Robust Design and Uncertainty Quantification

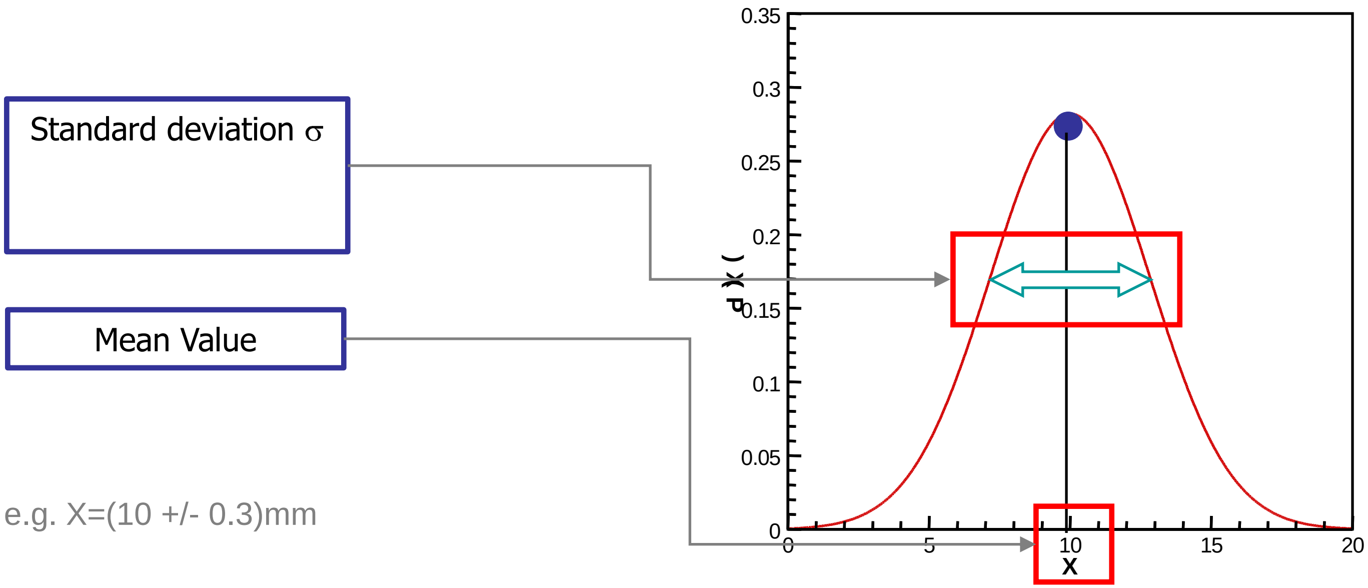
Monte Carlo, Latin hypercube and Polynomial Chaos

-For more information visit:
-www.esteco.com or send an e-mail to:
modeFRONTIER@esteco.com

Why robust designs optimizations?

- In many engineering problems **design parameters** may be **uncertain** (e.g. tolerances, fluctuating operating conditions, etc.)

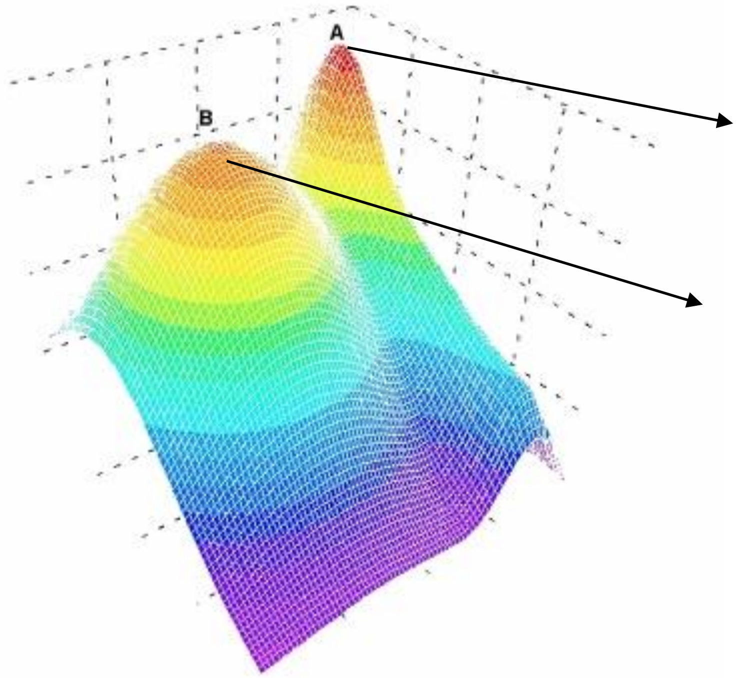
- Input parameters are then defined not by a deterministic value, but as a **Distribution** (each value have a statistical probability to occur)



e.g. X=(10 +/- 0.3)mm

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:

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

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

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

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

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

e. g.:

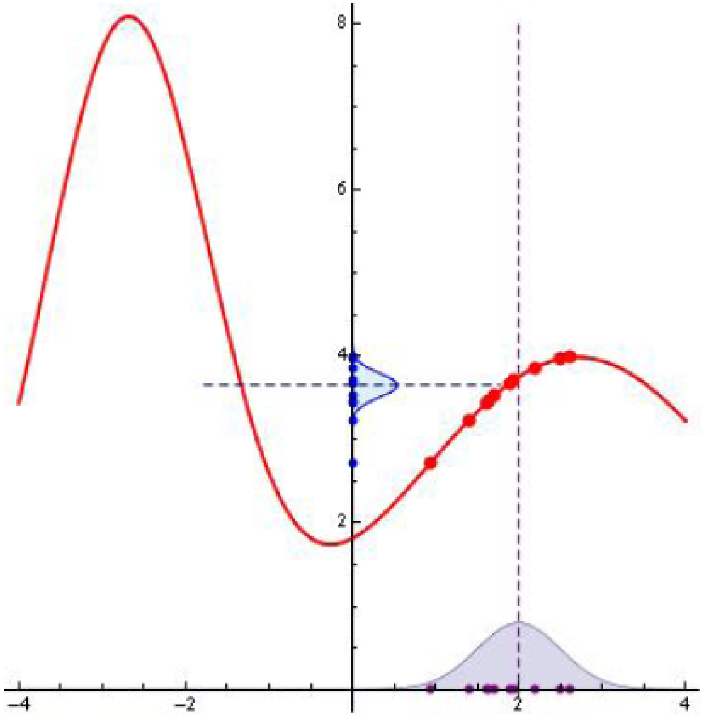
$$X_i = X_i(x_i) \quad \text{is Normal centered in } x_i \text{ with fixed } \sigma_i$$

A stochastic MOP

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

$$f(x_1, \ldots, x_n) \longrightarrow \underbrace{F(x_1, \ldots, x_n)}_{\text{random variable}},$$

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



Stochastic MOP

Simply rewriting

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

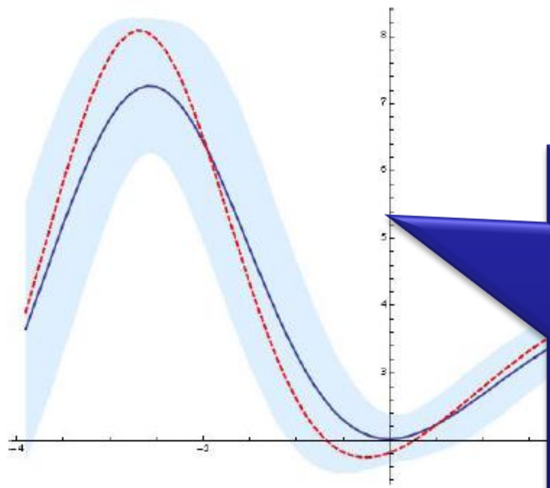
does not make sense.

We could consider, for instance,

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

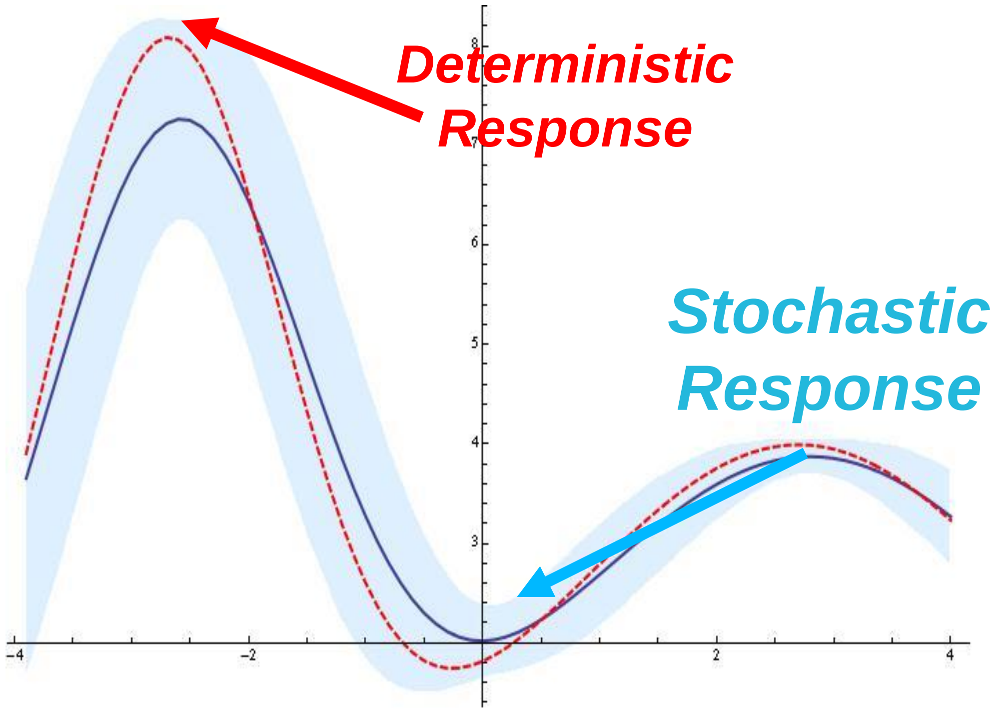
plainly, or more sophisticatedly,

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



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 $X_1,\ldots,X_k$.
- 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

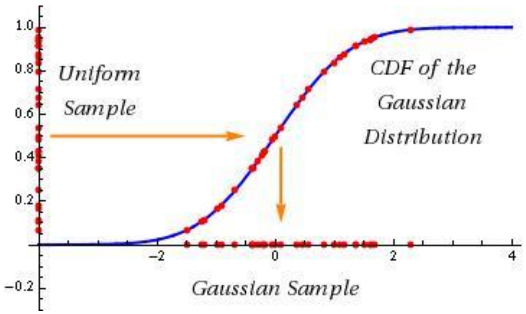
The most classical methodology consists in drawing a random sample $\{x^{(1)}, \ldots, x^{(N)}\}$ from the joint distribution $\mathcal{D}_1 \otimes \cdots \otimes \mathcal{D}_d$, i.e., drawing N random numbers $\left\{x_i^{(1)}, \ldots, x_i^{(N)}\right\}$ from every distribution $\mathcal{D}_i$ and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

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

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

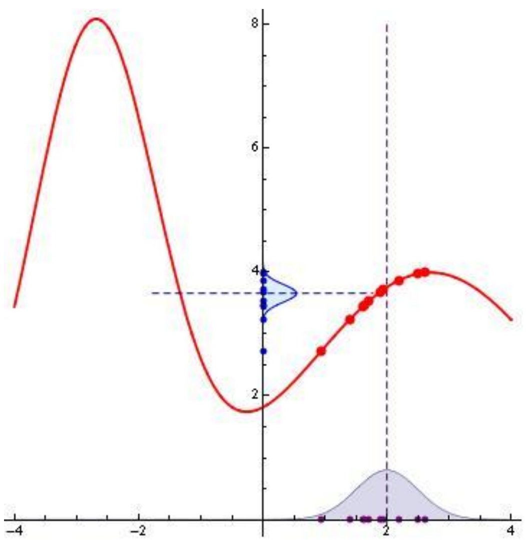
$$y^{(j)} := f\left(x_1^{(j)}, \ldots, x_d^{(j)}\right), \qquad j = 1, \ldots, N. \tag{2}$$

Monte Carlo



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

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



- 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

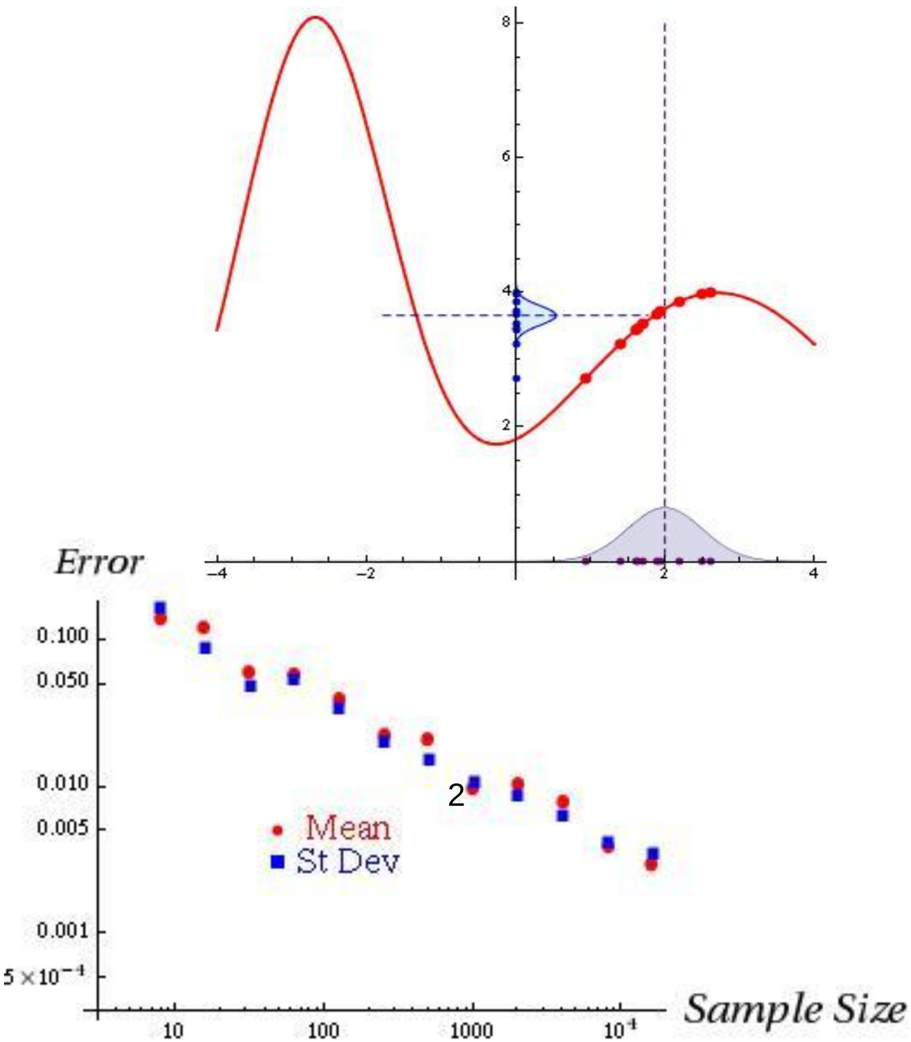
$$f(x) := 20\left(\frac{e^{-\frac{(x+2.7)^2}{2}}}{\sqrt{2\pi}} + \frac{e^{-\frac{(x-2.7)^2}{2(2^2)}}}{\sqrt{2\pi}2}\right),$$

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

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

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

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:

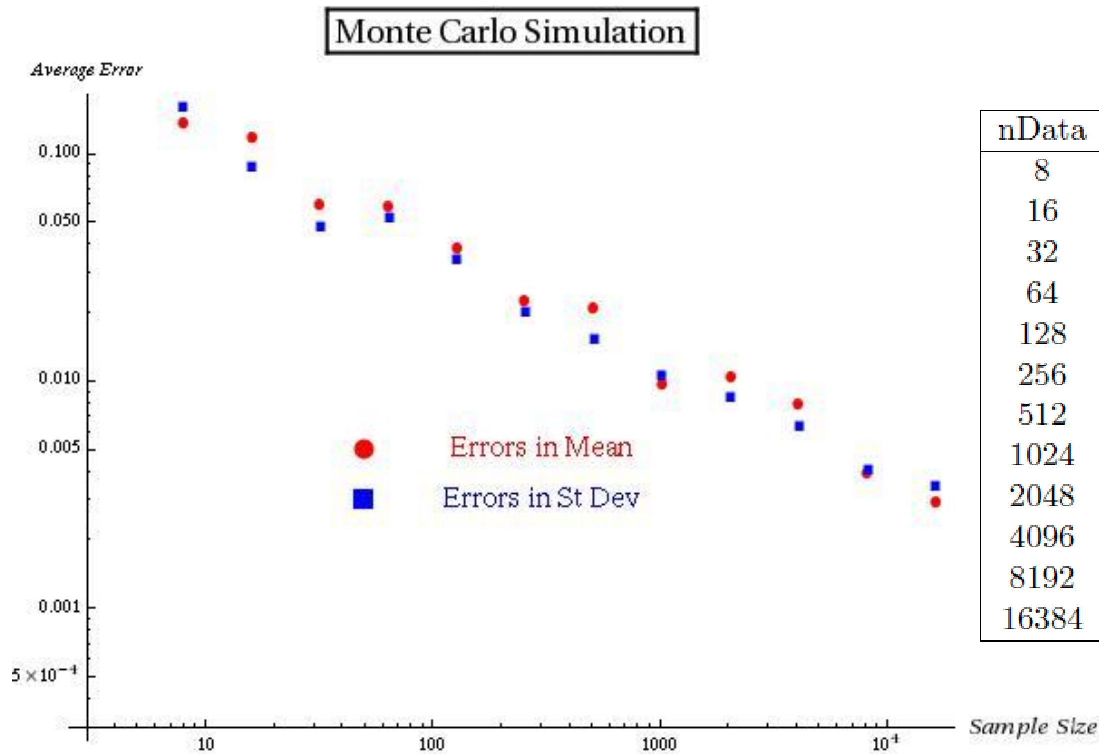
*to reach **1%** Error in*

Mean** needs **256 points

Standard Deviation

*needs **8192 points***

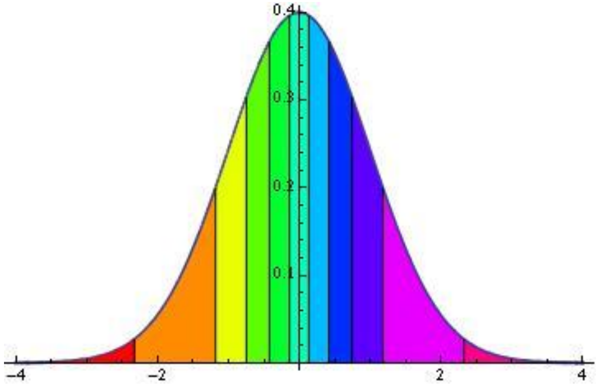
Monte Carlo Accuracy



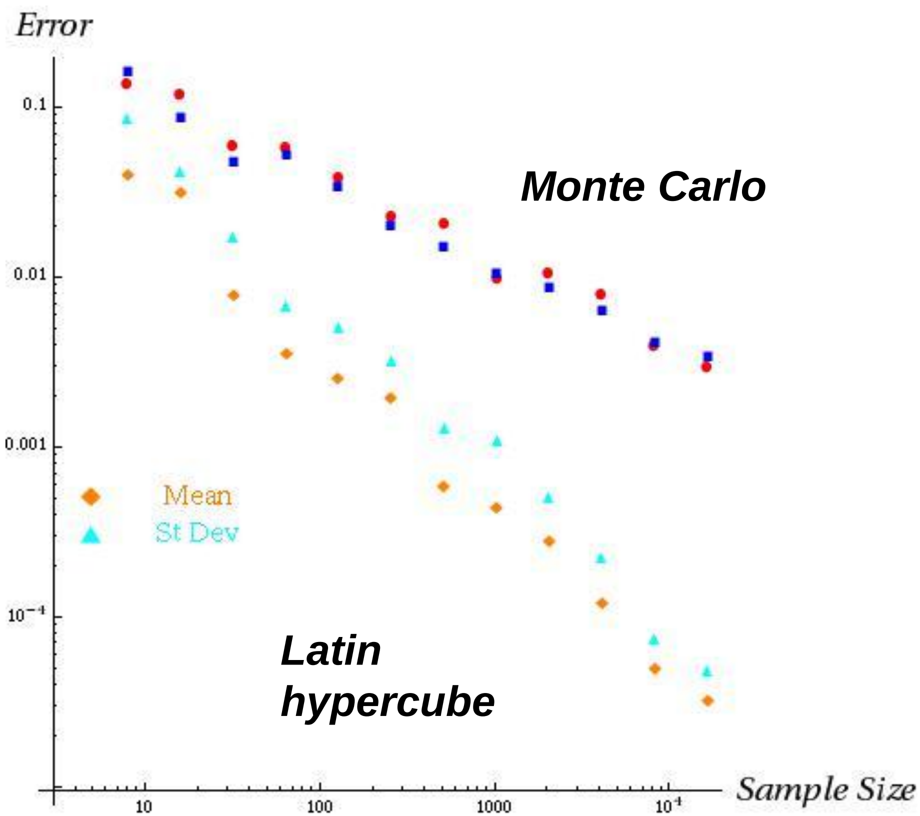
nData	Average Error Mean	Average Error StDev
8	0.13523321821576068	0.1610272359435758
16	0.11711965574510605	0.0864659032341357
32	0.059222020366550845	0.04713102966918838
64	0.057784528160956625	0.05253445820159186
128	0.03838430554585017	0.034136016860713006
256	0.022344036432755	0.020111546424191107
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

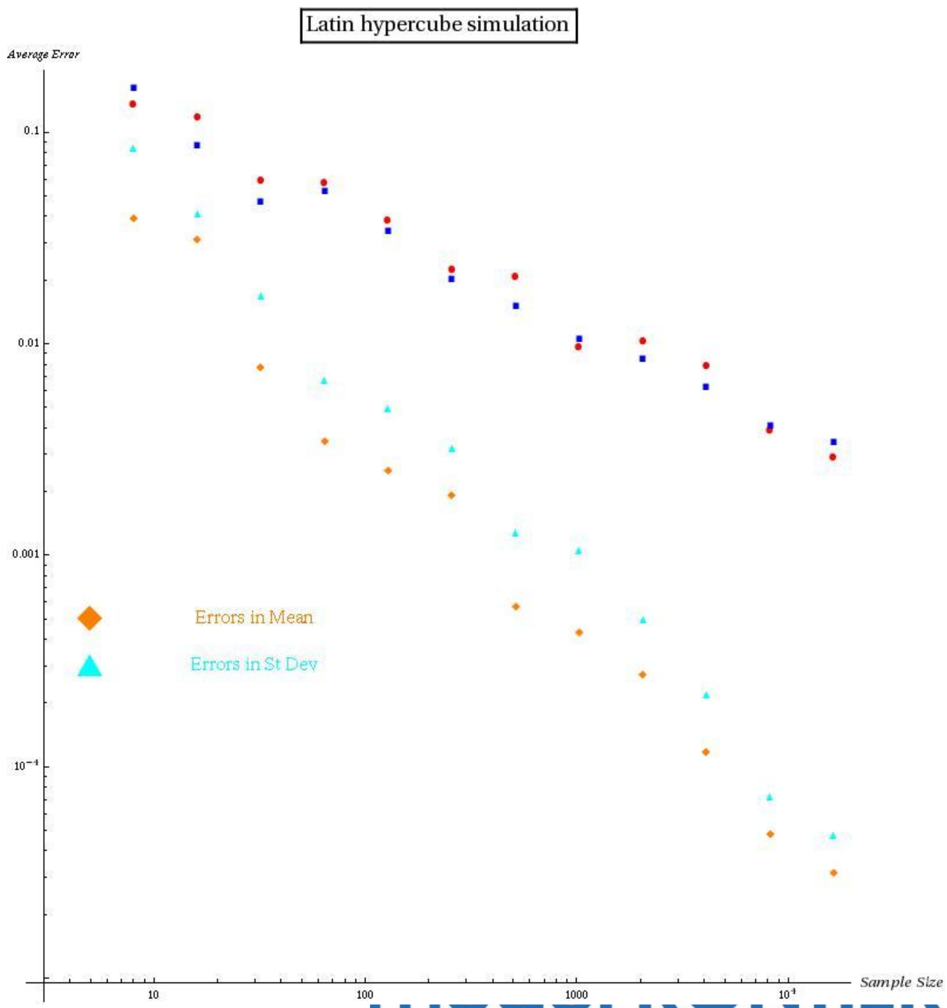
Standard Deviation

*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.

Polynomial degree

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

orthogonal polynomials

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

Polynomial Chaos Expansion

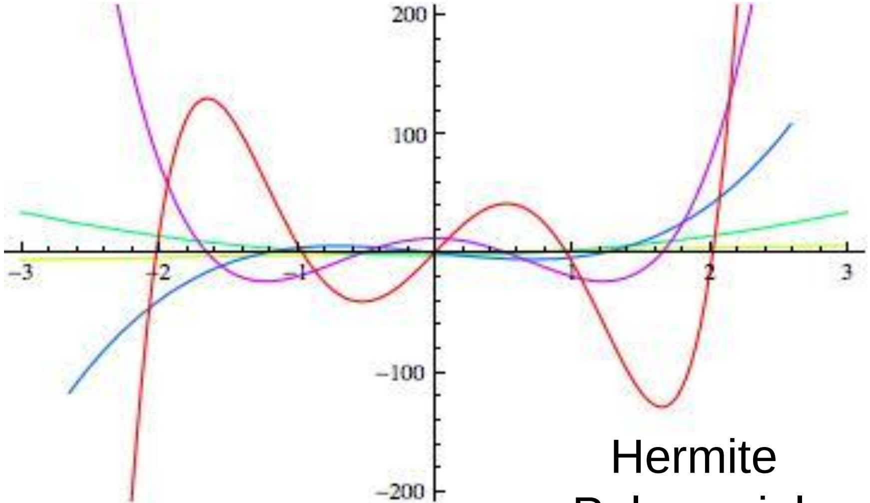
Alternatively, it is possible to expand the response in a special series, a ***Polynomial Chaos***.

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

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

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

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



Hermite Polynomials

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:

$$\langle y^m \rangle = \int (f(x))^m w(x)dx = \int \left(\sum_{i=0}^{k} \alpha_i p_i(x) \right)^m w(x)dx =$$

$$= \int \sum \alpha_{i_1} \dots \alpha_{i_m} p_{i_1}(x) \dots p_{i_m}(x) w(x)dx =$$

$$= \sum \alpha_{i_1} \dots \alpha_{i_m} \int p_{i_1}(x) \dots p_{i_m}(x) w(x)dx.$$

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)

$$\alpha_i : \qquad \min_{\alpha_i} \sum_{j=1}^{N} \left| f(\bar{x}_j) - \sum_{i=1}^{k} \alpha_i p_i(\bar{x}_j) \right|^2 ,$$

$$\{\bar{x}_1, \ldots, \bar{x}_N\}, \qquad \text{arbitrary sample}$$

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

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.

$N \geqslant \frac{(k+d)!}{k!d!}$	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$	$k=7$	$k=8$
$d=1$	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
$\vdots$								

Polynomial Chaos Accuracy

Table of results for the example

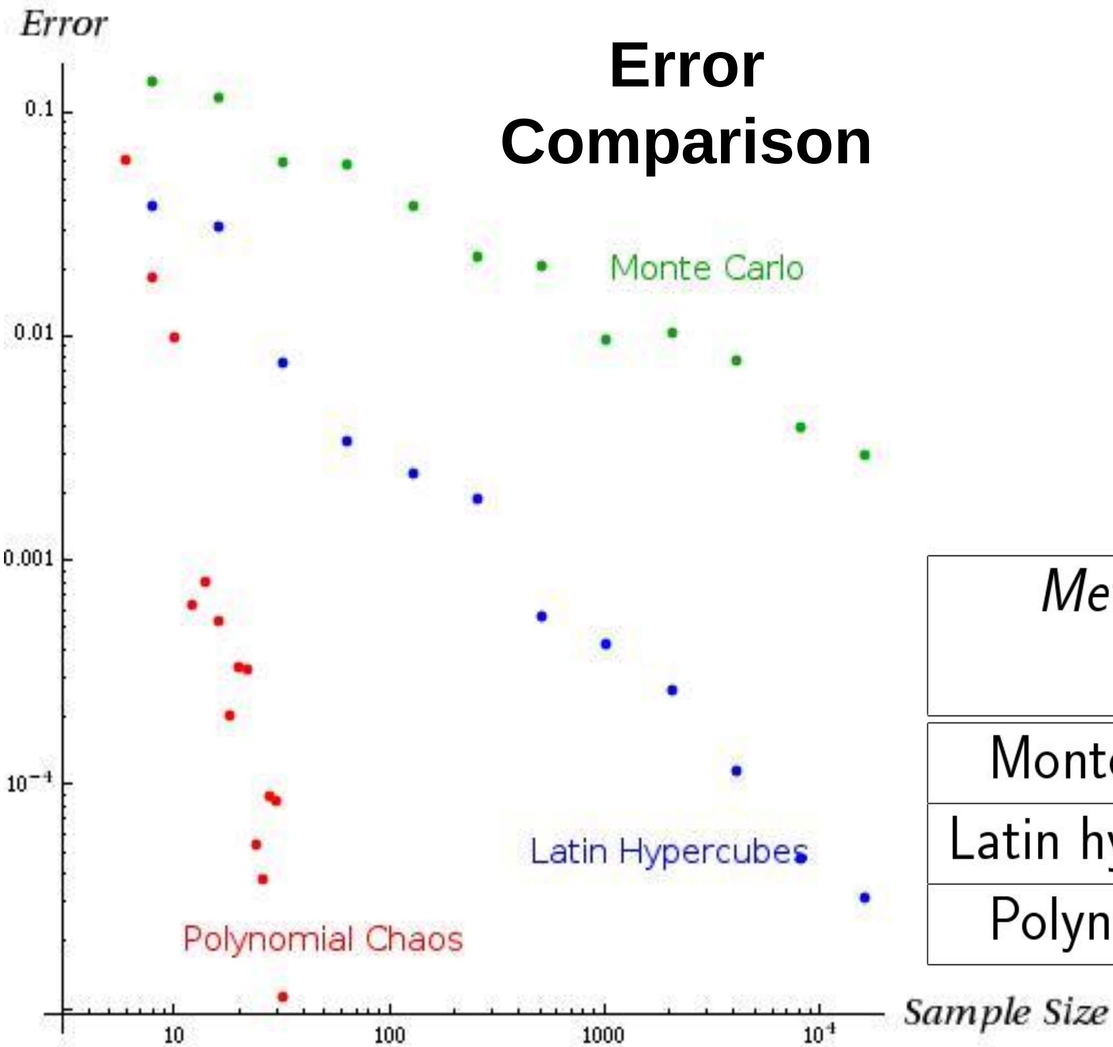
Chaos Order	Sample Size	Average Error Mean	Average Error StDev
1	4	0.09344069282015036	0.10429522822805484
2	6	0.019326055906163132	0.0697189377206368
3	8	0.011271788209210331	0.02654736033749695
4	10	8.03467578078676E-4	0.001684371972847451
5	12	7.877378937433344E-4	0.0028642320499342065
6	14	6.138461875910162E-4	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 / –($\simeq$ 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

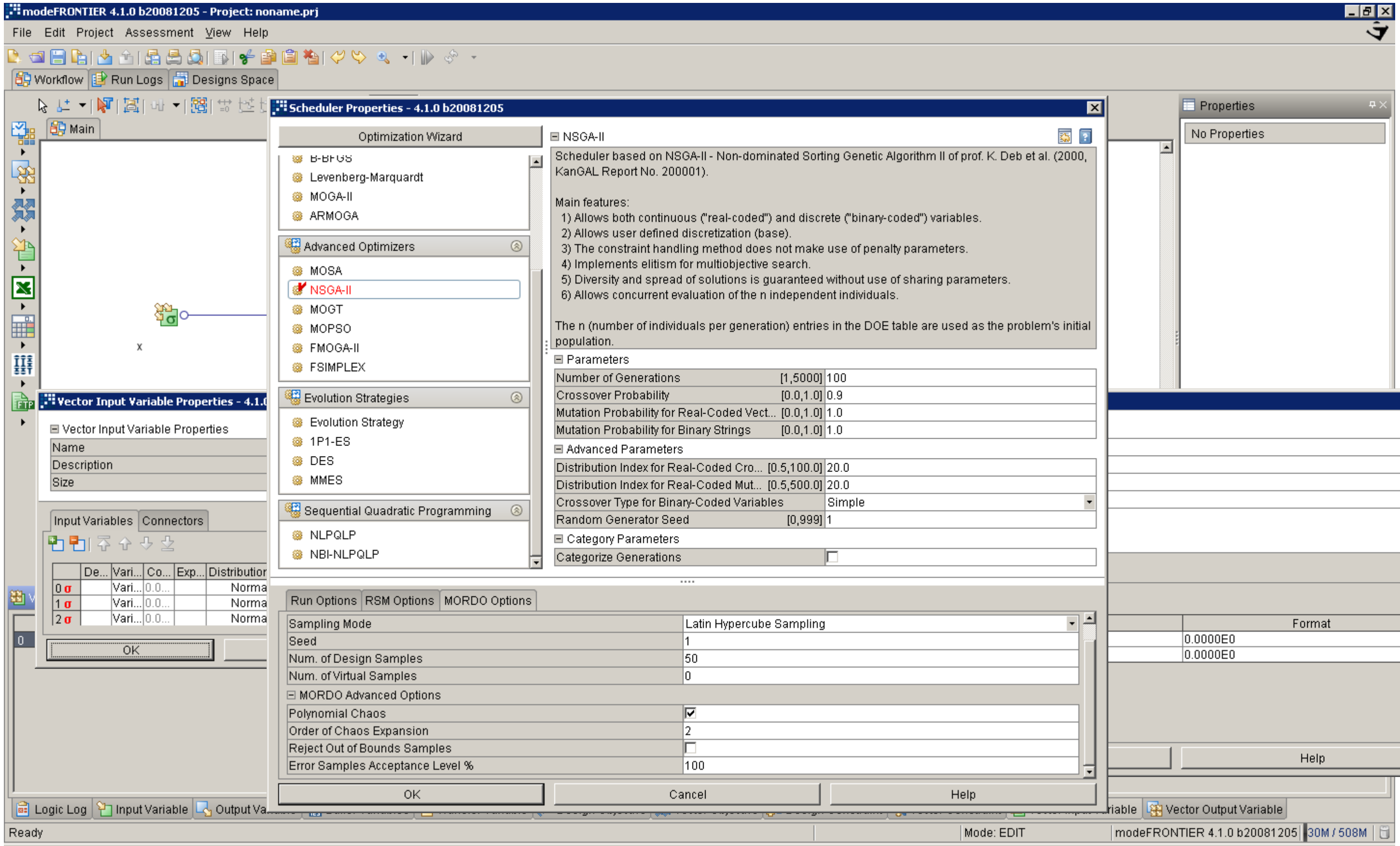


Former example:
*to reach **1% Error** in*
Mean** needs **8 points

Standard Deviation
*needs **12 points***

Method	*Mean* 1% error	*St Dev* 1% error
Monte Carlo	256	8192
Latin hypercube	16	128
Polyn Chaos	8	12

Polynomial Chaos in modeFRONTIER



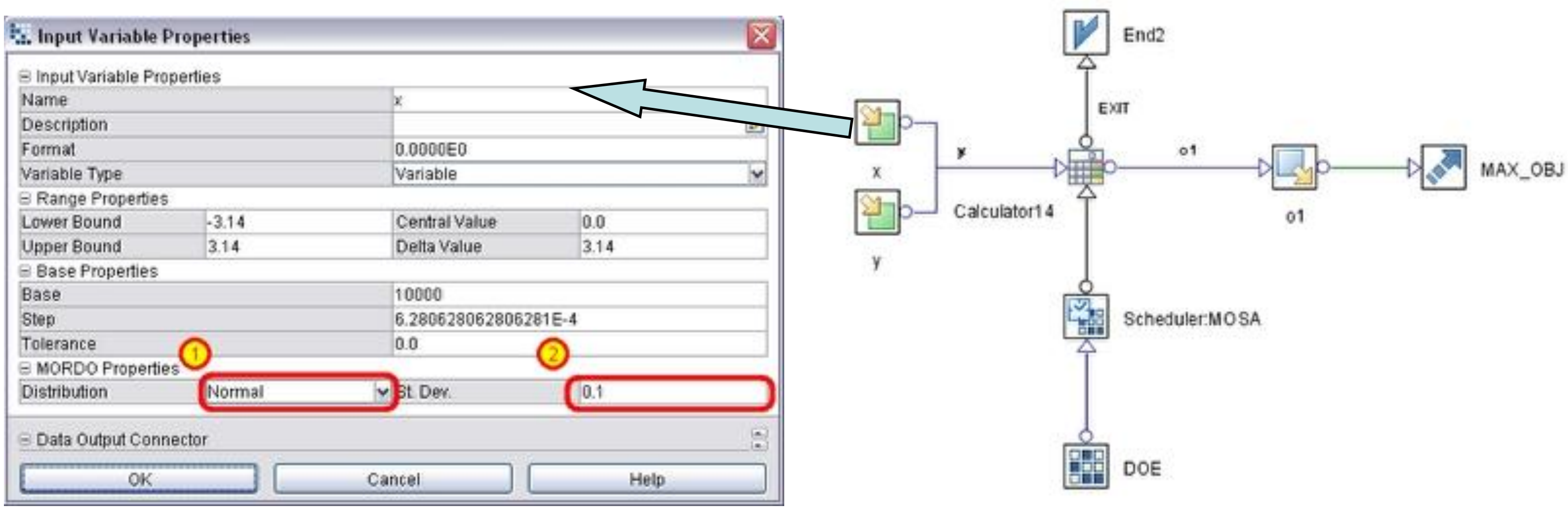
-modeFRONTIER® is a registered product of ESTECO srl
-Copyright © ESTECO srl 1999-2007

-For more information visit:
-www.esteco.com or send an e-mail to:
modeFRONTIER@esteco.com

modeFRONTIER
the multi-objective optimization and design environment

Create a stochastic project: variable distribution

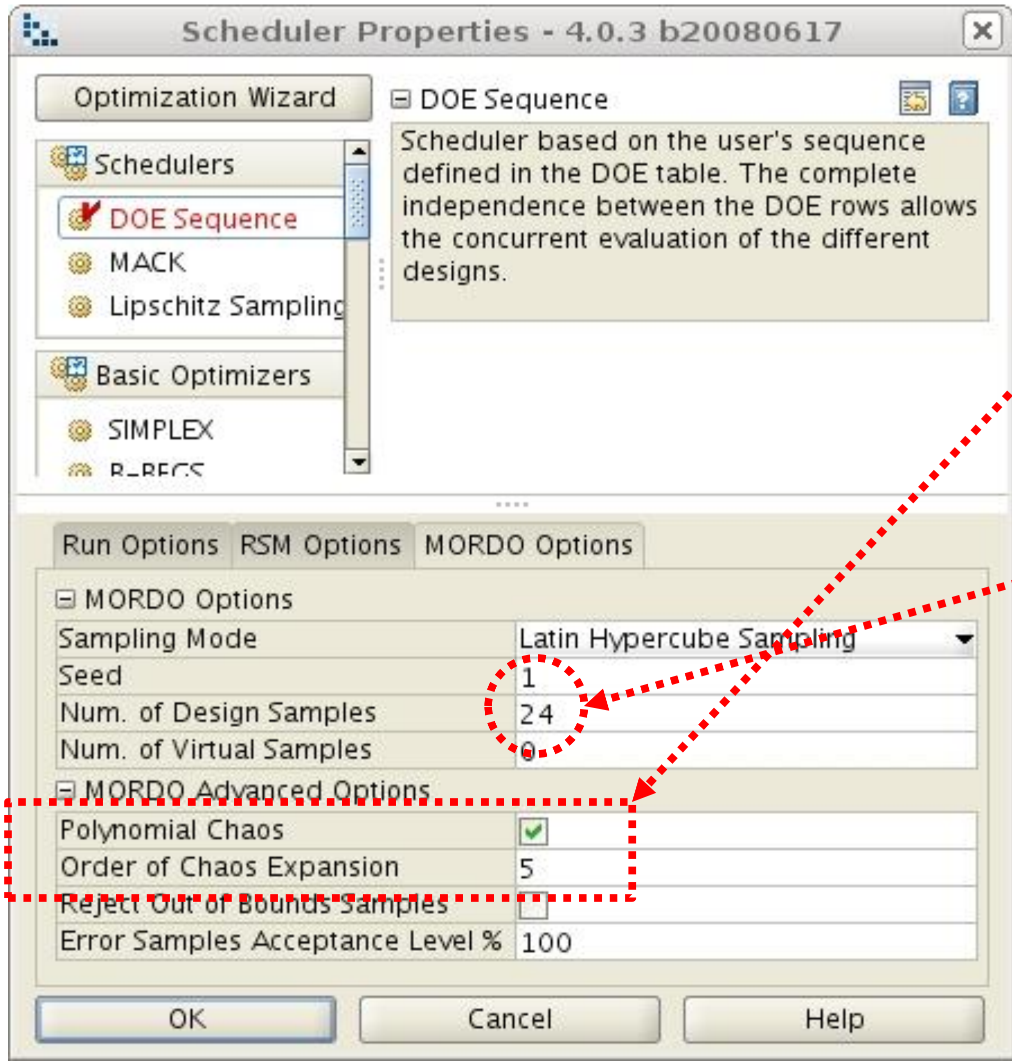
- Several **Distribution** to model Input variables uncertainties are available:
  - Cauchy
  - Logistic
  - Normal
  - Uniform
- Select it in **MORDO options** for each variable and specify the **parameters** (e.g., Standard Deviation for Normal type)



-For more information visit:
-www.esteco.com or send an e-mail to:
modeFRONTIER@esteco.com

modeFRONTIER
the multi-objective optimization and design environment

Polynomial Chaos Interface



Polynomial Chaos is switched on by default

Chaos order can vary from 1 to 7

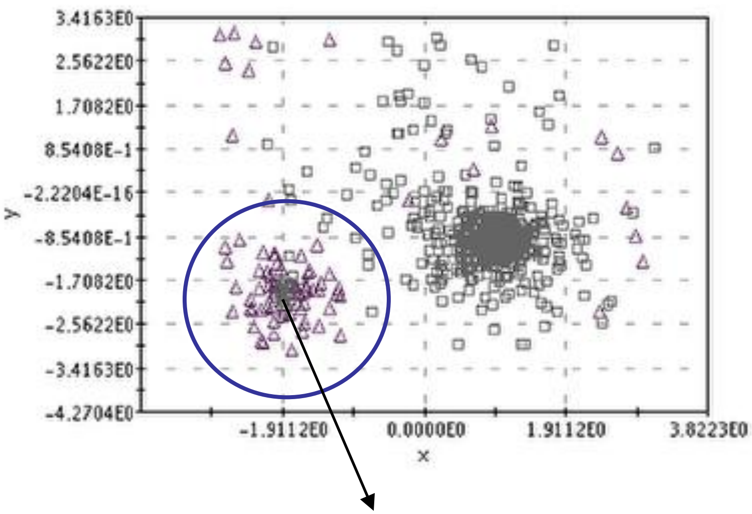
Number of Samples must be at least

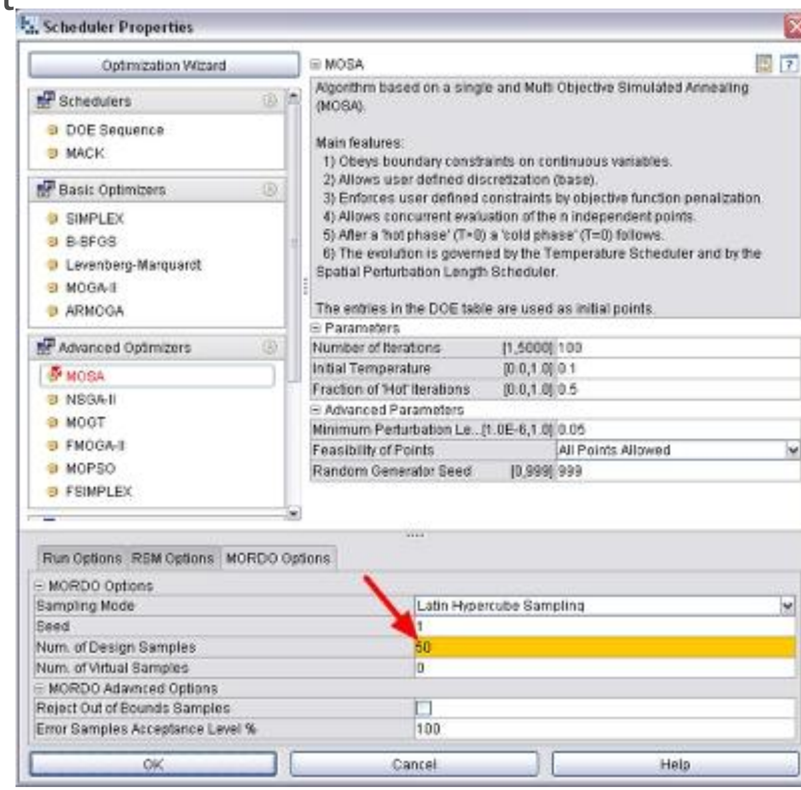
$$\frac{(nVar + order)!}{nVar!\,order!}$$

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





-For more information visit:
-www.esteco.com or send an e-mail to: modeFRONTIER@esteco.com

modeFRONTIER
the multi-objective optimization and design environment

Problems & Ideas

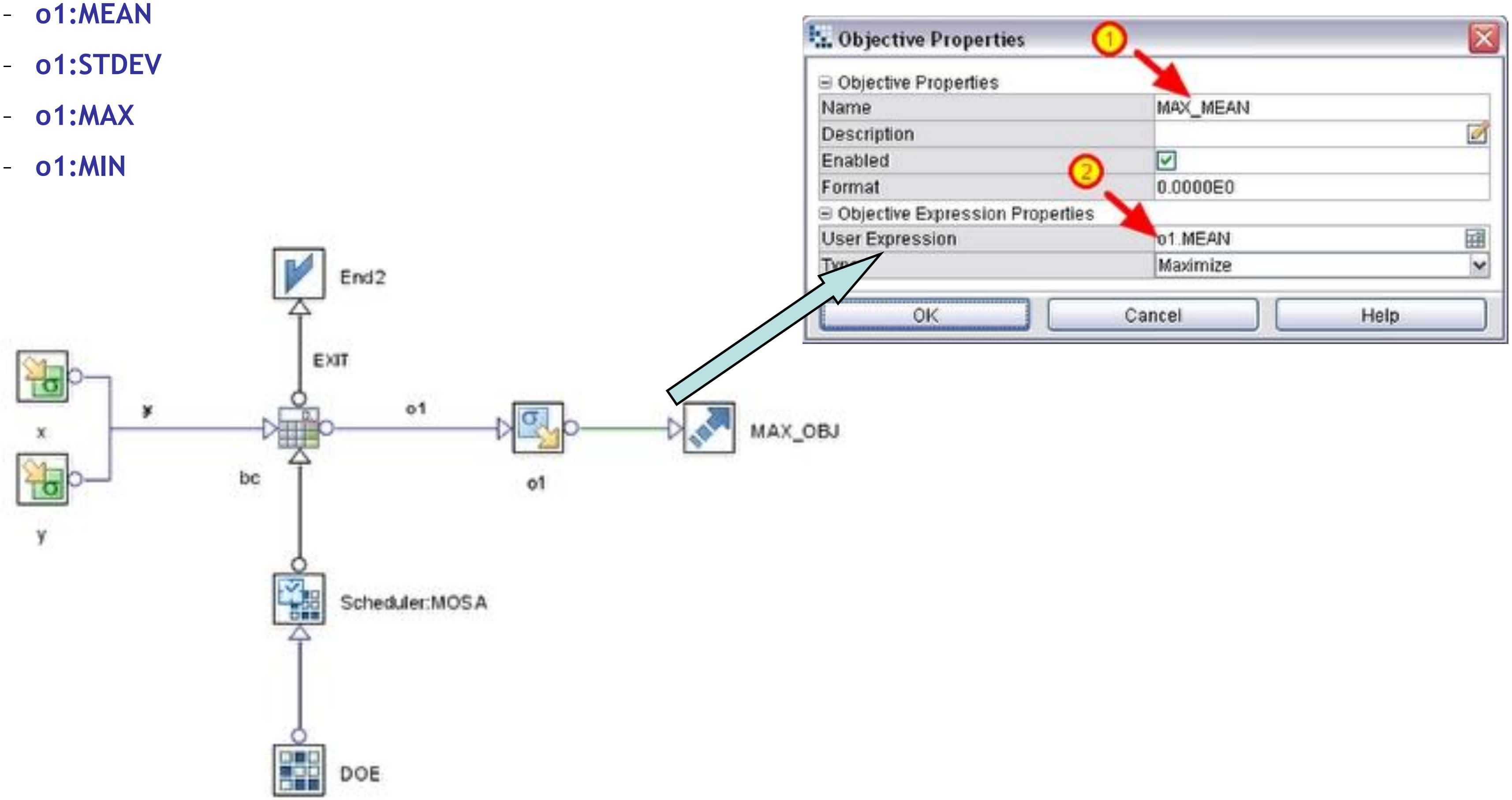
- Open Problems:
  - How to deal with bounds and constraints
  - What if the function cannot be expanded in a series (how can we decide that for a black-box function)
  - ….
- Ideas:
  - Use of chaos collocation to train a meta-model for virtual optimization
  - …

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

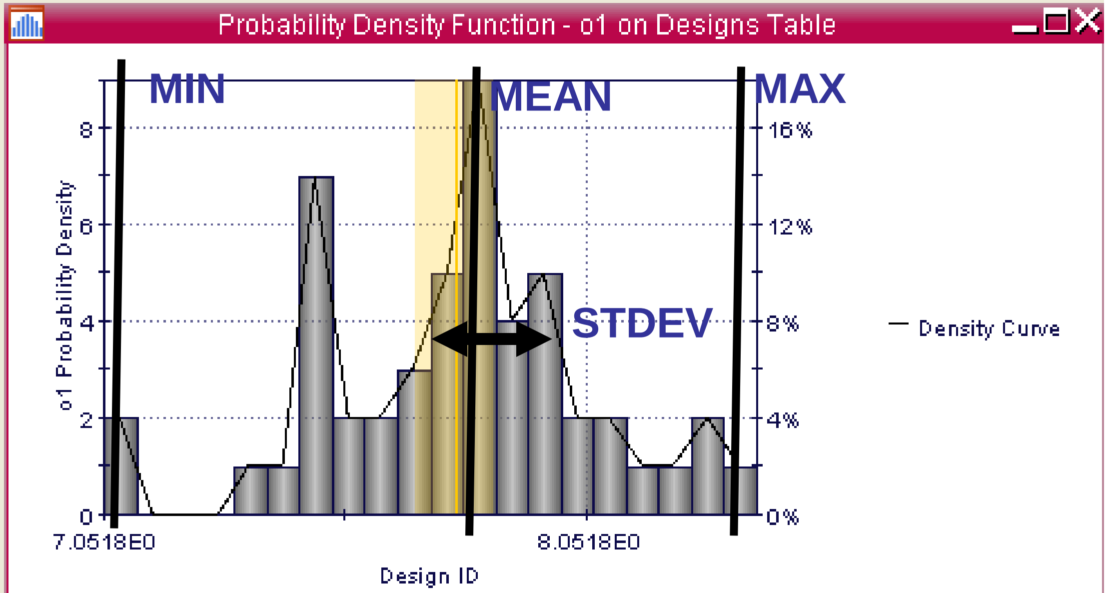
- The objectives (constraints) can be defined using **Stochastic value** for each output variable. If the latter is o1, these values are:
  - **o1:MEAN**
  - **o1:STDEV**
  - **o1:MAX**
  - **o1:MIN**



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:

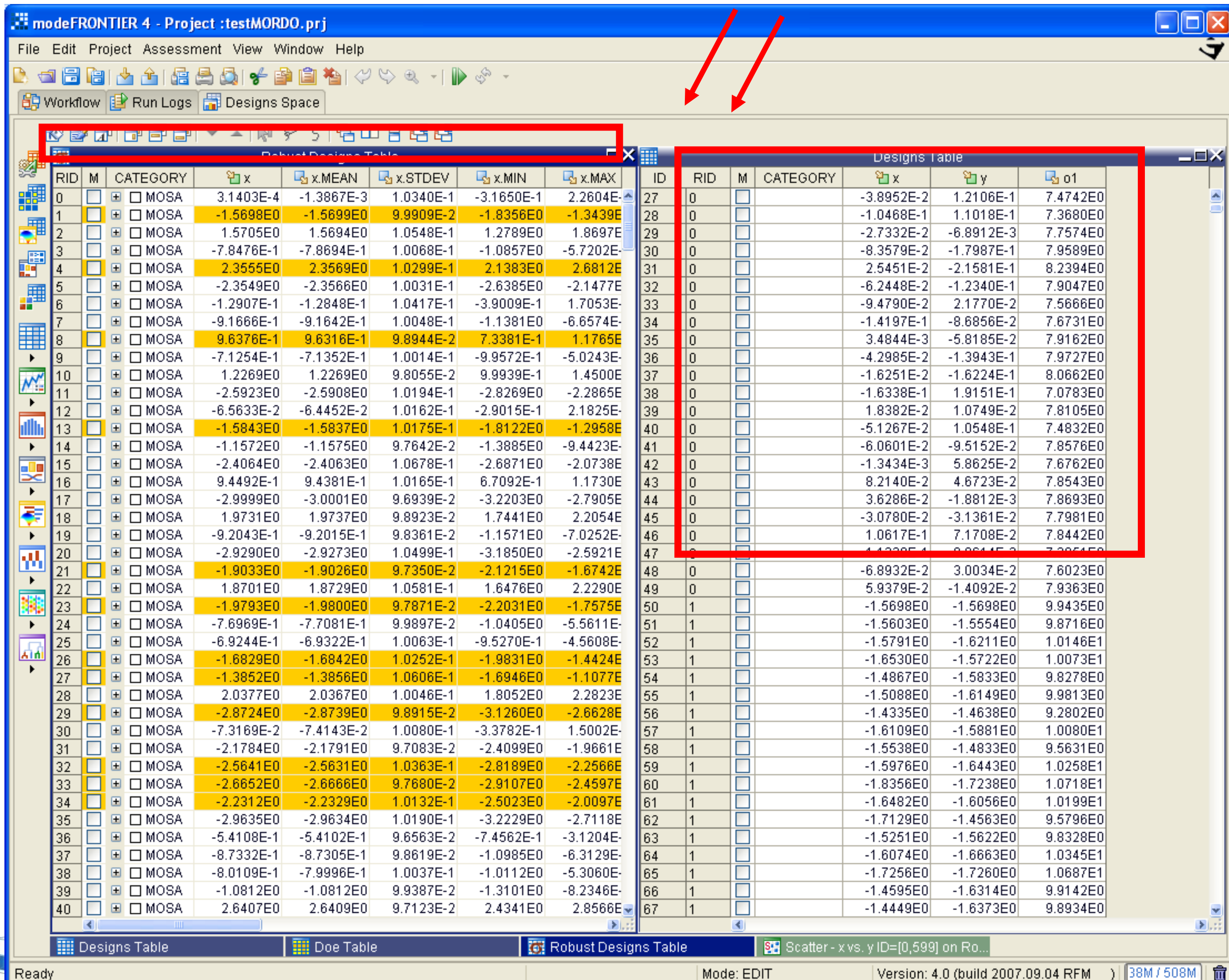


N. of Designs	50	Std. Deviation	2.8310E-1
N. of Classes	20	Skewness	-2.6821E-1
Min	7.0518E0	Kurtosis	3.3960E0
Max	8.4013E0	Confidence Int. 95%	(7.6980E0,7.8588E0)
Arithmetic Mean	7.7784E0	Confidence Int. 99%	(7.6712E0,7.8856E0)
Variance	8.0144E-2	Variation Coeff.	3.6395E-2

- you can **minimise MEAN and constrain STDEV**

- you can **minimise MAX** as alternative

Robust Design Table



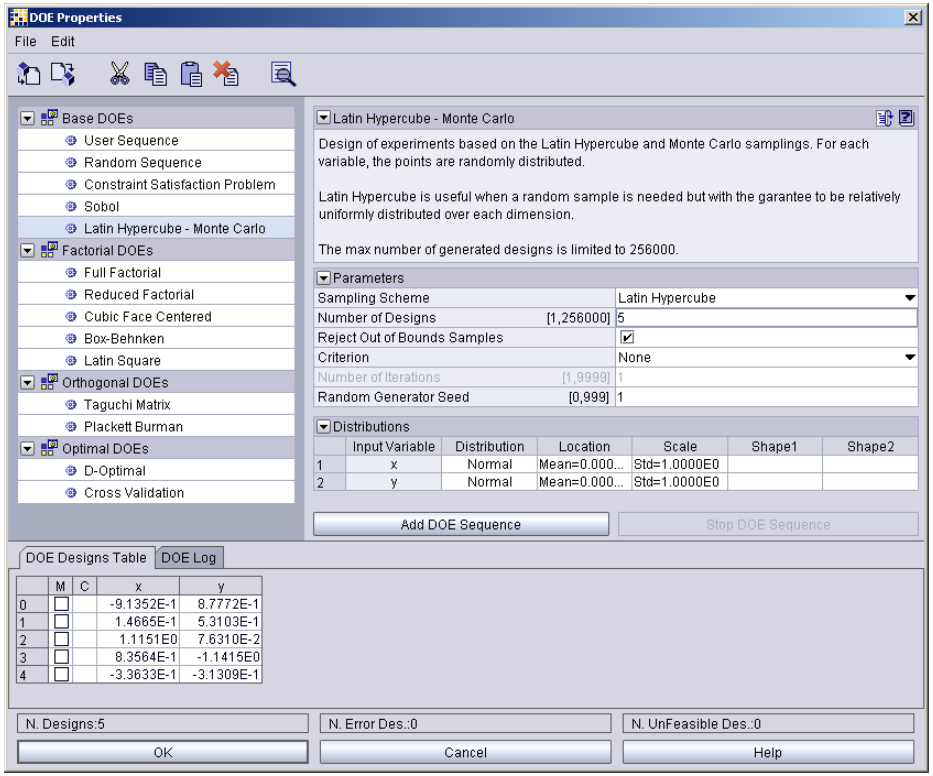
- Robust design table contains stochastic values for each design

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

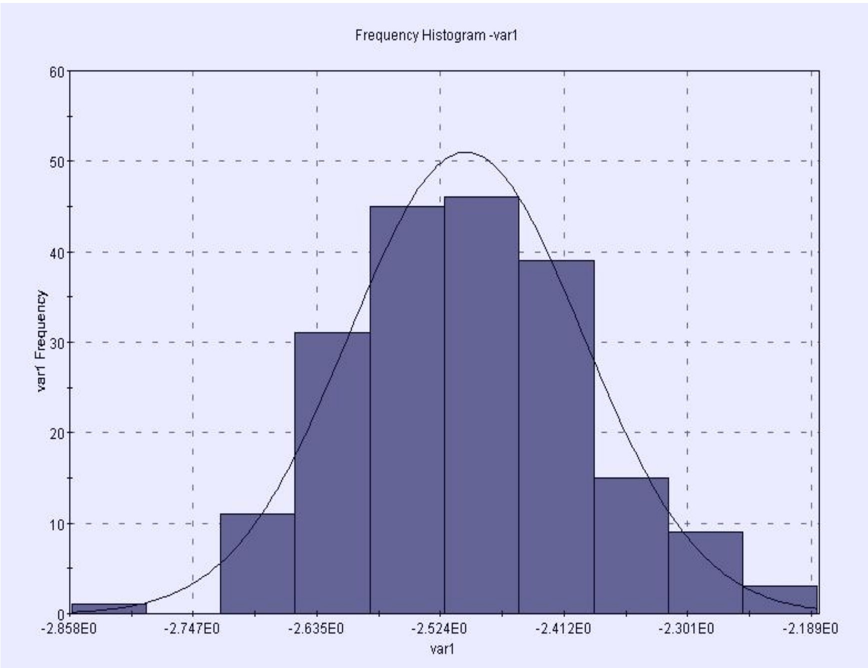
modeFRONTIER@esteco.com

Stability control

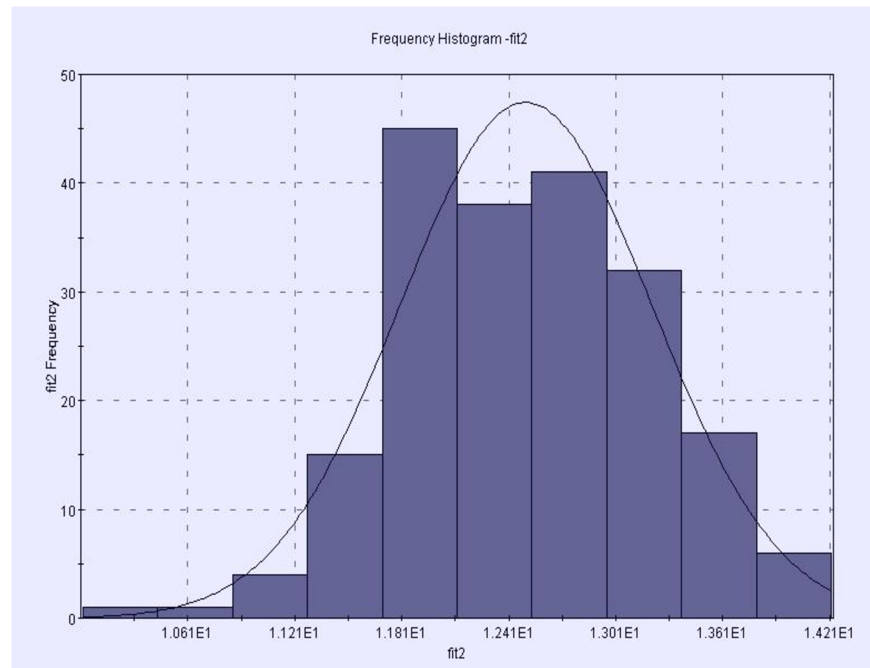
- The most useful DOE algorithm for stability control is Monte Carlo / Latin Hypercube
- Several different distributions can be defined for input parameters: Exponential, Gamma, Logistic, Lognormal, Normal, Student, Uniform, Weibull
- Monte Carlo perturbations around a point look for robust solutions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



Monte Carlo DOE



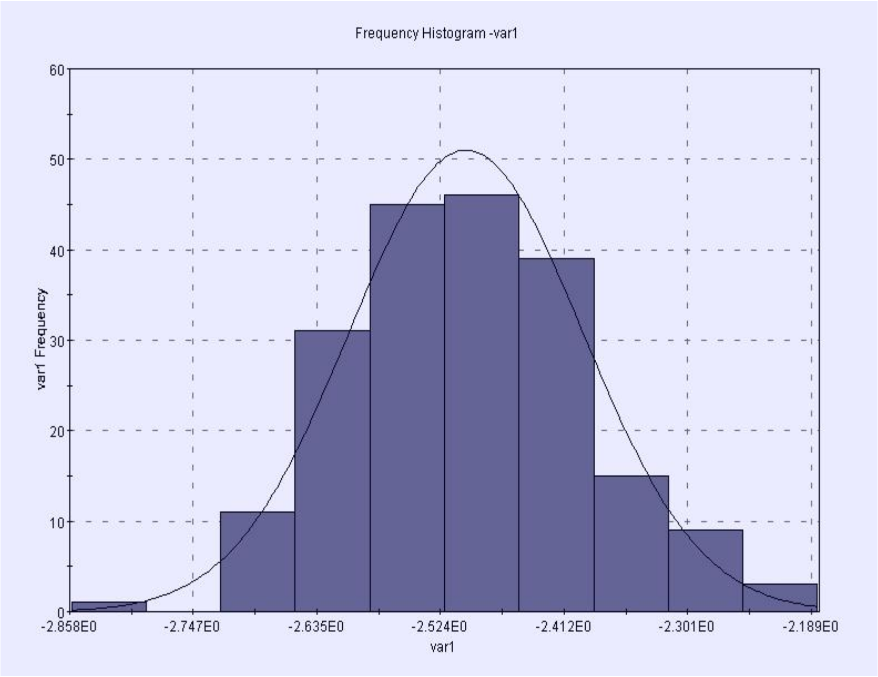
Monte Carlo Perturbation for input variables



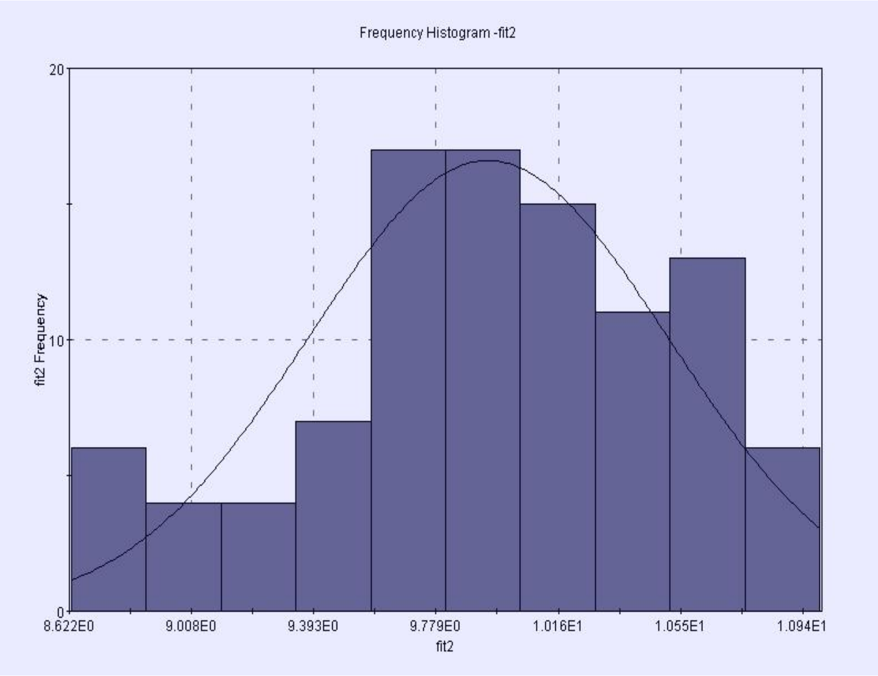
Frequency Histogram:

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

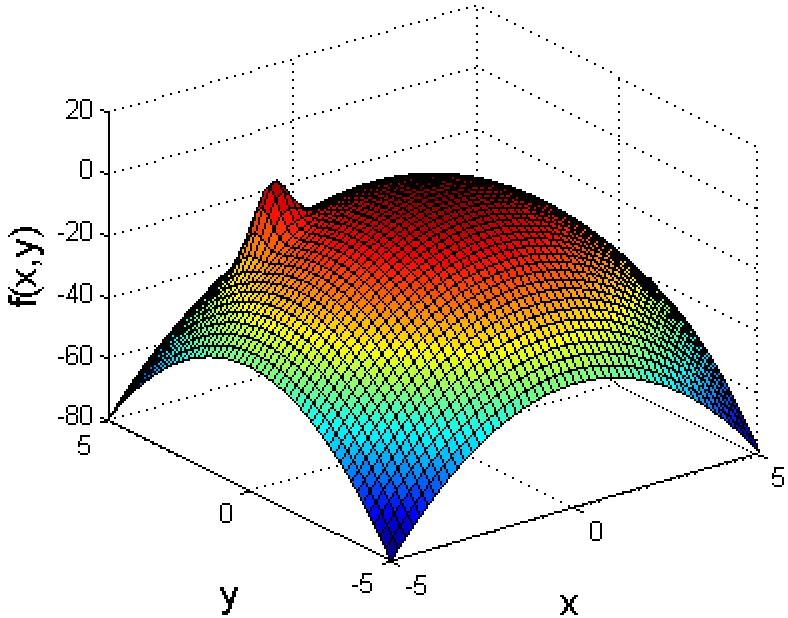
robust design optimization

Exercise 2

Maximize a Mathematical function

Maximize:

$$
\begin{aligned}
F(x,y) &= 20\exp\left(-\frac{\alpha}{2\sigma^2}\right) - 1.6\left(x^2+y^2\right)\\
\sigma &= 0.4\\
\alpha &= (x+2.5)^2 + (y-2.5)^2\\
x, y &\in [-5.0, 5.0]
\end{aligned}
$$



Maximize a Mathematical function

Global Maximum in:

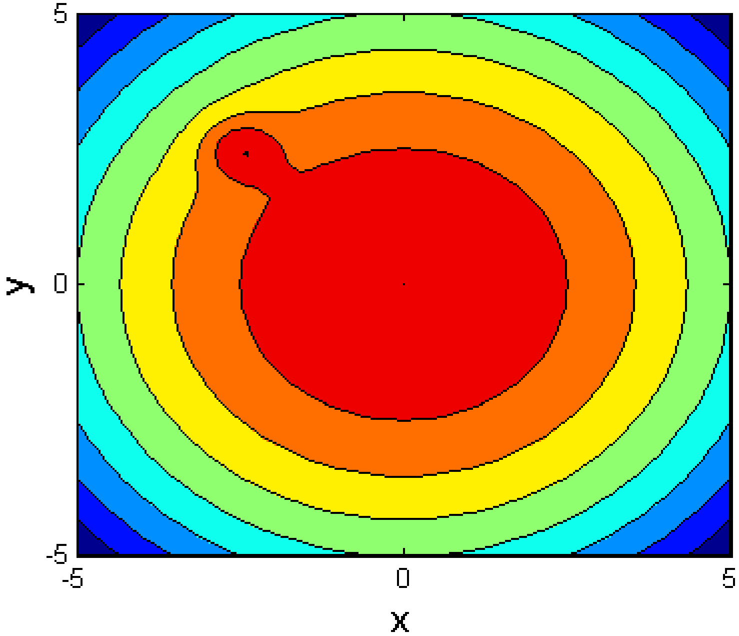
$$(x, y) \approx (-2.4360, 2.4360)$$

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

Robust Maximum in:

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

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



Stability control

Number of samples

Distribution's properties

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 garantee 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		☑
Criterion		Maximize Minimum Distance
Number of Iterations	[1,9999]	10
Random Generator Seed	[0,999]	1

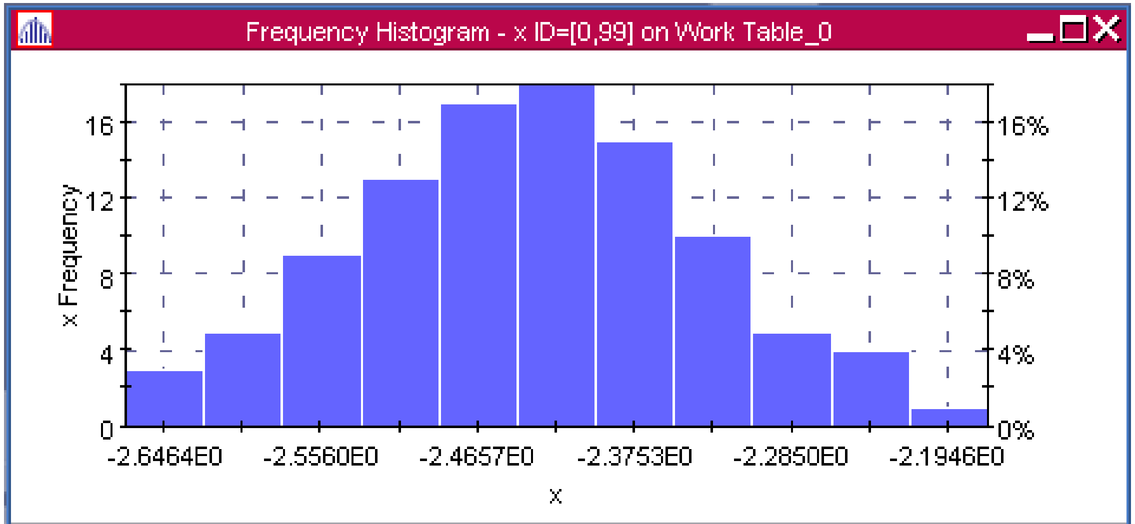
Distributions

	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		

- 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

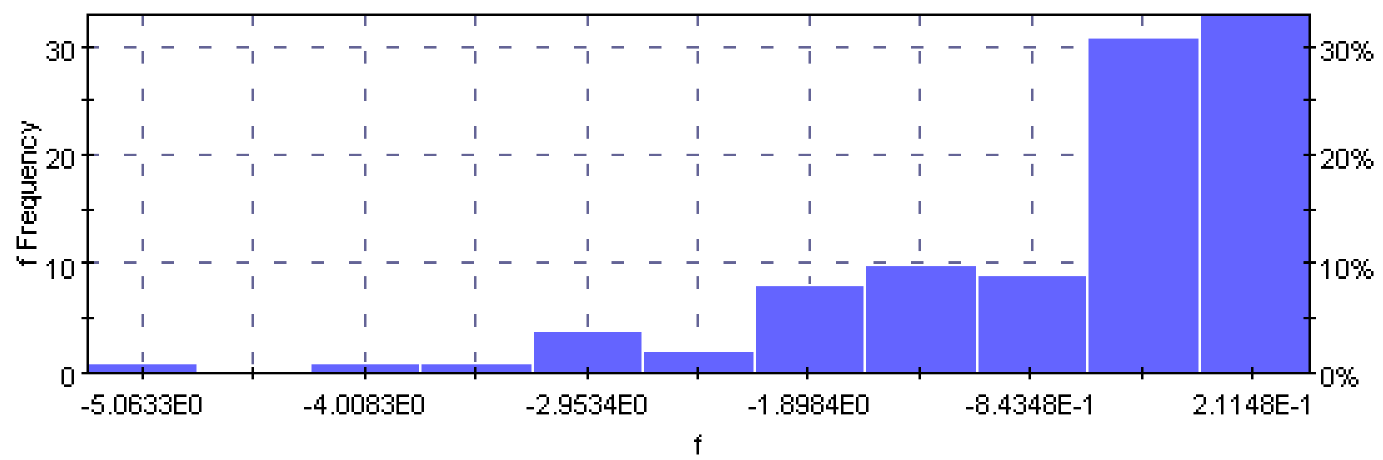
- Frequency chart for the input variable x
- Normal distribution with 100 samples using Latin Hypercube
- The table reports the moments of the distribution (mean, variance, skewness, kurtosis and confidence intervals)



N. of Designs	100	Std. Deviation	9.9573E-2
N. of Classes	11	Skewness	4.6902E-2
Min	-2.6690E0	Kurtosis	2.8153E0
Max	-2.1720E0	Confidence Int. 95%	(-2.4553E0,-2.4163E0)
Mean	-2.4358E0	Confidence Int. 99%	(-2.4615E0,-2.4101E0)
Variance	9.9148E-3	Variation Coeff.	-4.0879E-2

Latin Hypercube - Monte Carlo

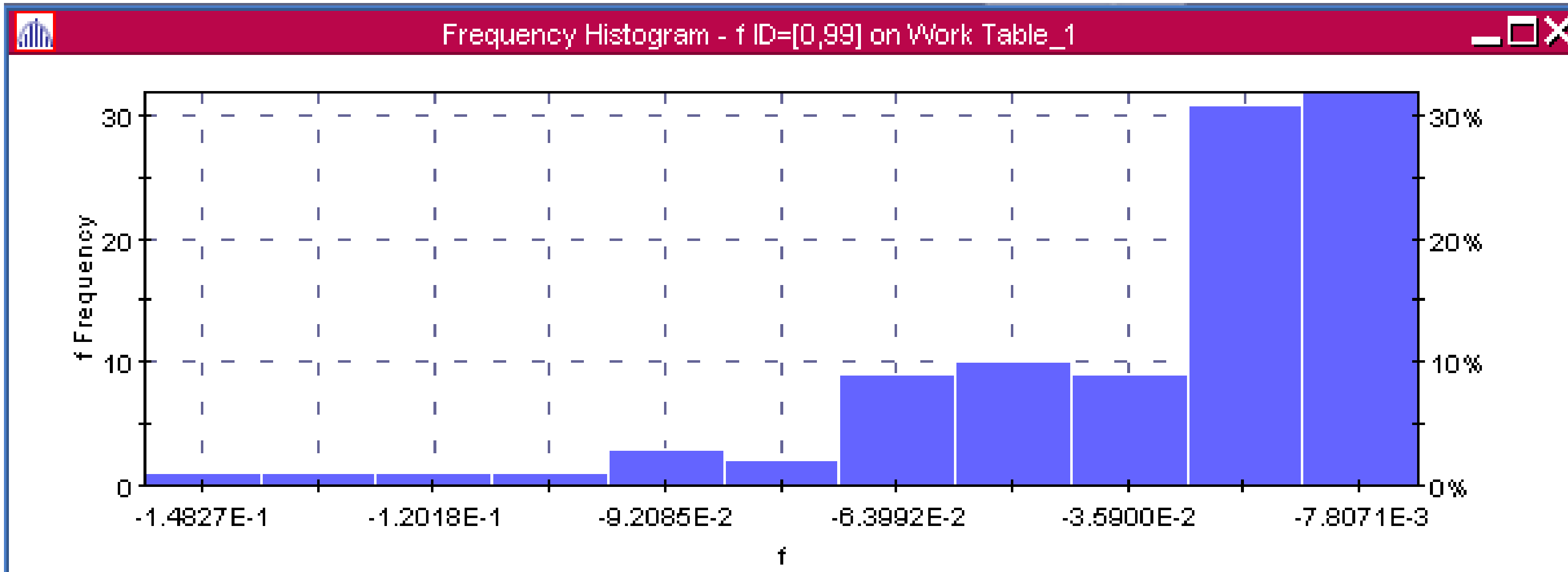
Frequency Histogram - f ID=[0,99] on Work Table_0



N. of Designs	100	Std. Deviation	1.0747E0
N. of Classes	11	Skewness	-1.6832E0
Min	-5.3270E0	Kurtosis	6.4949E0
Max	4.7521E-1	Confidence Int. 95%	(-8.6728E-1,-4.4599E-1)
Mean	-6.5664E-1	Confidence Int. 99%	(-9.3392E-1,-3.7935E-1)
Variance	1.1551E0	Variation Coeff.	-1.6367E0

- 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



N. of Designs	100	Std. Deviation	3.1033E-2
N. of Classes	11	Skewness	-1.7292E0
Min	-1.5529E-1	Kurtosis	6.3100E0
Max	-7.8400E-4	Confidence Int. 95%	(-3.8391E-2,-2.6226E-2)
Mean	-3.2308E-2	Confidence Int. 99%	(-4.0315E-2,-2.4301E-2)
Variance	9.6306E-4	Variation Coeff.	-9.6054E-1

- 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

modeFRONTIER
the multi-objective optimization and design environment

Monte Carlo DOE

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

	Mean	**Variance**
Global maximum	-0.66	1.15
Robust maximum	-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