Robust construction of a spatio-temporal surrogate model - Application in thermal engineering

Jonathan Guerra[1 3], Patricia Klotz[1],
Béatrice Laurent[2] and Fabrice Gamboa[2]

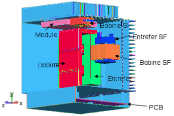
April 8$^{\text{th}}$, 2015

[1]ONERA
[2]Institut de Mathématiques de Toulouse
[3]Epsilon Ingénierie

Table of Contents

Physical problem: Electronic equipment in the avionic bay

- Physical modeling of an avionic bay isn't easy: numerous interactions between the equipment, fluid dynamics, radiation...



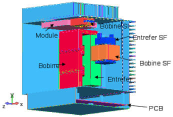
- The equations to solve are Navier-Stokes ones coupled with heat equation.

→ **Thermo-fluidic modeling to perform**. Several tools exist:

- ◇ Commercial softwares (FLOTHERM, Fluent,etc ...)
- ◇ ONERA softwares (CEDRE coupling CHARME and ACACIA)
- ◇ Physical reduced models such as nodal models

Physical problem: Electronic equipment in the avionic bay

- Physical modeling of an avionic bay isn't easy: numerous interactions between the equipment, fluid dynamics, radiation...



- Because there are applications where an extensive use of numerical simulations is necessary:
  1. Optimization (of the lifetime of the equipment for instance)
  2. Multilevel simulation of the avionic bay
  3. etc...

$\Longrightarrow$ There is a need of surrogate models in thermal engineering!

Surrogate model

- Let $\mathbf{y} = f(\mathbf{x})$ $\mathbf{y} \in \mathbb{R}^{N_y}$, $\mathbf{x} \in \mathbb{R}^{N_x}$ with f the costly reference model
- Surrogate model: **Low-cost analytic** model adjusted to the reference model **thanks to observations** coming from it $\{\mathbf{x}_i, f(\mathbf{x}_i)\}$
- Construction of the surrogate $\hat{F}(\mathbf{x}\ ;\ \mathbf{w})$ from the observations:

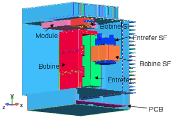
$$\begin{array}{cccc} \hat{F} & : & \mathbb{R}^{N_x} & \longrightarrow & \mathbb{R}^{N_y} \\ & & \mathbf{x} & \longmapsto & \hat{F}(\mathbf{x}\ ;\ \mathbf{w}) = \hat{\mathbf{y}} \end{array}$$

By solving:

$$\mathbf{w} = \underset{\mathbf{v} \in \mathbb{R}^{N_w}}{\operatorname{argmin}} \left\{ \sum_{j \in \mathcal{A}} \left\| \hat{F}(\mathbf{x}_j\ ;\ \mathbf{v}) - \mathbf{y}_j \right\|_2^2 \right\}$$

Physical problem: Electronic equipment in the avionic bay

- Physical modeling of an avionic bay isn't easy: numerous interactions between the equipment, fluid dynamics, radiation...



$\Longrightarrow$ This complex modeling implies for the surrogate:

- ◇ Few learning trajectories
- ◇ The input/output dimension can be important
- ◇ Construction time must be reasonable
- ◇ Long-term in time prediction

Table of Contents

Surrogate model for transient thermal engineering

NARX (Non-linear AutoRegressive with eXogenous inputs) time series are quite adapted. Their general form is:

$$\hat{\mathbf{y}}^k = \hat{F}\left(\hat{\mathbf{y}}^{k-1}, \cdots, \hat{\mathbf{y}}^{k-p}, \mathbf{u}^k, \cdots, \mathbf{u}^{k-q}\right) \tag{1}$$

Moreover, the phenomenon follows heat equation so a first order in time phenomenon:

$$\frac{\partial T}{\partial t}(x,t) = D\Delta T(x,t) + C$$

By discretizing temporally this equation $t_k = k\Delta t$ with $k \in [0, N_t]$, and by identification with (1), an interesting use of the surrogate $\hat{F}$ in this case is:

$$\boxed{\left(T_1^k, \cdots, T_{N_p}^k\right) = \hat{F}\left(T_1^{k-1}, \cdots, T_{N_p}^{k-1}, \mathbf{u}^k\right)} \tag{2}$$

with N_p the number of points of interest

Dynamical model for spatio-temporal prediction

- Principle: a recursive formulation is used

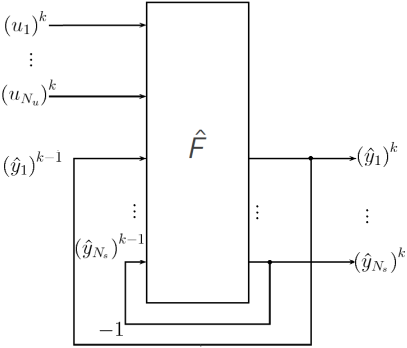
- It means that the outputs at time t_{k-1} become the inputs of the same model at time t_k

- It can be written

$$\begin{cases} \hat{\mathbf{y}}^k = \hat{F}\left(\hat{\mathbf{y}}^{k-1}, \mathbf{u}^k \; ; \mathbf{w}\right) \\ \hat{\mathbf{y}}^0 = \mathbf{y}^0 \end{cases}$$

- The parameters $\mathbf{w}$ are the one minimizing:

$$E_{learning} = \sum_{j=1}^{N_y} \sum_{sample}^{l} \sum_{k=1}^{N_t} \left\| y_j^{k,l} - \hat{y}_j^{k,l} \right\|_2^2$$



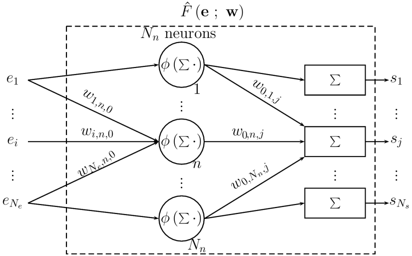
- Remark: $\hat{F}$ does not depend on time

Definition of an artificial neural network - Multi Layer Perceptron

Mathematically, an artificial neural network can be written:

$$\{\mathbf{s}\}_{j\in[\![1,N_s]\!]} = \left\{\hat{F}\left(\mathbf{e};\mathbf{w}\right)\right\}_{j\in[\![1,N_s]\!]} = \left\{\sum_{n=1}^{N_n} w_{0,n,j}\,\phi\left(\sum_{i=1}^{N_e} w_{i,n,0}e_i + w_{0,n,0}\right) + w_{0,0,j}\right\}_{j\in[\![1,N_s]\!]}$$

ϕ is the logistic function



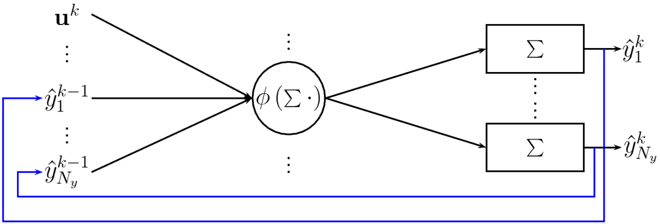
Derivative of the output s_j relatively to its weights:

$$\frac{\partial s_j}{\partial w} = \frac{\partial \hat{F}_j(\mathbf{e}\ ;\mathbf{w})}{\partial w}$$

Neural network for spatio-temporal prediction

For spatio-temporal prediction:

$$\begin{cases} \hat{\mathbf{y}}^k = \hat{F}\left(\hat{\mathbf{y}}^{k-1}, \mathbf{u}^k \; ; \mathbf{w}\right) \\ \hat{\mathbf{y}}^0 = \mathbf{y}^0 \end{cases}$$



Derivative of the output of the network relatively to its weights:

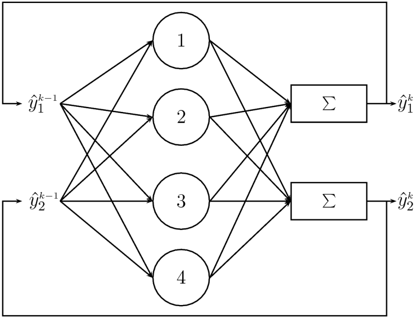
$$\frac{\partial \hat{y}_j^k}{\partial w} = \left.\frac{\partial \hat{F}_j(\mathbf{u}^k, \mathbf{x} \; ; \mathbf{w})}{\partial w}\right|_{\mathbf{x}=\hat{\mathbf{y}}^{k-1}} + \sum_{m=1}^{N_y} \frac{\partial \hat{F}_j\left(\mathbf{u}^k, \hat{\mathbf{y}}^{k-1} \; ; \mathbf{w}\right)}{\partial \hat{y}_m^{k-1}} \frac{\partial \hat{y}_m^{k-1}}{\partial w}$$

Table of Contents

Table of Contents

Multilevel optimization - Justification with 2 outputs and 4 neurons

Theoretically, the optimization of the weights should be performed on the following complete network:

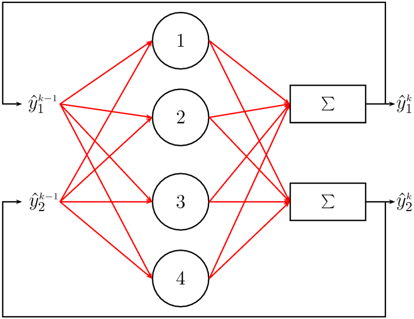


$$\begin{cases} \hat{y}_1^k = \hat{F}_1\left(\hat{y}_1^{k-1}, \hat{y}_2^{k-1}; \mathbf{w}\right) \\ \hat{y}_2^k = \hat{F}_2\left(\hat{y}_2^{k-1}, \hat{y}_1^{k-1}; \mathbf{w}\right) \\ \hat{y}_j^0 = y_j^0 \end{cases}$$

$$\min_{\mathbf{w}} \left\{ \sum_{j=1}^{2} \sum_{sample}^{l} \sum_{time}^{k} \left\| y_j^{k,l} - \hat{y}_j^{k,l} \right\|_2^2 \right\}$$

Multilevel optimization - Justification with 2 outputs and 4 neurons

Theoretically, the optimization of the weights should be performed on the following complete network:



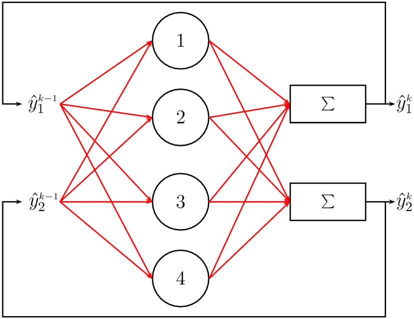
$$\begin{cases} \hat{y}_1^k = \hat{F}_1\left(\hat{y}_1^{k-1}, y_2^{k-1}; \mathbf{w}\right) \\ \hat{y}_2^k = \hat{F}_2\left(\hat{y}_2^{k-1}, y_1^{k-1}; \mathbf{w}\right) \\ \hat{y}_j^0 = y_j^0 \end{cases}$$

$$\min_{\mathbf{w}} \left\{ \sum_{j=1}^{2} \sum_{sample}^{l} \sum_{time}^{k} \left\| y_j^{k,l} - \hat{y}_j^{k,l} \right\|_2^2 \right\}$$

But it requires to solve a high dimensional optimization problem: for instance with 6 outputs, 4 exogenous variables and 10 neurons, there are 176 weights to fit.

Multilevel optimization - Justification with 2 outputs and 4 neurons

Theoretically, the optimization of the weights should be performed on the following complete network:



$$\begin{cases} \hat{y}_1^k = \hat{F}_1\left(\hat{y}_1^{k-1}, y_2^{k-1}; \mathbf{w}\right) \\ \hat{y}_2^k = \hat{F}_2\left(\hat{y}_2^{k-1}, y_1^{k-1}; \mathbf{w}\right) \\ \hat{y}_j^0 = y_j^0 \end{cases}$$

$$\min_{\mathbf{w}} \left\{ \sum_{j=1}^{2} \sum_{sample}^{l} \sum_{time}^{k} \left\| y_j^{k,l} - \hat{y}_j^{k,l} \right\|_2^2 \right\}$$

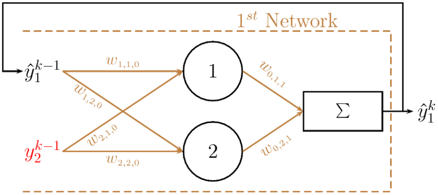
$\Rightarrow$ This great dimension implies a degraded solving of the weights.

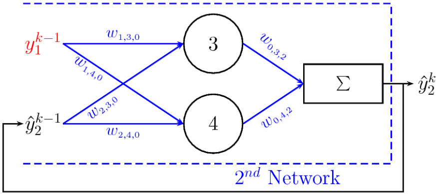
Solution: A multilevel framework has to be introduced to overcome this problem : the optimization is decomposed **output by output**

Initialization : Optimization with measured inputs (Teacher Forcing way)

$\forall j \in [\![1, N_y]\!]$, the weights $\mathbf{w_j^0}$ are optimized for each output y_j. The number of neurons is chosen at this step.

Example with two outputs and two neurons per network: ◂ Back



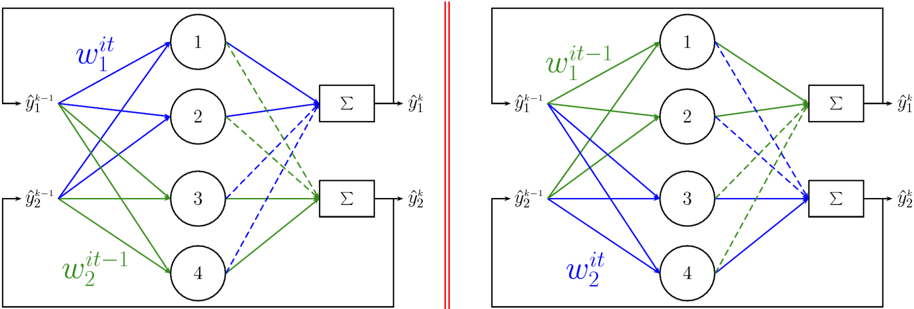


$$\min_{\mathbf{w_1^0}} \left\{ \sum_{l=1}^{N_l} \sum_{k=1}^{N_t} \left(y_1^{k,l} - \hat{F}_1\left(\hat{y}_1^{k-1,l}, y_2^{k-1,l}; \mathbf{w_1^0}\right) \right)^2 \right\}$$

$$\min_{\mathbf{w_2^0}} \left\{ \sum_{l=1}^{N_l} \sum_{k=1}^{N_t} \left(y_2^{k,l} - \hat{F}_2\left(\hat{y}_2^{k-1,l}, y_1^{k-1,l}; \mathbf{w_2^0}\right) \right)^2 \right\}$$

Iteration it > 0 : Optimization with predicted inputs

- Only weights $\mathbf{w}_j$ that are used to compute the output y_j are being optimized at iteration *it* while the other outputs are computed thanks to the weights optimized at the last step $it - 1$. Back



$$\min_{\mathbf{w}_1^{it}} \left\{ \sum_{l=1}^{N_l} \sum_{k=1}^{N_t} \left(y_1^{k,l} - \hat{F}_1\left(\hat{y}_1^{k-1,l}, \hat{y}_2^{k-1,l}; \mathbf{w}_1^{it}, \mathbf{w}_2^{it-1}\right) \right)^2 \right\}$$

$$\min_{\mathbf{w}_2^{it}} \left\{ \sum_{l=1}^{N_l} \sum_{k=1}^{N_t} \left(y_2^{k,l} - \hat{F}_2\left(\hat{y}_2^{k-1,l}, \hat{y}_1^{k-1,l}; \mathbf{w}_2^{it}, \mathbf{w}_1^{it-1}\right) \right)^2 \right\}$$

Multilevel optimization - Recap of the process

❶ **Initialization of the process:** Complexity selection and optimization of the weights on one-output networks with measured inputs (except for the one corresponding to the output computed by the network)
▸ Detail

❷ **For it > 0:** The network is initialized with w^{it-1} optimized at $it - 1$, and the weights involved in the computation of each y_j are separately optimized
▸ Detail

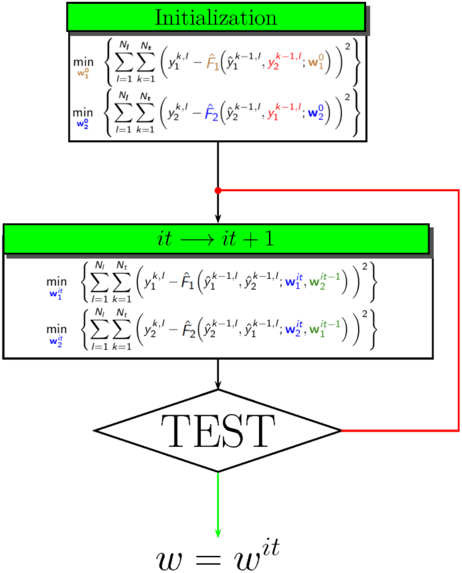


Table of Contents

Model Selection by Cross Validation

- Model selection gives an estimation of the generalization error which is used to
  - ◇ Choose the complexity (number of neurons)
  - ◇ Stop the multilevel optimization

- Why Cross Validation?
  - ◇ It uses all the samples at disposal by resampling

- Principle :
  - ◇ Samples at disposal are splitted in 5 groups (also called folds)
  - ◇ One of the folds is only used to test (test samples) the network built with the 4 other ones (learning samples)

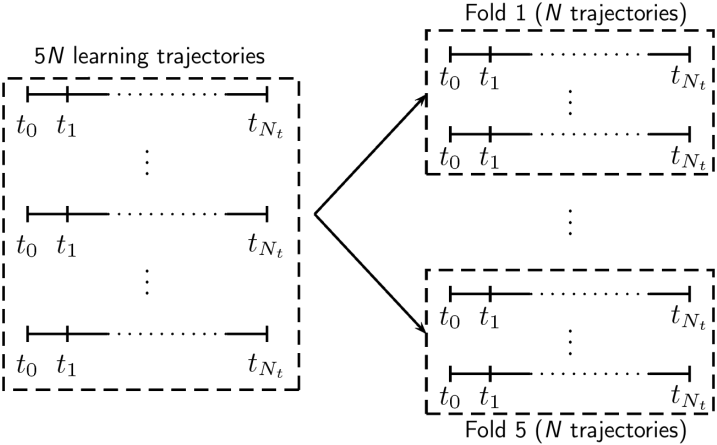
  $\Longrightarrow$ 5 models are constructed

- The final model is obtained by averaging the outputs of those 5 models :

$$\hat{F}_j = \frac{1}{5} \sum_{\text{fold}=1}^{5} \hat{F}_j^{\text{fold}}$$

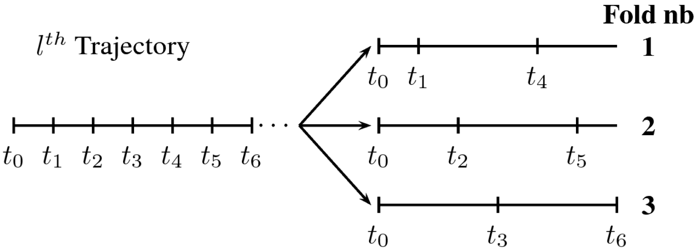
Cross Validation- How to split spatio-temporal examples?

- Whole trajectories: problem of robustness (because some fold can poorly fill the entire input space)



Cross Validation- How to split spatio-temporal examples?

- Whole trajectories: problem of robustness (because some fold can poorly fill the entire input space)
  - $\rightarrow$ Solution: stop considering entire trajectories, and split the time steps into the folds (Subtrajectories)



Cross Validation- How to split spatio-temporal examples?

- Whole trajectories: problem of robustness (because some fold can poorly fill the entire input space)
  - $\rightarrow$ Solution: stop considering entire trajectories, and split the time steps into the folds (Subtrajectories)
- Subtrajectories: problem of proximity between test folds and learning folds

Cross Validation- How to split spatio-temporal examples?

- Whole trajectories: problem of robustness (because some fold can poorly fill the entire input space)
  - → Solution: stop considering entire trajectories, and split the time steps into the folds (Subtrajectories)

- Subtrajectories: problem of proximity between test folds and learning folds
  - → Solution: get rid of the redundant points of the learning examples before dividing them into the folds ▸ Criteria

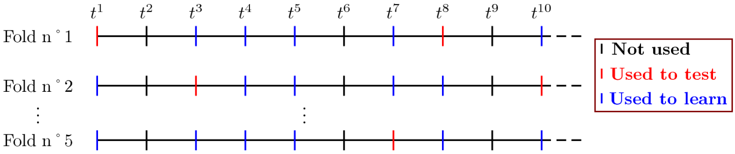


Table of Contents

Sensitivity Analysis - Need

- The number of input dimensions can be quite important. This implies:
  - ⇒ An important number of weights to optimize
  - ⇒ A degraded solution when they are optimized

- Moreover, the aim is to build a parcimonious model (a point of interest is not equally influenced by the other ones!)

- Solution: sensitivity analysis is used on the surrogate model to quantify the impact of each input. Once the non-influent inputs are detected, a new network is built without those inputs.

Process

Steps

1. A first network is built
2. Non-influent inputs are determined (thanks to sensitivity analysis)
3. A new network is built without the non-influent inputs of step 2 (so with less dimensions)
4. The result after and before Sensitivity Analysis can be compared

Sensitivity Analysis - Choice of the method

- Tool: DGSM (Derivative-based Global Sensitivity Measures)
- Its application: Because it does not depend on time, the sensitivity analysis will be applied directly on the artificial neural network
- Limit: strong correlations between the inputs of each models (the signification of those coefficient is not obvious in this case)

Sensitivity Analysis - Definition of DGSM

- Explanation: it is based on the fact that if the derivative of the model output relatively to one of its inputs is important, it means that this input has a great influence on this output (at least locally)

- It is defined as follows:
  To measure the influence of an input $x_j \in H$ on the output $y = \hat{F}(\mathbf{x}\ ; \mathbf{w})$, one has to compute:

$$\nu_j = \mathbb{E}\left[\left(\frac{\partial \hat{F}(\mathbf{X}\ ; \mathbf{w})}{\partial x_j}\right)^2\right] = \int_{H^{N_x}} \left(\frac{\partial \hat{F}(\mathbf{x}\ ; \mathbf{w})}{\partial x_j}\right)^2 d\mu(\mathbf{x})$$

- With the hypothesis that $\frac{\partial \hat{F}(\mathbf{X};\ \mathbf{w})}{\partial x_j}$ is square-integrable, those coefficients exist

Sensitivity Analysis - Application to the network

The **derivative of a neural network relatively to one of its inputs** has an analytical formulation:

$$\frac{\partial \hat{F}_k(\mathbf{x}\,;\mathbf{w})}{\partial x_j} = \sum_{n=1}^{N_n} w_{0,n,k} w_{j,n,0} \phi' \left(\sum_{i=1}^{N_e} w_{i,n,0} x_i + w_{0,n,0} \right)$$

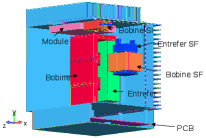
- k being the index of the output
- x_j being an exogenous variable or an output at the previous time step

$\Rightarrow$ The mathematical definition of those coefficient gives a meaning to them

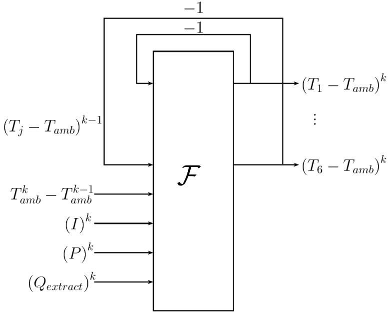
Table of Contents

Presentation of the test case

- To evaluate the presented methodology, a mathematical model of an equipment is used (Thermal nodal network model):

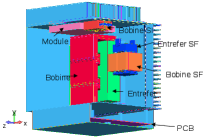


- The neural network built has 4 exogenous inputs and 6 outputs which represent:
  1. T upper wall
  2. T left wall
  3. T radiator
  4. T air
  5. T component
  6. T wall next to the component

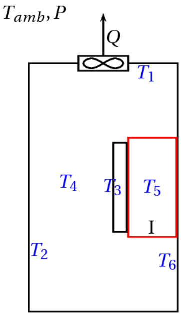


Presentation of the test case

- To evaluate the presented methodology, a mathematical model of an equipment is used (Thermal nodal network model):



- The neural network built has 4 exogenous inputs and 6 outputs which represent:
  1. T upper wall
  2. T left wall
  3. T radiator
  4. T air
  5. T component
  6. T wall next to the component



Design of Experiment

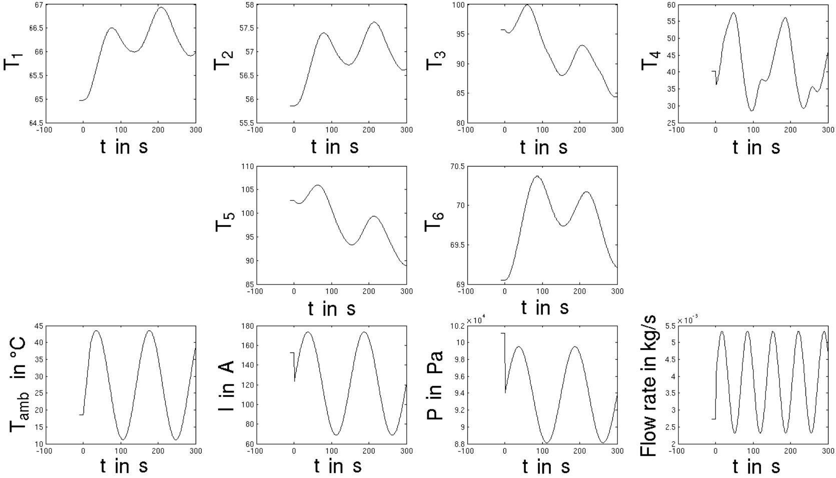
- The Design of Experiment is composed of 37 trajectories of 300s each.

- They are generated from the following set of inputs in the thermal network model: $(T(t=0), T_{amb}, I, P,$Flow rate).

  Those boundary conditions and forcing terms are chosen randomly (on the basis of the sinusoid or the crenel functions)

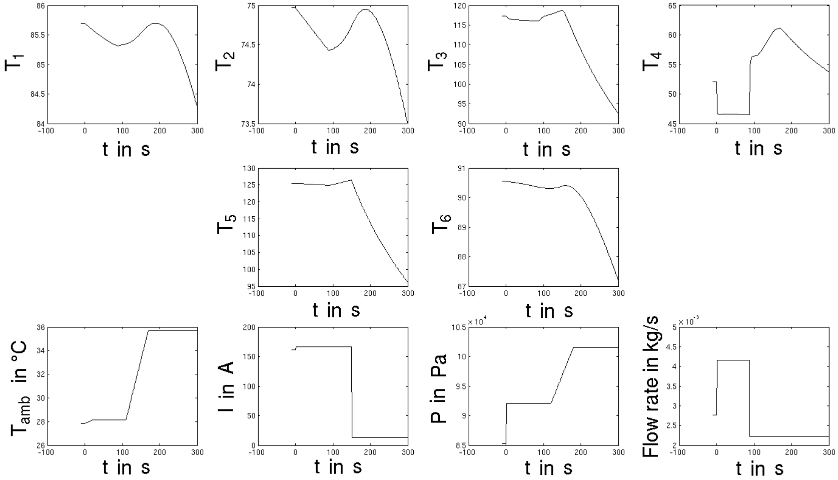
Design of Experiment

- The Design of Experiment is composed of 37 trajectories of 300s each.



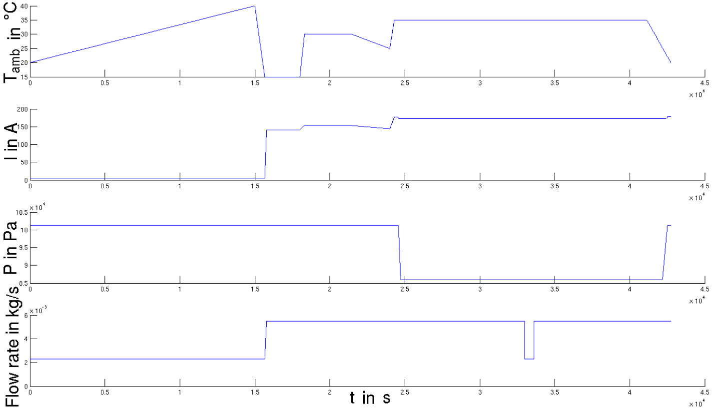
Design of Experiment

- The Design of Experiment is composed of 37 trajectories of 300s each.



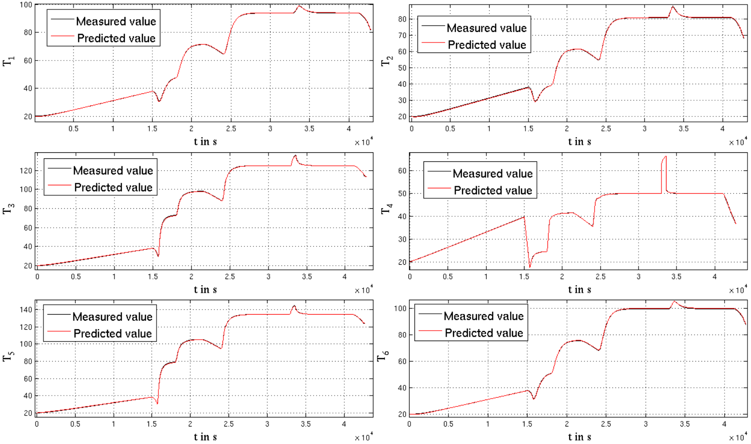
Test sample - Result of Multilevel optimization

- The test sample is an in-flight profile which has the following exogenous inputs:



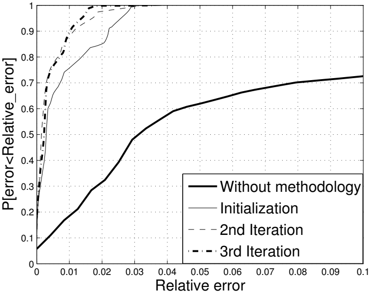
Test sample - Result of Multilevel optimization

- The test sample is an in-flight profile which has the following exogenous inputs:
- After optimization, it gives this result on the six outputs:



Multilevel splitting result

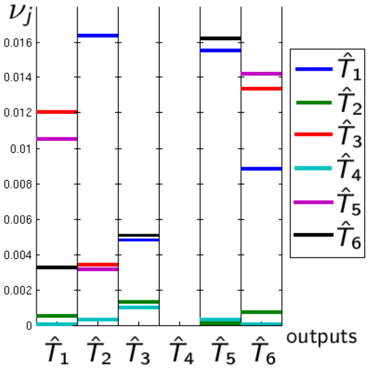
- To proove the benefits of the methodology, multilevel splitting and construction without it are compared on 37 test trajectories (including the in-flight profile). Here are the results:



- Without methodology, the result is obtained with 30 neurons. With the methodology, it is able to go up to 47

Sensitivity Analysis: Computation of the coefficients

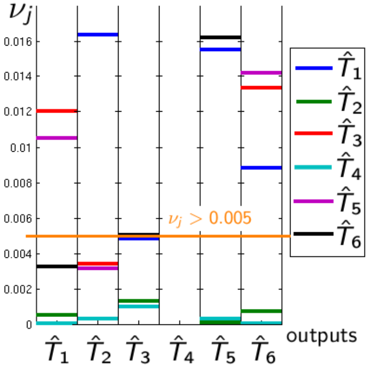
- The DGSM coefficients are computed on the outputs of the previously built network



Sensitivity Analysis: Computation of the coefficients

- The DGSM coefficients are computed on the outputs of the previously built network

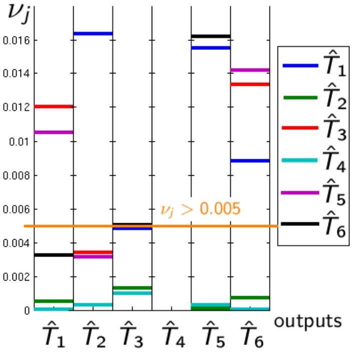
only the inputs j with $\nu_j > 0.005$ are kept:



Sensitivity Analysis: Computation of the coefficients

- The DGSM coefficients are computed on the outputs of the previously built network

only the inputs j with $\nu_j > 0.005$ are kept:



$\Longrightarrow$

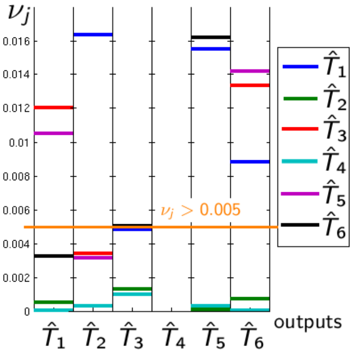
This gives the "matrix of influences":

$T_{amb} \rightarrow$	1	1	1	1	1	1
$P \rightarrow$	1	1	1	1	1	1
$Q \rightarrow$	1	1	1	1	1	1
$I \rightarrow$	1	1	1	1	1	1
$\hat{T}_1 \rightarrow$	1	1	0	1	1	1
$\hat{T}_2 \rightarrow$	0	1	0	1	0	0
$\hat{T}_3 \rightarrow$	1	0	1	1	1	1
$\hat{T}_4 \rightarrow$	0	0	0	1	0	0
$\hat{T}_5 \rightarrow$	1	0	1	1	1	1
$\hat{T}_6 \rightarrow$	0	1	1	1	1	1
	$\downarrow$ $\hat{T}_1$	$\downarrow$ $\hat{T}_2$	$\downarrow$ $\hat{T}_3$	$\downarrow$ $\hat{T}_4$	$\downarrow$ $\hat{T}_5$	$\downarrow$ $\hat{T}_6$

Sensitivity Analysis: Computation of the coefficients

- The DGSM coefficients are computed on the outputs of the previously built network

only the inputs j with $\nu_j > 0.005$ are kept:



$\Longrightarrow$

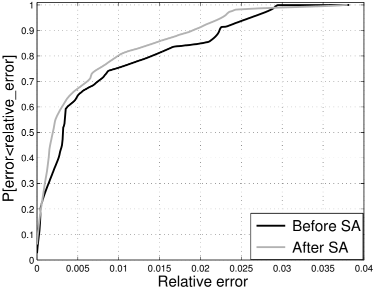
This gives the "matrix of influences":

$$
\begin{array}{cc}
T_{amb} & \rightarrow \\
P & \rightarrow \\
Q & \rightarrow \\
I & \rightarrow \\
\hat{T}_1 & \rightarrow \\
\hat{T}_2 & \rightarrow \\
\hat{T}_3 & \rightarrow \\
\hat{T}_4 & \rightarrow \\
\hat{T}_5 & \rightarrow \\
\hat{T}_6 & \rightarrow \\
\end{array}
\begin{pmatrix}
1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 0 & 1 & 1 & 1 \\
0 & 1 & 0 & 1 & 0 & 0 \\
1 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 1 & 1 & 1 & 1 \\
0 & 1 & 1 & 1 & 1 & 1 \\
\downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\
\hat{T}_1 & \hat{T}_2 & \hat{T}_3 & \hat{T}_4 & \hat{T}_5 & \hat{T}_6
\end{pmatrix}
$$

- Remark: The air temperature needs all the other outputs to be computed, but the other outputs don't need it in input (the flow rate and ambient temperature suffice in this case)

Sensitivity Analysis: Result

- A new network is built with less inputs for each output.
- The result before and after sensitivity analysis can now be compared:



- By decreasing the size of the optimization problem (378 weights instead of 695 or 245 if we get rid of $\hat{T}_2$ and $\hat{T}_4$), the solution obtained is better

Conclusions and perspectives

- This presentation has introduced an innovative construction of a spatio-temporal surrogate model based on :
  - ◇ A multilevel framework to optimize the weights
  - ◇ Cross validation for the model selection
  - ◇ Sensitivity analysis to reduce the input dimension

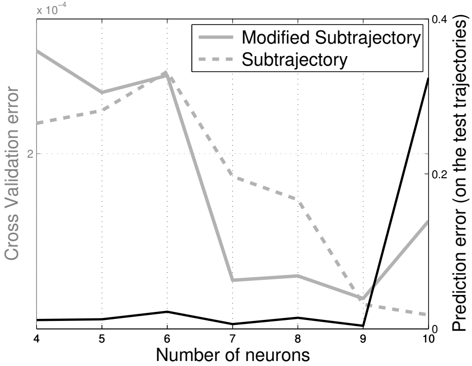
- Concerning the perspectives:
  - ◇ Validation of the construction of a spatio-temporal design of experiment
  - ◇ The ability to propagate the uncertainties thanks to this surrogate has to be proven
  - ⇒ We have to manage the errors due to the RNN in the propagation and to be able to compute the implied bias on the quantile estimation

THANK YOU

Questions?

Improvement of the model selection

By getting rid of some of the time steps, the model selection is more effective. For instance here is a comparison of the cross validation error on the test case with the actual prediction error:



Iterative construction of a spatio-temporal DOE

- We propose to build iteratively a DOE of N_l trajectories from a pool of N_l' ($> N_l$) trajectories.
- To do that, we need to define a criterion which will allow us to choose the best trajectory to add at each step.
- We propose to generalize a static criterion to the spatio-temporal case

Spatial DOE: Maxmin criterion

- In the static case, a point $x_l^\star$ is added to the DOE if it maximizes the distance to the current DOE.

$$x_l^\star = \underset{x_l \notin DOE}{\mathrm{Argmax}} \min_{\bar{x} \in DOE} \|x_l - \bar{x}\|_2$$

- In the spatio-temporal case, trajectories T^l are considered as sets of points (without considering time)

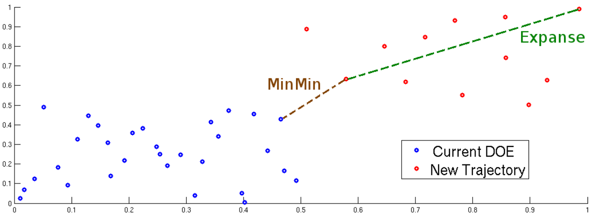
$$\begin{cases} T_l & = & \left\{ \mathbf{x}^{k,l} \ : \ k \in [\![0, N_t]\!] \right\} \\ DOE & = & \{ T_l : l \in [\![1, N_l]\!] \} \end{cases}$$

- We then have to generalize the Maxmin criterion to a set of points

Maxmin with sets of points

- An additional criterion needs to be introduced to extend Maxmin:

$$\begin{cases} \text{Maxmin}(T_l, DOE) & = & \max_{x \in T_l} \min_{\bar{x} \in DOE} \|x - \bar{x}\|_2 \\ \text{Minmin}(T_l, DOE) & = & \min_{x \in T_l} \min_{\bar{x} \in DOE} \|x - \bar{x}\|_2 \\ \text{Expanse}(T_l, DOE) & = & \text{Maxmin}(T_l, DOE) \\ & & -\text{Minmin}(T_l, DOE) \end{cases}$$

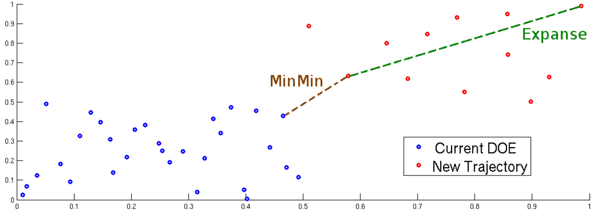


- Because one of them can be found with the two others by linear combination, it is possible to use only Minmin (translation of the previous Maxmin) and Expanse

Spatio-temporal design of experiments

- A trajectory T_l is added to the DOE if it is solution of:

$$\max_{T_k \not\subset DOE_{current}} \mathrm{Minmin}(T_k, DOE_{current})$$
$$\max_{T_k \not\subset DOE_{current}} \mathrm{Expanse}(T_k, DOE_{current})$$



- Remark: the first trajectory to add will simply be the one maximizing the Expanse criterion

Criteria to exclude some points

1. First one: The points t_k **are kept** if:

$$\begin{cases} \left|\overline{\frac{\partial y_j}{\partial t}}(t_k)\right| + \left|\overline{\frac{\partial y_j}{\partial t}}(t_{k+1})\right| > k_1 & \text{if } k \neq N_t \\ \left|\overline{\frac{\partial y_j}{\partial t}}(t_{N_t})\right| > k_2 & \text{elsewhere} \end{cases}$$

2. Second one: time between two time steps kept **cannot exceeds**

$$\text{threshold} = 5 \cdot \Delta t = 5 \cdot \frac{t_{N_t} - t_0}{N_t}$$

3. Third one: if t_i is **time step kept** and if t_j is as $j > i + 1$ and does not verify the two first conditions, t_j is kept if :

$$\left|\sum_{k=i+1}^{j} \overline{\frac{\partial y_j}{\partial t}}(t_k)\right| > k_3$$

Back