Solar power forecasting, data assimilation, and El Gato

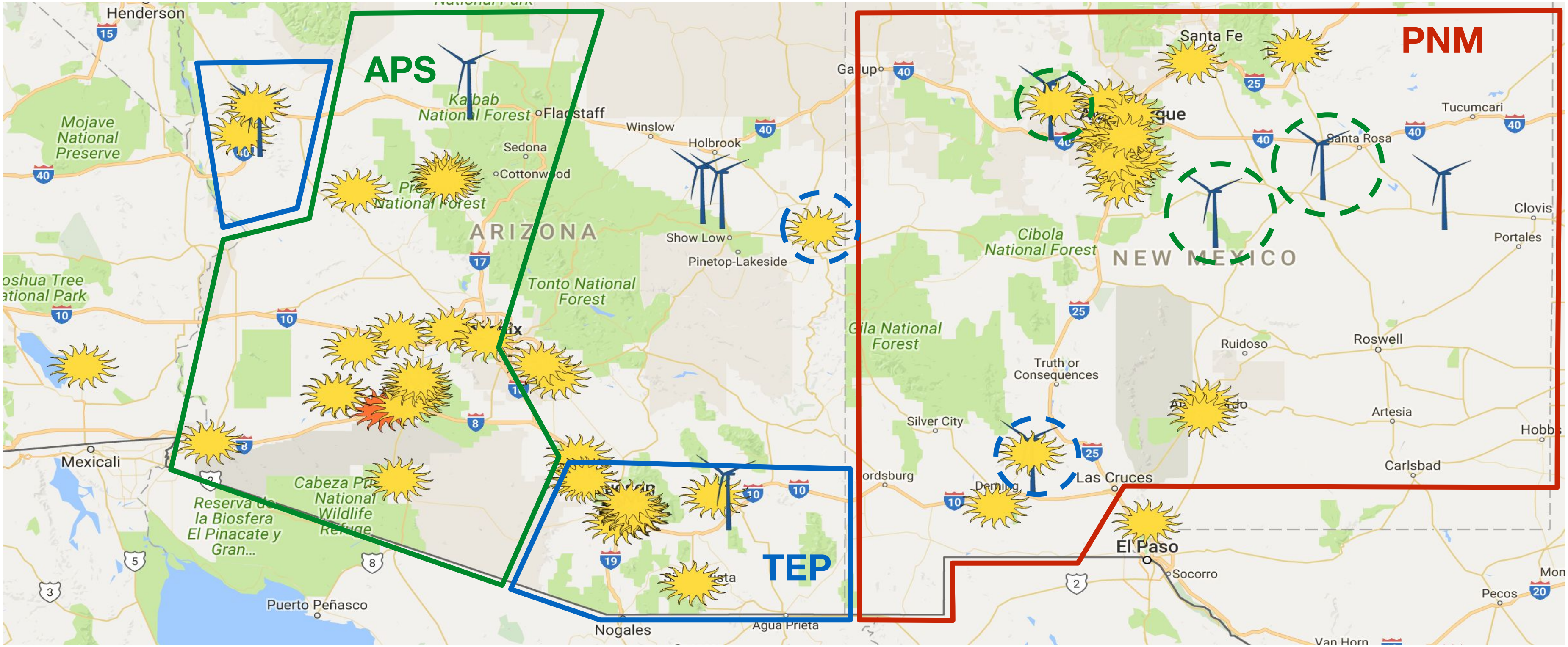
Tony Lorenzo
IES Renewable Power Forecasting Group

THE UNIVERSITY OF ARIZONA

Outline

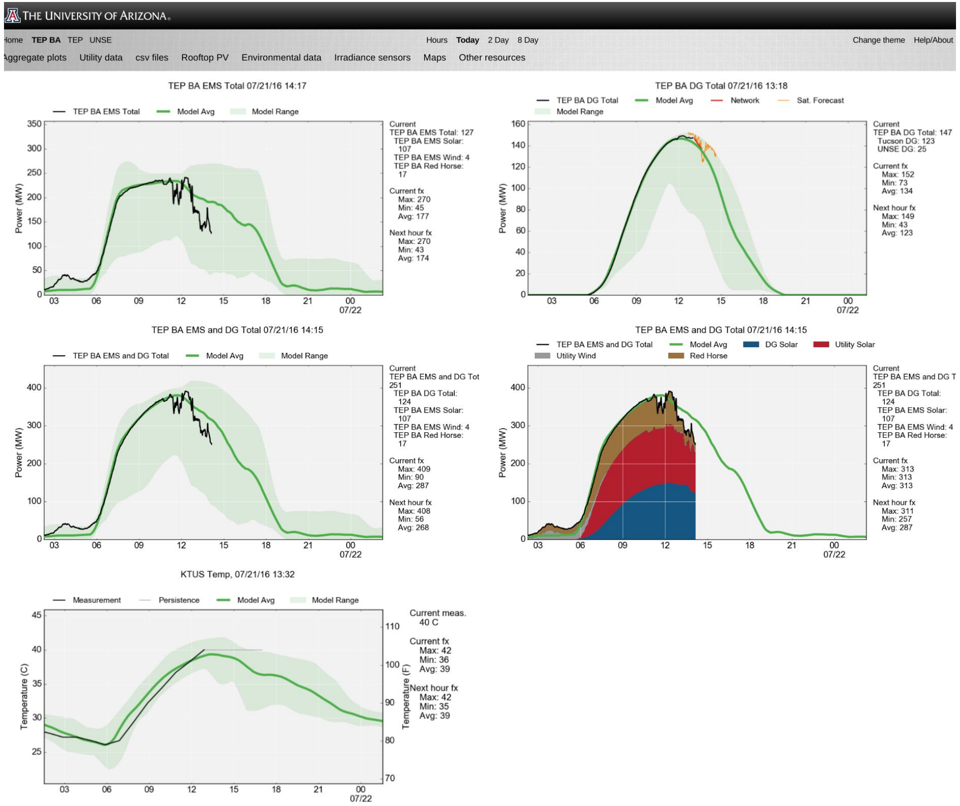
- Motivation & Background
- Solar forecasting techniques
- Satellite data assimilation
- Computational challenges and resources
- Future work

Forecasting Partners



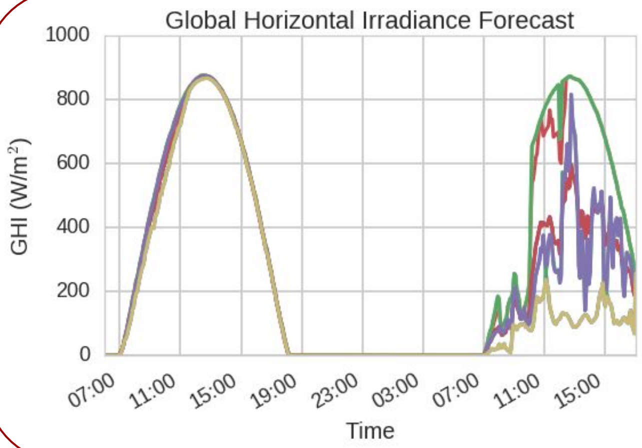
Solar Variability

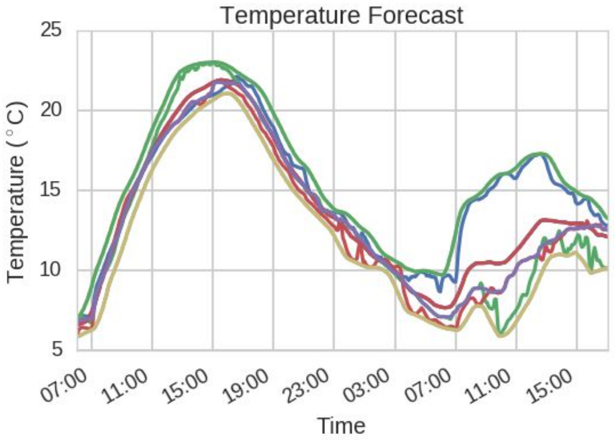
Operational Forecasting for Utilities



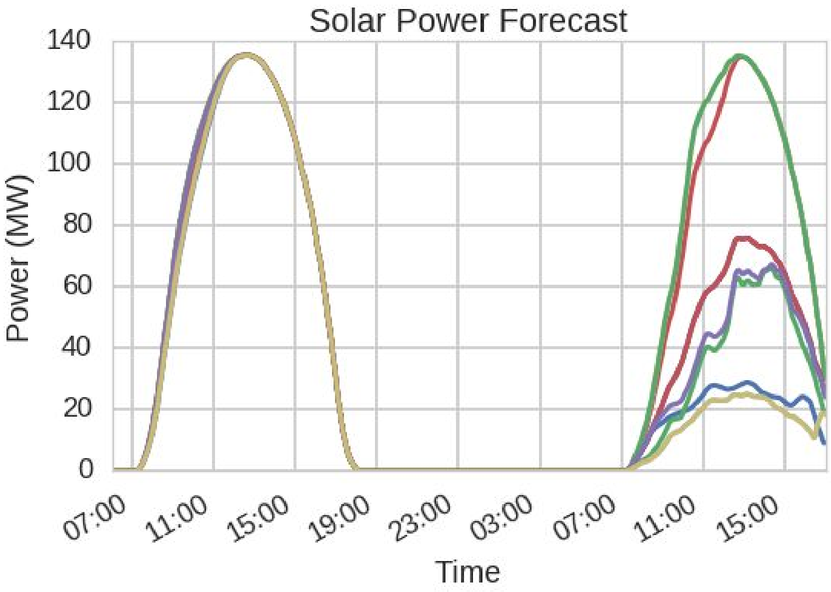
- Final result is a web page with graphics and information meant to help the utilities understand and use the forecasts
- Also have a HTTP API for programmatic access

Irradiance to Power Conversion





PV System Model

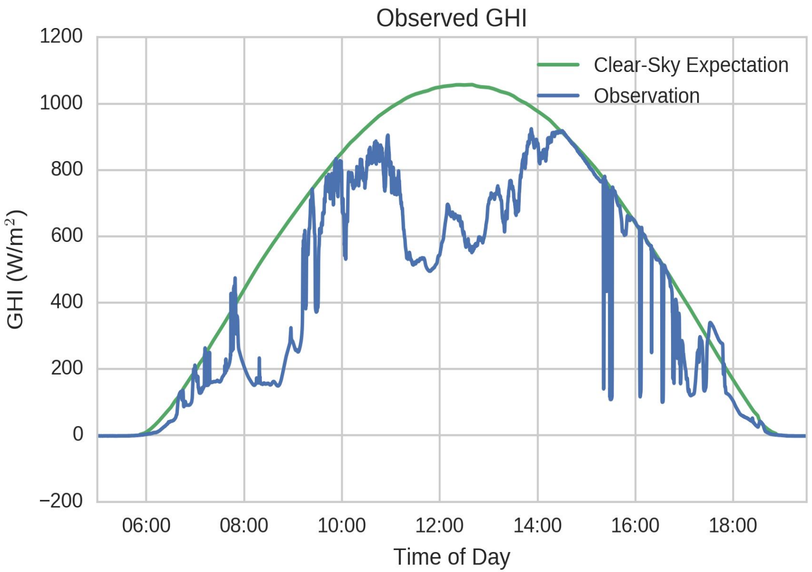


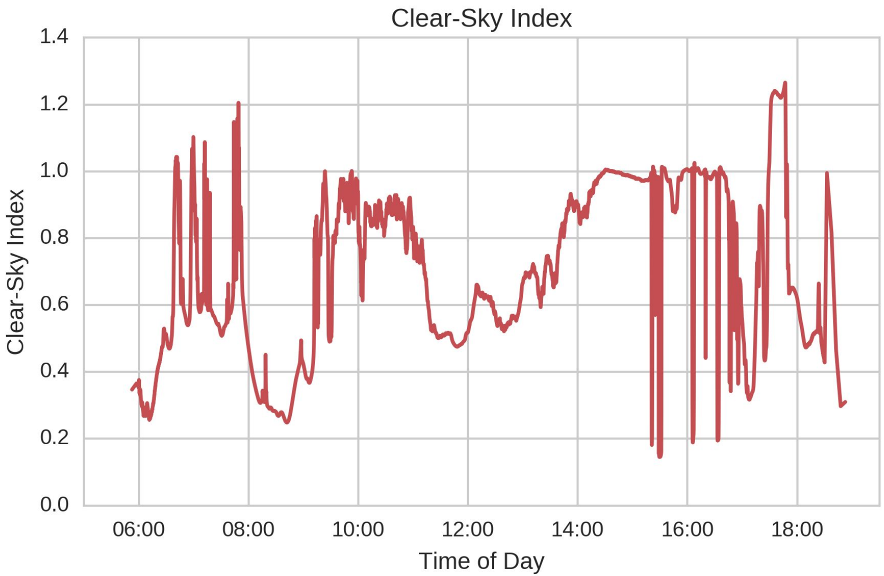
Outline

- Motivation & Background
- Solar forecasting techniques
- Satellite data assimilation
- Computational challenges and resources
- Future work

Clear-Sky Index

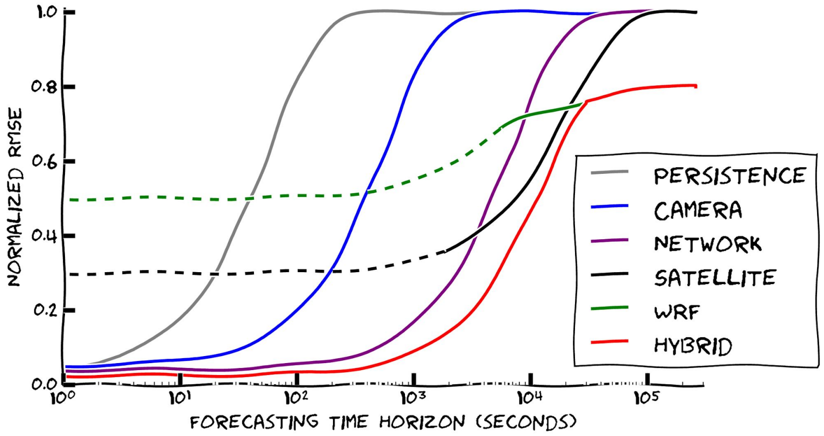
Clear-Sky Index = Observations / Clear-Sky Expectation



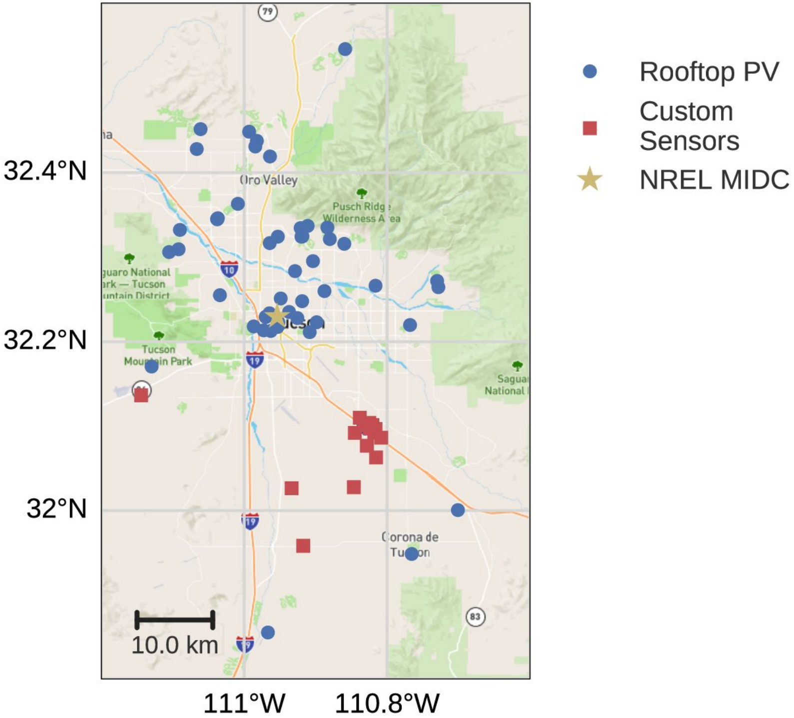


History

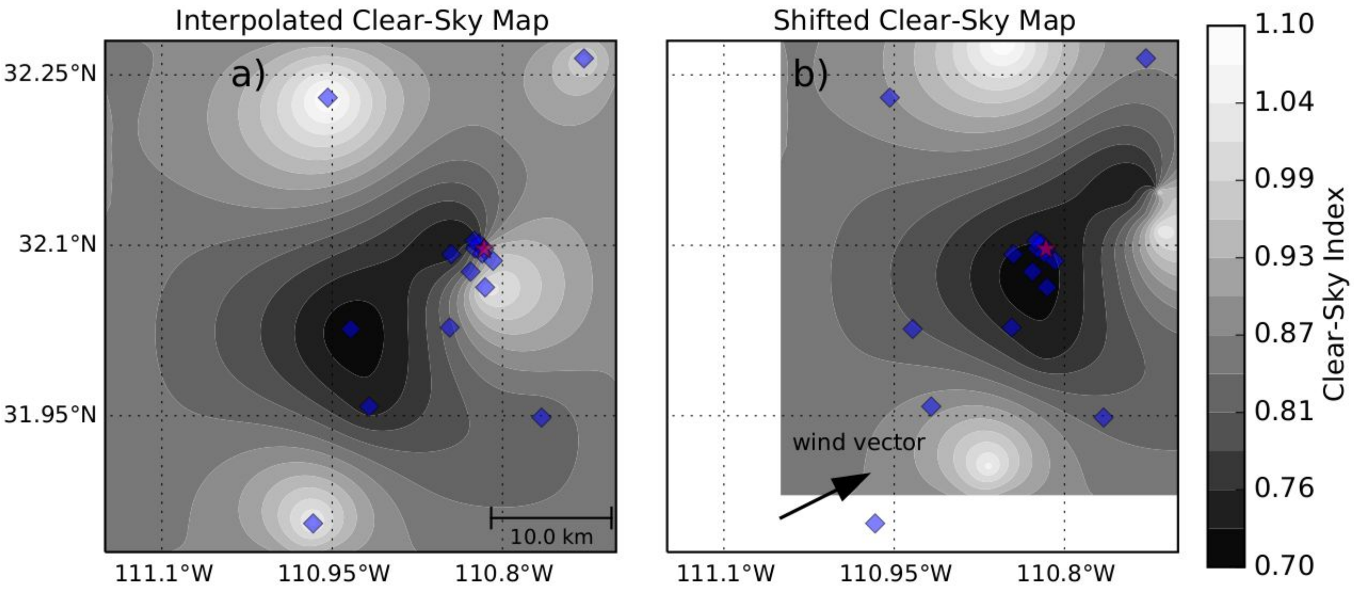
- TEP asked for solar forecasts because they saw variability as an issue
  - Atmospheric Sciences provided WRF forecasts
  - Physics explored cloud camera and sensor network approaches



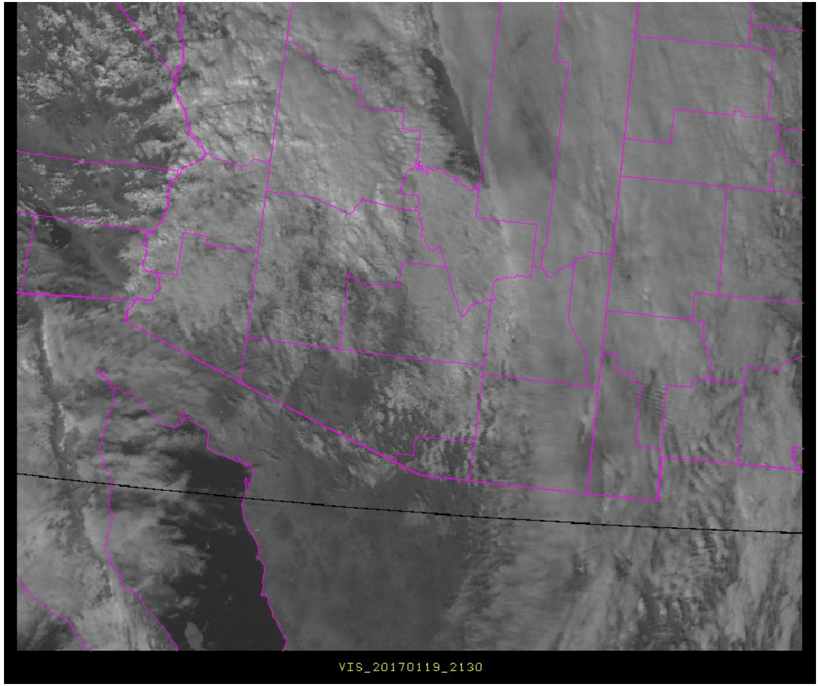
Irradiance Sensor Network

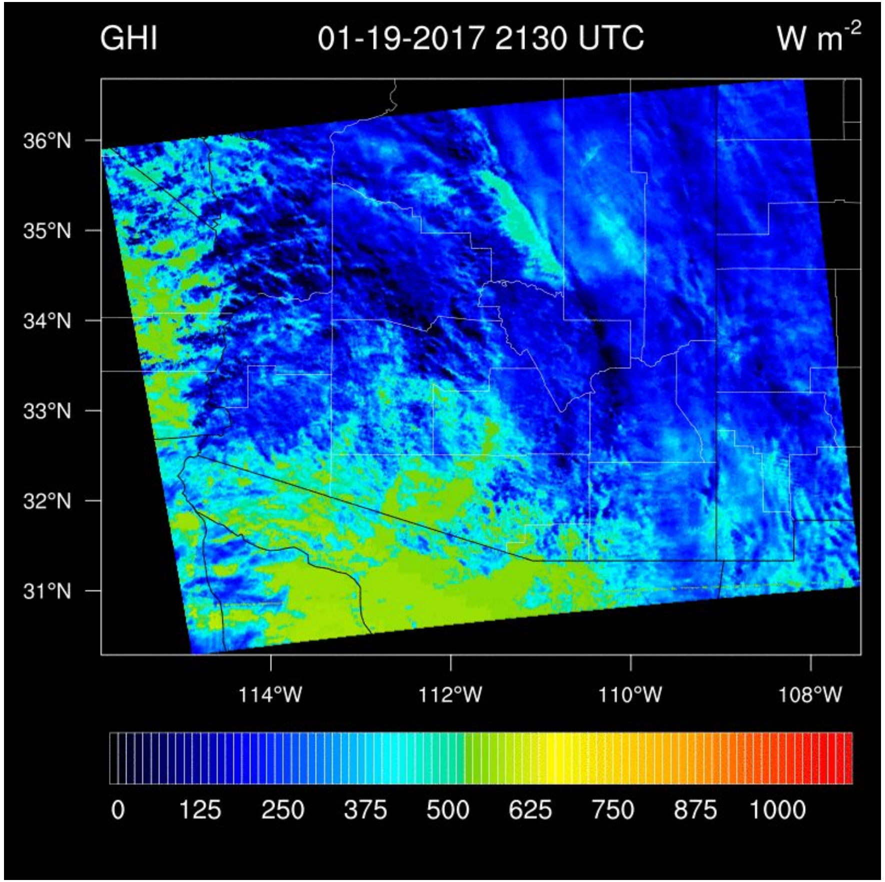


Network Forecasts

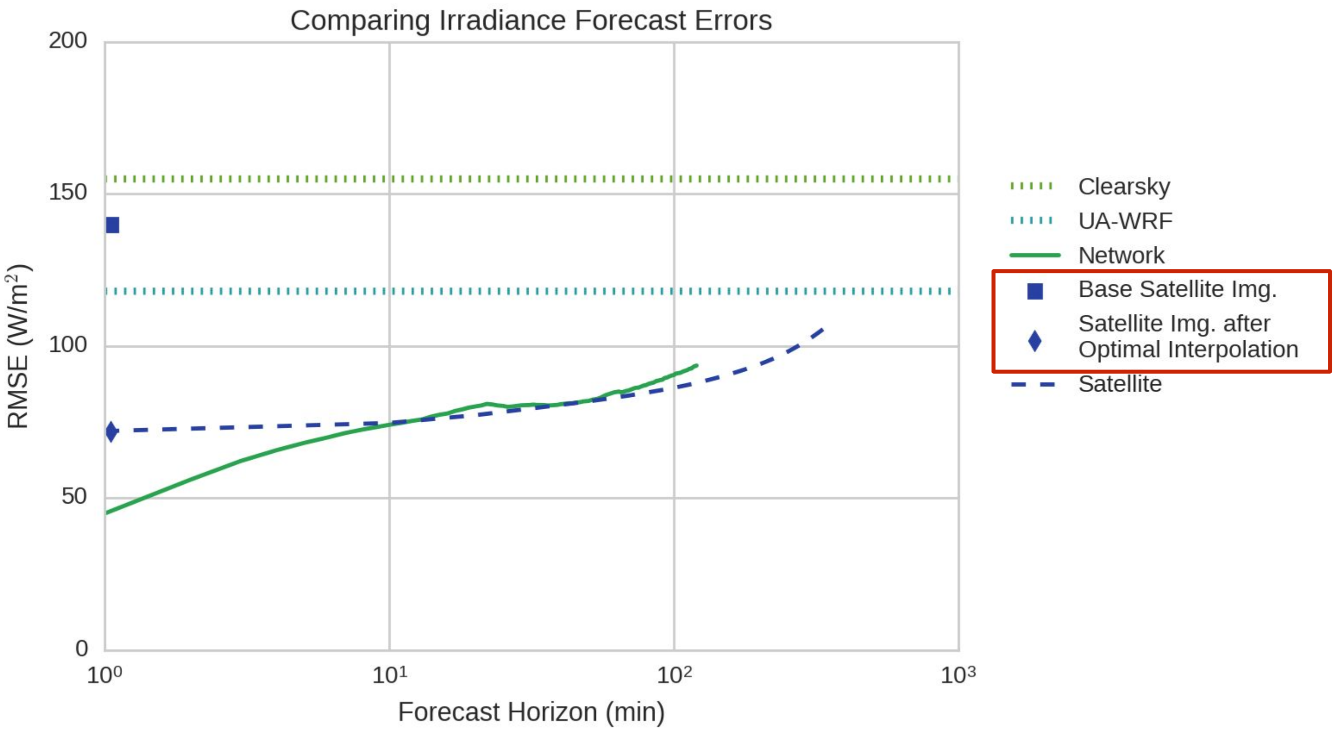


Satellite Derived Irradiance





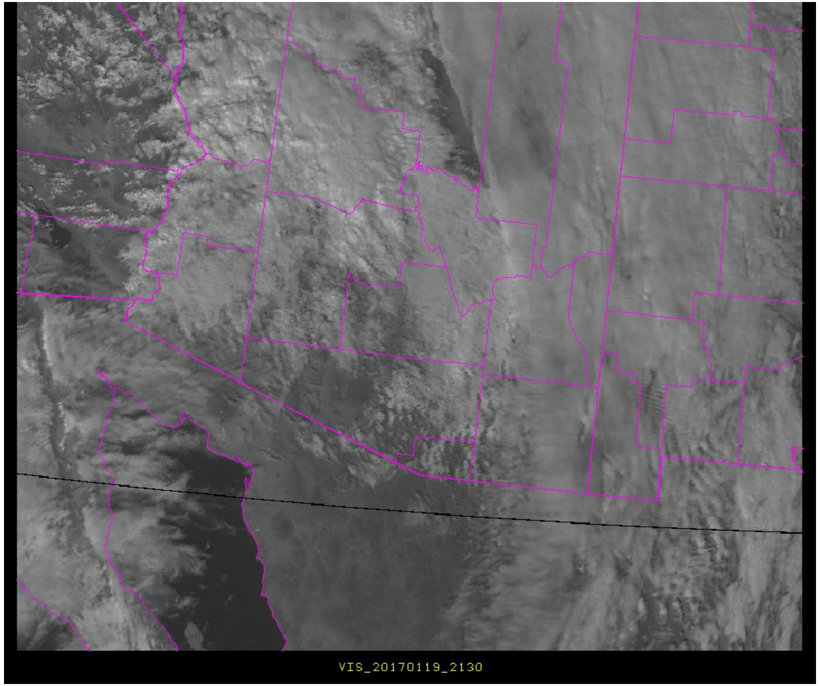
Summary of Results

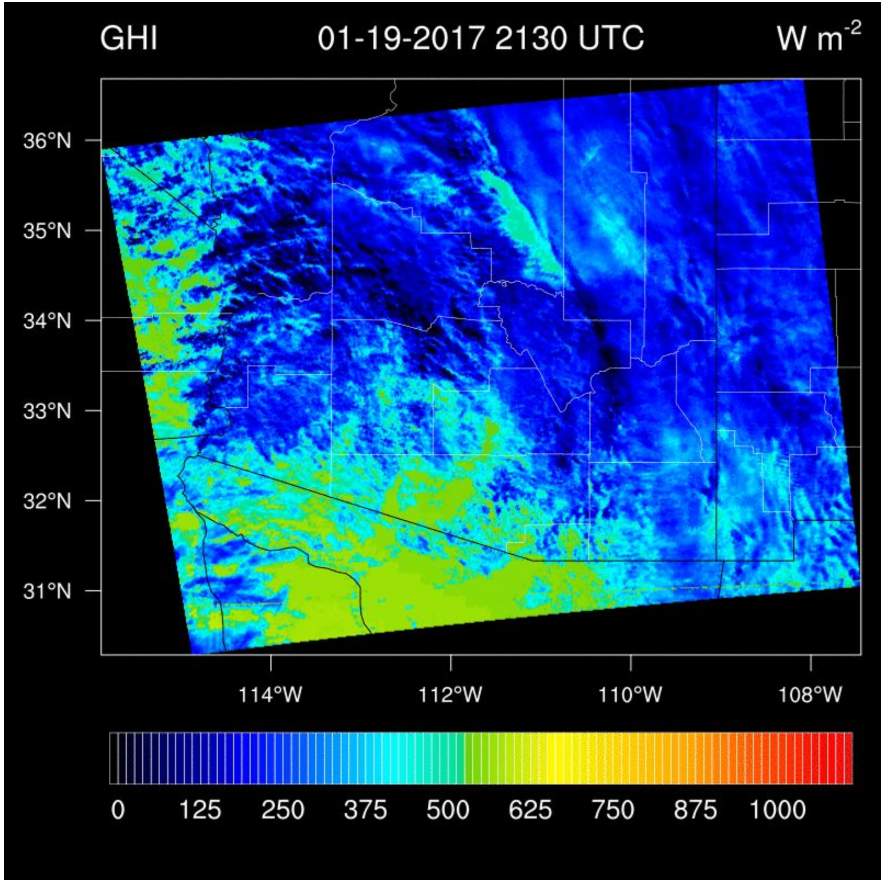


Outline

- Motivation & Background
- Solar forecasting techniques
- Satellite data assimilation
- Computational challenges and resources
- Future work

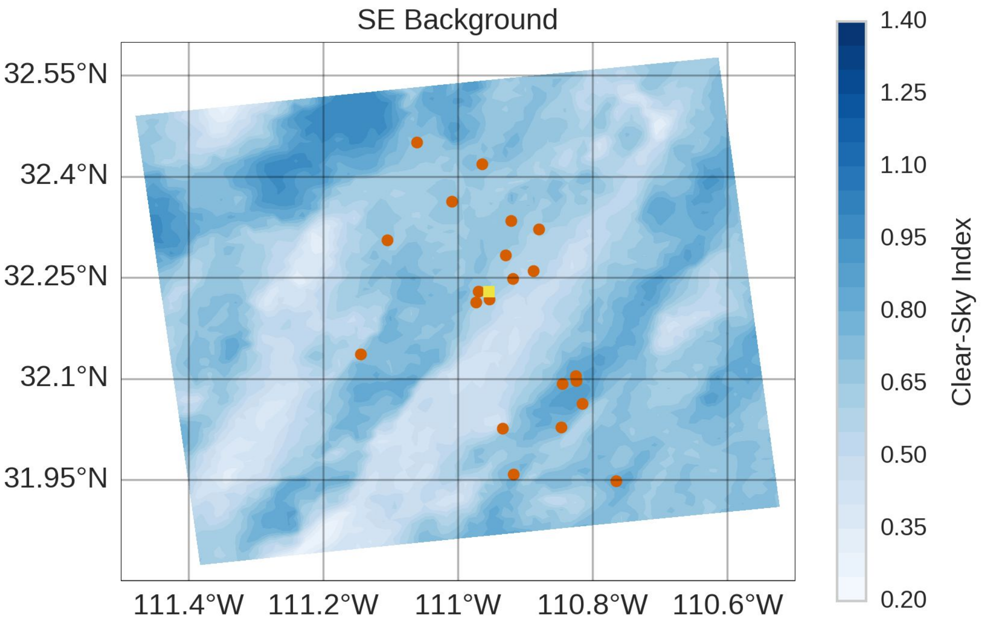
Satellite Derived Irradiance



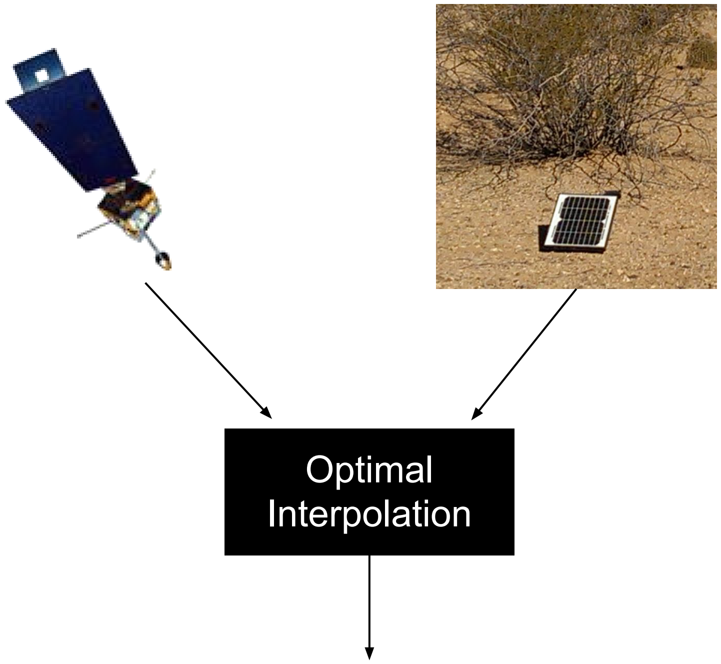


Satellite-derived GHI estimate

- Two conversion models:
  - An semi-empirical (SE) model that applies a regression to data from visible images
  - A physical model that estimates cloud properties and performs radiative transfer (UASIBS)
- Nominally 1 km resolution
- Using 75 km x 82 km area over Tucson



Optimal Interpolation

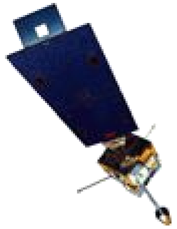


Better satellite-derived estimate of GHI

- Bayesian technique derived by minimizing the mean squared distance between the field and observations
- Is the best linear unbiased estimator of the field
- Same as the update step in the Kalman filter

Satellite image from http://goes.gsfc.nasa.gov/text/goesnew.html

Optimal Interpolation



Optimal Interpolation

Better satellite-derived estimate of GHI

Satellite Derived Irradiance:

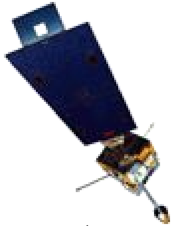
$$\mathbf{x}_b = \mathbf{x}_t + \mathbf{g}$$
$$\mathbf{g} \sim N(\mathbf{0}, \mathbf{P})$$

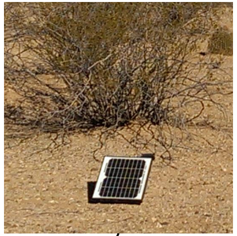
Observations:

$$\mathbf{y} = \mathbf{H}\mathbf{x}_t + \mathbf{e}$$
$$\mathbf{e} \sim N(\mathbf{0}, \mathbf{R})$$

Satellite image from http://goes.gsfc.nasa.gov/text/goesnew.html

OI Algorithm





Maps points from satellite image to observations

Better GHI estimate

$$x_a = x_b + W(y - Hx_b)$$

$$W = PH^T(R + HPH^T)^{-1}$$

Need to a way to estimate these error covariances

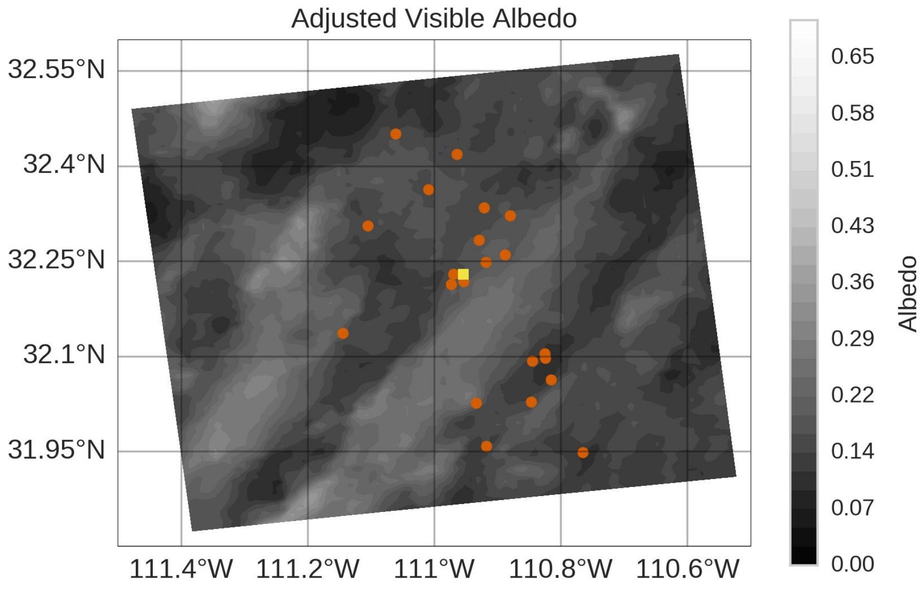
Satellite image from http://goes.gsfc.nasa.gov/text/goesnew.html

Error Covariances: P and R

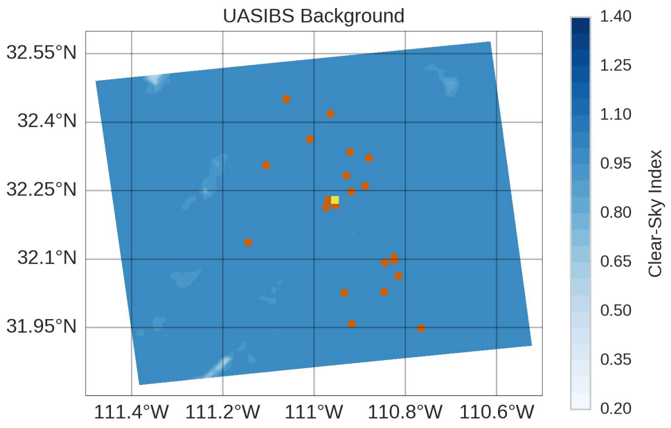
- Decompose **P** into diagonal variance matrix and correlation matrix:

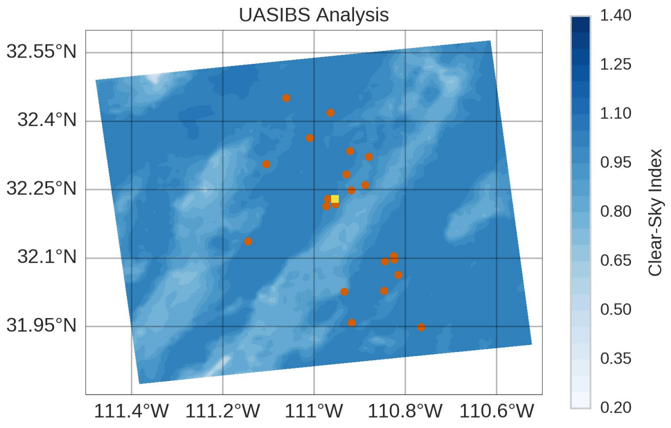
$$\mathbf{P} = \mathbf{D}^{1/2}\,\mathbf{C}\,\mathbf{D}^{1/2}$$

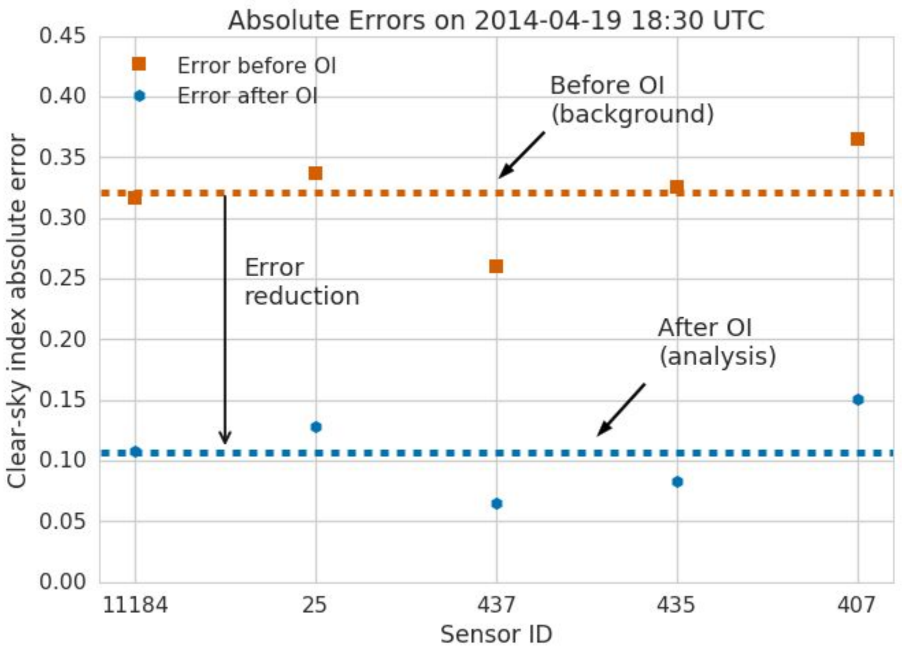
- Prescribe a correlation between image pixels based on the ***difference in cloudiness*** to construct **C**
- Compute **D** from cloud free training images
- Assume observation errors are uncorrelated and estimate **R** from data



Results (one image)

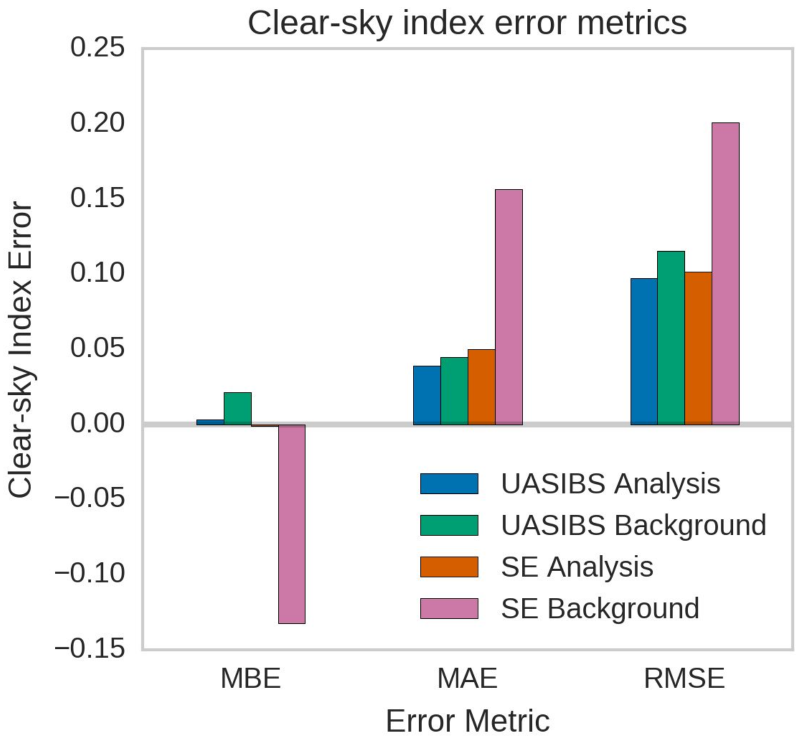


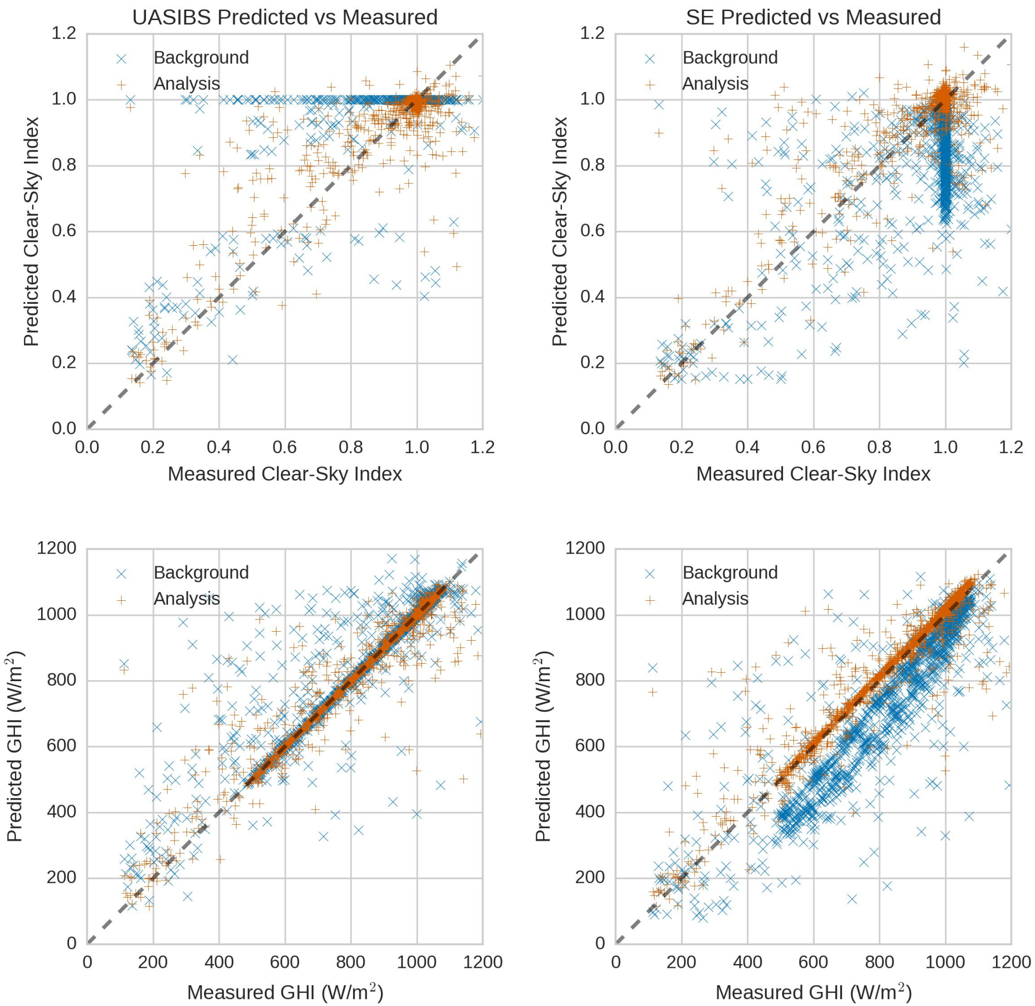




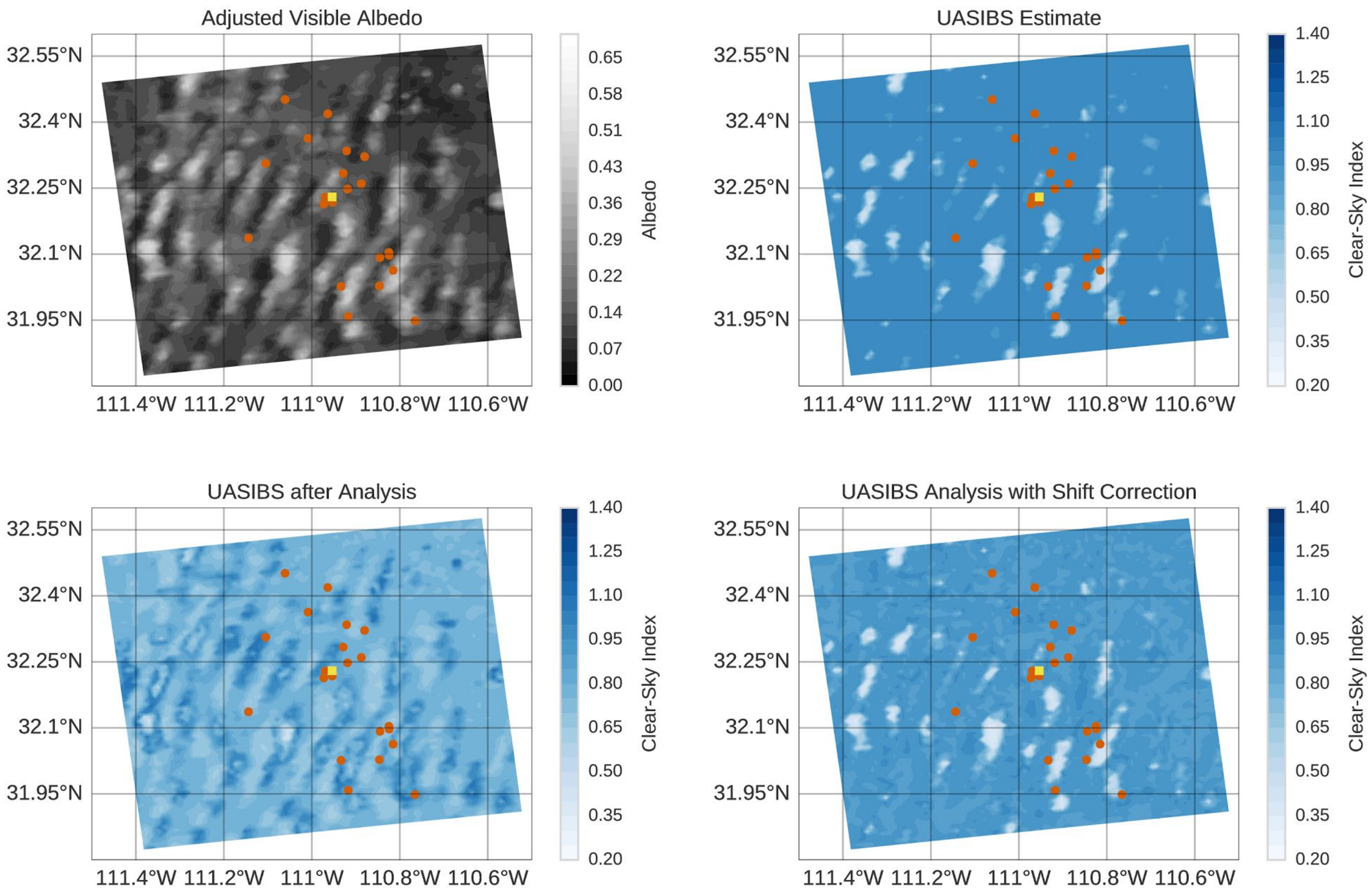
Results

- 900 verification images analyzed
- Six-fold cross-validation over sensors performed
- The large bias for the empirical model was nearly eliminated
- RMSE reduced by 50%

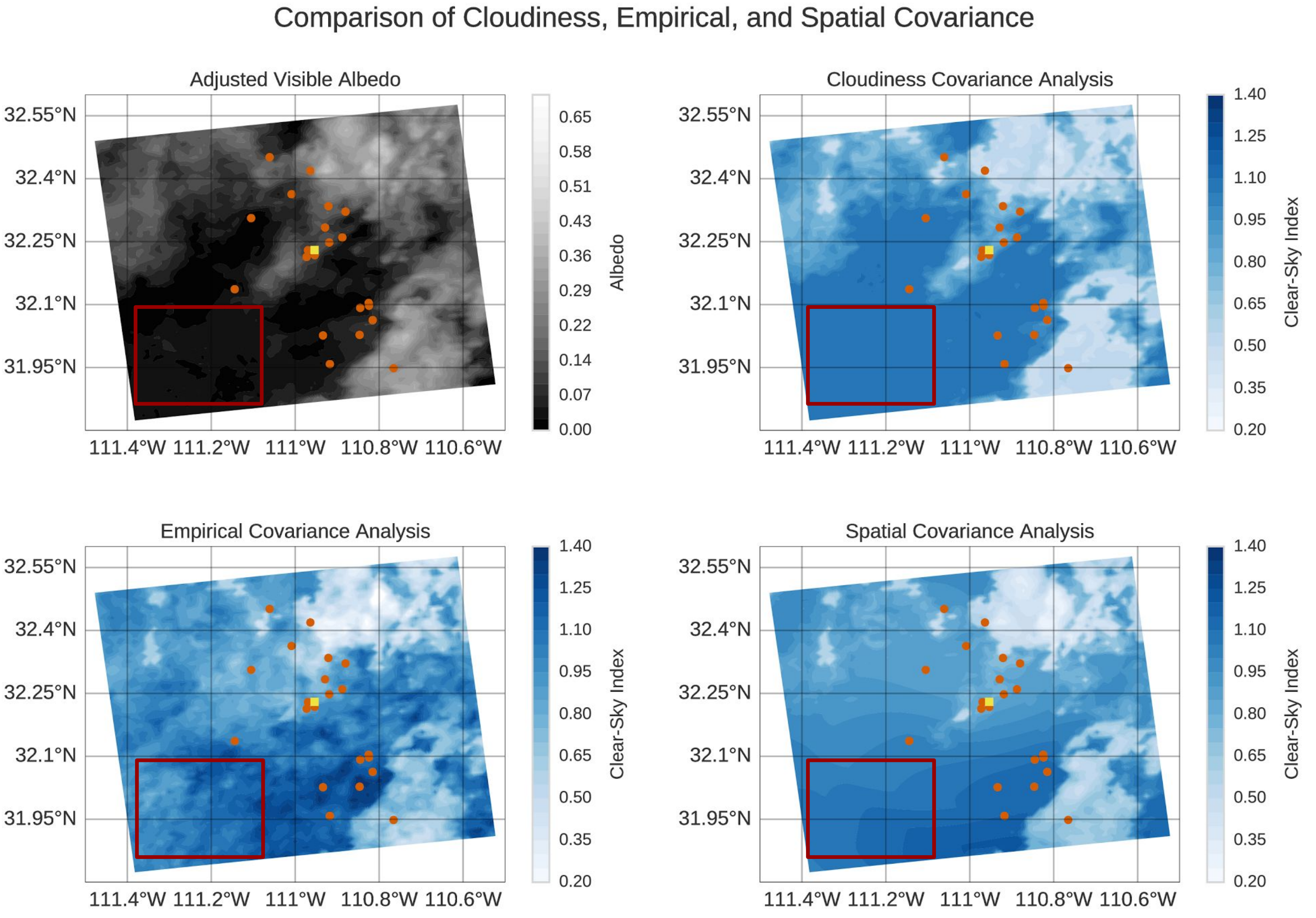




UASIBS Predicted vs Measured
SE Predicted vs Measured
Background
Analysis
Predicted Clear-Sky Index
Measured Clear-Sky Index
Predicted GHI (W/m²)
Measured GHI (W/m²)



Adjusted Visible Albedo
UASIBS Estimate
UASIBS after Analysis
UASIBS Analysis with Shift Correction
32.55°N
32.4°N
32.25°N
32.1°N
31.95°N
111.4°W 111.2°W 111°W 110.8°W 110.6°W
Albedo
0.65 0.58 0.51 0.43 0.36 0.29 0.22 0.14 0.07 0.00
Clear-Sky Index
1.40 1.25 1.10 0.95 0.80 0.65 0.50 0.35 0.20



Comparison of Cloudiness, Empirical, and Spatial Covariance
Adjusted Visible Albedo
Cloudiness Covariance Analysis
Empirical Covariance Analysis
Spatial Covariance Analysis
32.55°N
32.4°N
32.25°N
32.1°N
31.95°N
111.4°W 111.2°W 111°W 110.8°W 110.6°W
Albedo
0.65
0.58
0.51
0.43
0.36
0.29
0.22
0.14
0.07
0.00
Clear-Sky Index
1.40
1.25
1.10
0.95
0.80
0.65
0.50
0.35
0.20

OI Parameters

$$\mathbf{P} = \mathbf{D}^{1/2}\mathbf{C}\mathbf{D}^{1/2}$$

$$\mathbf{D} = d\mathbf{D}'$$

$$C_{ij} = k(r_{ij})$$

Correlation Functions

$$k(r) = \begin{cases} 1 - \frac{r}{l} & r < l \\ 0 & r \geq l \end{cases}$$

$$k(r) = \exp\left(-\frac{r}{l}\right)$$

$$k(r) = \exp\left(-\frac{r^2}{l^2}\right)$$

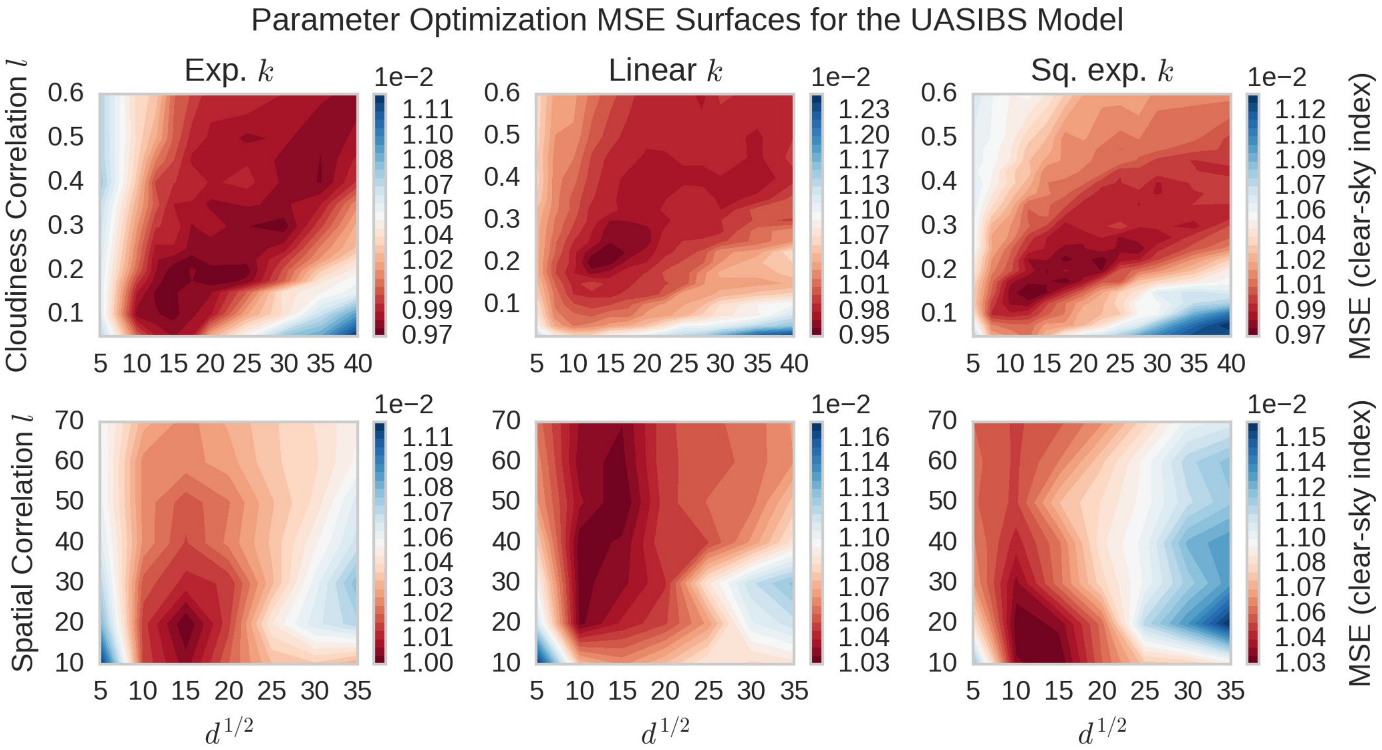
Distance Metrics

$$r_{ij} = |z_i - z_j|$$

$$r_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

Need to tune $d, \; k, \; l, \; r$

Parameter Optimization



Outline

- Motivation & Background
- Solar forecasting techniques
- Satellite data assimilation
- Computational challenges and resources
- Future work

Parameter Optimization

- Satellite to irradiance model
  - UASIBS
  - Semi-empirical
- Correlation method
  - Cloudiness
  - Spatial
- Correlation function
  - Linear
  - Exponential
  - Squared Exponential
- Correlation length
- **P** error inflation
- Cloud height adjustment

500 training images * 2 models * 6 fold cross validation * 50 height adj. * 2 corr. methods * 3 corr. fcns. * ~10 corr. lengths * ~10 inflation params = 200 million OI analyses

1 year on a 4 core laptop!

7 weeks on a 24 core server

<1 week using GPUs on El Gato

Translating code for the GPU

```python
import numpy as np
from scipy import linalg

def compute_analysis_cpu(xb, y, R, P, H):
    HT = np.transpose(H)
    hph = np.dot(H, np.dot(P, HT))
    inv = linalg.inv(R + hph)
    W = np.dot(P, np.dot(HT, inv))
    xa = xb + np.dot(W, y - np.dot(H, xb))
    return xa

xa = compute_analysis_cpu(xb, y, R, P, H)
```

```python
import skcuda.linalg as cu
from pycuda import gpuarray

def compute_analysis_cuda(xb, y, R, P, H):
    HT = cu.transpose(H)
    hph = cu.dot(H, cu.dot(P, HT))
    inv = cu.inv(R + hph)
    W = cu.dot(P, cu.dot(HT, inv))
    xa = xb + cu.dot(W, y - cu.dot(H, xb))
    return xa

xb_gpu = gpuarray.to_gpu(xb)
...
xa_gpu = compute_analysis_cuda(
    xb_gpu, y_gpu, R_gpu, P_gpu, H_gpu)
xa = xa_gpu.get()
```

UA HPC Resources

- **Free** allocations for research groups
- HPC consultants ready to help

El Gato

- 136 nodes
- 140 NVIDIA Tesla K20x GPUs
- 20 Intel Phi coprocessors

Ocelote

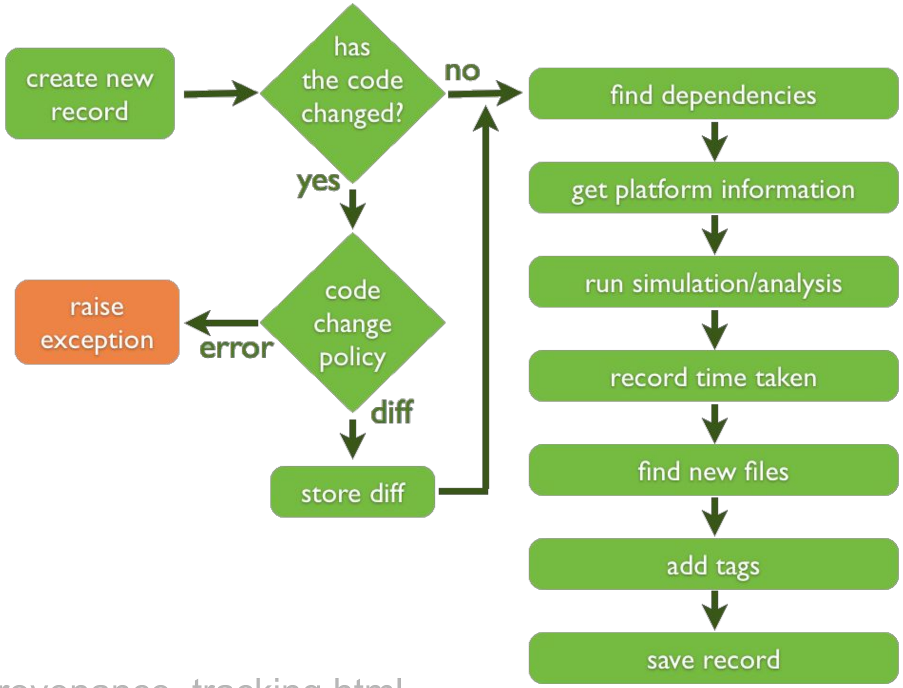
- 336 nodes
- 15 NVIDIA Tesla K80 GPUs
- 10044 cores

Other Resources

- Dask: parallel computing library
- Numba: JIT for high performance Python
- Singularity: containers on HPC
- PyCUDA: pythonic access to CUDA
- scikit-cuda: CUDA scientific library wrapper (cuBLAS)
- Sumatra: automated provenance tracking

Sumatra Provenance Tracking:
Computational Lab Notebook

- No more resultsV1, results_best_maybe?
- Keeps track of:
  - Simulation parameters
  - Input files
  - Output files
  - Code version
  - Start/end time
  - Custom tags & comments

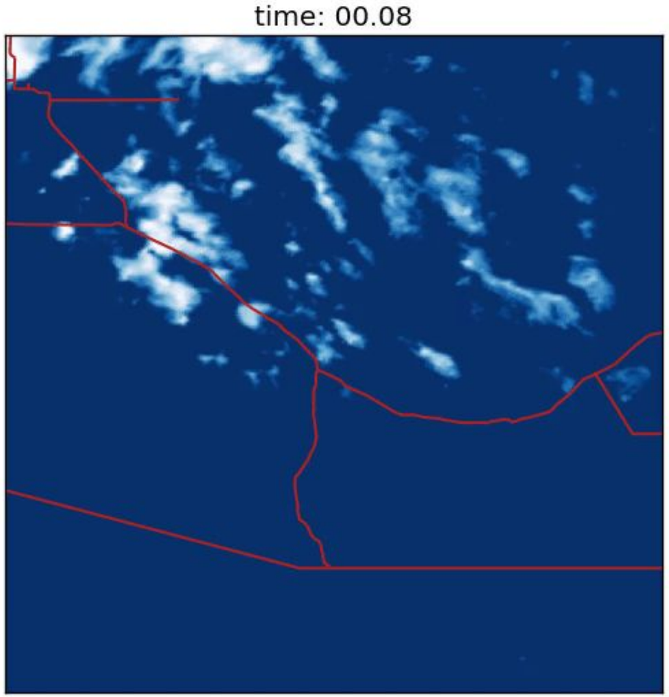


More info at http://rrcns.readthedocs.io/en/latest/provenance_tracking.html

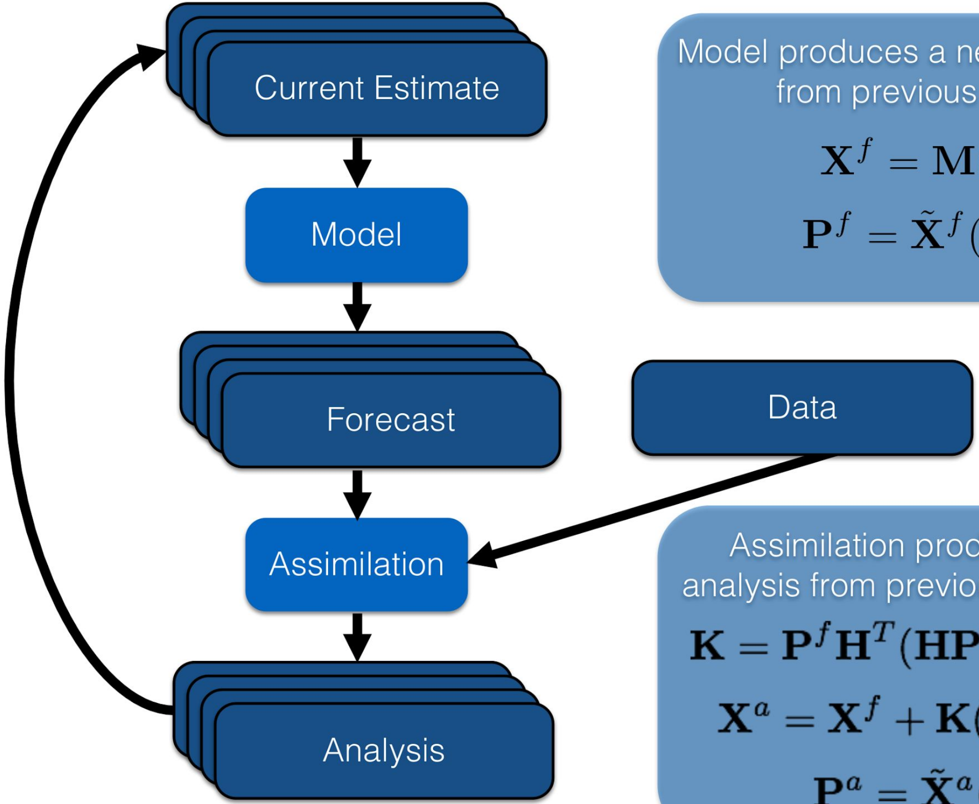
Outline

- Motivation & Background
- Solar forecasting techniques
- Satellite data assimilation
- Computational challenges and resources
- Future work

Cloud Advection



Ensemble Kalman Filter



Model produces a new background from previous analysis

$$\mathbf{X}^f = \mathbf{M}(\mathbf{X}^a)$$

$$\mathbf{P}^f = \tilde{\mathbf{X}}^f(\tilde{\mathbf{X}}^f)^T$$

Assimilation produces a new analysis from previous background

$$\mathbf{K} = \mathbf{P}^f\mathbf{H}^T(\mathbf{H}\mathbf{P}^f\mathbf{H}^T + \mathbf{R})^{-1}$$

$$\mathbf{X}^a = \mathbf{X}^f + \mathbf{K}(\mathbf{Y} - \mathbf{H}\mathbf{X}^f)$$

$$\mathbf{P}^a = \tilde{\mathbf{X}}^a(\tilde{\mathbf{X}}^a)^T$$

Thank you!

RESEARCH, DISCOVERY & INNOVATION
Institute for Energy Solutions