SAND2017-7294R

PV-RPM v2.0 beta – SAM Implementation

DRAFT User Instructions

July 7, 2017

Authors:

Geoff Klise[1], Janine Freeman[2], Olga Lavrova[1] and Renee Gooding[1]

1 – Sandia National Laboratories
2 – National Renewable Energy Laboratory

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

Introduction

This user manual is intended to provide instructions to volunteer beta testers on how to use Sandia National Laboratories (SNL) PV Reliability Performance Model (PV-RPM) features in the National Renewable Energy Laboratory (NREL) System Advisor Model (SAM) version 2017.1.17 r4 (NREL, 2017). This new feature is provided in SAM to allow users with reliability data the ability to develop and run scenarios where PV performance and costs are impacted from components that can fail stochastically.

This is intended to be an advanced user feature as it requires knowledge and data regarding different PV component failure modes. It also relies heavily on the SAM LK scripting language, which is not utilized by a majority of SAM users. NREL has published a SAM LK users guide (Dobos, 2017) and has multiple on-line help topics and videos to get users familiar with the scripting language and what it can do.

This user instruction manual will provide some background on how data collected from a PV system can be used as inputs in the PV-RPM model, which will give data owners the ability to develop their own reliability and repair distributions outside of the example provided here.

Background

The PV-RPM model was initially developed in 2010 by SNL as a proof-of-concept for better simulating the uncertainty when components experience faults or failures in a fielded PV system. As the events occur randomly, they can be represented as a probability distribution with specific parameters to define the severity of the event and when it may occur over a specific time-frame. Repairs or replacements are also represented with probability distributions, where the component remains in a failed state until the repair distribution is sampled and results in the component being returned to an operating state.

The previous platform was limited in its ability to simulate components other than PV modules and inverters. System design characteristics and the types of failure distributions available for use were also limited in the previous version.

In 2016, SNL partnered with NREL to move the PV-RPM algorithms from the proof-of-concept platform into SAM, via the LK scripting environment. Doing this allows users to see how the code works and gives them the ability to modify the code for their own purposes. The code is available in SAM through an open-source license, with copyright asserted from the DOE Solar Energy Technologies Office on 12/16/2016. The copyright language can be found within each of the SAM LK script files distributed with this instruction manual.

The algorithm was thoroughly tested in SAM to ensure that the addition of these new features would not impact the SAM source code, and to ensure consistency in results between theoretical scenarios analyzed in the proof-of-concept and new SAM version of PV-RPM. This testing was completed primarily in 2016. The PV-RPM model was also validated using real data from a large PV system, where comparisons were made between the number of failures captured by the proof-of-concept and the number of failures estimated by the new SAM version. System behavior was also analyzed, comparing failure and energy loss results considering repairs, or leaving components in a failed state. This provided additional certainty that the bottom-up modeling approach, where every component item can be subject to failure and repair, worked as designed. The results of these efforts are outlined in more detail by Klise et al. (2017), which is also provided in the e-mail to beta testers. More references on the background of this effort, including links to supporting research can be found in this paper.

1. Types of Distributions Available for Analysis

As SAM uses the Sandia Latin Hypercube Sampling method for Monte Carlo analysis, the probability distribution function inputs are limited to that implementation. For more information on the LHS code, see the SNL Dakota user manual.[1] Table 1 shows the distributions that can be used in PV-RPM, though not all will be used for reliability analysis. The name of the parameters necessary for describing the distribution is also provided. More detail on the reliability distributions is provided in Appendix A.

Table 1 – SAM LHS available distributions

Distribution	**First Parameter**	**Second Parameter**	**Third Parameter**
Uniform	Min	Max	
Normal	Mean (mu)	Std. Dev. (sigma)	
Lognormal*	Mean	Std. Dev.	
Lognormal-N	Mean	Std. Dev.	
Triangular	A	B	C
Gamma	Alpha	Beta	
Poisson	Lambda		
Binomial	P	N	
Exponential	Lambda		
Weibull	Alpha or k (shape)	Beta or Lambda (scale)	

*The Sandia LHS library included in SAM requires mean and error factor inputs into lognormal function. The Lognormal-N function requires the mean and standard deviation of the UNDERLYING normal distribution. However, we anticipate that most users will have the mean and standard deviation of the actual lognormal distribution. Therefore, the LHS function implemented in the PV-RPM script translates from input mean and standard deviation to the error factor before calling the lognormal LHS function. The translation equations used can be found at https://dakota.sandia.gov/content/latest-reference-manual, Keywords>Variables>lognormal_uncertain.

2. Running a Simple Model

This section provides instructions on how to run a reliability scenario “out of the box” using the two LK scripts along with the SAM file that was provided for analysis. Later sections and the Appendix provide detail on how to change reliability distribution parameters, along with a discussion on how reliability distributions can be developed and validated using time-interval failure and repair data. Users of SAM should be familiar with the PV model technical reference (Gilman, 2015)

2.1 SAM and Base Input

The first step is to ensure the proper version of SAM is installed. To do that, open SAM and look at the announcements banner. Just above that, you will see a box in “red” that will let the user know that a newer version of SAM is available. The user can also click on the “Check for Updates” box to get the latest release. In Figure 1 below, you can see that r2 is installed, though a newer minor update is available. The major release associated with the PV-RPM code is SAM 2017.1.17, and the release version at the time of this document is r4. The PV-RPM scripts will not work with earlier major release versions of SAM.

[1] https://dakota.sandia.gov/content/latest-reference-manual

Next, save the "**sample_small_project.sam**", "**PVRPM_Function.lk**", "**PVRPM_Main_Script.lk**," and "**confidence_interval.xlsx**" files to the same folder location.

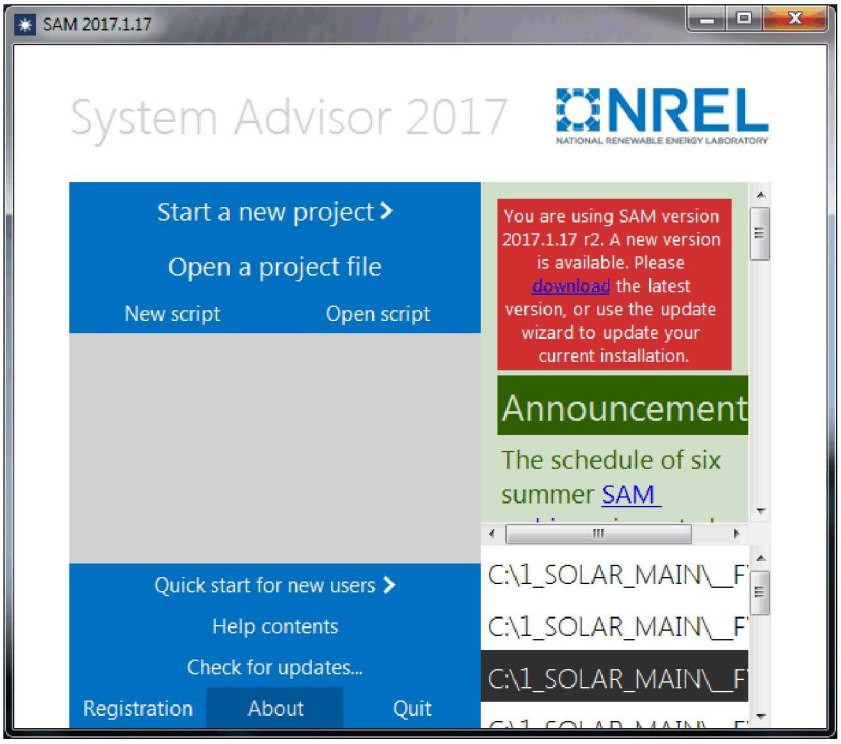


Figure 1. SAM model dashboard

Next, open the file "**sample_small_project.sam**" to get started. This file has a small example system pre-built to allow the user to explore the code without having to build a new system in the SAM dashboard. Many of these are defaults for specific SAM model types. In this case, we are starting with the "Photovoltaic (Detailed): Residential (Distributed)" performance model. The LK script PV-RPM model requires the performance simulation to be run in "lifetime mode", meaning that SAM simulates the performance for *every year* in the analysis, and not just the first year with derating factors applied in subsequent years. This means that only the Detailed PV model can be used, as lifetime mode is not available for PVWatts. The following inputs in this file are presented in Table 1 as a summary for the user. If specific areas and input values are not presented below, then default values from the scenario are assumed.

Table 2 – Small system input parameters

Model Input Steps	User Choice
Location and Resource	
TMY	USA AZ Phoenix (TMY2)
Albedo – Sky Diffuse Model - Irradiance	0.2 / Perez / DNI and DHI
Module (CEC database)	
Manufacturer and Model	SunPower SPR-X21-355-BLK
Temperature Correction	NOCT

Inverter (CEC database)	
Manufacturer and Model	SMA America: SB3800TL-US-22 (240V) CEC 2013
System Design (4 kWdc)	
Modules per String	6
Strings in Parallel	4
Number of Inverters	2
Tracking	Fixed
Tilt (deg)	20
Azimuth (deg)	180
Ground Coverage Ratio	0.3
Shading	None
Losses (Only Subarray 1)	
Irradiance - Soiling	5% for each month
Module Mismatch	2%
Diodes and Connections	0.5%
DC Wiring	2%
AC Wiring	1%
Lifetime	PV simulation over analysis period
Module Degradation Rate	0%[i]
Enable lifetime daily DC Losses	Check Box Not Selected [i]
Enable lifetime daily AC Losses	Check Box Not Selected [i]
Financial Parameters	
Analysis period[ii]	5 years

i – this will be defined in the script and discussed in a later section.
ii - Even though there are no loan, tax, insurance or salvage costs analyzed by the PV-RPM model, the analysis period needs to be set on this page, and the financial parameters chosen will affect SAM's calculation of the LCOE.

In the "Lifetime" area, you may notice a new box with "Lifetime Daily Losses." This was implemented in the last major version as a way to allow the reliability model to represent the downtime associated with faults or failures of any of the components being simulated. The LK PV-RPM model passes the DC Losses and AC Losses to this input to allow for the time component of the outage to be used in determining energy loss and the effect on system costs. This is different than the "Curtailment and Availability" box on the "Losses" page where these are used to schedule specific events that occur at the same time every year. The PV-RPM script will automatically set the length of these loss arrays to match the number of days that are specified in the "analysis period"- in this example case, 1852 days (5 years). Likewise, the PV-RPM script will also automatically set these inputs with the values calculated by the PV-RPM script, so the user should not enter anything for this option.

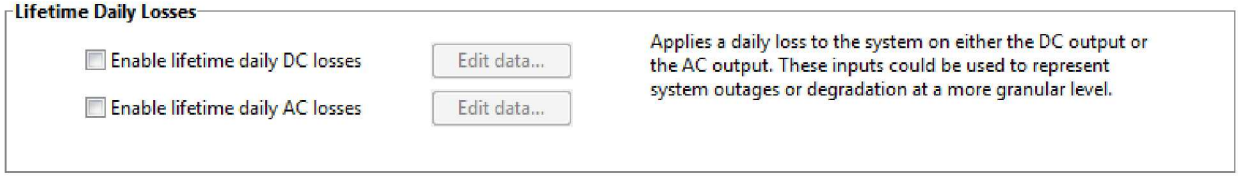


Figure 2. Loss inputs utilized by PV-RPM

2.2 LK Script Input

The next step involves opening the project scripts. For the reliability model to run, only the "PVRPM_Main_Script.lk" needs to be opened. The "PVRPM_Function.lk" script should not be changed by the user. However, experienced users of LK can change the output file parameters in the Function script in the very last section "Calculate Statistics and Write Results" if there are specific results that may be of interest that are not in the .csv output files.

Open the script by going to File/Open script. From here, navigate to the folder where the .sam and two .lk scripts reside. Open the "PVRPM_Main_Script.lk" file.

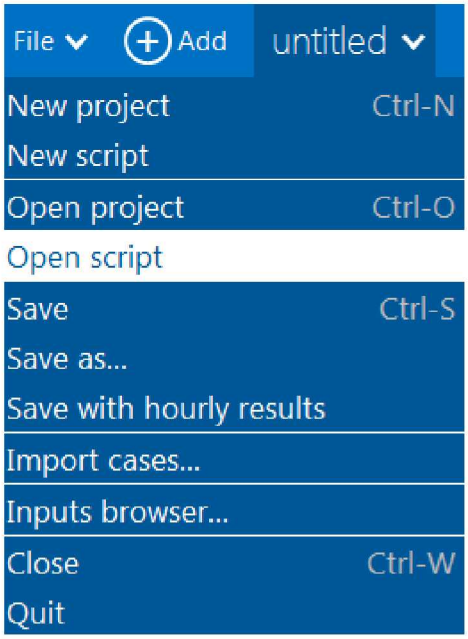


Figure 3. File Window

Opening the file in the LK environment will allow the script to be run within SAM. The top of the file should look somewhat like Figure 4. If the other script is not in the same folder location, an error message will show up in the lower dialog box. From here, the user needs to set a folder for the .csv results files. On line 48, set the path including the folder location. In this example, the script will place all the output files in a pre-existing folder named "pvrpm_results". Those who have seen demos of the LK PV-RPM script know that realization graphs pop up after a simulation. These are turned off on line 49 as the Monte Carlo sampling can result in too many open dialog boxes. To turn these graphs back on, simply change the variable "show_realization_graphs" to 'true'. In some instances, too many open graphs may cause the script to fail so it is recommended to the user to create multiple "cases" or use the standalone DView software for visualizing results. Later, we will show a few examples programmed into the code that allows the user to visualize some of the timeseries results outside of the data provided in the .csv file (Section 2.3.1).

Line 50 contains "number_of_realization_cases". This allows the user to specify if they would like the script to create new SAM cases representing specific stochastic realizations. If

"number_of_realization_cases" is set to 0, the PVRPM script will not create any new cases, but if it is set to a number *n* greater than zero, the script will create cases for the first *n* realizations that it runs, so that the user can take a more detailed look at what some of the realizations look like. A SAM case is considered a complete set of input data and results, like tabs on a worksheet. Since the input data is the same, the stochastic results can be analyzed for up to 10 cases/realizations on the SAM dashboard.

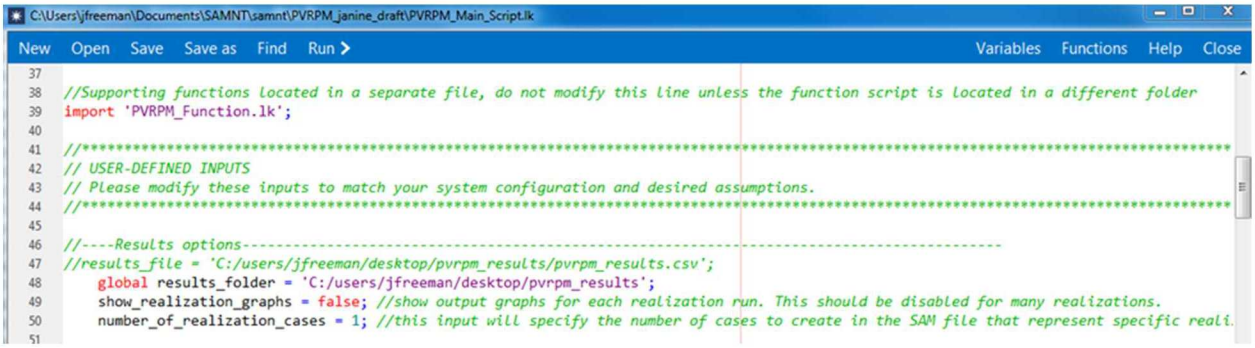


Figure 4. File location and realization graphs in LK script

2.2.1 Additional Setup Inputs

The next set of inputs have to do with additional system details that are not included in the main SAM dashboard. These are the number of DC combiners, number of transformers and number of trackers. Financial inputs such as the labor rate for O&M activities associated with a failed component, and inflation rate for looking at future costs of repair can also be changed here. The tracker failure algorithm input is explained later in this manual.

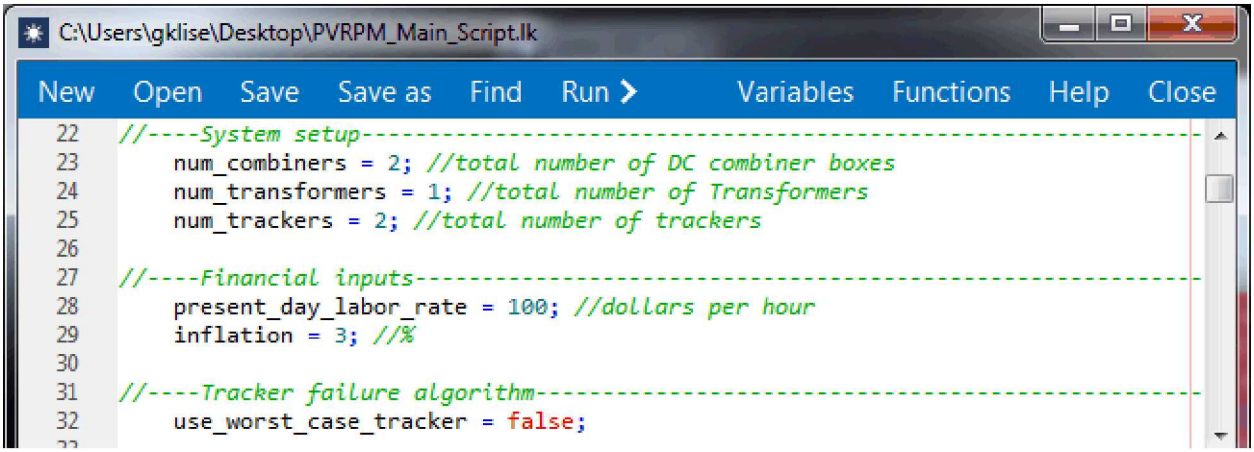


Figure 5. Additional component information and financial inputs in LK script

Figure 6 below illustrates how the system is currently configured, with circles showing the additional features added to the base model in SAM through the LK script. These components need to be defined here even if not simulated with failure modes in the model. Stand-alone transformers are not typically

found in small residential systems, though one is included in the small system demonstration model. Trackers are not included in the base model as Figure 1 indicates that this is a fixed-tilt system. Whether the tracker can fail or not can be controlled later in the script. For now, the number of trackers will be entered here as "2" for a later scenario as presented in Section 2.4. An AC disconnect is not an input, but it is available to model and matches the number of inverters.

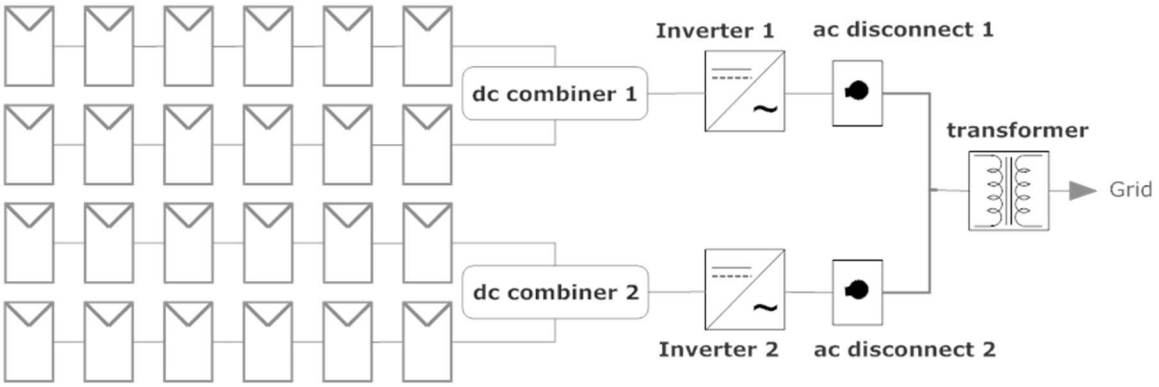


Figure 6. Layout of small PV system for demonstration model

2.2.2 Stochastic Analysis Inputs

This section defines how many realizations to run for the simulation. Figure 7 shows the code for the stochastic analysis inputs. To be able to calculate a desired exceedance probability (p_value), at least 2 realizations must be run. To calculate the confidence interval around the mean of multiple realizations (conf_interval), two or more realizations must be selected. If the desired confidence interval is 95%, 100 realizations is suggested. The Latin Hypercube Sampling method within the Monte Carlo analysis provides good sampling results with fewer iterations as it creates equal sampling intervals within the probability distribution that are each sampled, rather than just randomly sampling the entire distribution.

Note that although SAM runs hourly simulations, the PV-RPM model calculates failures and repairs in time steps of days. Therefore, all of the input data units should also be in "days". So any failure or repair distribution has to be developed with that specific time unit in mind.

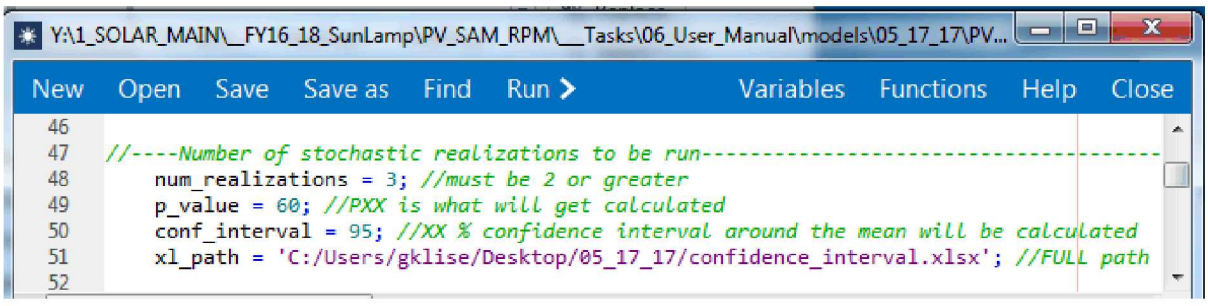


Figure 7. Stochastic analysis inputs in LK script

2.2.3 Module Inputs

The reliability model inputs for PV-RPM start at the module level. The user can turn the failure and repair modes (can_fail and can_repair) on or off by changing true to false (Figure 8). The paper by Klise et al. (2017) shows resultant failure and energy production behavior when repairs are turned off and failures can be measured over time on a decreasing population of operable components.

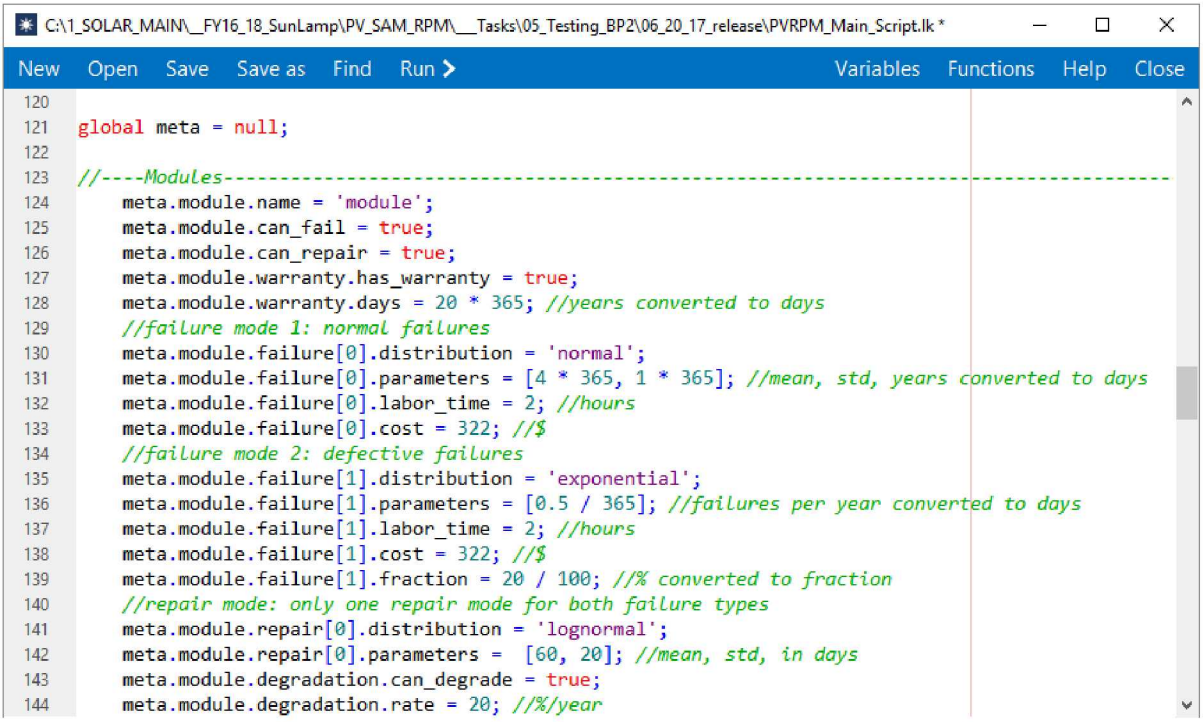


Figure 8. PV Module inputs in LK script

The next two lines, 127 and 128 address module warranty. The user can specify that the component has a warranty from the start of the simulation, or not, and determine how long that warranty will last before it expires. During that warranty time, it is assumed that the cost of the equipment replacement is covered, however the labor is not. After that period, both equipment cost and labor cost will be applied to that component. Lines under each failure mode allow the user to enter the labor time and component cost. The labor rate is already assigned for the entire simulation, as shown in Figure 5. It may be more realistic to have separate labor rates per failure mode type.

More than one failure mode can be assigned and more than one repair distribution can be assigned if there are multiple failure modes. Each failure mode is represented by a number starting at [0], [1]… [n]. Each failure mode must contain the four parameters “distribution” (selected from Table 1), “parameters” (the parameters associated with the selected distribution, also defined in Table 1), “labor time”, and “cost”. Repair distributions, on the other hand, only require the “distribution” and “parameters” to be specified. If only one repair distribution is defined, then all failure modes will be repaired using the same repair distribution. However, it may be desirable to have different types of failures matched with different repair distributions. For example, cycling faults would likely be

represented with a distribution that has a quick response time as the inverter may cycle without having a repair technician on-site, but other failures may be considered catastrophic and would fall under "defective failures" and may take longer to repair as the part may need to be ordered, or fixed later with an on-site warranty repair. To represent this in the PV-RPM script, the user must specify the same number of repair distributions as failure distributions, and in the same order. Matching the index value in the failure and the repair will ensure that when the component fails due to failure type [n], it is repaired with failure distribution [n].

To represent defective-type failures (i.e. failures that will only occur for a subset of components that are defective, rather than for all components) a user may specify an optional fifth parameter "fraction" in the failure distribution, as shown on line 110 above. "Fraction" specifies the fraction of the total population of that component type that is assumed to be defective. In this example, 20% of the modules will be eligible to fail per the exponential failure type [n].

The example model has titles for failure modes 1 and 2, which are just illustrative of how one might use comments to help keep track of different failure distributions.

Syntax is important here, so when changing data inputs, having the right brackets, semi-colons and single quotes is important.

The degradation rate is also available here for modules. Note that with the way the PV-RPM script is currently written, only PV modules are allowed to degrade. The PV-RPM script degradation is a substitute for the degradation specified in the SAM user interface. If any degradation is specified in the user interface other than 0, the PV-RPM script will throw a warning and can overwrite that input with 0 if the user agrees. The reason that the PV-RPM script handles degradation separately from the degradation input in SAM is that the SAM degradation input assumes that the degradation is constant for a whole year. We wished to show a more continuous degradation throughout the lifetime of the project, and also to be able to calculate how replacing modules might actually improve the fleetwide degradation rate. For that reason, fleetwide module degradation is calculated on a daily basis in the PV-RPM script and rolled up as part of the daily DC losses input, rather than entered into the degradation input in SAM. To enable module degradation, "can_degrade" should be set to true, and the annual degradation rate should be specified in the "rate" parameters in units of percent per year.

The two failure distributions for the small system demonstration are normal and exponential. The repair distribution is lognormal. Appendix A provides a discussion on translating the distribution parameters in terms of severity and time, and which parameters are more appropriate for certain failure modes. A user that will be analyzing their own data and developing distributions from that data will want to review the Appendix.

2.2.4 String and DC Combiner Inputs

Both String and DC Combiner (called "combiner" in LK) inputs are added in a similar manner as modules (Figure 9). String failures are generally due to cabling issues, where wire is damaged or the connection has come apart. DC Combiner inputs would be any issue with fuses, circuit breakers, busbars, circuitry, internal disconnect and internal wiring, starting where the wiring terminates in the combiner box, and before the combined wiring enters the inverter. In cases where the combiner is integrated with the inverter, it is up to the user to decide whether to model these separately, or ignore the combiner

altogether and create a failure mode within the inverter to represent the combiner portion, as long as any issue on the combiner side would stop all power flow to the inverter.

PV-RPM does not currently have the ability to model recombiners/subcombiners.

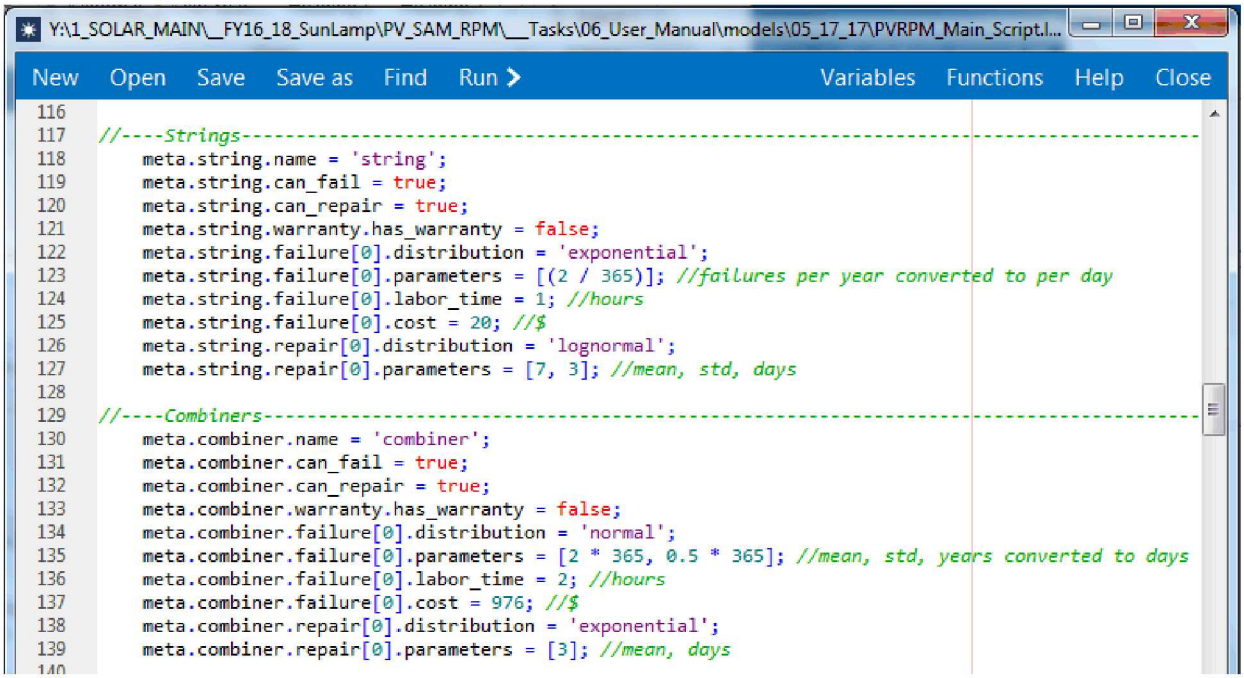


Figure 9. String and Combiner inputs in LK script

2.2.5 Inverter Inputs

The inverter script inputs are generally the same as modules, strings and combiners, but in this example script there is one additional cost consideration for what are considered ‘routine’ failures, which could be classified as nuisance tripping events, or other events that may be high frequency, but low in terms of energy loss ‘consequence.’ The user can set the inverter size and cost thresholds for this type of failure (Figure 10). The user can even choose to comment out this section and only look at one type of event and repair mode, or one failure mode with no repairs. See Section 2.3.2 below on how to comment out lines of code. Using the index feature [0], [1]…[n] allows the user to match specific repair distributions to failure events.

The line above each failure mode here is just descriptive text describing each failure mode. The example script that says ‘routine failures’ is only an example; the failure and associated repair distribution are not taken from fielded PV inverter data. Rather, they are just examples for this user guide.

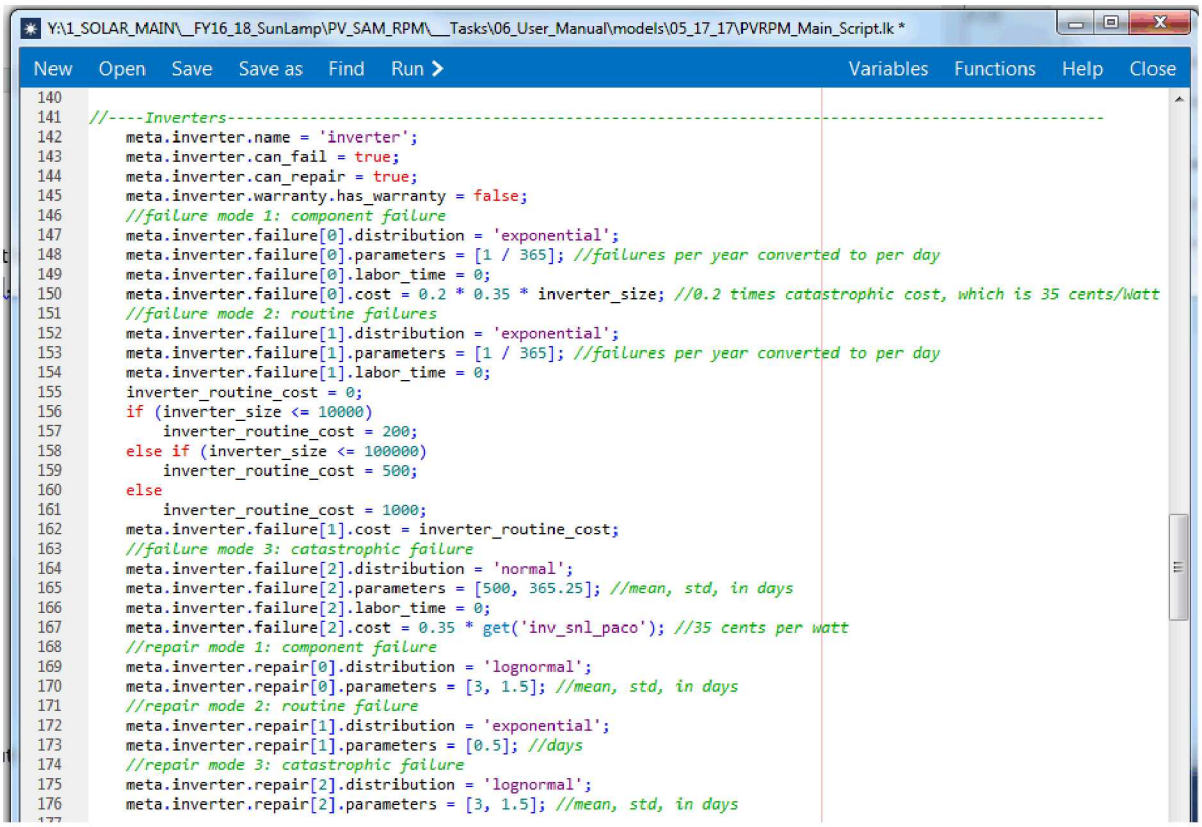


Figure 10. Inverter inputs in LK script

2.2.6 AC Combiner and Transformer Inputs

Just like other components shown above, AC disconnects combiner and transformer events can be simulated. The AC disconnect number matches the number of inverters. There is no separate AC combiner component in PV-RPM, however, depending on the system configuration, the transformer failure and repair distributions can be utilized for an AC combiner. Only one failure mode is shown below, however additional failure modes can be added as shown in the module and inverter sections.

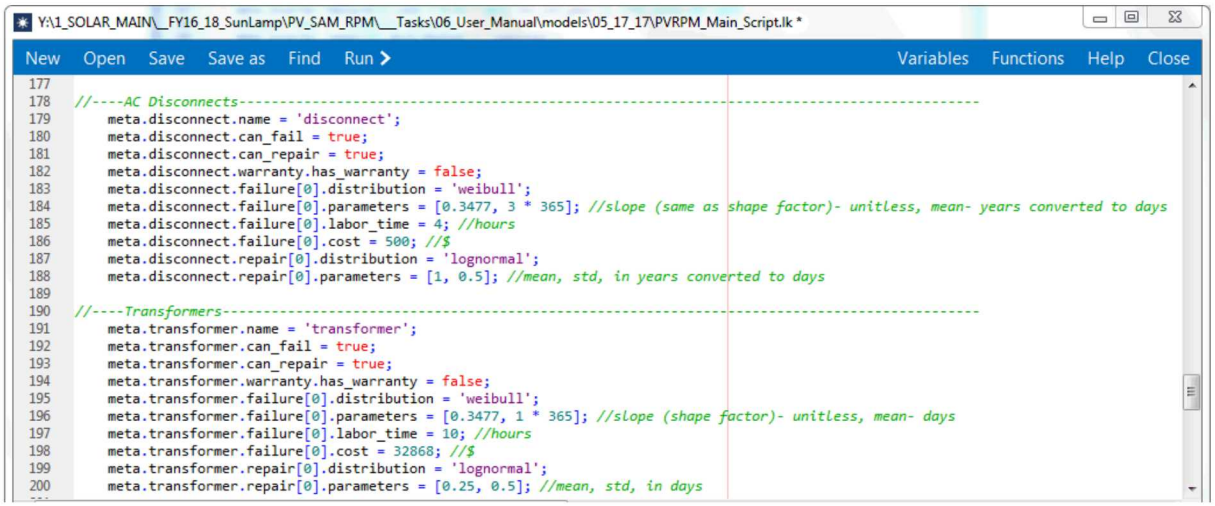


Figure 11. AC Combiner and Transformer inputs in LK script

2.2.7 Grid Inputs

PV-RPM gives the user the ability to stochastically simulate grid impacts, where the grid is essentially 'unavailable' for accepting energy (Figure 12). For users that want to look at scheduled curtailment and scheduled O&M events, that can be done in the main SAM user interface using the "Availability and Curtailment" inputs.

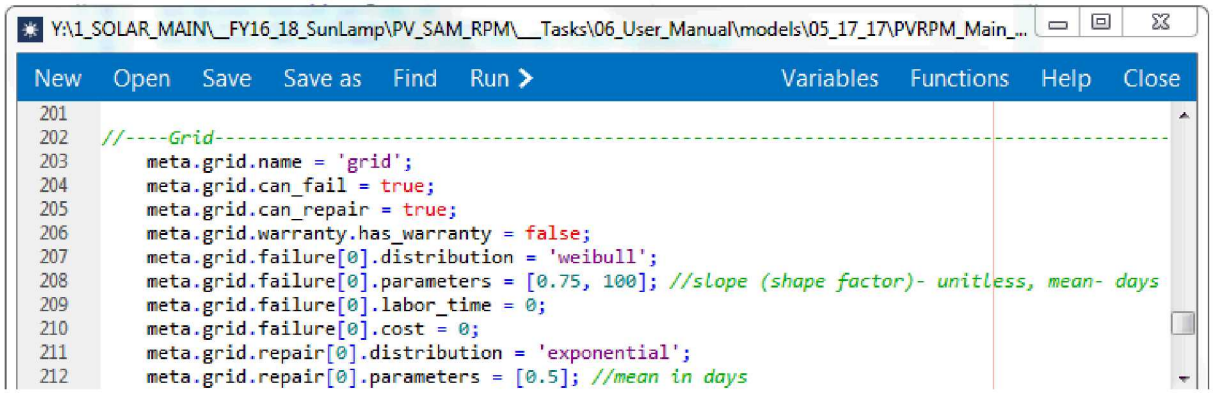


Figure 12. Grid inputs in LK script

For grid events that may potentially impact the inverter, AC disconnect or transformer, a failure distribution would need to be developed for that component and not the grid failure mode.

2.3 Running a Simulation and Results

After entering inputs for each component and failure mode, the model should be ready to run. If there are comments in the dialog box and a red vertical bar next to the code line number, that means there is a syntax issue that needs to be addressed (Figure 13). In this case, the exponential distribution is missing the denominator. In addition, the code will not inform the user if there is an issue with the desired path

to store the created .csv output files (rather, it will store the files EXACTLY where the erroneous path specifies), therefore it is important to review the syntax as shown in Sections 2.2. and 2.2.2.

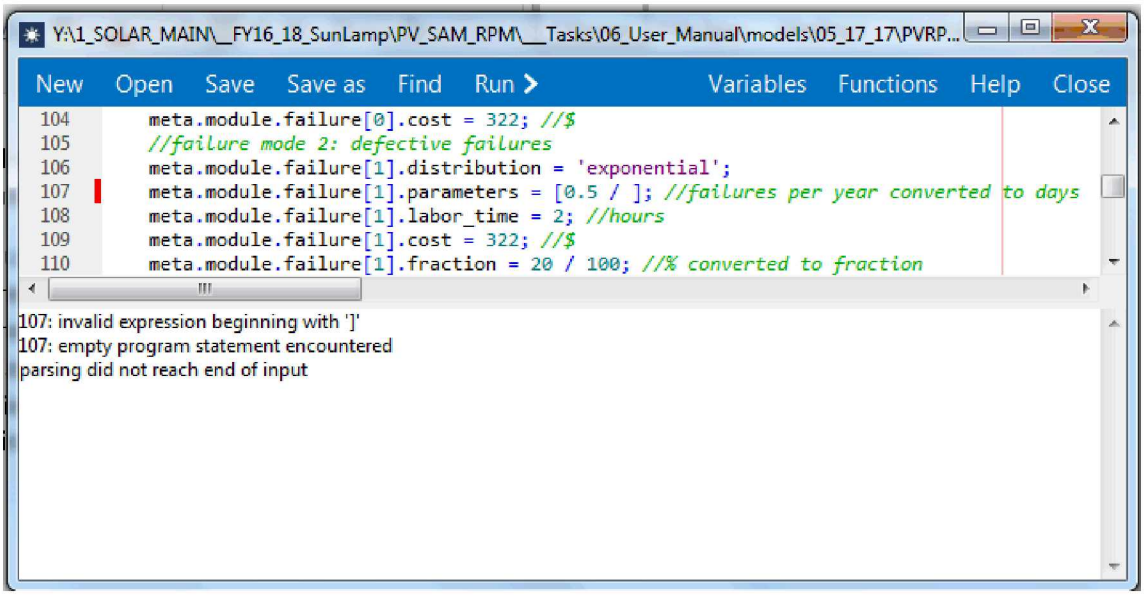


Figure 13. Error Message in LK script example

To run the model, press “Run>” in the top blue bar of PVRPM_Main_Script.lk.

The PV-RPM script draws several of its inputs from the SAM file that a user has created, including the number of modules per string, number of strings in parallel, number of inverters, etc. At this time, the PV-RPM LK script is limited to “regular” systems- meaning that there is a constant, integer number of modules per string, strings per combiner, combiners per inverter, disconnects per inverter, and inverters per transformer. The PV-RPM function does some error-checking of the SAM file that the user has set up, to try and prevent the script from crashing. If inputs are discovered that will make the PV-RPM script fail, message boxes appear alerting the user to the error and advising how to fix it. In some cases, logic is built into the script that enables the script to fix the error automatically and the user simply has to approve the fix.

While the user is running simulations, the PV-RPM script interacts with the SAM user interface (users may notice that sometimes this causes switching back and forth between the user interface and the script). Three inputs are set in the SAM user interface by the PV-RPM script for each realization:

- Daily DC losses: an array of losses that represents the combined effect of module degradation, module failures, string failures, DC combiner failures on the DC power delivered to the inverters.
- Daily AC losses: an array of losses that represents the combined effect of inverter failures, AC disconnect failures, transformer failures, and grid failures on the AC power delivered just past the point of interconnection.
- Fixed Annual O&M Costs (on the System Costs page): an array of costs incurred each year due to repairing failed components.

If the SAM user has specified that some realization cases be created, new cases will be created while the script is running, and these inputs will be set in those cases. If no realization cases are to be created, or

the number of realization cases has already been exceeded, these inputs will be set in the "base case" (the original user-specified case). After setting these inputs, the LK script runs the simulation in the SAM user interface, and then pulls output results from that simulation. Most of the output results are stored in memory, but the time series AC power (in kW), DC power (in kW), degradation, as well as the annual O&M costs by component, are stored in temporary .csv files in the same folder as the results will go for memory management purposes. This process repeats for the number of realizations specified by the user. After all the realizations are complete, the script uses the results stored in memory to calculate output statistics (described in Section 2.3.3). It also combines all the temporary .csv files into the final .csv output files and deletes the temporarily created ones. (If the script fails in the middle of the realizations for any reason, these temporary files may not be deleted automatically and the user will have to delete them manually). Lastly, the script re-sets the base (original user-specified) SAM case back to its original state.

After running the model, the dialog box at the bottom of the script will provide some summary data on annual O&M costs as well as failures per realization (Figure 14). At the bottom of the output, the total runtime is shown. This data is also presented in the output .csv files along with additional data on power, energy, mean time to failure and summary statistics.

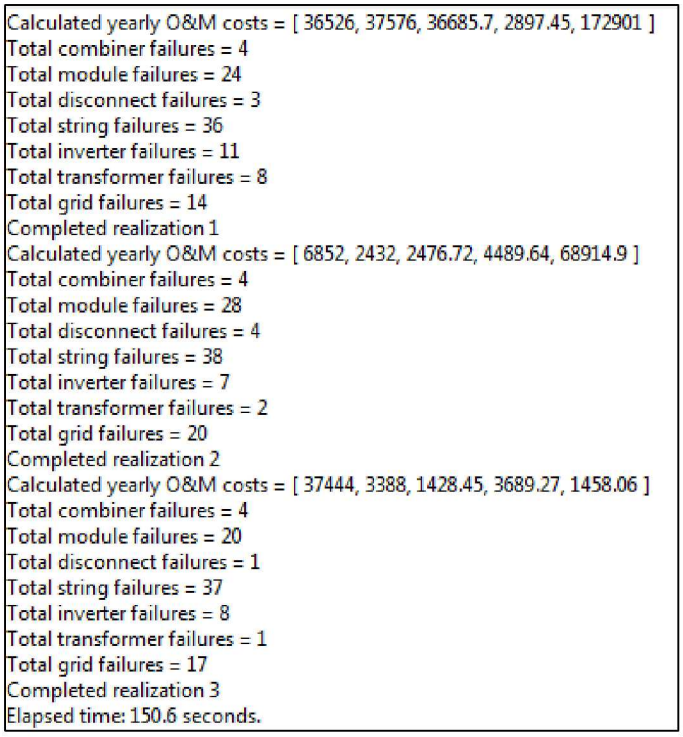


Figure 14. Failure results and costs per realization

2.3.1 Utilizing DView Graphs

DView is a time series viewer included in SAM that can also be downloaded separately.[2] Utilizing the stand-alone viewer will allow a user to view some of the output .csv files presented below in Section

[2] https://beopt.nrel.gov/downloadDView

2.3.3. For this section, we present the ability to toggle plotting per realization either on or off. Changing the 'false' to 'true' on line 20 will allow DView to plot results per realization for DC and AC Equipment Operation (like an operational availability), and DC and AC Power. Figure 13 below shows results of these outputs for two realizations. Each realization will have different results as each failure and repair event is sampled from a probability distribution. These graphs can be useful for analyzing behavior over a few realizations before choosing to run 100s of simulations. Other plots can be created using DView graphs in the "PVRPM_Function.lk" script by users that are familiar with the scripting language and referring to the LK scripting guide (Dobos, 2017). Note that the graphing display algorithm may sometimes create spikes above 1 when zoomed out sufficiently- if one zooms in on the spikes shown in Figure 13, it will become apparent that the value in fact never exceeds 1.

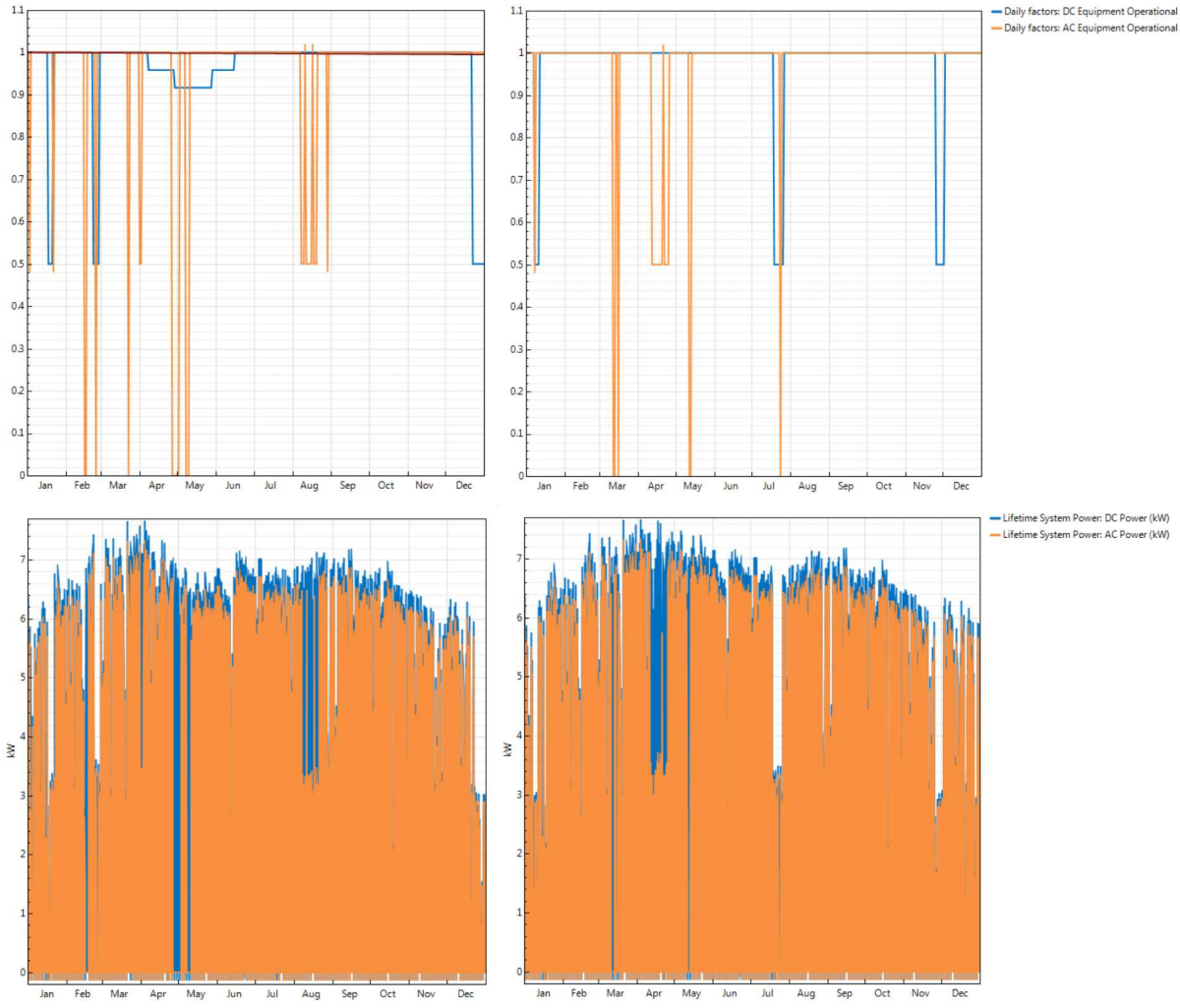


Figure 13. DView Output for Equipment Operational State and Power (Both DC and AC)

This graphic presents results some insight into how power production is impacted from component-level outages. The top two graphs are representations of the fraction of power that can be delivered as a function of a specific component or group of component outages. The bottom two are system DC and AC power in kW over the same time period. In this example, AC disconnect, transformer and grid outages all would appear to be the same in the 'daily factors' fraction, where a value of zero indicates

that no power can be delivered to the grid, whether the issue is on the grid side or on the PV side downstream of the interconnection location. This is also shown in the bottom set of graphs where the AC power drops to zero, though DC side production remains intact. On the DC side, it is a little easier to interpret which components are failing: the smaller dips in the top graphs are module failures, and the larger dips would represent a whole string or DC combiner failing in this system.

It is important to note that these graphs are not showing individual component failures (although those can sometimes be inferred from the magnitude of the step changes in the top plots), but rather the cumulative power impacts as measured at a specific point in the system. The DC power and availability graphs should be interpreted to be the sum of the power delivered immediately before the inverter inputs, and the AC power and availability graphs should be interpreted as the power measured at a point immediately past the interconnection to the grid.

2.3.2 Commenting out Code

To have a section of code completely ignored in the calculations, it can be commented out using the following methods in Figure 14. This is useful if the user wants to switch between failure modes for different simulations. For example, the code below on the left has two failure modes and two repair modes. The code on the right has the second failure and repair mode commented out to make the code only run through the "normal failures" case.

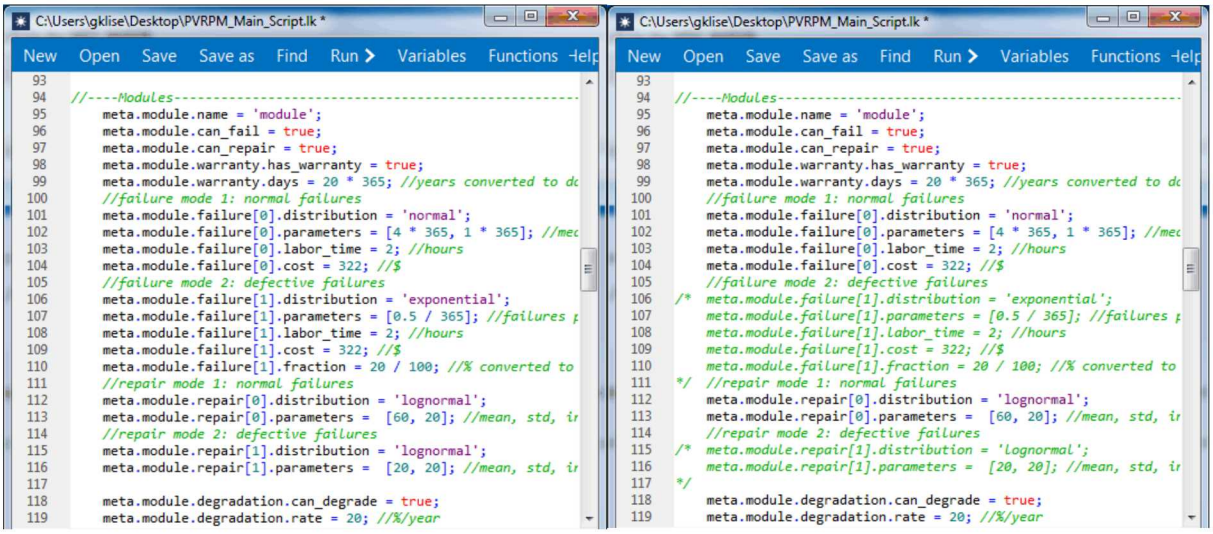


Figure 14. Code with two failure and repair modes (left) and one commented out (right)

Using the forward slash (/) and star (*) as shown on the right side of Figure 14, the "defective failures" failure and repair distributions are turned off and only the "normal failures" failure and repair distributions will run. As stated earlier, to turn off all reliability functionality for a particular component, change the meta.module.can_fail to 'false' instead of 'true'.

2.3.3 csv result files

Currently, results are presented in five .csv files:

- Timeseries_DC_Power.csv

- Timeseries_AC_Power.csv
- Daily_Degradation.csv
- Yearly_Costs_By_Component.csv
- PVRPM_Summary_Results.csv

Timeseries_DC_power and AC_Power provide hourly results over the entire time period for each realization. For the example files, there are DC and AC power results for 5 years (43800 hours).

Daily_Degradation provides daily results of the module degradation factor, expressed as (1 - degradation percentage). This was included for analysis of PV modules to help analyze situations where the degradation rate resets for modules that have been replaced. If a user wants to run a simulation with a high module failure rate which may represent a serial defect, then the degradation rate over the lifetime of the system can be estimated based on the fact that new modules will then degrade at a different rate than those that are not replaced.

Yearly_Costs_By_Component shows, for each realization, the breakdown of the O&M costs for each component, for each year of the analysis, as well as the total O&M cost for that year (the sum of the costs by component).

The PVRPM_Summary_Results file has summary statistics results as presented in Tables 3 and 4. The csv file presents the file as a matrix, which can increase the rows and columns depending on the number of years, realizations and component failure modes assigned for a simulation. Table 3 provides detail on the output parameter, and table 4 provides detail on the output results for each row, primarily the summary statistics for each parameter result.

Table 3 – Summary Results Output Parameters (columns in csv file)

Rows starting with [component] indicate that the metric described is presented for every component in the system (modules, strings, DC combiners, inverters, AC disconnects, transformers, and the grid)

Parameter	**Description**
LCOE	Real Levelized cost of energy, as calculated by SAM
[component]_failures_by_type_0*	The number of component failures by failure mode index [0]
[component]_failures_by_type_n*	The number of component failures by failure mode index [n]
[component]_total_failures	The total number of component failures for all failure modes
[component]_mtbf**	The mean time between failure (in days) for component failures, defined as the total uptime for that type of component divided by the total number of failures of that type of component. ***If there are no failures, it is the total uptime for the system lifetime.
[component]_availability	The availability of the component, considering uptime and downtime. 1 – (daylight downtime / Total number of daylight hours per simulation). Downtime is defined during periods where the PV system should be operational. E.g., at times when the irradiance level can power up the inverter.

annual_energy_1	Annual AC energy produced for year 1 (kWh)
annual_energy_n	Annual AC energy produced for year n (kWh)
cumulative_ac_energy_1	Cumulative energy produced through year 1 (kWh). For year 1, this will be equivalent to annual_energy_1
cumulative_ac_energy_n	Cumulative energy produced through year n (kWh). For example, cumulative_ac_energy_3 would be equal to the sum of the annual energy output from years 1, 2, and 3 combined
DC_energy_1	Annual DC energy produced for year 1 (kW)
DC_energy_n	Annual DC energy produced for year n (kW)

*Note that if only one failure mode is defined for a component, then only the total_failures column will appear in the results file
**Interpreting the MTBF should be done with caution as the failure mode selected may not be representative of a 'constant' failure rate. This should only be used to describe a failure rate when it is known that the component is not in either early wear out or end of life stages.
*** Mean time between failure definition found at http://www.weibull.com/hotwire/issue94/relbasics94.htm

Table 4 – Summary Results (rows in csv file)

Result	**Description**
Base	Specific parameter value for the base case simulation with all failure and repair distributions turned off (useful for comparison)
Realization_1	Specific parameter value for the first realization in the simulation
Realization_n	Specific parameter value for the n^{th} realization in the simulation
Min	The minimum value of all realizations for a specific parameter
Max	The maximum value of all realizations for a specific parameter
Mean	The mean value of all realizations for a specific parameter
Median	The median value of all realizations for a specific parameter
stdev	Standard deviation of all realizations for a specific parameter
XX% Lower Conf Int of Mean	The upper confidence interval of the mean of all realizations for the specific parameter based on the confidence interval chosen by the user
XX% Upper Conf Int of Mean	The lower confidence interval of the mean of all realizations for the specific parameter based on the confidence interval chosen by the user (two-sided, so choosing 95%, results in 5%; 80 % results in 20%, for example)
PXX	Exceedance probability result of all realizations for specific parameter based on P value chosen by the user

These results are available in a .csv file, located in the path the user defines within the top lines of code in PVRPM_Main_Script.lk. This information can be then plotted in an application of the user's choice. One example below is the cumulative annual energy production, using the mean and upper/lower 95% confidence interval value.

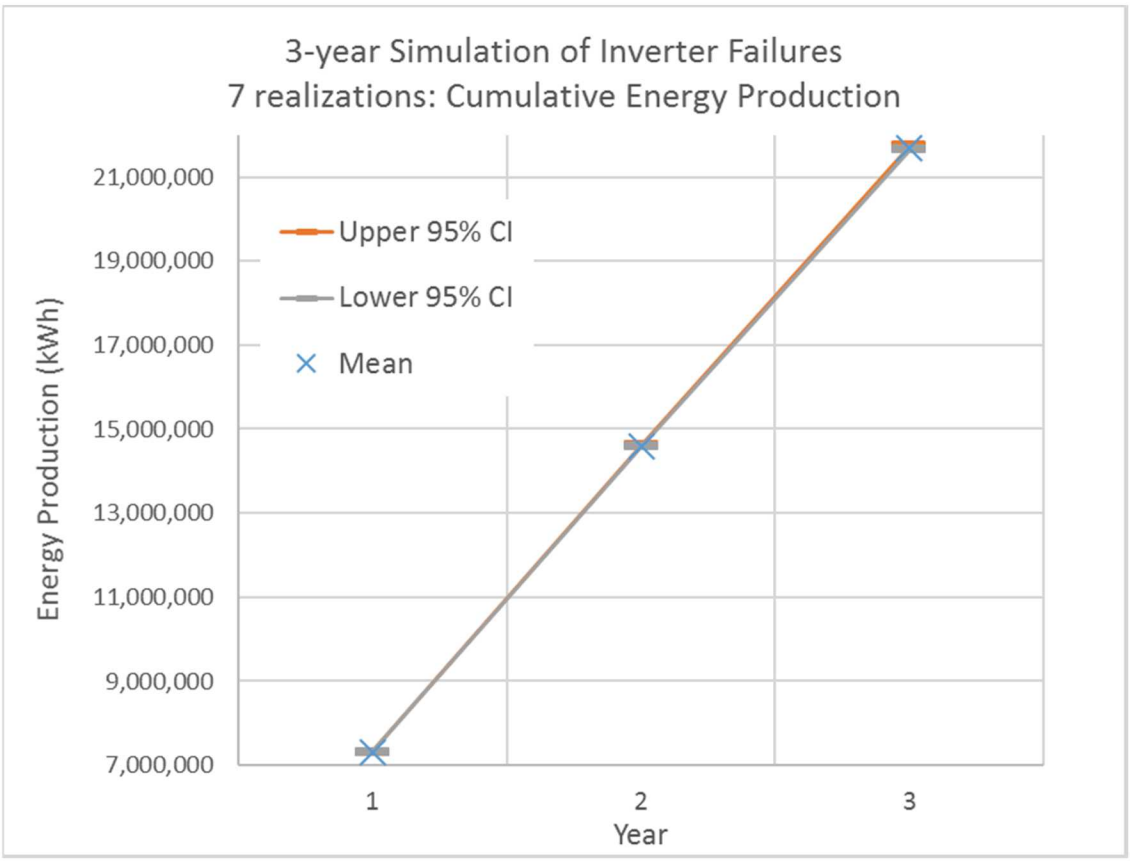


Figure 15. An Example of a Post-Processed Plot using Output Data

2.4 Simulation with Tracker Failure Enabled

If the user specifies a single-axis tracking system instead of a fixed-tilt system, there are a few additional considerations. At this time, single-axis tracking systems must be flat (zero degree tilt in the main SAM System Design page), due to the way that the power loss is calculated (explained below). Additionally, while SAM can currently model up to four subarrays with different orientations and tracking types, the PV-RPM script is constrained to only one subarray if single-axis tracking is involved, due to the complexity of the power loss calculation.

Unlike other system component failures, a tracker failure does not fully shut off the output from anything upstream of it. For example, if a string fails, none of the modules connected to that string can deliver power, but if a tracker fails, the modules on the tracker can still deliver a modified amount of power.

PV-RPM has two methods built in to represent the modification of power: a worst-case assumption and a best-case assumption.

- In the worst-case assumption, the tracker fails at its rotation limit facing west, north, or northwest (depending on the system azimuth).
- In the best-case assumption, the tracker fails flat, or “in stow” facing upwards with no tilt.

Which assumption to use can be set using the "use_worst_case_tracker" input in PVRPM_Main_Script.lk. Note that tracking systems will typically fail in one of those two positions, so they represent reasonable assumptions.

To determine how much power is lost due to a failed tracker, the LK PV-RPM algorithm is as follows:

Pre-calculate the benefit of the tracking system:

1. Run a base case simulation (no failures) with all trackers operating normally.
2. Run the same simulation, but with 100% of the trackers stuck for the entire year in the desired position (in effect, this is simulating a fixed tilt system at the failed position).
3. For each day, calculate the ratio of the energy produced by the "failed" system to the energy produced by the normally operating system- this is the "benefit" *D* that the tracker provides to the system.
    a. Note that these values are calculated on a daily basis because PV-RPM runs on a daily basis.

During a realization:

4. On a given day that a tracker has failed, find the corresponding daily "benefit" in the pre-calculated daily array, then apply a loss that is linearly proportional to the number of trackers that have failed, following the equation:

$$Power\ Loss\ Factor = D + X(1 - D)$$

    where: D is the ratio of energy production without trackers to energy production with trackers, and X is the fraction of operational trackers.
5. Multiply the calculated power loss factor with the normally calculated degradation and DC power loss factors (due to module, string, and DC combiner failures) to get the total DC Daily Loss for that day.

As an example, on a given day, imagine that the two pre-calculated simulations showed 1000 kWh of energy with trackers, and 900 kWh of energy without trackers. Therefore, 900/1000 kWh (90%) of the power would have been garnered even if the entire system were stuck in the fixed position. The extra 100 kWh (10%) is the extra "benefit" due to the tracking system on that day. However, if only 25% of the trackers are failed on that day, then the system is assumed to see 75% of the 100 kWh benefit of a fully operational tracking system. Plugging this into our equation above:

$$D = \frac{900}{1000} = 0.9$$

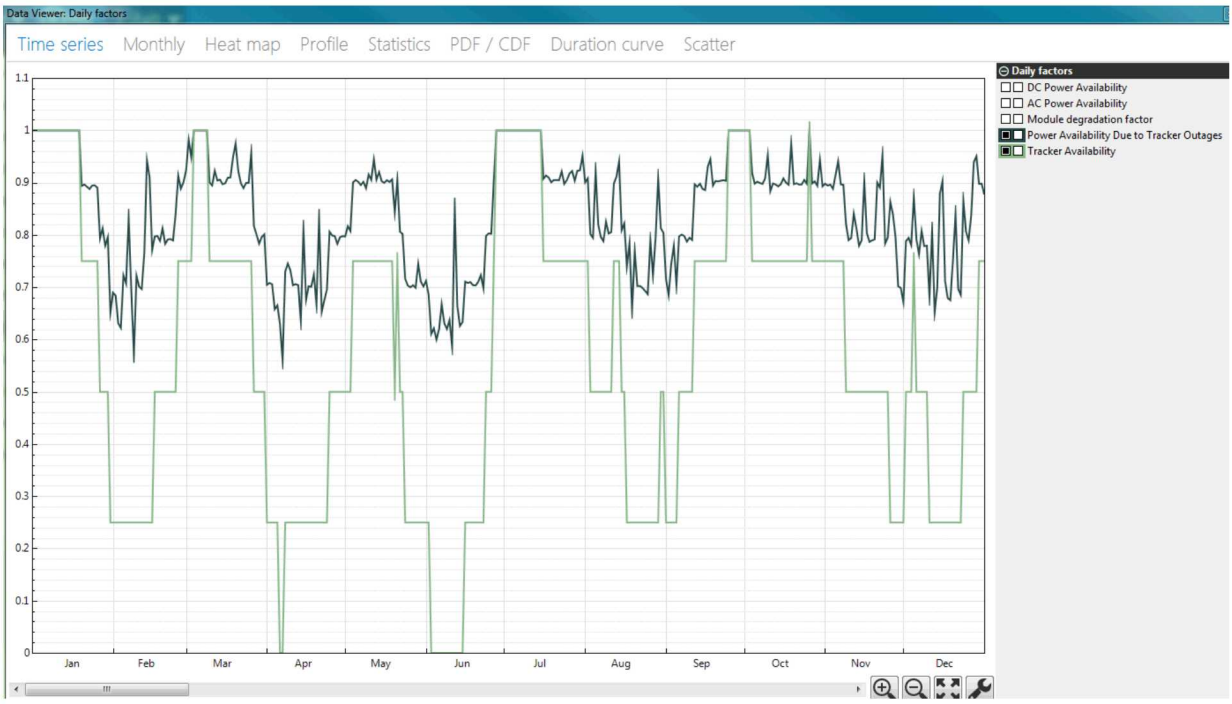
$$Power\ Loss\ Factor = D + X(1 - D) = \ 0.9 + 0.75(1 - 0.9) = 0.975$$

In our example, if the 25% of failed trackers were the only DC power loss occurring on that day (no other DC failures and no module degradation), then the system would produce $0.975 * 1000\ kWh = 975\ kWh$ of power on that day.

Note that having trackers failed flat may actually *increase* the system power production in rare instances where the majority of the irradiance is diffuse. This is because flat panels will see a higher portion of the sky dome than tilted panels throughout the day, resulting in a greater ability to use the diffuse light.

One other note is that in the current implementation, tracker replacement costs are the only component cost that is assumed to escalate due to inflation.

In a system with trackers, if a user enables the “show_realization_graphs” variable, then the plots will also include the effects of tracker outages. By checking or unchecking the boxes on the right hand side of the DView window, a user may show or hide different lines on the graph. Example graphs of the tracking outputs of a fictitious system with unrealistically high failures are shown below to illustrate the information found in the graphs.



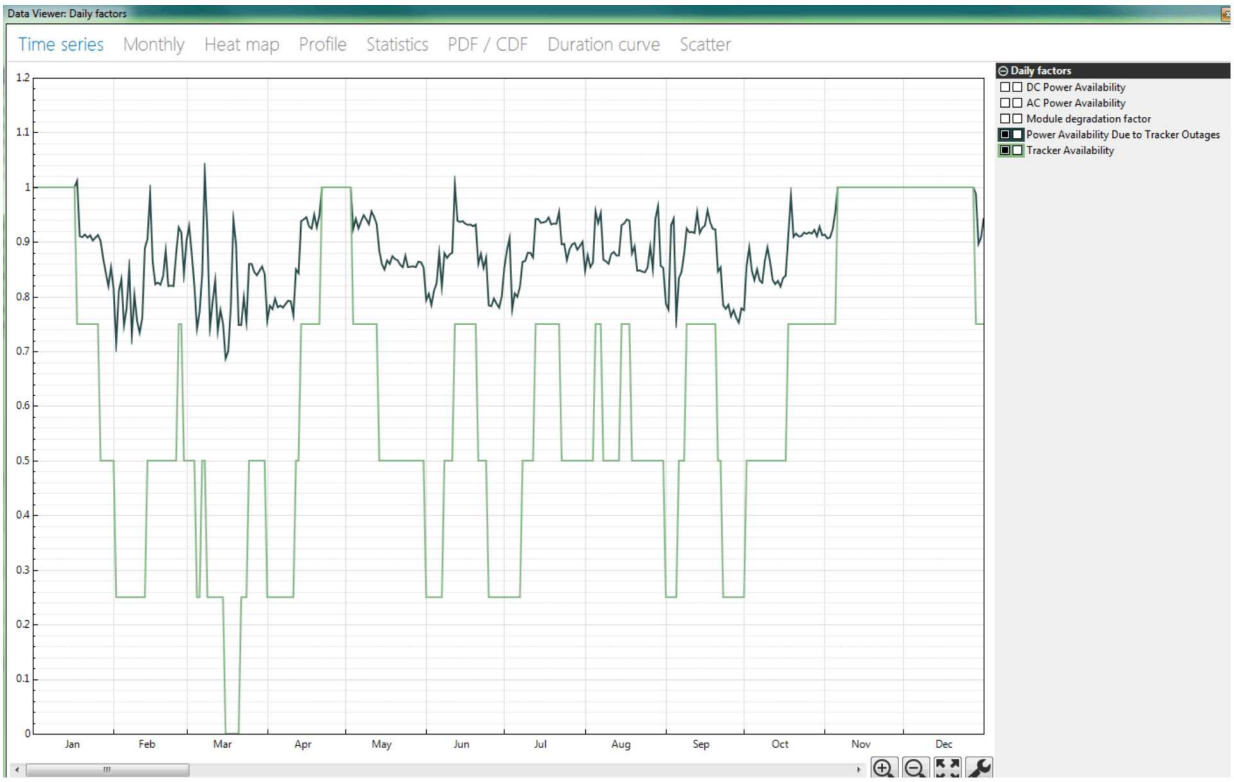


Figure 16. Example Tracker Availability Data in the Output Graphs

The “Tracker Availability” line shows the percentage of trackers that are available on a given day. The “Power Availability Due to Tracker Outages” line shows the power loss factor explained above. The top plot in Figure 16 shows a tracking system using the worst-case tracker failure assumption, and the bottom plot in Figure 16 shows the flat-failure assumption. Note that in March, the power loss factor spikes above 1, due to the reasons explained above.

References

Dobos, A. (2017), “LK Scripting Language Reference,” National Renewable Energy Laboratory, January 13, 2017. 46 pp. [*this file is accessible by clicking on the SAM help button, then “Scripting Reference” on the top menu bar*].

Gilman, P. (2015), “SAM Photovoltaic Model Technical Reference,” National Renewable Energy Laboratory, NREL/TP-6A20-64102. May 2015. 63 pp.

Klise, G.T., O. Lavrova, and J. M. Freeman (2017), “Validation of PV-RPM Code in the System Advisor Model,” SAND2017-3676, Sandia National Laboratories, Albuquerque, NM. 32pp.

System Advisor Model Version 2017.1.17 (SAM 2017.1.17). National Renewable Energy Laboratory. Golden, CO. Accessed July 7, 2017. https://sam.nrel.gov/content/downloads.

Appendix A –

Primer on Interpreting and Developing Failure Distributions for PV Components

Introduction

This Appendix provides some background on different reliability distributions that are used to represent failure and repair activities for PV system components. As this section is intended to be a primer on reliability distributions for PV components, it will not cover all reliability topics that could potentially be applied to PV component analysis. A good reference on how to select reliability distributions for stochastic analysis is presented by Seiler and Alvarez (1995). The purpose here is to educate users of the new feature in SAM on what types of reliability distributions may match best to certain types of failure and repair activities as seen and applied to PV components.

The reliability distributions presented here are those that are available for use in SAM, as implemented using SNLs open source DAKOTA package for Latin Hypercube and Monte Carlo sampling for uncertainty quantification. It is possible that some of the distributions presented in Table A-1 will not be used and others will be used more frequently. This table shows ten different distributions and the required input parameters. The Following sections will describe what changing distribution parameters will do and present visual representations and results of distributions utilized in the small system test scripts.

Table A-1 – SAM LHS Available Distributions

Distribution	**First Parameter**	**Second Parameter**	**Third Parameter**
Uniform	Min	Max	
Normal	Mean (mu)	Std. Dev. (sigma)	
Lognormal*	Mean	Std. Dev.	
Lognormal-N	Mean	Std. Dev.	
Triangular	A	B	C
Gamma	Alpha	Beta	
Poisson	Lambda		
Binomial	P	N	
Exponential	Lambda		
Weibull	Alpha or k (shape)	Beta or Lambda (scale)	

*The Sandia LHS library included in SAM requires mean and error factor inputs into lognormal function. The Lognormal-n function requires the mean and standard deviation of the UNDERLYING normal distribution. However, we anticipate that most users will have the mean and standard deviation of the actual lognormal distribution. Therefore, the LHS function implemented in the PV-RPM script translates from input mean and standard deviation to the error factor before calling the lognormal LHS function. The translation equations used can be found at https://dakota.sandia.gov/content/latest-reference-manual, Keywords>Variables>lognormal_uncertain.

Uniform

The uniform distribution is one that would likely not be utilized for reliability analysis of photovoltaic systems as it has a constant probability where there is an equal likelihood that an event would occur over the entire distribution. Figure A-1 below shows a pdf of a uniform distribution with a minimum value of 2 and a maximum value of 6.

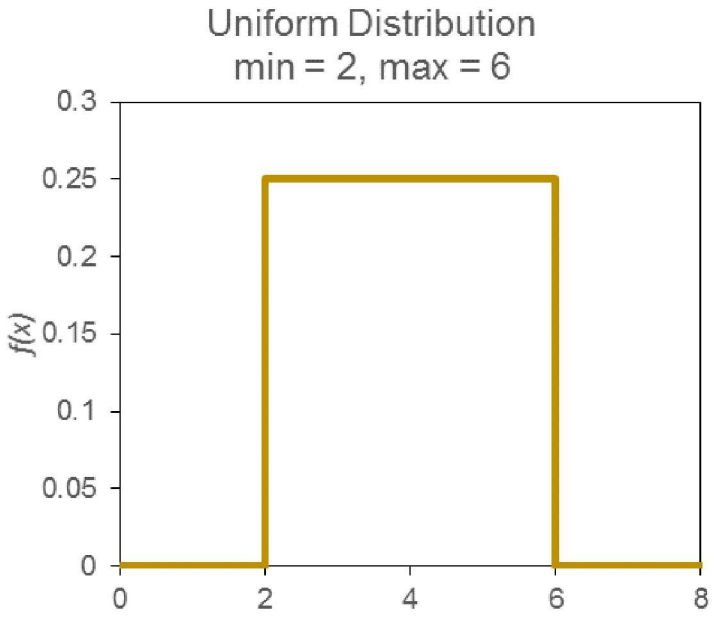


Figure A-1. Uniform Distribution

Normal

A normal (Gaussian) distribution is typically represented by the classic bell shaped curve, where the mean (mu or μ) is the location where the apex of the pdf occurs and the standard deviation (sigma or σ)) defines the height of the distribution, where 68% of the data that is sampled from the distribution will be found (Figure A-2).

Normal distributions are used when a component is expected, or known to have an increasing failure rate over time followed by a reduced failure rate later in life, for a mechanical system where there is external stress that creates a wearout effect, and for failures as a result of chemical processes that can result in corrosion, for example (Pham, 2006). A concern about using a normal distribution for reliability analysis is that if the standard deviation is too large, then negative time values may result. If the standard deviation is small, this can prevent that behavior.[3]

[3] http://reliawiki.org/index.php/The_Normal_Distribution

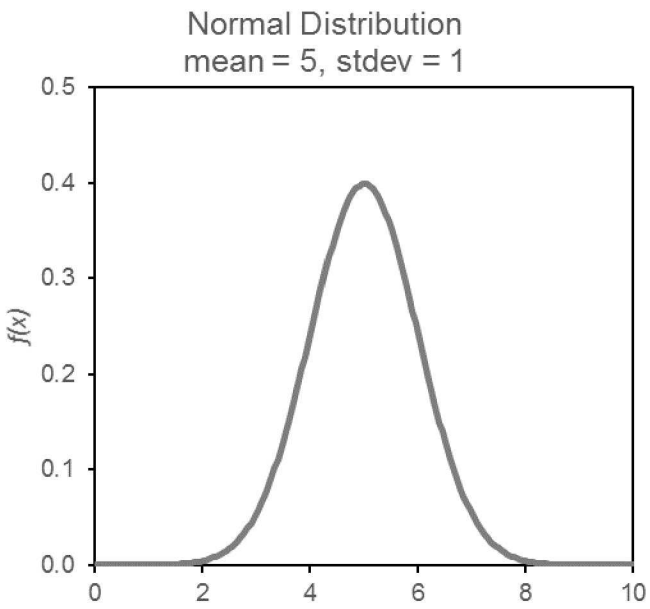


Figure A-2. Normal Distribution

Figure A-3 shows what happens when the mean is held constant but the standard deviation increases. The distribution peak moves down as the first standard deviation spreads out further to the left and right. The left tail of the flatter normal distribution shows where the negative time values may result.

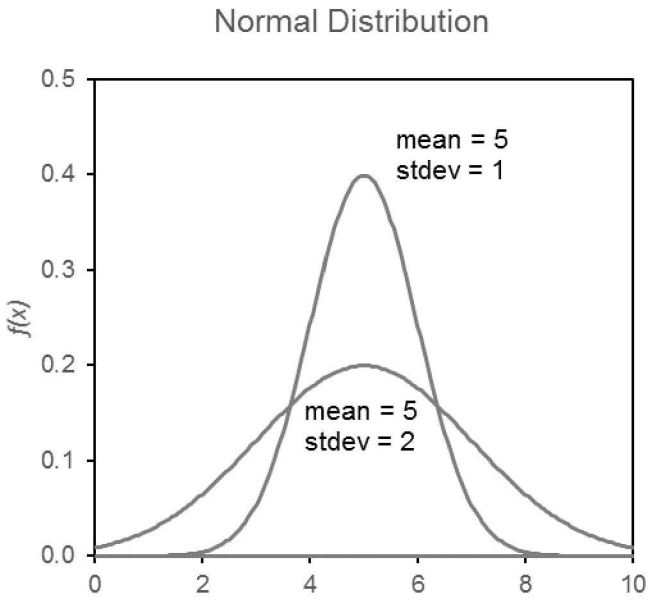


Figure A-3. Normal Distribution: Change in Standard Deviation

Lognormal

The lognormal distribution is useful for approximating component behavior due to fatigue related stress. This type of distribution is also good for modeling repairable systems, which can lead to time to repair

(TTR) estimates and repair distributions using maintainability data. When the data is positively skewed, it is possible to take the log of the data to approximate a normal distribution.

As mentioned in the Table A-1 footnote, the SNL LHS code as implemented in SAM requires mean and error factor inputs into the lognormal function. The Lognormal-n function requires the mean and standard deviation of the UNDERLYING normal distribution. However, we anticipate that most users will have the mean and standard deviation of the actual lognormal distribution. Therefore, the LHS function implemented in the PV-RPM script translates from input mean and standard deviation to the error factor before calling the lognormal LHS function. The translation equations used can be found at https://dakota.sandia.gov/content/latest-reference-manual, Keywords>Variables>lognormal_uncertain. Depending on the software used to develop the distribution, some lognormal inputs may have a negative value for the mean. The use of lognormal-n allows a negative mean value to be processed.

The parameters used for a lognormal distribution are the mean (mu or μ) and standard deviation (sigma or σ). Figure A-4 provides four different plots of the lognormal distribution to show how changing the mean and standard deviation impacts the spread and skewness of the pdf. In this case, the solid line plots have the same mean, and increasing the standard deviation from 0.5 to 1 results in a shorter peak that then shifts left on the x-axis (happens earlier time) becoming more right skewed. When the standard deviation is held constant as shown with the dotted lines, the distribution flattens out more as the mean increases, becoming less right skewed.

Considering a failure event that could be expressed by this distribution, there is an increased likelihood that the event will happen early on during the component lifetime, though over time, the probability that it will happen starts decreasing, either sharply, or more gradually. Using this as a repair distribution, there is a high likelihood that the failure will be fixed soon after the event rather than much later, such as nuisance tripping events for an inverter.

Much of what can be represented by a lognormal distribution can also be approximated with a Weibull distribution.

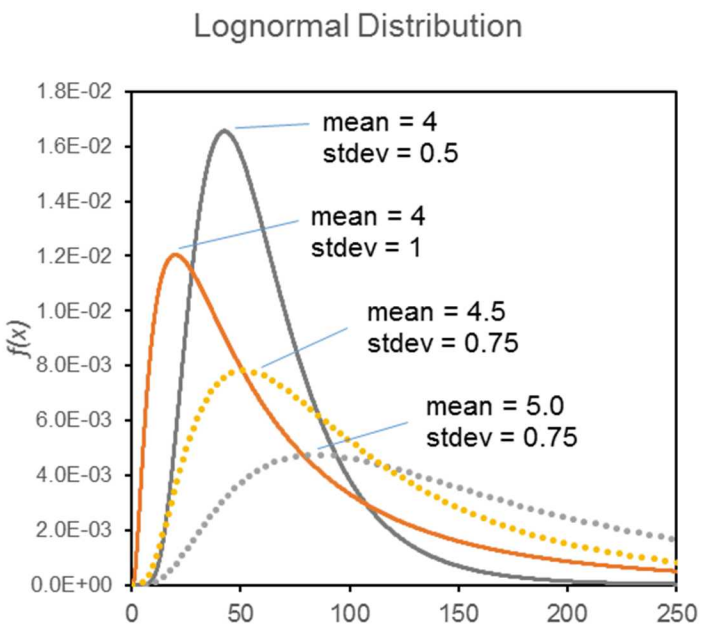


Figure A-4. Lognormal Distribution: Change in Mean and Standard Deviation

Triangular

This type of distribution is used when the component's behavior may be known, but there isn't a large enough dataset to develop a representative distribution. This allows the user the ability to define a minimum, maximum and most probable value. The triangle can be symmetric, or skewed either left or right. The SAM implementation asks for variables A, B and C in order of input into the function. A is the minimum x value where y = 0. B is the 'mode' or peak of the triangle. C is the maximum x value where y = 0.

The example below shows a non-symmetrical triangle, with a minimum time of 0 and maximum of 6, with the highest probability of an event at time 2.

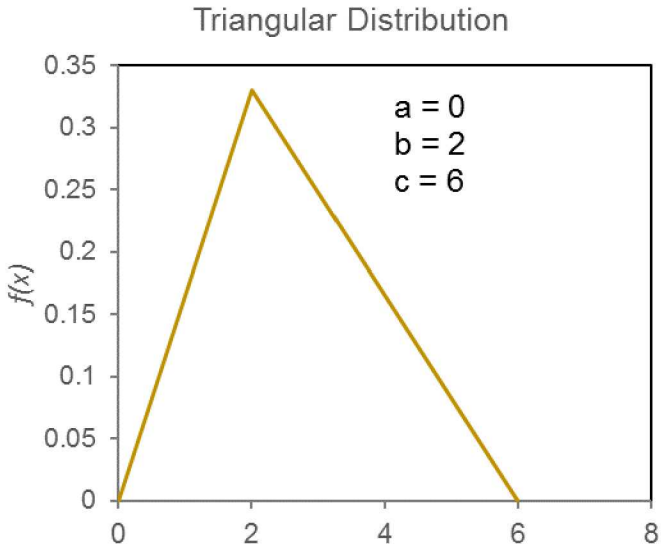


Figure A-5. Triangular Distribution

Gamma

A gamma distribution is one that can be used to represent a failure event where multiple 'partial' failures occur over time, resulting in complete failure of the component. It is not however a common distribution used for 'common failure mechanisms'.[4]

Alpha and Beta parameters are used in the Gamma distribution. Examples of holding the alpha constant and beta constant are presented in Figure A-6. When holding the alpha constant, an increasing beta lowers the peak and shifts it to the right. When holding beta constant, increasing alpha also lowers the peak and shifts it to the right.

[4] http://reliawiki.org/index.php/The_Gamma_Distribution

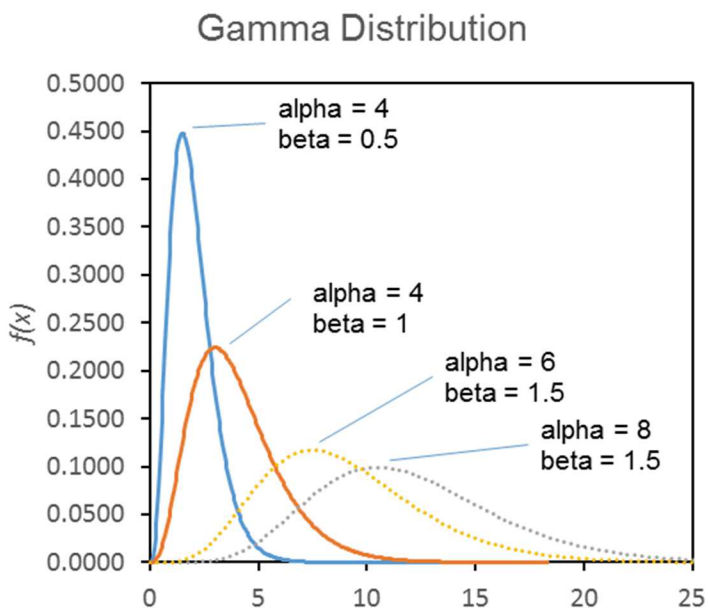


Figure A-6. Gamma Distribution: Change in Alpha and Beta

Poisson

A Poisson distribution is typically used in reliability settings to represent discrete events with a constant failure rate over a given time interval. This distribution is essentially a binomial distribution when there are low occurrence probabilities. Lightning events impacting a PV system can be modeled using a Poisson distribution. Spare parts analysis can also be done using a Poisson distribution, if a constant failure rate is already known.[5]

The symbol used in the Poisson distribution is Lambda (Shape parameter) which can be thought of the expected or average number of events. Increasing Lambda from 0 results in a shift of the distribution to the right, and a lowering of the peak value.

[5] https://src.alionscience.com/pdf/POIS_APP.pdf

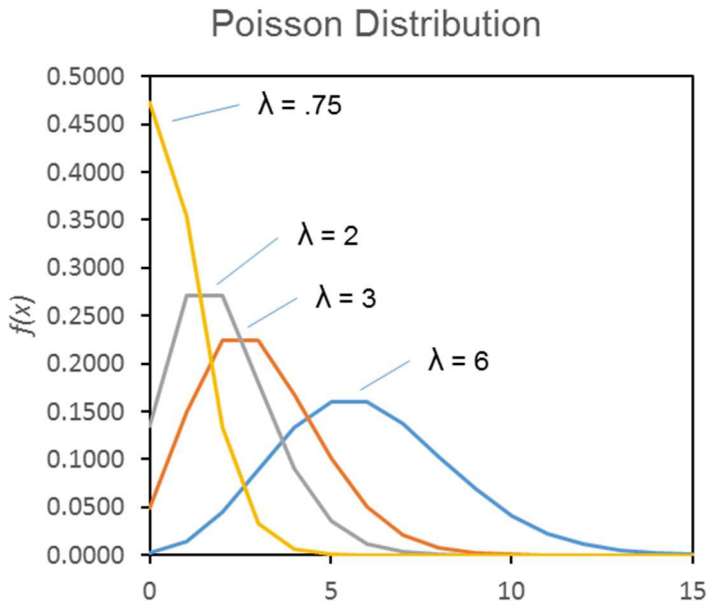


Figure A-7. Poisson Distribution: Change in Lambda

Binomial

Like Poisson, integer values are used as random numbers. However, binomial distributions are typically used in experiments where there is a "pass" or "fail" criterial. These will likely not be used in a system-level analysis of a PV plant and are more appropriate to use say in a manufacturing setting when analyzing defective parts used to build a specific component.

Exponential

An exponential distribution is used for components that have a constant failure rate. Electronic equipment is one area that can be modeled using an exponential distribution. For PV, inverters may have failure modes that follow an exponential distribution.

In this case, we are only considering a one-parameter exponential distribution. As Lambda increases, the distribution moves left and the peak increases (Figure A-8). The inverse of Lambda is the component's mean time between failure. However, that is only true if the component has a constant failure rate (it cannot be decreasing or increasing over time).

An exponential distribution is also the same as a Weibull distribution when the Beta/slope (shape) is equal to 1, meaning there is a constant failure rate.

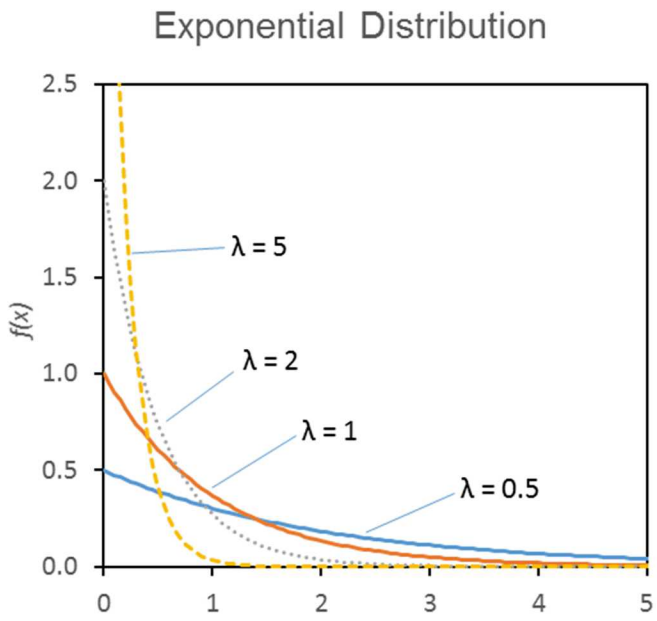


Figure A-8. Exponential Distribution: Change in Lambda

Weibull

Weibull distributions are the most versatile of all probability distributions and can be used in place of many of the other distributions presented in this appendix as it can handle constant and non-constant (decreasing or increasing) failure rates. It can be used to model component fatigue, corrosion, diffusion, abrasion and other degradation processes.

The Weibull distribution is changed primarily through the shape (slope) and scale (spread) parameters. There are many different parameter labels used in software programs. Therefore, remembering the shape and the scale will translate across different greek symbols used by different scholars. The most important aspects of the Weibull distribution are as follows:

- A shape parameter less than 1 means that there is a decreasing failure rate for that component.
    - This can indicate the infant mortality phase where most of the failures have already occurred and become less frequent over time.
- A shape parameter equal to one means the component has a constant failure rate.
- A shape parameter greater than 1 means there is an increasing failure rate.
    - As the component ages, the failure rate may start increasing as it reaches the end of its life.
- The scale parameter helps define the spread of the data and is the 63.2 percentile of the failure data.
    - For the first plot in blue (Shape = 0.5, Scale = 2), (Figure A-9) the scale of 2 would mean that 63.2 percent of the component would fail in the first 2 years (years on x-axis).

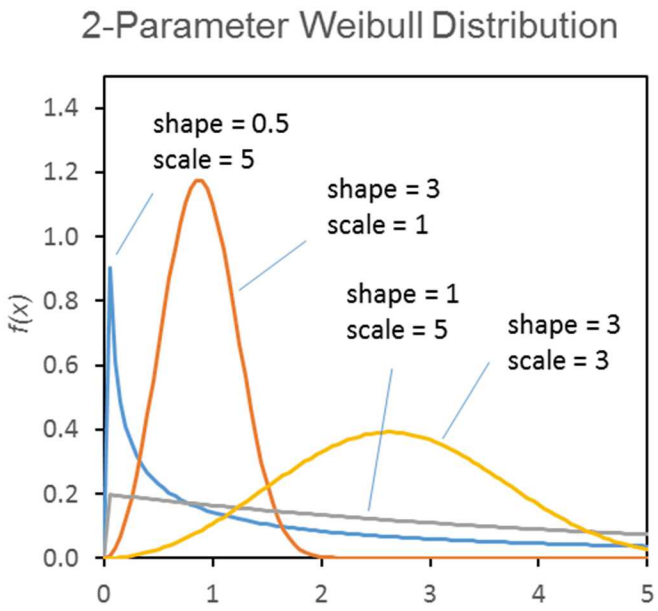


Figure A-9. Weibull Distribution: Change in Shape and Scale

The quintessential bathtub curve that is shown in many discussions of reliability engineering can be constructed from three different Weibull distributions.[6] If, for example, you want to simulate an inverter failure and have some knowledge that the inverter has not yet been extensively field tested. Figure A-10 shows three different distributions that can be used to simulate either general inverter failures, or can be used to isolate a specific component. How this can be done in SAM (As shown in Section 2.2.5) is to develop three failure distributions. For the first, using meta.inverter.failure[0].distribution = 'weibull', then on the next line for 'parameters', add in the shape first, and then scale parameter [0.5,2]. This can be repeated for the next two failure modes meta.inverter.failure[1].distribution and meta.inverter.failure[2].distribution. Specific repair distributions can also be defined for each failure mode, with parameters chosen to replicate how fast the repair will be addressed depending on the severity of the modeled component, or stage in the component lifetime.

As Weibull distributions are like others presented here, being able to compare different distributions may be of interest. A good way to make this comparison is available in this on-line calculator.[7]

[6] http://www.weibull.com/hotwire/issue14/relbasics14.htm
[7] http://biodevices.et.tudelft.nl/ReliabilityEngineering/Distributions/Compare/

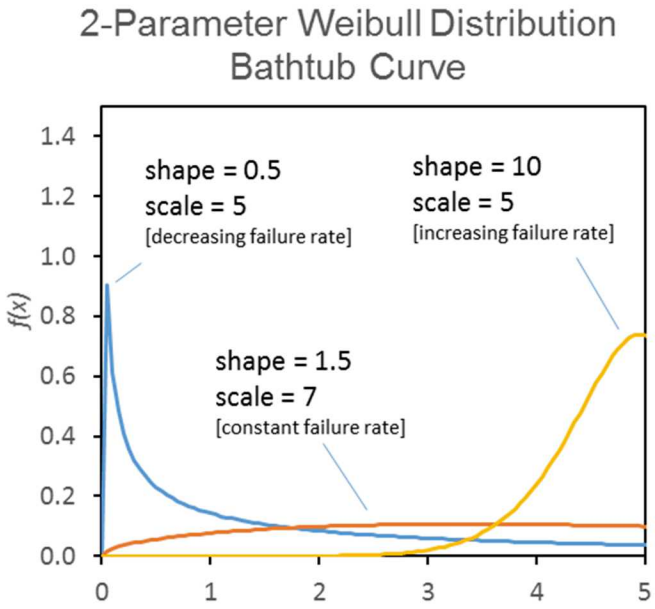


Figure A-10. Three distributions used to develop bathtub curve in a probability plot

Developing Failure and Repair Distributions

To determine the best fit reliability distribution for failure and repair activities, the time to failure (TTF) and time to repair (TTR) for the event in consideration must be calculated. The software used here to develop the distributions may use different conventions than other software, so the TTF and TTR presented here may be different than other software packages. To calculate the TTF, the commissioning time for the PV inverter is subtracted from each downtime start as shown in Figure A-11. For this example, all of the data is in days, however this can also be done in hours or in years, depending on the type of analysis platform the data will be utilized within. For the SAM PV-RPM feature, the data must be in days. The distribution parameters cannot be converted from hours to another time unit, so it's important to determine what time unit is necessary before making the calculations.

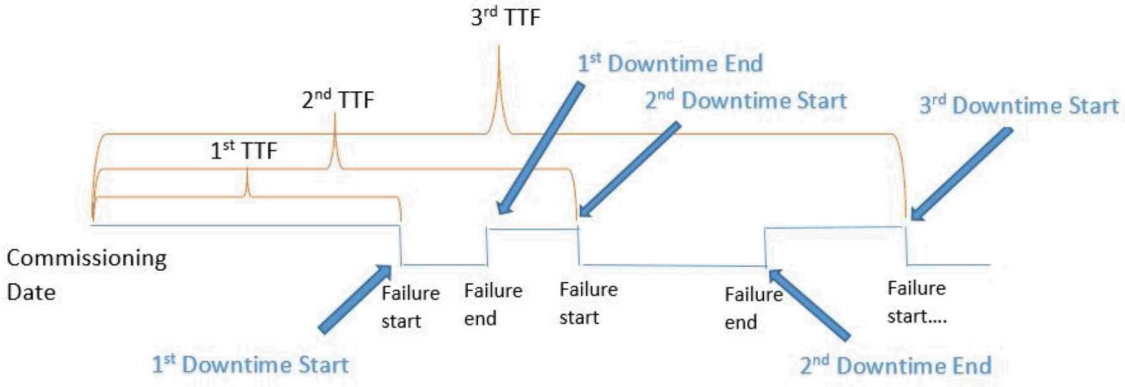


Figure A-11. Calculation of Time to Failure

To calculate the TTR, the difference between each failure end time and the associated failure start time is calculated as seen in Figure A-12. Both of these calculations for a hypothetical PV system are presented in Table A-2 as an example of how to take raw event data, in this case hypothetical inverter fan events, and develop the correct TTF or TTR for reliability probability distribution development.

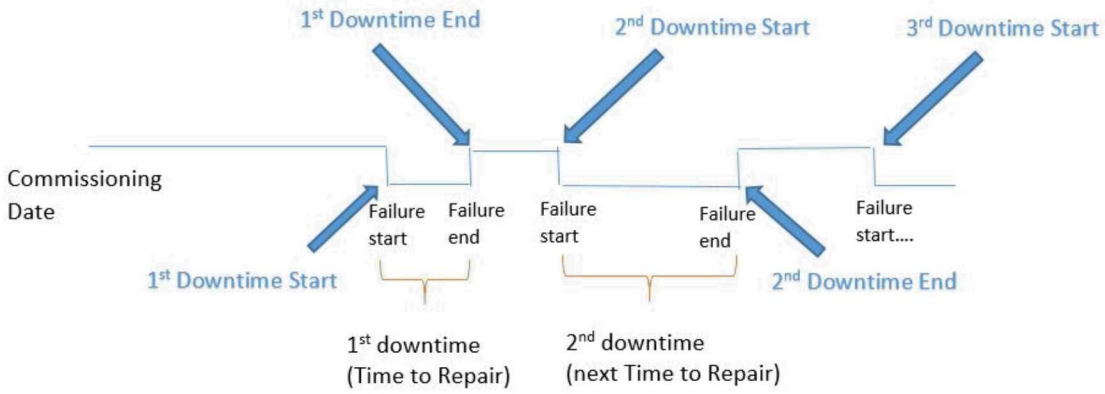


Figure A-12. Calculation of Time to Repair

Table A-2 – Example dataset for calculating the TTF and TTR

Event	Inverter Commissioning Date	Downtime Start	Downtime End	TTF (days) = Downtime Start – Commissioning Date	TTR(days) = Downtime End – Downtime Start
Fan failure	6/15/2016 0:00	6/30/2016 14:05	7/1/2016 23:59	=6/30/2016 14:05 - 6/15/2016 0:00 = 15.586	= 7/1/2016 23:59 - 6/30/2016 14:05 = 1.412
Fan failure		7/13/2016 13:15	7/13/2016 15:05	=7/13/2016 13:15 – 6/15/2016 0:00 = 28.552	= 7/13/2016 15:05 - 7/13/2016 13:15 = 0.076
Fan failure		7/14/2016 12:10	7/14/2016 14:46	=7/14/2016 12:10 – 6/15/2016 0:00 = 29.507	=7/14/2016 14:46 – 7/14/2016 12:10 =0.108

i – These are example times for a hypothetical failure mode

Once the TTF and TTR have been calculated, the best fit reliability distributions are developed by comparing the fit of some of the more commonly used distributions that approximate the faults and failures SNL has seen for PV systems. For TTF, these distributions include Weibull, Gamma and Exponential. For TTR these distributions include Normal, Lognormal and Exponential. For each distribution, the parameters of interest are listed in Table A-1. Probability plots are used to evaluate the fit of each distribution by estimating a cumulative distribution function through plotting the observation

against its estimated cumulative probability.[8] Using a program such as Minitab, Weibull++ or other software environment such as Matlab or Python, these plots are generated. To determine which distributions can be eliminated, visual inspection along with goodness-of-fit statistics are then evaluated. The examples below are from Minitab.

The Anderson-Darling statistic (AD) tests whether the sample data comes from a given distribution. For a 'good fit' the AD statistic should be less than one; however, to determine if one distribution is a better fit than another, the AD statistic should be significantly lower than the other distribution. In addition to the AD statistic, the probability value, or 'p-value,' is used. For some significance level α, (usually 0.05), a p-value $\leq \alpha$ indicates the data does not follow the distribution while a p-value $> \alpha$ indicates that the distribution should fail to be rejected. Generally, when comparing different distributions, the highest p-value will indicate the better fitting distribution. Visually one can use the probability plot to further determine if the distribution is a good fit by ensuring that the large majority of the points fall within the confidence intervals and the data follows the straight line of the plot.[9] Using a combination of these three goodness-of-fit evaluations, the best fit probability distribution can eventually be determined by a process of eliminating the distributions that are not a good fit to the underlying data.

As an example, we will consider the TTF to evaluate what failure distributions may have the best fit for the inverter fan failure data. A repair analysis will not be shown here, though following the same steps below would help define a repair distribution. It is assumed that all of the events occurred at one site, and impacted every one of the inverters. Figure A-13 shows the probability plots for each distribution of interest. The AD statistic for each distribution is greater than one and the p-values are all smaller than 0.05, both indicating that the data is not necessarily a good fit any of the distributions. Notice, in each plot there seems to be three different slopes to the data. Often, this is indicative of different underlying failure modes. Not every dataset will need to be separated, however it is important to check maintenance logs of different failure events to ensure that they are cataloged correctly. It may be that the root cause of the issue has not yet been determined and the resulting data may indicate that it needs to be broken up and re-analyzed.

Separating out each set of input data, we again consider the probability plots for each grouping as shown in Figure A-14 where Probability Plot 1 is the bottom slope and 3 is the top slope. The p-values and AD statistic for each plot are shown in Table A-3.

[8] http://support.minitab.com/en-us/minitab/17/topic-library/basic-statistics-and-graphs/graphs/graphs-of-distributions/probability-plots/probability-plot/

[9] http://blog.minitab.com/blog/adventures-in-statistics-2/how-to-identify-the-distribution-of-your-data-using-minitab

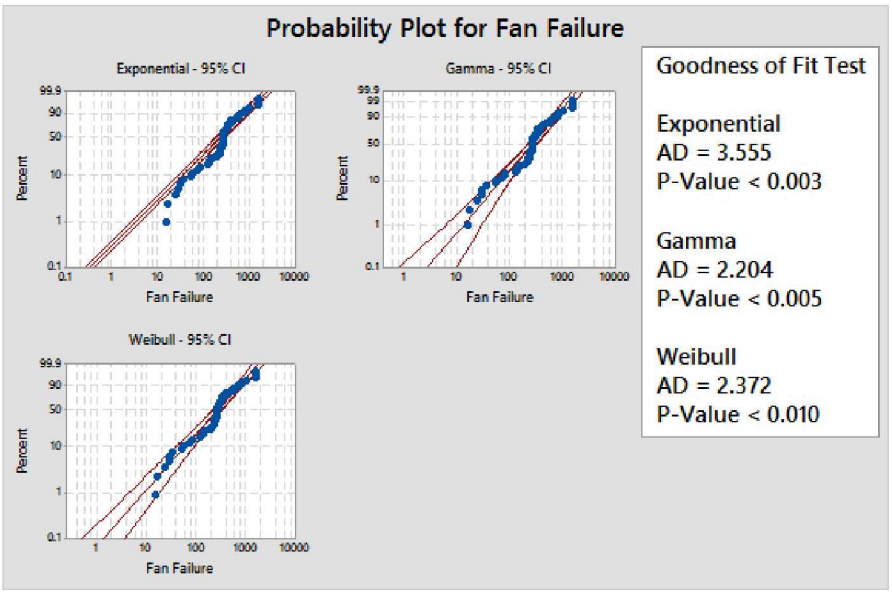


Figure A-13. Probability plots for Fan Failure in days

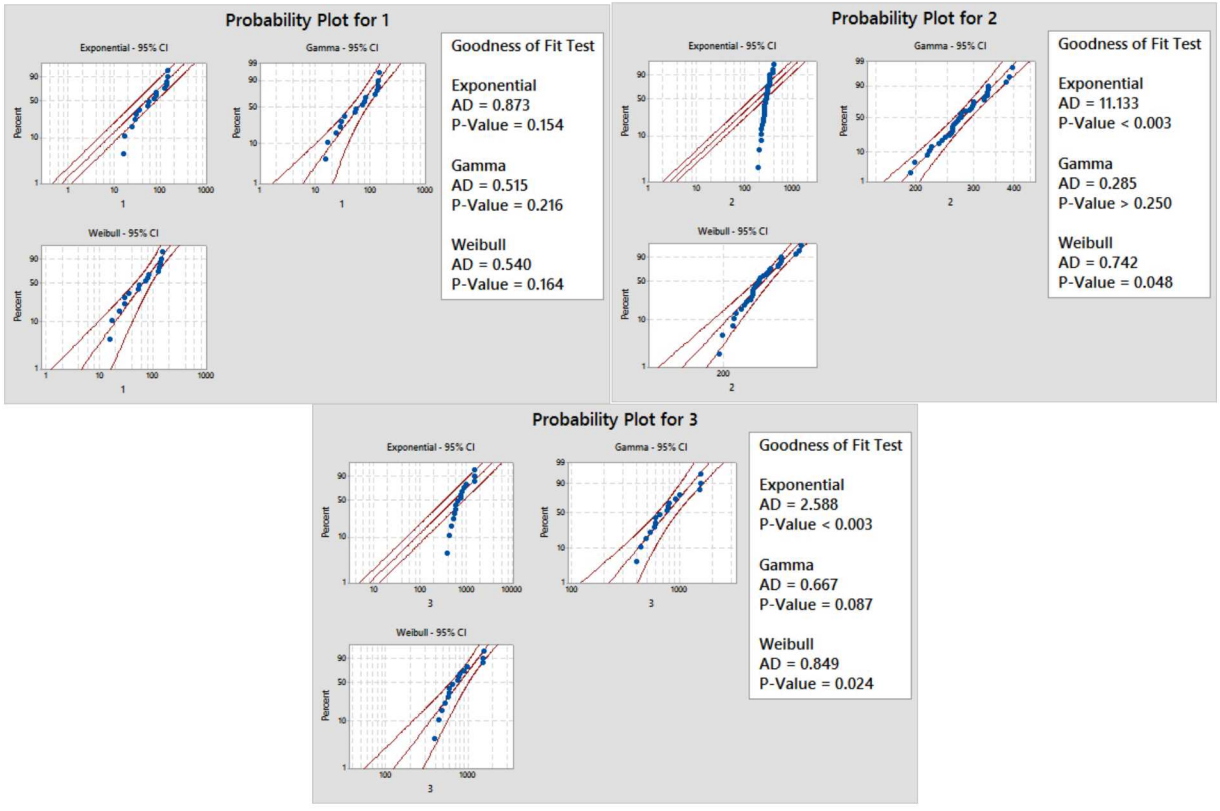


Figure A-14. Probability plots for separated slopes of Fan Failure Data

Table A-3 – Probability Plot Data from Figure A-15

Probability Plot 1			
Distribution	*Exponential*	*Weibull*	*Gamma*
AD statistic	0.873	0.540	0.515
p-value	0.154	0.164	0.216
Probability Plot 2			
Distribution	*Exponential*	*Weibull*	*Gamma*
AD statistic	11.133	0.742	0.285
p-value	<0.003	0.048	>0.250
Probability Plot 3			
Distribution	*Exponential*	*Weibull*	*Gamma*
AD statistic	2.588	0.849	0.285
p-value	<0.003	0.024	0.087

Consider probability plot 1. The exponential distribution can be eliminated first as it has the highest AD statistic and lowest p-value. Comparing the Weibull and Gamma distributions, both AD statistics are close so we rely on the larger of the two p-values to determine Gamma as the best fit distribution. For probability plot 2, the exponential distribution can again be eliminated right away as the data does not follow the cumulative distribution in the probability plot and does not stay within the confidence intervals. The Weibull distribution can also be eliminated as the p-value is lower than 0.05, leaving Gamma as the best fit. Using the same methodology, Gamma is determined to be the best fit for the third data set as well. Because each of the above smaller sets of data came from the TTF data for Fan Failure, it can be decided that Gamma is the best fit distribution for the Fan Failure. Again, using a program such as Minitab, the parameters for each smaller set of Fan Failure data are estimated.

Table A-4 – Gamma distribution data from best fit of each probability plot

	Alpha	Beta
Probability Plot 1	2.10	34.64
Probability Plot 2	35.76	7.75
Probability Plot 3	5.36	153.01

As mentioned above, the first plot in Figure A-13 was separated into three different plots in Figure A-14. The gamma parameters in Table A-4 could then be used to represent three different failure modes (at different life stages) if the fan failure data revealed that there were three distinct events that based on an analysis of the event data by an operations/inverter expert, for example.

References

Pham, H., (2006), “System Software Reliability,” Chapter 2 – System Reliability Concepts. Springer, 440 p.

Seiler, F.A., J.L. Alvarez (1996), “On the Selection of Distributions for Stochastic Variables,” Risk Analysis 16(1), p. 5-18.