Exam Algorithms for Uncertainty Quantification (T. Neckel, I. Farcas) ST17	Page 1/11
Last name, first name, student ID:	Signature:

General Instructions

Material:

You may only use one hand-written sheet of paper (size A4, on both pages).

Any other material including electronic devices of any kind is forbidden.

Use only the exam paper that was handed out to solve the exercises. In case the space on a page is not enough, mark that you continue with your solution and use the reverse side of the preceding page. For additional notes and sketches, you can obtain additional exam sheets.

Do not use pencil, or red or green ink.

General hint:

Often, exercises b), c), etc. can be solved without the results from the previous exercise a); if you are stuck with exercise a), then don't immediately skip exercises b), c), etc.

Working time:

75 minutes + 5 minutes reading time.

Please switch off your cell phones!

Good luck!

Proposed Solution

1	2	3	4	5	$\sum$
/11	/11	/10	/6	/4	$/\approx 42$

Last name, first name, student ID:

1 Sampling Methods (≈ 3 + 2 + 6 = 11 points)

In this problem set, the focus is on sampling methods.

(a) You are given a set of five samples $X = \{1.80, 1.66, 1.82, 1.72, 1.60\}$ representing five persons' heights in meters. Moreover, you are told that the mean value is 1.72 and the variance is 0.00688. Are these estimates coming from biased or unbiased estimators? Show your calculations.

The unbiased mean estimator is

$$\bar{X} = \frac{\sum_{i=1}^{5} X_i}{5},$$

whereas the unbiased variance estimator is

$$S^2 = \frac{1}{4}\sum_{i=1}^{5}(X_i - \bar{X})^2.$$

Plugging in the numbers, we get $\bar{X} = 1.72$, $S^2 = 0.0086$. Therefore, the mean estimate comes from an unbiased estimator, whereas the variance estimate is biased (hint: it comes from the samples variance estimator, where the division is made by 5, not 4).

weighting (total: 3 points):

- 0.5 computation of $\bar{X}$ + 0.5 unbiased mean
- 1.0 computation of variance
- 1.0 biased variance

(b) Evaluate $\int_{-\infty}^{\infty}(x + x^2)\exp(-x^2/2)dx$

$$\int_{-\infty}^{\infty}(x + x^2)\exp(-x^2/2)dx =$$
$$\int_{-\infty}^{\infty} x\exp(-x^2/2)dx + \int_{-\infty}^{\infty} x^2\exp(-x^2/2)dx =$$
$$\sqrt{2\pi}\int_{-\infty}^{\infty} x\frac{\exp(-x^2/2)}{\sqrt{2\pi}}dx + \sqrt{2\pi}\int_{-\infty}^{\infty} x^2\frac{\exp(-x^2/2)}{\sqrt{2\pi}}dx =$$
$$\sqrt{2\pi},$$

because, if $X \sim \mathcal{N}(0,1)$, then $\int_{-\infty}^{\infty} x\frac{\exp(-x^2/2)}{\sqrt{2\pi}}dx = \mathbb{E}[X] = 0$ and $\int_{-\infty}^{\infty} x^2\frac{\exp(-x^2/2)}{\sqrt{2\pi}}dx = \mathrm{Var}[X] = 1$.

weighting (total: 2 points):

- 1.0 for splitting the integral + normalisation
- 1.0 observation definition of expectation & variance of normal distr.

(c) Briefly explain what standard and quasi-Monte Carlo methods are, including two commonalities and two differences

Standard Monte Carlo: basic method for uncertainty propagation/numerical integration etc., based on (pseudo-)random samples, slow convergence, complexity independent of the dimension, embarrassingly parallel etc.

Quasi-Monte Carlo: more advanced sampling technique, based on deterministic samples (for example, low-discrepancy sequences), faster convergence than standard algorithm, but the complexity depends on the dimensionality

Commonalities: both are sampling methods; both can be used to evaluate integrals numerically or do uncertainty propagation etc.

Differences: standard Monte Carlo employs (pseudo-)random samples, quasi-Monte Carlo deterministic sequences; standard Monte Carlo works for arbitrary domains, quasi-Monte Carlo is defined in the unit hypercube etc.

weighting (total: 6 points):

- 1.0 explanation standard MC
- 1.0 explanation QMC
- 2×1.0 for commonalities
- 2×1.0 for differences

2 Non-intrusive Approaches (≈ 2 + 2 + 3 + 4 = 11 points)

(a) Which class of orthogonal polynomials is related to uniform distributions in the context of UQ? And which to normal distributions?

uniform: Legendre polynomials
normal: Hermite polynomials (probabilist scaling)

weighting (total: 2 points (+1 bonus point)):

- 1.0 uniform + Legendre
- 1.0 normal + Hermite
- 1.0 bonus: probabilist vs. physicist

(b) Explain the curse of dimensionality. What approach can be used to circumvent or delay it?

The curse of dimensionality is related to the direct exponential contribution of the dimension N of the underlying stochastic space to the calculations that have to be made for a certain operation (quadrature, e.g.) in a tensor product sense.
One approach to delay that is via sparse grids which involve a logarithmic instead of linear term and a power N-1 instead of N.

weighting (total: 2 points):

- 1.0 explanation "exponential"
- 1.0 sparse grids idea (or similar)

(c) Consider the *Ishigami* function $f : [-\pi, \pi]^3 \to \mathbb{R}$

$$f(\boldsymbol{x}) = \sin(x_1) + a \sin^2(x_2) + b x_3^4 \sin(x_1),$$

where $a, b \in \mathbb{R}$. We take $a = 7, b = 0.1, x_2 = x_3 = \pi/2$. Assume that $x_1 \sim \mathcal{U}(-\pi, \pi)$ and let $y(x_1)$ denote the stochastic version of $f(\boldsymbol{x})$. The polynomial chaos expansion of $y(x_1)$ reads

$$y(x_1) = \sum_{n=0}^{N-1} \hat{y}_n \Phi_n(x_1), \tag{1}$$

where

$$\hat{y}_n = \sum_{k=0}^{K-1} y(\zeta_k) \Phi_n(\zeta_k) \omega_k,$$

where $\{\zeta_k, \omega_k\}_{k=0}^{K-1}$ are suitably chose quadrature nodes and weights.

You are given the procedure PSA (see next page).

Last name, first name, student ID:

1: **procedure** PSA(N, K)
2: generate K nodes and weights $\{\zeta_k, \omega_k\}_{k=0}^{K-1}$
3: generate N orthogonal polynomials $\{\Phi_n\}_{n=0}^{N-1}$
4: create a vector $\hat{y}$ to store the coefficients
5: **for** $n \leftarrow 0, N-1$ **do**
6: $\hat{y}_n \leftarrow 0$
7: **for** $k \leftarrow 0, K-1$ **do**
8: $\hat{y}_n \leftarrow \hat{y}_n + \text{IshigamiFunction}(\zeta_k)\Phi_n(\zeta_k)\omega_k$
9: **end for**
10: **end for**
11: **end procedure**

- Is the procedure PSA correctly implementing the pseudo-spectral approach?

  Yes :) It implements the two formulas above correctly. At the end, the polynomial chaos coefficients are correctly evaluated.

  weighting (total: 1 point):
  - 1.0 yes

- Would you implement PSA in your favorite programming language as given? Why? If you would implement it differently, sketch the modified procedure below.

  No, because the underlying model is redundantly evaluated for each polynomial chaos expansion coefficient. The updated version is

```
for k = 0 ... K - 1
    evaluate the IshigamiFunction at all nodes
    save the results in a vector y
end for

for n = 0 ... N - 1
    for k = 0 ... K -1
      proceed as in the given pseudo-code
      but instead of evaluating the model,
      use the vector y
    end for
end for
```

  weighting (total: 3 points):
  - 1.0 no + reason efficiency
  - 0.5 separate loop + 0.5 usage of vector y instead of function call

(d) Assume that initially, the propagation of uncertainty though $y(x_1)$ was performed via standard Monte Carlo sampling using 10^6 samples. The corresponding values for the mean and variance are $\mathbb{E}[y(x_1)]_{MCS} = 7.001$, $\text{Var}[y(x_1)]_{MCS} = 1.294$.

Now, someone who did not participate in the *Algorithms for Uncertainty Quantification* course tries to use the polynomial chaos expansion + the pseudo-spectral approach to propagate the uncertainty through $y(x_1)$. The following questions come up:

- Once you asses the coefficients from Eq. (1), how do you compute the mean and variance?
  Mean:
  $$\mathbb{E}[y(x_1)]_{PCE} = \hat{y}_0$$
  Variance:
  $$\text{Var}[y(x_1)]_{PCE} = \sum_{n=1}^{N-1} \hat{y}_n^2/\gamma_n$$
  Note: if Φ_n are not normalized, $\gamma_n = \mathbb{E}[\Phi_n^2(x_1)]$, else $\gamma_n \equiv 1$!!!

  weighting (total: 2 points (+ 1 bonus point):
  - 1.0 expectation=y_0
  - 1.0 correct formula for variance
  - 1.0 bonus point: specify normalisation constant

- Assuming that a Gauss-Legendre quadrature rule with $K = 16$ was used to evaluate $N = 6$ coefficients. The results are
  $$\hat{y}_0 = 7.000E00, \quad \hat{y}_1 = 4.890E-01, \quad \hat{y}_2 = -3.385E-15$$
  $$\hat{y}_3 = -1.502E-01, \quad \hat{y}_4 = -1.580E-15, \quad \hat{y}_5 = 9.079E-03$$
  Compute the mean and variance using the polynomial chaos coefficients. What do you observe when you compare these values with the Monte Carlo results? Why?
  The mean value is correct ($\mathbb{E}[y(x_1)]_{PCE} = \hat{y}_0 = 7.00$), but for the variance, if we just just square the coefficients starting from the second, we get $\text{Var}[y(x_1)]_{PCE} = \sum_{n=1}^{N-1} \hat{y}_n^2 = 0.2618$; the exact value of this computation is not relevant. What is relevant is to observe that just squaring the coefficients is not enough. Why? Because the polynomials are not normalized!!!

  weighting (total: 2 points):
  - 1.0 expectation correct (and why)
  - 1.0 variance correct (and why: argument values $y_i << 1$)

3 Intrusive Approaches (≈ 2 + 2 + 6 = 10 points)

(a) Name one advantage and one drawback of the stochastic Galerkin method.

advantage: more accurate solution (no quadrature error)
drawback: underlying code has to be changed (considerably)

weighting (total: 2 points):

- 1.0 advantage
- 1.0 drawback

(b) Consider a simplified version of the damped oscillator model problem

$$\begin{cases} \frac{d^2y}{dt^2}(t) + ky(t) = 1.0 \\ y(0) = y_0 \\ \frac{dy}{dt}(0) = y_1, \end{cases} \tag{2}$$

where k is uncertain and distributed as $k \sim \mathcal{N}(0, 4)$. Derive the polynomial chaos expansion of k.

We know that if $X \sim \mathcal{N}(0,1), Y \sim \mathcal{N}(\mu, \sigma^2)$, then $Y = \mu + \sigma X$. Therefore, $k = 2x$. Knowing that in any orthogonal polynomial sequence $\{\Phi_i\}_{i\geq 0}$, $\Phi_0 \equiv 1$ and $\Phi_1 = x$, we have $k = \hat{k}_1\Phi_1(x), \quad \hat{k}_1 = 2.$

weighting (total: 2 points):

- 1.0 transformation
- 1.0 PCE as one term

(c) The stochastic solution $y(t, u_s)$ of the given problem (2) has to be approximated via a polynomial chaos approximation

$$y(t, u_s) = \sum_{n=0}^{N-1} \hat{y}_n(t)\Phi_n(u_s),$$

where the basis polynomials are orthonormal, i.e. $\mathbb{E}[\Phi_n\Phi_m] = \delta_{nm}$. Use the stochastic Galerkin approach to derive the system of equations needed to compute the coefficients $\hat{y}_n(t)$. *Hint: If you did not solve (b), take the polynomial chaos approximation of k to be $k = \hat{k}_0\Phi_0(u_s) + \hat{k}_1\Phi_1(u_s)$.*

$$\frac{d^2y}{dt^2}(t, u_s) + ky(t, u_s) = 1$$

$$\Leftrightarrow$$

$$\frac{d^2\left(\sum_{n=0}^{N-1} \hat{y}_n(t)\Phi_i(u_s)\right)}{dt^2} + 2\Phi_1(u_s)\left(\sum_{n=0}^{N-1} \hat{y}_n(t)\Phi_n(u_s)\right) = 1$$

$$\Leftrightarrow$$

$$\sum_{n=0}^{N-1} \frac{d^2\hat{y}_n(t)}{dt^2}\Phi_n(u_s) + 2\sum_{n=0}^{N-1} \hat{y}_n(t)\Phi_1(u_s)\Phi_n(u_s) = 1$$

$$\Leftrightarrow$$

$$\sum_{n=0}^{N-1} \frac{d^2\hat{y}_n(t)}{dt^2}\langle\Phi_n, \Phi_m\rangle + 2\sum_{n=0}^{N-1} \hat{y}_n(t)\langle\Phi_1\Phi_n, \Phi_m\rangle = \langle 1, \Phi_m\rangle$$

$$\Leftrightarrow$$

$$\frac{d^2\hat{y}_n(t)}{dt^2} + 2\sum_{n=0}^{N-1} \hat{y}_n(t)\Phi_{1nm} = \delta_{0m},$$

where $\langle\Phi_1\Phi_n, \Phi_m\rangle = \Phi_{1nm}$

weighting (total: 6 points):

- 1st trafo of equ.: 0.5 plug in PCE + 0.5 for k expression
- 2nd trafo: 1.0 switch sum & derivative

- 3rd trafo: 1.0 multipl. w. polynomial + 1.0 expectation
- last trafo: 1.0 orthonormality (left) + 1.0 for $1 = \Phi_0$

4 Global Sensitivity Analysis & Random Fields($\approx$ 1 + 1 + 2 + 2 = 6 points)

(a) What is the main goal of global Sensitivity Analysis?

The main goal is to identify the importance or contribution of each uncertain parameter to the overall UQ output

weighting (total: 1 point):

- 1.0 correct goal: importance

(b) What technique allows to do a global Sensitivity Analysis in the context of UQ?

Sobol indices (via Monte Carlo or via generalised polynomial chaos (gPC) computed from the coefficients).

weighting (total: 1 point):

- 1.0 Sobol indices

(c) What is a random field?

A random field is a collection of random variables indexed by elements x in some topological space ($\mathbb{R}^3$, e.g.).

weighting (total: 2 points):

- 1.0 "collection"
- 1.0 indexing via some space

(d) If you encounter a problem involving a random field and you have to formulate the problem as a classical forward UQ problem, what option do you have?

We can formulate the problem as a classical UQ problem by approximating the random field with finitely many terms. The Karhunen-Loeve expansion is the standard way to do so.

weighting (total: 2 points):

- 1.0 approximation
- 1.0 method to approximate (KL)

5 Software for UQ (≈ 4 points)

You are given a forward model with the following properties:

- a single run takes about 50 seconds.
- You know what phenomena the given code models, but you can use it only as a legacy code.
- 18 of its input parameters are uncertain.

What UQ methodology and which library/framework/toolkit would you use for the propagation of uncertainty? Why?

Given the relatively high stochastic dimensionality (18 uncertain inputs $\Rightarrow$ stochastic dimensionality = 18) and the rel. low runtime/simulation, a sampling-based method (standard Monte Carlo or variants) would be fitted for this kind of problem. Software-wise, this is available e.g. in `chaospy` or `UQtk` or even `Dakota`.

Alternative: Do a global Sensitivity Analysis on a mid-size resolution and reduce the stochastic dimension to 5-10. Then, sparse-grid techniques may represent a nice alternative.

weighting (total: 4 points):

- 1.0 correct methodology
- 1.0 argument 1 (dimension)
- 1.0 argument 2 (runtime)
- 1.0 software