*SAND 2009-0511P*

# DAKOTA 101

## Uncertainty Quantification

**http://www.cs.sandia.gov/dakota**

**Learning goals:**

- **Define uncertainty quantification and know when and why to apply it**
- **Run a sampling study**
- **Understand options for postprocessing the output uncertainty**

*Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

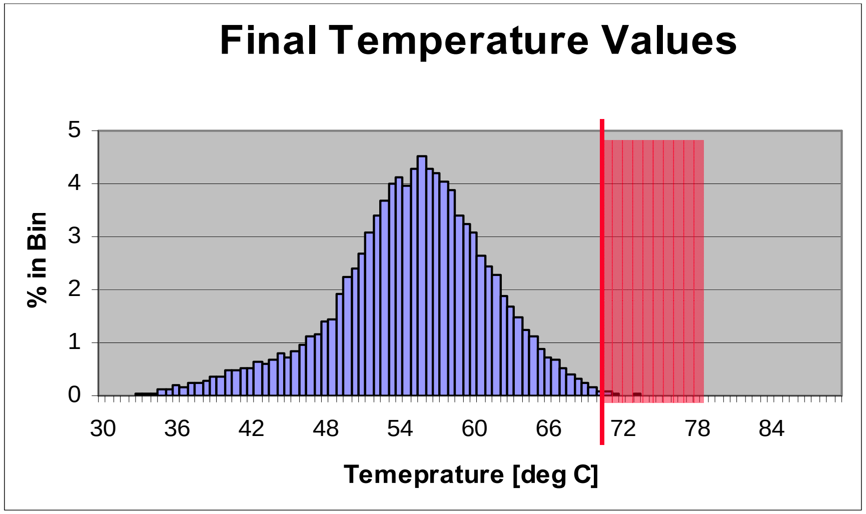
# Uncertainty Quantification Methods

- Most computer models for engineering applications are developed to help assess a design or regulatory requirement.
- Formal DOE process for Quantification of Margins and Uncertainties (QMU): quantifying impact of uncertainties in the decision context:

  Prob (System Response > T) < 0.01

## Goals of UQ methods:

- Based on uncertain inputs (UQ), determine distribution function (uncertainty) of outputs and probabilities of failure
  (reliability metrics)
- Identify the mean, variance, and higher moments of the output.
- Identify inputs whose variances contribute most to output variance (global sensitivity analysis)

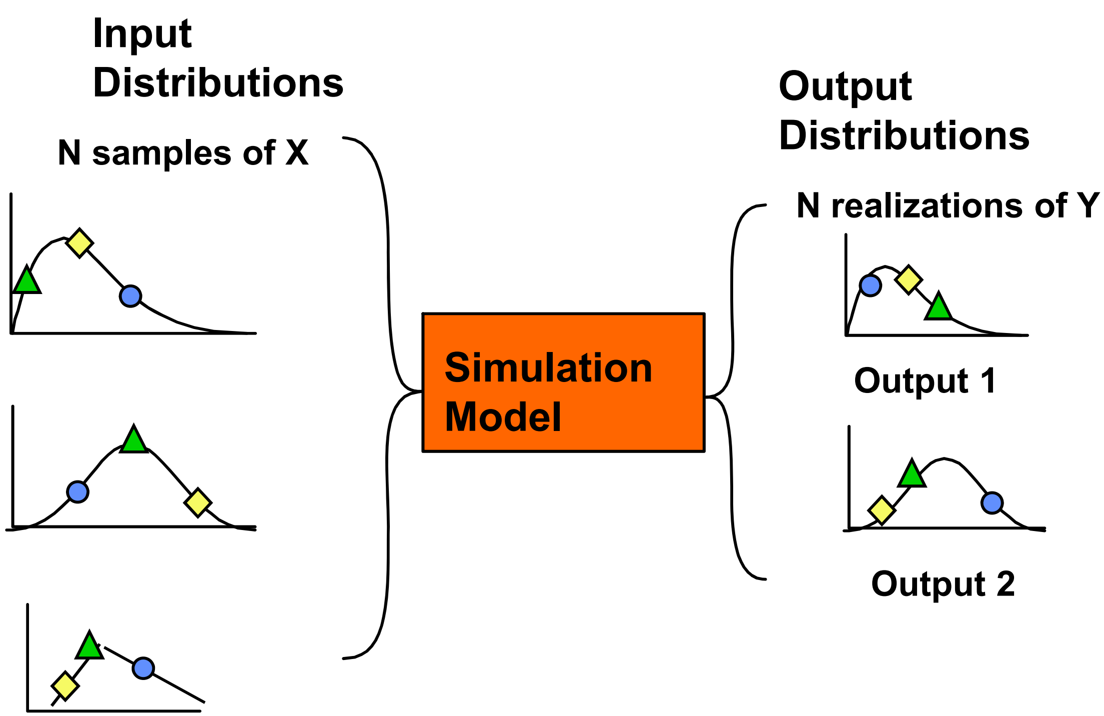



*Class UQ goals?*

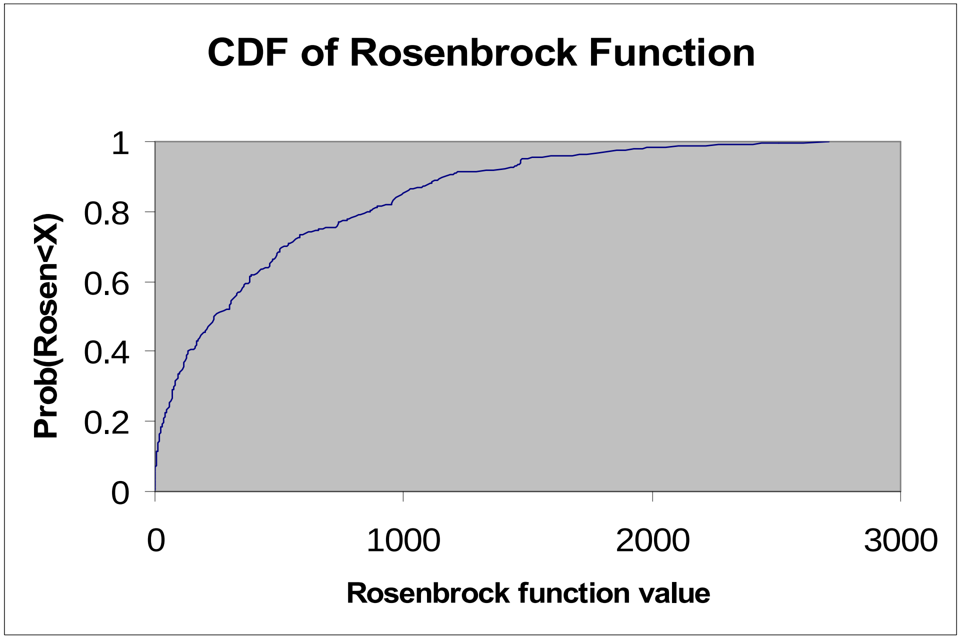
# Common UQ Method: Random Sampling

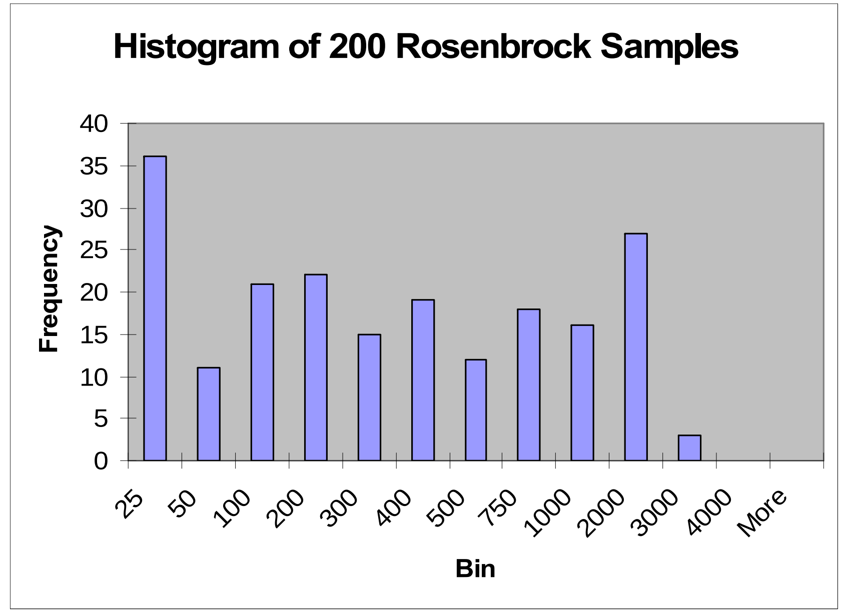
- Assume distributions on the uncertain input values
- Sample from those distributions
- Run the model with the sampled values

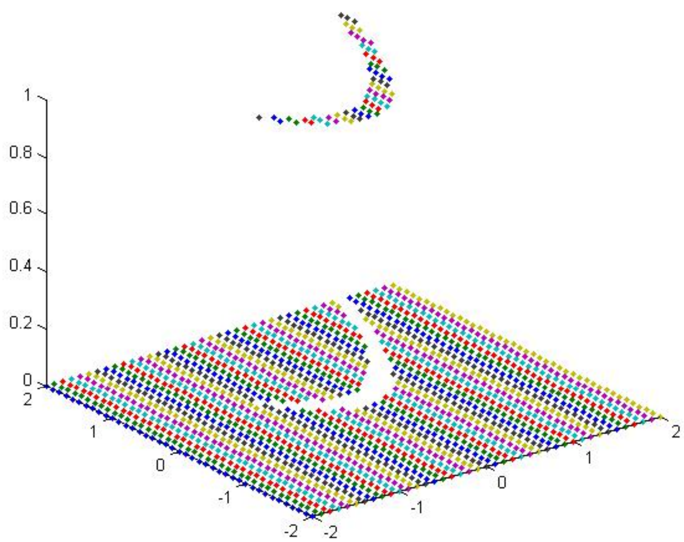
Repeat to build up a distribution of the outputs.

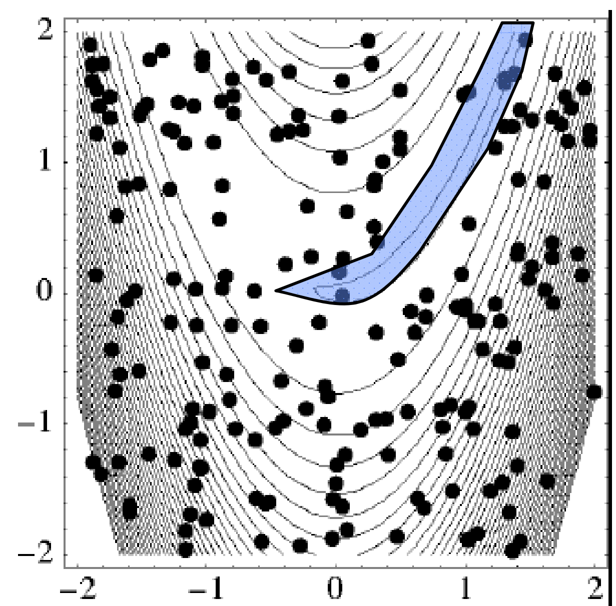


# Probability of Failure,
# e.g., P(`rosenbrock` < 3)



# Class Exercise: UQ with Sampling

- Recall higher level goal: determine mean system response, variability, margin to failure
- GOAL: Perform UQ on Rosenbrock using sampling:
  - Different seed values. *How do the failure probabilities differ?*
  - Different numbers of samples. *What happens to the confidence intervals on the mean and standard deviation as the number of samples increases or decreases from 200?*
  - Different distributions?
    - Instead of uniform on [-2,2] try a normal distribution for both variables, with mean 0 and standard deviation 0.5
    - What do you expect would happen when changing from uniform to normal?
    - What if the normal distributions have mean 1?

- Build from scratch or see: `examples/tutorial/dakota_rosenbrock_nond.in`

# Other UQ methods

- Reliability Methods: Focus on finding the probability of failure by transforming the UQ problem to an optimization problem.

$$p_f = \int \ldots \int_{g() < 0} f_X(x_1, x_2, \ldots, x_n) dx_1 dx_2 \ldots dx_n$$

- Stochastic Expansion (e.g. Polynomial Chaos)
  - Represent the uncertain output as a stochastic process, specifically as a spectral expansion in terms of suitable orthogonal polynomials

$$f(X) \approx R = \sum_{j=0}^{P} a_j H_j(\xi)$$

- Interval Analysis: Given interval bounds on the inputs, determine interval bounds on the outputs (optimization vs. sampling)
- Second-order Probability: Epistemic "outer loop" and aleatory "inner loop)
- Dempster-Shafer: Propagate "belief structures" on inputs to outputs

# EXTRA SLIDES

# Example Input/Output: Sampling

```
strategy,
        single_method
          graphics
          tabular_graphics_data

method,
          nond_sampling
          samples = 200 seed = 17
          sample_type random
          response_levels = 100.0

model,
        single

variables,
         uniform_uncertain = 2
          lower_bounds -2.0  -2.0
          upper_bounds  2.0   2.0
          descriptor s        'x1'  'x2'

interface,
        direct
          analysis_driver = 'rosenbrock‘

responses,
         num_response_functions = 1
        no_gradients
        no_hessians
```

Sampling Input (examples/tutorial/dakota_rosenbrock_nond.in)

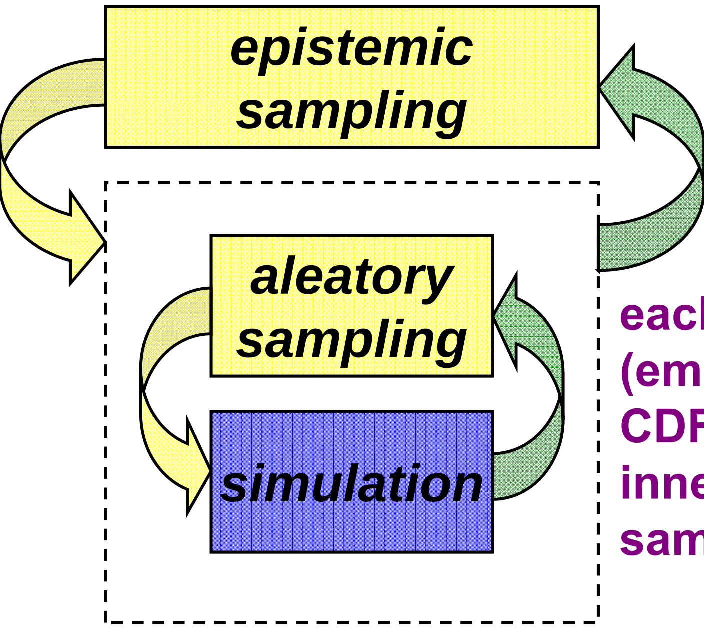
```
Statistics based on 200 samples:

Moments for each response function:
response_fn_1:  Mean = 4.43855e+02  Std. Dev. = 5.88920e+02  Coeff. of Variation
= 1.32683e+00

95% confidence intervals for each response function:
response_fn_1:  Mean = ( 3.61736e+02, 5.25973e+02 ), Std Dev = ( 5.36308e+02,
6.53068e+02 )

Probabilities for each response function:
Cumulative Distribution Function (CDF) for response_fn_1:
     Response Level      Probability Level  Reliability Index  General Rel Index
     --------------          -----------------       -----------------  -----------------
   1.0000000000e+02   3.7000000000e-01

Simple Correlation Matrix among all inputs and outputs:
                       x1            x2 response_fn_1
          x1  1.00000e+00
          x2 -4.33667e-03  1.00000e+00
response_fn_1  6.45646e-02 -4.81363e-01  1.00000e+00

Partial Correlation Matrix between input and output:
             response_fn_1
          x1  7.12791e-02
          x2 -4.82094e-01
```

Sampling Output

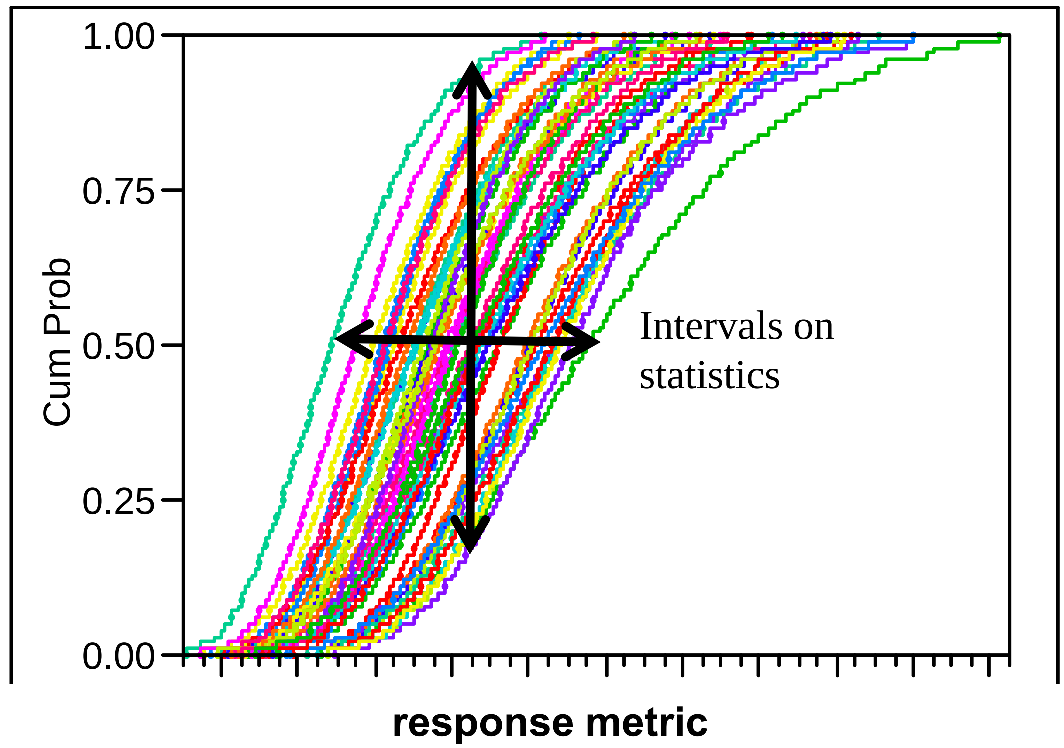
# Second-order Probability

- **For each outer loop sample of epistemic (interval) variables, run an inner loop UQ study over aleatory (probability) variables**

**50 outer loop samples**
**→ 50 CDF traces**

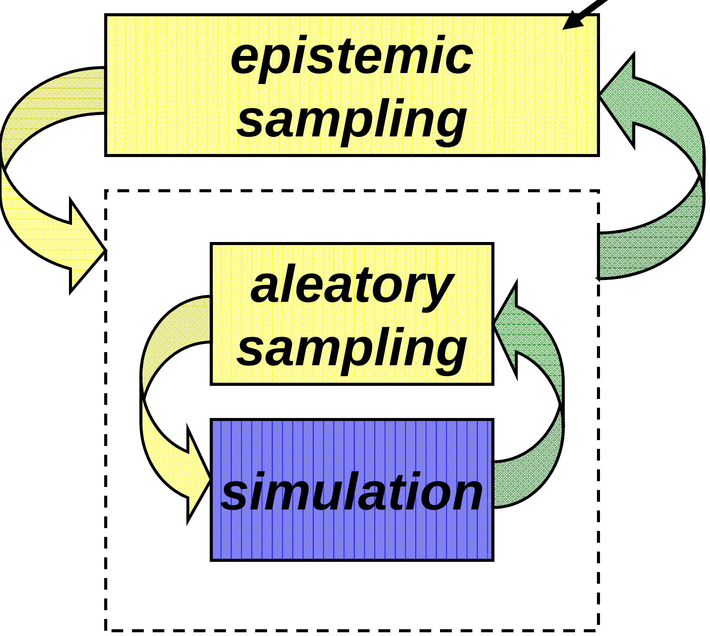



**each discrete (empirical) CDF: 100 inner loop samples**




***"Envelope" of CDF traces represents response epistemic uncertainty***

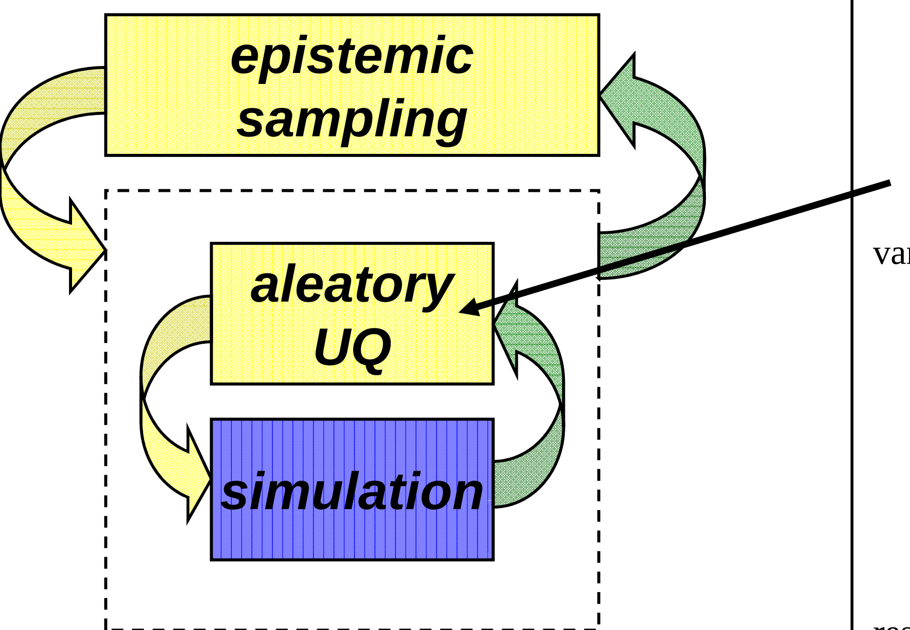
# Example Input: Second-Order Probability




```
method,
        id_method = 'EPISTEMIC'
        model_pointer = 'EPIST_M'
        nond_sampling   samples = 50 seed = 12347

model,
        id_model = 'EPIST_M'
        nested
          variables_pointer  = 'EPIST_V'
          sub_method_pointer = 'ALEATORY'
          responses_pointer  = 'EPIST_R'
          primary_variable_mapping   = 'X'    'Y'
          secondary_variable_mapping = 'mean' 'mean'
          primary_response_mapping   = 1. 0. 0. 0. 0. 0. 0. 0.
                                       0. 0. 0. 0. 1. 0. 0. 0
                                       0. 0. 0. 0. 0. 0. 0. 1.
variables,
        id_variables = 'EPIST_V'
        interval_uncertain =2
         num_intervals = 1 1
         interval_probs = 1.0 1.0
         interval_bounds = 400. 600. 800. 1200.
         descriptors      'X_mean' 'Y_mean'

responses,
        id_responses = 'EPIST_R'
        num_response_functions = 3
        response_descriptors = 'mean_wt' 'ccdf_beta_s' 'ccdf_beta_d'
```

PCE Input (examples/methods/dakota_uq_cantilever_sop_rel.in)

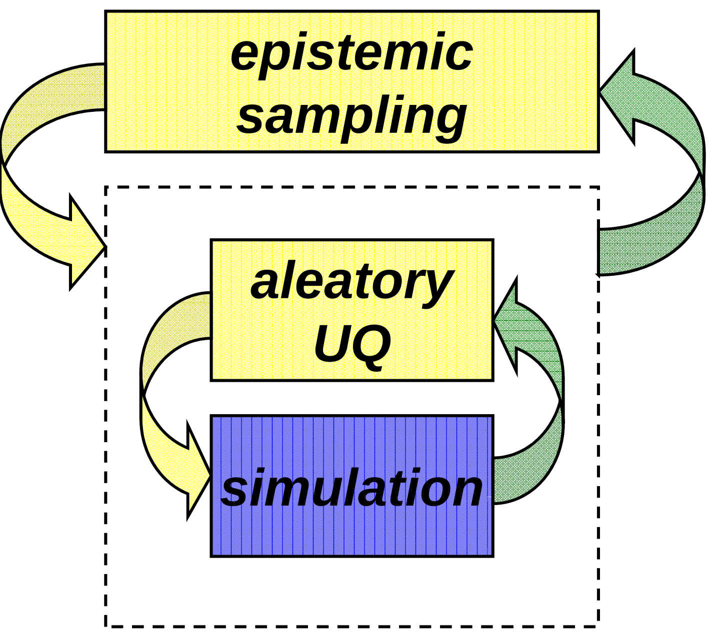
# Example Input: Second-Order Probability




```
method,
          id_method = 'ALEATORY'
          model_pointer = 'ALEAT_M'
          nond_local_reliability
            mpp_search no_approx
            num_response_levels = 0 1 1
            response_levels = 0.0 0.0
            compute reliabilities  complementary distribution
model,
          id_model = 'ALEAT_M'
          single
            variables_pointer = 'ALEAT_V'
            interface_pointer = 'ALEAT_I'
            responses_pointer = 'ALEAT_R'

variables,
          id_variables = 'ALEAT_V'
          continuous_design = 2
            initial_point    2.4522 3.8826
            descriptors'beam_width' 'beam_thickness'
          normal_uncertain = 4
            means             =  40000. 29.E+6 500. 1000.
            std_deviations    =  2000. 1.45E+6 100. 100.
            descriptors       =  'R' 'E' 'X' 'Y'
responses,
          id_responses = 'ALEAT_R'
          num_response_functions = 3
          response_descriptors = 'weight' 'stress' 'displ'
```

PCE Input (examples/methods/dakota_uq_cantilever_2nd_order.in)

# Example Output: Second-Order Probability




```
<<<<< Iterator nond_sampling completed.
<<<<< Function evaluation summary (ALEAT_I): 971 total (971 new, 0 duplicate)

Statistics based on 50 samples:

Min and Max values for each response function:
mean_wt:  Min = 9.5209117200e+00  Max = 9.5209117200e+00
ccdf_beta_s:  Min = 1.8001336086e+00  Max = 4.0744019409e+00
ccdf_beta_d:  Min = 1.9403177486e+00  Max = 3.7628144053e+00

Simple Correlation Matrix between input and output:
              mean_wt  ccdf_beta_s  ccdf_beta_d
     X_mean  9.40220e-16 -6.38145e-01 -9.14016e-01
     Y_mean  1.38778e-15 -7.93481e-01 -4.39133e-01
      ….
```

PCE Input (examples/methods/dakota_uq_cantilever_2nd_order.in)

# Reliability Analysis

- Assume that the probability of failure is based on a specific performance criterion which is a function of random variables, denoted $X_i$.
- The performance function is described by Z:

  $Z = g(X_1, X_2, X_3, \ldots, X_n)$
- The failure surface or limit state is defined as Z = 0. It is a boundary between safe and unsafe regions in a parameter space.
- Now we have a more general form of $P_{failure}$:

$$p_f = P(failure) = P(Z < 0)$$

$$p_f = \int \ldots \int_{g() < 0} f_X(x_1, x_2, \ldots, x_n) dx_1 dx_2 \ldots dx_n$$

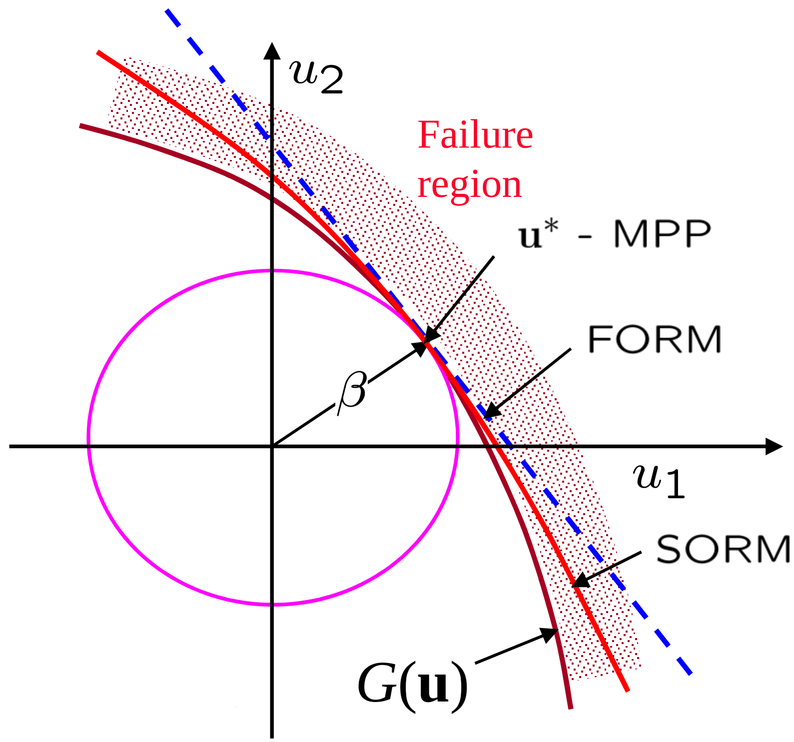
# Most Probable Point Methods

- Transform the uncertainty propagation problem into an optimization one:  first transform all of the non-normal random variables into independent, unit normal variables. Then, find the point on the limit state surface with minimum distance to the origin.

- The point is called the Most Probable Point (MPP).  The minimum distance, $\beta$, is called the safety index or reliability index.

- X is often called the original space, U is the transformed space.

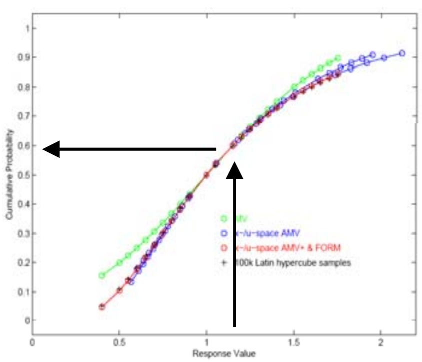
# MPP Search Methods




## Reliability Index Approach (RIA)

$$\text{minimize} \quad \mathbf{u}^T\mathbf{u}$$
$$\text{subject to} \quad G(\mathbf{u}) = \bar{z}$$

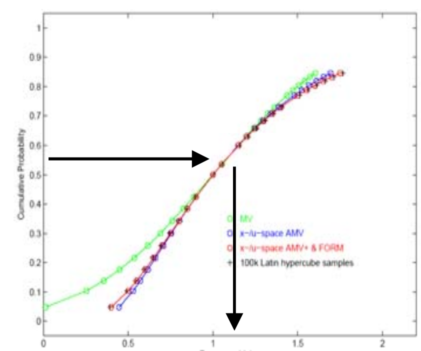
Find min dist to *G* level curve
Used for fwd map *z* → *p*/$\beta$




## Performance Measure Approach (PMA)

$$\text{minimize} \quad \pm G(\mathbf{u})$$
$$\text{subject to} \quad \mathbf{u}^T\mathbf{u} = \bar{\beta}^2$$

Find min *G* at $\beta$ radius
Used for inv map *p*/$\beta$ → *z*




Nataf x → u:

$$\Phi(z_i) = F(x_i)$$
$$\mathbf{z} = \mathbf{L}\mathbf{u}$$

# Local Reliability Example

- **Compare sampling methods to local reliability methods (modify `dakota_rosenbrock_nond.in` or see `examples/methods/dakota_uq_reliability.in`)**

# Example Input/Output: Reliability

```
strategy,
      single_method graphics

method,
      nond_local_reliability
      mpp_search no_approx
       response_levels = .4 .5 .55 .6 .65 .7  .75 .8  .85
.9 1. 1.05 1.15 1.2 1.25 1.3 1.35 1.4 1.5 1.55 1.6 1.65
1.7 1.75

variables,
      lognormal_uncertain = 2
         means             =  1.  1
         std_deviations    =  0.5 0.5
         descriptors       =  'TF1ln'   'TF2ln'
         uncertain_correlation_matrix =      1   0.3
                                             0.3  1

interface,
            system asynch
              analysis_driver = 'log_ratio'

responses,
            num_response_functions = 1
            numerical_gradients
              method_source dakota
              interval_type central
              fd_gradient_step_size = 1.e-4
```

Reliability Input (examples/methods/dakota_uq_reliability.in)

```
<<<<< Iterator nond_local_reliability completed.
<<<<< Function evaluation summary: 657 total (561 new, 96 duplicate)
-----------------------------------------------------------------
Cumulative Distribution Function (CDF) for response_fn_1:
     Response Level  Probability Level  Reliability Index  General Rel Index
     --------------  -----------------  -----------------  -----------------
   4.0000000000e-01   4.7624085962e-02   1.6683404020e+00   1.6683404020e+00
   5.0000000000e-01   1.0346525475e-01   1.2620507942e+00   1.2620507942e+00
   5.5000000000e-01   1.3818404972e-01   1.0885143628e+00   1.0885143628e+00
   6.0000000000e-01   1.7616275822e-01   9.3008801339e-01   9.3008801339e-01
   6.5000000000e-01   2.1641741368e-01   7.8434989944e-01   7.8434989944e-01
   7.0000000000e-01   2.5803428381e-01   6.4941748143e-01   6.4941748143e-01
   7.5000000000e-01   3.0020938124e-01   5.2379840557e-01   5.2379840557e-01
   8.0000000000e-01   3.4226491013e-01   4.0628960782e-01   4.0628960782e-01
   8.5000000000e-01   3.8365052982e-01   2.9590705956e-01   2.9590705956e-01
   9.0000000000e-01   4.2393548232e-01   1.9183562480e-01   1.9183562480e-01
   1.0000000000e+00   5.0000000000e-01   4.7212046773e-12   4.7212177145e-12
   ...
```

Reliability Output

# Interval Analysis

- Very simple approach, often used to model epistemic uncertainties (e.g. lack of knowledge uncertainties where you can only specify an upper and lower bound on the value of a variable, not a distribution)
- Transform the problem into two optimizations:
  - Find min over bounded domain
  - Find max over bounded domain
- Customized approach uses an adaptive, Gaussian process and an optimization method which exploits properties of the GP to place points where they most “inform” the optimization

# Interval Analysis Example

- **Run `test/dakota_uq_cantilever_interval.in`**
- **Look at results from method nond_global_interval_est lhs vs. nond_global_interval_est ego**

```
INTERVAL ANALYSIS based on LHS
<<<<< Function evaluation summary: 100 total (100 new, 0 duplicate)
-----------------------------------------------------------------
Min and Max estimated values for each response function:
weight:  Min = 1.6910009299e+00  Max = 8.8008981270e+01
stress:  Min = -9.7304216538e-01  Max = 9.0566017044e+00
displ:  Min = -9.9149236815e-01  Max = 2.1569369595e+01
-----------------------------------------------------------------

INTERVAL ANALYSIS based on EGO
<<<<< Function evaluation summary: 87 total (86 new, 1 duplicate)
-----------------------------------------------------------------
Min and Max estimated values for each response function:
weight:  Min = 1.0000169352e+00  Max = 9.9999830649e+01
stress:  Min = -9.7749982853e-01  Max = 2.1499428450e+01
displ:  Min = -9.9315677360e-01  Max = 6.7429714485e+01
-----------------------------------------------------------------
```

# Polynomial Chaos Expansions (PCE)

- Represent a stochastic process (the uncertain output f(X)) as a spectral expansion in terms of suitable orthonormal eigenfunctions with weights associated with a particular density

$$f(X) \approx R = \sum_{j=0}^{P} a_j H_j(\xi)$$

- The uncertain output f(X) is approximated by finite dimensional series based on unit Gaussian distributions
- In the expansion, the H terms are Hermite polynomials (multi-dimensional orthogonal polynomials), the $\xi$ are standard normal random variables, and the coefficients $a_j$ are deterministic but unknown.
- The job of PCE is to determine the coefficients $a_j$. Then, one has an approximation that can be sampled many times to calculate desired statistics

# Example Input/Output: PCE

```
strategy,
      single_method

method,
            nond_polynomial_chaos
              expansion_order    = 4 2
              quadrature_order   = 5 3
              samples = 10000
              seed    = 12347
              response_levels =
              .1 1. 50. 100.
              500. 1000.

variables,
            uniform_uncertain = 2
              lower_bounds     = -2.  -2.
              upper_bounds     =  2.   2.
              descriptors      = 'x1' 'x2'

interface,
            direct
            analysis_driver = 'rosenbrock'
responses,
            num_response_functions = 1
            no_gradients
            no_hessians
```

PCE Input (examples/tutorial/dakota_pce.in)

```
<<<<< Function evaluation summary: 15 total (15 new, 0 duplicate)
Polynomial Chaos coefficients for response_fn_1:
      coefficient   u1   u2
      ----------- ---- ----
  4.5566666667e+02   P0   P0
 -4.0000000000e+00   P1   P0
  9.1695238095e+02   P2   P0
...
Statistics derived analytically from polynomial expansion:
Moments for each response function:
response_fn_1:  Mean = 4.5566666667e+02  Std. Dev. = 6.0656024184e+02  Coeff.
of Variation = 1.3311490311e+00

Sensitivities for each response function evaluated at uncertain variable means:
response_fn_1:   [ -2.0000000000e+00  2.4505397711e-13 ]

Statistics based on 10000 samples performed on polynomial expansion:
Probabilities for each response function:
Cumulative Distribution Function (CDF) for response_fn_1:
    Response Level  Probability Level  Reliability Index  General Rel Index
    --------------  -----------------  -----------------  -----------------
  1.0000000000e-01   1.9000000000e-03
  1.0000000000e+00   1.3600000000e-02
  5.0000000000e+01   2.4390000000e-01 ...
```

PCE Output