Surrogate-based adaptive techniques for reliability analysis
–
Challenges, benefits and limitations

Jean-Marc BOURINET
SIGMA Clermont / LIMOS

OpenTURNS Users' Day – June 23, 2023 – EDF Lab Saclay

Outline

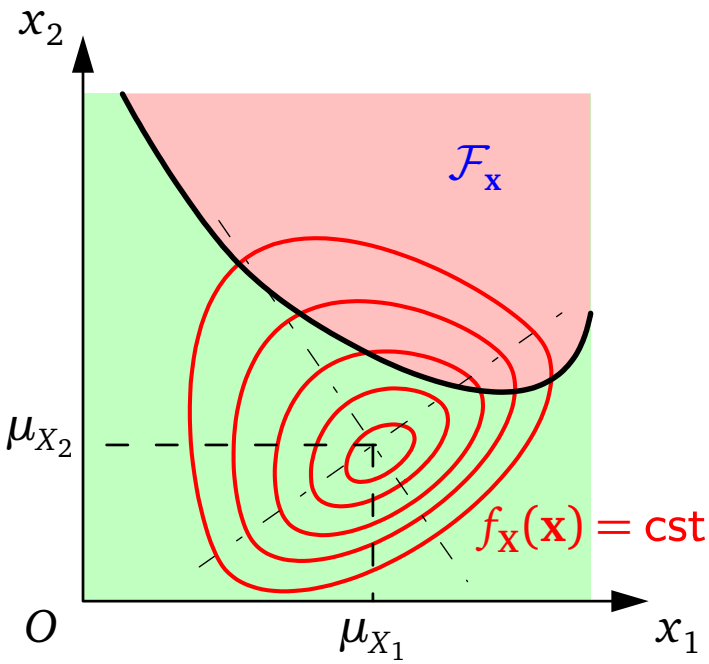
Reliability assessment (RA)

Compute failure probability:

$$p_{\mathrm{f}} = \mathbb{P}\left(g\left(\mathbf{X}\right) \leq 0\right) = \int_{\mathcal{X}} \mathbb{1}_{\mathcal{F}_{\mathbf{X}}}\left(\mathbf{x}\right) f_{\mathbf{X}}(\mathbf{x})\,\mathrm{d}\mathbf{x}$$

where

- ✓ $f_{\mathbf{X}} : \mathbb{R}^n \to \mathbb{R}_{\geq 0}$: joint PDF of random vector $\mathbf{X}$ with support $\mathcal{X} \subseteq \mathcal{D}_g$
- ✓ $g : \mathcal{D}_g \subseteq \mathbb{R}^n \to \mathbb{R}$: limit-state function (LSF)
- ✓ $\mathcal{F}_{\mathbf{X}} = \{\mathbf{x} \in \mathcal{D}_g : g(\mathbf{x}) \leq 0\}$: failure domain
- ✓ $\mathbb{1}_{\mathcal{F}_{\mathbf{X}}}$: indicator function of failure domain



Mapping to the standard normal space ($\mathbf{u}$-space)

RA conveniently solved in the standard normal space ($\mathbf{u}$-space) by considering a mapping $\mathbf{u} = T(\mathbf{x})$

Solutions

- ✓ Copula dependence structure (Gaussian, block, Vine)
  Ex.: Gaussian copula ▶ Nataf transformation (see next slide)
- ✓ Joint distribution and conditional CDFs known ▶ Rosenblatt transformation

No mapping available

- ✓ Markov chains, hidden Markov models
- ✓ Distributions difficult to sample from (MCMC-based)

Example: Nataf transformation

Nataf transformation (Liu and Der Kiureghian, 1986) $\equiv$ Gaussian copula parameterized by correlation matrix $\mathbf{R}_0$ (Lebrun and Dutfoy, 2009)

$$\mathbf{x} \in \mathcal{X} \xrightarrow[\substack{v_i = F_{X_i}(x_i) \\ \text{for } i = 1, \ldots, n}]{} \mathbf{v} \in [0,1]^n \xrightarrow[\substack{z_i = \Phi^{-1}(v_i) \\ \text{for } i = 1, \ldots, n}]{} \mathbf{z} \in \mathbb{R}^n \xrightarrow[\mathbf{u} = \mathbf{L}_0^{-1}\mathbf{z}]{} \mathbf{u} \in \mathbb{R}^n$$

where $\mathbf{R}_0 = \left[\rho_{0,ij}\right]_{1 \leq i,j \leq n} = \mathbf{L}_0 \mathbf{L}_0^{\mathrm{T}} = \left[\mathrm{Cov}\left[Z_i, Z_j\right]\right]_{1 \leq i,j \leq n}$ satisfies:

$$\rho_{ij} = \int_{\mathbb{R}} \int_{\mathbb{R}} \left(\frac{F_{X_i}^{-1}\left(\Phi\left(z_i\right)\right) - \mu_i}{\sigma_i} \right) \left(\frac{F_{X_j}^{-1}\left(\Phi\left(z_j\right)\right) - \mu_j}{\sigma_j} \right) \varphi_2\left(z_i, z_j; \rho_{0,ij}\right) \mathrm{d}z_i \mathrm{d}z_j$$

Example: Nataf transformation

Nataf transformation (Liu and Der Kiureghian, 1986) $\equiv$ Gaussian copula parameterized by correlation matrix $\mathbf{R}_0$ (Lebrun and Dutfoy, 2009)

$$\mathbf{x} \in \mathcal{X} \xrightarrow[\substack{v_i = F_{X_i}(x_i) \\ \text{for } i = 1, \ldots, n}]{} \mathbf{v} \in [0,1]^n \xrightarrow[\substack{z_i = \Phi^{-1}(v_i) \\ \text{for } i = 1, \ldots, n}]{} \mathbf{z} \in \mathbb{R}^n \xrightarrow[\mathbf{u} = \mathbf{L}_0^{-1}\mathbf{z}]{} \mathbf{u} \in \mathbb{R}^n$$

where $\mathbf{R}_0 = \left[\rho_{0,ij}\right]_{1 \leq i,j \leq n} = \mathbf{L}_0 \mathbf{L}_0^{\mathrm{T}} = \left[\mathrm{Cov}\left[Z_i, Z_j\right]\right]_{1 \leq i,j \leq n}$ satisfies:

$$\rho_{ij} = \int_{\mathbb{R}} \int_{\mathbb{R}} \left(\frac{F_{X_i}^{-1}\left(\Phi\left(z_i\right)\right) - \mu_i}{\sigma_i} \right) \left(\frac{F_{X_j}^{-1}\left(\Phi\left(z_j\right)\right) - \mu_j}{\sigma_j} \right) \varphi_2\left(z_i, z_j; \rho_{0,ij}\right) \mathrm{d}z_i \mathrm{d}z_j$$

Remarks

- This transformation only requires the definition of the marginals of $\mathbf{X}$ and the linear correlations ρ_{ij} between its components
- Unknown $\rho_{0,ij}$ inside the integral ▶ Solved by numerical integration

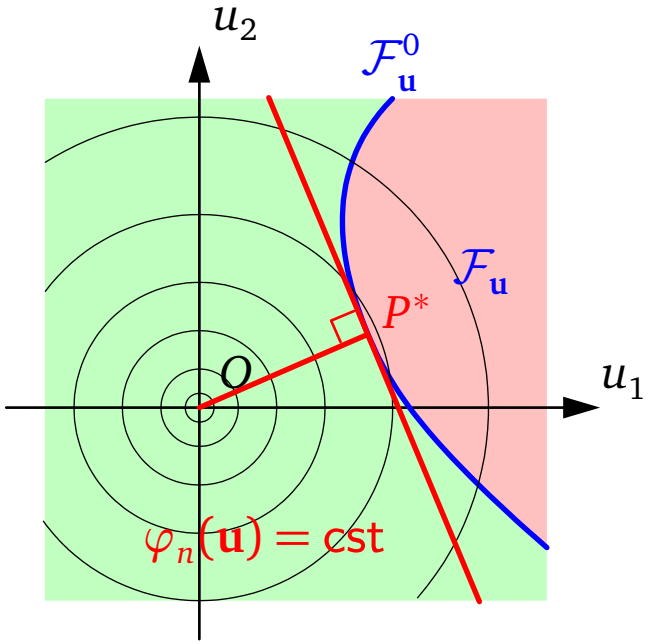
RA in the standard normal space ($\mathbf{u}$-space)

Compute failure probability:

$$p_{\mathrm{f}} = \mathbb{P}\left(G\left(\mathbf{U}\right) \leq 0\right) = \int_{\mathbb{R}^n} \mathbb{1}_{\mathcal{F}_{\mathbf{u}}}\left(\mathbf{u}\right) \varphi_n(\mathbf{u}) \, \mathrm{d}\mathbf{u}$$

where

- ✓ φ_n : n-dimensional standard normal PDF
- ✓ $G : \mathbb{R}^n \to \mathbb{R}, \mathbf{u} \mapsto G(\mathbf{u}) = (g \circ T^{-1})(\mathbf{u})$: LSF mapped to $\mathbf{u}$-space
- ✓ $\mathcal{F}_{\mathbf{u}} = \{\mathbf{u} \in \mathbb{R}^n : G(\mathbf{u}) \leq 0\}$: failure domain
- ✓ $\mathcal{F}^0_{\mathbf{u}} = \{\mathbf{u} \in \mathbb{R}^n : G(\mathbf{u}) = 0\}$: limit-state surface (LSS)
- ✓ P^*: most probable failure point (MPFP) computed by FORM



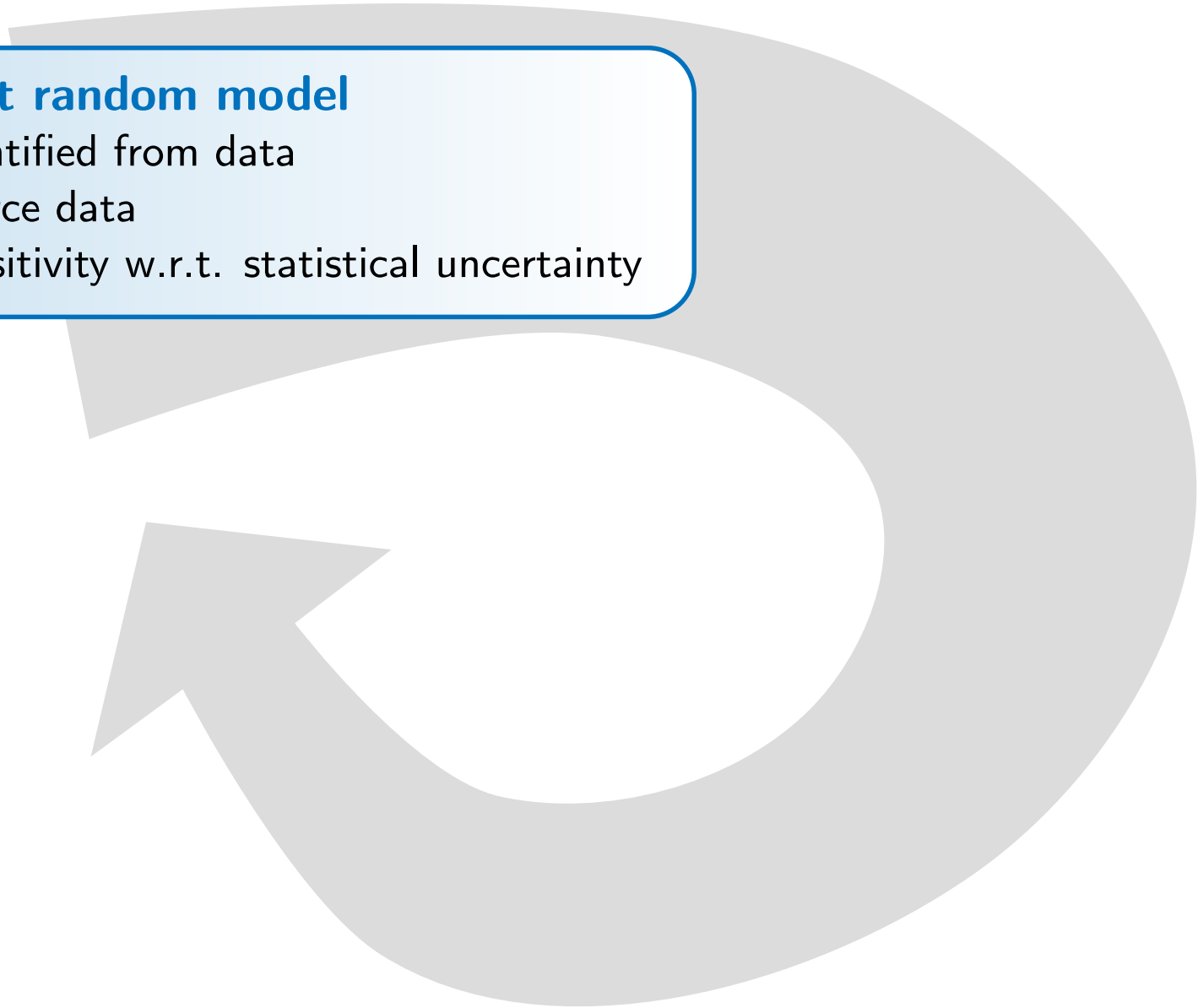
Important remarks

$$p_{\mathrm{f}} = \mathbb{P}\left(G\left(\mathbf{U}\right) \leq 0\right) = \int_{\mathbb{R}^n} \mathbb{1}_{\mathcal{F}_{\mathbf{u}}}\left(\mathbf{u}\right) \varphi_n(\mathbf{u}) \, \mathrm{d}\mathbf{u}$$

Important remarks

- Failure event assumed rare ($p_{\mathrm{f}} < 10^{-3}$) (systems with high safety level)
- Limit-state function G only known pointwise and expensive-to-evaluate

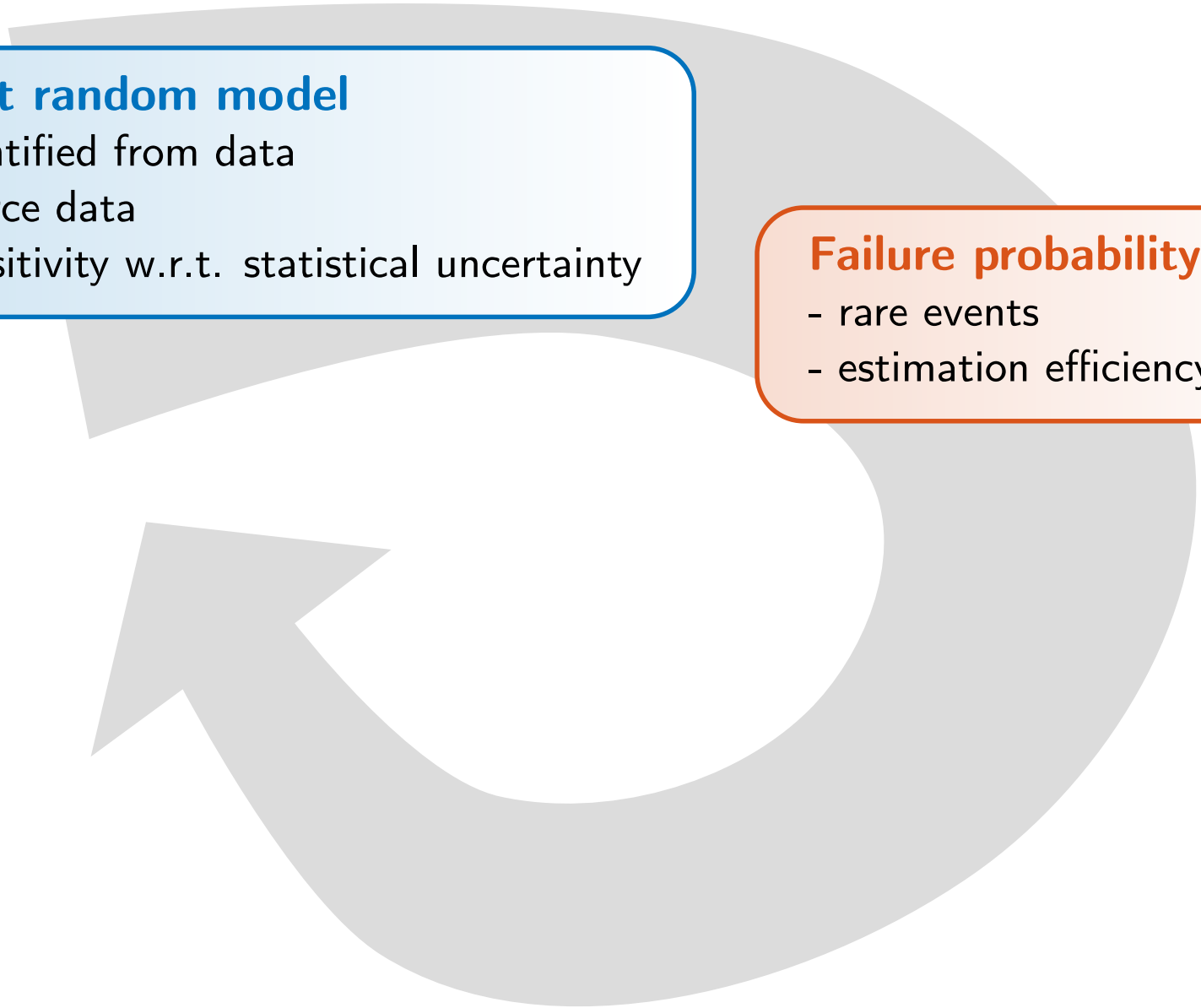
Main challenges in RA



Input random model

- identified from data
- scarce data
- sensitivity w.r.t. statistical uncertainty

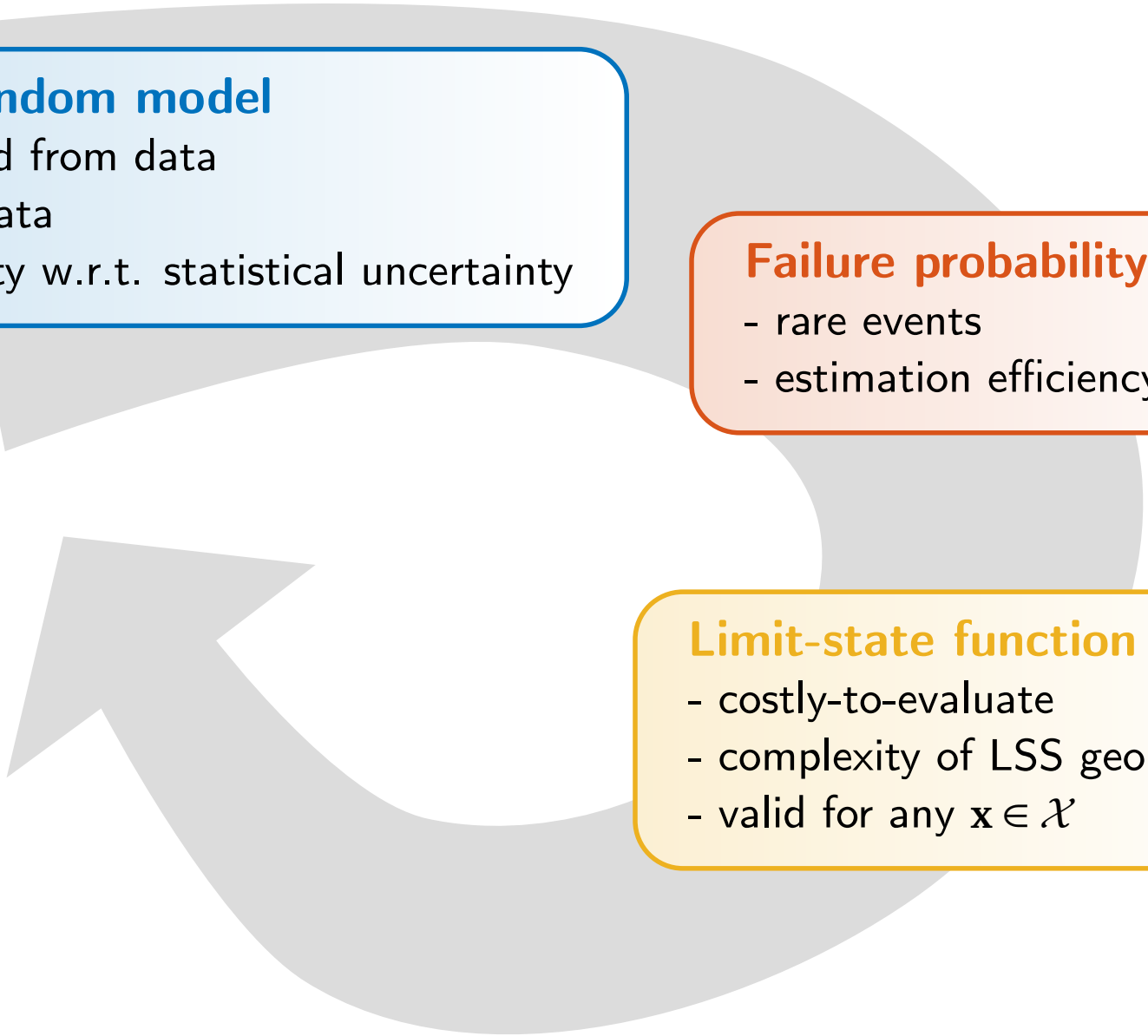
Main challenges in RA



Input random model
- identified from data
- scarce data
- sensitivity w.r.t. statistical uncertainty

Failure probability level
- rare events
- estimation efficiency

Main challenges in RA



Input random model
- identified from data
- scarce data
- sensitivity w.r.t. statistical uncertainty

Failure probability level
- rare events
- estimation efficiency

Limit-state function
- costly-to-evaluate
- complexity of LSS geometry
- valid for any $\mathbf{x} \in \mathcal{X}$

Main challenges in RA

Input random model
- identified from data
- scarce data
- sensitivity w.r.t. statistical uncertainty

Failure probability level
- rare events
- estimation efficiency

Limit-state function
- costly-to-evaluate
- complexity of LSS geometry
- valid for any $\mathbf{x} \in \mathcal{X}$

Problem dimensionality
- number of random inputs in $\mathbf{X}$
- curse of dimensionality

Problem to solve

$$\mathbf{u}^* = \arg\min_{\mathbf{u}\in\mathbb{R}^n} \frac{1}{2}\mathbf{u}^{\mathrm{T}}\mathbf{u} \quad \text{s.t.} \quad G(\mathbf{u}) = 0$$

where

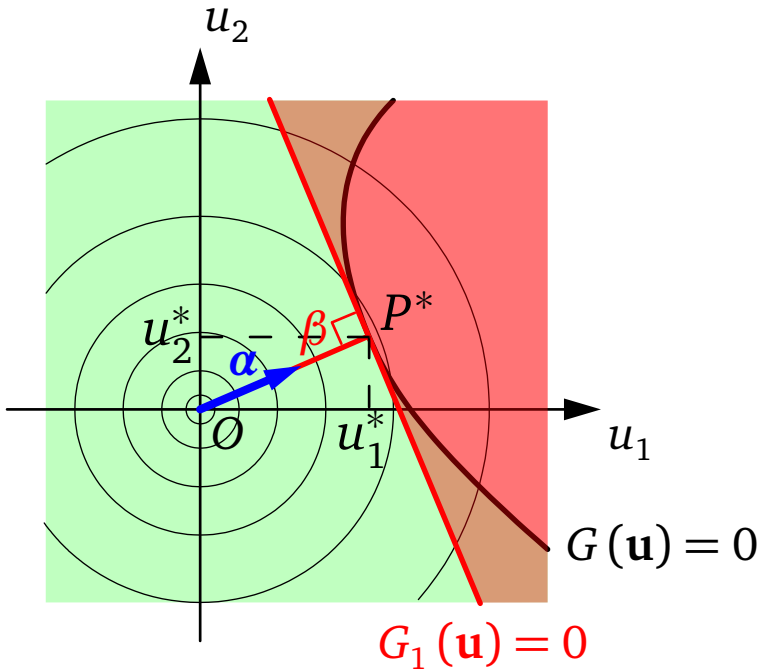
- ✓ $\mathbf{u}^*$: coordinates of most probable failure point P^* (MPFP)
- ✓ $\boldsymbol{\alpha} = -\nabla G(\mathbf{u}^*)/\|\nabla G(\mathbf{u}^*)\|$: unit α-vector
- ✓ $\beta_{\mathrm{HL}} = \beta = \boldsymbol{\alpha}^{\mathrm{T}}\mathbf{u}^*$: (Hasofer-Lind) reliability index

FORM approximation

First-order Taylor polynomial of G at $\mathbf{u}^*$

$$G_1(\mathbf{u}) = G(\mathbf{u}^*) + \nabla G(\mathbf{u}^*)^{\mathrm{T}}(\mathbf{u} - \mathbf{u}^*)$$

$$\begin{aligned} p_{\mathrm{f}} &= \mathbb{P}(G(\mathbf{U}) \le 0) \\ &\approx \mathbb{P}(G_1(\mathbf{U}) \le 0) = \Phi(-\beta) \end{aligned}$$

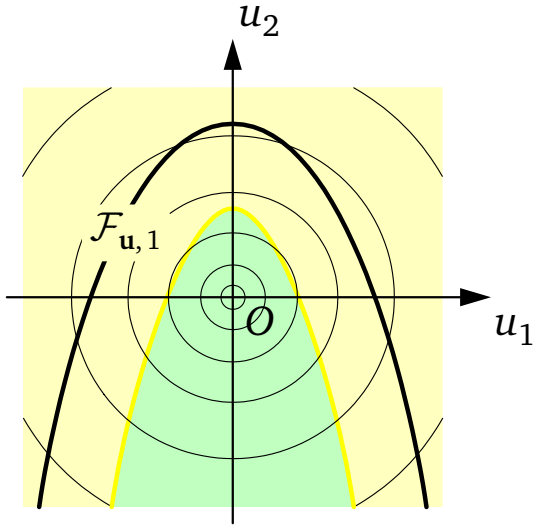


Conceptual idea of subset simulation (SS)

$$p_{\mathrm{f}} = \mathbb{P}(E_m|E_{m-1})\,\mathbb{P}(E_{m-1}|E_{m-2})\ldots\mathbb{P}(E_2|E_1)\,\mathbb{P}(E_1)$$

where $E_m \subset E_{m-1} \subset \cdots \subset E_2 \subset E_1$
are nested intermediate failure events

$$E_s = \{G(\mathbf{U}) \leq y_s\} \quad \text{for } s = 1, \ldots, m$$



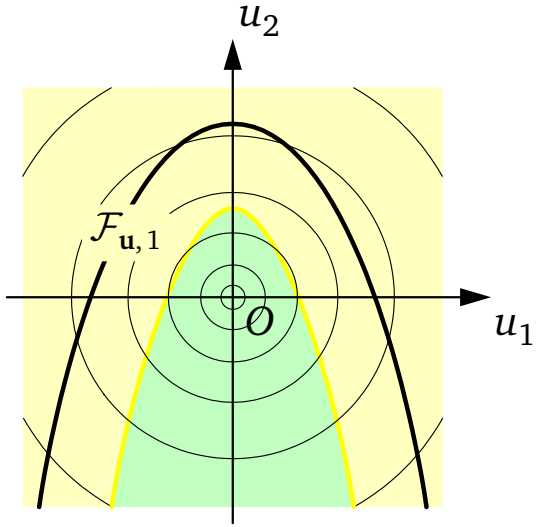
Subset simulation (SS)

Conceptual idea of subset simulation (SS)

$$p_{\mathrm{f}} = \mathbb{P}(E_m|E_{m-1})\,\mathbb{P}(E_{m-1}|E_{m-2})\ldots\mathbb{P}(E_2|E_1)\,\mathbb{P}(E_1)$$

where $E_m \subset E_{m-1} \subset \cdots \subset E_2 \subset E_1$ are nested intermediate failure events

$$E_s = \{G(\mathbf{U}) \leq y_s\} \quad \text{for } s = 1, \ldots, m$$



SS estimation

$$\widehat{p}_{\mathrm{f}}^{\mathrm{SS}} = \prod_{s=1}^{m} \widehat{p}_s \quad \text{where} \quad \widehat{p}_s = \frac{1}{N}\sum_{j=1}^{N} \mathbb{1}_{\mathcal{F}_{\mathbf{u},s}}\left(\mathbf{U}_s^{(j)}\right)$$

where $\mathcal{F}_{\mathbf{u},s} = \{\mathbf{u} \in \mathbb{R}^n : G(\mathbf{u}) \leq y_s\}$ and

$$\begin{vmatrix} \mathbf{U}_1^{(1)}, \ldots, \mathbf{U}_1^{(N)} \overset{\text{i.i.d.}}{\sim} \varphi_n \\ \mathbf{U}_s^{(1)}, \ldots, \mathbf{U}_s^{(N)} \overset{\text{i.i.d.}}{\sim} \varphi_n(\cdot|E_{s-1}) \text{ for } s > 1 \end{vmatrix}$$

Remark

- Samples $\sim \varphi_n(\cdot|E_{s-1})$ generated by MCMC with the modified Metropolis algorithm of Au and Beck (2001)

SS-based reliability sensitivities in standard normal space

- Consider

$$\varphi_n(\mathbf{u}) = \varphi_n(\mathbf{u};\boldsymbol{\theta}) \quad \text{where } \boldsymbol{\theta} = (\boldsymbol{\mu}, \boldsymbol{\sigma}) \text{ and } \left| \begin{array}{l} \boldsymbol{\mu} = (0, \ldots, 0) \\ \boldsymbol{\sigma} = (1, \ldots, 1) \end{array} \right.$$

SS-based reliability sensitivities in standard normal space

- Consider

$$\varphi_n(\mathbf{u}) = \varphi_n(\mathbf{u};\boldsymbol{\theta}) \quad \text{where } \boldsymbol{\theta} = (\boldsymbol{\mu}, \boldsymbol{\sigma}) \text{ and } \begin{vmatrix} \boldsymbol{\mu} = (0, \ldots, 0) \\ \boldsymbol{\sigma} = (1, \ldots, 1) \end{vmatrix}$$

- For $s = 1, \ldots, m$, define

$$\pi_s = \prod_{k=1}^{s} p_k$$

where $\begin{vmatrix} p_1 = \mathbb{P}(E_1) \\ p_k = \mathbb{P}(E_k | E_{k-1}) \text{ for } k > 1 \end{vmatrix}$

and $E_m \subset E_{m-1} \subset \cdots \subset E_2 \subset E_1$ are nested intermediate failure events

$$E_k = \{G(\mathbf{U}) \leq y_k\} \quad \text{for } k = 1, \ldots, m$$

SS-based reliability sensitivities in standard normal space

- Consider

$$\varphi_n(\mathbf{u}) = \varphi_n(\mathbf{u};\boldsymbol{\theta}) \quad \text{where } \boldsymbol{\theta} = (\boldsymbol{\mu},\boldsymbol{\sigma}) \text{ and } \left| \begin{array}{l} \boldsymbol{\mu} = (0,\ldots,0) \\ \boldsymbol{\sigma} = (1,\ldots,1) \end{array} \right.$$

- For $s = 1,\ldots,m$, define

$$\pi_s = \prod_{k=1}^{s} p_k$$

where $\left| \begin{array}{l} p_1 = \mathbb{P}(E_1) \\ p_k = \mathbb{P}(E_k|E_{k-1}) \text{ for } k > 1 \end{array} \right.$

and $E_m \subset E_{m-1} \subset \cdots \subset E_2 \subset E_1$ are nested intermediate failure events

$$E_k = \{G(\mathbf{U}) \leq y_k\} \quad \text{for } k = 1,\ldots,m$$

- Normalized SS-based sensitivities of π_s w.r.t. $\theta \in \boldsymbol{\theta}$ (Bourinet, 2018)

$$\frac{1}{\pi_s}\frac{\partial \pi_s}{\partial \theta} = \sum_{k=1}^{s} \frac{1}{p_k}\frac{\partial p_k}{\partial \theta} \quad \text{for } s = 1,\ldots,m$$

Common practice to publishing

- **Usual structure of a scientific paper**
  - ✓ Review (list) of related works
  - ✓ Proposition of a new method w.r.t. those available in literature
  - ✓ Application to a couple of selected examples (couple = 2 to 4)
  - ✓ Conclusion: "My method outperforms the others" (most often)

Common practice to publishing

Usual structure of a scientific paper

- ✓ Review (list) of related works
- ✓ Proposition of a new method w.r.t. those available in literature
- ✓ Application to a couple of selected examples (couple = 2 to 4)
- ✓ Conclusion: "My method outperforms the others" (most often)

Biased process

- ✓ The proposed method outperforms the others for the selected examples
- ✓ Selected examples often too easy to solve (e.g., in reliability assessment: low dimensional input space, almost linear limit-state surface in standard normal space, not so low failure probability, ...)
- ✓ Results to be considered with cautious (failure probability point estimate, a uniquely defined initial DOE, ...)

Remedies?

- **Benchmarking, challenges**
  - ✓ Definition of challenging examples + performance metrics
  - ✓ Quite common in other fieds than reliability assessment
    - Continuous optimization: black-box optimization benchmarking (BBOB)
    - Data science, machine learning: Kaggle competitions
  - ✓ In reliability assessment (RA):
    - Benchmark study on reliability estimation in high dimensions of structural systems – G.I. Schuëller, H.J. Pradlwarter (ICOSSAR 2005 + Structural Safety 2007)
    - Black-box reliability challenge 2019 – Á. Rózsás, A. Slobbe

Remedies?

Benchmarking, challenges

- ✓ Definition of challenging examples + performance metrics
- ✓ Quite common in other fieds than reliability assessment
  - Continuous optimization: black-box optimization benchmarking (BBOB)
  - Data science, machine learning: Kaggle competitions
- ✓ In reliability assessment (RA):
  - Benchmark study on reliability estimation in high dimensions of structural systems – G.I. Schuëller, H.J. Pradlwarter (ICOSSAR 2005 + Structural Safety 2007)
  - Black-box reliability challenge 2019 – Á. Rózsás, A. Slobbe

Objective (Bourinet, 2019a)

- ✓ Identify challenging application examples yet hopefully solvable by surrogate models
- ✓ Dos and don'ts for example selection

Proposed benchmark examples

Challenging RA examples

8 examples with documentation and Matlab/FERUM 4.1 source files
Available at http://www.gdr-mascotnum.fr/benchmarks.html

Step 1: Install FERUM 4.1

FERUM 4.1 available at https://www.sigma-clermont.fr/en/ferum
Simply unzip the archive!

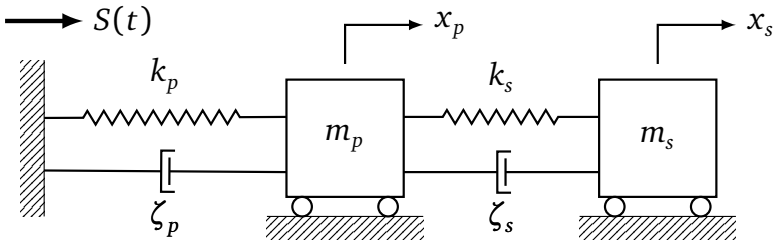
Step 2: FERUM-based LSF evaluations

Evaluate LSF at $\mathbf{U} \sim \mathcal{N}_n(0,1)$ with the following lines

```
addpath('/xxx/xxx/FERUM4.1');

inputfile_ex1 % read input file of the selected example

[probdata,gfundata,analysisopt] = update_data(1,probdata,analysisopt,gfundata,femodel);
probdata.marg = distribution_parameter(probdata.marg);
[ Ro, dRo_drho, dRo_dthetafi, dRo_dthetafj ] = mod_corr_solve(probdata.marg,probdata.correlation,0);
probdata.Ro = Ro;
[ Lo , ierr ] = my_chol(Ro);
probdata.Lo = Lo;
iLo = inv(Lo);
probdata.iLo = iLo;
nrv = size(probdata.marg,1);

nU = 10;
U = randn(nrv,nU); % define realizations in the standard normal space
X = u_to_x(U,probdata); % map U back to the x-space
[ G , dummy ] = gfun(1,X,'no ',probdata,analysisopt,gfundata,femodel,randomfield); % evaluate the LSF
```

Application example #3

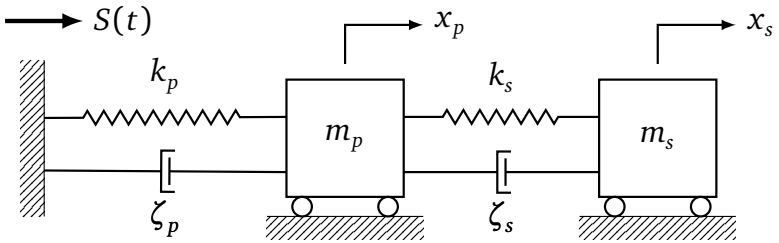
- **Problem description** (adapted from De Stefano and Der Kiureghian,1990)
  - ✓ 2-DOF primary-secondary system under white noise base acceleration
  - ✓ 8 independent and lognormally distributed random inputs
  - ✓ LSF:
    $g(\mathbf{x}) = F_s - F\left(m_p, m_s, k_p, k_s, \zeta_p, \zeta_s, S_0\right)$
  - ✓ Failure probability: $p_{\text{fref}} = \begin{cases} 4.78 \times 10^{-3} & \text{for } \mu_{F_s} = 15 \\ 3.78 \times 10^{-7} & \text{for } \mu_{F_s} = 27.5 \end{cases}$



Application example #3

Problem description (adapted from De Stefano and Der Kiureghian,1990)

- ✓ 2-DOF primary-secondary system under white noise base acceleration
- ✓ 8 independent and lognormally distributed random inputs
- ✓ LSF:
  $g(\mathbf{x}) = F_s - F\left(m_p, m_s, k_p, k_s, \zeta_p, \zeta_s, S_0\right)$



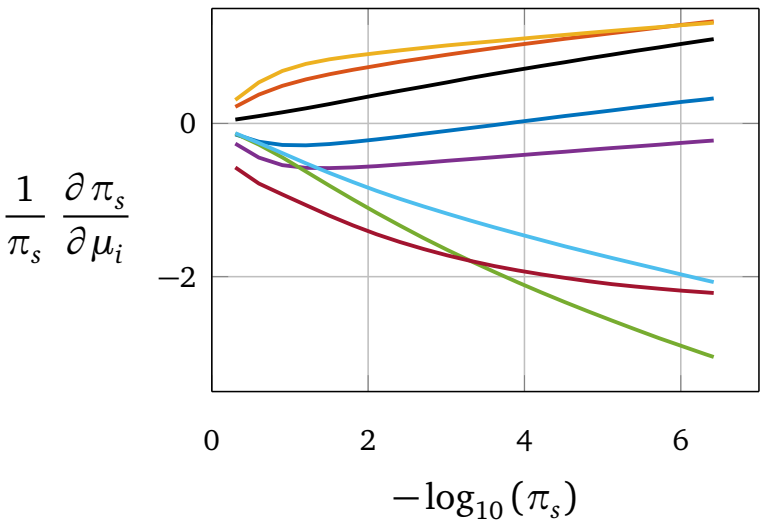
- ✓ Failure probability: $p_{\text{fref}} = \begin{cases} 4.78 \times 10^{-3} & \text{for } \mu_{F_s} = 15 \\ 3.78 \times 10^{-7} & \text{for } \mu_{F_s} = 27.5 \end{cases}$

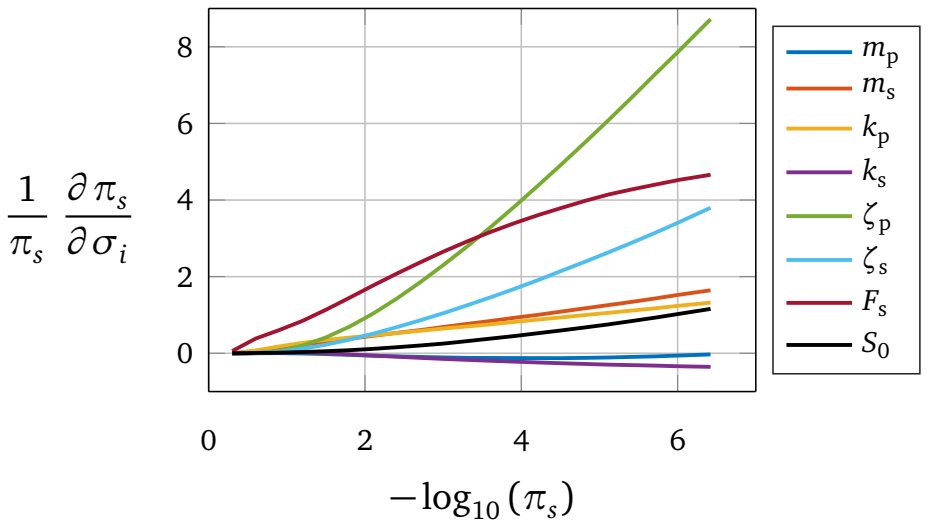
Remarks

- ✓ Single MPFP but very hard to find with FORM!
- ✓ Highly curved limit-state surface at MPFP
- ✓ Sensitivities w.r.t. random inputs different at mean and MPFP

Application example #3

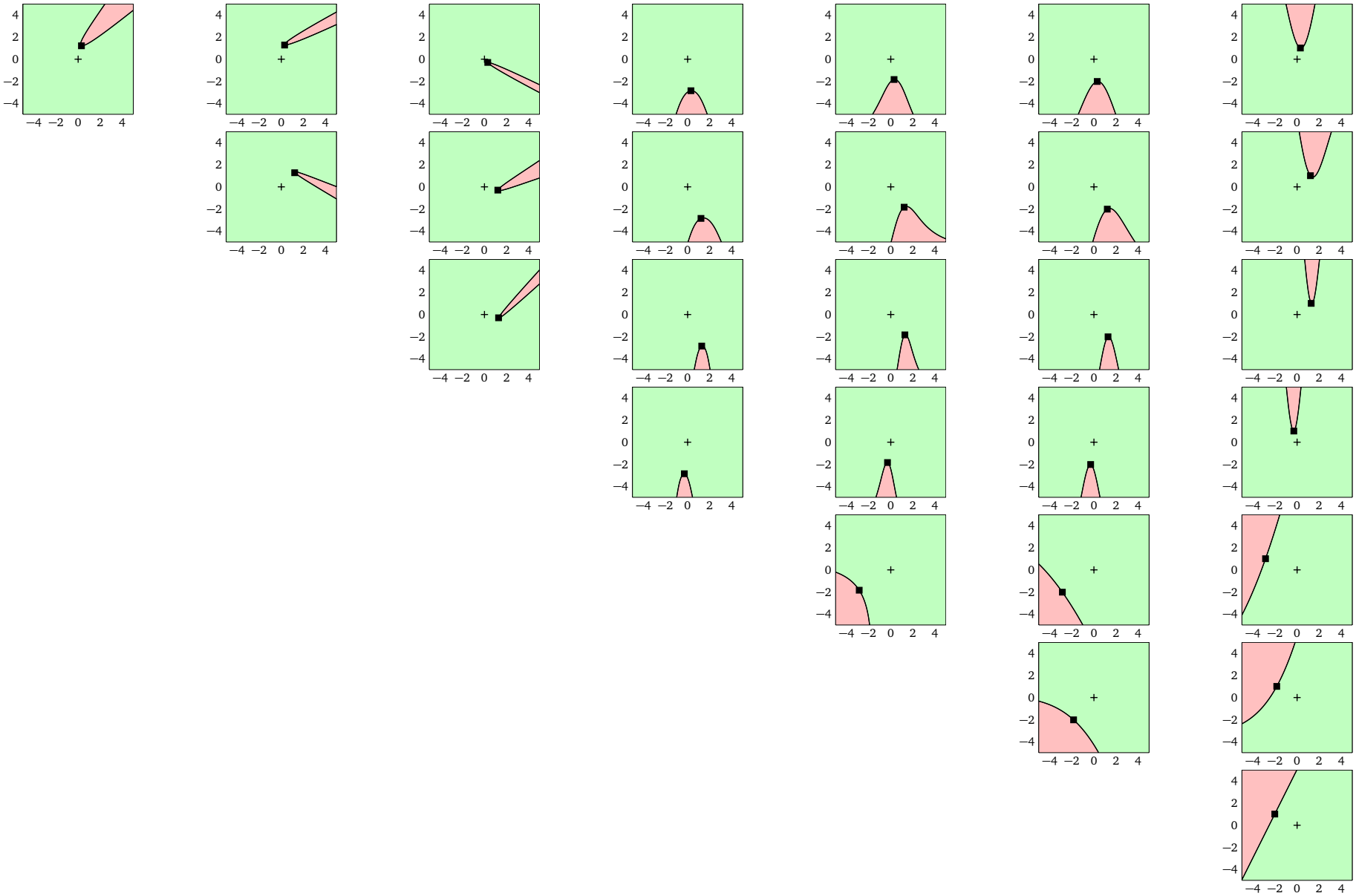
SS-based sensitivities in standard normal space





Application example #3

(P^*, u_i, u_j) cross-cuts at MPFP in standard normal space



Application example #3

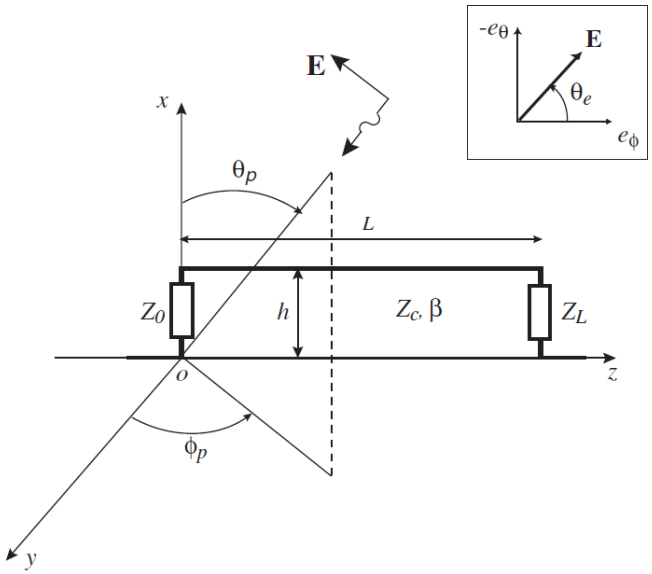
Method	$\mu_{F_s} = 15$ p_f estimate (#LSF calls)	$\mu_{F_s} = 27.5$ p_f estimate (#LSF calls)
Subset simulation (reference)	$\mathbf{4.78 \times 10^{-3}}$	$\mathbf{3.78 \times 10^{-7}}$
EGRA (Bichon et al., 2008)	5.10×10^{-3} (1694) $^{(a)}$	-
^{2}SMART (Bourinet et al., 2011)	4.78×10^{-3} (1719) $^{(b)}$	3.66×10^{-7} (4011) $^{(b)}$
AK-MCS (Echard et al., 2011)	4.83×10^{-3} (1106) $^{(a)}$	-
Meta-IS (Dubourg et al., 2013)	4.80×10^{-3} (664)	3.76×10^{-7} (680)
GSAS (Hu and Mahadevan, 2016)	4.81×10^{-3} (867)	-
DA-SED based G-ANOVA (Chakraborty and Chowdhury, 2016)	4.8×10^{-3} (248)	-
GTM (Sundar and Shields, 2016)	4.76×10^{-3} (610-650)	-
ASVR-iso-Gaussian (Bourinet, 2016)	-	3.81×10^{-7} (648) $^{(b)}$
iASVR-iso-Gaussian (Bourinet, 2017)	4.80×10^{-3} (310)	3.72×10^{-7} (540)
iASVR-ani-Gaussian (Bourinet, 2017)	4.73×10^{-3} (360)	3.73×10^{-7} (690)

$^{(a)}$ EGRA and AK-MCS results from Hu and Mahadevan (2016).

$^{(b)}$ ^{2}SMART and ASVR results are averages over independent runs (resp. 50 and 20 runs). Other results in table are point estimates.

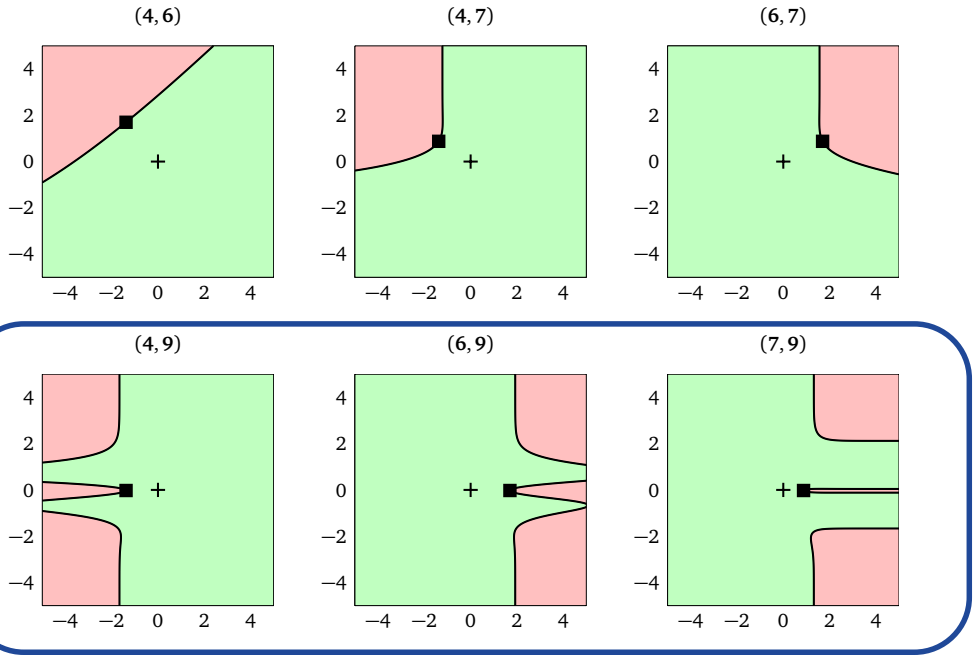
Application example #4

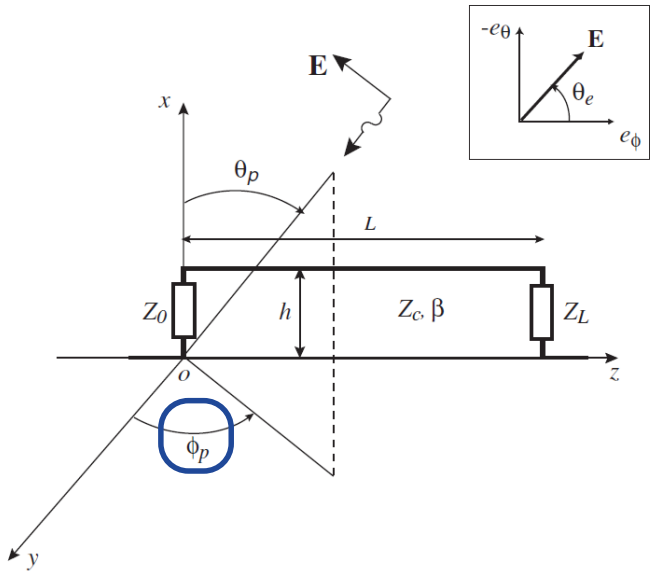
- **Problem description** (Kouassi et al., 2016)
  - ✓ 11 independent random inputs: 6 lognormal and 5 uniform
  - ✓ LSF: $g(\mathbf{x}) = 1.5 \times 10^{-4}\mathrm{A} - I_{Z_L}(\mathbf{x})$
  - ✓ Failure probability: $p_{\mathrm{fref}} = 2.24 \times 10^{-4}$ (MC with 10^9 samples)



Application example #4

- **Problem description** (Kouassi et al., 2016)
  - ✓ 11 independent random inputs: 6 lognormal and 5 uniform
  - ✓ LSF: $g(\mathbf{x}) = 1.5 \times 10^{-4}\mathrm{A} - I_{Z_L}(\mathbf{x})$
  - ✓ Failure probability: $p_{\mathrm{fref}} = 2.24 \times 10^{-4}$ (MC with 10^9 samples)





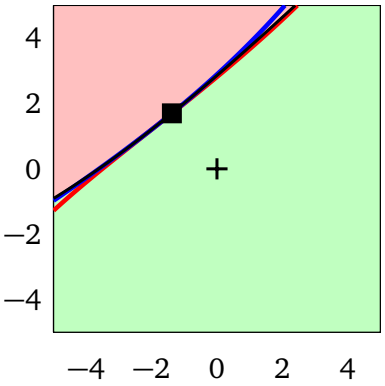
Representative cross-cuts (P^*, u_i, u_j) at MPFP P^* in standard normal space

▶ Very sharp peaks of LSS for each cross-cut involving the azimut angle $X_9 = \phi_p$

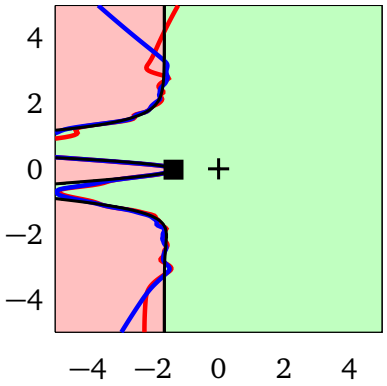
Application example #4

Method	p_f estimate (#LSF calls)
MC (reference)	$\mathbf{2.24 \times 10^{-4}}$ **(10^9)**
FORM	3.18×10^{-3} (9,144)
SORM-cf	1.10×10^{-4} (77)(+9,144)
iASVR + ani-Matérn	2.16×10^{-4} (730)

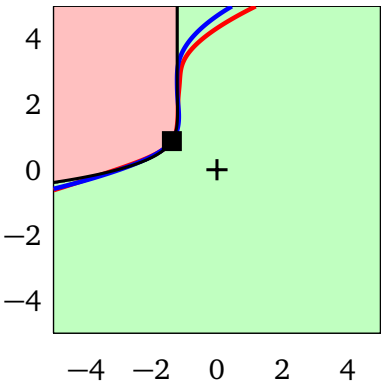
(Z_L, a_e)



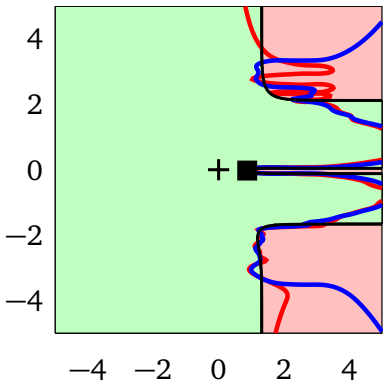
(Z_L, ϕ_p)



(Z_L, θ_e)

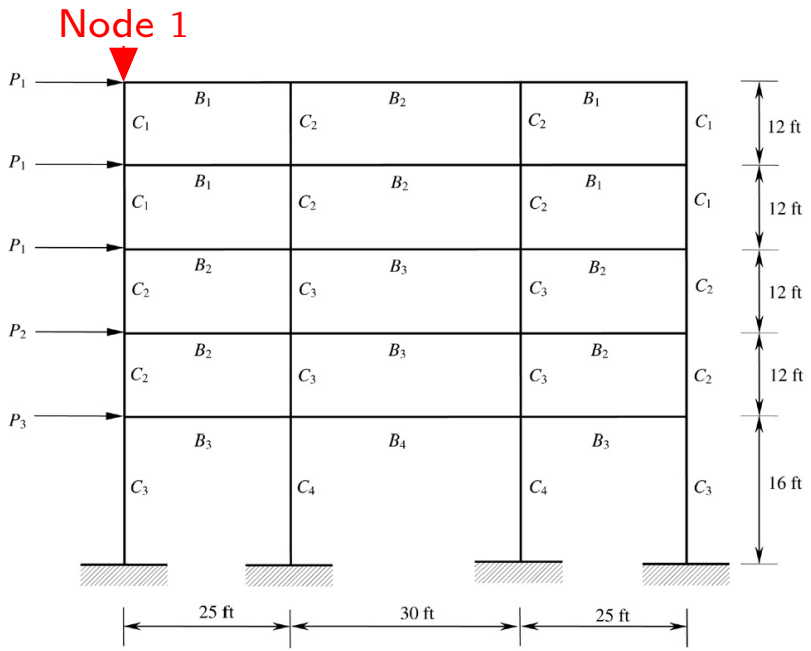


(θ_e, ϕ_p)



Application example #5

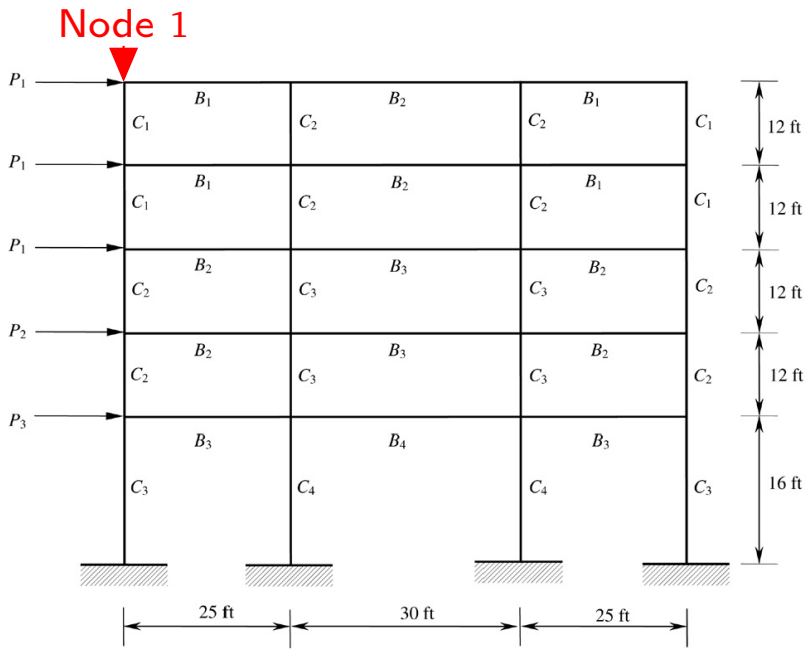
- **Problem description** (Blatman and Sudret, 2010)
  - ✓ 21 random and partly correlated inputs: 3 applied loads, 2 Young's moduli, 8 moments of inertia and 8 cross-sectional areas
  - ✓ LSF: $g(\mathbf{x}) = 7\text{cm} - u_1(\mathbf{x})$
  - ✓ Normal distributions left-truncated at zero for Young's moduli, moments of inertia and cross-sectional areas
  - ✓ Failure probability: $p_{\text{fref}} = 1.05 \times 10^{-4}$ (SS with $500,000$ samples per level)



Application example #5

Problem description (Blatman and Sudret, 2010)

- ✓ 21 random and partly correlated inputs: 3 applied loads, 2 Young's moduli, 8 moments of inertia and 8 cross-sectional areas
- ✓ LSF: $g(\mathbf{x}) = 7\text{cm} - u_1(\mathbf{x})$
- ✓ Normal distributions left-truncated at zero for Young's moduli, moments of inertia and cross-sectional areas
- ✓ Failure probability: $p_{\text{fref}} = 1.05 \times 10^{-4}$ (SS with $500,000$ samples per level)



Remark

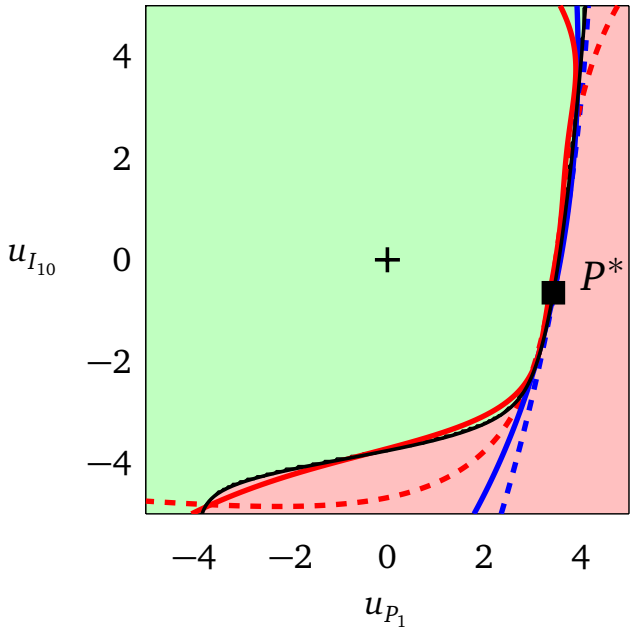
- ✓ Intricate geometry at MPFP due to left-truncated correlated random variables

Application example #5

Method	p_f estimate (#LSF calls)
Subset simulation (reference)	$\mathbf{1.05 \times 10^{-4}}$ **(**$\mathbf{2 \times 10^{6}}$**)**
FORM	1.40×10^{-5} (256)
SORM-cf	2.83×10^{-5} (252)(+256)
Sparse-PCE (Blatman and Sudret, 2010)	1.98×10^{-5} (450)
iASVR-iso-Gaussian (Bourinet, 2017)	2.89×10^{-5} (380)
iASVR-ani-Gaussian (Bourinet, 2017)	1.09×10^{-4} (575)

Application example #5

Method	p_f estimate (#LSF calls)
Subset simulation (reference)	$\mathbf{1.05 \times 10^{-4}}$ **(** $\mathbf{2 \times 10^6}$ **)**
FORM	1.40×10^{-5} (256)
SORM-cf	2.83×10^{-5} (252)(+256)
Sparse-PCE (Blatman and Sudret, 2010)	1.98×10^{-5} (450)
iASVR-iso-Gaussian (Bourinet, 2017)	2.89×10^{-5} (380)
iASVR-ani-Gaussian (Bourinet, 2017)	1.09×10^{-4} (575)

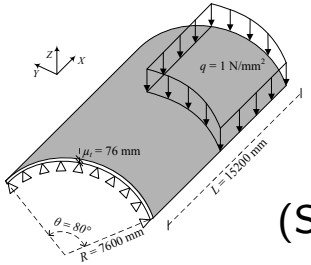


Cross-cut $(P^*, u_{P_1}, u_{I_{10}})$
at MPFP P^*
in standard normal space

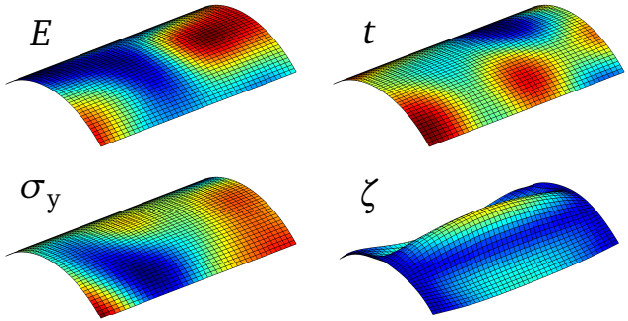
Application examples #6 and 7

- **Problem description** (Bourinet, 2018)
  - ✓ FE-based buckling with random fields ≈ series system with 4 MPFPs
  - ✓ 16 or 93 i.i.d. random inputs $U_i \sim \mathcal{N}(0,1)$
  - ✓ LSF: $G(\mathbf{u}) = \min_{k \in \{1,\dots,4\}} G_k(\mathbf{u})$

    where $\left| \begin{array}{l} G_k(\mathbf{u}) = \beta_k - \boldsymbol{\alpha}_k^{\mathrm{T}} \mathbf{u} \\ \beta_k \approx 4 \end{array} \right.$ for $k = 1, \dots, 4$



Buckling of a shell roof under uniformly-distributed vertical load (Scordelis and Lo, 1964)

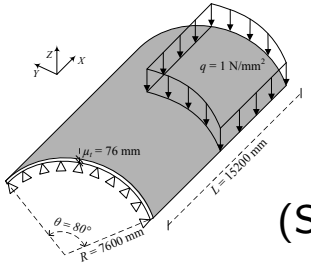


Application examples #6 and 7

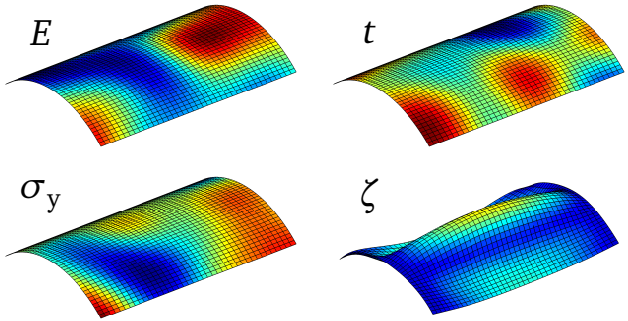
Problem description (Bourinet, 2018)

- ✓ FE-based buckling with random fields ≈ series system with 4 MPFPs
- ✓ 16 or 93 i.i.d. random inputs $U_i \sim \mathcal{N}(0,1)$
- ✓ LSF: $G(\mathbf{u}) = \min_{k \in \{1,\dots,4\}} G_k(\mathbf{u})$

where $\left| \begin{array}{l} G_k(\mathbf{u}) = \beta_k - \boldsymbol{\alpha}_k^{\mathrm{T}} \mathbf{u} \\ \beta_k \approx 4 \end{array} \right.$ for $k = 1, \dots, 4$



Buckling of a shell roof under uniformly-distributed vertical load (Scordelis and Lo, 1964)



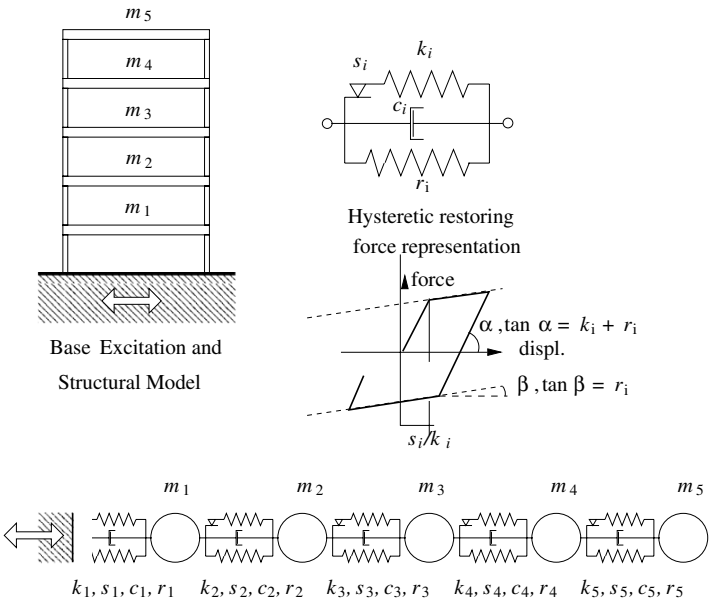
Remarks

- ✓ 4 MPFPs of equal weights
- ✓ Pairwise distances between the 4 MPFPs:

$$\mathbf{D} = [d_{ij}]_{1 \le i,j \le 4} = \begin{bmatrix} 0 & 5.69 & 5.71 & 4.69 \\ 5.69 & 0 & 4.76 & 5.76 \\ 5.71 & 4.76 & 0 & 5.65 \\ 4.69 & 5.76 & 5.65 & 0 \end{bmatrix} \quad \text{where} \quad d_{ij} = \mathrm{dist}(P_i^*, P_j^*)$$

Application example #8

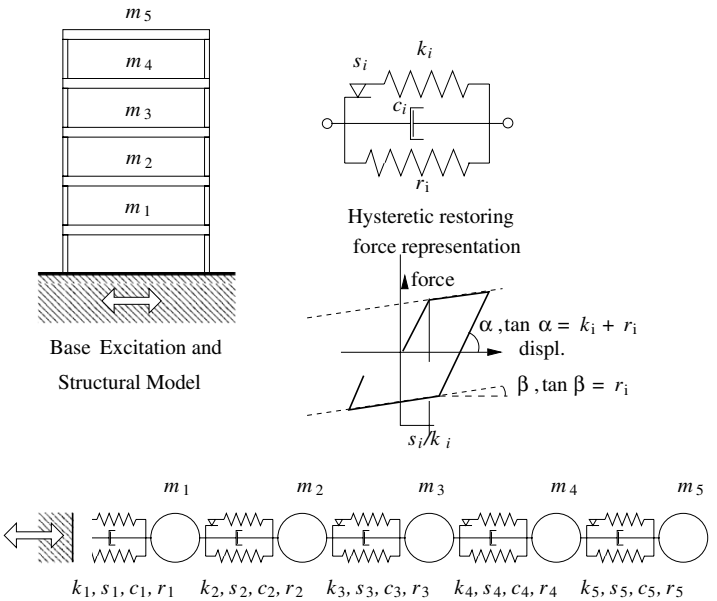
- **Problem description** (Schuëller and Pradlwarter, 2007)
  - ✓ 225 random inputs: 25 independent structural parameters, horizontal ground acceleration modeled by a random process with 200 i.i.d. standard normal rv's
  - ✓ LSF: $g(\mathbf{x}) = 0.022\text{m} - \max_{[0-20\text{sec}]}\left(\left|u_5 - u_4\right|\right)$
  - ✓ Failure probability: $p_{\text{fref}} = 2.41 \times 10^{-4}$ (SS)



Application example #8

Problem description (Schuëller and Pradlwarter, 2007)

- ✓ 225 random inputs:
  25 independent structural parameters, horizontal ground acceleration modeled by a random process with 200 i.i.d. standard normal rv's
- ✓ LSF: $g(\mathbf{x}) = 0.022\text{m} - \max_{[0-20\text{sec}]}\left(\left|u_5 - u_4\right|\right)$
- ✓ Failure probability:
  $p_{\text{fref}} = 2.41 \times 10^{-4}$ (SS)

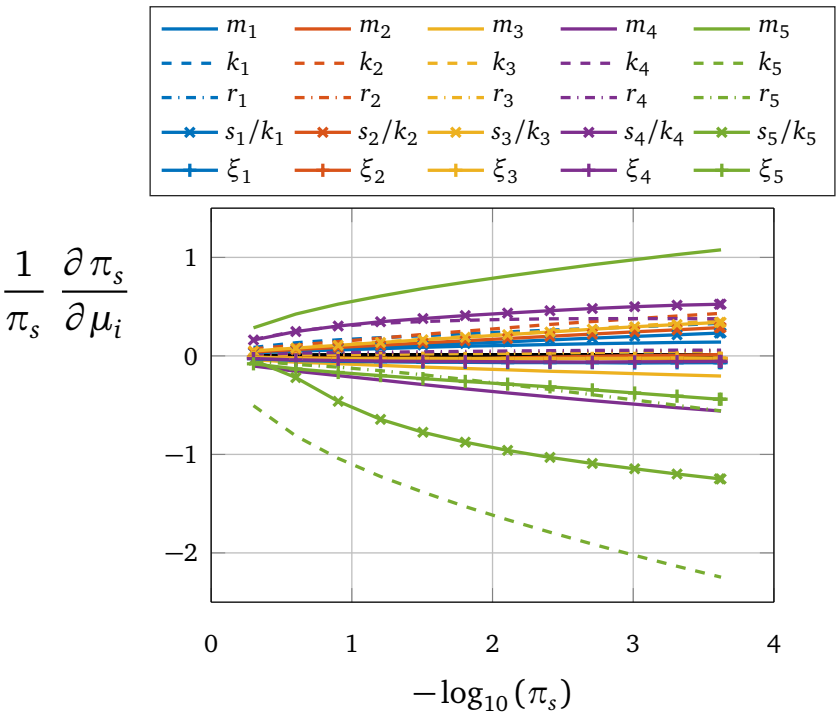


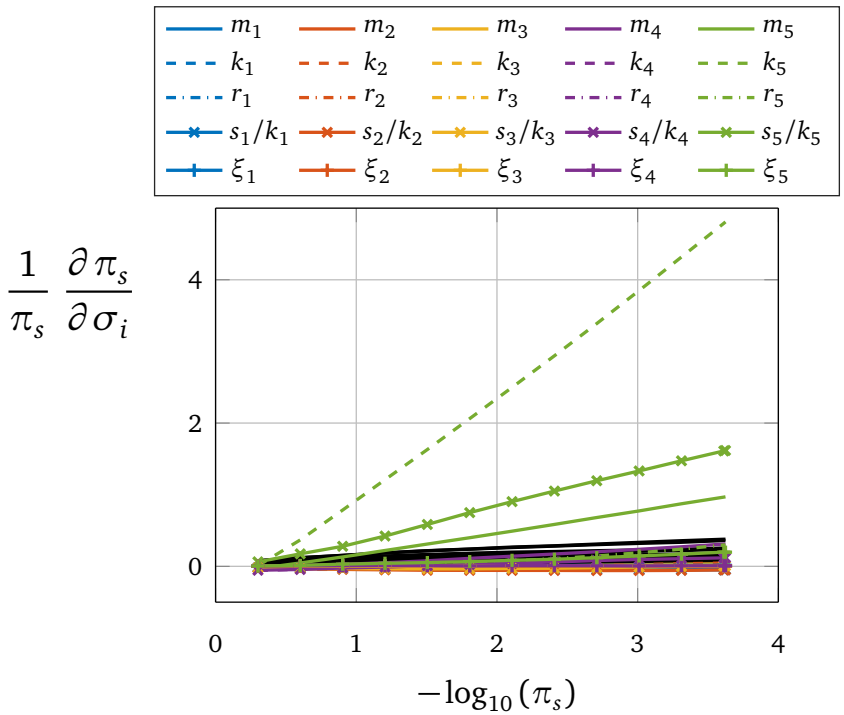
Remark

- ✓ FORM solution not available (no convervence)
- ✓ The 200 random inputs of the load representation play as a group

Application example #8

SS-based sensitivities in standard normal space





Application example #1

- **Problem description** (Bourinet, 2018)
  - ✓ 2 i.i.d. random inputs $U_i \sim \mathcal{N}(0,1)$
  - ✓ LSF: $G(\mathbf{u}) = \min_{k\in\{1,2\}} G_k(\mathbf{u})$

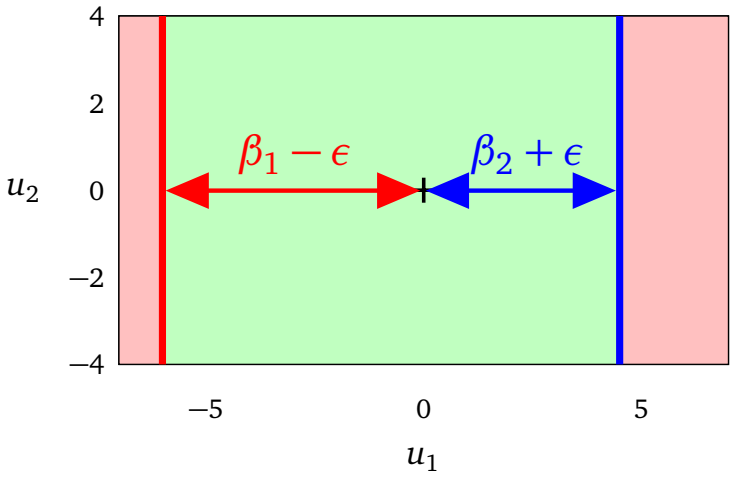
    where $G_1(\mathbf{u}) = (u_1 - \epsilon) + \beta_1$, $G_2(\mathbf{u}) = \beta_1 \left(1 - \left[\frac{1}{2}\left(\frac{u_1-\epsilon}{\beta_2} + \left|\frac{u_1-\epsilon}{\beta_2}\right|\right)\right]^{\gamma}\right)$

    and where $\beta_1 = 6$, $\beta_2 = 4.5$, $\gamma = 30$ and $\epsilon = 10^{-6}$
  - ✓ Failure probability:

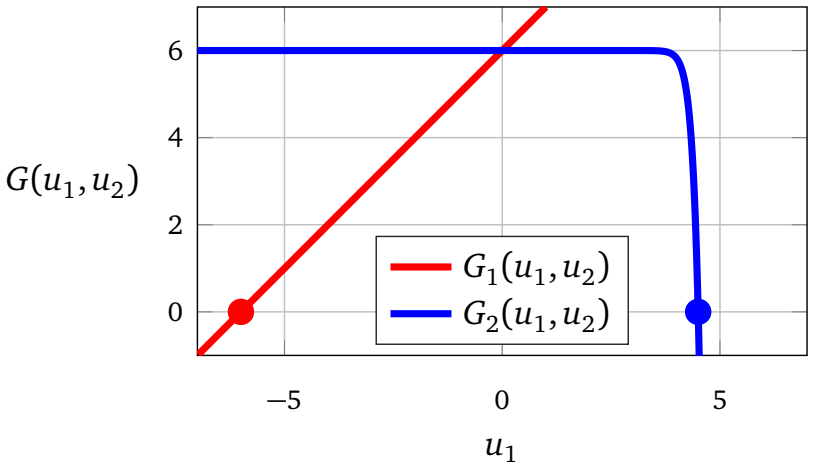
    $p_{\text{fref}} = \Phi(-(\beta_1 - \epsilon)) + \Phi(-(\beta_2 + \epsilon)) \approx \Phi(-\beta_2) = 3.40 \times 10^{-6}$

Remark

- ✓ Biased solution with subset simulation ($\approx$ 3 orders of magnitude here on failure probability), due to specific gradient of LSF



(a) LSS in standard normal space



(b) LSF G_1 and G_2 vs. u_1

Take-home messages

- **Complexity of reliability assessment problems**
  - Aggregates dimension of input space AND complexity of LSF/LSS
  - Highly nonlinear computational models may often exhibit a linear LSS in the $\mathbf{u}$-space!
  - Do not be impressed by FE models with large numbers of d.o.f., their LSS may often be linear in the $\mathbf{u}$-space!

Take-home messages

- **Complexity of reliability assessment problems**
  - Aggregates dimension of input space AND complexity of LSF/LSS
  - Highly nonlinear computational models may often exhibit a linear LSS in the $\mathbf{u}$-space!
  - Do not be impressed by FE models with large numbers of d.o.f., their LSS may often be linear in the $\mathbf{u}$-space!

- **Reliability assessement methods**
  - Clearly present the properties of the problem which are solved
  - If your method has not been tested on more difficult RA problems, then assume that it cannot solve them!

Take-home messages

- **Complexity of reliability assessment problems**
  - Aggregates dimension of input space AND complexity of LSF/LSS
  - Highly nonlinear computational models may often exhibit a linear LSS in the $\mathbf{u}$-space!
  - Do not be impressed by FE models with large numbers of d.o.f., their LSS may often be linear in the $\mathbf{u}$-space!
- **Reliability assessement methods**
  - Clearly present the properties of the problem which are solved
  - If your method has not been tested on more difficult RA problems, then assume that it cannot solve them!
- **Benchmarks/challenges**
  - So far no universal method has really emerged in surrogate-based RA
  - We need benchmarks/challenges to clearly identify the techniques which work or do not work

Objective

Construct a surrogate $\widetilde{G}$ of G : $\left| \begin{array}{l} \mathbb{R}^n \to \mathbb{R} \\ \mathbf{u} \mapsto y = G(\mathbf{u}) \end{array} \right.$

from a set of training samples $\mathcal{T} = \{(\mathbf{u}_i, y_i) \in \mathbb{R}^n \times \mathbb{R}, 1 \leq i \leq N\}$ as small as possible, where $y_i = G(\mathbf{u}_i)$

Supervised learning
Specific context: **small data sets**

Surrogate-based RA

Objective

Construct a surrogate $\widetilde{G}$ of G : $\left| \begin{array}{l} \mathbb{R}^n \to \mathbb{R} \\ \mathbf{u} \mapsto y = G(\mathbf{u}) \end{array} \right.$

from a set of training samples $\mathcal{T} = \{(\mathbf{u}_i, y_i) \in \mathbb{R}^n \times \mathbb{R}, 1 \leq i \leq N\}$ as small as possible, where $y_i = G(\mathbf{u}_i)$

Supervised learning
Specific context: **small data sets**

Choices

- ✓ Use of kernel techniques (Gaussian processes, support vector machines) as surrogate models
- ✓ Regression problem to be solved
- ✓ Adaptive construction of $\widetilde{G}$, with sequential enrichments of the training set

Focus of the talk:
Adaptive RA based on support vector regression (SVR)

Kernel-based approximation

We search for an approximation of $y(\mathbf{x})$ in the form $\widetilde{y}(\mathbf{x};\mathcal{T}) = \widetilde{y}(\mathbf{x}) = h(\mathbf{x}) + b$ where $h \in \mathcal{H}_k$ and $b \in \mathbb{R}$ are solutions of:

$$\min_{h,b} C \sum_{i=1}^{N} \ell\left(y_i, h(\mathbf{x}_i) + b\right) + \frac{1}{2}\|h\|^2_{\mathcal{H}_k}$$

where $\ell : \mathcal{Y} \times \mathbb{R} \to \mathbb{R}$ is a loss function and $C > 0$ a regularization parameter

Kernel-based approximation

We search for an approximation of $y(\mathbf{x})$ in the form $\widetilde{y}(\mathbf{x};\mathcal{T}) = \widetilde{y}(\mathbf{x}) = h(\mathbf{x}) + b$ where $h \in \mathcal{H}_k$ and $b \in \mathbb{R}$ are solutions of:

$$\min_{h,b} \boxed{C \sum_{i=1}^{N} \ell\left(y_i, h\left(\mathbf{x}_i\right) + b\right)} + \boxed{\frac{1}{2}\|h\|_{\mathcal{H}_k}^2}$$

where $\ell : \mathcal{Y} \times \mathbb{R} \to \mathbb{R}$ is a loss function and $C > 0$ a regularization parameter

First term

Enforces the fit of the approximate model $\widetilde{y}$ to the training data $\mathcal{T}$

Second term

Enforces a small norm of h in the RKHS, which results in a sufficiently smooth solution $\widetilde{y}$

Kernel-based approximation

We search for an approximation of $y(\mathbf{x})$ in the form $\widetilde{y}(\mathbf{x};\mathcal{T}) = \widetilde{y}(\mathbf{x}) = h(\mathbf{x}) + b$ where $h \in \mathcal{H}_k$ and $b \in \mathbb{R}$ are solutions of:

$$\min_{h,b} \boxed{C \sum_{i=1}^{N} \ell\left(y_i, h\left(\mathbf{x}_i\right) + b\right)} + \boxed{\frac{1}{2}\|h\|_{\mathcal{H}_k}^2}$$

where $\ell : \mathcal{Y} \times \mathbb{R} \to \mathbb{R}$ is a loss function and $C > 0$ a regularization parameter

First term

Enforces the fit of the approximate model $\widetilde{y}$ to the training data $\mathcal{T}$

Second term

Enforces a small norm of h in the RKHS, which results in a sufficiently smooth solution $\widetilde{y}$

The solution h is given by: $\boxed{h\left(\mathbf{x}\right) = \sum_{i=1}^{N} c_i k(\mathbf{x}_i, \mathbf{x})}$

where $\mathbf{c} = (c_1, \ldots, c_N)^{\mathrm{T}} \in \mathbb{R}^N$ is the vector of unknown expansion coefficients and $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a positive definite kernel (reproducing kernel of RKHS)

Overview of kernel-based methods

	Loss function $\ell(y,u)$	Usual notation
Binary classification	$(1-yu)_+$ (hinge)	L1-SVC
	$\frac{1}{2}(1-yu)_+^2$ (square hinge)	L2-SVC
Regression	$(\lvert y-u\rvert-\epsilon)_+$ (ϵ-insensitive)	L1-ϵ-SVR
	$\frac{1}{2}(\lvert y-u\rvert-\epsilon)_+^2$ (squared ϵ-insensitive)	L2-ϵ-SVR
	$\frac{1}{2}(y-u)^2$ (square)	LS-SVR, kriging

Important remarks

- Least squares support vector regression (LS-SVR) (Suykens et al., 2002) $\equiv$ L2-ϵ-SVR with $\epsilon = 0$
- LS-SVR approximation $\widetilde{y} \equiv$ kriging mean μ (Bourinet, 2018)

Noisy kriging with a trend

- Optimality condition

$$\begin{bmatrix} \mathbf{K} + \tau^2 \mathbf{1} & \mathbf{P} \\ \mathbf{P}^{\mathrm{T}} & \mathbf{0} \end{bmatrix} \begin{pmatrix} \mathbf{c} \\ \mathbf{d} \end{pmatrix} = \begin{pmatrix} \mathbf{y} \\ \mathbf{0} \end{pmatrix}$$

- Kriging mean

$$\begin{aligned} \widetilde{y}(\mathbf{x}) &= \mu_{\mathsf{UK+noise}}(\mathbf{x}) \\ &= \mathbf{k}(\mathbf{x})^{\mathrm{T}} \mathbf{c} + \mathbf{p}(\mathbf{x})^{\mathrm{T}} \mathbf{d} \\ &= \sum_{i=1}^{N} c_i k(\mathbf{x}_i, \mathbf{x}) + \sum_{j=1}^{M} d_j p_j(\mathbf{x}) \end{aligned}$$

Noisy kriging with a trend vs. LS-SVR

Noisy kriging with a trend

- Optimality condition

$$\begin{bmatrix} \mathbf{K} + \tau^2 \mathbf{1} & \mathbf{P} \\ \mathbf{P}^{\mathrm{T}} & \mathbf{0} \end{bmatrix} \begin{pmatrix} \mathbf{c} \\ \mathbf{d} \end{pmatrix} = \begin{pmatrix} \mathbf{y} \\ \mathbf{0} \end{pmatrix}$$

- Kriging mean

$$\begin{aligned} \widetilde{y}(\mathbf{x}) &= \mu_{\mathsf{UK+noise}}(\mathbf{x}) \\ &= \mathbf{k}(\mathbf{x})^{\mathrm{T}} \mathbf{c} + \mathbf{p}(\mathbf{x})^{\mathrm{T}} \mathbf{d} \\ &= \sum_{i=1}^{N} c_i k(\mathbf{x}_i, \mathbf{x}) + \sum_{j=1}^{M} d_j p_j(\mathbf{x}) \end{aligned}$$

LS-SVR

- Dual optimization problem

$$\begin{bmatrix} \mathbf{K} + C^{-1} \mathbf{I} & \mathbf{1} \\ \mathbf{1}^{\mathrm{T}} & 0 \end{bmatrix} \begin{pmatrix} \boldsymbol{\alpha} \\ b \end{pmatrix} = \begin{pmatrix} \mathbf{y} \\ 0 \end{pmatrix}$$

- Decision function

$$\begin{aligned} \widetilde{y}(\mathbf{x}) &= \widetilde{f}(\mathbf{x}) \\ &= \mathbf{k}(\mathbf{x})^{\mathrm{T}} \boldsymbol{\alpha} + b \\ &= \sum_{i=1}^{N} \alpha_i k(\mathbf{x}_i, \mathbf{x}) + b \end{aligned}$$

Noisy kriging with a trend vs. LS-SVR

Noisy kriging with a trend

- Optimality condition

$$\begin{bmatrix} \mathbf{K} + \tau^2 \mathbf{1} & \mathbf{P} \\ \mathbf{P}^{\mathrm{T}} & \mathbf{0} \end{bmatrix} \begin{pmatrix} \mathbf{c} \\ \mathbf{d} \end{pmatrix} = \begin{pmatrix} \mathbf{y} \\ \mathbf{0} \end{pmatrix}$$

- Kriging mean

$$\begin{aligned} \widetilde{y}(\mathbf{x}) &= \mu_{\mathrm{UK+noise}}(\mathbf{x}) \\ &= \mathbf{k}(\mathbf{x})^{\mathrm{T}} \mathbf{c} + \mathbf{p}(\mathbf{x})^{\mathrm{T}} \mathbf{d} \\ &= \sum_{i=1}^{N} c_i k(\mathbf{x}_i, \mathbf{x}) + \sum_{j=1}^{M} d_j p_j(\mathbf{x}) \end{aligned}$$

LS-SVR

- Dual optimization problem

$$\begin{bmatrix} \mathbf{K} + C^{-1} \mathbf{I} & \mathbf{1} \\ \mathbf{1}^{\mathrm{T}} & 0 \end{bmatrix} \begin{pmatrix} \boldsymbol{\alpha} \\ b \end{pmatrix} = \begin{pmatrix} \mathbf{y} \\ 0 \end{pmatrix}$$

- Decision function

$$\begin{aligned} \widetilde{y}(\mathbf{x}) &= \widetilde{f}(\mathbf{x}) \\ &= \mathbf{k}(\mathbf{x})^{\mathrm{T}} \boldsymbol{\alpha} + b \\ &= \sum_{i=1}^{N} \alpha_i k(\mathbf{x}_i, \mathbf{x}) + b \end{aligned}$$

Remarks

- LS-SVR with bias term b fully identical to ordinary kriging with noise ($M = 1$, with $p_1 : \mathcal{X} \to \mathbb{R}, \mathbf{x} \mapsto 1$). τ^2 and C^{-1} play the same role
- Remember that LS-SVR is a subcase of L2-ϵ-SVR with ϵ set to zero

Optimization problem to solve for L1-ϵ-SVR

Dual optimization problem in matrix form

$$\min_{\boldsymbol{\alpha},\boldsymbol{\alpha}^*} \frac{1}{2}\begin{pmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\alpha}^*\end{pmatrix}^{\mathrm{T}}\begin{bmatrix}\mathbf{K} & -\mathbf{K}\\ -\mathbf{K} & \mathbf{K}\end{bmatrix}\begin{pmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\alpha}^*\end{pmatrix}+\begin{pmatrix}\epsilon\mathbf{1}-\mathbf{y}\\ \epsilon\mathbf{1}+\mathbf{y}\end{pmatrix}^{\mathrm{T}}\begin{pmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\alpha}^*\end{pmatrix}$$

$$\text{s.t.}\quad \begin{pmatrix}\mathbf{1}\\ -\mathbf{1}\end{pmatrix}^{\mathrm{T}}\begin{pmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\alpha}^*\end{pmatrix}=0 \quad , \quad \begin{pmatrix}\mathbf{0}\\ \mathbf{0}\end{pmatrix}\leq\begin{pmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\alpha}^*\end{pmatrix}\leq C\begin{pmatrix}\mathbf{1}\\ \mathbf{1}\end{pmatrix}$$

where $\mathbf{y}=(y_1,\ldots,y_N)^{\mathrm{T}}$, $\mathbf{K}=\left[k\left(\mathbf{x}_i,\mathbf{x}_j\right)\right]_{1\leq i,j\leq N}$ is the Gram matrix

Optimization problem to solve for L1-ϵ-SVR

- **Dual optimization problem in matrix form**

$$\min_{\boldsymbol{\alpha},\boldsymbol{\alpha}^*} \frac{1}{2}\begin{pmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\alpha}^*\end{pmatrix}^{\mathrm{T}}\begin{bmatrix}\mathbf{K} & -\mathbf{K}\\ -\mathbf{K} & \mathbf{K}\end{bmatrix}\begin{pmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\alpha}^*\end{pmatrix}+\begin{pmatrix}\epsilon\mathbf{1}-\mathbf{y}\\ \epsilon\mathbf{1}+\mathbf{y}\end{pmatrix}^{\mathrm{T}}\begin{pmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\alpha}^*\end{pmatrix}$$

$$\text{s.t.} \quad \begin{pmatrix}\mathbf{1}\\ -\mathbf{1}\end{pmatrix}^{\mathrm{T}}\begin{pmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\alpha}^*\end{pmatrix}=0 \quad , \quad \begin{pmatrix}\mathbf{0}\\ \mathbf{0}\end{pmatrix}\leq\begin{pmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\alpha}^*\end{pmatrix}\leq C\begin{pmatrix}\mathbf{1}\\ \mathbf{1}\end{pmatrix}$$

where $\mathbf{y}=(y_1,\ldots,y_N)^{\mathrm{T}}$, $\mathbf{K}=\left[k\left(\mathbf{x}_i,\mathbf{x}_j\right)\right]_{1\leq i,j\leq N}$ is the Gram matrix

Remarks

- Dual QP problem to solve of size $2N$
- Dual QP convex, since $\mathbf{K}$ (and therefore $\begin{bmatrix}\mathbf{K} & -\mathbf{K}\\ -\mathbf{K} & \mathbf{K}\end{bmatrix}$) are positive semi-definite matrices

Kriging versus ϵ-SVR

	Kriging	ϵ-SVR
Loss function	square	ϵ-insensitive
Problem to solve	linear	QP
Regularization	homogeneous noise τ^2 (nugget effect)	$\frac{1}{C}$
Unregularized term		
✓ single constant term	ordinary kriging (unknown mean)	standard SVR (bias term b)
✓ Set of real-valued functions	universal kriging	semi-parametric SVR
Hyperparameter tuning	MLE (,CV)	CV
Probabilistic framework	yes	no

Selected kernels

Gaussian kernel

Isotropic $\quad k_{\mathsf{i}}(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta}) = \exp\left(-\gamma \|\mathbf{x} - \mathbf{x}'\|^2\right)$ where $\boldsymbol{\theta} = \gamma \in \mathbb{R}_{>0}$

Anisotropic $\quad k_{\mathsf{a}}(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta}) = \prod_{i=1}^{n} k_{\mathsf{i}}(x_i, x_i'; \gamma_i)$ where $\boldsymbol{\theta} = (\gamma_1, \ldots, \gamma_n) \in \mathbb{R}^n_{>0}$

Selected kernels

Gaussian kernel

Isotropic $\quad k_{\mathrm{i}}(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta}) = \exp\left(-\gamma \|\mathbf{x} - \mathbf{x}'\|^2\right)$ where $\boldsymbol{\theta} = \gamma \in \mathbb{R}_{>0}$

Anisotropic $\quad k_{\mathrm{a}}(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta}) = \prod_{i=1}^{n} k_{\mathrm{i}}(x_i, x_i'; \gamma_i)$ where $\boldsymbol{\theta} = (\gamma_1, \ldots, \gamma_n) \in \mathbb{R}^n_{>0}$

$C^{2\nu}$-Matérn kernel

Isotropic $\quad k_{\mathrm{i}}(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta}) = \frac{1}{2^{\nu-1}\Gamma(\nu)} \left(\gamma\sqrt{2\nu}\|\mathbf{x} - \mathbf{x}'\|\right)^{\nu} K_{\nu}\left(\gamma\sqrt{2\nu}\|\mathbf{x} - \mathbf{x}'\|\right)$

where $\boldsymbol{\theta} = (\nu, \gamma)$, $\nu \in \mathbb{R}_{\geq \frac{1}{2}}$ (regularity parameter), $\gamma \in \mathbb{R}_{>0}$

and K_{ν} : modified Bessel function of 2nd kind of order ν

Anisotropic $\quad k_{\mathrm{a}}(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta}) = \prod_{i=1}^{n} k_{\mathrm{i}}(x_i, x_i'; (\nu_i, \gamma_i))$

where $\boldsymbol{\theta} = (\boldsymbol{\nu}, \boldsymbol{\gamma})$,

$\boldsymbol{\nu} = (\nu_1, \ldots, \nu_n) \in \mathbb{R}^n_{\geq \frac{1}{2}}$ and $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_n) \in \mathbb{R}^n_{>0}$

Hyperparameter selection

The approximate model $\widetilde{y}(\mathbf{x};\boldsymbol{\theta})$ to train on $\mathcal{T}$ is defined by means of a set of hyperparameters $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_{n_{\boldsymbol{\theta}}}) \in \Theta$ which gathers those

- ✓ of the selected kernel $k(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta}_k)$, denoted by $\boldsymbol{\theta}_k$
- ✓ of the SVR formulation: regularization parameter C, width ϵ of the ϵ-tube

Hyperparameter selection

The approximate model $\widetilde{y}(\mathbf{x};\boldsymbol{\theta})$ to train on $\mathcal{T}$ is defined by means of a set of hyperparameters $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_{n_\theta}) \in \Theta$ which gathers those

- ✓ of the selected kernel $k(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta}_k)$, denoted by $\boldsymbol{\theta}_k$
- ✓ of the SVR formulation: regularization parameter C, width ϵ of the ϵ-tube

Hyperparameter tuning

Find $\left(C^*, \epsilon^*, \boldsymbol{\theta}_k^*\right)$ solution of

$$\left(C^*, \epsilon^*, \boldsymbol{\theta}_k^*\right) = \underset{C,\epsilon,\boldsymbol{\theta}_k}{\arg\min}\ \mathrm{Err}_{\mathrm{LOO},\ell_1}$$

where $\mathrm{Err}_{\mathrm{LOO},\ell_1} = \dfrac{1}{N}\displaystyle\sum_{i=1}^{N} |y_i - \widetilde{y}^{(-i)}(\mathbf{x}_i)|$ is conveniently replaced by:

$$\mathrm{Err}^{\mathrm{span}}_{\mathrm{LOO},\ell_1} = \frac{1}{N}\sum_{i=1}^{N}(\alpha_i + \alpha_i^*)S_i^2 + \frac{1}{N}\sum_{i=1}^{N}(\xi_i + \xi_i^*) + \epsilon$$

Span approximation of the LOO error (Chang and Lin, 2005)

and $S_i^2 = \dfrac{1}{(\widetilde{\mathbf{K}}^{-1})_{ii}}$, where $\widetilde{\mathbf{K}} = \begin{bmatrix} \mathbf{K} & \mathbf{1} \\ \mathbf{1}^{\mathrm{T}} & 0 \end{bmatrix}$ and $\mathbf{K} = [k(\mathbf{x}_i, \mathbf{x}_j; \boldsymbol{\theta}_k)]_{1 \le i,j \le N}$

SVR in practice!

Dual quadratic program to solve

- ✓ SVR prediction accuracy requires a fine hyperparameter tuning
  - ▶ Accuracy needed on span approximation of LOO error
  - ▶ Accuracy needed on solution of QP solution: $\boldsymbol{\alpha}, \boldsymbol{\alpha}^*, \xi, \xi^*$
  - ▶ Avoid Sequential Minimal Optimization (SMO) algorithm such as implemented in LIBSVM (Chang and Lin, 2011): fast for large training sets but inaccurate
  - ▶ Prefer interior-point methods
- ✓ My feedback: MATLAB Optimization Toolbox (The MathWorks, 2012), MOSEK (MOSEK ApS, 2014), QPC (Wills, 2009)

SVR in practice!

Dual quadratic program to solve

- ✓ SVR prediction accuracy requires a fine hyperparameter tuning
  - ▶ Accuracy needed on span approximation of LOO error
  - ▶ Accuracy needed on solution of QP solution: $\boldsymbol{\alpha}, \boldsymbol{\alpha}^*, \xi, \xi^*$
  - ▶ Avoid Sequential Minimal Optimization (SMO) algorithm such as implemented in LIBSVM (Chang and Lin, 2011): fast for large training sets but inaccurate
  - ▶ Prefer interior-point methods
- ✓ My feedback: MATLAB Optimization Toolbox (The MathWorks, 2012), MOSEK (MOSEK ApS, 2014), QPC (Wills, 2009)

Hyperparameter tuning

- ✓ Optimization problem hard to solve (noisy, several local minima)
- ✓ Solved by means of CMA-ES algorithm (Hansen, 2016) with specifically-derived stopping criteria
- ✓ QP sent in parallel for sampled hyperparameters on a multi-core CPU
- ✓ Kernel evaluation tasks sent to a GPU (NVIDIA Tesla P100)

SVR-based adaptive method for reliability assessment

- **Solving strategy** (Bourinet, 2016)
  1. Define: initial data set $\mathcal{D} = \{ (\mathbf{u}_i, G(\mathbf{u}_i)) , 1 \leq i \leq N \}$, initial LSF level y_1

SVR-based adaptive method for reliability assessment

- **Solving strategy** (Bourinet, 2016)
  1. Define: initial data set $\mathcal{D} = \{ (\mathbf{u}_i, G(\mathbf{u}_i)) , 1 \leq i \leq N \}$, initial LSF level y_1
  2. Iterate on s until convergence
     - Define training set $\mathcal{T}_s = \left\{ (\mathbf{u}_i, G(\mathbf{u}_i)) , i \in \mathcal{I}_{\text{train},s} \right\}$

SVR-based adaptive method for reliability assessment

- **Solving strategy** (Bourinet, 2016)
  1. Define: initial data set $\mathcal{D} = \{ (\mathbf{u}_i, G(\mathbf{u}_i)) , 1 \leq i \leq N \}$, initial LSF level y_1
  2. Iterate on s until convergence
     - Define training set $\mathcal{T}_s = \left\{ (\mathbf{u}_i, G(\mathbf{u}_i)) , i \in \mathcal{I}_{\text{train},s} \right\}$
     - Train LSF surrogate $\widetilde{G}_s$ on $\mathcal{T}_s$

SVR-based adaptive method for reliability assessment

- **Solving strategy** (Bourinet, 2016)
  1. Define: initial data set $\mathcal{D} = \{ (\mathbf{u}_i, G(\mathbf{u}_i)) , 1 \leq i \leq N \}$, initial LSF level y_1
  2. Iterate on s until convergence
     - Define training set $\mathcal{T}_s = \left\{ (\mathbf{u}_i, G(\mathbf{u}_i)) , i \in \mathcal{I}_{\text{train},s} \right\}$
     - Train LSF surrogate $\widetilde{G}_s$ on $\mathcal{T}_s$
     - Compute probability estimate $\widehat{p}_s$ of $p_s = \mathbb{P}\left(\widetilde{G}_s(\mathbf{U}) \leq y_s\right)$

SVR-based adaptive method for reliability assessment

- **Solving strategy** (Bourinet, 2016)
  1. Define: initial data set $\mathcal{D} = \{ (\mathbf{u}_i, G(\mathbf{u}_i)) , 1 \leq i \leq N \}$, initial LSF level y_1
  2. Iterate on s until convergence
     - Define training set $\mathcal{T}_s = \left\{ (\mathbf{u}_i, G(\mathbf{u}_i)) , i \in \mathcal{I}_{\text{train},s} \right\}$
     - Train LSF surrogate $\widetilde{G}_s$ on $\mathcal{T}_s$
     - Compute probability estimate $\widehat{p}_s$ of $p_s = \mathbb{P}\left(\widetilde{G}_s(\mathbf{U}) \leq y_s\right)$
     - Select N_{a} new points and add corresponding data pairs to $\mathcal{D}$

SVR-based adaptive method for reliability assessment

Solving strategy (Bourinet, 2016)

1. Define: initial data set $\mathcal{D} = \{ (\mathbf{u}_i, G(\mathbf{u}_i)) , 1 \leq i \leq N \}$, initial LSF level y_1
2. Iterate on s until convergence
   - Define training set $\mathcal{T}_s = \left\{ (\mathbf{u}_i, G(\mathbf{u}_i)) , i \in \mathcal{I}_{\text{train},s} \right\}$
   - Train LSF surrogate $\widetilde{G}_s$ on $\mathcal{T}_s$
   - Compute probability estimate $\widehat{p}_s$ of $p_s = \mathbb{P}\left(\widetilde{G}_s(\mathbf{U}) \leq y_s\right)$
   - Select N_{a} new points and add corresponding data pairs to $\mathcal{D}$
   - Update y_s and $\mathcal{T}_s$

SVR-based adaptive method for reliability assessment

Solving strategy (Bourinet, 2016)

1. Define: initial data set $\mathcal{D} = \{ (\mathbf{u}_i, G(\mathbf{u}_i)) , 1 \leq i \leq N \}$, initial LSF level y_1
2. Iterate on s until convergence
   - Define training set $\mathcal{T}_s = \left\{ (\mathbf{u}_i, G(\mathbf{u}_i)) , i \in \mathcal{I}_{\text{train},s} \right\}$
   - Train LSF surrogate $\widetilde{G}_s$ on $\mathcal{T}_s$
   - Compute probability estimate $\widehat{p}_s$ of $p_s = \mathbb{P}\left(\widetilde{G}_s(\mathbf{U}) \leq y_s\right)$
   - Select N_{a} new points and add corresponding data pairs to $\mathcal{D}$
   - Update y_s and $\mathcal{T}_s$
3. Failure probability estimate $= \widehat{p}_{s_{\max}} = \mathbb{P}\left(\widetilde{G}_{s_{\max}}(\mathbf{U}) \leq y_{s_{\max}}\right)$ where $y_{s_{\max}} = 0$

SVR-based adaptive method for reliability assessment

Solving strategy (Bourinet, 2016)

1. Define: initial data set $\mathcal{D} = \{ (\mathbf{u}_i, G(\mathbf{u}_i)) , 1 \leq i \leq N \}$, initial LSF level y_1
2. Iterate on s until convergence
   - Define training set $\mathcal{T}_s = \left\{ (\mathbf{u}_i, G(\mathbf{u}_i)) , i \in \mathcal{I}_{\text{train},s} \right\}$
   - Train LSF surrogate $\widetilde{G}_s$ on $\mathcal{T}_s$
   - Compute probability estimate $\widehat{p}_s$ of $p_s = \mathbb{P}\left(\widetilde{G}_s(\mathbf{U}) \leq y_s\right)$
   - Select N_{a} new points and add corresponding data pairs to $\mathcal{D}$
   - Update y_s and $\mathcal{T}_s$
3. Failure probability estimate $= \widehat{p}_{s_{\max}} = \mathbb{P}\left(\widetilde{G}_{s_{\max}}(\mathbf{U}) \leq y_{s_{\max}}\right)$ where $y_{s_{\max}} = 0$

Main conceptual ideas

- ✓ $\mathcal{T}_s \subset \mathcal{D}$ and updated at each s (local regression)
- ✓ Subset simulation with componentwise Metropolis algorithm (Au and Beck, 2001) for:
  - Estimation of p_s
  - Selection of additional training points
  - Definition of convergence criteria
- ✓ Weighted averaged surrogate models

Weighted averaged surrogate models

$$\overline{\widetilde{G}}_s(\mathbf{u}) = \sum_{k=\max\{1,s-4\}}^{s} \omega_k \widetilde{G}_k(\mathbf{u}) \quad \text{where} \quad \omega_k = \frac{\min\{s,5\}+k-s}{\sum_{i=1}^{\min\{s,5\}} i}$$

We have $\omega_{s-4} = \frac{1}{15}$, $\ldots$,$\omega_s = \frac{5}{15}$ for $s \geq 5$

Weighted averaged surrogate models

$$\overline{\widetilde{G}}_s(\mathbf{u}) = \sum_{k=\max\{1,s-4\}}^{s} \omega_k \widetilde{G}_k(\mathbf{u}) \quad \text{where} \quad \omega_k = \frac{\min\{s,5\}+k-s}{\sum_{i=1}^{\min\{s,5\}} i}$$

We have $\omega_{s-4} = \frac{1}{15}$, $\ldots$,$\omega_s = \frac{5}{15}$ for $s \geq 5$

Remarks

- Better accuracy of averaged surrogates $\overline{\widetilde{G}}_s$ than non-averaged ones $\widetilde{G}_s$
- Width of the gap between the two following hypersurfaces:

  $$\{\mathbf{u} \in \mathbb{R}^n : \overline{\widetilde{G}}_s(\mathbf{u}) = y_s\} \text{ and } \{\mathbf{u} \in \mathbb{R}^n : \widetilde{G}_s(\mathbf{u}) = y_s\}$$

  used as one of the three convergence criteria

Surrogate-based MCMC samples, new training points

- Apply subset simulation (SS) with componentwise Metropolis algorithm to $\widetilde{G}_s(\mathbf{u}) - y_s$ and $\overline{\widetilde{G}}_s - y_s$ with $N_{\mathrm{SS}} = 100,000$ samples per level
- Keep samples in the last level m of SS verifying:

$$\begin{cases} \mathcal{U}_s = \{\mathbf{u}^{(j)} \in \mathcal{U}_{\mathrm{SS};s}(m) : \widetilde{G}_s(\mathbf{u}^{(j)}) \leq y_s\} \\ \overline{\mathcal{U}}_s = \{\mathbf{u}^{(j)} \in \overline{\mathcal{U}}_{\mathrm{SS};s}(m) : \overline{\widetilde{G}}_s(\mathbf{u}^{(j)}) \leq y_s\} \end{cases}$$

Surrogate-based MCMC samples, new training points

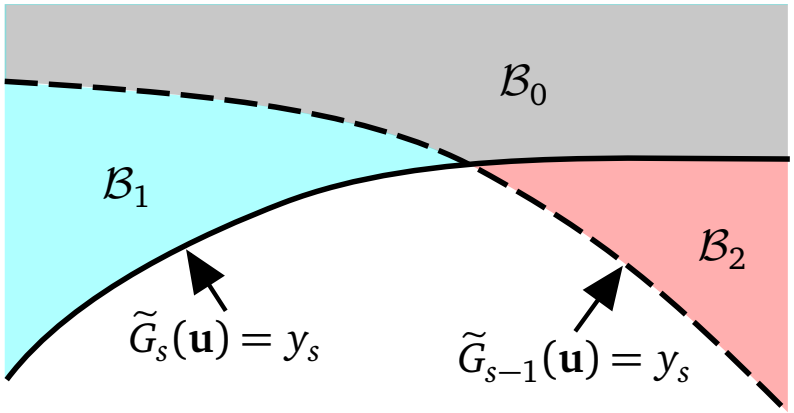
- Apply subset simulation (SS) with componentwise Metropolis algorithm to $\widetilde{G}_s(\mathbf{u}) - y_s$ and $\overline{\widetilde{G}}_s - y_s$ with $N_{\mathrm{SS}} = 100,000$ samples per level
- Keep samples in the last level m of SS verifying:

$$\begin{cases} \mathcal{U}_s = \{\mathbf{u}^{(j)} \in \mathcal{U}_{\mathrm{SS};s}(m) : \widetilde{G}_s(\mathbf{u}^{(j)}) \leq y_s\} \equiv \mathcal{B}_0 \cup \mathcal{B}_1 \\ \overline{\mathcal{U}}_s = \{\mathbf{u}^{(j)} \in \overline{\mathcal{U}}_{\mathrm{SS};s}(m) : \overline{\widetilde{G}}_s(\mathbf{u}^{(j)}) \leq y_s\} \end{cases}$$

- Define sample sets $\mathcal{B}_0$, $\mathcal{B}_1$, $\mathcal{B}_2$
- New training points in:
  - ✓ $\mathcal{B}_0$ and $\mathcal{B}_1$ is $y_s > 0$
  - ✓ $\mathcal{B}_1$ and $\mathcal{B}_2$ if $y_s = 0$
- Number of new training points:

  $\propto \frac{\#\mathcal{B}_0}{\#\mathcal{U}_s}$, $\frac{\#\mathcal{B}_1}{\#\mathcal{U}_s}$ and $\frac{\#\mathcal{B}_2}{\#\mathcal{U}_{s-1}}$

  Sum constrained to $N_{\mathrm{a}} = 5$



Convergence criteria

Convergence in 3 phases

- **Phase 1** until $y_s < 0$
- **Phase 2** until:

$$r_{\mathcal{B}_1\mathcal{B}_2;0.5}(s) < 0.2$$
$$\& \quad r_{\overline{\mathcal{B}}_1\overline{\mathcal{B}}_2;0.5}(s) < 0.03$$
$$\& \quad r_{\overline{\mathcal{D}}_1\mathcal{D}_2;0.5}(s) < 0.2$$

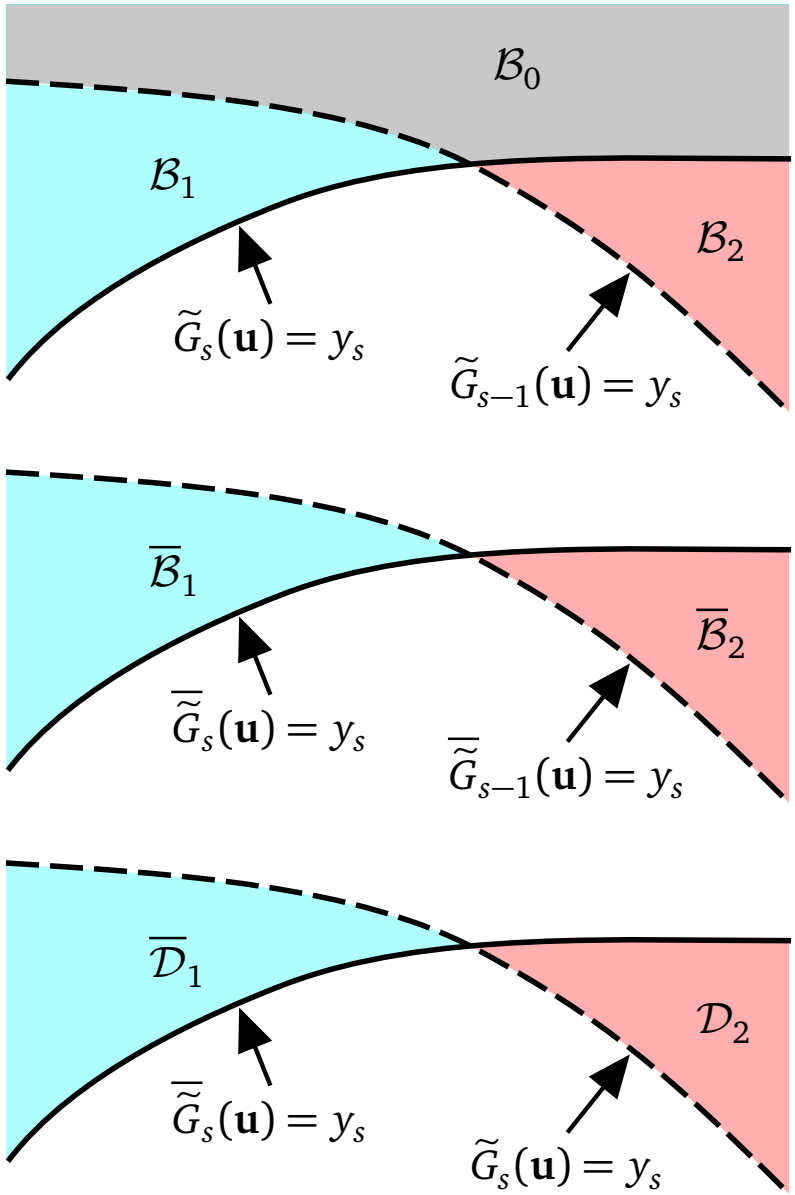
where:

$$r_{\mathcal{B}_1\mathcal{B}_2;0.5}(s) = \frac{1}{2}\left(\frac{\#\mathcal{B}_1}{\#\mathcal{U}_s} + \frac{\#\mathcal{B}_2}{\#\mathcal{U}_{s-1}}\right)$$

$$r_{\overline{\mathcal{B}}_1\overline{\mathcal{B}}_2;0.5}(s) = \frac{1}{2}\left(\frac{\#\overline{\mathcal{B}}_1}{\#\overline{\mathcal{U}}_s} + \frac{\#\overline{\mathcal{B}}_2}{\#\overline{\mathcal{U}}_{s-1}}\right)$$

$$r_{\overline{\mathcal{D}}_1\mathcal{D}_2;0.5}(s) = \frac{1}{2}\left(\frac{\#\overline{\mathcal{D}}_1}{\#\overline{\mathcal{U}}_s} + \frac{\#\mathcal{D}_2}{\#\mathcal{U}_{s-1}}\right)$$

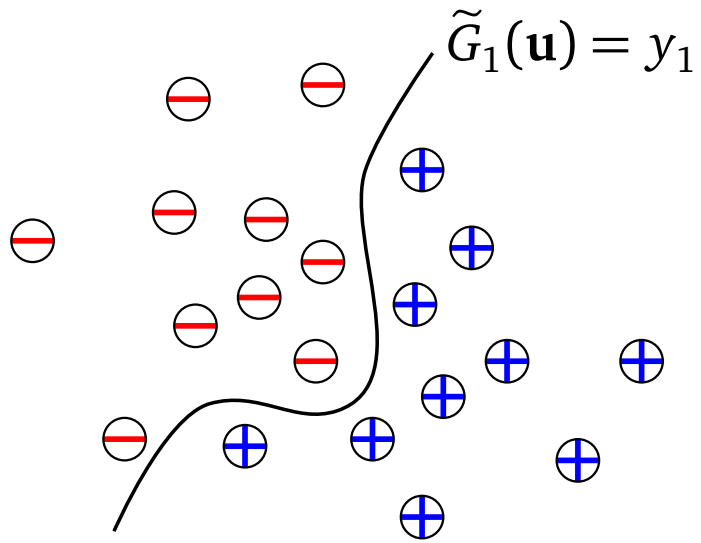
- **Phase 3** : max 20 extra iterations



Training set $\mathcal{T}_s$ and LSF threshold value y_s (1/6)

$\oplus$: $\{\mathbf{u}_i : G(\mathbf{u}_i) > y_1\}$

$\ominus$: $\{\mathbf{u}_i : G(\mathbf{u}_i) < y_1\}$

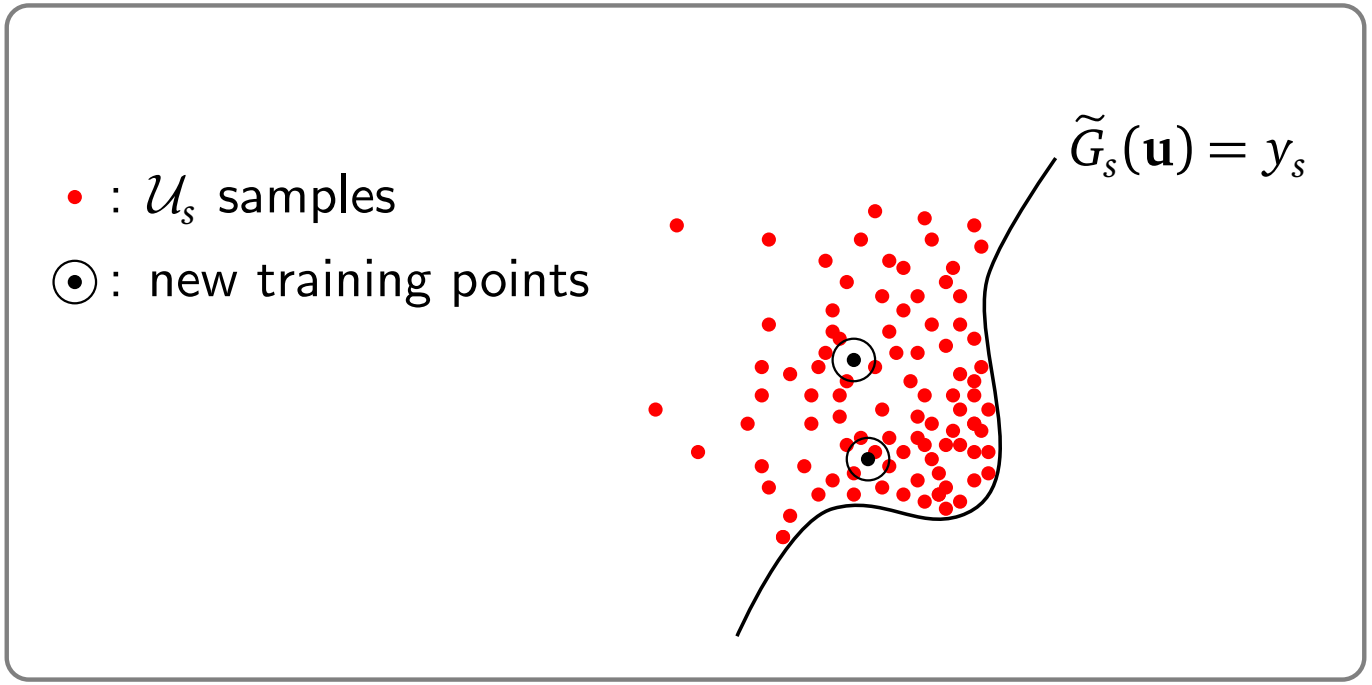


- Selection of 1st intermediate level:

$$y_1 = \text{median}\left(G(\mathbf{u}_i)\,,\ i \in \mathcal{I}_{\text{train};1} \right) \quad \text{where} \quad \mathcal{I}_{\text{train};1} = 1, \ldots, N$$

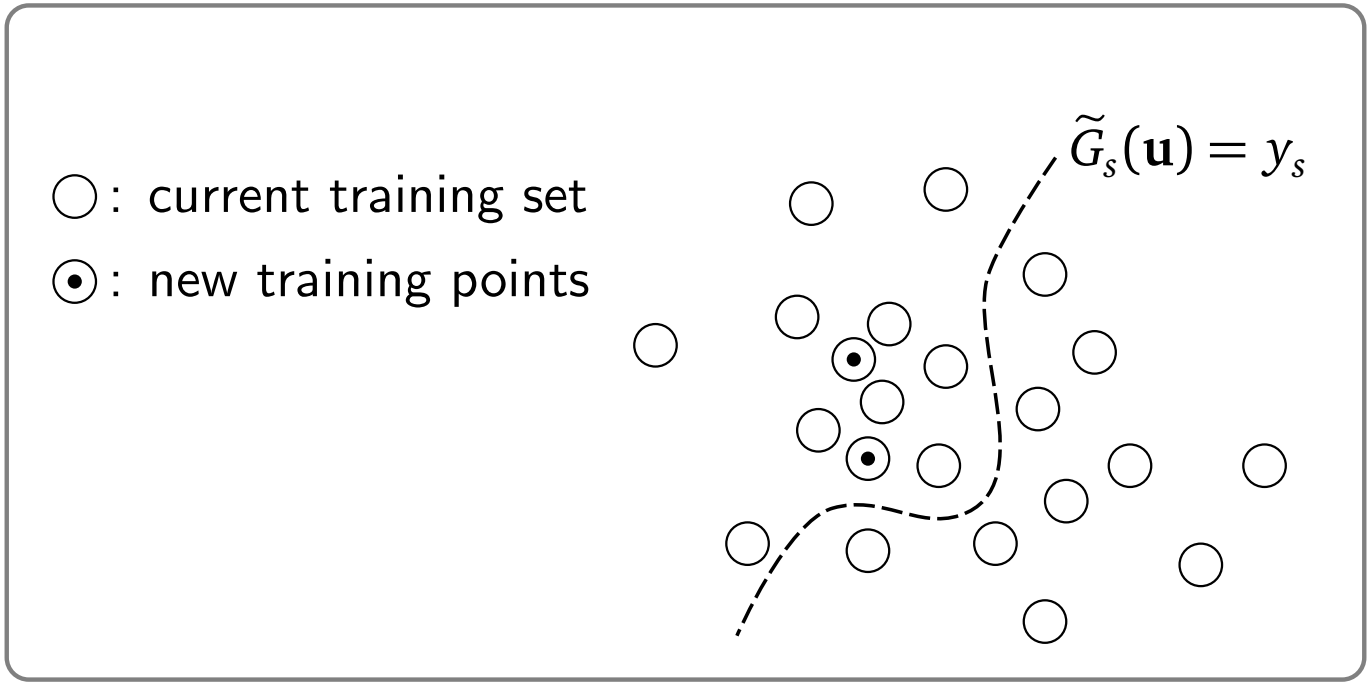
- Train initial SVR model $\widetilde{G}_1$

Training set $\mathcal{T}_s$ and LSF threshold value y_s (2/6)



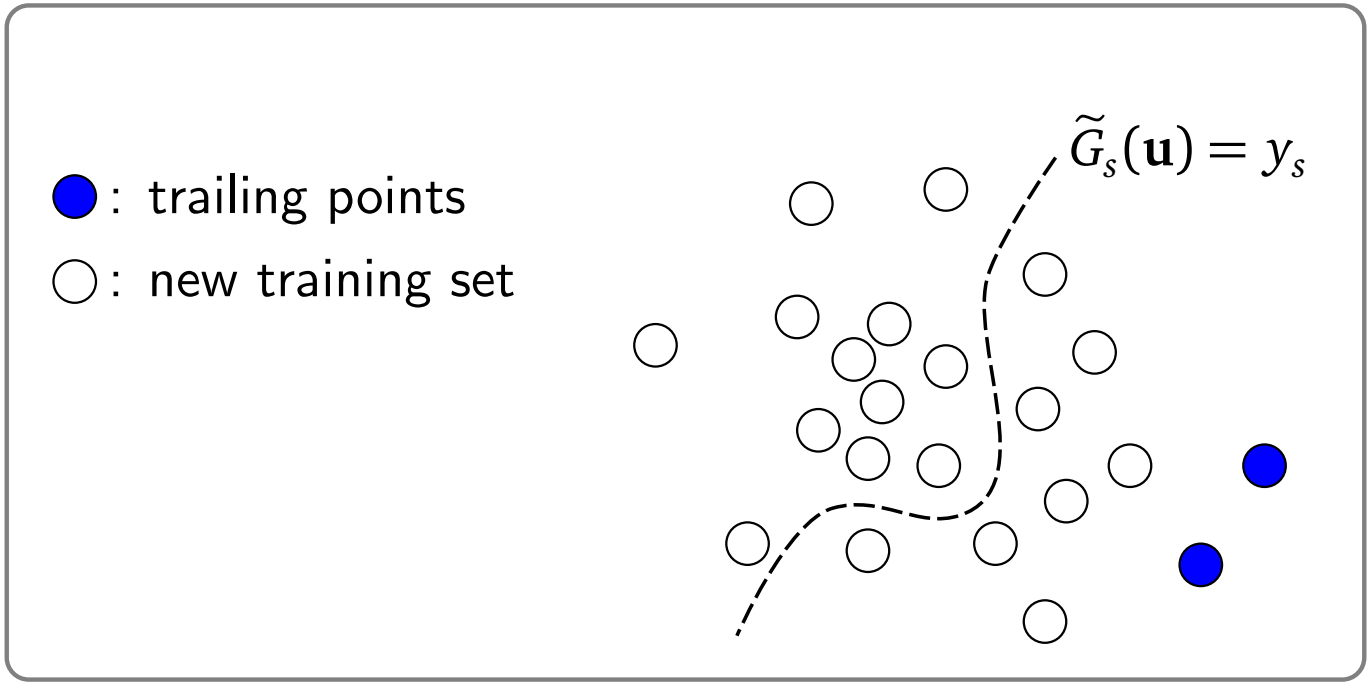
- Generate sample set $\mathcal{U}_s = \{\mathbf{u}^{(j)} \in \mathcal{U}_{SS;s}(m) : \widetilde{G}_s(\mathbf{u}^{(j)}) \leq y_s\}$
  Subset of m^{th} and last level samples $\mathcal{U}_{SS;s}(m)$ obtained by componentwise Metropolis algorithm (Au and Beck, 2001) applied to LSF $\widetilde{G}_s(\mathbf{u}) - y_s$
- Random selection of new training points in $\mathcal{U}_s$

Training set $\mathcal{T}_s$ and LSF threshold value y_s (3/6)



- New points added to the next training set $\mathcal{T}_{s+1}$

Training set $\mathcal{T}_s$ and LSF threshold value y_s (4/6)

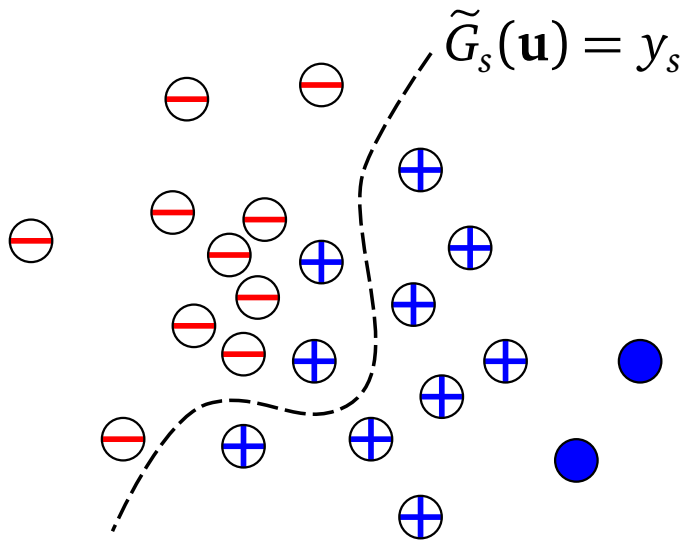


- Points with largest LSF values excluded from $\mathcal{T}_{s+1}$

Training set $\mathcal{T}_s$ and LSF threshold value y_s (5/6)

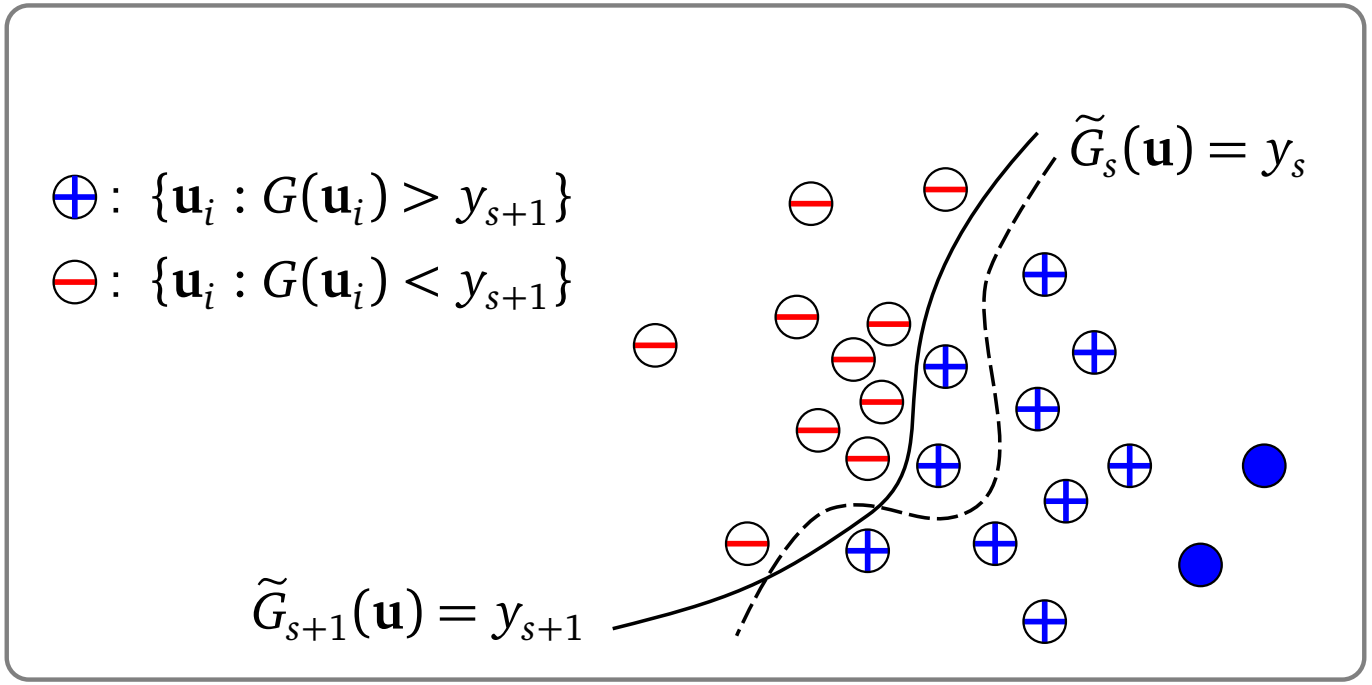
$\oplus$: $\{\mathbf{u}_i : G(\mathbf{u}_i) > y_{s+1}\}$

$\ominus$: $\{\mathbf{u}_i : G(\mathbf{u}_i) < y_{s+1}\}$



- Update of intermediate LSF level $y_{s+1} = \text{median}\left(G(\mathbf{u}_i)\,,\ i \in \mathcal{I}_{\text{train};s+1} \right)$

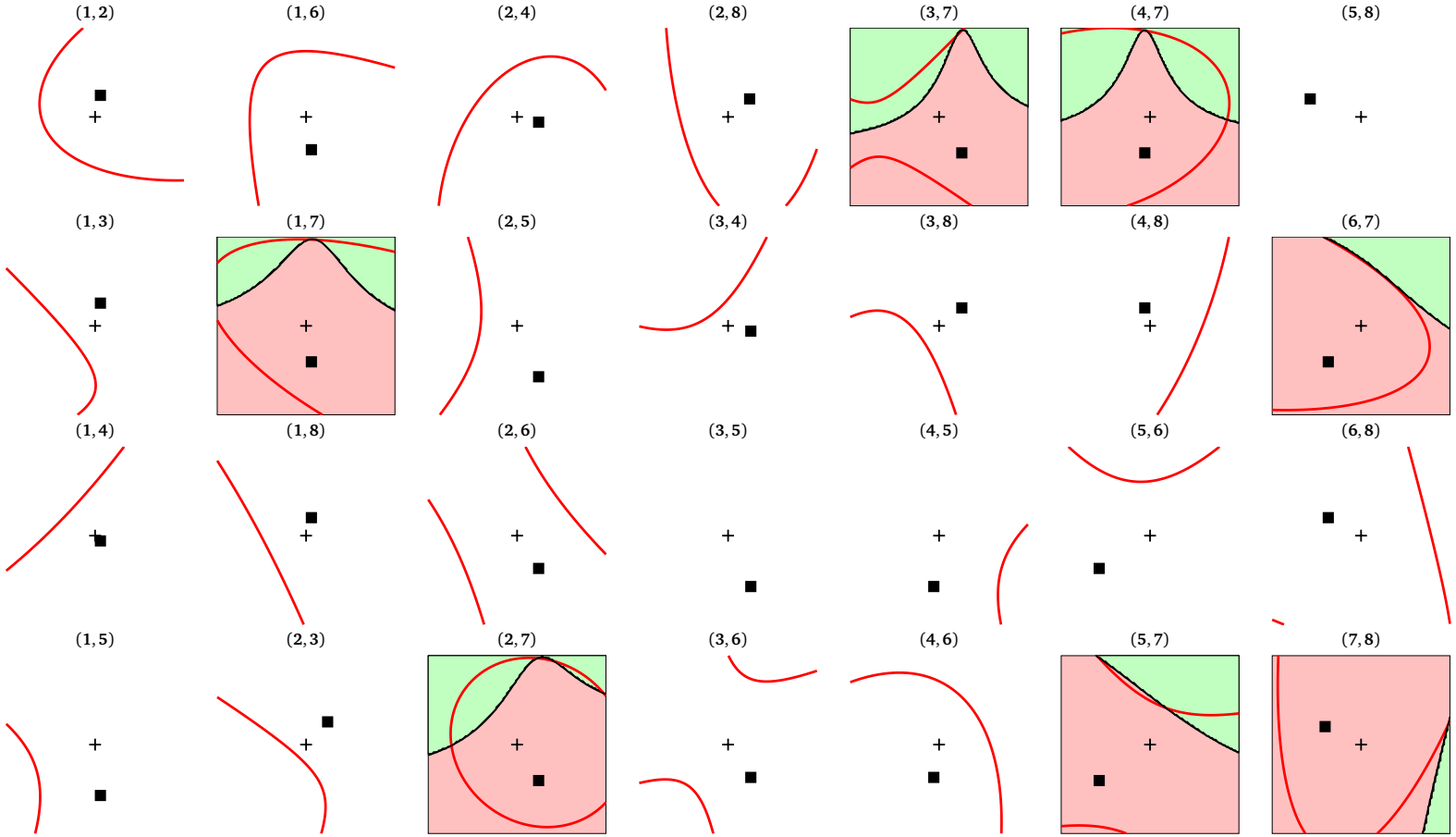
Training set $\mathcal{T}_s$ and LSF threshold value y_s (6/6)



- Train the updated SVR model $\widetilde{G}_{s+1}$

Application example #3 - Results (1/2)

- (P^*, u_i, u_j) **cross-cuts at MPFP in standard normal space**



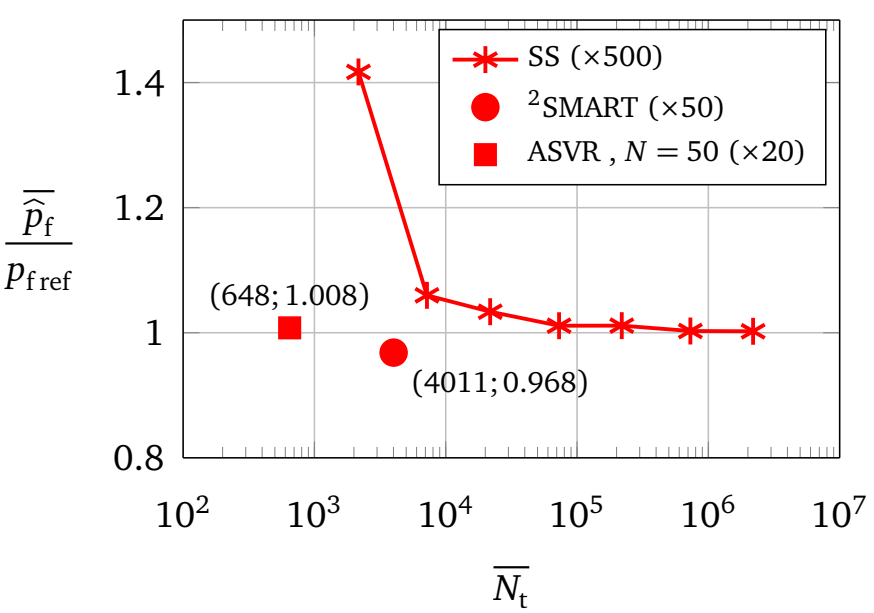
Application example #3 - Results (2/2)

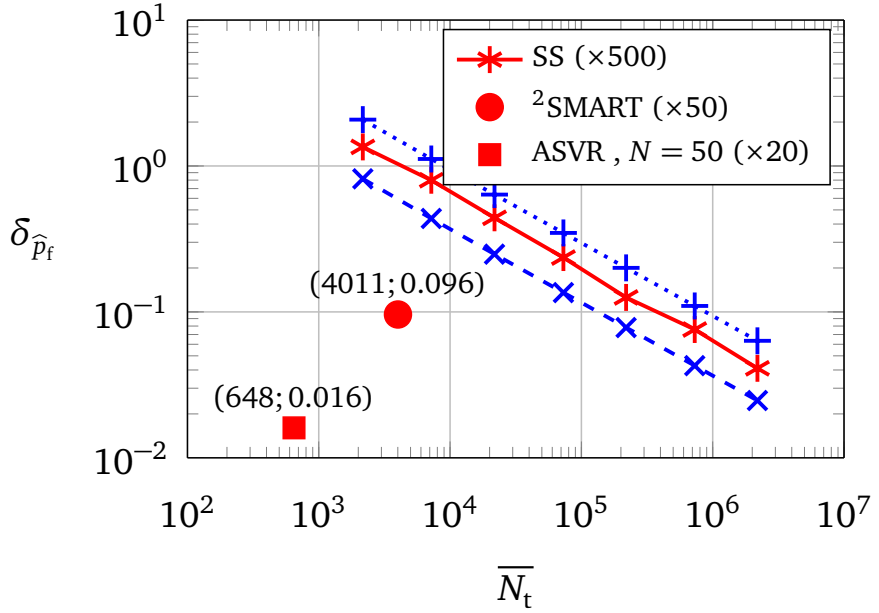
Remarks

Two sources of randomness for failure probability estimation:

- additional training points randomly selected from MCMC samples
- failure probability estimated by SS at each intermediate LSF level

- **Comparison of ASVR** (Bourinet, 2016) **with SS** (Au and Beck, 2001) **and** ^{2}**SMART** (Bourinet et al., 2011)





Application example #4 - Results (1/3)

Method	p_f estimate (#LSF calls)
MC (reference)	$\mathbf{2.24 \times 10^{-4}}$ **(10^9)**
FORM	3.18×10^{-3} (9,144)
SORM-cf	1.10×10^{-4} (77)(+9,144)
iASVR + ani-Matérn (Bourinet, 2019b)	2.16×10^{-4} (730)

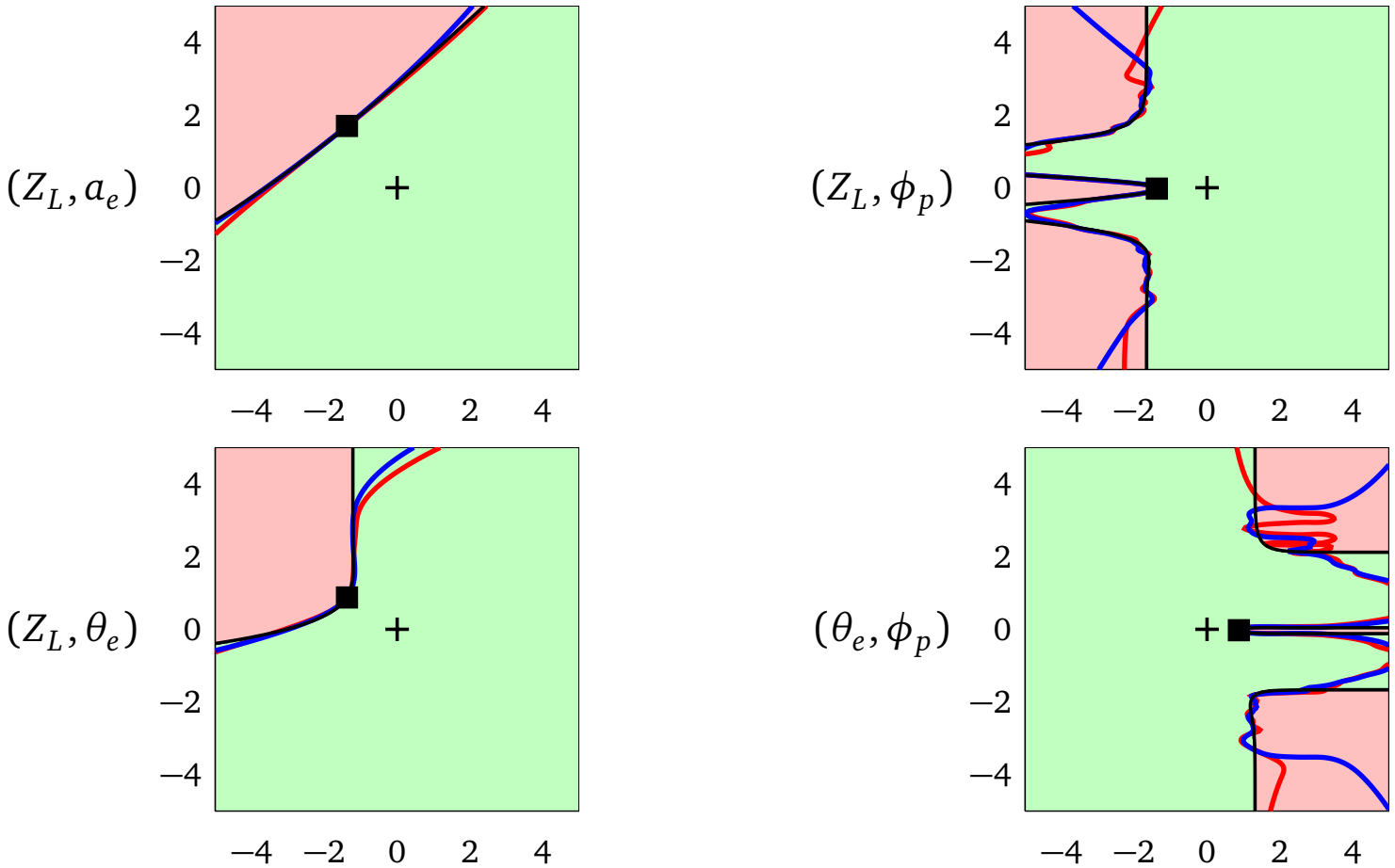
Comments

- Use of analytical expression of Matérn kernel for $\nu = m + 1/2$, $m \in \mathbb{N}$
- Good accuracy on p_f estimate (3.5% relative error)

Application example #4 - Results (2/3)

Comments

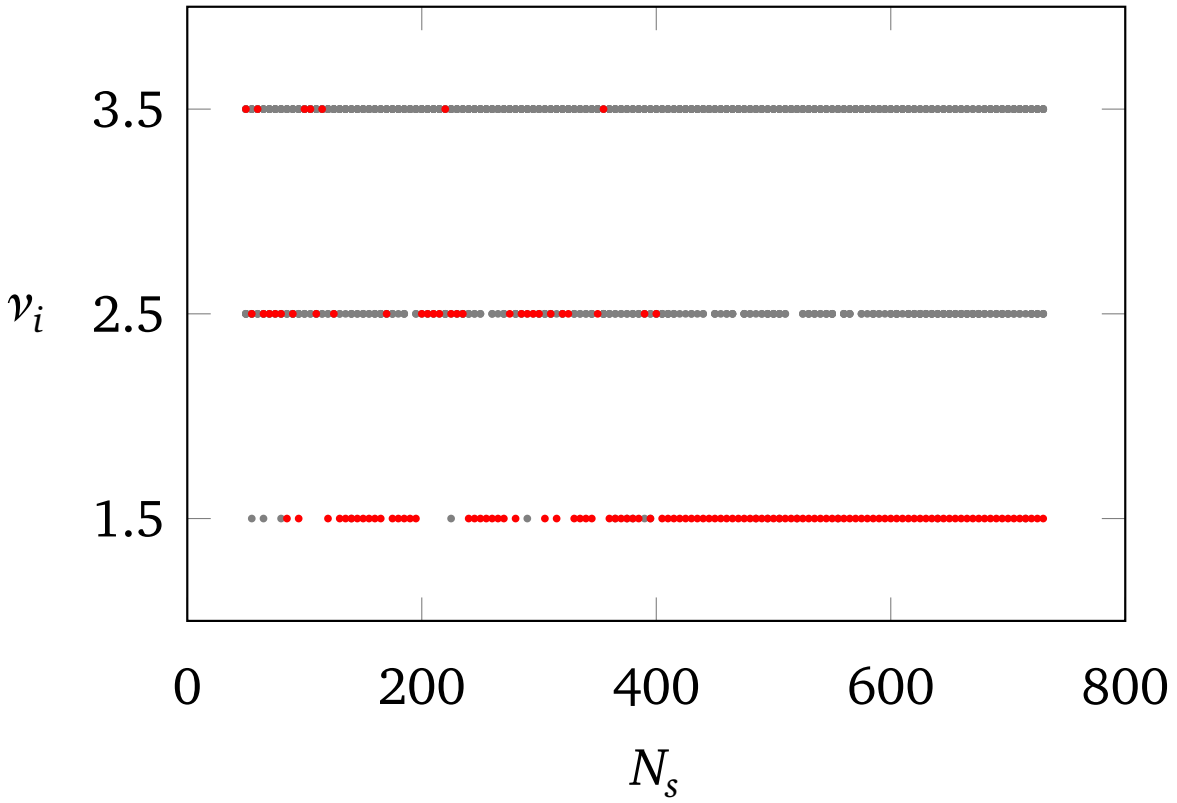
- SVR surrogate model able to fit the non-smooth LSS



Application example #4 - Results (3/3)

Comments

- Regularity parameter $\nu_9 \approx 3/2$ for $X_9 = \phi_p$, while other parameters ν_i for $i \neq 9$ are usually greater than $5/2$



ν_i of ϕ_p plotted in red

ν_i of all other inputs plotted in grey

Open questions and discussion (1/2)

1. Is there always a gain to apply a surrogate-based adaptive technique for RA?
   Trade-off between cost of LSF evaluation and training cost

2. Which type of surrogate model performs the best?
   Do you prefer kernel-based technique? Are you an unconditional fan of GP emulators (kriging) and its nice variance?
   Or you see PCE as the most suitable technique?
   Why would you use SVR?

3. For hyperparameter tuning, is it better to apply MLE or LOO-CV?
   How is it influenced by the distribution of points in the DOE?
   How is it influenced by the hyperparemeter search range?

4. Kernel choice: can we assume that the selected kernel works?
   Isotropic vs. anisotropic?
   Gaussian or Matérn? If Matérn, which ν or ν_i's?
   Can we switch to non-stationary kernel?

Open questions and discussion (1/2)

1. Is there always a gain to apply a surrogate-based adaptive technique for RA?
   Trade-off between cost of LSF evaluation and training cost

2. Which type of surrogate model performs the best?
   Do you prefer kernel-based technique? Are you an unconditional fan of GP emulators (kriging) and its nice variance?
   Or you see PCE as the most suitable technique?
   Why would you use SVR?

3. For hyperparameter tuning, is it better to apply MLE or LOO-CV?
   How is it influenced by the distribution of points in the DOE?
   How is it influenced by the hyperparemeter search range?

4. Kernel choice: can we assume that the selected kernel works?
   Isotropic vs. anisotropic?
   Gaussian or Matérn? If Matérn, which ν or ν_i's?
   Can we switch to non-stationary kernel?

Open questions and discussion (1/2)

1. Is there always a gain to apply a surrogate-based adaptive technique for RA?
   Trade-off between cost of LSF evaluation and training cost

2. Which type of surrogate model performs the best?
   Do you prefer kernel-based technique? Are you an unconditional fan of GP emulators (kriging) and its nice variance?
   Or you see PCE as the most suitable technique?
   Why would you use SVR?

3. For hyperparameter tuning, is it better to apply MLE or LOO-CV?
   How is it influenced by the distribution of points in the DOE?
   How is it influenced by the hyperparemeter search range?

4. Kernel choice: can we assume that the selected kernel works?
   Isotropic vs. anisotropic?
   Gaussian or Matérn? If Matérn, which ν or ν_i's?
   Can we switch to non-stationary kernel?

Open questions and discussion (1/2)

1. Is there always a gain to apply a surrogate-based adaptive technique for RA?
   Trade-off between cost of LSF evaluation and training cost

2. Which type of surrogate model performs the best?
   Do you prefer kernel-based technique? Are you an unconditional fan of GP emulators (kriging) and its nice variance?
   Or you see PCE as the most suitable technique?
   Why would you use SVR?

3. For hyperparameter tuning, is it better to apply MLE or LOO-CV?
   How is it influenced by the distribution of points in the DOE?
   How is it influenced by the hyperparemeter search range?

4. Kernel choice: can we assume that the selected kernel works?
   Isotropic vs. anisotropic?
   Gaussian or Matérn? If Matérn, which ν or ν_i's?
   Can we switch to non-stationary kernel?

Open questions and discussion (2/2)

5. Regularization: important or not? C (or $1/\tau$) to be tuned? Which range?
   Unregularized term: single scalar (ordinary kriging)? functional basis (universal kriging)?
   Regularizer: 2-norm (SVR)? 1-norm? Other?

6. Dealing with high dimensional inputs
   Should we reduce the dimension of the input space? Any pitfall?

Open questions and discussion (2/2)

5. Regularization: important or not? C (or $1/\tau$) to be tuned? Which range?
   Unregularized term: single scalar (ordinary kriging)? functional basis (universal kriging)?
   Regularizer: 2-norm (SVR)? 1-norm? Other?

6. Dealing with high dimensional inputs
   Should we reduce the dimension of the input space? Any pitfall?

Main references about RA with adaptive SVR surrogates

Bourinet J.-M. (2016). Rare-event probability estimation with adaptive support vector regression surrogates. *Reliability Engineering & System Safety*, 150, pp. 210–221.

Bourinet J.-M. (2017). Anisotropic-kernel-based support vector regression for reliability assessment. In: *Proc. 12th International Conference on Structural Safety and Reliability (ICOSSAR 2017), Vienna, Austria, August 6–10, 2017*. TU Verlag.

Bourinet J.-M. (2018). Reliability analysis and optimal design under uncertainty – Focus on adaptive surrogate-based approaches. HDR Report. Université Clermont Auvergne, France.

Bourinet J.-M. (2019a). Proposal of benchmark problems for surrogate-based reliability analysis. 3nd International Conference on Uncertainty Quantification in Computational Sciences and Engineering (UNCECOMP 2019), Crete, Greece, June 24–26, 2019.

Bourinet J.-M. (2019b). Reliability assessment by adaptive kernel-based surrogate models - Approximation of non-smooth limit-state functions. In: *Proc. 13th International Conference on Applications of Statistics and Probability in Civil Engineering (ICASP 13), Seoul, South Korea, May 26–30, 2019*. Ed. by J. Song. Seoul National University.

Thank you for your attention!

References I

Au S.-K., Beck J.L. (2001). Estimation of small failure probabilities in high dimensions by subset simulation. *Probabilistic Engineering Mechanics*, 16(4), pp. 263–277.

Bichon B.J., Eldred M.S., Swiler L.P., Mahadevan S., McFarland J.M. (2008). Efficient global reliability analysis for nonlinear implicit performance functions. *AIAA Journal*, 46(10), pp. 2459–2468.

Blatman G., Sudret B. (2010). An adaptive algorithm to build up sparse polynomial chaos expansions for stochastic finite element analysis. *Probabilistic Engineering Mechanics*, 25(2), pp. 183–197.

Bourinet J.-M. (2016). Rare-event probability estimation with adaptive support vector regression surrogates. *Reliability Engineering & System Safety*, 150, pp. 210–221.

Bourinet J.-M. (2017). Anisotropic-kernel-based support vector regression for reliability assessment. In: *Proc. 12th International Conference on Structural Safety and Reliability (ICOSSAR 2017), Vienna, Austria, August 6–10, 2017*. TU Verlag.

Bourinet J.-M. (2018). Reliability analysis and optimal design under uncertainty – Focus on adaptive surrogate-based approaches. HDR Report. Université Clermont Auvergne, France.

References II

Bourinet J.-M. (2019a). Proposal of benchmark problems for surrogate-based reliability analysis. 3nd International Conference on Uncertainty Quantification in Computational Sciences and Engineering (UNCECOMP 2019), Crete, Greece, June 24–26, 2019.

Bourinet J.-M. (2019b). Reliability assessment by adaptive kernel-based surrogate models - Approximation of non-smooth limit-state functions. In: *Proc. 13th International Conference on Applications of Statistics and Probability in Civil Engineering (ICASP 13), Seoul, South Korea, May 26–30, 2019*. Ed. by J. Song. Seoul National University.

Bourinet J.-M., Deheeger F., Lemaire M. (2011). Assessing small failure probabilities by combined subset simulation and support vector machines. *Structural Safety*, 33(6), pp. 343–353.

Chakraborty S., Chowdhury R. (2016). Sequential experimental design based generalised ANOVA. *Journal of Computational Physics*, 317, pp. 15–32.

Chang C.-C., Lin C.-J. (2011). LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2 (3). Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`, 27:1–27:27.

Chang M.-W., Lin C.-J. (2005). Leave-one-out bounds for support vector regression model selection. *Neural Computation*, 17(5), pp. 1188–1222.

References III

Dubourg V., Sudret B., Deheeger F. (2013). Metamodel-based importance sampling for structural reliability analysis. *Probabilistic Engineering Mechanics*, 33, pp. 47–57.

Echard B., Gayton N., Lemaire M. (2011). AK-MCS: An active learning reliability method combining kriging and Monte Carlo simulation. *Structural Safety*, 33(2), pp. 145–154.

Hansen N. (2016). The CMA evolution strategy: a tutorial. *ArXiv e-prints*.

Hu Z., Mahadevan S. (2016). Global sensitivity analysis-enhanced surrogate (GSAS) modeling for reliability analysis. *Structural and Multidisciplinary Optimization*, 53(3), pp. 501–521.

Kimeldorf G.S., Wahba G. (1970). A correspondence between Bayesian estimation on stochastic processes and smoothing by splines. *The Annals of Mathematical Statistics*, 41(2), pp. 495–502.

Kouassi A., Bourinet J.-M., Lalléchère S., Bonnet P., Fogli M. (2016). Reliability and sensitivity analysis of transmission lines in a probabilistic EMC context. *Electromagnetic Compatibility, IEEE Transactions on*, 58(2), pp. 561–572.

Lebrun R., Dutfoy A. (2009). A generalization of the Nataf transformation to distributions with elliptical copula. *Probabilistic Engineering Mechanics*, 24(2), pp. 172–178.

References IV

Liu P.-L., Der Kiureghian A. (1986). Multivariate distribution models with prescribed marginals and covariance. *Probabilistic Engineering Mechanics*, 1(2), pp. 105–112.

MOSEK ApS (2014). *MOSEK optimization toolbox - Version 7.0*.

Schölkopf B., Herbrich R., Smola A.J. (2001). A generalized representer theorem. In: *Computational Learning Theory*. Ed. by D. Helmbold, B. Williamson. Vol. 2111. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 416–426.

Schuëller G.I., Pradlwarter H.J. (2007). Benchmark study on reliability estimation in higher dimensions of structural systems – an overview. *Structural Safety*, 29(3). A Benchmark Study on Reliability in High Dimensions, pp. 167–182.

Scordelis A.C., Lo K.S. (1964). Computer analysis of cylindrical shells. *ACI Journal*, 61, pp. 539–561.

Sundar V., Shields M.D. (2016). Surrogate-enhanced stochastic search algorithms to identify implicitly defined functions for reliability analysis. *Structural Safety*, 62, pp. 1–11.

Suykens J.A.K., Van Gestel T., De Brabanter J., De Moor B., Vandewalle J. (2002). *Least squares support vector machines*. World Scientific Pub. Co.

The MathWorks (2012). *Matlab optimization toolbox user's guide - Version 6.2*.

References V

Tikhonov A.N., Arsenin V.Y. (1977). *Solution of ill-posed problems*. Ed. by D. W. H. Winston Washington. New York, NY, USA: John Wiley & Sons.

Wahba G. (1990). *Spline models for observational data*. Society for Industrial and Applied Mathematics.

Wills A. (2009). *QPC - Quadratic Programming in C - Version 2.0*.

Regularization (1/2)

Learning from a finite set of training data $\mathcal{T}$ is an ill-posed problem

- Standard regularization (Tikhonov and Arsenin, 1977; Wahba, 1990)

Impose some degree of smoothness on the approximate function that constrains the hypothesis space in which a solution is sought

Regularization (1/2)

Learning from a finite set of training data $\mathcal{T}$ is an ill-posed problem

- Standard regularization (Tikhonov and Arsenin, 1977; Wahba, 1990)

Impose some degree of smoothness on the approximate function that constrains the hypothesis space in which a solution is sought

In the context of a kernel k and its associated RKHS $\mathcal{H}_k$

$$\min_{h \in \mathcal{H}_k} \sum_{i=1}^{N} \ell\left(y_i, h\left(\mathbf{x}_i\right)\right) + \lambda \Omega\left(\|h\|_{\mathcal{H}_k}\right)$$

where $\ell : \mathcal{Y} \times \mathbb{R} \to \mathbb{R} \cup \{+\infty\}$ is a given loss function, $\Omega : \mathbb{R}_{\geq 0} \to \mathbb{R}$ is a strictly monotonic increasing function and $\lambda \in \mathbb{R}_{>0}$ a regularization parameter

Regularization (1/2)

Learning from a finite set of training data $\mathcal{T}$ is an ill-posed problem

- Standard regularization (Tikhonov and Arsenin, 1977; Wahba, 1990)

Impose some degree of smoothness on the approximate function that constrains the hypothesis space in which a solution is sought

In the context of a kernel k and its associated RKHS $\mathcal{H}_k$

$$\min_{h \in \mathcal{H}_k} \sum_{i=1}^{N} \ell\left(y_i, h\left(\mathbf{x}_i\right)\right) + \lambda\, \Omega\left(\|h\|_{\mathcal{H}_k}\right)$$

where $\ell : \mathcal{Y} \times \mathbb{R} \to \mathbb{R} \cup \{+\infty\}$ is a given loss function, $\Omega : \mathbb{R}_{\geq 0} \to \mathbb{R}$ is a strictly monotonic increasing function and $\lambda \in \mathbb{R}_{>0}$ a regularization parameter

First term: enforces the fit of solution h to the training data $\mathcal{T}$

Regularization (1/2)

Learning from a finite set of training data $\mathcal{T}$ is an ill-posed problem

► Standard regularization (Tikhonov and Arsenin, 1977; Wahba, 1990)

Impose some degree of smoothness on the approximate function that constrains the hypothesis space in which a solution is sought

In the context of a kernel k and its associated RKHS $\mathcal{H}_k$

$$\min_{h \in \mathcal{H}_k} \sum_{i=1}^{N} \ell\left(y_i, h\left(\mathbf{x}_i\right)\right) + \lambda\, \Omega\left(\|h\|_{\mathcal{H}_k}\right)$$

where $\ell : \mathcal{Y} \times \mathbb{R} \to \mathbb{R} \cup \{+\infty\}$ is a given loss function, $\Omega : \mathbb{R}_{\geq 0} \to \mathbb{R}$ is a strictly monotonic increasing function and $\lambda \in \mathbb{R}_{>0}$ a regularization parameter

First term: enforces the fit of solution h to the training data $\mathcal{T}$

Second term: enforces a sufficiently smooth solution h

Regularization (1/2)

Learning from a finite set of training data $\mathcal{T}$ is an ill-posed problem

- Standard regularization (Tikhonov and Arsenin, 1977; Wahba, 1990)

Impose some degree of smoothness on the approximate function that constrains the hypothesis space in which a solution is sought

In the context of a kernel k and its associated RKHS $\mathcal{H}_k$

$$\min_{h \in \mathcal{H}_k} \sum_{i=1}^{N} \ell\left(y_i, h\left(\mathbf{x}_i\right)\right) + \lambda \Omega\left(\|h\|_{\mathcal{H}_k}\right)$$

where $\ell : \mathcal{Y} \times \mathbb{R} \to \mathbb{R} \cup \{+\infty\}$ is a given loss function, $\Omega : \mathbb{R}_{\geq 0} \to \mathbb{R}$ is a strictly monotonic increasing function and $\lambda \in \mathbb{R}_{>0}$ a regularization parameter

First term: enforces the fit of solution h to the training data $\mathcal{T}$

Second term: enforces a sufficiently smooth solution h

Remark

ℓ and Ω convex $\Rightarrow$ dual optimization problem convex

- Unique global minimum (can be obtained by convex optimization methods)

Regularization (2/2)

Solution by virtue of the representer theorem (Kimeldorf and Wahba, 1970)

$$h(\mathbf{x}) = \sum_{i=1}^{N} c_i k(\mathbf{x}_i, \mathbf{x})$$

where $\mathbf{c} = (c_1, \ldots, c_N)^{\mathrm{T}} \in \mathbb{R}^N$ is the vector of unknown expansion coefficients

Regularization (2/2)

Solution by virtue of the representer theorem (Kimeldorf and Wahba, 1970)

$$h(\mathbf{x}) = \sum_{i=1}^{N} c_i k(\mathbf{x}_i, \mathbf{x})$$

where $\mathbf{c} = (c_1, \ldots, c_N)^{\mathrm{T}} \in \mathbb{R}^N$ is the vector of unknown expansion coefficients

Remarks

- The solution of an optimization problem in a potentially infinite dimensional space $\mathcal{H}_k$ can be expressed in terms of a linear combination of the kernel evaluated at the points of a finite dimensional set $(\mathbf{x}_1, \ldots, \mathbf{x}_N)$

Regularization (2/2)

Solution by virtue of the representer theorem (Kimeldorf and Wahba, 1970)

$$h(\mathbf{x}) = \sum_{i=1}^{N} c_i k(\mathbf{x}_i, \mathbf{x})$$

where $\mathbf{c} = (c_1, \ldots, c_N)^{\mathrm{T}} \in \mathbb{R}^N$ is the vector of unknown expansion coefficients

Remarks

- The solution of an optimization problem in a potentially infinite dimensional space $\mathcal{H}_k$ can be expressed in terms of a linear combination of the kernel evaluated at the points of a finite dimensional set $(\mathbf{x}_1, \ldots, \mathbf{x}_N)$
- Solution with a bias term by means of the semiparametric representer theorem (Schölkopf et al., 2001)

$$\widetilde{f}(\mathbf{x}) = h(\mathbf{x}) + p(\mathbf{x}) = \sum_{i=1}^{N} c_i k(\mathbf{x}_i, \mathbf{x}) + \sum_{j=1}^{M} d_j p_j(\mathbf{x})$$

where $p_j : \mathcal{X} \to \mathbb{R}$ for $1, \ldots, M$ is a set of M real-valued functions, $M \in \mathbb{N}$, and $d_1, \ldots, d_M \in \mathbb{R}$

Simple kriging: settings and primal formulation

Let $Y(\mathbf{x})$ be a square-integrable random process with zero mean

Objective

Construct a predictor $\widehat{Y}(\mathbf{x})$ of $Y(\mathbf{x})$ at a given point $\mathbf{x} \in \mathcal{X}$
$\widehat{Y}(\mathbf{x})$ chosen in the class of linear predictors

$$\widehat{Y}(\mathbf{x}) = \boldsymbol{\lambda}(\mathbf{x})^{\mathrm{T}} \mathbf{Y}$$

where $\mathbf{Y} = (Y(\mathbf{x}_1), \ldots, Y(\mathbf{x}_N))^{\mathrm{T}}$ and $\boldsymbol{\lambda}(\mathbf{x}) \in \mathbb{R}^N$ is a vector of unknown weights

Simple kriging: settings and primal formulation

Let $Y(\mathbf{x})$ be a square-integrable random process with zero mean

Objective

Construct a predictor $\widehat{Y}(\mathbf{x})$ of $Y(\mathbf{x})$ at a given point $\mathbf{x} \in \mathcal{X}$
$\widehat{Y}(\mathbf{x})$ chosen in the class of linear predictors

$$\widehat{Y}(\mathbf{x}) = \boldsymbol{\lambda}(\mathbf{x})^{\mathrm{T}} \mathbf{Y}$$

where $\mathbf{Y} = (Y(\mathbf{x}_1), \ldots, Y(\mathbf{x}_N))^{\mathrm{T}}$ and $\boldsymbol{\lambda}(\mathbf{x}) \in \mathbb{R}^N$ is a vector of unknown weights

The kriging predictor $\widehat{Y}(\mathbf{x})$ is unbiased

$$\mathbb{E}\left[\widehat{Y}(\mathbf{x})\right] = \boldsymbol{\lambda}(\mathbf{x})^{\mathrm{T}} \mathbb{E}\left[\mathbf{Y}\right] = \boldsymbol{\lambda}(\mathbf{x})^{\mathrm{T}} \mathbf{0} = 0 = \mathbb{E}\left[Y(\mathbf{x})\right]$$

Simple kriging: settings and primal formulation

Let $Y(\mathbf{x})$ be a square-integrable random process with zero mean

Objective

Construct a predictor $\widehat{Y}(\mathbf{x})$ of $Y(\mathbf{x})$ at a given point $\mathbf{x} \in \mathcal{X}$
$\widehat{Y}(\mathbf{x})$ chosen in the class of linear predictors

$$\widehat{Y}(\mathbf{x}) = \boldsymbol{\lambda}(\mathbf{x})^{\mathrm{T}} \mathbf{Y}$$

where $\mathbf{Y} = (Y(\mathbf{x}_1), \ldots, Y(\mathbf{x}_N))^{\mathrm{T}}$ and $\boldsymbol{\lambda}(\mathbf{x}) \in \mathbb{R}^N$ is a vector of unknown weights

The kriging predictor $\widehat{Y}(\mathbf{x})$ is unbiased

$$\mathbb{E}\left[\widehat{Y}(\mathbf{x})\right] = \boldsymbol{\lambda}(\mathbf{x})^{\mathrm{T}} \mathbb{E}\left[\mathbf{Y}\right] = \boldsymbol{\lambda}(\mathbf{x})^{\mathrm{T}} \mathbf{0} = 0 = \mathbb{E}\left[Y(\mathbf{x})\right]$$

Best predictor (mean square sense) obtained by minimizing $\mathrm{Var}\left[\widehat{Y}(\mathbf{x}) - Y(\mathbf{x})\right]$

$$\min_{\boldsymbol{\lambda}(\mathbf{x})} \boldsymbol{\lambda}(\mathbf{x})^{\mathrm{T}} \mathbf{K} \boldsymbol{\lambda}(\mathbf{x}) - 2\boldsymbol{\lambda}(\mathbf{x})^{\mathrm{T}} \mathbf{k}(\mathbf{x}) + k(\mathbf{x}, \mathbf{x})$$

where $\mathbf{K} = [k(\mathbf{x}_i, \mathbf{x}_j)]_{1 \leq i,j \leq N}$ is the covariance matrix of the random variables $\{Y(\mathbf{x}_i), 1 \leq i \leq N\}$ and $\mathbf{k}(\mathbf{x}) = (k(\mathbf{x}_1, \mathbf{x}), \ldots, k(\mathbf{x}_N, \mathbf{x}))^{\mathrm{T}}$

Simple kriging: settings and primal formulation

Let $Y(\mathbf{x})$ be a square-integrable random process with zero mean

Objective

Construct a predictor $\widehat{Y}(\mathbf{x})$ of $Y(\mathbf{x})$ at a given point $\mathbf{x} \in \mathcal{X}$
$\widehat{Y}(\mathbf{x})$ chosen in the class of linear predictors

$$\widehat{Y}(\mathbf{x}) = \boldsymbol{\lambda}(\mathbf{x})^{\mathrm{T}} \mathbf{Y}$$

where $\mathbf{Y} = (Y(\mathbf{x}_1), \ldots, Y(\mathbf{x}_N))^{\mathrm{T}}$ and $\boldsymbol{\lambda}(\mathbf{x}) \in \mathbb{R}^N$ is a vector of unknown weights

The kriging predictor $\widehat{Y}(\mathbf{x})$ is unbiased

$$\mathbb{E}\left[\widehat{Y}(\mathbf{x})\right] = \boldsymbol{\lambda}(\mathbf{x})^{\mathrm{T}} \mathbb{E}\left[\mathbf{Y}\right] = \boldsymbol{\lambda}(\mathbf{x})^{\mathrm{T}} \mathbf{0} = 0 = \mathbb{E}\left[Y(\mathbf{x})\right]$$

Best predictor (mean square sense) obtained by minimizing $\mathrm{Var}\left[\widehat{Y}(\mathbf{x}) - Y(\mathbf{x})\right]$

$$\min_{\boldsymbol{\lambda}(\mathbf{x})} \boldsymbol{\lambda}(\mathbf{x})^{\mathrm{T}} \mathbf{K} \boldsymbol{\lambda}(\mathbf{x}) - 2\boldsymbol{\lambda}(\mathbf{x})^{\mathrm{T}} \mathbf{k}(\mathbf{x}) + k(\mathbf{x}, \mathbf{x})$$

where $\mathbf{K} = [k(\mathbf{x}_i, \mathbf{x}_j)]_{1 \leq i,j \leq N}$ is the covariance matrix of the random variables $\{Y(\mathbf{x}_i), 1 \leq i \leq N\}$ and $\mathbf{k}(\mathbf{x}) = (k(\mathbf{x}_1, \mathbf{x}), \ldots, k(\mathbf{x}_N, \mathbf{x}))^{\mathrm{T}}$

Optimality condition

$$\mathbf{K} \boldsymbol{\lambda}(\mathbf{x}) = \mathbf{k}(\mathbf{x})$$

Simple kriging: dual formulation

Why a dual formulation?

We do not want to solve $\mathbf{K}\boldsymbol{\lambda}(\mathbf{x}) = \mathbf{k}(\mathbf{x})$ for each point $\mathbf{x}$ where the prediction needs to be defined

Simple kriging: dual formulation

Why a dual formulation?

We do not want to solve $\mathbf{K}\boldsymbol{\lambda}(\mathbf{x}) = \mathbf{k}(\mathbf{x})$ for each point $\mathbf{x}$ where the prediction needs to be defined

$\widehat{Y}(\mathbf{x}) = \boldsymbol{\lambda}(\mathbf{x})^{\mathrm{T}}\mathbf{Y}$

Simple kriging: dual formulation

Why a dual formulation?

We do not want to solve $\mathbf{K}\boldsymbol{\lambda}(\mathbf{x}) = \mathbf{k}(\mathbf{x})$ for each point $\mathbf{x}$ where the prediction needs to be defined

$\widehat{Y}(\mathbf{x}) = \boldsymbol{\lambda}(\mathbf{x})^{\mathrm{T}}\mathbf{Y} \quad = \left(\mathbf{K}^{-1}\mathbf{k}(\mathbf{x})\right)^{\mathrm{T}}\mathbf{Y} \quad \mathbf{K}$ assumed full rank

Simple kriging: dual formulation

Why a dual formulation?

We do not want to solve $\mathbf{K}\boldsymbol{\lambda}(\mathbf{x}) = \mathbf{k}(\mathbf{x})$ for each point $\mathbf{x}$ where the prediction needs to be defined

$$\begin{aligned}
\widehat{Y}(\mathbf{x}) = \boldsymbol{\lambda}(\mathbf{x})^{\mathrm{T}}\mathbf{Y} &= \left(\mathbf{K}^{-1}\mathbf{k}(\mathbf{x})\right)^{\mathrm{T}}\mathbf{Y} && \mathbf{K} \text{ assumed full rank} \\
&= \mathbf{k}(\mathbf{x})^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{Y} && \mathbf{K} \text{ symmetric and therefore its inverse}
\end{aligned}$$

Simple kriging: dual formulation

Why a dual formulation?

We do not want to solve $\mathbf{K}\boldsymbol{\lambda}(\mathbf{x}) = \mathbf{k}(\mathbf{x})$ for each point $\mathbf{x}$ where the prediction needs to be defined

$$
\begin{aligned}
\widehat{Y}(\mathbf{x}) = \boldsymbol{\lambda}(\mathbf{x})^{\mathrm{T}}\mathbf{Y} &= \left(\mathbf{K}^{-1}\mathbf{k}(\mathbf{x})\right)^{\mathrm{T}}\mathbf{Y} && \mathbf{K} \text{ assumed full rank} \\
&= \mathbf{k}(\mathbf{x})^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{Y} && \mathbf{K} \text{ symmetric and therefore its inverse} \\
&= \mathbf{k}(\mathbf{x})^{\mathrm{T}}\mathbf{C}
\end{aligned}
$$

where the random vector $\mathbf{C} = (C_1, \ldots, C_N)^{\mathrm{T}}$ is the solution of

$$\mathbf{K}\mathbf{C} = \mathbf{Y}$$

Simple kriging: dual formulation

Why a dual formulation?

We do not want to solve $\mathbf{K}\boldsymbol{\lambda}(\mathbf{x}) = \mathbf{k}(\mathbf{x})$ for each point $\mathbf{x}$ where the prediction needs to be defined

$$\begin{aligned}
\widehat{Y}(\mathbf{x}) = \boldsymbol{\lambda}(\mathbf{x})^{\mathrm{T}}\mathbf{Y} &= \left(\mathbf{K}^{-1}\mathbf{k}(\mathbf{x})\right)^{\mathrm{T}}\mathbf{Y} && \mathbf{K} \text{ assumed full rank} \\
&= \mathbf{k}(\mathbf{x})^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{Y} && \mathbf{K} \text{ symmetric and therefore its inverse} \\
&= \mathbf{k}(\mathbf{x})^{\mathrm{T}}\mathbf{C}
\end{aligned}$$

where the random vector $\mathbf{C} = (C_1, \ldots, C_N)^{\mathrm{T}}$ is the solution of

$$\mathbf{K}\mathbf{C} = \mathbf{Y}$$

Solved only once for the available realization $\mathbf{y} = (y_1, \ldots, y_N)^{\mathrm{T}}$ of $\mathbf{Y} = (Y(\mathbf{x}_1), \ldots, Y(\mathbf{x}_N))^{\mathrm{T}}$

$$\mathbf{K}\mathbf{c} = \mathbf{y}$$

where $\mathbf{c} = (c_1, \ldots, c_N)^{\mathrm{T}} \in \mathbb{R}^N$

Simple kriging: dual formulation

Why a dual formulation?

We do not want to solve $\mathbf{K}\boldsymbol{\lambda}(\mathbf{x}) = \mathbf{k}(\mathbf{x})$ for each point $\mathbf{x}$ where the prediction needs to be defined

$$\begin{aligned}
\widehat{Y}(\mathbf{x}) = \boldsymbol{\lambda}(\mathbf{x})^{\mathrm{T}}\mathbf{Y} &= \left(\mathbf{K}^{-1}\mathbf{k}(\mathbf{x})\right)^{\mathrm{T}}\mathbf{Y} && \mathbf{K} \text{ assumed full rank} \\
&= \mathbf{k}(\mathbf{x})^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{Y} && \mathbf{K} \text{ symmetric and therefore its inverse} \\
&= \mathbf{k}(\mathbf{x})^{\mathrm{T}}\mathbf{C}
\end{aligned}$$

where the random vector $\mathbf{C} = (C_1, \ldots, C_N)^{\mathrm{T}}$ is the solution of

$$\mathbf{K}\mathbf{C} = \mathbf{Y}$$

Solved only once for the available realization $\mathbf{y} = (y_1, \ldots, y_N)^{\mathrm{T}}$ of $\mathbf{Y} = (Y(\mathbf{x}_1), \ldots, Y(\mathbf{x}_N))^{\mathrm{T}}$

$$\mathbf{K}\mathbf{c} = \mathbf{y}$$

where $\mathbf{c} = (c_1, \ldots, c_N)^{\mathrm{T}} \in \mathbb{R}^N$

Kriging mean prediction at location $\mathbf{x}$

$$\mu_{\mathrm{SK}}(\mathbf{x}) = \mathbf{k}(\mathbf{x})^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y} = \mathbf{k}(\mathbf{x})^{\mathrm{T}}\mathbf{c} = \sum_{i=1}^{N} c_i k(\mathbf{x}_i, \mathbf{x})$$

Noisy kriging with a trend

Hypotheses of noisy kriging with a trend

✓ Noisy observations of the true outputs $y(\mathbf{x}_i)$

$$y_i = y(\mathbf{x}_i) + \epsilon_i$$

where ϵ_i are zero-mean random variables with variances $\tau_1^2 = \ldots = \tau_N^2 = \tau^2$ (homogeneous level of noise)

Hypotheses of noisy kriging with a trend

✓ Noisy observations of the true outputs $y(\mathbf{x}_i)$

$$y_i = y(\mathbf{x}_i) + \epsilon_i$$

where ϵ_i are zero-mean random variables with variances $\tau_1^2 = \ldots = \tau_N^2 = \tau^2$ (homogeneous level of noise)

✓ $Y(\mathbf{x})$ considered as the sum of a drift and a residual

$$Y(\mathbf{x}) = \mu(\mathbf{x}) + Z(\mathbf{x})$$

where $Z(\mathbf{x})$ is a zero-mean square-integrable random process and where

$$\mu(\mathbf{x}) = \sum_{j=1}^{M} d_j p_j(\mathbf{x}) = \mathbf{p}(\mathbf{x})^{\mathrm{T}} \mathbf{d}$$

$\mathbf{p}(\mathbf{x}) = (p_1(\mathbf{x}), \ldots, p_M(\mathbf{x}))^{\mathrm{T}}$ is a given basis of M real-valued functions, where $p_j : \mathcal{X} \to \mathbb{R}$ for $1, \ldots, M$, and $\mathbf{d} = (d_1, \ldots, d_M)^{\mathrm{T}} \in \mathbb{R}^M$ is a vector of unknown coefficients

Noisy kriging with a trend

- Optimality condition

$$\begin{bmatrix} \mathbf{K} + \tau^2 \mathbf{1} & \mathbf{P} \\ \mathbf{P}^{\mathrm{T}} & \mathbf{0} \end{bmatrix} \begin{pmatrix} \mathbf{c} \\ \mathbf{d} \end{pmatrix} = \begin{pmatrix} \mathbf{y} \\ \mathbf{0} \end{pmatrix}$$

- Kriging mean

$$\begin{aligned} \widetilde{y}(\mathbf{x}) &= \mu_{\mathsf{UK+noise}}(\mathbf{x}) \\ &= \mathbf{k}(\mathbf{x})^{\mathrm{T}} \mathbf{c} + \mathbf{p}(\mathbf{x})^{\mathrm{T}} \mathbf{d} \\ &= \sum_{i=1}^{N} c_i k(\mathbf{x}_i, \mathbf{x}) + \sum_{j=1}^{M} d_j p_j(\mathbf{x}) \end{aligned}$$

Noisy kriging with a trend vs. LS-SVR

Noisy kriging with a trend

- Optimality condition

$$\begin{bmatrix} \mathbf{K} + \tau^2 \mathbf{1} & \mathbf{P} \\ \mathbf{P}^{\mathrm{T}} & \mathbf{0} \end{bmatrix} \begin{pmatrix} \mathbf{c} \\ \mathbf{d} \end{pmatrix} = \begin{pmatrix} \mathbf{y} \\ \mathbf{0} \end{pmatrix}$$

- Kriging mean

$$\begin{aligned} \widetilde{y}(\mathbf{x}) &= \mu_{\mathsf{UK+noise}}(\mathbf{x}) \\ &= \mathbf{k}(\mathbf{x})^{\mathrm{T}} \mathbf{c} + \mathbf{p}(\mathbf{x})^{\mathrm{T}} \mathbf{d} \\ &= \sum_{i=1}^{N} c_i k(\mathbf{x}_i, \mathbf{x}) + \sum_{j=1}^{M} d_j p_j(\mathbf{x}) \end{aligned}$$

LS-SVR

- Dual optimization problem

$$\begin{bmatrix} \mathbf{K} + C^{-1} \mathbf{I} & \mathbf{1} \\ \mathbf{1}^{\mathrm{T}} & 0 \end{bmatrix} \begin{pmatrix} \boldsymbol{\alpha} \\ b \end{pmatrix} = \begin{pmatrix} \mathbf{y} \\ 0 \end{pmatrix}$$

- Decision function

$$\begin{aligned} \widetilde{y}(\mathbf{x}) &= \widetilde{f}(\mathbf{x}) \\ &= \mathbf{k}(\mathbf{x})^{\mathrm{T}} \boldsymbol{\alpha} + b \\ &= \sum_{i=1}^{N} \alpha_i k(\mathbf{x}_i, \mathbf{x}) + b \end{aligned}$$

Noisy kriging with a trend vs. LS-SVR

Noisy kriging with a trend

- Optimality condition

$$\begin{bmatrix} \mathbf{K} + \tau^2 \mathbf{1} & \mathbf{P} \\ \mathbf{P}^{\mathrm{T}} & \mathbf{0} \end{bmatrix} \begin{pmatrix} \mathbf{c} \\ \mathbf{d} \end{pmatrix} = \begin{pmatrix} \mathbf{y} \\ \mathbf{0} \end{pmatrix}$$

- Kriging mean

$$\begin{aligned} \widetilde{y}(\mathbf{x}) &= \mu_{\mathsf{UK+noise}}(\mathbf{x}) \\ &= \mathbf{k}(\mathbf{x})^{\mathrm{T}} \mathbf{c} + \mathbf{p}(\mathbf{x})^{\mathrm{T}} \mathbf{d} \\ &= \sum_{i=1}^{N} c_i k(\mathbf{x}_i, \mathbf{x}) + \sum_{j=1}^{M} d_j p_j(\mathbf{x}) \end{aligned}$$

LS-SVR

- Dual optimization problem

$$\begin{bmatrix} \mathbf{K} + C^{-1} \mathbf{I} & \mathbf{1} \\ \mathbf{1}^{\mathrm{T}} & 0 \end{bmatrix} \begin{pmatrix} \boldsymbol{\alpha} \\ b \end{pmatrix} = \begin{pmatrix} \mathbf{y} \\ 0 \end{pmatrix}$$

- Decision function

$$\begin{aligned} \widetilde{y}(\mathbf{x}) &= \widetilde{f}(\mathbf{x}) \\ &= \mathbf{k}(\mathbf{x})^{\mathrm{T}} \boldsymbol{\alpha} + b \\ &= \sum_{i=1}^{N} \alpha_i k(\mathbf{x}_i, \mathbf{x}) + b \end{aligned}$$

Remarks (Bourinet, 2018)

- LS-SVR with bias term b fully identical to ordinary kriging with noise ($M = 1$, with $p_1 : \mathcal{X} \to \mathbb{R}, \mathbf{x} \mapsto 1$). τ^2 and C^{-1} play the same role
- Remember that LS-SVR is a subcase of L2-ϵ-SVR with ϵ set to zero