PHIMECA

… solutions for robust engineering

Simulation and identification of non-Gaussian, non-stationary random processes using OpenTURNS

V. Dubourg[1], M. Marcilhac[1], G. Blatman[2], M. Berveiller[2], R. Décatoire[1], T. Yalamas[1]

[1] Phimeca Engineering SA

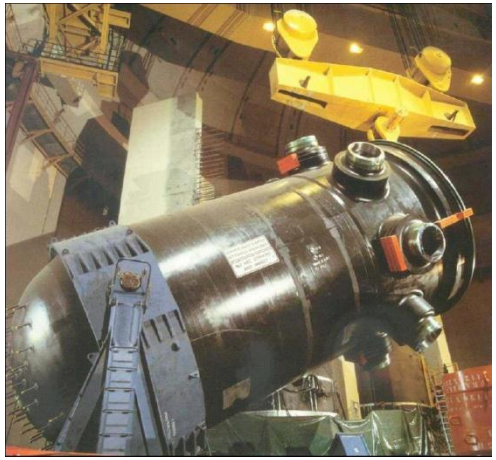
[2] EdF MMC, Site des Renardières

Introduction

Context

Scientific Network

- A group of partners around the problematique of *ageing components in nuclear power plants*.
- Means:
  - testing (subatomic scale microscope, non-destructive testing)
  - numerical simulation (HPC)
  - theoretical developments
  - courses



Phimeca, EdF MMC & Institut Navier work together at implementing tools for the *modeling of spatially and randomly varying structural properties*.

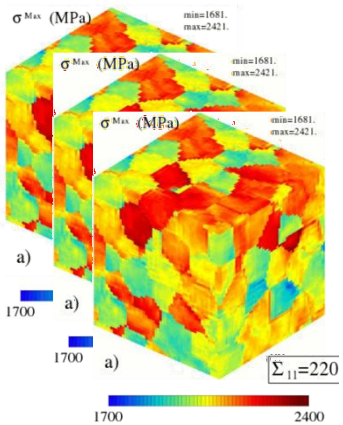
Introduction

Objective

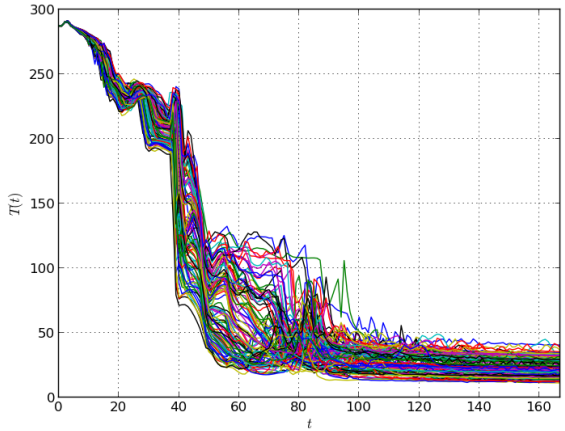
- Let suppose we have *a sample of experimental data*:

$$\boldsymbol{\mathcal{Y}} = \left\{ \left\{ \boldsymbol{y}^{(q)}\left(\boldsymbol{x}^{[i]}\right), i = 1, \dots, N^{(q)} \right\}, q = 1, \dots, Q \right\}, \qquad \boldsymbol{x} \in \mathbb{X} \subseteq \mathbb{R}^d$$

- Ex:



Stress field within polycristalline aggregates of 16MND5 steel (Code_Aster output)



Thermodynamic transient state in the vessel of a nuclear powerplant (Cathare output)

We aim at *building a model* representative of the spatial and/or temporal random variations of these properties

Introduction

Available representation techniques (simulation and identification)

- (Assumed) stationary and Gaussian random processes:
  - Spectral representation (periodogram analysis)
  - Optimal linear estimation, *a.k.a.* Kriging (variographic analysis)

  ➢ Available in OpenTURNS (see the model_process submodule)

- Non-stationary and non-Gaussian random processes:
  - Karhunen-Loève expansion
  - and that's all folks!... (yet, to my knowledge)

  ➢ Not available (yet) in OpenTURNS

We assume the data is complete enough for estimating its non-stationarity and non-gaussianity.

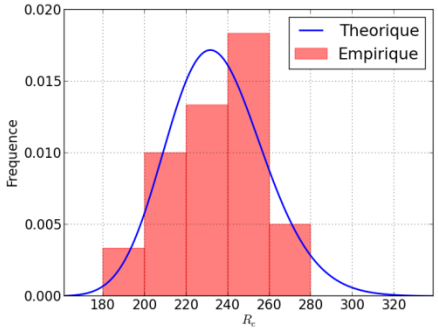
Outline

- Definition of a random field
- The Karhunen-Loève representation of random fields
  - Theoretical and practical representations
  - Galerkin-based discretization of the autocovariance function
    - Global approaches
    - A word on local approaches
- Application to the simulation of random fields
  - Gaussian random fields
  - Translation fields
- Application to the identification of random fields
  - Estimating the mean and the covariance
  - Identification of the distribution of the KL coefficients
- Implementation
  - OpenTURNS-based Python implementation
  - Examples
- Conclusions & Perspectives

Definition

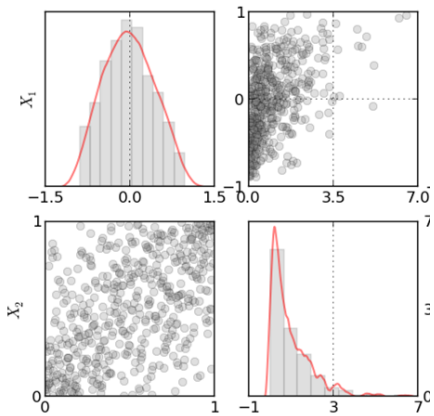
- Random variables

$$Y : \omega \to y = Y(\omega)$$



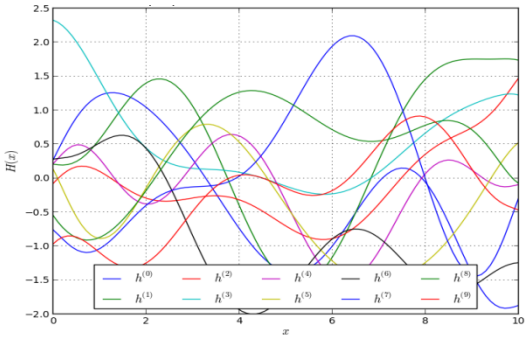
- Random vectors

$$\boldsymbol{Y} : \omega \to \boldsymbol{y} = \boldsymbol{Y}(\omega) = \begin{bmatrix} Y_1(\omega) \\ \vdots \\ Y_n(\omega) \end{bmatrix}, \qquad n \in \mathbb{N}$$



- Scalar-valued random fields

$$Y : \omega \to y(\boldsymbol{x}) = Y(\boldsymbol{x}, \omega), \qquad \boldsymbol{x} \in \mathbb{X} \subset \mathbb{R}^d$$



Outline

- Definition of a random field
- The Karhunen-Loève representation of random fields
  - Theoretical and practical representations
  - Galerkin-based discretization of the autocovariance function
    - Global approaches
    - A word on local approaches
- Application to the simulation of random fields
  - Gaussian random fields
  - Translation fields
- Application to the identification of random fields
  - Estimating the mean and the covariance
  - Identification of the distribution of the KL coefficients
- Implementation
  - OpenTURNS-based Python implementation
  - Examples
- Conclusions & Perspectives

The Karhunen-Loève expansion

A naïve approach to the modelling of random fields

- Let suppose we have some *discretized sample paths*:

$$\boldsymbol{\mathcal{Y}} = \left\{\left\{y^{(q)}\left(\boldsymbol{x}^{[i]}\right), i = 1, \dots, N^{(q)}\right\}, q = 1, \dots, Q\right\}, \qquad \boldsymbol{x} \in \mathbb{X}$$

- We can *recast* these sample paths *to a common functional representation* (using *e.g.* ordinary least-squares regression):

$$\mathcal{F} = \left\{\varphi\left(\boldsymbol{x}; \boldsymbol{\xi}^{(q)}\right), q = 1, \dots, Q\right\}, \qquad \boldsymbol{x} \in \mathbb{X}$$

Ex:

$$\varphi(x; \boldsymbol{\xi}) = \sum_{k=1}^{K} \xi_k \cos\left(\frac{2\pi}{\overline{\mathbb{X}}} kx\right), \qquad x \in \mathbb{X}$$

- Eventually, *we are back into « ordinary statistics »*: what is the joint distribution of the vector $\boldsymbol{\Xi}$?
  - Marginal distributions
  - Complete the joint distribution by composing with copulas

The Karhunen-Loève expansion

- There exists an optimal representation (Loève, 1957)
  - The Karhunen-Loève expansion reads:

$$Y(\boldsymbol{x}) = \mu_Y(\boldsymbol{x}) + \sum_{k=1}^{+\infty} \sqrt{\lambda_k}\Xi_k\varphi_k(\boldsymbol{x}), \qquad \boldsymbol{x} \in \mathbb{X}$$

where:

- the *infinite but countable set of 2-uples* $\{(\lambda_k, \varphi_k), k \in \mathbb{N}^*\}$ are solutions of the eigendecomposition of the field's autocovariance function C:

$$\int_{\mathbb{X}} C(\boldsymbol{x}, \boldsymbol{x}')\varphi_k(\boldsymbol{x}')\mathrm{d}\boldsymbol{x}' = \lambda_k\varphi_k(\boldsymbol{x}), \qquad \boldsymbol{x} \in \mathbb{X}$$

$$C(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}\big[\big(Y(\boldsymbol{x}) - \mu_Y(\boldsymbol{x})\big)\big(Y(\boldsymbol{x}') - \mu_Y(\boldsymbol{x}')\big)\big], \qquad (\boldsymbol{x}, \boldsymbol{x}') \in \mathbb{X} \times \mathbb{X}$$

- the coefficients are *zero-mean*, *unit-variance*, *non-correlated* random variables:

$$\begin{cases} \mathbb{E}[\Xi_k] = 0, & \forall k \in \mathbb{N}^* \\ \mathbb{E}[\Xi_i\Xi_j] = \delta_{ij}, & (i,j) \in \mathbb{N}^* \times \mathbb{N}^* \end{cases}$$

- « Optimality » holds *in the mean square sense*.

The Karhunen-Loève expansion

In practice

- The *truncated* Karhunen-Loève expansion reads:

$$\hat{Y}_M(\boldsymbol{x}) = \mu_Y(\boldsymbol{x}) + \sum_{k=1}^{M} \sqrt{\lambda_k}\Xi_k\varphi_k(\boldsymbol{x}), \qquad \boldsymbol{x} \in \mathbb{X}, M \in \mathbb{N}^*$$

where:

  - the infinite countable set of 2-uples $\{(\lambda_k, \varphi_k), k \in \mathbb{N}^*\}$ are *ordered in decreasing order of eigenvalues*:

$$\lambda_1 > \cdots > \lambda_M > \lambda_{M+1} > \cdots$$

  - the random coefficients are zero-mean, unit-variance, non-correlated random variables:

$$\begin{cases} \mathbb{E}[\Xi_k] = 0, & k = 1, \dots, M \\ \mathbb{E}[\Xi_i\Xi_j] = \delta_{ij}, & i, j = 1, \dots, M \end{cases}$$

- Truncation results in an *approximation of the given autocovariance function* (variance is underestimated).

Galerkin-based discretization

- Numerical resolution of the Fredholm integral equation

$$\int_{\mathbb{X}} C(\boldsymbol{x},\boldsymbol{x}')\varphi_k(\boldsymbol{x}')\mathrm{d}\boldsymbol{x}' = \lambda_k \varphi_k(\boldsymbol{x}), \qquad \boldsymbol{x} \in \mathbb{X}$$

- This equation admits *analytical solutions for a (very?) few parametric covariance functions*.

  Ex: the absolute exponential (stationary) autocovariance function:

$$C(\boldsymbol{x},\boldsymbol{x}') = \sigma^2 \exp\left(\sum_{i=1}^{d} \frac{|x_i - x_i'|}{\ell_i}\right), \qquad (\boldsymbol{x},\boldsymbol{x}') \in \mathbb{X} \times \mathbb{X}$$

- Otherwise, we need to resort to numerical discretization using a *Galerkin scheme* (Ghanem & Spanos, 1993; Sudret, 2000).

Galerkin-based discretization

- An optimal approximation of the unknown eigenfunctions
  - We *approximate the eigenfunctions* with a linear combination of square-integrable functions:

$$\varphi_k(\boldsymbol{x}) = \sum_{i=1}^{N} d_i^{(k)} h_i(\boldsymbol{x}), \qquad \boldsymbol{x} \in \mathbb{X}$$

  - The k-th residual of the Fredholm integral equation reads:

$$\varepsilon_k(\boldsymbol{x}) = \sum_{i=1}^{N} d_i^{(k)} \left[\int_{\mathbb{X}} C(\boldsymbol{x}, \boldsymbol{x}') \varphi_k(\boldsymbol{x}') \mathrm{d}\boldsymbol{x}' - \lambda_k h_i(\boldsymbol{x}) \right], \qquad \boldsymbol{x} \in \mathbb{X}, k = 1, \dots, K$$

  - The Galerkin scheme consists in *requiring that this residual is orthogonal to the approximation basis*:

$$\int_{\mathbb{X}} \varepsilon_k(\boldsymbol{x}) h_i(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} = 0, \qquad k = 1, \dots, M, i = 1, \dots, N$$

Galerkin-based discretization

An optimal approximation of the unknown eigenfunctions (cont'd)

- Eventually, we obtain $N+1$ unknowns for N equations for each 2-uples of eigensolutions, which can be put in the following matrix form:

$$\mathbf{CD} = \mathbf{\Lambda BD}$$

where:

$$C_{ij} = \int_{\mathbb{X}} \int_{\mathbb{X}} C(\boldsymbol{x}, \boldsymbol{x}') h_i(\boldsymbol{x}) h_j(\boldsymbol{x}') \mathrm{d}\boldsymbol{x}\, \mathrm{d}\boldsymbol{x}', \qquad i,j = 1, \dots, N$$

$$B_{ij} = \int_{\mathbb{X}} h_i(\boldsymbol{x}) h_j(\boldsymbol{x}) \mathrm{d}\boldsymbol{x}, \qquad i,j = 1, \dots, N$$

- Then, we resort to a *numerical solver for finding the $M \leq N$ solutions* of the generalized eigenvalue problem that have the *largest eigenvalues*:

$$\mathbf{\Lambda} = \mathrm{diag}\left(\{\lambda_k, k = 1, \dots, M\}\right)$$

$$\mathbf{D} = \left[d_i^{(k)}, i = 1, \dots, N, k = 1, \dots, M\right]$$

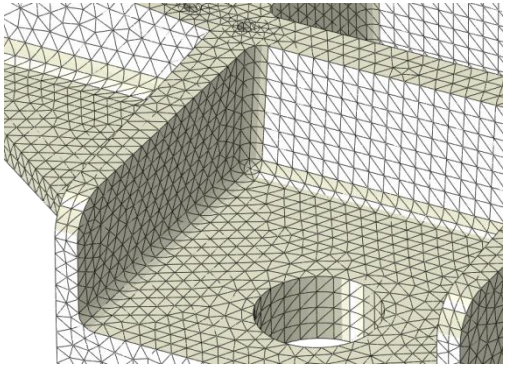
Global vs. local approximation

Global approximation

- We can choose any (square-integrable) functional basis with *sympathetic properties*:
  - Legendre polynomials (present implementation):
    - so that C can be calculated using *Gauss-Legendre quadrature*
    - orthogonal, hence B is diagonal and norm is known analytically
  - Haar wavelets (Phoon, 2002), or any other wavelets family:
    - so that C is the *discrete wavelet transform* of the autocovariance function
    - orthogonal, hence B is diagonal and norm is known analytically
- Or any other basis but with more pain!

Local approximation (Recek, 2007; Perrin, 2013)

- Shape functions over a finite element mesh
- Better suited for non-rectangular domains
- … $\mathbf{C}$ and $\mathbf{B}$ can be dense though



➢ FE-software-dependent

Outline

- Definition of a random field
- The Karhunen-Loève representation of random fields
  - Theoretical and practical representations
  - Galerkin-based discretization of the autocovariance function
    - Global approaches
    - A word on local approaches
- Application to the simulation of random fields
  - Gaussian random fields
  - Translation fields
- Application to the identification of random fields
  - Estimating the mean and the covariance
  - Identification of the distribution of the KL coefficients
- Implementation
  - OpenTURNS-based Python implementation
  - Examples
- Conclusions & Perspectives

Simulation of random fields

Simulation of Gaussian random fields

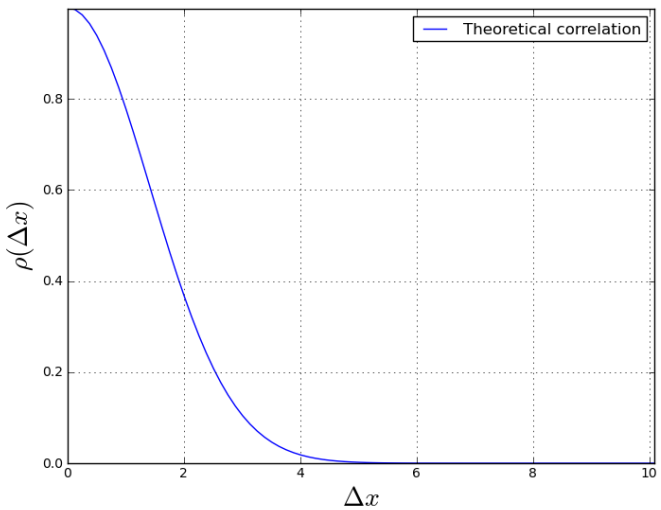
- Definition:
  - rectangular domain
  - mean function
  - autocovariance functions

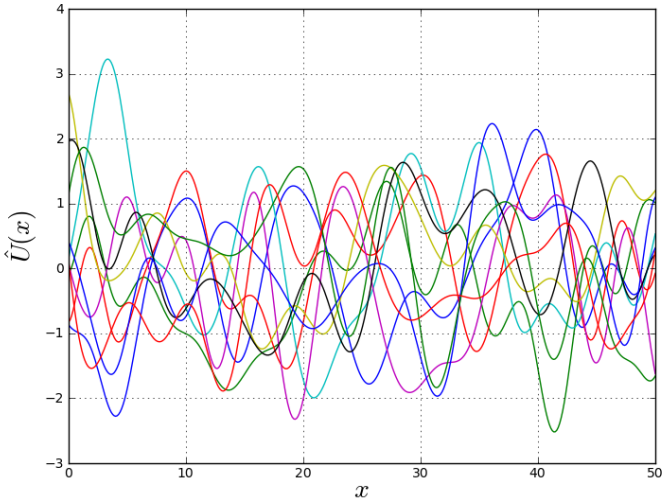
- Numerical discretization using either global or local approximation

- The field being assumed Gaussian, the KL coefficients are jointly normally distributed:

$$\boldsymbol{\Xi} \sim \mathcal{N}_M(\mathbf{0}, \mathbf{1})$$

- Ex: Zero-mean, unit-variance with a square exponential autocovariance





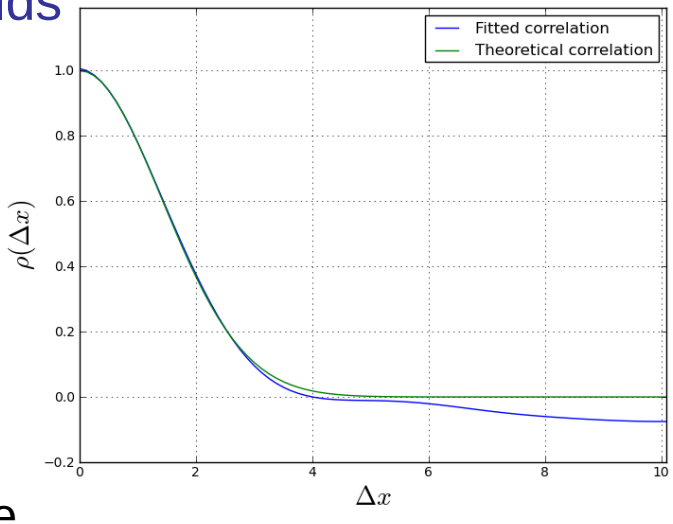
Simulation of random fields

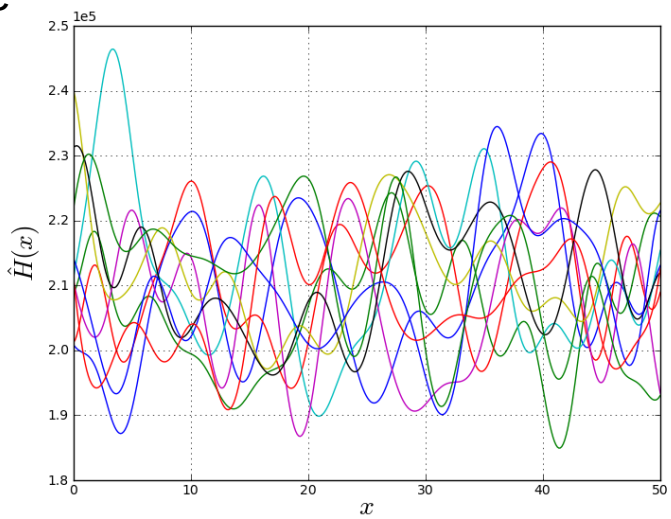
Simulation of non-Gaussian random fields

- Definition:
  - rectangular domain
  - mean function
  - autocovariance function
  - « pointwise distribution »
- Numerical discretization using either global or local approximation
- Simulation of a zero-mean, unit-variance Gaussian process
- Translation into the targeted non-Gaussian randomfield using a pointwise isoprobabilistic transformation:

$$Y(\boldsymbol{x}) = T(U(\boldsymbol{x}), \boldsymbol{x}), \qquad x \in \mathbb{X}$$

- Ex: A lognormal translation field





Simulation of random fields

Simulation of non-Gaussian random fields

- Premise of a number of authors:
  - « I can't get the autocorrelation and the distribution! »
  - The *translation alters the given autocovariance function*
  - Workaround: well, let us try to solve our favorite problem:

$$\rho(\boldsymbol{x},\boldsymbol{x}') = \int_{\mathbb{R}}\int_{\mathbb{R}} \begin{matrix} \left(F_{Y(\boldsymbol{x})}^{-1}(\Phi(z)) - \mu_Y(\boldsymbol{x})\right)\left(F_{Y(\boldsymbol{x}')}^{-1}(\Phi(z')) - \mu_Y(\boldsymbol{x}')\right) \times \cdots \\ \ldots \times \varphi_2(z,z';\rho_0(\boldsymbol{x},\boldsymbol{x}'))\mathrm{d}z\mathrm{d}z' \end{matrix}, \qquad (\boldsymbol{x},\boldsymbol{x}') \in \mathbb{X}\times\mathbb{X}$$

  - ➢ Pearson stroke again! (Lebrun & Dutfoy, 2009a,b,c; Lebrun, 2013)

- What you can get though is:
  - a pointwise distribution and a *fractile autocorrelation* (Phoon, 2004)

$$\rho_0(\boldsymbol{x},\boldsymbol{x}') = 2\sin\left(\frac{\pi}{6}\rho_S(\boldsymbol{x},\boldsymbol{x}')\right)$$

  - … and, between the lines: an assumed Gaussian copula parameterized with a Spearman rank autocorrelation function!

Outline

- Definition of a random field
- The Karhunen-Loève representation of random fields
  - Theoretical and practical representations
  - Galerkin-based discretization of the autocovariance function
    - Global approaches
    - A word on local approaches
- Application to the simulation of random fields
  - Gaussian random fields
  - Translation fields
- Application to the identification of random fields
  - Estimating the mean and the covariance
  - Identification of the distribution of the KL coefficients
- Implementation
  - OpenTURNS-based Python implementation
  - Examples
- Conclusions & Perspectives

Identification of random fields

Identification of non-stationary, non-Gaussian random fields

- Interpolation of the sample paths (*e.g.* linear)
- Pointwise *empirical estimation* of the mean and autocovariance functions:

$$\hat{\mu}_Y(\boldsymbol{x}) = \frac{1}{Q}\sum_{q=1}^{Q} y^{(q)}(\boldsymbol{x}), \qquad \boldsymbol{x} \in \mathbb{X}$$

$$\hat{C}(\boldsymbol{x},\boldsymbol{x}') = \frac{1}{Q-1}\sum_{q=1}^{Q}\left(y^{(q)}(\boldsymbol{x}) - \hat{\mu}_Y(\boldsymbol{x})\right)\left(y^{(q)}(\boldsymbol{x}') - \hat{\mu}_Y(\boldsymbol{x}')\right), (\boldsymbol{x},\boldsymbol{x}') \in \mathbb{X}\times\mathbb{X}$$

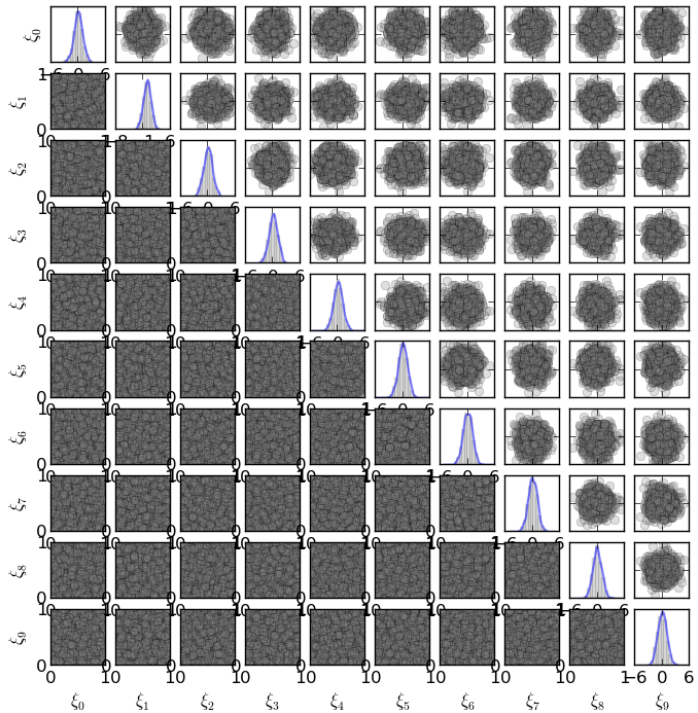
- Numerical discretization using either global or local approximation

- Compute *the sample of KL coefficients* associated to the set of sample paths by exploiting *the property of orthogonality of eigenfunctions*:

$$\xi_k^{(q)} = \frac{1}{\sqrt{\lambda_k}}\int_{\mathbb{X}} \varphi_k(\boldsymbol{x})\left(y^{(q)}(\boldsymbol{x}) - \mu_Y(\boldsymbol{x})\right)\mathrm{d}\boldsymbol{x}, \qquad k = 1,\dots,M, q = 1,\dots,Q$$

Identification of random fields

- Identification of the KL coefficients' joint distribution



- Parametric or non-parametric models:
  - for the marginal distributions
  - for the (composed) copula(s)
- Despite the number of KL coefficients is optimal (minimal), it can remain quite large!

Outline

- Definition of a random field
- The Karhunen-Loève representation of random fields
  - Theoretical and practical representations
  - Galerkin-based discretization of the autocovariance function
    - Global approaches
    - A word on local approaches
- Application to the simulation of random fields
  - Gaussian random fields
  - Translation fields
- Application to the identification of random fields
  - Estimating the mean and the covariance
  - Identification of the distribution of the KL coefficients
- Implementation
  - OpenTURNS-based Python implementation
  - Usage through examples
- Conclusions & Perspectives

OpenTURNS-based Python implementation

The KarhunenLoeveExpansion object

- __init__:
  - The constructor discretizes the random field using *Gauss-Legendre quadrature* and *tensorized Legendre polynomials* from OpenTURNS
  - It takes μ, C, the bounds of the rectangular domain and the truncation order M as arguments
  - It also features a set of keyword arguments:
    - the order of the *tensorized Legendre* approximation basis ($N = \binom{d+p}{p} > M$)
    - the quadrature order for calculating $\mathbf{C}$ and $\boldsymbol{\xi}$'s
- __call__:
  - The object is callable and takes $\boldsymbol{x}$'s and $\boldsymbol{\xi}$'s as input, it calculates the required sample paths values.
- _eigenvalues, _eigenfunctions
- _compute_approximated_covariance:
  - It calculates the covariance of the truncated expansion for comparison:

$$\hat{C}(\boldsymbol{x}, \boldsymbol{x}') = \sum_{k=1}^{M} \lambda_k \, \varphi_k^2(\mathbf{x}) = f(\boldsymbol{x}, \boldsymbol{M}, \boldsymbol{N})$$

- _compute_coefficients (uses Gauss-Legendre quadrature)

OpenTURNS-based Python implementation

Identification of a random field using Karhunen-Loeve expansion — Phimeca's Python tools - Mozilla Firefox

file:///home/dubourg/pythonTools/doc/build/html/auto_examples/randomfields/plot_karhunen_loeve_identification.html

PHIMECA 's PYTHON TOOLS

Phimeca's Python tools » Examples gallery »

previous | next | index

Identification of a random field using Karhunen-Loeve expansion

This scripts shows how to identify the parameters characterizing a random field H(x) from a set of sample paths using the Karhunen-Loeve expansion method.

The set of sample paths is generated from a one-dimensional Gaussian random field defined over [0; 10] with zero mean, unit variance and a squared exponential covariance function with correlation length 2.

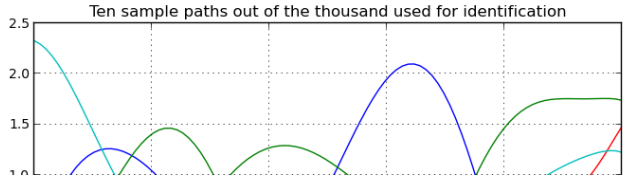
From phimeca.randomfields, it uses *KarhunenLoeveExpansion* for:

- solving the spectral decomposition problem of the estimated covariance function of the field H.
- calculating the coefficients of the Karhunen-Loeve expansion truncated to a given order depending on the decrease of the formerly computed eigenvalues. These coefficients are uncorrelated, zero mean and unit variance random variables.

Their joint distribution is then estimated (here, using kernel smoothing) and new sample paths are eventually generated using the discretized random field computed by the instanciated KarhunenLoeveExpansion.

The identification procedure consists in the following steps:

1. Modelling of the mean and covariance functions by some interpolation technique.
2. Resolution of the integral eigenvalue problem of the covariance function (i.e. discretization of the random field into a Karhunen-Loeve expansion).
3. Computation of the coefficients of the Karhunen-Loeve expansion by functional projection using Gauss-Legendre quadrature.
4. Identification of the joint distribution of these coefficients. Here, this resorts to a kernel smoothing non-parametric technique for the marginal distributions and a Gaussian copula is assumed for illustration.
5. Generation of new sample paths.



Previous topic

Random fields simulation example

Next topic

Simulation of a random field using Karhunen-Loeve expansion

This Page

Show Source

Quick search

Go

Enter search terms or a module, class or function name.

Conclusions & Perspectives

Conclusions

- OpenTURNS has made a long way towards scientific Python:
  - *Less typing* (ot.Distribution(ot.Normal()) → ot.Normal())
  - Better casting from Sample/Point to *Numpy Array* (and vice-versa)
- But it still lacks:
  - *Easy typing* (remaining ot.Indices types in *e.g.* getMarginalDistribution)
  - Docstrings
- Features request:
  - a built-in method for computing integrals using Gauss-* quadrature
  - completing quadrature schemes such as Gauss-Kronrod (?)

Perspectives

- Implementation of the Karhunen-Loève representation of random fields within OpenTURNS:
  - Global approach: definitely go for *wavelets*!
  - Local approach: using *Vtk*? Code_Aster?
- Some literature about *vector-valued KL expansions*: Perrin, 2013.