Technische Universität München
Lehrstuhl für Informatik V
Dr. Tobias Neckel
M. Sc. Friedrich Menhorn

SS 2018

July 04, 2018

Algorithms for Uncertainty Quantification

Tutorial 10: Random fields in Uncertainty Quantification

In this worksheet, we focus on random fields in Uncertainty Quantification.

Covariance functions

In the lecture, we saw that the covariance function describes spatial/temporal covariance of a random field.

$$C(t, s) := \mathrm{cov}(X_t, X_s)$$

From the definition, we have that the covariance function is symmetric and positive semi-definite. Generally, a random field is given in terms of its mean and covariance functions. If these two quantities are available, one could "sample" the random field.

Assignment 1

Consider a two-dimensional Gaussian random field $\mathcal{G}$ defined on $[0, 1]^2$ having mean function $m(\boldsymbol{x}) = 0.1$ and covariance function $C(\boldsymbol{x}, \boldsymbol{y})$. To sample this random field, we can do the following

- discretize the domain $[0, 1]^2$ using an $N \times N$ grid. Put each discretization point in e.g. the middle of one discretization cell
- evaluate $m(\boldsymbol{x})$ and $C(\boldsymbol{x}, \boldsymbol{y})$ at all grid points, i.e. obtain mean vector $\hat{m} \in \mathbb{R}^{N^2}$ and the covariance matrix $\hat{C} \in \mathbb{R}^{N^2 \times N^2}$
- $\hat{C}$ is symmetric and positive semi-definite; perform a Cholesky decomposition to obtain $\hat{C} = EE^T$, where E is an upper triangular matrix

- obtain samples from $\mathcal{G}$ as $\mathcal{G}_i = \hat{m} + E\Psi$, where $\Psi \sim \mathcal{N}(\mathbf{0}_{N^2}, \boldsymbol{I}_{N^2})$, $\boldsymbol{I}_{N^2}$ being the identity matrix of size $N^2 \times N^2$

Considering $N = 40$ and $l = 1.0$, implement the above algorithm in a `python` program. As covariance function use the exponential $C_1(\boldsymbol{x}, \boldsymbol{y}) = \exp(-||\boldsymbol{x} - \boldsymbol{y}||_2/l), \quad l \in \mathbb{R}$ and squared exponential kernel $C_2(\boldsymbol{x}, \boldsymbol{y}) = \exp(-||\boldsymbol{x} - \boldsymbol{y}||_2^2/l^2), \quad l \in \mathbb{R}$. Generate three samples each and visualize them using `matplotlib`'s `imshow` function.
The following code snippet could be used to generate the discretization of $[0, 1]^2$ using an $N \times N$ grid and putting each discretization point in the middle of the cell

```
import numpy as np

...
# mesh size in x and y directions; in this case, N
mesh_size_x = N
mesh_size_y = N

# x and y coordinates in the middle of each Cartesian cell
x_coord = np.arange(1./(2*mesh_size_x), 1., 1./mesh_size_x)
y_coord = np.arange(1./(2*mesh_size_y), 1., 1./mesh_size_y)

# mesh_coord makes the evaluation of the cov function easier
mesh_coord = {}
for i in xrange(mesh_size_x):
        for j in xrange(mesh_size_y):
                mesh_coord[i*mesh_size_x + j] = x_coord[i],
                   y_coord[j]

mesh_coord = mesh_coord.values()
```

Hint: In the end, you will get a random vector of size N^2. To visualize the random field realization, transform this vector into an $N \times N$ matrix.

The Wiener process

In the lecture, we saw that the Brownian motion/Wiener process is defined as a stochastic process $\{W_t : t \in \mathbb{T}\}$ s.t.

- $W_0 = 0$

- W_t is continuous in t
- the increments $W_1 - W_0, W_2 - W_1, \ldots$ are (stochastically) independent
- $W_{t+u} - W_t \sim \mathcal{N}(0, u) \Leftrightarrow W_t - W_s \sim \mathcal{N}(0, t - s)$
- $C(t, s) = \min(t, s)$

For simplicity, assume that $\mathbb{T} = [0, 1]$. The Karhunen-Loève expansion of the Wiener process reads

$$W_t = \sqrt{2} \sum_{n=1}^{\infty} \frac{\sin((n + 0.5)\pi t)}{(n + 0.5)\pi} \zeta_n, \quad \zeta_n \sim \mathcal{N}(0, 1)$$

Assignment 2

Write a `python` program to generate a realization of the Wiener process defined above for $t \in [0, 1]$. Initially, use the Wiener's process definition and $N = 1000$ samples. Afterwards, employ the Karhunen-Loève expansion approximation with $M = [10, 100, 1000]$ samples. To qualitatively observe the "convergence" of the Karhunen-Loève approximation, use the same random variables for all values of M.
Hint 1. To generate a realization of the Wiener process using its definition, use that $W_t - W_s \sim \mathcal{N}(0, t - s)$; therefore, for an increment dt, $dW := W_{t+dt} - W_{dt} \sim \mathcal{N}(0, dt)$.
Hint 2. Using the same random variables for all $M = [M_1, \ldots, M_{max}]$ means to first generate M_{max} random variables and then, for $M_i < M_{max}$, to take the first M_i values.

Assignment 3 (Optional)

The KL terms for the Wiener process for an arbitrary time period $t \in [0, T]$ are given by

$$\phi_i(t) = \sqrt{\frac{2}{T}} \sin\left(\frac{(i + \frac{1}{2})\pi t}{T}\right)$$

and

$$\lambda_i = \frac{T^2}{(i + \frac{1}{2})^2 \pi^2}$$

for $i \in [1, \infty]$, see [1].

[1] http://www.maths.lth.se/matstat/staff/georg/Publications/lecture2004.pdf, page 94

We decide now to model the source term f as a Wiener process in our oscillator model

$$\begin{cases} \frac{d^2y}{dt^2}(t) + c\frac{dy}{dt}(t) + ky(t) = f\cos(\omega t) \\ y(0) = y_0 \\ \frac{dy}{dt}(0) = y_1. \end{cases} \tag{1}$$

Write a `python` program to propagate the uncertainty of f through the model in Eq. (1) using the Wiener process definition and KL expansion from Assignment 2. Use the following deterministic values $c = 0.5$, $k = 2.0$, $y_0 = 0.5$, $y_1 = 0.$, $w = 1.0$ and $\mu_f = 0.5$ for the mean value of f. Consider the time period $t \in [0, 10]$ and $dt = 0.01$. Next, use Monte Carlo sampling to generate $N = 100$ solutions using $M = [5, 10, 100]$ KL terms and compute the mean and variance over time. Think about how you could employ other methods discussed in the lecture.
Hint: Adapt the deterministic solver from Assignment 2, Worksheet 1.