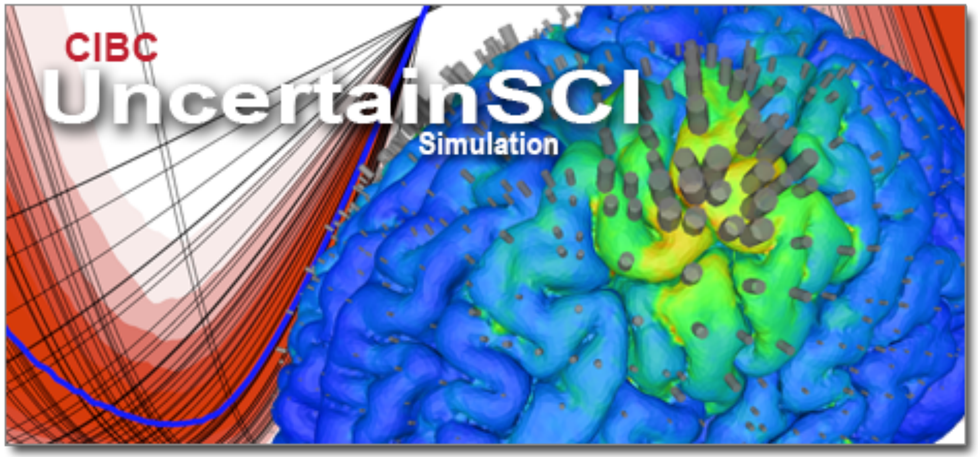
UncertainSCI: a Tool for Uncertainty Quantification in Brain Simulation



Jess D Tate, Zexin Liu, Jake A Bergquist, Sumientra Rampersad, Dan White, Chantel Charlebois, Lindsay C Rupp, Dana H Brooks, Akil Narayan, Rob S MacLeod

Scientific Computing and Imaging (SCI) Institute
University of Utah
Northeastern University

jess@sci.utah.edu

UncertainSCI Design Goals

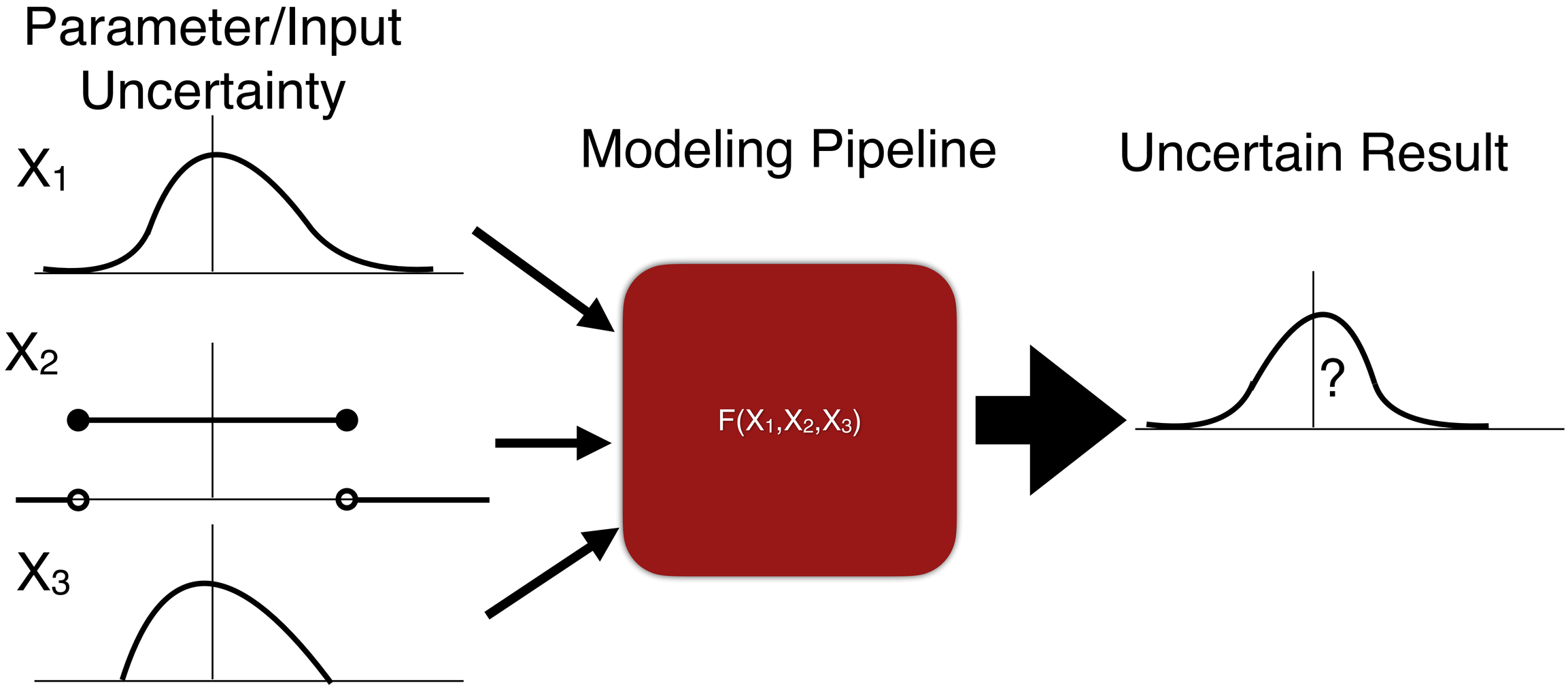
Numerical accuracy

Adaptability to multiple problem types

Interfacing with diverse tools

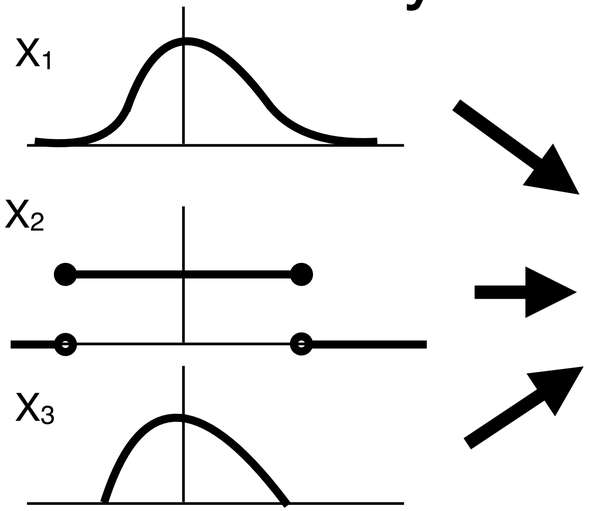
Simple API

Propagation of Uncertainty



UQ Pipeline

Parameter/Input Uncertainty



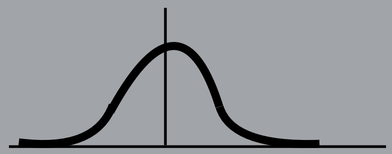
Weighted Sampling

Modeling Pipeline

$F(X_1,X_2,X_3)$

UQ Estimation

Uncertainty Statistics



Mean
Median
Stand. Dev.
Quantiles
Sensitivities

UncertainSCI

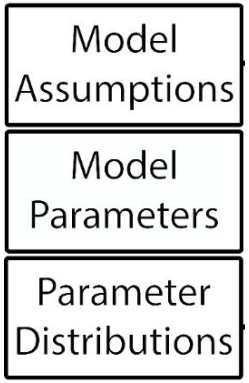
Non-invasive

UncertainSCI and basic usage

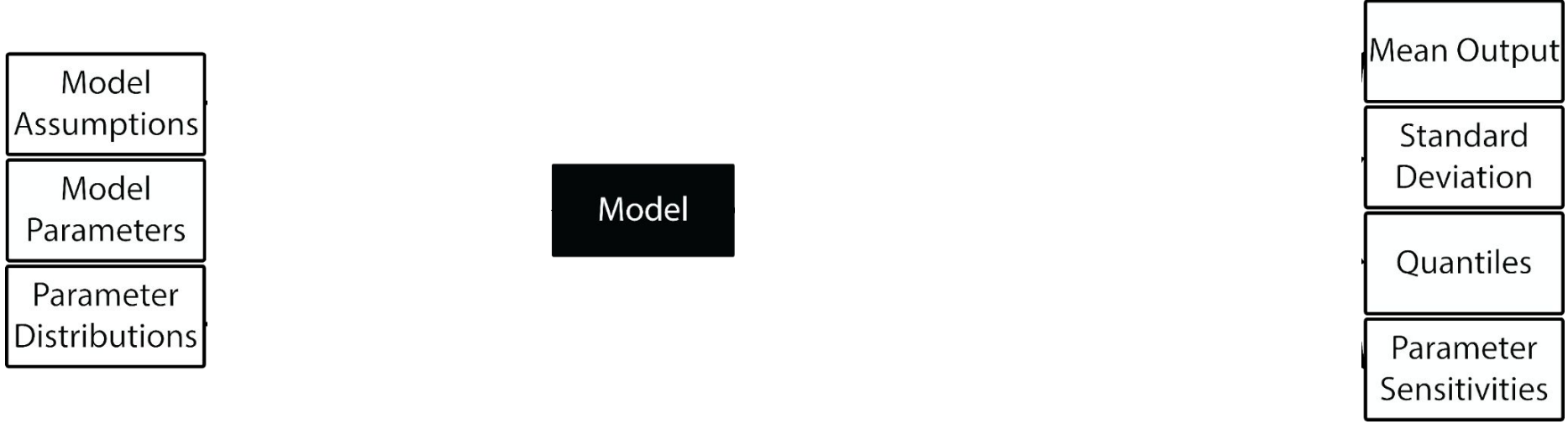

Model

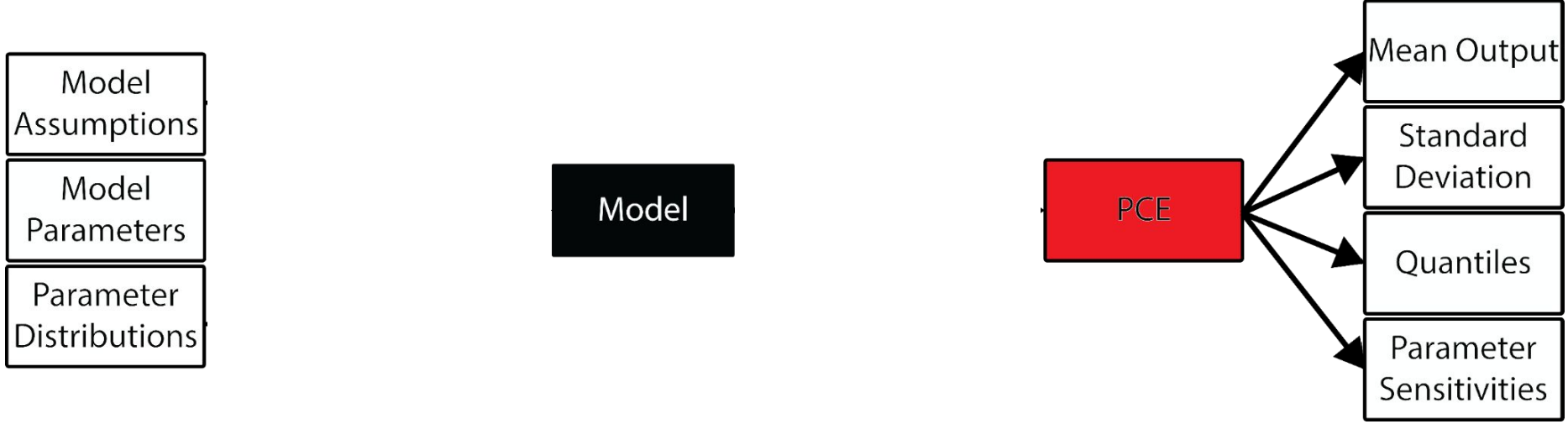

Model
Parameters
Model

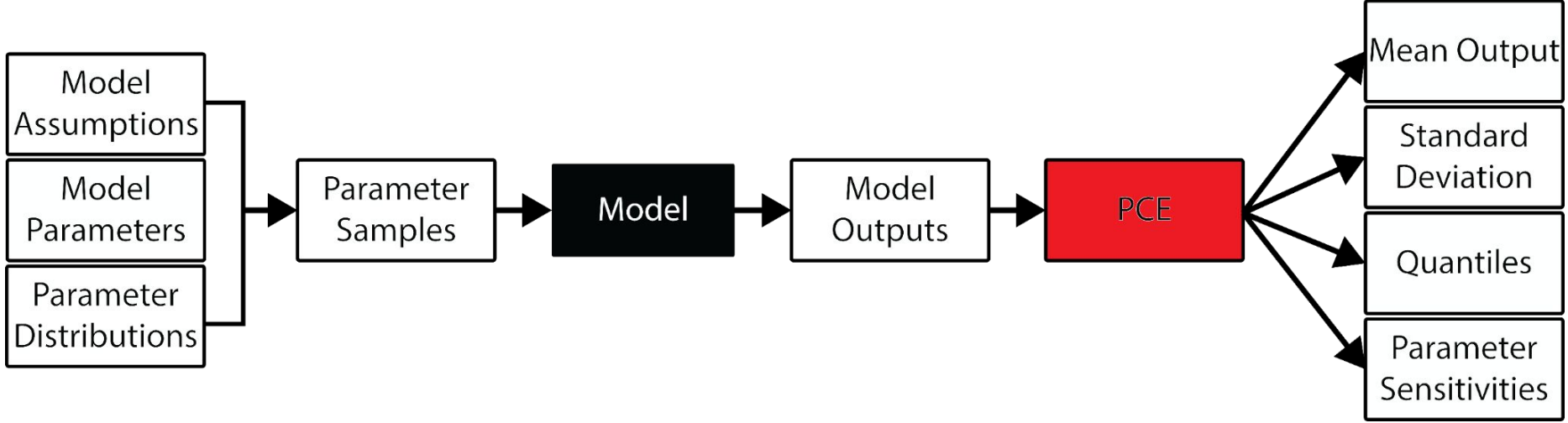

Model
Assumptions
Model
Parameters
Model

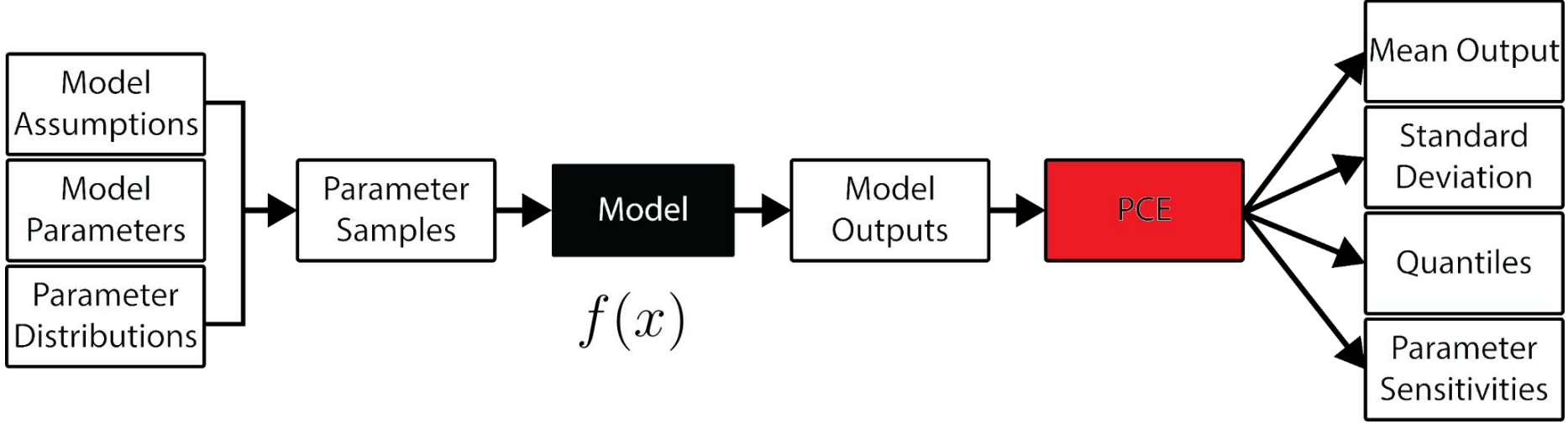

Model Assumptions
Model Parameters
Parameter Distributions

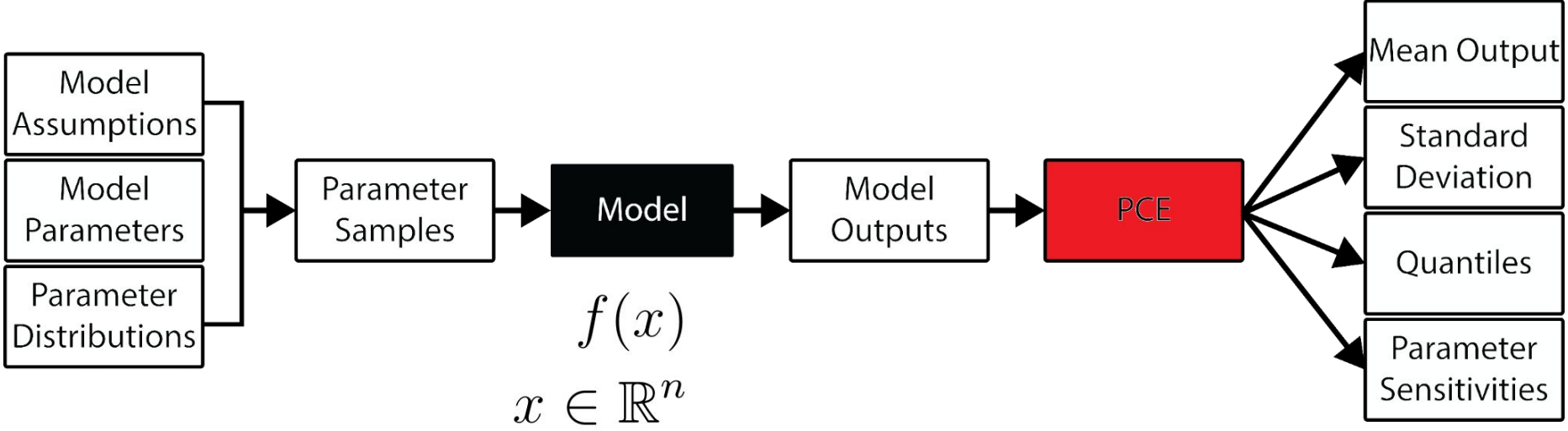
Model

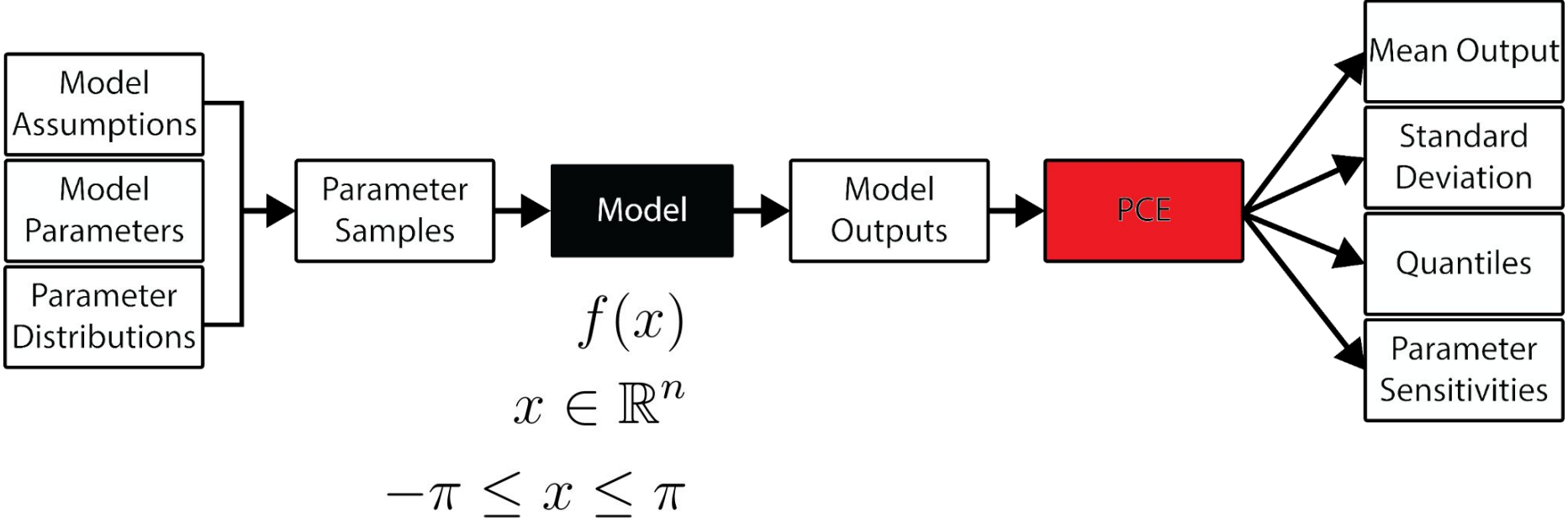

Model Assumptions
Model Parameters
Parameter Distributions
Model
Mean Output
Standard Deviation
Quantiles
Parameter Sensitivities


Model Assumptions
Model Parameters
Parameter Distributions
Model
PCE
Mean Output
Standard Deviation
Quantiles
Parameter Sensitivities


Model Assumptions
Model Parameters
Parameter Distributions
Parameter Samples
Model
Model Outputs
PCE
Mean Output
Standard Deviation
Quantiles
Parameter Sensitivities


Model Assumptions
Model Parameters
Parameter Distributions
Parameter Samples
Model
$f(x)$
Model Outputs
PCE
Mean Output
Standard Deviation
Quantiles
Parameter Sensitivities

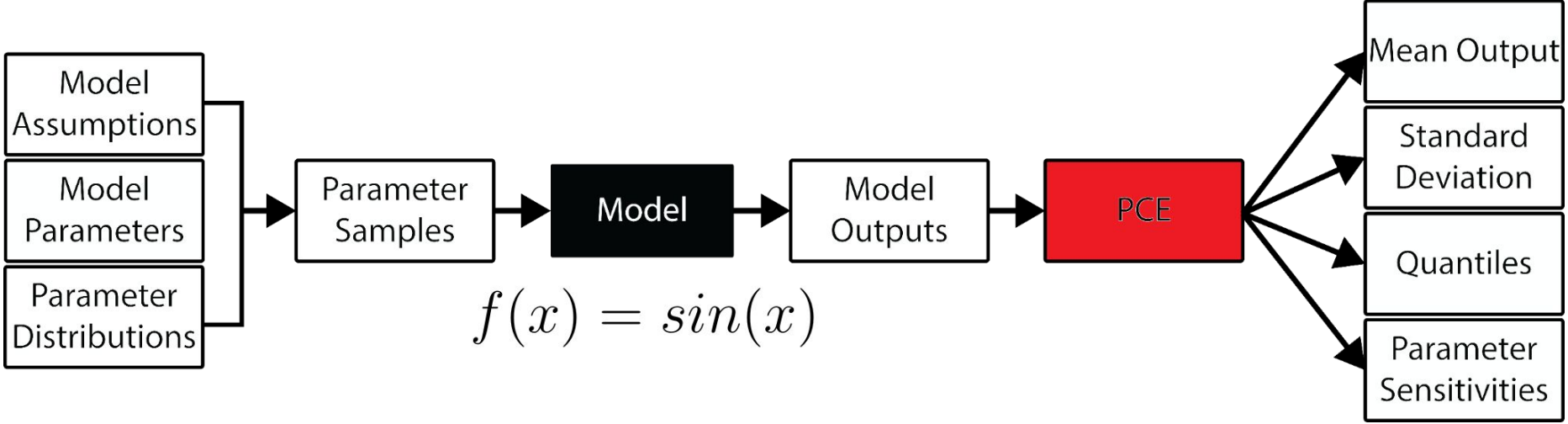

Model Assumptions
Model Parameters
Parameter Distributions
Parameter Samples
Model
Model Outputs
PCE
Mean Output
Standard Deviation
Quantiles
Parameter Sensitivities
$f(x)$
$x \in \mathbb{R}^n$

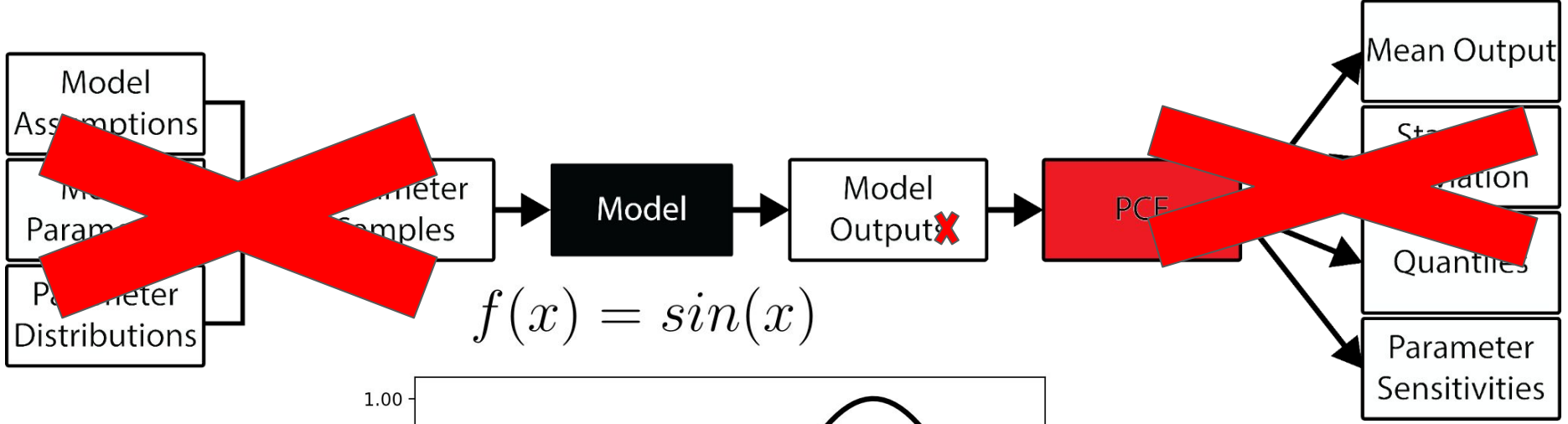


$$f(x)$$

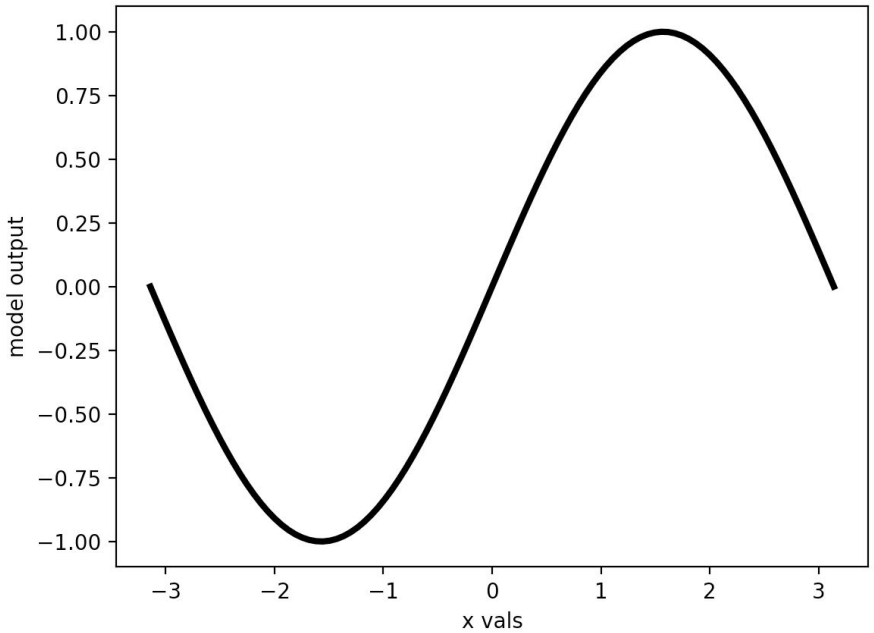
$$x \in \mathbb{R}^n$$

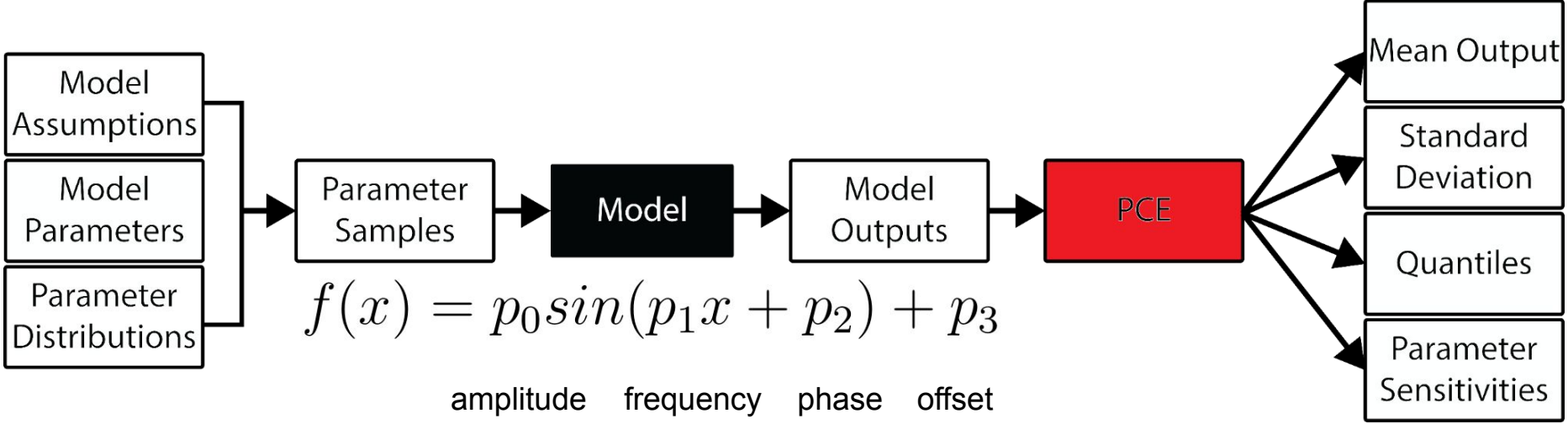
$$-\pi \leq x \leq \pi$$

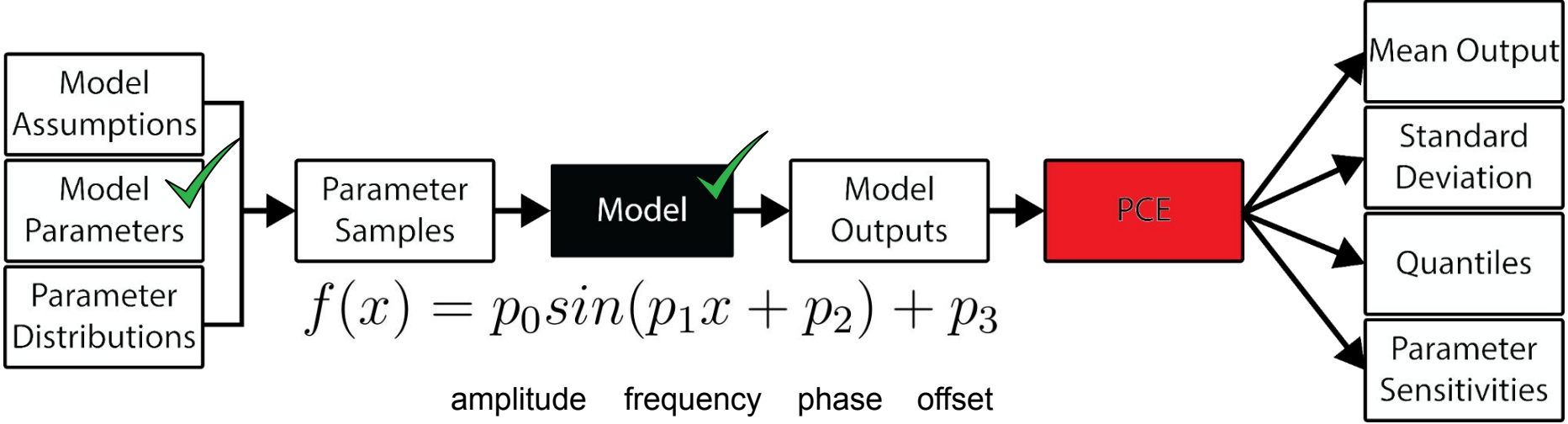

Model Assumptions
Model Parameters
Parameter Distributions
Parameter Samples
Model
Model Outputs
PCE
Mean Output
Standard Deviation
Quantiles
Parameter Sensitivities
$f(x) = sin(x)$

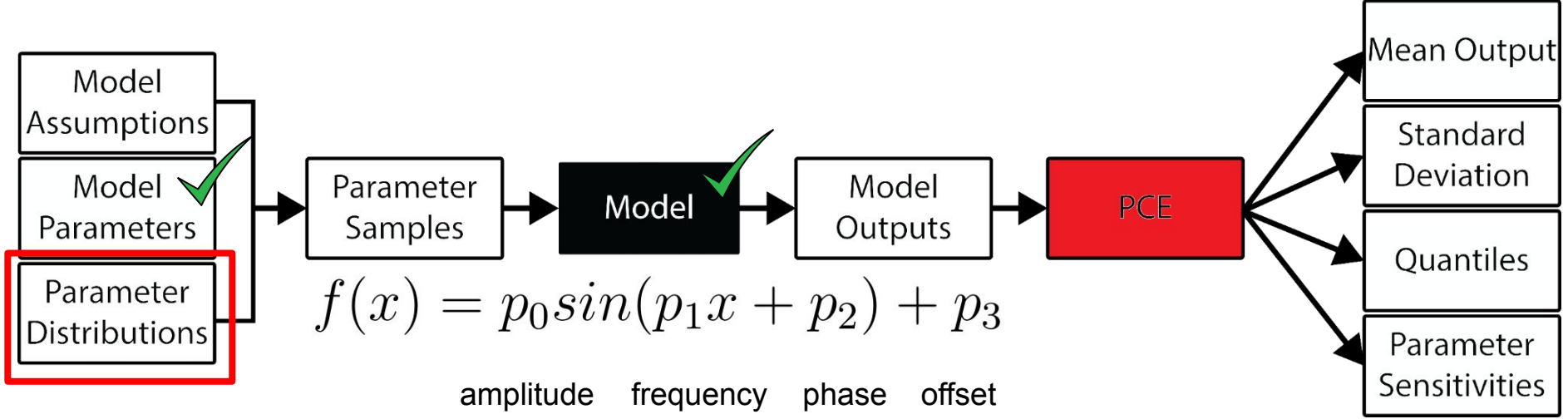

Model
Assumptions
Parameter
Distributions
Model
Model
Output
PCE
Mean Output
Quantiles
Parameter
Sensitivities

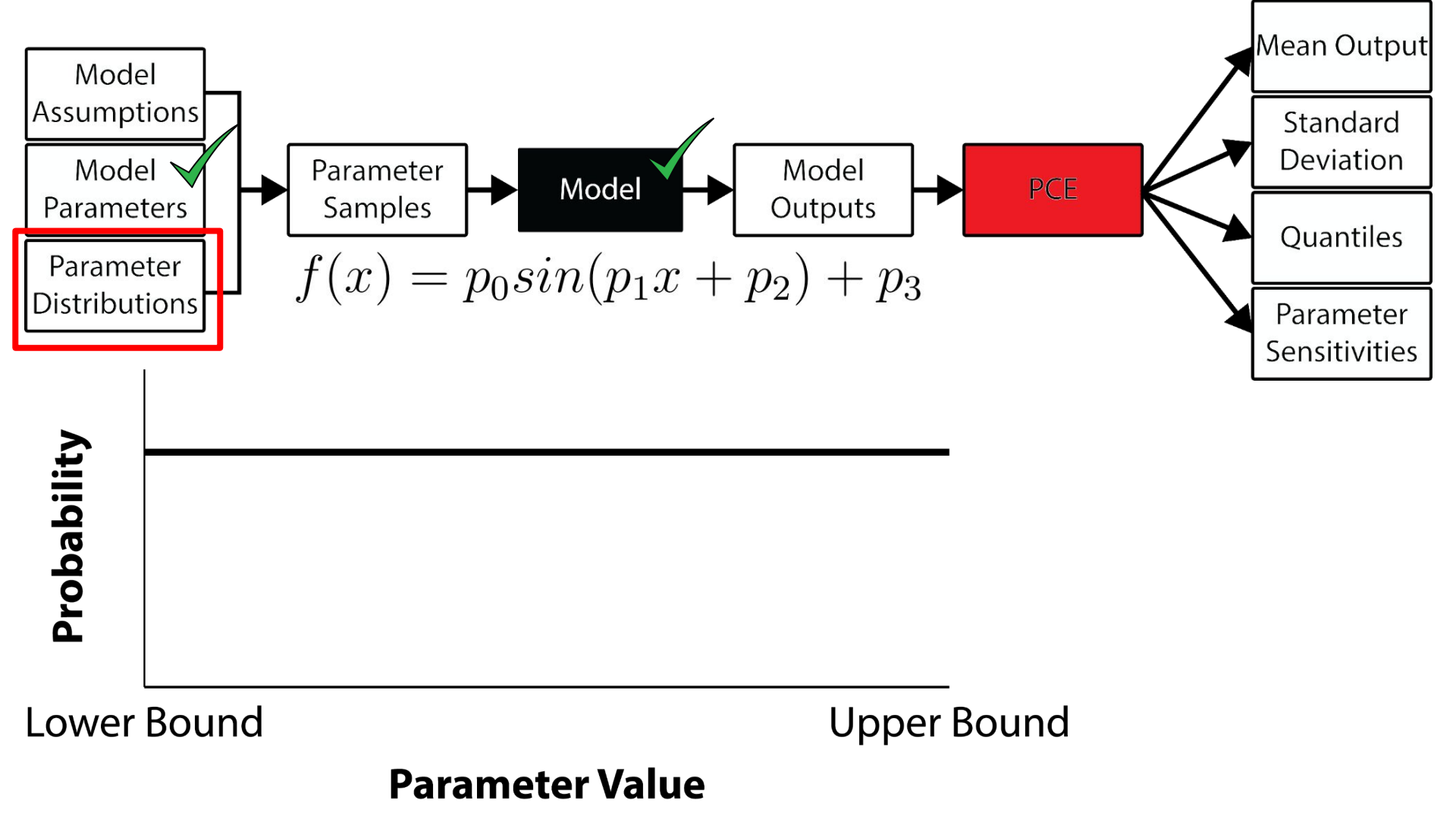
$$f(x) = sin(x)$$

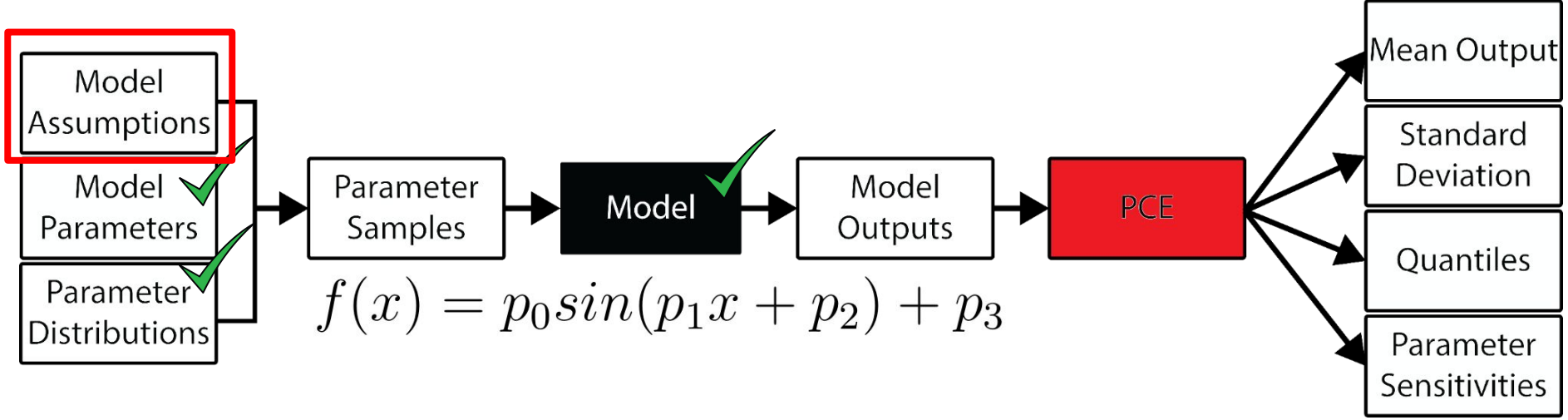

model output
x vals
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−3
−2
−1
0
1
2
3

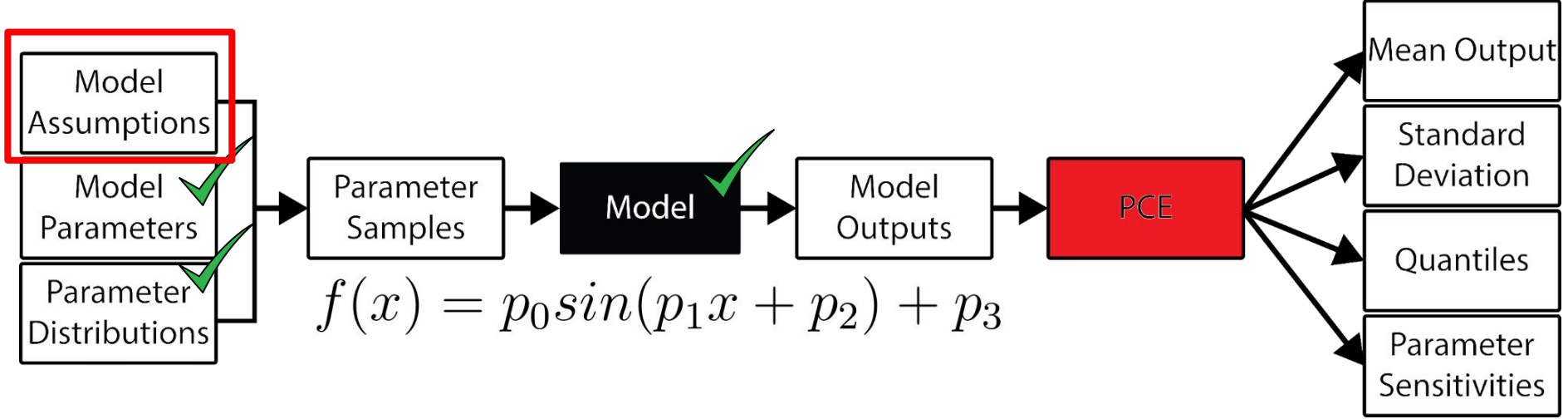

Model Assumptions
Model Parameters
Parameter Distributions
Parameter Samples
Model
Model Outputs
PCE
Mean Output
Standard Deviation
Quantiles
Parameter Sensitivities
$f(x) = p_0 sin(p_1 x + p_2) + p_3$
amplitude
frequency
phase
offset


Model Assumptions
Model Parameters
Parameter Distributions
Parameter Samples
Model
Model Outputs
PCE
Mean Output
Standard Deviation
Quantiles
Parameter Sensitivities
$f(x) = p_0 sin(p_1 x + p_2) + p_3$
amplitude
frequency
phase
offset


Model Assumptions
Model Parameters
Parameter Distributions
Parameter Samples
Model
Model Outputs
PCE
Mean Output
Standard Deviation
Quantiles
Parameter Sensitivities
$f(x) = p_0 sin(p_1 x + p_2) + p_3$
amplitude
frequency
phase
offset


Model Assumptions
Model Parameters
Parameter Distributions
Parameter Samples
Model
Model Outputs
PCE
Mean Output
Standard Deviation
Quantiles
Parameter Sensitivities
$f(x) = p_0 sin(p_1 x + p_2) + p_3$
Probability
Lower Bound
Upper Bound
Parameter Value


Model Assumptions
Model Parameters
Parameter Distributions
Parameter Samples
Model
Model Outputs
PCE
Mean Output
Standard Deviation
Quantiles
Parameter Sensitivities
$f(x) = p_0 sin(p_1 x + p_2) + p_3$

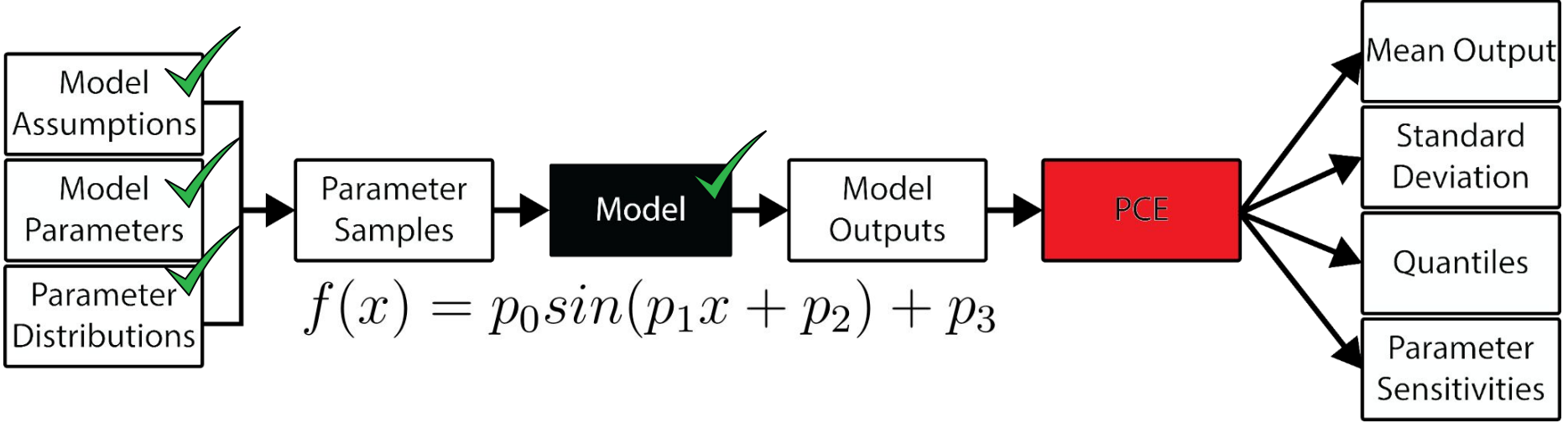


$$0.5 \leq p_0 \leq 1$$

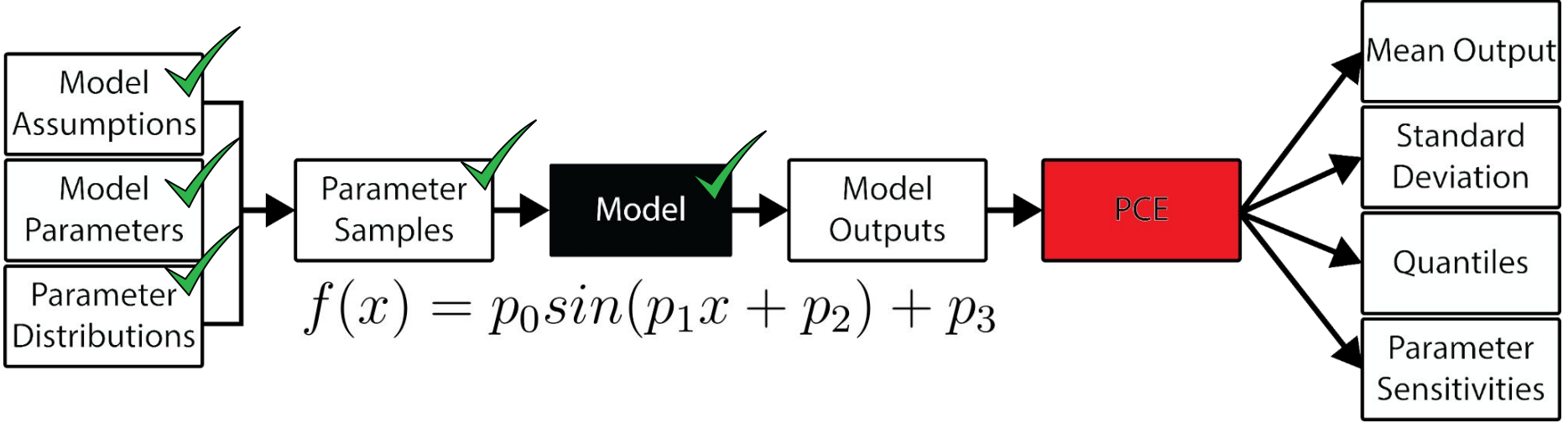
$$1 \leq p_1 \leq 1$$

$$0 \leq p_2 \leq 0$$

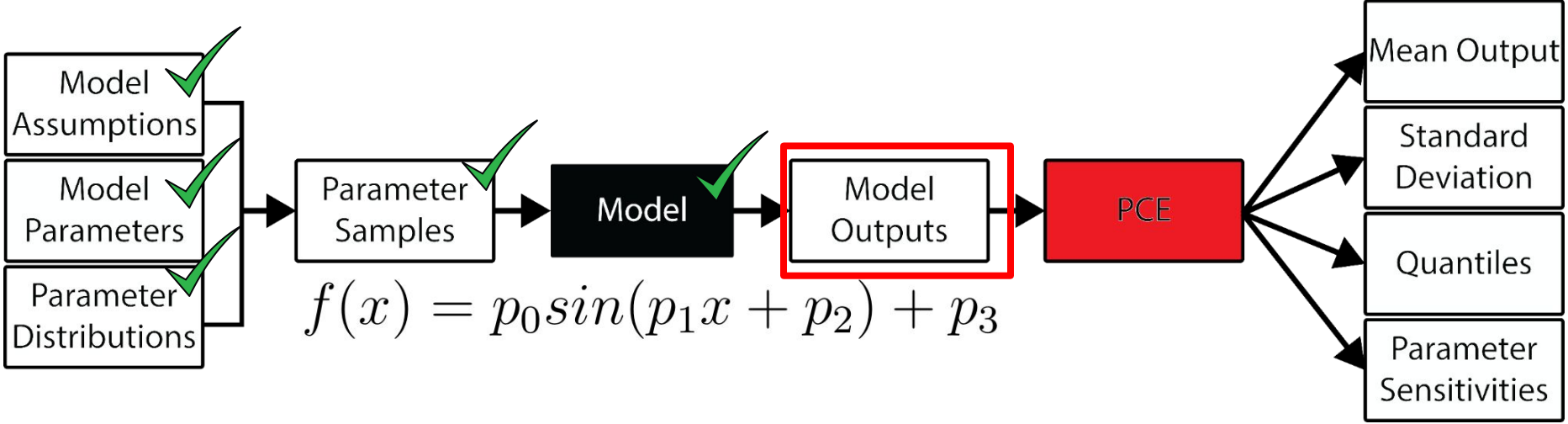
$$-0.1 \leq p_3 \leq 0.1$$



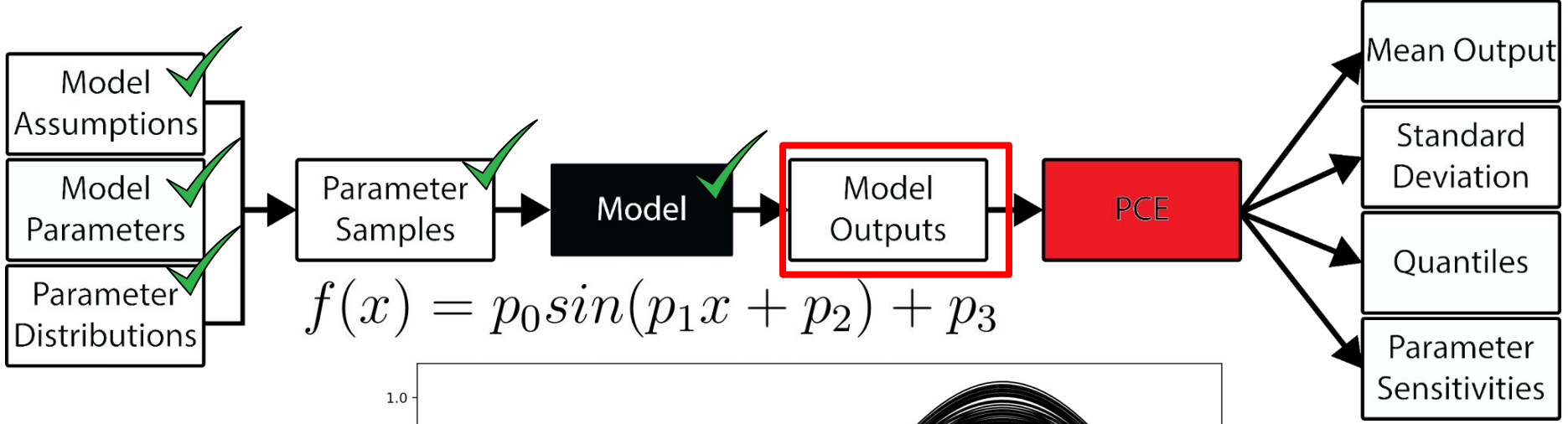
$p_0 \in [0.5, 1], p_3 \in [0 - 0.1, 0.1]$



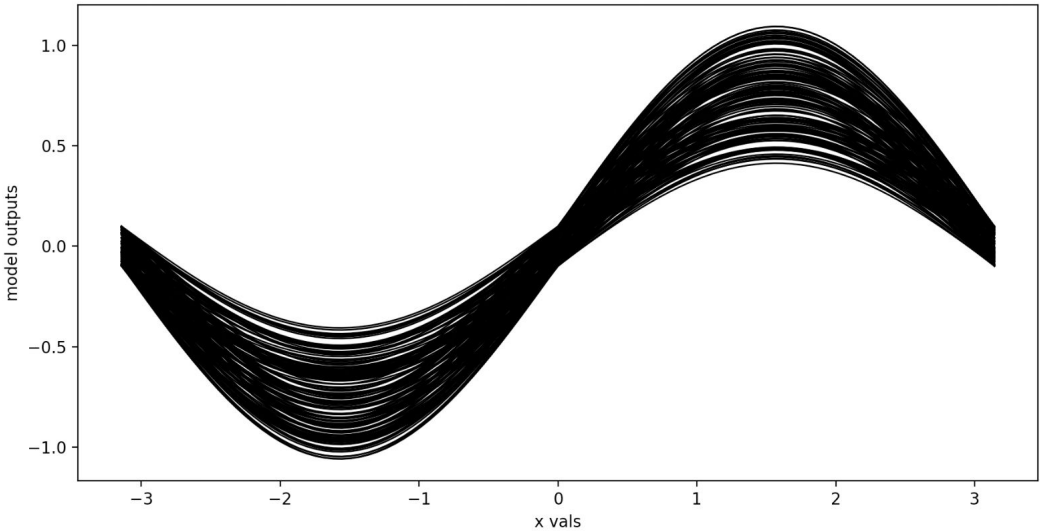
$p_0 \in [0.5, 1], p_3 \in [0 - 0.1, 0.1]$



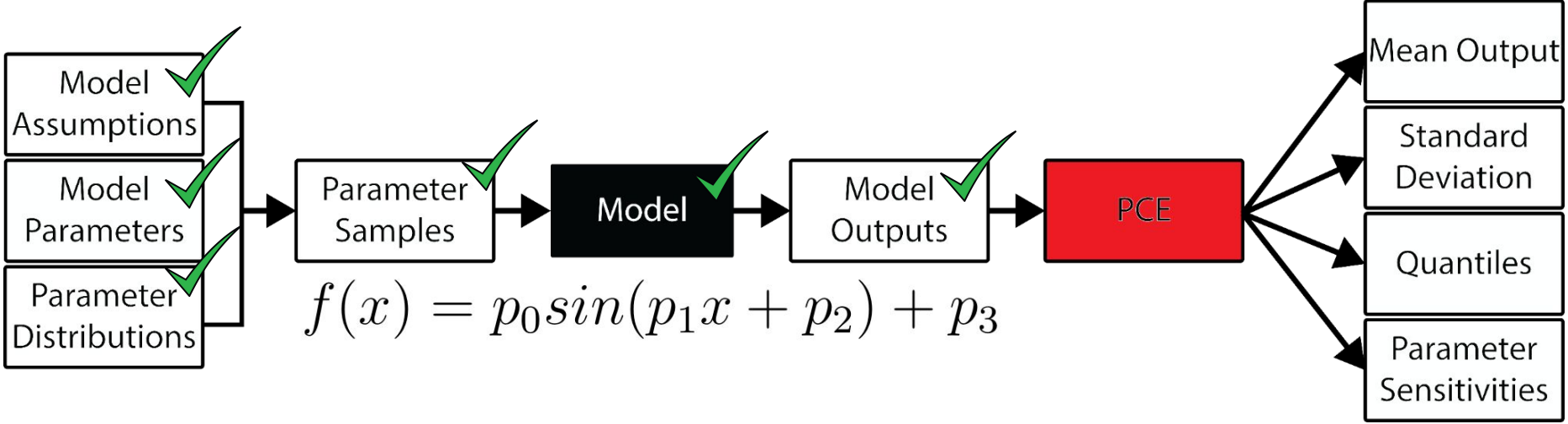
$p_0 \in [0.5, 1], p_3 \in [0 - 0.1, 0.1]$



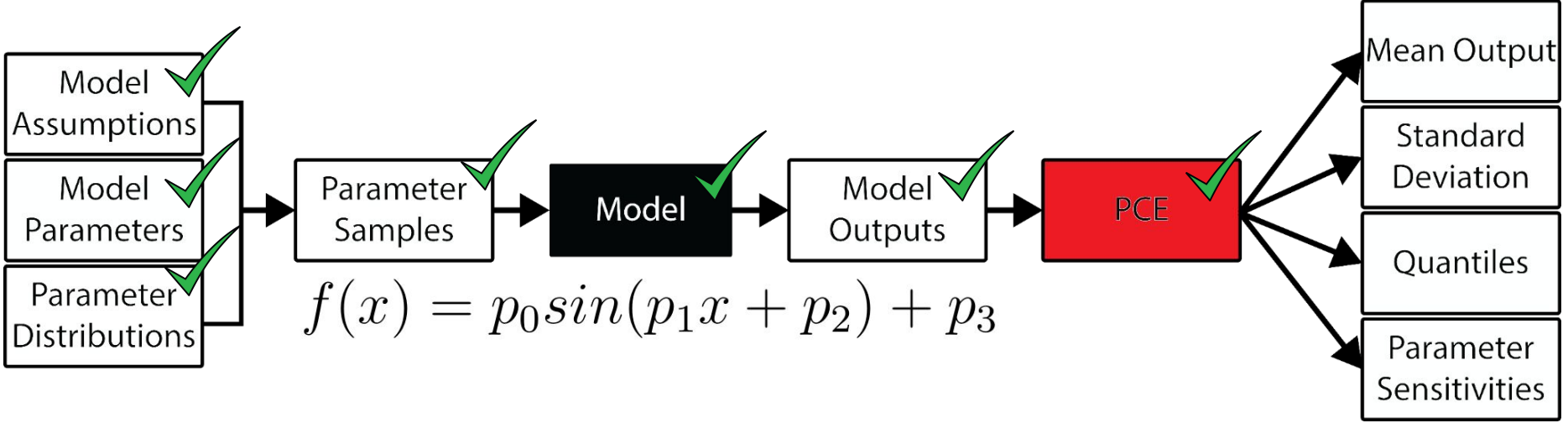
$$f(x) = p_0 sin(p_1 x + p_2) + p_3$$



$$p_0 \in [0.5, 1], p_3 \in [0 - 0.1, 0.1]$$



$p_0 \in [0.5, 1], p_3 \in [0 - 0.1, 0.1]$



$p_0 \in [0.5, 1], p_3 \in [0 - 0.1, 0.1]$

Model Assumptions

Model Parameters

Parameter Distributions

Parameter Samples

Model

Model Outputs

PCE

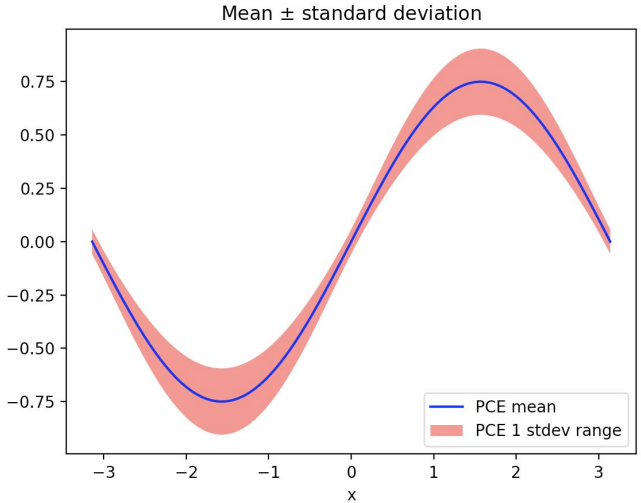
Mean Output

Standard Deviation

Quantiles

Parameter Sensitivities

$$f(x) = p_0 sin(p_1 x + p_2) + p_3$$



$$p_0 \in [0.5, 1], p_3 \in [0 - 0.1, 0.1]$$

Model Assumptions

Model Parameters

Parameter Distributions

Parameter Samples

Model

Model Outputs

PCE

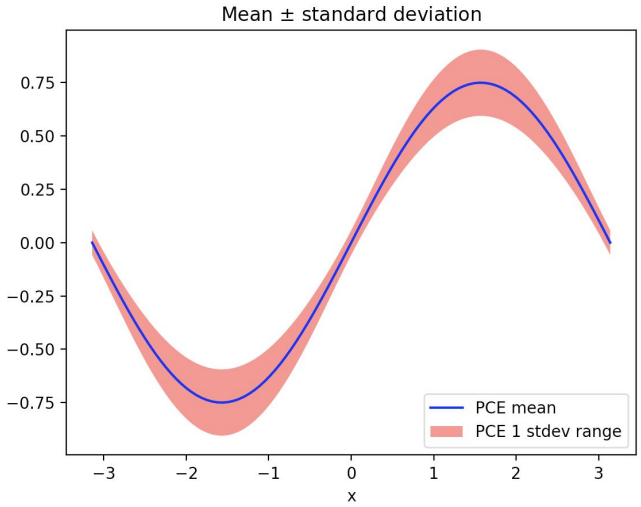
Mean Output

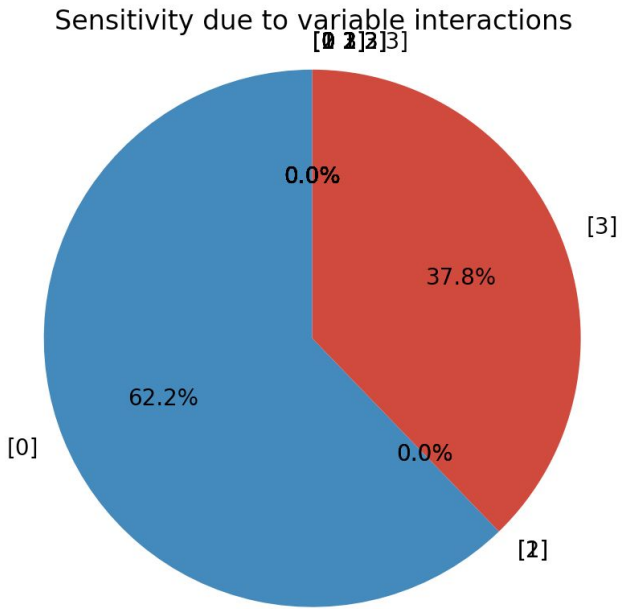
Standard Deviation

Quantiles

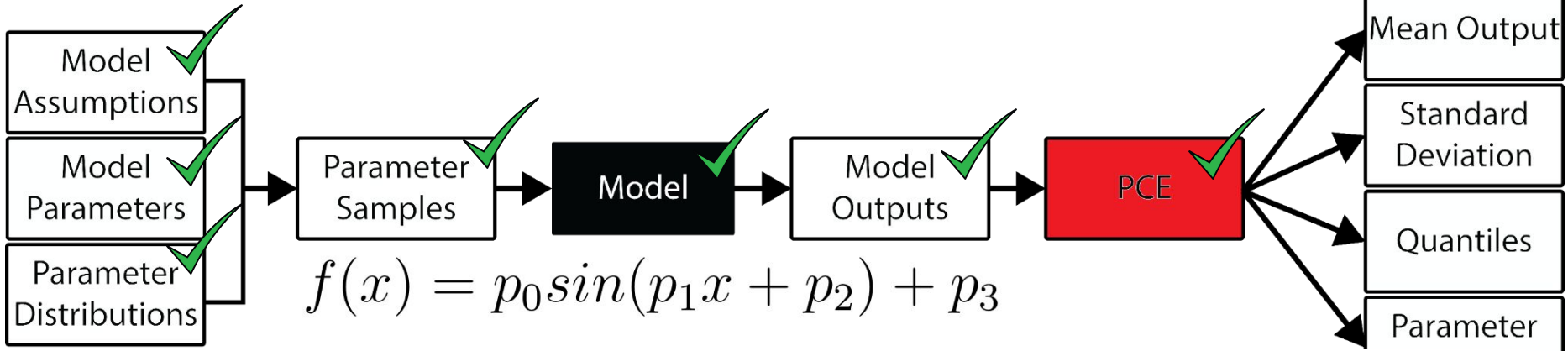
Parameter Sensitivities

$$f(x) = p_0 sin(p_1 x + p_2) + p_3$$

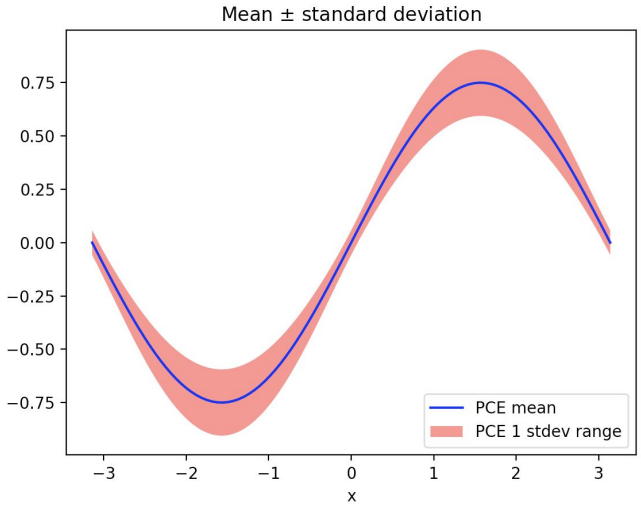


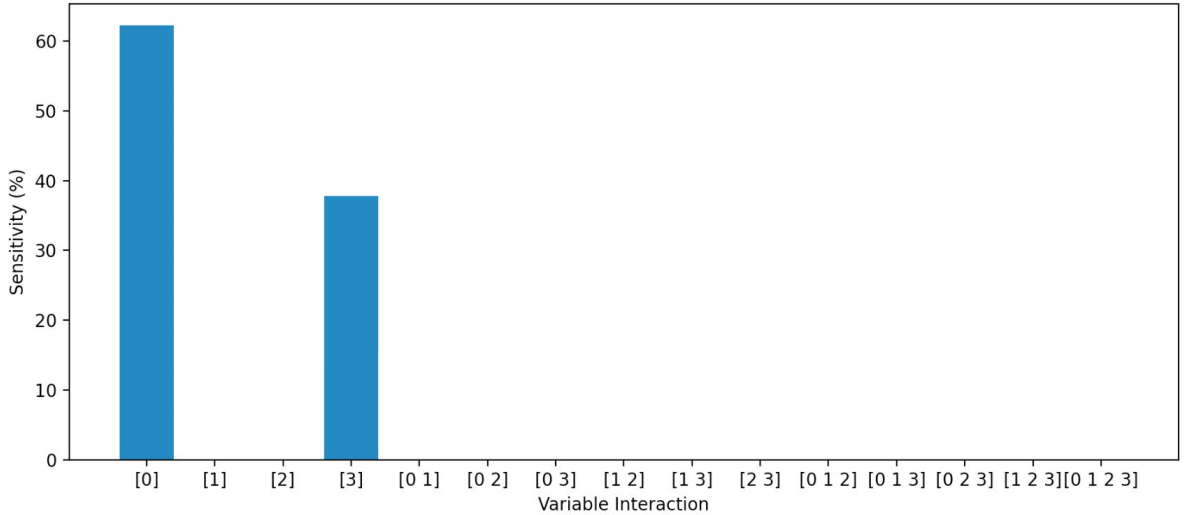


$$p_0 \in [0.5, 1], p_3 \in [0 - 0.1, 0.1]$$

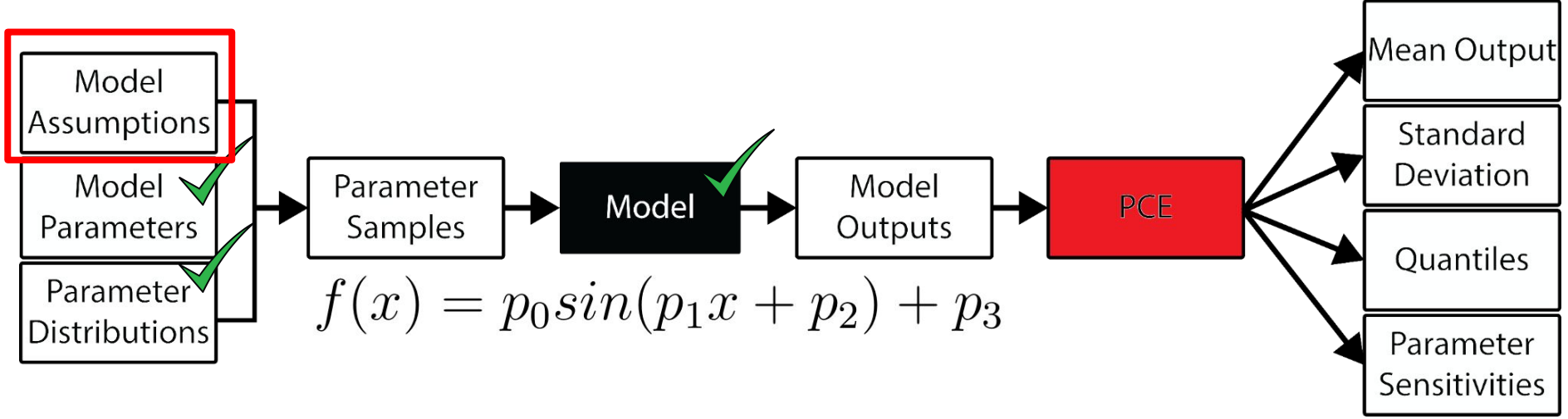


$$f(x) = p_0 sin(p_1 x + p_2) + p_3$$





$$p_0 \in [0.5, 1], p_3 \in [0 - 0.1, 0.1]$$



$$0.5 \leq p_0 \leq 1$$

$$1 \leq p_1 \leq 1$$

$$0 \leq p_2 \leq 0$$

$$-0.1 \leq p_3 \leq 0.1$$

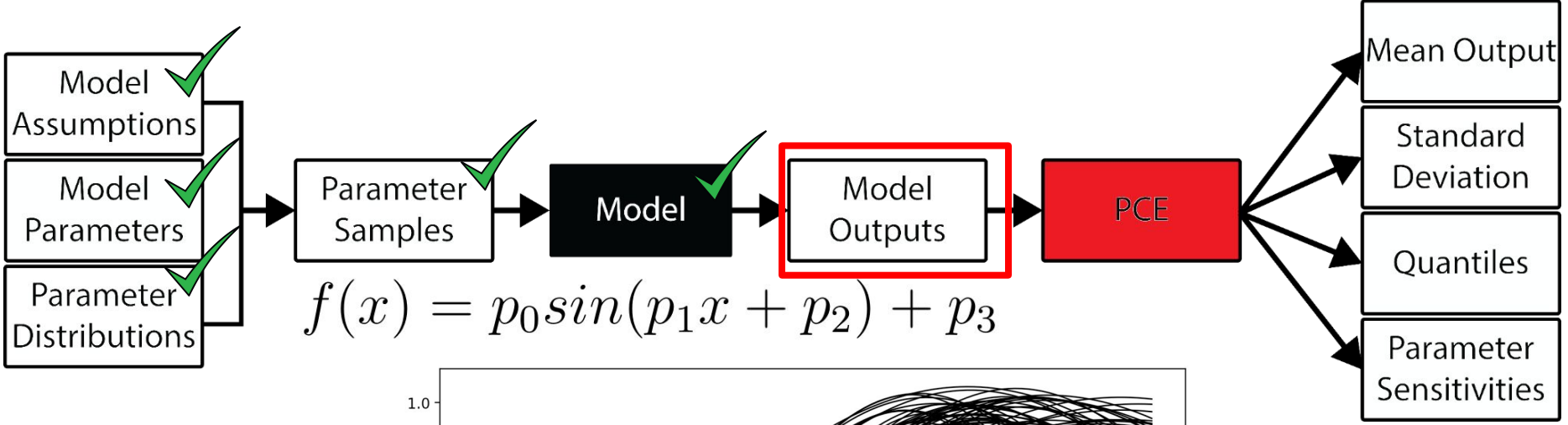
$$\rightarrow$$

$$0.5 \leq p_0 \leq 1$$

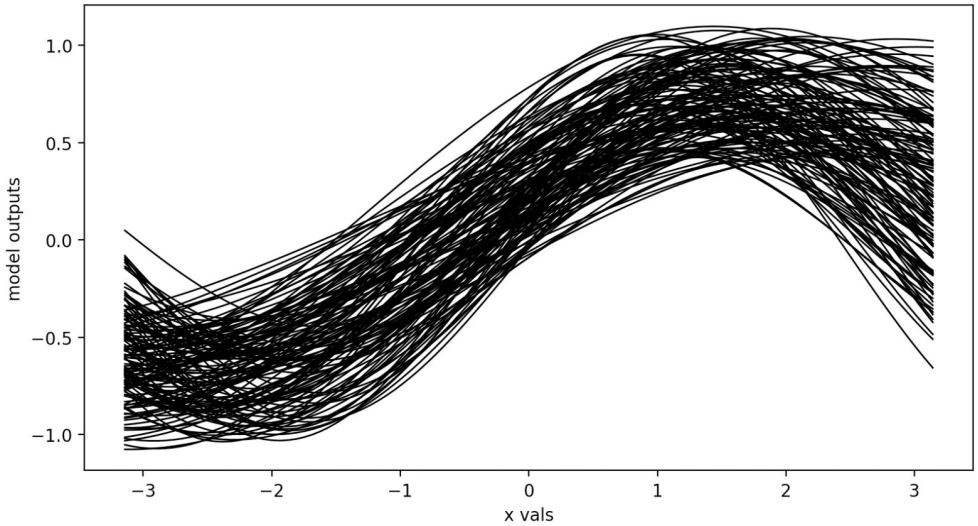
$$0.5 \leq p_1 \leq 1$$

$$0 \leq p_2 \leq \frac{\pi}{4}$$

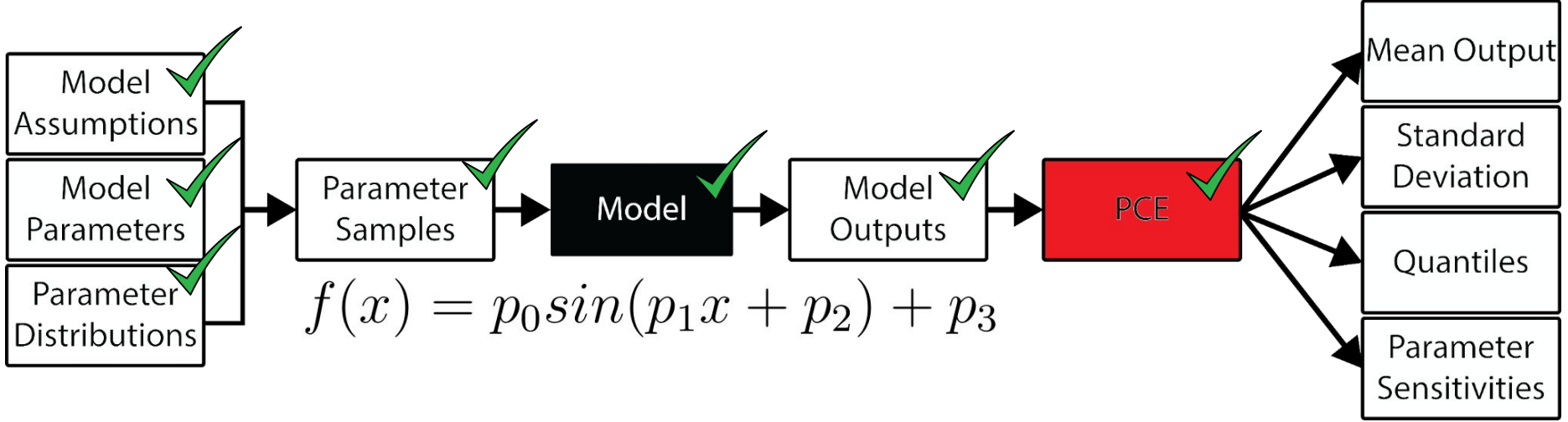
$$-0.1 \leq p_3 \leq 0.1$$



$$f(x) = p_0 sin(p_1 x + p_2) + p_3$$



$$p_0 \in [0.5, 1], p_1 \in [0.5, 1], p_2 \in [0, \frac{\pi}{4}], p_3 \in [0 - 0.1, 0.1]$$



$p_0 \in [0.5, 1], p_1 \in [0.5, 1], p_2 \in [0, \frac{\pi}{4}], p_3 \in [0 - 0.1, 0.1]$

Model Assumptions

Model Parameters

Parameter Distributions

Parameter Samples

Model

Model Outputs

PCE

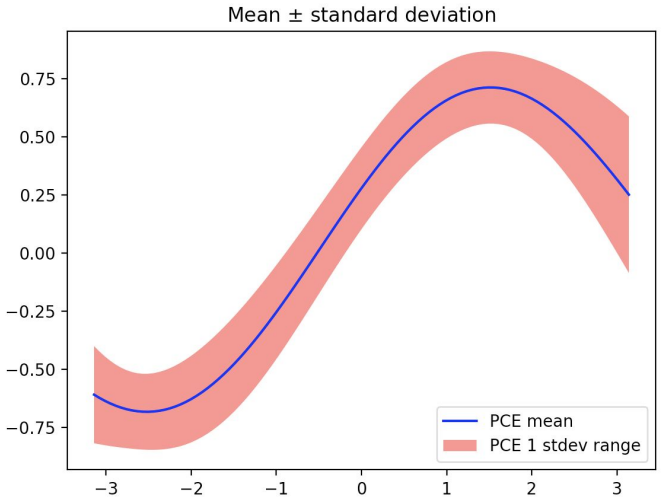
Mean Output

Standard Deviation

Quantiles

Parameter Sensitivities

$$f(x) = p_0 sin(p_1 x + p_2) + p_3$$



$$p_0 \in [0.5, 1], p_1 \in [0.5, 1], p_2 \in [0, \frac{\pi}{4}], p_3 \in [0 - 0.1, 0.1]$$

Model Assumptions

Model Parameters

Parameter Distributions

Parameter Samples

Model

Model Outputs

PCE

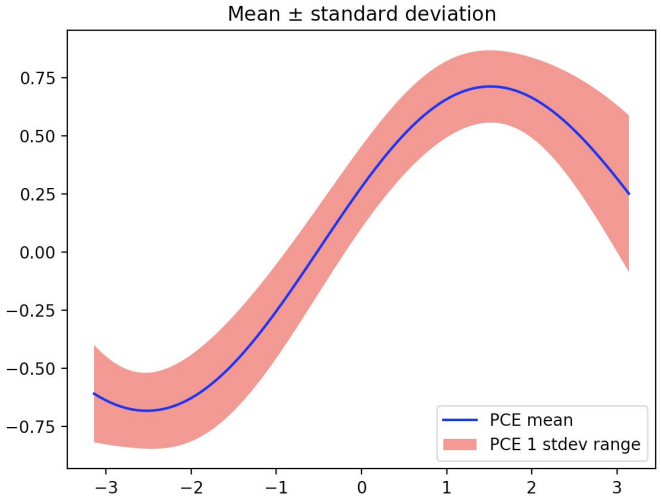
Mean Output

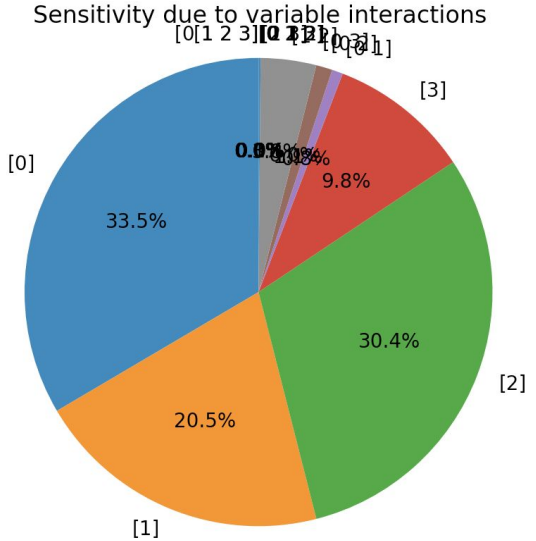
Standard Deviation

Quantiles

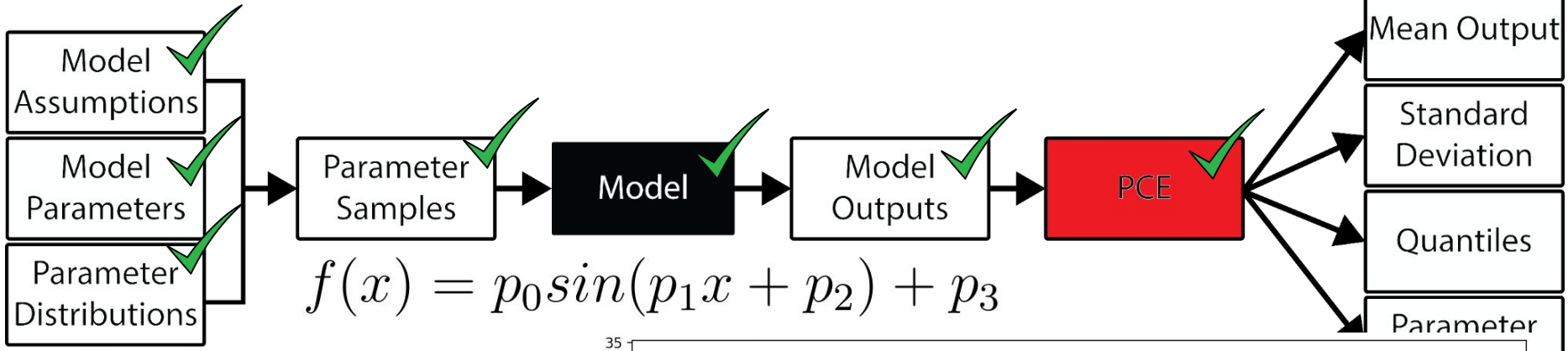
Parameter Sensitivities

$$f(x) = p_0 sin(p_1 x + p_2) + p_3$$

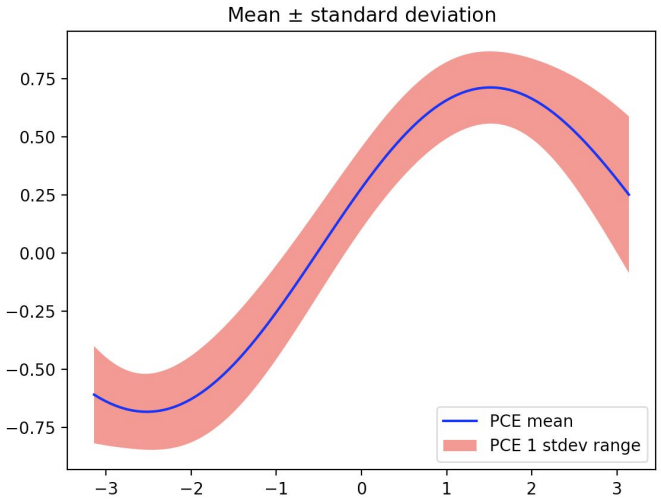


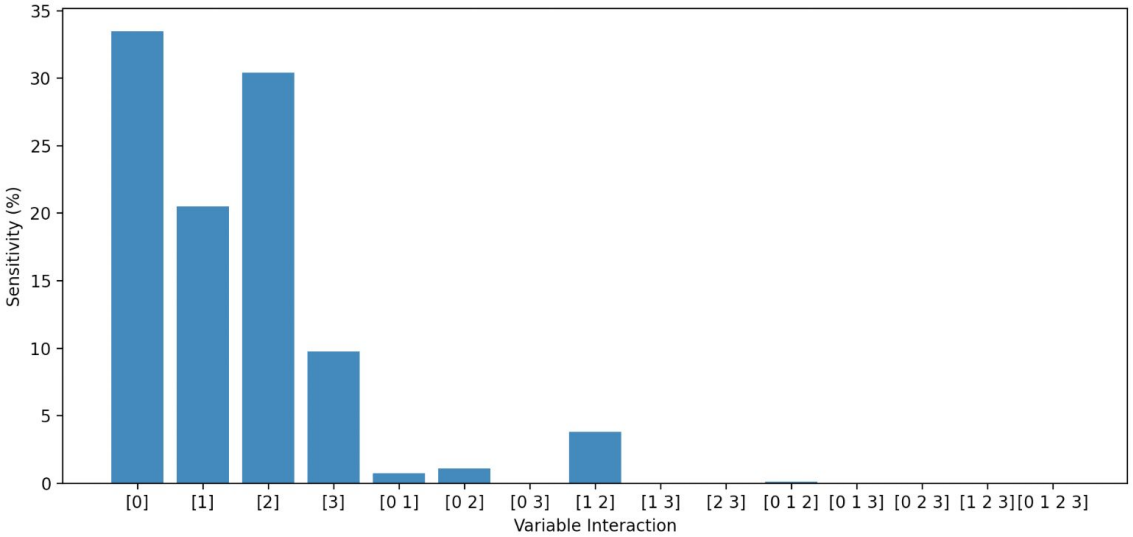


$$p_0 \in [0.5, 1], p_1 \in [0.5, 1], p_2 \in [0, \frac{\pi}{4}], p_3 \in [0 - 0.1, 0.1]$$



$$f(x) = p_0 sin(p_1 x + p_2) + p_3$$





$$p_0 \in [0.5, 1], p_1 \in [0.5, 1], p_2 \in [0, \frac{\pi}{4}], p_3 \in [0 - 0.1, 0.1]$$

Simple Example

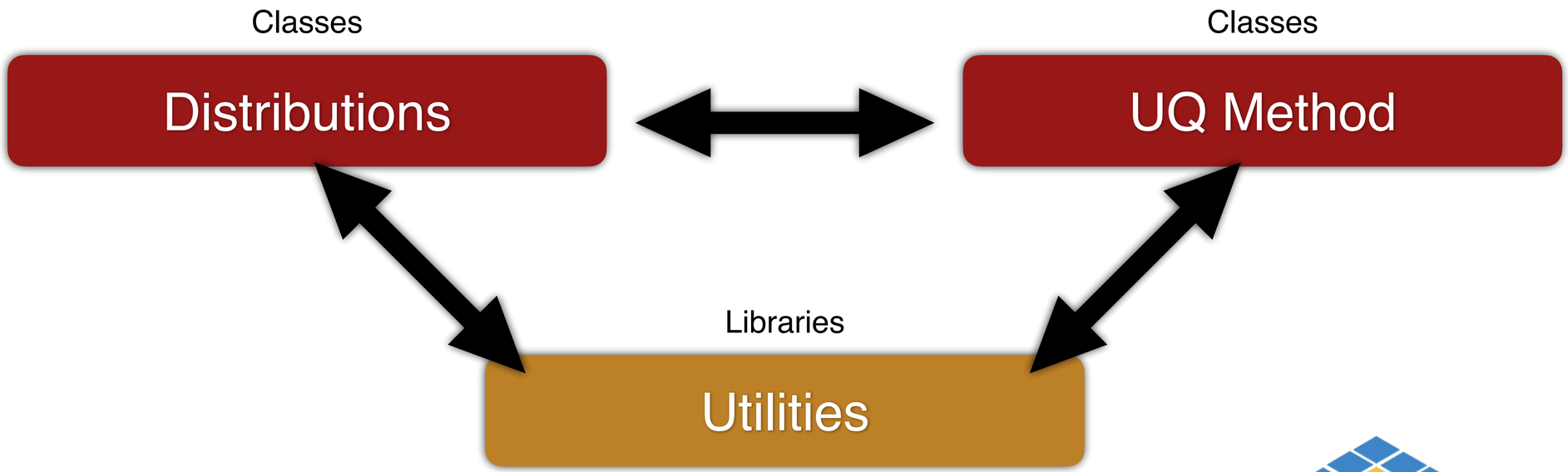
```
dimension = 4
dist = BetaDistribution(alpha=1, beta=1, dim=dimension)
order = 5
index_set = TotalDegreeSet(dim=dimension, order=order)

xVals = np.linspace(-1*np.pi,1*np.pi,100)
bounds = [0.5, 1, 1, 1, 1, 1, -1, 1]
model = lambda p: modelFunction(p, x = xVals, paramBounds=bounds)

pce = PolynomialChaosExpansion(index_set, dist)
pce.build(model)

# The parameter samples and model evaluations are accessible:
parameter_samples = pce.samples
model_evaluations = pce.model_output
```

Simple Example

```
# output statistics

mean = pce.mean()
stdev = pce.stdev()

variable_interactions = list(chain.from_iterable(combinations(range(dimension),
                        r) for r in range(1, dimension+1)))

global_sensitivity = pce.global_sensitivity(variable_interactions)
```

https://github.com/SCIInstitute/UncertainSCI/blob/master/demos/basic_uq_example.py

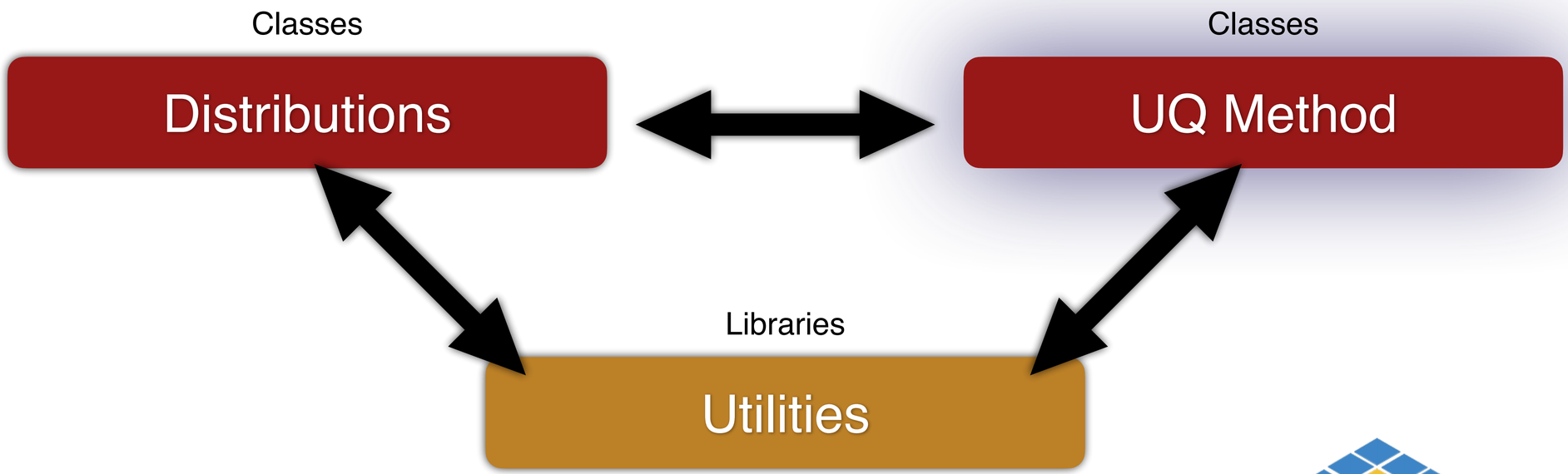
UncertainSCI Architecture



SciPy

NumPy

UncertainSCI Architecture



SciPy

NumPy

UQ Methods

Polynomial Chaos Expansion (PCE)

Monte Carlo

More to come

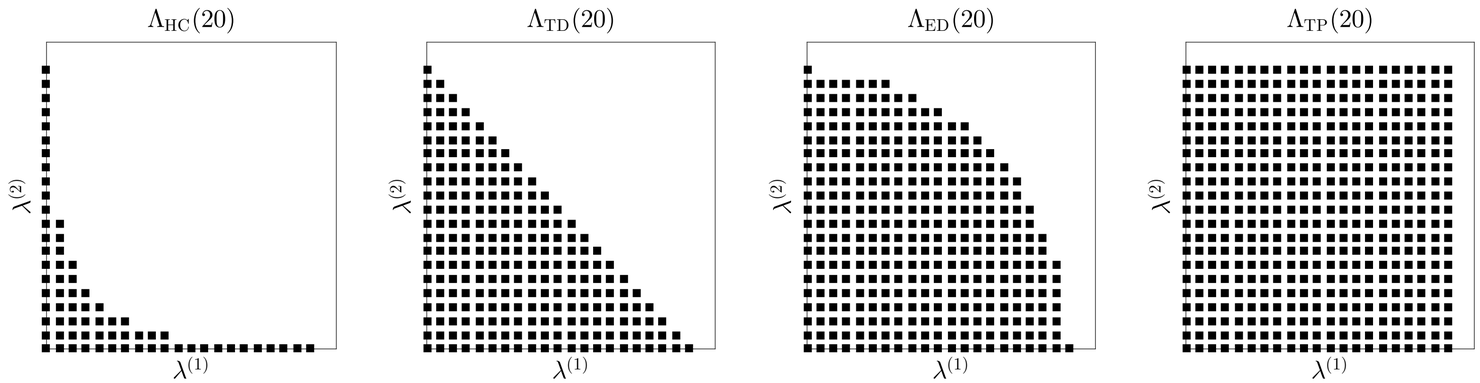
PC Recap

Let $f(P)$ be a quantity of interest that depends on random parameters P

- $P \in \mathbb{R}^d$ is a random variable with probability density w
- $f : \mathbb{R}^d \to \mathbb{R}$ is the forward simulation

PC appraoches construct the emulator

$$f(P) \approx f_N(P) := \sum_{j=1}^{N} \widehat{f}_j \phi_j(P), \qquad \mathbb{E}[\phi_j(P)\phi_k(P)] = \delta_{j,k}$$



Compute $\widehat{f}_j$ such that $f(p_m) \approx f_N(p_m)$ for $m = 1, \ldots, M$

PCE Class

Functions:

- set_distribution
- generate_samples (WAFP)
- build
- Stats:
  - Mean, Median, Stdev, Quantiles, Sensitivities
- adapt_expressivity

pce.build

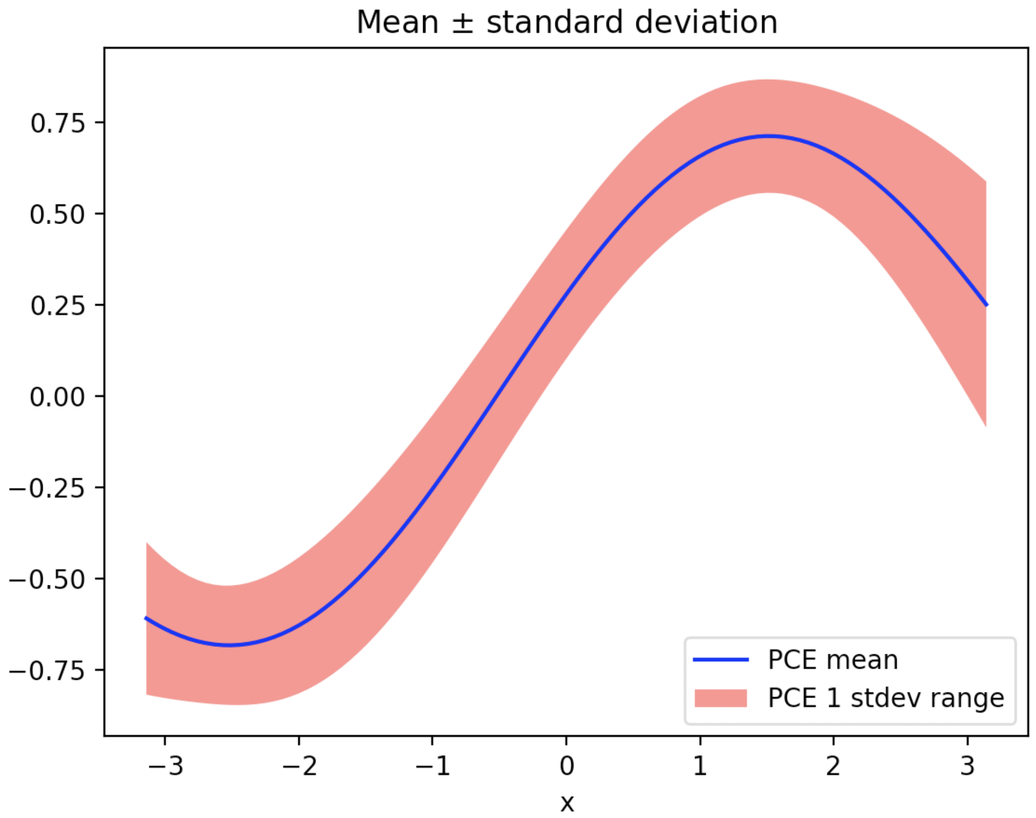
```
# lambda function
pce.build(model)

# Model Solutions only
pce.build(model_output=model_output)

# saved samples and solutions
pce.build(model_output=model_output, samples = samples)
```

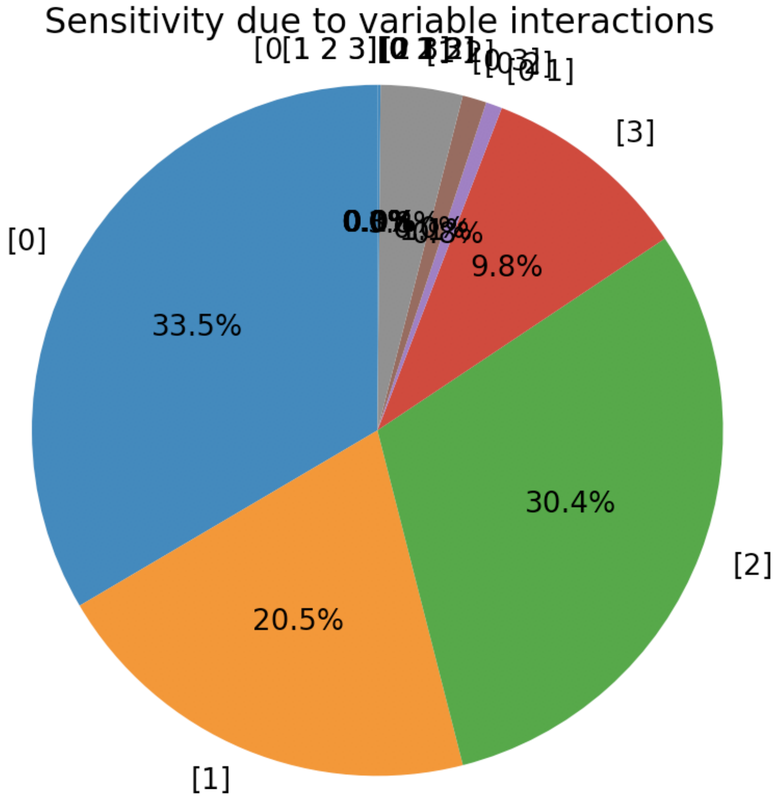
PCE Class

Functions:

- set_distribution
- generate_samples (WAFP)
- build
- Stats:
  - Mean, Median, Stdev, Quantiles, Sensitivities
- adapt_expressivity

Output Statistics

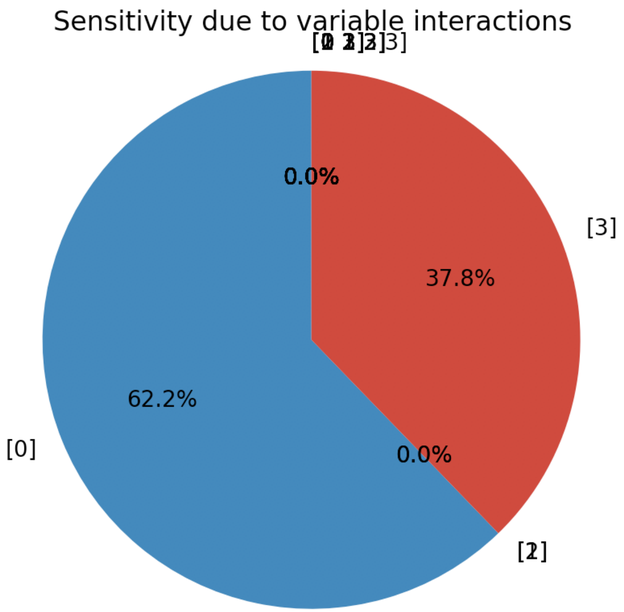




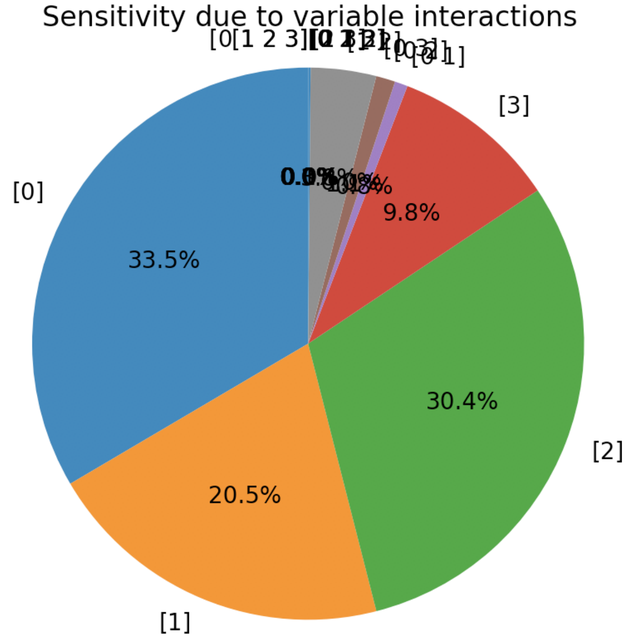
Output Statistics

```
mean = pce.mean()
stdev = pce.stdev()

variable_interactions = list(chain.from_iterable(combinations(range(dimension),
                        r) for r in range(1, dimension+1)))

global_sensitivity = pce.global_sensitivity(variable_interactions)

total_sensitivity = pce.total_sensitivity()

dq = 0.5/(Q+1)
q_lower = np.arange(dq, 0.5-1e-7, dq)[::-1]
q_upper = np.arange(0.5 + dq, 1.0-1e-7, dq)
quantile_levels = np.append(np.concatenate((q_lower, q_upper)), 0.5)

quantiles = pce.quantile(quantile_levels, M=int(2e3))
median = pce.quantile(0.5, M=int(1e3))[0, :]
```

Global Sensitivities



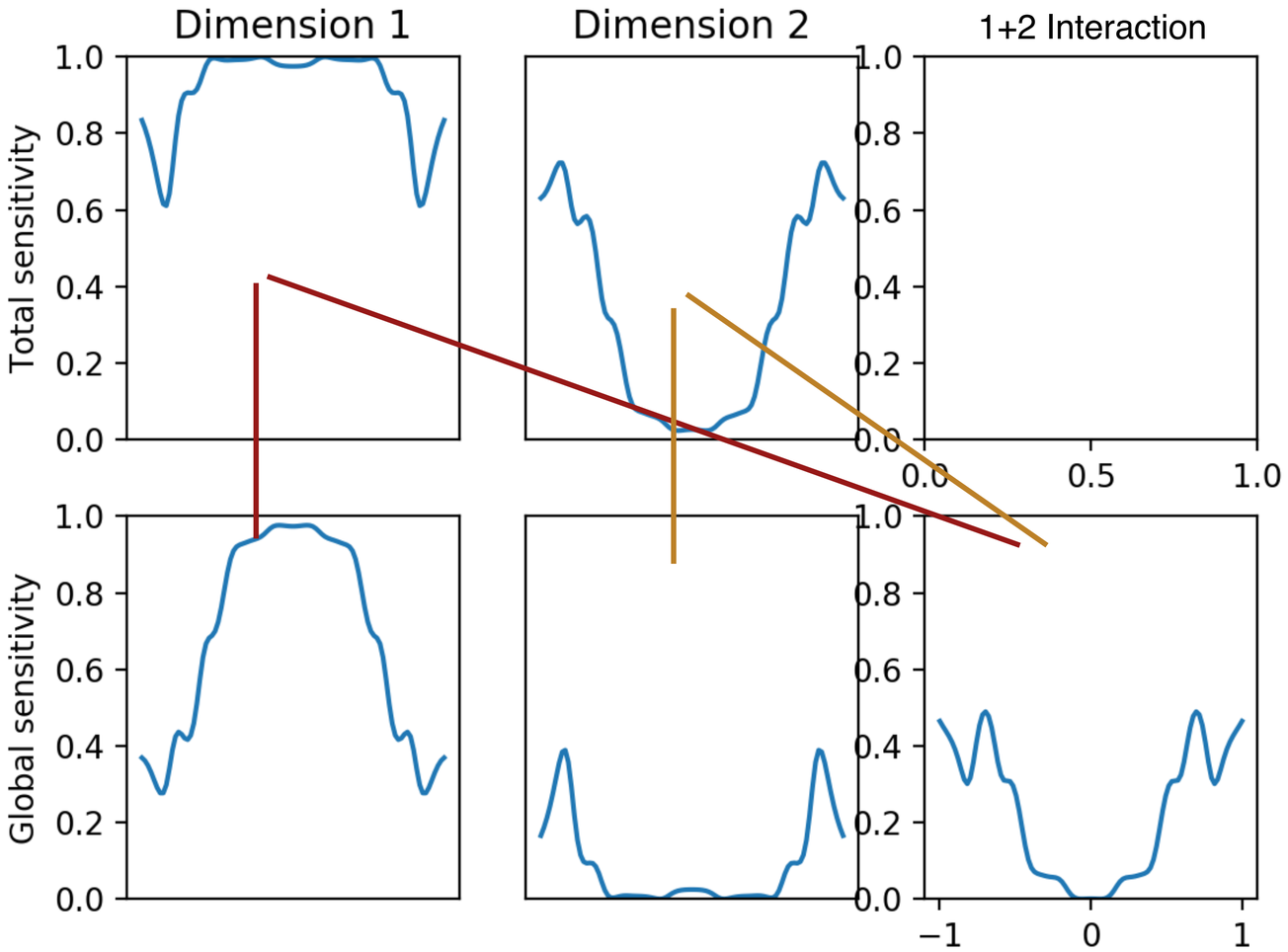
Includes interactions
Sums to 1
Fraction of variance



2 Parameters

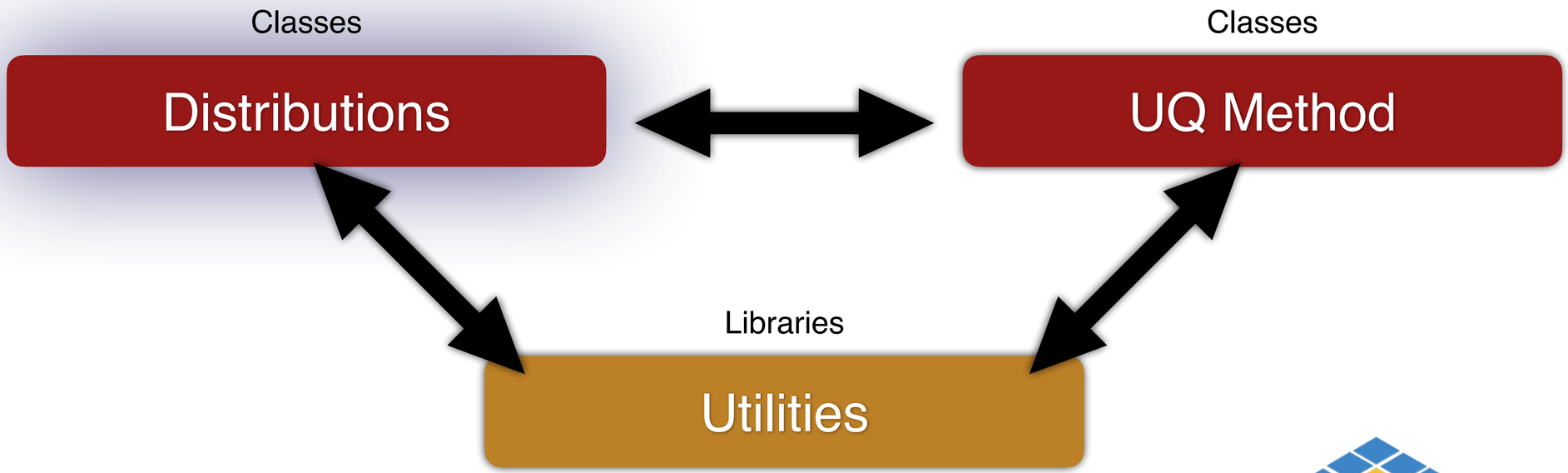
4 Parameters

Total Sensitivities



For each input
May not sum to 1
Indirectly related to variance

UncertainSCI Architecture



SciPy

NumPy

Multivariate Distributions

Gaussian

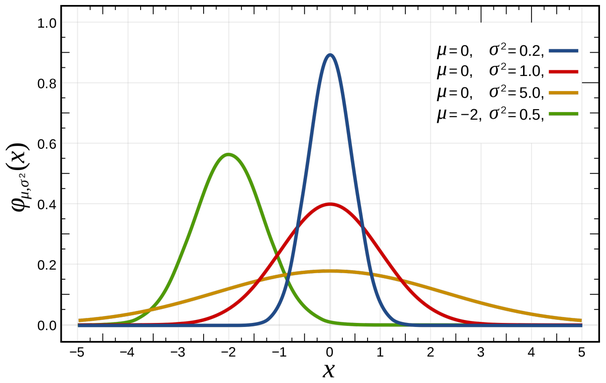
Exponential

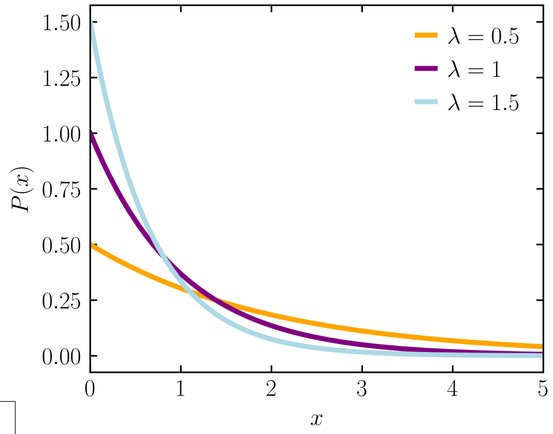
Beta

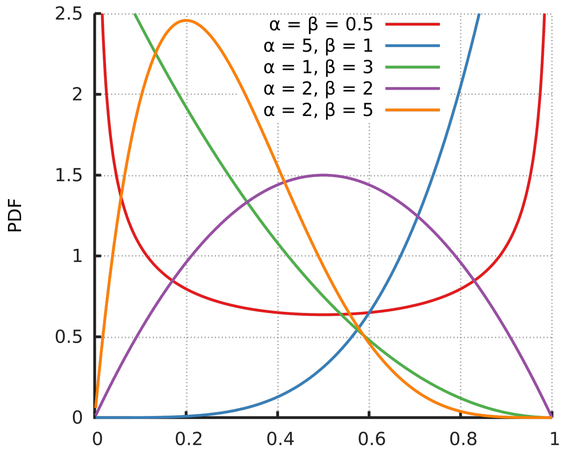
Uniform

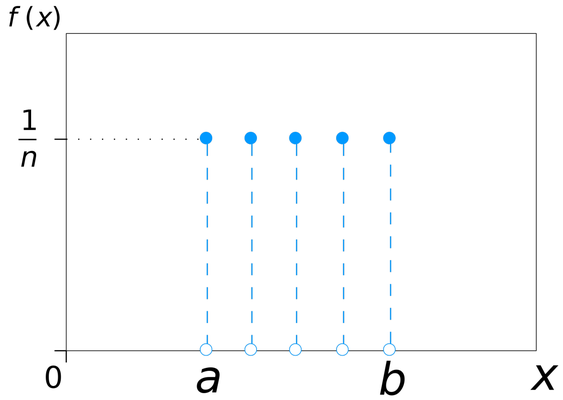
Discreet Uniform

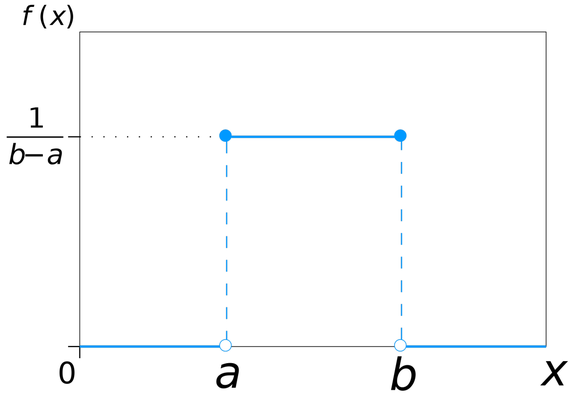
Tensorial











Images from wikipedia.com

Multivariate Distributions

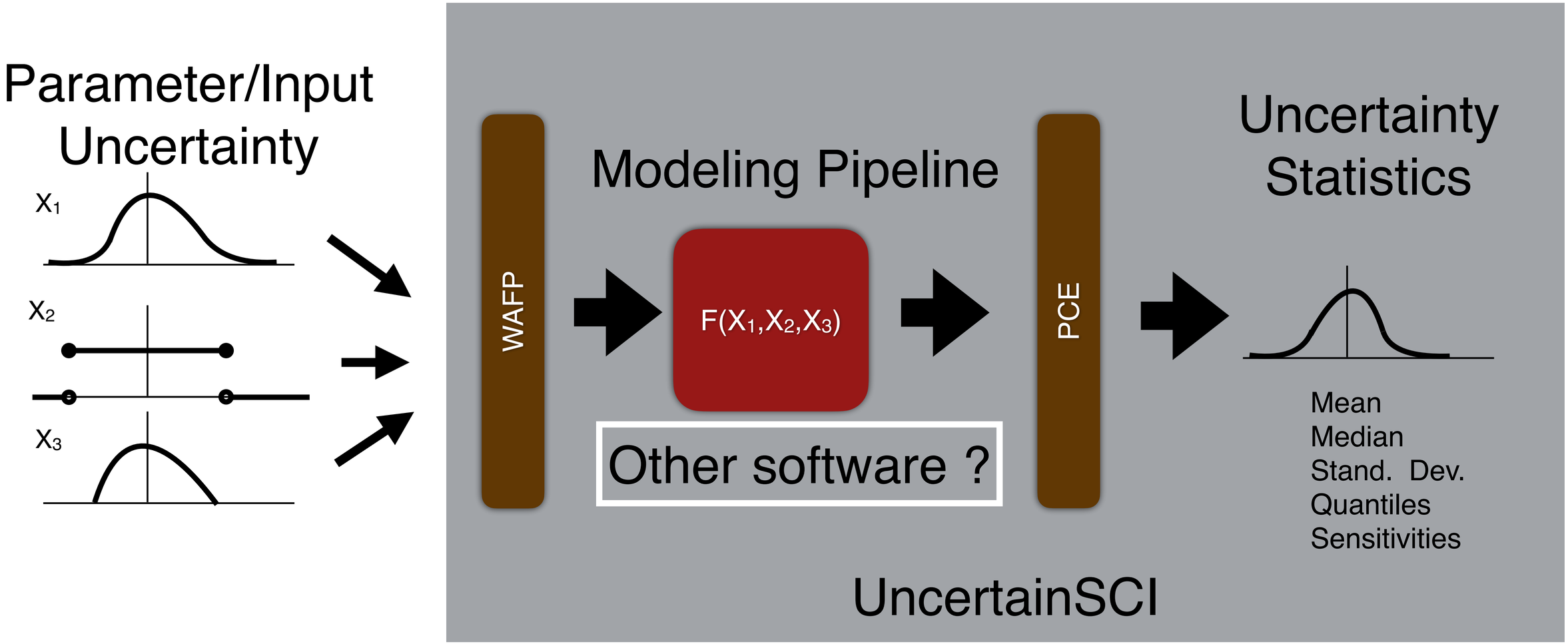
Distribution Class

- Constructor
  - (mean, std, domain, cov, etc.)
- MC_samples

Distribution Examples

```
dist = NormalDistribution(mean=mean, cov=cov, dim=dimension)

dist = BetaDistribution(alpha=alpha, beta=beta, dim=dimension)

dist = ExponentialDistribution(lbd=lbd, loc=loc)
```

UQ Pipeline


Parameter/Input Uncertainty
X1
X2
X3
WAFP
Modeling Pipeline
F(X1,X2,X3)
Other software ?
PCE
Uncertainty Statistics
Mean
Median
Stand. Dev.
Quantiles
Sensitivities
UncertainSCI

UncertainSCI with Matlab

jess@sci.utah.edu

Starting Matlab in Python

```
import matlab.engine

if len(matlab.engine.find_matlab()) < 1:
    eng = matlab.engine.start_matlab('-desktop')
    print('Starting new matlab')
else:
    eng = matlab.engine.connect_matlab()
    print('Connected to existing matlab')
print('Matlab Started')
```

https://www.mathworks.com/help/matlab/matlab-engine-for-python.html?s_tid=CRUX_lftnav

Define Model

```
def ForwardModel(p = [0.5, 0.5, 0.5]):
    eng.workspace['param'] = p

    # RunFWDForUQ is a matlab function
    result = np.double(eng.eval('RunFWDForUQ(param)'))

    #Set the centroid translation
    return result.reshape((result.size,))
```

Run UncertainSCI

```
# set up UncertainSCI
dimension = 3
alpha = 1.
beta = 1.
dist = BetaDistribution(alpha, beta, dimension)
order = 5
indices = TotalDegreeSet(dim=dimension, order=order)
pce = PolynomialChaosExpansion(indices, dist)

# Define model
model = lambda p: ForwardModel(p)
# Compute PCE (runs model)
lsq_residuals = pce.build_pce_wafp(model)

# move UQ results to matlab
eng.workspace['meanSig'] = pce.mean().tolist()
eng.workspace['std'] = pce.stdev().tolist()
eng.workspace['quantile_5'] = pce.quantile(.5).tolist()
eng.workspace['quantile_25'] = pce.quantile(.25).tolist()
eng.workspace['quantile_75'] = pce.quantile(.75).tolist()
```

Running Uncertainty with Another Software

SCIRun, CARP, ECGSim, Slicer, etc

Generate and Save Samples

```
# Setup and parameter samples

domain = np.array([[-125, 125], [-85, 85], [-60, 60], [-40, 40]]).T
sample_params = { "dimension": 4, "alpha": 1, "beta": 1, "domain": domain}

pce = set_distribution(sample_params)

pce.generate_samples()
scipy.io.savemat(Filename, dict(samples=pce.samples))
```

Run Model Through Python or Externally

```
def external_model(sample_file):

  Run_other_software(sample_file)

  return solution_file
```

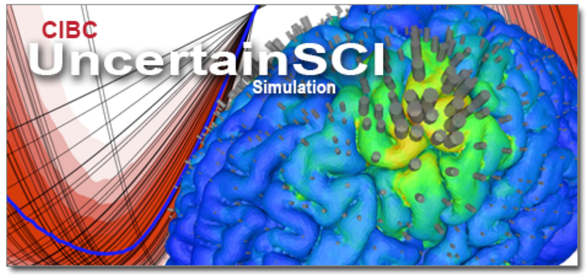
Or run asynchronously

Run PCE with Solutions and Samples

```
# load model solutions
tmp = scipy.io.loadmat(solution_file[:-4]+".mat")
model_output  = tmp["model_solutions"]

# load Samples saved from UncertainSCI
tmp_samp = scipy.io.loadmat(samples_file)
samples = tmp_samp['samples']


pce.build(model_output=model_output, samples = samples)
```

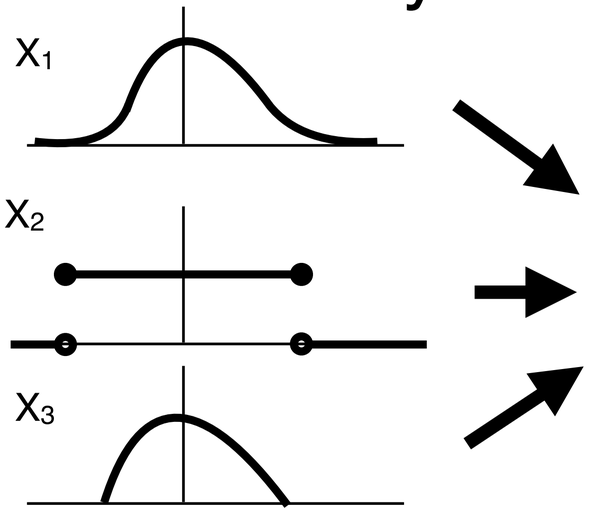
Compute Statistics

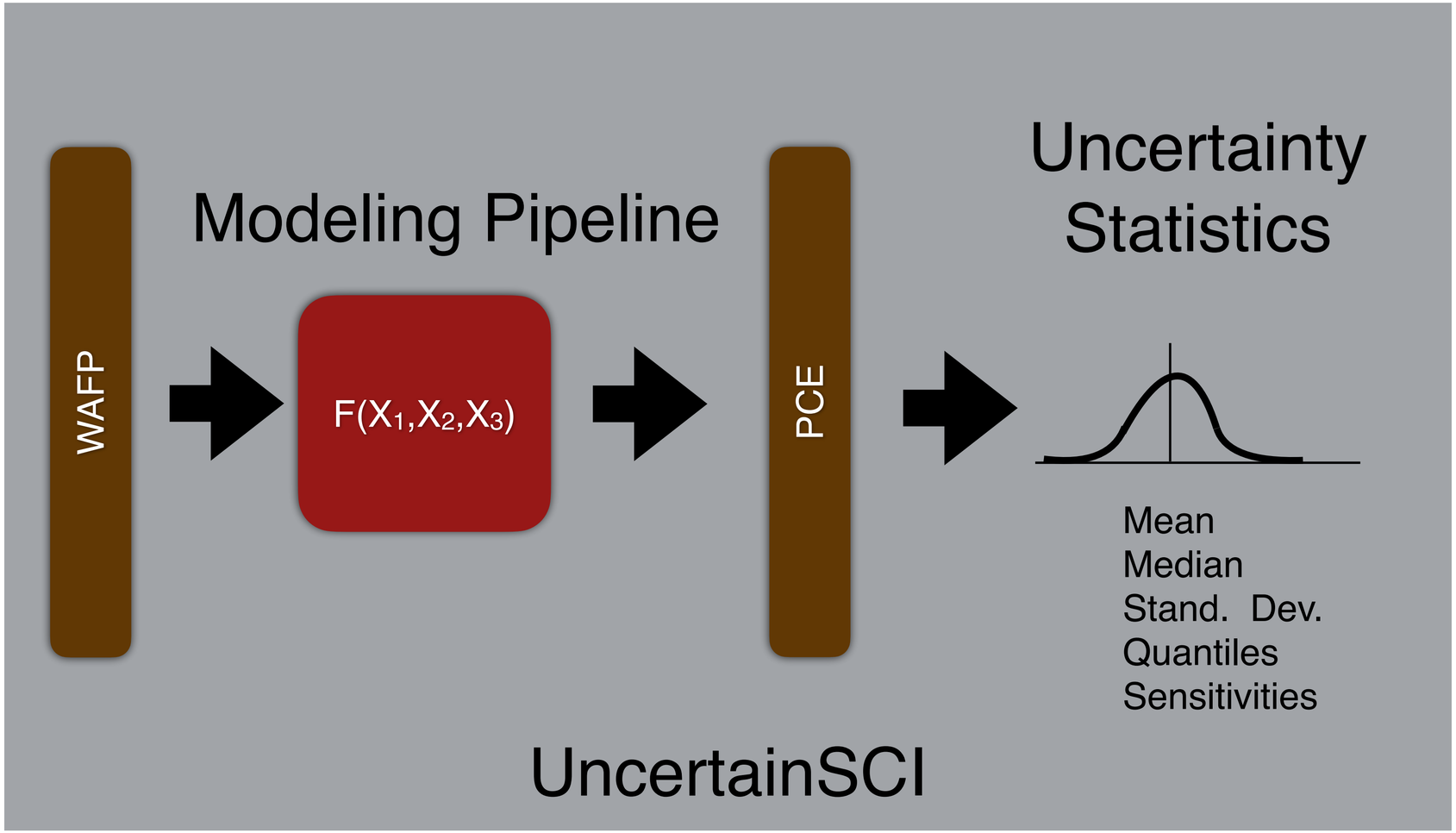
```
# output statistics
mean = pce.mean()
stdev = pce.stdev()

variable_interactions = list(chain.from_iterable(combinations(
      range(sample_params['dimension']), r)
      for r in range(1, sample_params['dimension']+1)))
global_sensitivity = pce.global_sensitivity(variable_interactions)
total_sensitivity = pce.total_sensitivity()

dq = 0.5/(Q+1)
q_lower = np.arange(dq, 0.5-1e-7, dq)[::-1]
q_upper = np.arange(0.5 + dq, 1.0-1e-7, dq)
quantile_levels = np.append(np.concatenate((q_lower, q_upper)), 0.5)

quantiles = pce.quantile(quantile_levels, M=int(2e3))
median = pce.quantile(0.5, M=int(1e3))[0, :]
```



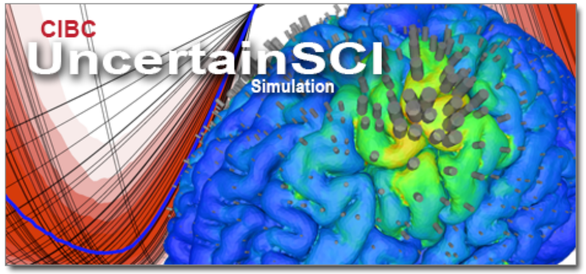
UncertainSCI Pipeline

Parameter/Input Uncertainty





UncertainSCI Design Goals



Numerical accuracy

Adaptability to multiple problem types

Interfacing with diverse tools

Simple API

Acknowledgements

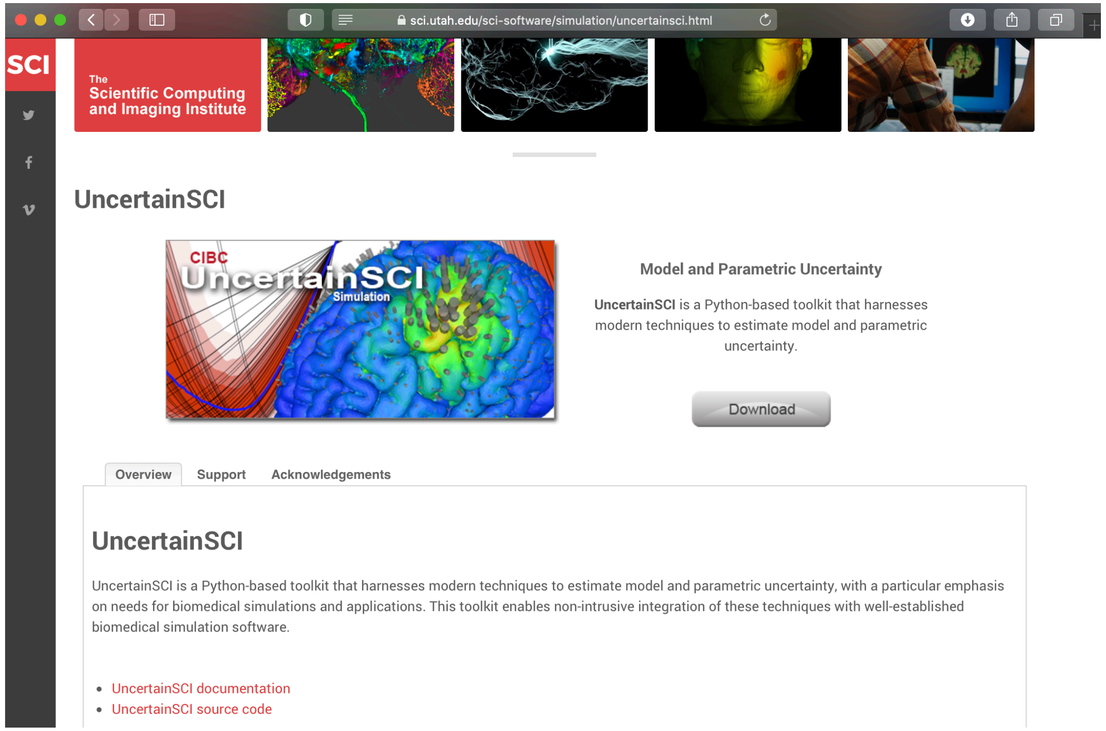
People

Jess D Tate
Zexin Liu
Jake A Bergquist
Sumientra Rampersad
Dan White
Chantel Charlebois
Lindsay C Rupp
Dana H Brooks
Akil Narayan
Rob S MacLeod

Support

Center for Integrative Biomedical Computing:
NIGMS P41 GM103545, NIGMS R24 GM136986
UncertainSCI: NIBIB U24EB029012

Get UncertainSCI Today



https://sci.utah.edu/sci-software/simulation/uncertainsci.html

https://github.com/SCIInstitute/UncertainSCI