Uncertainty in Geo-science: A Workshop on Hazard Analysis

Hands on Module

H. Aghakhani, Z. Cao, P. Shekhar

University at Buffalo

15-17 March , 2016

OUTLINE

Problem Statement

Study of uncertainty in input parameters of complex computer models:

- **Intrusive**: change the original governing equation.
  - Polynomial Chaos / Stochastic Galerkin.
  - Perturbation methods
  - ...
- **Non-intrusive**: *do not* change the original governing equation.
  - Monte Carlo & LHS
  - Important sampling Methods
  - Non-intrusive spectral projection (NISP)
  - ...

Test Problem – uncertainty in plume model BENT

- Bent – One-dimensional, numerical eruption column or plume models (Bursik '01, Pouget et al '16)
- Provides estimate of the amount of ash emplaced into the atmosphere at an estimated plume height
- Inputs to model include vent radius, vent velocity, water content, temperature

Latin Hypercube Sampling (LHS)

Generating a sample set of uncertain values from equally probable intervals of the probability density function.

1. Select the pdf
2. Select the number of samples
3. Divide the pdf function into equal probability intervals.
4. Generate random samples on each interval.
5. *Starting point for other methods – Gaussian Processes, BLM etc.*

Advantage:

- Fewer samples require for convergence compared to MC
- No need to change the original solver (like any other non-intrusive method)

Instructions for LHS script

Scripts are available for downloads and are easily customized to run problems other than the demo problems. To run the LHS_UQ you just need to do the following:

- Open a python interpreter by typing python on your terminal.
- To load the LHS_UQ function type: from LHS import LHS_UQ
- To run the function you need to provide the following arguments : LHS_UQ(num_samples , min_wat_cont , range_ wat_cont , min_temp ,range_temp):
    1. number of samples
    2. minimum of water content
    3. range of water content
    4. minimum of temperature
    5. range of temperature

Polynomial Chaos Quadrature (PCQ)

The basic idea comes from projection theory that each function can be written as an expansion of a series of orthogonal functions:

$$h(\eta) = \sum_{i=1}^{\infty} a_i \Psi_i(\eta) \tag{1}$$

So any uncertain parameter in the model can be expressed as above. The inner product of $h(\eta)$ and jth basis function ψ_j is $< h(\eta), \Psi_j(\eta) > \equiv \int_{-\infty}^{\infty} h(\eta)\Psi_j(\eta)\rho(\eta)d\eta$, $\rho(x)$ is a weight function

$$< h(\eta), \Psi_j(\eta) > = < \sum_i a_i \Psi_i(\eta), \Psi_j(\eta) > \tag{2}$$

Due to orthogonality of basis:

$$< h(\eta), \Psi_j(\eta) > = a_i < \Psi_i(\eta), \Psi_j(\eta) > \tag{3}$$

$$a_i = \frac{< h(\eta), \Psi_j(\eta) >}{< \Psi_i(\eta), \Psi_i(\eta) >} \tag{4}$$

Polynomial Chaos Quadrature (PCQ)

The integration is then approximated using numerical quadrature.
By definition, inner product with respect to certain distribution is:

$$
\begin{aligned}
< h(\eta), \Psi_j(\eta) > &= \int_{-\infty}^{\infty} h(\eta)\Psi_j(\eta)\rho(\eta)d\eta \\
&\approx \sum_{i=1}^{N_q} h(\eta_i)\Psi_j(\eta_i)\rho(\eta_i)w_i
\end{aligned} \tag{5}
$$

$$
< \Psi_j(\eta), \Psi_j(\eta) > \approx \sum_{k=1}^{N_q} \Psi_j(\eta_k)\Psi_j(\eta_k)\rho(\eta_i)w_k \tag{6}
$$

Polynomial Chaos Quadrature (PCQ)

When we have multiple variable i.e. $h(\eta, \phi, ...)$ we can form basis functions by a tensor product. While reasonable for small dimensions the number of basis functions and quadrature points rises as $(N_q)^d$ – the famous *curse of dimensionality*.
Fixes include

- Sparse Grid
- Conjugate Unscented Transform (CUT)

Polynomial Chaos Quadrature (PCQ)

For some cases the orthogonality of basis can not be guaranteed, then the following system of equations (j=1..N) need to be solved for evaluating coefficients a_i.

$$\begin{aligned}
< \Psi_1(\eta), \Psi_1(\eta) > a_1 + ... + < \Psi_N(\eta), \Psi_1(\eta) > a_N &= < h(\eta), \Psi_1(\eta) > \\
< \Psi_1(\eta), \Psi_2(\eta) > a_1 + ... + < \Psi_N(\eta), \Psi_2(\eta) > a_N &= < h(\eta), \Psi_2(\eta) > \\
< \Psi_1(\eta), \Psi_3(\eta) > a_1 + ... + < \Psi_N(\eta), \Psi_3(\eta) > a_N &= < h(\eta), \Psi_3(\eta) > \\
... &= ...
\end{aligned}$$

Instructions for PCQ script (quick start)

To run PCQ_UQ you just need to do the following:

- Open a python interpreter by typing python on your terminal.
- To load the PCQ_UQ function type: from PCQ import PCQ_UQ
- To run the function you need to provide the following arguments by typing:
  PCQ_UQ (num_samples, mean_rv, range_rv, v_dis, v_name, rv_value, alf, beta):
  1. number of samples
  2. mean of random variables
  3. range of random variables
  4. name of distribution for each random variables
  5. name of two random parameters
  6. default value of two random parameters (222 for speed, 333 for radius, 0.33 for water fraction, 1111 for temperature)
  7. alf and beta are parameters for "Beta" distribution

Instructions for PCQ script (quick start)

For example:
PCQ_UQ ([2,2], [200, 200], [100, 100], ["Uniform", "Uniform"], ['radius', 'speed'], [222, 333], 0.5, 0.5)
For this example, the input parameters are:

1. 2 Gaussian quadrature points for each random input parameter.
2. Mean for each RV is 200
3. Range of each RV is 100
4. Assume "Uniform" distribution for both RVs
5. Select radius and speed as input random parameters
6. The default value
7. α and β is useless in this case

Instructions for PCQ script (more details)

The basic procedure of running the PCQ script:

- Generate sample points and quadrature weight
- Run simulation with sampled parameters
- Parse output file and extract desired properties
- Prepare for coefficient computing
- compute coefficient by PCQ (or by solving system of equations)
- Post process

Instructions for PCQ script(more details)

Before starting, you have to determine:

- **Random parameters**
  The recommended input parameters are: eruption speed, vent radius, mass fraction of water in erupted material, temperaure of erupted material.
- **Underlying distribution**
  For each random variable (given by dis_i in the script)
- **Mean and range**
  For each random parameters. (given by mean_xxx and range_xxx in the script)
- **Number of smaple points**
  For each random variable

Instructions for PCQ script(more details)

- **Generate sample points and quadrature weight**:
  - The Gaussian Quadrature points generator should be called accordingly based on distribution that you selected: Hermite for Gaussian, Legendre for Uniform, Leguerre for Gamma, and Jacobi for Beta (modify smplingx=...)
  - Transfer and scale sample points
- **Run simulations with sampled parameters**:
  This will be done automatically.
- **Parse output file and extract desired properties**:
  The output is eruption mass flux and plume height. These two properties will be extracted with parse script automatically.
- **Prepare for coefficient computing**:
  Basically, just re-organize array into matrix of desired dimension.

Instructions for PCQ script(more details)

- **compute coefficient by PCQ (or by solving system of equations)**:
  When the orthogonality of the basis is guaranteed, use PCQ or more general method (sovling system of equations). Otherwise, only the more general method could be used. You can switch from these two method by commentting off/on.
- **Post process**:
  - Plot plume height (or mass flux) as a function of random parameters. num_plot_xxx gives number of re-sample points for plot.
  - Plot histogram of plume height (or mass flux) for specific distribution of input parameters.
- **Usage of subroutines is given very explicitly in the script**

Thank You . . .