Metocean Time Series (METIS) Toolbox Documentation

Valérie Monbet, Pierre Ailliot
IFREMER, TMSI/RED/HO, BP 70, F-29280 Plouzané

10th August 2005

Contents

Chapter 1

Introduction

1.1 What is METIS?

METIS (MEtocean Time Series) is a toolbox of Matlab routines for analysis and simulation of stochastic processes particularly adapted for sea state processes. The major task of METIS are the estimation of stochastic Markov models for uni or bivariate processes, simulation of time series given a chosen model and statistical comparison of the generated time series to a reference time series. For instance, METIS allows to fit several models given an observed wind time series and then to use these models to generate new wind time series with statistical properties closed to those of the observed time series. Then some routines permit to compare the signals.

METIS has a modular structure driven by the proposed models. The modules of the toolbox handle

- Translated Gaussian Process model
- Markov Switching Auto Regressive model
- Filtering and smoothing
- Validation

The toolbox has been developped with Matlab 6.5 Release 13.

1.2 Organization of METIS

In this section, we make a brief presentation of each module. But first of all, it is important to remark that some of the METIS toolbox call routines of WAFO *(http://www.maths.lth.se/matstat/wafo/)* and HMM toolbox *http://www.ai.mit.edu/ murphyk/Software/HMM/*.

Translated Gaussian Process

The Translated Gaussian Process (TGP) module gathers all the routines needed for the simulation of bivariate process based on the transformation into a Gaussian process. In this method, the observed time series is first transformed in a time series which marginal distribution is approximately gaussian. Then, this new time series is assumed to be a realization of a Gaussian process, and new realizations of this process are simulated using an exact simulation method.

Markov Switching Auto Regressive model

In this module you will find the routines for estimation and simulation of Markov Switching Auto Regressive (MSAR) model. Several models are proposed:

- homogeneous MSAR with Gaussian innovation
- homogeneous MSAR with Gamma or lognormal innovation
- non homogeneous MSAR with Gamma innovation

Filtering

Non parametric filtering can be used to predict a component of a multivariate process given the other components if the process is completly observed during a sufficiently long period. For instance filtering can be used to predict significant wave height given observed wind.

Validation

A set of routines as been developped in order to validate the simulation models considering that the genrated time series should have the same statistical properties as the reference time series. Graphical tools and statistical tests are proposed.

Demonstrations

For each module, a demonstration program is provide. For this, we have used hindcast data, produced by Oceanweather, for a point of coordinates (46.25 N,1.67 E) located near the French Atlantic coast. It consists in 22 years of data, with a record every 6 hours. We concentrate our study on the month of January and we will assume, as it is usual for meteorological time series, that the 22 months of January available in our database are independant realizations of a stationary process.

Chapter 2

Translated Gaussian Process

The idea of the Translated Gaussian Process, also refered as TGP, is to transform the observed time series into a realization that can be assumed to be Gaussian. Several transformations can be applied. In Box and Jenkins's approach (Box, 1956; Brown, 1984), a Box-Cox transformation is used and the obtained time series is then modelled by an autoregressive process. Here, a slightly different method is used.

This method has already been used by (55) and (42) to simulate sea-state parameters and by (23) to simulate wind pressure fluctuations on a building (see also reference therein).

2.1 Model description

Let $\{Y_t\}$ be a stationary process with values in $\mathbf{R}^d$. The method described in this first part assumes that there exists a one to one mapping $f : \mathbf{R}^d \to \mathbf{R}^d$ and a stationary Gaussian process $\{X_t\}$ such that $Y_t = f(X_t)$. This procedure has three main steps:

- Model calibration, which consists in determining the function f and the second order structure of the process $\{X_t\}$.
- Sample generation in which realizations of the process $\{X_t\}$ are simulated given the second order structure estimated in model calibration step. There exist several algorithms to simulate realizations of stationary Gaussian processes given the second order structure. In this work, the method described by Scheffner (1992) is used.
- Mapping. In this step the generated samples of $\{X_t\}$ are transformed into samples of $\{Y_t\}$ using the transformation f.

In order to estimate the second order structure of the process, the empirical covariance function of $\hat{g}(y_t)$ is calculated where y_t represents a realization of $\{Y_t\}$ and $\hat{g}$ the empirical estimate of g. Other calibration methods are proposed for instance in papers of Gioffré (2000) but they are not implemented in this version of the toolbox.

2.2 Example: wind simulation

In this example, we consider that $\{Y_t\}$ is the bivariate process (U_t, Φ_t) with U the wind intensity and Φ the wind direction.

We first focus on the choice of the mapping f. In fact, we describe the inverse transformation $g = f^{-1}$ which is chosen such that the marginal distribution of the process $\{g(Y_t)\}$ is approximately Gaussian. Generally g consists in normal-score transformations applied independently on the d components, that is $g(y_1, ..., y_d) = (g_1(y_1), ..., g_d(y_d))$ with $g_k(y_k) = \Phi^{-1} \circ F_{Y_k}(y_k)$ where Φ and

F_{Y_k} represent respectively the distribution function of the standard Gaussian and the marginal distribution function of the k^{th} component of $\{Y_t\}$. As shown in Monbet (2001a), the transformation does not work when there exists a complex dependence between the different components of $\{Y_t\}$. On figure 2.1 the marginal distribution of the process $\{u_t, v_t\}$, the zonal and meridional components of the wind, is plotted. We can see that this distribution is bimodal, each mode corresponding to a different weather type and that there are few values around the origin. This feature can not be caught by the above transformation.

The "conditional" normal score transformation, also refered as Rozenblatt trnsformation and proposed in (42), has been adapted for the bivariate process $Y_t = (U_t, \Phi_t)$. We propose the transformation $g = g_2 \circ g_1$ with

$$g_1 : (u, \phi) \to (F_\Phi(\phi), R^{-1} \circ F_{U|\Phi=\phi}(u))$$
$$g_2 : (u, \phi) \to (u\cos(2\pi\phi), u\sin(2\pi\phi))$$

where F_Φ, R and $F_{U|\Phi=\phi}$ denote respectively the distribution functions of the marginal distribution of $\{\Phi_t\}$, the Rayleigh distribution and the conditional distribution of Φ_t given $U_t = a$. It is easy to check that the marginal distribution of process $\{g(Y_t)\}$ is Gaussian with zero mean and identity covariance matrix. In practice, the distribution functions which appear in the definition of g are estimated using the usual empirical estimates.

```
[ro,phi] = rozenblat(U,Phi,20,'rayleigh','uniform') ;
[u,v] = pol2cart(2*pi*phi,ro);
```

Now, u and v are supposed to be Gaussian processes. In function simgausspr.m the autocovariance structure of (u,v) is computed and used as reference to generate a new Gaussian process (ug,vg) with the same second order properties.

```
Nsim = 100;
[ug,vg] = simgausspr(u,v,1,Nsim,1);
```

And the inverse Rozenblatt transformation is computed.

```
[phig,rog] = cart2pol(ug,vg);
[Ug,Phig] = invrozenblat(U(:),Phi(:),rog(:),mod(phig(:),2*pi)/2/pi,20,'rayleigh','uniform');
```

Finally statiscal properties of simulated time series (Ug,Phig) can be compared to the statistical properties of the reference time series (U,Phi). For more detail about this task please read below the Validation chapter.

```
graph_valid_bivar(U,Ug,Phi,Phig,0.05,1) ;
```

The validation function returns figures such as Figures 2.1 to 2.2 and also the pvalues of the tests described hereafter in chapter 5. Figure 2.1 shows that the histograms of the observed wind time series (solid line) and the histogram of the generated time series (dashed line) match. It is not surprising because the simulation TGP is built in order that the marginal distributions of the observation and the simulated time series are the same. Now, on Figure 2.2 distributions of time duration of sejourn over (left) and under (right) level $\frac{2}{3}max(Y)$ are plotted and it is clear that TGP fails to correctly restore the distribution of time duration of sejourn under. It is

The command graph_valid_bivar returns also graphics for joint statistics of the intensity and the direction of wind. As example, Figure 2.3 shows the joint density of couple $(U\cos(Phi), U\sin(Phi))$ (left) and the conditionnal mean of the intensity givzn the wind direction. But this function does not compute any statistical test.

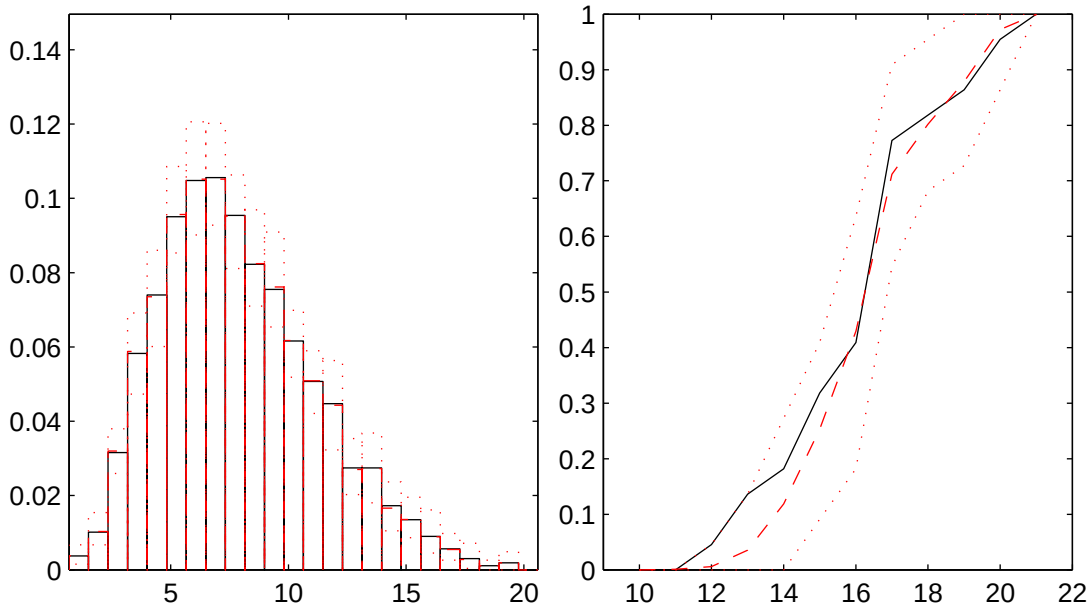

Figure 2.1: At the left: histogram of wind intensity. At the right: distribution of annual maxima of wind intensity - solid line observation; dashed line: simulation; dotted line: 95% interquantile range

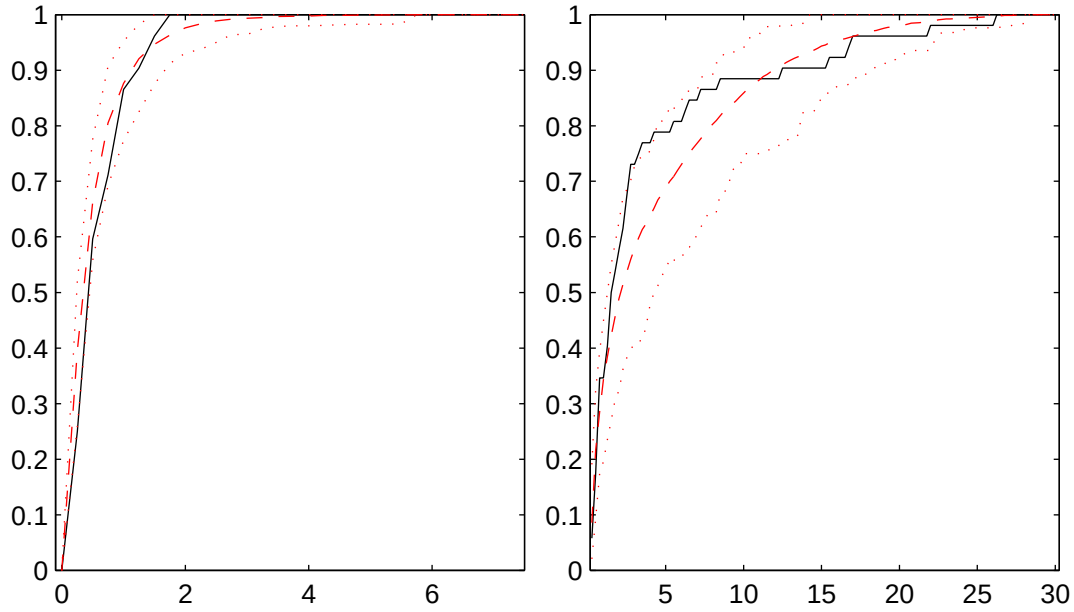

Figure 2.2: At the left: distribution of time duration wind intensity. At the right: distribution of annual maxima of wind intensity - solid line observation; dashed line: simulation; dotted line: 95% interquantile range

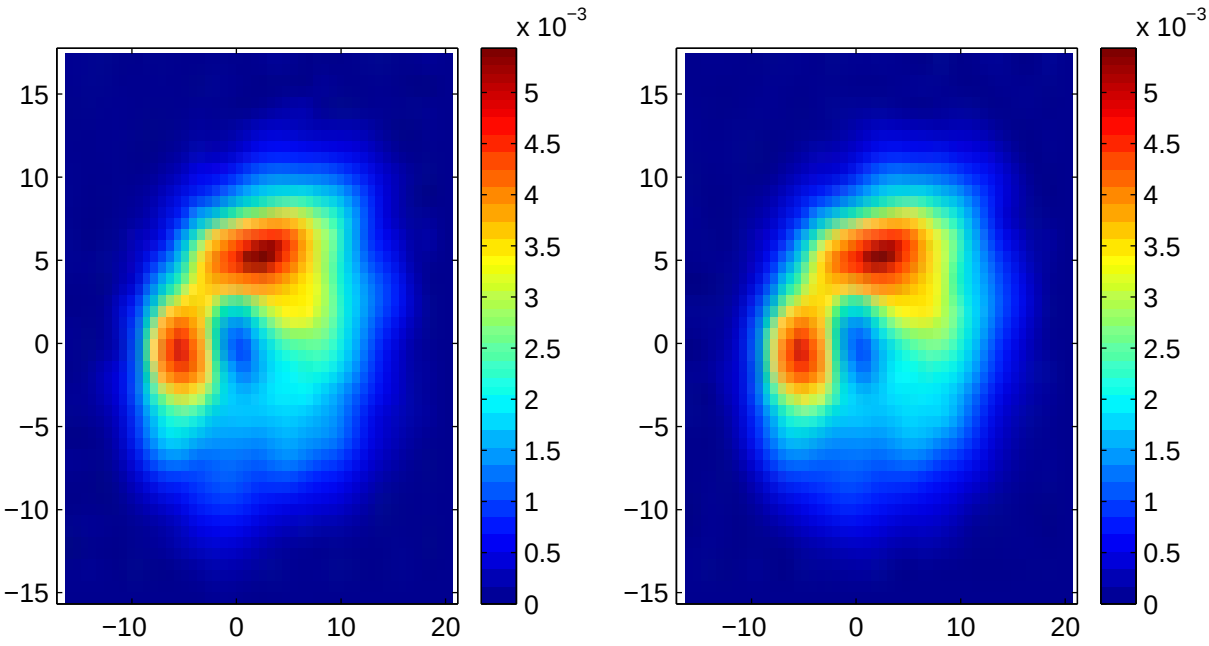

Figure 2.3: Joint probability density function of $(U\cos(Phi), U\sin(Phi))$ - Left: observation, right: simulation

```
graph_valid_bivar(U,Ug,Phi,Phig,1) ;
```

Figures 2.3 and 2.4 show how the instantaneous joint distribution of wind intensity given wind direction is restored by TGP.
More details about available tools for validation or comparison of time series simulation models are given in chapter 5. In particular, in section 5.3 a list is given of the statistics that can be plotted.

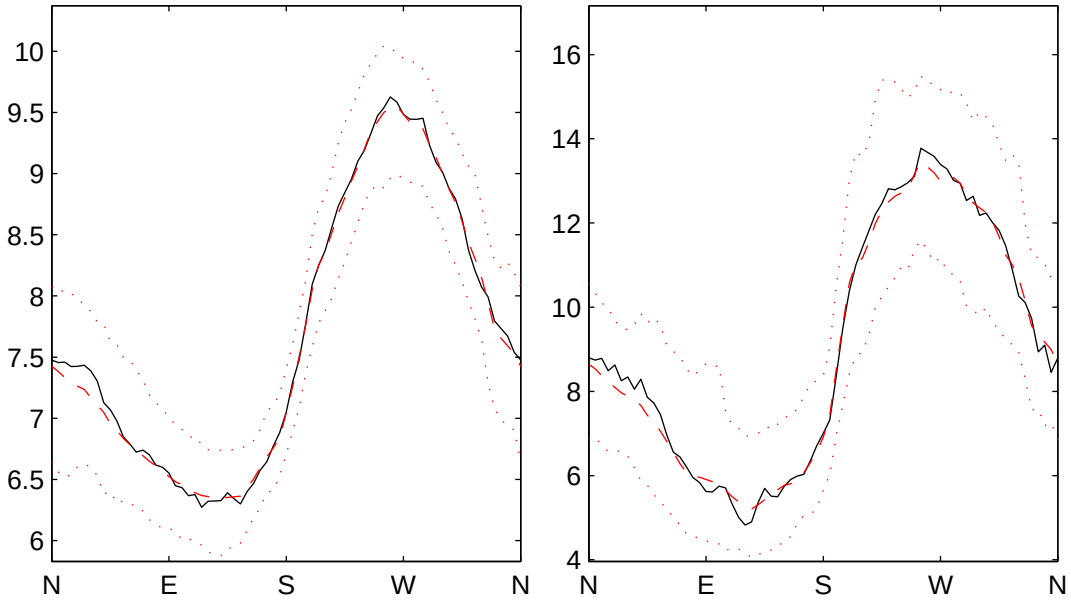

Figure 2.4: At the left: Conditional mean of wind intensity given wind direction; At the right: conditional standard diviation - solid line observation; dashed line: simulation; dotted line: 95% interquantile range

2.3 Description of the routines

2.3.1 Simulation of bivariate Gaussian process

SIMGAUSSPR simulation of a bivariate stationary gaussian process with same spectrum as a given observed sequence.

[un,vn] = simgausspr(u,v,dt,Nsim,s) ;

INPUTS
- u,v : components of the given bivariate gaussian process, size nxTxNobs
- s : size of the window to smooth the spectra (default s=1)
- Nsim : number of simulated samples, each sample will be 1xT (default Nsim = Nobs)

OUTPUTS
- un,vn : components of the simulated bivariate gaussian process

Reference
Scheffner N.W., Borgman L.E. (1992)
Stochastic Time-Series Representation of Wave Data
Journal of Waterway, Port, Coastal and Ocean Engineering, Vol 118 No.4

2.3.2 Rozenblatt transformation

ROZENBLAT like bivariate Rozenblatt transformation

Transformation of x is first computed and then transformation of y given x is made. The conditional distribution of y is computed by grouping x variable into k classes.
By default, the couple is transformed to Gaussian distribution.

[u,v]= rozenblat(x,y,k,dist1,dist2);

INPUTS
- x,y : sample to be transformed
- k : number of classes for the conditional transformation
- dist1 : distribution after transformation for 1st variable
evalable laws : gauss, rayleigh, uniform
- dist2 : distribution of y given x after transformation
evalable laws : gauss, rayleigh, uniform

OUTPUTS
- u,v : samples after conditionnal transformation
Distribution of u is dist1 and distribution of v given u is dist2

INVROZENBLAT inverse Rozenblat transformation
[xs,ys]=invrozenblat(x,y,us,vs,k,dist1,dist2);
INPUTS
- x,y : observed samples
- us,vs : Gaussian samples

OUTPUTS
- xs, ys : transformed process

GAUS2NGAUS inverse normal score transformation for bivariate processes
The inverse normal score transformation is performed with the empirical distribution functions of observed data (xi,yi). The input is the bivariate Gaussian process (un,vn).
Missing data are taking into account.

[xn,yn] = gaus2ngaus(xi, yi,un,vn,K,l) ;

INPUTS
- xi, yi : observations
- un, vn : Gaussian processes
- K : number of classes to estimate the distribution functions
- l : time interval
if l¡0, then the data are used to estimate the distribution function (stationarity)
OUTPUTS
- xn, yn : generated processes with the same instantaneous distribution as (xi,yi)

Chapter 3

Markov Switching Autoregressive Model

In Markov Switching Autoregressive Model the observed process $\{Y_t\}$ is modeled by an autoregressive model of order r with parameters depending on a non observable process $\{S_t\}$ which is a Markov chain. An important subclass of MS-AR model are the Hidden Markov Models (HMM), which correspond to the case $r = 0$. In these models the conditionnal distribution of Y_t given $\{Y_{t'}\}_{t'<t}$ and $\{S_{t'}\}_{t'\leq t}$ only depends on S_t : two successive values of the observed process are assumed to be conditionally independent given the values of the hidden process. HMM are used in many different areas, see Ephraim (2002) and references therein. In particular they have already been used for modelling the existence of weather type in meteorological time series like rainfall or wind direction, see MacDonald (1997).

3.1 Model description

A MS-AR process is a bivariate process $\{S_t, Y_t\}$ such that

- $\{S_t\}$ is a Markov chain on a finite space $\mathbf{S} = \{1, ..., M\}$ with $M > 0$ the number of regimes. This process is supposed to be non observable (or "hidden") and it may be interpreted as the weather type.
- Conditionally to $\{S_t\}$, $\{Y_t\}$ is a non-homogeneous Markov chain of order $r \geq 0$ on $\mathbf{Y} \subset \mathbf{R}^d$. More precisely, we assume that the conditional distribution of Y_t given $\{Y_{t'}\}_{t'<t}$ and $\{S_{t'}\}_{t'\leq t}$ only depends on S_t and $\overline{Y}_{t-1} = (Y_{t-1}, \cdots, Y_{t-r})$. We assume also that for each $s_t \in \mathbf{S}, \overline{y}_{t-1} \in \mathbf{Y}^r$, the conditional distribution $P(Y_t|\overline{Y}_{t-1} = \overline{y}_{t-1}, S_t = s_t)$ is a gamma distribution with mean $\sum_{i=1}^{p} a_i^{(s_t)} y_{t-i} + b^{(s_t)}$ and standard deviation $\sigma^{(s_t)}$.

The process $\{X_t\} = \{S_t, \overline{Y}_t\}$ is a first order Markov chain on $\mathbf{S} \times \mathbf{Y}$.

3.1.1 Gaussian MS-AR model

In MS-AR model, different types of autoregressive models can be used in order to describe the evolution of the observed process $\{Y_t\}$ in the different regimes. Generally, a functional autoregressive model of the form (3.1) is used.

$$Y_t = f^{(S_t)}(\overline{Y}_{t-1}) + \Sigma^{(S_t)} E_t \tag{3.1}$$

where $\{E_t\}$ represents a white noise such that E_t is independent of $Y_{t'}$ for $t' < t$. The functions $f^{(s)}$ can be linear (see Hamilton (1989) and Krolzig (1997)) or not (see Rynkiewicz (2000)).

In this toolbox, estimation and simulation tools have been implemented for MS-AR model with linear functions f^S and Gaussian innovation. The main advantage of Gaussian model is that most formula of the estimation step can be expressed explicitly. The programs could be easily adapted for other distributions. An example of application is given in section 3.3.

3.1.2 Gamma or Lognormal MS-AR model

Models of the form (3.1) may not suited to describe time series which take their values in a given subset of $\mathbf{R}^d$ as it is the case for most sea state parameters, for which $\mathbf{Y} = \mathbf{R}^+$ for instance. It is then more natural to specify directly the conditional distribution $P(Y_t|\overline{Y}_{t-1} = \overline{y}_{t-1}, S_t = s_t)$. For example, in order to describe the evolution of the wind speed, it is convenient to use a Markov Switching Gamma Autoregressive models, in which the conditional distributions are gamma distributions with

- mean $\mu^{(s_t)}(\overline{y}_{t-1}) = \sum_{i=1}^{r} a_i^{(s_t)} y_{t-i} + b^{(s_t)}$
- standard deviation $\sigma^{(s_t)}$

The constraints $a_i^{(s)} \geq 0$, $b^{(s)} > 0$ and $\sigma^{(s)} > 0$ have to be verified, for $s \in \mathbf{S}$ and $i \in \{1...r\}$, in order the model to be well defined.

The gamma distribution has been chosen because it is defined on $\mathbf{R}^+$ and its density can be easily expressed from its first two moments. Other distributions, like the log-normal or Weibull distributions, can also be used, however there exists no physical evidence nor statistical criteria that permits to choose the shape of this conditional distribution.

In the toolbox, the estimation and simulation tools have been implemented for Gamma and log-normal MS-AR models.

3.1.3 Non homogeneous gamma MS-AR model

In the model described above, only the wind speed was considered. However, for some applications, like erosion, the evolution of the studied phenomenum depends also on the wind direction. For this reason, it can be interesting to develop a stochastic model which describes the evolution of the bivariate process (U, Φ). There exists a complex relation between these two processes as we can see on Figure **??**, which represents the bivariate marginal distribution of this process. This distribution is bimodal, with a first peak which corresponds to cyclonic conditions and the other one to anticyclonic conditions. To model such relation, we propose an extension of the MS-γAR model discussed in the previous section. We have shown that there exists a strong relation between the hidden process and the wind direction, and that the different weather type are more likely to be associated to wind blowing from different directions. To descibe this relation, we will assume that the hidden process is a non-homogeneous Markov chain whose transition matrix depends on the wind direction. In this section, the wind direction is considered as a covariate. A brief discussion on the simulation of this process is given the section "model validation" below.

More precisely, the NHMS-γAR model we propose is such that :

- the evolution of the hidden variable $\{S_t\}$ is modeled by a non-homogeneous Markov chain with transition probabilities

$$q_\theta^{(t)}(i,j) \sim q_{i,j} \exp(\kappa_j cos(\Phi_t - \Phi_j)) \tag{3.2}$$

The constraints $\sum_{j=1}^{M} q_{i,j} = 1$ are imposed in order to ensure the identifiability of the parameters.

- the evolution of the wind speed in the different regimes is described by a gamma autoregressive model with parameters indexed by $\{S_t\}$ as in the MS-γAR model.

The relation between the different processes is summarized on the directed acyclic graph below :

$$\begin{array}{ccccccccc} \textit{covariate} & \cdots & & \Phi_{t-1} & & \Phi_t & & \Phi_{t+1} & & \cdots \\ & & & \downarrow & & \downarrow & & \downarrow & & \\ \textit{hidden process} & \cdots & \rightarrow & S_{t-1} & \rightarrow & S_t & \rightarrow & S_{t+1} & \rightarrow & \cdots \\ & & & \downarrow & & \downarrow & & \downarrow & & \\ \textit{observation} & \cdots & \rightarrow & Y_{t-1} & \rightarrow & Y_t & \rightarrow & Y_{t+1} & \rightarrow & \cdots \end{array} \tag{3.3}$$

In particular, we assume that

- $P(Y_t|\overline{Y}_0, ..., \overline{Y}_{t-1}, S_0, ..., S_t, \Phi_0, ..., \Phi_t) = P(Y_t|\overline{Y}_{t-1}, S_t)$. This implies that the evolution of the wind speed is conditionnally independant of the wind direction given the hidden process. We will see below, in the validation part, that the model is abble to catch the complex relation that exists between the wind speed and direction, what justifies this assumption.
- $P(S_t|\Phi_0, ..., \Phi_t, S_0, ..., S_{t-1}) = P(S_t|\Phi_t, S_{t-1})$. This means that the hidden process is a non-homogeneous Markov chain. The parametrization of its transition matrix is given by (3.2). Let us explain this choice. It is easy to check, thanks to the Bayes formula, that

$$P(S_t = j|S_{t-1} = i, \Phi_t) \backsim P(\Phi_t|S_{t-1} = i, S_t = j) \times P(S_t = j|S_{t-1} = i)$$

  Then, if we assume that the conditionnal distribution $P(\Phi|S_{t-1} = i, S_t = j)$ only depends on j, that the conditionnal distribution $P(\Phi|S_t = j)$ is a Von-Mises distribution with parameters (κ_j,Φ_j) and $P(S_t = j|S_{t-1} = i) = q_{i,j}$, we get the parametrization (3.2).

Non-homogeneous hidden Markov (NHHMM) models have already been used to describe meteorological time series. In Hugue et al. (1999), it is used in order to relate broad scale atmospheric circulation variables (which play the role of the covariate) to local rainfall (the observed process). In this model, the hidden process is also interpreted as a "weather type". In MacDonald et al. (1997), a NHHMM is proposed to model seasonal and daily components existing in time series of wind direction in Koeberg (South Africa). In their model, the transition matrix of the hidden process is a fonction that depends only on the time. We also found that the NHMS-γAR can be used to describe the daily components that exist in wind time series during the summer, but this application is not discussed here. More details can be found in Ailliot (2004).

3.2 Parameter estimation

In this part, we discuss the problem of parameter estimation in MS-AR models. As the process $\{S_t\}$ is not observable, the inference on the parameter θ has to be carried out only in terms on the observable process $\{Y_t\}$.

We denote $\theta_R^{(s)} = (a_1^{(s)}, ..., a_r^{(s)}, b^{(s)}, \sigma^(s)) \in \mathbf{\Theta_R}$ the parameters that describe the evolution of the observed process in the regime $s \in \mathbf{S}$, with $\mathbf{\Theta_R}$ a given compact subset of $(\mathbf{R}^+)^r \times \mathbf{R}^{+*} \times \mathbf{R}^{+*}$. Furthermore, $\{y_t\}_{t=-r+1}^{T}$ will denote an observation of this process and $\theta_0 = (\theta_{S,0}, \theta_{R,0}^{(1)}, ..., \theta_{R,0}^{(M)}) \in \mathbf{\Theta}$ the true value of the parameter.

The numerical computation of the MLE in models with hidden variables has been addressed by many authors. The most popular method is probably the EM algorithm which has been first introduced by Baum et al. (1970) for HMM and then generalized to other models with hidden variables

by Dempster et al. (1977). The description of the particular form of this algorithm for MS-AR models can be found in Hamilton (1989). This algorithm has several well-known limitations. First, as the likelihood function can be multi-modal, it may converge to a local maximum depending on the initial value. Another drawback is its slow rate of convergence near the maxima (linear rate of convergence) while a quasi-Newton algorithm is more efficient near these points (superlinear rate of convergence).

In practice, an hybrid algorithm is used and the estimation part is broken up into three steps:

1. The EM algorithm is first run with several randomly chosen initial values in order to locate an "interesting" maximum. For each initial point, N_1 iteration of EM are computed, with N_1 quite low, and the best parameter in the sense of maximum likelihood is retained. In practice, a forward-backward algorithm is implemented that returns the probability $S_t = s$ given the observation time series $y_1, \cdots, y_T$.

2. Then, EM is used again with the retained parameter as initial value. The algorithm is stopped when
$$\frac{l(\theta^{(n)}) - l(\theta^{(n-1)})}{l(\theta^{(n)}) - l(\theta^{(n-1)})} < \epsilon$$
where l denotes the function to be maximized and $\theta^{(n)}$ the value of the parameter at iteration n. By default, ϵ is chosen equal to 10^{-4}.

3. Finally, a quasi-Newton algorithm is used in order to get the final estimate.

This procedure has been tested with success on synthetic data. It is worth noting that it is used both for homogeneous and non homogeneous MS-AR models.

3.3 Example: wind simulation

Let us denote Y the wind process. In a first time, Y corresponds to the wind intensity only.

MS-AR model with Gaussian innovations

We choose here to identify a MS-AR model of order 1 and with 3 regimes and Gaussian innovations. A justification for such a model is given for instance in Ailliot and Monbet (2004).

order = 1 ;
M = 2 ;

As we said above, the parameter of the model are estimated by an EM algorithm. Wa have first to choose the initial conditions for computation using cond_ini function, then to fix algorithm parameters such as maximal iteration number max_iter and convergence precision eps. In cond_ini function, initial parameters (prior distribution prior, initial transition matrix transmat, initial regression parameters A, sigma, moy) are chosen randomly.

[prior, transmat, A, sigma,moy] = cond_ini(ordre,Y,M);
max_iter = 100; eps = 10^{-7};

Now, EM algorithm can be computed.

As output of AR_EM function, we have the estimated parameters of the MS-AR model and the likelihood LL of the identified model, the most likely hidden sequence given the observation time series path obtained by a Viterbi algorithm and the number of parameter in the model nbpar that

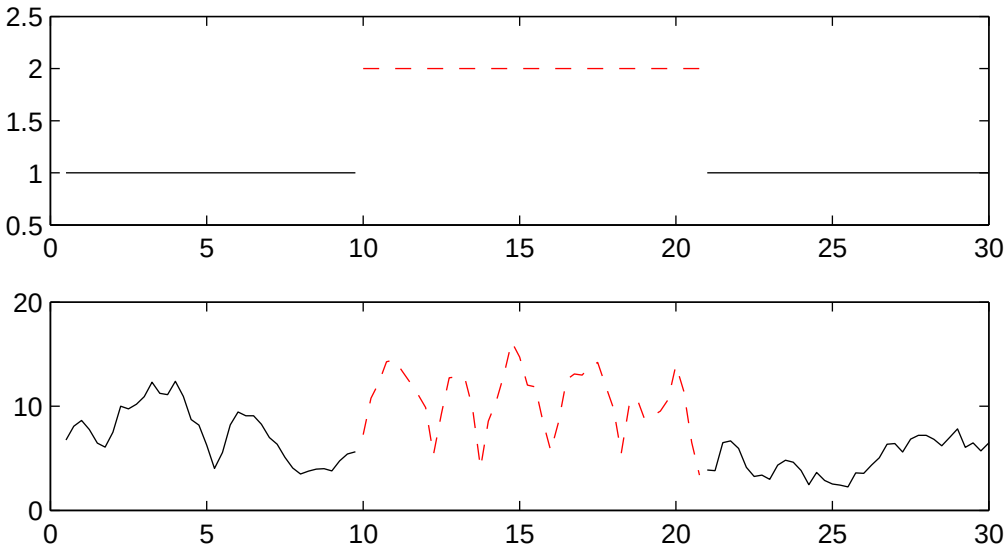

Figure 3.1: Gaussian MS-AR model with 2 regimes. Most likely evolution of the hidden process computed by Viterbi algorithm (at the top) for wind intensity of a january mounth (at the bottom). Time in abscissa is given in days.

is usefull to compute BIC critera for instance. Matlab variable path permits also to help for the interpretation of the model. Let us, as example, compare in Figures 3.1 the evolution of the hidden Markov chain of the model with 2 regimes. We remark on Figure 3.1 that regime 1 corresponds to weather type where the evolution of the wind intensity is slower than in regime 2. Regime 2 seems to correspond to cyclonic conditions. See also program MSARdemo.m.

```
[LL,prior,transmat,A,sigma,moy,path,nbpar] = AR_EM(X,ordre,prior,transmat,A,sigma,moy,max_iter,eps);


ech = 3 ;
pp1 = path(ech,:) ; XX1 = X(1,:,ech) ;
f1 = find(pp1 == 1 ); p1 = NaN*zeros(size(pp1)) ; p1(f1) = pp1(f1) ;
X1 = NaN*zeros(size(pp1)) ; X1(f1) = XX1(f1) ;
f2 = find(pp1 == 2 ); p2 = NaN*zeros(size(pp1)) ; p2(f2) = pp1(f2) ;
X2 = NaN*zeros(size(pp1)) ; X2(f2) = XX1(f2) ;
time = (1:121)/4 ;

figure
subplot(2,1,1)
plot(time,p1,'k',time,p2,'r-')
xlabel('days')
axis([0 30 0.5 3.5])
subplot(2,1,2)
plot(time,X1,'k',time,X2,'r-')
axis([0 30 0 20])
xlabel('days')
```

Then, the model can be used in simulation as follows:

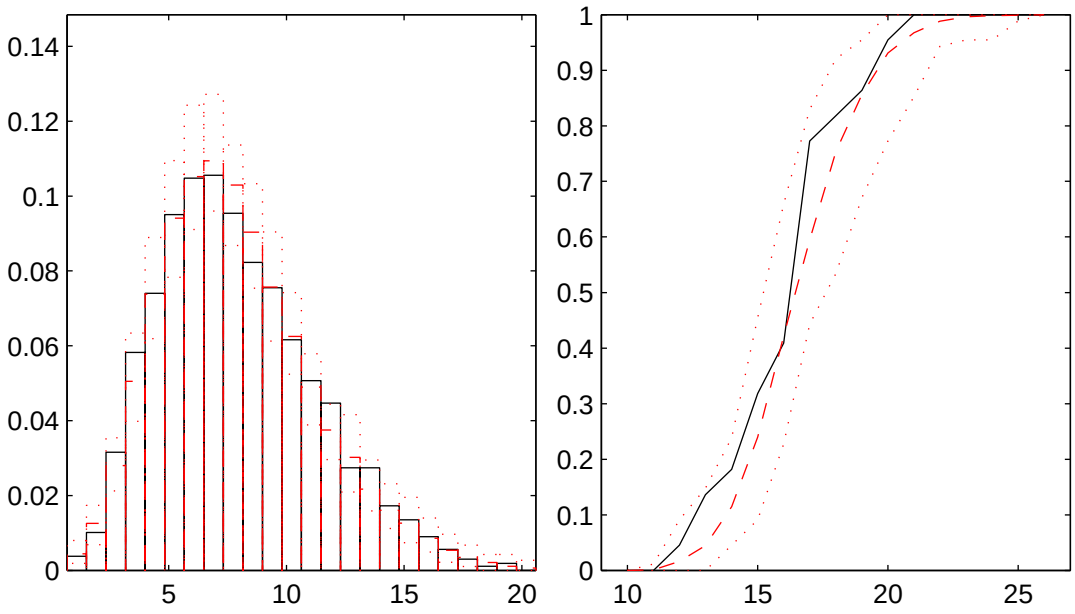

Figure 3.2: At the left: histogram of wind intensity. At the right: distribution of annual maxima of wind intensity - solid line observation; dashed line: simulation; dotted line: 95% interquantile range

```
Nsim = 1100; % number of samples to be simulated
X_sim = zeros(size(X,1),Tsais,Nsim);
for ech=1:Nsim,
        % simulation of the hidden Markov chain
        CM(ech,:) = simule_CM(transmat,prior,Tsais)';
        % simulation of the observation time series
        X_sim(:,:,ech) = simule_AR(CM(ech,:),A,sigma,moy,X(:,:,floor(10*rand(1))+1));
end;
```

The validation can be performed graphically as for TGP.

```
graph_valid_univ(X,X_sim,0.05,1,1) ;
```

Some of the return figures are included hereafter.

MS-AR model with Gamma innovations

The steps performed just above for Gaussian MS-AR model can be computed as follows for Gamma MS-AR model.

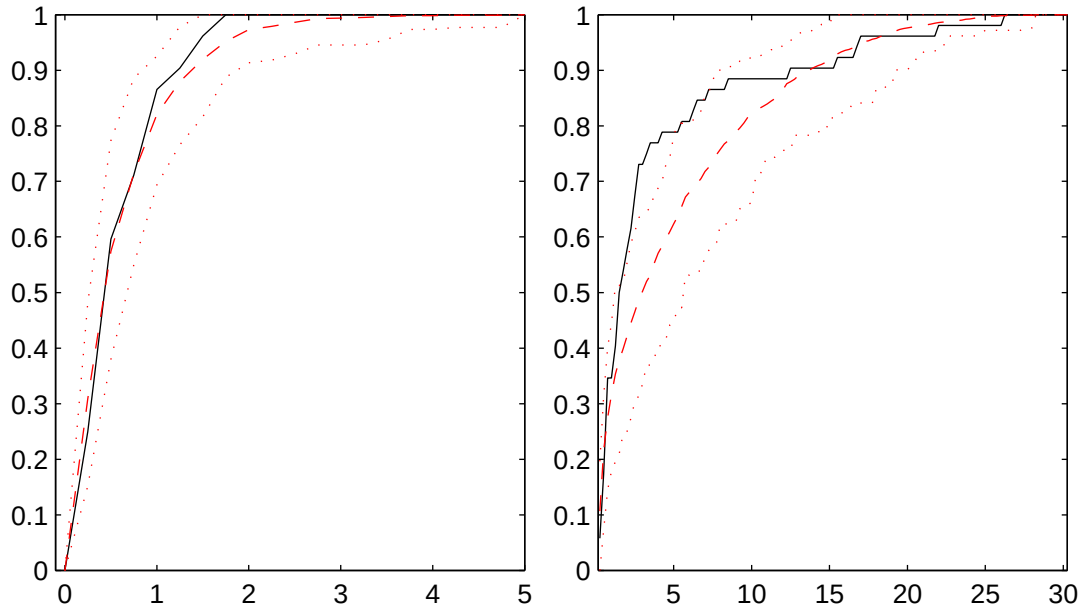

Figure 3.3: At the left: distribution of time duration wind intensity. At the right: distribution of annual maxima of wind intensity - solid line observation; dashed line: simulation; dotted line: 95% interquantile range

```
[A,sigma,moy,prior,transmat] = cond_ini_gamma(M,ordre);

%EM algorithm
max_iter = 50;
a2 = [A,moy,sigma] ;
[LL,prior2,transmat2,aa2,path,nbpar] = ARgamma(X,prior,transmat,a2,'gamma',max_iter,eps);

% SIMULATION
X_sim2 = zeros(1,Tsais,Nsim); CM = zeros(Nsim,Tsais);
for ex=1:Nsim
CM(ex,:) = simule_CM(transmat2,prior2,Tsais)';
X_sim2(:,:,ex) = simule_ARgamma(CM(ex,:),aa2,X(:,1:ordre,floor(9*rand(1)+1)),'gamma');
end

% VALIDATION
stat_valid_univ(X,X_sim,0.05,1,1) ;
```

The Gamma distribution is replaced by the lognormal distribution in the above commands by substituting 'gamma' by 'lognormal' in ARgamma and simule_ARgamma.

Let us check the physical realism of the model with $M = 2$ regimes. The MLE parameters corresponding to this model are given below, the values in brackets corrsponding to the standard deviations of the MLE obtained from the observed information matrix.

- **regime 1** $a_1^{(1)} = 0.79[0.016]$, $b^{(1)} = 1.46[0.11]$, $\sigma^{(1)} = 1.37[0.057]$
- **regime 2** $a_1^{(2)} = 0.77[0.018]$, $b^{(2)} = 2.40[0.23]$, $\sigma^{(2)} = 2.40[0.074]$
- **transition matrix** $q(1,1) = 0.98[0.0060]$, $q(2,2) = 0.97[0.0092]$

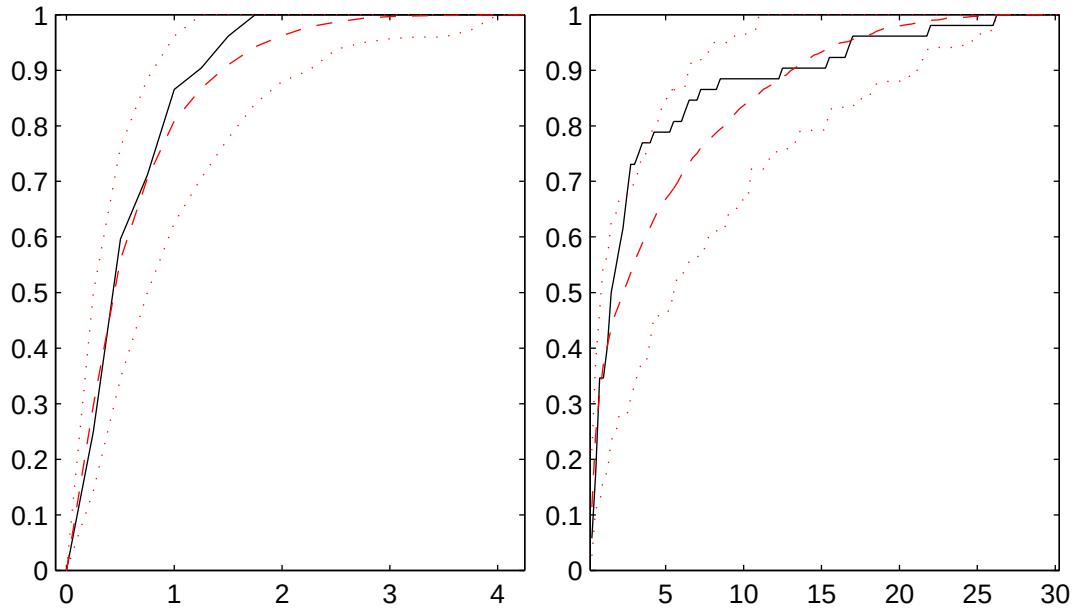

Figure 3.4: At the left: distribution of time duration of sejourn over level $\frac{2}{3}\max U$. At the right: distribution of time duration of sejourn under level $\frac{2}{3}\max U$ - solid line observation; dashed line: simulation; dotted line: 95% interquantile range

According to these values, we can propose the following meteorological interpretation. First, we can see that $\sigma^{(2)}$ is higher than $\sigma^{(1)}$. This means that the second regime corresponds to periods when the wind speed evolve quickly, as it is the case in the presence of stormy conditions, whereas the first one is associated to slowly evolving wind, and thus corresponds anticyclonic conditions. Parameters $a^{(1)}$ and $a^{(2)}$ are closed to each other but $b^{(2)}$ is higher than $b^{(1)}$. This imply that the mean of the stationary distribution corresponding to the second regime (which is given by $b^{(2)}/(1-a^{(2)})$) is higher than the mean of the first regime : we find again that the wind is usually higher in cyclonic conditions. The values on the diagonal of the transition matrix are high, and thus the different states have a long mean duration time: about 2 weeks for the first one and 1 week for the second. These values seem physically realistic according to the local climatology.

Validation function stat_valid_univ used to compare the generated time series of MS-AR gamma model with the observations permits to plot figures such as figures 3.4.

Non homogeneous MS-AR with gamma innovations

In the Gamma ARHMM model for the wind intensity given the wind direction, the transition matrix of the hidden Markov chain is conditionned by the wind direction. Estimation of the parameters is performed as follows:

```
M = 3 ; ordre = 1 ;
max_iter = 30;
eps = 1e-4;
bestrun = 1 ;
run = 1 ;
[A,sigma,moy,prior,transmat,par] = cond_ini_nh(M,ordre);
[LLgrun, priorgrun, transmatgrun,Agrun,sigmagrun,moygrun, ...
pargrun,nbpargrun,pathgrun] ...
= ARnhgamma(U,Phi, prior,transmat,A,sigma,moy,par,max_iter,eps);
for run=2:5
        [A,sigma,moy,prior,transmat,par] = cond_ini_nh(M,ordre);
        [LLgrun, priorgrun, transmatgrun,Agrun,sigmagrun,moygrun,...
        pargrun,nbpargrun,pathgrun] ...
        = ARnhgamma(U,Phi, prior,transmat,A,sigma,moy,par,max_iter,eps);
        if LLgrun(end)¿LLgrun-1(end), bestrun = run ; end ;
end
```

Then, it is convenient to check the interpretability of coefficients that drive the evolution of the hidden Markov chain.

```
transmat = transmatgbestrun ;
par = pargbestrun ;
tab = 0:360;
transition = mk_transition(transmat,mod(pi/180*(270-tab),2*pi),par);
for m=1:M,
        subplot(1,M,m)
        tmp = transition(m,m,:);
        plot(tab,tmp(:));
        axis([0,360,0,1])
end
```

As a first approximation, the wind direction is modeled by a first order Markov chain. Local bootstrap algorithm (Monbet, 2004) can be used to simulate time series of wind direction. Such an algorithm is available in C language but it is not interfaced with matlab until now.

```
tab1 = pi/180*(0:20:360);
trans = evalue_transition(mod(Phi,2*pi),tab1);
tmp = trans'Î00; prior_dir = tmp(:,1);
```

The parameters which govern the evolution of $\{Y_t\}$ in the different regimes are given below:

- **regime 1** $a_1^{(1)} = 0.85[0.001]$, $b^{(1)} = 1.19[0.029]$, $\sigma^{(1)} = 1.18[0.010]$
- **regime 2** $a_1^{(2)} = 0.58[0.002]$, $b^{(2)} = 1.85[0.064]$, $\sigma^{(2)} = 1.60[0.011]$
- **regime 3** $a_1^{(3)} = 0.71[0.002]$, $b^{(3)} = 4.01[0.279]$, $\sigma^{(3)} = 1.81[0.007]$

The parameters $\gamma_{i,j}, \kappa_j$ and Φ_j are more difficult to interpret directly. We have plotted the conditional transition probabilities $\Phi \to q_\theta^{(\Phi)}(i,j)$ on the figure 3.5.

In this case, reasonable meteorological interpretation can be given to the different weather types. For example, the third regime corresponds to cyclonic conditions: the wind speed evolves quickly, can reach high values, and it is generally associated to winds blowing from the South-West.

Now, simulation of wind direction and wind intensity can be run.

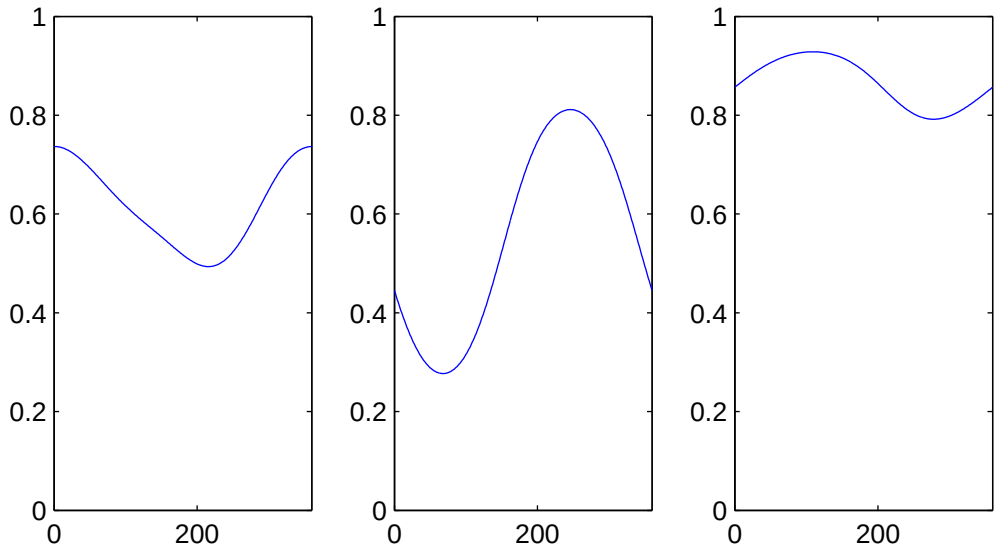

Figure 3.5: Conditional transition probabilities $q_{\theta}^{(\Phi)}(i,j)$

```
% Wind direction simulation
tab2 = pi/180*(10:20:360);
Phi_sim = zeros(1,Tsais,Nsim);
for i=1:Nsim,
        Phi_sim(1,:,i) = tab2(simule_CM(trans,prior_dir,Tsais));
end

% Wind intensity simulation
a2 = [Agbestrun,moygbestrun,sigmagbestrun] ;
Phi_sim = Phi ; Nsim = size(Phi,3) ;
U_sim = zeros(size(Phi_sim));
for i=1:Nsim,
        transition = mk_transition(transmat,Phi_sim(1,:,i),pargbestrun); % transitions
        chemin = nhcm(transition,prior); % hidden variable
        U_sim(:,:,i) = simule_ARgamma(chemin,a2,U(1,1:ordre,floor(rand(1)*9)+1));
end
```

Validation can be performed calling:

```
stat_valid_bivar(X,X_sim,Y,Y_sim,0.05,1,1)
```

or

```
graph_valid_bivar(X,X_sim,Y,Y_sim,0.05,1,1)
```

stat_valid_bivar plots curves in order to compare some statistics of the observed sequence with those of the simulation and it computes confidence intervalles and the pvalues of statistic test described in chapter 5. graph_valid_bivar plots all the graphics of stat_valid_bivar and the joint probability fonction. It desn't return the pvalues.

3.4 Description of the routines

3.4.1 Gaussian MS-AR

Estimation

AR_EM estimates the parameters of a Gaussian autoregressive model with Markovian switching. Input may be a vector and the order of the model can be chosen.

[LL, init_state_prob, transmat, A, Sigma,moy,path,nbpar] =
AR_EM(data, order, init_state_prob, transmat, A, Sigma,
moy, [max_iter, [thresh, [verbose, [cov_type, [static)

INPUTS

- data	matrix of data, size: K*Tsais*Nech
- order	order of the autoregressive model
- init_state_prob	initial discrete probability of the hidden Markov chain (size: M*1)
- transmat	initial transition matrix of the hidden Markov chain (size: M*M)
- A,Sigma,moy	parameters of autoregressive models. When C(t)=i, data(:,t,ex)=A1,i* data(:,t-1,ex)+...+Aorder,i* data(:,t-order,ex)+moy(:,i)+E(t) with E(t) N(0,sigma(:,:,i))

optionnal inputs

- max_iter	max. num EM steps to take (default 100)
- thresh	threshold for stopping EM (default 1e-4)
- verbose	0 to suppress the display of the log lik at each iteration (Default 1).
- cov_type	'full', 'diag' or 'spherical' (default 'full')
- static	1 if we don't want to re-estimate prior and transmat, i.e., no dynamics (default = 0)

OUTPUTS

- LL	loglikelihood of the identified model
- path	most likely hidden sequence given the observation time series, computed by a Viterbi algorithm.
- nbpar	number of parameters of the model (for AIC or BIC computation)

E_STEP Estimation step (Forward-Backward) of the Baum-Welch algorithm for a Gaussian AR model with Markovian switching.

[loglik, exp_num_trans, exp_num_visits1, postmix, m, m_1, op, op_1,op_2,temp,proba]
= e_step(prior, transmat, A, Sigma, mu, data, order)

INPUTS

- prior	initial discrete probability of the hidden Markov chain (size: M*1)
- transmat	initial transition matrix of the hidden Markov chain (size: M*M)
- A,Sigma,mu	parameters of autoregressive models. When C(t)=i, data(:,t,ex)=A1,i* data(:,t-1,ex)+...+Aorder,i* data(:,t-order,ex)+moy(:,i)+E(t) with E(t) N(0,sigma(:,:,i))
- data	matrix of data, size: K*Tsais*Nech
- order	order of the autoregressive model

OUTPUTS

- loglik	log likelihood of the model
- exp_num_trans	
- exp_num_visits1	
- postmix	
- m, m_1	
- op, op_1, op_2	

———- forwards-backwards function (called in e_step.m) ————-

INPUTS (of forwards-backwards function)
- PRIOR(I) = Pr(Q(1) = I)
- TRANSMAT(I,J) = Pr(Q(T+1)=J — Q(T)=I)
- B(I,T) = OBSLIK(I,T) = Pr(Y(T) — Q(T)=I)

OUTPUTS (of forwards-backwards function)
- gamma(i,t) = Pr(X(t)=i — O(1:T))
- xit(i,j,t) = Pr(X(t)=i, X(t+1)=j — O(1:T)) t ¡= T-1

mk_ARgaussian calculates the conditional likelihood for Gaussian AR model for each hidden state.

B = mk_ARgaussian(data, A, Sigma, mu, order)

INPUTS

- data	matrix of data (points for which the log likelihood is calculated), size K *Tsais *Nech
- A,Sigma,mu	parameters of autoregressive models. When C(t)=i, Y(:,t,ex)=A1,i*Y(:,t-1,ex) + ... + Aorder,i*Y(:,t-order,ex) + mu(:,i) + R(t) with R(t) N(0,sigma(:,:,i)) Residuals R are supposed to be i.d. Gaussian
- order	order of the autoregressive model

OUTPUTS

- B	conditional likelihood, B(j,t) = Pr(data(t)—S(t) = j)

Simulation

simule_AR simulates an AR process with markovian switching. The schwitchings are governed by an Hidden Markov chain.

Y=simule_AR(CM,A,sigma,mu,Y0);

INPUTS

- CM	path of the Hidden Markov variable
- A,Sigma,mu	parameters of autoregressive models. When C(t)=i, Y(:,t,ex)=A1,i*Y(:,t-1,ex) + ... + Aorder,i*Y(:,t-order,ex) + mu(:,i) + R(t) with R(t) N(0,sigma(:,:,i)) Residuals R are supposed to be i.d. Gaussian
- Y0	matrix of initial conditions

OUTPUT

- Y	generated time series

3.4.2 Gamma MS-AR

Estimation

ARgamma estimates GARHMM model. C(t) is a Hidden Markov chain defined on states 1...K, with transition matrix transmat initial distribution at t=0 prior. The distribution of X(t+1) is Gamma with mean m such that

$$m(t+1) = A(C(t+1), ordre)X(t) + ... + A(C(t+1), 1)X(t - ordre) + moy(C(t))$$

and variance Sigma(C(t+1))

[LL, prior, transmat,a,path,nbpar]
= ARgamma(data, prior, transmat,a,dist[, max_iter[, thresh[, verbose[, cov_type[, static)

INPUTS

- data	matrix of data, size: Nex*Tsais*Nyear
- order	order of the autoregressive model
- prior	initial discrete probability of the hidden Markov chain (size: M*1)
- transmat	transition matrix of the hidden Markov chain (size: K*K)
- a	parameters of autoregressive models. a = [A,Sigma,moy] size(A) = K*order, size(Sigma) = K*1, size(moy) = K*1
- dist	chosen distribution dist = 'gamma' (default) or dist = 'lognormal'

optional inputs

- max_iter	max. num EM steps to take (default 100)
- thresh	threshold for stopping EM (default 1e-6)
- verbose	0 to suppress the display of the log lik at each iteration (Default 1).
- cov_type	'full', 'diag' or 'spherical' (default 'full')
- static	1 if we don't want to re-estimate prior and transmat, i.e., no dynamics (default = 0)

OUTPUTS

- LL	log likelihood of the model
- prior	initial discrete probability of the hidden Markov chain (size: M*1)
- transmat	transition matrix of the hidden Markov chain (size: K*K)
- a	parameters of autoregressive models. a = [A,Sigma,moy] size(A) = K*order, size(Sigma) = K*1, size(moy) = K*1
- path	most likely path of the hidden chain C given the observation Y time series, computed by a Viterbi algorithm
- nbpar	number of parameters in the model

FB_GAMMA runs forward-backwards algorithm for GARHMM

[loglik, exp_num_trans, exp_num_visits1, postmix, proba]
= fb_gamma(prior,transmat,a,data,dist)

INPUTS

- prior	initial discrete probability of the hidden Markov chain (size: M*1)
- transmat	transition matrix of the hidden Markov chain (size: K*K)
- a	parameters of autoregressive models. a = [A,Sigma,moy], size(A) = K*order, size(Sigma) = K*1, size(moy) = K*1
- dist	chosen distribution dist = 'gamma' (default) or dist = 'lognormal'

OUTPUTS

- loglik	loglikelihood
- exp_num_trans(i,j)	sum_l(sum_t(P(C(t+1)=j,C(t)=i — obs(l))) where l denotes the sample number in 1,···,Nech and C the Hidden Markov chain.
- exp_num_visits1(i)	sum_l Pr(C(1)=i — Obs(l))
- postmix(i)	sum_l(sum_t(proba(i,t,l))
- proba(i,t,l)	Pr(C(t)=i — Obs(l))

LL_ARGAMMA computes the log likelihood to be maximized to identify the GARHMM model.

[ll,dll] = ll_ARgamma(a,data,cond_pr,dist);

INPUTS

- a	parameter
- data	matrix of data, size: K(or 1?)*Tsais*Nech
- cond_pr	probability such that (cond_pr(i,t,l) = Pr(C(t)=i — Obs(l)))
- order	order of the autoregressive model
- dist	chosen distribution dist = 'gamma' (default) or dist = 'lognormal'

OUTPUTS

- ll	-loglikelihood
- dll	gradient of -loglikelihood

Simulation

simule_ARgamma simulation of an GARHMM process with markovian switching. The schwitchings are governed by an Hidden Markov chain. Y(:,t,ex) have distribution 'dist' with mean defined by

$$mu + A\{1,i\} * Y(:,t-1,ex) + ... + A\{order,i\}Y(:,t-order,ex) \tag{3.4}$$

and variance given by Sigma.

Y = simule_ARgamma(CM,a,Y0,dist);

INPUTS

- CM	path of the Hidden Markov variable
- a	parameters of autoregressive models. a = [A,Sigma,mu] When C(t)=i, Y(:,t,ex) have distribution 'dist' with mean defined by A1,i*Y(:,t-1,ex) + ... + Aorder,i*Y(:,t-order,ex) equation (3.4) and variance given by Sigma+ mu(:,i) + R(t)
- Y0	initial value for Y (size : order*1)
- dist	chosen distribution dist = 'gauss' dist = 'gamma' (default) or dist = 'lognormal'

OUTPUT

- Y	generated time series

3.4.3 Non homogeneous MS-AR model

Estimation

ARnhgamma estimates the non homogeneous GARHMM model.

C(t) is a Hidden Markov chain defined on states 1...K, with
transition matrix transmat
initial distribution at t=0 prior

The transition probabilities of the non homogeneous Markov chain are parametrized using von Mises distribution P(C(t+1)=j/C(t)=i,
entr(t+1)) $\sim$ transmat(i,j)/besseli(par(j,2))*exp(par(j,2)*cos(entr(t+1)-par(j,1)))
X(t+1) $\sim$ gamma with mean

$$m(t+1) = A(C(t+1), ordre)X(t) + ... + A(C(t+1), 1)X(t - ordre) + moy(C(t))$$

and variance Sigma(C(t+1))

[LL, init_state_prob, transmat, A, Sigma,moy,par,nbpar,path]
= ARnhgamma(data,entr,init_state_prob,transmat,A,Sigma,moy,par,
max_iter,thresh,verbose,cov_type,static)

INPUTS

- data	matrix of data, size: Nex*Tsais*Nyear
- entr	process which "drives" the non homogeneous Markov chain
- order	order of the autoregressive model
- prior	initial discrete probability of the hidden Markov chain (size: M*1)
- transmat	transition matrix of the hidden Markov chain (size: K*K)
- a	parameters of autoregressive models. a = [A,Sigma,moy] size(A) = K*order, size(Sigma) = K*1, size(moy) = K*1
- par	parameters of Von-Mises distributions. par(i,1) = moy(i), par(i,2) = kappa(i)

optional inputs

- max_iter	max. num EM steps to take (default 10)
- thresh	threshold for stopping EM (default 1e-4)
- verbose	0 to suppress the display of the log lik at each iteration (Default 1).
- cov_type	'full', 'diag' or 'spherical' (default 'full')
- static	1 if we don't want to re-estimate prior and transmat, i.e., no dynamics (default = 0)

OUTPUTS

- LL	log likelihood of the model
- prior	initial discrete probability of the hidden Markov chain (size: M*1)
- transmat	transition matrix of the hidden Markov chain (size: K*K)
- a	parameters of autoregressive models. a = [A,Sigma,moy] size(A) = K*order, size(Sigma) = K*1, size(moy) = K*1
- par	parameters of Von-Mises distributions for nh Markov chain
- nbpar	number of parameters in the model
- path	most likely path of the hidden Markov chain given the observation, computed by a Viterbi algorithm

FB_NHGAMMA runs forward-backwards algorithm for non homogeneous GARHMM

[loglik,exp_num_trans,exp_num_visits1,postmix,proba,xi]
= fb_nhgamma(prior,transmat,A,Sigma,mu,par,data,entr)

INPUTS

- prior	initial discrete probability of the hidden Markov chain (size: M*1)
- transmat	transition matrix of the hidden Markov chain (size: K*K)
- a	parameters of autoregressive models.
	a = [A,Sigma,moy], size(A) = K*order, size(Sigma) = K*1, size(moy) = K*1
- par	parameters of Von-Mises distributions for nh Markov chain
- data	observations (size K*Tsais*Nsamp)
- entr	process which "drives" the nh Markov chain

OUTPUTS

- loglik	loglikelihood
- exp_num_trans(i,j)	sum_l(sum_t(P(C(t+1)=j,C(t)=i — obs(l)))
	where l denotes the sample number in 1,...,Nech
	and C the Hidden Markov chain.
- exp_num_visits1(i)	sum_l Pr(C(1)=i — Obs(l))
- postmix(i)	sum_l(sum_t(proba(i,t,l)) with
- proba(i,t,l)	Pr(C(t)=i — Obs(l))
- xi(i,j,t,l)	Pr(C(t)=j, C(t-1)=i — Obs(l))

LL_HMM_INP computes the log likelihood to estimate the non homogeneous Markov chain.

ll = ll_hmm_inp(Y,entr,xi);

INPUTS

Y	vector of parameters
entr	process which "drives" the nh Markov chain
xi	

OUTPUT

ll	log likelihoood

NHCM generates a Markov chain with transition matix changing with time y = nhcm(transition,y0)

INPUTS

transition	transition matrix
y0	initial value

OUTPUT

y	Markov chain

Chapter 4

Filtering and smoothing

In metocean field, Kalman filtering is for instance applied to predict wave given wind observation or to assimilate data (Kobayashi, 2004). In this chapter, a non parametric estimation method for state space model is introduced and an application to filtering and smoothing is presented.

4.1 State space model

State space model is basically a Markov chain whose internal state cannot be observed directly but only through some probabilistic function (Rabiner, 1993). That is, the internal state of the model only determines the probability distribution of the observed variables.
Let us introduce standard notations to describe more precisely the model. With $\{S_t\}$ the state process and $\{Y_t\}$ the observed process and $t \in \mathbb{N}$, we assume that

(M1) $\{S_t\}$ is an homogeneous Markov chain defined on a compact set $\mathcal{S} \subset \mathbb{R}$. We denote by $Q(s, A)$ the Markov transition kernel of the chain. We assume that the transition kernel Q has a density $a(s'|s)$ with respect to Lebesgue measure and admits a unique invariant distribution which admits a density denoted by f. In addition, we assume that the initial distribution of S_1 is absolutly continuous with respect to Lebesgue measure and ξ_1 denotes the corresponding density.
(M2) $\{Y_t\}$ takes their values in a compact set $\mathcal{Y} \subset \mathbb{R}^d$ with $d \geq 1$. Futhermore, for each $r \geq 1$, Y_r is conditionally independant of $\{Y_t\}_{t=1,\cdots,r-1}$ given S_r. We also assume that for each $s \in \mathcal{S}$, the conditional distribution $P(Y_r|S_r = s)$ has a density $b(y|s)$ with respect to Lebesgue measure.

This model is a special case of a graphical model on a acyclic directed graph (Lauritzen, 1996).

$$\begin{array}{ccccccccccc} \textit{state process} & \cdots & \rightarrow & S_{t-1} & \rightarrow & S_t & \rightarrow & S_{t+1} & \rightarrow & \cdots \\ & & & \downarrow & & \downarrow & & \downarrow & & \\ \textit{observation} & \cdots & & Y_{t-1} & & Y_t & & Y_{t+1} & & \cdots \end{array} \tag{4.1}$$

One usually calls this model a *Hidden Markov Model* (HMM) when the hidden state space $\mathcal{S}$ is finite and *state space model* when it is infinite. The basic model may be extended by introducing additional arrows on the graph.

4.1.1 Filtering and smoothing recursions

For $r \leq t$, we define $y_r^t = (y_r, \cdots, y_t)$ and similarly s_r^t. We denote the conditional density of S_t given $Y_1^r = y_1^r$ by $f_{t|r}(.)$ and distinguish between prediction ($r < t$), filtering ($r = t$) and smoothing ($r > t$). Otherwise we use the sloppy but useful notation $p(u|v)$ for the conditional

density $p(U = u|V = v)$ of a stochastic vector U given another stochastic vector V. For example, $f_{t|r}(s_t) = p(s_t|y_1^r)$. It is well known that the filtering and smoothing densities verify respectively the recursion (4.2) and (4.4). We have $f_{1|1} = \xi_1$, and for $t > 1$,

$$f_{t|t}(s_t) = \Pi_F^{(t)} f_{t-1|t-1}(s_{t-1}) \tag{4.2}$$

where $\Pi_F^{(t)}$ denotes here the filtering operator at time t. This Markovian operator is defined for any probability density function ξ by

$$\Pi_F^{(t)}\xi(s_t) = \frac{\int a(s_t|s_{t-1})b(y_t|s_t)\xi(s_{t-1})ds_{t-1}}{\int\int \xi(s_{t-1})a(s_t|s_{t-1})b(y_t|s_t)ds_t ds_{t-1}} \tag{4.3}$$

The proof of this relation is straightforward using the independence properties together with the law of total probability and Bayes' theorem (Künsch, 2001). The smoothing recursion is written in the same way starting with $f_{T|T}(s_T|y_1^T))$ and for $t < T$

$$f_{t|T}(s_t|y_1^T) = \Pi_B^{(t)} f_{t+1|T}(s_{t+1}|y_1^T)$$

where $\Pi_B^{(t)}$ denotes the smoothing operator. For any density probability function ξ on $\mathcal{S}$, this Markovian operator is given by

$$\Pi_B^{(t)}\xi(s_t) = \int p(s_t|s_{t+1}, y_1^T)\xi(s_{t+1})ds_{t+1} \tag{4.4}$$

$$= f_{t|t}(s_t|y_1^t)\int \frac{a(s_{t+1}|s_t)}{p(s_{t+1}|y_1^t)}\xi(s_{t+1})ds_{t+1}) \tag{4.5}$$

with one step ahead prediction density

$$p(s_t|y_1^{t-1}) = \int p(s_{t-1}|y_1^{t-1})a(s_t|s_{t-1})d(s_{t-1})$$

One of the most common problem in HMM consists in reconstructing the hidden sequence $\{s_1^T\}$ for a given observation $\{y_1^T\}$. Several algorithms can be found in the literature to compute the smoothing densities $f_{t|T}$ reconstruct the hidden states time series of state space models (Ephraim and Merhav, 2002). When the state space $\mathcal{S}$ is finite, all the integral in filtering and smoothing recursions are simply sums and the calculation is straightforward. The algorithm is then referred to as forward-backward algorithm. In this case, the Viterbi algorithm returns the most likely sequence $\{s_1^T\}$ for a given observation $\{y_1^T\}$. Another case where the general recursions simplify considerably is the linear state space model with Gaussian innovations. In this case the algorithm is the well known Kalman filter. And, Monte Carlo methods such as particular filtering permits to approximate general state space models.

4.1.2 Non parametric approximation of the state space model

Lets us now describe the proposed method for estimating the state space model parameters as well as the filtering and smoothing densities.

Assume that (S, Y) is a state space process satisfying model **(M1)-(M2)**. Suppose also that we have a realization $(\hat{s}_1^n, \hat{y}_1^n)$ of (S, Y) and a realization $\{y_1^T\}$ of $\{Y\}$ such that

- $\hat{s}_1^n$ and $\hat{y}_1^n$ are observed. State S is not hidden as this stage. $(\hat{s}_1^n, \hat{y}_1^n)$ is considered as the learning time series and it will be used to estimate the model parameters.
- y_1^T is observed while s_1^T is hidden.

Our goal consists in reconstructing s_1^T given y_1^T.

When the smoothing density $f_{t|T}(s_t|y_1^T)$ is known, the state s_t can be approximated by $s_t^* = E(S_t|Y_1^T = y_1^T)$ for each time $t \in \{1, \cdots, T\}$.

If S and Y are simultaneously observed during a given period, as it is the case with $(\hat{s}_1^n, \hat{y}_1^n)$, then a and b can be estimated by kernel density estimators as follows. Estimators of densities $f_{t|t}$ and $f_{t|T}$ can be deduced for all t and more genreally estimators of the forward and backward operators $\Pi_F^{(t)}$ and $\Pi_B^{(t)}$.

We define

$$\begin{aligned}
\hat{f}(s) &= h_1(n)^{-1} \sum_{k=1}^{n} K_1((s - \hat{s}_k)h_1(n)^{-1}) \\
\hat{q}_1(s', s) &= h_2(n)^{-2} \sum_{k=2}^{n} K_2((s' - \hat{s}_{k-1}, s - \hat{s}_k)' h_2(n)^{-1}) \\
\hat{q}_2(s, y) &= h_3(n)^{-(d+1)} \sum_{k=2}^{n} K_3((y - \hat{y}_k, s - \hat{s}_k)' h_3(n)^{-1})
\end{aligned}$$

$\hat{f}$ is an estimator of the stationary density function of the Markov chain $\{S_t\}$, $\hat{q}_1$ is an estimator of the joint probability density function of (S_{t-1}, S_t) and $\hat{q}_2$ is an estimator of the joint probability density function of (Y_t, S_t). So that, we deduce estimators of transition and emission densities as follows:

$$\hat{a}(s'|s) = \frac{\hat{q}_1(s, s')}{\hat{f}(s)} \text{ if } \hat{f}(s) > \gamma n^{-1}, \quad \hat{a}(s'|s) = 0 \text{ otherwise} \tag{4.6}$$

$$\hat{b}(y|s) = \frac{\hat{q}_2(y, s)}{\hat{f}(s)} \text{ if } \hat{f}(s) > \gamma n^{-1}, \quad \hat{b}(y|s) = 0 \text{ otherwise} \tag{4.7}$$

where K_1, K_2 and K_3 are kernel functions with standard and where $h_1(n)$, $h_2(n)$ and $h_3(n)$ are bandwidth parameters.

In practice, $\hat{a}$ and $\hat{b}$ are computed using Epanechnikov kernels by tridensker.m or kde_loc.m function. Multivariate kernels are defined as products of univariate kernels. The bandwidth parameters can be specified as inputs of the estimation functions or they are computed as follows. Each univariate sample is tansformed to a Gaussian sample, the optimal bandwidth is computed and an inverse transformation is applied (see kde.m function of WAFO or (Silverman, 1986) for more details).

Now, given $\hat{a}$ and $\hat{b}$, one can deduce non parametric estimators of the filtering operator for all observations y_1^t by subtituting a and b by $\hat{a}$ and $\hat{b}$ in equation (4.3):

$$\hat{\Pi}_F^{(t)} \xi(s_t) = \frac{\int \hat{a}(s_t|s_{t-1}) \hat{b}(y_t|s_t) \xi(s_{t-1}) ds_{t-1}}{\int \int \xi(s_{t-1}) \hat{a}(s_t|s_{t-1}) \hat{b}(y_t|s_t) ds_t ds_{t-1}} \tag{4.8}$$

and we have for $t > 1$

$$\hat{f}_{t|t}(s_t|y_1^t) = \hat{\Pi}_F^{(t)} \hat{f}_{t-1|t-1}(s_{t-1}|y_1^{t-1})$$

For $t = 1$, $\hat{f}_{1|1}$ is chosen equal to ξ_1.

In the same way, non parametric estimators of the smoothing densities can be defined.

$$\hat{\Pi}_B^{(t)} \xi(s_t) = \hat{f}_{t|t}(s_t|y_1^t) \int \frac{\hat{a}(s_{t+1}|s_t)}{\hat{f}_{t+1|t}(s_{t+1}|y_1^t)} \xi(s_{t+1}) ds_{t+1}) \tag{4.9}$$

with

$$\hat{f}_{t+1|t}(s_{t+1}|y_1^t) = \int \hat{a}(s_{t+1}|s_t)\hat{f}_{t|t}(s_t|y_1^t)ds_t$$

In practice, the continuous space process S and Y are discretized in order to use discret space algorithm that are less expensive for computation. The discretisation step can be small and it induces no significant loss of accuracy for the applications. The forward-backward algorithm is implemented in forw_back.m function, and the complete smoothing method in HMM_smoothing.m function. More details about convergence properties of this model can be found in (Monbet *et al.*, 2005).

4.2 Example: significant wave height reconstruction for given wind

As example, we propose to reconstruct a time series of significant wave height for given wind time series.

Let us denote H the significant wave height, U the wind intensity and Φ the wind direction. We suppose that the process (H, U, Φ) can be modeled by a hidden Markov chain of order 1, with the state $S_t = H_t$ and the observation $Y_t = (U_{t-\tau}\cos(\Phi_{t-\tau}), U_{t-\tau}\sin(\Phi_{t-\tau}))$. The delay τ allows to shift the wind when the maximum of dependance between waves and wind is reached for a lag different as zero. When the significant wave height H is modeled by the state, it seems that the waves induce the wind. It is physically more natural to consider that the wind induces the waves. But, in practice in the considered problem, the wind effectively represents the observation while the wave time series is supposed to be only partially observed.

Buoy data of the National Oceanic and Atmospheric Administration (NOAA) are used in order to test the model on real data. These data are public available on NOAA's ftp server (ftp://polar.wwb.noaa.gov/). We will considered the buoy 42039 (Pensacola, Gulf of Mexico) with geographical coordinates (28.80N,86.06W). This buoy has been chosen because it is located in an area with almost no swell so that essentielly wind sea is recorded. The data files provide in situ measurements of wind intensity, wind direction and significant wave height for several years with more or less missing data depending on the periods of time. The recording time step is one hour. In practice, the observed time series is split in two parts: the first one $\{\hat{h}_t, \hat{u}_t, \hat{\phi}_t\}_{t=1,\cdots,n}$ is used to estimate the transition and emission probability density functions and the second one $\{h_t, u_t, \phi_t\}_{t=1,\cdots,T}$ is used as a validation time series.

Now we remark that, for the considered applications, it is suffisant to restore the significant wave height with a precision about some centimeters, say for instance about 10 cm. So that, the wave height can be discretized without loss of accuracy for the applications. The same remark holds for wind.

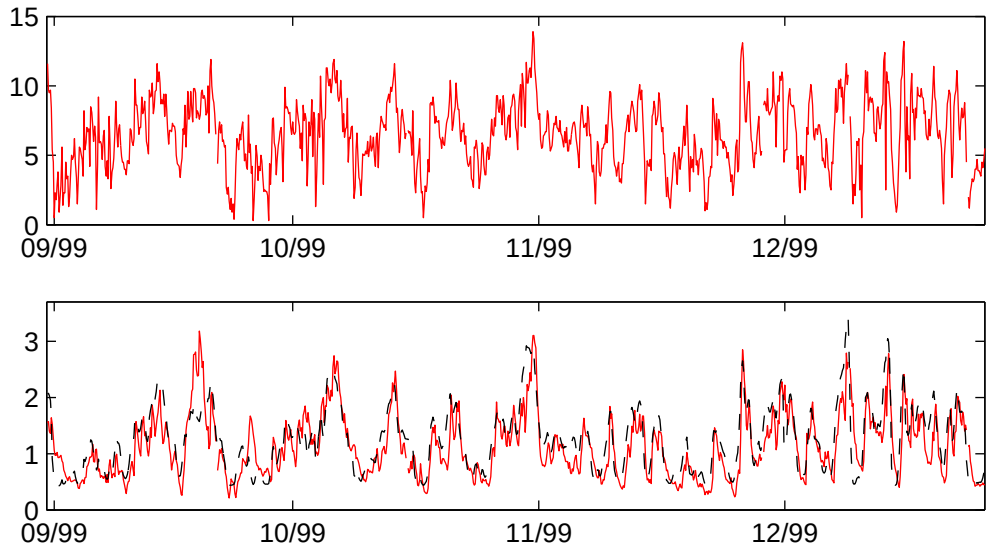

Figure 4.1: Wind intensity (top) and significant wave height (bottom). *Solid line: buoy observations, dotted line: reconstructed time series*

```
% load data
% learning sequence: years 2001 to 2003
% validation sequence : year 1999
load filtering.mat ;


Xfbd = HMM_smoothing([data(:,2:3),data(:,1)],data_val(:,2:3)) ;

figure
subplot(2,1,1) ;
plot(Wsval,'r') ;
subplot
plot(Hsval,'r') ;
hold on
plot(Xfbd,'k–') ;
```

The reconstructed significant wave height time series is compared to the observed one. In the example below, the learning time series corresponds to the available buoy data from beginning of 2000 to end 2003. And the validation data set correponds to measurements of period from 09/01/1999 to 12/31/1999. Figure 4.3 illustrates how the reconstructed time series matches to the validation data set. One remark that the reconstructed signal over estimates the low significant wave heights especiality during november, while peaks are relatively well restored. It may be observed that the reconstruction method tends to smooth the time series. It is probably due to the conditional expectation. The hidden Markov model allows to reconstruct quite well wave time series. Furthermore its computational cost is quite low in comparison to hindcast models traditionally used to reconstruct waves for given wind. It is important to remark that the model performs well in the described example because the swell is always very low in this area of the Gulf of Mexico. Clearly, the proposed model does not take into account any propagated swell.

Alternative models are linear models or artificial neural networks. For instance, filtering using a linear autoregressive model:

```
% learning sequence
lx = length(X) ;
Xx = X(1:dt:lx) ;
Yy = Y(1:dt:lx) ;
Yyd = Ydir(1:dt:lx) ;
lx = length(Xx) ;
fdnn = find( isnan(Xx(2:lx-1)) ...
.* isnan(Yy(1:lx-2)) .* isnan(Yy(2:lx-1)) .* isnan(Yy(3:lx)) ...
.* isnan(Yyd(2:lx-1))) ; AppNN = [Yy(fdnn), Yy(fdnn+1), Yy(fdnn+2), ...
cos((270-Yyd(fdnn+1))/180*pi), sin((270-Yyd(fdnn+1))/180*pi), Xx(fdnn+1) ] ;

% validation sequence lx = length(Xv(indcal)) ;
Xxv = Xv(indcal) ;
Yyv = Yval(indcal) ;
Yydv = Ydval(indcal) ;
lxv = length(Xxv) ;
fdnn = find( isnan(Xxv(2:lxv-1)) ...
        .* isnan(Yyv(1:lxv-2)) .* isnan(Yyv(2:lxv-1)) .* isnan(Yyv(3:lxv)) ...
        .* isnan(Yydv(2:lxv-1))) ;
ValNN = [Yyv(fdnn), Yyv(fdnn+1), Yyv(fdnn+2), ...
cos((270-Yydv(fdnn+1))/180*pi), sin((270-Yydv(fdnn+1))/180*pi), Xxv(fdnn+1) ] ;

l1 = length(AppNN(:,6)) ;
y = log(AppNN(:,6)+0.1) ;
Xlapp = [ones(size(AppNN(:,1))),AppNN(:,1:5)] ;
l2 = length(ValNN(:,6)) ;
Xlval = [ones(size(ValNN(:,1))),ValNN(:,1:5)] ;

[B,BINT,R,RINT,STATS] = regress(y,Xlapp) ; % for regress you need the stats toolbox

Xg = zeros(size(Xlval(:,1))) ;
Xg(1:2) = ValNN(1:2,6) ;
for k = 3:length(Xlval),
        Xg(k) = [Xlval(k,:)]*B ;
end;

Xar = ones(size(Xv(indcal)))*NaN ;
Xar(fdnn+1) = exp(Xg)-0.1 ;
```

For Artificial Neural Network, the netlab toolbox can be used. It can be download on http://www.ncrg.aston.ac.uk/netlab/down.php. In the next example, filtering is done using an Artificial Neural Network model. More precisely a multilayer perceptron is fit with 1 hidden layer, 10 hidden units and linear link function.

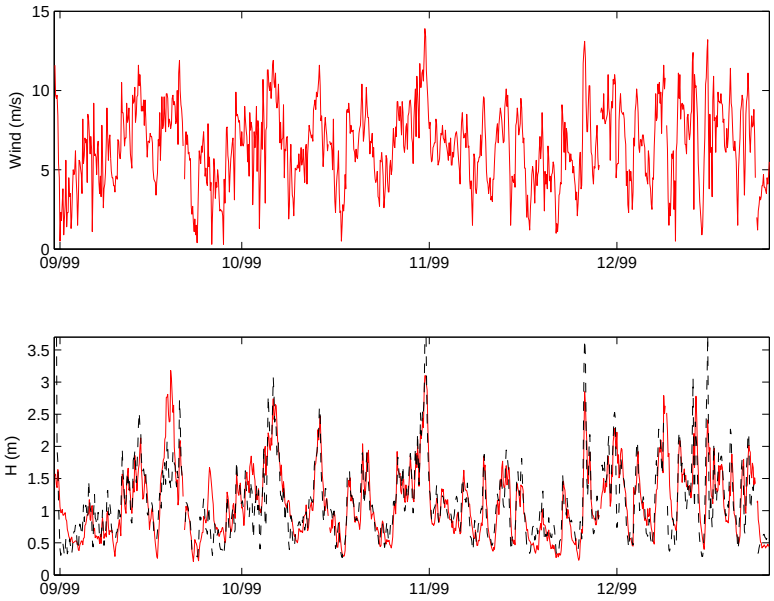

Figure 4.2: Wind intensity (top) and significant wave height (bottom). *Solid line: buoy observations, dotted line: reconstructed time series by linear model*

```
nin = 5; % Number of inputs.
nhidden = 10; % Number of hidden units.
nout = 1; % Number of outputs.
alpha = 0.01; % Coefficient of weight-decay prior.

% Create and initialize network weight vector.

net = mlp(nin, nhidden, nout, 'linear', alpha);

% Set up vector of options for the optimiser.

options = zeros(1,18);
options(1) = 1; % This provides display of error values.
options(14) = 100; % Number of training cycles.

% Train using scaled conjugate gradients.
[net, options] = netopt(net, options, AppNN(:,1:5), y,'scg');

Xnet = ones(size(Xv(indcal)))*NaN ;
Xnet(fdnn+1) = exp( mlpfwd(net,Xlval(:,2:6)) )-0.1;
```

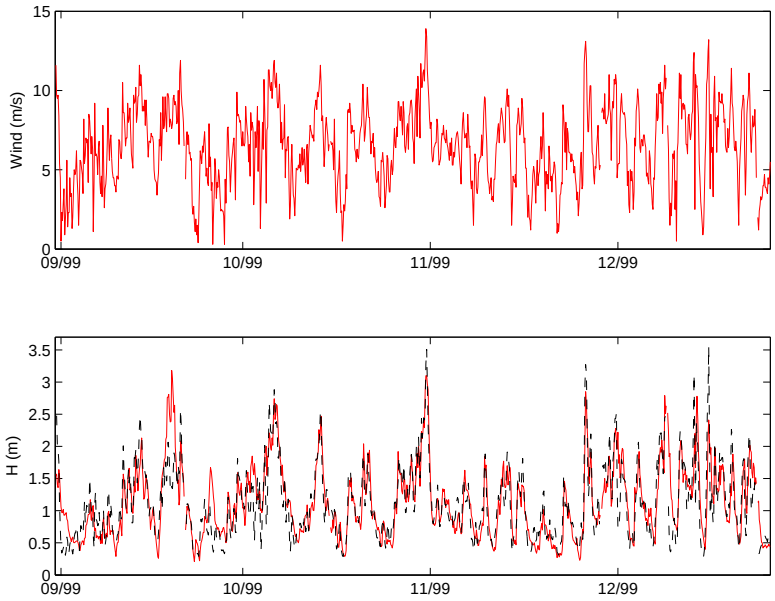

Figure 4.3: Wind intensity (top) and significant wave height (bottom). *Solid line: buoy observations, dotted line: reconstructed time series by artificial neural network*

```
figure
subplot(2,1,1) ;
plot(Yval(indcal),'r')
axis([0 976 0 15])
set(gca,'XTick',cumsum([ 1,31,32,32])*8,'XTickLabel',['09/99';'10/99';'11/99';'12/99']) ;
ylabel('Wind (m/s)')
subplot(2,1,2) ;
plot(Xv(indcal),'r')
hold on
plot(Xnet,'k–')
axis([0 976 0 3.7])
ylabel('H (m)')
set(gca,'XTick',cumsum([ 1,31,32,32])*8,'XTickLabel',['09/99';'10/99';'11/99';'12/99']) ;
```

4.3 Description of the routines

4.3.1 Filtering

HMM_smoothing non parametric smoothing for hidden Markov model
[Xfbd,gamma,th1,prior,ptrn,B] = HMM_smoothing(data,obs,h,ncl) ;

INPUTS

- data	data for the learning step
- obs	data for reconstruction
- h	bandwidth for non parametric density estimators [h_U,h_V,h_H]
- ncl	number of classes for discretisation of observation [ncl_U,ncl_V,ncl_H]

OUTPUTS

- Xfbd	reconstructed time series
- gamma	
- th1	abscissa for gamma
- prior	estimated stationary density of state
- ptrn	estimated transition matrix
- B	estimated emission density

FORW_BACK Computes the posterior probs. in an HMM using the forwards backwards algo.

[gamma, loglik] = forw_back(prior, transmat, obslik, filter_only)
'filter_only' is an optional argument (default: 0). If 1, we do filtering, if 0, smoothing.

INPUTS

- prior(I)	Pr(Q(1) = I)
- transmat(I,J)	Pr(Q(T+1)=J — Q(T)=I)
- obslik(I,T)	Pr(Y(T) — Q(T)=I)

OUTPUTS

- gamma(i,t)	Pr(X(t)=i — O(1:T))
- xi(i,j,t)	Pr(X(t)=i, X(t+1)=j — O(1:T)) t ¡= T-1

See also HMM toolbox

4.3.2 Non parametric estimation

KDE_LOC Local kernel density estimator (kde)
This function proposes a combination of kde and nearest neighbour approachs for non parametric estimation of conditionnal probability P(x—y=ty).

[p,tx1,tx2,ty1,ty2,ty3] = kde_loc(x,y,dx,dy,dcl,nbn) ;

INPUTS

- x	of size n*mx with mj=2
- y	of size n*my with mj=3
- dx, dy	discretisation step (size 1*mx, resp. 1*my) or abscissa for computation of estimators
- dcl	maximal distance between the current point and the nearest neibourghs
- nbn	number of nearest neighbours

OUTPUTS

- p	estimation of the density
- tx1,tx2,ty1,ty2,ty3	abscissa (some of the vectors may be empty)

TRIDENSKER kernel density estimator in 3D with a polynomial kernel of degree 2 (Epanechnikov). In practice,

$$f(t) = 1/n/h^2 \sum_i K\left(\frac{t - X(i)}{h}\right)$$

with $t = (tx, ty, tz)'$, $X = (x, y, z)'$ and $h = (hx, hy, hz)$ and $K(x) = (1 - x'x)^2$ if $x'x \in [-1, 1]$, $K(x) = 0$ otherwise.

[f,tx,ty,h,h1] = tridensker(x,y,z,[nx,[ny,[nz,[h);

INPUTS

- x,y,z	samples of length n
- nx,ny,nz	number of bins or abscissa for kde computation
- h	bandwidth parameters h = [hx,hy,hz] if length(h) = 1, hx=hy=hz=h (default hx=hy=hz=$2.04/n^{1/6}$)

OUTPUTS

- f	kde
- tx,ty,tz	abscissa for f
- h	bandwidth parameter

See also the routines of WAFO toolbox

Chapter 5

Validation

5.1 Introduction

We will now focus on the statistical validation of simulation models.

The most widespread method for model validation consists in comparing certain statistics calculated from the observations with those corresponding to the model. In general, several criteria are used, such as the matching of the mean and the variance of the marginal distributions, or more generally its cdf. When the temporal dependence is important for the applications, other features are also considered, like the autocorrelation functions or the distribution of the time duration of sojourns above given levels.

Meanwhile, the authors often only perform visual comparisons. Such approach remains are not entirely satisfactory because it does not make it possible to decide whether observed differences are significant or not. A more formal method, based on Monte Carlo tests, is proposed below. For the sake of simplicity, it is presented in the simple case of comparing means, but its generalization to other statistics is not difficult.

Let $\{y_t\}_{t=1,\cdots,T}$ be an observed sequence of a real valued process $\{Y_t\}$ with mean m and $\{Z_t\}$ a process corresponding to the model which has to be validated. The mean of the marginal distribution of $\{Z_t\}$ is denoted m_0. We want to test

$$H_0 : m = m_0 \quad \text{versus} \quad H_1 : m \neq m_0 \tag{5.1}$$

The considered test statistics is the empirical mean $\overline{Y} = \frac{1}{T}\sum_{t=1}^{T} Y_t$, and the associated decision rule is given by

$$H_0 \text{ is rejected if } \overline{y} = \frac{1}{T}\sum_{t=1}^{T} y_t \in R(\alpha)$$

where α is the level of the test. The critical region $R(\alpha)$ is such that $P_{H_0}(\overline{Y} \in R(\alpha)) = \alpha$.

In order to compute $R(\alpha)$, we need to know the distribution of the test statistic $\overline{Y}$ when H_0 is true. When the model is complex, it is not always possible to derive the exact distribution of the test statistic. In this case, we can use a Monte Carlo method to approximate this distribution as follows:

1. Simulate B time series of length T with the model:

$$\begin{matrix} \{z_1^{(1)}, & \cdots & , z_T^{(1)}\} \\ & \vdots & \\ \{z_1^{(B)}, & \cdots & , z_T^{(B)}\} \end{matrix}$$

2. Compute the empirical mean $\overline{z}^{(i)}$ corresponding to $\{z_1^{(i)}, \cdots, z_T^{(i)}\}$ for $i = 1, \cdots, B$

3. Approximate the cdf of $\overline{Y}$ by the empirical cdf of $\{\overline{z}^{(1)}, \cdots, \overline{z}^{(B)}\}$. This allows to compute an approximation of $P_{H_0}(\overline{Y} \in R)$ for all $R \subset \mathbb{R}$ or equivalently deduce an approximative critical region $\tilde{R}(\alpha)$ such that $\frac{1}{B} card(\{i \in \{1, \cdots, B\} | \overline{y}^{(i)} \in \tilde{R}(\alpha)\}) = \alpha$.

This framework can be applied to compare other feature like cdf or autocorrelation functions. For this, a test statistic has to be chosen. As concern cdf, the most popular test statistic is probably the Kolmogorov-Smirnov distance. However, it is well known that it is more sensitive near the center of the distribution than at the tails. Due to this limitation, many analysts prefer to use the Anderson-Darling statistic which gives more weight to the tails. But, as this is an integrated deviation, like the Cramer-Von-Mises or chi-square distance, it can mask local differences.

Firstly, in order to measure the goodness of fit of the distribution at the different locations x, we have computed the p-values $p_v(x)$ of the bilateral test with null hypothesis "$F(x) = F_0(x)$" for different values of x, e.g.:

$$p_v(x) = 2\min(P_{H_0}(F_0(x) > \hat{F}(x)), P_{H_0}(F_0(x) < \hat{F}(x)))$$

where $\hat{F}(x)$ denotes the observed distribution function. Then, in order to measure the global agreement between F and F_0 we have used the test statistic

$$S = \inf_{\{x | t_1 < F_0(x) < t_2\}} p_v(x) \tag{5.2}$$

with $0 < t_1 < t_2 < 1$. The minimum is taken on the interval $\{x | t_1 < F_0(x) < t_2\}$ because the variability of the sample estimate of $F_0(x)$ is null under H_0 if $F_0(x) = 0$ or $F_0(x) = 1$. The generalization of this procedure to other functional statistics, like the autocovariance functions, is immediate and is not given here.

Such a statistical test consists, actually, in a tool or an alarm that detects a mismatch between the model and the observed sample for the considered parameter. Then, in order to interpret more precisely the lack of fit, a figure can be plotted. See for instance Figure 5.1 on which, for all x, a 95% fluctuation interval $I(x)$ is added to help to compare the distribution functions $F(x)$ and $F_0(x)$ at x tacking into account the variability due to the estimation of the distribution function.

5.2 Example

Let us denote U an observed wind speed time series and Ug an artificial sequence. The matlab command lines below produces a list of figures like Figure 5.1 and returns the test statistics `pv` of the test U and Ug have the same statistical features. According to the applications, a list of criteria may be retained in the validation task as features which must be restored in the simulated time series:

- F_U: cdf of U
- C_U: autocorrelation function
- F_{extr}: cdf of monthly maxima of U
- $F_{[U>2/3]}$: cdf of time duration of sojourns above level $2/3 \max(u)$, with $\max(u)$ the largest wind speed in the observed time series (about 21 ms^-1)
- $F_{[U<2/3]}$: cdf of time duration of sojourns under level $2/3 \max(u)$
- $F_{[U<1/3]}$: cdf of time duration of sojourns under level $1/3 \max(u)$

Cdf F_U is the most important criterion for many applications since the repartition of the wind speed constraints the evolution of the phenomenon. The autocorrelation function C_U permits to measure the linear dependence in time. F_{extr} describes the interannual variability of the process. The distribution functions of sojourn durations $F_{[Y>2/3]}$, $F_{[Y<2/3]}$ and $F_{[Y<1/3]}$ describe the time duration of the storms and inter-arrival between storms. Such criteria are important for fatigue studies or erosion, for instance. Indeed, the way the storms succeed has an impact on the behavior of dynamical systems: for example, the rapid succession of two storms may cause more damages than if they are separated by a long calm weather.

```
alpha = 0.05 ;
[pv,CI] = stat_valid_univ(U,Ug,alpha,1) ;
```

Third parameter alpha determines the risk for computation of the fluctuation intervals CI for the test statistics and also for the fluctuation intervals of the figures. For each selected criteria a Monte Carlo test has been run on the basis of $N = 1000$ synthetic time series, each of them having the same length as the initial wind time series. The critical region of the tests is computed as well as the 95% interquantile range like the one plotted on Figure 5.1.

We propose here an example where two models are compared for the wind intensity: translated Gaussian process (TGP) and Marokov switching autoregressive model (MS-AR). Table 5.1 is build using the results of stat_valid_univ calls and it gives, for each considered model and each feature, the value s^{obs} of the test statistic corresponding to the data and the cut-off value s_α for a risk $\alpha = 0.05$. The null hypothesis is rejected at level α if $s^{obs} < s_\alpha$.

	TGP	MS-AR
F_U	0.808 [0.012]	0.641 [0.024]
C_U	0.062 [0.012]	0.086 [0.022]
F_{extr}	0.068 [0.008]	0.371 [0.022]
$F_{[U>2/3]}$	0.053 [0.012]	0.080 [0.016]
$F_{[U<2/3]}$	0.002 [0.006]	0.053 [0.009]
$F_{[U<1/3]}$	0.124 [0.031]	0.346 [0.004]

Table 5.1: Numerical results. The first value is the observed statistic s^{obs} and the value in bracket the cut-off value s_α with $\alpha = 0.05$. The null hypothesis is rejected at the level α if $s^{obs} < s_\alpha$

Globally both models restore all the selected features except TGP which fails to reproduce the cdf of the interarrival of storms. In order to interpret this lack of fit, we have plotted the corresponding cdf on Figure 5.1. It shows that TGP simulates too few short interarrival durations. In other words, it can not reproduce the rapid succession of storms. This non linearity is successfully reproduced by MS-AR model.

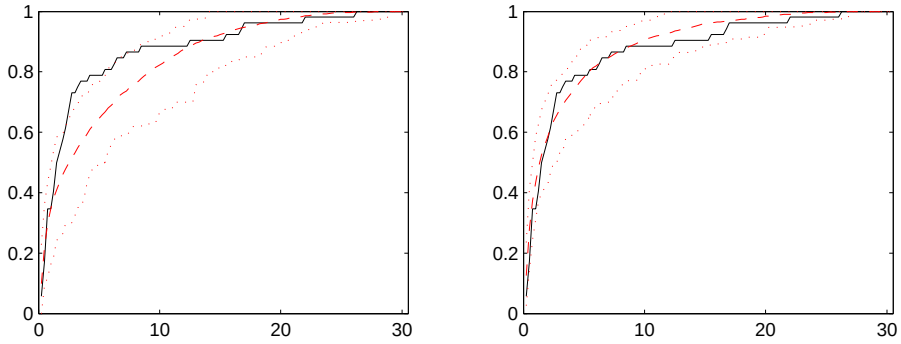

Figure 5.1: Comparison of the cdf of time duration of interarrival of storms. Left: TGP - Right: MS-AR. Solid: observation, dashed: simulated, dotted 95% interquantile interval. The time in abscissa is expressed in days.

5.3 Short description of the routines

Routines are proposed to compare univariate and bivariate process.

- stat_valid_univ: statistical comparison absed on Monte Carlo tests for univariate processes, figures can be plotted
- graph_valid_univ: graphical comparison for univariate processes
- stat_valid_bivar: statistical comparison absed on Monte Carlo tests for univariate processes, figures can be plotted
- graph_valid_bivar: graphical comparison for univariate processes
- hist_circ: compares the histogram of two variables on a circle
- couple: computes and plot joint pdf of a bivariate process

For univariate processes U, the list of statistics just below is retained to compare the models in the stat_valid_univ function.

1. F_U: distribution function of U
2. F_{extr}: distribution of monthly maxima of U
3. $F_{[U>2/3]}$: distribution of time duration of sojourn above level $2/3 \max(U^{obs})$, with $\max(U^{obs})$ the largest wind intensity in the observed time series
4. $F_{[U<2/3]}$: distribution of time duration of sojourn under level $2/3 \max(U^{obs})$
5. $F_{[U<1/3]}$: distribution of time duration of sojourn under level $1/3 \max(U^{obs})$
6. C_U: autocovariance function of U

For bivariate processes (U, Φ), with U an intensity and Φ a direction following statistics are added in the stat_valid_bivar function.

1. F_Φ: distribution function of Φ
2. $F_{(U,\Phi)}$: distribution function of (U, Φ)

3. C_u: autocovariance function of $u = U\cos(\Phi)$
4. C_v: autocovariance function of $v = U\sin(\Phi)$

In graph_valid_bivar the joint probability density function is plotted as the four fist conditional moments.

Appendix A

Contents of the toolbox

- TRANSLATED GAUSSIAN PROCESS ————————————
  - SIMGAUSSPR Simulation of a multivariate stationary Gaussian process
  - ROZENBLAT Like bivariate Rozenblatt transformation transform a couple of dependant variable (x,y) into a couple of independant variables (u,v) with given distributions
  - INVROZENBLAT Inverse of like bivariate Rozenblatt transformation
  - GAUS2NGAUS Inverse normal score transformation for bivariate processes
  - DEMOTGP Demonstration of TGP method for bivariate process (Hs,Hd)
  - SPECTRE Estimates the one sided spectrum of x
- MS-AR ————————————————————————
  - *Gaussian*
    - GAUSSIAN_PRO Evaluate the multi-variate density with mean vector m and covariance
    - AR_EM estimation of the parameters of a Gaussian autoregressive model
    - COND_INI Definition of initial conditions for Gaussian AR model with
    - E_STEP Estimation step (Forward-Backward) of the Baum-Welch algorithm for a Gaussian AR model with Markovian switching.
    - mk_ARgaussian calculates the conditional likelihood for Gaussian AR model for each hidden state.
    - simule_AR simulation of an AR process with markovian switching.
    - simule_AR2 simulation of an AR process with markovian switching.
    - simule_CM generation of a Markov chain
  - *Gamma or lognormal*
    - ARgamma Estimation of GARHMM model
    - LOG_DENS Computes the log likelohood of of Gamma (or lognormal) AR model
    - LOG_LL Calculates the log likelohood and its gradient for Gamma (or lognormal) AR model
    - FB_GAMMA Runs forward-backwards algorithm for GARHMM
    - ll_ARgamma Log likelihood to be maximized to identify the GARHMM model
    - COND_INI_GAMMA Random choice of initial conditions for GARHMM
    - FORW_GAMMA Computes the filtered probs. of MS-AR using the forwards algo.
    - simule_ARgamma Simulation of an GARHMM process with markovian switching.

  - *Non homogeneous model*
    * ARnhgamma Estimation of non homogeneous GARHMM model
    * FB_NHGAMMA Runs forward-backwards algorithm for non homogeneous GARHMM
    * MK_TRANSIT Computation of the transition matrix for the non homogeneous
    * MK_TRANSITION Computation of the transition matrix for the non homogeneous
    * VON_MISES_PDF Probability density fucntion of von Mises
    * PARA_TRANS Parametrisation of a stochastique transition matrix
    * PARA_TRANS_INV Inverse parametrisation of a stochastique transition matrix
    * FOLDING Folding of the parameters for the maximisation
    * UNFOLDING Unfolding of the parameters for the maximisation
    * LL_HMM_INP Computes the log likelihood to estimate the nh Markov chain
    * NHVITERBI Viterbi algorithm for HN GARHMM model
    * SIMULE_NHCM Simulation of a non homogeneous Markov chain

- FILTERING ————————————————————

  - FORW_BACK Computes the posterior probs. in an HMM using the forwards backwards algorithm
  - HMM_smoothing non parametric smoothing for hidden Markov model
  - KDE_LOC Local kernel density estimator (kde)
  - READ_BUOY_DATA read buoy data of NOAA
  - TRIDENSKER kernel density estimator in 3D

- VALIDATION ————————————————————-

  - STAT_VALID_UNIV statistical validation for time series simulation
  - STAT_VALID_BIVAR statistical validation for bivariate time series simulation (intensity, direction)
  - GRAPH_BICOV plot covariance functions of cartesian components
  - GRAPH_CONDMOM plots conditional moments for bivariate time series
  - GRAPH_JPDF plots jpdf of cartesian components
  - GRAPH_VALID_UNIV graphical validation for time series simulation
  - GRAPH_VALID_BIVAR graphical validation for bivariate time series simulation (intensity, direction)
  - HIST_CIRC compares the histogram of x and y on a circle
  - HISTOM computation of abscissa and ordinates for histograms
  - CDF_EMPIR empirical cumulative distribution function
  - CONDMOM compute the first conditional moments of intensity given direction
  - COUPLE computes and plot joint pdf of a bivariate process
  - CROSS_COV cross covariance function for bivariate processes
  - PERS_DIR_ALL duration of all sejourn above a given level
  - PERS_DIREC Time duration of all sejourn above a given level and corresponding mean direction
  - BIDENSKER kernel estimate of jpdf for couple of variables

Bibliography

[1] Ailliot, P., Prevosto, M., (2001). Two methods for simulating the bivariate process of wave height and direction. *Proc. of ISOPE Conf.*, **III**, 15-18.

[2] Ailliot, P., Prevosto, M., Soukissian, T., Diamanti, C., Theodoulides, A., Politis C., (2003). Simulation of sea state parameters process to study the profitability of a maritime line. *Proc. of ISOPE Conf.*, **III**, 51-57.

[3] Ailliot, P., (2004). *Modèles autorégressifs à changement de régimes markovien - Application à la simulation du vent.* PhD Thesis. Université de Rennes 1.

[4] Ailliot, P., Monbet, V., (2004). A non homogeneous Markov switching model for wind time series. *preprint*

[5] Arena, F., Puca, S., Tirozzi, B., (2002). A new approach for the reconstruction of significant wave height time series. *Proc. OMAE.*

[6] Baxevani and I. Rychlik (2004). Fatigue Life Prediction for a Vessel Sailing the North Atlantic Route. *Proc. ISOPE PACOMS* .

[7] Borgman L.E., Sheffner N.W. (1991). *Simulation of time sequences of wave height, period and direction.* Technical report US Army Corps of Engineers.

[8] Box, G.E.P., Jenkins, G.M., (1976). *Time series analysis, forecasting and control.* (revised edn.) Holden-Day, San Francisco.

[9] Breckling, J. (1989). *The Analysis of Directional Time Series.* Lecture Notes in Statistics Series.

[10] Brockwell, P., Davis, R., (1991). *Time series: theory and methods, 2nd edition.* Springer Verlag, New York.

[11] Brown, B.G., Katz, R.W, Murphy, A.H. (1984). Time series models to simulate and forecast wind speed and wind power. *J. of clim. and appl. meteor.* **23**, 1184-1195.

[12] Buishand, T.A., Brandsma, T., (2001). Multi-site simulation of daily precipitation and temperature in the Rhine basin by nearest-neighbor resampling. *Water Resources Research*, **37**, 2761-2776.

[13] Caires, S., Sterl, A., (2005). A new non-parametric method to correct model data: Application to significant wave height from the ERA-40 reanalysis. *J. Atmospheric and Oceanic Tech., In press.*

[17] Castino, F. , Festa, R., Ratto, C.F. (1998). Stochastic modelling of wind velocities time series, *J. of Wind Engineering & Industrial Aerodynamics*, **74-76**, 141-151.

[15] Chan, K.S., Tong H., Stenseth N.C., (1997). Analyzing abundance data from periodically fluctuating rodent populations by threshold models: a nearest block bootstrap approach. *Technical Report, No 258, Depart. of Statistics and Actuarial Science, University of Iowa.*

[16] Cunha, C., Guedes Soares, C. (1999). On the choice of data transformation for modelling time series of significant wave height. *Ocean Engng*, **26**, 489-506.

[17] Castino F., Festa R., Ratto C.F. (1998). Stochastic modelling of wind velocities time series. *Journ. Wind Eng.*, **74-76**, 141-151.

[18] Daniel, A.R., Chen, A.A. (1991). Stochastic simulation and forecasting of hourly average wind speed sequences in jamaica. *Solar Energy*, **46**(1), 1-11.

[19] DelBalzo D.R., Schultz, J.R., Marshall, D.E., (2003). Stochastic time-series simulation of wave parameters using ship observations. *Ocean Engng.* **30**(11), 1417-1432.

[20] Deo, M.C., Jha, A., Chapekar, A.S., Ravikant, K., (2001). Neural network for wave forcasting.*Ocean Engng*, 28, 889-898.

[21] Efron, B., Tibshirani, R., (1993). *An introduction to the bootstrap*. Chapman Hall, New York.

[22] Fisher, N.I, Lee, A.J. (1994). Time series analysis of circular data. *J.R Statist. Soc. B*, **56**(2), 327-339.

[23] Gioffre M., Gusella V., Griogriu M. (2000). Simulation on non-Gaussian field applied to wind pressure fluctuations. *Probabilistic Engineering Mechanics*, **15**, 339-345.

[24] Guedes Soares, C., Ferreira, A.M., (1996). Representation of non-stationary time series of significant wave height with autoregressive models, *Prob. Engng. Mech.*, **11**, 139-148.

[25] Guedes Soares, C., Cunha, C., (2000). Bivariate autoregressive models for time series of significant wave height and mean period, *Coastal. Engng.*, **40**, 297-311.

[26] Gurney, K., (1997). *An Introduction to Neural Networks*, UCL Press.

[27] Hamilton, J.D, (1989). A New Approach to the Economic Analysis of Nonstationary Time Series and the Business Cycle,*Econometrica*, **57**, 357-384.

[28] Härdle, W., Horowitz, J., Kreiss, J.P., (2003). Bootstrap methods for time series. *Int. Stat. Review* **71**(2). 435-459

[29] Huang, Z., Chalabi, Z.S., (1995). Use of time series analysis to model and forecast wind speed. *J. Wind Eng. Ind. Aerodyn.*, **56**, 311-322.

[30] Ho, P.C., Yim, J.Y., (2005). A study of the data trnasferability between two wave-measuring stations. *Coastal Engng, In press.*

[31] Hogben, N., Standing, R.G., (1987). A Method for Synthesising Time History Data from Persistence Statistics and its use in Operational Modelling. *Underwater Technology, Society for Underwater Technology*, **13**(4), 11-18.

[32] Hugue, J.P, Guttorp, P., Charles, S.P., (1999). A non homogeneous hidden markov model for precipitation occurrence, *Appl. Statit.*,**48**, 15-30.

[33] IAHR, (1989). List of sea state parameters, *J. Waterway Port Coast. Ocean. Eng.*, **115** (6), 793-808.

[34] Izquierdo, P., Guedes Soares, C., (2005). Analysis of sea waves and wind from X-band radar. *Ocean Engng, In press.*

[35] Jenkins, A.D., (2002). Wave duration/persistence statistics, recording interval, and fractal dimension. *International Journal of Offshore and Polar Engineering*, **12**(2), 109-113. 2002

[36] Lall,U., Sharma, A., (1996). A nearest neighbor bootstrap for resampling hydrologic time series. *Water Ressources Res.*, **32**, 679-693.

[37] Liu, R.Y., Singh K. (1992). Moving blocks jackknife and bootstrap capture weak dependence In , *Exploring the limits of the bootstrap* (Eds R. LePage and L. Billiard), pp. 224-248. New York: Wiley.

[38] MacDonald, I. L., Zucchini W., (1997). *Hidden Markov and Other Models for Discrete-Valued Time Series*. New York: Chapman and Hall.

[39] Makarynskyy, O., Pires-Silva, D., Makarynska, D., Ventura-Soares, C., (2004). Artificial neural networks in wave prediction at the west coast of Portugal, *Computer & Geosciences, In press*

[40] Marteau, P.F., Monbet, V., Ailliot, P., (2004a). Non parametric modeling of cyclo-stationary markovian processes - part 2. *Proc. ISOPE Conf.*, **III**, 145-151.

[41] Marteau P.F., Monbet V., (2004b). Conditional prediction of Markov processes using non parametric Viterbi algorithm - Comparison with MLP and GRNNmodels, *WSEAS Trans. on systems*, **3**(2), 346-351.

[42] Monbet, V., Prevosto, M., (2001a). Bivariate Simulation of Non Stationary and Non Gaussian Observed Processes . Application to Sea State Parameters. *Applied Ocean Research*, **23**, 139-145.

[43] Monbet V., Marteau P.F., (2001b). Continuous Space Discrete Time Markov Models for Multivariate Sea State Parameter Processes. *Proc. ISOPE Conf.*, **III**, 10-14.

[44] Monbet V., Marteau P.F., (2004). Non parametric modeling of cyclo-stationary markovian processes. *Proc. ISOPE Conf.*, **III**, 138-144.

[45] Monbet V., Marteau P.F., (2005a). The Local Grid Bootstrap for Stationary Multivariate Markov Processes, *J. Statistical Planning and Inference, in press.*

[46] Monbet V., Ailliot P., (2005b). L^1-convergence of smoothing densities
in non parametric state space models *submitted to Stat. Inf. for Stoch. Proc.*

[47] More, A., Deo, M.C., (2003). Forecasting wind with neural networks. *Marine structures*, **16** , 35-49.

[48] Nfaoui, H., Buret, J., Sayigh, A.A., (1996). Stochastic simulaton of hourly average wind speed sequences in Tangiers (Morocco). *Solar Energy*, **56**(3), 301-314.

[49] O'Carroll, (1984). Weather Modelling for Offshore Operations. The Statistician, 33, pp 161-169.

[50] Pittalis, S., Bruschi, A., Puca, S., Tirozzi, B., (2003). Reconstruction of sea events and extreme value analysis. *Proc. ISOPE Conf.*, **III**.

[51] Poggi, P., Muselli, M., Notton, G., Cristofari, C., Louche, A., (2003). Forcastinf and simulating wind speed in Corsica by using an autoregressive model. *Energy conversion and management* **44**, 3177-3196.

[52] Popescu, R., Deodatis, G., Prevost, J.H. (1998). Simulation of homogeneous nonGaussian stochastic vector fields. *Prob. Engng. Mech*, **13**(1), 1-13.

[53] Rynkiewicz, J. (2000). Estimation de modèles autorégressifs à changements de régime markovien. *Cahiers de la MSE*, **60**.

[54] Sahin A.J, Sen Z. (2001). First-order Markov chain approach to wind speed modelling. *J. of Wind Eng. and Indust. Aerodyn.*, **89**, pp 263-269.

[55] Scheffner, N.W., Borgman, L.E., (1992). Stochastic time-series representation of wave data. *J. Waterway, Port and Ocean Engineering*, **118**(4), 337-51.

[56] Scotto, M.G., Guedes Soares, C., (2000). Modelling the long-term time series of significant wave height with non linear threshold models. *Coastal Engng*, **40**, 313-327.

[57] Shi S.G., (1991). Local Bootstrap. *Ann. Institut. Statist. Math.*, **43**, 667-676.

[58] Soukissian, T., Theochari, Z., (2001). Joint occurrence of sea states and associated durations, *Proc. ISOPE Conf.*, **III**, 33-39.

[59] Stephanakos, C.N., (1999). *Nonstationary stochastic modelling of time series with applications to environmental data.* PhD Thesis. NTAU.

[60] Stephos, A., (2000). A comparison of various forecasting techniques applied to mean hourly wind time series. *Renewable Energy*, **21**, 23-35.

[61] Toll, R.S.J., (1997). Autoregressive conditional heteroscedasticity in dayly wind speed measurements, *Theor. Appl. Climatol.*, **56**, 113-122.

[62] Tsai, C.P., Lin, C., Shen, J.N., (2002). Neural network for wave forcasting amoung multi-stations, *Ocean Engineering*, 29, 1683-1695.

[63] Turlach, B.A., (1993). Bandwidth selection in kernel density estimation: a review. *Techn. report.*

[64] Vik I. (1981). *The tile series generating module-Modelling aspects* Ocean Research- operational criteria, CNRD 3-2 task 2.

[65] Waeles B., Le Hir P., Silva Jacinto R. (2004). Modélisation morphodynamique cross-shore d'un estran vaseux. *C. R. Geoscience*, **336**, 1025-1033.

[66] Walton, T.L., Borgman, L.E. (1990). Simulation of non-stationary, non-gaussian water levels on the great lakes, *J. of Waterways, Ports, Coastal and Ocean Division, ASCE*, **116**(6).

[67] Woolf, D.K., Challenor, P.G., (2002). Statistical comparison of satellite and model waves climatologies, *Proc. 4th symp. WAVES*

[68] Yim, J.Z., Chou, C., Ho, P., (2002). A study on simulating the time series of significant wave height near the keelung harbor. *Proc. ISOPE Conf.*, vol III.

[69] Young, P.C., NG, C.N., Lane, K., Parker, D., (1991). Recursive forcasting, smoothing and seasonal adjustement of non-stationary environmental data, *J. Forecast*, **10**, 58-89.

[70] Young, K.C., (1994). A multivariate chain model for simulating climatic parameters from daily data, *J. Appl. Meteorol.*, 33, 661-671.