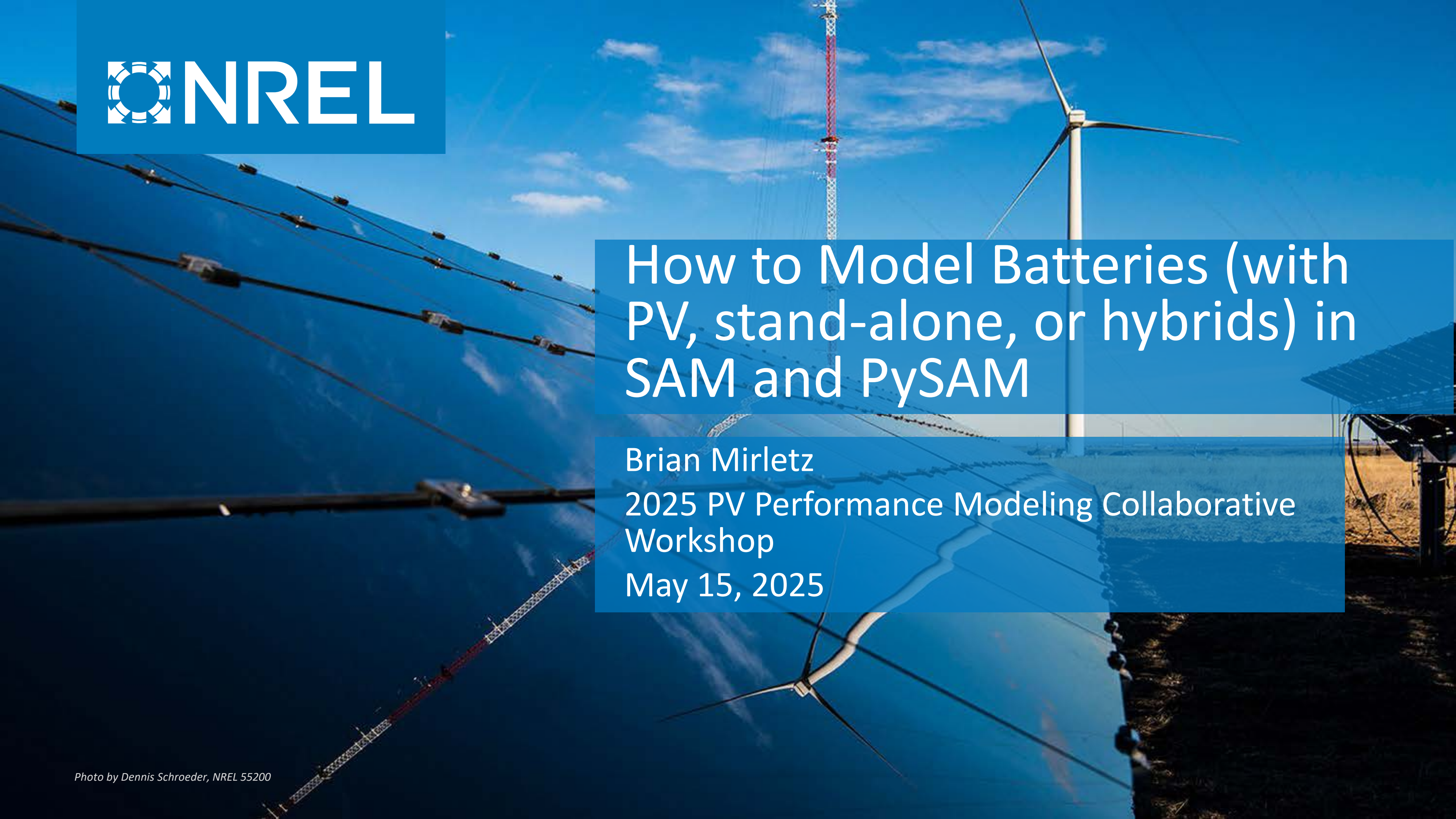


NREL

How to Model Batteries (with PV, stand-alone, or hybrids) in SAM and PySAM

Brian Mirletz

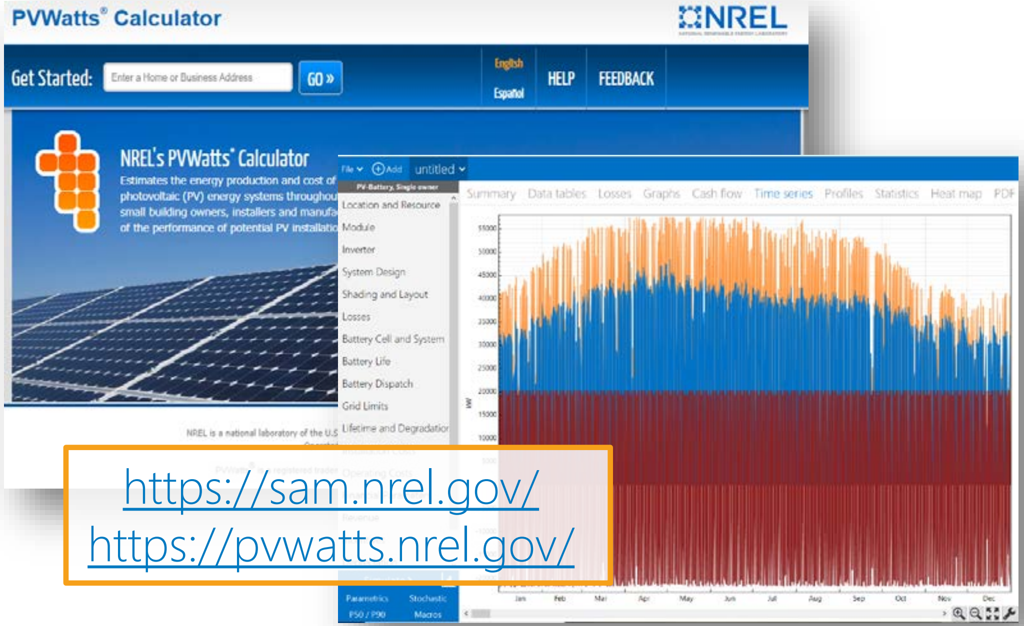
2025 PV Performance Modeling Collaborative Workshop

May 15, 2025

Photo by Dennis Schroeder, NREL 55200

System Advisor Model (SAM) & PVWatts

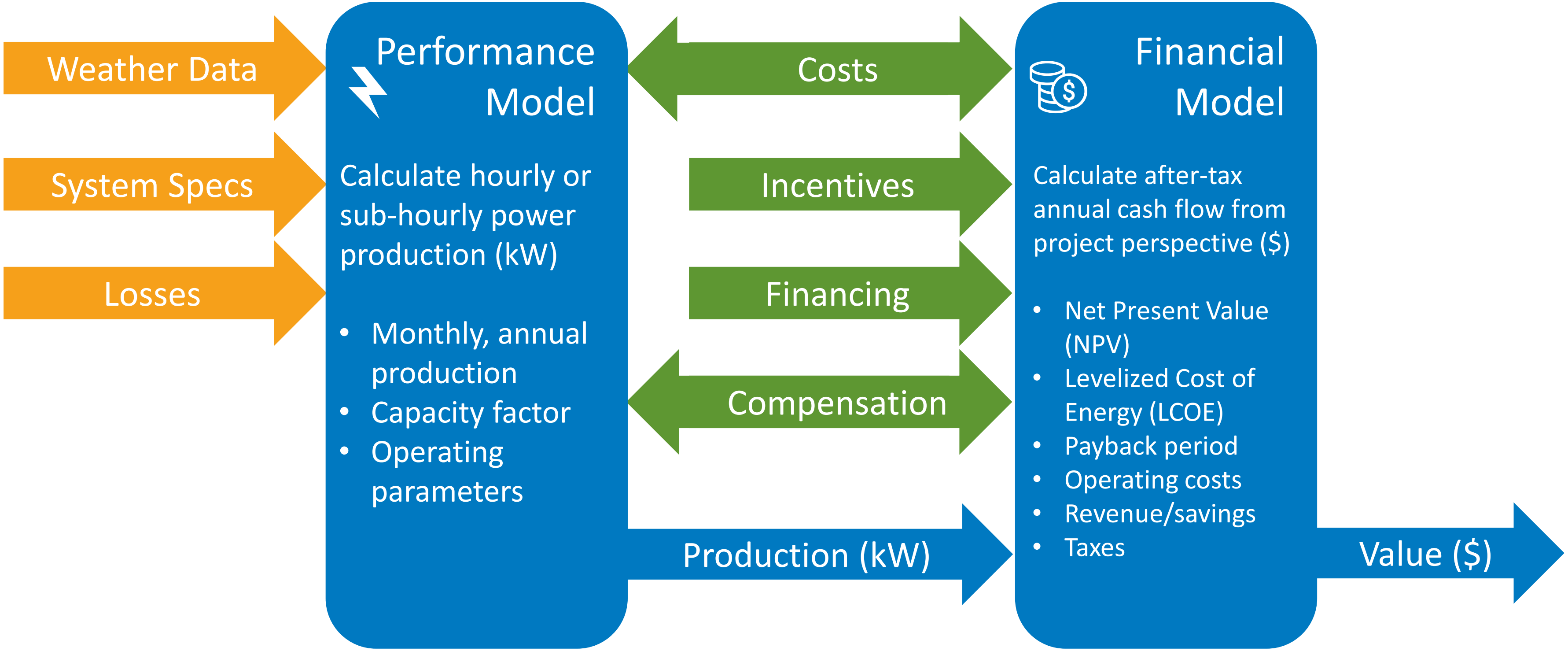
Free software that enable detailed performance and financial analysis for renewable energy systems



https://sam.nrel.gov/
https://pvwatts.nrel.gov/

- ✓ Desktop application
- ✓ PVWatts web tool & API
- ✓ Software development kit
- ✓ PySAM Python package
- ✓ Open-source code
- ✓ Extensive documentation
- ✓ User support

Model Structure



Technologies

Photovoltaic

Energy storage

- Electric battery
- Electric thermal storage

Concentrating solar power

Industrial process heat

Marine energy

Wind power

Fuel cell

Geothermal power

Solar water heating

Biomass combustion

Generic system

PLUS: HYBRID SYSTEMS

Financial Models

Power purchase agreements

- Single owner
- Partnership flips
- Sale leaseback

Residential

Commercial

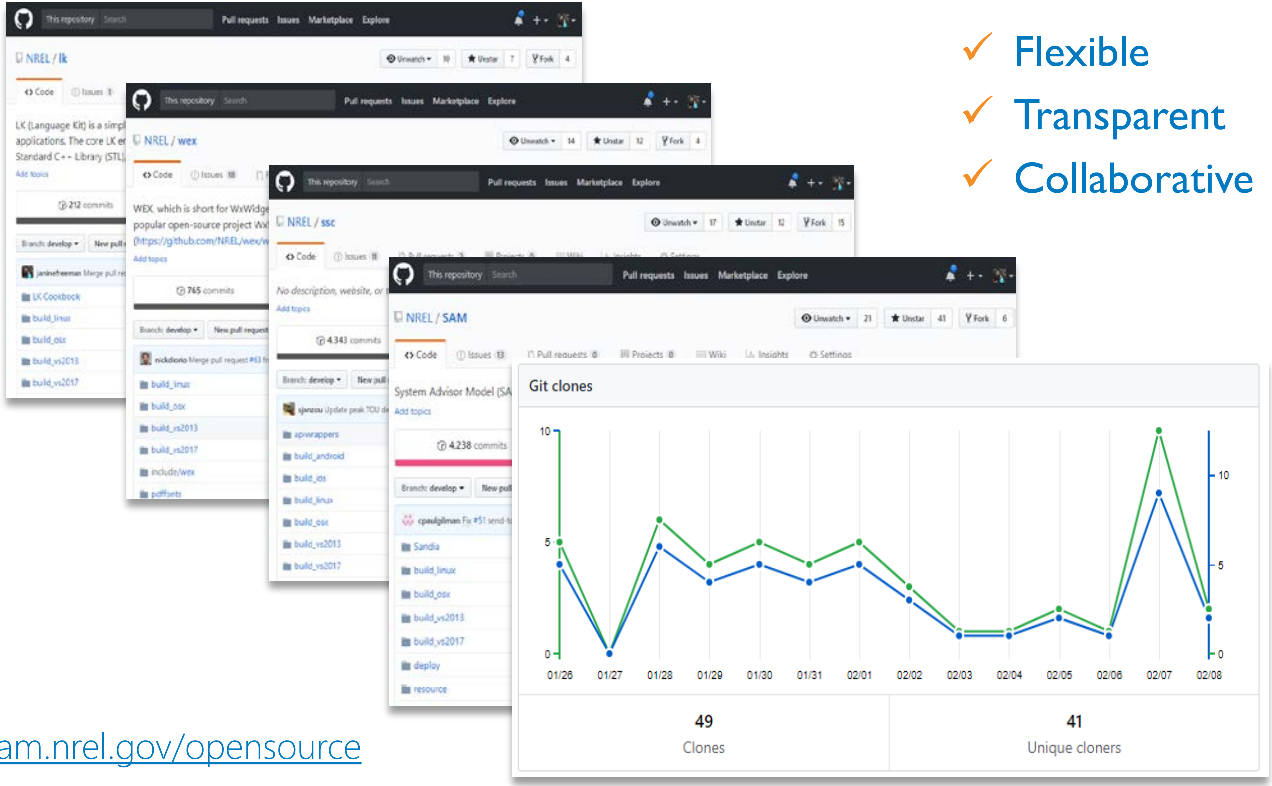
Third party ownership

Merchant plant

Community solar

Simple Levelized Cost of Energy (LCOE) calculator

Open-Source Code

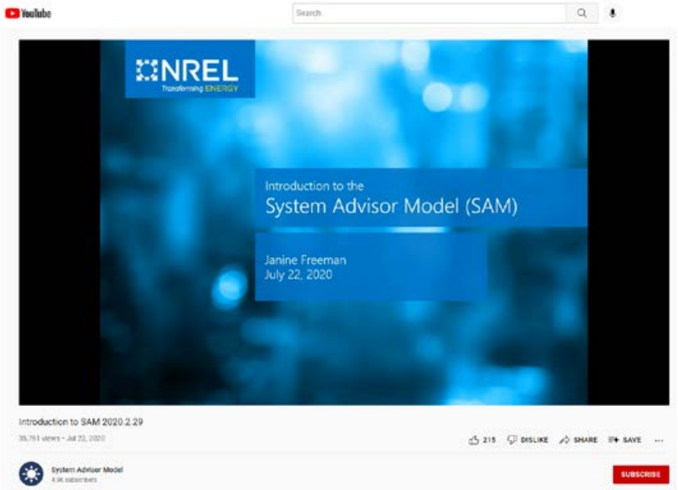


- ✓ Flexible
- ✓ Transparent
- ✓ Collaborative

http://sam.nrel.gov/opensource

Extensive Help Documentation

- Website – http://sam.nrel.gov
  - Support Forum – Ask your question!
  - General info/ online help file / contact info
- YouTube Channel
  - https://www.youtube.com/user/SAMDemoVideos
  - All prior webinars and seminars
- Monthly Round Table sessions
  - SAM team asks questions live and interactively
- Email Support
  - SAM support can provide email support if question/bug is involved



Outline

1. Battery Model Overview
2. Configurations and Dispatch
3. Demo: Hybrids with self-consumption dispatch
4. Demo: Dispatch with Interconnection and Curtailment Limits
5. PySAM
6. Demo: SAM Battery with pvlib Generation
7. Questions

Outline

1. **Battery Model Overview**
2. Configurations and Dispatch
3. Demo: Hybrids with self-consumption dispatch
4. Demo: Dispatch with Interconnection and Curtailment Limits
5. PySAM
6. Demo: SAM Battery with pvlib Generation
7. Questions

Why batteries in SAM?

- Exploding interest in modeling hybrid and battery systems generally
- Existing SAM users want to know about resiliency of the system
    - Ability to implement a power outage in SAM enables ability to answer "Will this system adequately provide backup during an outage?"
    - Of greater interest to users modeling systems on fragile grids.
- Storage modeling for future R&D
    - Robust hour by hour calculation of the LCOS (levelized cost of storage) not just using average lifetime values for usage
    - SAM is also being used to model long-duration thermal storage and potentially other options as well.

Component Models

Cell Capacity

- Sizing for power and capacity of the battery system

Lifetime Fade

- How capacity decreases over time due to calendar (pure storage) and cycling fade
- Calculating needed replacements as a function of performance

Voltage Curve

- Voltage discharge curve as a function of the state-of-charge (SOC)

Thermal

- Heat transfer with environment for instantaneous thermal effects on capacity, resistance, and lifetime

Voltage Models

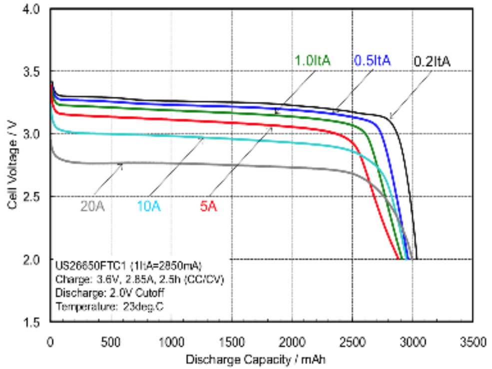
Examples

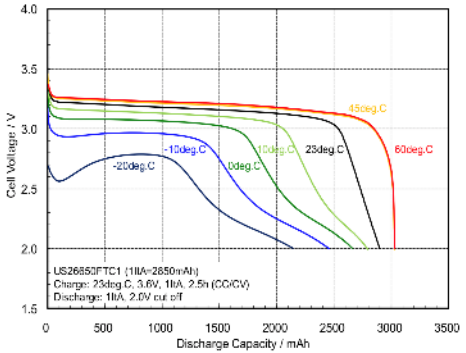
- Sony Fortelion IJ1001M - LFP
- Toshiba SCiB SPCS-LIB010A – LMO/LTO
- LG Chem RESU HV - NMC

Lithium Ion models share same capacity, voltage, and lifetime models

Lead-acid and flow batteries use different underlying models

Parameters from scientific paper and manufacturer datasheets

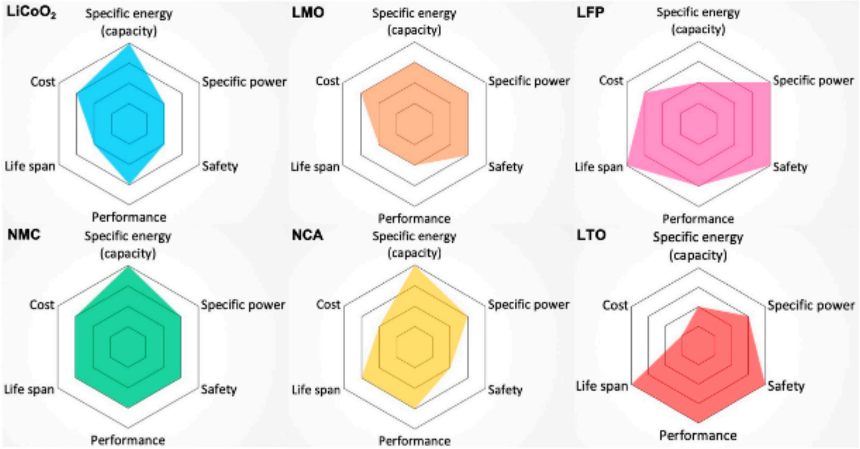




Yasuda, Masayuki. Sony energy storage system using olivine type battery.

Lifetime Models

- Cycle and Calendar Model
  - Based on Rainflow stress algorithm
  - Potentially available from battery data sheets
- NMC (Nickel/Manganese/Cobalt)/Graphite
  - Analytical model based on Smith et al 2017
- LMO (Lithium Manganese Oxide)/LTO (Lithium Titanate)
  - Analytical model from the NREL-led BTMS (Behind the Meter Storage) consortium



Miao et al. Current Li-Ion Battery Technologies in Electric Vehicles and Opportunities for Advancements. Energies 2019.

Thermal Model

- Account for lifetime and capacity effects of cell temperature
- Simple heat transfer model for calculating battery internal temperature

Battery Thermal

Thermal Behavior

The battery thermal model assumes that the battery with specific heat Cp is in an environment at the specified environment temperature. Heat transfer between the battery and its environment is proportional to the heat transfer coefficient h.

Thermal Properties

Specific heat Cp 1500 J/KgK

Heat transfer coefficient h 100 W/m2K

Environment temp option Enter single fixed temperature

Single environment temperature 25 C

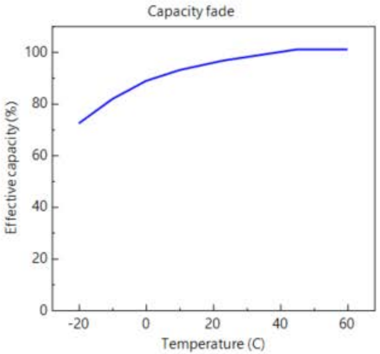
Time series environment temperature Edit array... C

The time series environment temperature array must match the timestep of the selected weather file. For hybrid systems, the length of the array must match the smallest timestep weather file.

Import... Export... Copy Paste

Rows: 7

Temp (C)	Capacity(%)
-20	72.3333
-10	81.8
0	88.8
10	93
23	96.6667
45	101
60	101



Physical Properties

Specific energy per mass 70 Wh/kg

Battery mass 3,428,575.776 kg

Compute battery surface area as cube

Specific energy per volume 82 Wh/L

Battery volume 2,926.833 m3

Energy capacity of one module 400.000 kWh

Battery surface area 18,000.023 m2

Surface area of one module 30.000 m2

The battery thermal model does not account for power required for thermal conditioning. Use the inputs under "Battery Losses" or on the Losses page to account for power used by battery heating or cooling equipment.

Outline

1 Battery Model Overview

2 **Configurations and Dispatch**

3 Demo: Hybrids with self-consumption dispatch

4 Demo: Dispatch with Interconnection and Curtailment Limits

5 PySAM

6 Demo: SAM Battery with pvlib Generation

7 Questions

Battery Configurations

Configuration	Connection Type	Charging Source	Financial Models
Detailed PV + Battery	DC or AC	PV or Grid	All
PVWatts + Battery	AC	PV or Grid	Behind the meter
Custom Generation Profile + Battery	AC	“System” or Grid	All
Stand Alone Battery	AC	Grid	All
Hybrid Systems	AC	System(s) or Grid	Single Owner, Host Developer

Behind The Meter Dispatch Modes

Dispatch Mode	Inputs	Use Case
Peak Shaving	PV and Load forecast	Peak Demand Charges
Input Grid Power Targets	Monthly or time series targets	Specify more detailed peak power
Custom Dispatch	Time series	PySAM / outside optimization
Manual Dispatch	Schedule by hour and month	Energy Arbitrage
Retail Rates Dispatch	Upcoming PV and Load forecast and utility rates	Energy arbitrage and peak shaving
Grid outage	Current PV and Critical Load	Resilience
Self-consumption	Current PV and Load	System sizing for 24/7 energy

Bold: defaults
Italics: Available in PVWatts-Battery model

Front of Meter Dispatch Modes

Dispatch Mode	Inputs	Use Case
Automated dispatch	PV and Price forecast	Energy Arbitrage
PV Smoothing	PV generation and (optionally) forecast	Ramp rate violation reduction
Custom Dispatch	Time series	PySAM / outside optimization
Manual Dispatch	Schedule by hour and month	Energy Arbitrage

Bold: defaults

Outline

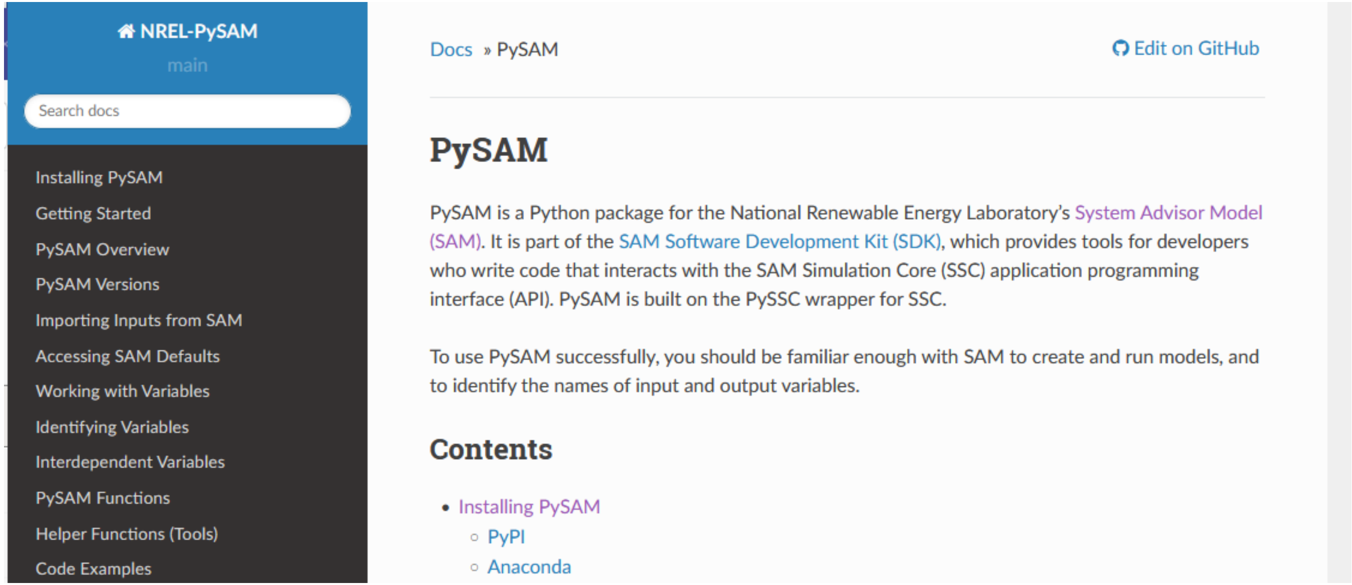
Outline

Outline

What is PySAM?

Python package that enables you to run the underlying modules that make up a simulation in SAM

- Package name nrel-pysam
- Unit modules called compute_modules in the SSC code


NREL-PySAM
main
Search docs

Installing PySAM
Getting Started
PySAM Overview
PySAM Versions
Importing Inputs from SAM
Accessing SAM Defaults
Working with Variables
Identifying Variables
Interdependent Variables
PySAM Functions
Helper Functions (Tools)
Code Examples

Docs » PySAM
Edit on GitHub

PySAM

PySAM is a Python package for the National Renewable Energy Laboratory's System Advisor Model (SAM). It is part of the SAM Software Development Kit (SDK), which provides tools for developers who write code that interacts with the SAM Simulation Core (SSC) application programming interface (API). PySAM is built on the PySSC wrapper for SSC.

To use PySAM successfully, you should be familiar enough with SAM to create and run models, and to identify the names of input and output variables.

Contents

- Installing PySAM
  - PyPI
  - Anaconda

What is PySAM?

Python package that enables you to run the underlying modules that make up a simulation in SAM

- Unit modules called compute_modules in the SSC code

A single simulation is a process chaining together multiple unit modules

- Order
- Information needs to be passed from one to the next

Generic System-Battery – Commercial Owner

Generic system model with battery storage. Renewable energy system displaces commercial building electric load.

Configuration name for defaults: *"GenericBatteryCommercial"*

GenericSystem, Battery, Grid, Utilityrate5, Cashloan

https://nrel-pysam.readthedocs.io/en/main/

What is PySAM?

Python package that enables you to run the underlying modules that make up a simulation in SAM

- Unit modules called compute_modules in the SSC code

A single simulation is a process chaining together multiple unit modules

- Order
- Information needs to be passed from one to the next
- Assembled behind the scenes in SAM user interface

PySAM, and SAM's other software development kits, expose these unit modules so that they can be customized and embedded in software applications

https://nrel-pysam.readthedocs.io/en/main/

What is PySAM?

Python package that enables you to run the underlying modules that make up a simulation in SAM

- Unit modules called compute_modules in the SSC code

A single simulation is a process chaining together multiple unit modules

- Order
- Information needs to be passed from one to the next
- Assembled behind the scenes in SAM user interface

PySAM, and SAM's other software development kits, expose these unit modules so that they can be customized and embedded in software applications

PySAM does NOT contain all the features in the SAM GUI

https://nrel-pysam.readthedocs.io/en/main/

Outline

Related Links

- SAM Website: https://sam.nrel.gov/
- SAM open source: https://github.com/NREL/SAM/
- Link to the PySAM example: https://github.com/NREL/pysam/tree/main/Examples/PVPMC_pvlib_PySAM_example
- PySAM documentation: https://nrel-pysam.readthedocs.io/en/main/
- NREL lifetime and degradation modeling: https://github.com/NREL/BLAST-Lite

Thank you! Questions?

Janine (Freeman) Keith – project lead, photovoltaic and wind models
Brian Mirletz – project lead, software, battery models
Nate Blair – emeritus lead, financials, costs, systems
Darice Guittet – software development, battery models
Matt Prilliman – photovoltaic, geothermal, and marine energy models
Steve Janzou – programming, utility rates, financials (subcontractor)
Paul Gilman – user support and documentation (subcontractor)
Ty Neises, William Hamilton, and Taylor Brown – concentrating solar power models

Photo from iStock-627281636

Thank you!

www.nrel.gov

NREL/PR-7A40-93555

This work was authored by the National Renewable Energy Laboratory for the U.S. Department of Energy (DOE) under Contract No. DE-AC36-08GO28308. Funding provided by the U.S. Department of Energy Office of Energy Efficiency and Renewable Energy Strategic Analysis Team. The views expressed in the article do not necessarily represent the views of the DOE or the U.S. Government. The U.S. Government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this work, or allow others to do so, for U.S. Government purposes.

pvlib

- Organized into individual functions
- Fully customizable in Python
- Focused primarily on PV models
- Implemented in Python
- May be better suited for PV research applications

Both!

- Robust implementations of PV modeling algorithms
- End-to-end PV model with limited model choices available in each tool
  - ModelChain in pvlib
  - pvsamv1 in PySAM
- Open-source
- Example scripts to help you get started
- Available via pip install
- Shared module and inverter libraries
  - PySAM for module coefficients
  - pvlib for inverter coefficients
- Great for use in your own Python project!

System Advisor Model PySAM

- Primarily organized into functions for complete resource-to-energy system simulation
  - Some sub-functions available (module, inverter, irradiance)
- Minimal coding required to perform a PV simulation
- Export system setups to/from the SAM desktop tool
- Implemented in C++ and accessed as a Python package
- Includes financial models
- May be better suited for batch analysis or PVsyst-type simulations

Lifetime Models

Cycle and Calendar model uses empirical data to compute degradation based on modelled cycles. Sum of cycle and calendar models is used. (This is a change as of 2024.12.12)

Cycle and Calendar Degradation

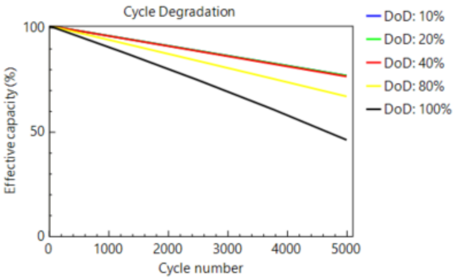
Use the options and tables below to define cycle degradation and calendar degradation curves. SAM automatically updates these inputs when you choose a battery type on the Battery Cell and System page. The empirical calendar degradation model is suitable for Lithium-ion batteries. See help for details.

Cycle Degradation

Import... Export... Copy Paste

Rows: 25

Depth-of-discharge (%)	Cycles Elapsed	Capacity (%)
10	0	100.853
10	1250	94.884
10	2500	88.9147
10	3750	82.9454
10	5000	76.9761
20	0	100.853
20	1250	94.8798
20	2500	88.9063

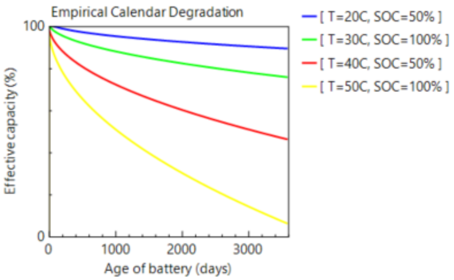


Calendar Degradation

None
Empirical
Custom

q = 1.02 - k_cal * sqrt(t)

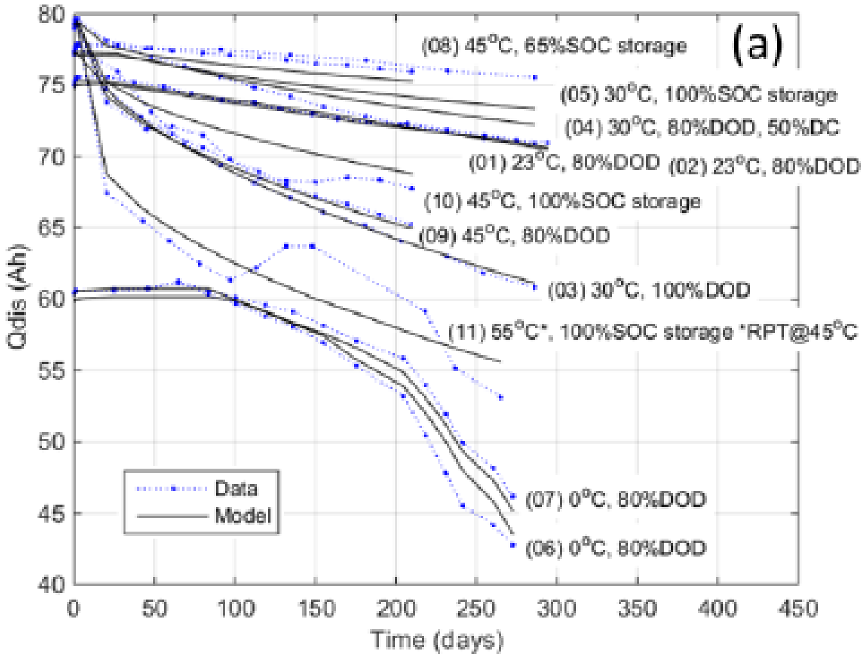
k_cal = 0.00266*exp[-7280*(1/T-1/296)]*exp[939*(SOC/T-1/296)]



DOD = depth of discharge
SOC = state of charge

Lifetime Models

Latest version of SAM includes analytical models for cycles, storage state of charge, and temperature



DOD = depth of discharge
SOC = state of charge

Smith, Kandler, et al. "Life prediction model for grid-connected Li-ion battery energy storage system." *2017 American Control Conference (ACC)*. IEEE, 2017.