SAND2018-5144C

DAKOTA

Explore and predict with confidence

CCR

Center for Computing Research

Dakota Optimization and UQ

Explore and Predict with Confidence

Brian M. Adams, J. Adam Stephens

Optimization and Uncertainty Quantification

Patricia D. Hough

Quantitative Modeling and Analysis

http://dakota.sandia.gov

2018 SIAM Conference on Uncertainty Quantification
April 16—19, 2018
Garden Grove, CA

U.S. DEPARTMENT OF ENERGY

NNSA National Nuclear Security Administration

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

SNL Mission: Advanced Science and Engineering for National Security

- Nuclear Weapons
- Defense Systems and Assessments
- Energy and Climate
- Global Security

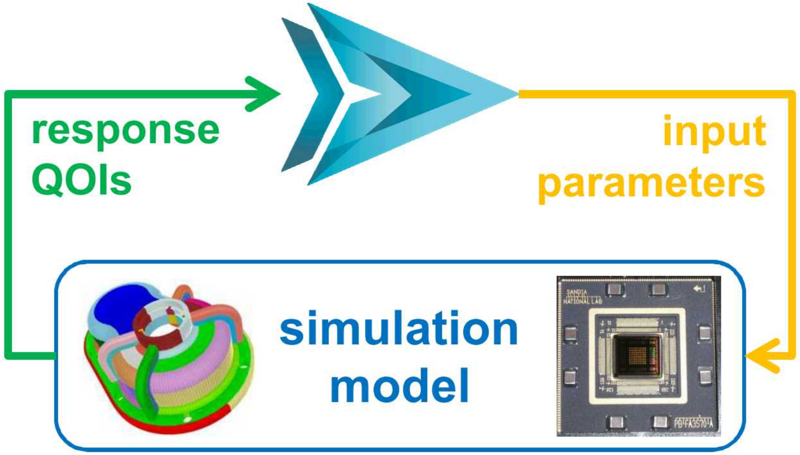
Strong research foundations span many disciplines

Dakota Mission:
To serve Sandia's mission through state-of-the-art research and robust, usable software for optimization and uncertainty quantification.

Dakota Team: has balanced strengths in algorithm research, software design and development, and application deployment and support

Dakota: Algorithms for Design Exploration and Uncertainty Quantification

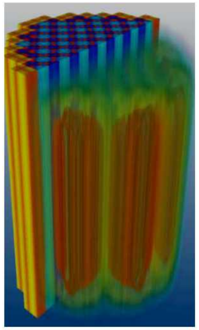
- Suite of iterative mathematical and statistical methods that interface to computational models
- Makes sophisticated parametric exploration of black-box simulations practical for a computational design-analyze-test cycle:
  - Sensitivity Analysis
  - Uncertainty Quantification
  - Design Optimization
  - Model Calibration



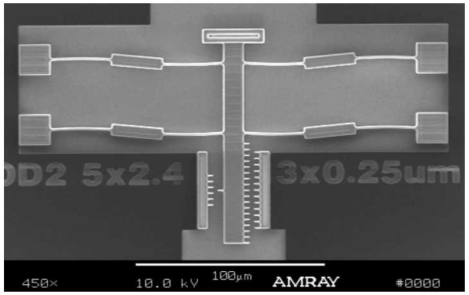
- *Goal: provide scientists and engineers (analysts, designers, decision makers) richer perspective on model predictions*

Practical Design Exploration and UQ for Diverse Simulations

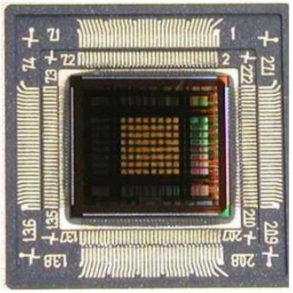
- Applied to many science and engineering domains
  mechanics, structures, shock, fluids, electrical, radiation, bio, chemistry, climate, infrastructure, etc.
- Diverse, often costly, simulation codes:
  finite element, coupled multi-physics, discrete event, Matlab, Python models



UQ and Bayesian inference for nuclear reactor core analysis



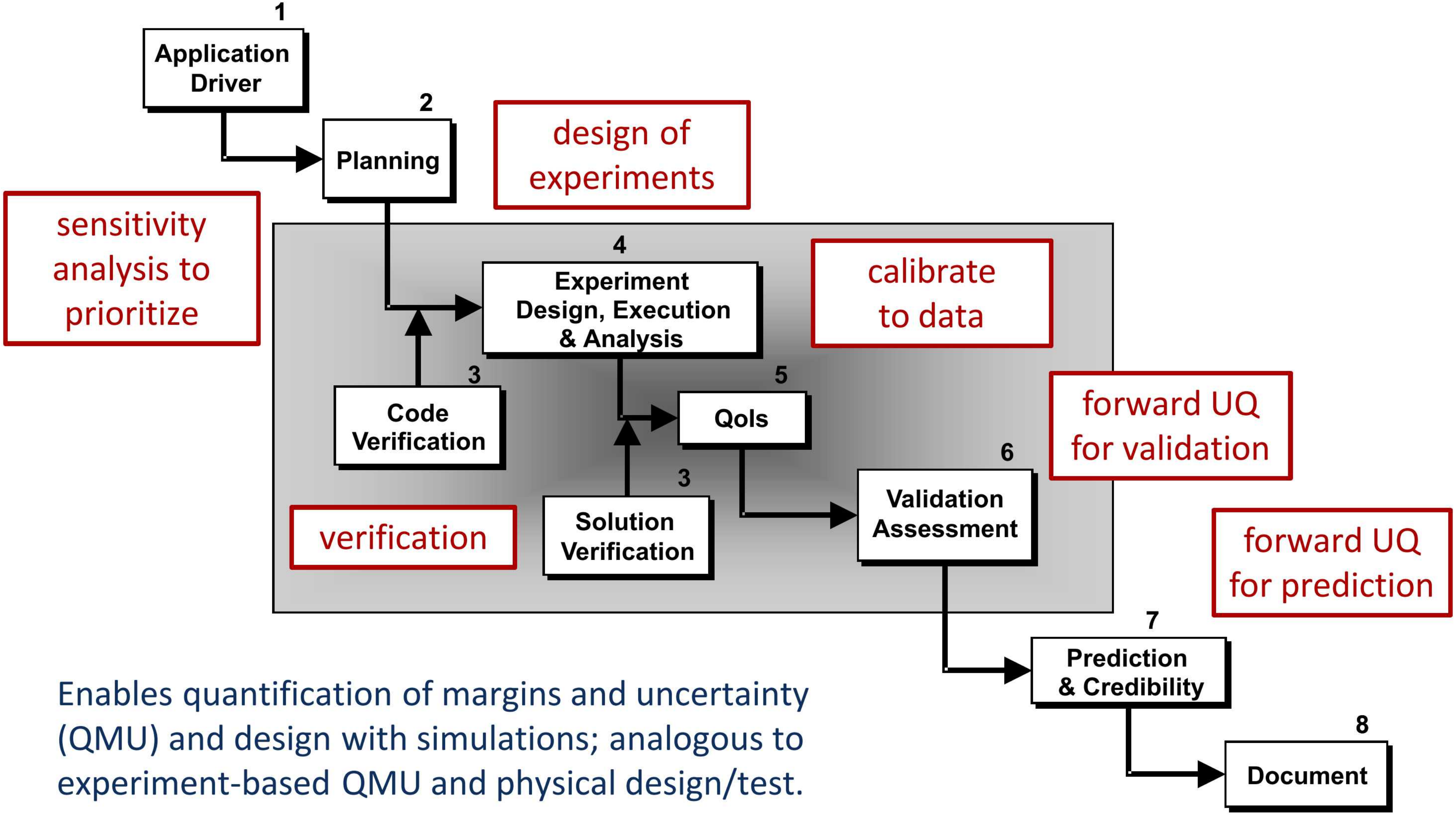
Reliability-based design optimization of micro-electro-mechanical systems (MEMS)



UQ for qualification of electrical circuits in harsh environments

Adjoint-based UQ for robust design of satellite radiation shields

Supports Credible Prediction



Enables quantification of margins and uncertainty (QMU) and design with simulations; analogous to experiment-based QMU and physical design/test.

Many Methods in One Tool

Sensitivity Analysis • Designs: param. sweeps, MC/LHS, DACE, sparse grid, one-at-a-time • Analysis: correlations, scatter, Morris effects, Sobol indices	Uncertainty Quantification • MC/LHS/Adaptive sampling • Local/global reliability • Stochastic expansions • Epistemic and interval methods • Multi-fidelity/multi-level
Optimization • Gradient-based local • Derivative-free local • Global/heuristics • Surrogate-based, multi-fidelity	Calibration • Tailored gradient-based • Use any optimizer • Bayesian inference

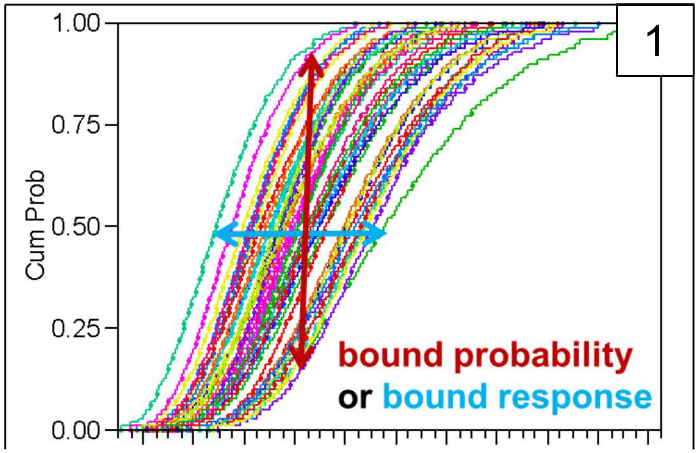
✓ One flexible simulation interface, many methods: once interface created, apply appropriate algorithm depending on question at hand

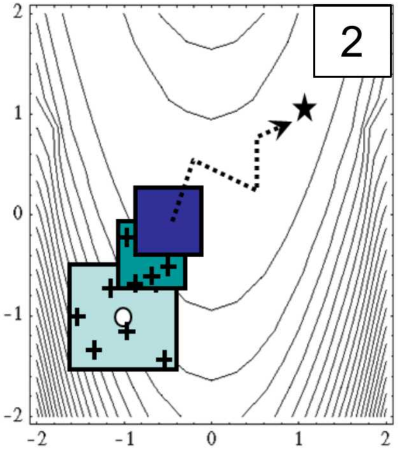
✓ Scalable parallel computing from desktop to HPC

Engineering Needs Drive Dakota R&D

Advanced approaches help you solve practical problems:

- Characterize parameter uncertainty → Bayesian calibration
- Hybrid analysis goals → mix methods, surrogates, and models
- Mixed uncertainty characterizations → epistemic and mixed UQ approaches [1]
- Costly or noisy simulations → surrogate-based optimization [2] and UQ
- Build in safety or robustness → mixed deterministic/probabilistic methods [3]





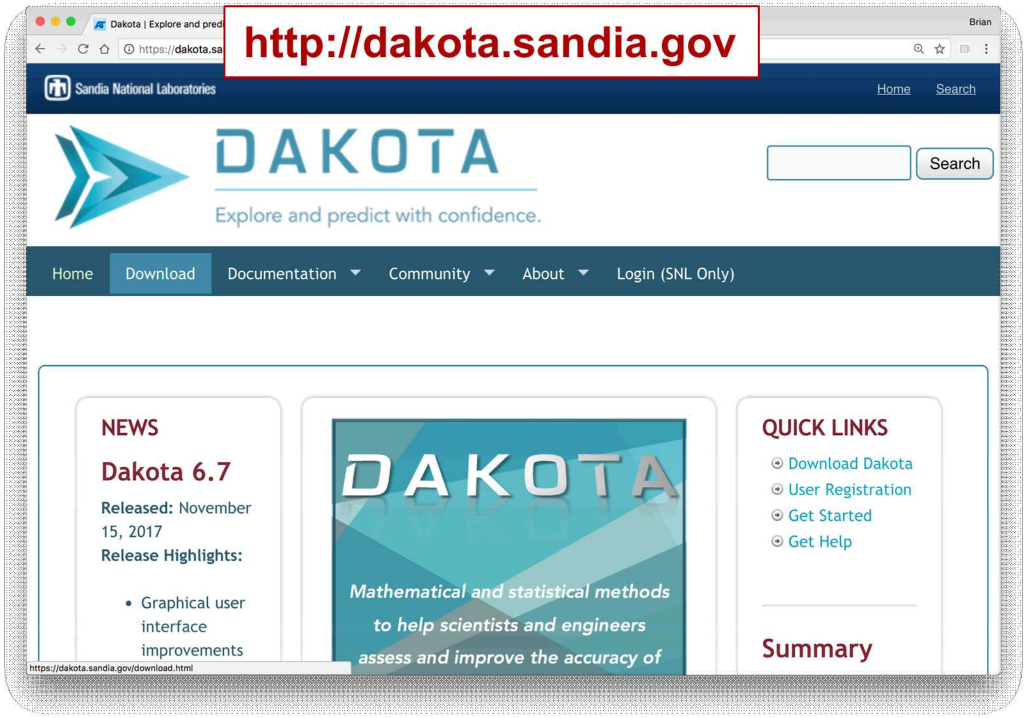
3

$$\begin{aligned} \text{min} \quad & f(d) + W\, s_u(d) \\ \text{s.t.} \quad & g_l \leq g(d) \leq g_u \\ & h(d) = h_t \\ & d_l \leq d \leq d_u \\ & a_l \leq A_i\, s_u(d) \leq a_u \\ & A_e\, s_u(d) = a_t \end{aligned}$$

Dakota History and Resources

- Genesis: 1994 optimization LDRD
- Modern software quality and development practices
- Released every May 15 and Nov 15
- Established user support process and mailing list

Mike Eldred, Founder



http://dakota.sandia.gov

NEWS

Dakota 6.7

Released: November 15, 2017

Release Highlights:

- Graphical user interface improvements

DAKOTA

Mathematical and statistical methods to help scientists and engineers assess and improve the accuracy of

QUICK LINKS

- Download Dakota
- User Registration
- Get Started
- Get Help

Summary

https://dakota.sandia.gov/download.html

Lab mission-driven algorithm R&D deployed in production software

- Extensive website: documentation, training slides/videos, downloads
- Open source to facilitate collaboration; widely downloaded

Dakota Capability Directions

DAKOTA

Explore and predict with confidence

2016-2020 STRATEGIC PLAN

What is Dakota

Who We Are

Trends and Opportunities

Where We Are Going

Sandia National Laboratories

2016—2020 Strategic Plan stewards Dakota's research program while improving production deployment and impact

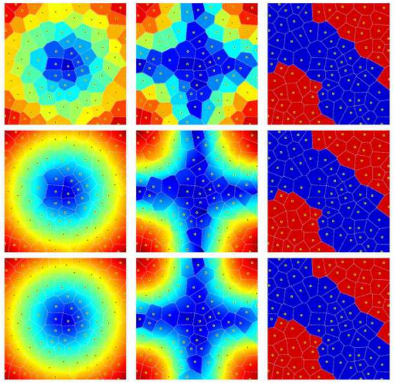
R&D Challenges

Tackle design exploration and UQ computational challenges (focusing on DOE national security mission):

- Severely constrained simulation budgets
- High-dimensional parameter spaces
- Non-smooth, multi-modal, or unreliable quantities of interest
- Quantifying small probabilities

Active Algorithm R&D

- Core UQ: robust, scalable, adaptive sampling and stochastic expansions
- Multi-fidelity, multi-level optimization, inference, UQ methods
- Calibration: usable, robust deterministic and Bayesian methods
- Optimization: new gradient-based and discrete optimization methods
- Algorithms and interfaces that directly treat functional (time/space) data
- Scalability in number of parameters / responses and to next-generation architectures and parallelism models
- Reduced-order models: surrogates, active subspace, random field models



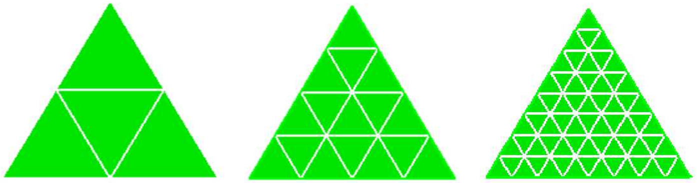
Discontinuity-detecting piecewise local surrogate models (Ebeida)

Multi-level/Multi-fidelity Methods

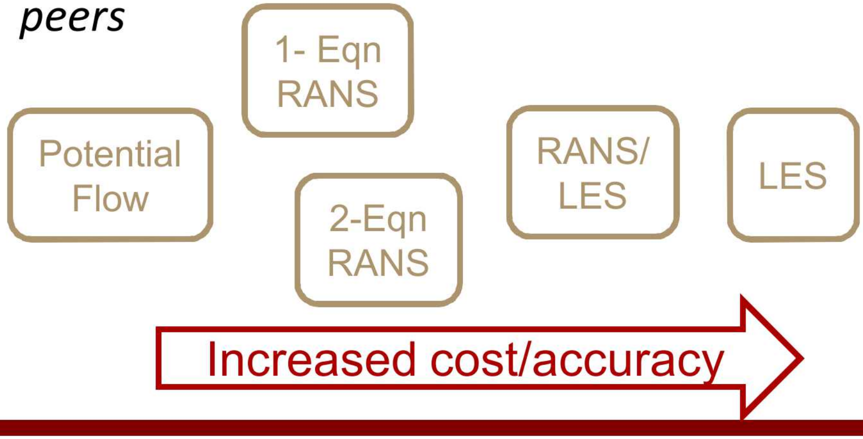
Goal: Use models of varying fidelities and/or levels to converge more quickly

- ML/MF sampling, stochastic expansions, optimization, calibration
- Automatically balance evaluations of each model

Multi-level: hierarchy of numerical approximation accuracies, *e.g., mesh resolution, solver tolerance*



Multi-fidelity: physics model detail; *hierarchy or peers*



Potential Flow

1- Eqn RANS

2-Eqn RANS

RANS/ LES

LES

Increased cost/accuracy

level

fidelity

$ 1000's	$$ 100's	$$$ 10's
$$ 100's	$$$ 10's	$$$$ 1

See talks by Eldred, Geraci

Inference: Bayesian Calibration

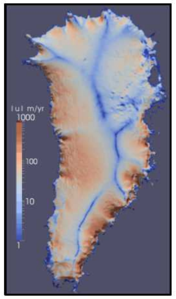
Goal: Obtain statistical characterization of parameters consistent with data

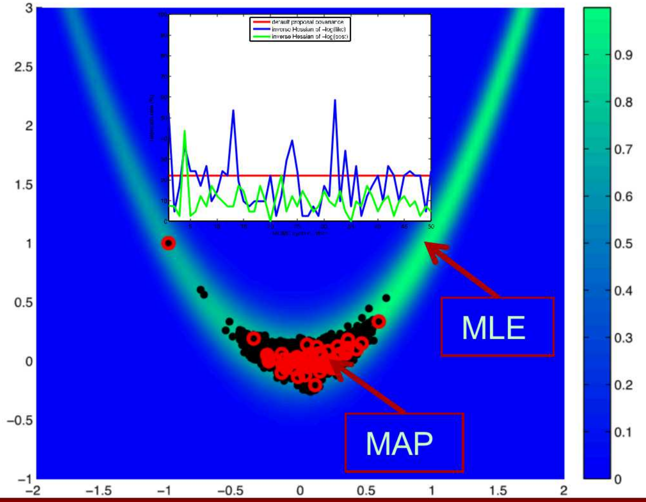
- MCMC with QUESO/DRAM and DREAM
- Adaptive, surrogate-based inference, including with ML/MF PCE
- Mutual information guided calibration of low-fi to high-fi model
- Usability: chain post-processing, statistics, credible/prediction intervals, KDE-smoothed posteriors, mutual information

Emerging capabilities

- Discrepancy models
- Metropolis-adjusted Langevin Algorithm
- Non-MCMC-based approaches
- Automated diagnostics and filtering
- Coupling with forward UQ methods

See talks by Jakeman, Maupin, Perego, Wildey





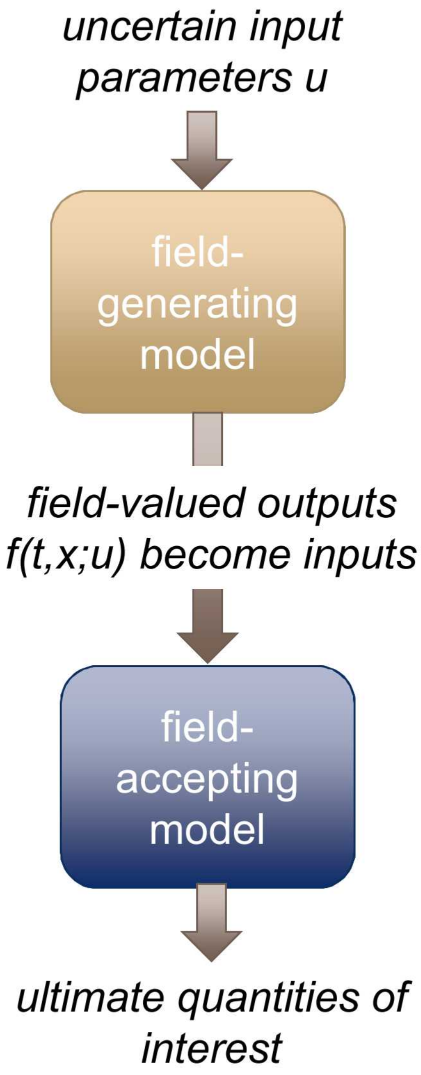
Random Field Modeling

Goal: perform UQ with field-valued (time- or space-varying) input uncertainties *f(t, x)*, e.g., boundary conditions

- Generate realizations of *f(t; u)*: either sample the field-generating model or use offline data
- Approximate uncertainty in *f(t; u)*, e.g., by PCA + GP, or a Karhunen–Loève expansion with normal coefficients ω

$$\hat{\boldsymbol{f}}(t,\omega) = \boldsymbol{\mu}_{\boldsymbol{f}}(\boldsymbol{t}) + \sum_{i=1}^{P} c_i(\omega)\,\varphi_i\ (t)$$

- Propagate: perform UQ over ω, generating realizations of the approximate field $\hat{\boldsymbol{f}}(t,\omega)$ and propagate through the field-accepting model



uncertain input parameters u

field-generating model

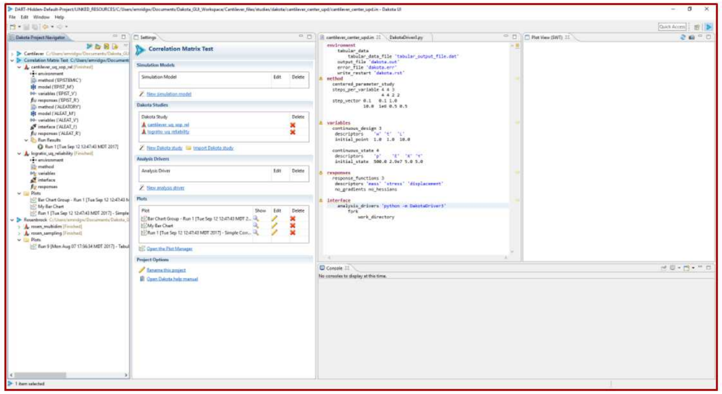
field-valued outputs f(t,x;u) become inputs

field-accepting model

ultimate quantities of interest

Other Development Areas

- Modular architecture
  - Ease Dakota development and give greater control to advanced users
  - Example 1: Surrogate model
  - Example 2: Optimizer traits and APIs



```
environment
    tabular_data
        tabular_data_file 'tabular_output_file.dat'
    output_file 'dakota.out'
    error_file 'dakota.err'
    write_restart 'dakota.rst'
method
  centered_parameter_study
  steps_per_variable 4 4 3
                     4 4 2 2
  step_vector 0.1   0.1 1.0
              10.0  1e6 0.5 0.5


variables
  continuous_design 3
   descriptors    'w' 't'  'L'
   initial_point  1.0  1.0  10.0

  continuous_state 4
   descriptors    'p'    'E'  'X'  'Y'
   initial_state  500.0 2.9e7 5.0 5.0

responses
  response_functions 3
   descriptors 'mass' 'stress' 'displacement'
   no_gradients no_hessians

interface
    analysis_drivers 'python -m DakotaDriver3'
        fork
            work_directory
```

No consoles to display at this time.

- Usability and user resources
  - Example 1: GUI
  - Example 2: Output database
  - Example 3: `dakota.interfacing` Python module
  - Example 4: Examples repo

- Improved user community engagement
  - Example 1: Dakota user-stakeholder meetings
  - Example 2: Surveys and interviews

DAKOTA

Explore and predict with confidence

http://dakota.sandia.gov

Thanks for your attention!

briadam@sandia.gov

jasteph@sandia.gov pdhough@sandia.gov

Other team members presenting at SIAM UQ:
Eldred, Geraci, Jakeman, Khalil, Maupin, Swiler

BACKUP (ADDITIONAL DETAILS)

HPC Integration

Dakota readily integrates into HPC environments

Massively Serial

Dakota runs in parallel and drives a large number of serial simulations

Evaluation Tiling

Dakota runs serially as part of a submitted job and "tiles" intermediate-sized parallel simulations across the job

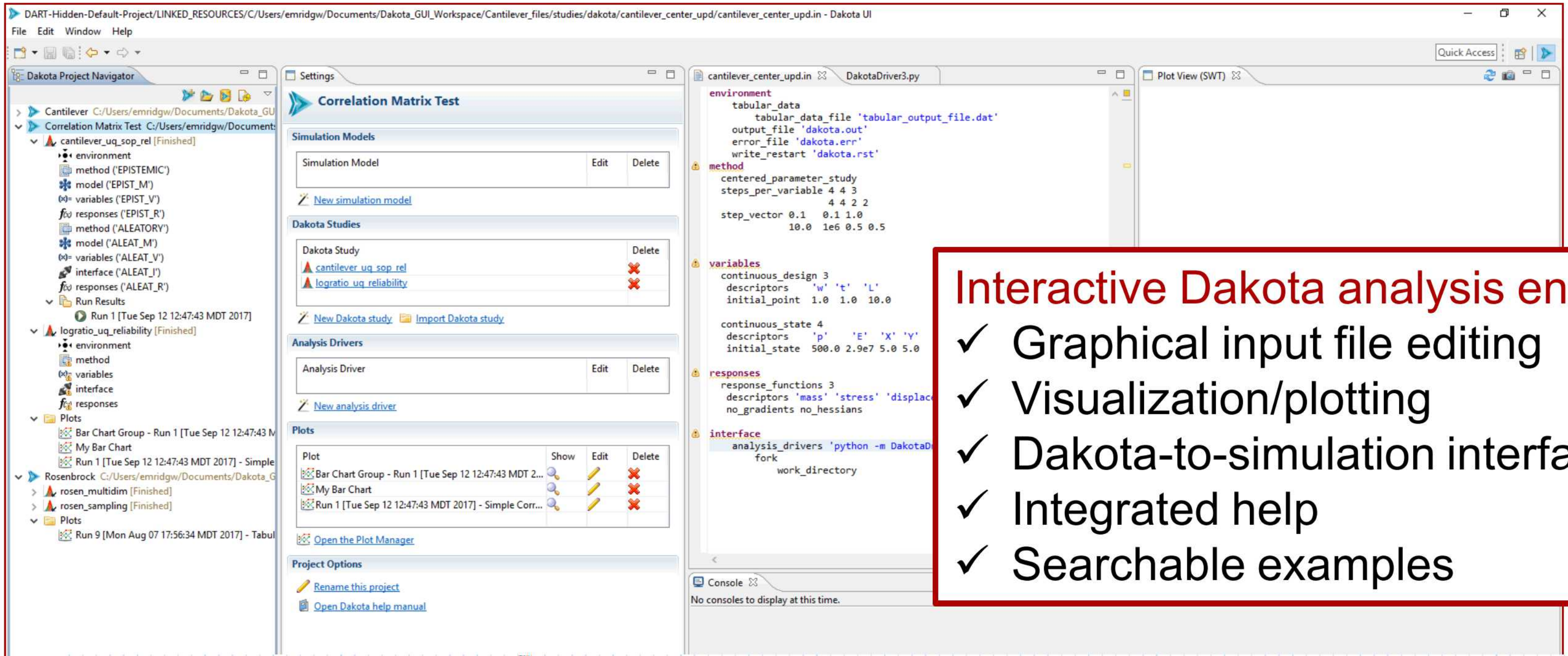
Evaluation Submission

Dakota runs serially on the login node and submits large parallel simulations as separate jobs.

Three common use cases

- ➢ Example interface scripts provided
- ➢ Python modules in development

Graphical User Interface



DART-Hidden-Default-Project/LINKED_RESOURCES/C/Users/emridgw/Documents/Dakota_GUI_Workspace/Cantilever_files/studies/dakota/cantilever_center_upd/cantilever_center_upd.in - Dakota UI

- Cantilever C:/Users/emridgw/Documents/Dakota_GU
- Correlation Matrix Test C:/Users/emridgw/Document
  - cantilever_uq_sop_rel [Finished]
    - environment
    - method ('EPISTEMIC')
    - model ('EPIST_M')
    - variables ('EPIST_V')
    - responses ('EPIST_R')
    - method ('ALEATORY')
    - model ('ALEAT_M')
    - variables ('ALEAT_V')
    - interface ('ALEAT_I')
    - responses ('ALEAT_R')
    - Run Results
      - Run 1 [Tue Sep 12 12:47:43 MDT 2017]
  - logratio_uq_reliability [Finished]
    - environment
    - method
    - variables
    - interface
    - responses
  - Plots
    - Bar Chart Group - Run 1 [Tue Sep 12 12:47:43 M
    - My Bar Chart
    - Run 1 [Tue Sep 12 12:47:43 MDT 2017] - Simple
- Rosenbrock C:/Users/emridgw/Documents/Dakota_G
  - rosen_multidim [Finished]
  - rosen_sampling [Finished]
  - Plots
    - Run 9 [Mon Aug 07 17:56:34 MDT 2017] - Tabul

1 item sele

Correlation Matrix Test

Simulation Models

Simulation Model	Edit	Delete

New simulation model

Dakota Studies

Dakota Study	Delete
cantilever_uq_sop_rel	✗
logratio_uq_reliability	✗

New Dakota study Import Dakota study

Analysis Drivers

Analysis Driver	Edit	Delete

New analysis driver

Plots

Plot	Show	Edit	Delete
Bar Chart Group - Run 1 [Tue Sep 12 12:47:43 MDT 2...	🔍	✎	✗
My Bar Chart	🔍	✎	✗
Run 1 [Tue Sep 12 12:47:43 MDT 2017] - Simple Corr...	🔍	✎	✗

Open the Plot Manager

Project Options

Rename this project

Open Dakota help manual

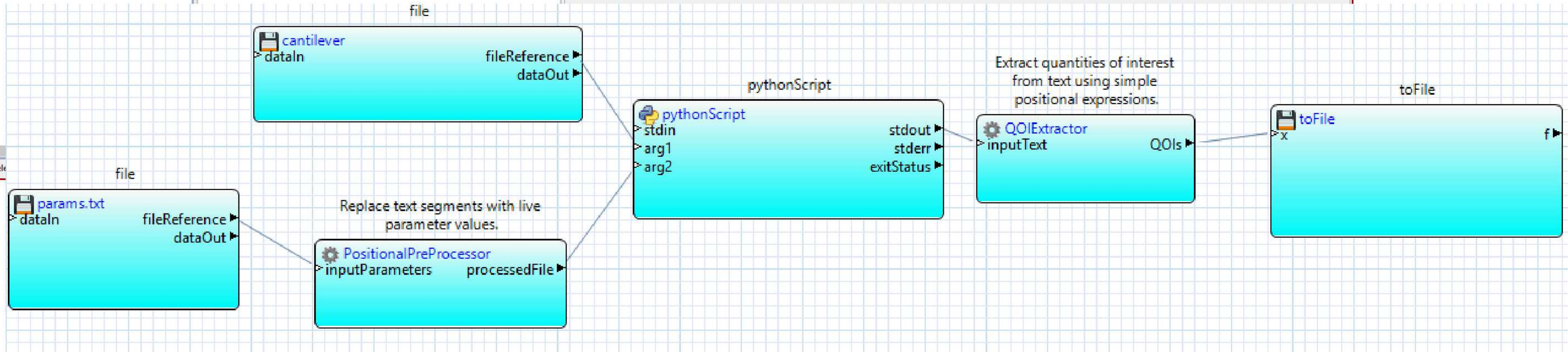
```
environment
    tabular_data
        tabular_data_file 'tabular_output_file.dat'
    output_file 'dakota.out'
    error_file 'dakota.err'
    write_restart 'dakota.rst'
method
  centered_parameter_study
  steps_per_variable 4 4 3
                     4 4 2 2
  step_vector 0.1   0.1 1.0
              10.0  1e6 0.5 0.5


variables
  continuous_design 3
   descriptors    'w' 't'  'L'
   initial_point  1.0  1.0  10.0

  continuous_state 4
   descriptors    'p'    'E'  'X' 'Y'
   initial_state  500.0 2.9e7 5.0 5.0

responses
  response_functions 3
   descriptors 'mass' 'stress' 'displac
   no_gradients no_hessians

interface
    analysis_drivers 'python -m DakotaD
        fork
            work_directory
```

Console

No consoles to display at this time.

Interactive Dakota analysis environment

- ✓ Graphical input file editing
- ✓ Visualization/plotting
- ✓ Dakota-to-simulation interface wizard
- ✓ Integrated help
- ✓ Searchable examples



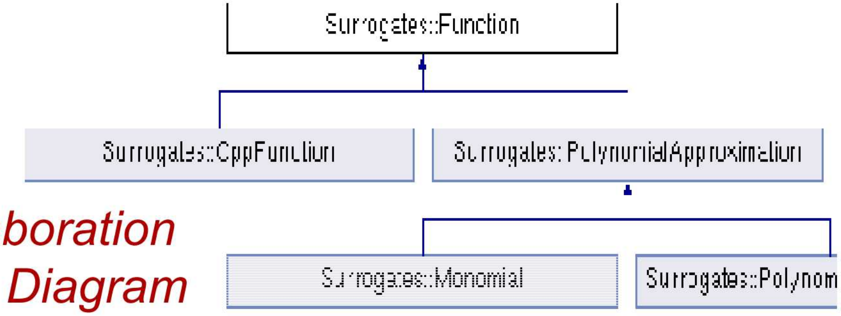
Extract quantities of interest from text using simple positional expressions.

Replace text segments with live parameter values.

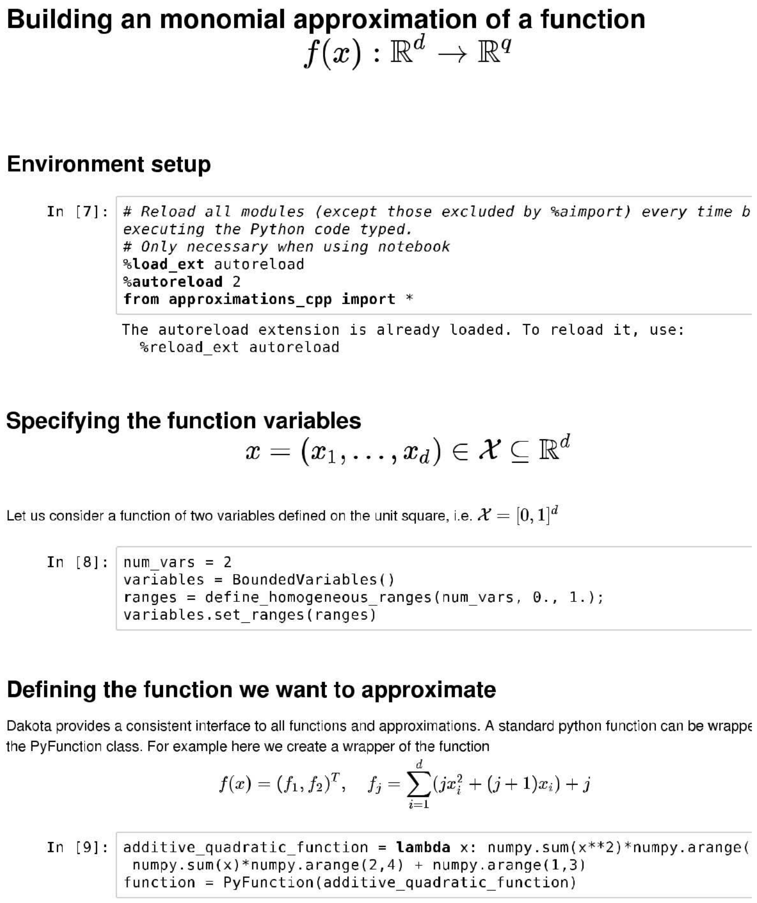
Dakota Modularization: Surrogate Model Library

Initial prototype demonstrated

- Modularize foundational utilities and surrogate models
- Progress on unifying Dakota, Surfpack, Pecos
- Increasing unit testing, coverage, and software quality
- Python interfaces to C++ components for ease of use by advanced users
- Jupyter-based interactive Python tutorial demonstrates more usable coordination of model components with Python



Collaboration Diagram



Building an monomial approximation of a function

$$f(x) : \mathbb{R}^d \to \mathbb{R}^q$$

Environment setup

```
In [7]: # Reload all modules (except those excluded by %aimport) every time b
        executing the Python code typed.
        # Only necessary when using notebook
        %load_ext autoreload
        %autoreload 2
        from approximations_cpp import *

        The autoreload extension is already loaded. To reload it, use:
          %reload_ext autoreload
```

Specifying the function variables

$$x = (x_1, \ldots, x_d) \in \mathcal{X} \subseteq \mathbb{R}^d$$

Let us consider a function of two variables defined on the unit square, i.e. $\mathcal{X} = [0,1]^d$

```
In [8]: num_vars = 2
        variables = BoundedVariables()
        ranges = define_homogeneous_ranges(num_vars, 0., 1.);
        variables.set_ranges(ranges)
```

Defining the function we want to approximate

Dakota provides a consistent interface to all functions and approximations. A standard python function can be wrappe the PyFunction class. For example here we create a wrapper of the function

$$f(x) = (f_1, f_2)^T, \quad f_j = \sum_{i=1}^{d}(jx_i^2 + (j+1)x_i) + j$$

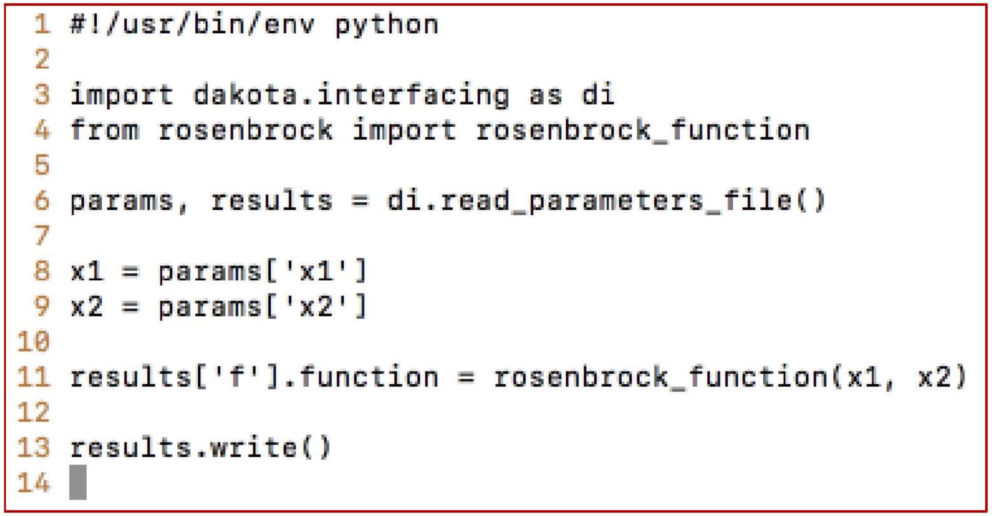
```
In [9]: additive_quadratic_function = lambda x: numpy.sum(x**2)*numpy.arange(
         numpy.sum(x)*numpy.arange(2,4) + numpy.arange(1,3)
        function = PyFunction(additive_quadratic_function)
```

iPython Notebook

dakota.interfacing

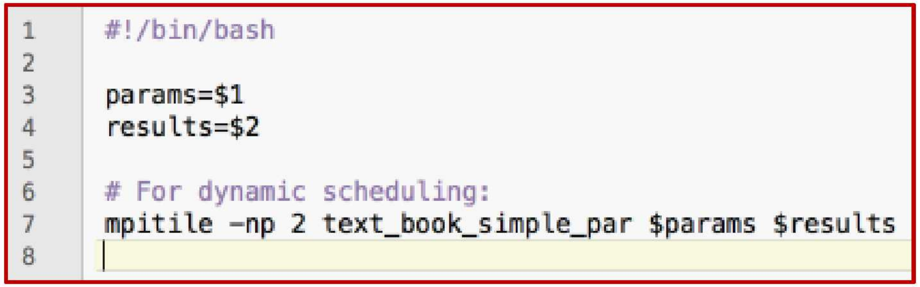
Tools for Python-based black-box simulation interfacing

- Read params files/write results files



```
#!/usr/bin/env python

import dakota.interfacing as di
from rosenbrock import rosenbrock_function

params, results = di.read_parameters_file()

x1 = params['x1']
x2 = params['x2']

results['f'].function = rosenbrock_function(x1, x2)

results.write()

```

- Evaluation tiling helper



```
#!/bin/bash

params=$1
results=$2

# For dynamic scheduling:
mpitile -np 2 text_book_simple_par $params $results

```

TODO

- ❑ Evaluation Submission helpers
- ❑ Template substitution
- ❑ Integration into GUI

Promote User and Development Community Engagement

- Web resources:
  - Interactive user forums
  - Capability maturity ratings and test linkage
  - Community repository of code, examples, scripts
- Training materials: presentations, videos, exercises
- New graphical user interface for Dakota analysis

- Improved modularity so users can extend, contribute components, e.g.,
  - Surrogate model module with Python bindings
  - More usable simulation interfacing that encourages best practices
- Communicate development practices to encourage contribution, e.g., principles, code standards, easier build/test on new platforms

- Portability to and scalability on new high-performance computers

DAKOTA

Explore and predict with confidence

http://dakota.sandia.gov

Thanks for your attention!

briadam@sandia.gov

jasteph@sandia.gov pdhough@sandia.gov

Other team members presenting at SIAM UQ:
Eldred, Geraci, Jakeman, Khalil, Maupin, Swiler