Machine Learning Strategies for Time Series Prediction

Machine Learning Summer School (Hammamet, 2013)

Gianluca Bontempi

Machine Learning Group, Computer Science Department

Boulevard de Triomphe - CP 212

http://www.ulb.ac.be/di

Introducing myself

- 1992: Computer science engineer (Politecnico di Milano, Italy),
- 1994: Researcher in robotics in IRST, Trento, Italy,
- 1995: Researcher in IRIDIA, ULB Artificial Intelligence Lab, Brussels,
- 1996-97: Researcher in IDSIA, Artificial Intelligence Lab, Lugano, Switzerland,
- 1998-2000: Marie Curie fellowship in IRIDIA, ULB Artificial Intelligence Lab, Brussels,
- 2000-2001: Scientist in Philips Research, Eindhoven, The Netherlands,
- 2001-2002: Scientist in IMEC, Microelectronics Institute, Leuven, Belgium,
- since 2002: professor in Machine Learning, Modeling and Simulation, Bioinformatics in ULB Computer Science Dept., Brussels,
- since 2004: head of the ULB Machine Learning Group (MLG).
- since 2013: director of the Interuniversity Institute of Bioinformatics in Brussels (IB)^2, `ibsquare.be`.

... and in terms of distances

According to MathSciNet

- Distance from Erdos= 5
- Distance from Zaiane= 6
- Distance from Deisenroth= 8

ULB Machine Learning Group (MLG)

- 3 professors, 10 PhD students, 5 postdocs.
- Research topics: Knowledge discovery from data, Classification, Computational statistics, Data mining, Regression, Time series prediction, Sensor networks, Bioinformatics, Network inference.
- Computing facilities: high-performing cluster for analysis of massive datasets, Wireless Sensor Lab.
- Website: `mlg.ulb.ac.be`.
- Scientific collaborations in ULB: Hopital Jules Bordet, Laboratoire de Médecine experimentale, Laboratoire d'Anatomie, Biomécanique et Organogénèse (LABO), Service d'Anesthesie (ERASME).
- Scientific collaborations outside ULB: Harvard Dana Farber (US), UCL Machine Learning Group (B), Politecnico di Milano (I), Universitá del Sannio (I), Inst Rech Cliniques Montreal (CAN).

ULB-MLG: recent projects

1. Machine Learning for Question Answering (2013-2014).
2. Adaptive real-time machine learning for credit card fraud detection (2012-2013).
3. Epigenomic and Transcriptomic Analysis of Breast Cancer (2012-2015).
4. Discovery of the molecular pathways regulating pancreatic beta cell dysfunction and apoptosis in diabetes using functional genomics and bioinformatics: ARC (2010-2015)
5. ICT4REHAB - Advanced ICT Platform for Rehabilitation (2011-2013)
6. Integrating experimental and theoretical approaches to decipher the molecular networks of nitrogen utilisation in yeast: ARC (2006-2010).
7. TANIA - Système d'aide à la conduite de l'anesthésie. WALEO II project funded by the Région Wallonne (2006-2010)
8. "COMP^{2}SYS" (COMPutational intelligence methods for COMPlex SYStems) MARIE CURIE Early Stage Research Training funded by the EU (2004-2008).

What you are supposed to know

- Basic notions of probability and statistics
- Random variable
- Expectation, variance, covariance
- Least-squares

What you are expected to get acquainted with

- Foundations of statistical machine learning
- How to build a predictive model from data
- Strategies for forecasting

What will remain

- Interest, curiosity for machine learning
- taste for prediction
- Contacts
- Companion webpage
  `http://www.ulb.ac.be/di/map/gbonte/mlss.html`

Outline

- Notions of time series (**30 mins**)
  - conditional probability
- Machine learning for prediction (**45 mins**)
  - bias/variance
  - parametric and structural identification
  - validation
  - model selection
  - feature selection
- **COFFEE BREAK**
- Local learning (**15 mins**)
- Forecasting: one-step and multi-step-ahed (**30 mins**)
- Some applications (**15 mins**)
  - time series competitions
  - wireless sensor
  - biomedical
- Future directions and perspectives (**15 mins**)

What is machine learning?

Machine learning is that domain of computational intelligence which is concerned with the question of how to construct computer programs that automatically improve with experience. [16]

Reductionist attitude: *ML is just a buzzword which equates to statistics plus marketing*

Positive attitude: ML paved the way to the treatment of real problems related to data analysis, sometimes overlooked by statisticians (nonlinearity, classification, pattern recognition, missing variables, adaptivity, optimization, massive datasets, data management, causality, representation of knowledge, parallelisation)

Interdisciplinary attitude: ML *should* have its roots on statistics and complements it by focusing on: algorithmic issues, computational efficiency, data engineering.

Why study machine learning?

- Machine learning is cool.
- Practical way to understand: *All models are wrong but some are useful...*
- The fastest way to become a data scientist ... the sexiest job in the 21st century
- *Someone who knows statistics better than a computer scientists and programs better than a statistician...*

Notion of time series

Time series

Definition A time series is a sequence of observations $s_t \in \mathbb{R}$, usually ordered in time.

Examples of time series in every scientific and applied domain:

- Meteorology: weather variables, like temperature, pressure, wind.
- Economy and finance: economic factors (GNP), financial indexes, exchange rate, spread.
- Marketing: activity of business, sales.
- Industry: electric load, power consumption, voltage, sensors.
- Biomedicine: physiological signals (EEG), heart-rate, patient temperature.
- Web: clicks, logs.
- Genomics: time series of gene expression during cell cycle.

Why studying time series?

There are various reasons:

Prediction of the future based on the past.

Control of the process producing the series.

Understanding of the mechanism generating the series.

Description of the salient features of the series.

Univariate discrete time series

- Quantities, like temperature and voltage, change in a continuous way.
- In practice, however, the digital recording is made discretely in time.
- We shall confine ourselves to **discrete** time series (which however take continuous values).
- Moreover we will consider **univariate** time series, where one type of measurement is made repeatedly on the same object or individual.
- Multivariate time series are out of the scope of this presentation but represent an important topic in the domain.

A general model

Let an observed discrete univariate time series be $s_1, \ldots, s_T$. This means that we have T numbers which are observations on some variable made at T equally distant time points, which for convenience we label $1, 2, \ldots, T$.

A fairly general model for the time series can be written

$$s_t = g(t) + \varphi_t \qquad t = 1, \ldots, T$$

The observed series is made of two components

Systematic part: $g(t)$, also called *signal* or *trend*, which is a determistic function of time

Stochastic sequence: a residual term φ_t, also called *noise*, which follows a probability law.

Types of variation

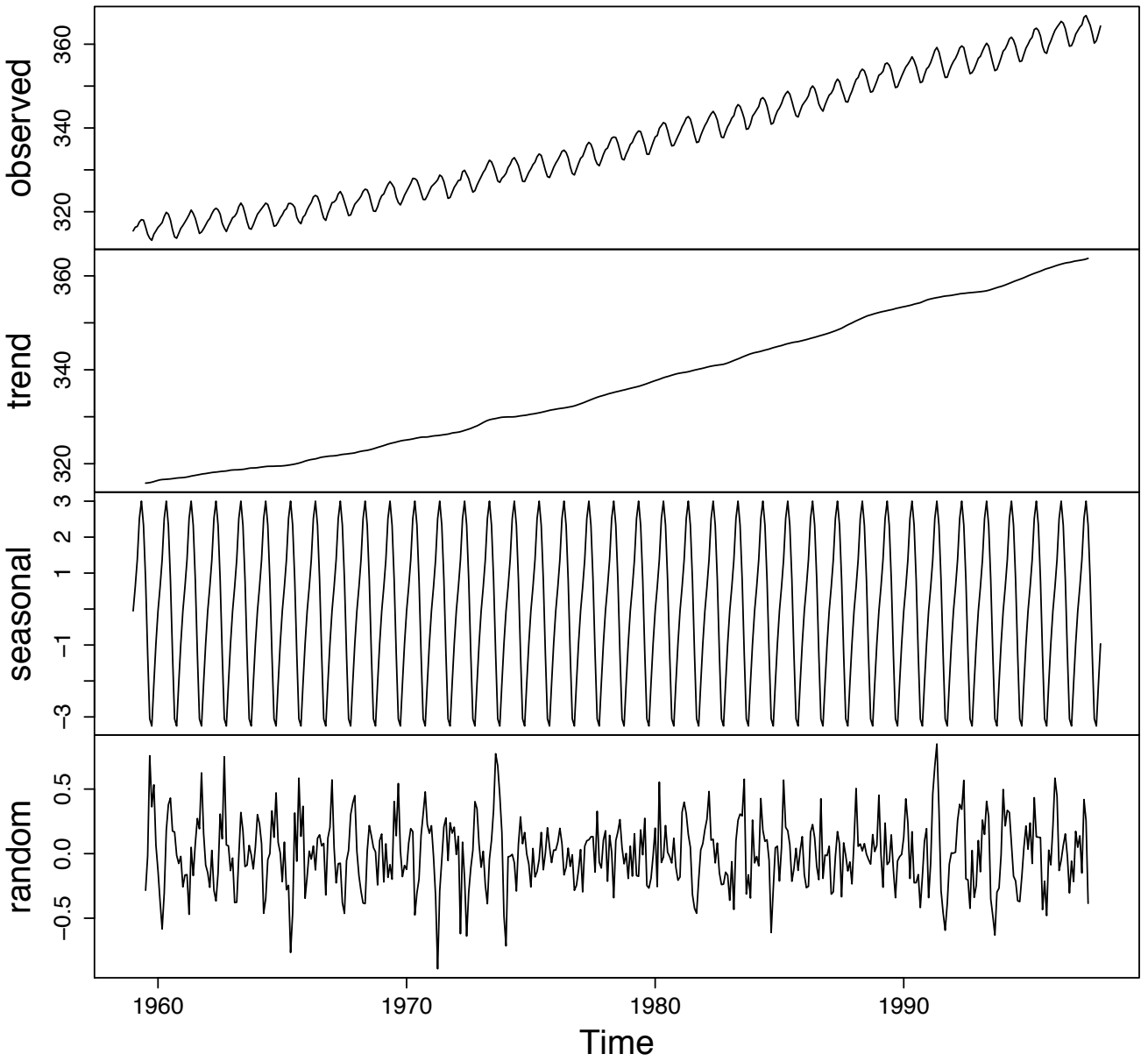
Traditional methods of time-series analysis are mainly concerned with decomposing the variation of a series s_t into:

Trend : this is a long-term change in the mean level, eg. an increasing trend.

Seasonal effect : many time series (sale figures, temperature readings) exhibit variation which is seasonal (e.g. annual) in period. The measure and the removal of such variation brings to *deseasonalized* data.

Irregular fluctuations : after trend and cyclic variations have been removed from a set of data, we are left with a series of residuals, which may or may not be *completely random*.

We will assume here that once we have detrended and deseasonalized the series, we can still extract information about the dependency between the past and the future. Henceforth φ_t will denote the detrended and deseasonalized series.



Decomposition returned by the R package `forecast`.

Probability and dependency

- Forecasting a time series is possible since **future depends on the past** or analogously because there is a relationship between the future and the past. However this relation is not deterministic and can hardly be written in an analytical form.
- An effective way to describe a nondeterministic relation between two variables is provided by the probability formalism.
- Consider two continuous random variables φ_1 and φ_2 representing for instance the temperature today (time t_1) and tomorrow (t_2). We tend to believe that φ_1 could be used as a predictor of φ_2 with some degree of uncertainty.
- The stochastic dependency between φ_1 and φ_2 is resumed by the joint density $p(\varphi_1, \varphi_2)$ or equivalently by the **conditional probability**

$$p(\varphi_2|\varphi_1) = \frac{p(\varphi_1, \varphi_2)}{p(\varphi_1)}$$

- If $p(\varphi_2|\varphi_1) \neq p(\varphi_2)$ then φ_1 and φ_2 are not independent or equivalently the knowledge of the value of φ_1 reduces the uncertainty about φ_2.

Stochastic processes

- The stochastic approach to time series makes the assumption that a time series is a realization of a stochastic process (like tossing an unbiased coin is the realization of a discrete random variable with equal head/tail probability).
- A discrete-time **stochastic process** is a collection of random variables φ_t, $t = 1, \ldots, T$ defined by a joint density

$$p(\varphi_1, \ldots, \varphi_T)$$

- Statistical *time-series analysis* is concerned with evaluating the properties of the probability model which generated the observed time series.
- Statistical *time-series modeling* is concerned with **inferring** the properties of the probability model which generated the observed time series **from a limited set of observations**.

Strictly stationary processes

- Predicting a time series is possible if and only if the dependence between values existing in the past is preserved also in the future.
- In other terms, though measures change, the stochastic rule underlying their realization does not. This aspect is formalized by the notion of stationarity.
- **Definition** A stochastic process is said to be **strictly stationary** if the joint distribution of $\varphi_{t_1}, \varphi_{t_2}, \ldots, \varphi_{t_n}$ is the same as the joint distribution of $\varphi_{t_1+k}, \varphi_{t_2+k}, \ldots, \varphi_{t_n+k}$ for all n, $t_1, \ldots, t_n$ and k.
- In other words shifting the time origin by an amount k has no effect on the joint distribution which depends only on the intervals between $t_1, \ldots, t_n$.
- This implies that the (marginal) distribution of φ_t is the same for all t.
- The definition holds for any value of n.
- Let us see what does it mean in practice for $n = 1$ and $n = 2$.

Properties

n=1 : If φ_t is strictly stationary and its first two moments are finite, we have

$$E[\varphi_t] = \mu_t = \mu, \qquad \text{Var}\,[\varphi_t] = \sigma_t^2 = \sigma^2$$

n=2 : Furthermore the autocovariance function $\gamma(t_1, t_2)$ depends only on the **lag** $k = t_2 - t_1$ and may be written by

$$\gamma(k) = \text{Cov}[\varphi_t, \varphi_{t+k}] = E\left[(\varphi_t - \mu)(\varphi_{t+k} - \mu)\right]$$

- In order to avoid scaling effects, it is useful to introduce the **autocorrelation function**

$$\rho(k) = \frac{\gamma(k)}{\sigma^2} = \frac{\gamma(k)}{\gamma(0)}$$

- Another relevant function is the the **partial autocorrelation function** $\pi(k)$ where $\pi(k), k > 1$ measures the degree of association between φ_t and φ_{t-k} when the effects of the intermediary lags $1, \ldots, k-1$ are removed

Weak stationarity

- A less restricted definition of stationarity concerns only the first two moments of φ_t

  Definition A process is called **second-order stationary** or **weakly stationary** if its mean is constant and its autocovariance function depends only on the lag.

- No assumptions are made about higher moments than those of second order.
- Strict stationarity implies weak stationarity but not viceversa in general.

  Definition A process is called **normal** is the joint distribution of $\varphi_{t_1}, \varphi_{t_2}, \ldots, \varphi_{t_n}$ is multivariate normal for all $t_1, \ldots, t_n$.

- In the special case of normal processes, weak stationarity implies strict stationarity. This is due to the fact that a normal process is completely specified by the mean and the autocovariance function.

Estimators of first moments

Here you will find some common estimators of the two first moments of a time series:

- The empirical mean is given by

$$\hat{\mu} = \frac{\sum_{t=1}^{T} \varphi_t}{T}$$

- The empirical autocovariance function is given by

$$\hat{\gamma}(k) = \frac{\sum_{t=1}^{T-k} (\varphi_t - \hat{\mu})(\varphi_{t+k} - \hat{\mu})}{T-k-1}, \quad k < T-2$$

- The empirical autocorrelation function is given by

$$\hat{\rho}(k) = \frac{\hat{\gamma}(k)}{\hat{\gamma}(0)}$$

Some examples of stochastic processes

Purely random processes

- It consists of a sequence of random variables φ_t which are mutually independent and identically distributed. For each t and k

$$p(\varphi_{t+k}|\varphi_t) = p(\varphi_{t+k})$$

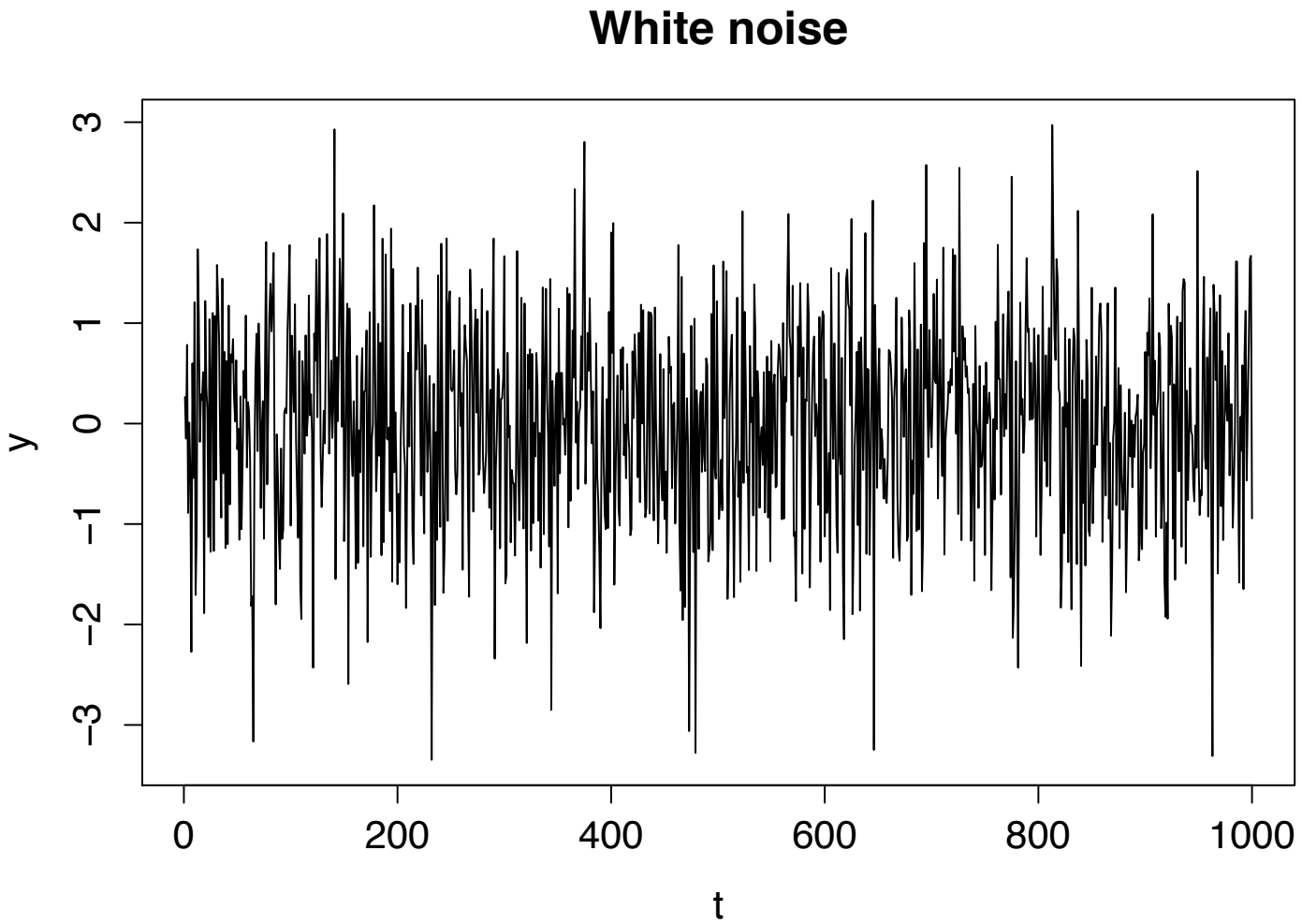
- It follows that this process has constant mean and variance. Also

$$\gamma(k) = \mathrm{Cov}[\varphi_t, \varphi_{t+k}] = 0$$

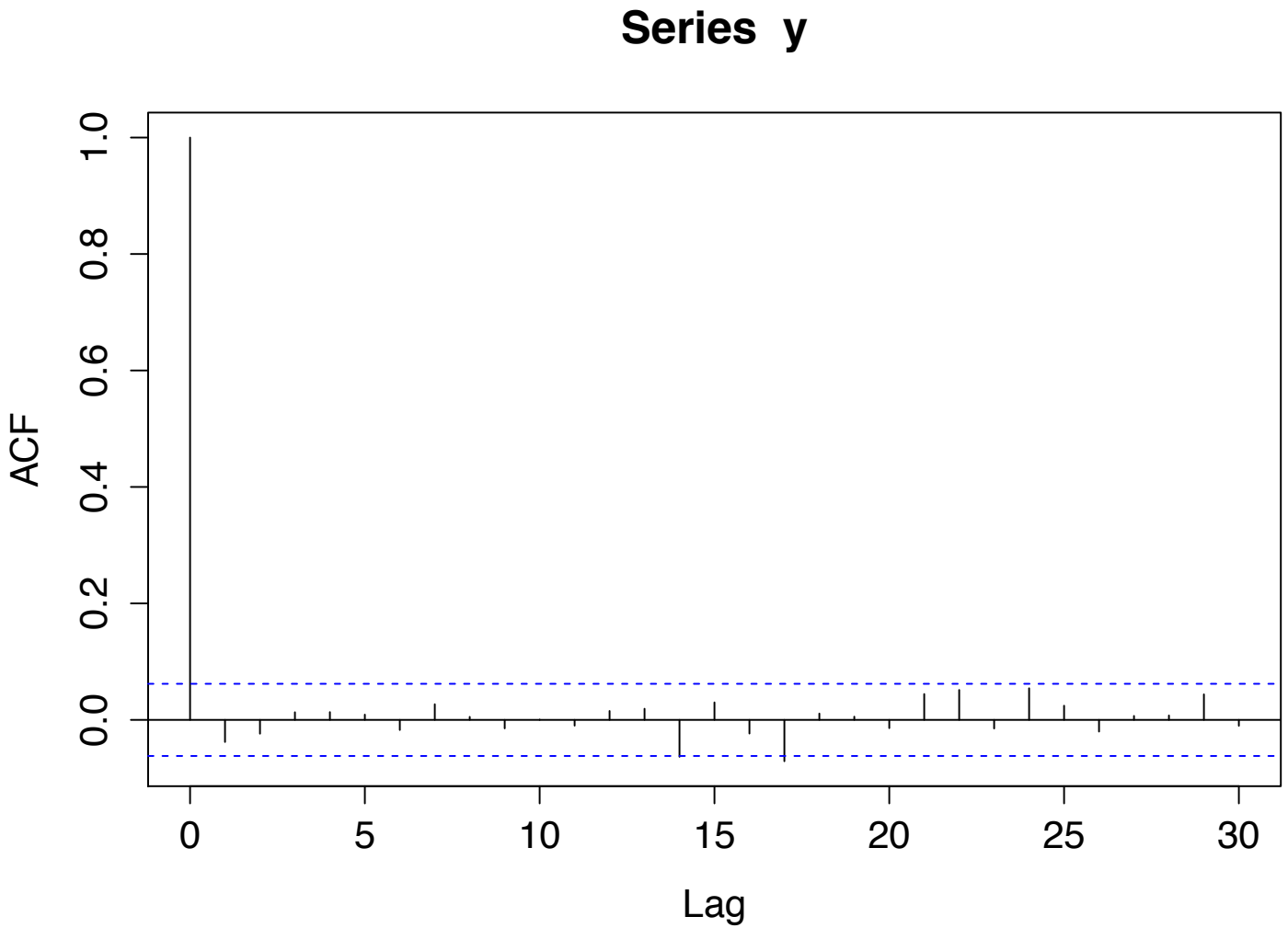
  for $k = \pm 1, \pm 2, \ldots$.

- A purely random process is strictly stationary.
- A purely random process is sometimes called **white noise** by engineers.
- An example of purely random process is the series of numbers drawn by a roulette wheel in a casino.

Example: Gaussian purely random



Example: autocorrelation function



Random walk

- Suppose that $\mathbf{w}_t$ is a discrete, purely random process with mean μ and variance $\sigma^2_{\mathbf{w}}$.
- A process φ_t is said to be a **random walk** if

$$\varphi_t = \varphi_{t-1} + w_t$$

- The next value of a random walk is obtained by summing a random shock to the latest value.
- If $\varphi_0 = 0$ then

$$\varphi_t = \sum_{i=1}^{t} w_i$$

- $E[\varphi_t] = t\mu$ and Var $[\varphi_t] = t\sigma^2_{\mathbf{w}}$.
- As the mean and variance change with t the process is non-stationary.

Random walk (II)

- The first differences of a random walk given by

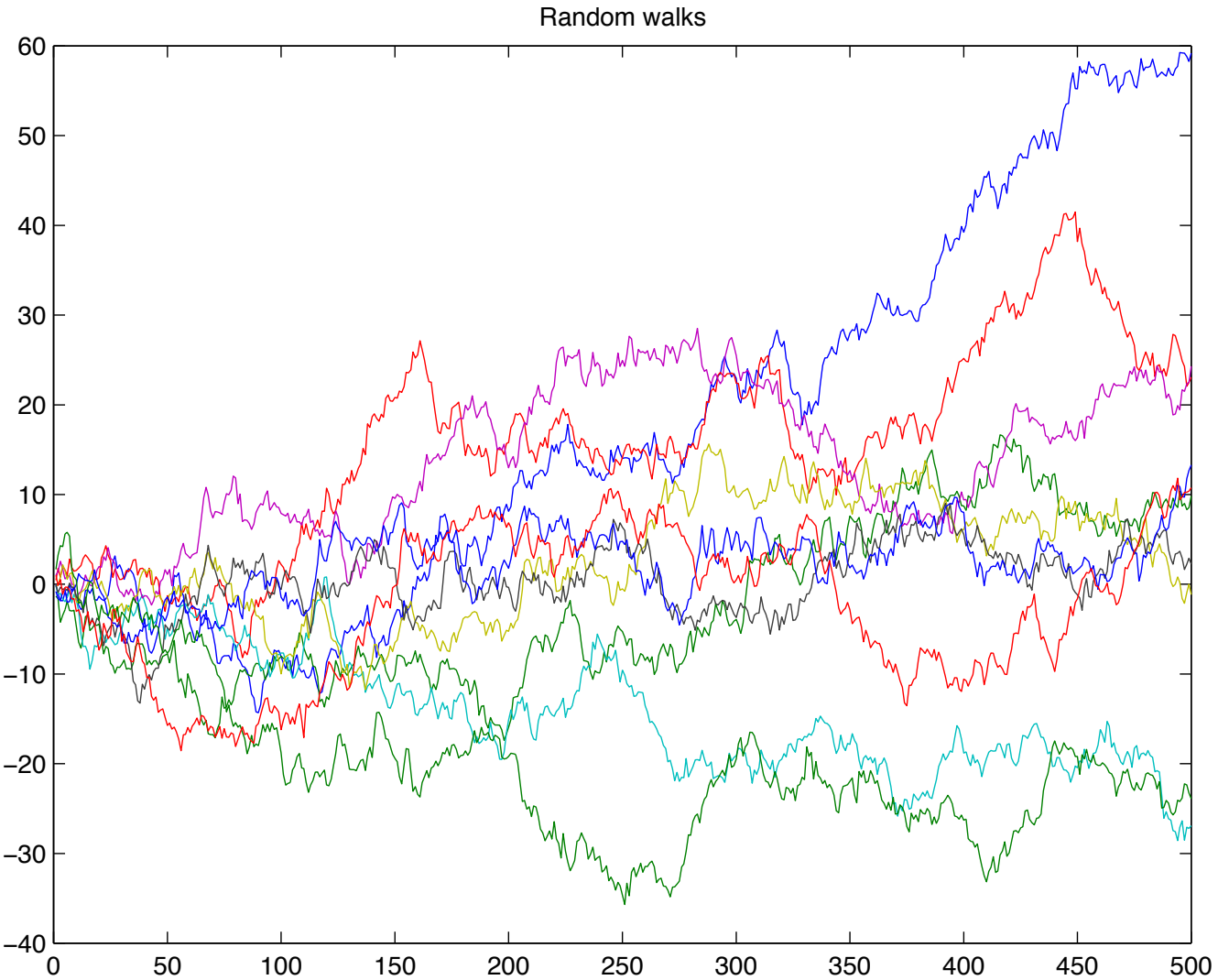
$$\nabla\varphi_t = \varphi_t - \varphi_{t-1}$$

  form a purely random process, which is stationary.

- Examples of time series which behave like random walks are
  - stock prices on successive days.
  - the path traced by a molecule as it travels in a liquid or a gas,
  - the search path of a foraging animal

Ten random walks

Let $\mathbf{w} \sim \mathcal{N}(0, 1)$.



Autoregressive processes

- Suppose that $\mathbf{w}_t$ is a purely random process with mean zero and variance $\sigma^2_{\mathbf{w}}$.
- A process φ_t is said to be an **autoregressive process** of order n (also an **AR(n)** process) if

$$\varphi_t = \alpha_1 \varphi_{t-1} + \cdots + \alpha_n \varphi_{t-n} + w_t$$

- This means that the next value is a linear weighted sum of the past n values plus a random shock.
- Finite memory filter.
- If $\mathbf{w}$ is a normal variable, φ_t will be normal too.
- Note that this is like a linear regression model where φ is regressed not on independent variables but on its past values (hence the prefix “auto”).
- The properties of stationarity depends on the values $\alpha_i, i = 1, \ldots, n$.

First order AR(1) process

If $n = 1$, we have the so-called **Markov process** AR(1)

$$\varphi_t = \alpha\varphi_{t-1} + w_t$$

By substitution it can be shown that

$$\varphi_t = \alpha(\alpha\varphi_{t-2} + w_{t-1}) + w_t = \alpha^2(\alpha\varphi_{t-3} + w_{t-2}) + \alpha w_{t-1} + w_t =$$
$$= w_t + \alpha w_{t-1} + \alpha^2 w_{t-2} + \dots$$

Then

$$E[\varphi_t] = 0 \qquad \text{Var}\,[\varphi_t] = \sigma_{\mathbf{w}}^2(1 + \alpha^2 + \alpha^4 + \dots)$$

Then if $|\alpha| < 1$ the variance if finite and equals

$$\text{Var}\,[\varphi_t] = \sigma_{\varphi}^2 = \sigma_{\mathbf{w}}^2/(1 - \alpha^2)$$

and the autocorrelation is

$$\rho(k) = \alpha^k \qquad k = 0, \dots, 1, 2$$

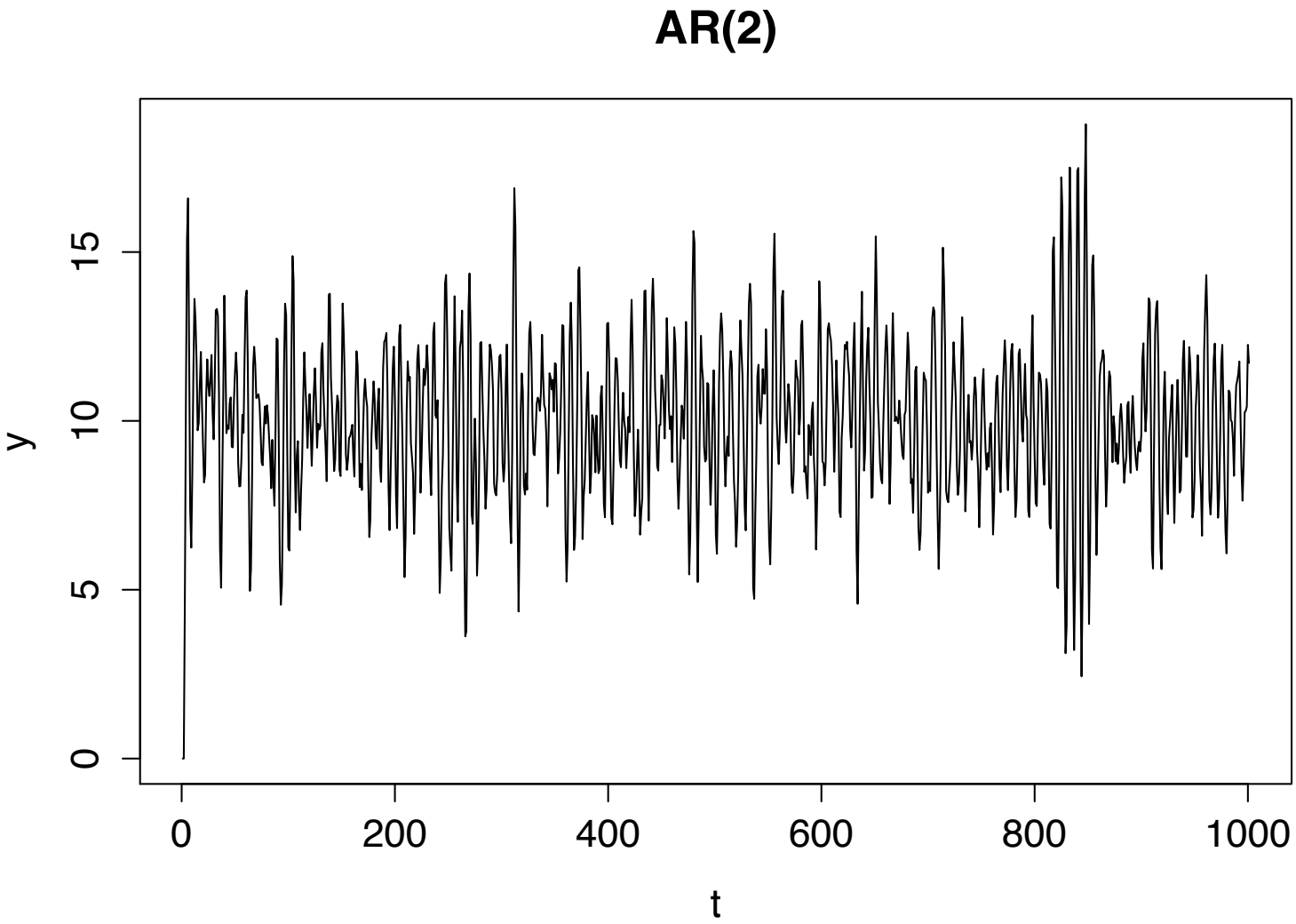
General order AR(n) process

- It has been shown that condition necessary and sufficient for the stationarity is that the complex roots of the equation

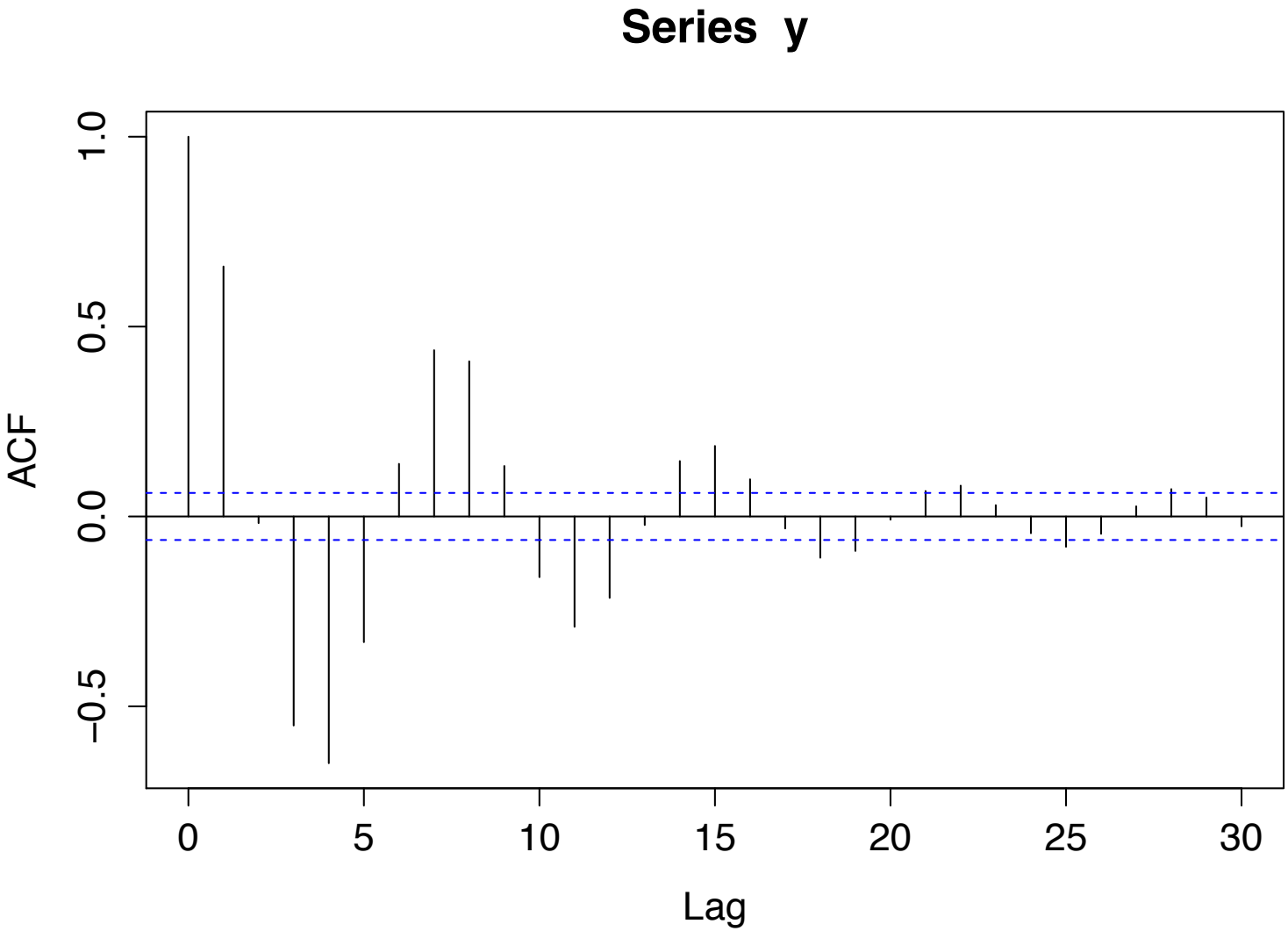
$$\phi(z) = 1 - \alpha_1 z - \cdots - \alpha_n z^n = 0$$

  lie outside the unit circle.
- The autocorrelation function of an AR(n) attenuates slowly with the lag k (exponential decay or damped sine wave pattern).
- On the contrary the partial autocorrelation function cuts off at $k > n$, i.e. it is not significantly different from zero beyond the lag n.

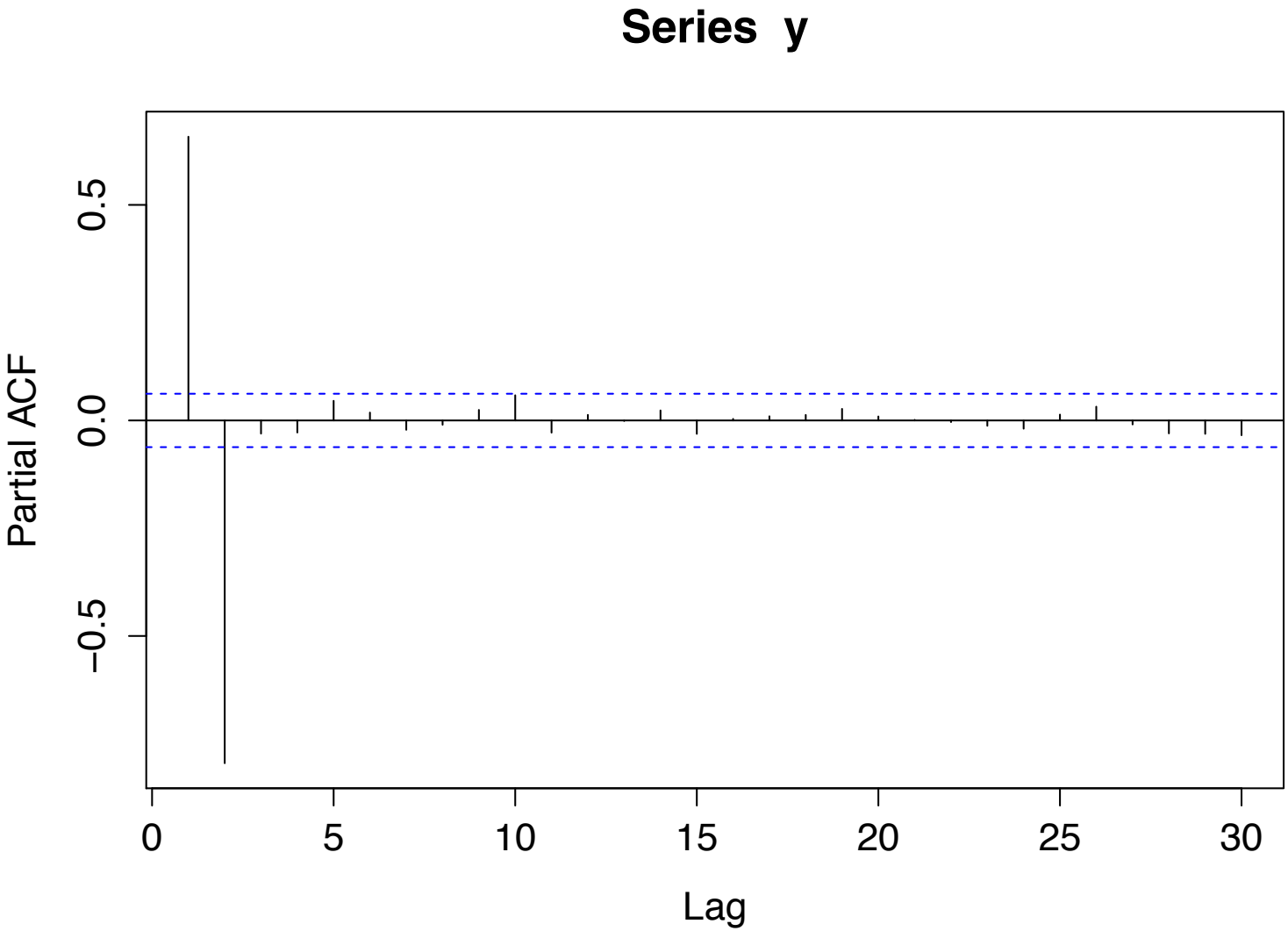
Example: AR(2)



Example: AR(2)



Partial autocorrelation in AR(2)



Fitting an autoregressive process

The estimation of an autoregressive process to a set of data $D_T = \{\varphi_1, \ldots, \varphi_T\}$ demands the resolution of two problems:

1. The estimation of the order n of the process. This is typically supported by the analysis of the partial autocorrelation function.
2. The estimation of the set of parameters $\{\alpha_1, \ldots, \alpha_n\}$.

Estimation of AR(n) parameters

Suppose we have an AR(n) process of order n

$$\varphi_t = \alpha_1 \varphi_{t-1} + \cdots + \alpha_p \varphi_{t-n} + w_t$$

Given T observations, the parameters may be estimated by least-squares by minimizing

$$\hat{\alpha} = \arg\min_{\alpha} \sum_{t=n+1}^{T} \left[\varphi_t - \alpha_1 \varphi_{t-1} - \cdots - \alpha_n \varphi_{t-n}\right]^2$$

In matrix form this amounts to solve the multiple least-squares problem $Y = X\alpha$ where

$$X = \begin{bmatrix} \varphi_{T-1} & \varphi_{T-2} & \cdots & \varphi_{T-n-1} \\ \varphi_{T-2} & \varphi_{T-3} & \cdots & \varphi_{T-n-2} \\ \vdots & \vdots & \vdots & \vdots \\ \varphi_n & \varphi_{n-1} & \cdots & \varphi_1 \end{bmatrix} \quad Y = \begin{bmatrix} \varphi_T \\ \varphi_{T-1} \\ \vdots \\ \varphi_{n+1} \end{bmatrix} \tag{1}$$

Least-squares estimation of AR(n) parms

- Let N be the number of rows of X. In order to estimate the AR(n) parameters we compute **the least-squares estimator**

$$\hat{\alpha} = \arg\min_a \sum_{i=1}^{N} (y_i - x_i^T a)^2 = \arg\min_a \left((Y - Xa)^T (Y - Xa) \right)$$

- It can be shown that

$$\hat{\alpha} = (X^T X)^{-1} X^T Y$$

  where the $X^T X$ matrix is a symmetric $[n \times n]$ matrix which plays an important role in multiple linear regression.

- Conventional linear regression theory provides also confidence interval and significativity tests for the AR(n) coefficients.
- A recursive version of least-squares, i.e. where time samples arrive sequentially, is provided by the RLS algorithm.

From linear to nonlinear setting

The NAR representation

- AR models assume that the relation between past and future is linear
- Once we assume that the linear assumption does not hold, we may extend the AR formulation to a Nonlinear Auto Regressive (NAR) formulation

$$\varphi_t = f\left(\varphi_{t-1}, \varphi_{t-2}, \ldots, \varphi_{t-n}\right) + w(t)$$

  where the missing information is lumped into a noise term $\mathbf{w}$.
- In what follows we will consider this relationship as a particular instance of a dependence

$$\mathbf{y} = f(x) + \mathbf{w}$$

  between a multidimensional input $x \in \mathcal{X} \subset \mathbb{R}^n$ and a scalar output $\mathbf{y} \in \mathbb{R}$.
- NOTA BENE. In what follows y will denote the next value φ_t to be predicted and

$$x = [\varphi_{t-1}, \varphi_{t-2}, \ldots, \varphi_{t-n}]$$

  will denote the n-dimensional input vector also known as *embedding vector*.

Nonlinear vs. linear time series

The advantage of linear models are numerous:

- the least-squares $\hat{\alpha}$ estimate can be expressed in an analytical form
- the least-squares $\hat{\alpha}$ estimate can be easily calculated through matrix computation.
- statistical properties of the estimator can be easily defined.
- recursive formulation for sequential updating are avaialble.
- the relation between empirical and generalization error is known,

BUT...

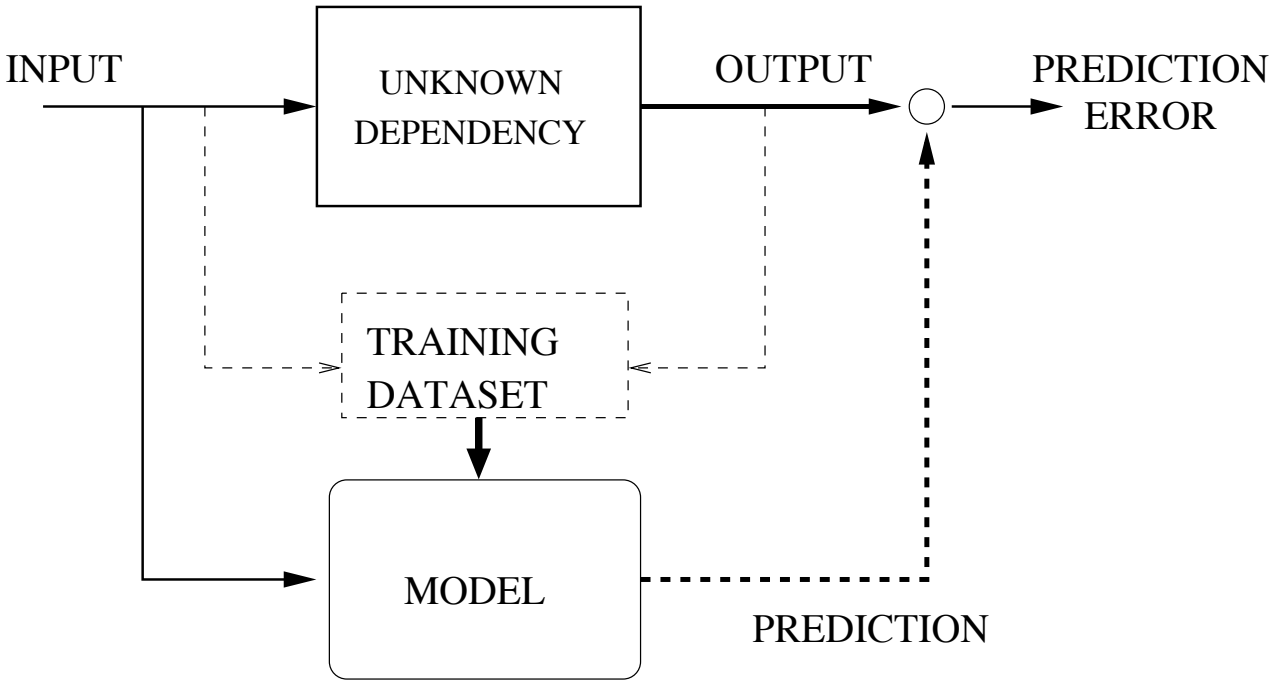
Nonlinear vs. linear time series

- linear methods interpret all the structure in a time series through linear correlation
- deterministic linear dynamics can only lead to simple exponential or periodically oscillating behavior, so all irregular behavior is attributed to external noise while deterministic nonlinear equations could produce very irregular data,
- in real problems it is extremely unlikely that the variables are linked by a linear relation.

In practice, the form of the relation is often unknown and only a limited amount of samples is available.

Machine learning for prediction

Supervised learning



From now on we consider the prediction problem as a problem of **supervised learning** problem, where we have to infer from historical data the possibly nonlinear dependance between the input (past embedding vector) and the output (future value).

Statistical machine learning is the discipline concerned with this problem.

The regression plus noise form

- A typical way of representing the unknown input/output relation is the **regression plus noise form**

$$\mathbf{y} = f(x) + \mathbf{w}$$

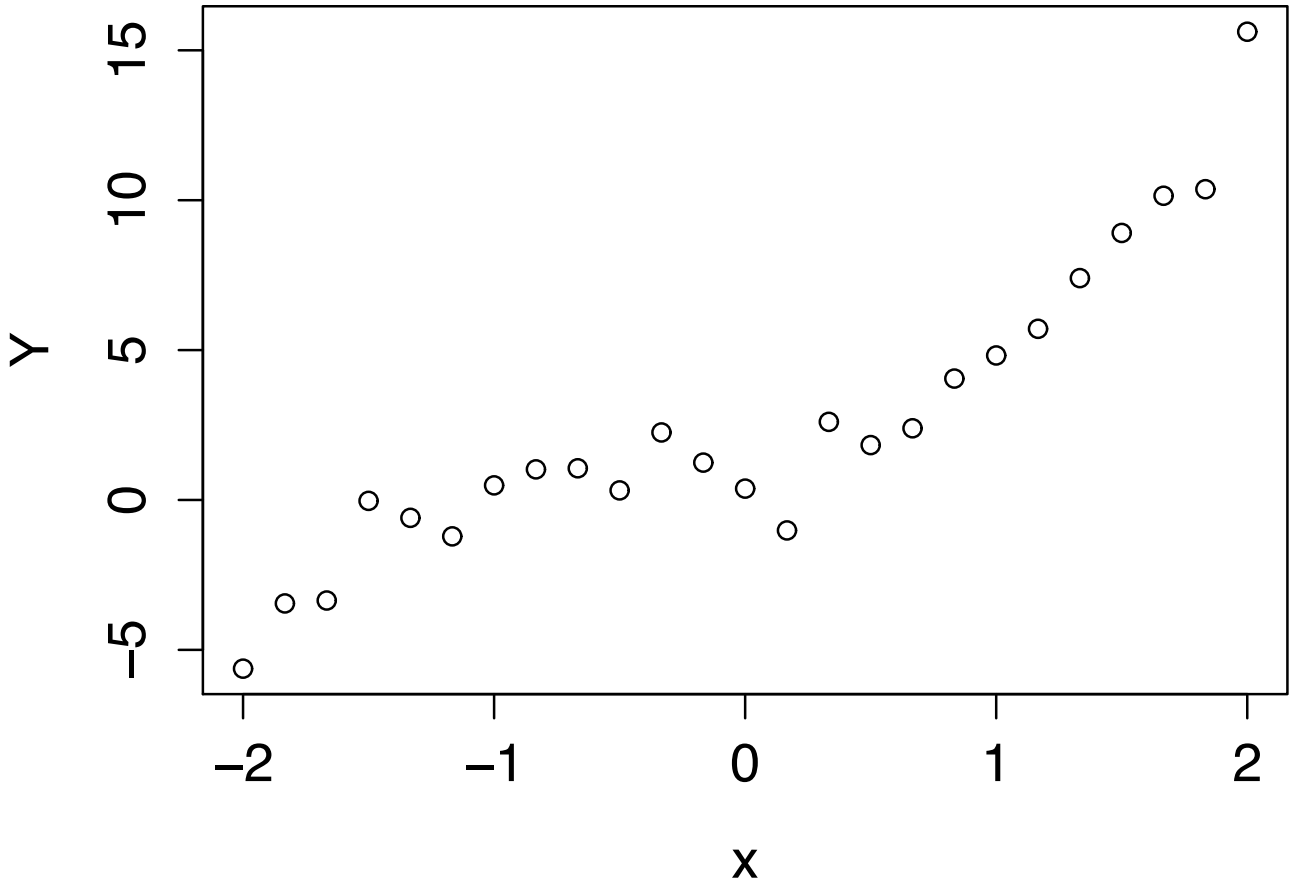
  where $f(\cdot)$ is a deterministic function and the term $\mathbf{w}$ represents the noise or random error. It is typically assumed that $\mathbf{w}$ is independent of $\mathbf{x}$ and $E[\mathbf{w}] = 0$.

- Suppose that we have available a **training set** $\{\langle x_i, y_i \rangle : i = 1, \ldots, N\}$, where $x_i = (x_{i1}, \ldots, x_{in})$ and y_i, generated according to the previous model.

- The goal of a learning procedure is to estimate a model $\hat{f}(x)$ which is able to give a good approximation of the unknown function $f(x)$.

- But how to choose $\hat{f}$, if we do not know the probability distribution underlying the data and we have only a limited training set?

A simple example with $n = 1$



NOTA BENE: this is NOT a time series ! $y = \varphi_t, x = \varphi_{t-1}$. The horizontal axis does not represent time but the past value of the series.

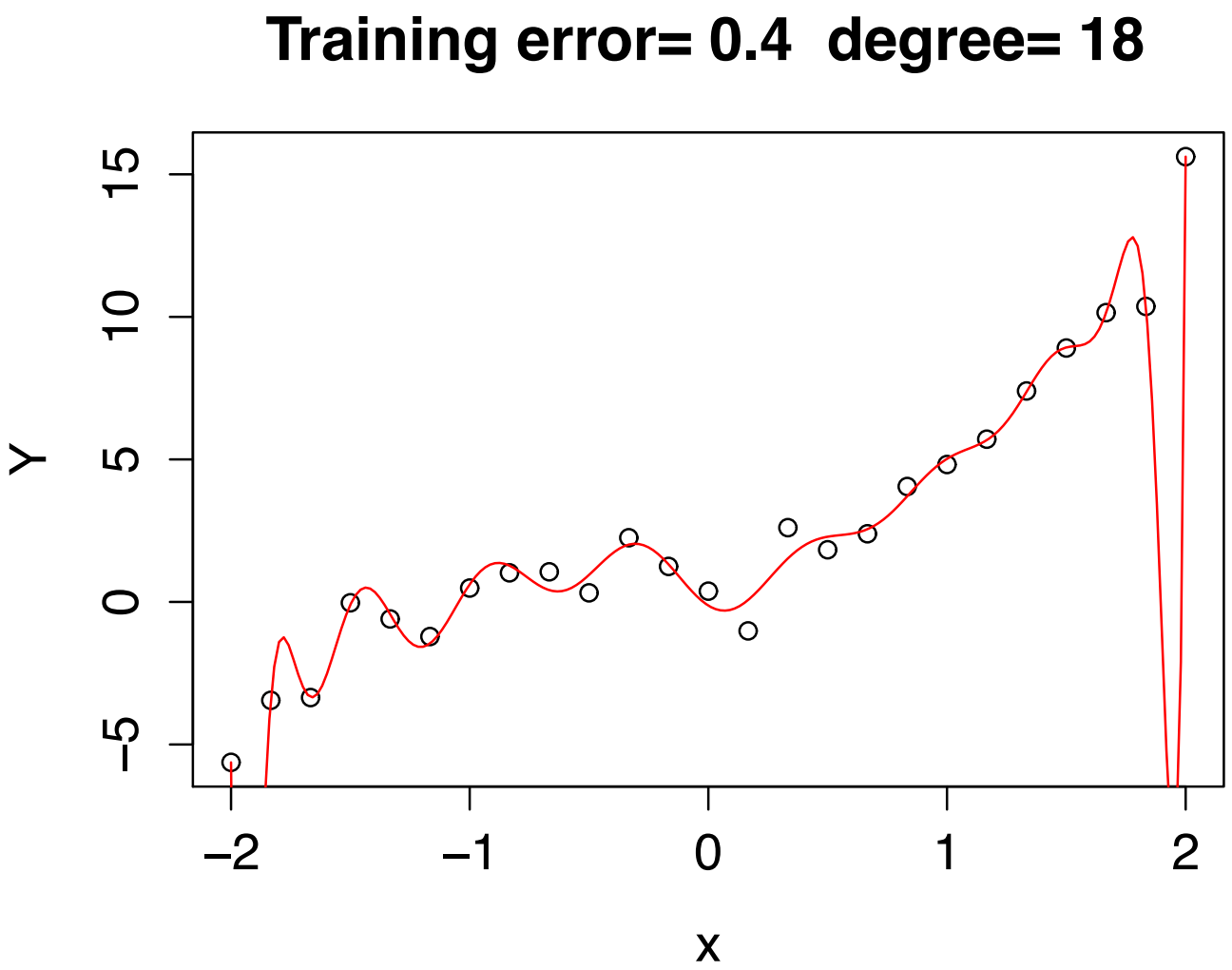
Model degree 1

$$\hat{f}(x) = \alpha_0 + \alpha_1 x$$

Model degree 3

$$\hat{f}(x) = \alpha_0 + \alpha_1 x + \cdots + \alpha_3 x^3$$

Model degree 18



$$\hat{f}(x) = \alpha_0 + \alpha_1 x + \cdots + \alpha_{18} x^1 8$$

Generalization and overfitting

- How to estimate the quality of a model? Is the training error a good measure of the quality?
- The goal of learning is to find a model which is able to **generalize**, i.e. able to return good predictions for input values independent of the training set.
- In a nonlinear setting, it is possible to find models with such a complicate structure that they have null training errors. Are these models good?
- Typically NOT. Since doing very well on the training set could mean doing badly on new data.
- This is the phenomenon of **overfitting**.
- Using the same data for training a model and assessing it is typically a wrong procedure, since this returns an over optimistic assessment of the model generalization capability.

Bias and variance of a model

- A fundamental result of estimation theory shows that the mean-squared-error, i.e. a measure of the generalization quality of an estimator can be decomposed into three terms:

$$\text{MISE} = \sigma^2_{\mathbf{w}} + \text{squared bias} + \text{variance}$$

  where the intrinsic noise term reflects the target alone, the bias reflects the target's relation with the learning algorithm and the variance term reflects the learning algorithm alone.

- This result is purely theoretical since these quantities cannot be measured on the basis of a finite amount of data.
- However, this result provides insight about what makes accurate a learning process.

The bias/variance trade-off

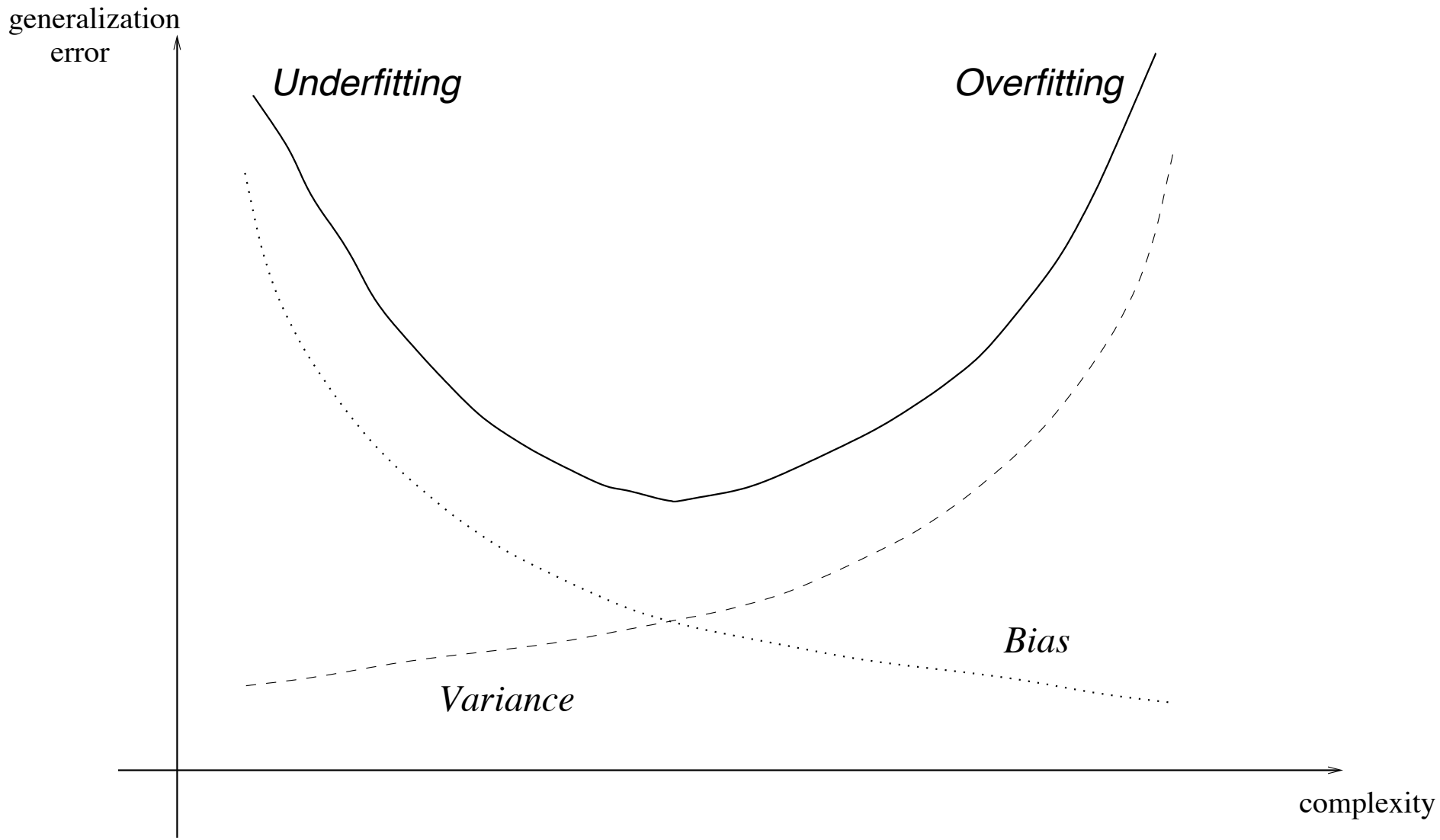
- The first term is the variance of y around its true mean $f(x)$ and cannot be avoided no matter how well we estimate $f(x)$, unless $\sigma^2_{\mathbf{w}} = 0$.
- The bias measures the difference in x between the average of the outputs of the hypothesis functions $\hat{f}$ over the set of possible D_N and the regression function value $f(x)$
- The variance reflects the variability of the guessed $\hat{f}(x, \boldsymbol{\alpha}_N)$ as one varies over training sets of fixed dimension N. This quantity measures how sensitive the algorithm is to changes in the data set, regardless to the target.

The bias/variance dilemma

- The designer of a learning machine has not access to the term MISE but can only estimate it on the basis of the training set. Hence, the bias/variance decomposition is relevant in practical learning since it provides a useful hint about the features to control in order to make the error MISE small.
- The bias term measures the lack of representational power of the class of hypotheses. To reduce the bias term we should consider complex hypotheses which can approximate a large number of input/output mappings.
- The variance term warns us against an excessive complexity of the approximator. This means that a class of too powerful hypotheses runs the risk of being excessively sensitive to the noise affecting the training set; therefore, our class could contain the target but it could be practically impossible to find it out on the basis of the available dataset.

- In other terms, it is commonly said that an hypothesis with large bias but low variance *underfits* the data while an hypothesis with low bias but large variance *overfits* the data.
- In both cases, the hypothesis gives a poor representation of the target and a reasonable trade-off needs to be found.
- The task of the model designer is to search for the optimal trade-off between the variance and the bias term, on the basis of the available training set.

Bias/variance trade-off



The learning procedure

A learning procedure aims at two main goals:

1. to choose a parametric family of hypothesis $\hat{f}(x, \alpha)$ which contains or gives good approximation of the unknown function f (**structural identification**).
2. within the family $\hat{f}(x, \alpha)$, to estimate on the basis of the training set D_N the parameter α_N which best approximates f (**parametric identification**).

In order to accomplish that, a learning procedure is made of two *nested loops*:

1. an external structural identification loop which goes through different model structures
2. an inner parametric identification loop which searches for the best parameter vector within the family structure.

Parametric identification

The parametric identification of the hypothesis is done according to ERM (Empirical Risk Minimization) principle where

$$\alpha_N = \alpha(D_N) = \arg\min_{\alpha \in \Lambda} \widehat{\mathsf{MISE}}_{\mathsf{emp}}(\alpha)$$

minimizes the **training error**

$$\widehat{\mathsf{MISE}}_{\mathsf{emp}}(\alpha) = \frac{\sum_{i=1}^{N} \left(y_i - \hat{f}(x_i, \alpha) \right)^2}{N}$$

constructed on the basis of the training data set D_N.

Parametric identification (II)

- The computation of α_N requires a procedure of multivariate optimization in the space of parameters.
- The complexity of the optimization depends on the form of $\hat{f}(\cdot)$.
- In some cases the parametric identification problem may be an NP-hard problem.
- Thus, we must resort to some form of heuristic search.
- Examples of parametric identification procedure are linear least-squares for linear models and backpropagated gradient-descent for feedforward neural networks.

Model assessment

- We have seen before that the training error is not a good estimator (i.e. it is too optimistic) of the generalization capability of the learned model.
- Two alternative exists:
  1. Complexity-based penalty criteria
  2. Data-driven validation techniques

Complexity-based penalization

- In conventional statistics, various criteria have been developed, often in the context of linear models, for assessing the generalization performance of the learned hypothesis without the use of further validation data.
- Such criteria take the form of a sum of two terms

$$\hat{G}_{\text{PE}} = \widehat{\text{MISE}}_{\text{emp}} + \text{complexity term}$$

  where the complexity term represents a penalty which grows as the number of free parameters in the model grows.
- This expression quantifies the qualitative consideration that simple models return high training error with a reduced complexity term while complex models have a low training error thanks to the high number of parameters.
- The minimum for the criterion represents a trade-off between performance on the training set and complexity.

Complexity-based penalty criteria

If the input/output relation is linear, well-known examples of complexity based criteria are:

- the Final Prediction Error (FPE)

$$\text{FPE} = \widehat{\text{MISE}}_{\text{emp}}(\alpha_N)\frac{1+p/N}{1-p/N}$$

  with $p = n + 1$,

- the Generalized Cross-Validation (GCV)

$$\text{GCV} = \widehat{\text{MISE}}_{\text{emp}}(\alpha_N)\frac{1}{(1-\frac{p}{N})^2}$$

- the Akaike Information Criterion (AIC)

$$\text{AIC} = \frac{p}{N} - \frac{1}{N}L(\alpha_N)$$

  where $L(\cdot)$ is the log-likelihood function,

- the C_p criterion proposed by Mallows

$$C_p = \frac{\widehat{\mathsf{MISE}}_{\mathsf{emp}}(\alpha_N)}{\hat{\sigma}_w^2} + 2p - N$$

  where $\hat{\sigma}_w^2$ is an estimate of the variance of noise,

- the Predicted Squared Error (PSE)

$$\mathsf{PSE} = \widehat{\mathsf{MISE}}_{\mathsf{emp}}(\alpha_N) + 2\hat{\sigma}_w^2 \frac{p}{N}$$

  where $\hat{\sigma}_w^2$ is an estimate of the variance of noise.

Data-driven validation techniques

If no (e.g. linear) assumptions are made, how to measure MISE in a reliable way on a finite dataset? The most common techniques to return an estimate $\widehat{\text{MISE}}$ are

Testing: a *testing sequence* independent of D_N and distributed according to the same probability distribution is used to assess the quality. In practice, unfortunately, an additional set of input/output observations is rarely available.

Holdout: The *holdout* method, sometimes called test sample estimation, partitions the data D_N into two mutually exclusive subsets, the training set D_{tr} and the holdout or test set $D_{N_{ts}}$.

k-fold Cross-validation: the set D_N is randomly divided into k mutually exclusive test partitions of approximately equal size. The cases not found in each test partition are independently used for selecting the hypothesis which will be tested on the partition itself. The average error over all the k partitions is the cross-validated error rate.

The K-fold cross-validation

This is the algorithm in detail:

1. split the dataset D_N into k roughly equal-sized parts.
2. For the kth part $k = 1, \ldots, K$, fit the model to the other $K - 1$ parts of the data, and calculate the prediction error of the fitted model when predicting the k-th part of the data.
3. Do the above for $k = 1, \ldots, K$ and average the K estimates of prediction error.

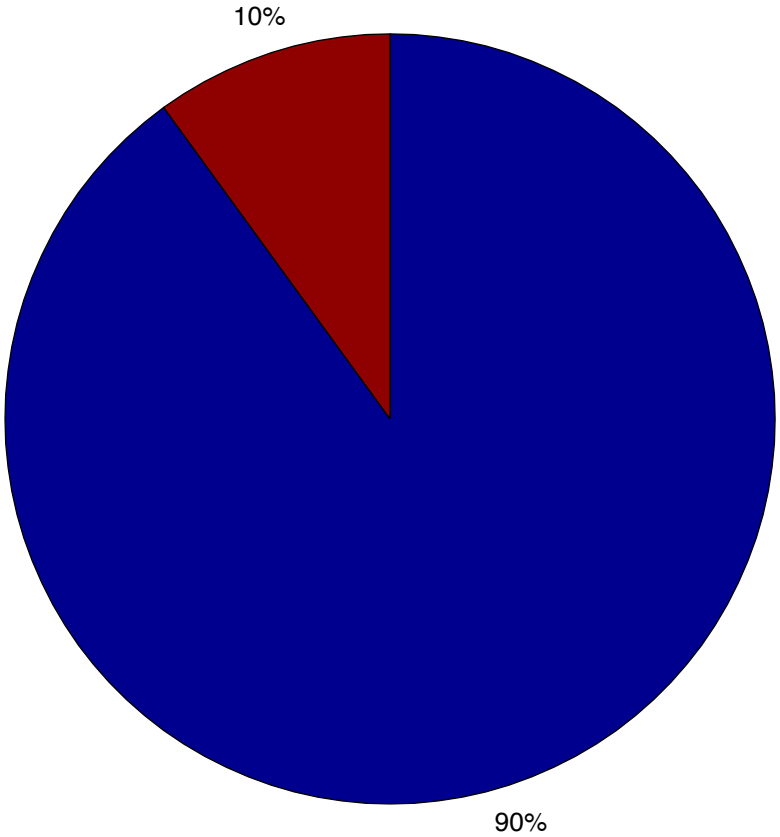
Let $k(i)$ be the part of D_N containing the ith sample. Then the cross-validation estimate of the MISE prediction error is

$$\widehat{\mathsf{MISE}}_{\mathsf{CV}} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i^{-k(i)})^2 = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{f}(x_i, \alpha^{-k(i)})\right)^2$$

where $\hat{y}_i^{-k(i)}$ denotes the fitted value for the ith observation returned by the model estimated with the $k(i)$th part of the data removed.

10-fold cross-validation

$K = 10$: at each iteration 90% of data are used for training and the remaining 10% for the test.



Leave-one-out cross validation

- The cross-validation algorithm where $K = N$ is also called the **leave-one-out** algorithm.
- This means that for each ith sample, $i = 1, \ldots, N$,
  1. we carry out the parametric identification, leaving that observation out of the training set,
  2. we compute the predicted value for the ith observation, denoted by $\hat{y}_i^{-i}$

The corresponding estimate of the MISE prediction error is

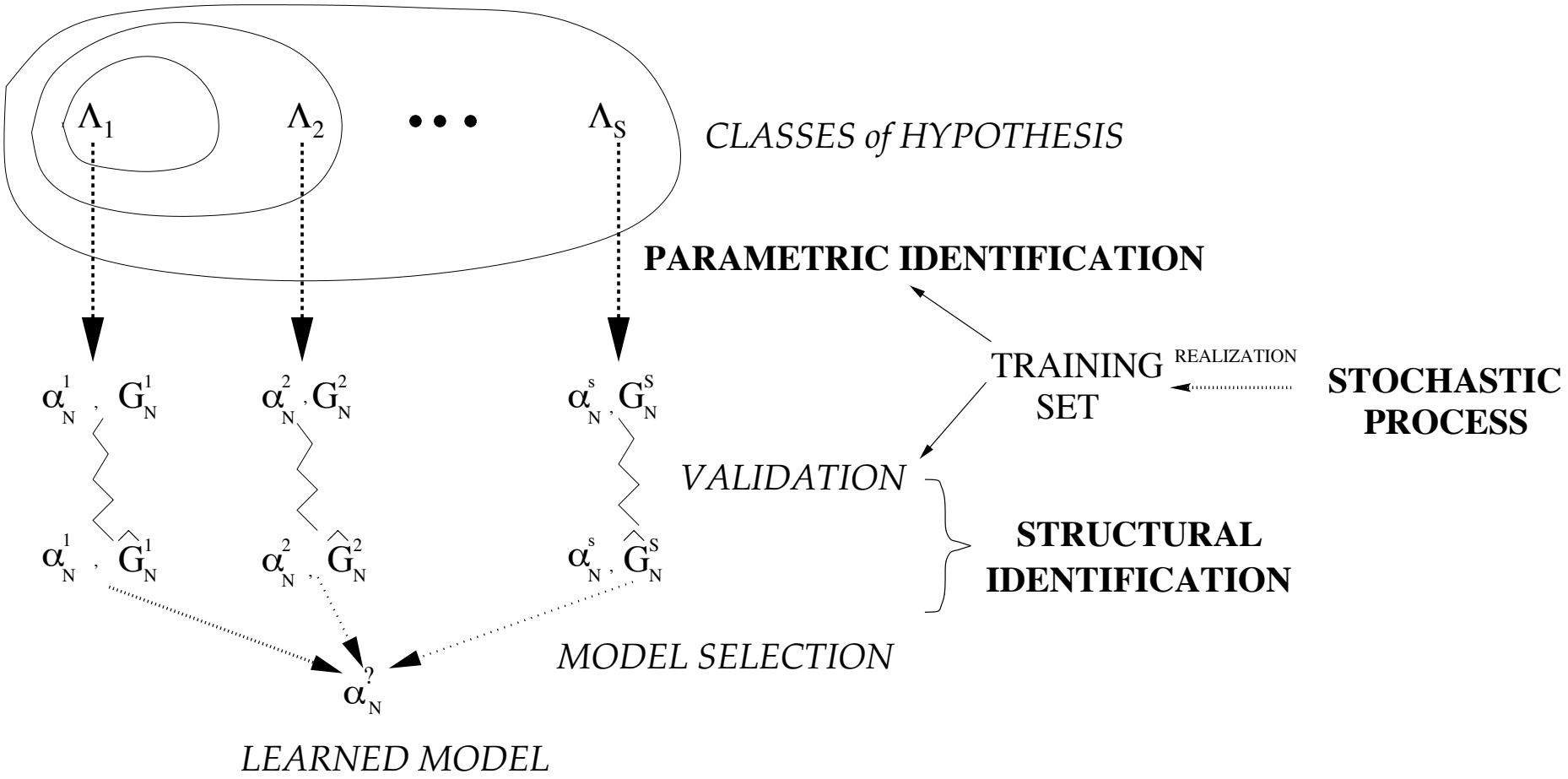
$$\widehat{\mathsf{MISE}}_{\mathsf{LOO}} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i^{-i})^2 = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{f}\left(x_i, \alpha^{-i}\right)\right)^2$$

where α^{-i} is the set of parameters returned by the parametric identification perfomed on the training set with the ith sample set aside.

Model selection

- Model selection concerns the final choice of the model structure
- By structure we mean:
  - family of the approximator (e.g. linear, non linear) and if nonlinear which kind of learner (e.g. neural networks, support vector machines, nearest-neighbours, regression trees)
  - the value of hyper parameters (e.g. number of hidden layers, number of hidden nodes in NN, number of neighbors in KNN, number of levels in trees)
  - number and set of input variables
- this choice is typically the result of a compromise between different factors, like the quantitative measures, the personal experience of the designer and the effort required to implement a particular model in practice.
- Here we will consider only quantitative criteria. Two are the possible approaches:
  1. the *winner-takes-all* approach
  2. the *combination of estimators* approach.

Model selection



Winner-takes-all

The best hypothesis is selected in the set $\{\alpha_N^s\}$, with $s = 1, \ldots, S$, according to

$$\tilde{s} = \arg \min_{s=1,\ldots,S} \widehat{\mathsf{MISE}}^s$$

A model with complexity $\tilde{s}$ is trained on the whole dataset D_N and used for future predictions.

Winner-takes-all pseudo-code

1. for $s = 1, \ldots, S$: (Structural loop)
   - for $j = 1, \ldots, N$

     (a) Inner parametric identification (for l-o-o):

$$\alpha_{N-1}^{s} = \arg\min_{\alpha \in \Lambda_s} \sum_{i=1:N, i \neq j} (y_i - \hat{f}(x_i, \alpha))^2$$

     (b) $e_j = y_j - \hat{f}(x_j, \alpha_{N-1}^{s})$
   - $\widehat{\text{MISE}}_{\text{LOO}}(s) = \frac{1}{N} \sum_{j=1}^{N} e_j^2$
2. Model selection: $\tilde{s} = \arg\min_{s=1,\ldots,S} \widehat{\text{MISE}}_{\text{LOO}}(s)$
3. Final parametric identification:
   $\alpha_N^{\tilde{s}} = \arg\min_{\alpha \in \Lambda_{\tilde{s}}} \sum_{i=1}^{N} (y_i - \hat{f}(x_i, \alpha))^2$
4. The output prediction model is $\hat{f}(\cdot, \alpha_N^{\tilde{s}})$

Model combination

- The winner-takes-all approach is intuitively the approach which should work the best.
- However, recent results in machine learning show that the performance of the final model can be improved not by choosing the model structure which is expected to predict the best but by creating a model whose output is the combination of the output of models having different structures.
- The reason is that in reality any chosen hypothesis $\hat{f}(\cdot, \alpha_N)$ is only an estimate of the real target and, like any estimate, is affected by a bias and a variance term.
- Theoretical results on the combination of estimators show that the combination of unbiased estimators leads an unbiased estimator with reduced variance.
- This principle is at the basis of approaches like *bagging* or *boosting*.

Feature selection problem

- Machine learning algorithms are known to degrade in performance (prediction accuracy) when faced with many inputs (aka *features*) that are not necessary for predicting the desired output.
- In the feature selection problem, a learning algorithm is faced with the problem of selecting some subset of features upon which to focus its attention, while ignoring the rest.
- Using all available features may negatively affect generalization performance, especially in the presence of irrelevant or redundant features.
- Feature selection can be seen as an instance of model selection problem.

Benefits and drawbacks of feature selection

There are many potential benefits of feature selection:

- facilitating data visualization and data understanding,
- reducing the measurement and storage requirements,
- reducing training and utilization times of the final model,
- defying the curse of dimensionality to improve prediction performance.

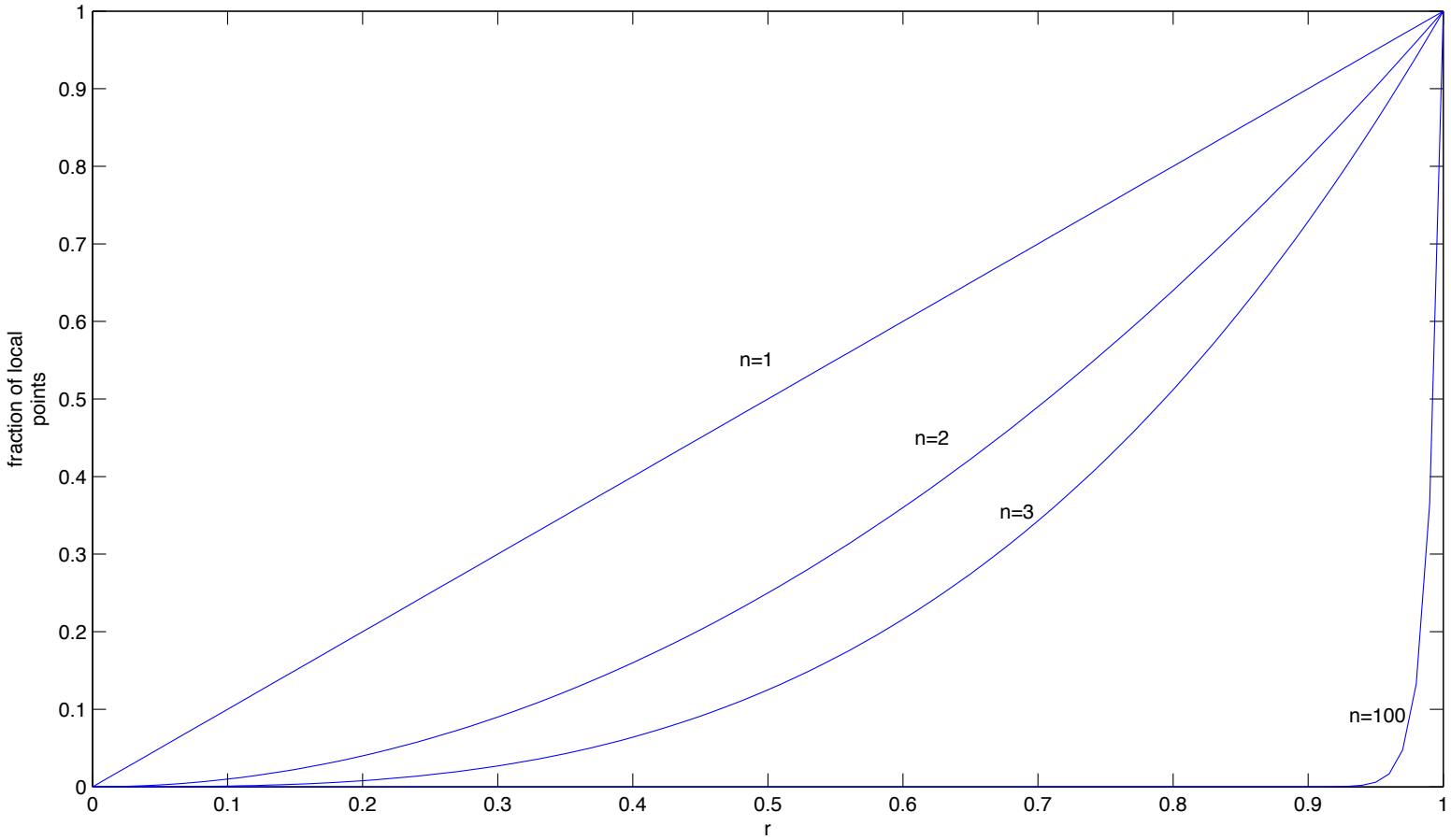
Drawbacks are

- the search for a subset of relevant features introduces an additional layer of complexity in the modelling task. The search in the model hypothesis space is augmented by another dimension: the one of finding the optimal subset of relevant features.
- additional time for learning.

Curse of dimensionality

- The error of the best model decreases with n but the mean integrated squared error of models increases faster than linearly in n.
- In high dimensions, all data sets are sparse.
- In high dimensions, the number of possible models to consider increases superexponenetially in n.
- In high dimensions, all datasets show multicollinearity.
- As n increases the amount of local data goes to zero.
- For a uniform distribution around a query point x_q the amount of data that are contained in a ball of radius $r < 1$ centered in x_q grows like r^n.

Local density for large n



The size of the neighborhood on which we can estimate local features of the output (e.g. $E[y|x]$) increases with dimension n, making the estimation coarser and coarser.

Methods of feature selection

Two are the main approaches to feature selection:

Filter methods: they are preprocessing methods. They attempt to assess the merits of features from the data, ignoring the effects of the selected feature subset on the performance of the learning algorithm. Examples are methods that select variables by ranking them through compression techniques (like PCA or clustering) or by computing correlation with the output.

Wrapper methods: these methods assess subsets of variables according to their usefulness to a given predictor. The method conducts a search for a good subset using the learning algorithm itself as part of the evaluation function. The problem boils down to a problem of stochastic state space search. Example are the stepwise methods proposed in linear regression analysis.

Embedded methods: they perform variable selection as part of the learning procedure and are usually specific to given learning machines. Examples are classification trees, random forests, and methods based on regularization techniques (e.g. lasso)

Local learning

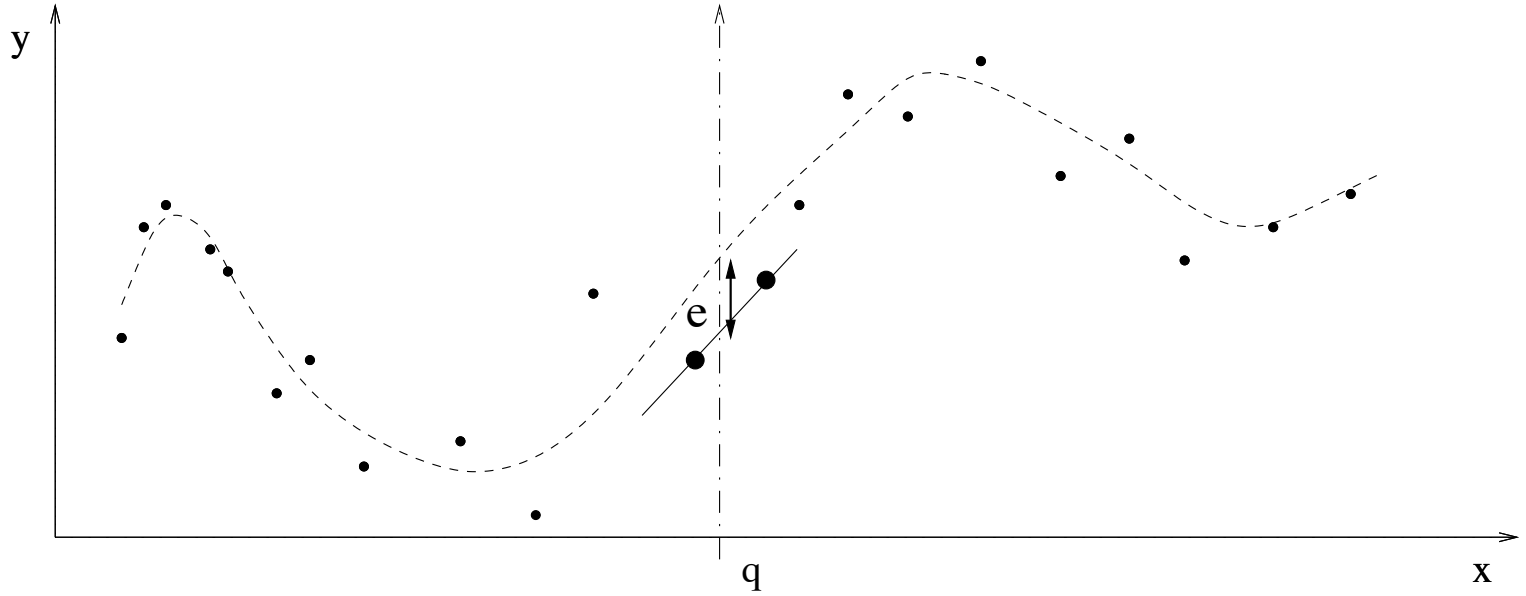
Local modeling procedure

The learning of a local model in $x_q \in \mathbb{R}^n$ can be summarized in these steps:

1. Compute the distance between the query x_q and the training samples according to a predefined *metric*.
2. Rank the neighbors on the basis of their distance to the query.
3. Select a subset of the k nearest neighbors according to the *bandwidth* which measures the size of the neighborhood.
4. Fit a *local model* (e.g. constant, linear,...).

Each of the local approaches has one or more structural (or smoothing) parameters that control the amount of smoothing performed.
Let us focus on the *bandwidth selection*.

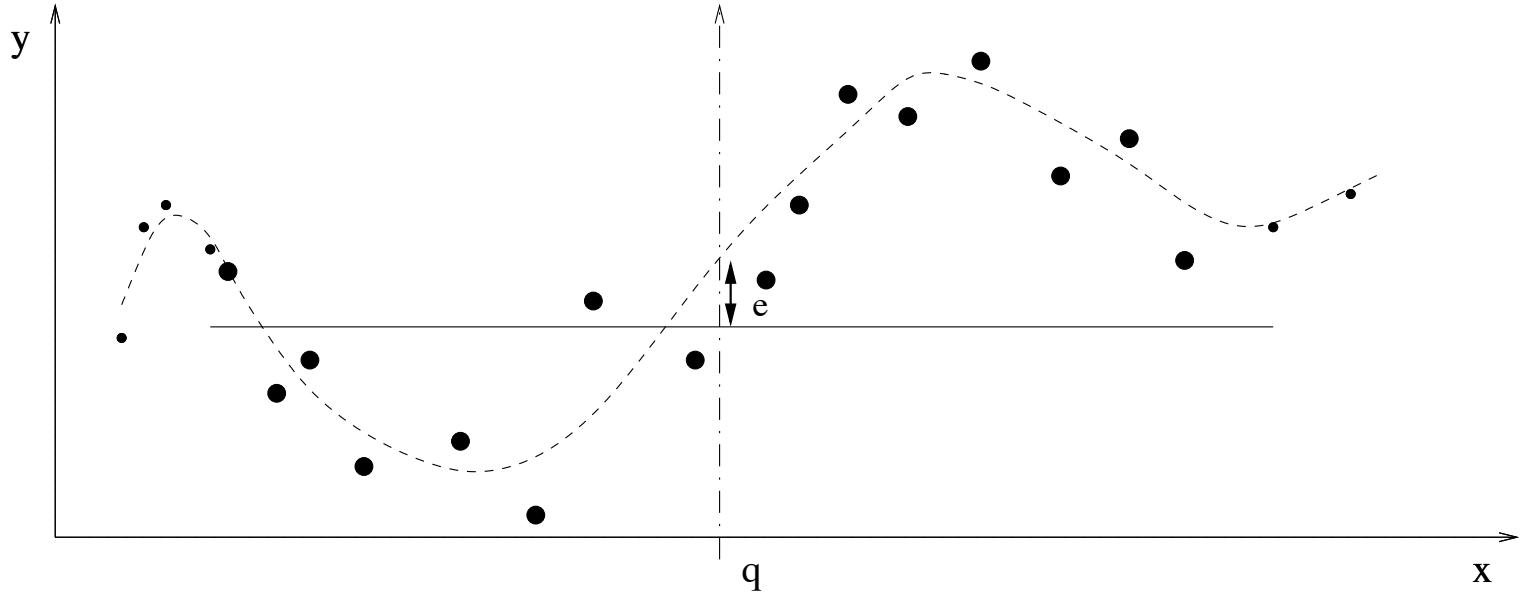
The bandwidth trade-off: overfit



Too narrow bandwidth $\Rightarrow$ overfitting $\Rightarrow$ large prediction error e.

In terms of bias/variance trade-off, this is typically a situation of high variance.

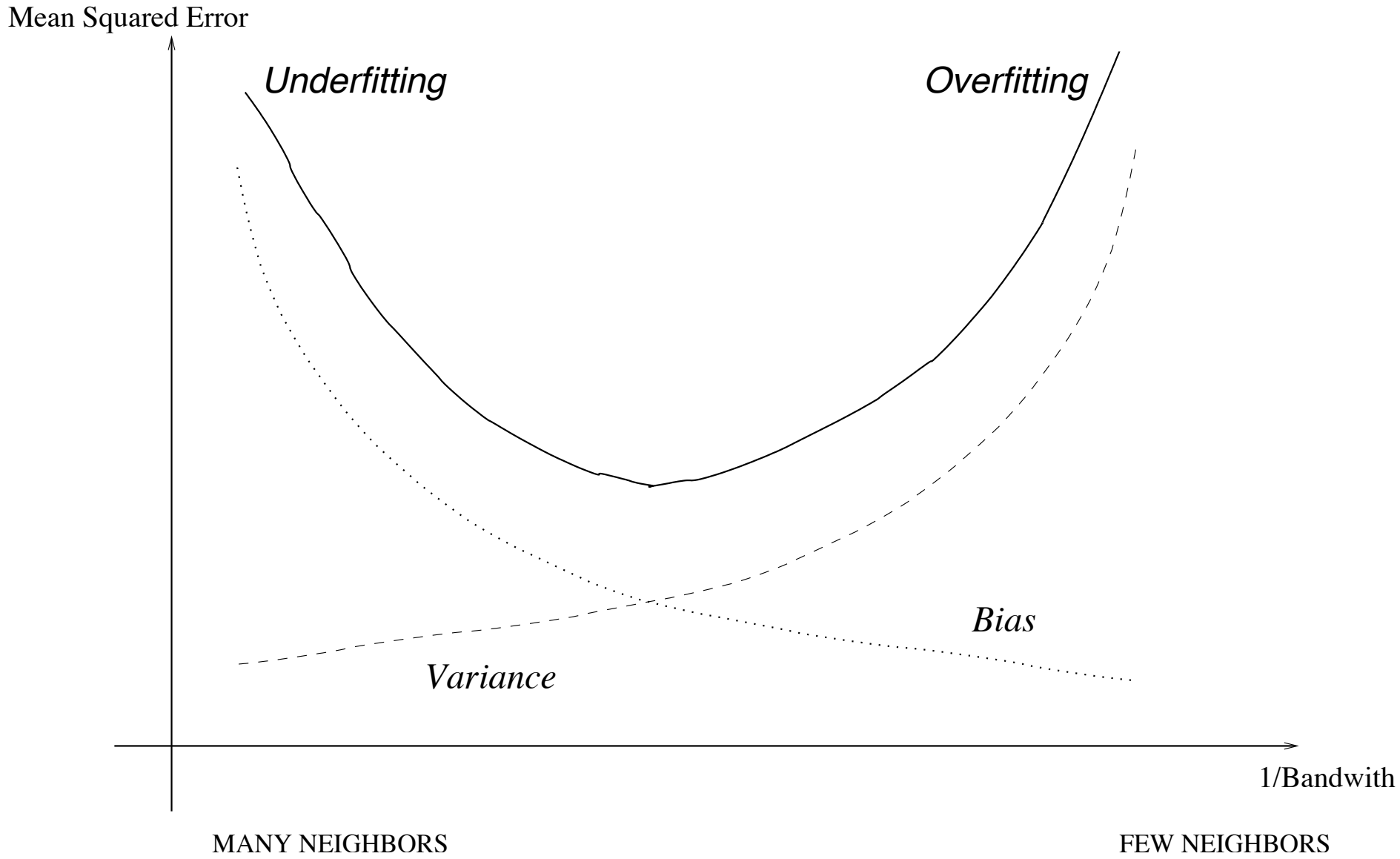
The bandwidth trade-off: underfit



Too large bandwidth $\Rightarrow$ underfitting $\Rightarrow$ large prediction error e

In terms of bias/variance trade-off, this is typically a situation of high bias.

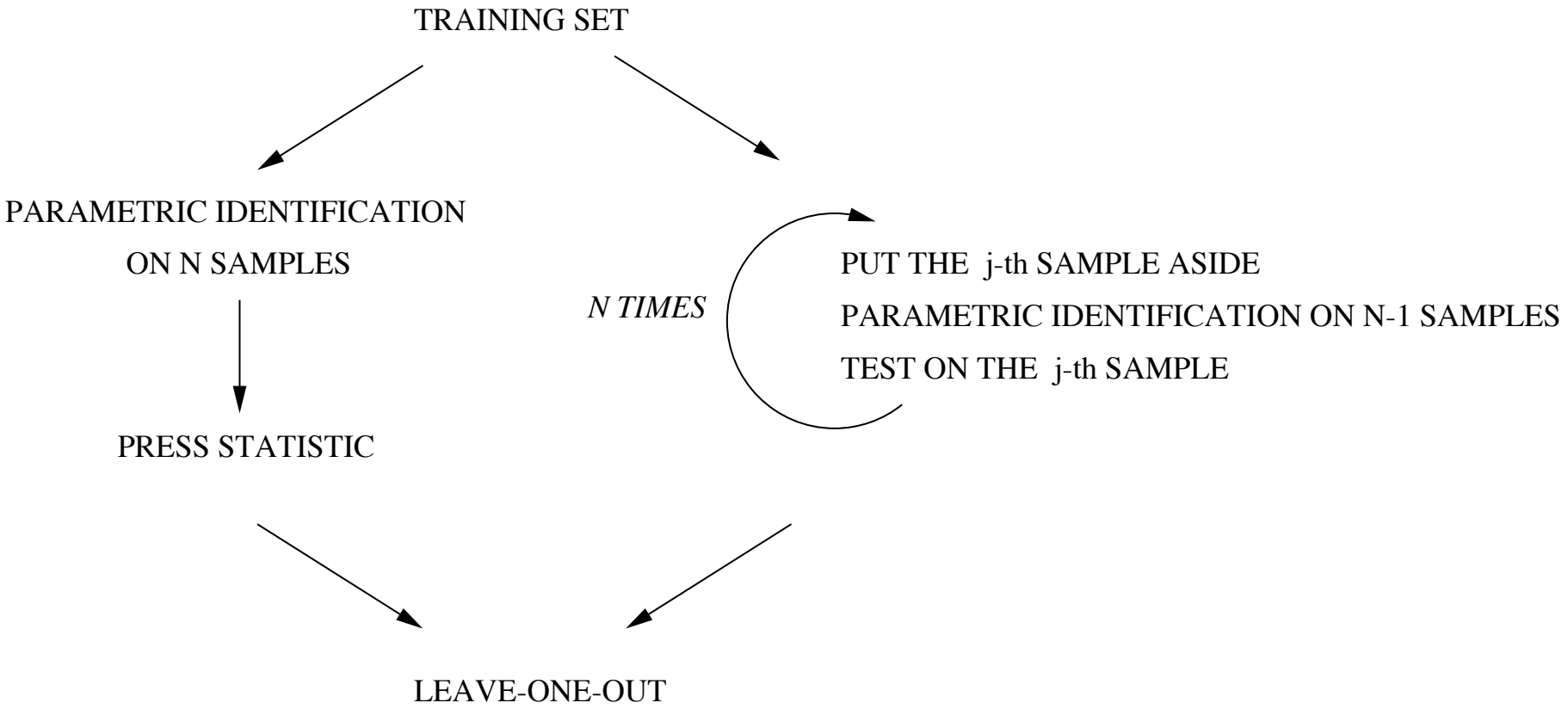
Bandwidth and bias/variance trade-off



The PRESS statistic

- Cross-validation can provide a reliable estimate of the algorithm generalization error but it requires the training process to be repeated K times, which sometimes means a large computational effort.
- In the case of linear models there exists a powerful statistical procedure to compute the leave-one-out cross-validation measure at a reduced computational cost
- It is the PRESS (Prediction Sum of Squares) statistic, a simple formula which returns the leave-one-out (l-o-o) as a by-product of the least-squares.

Leave-one-out for linear models



The leave-one-out error can be computed in two equivalent ways: the slowest way (on the right) which repeats N times the training and the test procedure; the fastest way (on the left) which performs only once the parametric identification and the computation of the PRESS statistic.

The PRESS statistic

- This allows a fast cross-validation without repeating N times the leave-one-out procedure. The PRESS procedure can be described as follows:
  1. we use the whole training set to estimate the linear regression coefficients
  $$\hat{\alpha} = (X^T X)^{-1} X^T Y$$
  2. This procedure is performed only once on the N samples and returns as by product the Hat matrix
  $$H = X(X^T X)^{-1} X^T$$
  3. we compute the residual vector e, whose j^{th} term is $e_j = y_j - x_j^T \hat{\alpha}$,
  4. we use the PRESS statistic to compute e_j^{loo} as
  $$e_j^{\text{loo}} = \frac{e_j}{1 - H_{jj}}$$
  where H_{jj} is the j^{th} diagonal term of the matrix H.

The PRESS statistic

Thus, the leave-one-out estimate of the local mean integrated squared error is:

$$\widehat{\mathrm{MISE}}_{\mathrm{LOO}} = \frac{1}{N}\sum_{i=1}^{N}\left\{\frac{y_i - \hat{y}_i}{1 - H_{ii}}\right\}^2$$

Note that PRESS is not an approximation of the loo error but simply a faster way of computing it.

Selection of the number of neighbours

- For a given query point x_q, we can compute a set of predictions

$$\hat{y}_q(k) = x_q^T \hat{\alpha}(k)$$

, together with a set of associated leave-one-out error vectors $\widehat{\mathsf{MISE}}_{\mathsf{LOO}}(k)$ for a number of neighbors ranging in $[k_{\mathsf{min}}, k_{\mathsf{max}}]$.

- If the selection paradigm, frequently called *winner-takes-all*, is adopted, the most natural way to extract a final prediction $\hat{y}_q$, consists in comparing the prediction obtained for each value of k on the basis of the classical *mean square error* criterion:

$$\hat{y}_q = x_q^T \hat{\alpha}(\hat{k}), \quad \text{with } \hat{k} = \arg\min_k \widehat{\mathsf{MISE}}_{\mathsf{LOO}}(k)$$

Local Model combination

- As an alternative to the *winner-takes-all* paradigm, we can use a combination of estimates.
- The final prediction of the value y_q is obtained as a weighted average of the best b models, where b is a parameter of the algorithm.
- Suppose the predictions $\hat{y}_q(k)$ and the loo errors $\widehat{\mathsf{MISE}}_{\mathsf{LOO}}(k)$ have been ordered creating a sequence of integers $\{k_i\}$ so that $\widehat{\mathsf{MISE}}_{\mathsf{LOO}}(k_i) \leq \widehat{\mathsf{MISE}}_{\mathsf{LOO}}(k_j)$, $\forall i < j$. The prediction of $\hat{y}_q$ is given by

$$\hat{y}_q = \frac{\sum_{i=1}^{b} \zeta_i \hat{y}_q(k_i)}{\sum_{i=1}^{b} \zeta_i},$$

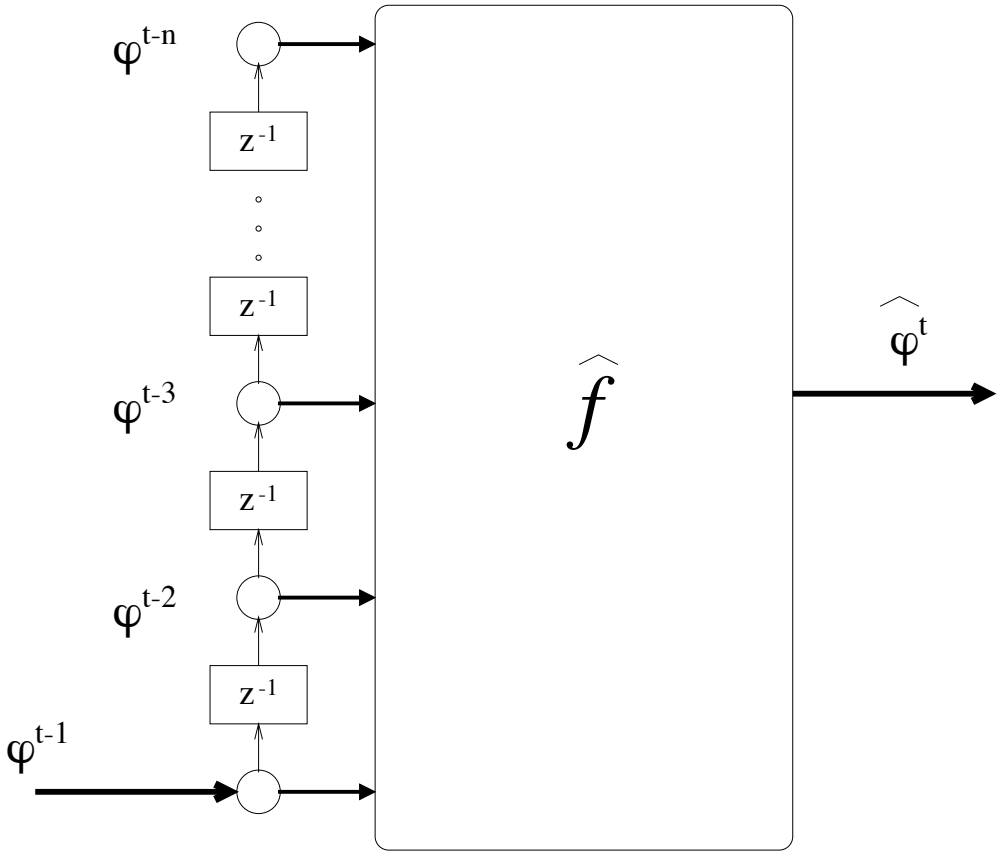
where the weights are the inverse of the mean square errors: $\zeta_i = 1/\widehat{\mathsf{MISE}}_{\mathsf{LOO}}(k_i)$.

Forecasting

One step-ahead and iterated prediction

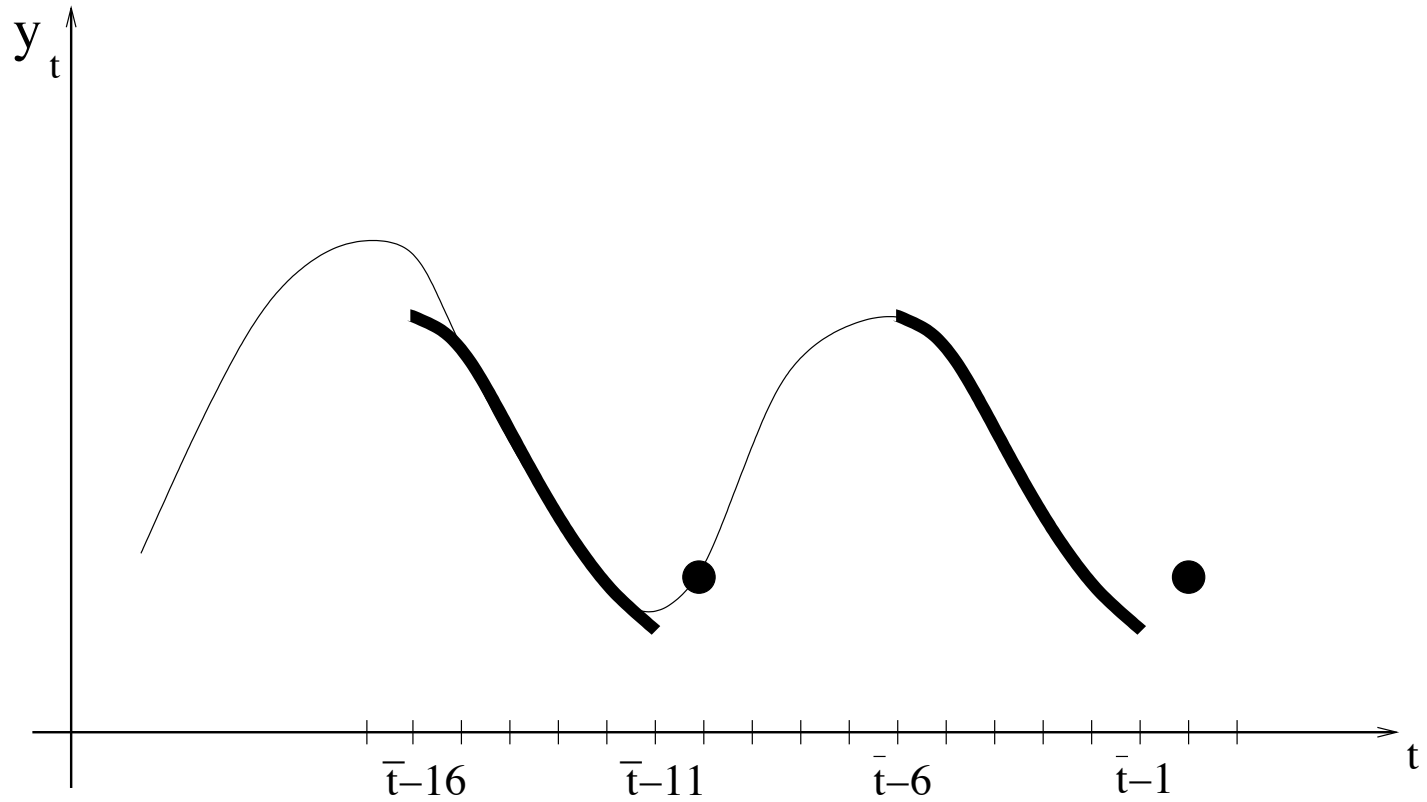
- Once a model of the embedding mapping is available, it can be used for two objectives: *one-step-ahead* prediction and *iterated* prediction.
- In one-step-ahead prediction, the n previous values of the series are available and the forecasting problem can be cast in the form of a generic regression problem
- In literature a number of supervised learning approaches have been used with success to perform one-step-ahead forecasting on the basis of historical data.

One step-ahead prediction



The approximator $\hat{f}$ returns the prediction of the value of the time series at time $t+1$ as a function of the n previous values (the rectangular box containing z^{-1} represents a unit delay operator, i.e., $\varphi^{t-1} = z^{-1}\varphi^{t}$).

Nearest-neighbor one-step-ahead forecasts



We want to predict at time $\bar{t}-1$ the next value of the series y of order $n=6$. The pattern $y_{\bar{t}-16}, y_{\bar{t}-15}, \ldots, y_{\bar{t}-11}$ is the most similar to the pattern $\{y_{\bar{t}-6}, y_{\bar{t}-5}, \ldots, y_{\hat{t}-1}\}$. Then, the prediction $\hat{y}_{\bar{t}} = y_{\bar{t}-10}$ is returned.

Multi-step ahead prediction

- The prediction of the value of a time series $H > 1$ steps ahead is called **H-step-ahead prediction.**
- We classify the methods for H-step-ahead prediction according to two features: the *horizon of the training criterion* and the single-output or multi-output nature of the predictor.

Multi-step ahead prediction strategies

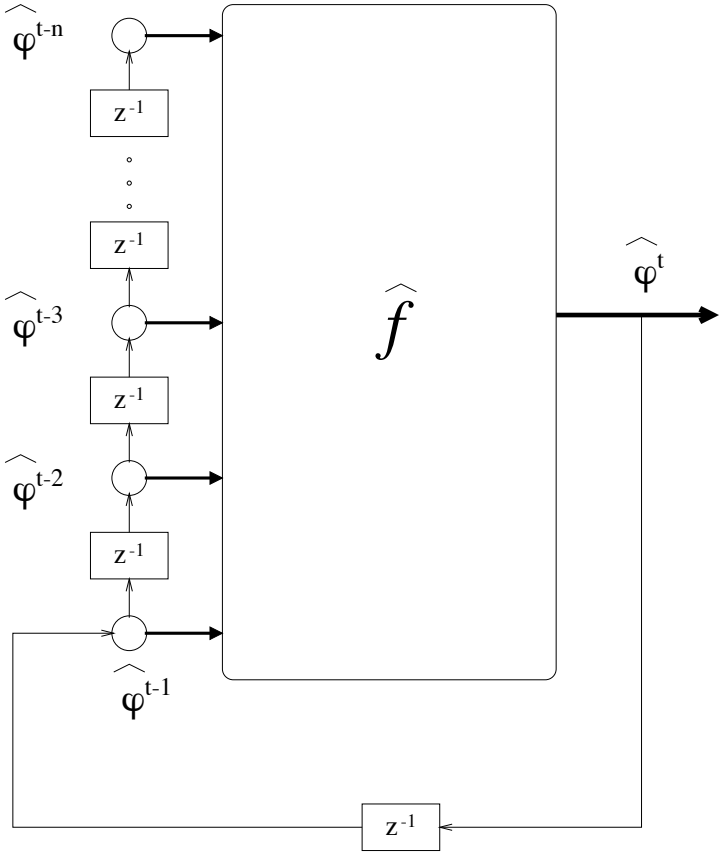
The most common strategies are

1. Iterated: the model predicts H steps ahead by iterating a one-step-ahead predictor whose parameters are optimized to minimize the training error on one-step-ahead forecast (one-step-ahead training criterion).
2. Iterated strategy where parameters are optimized to minimize the training error on the iterated h_{tr}-step-ahead forecast (h_{tr}-step-ahead training criterion) where $1 < h_{tr} \leq H$.
3. Direct: the model makes a direct forecast at time $t + h - 1, h = 1, \ldots, H$ by modeling the time series in a multi-input single-output form
4. Direc: direct forecast but the input vector is extended at each step with predicted values.
5. MIMO: the model returns a vectorial forecast by modeling the time series in a multi-input multi-output form

Iterated (or recursive) prediction

- In the case of iterated prediction, the predicted output is fed back as input for the next prediction.
- Here, the inputs consist of predicted values as opposed to actual observations of the original time series.
- As the feedback values are typically distorted by the errors made by the predictor in previous steps, the iterative procedure may produce undesired effects of accumulation of the error.
- Low performance is expected in long horizon tasks. This is due to the fact that they are essentially models tuned with a one-step-ahead criterion which is not capable of taking temporal behavior into account.

Iterated prediction

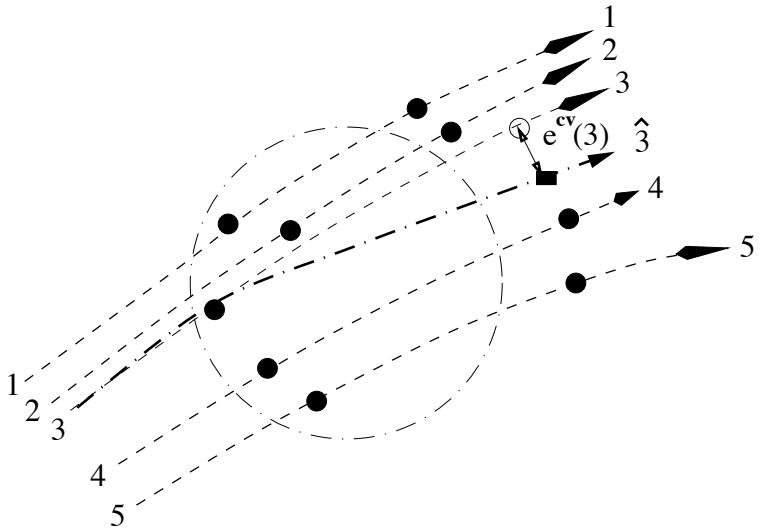


The approximator $\hat{f}$ returns the prediction of the value of the time series at time $t+1$ by iterating the predictions obtained in the previous steps (the rectangular box containing z^{-1} represents a unit delay operator, i.e., $\hat{\varphi}^{t-1} = z^{-1}\hat{\varphi}^{t}$).

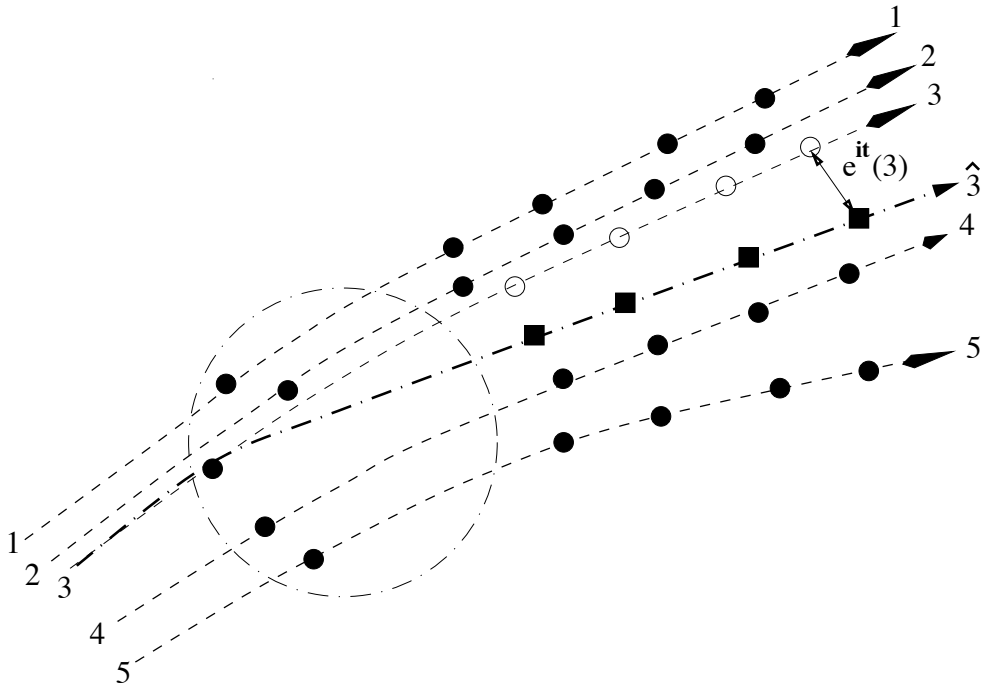
Iterated with h-step training criterion

- This strategy adopts one-step-ahead predictors but adapts the model selection criterion in order to take into account the multi-step-ahead objective.
- Methods like Recurrent Neural Networks belong to such class. Their recurrent architecture and the associated training algorithm (temporal backpropagation) are suitable to handle the time-dependent nature of the data.
- In [4] we proposed an adaptation of the Lazy Learning algorithm where the number of neighbors is optimized in order to minimize the leave-one-out error over an horizon larger than one. This technique ranked second in the 1998 KULeuven Time Series competition.
- A similar technique has been proposed by [14] who won the competition.

Conventional and iterated leave-one-out

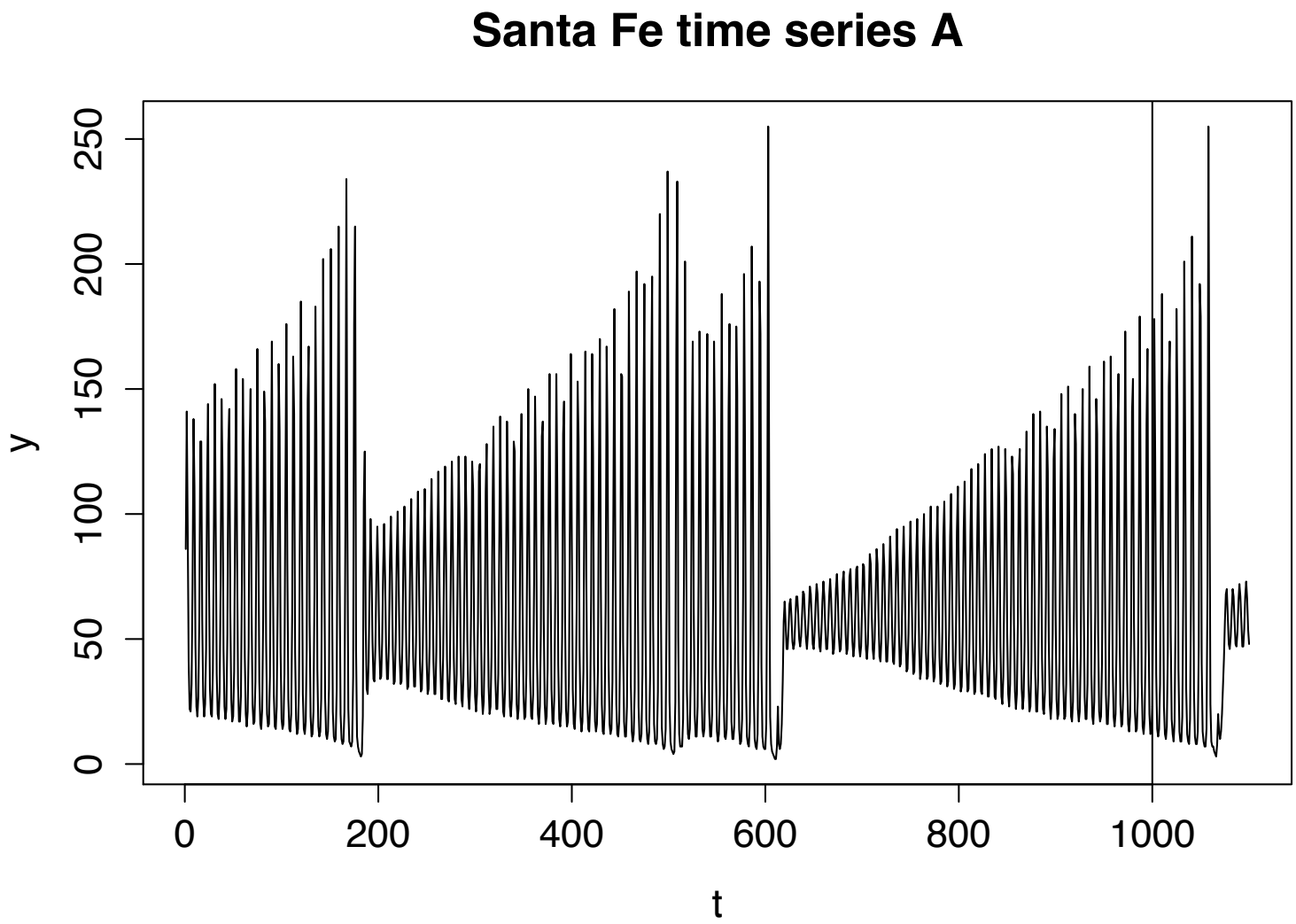


a)



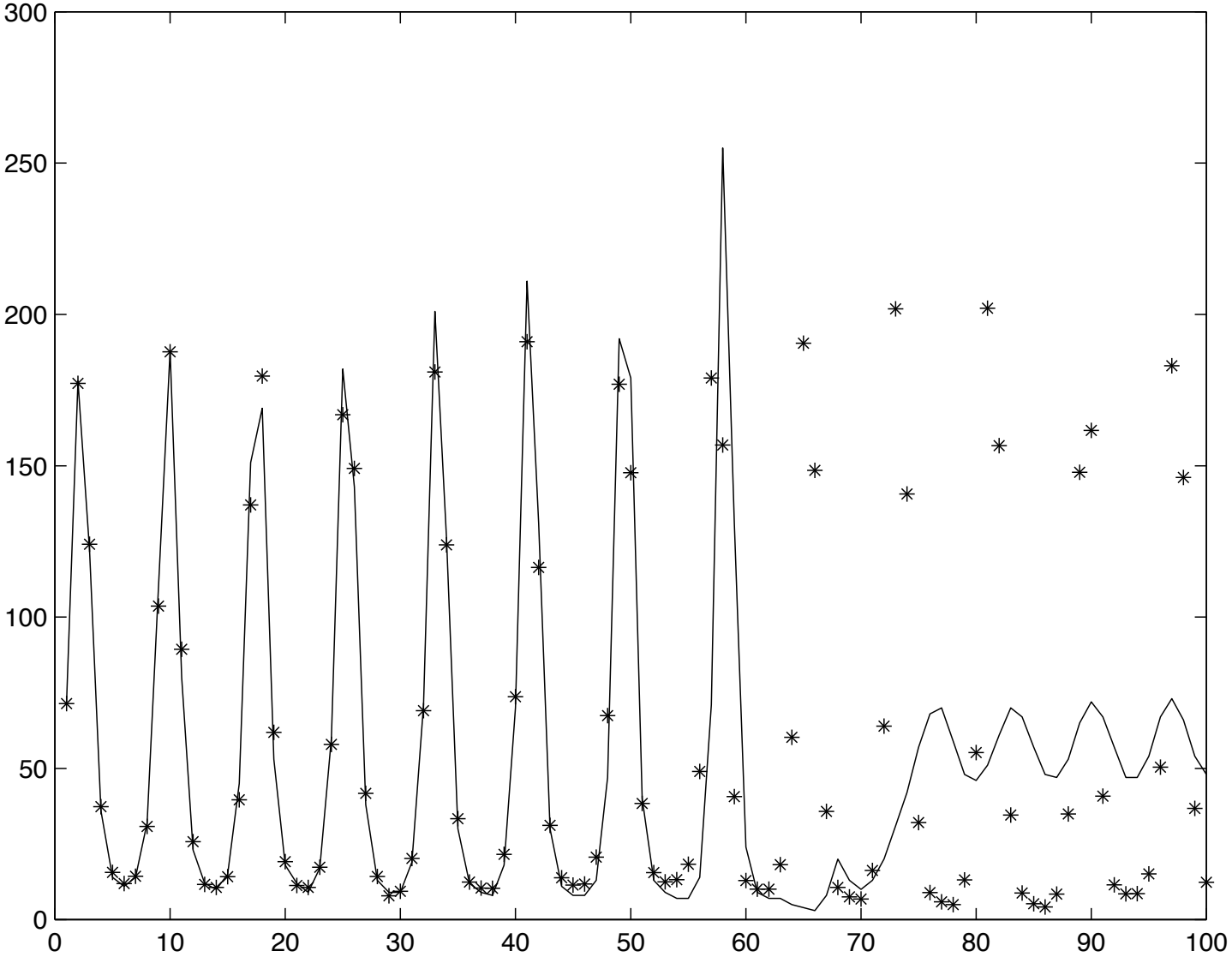
b)

Santa Fe time series A

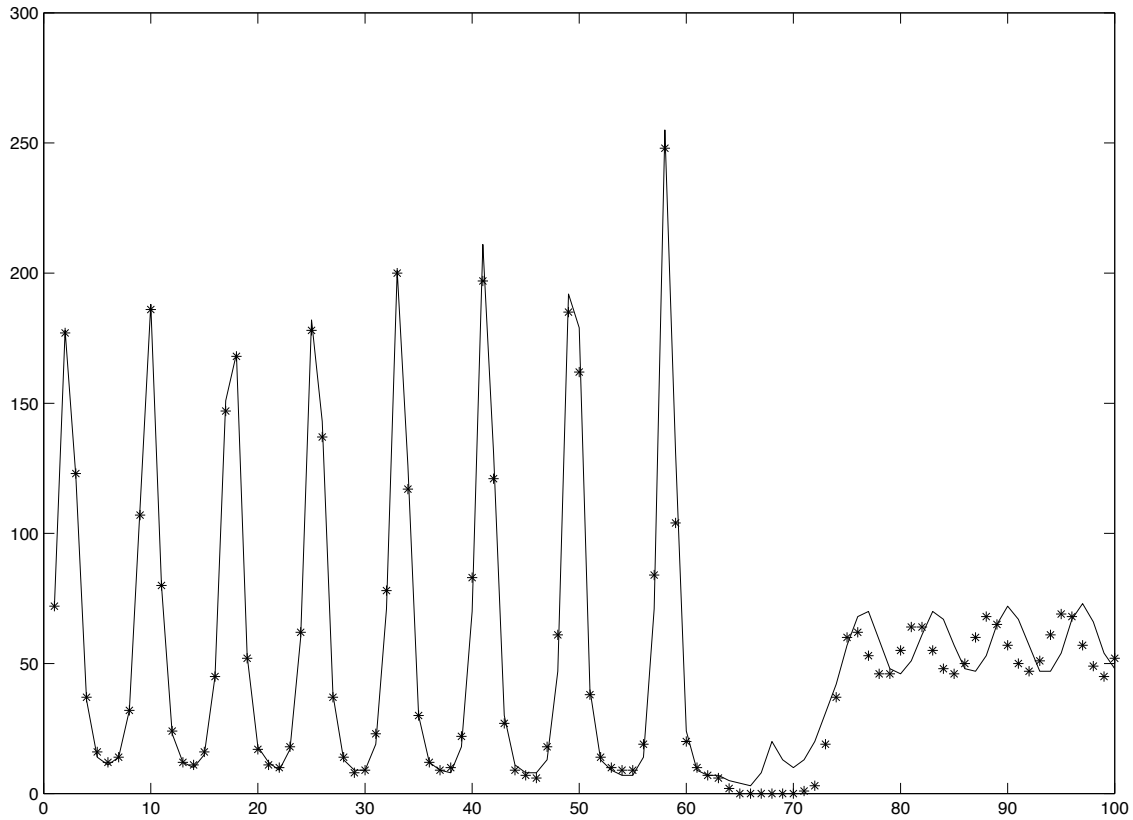


The *A* chaotic time series has a training set of 1000 values: the task is to predict the continuation for 100 steps, starting from different points.

One-step assessment criterion



Multi-step assessment criterion



Direct strategy

- The *Direct* strategy [22, 17, 7] learns independently H models f_h

$$\varphi_{t+h-1} = f_h(\varphi_{t-1}, \ldots, \varphi_{t-n}) + w_{t+h-1}$$

  with $h \in \{1, \ldots, H\}$ and returns a multi-step forecast by concatenating the H predictions.

- Several machine learning models have been used to implement the Direct strategy for multi-step forecasting tasks, for instance neural networks [10], nearest neighbors [17] and decision trees [21].
- Since the Direct strategy does not use any approximated values to compute the forecasts, it is not prone to any accumulation of errors, since each model is tailored for the horizon it is supposed to predict.
- Notwithstanding, it has some weaknesses.

Direct strategy limitations

- Since the H models are learned independently no statistical dependencies between the predictions $\hat{\varphi}_{t+h-1}, h = 1, \ldots, H$ [3, 5, 10] is guaranteed.
- Direct methods often require higher functional complexity [20] than iterated ones in order to model the stochastic dependency between two series values at two distant instants [9].
- This strategy demands a large computational time since the number of models to learn is equal to the size of the horizon.

DirRec strategy

- The *DirRec* strategy [18] combines the architectures and the principles underlying the Direct and the Recursive strategies.
- DirRec computes the forecasts with different models for every horizon (like the Direct strategy) and, at each time step, it enlarges the set of inputs by adding variables corresponding to the forecasts of the previous step (like the Recursive strategy).
- Unlike the previous strategies, the embedding size n is not the same for all the horizons. In other terms, the DirRec strategy learns H models f_h from the time series where

$$\varphi_{t+h-1} = f_h(\varphi_{t+h-1}, \ldots, \varphi_{t-n}) + w_{t+h-1}$$

  with $h \in \{1, \ldots, H\}$.
- The technique is prone to the curse of dimensionality. The use of feature selection is recommended for large h.

MIMO strategy

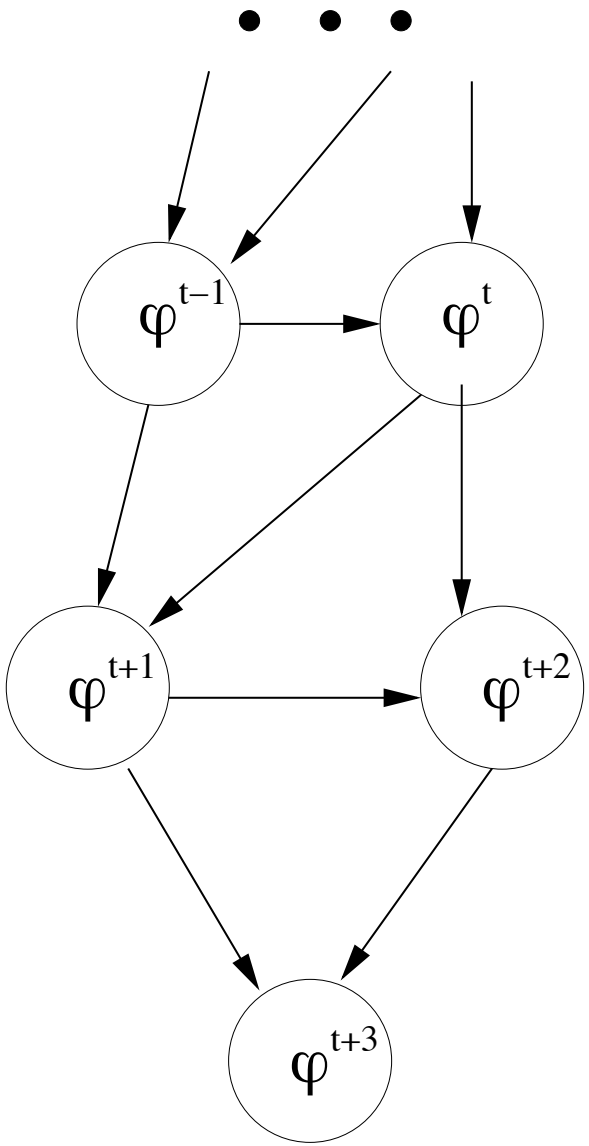
- This strategy [3, 5] (also known as Joint strategy [10]) avoids the simplistic assumption of conditional independence between future values made by the Direct strategy [3, 5] by learning a single multiple-output model

$$[\varphi_{t+H-1}, \ldots, \varphi_t] = F(\varphi_{t-1}, \ldots, \varphi_{t-n}) + \mathbf{w}$$

  where $F : \mathbb{R}^d \rightarrow \mathbb{R}^H$ is a vector-valued function [15], and $\mathbf{w} \in \mathbb{R}^H$ is a noise vector with a covariance that is not necessarily diagonal [13].

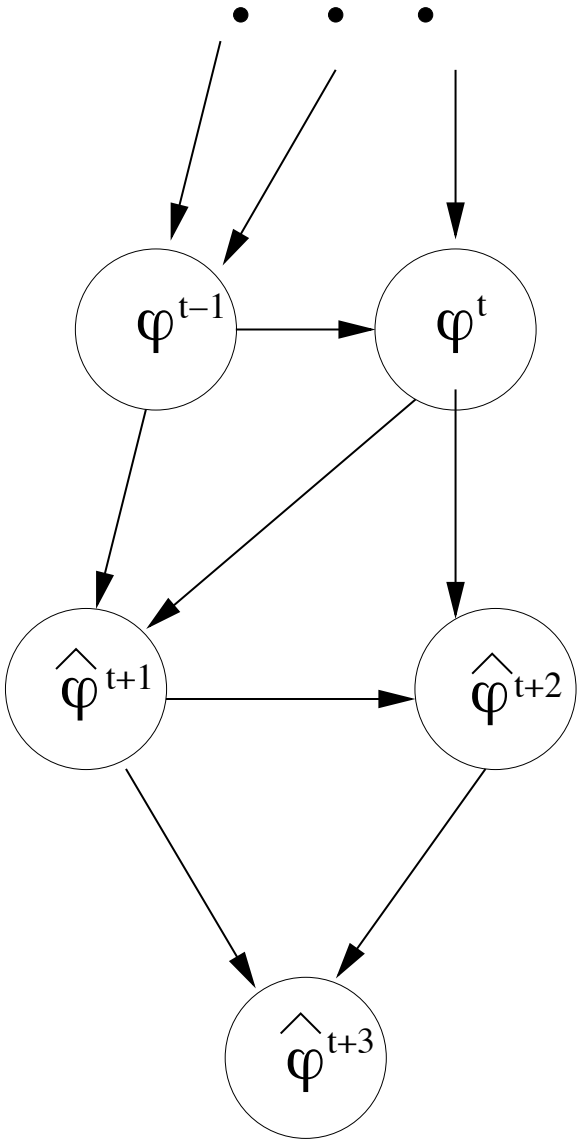
- The forecasts are returned in one step by a multiple-input multiple-output regression model.
- In [5] we proposed a multi-output extension of the local learning algorithm.
- Other multi-output regression model could be taken into consideration like multi-output neural networks or partial least squares.

Time series dependencies

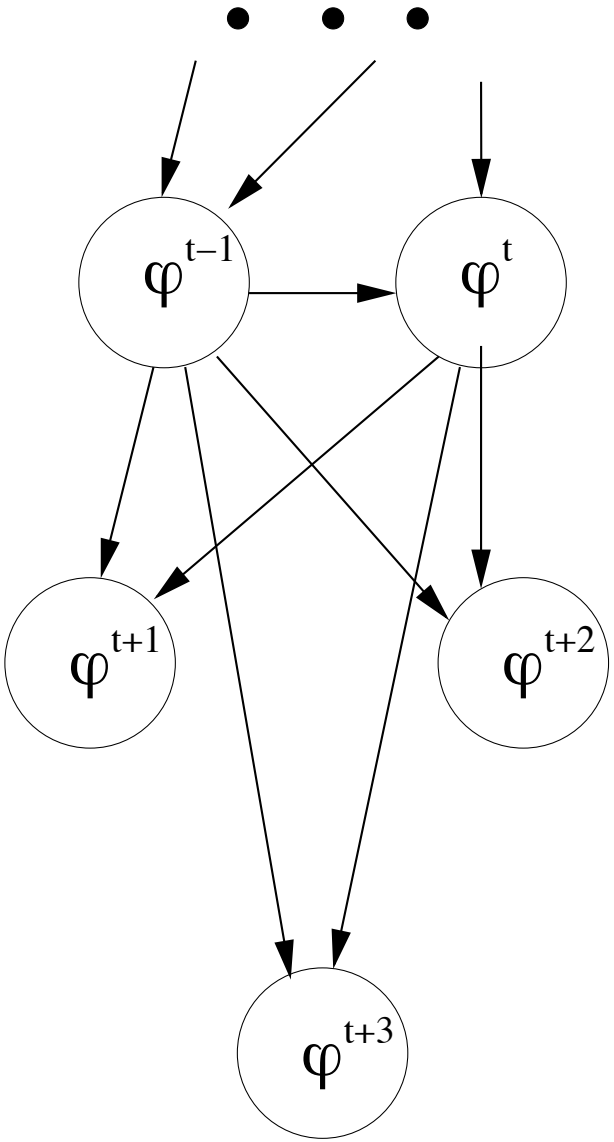


$n = 2$ NAR dependency $\varphi_t = f(\varphi_{t-1}, \varphi_{t-2}) + w(t)$.

Iterated modeling of dependencies



Direct modeling of dependencies



MIMO strategy

- The rationale of the MIMO strategy is to model, between the predicted values, the stochastic dependency characterizing the time series. This strategy avoids the conditional independence assumption made by the Direct strategy as well as the accumulation of errors which plagues the Recursive strategy.
- So far, this strategy has been successfully applied to several real-world multi-step time series forecasting tasks [3, 5, 19, 2].
- However, the wish to preserve the stochastic dependencies constrains all the horizons to be forecasted with the same model structure. Since this constraint could reduce the flexibility of the forecasting approach [19], a variant of the MIMO strategy (called DIRMO) has been proposed in [19, 2] .
- Extensive validation on the 111 times series of the NN5 competition showed that MIMO are invariably better than single-output approaches.

Validation of time series methods

- The huge variety of strategies and algorithms that can be used to infer a predictor from observed data asks for a rigorous procedure of comparison and assessment.
- Assessment demands benchmarks and benchmarking procedure.
- Benchmarks can be defined by using
  - Simulated data obtained by simulating AR, NAR and other stochastic processes. This is particular useful for validating theoretical properties in terms of bias/variance.
  - Public domain benchmarks, like the one provided by Time Series Competitions.
  - Real measured data

Competitions

- Santa Fe Time Series Prediction and Analysis Competition (1994) [22]:
- International Workshop on Advanced Black-box techniques for nonlinear modeling Competition (Leuven, Belgium; 1998)
- NN3 competition [8]: 111 monthly time series drawn from homogeneous population of empirical business time series.
- NN5 competition [1]: 111 time series of the daily retirement amounts from independent cash machines at different, randomly selected locations across England.
- Kaggle competition.

Accuracy measures

Let

$$e_{t+h} = \varphi_{t+h} - \hat{\varphi}_{t+h}$$

represent the error of the forecast $\hat{\varphi}_{t+h}$ at the horizon $h = 1, \ldots, H$. A conventional measure of accuracy is the Normalized Mean Squared Error

$$\text{NMSE} = \frac{\sum_{h=1}^{H}(\varphi_{t+h} - \hat{\varphi}_{t+h})^2}{\sum_{h=1}^{H}(\varphi_{t+h} - \hat{\mu})^2}$$

This quantity is smaller than one if the predictor performs better than the naivest predictor, i.e. the average $\hat{\mu}$.
Other measures rely on relative or percentage error

$$\text{pe}_{t+h} = 100\frac{\varphi_{t+h} - \hat{\varphi}_{t+h}}{\varphi_{t+h}}$$

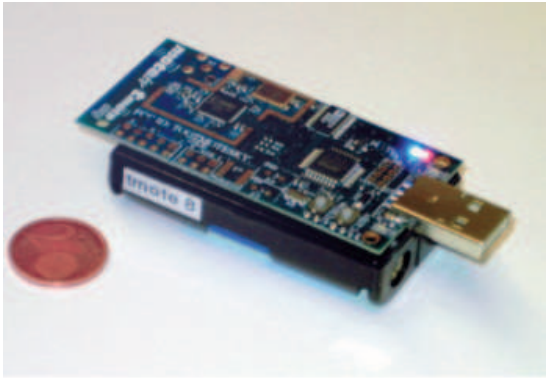
like

$$\text{MAPE} = \frac{\sum_{h=1}^{H}|\text{pe}_{t+h}|}{H}$$

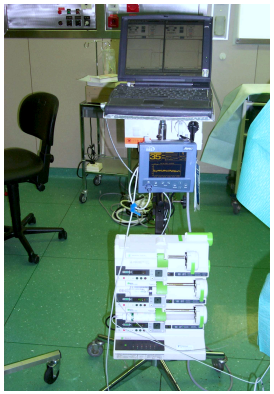
Applications in my lab

MLG projects on forecasting

- Low-energy streaming of wireless sensor data



- Decision support in anesthesia

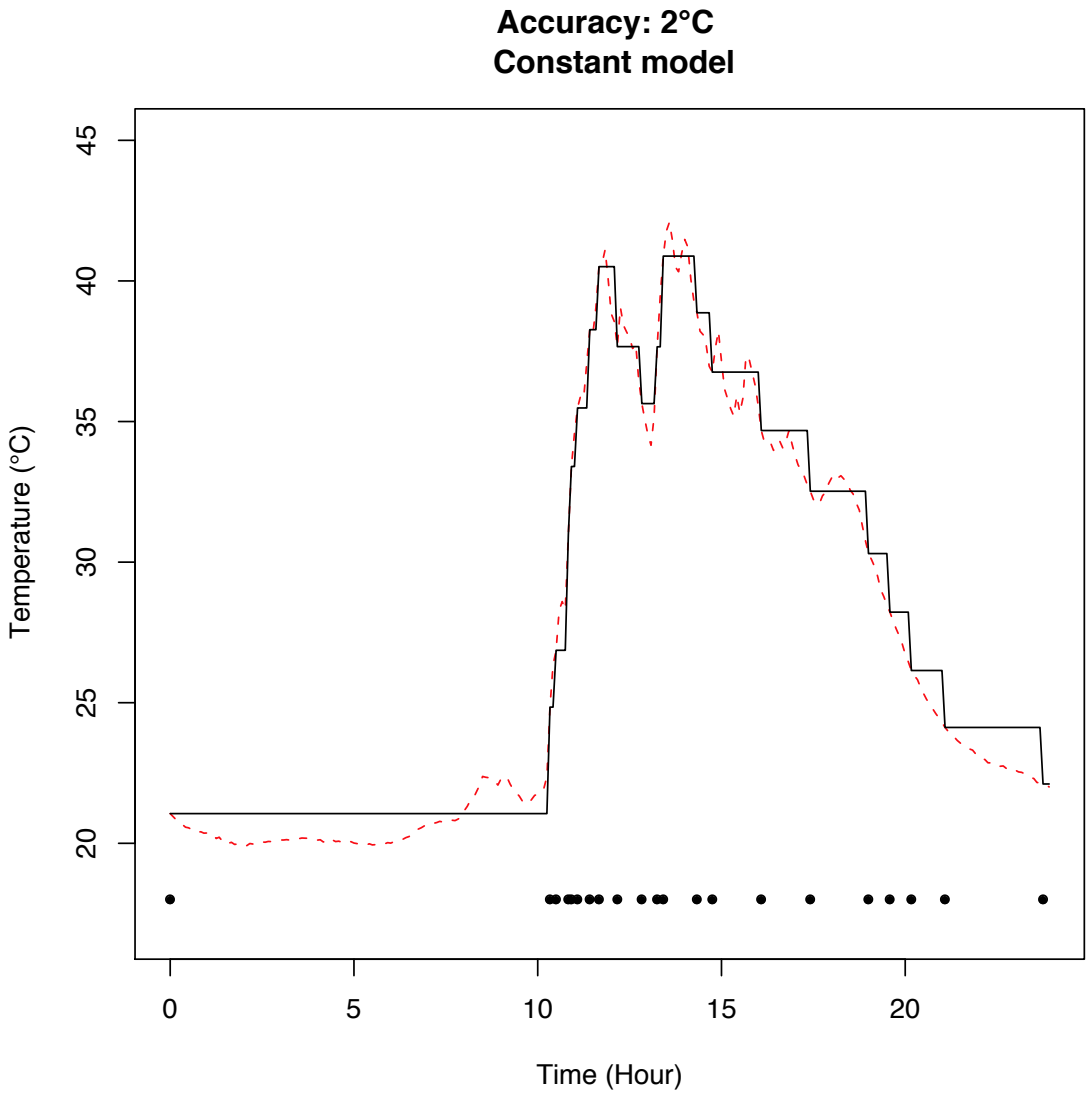


- Side-channel attack

Low-energy streaming of wireless sensor data

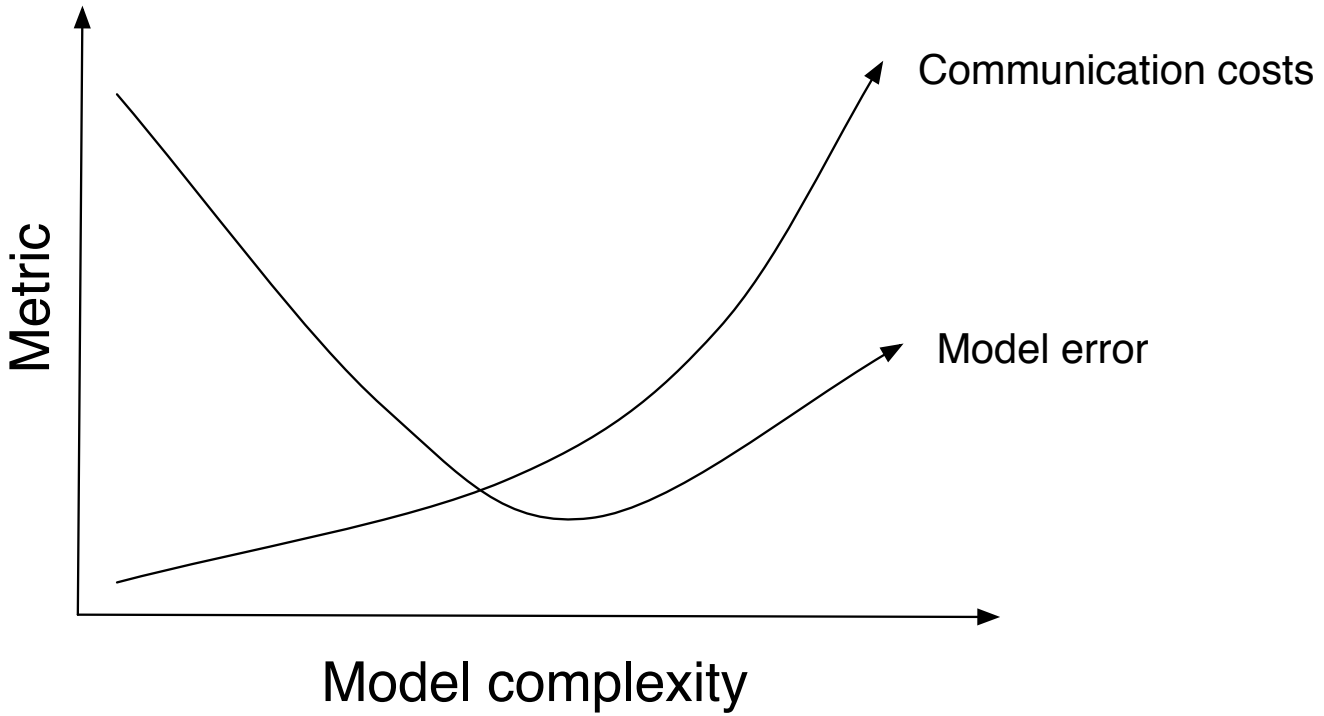
- In monitoring applications, only an approximation to sensor readings is sufficient (e.g. $\pm 0.5C$, $\pm 2\%$ humidity, ..). In this context a Dual Prediction Scheme is effective.
- A sensor node is provided with a time series prediction model $\varphi_t = f(\varphi_{t-1}, \alpha)$ (e.g. autoregressive models) and a learning method for identifying the best set of parameters α (e.g. recursive least squares).
- The sensor node then sends the parameters of the model instead of the actual data to the recipient. The recipient node then runs the models to reconstruct an approximation of the data stream collected on the distant node.
- The sensor node also runs the prediction model. When its predictions diverges by more than $\pm\epsilon$ from the actual reading, a new model is sent to the recipient.
- This allows to reduce the communication effort if an appropriate model is run by the sensor node.
- Since the cost of transmission is much larger than the cost of computation, in realistic situations this scheme allows economy of power consumption.

Low-energy streaming of wireless sensor data



Adaptive model selection

Tradeoff: more complex models predict better measurements but have a higher number of parameters



$$\mathrm{AR(p)} : \hat{s}_i[t] = \sum_{j=1}^{p} \theta_j s_i[t-j]$$

Adaptive Model Selection

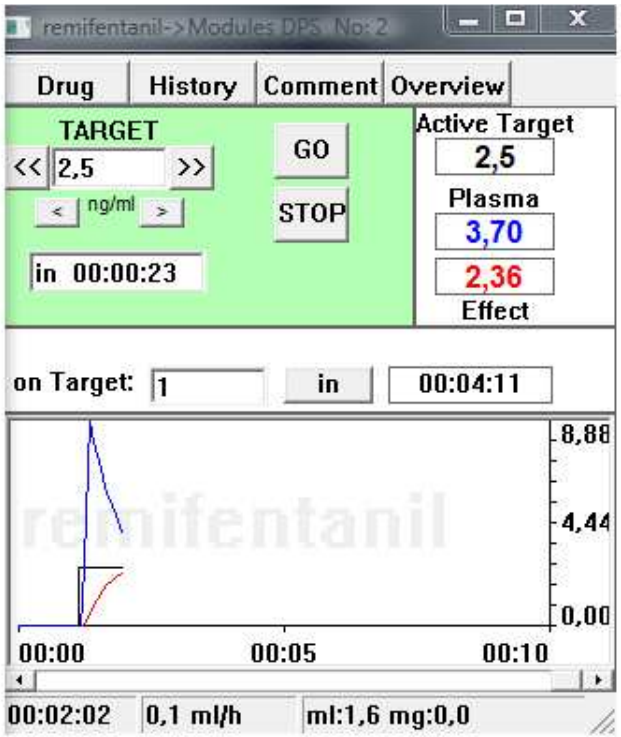
We proposed an Adaptive Model Selection strategy [11] that

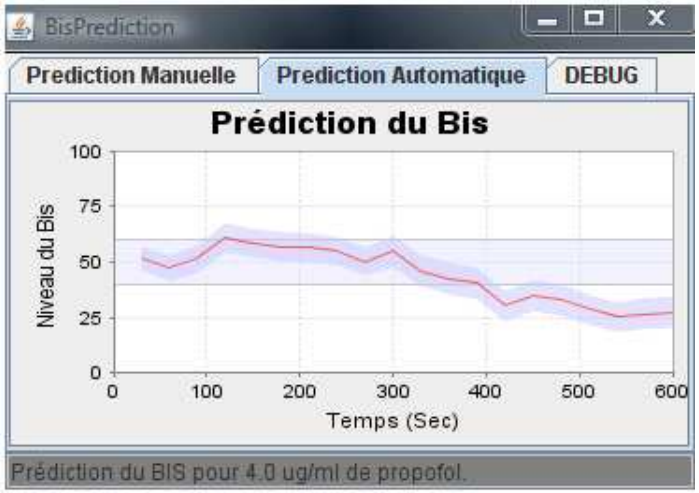
- takes into account the cost of sending model parameters,
- allows sensor nodes to determine autonomously the model which best fits their measurements,
- provides a statistically sound selection mechanism to discard poorly performing models,
- gave in experimental results about 45% of communication savings on average,
- was implemented in TinyOS, the reference operating system.

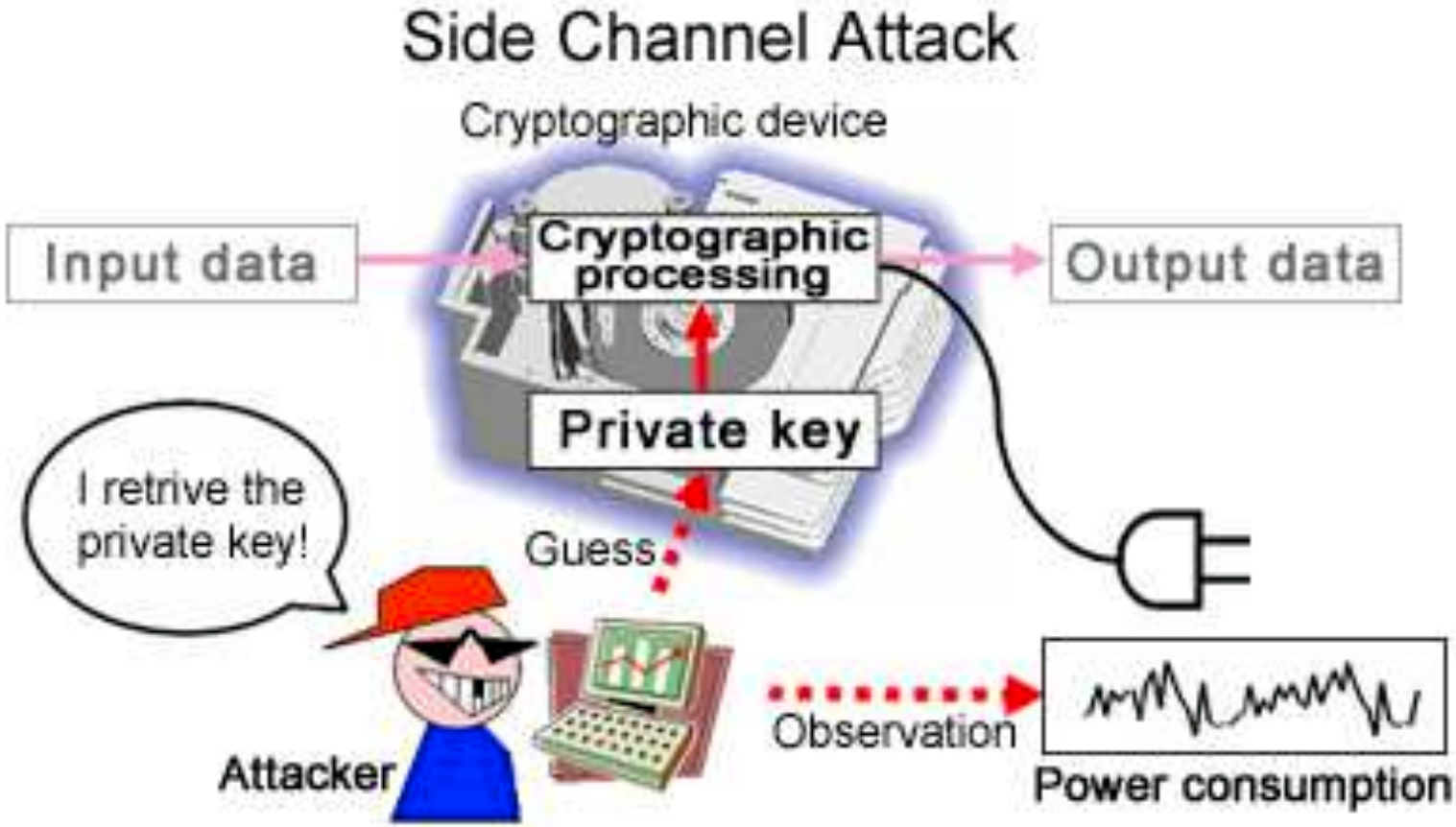
Predictive modeling in anesthesiology

- During surgery, anesthesiologists controls the depth of anesthesia by means of types of drugs
- Anesthesiologists observe the patient state by observing unconsciousness signal in real-time which are collected by monitors connected via electrodes to the patient's forehead
- The bispectral index BIS monitor (by Aspect Medical Systems) is a single dimensionless number between 0 to 100 where 0 equals EEG silence, 100 is the expected value for a fully awake adult, and between 40 and 60 indicates a recommended level.
- It remains difficult for the anesthesiologist, especially if inexperienced, to predict how the BIS signal could vary after a change in the administered anesthetic agents. This is generally due to inter-individual variability problem
- We designed a ML system [6] to predict multi-step-ahead the evolution of the BIS on the basis of historical data.

Predictive modeling in anesthesiology






Side Channel Attack
Cryptographic device
Input data
Cryptographic processing
Output data
Private key
I retrive the private key!
Guess
Attacker
Observation
Power consumption

Side channel attack

- In cryptography, a side channel attack is any attack based on the analysis of measurements related to the physical implementation of a cryptosystem.
- Side channel attacks take advantage of the fact that instantaneous power consumption, encryption time or/and electromagnetic leaks of a cryptographic device depend on the processed data and the performed operations.
- Power analysis attacks are an instance of side-channel attacks which assume that different encryption keys imply a different power consumptions.
- Nowadays, the possibility of collecting a large amount of power traces (i.e. time series) paves the way to the adoption of machine learning techniques.
- Effective side channel attacks enables effective countermeasures.

SCA and time series classification

- The power consumption of a crypto device using a secret key $Q \in \{0,1\}^k$ (size $k = 8$) can be modeled as a time series $T^{(Q)}$ of order n

$$T^{(Q)}_{(t+1)} = f(T^{(Q)}_{(t)}, T^{(Q)}_{(t-1)}, ..., T^{(Q)}_{(t-n)}) + \epsilon$$

- For each key Q_j we infer a predictive model $\hat{f}$ [12] such that

$$T^{(Q_j)}_{(t+1)} = \hat{f}_{Q_j}(T^{(Q_j)}_{(t)}, T^{(Q_j)}_{(t-1)}, ..., T^{(Q_j)}_{(t-n)}) + \epsilon$$

- These models can be used to classify an unlabeled time series T and predict the associated key by computing a distance for each Q_j

$$\mathsf{D}(Q_j, T) = \frac{1}{N-n+1} \sum_{t=n}^{N} \left(\hat{f}_{Q_j} \left(T_{(t-1)}, T_{(t-2)}, ..., T_{(t-n-1)} \right) - T_{(t)} \right)^2$$

and choosing the key minimizing it

$$\hat{Q} = \arg \min_{j \in [0, 2^{(k-1)}]} \mathsf{D}(Q_j, T)$$

Conclusions

Open-source software

Many commercial solutions exist but only open-source software can cope with

- fast integration of new algorithms
- portability over several platforms
- new paradigms of data storage (e.g. Hadoop)
- integration with different data formats and architectures

A de-facto standard in computational statistics, machine learning, bioinformatics, geostatistics or more general analytics is nowadays

Highly recommended !

All that we didn't have time to discuss

- ARIMA models
- GARCH models
- Frequence space representation
- Nonstationarity
- Vectorial time series
- Spatio temporal time series
- Time series classification

Suggestions for PhD topics

- Time series and big data
  - streaming data (environmental data)
  - large vectorial time series (weather data)
- Spatio-temporal time series and graphical models
- Beyond cross-validation for model/input selection
- Long term forecasting (effective integration of iterated and directed models)
- Causality and time-series
- Scalable machine learning

Suggestion: use methods and models to solve problems... not problems to sanctify methods or models...

Conclusion

- Popper claimed that, if a theory is falsifiable (i.e. it can be contradicted by an observation or the outcome of a physical experiment.), then it is scientific. Since prediction is the most falsifiable aspect of science it is also the most scientific one.
- Effective machine learning is an extension of statistics, in no way an alternative.
- Simplest (i.e. linear) model first.
- Local learning techniques represent an effective trade-off between linearity and nonlinearity.
- Modelling is more an art than an automatic process... then experience data analysts are more valuable than expensive tools.
- Expert knowledge matters..., data too
- Understanding what is predictable is as important as trying to predict it.

Forecasting is difficult, especially the future

- "Computers in the future may weigh no more than 1.5 tons." –Popular Mechanics, forecasting the relentless march of science, 1949
- "I think there is a world market for maybe five computers." –Chairman of IBM, 1943
- "Stocks have reached what looks like a permanently high plateau." –Professor of Economics, Yale University, 1929.
- "Airplanes are interesting toys but of no military value." –Professor of Strategy, Ecole Superieure de Guerre.
- "Everything that can be invented has been invented." –Commissioner, U.S. Office of Patents, 1899.
- "Louis Pasteur's theory of germs is ridiculous fiction". –Professor of Physiology at Toulouse, 1872
- "640K ought to be enough for anybody." – Bill Gates, 1981

References

[1] Robert R. Andrawis, Amir F. Atiya, and Hisham El-Shishiny. Forecast combinations of computational intelligence and linear models for the NN5 time series forecasting competition. *International Journal of Forecasting*, January 2011.

[2] S. Ben Taieb, A. Sorjamaa, and G. Bontempi. Multiple-output modelling for multi-step-ahead forecasting. *Neurocomputing*, 73:1950–1957, 2010.

[3] 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*, pages 145–154, Helsinki, Finland, February 2008.

[4] G. Bontempi, M. Birattari, and H. Bersini. Local learning for iterated time-series prediction. In I. Bratko and S. Dzeroski, editors, *Machine Learning: Proceedings of the Sixteenth International Conference*, pages 32–38, San Francisco, CA, 1999. Morgan Kaufmann Publishers.

[5] G. Bontempi and S. Ben Taieb. Conditionally dependent strategies for multiple-step-ahead prediction in local learn-

ing. *International Journal of Forecasting*, 27(3):689–699, 2011.

[6] Olivier Caelen, Gianluca Bontempi, and Luc Barvais. Machine learning techniques for decision support in anesthesia. In *Artificial Intelligence in Medicine*, pages 165–169. Springer Berlin Heidelberg, 2007.

[7] Haibin Cheng, Pang-Ning Tan, Jing Gao, and Jerry Scripps. Multistep-ahead time series prediction. In *PAKDD*, pages 765–774, 2006.

[8] Sven F. Crone, Michèle Hibon, and Konstantinos Nikolopoulos. Advances in forecasting with neural networks? empirical evidence from the nn3 competition on time series prediction. *International Journal of Forecasting*, 27, 2011.

[9] M. Guo, Z. Bai, and H.Z. An. Multi-step prediction for nonlinear autoregressive models based on empirical distributions. *Statistica Sinica*, pages 559–570, 1999.

[10] D. M. Kline. Methods for multi-step time series forecasting with neural networks. In G. Peter Zhang, editor, *Neural Networks in Business Forecasting*, pages 226–250. Information Science Publishing, 2004.

[11] Yann-Aël Le Borgne, Silvia Santini, and Gianluca Bontempi. Adaptive model selection for time series prediction in wireless sensor networks. *Signal Processing*, 87(12):3010–3020, 2007.

[12] Liran Lerman, Gianluca Bontempi, Souhaib Ben Taieb, and Olivier Markowitch. A time series approach for profiling attack. In *SPACE*, pages 75–94, 2013.

[13] José M. Matías. Multi-output nonparametric regression. In Carlos Bento, Amílcar Cardoso, and Gaël Dias, editors, *EPIA*, volume 3808 of *Lecture Notes in Computer Science*, pages 288–292. Springer, 2005.

[14] J. McNames. A nearest trajectory strategy for time series prediction. In *Proceedings of the InternationalWorkshop on Advanced Black-Box Techniques for Nonlinear Modeling*, pages 112–128, Belgium, 1998. K.U. Leuven.

[15] Charles A. Micchelli and Massimiliano A. Pontil. On learning vector-valued functions. *Neural Comput.*, 17(1):177–204, 2005.

[16] T. M. Mitchell. *Machine Learning*. McGraw Hill, 1997.

[17] A. Sorjamaa, J. Hao, N. Reyhani, Y. Ji, and A. Lendasse. Methodology for long-term prediction of time series. *Neurocomputing*, 70(16-18):2861–2869, October 2007.

[18] A. Sorjamaa and A. Lendasse. Time series prediction using dirrec strategy. In M. Verleysen, editor, *ESANN06, European Symposium on Artificial Neural Networks*, pages 143–148, Bruges, Belgium, April 26-28 2006. European Symposium on Artificial Neural Networks.

[19] Souhaib Ben Taieb, Gianluca Bontempi, Antti Sorjamaa, and Amaury Lendasse. Long-term prediction of time series by combining direct and mimo strategies. *International Joint Conference on Neural Networks*, 2009.

[20] H. Tong. *Threshold models in Nonlinear Time Series Analysis*. Springer Verlag, Berlin, 1983.

[21] Van Tung Tran, Bo-Suk Yang, and Andy Chit Chiow Tan. Multi-step ahead direct prediction for the machine condition prognosis using regression trees and neuro-fuzzy systems. *Expert Syst. Appl.*, 36(5):9378–9387, 2009.

[22] A.S. Weigend and N.A. Gershenfeld. *Time Series Prediction: forecasting the future and understanding the past*. Addison Wesley, Harlow, UK, 1994.