Pacific Northwest NATIONAL LABORATORY

PNNL-38021

MCOR User Guide

August 2025

Sarah F Newman
Emily L Barrett

U.S. DEPARTMENT of ENERGY

Prepared for the U.S. Department of Energy
under Contract DE-AC05-76RL01830

PACIFIC NORTHWEST NATIONAL LABORATORY
operated by
BATTELLE
for the
UNITED STATES DEPARTMENT OF ENERGY
under Contract DE-AC05-76RL01830

Printed in the United States of America

Available to DOE and DOE contractors from
the Office of Scientific and Technical Information,
P.O. Box 62, Oak Ridge, TN 37831-0062
www.osti.gov
ph: (865) 576-8401
fox: (865) 576-5728
email: reports@osti.gov

Available to the public from the National Technical Information Service
5301 Shawnee Rd., Alexandria, VA 22312
ph: (800) 553-NTIS (6847)
or (703) 605-6000
email: info@ntis.gov
Online ordering: http://www.ntis.gov

PNNL-38021

MCOR User Guide

August 2025

Sarah F Newman
Emily L Barrett

Prepared for
the U.S. Department of Energy
under Contract DE-AC05-76RL01830

Suggested Citation for MCOR tool:

Newman, S., K. Shiozawa, J. Follum, E. Barrett, T. Douville, T. Hardy, and A. Solana. 2020. “A Comparison of PV Resource Modeling for Sizing Microgrid Components.” *Renewable Energy* 162: 831–843. https://doi.org/10.1016/j.renene.2020.08.074.

Pacific Northwest National Laboratory
Richland, Washington 99354

Overview

The Microgrid Component Optimization for Resilience (MCOR) tool simulates the operation of a renewable energy, battery, and back-up generator microgrid under a large range of outage conditions to understand how a potential system can meet the resilience goals of a particular site. It is an open-source, command line, Python-based tool that produces an output Excel spreadsheet as well as several types of plots to enable a user to compare different microgrid system sizes and costs. It is intended for high-level system planning and opportunity identification, and not for detailed electric system modeling and design. The tool includes a range of input parameters that can be adjusted or tuned to provide a more custom analysis as needed.

Acknowledgments

The development of MCOR was funded by the United States Army Reserve, the U.S. Department of Defense's Army Office of Energy Initiatives, and the U.S. Department of Energy's Office of Water Power under contract number DE-AC05-76RL0-1830.

Acronyms and Abbreviations

ASP	Alternative Solar Profiles
DHI	Diffuse Horizontal Irradiance
DNI	Direct Normal Irradiance
GHI	Global Horizontal Irradiance
LBNL	Lawrence Berkeley National Laboratory
MCOR	Microgrid Component Optimization for Resilience
MEA	Marine Energy Atlas
NREL	National Renewable Energy Laboratory
NSRDB	National Solar Resource Database
PNNL	Pacific Northwest National Laboratory
PV	Photovoltaic
RE	Renewable Energy
TMY	Typical Meteorological Year

Contents

Overview ii
Acknowledgments iii
Acronyms and Abbreviations iv
1.0 Tool Methodology and Assumptions 1
1.1 MCOR Methodology 1
1.1.1 Generation of Renewable Energy Profile Scenarios 1
1.1.2 Calculation of Renewable Power Generation 2
1.1.3 Sizing of Renewable Energy and Battery Components 4
1.1.4 Simulation of Microgrid Power Dispatch 7
1.1.5 Back-up Generator Sizing 8
1.1.6 Post-processing and System Ranking 9
1.2 Model Assumptions 13
1.2.1 Solar Modeling Assumptions 13
1.2.2 Tidal Modeling Assumptions 13
1.2.3 Cost Assumptions 14
2.0 Quick Start Guide 16
2.1 Setup 16
2.2 Running the Tool 16
3.0 Advanced Options 19
3.1 Specifying Specific PV, Tidal, and Battery Sizes 28
3.2 Specifying Specific Generator Sizes 28
3.3 Setting Custom Costs 29
3.4 Setting Net-Metering Restrictions 29
3.5 Using Custom 8760 Power Generation Data 30
3.6 Using Different Load Profiles for Normal and Emergency Operation 30
3.7 Sizing a Battery to Avoid Exporting Excess PV to the Grid 30
3.8 Running Sensitivity Studies 30
3.9 Modifying the Dispatch Plots and Generating Custom Plots 32
3.9.1 Dispatch Plot Options 33
3.9.2 System Comparison Plots 33
3.9.3 Custom Plots and Analysis 34
4.0 Troubleshooting 36
4.1 Validation Errors 36
4.2 Solar Profile Generation Errors 36
5.0 References 37

Figures

Figure 1. Overview of MCOR simulation structure for a simulation with PV, batteries, and a diesel generator. Each of the microgrid system configurations, defined by a MicrogridSystem Python object, is simulated for each of the solar outage profiles, within a PVBattGenSimulator Python object. The overall simulation is coordinated by an Optimizer Python object. 8
Figure 2. Sample dispatch figure, showing how the load is met by different microgrid components for the outage periods with the highest (left) and lowest (right) solar insolation. The load is shown in blue, PV generation in orange, battery charging and discharging in green, and diesel generator consumption in red. 11
Figure 3. Sample comparison figure, showing how mean fuel consumption varies across systems with different PV capacity and battery power ratings. Darker green squares denote lower fuel consumption (in gallons). 12
Figure 4. Sample comparison figure, showing how mean fuel consumption and capital cost vary across systems with different PV capacity and battery power ratings. Circle color denotes PV capacity in kW with darker green representing larger capacities and circle size denotes battery power in kW with larger size representing higher power ratings. 12
Figure 5. Sample TMY PV power profile showing the power generated by a 1 kW array under typical conditions. 17
Figure 6. Sample max and min power profiles showing the power generated by a 1 kW array for the solar profiles with the maximum (left) and minimum (right) total production. 18
Figure 7. Sensitivity study output graphs showing how the storage recovery percent varies across simulations with different outage lengths. These were generated with the following function calls:
(a) plot_comparison_graphs(output_dict, sensitivity_param, 'storage_recovery_percent', 'least_fuel', 'avg'),
(b) plot_comparison_graphs(output_dict, sensitivity_param, 'storage_recovery_percent', 'least_fuel', 'distribution'),
(c) plot_comparison_graphs(output_dict, sensitivity_param, 'storage_recovery_percent', 'all', 'distribution'). 32

Tables

Table 1. PV system costs for different racking and tracking types and system sizes. 14
Table 2. System operation and maintenance costs. 15
Table 3. MCOR input parameters. 19

1.0 Tool Methodology and Assumptions

1.1 MCOR Methodology

The goal of the MCOR tool is to provide several viable microgrid sizing options that can meet the resilience goals of a site and maximize economic benefits. This means meeting a set of critical loads for a specified period of time without any power supply from the electrical grid. The tool simulates power dispatch during an outage for many different system configurations and ranks the systems according to the goals of the site.

A MCOR simulation is comprised of several different stages:

1. Generation of renewable energy resource profile scenarios
2. Calculation of renewable energy power generation
3. Sizing of renewable energy and battery components
4. Simulation of microgrid power dispatch
5. Back-up generator sizing
6. Post-processing and system ranking

1.1.1 Generation of Renewable Energy Profile Scenarios

MCOR currently models two types of renewable energy resources, PV and tidal energy, to simulate microgrid power dispatch under a large range of potential renewable energy resource conditions. Each resource is modeled separately to develop resource profiles for a series of simulated outages. Power generation is then calculated for each outage profile, before combining these power profiles to simulate microgrid dispatch during the outages.

1.1.1.1 Solar Profiles

MCOR uses a statistical model, the Alternative Solar Profiles (ASP) model (Newman et al. 2020), to generate a set of solar (GHI, DNI, and cloud cover) and temperature profiles using historical modeled solar data from NREL's NSRDB[1] (Sengupta et al. 2018). These profiles begin at random times during the year, are of a specified length, and, assuming enough profiles are generated, allow for a large range of solar conditions to be simulated. The first hour of each profile is randomly selected from an hour in the historical dataset. Subsequent hours are then modeled based on the historical likelihood of solar conditions changing from that starting state (GHI, DNI, cloud cover) to a different state.

MCOR simulates each system configuration for each of the solar and temperature profiles, or "scenarios". By using a statistical representation of solar resources under hundreds of credible scenarios (rather than a single typical data set), this tool quantifies the confidence that the specified resources within the microgrid have sufficient power and energy capacity to meet the site's resilience goals under all conditions. For example, not only does the starting time of the outage (time of day and time of year) impact the ability of a microgrid to meet resilience goals, but also the weather conditions during the outage period, which can be highly variable. This allows a user to be confident that the tool's results encompass any "worst-case scenario"

[1] https://nsrdb.nrel.gov/

weather situations, helping them to make better informed decisions on equipment sizing and providing a clearer understanding of risk. For example, a worst-case scenario may be an outage where there is simultaneously low renewable energy generation and high load. MCOR defaults to simulating two hundred random outages that last for the user-specified duration. Two hundred outages were found to be sufficient to reach convergence for the generator size to meet all load during the worst-case outage condition (Newman et al. 2020).

1.1.1.2 Tidal Profiles

MCOR uses a similar approach to modeling tidal resources and simulates hundreds of potential outage scenarios which are randomly selected. However, as tidal resources are more predictable than solar, instead of creating a statistical model based on historical resources, MCOR instead samples outage periods from a 19-year tidal epoch model for the given site. The individual scenario start date and time are chosen to match those of the solar profiles if both solar and tidal resources are included in the microgrid model.

MCOR requires hourly tidal velocity data at a range of water column depths for a full year as an input to this model. An example analysis was performed for the Salish Sea with a tidal turbine location at Port Angeles, WA, and this example data is included in the MCOR repository. The data for this model came from Yang et al. (2021). Data for other sites could be obtained from the Marine Energy Atlas (MEA)[1]. Once this data is obtained, MCOR extrapolates from one year to the 19-year tidal epoch using the UTide[2] Python package. Then tidal resource scenarios are created by selecting the equivalent time periods from the tidal epoch as were used for the solar profiles.

1.1.2 Calculation of Renewable Power Generation

1.1.2.1 PV Power Generation

For each of the solar scenarios, MCOR calculates the AC power produced from a 1kW array using the pvlib-python[3] library of functions (Holmgren et al. 2018). Pvlib-python is a publicly available and well-documented library developed at Sandia National Laboratories as part of the PV Performance Modeling Collaborative[4] (Stein 2012). The most recent pvlib-python version tested with MCOR is '0.9.4a2.dev3+geefc35c'. While this library allows for almost infinite customization, we make several simplifying assumptions (such as panel and inverter models and DC/AC ratio) that can be modified for sites that have more detailed information on existing or potential PV systems. Panel and inverter performance details are based on published test data from the California Energy Commission[5]. The following reference models are used in the power calculation:

- Solar position calculation: Reda and Andreas 2004; Reda and Andreas 2007
- Extraterrestrial radiation: Reno, Hansen and Stein 2012; Partridge and Platt 1976; Duffie and Beckman 1991; ASCE 2005
- Single-axis tracking: Anderson and Mikofski 2020

[1] https://maps.nrel.gov/marine-energy-atlas/
[2] https://pypi.org/project/UTide/
[3] https://github.com/pvlib/pvlib-python
[4] https://pvpmc.sandia.gov/
[5] https://solarequipment.energy.ca.gov/Home/Index

- Diffuse Irradiation on a tilted surface and from the ground: Loutzenhiser et al. 2007; Hay & Davies 1980
- Cell and module temperature: King et al. 2004
- DC power calculation (using the single diode model): De Soto et al. 2006; Dobos 2012; Madelung 2004; Wenham, Green & Watt 1995; Jain & Kapoor 2004; King et al. 2004
- PV losses: Dobos 2014
- AC power calculation: King et al. 2007

Once the AC power output is calculated for each solar scenario for a 1 kW array, the power is scaled for each system configuration (as described below) according to the total PV capacity.

1.1.2.2 Tidal Power Generation

Once tidal velocity profiles have been created as described above, the power generated by the tidal resource is calculated as follows according to O'Doherty et al. 2018:

$$P = \begin{cases} \min\begin{Bmatrix} 0.5 C_{\mathrm{p}} \rho u^3 A_{\mathrm{rotor}} n_{\mathrm{rotor}} \\ P_{\mathrm{rated}} \end{Bmatrix}, & u_{\text{cut-in}} < u < u_{\text{cut-out}} \\ 0, & u < u_{\text{cut-in}} \\ 0, & u > u_{\text{cut-out}} \end{cases} \tag{1}$$

where
C_{p} = tidal generator power coefficient
ρ = fluid density
u = tidal velocity
A_{rotor} = area of the rotor
n_{rotor} = number of rotors per turbine
P_{rated} = turbine rated power
$u_{\text{cut-in}}$ = cut-in velocity
$u_{\text{cut-out}}$ = cut-out velocity

Several parameters need to be determined to complete this calculation. These include:

- Turbine rated power
- Number of rotors
- Rotor diameter
- Cut-in and cut-out velocities
- Depth of turbine

The first three are determined by the turbine model. MCOR provides several example tidal turbine models in the *MCOR_Prices.xlsx* spreadsheet, and turbine rated power, number of rotors, and rotor diameter are taken from the selected model. The cut-in and cut-out velocities can also be model-dependent, however, MCOR uses defaults for these, which can be overridden by the user. Finally, the turbine depth must be selected by the user and is very site dependent.

Note that the calculation used by MCOR assumes power is generated in both directions of tidal flow, as would be consistent with a cross-flow turbine. For an axial flow turbine with more directionality, more alignment with the maximum current direction would be necessary to calculate power output.

1.1.3 Sizing of Renewable Energy and Battery Components

Before running the system dispatch simulations, a set of renewable resource and battery system sizes is determined based on the annual load profile for the site and renewable power production. There are a few different options for determining the capacities of the microgrid components, depending on the resources that the user opts to include as well as a few tunable parameters. These tunable parameters are:

- The choice to size resources based on TMY (typical meteorological year) data or the aggregated outage profiles described above
- The choice to factor in battery losses when sizing resources

In addition to the sizes chosen by MCOR, the user can choose to simulate other specific system sizes, so this set of configurations is intended as a starting point for users who do not already have specific component sizes determined. The methodology for determining these sizes is described below.

1.1.3.1 Tidal

The methodology for sizing tidal resources is the same with or without additional PV resources. MCOR supplies a list of tidal generators that the user can choose from based on the specifications of their location. They can also add information for a specific generator not on the included list.

Given the tidal generator model chosen by the user, MCOR will start with one tidal turbine and continue adding systems with more turbines until the total energy generated by the tidal turbines over the course of a year exceeds the total annual load of the site or until the maximum number of turbines (as specified by the user) is reached, whichever comes first.

For example, for a site with a total annual load of 20,000 kWh, and a tidal turbine model that generates 6,000 kWh per year, MCOR will model microgrid configurations with 1, 2, 3, and 4 tidal turbines. If the user indicates that only 3 tidal turbines can be included at the site, then MCOR will model microgrid configurations with 1, 2, and 3 turbines.

1.1.3.2 PV

There are two different methods for sizing PV capacities, depending on if tidal resources are also included in the microgrid.

Microgrids with Only PV

For microgrid systems with only PV resources, four PV system capacities are determined as follows:

- Net-zero plus efficiency losses
- Net-zero

- 1/2 Net-zero
- 1/4 Net-zero

The Net-zero PV capacity is determined as the capacity required to generate the total energy used over the course of the year based on the TMY solar generation profile:

$$NZ = L/PV \tag{2}$$

where NZ = net-zero capacity
L = total annual load
PV = total annual PV generated by a 1 kW system

Half and quarter Net-zero capacities are simply the Net-zero capacity divided by two and four, respectively. The Net-zero plus efficiency losses capacity increases the net-zero capacity to account for efficiency losses from charging and discharging the battery:

$$NZ^{+} = \frac{L + eff}{PV} \tag{3}$$

$$eff = exc \times (1 - RTE) \tag{4}$$

where NZ^{+} = Net-zero plus capacity
eff = efficiency losses from charging the battery
exc = total annual excess PV generation beyond what is used instantaneously to serve load based on the Net-zero capacity
RTE = battery system round trip efficiency

Microgrids with Both PV and Tidal

For microgrid systems with both PV and tidal resources, a different methodology is used. First, the Net-zero size as described above is included with zero tidal turbines. Then for each tidal turbine number, PV resources are added such that the combined total annual generation of both PV and tidal is equal to the total annual load.

Following the example above where the total annual load is 20,000 kWh, and the selected tidal turbine model generates 6,000 kWh per year, with an annual PV generation of 2,000 kWh per 1 kW of installed PV, the following system configurations would be modeled by MCOR:

- 1 tidal turbine, 7 kW PV
- 2 tidal turbines, 4 kW PV
- 3 tidal turbines, 1 kW PV
- 4 tidal turbines, 0 kW PV
- 0 tidal turbines, 10 kW PV

1.1.3.3 Battery

There are three different methods that are used to calculate the battery power and capacity, the “longest night” method, the “no-RE export” method, and the “unmet load” method.

Longest Night Method

Using the “longest night” method, five battery capacities are determined, with the largest battery capacity calculated based on the energy necessary to meet the entire load on the longest night of the year:

$$\text{Max capacity} = \text{Maximum nightly load}/RTE \tag{5}$$

The other sizes are calculated as fractions of this capacity as follows:

- Max capacity
- 3/4 Max capacity
- 1/2 Max capacity
- 1/4 Max capacity
- 0

For each capacity size, the power rating is determined from a specified battery power to energy ratio (either user entered or a default). Each combination of battery and PV sizes is included in the set of system configurations, along with a system with no PV or batteries, as follows:

PV	Battery
Net-zero plus efficiency losses	Max capacity
Net zero	Max capacity
½ net-zero	Max capacity
¼ net-zero	Max capacity
Net-zero plus efficiency losses	¾ Max capacity
Net zero	¾ Max capacity
½ net-zero	¾ Max capacity
¼ net-zero	¾ Max capacity
Net-zero plus efficiency losses	½ Max capacity
Net zero	½ Max capacity
½ net-zero	½ Max capacity
¼ net-zero	½ Max capacity
Net-zero plus efficiency losses	¼ Max capacity
Net zero	¼ Max capacity
½ net-zero	¼ Max capacity
¼ net-zero	¼ Max capacity
Net-zero plus efficiency losses	0
Net zero	0
½ net-zero	0
¼ net-zero	0
0	0

Note that this method is only valid for microgrid configurations with PV only (i.e. no tidal).

No-RE Export Method

The second battery sizing method is called "no-RE export" and determines the battery capacity and power required to store all excess RE generation (beyond that used instantaneously to serve load) for each of the included RE capacities (either PV only or PV + tidal). In addition to the battery size that captures 100% of excess generation, MCOR will also include battery sizes that capture 75% and 50% of excess generation. In the example with PV only, thirteen system configurations are generated, three for each of the matched PV and battery sizes and one with no PV or batteries.

Unmet Load Method

The third battery sizing method is called "unmet load" and determines the battery capacity and power required to discharge the battery to meet all load not met by renewable resources (assuming the battery is fully charged). This method is only valid for microgrid configurations with both PV and tidal (or tidal only). As for the "no-RE export" method, one battery size is determined for each RE configuration. For each RE size, the battery capacity is calculated as the maximum daily load not met by the RE resources and the battery power is calculated as the maximum excess hourly RE generation. In addition to the battery size that captures 100% of either unmet load, MCOR will also include battery sizes that capture 75% and 50% of unmet load.

1.1.4 Simulation of Microgrid Power Dispatch

To understand microgrid operation during an unplanned power outage under varying resource conditions, each of the system configurations (RE and battery size) are simulated using each of the solar and/or tidal generation scenarios (see Figure 1). Each RE generation time series consists of hourly values for AC generation for 1 kW of installed solar or tidal energy, which are then scaled by the RE capacity for each microgrid configuration. The load profile for each simulation period is taken from the same dates and times as the corresponding RE generation profile and consists of hourly demand in kW. The user uploaded demand profile can reflect total annual load or only the critical load portion. See Section 3.6 for information on how to run a simulation with both typical and critical load profiles.

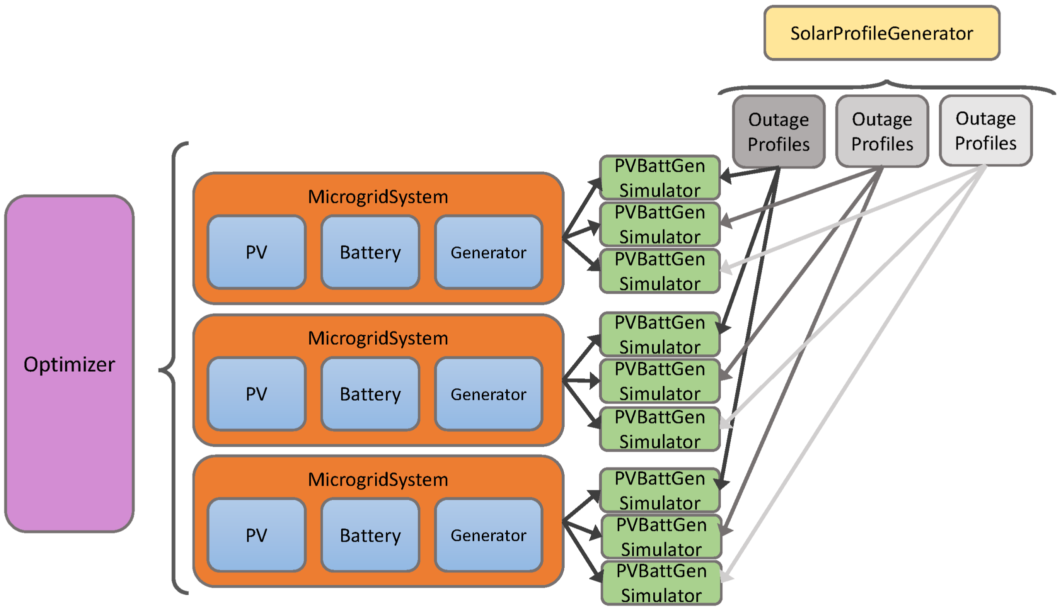

Figure 1. Overview of MCOR simulation structure for a simulation with PV, batteries, and a diesel generator. Each of the microgrid system configurations, defined by a MicrogridSystem Python object, is simulated for each of the solar outage profiles, within a PVBattGenSimulator Python object. The overall simulation is coordinated by an Optimizer Python object.

For each individual simulation, MCOR uses a fixed dispatch strategy, as follows:

- PV and/or MRE are dispatched to meet the load. Any excess generation is used to charge the battery.
- The battery is dispatched. During the day, the battery will discharge at up to its power rating to meet the load not served by RE (and constrained by remaining capacity), and during the nighttime, there are three battery dispatch strategies:
  - Discharge the battery at a constant rate based on the available capacity at the beginning of the night and the length of the night to use all the available capacity by the end of the night.
  - As above but update the discharge rate at each hour based on the remaining available capacity. This is the default strategy.
  - Discharge the battery at up to its power rating to meet the load not served by RE. This is the recommended method for microgrid configurations with included tidal generation.
- The generator is dispatched to meet the remaining unmet load.

1.1.5 Back-up Generator Sizing

For each system/scenario combination, the generator is sized as a simulation post-processing step. The generator capacity is determined such that the electrical load is met at each hour of the simulation. This capacity is then increased according to a specified percentage buffer and is chosen from a list of default diesel generators (see Section 1.2.3). While the chosen generator for each simulation is based on meeting all critical load, information on less conservative, but also less expensive configurations can be found in the tool outputs. As with the generator sizing, fuel consumption is calculated as a post-processing step for each simulation (system/scenario) combination. Based on the selected generator (which is chosen to meet all unmet load from the RE and battery systems), fuel consumption is calculated based on a stepped linear model. The required fuel storage tank capacity for the duration of the specified outage is also calculated and based on a list of default tank sizes.

1.1.6 Post-processing and System Ranking

Once each simulation has been run, the results are aggregated for each system configuration across all outage scenarios. Metrics are either calculated as the average (mean) value across all simulations, or the most-conservative (max) value across all simulations, allowing for design of a more resilient system. Metrics are returned to the user in an Excel spreadsheet, with each row representing a single system configuration (combination of PV, tidal and battery size). System configurations are ordered and filtered by the user-specified ranking and filtering criteria, respectively. Additional Excel sheets include the user-specified annual load profile, the calculated TMY solar generation profile (if solar is included), the calculated TMY tidal generation profile (if tidal is included), the user-specified input parameter values, and the default parameter values and assumptions. Although all dispatch simulations are run under outage conditions, a few of the metrics below are calculated for grid-connected mode as a post-processing step using the TMY generation data (e.g. annual RE net-metering revenue, annual RE demand savings, and simple payback time).

The output metrics are as follows:

- PV capacity, tidal capacity and battery power/capacity for the specified system
- Generator power capacity for the specified system, both typical (the average across all simulations) and most-conservative (the largest generator size across all simulations)
- Total fuel tank capacity (based on the largest amount of fuel needed across all simulations)
- Required area for the PV system. This is based on the panel size multiplied by a specified spacing buffer
- Total system capital cost. This does not include the cost of existing systems.
- PV capital
- Tidal capital
- Battery capital, including inverter costs.
- Generator capital, based on the most conservative size, and including installation labor
- Fuel tank cost, based on the most conservative size
- Annual PV O&M costs
- Annual tidal O&M costs
- Annual battery O&M costs
- Annual generator O&M costs
- Annual RE net-metering revenue, which can be calculated according to several different net-metering set-ups. These are discussed in more detail in Section 3.0 (Advanced Options)

- Annual RE demand savings. This will be $0 unless a demand rate was included in the simulation inputs
- Simple payback time, based on capital costs, O&M costs, and the annual RE net-metering and demand savings revenue
- Mean PV hourly load across all simulations
- Mean peak PV hourly load across all simulations
- Max peak PV hourly load across all simulations
- Mean tidal hourly load across all simulations
- Mean peak tidal hourly load across all simulations
- Max peak tidal hourly load across all simulations
- Mean generator hourly load across all simulations
- Mean peak generator hourly load across all simulations
- Max peak generator hourly load across all simulations
- Mean total generator load across all simulations
- Max total generator load across all simulations
- Mean total outage load across all simulations
- Max total outage load across all simulations
- Mean hours before generator needed across all simulations
- Min hours before generator needed across all simulations
- Mean hourly battery load across all simulations
- Mean peak hourly battery load across all simulations
- Max peak hourly battery load across all simulations
- PV/Tidal/Battery/Generator percent, the average percentage of energy met by the PV system, tidal system, batteries, and generator, respectively
- Fuel consumption for the resilience period, both typical (the average across all simulations) and most-conservative (the largest amount of fuel consumed across all simulations)
- Storage recovery percentage, the percentage of excess RE which is charged into the battery

Beyond output metrics, the tool can produce several different types of graphs:

Dispatch graph. This shows how the individual system components are dispatched at each hour to meet the electrical load during an outage. While this can be displayed for any individual system configuration (PV, tidal and battery size combination), by default, this graph is generated for three systems: (1) The ‘best’ system according to the user-specified ranking criteria (simple payback by default), (2) The ‘best’ system that also includes a battery, since the potential economic benefits of batteries are not included in the payback calculation, and (3) the system that uses the least fuel overall. For each system, the dispatch graph can be plotted for either a specific outage scenario, the scenarios with the highest and lowest solar insolation, or the scenarios with the largest and smallest fuel consumption. This figure can be shown as a line graph or a stacked area graph. An example is shown in Figure 2.

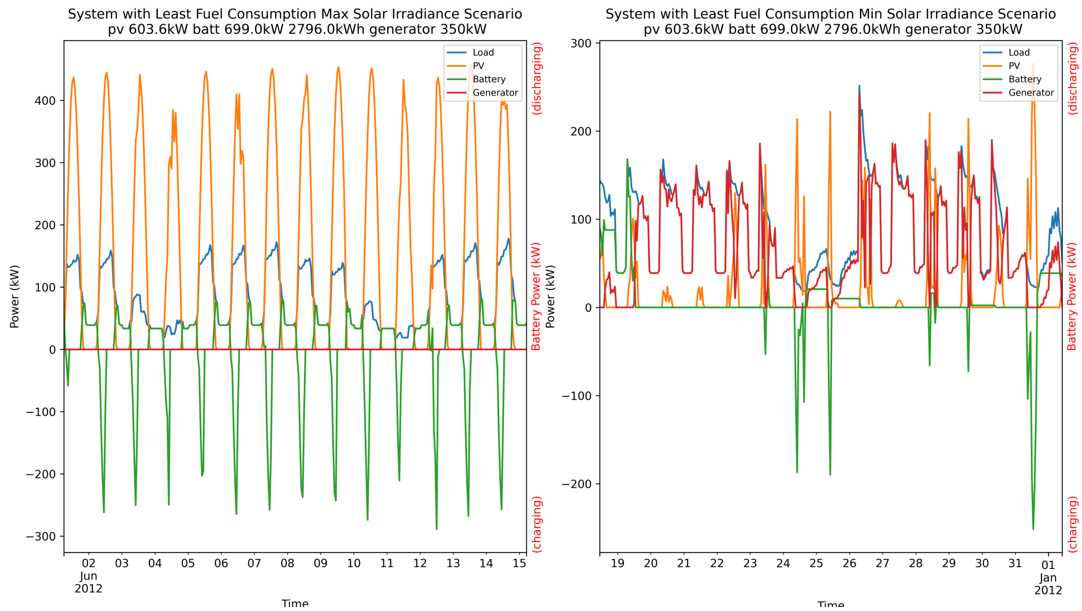

Figure 2. Sample dispatch figure, showing how the load is met by different microgrid components for the outage periods with the highest (left) and lowest (right) solar insolation. The load is shown in blue, PV generation in orange, battery charging and discharging in green, and diesel generator consumption in red.

System comparison graphs. Users can also compare system configurations against each other graphically by selecting either one output metric or two and showing those for all systems. Figures 3 and 4 show examples where one or two metrics are compared, respectively. Note: these two figures will only work for systems with PV only (i.e. no tidal).

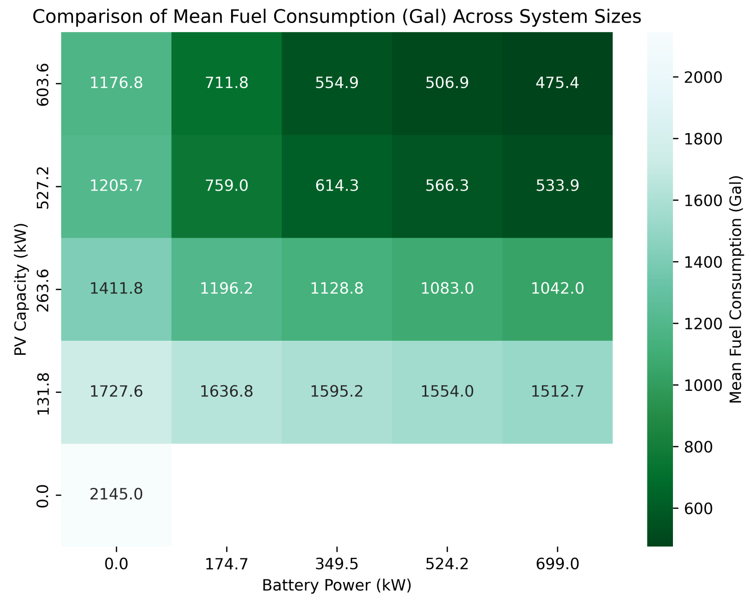

Figure 3. Sample comparison figure, showing how mean fuel consumption varies across systems with different PV capacity and battery power ratings. Darker green squares denote lower fuel consumption (in gallons).

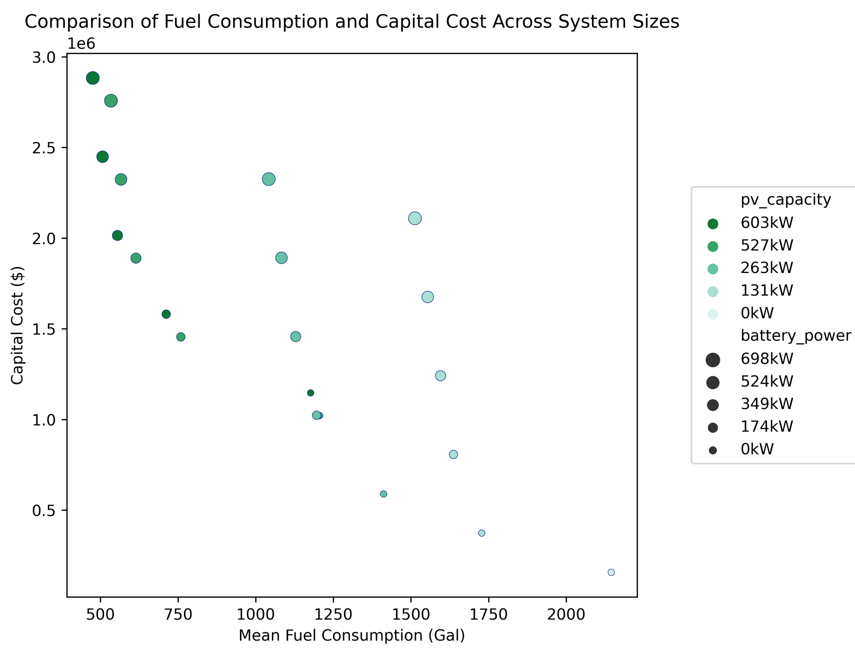

Figure 4. Sample comparison figure, showing how mean fuel consumption and capital cost vary across systems with different PV capacity and battery power ratings. Circle color denotes PV capacity in kW with darker green representing larger capacities and circle size denotes battery power in kW with larger size representing higher power ratings.

1.2 Model Assumptions

As MCOR is intended to provide a high-level recommendation of system sizing, we make several simplifying assumptions:

- MCOR considers load and generation within a single electrical node (bucket of energy analysis), with no constraints in the distribution system that would prevent any generation source from delivering energy to any node
- It does not calculate distribution losses
- It assumes sufficient fuel and fuel storage is available to power all diesel generators
- It does not model sub-hourly variation
- It does not consider transient effects such as equipment switching on/off
- It does not account for changes in load pattern before and after daylight savings time
- The net metering and demand revenue and payback listed in the output spreadsheet are based only on operation of the RE system during the year and do not include any contribution from demand response or other battery operation strategies

1.2.1 Solar Modeling Assumptions

Most of the solar modeling assumptions (besides user-defined inputs) are hardcoded in the solar profile generation modules or are contained in the pvlib defaults. For more on the latter, you can go to the pvlib-python documentation page[1]. The MCOR-defined solar modeling assumptions are described below:

- Default solar module: SunPower SPR-X22-360-C-AC, selected by choosing a newer module similar to the top U.S. installed panel according to NREL's OpenPV database in 2018 (now hosted at LBNL's Tracking the Sun website[2]).
- Default inverter: SMA America SB5000US-240V-240V, selected as a reasonable size (5 kW) from a top U.S. inverter manufacturer (SMA - manufactures U.S. inverters in Colorado)
- Default modules per string: 7, with two strings in parallel, corresponding to a total of 5.04 kW, with a voltage of 490 V for a 5 kW inverter
- Albedo: 0.12
- PV losses: standard losses used by PVWatts (14%) (Dobos 2014)
- For single-axis tracking, a description of the functions used can be found here: https://github.com/pvlib/pvlib-python/blob/main/docs/tutorials/tracking.ipynb

1.2.2 Tidal Modeling Assumptions

Several default assumptions are used for the tidal generator model and can be modified in the *main_example.py* module when running an analysis. These include:

- Maximum power coefficient (C_p): 0.42

---

1 https://pvlib-python.readthedocs.io/en/stable/
2 https://emp.lbl.gov/tracking-the-sun

- Tidal cut-in velocity: 0.5 m/s
- Tidal cut-out velocity: 3 m/s
- Tidal inverter efficiency: 0.9
- Tidal turbine losses: 10%

1.2.3 Cost Assumptions

Most of the costs used in MCOR are either user-specified or contained in an Excel document that can be edited by the user. The electricity rate, demand rate, and the net-metering are all included as user-defined inputs. If the electricity rate is not specified, the most recent average retail price from the U.S. Energy Information Administration is used for the relevant U.S. state[1]. If no demand rate is specified, it is assumed that there is not a separate demand charge, and revenue from demand charge savings is not calculated for the simulation. If no net-metering rate is specified, the net-metering rate is assumed to be the same as the electricity rate.

PV capital costs are determined as follows. Ground-mounted (fixed and single-axis) and roof-mounted costs are based on the NREL report U.S. Solar Photovoltaic System and Energy Storage Cost Benchmarks: Q1 2021 (Ramasamy et al. 2021), which lists capital costs for PV systems of various sizes (residential, commercial, and utility-scale). Costs are only listed for ground-mounted fixed commercial and utility-scale, ground-mounted single-axis utility scale, and roof-mounted residential and commercial scales. From these costs, adders for different racking types were extrapolated as follows: roof to ground cost adder: +5%, fixed to single-axis tracking adder: +7%. Carport adders were determined from the Vermont Solar Cost Study (Seddon 2016) to be +19% as compared with roof-mounted. For utility-scale roof and carport-mounted, the utility-scale ground-mounted costs were used, because utility-scale roof or carport mounts are not feasible. In Table 1, the costs that came directly from the NREL report are highlighted in yellow, while the other costs were extrapolated using the adders as described above.

Table 1. PV system costs for different racking and tracking types and system sizes.

Size Limit (kW)	0-100kW	100-5000kW	5-100MW
Racking, tracking types	($/W)	($/W)	($/W)
ground, fixed	2.78	1.64	0.83
ground, single_axis	2.97	1.75	0.89
roof, fixed	2.65	1.56	0.83
carport, fixed	3.15	1.86	0.83

Two types of costs are included for tidal turbine generators. Costs for specific models are derived from Coles et al. 2021 and costs for reference model turbines RM1, RM2 and RM4 are derived from Neary et al. 2014a, Neary et al. 2014b, and Neary et al. 2014c, respectively. All costs are included in the *MCOR_Prices.xlsx* spreadsheet in the ‘mre_costs’ tab.

[1] https://www.eia.gov/electricity/state/

Battery costs are calculated assuming $0.521/Wh for the battery system and construction/commissioning and $0.401/W for inverter and balance of system costs. These are based on the average of 2018 and projected 2025 costs from the report “Energy Storage Technology and Cost Characterization” (Mongird et al. 2019).

The cost for generators is based on CAT model costs from the GSA advantage website[1] and then extrapolated to $191/kW. This is then doubled to $382/kW with the assumption that total capital cost is twice the cost listed to account for installation costs. The cost for any given system is based on the largest generator size from all of the outage scenarios.

The fuel tank sizes and costs are based on ConVault tank systems with prices from the GSA Advantage website[1].

Operations and maintenance costs are taken from a variety of sources. Generator costs are from “The Market and Technical Potential for Combined Heat and Power in the Industrial Sector (ONSITE SYCOM Energy Corporation 2000), battery costs are from “New Energy Outlook 2015: Long-Term Projections of the Global Energy Sector” (BNEF 2015), PV costs are from “The Role of Advancements in Solar Photovoltaic Efficiency, Reliability, and Costs” (Woodhouse et al. 2016), and tidal turbine costs are from Previsic and Bedard 2009. The generator costs provided in the cited resource include fixed and per kWh costs for three different generator sizes. A blended $/kW rate has been calculated by prorating the costs for inflation from 2000 to 2019, assuming an annual usage of 336 hours, and extrapolating the three data points to an exponential function (Annual cost = scalar x capacity^exponent). Costs for fuel consumption are not included in the MCOR model, as it is assumed that the generator is only operated during emergency outage conditions. The operations and maintenance cost assumptions are shown in Table 2. All operations and maintenance costs do not include replacement costs.

Table 2. System operation and maintenance costs.

Component	Generator scalar	Generator exp	Battery	PV ground, fixed	PV ground, single_axis	PV roof, fixed	PV carport, fixed	Tidal
Unit	$/kW-yr		$/kWh-yr	$/kW-yr	$/kW-yr	$/kW-yr	$/kW-yr	$/kW-yr
Cost	102.65	0.669	2.41	15	18	17.5	12.5	510

[1] https://www.gsaadvantage.gov

2.0 Quick Start Guide

2.1 Setup

A Python 3.10 installation is required to run MCOR. It is also highly recommended to run MCOR using an IDE, such as VSCode or PyCharm. To install Python as well as many commonly used packages, you can download the Anaconda distribution (https://www.anaconda.com/products/distribution#download-section). You can clone the MCOR repository from Github here: https://github.com/pnnl/MCOR. Once cloned, ensure that the "mcor" directory is in your Python path.

Required packages and versions are included in the requirements.txt file. You can install all of these by running:

```
pip install -r requirements.txt
```

You can optionally install numba to speed up the solar power calculation: http://numba.pydata.org/#installing.

You will also need to create your own credentials file with an NREL API key to download historical solar data from the NSRDB. To do so:

1. Sign up for an API key here: https://developer.nrel.gov/signup/
2. Create a file called *creds.yaml* in the root directory and then add your email address and API key to the file.

   The file should have the following format:

   ```
   nrel_api_key: <your key>
   nrel_api_email: <your email>
   ```

2.2 Running the Tool

To run an MCOR simulation, you must edit and then run an input Python file with the parameters for your site. There is a sample input file called *main_example.py* in the *main_files/* directory. It is recommended that you copy this file for each new site, so that you can still refer to the sample main file and do not encounter any merge conflicts when pulling updates from Github.

You can update input parameters in the second half of the *main_example.py* file (or your copy) after the "Define simulation parameters here" text. All the input parameters are stored in the input_dict dictionary, nested in sub-dictionaries by category. Many of the parameters defined in *main_example.py* are for more advanced use of MCOR, but at a minimum, you should specify the following parameter values:

- System inputs:
  - latitude
  - longitude

  - timezone
  - length_trials (the number of hours in your simulated outage period)
  - renewable_resources (which RE resources to include in the microgrid, options include ‘pv’ and/or ‘mre’)
- Load inputs:
  - annual_load_profile (an 8760 annual load profile for the site. Make sure to update the filename and column name which includes the load data)
- Financial inputs:
  - utility_rate (blended electricity rate)
- Output inputs:
  - save_filename (the name to be used for output results files. If you don’t change this for different runs, your results will be overwritten.)

Note that the code required to download historical solar data and generate the solar profiles can take a long time to run (up to 10 minutes, depending on the number of profiles generated and CPUs on your computer). Fortunately, after the ASP code is run, the scenario data are saved to csv files, so if you are not running the code for a new site, you should set “get_solar_data” to False and “get_solar_profiles” to False, under “pv_inputs”. Likewise, if you are running a new site, make sure those parameters are set to True. While developing the tidal model is much faster, you can also set “get_tidal_profiles” under “mre_inputs” to False after running MCOR for a given site the first time.

After MCOR is finished running through all the simulations, it will print out a list of the system configurations to the Python Console and generate dispatch plots for the best systems as described in Section 1.1.6.

It will also produce a few plots to show how the PV system is performing during normal operation using the TMY solar data (Figures 5 and 6). These can be used to 'gut-check' that the PV profiles look reasonable for the given location.

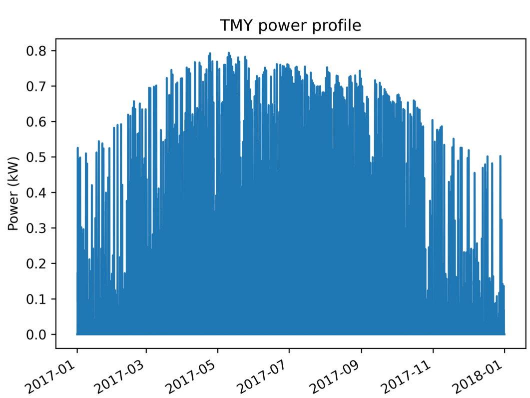

Figure 5. Sample TMY PV power profile showing the power generated by a 1 kW array under typical conditions.

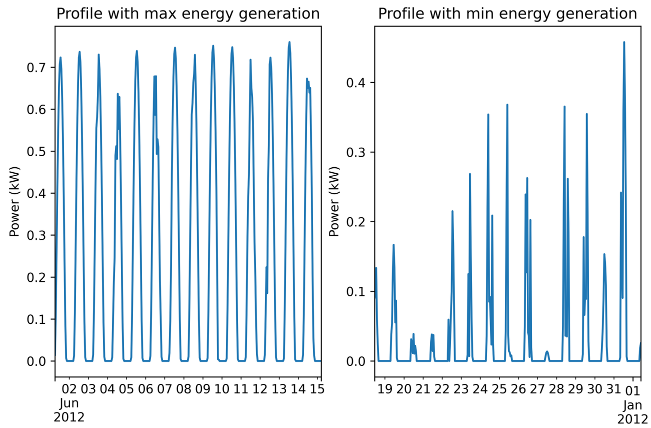

Figure 6. Sample max and min power profiles showing the power generated by a 1 kW array for the solar profiles with the maximum (left) and minimum (right) total production.

Finally, to view the quantitative results from a simulation (system sizes, fuel consumption, cost, etc.), migrate to the directory *mcor/output* and open the Excel file with the filename you specified in *main_example.py*. The contents of this file are described in Section 1.1.6. MCOR will also create a binary pickle file and json file with the full results of the simulation. The binary file can be saved for reference and loaded at any point to retrieve or re-plot your simulation results.

3.0 Advanced Options

Beyond the parameters specified in Section 2.0, MCOR allows you to define more advanced inputs to override many of the defaults. A full list of all the MCOR input parameters is shown in Table 3.

Table 3. MCOR input parameters.

Parameter	Default	Units	Example	Data Type	Description
system_inputs					
latitude	N/A	degrees	46.34	Float	
longitude	N/A	degrees	-119.28	Float	
timezone	N/A	None	US/Alaska, US/Aleutian, US/Arizona, US/Central, US/East-Indiana, US/Eastern, US/Hawaii, US/Indiana-Starke, US/Michigan, US/Mountain, US/Pacific, US/Pacific-New, US/Samoa (or any other valid pytz timezone name)	String	
altitude	0	Meters	10	Float	
num_trials	200	None	200	Int	This is the number of distinct outage periods that are simulated for each system. For testing purposes, 10 is sufficient, but for modeling actual sites, choose at least 200.

Parameter	Default	Units	Example	Data Type	Description
length_trials	336	Hours	336	Int	The number of hours in each outage period, or the resilience goal.
renewable_resources	[‘mre’, ‘pv’]	None	[‘pv’], [‘pv’, ‘mre’], [‘mre’]	List of Strings	Indicates which RE resources should be modeled in the microgrid. Options include photovoltaics (‘pv’) and marine renewable energy (‘mre’).
dispatch_strategy	available_capacity	None	available_capacity, night_const_batt, night_dynamic_batt	String	The battery dispatch strategy. ‘available_capacity’ always discharges the battery up to the available capacity, 'night_const_batt’ discharges the battery up to the available capacity during the day and at a constant rate at night, ‘night_dynamic_batt’ discharges the battery up to the available capacity during the day and at a dynamic rate at night to account for the remaining capacity at any given hour.
size_re_resources_based_on_tmy	False	None	True, False	Boolean	Determines if renewable resource capacities are determined using TMY data or outage profile data.

Parameter	Default	Units	Example	Data Type	Description
size_battery_based_on_tmy	True	None	True, False	Boolean	Determines if the battery capacity is determined using TMY data or outage profile data.
size_resources_with_battery_eff_term	True	None	True, False	Boolean	Determines if battery and inverter efficiency should be considered when sizing RE resources.
start_datetimes	None	None	[dt.strptime('11/4/2022 22:00', '%m/%d/%Y %H:%M')]	List of Datetime objects	Start datetimes for modeling outages. This should only be used if you want to look at specific dates for outages rather than having MCOR randomly generate them.
re_constraints	{}	None	{'total': 2000, 'pv': 2000, 'mre': 2000}	Dictionary	Capacity limits for PV, MRE and/or total RE.
			pv_inputs		
tilt	20	Degrees	20	Float	Tilt of the PV system
azimuth	180	Degrees	180	Float	Azimuth of PV system
spacing_buffer	2	None	2	Float	Factor by which to scale panel area to get total PV array area.
pv_racking	ground	None	ground, roof, carport	String	Type of PV racking
pv_tracking	fixed	None	fixed, single_axis	String	Type of PV tracking
solar_data_source	nsrdb	None	nsrdb, himawari	String	Source for modeled solar data. Choose ‘nsrdb’ for sites in the U.S., Central America, Canada, and South

Parameter	Default	Units	Example	Data Type	Description
					America. Choose ‘himawari’ for sites in East Asia and the Pacific Islands. Coverage information for these datasets can be found at https://nsrdb.nrel.gov/data-viewer.
solar_data_start_year	1998	None	1998	Int	Must be between 1998 and 2022 for nsrdb and between 2016 and 2020 for himawari.
solar_data_end_year	2022	None	2022	Int	Must be between 1998 and 2022 for nsrdb and between 2016 and 2020 for himawari.
get_solar_data	True	None	True, False	Boolean	Determines whether to download NSRDB data. This is only required once for a given lat/lon.
get_solar_profiles	True	None	True, False	Boolean	Determines whether to generate solar outage profiles. This is only required once for a given lat/lon and outage period length.
			mre_inputs		
mre_device_name	‘RM1’	None	‘RM1’	String	Name of chosen tidal device. The full list can be found in the MCOR_prices.xlsx file under the ‘mre_costs’ sheet. Note: this parameter is specified

Parameter	Default	Units	Example	Data Type	Description
					outside of the mre_inputs dictionary.
marine_data_filename	'PortAngeles_2015_alldepths.csv'	None	'PortAngeles_2015_alldepths.csv'	String	Filename for tidal velocity input data. Note: this file needs to be in the MCOR tidal_data/tidal_current directory.
generator_type	'tidal'	None	'tidal', 'wave'	String	Type of MRE generator to model.
depth	10	Meters	10	Float	Depth of tidal turbine in the water column
maximum_cp	0.42	None	0.42	Float	Tidal generator power coefficient
tidal_cut_in_velocity	0.5	m/s	0.5	Float	Tidal velocity at which the turbine begins generating power
tidal_cut_out_velocity	3	m/s	3	Float	Tidal velocity at which the turbine stops generating power
tidal_inverter_efficiency	0.9	None	0.9	Float	Efficiency of tidal generator inverter
tidal_turbine_losses	10	Percent	10	Float	Percent losses from tidal turbine
get_tidal_profiles	True	None	True, False	Boolean	Determines whether to generate tidal outage profiles. This is only required once for a given lat/lon and outage period length.
			battery_inputs		
batter_power_to_energy	0.25	None	0.25	Float	Ratio for determining battery sizes to simulate. This is only used if

Parameter	Default	Units	Example	Data Type	Description
					'batt_sizing_method' is set to 'longest_night'.
initial_soc	1	None	1	Float	Initial battery state of charge
one_way_battery_efficiency	0.9	None	0.9	Float	
one_way_inverter_efficiency	0.95	None	0.95	Float	
soc_upper_limit	1	None	1	Float	
soc_lower_limit	0.2	None	0.2	Float	
batt_sizing_method	longest_night	None	longest_night, no_RE_export, unmet_load	String	How the battery is sized. See Section 1.1.3.3 for a description.
percent_at_night	0.1	None	0.1	Float	The fraction of maximum PV power below which it is considered night, for the purposes of battery discharging
			load_inputs		
annual_load_profile	N/A	kW	N/A	Pandas Series	Hourly load profile, read from csv
off_grid_load_profile	None	kW	N/A	Pandas Series	Used to specify separate load profiles for emergency and normal operation. See description below.
			financial_inputs		
utility_rate	N/A	$/kWh	0.25	Float	Utility rate to use for calculating PV net-metering revenue
demand_rate	None	$/kW	0.1	Float or List	Demand charge rate to use for calculating the reduction in demand

Parameter	Default	Units	Example	Data Type	Description
					charges from installation of PV. Can be either a single number or a list of 12 numbers, one for each month.
system_costs	-	None	N/A	dictionary of Pandas Dataframes	Contains rated power and fuel curves for various generator sizes and costs for each component type. Read from the Excel file: *data/MCOR prices.xlsx*
			multithreading_inputs		
multithreading	False	None	True, False	Boolean	Specifies if Python multithreading should be used to speed up the calculation of solar outage profiles. Note: this will not work on all platforms.
			post_processing_inputs		
filtering_constraints	[ ]	None	capital_cost_usd, pv_area_ft2, annual_benefits_usd, simple_payback_yr, fuel_used_gal mean, fuel_used_gal most-conservative, pv_percent mean. Each element takes the form: {parameter, type, value}, where type can be max or min.	List of dictionaries	Specifies any constraints that should be used to filter the simulated systems.

Parameter	Default	Units	Example	Data Type	Description
ranking_criteria	[{'parameter': 'simple_payback_yr', 'order_type': 'ascending'}]	None	capital_cost_usd, annual_benefits_usd, simple_payback_yr, fuel_used_gal mean, fuel_used_gal most-conservative. Each element takes the form: {parameter, order_type}, where order_type can be ascending or descending	List of dictionaries	Specifies any criteria that should be used to rank the simulated systems.
dispatch_plot_scenario_criteria	'pv'	None	pv, gen, mre	String	Specifies which scenarios are used in the dispatch plots. Choosing ‘pv’ shows the scenarios with the maximum and minimum solar insolation, choosing ‘mre’ shows the scenarios with the maximum and minimum tidal energy generation, and choosing ‘gen’ shows the scenarios with the highest and lowest fuel consumption.
dispatch_plot_scenario_num	None	None	1	Int	Specifies a specific scenario to be used in the dispatch plots, not used if set to None
			sizing_inputs		
existing_components	{ }	None	N/A	Dictionary	Component objects for PV, tidal or battery equipment that is

Parameter	Default	Units	Example	Data Type	Description
					already on-site. See examples in *main_example.py*.
include_pv	( )	kW	(100, 200)	Tuple	List of specific PV sizes to be included.
include_mre	( )	None	(5,)	Tuple	List of specific numbers of turbines to be included.
include_batt	( )	kWh, kW	((1000, 100), (2000, 200))	Tuple	List of specific battery sizes to be included.
			net_metering_inputs		
net_metering_limits	None	None	N/A	Dictionary	Specifies limits on net-metering revenue. See description below.
net_metering_rate	None	$/kWh	0.25	Float	Sets a distinct rate for net-metering revenue. If set to None, the utility rate is used.
			warning_inputs		
suppress_warnings	False	None	True, False	Boolean	Whether or not to suppress calculation warnings from the solar profile generator.
			output_inputs		
save_timeseries_json	True	None	True, False	Boolean	Whether or not to save all timeseries data in a json file.
save_filename	N/A	None	Project1	String	Filename used for output excel and pickle files

3.1 Specifying Specific PV, Tidal, and Battery Sizes

In many cases, it is useful to specify a specific component size. For example, if you are considering a specific roof for installing a PV array, and the size or corresponding capacity has been previously estimated. There are a few ways to accomplish this:

- Use existing_components: This option should only be used if the cost of the specified size should not be included in the capital costs, and if you do not want to see any sizes smaller than the one specified. For example, to specify an existing 100 kW PV array, you could include the following in your *main_example.py* file:

```
pv = PV(existing=True, pv_capacity=100, tilt=tilt, azimuth=azimuth,
   module_capacity=0.360, module_area=3, spacing_buffer=2,
   pv_tracking='fixed', pv_racking='ground')
input_dict['sizing_inputs']['existing_components'] = {'pv': pv}
```

  For an existing 1000 kWh/100 kW battery, you could include the following:

```
batt = Battery(existing=True, power=100, batt_capacity=1000,
   initial_soc=1, one_way_battery_efficiency=0.9,
   one_way_inverter_efficiency=0.95, soc_upper_limit=1,
   soc_lower_limit=0.1)
input_dict['sizing_inputs']['existing_components'] = {'battery':
   batt}
```

- Use include_pv, include_mre or include_batt: This option should be used if you want to add a specific size to the set of configurations being considered. This increases the total number of systems shown in the results Excel file. For example, to include a 400 kW and 500 kW PV array in the list of simulated systems, you could include the following in your *main_example.py* file:

```
input_dict['sizing_inputs']['include_pv'] = (500, 400)
```

  For a 1000 kWh/100 kW battery, you would include the following:

```
input_dict['sizing_inputs']['include_batt'] = ((1000, 100),)
```

3.2 Specifying Specific Generator Sizes

Specifying a generator size is not recommended for a few reasons: (1) Sizing a generator for a resilience need is a central capability within MCOR, (2) It is not guaranteed that a generator designed to work without a microgrid system can be reconfigured to do so, and may require expensive retrofitting, (3) If the specified generator is smaller than what is required to meet all load during resilience periods, MCOR will just assume you have multiple generators of that size. It is not possible to return a generator size that does not meet all the load during outage periods.

Given these caveats, if it is important to get MCOR results for a specific generator size, this can be accomplished by creating an alternate version of the MCOR Prices Excel file in the *mcor/data* directory with only one row in the “generator_costs” sheet containing the generator's information, and then load that file in *main_example.py* instead of *MCOR Prices*.xlsx by changing the filename referenced by input_dict['financial_inputs']['system_costs']. The information required includes:

- Power: generator capacity in kW
- Fuel consumed at ¼, ½, ¾, and full load (in gallons per hour)
- Cost: cost of one generator (including installation)

3.3 Setting Custom Costs

To update any of the default cost assumptions used by MCOR, create a copy of *MCOR Prices.xlsx* and modify the costs in the appropriate sheet. Note that the units and format cannot be changed, so the new prices must be in the same units and format as the ones in the current file. Then load that file in *main_example.py* instead of *MCOR Prices.xlsx*. Prices can also be modified in post-processing, since the costs are all included in the output Excel spreadsheet.

3.4 Setting Net-Metering Restrictions

There are several options for setting limits on the amount of revenue generated from net-metering.

- To specify a net-metering rate different from the utility rate, set the net_metering_rate parameter in units of $/kWh
- To specify no net-metering, set net_metering_rate to 0
- To enforce an installed capacity cap, use the constraints parameter with a pv_capacity limit, e.g. input_dict['filtering_constraints'] = [{'parameter': 'pv_capacity', 'type': 'max', 'value': 100}] with the value in kW
- To enforce an instantaneous export power cap (e.g. if you can only export 100kW to the grid at any given moment), set the net_metering_limits parameter, e.g. input_dict['net_metering_inputs']['net_metering_limits'] = {'type': 'capacity_cap', 'value': 100}, where value is in kW
- To enforce a cap as a percentage of load (e.g. if you cannot end the year with a positive net-metering balance), input_dict['net_metering_inputs']['net_metering_limits'] = {'type': 'percent_of_load', 'value': 100}, where value is in %
- In the case where there is no net-metering, but the battery is sized to capture all excess PV generation (i.e. if batt_sizing_method = 'no_RE_export'), and you want to estimate the revenue from using the battery during normal operation to capture and use this excess generation, set the net_metering_limits parameter, e.g. input_dict['net_metering_inputs']['net_metering_limits'] = {'type': 'no_nm_use_battery'}. This will not include any additional costs due to wearing out the battery more quickly from using it daily and does not capture revenue from using the battery for grid services. It also assumes that the battery is sized with the no_RE_export method.

Note: if there are any specified existing PV components, the revenue generated from those components will not be included in the annual benefits total.

3.5 Using Custom 8760 Power Generation Data

It is possible to have MCOR create outage profiles based on uploaded generation data instead of the Annual Solar Profiles (ASP) algorithm. This could be useful if your site is in a location for which there is no historical NREL or Himawari solar data, if you would like to compare results with known power generation at a site, or if you would like to analyze production for a resource other than PV or tidal. This can be done by setting input_dict['pv_inputs']['get_solar_data'] and input_dict['pv_inputs']['get_solar_profiles'] to False and commenting out the following line of code from *main_example.py*:

```
spg.get_power_profiles()
```

And adding the following code in the same place, replacing 'sample_pv_production.csv' with the filename for your production data:

```
tmy_production = pd.read_csv(os.path.join('data',
   'sample_pv_production.csv'), header=None, index_col=0)[1]
spg.get_power_profiles_from_upload(tmy_production)
```

Note that the parameter tmy_production must be a Pandas series. Also, it is assumed that the uploaded file contains the power production for a 1 kW array. This code will still create a series of outage profiles, but they will be sampled from the uploaded power profile instead of the ASP statistical model.

3.6 Using Different Load Profiles for Normal and Emergency Operation

To specify a separate load profile to use for emergency operation, set the off_grid_load_profile parameter in *main_example.py* to read in a csv or Excel file. The format restrictions for this parameter are the same as for annual_load_profile. If this parameter is specified (and not set to None by default), the annual_load_profile is used to size the PV and tidal systems and calculate annual revenue, and the off_grid_load_profile is used to size the battery and the generator, as well as calculate the resilience metrics. If, however, the battery_sizing_method parameter (described in the next section) is set to 'no_RE_export', then the annual_load_profile is also used for sizing the battery, since it should be sized to capture excess PV during normal, grid-connected operation.

3.7 Sizing a Battery to Avoid Exporting Excess PV to the Grid

If the site is in a location where net-metering is not allowed, it may be useful to size the system so that the battery will absorb all excess generation produced by the PV system during normal operation. This can be accomplished by setting the batt_sizing_method in *main_example.py* to 'no_RE_export' (the default is 'longest_night'). If this option is used, MCOR only returns one battery size per PV size. It is still possible to include specific PV sizes using the include_pv or existing_components options, and battery sizes will be determined for these as well.

3.8 Running Sensitivity Studies

You can run sensitivity studies with MCOR to run several microgrid simulations while varying one parameter. To do so, copy and modify the module *main_sensitivity_example.py* instead of *main_example.py*. After the “Define simulation parameters here” comment, you can specify the parameter to be varied as well as the values to iterate over by modifying the sensitivity_param dictionary. For example, to run simulations with outage lengths of 6, 12, and 24 hours, you would set:

```
sensitivity_param = {
        'param_category': 'system_inputs',
        'param_name': 'length_trials',
        'param_values': [6, 12, 24]
}
```

To view a list of parameters that can be used for the param_name, see the ALLOWED_SENSITIVITY_PARAMS list in *main_sensitivity_example.py*. Make sure that param_category is set to the dictionary within input_dict where the param_name parameter is set in *main_example_sensitivity.py*. Note that the value of that parameter set in input_dict will be overridden by the values specified in sensitivity_param[‘param_values’]. For sensitivity studies, the output Excel spreadsheet will contain the normal assumptions and TMY generation sheets in addition to a results sheet for each of the sensitivity parameter values.

The function plot_comparison_graphs can be used to compare the simulations across your sensitivity parameter. The following arguments can be modified to adjust the plot:

- comparison_metric: this controls which parameter value is plotted, options include ['pv_capacity', 'mre_capacity', 'battery_capacity', 'battery_power', 'generator_power_kW', 'fuel_tank_size_gal', 'pv_area_ft2', 'mre_area_ft2', 'capital_cost_usd', 'pv_capital', 'mre_capital', 'battery_capital', 'generator_capital', 'fuel_tank_capital', 'pv_o&m', 'mre_o&m', 'battery_o&m', 'generator_o&m', 'annual_benefits_usd', 'demand_benefits_usd', 'simple_payback_yr', 'pv_avg_load', 'pv_peak_load', 'mre_avg_load', 'mre_peak_load', 'gen_avg_load', 'gen_peak_load', 'batt_avg_load', 'batt_peak_load', 'pv_percent', 'mre_percent', 'batt_percent', 'gen_percent', 'fuel_used_gal', 'storage_recovery_percent']
- system_label: which microgrid system to display, options include ['least_fuel', 'least_cost', 'pv_only', 'mre_only', 'most_diversified', 'all']. This will control which of the several microgrid system configurations that MCOR simulates to display in the graph.
- sim_label: which simulation to display, options include ['avg', 'min', 'max', 'distribution']. If ‘distribution’ is selected, a boxplot will be displayed. Otherwise, a scatterplot will be displayed.

An example is shown in Figure 7 for a sample set of simulations that varied the length_trials parameter. In these plots, the storage_recovery_percent metric is shown with a few different options.

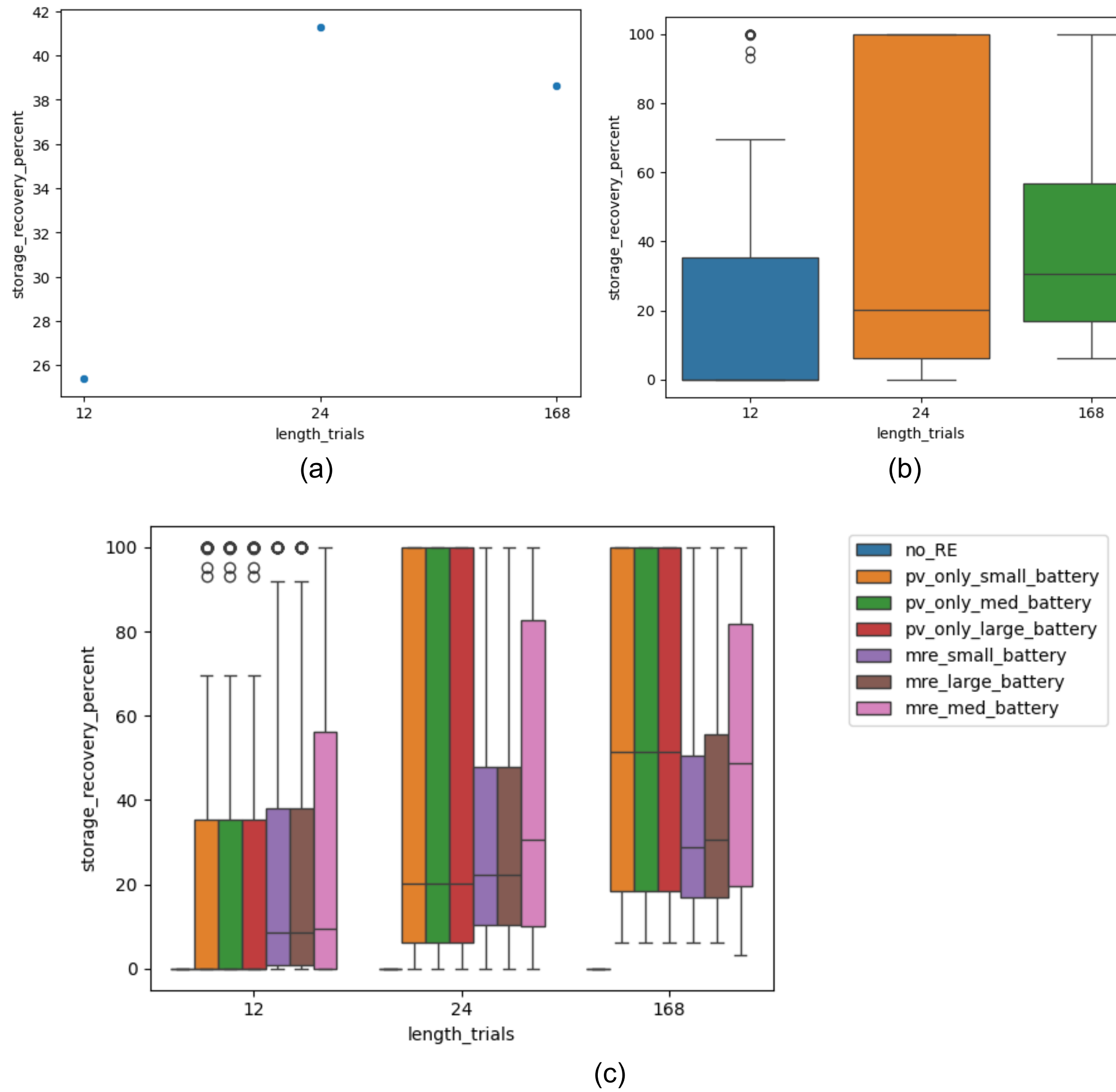

Figure 7. Sensitivity study output graphs showing how the storage recovery percent varies across simulations with different outage lengths. These were generated with the following function calls: (a) plot_comparison_graphs(output_dict, sensitivity_param, 'storage_recovery_percent', 'least_fuel', 'avg'),
(b) plot_comparison_graphs(output_dict, sensitivity_param, 'storage_recovery_percent', 'least_fuel', 'distribution'),
(c) plot_comparison_graphs(output_dict, sensitivity_param, 'storage_recovery_percent', 'all', 'distribution').

3.9 Modifying the Dispatch Plots and Generating Custom Plots

The dispatch graph plotting function includes several options to customize the plots. You can also generate figures which compare metrics across different system configurations or access the dispatch data directly for individual simulations to manipulate or plot as needed.

3.9.1 Dispatch Plot Options

The dispatch plots are generated in *main_example.py* using the function plot_best_system. You can set the dispatch plots to show either the two scenarios with maximum and minimum solar irradiance, the two scenarios with maximum and minimum tidal energy generation, the two scenarios with the least and most fuel consumption, or a specific scenario number. To toggle between these options, set dispatch_plot_scenario_criteria in *main_example.py* to either "pv", "mre" or "gen" or set dispatch_plot_scenario_num to the specific scenario number. You can also choose to have the plots displayed as a line graph or stacked graph by setting the argument stacked_graphs in the plot_best_system call to False or True.

3.9.2 System Comparison Plots

You can also generate plots that compare all system configurations using either one or two metrics (see examples in Section 1.1.6), by calling either the function plot_compare_metrics or plot_compare_sizes in *main_example.py*. For example, to generate a heat map showing how payback time varies across battery and PV sizes, you can run:

```
optim.plot_compare_sizes(var='simple_payback_yr')
```

To generate a bubble plot showing how payback and capital cost vary across battery and PV sizes, you can run:

```
optim.plot_compare_metrics(x_var='simple_payback_yr',
y_var='capital_cost_usd')
```

Below is the full list of metrics that you can include for these plots:

- generator_power
- fuel_tank_size_gal
- capital_cost_usd
- pv_capital
- battery_capital
- generator_capital
- fuel_tank_capital
- pv_o&m
- battery_o&m
- generator_o&m
- pv_area_ft2
- annual_benefits_usd
- demand_benefits_usd
- simple_payback_yr
- pv_percent mean

- batt_percent mean
- gen_percent mean
- generator_power_kW mean
- generator_power_kW std
- generator_power_kW most-conservative
- fuel_used_gal mean
- fuel_used_gal std
- fuel_used_gal most-conservative

Note: you will need to run a simulation first to generate the optim object. You can either make these calls directly in the Python console or add them to your *main_example.py file*.

3.9.3 Custom Plots and Analysis

To further dive into the results of an MCOR simulation and generate your own custom plots or analysis, it can be helpful to directly access a system or simulation object after running MCOR. The system object is described in detail in the Python module *microgrid_system.py*. To access a system object after an MCOR run, you can first call:

```
optim.print_grid()
```

to print the list of system names, then:

```
s = optim.get_system('pv_603.6kW_batt_174.7kW_699.0kWh')
```

with the chosen system name in quotes. To print general info on the system, type:

```
s
```

To see the full list of attributes and methods for the system, refer to the documentation for the MicrogridSystem class in *microgrid_system.py*.

You can also access a simulator object, which refers to an individual outage scenario for a particular system, using the system object from above and including the simulation number as an argument:

```
sim = s.get_simulation(5)
```

The simulation number is numerically assigned to each outage period.

The full list of simulator attributes and methods can be found in the documentation for the PVBattGenSimulator class in *microgrid_simulator.py*. A particularly useful attribute is the dispatch Dataframe, which is a Pandas Dataframe containing the dispatch information for every hour of the outage scenario. This can be accessed by running:

```
dispatch_data = sim.dispatch_df
```

You can also access the dispatch data by setting the input_dict['output_inputs']['save_timeseries_json'] parameter to True and loading the resulting json file saved to the *output/* directory.

4.0 Troubleshooting

4.1 Validation Errors

MCOR has a built-in validation checker that ensures that the input parameters are acceptable. If your inputs cause a validation error, you will get a message in the console describing the error and the allowable inputs for the associated parameter, e.g.:

```
validation.ParamValidationError: The parameter pv_tracking must be one
of ['fixed', 'single_axis']. You entered dual_axis
```

4.2 Solar Profile Generation Errors

Generation of the solar profiles relies on both an external API and code repository. Occasionally, the expected format or options for these change and it causes a problem with the MCOR code. If you encounter the following errors or any error that is raised in the *generate_solar_profile.py* module, please reach out to mcor@pnnl.gov as we may need to update the code:

```
NREL solar data empty. Check that you are using valid parameters to
access the NREL api.

Solar profile csvs not found. Please check that you have entered the
longitude, latitude, number, and length of trials for a site with
previously generated solar profiles.

Error downloading NSRDB data.

Failed to download solar data file, too many timeouts.
```

Note that you will need to have a valid *creds.yaml* file in the root mcor directory for the code to access the NSRDB api. If you get an error message related to the presence of that file, please check the README for instructions on how to properly format the file or see instructions under the Setup section above.

5.0 References

Anderson, K., and M. Mikofski. 2020. *Slope-Aware Backtracking for Single-Axis Trackers*. National Renewable Energy Laboratory NREL/TP-5K00-76626. Golden, CO. https://www.nrel.gov/docs/fy20osti/76626.pdf.

ASCE. 2005. *The ASCE Standardized Reference Evapotranspiration Equation*, edited by R. G. Allen, I. A. Walter, R. Elliott, T. Howell, D. Itenfisu, and M. Jensen. Reston, VA: American Society of Civil Engineers.

Bloomberg New Energy Finance (BNEF). 2015. *New Energy Outlook 2015: Long-Term Projections of the Global Energy Sector*. Bloomberg New Energy Finance. New York City, NY.

Coles, D. S., L. Mackie, D. White, and J. Miles. 2021. “Cost Modelling and Design Optimisation of Tidal Stream Turbines.” In *Proceeding of the European Wave and Tidal Energy Conference.* 2139-1–2139-11. September 5–9, 2021. https://eprints.soton.ac.uk/453917/.

De Soto, W., S. A. Klein, and W. A. Beckman. 2006. “Improvement and Validation of a Model for Photovoltaic Array Performance.” *Solar Energy* 80 (1): 78–88. https://doi.org/10.1016/j.solener.2005.06.010.

Dobos, A. 2012, “An Improved Coefficient Calculator for the California Energy Commission 6 Parameter Photovoltaic Module Model.” *Journal of Solar Energy Engineering* 134 (2): 021011. https://doi.org/10.1115/1.4005759.

Dobos, A. P. 2014. *PVWatts Version 5 Manual*. National Renewable Energy Laboratory NREL/TP-6A20-62641. Golden, CO. https://doi.org/10.2172/1158421.

Duffie, J. A. and W. A. Beckman. 1991. *Solar Engineering of Thermal Processes*, 2nd edn. New York: J. Wiley and Sons.

Hay, J. E. and J. A. Davies. 1980. “Calculations of the Solar Radiation Incident on an Inclined Surface.” In *Proceedings of the First Canadian Solar Radiation Data Workshop*, Toronto, Canada.

Holmgren, W. F., C. W. Hansen, and M. A. Mikofski. 2018. “pvlib python: A Python Package for Modeling Solar Energy Systems.” *Journal of Open Source Software* 3 (29): 884. https://doi.org/10.21105/joss.00884.

Jain, A., and A. Kapoor. 2004. “Exact Analytical Solutions of the Parameters of Real Solar Cells Using Lambert *W*-function.” *Solar Energy Materials and Solar Cells* 81 (2): 269–277. https://doi.org/10.1016/j.solmat.2003.11.018.

King, D. L., W. E. Boyson, and J. A. Kratochvill. 2004. “Photovoltaic Array Performance Model." Sandia National Laboratories SAND2004-3535. Albuquerque, NM. https://www.osti.gov/biblio/919131.

King, D. L., S. Gonzalez, G. M. Galbraith, and W. E. Boyson. 2007. “Performance Model for Grid-Connected Photovoltaic Inverters.” Sandia National Laboratories SAND2007-5036. Albuquerque, NM. https://www.osti.gov/biblio/920449.

Loutzenhiser, P. G., H. Manz, C. Felsmann, P. A. Strachan, T. Frank, and G. M. Maxwell. 2007. "Empirical Validation of Models to Compute Solar Irradiance on Inclined Surfaces for Building Energy Simulation." *Solar Energy* 81 (2): 254–267. https://doi.org/10.1016/j.solener.2006.03.009.

Madelung, O. 2004. *Semiconductors: Data Handbook*, 3rd edn. Berlin Heidelberg: Springer-Verlag.

Mongird, K., V. V. Viswanathan, P. J. Balducci, Md. J. E. Alam, V. Fotedar, V. S. Koritarov, and B. Hadjerioua. 2019. *Energy Storage Technology and Cost Characterization Report*. Pacific Northwest National Laboratory PNNL-28866. https://doi.org/10.2172/1573487.

Neary, V., M. Previsic, S. Jenne, and K. Hallett. 2014a. Reference Model 1 Cost Breakdown (RM1: Tidal Current Turbine). Marine and Hydrokinetic Data Repository. https://dx.doi.org/10.15473/1819896.

Neary, V., M. Previsic, S. Jenne, and K. Hallett. 2014b. Reference Model 2 Cost Breakdown (RM2: River Current Turbine). Marine and Hydrokinetic Data Repository. https://dx.doi.org/10.15473/1819895.

Neary, V., M. Previsic, S. Jenne, and K. Hallett. 2014c. Reference Model 4 Cost Breakdown (RM4: Ocean Current Turbine). Marine and Hydrokinetic Data Repository. https://dx.doi.org/10.15473/1819893.

Newman, S., Kaymie Shiozawa, Jim Follum, Emily Barrett, Travis Douville, Trevor Hardy, and Amy Solana. 2020. "A Comparison of PV Resource Modeling for Sizing Microgrid Components." *Renewable Energy* 162: 831–843. https://doi.org/10.1016/j.renene.2020.08.074.

O'Doherty, T., D. M. O'Doherty, and A. Mason-Jones. 2018. "Tidal Energy Technology." In *Wave and Tidal Energy*, edited by D. Greaves and G. Iglesias, 105–150. John Wiley & Sons.

ONSITE SYCOM Energy Corporation. 2000. *The Market and Technical Potential for Combined Heat and Power in the Industrial Sector*. ONSITE SYCOM Energy Corporation. Washington, DC. https://www.energy.gov/sites/prod/files/2013/11/f4/chp_industrial_market_potential.pdf.

Partridge, G. W., and C. M. R. Platt. 1976. *Radiative Processes in Meteorology and Climatology*. Amsterdam: Elsevier.

Previsic, M., and R. Bedard. 2009. *Yakutat Conceptual Design, Performance, Cost and Economic Wave Power Feasibility Study*. ReVision / Electric Power Research Institute EPRI-WP-006-Alaska. https://tethys-engineering.pnnl.gov/sites/default/files/publications/Previsicetal2009.pdf

Ramasamy, V., D. Feldman, J. Desai, and R. Margolis. 2021. *U.S. Solar Photovoltaic System and Energy Storage Cost Benchmarks: Q1 2021*. National Renewable Energy Laboratory NREL/TP-7A40-80694. Golden, CO. https://www.nrel.gov/docs/fy22osti/80694.pdf.

Reda, I., and A. Andreas. 2004. "Solar Position Algorithm for Solar Radiation Applications." *Solar Energy* 76 (5): 577–589. https://doi.org/10.1016/j.solener.2003.12.003.

Reda, I., and A. Andreas. 2007. “Corrigendum to ‘Solar Position Algorithm for Solar Radiation Applications’ [Solar Energy 76 (2004) 577–589].” *Solar Energy* 81 (6): 838. https://doi.org/10.1016/j.solener.2007.01.003.

Reno, M., C. Hansen, and J. Stein. 2012. *Global Horizontal Irradiance Clear Sky Models: Implementation and Analysis*. Sandia National Laboratories SAND2012-2389. Albuquerque, NM. https://www.osti.gov/biblio/1039404.

Seddon, L. 2016. *Vermont Solar Cost Study: A Report on Photovoltaic System Cost and Performance Differences Based on Design and Siting Factors*. Clean Energy States Alliance. Montpelier, VT. https://www.cesa.org/wp-content/uploads/Vermont-Solar-Cost-Study.pdf.

Sengupta, M. Y., Y. Xie, A. Lopez, A. Habte, G. Maclaurin, and J. Shelby. 2018. “The National Solar Radiation Data Base (NSRDB).” *Renewable and Sustainable Energy Reviews* 89: 51–60. https://doi.org/10.1016/j.rser.2018.03.003.

Stein, J. S., 2012. “The Photovoltaic Performance Modeling Collaborative (PVPMC).” *2012 38th IEEE Photovoltaic Specialists Conference*, Austin, TX, June 3–8, 2012. https://doi.org/10.1109/PVSC.2012.6318225.

Wenham, S. R., M. A. Green, and M. E. Watt. 1995. *Applied Photovoltaics*. Centre for Photovoltaic Devices and Systems.

Woodhouse, M., R. Jones-Albertus, D. Feldman, R. Fu, K. Horowitz, D. Chung, D. Jordan, and S. Kurtz. 2016. *On the Path to SunShot: The Role of Advancements in Solar Photovoltaic Efficiency, Reliability, and Costs*. National Renewable Energy Laboratory NREL/TP-6A20-65872. Golden, CO. http://www.nrel.gov/docs/fy16osti/65872.pdf.

Yang, Z., T. Wang, R. Branch, Z. Xiao, and M. Deb. 2021. “Tidal Stream Energy Resource Characterization in the Salish Sea.” *Renewable Energy* 172: 188–208. https://doi.org/10.1016/j.renene.2021.03.028.

Pacific Northwest
National Laboratory

902 Battelle Boulevard
P.O. Box 999
Richland, WA 99354

1-888-375-PNNL (7665)

www.pnnl.gov