



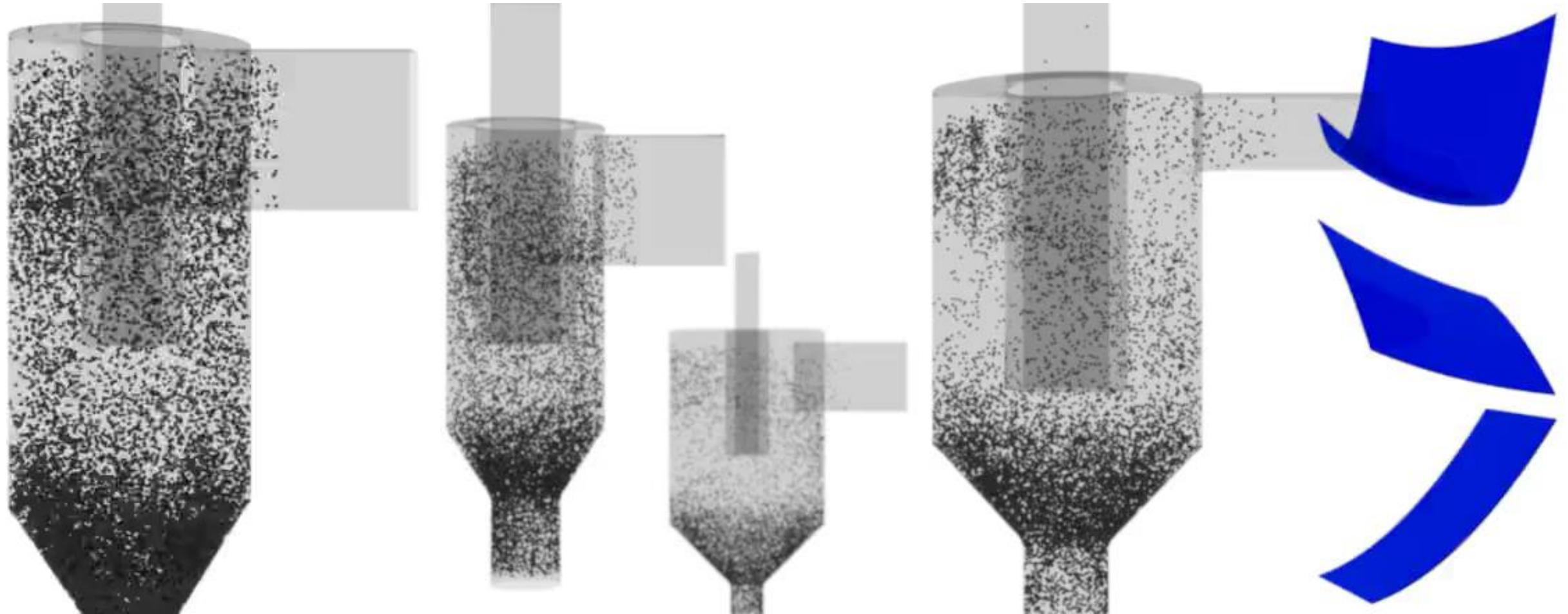
Nodeworks



August 28, 2020

Justin Weber | Aytekin Gel | Charles Tong

NETL | Alpemi, LLC | LLNL



So many options!



Next Generation Workflow (NGW)



Themis

Pegasus

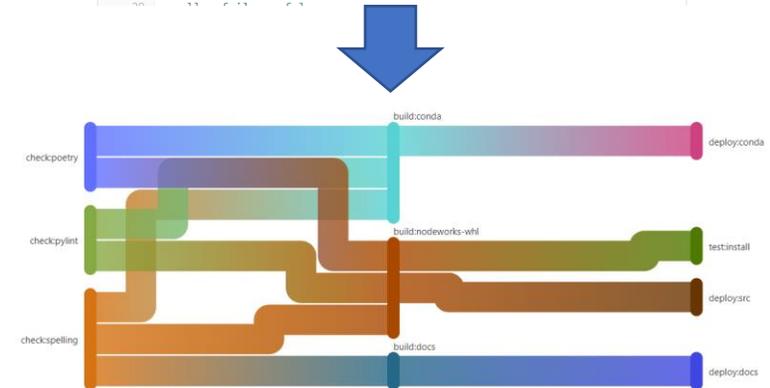
Low-Code/No-Code



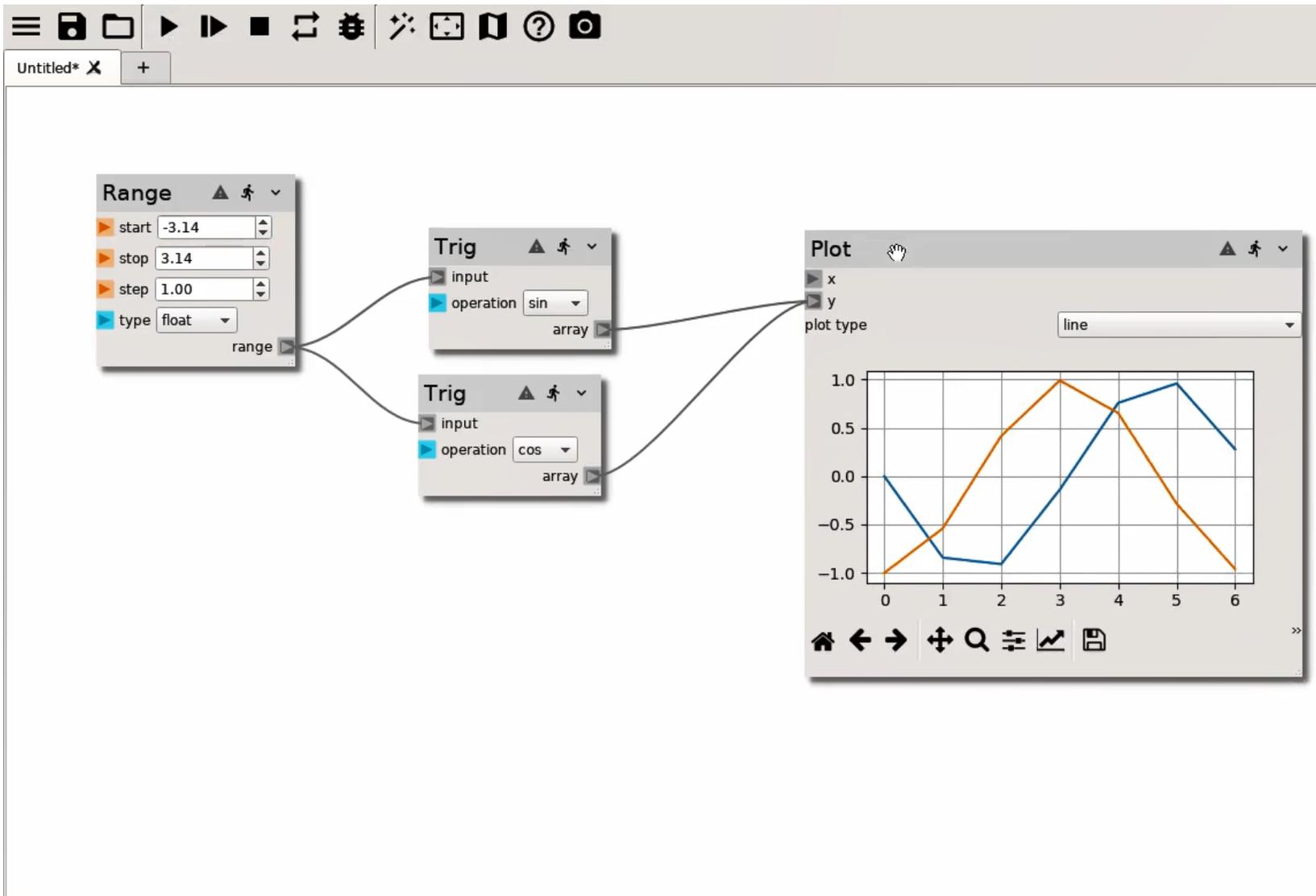
Apache Airflow

BEE

```
1 image: continuumio/anaconda3:2020.02
2
3 stages:
4   - check
5   - build
6   - test
7   - deploy
8
9 .only_template: &only_def
10 only:
11   - branches
12   - main
13   - tags
14
15 check:poetry:
16   stage: check
17   <<: *only_def
18   image: python:3.8-buster
19   variables:
20     QT_QPA_PLATFORM: "offscreen"
21   script:
22     - apt update && apt install xvfb -y
23     - pip install --upgrade pip
24     - python -m pip install poetry pytest pylint
25     - poetry config virtualenvs.create false
26     - poetry install
27     - pytest --verbose --cov=nodeworks --cov=tests tests
28     - pylint nodeworks
29
```

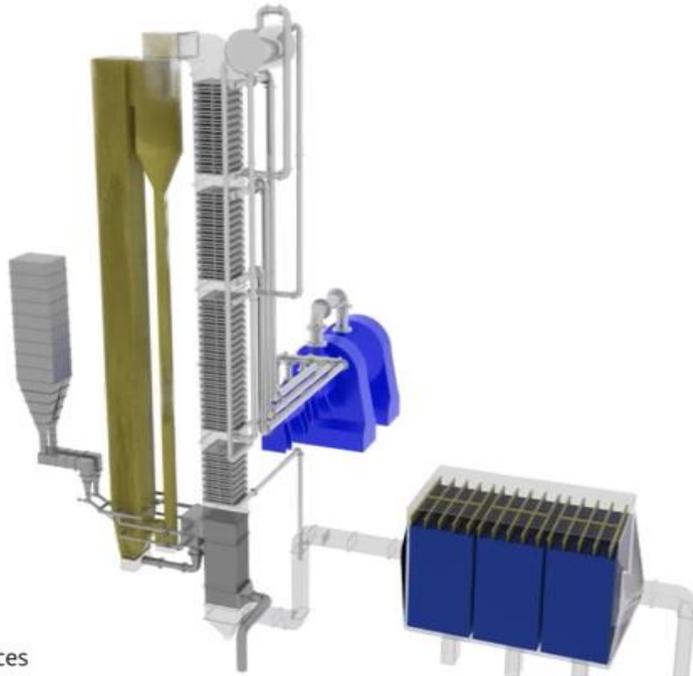


What is Nodeworks?

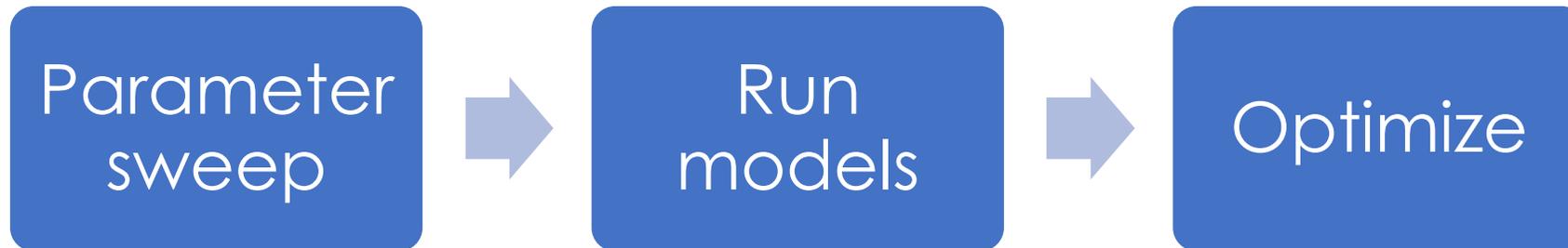


Application and
framework for
workflows

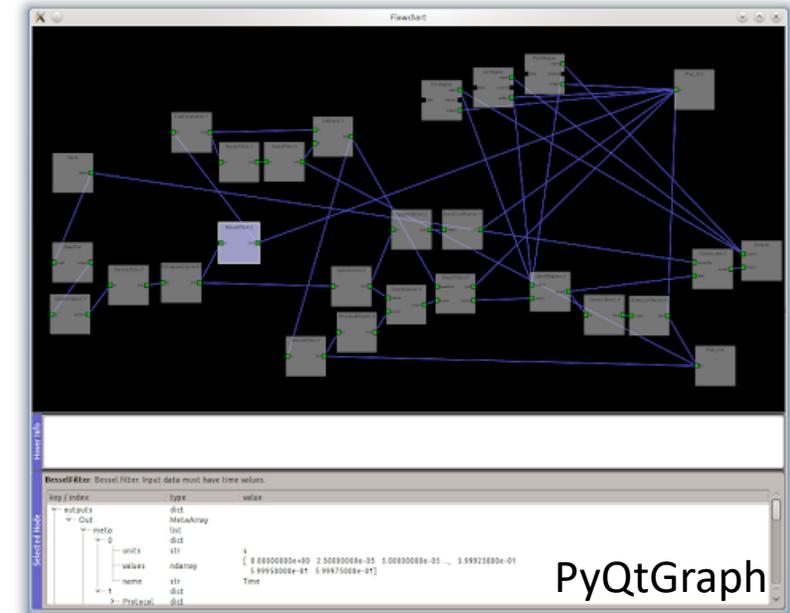
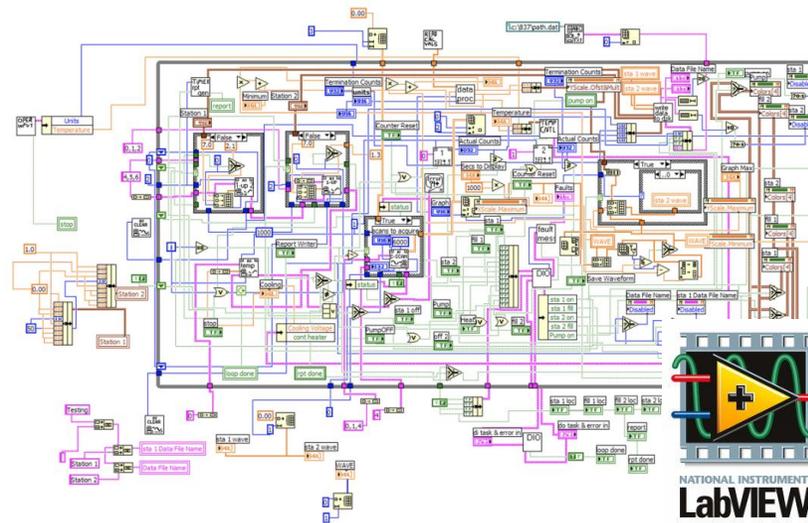
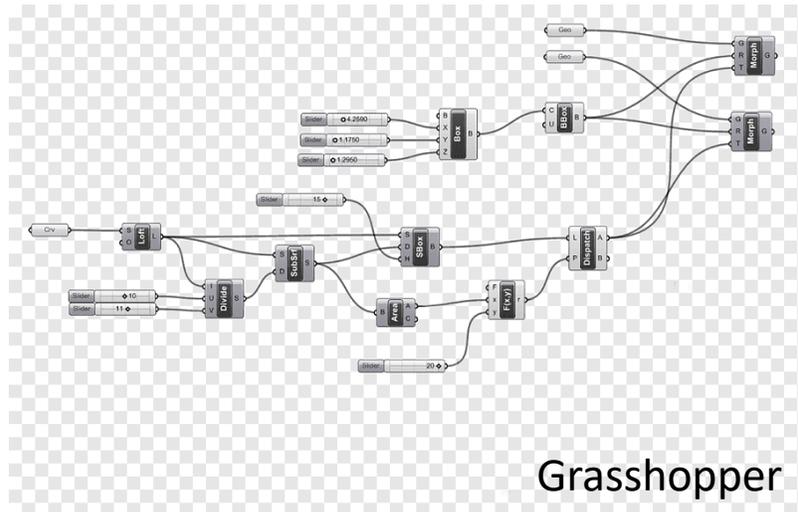
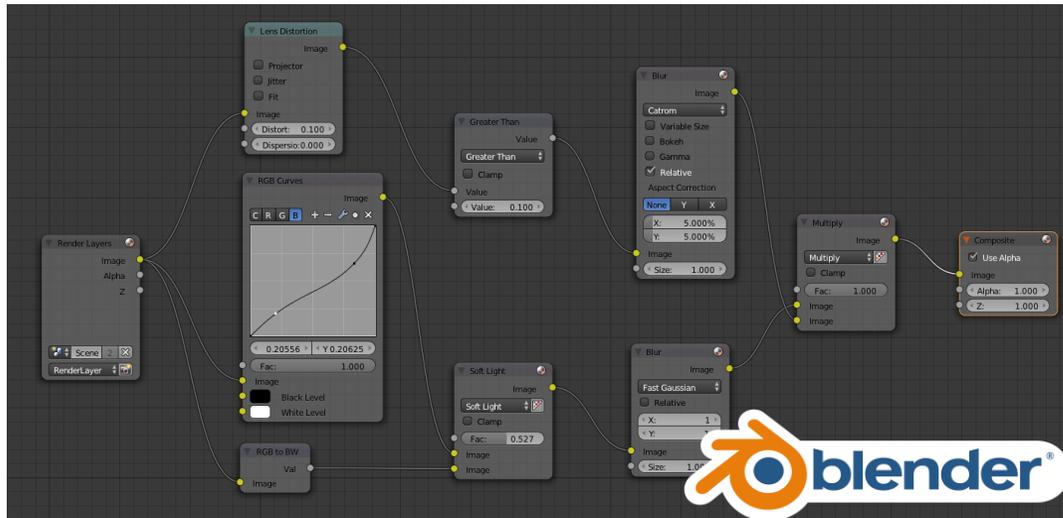
Why did you make this?



Justin Weber, Thermal Sciences



Why did you make this?



PLYNX

How is it constructed?

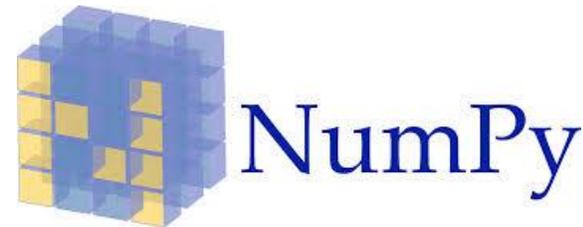


Tested operating systems:

- Linux (Centos @ Joule)
- OSX
- Windows

Deploy

- Conda
- Poetry
- Pyinstaller?



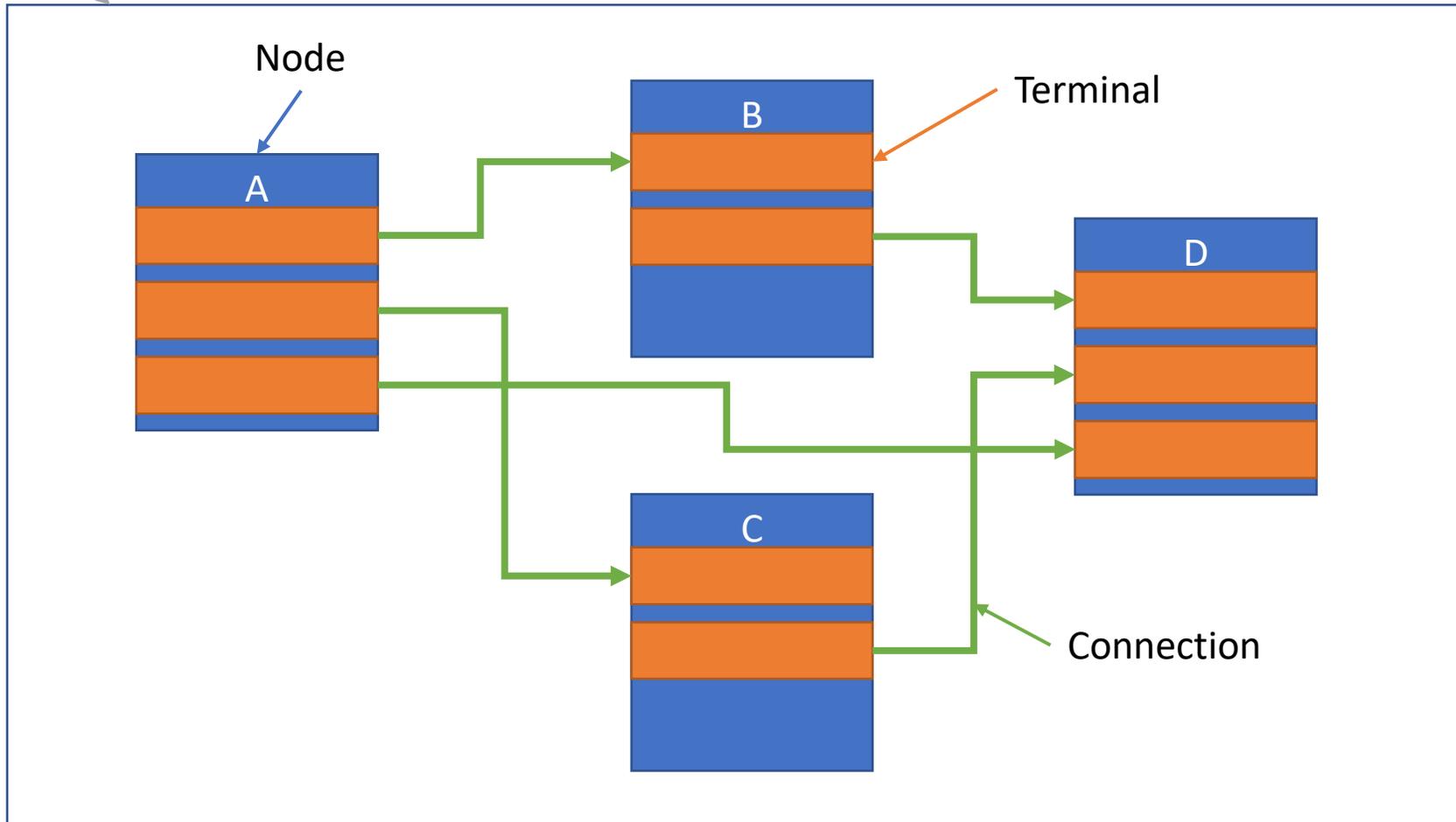
seaborn



matplotlib

How does it work?

Sheet

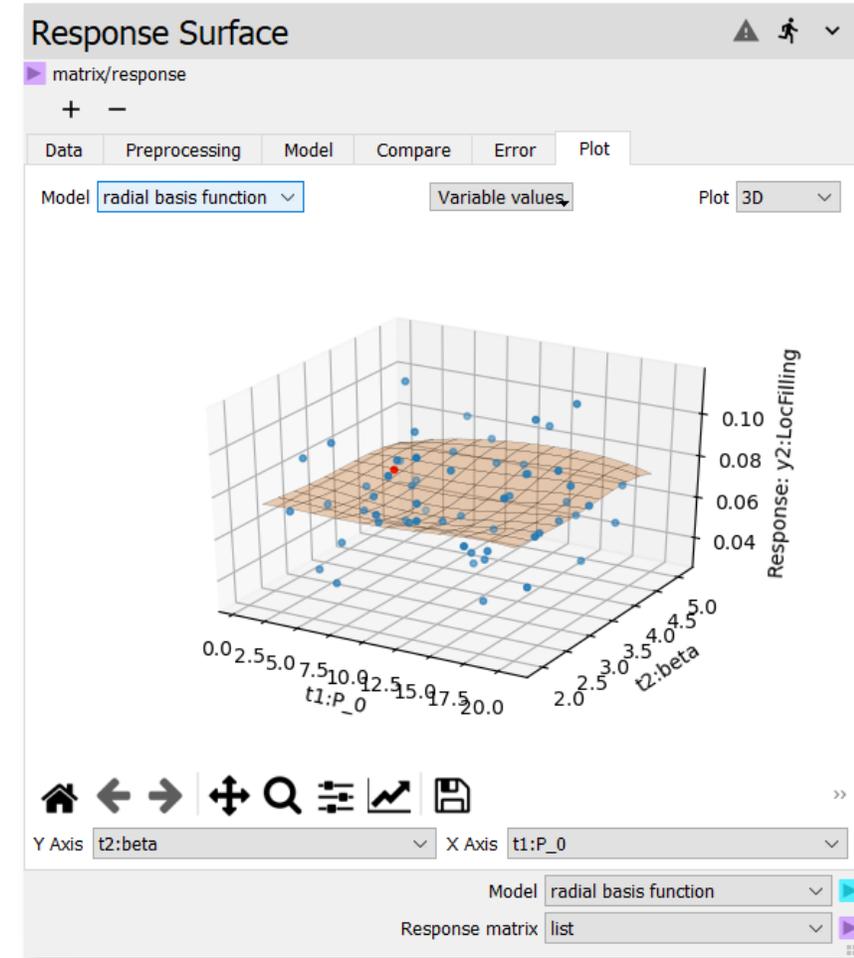


Topological sort

1. A
2. B, C
3. D

Super easy to make your own nodes!

```
class FloatNode(Node):  
    '''  
    Float Type  
  
    Output Terminals  
    -----  
    (float):  
        the float value of the spinbox  
    ...  
    name = 'Float'  
  
    def __init__(self):  
        self.terminalOpts = {  
            'float': {'widget': 'doublespinbox',  
                    'out': True,  
                    'dtype': float, },  
        }  
        Node.__init__(self)
```



Surrogate modeling and analysis toolset

Design of Experiments



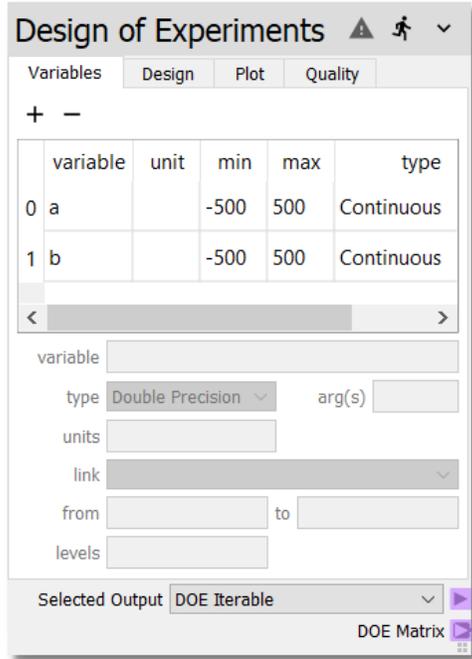
Model evaluation



Response Surface Construction



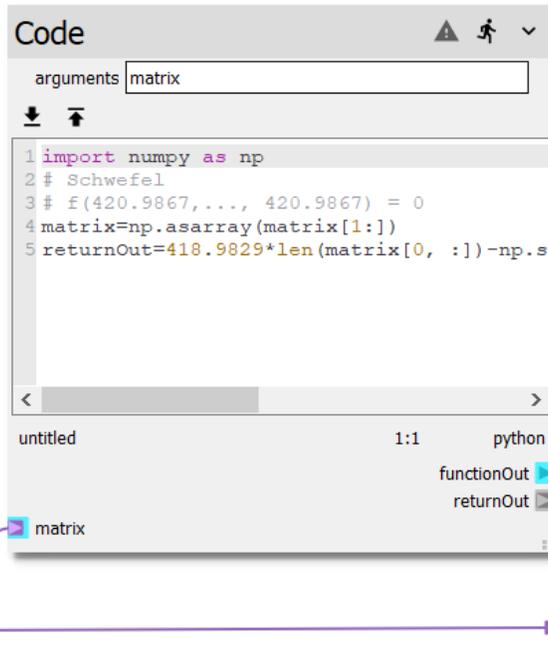
Optimization
Sensitivity
Forward Propagation



Design of Experiments

variable	unit	min	max	type
0 a		-500	500	Continuous
1 b		-500	500	Continuous

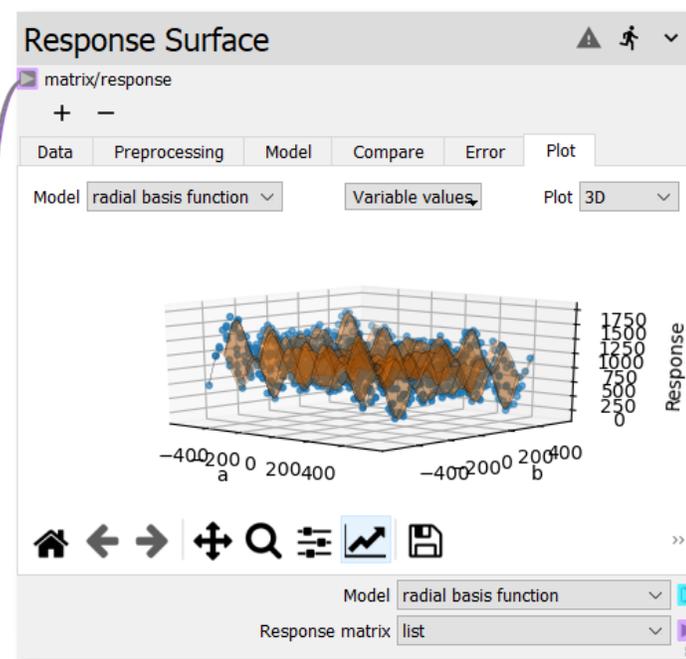
Selected Output: DOE Iterable
DOE Matrix



Code

```
arguments matrix  
1 import numpy as np  
2 # Schwefel  
3 f(420.9867,..., 420.9867) = 0  
4 matrix=np.asarray(matrix[1:])  
5 returnOut=418.9829*len(matrix[0, :])-np.s
```

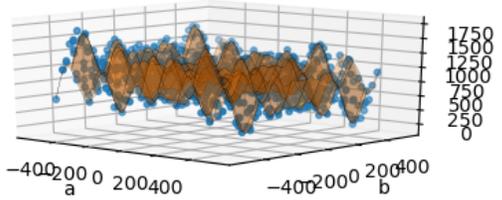
untitled 1:1 python
functionOut
returnOut



Response Surface

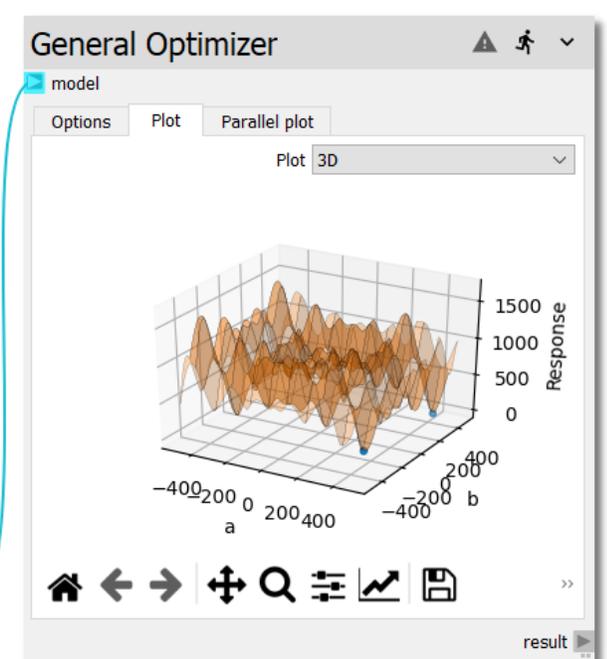
matrix/response

Model: radial basis function
Variable values
Plot: 3D



Response

Model: radial basis function
Response matrix: list

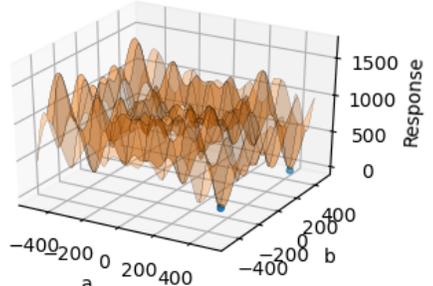


General Optimizer

model

Options Plot Parallel plot

Plot: 3D



Response

result

Design of Experiments

Design of Experiments

Variables Design Plot Quality

	variable	unit	min	max	type
1	ug	m/s	0	1	Continuous
2	beta	-	1	100	Continuous
3	alpha	-	1	5	Continuous

variable: alpha

type: Double Precision

units: -

link: None

from: 1 to 5

levels: 2

Selected Output: DOE Iterable

DOE Matrix

Factorial
Covary

Design of Experiments

Variables Design Plot Quality

Method: latin hypercube

Samples: 30

Optimize: genetic Iterations: 100

Randomize seed: 7835326

Repeat: 1 random samples: 1 times.

Randomize sample order

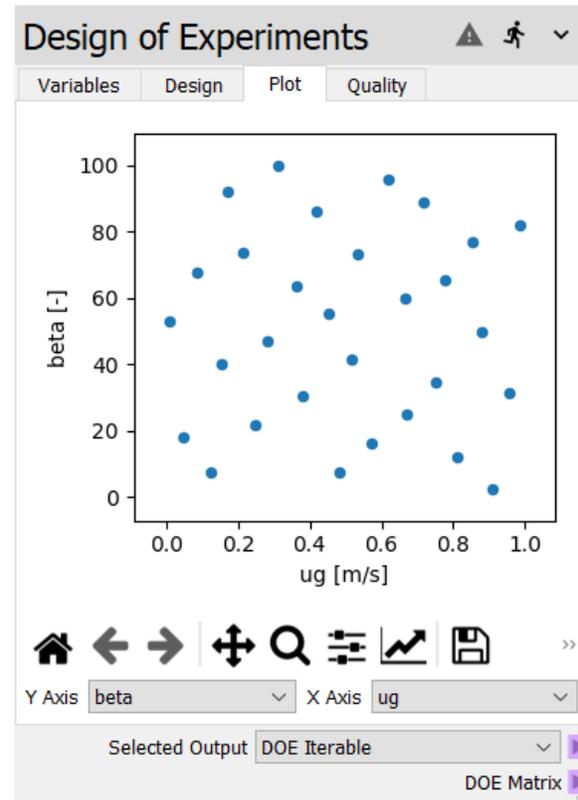
Import Build Export

	ug	beta	alp
1	0.753	34.8	1.7
2	0.879	49.9	4.14
3	0.669	25	2.83

Selected Output: DOE Iterable

DOE Matrix

Montecarlo
Latin hypercube



Central composite
Sobol

Design of Experiments

Variables Design Plot Quality

Pairwise distance based metrics

Distance Metric: euclidean

Min Distance: 0.147

Max Distance: 1.52

Ratio (Max/Min): 10.4

L2-discrepancy: 0.00398

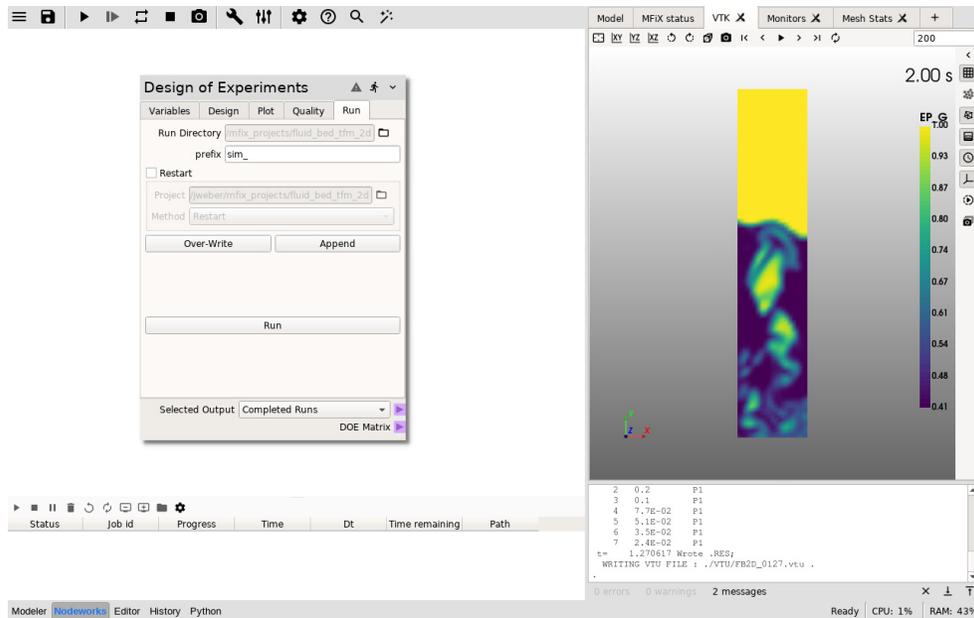
Selected Output: DOE Iterable

DOE Matrix

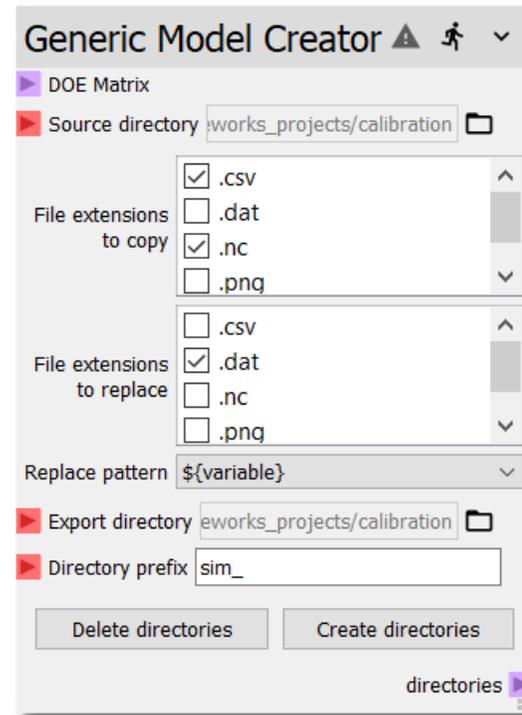
Hammersly
Halton

Building the models

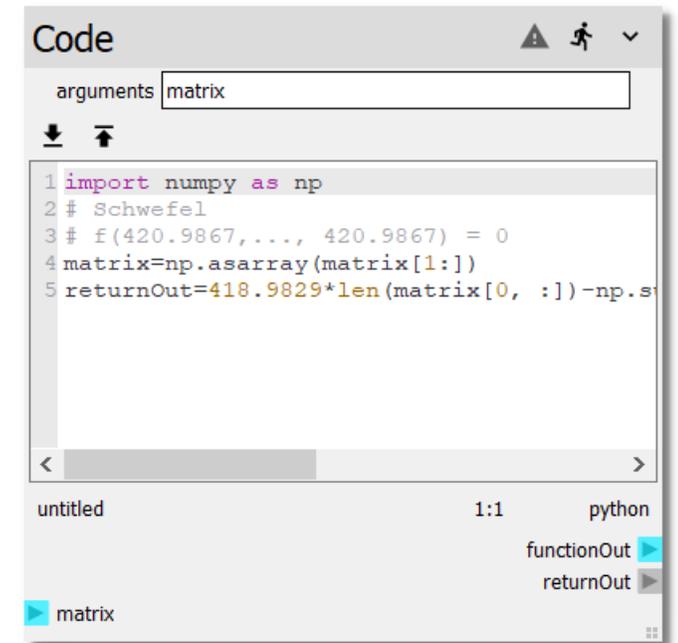
From within the MFiX GUI



Search/replace text

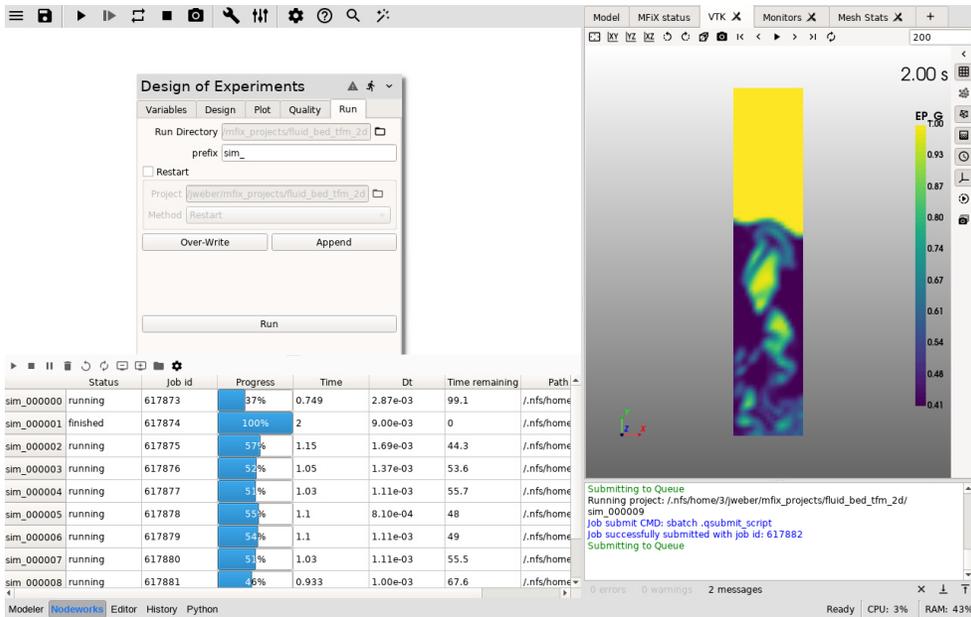


Custom python script



Running the models

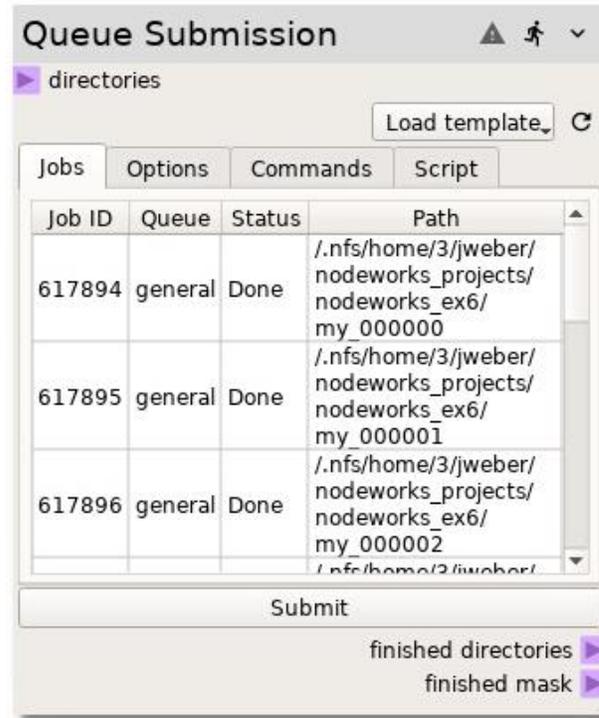
From within the MFiX GUI



The screenshot shows the MFiX GUI interface. On the left, the 'Design of Experiments' panel is visible, with fields for 'Run Directory' and 'Project'. Below it is a table showing the status and progress of multiple simulation jobs.

Status	Job id	Progress	Time	Dt	Time remaining	Path	
sim_000000	running	617873	37%	0.749	2.87e-03	99.1	./nfs/home
sim_000001	finished	617874	100%	2	9.00e-03	0	./nfs/home
sim_000002	running	617875	57%	1.15	1.69e-03	44.3	./nfs/home
sim_000003	running	617876	52%	1.05	1.37e-03	53.6	./nfs/home
sim_000004	running	617877	53%	1.03	1.11e-03	55.7	./nfs/home
sim_000005	running	617878	55%	1.1	8.10e-04	48	./nfs/home
sim_000006	running	617879	54%	1.1	1.11e-03	49	./nfs/home
sim_000007	running	617880	53%	1.03	1.11e-03	55.5	./nfs/home
sim_000008	running	617881	56%	0.933	1.00e-03	67.6	./nfs/home

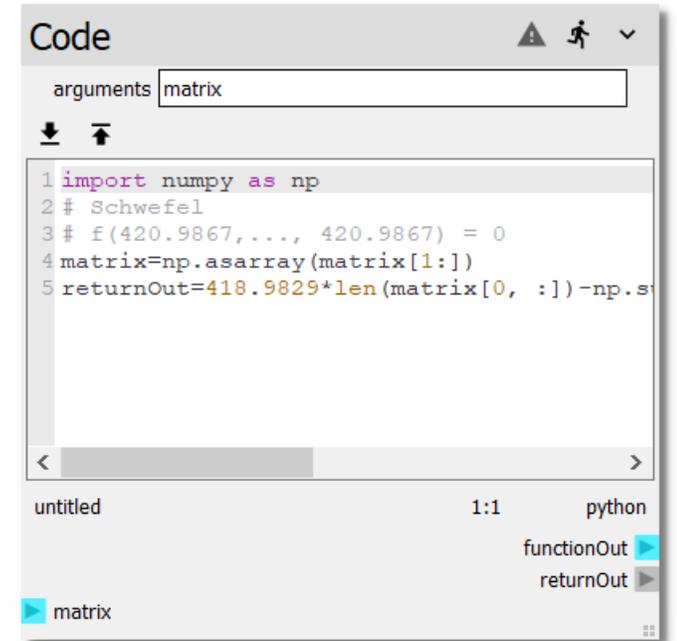
Queue submission



The screenshot shows the 'Queue Submission' dialog box. It contains a table with columns for Job ID, Queue, Status, and Path. The jobs listed are 617894, 617895, and 617896, all with a status of 'Done'.

Job ID	Queue	Status	Path
617894	general	Done	./nfs/home/3/jweber/nodeworks_projects/nodeworks_ex6/my_000000
617895	general	Done	./nfs/home/3/jweber/nodeworks_projects/nodeworks_ex6/my_000001
617896	general	Done	./nfs/home/3/jweber/nodeworks_projects/nodeworks_ex6/my_000002

Custom python script



The screenshot shows a code editor window with a Python script. The script imports numpy and performs a calculation on a matrix.

```
1 import numpy as np
2 # Schwefel
3 # f(420.9867,..., 420.9867) = 0
4 matrix=np.asarray(matrix[1:])
5 returnOut=418.9829*len(matrix[0,:])-np.s
```

Response Surface

Response Surface

matrix/response

Data Preprocessing Model Compare Error Plot

Cross validation points 10 % Repeats 10

fit	Model	MSE	SSE	R ²	L _{inf}
✓	gaussian process	7.52e+05	7.52e+07	-10	1
✓	polynomial	5.82e+04	5.82e+06	0.161	0.412
✓	multilayer perceptron	1.06e+05	1.06e+07	-0.522	0.542
✓	support vector machine	6.74e+04	6.74e+06	0.0297	0.472
✓	decision tree	2.15e+04	2.15e+06	0.686	0.369

kernel: 1*RBF
alpha: 1e-10
optimizer: fmin_l_bfgs_b
optimizer restarts: 0
 normalize

Refit Model(s)

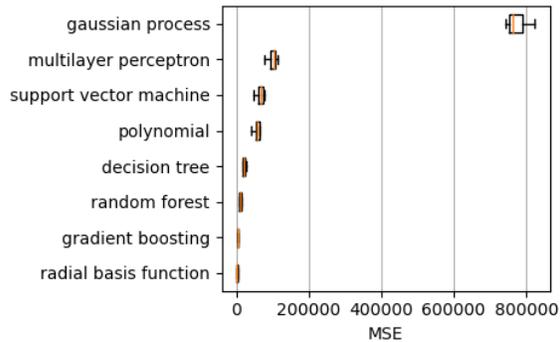
Model: radial basis function
Response matrix: list

Response Surface

matrix/response

Data Preprocessing Model Compare Error Plot

Error metric MSE Plot Type boxplot



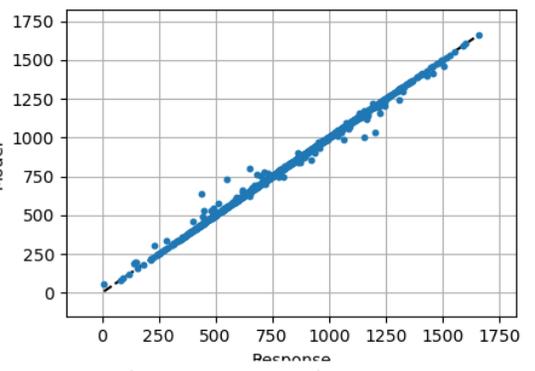
Model: gaussian process
Response matrix: list

Response Surface

matrix/response

Data Preprocessing Model Compare Error Plot

Model: radial basis function Plot: parity



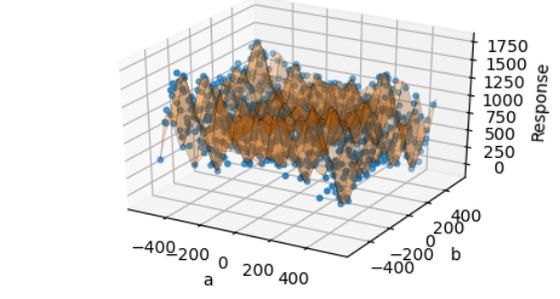
Model: gaussian process
Response matrix: list

Response Surface

matrix/response

Data Preprocessing Model Compare Error Plot

Model: radial basis function Variable values Plot: 3D



Model: gaussian process
Response matrix: list



Neural Net Regressor

Neural Net Regressor

matrix/response

Data Model Train Error Plot

Available layers

Model layers

Layer options

Search...

- Identity
- Linear
- ELU
- CELU
- GELU
- ReLU
- ReLU6
- LeakyReLU
- Sigmoid
- Hardsigmoid
- LogSigmoid
- Softplus
- Softsign
- Tanh
- Tanhshrink
- Threshold

linear.0
relu.0
linear.1
relu.1
linear.2
relu.2
linear.3

Features 100

Response matrix list

Neural Net Regressor

matrix/response

Data Model Train Error Plot

Optimizer Adam

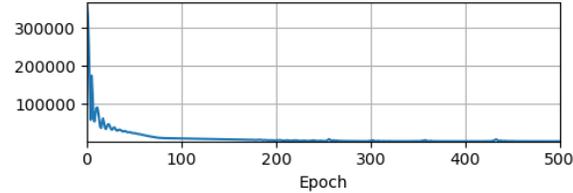
Learning rate 0.01

Loss function MSE

Epochs 500

Test 0.0 %

Train



300000
200000
100000
0

0 100 200 300 400 500

Epoch

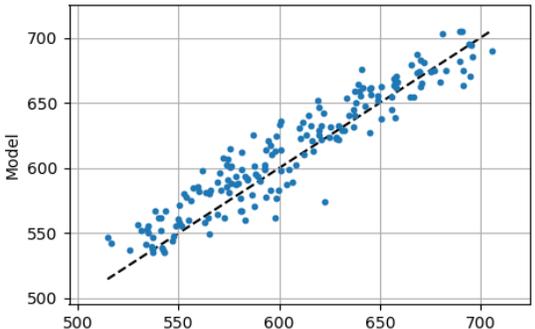
Response matrix list

Neural Net Regressor

matrix/response

Data Model Train Error Plot

Plot parity



700
650
600
550
500

500 550 600 650 700

Response: v1:dp2

x=691.929 y=586.504

Response matrix list

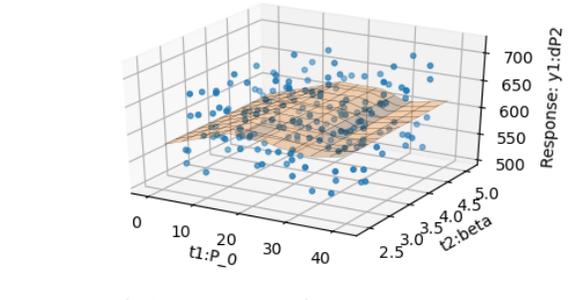
Neural Net Regressor

matrix/response

Data Model Train Error Plot

Variable values

Plot 3D



700
650
600
550
500

Response: y1:dp2

0 10 20 30 40

t1:P_0

2.5 3.0 3.5 4.0 4.5 5.0

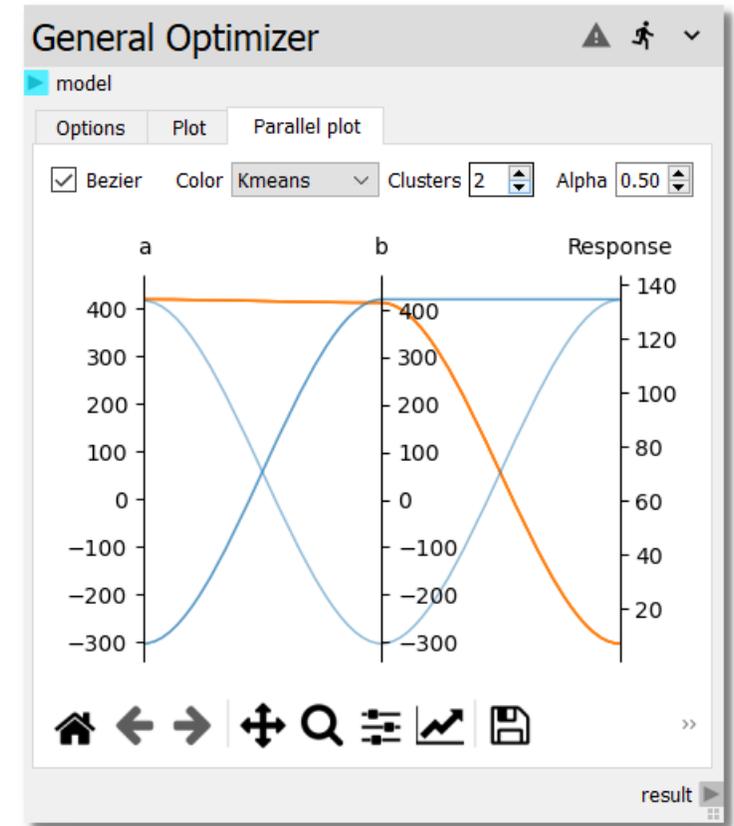
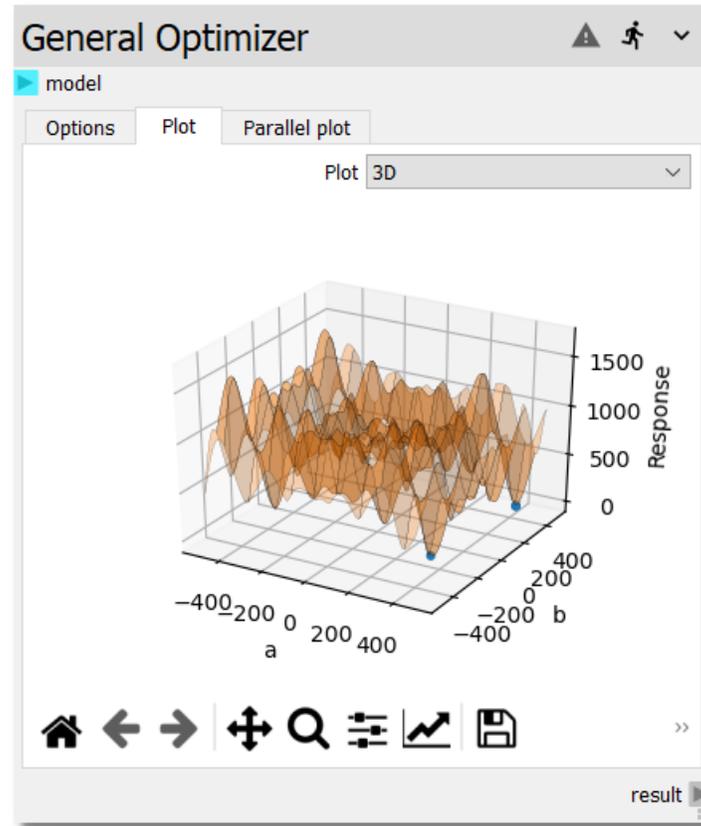
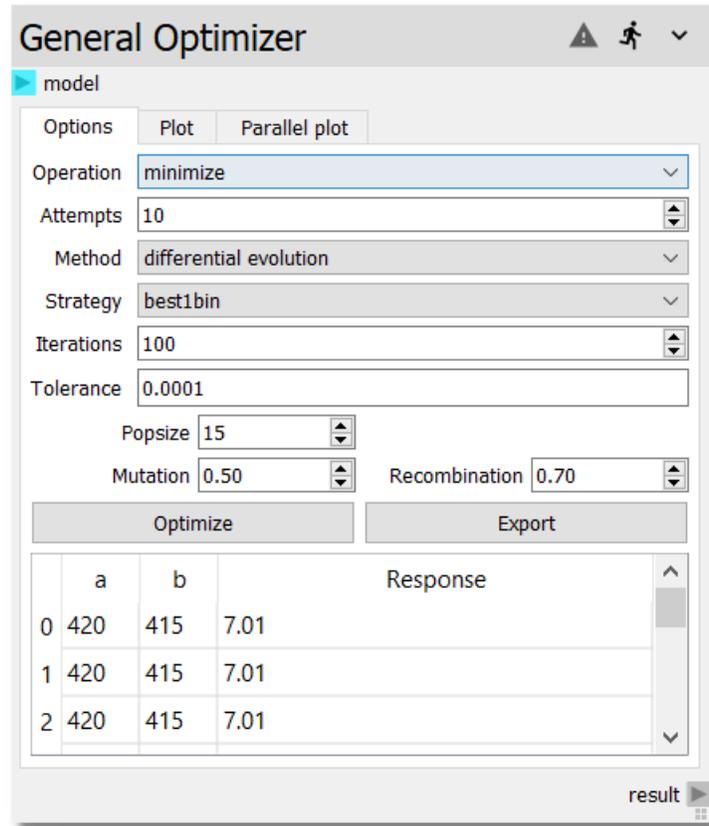
t2:beta

Y Axis t2:beta X Axis t1:P_0

Response matrix list



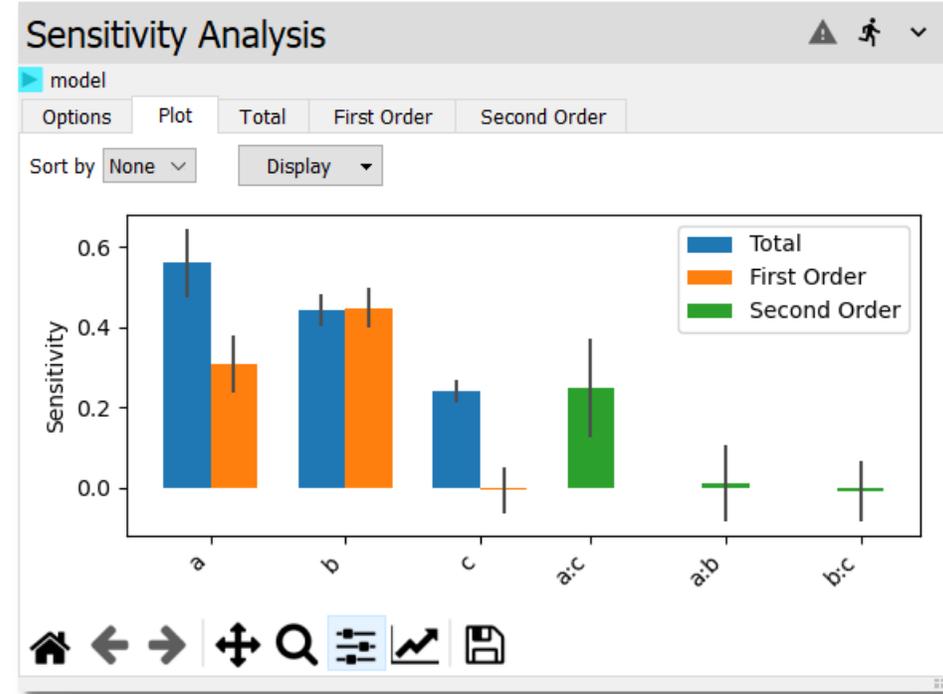
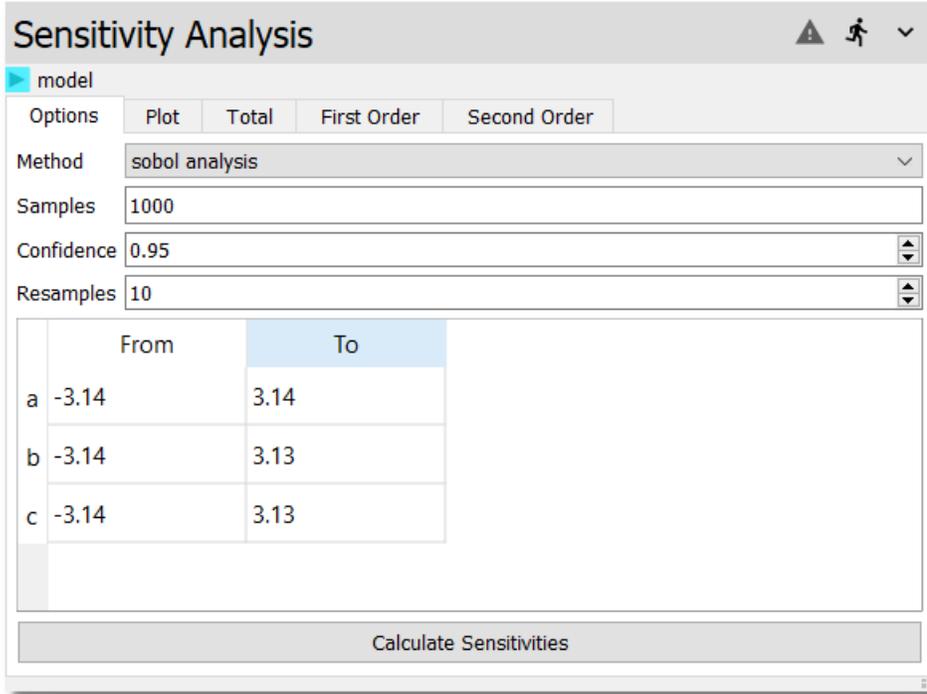
Optimization



Differential evolution
Basin hopping
SHGO - simplicial homology global optimization
Dual annealing



Sensitivity Analysis



- Sobol
- Method of Morris
- Fourier amplitude sensitivity test
- Delta moment-independent measure
- Random balance designs Fourier applicated sensitivity test

SALib

Forward Propagation

Forward Propagation

model

Options Bounds Probability Box

Aleatory samples 1000 Samples outside range re-draw

Epistemic samples 100 Epistemic Method latin hypercube

	type	distribution	from	to	m
a	aleatory	normal	-3.14	3.14	0
b	aleatory	normal	-3.14	3.14	0

Variable Type aleatory

Distribution normal

Mean 0.0

Standard Deviation 0.8

Calculate Propagation

Forward Propagation

model

Options Bounds Probability Box

The probability that the value will be 4 or less is between 87.4 % and 93 %

Draw on probability box

Given the prescribed input uncertainties with 95 % probability, the quantity of interest will be between 4.51 and 5.77

Draw on probability box

Export bounds to file

Calculate Propagation

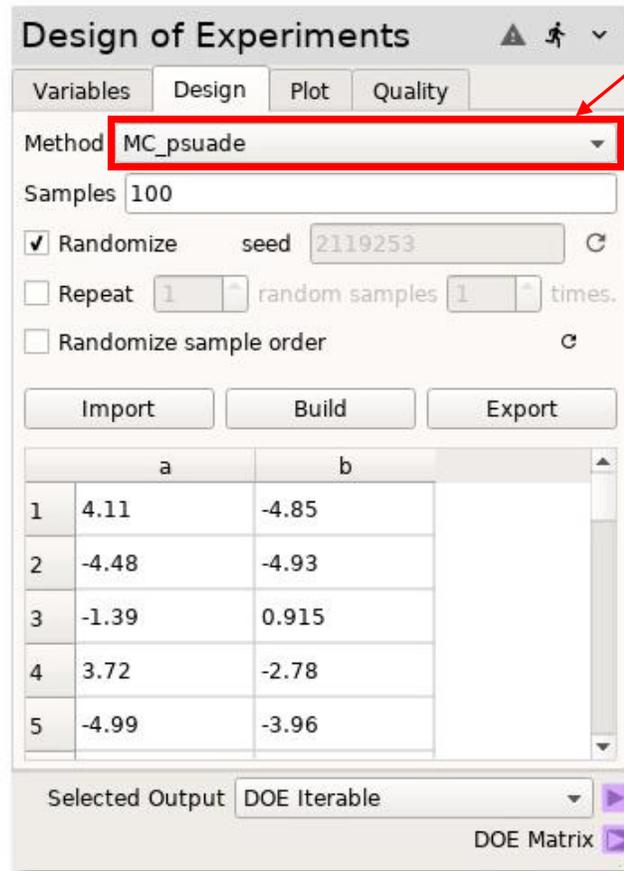
Forward Propagation

model

Options Bounds Probability Box

Calculate Propagation

Integration with PSUADE UQ Toolkit from LLNL



Design of Experiments

Variables Design Plot Quality

Method **MC_psuade**

Samples 100

Randomize seed 2119253

Repeat 1 random samples 1 times.

Randomize sample order

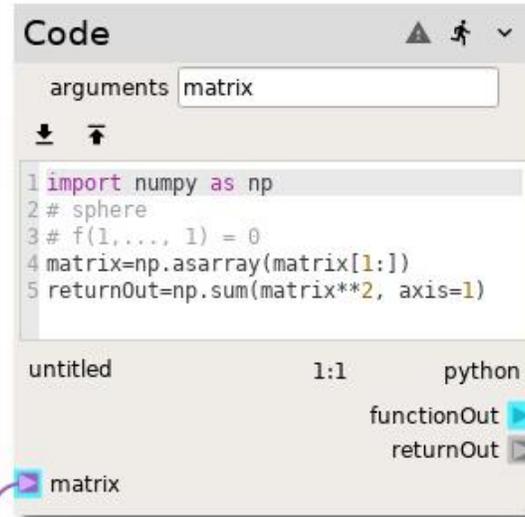
Import Build Export

	a	b
1	4.11	-4.85
2	-4.48	-4.93
3	-1.39	0.915
4	3.72	-2.78
5	-4.99	-3.96

Selected Output DOE Iterable

DOE Matrix

PSUADE's Monte Carlo Sampling Method selected



Code

arguments matrix

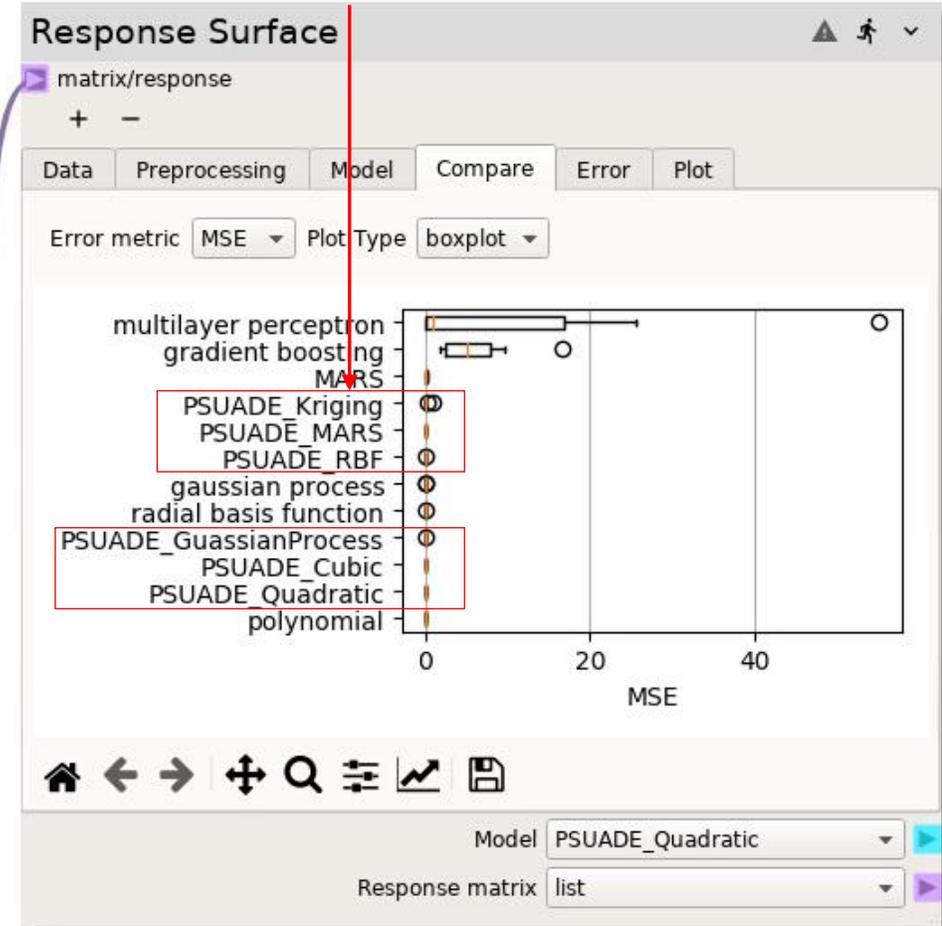
```
1 import numpy as np
2 # sphere
3 # f(1,..., 1) = 0
4 matrix=np.asarray(matrix[1:])
5 returnOut=np.sum(matrix**2, axis=1)
```

untitled 1:1 python

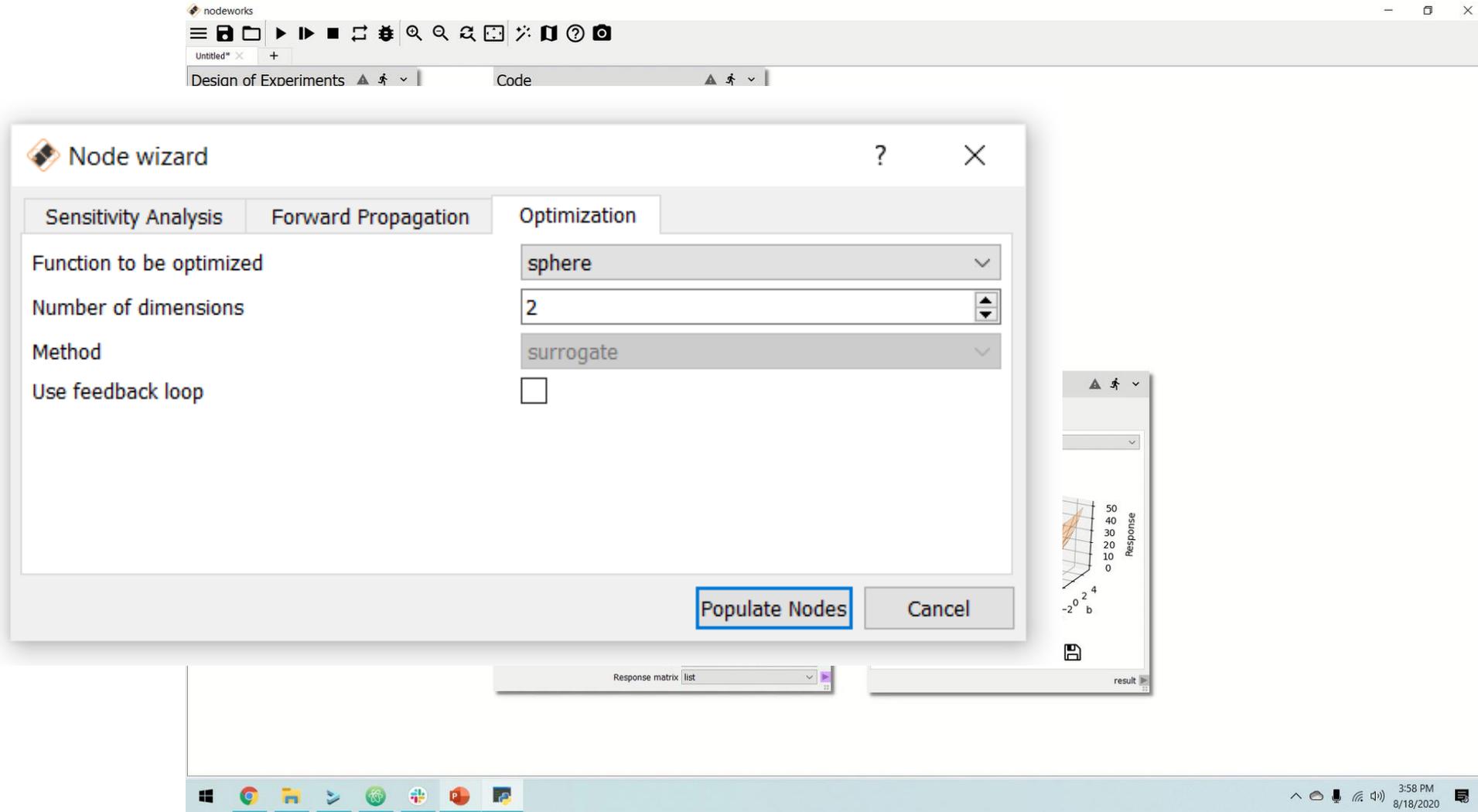
functionOut returnOut

matrix

PSUADE's Response Surface Methods also tested



Wizard for Quick Setup of Workflow Templates



nodeworks

Untitled* x +

Desian of Experiments ▲ ↻ ▾ | Code ▲ ↻ ▾

Node wizard ? X

Sensitivity Analysis Forward Propagation Optimization

Function to be optimized sphere ▾

Number of dimensions 2

Method surrogate ▾

Use feedback loop

Populate Nodes Cancel

Response matrix list

result

Response

50
40
30
20
10
0

-2 0 2 4
a b

Windows taskbar: 3:58 PM 8/18/2020

Feedback

Design of Experiments

Variables Design Plot Quality

Method latin hypercube

Samples 25

Optimize none Iterations 100

Randomize seed 4840932

Repeat 1 random samples 1 times.

Randomize sample order

	a	b
1	274	-124
2	-300	288
3	79.4	-241
4	-196	-356

Selected Output DOE Iterable

DOE Matrix

Code

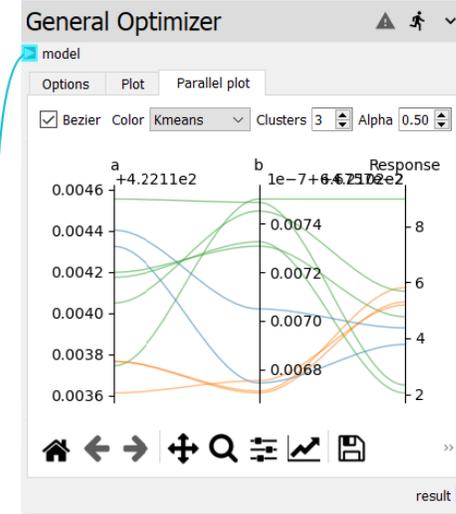
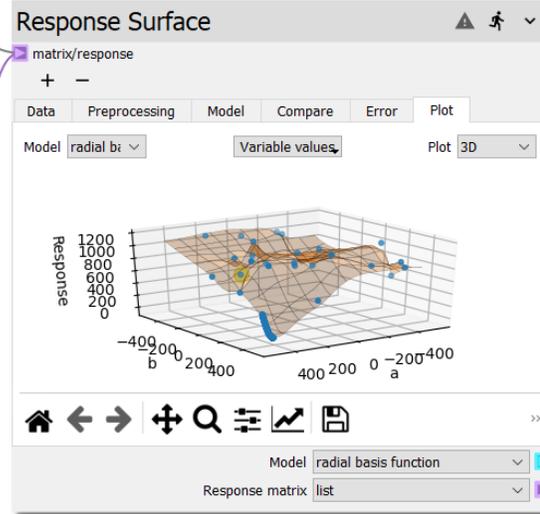
```
arguments matrix
```

```
1 import numpy as np
2 # Schwefel
3 # f(420.9867,..., 420.9867) = 0
4 matrix=np.asarray(matrix[1:])
5 returnOut=418.9829*len(matrix[0, :])-np.st
```

untitled 1:1 python

functionOut returnOut

matrix



Append new samples

Sample Aggregator

samples

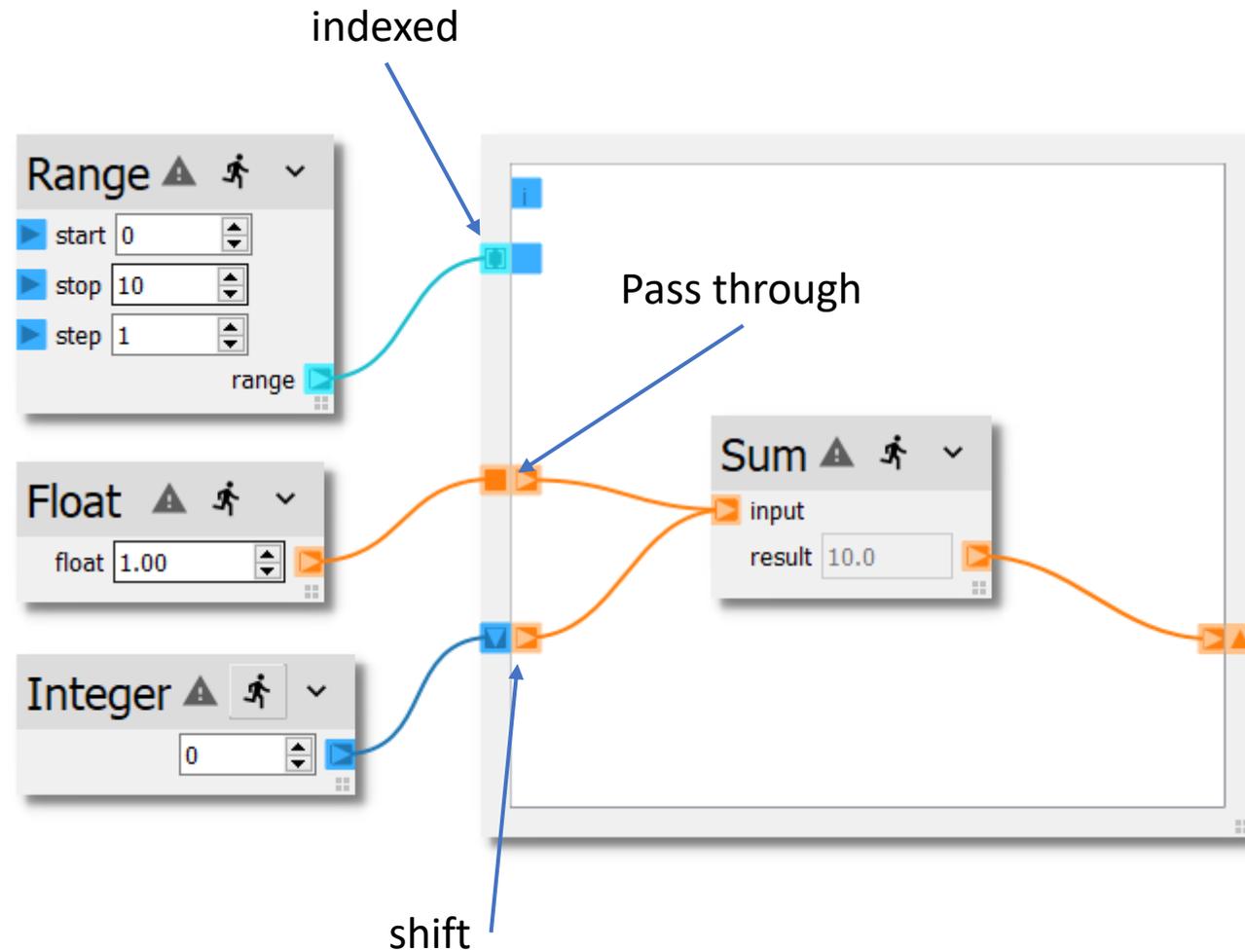
Remove duplicate samples

Reset

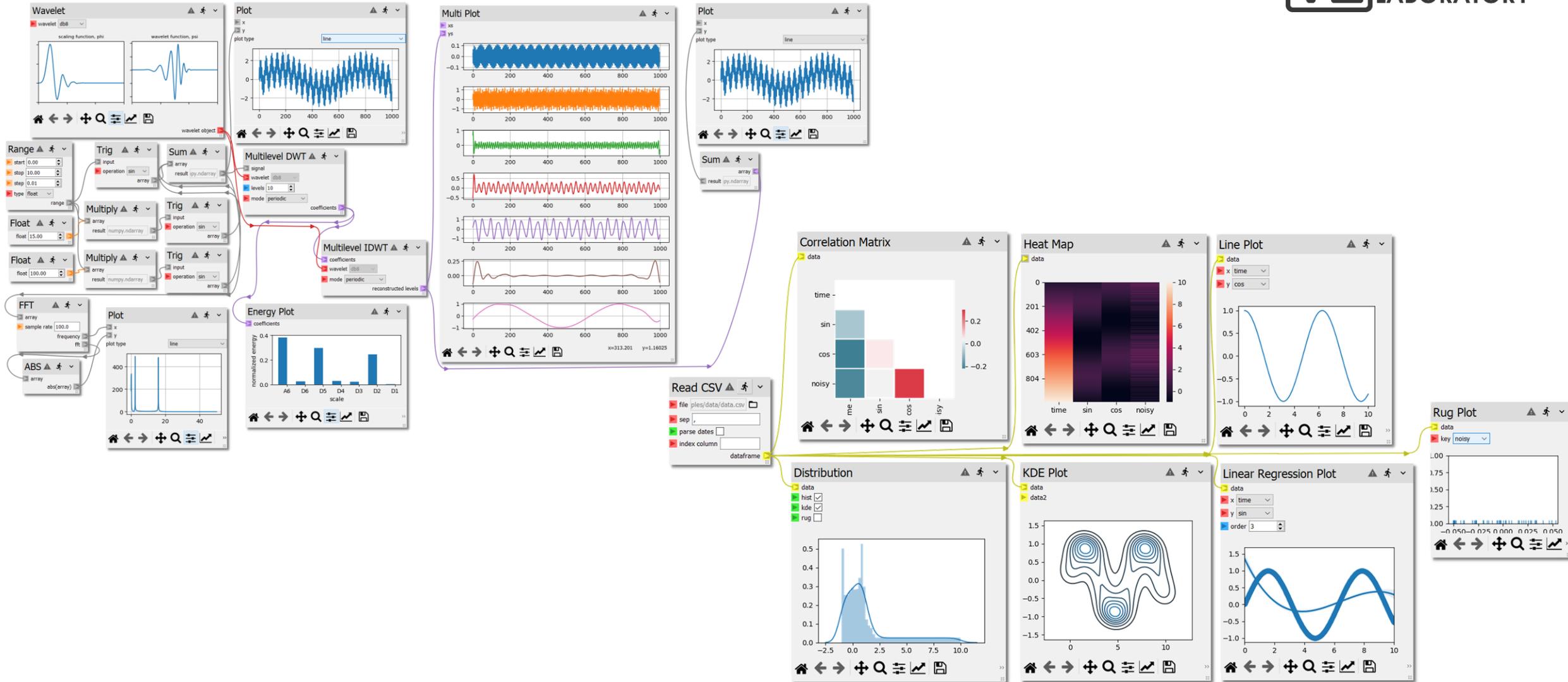
aggregated samples

Return "optimal" points for evaluation

Loops

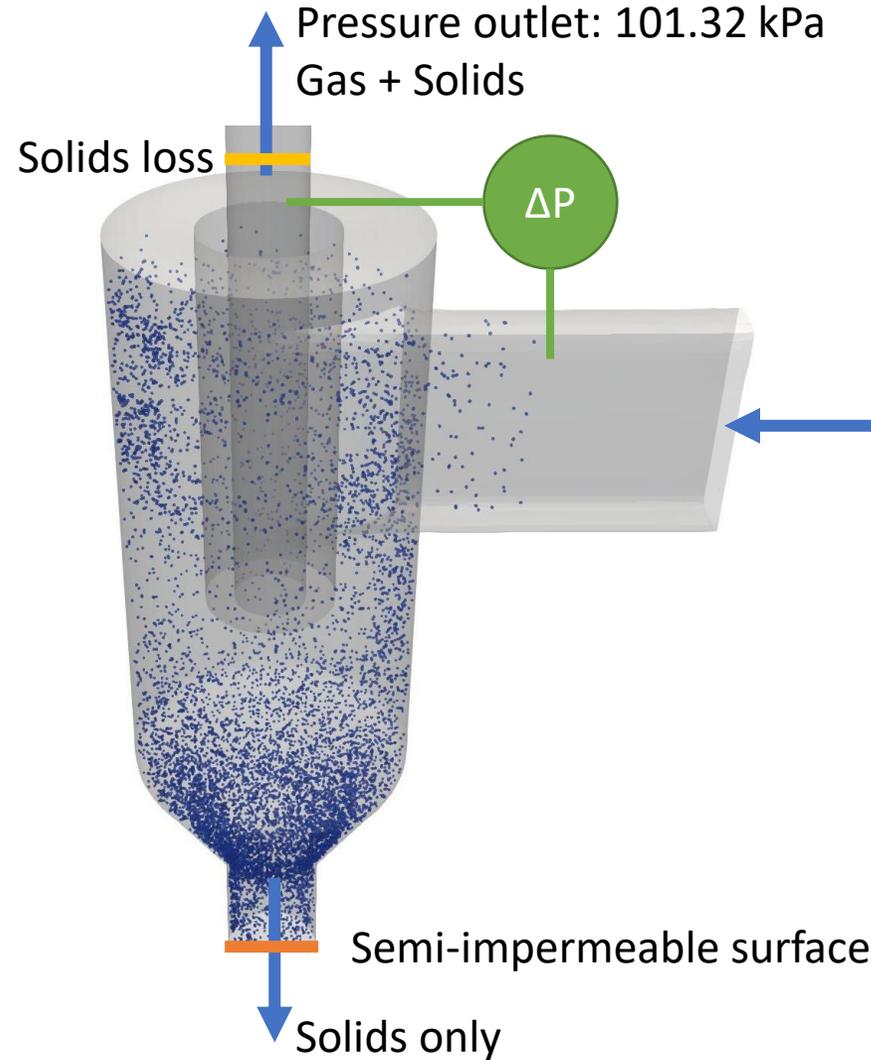
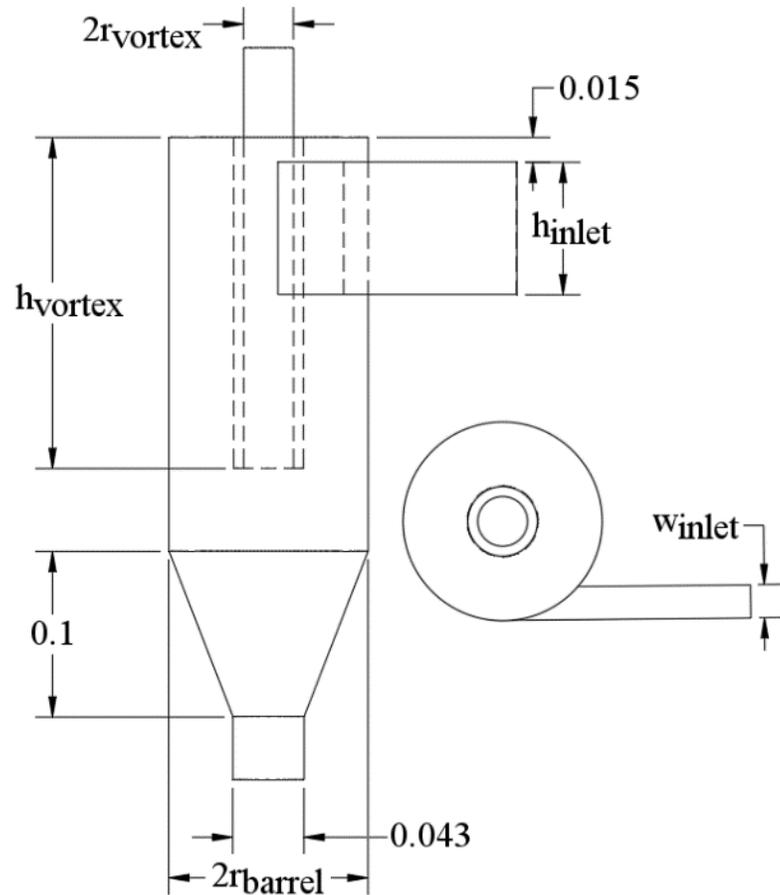


Other nodes



Examples

Example 1 | Cyclone Optimization



Cell size 5 x 5 x 5 mm, uniform

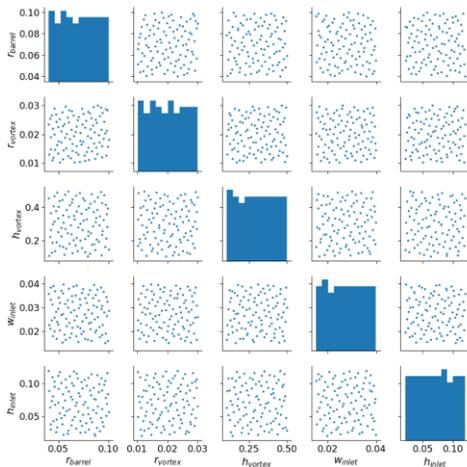
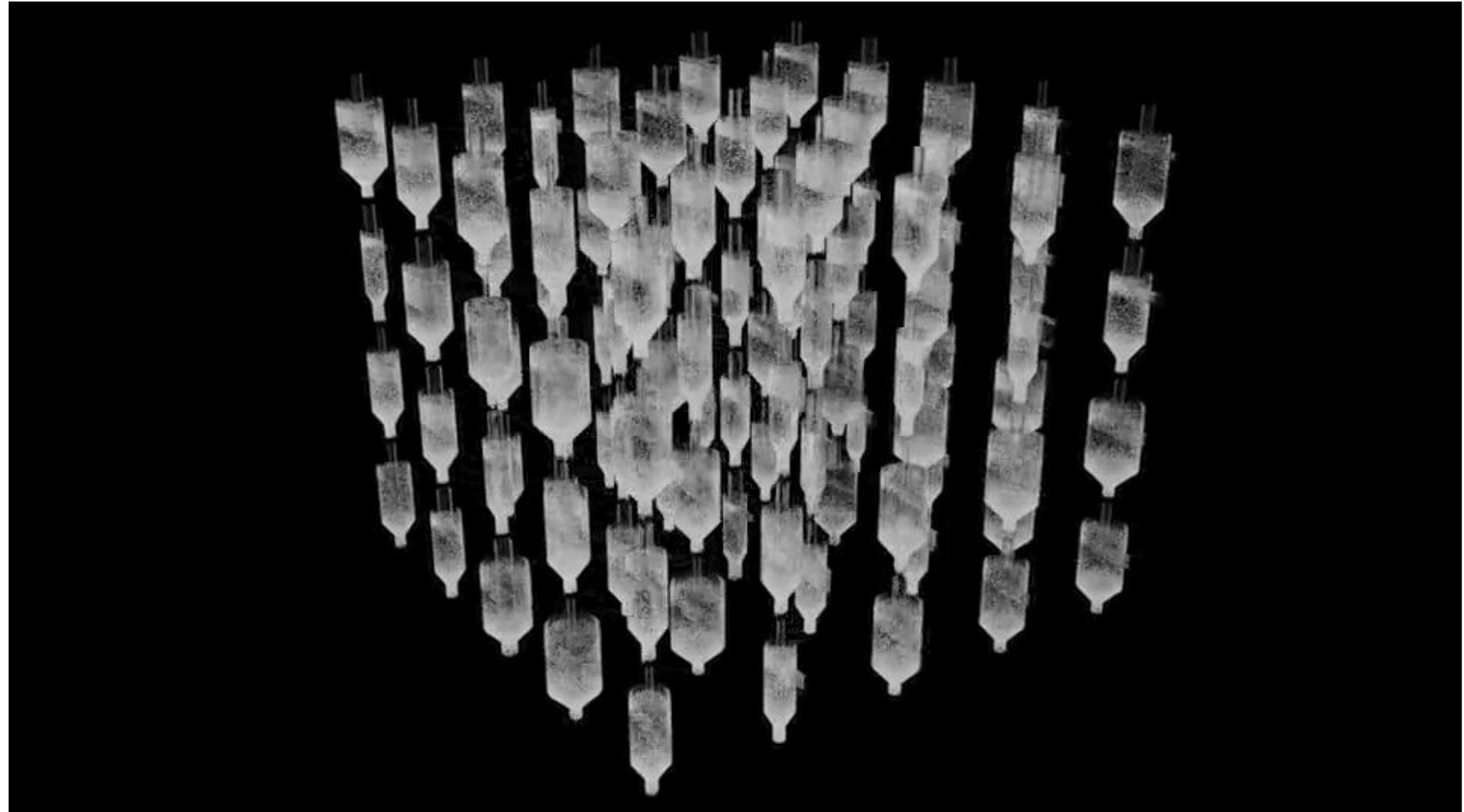
Gas 0.02 kg/s
Solids 0.08 kg/s

HDPE
Diameter: 871 μm
Density: 860 kg/m^3

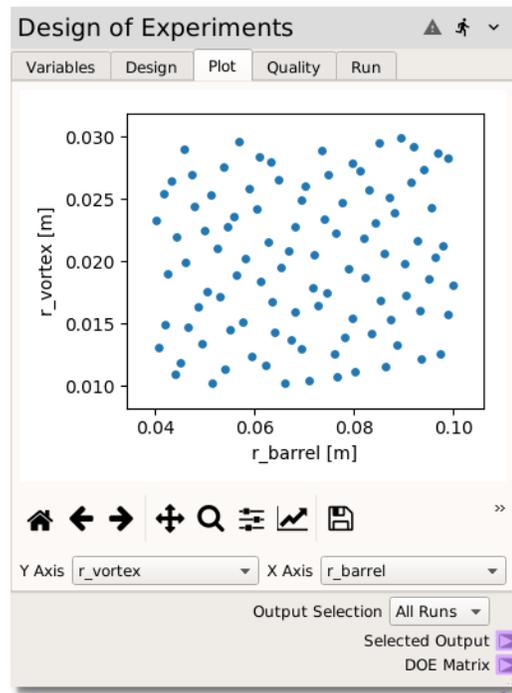
~1 particles/parcel

Example 1 | Run the models!

- All models ran simultaneously
- Took 21 minutes to 7 hours per model
- Cell count varied from 40,320 to 169,764
- Three models failed (6%), due to bad mesh



Example 1 | Workflow

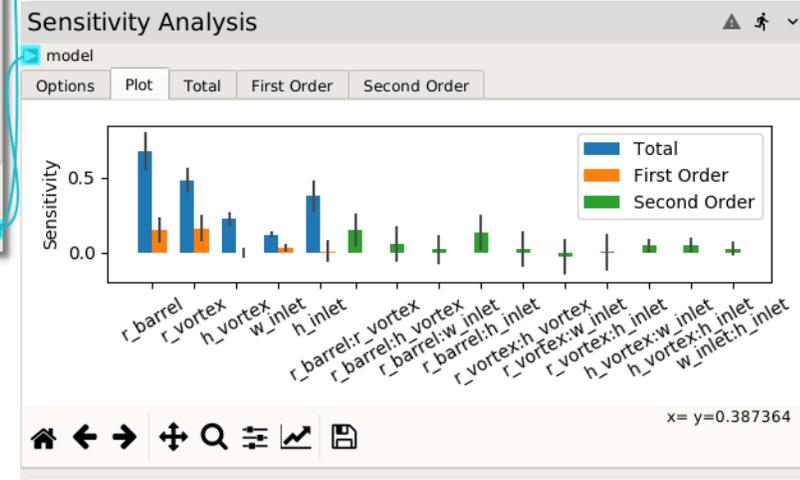
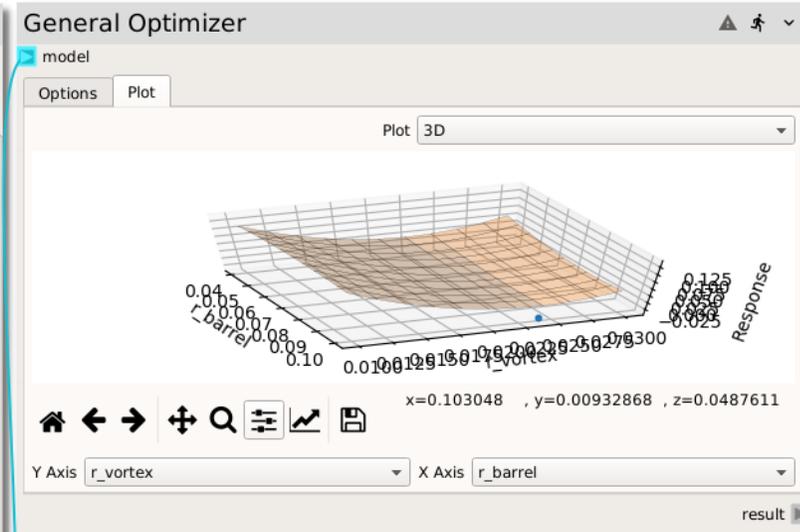
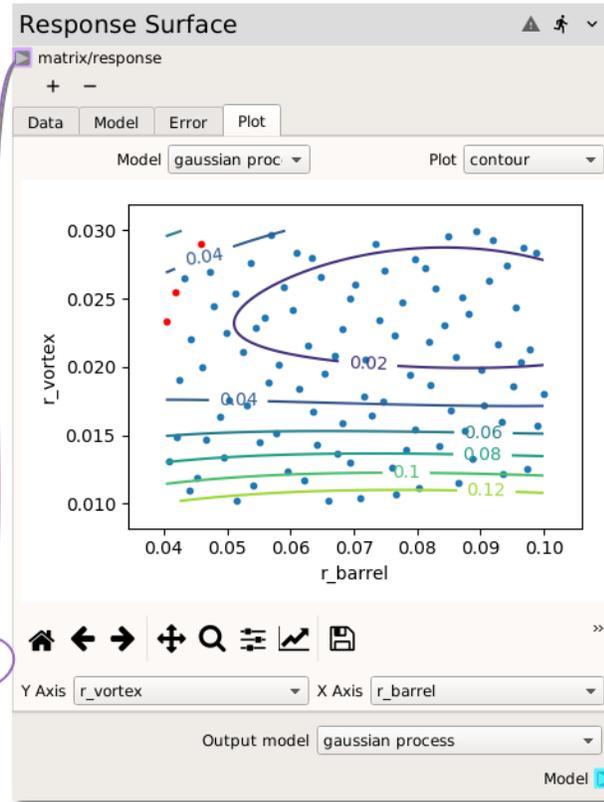


Code - Process output

```
arguments models  
function  
import glob  
import pandas as pd  
import os  
import numpy as np
```

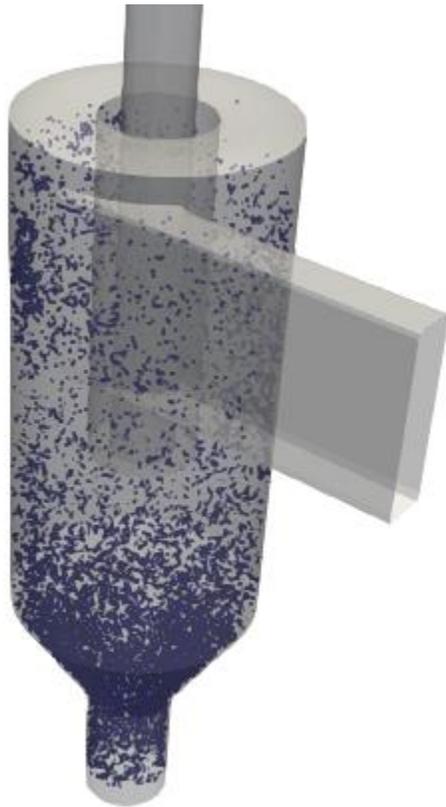
Save As | Load

functionOut: returnOut



Example 1 | Optimization

Original



Optimal



Using differential evolution

- 11 times lower pressure drop
- 2.3 times lower mass loss

Variable	Original (m)	Optimal (m)
r_{barrel}	0.06	0.096
r_{vortex}	0.015	0.026
h_{vortex}	0.4	0.373
h_{inlet}	0.08	0.12
w_{inlet}	0.02	0.015

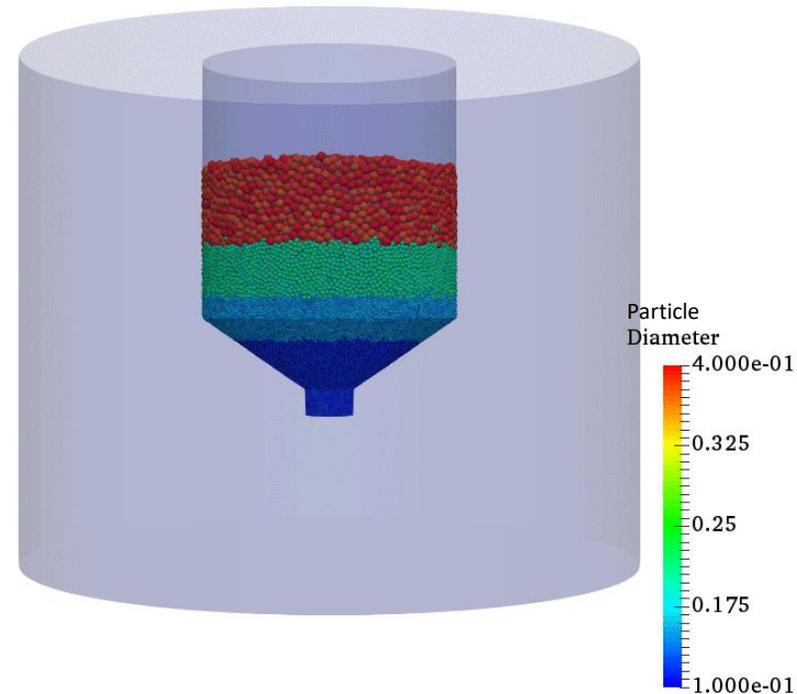
Edge of design space

Example 2 | Hopper Discharge Calibration

- Problem: Discharge of granular materials from a hopper.
- Frequently encountered setup in industrial settings.
- Typically design is based on empirical correlations, which doesn't necessarily always provide robust and efficient designs.
- Accurate modeling & simulation of granular material through Discrete Element Method (DEM) is critical for credible models.
- Use Nodeworks to perform model calibration (deterministic) for four modeling parameters in MFIx-DEM:

θ_1 : Particle-Particle Friction Coefficient θ_2 : Particle-Wall Friction Coefficient
 θ_3 : Particle-Particle Restitution Coefficient θ_4 : Particle-Wall Restitution Coefficient

Example visualization of hopper discharge modeled with MFIx-DEM



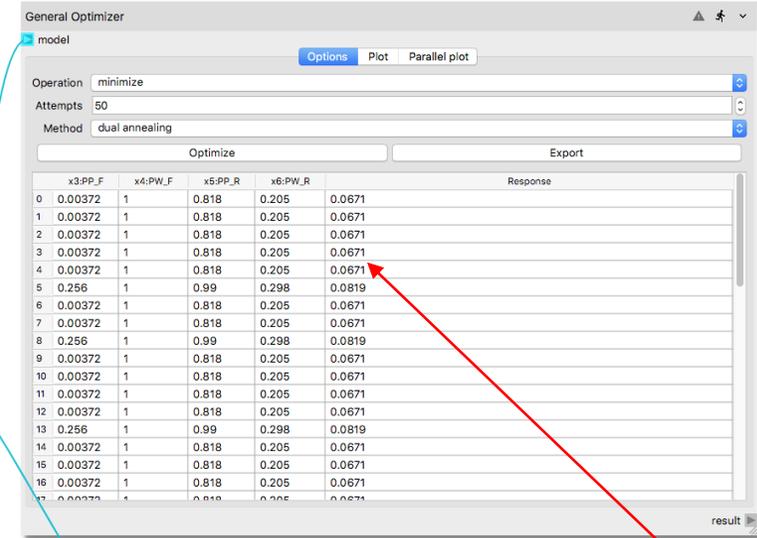
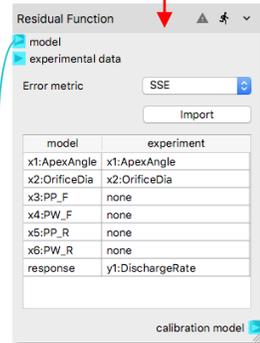
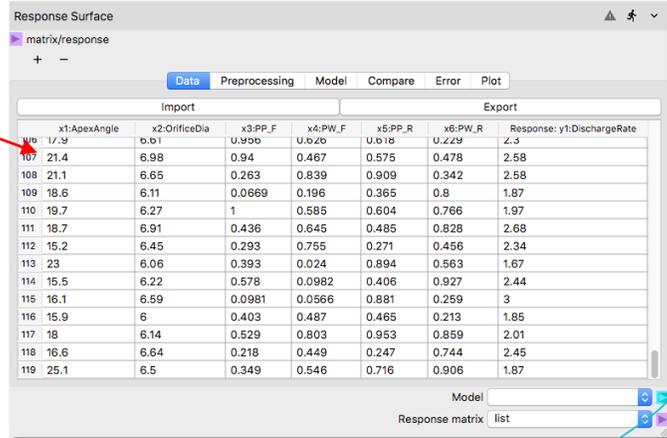
Source: Chen, Adep, Emady, Jiao, and Gel, "Enhancing the physical modeling capability of open-source MFIx-DEM Software for handling particle size polydispersity: Implementation and Validation" Powder Technology, 317 (2017) 117–125 doi: <http://dx.doi.org/10.1016/j.powtec.2017.04.055>

Example 2 | Hopper Discharge Calibration (cont'd)

Deterministic Calibration

imports experimental data and evaluates the response surface at each set of model/experimental parameters, calculating a residual metric

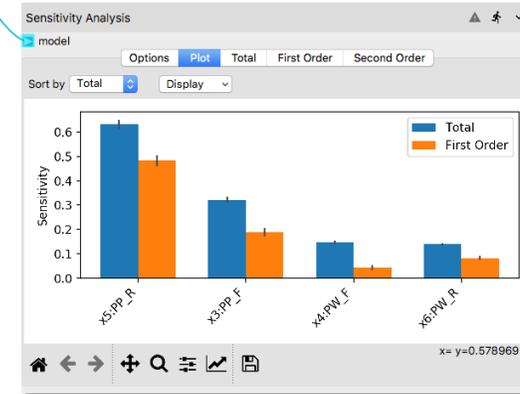
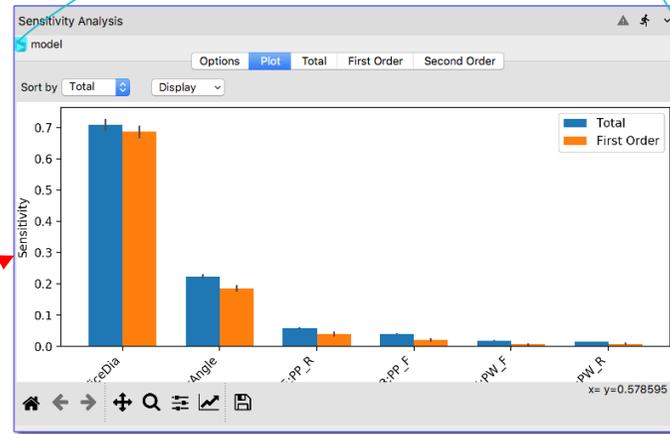
.CSV directly imported



120 sample simulation campaign performed externally in batch mode for 7 parameters:

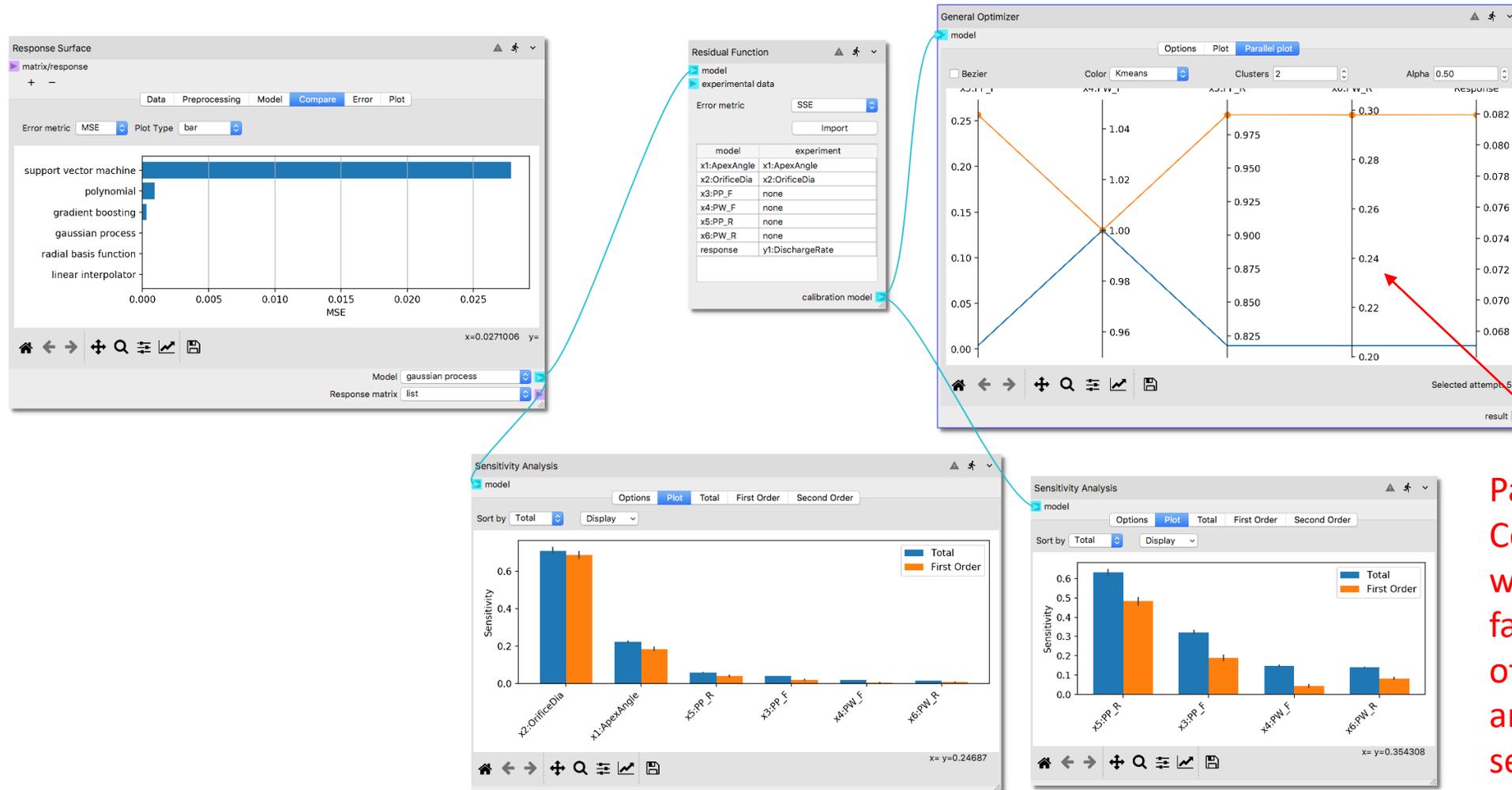
- x₁: Apex Angle
- x₂: Orifice Diameter
- x₃: θ₁: Particle-Particle Friction Coefficient
- x₄: θ₂: Particle-Wall Friction Coefficient
- x₅: θ₃: Particle-Particle Restitution Coefficient
- x₆: θ₄: Particle-Wall Restitution Coefficient

asses the sensitivity of the input parameters based on the simulation campaign results



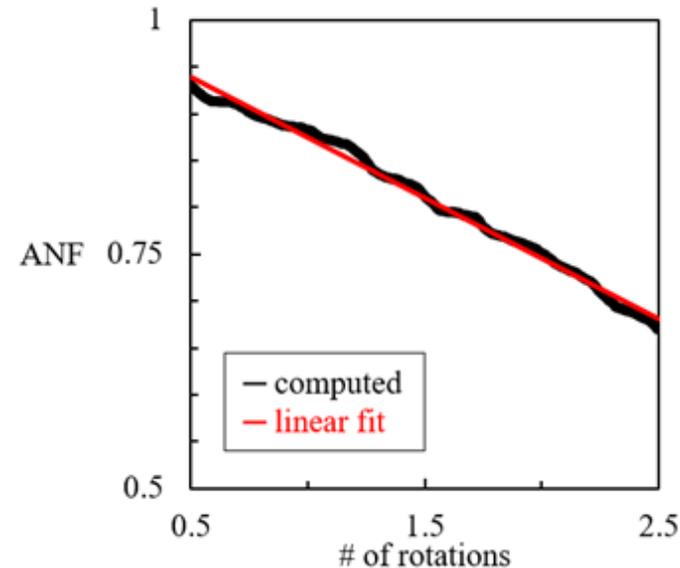
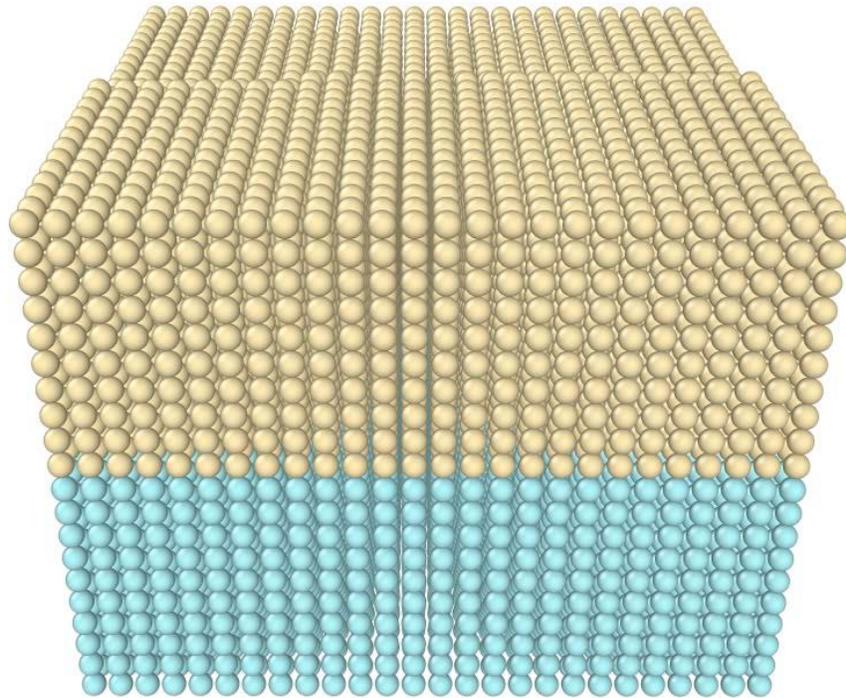
Minimize residual metric with multiple random starts, identifying optimal model inputs.

Example 2 | Hopper Discharge Calibration (cont'd)



Parallel Coordinates Plot, which enables faster identification of clustering around solution set.

Quantify mixing as the rate of decay of the **Alike Neighbor Fraction (ANF)**



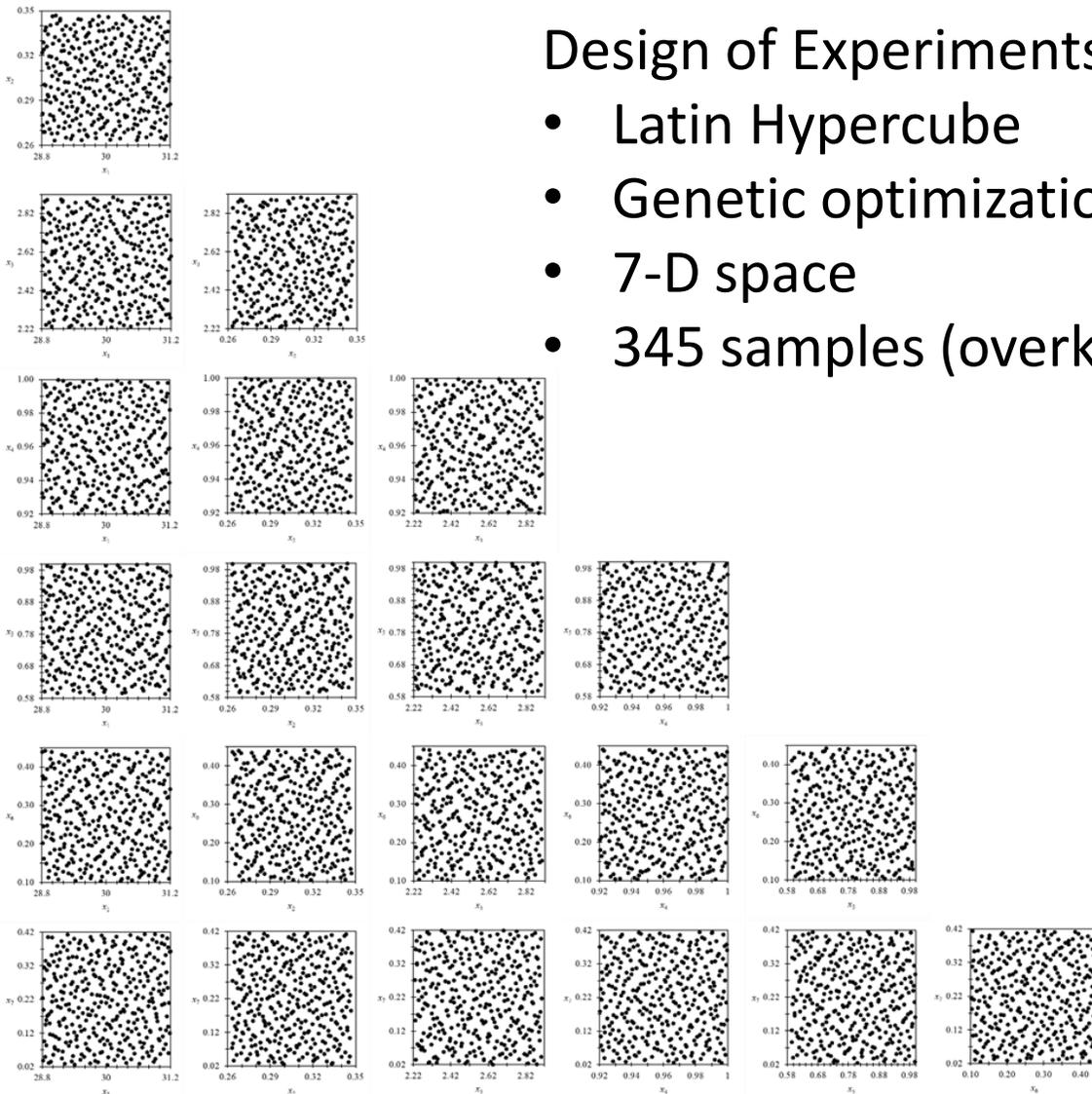
ANF = fraction of particles within $2.5r_p$ -radius of a given particle with the same color (averaged over all particles)

Source: Fullmer, W. D.; Dahl, S.; Weber, J. *Surrogate Modeling Approach to Uncertainty Quantification for a DEM Model of a Rotating Cubic Tumbler*; NETL-TRS-5-2019; NETL Technical Report Series; U.S. Department of Energy, National Energy Technology Laboratory: Morgantown, WV, 2019, p 24. DOI: 10.18141/1514272.

Example 3 | DOE → Simulations → Surrogate

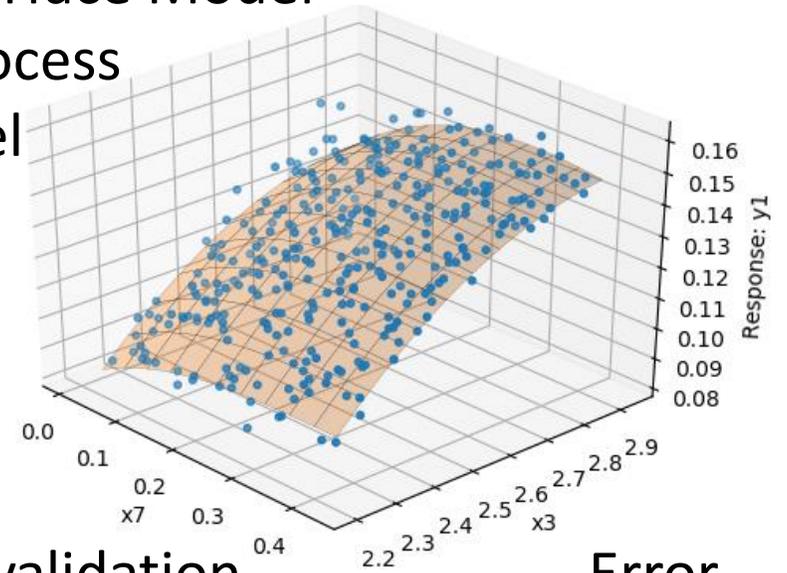
Design of Experiments

- Latin Hypercube
- Genetic optimization
- 7-D space
- 345 samples (overkill?)

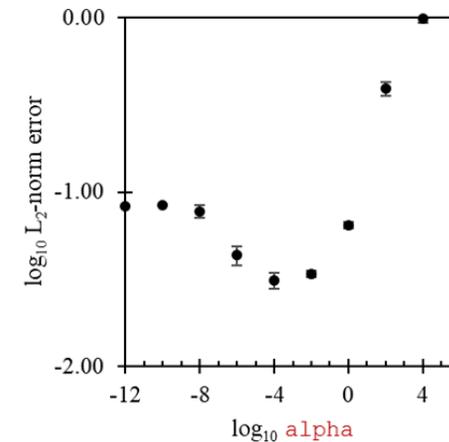


Response Surface Model Gaussian Process

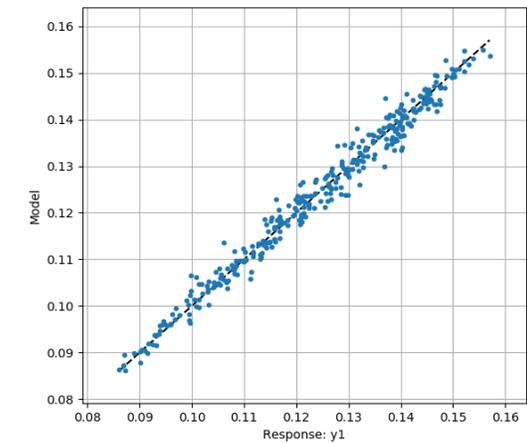
- RBF kernel



Cross-validation

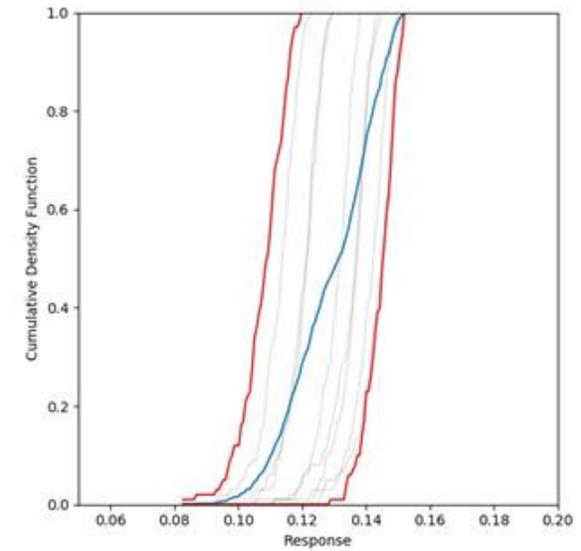
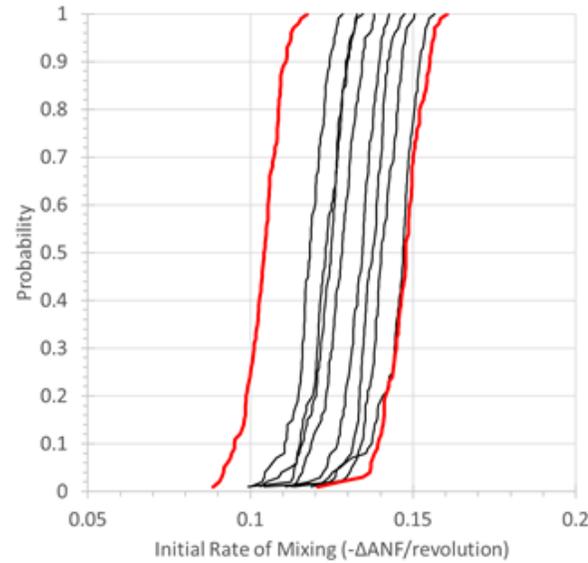
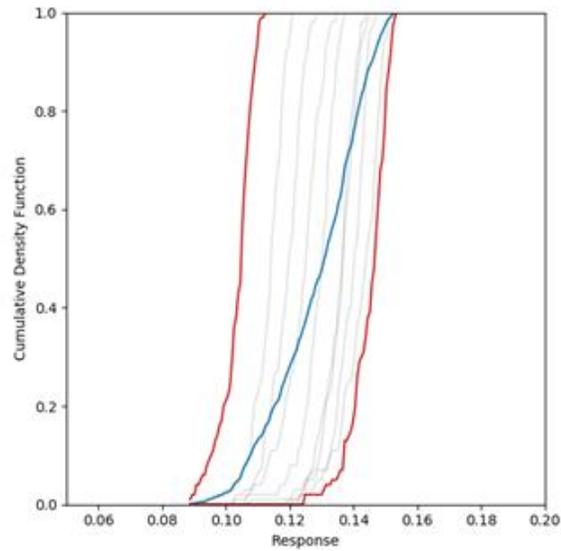


Error



Example 3 | Forward Propagation

Hybrid/nested sampling approach of Roy & Obekampf
10 epistemic samples, each with 100 aleatory samples

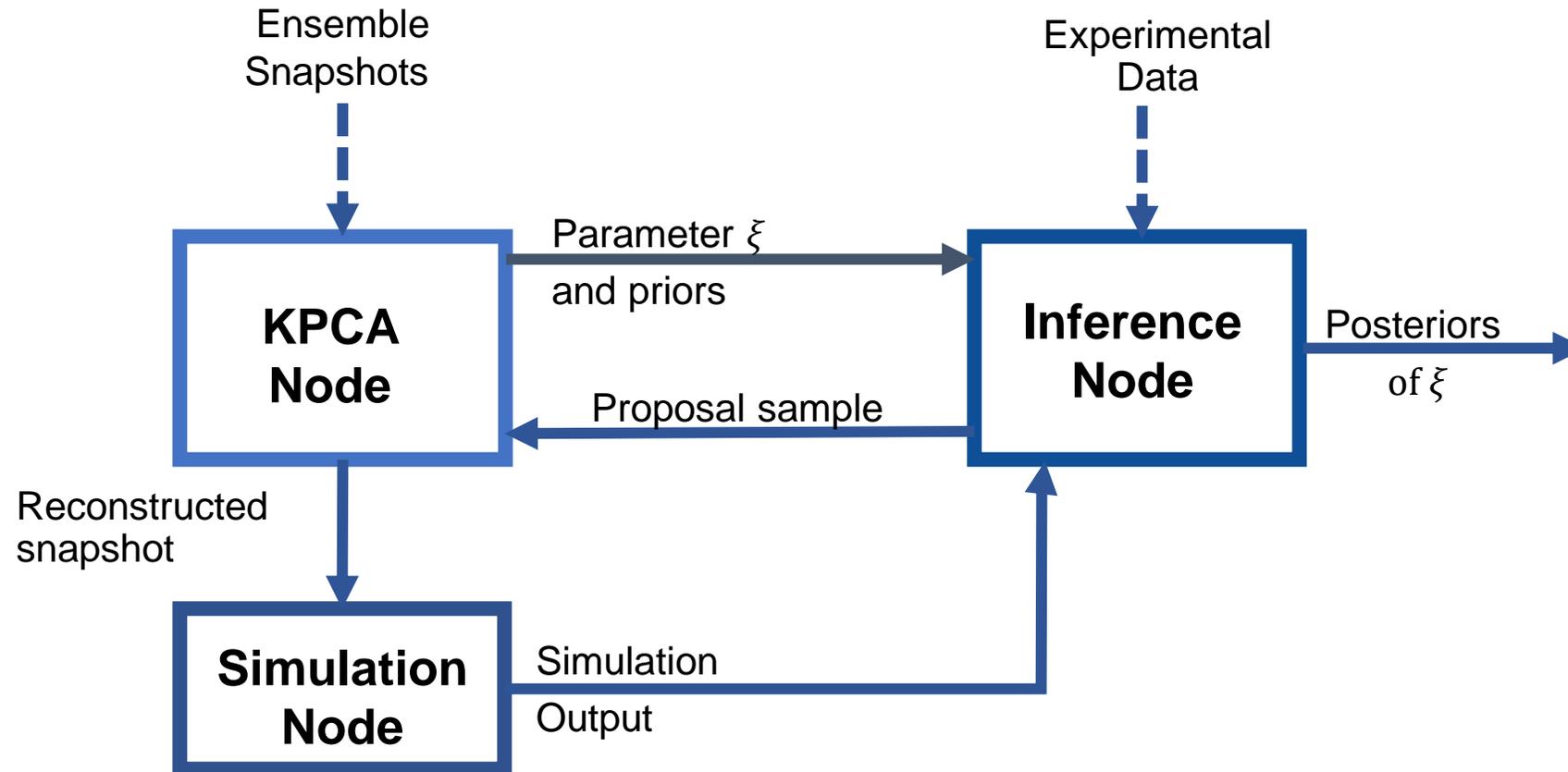


1000 CFD runs

Original direct sample p-box of Dahl et al (2019)

Examples of surrogate model propagated p-boxes

Example 4 | Stochastic Source Inversion (use case @LLNL)



- Simulation: linear elasticity in 2D
- Uncertain inputs: shear and Young's modulus (location-dependent: dimension=4050)
- Scenario: Given an observation strain tensor, recover the shear and Young's modulus
- Method: KPCA for dimension reduction + MCMC for inference

Example 4 | Stochastic Source Inversion

New customized nodes have been developed to encoding, inference, and decoding. Some of these tasks are performed by PSUADE behind the scenes though the interface.

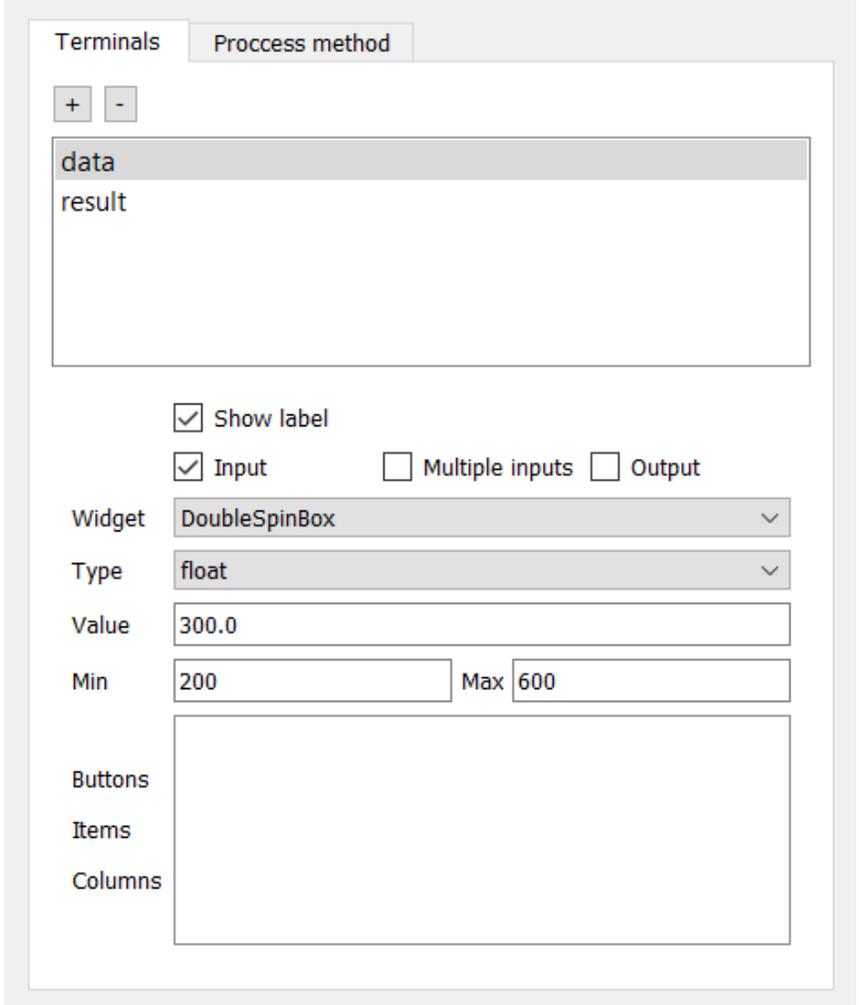
The screenshot shows the nodeworks interface with several nodes and data visualizations:

- Inference Node:** Contains a 'Simulation Result' section with tabs for 'Inputs', 'Experimental Data', 'Options', and 'Plot'. It has buttons for 'Import experiments' and 'Export data'. Below is a 'Likelihood options' section with radio buttons for 'Exp(-0.5 mse)' (selected) and 'Other'. At the bottom are buttons for 'Run MCMC', 'Terminate', and 'Save posterior'. On the right side, there are dropdown menus for 'Solution Snapshot Image', 'Proposal sample', and 'Posteriors'.
- KPCA Node:** Features a 'Feature Vector' section with tabs for 'Data', 'Model', and 'Plot'. It includes an 'Import Data' table with 5 columns (S1-S5) and 5 rows of data. Below the table are buttons for 'Create model', 'Save model', and 'Save reduced parameters'. On the right, there are dropdown menus for 'Reduced Parameters' and 'Reconstructed Vector'.
- Geocentric Node:** Contains a code editor with Python code for a function. It has buttons for 'Check input port' and 'Check Simulator'. On the right, there are dropdown menus for 'simResults' and 'simResultsImage'.
- True Snapshot Node:** Displays a heatmap labeled 'image' representing the true source distribution.
- MCMC-recovered Snapshot Node:** Displays a heatmap labeled 'image' representing the source distribution recovered by MCMC.

Arrows indicate data flow: from the 'Geocentric' node to the 'KPCA' node, and from the 'KPCA' node to both the 'True Snapshot' and 'MCMC-recovered Snapshot' nodes. A red arrow points from the 'Inference' node to the 'KPCA' node.

What is in the future?

- Better support for feedback loops
- Add more nodes for machine learning workflows
- Build a node creator
- Look into better dispatch tools
 - Cloud/local/HPC
- Better integration with other UQ tools
- Export workflows
- Automatic report generation



Terminals Process method

+ -

data
result

Show label
 Input Multiple inputs Output

Widget DoubleSpinBox

Type float

Value 300.0

Min 200 Max 600

Buttons
Items
Columns

Thanks!

Website

mfix.netl.doe.gov/nodeworks/

Questions?

The screenshot displays the Nodeworks software interface with several panels:

- Design of Experiments:** A table with two variables, 'a' and 'b', both set to a unit of -2.
- Code:** A Python script defining a function `f` and calculating the sum of a matrix.
- Neural Net Regressor:** A panel with settings for Optimizer (Adam), Learning rate (0.05), Loss function (MSE), Epochs (100), and Test (0.0). It includes a 'Train' button and a plot showing the loss function decreasing over time.
- General Optimizer:** A panel with settings for Operation (minimize), Attempts (1), Method (differential evolution), Strategy (best1bin), Iterations (100), Tolerance (1e-4), Popsiz (15), Mutation (0.50), and Recombination (0.70). It includes 'Optimize' and 'Export' buttons.

Arrows indicate data flow: from the 'DOE Matrix' output to the 'matrix' input of the Neural Net Regressor, and from the 'matrix/response' output to the 'model' input of the General Optimizer.

This work was funded by the Department of Energy, National Energy Technology Laboratory, an agency of the United States Government, through a support contract with Leidos Research Support Team (LRST). Neither the United States Government nor any agency thereof, nor any of their employees, nor LRST, nor any of their employees, makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.