

# Apprentissage de dynamiques spatio-temporelles par réseaux de neurones artificiels

Matthias De Lozzo

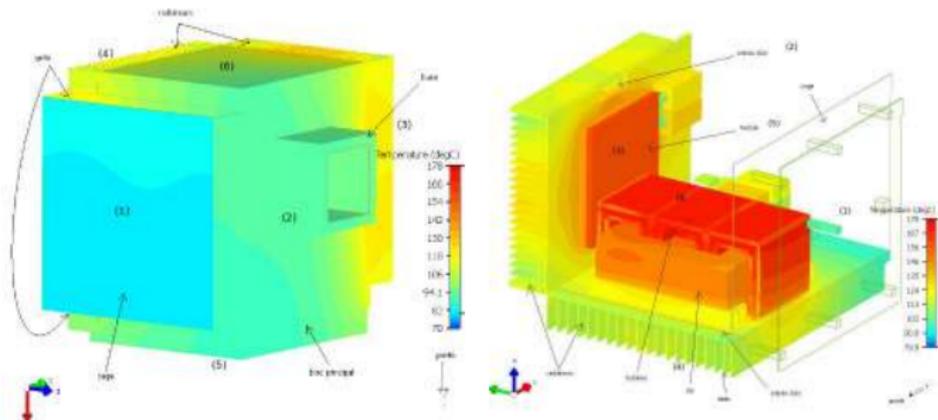
thèse encadrée par Béatrice Laurent (IMT) et Patricia Klotz (ONERA)



Journées MASCOT NUM 2012 - Mercredi 21 mars 2012

## Présentation de l'applicatif aéronautique

- ▶ **Composants électriques et électroniques**
- ▶ **Système de refroidissement** : entrées latérales + buse de sortie
- ▶ **Code numérique 3D** ↔ équations de la physique



**2 grands types de comportements**  $\neq$  :

- ▶ **convection forcée** : extraction d'air fonctionnant à 100% ;
- ▶ **convection naturelle** : extraction d'air en panne.

## Objectif

On s'intéresse à l'**estimation de la température** :

- ▶ **en certains points** d'intérêt ("les **sorties  $\underline{y}$** "),
- ▶ **à différents instants** et
- ▶ **pour différents cas tests** paramétrés par :
  - ▶  $u_1$ , l'**entrée** "pression" : sol ou vol ;
  - ▶  $u_2$ , l'**entrée** "débit d'extraction" : de 0 à normal ;  **robustesse**
  - ▶  $u_3$ , l'**entrée** "intensité du courant" ;
  - ▶  $u_4$ , l'**entrée** "température ambiante" ;
  - ▶ une température initiale (si régime transitoire) ;
  - ▶ une durée de mission (si régime transitoire).

On souhaite une estimation **rapide**, **précise** et **robuste**.

Pour cela, on cherche à construire :

un **modèle de substitution en régime transitoire et permanent**

⇒ **apprentissage statistique** de couples **entrées-sorties**  $(\underline{u}_i, \underline{y}_i)_{i \in \{1, \dots, n\}}$  issus du code numérique 3D.

# Plan

## Formulation du problème

## Modèle de substitution proposé

- Travaux existants

- Modélisation d'une dynamique d'ordre 1

- Résultats numériques

## Sélection de modèles

- Cadre de la régression homoscédastique

- Sélection de modèles non asymptotique

- Résultats numériques

# Plan

## Formulation du problème

### Modèle de substitution proposé

Travaux existants

Modélisation d'une dynamique d'ordre 1

Résultats numériques

### Sélection de modèles

Cadre de la régression homoscédastique

Sélection de modèles non asymptotique

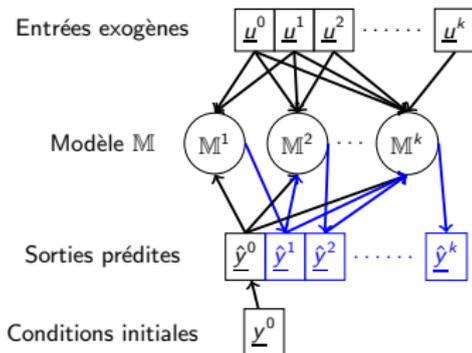
Résultats numériques

# Apprentissage statistique de données temporelles

Entrées :  $\underline{u} = (u_1, \dots, u_{n_u}) \in \mathcal{U} \subset \mathbb{R}^{n_u}$  ; sorties :  $\underline{y} = (y_1, \dots, y_{n_y}) \in \mathcal{Y} \subset \mathbb{R}^{n_y}$

Régime	permanent	transitoire
Entrées du modèle de référence	$\underline{u}$	$\underline{u}(t)$
Sorties du modèle de référence	$\underline{y}$	$\underline{y}(t)$
Base d'apprentissage	$(\underline{u}_i; \underline{y}_i)_{i \in \{1, \dots, n\}}$	$(\underline{u}_i^{k_i}; \underline{y}_i^{k_i})_{\substack{i \in \{1, \dots, n\} \\ k_i \in \{0, \dots, K_i\}}}$
Critère à minimiser (MSE)	$\sum_{1 \leq i \leq n} \ \underline{y}_i - \hat{\underline{y}}_i\ _2^2$	$\sum_{\substack{1 \leq i \leq n \\ 1 \leq k_i \leq K_i}} \ \underline{y}_i^{k_i} - \hat{\underline{y}}_i^{k_i}\ _2^2$

## Modèle de substitution **récurif** $\mathbb{M}$



## Contraintes en prédiction :

- ▶ Conditions initiales :  $(\underline{u}^0, \underline{y}^0)$
- ▶ À  $t_k$ ,  $\hat{\underline{y}}^k$  connaît seulement :
  - ▶  $\hat{\underline{y}}^{k-1}, \dots, \hat{\underline{y}}^1, \underline{y}^0$  et
  - ▶  $\underline{u}^k, \dots, \underline{u}^0$ .
- ▶ À  $t_k$ ,  $\underline{e}^{k-1} = \underline{y}^{k-1} - \hat{\underline{y}}^{k-1}$   
**inconnue** : recalage impossible

## Approche par série temporelle

- ▶ Lennart Ljung, *System identification (2nd ed.) : theory for the user*, Prentice Hall PTR, 1999
- ▶ Olivier Nerrand et al., *Training Recurrent Neural Network : Why and How ? An Illustration in Dynamical Process Modeling*, IEEE Transactions on Neural Networks, 1994

Cas linéaire :

$$\underline{\hat{y}}^k = \sum_{i=1}^{\alpha} A_i \cdot \underline{\hat{y}}^{k-i} + \sum_{i=1}^{\beta+1} B_i \cdot \underline{u}^{k-i+1} + \underline{\varepsilon}^k \text{ avec } \underline{\hat{y}}^0 := \underline{y}^0$$

Cas non linéaire :

$$\underline{\hat{y}}^k = f(\underline{\hat{y}}^{k-1}, \dots, \underline{\hat{y}}^{k-\alpha}, \underline{u}^k, \dots, \underline{u}^{k-\beta}; \underline{w}) + \underline{\varepsilon}^k \text{ avec } \underline{\hat{y}}^0 := \underline{y}^0$$

## En pratique

- ▶ À  $t_1$ , on ne connaît du modèle de référence que  $\underline{y}^0$  et  $\underline{u}^0, \underline{u}^1$ .
- ▶ Problème pour utiliser des données avec des pas de temps différents : les considérer comme une entrée ?

## Approche par série temporelle

- ▶ Lennart Ljung, *System identification (2nd ed.) : theory for the user*, Prentice Hall PTR, 1999
- ▶ Olivier Nerrand *et al.*, *Training Recurrent Neural Network : Why and How ? An Illustration in Dynamical Process Modeling*, IEEE Transactions on Neural Networks, 1994

Cas linéaire :

$$\underline{\hat{y}}^k = A_1 \cdot \underline{\hat{y}}^{k-1} + B_1 \cdot \underline{u}^k + B_2 \cdot \underline{u}^{k-1} + \underline{\varepsilon}^k \text{ avec } \underline{\hat{y}}^0 := \underline{y}^0$$

Cas non linéaire :

$$\underline{\hat{y}}^k = f(\underline{\hat{y}}^{k-1}, \underline{u}^k, \underline{u}^{k-1}; \underline{w}) + \underline{\varepsilon}^k \text{ avec } \underline{\hat{y}}^0 := \underline{y}^0$$

## En pratique

- ▶ À  $t_1$ , on ne connaît du modèle de référence que  $\underline{y}^0$  et  $\underline{u}^0, \underline{u}^1$ .
- ▶ Problème pour utiliser des données avec des pas de temps différents : les considérer comme une entrée ?

# Une alternative aux séries temporelles

## La modélisation de la variation instantanée

$$\frac{d\underline{\hat{y}}(t)}{dt} = \hat{f}(\underline{u}(t), \underline{\hat{y}}(t); \underline{w}) \text{ où } \hat{f} \text{ non linéaire en les entrées et les paramètres } \underline{w}$$

+ un intégrateur numérique (Euler, Runge-Kutta 4,...)

$$\text{Ex : } \underline{\hat{y}}^k = \underline{\hat{y}}^{k-1} + \Delta_k \cdot f(\underline{u}^{k-1}, \underline{\hat{y}}^{k-1}; \underline{w}), \Delta_k = t_k - t_{k-1} \text{ et } \underline{\hat{y}}(t_0) := \underline{y}^0$$

- ▶ Yi-Jen Wang and Chin-Teng Lin, *Runge-Kutta Neural Network for Identification of Dynamical Systems in High Accuracy*, IEEE Transactions on neural networks, 1998
- ▶ R.P.M. Marin, P.M. Tasinaffo, E.D. Yano, *Uma análise comparativa entre metodologias de estrutura de integração neural aplicados a sistemas dinâmicos não-lineares*, XVIII Congresso Brasileiro de Automática, 2010

## Modèle proposé

On s'inspire du modèle précédent en supposant que la dynamique s'écrit :

$$\frac{d\underline{y}(t)}{dt} = f(\underline{u}(t), \underline{y}(t))$$

que l'on injecte dans un développement limité :

$$\begin{aligned} \underline{y}(t_k) &= \underline{y}(t_{k-1}) + \Delta_k \left. \frac{d\underline{y}(t)}{dt} \right|_{t=t_{k-1}} + \frac{1}{2} \left. \frac{d^2\underline{y}(t)}{dt^2} \right|_{t=\xi_k} \Delta_k^2, \quad \xi_k \in ]t_k, t_{k-1}[ \\ &= \underline{y}(t_{k-1}) + \Delta_k \left( f(\underline{u}(t_{k-1}), \underline{y}(t_{k-1})) + \frac{1}{2} \left. \frac{d^2\underline{y}(t)}{dt^2} \right|_{t=\xi_k} \Delta_k \right) \end{aligned}$$

En approchant le terme non linéaire “**dérivée + reste**” par un modèle de substitution  $\hat{f}$ , on obtient le modèle de substitution dynamique :

$$\begin{cases} \underline{\hat{y}}^k &= \underline{\hat{y}}^{k-1} + \Delta_k \cdot \hat{f}(\underline{u}^{k-1}, \underline{\hat{y}}^{k-1}, \Delta_k; \underline{w}) \\ \underline{\hat{y}}^0 &= \underline{y}^0 \end{cases}$$

En pratique,  $\hat{f}$  est un réseau de neurones artificiel.

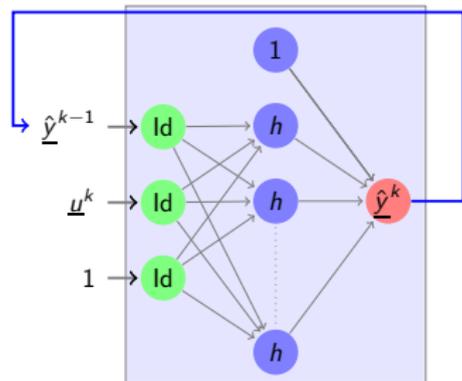
## Approximation par réseau de neurone artificiel

On peut approcher une fonction non linéaire  $f$  par un réseau à  $N$  neurones à sortie multidimensionnelle réelle  $f_{R.N.} = (f_{1,R.N.}, \dots, f_{n_y,R.N.})$  :

$$f_{l,R.N.}(\underline{x}) = \sum_{i=1}^N w_i^{(l)} h \left( \sum_{j=1}^{n_x} w_{ij} x_j + w_{i0} \right) + w_0^{(l)}$$

$$h(s) = \frac{2}{1 + \exp(-2s)} - 1$$

- ▶ Approximateur universel [Hornik *et al.* 1989]
- ▶ Modèle parcimonieux [Barron 1993]
- ▶  $N(N_x + N_y + 1) + N_y$  paramètres



$$\underline{x} = \begin{pmatrix} \hat{y}_1^{k-1} \\ \vdots \\ \hat{y}_{n_y}^{k-1} \\ u_1^k \\ \vdots \\ u_{n_u}^k \end{pmatrix}$$

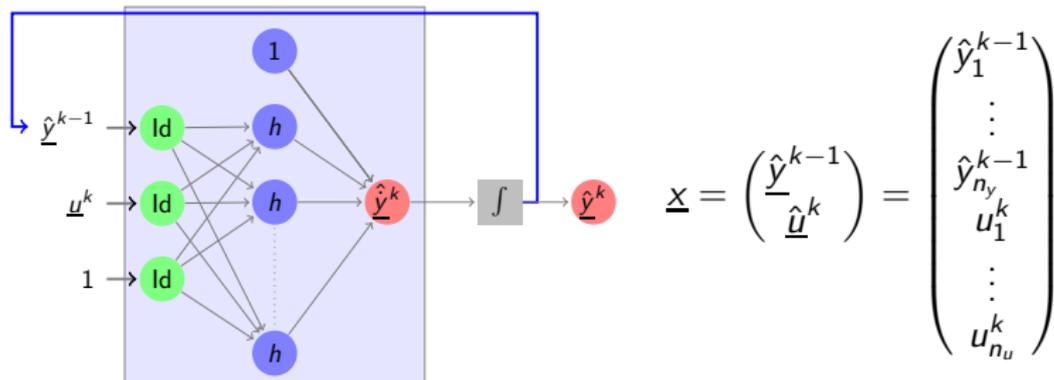
## Approximation par réseau de neurone artificiel

On peut approcher une fonction non linéaire  $f$  par un réseau à  $N$  neurones à sortie multidimensionnelle réelle  $f_{R.N.} = (f_{1,R.N.}, \dots, f_{n_y,R.N.})$  :

$$f_{l,R.N.}(\underline{x}) = \sum_{i=1}^N w_i^{(l)} h \left( \sum_{j=1}^{n_x} w_{ij} x_j + w_{i0} \right) + w_0^{(l)}$$

$$h(s) = \frac{2}{1 + \exp(-2s)} - 1$$

- ▶ Approximateur universel [Hornik *et al.* 1989]
- ▶ Modèle parcimonieux [Barron 1993]
- ▶  $N(N_x + N_y + 1) + N_y$  paramètres



## Récurtivité des écritures

### Le modèle

$$\underline{\hat{y}}^k = \underline{\hat{y}}^{k-1} + \Delta_k \cdot \hat{f}(\underline{u}^{k-1}, \underline{\hat{y}}^{k-1}, \Delta_k; \underline{w}) = \underline{y}^0 + \sum_{t=1}^k \Delta_t \cdot \hat{f}(\underline{u}^{k-t}, \underline{\hat{y}}^{k-t}, \Delta_t; \underline{w})$$

Le gradient de l'erreur d'apprentissage  $e_n = \sum_{i=1}^n \sum_{j=1}^{n_y} \sum_{k=1}^{K_j} e_{k,i,j}$

$$e_{k,i,j} = (\hat{y}_{j,i}^k - y_{j,i}^k)^2 = (\hat{y}_{j,i}^{k-1} + \Delta_k \cdot \hat{f}(\underline{u}_i^{k-1}, \underline{\hat{y}}_i^{k-1}, \Delta_k; \underline{w}) - y_{j,i}^k)^2$$

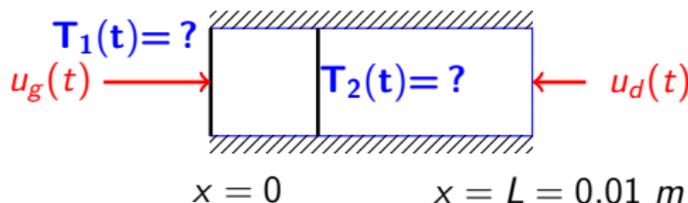
$$\Rightarrow \frac{de_{k,i,j}}{dw} \text{ fonction de } \frac{d\hat{y}_{j,i}^k}{dw} \text{ avec } \underline{\hat{y}}_i^{k-1} \text{ fixé}$$

mais aussi fonction de  $\frac{d\underline{\hat{y}}_i^{k-1}}{dw}$  et donc aussi fonction de  $\frac{d\underline{\hat{y}}_i^{k-2}}{dw}$  etc etc...



**Coût calculatoire accru pour le problème d'optimisation !**

## Cas simplifié : l'équation de la chaleur en 1D (I)



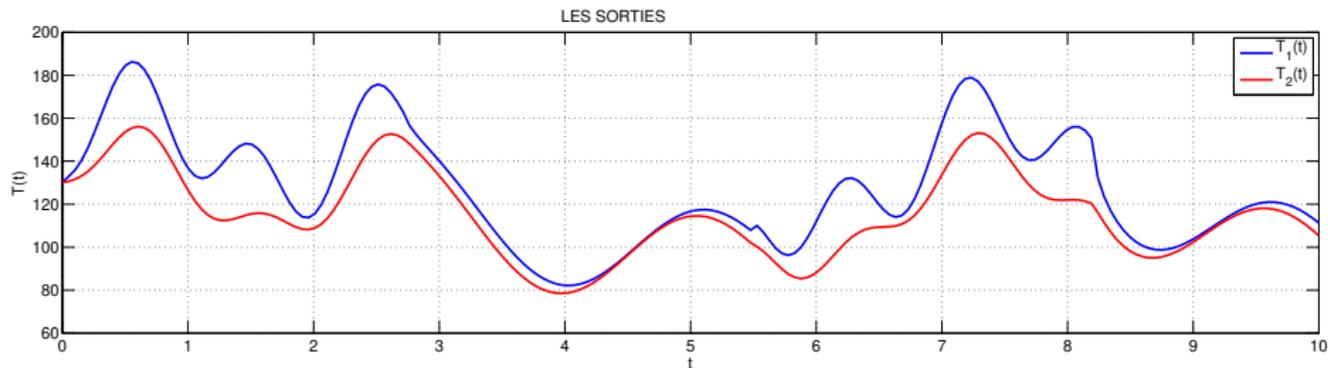
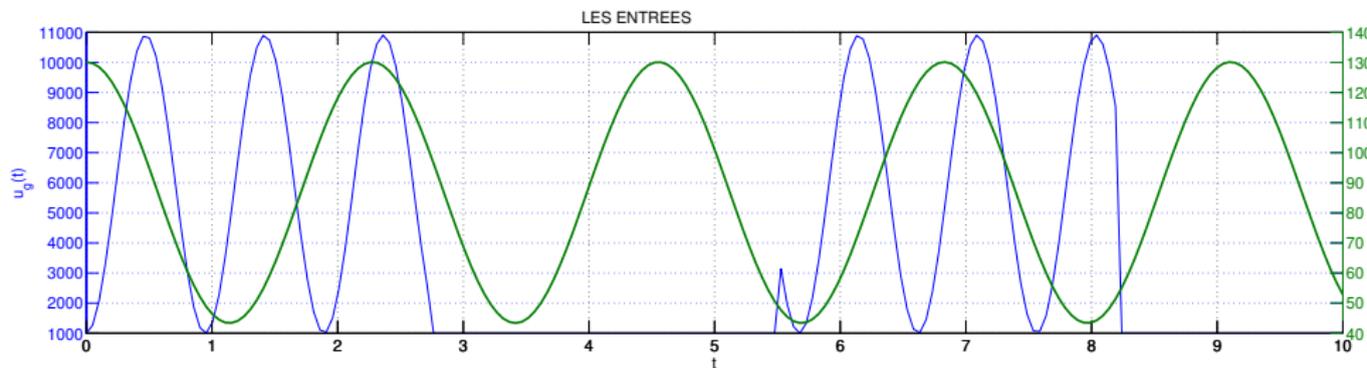
$$\frac{\partial T(x, t)}{\partial t} - \alpha(x, T(x, t)) \frac{\partial^2 T(x, t)}{\partial x^2} = 0, \quad \forall (x, t) \in ]0, L[ \times ]0, \tau[$$

- ▶  $\tau = 10 \text{ s}$  ;
- ▶  $\alpha = 0.00001(1 - 0.9x/L) \text{ [m}^2 \cdot \text{s}^{-1}\text{]}$  la diffusivité thermique du matériau ;
- ▶ C.I. :  $\forall x \in [0, L], T(x, t)|_{t=0} = u_x(x) = u_x$  ;
- ▶ C.L. :  $\forall t \in [0, \tau], \frac{\partial T(x, t)}{\partial x} \Big|_{x=0} = u_g(t)$  ;
- ▶ C.L. :  $\forall t \in [0, \tau], T(x, t)|_{x=L} = u_d(t)$ .

### Problème

Modéliser les sorties  $T_1(t)$  et  $T_2(t)$ ,  $t \in [0, \tau]$ , en fonction des entrées  $u_g(t)$ ,  $u_d(t)$  et  $u_x$ .

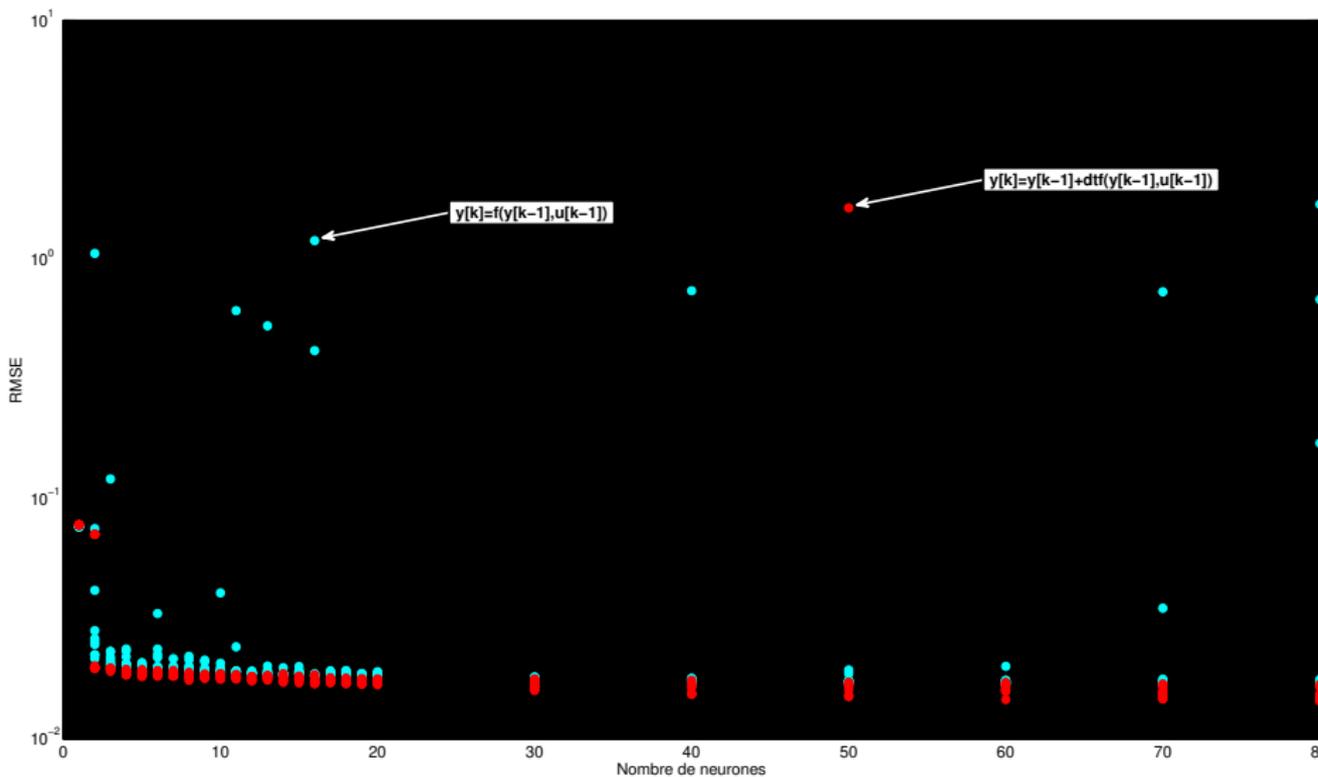
# Cas simplifié : l'équation de la chaleur en 1D (II)



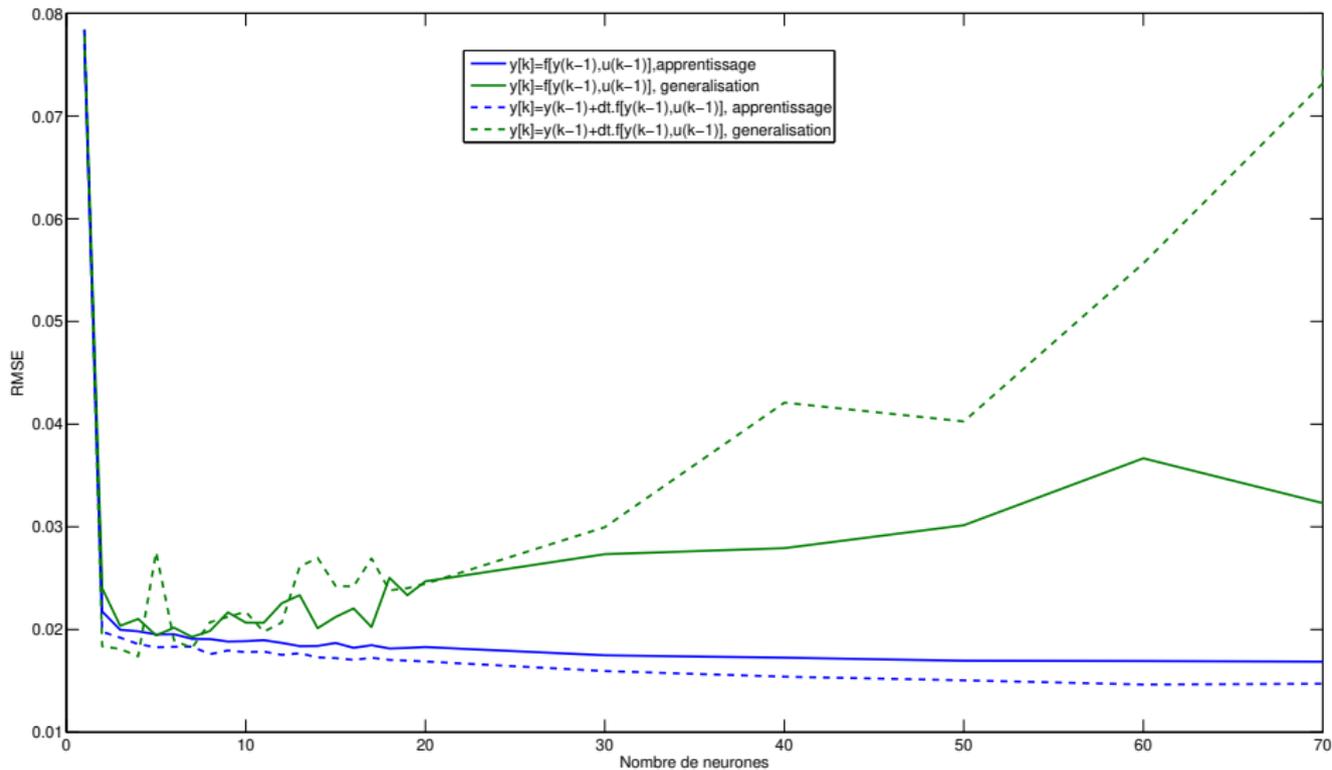
## Cas simplifié : l'équation de la chaleur en 1D (III)

- ▶ Équation de la chaleur résolue par différences finies pour l'espace et Euler implicite pour le temps :
  - ▶ sous Matlab R2007a
  - ▶ 200 points équiésespacés dans le temps
- ▶ Algorithme "Levenberg-Marquardt"  
$$\rightarrow \underline{w}^{(iter)} = \underline{w}^{(iter-1)} - (\nabla_{\underline{w}} e_n^{(iter-1)} \nabla_{\underline{w}} e_n^{(iter-1)T} + \mu \mathbf{I})^{-1} \nabla_{\underline{w}} e_n^{(iter-1)}$$
- ▶ 4 entrées (entrées  $u_g$  et  $u_d$  + sorties  $T_1$  et  $T_2$ )
- ▶ 2 sorties ( $T_1$  et  $T_2$ )
- ▶ 10 couples entrées-sorties en apprentissage
- ▶ 1000 couples entrées-sortie en "généralisation"
- ▶ Construction des modèles de substitution en Fortran 90

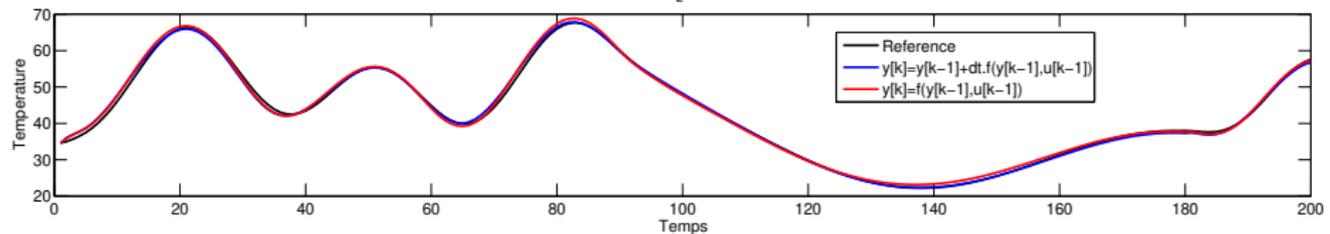
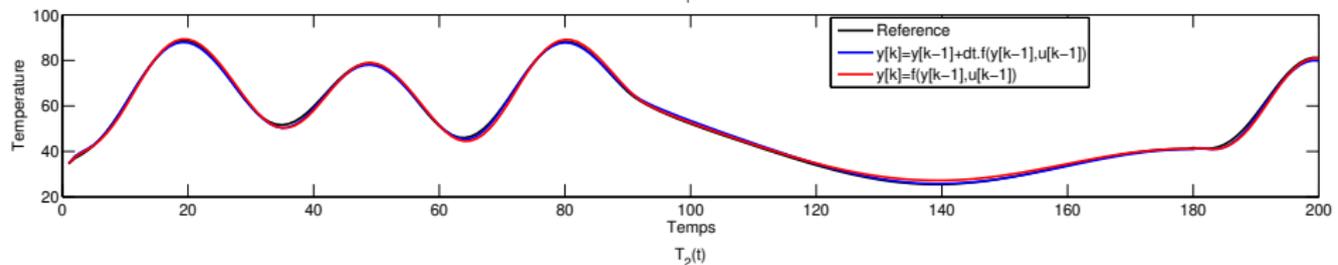
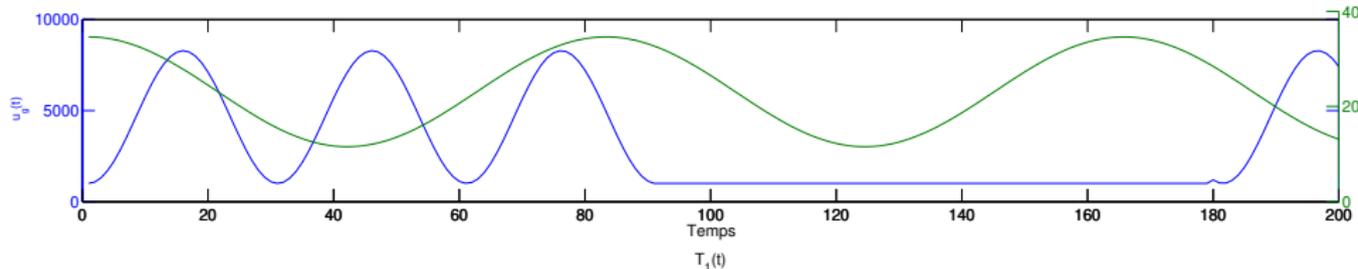
# Résultats : erreur d'apprentissage avec 10 initialisations $\neq$



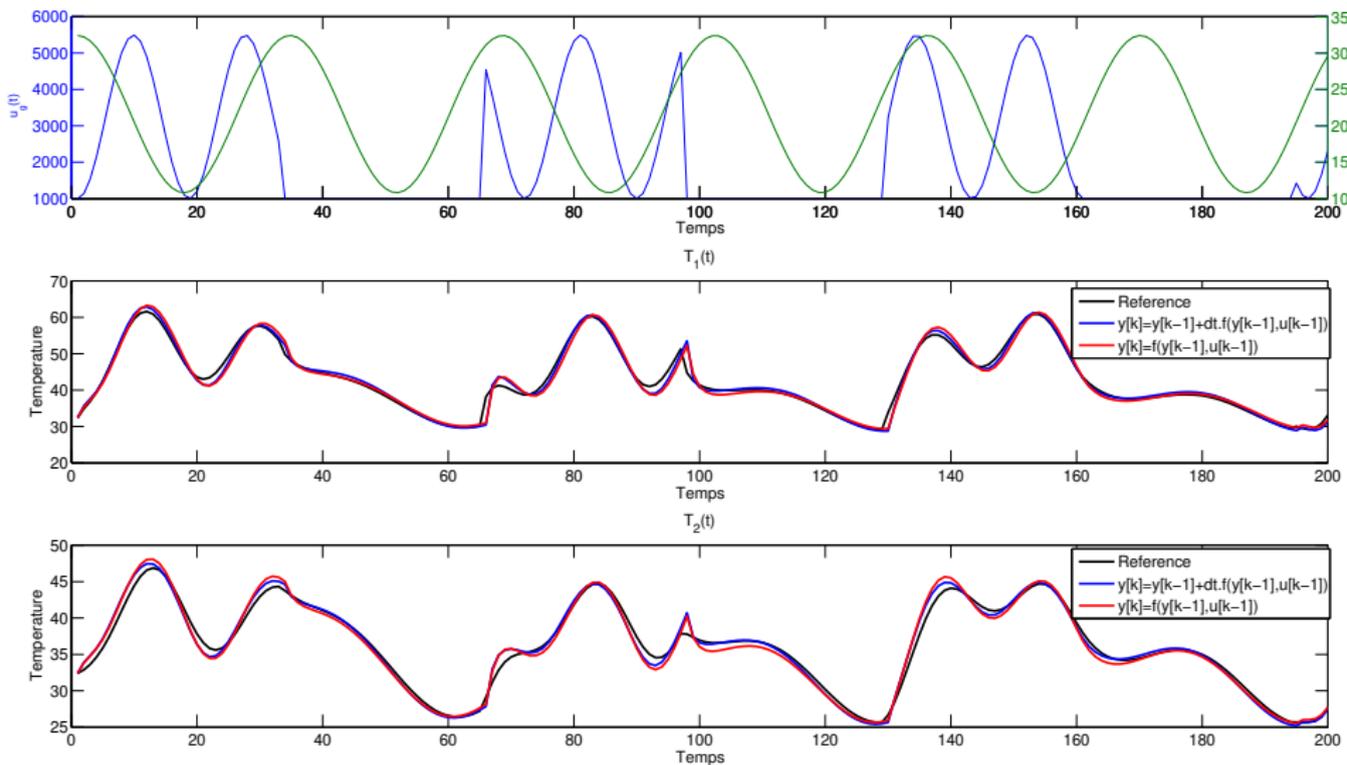
# Résultats : erreur de généralisation



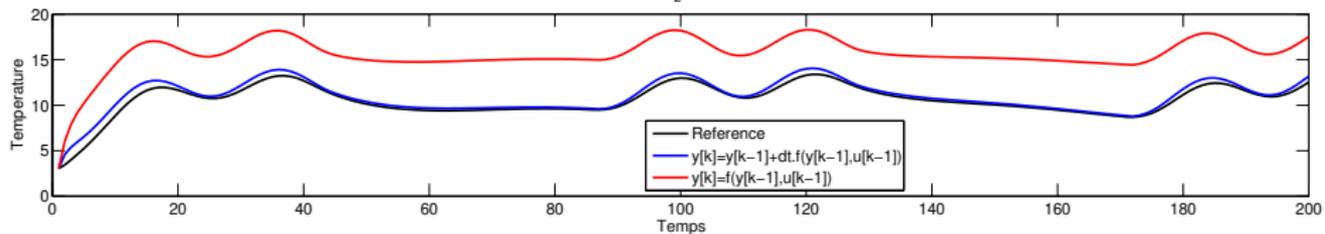
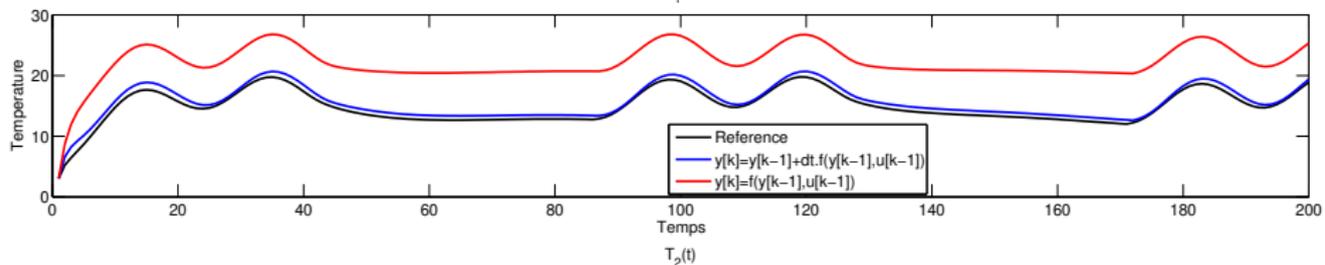
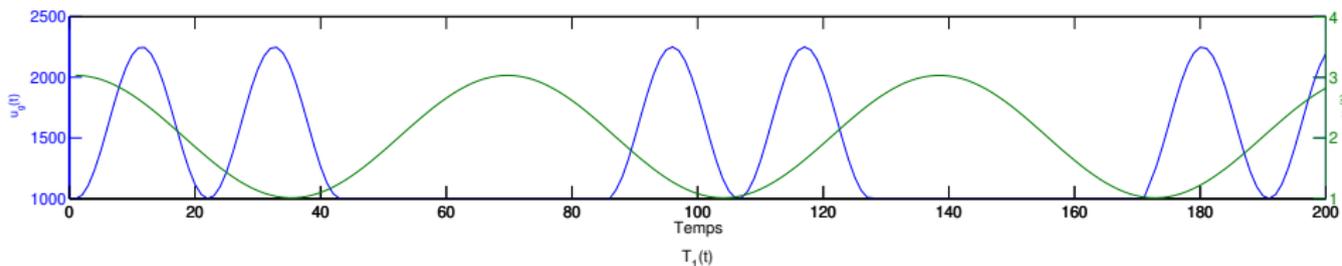
# Résultats : généralisation I



# Résultats : généralisation II



# Résultats : généralisation III



# Plan

## Formulation du problème

## Modèle de substitution proposé

Travaux existants

Modélisation d'une dynamique d'ordre 1

Résultats numériques

## Sélection de modèles

Cadre de la régression homoscédastique

Sélection de modèles non asymptotique

Résultats numériques

## Présentation du problème

Soit un  $n$ -échantillon  $\mathcal{A} = ((X_1, Y_1), \dots, (X_n, Y_n)) \in \Xi^n$  où  $\Xi \subset \mathbb{R}^p \times \mathbb{R}$  et  $(X_i, Y_i)$  i.i.d. de loi  $P$ .

On considère le modèle de régression :

$$\mathbf{Y}_i = \mathbf{s}(\mathbf{X}_i) + \varepsilon_i, \quad \mathbf{i} \in \{\mathbf{1}, \dots, \mathbf{n}\}$$

où  $\varepsilon_1, \dots, \varepsilon_n$  sont i.i.d. tels que  $\mathbb{E}[\varepsilon_i/X_i] = 0$  et  $\mathbb{E}[\varepsilon_i^2/X_i] = \sigma^2$ .

**On veut estimer  $s$  au moyen du  $n$ -échantillon et d'une collection de modèles  $(S_m)_{m \in \mathcal{M}}$  de  $\mathcal{S}$  à définir, où  $\mathcal{M} = \{1, \dots, \#\mathcal{M}\}$ .**

Ici,  $S_m$  est l'ensemble des réseaux à  $N_m$  neurones,  $N_m : \mathcal{M} \mapsto \mathbb{N}^*$  :

$$S_m = \left\{ f(x; \underline{w}) = \sum_{i=1}^{N_m} w_i h \left( \sum_{j=1}^p w_{ij} x_j + w_{i0} \right) + w_0 \right\}, \quad h(z) = \frac{2}{1 + e^{-2z}} - 1$$

et  $\mathcal{S}$  l'ensemble des réseaux de neurones.

## Estimation de $s$ dans $S_m$

- ▶ Le **contraste des moindres carrés**  $\gamma : \mathcal{S} \times \Xi \rightarrow \mathbb{R}_+$

$$\gamma(t, (x, y)) = (y - t(x))^2 \text{ en régression homoscedastique}$$

- ▶  $\mathcal{R}(t) = \int_{\Xi} (y - t(x))^2 dP(x, y)$

**On cherche un  $s_m$  dans  $S_m$  qui minimise  $\mathcal{R}(t)$ .**

**$\Rightarrow$  Nécessité de connaître  $P$  !**

Alors...

...on prend un  $\hat{s}_m$  dans un  $S_m$  qui minimise  $\gamma_n(t, \mathcal{A})$  :

$$\gamma_n(t, \mathcal{A}) = \frac{1}{n} \sum_{i=1}^n (y_i - t(x_i))^2$$

## Boîte à outils classique en sélection de modèles

Pour garantir une bonne erreur de généralisation, on choisit un modèle  $S_m$  "pas trop complexe" à l'aide de :

- ▶ la **validation croisée  $K$ -folds** (courant en réseau de neurones);
- ▶ une approche de type **hold-out** : choisir  $S_m$  dans lequel l'estimateur  $\hat{S}_m$  est d'erreur minimale sur une base de test.
- ▶ de **critères de sélection** de type :
  - ▶ **AIC** =  $-2 \ln \mathcal{L}(\underline{w}) + 2\#\underline{w}$ ;
  - ▶ **BIC** =  $-2 \ln \mathcal{L}(\underline{w}) + 2\#\underline{w} \log(n)$ .

### Autre approche

**Utiliser une méthode non asymptotique et peu gourmande.**

→ Quid de l'**heuristique de pente** et du **saut de dimension** ?

## Sélection de modèles par critère pénalisé

Soit  $(S_m)_{m \in \mathcal{M}}$  une collection de modèles.

▶ On a une collection  $\hat{s}_1, \dots, \hat{s}_{\#\mathcal{M}}$ .

▶ **Modèle idéal**  $S_{m(s)}$  :

$$S_m \text{ t.q. } S_m \ni \hat{s}_{m(s)} = \underset{\hat{s}_1, \dots, \hat{s}_{\#\mathcal{M}}}{\operatorname{argmin}} \|\hat{s}_m - s\|_{2,P}^2$$

$\hat{s}_{m(s)} \neq$  un estimateur de  $s$  car fonction de  $P$  inconnu.

On l'appelle l'**Oracle**.

▶ **But** : Trouver un estimateur  $\hat{s}_{\hat{m}}$  de  $s$  proche de l'Oracle à partir **des**  $\hat{s}_1, \dots, \hat{s}_{\#\mathcal{M}}$ , **de**  $\mathcal{A}$ , **de**  $n$  et **des propriétés des modèles**  $S_1, \dots, S_{\#\mathcal{M}}$ .

▶ **Moyen** : Critère pénalisé assurant un compromis **biais-variance** :

$$\hat{m} = \underset{m \in \mathcal{M}}{\operatorname{argmin}} \underbrace{\{\gamma_n(\hat{s}_m, \mathcal{A}) + \operatorname{pen}(m)\}}_{\operatorname{crit}(m)}$$

## Choix de la pénalité

Minimiser  $\text{crit}(m) = \gamma_n(\hat{s}_m, \mathcal{A}) + \text{pen}(m)$  revient à minimiser

$$\text{crit}(m) = \|\hat{s}_m - s\|_{2,P}^2 + \gamma_n(\hat{s}_m, \mathcal{A}) - \|\hat{s}_m - s\|_{2,P}^2 + \text{pen}(m) \stackrel{\text{Oracle}}{=} \|\hat{s}_m - s\|_{2,P}^2$$

**La pénalité idéale s'écrit alors  $\text{pen}(m) = \|\hat{s}_m - s\|_{2,P}^2 - \gamma_n(\hat{s}_m, \mathcal{A})$  et est fonction de  $P$  inconnue.**

**Hypothèse :**  $\exists$  des résultats théoriques fournissant une pénalité optimale

$$\text{pen}_{\text{opt}}(m) = \kappa_{\text{opt}} \text{pen}_{\text{shape}}(m)$$

où  $\kappa_{\text{opt}}$  est une constante numérique à **déterminer**.

$\Rightarrow \text{pen}_{\text{shape}}(m) \approx \frac{N_m(p+1)}{n}$  dans le cadre des réseaux de neurones [Barron, Birgé et Massart, 1995]

## Calibration de $\kappa_{\text{opt}}$ dans $\text{pen}_{\text{opt}}(m) = \kappa_{\text{opt}} \text{pen}_{\text{shape}}(m)$

Heuristique de pente [Birgé et Massart, 2006] et saut de dimension [Arlot et Massart, 2009] implémentés dans CAPUSHE (Matlab et R) :

- ▶ J.-C. Baudry, C. Maugis et B. Michel, *Slope heuristics : overview and implementation*, Stat Comput, 2012

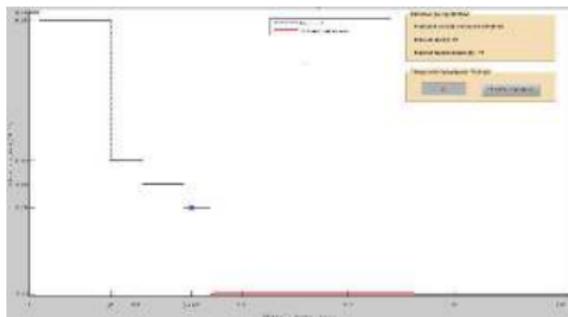
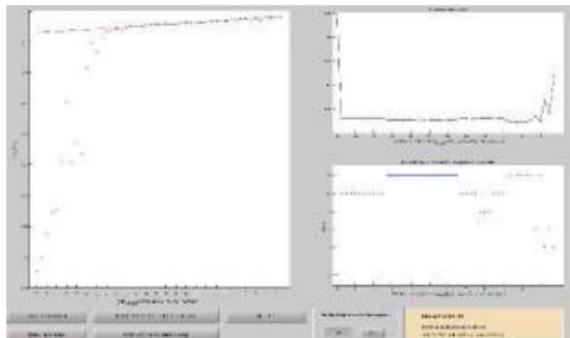
*Data-driven slope estimation (DDSE)*

Pour des modèles complexes, on a :

$$-\gamma_n(\hat{s}_m, \mathcal{A}) \approx \frac{\kappa_{\text{opt}}}{2} \text{pen}_{\text{shape}}(m) + \beta$$

*Dimension Jump (DJUMP)*

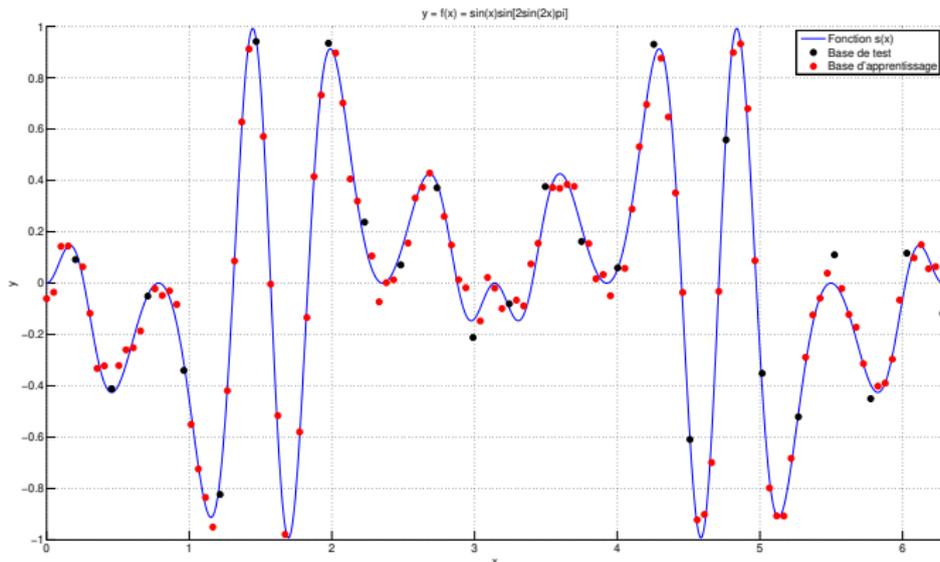
1. chercher  $\kappa$  entraînant le plus grand saut de dimension, de complexité ;
2. prendre  $\kappa_{\text{opt}} = 2\kappa$ .



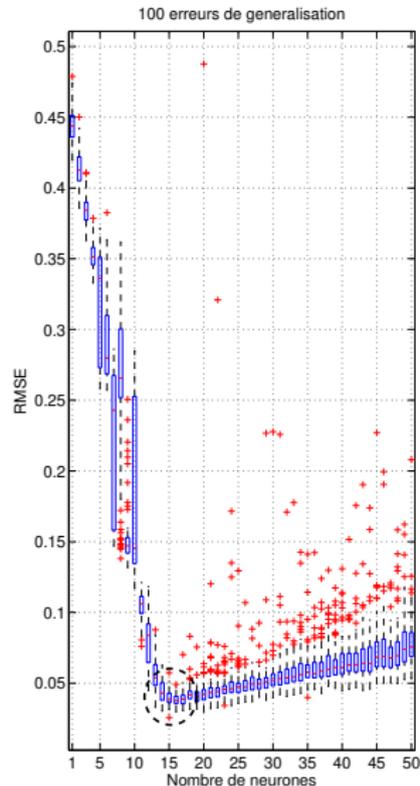
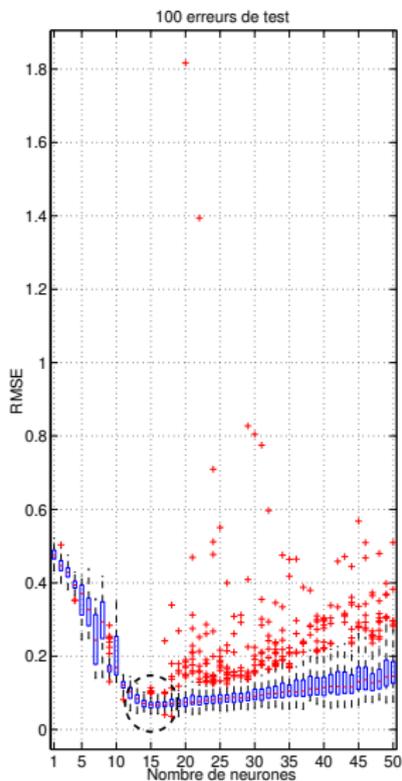
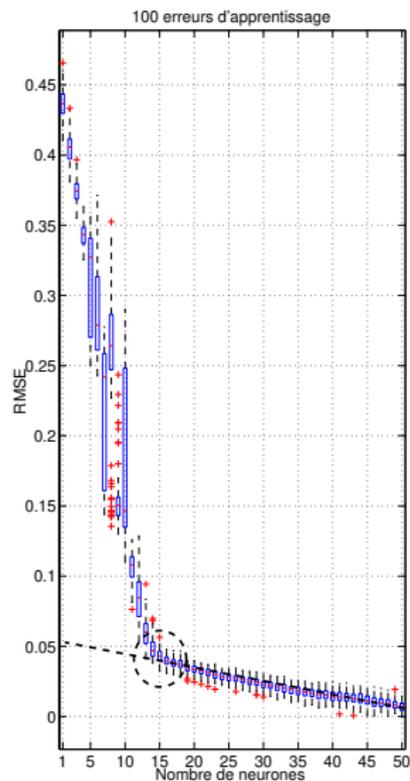
## Cas d'étude

$$Y_i = s(X_i) + \varepsilon_i = \sin(X_i) \sin(2\pi \sin(2X_i)) + \mathcal{N}(0, \sigma^2)$$

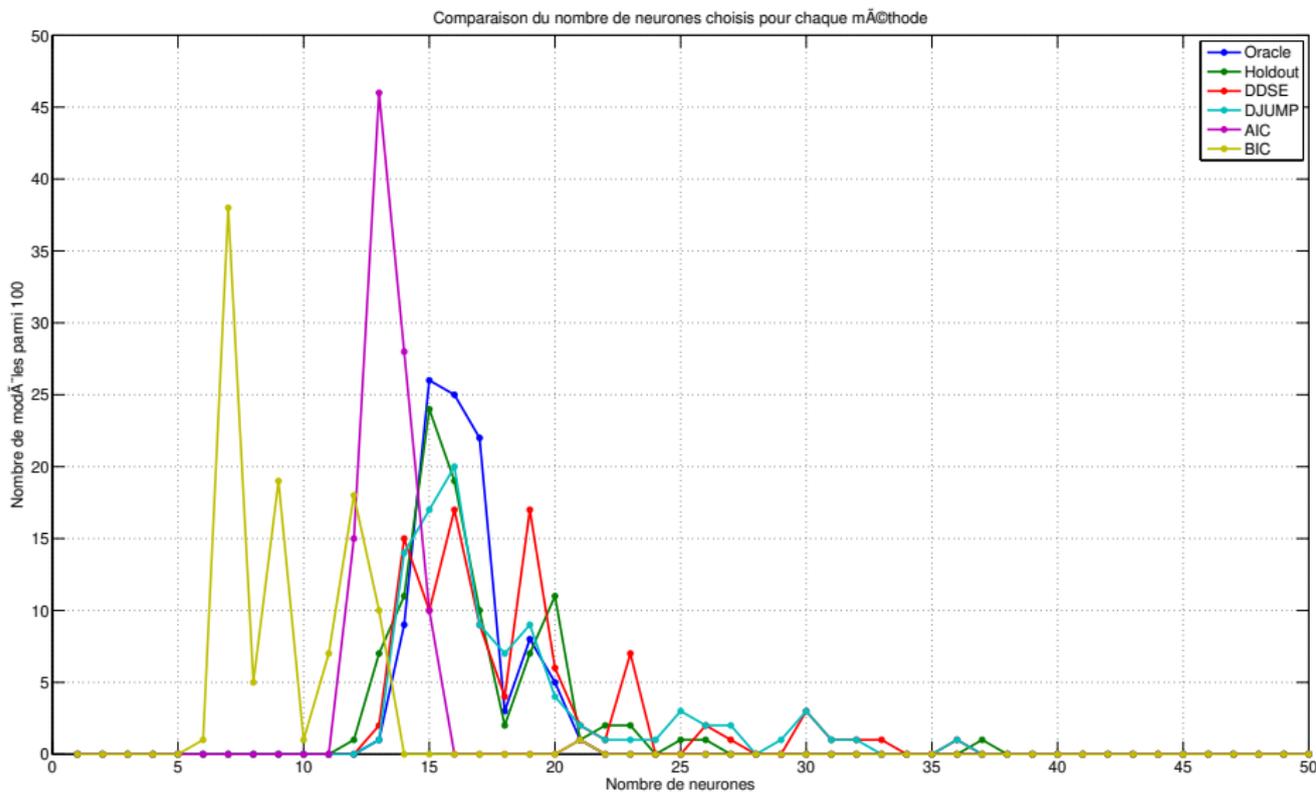
$$X_i = 2(i-1)\pi/(n-1), \quad i \in \{1, \dots, n\} \text{ avec } n = 100, \quad \sigma = 0.05$$



→ Utilisation de 100 jeux de données  $\mathcal{A} = (X_i, Y_i)_{i \in \{1, \dots, n\}}$

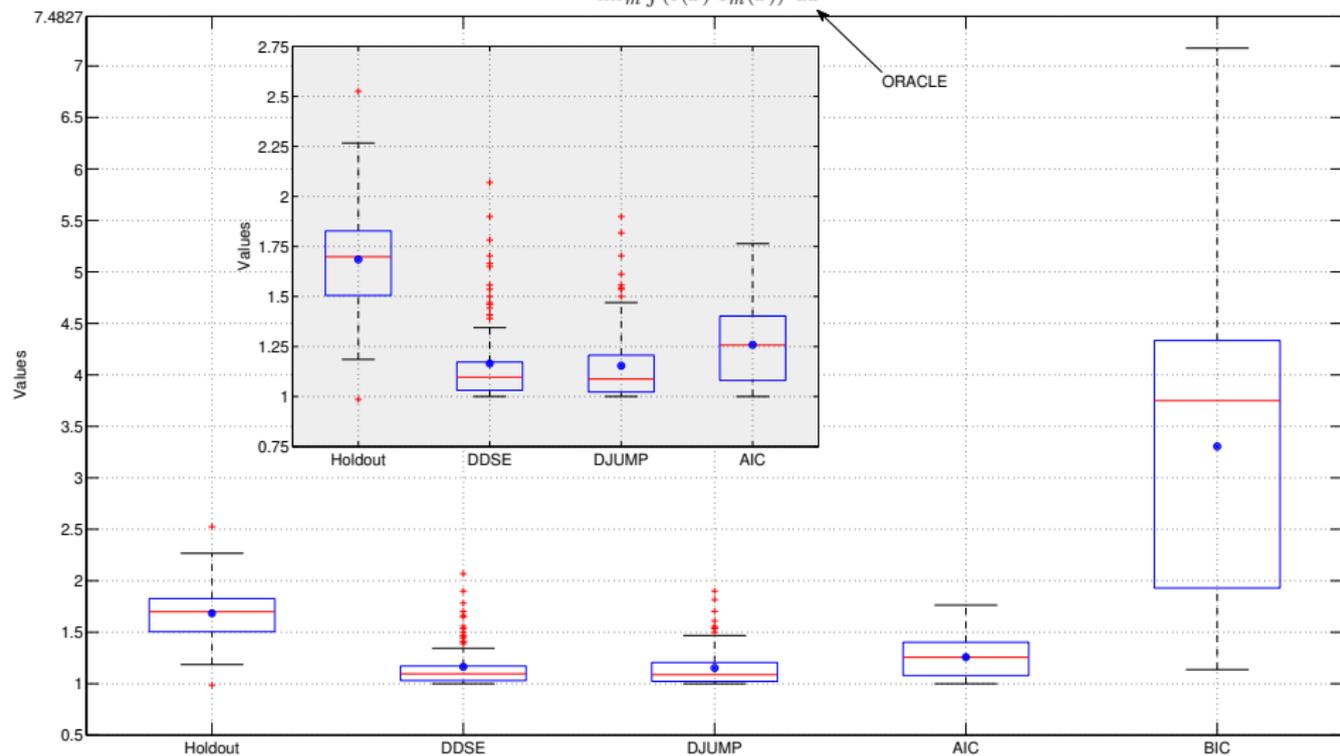
Variabilité de la base d'apprentissage  $\mathcal{A} = (X_i, Y_i)_{i \in \{1, \dots, n\}}$ 

# Fréquence de sélections de modèles pour 100 $n$ -échantillons

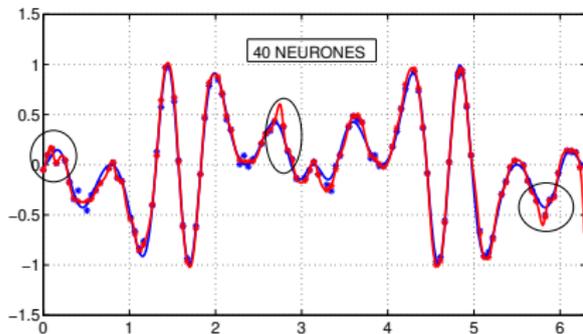
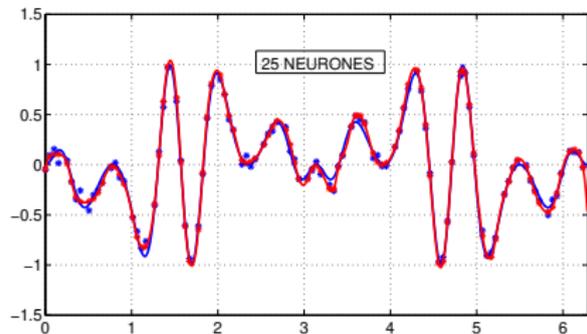
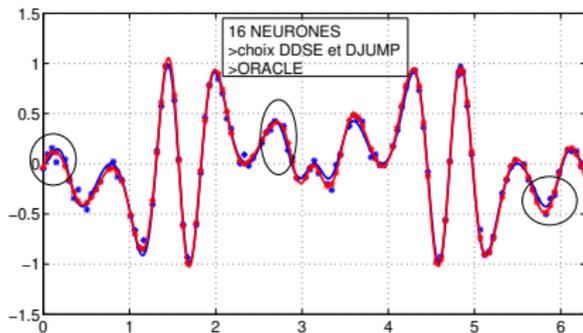
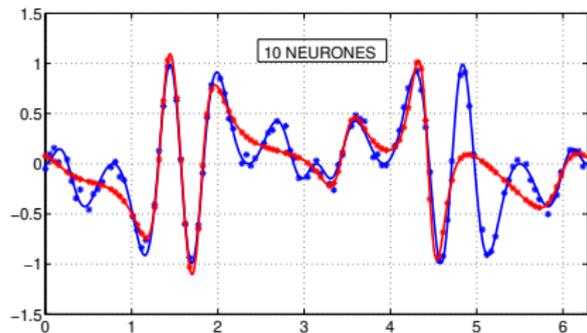


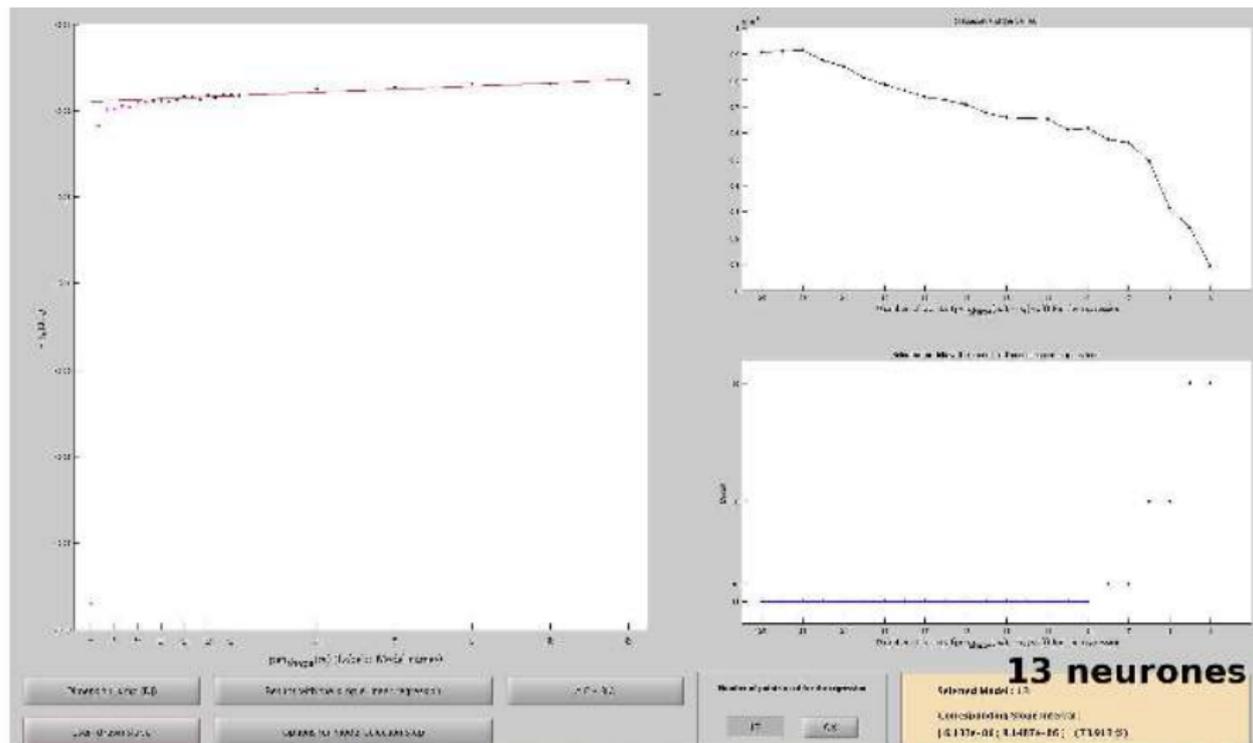
# Distribution des distances à l'Oracle

Distribution de  $\frac{\int (s(x) - \hat{s}_m(x))^2 dx}{\inf_m \int (s(x) - \hat{s}_m(x))^2 dx}$  selon  $(x_1, \dots, x_n)$

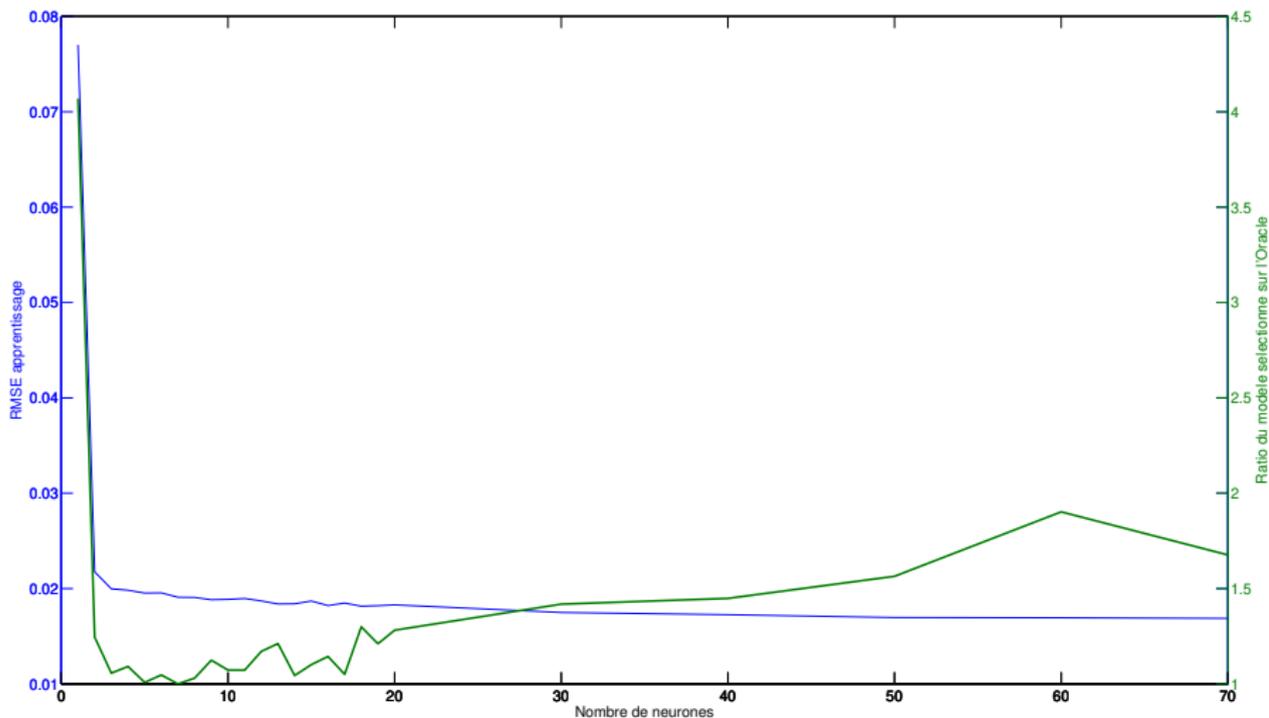


# Qualité de prédiction et distance à l'Oracle

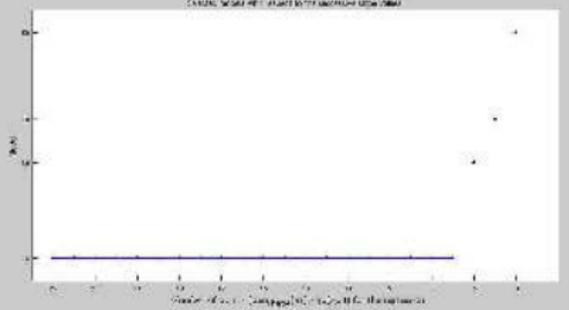
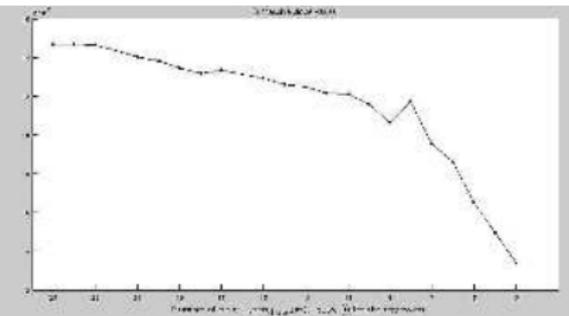
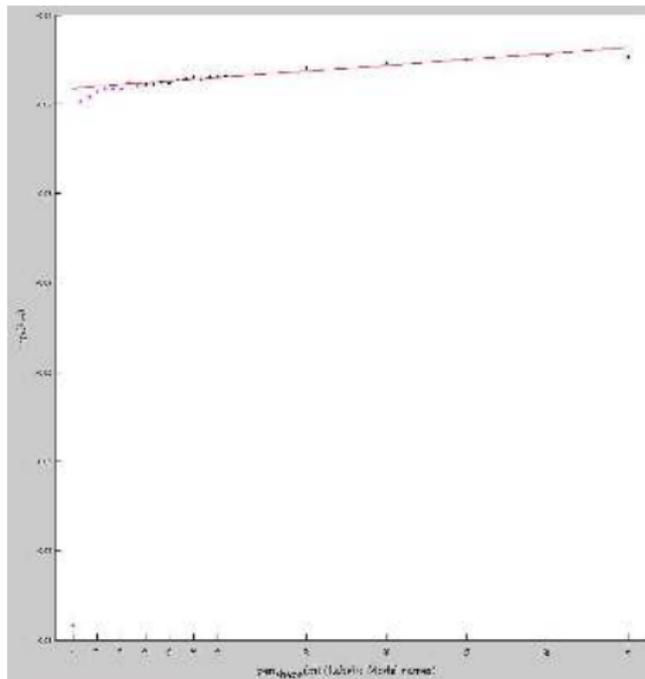


Équation de la chaleur  $\hat{y}_k = f_{\text{R.N.}}(\underline{u}_{k-1}, \hat{y}_k)$ 

# Équation de la chaleur $\hat{y}_l = f_{\text{R.N.}}(\underline{u}_{k-1}, \hat{y}_l)$

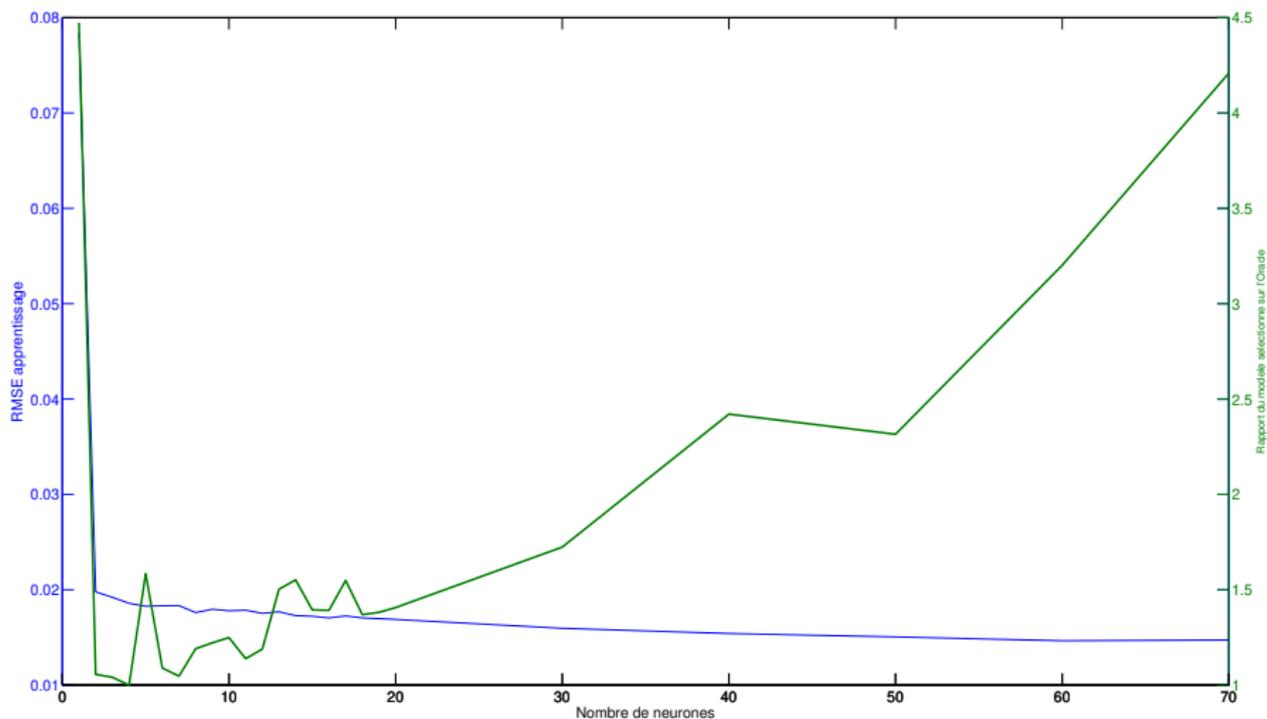


# Équation de la chaleur $\hat{y}_k = \hat{y}_{k-1} + \Delta_k \cdot f_{R.N.}(\underline{u}_{k-1}, \hat{y}_{k-1})$



Détecter Model: **8 neurones**  
 Corresponding slope interval:  
 [1.4048e-06; 1.4998e-05] (38.936%)

# Équation de la chaleur $\hat{y}_\nu = \hat{y}_{\nu-1} + \Delta_k \cdot f_{\text{R.N.}}(\underline{u}_{k-1}, \hat{y}_{\nu-1})$



# Apprentissage de dynamiques spatio-temporelles par réseaux de neurones artificiels

MERCI!



Journées MASCOT NUM 2012 - Mercredi 21 mars 2012