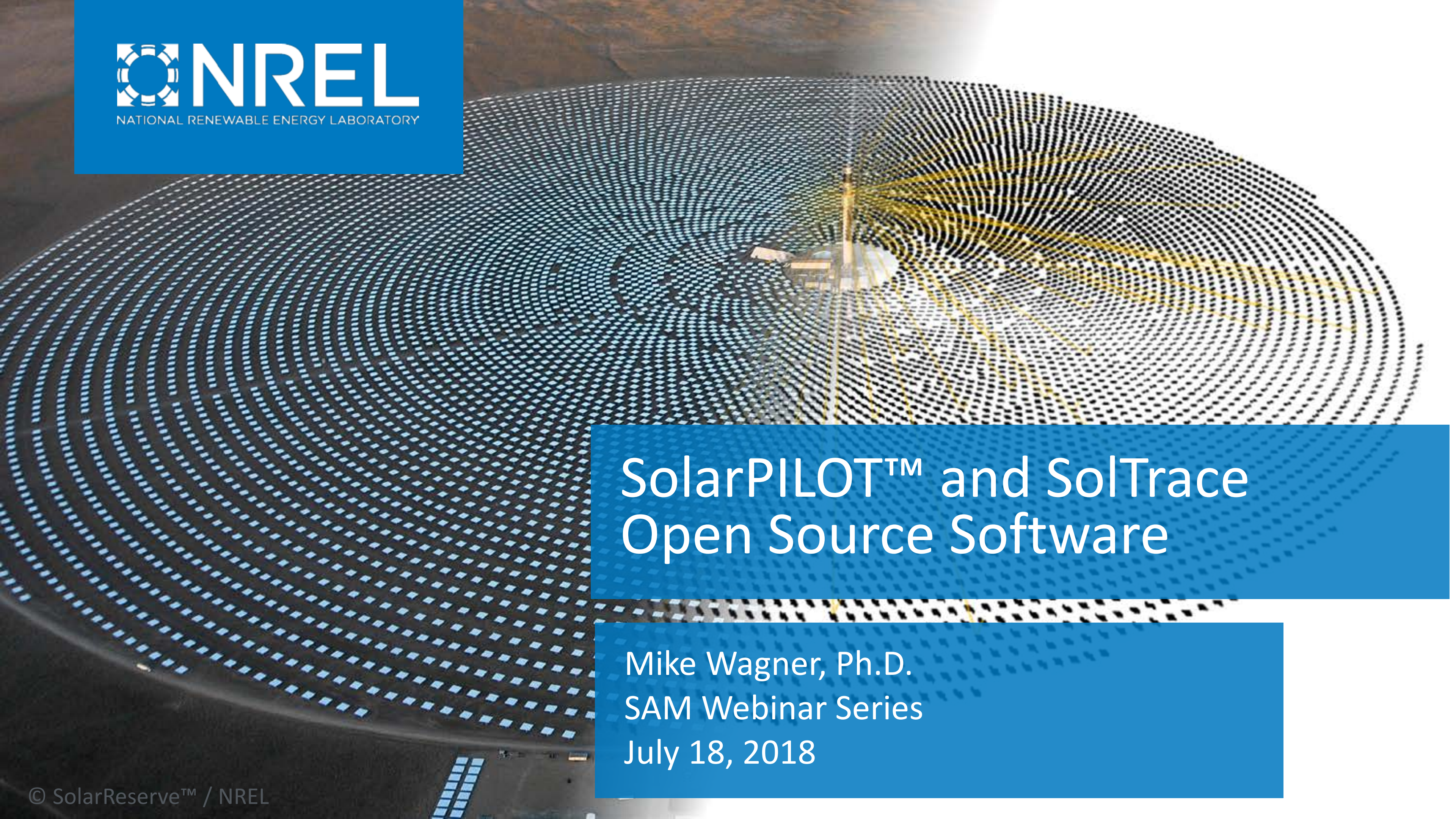


NREL
NATIONAL RENEWABLE ENERGY LABORATORY

SolarPILOT™ and SolTrace Open Source Software

Mike Wagner, Ph.D.
SAM Webinar Series
July 18, 2018

Abstract

NREL is excited to announce that its SolarPILOT™ and SolTrace software packages have both recently been made open source software! Join Dr. Mike Wagner for a webinar in which he will present recent updates to both software packages, the workflow for building and/or contributing to the open source projects, and an introduction to software usage. SolarPILOT™ is NREL's tool for the design, characterization, and optimization of concentrating solar power tower solar field geometry. It is capable of modeling optical performance either with an analytical algorithm, or with Monte-Carlo ray tracing using SolTrace's embedded 'coretrace' engine. SolarPILOT is included in SAM's power tower models in limited form. While SolarPILOT is used by SAM's power tower models, the stand-alone user interface offers substantially more functionality; features include scripting, optimization, flux intensity plotting, complex land boundaries, multiple heliostat or designs, and others. SolTrace is a general ray tracing tool for concentrating optics, and has been in use by NREL, industry, and researchers for nearly 20 years.

The webinar will consist of three parts. In the first half hour, Mike will cover updates to SolarPILOT, including review of the optimization tool and field plotting tool. The second half hour will include a review of SolTrace and demonstrate its use in modeling a power tower system. In the remaining time, Mike will cover the open source projects and allow time for questions and answers.

Outline

- Introduction and overview
- SolarPILOT
  - Overview and recent development
  - Optimizing power tower systems [example]
  - Scripting [example]
- SolTrace
  - Overview and recent development
  - Modeling power tower optical performance
- Open source project overview
- Retrieving and compiling the code
- Contributing
- Q&A

Registration links and webinar recordings

NREL
NATIONAL RENEWABLE ENERGY LABORATORY

System Advisor Model (SAM)

Login | Register

Search

HOME DOWNLOAD SDK SUPPORT RESOURCES CONTACT ACCOUNT ABOUT

Webinars and Videos

published by Paul Gilman on Thu, 2015-05-21 14:51

Upcoming Webinars

SAM webinars are one-hour lectures with question and answer sessions that cover various topics about SAM. All webinars are free and start at 2 pm Mountain.

- NREL's SolarPilot™ and SolTrace Open-source Software, July 18, 2018 2:00 PM MDT

Round Tables

SAM round tables are 30-minute informal discussions online with the SAM team scheduled every other Thursday at 2:30 pm Mountain Time.

- Register for free

How-to Videos

How-to videos are short demonstrations of specific tasks in SAM.

- Sizing a PV-plus-battery system in SAM
- P50/P90 Analysis in SAM 2017.9.5
- Import Electric Load Data
- Generate SDK code from SAM
- Overview of SAM Open Source repositories

Recordings of Past Webinars

Video recordings and presentation materials are available for the following webinars.

Open Source Project

https://sam.nrel.gov/webinars

Presenter Bio

Mike Wagner

- Mechanical Engineer and Principal Investigator, 2009-current
- Author of SolarPILOT and several CSP models in SAM
- Other research includes system design, operations, and maintenance optimization; Next-gen CSP technology, gas phase HTF system development
- Author of 40+ conference and journal publications
- Education: BS/MS University of Wisconsin, PhD Colorado School of Mines

https://www.nrel.gov/research/michael-wagner.html

Acknowledgement: **Tim Wendelin**

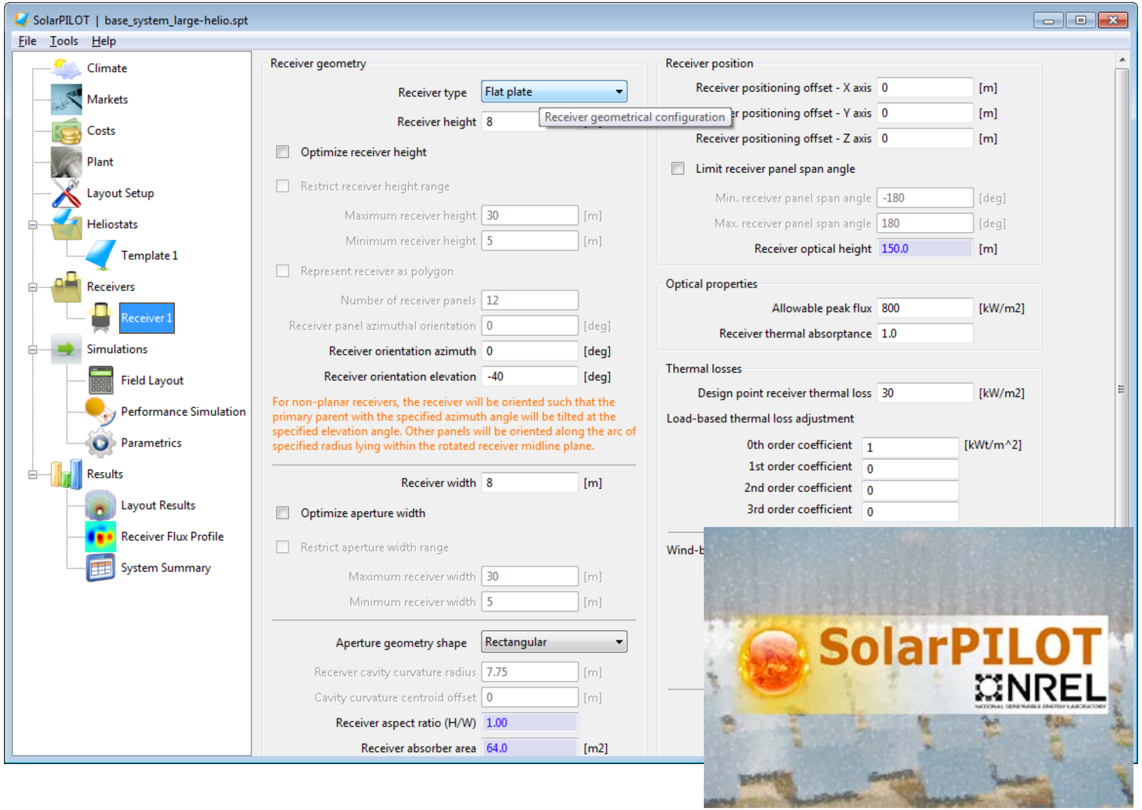
- Lead author of SolTrace
- Sr. Engineer at SolarDynamics (2017-current)
- NREL (retired, 1984-2017)

SolarPILOT

Overview and recent development

Solar Power tower Integrated Layout and Optimization Tool

- Definition and characterization of power tower CSP solar fields
- Research and design tool for power tower optical subsystem
- User interface tied into detailed simulation code
- Developed at NREL with DOE funding
- Integrated into SAM as the power tower optical characterization engine



Wagner, M.J., and Tim Wendelin. "SolarPILOT : A Power Tower Solar Field Layout and Characterization Tool." *Solar Energy* 171, no. September 2018 (2018): 185–96.

What is SolarPILOT?

What it is…

A tool for

- creating heliostat field layouts and characterizing optical performance
- screening potential development sites
- optimizing solar field design parameters
- investigating heliostat performance in detail
- optical calculations using both ray trace and analytical routines
- use by researchers, developers, and universities

What it is *not*…

A tool for

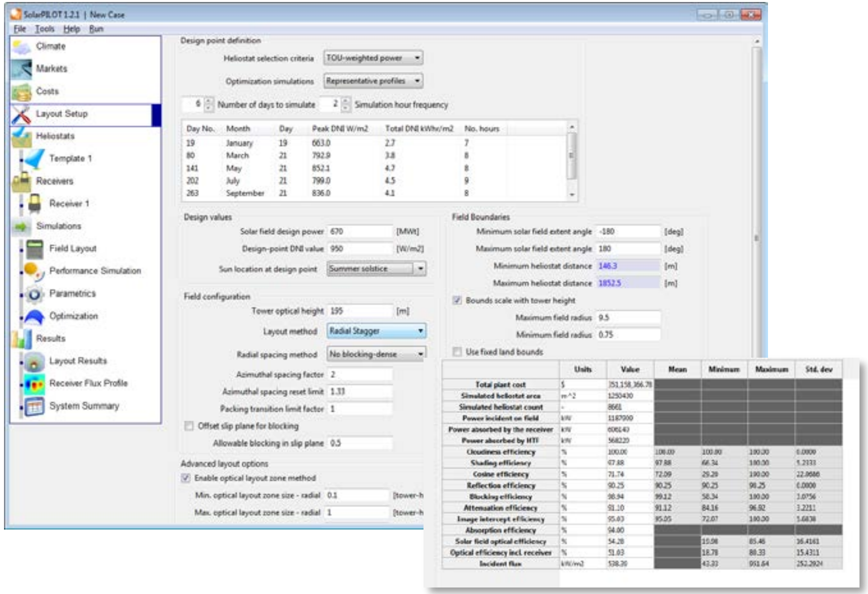
- annual solar field production analysis
- power block or thermal energy storage simulation
- detailed cost analysis
- financial modeling
- ignoring due diligence
- always getting the right answer

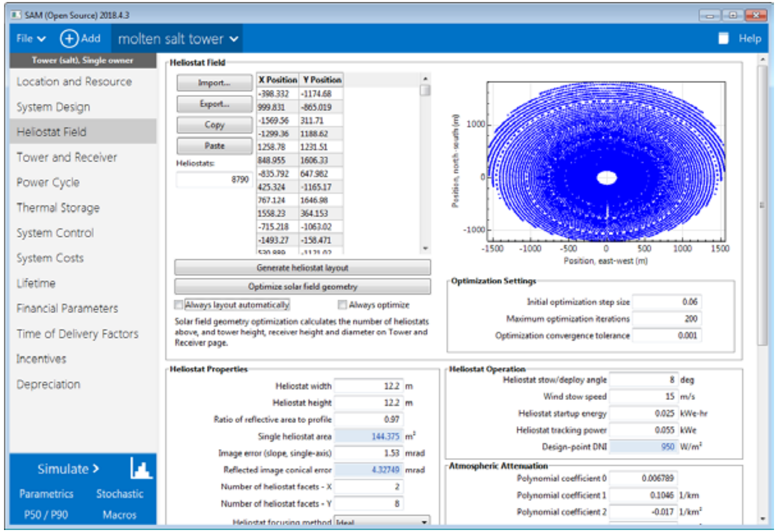
Capabilities

- Create solar field layouts, or import heliostat positions manually
- Simulate optical performance of the solar field or of individual heliostats
- Simulate different receiver geometries, multiple receivers
- Characterize receiver flux profiles at different solar positions and irradiance levels
- Impose heliostat aiming algorithms for flux profile control
- Optimize a wide range of parameters for lowest cost of energy
- Plot field performance and flux intensity
- Script layout and simulation actions using LK
- Simulate with analytical & ray-trace methods
  - Direct integration with NREL's **SolTrace**

Standalone versus SAM-integrated tool

- SolarPILOT comes in two forms
  - Standalone tool that provides substantial functionality
  - SAM power tower characterization engine with limited functionality



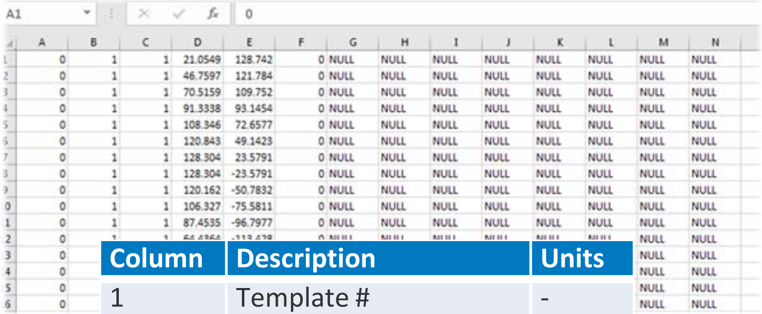


Field layout process

To generate a new layout, SolarPILOT:

1. **Collects** inputs and settings from the input pages.
2. **Pre-processes** the weather data or solar irradiance model to generate a set of simulation points for design evaluation.
3. **Generates a list of all possible heliostat positions** that satisfy the land boundary and layout rules that you choose on the Layout Setup page.
4. **Simulates the performance of the field** - including each possible heliostat in the layout from step (3) - over the set of simulation points generated in step (2).
5. **Sorts all possible heliostats by the performance metric** that you specify in the Heliostat selection criteria field on the Layout Setup page.
6. **Calculates the power delivered** by each heliostat at the **reference condition** by running a single simulation using the reference values specified on the Layout Setup page.
7. **Deletes heliostats in order of worst-performing to best-performing** while the power delivered at the reference condition is still satisfied.

User-specified heliostat positions

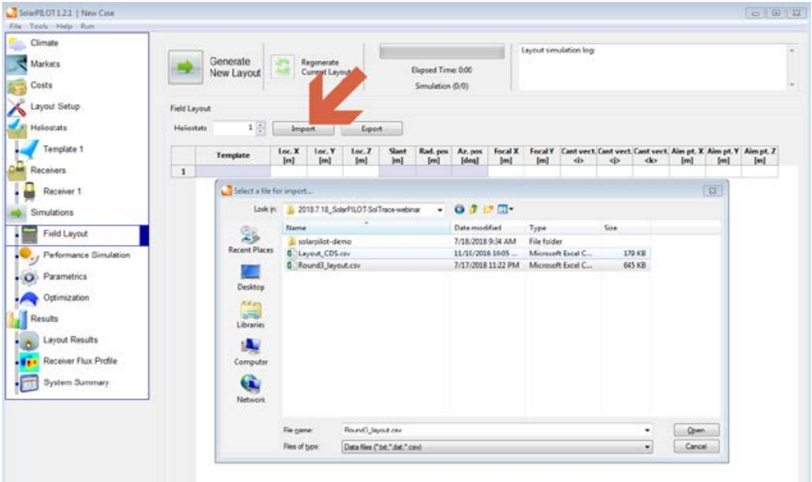


	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	0	1	1	21.0549	128.742	0	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
2	0	1	1	46.7597	121.784	0	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
3	0	1	1	70.5159	109.752	0	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
4	0	1	1	91.3338	93.1454	0	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
5	0	1	1	108.346	72.6577	0	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
6	0	1	1	120.843	49.1423	0	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
7	0	1	1	128.304	23.5791	0	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
8	0	1	1	128.304	-23.5791	0	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
9	0	1	1	120.162	-50.7832	0	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
10	0	1	1	106.327	-75.5811	0	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
11	0	1	1	87.4535	-96.7977	0	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
12	0												NULL	NULL
13	0												NULL	NULL
14	0												NULL	NULL
15	0												NULL	NULL
16	0												NULL	NULL

Column	Description	Units
1	Template #	-
2	Heliostat is enabled	-
3	Heliostat at position	-
4	Location (x)	m
5	Location (y)	m
6	Location (z)	m
7	Focal length (x)	m
8	Focal length (y)	m
9	Cant vector (i)	-
10	Cant vector (j)	-
11	Cant vector (k)	-
12	Aim pt (x)	m
13	Aim pt (y)	m
14	Aim pt (z)	m

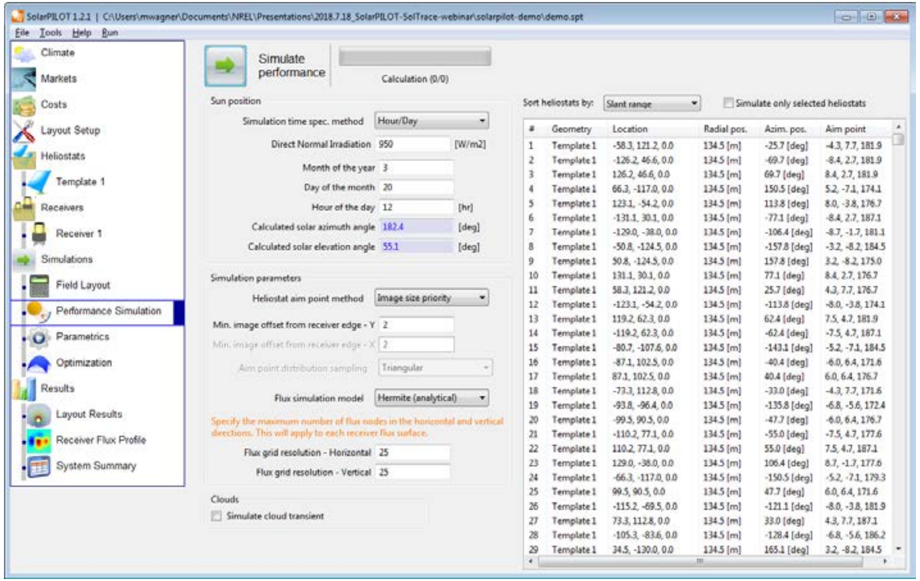
To import a layout:

- Specify the receiver, tower, and heliostat properties
- On the Field Layout page, click "Import"
- The selected file must be formatted according to the guidelines in the help menu!



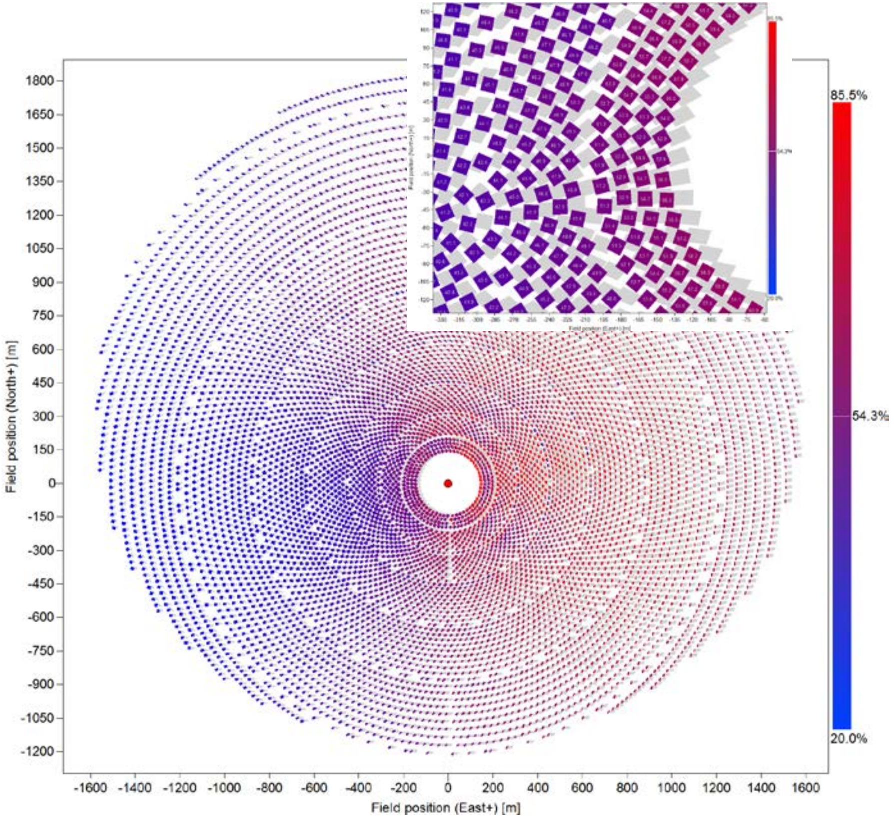
Performance calculations

- Ray Tracing (numerical)
  - Can model any surface-based geometry
  - Characterize flux intensity, directionality, and multiple surface interactions
  - Computationally intensive
- Analytical with Gaussian approximation
  - Rapidly compute flux characteristics and system efficiency
  - Ideal for use in layout and optimization calculations
  - Does not provide directionality detail
  - Not as accurate as ray tracing
- Two approaches are complementary



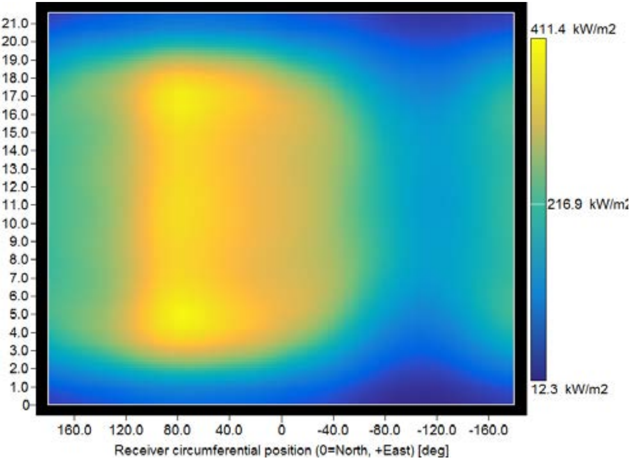
Field performance plot

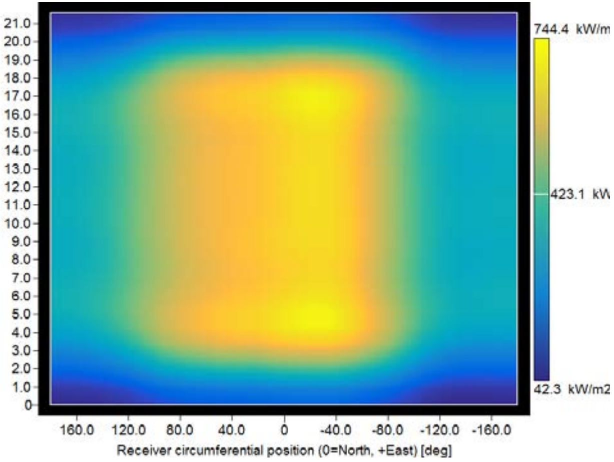
- Displays heliostat positions, land constraints, and efficiency values for each heliostat.
- Interactive selection of heliostats and removal from layout (new)
- Data can be exported into tabular format

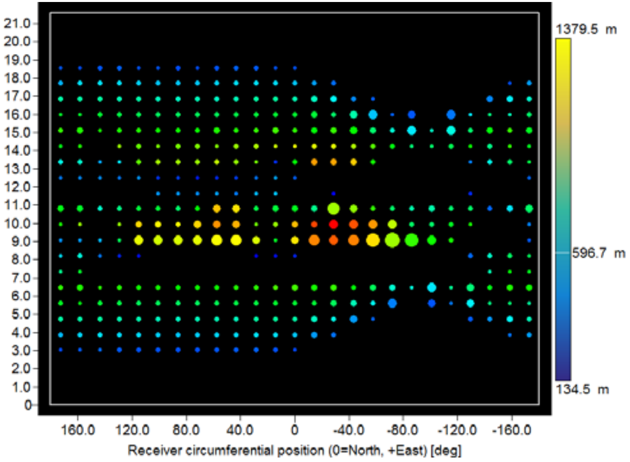


Flux and aim plots

- SolarPILOT generates flux intensity plots on the receiver surfaces
- The field can be simulated at any sun position or DNI value
- Plots can be scaled and adjusted for resolution
- The heliostat aim point plot illustrates the aiming strategy for the selected flux plot







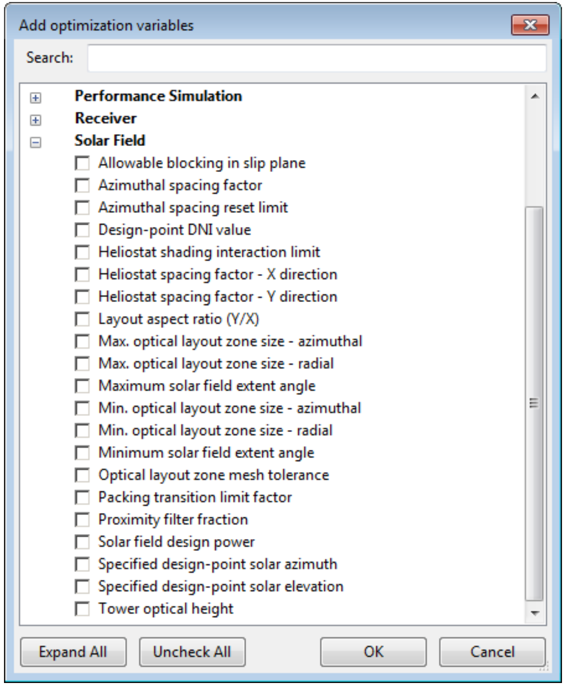
SolarPILOT

Optimizing power tower systems

Optimization overview

- Power tower design is complex, with many trade-offs!
- Many input parameter values can be optimized
- SolarPILOT uses the COBYLA algorithm (part of the NLOpt library) to optimize the selected variables
  - Represents the objective function as a multi-dimensional linear surface within a local trust region
  - Incorporates nonlinear constraint (allowable peak flux)
  - SolarPILOT calculates the actual peak flux for a given design and compares it with this value
  - The objective function seeks to minimize pseudo LCOE

$$obj = \frac{cost_{tot}}{E_{expect}} \cdot \left(1 + \left(1 - \frac{\dot{Q}_{sf}}{\dot{Q}_{sf,des}}\right) \cdot R_{penalty}\right)$$



Case study

- Consider a new plant near Phoenix, AZ
- An agricultural enterprise is selling cultivated land near transmission and with good DNI resource
- A developer would like to build a tower facility at the location, but needs to evaluate the property for power production potential, tower height, and tower location

SolarPILOT can help!

Case study steps: **Live Demo**

1. Identify the location, and download the weather file
2. Open SolarPILOT and specify parameters
   - Thermal power rating = 450 MWt
   - Climate file
   - Best guess tower height and position
   - Land constraint type
3. Create land geometry (Google Earth, e.g.)
4. Import land geometry and tower position
5. Run layout
6. Set up optimization table
7. Optimize
8. Apply (if satisfied)

Optimization variable settings	Initial	Lower bound	Upper bound	Initial step
Tower height [m]	150	None	None	10
Tower offset – x [m]	-200	-300	500	50
Tower offset –y [m]	400	-50	600	50

SolarPILOT

Scripting

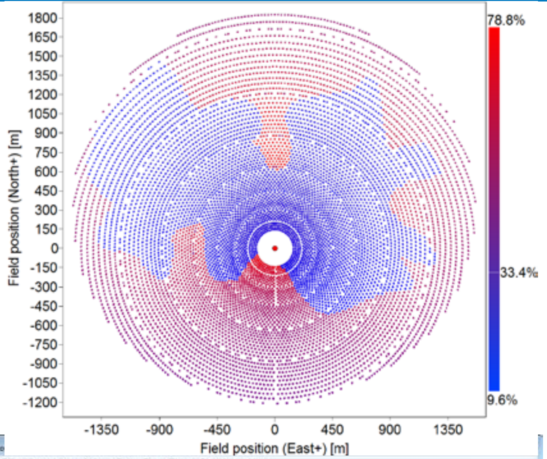
Scripting

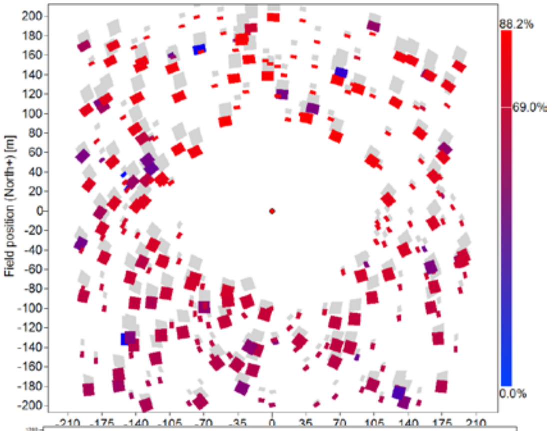
- SolarPILOT includes approximately 375 settings, input variables, and parameters that can be manipulated by the user
- A primary goal of the scripting interface is to provide a mechanism for assignment and retrieval of input and output parameters
- A wide range of tasks are achievable through scripting

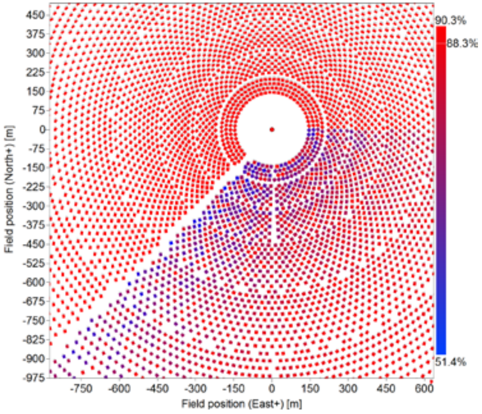
```
/*
Script to test modifying heliostat soiling
*/

//create an arbitrary function to affect soiling in an area near a road
function soil( loc )
{
        x = loc[0];
        y = loc[1];

        if( y > 0 || x < y)
                return 1.;

        return 1. - 0.5*rand()*exp(-abs(x-y)/500);
} //.. end of function

//add the access road and assign variables
radmax = 2000; rw=40;
add_land_area([[0,0],[-radmax,-radmax],[-radmax,-radmax-rw],[0,-rw]], 'exclusion', false);
run_layout();           //Generate a new heliostat field layout
hels = get_layout();    //get information about all heliostats in the layout
//modify the soiling factor based on location
refs = []; hids = [];
for(i=0; i<#hels; i++)
{
        refl = 0.95 * soil( [hels[i][1],hels[i][2]] ); //call the soiling function
        hids[i] = hels[i][0];
        refs[i] = refl;
}

modify_heliostats(hids, {'soiling'=refs});    //update heliostat soiling
run_performance();    //run a performance simulation
update_interface();   //update the solarpilot GUI with the results
```

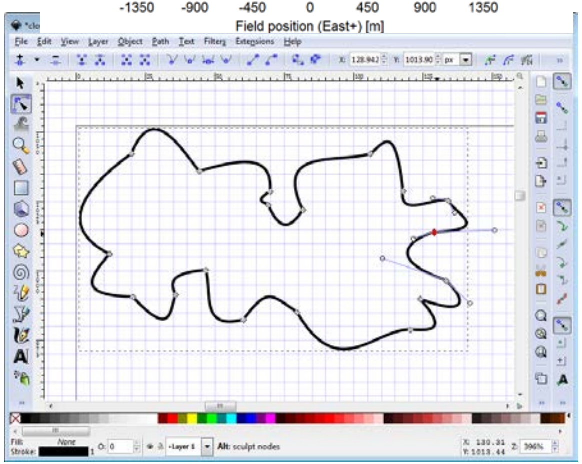
Scripting examples

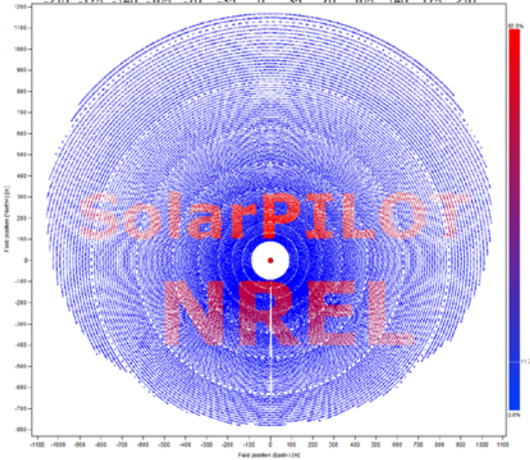
- Local soiling
- Cloud shapes
- Random templates
- Enabling/disabling











SolTrace

Overview and recent development

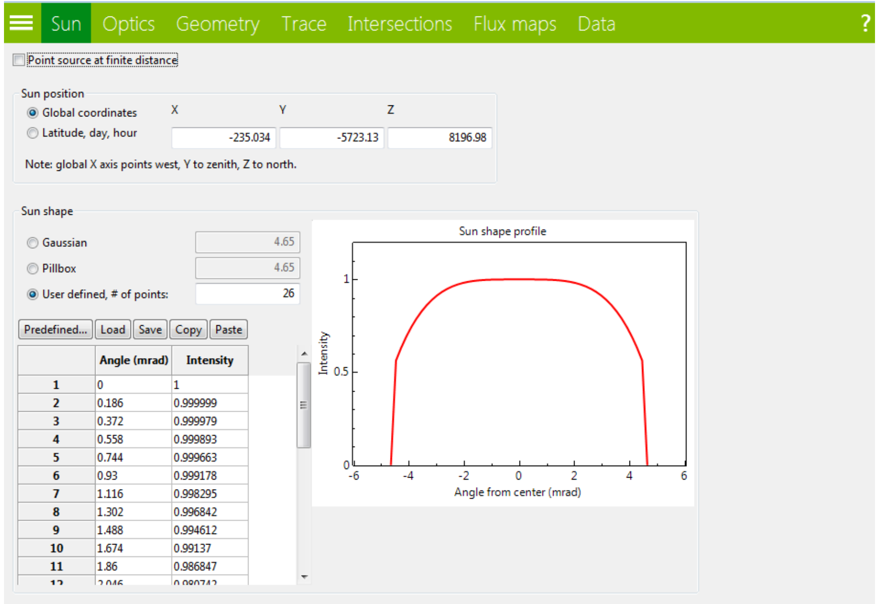
Overview

- A specified number of rays are traced from the sun through the system
- Each traces through the defined system while encountering various optical interactions
- Rays replicate real photon interactions and can therefore provide accurate results for complex systems that cannot be modeled otherwise
- Accuracy increases with the number of rays traced, and larger ray numbers means more processing time
- Organization:
  - The optical system is organized into “stages”
  - A stage is comprised of “elements” consisting of a surface, an optical interaction type, an aperture shape, and, if appropriate, a set of optical properties
  - Two characteristics completely define the “sun” as the light source: the angular intensity distribution of light across the sun’s disk, referred to as the sun shape, and the sun’s position.

Wendelin, Tim. “SolTrace: A New Optical Modeling Tool for Concentrating Solar Optics.” *NREL/TP-5500-59163*. Golden, CO, 2013.

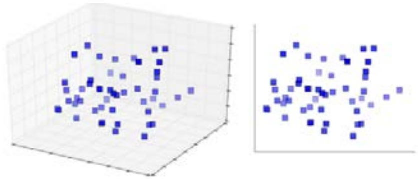
Recent work

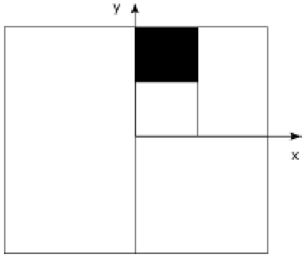
- User interface
  - The interface has been ported from Qt to wxWidgets and C++
  - The new GUI is consistent with SAM, SDKtool, and other CSP tools
  - The interface features enhanced ray plotting and is more responsive
- Speed improvement
  - NREL modified SolTrace to improve run times for power tower systems

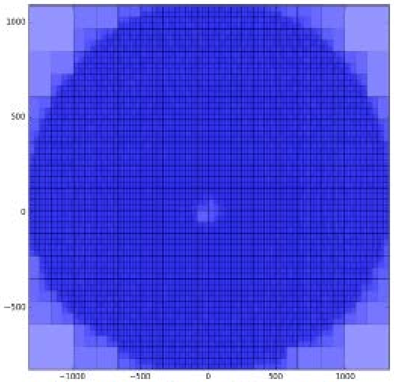


Speed improvement for point focus

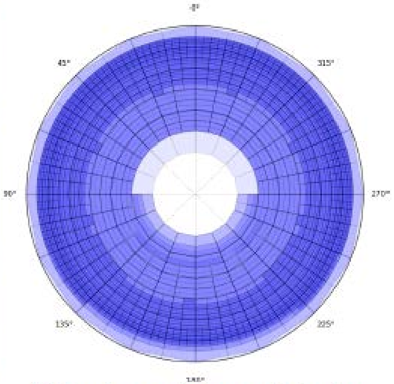
- SolTrace provides general ray-tracing functionality that is slow for systems with very large numbers of elements (e.g., power tower heliostat fields)
- Computational expense is reduced by creating localized groups of elements, a small subset of which are retrieved for intersection testing based on the position of the random sun ray
- Elements are grouped by their projection onto the plane normal to the sun vector using a binary search tree







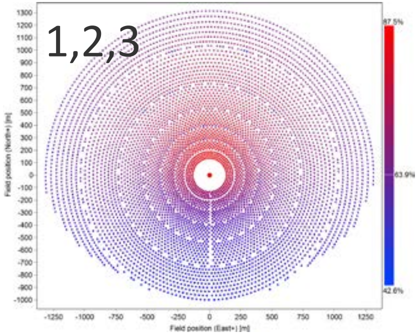
(a) Sun-projection mesh in cartesian coordinates.

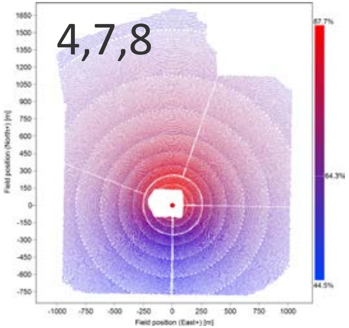


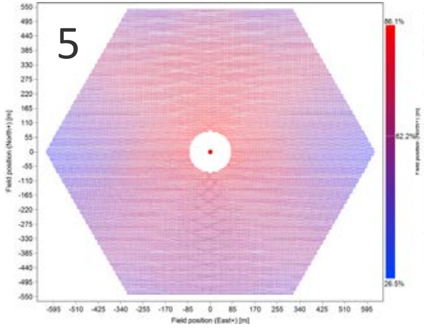
(b) Receiver-projection mesh in polar coordinates.

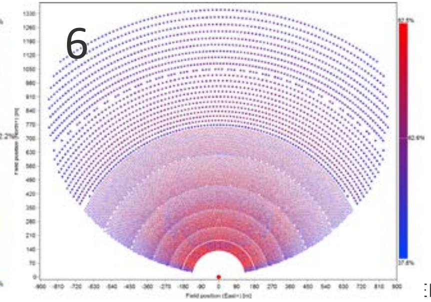
Speed improvement results

Case	# Elements	Description	Base [s]	New Time [s]	Improve. Factor
1	100,512	Default SolarPILOT case, 500 MWt	11,672	6.7	1742x
2	100,512	Same as Case 1, afternoon sun position, θ = 23°, α=253°	8,325	6.4	1300x
3	6,282	Default SolarPilot case with single-facet heliostats	927	2.5	370x
4	95,174	Ivanpah-like solar field	6,531	9.6	680x
5	34,188	eSolar-like hexagonal field	5,447	6.1	893x
6	79,812	North field, mixed heliostats (small near tower, large outer ring)	6,683	6.2	1078x
7	60,127	Ivanpah Unit 2 solar field with random vertical heliostat displacement	4,042	3.7	1092x
8	60,127	Ivanpah Unit 2 solar field with high optical slope error (~10 mrads)	3,830	3.7	1035x







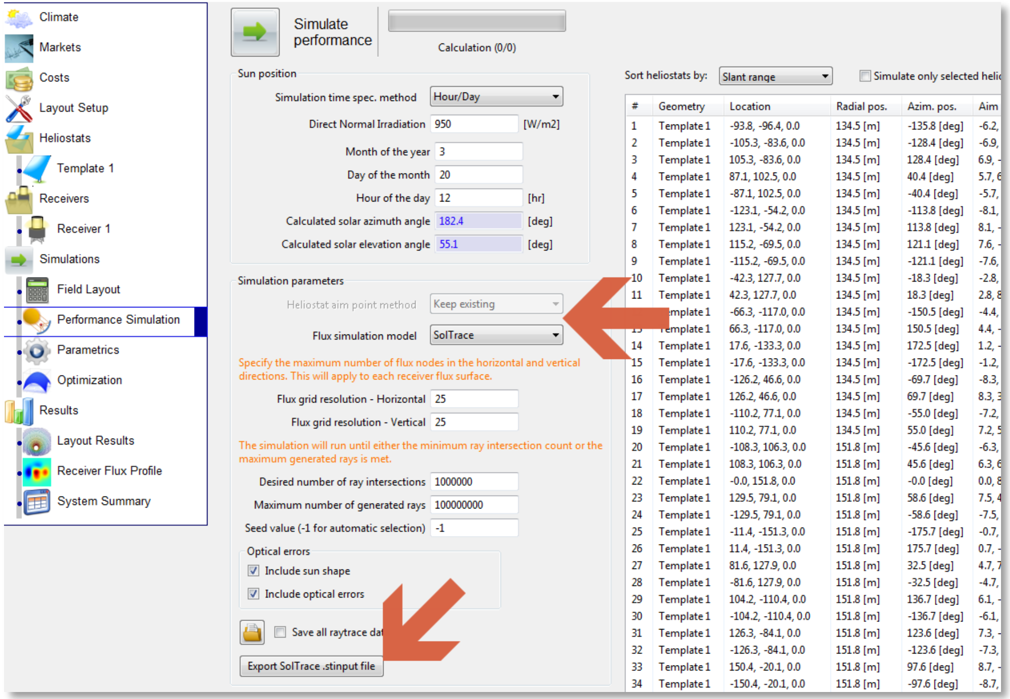


SolTrace

Modeling power tower optical performance

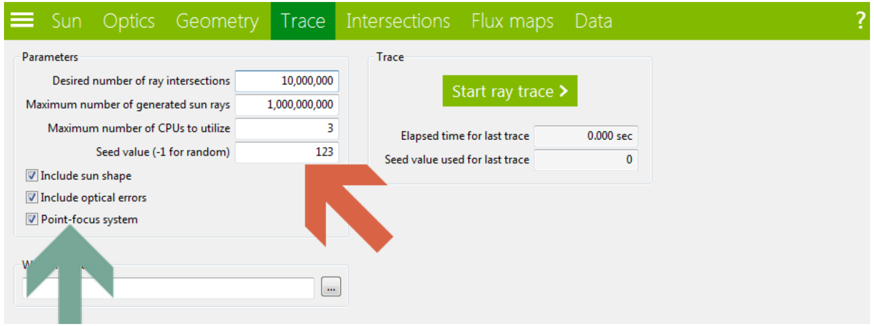
Creating SolTrace input from SolarPILOT

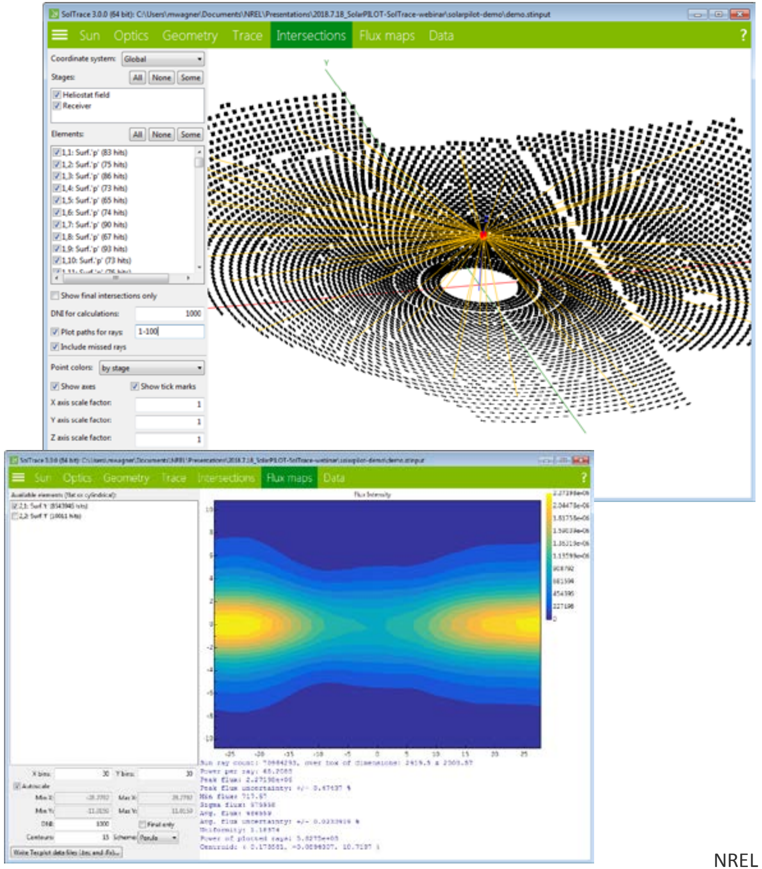
- Create a heliostat field layout with SolarPILOT
- Run a performance simulation with the analytical model to set aimpoints
- Select “Keep existing” for heliostat aim point method
- Select “SolTrace” for flux simulation model
- Export SolTrace .stinput file
- Open in SolTrace



Tracing and interpreting results

- Trace options should be customized for the analysis you are doing
- Make sure you run sufficient rays to achieve problem convergence
- Set the number of CPU's as desired
- For point focus systems, select the point focus option. This enables the speed improvement method.





Open source projects

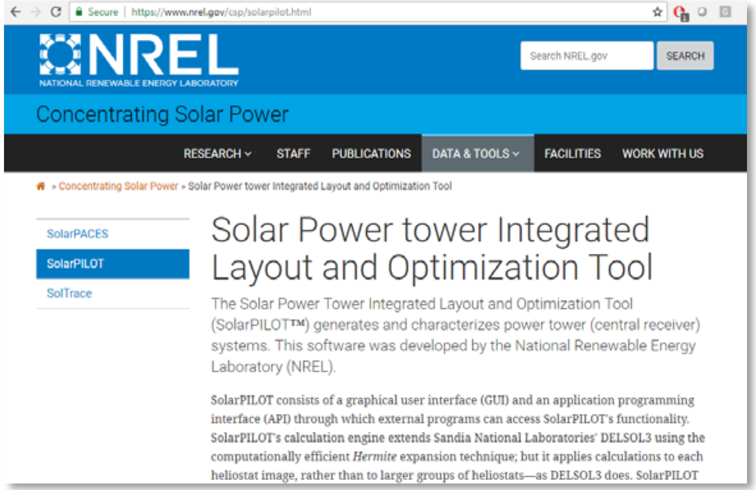
Overview

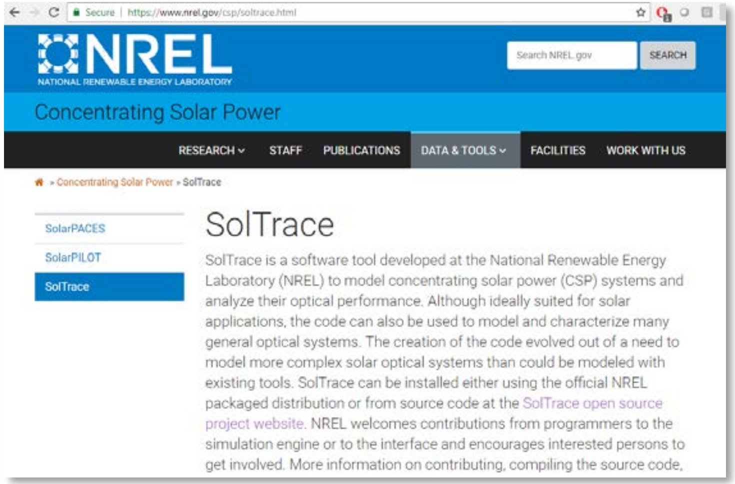
Motivation

- We are excited to continue working on SolarPILOT and SolTrace and fostering a new community of contributors.
- **Transparency**
  - Look at the underlying code that you are interested in
- **Flexibility**
  - Change the way a model works for research purposes
  - Change heliostat geometries to be specific to a developer
- **Collaboration**
  - Add new heliostat or receiver models
  - Add new layout techniques or error distributions
- We'd love to learn how you use NREL's open-source code! It helps us tailor our efforts and get funding to develop the tools.

NREL releases

- NREL will continue to maintain and release official desktop versions of SolarPILOT and SolTrace.
  - Releases built from the open-source repositories
  - User contributions can be considered for inclusion in official versions





Code licenses

- Licensed under a mixed MIT-type license and GPLv3 license.
- Commercial businesses can use SolarPILOT, SolTrace, SSC and SAM under the MIT-type restrictions
  - You can use these in software you develop for your business.
- Research entities, including national labs, institutions of higher learning, and non-profits are restricted under a GPLv3-type license.
  - You can use the code in your research, but must make your changes publicly available.

Code licenses

- Why the mixed license?
  - Want to encourage companies to use SolarPILOT and SolTrace as a foundation for growing their business in a fairly unrestricted way.
  - Want to encourage research institutions to share back any new innovations or make them publicly available so that the community as a whole benefits.
- Please see full licenses here:
  - https://github.com/NREL/SolarPILOT/blob/develop/LICENSE.md
  - https://github.com/NREL/SolTrace/blob/develop/LICENSE.md

Open source projects

Retrieving and compiling code

SolarPILOT code architecture

Core SolarPILOT code	SolarPILOT	Graphical user interface
	SSC/solarpilot	Analytical calculation routines
	coretrace	SolTrace calculation engine
NREL Libraries	WEX	Custom widgets for GUI
	LK	Scripting support
Development dependencies	wxWidgets	wxWidgets 3.1.0 – GUI framework
	C++ compiler	Visual Studio 2013, GCC 4.8.5 minimum, libc 2.17
	Operating system	Windows XP+, Linux (Ubuntu 14.04, Mint 18.2)

GoogleTest

SolTrace code architecture

Core SolTrace code

- SolTrace — Graphical user interface
- coretrace — SolTrace calculation engine

NREL Libraries

- WEX — Custom widgets for GUI
- LK — Scripting support

Development dependencies

- wxWidgets — wxWidgets 3.1.0 – GUI framework
- C++ compiler — Visual Studio 2013, GCC 4.8.5 minimum, libc 2.17
- Operating system — Windows XP+, OSX, Linux (Ubuntu 14.04, Mint 18.2)

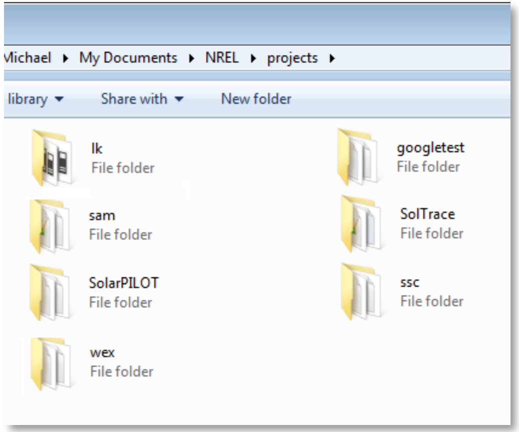
Git repository locations

Repository	URL
wxWidgets	https://www.wxwidgets.org/downloads
WEX	https://github.com/NREL/wex
LK	https://github.com/NREL/lk
SSC	https://github.com/NREL/ssc
SolTrace	https://github.com/NREL/SolTrace
SolarPILOT	https://github.com/NREL/SolarPILOT

If you are new to Git and GitHub, please checkout: https://guides.github.com/

Build instructions

For detailed build instructions
see the SolarPILOT or SolTrace wikis



General quick steps:

- Set up your development tools:
  - Windows: Visual Studio 2017 Community
  - Linux: g++ compiler
- Download the wxWidgets 3.1.1 source code
- Build wxWidgets and set environment variables
- For each dependency, fork and clone the repository into a local project folder, build the project, and then (Windows only) create environment variables
- Build the projects in order
  - SolarPILOT: wxWidgets, LK, WEX, SolTrace/coretrace, SolarPILOT
  - SolTrace: wxWidgets, LK, WEX, SolTrace/app
- This process can take some time, depending on computer speed

Open source projects

Contributing

Contributing to SolarPILOT, SolTrace, or SSC

First Steps

- Read contribution instructions:
  - https://github.com/NREL/solarpilot/blob/develop/CONTRIBUTING.md
  - https://github.com/NREL/soltrace/blob/develop/CONTRIBUTING.md
- Send an email to solarpilot.support@nrel.gov or soltrace.support@nrel.gov agreeing to the contribution policy

Second Steps

- Scope your change and estimate how much time it will take. If the contributions are small (i.e, bug fixes), simply submit changes via pull request.
- If contributions are large, you will need to submit a description of the change for review. If the contribution fits within the project goals, we will work with you to create a plan to get the change incorporated.

Knowing where to contribute

SolarPILOT consists of several code repositories, so you will need to determine where to make your contribution

- If you are making a change to the SolarPILOT's underlying performance algorithms, only the SSC repository is affected
- If you are adding a new feature that changes both calculations and the user interface, then both the SolarPILOT repository and the SSC repository are affected
- If you are adding scripting functionality, the SolarPILOT repository is affected. If modifying script language tools, the LK repository is affected.
- If you need help figuring out where your contribution should go, please let us know

Technical contribution process

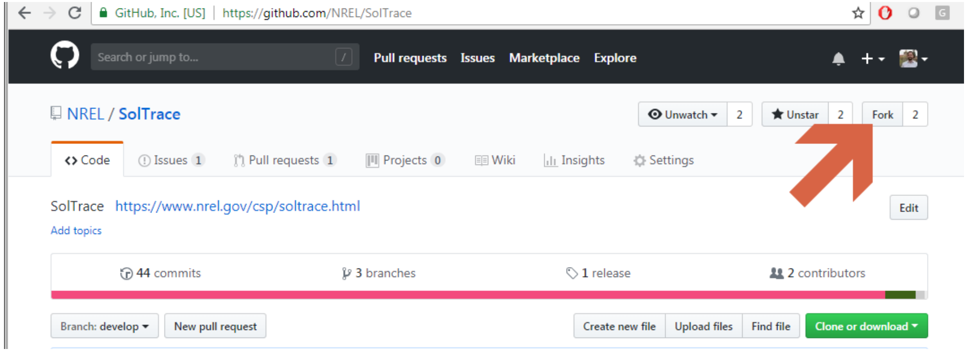
1. Install your favorite Git client application

GitKraken

GitHub Desktop

Git Bash

2. Create a fork of the repo (or all) to which you are contributing



3. Clone your fork(s) and build according to instructions

```
$ git clone https://github.com/NREL/lk
$ git clone https://github.com/NREL/wex
$ git clone https://github.com/NREL/soltrace
$ git clone https://github.com/NREL/ssc
$ git clone https://github.com/NREL/solarpilot
```

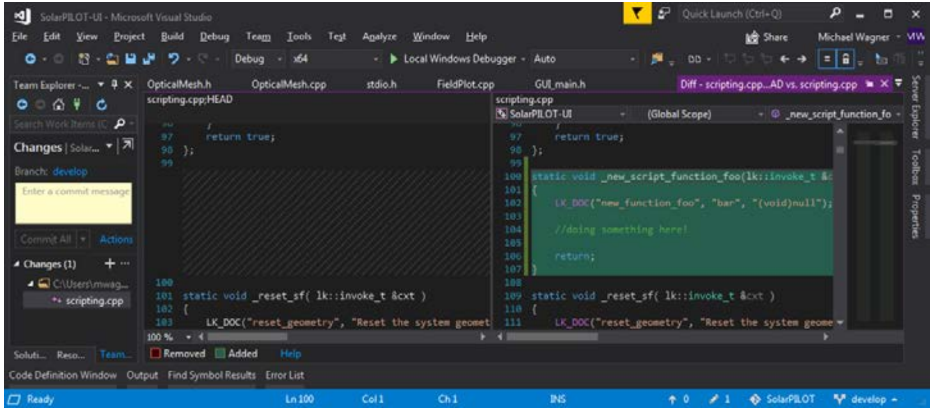
4. Create a branch on the fork off of develop (new feature) or a tagged release (bug fix)

```
$ git checkout v1.
v1.1.0   v1.2.0   v1.2.1
```

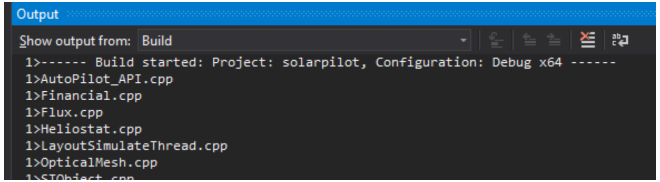
```
$ git checkout -b my-patch-branch
```

Technical contribution process

5. Make your code modifications



6. Build and test affected projects



7. Commit and push changes to your branch

```
$ git commit -a -m "adding bug fix for heliostat aiming vector"
```

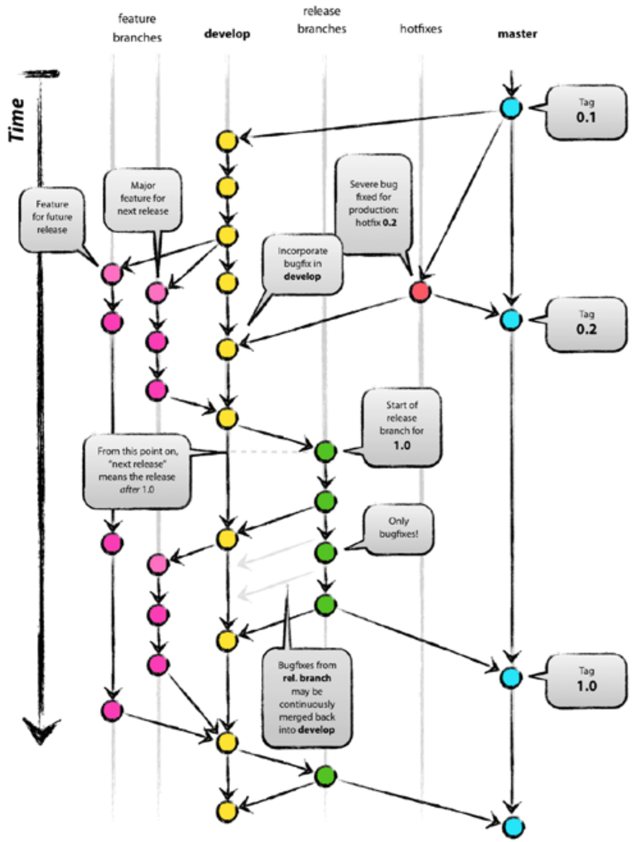
```
$ git push myremote my-patch-branch
```

8. Start a pull request on GitHub. We will review, comment, and merge in official version

More on branches

Branch	Access	Description	Branch from	Merge from	Merge to
master	admin	Current stable version	---	patch, release	---
develop	SolarPILOT-RW	Bleeding edge development version	---	feature, patch, release	---
patch	Contributors	Hotfixes for current major.minor release	Master @tag	---	master, develop, patch
release	SolarPILOT-RW	Next release version	develop	patch	master, develop
feature-<name>	Contributors	All new development	develop, feature-<other>	develop, feature-<other>	develop, feature-<other>

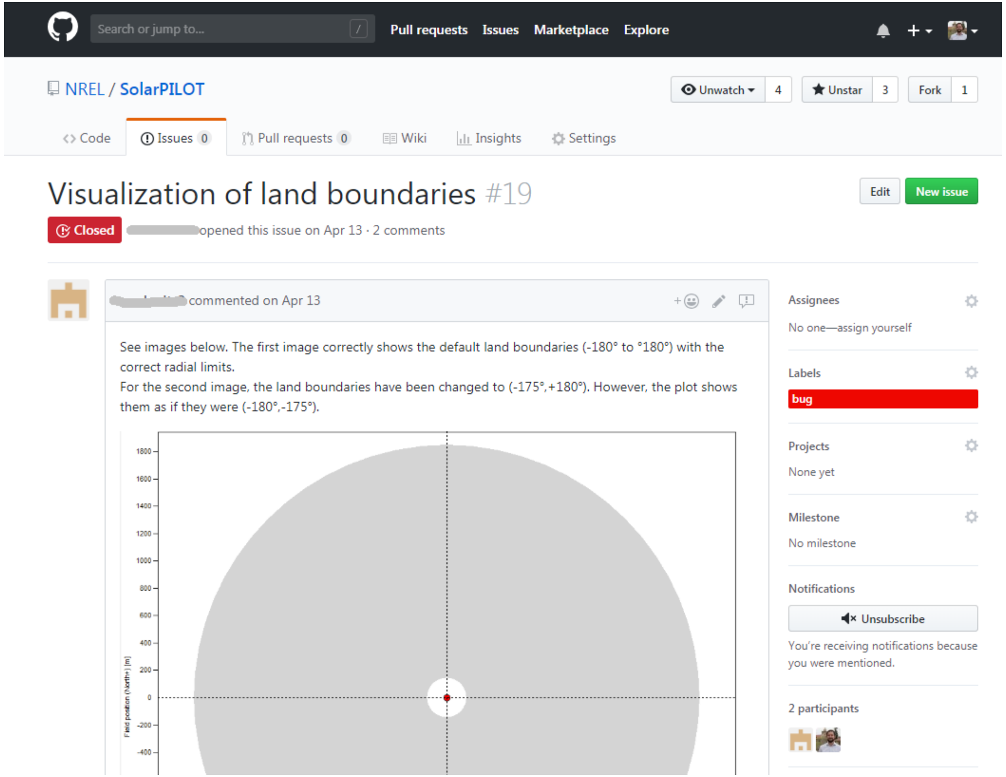
https://github.com/NREL/SolarPILOT/blob/master/CONTRIBUTING.md



https://nvie.com/posts/a-successful-git-branching-model/

Creating GitHub issues

If you discover a bug in the code, want to add a new feature, or have a question, use **GitHub issues** to tell us



Thank you

www.nrel.gov

NREL is a national laboratory of the U.S. Department of Energy, Office of Energy Efficiency and Renewable Energy, operated by the Alliance for Sustainable Energy, LLC.

NREL
NATIONAL RENEWABLE ENERGY LABORATORY