Metamodeling

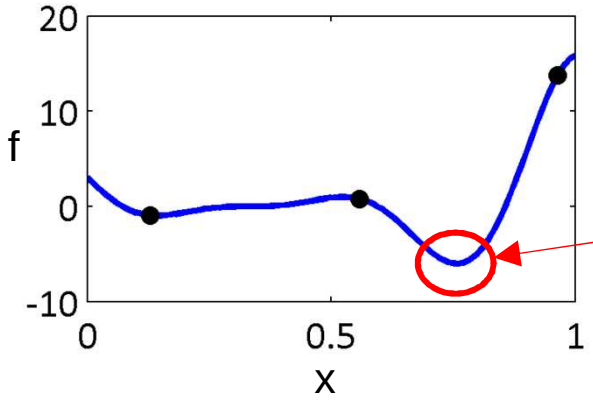
- Most modern area of optimization research
- For computationally demanding objective functions as well as constraint functions
- Main idea is that real functions are not convex but are in some sense “continuous”

Metamodeling

- Goal: find an approximation (meta-model) M of a problem (model) P such that:
  - M is less demanding than P
  - Minimum of M is equal to minimum of P (for objective functions) or the hyperplane dividing the space into feasible and unfeasible space is the most accurate (for constraint functions)

Metamodeling for optimization

- Objective function

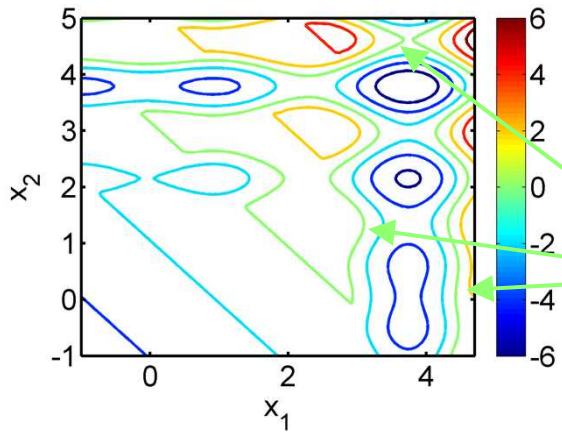


Optimization task:

$$\min f(x) = (6x-2)^2 \sin(12x-4)$$

interesting region

- Constraint function



Optimization task:

$$\min f(x) = (x_1 - 3.7)^2 + (x_2 - 4)^2$$

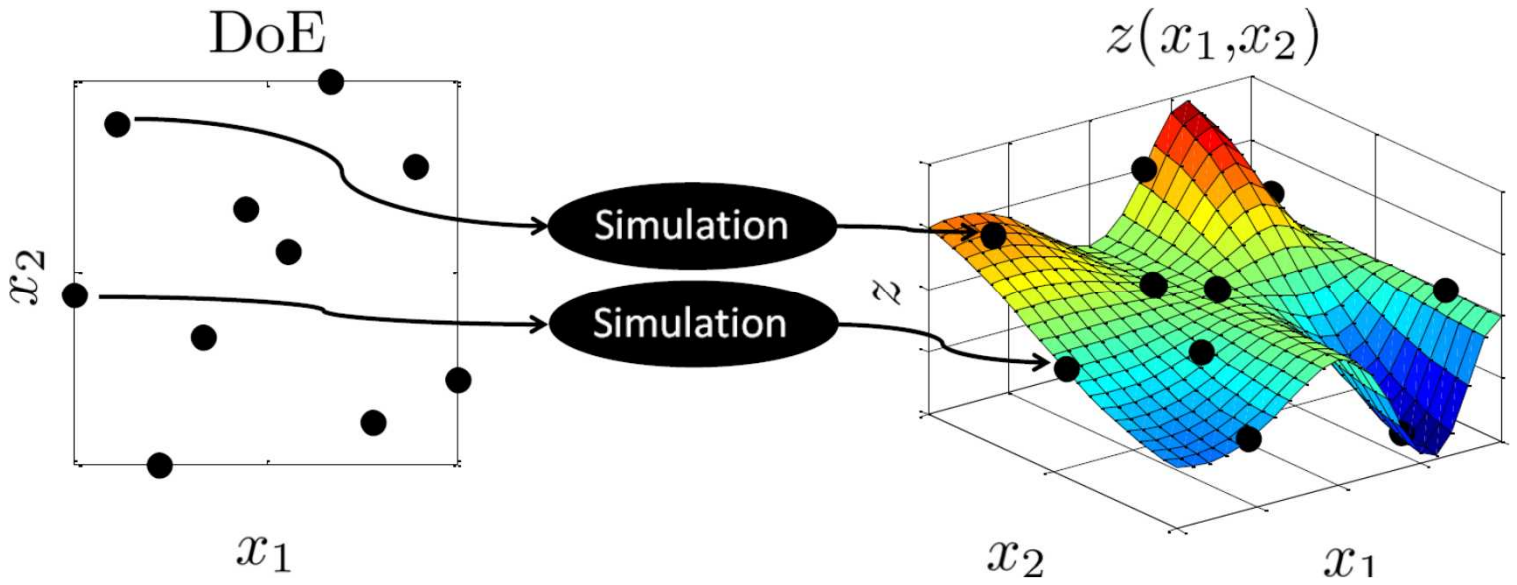
$$\text{s.t.} \quad -x_1 \sin(4x_1) - 1.1x_2 \sin(2x_2) \geq 0$$

$$x_1 + x_2 - 3 \geq 0$$

interesting hyperplane
(contour for 2D problem)

Metamodeling

- Original model still necessary to evaluate few times
- Choosing points where to enumerate original model - Design of Experiments (DoE)



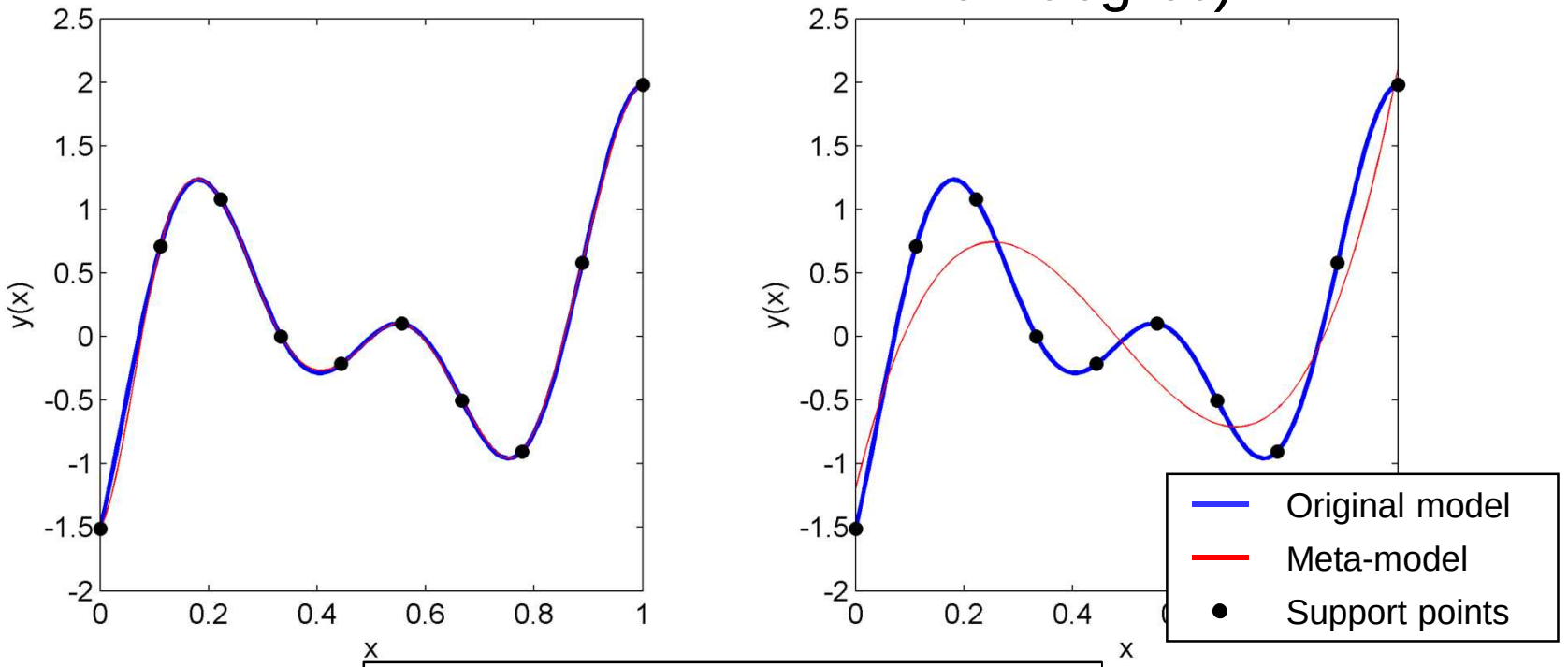
General division of meta-models

- Interpolating models intersecting all support points based on an idea of linear combination of some basis functions
- Non-interpolating models minimizing sum of squares errors for some predetermined functional form
- Combination of above models

General division of meta-models

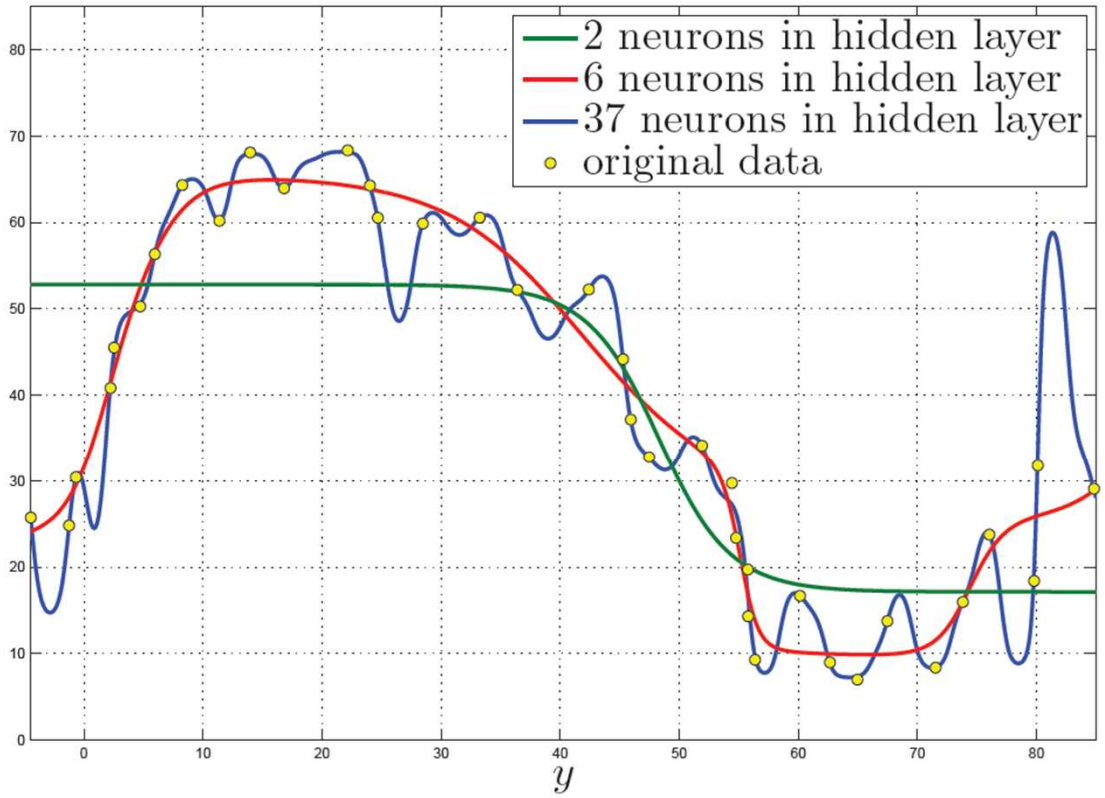
Example of interpolating model (Radial basis functions)

Example of non-interpolating model (Legendre polynomials: 3rd degree)



Original model: $y(x) = (4x - 2)\sin(12x - 4)$

Under-learning and Overtraining issues



Approximation of data by multi-layer perceptron with different topology.

Under-learning and Overtraining issues

- Solution: Three distant sets of points:
  - Traing set
  - Testing set
  - Validation set

Structure of generic metamodel

DoE	Model choice	Model fitting	Example
Factorial	polynomial	Least squares regression	Response Surface Methodology
Central composite	Splines	Weighted least squares	
D-optimal	Random field realization	Best linear predictor	Kriging
Fully random	Set of functions and terminals	Genetic Algorithm	Genetic programming
Latin Hypercube	Neural net	Back propagation	BP Neural Networks
Selected by hand	Decision tree		
Orthogonal array	Radial basis function	Minimization of entropy	Inductive Learning

Design of Experiments (DoE)

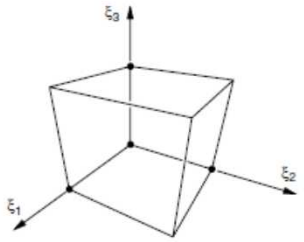
Many choices:

Random | *LHS design* | *Optimal* | *Optimal LHS* | *Factorial* | *Sparse grid KPU* | *Sparse grid GQU*

DoE

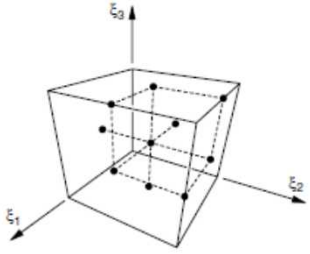
- **Saturated designs**
  - Number of support points are sufficient to represent a certain class of a meta-model exactly
  - **Linear saturated design**

$$m = n + 1$$



  - **Quadratic saturated design**

$$m = \frac{n(n+1)}{2} + n + 1$$



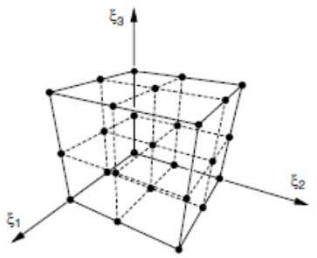
For $n = 3$

DoE

- **Redundant designs**
  - Number of support points is higher than necessary → error check
  - **Full factorial method (grid search)**

$$m = q^n$$

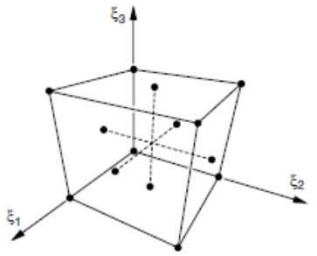
(q samples for each variable)



For $q = 3$ and $n = 3$

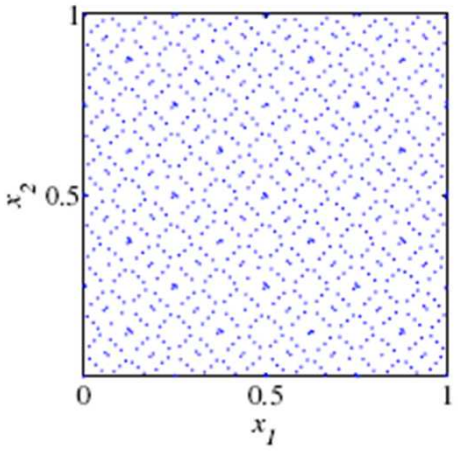
  - **Central composite design**
    - full factorial design with $q = 2$ and collection of all center points of the faces of an n-dimensional hypercube

$$m = 2^n + 2n$$

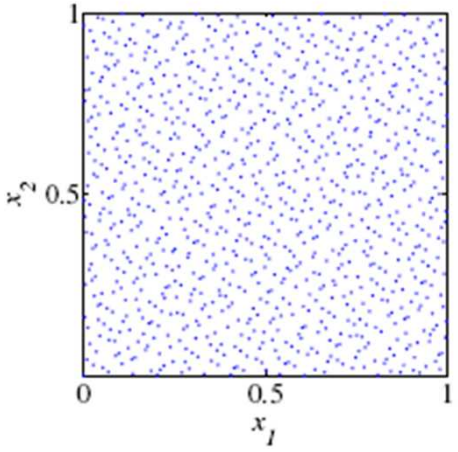


DoE

- D-optimal designs
- Random designs
  - Uniform Monte Carlo sampling
  - Latin Hypercube sampling
- Quasi-ranom designs
  - Halton, Sobolov, Faure
  - Van der Corput for 1D
- Distance-based designs
  - Maximin and Minimax designs
- Entropy-based designs



(b) Sobol - 2D - 1000 bodů.



(e) Halton - 2D - 1000 bodů.

DoE

- Monte Carlo method
  - sampling method, uses random (pseudo) numbers generation
  - generates vectors with prescribed probability distributions
- Quasi-Monte Carlo
  - uses quasi random numbers generation

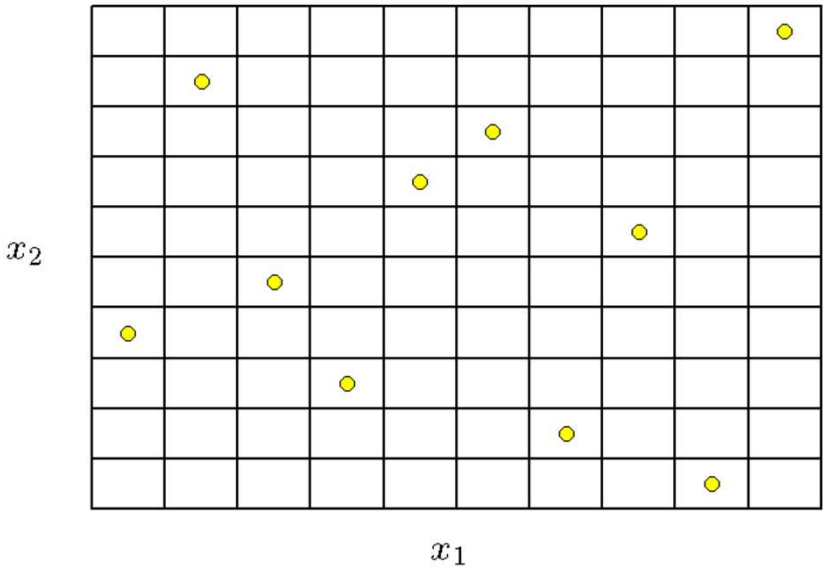
LHS method

- Latin Hypercube Sampling
  - Uses less number of samples than MC
  - Divides each variable into N_{sim} equal (in probability sense) stripes
  - A sample is selected as a mean of each stripe
  - Then change of an order of samples, not their values

LHS method

realizations

variables							
x_n	3	4	5	6	7	8	9
	-4	-3	-2	-1	0	1	2
...	1	2	3	4	5	6	7
	32	34	36	38	40	42	44
	4	5	6	7	8	9	10
x_2	73	76	79	82	85	88	91
x_1	6	7	8	9	10	11	12



Optimal Latin Hypercube Sampling

- To optimize “space cover” by all samples
- Possible methods:
  - maximization of entropy / minimization of discrepancy
  - maximization of minimum distance among points
  - Potential energy based norm
  - Prescribed correlation matrix
- Optimization e.g. using Simulated Annealing

Objectives for uniformity

- Audze – Eglais (AE) *[P. Audze, V. Eglais,1977]*

$$E^{\mathrm{AE}} = \sum_{i=1}^{n} \sum_{j=i+1}^{n} \frac{1}{L_{ij}^2},$$

- Potential energy based norm

- Eucledian maximin distance (EMM) *[M. Johnson,1990]*

$$E^{\mathrm{EMM}} = -\min\{..., L_{ij}, ...\}, \quad i = 1...n, \quad j = (i+1)...n$$

- Modified L_2 discrepancy (ML2) *[T. M. Cioppa, 2007]*

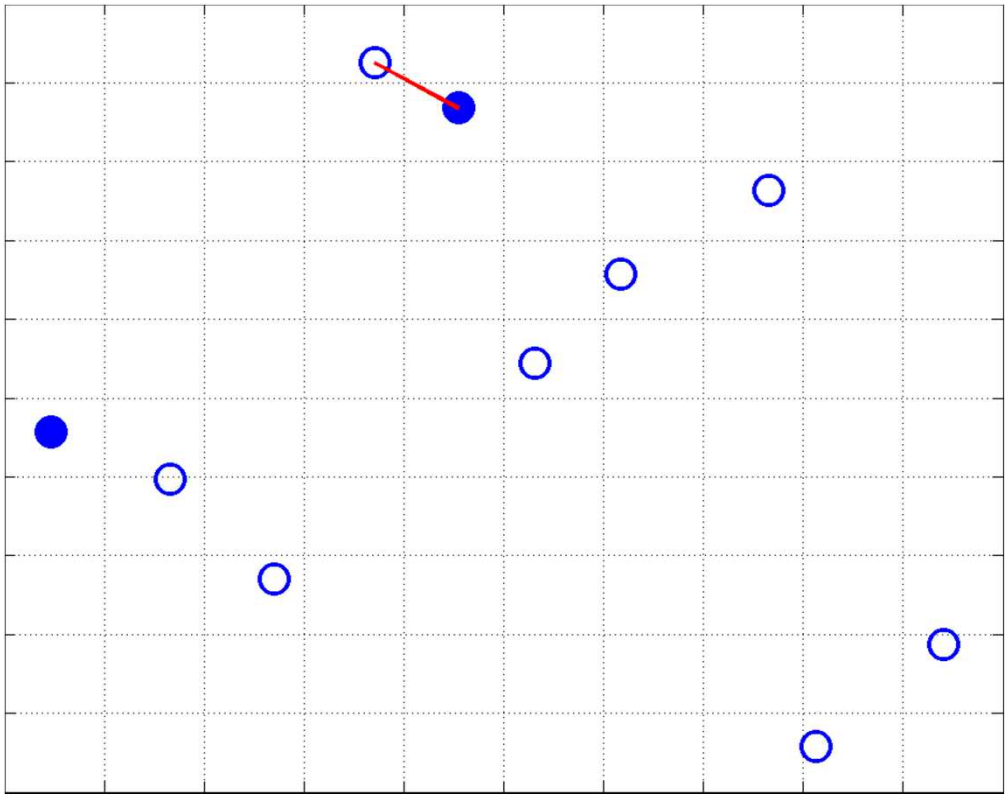
$$E^{\mathrm{ML_2}} = \left(\frac{4}{3}\right)^k - \frac{2^{(1-k)}}{n} \sum_{d=1}^{n} \prod_{i=1}^{k} (3 - x_{di}^2) + \frac{1}{n^2} \sum_{d=1}^{n} \sum_{j=1}^{n} \prod_{i=1}^{k} [2 - \max(x_{di}, x_{ji})],$$

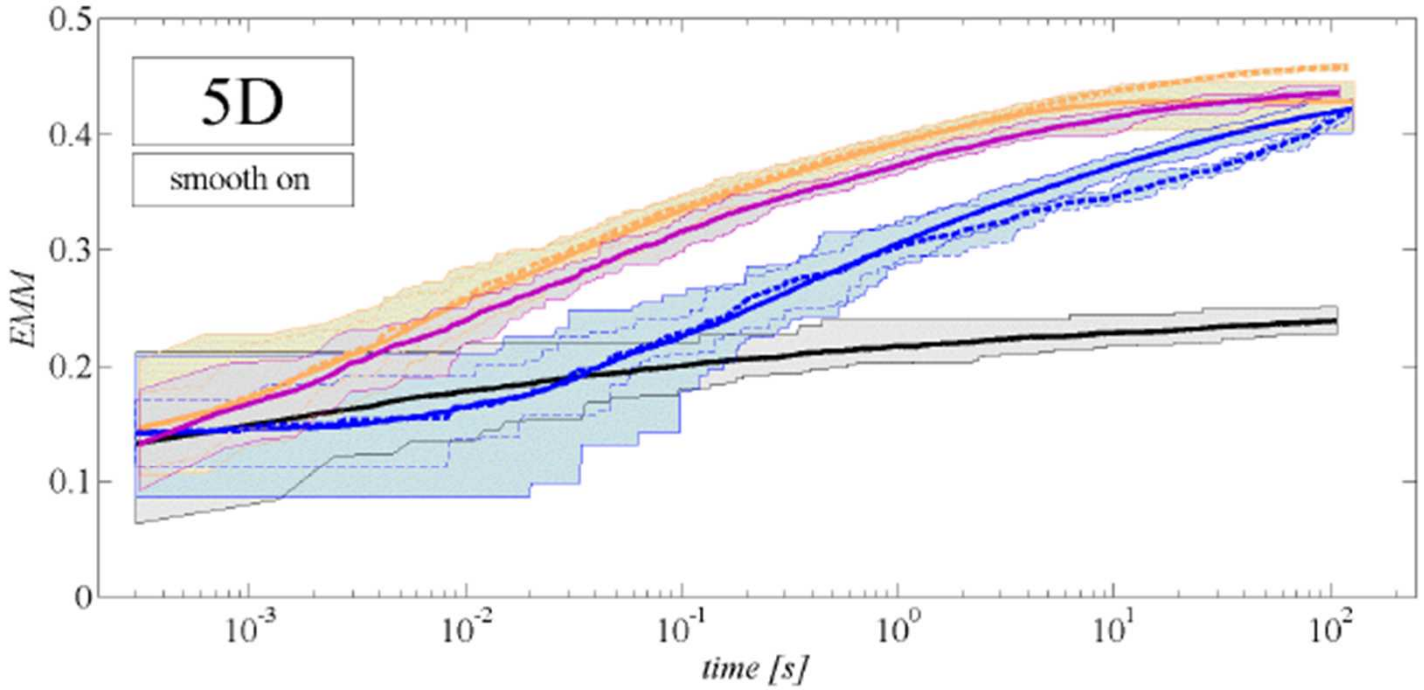
- D-optimality (Dopt) *[Kirsten Smith, 1918; M. Hofwing, 2010]*

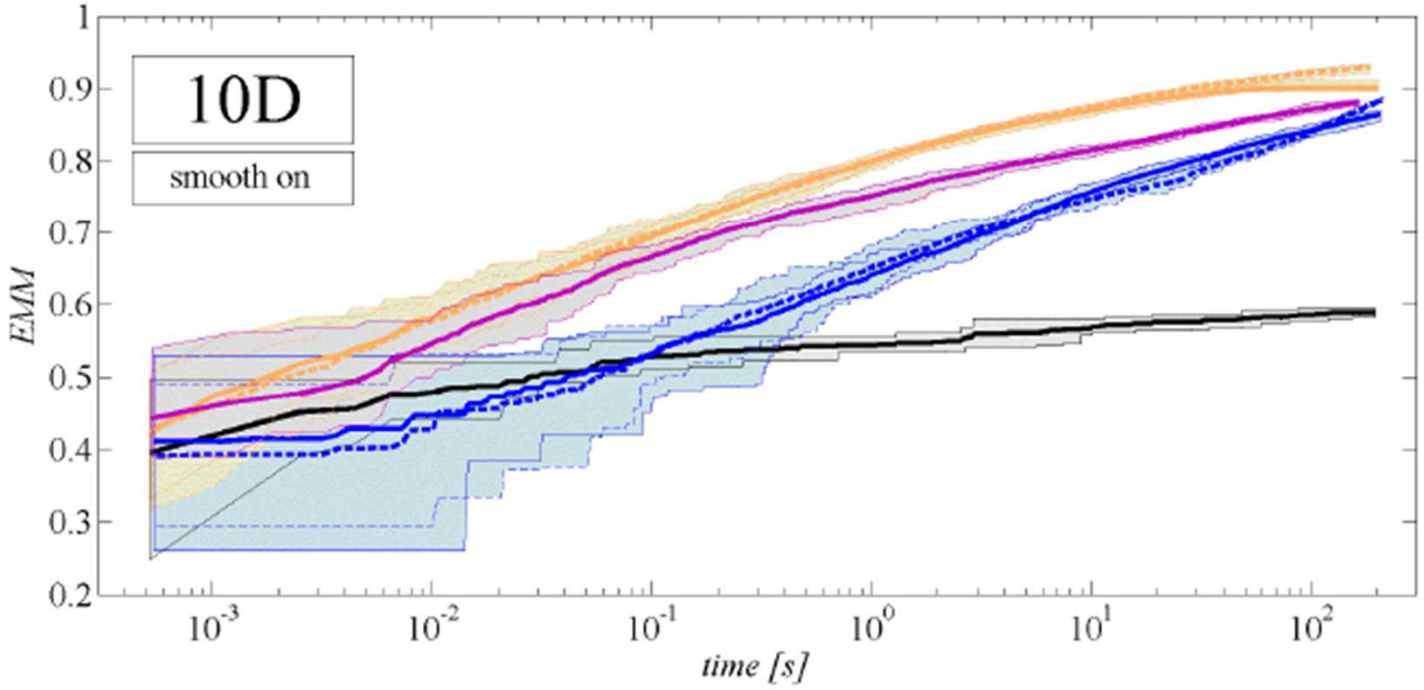
$$E^{\mathrm{Dopt}} = -\det(\mathbf{Z}^{\mathbf{T}}\mathbf{Z}),$$

$$\mathbf{Z} = \begin{bmatrix} 1 & x_{11} & x_{12} & x_{11}^2 & x_{12}^2 & x_{11}x_{12} \\ 1 & x_{21} & x_{22} & x_{21}^2 & x_{22}^2 & x_{21}x_{22} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_{n1} & x_{n2} & x_{n1}^2 & x_{n2}^2 & x_{n1}x_{n2} \end{bmatrix}$$

Optimized LHS – heuristic procedure plus Simulated annealing




5D
smooth on
EMM
time [s]


10D
smooth on
EMM
time [s]

Orthogonality measures

- Conditional number (CN) *[T. M. Cioppa, 2007]*

$$E^{\mathrm{CN}} = \mathrm{cond}(\mathbf{X}^{\mathbf{T}}\mathbf{X}) = \frac{\lambda_{\mathbf{1}}}{\lambda_{\mathbf{n}}}$$

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1k} \\ x_{21} & x_{22} & \cdots & x_{2k} \\ \vdots & \vdots & & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nk} \end{bmatrix}$$

- Pearson's correlation coefficient (PMCC)

$$E^{\mathrm{PMCC}} = \sqrt{\sum_{i=1}^{k}\sum_{j=i+1}^{k} c_{ij}^2}\,. \qquad c_{ij} = \frac{\mathrm{Cov}\,(x_i, x_j)}{\sigma_{x_i}\sigma_{x_j}} = \frac{\sum_{a=1}^{n}(x_{a,i} - \overline{x_i})(x_{a,j} - \overline{x_j})}{\sqrt{\sum_{a=1}^{n}(x_{a,i} - \overline{x_i})^2 \sum_{a=1}^{n}(x_{a,j} - \overline{x_j})^2}}$$

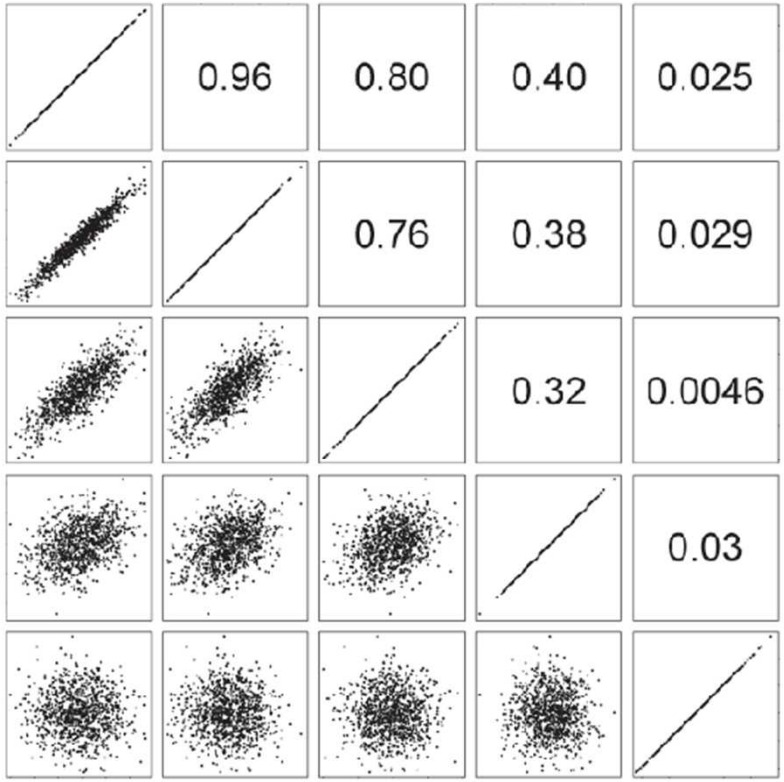
- Spearman's rank-based correlation coefficient (SRCC)

$$E^{\mathrm{SRCC}} = \sqrt{\sum_{i=1}^{k}\sum_{j=i+1}^{k} \rho_{ij}^2} \qquad \rho_{ij} = 1 - \frac{6\sum_{a=1}^{n}\left(r(x_{a,i}) - r(x_{a,j})\right)^2}{n(n^2-1)}\,.$$

- Kendall's rank-based correlation coefficient (KRCC)

$$E^{\mathrm{KRCC}} = \sqrt{\sum_{i=1}^{k}\sum_{j=i+1}^{k} \tau_{ij}^2} \qquad \tau_{ij} = \frac{T_{c,ij} - T_{d,ij}}{n(n-1)/2}$$

Prescribed correlation matrix

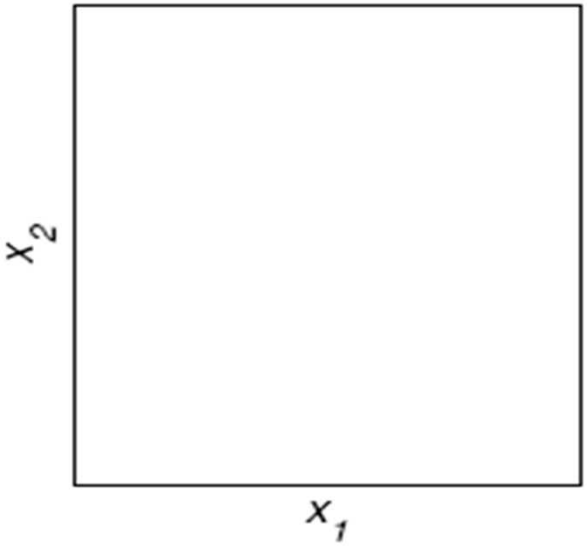


Positive linear correlations between 1000 pairs of numbers.

Design domain

- Based on existence and/or shape of constrains
  - bounded domain

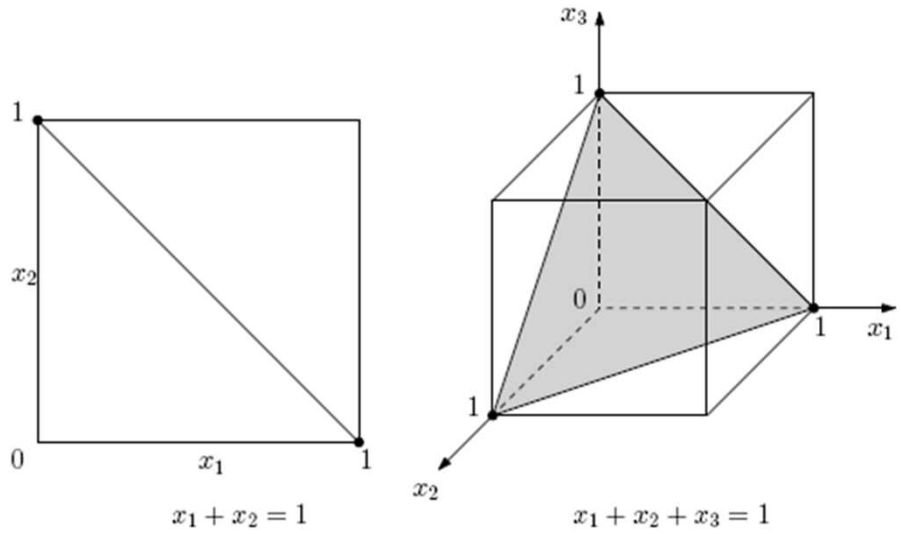
    –> hypercube



Design domain

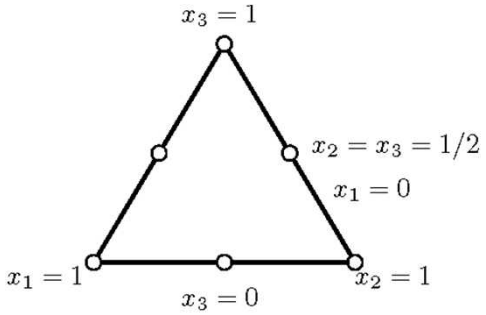
- mixture experiment
  - –
  - –> simplex

$$x_1 + x_2 + \ldots + x_n = 1;\ 0 \le x_i \le 1;\ i = 1, \ldots, n$$

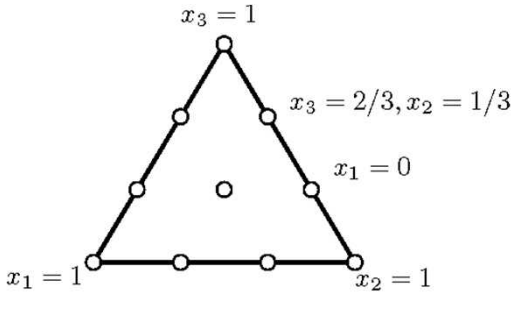


Simplex DoEs

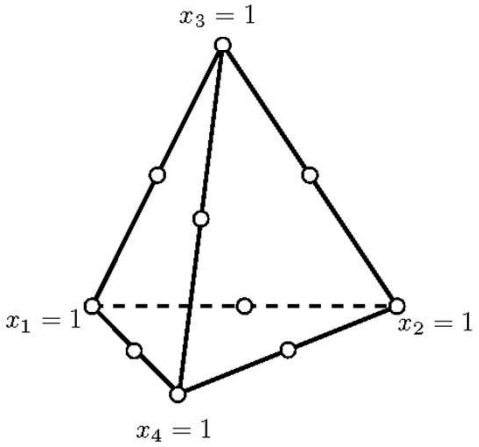
- Classical templates/patches



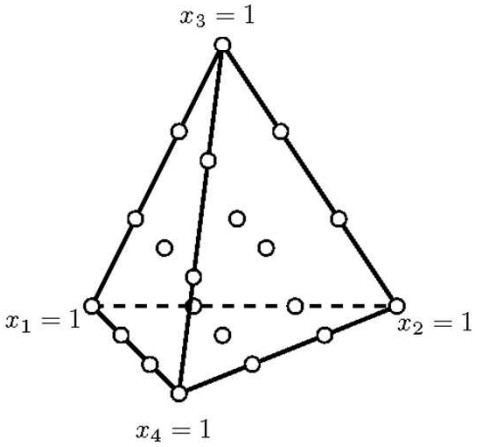
[3,2] lattice



[3,3] lattice



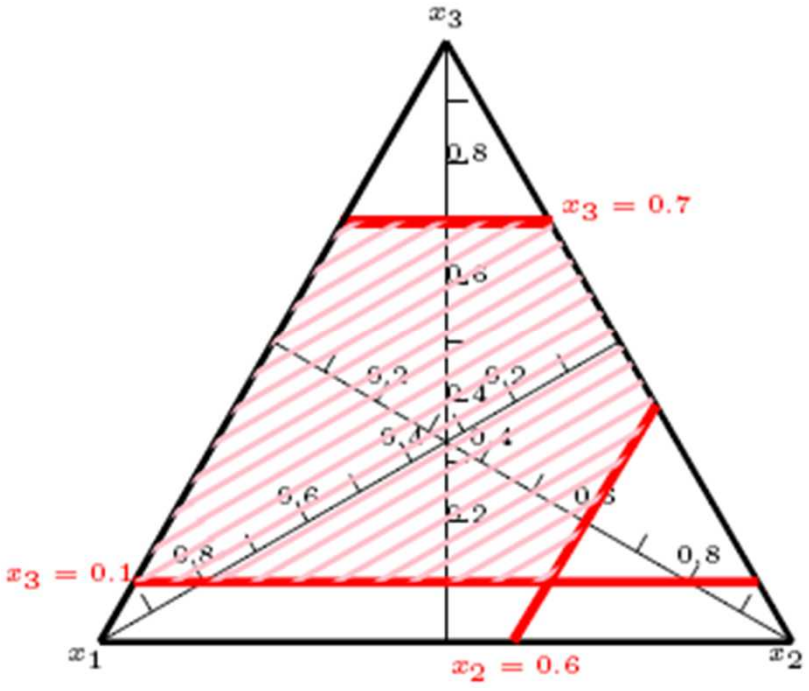
[4,2] lattice



[4,3] lattice

Design domain

- Additional linear conditions–> polytope



Mixture condition:

$$x_1 + x_2 + x_3 = 1$$

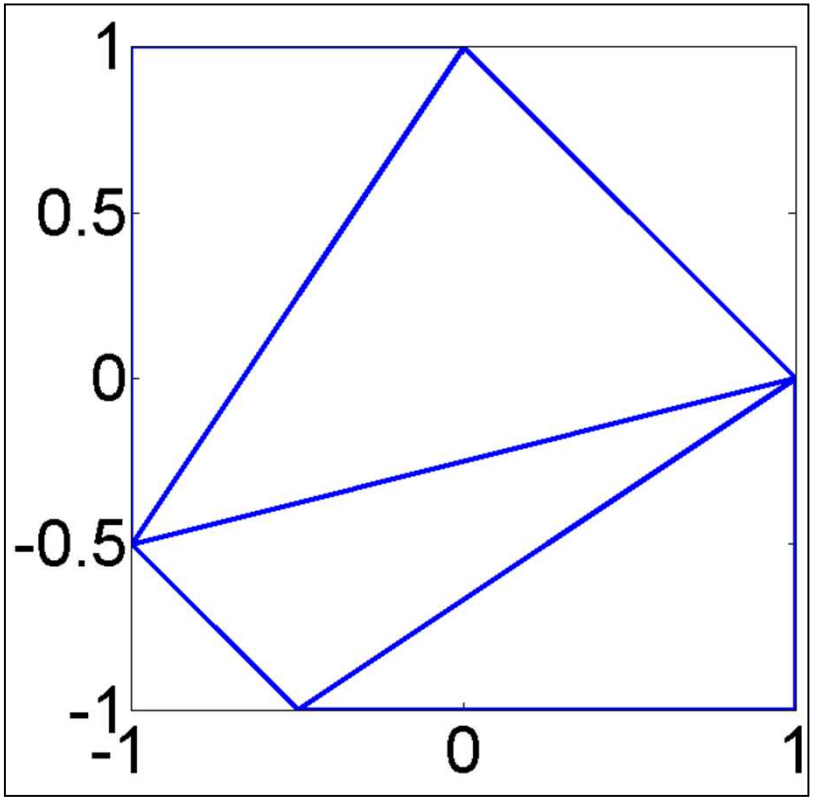
Relative amount constrains:

$$x_2 \leq 0.6$$

$$x_3 \geq 0.1$$

$$x_3 \leq 0.7$$

Distmesh tool

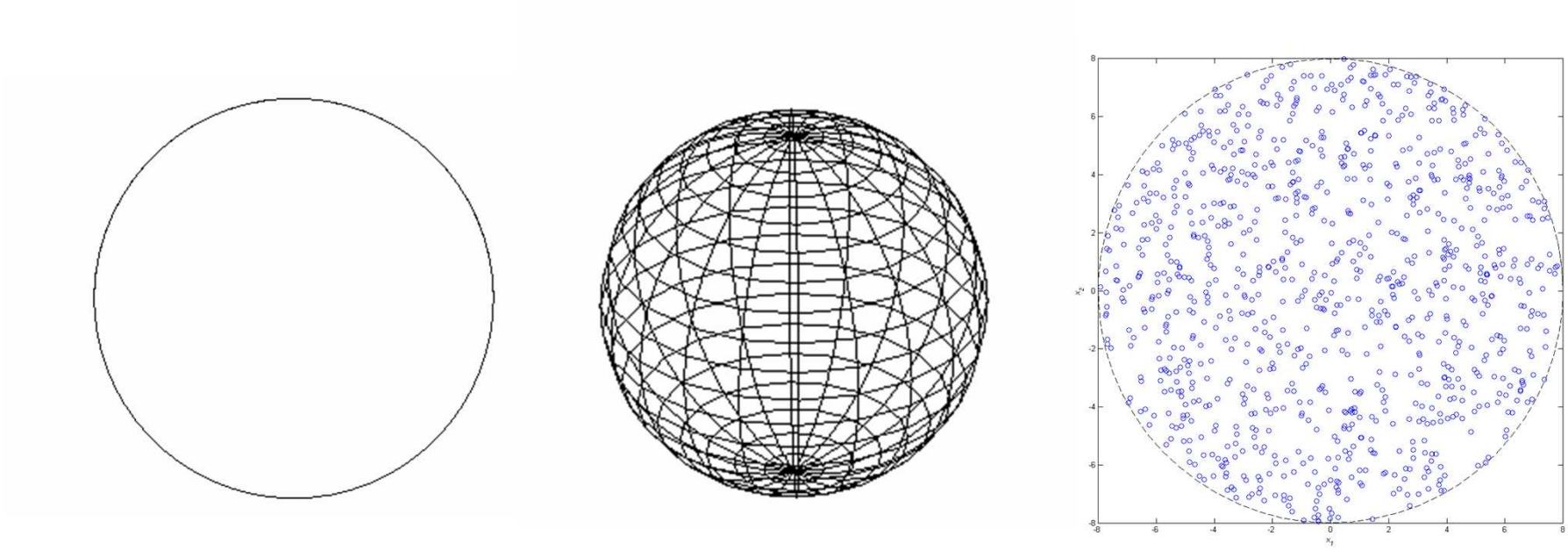


Design domain

- Limited distance to given point (origin)

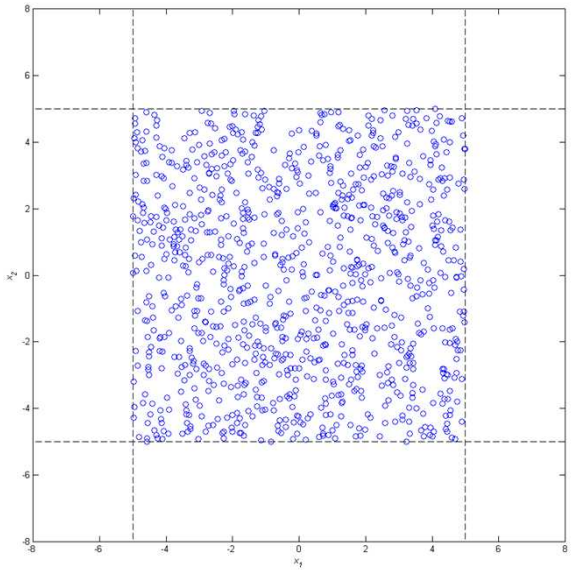
  –> $\boxed{\|x\|_2 \le R}$

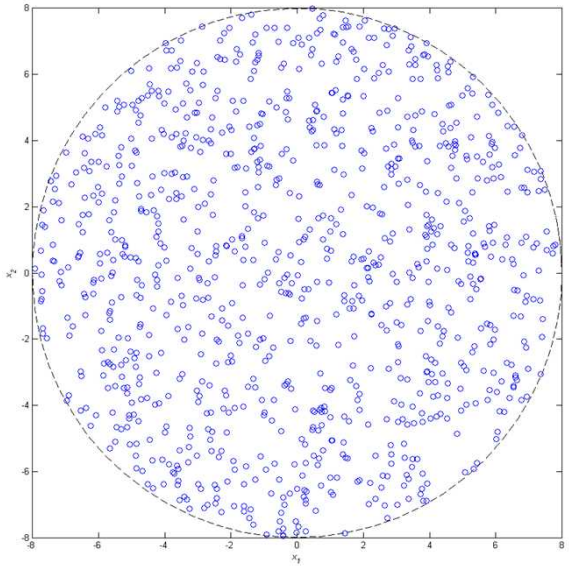
  hypersphere



Note: volume of unit hypercube vs. unit hypersphere

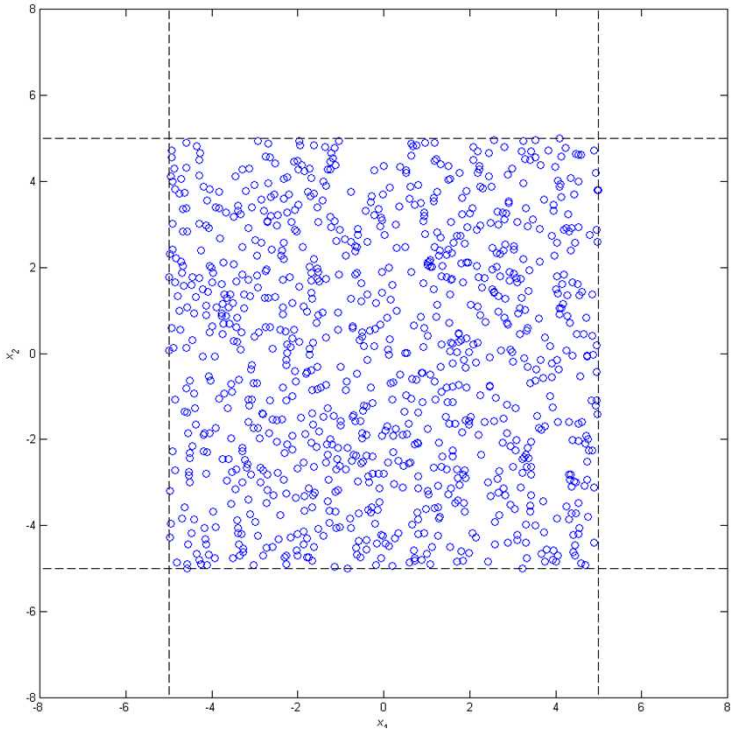
DIM	Cube	Sphere
2	1	0.785
3	1	0.524
10	1	0.00249
20	1	$2.46*10^{-8}$





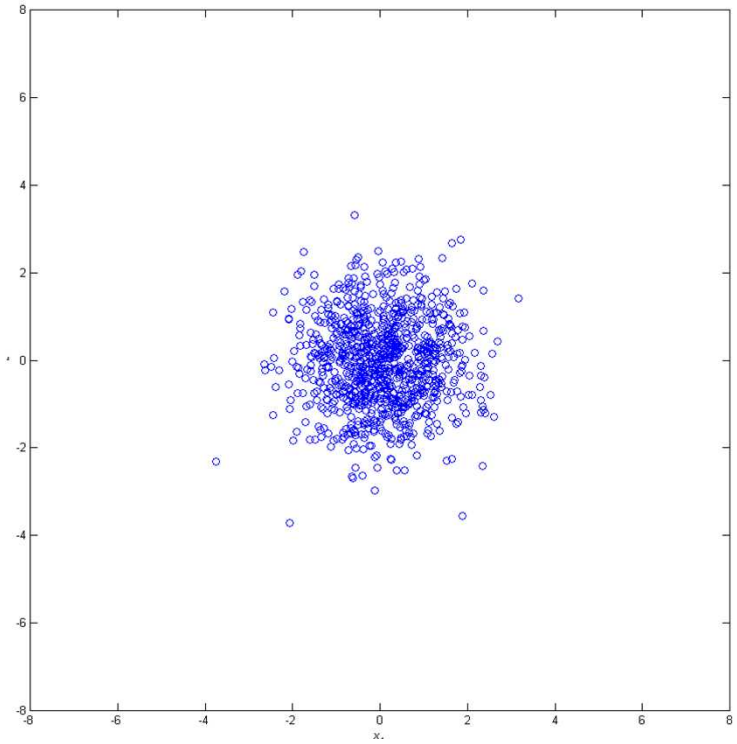
Sampling from hypercube

- ✓ Known methodology
- ✓ Fast and simple
- ✓ Enables adaptive sampling

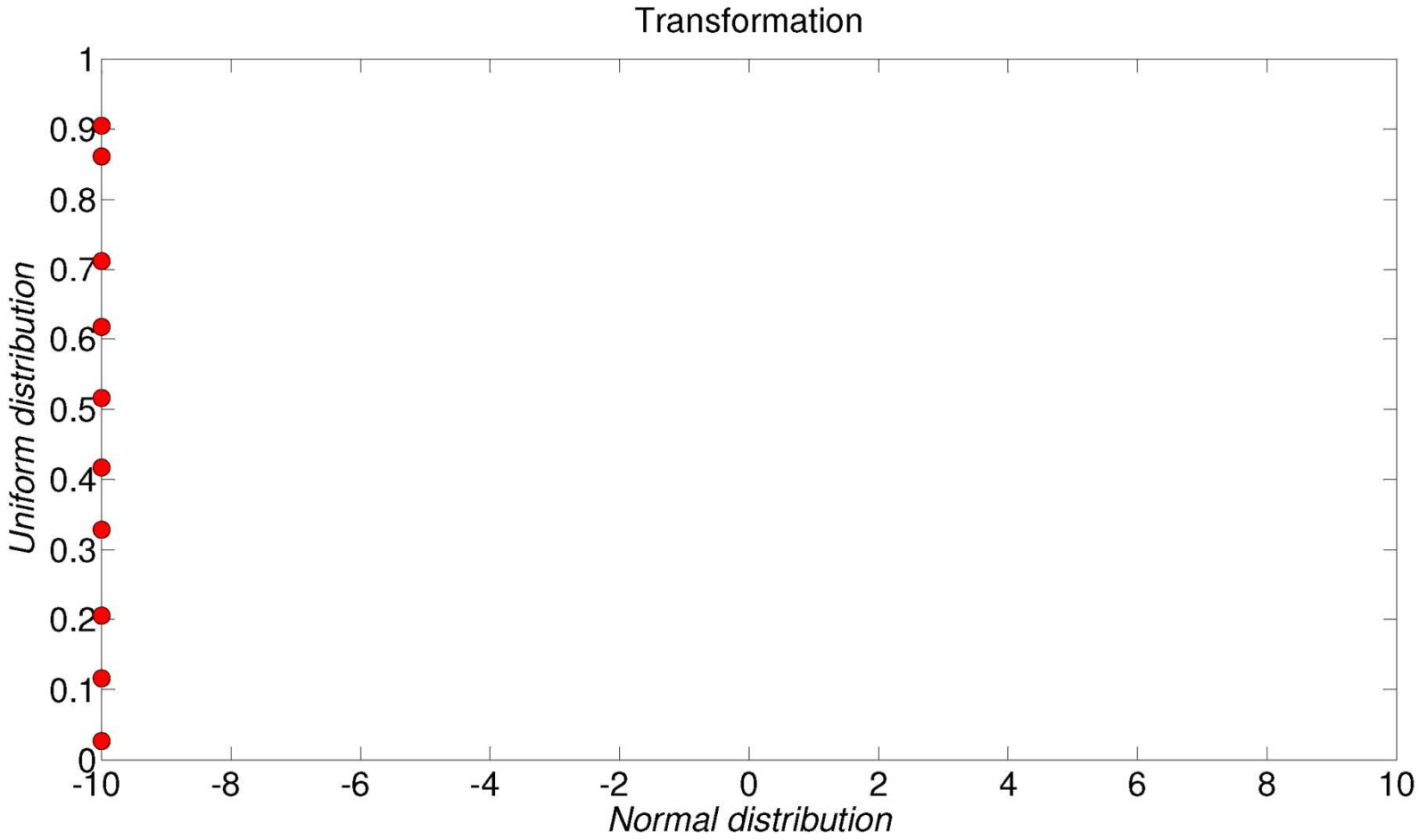


- ✗ Omits solutions outside bounds!

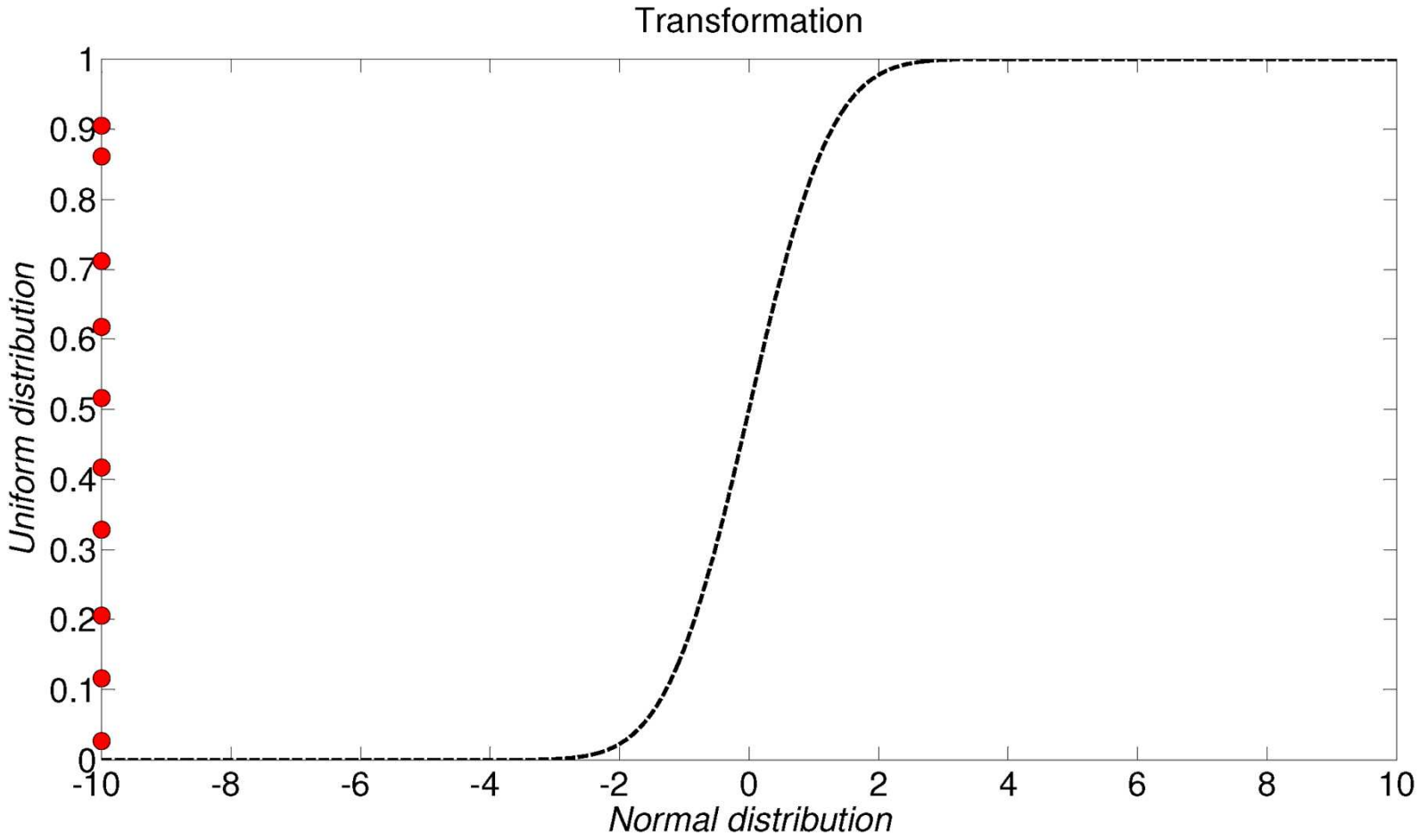
Sampling from prescribed distributions



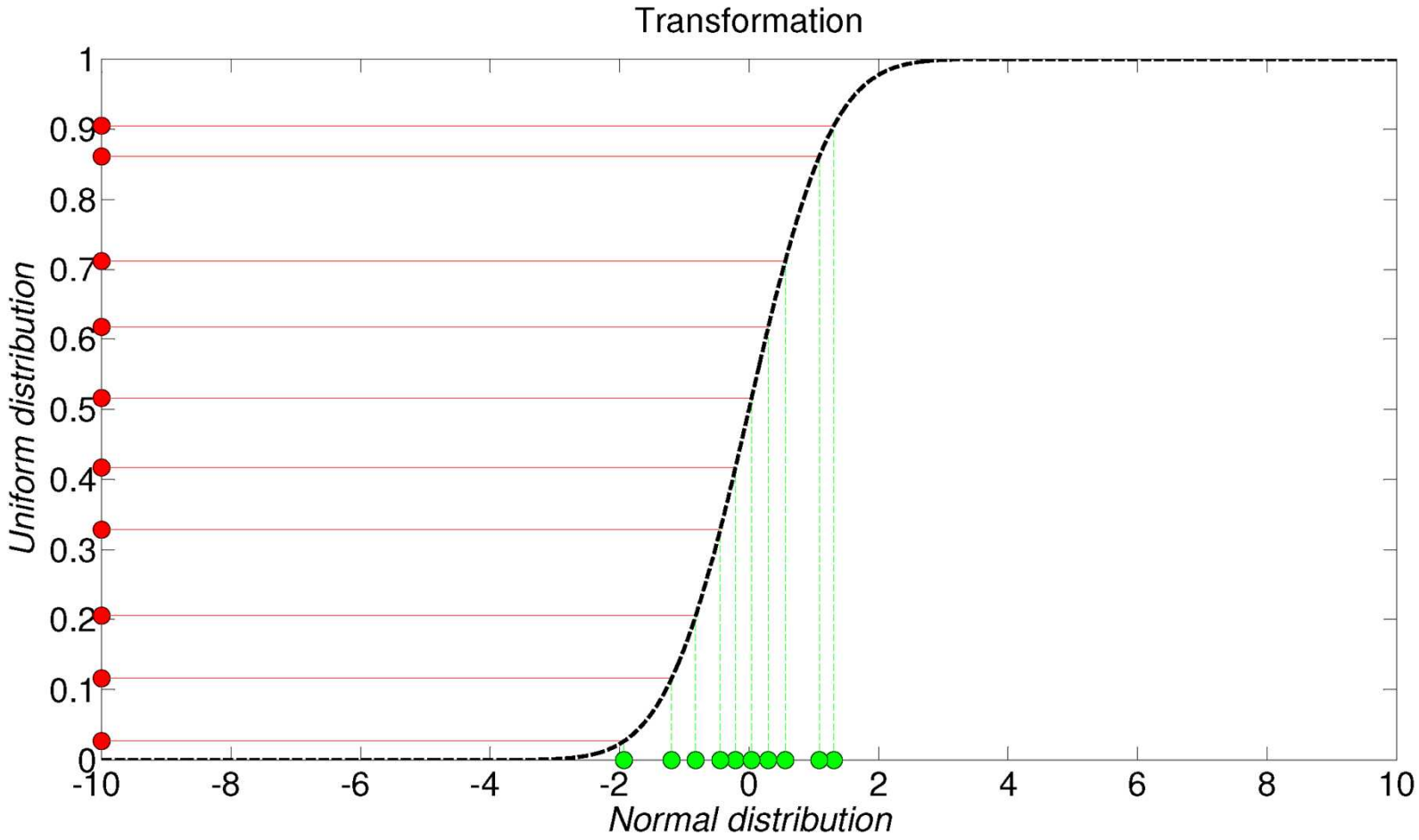
Transformation from uniform distribution



Transformation from uniform distribution



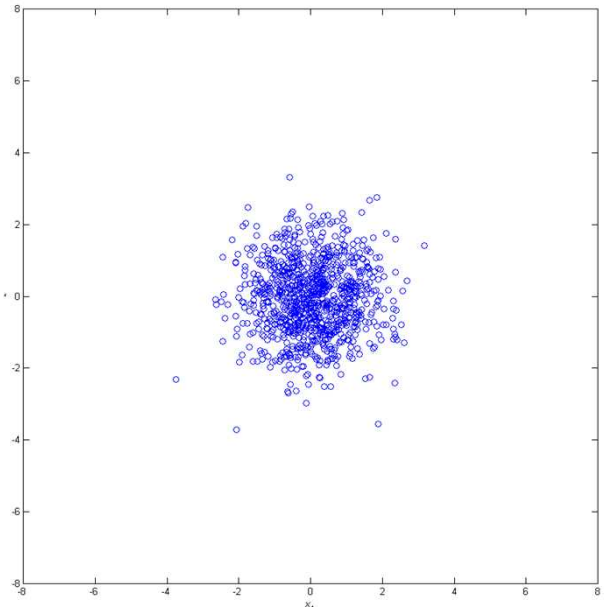
Transformation from uniform distribution



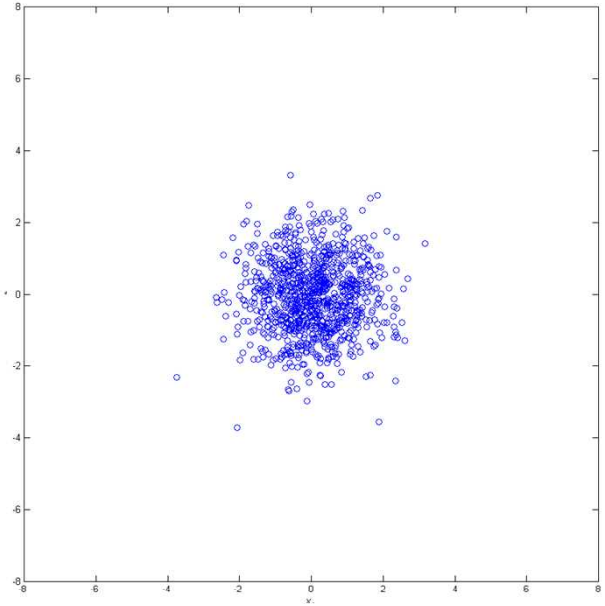
Sampling from prescribed distributions

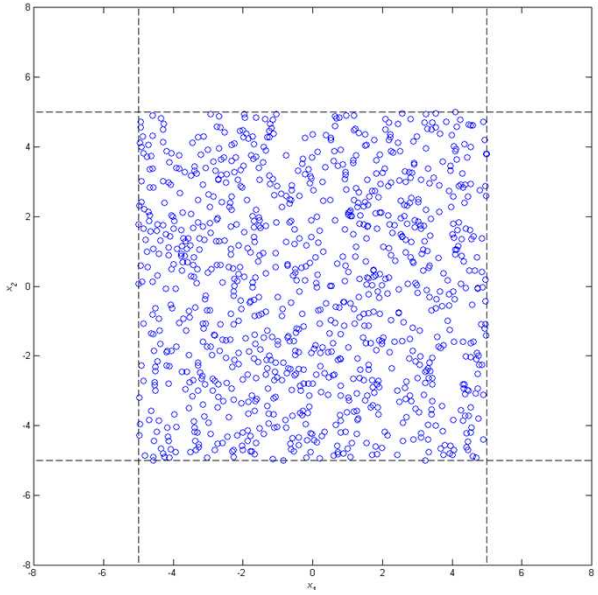
- ✓ Known methodology

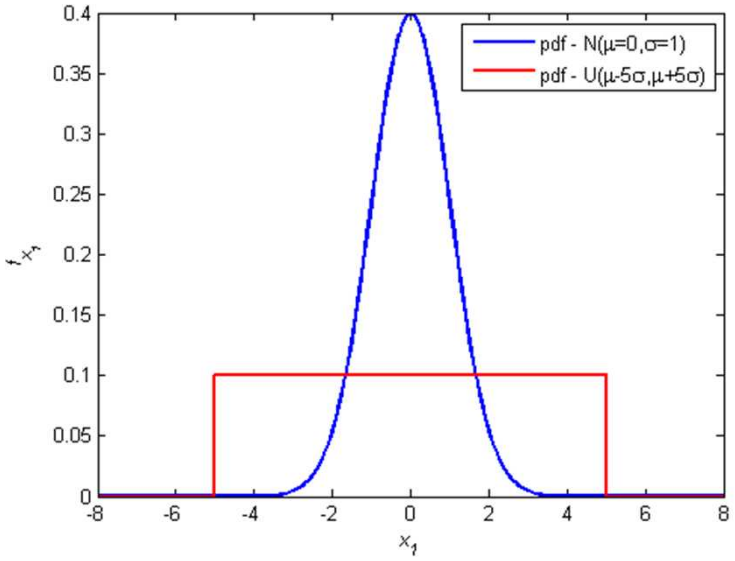
- ✗ Sampling around mean
- ✗ May miss failure region
- ✗ Problems with adaptive sampling



Hypercube vs. prescribed distribution?







Response Surface Methodology

- (unknown) function: $y(\mathbf{x}) = f(\mathbf{x}) + \boldsymbol{\varepsilon}$

error with normal distribution

$$E(\varepsilon_i) = 0, \quad V(\varepsilon_i) = \sigma^2$$

- approximation:

$$\hat{y}(\mathbf{x}) = \beta_0 + \sum_{i=1}^{N} \beta_i x_i + \sum_{i=1}^{N} \sum_{j \le i} \beta_{ij} x_i x_j \ \ldots$$

- in known points:

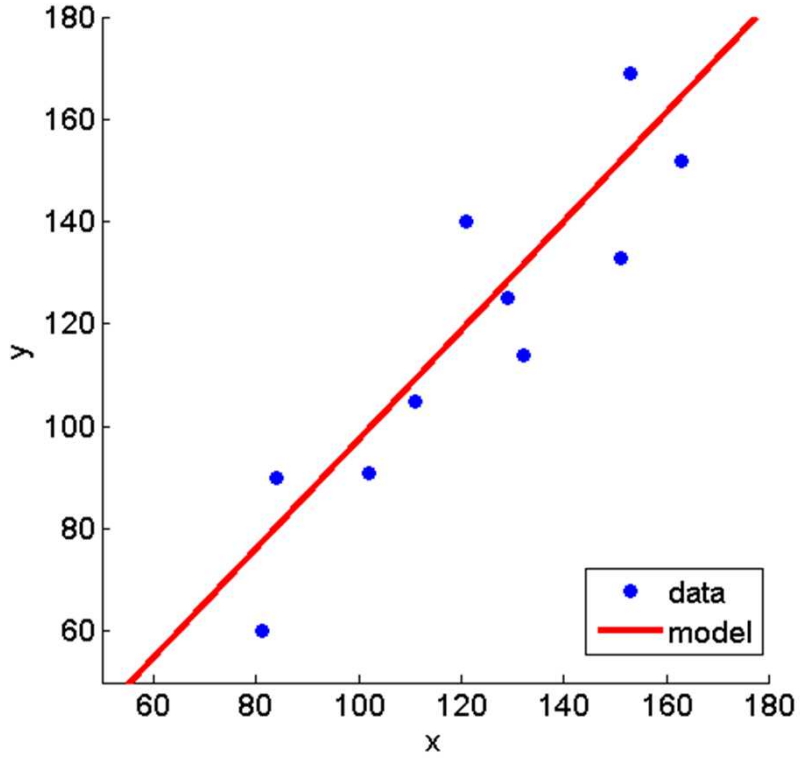
$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$$

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^{\mathbf{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathbf{T}}\mathbf{f}$$

Example: Linear regression

```
x = [121, 153, 132, 84, 102, 111, 163, 81, 151, 129];
y = [140, 169, 114, 90, 91, 105, 152, 60, 133, 125];
X = [ones(1,10) x];  % information matrix
Beta_app = (X‘* X)\(X‘*y‘)

Beta_app = -8.99340124551449
            1.0341760492707
```



x	y
121	140
153	169
132	114
84	90
102	91
111	105
163	152
81	60
151	133
129	125

Model:

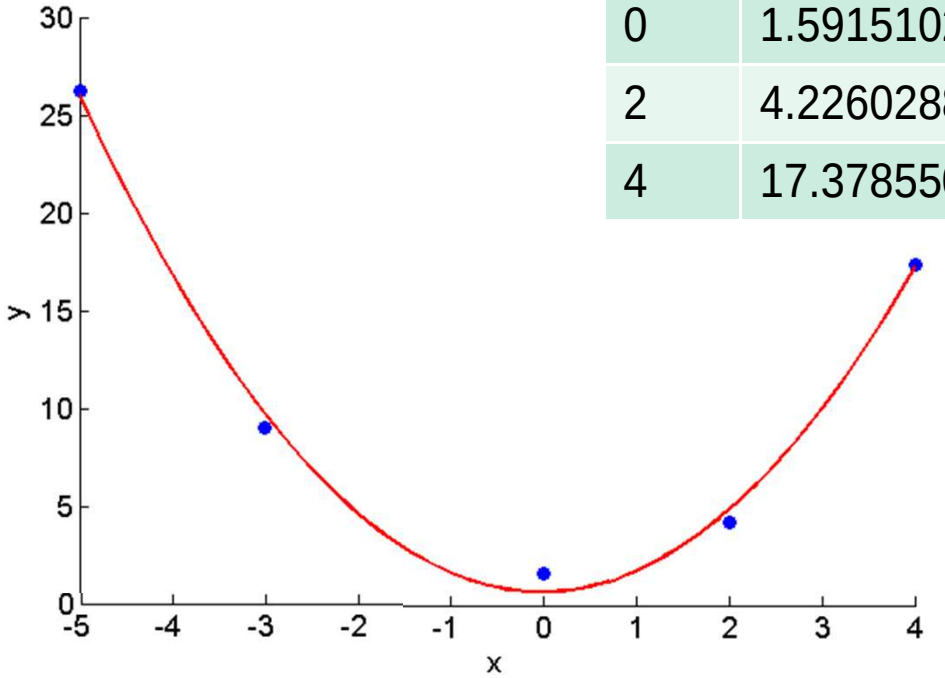
$$\hat{y} = -8.9934 + 1.0341x$$

Example: quadratic regression

```
x = [-5, -3, 0, 2, 4];
y = [26.274046,9.0840786,1.5915102,4.2260288,17.378550];
X = [ones(1,5) x‘ x‘.^2];  % information matrix
Beta_app = (X‘*X)\(X‘*y‘)

Beta_app = 0.679526827730912
           0.047343996311551
           1.02317162054266
```

x	y
-5	26.274046
-3	9.0840786
0	1.5915102
2	4.2260288
4	17.378550



Model:

$$\hat{y} = 0{,}67953 + 0{,}04734x + 1{,}02317x^2$$

Polynomial regression

- For complete polynomials in 2D

				1					Constant	Min. 1 point
			x		y				Linear	Min. 3 points
		x^2		xy		y^2			Quadratic	Min. 6 points
	x^3		x^2y		xy^2		y^3		Qubic	Min. 10 points
x^4		x^3y		x^2y^2		xy^3		y^4	4^{th} order	Min. 15 points
									m^{th} order	Min. (m+1)!

- Needed points for *n* dimensions and m^{th} order

$$\frac{(m+n)!}{m!\ n!}$$

Kriging

error with normal distribution and non-zero covariance

- (unknown) function : $y(\mathbf{x}) = f(\mathbf{x}) + \mathbf{Z}(\mathbf{x})$

$$E(Z_i) = 0, \quad V(Z_i) = \sigma^2, \quad \mathrm{Cov}[\mathbf{Z}(\mathbf{x}^i)\mathbf{Z}(\mathbf{x}^j)] = \sigma^2 \mathbf{R}([R(\mathbf{x}^i \mathbf{x}^j)]$$

$$\mathbf{R}(\mathbf{x}^i, \mathbf{x}^j) = \exp\left[\sum_{k=1}^{N} \theta_k \left| x_k^i - x_k^j \right|^2\right]$$

correlation matrix

- approximation:

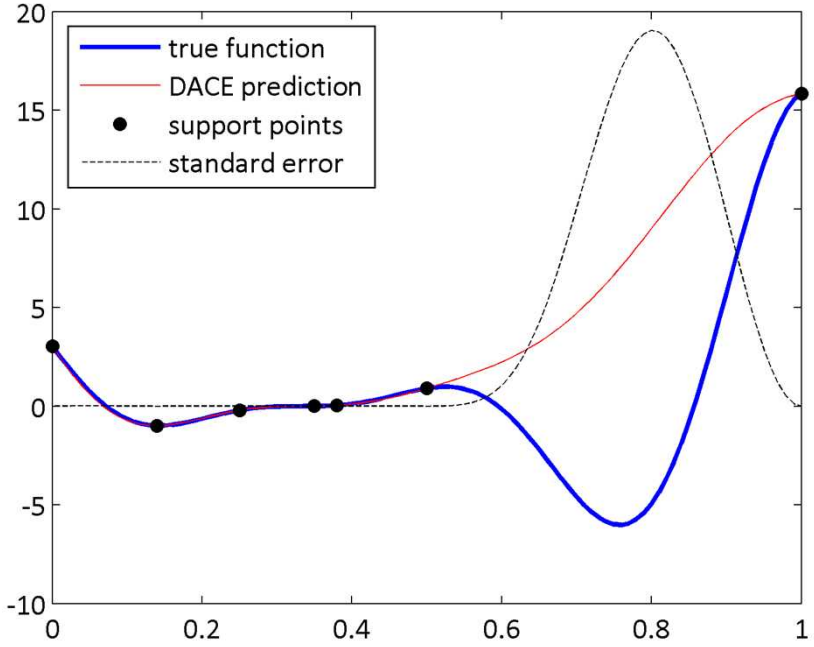
vector of correlations between the observed data and new prediction

$$\hat{y}(\mathbf{x}) = \beta + (\mathbf{y} - 1\beta)^T \mathbf{R}^{-1} \mathbf{r}(\mathbf{x})$$

Kriging

- Mean Squared Error: $$s^2(x^*) = \sigma^2 \left[1 - \mathbf{r}^T \mathbf{R}^{-1} \mathbf{r} + \frac{(1 - \mathbf{1}^T \mathbf{R}^{-1} \mathbf{r})^2}{\mathbf{1}^T \mathbf{R}^{-1} \mathbf{1}} \right]$$

- Standard error of Kriging prediction $$s(x^*) = \sqrt{s^2(x^*)}$$

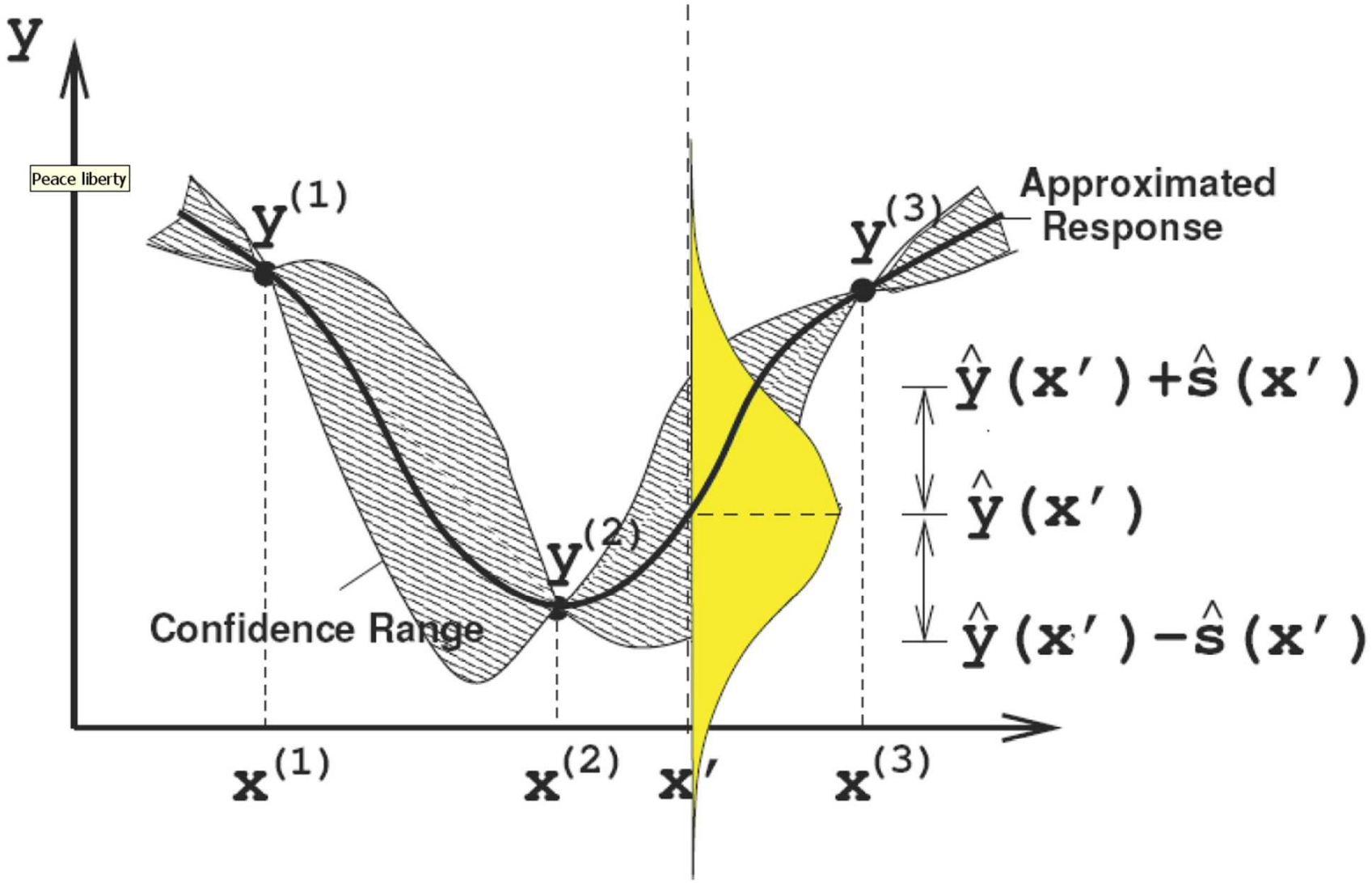


- $s(x*) = 0$ in support points
- $s(x*)$ is maximal in the most unknown areas

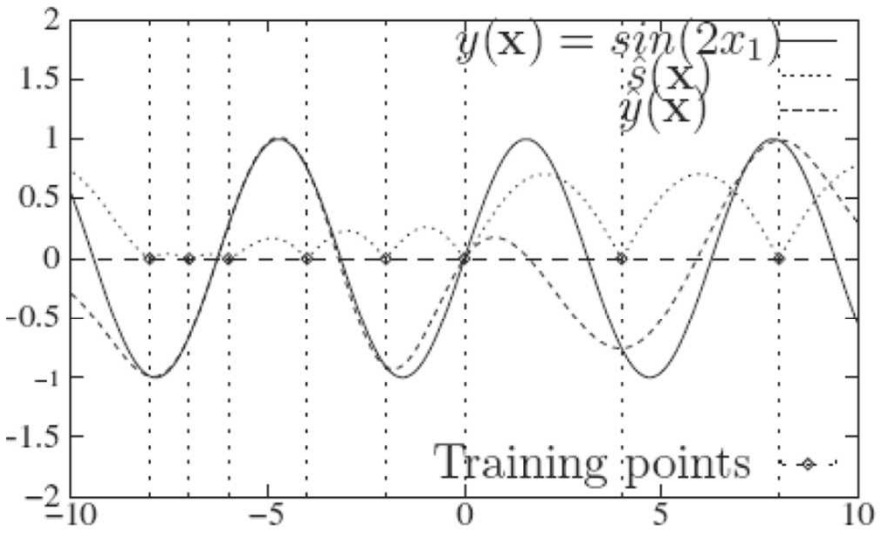
Note: true function:

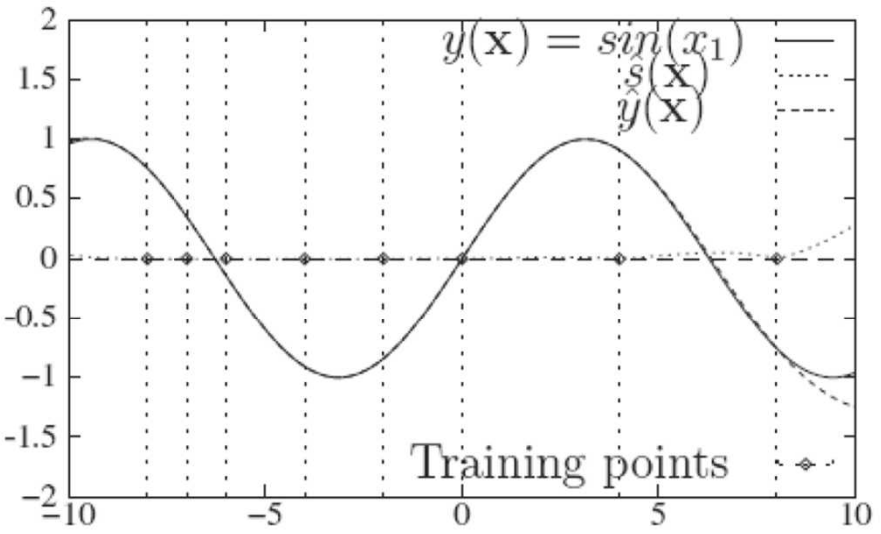
$$y(x) = (6x - 2)^2 \sin(12x - 4)$$

Kriging


y
Peace liberty
$y^{(1)}$
$y^{(2)}$
$y^{(3)}$
Approximated Response
$\hat{y}(x') + \hat{s}(x')$
$\hat{y}(x')$
$\hat{y}(x') - \hat{s}(x')$
Confidence Range
$x^{(1)}$
$x^{(2)}$
x'
$x^{(3)}$

Kriging





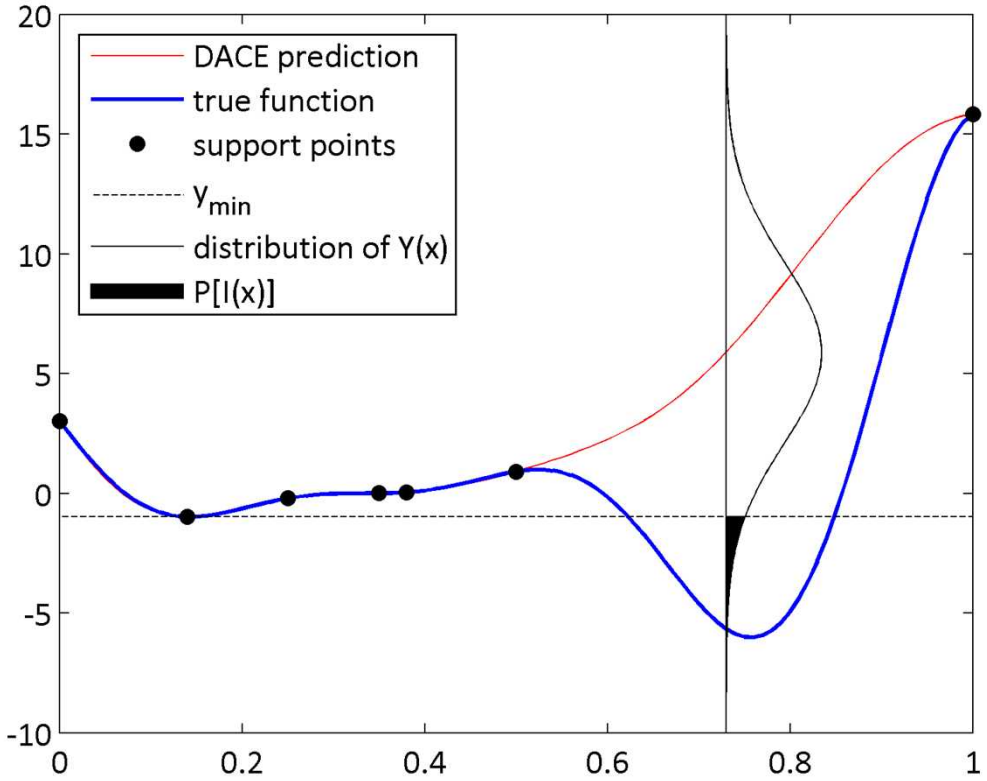
- Predicted values are precise in given points (interpolation)
- Error predictions are big on rough landscapes and oppositely small on flat ones
- Error predictions grow with increasing distance

Kriging

Probability of improvement

$$P[I(x)] = \frac{1}{s\sqrt{2\pi}} \int_{-\infty}^{0} \frac{(I - \hat{y}(x))^2}{2s^2} \mathrm{d}I$$

$$I = \max\left(y_{\min} - Y, 0\right)$$



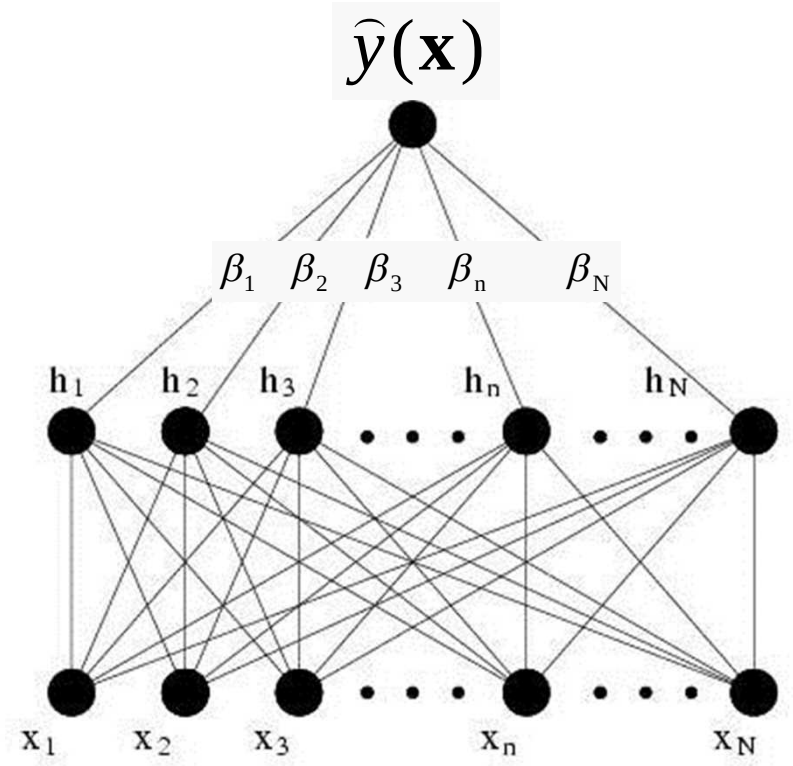
RBFN (Radial-Basis Function Network)

– Approximation:

$$\hat{y}(\mathbf{x}) = \sum_{i=1}^{N} \beta_i h_i(\mathbf{x})$$

Basis function : $h_i(\mathbf{x}) = e^{-\|\mathbf{x}-\mathbf{x}_i\|^2 / r}$

Weights β_i computed from equality of approximation and original function in training points ... leads to **a system of linear equations!!!**



Training points

Radial Basis Function Model

- approximation:

model parameters

basis function centres

response

$$\hat{f}(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{\psi} = \sum_{i=1}^{nc} w_i \psi\left(\left\|\boldsymbol{x} - \boldsymbol{c}^i\right\|\right) = \boldsymbol{y}$$

- bases functions:
  - $\psi(r) = r, \psi(r) = r^3, \psi(r) = \exp\left(\frac{-r^2}{2\sigma^2}\right), \ldots$
- Model parameters estimation:
  - $\boldsymbol{w} = \boldsymbol{\Psi}^{-1}\boldsymbol{y}$
  - $\Psi_{i,j} = \psi\left(\left\|\boldsymbol{x}^{(i)} - \boldsymbol{x}^{(j)}\right\|\right)$

$c^{(i)} = x^{(i)}$

Task of fitting metamodel

- Approximation usually much simpler than the optimization problem
- Approximation based only on DoE is imprecise => need for iterative approach

Algorithm ①

1. DoE creates new solutions, evaluates them on P
2. New solutions are added to M
3. Fitting of M
4. Optimization of M – get new guesses
5. New solutions are evaluated on P
6. While not *convergence*, go to 2

Algorithm ②

1. Start of Optimization Algorithm OA – usually DoE, evaluated on P, fitting of M
2. New solutions are created within OA
3. M creates „guesses“ of these new solutions
4. Based on “guesses” OA selects which solutions will be evaluated on P
5. New solutions evaluated on P, fitting of M
6. While not *convergence*, go to 2

Comparison of ① and ②

- Difference is in our trust in metamodel:
  - Algorithm ① is based on *full trust* (meta-model controls optimization)
  - Algorithm ② is based on *distrust* (optimization uses metamodel on demand)

RBFN (Radial-Basis Function Network)

– Approximation:

$$\hat{y}(\mathbf{x}) = \sum_{i=1}^{N} \beta_i h_i(\mathbf{x})$$

Weights β_i computed from equality of approximation and original function in training points ... leads to **a system of linear equations!!!**

Basis function : $h_i(\mathbf{x}) = e^{-\|\mathbf{x}-\mathbf{x}_i\|^2 / r}$

– optimum of approximation found by GA

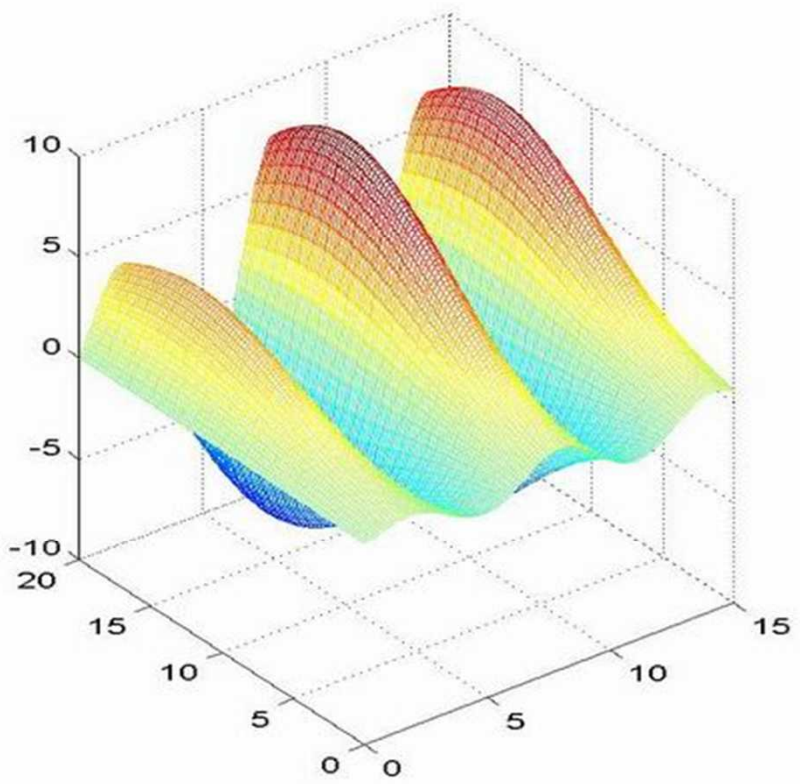
– Addition of new training points

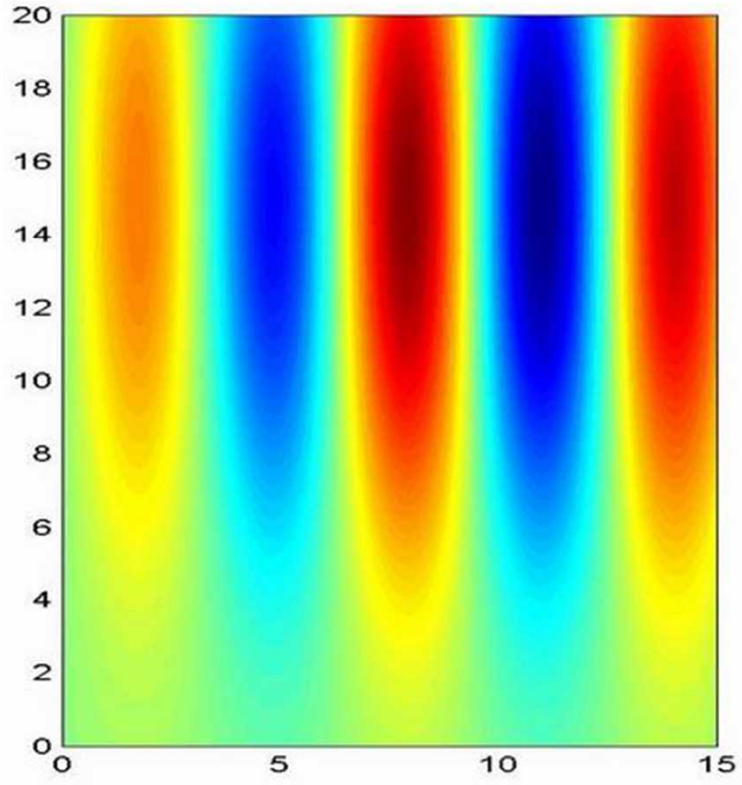
- Optimum found by GA
- Random point
- New point in the direction of two last optima (simple gradient)

RBFN

– Simple example

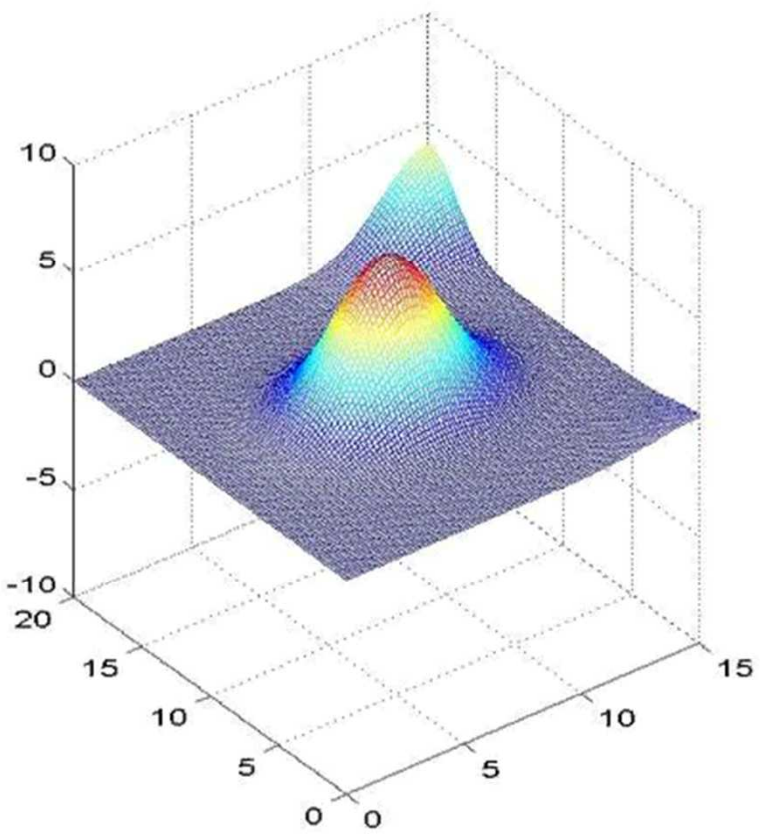
$$ex1(x, y) = 10e^{\left(-0.01(x-10)^2 - 0.01(y-15)^2\right)} \sin(x)$$

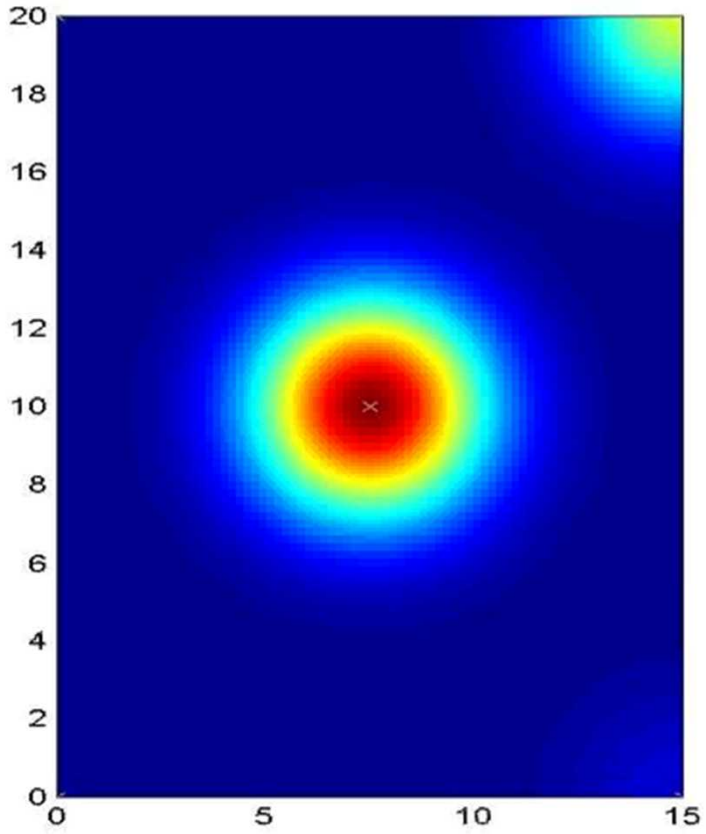




RBFN

– Example ex1 with GRADE algorithm

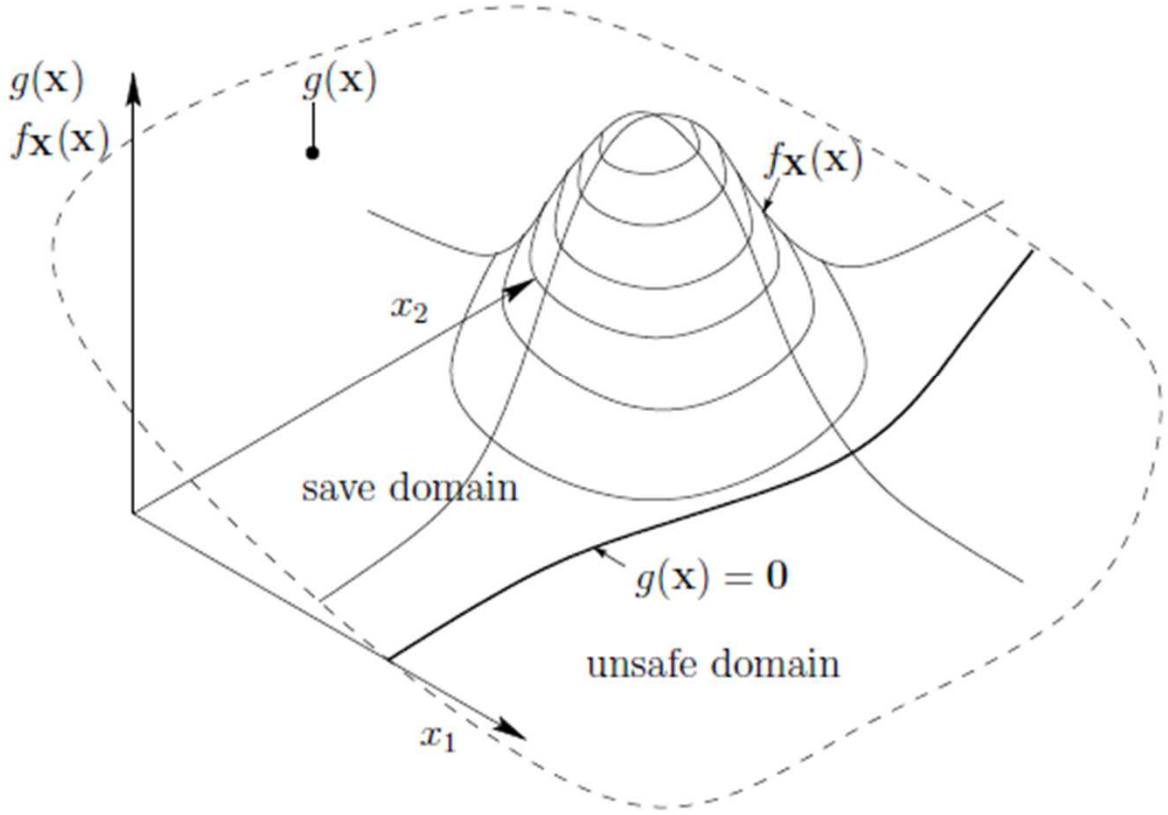




RBFN

Test function	N	SADE		GRADE		GRADE+CERAF		GRADE+RBFN	
		SR %	ANFC	SR %	ANFC	SR %	ANFC	SR %	ANFC
F1	1	100.0	61	100.0	61	100.0	60	100.0	23
F3	1	100.0	87	100.0	97	100.0	94	96.7	159
Branin	2	100.0	668	100.0	371	100.0	368	100.0	43
Camelback	2	100.0	306	100.0	223	100.0	222	100.0	61
Goldprice	2	100.0	634	100.0	360	100.0	358	11.6	472
PShubert1	2	100.0	1518	100.0	5501	100.0	1844	2.1	466
PShubert2	2	100.0	1043	100.0	1403	100.0	970	2.5	530
Quartic	2	100.0	534	100.0	341	100.0	339	100.0	77
Shubert	2	100.0	682	100.0	649	100.0	654	18.0	506
Hartman1	3	100.0	478	100.0	319	100.0	320	99.9	63
Shekel1	4	100.0	7719	100.0	33776	100.0	3434	0.0	-
Shekel2	4	100.0	4595	100.0	13522	100.0	2638	0.0	-
Shekel3	4	100.0	4127	100.0	10857	100.0	2650	0.0	-
Hartman2	6	71.2	57935	60.8	165622	100.0	10284	97.7	163
Hosc45	10	100.0	7759	100.0	2265	100.0	2274	-	-
Brown1	20	91.1	160515	100.0	209214	100.0	195250	-	-
Brown3	20	100.0	60554	100.0	36339	100.0	36429	-	-
F5n	20	94.4	26786	99.8	7197	100.0	7259	-	-
F10n	20	66.4	227577	70.3	90687	98.2	289702	-	-
F15n	20	97.5	48533	99.4	23358	100.0	24894	-	-

Adaptive sampling around LSF



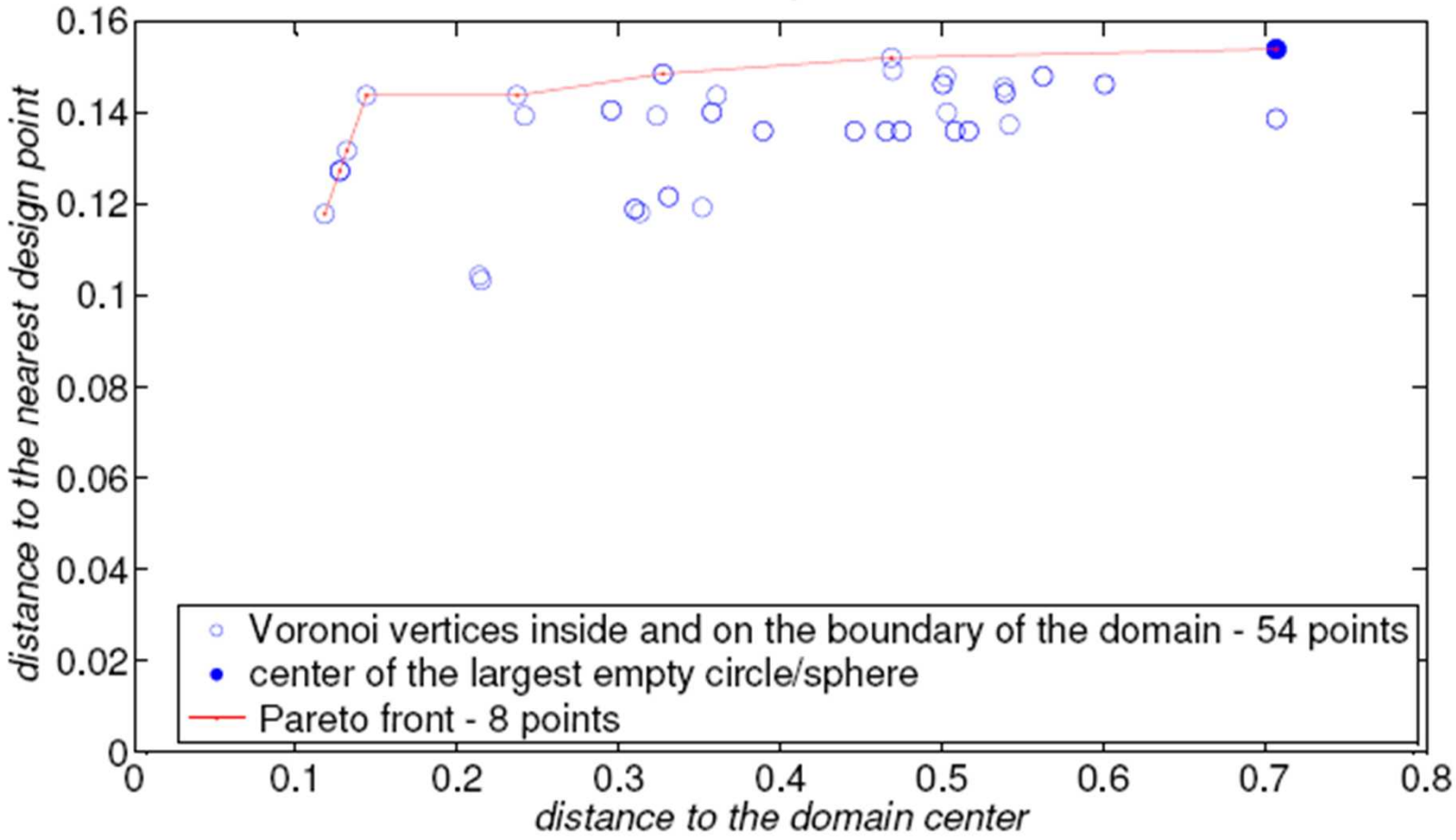
[Roos, 2006]

Surrogate Model

- Appropriate number of sampling points is needed
- Adaptive updating procedure
  - Multi-objective optimization problem
    - Maximization of the nearest distance of the added point from already sampled points (like miniMax metric)
    - To be as close as possible to the approximate limit state surface

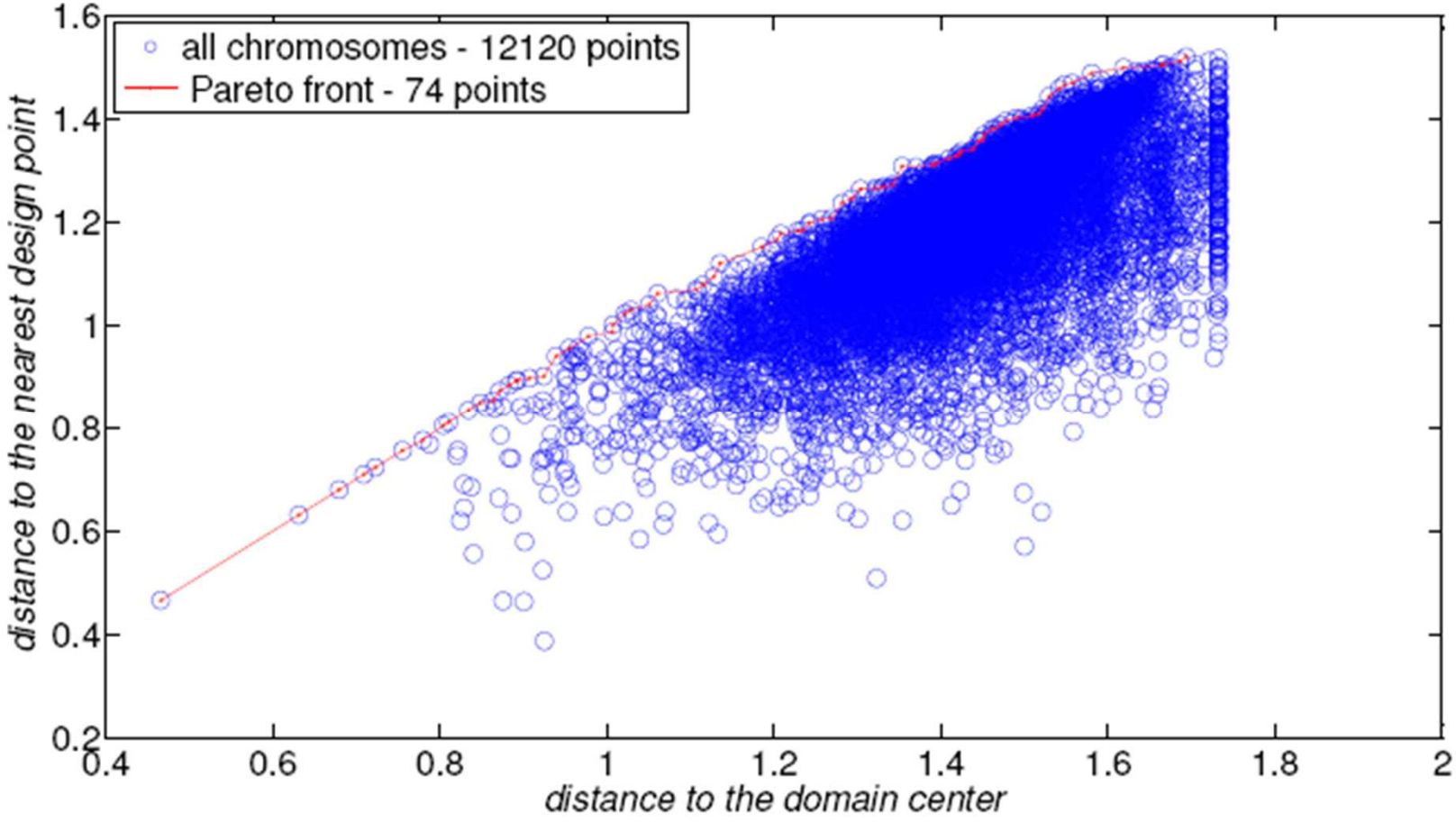
Multi-objective adaptive sampling

2D, 27 points



Multi-objective adaptive sampling

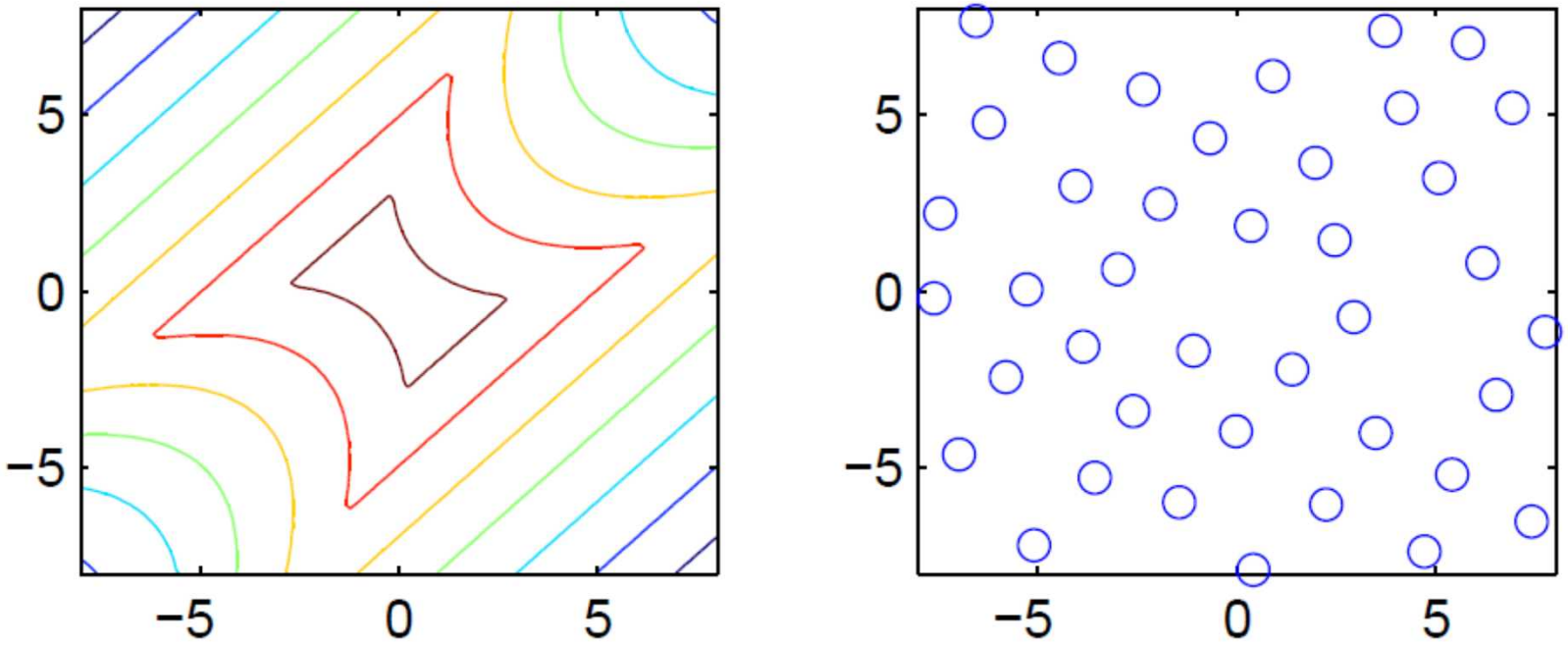
12D, 65 points



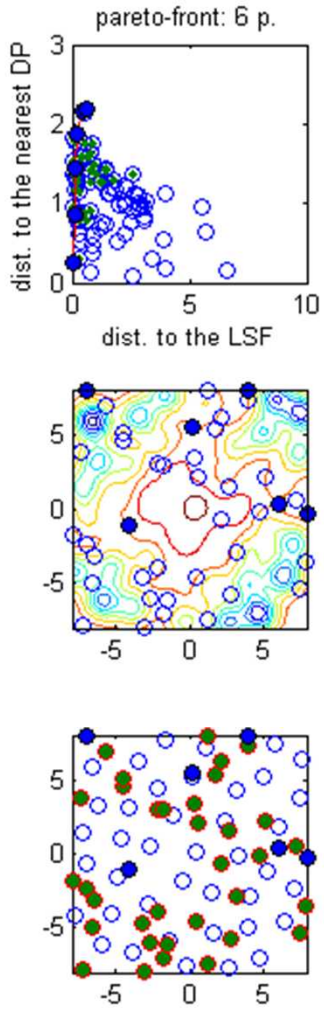
Implemented Meta-Models

- RBFN from Matlab
  - Neural Network based
- CTU implementation of RBFN
  - with different polynomial regression parts
- Kriging
  - DACE toolbox in Matlab
  - with different polynomial regression parts
  - with regression part found by Genetic Programming

Adaptive update of meta-model



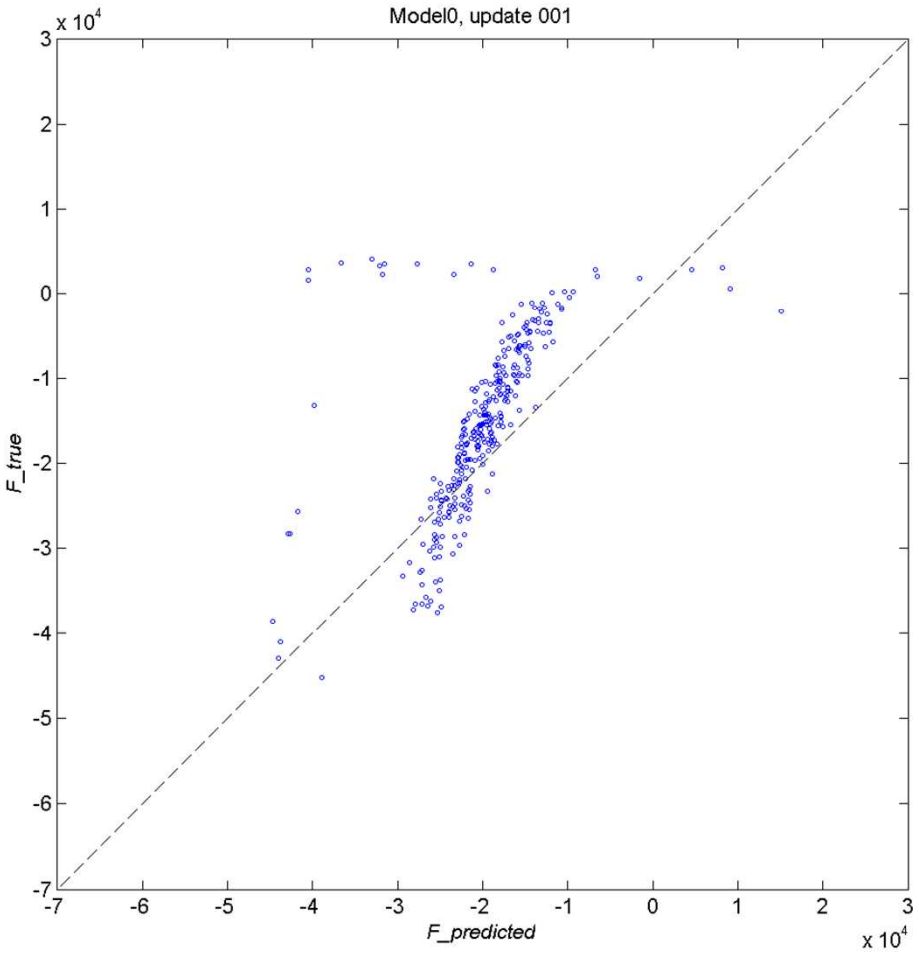
Contours of the example (left) and starting DoE (right). Note that the red contour is for F(x) = 0.



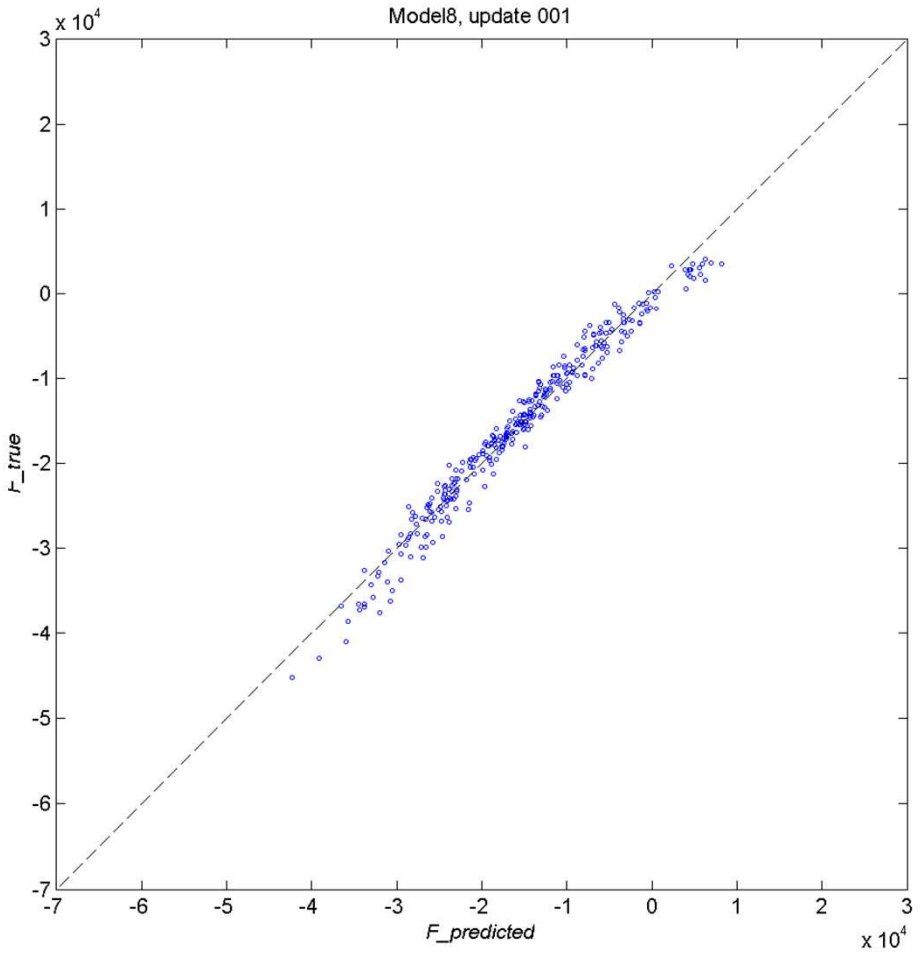
Pareto front (top), contours of the problem with DoEs (middle) and DoEs' points (bottom).
Key: Red – added and computed solutions, Blue – points that were too close to other Pareto front points, Green – the remaining points of population and Blue empty points – the original DoE.

Quality of a metamodel

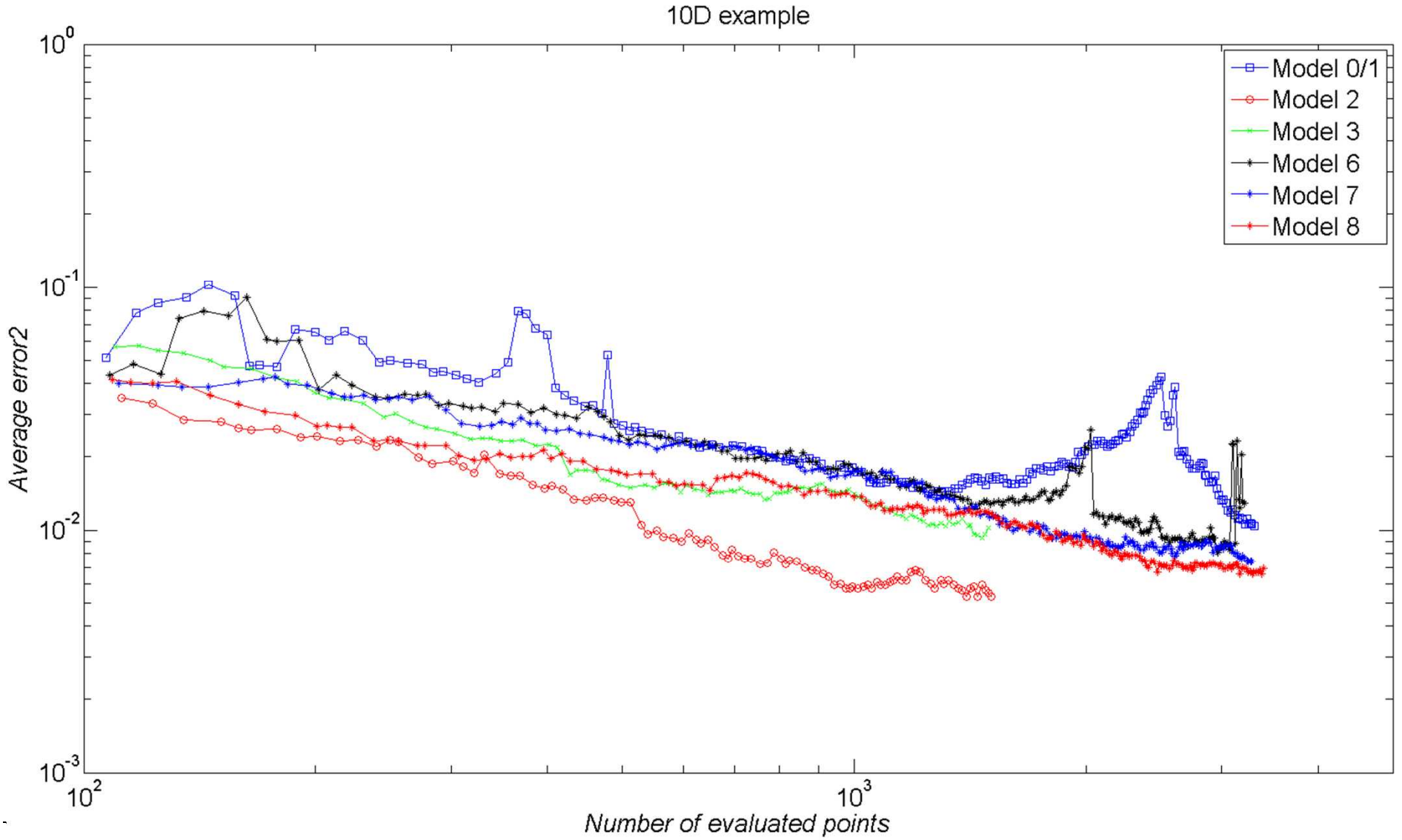
RBFN (Matlab)



Kriging



Quality of updating procedure



References

[1] Jin, Y. (2003) A comprehensive survey of fitness approximation in evolutionary computation. Soft Computing Journal, 9(1):3-12.

[2] T. W. Simpson, J. D. Peplinski, P. N. Koch and J. K. Allen. (2001) Metamodels for Computer-based Engineering Design: Survey and recommendations. Engineering with Computers 17: 129–150.

[3] H. Nakayama, K. Inoue and Y. Yoshimori (2004) Approximate optimization using computational intelligence and its application to reinforcement of cable-stayed bridges, ECCOMAS.

[4] M. K. Karakasis and K. C. Giannakoglou (2004) On the use of surrogate evaulation models in multi-objective evolutionary algorithms, ECCOMAS.

References

[5] Ibrahimbegovič, A., Knopf-Lenoir, C., Kučerová, A., Villon, P., (2004) Optimal design and optimal control of elastic structures undergoing finite rotations, International Journal for Numerical Methods in Engineering.

[6] Forrester, A., Sobester A., and Keane A., (2008) Engineering design via surrogate modelling: a practical guide. John Wiley & Sons.

[7] Weise, Thomas, et al. "Why is optimization difficult?" *Nature-Inspired Algorithms for Optimisation*. Springer Berlin Heidelberg, 2009. 1-50.

[8] D. C. Montgomery. Design and Analysis of Experiments, 5th Edition. Wiley, June 2000.

[9] R. H. Myers and D. C. Montgomery. Response Surface Methodology: Process and Product Optimization Using Designed Experiments. New York: Wiley, 1995.

Some of the parts of this lecture has been kindly provided by Adéla Hlobilová (Pospíšilová) at CTU in Prague, Faculty of Civil Engineering.

A humble plea. Please feel free to e-mail any suggestions, errors and typos to **`matej.leps@fsv.cvut.cz`**.

Date of the last version: 28.11.2019
Version: 003