RESPONSE SURFACE METHODOLOGIES - METAMODELS

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

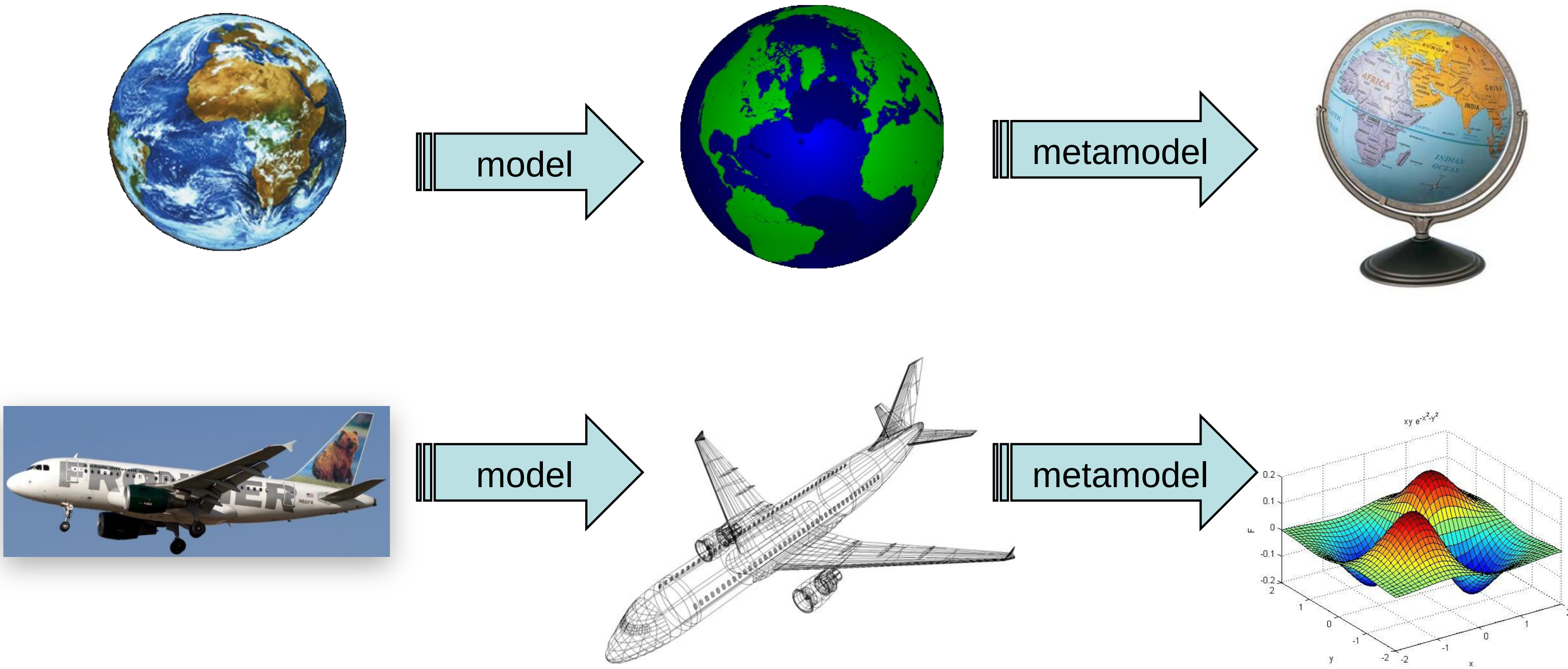
modeFRONTIER
the multi-objective optimization and design environment

Metamodels

Metamodels (or surrogate models, response surface models - RSM), are analytic models that approximate the multivariate input/output behavior of complex systems, based on a limited set of computational expensive simulations.

It can be used to predict the system response in unknown configurations, for design automation, parametric studies, design space exploration, optimization and sensitivity analysis. Extremely useful when many runs are needed but the cost of a single run is very high.

Metamodels



Metamodels

- A collection of mathematical and statistical techniques useful for the modeling of problems
- RSM is used in engineering design to construct approximations of analysis codes
- Predictions made within the observed space of variable values are called **interpolations**. Predictions outside the observed values are called **extrapolations** and require caution
- Export formats: JAVA, C, FORTRAN, EXCEL

modeling sequence for engineering design

1. **Model formulation**: in the workflow, identify the problem's input and output parameters;

2. **Design selection**: using DOE tool, specify the variable settings at which to run the disciplinary models. The responses obtained from these runs are used for fitting the RSM.

3. **RSM fitting**: requires specifying the type of the RSM.

4. **Assessment of the RSM:** involves specifying the performance measures that will be used to characterize the fidelity of a RSM

5. **Using the RSM:** predicting responses at untried inputs and performing optimization runs, trade-off studies, or further exploring the design space.

Metamodels for optimization

DOE

↓

Statistical analysis

↓

Metamodels

↓

Optimization

The cost of a single run is negligible

↓

Validation

Of optimized solutions only

Metamodels can be improved every time we have a new reliable information on the system behavior

↓

Expert systems

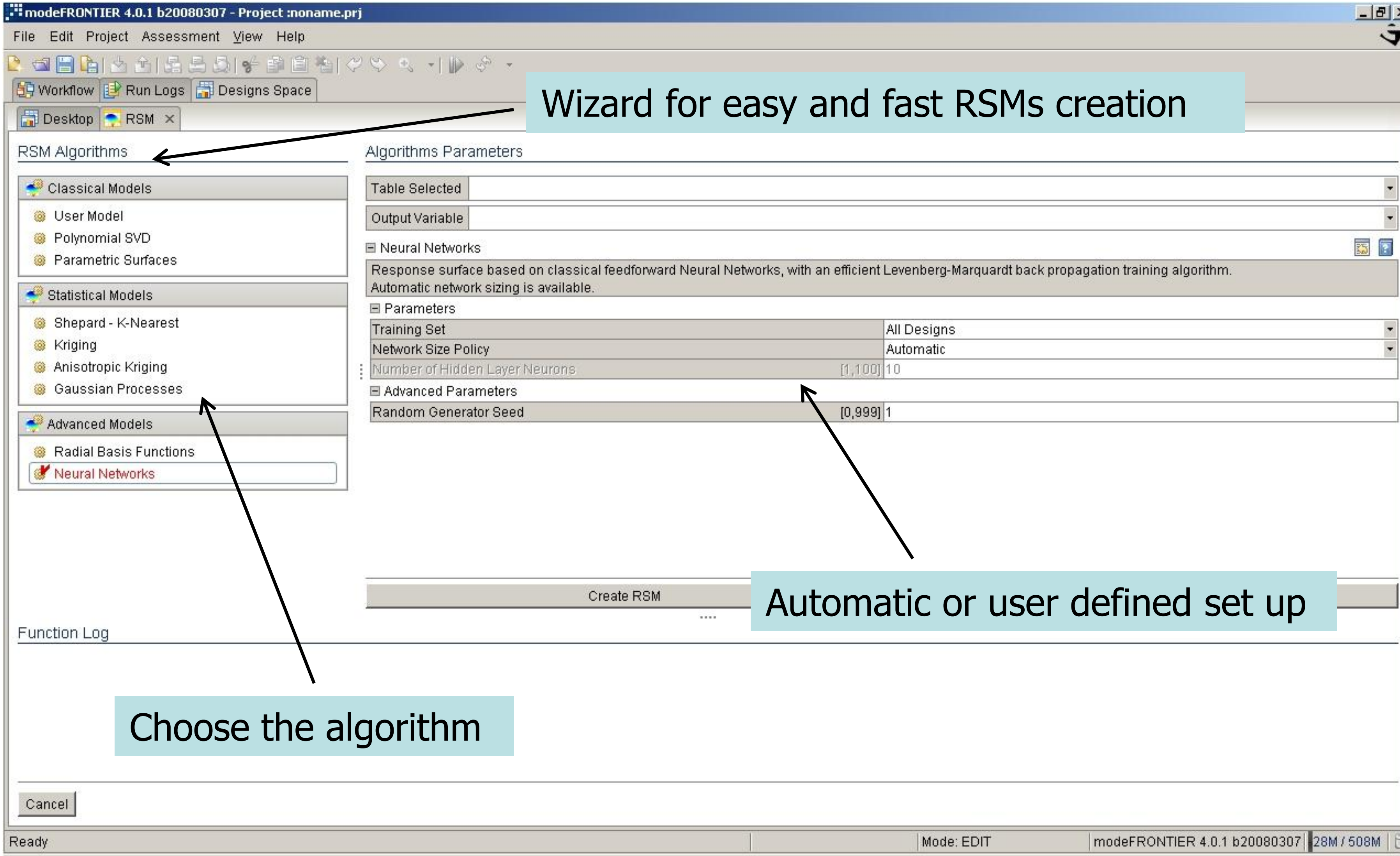
RSM: methods in modeFRONTIER

Classical Models
- User Model
- Polynomial
- Parametric

Statistical Models
- K-nearest
- Kriging
- Gaussian process

Advanced Models
- Radial basis functions
- Neural network
- Evolutionary Design

RSM: the wizard



Radial Basis Functions

- Radial Basis Functions (RBF) are a powerful tool for multivariate scattered data interpolation.

- *Scattered* data means that training points are not sampled on a regular grid: in fact RBF is a *meshless* method.

- RBF are *interpolant* response surfaces: they pass exactly through training points.

- Despite their simple formulation RBF hide a sound theoretical framework.

Radial Basis Functions

$f(\mathbf{x}_i) = f_i \,, \quad i = 1, \ldots, n$ training dataset

$s(\mathbf{x}) = \sum_{j=1}^{n} c_j \phi \left(\|\mathbf{x} - \mathbf{x_j}\| / \, \delta \right)$ RBF model

δ scaling parameter

Euclidean norm (distance)

$\phi(r)$ radial function

c_j coefficients: free parameters of RBF model

$s(\mathbf{x}_i) = f(\mathbf{x}_i) = f_i \,, \quad \forall i = 1, \ldots, n$ interpolation equations

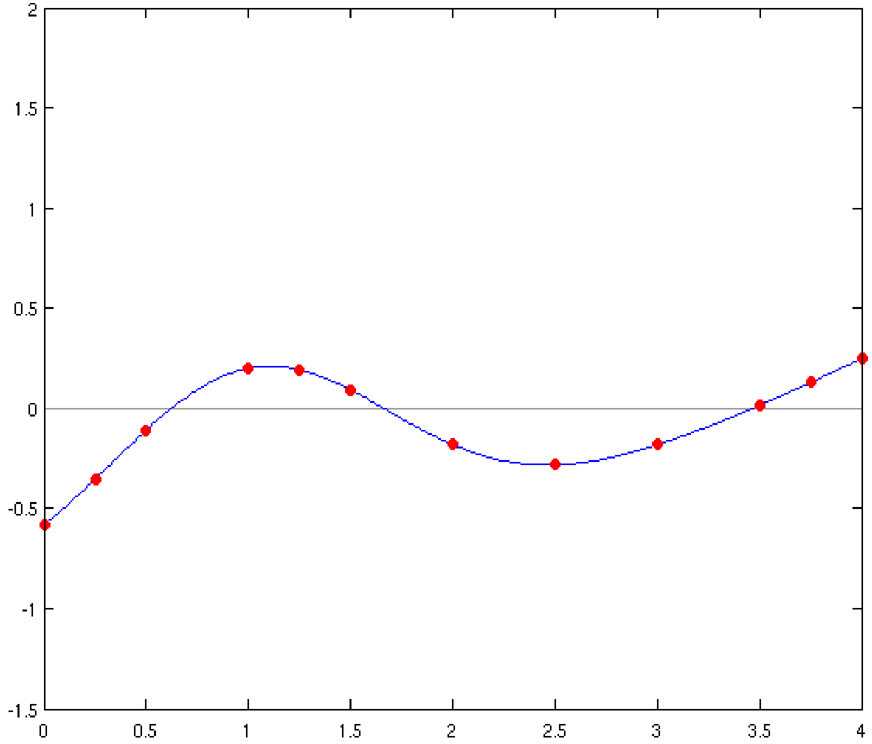
RBF interpolant is simply a linear combination of radial functions centered at training points

the radial function is a suitable fixed function chosen out of a given list

coefficients are obtained solving the interpolation equations (linear system)

Radial Basis Functions

Example



RBF model

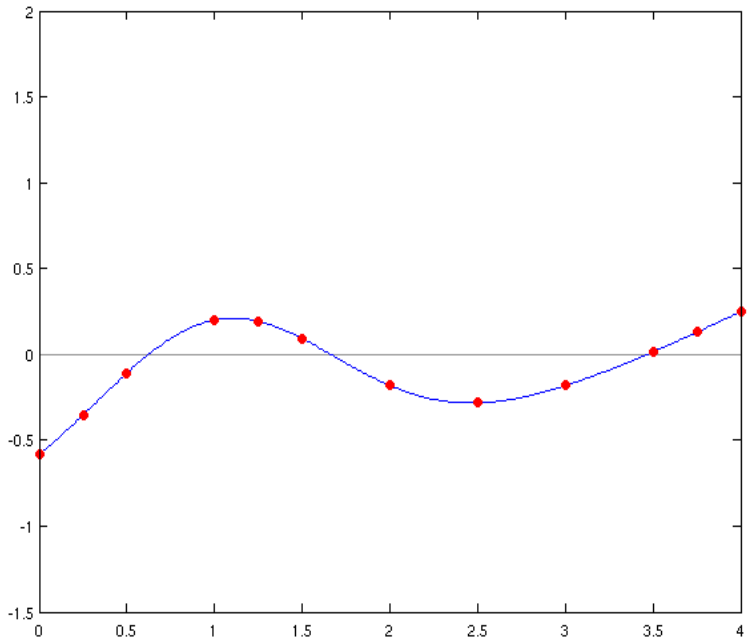
$$s(\mathbf{x}) = \sum_{j=1}^{n} c_j \phi\left(\|\mathbf{x} - \mathbf{x_j}\| / \delta\right)$$

RBF-G

$\phi(r)$ Gaussians radial functions

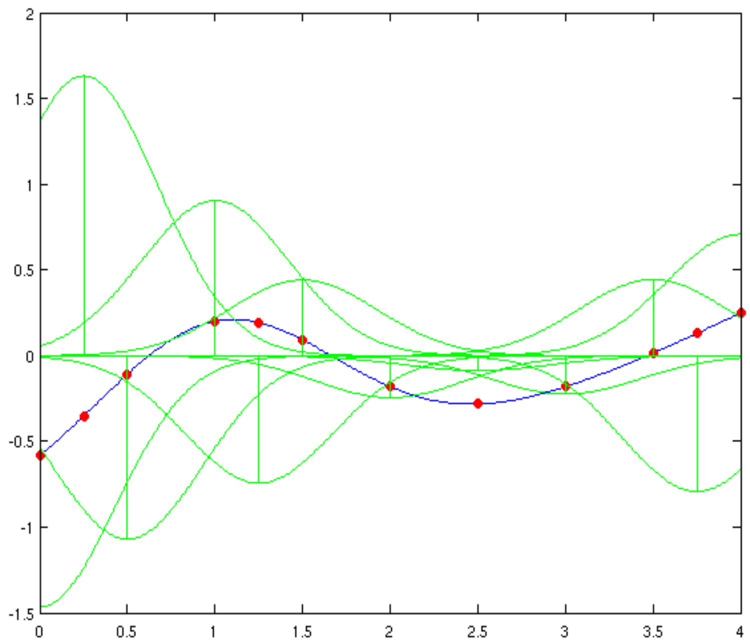
Radial Basis Functions

-Example



-RBF model $s(\mathbf{x}) = \sum_{j=1}^{n} c_j \phi\left(\|\mathbf{x} - \mathbf{x_j}\| / \delta\right)$

-RBF-G $\phi(r)$ -Gaussians radial functions



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

Radial Basis Functions

RBF interpolant

$$s(\mathbf{x}) = \sum_{j=1}^{n} c_j \phi \left(\|\mathbf{x} - \mathbf{x_j}\| / \, \delta \right)$$

proper for so-called *positive definite* (PD) radial functions

(the matrix of the linear system S=C*P of interpolation equations is positive definite)

in case of so-called *conditionally positive definite* (CPD) radial functions:

$$s(\mathbf{x}) = \sum_{j=1}^{n} c_j \phi \left(\|\mathbf{x} - \mathbf{x_j}\| / \, \delta \right) + p_m(\mathbf{x})$$

additional polynomial term is required

Radial Basis Functions

- Five different radial functions are available:
- Gaussians (G)
- Duchon's Polyharmonic Splines (PS)
- Hardy's MultiQuadrics (MQ)
- Inverse MultiQuadrics (IMQ)
- Wendland's Compactly Supported C^2 (W2)

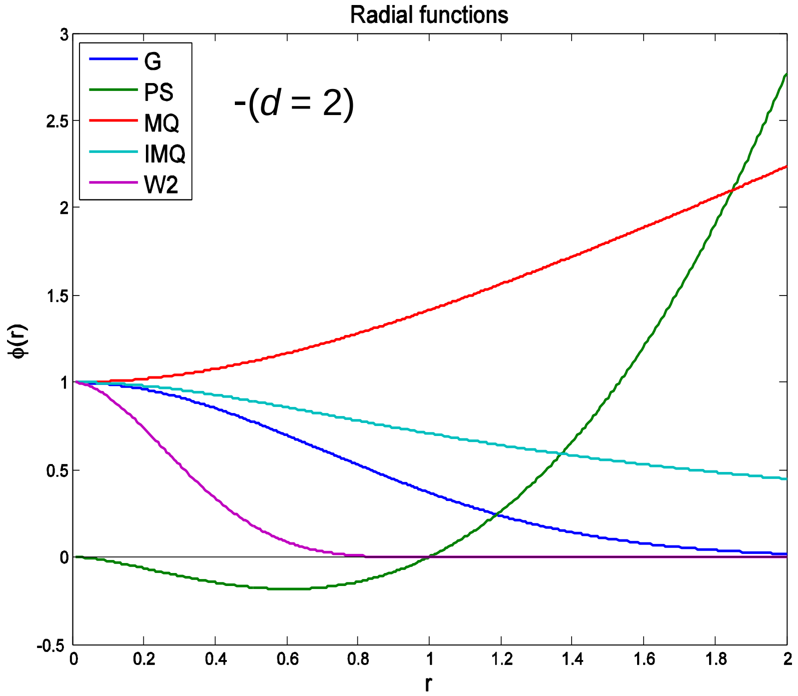
This list represents a complete set of state of the art and widely used radial functions that can be found in literature.

PS include Thin-Plate Splines (for d = 2)
and the usual Natural Cubic Splines (for d = 1).

Radial Basis Functions

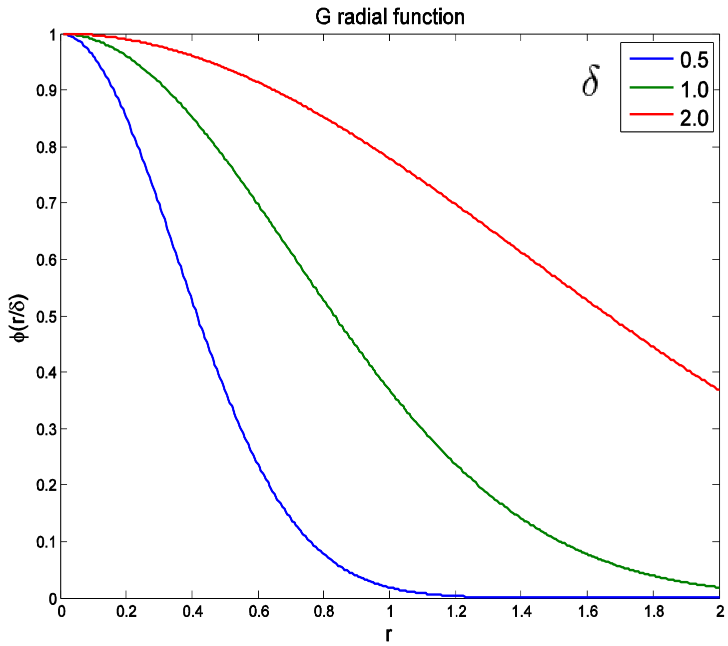
Available radial functions:

G	$\phi(r) = \exp(-r^2)$	PD
PS	$\phi(r) = \begin{cases} r^3 & d \text{ odd} \\ r^2 \log(r) & d \text{ even} \end{cases}$	$m = (d+1)/2$ $m = d/2$
MQ	$\phi(r) = (1+r^2)^{(1/2)}$	$m = 0$
IMQ	$\phi(r) = (1+r^2)^{(-1/2)}$	PD
W2	$\phi(r) = \begin{cases} (1-r)_+^3\,(3r+1) & d=1 \\ (1-r)_+^4\,(4r+1) & d=2,3 \\ (1-r)_+^5\,(5r+1) & d=4,5 \end{cases}$	PD



Radial Basis Functions

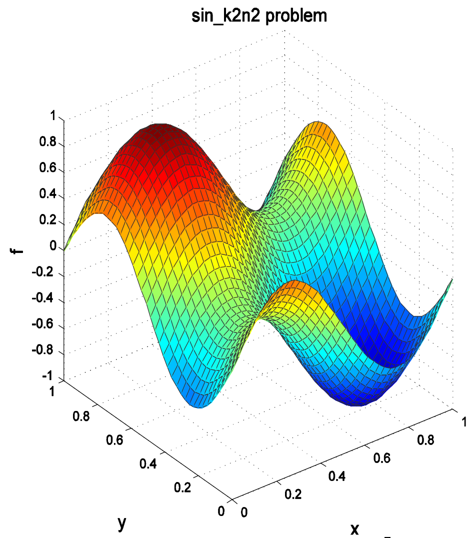
The scaling parameter determines the shape of the radial function.



Its value has to be set accordingly to the specific problem one is facing.

The scaling parameter affects both the computational difficulty of the interpolation equations (*condition number*), and the goodness of the RBF model.

Radial Basis Functions

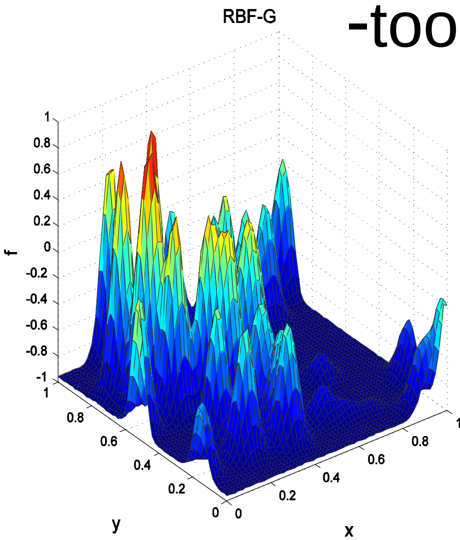


-Example

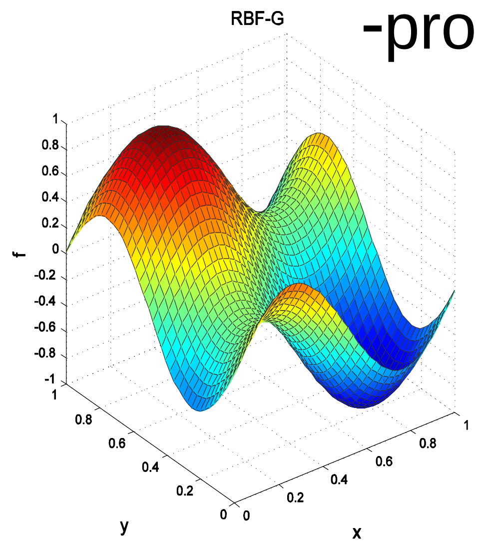
$$x_i \in [0,1]\ , \quad i = 1, 2\ ,$$
$$f(\mathbf{x}) = \frac{1}{2}\Big[\sin(2\pi x_1) - \sin(2\pi x_2)\Big]$$

-40 Random training points

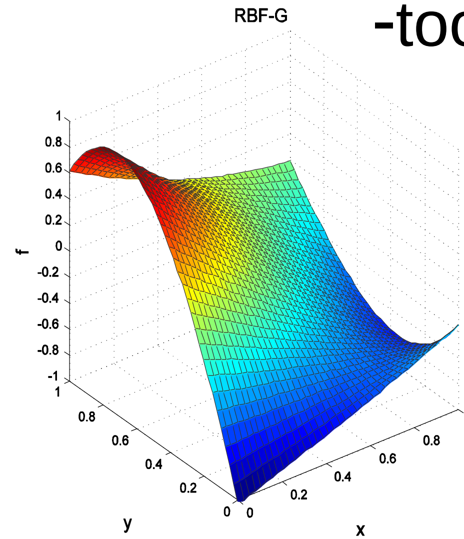
δ -scaling parameter



-too low



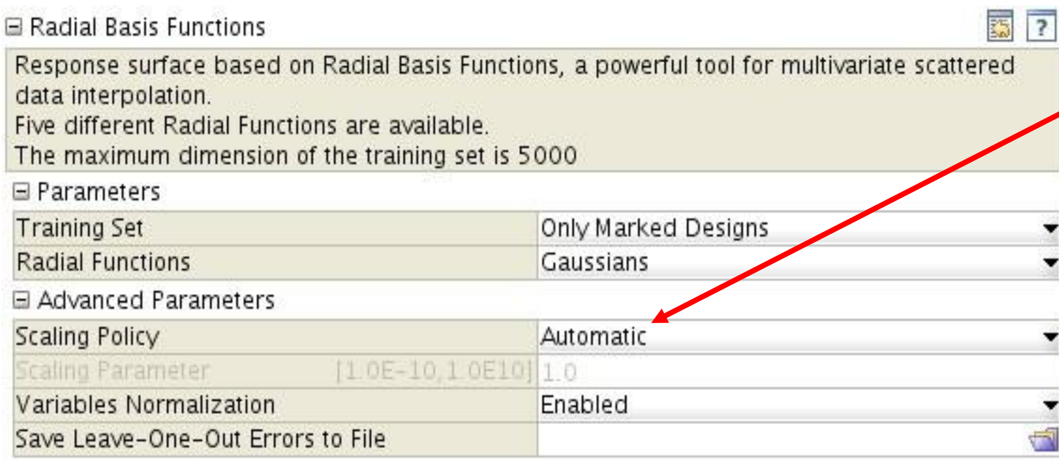
-proper



-too high

Radial Basis Functions

RBF parameters panel



Automatic Scaling Policy: the proper value of the scaling parameter is automatically found by minimizing the mean leave-one-out error.

The leave-one-out methodology is a suitable system for checking the goodness of an interpolant model.

Usually it has a huge computational demand. But this is not the case for RBF: in fact there is a convenient method for computing it.

Radial Basis Functions

RBF log

```
             Scaling Parameter = 1.083100428153687
10:34:39:781 Mean Leave-One-Out Error = 1.1277096605766364E-4
10:34:39:813 --------------------
10:34:40:173 Scaling Parameter = 1.111809300020706
10:34:40:205 Mean Leave-One-Out Error = 1.10564520465007E-4
10:34:40:237 --------------------
10:34:40:592 Scaling Parameter = 1.1059329239279752
10:34:40:624 Mean Leave-One-Out Error = 1.0991631752454632E-4
10:34:40:656 --------------------
10:34:40:691 TRAINING COMPLETED
10:34:40:720 Automatic Scaling Parameter = 1.1059329239279752
10:34:40:752 Mean Leave-One-Out Error: 1.0991631752454632E-4
10:34:40:785 GOOD FITTING
10:34:40:817 Condition Number: 1.306037899231765 12E17
10:34:40:881 EXIT 0
```

Automatic Scaling Policy: the optimum value of the scaling parameter is automatically found.

The mean leave-one-out error is useful for checking the goodness of the RBF model.

Neural Networks

- Neural Networks (NN) are inspired by the functioning of biological nervous system. NN are used to solve a wide variety of complex problems.

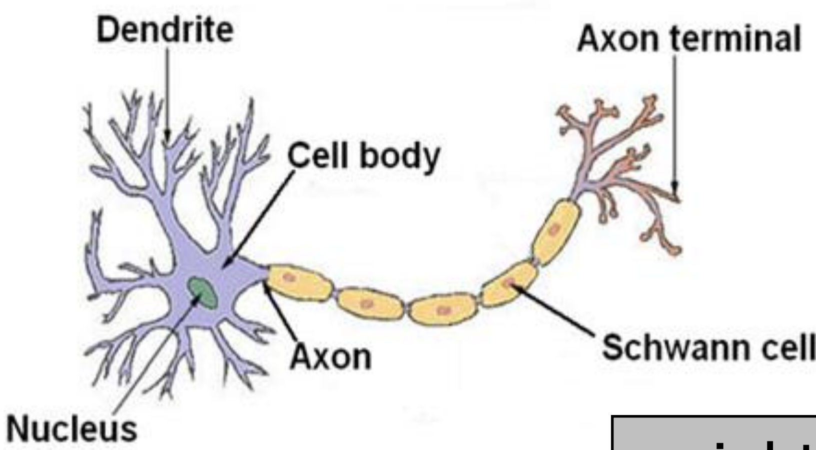
- The behavior of a NN is defined by the way its neurons are connected.

- A NN can learn, and therefore can be trained to perform a particular job.

- NN have no limiting assumptions of normality or linearity.

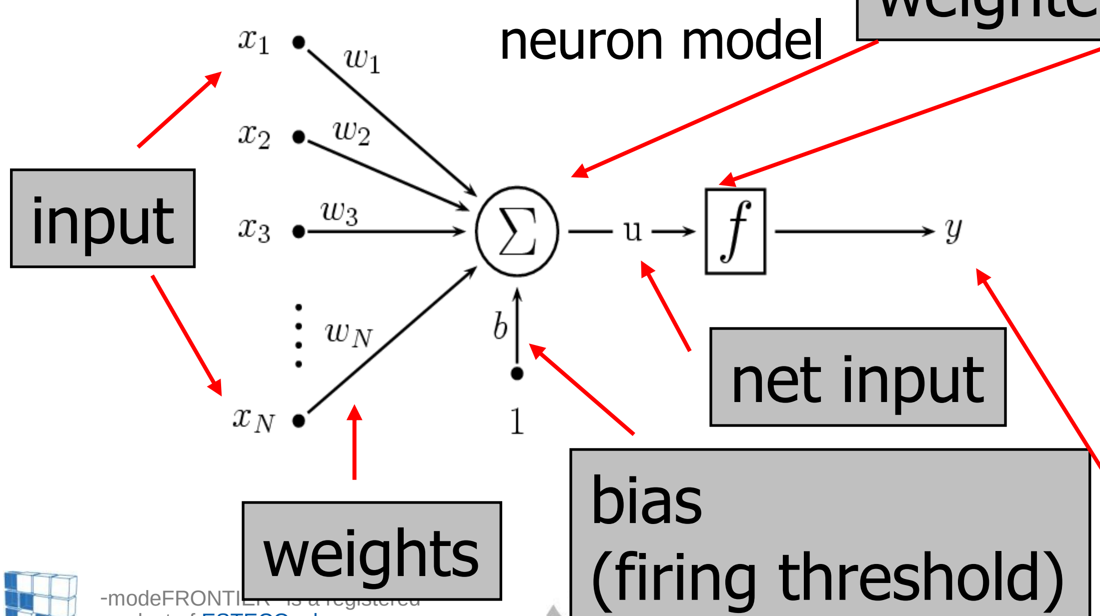
Neural Networks

real neuron



NN are formed by simple individual computing elements – the neurons – operating in parallel

neuron model

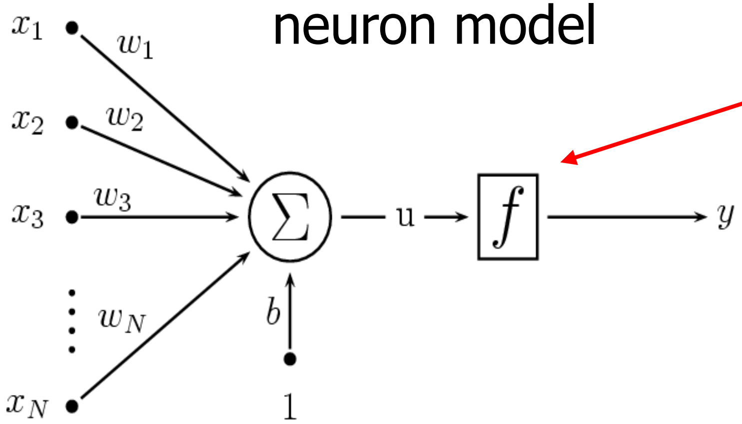


$$u = \sum_{i=1}^{N} w_i\, x_i + b$$

$$y = f(u) = f\left(\sum_{i=1}^{N} w_i\, x_i + b\right)$$

Neural Networks

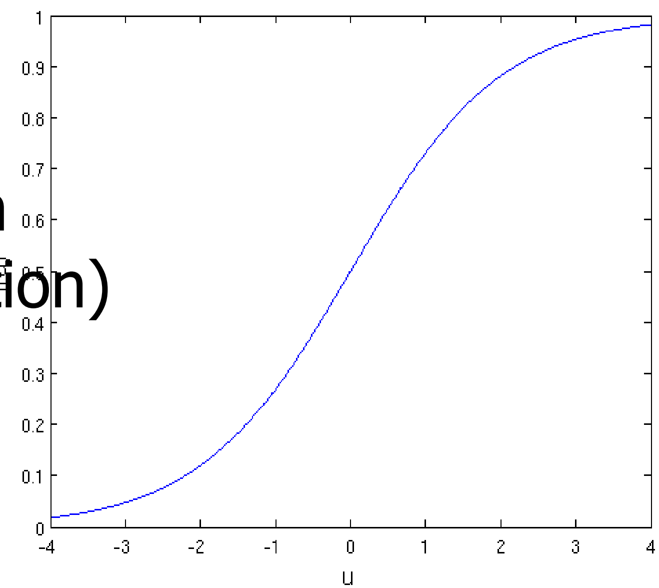
neuron model



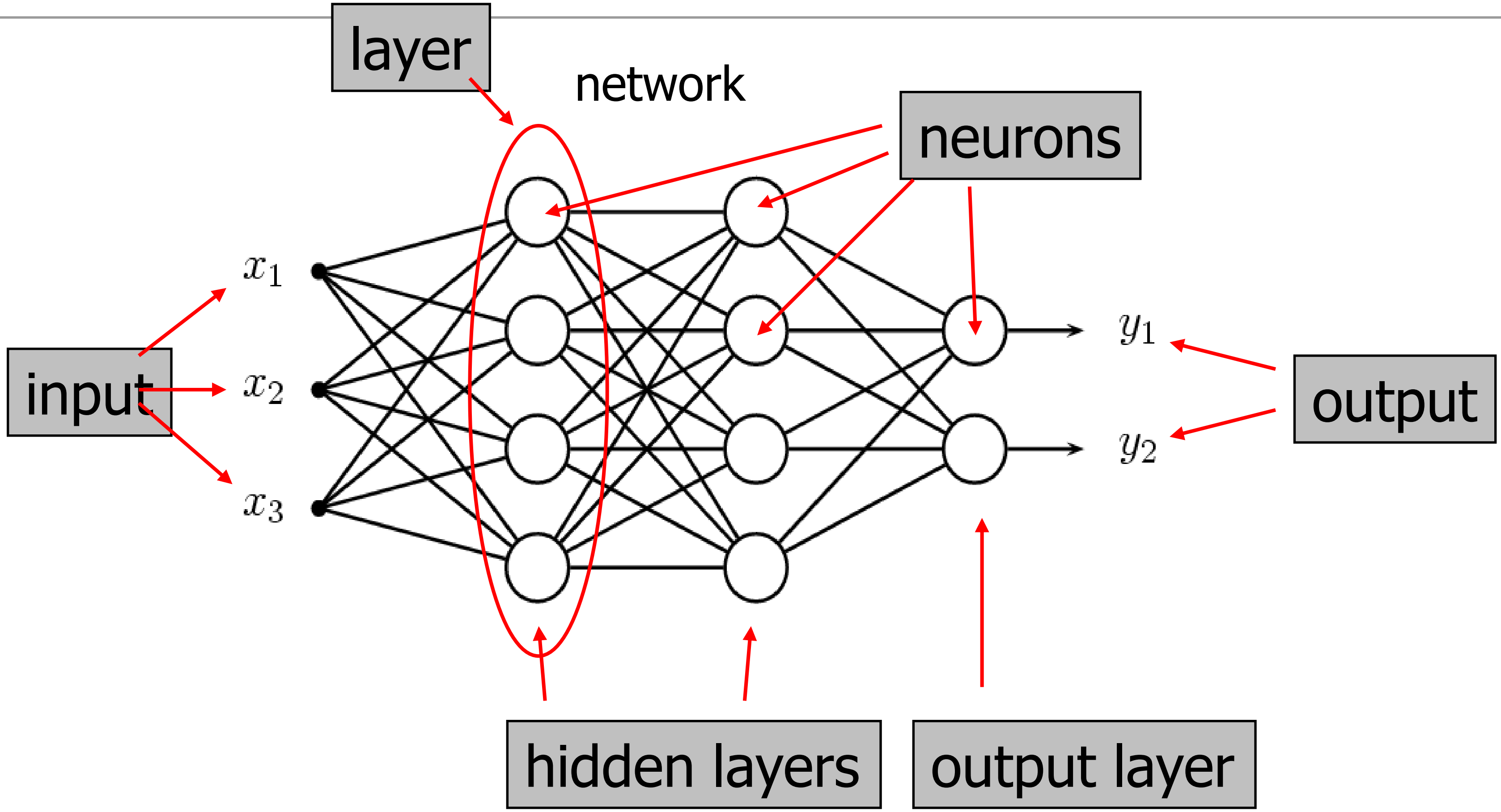
transfer function
(or activation function)

sigmoid function
(or logistic function)

$$f(u) = \frac{1}{1 + e^{-u}}$$



Neural Networks



NN can create a mathematical model that correctly maps the input to the output.

Neural Networks

- NN learn by example.
- Given a training data set (i.e. a set of $\{\boldsymbol{x}^i, \boldsymbol{y}^i\}$ values), NN learn to model the relationship between input ($\boldsymbol{x}$) and output ($\boldsymbol{y}$) variables by means of a training algorithm.

- The best known learning algorithm is back propagation.
- Back propagation adjusts the network's weights and biases in order to minimize the errors in its predictions on the training data.

- Once the NN has been properly trained, it can subsequently be used to make predictions where the output is not known.

Neural Networks

- Classical feed-forward network with one hidden layer: this is a powerful configuration for RSM purposes, as outlined by Irie and Miyake.
- Efficient **Levenberg-Marquardt** back propagation training algorithm. Faster training.
- Proper initialization method of Nguyen and Widrow for network's parameters. Faster training, again.
- Automatic network sizing is available.
- User friendly: many parameters are automatically set by the algorithm.
- Export in Java, C, FORTRAN, EXCEL formats is now available.

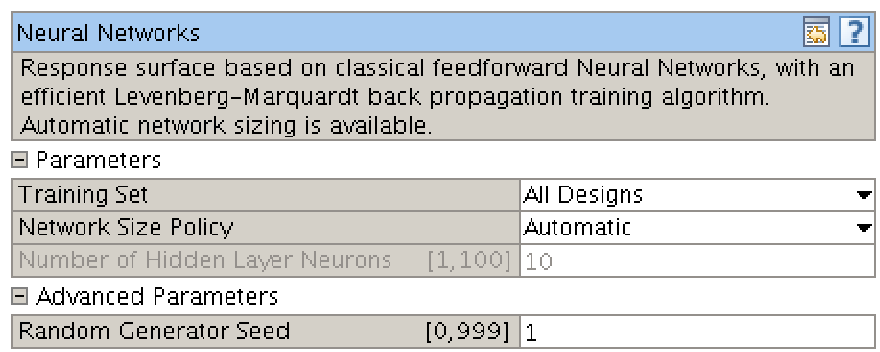
Neural Networks

User friendliness

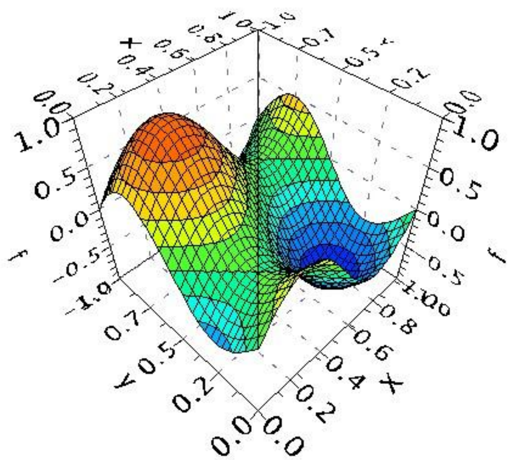
How many nodes (neurons) should I use?

Cumbersome setting of error parameters: trial and error

Automatic network sizing



Neural Networks

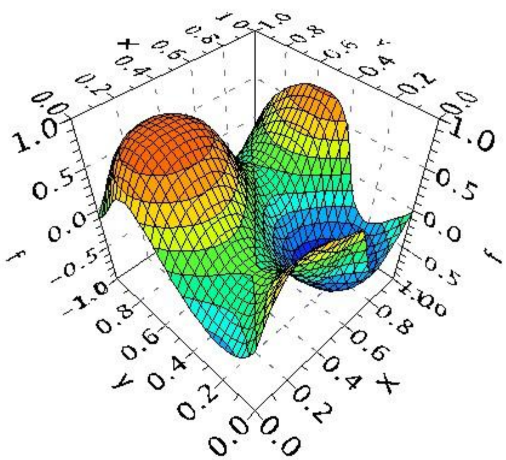


Performance comparison: new vs. old

test case:

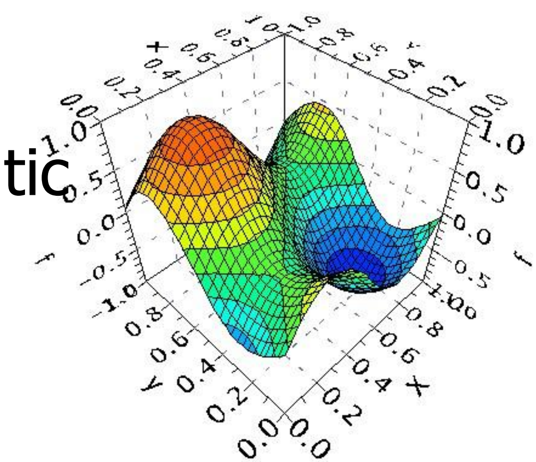
$$f = \frac{1}{2}\left(\sin(2\pi x) - \sin(2\pi y)\right), \quad x, y \in [0,1]$$

training set: 50 DOE random



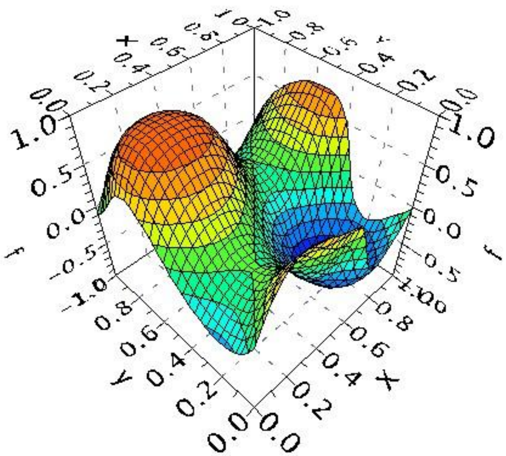
old
6 nodes
0.05 max err.
0.01 max av. err.
(fine tuning after several trials)

new
automatic



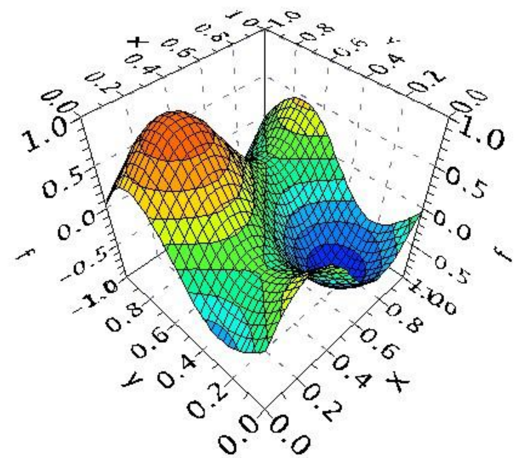
Neural Networks

Performance comparison: new vs. old



old

new

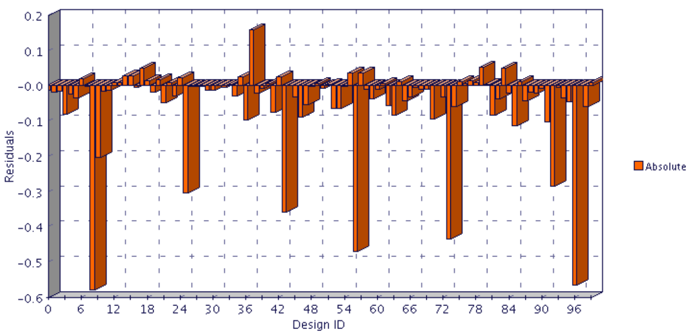


training time:

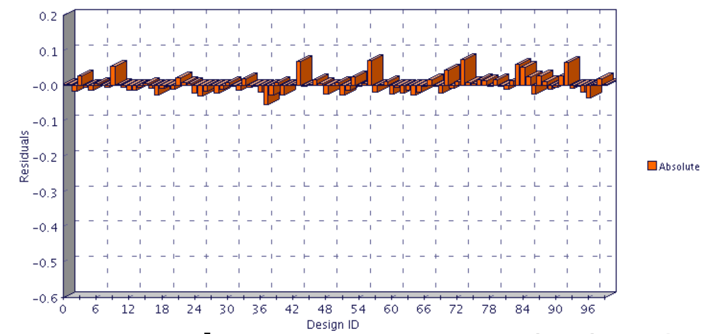
2 m 34 s (154 s) 1.2 s

(Intel P4 2.5GHz)

validation set: 100 DOE random



accuracy



max abs. err. = 0.58
mean abs. err. = 0.058

max abs. err. = 0.076
mean abs. err. = 0.016

Genetic Programming

What is it?

- A meta-heuristic method

Genetic Programming is part of a group of methods, called meta-heuristic.

These methods have the ambition to solve optimization problems for which we do not know a polynomial algorithm.

What is it?

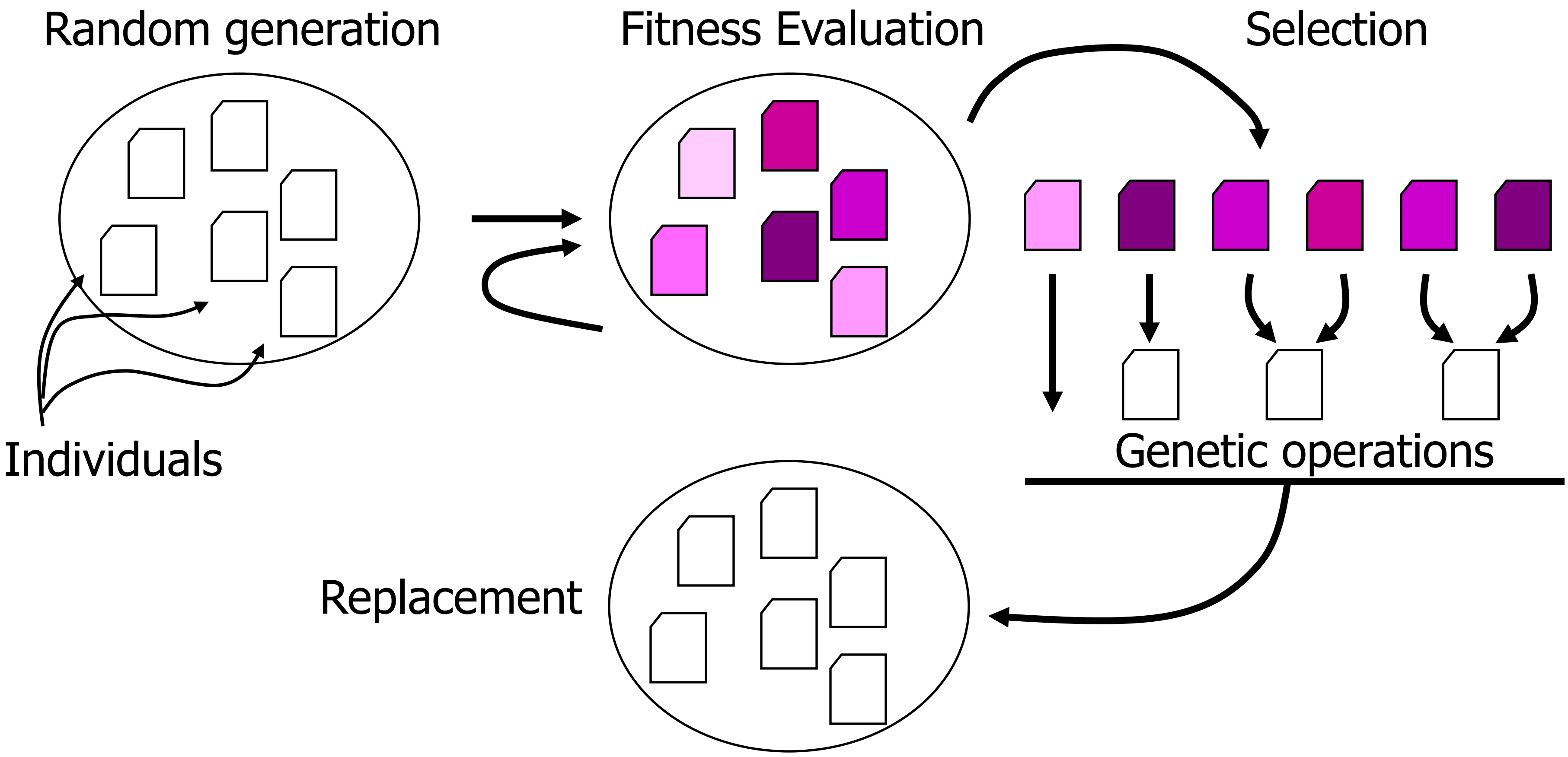
Some common characteristics:

- Stochastic: they make it possible to face the combinative explosion of the space of solutions.

- Inspired by analogies with:
  - physics (simulated annealing),
  - biology (evolutionary algorithms),
  - ethology (ant colonies, particle swarm).

Genetic Programming belongs to the group of the evolutionary algorithms.

Evolutionary algorithms

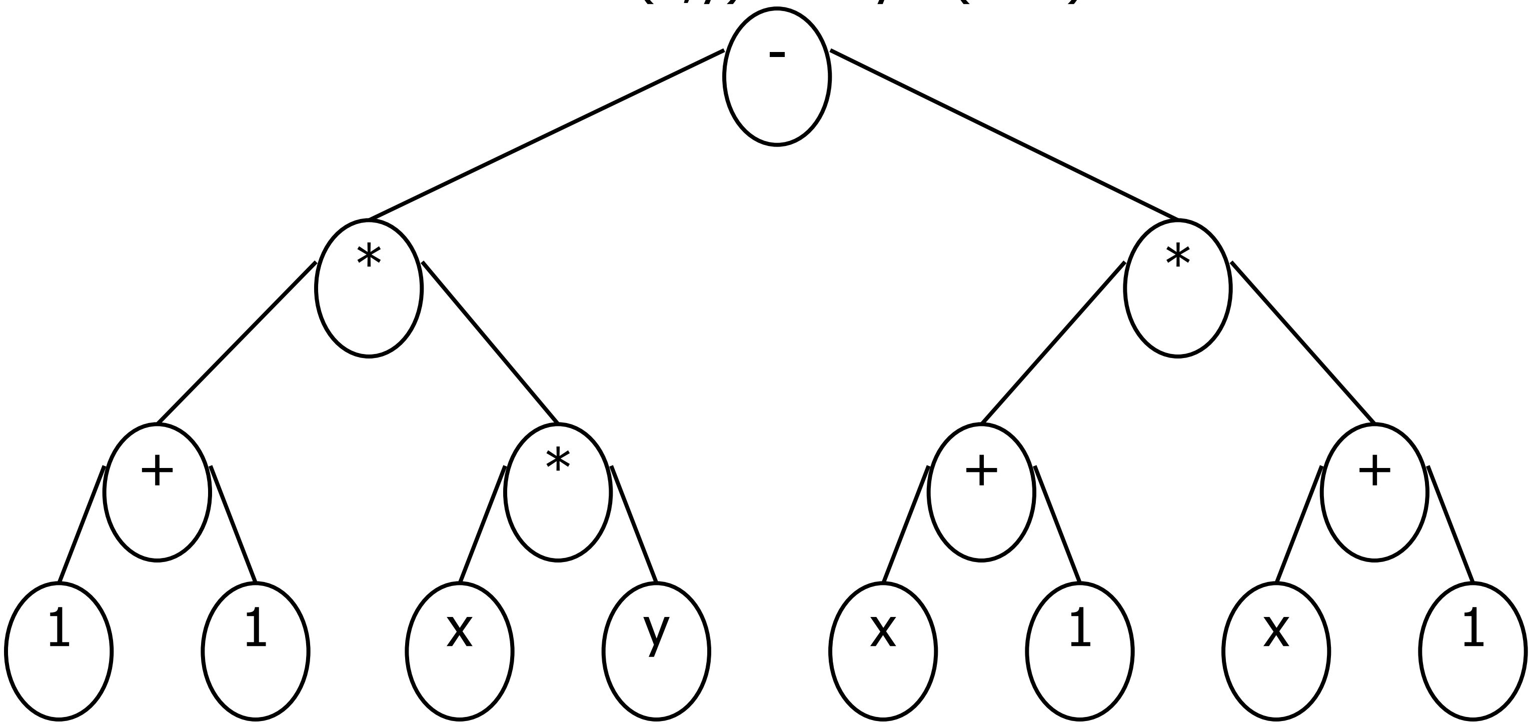


Genetic Programming in a nutshell

- Individual representation in GP

  Mathematical expressions are represented by trees.

$$F(x,y) = 2xy - (x+1)^2$$

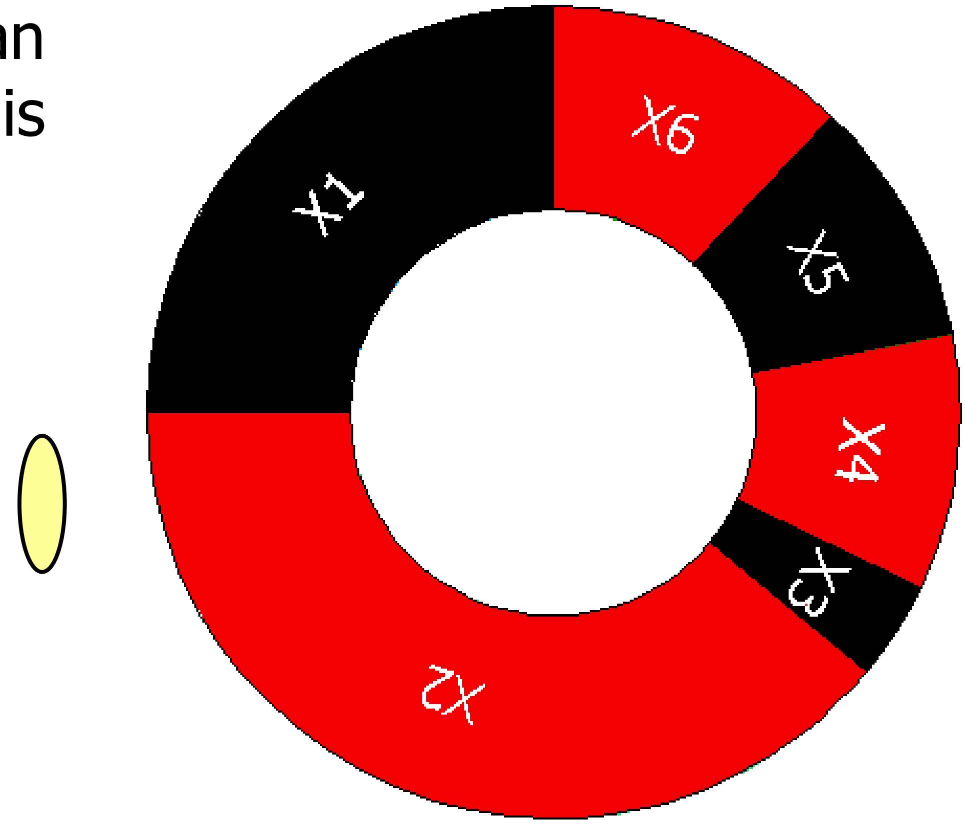


Genetic Programming in a nutshell

- Fitness proportionate or roulette wheel selection

The probability that an individual X_i will be select is simply his normalized fitness.

Individuals	Normalized fitness
X1	0.25
X2	0.39
X3	0.04
X4	0.10
X5	0.10
X6	0.12



Nf(x1, … , x6) = (0.25 0.39 0.04 0.1 0.1 0.12)

Genetic Programming in a nutshell

- Selection – **elitism**

Main idea: copy the best or the few best individuals in the next generation.

Why? During the evolution we have the risk to lose the best individual.

Elitism can increase performance of genetic programming because it prevents losing the best known solution.

Genetic Programming in a nutshell

- Genetic operators: roles and typologies

Genetic operators can be classified in 2 main categories:

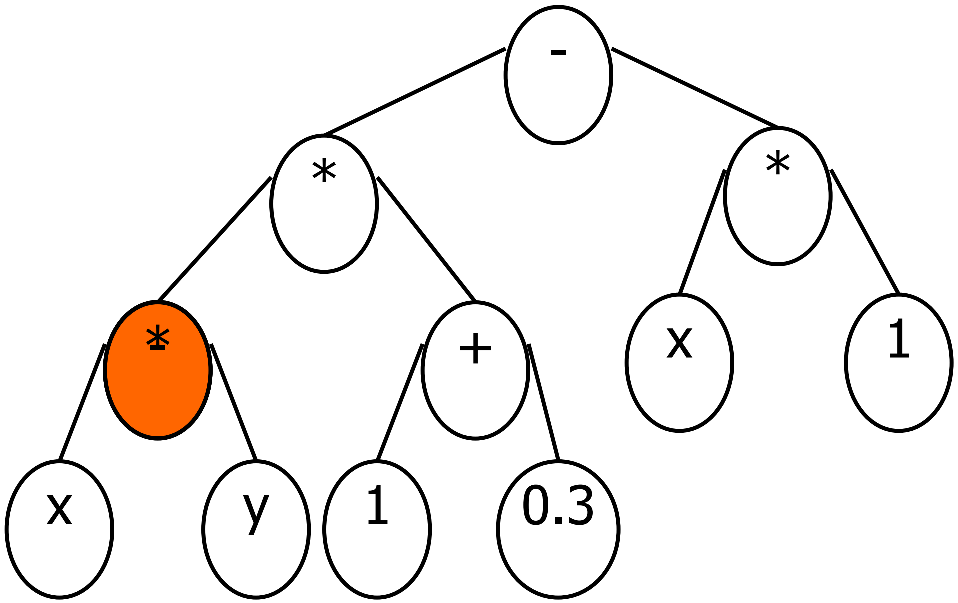
- Recombination operators, which use several parents in order to create one or more offspring.

- Mutation and duplication operators which transform only one individual.

Genetic Programming in a nutshell

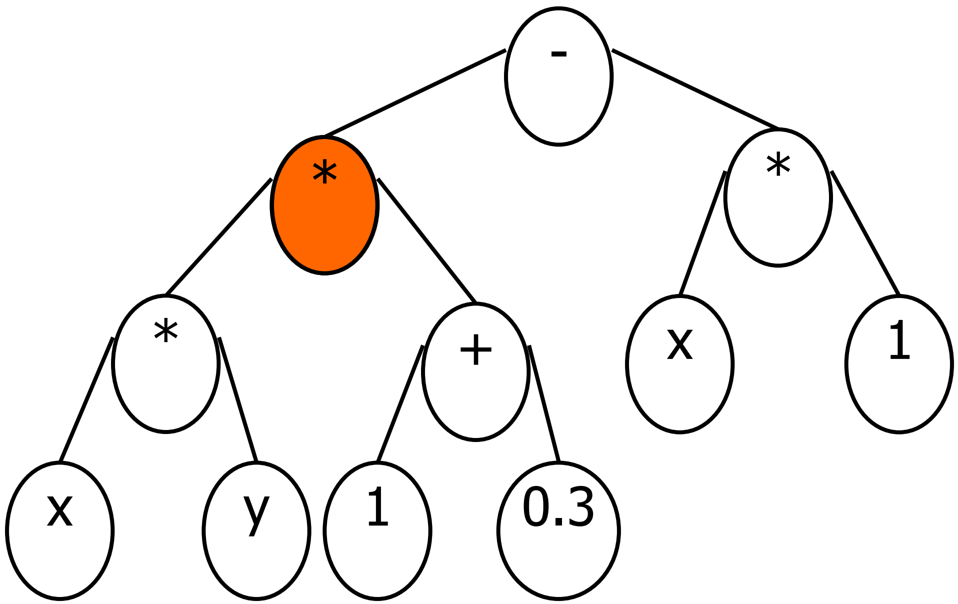
Mutation operation introduces random changes in the tree structure of an individual.

One point mutation



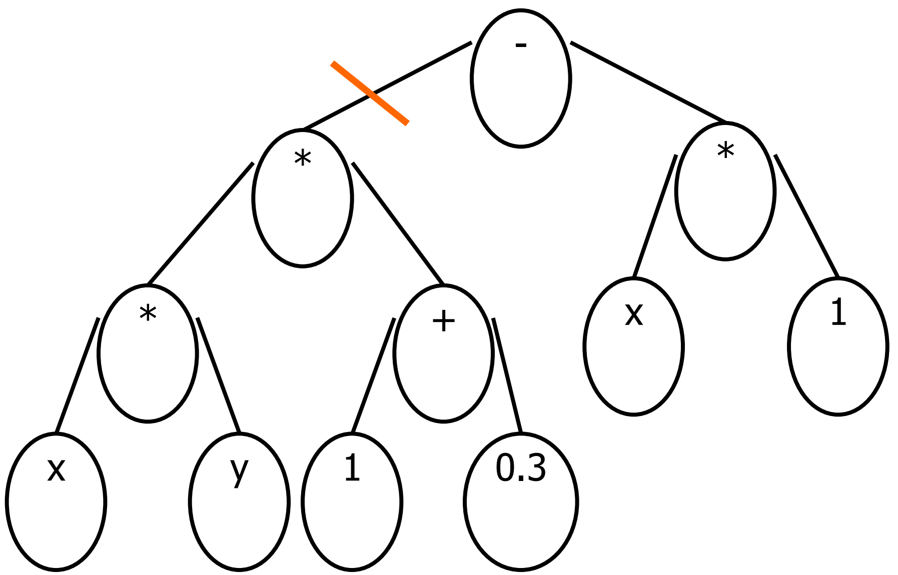
Genetic Programming in a nutshell

Promote mutation



Genetic Programming in a nutshell

Crossover operation produce new offspring that consist of parts taken from each parent.



Applications

Data classification

Handley uses GP to predict whether each part of a protein would have a particular geometric shape or not, using the protein's chemical composition. [Han1993]

Robot control

Spencer used GP to generate programs that enable a "six-legged insect" to walk in a simulated environment. He reports that GP evolved programs were at least as good and often better than those created by humans. [Spe1993]

Applications

Symbolic regression

La Roche presents a genetic programming approach to detect de-authentication attacks on wireless networks based on the 802.11 protocol. [Roc2006]

Time series prediction

Vazquez uses GP to forecast meteorological data. [Vaz2001]

Applications

Signal processing

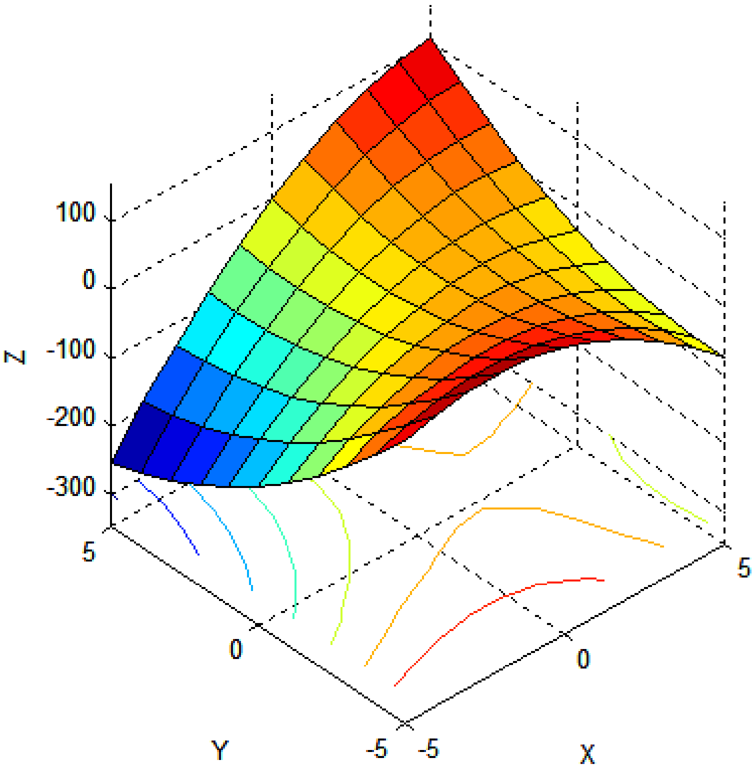
GP has been used to construct a program which detects engine misfires in cars. GP approach was found to be superior to one based on neural networks.

The program produced was more easily integrated into the existing (software based) engine management system than would have been the case with other approaches, such as the neural network.

This may be the case in many embedded systems. [HBM1994]

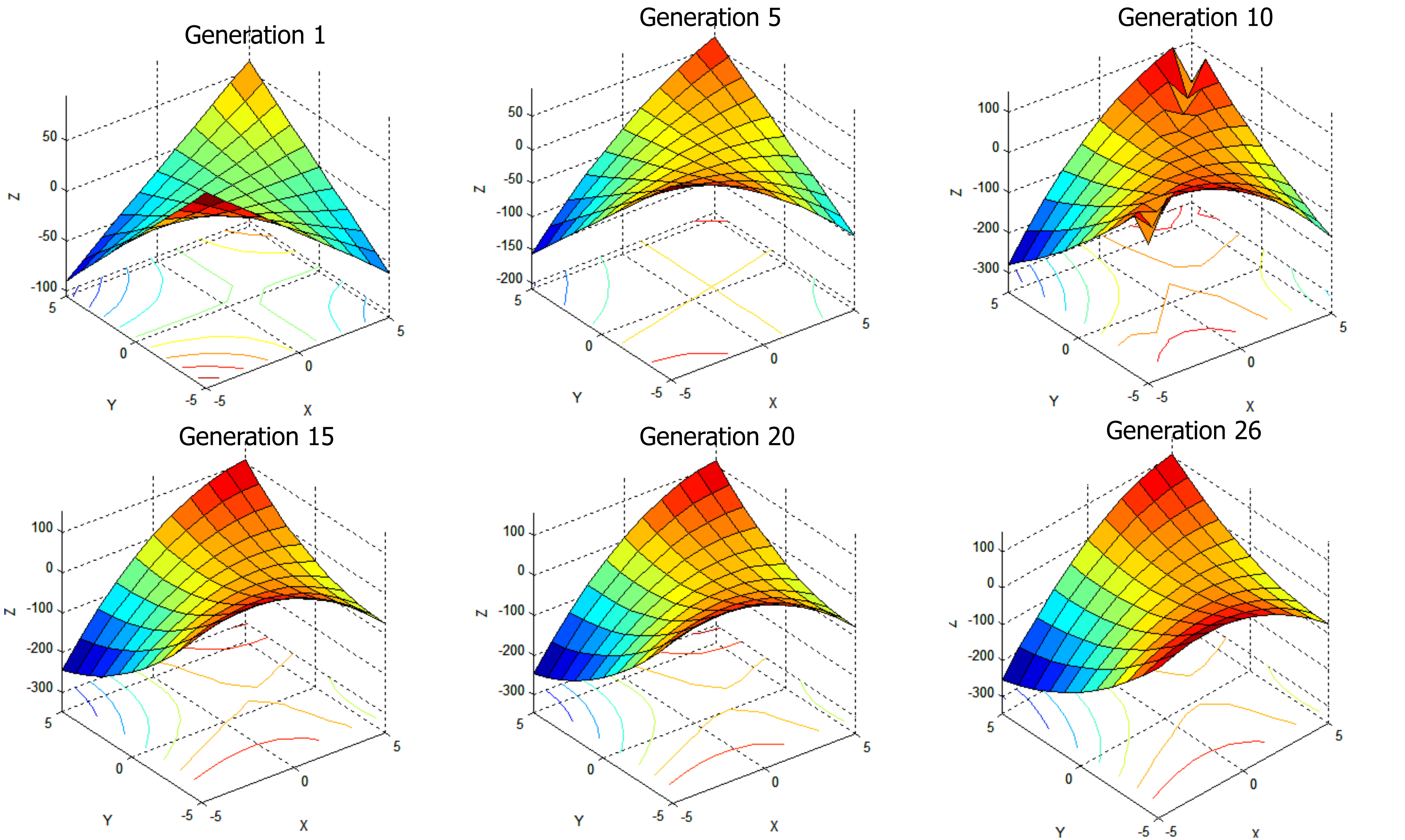
Exercise

Symbolic regression problem: the goal is to find a mathematical expression, in symbolic form, that fits the surface below.



X	Y	Z
-5	-5	67
-5	-4	80
-5	-3	87
-5	-2	88
-5	-1	83
-5	0	72
-5	1	55
-5	2	32
-5	3	3
...	...	...

Evolution towards solution



Solutions

Generation	Best individual
1	(((y-1)*(x+x))-((x/1)-(x*y)))
5	(((((x-y)-(1-x))*((x/x)-(1-y)))+(((x/x)-(1-y))+((y-1)*(x+x))))-((((1/1)+(1-1))-((y+y)-(x+y)))-((((1-1)*((y+y)+(x/x)))+(y-x))-((1+1)-(1+1)))))
10	(((y-1)*x)-(((((x-y)*(1-y))+(1-1))-((((((1-1)+(x*y))+(((1/(1+1))*(y*x))-((x+1)+(y*y))))+(x*x))-((((1/(x-y))+(1-1))-((y+y)-(y*y)))-(((x*y)+(y-x))-((1+1)-((y*1)-(1/1))))))-(x+y)))-(((x*((y+y)+(x/x)))+(y-x))-((1+1)-(y+y)))))
15	(((((1-1)+x)+((((1/1)*(y*x))-(x+(y*y)))-((x+1)+((((x-y)+(1-1))-((y+y)-(((x/1)-((((1-1)+(y+y))+(((1/1)*(y*x))-((x+1)+(y*y))))+(x*x)))-((y+y)-(x+y)))))-(((x*((y+y)+(x/x)))+(y-x))-((1+y)-(1*y)))))))+(x*x))-((((1/1)+(1+1))-((y+y)-(x+y)))-(((y-1)*(x-y))-((1+1)-((y*1)-(1/1))))))
20	(((((1-1)+x)+(((1*(y*x))-(x+(y*y)))-((x+1)+((((1/1)+1)-((y+y)-(((x/1)-((((1-1)+(y+y))+(((1/1)*(y*x))-(((1+1)-(y+y))+(y*y))))+(x*x)))-((y+y)-(x+y)))))-(((x*(y+y))+(y-x))-((1+y)-(1*y)))))))+(x*x))-((((1/1)+(1+1))-((y+y)-(x+y)))-(((y-1)*(x-y))-((1+(1/1))-((y*1)-(1/1))))))
26	((((((y+y)-(x+y))+x)+((((1/1)*(y*x))-(x+(y*y)))-((1/1)+((((x-y)+(x+y))-((y+y)-(((x/1)-((((1-1)+(y+y))+(((1/1)*(y*x))-((x+1)+(y*y))))+(x*x)))-((y+y)-(x+y)))))-(((x*((y+y)+(x/x)))+(y-x))-((1+y)-(1*y)))))))+(x*x))-((((1/1)+(1+1))-((y+y)-((1+1)-(((1/(1/1))+1)-(1+1)))))-(((y-1)*(x-y))-((1+1)-((y*1)-((1/1)+(1+1)))))))

Evolutionary Design

⊟ Evolutionary Design

Response surface based on Evolutionary Design, an effective implementation of Genetic Programming methodology for symbolic regression.
Results of different jobs are all presented in the log, and the best result is automatically chosen.
The maximum dimension of the training set is 5000

⊟ Parameters

Training Set		All Designs
Jobs Number	[2,20]	5
Maximum Allowable Time (s)	[1,10000]	600
Crossover Depth	[5,15]	10

⊟ Advanced Parameters

Generations Number	[500,2000]	1000
Population Size	[200,2000]	500
Random Generator Seed	[0,999]	1
Functions Set		Edit Function Sets

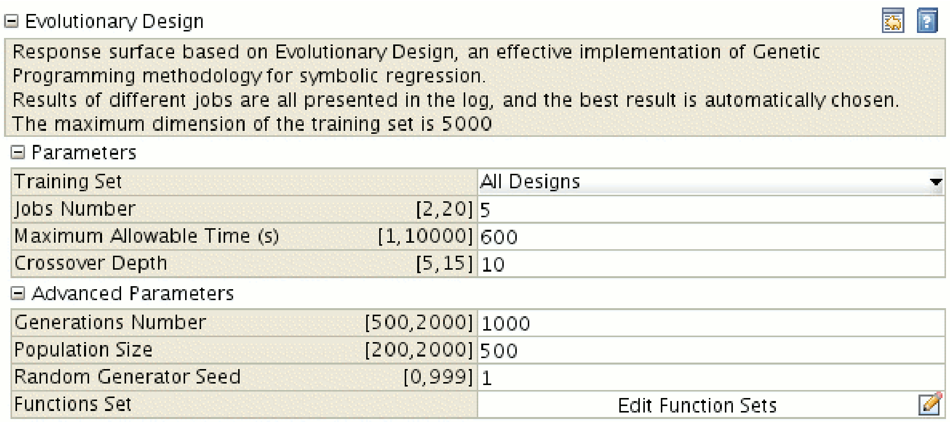
Algorithm settings:

number of different jobs run by ED

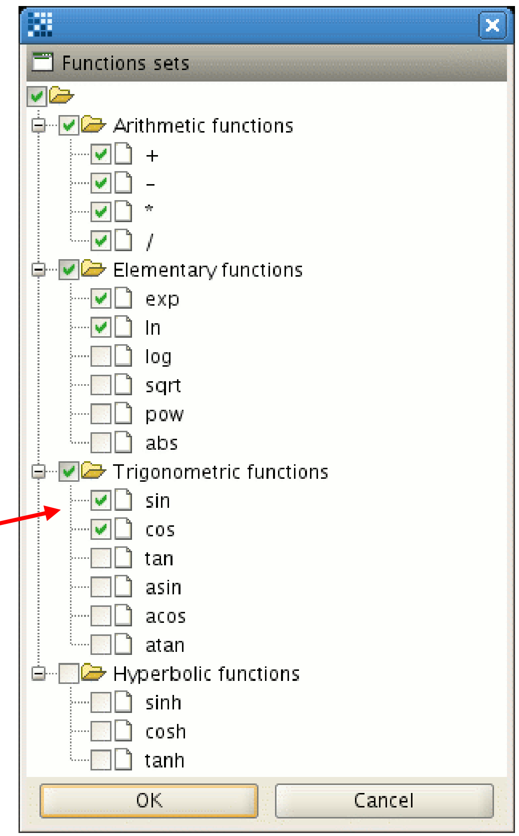
maximum allowable time for ED run

- GP algorithm settings:
  - Crossover depth
  - Generations number
  - Population size

Evolutionary Design



Function sets:



It is possible to choose which primitive functions should be included in the model functional form

References

Books:

1. Koza J. R.: **"Genetic Programming: On the Programming of Computers by Means of Natural Selection"**. MIT Press (1992)
2. Banzhaf, W., Nordin, P., Keller, R.E., Francone, F.D.: **"Genetic Programming – An Introduction; On the Automatic Evolution of Computer Programs and its Applications"**. (1998), Morgan Kaufmann, dpunkt.verlag

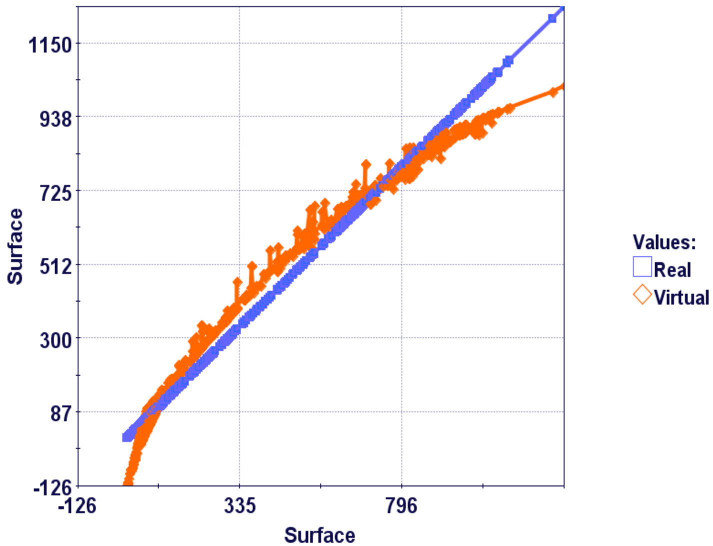
Web:

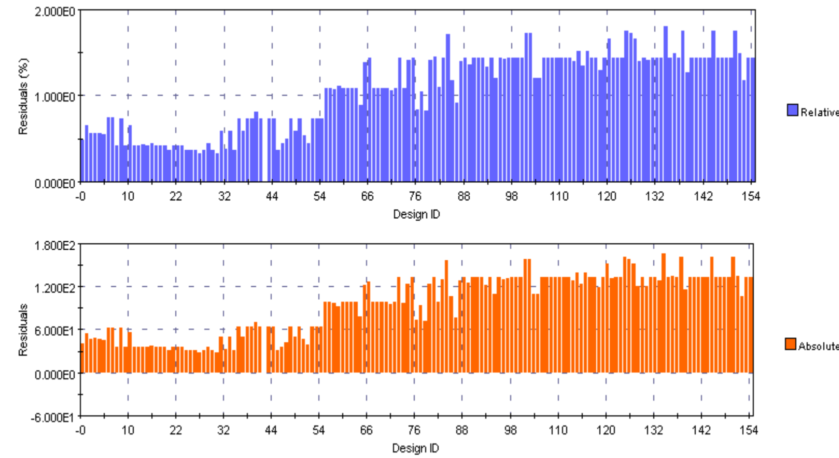
1. www.genetic-programming.org

Conferences:

1. EuroGP
2. Congress on Evolutionary Computation

RSM charts (residual and coefficients)

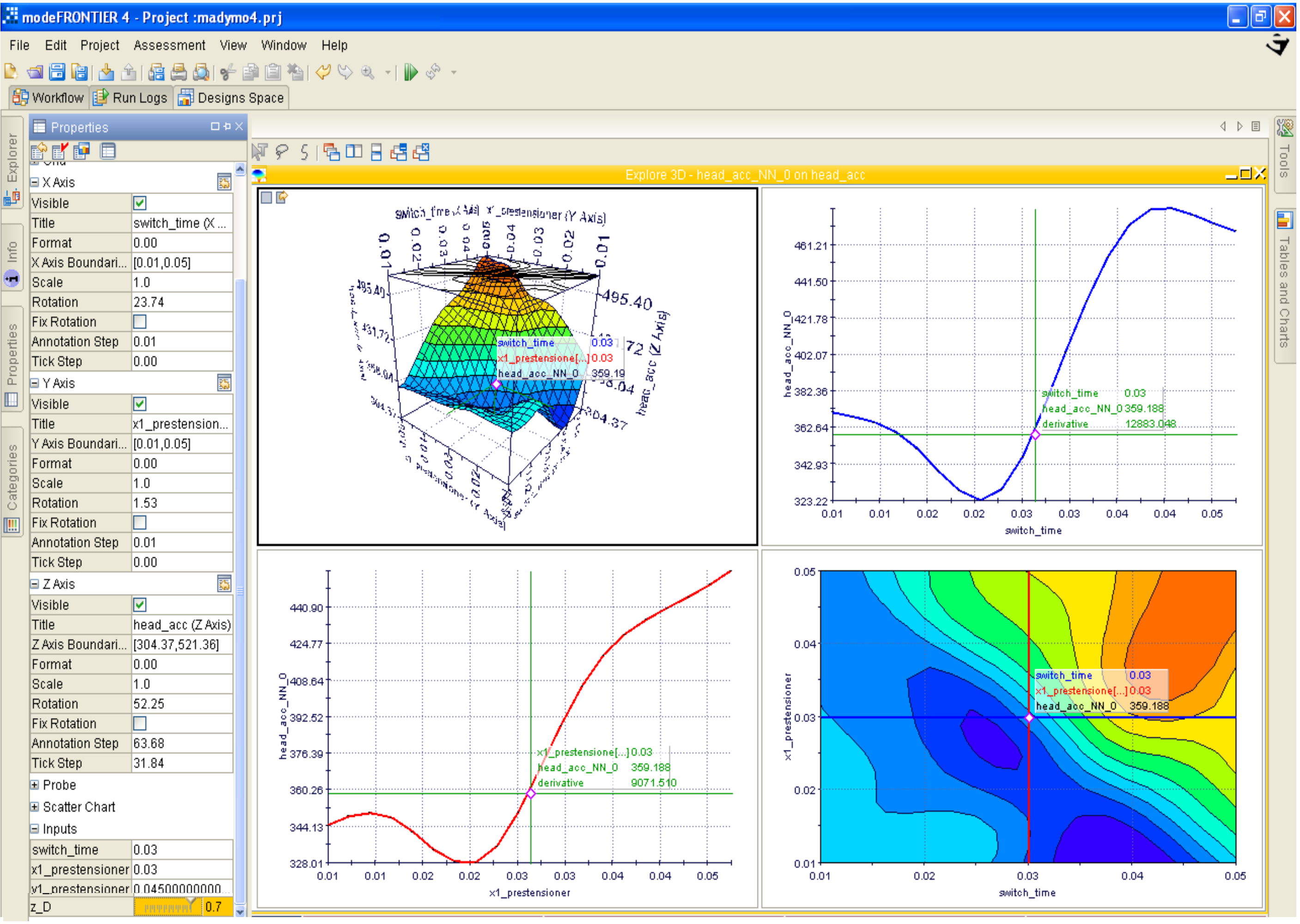




Max Abs. Error	1.665E2	Max Rel. Error	1.806E0
Average Abs. Error	9.579E1	Average Rel. Error	1.064E0
Regression	8.937E-1	Errors Mean	9.578E1

• Distance and Residual charts, useful to evaluate relative (or absolute) extrapolation error (to have evaluation of extrapolation error, leave few designs outside training set, and look at error on them)

RSM charts (Exploration 3D)

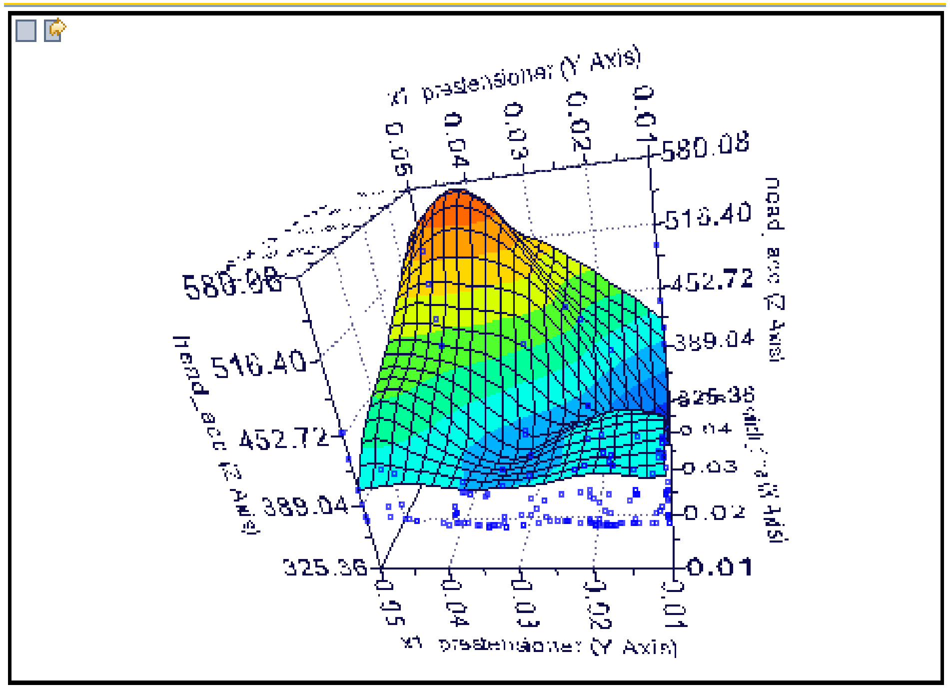


- 3D RSM is plot in function of 2 variables
- 1D curve section for the two variables is plot
- from Properties chart, slider can modify value of any variables: all charts are updated

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

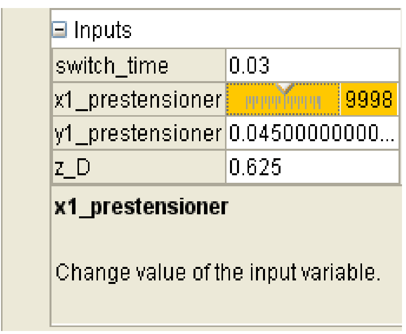
modeFRONTIER
the multi-objective optimization and design environment

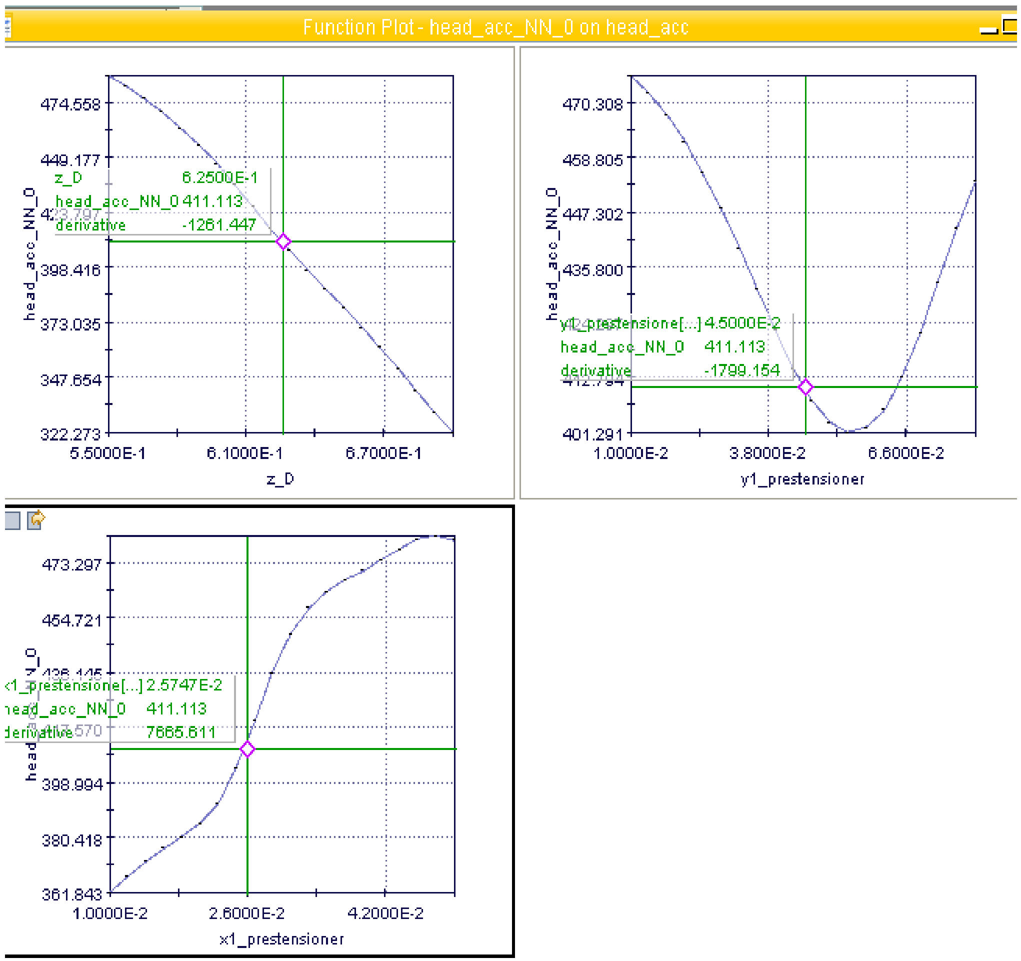
RSM charts (Exploration 3D)



- from Properties>Scatter, the visible option turned on shows existing database points overimposed on the charts

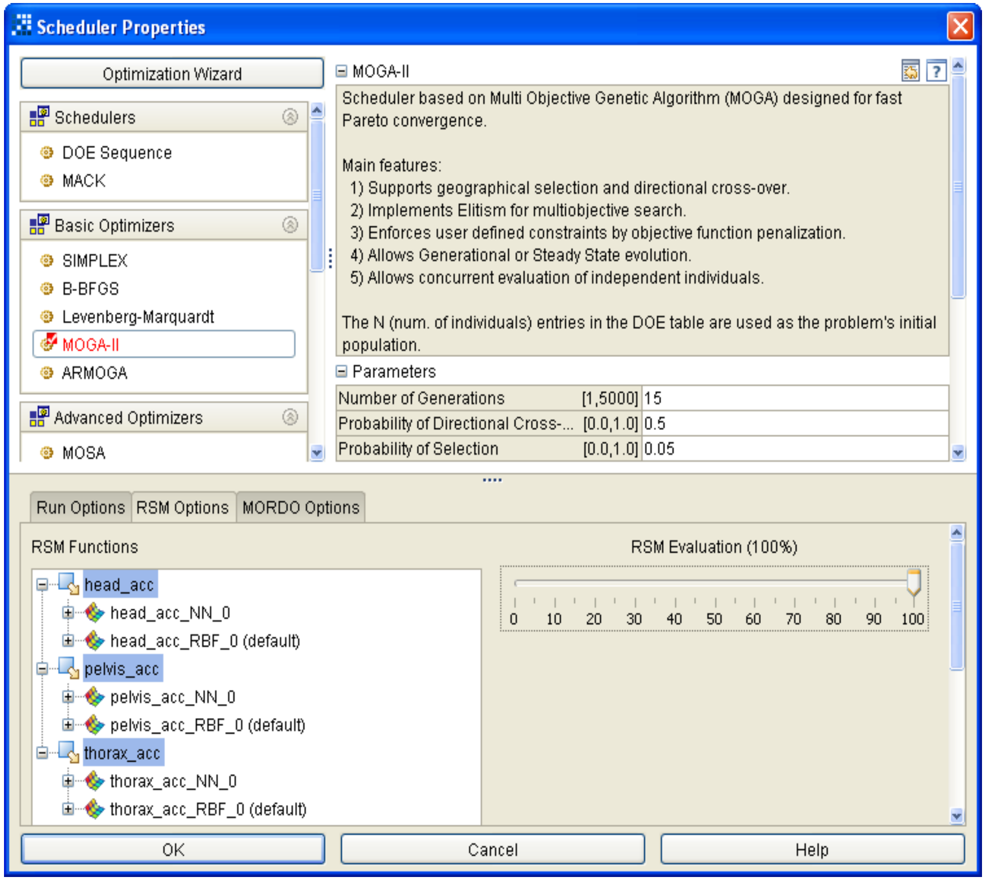
RSM charts (all input variables plot)





- Function plots can be defined for all the input variables: moving the point on the curve or the sliders, the RSM is updated in all plots

Virtual optimization



-Once RSMs are trained and tested for each output, the optimization can be run setting a % of RSM evaluation (in Scheduler Properties)