Forty-Two Days in the SPA, Building a Stability Parameter Analyzer to Probe Degradation Mechanisms in Perovskite Photovoltaic Devices

Sean P. Dunfield[a†], Amy E. Louks[bc], Jay Waxse[a], Robert Tirawat[c], Steve Robbins[a], Joseph J. Berry[a], Matthew O. Reese[a]

[a]Materials Science Center, National Renewable Energy Laboratory, Golden, CO, 80401, USA.

[b]Chemistry Department, Colorado School of Mines, Golden, CO, 80401, USA.

[c]Chemistry and Nanoscience Center, National Renewable Energy Laboratory, Golden, CO, 80401, USA.

Figure S1. Representative sensor readings for light intensity, relative humidity, and temperature for samples at 25 °C in blue, 50 °C in green, 65 °C in orange, and 85 °C in red.

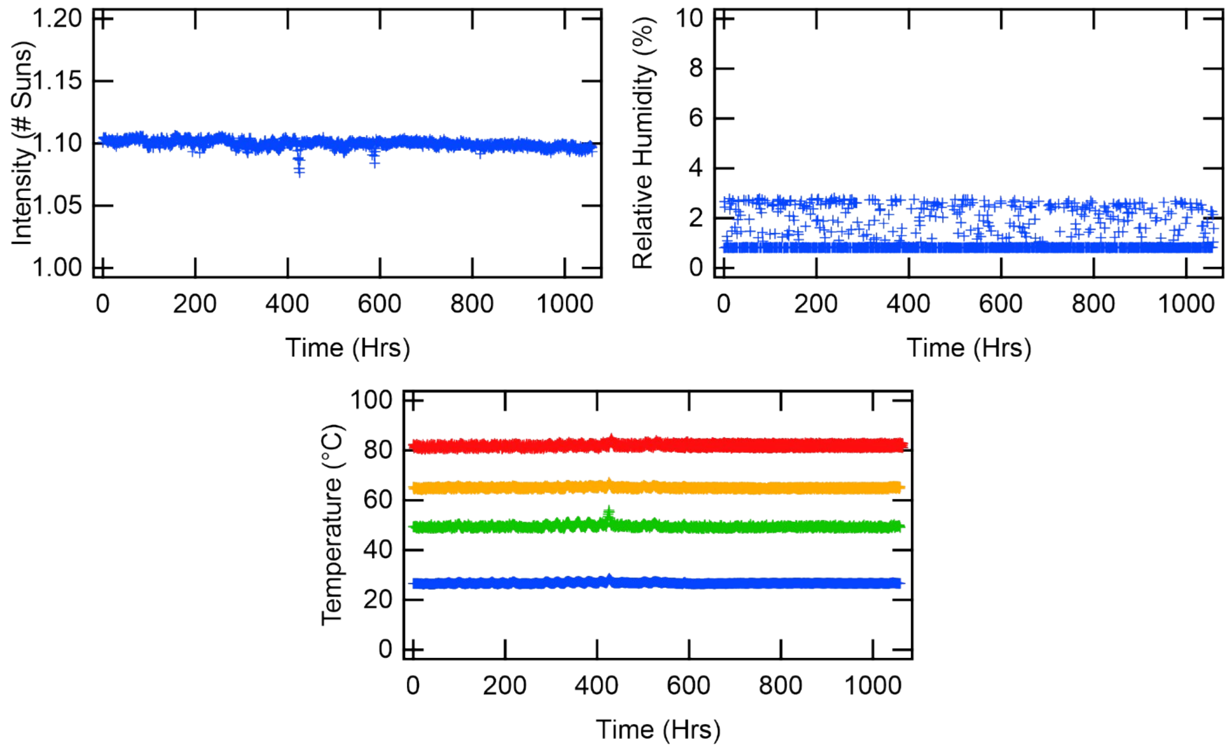
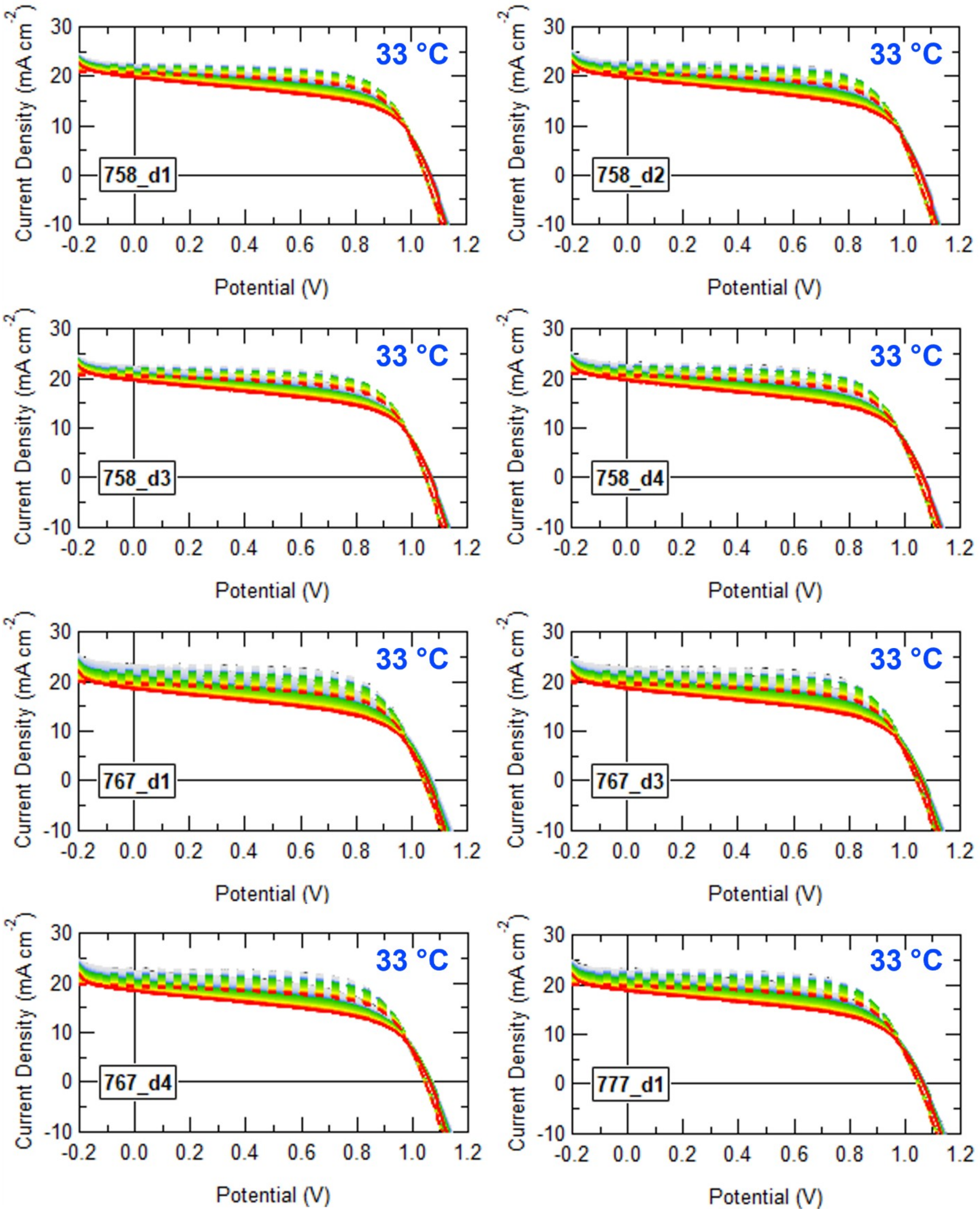
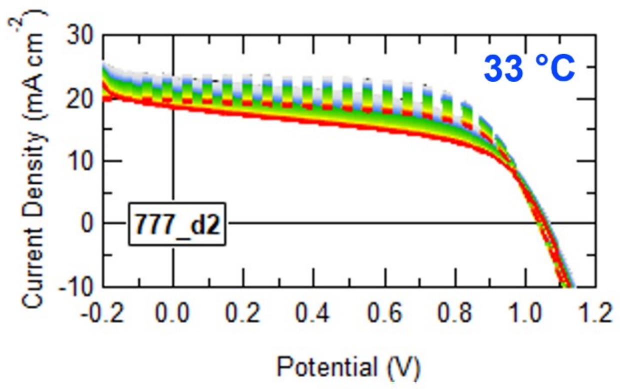
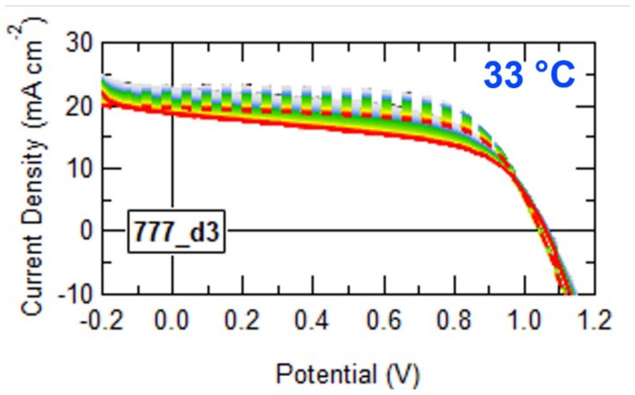


Figure S2. JV Curves for samples stressed at 33 °C. Forward scans are represented by the solid line and reverse scans by the dashed line.

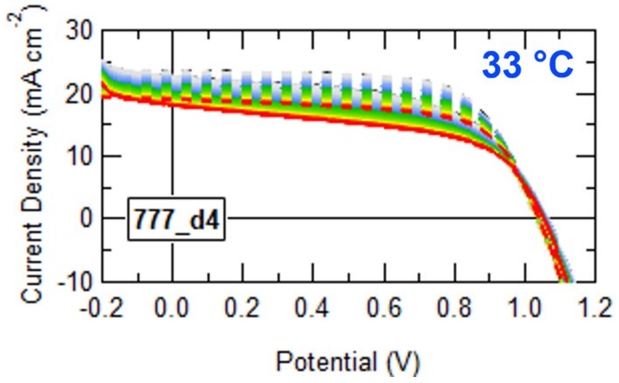




33 °C
777_d2
Current Density (mA cm^{-2})
Potential (V)

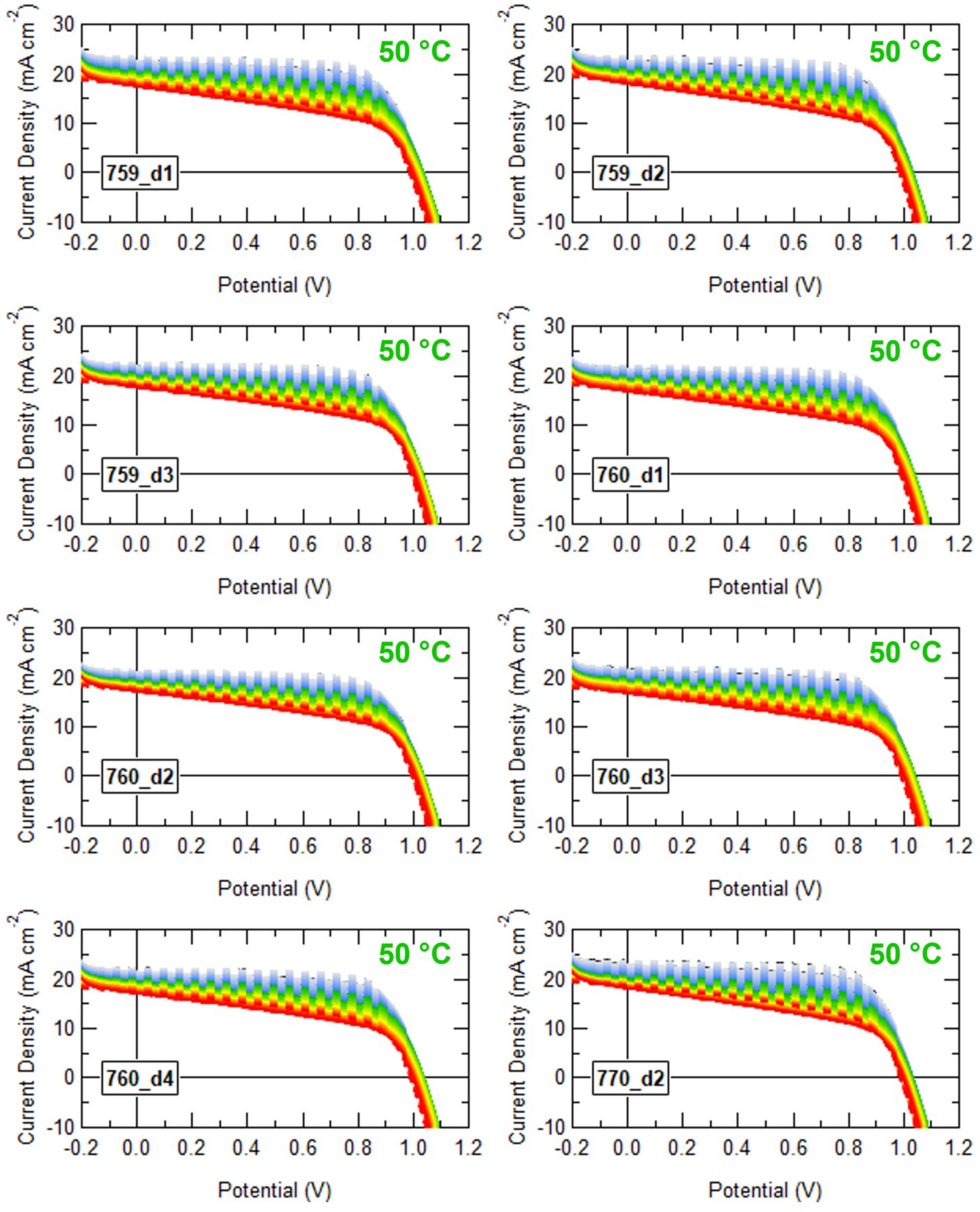


33 °C
777_d3
Current Density (mA cm^{-2})
Potential (V)



33 °C
777_d4
Current Density (mA cm^{-2})
Potential (V)

Figure S3. JV Curves for samples stressed at 50 °C. Forward scans are represented by the solid line and reverse scans by the dashed line.



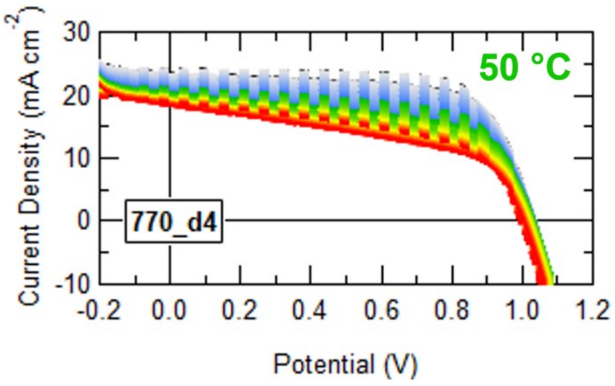
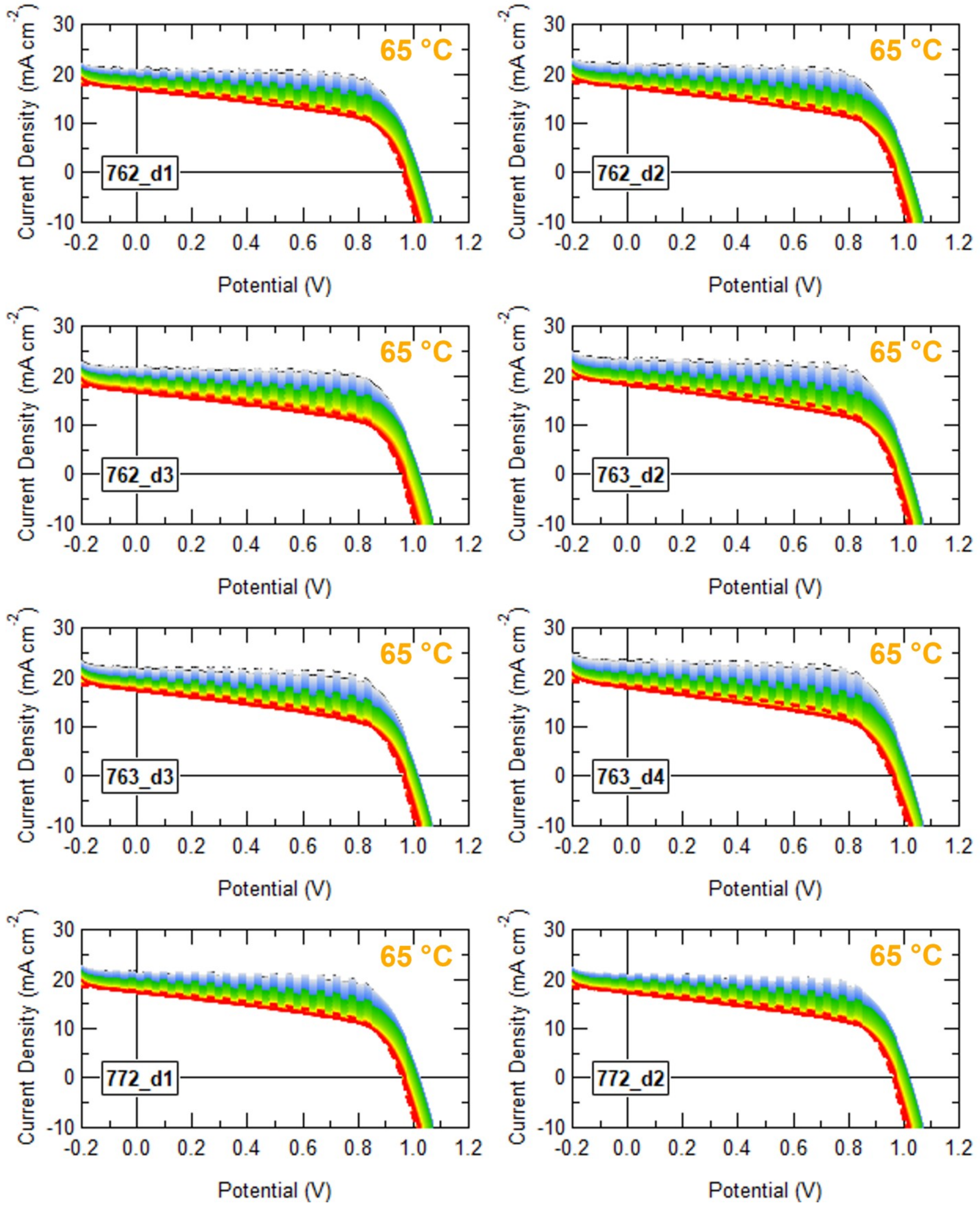

50 °C
770_d4
Current Density (mA cm^{-2})
Potential (V)
30
20
10
0
-10
-0.2
0.0
0.2
0.4
0.6
0.8
1.0
1.2

Figure S4. JV Curves for samples stressed at 65 °C. Forward scans are represented by the solid line and reverse scans by the dashed line.



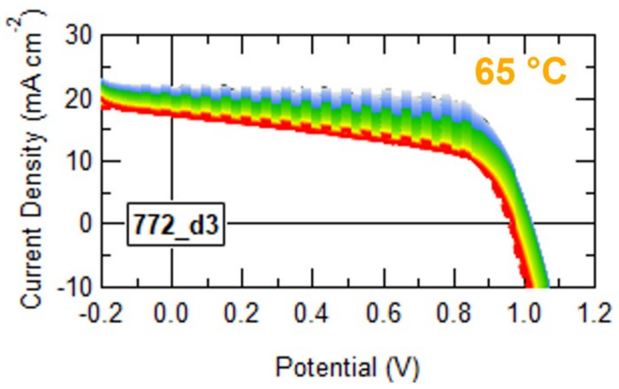

65 °C
772_d3
Current Density (mA cm^{-2})
Potential (V)
30
20
10
0
-10
-0.2
0.0
0.2
0.4
0.6
0.8
1.0
1.2

Figure S5. JV Curves for samples stressed at 85 °C. Forward scans are represented by the solid line and reverse scans by the dashed line.

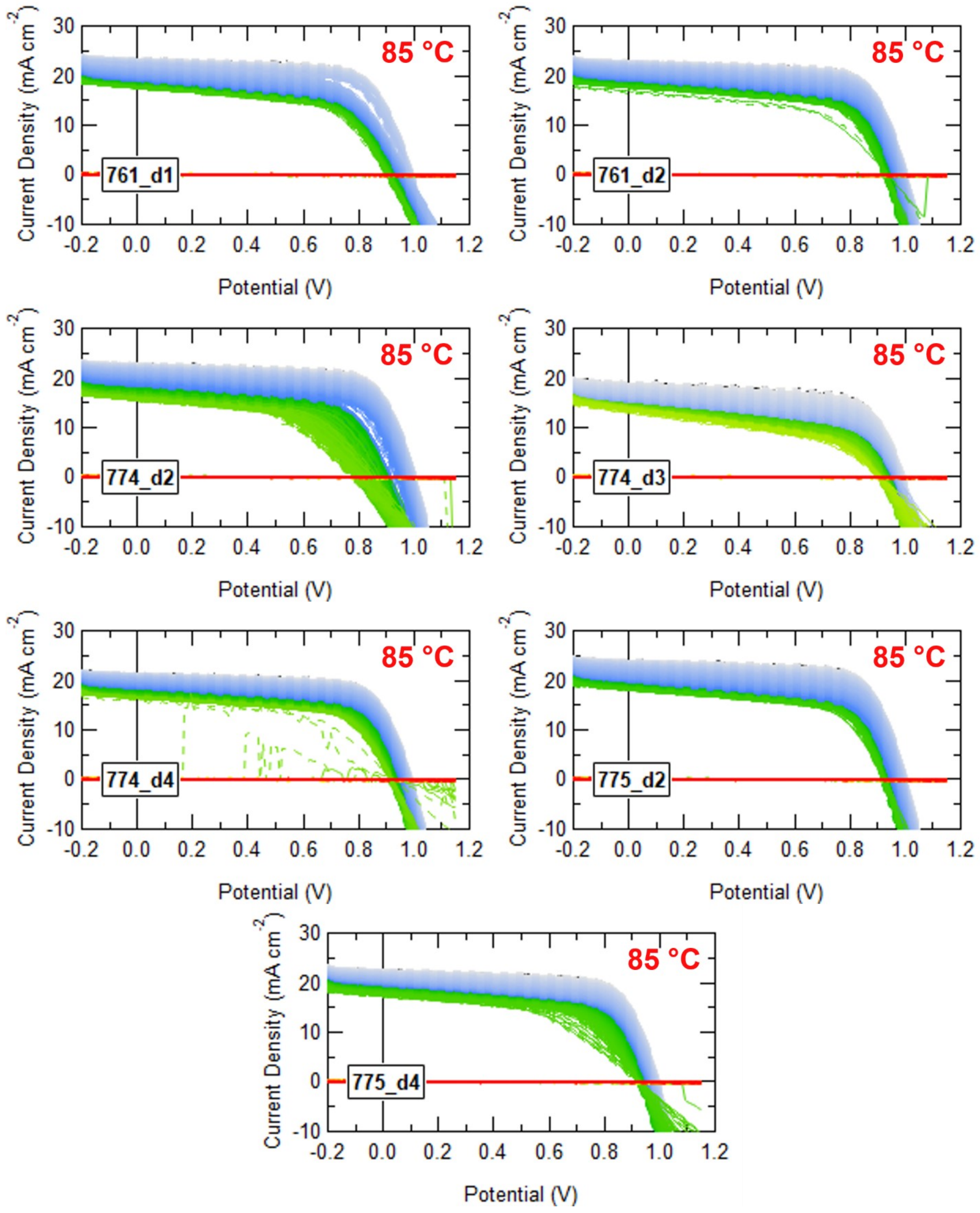


Figure S6. Forward and reverse power conversion efficiency (PCE), short circuit current density (Jsc), open circuit voltage (Voc), and fill factor (FF) for devices stressed at 33 °C. Individual data points for each pixel are represented by the green + signs, best fit trend lines for each pixel are represented by the light green line, averages for the entire data set are represented by the red line, and the standard deviation from the average is represented by the blue shading.

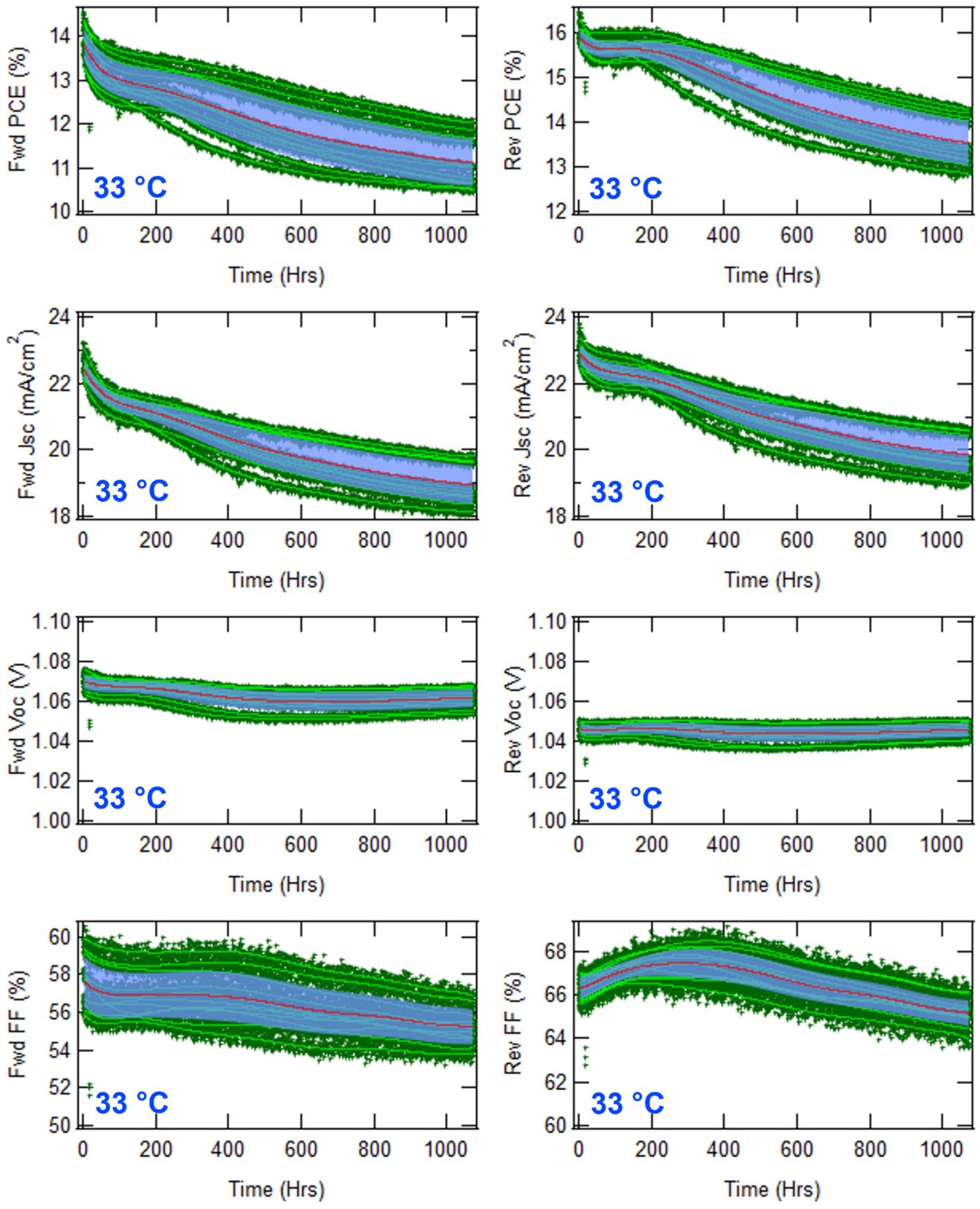


Figure S7. Forward and reverse power conversion efficiency (PCE), short circuit current density (Jsc), open circuit voltage (Voc), and fill factor (FF) for devices stressed at 50 °C. Individual data points for each pixel are represented by the green + signs, best fit trend lines for each pixel are represented by the light green line, averages for the entire data set are represented by the red line, and the standard deviation from the average is represented by the blue shading.

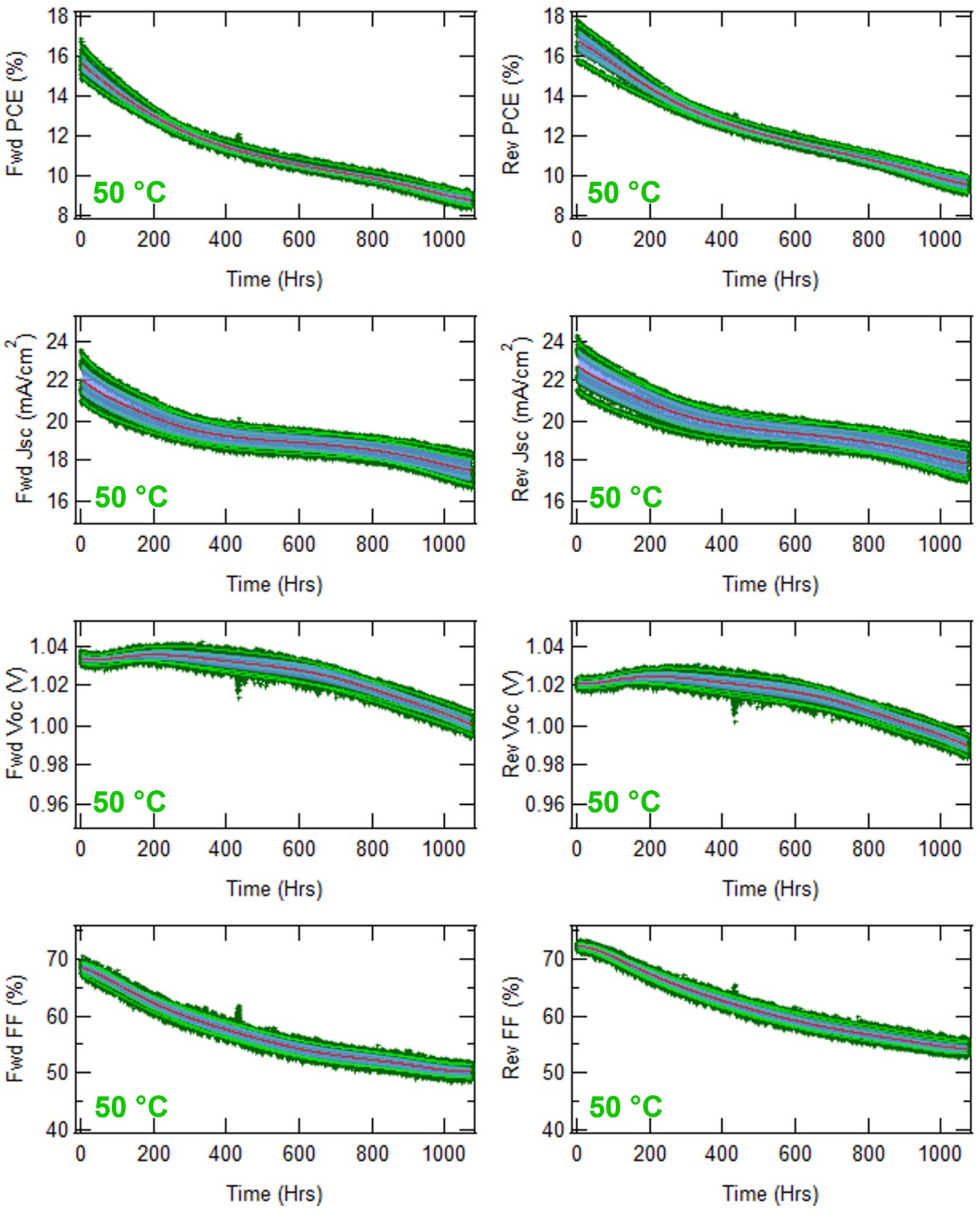


Figure S8. Forward and reverse power conversion efficiency (PCE), short circuit current density (Jsc), open circuit voltage (Voc), and fill factor (FF) for devices stressed at 65 °C. Individual data points for each pixel are represented by the green + signs, best fit trend lines for each pixel are represented by the light green line, averages for the entire data set are represented by the red line, and the standard deviation from the average is represented by the blue shading.

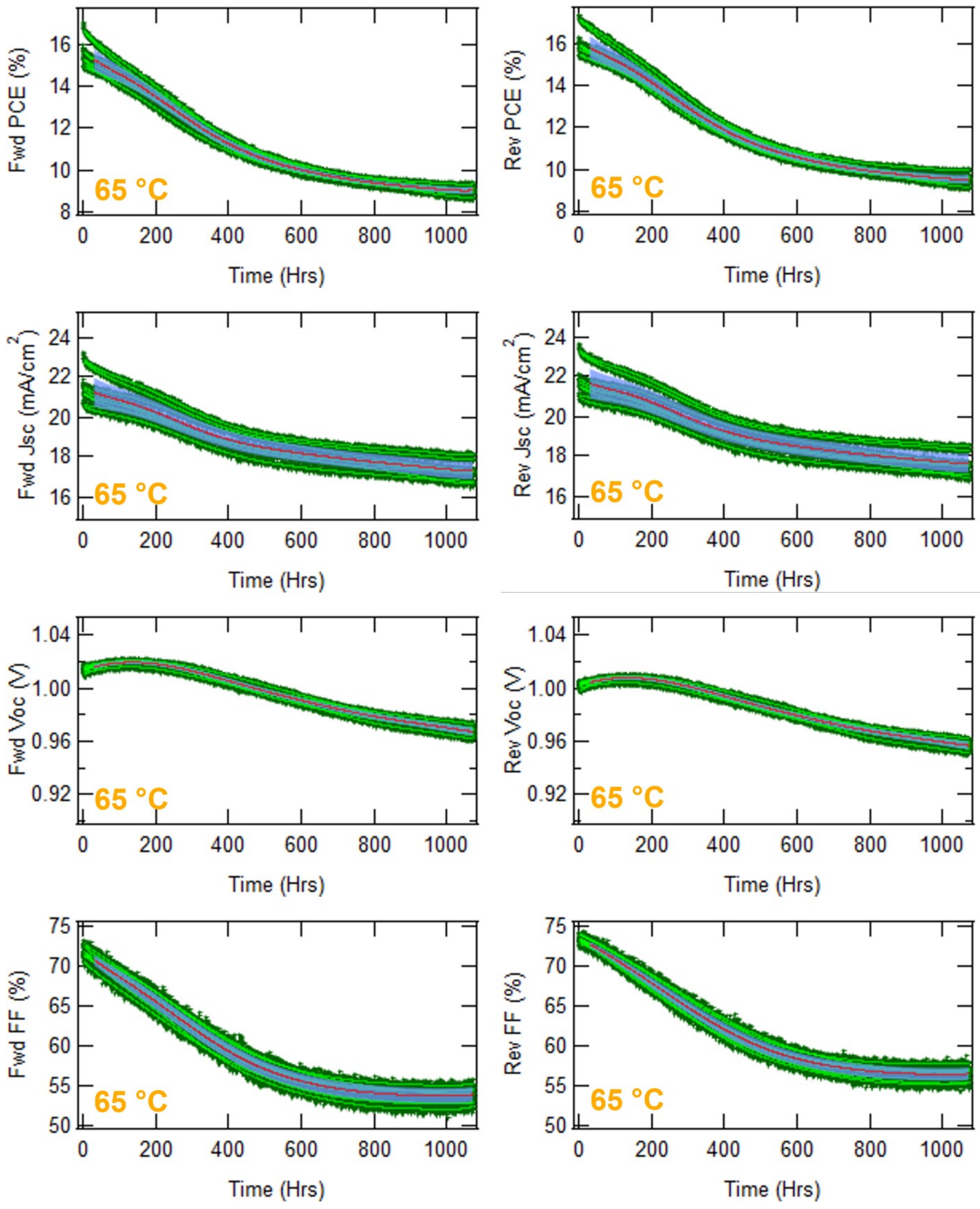


Figure S9. Forward and reverse power conversion efficiency (PCE), short circuit current density (Jsc), open circuit voltage (Voc), and fill factor (FF) for devices stressed at 85 °C. Individual data points for each pixel are represented by the green + signs, best fit trend lines for each pixel are represented by the light green line, averages for the entire data set are represented by the red line, and the standard deviation from the average is represented by the blue shading.

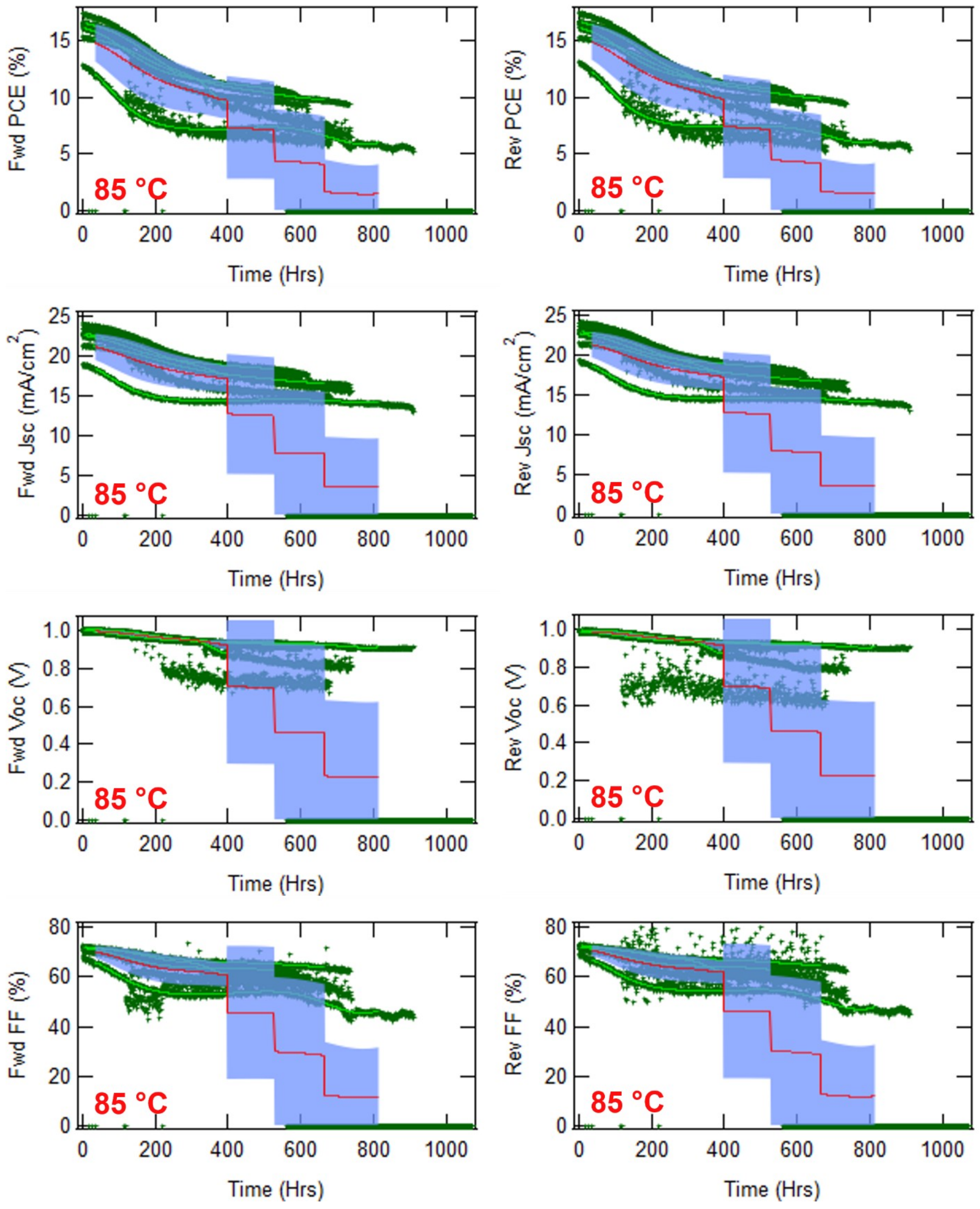


Figure S10. Normalized forward and reverse power conversion efficiency (PCE), short circuit current density (Jsc), open circuit voltage (Voc), and fill factor (FF) for devices stressed at 33 °C. Individual data points for each pixel are represented by the green + signs, best fit trend lines for each pixel are represented by the light green line, averages for the entire data set are represented by the red line, and the standard deviation from the average is represented by the blue shading.

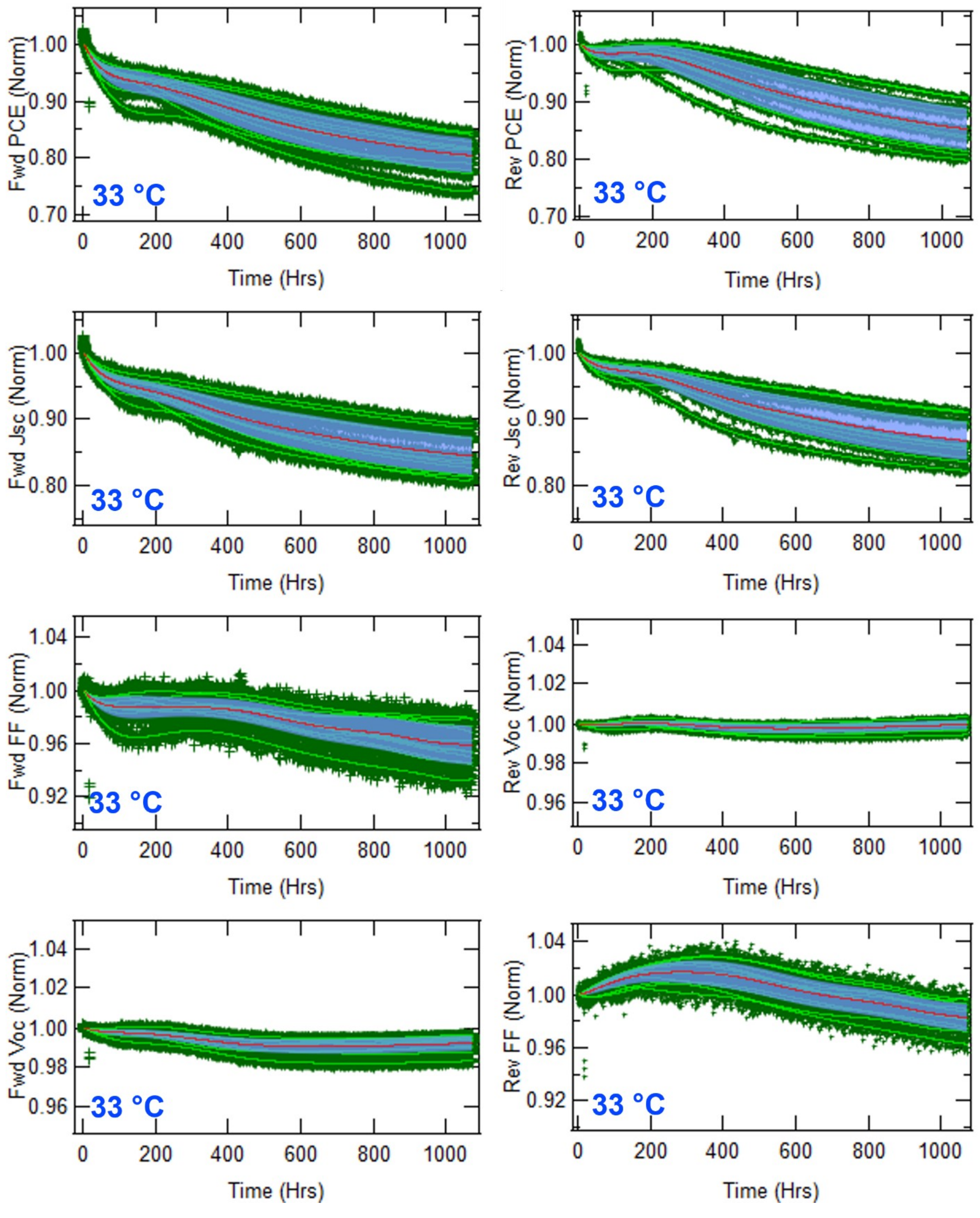


Figure S11. Normalized forward and reverse power conversion efficiency (PCE), short circuit current density (Jsc), open circuit voltage (Voc), and fill factor (FF) for devices stressed at 50 °C. Individual data points for each pixel are represented by the green + signs, best fit trend lines for each pixel are represented by the light green line, averages for the entire data set are represented by the red line, and the standard deviation from the average is represented by the blue shading.

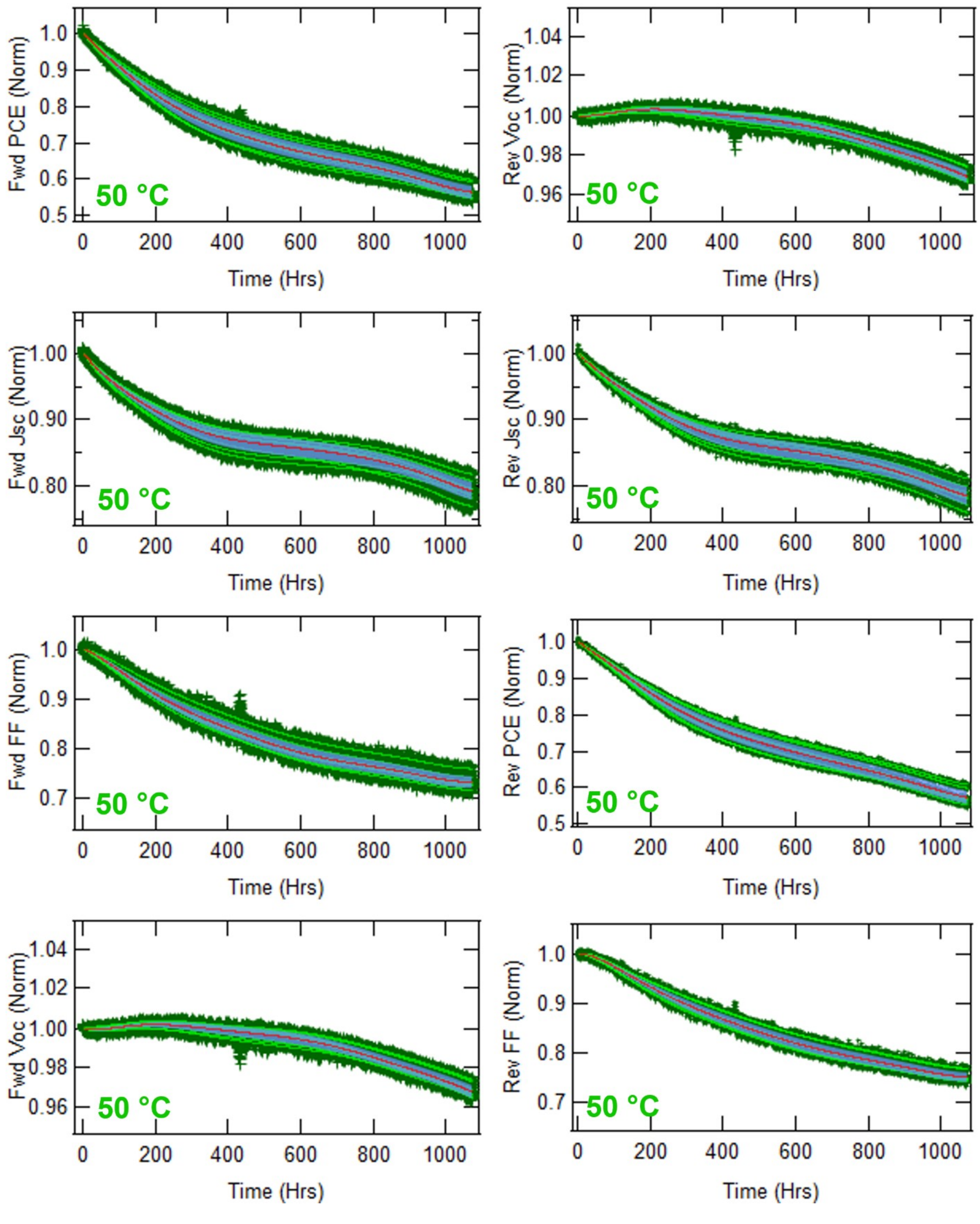


Figure S12. Normalized forward and reverse power conversion efficiency (PCE), short circuit current density (Jsc), open circuit voltage (Voc), and fill factor (FF) for devices stressed at 65 °C. Individual data points for each pixel are represented by the green + signs, best fit trend lines for each pixel are represented by the light green line, averages for the entire data set are represented by the red line, and the standard deviation from the average is represented by the blue shading.

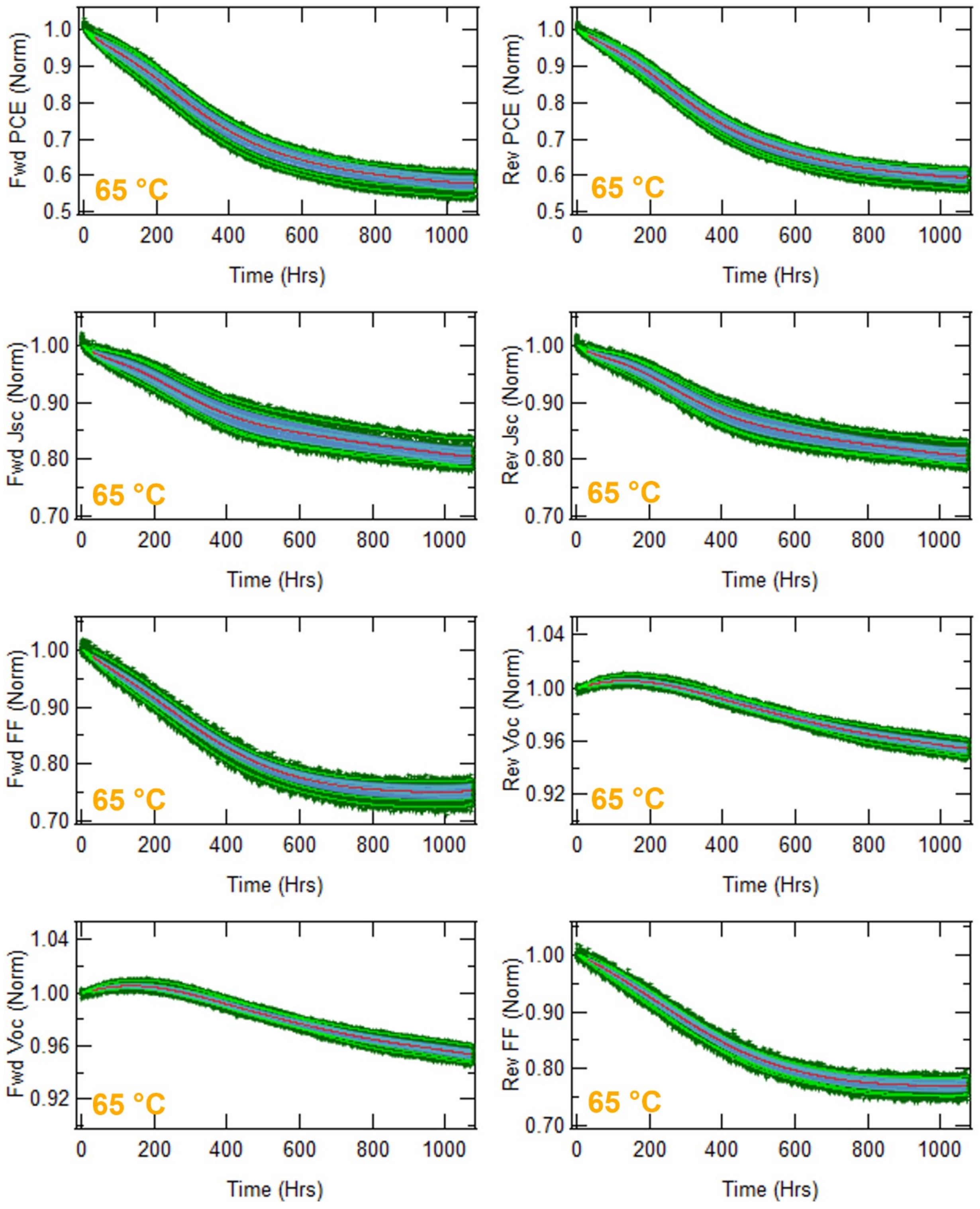


Figure S13. Normalized forward and reverse power conversion efficiency (PCE), short circuit current density (Jsc), open circuit voltage (Voc), and fill factor (FF) for devices stressed at 85 °C. Individual data points for each pixel are represented by the green + signs, best fit trend lines for each pixel are represented by the light green line, averages for the entire data set are represented by the red line, and the standard deviation from the average is represented by the blue shading.

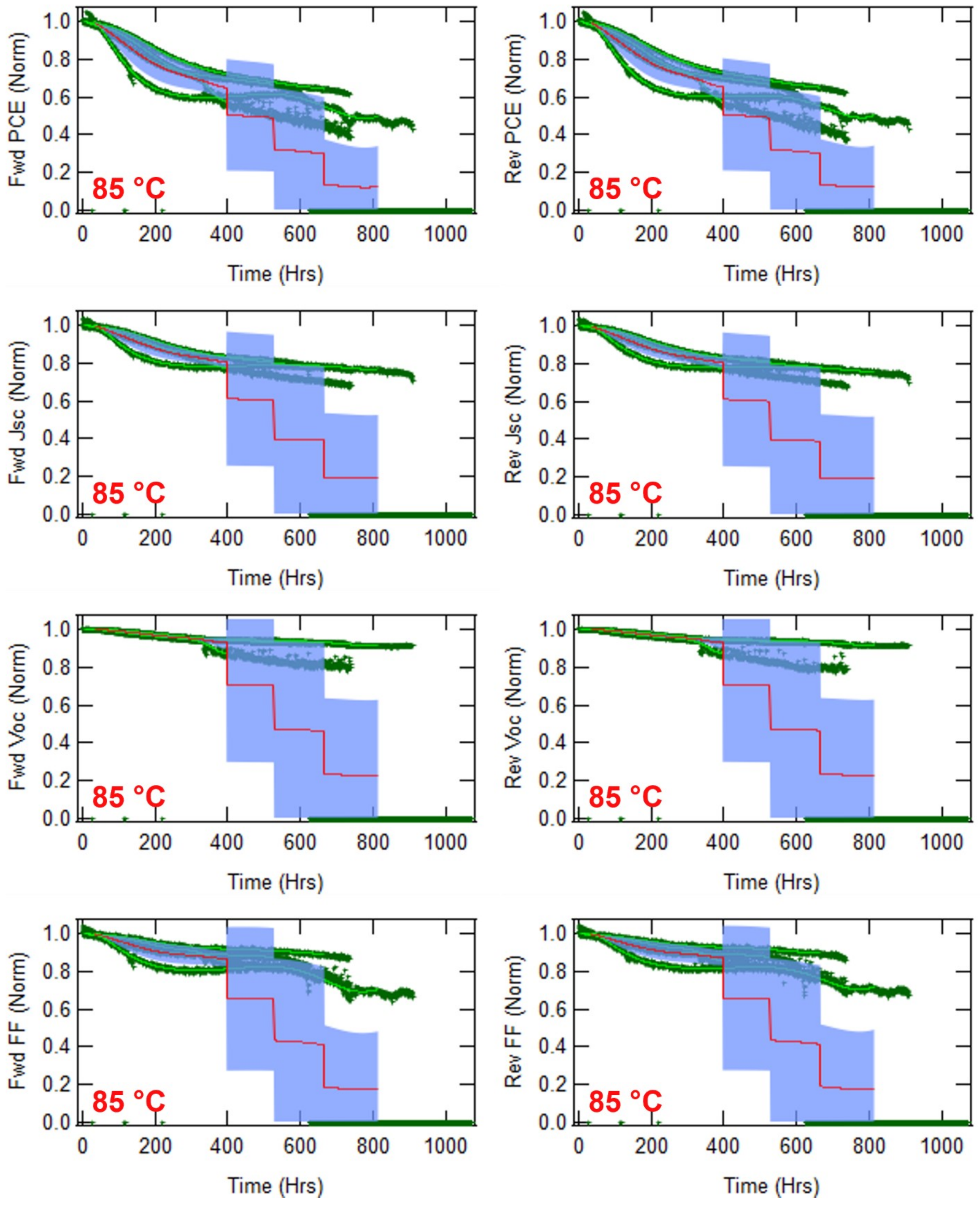


Figure S14. Hysteresis versus time for devices aged at 33 °C (blue), 50 °C (green), 65 °C (yellow), and 85 °C (red). Hysteresis is calculated by subtracting the reverse from the forward wave.

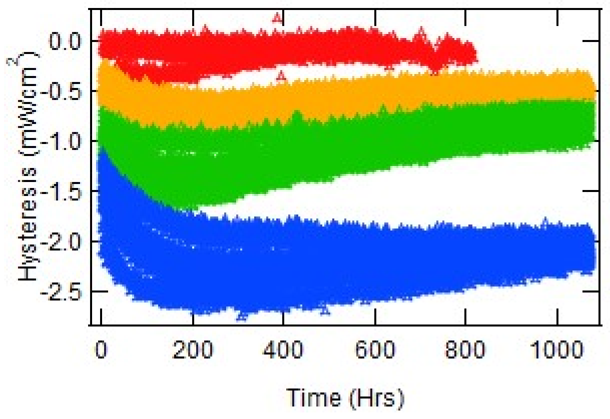
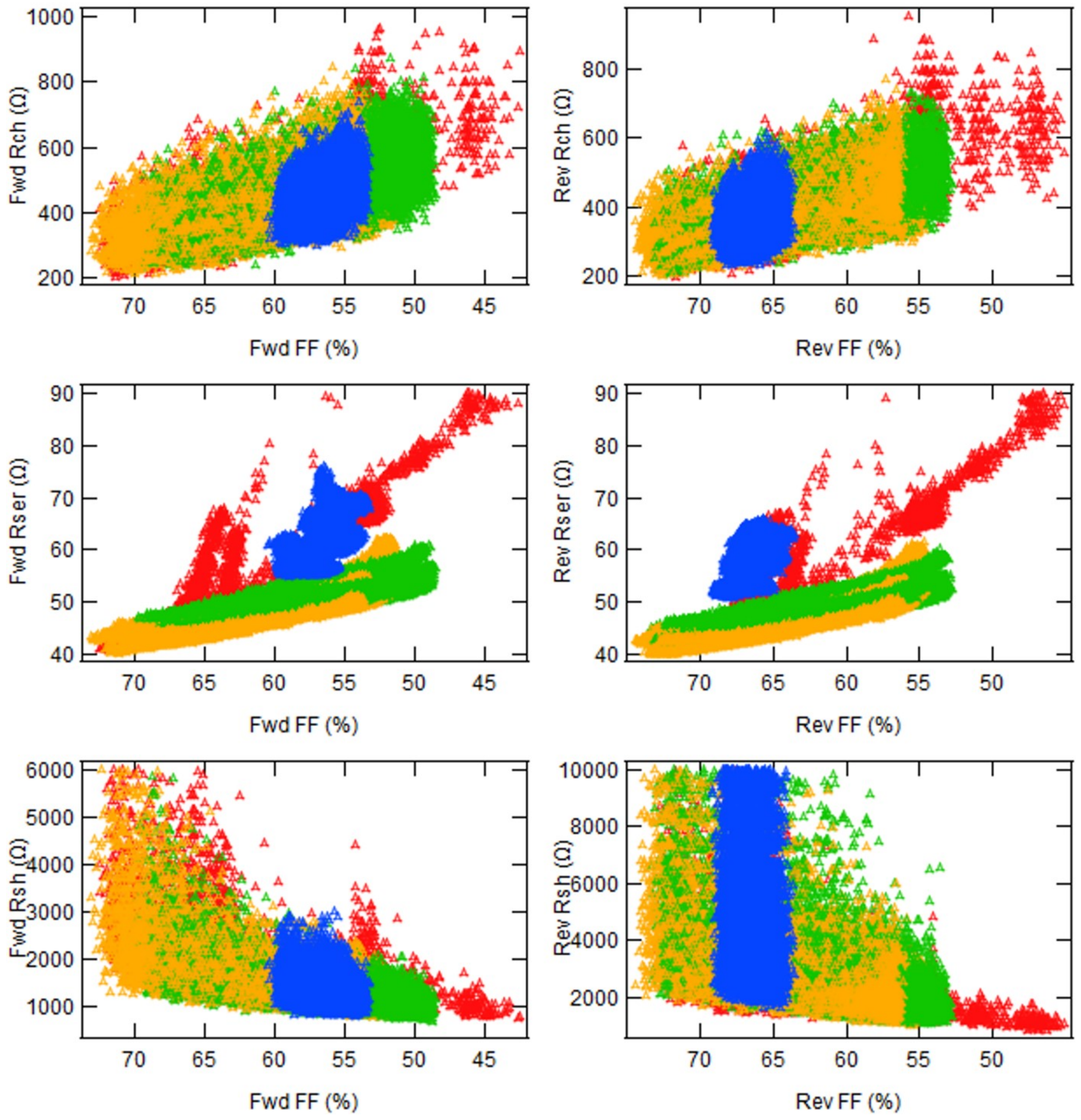


Figure S15. Forward and reverse series (Rser), shunt (Rsh), and characteristic (Rch) resistances versus fill factor (FF) for devices aged at 33 °C (blue), 50 °C (green), 65 °C (yellow), and 85 °C (red).



Supplementary Note 1

Main User Interface (Figure S16)

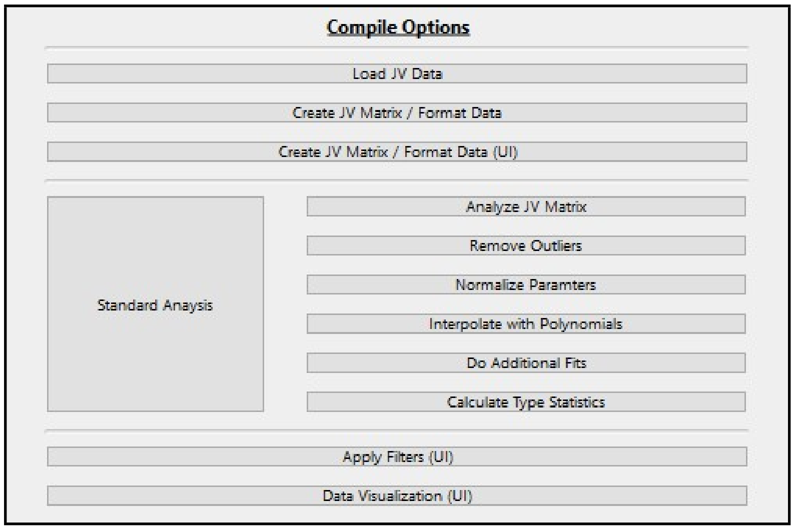


Figure S16. Main User Interface for degradation program.

The general UI has 5 main functions including:

1) Load IV Data
2) Data Formatters
    - 2a) Create JV Matrix / Format Data (Automated)
    - 2b) Create JV Matrix / Format Data (UI)
3) Standard Analysis
    - 3a) Analyze JV Matrix
    - 3b) Remove Outliers
    - 3c) Normalize Parameters
    - 3d) Interpolate with Polynomials
    - 3e) Do Additional Fits
    - 3f) Calculate Type Statistics
4) Apply Filters (UI)
5) Data Visualization (UI)

Function (1) "Load JV Data": has been left blank as we could not anticipate the vast number of possible data formats used by groups. The user should navigate to the function "**Load_JV_Data_Button()**" in the procedure file "**DegHawk:DeghawkGUI.py**" to add a macro that properly loads the desired IV or JV over time data.

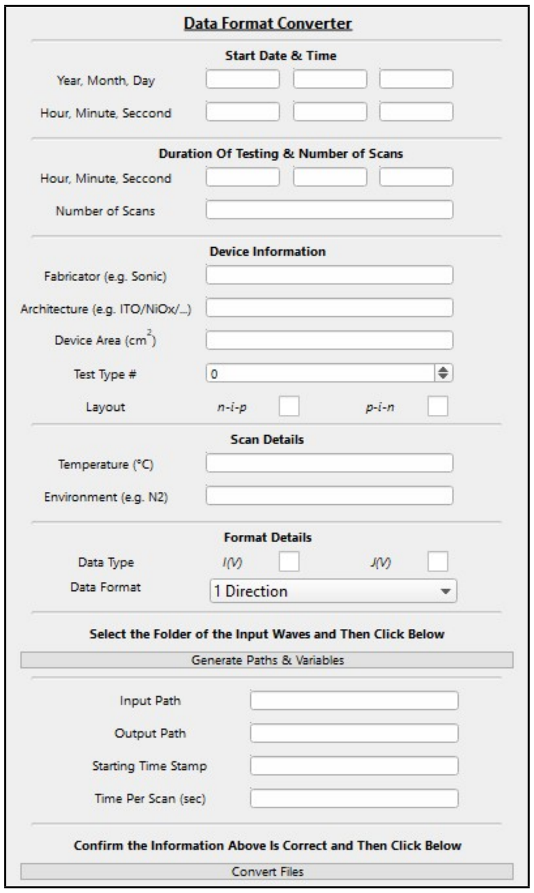
At the end of this function you should have the raw data loaded into Igor.

Function (2a) "Create JV Matrix / Format Data": has been left blank as we could not anticipate the vast number of possible data formats used by groups. The user should navigate to the function

"**Create_JV_Matrix_Button()**" in the procedure file "**DegHawk:DeghawkGUI.py**" to add a macro that properly formats the data or use the UI provided below for a more general approach.

At the end of this function you should have J(V) waves that are chronologically ordered in a folder with the device name.

Function (2b) "Create JV Matrix / Format Data (UI)": launches a UI that asks for the information necessary to sort and format various forms of I(V) or J(V) data in chronological order into one that is useable by the program (**Figure S17**).



Test Type # 0

Data Format 1 Direction

Figure S17. UI for converting loaded data into a form useable by the software package.

This UI creates the folder hierarchy "root:<Fabricator>:x<Start Date>:<Device Type #>:<Device ID>" (e.g. root:Sonic:x200710:Type1:SPD_001_d1) using the information input in the UI. Then, it creates "**Device_Info**" in the type folder (e.g. root:Sonic:x200710:Type1:Device_Info) to store important information regarding the device architecture and type of degradation test run, and "**JV_Waves**" in the device folder (e.g. root:Sonic:x200710:Type1:SPD_001_d1:JV_Waves) to store the devices degradation data. The former wave, **"Device_Info",** is a text matrix where the first column corresponds to the device's orientation (p-i-n or n-i-p) and its architecture/device layers while the second column refers to the test conditions utilized (device area, environment, temp). The latter wave, "**JV_Waves**" is a V, J_1, J_2, J_3, J_n numerical matrix where each of the current headers follow the format:

x<year><month><day>_<hour><minute><second>_Current_<direction> (e.g. x200710_142334_Current _Fwd).

*At the end of this function you should have the required folder hierarchy for analysis (root:<Fabricator>:<date>:<type#>:DevID), "**JV_Waves**" in the device folder, and "**Device_Info**" in the type folder filled out and complete.*

Function (3) "Standard Analysis": uses "**JV_Waves**", "**Device_Info**", and the hierarchy created in the last step to conduct our standardized analysis using functions 3a) to 3f). Note that each function performed in the standard analysis has been broken into separate functions so that any individual step can be replaced. Overall, this function controls all calculations done within the kit (additional functions are for visualization and grouping of the data). The waves created by this function are below:

At the end of this function there should be the following set of waves in every device folder:

"JV_Waves"	*Contains all JV waves*
"Param_v_Time"	*Contains all parameters extracted from the JV waves*
"Diff_v_Volt"	*Contains the difference between the forward and reverse wave for each time stamp*
"Ideality_v_Volt"	*Contains the voltage dependent ideality factor for each time stamp*
"Param_v_Time_Used"	*Contains a subsection of "**Param_v_Time**" where outlier points have been removed*
"Param_FitCoeff"	*Contains t_start, t_end, chi2 (χ^2), and fitting coefficients for linear, 4th degree polynomial, 10th degree polynomial, exponential, and double exponential fits for each main parameter (J_{sc}, V_{oc}, FF, PCE)*
"Param_Fits"	*Uses the 10 degree polynomial fit above to reproduce main parameter fits from t=0 to t=t_end. Data points which do not make sense or are after the last point/before the first point are set to NaN. This creates waves that are smooth and easy to compare with uniform time spacing of 1 hour.*
"Param_Fits_v_PCE"	*Calculates the FF, J_{sc}, V_{oc} for a subsection of PCE values (2.5% steps from %100 to %75) using the above waves*
"Param_Fits_v_PCE_1D"	*1D version of the above wave, used for other analysis*
"Param_Fits_v_Time"	*Calculates the FF, J_{sc}, V_{oc} for a subsection of time values (Max, 10, 100, 200, 500, 750, 1000 hours) using the above waves*

"Param_Fits_v_Time_1D"	*1D version of the above wave, used for other analysis*
"OnePlot"	*Contains % loss in PCE as a function of losses in FF, J_{sc}, and V_{oc} represented as independent variables*

And the following set of waves in every Type Folder:

*"**Type_Info**"*	*first column corresponds to the devices orientation (p-i-n or n-i-p) and its architecture/device components while the second column refers to test conditions (device area, environment, temp)*
"Param_Fits_Avg"	*This wave is the same as "**Param_Fits**" but has averages for each of the waves*
"Param_Fits_StdDev"	*This wave is the same as "**Param_Fits**" but has standard deviations for each of the values*
"OnePlot"	*This is the same as "**OnePlot**" in the device folder, but is calculated from "**Param_Fits_Avg**"*

Function (3a) "Analyze JV Matrix": analyzes the "**JV_Waves**" matrix made by "**Launch Data Formatting UI.**" The main part of this function obtains the short circuit current density (J_{sc}), open circuit voltage (V_{oc}), fill factor (FF), power conversion efficiency (PCE), max power point voltage (V_{mp}), max power point current (J_{mp}), max power point power (P_{mp}), shunt resistance (R_{sh}), series resistance (R_s), and characteristic resistance (R_{ch}) over time, storing the information in "**Param_v_Time**". Each standard parameter is calculated as described below:

J_{sc}	Short Circuit Current Density	mA/cm^2	y-intercept of the JV curve (max current)
V_{oc}	Open Circuit Voltage	V	x-intercept of the JV curve (max voltage)
FF	Fill Factor	%	P_{mp} divided by J_{sc} and V_{oc}
PCE	Power Conversion Efficiency	%	V_{oc} * J_{sc} * FF
V_{mp}	Max Power Point Voltage	V	Value of voltage at the max power point
J_{mp}	Max Power Point Current	mA/cm^2	Value of current at the max power point
P_{mp}	Max Power Point Power	mW/cm^2	Value of power at the max power point. Max of I(V)*V
R_{ser}	Series Resistance	Ω	Inverse slope of the JV curve at V_{OC} (uses 0.05 V before and after)
R_{sh}	Shunt Resistance	Ω	Inverse slope of the JV curve at J_{SC} (uses

			0.05 V before and after)
R_{ch}	Characteristic Resistance	Ω	Inverse slope of the JV curve at maximum power point (uses 0.05 V before and after)

The kit also creates two additional matrix waves for analysis. The first, **"Diff_v_Volt"** calculates the difference between the forward and reverse scans and lists them in a V, J_1, J_2, J_3… matrix where the columns have time stamps. The second, "**Ideality_v_Volt**" calculates the voltage dependent ideality factor at each point assuming a modified ideal diode equation:

$$J(V) \approx J_{sc} - J_0\left(\exp\left(\frac{qV}{nkT}\right) - 1\right)$$
$$J_{sc} - J(V) \approx J_0\left(\exp\left(\frac{qV}{nkT}\right) - 1\right)$$

At values V > 0.1, the exponential term can be assumed to dominate, therefore:

$$J_{sc} - J(V) \approx J_0 \exp\left(\frac{qV}{nkT}\right)$$
$$\ln\left(J_{sc} - J(V)\right) \approx \ln\left(J_0 \exp\left(\frac{qV}{nkT}\right)\right)$$
$$\ln\left(J_{sc} - J(V)\right) \approx \ln\left(J_0\right) + \ln\left(\exp\left(\frac{qV}{nkT}\right)\right)$$
$$\frac{\partial}{\partial V}\left[\ln\left(J_{sc} - J(V)\right)\right] \approx \frac{\partial}{\partial V}\left[\ln\left(J_0\right) + \frac{qV}{nkT}\right]$$
$$\frac{\partial}{\partial V}\left[\ln\left(J_{sc} - J(V)\right)\right] \approx \frac{\partial}{\partial V}\left[\frac{qV}{nkT}\right]$$
$$\frac{\partial}{\partial V}\left[\ln\left(J_{sc} - J(V)\right)\right] \approx \frac{q}{nkT}$$
$$n(V) = \frac{q}{kT}\left(\frac{\partial Ln(J(V))}{\partial V}\right)^{-1}$$

and lists them in a V, J_1, J_2, J_3… matrix where the columns have time stamps and directions. These two matrices and a few other functions are then used to fill out additional columns on the "**Param_v_Time**" matrix, including:

Hysteresis_Voc_Jsc	Calculates the difference in the forward and reverse scans between J_{sc} and V_{oc}
n_at_Jsc_<direction>	Calculates ideality factor at J_{sc}
n_at_Voc_<direction>	Calculates ideality factor at V_{oc}
n_at_Vmp_<direction>	Calculates ideality factor at V_{mp}

v_bias_<direction>	Calculates the bias voltage of the device assuming a 510 ohm resistor.
V_diff_<direction>	Calculates difference between the bias voltage and the maximum power point of the device

It also creates a "T_80 column" (time in hrs it took for device to reach 80% of its max PCE) in the "**Param_v_Time**" matrix that is not filled out until step 3c and duplicates "**Param_v_Time**" to create a "**Param_v_Time_Used**" matrix. In 3b, this matrix will be modified to remove outliers, however putting the duplication step here allows 3b to be skipped if desired (although this is not recommended).

*At the end of this function you should also have **"Diff_v_Volt"** and **"Ideality_v_Volt"** filled out and complete in the Device Folder. "**Param_v_Time**" should exist and have all non-normalized values filled out. There should also be **"Param_v_Time_Used"**, which at this point should be identical to **"Param_v_Time"**.*

Function (3b) "Remove Outliers": This step removes outliers using a method that was optimized on a subsection of our data to produce ***"Param_v_Time_Used"***. It can be skipped if desired, although it is not recommended. To find the bad points the code uses a multistep algorithm.

Step 1, check the standard device metrics in ***"Param_v_Time"*** that were calculated in the last step (J_{sc}, V_{oc}, FF, PCE) to ensure they have realistic values. If the value for any parameter is infinity, NaN, or less than 0, all values for that time step are set to NaN in ***"Param_v_Time"***.

Step 2, average the **"PCE_Fwd"** and **"PCE_Rev"** columns in ***"Param_v_Time"***. Set the start point as the first point that has a slope less than the average magnitude slope, throwing out data were the PCE of the device is increasing rapidly. Normalize the data by the maximum value from the start point to 24 hours after that point. Set the endpoint as the first point where the PCE drops below 1% of its maximum value, throwing out data where the device is too close to dead to care about its degradation.

Step 3, filter through the "***Param_v_Time***" wave from the start value to end value (both determined above). Calculate the points which may be dropouts by looking at the first/second derivatives of each parameter versus time wave. Make the dropouts NaN in "***Param_v_Time_Used.***" More specifically, this section looks at J_{sc}, V_{oc}, FF, and PCE scans to determine dropouts. For each of the parameters, it creates three waves. These waves hold the derivative in the forward and reverse directions as well as the absolute value of the centered double derivative, i.e.:

$$f'(x)_{FWD} = f(i) - f(i+1)$$
$$f'(x)_{REV} = f(i-1) - f(i)$$
$$|f''(x)| = |f'(i)_{REV} - f'(i+1)_{REV}| = |f(i-1) - 2 * f(i) + f(i+1)|$$

Then, for each wave it calculates the average, denoted μ, and standard deviation, denoted σ. A point is flagged as a potential bad dropout forward/reverse point if the following set of conditions are met:

1) f'(x) must satisfy one of the two following conditions:

$$f'(x)_{FWD/REV} > \mu[f^{'}(x)_{FWD/REV}] + 2 * \mu[f^{'}(x)_{FWD/REV}]$$
$$f'(x)_{FWD/REV} < \mu[f^{'}(x)_{FWD/REV}] - 2 * \mu[f^{'}(x)_{FWD/REV}]$$

2) f''(x) must satisfy the following condition:

$$f^{''}(x) > \mu[f^{''}(x)] + \sigma[f^{''}(x)]$$

This flags the points before and after the dropout, as shown below (**Figure S18**):

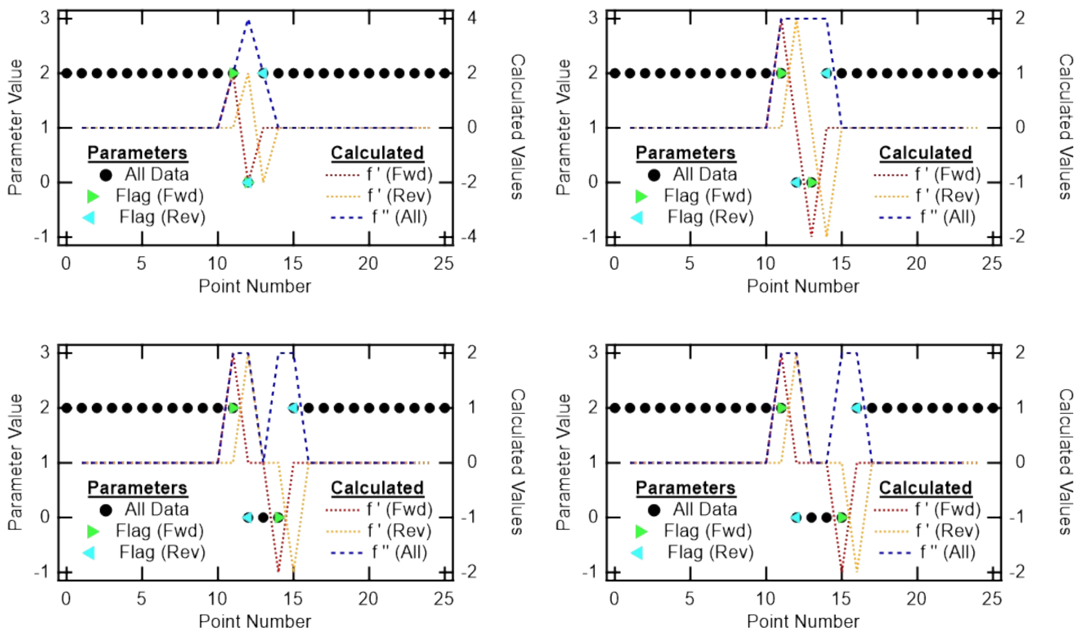


Figure S18. Method for flagging potential bad points/dropouts in data.

Next the algorithm tests each of the bad points. If the following conditions are met, the points are confirmed to be dropouts (note that there are visual representations of 2-5 below):

1. The forward "dropout" (from 1st green arrow to 1st cyan arrow) must come before the reverse "drop in" (from 2nd green arrow to 2nd cyan arrow)
2. The dropout can only span 15% of the data. [$Dx < 0.15*Dx_{tot}$]
3. The change in parameter value from just before the dropouts to just after the dropouts must be smaller than both the drop out and the drop in. [$Db < Da$ & $Db < Dc$]
4. The change in parameter value from just before the dropouts to just after the dropouts must be larger than 3/8 times the sum of the dropout and drop in magnitudes. [$Db < (3/8)*(Da+Dc)$]
5. The change in parameter value at the dropout and drop in points added together must be larger than 1.5 times the change in parameter value at the points just before/after the dropout/drop in points. [$Da + Dc < (3/8)*(Dd+ De)$]
6. The standard deviation of the points just before/after the dropout/drop in must be less than the standard deviation of the same two points and the average of the dropout points.

7. The median value of the dropout points must be closer to the values just before/after the dropout/drop in (1st green arrow, 2nd cyan arrow) than the values just inside (1st cyan arrow, 2nd green arrow).

These are shown below **Figure S19**.

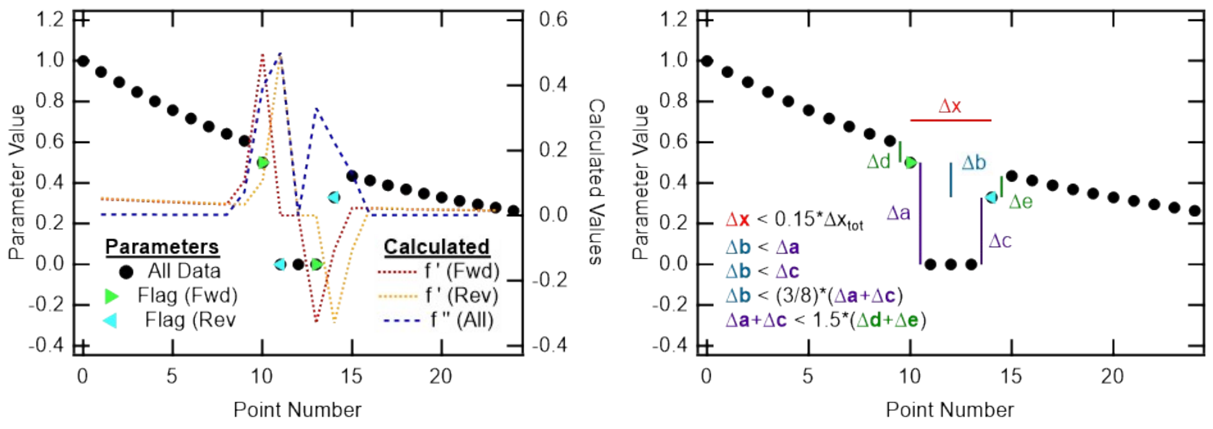


Figure S19. Method to check quality of potential bad points/dropouts to find out which ones to remove.

Next, the dropout points (3 shown above) are set to NaN in "***Param_v_Time_Used***" and removed from the double derivative and directional derivative waves. For the values just outside the dropout points, this is done by replacing the indices that reference them with the values just before/after the dropouts/drop-ins. For example, if points 21, 22, and 23 are considered dropouts, they are replaced with the following:

$$f'(20)_{REV} = f'(20)_{FWD} = f(20) - f(24)$$
$$f''(20) = f(19) - 2 * f(20) + f(24)$$

$$f''(24) = f(20) - 2 * f(24) + f(25)$$
$$f'(24)_{REV} = f'(24)_{FWD} = f(20) - f(24)$$

For the values that are considered dropout points, this is done by setting the double derivative and directional derivative waves to 0, ensuring they will not be flagged again.

Finally, the process is repeated until no more dropouts are identified.

Step 4, fit ***"Param_v_Time_Used"*** from the start point to the end point, ignoring values in the fit which were tagged as potentially bad in step 3. Calculate the difference between the fit wave and ***"Param_v_Time"*** wave to create the residual wave; take its absolute value and find its average and standard deviation. A point is set to NaN in ***"Param_v_Time_Used"*** and added to the list of dropout points generated in step 3 if the following set of conditions are met (note that 4 is optimized based on the quality of the data):

1. $Residual(x) = |f(x)_{EXP} - f(x)_{FIT}| > \mu_{residual} + 4 * \sigma_{residual}$
2. $Residual(x) = |f(x)_{EXP} - f(x)_{FIT}| > 4 * \mu_{residual}$

On the other hand, a point is set to its value in *"**Param_v_Time**"* and removed from the list of dropout points if:

1. $Residual(x) = |f(x)_{EXP} - f(x)_{FIT}| < \mu_{residual}$

Repeat twice.

Step 5, cycle through ***"Param_v_Time_Used"*** to make sure there are not any large gaps in data near the start/end of it without a sufficient amount of good data before/after it. Specifically, the code calculates the locations in ***"Param_v_Time_Used"*** with 5 or more dropout points in a row and ensures that there are at least 3*n good points before/after it (where n is the number of dropout points). If there are not a sufficient amount of good points before/after the bad points the startpoint/endpoint is adjusted to be just after/before the dropouts.

Step 6, Consider a device test dead and set all values in ***"Parm_v_Time_Used"*** to NaN if any of the following conditions are met:

1. There are less than 10 points available (note that 10 points are required for the 10th degree polynomial fitting utilized)
2. Over 75% of the points have been removed between the calculated start and end points
3. Start point is not within first half of the wave (too much bad data to start)
4. There are not 7 points of data in a row between the start and end points

*At the end of this function **"Param_v_Time_Used"** should be identical to **"Param_v_Time"** but have outlier values removed.*

Function (3c) "Normalize Parameters": Calculates the normalized data for the "**Param_v_Time**" matrix. First, it calculates the first and last data points containing real numbers (this is the start point and endpoint from above). Then, it fits the "PCE_<direction>" column in "**Param_v_Time_Used**" from the start point to end point with a 10th degree polynomial and calculates maximum efficiency of the forward and reverse scans from the start point to 24 hours after that point. Next, it fits each of the device metrics with a 10 degree polynomial over the same range and normalizes the data in "**Param_v_Time**" with the value of the parameter at the time of maximum efficiency. Finally, it fills out the "T80" column on the "**Param_v_Time**" wave. Note that skipping the previous step can cause the 10th degree polynomial used for fitting/normalization to be off, leading to erroneous normalization values.

*At the end of this function the normalized values of "**Param_v_Time**" should be filled out and complete, as should the T80 value.*

Function (3d) "Fit Parameters": Calculates fits for standard device metrics including J_{sc}, V_{oc}, FF, and PCE in both forward and reverse scan directions and regular and normalized form. The fits, stored in "**Param_FitCoeff**", include:

Poly 4:
$$J(V) = \sum_{1}^{4} p4\_i * V^{i-1}$$

Poly 10:
$$J(V) = \sum_{1}^{10} p10\_i * V^{i-1}$$

Line $J(V) = Line\_1 + Line\_2 * x$

Exp $J(V) = Exp\_1 + Exp\_2 * e^{-x/Exp\_3}$

Double_Exp $J(V) = D\_Exp\_1 + D\_Exp\_2 * e^{-x/Exp\_3} + D\_Exp\_4 * e^{-x/Exp\_5}$

The Chi2 for each fit is stored under its name; results/coefficients are stored under the variable names used above. To ensure converging and meaningful fits, bounds were put on the constants for the exponential fits as follows:

0	<	Exp_1/D_Exp_1	<	100
0	<	Exp_2/D_Exp_2	<	1000
0	<	Exp_3/D_Exp_3	<	100000
0	<	Exp_4/D_Exp_4	<	1000
0	<	Exp_5/D_Exp_5	<	100000

In addition to creating these rows and filling them with variables, it creates the "**Best_Fit**" row in this matrix which stores a number from 1 to 3 to describe the function that produces the best fit (lowest Chi2), where 1 = linear fit, 2 = exponential fit, and 3 = double exponential fit. If Chi2 values are even, linear and then exponential fits are used to minimize the number of variables. Note that these values can be used to describe the functional decay of various parameters, allowing the user to reduce the number of points necessary to describe the degradation data of one device from (timesteps)*(voltagesteps) to 20 points in the case of the double exponential, 12 in the case of the singular exponential, and 8 in the case of the line. Although not done here, these variables could be input into various machine learning algorithms to extract trends over a large group of devices.

*At the end of this function **"Param_FitCoeff"** should be filled out.*

Function (3e) "Interpolate Parameters (10th Deg Polynomial)": generates waves that are useful for comparing devices. First, it uses the 10th degree polynomial fit coefficients calculated above to create parameter degradation waves with uniform time spacing of one hour, storing the results in "**Param_Fits**". Then it uses "**Param_Fits**" to generate waves which store the degradation of various parameters as a function of PCE (***"Param_Fits_v_PCE")*** or time elapsed (***"Param_Fits_v_Time***") and 1D waves that have the same data but are used by the kit (same name with _1D tacked on). Additionally, it takes the forward point derivative of ***"Param_Fits_v_PCE***" (as a function of PCE) and stores it in "***Param_Fits_v_PCE_D***" and "***Param_Fits_v_PCE_D_1D***".

Then, it creates the wave "**OnePlot**", which attempts to describe the decay of PCE as a summation of losses from FF, J_{sc}, and V_{oc}. To do this, it starts with the non-normalized data in the "**Param_Fits**" wave and normalizes them to the maximum PCE time over the entire duration of testing to ensure that PCE loss is always > 0. Not doing this causes a strange inflection point anywhere PCE loss crosses 0, as the various parameters (which have seen change from the initialization spot) get multiplied by a PCE loss of 0. Then, it calculates the losses as independent variables, as shown below:

$$\%Loss_{J_{sc}} = \frac{Loss_{J_{sc}}^{Norm}}{Loss_{J_{sc}}^{Norm} + Loss_{V_{oc}}^{Norm} + Loss_{FF}^{Norm}} * Loss_{PCE}^{Norm}$$

$$\%Loss_{V_{oc}} = \frac{Loss_{V_{oc}}^{Norm}}{Loss_{J_{sc}}^{Norm} + Loss_{V_{oc}}^{Norm} + Loss_{FF}^{Norm}} * Loss_{PCE}^{Norm}$$

$$\%Loss_{FF} = \frac{Loss_{FF}^{Norm}}{Loss_{J_{sc}}^{Norm} + Loss_{V_{oc}}^{Norm} + Loss_{FF}^{Norm}} * Loss_{PCE}^{Norm}$$

Note that these parameters are not actually independent variables, however the method used provides a facile, quick, and informative way to visualize the data.

*At the end of this function **"Param_Fits"**, **"Param_Fits_v_PCE"**, **"Param_Fits_v_PCE_1D"**, **"Param_Fits_v_Time"**, and **"Param_Fits_v_Time_1D"**, and **"Param_Fits_v_PCE_D"**, **"Param_Fits_v_PCE_D_1D"**, should be filled out and complete in the Device Folder. "**OnePlot**" should also be generated in the Device Folder and filled out.*

Function (3f) "Calculate Type Statistics": Calculates statistics for each of the device type folders. When averaging the devices, there are three options for dealing with devices that have premature catastrophic failure: (1) do nothing, averaging everything including dead devices, (2) include devices in the average until they die, and (3) completely exclude devices which die prematurely. Choice (1) causes a device that prematurely dies to bring down the statistics of the entire device set. Choice (2) causes a device that prematurely dies to bring up the sample set, as it is no longer included in the average. For most cases, choice (3) appears to be the best option, as it eliminates any jagged features, but requires us to choose which waves to average and what testing duration to use; a clear exception to this is exemplified by the 85 C case, for which devices catastrophically fail one at a time.

To do this, the code first calculates the endpoint time for each device test (reminder that this will either be the last time point of testing or where the PCE drops below 1%). Then, for each endpoint, it calculates the number of other devices that have an endpoint time less than 100 hours after it. Finally, it calculates the max value of the latter. If over 50% of the devices have a common end point (within 100 hours), it averages all waves with endpoint values greater than or equal to the chosen endpoint, removing outliers which prematurely fail. If such a grouping cannot be found, it averages all devices, resulting in a decrease in the average and increase in the standard deviation of parameters as they fail. It stores the averaged waves in "**Param_Fits_Avg**" and the standard deviation at each point in "**Param_Fits_StdDev**". Finally, it uses the average wave to generate the "**OnePlot**" matrix, as described above in Function 3e. Note that if multiple endpoints have the same amount of devices with endpoint less than 100 after then, it defaults to using the smaller value of the two so that the maximum number of traces can be included in the average.

*At the end of this function "**Param_Fits_Avg**", "**Param_Fits_StdDev**", and "**OnePlot**" should be created and filled out in Type Folder.*

Function (4) "Apply Filters (UI)": launches a UI that handles applying material, parameter, and/or functional form of degradation filters to an experiment that has already been worked up using the standard analysis functions. The function searches "**Device_Info**", "**Param_FitCoeff**", and "**Param_Fits_v_Time**" to generate the list of a devices that adhere to the material, parameter, and fit filters set in the UI. Then, it generates a new folder at the path specified in the UI that contains several waves, including:

"**DeviceInfo**"	Contains device fabricator, starting test date, device ID, device pixel, and type #
"**Param_Fits_v_Time**"	Contains maximum values for all parameters as well as normalized values at 10 h, 100 h, 250 h, 500 h, 750 h, and 1000 h.
"**Mode_v_Time**"	Contains the dominant mode of degradation over each of the time stamps listed above (defined as the parameter with the largest change from *i* to *i-1* in the normalized wave).
"**Tally_Modes_v_Time**"	Tallies each time J_{sc}, V_{oc}, and FF are the dominant modes of degradation over each of the time stamps above
"**Dominant_Mode_v_Time**"	Determines the dominant mode of degradation over the duration of testing (the parameters with the most ticks above)
"**Param_Fits_v_PCE**"	Contains maximum values for all parameters as well as normalized values at 97.5%, 95.0%, 92.5%, 90.0%, 87.5%, 85.0%, 82.5%, 80%, 77.5%, and 75%.
"**Dominant_Mode_v_PCE**"	Contains the dominant mode of degradation over each of the PCE stamps listed above (defined as the parameter with the largest change from *i* to *i-1* in the normalized wave).
"**Mode_v_PCE**"	Tallies each time J_{sc}, V_{oc}, and FF are the dominant modes of degradation over each of the PCE stamps above.
"**Tally_Modes_v_PCE**"	Determines the dominant mode of degradation from 100% to 75% of initial performance (the parameters with the most ticks above)

“Param_Fits_v_PCE_D”	Contains the derivative of Param_Fits_v_PCE as a function of PCE

Additionally, it creates a folder called “Bins” which can be selected to graph dominant modes over time or PCE. The waves within this folder are simply for graphing purposes, but include:

“Bin_Modes_v_PCE”	Matrix that displays the % of devices where J_{sc}, V_{oc}, and FF dominated device degradation as function of remaining PCE. In this matrix J_{sc}, V_{oc}, and FF are the rows, and “<Direction>_<% PCE as decimal>_Dom” are the column headers
“Bin_Modes_v_PCE_1D”	The same matrix as above but transverse. In this matrix “<Direction>_<% PCE as decimal>_Dom” are the rows an Jsc_Bin, Voc_Bin, and FF_Bin are the columns.
“Param_Fits_v_PCE_Binned”	Matrix that displays the # of times that each of the parameters were the dominant mode of degradation for each PCE step.
“Bin_Modes_v_Time”	Matrix that displays the % of devices where J_{sc}, V_{oc}, and FF dominated device degradation as a function of device aging time. In this matrix J_{sc}, V_{oc}, and FF are the rows, and “<Direction>_<time in hrs>_Dom” are the column headers
“Bin_Modes_v_Time_1D”	The same matrix as above but transverse. In this matrix “<Direction>_<time in hrs>_Dom” are the rows an Jsc_Bin, Voc_Bin, and FF_Bin are the columns.
“Param_Fits_v_Time_Binned”	Matrix that displays the # of times that each of the parameters were the dominant mode of degradation over the last timestep.

*At the end of this function you should have a Folder in root:Filters:<folder name> with the specified name. In it should contain “**DeviceInfo**”, “**Param_Fits_v_Time**”, “**Mode_v_Time**”, “**Tally_Modes_v_Time**”, “**Param_Fits_v_PCE**”, “**Mode_v_PCE**”, “**Tally_Modes_v_PCE**”, and the “**Bins**” folder. The “**Bins**” folder should contain “**Bin_Modes_v_PCE**”, “**Bin_Modes_v_PCE_1D**”, “**Param_Fits_v_PCE_Binned**”, “**Bin_Modes_v_Time**”, “**Bin_Modes_v_Time_1D**”, “**Param_Fits_v_Time_Binned**”.*

Function (5) “Data Visualization (UI)”: Launches a UI that handles all data visualization options included in the package. The panel consists of explanations of what each of the buttons do, options for how/where to use them, and buttons to make the plots.

This function does not generate new waves, folders, or data.

Data Formatting User Interface: Clicking the "Create JV Matrix (UI)" Button on the Main UI generates the following UI (**Figure S20**):

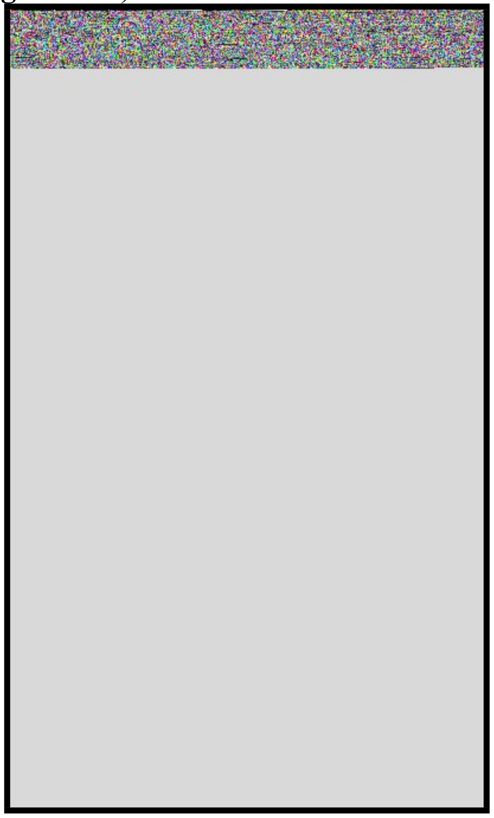


Figure S20. UI for converting x,y data into the format used by the code.

This UI handles formatting data from a number of plausible inputs into one useable by the kit. To do so it makes several assumptions 1) the data starts in a folder with the name of the device, 2) the data is I(V) or J(V) data and the waves are listed in chronological order, and 3) the time spacing between the scans can be approximated as equal. The first panel allows the user to input the start date and time for the test. The second panel allows the user to enter the duration of the test (either the duration of one scan time step or the duration of all the scans can be used, the only thing that matters is the duration per scan). The third panel allows the user to input the information necessary to describe the device, including architecture, device/test type, and fabricator. The fourth panel allows the user to enter the information necessary to describe the test, including temperature and atmospheric conditions. The fifth panel allows the user to describe the format of the incoming data. I(V) implies that the raw data has not been scaled by area; J(V) implies that it has. In addition to these toggle boxes, the user should select the type of data format. Possible options are 1 direction, 2 directions (fwd then rev), or 2 directions (rev then fwd).

Once the top half of the UI is filled out, the user should select the folder with the raw data in it and click the "**Generate Paths & Variables**" button to populate the information in the fourth panel. If the information there looks good (free from typos), hit "**Convert Files**". This will create the "**JV_Matrix**" wave at the output path location using files from the input path with the given starting time stamp and time per scan and the "**Type_Info**" wave one folder closer to root (in the Type Folder). The rest of the kit relies on the folder hierarchy and these two waves. Note that for this UI, the bulk of the information about the test is inputted manually, but the final data name and where the data is copied from is determined by what is selected in the data browser. This should facilitate quicker loading.

Data Filtering User Interface: Clicking the "**Apply Filters**" Button on the Main UI generates the following UI (**Figure S21**):

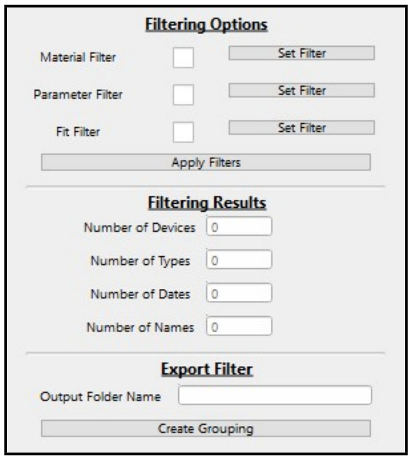
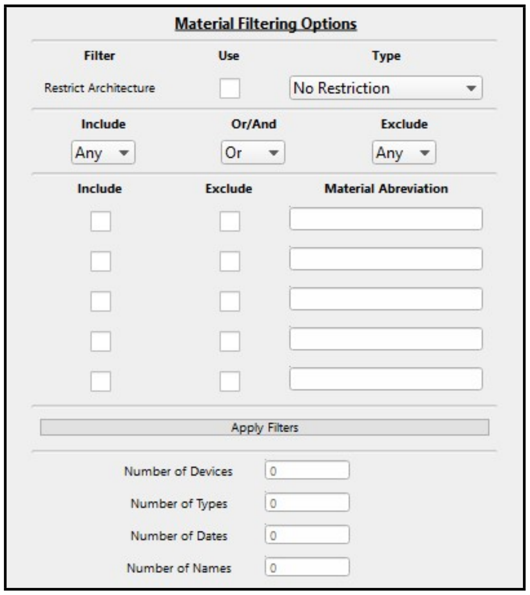


Figure S21. Main Filtering UI

This UI manages the overall filtering functions. The first panel controls what filters are applied. Each type of filter (material, parameter, fit) has a checkbox to toggle on/off the filter and a "**Set Filter**" button to adjust the parameters set by each individual filter. At the bottom of the section is an "**Apply Filter**" button that updates the filtering results. The second panel then displays various important metrics from the filter, including the number of devices, device types, dates, and fabricators included in the given filter. Finally, the bottom panel has a blank text box for the user to input a folder name to save the results to and a "**Create Grouping**" button to apply the filters / create the resulting files in the specified folder (root:Filters:<Output Folder Name>).

(a) Clicking on the "**Set Filter**" button for "**Material Filter**" in the first panel generates the following UI (**Figure S22**):

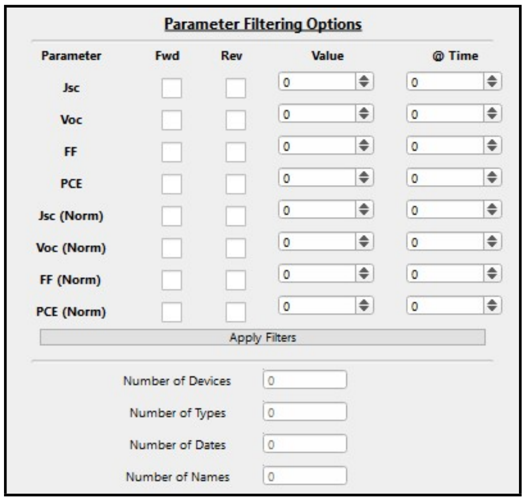


Number of Devices: 0
Number of Types: 0
Number of Dates: 0
Number of Names: 0

Figure S22. Material Filtering UI.

This UI controls filtering by material/architecture. The first panel allows the user to restrict architectures to n-i-p or p-i-n. The second panel sets up the logic statements required for the include and exclude statements in the third panel. The third panel controls what material is being searched for and whether devices should include or exclude the specified material. The second to last panel has an "**Apply Filters**" button to update the filters. The last panel has the same device metrics displayed as the main UI (# of devices, types, dates, and fabricators).

(b) Clicking on the "**Set Filter**" button for "**Parameter Filter**" in the first panel generates the following UI (**Figure S23**):



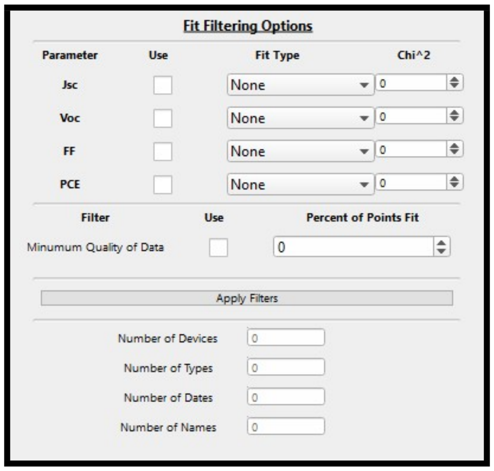
Parameter	Fwd	Rev	Value	@ Time
Jsc			0	0
Voc			0	0
FF			0	0
PCE			0	0
Jsc (Norm)			0	0
Voc (Norm)			0	0
FF (Norm)			0	0
PCE (Norm)			0	0

Number of Devices: 0
Number of Types: 0
Number of Dates: 0
Number of Names: 0

Figure S23. Parameter Filtering UI.

This UI controls filtering by various parameters. The first panel contains a list of parameters on the left, two checkboxes to reference the forward or reverse waves in the middle, and two open fields to input a time or value on the right. At the bottom of the panel, there is an "**Apply Filters**" button that will update the set of filters and display the same set of device metrics displayed on the main UI (# of devices, types, dates, and fabricators).

(c) Clicking on the "**Set Filter**" button for "**Fit Filter**" in the first panel generates the following UI (**Figure S24**):



Fit Filtering Options

Parameter	Use	Fit Type	Chi^2
Jsc	☐	None	0
Voc	☐	None	0
FF	☐	None	0
PCE	☐	None	0

Filter	Use	Percent of Points Fit
Minumum Quality of Data	☐	0

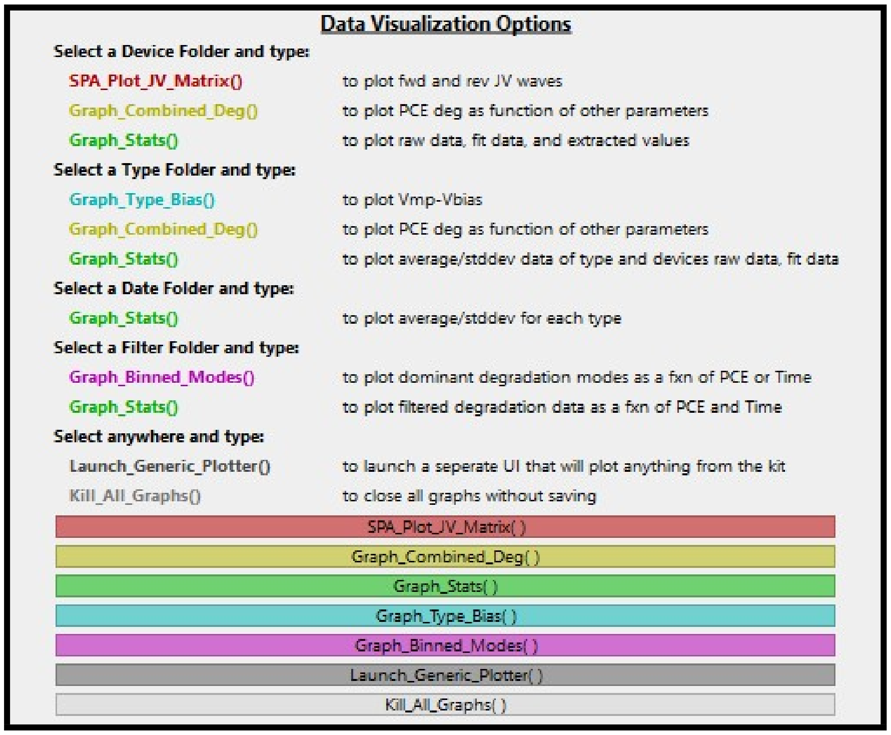
Apply Filters

Number of Devices	0
Number of Types	0
Number of Dates	0
Number of Names	0

Figure S24. Fit filtering UI.

This UI controls filtering by various functional forms of degradation. The first panel has a list of parameters on the left-hand-side, a toggle box to activate each of the filters in the middle, and inputs on the right to select the desired fit type and maximum fit quality (Chi^2). The second panel has options to

<u>Data Visualization User Interface:</u> Clicking the ***"Launch Graphing Panel"*** on the Main UI generates the following UI (**Figure S25**):



Data Visualization Options

Select a Device Folder and type:

SPA_Plot_JV_Matrix()	to plot fwd and rev JV waves
Graph_Combined_Deg()	to plot PCE deg as function of other parameters
Graph_Stats()	to plot raw data, fit data, and extracted values

Select a Type Folder and type:

Graph_Type_Bias()	to plot Vmp-Vbias
Graph_Combined_Deg()	to plot PCE deg as function of other parameters
Graph_Stats()	to plot average/stddev data of type and devices raw data, fit data

Select a Date Folder and type:

Graph_Stats()	to plot average/stddev for each type

Select a Filter Folder and type:

Graph_Binned_Modes()	to plot dominant degradation modes as a fxn of PCE or Time
Graph_Stats()	to plot filtered degradation data as a fxn of PCE and Time

Select anywhere and type:

Launch_Generic_Plotter()	to launch a seperate UI that will plot anything from the kit
Kill_All_Graphs()	to close all graphs without saving

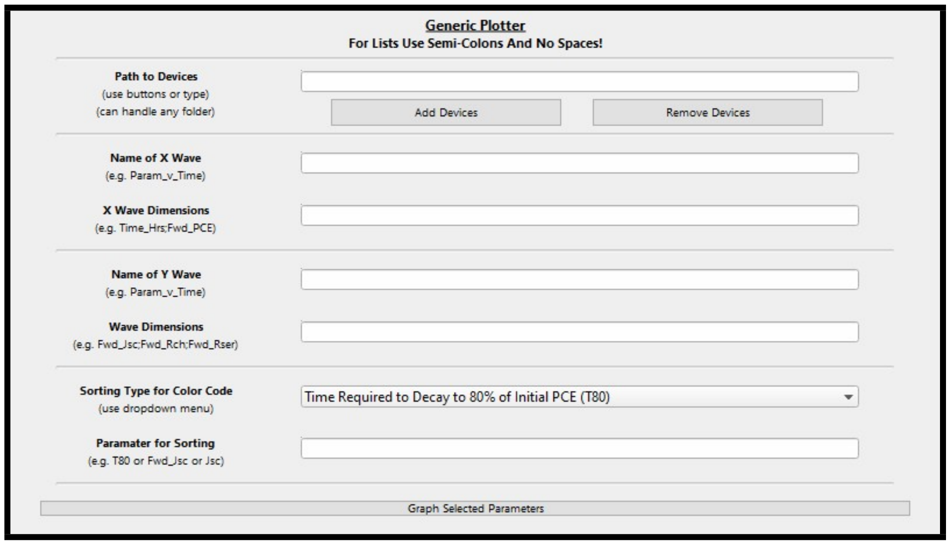
Figure S25. Developed visualization methods.

This UI handles all premade data visualization options. The top panel has a list of instructions. The bottom panel has a number of buttons to generate various plots. These plots include:

SPA_Plot_JV_Matrix()	Plots all JV waves for a particular device. Solid traces represent forward scans. Dashed traces represent reverse scans. Individual traces are color coded from black (first scan) to red (most recent scan)
Graph_Combined_Deg()	Graphs the degradation of PCE as a function of V_{oc}, J_{sc}, and FF for a given device or device type by representing them as independent variables (e.g. PCE_Loss = FF_loss + Jsc_Loss + Voc_Loss)
Graph_Stats()	Graphs statistics depending on the folder that has been selected.
	For device: plots raw data, best fit line, and used data points
	For type: plots raw data, best fit line, average, and standard deviation
	For types: plots average and standard deviation
	For filter: plots degradation loss as function of time and PCE
Graph_Type_Bias()	Graphs the difference between the bias each pixel saw over testing and its maximum power point voltage. Displays all pixels on one plot with reverse pixels blue and forward pixels red. Assumes 510 Ohm load, which can be changed in the code.

Graph_Binned_Modes()	Graphs a tally of the dominant modes v PCE or time for a given set of filtered data.
Launch_Generic_Plotter()	Launches a separate UI that allows plotting of a set of variables versus another set of variables with various options for color coding.
Kill_All_Graphs()	Closes all graphs without saving them

As noted above, clicking the **"*Launch_Generic_Plotter()*"** button on the Data Visualization UI launches the following UI (**Figure S26**):


Generic Plotter
For Lists Use Semi-Colons And No Spaces!

Path to Devices
(use buttons or type)
(can handle any folder)

Add Devices
Remove Devices

Name of X Wave
(e.g. Param_v_Time)

X Wave Dimensions
(e.g. Time_Hrs;Fwd_PCE)

Name of Y Wave
(e.g. Param_v_Time)

Wave Dimensions
(e.g. Fwd_Jsc;Fwd_Rch;Fwd_Rser)

Sorting Type for Color Code
(use dropdown menu)
Time Required to Decay to 80% of Initial PCE (T80)

Paramater for Sorting
(e.g. T80 or Fwd_Jsc or Jsc)

Graph Selected Parameters

Figure S26. Generic Plotting UI.

This UI handles graphing various outputs of the coding package. The top panel allows the user to select the list of devices included in the plot. To add/remove a device or set of devices, select a folder in the data browser (can be a filter, fabricator, date, type, device folder, etc) and click "***Add Devices***"/ "***Remove Devices***" on the UI. Note that the path to devices will update to reflect the folders currently included in the set, but that removing a subset of devices from a larger set of devices which has already been added (i.e. removing a type of device from a test date) works, but will not be reflected in the list. The second and third panels then allow the user to input the name and dimensions of the x and y waves that will be plot. Finally, the last panel allows the user to select a number of schemes for coloring the traces added to graph. Currently, the kit has options for T80, value at starting time using the average of the forward & reverse parameters, and value at starting time using the exact parameter name entered. For these filters the maximum and minimum parameter values in the filter set are calculated and the region from maximum to minimum value is divided into 5 ranges. Devices with parameter values in top 80-100% of values, i.e. devices where:

$$Param > Param_{Max} - \frac{Param_{Max} - Param_{Min}}{5} * iteration$$

with iteration=1 are color-coded red while devices with 80-60% of the max value (iteration =2) are orange, devices with 60-40% are green, devices with 20-40% are blue, and devices with 0-20% are purple.