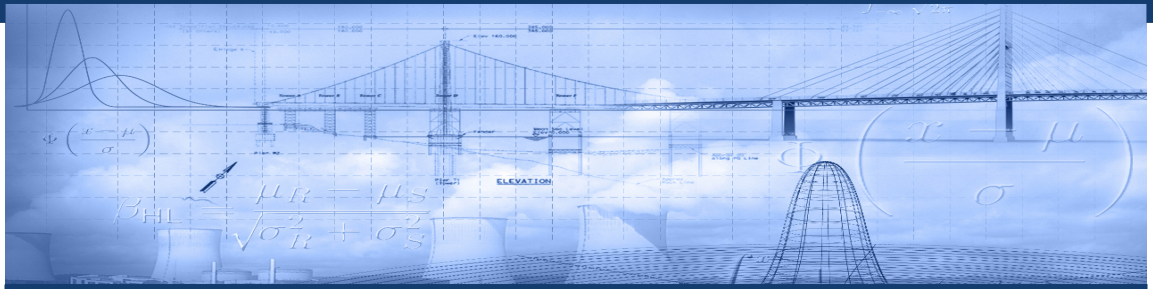
Surrogate modelling approaches for stochastic simulators

ETH zürich



Surrogate modelling approaches for stochastic simulators

Bruno Sudret

Chair of Risk, Safety and Uncertainty Quantification, ETH Zurich

This presentation is an invited lecture given in the *Seminar for machine learning and uncertainty quantification in scientific computing* at the Centrum Wiskunde & Informatica, Amsterdam (The Netherlands) on June 17, 2021.

How to cite ?

Sudret, B., *Surrogate modelling approaches for stochastic simulators*, Centrum Wiskunde & Informatica, Amsterdam (The Netherlands), invited lecture, June 17th, 2021.

Chair of Risk, Safety and Uncertainty quantification

The Chair carries out research projects in the field of uncertainty quantification for engineering problems with applications in structural reliability, sensitivity analysis, model calibration and reliability-based design optimization

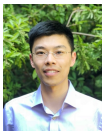
Research topics

- Uncertainty modelling for engineering systems
- Structural reliability analysis
- Surrogate models (polynomial chaos expansions, Kriging, support vector machines)
- Bayesian model calibration and stochastic inverse problems
- Global sensitivity analysis
- Reliability-based design optimization

`http://www.rsuq.ethz.ch`

The SAMOS project

The SAMOS project (**"SurrogAte Modelling for stOchastic Simulators"**) is funded by the Swiss National Science Foundation under Grant # 175524 (2018 –2022).



Xujia Zhu

Stochastic emulators using generalized lambda distributions

Nora Lüthen

Sparse polynomial chaos expansions / Random field representations for stochastic emulators

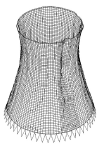
Computational models in engineering

Complex engineering systems are designed and assessed using computational models, a.k.a. simulators

A computational model combines:

- A mathematical description of the physical phenomena (governing equations), *e.g.* mechanics, electromagnetism, fluid dynamics, etc.

$$\begin{aligned} \operatorname{div} \boldsymbol{\sigma} + \boldsymbol{f} &= \mathbf{0} \\ \boldsymbol{\sigma} &= \mathbf{D} \cdot \boldsymbol{\varepsilon} \\ \boldsymbol{\varepsilon} &= \frac{1}{2}\left(\nabla \boldsymbol{u} + {}^{\mathsf{T}}\nabla \boldsymbol{u}\right) \end{aligned}$$

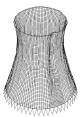


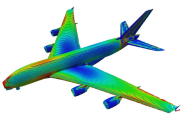
- Discretization techniques which transform continuous equations into linear algebra problems
- Algorithms to solve the discretized equations

Computational models in engineering

Computational models are used:

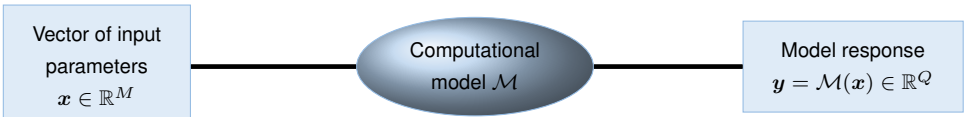
- To explore the design space (“virtual prototypes”)
- To optimize the system (*e.g.* minimize the mass) under performance constraints
- To assess its robustness w.r.t. uncertainty and its reliability
- Together with experimental data for calibration purposes



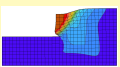



Computational models: the abstract viewpoint

A computational model may be seen as a black box program that computes quantities of interest (QoI) (a.k.a. model responses) as a function of input parameters



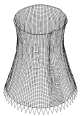
- Geometry
- Material properties
- Loading

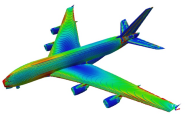


- Analytical formula
- Finite element model
- Comput. workflow

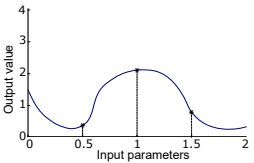
- Displacements
- Strains, stresses
- Temperature, etc.

Deterministic vs. stochastic simulators



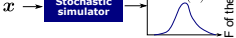



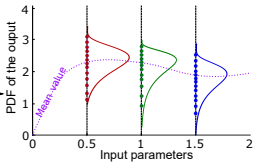
Deterministic simulators



Output $\mathcal{M}_d(\boldsymbol{x})$ is a real number

Stochastic simulators





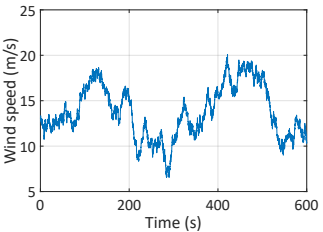
Output $\mathcal{M}_s(\boldsymbol{x})$ is a random variable

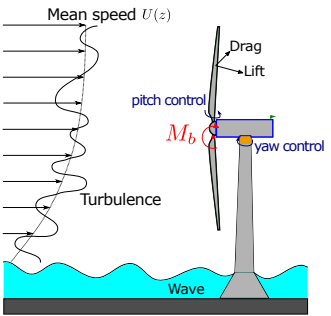
Example: wind turbine simulation

Wind generation → Aero-servo-elastic simulator

Input parameters

- Mean speed
- Turbulence intensity
- Shear exponent
- Air density
- Inclination angle





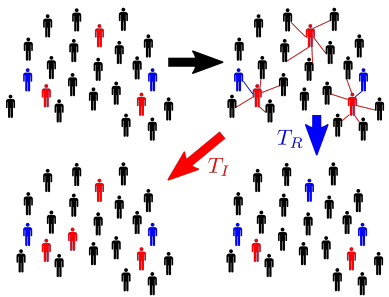
Example: epidemiology

Terminology

- S_t: number of susceptible individuals at time t
- I_t: number of infected individuals at time t
- R_t: number of recovered individuals at time t

System dynamics

- Susceptible individuals can get infected due to close contact with infected individuals ($S \to I$)
- Infected individuals can recover and becomes immune to future infections ($I \to R$)
- Random contact and recovery modelled with Poisson processes

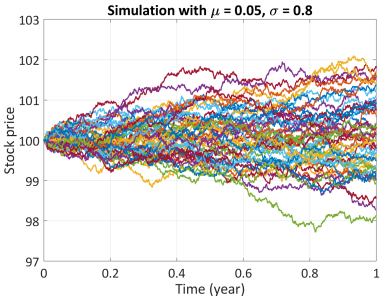


Example: mathematical finance

Geometrical Brownian motion

$$\mathrm{d}S_t = \mu\, S_t\, \mathrm{d}t + \sigma\, S_t\, \mathrm{d}W_t$$

- S_t: stock price, W_t: Wiener process
- μ: drift, σ: volatility



Asian option

- The payoff (of a call option) is contingent on the average price of the underlying asset

$$C = \max\left\{A_T - K, 0\right\}, \text{ with } A_t = \frac{1}{t}\int_0^t S_u \mathrm{d}u.$$

Outline

Formal definition

- A (scalar) stochastic simulator $\mathcal{M}_s$ is a mapping:

$$\begin{aligned} \mathcal{M}_s : \mathcal{D}_{\boldsymbol{X}} \times \Omega &\to \mathbb{R} \\ (\boldsymbol{x}, \omega) &\mapsto \mathcal{M}_s(\boldsymbol{x}, \omega) \end{aligned}$$

  where $\mathcal{D}_{\boldsymbol{X}}$ is the input parameters space and $\{\Omega, \mathcal{F}, \mathbb{P}\}$ is a probability space

- When fixing $\boldsymbol{x} = \boldsymbol{x}_0$, the output is a random variable $Y|\boldsymbol{X} = \boldsymbol{x}_0 \equiv \mathcal{M}_s(\boldsymbol{x}_0, \omega)$
- When fixing the seed $\omega = \omega_0$ we get a deterministic simulator $\boldsymbol{x} \mapsto \mathcal{M}_s(\boldsymbol{x}, \omega_0)$ (a.k.a. trajectory)

Latent variables

$\mathcal{M}_s$ can be seen as a deterministic function $\mathcal{M}$ of input parameters $\boldsymbol{x}$ and latent variables $\boldsymbol{Z}$:

$$\mathcal{M}_s(\boldsymbol{x}, \omega) = \mathcal{M}\left(\boldsymbol{x},\, \boldsymbol{Z}(\omega)\right)$$

Computational costs induced by stochastic simulators

- Replications are needed to estimate the PDF of $Y|\boldsymbol{X} = \boldsymbol{x}$
- Many runs must be carried out by varying $\boldsymbol{X}$ for uncertainty propagation, sensitivity analysis, optimization, etc.
- Realistic simulators (e.g. for wind turbine design) are costly

Need for surrogate models

Surrogate models (deterministic simulators)

A surrogate model $\tilde{\mathcal{M}}$ is an approximation of the original computational model $\mathcal{M}$:

$$\tilde{\mathcal{M}}_d(\boldsymbol{x}) \approx \mathcal{M}_d(\boldsymbol{x}) \qquad \text{in some sense}$$

with the following features:

- It assumes some regularity of the model $\mathcal{M}$ and some general functional shape
- It is built from a limited set of runs of the original model $\mathcal{M}$ called the experimental design $\mathcal{X} = \left\{\boldsymbol{x}^{(i)},\, i = 1,\, \ldots\, , N\right\}$

  Simulated data

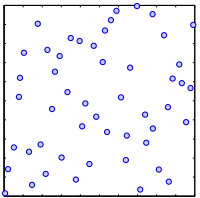
- It is fast to evaluate!

Surrogate models for uncertainty quantification

Name	Shape	Parameters
Polynomial chaos expansions	$\tilde{\mathcal{M}}(\boldsymbol{x}) = \sum_{\boldsymbol{\alpha} \in \mathcal{A}} c_{\boldsymbol{\alpha}} \, \Psi_{\boldsymbol{\alpha}}(\boldsymbol{x})$	$\boldsymbol{c_\alpha}$
Low-rank tensor approximations	$\tilde{\mathcal{M}}(\boldsymbol{x}) = \sum_{l=1}^{R} b_l \left(\prod_{i=1}^{M} v_l^{(i)}(x_i) \right)$	$b_l, \, z_{k,l}^{(i)}$
Kriging (a.k.a Gaussian processes)	$\tilde{\mathcal{M}}(\boldsymbol{x}) = \boldsymbol{\beta}^{\mathsf{T}} \cdot \boldsymbol{f}(\boldsymbol{x}) + Z(\boldsymbol{x}, \omega)$	$\boldsymbol{\beta}$, σ_Z^2 , $\boldsymbol{\theta}$
Support vector machines	$\tilde{\mathcal{M}}(\boldsymbol{x}) = \sum_{i=1}^{m} c_i \, K(\boldsymbol{x}_i, \boldsymbol{x}) + b$	$\boldsymbol{c}$, b
(Deep) Neural networks	$\tilde{\mathcal{M}}(\boldsymbol{x}) = f_n \left(\cdots f_2 \left(b_2 + f_1 \left(b_1 + \boldsymbol{w}_1 \cdot \boldsymbol{x} \right) \cdot \boldsymbol{w}_2 \right) \right)$	$\boldsymbol{w}, \boldsymbol{b}$

Ingredients for building a surrogate model

- Select an experimental design $\mathcal{X}$ that covers at best the domain of input parameters:
  - (Monte Carlo simulation)
  - Latin hypercube sampling (LHS)
  - Low-discrepancy sequences



- Run the computational model $\mathcal{M}$ onto $\mathcal{X}$ exactly as in Monte Carlo simulation

Ingredients for building a surrogate model

- Smartly post-process the data $\{\mathcal{X}, \mathcal{M}(\mathcal{X})\}$ through a learning algorithm

Name	Learning method
Polynomial chaos expansions	sparse grid integration, least-squares, compressive sensing
Low-rank tensor approximations	alternate least squares
Kriging	maximum likelihood, Bayesian inference
Support vector machines	quadratic programming

- Validate the surrogate model, *e.g.* estimate the generalization error $\varepsilon = \mathbb{E}\left[\left(\mathcal{M}(\boldsymbol{X}) - \tilde{\mathcal{M}}(\boldsymbol{X})\right)^2\right]$

Back to stochastic simulators

Our goal is to develop a methodology that is:

- Non-intrusive (*i.e.* that considers the stochastic simulator as a black box)
- General-purpose: no restrictive assumption (*e.g.* Gaussian) on the family of the output distribution is made
- Able to tackle the full distribution of $Y|\boldsymbol{X} = \boldsymbol{x}$, but also quantities of interest (*e.g.* mean, variance, quantiles)
- Providing a representation of $Y|\boldsymbol{X} = \boldsymbol{x}$ easy to sample from

Existing approaches

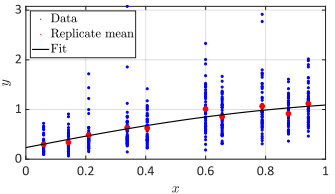
The literature on stochastic simulators is both old and new:

- Replication-based approaches
- Gaussian models
- Estimation of the conditional distribution
- Latent variable models
- Random field representations
- Quantile regression

Replication-based approaches

Main idea

- Estimate distributions/QoIs based on replications
- Treat the estimated parameters as outputs from a deterministic simulator and apply standard surrogate models



Literature

- Stochastic Kriging: Ankenman *et al.* (2006) *Stochastic Kriging for simulation metamodeling*, Oper. Res.
- Quantile Kriging: Plumlee & Tuo (2014) *Building accurate emulators for stochastic simulations via quantile Kriging*, Technometrics
- Kernel density estimation: Moutoussamy *et al.* (2015) *Emulators for stochastic simulation codes*, ESAIM: Math. Model. Num. Anal.
- Generalized lambda model: Zhu & Sudret (2020) *Replication-based emulation of the response distribution of stochastic simulators using generalized lambda distributions*, Int. J. Uncertainty Quantification

Assuming normality: Kriging models

Main idea

- Response distributions are normal
- Mean function $\mu(\boldsymbol{x})$ and log-variance function $\log(V(\boldsymbol{x}))$ are modeled by Gaussian processes

Literature

- Full Bayesian setup: Goldberg *et al.* (1997) *Regression with input-dependent noise: a Gaussian process treatment*, NIPS10
- Iterative fitting: Marrel *et al.* (2012) *Global sensitivity analysis of stochastic computer models with joint metamodels*, Stat. Comput.
- Maximum likelihood: Binois *et al.* (2018) *Practical heteroscedastic Gaussian process modeling for large simulation experiments*, J. Comput. Graph. Stat.

Conditional distribution estimation

Main approaches

- Estimate the joint distribution of $(\boldsymbol{X}, Y)$ by kernel smoothing, then compute the conditional PDF by:

$$f(y \mid \boldsymbol{x}) = \frac{f(\boldsymbol{x}, y)}{f(\boldsymbol{x})}$$

- Use parametric models to represent the conditional distribution directly

Literature

- Kernel smoothing: Hall *et al.* (2004) *Cross-validation and the estimation of conditional probability densities*, J. Amer. Stat. Assoc.
- Vine copula: Kraus & Czado (2017) *D-vine copula based quantile regression*, Comput. Stat. Data Anal.
- Generalized lambda model: Zhu & Sudret (2021) *Emulation of stochastic simulators using generalized lambda models*, Submitted to SIAM/ASA J. Unc. Quant.

Latent variable models

Main idea

- Introduce explicitly latent variables $\tilde{\boldsymbol{Z}}$ into a deterministic model to emulate the random nature of stochastic simulators

$$Y(\boldsymbol{x}) \overset{\mathrm{d}}{=} \tilde{\mathcal{M}}(\boldsymbol{x}, \tilde{\boldsymbol{Z}})$$

Literature

- Yan & Perdikaris (2019) *Conditional deep surrogate models for stochastic, high-dimensional, and multi-fidelity systems*, Comput. Mech.
- Stochastic polynomial chaos expansions for emulating stochastic simulators

Zhu & Sudret, EMI/PMC Conference, Columbia University, 2021

Random field approaches

Main idea

- Consider the stochastic simulator as a random field indexed by the input variables:

$$Y_{\boldsymbol{x}}(\omega) = \mathcal{M}\left(\boldsymbol{x}, \boldsymbol{Z}(\omega)\right)$$

- Fixing the internal stochasticity ($\omega = \omega_0$) gives access to trajectories $\boldsymbol{x} \mapsto \mathcal{M}(\boldsymbol{x}, \boldsymbol{Z}(\omega_0))$

Literature

- Azzi *et al.* (2019) *Surrogate modeling of stochastic functions-application to computational electromagnetic dosimetry*, Int. J. Uncertainty Quantification
- Azzi *et al.* (2020) *Sensitivity analysis for stochastic simulators using differential entropy*, Int. J. Uncertainty Quantification
- Surrogating stochastic simulators using spectral methods and advanced statistical modeling (PhD thesis N. Lüthen)

Outline

Definition

- The Freimer-Mudholkar-Kollia-Lin (FMKL) lambda distribution is defined through its quantile function $Q(u; \boldsymbol{\lambda})$ by 4 parameters

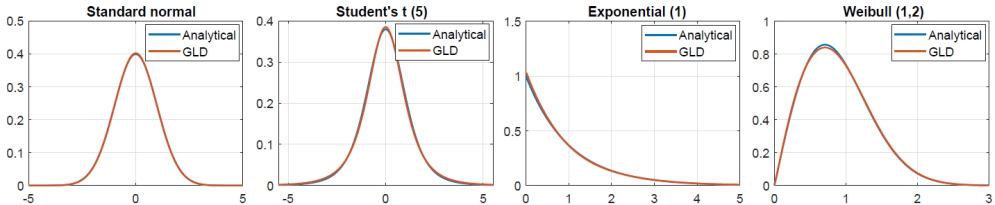
$$Q(u; \boldsymbol{\lambda}) = \lambda_1 + \frac{1}{\lambda_2}\left(\frac{u^{\lambda_3} - 1}{\lambda_3} - \frac{(1-u)^{\lambda_4} - 1}{\lambda_4}\right)$$

  where:
  - λ_1 is the location parameter
  - $\lambda_2 > 0$ is the scale parameter
  - λ_3, λ_4 are shape parameters

- The PDF is obtained by:

$$f_Y(y; \boldsymbol{\lambda}) = \frac{1}{Q'(u; \boldsymbol{\lambda})} = \frac{\lambda_2}{u^{\lambda_3 - 1} + (1-u)^{\lambda_4 - 1}} \qquad \text{with } u = Q^{-1}(y; \boldsymbol{\lambda})$$

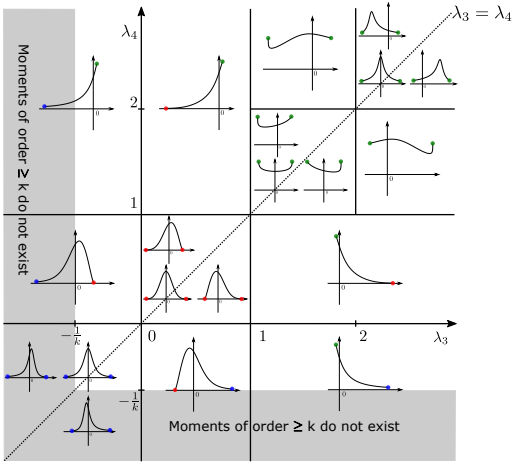
Properties



- GLDs approximate well unimodal PDFs (bell-, U-shaped, bounded and unbounded)
- λ_3 and λ_4 control the shape and boundedness

$$B_l(\boldsymbol{\lambda}) = \begin{cases} -\infty, & \lambda_3 \leq 0 \\ \lambda_1 - \frac{1}{\lambda_2 \lambda_3}, & \lambda_3 > 0 \end{cases}, \quad B_u(\boldsymbol{\lambda}) = \begin{cases} +\infty, & \lambda_4 \leq 0 \\ \lambda_1 + \frac{1}{\lambda_2 \lambda_4}, & \lambda_4 > 0 \end{cases}$$

Summary chart



- Blue points: infinite support
- Red points: finite support, with PDF = 0 at the bound
- Green points: finite support, with PDF $\neq$ 0 at the bound

Zhu & Sudret (2020), *Replication-based emulation of the response distribution of stochastic simulators using generalized lambda distributions*, Int. J. Uncertainty Quantification

Outline

Problem statement

Let us consider a stochastic simulator $\mathcal{M}_S$ and the following experimental design with replications

- Experimental design of size N in the $\boldsymbol{X}$-space: $\mathcal{X} = \left\{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(N)}\right\}$
- R replications for each $\boldsymbol{x}^{(i)} \in \mathcal{X}$: $\mathcal{Y}^{(i)} = \left\{y^{(i,1)}, y^{(i,2)}, \ldots, y^{(i,R)}\right\}$

Two approaches

Zhu & Sudret (2020) *Replication-based emulation of the response distribution of stochastic simulators using generalized lambda distributions*, Int. J. Uncertainty Quantification

- Infer-and-fit: infer a lambda distribution for each point $\boldsymbol{x}^{(i)}$ of the experimental design, then fit a sparse polynomial chaos expansion to the parameters $\boldsymbol{\lambda}$
- Joint inference: improve the previous results by maximum likelihood optimization

Local inference of lambda distributions (“Infer”)

Estimate $\boldsymbol{\lambda}^{(i)}$ based on $\mathcal{Y}^{(i)} = \left\{ y^{(i,1)}, \ldots, y^{(i,R)} \right\}$

- Maximum likelihood estimation

$$\hat{\boldsymbol{\lambda}}^{(i)} = \arg\max_{\boldsymbol{\lambda}} \sum_{r=1}^{R} \log \left(\frac{\lambda_2}{u_{i,r}^{\lambda_3 - 1} + (1 - u_{i,r})^{\lambda_4 - 1}} \right)$$

where

$$u_{i,r} = Q^{-1}\left(y^{(i,r)}; \boldsymbol{\lambda} \right), \quad y^{(i,r)} = Q(u_{i,r}; \boldsymbol{\lambda}) = \lambda_1 + \frac{1}{\lambda_2} \left(\frac{u_{i,r}^{\lambda_3} - 1}{\lambda_3} - \frac{(1 - u_{i,r})^{\lambda_4} - 1}{\lambda_4} \right)$$

- Nonlinear equation that can be solved numerically (Q is a monotonically increasing function)

Polynomial chaos expansions

- Fit 4 PCEs from the lambda data points $\Lambda = \left\{ \hat{\boldsymbol{\lambda}}^{(1)}, \ldots, \hat{\boldsymbol{\lambda}}^{(N)} \right\}$

Polynomial chaos expansions in a nutshell

Ghanem & Spanos (1991; 2003); Xiu & Karniadakis (2002); Soize & Ghanem (2004)

- We assume here for simplicity that the input parameters are independent with $X_i \sim f_{X_i}$, $i = 1, \ldots, M$
- PCE is also applicable in the general case using an isoprobabilistic transform $\boldsymbol{X} \mapsto \boldsymbol{\Xi}$

The polynomial chaos expansion for a deterministic simulator $\boldsymbol{X} \mapsto \mathcal{M}_d(\boldsymbol{X})$ reads:

$$\mathcal{M}_d(\boldsymbol{X}) = \sum_{\boldsymbol{\alpha} \in \mathbb{N}^M} a_{\boldsymbol{\alpha}} \, \Psi_{\boldsymbol{\alpha}}(\boldsymbol{X}) \qquad \Psi_{\boldsymbol{\alpha}}(\boldsymbol{X}) = \prod_{i=1}^{M} P_{\alpha_i}^{(i)}(X_i)$$

where:

- $\Psi_{\boldsymbol{\alpha}}(\boldsymbol{X})$ are basis functions (multivariate orthonormal polynomials)
- $a_{\boldsymbol{\alpha}}$ are coefficients to be computed (coordinates)

Polynomial chaos expansions

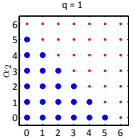
Truncation schemes

- "Full basis" of degree p:

$$\boldsymbol{\alpha} \in \mathcal{A}^{p,M} = \left\{ \boldsymbol{\alpha} \in \mathbb{N}^M ; \sum_{i=1}^{M} \alpha_i \leq p \right\}$$

- q-norm truncation:

$$\boldsymbol{\alpha} \in \mathcal{A}^{p,q,M} = \left\{ \boldsymbol{\alpha} \in \mathbb{N}^M ; \|\boldsymbol{\alpha}\|_q \overset{\text{def}}{=} \left(\sum_{i=1}^{M} \alpha_i^q \right)^{\frac{1}{q}} \leq p \right\}, \qquad 0 < q < 1$$



- Sparse expansion: $\mathcal{A}$ only contains relevant basis functions taken out of a candidate basis:
  - Least-angle regression
  - Orthogonal matching pursuit, subspace pursuit

  etc.

Fitting polynomial chaos expansions ("Fit")

- Gather the data from the "Fit"-step: $\left\{\left(\boldsymbol{x}^{(1)}, \boldsymbol{\lambda}^{(1)}\right), \left(\boldsymbol{x}^{(2)}, \boldsymbol{\lambda}^{(2)}\right), \ldots, \left(\boldsymbol{x}^{(N)}, \boldsymbol{\lambda}^{(N)}\right)\right\}$
- Represent the distribution parameters $\boldsymbol{\lambda}$ by polynomial chaos expansions

$$\lambda_k(\boldsymbol{x}) \approx \lambda_k^{\mathsf{PC}}(\boldsymbol{x}; \boldsymbol{a}) = \sum_{\boldsymbol{\alpha} \in \mathcal{A}_k} a_{k,\boldsymbol{\alpha}} \psi_{\boldsymbol{\alpha}}(\boldsymbol{x}) \qquad \text{for } k = 1, 3, 4$$

$$\lambda_2(\boldsymbol{x}) \approx \lambda_2^{\mathsf{PC}}(\boldsymbol{x}; \boldsymbol{a}) = \exp\left(\sum_{\boldsymbol{\alpha} \in \mathcal{A}_2} a_{2,\boldsymbol{\alpha}} \psi_{\boldsymbol{\alpha}}(\boldsymbol{x})\right)$$

- PCE coefficients are calibrated by the sparse regression method Hybrid-LARS

Blatman & Sudret (2011) *Adaptive sparse polynomial chaos expansion based on Least Angle Regression*, J. Comput. Phys.

Improvement: joint fitting

Rationale

- The results of the Infer-and-fit algorithm depends on the accuracy of the local inference
- **Idea:** build a global model for the joint distribution of inputs and outputs:

$$f_{\boldsymbol{X},Y}(\boldsymbol{x}, y) = f_{Y|\boldsymbol{X}}\left(y \mid \boldsymbol{x}\right) \cdot f_{\boldsymbol{X}}(\boldsymbol{x})$$

where the conditional PDF is represented by a lambda model:

$$f^{\text{GLD}}_{\boldsymbol{X},Y}(\boldsymbol{x}, y; \boldsymbol{a}) = f^{\text{GLD}}_{Y|\boldsymbol{X}}\left(y; \boldsymbol{\lambda}^{\text{PC}}(\boldsymbol{x}; \boldsymbol{a})\right) \cdot f_{\boldsymbol{X}}(\boldsymbol{x})$$

Procedure

Find the optimal PCE coefficients $\boldsymbol{a}$ that minimize the Kullback-Leibler divergence between $f_{\boldsymbol{X},Y}$ and $f^{\text{GLD}}_{\boldsymbol{X},Y}$:

$$\hat{\boldsymbol{a}} = \arg\min_{\boldsymbol{a}} D_{\text{KL}}\left(f_{\boldsymbol{X},Y} \,||\, f^{\text{GLD}}_{\boldsymbol{X},Y}(\,.\,;\boldsymbol{a})\right)$$

Corresponding loss function

Solution

After some basic algebra, the minimization problem reduces to:

$$\hat{\boldsymbol{a}} = \arg\min_{\boldsymbol{a}} \mathbb{E}_{\boldsymbol{X},Y}\left[\log \hat{f}_{Y|\boldsymbol{X}}\left(Y;\, \boldsymbol{\lambda}^{\mathsf{PC}}(\boldsymbol{X};\, \boldsymbol{a})\right)\right]$$

Estimator

- The PCE bases for $\boldsymbol{\lambda}^{\mathsf{PC}}(\boldsymbol{x})$ are taken from the “Infer-and-Fit” approach
- The expectation w.r.t. $\boldsymbol{X}$ and Y is computed from the experimental design $\mathcal{X} = \left\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(N)}\right\}$, with R replications in each point (outputs $y^{(i,r)}$):

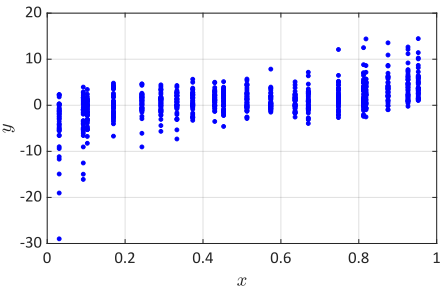
$$\hat{\boldsymbol{a}} = \arg\min_{\boldsymbol{a}} \frac{1}{N\,R} \sum_{i=1}^{N} \sum_{r=1}^{R} \log \hat{f}_{Y|\boldsymbol{X}}\left(y^{(i,r)};\, \boldsymbol{\lambda}^{\mathsf{PC}}(\boldsymbol{x}^{(i)};\, \boldsymbol{a})\right)$$

Illustration of the algorithm

Toy example

- Analytical lambda model

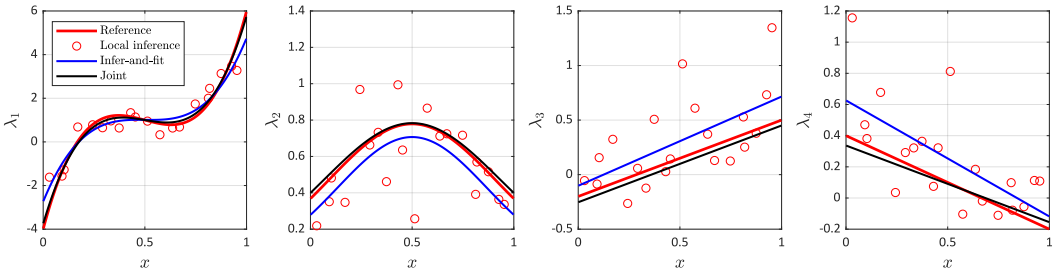
$$\lambda_1(x) = 50x^3 - 75x^2 + 35x - 4$$
$$\lambda_2(x) = \exp(-3x^2 + 3x - 1)$$
$$\lambda_3(x) = -0.2 + 0.7x$$
$$\lambda_4(x) = 0.4 - 0.6x$$



- Experimental design of $N = 20$, replications $R = 40$

Illustration of the algorithm

Joint fitting



Outline

Introduction

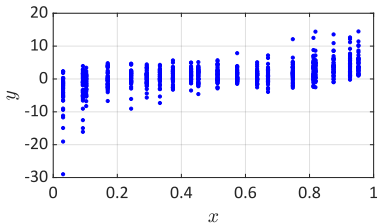
Goal

Propose a framework to build a stochastic emulator without replications (... that can also be applied if there is replicated data)

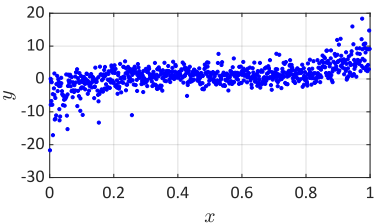
Rationale

The joint fitting estimator does not require replications!

$$\hat{\boldsymbol{a}} = \arg\min_{\boldsymbol{a}} \mathbb{E}_{\boldsymbol{X},Y}\left[\log \hat{f}_{Y|\boldsymbol{X}}\left(Y; \boldsymbol{\lambda}^{\mathsf{PC}}(\boldsymbol{X}; \boldsymbol{a})\right)\right]$$



With



Without

Procedure

Ingredients

- An experimental design $\mathcal{X} = \left\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(N)}\right\}$ and model evaluations $y^{(i)} \overset{\text{def}}{=} \mathcal{M}(\boldsymbol{x}^{(i)}, \omega_i)$
- Pre-selected PC bases for the four polynomial chaos expansions of $\lambda_i(\boldsymbol{x})$, $i = 1, \ldots, 4$

Selection of PCE bases

- PCE models for the mean and variance of the model output built using the feasible generalized least-square method
- Use the PCE basis of $\mu(\boldsymbol{x})$ (resp. $\log \sigma^2(\boldsymbol{x})$) for λ_1 (resp. λ_2)
- PCE of degree 1 for λ_3 and λ_4 (it is assumed that the shape of the response distribution does not vary nonlinearly with $\boldsymbol{x}$)

Algorithm 1: Feasible generalized least-squares (FGLS)

Input: Computational budget N

Initialization

Experimental design $\mathcal{X} = \{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(N)}\}$

Model evaluations $\mathcal{Y} = \{y^{(i)} \overset{\text{def}}{=} \mathcal{M}(\boldsymbol{x}(i), \omega_i), i = 1, \ldots, N\}$

Set $k = 0$. Estimate the mean function $\hat{\mu}_0(\boldsymbol{x})$ by ordinary least-squares % Fix the basis after (p, q) search

FGLS

while *NotConverged* **do**

Subtract the estimated mean from the data, *i.e.* $r^{(i)} = y^{(i)} - \hat{\mu}_k\left(\boldsymbol{x}^{(i)}\right)$

Estimate the variance function $\hat{\sigma}^2_k(\boldsymbol{x})$ based on $\left\{\left(\boldsymbol{x}^{(i)}, 2\log\left|r^{(i)}\right|\right), i = 1, \ldots, N\right\}$

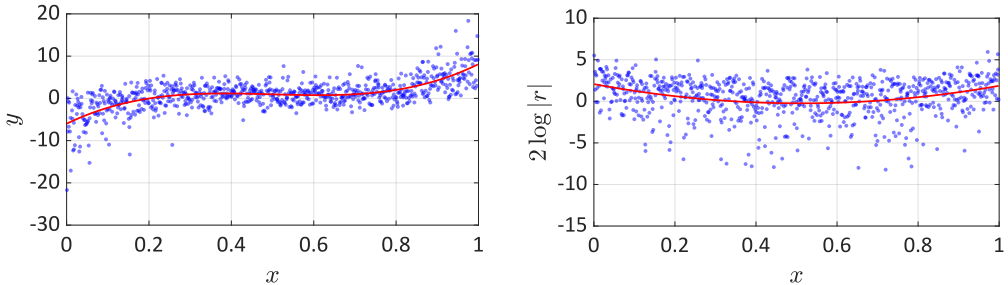
$k \leftarrow k + 1$

Use $\hat{\sigma}^2_k(\boldsymbol{x})$ as weight to estimate $\hat{\mu}_k(\boldsymbol{x})$ by weighted least-squares

end

Return PCE expansions of $\mu(\boldsymbol{x})$ and $\log\sigma^2(\boldsymbol{x})$

Feasible generalized least-squares (FGLS): illustration



Iteration 10

Outline

Error metrics for stochastic emulators

Wasserstein distance

- It is the L^2 distance between the quantile functions:

$$d^2_{\mathrm{WS}}(Y, \hat{Y}) \overset{\text{def}}{=} \|Q_Y - \hat{Q}_Y\|^2_{L^2} = \int_0^1 \left(Q_Y(u) - \hat{Q}_Y(u)\right)^2 du$$

Normalized Wasserstein distance

$$\varepsilon = \frac{\mathbb{E}_{\boldsymbol{X}}\left[d^2_{\mathrm{WS}}\left(Y(\boldsymbol{X}), Y^{\mathsf{GLaM}}(\boldsymbol{X})\right)\right]}{\mathrm{Var}\left[Y\right]}$$

Geometric Brownian motion

$$\mathrm{d}S_t = x_1\, S_t\, \mathrm{d}t + x_2\, S_t\, \mathrm{d}W_t$$

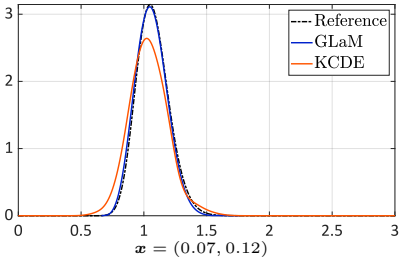
- S_t: stock price, W_t: Wiener process
- x_1: drift, x_2: volatility
- The analytical distribution of S_t exists (Itô's calculus)

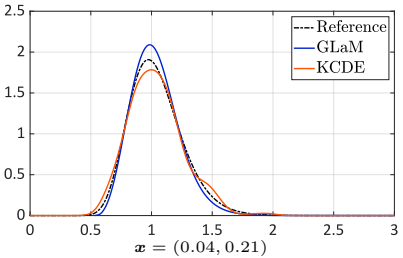
$$S_t(\boldsymbol{x})/S_0 \sim \mathcal{LN}\left(\left(x_1 - x_2^2/2\right) t, x_2\, t\right)$$

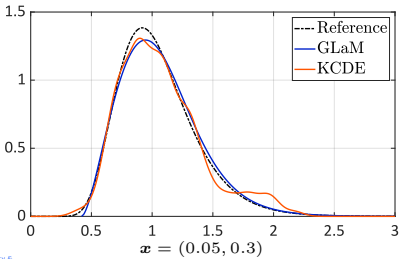
Setup

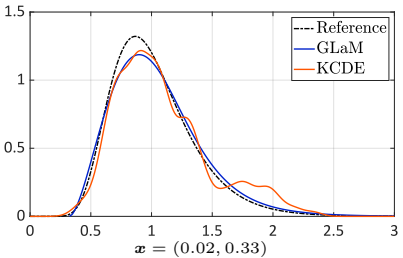
- $X_1 \sim \mathcal{U}(0, 0.1)$, $X_2 \sim \mathcal{U}(0.1, 0.4)$
- $Y = S_1$ for $t = 1$ is of interest, *i.e.* $Y(\boldsymbol{x}) \sim \mathcal{LN}\left(x_1 - x_2^2/2, x_2\right)$
- Experimental design: $\boldsymbol{\mathcal{X}}$ are generated using the Latin hypercube sampling
- No replications

PDF predictions (ED with $N = 500$)





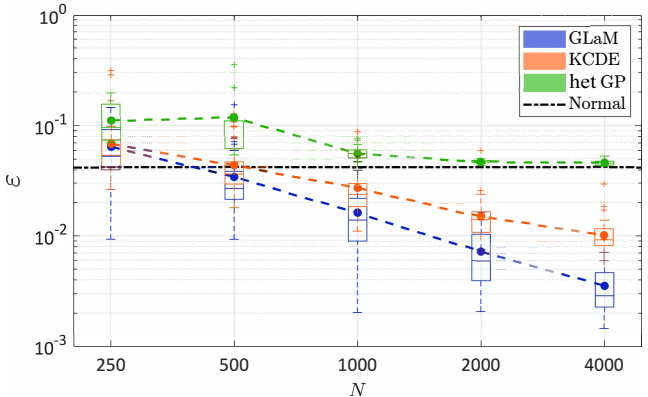




Convergence study

- Experimental design of size 250, 500, 1000, 2000, 4000
- 50 independent runs for each scenario
- Normalized Wasserstein distance as a performance indicator

$$\varepsilon = \frac{\mathbb{E}_{\boldsymbol{X}}\left[d_{\mathrm{WS}}^2\left(Y(\boldsymbol{X}), Y^{\mathsf{GLaM}}(\boldsymbol{X})\right)\right]}{\mathrm{Var}\left[Y\right]}$$



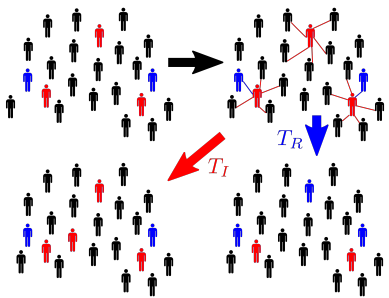
Outline

Stochastic SIR model

- $M_t = S_t + I_t + R_t$: total population
- S_t: number of susceptible individuals at time t
- I_t: number of infected individuals at time t
- R_t: number of recovered individuals at time t

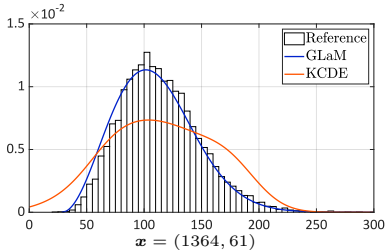
Setup

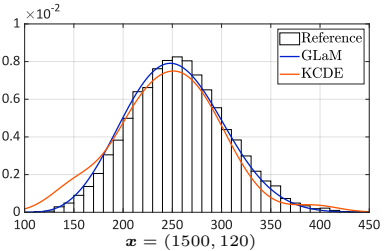
- Total population $M_t = 2000$
- The initial condition: $S_0 \sim \mathcal{U}(1300, 1800)$, $I_0 \sim \mathcal{U}(20, 200)$
- $Y(\boldsymbol{x})$: total number of infected individuals during the outbreak (without counting I_0)

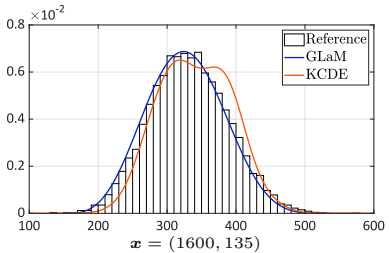


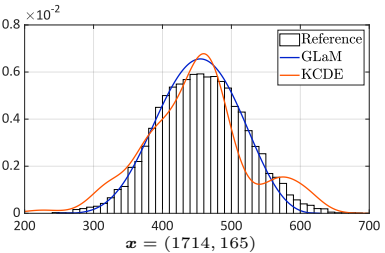
Binois *et al.* (2018) *Practical heteroscedastic Gaussian process modeling for large simulation experiments*, J. Comput. Graph. Stat.

PDF predictions (ED with $N = 500$)





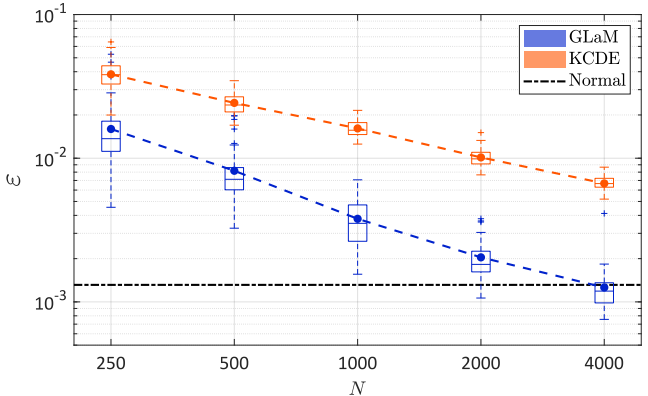




Convergence study

- Experimental design of size 250, 500, 1000, 2000, 4000
- 50 independent runs for each scenario
- Normalized Wasserstein distance as a performance indicator

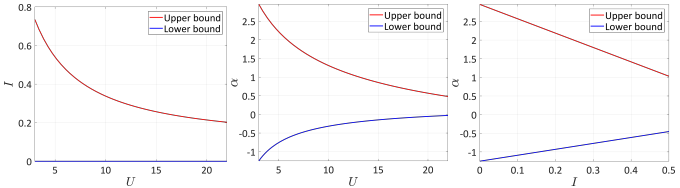
$$\varepsilon = \frac{\mathbb{E}_{\boldsymbol{X}}\left[d_{\mathrm{WS}}^2\left(Y(\boldsymbol{X}), Y^{\mathrm{GLaM}}(\boldsymbol{X})\right)\right]}{\mathrm{Var}\left[Y\right]}$$

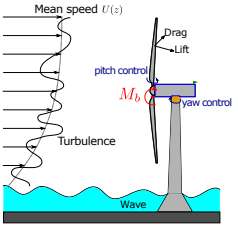


Outline

Wind turbine application

- **Five input variables:**
  - Mean speed U, turbulence intensity I and shear exponent α are uniformly distributed in the following domain





  - Air density $\rho \sim \mathcal{U}(0.8, 1.4)$, inclination angle $\beta \sim \mathcal{U}(-10, 10)$

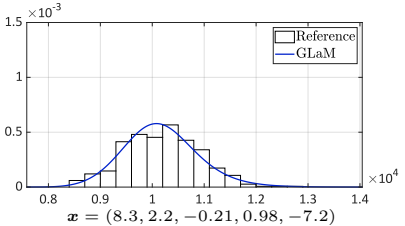
- **Model output:** maximum flapwise bending moment $Y = M_b$

Simulation

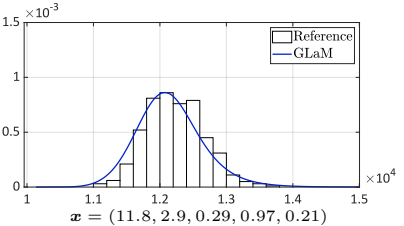
- 485 training points with 50 replications
- 120 test points with 500 replications

PDF predictions

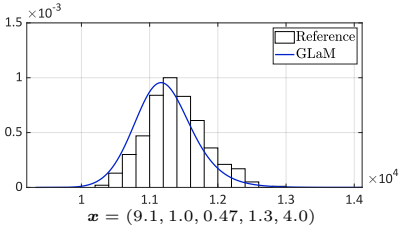
The normalized Wasserstein distance is $\varepsilon = 0.013$



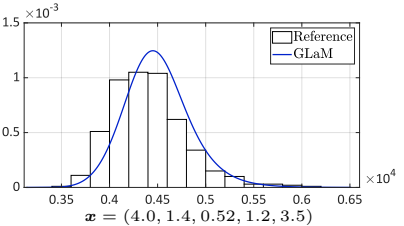
$\boldsymbol{x} = (8.3, 2.2, -0.21, 0.98, -7.2)$



$\boldsymbol{x} = (11.8, 2.9, 0.29, 0.97, 0.21)$



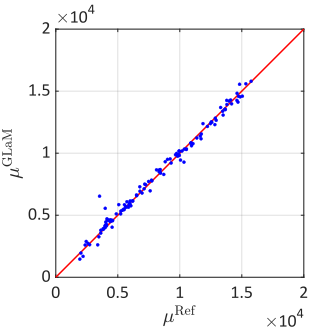
$\boldsymbol{x} = (9.1, 1.0, 0.47, 1.3, 4.0)$



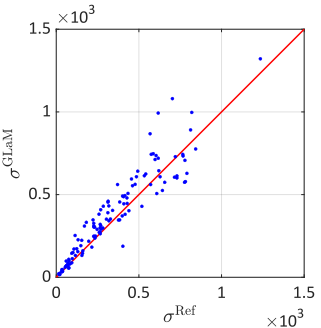
$\boldsymbol{x} = (4.0, 1.4, 0.52, 1.2, 3.5)$

Predictions of quantities of interest

Mean and standard deviation



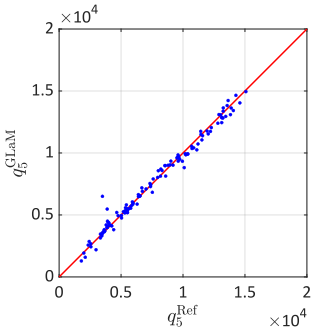
$R^2 = 0.987$



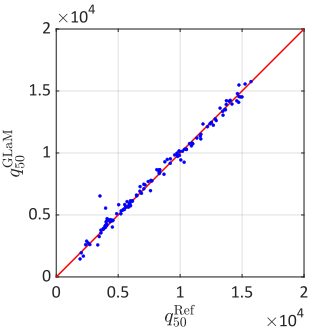
$R^2 = 0.836$

QoI predictions

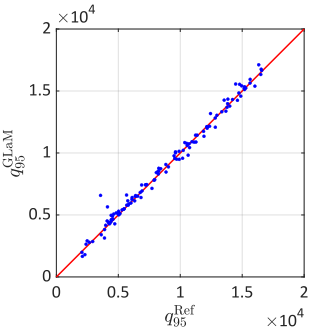
Quantiles



$R^2 = 0.983$



$R^2 = 0.987$



$R^2 = 0.988$

One word on sensitivity analysis

Due to the stochasticity in the simulator $\mathcal{M}_S(\boldsymbol{x}) \equiv \mathcal{M}_d(\boldsymbol{x}, \boldsymbol{z})$, various Sobol' indices can be defined:

- Classical indices based on partial variances:

$$S_{\mathbf{u}} = \frac{\mathrm{Var}\left[\mathbb{E}\left[Y|\boldsymbol{X}_{\mathbf{u}}\right]\right]}{\mathrm{Var}\left[Y\right]} \qquad \mathbf{u} = \{i_1, \ldots, i_k\} \subset \{1, \ldots, M\}$$

- QoI-based indices, corresponding to a particular feature of $Y(\boldsymbol{x})$, *e.g.* mean, variance, quantile

$$S_{\mathbf{u}}^{\mathrm{QoI}} = \frac{\mathrm{Var}\left[\mathbb{E}\left[\mathrm{QoI}(\boldsymbol{X})|\boldsymbol{X}_{\mathbf{u}}\right]\right]}{\mathrm{Var}\left[\mathrm{QoI}(\boldsymbol{X})\right]}, \qquad \mathrm{QoI}(\boldsymbol{x}) \overset{e.g.}{=} \mu_Y(\boldsymbol{x}), \sigma_Y^2(\boldsymbol{x}), q_\alpha(\boldsymbol{x})$$

- Trajectory-based indices that are conditioned on the latent variables $\boldsymbol{Z}$:

$$S_{\mathbf{u}}^{traj}(\boldsymbol{Z}) = \frac{\mathrm{Var}\left[Y \mid \boldsymbol{X}_{\mathbf{u}}, \boldsymbol{Z}\right]}{\mathrm{Var}\left[Y \mid \boldsymbol{Z}\right]}$$

See details in: Zhu & Sudret, Global sensitivity analysis for stochastic simulators based on generalized lambda surrogate models, Reliab. Eng. Sys. Safety (2021)

Conclusions

- Stochastic simulators are used in many fields of applied sciences and engineering
- Building general-purpose emulators is necessary for optimization, sensitivity analysis, etc.
- We propose a framework based on generalized lambda distributions and polynomial chaos expansions
- Replications are not mandatory ... but can be used !
- General-purpose algorithms are available using bricks from UQLab

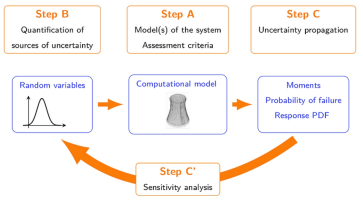
UQLab

The Framework for Uncertainty Quantification

OVERVIEW FEATURES DOCUMENTATION DOWNLOAD/INSTALL ABOUT COMMUNITY

"Make uncertainty quantification available for anybody, in any field of applied science and engineering"

www.uqlab.com



- MATLAB®-based Uncertainty Quantification framework
- State-of-the art, highly optimized open source algorithms
- Fast learning curve for beginners
- Modular structure, easy to extend
- Exhaustive documentation

UQLab: The Uncertainty Quantification Software

- **free access to academia**
- 3,700 registered users
- 1,450+ active users from 92 countries

http://www.uqlab.com

- The cloud version of UQLab, accessible via an API (SaaS)
- **Available with python bindings for beta testing**

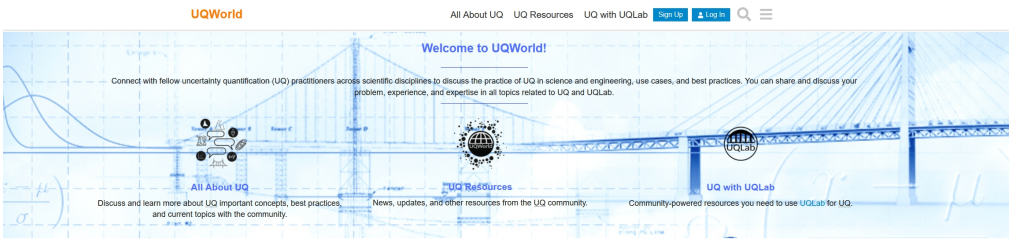
https://uqpylab.uq-cloud.io/

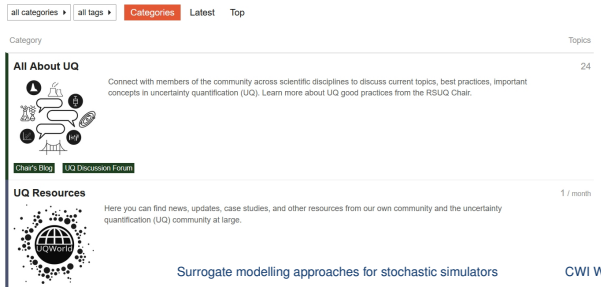
Country	# Users
United States	590
China	512
France	345
Switzerland	292
Germany	275
United Kingdom	157
Italy	148
Brazil	130
India	123
Canada	88

As of June 15, 2021

UQWorld: the community of UQ

https://uqworld.org/





Questions ?

Chair of Risk, Safety & Uncertainty Quantification

www.rsuq.ethz.ch

The Uncertainty Quantification Software

www.uqlab.com

The Uncertainty Quantification Community

www.uqworld.org

References

Ankenman, B., B. Nelson, and J. Staum (2010).
Stochastic Kriging for simulation metamodeling.
Oper. Res. 58, 371–382.

Azzi, S., Y. Huang, B. Sudret, and J. Wiart (2019).
Surrogate modeling of stochastic functions – application to computational electromagnetic dosimetry.
Int. J. Uncertainty Quantification 9(4), 351–363.

Azzi, S., B. Sudret, and J. Wiart (2020).
Sensitivity analysis for stochastic simulators using differential entropy.
Int. J. Uncertainty Quantification 10(1), 25–33.

Binois, M., R. B. Gramacy, and M. Ludkovski (2018).
Practical heteroscedastic Gaussian process modeling for large simulation experiments.
J. Comput. Graph. Stat. 27(4), 808–821.

Blatman, G. and B. Sudret (2011).
Adaptive sparse polynomial chaos expansion based on Least Angle Regression.
J. Comput. Phys 230, 2345–2367.

Freimer, M., G. Kollia, G. S. Mudholkar, and C. T. Lin (1988).
A study of the generalized Tukey lambda family.
Comm. Stat. Theor. Meth. 17, 3547–3567.

Goldberg, P., C. Williams, and C. M. Bishop (1997).
Regression with input-dependent noise: a gaussian process treatment.
In *Proc. 10th Int. Conf. on Advances in neural information processing systems (NIPS10), Colorado, USA*, pp. 493–499.

Marrel, A., B. Iooss, S. Da Veiga, and M. Ribatet (2012).
Global sensitivity analysis of stochastic computer models with joint metamodels.
Stat. Comput. 22, 833–847.

Moutoussamy, V., S. Nanty, and B. Pauwels (2015).
Emulators for stochastic simulation codes.
ESAIM: Math. Model. Num. 48, 116–155.

Xiu, D. and G. E. Karniadakis (2002).
The Wiener-Askey polynomial chaos for stochastic differential equations.
SIAM J. Sci. Comput. 24(2), 619–644.

Zhu, X. and B. Sudret (2020).
Replication-based emulation of the response distribution of stochastic simulators using generalized lambda distributions.
Int. J. Uncertainty Quantification 10(3), 249–275.

Zhu, X. and B. Sudret (2021a).
Emulation of stochastic simulators using generalized lambda models.
Submitted to SIAM/ASA J. Unc. Quant..
(ArXiv: 2007.00996).

Zhu, X. and B. Sudret (2021b).
Global sensitivity analysis for stochastic simulators based on generalized lambda surrogate models.
Reliab. Eng. Sys. Safety 214(107815).

Zhu, X. and B. Sudret (2021c).
Stochastic polynomial chaos expansions for emulating stochastic simulators.
Submitted.