Support Vector Machine for Efficient Subset Simulations: ^{2}SMART method

F. Deheeger & M. Lemaire

Université Blaise Pascal
Institut Français de Mécanique Avancée

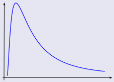
10th International Conference on Applications of Statistics and Probability in Civil Engineering

Context: Coupling mechanical and stochastic models

Our objective: the reliability

aims to evaluate the performance of mechanical systems under data uncertainty

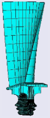
Stochastic model



- Geometry
- Loads
- Materials

Mechanical model

- FE model
- ...

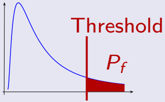


X

Performance function $G(X)$

Mechanical
and
Stochastic model

Reliability analysis



$$P_f = \text{Prob}\{G(X) < 0\}$$

Outlook of the work

Objective

"to make efficient" the coupled approach for application to industrial problems

Implicit performance function

Complex behavior
& large number of variables

Ways:

- To decrease the number of calls to the mechanical model
- To improve calculation convergence

The objective is to get the maximum of information
with a minimum of mechanical model evaluations

→ Limit-state substitution and statistical learning

Contents

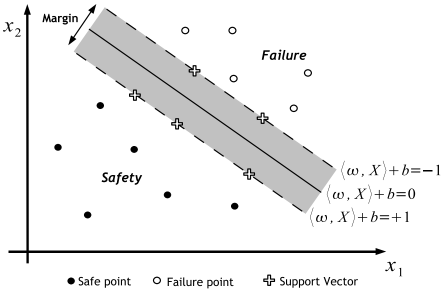
Contents

A classification problem

Binary classification in linearly separable case

- A training set: labeled points $X_1, \ldots, X_n$ defined in a domain D
- 2 classes (binary classification): in reliability, the failure domain and the safety domain
- Objective: to find a hyperplane which is the optimal data classifier (linearly separable classes)

$$G(X) = \langle \varpi, X \rangle + b = 0$$

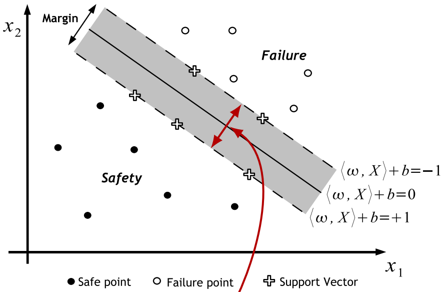


A classification problem

Binary classification in linearly separable case

- A training set: labeled points $X_1, \dots, X_n$ defined in a domain D
- 2 classes (binary classification): in reliability, the failure domain and the safety domain
- Objective: to find a hyperplane which is the optimal data classifier (linearly separable classes)

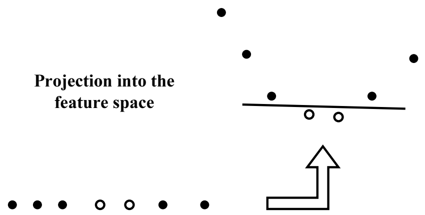
$$G(X) = \langle \varpi, X \rangle + b = 0$$



The optimal classifier is obtained by maximizing the margin

The non linear case

For non linear cases, a space transformation is applied:



A non linear projector Φ transforms the starting space towards a space of higher size: the feature space

The non linear classification in the standard space becomes a linear problem after the projection

→ The Kernel Trick

The optimization problem

How to maximize the margin ?

After mathematical manipulations, maximizing the margin corresponds to the minimization of the norm of ϖ, the optimization problem is written in the form:

$$\min \frac{\|\varpi\|^2}{2} \text{ subject to } \underline{c_i(\langle \varpi, X_i \rangle + b) \geq 1, \; i = 1, \ldots, n}$$

Constraints of right classification

The Lagragian formulation:

$$\boxed{L(\varpi, b, \alpha) = \frac{\|\varpi\|^2}{2} - \sum_{i=1}^{n} \alpha_i \left[c_i(\langle \varpi, X_i \rangle + b) - 1\right]}$$

Convex quadratic function

The Karush-Kuhn-Tucker conditions:

$$\frac{\partial L(\varpi, b, \alpha)}{\partial b} = 0 = \sum_{i=1}^{n} \alpha_i \, c_i \text{ and } \frac{\partial L(\varpi, b, \alpha)}{\partial \varpi} = 0 = \varpi - \sum_{i=1}^{n} \alpha_i \, c_i \, X_i$$

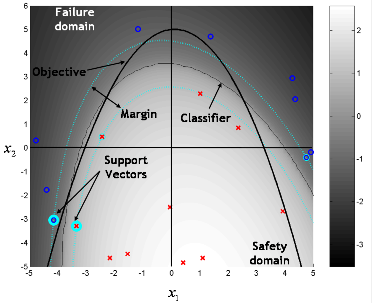
The optimization problem

The nonnull α_i correspond to realizations on the margin, they are called the support vectors

The separation can be defined only starting from these realizations:

$$\varpi = \sum_{j=1}^{S} \alpha_j C_j X_j$$

where S is the number of supports vectors.



Then we obtain the main properties for our use of SVM:
only points in the margin are useful to affine the separation

The **SMART** method

SVM, Classification and Monte-Carlo

[J.E.Hurtado, Structural reliability - Statistical Learning Perspectives, 2004, Springer editions]

[F.Deheeger and M.Lemaire, Reliability Analysis by Support Vector Machine Classification, 2006, ASRANet Colloquium Glasgow]

SMART

- means: Support-vector Margin Algorithm for Reliability esTimation
- is specially design for the determination of the limit between failure and safety domains for the estimation of failure probability

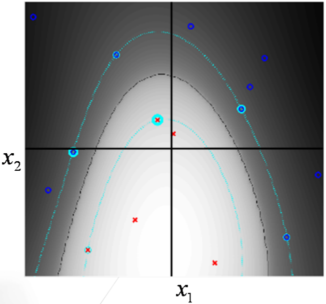
The idea: analyze Monte-Carlo simulation like a classification problem

The principle is to classify all points of a Monte-Carlo simulation by calling the limit-state function for a minimum of realizations.

SMART steps

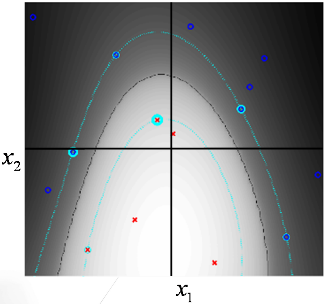
1. **Work in standard space**
   Transformation of basic variables into the standard space
2. **Design of experiment**
3. **First classifier**
4. **Generation of the work population**
5. **Learning points selection**
6. **A new classifier**
7. **The approximation step**
8. **The precision step**

SMART steps



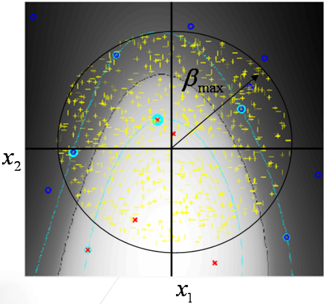
1. **Work in standard space**
2. **Design of experiment**
   The first plan is designed by latin square sampling (or from the expert knowledge)
3. **First classifier**
4. **Generation of the work population**
5. **Learning points selection**
6. **A new classifier**
7. **The approximation step**
8. **The precision step**

SMART steps



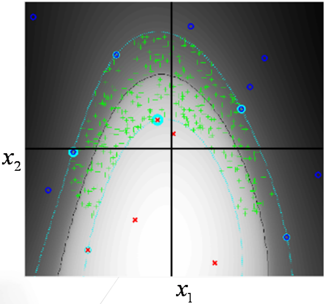
1. **Work in standard space**
2. **Design of experiment**
3. **First classifier**
   Evaluation of the limit-state for the first points, and optimization of the first classifier
4. **Generation of the work population**
5. **Learning points selection**
6. **A new classifier**
7. **The approximation step**
8. **The precision step**

SMART steps



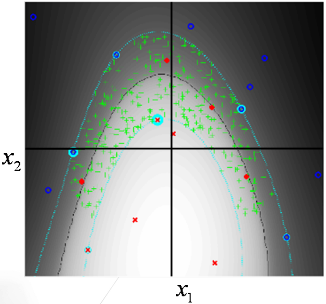
1. **Work in standard space**
2. **Design of experiment**
3. **First classifier**
4. **Generation of the work population**
   Generation of an uniform distribution in a hypersphere of radius β_{max}
5. **Learning points selection**
6. **A new classifier**
7. **The approximation step**
8. **The precision step**

SMART steps



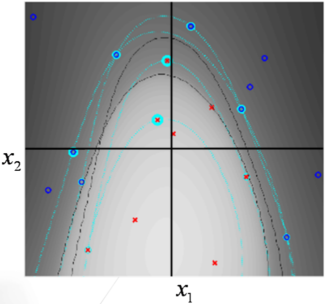
1. Work in standard space
2. Design of experiment
3. First classifier
4. Generation of the work population
5. Learning points selection
   Selection of margin points from the work population, and compression by clustering, as to keep a few optimal points
6. A new classifier
7. The approximation step
8. The precision step

SMART steps



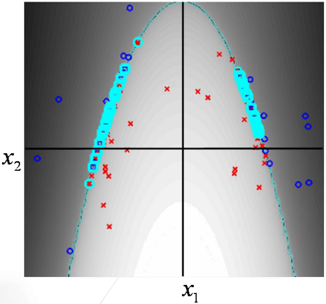
1. Work in standard space
2. Design of experiment
3. First classifier
4. Generation of the work population
5. Learning points selection
6. A new classifier
   Evaluation of the limit-state function for the selected points and optimization of a new classifier
7. The approximation step
8. The precision step

SMART steps



1. **Work in standard space**
2. **Design of experiment**
3. **First classifier**
4. **Generation of the work population**
5. **Learning points selection**
6. **A new classifier**
7. **The approximation step**
   Back to step 5 until an empty margin, or a maximum number of learning iterations
8. **The precision step**

SMART steps



1. Work in standard space
2. Design of experiment
3. First classifier
4. Generation of the work population
5. Learning points selection
6. A new classifier
7. The approximation step
8. The precision step
   Back to step 4 with a Gaussian distribution of the size of the Monte-Carlo simulation as work population

SMART: key points of the learning process

First design plan

A compromise between number of learning points and space exploration: latin square design

Multi-scale approach

3 steps are defined for the learning process:

1. the positioning step: the work population is uniform and sparse
2. the stabilization step: the work population is still uniform but denser
3. the precision step: the work population is the Monte-Carlo simulation

Margin points selection

New points added to the database are selected from work population in the margin:

- clustered points: a good dispersion along the margin
- instable points: points whose class are changing between iterations
- closest points: points close to the analytical separation

SMART: some limits

Dependence on the probability objective

- The size of the Monte-Carlo population depends on the failure probability that is not a priori define
- A crucial 1st step: the size of the first design plan depends on the failure probability objective because the algorithm needs failure points to start

So: need a less dependent approach...

$\longrightarrow$ $\boxed{\text{subset simulation}}$

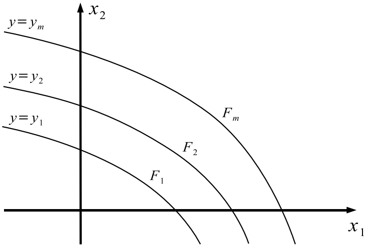
Contents

Subset simulation - The idea

[S.K.Au and J.L.Beck, Estimation of small failure probabilities in high dimensions by subset simulation, 2001, Prob.Eng.Mech.]

The desired probability $P(F_m) = P(F)$ is written by using the conditional probability theory:

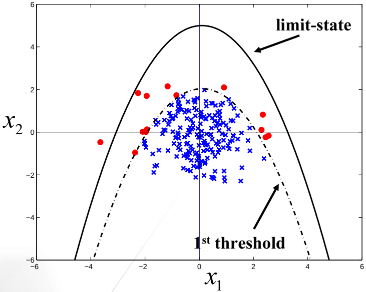
$$\begin{aligned} P_f &= P(F) \\ &= P(F \mid F_{m-1})P(F_{m-1}) \\ &= \ldots \\ &= P(F_1)\prod_{i=2}^{m} P(F_i \mid F_{i-1}) \end{aligned}$$



P_f is evaluated by estimating all factors.

Successive thresholds are automatically selected such that the evaluated conditional probabilities are about $\alpha = 10\%$:
each factor can be evaluated efficiently by simulation

Subset simulation steps

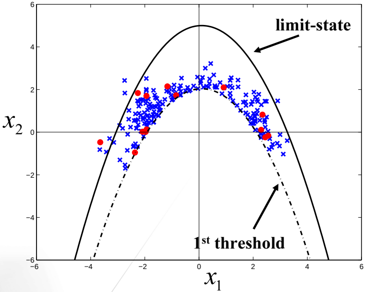


Subset first step

The first probability $P(F_1)$ is evaluated by crude Monte-Carlo

- N simulations are generated
- the first threshold y_1 is defined as to find $P(F_1) \approx \alpha = 0.1$

Subset simulation steps



Subset first step

The first probability $P(F_1)$ is evaluated by crude Monte-Carlo

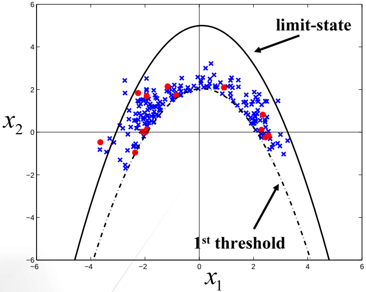
- N simulations are generated
- the first threshold y_1 is defined as to find $P(F_1) \approx \alpha = 0.1$

Subset second step

The second probability $P(F_2)$ is evaluated by conditional simulations

- N conditional simulations are generated by Markov Chains from germs: points of the preceding step in F_1
- the second threshold y_2 is defined as to find $P(F_2) \approx \alpha$

Subset simulation steps



Subset first step

The first probability $P(F_1)$ is evaluated by crude Monte-Carlo

- N simulations are generated
- the first threshold y_1 is defined as to find $P(F_1) \approx \alpha = 0.1$

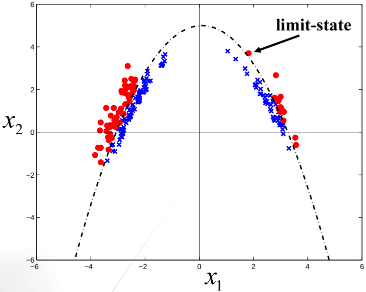
Subset second step

The second probability $P(F_2)$ is evaluated by conditional simulations

- N conditional simulations are generated by Markov Chains from germs: points of the preceding step in F_1
- the second threshold y_2 is defined as to find $P(F_2) \approx \alpha$

...

Subset simulation steps



Subset first step

The first probability $P(F_1)$ is evaluated by crude Monte-Carlo

- N simulations are generated
- the first threshold y_1 is defined as to find $P(F_1) \approx \alpha = 0.1$

Subset second step

The second probability $P(F_2)$ is evaluated by conditional simulations

- N conditional simulations are generated by Markov Chains from germs: points of the preceding step in F_1
- the second threshold y_2 is defined as to find $P(F_2) \approx \alpha$

...

Subset last step

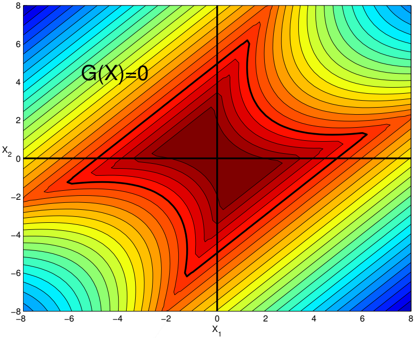
The last probability $P(F_m)$ is evaluated by conditional simulations

- N conditional simulations are generated by Markov Chains
- the last threshold $y_m = 0$ is reached after m steps (m depends on P_f value)

^{2}SMART - Idea

- **Objective**: to cumulate advantages of both approaches, **SMART** and Subsets simulations
- **Idea**: to use the learning strategy of **SMART** for the limit defined at each step of the subset
- **Strategy**: At each steps of subset, 2 phases:
  1. determine the threshold: by direct simulation (100 points), or regression (25 points) and simulation
  2. using **SMART** strategy with adapted work populations (conditional sampling by Markov Chains) for the evaluation of the failure probability at each step

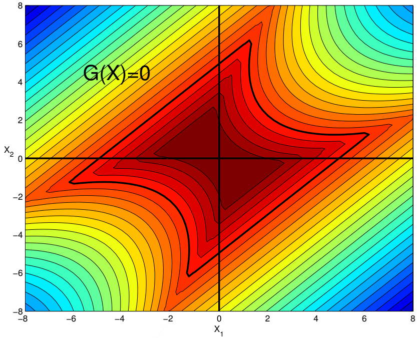
A short illustrative example



$$G(X) = \min \begin{cases} 3 + 0.1\,(X_1 - X_2)^2 - \dfrac{X_1 + X_2}{\sqrt{2}} \\ 3 + 0.1\,(X_1 - X_2)^2 + \dfrac{X_1 + X_2}{\sqrt{2}} \\ X_1 - X_2 + 3.5\,\sqrt{2} \\ X_2 - X_1 + 3.5\,\sqrt{2} \end{cases}$$

$X_{1,2}$ are standard Gauss variables

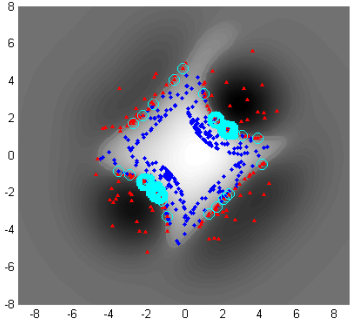
A short illustrative example



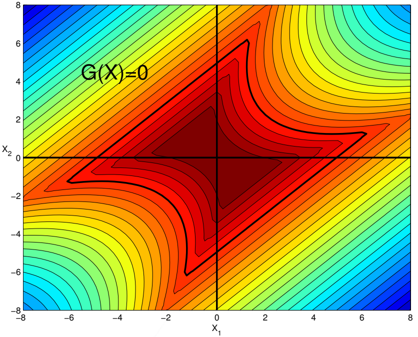
$$G(X) = \min \begin{cases} 3 + 0.1\,(X_1 - X_2)^2 - \dfrac{X_1 + X_2}{\sqrt{2}} \\ 3 + 0.1\,(X_1 - X_2)^2 + \dfrac{X_1 + X_2}{\sqrt{2}} \\ X_1 - X_2 + 3.5\,\sqrt{2} \\ X_2 - X_1 + 3.5\,\sqrt{2} \end{cases}$$

$X_{1,2}$ are standard Gauss variables

	MC	FORM	SS(AB)	SS(AB)	^{2}SMART
N calls	10^6	12	3000	30000	350
P_f (10^{-3})	2.22	1.34	2.35	2.44	2.21
β	2.85	3.00	2.83	2.81	2.85
Cov	0.02	—	0.23	0.06	0.01



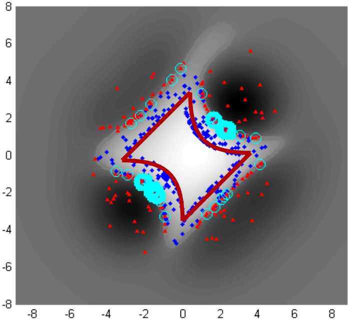
A short illustrative example



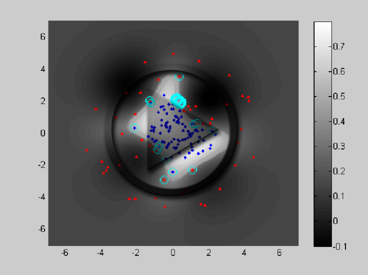
$$G(X) = \min \begin{cases} 3 + 0.1\,(X_1 - X_2)^2 - \dfrac{X_1 + X_2}{\sqrt{2}} \\ 3 + 0.1\,(X_1 - X_2)^2 + \dfrac{X_1 + X_2}{\sqrt{2}} \\ X_1 - X_2 + 3.5\,\sqrt{2} \\ X_2 - X_1 + 3.5\,\sqrt{2} \end{cases}$$

$X_{1,2}$ are standard Gauss variables

	MC	FORM	SS(AB)	SS(AB)	^{2}SMART
N calls	10^6	12	3000	30000	350
P_f (10^{-3})	2.22	1.34	2.35	2.44	2.21
β	2.85	3.00	2.83	2.81	2.85
Cov	0.02	–	0.23	0.06	0.01

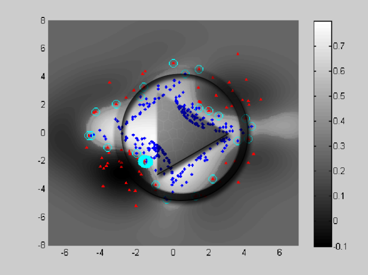


^{2}SMART - Steps illustration



First subset step

Second subset step



Contents

A non smooth performance function

A non linear oscillator

- Limit-state function definition:

$$G(X) = F_s - 3k_s \sqrt{\pi \frac{S_0}{4\zeta_s \omega_s^3} \frac{\zeta_a \zeta_s}{\zeta_p \zeta_s (4\zeta_a^2 + \theta^2) + \gamma \zeta_a^2} \frac{(\zeta_p \omega_p^3 + \zeta_s \omega_s^3)\omega_p}{4\zeta_a \omega_a^4}}$$

with:

$$\omega_p = \sqrt{\frac{k_p}{m_p}} \qquad \omega_a = \frac{1}{2}(\omega_p + \omega_s)$$

$$\omega_s = \sqrt{\frac{k_s}{m_s}} \qquad \zeta_a = \frac{1}{2}(\zeta_p + \zeta_s)$$

$$\gamma = \frac{m_s}{m_p} \qquad \theta = \frac{1}{\omega_a}(\omega_p - \omega_s)$$

- Random variables definition:

Random variables	pdf	Mean	cov
m_p	Lognormal	1.5	10%
m_s	Lognormal	0.01	10%
k_p	Lognormal	1	20%
k_s	Lognormal	0.01	20%
ζ_p	Lognormal	0.05	40%
ζ_s	Lognormal	0.02	50%
F_s	Lognormal	15	10%
S_0	Lognormal	100	10%

A non smooth performance function

A non linear oscillator

- Results from various methods and various threshold:

FORM	MC (100 runs)	SS(AB) (500 runs) (3.10^5 sim/step)	^{2}SMART (50 runs) (300 sim/step)
P_f (N calls)	P_f (N calls / Cov %)	P_f (N calls / Cov %)	P_f (N calls / Cov %)
$9.70 \cdot 10^{-2}$ (234)	$3.71 \cdot 10^{-2}$ (27000 / 3.2)	$3.71 \cdot 10^{-2}$ ($\approx 2 \times 3.10^5$ / 3.2)	$3.69 \cdot 10^{-2}$ ($\approx 2 \times 300$ / 2.1)
$2.19 \cdot 10^{-2}$ (1179)	$4.79 \cdot 10^{-3}$ (200000 / 2.9)	$4.78 \cdot 10^{-3}$ ($\approx 3 \times 3.10^5$ / 1.2)	$4.78 \cdot 10^{-3}$ ($\approx 3 \times 300$ / 3.4)
$2.72 \cdot 10^{-3}$ (3099)	$4.22 \cdot 10^{-4}$ (2300000 / 3.1)	$4.23 \cdot 10^{-4}$ ($\approx 4 \times 3.10^5$ / 1.9)	$4.18 \cdot 10^{-4}$ ($\approx 4 \times 300$ / 5.6)
do not conv.	-	$4.42 \cdot 10^{-5}$ ($\approx 5 \times 3.10^5$ / 2.5)	$4.42 \cdot 10^{-5}$ ($\approx 5 \times 300$ / 6.8)
do not conv.	-	$3.78 \cdot 10^{-7}$ ($\approx 7 \times 3.10^5$ / 4.0)	$3.66 \cdot 10^{-7}$ ($\approx 7 \times 300$ / 9.6)

Influence of the number of random variables

[R.Rackwitz, Reliability analysis - a review and some perspectives, 2001, Structural Safety]

- Test case evaluated for 3 dimensions: 40, 100 and 250

Dimension	40	100	250
P_f (10^{-3})	1.98	1.73	1.59
SS(AB)	2.4% 10^5 calls by step	2.3% 10^5 calls by step	2.4% 10^5 calls by step
SS(AB)	22% 1243 calls by step	17% 2012 calls by step	11% 3569 calls by step
^{2}SMART	2.8% 1243 calls by step 0.99	2.2% 2012 calls by step 1.01	2.6% 3569 calls by step 1.01

Coefficient of variation and error values are calculated on the basis of 20 runs.

Some remarks

- There is no biais on results obtained by the ^{2}SMART method, even in large dimension.
- Coefficients of variation obtained by the ^{2}SMART approach using about 2000 limit-state calls require 10^5 limit-state calls by subset simulation.

A random field application

8 holes plate with random Young modulus

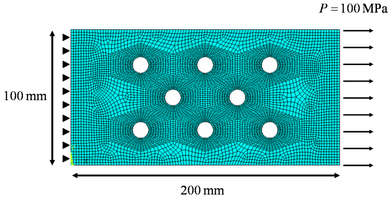
[P-L.Liu and A. Der Kiureghian, *Optimization algorithms for structural reliability analysis*, 1986, Tech. Report, University of California, Berkeley]

[B.Sudret and A. Der Kiureghian, *Stochastic Finite Element Methods and Reliability - A State-of-the-Art Report*, 2000, Tech. Report, University of California, Berkeley]

- The limit-state definition:

$$G(X) = \sigma_{limit} - \max(\sigma_{VonMises})$$

- Graphical representation of the structure:



- Random variables:

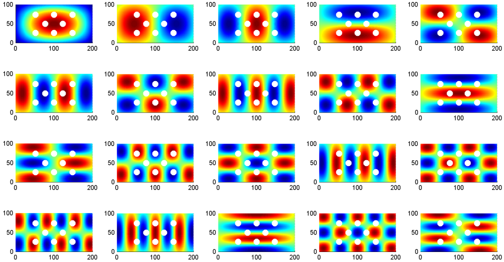
  20 random variables are used to defined the Young modulus of the plate by the EOLE method.

Difficulties are due to multiple design points, dimension and coupled procedure (FE model)

A random field application

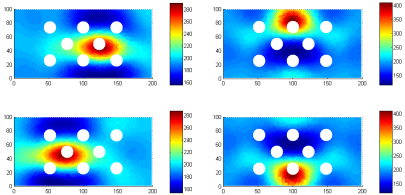
A pecision about random variables

- Graphical representation of the signification of each random variable into the Young modulus of the plate:

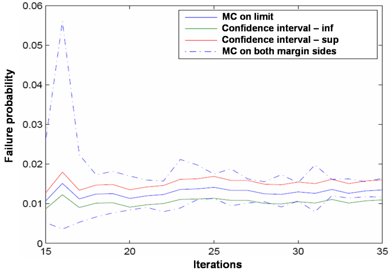


A random field application

Reliability results



Some probable failure points



A convergence index

	FORM	Multi-FORM	SS(AB)	SS(AB)	^{2}SMART
N calls	200	1000	2000	20000	600
P_f (10^{-2})	0.37	0.37	1.44	1.43	1.38
β	2.67	2.67	2.18	2.19	2.21

Conclusion

Important points

- Classification allows a serious reduction of limit-state calls.
- SVM is flexible respect to limit-state form, even for system approach...
- Adaptive approach: the number of constructed database points is more or less flexible...

Conclusion

- We dispose, with ^{2}SMART, of a tool for solving applications involving significant models of industrials products.

Support Vector Machine for Efficient Subset Simulations: ^{2}SMART method

F. Deheeger & M. Lemaire

Université Blaise Pascal
Institut Français de Mécanique Avancée

10th International Conference on Applications of Statistics and Probability in Civil Engineering