SAND2017-0721PE

PV Performance Modeling Collaborative

PV System Performance Modeling

DOE MOSAIC Annual Review
San Francisco, CA
Jan 17-18 2017

Joshua S. Stein PhD.
Sandia National Laboratories,
Albuquerque, NM

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Outline

- Why is modeling PV performance difficult?
- 10 Standard PV performance modeling steps
- PVLIB open source functions available for setting up your own models
- Questions and discussion

Why Model PV Performance and What's so difficult about it?

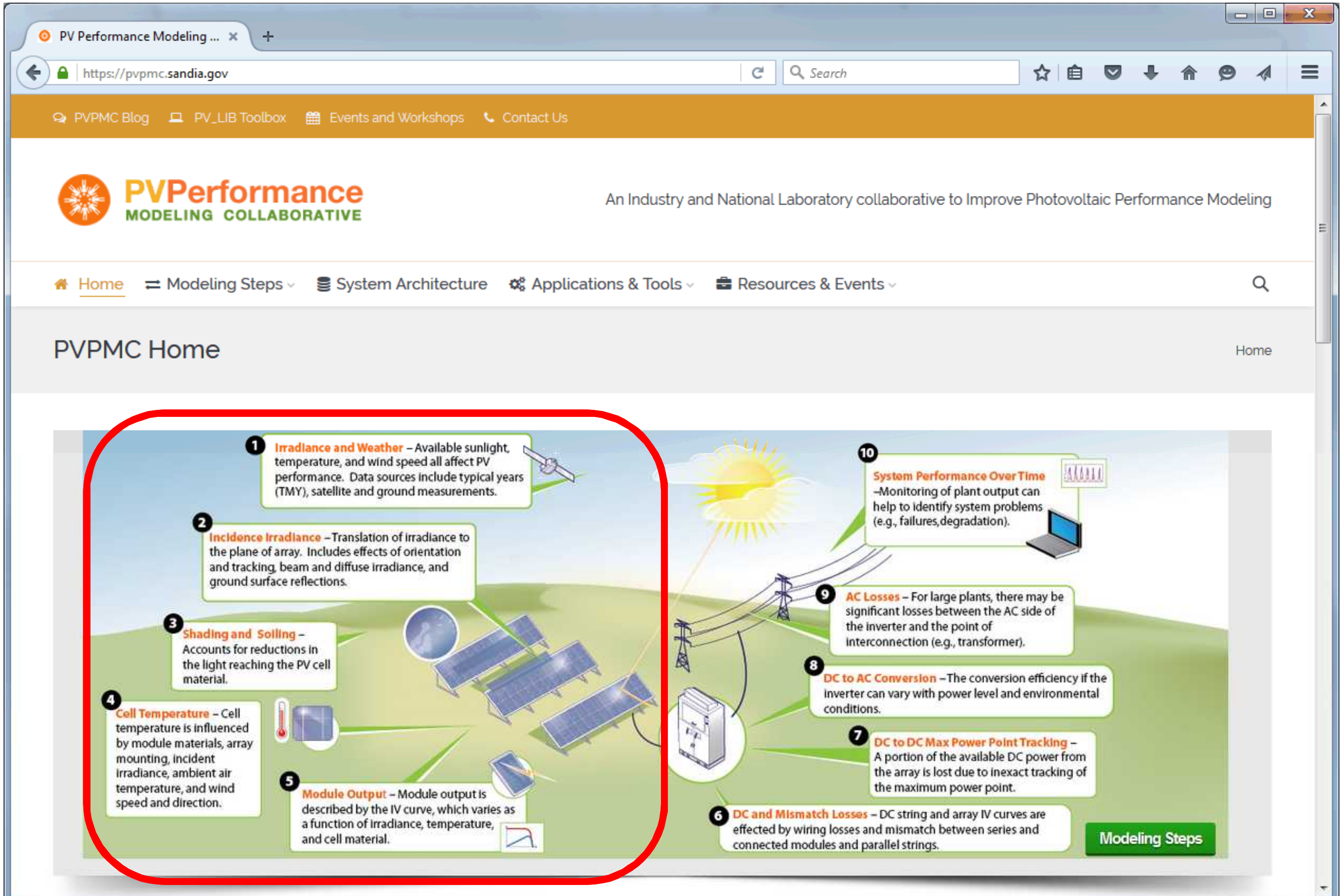
Performance modeling aims to answer these questions:

1. How much energy will a PV system produce over its lifetime?
2. What PV technologies and system designs will work best at my site?
3. Is my PV system operating as expected?

It is difficult to do accurately because:

- While PV responds quite predictably to uniform light (e.g., 1 sun, normal incidence, AM 1.5) and uniform temperature (25˚C), these conditions *NEVER* occur in the field.
  - Atmospheric variability, non-normal incidence, soiling, shading, wind effects, module mounting effects, wiring losses and mismatch, inverter efficiency, etc.

Standardized PV Performance Modeling Steps



PVLIB Toolbox offers a set of ~60 functions in Matlab and Python to evaluate each modeling step

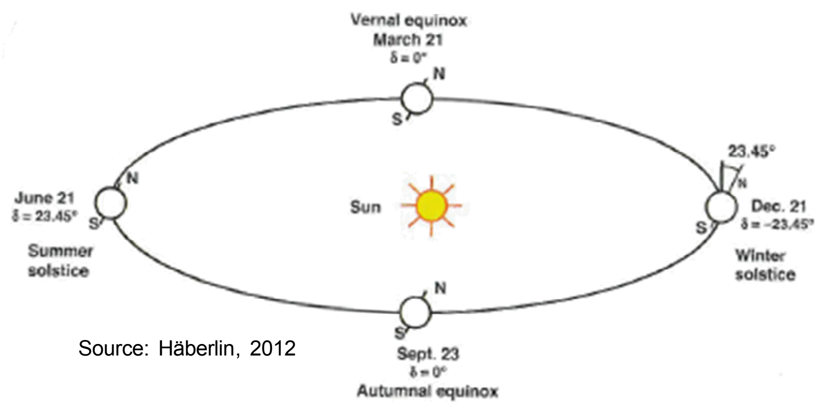
1. Irradiance and Weather

- **Irradiance** is a primary input for a PV performance model
  - Three options to choose from:
    - **Typical years** (TMY) (Irradiance is mostly modeled from other measurements)
    - **Satellite modeled data** (available everywhere, indirect measurement)
    - **Ground measurements** from site (short time period, accurate if maintained)
  - Varies with time and location (and instruments used for measurements). Data quality is important
  - Largest source of uncertainty in PV performance modeling

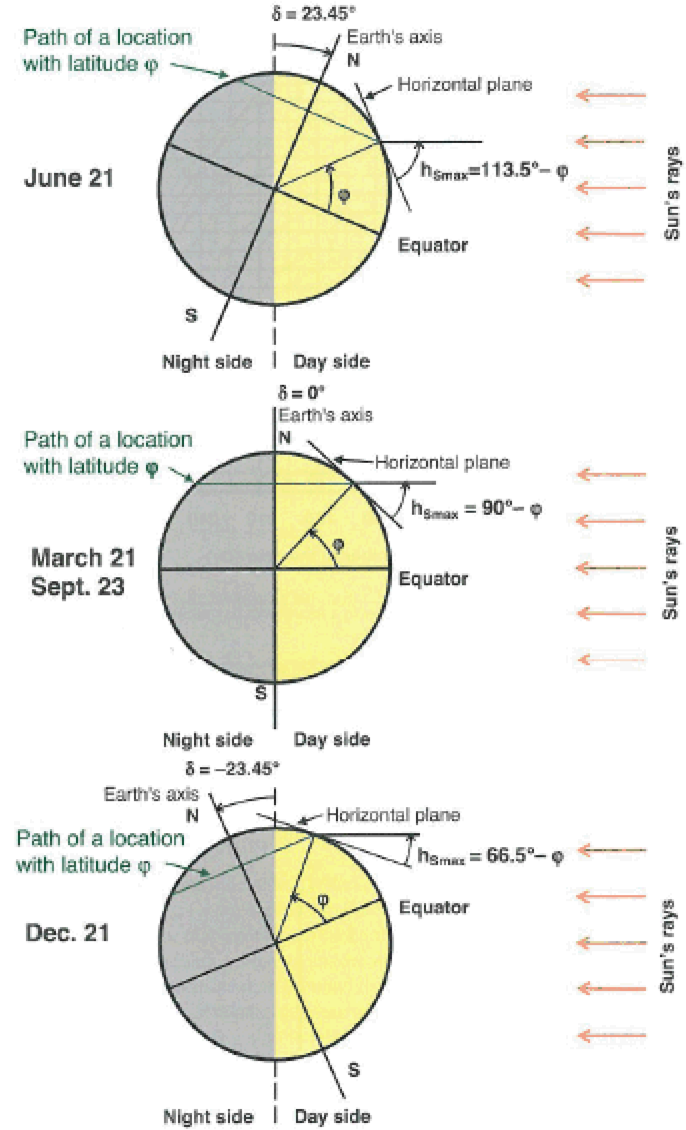
Earth-Sun System

Two Important Facts

- Earth's axis is tilted (23.45 deg) relative to Earth's orbital plane
  - Seasons!
- Orbit is elliptical (closest on Jan 4)
  - Affects extraterrestrial irradiance (6-7%) (1,415-1,322 W/m^2)



Source: Häberlin, 2012

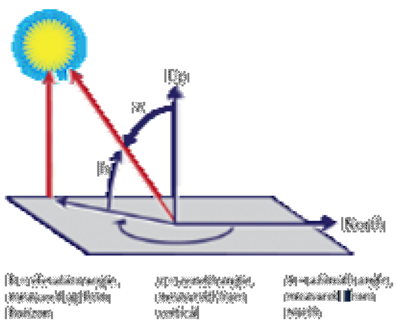


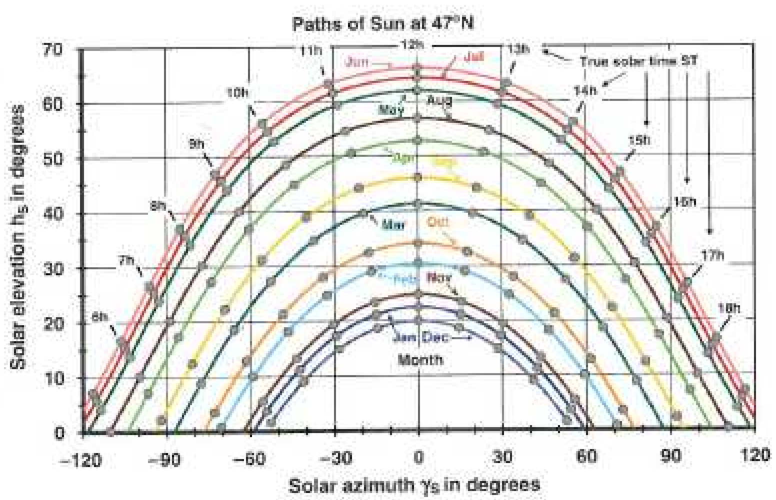
Source: Häberlin, 2012

PV_LIB functions
- pvl_extraradiation

Solar Position

- Solar position relative to an observer on Earth is a critical input to PV performance models.
- Imbedded in PV simulation programs
- Described with:
  - Sun Elevation Angle
    - Zenith = 90-elevation angle
  - Sun Azimuth Angle





Source: Häberlin, 2012

Example Algorithms

- Various online calculators
  - http://www.esrl.noaa.gov/gmd/grad/solcalc/azel.html
  - http://solardat.uoregon.edu/SolarPositionCalculator.html
  - http://www.pveducation.org/pvcdrom/properties-of-sunlight/sun-position-calculator
  - http://www.suncalc.net
  - http://www.sunearthtools.com/dp/tools/pos_sun.php

NREL Solar Position Algorithm (SPA) (also available in PV_LIB)

- "Gold Standard" (Most Accurate, but slow)

Sandia "ephemeris" algorithm (available in PV_LIB) (fast)

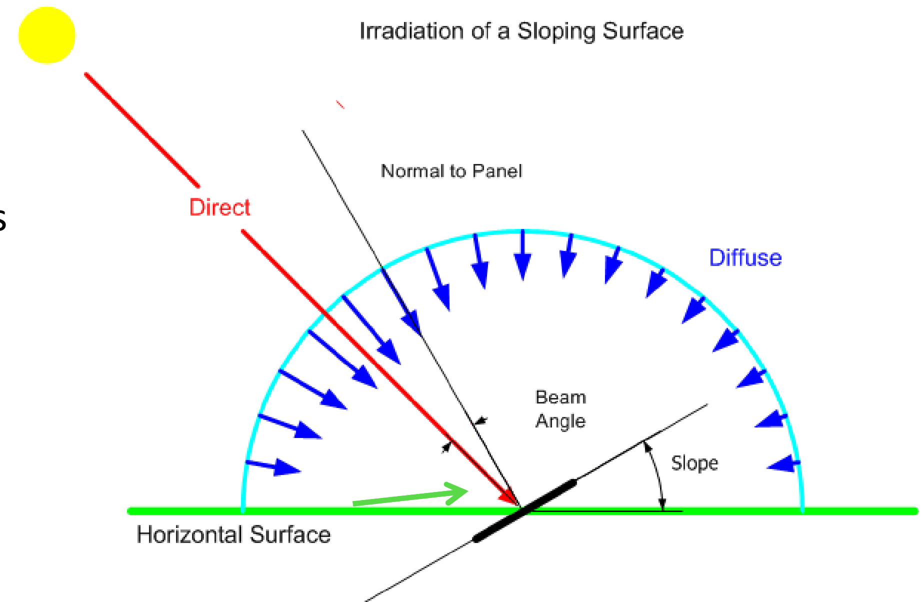
- Simple methods based on declination and hour angle can be off by as much as 5% in elevation angle – time and location dependent.

PV_LIB functions
- pvl_spa
- pvl_ephemeris

Irradiance Components

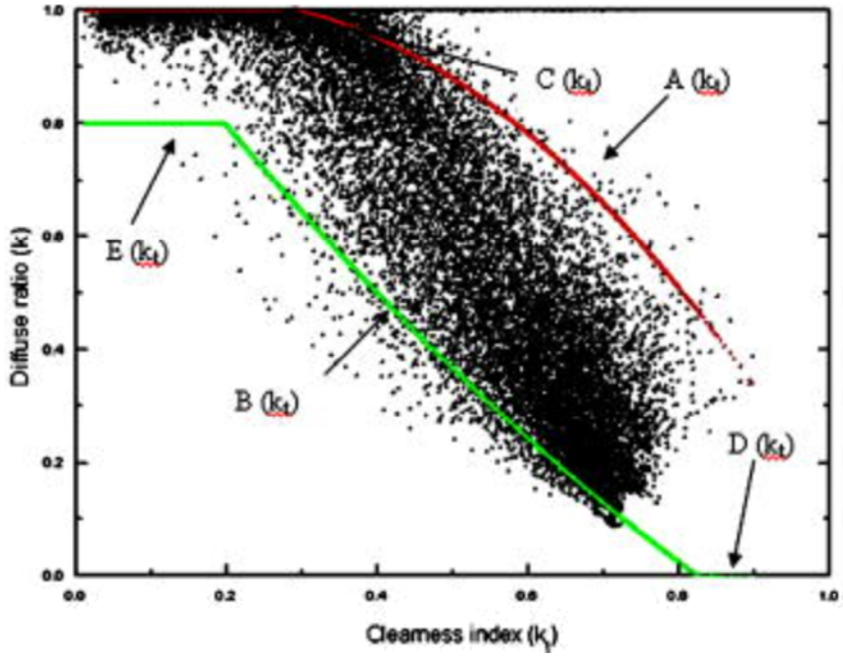
Reference Quantities

- **Direct Normal Irradiance (DNI)**
  - Light hitting a plane normal to the sun's rays that comes directly from the sun.
- **Global Horizontal Irradiance (GHI)**
  - All light hitting a horizontal plane
  - Horizontal beam irradiance = DNI*cos(Zenith Angle)
- **Diffuse Horizontal Irradiance (DHI)**
  - Light hitting a horizontal plane that does NOT come directly from the sun
- **Albedo**
  - Relative reflectivity of the ground surface (usually ~0.2, which means about 20% of the light hitting the ground is reflected)

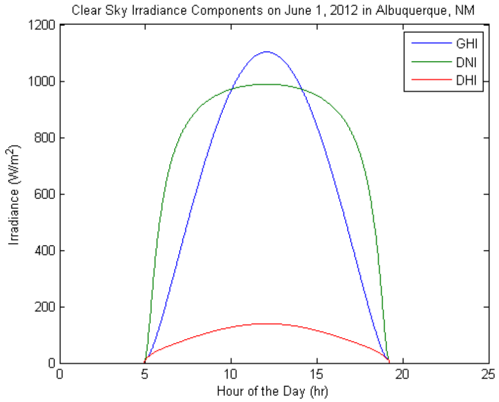


More Irradiance Concepts

- **Decomposition models**
  - Timescale matters (1-min, 1-hr, 1-day)
  - Estimate DNI from GHI
    - Erbs model
    - DISC model
    - DIRINT model
    - Many more…
- **Extraterrestrial radiation**
  - Irradiance outside atmosphere
- **Clear Sky Irradiance Models**
  - Assumes atmosphere but no clouds



Source: J. Sol. Energy Eng.. 2005;128(1):104-117. doi:10.1115/1.2148972

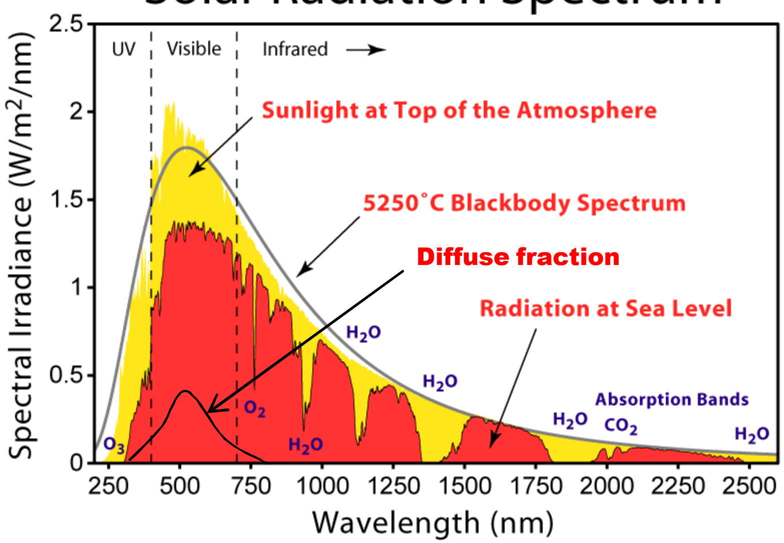


PV_LIB functions
- pvl_disc, pvl_dirint
- pvl_erbs, pvl_orgill_hollands, pvl_reindl_1, pvl_reindl_2
- pvl_clearsky_haurwitz, pvl_ineichen

Solar Spectrum

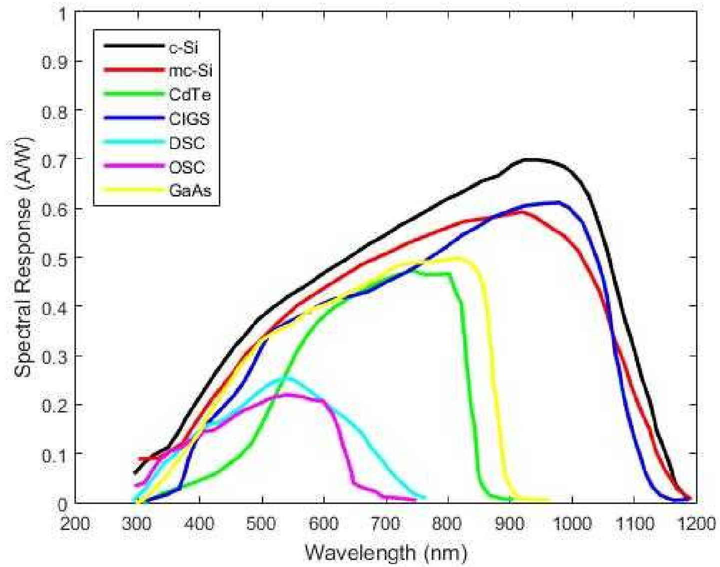
- Solar spectrum is influenced by the sun and thickness and composition of the atmosphere.
- PV cell technologies respond to the spectral range differently
- PV is rated at a standard spectrum (AM1.5 – computer model)

Solar Radiation Spectrum



- Airmass, precipitable water, and clearness are used as a proxy for changes in solar spectrum

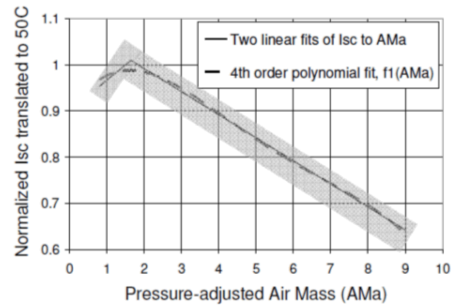
Spectral Response

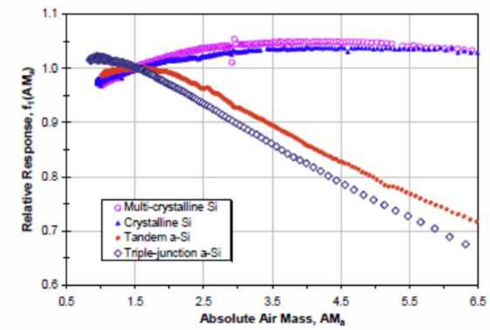


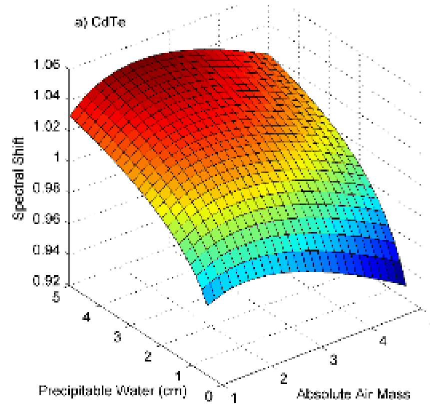
PV_LIB functions
- pvl_pres2alt
- pvl_alt2pres
- pvl_relativeairmass
- pvl_absoluteairmass
- pvl_calcPwat

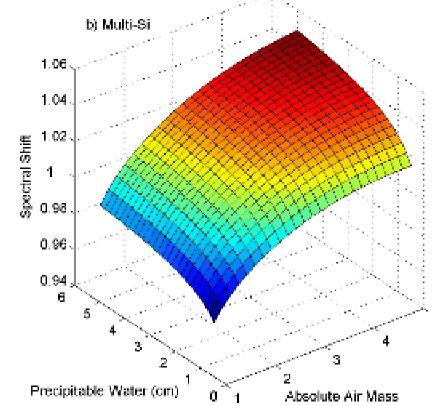
Spectral Correction Models

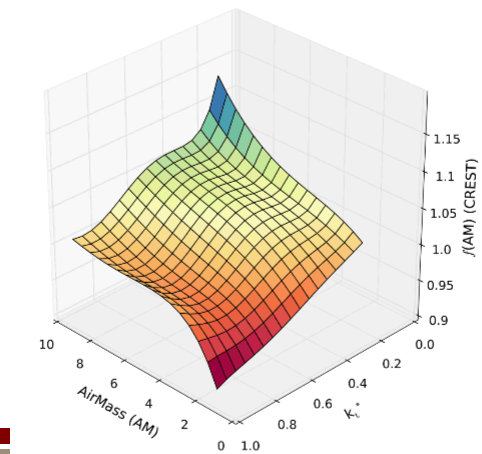
- Sandia (King et al., 2004) suggested using an empirical 4th order polynomial function of AM.
- First Solar uses as 2D response surface function of AM and Precipitable water
- CSIRO has proposed a 2D response surface function of AM and Clearness index (GHI/clear-sky irradiance).
- All these models require outdoor characterization of modules, which is not yet standardized.











PV_LIB functions
- pvl_relativeairmass
- pvl_absoluteairmass
- pvl_FSspeccorr

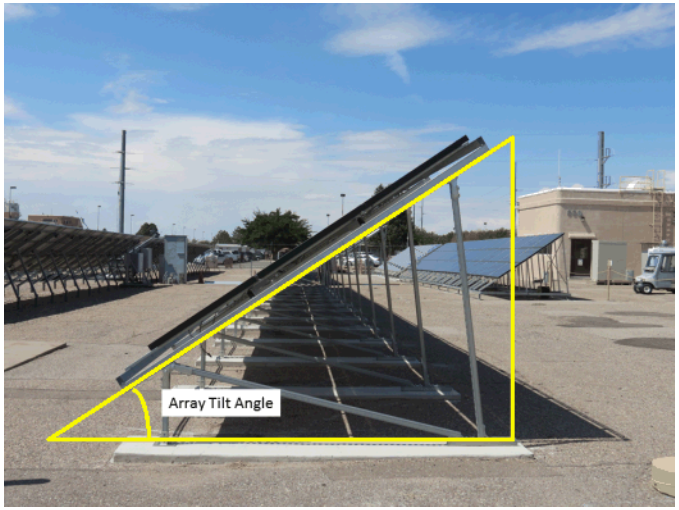
Weather Data

- Irradiance, temperature, wind speed, RH, precipitation (rain and snow) are all required to model PV system performance.
- Annual datasets are needed
- Multiple years of data are nice
  - Interannual variability
- "State of the Art" is to use "typical" meteorological year" (TMY)
  - NSRDB-TMY2 (237 locations) 1961-1990
  - NSRDB-TMY3 (1,454 locations) 1991-2005
  - NSRDB 1991-2010 (1,454 locations)
  - NSRDB 1998-2014 (30 min time steps from satellite data)
- TMY month selection weights based on 25% GHI, 25% DNI, and 50% on non irradiance factors)
  - PV would be better served by 100% on GHI (TGY: Typical GHI Year???)
- Satellite data is becoming more accepted.
  - Measurements available almost everywhere
  - Validation studies are leading to higher accuracies (e.g., snow effects)

PV_LIB functions
- pvl_readtmy2, pvl_readtmy3
- pvl_getISDdata, pvl_readISH
- pvl_maketimestruct
- pvl_makelocationstruct

2. Incident Irradiance

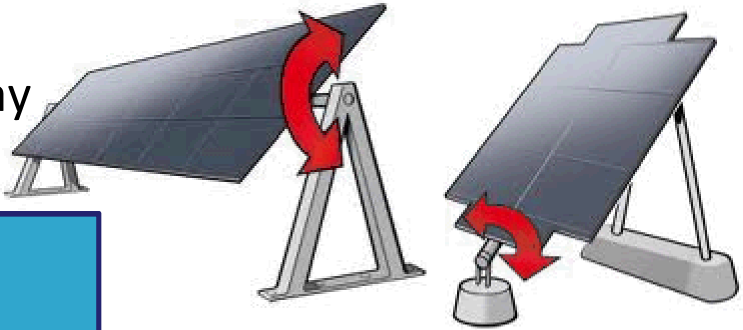
Array Tilt Angle



Array Azimuth Angle

$$AOI = \cos^{-1}\left[\cos(\theta_Z)\cos(\theta_T) + \sin(\theta_Z)\sin(\theta_T)\cos(\theta_A - \theta_{A,array})\right]$$

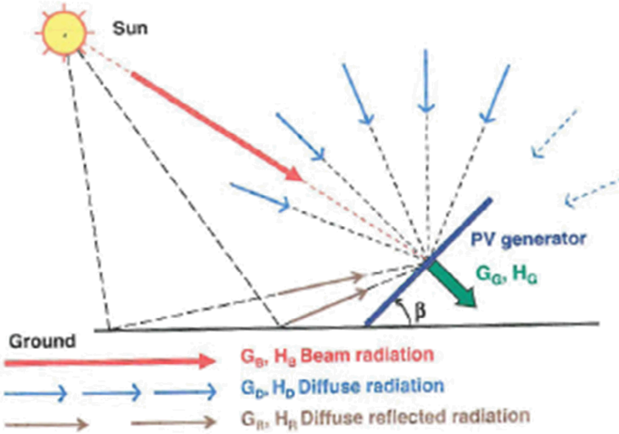
- Single axis tracking moves array to partially follow sun
- 2-axis tracking can keep array normal to the sun (AOI=0)



PV_LIB functions
- pvl_getaoi
- pvl_singleaxis

2. Incident Irradiance (POA)

- Incident irradiance is the light that hits the plane of the array (a.k.a **Plane-of-array irradiance**).
  - Beam irradiance = DNI * cos(AOI)
  - Sky diffuse irradiance
  - Ground reflected irradiance
    - Array tilt angle
    - Ground surface albedo
      - Grass = 0.15-0.25
      - Snow = 0.5 -0.82



Source: Häberlin, 2012

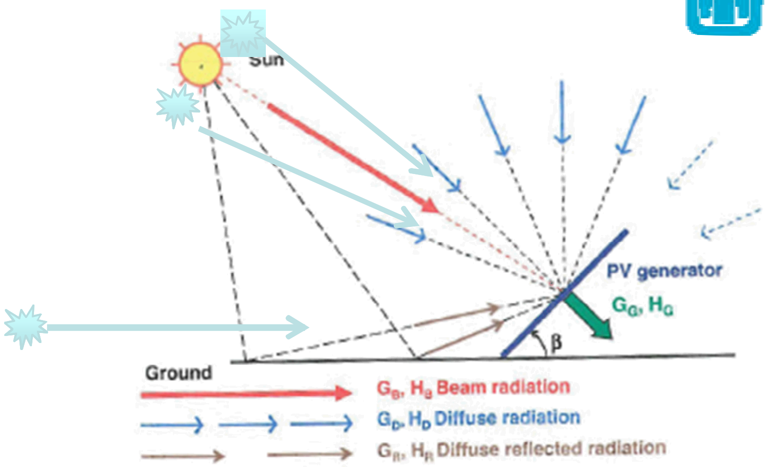
PV_LIB functions

- pvl_grounddiffuse
- pvl_isotropicsky
- pvl_haydavies1980
- pvl_klucher1979
- pvl_perez
- pvl_kingdiffuse
- pvl_reindl1990

2. Incident Irradiance

Sky Diffuse Models

- Isotropic model
  - Assumes uniform diffuse light and is geometric
- Hay and Davies model
  - Isotropic model + circumsolar enhancement (PVsyst)
- Reindl model
  - Isotropic + circumsolar + horizon brightening
- Perez model (1991)
  - Isotropic + circumsolar + horizon brightening, where circumsolar and horizon components are empirical functions based on clearness. Site calibration may be needed. (PVsyst)



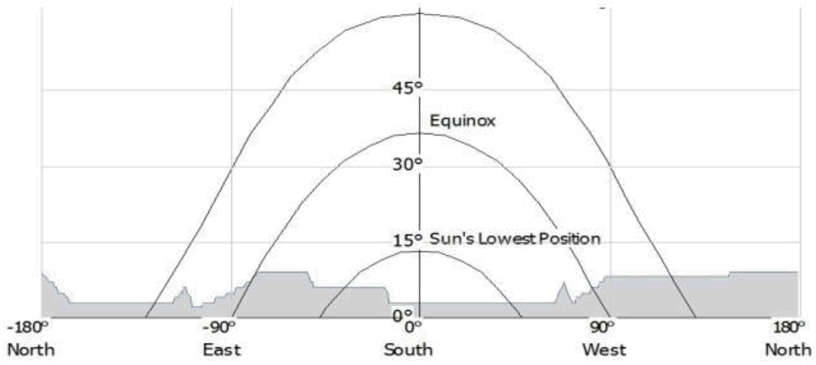
Source: Häberlin, 2012

- King diffuse model
  - Isotropic + lumped empirical term based on fitting data collected at Sandia. At Sandia, this model has proved to be most accurate. Could easily be fit to other data
- Many more….

3. Shading, Soiling, and Reflections

Not all POA irradiance is available!

- Soiling
  - Dirt and grime prevent light from getting to the cells (soiling)
  - Snow can cover all or part of array (soiling)
- Shading
  - Near shade has sharp edges
    - Buildings, trees, appliances, wires, chimney, etc.
    - Row-row shading within array
  - Far shade is more diffuse
    - Mountains, horizon

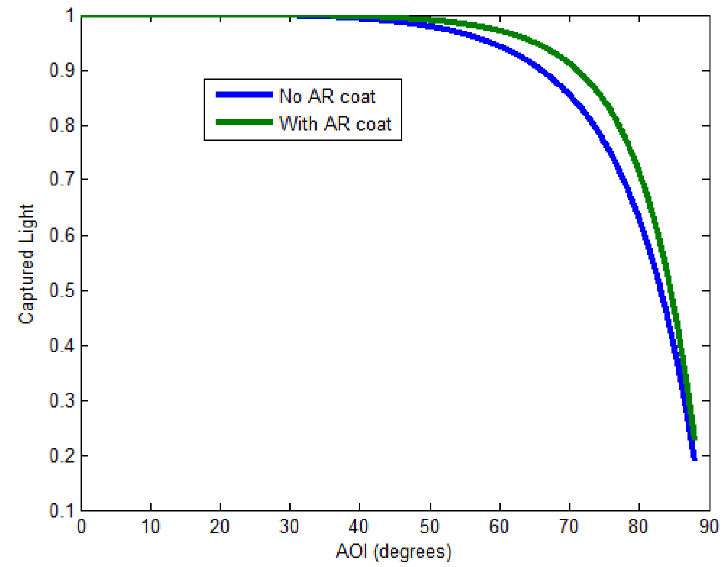


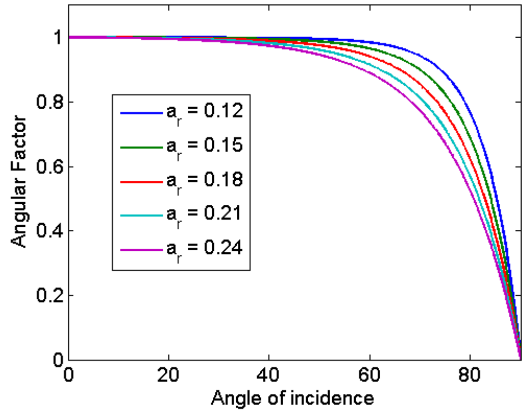
A 2D sunpath diagram in PV*SOL, with horizon data imported.

- Shading effects are assigned to certain time periods and irradiance is reduced.
- Partial shading is much more complex. Modeling approaches are still evolving.
  - Chris Deline (NREL) is developing models to account for partial shading.

3. Shading, Soiling, and Reflections

- Reflections are influenced by:
  - Angle of incidence
  - Coatings (e.g., anti-reflective)
  - Soiling
- Reflections models include:
  - Physical (optical) model based on Snell's and Bougher's laws
  - Empirical models
    - Sandia has used a 5th order polynomial form (F2)
    - ASHRAE
    - Martin and Ruiz (includes effects of soiling)

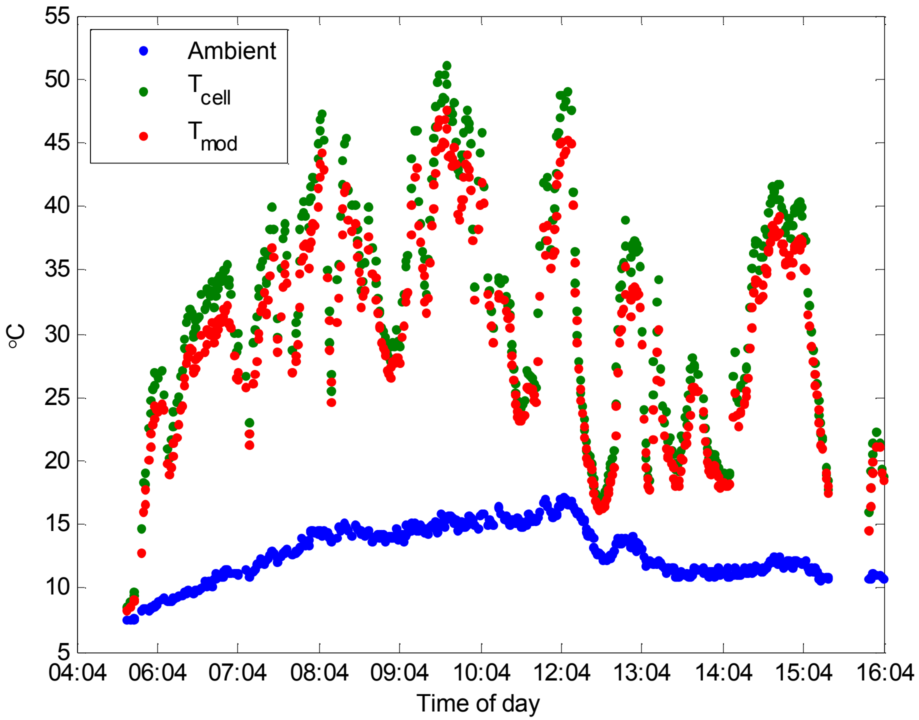




PV_LIB functions (reflection losses)
- pvl_physicaliam
- pvl_ashraeiam
- pvl_martinruiziam

4. Cell Temperature

- Power decreases with increasing cell temperature
  - –0.3 to -0.5 %/C (depends on cell tech., module materials)
- Cell temperature difficult to measure directly in situ
  - Can be inferred from Voc and Isc (e.g., IEC 60904-5)
  - Thermocouples attached to module backsheet + delta
- Cell temperature > ambient (~30C difference)
- Cell temperature > back-of-module temperature (delta = 2° – 4° C)



- Models predict cell temperature from POA irradiance, ambient temperature and wind conditions

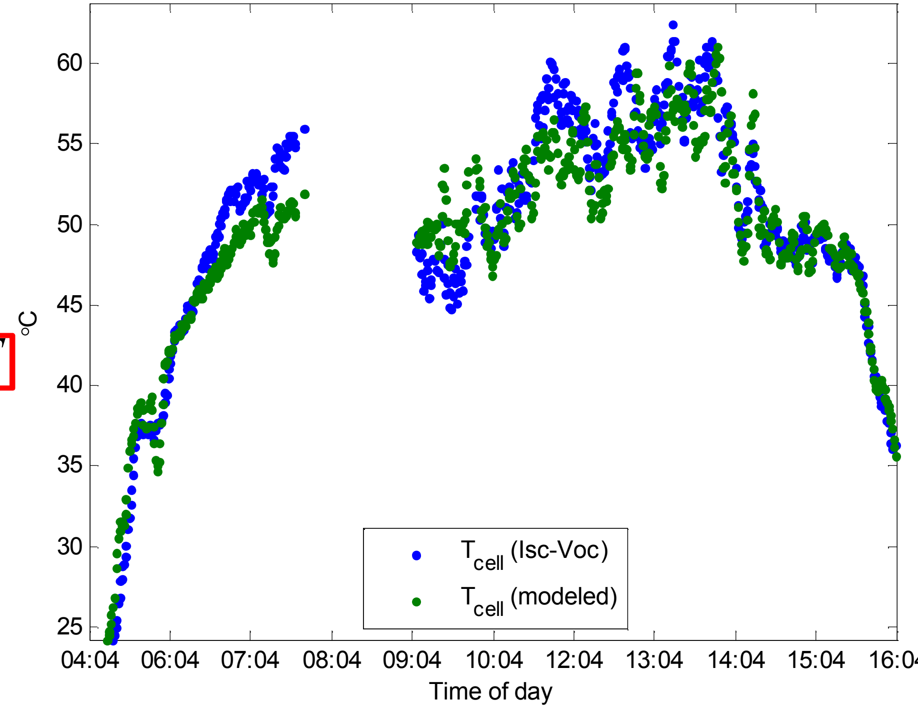
4. Cell Temperature Models

- Most are steady-state

$$T_C = T_M + \frac{E}{E_0}\Delta T$$

$$= T_{amb} + \frac{E}{E_0}\exp\left(a + bWS\right) + \frac{E}{E_0}\Delta T$$

- Typical assumptions:
  - Represents <u>average</u> cell temperature across module
  - Represents <u>average</u> across an array

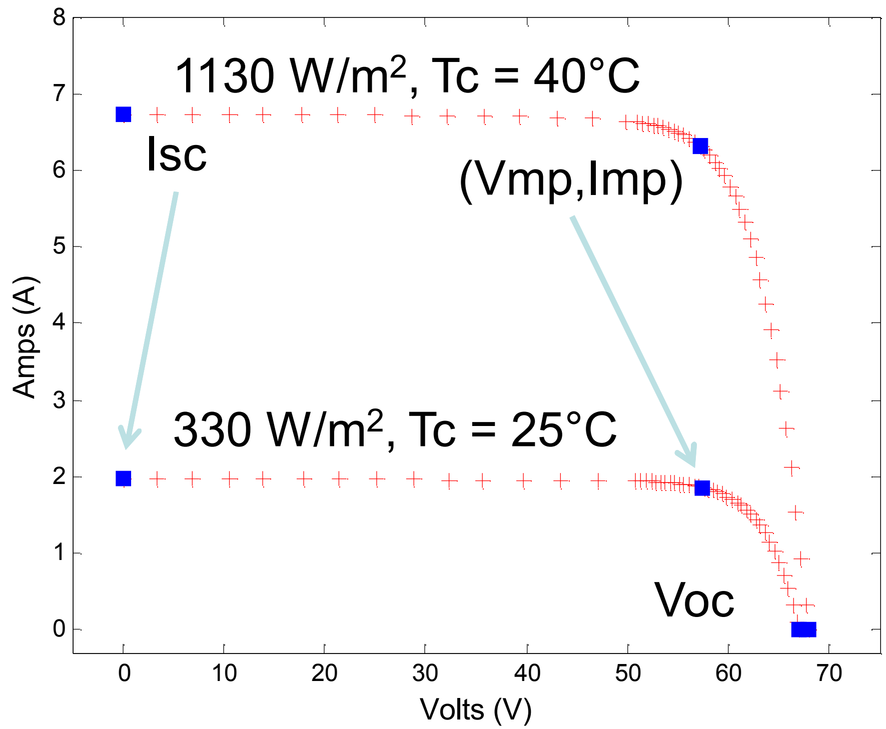


PV_LIB functions

- pvl_sapmcelltemp

5. Module IV Curve Models

- Don't confuse a model with the software that implements it
- Models predict DC voltage and current over the range of POA irradiance and cell temperature
  - **IV curve models (aka 'diode' models)**
    - E.g., '5 parameter model'
  - **Point models (focus on P_{MP})**
    - E.g., Sandia model
    - Loss Factors Model (LFM)
  - **Simple efficiency**
    - E.g., PVWatts
  - **Power rating model**
    - E.g., Huld



$$P_{dc} = \frac{I_{tr}}{1000} P_{dc0}(1 + \gamma(T_{cell} - T_{ref}))$$

$$P_{DC} = Ee \times (P_{MP0} + k_1 \log(Ee) + k_2 \log^2(Ee) + k_3(T_m - T_0) + k_4(T_m - T_0)\log(Ee) + k_5(T_m - T_0)\log^2(Ee) + k_6(T_m - T_0)^2)$$

The Sandia Array Performance Model

- Describes module output at SC, OC and MP points
- As a function of beam and diffuse irradiance (E_b and E_{diff}), cell temperature (T_C), air mass (AM_a) and angle of incidence (AOI)
- 14 empirical coefficients, 2 empirical functions (f_1 and f_2)
- With exception of f_2, coefficients determined for individual modules

$$E_e = f_1(AM_a)\left(E_b f_2(AOI) + E_{diff} f_d\right)$$

Effective irradiance: light flux that is available for conversion to electrical current

$$I_{SC} = I_{SC0} E_e \left(1 + \alpha_{SC}(T_C - T_0)\right)$$

$$V_{OC} = V_{OC0} + N_S n\delta(T_C)\ln(E_e) + \beta_{OC}(T_C - T_0)$$

$$V_{MP} = V_{MP0} + C_2 N_S n\delta(T_C)\ln(E_e) + C_3 N_S \left(n\delta(T_C)\ln(E_e)\right)^2 + \beta_{MP}(T_C - T_0)$$

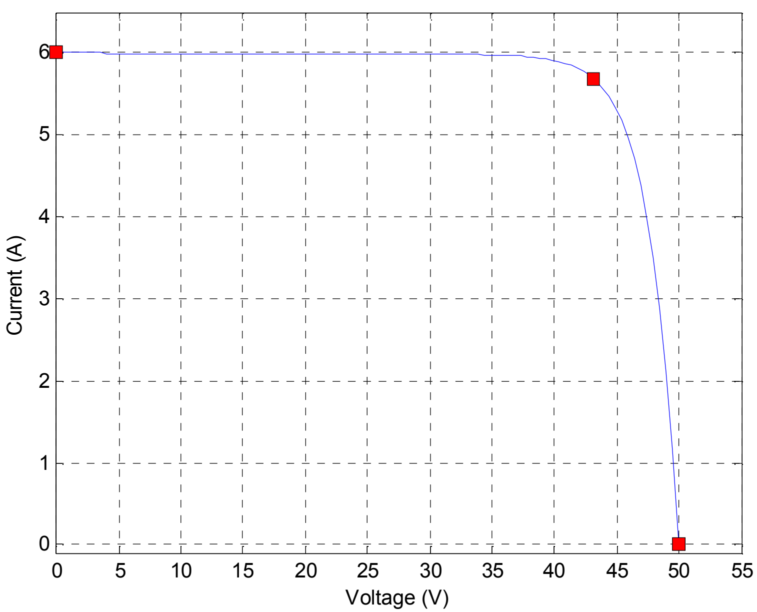
$$I_{MP} = I_{MP0}\left(C_0 E_e + C_1 E_e^2\right)\left(1 + \alpha_{MP}(T_C - T_0)\right)$$

PV_LIB functions
- pvl_sapm

Single Diode Models

PV_LIB functions
- pvl_singlediode
- pvl_calcparams_desoto
- pvl_calcparams_PVsyst
- Pvl_calcparams_CEC

- CEC, PVsyst, PV*SOL, others
  - IV curve described by single diode equation
  - "5 parameters" – for each IV curve
  - Additional equations describe how parameters change with effective irradiance *E*, temperature T_C



$$I = I_L - I_0\left[\exp\left(\frac{V+IR_s}{nV_T}\right)-1\right] - \frac{V+IR_s}{R_{sh}}$$

$$I_L(E,T_C) = \frac{E}{E_0}\left[I_{L0} + \alpha_{Isc}(T_C - T_0)\right]$$

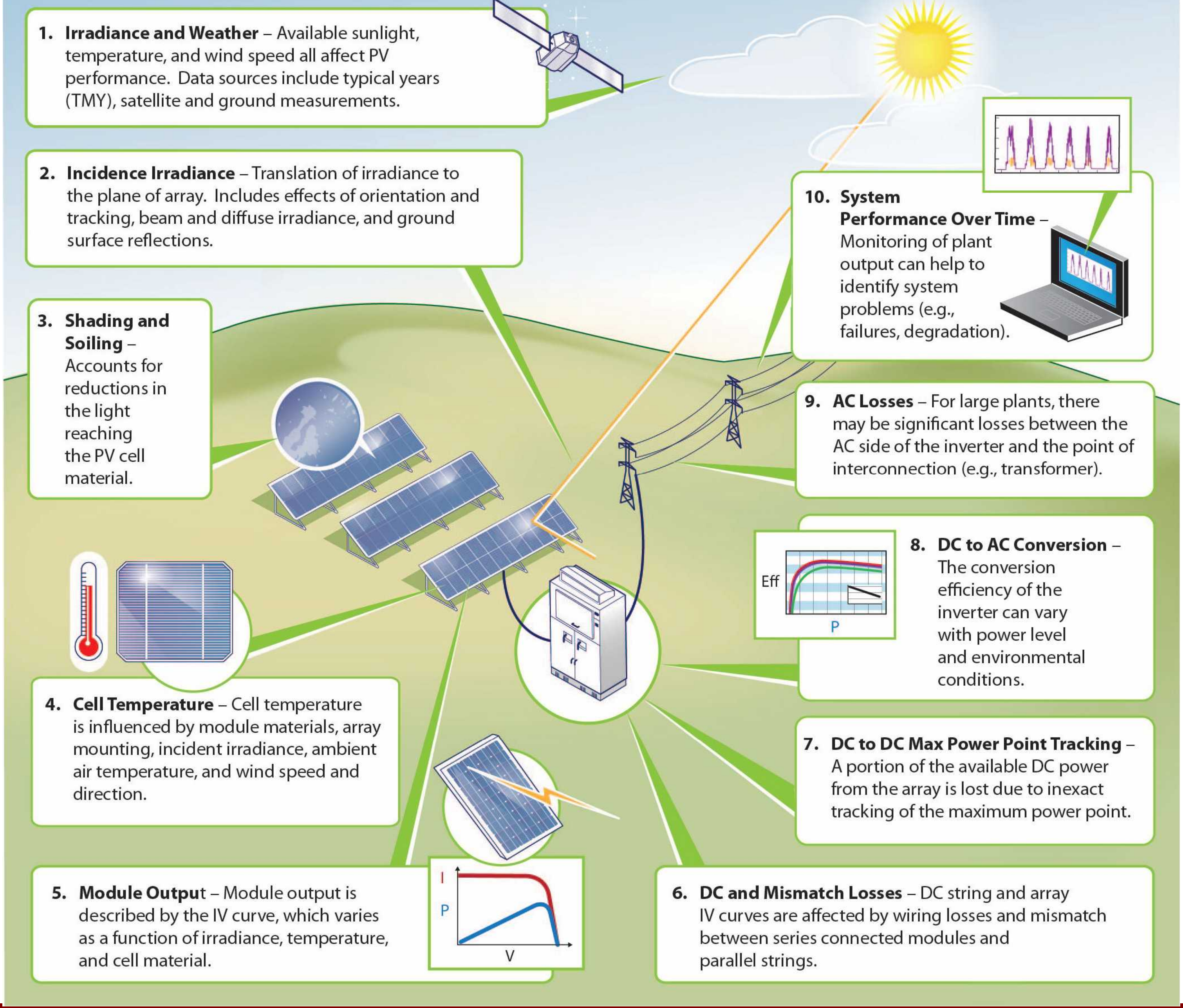
$$I_O = I_{O0}\left(\frac{T_C}{T_0}\right)^3 \exp\left(\frac{1}{k}\left(\frac{E_{g0}}{T_0} - \frac{E_g(T_C)}{T_C}\right)\right)$$

$$R_{sh} = R_{sh0}\frac{E_0}{E}$$

R_s, n constant

De Soto et al, 2006

PV Performance Modeling Steps



Thank You!

jsstein@sandia.gov

http://solar.sandia.gov

http://PV.sandia.gov

http://pvpmc.sandia.gov

Extra Slides

A bit of History

- Matlab version started as an internal tool at Sandia in 2010-2011 developed to help standardize analyses across the PV group.
  - PVLIB Version 1.0 – May 2012 – 29 functions
  - PVLIB Version 1.1 – Jan 2013 – 38 functions
  - PVILB Version 1.2 – Dec 2014 – 44 functions
  - PVLIB Version 1.3 – Dec 2015 – 59 functions
- Python version was initially developed from 2013-2014 by Rob Andrews under contract from Sandia.
- 2015 Python PVLIB converted to Open Source GitHub project largely managed by Will Holmgren at University of Arizona.

- Download links available on PVPMC website:
  - https://pvpmc.sandia.gov click on Applications and Tools link

Two Versions of PVLIB

PVLIB Matlab

- Integrates with Matlab environment (help, search)
- Extensively tested by Sandia National Laboratories
- No extra toolboxes required

- Requires Matlab license ($$)
- Not fully integrated into GitHub (yet)
- Updates have been slow to be released.
- No formal way report/fix bugs except for email.

PVLIB Python

- Free
- Can be integrated with a huge ecosystem of Python libraries
- Comprehensive unit tests
- A real Python library, not just a wrapper with awkward syntax
- High-level features that do not (yet?) exist in PVLIB MATLAB
- A growing community on GitHub

- Getting started with Python, NumPy, SciPy can be challenging
- Not as many functions as PVLIB MATLAB

New Functions in Matlab V.1.3

- **pvl_FSspeccorr** – Spectral mismatch modifier function contributed by First Solar based on precipitable water.
- **pvl_calcPwat** – function to estimate precipitable water content
- **pvl_huld** – PV performance model of Huld et al., 2011
- **pvl_PVsyst_parameter_estimation** – function to estimate PVsyst module parameters from IV curves.
- **pvl_calcparams_PVsyst** – Calculates the five parameters for an IV curve using the PVsyst model.
- **pvl_desoto_parameter_estimation** – function to estimate Desoto module parameters from IV curves.
- **pvl_getISDdata** – Functions to access ground measured weather data from NOAA's Integrated Surface Data network

New Functionality in Python V0.2, V0.3

- **PVSystem** and **SingleAxisTracker** classes – abstractions that can help with standard modeling tasks.
- **Standardized variable names** throughout library
- **Conda** installation packages on the pvlib and conda-forge channels
- **Location.from_tmy** – create a Location object from a TMY file
- **lookup_linke_turbidity** – refactored out of ineichen, supports daily monthly → daily interpolation
- **Ported more functions from PVLIB MATLAB**

see documentation for full listing

pvlib-python.readthedocs.io/en/latest/whatsnew.html

PVsyst PAN File Example

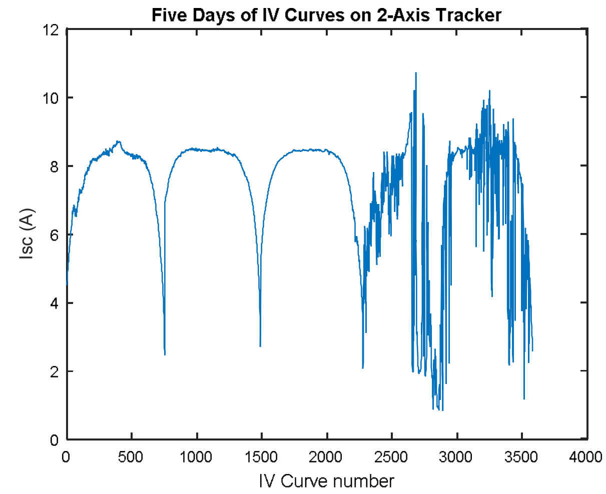
- 3,585 IV curves measured outdoors over 5 days on a 36 cell Mitsubishi c-Si module
- We use one function to estimate parameters:
  - pvl_PVsyst_parameter_estimation

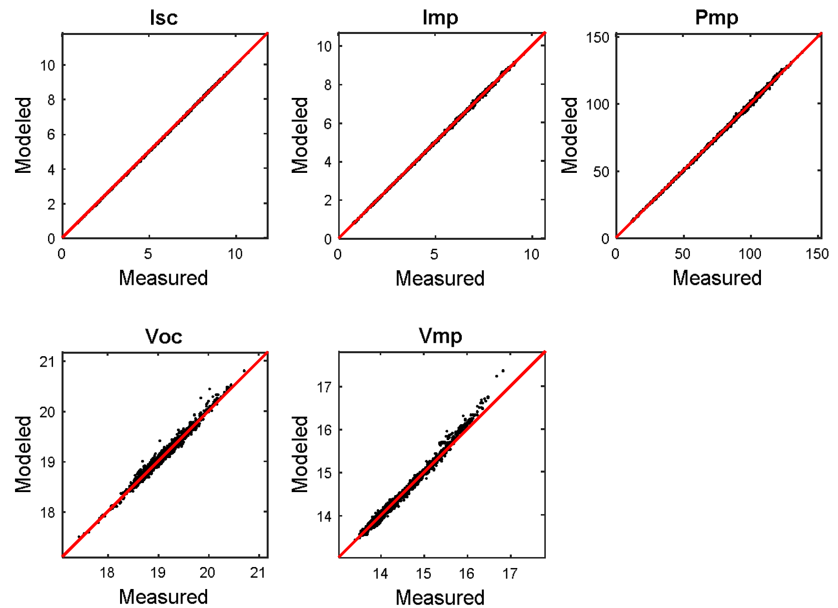
```
[PVsyst oflag] =
pvl_PVsyst_parameter_estimation
(IVCurves, Specs, Const,
maxiter, eps1, graphic);
```

- And two functions to run the model
  - pvl_calcparams_PVsyst
  - pvl_singlediode

```
[IL, Io, Rs, Rsh, nNsVth] =
pvl_calcparams_PVsyst([IVCurves
.Ee],[IVCurves.Tc],Specs.aIsc,P
Vsyst);

Modeled = pvl_singlediode(IL,
Io, Rs, Rsh, nNsVth);
```



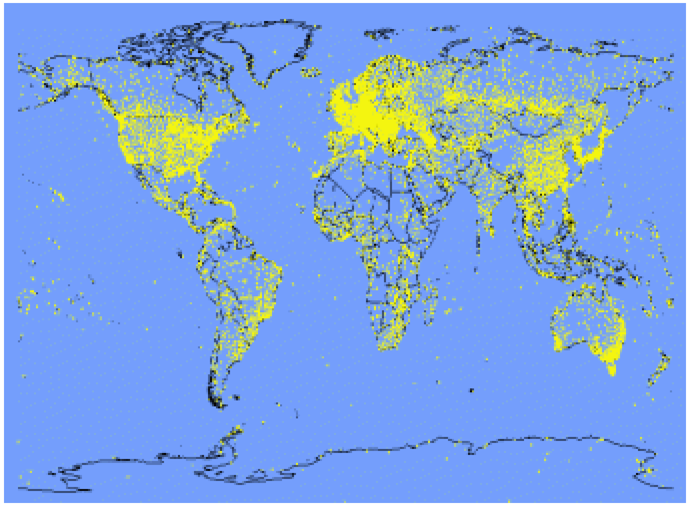


Finding Weather Data using PVLIB

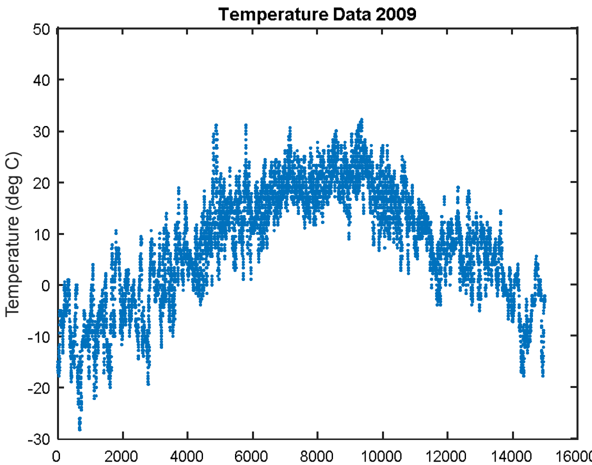
- Two functions allow users to obtain measured weather data from NOAA's Integrated Surface Database (ISD)
  - pvl_getISDdata
  - pvl_readISH
- Example below shows how to retrieve data for a specified location and year. The functions will find the closest station.
- We chose Williston, Vermont and 2009. It found a station within a few kms.

```
fname = pvl_getISDdata(44.465,-
73.105,2009,archive);

data = pvl_readISH([archive '\'
fname]);
```



Map showing the locations of ISD stations

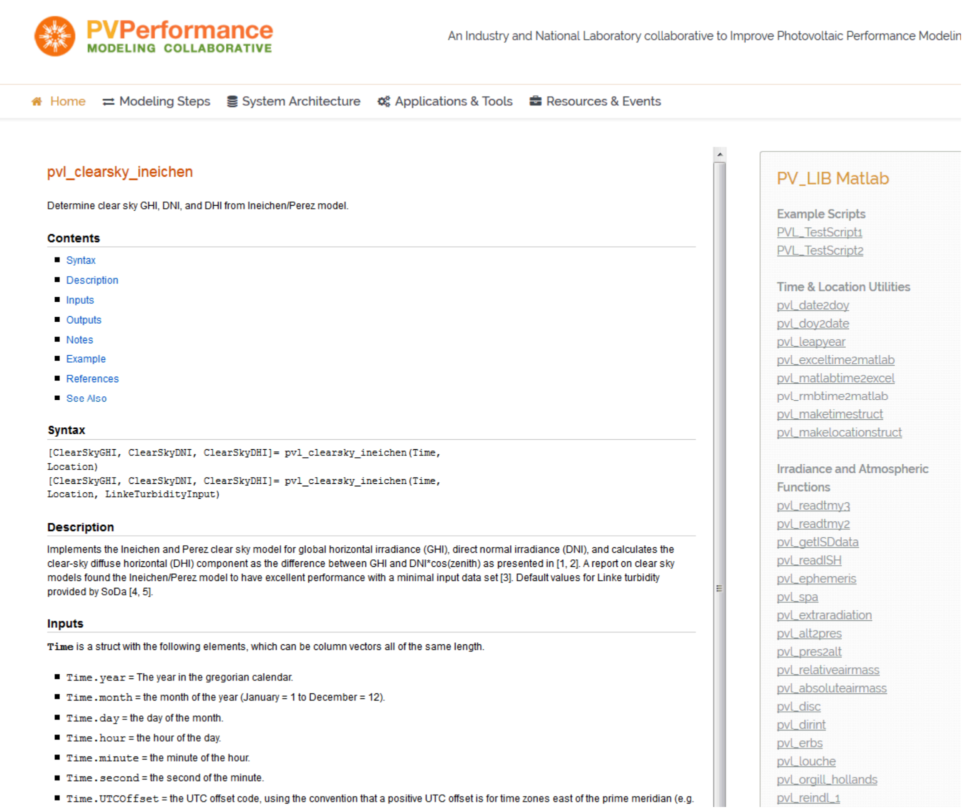


Plot of air temperature data near the point of interest from 2009 in Williston, VT

PVLIB Documentation

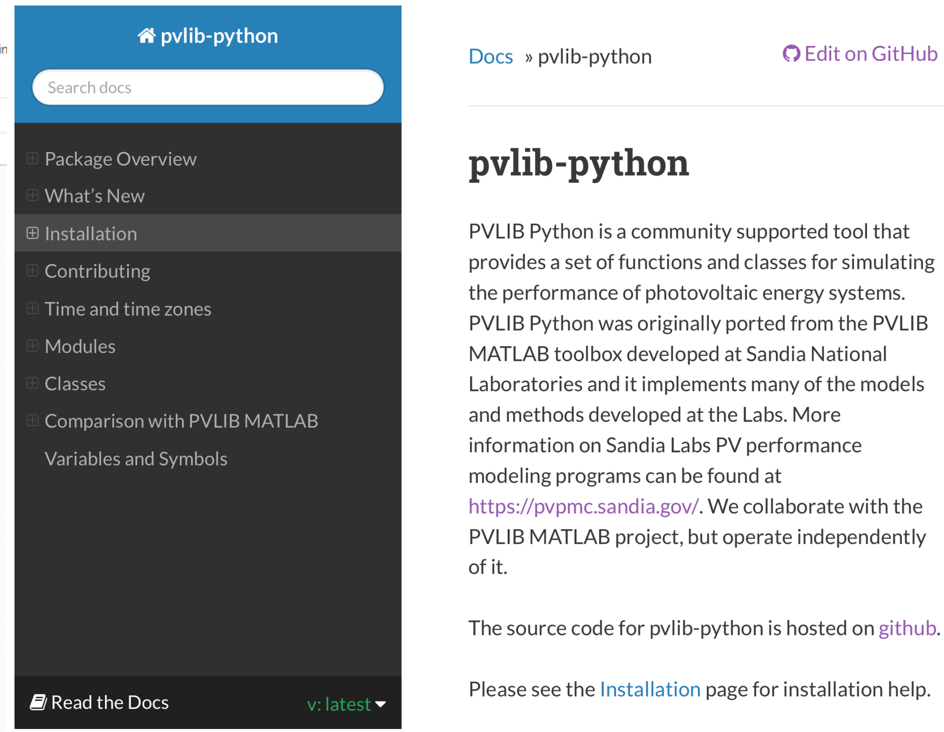
Matlab

pvpmc.sandia.gov/applications/pv_lib-toolbox/



Python

pvlib-python.readthedocs.io

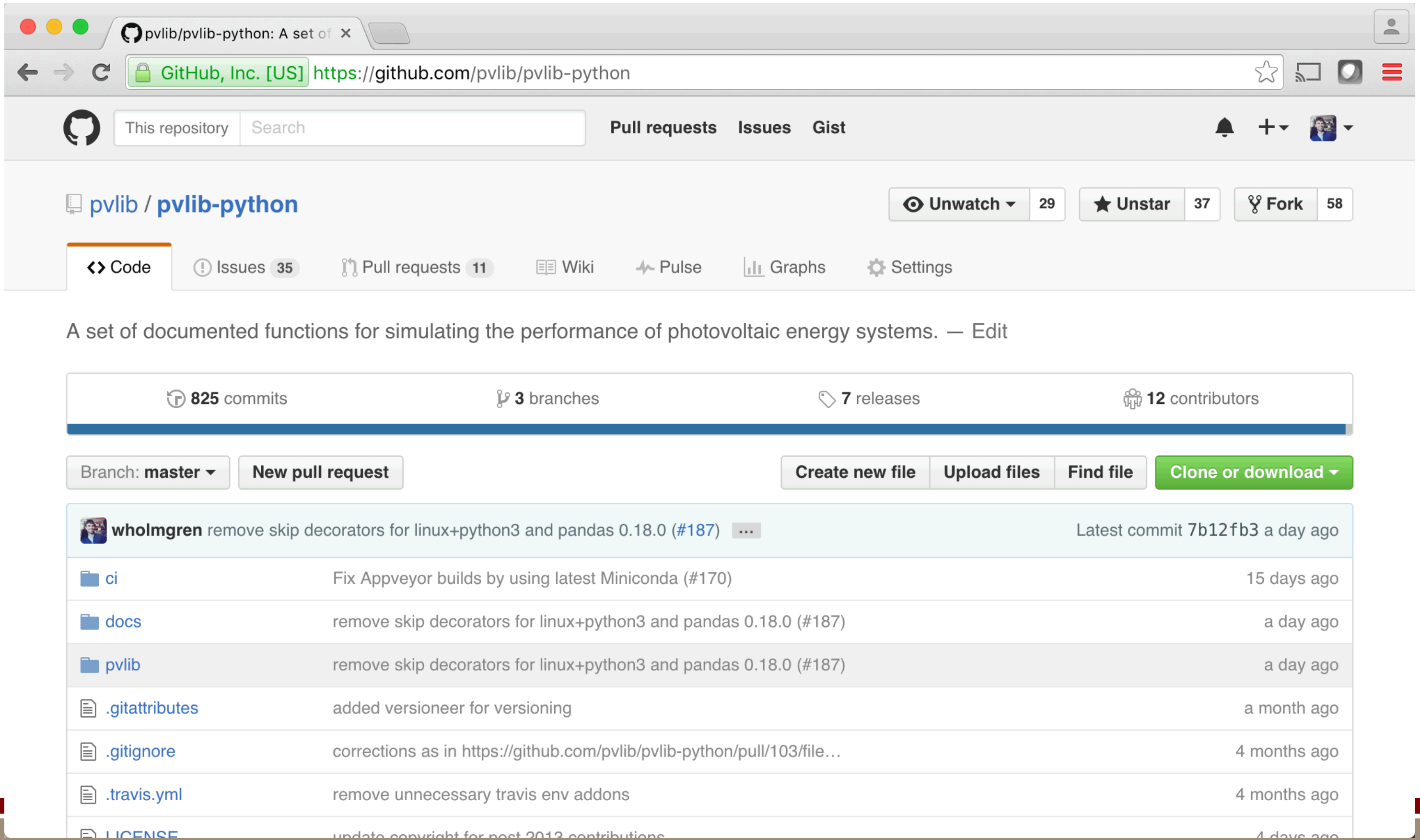


PVLIB on GitHub

github.com/pvlib/pvlib-python

github.com/sandialabs/MATLAB_PV_LIB

Please go here!



Challenges for PVLIB going forward

- Maintain sufficient overlap between the MATLAB and Python libraries
- Recruit new developers and maintainers
- High quality open source software takes time, therefore money
- Users must contribute their knowledge and expertise back to the community
  - Code is great, but discussion, documentation equally important!
- Industry must allow its developers spend time to improve the library.
  - Thanks First Solar, SunPower, SolarCity employees – need more!
- Funding agencies must appreciate the value, PVLIB community must communicate the value.
  - Value includes both technical merit and broader impacts
  - Difficult to quantify value