

# Robust Design and Uncertainty Quantification

Monte Carlo, Latin hypercube and Polynomial Chaos



-modeFRONTIER® is a registered  
product of [ESTECO srl](http://www.esteco.com)  
-Copyright © ESTECO srl 1999-2007

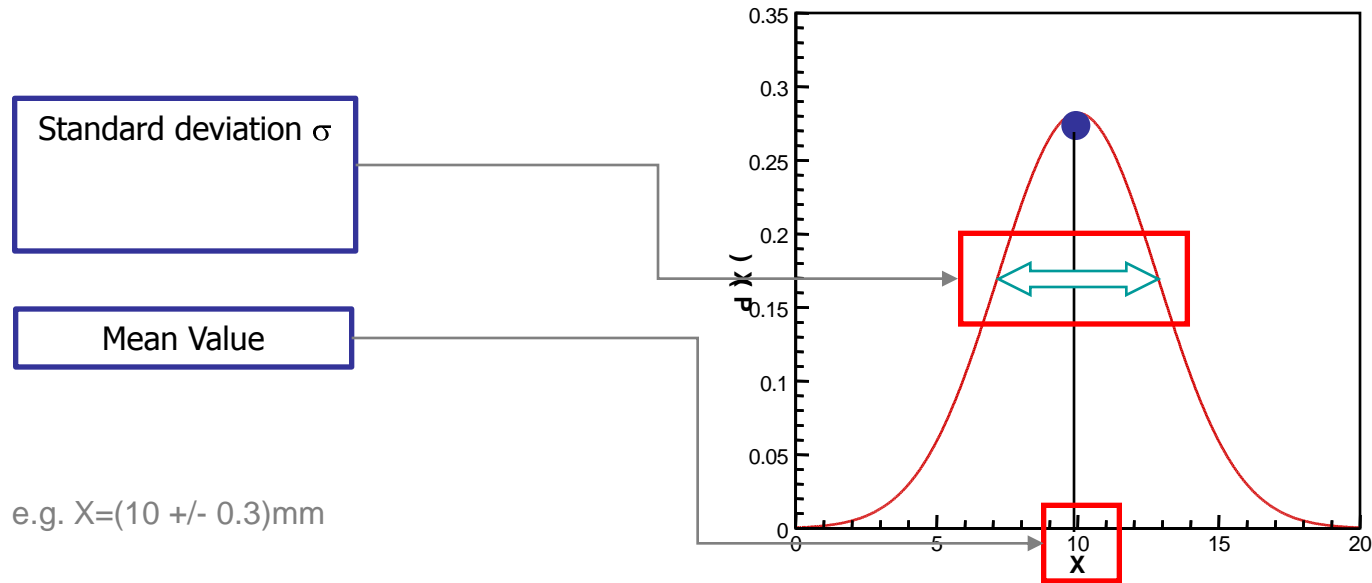


-For more information visit:  
-[www.esteco.com](http://www.esteco.com) or send an e-mail to:  
[modeFRONTIER@esteco.com](mailto:modeFRONTIER@esteco.com)

**modeFRONTIER**  
the multi-objective optimization and design environment

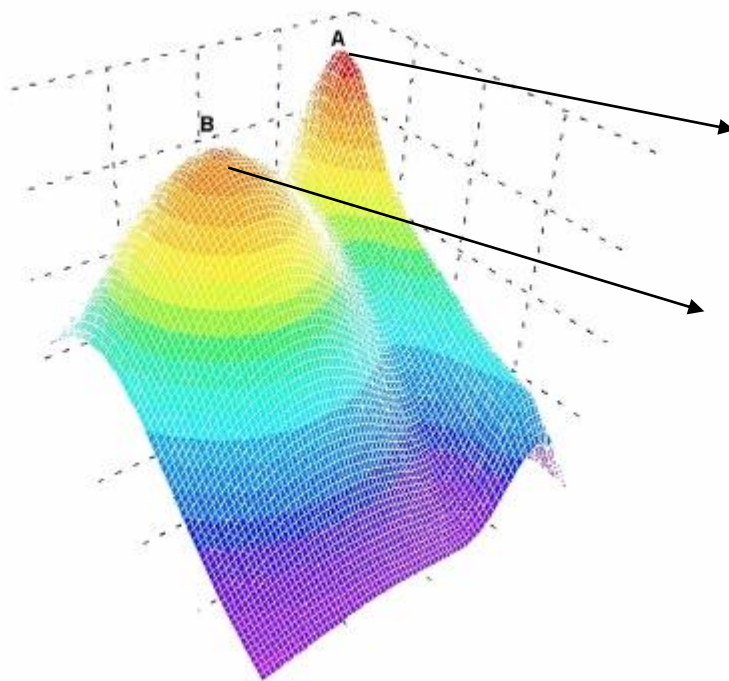
# Why robust designs optimizations?

- In many engineering problems **design parameters** may be **uncertain** (e.g. tolerances, fluctuating operating conditions, etc.)
- Input parameters are then defined not by a deterministic value, but as a **Distribution** (each value have a statistical probability to occur)



# Why robust designs optimizations?

- The Input parameters **uncertainty** is reflected in the **outputs** of the system: a solution good for a deterministic value of inputs, may be not **robust** for slight variations
- The **robustness** of the solution is defined as the characteristic of the system response to be insensitive to the variation of the input parameters
- A **Robust Design Optimisation** searches for Robust solutions



Best solution (if Robustness not considered)

Best solution (Robust Design approach)



# The Problem

---

Ordinary (MO) Optimization Problem:

$$\min_{x \in \mathbb{R}^n} (f_1(x), \dots, f_k(x))$$

How can we define a **stochasticized** (MO) optimization problem?

Assume  $f(x) = f(x_1, \dots, x_n)$  is still **deterministic** (as a **computer experiment**)

Otherwise  $x_1, \dots, x_n$  are affected by **uncertainty**, i.e.,

$$x_i \mapsto X_i \quad \text{random variable}$$

e. g.:

$X_i = X_i(x_i)$  is **Normal** centered in  $x_i$  with **fixed**  $\sigma_i$



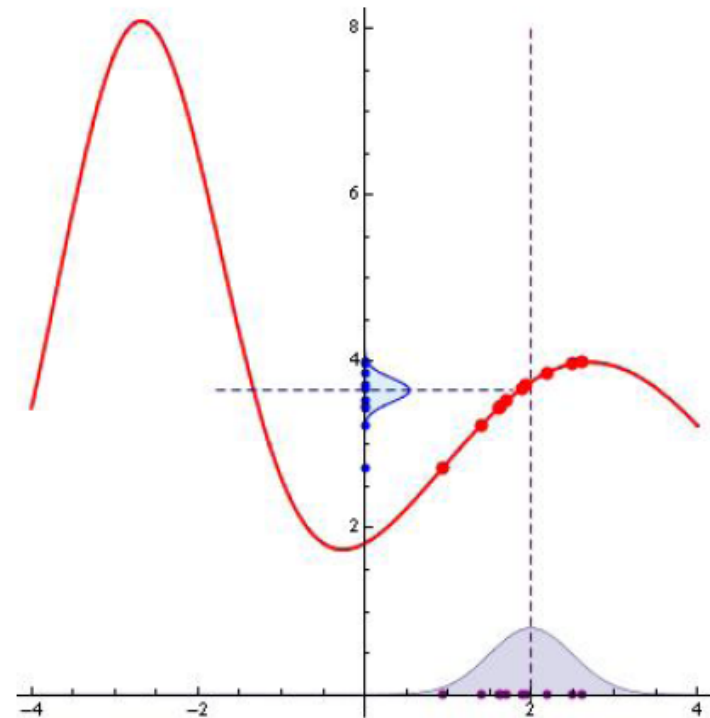
# A stochastic MOP

Also  $f$  is substituted by a random variable, defined as:

$$f(x_1, \dots, x_n) \longrightarrow F(x_1, \dots, x_n),$$

random variable

$$F(x_1, \dots, x_n) \stackrel{\text{def}}{\equiv} f(X_1(x_1), \dots, X_n(x_n))$$



# Stochastic MOP

Simply rewriting

$$\min_{x \in \mathbb{R}^n} (F_1(x), \dots, F_k(x))$$

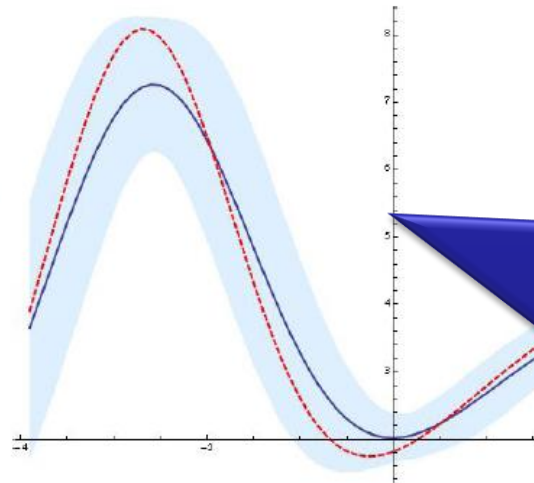
does **not make sense**.

We could consider, for instance,

$$\min_{x \in \mathbb{R}^n} (\mu_{F_1}(x), \dots, \mu_{F_k}(x))$$

plainly, or more sophisticatedly,

$$\min_{x \in \mathbb{R}^n} (\mu_{F_1}(x) + 3\sigma_{F_1}(x), \dots, \mu_{F_k}(x) + 3\sigma_{F_k}(x))$$



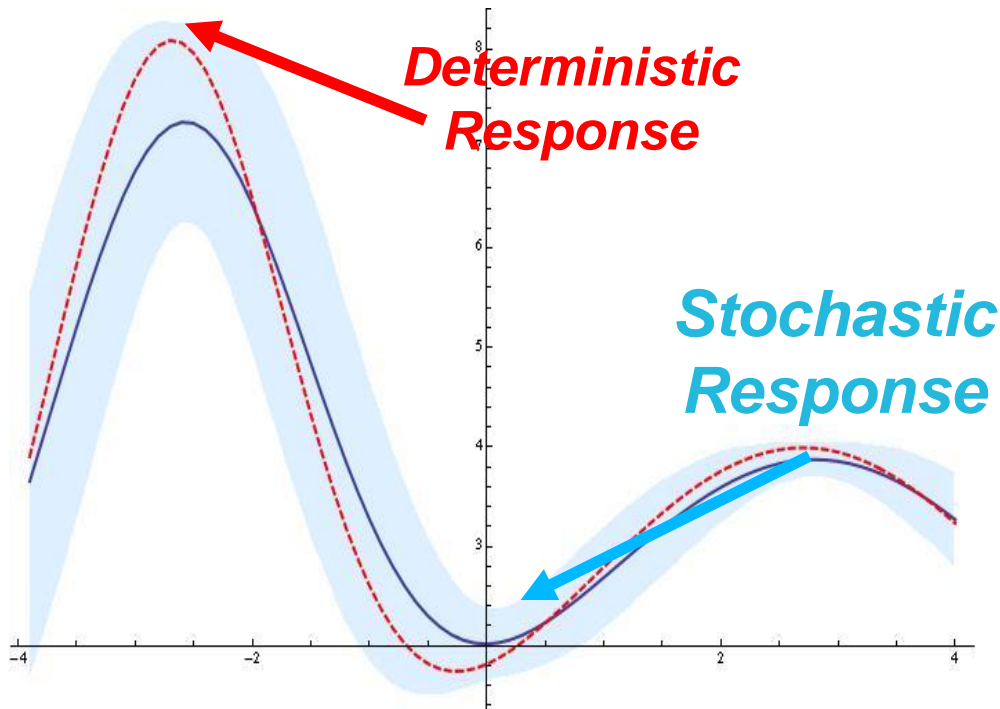
This means that, first of all, we have a problem of **uncertainty quantification**



# Uncertainty Quantification Comparisons

## *Multiobjective Robust Design Optimization (MORDO)*

searches for the optima of the mean and standard deviation of a stochastic response rather than the optima of the deterministic response (the output from the solver)



# Uncertainty Quantification

---

- How can we quantify uncertainty?
  1. Analytically
  2. Monte Carlo
  3. Latin Hypercube
  4. Polynomial Chaos
  5. ...





# Analytically

---

- There exists several special cases in which the statistical moments can be determined analytically on the basis of the statistical moments of the independent uncertain variables  $X_1, \dots, X_k$ .
- Consider for instance the trivial case:
  - $Y = f(X) := 3X$
  - It is clear that  $E[Y] = 3E[X]$
  - and that  $E[Y^2] = 9E[X^2]$ , then  $\sigma_Y = 3\sigma_X$
- solving the uncertainty quantification analytically may be difficult (or even impossible) for more complex functions and, moreover, the function **should be known**



# Monte Carlo

---

The most classical methodology consists in drawing a random sample  $\{x^{(1)}, \dots, x^{(N)}\}$  from the joint distribution  $\mathcal{D}_1 \otimes \dots \otimes \mathcal{D}_d$ , i.e., drawing  $N$  random numbers  $\{x_i^{(1)}, \dots, x_i^{(N)}\}$  from every distribution  $\mathcal{D}_i$  and then collecting the vectors

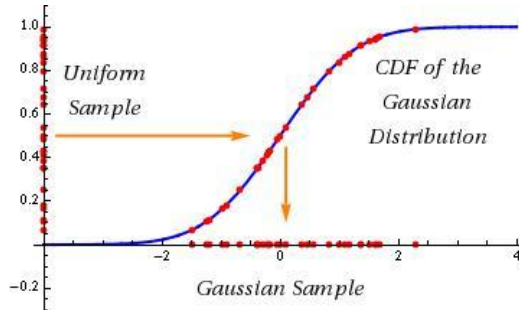
$$\begin{aligned}x^{(1)} &:= \left(x_1^{(1)}, \dots, x_d^{(1)}\right)^T, \\x^{(2)} &:= \left(x_1^{(2)}, \dots, x_d^{(2)}\right)^T, \\&\vdots \\x^{(N)} &:= \left(x_1^{(N)}, \dots, x_d^{(N)}\right)^T.\end{aligned}$$

A sample  $s_Y := \{y^{(1)}, \dots, y^{(N)}\}$  of the uncertain dependent variable  $Y$  is computed through application of  $f$  to each one of the  $x^{(j)}$ :

$$y^{(j)} := f\left(x_1^{(j)}, \dots, x_d^{(j)}\right), \quad j = 1, \dots, N. \quad (2)$$

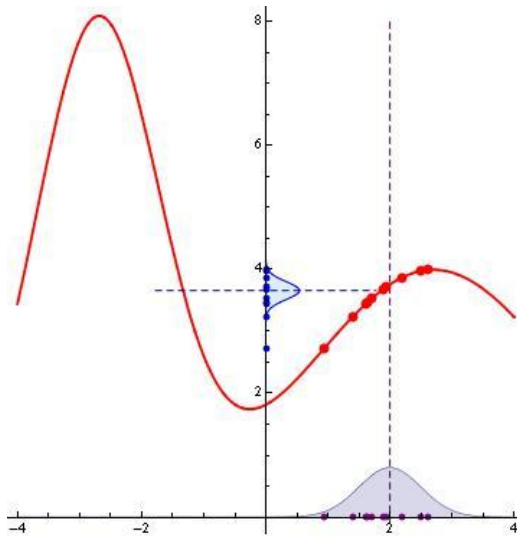


# Monte Carlo



$$m_Y \simeq \bar{m}_Y := \langle s_Y \rangle = \frac{1}{N} \sum_{j=1}^N y^{(j)},$$

$$\sigma_Y \simeq \bar{\sigma}_Y := \sqrt{\frac{1}{N-1} \sum_{j=1}^N (y^{(j)} - \bar{m}_Y)^2}.$$



- Monte Carlo is a robust method
- Statistics obtained via Monte Carlo are reliable
- **but very poor in accuracy**
- Too many points are needed



# Monte Carlo Accuracy

---

- The convergence of the method is

$$\frac{1}{\sqrt{N}}$$

- **To halve the estimation error it is necessary a four times larger sample**, to reduce of an order of magnitude it is necessary to take a sample 100 times larger.



# Example

---

Let  $Y = f(X)$ , where

$$f(x) := 20 \left( \frac{e^{-\frac{(x+2.7)^2}{2}}}{\sqrt{2\pi}} + \frac{e^{-\frac{(x-2.7)^2}{2(2^2)}}}{\sqrt{2\pi}2} \right),$$

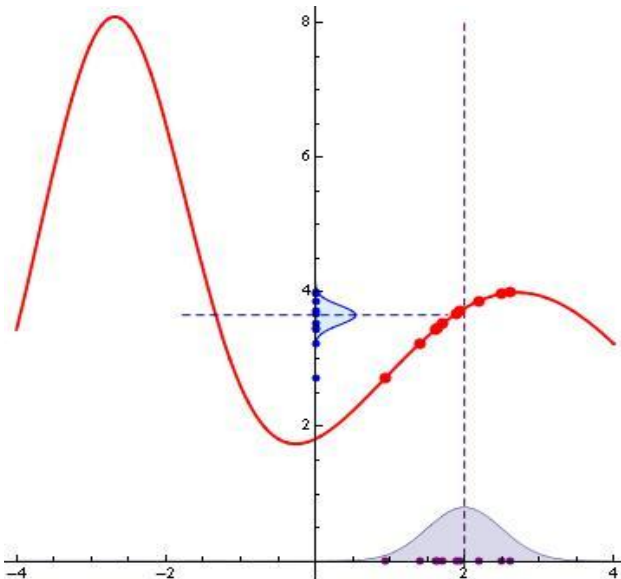
$$X \sim \mathcal{N}(2, 0.8), \quad \text{i.e.,} \quad w(x) = \frac{e^{-\frac{(x-2)^2}{2(0.8^2)}}}{\sqrt{2\pi}0.8}.$$

$$m_Y = E[f(X)] = \int f(x) w(x) dx \simeq 3.5209849377883446,$$

$$\sigma_Y = \sqrt{E[(f(x) - m_Y)^2]} = \sqrt{\int f(x)^2 w(x) dx - m_Y^2} \simeq 0.507313175$$



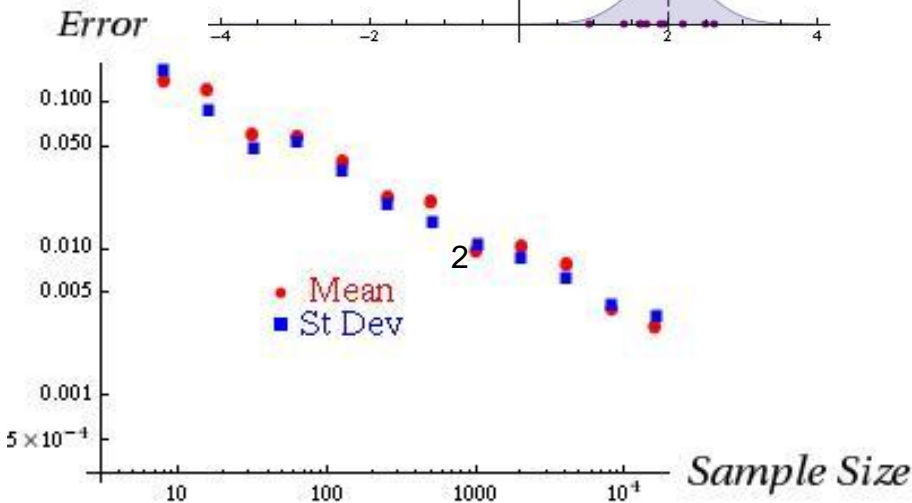
# Monte Carlo sampling



Statistics can be computed via Monte Carlo, i.e., **drawing a random sample** from the assigned stochastic distributions from the input variables

Monte Carlo converges slowly to true values statistics

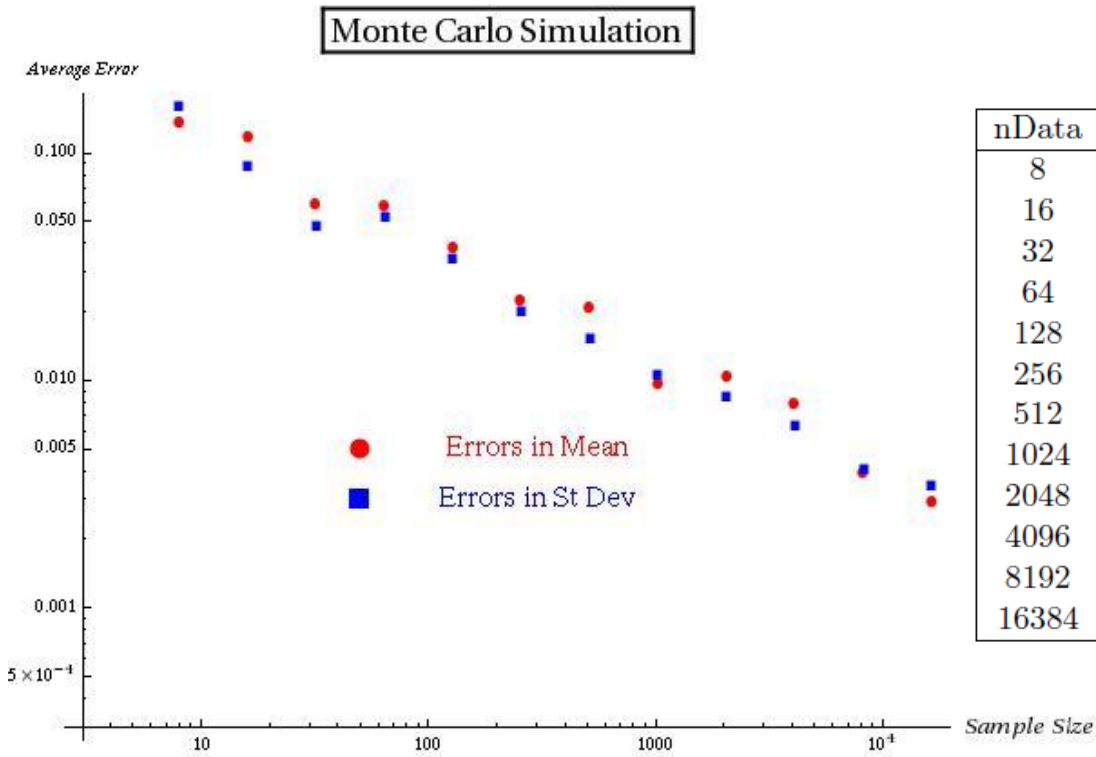
Large samples are needed to reach reliable results



*Former example:  
to reach **1% Error** in  
**Mean** needs **256 points**  
**Standard Deviation**  
needs **8192 points***



# Monte Carlo Accuracy



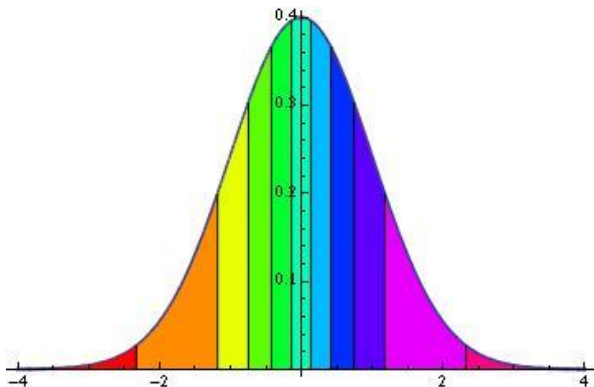
nData	Average Error Mean	Average Error StDev
8	0.13523321821576068	0.1610272359435758
16	0.11711965574510605	0.0864659032341357
32	0.059222020366550845	0.04713102966918838
64	0.057784528160956625	0.05253445820159186
128	0.03838430554585017	0.034136016860713006
256	0.022344036432755	0.020111546424191107
512	0.02061047535304621	0.015098327730704386
1024	0.009641651869702561	0.010470922893152998
2048	0.010312701913743005	0.008504295088329538
4096	0.007855294629528543	0.006264945824679183
8192	0.003879754790574208	0.004073598991190089
16384	0.0029156602991482483	0.0034092125680575958



# Latin Hypercube

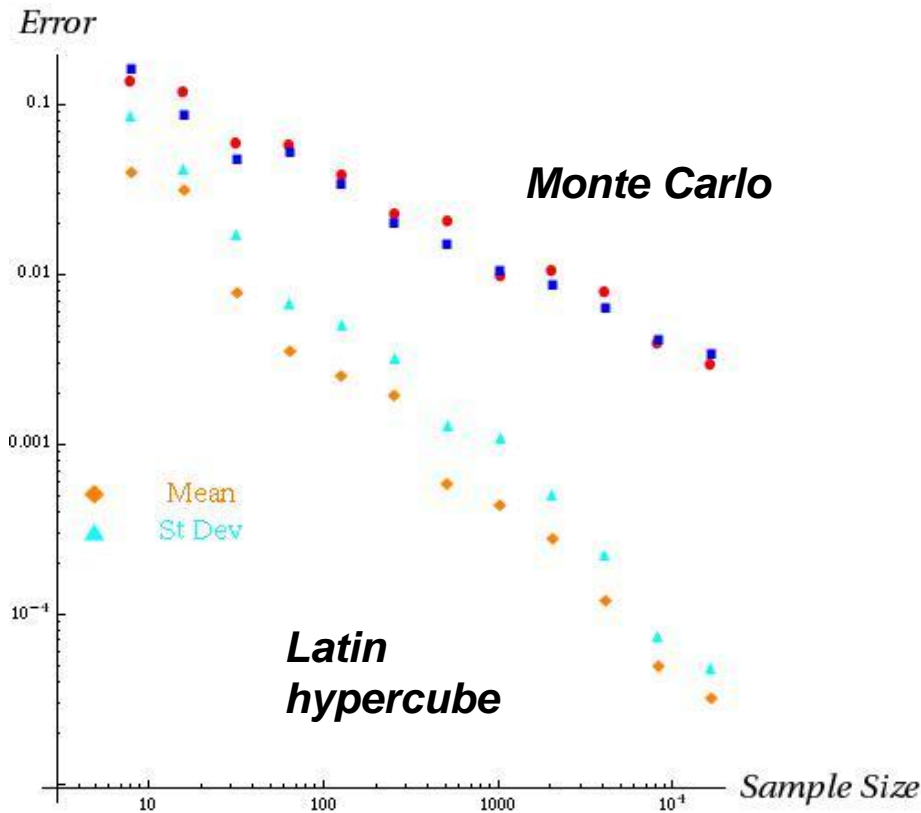
---

- To speed up the convergence of statistical moments to actual moments, there exists a smarter strategy: a Latin Hypercube Sampling (LHS)
- Latin Hypercubes is a sampling strategy derived from **stratified samplings**
- If LHS is composed of N points, and every variable is divided into N stratum with equal probability, every single stratum will be occupied by exactly one point.





# Latin hypercube sampling



The random sample can be substituted by a stratified sampling as **Latin hypercube**. Statistics converge faster to exact values

Medium-large samples are needed to reach good results

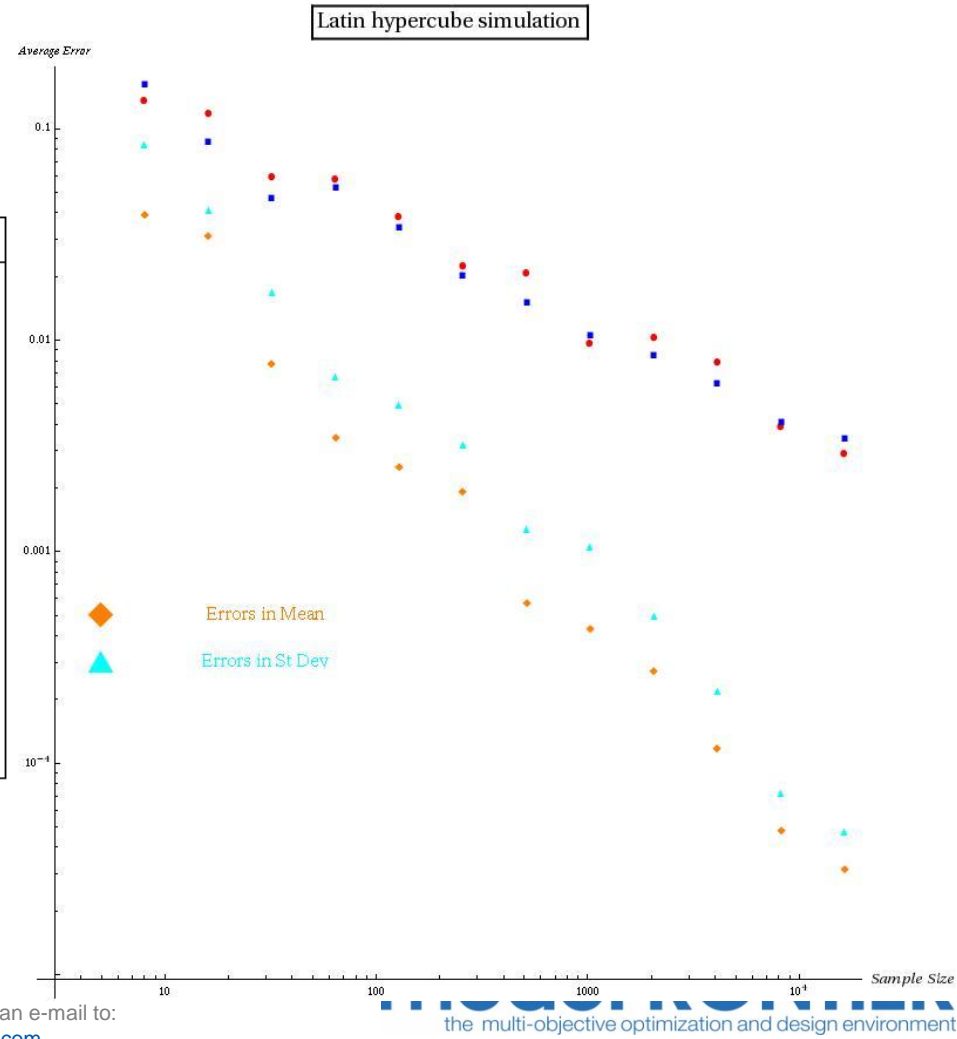
*Former example:*  
to reach **1% Error** in  
**Mean** needs **16 points**  
**Standard Deviation**  
needs **128 points**



# LHS Accuracy

Table of results for the example

nData	Average Error Mean	Average Error StDev
8	0.03838708340684853	0.0823640656227671
16	0.030567950211235484	0.040710534223220365
32	0.007571863928251643	0.01648128041162316
64	0.003418078142074843	0.006587126182655906
128	0.0024618618972161777	0.004848681021425105
256	0.0018873455437873998	0.003133046085056787
512	5.648087501843202E-4	0.0012508917015900789
1024	4.2176395159760905E-4	0.001042445154700511
2048	2.665184159121647E-4	4.888817764855613E-4
4096	1.1563639765082012E-4	2.1436136917031833E-4
8192	4.724986318238589E-5	7.123204436129126E-5
16384	3.0972434058185175E-5	4.675409217411719E-5



# Polynomial Chaos

- Polynomial Chaos is very efficient
- This methodology originates from the work of Norbert Wiener and is called **Polynomial Chaos Expansion**.
- This methodology consists essentially in expanding the uncertain variable in a **suitable series** and then **determine analytically the statistical moments** of the truncated expansion.

$$f(x) := \sum_{i=0}^k \alpha_i p_i(x).$$

Polynomial degree

$$\langle p_i(x), p_j(x) \rangle_w := \int p_i(x) p_j(x) w(x) dx = 0, \quad \text{whenever } i \neq j.$$

orthogonal  
polynomials



# Polynomial Chaos Expansion

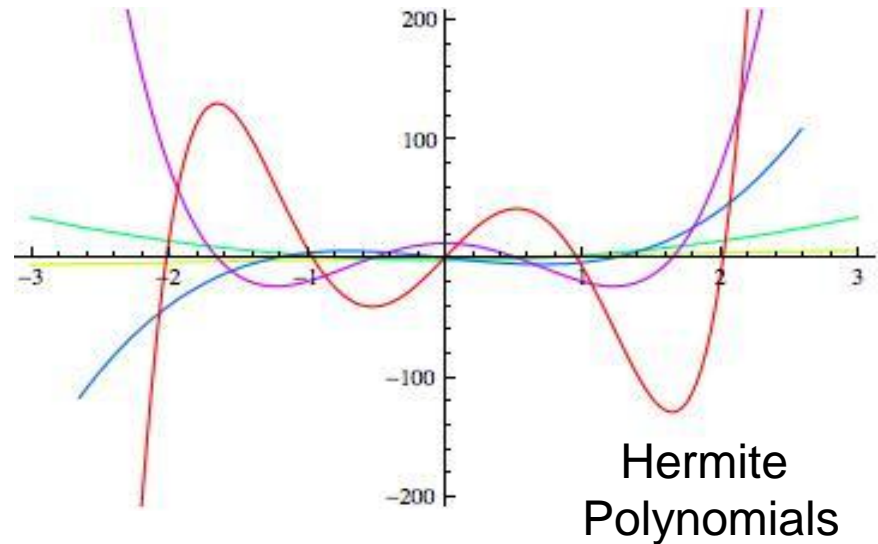
Alternatively, it is possible to expand the response in a special series, a **Polynomial Chaos**.

$$f(x) \cong \sum_{i=0}^k \alpha_i He_i(x)$$

$$Mean(Y) \cong \alpha_0$$

$$\sigma^2(Y) \cong \sum_{i=0}^k \alpha_i^2 (i!)$$

$$\# \{sample\} \geq \frac{(nVar + order)!}{nVar! order!}$$



The statistics (mean and standard dev) of a Polynomial Chaos are computed analytically

Estimate is extremely precise and sample are very small

Sample size depend on the number of stochastic variables ( $nVar$ ) and on the order of the expansion



# Polynomial Chaos

---

The m-th moment may be written as:

$$\begin{aligned}\langle y^m \rangle &= \int (f(x))^m w(x) dx = \int \left( \sum_{i=0}^k \alpha_i p_i(x) \right)^m w(x) dx = \\ &= \int \sum \alpha_{i_1} \dots \alpha_{i_m} p_{i_1}(x) \dots p_{i_m}(x) w(x) dx = \\ &= \sum \alpha_{i_1} \dots \alpha_{i_m} \int p_{i_1}(x) \dots p_{i_m}(x) w(x) dx.\end{aligned}$$

If the distribution is Gaussian we can use the Hermite polynomials and compute the moments as:

$$m_Y = \alpha_0.$$

$$\sigma_Y = \sqrt{\sigma_Y^2} = \sqrt{\langle Y^2 \rangle - m_Y^2}.$$



# Polynomial Chaos

- Wiener-Askey scheme on relations between probability distributions and orthogonal polynomials
- In the case of multiple variables, assuming they are independently distributed, it is possible to write “multivariate chaos” considering the tensorial products of the univariate polynomials.

Distribution	Probability Density	Orthogonal Polynomials	Support Range
Normal	$\frac{e^{-\frac{x^2}{2}}}{\sqrt{2\pi}}$	Hermite $He_n(x)$	$(-\infty, +\infty)$
Uniform	$1/2$	Legendre $P_n(x)$	$(-1, +1)$
Beta	$\frac{(1-x)^\alpha(1-x)^\beta}{2^{\alpha+\beta+1}B(\alpha+1,\beta+1)}$	Jacobi $P_n^{(\alpha,\beta)}(x)$	$(-1, +1)$
Exponential	$e^{-x}$	Laguerre $L_n(x)$	$(0, +\infty)$
Gamma	$\frac{x^\alpha e^{-x}}{\Gamma(\alpha+1)}$	Generalized Laguerre $L_n^{(\alpha)}(x)$	$(0, +\infty)$



# Computing the coefficients

---

- To find the coefficients we have to solve a single objective optimization problem (e.g. Levenberg–Marquardt)

$$\alpha_i : \min_{\alpha_i} \sum_{j=1}^N \left| f(\bar{x}_j) - \sum_{i=1}^k \alpha_i p_i(\bar{x}_j) \right|^2, \\ \{\bar{x}_1, \dots, \bar{x}_N\}, \quad \text{arbitrary sample}$$

- The size N of the sample has to be at least equal or larger than the number of parameters  $N \geq \frac{(k+d)!}{k!d!}$



# Number of parameters

Number of parameters in a chaos of order  $k$  in  $d$  variables, i.e., minimum size of a sample to be employed for chaos collocation.

$N \geq \frac{(k+d)!}{k!d!}$	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$	$k = 6$	$k = 7$	$k = 8$
$d = 1$	2	3	4	5	6	7	8	9
$d = 2$	3	6	10	15	21	28	36	45
$d = 3$	4	10	20	35	56	84	120	165
$d = 4$	5	15	35	70	126	210	330	495
$d = 5$	6	21	56	126	252	462	792	1287
$d = 6$	7	28	84	210	462	924	1716	3003
$d = 7$	8	36	120	330	792	1716	3432	6435
$d = 8$	9	45	165	495	1287	3003	6435	12870
$d = 9$	10	55	220	715	2002	5005	11440	24310
$d = 10$	11	66	286	1001	3003	8008	19448	43758
$d = 11$	12	78	364	1365	4368	12376	31824	75582
$d = 12$	13	91	455	1820	6188	18564	50388	125970
⋮								



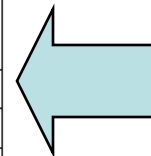


# Polynomial Chaos Accuracy

Table of results for the example

Chaos Order	Sample Size	Average Error Mean	Average Error StDev
1	4	0.09344069282015036	0.10429522822805484
2	6	0.019326055906163132	0.0697189377206368
3	8	0.011271788209210331	0.02654736033749695
4	10	8.03467578078676E-4	0.001684371972847451
5	12	7.877378937433344E-4	0.0028642320499342065
6	14	6.138461875910162E-4	0.0018260971296785306
7	16	2.663000586760944E-4	8.607678781524158E-4
8	18	3.311333063379607E-4	2.9924223556135885E-4
9	20	3.4444147888486044E-4	3.7666167066683075E-4
10	22	6.672160731088228E-5	1.358801147891109E-4
11	24	4.41927163589595E-5	1.2805374914218737E-4
12	26	8.515981157662944E-5	2.491946753026775E-4
13	28	8.775184447483708E-5	2.5355458755216276E-4
14	30	1.2559038776682741E-5	1.1160349235239675E-5

	Sample size needed to reach 1% accuracy for Mean / St Dev	Sample size needed to reach 0.1% accuracy for Mean / St Dev
Monte Carlo	256 / 8192	16384 / $\approx 800000?$
Latin hypercube	16 / 128	64 / 2048
Polynomial Chaos	8 / 12	12 / 20

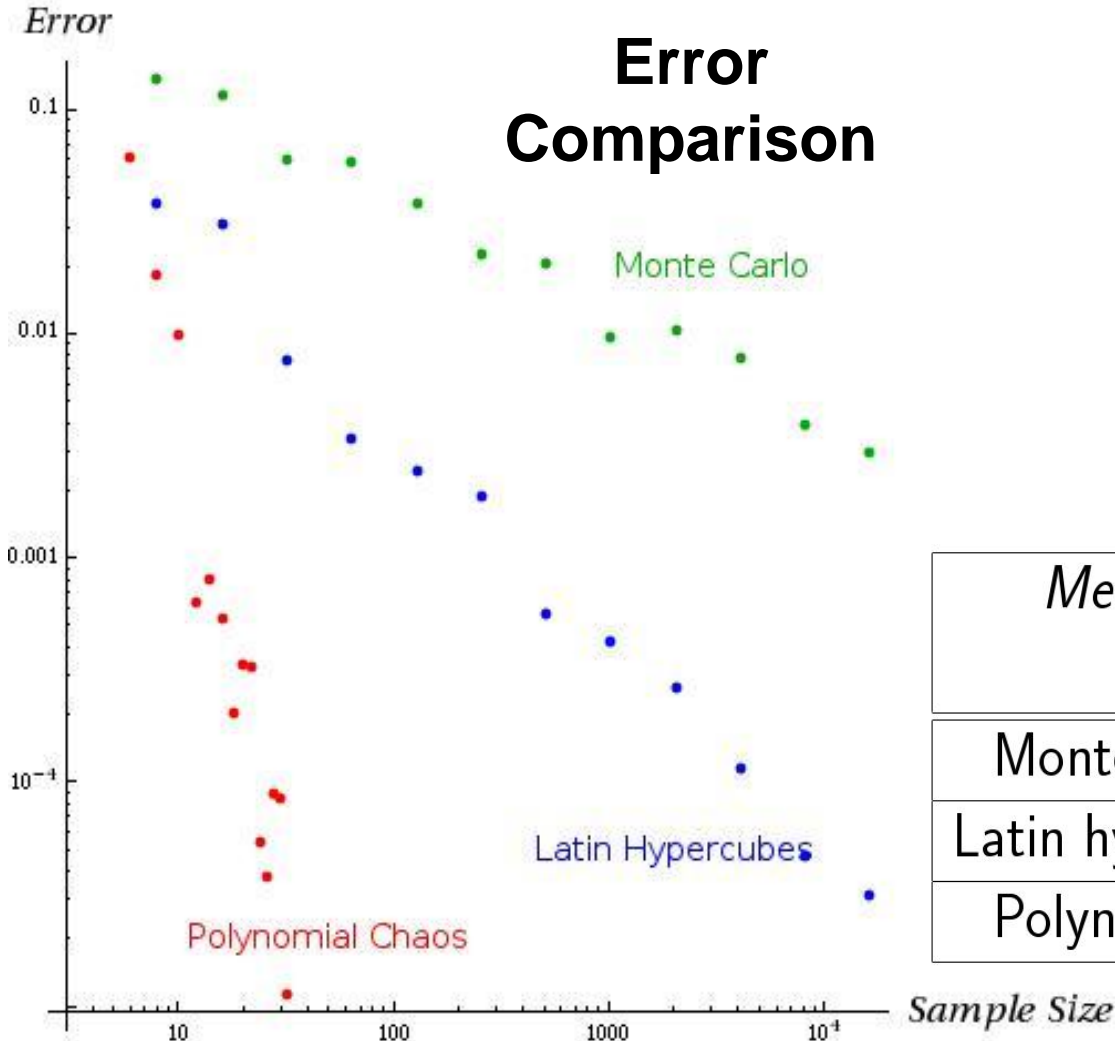


Summary of the performances and comparison between Monte Carlo, Latin hypercube and polynomial chaos for the example problem



# Polynomial Chaos Expansion

## Error Comparison



Former example:  
to reach **1% Error** in  
**Mean** needs **8 points**

**Standard Deviation**  
needs **12 points**

<i>Method</i>	<i>Mean</i> 1% error	<i>St Dev</i> 1% error
Monte Carlo	256	8192
Latin hypercube	16	128
Polyn Chaos	<b>8</b>	<b>12</b>



# Polynomial Chaos in modeFRONTIER

The screenshot displays the modeFRONTIER 4.1.0 software interface. The main window shows a design space with a single variable 'x'. A 'Scheduler Properties' dialog box is open, showing the 'Optimization Wizard' with 'NSGA-II' selected under 'Advanced Optimizers'. The 'Parameters' section is expanded, showing settings for 'Number of Generations' (1,500), 'Crossover Probability' (0.9), and 'Mutation Probability' (1.0). The 'Run Options' section is also expanded, showing 'Polynomial Chaos' checked under 'MORDO Advanced Options'.

**Scheduler Properties - 4.1.0 b20081205**

**Optimization Wizard**

- ↳ b-BF-GS
- ↳ Levenberg-Marquardt
- ↳ MOGA-II
- ↳ ARMOGA
- ↳ **Advanced Optimizers**
- ↳ MOSA
- ↳ **NSGA-II**
- ↳ MOGT
- ↳ MOPSO
- ↳ FMOGA-II
- ↳ FSIMPLEX
- ↳ Evolution Strategies
- ↳ Evolution Strategy
- ↳ 1P1-ES
- ↳ DES
- ↳ MMES
- ↳ Sequential Quadratic Programming
- ↳ NLPQLP
- ↳ NBI-NLPQLP

**NSGA-II**

Scheduler based on NSGA-II - Non-dominated Sorting Genetic Algorithm II of prof. K. Deb et al. (2000, KanGAL Report No. 200001).

Main features:

- 1) Allows both continuous ("real-coded") and discrete ("binary-coded") variables.
- 2) Allows user defined discretization (base).
- 3) The constraint handling method does not make use of penalty parameters.
- 4) Implements elitism for multiobjective search.
- 5) Diversity and spread of solutions is guaranteed without use of sharing parameters.
- 6) Allows concurrent evaluation of the n independent individuals.

The n (number of individuals per generation) entries in the DOE table are used as the problem's initial population.

**Parameters**

Number of Generations	[1,500]	100
Crossover Probability	[0.0,1.0]	0.9
Mutation Probability for Real-Coded Vect...	[0.0,1.0]	1.0
Mutation Probability for Binary Strings	[0.0,1.0]	1.0

**Advanced Parameters**

Distribution Index for Real-Coded Cro...	[0.5,100.0]	20.0
Distribution Index for Real-Coded Mut...	[0.5,500.0]	20.0
Crossover Type for Binary-Coded Variables		Simple
Random Generator Seed	[0,999]	1

**Category Parameters**

Categorize Generations		<input type="checkbox"/>
------------------------	--	--------------------------

**Run Options | RSM Options | MORDO Options**

Sampling Mode	Latin Hypercube Sampling
Seed	1
Num. of Design Samples	50
Num. of Virtual Samples	0
↳ MORDO Advanced Options	
Polynomial Chaos	<input checked="" type="checkbox"/>
Order of Chaos Expansion	2
Reject Out of Bounds Samples	<input type="checkbox"/>
Error Samples Acceptance Level %	100

**Vector Input Variable Properties - 4.1.0**

Name	Description	Size
De..._Vari..._Co..._Exp..._Distributor		
0	Vari..._0.0...	Norma
1	Vari..._0.0...	Norma
2	Vari..._0.0...	Norma

**Run Options**

Sampling Mode	Latin Hypercube Sampling
Seed	1
Num. of Design Samples	50
Num. of Virtual Samples	0
↳ MORDO Advanced Options	
Polynomial Chaos	<input checked="" type="checkbox"/>
Order of Chaos Expansion	2
Reject Out of Bounds Samples	<input type="checkbox"/>
Error Samples Acceptance Level %	100

**Properties**

No Properties

**Format**

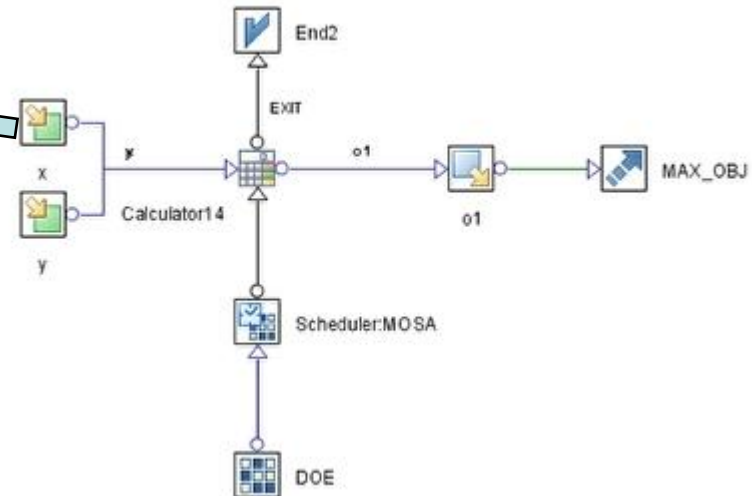
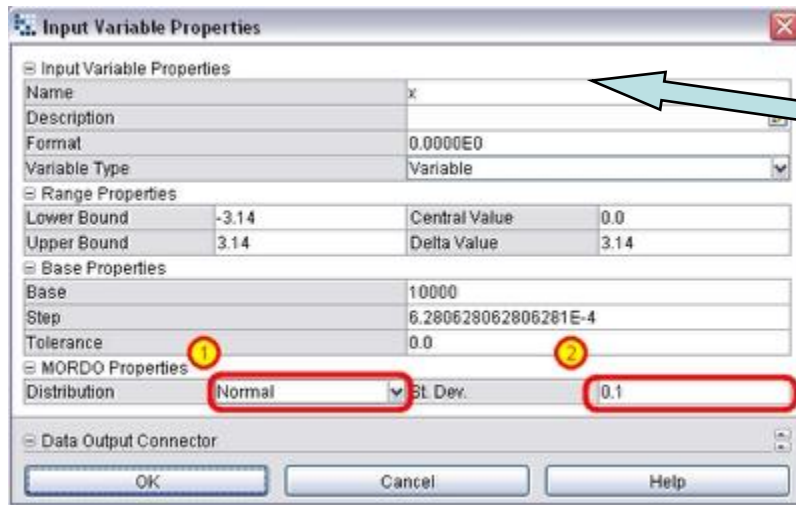
0.0000E0
0.0000E0

Ready | Mode: EDIT | modeFRONTIER 4.1.0 b20081205 | 30M / 508M

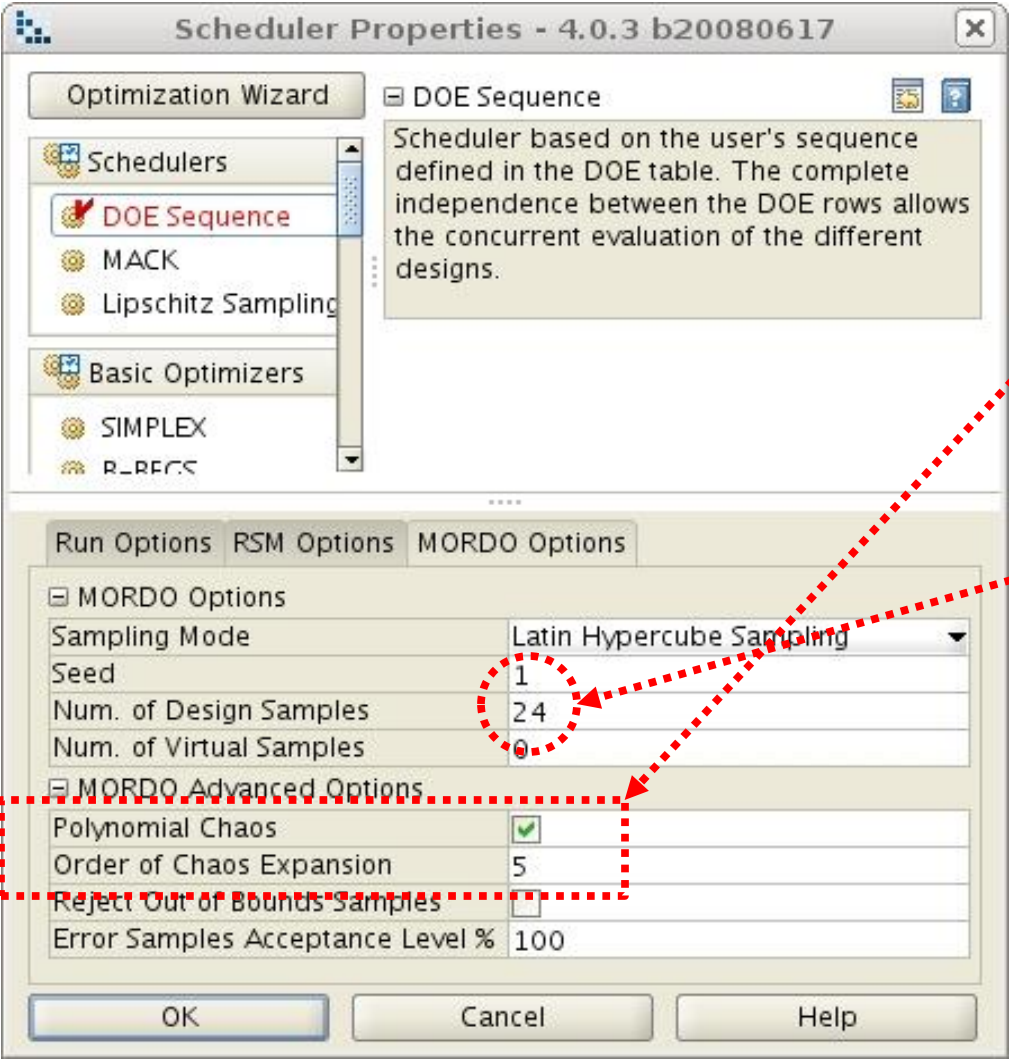


# Create a stochastic project: variable distribution

- Several **Distribution** to model Input variables uncertainties are available:
  - Cauchy
  - Logistic
  - Normal
  - Uniform
- Select it in **MORDO options** for each variable and specify the **parameters** (e.g., Standard Deviation for Normal type)



# Polynomial Chaos Interface



Polynomial Chaos is switched on by default

Chaos order can vary from 1 to 7

Number of Samples must be at least

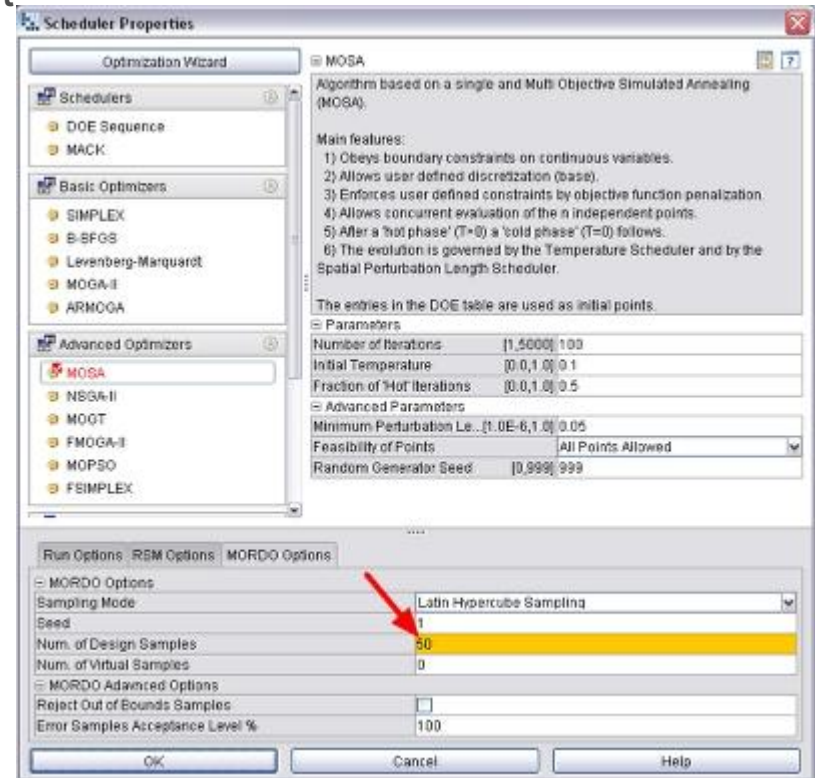
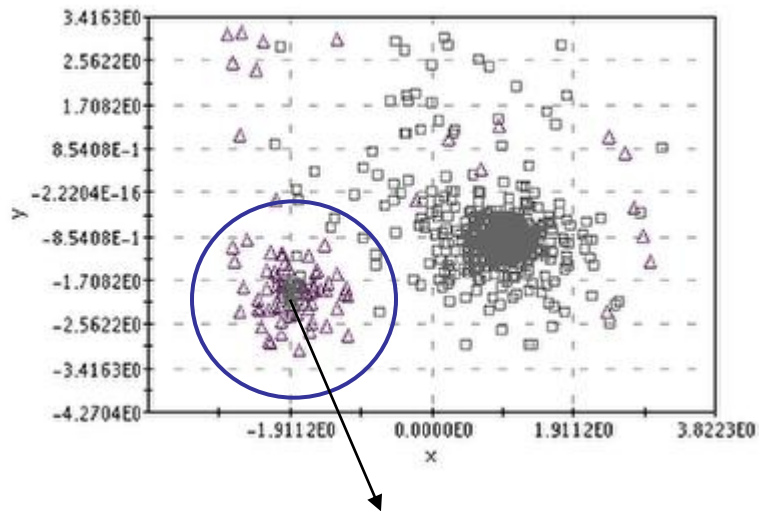
$$\frac{(nVar + order)!}{nVar!order!}$$

Virtual samples are to be avoided



# Create a stochastic project: sampling points

- The **number of samples** indicates how many simulations are run for each design during the optimisation, accordingly to input variable distribution
- Higher is the value, more accurate will be the outputs robustness analysis, but harder will be the computational effort





# Problems & Ideas

---

- Open Problems:

- How to deal with bounds and constraints
- What if the function cannot be expanded in a series (how can we decide that for a black-box function)
- ....

- Ideas:

- Use of chaos collocation to train a meta-model for virtual optimization
- ...



# Exercise 1

---

- Load polChaos.prj
- Check the function
- Run
  - Mean and variance estimation with polynomial chaos (e.g. polynomial degree 7, 30 sample points)
  - Mean and variance estimation with Latin Hypercube (keep the number of points fixed for a fair comparison)
  - Mean and variance estimation with Monte Carlo
- Verify the errors

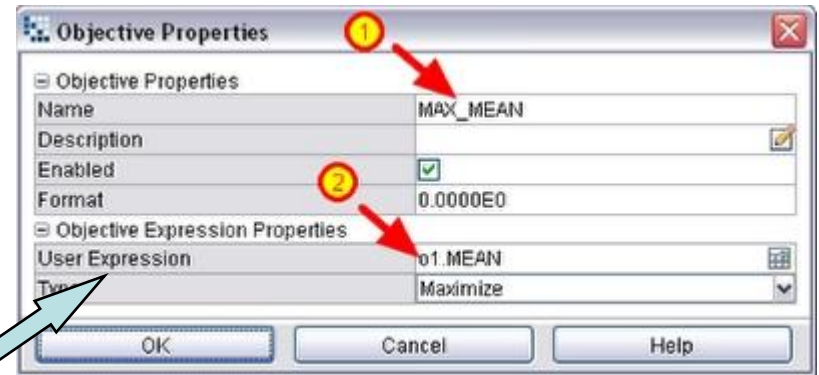
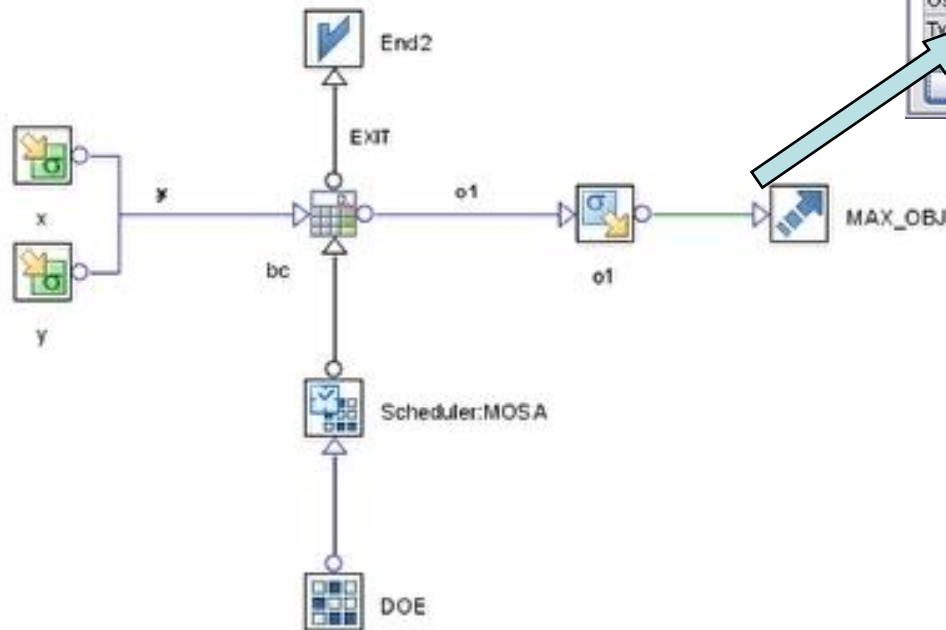




# Create a stochastic project: objectives definition

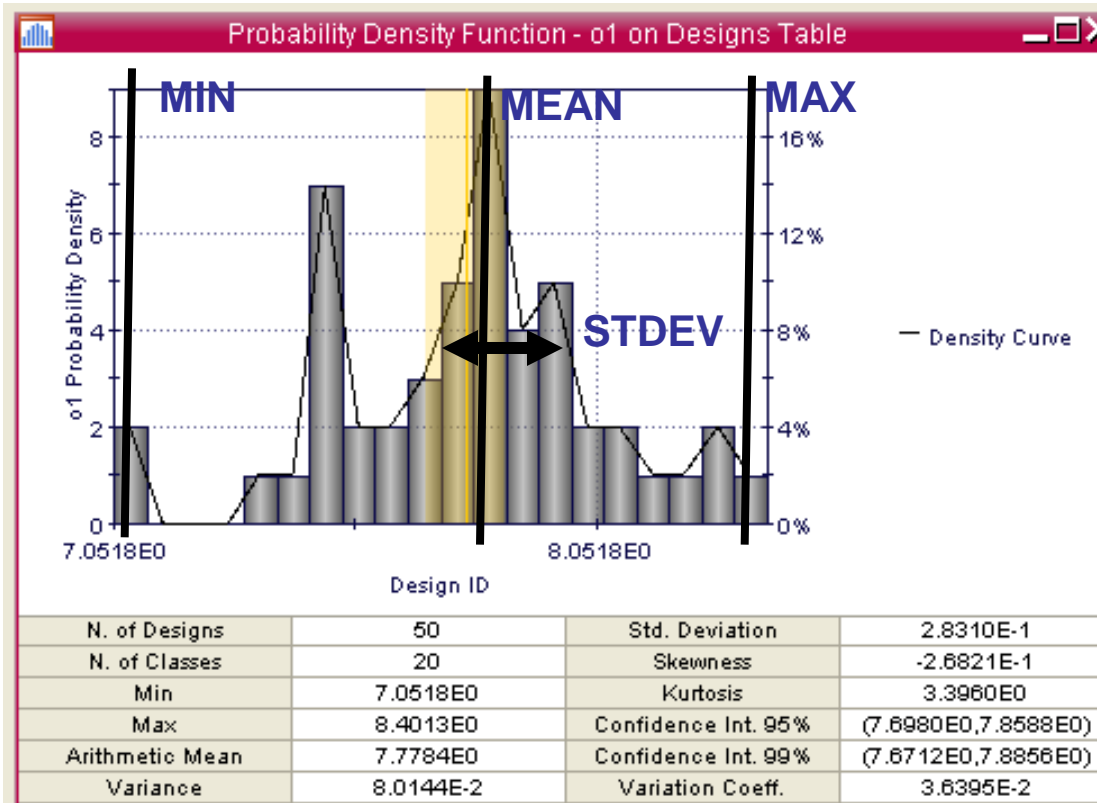
- The objectives (constraints) can be defined using **Stochastic value** for each output variable. If the latter is o1, these values are:

- o1:MEAN
- o1:STDEV
- o1:MAX
- o1:MIN



# Create a stochastic project: objectives definition

- For each design, a Distribution is obtained for the outputs, collecting the results of the samples:
- The quantities to be used in optimisation are:



- you can **minimise MEAN** and **constrain STDEV**
- you can **minimise MAX** as alternative



# Robust Design Table

modeFRONTIER 4 - Project :testMORDO.prj

File Edit Project Assessment View Window Help

Workflow Run Logs Designs Space

Robust Design Table

RID	M	CATEGORY	x	x.MEAN	x.STDEV	x.MIN	x.MAX
0		MOSA	3.1403E-4	-1.3867E-3	1.0340E-1	-3.1650E-1	2.2604E-1
1		MOSA	-1.5698E0	-1.5699E0	9.9909E-2	-1.8356E0	-1.3439E0
2		MOSA	1.5705E0	1.5694E0	1.0548E-1	1.2789E0	1.8697E0
3		MOSA	-7.8476E-1	-7.8694E-1	1.0068E-1	-1.0857E0	-5.7202E-1
4		MOSA	2.3555E0	2.3569E0	1.0299E-1	2.1383E0	2.6812E0
5		MOSA	-2.3549E0	-2.3566E0	1.0031E-1	-2.6385E0	-2.1477E0
6		MOSA	-1.2907E-1	-1.2848E-1	1.0417E-1	-3.9009E-1	1.7053E-1
7		MOSA	-9.1666E-1	-9.1642E-1	1.0048E-1	-1.1381E0	-6.6574E-1
8		MOSA	9.6376E-1	9.6316E-1	9.8944E-2	7.3381E-1	1.1765E0
9		MOSA	-7.1254E-1	-7.1352E-1	1.0014E-1	-9.9572E-1	-5.0243E-1
10		MOSA	1.2269E0	1.2269E0	9.8055E-2	9.9939E-1	1.4500E0
11		MOSA	-2.5923E0	-2.5908E0	1.0194E-1	-2.8269E0	-2.2865E0
12		MOSA	-6.5633E-2	-6.4452E-2	1.0162E-1	-2.9015E-1	2.1825E-1
13		MOSA	-1.5843E0	-1.5837E0	1.0175E-1	-1.8122E0	-1.2958E0
14		MOSA	-1.1572E0	-1.1575E0	9.7642E-2	-1.3895E0	-9.4423E-1
15		MOSA	-2.4064E0	-2.4063E0	1.0678E-1	-2.6871E0	-2.0738E0
16		MOSA	9.4492E-1	9.4381E-1	1.0165E-1	6.7092E-1	1.1730E0
17		MOSA	-2.9999E0	-3.0001E0	9.6939E-2	-3.2203E0	-2.7905E0
18		MOSA	1.9731E0	1.9737E0	9.8923E-2	1.7441E0	2.2054E0
19		MOSA	-9.2043E-1	-9.2015E-1	9.8361E-2	-1.1571E0	-7.0252E-1
20		MOSA	-2.9290E0	-2.9273E0	1.0499E-1	-3.1850E0	-2.5921E0
21		MOSA	-1.9033E0	-1.9026E0	9.7350E-2	-2.1215E0	-1.6742E0
22		MOSA	1.8701E0	1.8729E0	1.0581E-1	1.6476E0	2.2290E0
23		MOSA	-1.9793E0	-1.9800E0	9.7871E-2	-2.2031E0	-1.7575E0
24		MOSA	-7.6969E-1	-7.7081E-1	9.9897E-2	-1.0405E0	-5.5611E-1
25		MOSA	-6.9244E-1	-6.9322E-1	1.0063E-1	-9.5270E-1	-4.5608E-1
26		MOSA	-1.6829E0	-1.6842E0	1.0252E-1	-1.9831E0	-1.4424E0
27		MOSA	-1.3852E0	-1.3856E0	1.0606E-1	-1.6946E0	-1.1077E0
28		MOSA	2.0377E0	2.0367E0	1.0046E-1	1.8052E0	2.2823E0
29		MOSA	-2.8724E0	-2.8739E0	9.8915E-2	-3.1260E0	-2.6628E0
30		MOSA	-7.3169E-2	-7.4143E-2	1.0080E-1	-3.3782E-1	1.5002E-1
31		MOSA	-2.1784E0	-2.1791E0	9.7083E-2	-2.4099E0	-1.9661E0
32		MOSA	-2.5641E0	-2.5631E0	1.0363E-1	-2.8189E0	-2.2566E0
33		MOSA	-2.6652E0	-2.6666E0	9.7680E-2	-2.9107E0	-2.4597E0
34		MOSA	-2.2312E0	-2.2329E0	1.0132E-1	-2.5023E0	-2.0097E0
35		MOSA	-2.9635E0	-2.9634E0	1.0190E-1	-3.2229E0	-2.7118E0
36		MOSA	-5.4108E-1	-5.4102E-1	9.6563E-2	-7.4562E-1	-3.1204E-1
37		MOSA	-8.7332E-1	-8.7305E-1	9.8619E-2	-1.0995E0	-6.3129E-1
38		MOSA	-8.0109E-1	-7.9996E-1	1.0037E-1	-1.0112E0	-5.3060E-1
39		MOSA	-1.0812E0	-1.0812E0	9.9387E-2	-1.3101E0	-8.2346E-1
40		MOSA	2.6407E0	2.6409E0	9.7123E-2	2.4341E0	2.8566E0

Designs Table

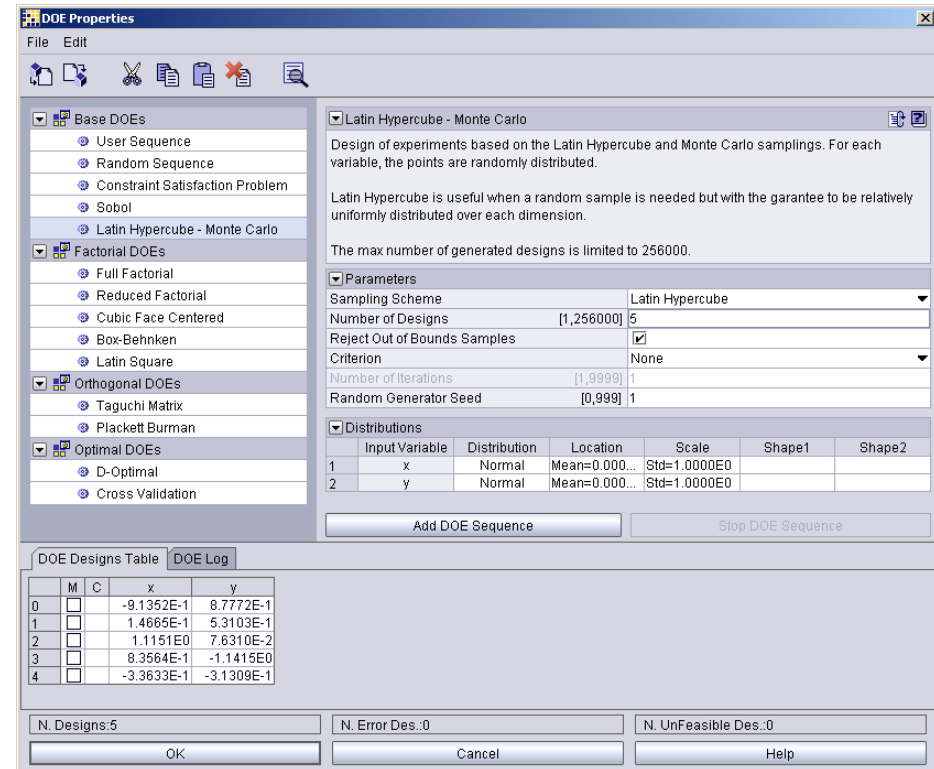
ID	RID	M	CATEGORY	x	y	o1
27	0		MOSA	-3.8952E-2	1.2106E-1	7.4742E0
28	0		MOSA	-1.0468E-1	1.1018E-1	7.3680E0
29	0		MOSA	-2.7332E-2	-6.8912E-3	7.7574E0
30	0		MOSA	-8.3579E-2	-1.7987E-1	7.9589E0
31	0		MOSA	2.5451E-2	-2.1581E-1	8.2394E0
32	0		MOSA	-6.2448E-2	-1.2340E-1	7.9047E0
33	0		MOSA	-9.4790E-2	2.1770E-2	7.5666E0
34	0		MOSA	-1.4197E-1	-8.6856E-2	7.6731E0
35	0		MOSA	3.4844E-3	-5.8185E-2	7.9162E0
36	0		MOSA	-4.2985E-2	-1.3943E-1	7.9727E0
37	0		MOSA	-1.6251E-2	-1.6224E-1	8.0662E0
38	0		MOSA	-1.6338E-1	1.9151E-1	7.0783E0
39	0		MOSA	1.8382E-2	1.0749E-2	7.8105E0
40	0		MOSA	-5.1267E-2	1.0548E-1	7.4832E0
41	0		MOSA	-6.0601E-2	-9.5152E-2	7.8576E0
42	0		MOSA	-1.3434E-3	5.8625E-2	7.6762E0
43	0		MOSA	8.2140E-2	4.6723E-2	7.8543E0
44	0		MOSA	3.6286E-2	-1.8812E-3	7.8693E0
45	0		MOSA	-3.0780E-2	-3.1361E-2	7.7981E0
46	0		MOSA	1.0617E-1	7.1708E-2	7.8442E0
47	0		MOSA	4.4328E-2	9.2644E-2	7.3264E0
48	0		MOSA	-6.8932E-2	3.0034E-2	7.6023E0
49	0		MOSA	5.9379E-2	-1.4092E-2	7.9363E0
50	1		MOSA	-1.5698E0	-1.5698E0	9.9435E0
51	1		MOSA	-1.5603E0	-1.5554E0	9.8716E0
52	1		MOSA	-1.5791E0	-1.6211E0	1.0146E1
53	1		MOSA	-1.6530E0	-1.5722E0	1.0073E1
54	1		MOSA	-1.4867E0	-1.5833E0	9.8278E0
55	1		MOSA	-1.5088E0	-1.6149E0	9.9813E0
56	1		MOSA	-1.4335E0	-1.4638E0	9.2802E0
57	1		MOSA	-1.6109E0	-1.5881E0	1.0080E1
58	1		MOSA	-1.5538E0	-1.4833E0	9.5631E0
59	1		MOSA	-1.5976E0	-1.6443E0	1.0258E1
60	1		MOSA	-1.8356E0	-1.7238E0	1.0718E1
61	1		MOSA	-1.6482E0	-1.6056E0	1.0199E1
62	1		MOSA	-1.7129E0	-1.4563E0	9.5796E0
63	1		MOSA	-1.5251E0	-1.5622E0	9.8328E0
64	1		MOSA	-1.6074E0	-1.6663E0	1.0345E1
65	1		MOSA	-1.7256E0	-1.7260E0	1.0687E1
66	1		MOSA	-1.4595E0	-1.6314E0	9.9142E0
67	1		MOSA	-1.4449E0	-1.6373E0	9.8934E0

- Robust design table contains stochastic values for each design

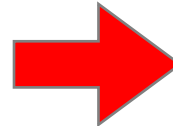
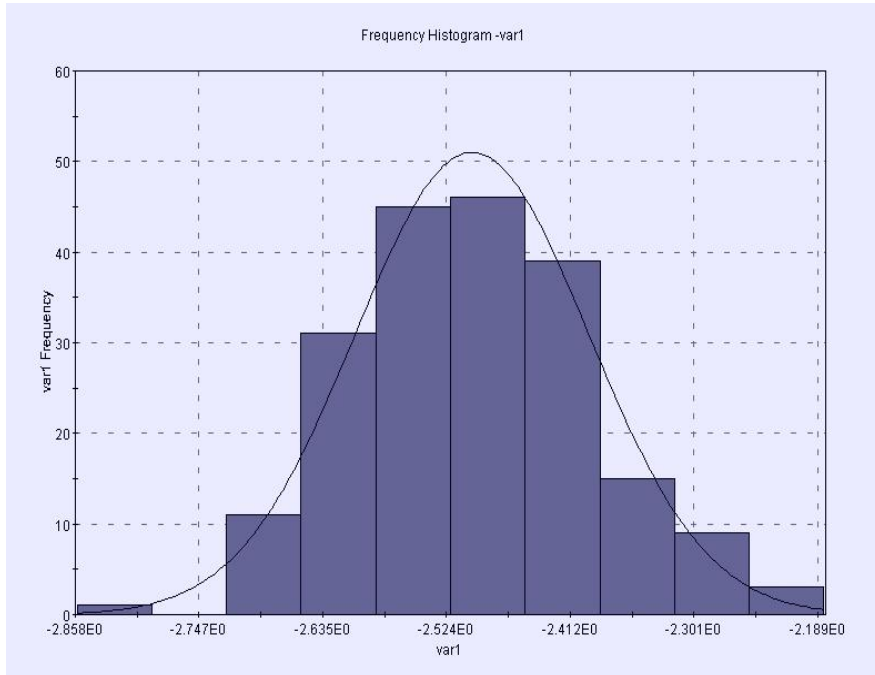
- Design Table reports values for each samples (ID number) of the corresponding design (RID number)

# Stability control

- The most useful DOE algorithm for stability control is Monte Carlo / Latin Hypercube
- Several different distributions can be defined for input parameters: Exponential, Gamma, Logistic, Lognormal, Normal, Student, Uniform, Weibull
- Monte Carlo perturbations around a point look for robust solutions that are not influenced by small variations of the design variables

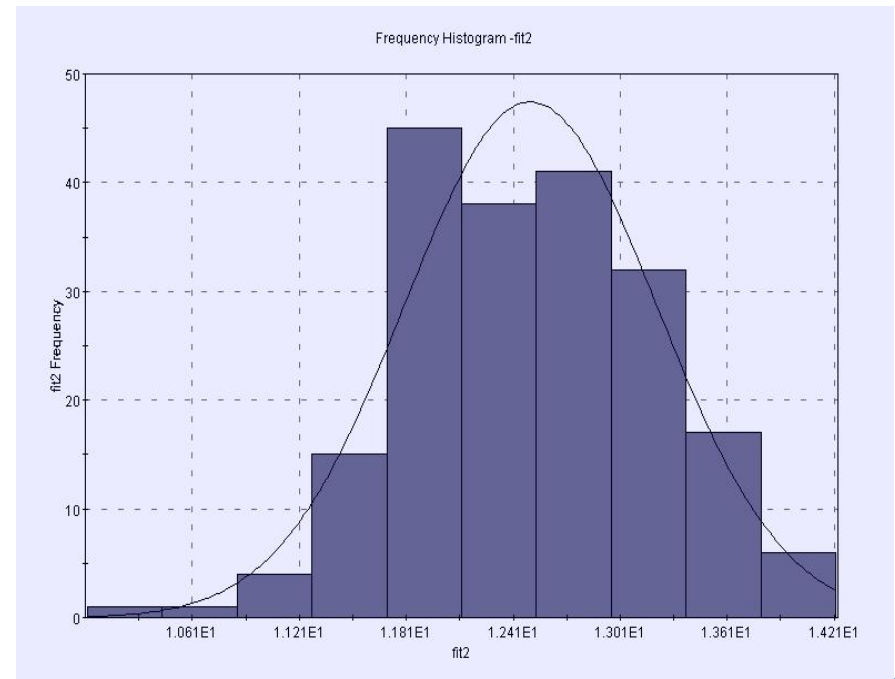
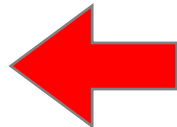


# Monte Carlo DOE

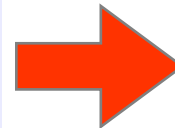
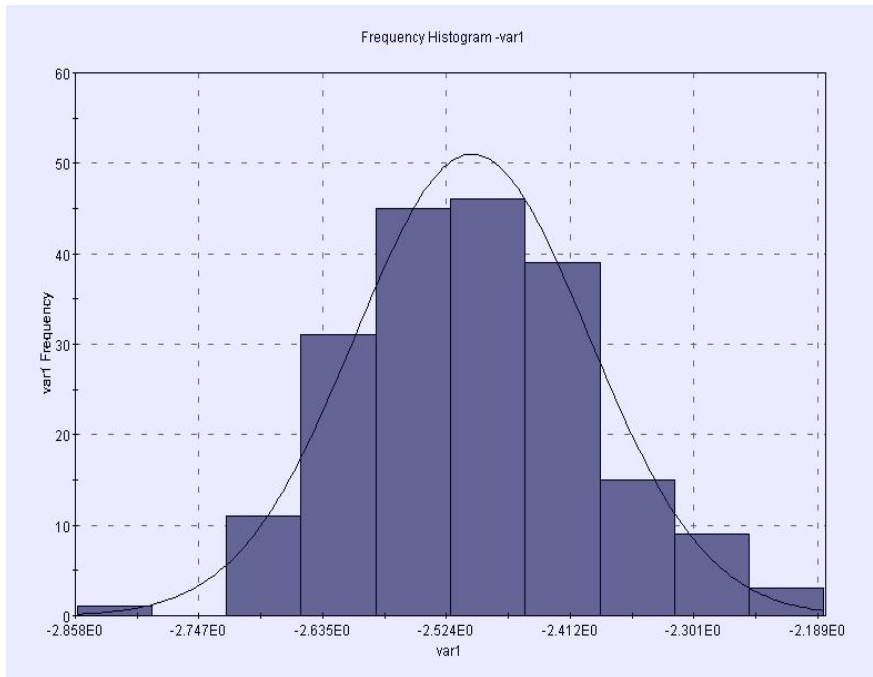


Monte Carlo Perturbation  
for input variables

Frequency Histogram:  
The output variable is not  
influenced by small variation  
of the design variables

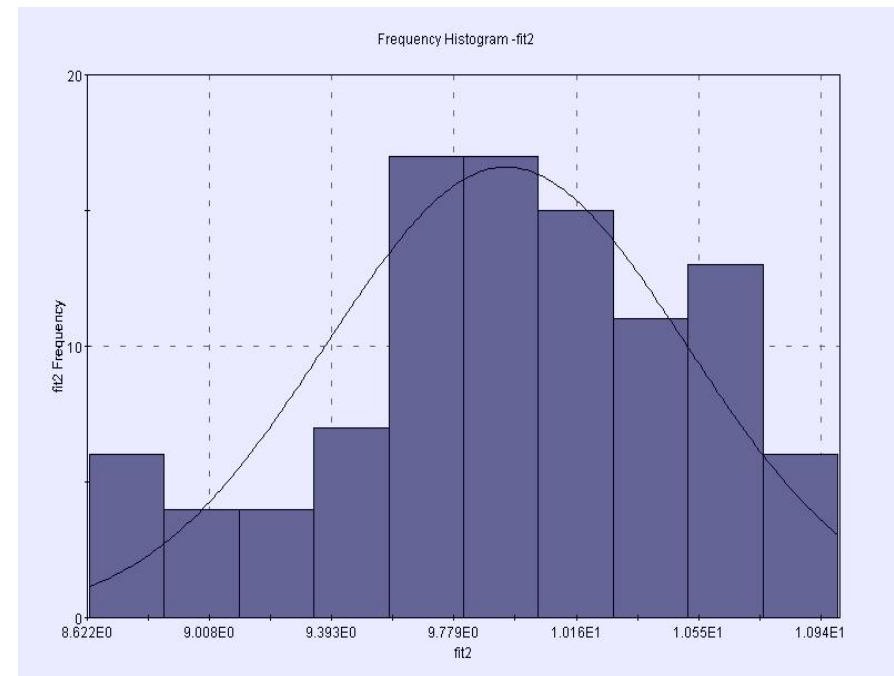
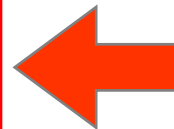


# Monte Carlo DOE



Monte Carlo Perturbation  
for input variables

Frequency Histogram:  
The output variable is  
influenced by small variation  
of the design variables



# Exercise 2

---



-modeFRONTIER® is a registered product of [ESTECO srl](#)  
-Copyright © ESTECO srl 1999-2007



-For more information visit:  
-[www.esteco.com](http://www.esteco.com) or send an e-mail to:  
[modeFRONTIER@esteco.com](mailto:modeFRONTIER@esteco.com)

**modeFRONTIER**  
the multi-objective optimization and design environment



# Maximize a Mathematical function

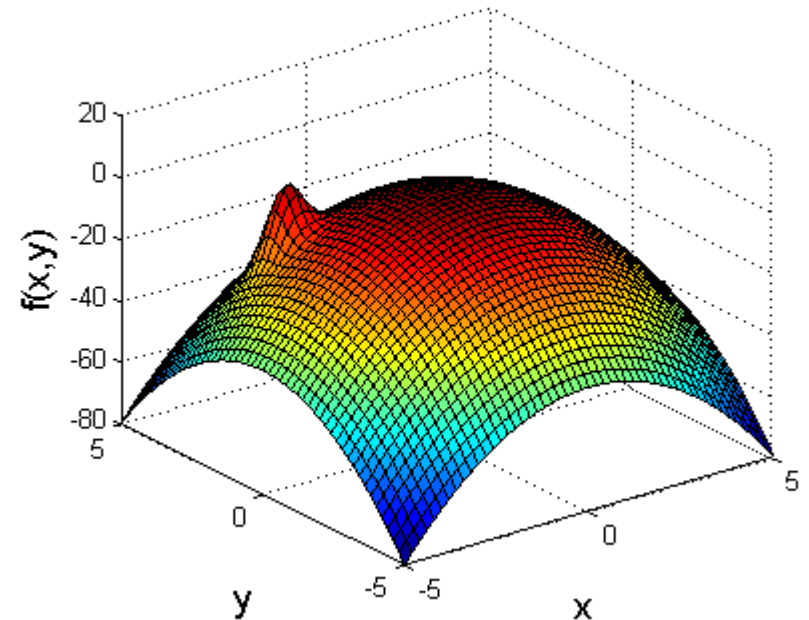
Maximize:

$$F(x, y) = 20 \exp\left(-\frac{\alpha}{2\sigma^2}\right) - 1.6(x^2 + y^2)$$

$$\sigma = 0.4$$

$$\alpha = (x + 2.5)^2 + (y - 2.5)^2$$

$$x, y \in [-5.0, 5.0]$$





# Maximize a Mathematical function

**Global Maximum in:**

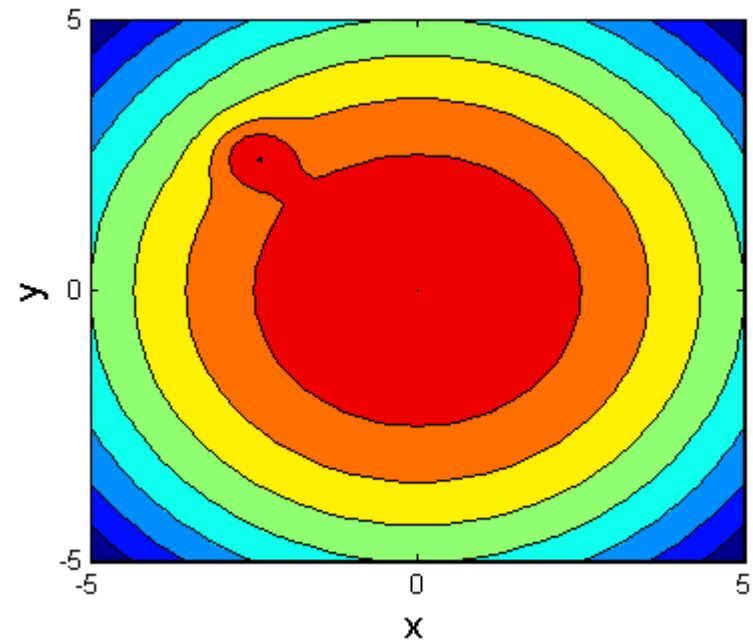
$$(x, y) \approx (-2.4360, 2.4360)$$

$$f(x, y) \approx 0.5054$$

**Robust Maximum in:**

$$(x, y) \approx (0, 0)$$

$$f(x, y) \approx 0$$



# Stability control

Number of samples

Latin Hypercube - Monte Carlo

Design of experiments based on the Latin Hypercube and Monte Carlo samplings. For each variable, the points are randomly distributed.

Latin Hypercube is useful when a random sample is needed but with the guarantee to be relatively uniformly distributed over each dimension.

The max number of generated designs is limited to 256000.

Parameters

Sampling Scheme	Latin Hypercube
Number of Designs	[1,256000] 100
Reject Out of Bounds Samples	<input checked="" type="checkbox"/>
Criterion	Maximize Minimum Distance
Number of Iterations	[1,9999] 10
Random Generator Seed	[0,999] 1

Distributions

	Input Variable	Distribution	Location	Scale	Shape1	Shape2
1	x	Normal	Mean=0.0000E0	Std=1.0000E-1		
2	y	Normal	Mean=0.0000E0	Std=1.0000E-1		

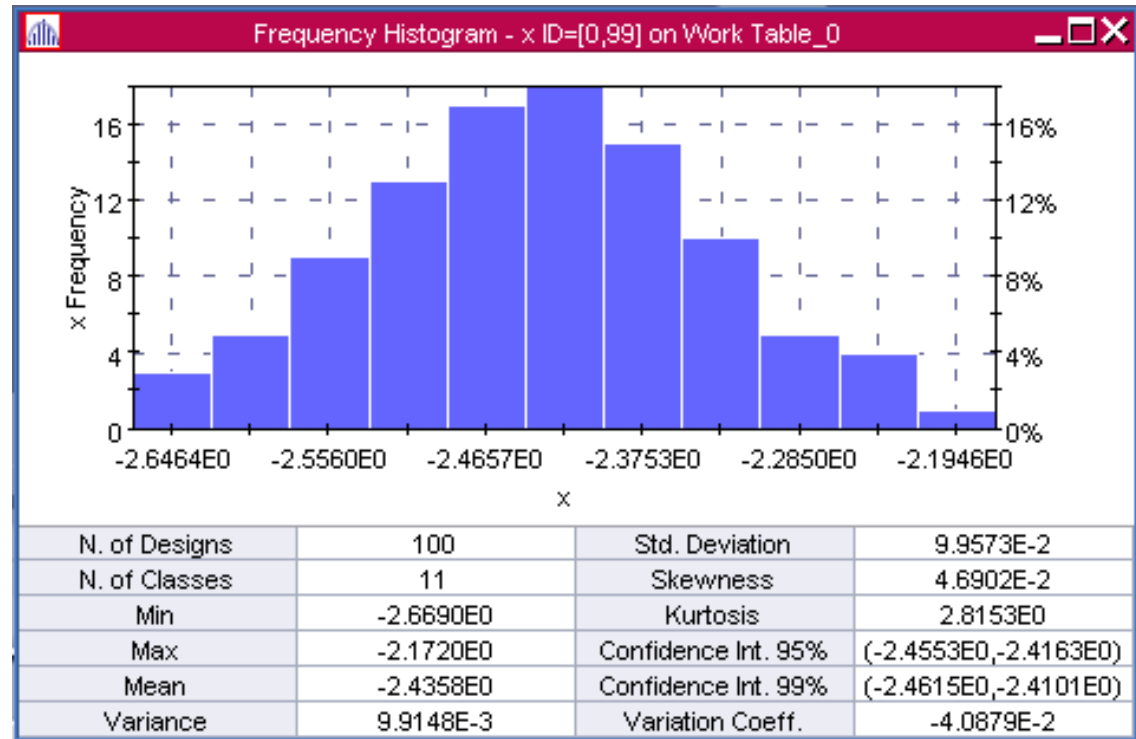
Distribution's properties

- The most useful DOE algorithm for stability control is Latin Hypercube-Monte Carlo
- Several different distributions are available: Exponential, Gamma, Logistic, Lognormal, Normal, Student, Uniform, Weibull
- Perturbations around a point look for robust solutions that are not influenced by small variations of the design variables

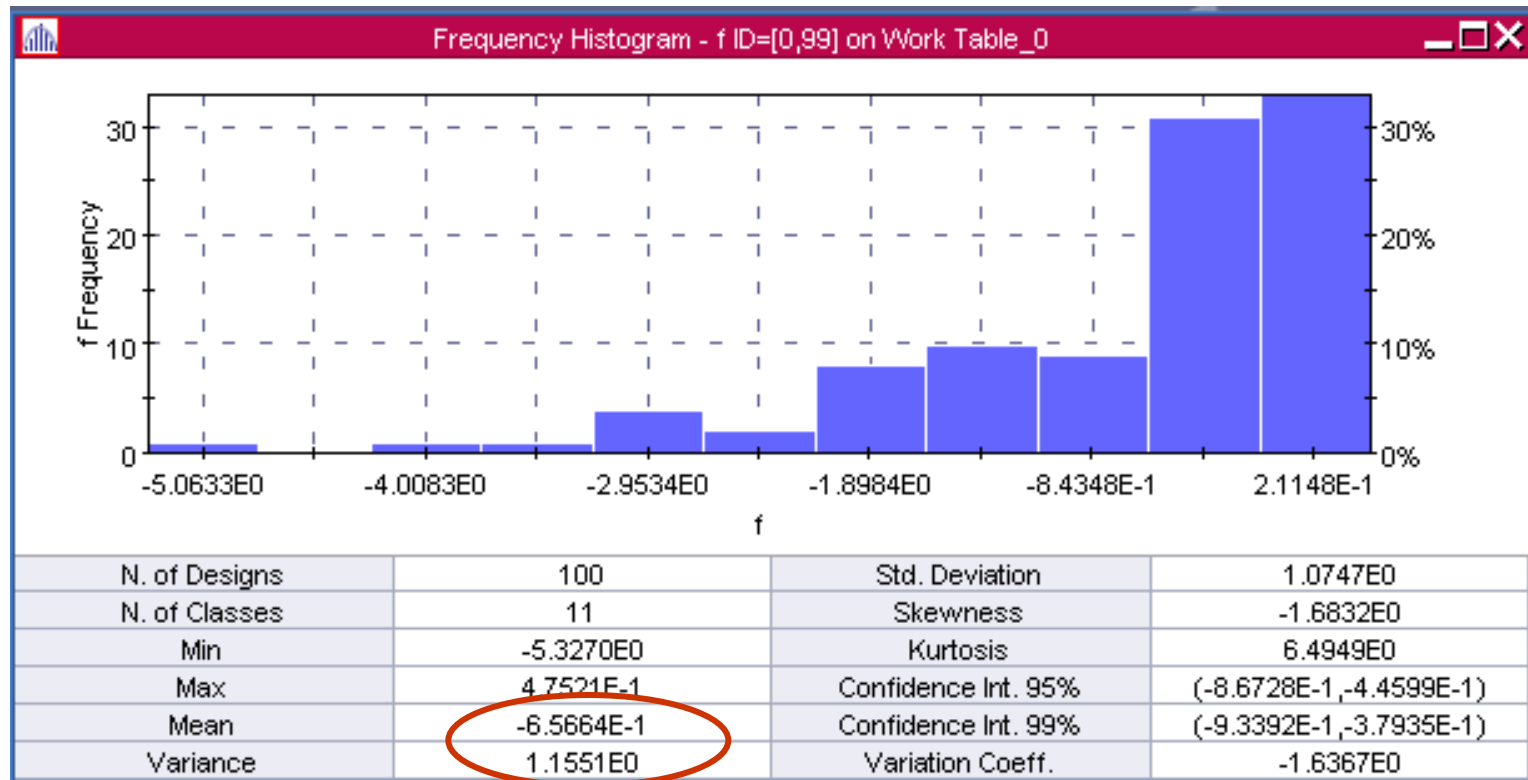


# Latin Hypercube - Monte Carlo

- Frequency chart for the input variable  $x$
- Normal distribution with 100 samples using Latin Hypercube
- The table reports the moments of the distribution (mean, variance, skewness, kurtosis and confidence intervals)



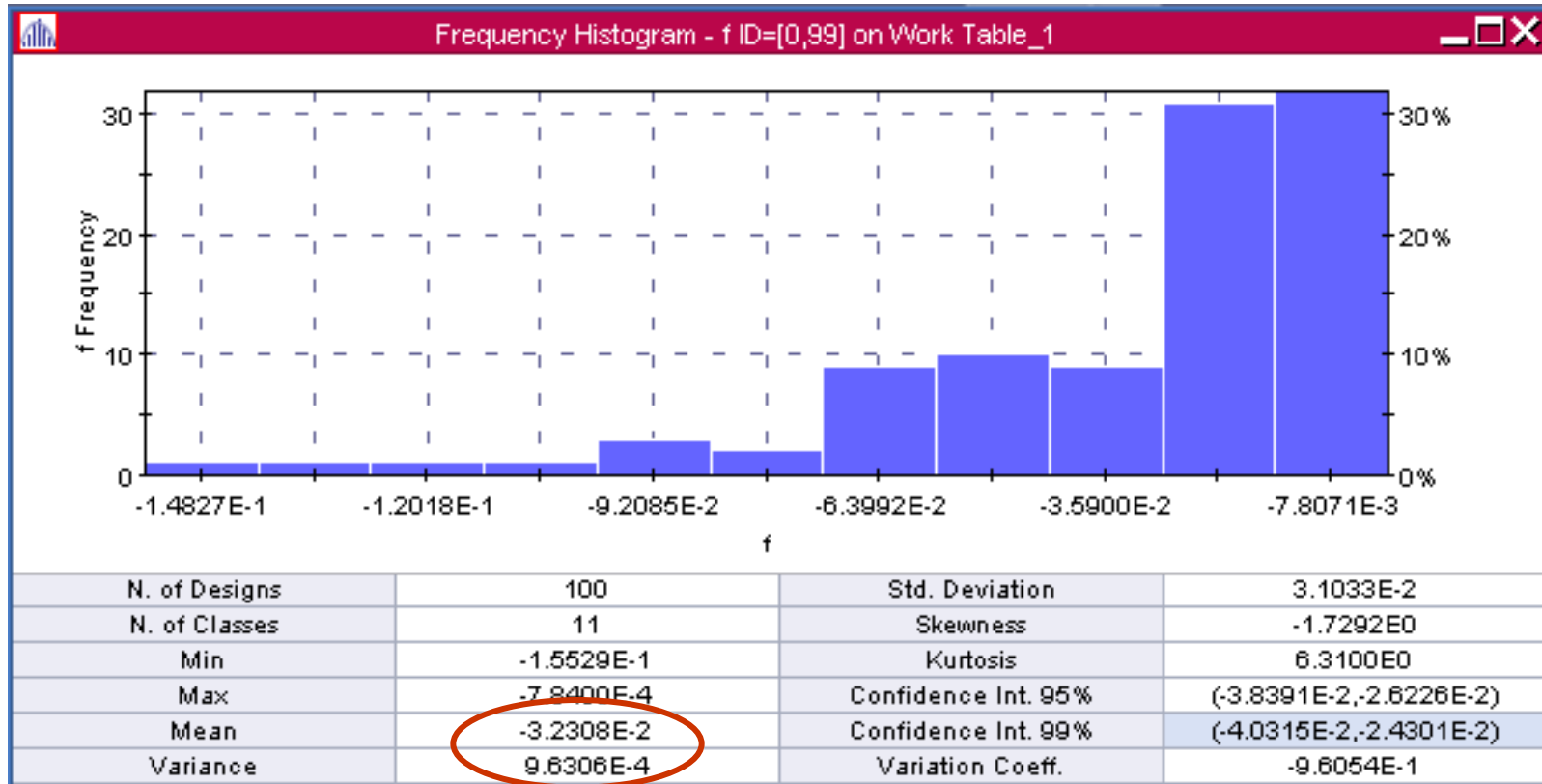
# Latin Hypercube - Monte Carlo



- Frequency chart for the output variable for the non-robust maximum
- Values are in the range [-5.33,0.48]
- The distribution mean is -0.66, the variance is 1.15



# Latin Hypercube - Monte Carlo



- Frequency chart for the output variable for the robust maximum
- Values are in the range  $[-1.5529e-1, -7.8400e-4]$
- The distribution mean is  $-3.23e-2$ , the variance is  $-3.23e-2$



# Monte Carlo DOE

---

The Frequency Histogram shows if the output variable is **influenced** by small variation of the design variables.

	<b>Mean</b>	<b>Variance</b>
<b>Global maximum</b>	-0.66	1.15
<b>Robust maximum</b>	-3.23e-2	-3.23e-2



# References

---

- [1] R Askey and J Wilson 1985 Some basic hypergeometric polynomials that generalize Jacobi polynomials *Memoirs Amer. Math. Soc.* AMS Providence RI 319
- [2] R Ghanem and J Red{Horse 1999 Propagation of probabilistic uncertainty in complex physical systems using a stochastic finite element approach *Phisica D* 133 137{144
- [3] R Ghanem and P Spanos 1991 The stochastic nite element method: a spectral approach Springer
- [4] Xiu D and Karniadakis G 2002 The Wiener{Askey polynomial chaos for stochastic differential equations *SIAM J. Sci. Comput.* 24(2) 619-644
- [5] M D McKay, R J Beckmkan and W J Conover 1979 A comparison of three methods for selecting values of input variables in the analysis of output from a computer code *Technometrics* 21 239-245
- [6] N Wiener 1938 The homogeneous chaos *Amer. J. Math.* 60 897-936

