M2IM, Séries Chronologiques
2010-2011

Travaux Pratiques n°1

Introduction

Le TP 1 porte sur les points suivants :
- analyse visuelle d'une série chronologique,
- désaisonnalisation par moyenne mobile,
- processus $AR(1)$,
- analyse spectrale (transformée de Fourier discrète).

On effectuera ces travaux à l'aide du logiciel `R`.

1 Acquérir et visualiser des séries chronologiques

Pour chacun des 5 exemples donnés dans le cours (grippe, indice Dow Jones, rougeole, grèves, taches solaires) :

1.1 Acquérir les données à partir du répertoire spécifié et les visualiser sur votre écran avec la commande `plot`. La commande suivante permet de représenter l'indice Dow Jones avec une échelle logarithmique :

```
plot(dj,main="Dow Jones (ordonnée logarithmique)",col="blue",log="y")
```

1.2 Décrire sommairement ces séries. Notamment, discuter l'existence d'une tendance et d'une saisonnalité, l'existence d'un 'bruit' et sa stationnarité éventuelle.

1.3 Pour l'exemple rougeole, extraire une tendance débarrassée des variations saisonnières en utilisant une moyenne mobile, et représenter la série originale et la série lissée sur un même graphique. Faire la même opération sur l'exemple grippe. Commenter.

2 Le processus de marche au hasard et un $AR(1)$ très corrélé

2.1 Écrire un programme générant une trajectoire de longueur n, $(X_1, \ldots, X_n)$ d'un processus de type $AR(1)$:

$$X_t = Z_t + \phi_1 X_{t-1} \ , \quad \forall t \in \mathbb{Z} \ ,$$

où (Z_t) est un bruit blanc gaussien de variance σ^2 et ϕ_1 est un nombre réel.

2.2 Visualiser plusieurs réalisations de ce processus sur le même graphique (la fonction `matplot` est utile) lorsque $\phi_1 = 1$. Refaire la même opération lorsque $\phi_1 = 0.9$. Commenter.

3 Transformée de Fourier discrète

Faire une transformée de Fourier discrète (`fft`) du phénomène *Taches solaires* (série `sunspots.txt`). Afficher le périodogramme. Certaines fréquences vous semblent-elles plus 'importantes' que d'autres ? A quelle période correspondent-elles ? Interpréter.

Examiner de la même façon la transformée de Fourier discrète de réalisations des processus de l'exercice précédent. En particulier, faire varier ϕ_1 de 0 à 1, et commenter les résultats.

La classe `ts`.

`R` possède *une classe d'objets correspondant aux séries chronologiques*. C'est la classe `\ts` (pour *times series*). On peut convertir un vecteur `x` en objet de classe `ts` avec la commande :
`ts(x)`
Dans cette commande, deux options cruciales :

- `start` date de la première valeur
- `frequency` fréquence des observations

Par exemple, si on dispose d'observations trimestrielles et que la première observation est le second trimestre de l'année 1959, alors on précisera : `frequency=4, start=c(1959,2)`.

M2IM, Séries Chronologiques
2010-2011

Travaux Pratiques n°2

Représentation et estimation spectrale

Le TP 2 portera sur les points suivants :
- TFD de signaux simples,
- TFD d'un bruit blanc,
- simulation et densité spectrale d'un processus $MA(1)$.

1 Transformée de Fourier discrète de signaux simples

1. Générer les séries suivantes, pour $t = 1, \ldots, n$ avec $n = 200$:

$$a_t = \sin(2\pi t/10), \quad b_t = t \bmod 10, \quad c_t = 1 + t - t^2/20, \quad d_t = \sin(2t/3)$$

2. Examiner les transformées de Fourier discrètes (partie réelle, partie imaginaire et périodogramme) de ces séries. Par exemple, pour la série `a` :

```
f <- fft(a)/sqrt(n)
oldpar <- par(mfrow=c(3,1),oma=c(0,0,3,0))
plot(Re(f),main="Partie réelle de fft(a)",col="blue")
plot(Im(f),main="Partie imaginaire de fft(a)",col="blue")
plot(abs(f)^2,main="Périodogramme de a",col="blue")
mtext("TFD de a",line=1,side=3,outer=TRUE,cex=1.5)
par(oldpar)
```

Comment corriger cette suite d'instructions pour avoir des abscisses « naturelles » ? Et pour avoir la fréquence nulle au centre ?

2 TFD de bruits blancs

On définit deux bruits blancs forts. Le premier, ε, gaussien et de variance 16. Le second, $\widetilde{\varepsilon}$, de marginale la loi de Pareto recentrée de paramètres $(1, 3)$. On rappelle la fonction de distribution d'une loi de Pareto (non recentrée) de paramètres $(1, 3)$:

$$F(x) = \begin{cases} 1 - \frac{1}{x^3} & \text{si } x \geqslant 1 \\ 0 \text{ si } x < 1 \end{cases}$$

2.1 Simuler une réalisation de longueur 100 de ε, la tracer puis en examiner la transformée de Fourier discrète et le périodogramme.

2.2 Faire la même chose pour $\widetilde{\varepsilon}$.

2.3 Comparer les périodogrammes de a_t et $(a_t + \varepsilon_t)$.

2.4 Faire la même chose pour les séries b et c

3 Analyse spectrale d'un MA(1)

Soit θ un réel fixé. Soient X et $\widetilde{X}$ deux processus $MA(1)$ définis par :

$$X_t = \varepsilon_t + \theta\varepsilon_{t-1} \ ,$$

et

$$\widetilde{X}_t = \widetilde{\varepsilon}_t + \theta\widetilde{\varepsilon}_{t-1} \ ,$$

où ε et $\widetilde{\varepsilon}$ sont définis dans l'exercice précédent.

3.1 Ecrire une fonction qui génère une réalisation de longueur n d'un processus $MA(1)$ à partir de θ et d'une réalisation de longueur $n+1$ du bruit blanc le dirigeant.

3.2 Observer une réalisation de longueur 100 de X et $\widetilde{X}$ pour diverses valeurs de θ et comparer les périodogrammes de ces réalisations aux densités spectrales de X et $\widetilde{X}$.

3.3 Examiner les périodogrammes de $a+X$ et $b+X$ pour $\theta = 2$.

M2IM, Séries Chronologiques
2010-2011

Travaux Pratiques n°3

Tendances et facteurs saisonniers

Ces Travaux Pratiques porteront sur les points suivants :
- Autocorrélation et autocovariance empiriques,
- Analyse visuelle des données,
- Estimation et élimination de la tendance et du facteur saisonnier.

1 Autocovariance et Autocorrélation empiriques

Ecrire une fonction `covx` qui prend en argument un vecteur `x` (la série chronologique) et qui retourne un vecteur de longueur `length(x)` contenant les autocovariances empiriques de `x`.

2 Analyse visuelle des données

Télécharger les fichiers `airline.rda`, `copper.rda` et `oil.rda`. Ces fichiers correspondent aux données suivantes :
- airline : fréquentation internationale mensuelle des compagnies aériennes de 1949 à 1957
- copper : prix annuel du cuivre en dollars de 1800 à 1997
- oil : évolution annuelle du prix du pétrole en dollars

2.1 Tracer les courbes correspondantes. Décrire sommairement chaque série : proposer un modèle lorsque c'est possible (multiplicatif ou additif) puis la forme de la tendance, la présence éventuelle d'une saisonnalité.

2.2 Calculer et représenter graphiquement pour chacune de ces séries la fonction d'autocorrélation empirique et le périodogramme correspondants. Commenter rapidement les résultats obtenus.

3 Estimation et élimination de la tendance et du facteur saisonnier

Pour la série chronologique issue du fichier `airline.rda`, effectuer les opérations suivantes :

3.1 changement de coordonnées pour obtenir une variation saisonnière d'amplitude à peu près constante.

3.2 traitement simultané de la tendance et du facteur saisonnier :
- première estimation de la tendance par moyenne mobile de longueur imposée,
- estimation de la variation saisonnière centrée,
- élimination de cette variation saisonnière,
- nouvelle estimation de la tendance par polynôme,
- analyse de la fonction de corrélation empirique du résidu obtenu.

3.3 Comparer ces résultats à ceux que l'on obtient sans le changement initial de coordonnées.

3.4 Faire également une désaisonnalisation par différenciation. Commenter.

M2IM, Séries Chronologiques
2010-2011

Travaux Pratiques n°4

Modèles d'état et filtrage de Kalman

Ces Travaux Pratiques porteront sur les points suivants :
- Génération de processus (notamment non stationnaires) au moyen de modèles d'état,
- Estimation par filtrage de Kalman et Kalman étendu.

1 Génération de processus au moyen de modèles d'état

1.1 Programmer les représentations par modèle d'état de l'exercice 1.5 page 10 du cours. Visualiser les résultats obtenus en faisant varier les paramètres.

1.2 Soit m_t un processus dont la pente $m_t - m_{t-1}$ est un processus de marche au hasard. Ecrire un modèle d'état dont l'observation serait $y_t = m_t + W_t$, avec W_t un bruit blanc gaussien indépendant de m_t, puis observer des réalisations de y en faisant varier la variance de l'incrément de la marche au hasard. En prenant une variance assez faible, obtenir une tendance localement linéaire.

2 Estimation récursive par filtrage de Kalman

On considère la série égale à la somme d'une tendance linéaire et d'un bruit blanc, où l'état à estimer est X et l'observation est Y (X est observé avec une erreur de mesure qui est un bruit blanc gaussien). On suppose donc inconnus les paramètres de X (valeur courante et pente de la tendance linéaire), et on connaît la variance de l'erreur de mesure.

2.1 Générer une réalisation Y de ce modèle, puis programmer les équations du filtre de Kalman afin d'estimer les paramètres inconnus de X et les valeurs estimées de X. Exécuter ce programme et tracer sur une même courbe :
- les valeurs vraies de la tendance linéaire
- les mesures (valeurs de Y)
- les valeurs estimées de la tendance linéaire

Interpréter ces résultats.

2.2 Exécuter les mêmes opérations sur le modèle de la question 1.2, en supposant qu'on connait la variance des bruits d'état et de mesure.

3 Filtrage non linéaire

Reprendre l'exercice 2 en supposant que la mesure Y est non pas égale à la tendance plus un bruit blanc, mais au carré de la tendance plus un bruit blanc (Filtre de Kalman étendu). Analyser le comportement du filtre en fonction de l'initialisation.

M2IM, Séries Chronologiques
2010-2011

Travaux Pratiques n°5

ARMA

Ces Travaux Pratiques porteront sur les points suivants :
- autocorrélation et autocovariance empiriques pour des $ARMA$ simples,
- estimation préliminaire de l'ordre d'un processus modélisé comme un MA,
- densité spectrale d'un $ARMA$ et estimation par le périodogramme d'une trajectoire,
- algorithme de Durbin Levinson et autocorrélation partielle empirique.

1 Autocorrélation et autocovariance empiriques

1.1 Ecrire un programme qui calcule la fonction de corrélation empirique d'une réalisation d'un processus, puis les intervalles de confiance à 95% de type bruit blanc et de type MA, et qui trace les trois sur un même graphique. Tester votre programme sur les 3 exemples simples suivants :
- $AR(1)$ avec $\phi_1 = 0.9$,
- $AR(1)$ avec $\phi_1 = 0.1$,
- $MA(1)$ avec $\theta_1 = 0.5$,

On prendra par exemple des échantillons de taille 500, et on représentera la fonction de corrélation empirique entre 0 et 40.

1.2 Pour les 3 échantillons, utiliser les graphiques obtenus pour :
- tester s'ils peuvent être considérés comme des bruits blancs ou non,
- évaluer le plus petit ordre M d'un processus MA qui s'ajuste convenablement à cet échantillon.

1.3 Pour les 3 échantillons, comparer la fonction de corrélation théorique du processus simulé et la fonction de corrélation empirique de l'échantillon.

2 Processus $ARMA$, densité spectrale et périodogramme

2.1 Soit X un processus $AR(3)$ causal défini par :

$$\phi(B)X_t = Z_t \ .$$

Faire une représentation graphique de sa densité spectrale de puissance pour plusieurs valeurs des racines du polynôme ϕ. Interpréter.

2.2 Pour une valeur donnée du polynôme ϕ, générer une trajectoire du processus avec `arima.sim()`, calculer son périodogramme et en faire une représentation graphique, à comparer au résultat obtenu au paragraphe précédent.

2.3 Calculer et représenter des périodogrammes lissés de ce même processus, en lissant dans le domaine spectral et dans le domaine temporel.

3 Algorithme de Durbin Levinson

3.1 Ecrire un programme pour l'algorithme de Durbin Levinson (calcul récursif des ϕ_{mk} et de v_m à partir de la covariance empirique).

3.2 Sur plusieurs exemples de réalisations de processus $ARMA$ (on choisira un $AR(1)$, un $AR(3)$, et un $MA(1)$), calculer la fonction d'autocorrélation partielle empirique en utilisant l'algorithme de Durbin Levinson, et la représenter avec le même intervalle de confiance que pour la fonction de corrélation empirique d'un bruit blanc, puis estimer ces processus en tant qu'AR : valeurs des ϕ_i et de σ^2.

M2IM, Séries Chronologiques
2010-2011

Travaux Pratiques n°6

ARMA

Ces Travaux Pratiques s'effectueront avec le logiciel Matlab et porteront sur les points suivants :

- Prédiction d'un $ARMA$ avec un filtre de Kalman,
- Estimation d'un $ARMA$ gaussien par maximum de vraisemblance.

1 Prédiction d'un $ARMA$ de paramètres connus avec un filtre de Kalman

1.1 Ecrire le modèle d'état donné dans le cours pour un $ARMA(p,q)$ avec $\sigma^2 = 1$. Préciser la valeur des initialisations $P_0(\mathbf{X_0})$ et $C_{0|0}$ du filtre de Kalman lorsque les paramètres de l'$ARMA$ sont connus.

1.2 Dans la suite, on suppose que $p = 3$ et $q = 2$. Ecrire un programme qui prend en entrée les paramètres $\phi_1, \ldots, \phi_3, \theta_1, \theta_2$ d'un $ARMA(3,2)$ avec $\sigma^2 = 1$ et une trajectoire de cet $ARMA$ et qui donne en sortie le vecteur des prédictions à un pas, le vecteur des innovations ainsi que les variances des innovations.

1.3 Montrer que ce programme marche en fait pour σ^2 quelconque, mais qu'on obtient en sortie les variances divisées par σ^2.

1.4 Utiliser ce programme sur la série `armaconnu`, du fichier `armaconnu.rda`, qui est une trajectoire d'un $ARMA(3,2)$ avec les paramètres suivants :

$$\phi_1 = 0.1,\ \phi_2 = 0.65,\ \phi_3 = -0.225 \quad \theta_1 = 0.2,\ \theta_2 = -0.48\ ,$$

et σ^2 est inconnu. Tracer sur un même graphe la série et les prédictions, et sur un autre graphe, l'autocorrélation empirique des innovations standardisées. Donner une estimation du paramètre σ^2.

2 Estimation d'un $ARMA$ (gaussien) d'ordre connu par maximum de vraisemblance

2.1 En utilisant le programme de l'exercice précédent, écrire un programme qui prend en entrée une série X, et qui donne en sortie les paramètres $\widehat{\phi}_1, \ldots, \widehat{\phi}_3, \widehat{\theta}_1, \widehat{\theta}_2$ qui minimisent la fonction suivante :

$$l(\phi, \theta) = \log\left(\frac{1}{n}\sum_{i=1}^{n}\frac{(X_i - \widehat{X}_i)^2}{r_{i-1}}\right) + \frac{1}{n}\sum_{i=1}^{n}\log r_{i-1}\ .$$

Votre programme doit aussi donner l'estimateur de σ^2 :

$$\widehat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}\frac{(X_i - \widehat{X}_i)^2}{r_{i-1}}\ .$$

On a noté $\widehat{X}_i$ la prédiction (linéaire) de X_i sachant $X_1, \ldots, X_{i-1}$ et r_{i-1} la variance, renormalisée par σ^2, de l'erreur de prédiction $X_i - \widehat{X}_i$ (ces quantités dépendent bien sûr de (ϕ, θ)). Pour effectuer l'optimisation, on peut utiliser la commande `optim` avec la syntaxe suivante (voir l'aide d'`optim`) :

```
res <- optim(init, fn, method="BFGS")
```

où :

- `fn` est la fonction à minimiser,
- `init` est une valeur d'initialisation pour l'argument $(\phi_1, \phi_2, \phi_3, \theta_1, \theta_2)$ de `fn` (on pourra partir de 0).

Remarque : Si vous utilisez une hypothèse de causalité à un moment (par exemple, dans le calcul de la fonction d'autocovariance), faites attention à rester dans la région de causalité lors de l'optimisation.

2.2 Tester votre programme sur la série `armaconnu`. Analyser les résidus.

M2IM, Séries Chronologiques
2010-2011

Travaux Pratiques n°7

ARMA

Ces Travaux Pratiques porteront sur les points suivants :
- l'identification d'un processus $ARMA$ à l'aide de la fonction `arima`,
- la somme de deux $ARMA$ indépendants,
- l'élimination de composantes spectrales.

Identification d'un processus $ARMA$ avec `arima`

Pour estimer un modèle $ARMA(p, q)$ avec moyenne, p et q étant fixés, sur la série `x` :
`out <- arima(x,order=c(p,0,q))`
Avec cette syntaxe, la méthode utilisée par R est la suivante : le bruit blanc dans la définition de l'$ARMA$ est supposé gaussien, et les coefficients estimés sont ceux qui maximisent la vraisemblance. Si on veut supposer la moyenne nulle, il faut utiliser l'option `include.mean=FALSE`.
Le résultat obtenu, auquel on a donné le nom `out` est une liste (de classe "Arima") qui comporte notamment les éléments suivants :
- `coef` : les coefficients du modèle estimé,
- `var.coef` : la matrice de covariance estimée des coefficients estimés,
- `sigma2` : la variance estimée des innovations,
- `residuals` : le vecteur des résidus,
- `aic` : la valeur de l'AIC correspondant.

On a accès à ces valeurs en tapant, par exemple, `out$coef`.

Rappel : lorsqu'on ne sait pas pour quelles valeurs de p et q utiliser `arima`, on emploie la méthode du cours : examen de l'autocorrélation et de l'autocorrélation partielle (fonctions `acf(,ci.type='ma')` et `pacf`, voir l'aide), détermination du plus petit ordre M tel que le processus puisse être "proche" d'un $MA(M)$ ou d'un $AR(M)$ puis estimation et validation de ce modèle. Ensuite, on cherche à réduire le nombre de paramètres en testant différentes valeurs pour (p, q) avec $p + q \leqslant M + 1$. On peut par exemple le faire de manière automatique en sélectionnant le couple ayant le plus petit AIC, mais ce choix n'est pas toujours le plus judicieux.

Validation d'un ajustement avec `diagts2`

La fonction `diagts2` (à récupérer dans le dossier de documentation) s'applique à un objet de classe "Arima" et donne un diagnostic visuel de la qualité de l'estimation. Elle renvoie un graphique où figurent :
- les résidus standardisés,
- leur fonction d'autocorrélation empirique,
- un "test" graphique de la normalité des résidus,
- des p-valeurs associées à la statistique de Ljung-Box.

1 Premiers pas sur un $AR(1)$

Générer un $AR(1)$ de longueur 1000 avec $\phi_1 = 1/1.05$ (utiliser par exemple la fonction `arima.sim`). Sur cette réalisation, essayer d'ajuster un $ARMA(p,q)$ avec les couples (p,q) suivants. Commenter chaque résultat.

$(p,q) = (0,3)$, $(p,q) = (1,0)$, $(p,q) = (2,1)$ et $(p,q) = (2,0)$.

2 Utilisation automatique d'AIC

2.1 Ecrire un programme qui prend en entrée une série x et un entier M, qui ajuste un $ARMA(p,q)$ pour chaque couple (p,q) tel que $p+q \leqslant M+1$, et qui retourne :

- le modèle ayant la plus petite valeur d'AIC.
- une matrice contenant les valeurs d'AIC pour tout couple (p,q) tel que $p+q \leqslant M+1$.

2.2 Tester ce programme sur `armaa.rda`, `armab.rda` et `armac.rda`

3 Un spectre avec 3 "pôles"

Un ingénieur a besoin d'analyser la série chronologique `poles1.rda`, qui compte 1000 observations. Dans cet exercice, nous allons essayer de lui proposer un modèle s'ajustant fidèlement à cette série.

3.1 Charger la série `poles1.rda` et tracez sa fonction d'autocorrélation ainsi que son périodogramme.

3.2 Premier modèle. Soient X un $AR(2)$ ayant deux pôles complexes conjugués et Y un $AR(1)$ indépendant de X. On définit $U = X + Y$.

(a) Si les pôles de X et Y sont assez proches du cercle unité (mais en dehors), quelle allure a la densité spectrale de puissance de U ?

(b) On sait par ailleurs que U est un $ARMA(3,2)$ (cf. exercice du cours 5.21 page 77). Essayer d'ajuster un $ARMA(3,2)$ sur `poles1.rda`.

3.3 Second modèle Soit Y un $AR(1)$ et U un processus défini comme suit :

$$U_t = Y_t + A\sin(\omega_0.t) \ ,$$

où A est un réel, et $\omega_0 \in [0, 2\pi[$.

(a) Quelle sera l'allure du periodogramme d'une réalisation de U ? Montrer que U admet une représentation en $ARMA(3,2)$ non causal.

(b) On suppose maintenant que `poles1.rda` suit le modèle précédent. Essayer d'ajuster ce modèle. On pourra commencer par estimer ω_0 puis appliquer un filtre éliminant la composante sinusoïdale et enfin estimer un AR(1) sur la série V_t débarrassée de la composante sinusoïdale.

(c) Montrer que le signal filtré V_t suit en fait un ARMA(3,2). Ajuster ce troisième modèle.

3.4 Sélection du meilleur modèle. En fait, le vrai but de l'ingénieur est de prédire à 5 pas dans le futur (un pas = une période d'échantillonnage). Pour choisir le meilleur modèle prédictif à 5 pas, vous disposez d'un échantillon de 500 valeurs, `poles2.rda` qui ont été enregistrées dans la foulée du premier échantillon. Quel modèle choisissez-vous ? Mettez en compétition les trois modèles précédents et le modèle 'trivial' qui consiste à prédire avec la valeur courante. Commenter.

Pour prédire, vous pourrez utiliser la fonction `kalmanpred.R` qui effectue la prédiction par filtre de Kalman, prend en entrée la série `x`, les paramètres de l'$ARMA$ `phi` et `theta` et l'horizon de prédiction `n.ahead` et renvoie en sortie le vecteur y des prédictions à l'horizon `n.ahead` : $y_t = P(x_{t+n.ahead}|x_1, \ldots, x_t)$.

M2IM, Séries Chronologiques
2010-2011

Travaux Pratiques n°8

Processus $(S)ARIMA$ et $ARMA/GARCH$

Ces Travaux Pratiques porteront sur les points suivants :
- génération d'un processus $SARIMA$.
- l'identification d'un processus $ARIMA$ à l'aide de la fonction `arima`,
- l'identification d'un processus $SARIMA$ à l'aide de la fonction `arima`.

1 Génération d'un processus $SARIMA$

1.1 Ecrire un programme général de génération d'un processus SARIMA avec s quelconque, et $p = d = q = P = D = Q = 1$.

1.2 Utiliser ce programme avec diverses valeurs des paramètres pour vérifier que l'on peut effectivement représenter les variations énumérées dans le cours ; par exemple, pour une longueur $n = 100$,

$$s = 24, \quad \phi = -0.7, \quad \Phi = 0.95, \quad \theta = 0, \quad \Theta = 0\ .$$

Représenter comme dans le cours la série non intégrée et les séries intégrées (une fois à la période de l'échantillon, une fois à la période de la saison, puis les deux).

2 Identification d'un processus $ARIMA$

Le fichier `data_arima.rda` contient une réalisation d'un processus $ARIMA$.

2.1 Représenter graphiquement ses fonctions d'autocorrélation et d'autocorrélation partielle empirique. Que peut-on dire à partir de l'analyse visuelle de ces fonctions ?

2.2 Identifier ce processus $ARIMA$ en suivant la méthode donnée dans le cours pour estimer l'ordre (p, d, q) : on identifiera un $ARMA(p, q)$ (avec moyenne) sur la série différenciée d fois.

2.3 Renouveler l'identification en utilisant la fonction `arima` directement sur la série originale.

Attention, par défaut, lorsqu'on utilise `arima` *avec d au moins égal à 1, R prend comme modèle "$\nabla^d X$ est un $ARMA(p, q)$ de moyenne nulle". Ceci n'est pas toujours raisonnable (est-ce raisonnable ici ?). Pour contourner ce problème, il conviendra d'ajouter des régresseurs à l'appel d'*`arima` *en ajoutant* `xreg=(1:n)^d` *si la série est de longueur* `n`.

3 Modélisation à l'aide d'un processus $SARIMA$

On veut modéliser la série `air.rda` à l'aide d'un $SARIMA$.

3.1 Quelle est la période de la saisonnalité de cette série ?

3.2 Modéliser cette série avec un $SARIMA$ en utilisant la méthode indiquée dans le cours et la fonction `arima` directement sur la série originale.

3.3 Quel est le modèle final pour `air.rda` ?

3.4 A l'aide de la fonction `predict`, prédire les valeurs de `air.rda` sur 12 mois (année 1961). A l'aide de `ts.plot`, représenter sur un même graphique les valeurs de la série initiale et les valeurs prédites assorties des intervalles de confiance à 95%.

M2IM, Séries Chronologiques
2010-2011

Travaux Pratiques n°9

Processus à mémoire longue et $GARCH$

Ces Travaux Pratiques s'effectueront avec R et porteront sur les points suivants :
- simulation et identification d'un processus $ARFIMA$,
- simulation et identification d'un processus $ARMA/GARCH$.

1 Simulation d'un $ARFIMA$

1.1 Ecrire un programme qui simule un $ARFIMA(p, d, q)$ par la méthode spectrale.

1.2 Visualiser une trajectoire pour :
- $d = 0.4$, $(p, q) = (0, 0)$,
- $d = 0.4$, $(p, q) = (1, 0)$ et $\phi_1 = 0.9$,
- $d = -0.3$, $(p, q) = (1, 0)$ et $\phi_1 = 0.9$,
- $d = -0.3$, $(p, q) = (0, 1)$ et $\theta_1 = 0.5$.

Pour chacune de ces trajectoires, observer le périodogramme ainsi que les acf et pacf.

2 Identification d'un $ARFIMA(1, d, 1)$

2.1 Ecrire un programme qui ajuste par maximum de vraisemblance (en utilisant la fonction `optim`) un modèle $ARFIMA(1, d, 1)$.

2.2 Tester votre programme sur les données `arfima1d1.rda`.

3 Simulation d'un $GARCH$

3.1 Ecrire un programme qui simule un $ARMA(p, q)/GARCH(1, 1)$.

3.2 Visualiser une trajectoire pour $\alpha_0 = 1$ et :
- $\alpha_1 = 0.3$, $\beta_1 = 0.65$, $(p, q) = (1, 0)$, $\phi_1 = 0.9$,
- $\alpha_1 = 0.9$, $\beta_1 = 0.05$, $(p, q) = (1, 0)$, $\phi_1 = 0.9$.

4 Identification d'un $GARCH$

Les données `pepper.rda` représentent le prix moyen du poivre blanc sur le marché européen.

4.1 Représenter cette série graphiquement

4.2 Appliquer une transformation logarithmique et les transformations nécessaires afin de rendre la série stationnaire. Soit Y la série ainsi transformée.

4.3 Ajuster un modèle $ARMA$ à Y. Analyser les résidus.

4.4 Analyser les carrés des résidus.

4.5 Ajuster un modèle $GARCH$ pour les résidus (par maximum de vraisemblance) et écrire le modèle complet pour Y.

M2IM, Séries Chronologiques
2010-2011

Aide-mémoire pour le logiciel R

1 Aide

- `help()` affiche l'aide sur une fonction ou un package
- `help.search()` recherche dans l'aide les sujets liés à la chaîne de caractères passée en argument

2 Les packages

- `library()` lister les packages installés
- `install.packages()` pour installer un package
- `library()` charger un package qui n'est pas chargé au démarrage de R

3 Les données

3.1 Importer des données

- `scan()` transforme les données d'un fichier en un vecteur ou une liste
- `read.table()` transforme les données d'un fichier au format tableau en un objet de type `data.frame` (tableau de données)
- `save()` enregistre des données au format R
- `load()` charge des données enregistrées avec `save`
- `ls()` donne la liste des objets R chargés
- `data()` permet de charger les paquets de données ou bien des données présentes dans le sous-répertoire `data` du répertoire courant. Un fichier `.rda` est chargé avec `load`, un fichier `.R` avec `source`, les fichiers `.txt` ou `.csv` avec `read.table`.
- `attach()` rend accessible les colonnes d'une matrice ou les éléments d'une liste par leur nom, sans passer par le nom de la matrice ou de la liste

3.2 Données Matlab

Il existe un package, `R.matlab`, qui permet de lire et écrire les données au format Matlab (et également de communiquer avec Matlab). On y trouve notamment les commandes suivantes :

- `readMat()` lire un fichier Matlab
- `writeMat()` écrire des données au format Matlab

4 Les graphiques

- `x11()` ouvre une fenêtre graphique
- `plot()` trace un graphe. Le paramètre `type` permet de spécifier si on joint les points par des lignes (`type="l"`), si on marque seulement les points (`type="p"`), si on fait les deux (`type="b"`), si on trace une fonction en escalier (`type="s"`) ou un diagramme en bâtons (`type="h"`) etc. Le type de points peut être spécifié par un numéro avec le paramètre `pty` et le type de lignes avec `lty`.
- `lines()` trace un graphe de type ligne sur le graphique actif (permet de superposer des graphes)

- `points()` trace un graphe de type points sur le graphique actif (permet de superposer des graphes)
- `matplot()` trace simultanément plusieurs graphes regroupés dans une matrice
- `ts.plot()` pour tracer les séries temporelles
- `par()` permet de contrôler les paramètres graphiques. Notamment `mfrow` pour partitionner le graphique, et `oma` pour rajouter des marges extérieures (permet de rajouter un titre global à un graphique partitionné, via `mtext(...,line=,side=,outer=TRUE,cex=)`)
- `axis()` ajoute un axe au graphe existant
- `colors()` liste les noms de couleurs reconnus par `R`
- `palette()` permet de contrôler la palette utilisée par `R` lorsqu'on utilise `col=num` dans `plot`
- `layout()` partitionne un graphique en plusieurs fenêtres (une alternative à `par(mfrow=...)`)
- `graphics.off()` ferme toutes les fenêtres graphiques
- `dev.cur()` retourne le numéro et le nom du périphérique graphique actif (typiquement une fenêtre graphique ou un fichier postscript)
- `dev.print()` imprime le graphique du périphérique graphique actif sur un nouveau périphérique (par défaut un fichier postscript)
- `dev.off()` ferme un périphérique graphique
- `dev.set()` rend actif le périphérique spécifié par son numéro
- `dev.list()` liste tous les périphériques graphiques ouverts
- `pdf(file=...)` ouvre un fichier pdf pour y afficher les graphiques (penser à fermer le fichier avec `dev.off()`)

Il existe un package `iplots` qui permet d'avoir des graphiques interactifs (on peut par exemple zoomer sur une fenêtre, changer le titre etc.). Remarque : `iplots` charge le package `rJava`. Fonction principale : `iplot()`

5 Matrices, vecteurs, variables etc.

- `matrix()` créer une matrice
- `numeric()` créer un vecteur nul de taille donnée (de type double)
- `c()` concaténer des objets
- `cbind()` juxtaposer en colonnes des vecteurs ou des matrices
- `rbind()` juxtaposer en ligne des vecteurs ou des matrices
- `dim()` connaître la dimension d'une matrice
- `list()` créer une liste
- `seq()` créer des suites arithmétiques
- `rep()` créer des vecteurs constants
- `mode()` connaître le mode
- `typeof()` connaître le type d'une variable
- `get()` appliqué à une chaîne de caractère, renvoit la valeur de la variable ayant pour nom cette chaîne. L'inverse se fait avec `assign()`
- `NA` « Not Available » : désigne une valeur manquante
- `Inf` désigne $+\infty$
- `NULL` objet nul

6 Chaînes de caractères

- `as.character()` convertir en chaîne de caractères

- `paste()` accoler des chaînes de caractères, le séparateur est précisé par `sep=` (une option intéressante : `collapse`)

7 Génération aléatoire

- `rnorm()` générer un vecteur de gaussiennes indépendantes
- `runif()` générer un vecteur de variables aléatoires uniformes indépendantes
- `set.seed()` assigner une valeur à la graine du générateur aléatoire

8 Interface graphique

voir le package `JGR` (charge `iplots`)

9 Les séries temporelles

- `ts()` met un vecteur au format série temporelle. Options importantes :
  - `start` date de la première valeur
  - `frequency` fréquence des observations

  Par exemple, si on dispose d'observations trimestrielles et que la première observation est le second trimestre de l'année 1959, alors on précisera : `frequency=4, start=c(1959,2)`
- `lag.plot()` trace les graphes de x_t en fonction de x_{t-h} pour les valeurs de h allant de 1 à `lags`
- `filter()` permet de filtrer une série par une moyenne mobile, mais aussi de filtrer de manière autorécursive
- `ts.plot()` pour tracer les séries temporelles

10 Divers

- `options()` retourne la liste des valeurs courantes des options globales, et permet de changer leurs valeurs
- `getOption()` obtenir la valeur des options globales
- `getwd()` donne le répertoire courant
- `setwd()` permet de changer de répertoire courant
- `list.files()` donne la liste des fichiers du répertoire courant
- `getAnywhere()` recherche une fonction R. Mentionnons qu'un certain nombre de fonctions R font appel à des routines C qui ne sont visibles que si l'on dispose des fichiers source de R. Si on est intéressé par ce que fait une de ces routines, il faut alors retrouver le fichier C associé dans les sources.
- `source()` permet de charger un programme ou une fonction
- `q()` fait quitter `R`