

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/352993637>

Recent developments on surrogate models for stochastic simulators

Presentation · June 2021

DOI: 10.3929/ethz-b-000492981

CITATIONS
0

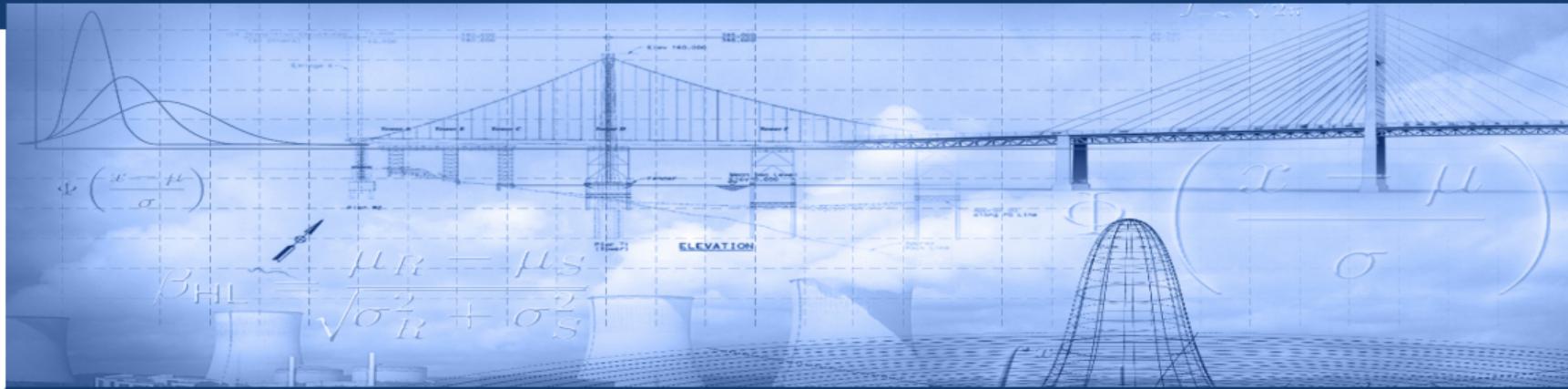
READS
584

1 author:



Bruno Sudret
ETH Zurich
438 PUBLICATIONS 17,800 CITATIONS

[SEE PROFILE](#)



Recent Developments on Surrogate Models for Stochastic Simulators

Bruno Sudret

Chair of Risk, Safety and Uncertainty Quantification, ETH Zurich

How to cite?

This presentation is a plenary lecture given at the 4th International Conference on Uncertainty Quantification in Computational Sciences and Engineering on June 30, 2021.

How to cite ?

Sudret, B., *Recent Developments on Surrogate Models for Stochastic Simulators*,
4th International Conference on Uncertainty Quantification in Computational Sciences and
Engineering, streamed from Athens (Greece), June 30th, 2021.

The SAMOS project

- The SAMOS project (“SurrogAte Modelling for stOchastic Simulators”) is funded by the Swiss National Science Foundation under Grant # 175524 (May 2018 - April 2022).



Xujia Zhu

Use of generalized lambda distributions and sparse polynomial chaos expansions



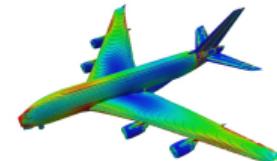
Nora Lüthen

UNCECOMP'2021 paper U18995: “[Surrogating stochastic simulators using Karhunen-Loève expansion, sparse PCE and advanced statistical modelling](#)”

Computational models in engineering

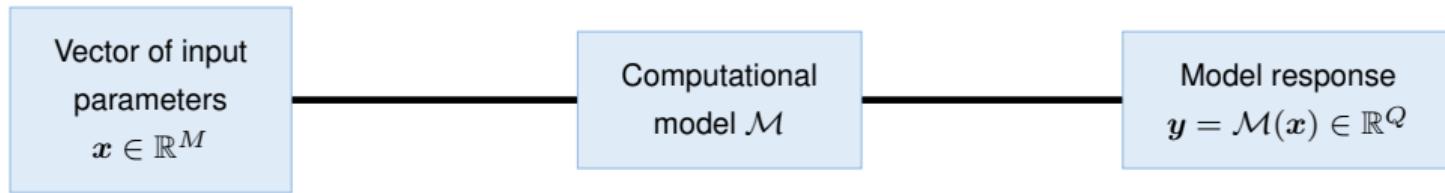
Computational models in science and engineering are used:

- 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



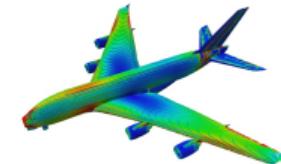
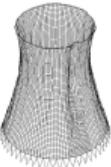
Computational models: the abstract viewpoint

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



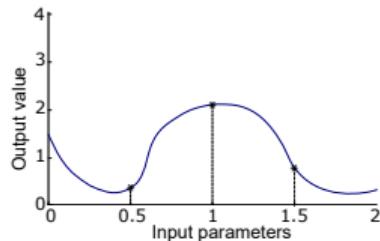
Allows for purely **data-driven** approaches based on $\{\mathcal{X}, \mathcal{M}(\mathcal{X})\}$

Deterministic vs. stochastic simulators



Deterministic simulators

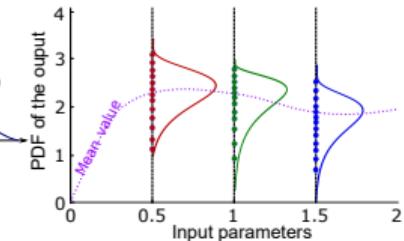
$x \rightarrow$ Deterministic simulator $\rightarrow y$



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

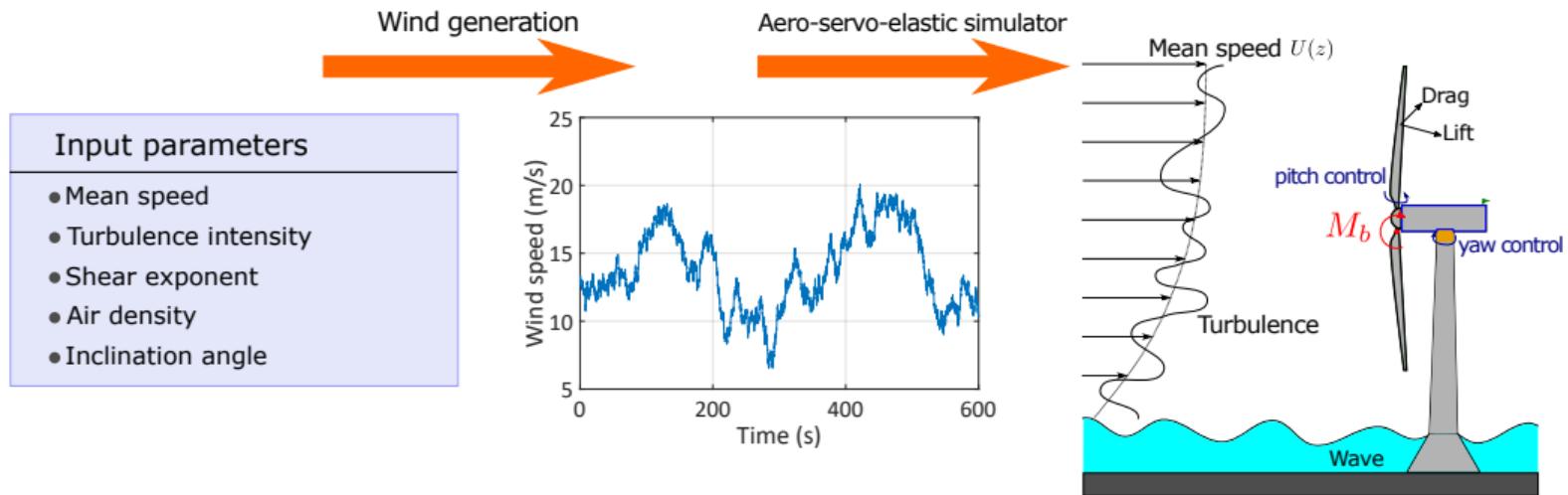
Stochastic simulators

$x \rightarrow$ Stochastic simulator $\rightarrow Y(x)$



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

Example: wind turbine simulation



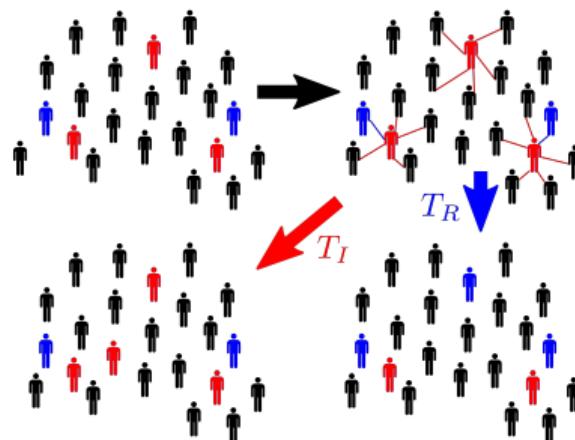
Example: epidemiology

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

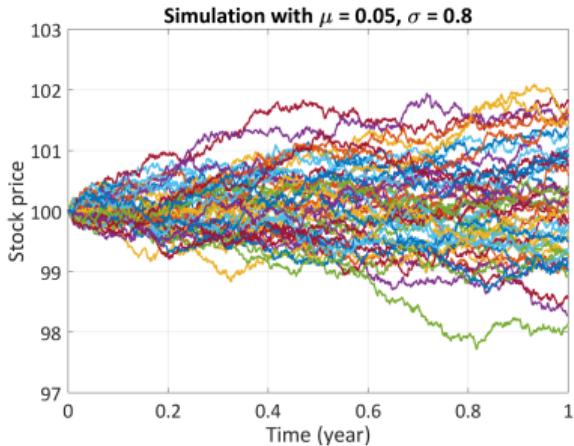


Example: mathematical finance

Geometrical Brownian motion

$$dS_t = \mu S_t dt + \sigma S_t dW_t$$

- 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

$$C = \max \{A_T - K, 0\}, \text{ with } A_t = \frac{1}{t} \int_0^t S_u du.$$

Outline

Introduction and literature

Generalized lambda distributions

Generalized lambda/PCE models

With replications

Without replications

Application examples

Analytical example

Stochastic SIR model

Wind turbine application

Conclusions & outlook

Formal definition

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

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

where \mathcal{D}_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 $Y|X = \boldsymbol{x}_0 \equiv \mathcal{M}_s(\boldsymbol{x}_0, \omega)$
- 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} :

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

Computational costs induced by stochastic simulators

- Replications are needed to estimate the PDF of $Y|X = x$
- Many runs must be carried out by varying 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** $\tilde{\mathcal{M}}$ (a.k.a. **emulator**) is an **approximation** of the original computational model \mathcal{M} :

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

with the following features:

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

Simulated data

- It is **fast to evaluate!**

Surrogate models for uncertainty quantification

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

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 $\varepsilon = \mathbb{E} \left[(\mathcal{M}(\mathbf{X}) - \tilde{\mathcal{M}}(\mathbf{X}))^2 \right]$

Back to stochastic simulators

Our goal is to develop emulators that are:

- **Data-driven** (*i.e.* non-intrusive w.r.t. to the original simulator)
- **General-purpose**: no restrictive assumption (*e.g.* Gaussian) on the family of the output distribution is made
- Able to tackle the full distribution of $Y|X = x$, but also **quantities of interest** (*e.g.* mean, variance, quantiles)
- Providing a representation of $Y|X = x$ 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 ¹
- Random field representations ²

¹ Yan & Perdikaris (2019), Comput. Mech.

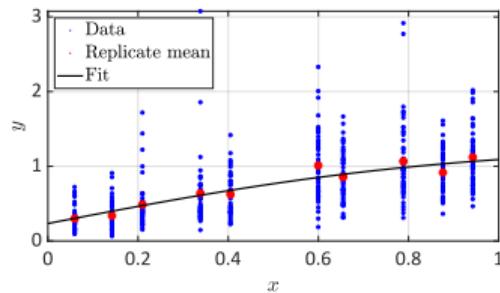
Zhu & Sudret, EMI/PMC Conference, Columbia University, 2021

² Azzi *et al.* (2019, 2020), Int. J. Uncertainty Quantification; Lüthen *et al.*, UNCECOMP'2021

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 $\mu(x)$ and log-variance function $\log(V(x))$ 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 (X, Y) by kernel smoothing, then compute the conditional PDF by:

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

- 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.

Outline

Introduction and literature

Generalized lambda distributions

Generalized lambda/PCE models

Application examples

Conclusions & outlook

Definition

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

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

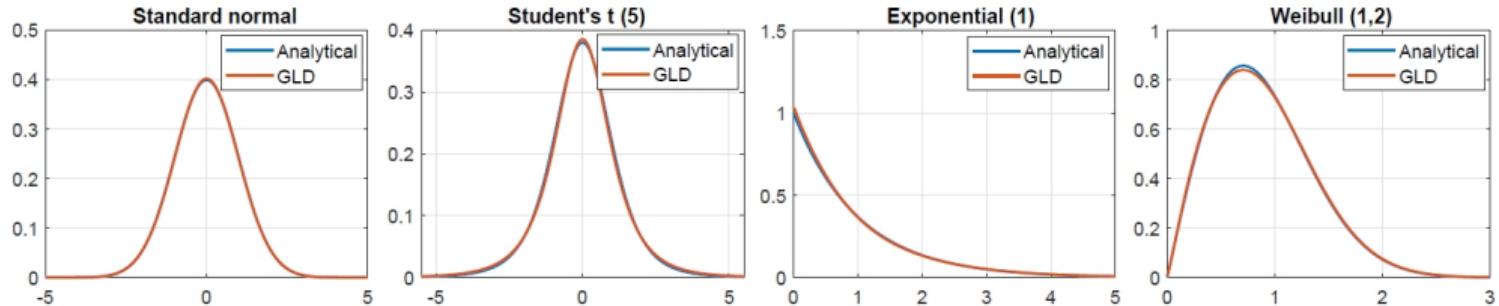
where:

- 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:

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

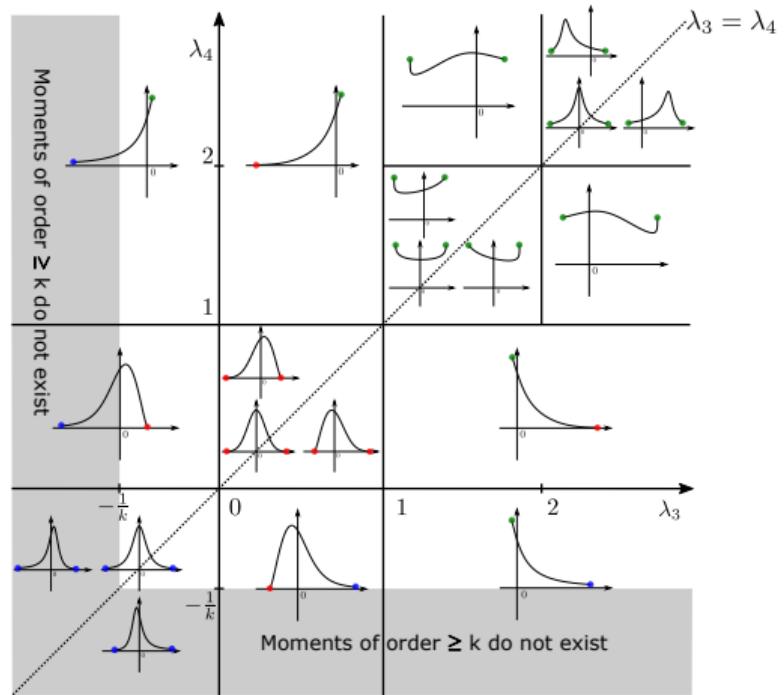
Properties



- 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 $\text{PDF} = 0$ at the bound
- **Green points:** finite support, with $\text{PDF} \neq 0$ 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

Introduction and literature

Generalized lambda distributions

Generalized lambda/PCE models

With replications

Without replications

Application examples

Conclusions & outlook

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 X -space: $\mathcal{X} = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \dots, \boldsymbol{x}^{(N)}\}$
- R replications for each $\boldsymbol{x}^{(i)} \in \mathcal{X}$: $\mathcal{Y}^{(i)} = \{y^{(i,1)}, y^{(i,2)}, \dots, y^{(i,R)}\}$

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 λ
- **Joint inference**: improve the previous results by maximum likelihood optimization

Local inference of lambda distributions (“Infer”)

Estimate $\lambda^{(i)}$ based on $\mathcal{Y}^{(i)} = \{y^{(i,1)}, \dots, y^{(i,R)}\}$

- 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

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

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

Polynomial chaos expansions

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

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 $X_i \sim f_{X_i}$, $i = 1, \dots, M$
- 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:

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

where:

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

Polynomial chaos expansions

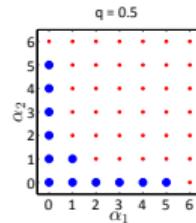
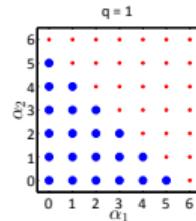
Truncation schemes

- “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:

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



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: $\{(x^{(1)}, \lambda^{(1)}), (x^{(2)}, \lambda^{(2)}), \dots, (x^{(N)}, \lambda^{(N)})\}$
- Represent the distribution parameters λ by polynomial chaos expansions

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

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

- 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:

$$f_{\mathbf{X},Y}(\mathbf{x}, y) = f_{Y|\mathbf{X}}(y | \mathbf{x}) \cdot f_{\mathbf{X}}(\mathbf{x})$$

where the conditional PDF is represented by a lambda model:

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

Procedure

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

$$\hat{\mathbf{a}} = \arg \min_{\mathbf{a}} D_{\text{KL}}(f_{\mathbf{X},Y} || f_{\mathbf{X},Y}^{\text{GLD}}(\cdot; \mathbf{a}))$$

Corresponding loss function

Solution

After some basic algebra, the minimization problem reduces to:

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

Estimator

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

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

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 $R = 40$

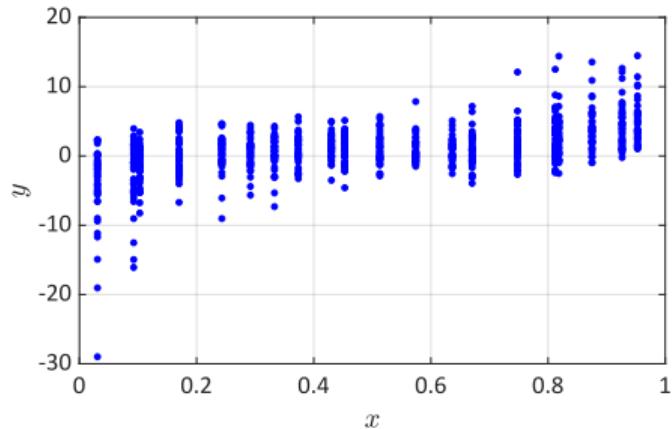
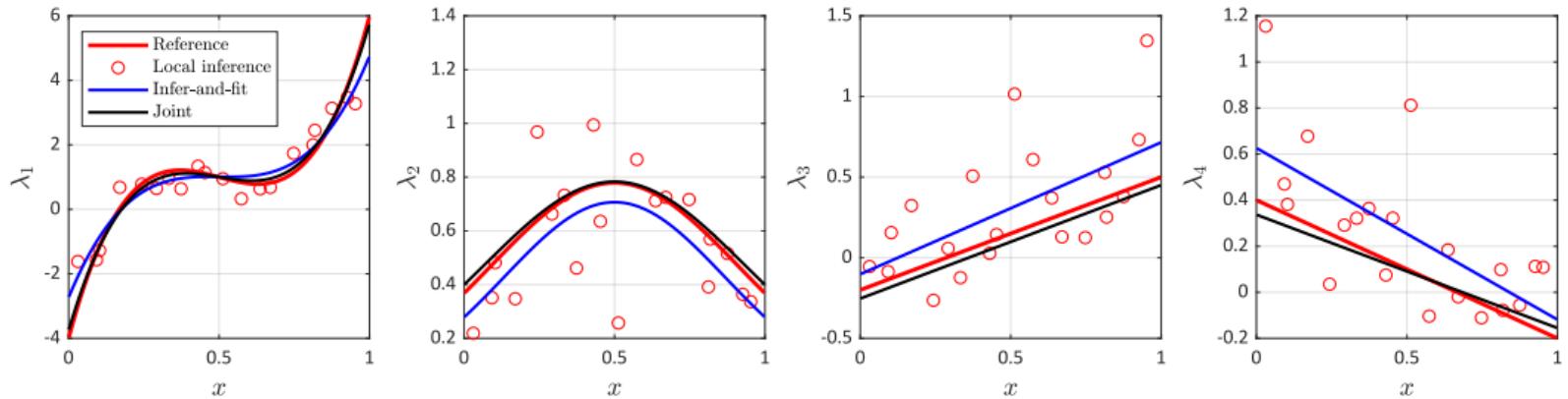


Illustration of the algorithm

Joint fitting



Outline

Introduction and literature

Generalized lambda distributions

Generalized lambda/PCE models

With replications

Without replications

Application examples

Conclusions & outlook

Introduction

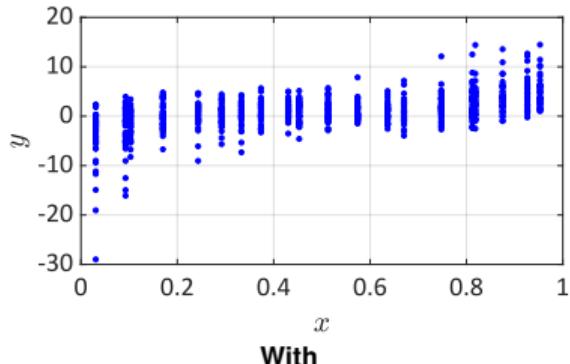
Goal

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

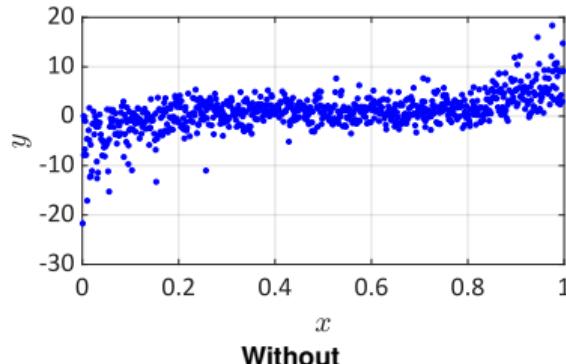
Rationale

The joint fitting estimator does not require replications!

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



With



Without

Procedure

Ingredients

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

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})

Outline

Introduction and literature

Generalized lambda distributions

Generalized lambda/PCE models

Application examples

Analytical example

Stochastic SIR model

Wind turbine application

Conclusions & outlook

Error metrics for stochastic emulators

Wasserstein distance

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

$$d_{\text{WS}}^2(Y, \hat{Y}) \stackrel{\text{def}}{=} \|Q_Y - \hat{Q}_Y\|_{L^2}^2 = \int_0^1 (Q_Y(u) - \hat{Q}_Y(u))^2 du$$

Normalized Wasserstein distance

$$\varepsilon = \frac{\mathbb{E}_{\mathbf{X}} [d_{\text{WS}}^2 (Y(\mathbf{X}), Y^{\text{GLaM}}(\mathbf{X}))]}{\text{Var}[Y]}$$

Geometric Brownian motion

$$dS_t = x_1 S_t dt + x_2 S_t dW_t$$

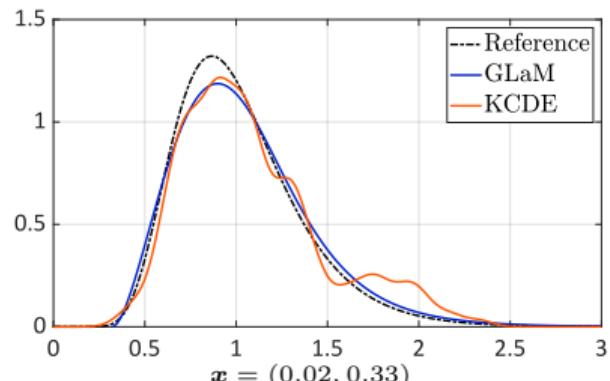
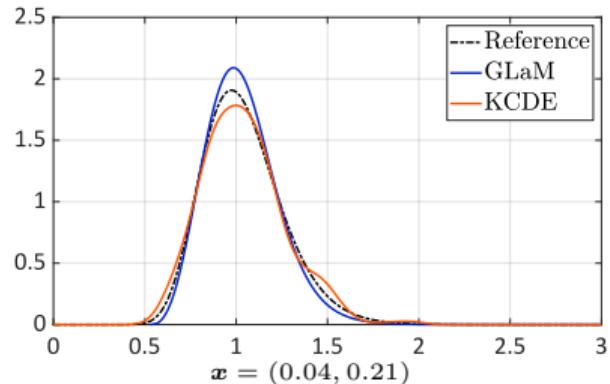
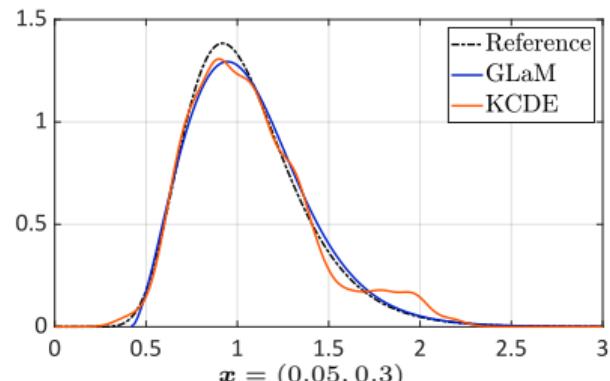
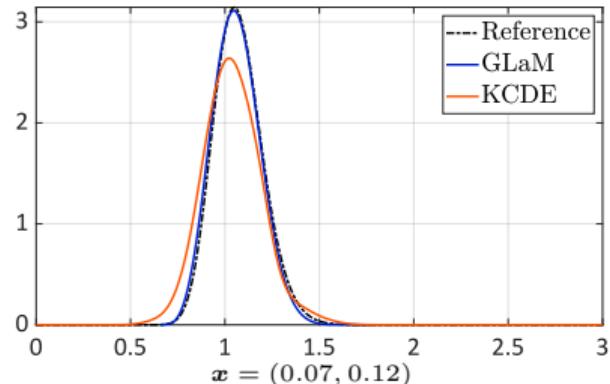
- S_t : stock price, W_t : Wiener process
- x_1 : drift, x_2 : volatility
- The analytical distribution of S_t exists (Itô's calculus)

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

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: \mathcal{X} are generated using the Latin hypercube sampling
- No replications

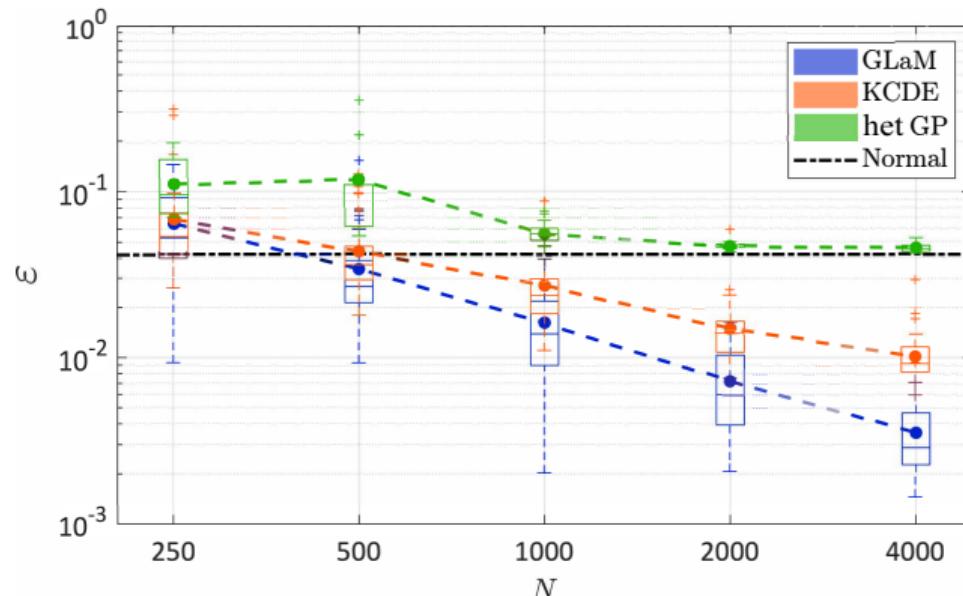
PDF predictions (ED with $N = 500$)



Convergence study

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

$$\varepsilon = \frac{\mathbb{E}_{\mathbf{X}} [d_{\text{WS}}^2 (Y(\mathbf{X}), Y^{\text{GLaM}}(\mathbf{X}))]}{\text{Var}[Y]}$$



Outline

Introduction and literature

Generalized lambda distributions

Generalized lambda/PCE models

Application examples

Analytical example

Stochastic SIR model

Wind turbine application

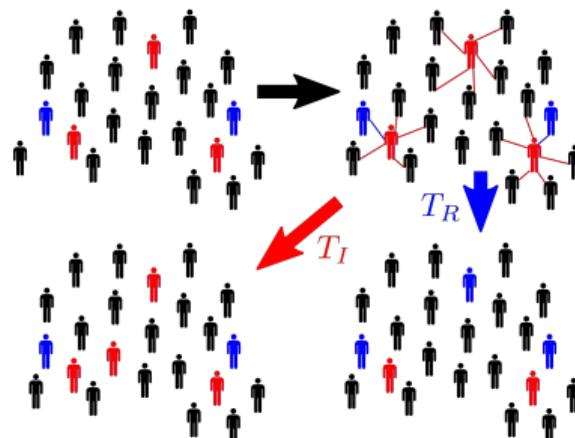
Conclusions & outlook

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

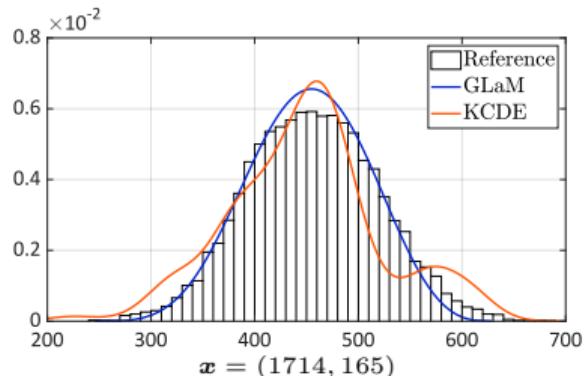
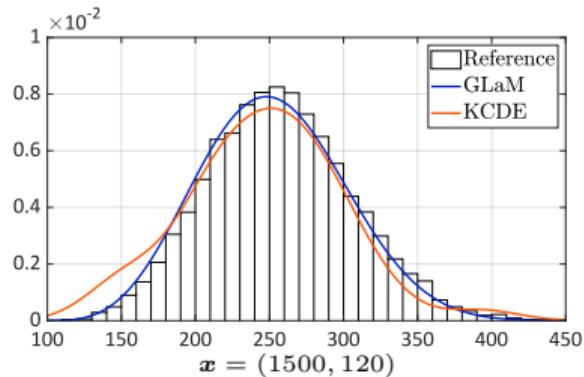
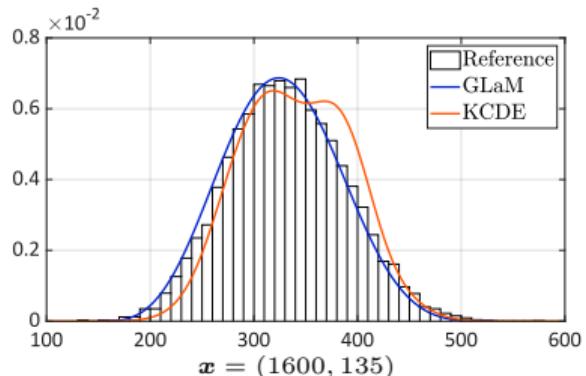
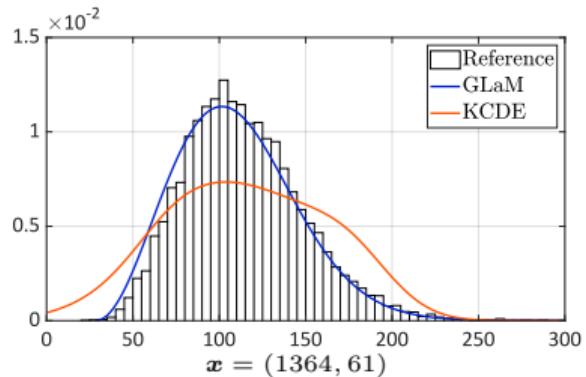
Setup

- Total population $M_t = 2000$
- The initial condition: $S_0 \sim \mathcal{U}(1300, 1800)$,
 $I_0 \sim \mathcal{U}(20, 200)$
- $Y(x)$: **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.

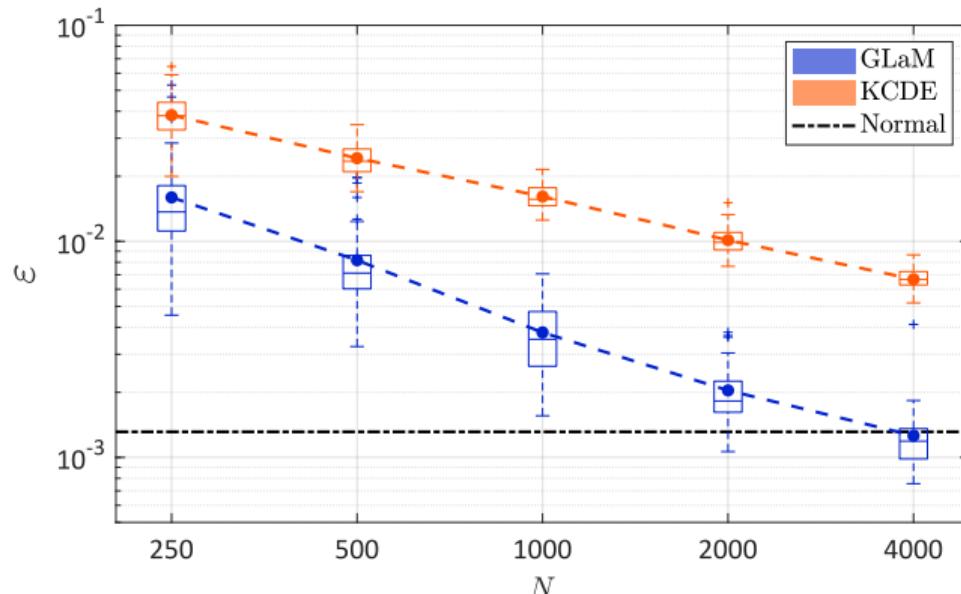
PDF predictions (ED with $N = 500$)



Convergence study

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

$$\varepsilon = \frac{\mathbb{E}_{\mathbf{X}} [d_{\text{WS}}^2(Y(\mathbf{X}), Y^{\text{GLaM}}(\mathbf{X}))]}{\text{Var}[Y]}$$



Outline

Introduction and literature

Generalized lambda distributions

Generalized lambda/PCE models

Application examples

Analytical example

Stochastic SIR model

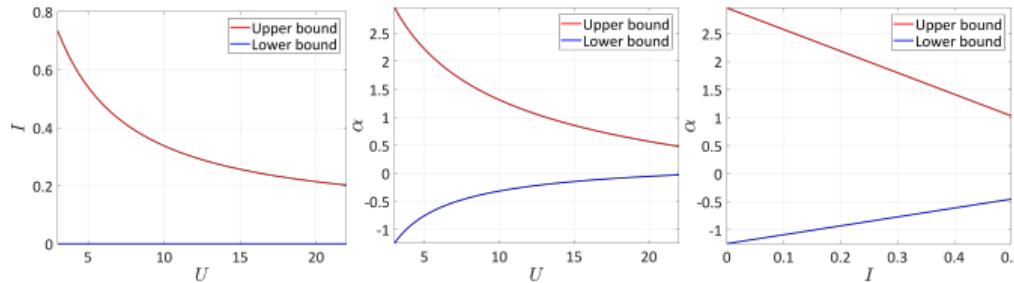
Wind turbine application

Conclusions & outlook

Wind turbine application

- **Five input variables:**

- Mean speed U , turbulence intensity I and shear exponent α are uniformly distributed in the following domain

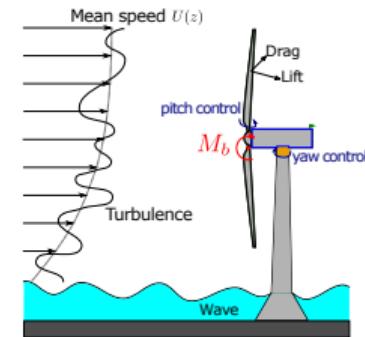


- Air density $\rho \sim \mathcal{U}(0.8, 1.4)$, inclination angle $\beta \sim \mathcal{U}(-10, 10)$

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

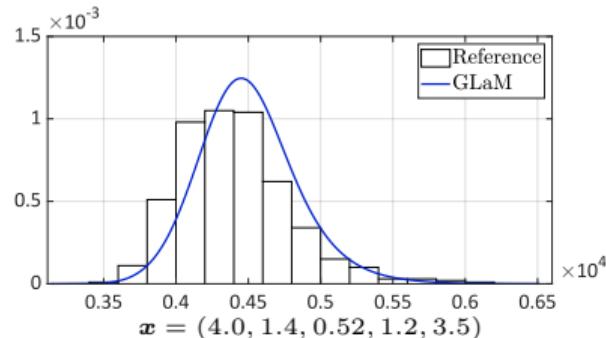
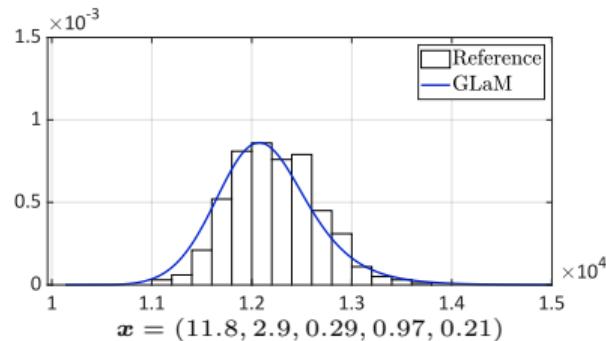
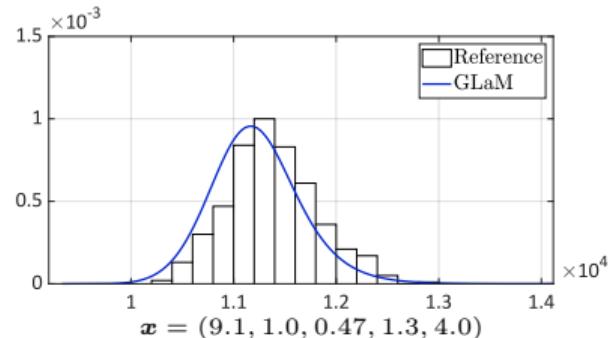
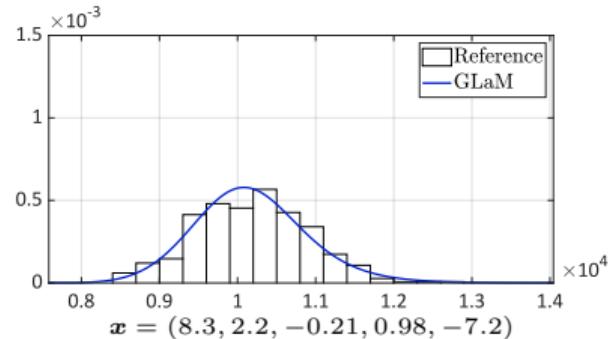
Simulation

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



PDF predictions

The normalized Wasserstein distance is $\varepsilon = 0.013$



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:

$$S_{\mathbf{u}} = \frac{\text{Var} [\mathbb{E} [Y | \mathbf{X}_{\mathbf{u}}]]}{\text{Var} [Y]} \quad \mathbf{u} = \{i_1, \dots, i_k\} \subset \{1, \dots, M\}$$

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

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

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

$$S_{\mathbf{u}}^{\text{traj}}(\mathbf{Z}) = \frac{\text{Var} [Y | \mathbf{X}_{\mathbf{u}}, \mathbf{Z}]}{\text{Var} [Y | \mathbf{Z}]}$$

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

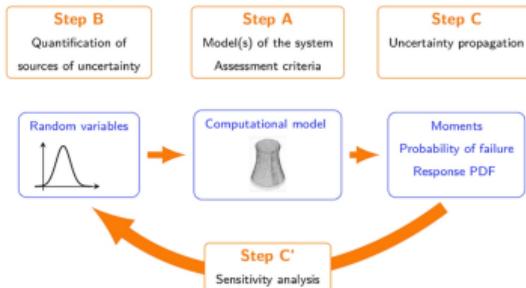
UQLab

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/>

The screenshot shows the UQWorld homepage with a background graphic of a bridge under construction. The top navigation bar includes links for "All About UQ", "UQ Resources", "UQ with UQLab", "Sign Up", "Log In", a search icon, and a menu icon.

Welcome to UQWorld!

Connect with fellow uncertainty quantification (UQ) practitioners across scientific disciplines to discuss the practice of UQ in science and engineering, use cases, and best practices. You can share and discuss your problem, experience, and expertise in all topics related to UQ and UQLab.

All About UQ

Discuss and learn more about UQ important concepts, best practices, and current topics with the community.

UQ Resources

News, updates, and other resources from the UQ community.

UQ with UQLab

Community-powered resources you need to use UQLab for UQ.

Navigation buttons: all categories, all tags, Categories (highlighted), Latest, Top.

Category: All About UQ (Icon: people, speech bubbles, chart). Topics: 24. Description: Connect with members of the community across scientific disciplines to discuss current topics, best practices, important concepts in uncertainty quantification (UQ). Learn more about UQ good practices from the RSUQ Chair. Buttons: Char's Blog, UQ Discussion Forum.

Category: UQ Resources (Icon: globe). Topics: 1 / month. Description: Here you can find news, updates, case studies, and other resources from our own community and the uncertainty quantification (UQ) community at large.

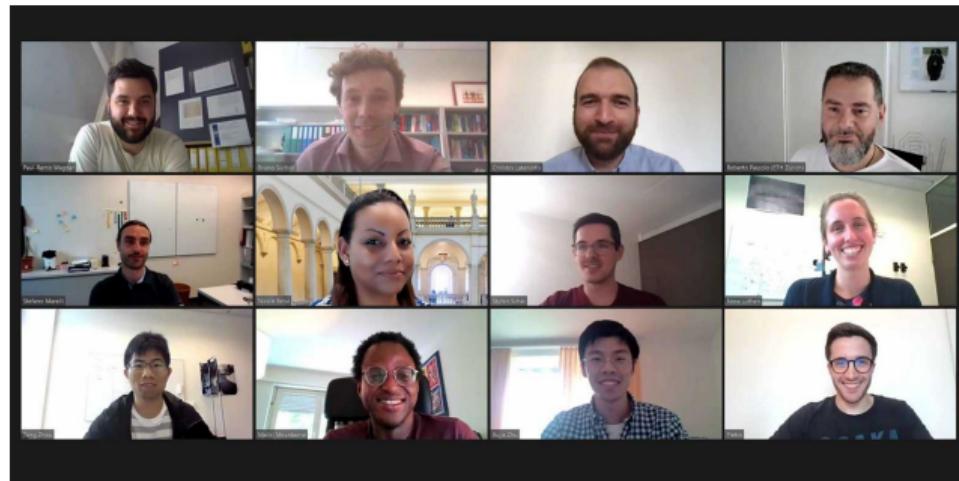
Surrogate models for stochastic simulators

UNCECOMP'2021 – June 30, 2021

B. Sudret

42 / 42

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

