Python-based Equivalent Circuit Network (PyECN) Modelling Framework for Lithium-ion Batteries

Shen Li [a, b, *], Sunil Rawat [a], Tao Zhu [a], Gregory J. Offer [a, b], Monica Marinescu [a, b]

* Corresponding author.
E-mail address: lishensean@gmail.com (S. Li).

Affiliations:

a. Department of Mechanical Engineering, Imperial College London, London SW7 2AZ, United Kingdom

b. The Faraday Institution, Harwell Science and Innovation Campus, Didcot OX11 0RA, United Kingdom

Abstract

Lithium-ion batteries are now an essential part of future transportation and electricity networks. This has led to an increase in demand for accurate and fast models to support their development and deployment. Commercial codes can represent a barrier to entry for some, particularly academic research groups, and open-source modelling environments for physics-based models of batteries, such as PyBaMM have been developed in response to this. Equivalent circuit network (ECN) models are routinely used as a faster alternative to physics-based models, as they are easier and cheaper to parameterise, and are more suited for problems requiring a high level of discretisation, such as cell and system design and optimisation. However, there is no equivalent open-source code for this type of model. In response to this, the open-source Python package PyECN (**Py***thon* **E***quivalent* **C***ircuit* **N***etwork*), developed at Electrochemical Science

& Engineering Group (ESE) in Imperial College London, provides a comprehensive modelling framework for Lithium-ion battery simulation. PyECN is able to simulate the electrical and thermal performances of cylindrical, prismatic and pouch cell form factors. The code package presented here should be of immediate interest to cell manufacturers, module and pack designers and battery researchers.

1. Introduction

The ability to correctly predict the multi-physics behaviour of lithium-ion batteries is critical for their safety, performance, cost and lifetime. There are multiple commercial software products that include battery models, e.g., STAR CCM+ [1], COMSOL [2], ANSYS [3], Simulink [4] etc. Most of those products provide graphical user interface (GUI) for user to create battery models in a convenient way. These software are also integrated within general modelling frameworks covering multi-disciplines. However, they still have some limitations. Firstly, access to ultimate control. The frameworks of those commercial software are usually black boxes which provide barriers for researchers who want to improve or add functionality rather than just use them as tools to solve engineering problems. Secondly, licensing can be a significant cost burden for battery research, particularly in academia.

In response, there are many professional coding frameworks established for academic battery research, some open source, such as PyBaMM [7], DandeLiion [8], DUALFOIL [9], LIONSIMBA [10] and MSMD [11][12] etc. Those codes have played an important role in helping understand many aspects of the fundamentals and mechanisms for Lithium-ion battery. However, all of these codes solve electrochemical models with complex differential algebraic equations (DAE), and they are computationally expensive in nature. A significant number of fundamental parameters in the equations are also needed and they are not easily obtained. Most of those codes are based on 1D or 2D model assumptions, i.e., discretisation is sacrificed for lower computational cost. For a fully 3D model, the simulation of long-term degradation tests is unlikely to be computationally affordable [13]. These codes therefore have limited applicability for complex battery engineering problems.

In this work, an equivalent circuit network (ECN) battery modelling framework PyECN (**Py***thon* **E***quivalent* **C***ircuit* **N***etwork*) is created and coded in the open-source Python language. The package is capable of predicting the 3D electro-thermal behaviour of batteries of 3 different form factors: pouch, cylindrical and prismatic cells. The detailed internal structures are considered such as the metal can and tab configuration. Several models of different commercially available cells have been experimentally validated. To create a new cell model, the parametrisation is easy and affordable. As an example, the effect of the tab design and thermal management strategies on the battery performance is studied. The package is being made open source to speed up the development of new battery technologies for the benefit of all. The model and code package presented here should be of immediate interest to cell manufacturers, module and pack designers and battery researchers.

2. Overview of PyECN

PyECN solves the coupled equations of electro-thermal problems defined by Eq. (a-f). The equations are discretized by finite difference method. The schematic of the distributed ECN model for cylindrical cell and pouch cell (as examples) is shown in **Fig. 1**. In this modelling framework, the whole cell is distributed into coupled electrical and thermal ECN units. Within each ECN unit, current collector and electrodes are considered, and they have different electrical and thermal properties. For the electrical model in the electrode domain, the typical local ECN includes a voltage source E_s

representing the Open Circuit Voltage (OCV), a series resistance R_0 representing the instantaneous ohmic resistance, and a set of Resistor-Capacitor (RC) branches, R_i and C_i that capture the transient response. **Fig. 1** illustrates the set of three Resistor-Capacitor (RC) branches where resistor R_i & capacitor C_i represents R_1, R_2 and R_3 and C_1, C_2, and C_3 respectively for $i = 1, 2$ and 3. According to Kirchhoff's voltage law, the terminal voltage $\Delta\phi^{\mathrm{El}}$ of the electrode pair unit is given by Eq. (a), where I_i is the branch current in the resistor R_i and I is the current in resistor R_0. For the current collector components, charge balance conservation is considered as in Eq. (b), where σ is the conductivity of the current collector and ϕ^{CC} its local potential.

$$\left\{\begin{array}{ll}
\Delta\phi^{\mathrm{El}} = E_{\mathrm{s}} - \sum_{i=1}^{3} R_i I_i - R_0 I & \text{(a)} \\
\sigma\nabla\cdot(\nabla\phi^{\mathrm{CC}}) = \sigma\dfrac{\partial^2\phi^{\mathrm{CC}}}{\partial x^2} + \sigma\dfrac{\partial^2\phi^{\mathrm{CC}}}{\partial y^2} = 0 & \text{(b)} \\
\rho c\dfrac{\partial T}{\partial t} = \lambda_x\dfrac{\partial^2 T}{\partial x^2} + \lambda_y\dfrac{\partial^2 T}{\partial y^2} + \lambda_z\dfrac{\partial^2 T}{\partial z^2} + q & \text{(c)} \\
Q_{\mathrm{conv}} = h(T - T_{\mathrm{amb}}) & \text{(d)} \\
q^{\mathrm{El}} = \left(\sum_{i=1}^{3} I_i^2 R_i + I^2 R_0 + IT\dfrac{\mathrm{d}E_{\mathrm{s}}}{\mathrm{d}T}\right) / V^{\mathrm{El}} & \text{(e)} \\
q^{\mathrm{CC}} = \left(\dfrac{1}{2}\sum_{i=1}^{4}\left(I_i^{\mathrm{CC}}\right)^2 R_i^{\mathrm{CC}}\right) / V^{\mathrm{CC}} & \text{(f)}
\end{array}\right.$$

For the thermal model in the electrode component, the anode, cathode and separator are lumped into one bulk material (electrode pair) in each thermal ECN. The heat transfer equation for the electrode pair, and current collector materials are given by Eq. (c), where ρ, c, λ and are the mass density, heat capacity, thermal conductivity and temperature of the electrodes pair. λ_x, λ_y and λ_z are the corresponding equivalent thermal conductivities in the three directions, and q is the heat source. For current collector domain, Eq. (c) is still applied with thermal properties of aluminum for positive current collector foil or copper for negative one. For the metal can outside, the heat generation source is ignored, i.e., the q term in Eq. (c) is 0. The code accepts two types of thermal boundary conditions: convective boundary condition and Dirichlet-type fixed boundary condition. A convection boundary condition is imposed as Eq. (d), where Q_{conv} is the heat transferred due to convection, h is convective heat transfer coefficient, T is the temperature at the boundary and T_{amb} is the ambient temperature. Dirichlet-type temperature boundary condition have also been implemented, as well as combinations of convection and Dirichlet for each of the surfaces of the metal can.

The electrical model and thermal model are fully coupled, i.e., the electrical parameters (resistance, capacitance and OCV) are function of temperature, while heat generation from the electrical model contributes to the heat source of the thermal model. The overall heat source q^{El} for the electrode pair is the sum of reversible and irreversible heat as expressed in Eq. (e), where V^{El} is the electrode pair volume in each element. The heat source q^{CC} for current collector is written as Eq. (f), where I_i^{CC} and R_i^{CC} are the current and resistance in the aluminum or copper foil domains, as shown in **Fig. 1**, while V^{CC} is the current collector volume.

The ECN model describes a temporal and spatial evolution of various variables of interest such as current density and temperature, for a given load and thermal management choice.

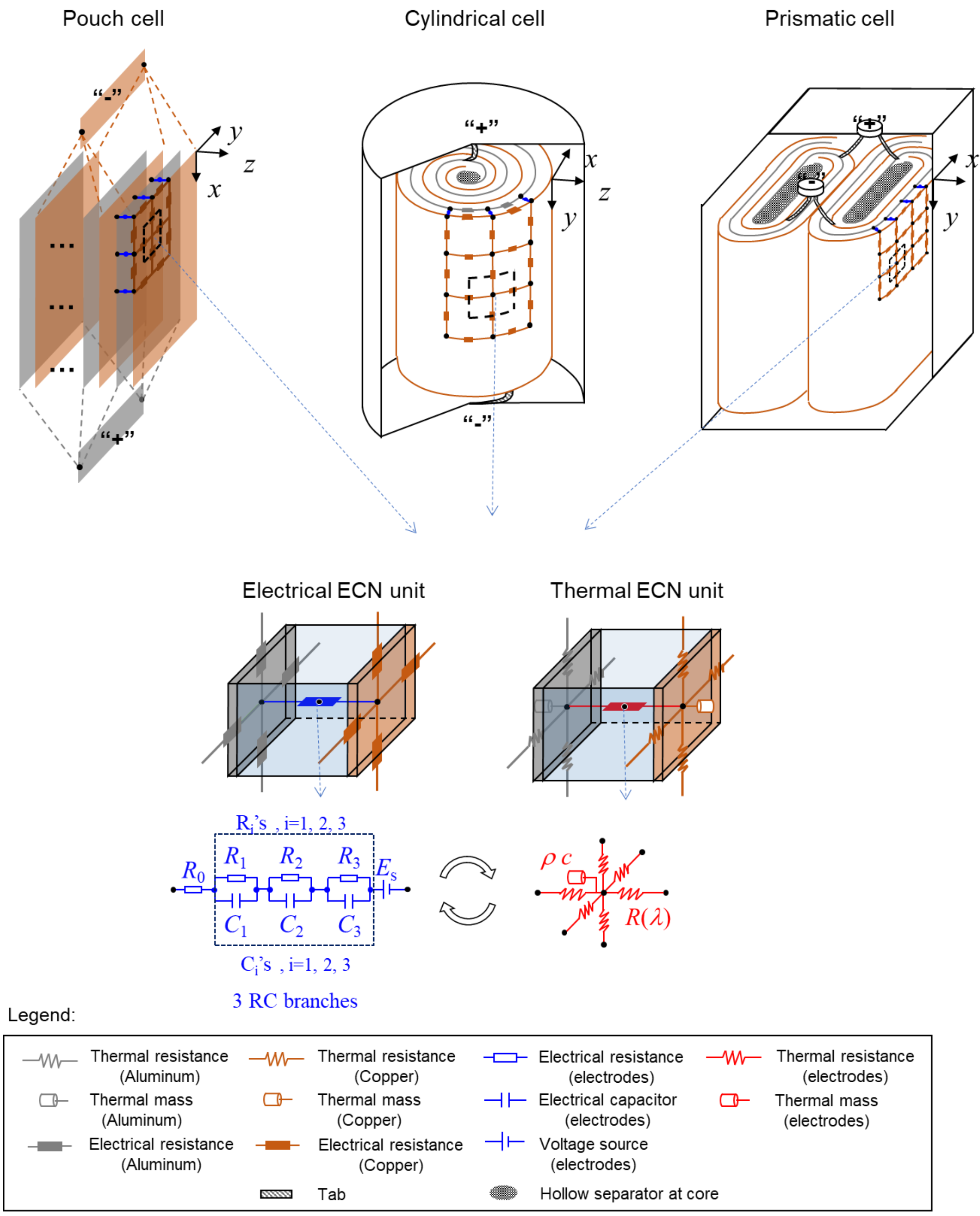
Pouch cell
Cylindrical cell
Prismatic cell
"-"
"+"
x
y
z
Electrical ECN unit
Thermal ECN unit
R_i's , i=1, 2, 3
R_0
R_1
R_2
R_3
E_s
C_1
C_2
C_3
C_i's , i=1, 2, 3
3 RC branches
ρ c
R(λ)
Legend:
Thermal resistance (Aluminum)
Thermal mass (Aluminum)
Electrical resistance (Aluminum)
Thermal resistance (Copper)
Thermal mass (Copper)
Electrical resistance (Copper)
Tab
Electrical resistance (electrodes)
Electrical capacitor (electrodes)
Voltage source (electrodes)
Hollow separator at core
Thermal resistance (electrodes)
Thermal mass (electrodes)

Fig. 1. Schematic of the ECN model for pouch cell, cylindrical cell, and prismatic cell.

The workflow of PyECN is illustrated in **Fig. 2**. PyECN package is composed of multiple modules. The simulation setup required (geometry dimensions, discretization, electrical and thermal parameters, and boundary conditions) is specified in a TOML-format configuration file and then imported in the main pyecn module. The necessary lookup tables such as R_i's & C_i's (as a function of SoC, temperature), entropic coefficient ($\frac{dE_s}{dT}$) (as a function of SoC) and the experimental data used for comparison with modelling results are also imported into the main pyecn module. The pyecn module processes all the inputs and makes them accessible in the main computation loop and other modules/classes. In the preprocessing stage, the core battery class module core.py and other required classes such as module.py, module_4T.py and classes related to the external thermal entity (e.g. can_prismatic.py module for prismatic cell casing) are invoked according to the user inputs. The auxiliary functions and variables which will remain unchanged in the entire simulation are generated here before the loop iteration. This manipulation will accelerate the simulation speed.

Core battery class module core.py is the class which has functions shared by all three form factor battery classes (cylindrical.py, pouch.py, prismatic.py i.e., classes for cylindrical cell, pouch cell, and prismatic cell respectively). Class module, LUT.py, is used for reading all the necessary look up tables and imported into core battery class module i.e. core.py. All developed new future classes should also be imported into core battery class module i.e. core.py.

Finally, the solutions of the equations are output by the solver. The code provides both explicit solver and implicit solver. Both solvers are self-coded and the implicit one

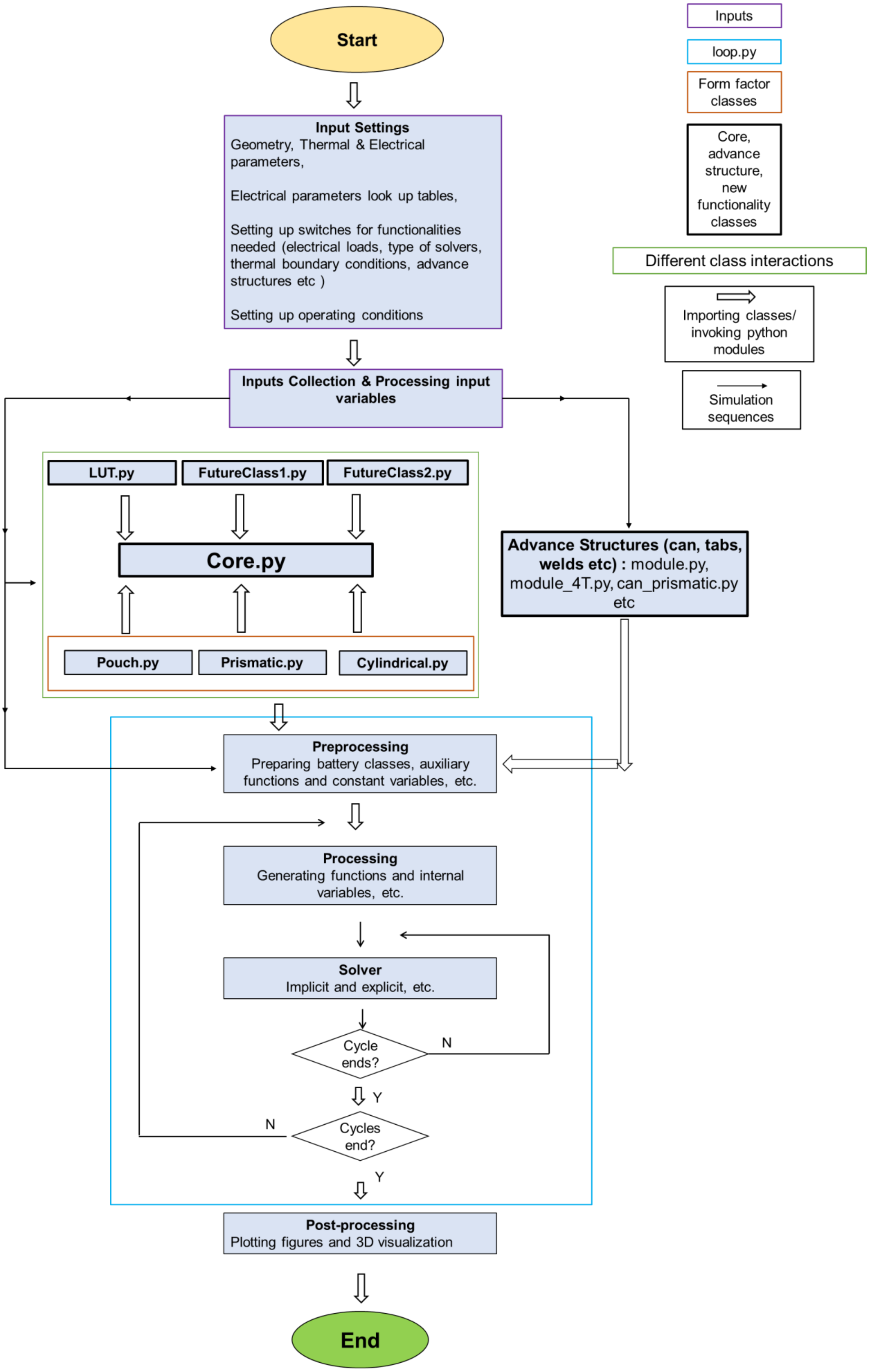
Start
Input Settings
Geometry, Thermal & Electrical parameters,
Electrical parameters look up tables,
Setting up switches for functionalities needed (electrical loads, type of solvers, thermal boundary conditions, advance structures etc)
Setting up operating conditions
Inputs Collection & Processing input variables
LUT.py
FutureClass1.py
FutureClass2.py
Core.py
Pouch.py
Prismatic.py
Cylindrical.py
Advance Structures (can, tabs, welds etc) : module.py, module_4T.py, can_prismatic.py etc
Preprocessing
Preparing battery classes, auxiliary functions and constant variables, etc.
Processing
Generating functions and internal variables, etc.
Solver
Implicit and explicit, etc.
Cycle ends?
N
Y
Cycles end?
N
Y
Post-processing
Plotting figures and 3D visualization
End
Inputs
loop.py
Form factor classes
Core, advance structure, new functionality classes
Different class interactions
Importing classes/ invoking python modules
Simulation sequences

Fig. 2. Flow chart of the PyECN code.

uses the scipy.sparse.linalg.spsolve function from the python Scipy package. There are two iteration loops. The first loop is for single test profile, e.g., the constant current discharge or drive cycle simulation will be executed within this loop. Simulation involving cycling will be executed by the second iteration loop. After the simulation, the results will be post-processed to make plots and 3D visualization according to the user inputs.

3. Usage example

3.1 Simulation of 4680 cylindrical cell with tabless design

To show the flexibility of PyECN on considering the internal structure, the 4680 cylindrical cell with tabless design is simulated. For the electrical tab configuration, the positive and negative current collectors at top/base sides are folded to make plane connections with the plane terminals (metal can top/base sides). The cell model is set to discharge at 1C constant current, with the cut-off voltage of 2.5 V at end of discharge. The cooling temperature of 25 ºC is applied on the base of the metal can. All the other surfaces including the top surface and side surface are thermally insulated. In the majority of previous cylindrical cell models [12][14][15], the thermal boundary conditions are applied directly on jellyroll models for model simplification. The setup model here is able to capture the internal thermal pathway which significantly affects the temperature performance [6]. The electrical parameters and for this 4680 cell are from a parameterized commercial 2170 cylindrical cell (LG M50T 5 Ah) [6] scaled by the electrode plate area. The rated capacity for this 4680 cell is 27.07 Ah. The thermal properties (thermal conductivity, mass density and heat capacity) are also taken from the commercial cell [6].

The simulation results from the PyECN code are shown in **Fig. 3**. The internal temperature distribution at any time step could be traced. **Fig. 3** shows the internal temperature distribution at end of discharge, visualized using the Mayavi Python package. The terminal voltage, the volume-average temperature rise and temperature difference between the maximum and minimum for the discharge process are shown. The electrodes temperature, current density and state of charge (SoC) are also shown in jellyroll unwound view. It is visible that the jellyroll top is hotter than the bottom for this base cooling scheme, so that the jellyroll top is discharged more than the bottom as the resistance is low at high temperature for this cell. The SoC for the jellyroll bottom is higher than the top. The current density for both jellyroll top and bottom edge are higher than the middle, this is because the resistance for top area with high temperature and the bottom area with high SoC are lower than the middle area.

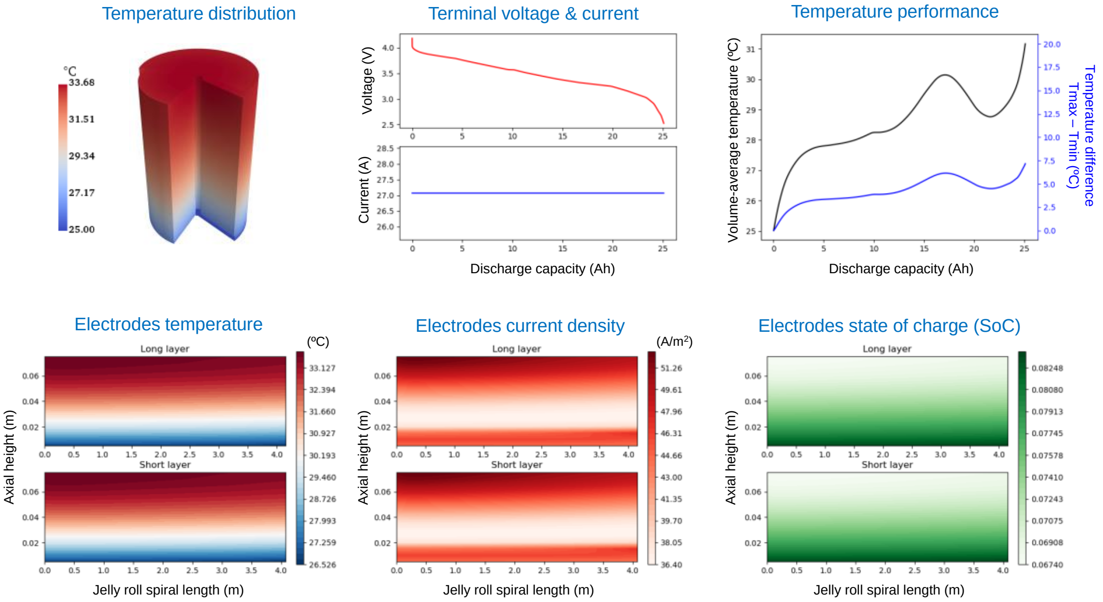

Fig. 3. PyECN simulation results of 4680 tabless cylindrical cell under base cooling scenario for 1C constant current discharge process.

3.2 Simulation of pouch cell under different thermal management schemes

The simulation was performed for a commercial pouch cell (Kokam 5 Ah, model: SLPB1543140H5) using PyECN. Tab cooling and surface cooling scenarios are two typical thermal management approach for pouch cell. In the simulation, both scenarios were applied for the pouch cell. The cell is discharged at 6C constant current as this power cell was designed for high C rate load. The temperature distribution at end of discharge (after 600 s of discharge) is shown in **Fig. 4**. For the simulations, an ambient temperature of 25ºC was selected. This temperature was also used as cooling temperature. In the tab cooling case, the colder tab surface had a heat transfer coefficient of 1810 $W\cdot m^{-2}\cdot K^{-1}$ and warmer tab had a heat transfer coefficient of 1080 $W\cdot m^{-2}\cdot K^{-1}$. All other faces of the cell were insulated. In case of the surface cooling, the temperature on both faces was fixed. All other faces were thermally insulated. It is clearly visible that the volume-average temperature in the tab cooling scenario is much higher than that in surface cooling scenario. The thermal gradient in the surface-cooled cell is more obvious than that in the tab-cooled cell. The volume-average temperature and thermal gradient behaviour for the two cells reproduces the experimental findings [16] and simulation results [17]. The PyECN is capable of capturing the thermal performance of this pouch cell under different thermal management strategies.

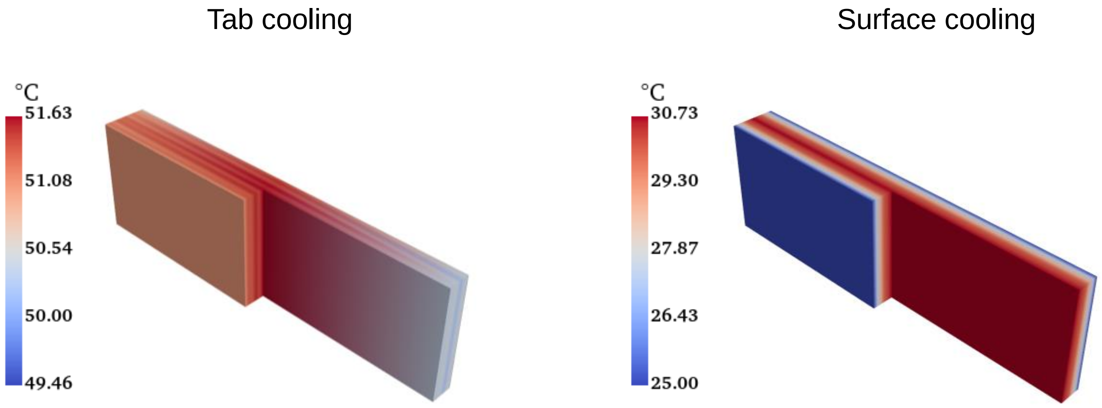

Fig. 4. PyECN simulation results of pouch cell at end of 6C constant current discharge. A part of the cell is cut away for the visualization of the internal temperature.

3.3 Simulation of cell with complicated structures

3.3.1 Prismatic cell with dual jellyroll:

Code for a custom prismatic cell with dual jellyroll inside and a metal can outside is included as an example. The cell model is formed by invoking the prismatic jellyroll class (prismatic.py, module.py & module_4T.py) and the metal can class (can_prismatic.py). The dual jellyrolls are electrically connected in parallel. For the metal can only the thermal model is enabled for this simulation case. The heat transfer between the dual jellyrolls and metal can so that the thermal management applied on the metal can will affect the thermal behaviors of the cell.

The electrical parameters of this custom cell are generated by scaling the parameterized LG M50T commercial cell with the electrode plate area, assuming the prismatic cell is a larger version of the LG M50T cell. The rated capacity is 110 Ah for this prismatic cell.

3.3.2 Pouch cell with welding lines and tab lines:

Another example is demonstrated on a pouch cell with welding lines and tab lines. The cell tab and weld design could significantly affect the heat rejection rate [18][19][20][21]. This example model is focused on the pouch cell with delicate weld and tab structures. In this model, the pouch cell is thermally connected with the welding lines for both positive and negative side. The two welding lines are thermally connected with the two tabs at both sides, where the active cooling is applied. A 1C constant current discharge case is simulated for the pouch cell whose surfaces are in convection. The temperature behavior for the cell is shown in **Fig. 5** . For the scenario shown in **Fig. 5**, the heat transfer coefficient on the faces with the tabs was selected as 60 $W·m^{-2}·K^{-1}$ inside the faces, 70 $W·m^{-2}·K^{-1}$ on the edges, and 1080 $W·m^{-2}·K^{-1}$ on the Al-current collector and 1810 $W·m^{-2}·K^{-1}$ on the Cu current collector. The heat transfer coefficient on the largest two faces was 7 $W·m^{-2}·K^{-1}$ and on the remaining faces was 120 $W·m^{-2}·K^{-1}$. On all faces the same cooling temperature of 25ºC was applied. The electrical parameters of the pouch cell are from the parameterized Kokam cell (model: SLPB1543140H5), but the cell form factor is modified to demonstrate modelling the tabs.

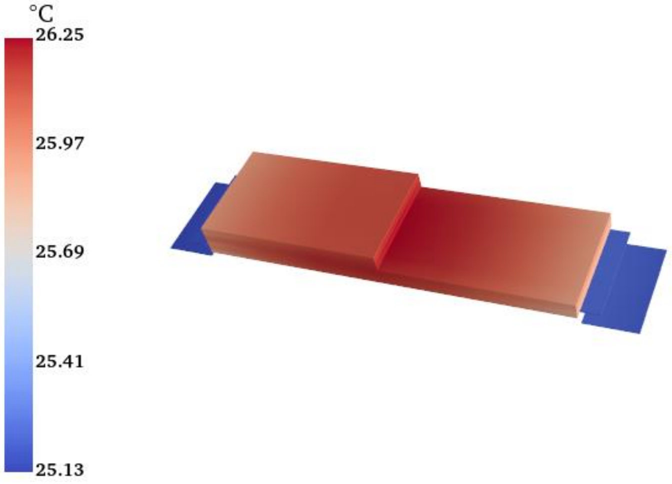

Fig. 5 PyECN simulation results of pouch cell at end of 1C constant current discharge. A part of the stack for pouch cell is cut away for the visualization of the internal temperature.

4. Availability

PyECN package is available on GitHub (https://www.github.com/ImperialCollegeLondon/PyECN) with instructions. Future updates will be made available as they are developed, and we would welcome additional creators who want to help to get in touch. The current version requires Python 3.10.5.

Conclusions

The ability to predict the internal states of Lithium-ion batteries is critical for battery development to improve safety, performance, cost and lifetime. PyECN package is a modelling framework created for Lithium-ion battery research and engineering applications. It has been demonstrated that the PyECN is able to model Lithium-ion batteries of 3 different form factors: pouch, cylindrical and prismatic cells. The PyECN is based on distributed equivalent circuit network model, which offers an excellent balance between functionality and computational cost. This PyECN package has the following key features simultaneously:

- Ease of parametrization (both for electrical and thermal models)
- 3D distribution with custom meshing
- Electro-thermal fully coupling
- Ability to consider detailed internal and external structures
- Ability to model multi-cell system
- Consideration of different components (electrodes and current collectors)
- Open source
- Open framework for adding more physics

The cylindrical cell model constructed using the PyECN has been experimentally validated in our previous work [6]. The PyECN package presented here is expected to be an efficient tool for battery design and control. More functionalities are being developed.

Acknowledgements

This work was generously supported by the EPSRC Faraday Institution Multi-Scale Modelling project (EP/S003053/1, grant number FIRG003), the Innovate UK WIZer project (grant number 104427) and the EPSRC CASE (grant number EP/R513052/1) award by Williams Advanced Engineering. The authors are grateful to Dr Daniel Cummins for his support and contribution to improving the PyECN code.

Credit statement

Shen Li: Conceptualisation, Methodology: Creator and lead developer of PyECN, Writing and Review, **Sunil Rawat**: Contributor of PyECN, Discussion, Writing and Review, **Tao Zhu**: Contributor of PyECN, Discussion, Writing and Review, **Gregory J Offer**: Conceptualisation, Funding Acquisition, Supervision, Writing – Review & Editing, **Monica Marinescu**: Conceptualisation, Funding Acquisition, Supervision, Writing – Review & Editing.

Nomenclature

Abbreviations	*Description*
C	C-rate
conv	Convection
DAE	Differential algebraic equations
ECN	Equivalent Circuit Network
Eq.	Equation
OCV	Open Circuit Voltage
RC	Resistor-Capacitor
SoC	State of Charge

Latin symbols Unit Description

Symbols	**Unit**	**Description**
c	J·kg^{-1}·K^{-1}	Specific heat capacity
C_i	F	Branch capacitance of one electrical ECN element
E_s	V	Voltage source i.e. Open Circuit Voltage (OCV)
h	W·m^{-2}·K^{-1}	Convective heat transfer coefficient
I	A	Total current for electrodes in an electrical ECN element i.e. current in the resistor R_0
I_i	A	Branch current for electrodes in an electrical ECN element i.e. current in the resistor R_i
I_i^CC	A	Constituent current for current collector in an ECN element
q	W·m^{-3}	Heat source in a thermal ECN element per unit volume
q^CC	W·m^{-3}	Overall current collector heat generation rate per unit volume
q^El	W·m^{-3}	Overall electrode heat generation rate per unit volume
Q_conv	W·m^{-2}	Heat transferred due to convection
R_0	Ω	Series resistance of one electrical ECN element
R_i	Ω	Branch resistance of one electrical ECN element
R_i^CC	Ω	Current collector resistance in aluminium or copper foil domains
t	s	Time
T	K	Temperature of unit volume
T_ambient	K	Ambient temperature for convective heat transfer

V^{CC}	m^3	Current collector foil volume
V^{El}	m^3	Volume of electrodes pair in each ECN unit
x	m	Spatial coordinate along angular direction (Prismatic & Cylindrical) or longest side of stack (Pouch)
y	m	Spatial coordinate along axial direction (Prismatic & Cylindrical) or shortest side of stack (Pouch)
z	m	Spatial coordinate along radial direction (Prismatic & Cylindrical) Or across the stack thickness (Pouch)

Greek symbols	Unit	Description
φ^{CC}	V	Electric potential of current collector
φ^{El}	V	Terminal voltage of electrodes pair unit
λ	$\mathrm{W \cdot m^{-1} \cdot K^{-1}}$	Thermal conductivity
λ_{x}	$\mathrm{W \cdot m^{-1} \cdot K^{-1}}$	Thermal conductivity in x direction
λ_{y}	$\mathrm{W \cdot m^{-1} \cdot K^{-1}}$	Thermal conductivity in y direction
λ_{z}	$\mathrm{W \cdot m^{-1} \cdot K^{-1}}$	Thermal conductivity in z direction
ρ	$\mathrm{kg \cdot m^{-3}}$	Mass density
σ	$\mathrm{S \cdot m^{-1}}$	Electrical conductivity of current collector

Subscripts	*Description*
CC	Current collector
conv	Heat convection

Al	Aluminum foil
Cu	Copper foil
El	Electrodes including anode, cathode and separator

References

[1] Siemens Industries Digital Software. Simcenter STAR-CCM+, version 2021.1, Siemens 2021. n.d.

[2] COMSOL Inc. COMSOL multiphysics reference manual, version 5.4. www.comsol.com. n.d.

[3] Ansys® Academic Research Mechanical, Release 18.1 n.d.

[4] Documentation, S., 2020. Simulation and Model-Based Design, MathWorks. Available at: https://www.mathworks.com/products/simulink.html. n.d.

[5] OpenFOAM v10 User Guide. The OpenFOAM Foundation. https://openfoam.org n.d.

[6] Li S, Kirkaldy N, Zhang C, Gopalakrishnan K, Amietszajew T, Bravo Diaz L, et al. Optimal cell tab design and cooling strategy for cylindrical lithium-ion batteries. J Power Sources 2021;492:229594. https://doi.org/10.1016/j.jpowsour.2021.229594.

[7] Sulzer V, Marquis SG, Timms R, Robinson M, Chapman SJ. Python Battery Mathematical Modelling (PyBaMM). J Open Res Softw 2021;9:1–8. https://doi.org/10.5334/JORS.309.

[8] Korotkin I, Sahu S, O'Kane SEJ, Richardson G, Foster JM. DandeLiion v1: An Extremely Fast Solver for the Newman Model of Lithium-Ion Battery (Dis)charge. J Electrochem Soc 2021;168:060544. https://doi.org/10.1149/1945-7111/ac085f.

[9] Newman J. FORTRAN programs for the simulation of electrochemical systems. www.cchem.berkeley.edu/jsngrp/ fortran.html. n.d.

[10] Torchio M, Magni L, Gopaluni RB, Braatz RD, Raimondo DM. LIONSIMBA: A Matlab Framework Based on a Finite Volume Model Suitable for Li-Ion

Battery Design, Simulation, and Control. J Electrochem Soc 2016;163:A1192–205. https://doi.org/10.1149/2.0291607jes.

[11] Kim G-H, Smith K, Lee K-J, Santhanagopalan S, Pesaran A. Multi-Domain Modeling of Lithium-Ion Batteries Encompassing Multi-Physics in Varied Length Scales. J Electrochem Soc 2011;158:A955. https://doi.org/10.1149/1.3597614.

[12] Erhard S V., Osswald PJ, Wilhelm J, Rheinfeld A, Kosch S, Jossen A. Simulation and Measurement of Local Potentials of Modified Commercial Cylindrical Cells. J Electrochem Soc 2015;162:A2707–19. https://doi.org/10.1149/2.0431514jes.

[13] Kim H-K, Choi JH, Lee K-J. A Numerical Study of the Effects of Cell Formats on the Cycle Life of Lithium Ion Batteries. J Electrochem Soc 2019;166:A1769–78. https://doi.org/10.1149/2.0261910jes.

[14] Sturm J, Rheinfeld A, Zilberman I, Spingler FB, Kosch S, Frie F, et al. Modeling and simulation of inhomogeneities in a 18650 nickel-rich , silicon- graphite lithium-ion cell during fast charging. J Power Sources 2019;412:204–23. https://doi.org/10.1016/j.jpowsour.2018.11.043.

[15] Lee KJ, Smith K, Pesaran A, Kim GH. Three dimensional thermal-, electrical-, and electrochemical-coupled model for cylindrical wound large format lithium-ion batteries. J Power Sources 2013;241:20–32. https://doi.org/10.1016/j.jpowsour.2013.03.007.

[16] Hunt IA, Zhao Y, Patel Y, Offer J. Surface Cooling Causes Accelerated Degradation Compared to Tab Cooling for Lithium-Ion Pouch Cells. J Electrochem Soc 2016;163:A1846–52. https://doi.org/10.1149/2.0361609jes.

[17] Zhao Y, Patel Y, Zhang T, Offer GJ. Modeling the Effects of Thermal Gradients Induced by Tab and Surface Cooling on Lithium Ion Cell Performance. J Electrochem Soc 2018;165:A3169–78. https://doi.org/10.1149/2.0901813jes.

[18] Brand MJ, Schmidt PA, Zaeh MF, Jossen A. Welding techniques for battery cells and resulting electrical contact resistances. J Energy Storage 2015;1:7–14. https://doi.org/10.1016/j.est.2015.04.001.

[19] Bazinski SJ, Wang X. Thermal Effect of Cooling the Cathode Grid Tabs of a

Lithium-Ion Pouch Cell. J Electrochem Soc 2014;161:A2168–74. https://doi.org/10.1149/2.0731414jes.

[20] Zhao Y, Diaz LB, Offer GJ. How to Cool Lithium Ion Batteries : Optimising Cell Design using a Thermally Coupled Model 2019;166:2849–59. https://doi.org/10.1149/2.0501913jes.

[21] Hales A, Diaz LB, Marzook MW, Patel Y, Offer G. The Cell Cooling Coefficient : A Standard to Define Heat Rejection from Lithium-Ion Batteries 2019;166:2383–95. https://doi.org/10.1149/2.0191912jes.