On the use of supervised learning techniques to speed up the design of aeronautics components.

Gianluca Bontempi[1], Olivier Caelen[1], Cecile Goffaux[2]

[1]Machine Learning Group, Department of Computer Science
Université Libre de Bruxelles, Belgium

[2]CENAERO, Centre of Excellence in Aeronautical Research,
Gosselies, Belgium

July 4, 2009

ULB Machine Learning Group (MLG)

- 10 researchers (2 prof, 4 PostDoc, 4 PhD students).
- Research topics: Classification, Computational statistics, Data mining, Regression, Time series prediction, Stochastic optimization.
- Applications: Bioinformatics, Biomedical, Industry, Sensor networks, Spatial data mining, Fraud detection.
- Computing facilities: cluster of 16 processors, Wireless Sensor Lab.
- Website: `www.ulb.ac.be/di/mlg`.

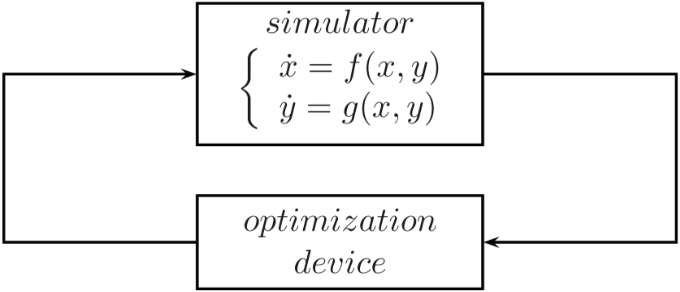
CENAERO

- Private Non-Profit Research Centre
- 3 universities (ULB, UCL, ULg)
- 1 research center (VKI)
- 50 industry members
- incorporated in 2002 in Gosselies
- 35 employees

Activities and Skills:

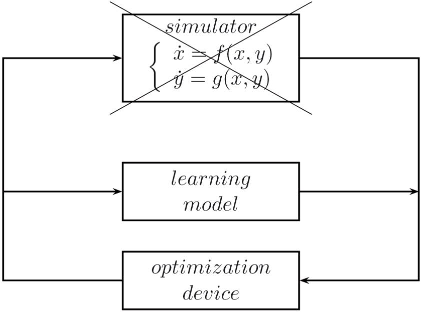
- development of simulation softwares for multidisciplinary problems in aeronautics
- R&D in supercomputing, advanced numerical methods, parallel computing
- advanced engineering studies for the industry
- High Performance Computing (HPC) center

Design space exploration with physical simulator



- Complex phenomenon modeled by a complex parametric simulator taking into account physics law and human expertise.
- Need of designing appropriate parameters in order to maximize some (complex) cost function.

Design space exploration with machine learning approximator



Two possible configurations

1. model replaces simulator
2. model allows a parmonious use of simulator (experimental design)

Motivations

- Computational and economic cost of the simulator.
- Slowness of the simulator.
- Understanding of the role of the design parameters.
- Dimensionality reduction.
- Sensitivity analysis.
- Taking advantage of historical observations.
- Generalization to other cost functions.
- Large range of applications: aeronautics [10, 11, 6], electronics [12], multimedia.

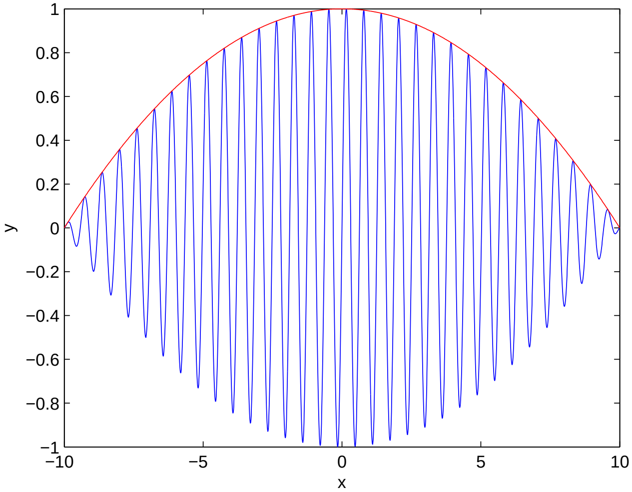
System design: formalization

- A vector $x \in \mathcal{X} \subset \mathbb{R}^n$ of design parameters (also design configuration), where $\mathcal{X}$ is called the *design space*.
- A vector $y \in \mathcal{Y} \subset \mathbb{R}^m$ of *design objectives* to be maximized which are used to assess the quality of a design.
- An evaluation or objective function $E : \mathcal{X} \to \mathcal{Y}$ which maps the design space into the objectif space.
- A search strategy π which explores the design space in order to find good or optimal configurations x according to the evaluation vector $E(x)$.
- A *strategy dependent* evaluation function $E^\pi : \mathcal{X} \to \mathcal{Y}$ where

$$E^\pi(x; K) = \max_{x^{(k)}, k=0,\dots,K} E(x^{(k)}),$$

where $x^{(0)} = x$ and $x^{(k)}$ is the set of states explored by the search strategy π starting in x and proceeding for K steps.

Strategy dependent evaluation function



Turbine example

In the case of tubine design

- the design parameters x define the blade geometry
- a design objectif y is the outlet flow angle,
- the role of evaluation function $E(x)$ could be played by Navier-Stokes simulator

Machine Learning and system design

Two existing approaches

1. estimation of the evalution function E from a set of N observed (simulated) pairs $\langle x_i, y_i \rangle$, $i = 1, \ldots, N$ independently of the search strategy
2. estimation of the strategy dependent evalution function E^π for a given search strategy π on the basis of a set of N observed search trajectories $\langle x_i^{(k)}, y_i^{(k)} \rangle$, $i = 1, \ldots, N$, $k = 0, \ldots, K$ (STAGE algorithm by Boyan and Moore [8]). The training dataset is then

$$D = \{x_i^{(0)}, \max_k(y_i^{(k)})\}, i = 1, \ldots, N$$

Note that E and E^π can be quite different! Also the amount of simulation required to setup the training set in the second case is much higher.

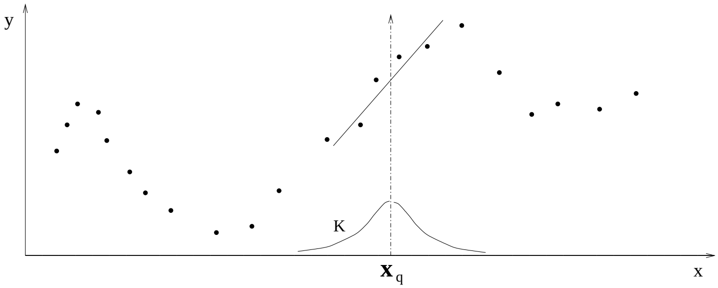
Machine Learning and system design

Once a number N of input/output pairs is obtained by running the simulator (and the optimization) the problem boils down to a supervised learning problem with some specificites:

- possibility of multiple outputs (multi-criteria problems)
- possible large dimensionality
- few samples
- on line or adaptive learning

In litterature local learning techniques appear to be often employed for performing such tasks.

Locally weighted regression



$$\sum_{i=1}^{N}\left\{\left(y_i-\mathbf{x}_i'\beta\right)^2 K\left(\frac{d(\mathbf{x}_i,\mathbf{x}_q)}{h}\right)\right\}$$

Local Learning for DSE

Local learning presents a set of specific features which make of it a promising tool in system design.

- **No a priori knowledge on the process underlying the data:** no assumption on the existence of a global function describing the data and no assumptions on the properties of the noise. This feature is particularly relevant in real datasets where problems of missing features, non stationarity and measurement errors make appealing a data-driven and assumption-free approach .
- **On-line learning capability:** LL can easily deal with on-line learning tasks where the number of training samples increases with time. In this case, the adaptiveness of the method is obtained by simply adding new points to the stored dataset. This is convenient in design problems where the number of available samples increases all along the exploration of the design space.

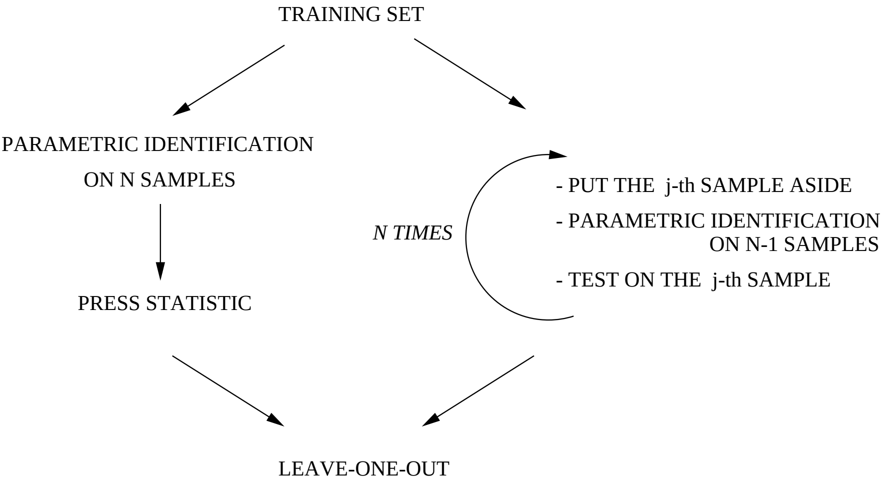
Local Learning for DSE

- **Effective feature selection:** The usefulness of a local modeling approach for reducing the cost of feature selection was first presented by Maron and Moore [9]. The idea consists in assessing a large number of feature subsets by performing cross-validation only on a reduced test set. On the basis of well-known statistical results, it is possible to show that families of good feature subsets can be rapidly found by quickly discarding the bad subsets and concentrating the computational effort on the better ones (*Hoeffding race* with reference to Hoeffding's formula which puts a bound on the accuracy of a sampled mean of N observations as an estimator of the expected value).
- **Gradient estimation:** Local search algorithms may take advantage of the local approximation of the objective function.

Lazy Learning [3, 4]

- Locally weighted regression which addresses the bias/variance dilemma.
- Automatic selection of the number of neighbors (aka bandwidth selection) by PRESS cross-validation
- Recursive least-squares to speed up the estimation of the alternative locally linear models
- Combination of local models.
- Confidence intervals associated to each prediction.
- Multi-input multi-output version [5].
- Available implementation in MATLAB and R.

Leave-one-out for linear models



Leave-one-out for linear models

$$e_j^{\mathrm{cv}} = y_j - \mathbf{x}_j' \hat{\beta}_{-j} = \frac{y_j - \mathbf{x}_j' \hat{\beta}}{1 - h_{jj}},$$

where $w_{ii} = \sqrt{K\left(d(\mathbf{x}_i, \mathbf{x}_q)/h\right)}$, $\mathbf{Z} = \mathbf{WX}$, $\mathbf{z}_j'$ is the j^{th} row of $\mathbf{Z}$ and therefore $\mathbf{z}_j = w_{jj}\mathbf{x}_j$, and where h_{jj} is the j^{th} diagonal element of the *Hat matrix* $\mathbf{H} = \mathbf{Z}(\mathbf{Z}'\mathbf{Z})^{-1}\mathbf{Z}'$.

Recursive leave-one-out

If the weight function $K(\cdot)$ is the indicator function

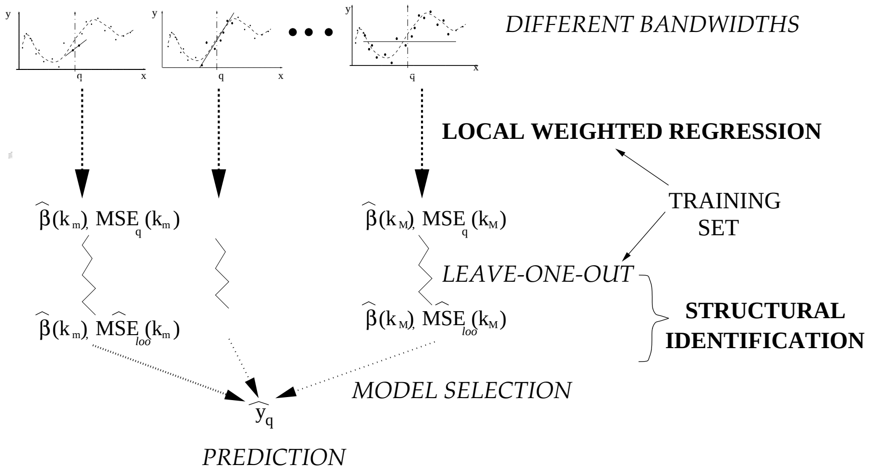
$$\begin{cases} \mathbf{P}(k+1) = \mathbf{P}(k) - \dfrac{\mathbf{P}(k)\mathbf{x}(k+1)\mathbf{x}'(k+1)\mathbf{P}(k)}{1+\mathbf{x}'(k+1)\mathbf{P}(k)\mathbf{x}(k+1)} \\ \gamma(k+1) = \mathbf{P}(k+1)\mathbf{x}(k+1) \\ e(k+1) = y(k+1) - \mathbf{x}'(k+1)\hat{\beta}(k) \\ \hat{\beta}(k+1) = \hat{\beta}(k) + \gamma(k+1)e(k+1) \end{cases}$$

where $\mathbf{P}(k) = (\mathbf{Z}'\mathbf{Z})^{-1}$ when $h = h(k)$, and where $\mathbf{x}(k+1)$ is the $(k+1)^{\text{th}}$ nearest neighbor of the query point.

$$e_i^{\text{cv}}(k+1) = \frac{y_i - \mathbf{x}_i'\hat{\beta}(k+1)}{1 - \mathbf{x}_i'\mathbf{P}(k+1)\mathbf{x}_i}$$

$$\hat{k} = \arg\min_k \text{MSE}(k) = \arg\min_k \frac{\sum_{i=1}^{k} \omega_i \left(\mathbf{e}_i^{\text{cv}}(k)\right)^2}{\sum_{i=1}^{k} \omega_i}$$

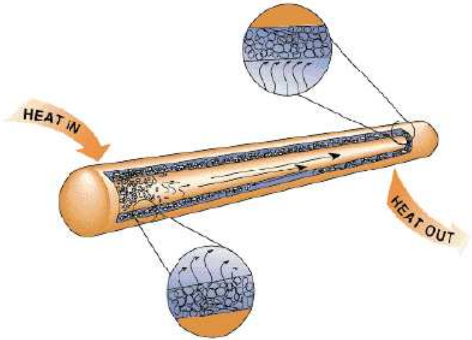
LL model selection



Heat pipes

- Since a couple of decades, the heat pipe technology has proven its efficiency in the thermal control of highly dissipative equipments such as the electronic component of satellites.
- A heat pipe is a closed thermodynamic system in which a liquid evaporates in the vicinity of a dissipative source and condenses in contact with a cold region. To insure its passive working in a microgravity environment, the heat pipe is composed of a vapour duct surrounded by a capillary structure.
- This structure allows for the fluid to return from the cold zone to the heat source.
- By using a large latent heat fluid, the heat pipe absorbs an important energy quantity during the phase change process, inducing a very high thermal transport capacity for weak variations in temperature.

A heat pipe

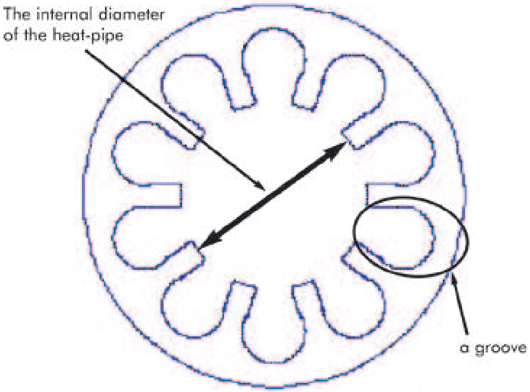


Heat enters the pipe and causes the liquid to boil. The resulting vapors expand into the pipe, carrying that heat. When they reach the cold end, the vapors condense back into a liquid, releasing the heat. Inside the pipe, a thin layer of material draws the liquid back along the pipe to the beginning point, where the cylcle is repeated.

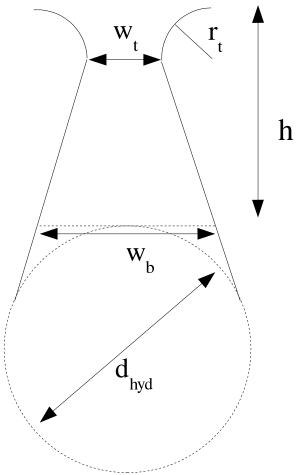
The Heat Pipe simulator

- Heat pipes are being used very often in particular applications when conventional cooling methods are not suitable. Once the need for heat pipe arises, the most appropriate heat pipe parameters need to be selected.
- In collaboration with Euro Heat Pipes (Nivelles, Belgium), CENAERO is continuously improving a thermohydraulic model Hea-P able to predict the heat transport capacity of grooved heat pipes for microgravity and gravity assisted applications.
- Hea-P includes a one-dimensional hydraulic model able to predict the maximum heat transport capacity of grooved heat pipes. The code relies on the equilibrium between the friction losses induced by the liquid and the vapor motions and the capillary pressure developed in the grooves. The convergence criterion imposed to calculate the maximum heat transport capacity assumes the maximum capillary pressure is reached at the end of the evaporator section.
- More details on Cenaero web page.

Section of a heat-pipe



Groove of a heat pipe



Simulator dataset 1

The first simulator dataset, henceafter $D1$, is composed of $N = 1260$ samples. It has $n = 3$ inputs and $m = 2$ outputs. The input design parameters are

- the internal diameter of the heat pipe.
- the diameter of the groove (d_{hyd}).
- the inclination angle of the heat pipe.

The output design criteria are:

- y_1: power (in Watt) released by the heat pipe (to be maximized).
- y_2: external diameter of the heat pipe (to be minimized).

Simulator dataset 2

The second simulator dataset, henceafter $D2$, is composed of $N = 820$ samples. It has $n = 6$ input variables and $m = 2$ output variables. The input design parameters are:

- The internal diameter of the heat pipe.
- The number of groove in the heat-pipe.
- The diameter of the groove (d_{hyd}).
- The width of the bottom of the grooves (w_b).
- The width of the top of the grooves (w_t).
- The depth of the grooves (h).

The two output design criteria are the same as in $D1$: y_1 (to be maximized) and y_2 (to be minimized).

Experimental setting

Experiments aims to assess the reliability obtained by replacing the simulator with a learned model.
For each dataset we carried out two experiments

1. prediction of the evaluation function

$$y = w_1 y_1 - w_2 y_2, \qquad w_1 + w_2 = 1$$

   in a training and test setting. We consider four learners: LIN (linear model), LL (Lazy Learning), NN (Feedforward Neural Network), SVM (SVM with radial kernel).

2. prediction of the highest value of the evaluation function y among the points in the test set. Comparison with random selection and local search.

Prediction of the evaluation function

The accuracy is measured in terms of Root Mean Square Error (averaged over different weightings of the evaluation function and different test sets)

- Dataset D1 (size training sets: $300, \ldots, 500$)

LIN	LL	NN	SVM
0.231	0.072	0.072	0.082

- Dataset D2 (size training sets: $50, \ldots, 200$)

LIN	LL	NN	SVM
0.43	0.39	1.33	0.44

Input and output data were normalized.

Prediction of the maximum

The error is the differenc between the real maximum in the test set and the value of the test instance what is predicted to be maximal. The accuracy is measured in terms of Root Mean Square Error (averaged over different weightings of the evaluation function and different test sets)

- Dataset D1

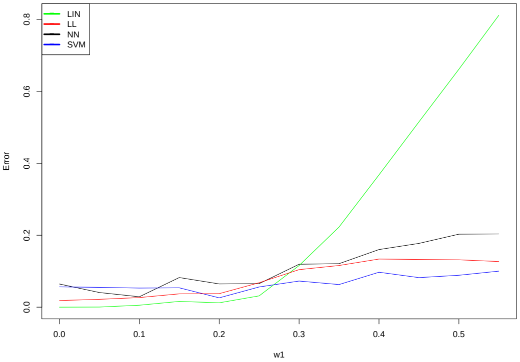
LIN	LL	NN	SVM	RAND	LOCAL
0.805	0.093	0.193	0.112	2.37	2.02

- Dataset D2

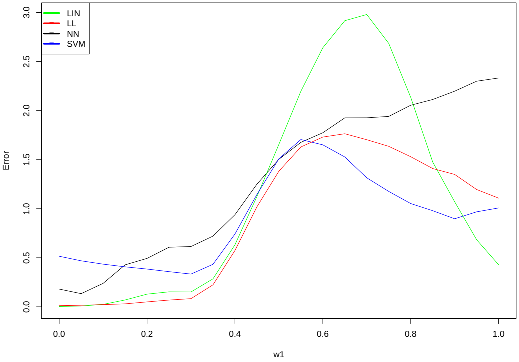
LIN	LL	NN	SVM	RAND	LOCAL
1.17	0.88	1.30	0.91	3.21	2.91

Dataset D1: Prediction of the maximum

w_1 denotes the weight of the criterion 1 in the multicriteria evaluation function $y = w_1 y_1 - w_2 y_2$



Dataset D2: Prediction of the maximum

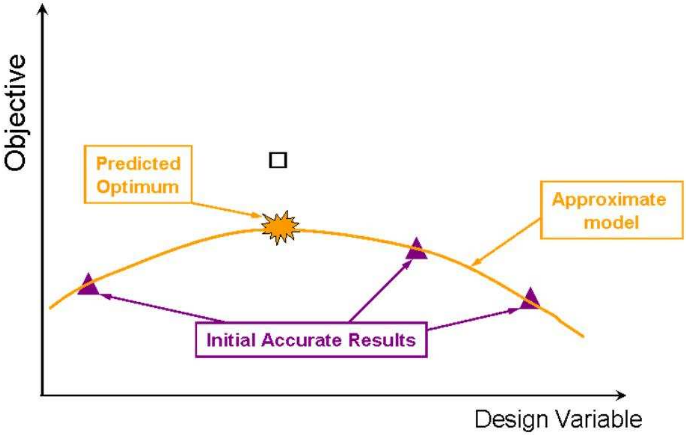


Note how the difficulty of the problem strongly varies with the weighting.

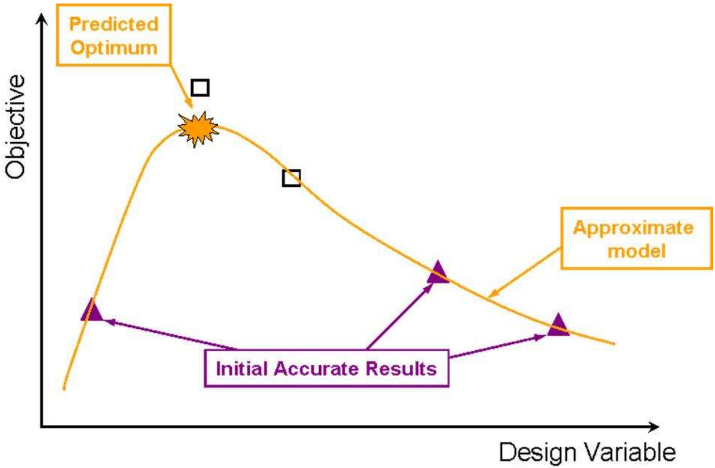
Experimental design for optimization

- So far we have considered the case where the learned model substitutes the simulator.
- There is however another configuration where learned model can be employed: predict where to perform the next experiment(s).
- Several algorithms have been proposed in machine learning [1]:
  - PMAX: next experiment is taken at the point which maximizes the estimate of E. Maximization can be performed by conventional optimization algorithms, e.g. evolutionary algorithms [7].
  - IEMAX: next experiment is taken at the point which maximizes the "optimistic model," i.e. the 95th-percentile values of the model's predicted confidence intervals.
  - Q2 algorithm [2]: it combines PMAX and the principle of Response Surface Methods by adopting local learning techniques (local quadratic approximation).

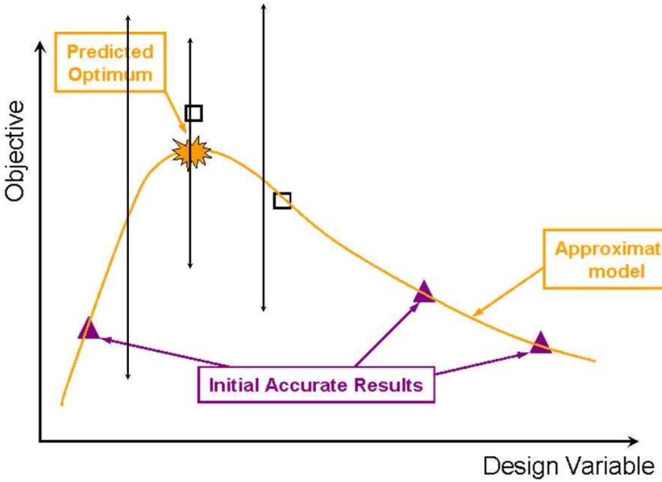
PMAX [7]


Objective
Predicted Optimum
Approximate model
Initial Accurate Results
Design Variable

PMAX


Predicted Optimum
Objective
Approximate model
Initial Accurate Results
Design Variable

PMAX


Predicted Optimum
Approximate model
Initial Accurate Results
Objective
Design Variable

Considerations

- In front of excessive uncertainty, does it make really sense to take the estimated maximum for next simulation?
- Or rather select the point which can improve more the probability of correct selection?
- Exploration vs. exploitation issues.
- Confidence returned by learner allows to take advantage of techniques for dealing with stochastic optimization problems, e.g. bandit algorithms.
- Machine learning does not provide any more the solution but an ingredient to help finding a solution.

Conclusions

- Design space exploration is a complex task in industrial settings (e.g. aereonautics) which demands considerable human intervention and insight.
- Most existing simulators are lenghty and expensive to run.
- Machine learning techniques can take advantage of collected simulated data and either replace or support the simulator in the design optimization process.
- In particular, multivariate and multicriteria tasks represent an important domain of application.
- Time for real multi-input multi-output techniques?
- Demand for learning techniques able to provide accurate predictions together with confidence intervals.

Future work

Development of specific optimization techniques to deal with estimated value functions. They should take into account

- the properties of the learning algorithm in the optimization strategy
- exploration vs. exploitation issues: notions of probability of correct selection
- cost of additional simulations vs. cost of optimization of an estimated value function
- cost of optimization vs. increased uncertainty (e.g. reduction of dimensionality at the cost of a less accurate evaluation function).

- Jeff Schneider Andrew Moore.
  Memory-based stochastic optimization.
  In D. Touretzky, M. Mozer, and M. Hasselm, editors, *Neural Information Processing Systems 8*, volume 8, pages 1066–1072. MIT Press, 1996.

- Justin Boyan Mary Soon Lee Andrew Moore, Jeff Schneider.
  Q2: Memory-based active learning for optimizing noisy continuous functions.
  In J. Shavlik, editor, *Proceedings of the Fifteenth International Conference of Machine Learning*, pages 386–394, 340 Pine Street, 6th Fl., San Francisco, CA 94104, 1998. Morgan Kaufmann.

- M. Birattari, G. Bontempi, and H. Bersini.
  Lazy learning meets the recursive least-squares algorithm.
  In M. S. Kearns, S. A. Solla, and D. A. Cohn, editors, *NIPS 11*, pages 375–381, Cambridge, 1999. MIT Press.

- G. Bontempi.

Local Learning Techniques for Modeling, Prediction and Control.
PhD thesis, IRIDIA- Université Libre de Bruxelles, 1999.

G. Bontempi.
Long term time series prediction with multi-input multi-output local learning.
In *Proceedings of the 2nd European Symposium on Time Series Prediction (TSP), ESTSP08*, 2008.

G. Bontempi, O. Caelen, S. Pierret, and C. Goffaux.
On the use of supervised learning techniques to speed up the design of aeronautics components.
WSEAS Transactions on Systems, 10(3):3098–3103, 2005.

Rajan Filomeno Coelho.
Optimization based on evolutionary algorithms for aeronautics.

Electronic presentation, 2006.

Andrew Moore Justin Boyan.

Learning evaluation functions for global optimization and boolean satisfiability.
In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, pages 3–10, 1998.

Andrew Moore Oded Maron.
The racing algorithm: Model selection for lazy learners.
In *Artificial Intelligence Review*, volume 11, pages 193–225, April 1997.

S. Pierret, A. Demeulenaere, B. Gouverneur, C. Hirsch, and R. Van den Braembussche.
Designing turbomachinery blades with the function approximation concept and the navier-stokes equations.
In *Proceeding of the 8th AIAA/NASA/USAF/ISSMO Symposium on MDO*, 2000.

S. Pierret and R.A. Van den Braembussche.
Turbomachinery blade design using a navier-stokes solver and artificial neural network.

ASME Trans., Journal of Turbomachinery, 121(2):326–332, 1999.

P. van den Hamer, W.P.M. van der Linden, P. Bingley, and N.W. Schellingerhout.
A system simulation framework.
In *Proceedings of the 37th Conference on Design Automation*, pages 699–704, 2000.