case study for calculating isotope effects using spreadsheets.[4] In 1992, Perrin and co-workers reported NMR measurements of secondary conformational kinetic isotope effects associated with C-N bond rotation in several amides.[5] Formamide exhibited a k_H/k_D of 1.16 ± 0.10 at 25°C. The transition state and ground states are shown in Figure 2. Provided structure files were optimized and frequency computations were completed at the HF[6]/6-31G(d)[7] level of theory.

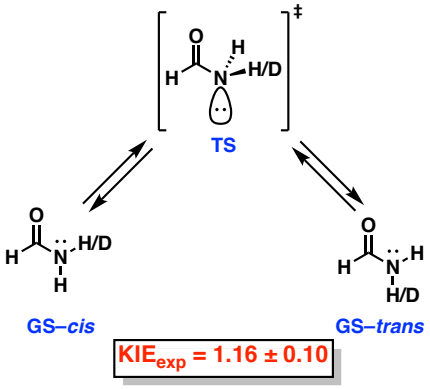


Figure 2. C-N bond rotation of formamide exhibiting a normal secondary conformational kinetic isotope effect.

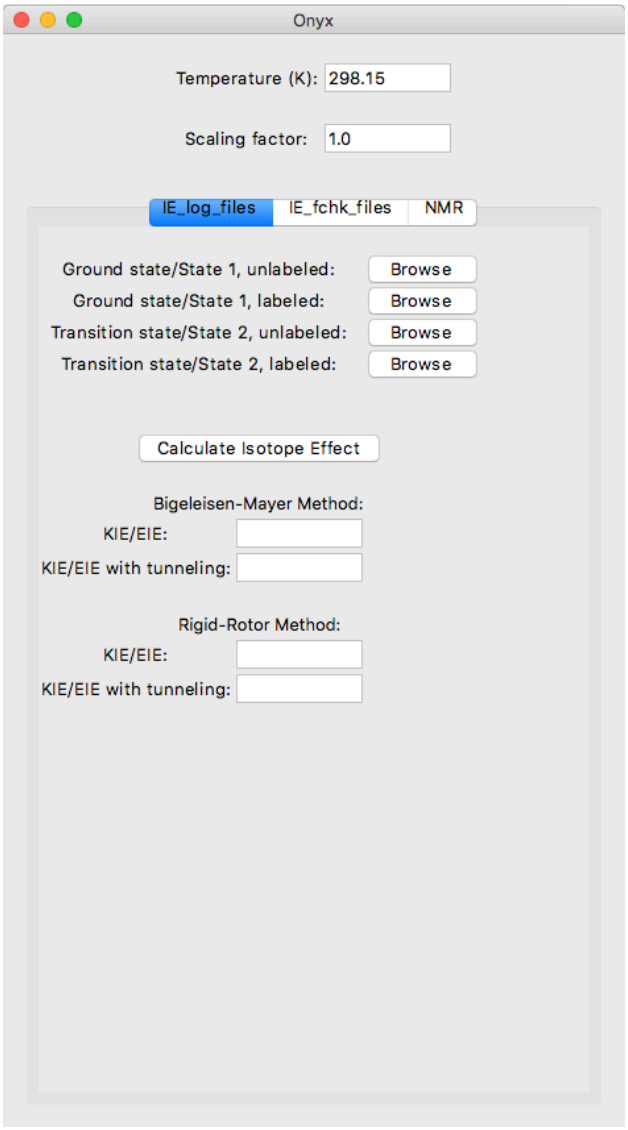
3.2: Using The GUI

3.2.1: Log Functionality

1) Open your favorite terminal interface.
2) If the Onyx program is in your PYTHONPATH, type `Onyx.py` and hit enter.
    a. If Onyx is not in your PYTHONPATH, navigate to the directory the Onyx program is in and type `Onyx.py` and hit enter.

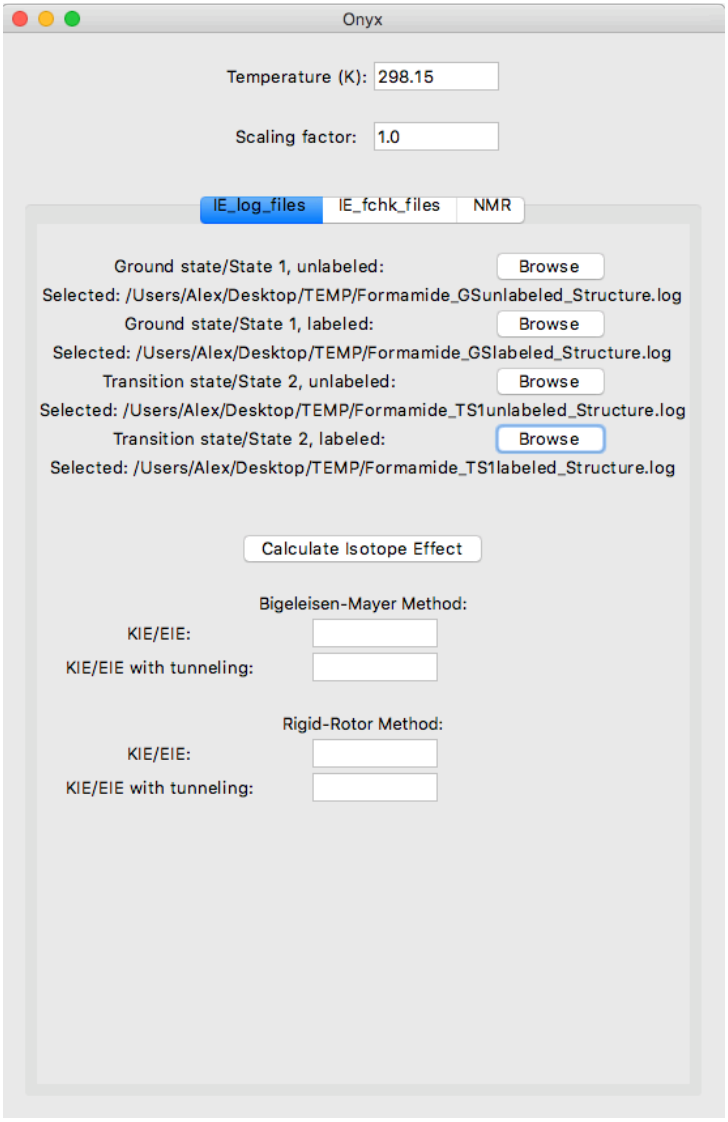
```
[MacBookPro:Alex:~ 55119] Onyx.py
```

3) A GUI screen will open.

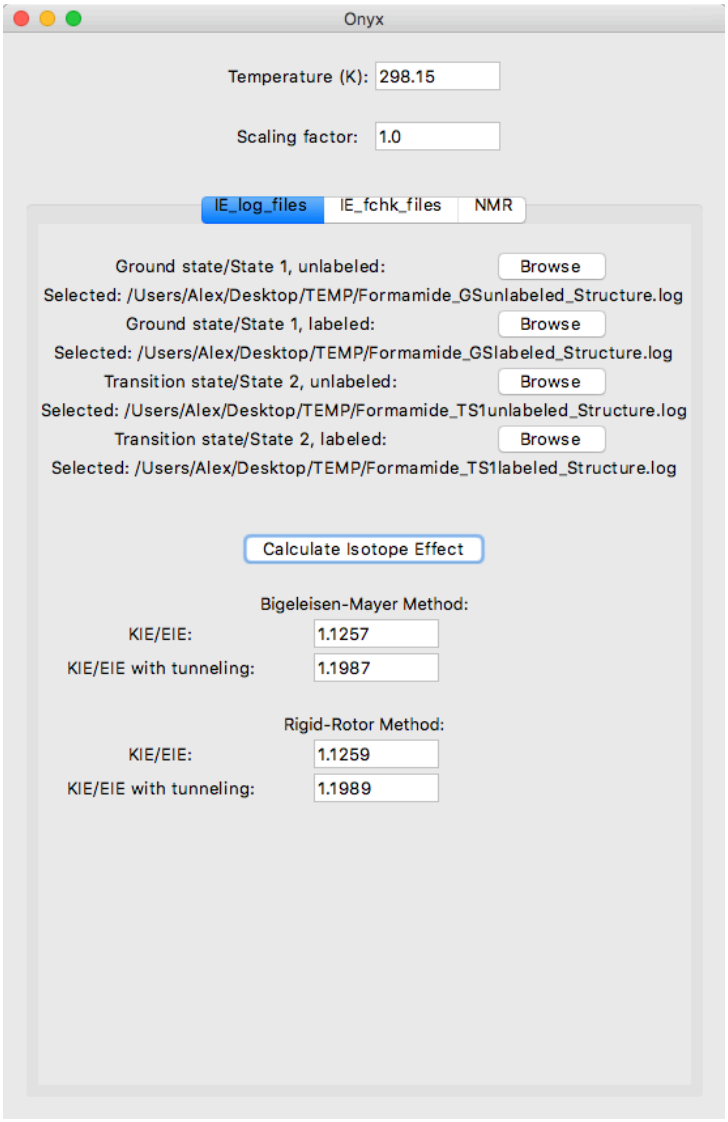


4) Leave the temperature and scaling factor as default for this example.

5) Click each Browse button and navigate to the location of the log files.
   a. Ground state/State1, unlabeled: "**Formamide_GSunlabeled_Structure.log**"
   b. Ground state/State1, labeled: "**Formamide_GSlabeled_Structure.log**"
   c. Transition state/State2, unlabeled: "**Formamide_TSunlabeled_Structure.log**"
   d. Transition state/State2, labeled: "**Formamide_TSlabeled_Structure.log**"



6) Press the “Calculate Isotope Effect” button.

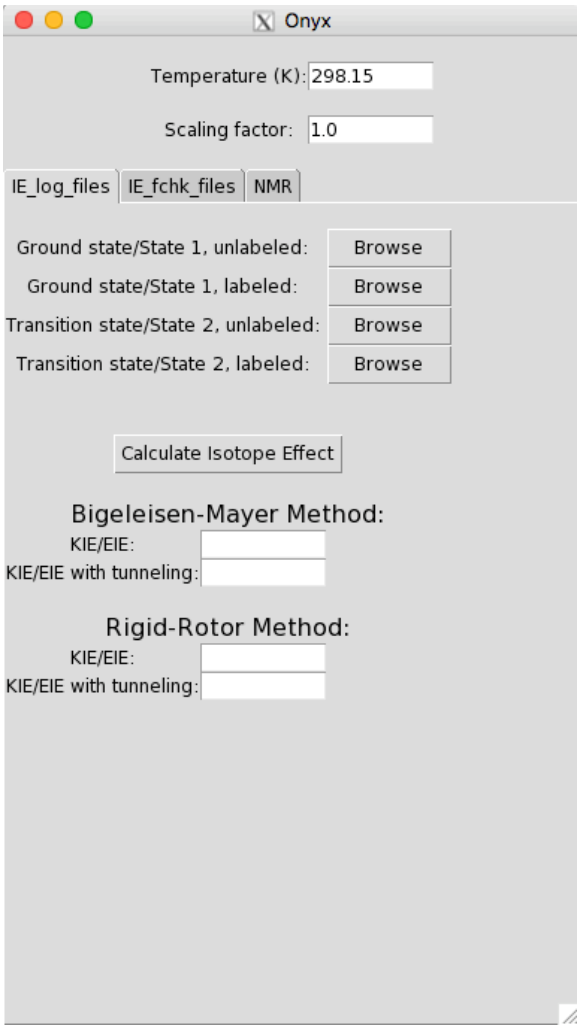


3.2.2: Freqchk Functionality

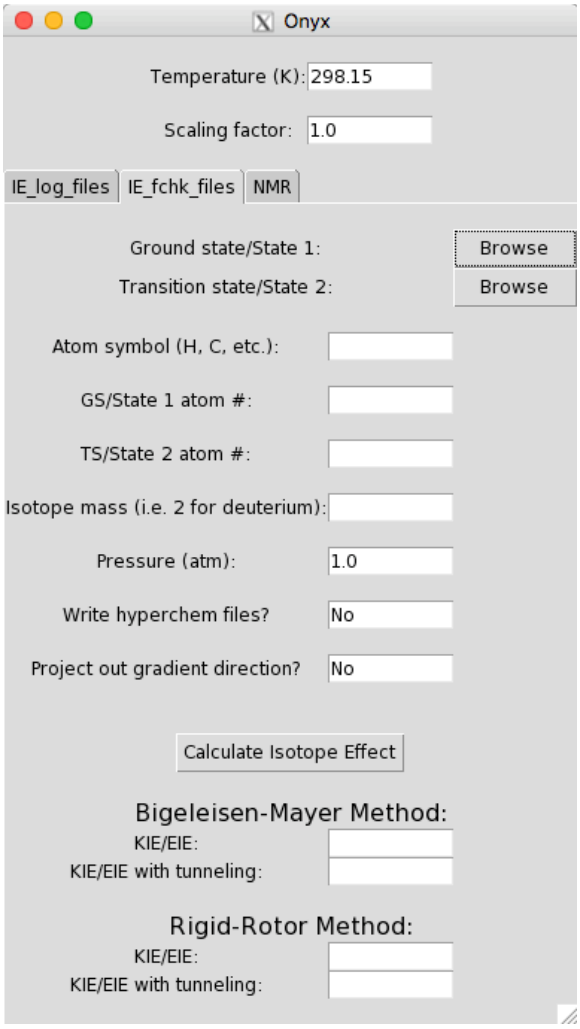
1) Make sure you have access to Gaussian 09 utilities!
2) Open your favorite terminal interface.
3) If the Onyx program is in your PYTHONPATH, type `Onyx.py` and hit enter.
    a. If Onyx is not in your PYTHONPATH, navigate to the directory the Onyx program is in and type `Onyx.py` and hit enter.

```
[MacBookPro:Alex:~ 55119] Onyx.py
```

4) A GUI screen will open.

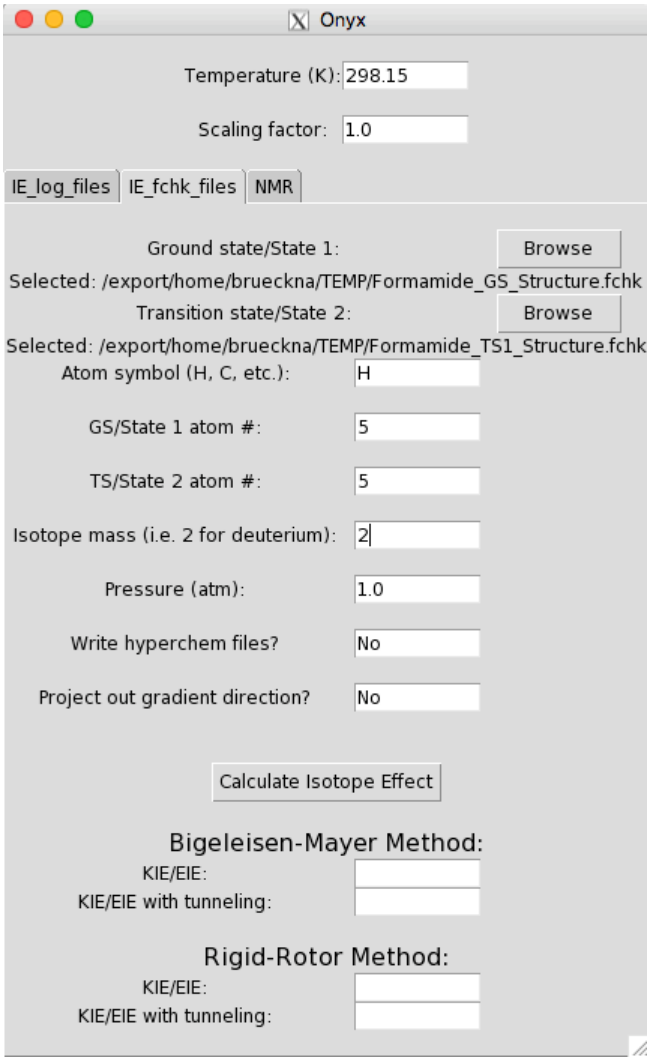


5) Click the “IE_fchk_files” tab.

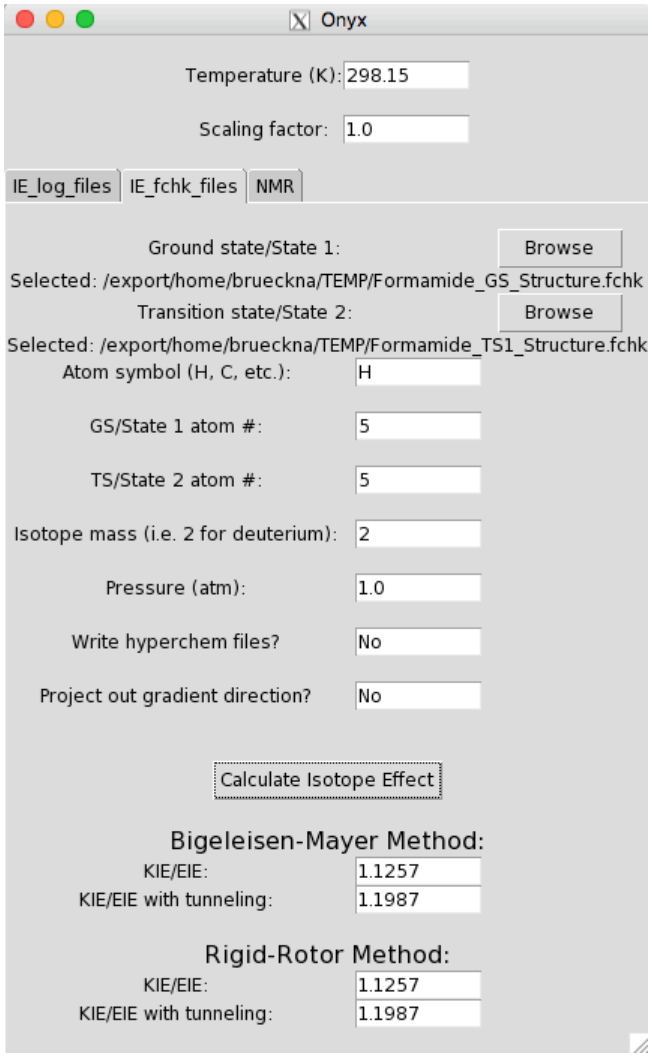


6) Leave the temperature and scaling factor as default for this example.
7) Click each Browse button and navigate to the location of the fchk files.
    a. Ground state/State1: “**Formamide_GS_Structure.fchk**”
    b. Transition state/State2: “**Formamide_TS1_Structure.fchk**”

8) Fill out the remaining text boxes:



7) Press the “Calculate Isotope Effect” button.



3.2.3: NMR Functionality

NEED OUTLINE

3.3: Using the Command Line

3.3.1: Log Functionality

1) To get an overview of the Onyx program functionality, type `Onyx.py -h` in your terminal (assuming Onyx.py is in your PYTHONPATH or you are in the directory that contains Onyx.py).

```
[MacBookPro:Alex:~ 55123] Onyx.py -h
usage: Onyx.py [-h] [-fp] [-lp] [-fi FCHKTXTNAME] [-li LOGTXTNAME]

A program to automatically compute isotope effects. To use the Graphical User
Interface (GUI), simply type 'python Onyx.py'.

optional arguments:
  -h, --help            show this help message and exit
  -fp, --freqchkprint   print input file for freqchk functionality. Fill out
                        txt file with necessary information
  -lp, --logprint       print input file for G09 log file functionality. Fill
                        out txt file with necessary information
  -fi FCHKTXTNAME, --freqchkinput FCHKTXTNAME
                        Calculate IE utilizing freqchk functionality. Pass in
                        filled out text input file (generated by -fi option)
  -li LOGTXTNAME, --loginput LOGTXTNAME
                        Calculate IE utilizing log file functionality. Pass in
                        filled out text input file (generated by -li option)
```

2) To generate the input for the log functionality, type `Onyx.py -lp`. A file will be generated in the current directory with the date and time in the name.

```
[MacBookPro:Alex:~ 55125] Onyx.py -lp
Please fill out the text file that was just populated in the current directory.
```

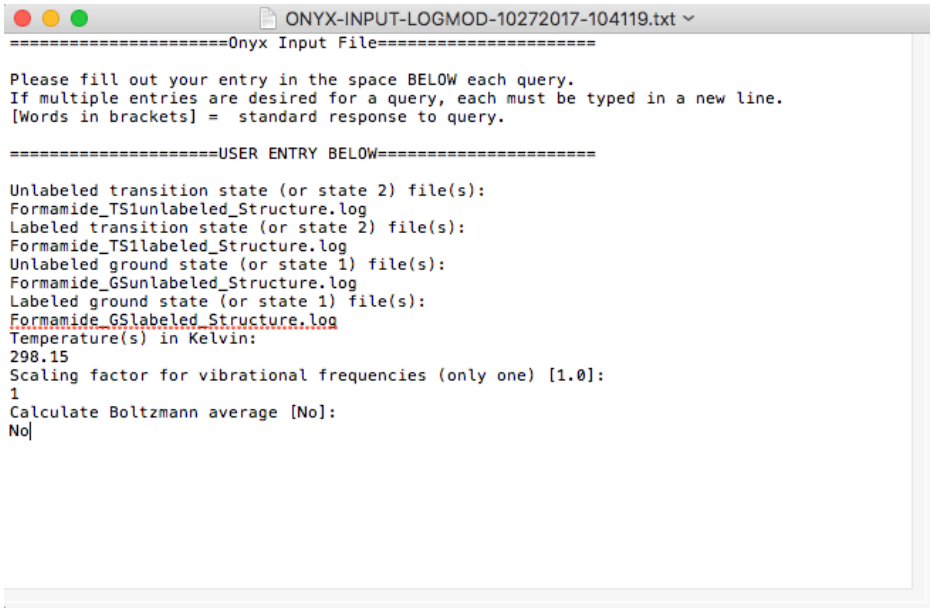
3) Type `open ONYX-INPUT-LOGMOD-XXX-YYY.txt` where XXX is the date and YYY is the time.

```
[MacBookPro:Alex:~ 55126] open ONYX-INPUT-LOGMOD-10272017-104119.txt
```

ONYX-INPUT-LOGMOD-10272017-104119.txt

```
======================Onyx Input File======================

Please fill out your entry in the space BELOW each query.
If multiple entries are desired for a query, each must be typed in a new line.
[Words in brackets] =  standard response to query.

====================USER ENTRY BELOW=====================

Unlabeled transition state (or state 2) file(s):

Labeled transition state (or state 2) file(s):

Unlabeled ground state (or state 1) file(s):

Labeled ground state (or state 1) file(s):

Temperature(s) in Kelvin:

Scaling factor for vibrational frequencies (only one) [1.0]:

Calculate Boltzmann average [No]:
```

4) Fill out the txt file and save and quit:



5) Type `Onyx.py -li ONYX-INPUT-LOGMOD-XXX-YYY.txt` where XXX is the date and YYY is the time and hit enter.

```
[MacBookPro:Alex:Desktop/TEMP 55147] Onyx.py -li ONYX-INPUT-LOGMOD-10272017-104119.txt

-------- Initializing Onyx Program --------

-- Parsing data for BM calculation 1 --
-- Manipulating data to get IE for BM calculation 1 --
-- Isotope effect successfully calculated for BM calculation 1 --

-- Parsing data for RR calculation 1 --
-- Manipulating data to get IE for RR calculation 1 --
-- Isotope effect successfully calculated for RR calculation 1 --

RESULTS:
============================================================ Calculation 1 ============================================================

       Temperature:            298.15
       Scale factor:           1.00
       GS_file_unlabeled:      Formamide_GSunlabeled_Structure.log
       GS_file_labeled:        Formamide_GSlabeled_Structure.log
       TS_file_unlabeled:      Formamide_TS1unlabeled_Structure.log
       TS_file_labeled:        Formamide_TS1labeled_Structure.log


BM (KIE) = SYM x MMI x EXC x ZPE
SYM             MMI             EXC             ZPE             KIE             KIE_tunneling
-------------------------------------------------------------------------------------------------
1.0             1.0188          1.0916          1.0122          1.1257          1.1987
-------------------------------------------------------------------------------------------------

RR (KIE) = ddH x ddS
ddH             ddS             KIE             KIE_tunneling
-------------------------------------------------------------
0.9601          1.1726          1.1259          1.1989
-------------------------------------------------------------
```

3.3.2: Fchk Functionality

1) Make sure you have access to Gaussian 09 utilities!
2) To get an overview of the Onyx program functionality, type `Onyx.py -h` in your terminal.

```
[MacBookPro:Alex:~ 55123] Onyx.py -h
usage: Onyx.py [-h] [-fp] [-lp] [-fi FCHKTXTNAME] [-li LOGTXTNAME]

A program to automatically compute isotope effects. To use the Graphical User
Interface (GUI), simply type 'python Onyx.py'.

optional arguments:
  -h, --help            show this help message and exit
  -fp, --freqchkprint   print input file for freqchk functionality. Fill out
                        txt file with necessary information
  -lp, --logprint       print input file for G09 log file functionality. Fill
                        out txt file with necessary information
  -fi FCHKTXTNAME, --freqchkinput FCHKTXTNAME
                        Calculate IE utilizing freqchk functionality. Pass in
                        filled out text input file (generated by -fi option)
  -li LOGTXTNAME, --loginput LOGTXTNAME
                        Calculate IE utilizing log file functionality. Pass in
                        filled out text input file (generated by -li option)
```

3) To generate the input for the fchk functionality, type `Onyx.py -fp`. A file will be generated in the current directory with the date and time in the name.

```
[Woodward:~ 3002] Onyx.py -fp
Please fill out the text file that was just populated in the current directory.
```

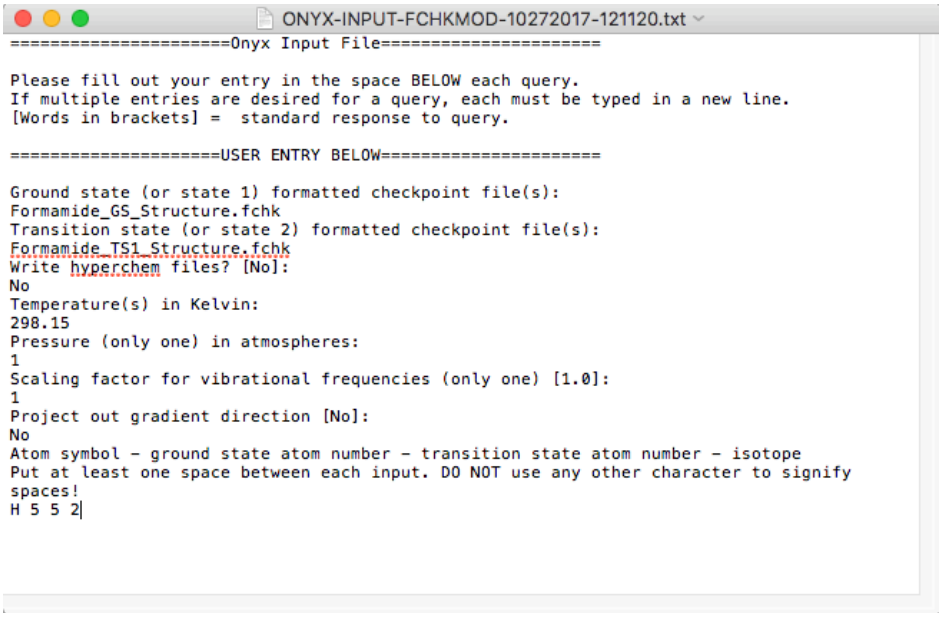
4) Type `open ONYX-INPUT-FCHKMOD-XXX-YYY.txt` where XXX is the date and YYY is the time.

```
[MacBookPro:Alex:Desktop/TEMP 55187] open ONYX-INPUT-FCHKMOD-10272017-121120.txt
```

ONYX-INPUT-FCHKMOD-10272017-121120.txt

```
=====================Onyx Input File=======================

Please fill out your entry in the space BELOW each query.
If multiple entries are desired for a query, each must be typed in a new line.
[Words in brackets] =  standard response to query.

====================USER ENTRY BELOW=======================

Ground state (or state 1) formatted checkpoint file(s):

Transition state (or state 2) formatted checkpoint file(s):

Write hyperchem files? [No]:

Temperature(s) in Kelvin:

Pressure (only one) in atmospheres:

Scaling factor for vibrational frequencies (only one) [1.0]:

Project out gradient direction [No]:

Atom symbol - ground state atom number - transition state atom number - isotope
Put at least one space between each input. DO NOT use any other character to signify
spaces!
```

5) Fill out the txt file and save and quit:



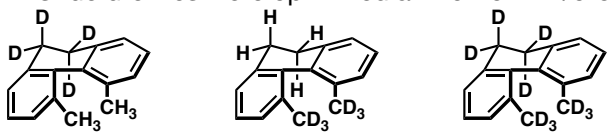
6) Type `Onyx.py -fi ONYX-INPUT-FCHKMOD-XXX-YYY.txt` where XXX is the date and YYY is the time and hit enter.

```
[Woodward:~ 3002] Onyx.py -fi ONYX-INPUT-FCHKMOD-10272017-121120.txt

-------- Initializing Onyx Program --------


-- Running freqchk for calculation 1 --
-- Parsing data for BM calculation 1 --
-- Manipulating data to get IE for BM calculation 1 --
-- Isotope effect successfully calculated for BM calculation 1 --

-- Parsing data for RR calculation 1 --
-- Manipulating data to get IE for RR calculation 1 --
-- Isotope effect successfully calculated for RR calculation 1 --

RESULTS:
================================================== Calculation 1 ==================================================

        Temperature:            298.15
        Scale factor:           1.00
        GS_file:                Formamide_GS_Structure.fchk
        TS_file:                Formamide_TS1_Structure.fchk
        GS_atom_number(s):      H5
        TS_atom_number(s):      H5


BM (KIE) = SYM x MMI x EXC x ZPE
SYM             MMI             EXC             ZPE             KIE             KIE_tunneling
--------------------------------------------------------------------------------------------
1.0             1.019           1.0915          1.0121          1.1257          1.1987
--------------------------------------------------------------------------------------------

RR (KIE) = ddH x ddS
ddH             ddS             KIE             KIE_tunneling
------------------------------------------------------------
0.9601          1.1725          1.1257          1.1987
------------------------------------------------------------
```

4: Advanced Functionality Tutorial

4.1: Introduction

In a previously published study, we investigated isotopic labeling on the rates of stereoinversion on two compounds.[8] In this tutorial, we will be investigating isotopic labeling on 9,10–dihydro–4,5–dimethylphenanthrene (Figure 3) by calculating three separate isotope effect values with a single input file for the Onyx program. All structure files were optimized at the B3LYP[9]/6-31G(d)[7] level of theory.



[D_4] [D_6] [D_{10}]

Figure 3. 9,10–dihydro–4,5–dimethylphenanthrene with three deuterium labeling scenarios.

4.2: Using the Command Line

4.2.1: Fchk Functionality

1) Make sure you have access to Gaussian 09 utilities!
2) To get an overview of the Onyx program functionality, type `Onyx.py -h` in your terminal.

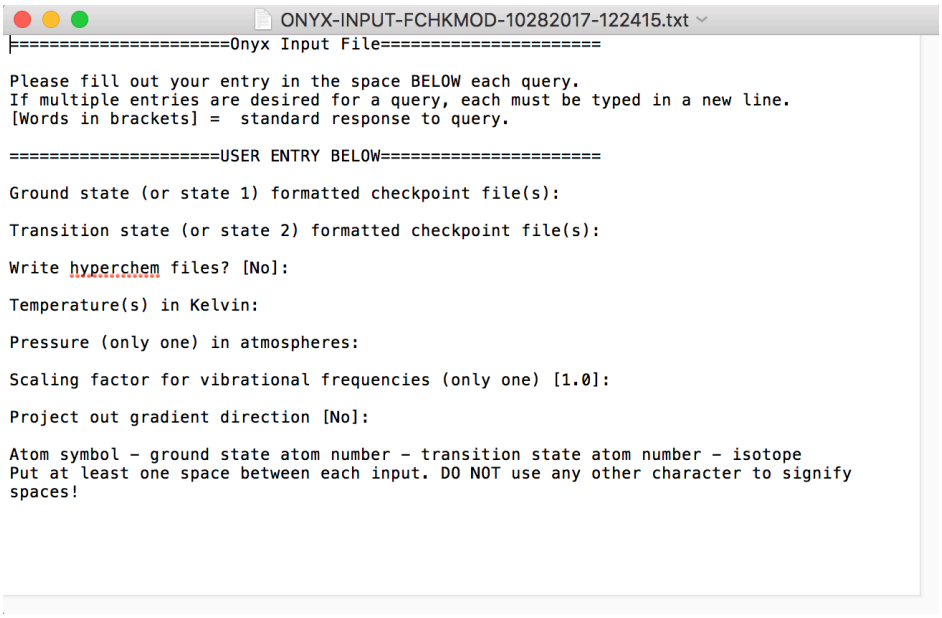
```
[MacBookPro:Alex:~ 55123] Onyx.py -h
usage: Onyx.py [-h] [-fp] [-lp] [-fi FCHKTXTNAME] [-li LOGTXTNAME]

A program to automatically compute isotope effects. To use the Graphical User
Interface (GUI), simply type 'python Onyx.py'.

optional arguments:
  -h, --help            show this help message and exit
  -fp, --freqchkprint   print input file for freqchk functionality. Fill out
                        txt file with necessary information
  -lp, --logprint       print input file for G09 log file functionality. Fill
                        out txt file with necessary information
  -fi FCHKTXTNAME, --freqchkinput FCHKTXTNAME
                        Calculate IE utilizing freqchk functionality. Pass in
                        filled out text input file (generated by -fi option)
  -li LOGTXTNAME, --loginput LOGTXTNAME
                        Calculate IE utilizing log file functionality. Pass in
                        filled out text input file (generated by -li option)
```

3) To generate the input for the fchk functionality, type `Onyx.py -fp`. A file will be generated in the current directory with the date and time in the name.

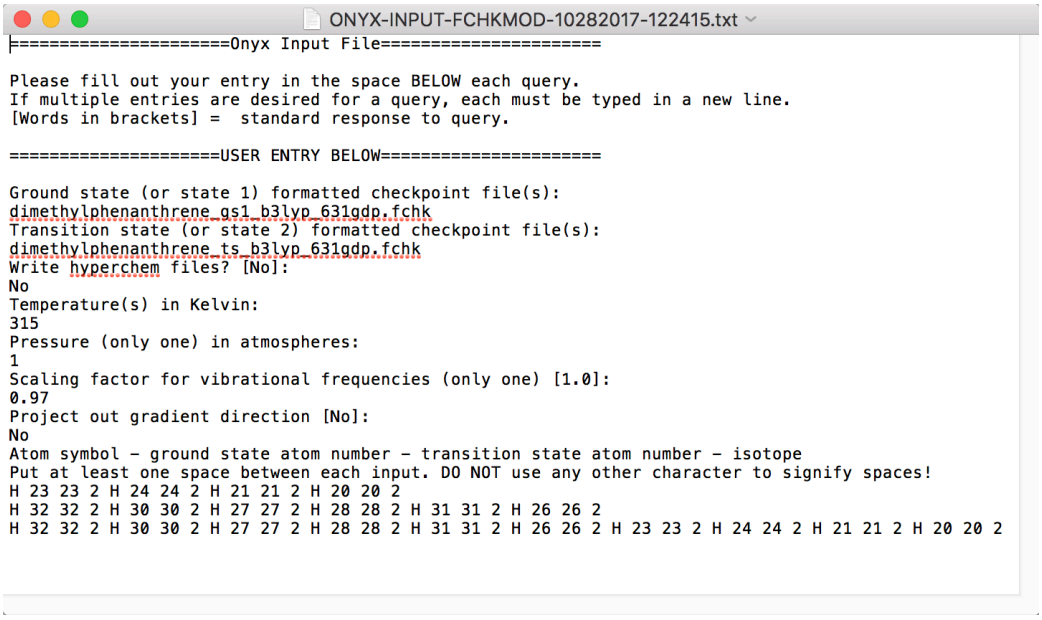
```
[Woodward:~ 3002] Onyx.py -fp
Please fill out the text file that was just populated in the current directory.
```

4) Type `open ONYX-INPUT-FCHKMOD-XXX-YYY.txt` where XXX is the date and YYY is the time.

```
[MacBookPro:Alex:Advanced_functionality/Fchk_Functionality 55280] open ONYX-INPUT-FCHKMOD-10282017-122415.txt
```



5) Fill out the txt file and save and quit. In this example input, the first line is for [D_4], the second is for [D_6] and the last is for [D_{10}] (Figure 3).



6) Type `Onyx.py -fi ONYX-INPUT-FCHKMOD-XXX-YYY.txt` where XXX is the date and YYY is the time and hit enter. The first calculation is for [D_4], the second is for [D_6] and the last is for [D_{10}] (Figure 3).

```
[Woodward:~/prj_Onyx 3179] Onyx.py -fi ONYX-INPUT-FCHKMOD-10282017-122415.txt

-------- Initializing Onyx Program --------


-- Running freqchk for calculation 1 --
-- Parsing data for BM calculation 1 --
-- Manipulating data to get IE for BM calculation 1 --
-- Isotope effect successfully calculated for BM calculation 1 --

-- Parsing data for RR calculation 1 --
-- Manipulating data to get IE for RR calculation 1 --
-- Isotope effect successfully calculated for RR calculation 1 --

-- Running freqchk for calculation 2 --
-- Parsing data for BM calculation 2 --
-- Manipulating data to get IE for BM calculation 2 --
-- Isotope effect successfully calculated for BM calculation 2 --

-- Parsing data for RR calculation 2 --
-- Manipulating data to get IE for RR calculation 2 --
-- Isotope effect successfully calculated for RR calculation 2 --

-- Running freqchk for calculation 3 --
-- Parsing data for BM calculation 3 --
-- Manipulating data to get IE for BM calculation 3 --
-- Isotope effect successfully calculated for BM calculation 3 --

-- Parsing data for RR calculation 3 --
-- Manipulating data to get IE for RR calculation 3 --
-- Isotope effect successfully calculated for RR calculation 3 --

RESULTS:
========================================================= Calculation 1 =========================================================

        Temperature:            315.00
        Scale factor:           0.97
        GS_file:                dimethylphenanthrene_gs1_b3lyp_631gdp.fchk
        TS_file:                dimethylphenanthrene_ts_b3lyp_631gdp.fchk
        GS_atom_number(s):      H23, H24, H21, H20
        TS_atom_number(s):      H23, H24, H21, H20


BM (KIE) = SYM x MMI x EXC x ZPE
SYM             MMI             EXC             ZPE             KIE             KIE_tunneling
-----------------------------------------------------------------------------------------------
1.0             1.0017          1.0064          0.946           0.9537          0.9538
-----------------------------------------------------------------------------------------------

RR (KIE) = ddH x ddS
ddH             ddS             KIE             KIE_tunneling
-------------------------------------------------------------
0.9299          1.0258          0.9538          0.9539
-------------------------------------------------------------
```

```
================================================== Calculation 2 ==================================================

        Temperature:            315.00
        Scale factor:           0.97
        GS_file:                dimethylphenanthrene_gs1_b3lyp_631gdp.fchk
        TS_file:                dimethylphenanthrene_ts_b3lyp_631gdp.fchk
        GS_atom_number(s):      H32, H30, H27, H28, H31, H26
        TS_atom_number(s):      H32, H30, H27, H28, H31, H26


BM (KIE) = SYM x MMI x EXC x ZPE
SYM             MMI             EXC             ZPE             KIE             KIE_tunneling
-------------------------------------------------------------------------------------------
1.0             0.9956          1.1878          0.742           0.8775          0.8791
-------------------------------------------------------------------------------------------

RR (KIE) = ddH x ddS
ddH             ddS             KIE             KIE_tunneling
-----------------------------------------------------------
0.731           1.2003          0.8775          0.8791
-----------------------------------------------------------


================================================== Calculation 3 ==================================================

        Temperature:            315.00
        Scale factor:           0.97
        GS_file:                dimethylphenanthrene_gs1_b3lyp_631gdp.fchk
        TS_file:                dimethylphenanthrene_ts_b3lyp_631gdp.fchk
        GS_atom_number(s):      H32, H30, H27, H28, H31, H26, H23, H24, H21, H20
        TS_atom_number(s):      H32, H30, H27, H28, H31, H26, H23, H24, H21, H20


BM (KIE) = SYM x MMI x EXC x ZPE
SYM             MMI             EXC             ZPE             KIE             KIE_tunneling
-------------------------------------------------------------------------------------------
1.0             0.9973          1.1955          0.7021          0.8371          0.8386
-------------------------------------------------------------------------------------------

RR (KIE) = ddH x ddS
ddH             ddS             KIE             KIE_tunneling
-----------------------------------------------------------
0.6798          1.231           0.8368          0.8384
-----------------------------------------------------------
```

5: References

[1] Python Software Foundation. Python Language Reference, version 2.7. Available at http://www.python.org.

[2] Gaussian 09, Revision **D.01**, Frisch, M. J.; Trucks, G. W.; Schlegel, H. B.; Scuseria, G. E.; Robb, M. A.; Cheeseman, J. R.; Scalmani, G.; Barone, V.; Mennucci, B.; Petersson, G. A.; Nakatsuji, H.; Caricato, M.; Li, X.; Hratchian, H. P.; Izmaylov, A. F.; Bloino, J.; Zheng, G.; Sonnenberg, J. L.; Hada, M.; Ehara, M.; Toyota, K.; Fukuda, R.; Hasegawa, J.; Ishida, M.; Nakajima, T.; Honda, Y.; Kitao, O.; Nakai, H.; Vreven, T.; Montgomery, J. A., Jr.; Peralta, J. E.; Ogliaro, F.; Bearpark, M.; Heyd, J. J.; Brothers, E.; Kudin, K. N.; Staroverov, V. N.; Kobayashi, R.; Normand, J.; Raghavachari, K.; Rendell, A.; Burant, J. C.; Iyengar, S. S.; Tomasi, J.; Cossi, M.; Rega, N.; Millam, J. M.; Klene, M.; Knox, J. E.; Cross, J. B.; Bakken, V.; Adamo, C.; Jaramillo, J.; Gomperts, R.; Stratmann, R. E.; Yazyev, O.; Austin, A. J.; Cammi, R.; Pomelli, C.; Ochterski, J. W.; Martin, R. L.; Morokuma, K.; Zakrzewski, V. G.; Voth, G. A.; Salvador, P.; Dannenberg, J. J.; Dapprich, S.; Daniels, A. D.; Farkas, Ö.; Foresman, J. B.; Ortiz, J. V.; Cioslowski, J.; Fox, D. J. Gaussian, Inc., Wallingford CT, 2009.

[3] For more information, see http://gaussian.com/freq/.

[4] Ogba, O. M.; Thoburn, J. D.; O'Leary, D. J. "Spreadsheet–Based Computational Predictions of Isotope Effects," *Applied Theoretical Organic Chemistry*, **World Scientific**, *In Press*.

[5] Perrin, C. L.; Thoburn, J. D.; Kresge, A. J. "Secondary kinetic isotope effects in C-N rotation of amides," *J. Am. Chem. Soc.*, **1992**, *114*, 8800–8807.

[6] Roothaan, C. C. J. "New Developments in Molecular Orbital Theory," *Rev. Mod. Phys.*, **1951**, *23*, 69-89.

[7] Hehre, W. J.; Ditchfield, R.; Pople, J. A. "Self–Consistent Molecular Orbital Methods. XII. Further Extensions of Gaussian–Type Basis Sets for Use in Molecular Orbital Studies of Organic Molecules," *J. Chem. Phys.*, **1972**, *56*, 2257–2261.

[8] O'Leary, D. J.; Rablen, P. R.; Meyer, M. P. "On the Origin of Conformational Kinetic Isotope Effects," Angew. Chem. Int. Ed., **2011**, *50*, 2564–2567.

[9] Becke, A. D. "Density–functional thermochemistry. III. The role of exact exchange," *J. Chem. Phys.*, **1993**, *98*, 5648–5652.