Oregon State University

# Design Under Uncertainty: Methods

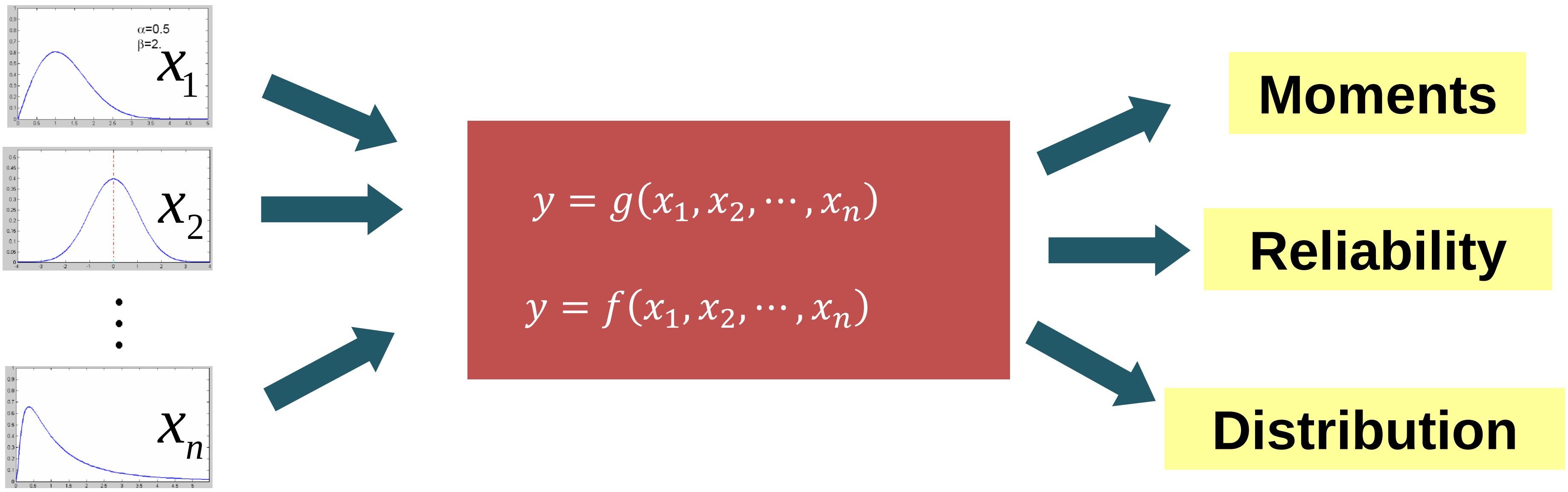
ME 615 Spring 2020


Dr. Chris Hoyle

MIME

# Uncertainty Quantification

Oregon State University
College of Engineering



# Uncertainty Quantification (UQ)



- Uncertainty Quantification (UQ) methodology:
  - Uncertainty in design inputs (I) creates uncertainty in the performance response (O).
  - If we can quantify the performance response uncertainty distribution, we can compare designs using utility theory or we can determine the probability of meeting requirements.
- Uncertainty Quantification (UQ) is fundamentally a process of computing a multidimensional integral for an arbitrary number of random dimension (X) given a system model.

$$\int_{\Omega} \cdots \int f_{\mathbf{X}}(\mathbf{x}) \mathrm{d}\mathbf{x}$$

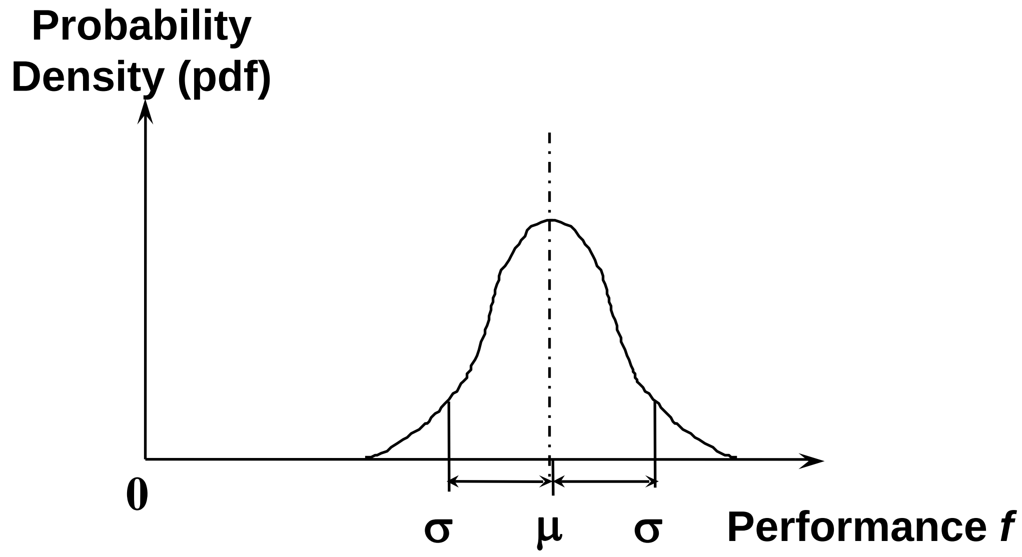
# Objectives vs. Constraints



- Objectives are expressed as:
  - **Minimize**: less is better
  - **Maximize**: more is better
  - **Target**: meet a given target
- Constraints are expressed as:
  - Must not **exceed** a certain value
  - Must be **below** a certain value
  - Must **equal** a certain value

# Objectives and Constraints

## Objectives

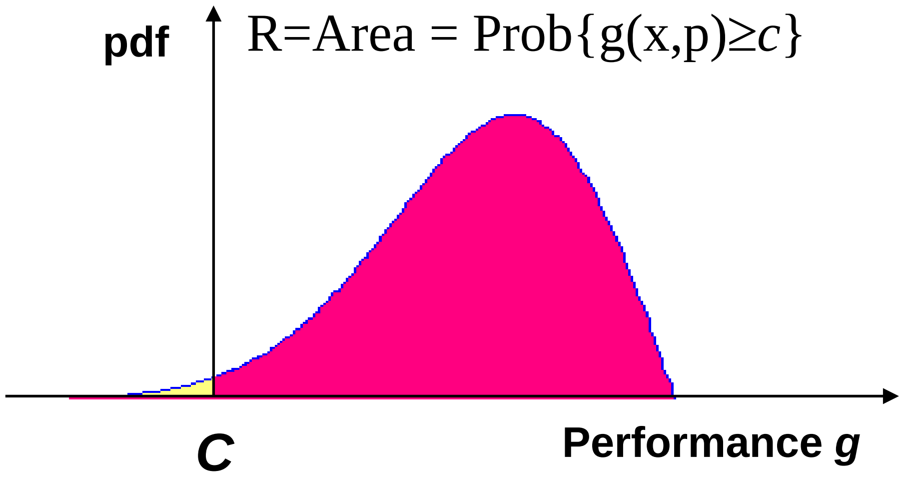



*Look at entire distribution*

**s.t.** $x \in X$

Considering the effect of variations without eliminating the causes

## Constraints




**Satisfy**

$$R = P\{g(\mathbf{x},\mathbf{p}) \geq c\} \geq R_0$$

**Limit State**

To assure proper levels of "safety" for the system designed

# Uncertainty Quantification for Optimization



- Simulation based method
  - Monte Carlo Simulation
  - Importance sampling, stratified sampling, adaptive sampling,…
- Local expansion based methods (Perturbation method)
  - Taylor series method (Mean Value First Order Second Moment-MVFOSM)
- MPP (Most probable point based methods)
  - FORM (first order reliability method)
  - SORM (second order reliability method)
- Numerical integration based method
  - Full factorial numerical integration (tensor product quadrature)
  - Dimension reduction method
- Functional expansion based method
  - Neumann expansion method
  - Polynomial chaos expansion method

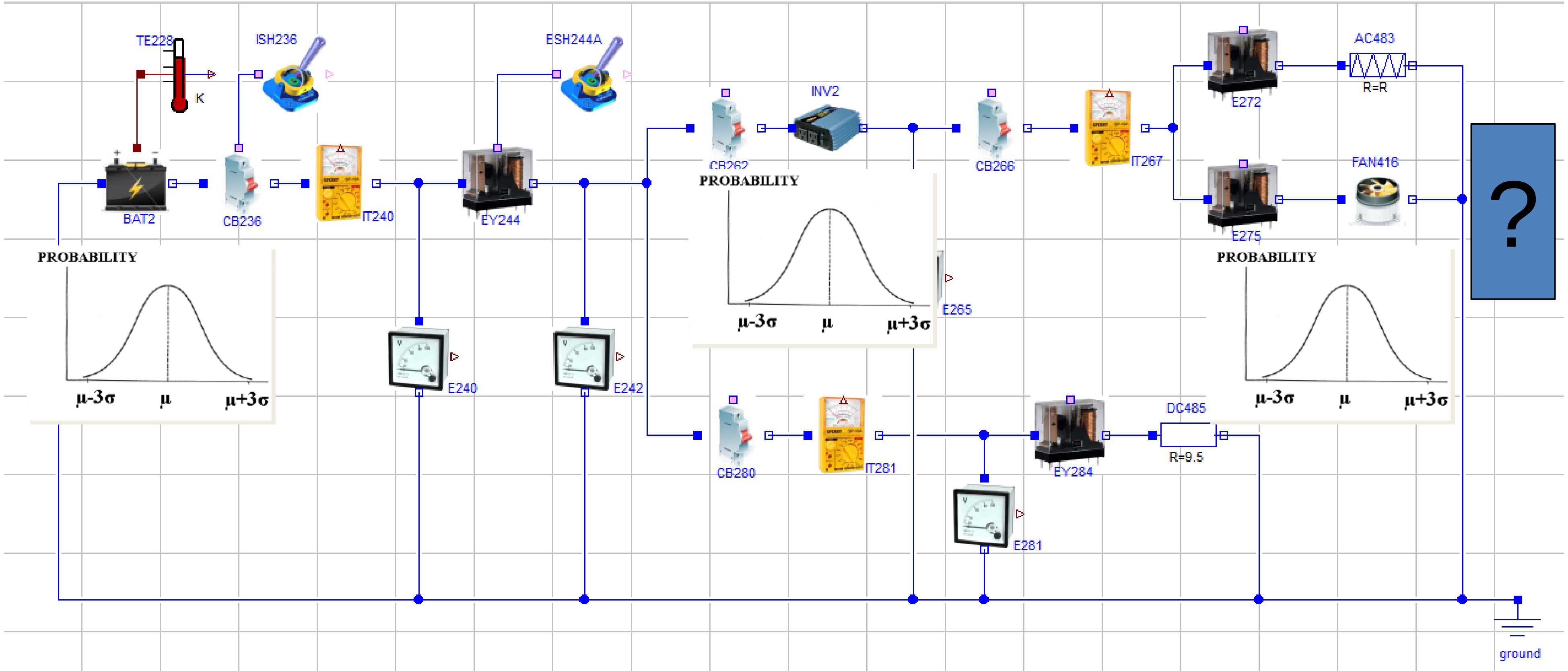
*These are all classified as methods for "black box" uncertainty quantification:*

- *They treat the underlying simulation model as a "black box"*

# Example Problem

Oregon State University College of Engineering

- An electrical power system with 3 sources of uncertainty:
  - Battery voltage
  - Inverter resistance
  - Fan resistance



# Question



- *What is the uncertainty in the fan speed (output) given the 3 sources of input uncertainty?*

# Methods

Uncertainty Quantification

# Monte Carlo Simulation



- A sample based approach to calculating the multi-dimensional integral.
  - Randomly (or pseudo-randomly) draw samples from the distributions representing uncertain quantities:
    - Design Variables (*X*)
    - Model Parameters (*P*)
  - Simulate your model using the set samples sequentially (or in parallel).
  - We can then numerically calculate quantities of interest (rather than computing integrals analytically)

# Monte Carlo Simulation Example

Oregon State University
College of Engineering

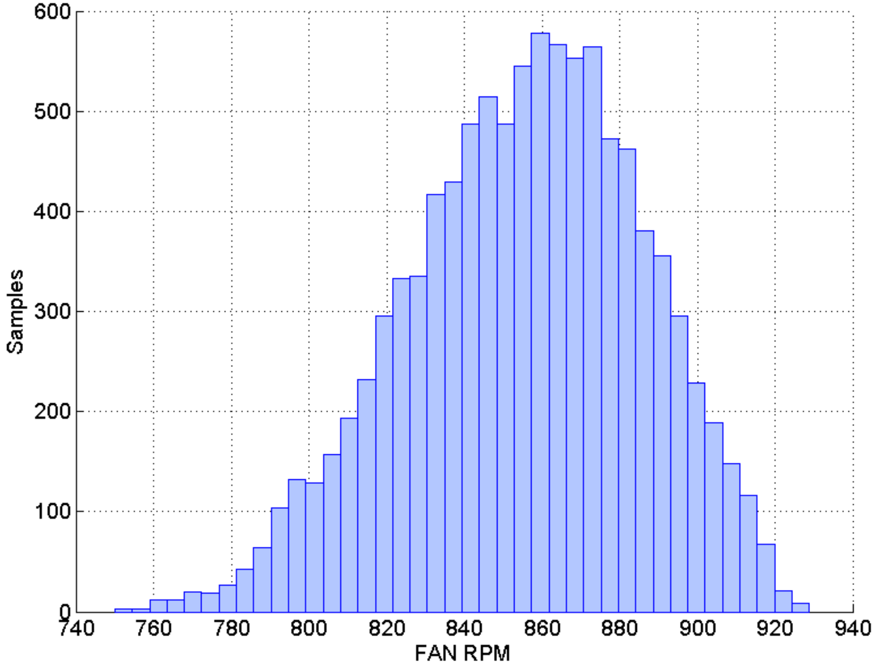
- Results of a Monte Carlo Simulation:

Uncertain inputs

- Battery voltage (~*N*)
- Inverter resistance (~*N*)
- Fan resistance (~*N*)

System Model

Output Distribution



# Monte Carlo Simulation for computing Moments

Oregon State University College of Engineering

- Generate random samples $X_i$ for X (normrnd, betarnd, weibrnd,… in Matlab)
- Record the model output for each random sample and plot a **Histogram**
- Calculate **sample** moments for the function f(**x**) or g(**x**):
  - Mean: $\hat{\mu}_f = \frac{1}{N}\sum_{i=1}^{N} f(x_i)$
  - Variance: $\hat{\sigma}_f^2 = \frac{1}{N-1}\sum_{i=1}^{N}\left(f(x_i) - \hat{\mu}_f\right)^2$
  - Skewness: $\hat{s}_f = \frac{\sum_{i=1}^{N}\left(f(x_i)-\hat{\mu}_f\right)^3}{\hat{\sigma}_f^3}$
  - Kurtosis: $\hat{k}_f = \frac{\sum_{i=1}^{N}\left(f(x_i)-\hat{\mu}_f\right)^4}{\hat{\sigma}_f^4}$

# Monte Carlo Algorithm for computing Moments

1. Define the random model inputs.
2. Generate a set of inputs randomly from a probability distribution over the domain.
   - *In Matlab, you can use normrnd, lognrnd , betarnd, unifrnd*
3. Perform a deterministic computation using your system model on this set of input values.
   - *This means you will need a method to send the input values generated by Matlab to your system model*
4. Record the model response of interest.
5. Repeat 2-4 *N* times, where *N* is the number of samples desired (usually on the order of $10^3$-$10^6$)
6. Compute sample moments:

$$\hat{\mu}_G = \frac{1}{N}\sum_{i=1}^{N} G(\mathbf{X}_\mathbf{i}) \qquad \hat{\sigma}_G^2 = \frac{1}{N-1}\sum_{i=1}^{N}\left(G(\mathbf{x}_\mathbf{i}) - \hat{\mu}_G\right)^2 \qquad \text{etc}$$

# Monte Carlo Simulation for calculating a Probability



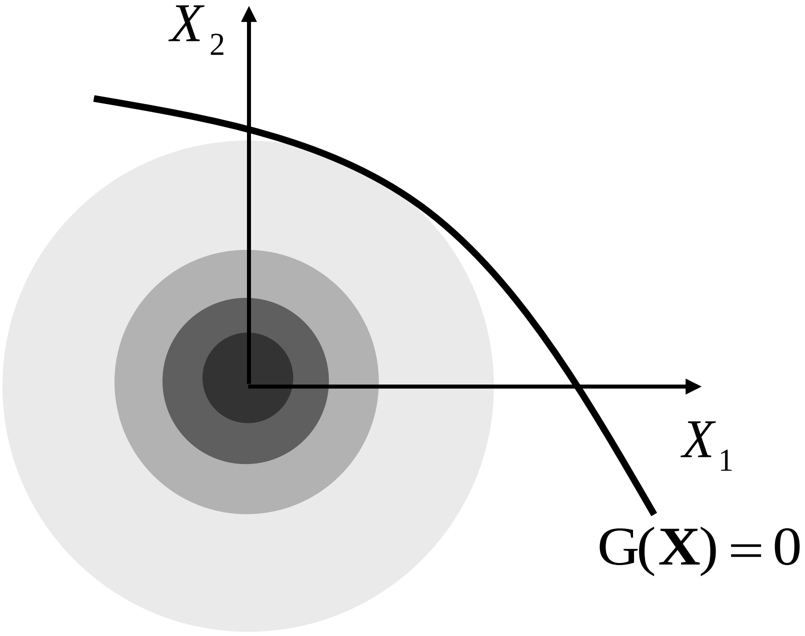
- Generate random samples $X_i$ for X (normrnd, betarnd, weibrnd,… in Matlab)
- Compute $G(X_i)$ and $I[G(X_i)<0]$
- Calculate probability of failure as follows:

$$P_f = \Pr[G(\mathbf{X}) \le 0] = \int_{G(\mathbf{X})\le 0} f(\mathbf{X})d\mathbf{X}$$

$$P_f = \int I[G(\mathbf{X}) \le 0] f(\mathbf{X})d\mathbf{X}$$

$$P_f \approx \frac{1}{N}\sum_{i=1}^{N} I[G(\mathbf{X}_i) \le 0]$$

$I[\bullet]$ : Indicator function



# Monte Carlo Algorithm for computing a probability

1. Define the random model inputs.
2. Generate a set of inputs randomly from a probability distribution over the domain.
   - In Matlab, you can use normrnd, lognrnd , betarnd, unifrnd
3. Perform a deterministic computation using your system model on this set of input values.
   - This means you will need a method to send the input values generated by Matlab to your system model
4. Check if simulated value meets the requirement.
   - Set I= 1 if it meets requirement
   - Set I = 0 If it does not meet requirement.
5. Repeat 2-4 *N* times, where *N* is the number of samples desired (usually on the order of $10^3$-$10^6$)
6. Sum I and compute probability of meeting requirement as
   - **sum I/N**

# Monte Carlo Simulation

Oregon State University
College of Engineering

- Characteristics of the MCS Method:
  - – Can handle any parametric or non-parametric representation of input uncertainty.
  - – The output uncertainty is not limited to a parametric distribution.
  - – Straightforward implementation.
  - – Expense not a function of number of input variables.
- Limitations of the MCS Method:
  - Requires much sampling, even with advanced sampling methods.
  - – Difficult to estimate the number of MCS samples needed *a priori*.

# Monte Carlo Simulation

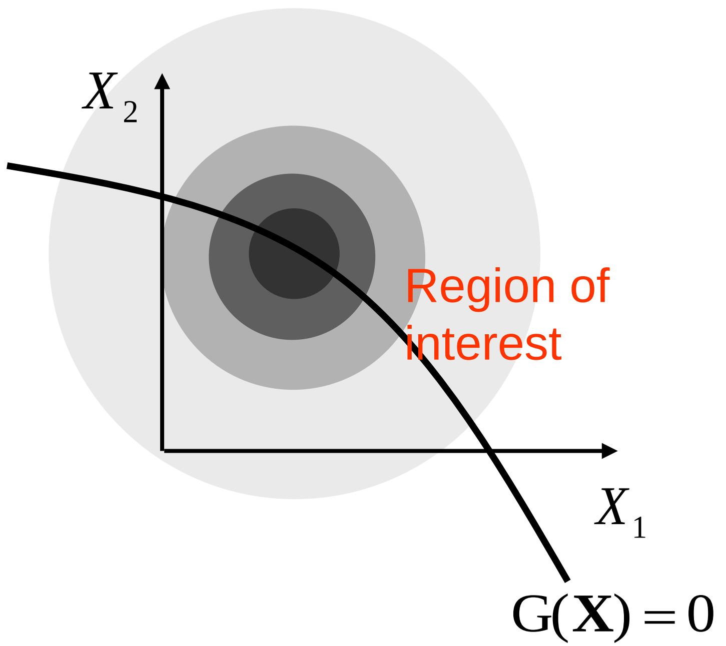
Oregon State University
College of Engineering

- Large sample required ($\sim 10^{-3}\ P_f$)
- Variance in result
- Importance sampling, stratified sampling, antithetic variants,…

$$P_f = \int I\left[G(\mathbf{X}) \le 0\right] f(\mathbf{X}) d\mathbf{X}$$

$$P_f = \int I\left[G(\mathbf{X}) \le 0\right] \frac{f(\mathbf{X})}{h(\mathbf{X})} h(\mathbf{X}) d\mathbf{X}$$

$$P_f \approx \frac{1}{N} \sum_i I\left[G(\mathbf{X}_i) \le 0\right] \frac{f(\mathbf{X}_i)}{h(\mathbf{X}_i)}$$




h(x): importance sampling density function

Choosing appropriate h(x) is often tricky.