SANDIA REPORT
SAND2021-12281
Printed October, 2021

Sandia National Laboratories

LIM1TR: Lithium-Ion Modeling with 1-D Thermal Runaway v1.0

Andrew Kurzawski, Randy Shurtz

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185
Livermore, California 94550

Issued by Sandia National Laboratories, operated for the United States Department of Energy by National Technology & Engineering Solutions of Sandia, LLC.

Available to DOE and DOE contractors from

U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-Mail: reports@osti.gov
Online ordering: http://www.osti.gov/scitech

Available to the public from

U.S. Department of Commerce
National Technical Information Service
5301 Shawnee Road
Alexandria, VA 22312

Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-Mail: orders@ntis.gov
Online order: https://classic.ntis.gov/help/order-methods

ABSTRACT

LIM1TR (Lithium-Ion Modeling with 1-D Thermal Runaway) is an open-source code that uses the finite volume method to simulate heat transfer and chemical kinetics on a quasi 1-D domain. The target application of this software is to simulate thermal runaway in systems of lithium-ion batteries. The source code for LIM1TR can be found at https://github.com/ajkur/lim1tr. This user guide details the steps required to create and run simulations with LIM1TR starting with setting up the Python environment, generating an input file, and running a simulation. Additional details are provided on the output of LIM1TR as well as extending the code with custom reaction models. This user guide concludes with simple example analyses of common battery thermal runaway scenarios. The corresponding input files and processing scripts can be found in the "Examples" folder in the on-line repository, with select input files included in the appendix of this document.

ACKNOWLEDGMENT

The authors would like to thank John Hewson for guidance in developing the feature set included in LIM1TR. Beta testing was key to the development process, and the authors thank Loraine Torres-Castro, Ala' Eyad Qatramez, Alexander Headley, and Daniel Foti for testing the code and providing valuable feedback. The authors would also like to thank Alex Bates and Joshua Lamb for reviewing this user guide and greatly improving the quality of this document.

This research was funded by the U.S. Department of Energy Office of Electricity Energy Storage Program through Sandia National Laboratories, under the guidance of Dr. Imre Gyuk.

CONTENTS

LIST OF FIGURES

NOMENCLATURE

Table 0-1.

Abbreviation	Definition
ARC	accelerating rate calorimetry
DSC	differential scanning calorimetry
HRR	heat release rate
LIM1TR	Lithium-Ion Modeling with 1-D Thermal Runaway
MKS	meters, kilograms, seconds unit system
ODE	ordinary differential equation

1. INTRODUCTION

LIM1TR is a finite volume code for modeling thermal runaway in Li-ion batteries. This guide covers the basics of getting started with LIM1TR, an overview of the input file and outputs, the theory and equations LIM1TR solves, additional features, and example problems.

This software was designed with the goal of enabling quick simulation of stacks of pouch or prismatic format cells and calibration of kinetics. These stacks can include batteries and other materials such as structural or spacer materials as seen in Figure 1-1. In the module level simulations, the domain is lumped in the plane of the electrodes and discretized in the through-plane direction. The heat equation is solved in 1-D with reaction source terms and user-specified boundary conditions. For calibration of reaction mechanisms (e.g. against calorimetry data), a single control volume is modeled and boundary conditions can be adiabatic or follow a fixed temperature rate. The code is tested through a series of unit and verification tests located in the Tests folder. Optimization of any user input is achievable through including LIM1TR as a Python module allowing the user to quickly interface with the optimization toolbox of their choice.

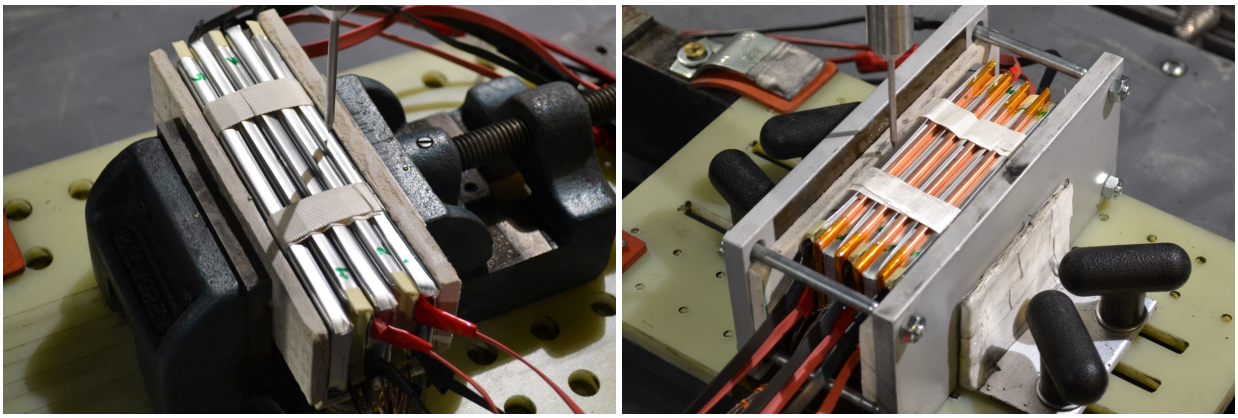

Figure 1-1. Example stacks of pouch cells with various spacer and structural materials from Torres-Castro et al. [11].

This page intentionally left blank.

2. GETTING STARTED

The LIM1TR repository can be cloned from the LIM1TR Github. LIM1TR is currently tested on Linux and Windows operating systems. The following software is required for running LIM1TR:

- Python 2.7 or 3.X
- Numpy
- Scipy
- Pandas
- Matplotlib
- Numba
- PyYAML

Parameterizing the simulations is done with a ".yaml" input file (detailed in Chapter 3). Here the user specifies the material properties, boundary conditions, reactions (if present), computational domain, and temporal discretization. We note that the units of the input and output are in MKS. To run a simulation via the command line, it is helpful to navigate to the location of the desired input file. The following command will launch a simulation:

```
$ python <path-to-lim1tr>/main_fv.py input_file.yaml
```

By default, the simulation progress will be printed to the screen while the temperatures and species concentrations (if present) at each node and a set of time steps are written to a series of output files in the current directory. Each output file contains a set number of time steps, and all files are combined into a single file upon completion of the simulation. The contents of the pickled output are detailed in Chapter 4.

This page intentionally left blank.

3. THE INPUT FILE

The input file provides the interface for setting up a simulation. The ".yaml" file format is used for easy parsing into nested Python dictionaries using the PyYAML module. When writing a .yaml file, indentations with spaces are required to define the hierarchy of the input parameters. Syntactically, key-value pairs are denoted by colons (e.x. key: value), and nested blocks of inputs are specified by a keyword followed by a colon and indented new line.

```
Keyword:
    Nested key: nested value
```

A few additional notes on syntax will assist user who are new to the ".yaml" format. Strings do not require quotations, but they are suggested as single or double quotes will force the keys or values to be parsed as strings. This may be useful in the instance where the user desires to name a key with an integer (i.e. '10'). Boolean variables are specified with 1 and 0 for true and false, respectively. Lists can be specified by either square brackets with entries separated by commas

```
example list: [1, 2, 3]
```

or by an indented new line and hyphens.

```
example list:
    - 1
    - 2
    - 3
```

There are seven main input sections: Materials, Domain Table, Time, Boundary, Species, Reactions, and Other. The Materials section contains the thermal properties of each bulk material in the simulation. The details of the spatial discretization and material arrangement are defined in the Domain Table, and the simulation time and temporal discretization are defined in Time. The Species and Reactions sections define the set of reactions used to model thermal runaway and the species involved in each reaction as well as the species initial conditions. Boundary conditions on the external surfaces of the system are defined in the Boundary section. The Other section contains parameters that do not readily fit in the other sections. A selection of example input files can be found in Appendix A.

3.1. Materials

Material properties are defined in the Materials section. At least one material must be defined, and an arbitrary number of additional materials may be defined. An example material section with two materials is as follows

```
Materials:
    Material A:
        k: 0.5      # W/m/K
        rho: 1800   # kg/m3
        cp: 800     # J/kg/K
    Material B:
        k: 100      # W/m/K
        rho: 900    # kg/m3
        cp: 1800    # J/kg/K
```

This example contains two materials with arbitrary names, Material A and Material B. These names will be used when building the stack of materials in the 1-D domain. Each material must have the following parameters defined:

- k: thermal conductivity in W/m/K
- rho: density in kg/m^3
- cp: specific heat in J/kg/K

3.2. Domain Table

The domain in the direction of discretization is defined using the Domain Table section of the input deck. A sample three layer domain is shown below

```
Domain Table:
    Material Name: [Material A, Material B, Material A]
    Thickness: [0.01, 0.002, 0.01]  # m
    dx: [0.005, 0.001, 0.005]       # m
```

Heat transfer through a 1-D stack of materials is assumed in LIM1TR, where some of the material layers may include heat sources defined by reactions. Three ordered lists are required to describe the system. The Material Name list contains the order of materials in the stack. Materials may be repeated and each material used in the list must be defined in the Materials section of the input file. The Thickness list is the thickness of each material in meters. The dx list is the spatial discretization in the x-direction for each material in meters. These values must be less than or equal to the respective material thickness values. In the event that the thickness is not evenly

divisible by the associated dx, the number of control volumes will be calculated by rounding the thickness divided by dx to the nearest integer. A new dx will be calculated by dividing the thickness by this integer.

An optional thermal contact resistance in units of m^2K/W may be specified between layers of the stack. This is done through a list of the following form

```
Contact Resistance: [0.001, 0.001]  # m2/K/W
```

where the number of entries in the list is the number of interfaces in the stack (i.e. the number of layers minus one). The thermal contact resistance represents imperfect contact between layers of the stack, i.e. thermal contact between adjacent cells, cells and structural materials, etc.

For some simulations, such as a model of a calorimetry experiment, the user may desire a single control volume. This can be accomplished by having a single item in each list and setting dx equal to the thickness. Ex:

```
Domain Table:
    Material Name: [Material A]
    Thickness: [0.01]   # m
    dx: [0.01]          # m
```

3.3. Time

The Time section contains inputs related to the temporal discretization, initial conditions, and outputs. There are three required parameters: Run Time is the total simulation time in seconds, dt is the temporal discretization in seconds, and T Initial is the initial temperature of the materials in Kelvin. A sample block with all possible parameters is as follows

```
Time:
    Run Time: 300.0         # (s) Required
    dt: 0.1                 # (s) Required
    T Initial: 298.15       # (K) Required
    Order: 1                # Optional, 1 or 2 default 1
    Output Frequency: 1     # Optional, default 1
    Print Progress: 1       # Optional, 0 or 1, default 1 for true
    Max Steps: 1e7          # Optional, default 1e7
```

The initial temperature may be specified in one of two ways: a single value or a list. A single value will be applied to the entire domain, whereas a list must have the same number of entries as material layers in the domain table. Each temperature in the list will be applied to the corresponding material in the domain table.

The order of the time integration may be specified with the Order keyword, where the default is 1 for first-order integration. Second order integration using a Crank-Nicolson scheme is specified with 2. More details on the time integration can be found in Section 5.3.

The Output Frequency keyword sets how often results are written to disk. The default value is 1 for saving every time step. To save disk space the user may wish to set a higher integer, x, to save data every x number of time steps.

If the user wishes to suppress output of the simulation progress to the screen, Print Progress may be set to 0. The Max Steps keyword is the maximum number of time steps, and it is set to a default value of 1e7 to prevent the simulation from running indefinitely in the event of an error. This value will become important when adaptive time stepping is implemented in a future release of LIM1TR.

3.4. Boundary

Boundary conditions must be specified on three surfaces referred to as the left face (where $x = 0$), right face (where $x = L$, the total length of the domain), and the external faces (the remaining four faces). Figure 3-1 shows the boundary faces for an example domain with a single cell.

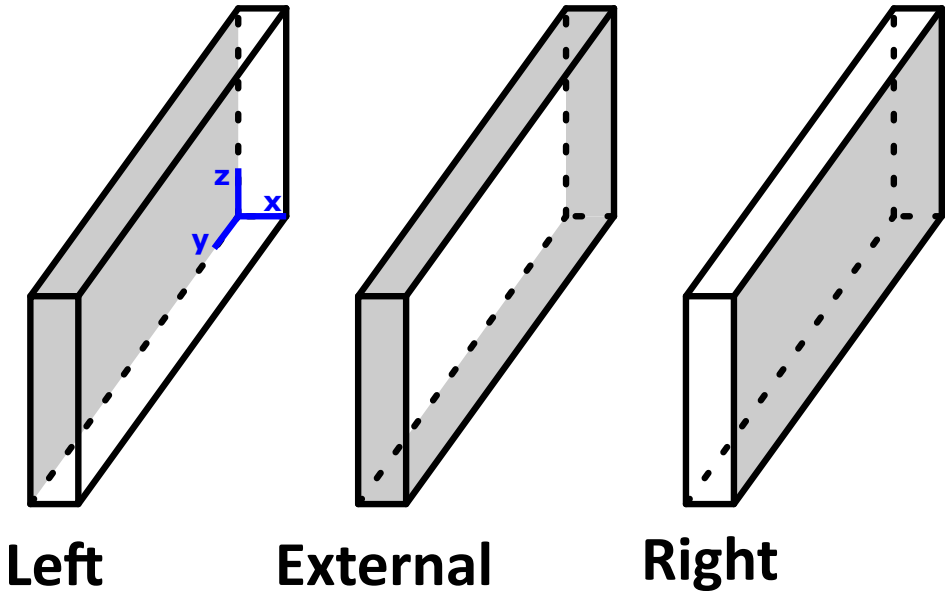

Figure 3-1. Boundary faces highlighted on a single cell.

An example Boundary section may look like the following

```
Boundary:
    Left:
        Type: Heat Flux
        Flux: 150.0         # W/m2
```

```
    Right:
        Type: Adiabatic
    External:
        Type: Convection
        h: 10.0             # W/m2/K
        T: 300.0            # K
```

Three boundary conditions are currently available: Adiabatic, Heat Flux, and Convection. The boundary condition is specified by the Type keyword. The Adiabatic condition has no additional required parameters and may be applied to any of the boundaries.

The Heat Flux boundary condition will apply a constant heat flux to the specified surface. This is set with the Flux keyword in units of W/m^2. This boundary condition may only be applied to the left or right boundaries.

The Convection boundary condition applies a convective flux and has two required parameters: the heat transfer coefficient (keyword h) in units of $W/m^2/K$, and the external temperature (keyword T) in units of Kelvin. This boundary condition may be applied to any of the boundaries.

Each boundary condition can contain an optional deactivation time specified as `Deactivation Time: 10`, where the units of time are in seconds. This will turn the boundary condition off (becoming an adiabatic boundary) after the specified time, and may be used for scenarios where a heater is used to trigger thermal runaway and is subsequently turned off.

We note that currently only one boundary condition may be applied per side. Future releases will target multiple boundary condition specification as well as radiative heat transfer.

3.5. Species

The Species section describes the species that are tracked by the reaction solver within the battery material. All species are assumed to have the bulk thermal properties of the battery material. There are four required inputs in the Species section: Names, Initial Mass Fraction, Molecular Weights, and Material Name. The following Species section shows an example of a three species material.

```
Species:
    Names: ['Species A', 'Species B', 'Inert Species']
    Initial Mass Fraction: [0.4, 0.0, 0.6]  # kg species/kg total
    Molecular Weights: [72.0, 90.0, 0.0]    # kg/kmol
    Material Name: Battery
```

In this example, three species are tracked on a material named Battery. This name must match a material name in the Materials section of the input file. Currently, only one reacting material is allowed in LIM1TR.

The lists in Names, Initial Mass Fraction, and Molecular Weights must all be the same length and may be specified in either list format. The Names key lists the arbitrary names of the species that will be used to construct the reactions. Initial Mass Fraction is an ordered list of the initial mass fractions of each species that must sum to one, otherwise an error will be thrown. In this case only Species A and the Inert Species are present in the beginning of the simulation. It is common to have an inert species that lumps together all non-reacting mass in the battery. The final list, Molecular Weights, corresponds to the molecular weights of each species in kg/kmol. Note that the molecular weight for the inert species may be specified as zero (or any arbitrary value), as it does not participate in any reactions.

3.6. Reactions

The Reaction section contains an arbitrary number of reaction descriptions keyed by increasing integers. The base reaction has an Arrhenius form of the rate constant

$$k = A \cdot exp\left(\frac{-E_a}{RT}\right) \tag{3.1}$$

For a detailed description of the reaction equations, see Section 5.

3.6.1. Base Reaction

A basic Arrhenius reaction is constructed in the input file as follows

```
Reactions:
    1:
        A: 1e10             # see discussion
        E: 1e5              # J/mol
        R: 8.314            # J/mol/K
        H: -1e6             # J/kg reactants
        Reactants:
            'Species A': 1  # kmol
        Products:
            'Species B': 1  # kmol
        Orders:
            'Species A': 2
```

The required inputs are the pre-exponential factor (A), activation energy (E), ideal gas constant (R), heat of reaction (H), stoichiometric coefficients of the reactants and products, and the reaction orders. Any number of reactants, products, or orders may be specified through key-value pairs where the key must match a species name listed in the Species section. The units of the reactants and products values are kmol.

The units of the four required constants are

- A: varies with reaction orders; should produce units of kg of reactants/m^3/s when multiplied with the concentration function
- E: J/mol
- R: J/mol/K
- H: J/kg of reactants

The specification of E and R are flexible as long as the units of E/R comes out in Kelvin.

Special consideration is given to the pre-exponential factor as it is typically given in literature as a frequency (1/s). As LIM1TR is a mass-based code, the pre-exponential factor must be specified such that the pre-exponential factor multiplied with the concentration produces a mass concentration rate in units of kg of reactants/m^3/s. Unit conversions to a mass concentration rate become more complicated when the overall reaction order deviates from unity; in such cases the pre-exponential factor must be updated when the cell build changes. Details of these conversions can be found in Section 5.4.

3.6.2. Custom Reactions

In addition to the basic Arrhenius form, custom reactions can be included in LIM1TR through user specified reaction models written in Python. Instructions for creating reaction models are given Section 6.1. Two custom models are currently included in LIM1TR: a critical thickness anode model and a short circuit model.

3.6.2.1. Critical Thickness Anode Model

The critical thickness anode model is based on the work of Shurtz et al. [8,9]. This reaction model is specified by `Type: 'Zcrit'`. An example from Shurtz et al. is as follows

```
A: 3.2718e+13       # see discussion
E: 16236.69493      # J/mol
R: 1                # J/mol/K
H: -2287100.0       # J/kg reactants
Reactants:
    'C6Li': 2       # kmol
    'EC': 1         # kmol
```

```
Products:
    'C6': 2          # kmol
    'Li2CO3': 1      # kmol
    'Gas': 28.054    # kmol
Type: 'Zcrit'
BET_C6: 1.1          # m2/g
tau_crit: 0.08       # unitless
C_t: 72.5            # unitless
Y_Graphite: 0.12     # kg graphite/kg total
```

The species C6Li and EC (ethylene carbonate) must be included as reactants, and the species C6 and Li2CO3 must be included as products. The lumped gas species may be divided into discrete products. We note that LIM1TR does not currently contain a venting model, and any gaseous products are assumed to remain where they are produced.

Four additional model parameters must be specified.

- BET_C6: the BET surface area of graphite in m^2/g
- tau_crit: critical tunneling barrier (z_{crit} from Shurtz et al.) divided by C_t
- C_t: fitting parameter for growth of the tunneling barrier
- Y_Graphite: mass fraction of graphite in the cell

3.6.2.2. Simple Short Circuit Model

A simplified short circuit model is included in LIM1TR for scenarios where an internal electrical connection is formed between the electrodes of a single cell (e.g. internal short, nail penetration tests, etc.). The short is a zero-order reaction that consumes active material in the anode and cathode, and it depends on the cell voltage, short resistance, and cell volume.

```
Reactants:
    'R1': 1              # kmol
    'R2': 1              # kmol
Products:
    'P1': 1              # kmol
    'P2': 1              # kmol
Type: 'Short'
Voltage: 4.2             # V
Short Resistance: 0.001 # Ohms
Volume: 3.36e-5          # m3
```

In this example, the reactants (R1 and R2) are the active materials in the anode and cathode that are converted into user-defined products (P1 and P2). The units of voltage are in volts, short resistance is in Ohms, and volume is in cubic meters.

3.6.3. Reaction Sub-Models

Two reaction sub-models are currently available in LIM1TR: an electrolyte limiter and a Damköhler number limiter. The syntax for specifying a sub-model on a reaction (number 1 in this case) is as follows:

```
1:
    A: ...
    ...
    Sub-model Name:
        Sub-model parameter 1: ...
        Sub-model parameter 2: ...
        ...
```

3.6.3.1. Electrolyte Limiter Sub-Model

The electrolyte limiter sub-model provides a transition from zero-order dependence to a first-order dependence of the reaction rate on electrolyte concentration as detailed by Shurtz et al. [8]. The form of the electrolyte limiter is

$$r = k\frac{\rho_{electrolyte}}{\rho_{electrolyte} + \rho_{elec,lim}} \tag{3.2}$$

for an arbitrary reaction rate r and rate constant k; the terms in this form have been rearranged for greater simplicity with respect to the previously published version. The limiter is applied multiplicatively to the rate constant, and it is a function of the electrolyte mass concentration, $\rho_{electrolyte}$, and user specified limiting constant, $\rho_{elec,lim}$. Figure 3-2 illustrates the concept of the electrolyte limiter for a characteristic electrode particle. On the left, a reservoir of electrolyte is available for electrode-electrolyte reactions, i.e. when the electrolyte concentration in Equation 3.2 is high. When the electrolyte is depleted, as shown in Figure 3-2 right, the electrolyte limiter reduces the reaction rate by transitioning to first order dependence on the electrolyte concentration.

This sub-model requires an electrolyte species name and limiting constant for the transition to first-order dependence (specified in kg/m^3).

```
Electrolyte Limiter:
    Species: 'EC'
    Limiting Constant: 1.2  # kg/m3
```

The limiting constant can be calculated as

$$\rho_{elec,lim} = a_{BET}\rho_{graphite}m_{50} \tag{3.3}$$

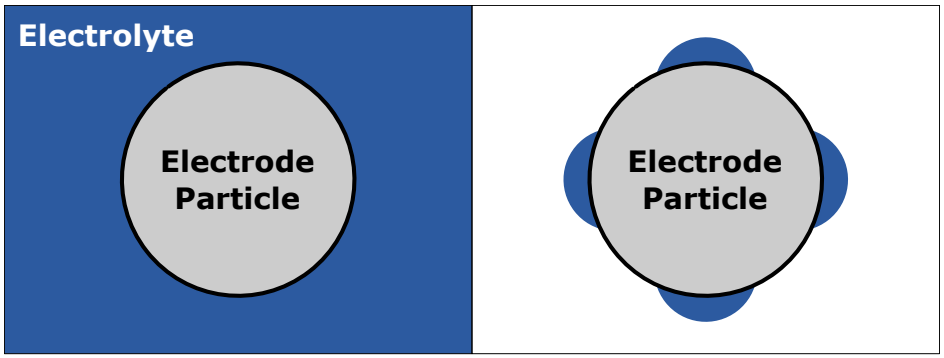

Figure 3-2. Electrolyte reactions with an electrode particle surrounded by plentiful electrolyte (left) are not constrained by the electrolyte concentration, whereas a particle in a depleted reservoir of electrolyte (right) has a limited amount of liquid reactants available.

where a_{BET} is the BET surface area of the graphite anode, $\rho_{graphite}$ is the mass concentration of graphite in the anode, and m_{50} is the mass per surface area order transition for the electrolyte. Shurtz et al. recommends a value of 0.005 g/m^2 for m_{50}, which corresponds to the mass of electrolyte per surface area at which Equation 3.2 reduces the rate by 50% [8].

3.6.3.2. Damköhler Limiter Sub-Model

The Damköhler limiter sub-model seeks to capture the effect of intra-particle diffusion on the reaction rate as described by Kurzawski et al. [4]. Figure 3-3 depicts a simplified representation of lithium diffusion in a spherical anode particle. The inner core where the radius is less than r_i contains reactive material (lithiated graphite in this example). The lithium must diffuse through a spherical shell of non-reactive material to the surface of the particle at r_o to react with electrolyte.

The rate constant (k) is modified to

$$\hat{k} = \frac{k}{1 + Da} \tag{3.4}$$

by matching rates of serial processes at the surface of the characteristic particle. The Damköhler number (Da) is a ratio of chemical reaction rate to the intra-particle diffusion rate. It is defined as

$$Da = \frac{k}{a_e \cdot \rho \cdot D \cdot exp(-E_D/RT)} \frac{r_o(r_o - r_i)}{r_i} \tag{3.5}$$

where r_i and r_o are the inner and outer diameter, respectively, of the characteristic particle in meters. The remaining model parameters are the specific edge area (a_e), material density (ρ), diffusion pre-exponential factor (D), and diffusion activation energy (E_D). With the exception of

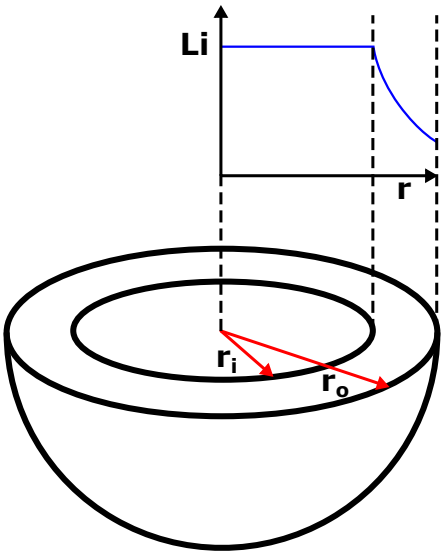

Figure 3-3. Spherical anode particle depicting lithium diffusion from the core of the particle through a spherical shell of non-reactive material to the surface.

a_e, the Damköhler limiter parameters are specified in a sub-model block within the target reaction as follows.

```
Damkohler:
    D: 2.0e-14      # m2/s
    E: 29000.0      # J/mol
    A: 6.667e+11    # 1/s
    r_i: 1.0e-6     # m
    r_o: 2.0e-6     # m
```

Note that the un-scaled pre-exponential factor from the rate constant (A in units of 1/s) is also required for the Damköhler limiter. See Section 5.4 for details of the units of this parameter. The user must also ensure that the units for the activation energy are the same between the Damköhler limiter and the parent reaction.

The final required parameter for the Damköhler limiter is the specific edge area (a_e) of the electrode particles in m^2/kg. The specific edge area is the relevant surface for diffusion of lithium in and out of graphite, and a method to estimate a_e from measurements of graphite BET surface area is available in Shurtz et al. [9]. When the Damköhler model is applied to other materials for which a specific type of diffusive surface area has not been identified, the total BET surface area is a reasonable estimate. This parameter is specified under the target reaction's numbered section with the keyword `a_edges`. This keyword is not needed for the critical thickness anode model as the value of the specific edge area is calculated internally. An example of the input syntax can be found in the repository under Examples/Damkohler.

3.6.4. Systems of Reactions

Reactions may be specified to only occur on specific cells in the stack. This can be useful in the scenarios where one cell is triggered by an internal short or where there are different cell chemistries present. This is done with the optional keyword `Active Cells: []` that can be included within each individual reaction. By default, all cells of the material specified in the Species section will have the reaction. Each reaction can have its own set of active cells.

A list of cells can be supplied to Active Cells with cell numbers corresponding to the Material Name list in the Domain Table section. For example, the following system is specified:

```
Domain Table:
    Material Name: [End Block, Battery, Spacer, Battery, End Block]
    ...
```

and reaction 1 only occurs on the second battery in the list of layers (counting from left to right). The Reactions section for material Battery will include

```
Reactions:
    1:
        ...
        Active Cells: [2]
    2:
        ...
```

If reaction 2 in this system occurs in all cells, the Active Cells list does not need to be specified, and reaction 2 will be applied to all cells by default.

3.7. Other

The Other section encapsulates two required parameters and a few optional parameters.

```
Other:
    Y Dimension: 0.05   # m
    Z Dimension: 0.07   # m
    Reaction Only: 1    # Optional, default 0
    DSC Mode: 1         # Optional, default 0
    DSC Rate: 0.1       # (K/S) Optional
```

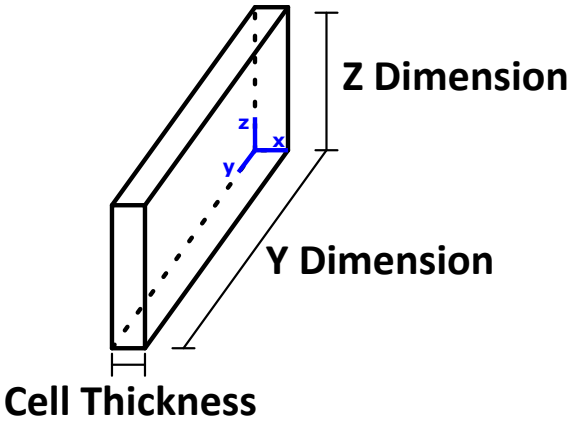

Figure 3-4. A single cell domain as represented in LIM1TR with the three dimensions labeled: the x-direction thickness as specified in the Domain Table, the Y Dimension of the cell (length), and the Z Dimension of the cell (height).

The Y Dimension and Z Dimension parameters represent the dimensions of the system in the plane perpendicular to the direction of discretization. These dimensions are required. An example cell with the dimensions labeled is shown in Figure 3-4.

The Reaction Only parameter is a boolean (1 or 0 for true or false respectively) that will deactivate the conduction solve and boundary conditions if it is set to true. This parameter is used for simulating calorimetry such as ARC (accelerating rate calorimetry) or DSC (differential scanning calorimetry) on a small sample modeled by a single control volume where the conduction solve is unnecessary. For an ARC simulation, it is recommended to have a single control volume the size of the sample with Reaction Only set as true and the initial temperature set to the onset of thermal runaway.

The DSC Mode parameter can be used to simulate a DSC experiment where the heating rate is fixed. If DSC Mode is set to true, a DSC rate must be specified in K/s. Similar to an ARC simulation, it is recommended to have a single control volume the size of the sample.

This page intentionally left blank.

4. OUTPUT

Output from LIM1TR is saved using the Python pickle file format. The name of the output file is "(input file name)_output.p" where the input file name is taken from the .yaml file. For example, "LCO_cell.yaml" will have an output file named "LCO_cell_output.p". Prior to creation of the final output file, checkpoint files are saved periodically. These files will have the naming structure "(input file name)_output_0000.p" where the four digit integer will increment by one with successive checkpoints. The number of time steps in each checkpoint file is calculated based on the domain size and number of species so that the file size of each checkpoint is approximately 5 MB. This serves to reduce the memory footprint during execution, limit I/O, and save simulation data should an error occur resulting in termination of the simulation. When the simulation time reaches the total run time, all checkpoint files will be combined to create the "_output.p" file.

The pickle format has the benefit of storing Python objects and data structures. Each output file contains a list of three Python dictionaries in the following order:

- The .yaml input file parsed into a dictionary
- The state variables at each control volume and simulation time
- The rate of change of state variables with respect to time at each control volume and simulation time

Parameters from the input file used to generate the output can be accessed through the same keys used in the input file where indentations define the keys of the parent dictionary. For example, the density of a material named Cell can be accessed with `input_dictionary['Materials']['Cell']['rho']`.

The two remaining dictionaries contain the simulation output at all nodes and selected time steps, and both dictionaries have a similar format. The state variable dictionary has, at minimum, the following keys and corresponding numpy arrays:

- "Time": simulation time (1-D array)
- "Grid": location of the center of each control volume (1-D array)
- "Temperature": temperature (2-D array)
- "Interface Temperature": temperature at each material interface (2-D array)

State variables are saved as 2-D arrays where the nodes correspond to the columns of the array and the times correspond to the rows. The Time array contains the simulation time in seconds at each of the output time steps specified by Output Frequency in the input file. The default frequency is 1 for saving every time step. For long duration simulations or a large number of control volumes, an output frequency greater than 1 may be desired to reduce the file size of the simulation results. The Grid array contains the node locations in meters in the x-direction centered in each control volume where the index of the node in the Grid array corresponds to the index of the column in the state variable arrays.

The temperature is solved in all simulations and stored in the Temperature array for all nodes and selected time steps in units of Kelvin. Experimental temperatures are typically collected at the interfaces between cells or other materials in the system. For convenience, the Interface Temperature array provides interface temperatures between material layers that are estimated as the average temperature of the control volumes on each side of the interface. The rows of this array correspond to the Time array while the columns correspond to the interfaces ordered by increasing x location.

For simulations where reactions are present, a 2-D state variable array is saved for the mass concentration of each species in kg/m^3. The dictionary keys for these arrays are the Names from the Species section of the input file. Indexing of the 2-D array follows the same format as the Temperature array.

The final dictionary contains the time rate-of-change ($1/s$) of each of the state variables due to chemical reactions (temperature and species). The indexing of the 2-D arrays follows the same pattern as the state variable dictionary described previously. This dictionary does not contain the grid or interface temperatures. At minimum, this dictionary has the following keys:

- “Time”: simulation time (1-D array)
- “Chemical Temperature Rate”: net temperature rate for all chemical reactions (2-D array)
- “HRR”: volumetric heat release rate from chemical reactions (2-D array)

The keys for the species rates are the same as the state variable dictionary, whereas the net temperature rate of change due to all chemical reactions (without thermal diffusion) is accessed with the “Chemical Temperature Rate” key. The rate dictionary contains one additional output array for the heat release rate (“HRR”). This is the volumetric heat release in W/m^3 at each node and time where positive values are heat generated and negative values are heat absorbed by the reactions. This corresponds to the heat source term at each control volume.

5. THEORY

This chapter outlines the conservation equations for the 1-D thermal runaway problem and the numerics used to solve these equations. LIM1TR solves the transient heat equation with reaction source terms over a domain that is discretized in one direction and lumped in the other two spatial directions. The domain is discretized using the finite volume method, and first order and second order time integrators are available. In reacting materials, the species are assumed to be homogeneously mixed within each control volume, and mass transport is neglected. The heat transport and reaction rates are solved separately using an operator splitting method.

5.1. Conservation Equations

Energy conservation is solved with the transient heat equation for temperature (T). The differential form is given as

$$\rho c_p \frac{\partial T}{\partial t} = \nabla \cdot (k \nabla T) + \dot{q}''' \tag{5.1}$$

where ρ is the density in kg/m^3, c_p is the specific heat in J/kg/K, k is the thermal conductivity in W/m/K, and $\dot{q}'''$ is the volumetric heat source (in W/m^3) which is a sum of the contributions from each reaction.

$$\dot{q}''' = \sum_{j=1}^{N_r} \Delta H_j r_j \tag{5.2}$$

The volumetric heat source term depends on the reaction rates (r_j in kg of reactants/m^3/s) and heat of reaction (ΔH_j in J/kg of reactants) for each of the N_r reactions. The general form of a reaction rate (from Eqn. 3.1) is

$$r_j = A_j \cdot exp\left(\frac{-E_j}{R_j T}\right) \prod_{i=1}^{N_s} \rho_i^{n_{ij}} \tag{5.3}$$

where A_j, E_j, and R_j are the pre-exponential factor, activation energy, and ideal gas constant for reaction j. The concentration function for reaction j is a function of the species concentrations (ρ_i) and their respective orders (n_{ij}) for the number of species listed in the input file (N_s).

Conservation of species i is given by

$$\frac{\partial \rho_i}{\partial t} = \sum_{j=1}^{N_r} (\nu''_{ij} - \nu'_{ij}) r_j \tag{5.4}$$

where the mass-based stoichiometric coefficients of species i consumed and produced in reaction j are ν'_{ij} and ν''_{ij} respectively. Conversion from the standard mole-based stoichiometric coefficients ($\hat{\nu}$) specified in the input file is done through

$$\nu_i = \frac{W_i \hat{\nu}_i}{\sum_k W_k \hat{\nu}_k} \tag{5.5}$$

where W_i is the molecular weight of species i in kg/kmol, and k is the set of all products or all reactants in the reaction, but not both.

5.2. Spatial Discretization

The system is discretized with a finite volume method using cell-centered nodes. Spatial discretization occurs only in the x-direction (see Fig. 5-1), and the y and z directions are assumed to be lumped. This assumption generally holds as the thermal conductivity in the plane of the electrodes (the y-z plane) is much higher than the thermal conductivity normal to the plane of the electrodes (x-direction) due to the layered structure of the cell with electrodes, binders, current collectors, separators and their associated thermal contact resistances.

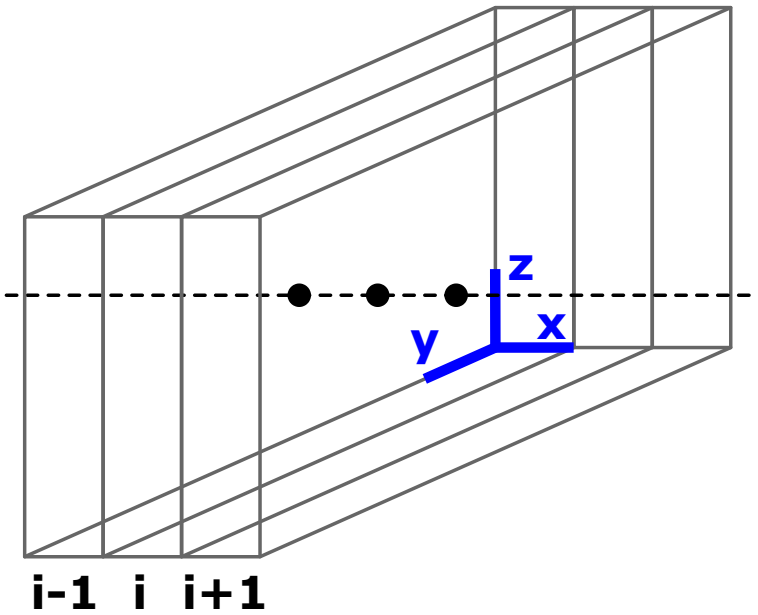

Figure 5-1. Example of three control volumes for nodes $i-1$, i, and $i+1$ depicting discretization in the x-direction and lumping in the y and z-directions.

For larger format cells, it is recommended that the Biot number be checked prior to constructing the simulation to determine if the lumped assumption holds in the y-z plane. Typically, a Biot number of less than 0.1 is sufficient for applying the lumped assumption [2]. The Biot number is given as

$$Bi = \frac{hL_c}{k_{in}} \tag{5.6}$$

where h is the external heat transfer coefficient in W/m^2/K, k_{in} is the in-plane thermal conductivity, and L_c is a characteristic length scale of the heat transfer in the y and z directions in m. This length scale is taken as the ratio of the y-z surface area to the external perimeter of the cell.

$$L_c = \frac{\Delta y \Delta z}{2(\Delta y + \Delta z)} \tag{5.7}$$

As an example, a 3 Ah lithium cobalt oxide pouch cell may have positive electrodes composed of lithium cobalt oxide particles on aluminum foil collector plates with dimensions of 4 cm by 12 cm. The negative electrodes are typically graphite particles on copper foil collector plates with the same dimensions. Ambient heat transfer coefficients are on the order of 10 W/m^2/K, and the composite in-plane thermal conductivity of the cell is 27 W/m/K [3]. These parameters result in a Biot number of about 0.006 which meets the criteria for the lumped assumption.

We begin the derivation of the finite volume formulation with the differential form of the heat equation with a volumetric source term (Eq. 5.1). Integrating over a volume (V) of dimensions Δx by Δy by Δz for a generic volume gives

$$\int_V \rho c_p \frac{\partial T}{\partial t} dV = \int_V \left[\nabla \cdot (k \nabla T) + \dot{q}'''\right] dV \tag{5.8}$$

Applying the divergence theorem

$$\int_V \rho c_p \frac{\partial T}{\partial t} dV = \int_S \left(k \frac{\partial T}{\partial x} \cdot \vec{n}\right) dS + \int_E (F_E \cdot \vec{n})\, dE + \int_V \dot{q}''' dV \tag{5.9}$$

The surface of the control volume is split into two parts: the internal faces perpendicular to the x-direction (S) and the faces around the external perimeter of the system (E). Conduction heat transfer is solved in the x-direction and boundary conditions are applied to surface E.

The volume integral is approximated by the value at the volume-centered node. The domain is comprised of N nodes indexed from 0 to $N-1$. For an arbitrary node i the volume integrals become

$$\int_V \rho c_p \frac{\partial T}{\partial t} dV = \Delta V_i \rho c_p \frac{\partial T_i}{\partial t} \tag{5.10}$$

and

$$\int_V \dot{q}'''dV = \Delta V_i \dot{q}_i''' \tag{5.11}$$

where the volume of control volume i is

$$\Delta V_i = \Delta x_i \Delta y \Delta z \tag{5.12}$$

For conduction between adjacent control volumes, the flux through surface S of control volume i is discretized as

$$\int_S \left(k \frac{\partial T}{\partial x} \cdot \vec{n} \right) dS = \int_S \left(k_{i-1} \frac{T_i - T_{i-1}}{x_i - x_{i-1}} \right) dS + \int_S \left(k_i \frac{T_{i+1} - T_i}{x_{i+1} - x_i} \right) dS \tag{5.13}$$

where the thermal conductivity is evaluated at the interfaces between control volumes. When the interface is located between two control volumes of the same material, the thermal conductivity is evaluated as the material conductivity. When the interface occurs between two materials, the thermal conductivity is evaluated as

$$k_i = (x_{i+1} - x_i) \left(\frac{\Delta x_i}{2k_i} + R'' + \frac{\Delta x_{i+1}}{2k_{i+1}} \right)^{-1} \tag{5.14}$$

where an optional thermal contact resistance (R'') may be specified.

At the left and right extents of the domain, nodes 0 and $N-1$ respectively, the three available boundary conditions are applied as follows. For the adiabatic case, the surface integral is simply set to zero. For a prescribed heat flux (q''), the surface integral is

$$\int_S q'' dS \tag{5.15}$$

The convective boundary condition is

$$\int_S \left[\frac{h_{left} \frac{2k_0}{\Delta x_0}}{h_{left} + \frac{2k_0}{\Delta x_0}} (T_{\infty,left} - T_0) \right] dS \tag{5.16}$$

on the left-most control volume, and

$$\int_S \left[\frac{h_{right} \frac{2k_{N-1}}{\Delta x_{N-1}}}{h_{right} + \frac{2k_{N-1}}{\Delta x_{N-1}}} (T_{\infty,right} - T_{N-1}) \right] dS \tag{5.17}$$

on the right-most control volume, where the left and right heat transfer coefficients are h_{left} and h_{right} respectively.

For all surfaces perpendicular to the x-direction, the surface integral is evaluated as the cross-sectional area of the stack.

$$\int_S dS = \Delta y \Delta z \tag{5.18}$$

Currently, two boundary conditions are available for the remaining surfaces around the perimeter of the stack: adiabatic and convective. For the adiabatic case, the flux (F_E) is zero, while the convective flux is

$$F_{E,i} = h_{\infty,E}(T_{\infty,E} - T_i) \tag{5.19}$$

for external heat transfer coefficient ($h_{\infty,E}$) and temperature ($T_{\infty,E}$).

The external surface integral is evaluated as

$$\int_E (F_{E,i} \cdot \vec{n})\, dE = 2\Delta x_i(\Delta y + \Delta z) F_{E,i} \tag{5.20}$$

5.3. Time Integration

Time integration is carried out by an operator splitting method described by Strang [10]. For the system

$$\frac{du}{dt} = Du + Ru \tag{5.21}$$

where u is a vector of temperature and species, D is the diffusion operatior, and R is the reaction operator, Strang splitting for a time step Δt is described as

$$\begin{aligned}
\frac{du^*}{dt} &= Du^*, \quad u^*(t_n) = u^n \quad \text{on} \quad \left[t_n, t_n + \frac{\Delta t}{2}\right] \\
\frac{du^{**}}{dt} &= Ru^{**}, \quad u^{**}(t_n) = u^*\left(t_n + \frac{\Delta t}{2}\right) \quad \text{on} \quad [t_n, t_n + \Delta t] \\
\frac{du^{n+1}}{dt} &= Du^{n+1}, \quad u^{n+1}\left(t_n + \frac{\Delta t}{2}\right) = u^{**}(t_n + \Delta t) \quad \text{on} \quad \left[t_n + \frac{\Delta t}{2}, t_n + \Delta t\right]
\end{aligned} \tag{5.22}$$

The first and third steps involve applying the diffusion operator to the temperature. For these steps, time is advanced with either a first-order backward Euler implicit scheme or the

second-order Crank-Nicolson scheme [7]. This discretized system of equations is solved with the Thomas (tridiagonal) algorithm [7].

The differential equation in the second step is a function of both temperature and species concentrations, and after spatial discretization it manifests as a set of non-linear ordinary differential equations (ODE) at each control volume. For the temperature, the ODE is

$$\rho c_p \frac{\partial T}{\partial t} = \sum_{j=1}^{N_r} \Delta H_j r_j \tag{5.23}$$

and for species the ODEs are given by Equation 5.4. Time integration of this step is carried out by the LSODA solver as implemented in SciPy [12] which requires evaluating the ODE and Jacobian over time steps that are typically much smaller than Δt. We note that positive species concentrations are enforced in this step.

5.4. Reaction Parameters

Chemical reaction rates in LIM1TR (Eq. 5.3) have units corresponding to mass of reactants consumed in their stoichiometric proportions per volume per unit time, or kg/m^3/s. Since the Arrhenius term in the rate constant (Eq. 3.1) is dimensionless, the units of the pre-exponential factor A must compensate for the concentration function in Equation 5.3 to yield the proper units for the rate. LIM1TR allows the user to specify reaction orders n_{ij} for any species in the material, independent of the reaction stoichiometry. Reaction orders not specified default to zero.

A wide variety of reaction orders can lead to very different units for the pre-exponential factors, and the user should carefully check the unit conversions from frequency units (s^{-1}). Frequency units are preferred for reporting and communications because not all codes modeling reactions use mass-based concentration units. Mass-based units will generally lump ratios of molecular weights and other parameters with the pre-exponential factor used in LIM1TR. The user should be mindful that some cases arise where nominally identical materials occurring in different cell builds (with different mass concentrations of active materials) require different inputs for the pre-exponential factor to obtain rates that are consistent with the same parameter in the original frequency units. This generally occurs when the sum of the reaction orders is not unity. The first example shown below has a non-unity overall reaction order that requires the pre-exponential factor to be adjusted in LIM1TR for different cell builds. The second example has reaction orders that cause the pre-exponential factor required by LIM1TR to be independent of reactant concentrations because the overall reaction order is unity.

5.4.1. Example of Pre-Exponential Factor with Non-Unity Reaction Order

If a reaction is first order in two species, the sum of the reaction orders is $\Sigma n_i = 2$, which is considered in this example. Similar derivation procedures must be followed for other cases where the overall reaction order is not unity, including fractional orders. In this case two reactants are

considered to contribute to the overall order, but a similar derivation holds for a reactant and a product that both contribute order 1 to the overall order (which can occur for an autocatalytic reaction) [1,4,6]. Unique pre-exponential factors are required in LIM1TR for different cell builds when the overall reaction order is not unity.

5.4.1.1. Reaction Definition

We assume that within a cathode particle, the lattice size or molar density is constant. Li_xMO_2 can be represented as x moles of $LiMO_2$ mixed with (1-x) moles of delithiated MO_2. The example reaction

$$MO_2 + LiMO_2 \rightarrow LiM_2O_4 \tag{5.24}$$

is a structural rearrangement that may be assumed first order in each reactant. We also assume there is no LiM_2O_4 in the initial lattice. For this homogenized approach we may assume that either the reactants are perfectly mixed or that transport of lithium atoms within the lattice is sufficiently fast at the lower temperatures for chemical kinetics to be limiting. For cases where mass transport of lithium in electrode particles is limiting, see the Damköhler limiter sub-model in Section 3.6.3.2.

5.4.1.2. Dimensionless Cathode Decomposition Rate

The consumption rate of a species can be written in terms of fractional conversion of the reactants

$$-r_{MO_2} = -\frac{dX_{MO_2}}{dt} = Aexp\left(\frac{-E}{RT}\right) X_{MO_2} X_{LiMO_2} \tag{5.25}$$

where X is a dimensionless fraction of the moles of lithiated $LiMO_2$ or delithiated MO_2 units with respect to the total moles of available metal atom sites n_M

$$\begin{aligned} X_{MO_2} &= \frac{n_{MO_2}}{n_M}, \\ X_{LiMO_2} &= \frac{n_{LiMO_2}}{n_M} \end{aligned} \tag{5.26}$$

The rate r_{MO_2} has units of (moles MO_2/moles M/time) = s^{-1}. The pre-exponential factor A therefore has frequency units of (moles M/moles $LiMO_2$/time) = s^{-1}.

Note that the reaction rate can also be written in terms of the consumption of $LiMO_2$ or the production of LiM_2O_4. The following expressions allow conversion of reaction rates between different species

$$X_{LiM_2O_4} = \frac{n_{LiM_2O_4}}{n_M}$$
$$-\frac{X_{MO_2}}{dt} = -\frac{X_{LiMO_2}}{dt} = \frac{X_{LiM_2O_4}}{dt} \tag{5.27}$$

Note that the maximum value of $X_{LiM_2O_4}$ at full conversion is 0.5 rather than 1.0 as defined here.

5.4.1.3. Molar Concentration Form of Rate Expression

The dimensionless form can be converted to molar concentration units c by applying the definitions of X (n_M is constant) and dividing by the computational control volume ΔV

$$-\frac{dc_{MO_2}}{dt} = -\frac{n_M}{\Delta V}\frac{dX_{MO_2}}{dt} = Aexp\left(\frac{-E}{RT}\right)c_M\frac{c_{MO_2}c_{LiMO_2}}{c_M c_M} \tag{5.28}$$

simplifying to

$$-\frac{dc_{MO_2}}{dt} = Aexp\left(\frac{-E}{RT}\right)\frac{c_{MO_2}c_{LiMO_2}}{c_M} \tag{5.29}$$

Thus, the pre-exponential factor may be divided by the constant molar concentration of metal sites c_M when operating in molar units, yielding A' with units of (volume/moles $LiMO_2$/time) = m^3/kmol/s. Note that units of kilomoles are used as LIM1TR operates in MKS units. This compensates for effects of cell build and inert mass on bulk concentrations.

5.4.1.4. Mass Concentration Form of Rate Expression, Including LIM1TR Form

The molar concentration form can be converted to mass concentration units ρ by multiplying the rate by the molecular weight W and expressing individual concentrations as $c = \rho/W$

$$-\frac{d\rho_{MO_2}}{dt} = -W_{MO_2}\frac{dc_{MO_2}}{dt} = Aexp\left(\frac{-E}{RT}\right)\frac{W_M\rho_{MO_2}\rho_{LiMO_2}}{W_{LiMO_2}\rho_M} \tag{5.30}$$

LIM1TR operates on a mass basis wherein chemical rates are expressed as the mass conversion rate of all the reactants in stoichiometric proportions. The rate expression in these units is

$$-\frac{d\rho_R}{dt} = -\left(W_{MO_2} + W_{LiMO_2}\right)\frac{dc_{MO_2}}{dt} = Aexp\left(\frac{-E}{RT}\right)\left(W_{MO_2} + W_{LiMO_2}\right)\frac{W_M\rho_{MO_2}\rho_{LiMO_2}}{W_{MO_2}W_{LiMO_2}\rho_M} \tag{5.31}$$

In this formulation, constant molecular weights and ρ_M are lumped with the pre-exponential factor. For either mass basis, this yields A'' with units of (volume/mass/time) = m^3/kg/s. The appropriate pre-exponential input for LIM1TR is therefore

$$A'' = A\frac{W_M\left(W_{MO_2} + W_{LiMO_2}\right)}{W_{MO_2}W_{LiMO_2}\rho_M} \tag{5.32}$$

and the ρ_M term causes A'' to vary with cell build. Some dependence of A'' on a reference concentration such as ρ_M will occur whenever the overall reaction order deviates from unity.

5.4.2. Example of Pre-Exponential Factor with Unity Reaction Order

If a reaction is first order in one species and zero-order with respect to all other species, the sum of the reaction orders is $\Sigma n_i = 1$, which is considered in this example. Similar derivations result for other cases where the overall reaction order is unity, including reactions that are half-order in two species. Unity overall reaction orders are a special case where the pre-exponential factor required by LIM1TR is independent of concentration.

5.4.2.1. Reaction Definition

We assume a single reactive solid species that decomposes to another solid and a gas according to

$$2MO_2 \rightarrow 2MO + O_2 \tag{5.33}$$

In the final step of this derivation, we also consider the special case where oxidation of the solvent ethylene carbonate (EC = $C_3H_4O_3$) occurs rapidly once oxygen is produced.

$$5MO_2 + EC \rightarrow 5MO + 3CO_2 + 2H_2O \tag{5.34}$$

For this derivation we assume that the representative solvent EC is not limiting with respect to this global reaction.

5.4.2.2. Dimensionless Cathode Decomposition Rate

The reaction rate based on a dimensionless mole fraction is

$$r_{MO_2} = -\frac{dX_{MO_2}}{dt} = Aexp\left(\frac{-E}{RT}\right)X_{MO_2} \tag{5.35}$$

The rate r_{MO_2} has units of (moles MO_2/moles M/time) = s^{-1}. The pre-exponential factor A therefore has frequency units of (moles M/moles MO_2/time) = s^{-1}.

5.4.2.3. Molar Concentration Form of Rate Expression

The dimensionless form can be converted to molar concentration units c by applying the definitions of X (n_M is constant) and dividing by the computational control volume ΔV:

$$-\frac{dc_{MO_2}}{dt} = -\frac{n_M}{\Delta V}\frac{dX_{MO_2}}{dt} = Aexp\left(\frac{-E}{RT}\right)c_M\frac{c_{MO_2}}{c_M} \tag{5.36}$$

simplifying to

$$-\frac{dc_{MO_2}}{dt} = Aexp\left(\frac{-E}{RT}\right)c_{MO_2} \tag{5.37}$$

No extra constants occur in this rate expression, so the pre-exponential factor A with frequency units is identical to the rate expression utilizing a dimensionless concentration.

5.4.2.4. Mass Concentration Form of Rate Expression, Including LIM1TR Form

The molar concentration form can be converted to mass concentration units ρ by multiplying the rate by the molecular weight W and expressing individual concentrations as $c = \rho/W$

$$-\frac{d\rho_R}{dt} = -\frac{d\rho_{MO_2}}{dt} = -W_{MO_2}\frac{dc_{MO_2}}{dt} = Aexp\left(\frac{-E}{RT}\right)\rho_{MO_2} \tag{5.38}$$

Since this reaction has only a single reactant, LIM1TR does not require any further unit conversions and the original frequency factor A is used as the pre-exponential input for LIM1TR.

However, if we assume fast oxidation of a representative solvent EC is effectively zero-order with respect to the abundant solvent (as in Equation 5.34), then the final form of the rate expression for LIM1TR would be

$$-\frac{d\rho_R}{dt} = -\left(W_{MO_2} + \frac{W_{EC}}{5}\right)\frac{dc_{MO_2}}{dt} = Aexp\left(\frac{-E}{RT}\right)\frac{\left(W_{MO_2} + \frac{W_{EC}}{5}\right)}{W_{MO_2}}\rho_{MO_2} \tag{5.39}$$

Where the factor of $W_{EC}/5$ indicates that decomposition of 5 MO_2 units produces enough oxygen to consume a single molecule of EC solvent. In this first-order formulation with a global reaction, the constant molecular weights are lumped with the pre-exponential factor in dimensionless ratios, so the modified pre-exponential factor

$$A' = A\frac{\left(W_{MO_2} + \frac{W_{EC}}{5}\right)}{W_{MO_2}} \tag{5.40}$$

that is used as a LIM1TR input retains units of s^{-1} and does not vary with cell build.

This page intentionally left blank.

6. OTHER FEATURES

6.1. User Specified Reactions

The advanced user who wishes to model reactions outside of the included forms can do so by creating a file reaction_models_user.py in the "Source" directory. The class `basic_rxn` in reaction_models_included.py can be referenced for syntax and the minimum requirements for a user specified reaction. Each reaction class must inherit from reaction_model_base.rxn_model and include functions for evaluating the concentration function and derivative of the concentration function with respect to each species.

The following shows a template for beginning a user specified function:

```
from __future__ import division
import numpy as np
import reaction_model_base

rxn_model_dictionary_user = {
    'My Fancy Reaction': 'fancy_rxn'
}

class fancy_rxn(reaction_model_base.rxn_model):
    def setup(self, reac_man):
        # set up model parameters

    def concentration_function(self, my_v):
        return # a float

    def concentration_derivative(self, my_v):
        '''Derivative of concentration function
        w.r.t. each species.
        '''
        return # a numpy array of length N species
```

The three imports at the top of the template are required, and a model dictionary containing key-value pairs of the input file key and class name for each user defined reaction is also required. In the example `fancy_rxn`, the `setup` function is optional and will be called during parsing of the input file. The function provides a convenient place to calculate useful quantities before solving and parse additional user inputs passed in through the dictionary member variable `rxn_info`.

These user defined inputs are read in from the numbered reaction section and accessed via the case-sensitive key in the input file. For example, in the critical thickness anode reaction the model parameter `C_t` is accessed in the reaction class by `self.rxn_info['C_t']`.

The concentration function (`concentration_function`) and derivative of the concentration function with respect to each species (`concentration_derivative`) both take a vector of the species concentrations as the sole argument. The index of each species in this vector can be accessed with the dictionary member variable `name_map`, which has case-sensitive keys corresponding to the names specified in the "Species" section of the input file. The concentration function must return a float, and the user is cautioned to ensure that the units of this value must balance the units of the pre-exponential factor to produce kg reactants/m^3/s. The concentration derivative function must return a numpy array with a length equal to the number of species. The order of the species derivatives must match the species ordering in the argument array, `my_v`.

6.2. Driving LIM1TR with Python

LIM1TR can be imported as a Python module and run inside another Python script as follows

```
from main_fv import lim1tr_model
model = lim1tr_model('path_to_input_file.yaml')
model.run_model()
```

This functionality is useful in scenarios where automating multiple evaluations of a model with different parameters is desired, such as parameter optimization or parameter sensitivity studies. Between calls to `run_model`, the input variables may be changed by directly accessing the dictionary captured from the input file, `model.parser.cap_dict`. For example, changing the thermal conductivity of a material labeled "A" is done with the following line

```
model.parser.cap_dict['Materials']['A']['k'] = 10
```

The next time `run_model` is called the input dictionary will be re-parsed. It is advisable to save a copy of the original input dictionary in memory to reset the dictionary stored in the model between parameter changes.

The location of the simulation output is controlled with `model.parser.fold_name`. Specifying the output location can be useful for saving all realizations of a parameter study.

7. EXAMPLES

7.1. Calorimetry

An example simulation of differential scanning calorimetry (DSC) can be found in the on-line repository under Examples/DSC and in Appendix A.1. This example models an anode with two reactions: SEI decomposition and the critical thickness anode-electrolyte reaction from Shurtz et al. [8,9]. We note that both reactions produce a lumped species labeled "AllGas" which is given a molecular weight of 1 kg/kmol as the composition of the gas is unknown. In the product specification section of each reaction section, "AllGas" is assigned the number of kmol that will produce the correct amount of total gas based on the assumed stoichiometry. The DSC mode is activated in the Other section of the input file with a temperature rate of 10 K/min. Note that LIM1TR requires MKS units so the temperature rate must be converted, i.e. 10 K/min = 0.1667 K/s. For this operational mode, all boundaries are set to adiabatic and only one control volume is modeled.

Figure 7-1 shows pertinent output quantities for analyzing DSC results, namely the heat flow per gram of initial anode material versus temperature (left) and the species densities over time (right). These plots were generated with the example Python script located at Examples/DSC/DSC_plot.py. This script shows examples of accessing temperature, species densities, and the heat release rate. Additionally, the values in the input file are accessed to calculate the initial density of the anode used for normalizing the heat flow in Figure 7-1 left.

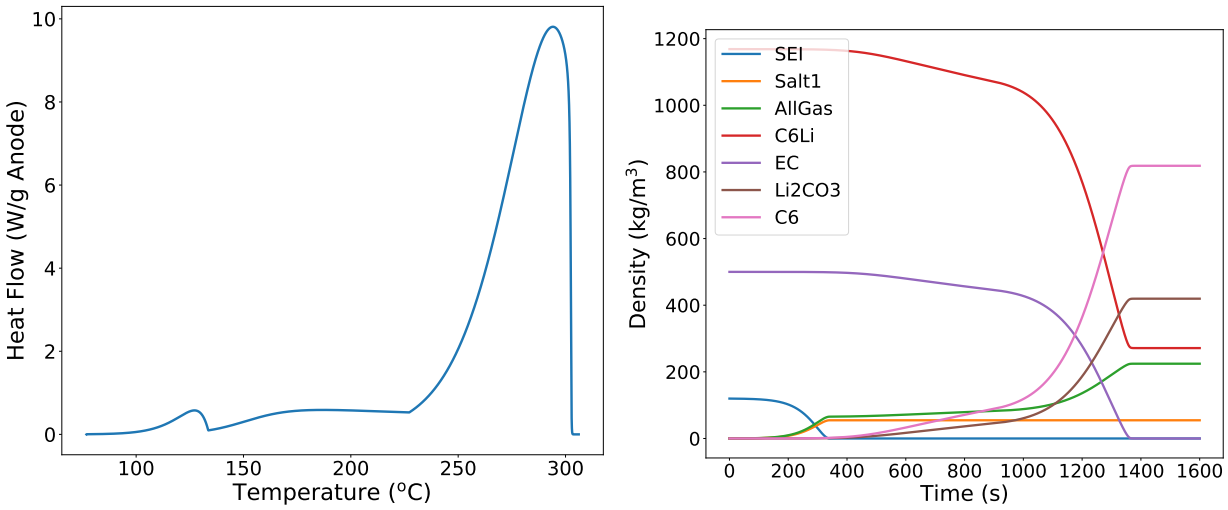

Figure 7-1. Predicted DSC heat flow (left) and species densities (right).

7.2. Stack of Cells with One Reaction

A stack of three cells ignited by a hot metal block is depicted in Figure 7-2. This scenario is typical of thermal runaway propagation modeling efforts such as those by Kurzawski et al. [4] and Li et al. [5]. The input file for this simulation can be found in the on-line repository under Examples/Propagation and in Appendix A.2.

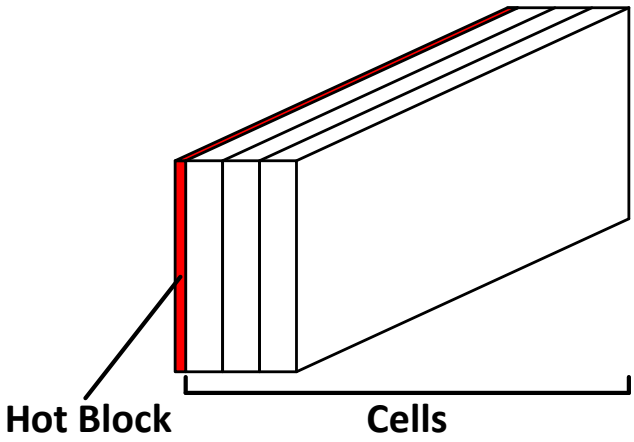

Figure 7-2. Domain of the 3 cell stack with a hot igniter block on the left.

The thermal and kinetic parameters assigned to the cells in this example are not specific to any cell chemistry but approximate the time scales for propagating thermal runaway in pouch cells of this size. A first-order reaction converts a single reactant "R" into a single product "P." We note that the units of the pre-exponential factor must be in kg of total reactants per kg of species "R" per second. As "R" is th only reactant, the units of the pre-exponential factor simplify to 1/s. The initial temperature is specified such that the hot igniter block begins at 973.15 K and the batteries start at room temperature, while only convective losses from the external faces of the stack are modeled.

This example problem comes packaged with a Jupyter Notebook (https://jupyter.org) containing sections of Python code for reading in the pickled output, generating line plots of outputs through the stack at specified times, and plotting the interface temperatures versus time. The line plot code produces a plot of the quantity of interest versus spatial location with a slider that allows for selection of the time step as seen in Figure 7-3 left.

The interface temperature plot provides a useful tool for displaying temperatures between layers of the stack of cells where thermocouples are typically placed during experiments. In Figure 7-3 right, the interface numbers 0, 1, and 2 correspond to the interface between the hot block and first cell, first cell and second cell, and second cell and third cell respectively.

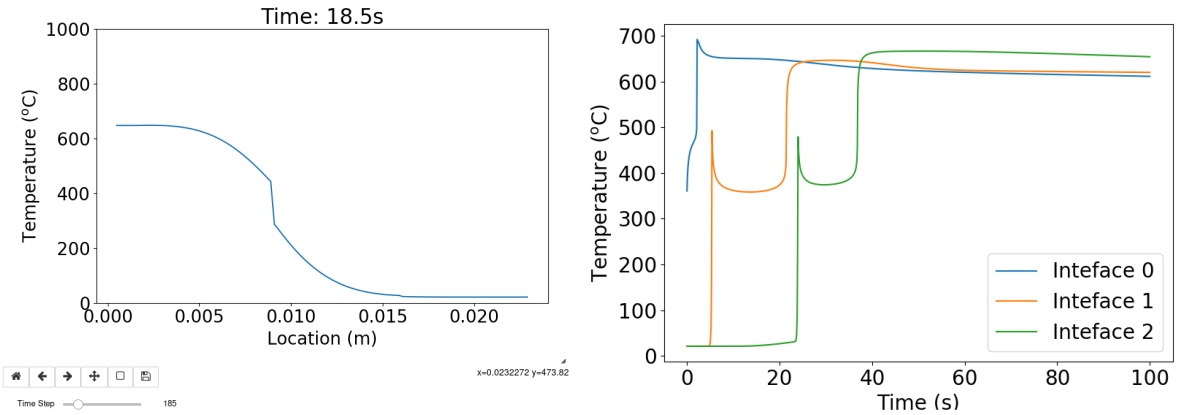

Figure 7-3. Temperature through the stack at 18.5 seconds (left) and interface temperatures versus time (right) from the example propagation case.

This page intentionally left blank.

REFERENCES

[1] HATCHARD, T., MACNEIL, D., BASU, A., AND DAHN, J. Thermal model of cylindrical and prismatic lithium-ion cells. *Journal of The Electrochemical Society 148*, 7 (2001), A755–A761.

[2] INCROPERA, F. P., LAVINE, A. S., BERGMAN, T. L., AND DEWITT, D. P. *Fundamentals of heat and mass transfer*. Wiley, 2007.

[3] KIM, G.-H., SMITH, K., LEE, K.-J., SANTHANAGOPALAN, S., AND PESARAN, A. Multi-domain modeling of lithium-ion batteries encompassing multi-physics in varied length scales. *Journal of the electrochemical society 158*, 8 (2011), A955.

[4] KURZAWSKI, A., TORRES-CASTRO, L., SHURTZ, R., LAMB, J., AND HEWSON, J. C. Predicting cell-to-cell failure propagation and limits of propagation in lithium-ion cell stacks. *Proceedings of the Combustion Institute 38*, 3 (2021), 4737–4745.

[5] LI, Q., YANG, C., SANTHANAGOPALAN, S., SMITH, K., LAMB, J., STEELE, L. A., AND TORRES-CASTRO, L. Numerical investigation of thermal runaway mitigation through a passive thermal management system. *Journal of Power Sources 429* (2019), 80–88.

[6] MACNEIL, D., AND DAHN, J. Test of reaction kinetics using both differential scanning and accelerating rate calorimetries as applied to the reaction of li x coo2 in non-aqueous electrolyte. *The Journal of Physical Chemistry A 105*, 18 (2001), 4430–4439.

[7] PLETCHER, R. H., TANNEHILL, J. C., AND ANDERSON, D. *Computational fluid mechanics and heat transfer*. CRC press, 2012.

[8] SHURTZ, R. C., ENGERER, J. D., AND HEWSON, J. C. Predicting high-temperature decomposition of lithiated graphite: Part I. review of phenomena and a comprehensive model. *Journal of The Electrochemical Society 165*, 16 (2018), A3878–A3890.

[9] SHURTZ, R. C., ENGERER, J. D., AND HEWSON, J. C. Predicting high-temperature decomposition of lithiated graphite: Part II. passivation layer evolution and the role of surface area. *Journal of The Electrochemical Society 165*, 16 (2018), A3891–A3902.

[10] STRANG, G. On the construction and comparison of difference schemes. *SIAM journal on numerical analysis 5*, 3 (1968), 506–517.

[11] TORRES-CASTRO, L., KURZAWSKI, A., HEWSON, J., AND LAMB, J. Passive mitigation of cascading propagation in multi-cell lithium ion batteries. *Journal of The Electrochemical Society 167*, 9 (2020), 090515.

[12] VIRTANEN, P., GOMMERS, R., OLIPHANT, T. E., HABERLAND, M., REDDY, T., COURNAPEAU, D., BUROVSKI, E., PETERSON, P., WECKESSER, W., BRIGHT, J., VAN DER WALT, S. J., BRETT, M., WILSON, J., MILLMAN, K. J., MAYOROV, N., NELSON, A. R. J., JONES, E., KERN, R., LARSON, E., CAREY, C. J., POLAT, İ., FENG, Y., MOORE, E. W., VANDERPLAS, J., LAXALDE, D., PERKTOLD, J., CIMRMAN, R., HENRIKSEN, I., QUINTERO, E. A., HARRIS, C. R., ARCHIBALD, A. M., RIBEIRO, A. H., PEDREGOSA, F., VAN MULBREGT, P., AND SCIPY 1.0 CONTRIBUTORS. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods 17* (2020), 261–272.

APPENDIX A. Example Input Files

This appendix serves as a reference for the input files corresponding to the examples in Chapter 7. The syntax in these input files can be referenced in the context of Chapter 3 at the time of publication of this report and the release of version 1.0 of LIM1TR. Any syntactical changes introduced before the next major release will be detailed in a change log in the on-line repository https://github.com/ajkur/lim1tr.

A.1. Calorimetry

```
Materials:
    A:
        k: 0.5                  # W/m/K
        rho: 2001.56            # kg/m^3
        cp: 778.0               # J/kg/K

Species:
    Names: ['EC', 'C6Li', 'SEI', 'Salt1', 'Li2CO3', 'C6', 'AllGas', 'Container']
    Initial Mass Fraction: [0.2498172134, 0.5837978725, 0.0598347191,
        0.0, 0.0, 0.0, 0.0, 0.106550195]
    Molecular Weights: [88.062, 79.007, 161.952, 73.89, 73.89, 72.066, 1.0, 0.0]
    Material Name: A

Reactions:
    1:
        A: 8.158513762e+16      # (kg reactants/s) / (kg SEI * m^3)^(1/2)
        E: 16236.69493          # K
        R: 1
        H: -635000.0            # J/kg reactants
        Reactants:
            'SEI': 1            # kmol
        Products:
            'Salt1': 1          # kmol
            'AllGas': 88.062    # kmol
        Orders:
            'SEI': 0.5
    2:
        A: 3.2718e+13           # kg reactants/kg C6Li/s
```

```
        E: 16236.69493          # K
        R: 1
        H: -2287100.0           # J/kg reactants
        Reactants:
            'C6Li': 2           # kmol
            'EC': 1             # kmol
        Products:
            'C6': 2             # kmol
            'Li2CO3': 1         # kmol
            'AllGas': 28.054    # kmol
        Type: 'Zcrit'
        BET_C6: 1.1             # BET surface area
        tau_crit: 0.08          # Critical anode conversion
        C_t: 72.5               # Model parameter for SEI growth
        Y_Graphite: 0.532509493 # Initial mass fraction of graphite
        Electrolyte Limiter:
            Species: 'EC'
            Limiting Constant: 1.333249314 # kg EC/m3

Domain Table:
    Material Name: [A]
    Thickness: [0.005]          # m
    dx: [0.005]                 # m

Boundary:
    External:
        Type: Adiabatic
    Left:
        Type: Adiabatic
    Right:
        Type: Adiabatic

Time:
    Run Time: 1600.0            # s
    T Initial: 350.0            # K
    dt: 0.1                     # s
    Order: 1                    # Conduction time stepper order

Other:
    Y Dimension: 0.003          # m
    Z Dimension: 0.003          # m
    Reaction Only: 1            # Only solve reactions
    DSC Mode: 1                 # Use the DSC model for temperature evolution
    DSC Rate: 0.166666667       # DSC heating rate in K/s
```

A.2. Stack of Cells with One Reaction

```
Materials:
    Battery:
        k: 0.5          # W/m/K
        rho: 1800       # kg/m3
        cp: 800         # J/kg/K

    Hot Block:
        k: 237          # W/m/K
        rho: 2700       # kg/m3
        cp: 900         # J/kg/K

Species:
    Names: ['R', 'P', 'Inert']
    Initial Mass Fraction: [0.35, 0.0, 0.65]    # kg species/kg total
    Molecular Weights: [1.0, 1.0, 0.0]          # kg/kmol
    Material Name: Battery

Reactions:
    1:                  # 1 R -> 1 P
        A: 1.0e+9       # kg total reactants/kg species R/s
        E: 110000       # J/mol
        R: 8.314        # J/mol/K
        H: -1.44e+6     # J/kg total reactants
        Reactants:
            'R': 1      # kmol
        Products:
            'P': 1      # kmol
        Orders:
            'R': 1

Domain Table:
    Material Name: [Hot Block, Battery, Battery, Battery]
    Thickness: [0.002, 0.007, 0.007, 0.007]                 # m
    dx: [0.001, 0.0002, 0.0002, 0.0002]                     # m
    Contact Resistance: [0.002, 0.004, 0.004]               # m2/K/W

Boundary:
    External:
        Type: Convection
        h: 10               # W/m2/K
        T: 294.15           # K
    Left:
```

```
        Type: Adiabatic
    Right:
        Type: Adiabatic

Time:
    Run Time: 100.0                                     # s
    T Initial: [973.15, 294.15, 294.15, 294.15]     # K
    dt: 0.01                                            # s
    Output Frequency: 10
    Order: 2

Other:
    Y Dimension: 0.12       # m
    Z Dimension: 0.04       # m
```

DISTRIBUTION

Email—External

Name	Company Email Address	Company Name
Dr. Imre Gyuk, DOE Program Director	imre.gyuk@hq.doe.gov	DOE Office of Electricity

Email—Internal

Name	Org.	Sandia Email Address
Technical Library	1911	sanddocs@sandia.gov

This page intentionally left blank.

Sandia National Laboratories

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.