

UNIVERSIDADE DE ÉVORA

Escola de Ciências e Tecnologia

Departamento de Matemática

MESTRADO EM MODELAÇÃO ESTATÍSTICA E ANÁLISE DE DADOS

MODELAÇÃO DE SÉRIES TEMPORAIS EM R

SÉRIES COM SUPORTE EM NÚMEROS REAIS E NÚMEROS INTEIROS

Dissertação de Mestrado sob

Orientação: Professora doutora
Dulce Gomes

Co-orientação: Professora doutora
Maria Filomena Mendes

José Luís Charrua dos Santos

Évora, 2011

Agradecimentos

À Professora doutora Dulce Gomes, a orientadora de todas as horas, as boas e as menos boas!

À Professora doutora Maria Filomena Mendes que encontrou dados para eu me dedicar a analisar...

Ao Professor doutor Paulo Infante, que realizou a tarefa impossível, de eu voltar a estudar Estatística... com recurso a software...

À Família, que perdeu mais do que eu jamais alcançarei com este trabalho...

É muito fácil sumariar as conclusões. É difícil relatar todos os erros, pistas falsas, indicações ignoradas, dedicação, trabalho árduo, abandono doloroso de ideias anteriores, que fazem parte da descoberta inicial de qualquer coisa interessante.

Índice

Índice	7
Índice de figuras	11
Resumo	15
Abstract	17
CAP. 1 - Introdução	
Introdução.....	19
CAP. 2 - Análise Box - Jenkins de Séries temporais	
2.1 O que são sucessões cronológicas?	21
2.2 Exemplos de séries temporais.....	22
2.3 R - O Software adoptado	24
2.3.1 Notação na utilização de comandos e apresentação de resultados	24
2.3.2 Considerações gerais elementares sobre o R.....	25
2.4 Processos Estocásticos	27
2.4.1 Processos Estocásticos Estacionários	28
2.4.2 Funções de Autocovariância e Autocorrelação.....	30
2.4.3 Função de Autocorrelação Parcial.....	31
2.4.4 Ruído Branco	31
2.4.5 Estimação dos parâmetros caracterizadores dos processos estacionários	32
2.4.6 Processos Estocásticos Estacionários e Lineares.....	33
2.4.6.1 Processos Autoregressivos de 1ª Ordem	33
2.4.6.2 Processos Autoregressivos de 2ª Ordem	37
2.4.6.3 Processos Autoregressivos de Ordem p.....	39
2.4.6.4 Processos Médias Móveis de Ordem q	40
2.4.6.5 Processos Mistos Autoregressivos e Médias Móveis	45
2.5 Processos Estocásticos Não Estacionários e Lineares	48
2.5.1 Não estacionariedade em média e sazonalidade	49
2.6 Modelação ARIMA de Sucessões Cronológicas.....	54
2.6.1 Identificação do Modelo.....	55
2.6.1.1 Passos na identificação do modelo	55

2.7	Estimação	57
[1]	Estimação dos parâmetros nos modelos autoregressivos.....	57
[2]	Estimação dos parâmetros nos modelos médias móveis	59
[3]	Estimação dos parâmetros nos modelos mistos.....	60
[4]	Estimação não Linear	61
[5]	Propriedades assintóticas dos estimadores da máxima verosimilhança	62
2.8	– Avaliação do Diagnóstico	66
2.8.1	Avaliação da qualidade estatística do(s) modelo(s) ajustado(s)	66
[1]	Significância estatística dos parâmetros.....	66
[2]	Estacionaridade e invertibilidade	68
[3]	Redundância.....	69
[4]	Correlação	69
2.8.2	Análise dos resíduos – Aplicação de testes estatísticos.	70
2.8.3	Testes estatísticos sobre a FAC e FACP.....	72
[1]	Teste de <i>Bartlett</i>	72
[2]	Teste de <i>Jenkins e Daniels</i>	72
[3]	Teste de <i>Box e Pierce</i>	73
[4]	Teste de <i>Ljung-Box</i>	74
[5]	Teste de <i>Kendal e Stuart</i>	74
2.8.4	CrITÉrios de Selecção de Modelos.....	74
[1]	CrITÉrio <i>Akaike</i>	75
[2]	CrITÉrio BIC	76
[3]	CrITÉrio SBC de <i>Schwartz's</i>	76
2.9	Previsão.	76
2.9.1	Previsão de Processos Não Estacionários.	77
2.9.2	Actualização das Previsões.	78
2.9.3	Intervalos de Confiança para Previsões.....	79
2.9.4	Previsão de Sucessões Logaritmizadas.	80
2.10	Análise de dados Reais	82

CAP. 3 - Processos Autoregressivos de valores inteiros não negativos de ordem 1	
3.1 A operação Thinning.....	93
3.2 Propriedades da Operação <i>Thinning</i>	94
3.3 Os Modelos autoregressivos inteiros de ordem 1.....	95
3.4 Momentos até à segunda ordem do modelo INAR(1).....	96
3.5 Identificação da ordem.	98
3.5.1 Critérios de selecção de modelos.....	98
3.5.2 Estimação de parâmetros do modelo	99
3.5.3 Validação do modelo.	100
3.5.4 Previsão.....	100
3.6 Simulação de um processo INAR(1) em R.	101
3.7 Análise de uma série de inteiros não negativos.....	108
3.7.1 Óbitos por NUT2.....	111
CAP. 4 - Conclusão	121
Anexos	
Anexo I	
Exemplos clássicos: Ruído Branco e Passeio Aleatório.....	125
Anexo II	
Comandos para simular e estimar um processo	129
Anexo III	
Análise completa de AR(1) em R	137
Anexo IV	
Geração e análise de um processo AR(1)em R com média e variância não nulas pelo método de Yule-Walker	165
Anexo V	
Resumo de comandos mais importantes	177
Bibliografia	183

Figura 1: Cronogramas das cotações de fecho de empresas tecnológicas.....	22
Figura 2: Cronograma da taxa de mortalidade, em Portugal, de crianças com menos de um ano de idade, por ano do óbito	23
Figura 3: Conteúdo da livraria tseries.....	26
Figura 4: Resultado do <i>help</i> do R sobre o comando <i>lillie.test</i>	26
Figura 5: Cronograma de um processo AR(1) simulado, $\phi = 0.7$	35
Figura 6: FAC e FACP do processo AR(1) simulado, $\phi = 0.7$	36
Figura 7: Cronograma de um processo AR(1) simulado, $\phi = -0.7$	36
Figura 8: FAC e FACP do processo AR(1) simulado, $\phi = -0.7$	36
Figura 9: Cronograma de um processo AR(2) simulado, $\phi_1 = -0.5, \phi_2 = -0.3$	38
Figura 10: FAC e FACP do processo AR(2) simulado, $\phi_1 = -0.5, \phi_2 = -0.3$	39
Figura 11: Cronograma do processo MA(1) simulado, $\theta = 0.7$	42
Figura 12: FAC e FACP do processo MA(1) simulado, $\theta = 0.7$	42
Figura 13: FAC e FACP do processo MA(1) simulado, $\theta = -0.7$	43
Figura 14: FAC e FACP do processo MA(1) simulado, $\theta = -0.7$	43
Figura 15: Cronograma do processo MA(2) simulado, $\theta_1 = -0.7, \theta_2 = -0.5$	44
Figura 16: FAC e FACP do processo MA(2) simulado, $\theta_1 = -0.7, \theta_2 = -0.5$	44
Figura 17: Cronograma do processo ARMA(1,1) simulado, $\phi = 0.7, \theta = 0.5$	47
Figura 18: FAC e FACP do processo ARMA(1,1) simulado, $\phi = 0.7, \theta = 0.5$	47
Figura 19: Sucessão com tendência linear	49
Figura 20: Sucessão com componente sazonal.....	49
Figura 21: Sucessão com tendência linear e sazonal.....	49
Figura 22 : Cronograma de <i>Electricity</i>	53
Figura 23 : Transformação de Box Cox com intervalos de confiança para λ	53
Figura 24 : Cronograma do log de <i>electricity</i>	54
Figura 25 : Cronograma da diferença de log de <i>electricity</i>	54
Figura 26: Cronograma dos resíduos.....	71
Figura 27: FAC dos resíduos.....	71
Figura 28: FACP dos resíduos	71
Figura 29: Cronograma da série gerada com a série ajustada representada por pontos.....	80
Figura 30: Série gerada com previsão e respectivos intervalos de confiança	81
Figura 31: Taxa de mortalidade feminina entre os 50 e 54 anos	82
Figura 32: gráfico de <i>output</i> da função <i>Box.cox.ar()</i>	83
Figura 33: Cronograma da série transformada pelo logaritmo.....	84
Figura 34: FAC da série transformada pelo logaritmo.....	84
Figura 35: FACP da série transformada pelo logaritmo	84
Figura 36: Diferenciação do logaritmo de taxa de mortalidade.....	85
Figura 37: FAC da diferença da série transformada pelo logaritmo.....	85
Figura 38: FACP da diferença da série transformada pelo logaritmo	85
Figura 39: FAC dos resíduos do modelo estimado	88
Figura 40: FACP dos resíduos do modelo estimado.....	88
Figura 41: FAC e FACP dos resíduos dos dois modelos estimados	89
Figura 42: Previsão e respectivos limites de confiança	92

Figura 43: Histograma de inovações com distribuição de Poisson de parâmetro $\lambda = 1$	102
Figura 44: Cronograma de inovações com distribuição de Poisson de parâmetro $\lambda = 1$	102
Figura 45: Cronograma de um processo INAR(1) simulado com $\alpha = 0.4$ $\lambda = 1$	104
Figura 46: Histograma de um processo INAR(1) simulado com $\alpha = 0.4$ $\lambda = 1$	104
Figura 47: FAC de um processo INAR(1) simulado com $\alpha = 0.4$, $\lambda = 1$	105
Figura 48: FACP de um processo INAR(1) simulado com $\alpha = 0.4$, $\lambda = 1$	105
Figura 49: Ajustamento um processo INAR(1) com $\tilde{\alpha} = 0.401$, $\tilde{\lambda} = 0.942$ estimados e simulado com $\alpha = 0.4$, $\lambda = 1$	106
Figura 50: Cronograma dos resíduos do processo resultantes da estimação de $\tilde{\alpha} = 0.401$, $\tilde{\lambda} = 0.942$	106
Figura 51: Histograma dos resíduos do processo resultantes da estimação de $\tilde{\alpha} = 0.401$, $\tilde{\lambda} = 0.942$	106
Figura 52: FAC dos resíduos do processo resultantes da estimação de $\tilde{\alpha} = 0.401$, $\tilde{\lambda} = 0.942$	107
Figura 53: FACP dos resíduos do processo resultantes da estimação de $\tilde{\alpha} = 0.401$, $\tilde{\lambda} = 0.942$	107
Figura 54: Previsão do processo com estimação de $\tilde{\alpha} = 0.401$, $\tilde{\lambda} = 0.942$	108
Figura 55: Cronograma de óbitos – Região Norte.....	113
Figura 56: Histograma de óbitos – Região Norte	113
Figura 57: FAC de óbitos – Região Norte	113
Figura 58: FACP de óbitos – Região Norte	113
Figura 59: Cronograma de óbitos – Região Centro.....	113
Figura 60: Histograma de óbitos – Região Centro	113
Figura 61: FAC de óbitos – Região Centro	113
Figura 62: FACP de óbitos – Região Centro.....	113
Figura 63: Cronograma de óbitos – Região Lisboa e Vale do Tejo.....	114
Figura 64: Histograma de óbitos – Região Lisboa e Vale do Tejo	114
Figura 65: FAC de óbitos – Região Lisboa e Vale do Tejo	114
Figura 66: FACP de óbitos – Região Lisboa e Vale do Tejo.....	114
Figura 67: Cronograma de óbitos – Região do Alentejo	114
Figura 68: Histograma de óbitos – Região do Alentejo.....	114
Figura 69: FAC de óbitos – Região do Alentejo	114
Figura 70: FACP de óbitos – Região do Alentejo.....	114
Figura 71: Cronograma de óbitos – Região do Algarve	115
Figura 72: Histograma de óbitos – Região do Algarve.....	115
Figura 73: FAC de óbitos – Região do Algarve.....	115
Figura 74: FACP de óbitos – Região do Algarve.....	115
Figura 75: Cronograma da diferença de ordem 1 de óbitos – Região Norte	116
Figura 76: Histograma da diferença de ordem 1 de óbitos – Região Norte	116
Figura 77: FAC da diferença de ordem 1 de óbitos – Região Norte	116
Figura 78: FACP da diferença de ordem 1 de óbitos – Região Norte.....	116
Figura 79: Cronograma da diferença de ordem 1 de óbitos – Região Centro	117
Figura 80: Histograma da diferença de ordem 1 de óbitos – Região Centro	117
Figura 81: FAC da diferença de ordem 1 de óbitos – Região Centro	117
Figura 82: FACP da diferença de ordem 1 de óbitos – Região Centro.....	117
Figura 83: Cronograma da diferença de ordem 1 de óbitos – Região de Lisboa e Vale do Tejo	117
Figura 84: Histograma da diferença de ordem 1 de óbitos – Região de Lisboa e Vale do Tejo	117
Figura 85: FAC da diferença de ordem 1 de óbitos – Região de Lisboa e Vale do Tejo	118

Figura 86: FACP da diferença de ordem 1 de órbitos – Região de Lisboa e Vale do Tejo	118
Figura 87: Cronograma da diferença de ordem 1 de órbitos – Região do Alentejo.....	118
Figura 88: da diferença de ordem 1 de órbitos – Região do Alentejo.....	118
Figura 89: FAC da diferença de ordem 1 de órbitos – Região do Alentejo	118
Figura 90: FACP da diferença de ordem 1 de órbitos – Região do Alentejo	118
Figura 91: Cronograma da diferença de ordem 1 de órbitos – Região do Algarve	119
Figura 92: Histograma da diferença de ordem 1 de órbitos – Região do Algarve	119
Figura 93: FAC da diferença de ordem 1 de órbitos – Região do Algarve	119
Figura 94: FACP da diferença de ordem 1 de órbitos – Região do Algarve.....	119

Resumo

MODELAÇÃO DE SÉRIES TEMPORAIS EM R

SÉRIES COM SUPORTE EM NÚMEROS REAIS E NÚMEROS INTEIROS

Séries temporais de contagem são exemplos de séries temporais de valor discreto que aparecem frequentemente na prática.

Vários modelos para processos estacionários têm sido propostos. Um desses modelos, usado em particular para séries de contagem são os processos autorregressivos inteiros de ordem 1, INAR(1).

Recolheram-se dados com o número de óbitos por Município em Portugal. A existência de *missing values*, levou à tentativa de modelação do tipo INAR(1) com a agregação dos dados por NUT2.

Introduzem-se, de forma não exaustiva, os processos do tipo AR, MA, ARMA e ARIMA

O capítulo 3 caracteriza processos do tipo INAR(1).

O R foi a ferramenta utilizada para simular, analisar e modelar, através de exemplos, os diferentes tipos de processos estudados.

O corpo do texto está escrito da mesma forma que aparece usualmente na diferente bibliografia consultada.

O “*core development*” da tese é reflectido nos anexos, onde são utilizados exemplos completos de simulação, análise e modelação, como não encontrei na bibliografia consultada. Podem ser consultados independentemente da leitura do corpo do texto, e constituem um manual de aprendizagem de R aplicado a séries temporais.

Abstract

Modeling Time Series with R

Series supported in real numbers and in integer numbers.

Discret-valued time series are examples of time series that are common in practice.

Several stationary models have been developed and proposed. One of those models, used particularly in counting series is first order integer autoregressive processes.

The numbers of death were collected per Municipality, in Portugal. The existence of missing values, led to the attempt of modelling using INAR(1) processes, with data aggregated by NUT2.

Processes of the type AR, MA, ARMA and ARIMA are introduced in a non-exhaustive way.

Chapter 3 characterizes INAR(1) processes.

R was the tool used to simulate, analyze and model, through examples, the different classes of processes studied.

The main part of the text is written in a way that is similar to the bibliography consulted.

The annexes reflect the core development of the thesis, where are used complete examples of simulation, analysis and modelation, in a way that was not found on the bibliography used. The annexes could be consulted independently, without reading the main text, and are a learning manual of R applied to times series.

Capítulo 1

Introdução

O interesse pelo estudo de séries temporais já vem de longe, de ainda mais longe vem a motivação para trabalhar com dados reais.

As referências bibliográficas sobre o tema são extensas, cada vez há mais, e de leitura mais fácil. Ainda não há muitos anos a leitura obrigatória sobre quem se decidia por estudar este tema era de *Brockwell e Davis* [1991], a bíblia, a minha primeira leitura era de 1986, e incluía uma disquete com um programa denominado “*PEST*”. É um texto de matemáticos para matemáticos, a apreensão desta obra é muito morosa, muito difícil, a abordagem teórica é muito rigorosa, talvez por esse motivo, *Brockwell e Davis* [2002] tenham lançado nova obra, com uma abordagem mais simples, e com inclusão de novo package, “*ITSM2000*”.

Embora tenha consultado as obras de *Brockwell e Davis*, as minhas preferências sempre foram para outro tipo de abordagem e as minhas principais fontes de estudo foram *Wei*, [1994], e de *Muller et al*, [1993], a bíblia portuguesa. São nestas duas obras que reside a minha formação de base e é na sua estrutura e abordagem que está assente a realização deste trabalho. Para aprofundamento do que aqui é exposto, ao nível teórico, a sua consulta é indispensável.

Na aplicação do **R** às séries temporais as principais obras utilizadas foram de *Cowpertwait e Metcalf* [2009], *Cryer e Chan* [2008] e de *Shumway e Stoffer* [2006].

Para especificidades de **R** a referência foi *Crawley* [2007], e inúmeros sites, blogues e fóruns de internet.

Durante a frequência da parte lectiva do mestrado o **R** não foi utilizado nas disciplinas que frequentei, o código apresentado é resultado de um trabalho “solo”.

O objectivo final desta tese é o desenvolvimento de *scripts* de **R** para se estudarem séries temporais de contagem, dado que até à data, não se conhecem trabalhos

nesse sentido. Numa primeira fase utiliza-se o **R** para a análise e estudo, em geral, de séries temporais, com aplicação aos modelos ARMA introduzidos por *Box & Jenkins* [1976].

O **R** tem incluídas algumas rotinas que realizam parte das operações necessárias, outras são realizadas recorrendo a livrarias, que se podem instalar de forma isolada, que permitem efectuar operações específicas não contempladas no programa base.

O **R** é uma linguagem escrita especificamente para utilização estatística, os comandos para as diferentes operações possíveis são *UNIX/LINUX like*, como tal, utiliza-se a linha de comando com recurso a editores e a ficheiros de *script*, as opções de cada comando e a sua sintaxe implicam uma demorada e lenta curva de aprendizagem. O capítulo 2, também, constitui em parte, uma introdução ao **R**.

No capítulo 2 introduzem-se os processos ARMA e os comandos de **R** para o seu estudo. Em anexo estão exemplos mais pormenorizados, onde se utiliza uma programação de scripts mais elaborada.

No final do capítulo 2 aplica-se a modelação à taxa de mortalidade de mulheres em Portugal, no intervalo etário de 50 a 54 anos, por ano de ocorrência do óbito.

O capítulo 3 aborda a teoria dos processos de contagem de ordem 1 e tenta-se modelar o número de óbitos, por NUT2, de Portugal Continental, de 1980 a 2007.

No capítulo 4 conclui-se de uma forma peculiar e deixam-se em aberto os horizontes.

Os anexos podem ser lidos de forma individual e sem ser necessária a leitura integral do trabalho, têm como propósito mostrar funcionalidades do **R**, em que não seja necessário apreender os conceitos introduzidos no capítulo 2.

Capítulo 2

Análise Box - Jenkins de Séries temporais

2.1 O que são sucessões cronológicas?

O termo série temporal, tem origem no inglês *time series*. Por definição, uma série temporal é uma sucessão de observações que evoluem no tempo, igualmente espaçadas, onde a ordem de recolha desempenha um papel primordial. Ou seja, são um conjunto de observações feitas em pontos ou períodos sucessivos de tempo, num dado intervalo fixo.

O nosso objecto de estudo é assim referido por vários autores como:

- Sucessões cronológicas
- Sucessões temporais
- Séries temporais
- *Time series* (única designação utilizada na língua inglesa)

Nos mais variados sectores há variáveis que são medidas de forma sequencial ao longo do tempo. Cotações de empresas, reservas de petróleo, número de óbitos, número de nascimentos, taxas de mortalidade, taxas de natalidade, valores médios diários de temperatura observada, etc.

Quando uma variável é medida sequencialmente ao longo do tempo, num intervalo fixo, conhecido como intervalo amostral, os dados resultantes dessas medições formam uma sucessão cronológica ou série temporal. Uma sequência de variáveis aleatórias definidas em intervalos amostrais de tempo fixo são designadas por processos estocásticos em tempo discreto, mas na maioria dos casos são designadas simplesmente por séries temporais. A teoria sobre processos estocásticos é vasta, pode ser estudada sem que seja necessária a sua aplicação a dados reais, ou realizações específicas de uma determinada variável aleatória, que inclua a modelação e ajustamento de modelos. Contudo o objectivo deste trabalho foca-se no ajustamento de modelos e análise de dados.

Além do modelo matemático que melhor poderá descrever o comportamento da variável ou variáveis do fenómeno em estudo há que averiguar se tal modelo ou modelos são válidos para se preverem ocorrências futuras.

Box e Jenkins [1976] sugeriram que a metodologia de análise de séries temporais fosse realizada por etapas:

- Etapa 1 → **Identificação** – onde se realiza a identificação do modelo.
- Etapa 2 → **Estimação** – onde se realiza a estimação dos parâmetros do modelo seleccionado.
- Etapa 3 → **Avaliação do diagnóstico** – onde se realiza a avaliação da qualidade e adequação do modelo seleccionado. Se o modelo for satisfatório avança-se para o objectivo último, a previsão baseada no modelo escolhido; se o modelo não for satisfatório recomeça-se o processo de novo na tentativa de identificar um modelo que melhor se adequa.

No final das etapas propostas realiza-se a previsão de novas ocorrências baseadas no modelo, identificado na etapa 1, com os parâmetros estimados na etapa 2, que melhores resultados produziram na etapa 3. O processo é iterativo.

2.2 Exemplos de séries temporais.

Cada uma das cotações diárias, de fecho, das diferentes empresas é um exemplo de série temporal. A pertinência da escolha destes títulos, deve-se ao facto de incluírem no período representado a crise do *sub prime*.

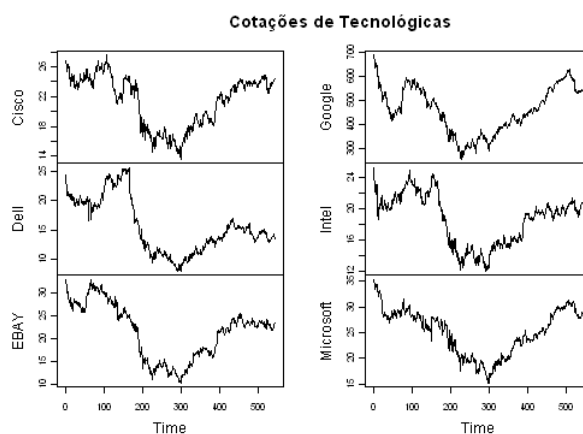


Figura 1: Cronogramas das cotações diárias de fecho de empresas tecnológicas

A tendência decrescente da cotação de todos os títulos, a partir de certa altura é notória, mas não se consegue identificar o momento de início em que tal acontece, nem o momento em que começam a recuperar, para os investidores esta é “*the million dollar question*”. O início do decréscimo dos valores das acções é de Agosto a Outubro de 2008, altura que os mercados têm consciência da crise do *sub prime*, a recuperação é iniciada em Março de 2009. Os pormenores destes acontecimentos foram noticiados nas televisões e jornais, aqui só se pretende mostrar o comportamento geral das cotações de fecho e de como são bons exemplos de séries temporais.

As taxas, independentemente da sua origem, também são bons exemplos para se aplicar a teoria de séries temporais. Prossegue-se com a taxa de mortalidade, por ano do óbito, de crianças, com menos de um ano de idade, em Portugal, no período de 1940 a 2006.

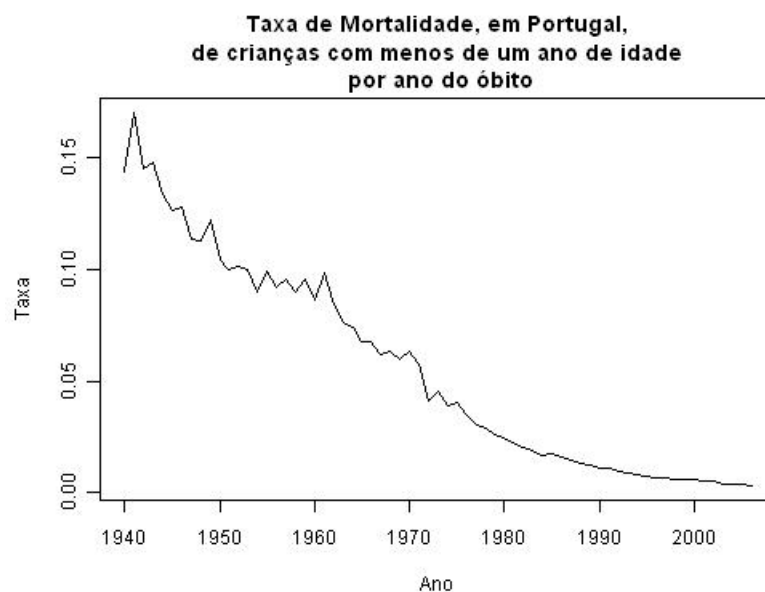


Figura 2: Cronograma da taxa de mortalidade, em Portugal, de crianças com menos de um ano de idade, por ano do óbito

Esta série temporal denota um grande decréscimo entre 1940 e 2006, pois o desenvolvimento da medicina, e a melhoria generalizada das condições sociais em Portugal, são causas empíricas que explicam que a taxa de mortalidade,

entre crianças, com menos de um ano de idade tenha vindo a diminuir ao longo do tempo.

Em termos, da análise de séries temporais, observa-se facilmente que esta série não é estacionária. Terminologia que se introduzirá mais à frente.

2.3 R - O Software adoptado

O **R** foi iniciado por *Ross Ihaka*, e *Robert Gentleman*, (1996) do Departamento de Estatística da Universidade de *Auckland*, Nova Zelândia, e é uma implementação *open source* do **S**, uma linguagem para análise de dados desenvolvida nos Laboratórios *Bell* (*Becker e al.* 1988).

O **R** encontra-se disponível para *download* em www.r-project.org. Há versões de **R** para sistemas operativos *Linux*, *MacOS X* e *Windows*. A instalação do **R** é realizada a partir de um pacote denominado “*base*”.

2.3.1 Notação na utilização de comandos e apresentação de resultados

A grande maioria dos comandos apresentados foram gravados em ficheiros de *script* e como tal não apresentam nada à esquerda do comando.

Exemplo: gerar uma sequência de 1 a 20 num *script*

```
x <- seq(from=1, to =20, by = 1)      # gera sequência de 1 a 20
```

Os comandos executados no *workspace* do **R** são precedidos por “>”

Exemplo: gerar uma sequência de 1 a 20 no *workspace*

```
> x <- seq(from=1, to =20, by = 1)
```

Os resultados (*outputs*) de **R** são precedidos de “ [#] ”, pelo menos a sua maioria,

Exemplo: resultado da sequência de 1 a 20

```
[1] 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
```

Se o comando for executado numa janela de *script* dentro de parêntesis curvos, além do comando aparece também o resultado no *workspace*.

Exemplo: geração de uma sequência com incremento 2

```
(x <- seq(from=0, to =10, by = 2))    # sequência de 0 a 10 com incremento 2  
[1] 0 2 4 6 8 10                    # output
```


Há casos em que os resultados (*outputs*) não são precedidos por “[número]”, normalmente são resultado de funções ou comandos que apresentam várias linhas com resultados. O que em cada linha fica à direita de “#”, depois de um comando é comentário.

Exemplo: informação sumária de estatísticas da sequência gerada

```
> summary(x) # isto é um comentário
  Min. 1st Qu. Median Mean 3rd Qu. Max.
  0.0  2.5    5.0   5.0  7.5   10.0
```

O código de **R**, necessário para o cálculo e apresentação de resultados, ao longo de todo o trabalho será apresentado em caixas (como os exemplos anteriores). Está fora do âmbito deste trabalho explicar detalhadamente como se trabalha com o **R**, apenas abordaremos como utilizar o **R** para a análise de séries temporais.


Note-se que não é necessário que as sequências de comandos sejam realizadas exactamente da forma como são apresentados. O **R** tem flexibilidade para que cada utilizador construa as suas próprias funções e, à medida que se avança, se possam ir automatizando procedimentos para utilizações posteriores. Por vezes, para se chegar a um resultado, podem-se realizar várias operações que podem estar incluídas num único comando, depende do que se pretende mostrar e do nível de conhecimentos do utilizador.

2.3.2 Considerações gerais elementares sobre o R

O **R** tem uma versão de instalação denominada base, além desse pacote base é possível instalar livrarias específicas para realizar certas e determinadas operações. As livrarias, regra geral, também incluem dados que se podem utilizar e que estão incluídos no *help* de cada livraria.

Exemplo: como instalar uma livraria e visualizar o seu conteúdo.

```
install.packages("tseries") # realiza o download da livraria tseries para o computador
library(tseries)           # carrega a livraria tseries no workspace do R
library(help="tseries")    # visualizar o conteúdo da livraria tseries
```



```

R Documentation for package 'tseries'

Information on package 'tseries'

Description:
Package:          tseries
Version:          0.10-22
Date:             2009-11-22
Title:            Time Series analysis and computational finance
Author:           Compiled by Adrian Trapletti
                  <a.trapletti@swissonline.ch>
Maintainer:      Kurt Hornik <Kurt.Hornik@R-project.org>
Description:      Package for time series analysis and computational
                  finance
Depends:          R (>= 2.4.0), quadprog, stats, zoo
Suggests:         its
Imports:          graphics, stats, utils
License:          GPL-2
Packaged:         2009-11-22 19:03:45 UTC; hornik
Repository:       CRAN
Date/Publication: 2009-11-22 19:06:50
Built:            R 2.11.1; i386-pc-mingw32; 2010-10-27 17:56:12 UTC;
                  windows

Index:
NelPlo           Nelson-Plosser Macroeconomic Time Series
USeconomic       U.S. Economic Variables

```

Figura 3: Conteúdo da livreria tseries

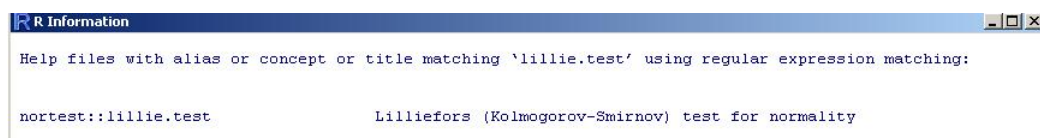
A livreria *tseries* é importante na análise de séries temporais. É onde se encontra, por exemplo, o comando “*arma()* → *Fit ARMA models to Time Series*”.

Ao se executar uma função, a qual se tem a certeza que existe, pode ocorrer um erro. Neste caso específico, o erro é devido a não estar carregada a livreria que permite executar o comando,

```

> lillie.test(x1) # aplicar o lillie.test à variável x1
Error: could not find function "lillie.test" # mensagem de erro, a função não está presente no workspace
> help.search("lillie.test") # procurar informação sobre a função lillie.test, abre a janela na figura
# seguinte

```



```

R Information
Help files with alias or concept or title matching 'lillie.test' using regular expression matching:

nortest::lillie.test           Lilliefors (Kolmogorov-Smirnov) test for normality

```

Figura 4: Resultado do *help* do R sobre o comando *lillie.test*

Como se pode verificar pela informação na figura 4, o *lillie.test* realiza o teste de normalidade de *Kolmogorov-Smirnov*, que se encontra na livreria *nortest*. O erro foi provocado por esta livreria específica não estar carregada no *workspace*.

Em caso de dúvida relativamente às funcionalidades de um comando, basta colocar um ponto de interrogação seguido do comando a pesquisar.

Exemplo: pesquisa sobre funcionalidades do comando ***ar***.

```
?ar # abre janela de ajuda sobre o comando ar
```

O *browser*, definido por defeito, abre uma janela em que se descreve e explica a utilização do comando e regra geral, no final há sempre exemplos de aplicação. Abaixo transcreve-se parte da ajuda sobre a função *ar*, que será utilizada posteriormente.

```

ar {stats} # o comando ar faz parte da livreria stats

Fit Autoregressive Models to Time Series

Description
Fit an autoregressive time series model to the data, by default selecting the complexity by AIC.

Usage
ar(x, aic = TRUE, order.max = NULL,
  method=c("yule-walker", "burg", "ols", "mle", "yw"),
  na.action, series, ...)
...

```

2.4 Processos Estocásticos

Os processos estocásticos, constituem a base dos conceitos fundamentais teóricos de suporte às sucessões cronológicas. Dada a extensão, especificidades e desenvolvimentos pormenorizados necessários para contextualizar resultados apresentados e utilizados, é aconselhável a consulta das referências bibliográficas onde se encontram em detalhe os temas abordados.

Considere-se um processo estocástico em tempo discreto, $\{X_t : t = 0, \pm 1, \pm 2, \dots\}$.

Para cada t , X_t é uma variável aleatória com a seguinte função de distribuição,

$$F(x, t) = P(X_t \leq x), \quad -\infty \leq x \leq +\infty \quad (2.1)$$

o que constitui uma descrição de 1ª ordem do processo.

A média e a variância, se existirem, designam-se por,

$$E(X_t) = \mu_t, \quad (2.2)$$

$$\text{Var}(X_t) = E\{(X_t - \mu_t)^2\} = \sigma_t^2. \quad (2.3)$$

Quando se tem uma sucessão de variáveis aleatórias i.i.d., como sucede na amostragem casual com reposição de uma população, a distribuição de 1ª

ordem é suficiente para a caracterizar. No entanto, sempre que as variáveis aleatórias da família associada ao processo não são independentes, como se verifica, regra geral, nos casos de sucessões cronológicas, passa a haver interesse nas descrições de ordem mais elevada. Assim, a descrição de 2ª ordem é particularmente importante, dado que envolve a função de distribuição de qualquer par X_{t_1}, X_{t_2} ,

$$F(x_1, x_2; t_1, t_2) = P(X_{t_1} \leq x_1; X_{t_2} \leq x_2), \quad (2.4)$$

permitindo estabelecer (com t_1 e t_2 inteiros arbitrários), a covariância e a correlação entre X_{t_1} e X_{t_2} . Estas duas estatísticas traduzem um importante aspecto das relações de dependência inerentes ao processo. Se existirem os respectivos valores esperados, a covariância e a correlação têm por expressão

$$\text{covariância} \rightarrow \gamma(t_1, t_2) = E\{(X_{t_1} - \mu_{t_1})(X_{t_2} - \mu_{t_2})\}, \quad (2.5)$$

$$\text{correlação} \rightarrow \rho(t_1, t_2) = \frac{\gamma(t_1, t_2)}{\sigma_{t_1} \sigma_{t_2}}. \quad (2.6)$$

2.4.1 Processos Estocásticos Estacionários

À medida que se avança na tentativa de modelação de uma sucessão cronológica é necessário saber se o processo estocástico é ou não temporalmente invariante do ponto de vista probabilístico. Se as características probabilísticas do processo estocástico variam ao longo do tempo, isto é, se o processo não é estacionário, torna-se difícil representar tanto as observações passadas bem como as futuras através de um modelo algébrico. Por outro lado, se o processo tem características probabilísticas temporalmente invariantes, isto é, se o processo é estacionário, então é possível representar o processo através de um modelo algébrico com coeficientes que podem ser estimados a partir de observações anteriores.

Um processo estacionário apresenta um equilíbrio estatístico em torno de um nível médio fixo, isto é, estruturalmente tem propriedades probabilísticas que são estáveis ou invariantes ao longo do tempo.

Considere-se um processo estocástico,

$$\{X_t : t = 0, \pm 1, \pm 2, \dots\}.$$

Sejam $t_1, t_2, \dots, t_N \in \mathbb{Z}$. O processo diz-se:

- (i) estacionário de 1ª ordem em distribuição se

$$F(x_1; t_1) = F(x_1; t_1 + k), \quad \forall t_1, k \in \mathbb{Z} \tag{2.7}$$

- (ii) estacionário de 2ª ordem em distribuição se

$$F(x_1, x_2; t_1, t_2) = F(x_1, x_2; t_1 + k, t_2 + k), \quad \forall t_1, t_2, k \in \mathbb{Z} \tag{2.8}$$

- (iii) estacionário de n-ésima ordem em distribuição se,

$$\begin{aligned} F(x_1, x_2, \dots, x_N; t_1, t_2, \dots, t_N) = \\ = F(x_1, x_2, \dots, x_N; t_1 + k, t_2 + k, \dots, t_N + k) \quad \forall t_1, t_2, \dots, t_N, k \in \mathbb{Z}. \end{aligned} \tag{2.9}$$

Um processo diz-se estritamente (ou fortemente) estacionário quando verifica (2.9) para qualquer N , $N = 1, 2, \dots$. Esta propriedade é de difícil verificação na prática, motivo pelo qual se utiliza a estacionariedade até à 2ª ordem que se introduz de seguida.

O processo diz-se (fracamente) estacionário de ordem n se, dados quaisquer inteiros t_1, \dots, t_N , os vectores aleatórios $(X_{t_1}, \dots, X_{t_N})$ e $(X_{t_1+k}, \dots, X_{t_N+k})$ têm os mesmos momentos até à ordem N .

A estacionariedade até 2ª ordem implica média e variância constantes. Ou seja, caso o valor esperado exista, $(E(|X_t|^2) < \infty)$,

$$E(X_t) = \mu, \quad \forall t \in \mathbb{Z} \tag{2.10}$$

$$Var(X_t) = \sigma^2, \quad \forall t \in \mathbb{Z}$$

$$Cov(X_t, X_s) = Cov(X_{t+k}, X_{s+k}) = \gamma(|t-s|), \quad \forall t, s \in \mathbb{Z}$$

Isto é, a covariância entre as variáveis, X_t e X_s , (representada por $\gamma(t, s)$ toma sempre o mesmo valor para todo o t e s , dependendo apenas da diferença de tempo entre as variáveis, $|t-s|$.

NOTA

Os processos fracamente estacionários até à segunda ordem, também são ditos estacionários no sentido lato ou estacionários em covariância.

Salvo menção em contrário, daqui em diante, a referência a processos estacionários será sempre relativa aos processos estacionários até à 2ª ordem.

2.4.2 Funções de Autocovariância e Autocorrelação

No estudo de sucessões cronológicas em tempo discreto têm um papel preponderante as funções de autocovariância e de autocorrelação, que de seguida se introduzem.

Seja $\{X_t : t = 0, \pm 1, \pm 2, \dots\}$, ou daqui para diante $\{X_t\}$, um processo estacionário com média e variância dadas por (2.10). O valor esperado,

$$\gamma_k = E\{(X_t - \mu)(X_{t+k} - \mu)\}, \quad (2.11)$$

que existe por hipótese, pode calcular-se para $k = 0, \pm 1, \pm 2, \dots$, e define a função de autocovariância do processo. Para cada k , a função γ_k mede a intensidade com que covariam pares de valores do processo separados por um intervalo - "lag" - de amplitude k .

Define-se também a função de autocorrelação do processo (abreviadamente, FAC) para cada k , como

$$\rho_k = \frac{\gamma_k}{\gamma_0}, \quad (2.12)$$

A representação gráfica de ρ_k em função de k designa-se por correlograma teórico e mede a correlação entre pares de valores do processo separados por um intervalo k .

Intuitivamente, interpreta-se ρ_k como uma medida da semelhança entre cada realização e a mesma realização deslocada k unidades de tempo.

O correlograma é um dos primeiros instrumentos de análise utilizados para tentar descortinar características de comportamento do processo subjacente à série de dados observados.

Normalmente, à medida que k aumenta, há uma tendência de decrescimento de ρ_k e de γ_k . A forma como se verifica o decaimento de ρ_k pode interpretar-se como uma medida da memória do processo.

2.4.3 Função de Autocorrelação Parcial

Além da correlação total, também é importante para a análise do processo a autocorrelação que possa existir entre X_t e X_{t+k} quando se fixam as variáveis intermédias $X_{t+1}, X_{t+2}, \dots, X_{t+k-1}$. Isto é, a correlação simples entre X_t e X_{t+k} depois de eliminar o efeito que sobre eles produzem as variáveis intermédias $X_{t+1}, X_{t+2}, \dots, X_{t+k-1}$.

A função de autocorrelação parcial (abreviadamente, FACP), ϕ_{kk} , é dada por,

$$\phi_{kk} = \frac{\begin{vmatrix} 1 & \rho_1 & \dots & \rho_{k-2} & \rho_1 \\ \rho_1 & 1 & \dots & \rho_{k-3} & \rho_2 \\ \dots & \dots & \dots & \dots & \dots \\ \rho_{k-1} & \rho_{k-2} & \dots & \rho_1 & \rho_k \end{vmatrix}}{\begin{vmatrix} 1 & \rho_1 & \rho_2 & \dots & \rho_{k-1} \\ \rho_1 & 1 & \rho_1 & \dots & \rho_{k-2} \\ \dots & \dots & \dots & \dots & \dots \\ \rho_{k-1} & \rho_{k-2} & \rho_{k-3} & \dots & 1 \end{vmatrix}}. \tag{2.13}$$

2.4.4 Ruído Branco

Um exemplo típico e que desempenha um papel fundamental na construção de muitos outros processos, lineares e não lineares, é o chamado **processo de ruído branco**.

Um processo $\{X_t, t = 0, \pm 1, \pm 2, \dots\}$ diz-se puramente aleatório ou ruído branco se

$$\text{Cov}(X_t, X_s) = 0, \quad \forall t \neq s.$$

$$E(X_t) = \mu, \quad \forall t$$

$$\text{Var}(X_t) = \sigma_x^2, \quad \forall t.$$

Salvo menção em contrário designa-se daqui por diante ruído branco por $\{\varepsilon_t\}$, com

$$\begin{cases} E(\varepsilon_t) = \mu_\varepsilon \\ V(\varepsilon_t) = \sigma_\varepsilon^2 \end{cases}, \quad \gamma_k = \begin{cases} \sigma_\varepsilon^2, & k = 0 \\ 0, & k \neq 0 \end{cases}, \quad \rho_k = \begin{cases} 1, & k = 0 \\ 0, & k \neq 0 \end{cases}. \quad (2.18)$$

Salvo menção em contrário, iremos supor que $\mu_\varepsilon = 0$. Quando seja conveniente considerar $\mu_\varepsilon \neq 0$, nas expressões onde esteja ε_t , deve substituir-se por

$$\varepsilon_t^* = \varepsilon_t - \mu.$$

2.4.5 Estimação dos parâmetros caracterizadores dos processos estacionários

Na modelação de uma sucessão cronológica é extremamente importante a estimação de parâmetros que caracterizem o processo estacionário subjacente.

Os processos estacionários são caracterizados pela média μ , a variância σ^2 , as autocorrelações ρ_k e as autocorrelações parciais ϕ_{kk} .

De seguida apresentam-se os estimadores mais usuais destes parâmetros.

O estimador da **média** é

$$\bar{X} = \frac{1}{N} \sum_{t=1}^N X_t, \quad (2.15)$$

Prova-se que este estimador é centrado e não enviesado.

Existem dois estimadores usuais, para a **função de autocovariância**, ambos não centrados. Contudo, o mais utilizado é o dado pela seguinte expressão,

$$\hat{\gamma}_k = \frac{1}{N} \sum_{t=1}^{N-k} (X_t - \bar{X})(X_{t+k} - \bar{X}), \quad \text{com } 0 \leq k \leq N-1. \quad (2.16)$$

Este estimador é o que apresenta menor variância, além de que é uma função não-negativa.

O estimador da **função de autocorrelação** é dado por,

$$\hat{\rho}_k = \frac{\hat{\gamma}_k}{\hat{\gamma}_0} = \frac{\sum_{t=1}^{N-k} (X_t - \bar{X})(X_{t+k} - \bar{X})}{\sum_{t=1}^N (X_t - \bar{X})^2} . \tag{2.17}$$

A representação gráfica de $\hat{\rho}_k$ em função de k designa-se por correlograma estimado. Dado que as funções de autocorrelação e autocovariância gozam de simetria, a estimação tanto de uma, como de outra, é somente, efectuada para valores de $k \geq 0$.

A estimação da **função de autocorrelação parcial** $\hat{\phi}_{kk}$ é obtida recursivamente pelo algoritmo de *Durbin-Levinson* (Brockwell & Davies [1987]), que é inicializado da seguinte forma:

$$\begin{aligned} \hat{\phi}_{kk} &= \hat{\rho}_1, \\ \hat{\phi}_{kk} &= \frac{\hat{\rho}_k - \sum_{j=1}^{k-1} \hat{\phi}_{k-1,j} \hat{\rho}_{k-j}}{1 - \sum_{j=1}^{k-1} \hat{\phi}_{k-1,j} \hat{\rho}_j}, \end{aligned} \tag{2.18}$$

Onde,

$$\hat{\phi}_{kj} = \hat{\phi}_{k-1,j} - \hat{\phi}_{kk} \hat{\phi}_{k-1,k-j}, \quad j = 1, 2, \dots, k-1$$

Esta expressão deriva da substituição em (2.13) da autocorrelação teórica, ρ_k , pela autocorrelação estimada, $\hat{\rho}_k$.

2.4.6 Processos Estocásticos Estacionários e Lineares

2.4.6.1 Processos Autoregressivos de 1ª Ordem

O processo X_t diz-se Autoregressivo de 1ª ordem (ou ordem 1), de média nula, abreviadamente AR(1), caso satisfaça a equação às diferenças estocástica,

$$X_t = \phi X_{t-1} + \varepsilon_t, \quad (2.19)$$

ou,

$$(1 - \phi B)X_t = \varepsilon_t.$$

A $\phi(B) = 1 - \phi B$ chama-se polinómio autoregressivo de 1ª ordem e B é denominado operador de atraso, tal que $BX_t = X_{t-1}$. Prova-se que este processo é sempre invertível e estacionário se $|\phi| < 1$.

Após breves cálculos, estabelece-se a seguinte expressão para a FAC dos processo AR(1),

$$\rho_k = \phi^k, \quad k = 1, 2, \dots, \quad (2.20)$$

No quadro 2.1 (pag. 48), encontram-se, em forma de resumo, não só as condições em que este processo (e os que se apresentarão de seguida) admitem estacionariedade e invertibilidade, bem como outras características importantes das FAC e FACP.

De igual modo, obtém-se a seguinte estrutura para a FACP do processo AR(1)

$$\phi_{11} = \phi, \quad \phi_{kk} = 0, \quad k \geq 2. \quad (2.21)$$

Quando k aumenta a FAC tende exponencialmente para zero, eventualmente com alternância de sinal (quando $\phi < 0$), enquanto a FACP decai bruscamente para zero para $k \geq 2$.

Tal como já foi referido, o **R** permite simular e analisar séries temporais. De seguida mostra-se como se realiza a simulação de um processo AR(1), se visualiza o cronograma e as respectivas FAC e a FACP de um processo AR(1).

A metodologia de simulação adoptada foi a seguinte:

- fixa-se a semente de geração de números aleatórios
- define-se o parâmetros ou parâmetros do modelo
- define-se um vector onde se “guardam” os valores da geração de uma série de ruído branco

- gera-se o processo através de um ciclo de acordo com o modelo em causa
- representa-se do cronograma do modelo e
- representa-se das FAC e FACP do modelo.

A metodologia de simulação adoptada para este caso (em termos de código de **R**), é generalizada para os casos seguintes que iremos introduzir.

Assim, mostra-se recorrendo a um processo de simulação, qual o comportamento típico de cada um dos processos que se vão introduzindo, mas não de forma exaustiva, dado que as mudanças de sinal nos parâmetros e outras variações possíveis não só conduzem a comportamentos perfeitamente identificados na literatura, como também são facilmente implementadas no código apresentado.

De acordo com o referido no parágrafo anterior, a geração do processo AR(1) apresentado em (2.19) pode realizar-se, no **R**, da seguinte forma,

```

set.seed(1)                                # fixa a semente
PHI<- 0.7                                  # valor de phi1
x <- w <- rnorm(250)                        # inicialização da série e geração do ruído branco gaussiano
                                             # com 250 observações

for (t in 2:250) x[t] <-PHI * x[t - 1] + w[t] # ciclo que constrói a série

plot(x, type = "l", xlab= "Tempo", main=" AR(1)") # cronograma
acf(x, main= "Função de Autocorrelação")         # FAC
pacf(x,main= "Função de Autocorrelação Parcial") # FACP

```

donde resulta o seguinte cronograma,

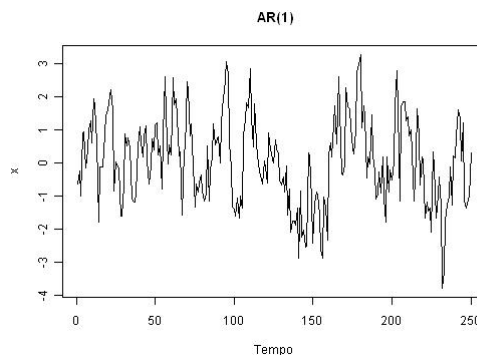


Figura 5: Cronograma de um processo AR(1) simulado, $\phi = 0.7$

Com FAC e FACP dadas por

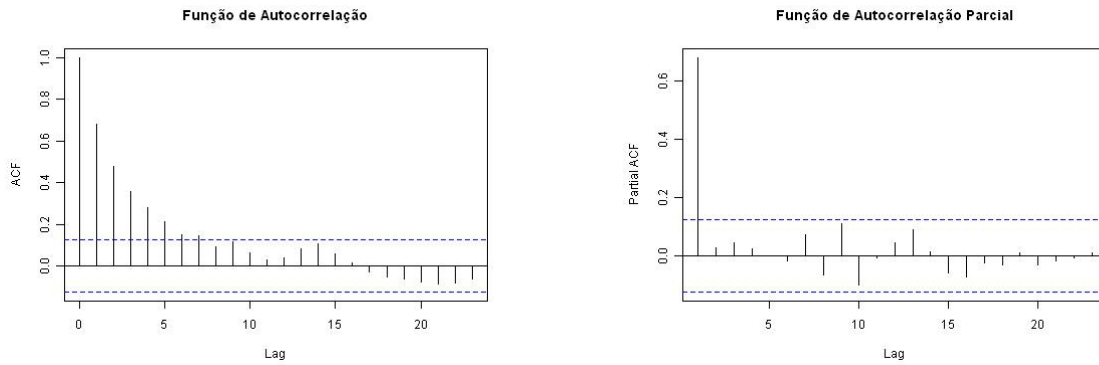


Figura 6: FAC e FACP do processo AR(1) simulado, $\phi = 0.7$

O decaimento da FAC acontece na forma de uma exponencial amortecida, pois ϕ é positivo, enquanto a FACP se anula abruptamente a partir do lag 1, dado tratar-se de um processo AR(1).

Mantendo as mesmas rotinas, e alterando o parâmetro para $\phi = -0.7$, obtém-se

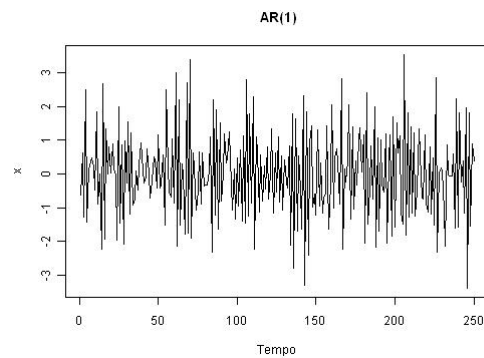


Figura 7: Cronograma de um processo AR(1) simulado, $\phi = -0.7$

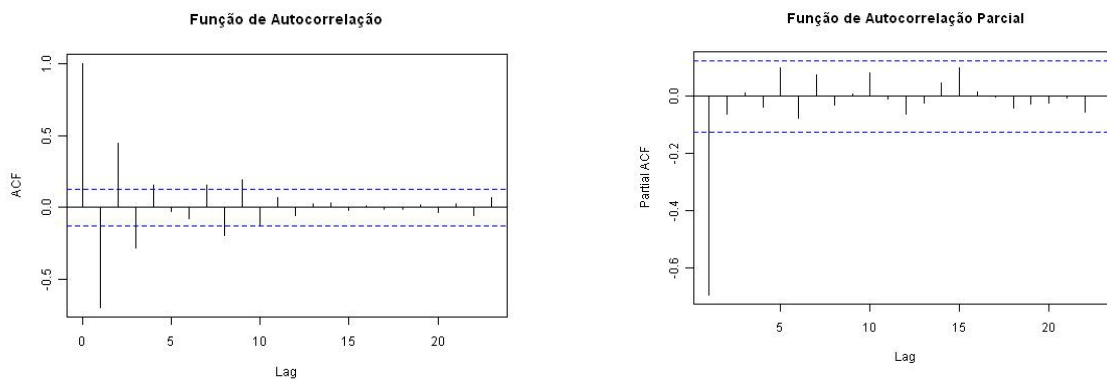


Figura 8: FAC e FACP do processo AR(1) simulado, $\phi = -0.7$

A FAC neste caso, e dado que ϕ é negativo, tem um amortecimento da forma sinusoidal, enquanto a FACP tende, de igual modo, rapidamente para zero, anulando-se a partir do lag 1.

2.4.6.2 Processos Autoregressivos de 2ª Ordem

O processo X_t diz-se AR(2), de média nula, quando satisfaz a equação às diferenças estocástica,

$$X_t - \phi_1 X_{t-1} - \phi_2 X_{t-2} = \varepsilon_t, \tag{2.23}$$

ou

$$\phi_2(B)X_t = \varepsilon_t,$$

com $\phi_2(B) = 1 - \phi_1 B - \phi_2 B^2$ que se designa polinómio autoregressivo de 2ª ordem.

Para os processos AR(2) prova-se que a função de autocorrelação é dada por,

$$\begin{aligned} \rho_1 &= \frac{\phi_1}{1 - \phi_2}, \\ \rho_2 &= \phi_2 + \frac{\phi_1^2}{1 - \phi_2}, \end{aligned} \tag{2.24}$$

Do mesmo modo, prova-se que a função de autocorrelação parcial é dada por,

$$\begin{aligned} \phi_{11} &= \rho_1 = \frac{\phi_1}{1 - \phi_2}, \\ \phi_{22} &= \frac{\begin{vmatrix} 1 & \rho_1 \\ \rho_1 & \rho_2 \end{vmatrix}}{\begin{vmatrix} 1 & \rho_1 \\ \rho_1 & 1 \end{vmatrix}} = \frac{\rho_2 - \rho_1^2}{1 - \rho_1^2} = \dots = \phi_2 \\ \phi_{33} &= \frac{\begin{vmatrix} 1 & \rho_1 & \rho_2 \\ \rho_1 & 1 & \rho_2 \\ \rho_2 & \rho_1 & \rho_3 \end{vmatrix}}{\begin{vmatrix} 1 & \rho_1 & \rho_2 \\ \rho_1 & 1 & \rho_1 \\ \rho_2 & \rho_1 & 1 \end{vmatrix}} = \dots = 0 \end{aligned} \tag{2.25}$$

A FACP, $\phi_{kk} = 0$, para $k \geq 3$. Ou seja, todo o processo AR(2) tem FACP nula a partir do *lag* 2.

As deduções da FAC e FACP, bem como as condições de estacionariedade (quadro 2.1) podem ser consultadas em *Muller et al.* [1990, (p.51)], *Wei* [1994, (p.38)] [1994] e *Pyndick et al.* [1998, (p.522)].

Recorrendo à programação em **R**, e de acordo com os mesmos critérios utilizados para o processo AR(1) para gerar o processo, representar o cronograma, a FAC e FACP, tem-se,

```

set.seed(1)                # fixa a semente
PHI1<- -0.5                # valor de phi1
PHI2 <- -0.3               # valor de phi2
x <- w <- rnorm(250)       # inicialização da série e geração do ruído branco gaussiano
for (t in 3:250) x[t] <- PHI1 * x[t - 1] + PHI2 * x[t - 2] + w[t] # ciclo que constrói a série
plot(x, type = "l", xlab= "Tempo", main=" AR(2)")                # cronograma
acf(x, main= "Função de Autocorrelação")                        # FAC
pacf(x,main= "Função de Autocorrelação Parcial")                # FACP

```

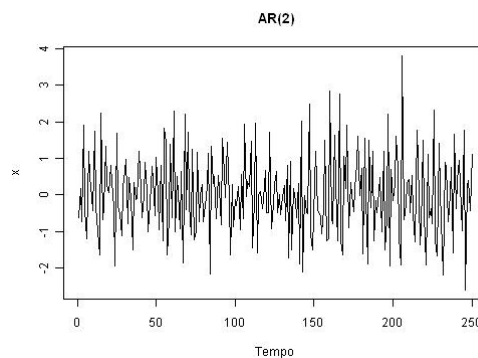


Figura 9: Cronograma de um processo AR(2) simulado, $\phi_1 = -0.5, \phi_2 = -0.3$

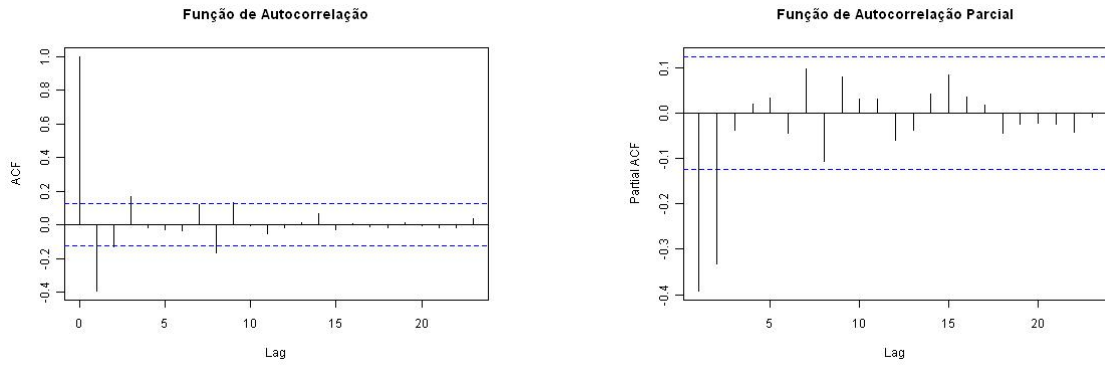


Figura 10: FAC e FACP do processo AR(2) simulado, $\phi_1 = -0.5, \phi_2 = -0.3$

Tal como se pode verificar, dados os dois coeficientes autoregressivos serem negativos o decaimento da FAC é da forma sinusoidal e a FACP decai rapidamente para zero a partir do lag 2.

2.4.6.3 Processos Autoregressivos de Ordem p

O processo X_t diz-se autoregressivo de ordem p , AR(p), de média nula quando satisfaz a equação às diferenças estocástica,

$$X_t - \phi_1 X_{t-1} - \phi_2 X_{t-2} - \dots - \phi_p X_{t-p} = \varepsilon_t, \tag{2.26}$$

ou,

$$\phi_p(B)X_t = \varepsilon_t,$$

com, $\phi_p(B) = 1 - \phi_1 B - \phi_2 B^2 - \dots - \phi_p B^p$.

A FAC obtém-se resolvendo o sistema de equações de *Yule-Walker*, que assume a forma matricial

$$\begin{bmatrix} 1 & \rho_1 & \rho_2 & \dots & \rho_{p-1} \\ \rho_1 & 1 & \rho_1 & \dots & \rho_{p-2} \\ \rho_2 & \rho_1 & 1 & \dots & \rho_{p-3} \\ \dots & \dots & \dots & \dots & \dots \\ \rho_{p-1} & \rho_{p-2} & \rho_{p-3} & \dots & 1 \end{bmatrix} \begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \dots \\ \phi_p \end{bmatrix} = \begin{bmatrix} \rho_1 \\ \rho_2 \\ \rho_3 \\ \dots \\ \rho_p \end{bmatrix}, \tag{2.27}$$

permitindo, assim, determinar ρ_1, \dots, ρ_p .

Para $k = 1, \dots, p$ a FACP pode obter-se pela resolução dos sistemas

$$\begin{bmatrix} 1 & \rho_1 & \rho_2 & \cdots & \rho_{k-1} \\ \rho_1 & 1 & \rho_1 & \cdots & \rho_{k-2} \\ \rho_2 & \rho_1 & 1 & \cdots & \rho_{k-3} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \rho_{k-1} & \rho_{k-2} & \rho_{k-3} & \cdots & 1 \end{bmatrix} \begin{bmatrix} \phi_{k1} \\ \phi_{k2} \\ \phi_{k3} \\ \cdots \\ \phi_{kk} \end{bmatrix} = \begin{bmatrix} \rho_1 \\ \rho_2 \\ \rho_3 \\ \cdots \\ \rho_p \end{bmatrix} \text{ com } (k = 1, \dots, p), \quad (2.28)$$

ou pelo algoritmo de *Durbin-Levinson*.

As condições de estacionariedade estão resumidas no quadro 2.1.

2.4.6.4 Processos Médias Móveis de Ordem q

Os processos médias móveis de ordem q de média nula são da forma.

$$X_t = \varepsilon_t - \theta_1 \varepsilon_{t-1} - \theta_2 \varepsilon_{t-2} - \dots - \theta_q \varepsilon_{t-q} \text{ ou } X_t = \theta_q(B) \varepsilon_t, \quad (2.29)$$

$$\theta_q(B) = 1 - \theta_1 B - \theta_2 B^2 - \dots - \theta_q B^q,$$

onde $\theta_q(B)$ é o polinómio médias móveis de ordem q .

Os processos médias móveis resultam da ideia de exprimir o processo X_t em termos de um processo mais simples ε_t . O modelo médias móveis implica que os efeitos produzidos pelas inovações, só perduram por um curto período de tempo, o que contrasta com o que se passa com os processos autoregressivos, em que os efeitos persistem por um tempo mais longo.

Prova-se que a FAC (*Muller et al.* (p.59) e *Wei* (p.51)) tem por expressão,

$$\rho_k = \begin{cases} \frac{-\theta_k + \theta_{k+1}\theta_1 + \dots + \theta_q\theta_{q-k}}{1 + \theta_1^2 + \dots + \theta_q^2}, & 0 \leq k \leq q \\ 0 & k > q \end{cases}, \quad (2.30)$$

É de assinalar especialmente a queda brusca da FAC dos processo MA(q) para $k \geq q + 1$.

Analogamente, mostra-se que a FACP é dada pela expressão

$$\phi_{kk} = \frac{-\theta^k(1-\theta^2)}{1-\theta^{2(k+1)}}, \quad k \geq 1.$$

A FACP decresce de forma exponencial, por valores positivos, ou por valores negativos, ou ainda por valores alternados, positivos e negativos, dependendo do sinal do parâmetro.

O processo MA(1) é, claro, um caso particular de um processo MA(q) quando q=1 e o processo MA(2) é o caso particular em que q=2.

O processo MA(1) admite, portanto, a seguinte forma

$$X_t = \varepsilon_t - \theta\varepsilon_{t-1} \tag{2.31}$$

onde,

$$\begin{aligned} \rho_k &= \frac{-\theta}{1+\theta^2}, & k=1 \\ &= 0, & k \geq 2, \end{aligned} \tag{2.32}$$

verificando-se a queda brusca da FAC para zero a partir do lag 1, ou seja, quando $k \geq 2$.

A FACP tem por expressão,

$$\begin{aligned} \phi_{11} &= \rho_1 = \left(\frac{-\theta}{1+\theta^2} \right) = \frac{-\theta(1-\theta^2)}{1-\theta^4}, \\ \phi_{22} &= \frac{\begin{vmatrix} 1 & \rho_1 \\ \rho_1 & 0 \end{vmatrix}}{\begin{vmatrix} 1 & \rho_1 \\ \rho_1 & 1 \end{vmatrix}} = -\frac{\rho_1^2}{1-\rho_1^2} = \frac{-\theta^2}{1+\theta^2+\theta^4} = \frac{-\theta^2(1-\theta^2)}{1+\theta^6}, \\ \phi_{33} &= \frac{\begin{vmatrix} 1 & \rho_1 & \rho_1 \\ \rho_1 & 1 & 0 \\ 0 & \rho_1 & 0 \end{vmatrix}}{\begin{vmatrix} 1 & \rho_1 & 0 \\ \rho_1 & 1 & \rho_1 \\ 0 & \rho_1 & 1 \end{vmatrix}} = \frac{\rho_1^3}{1-2\rho_1^2} = \frac{-\theta^3}{1+\theta^2+\theta^4+\theta^6} = \frac{-\theta^3(1-\theta^2)}{1-\theta^8}. \end{aligned}$$

Mantendo o mesmo raciocínio na simulação, cronograma, FAC e FACP, para um processo MA(1) tem-se,

```
set.seed(1)                # fixa a semente
TETHA<- 0.7                # valor de tetha
w <- rnorm(250)            # geração do ruído branco
for (t in 2:250) x[t] <- w[t] + TETHA * w[t - 1] # ciclo que constrói a série
plot(x, type = "l", xlab= "Tempo", main=" MA(1)") # cronograma
pacf(x,main= "Função de Autocorrelação Parcial") # FACP
```

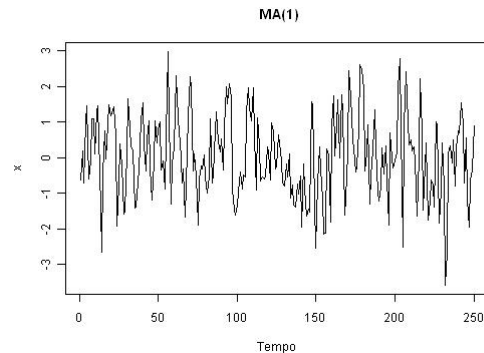


Figura 11: Cronograma do processo MA(1) simulado, $\theta = 0.7$

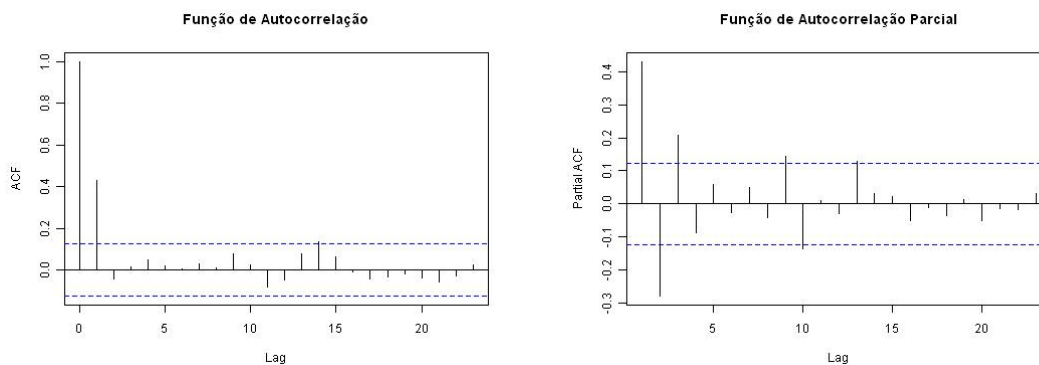


Figura 12: FAC e FACP do processo MA(1) simulado, $\theta = 0.7$

Tal como foi referido no parágrafo anterior, pode facilmente verificar-se que a FAC tem um decaimento rápido para zero a partir do *lag* 1, enquanto a FACP apresenta um decaimento lento de forma sinusoidal para zero .

A título meramente exemplificativo veja-se o que acontece quando θ é negativo

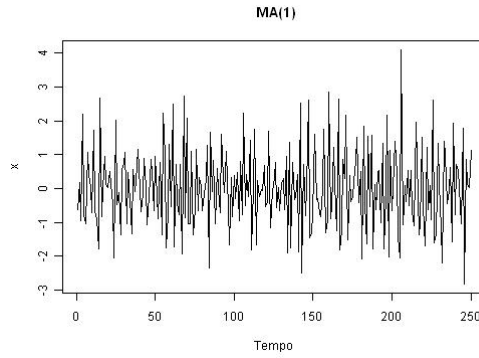


Figura 13: FAC e FACP do processo MA(1) simulado, $\theta = -0.7$

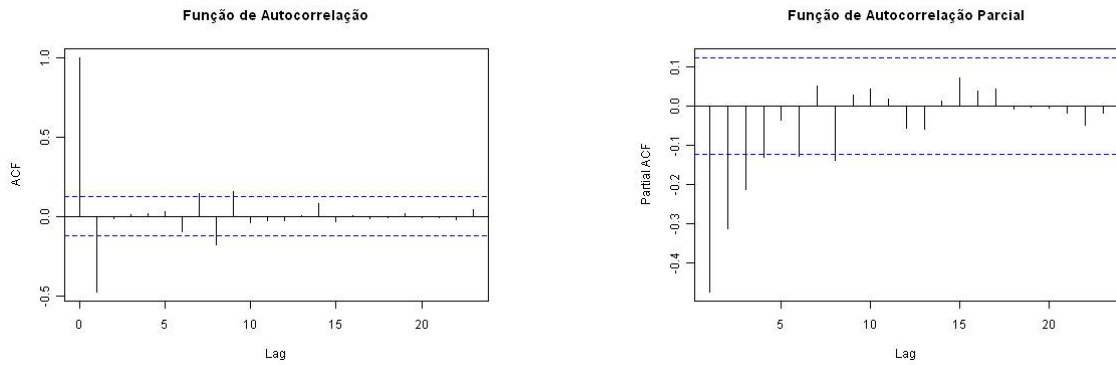


Figura 14: FAC e FACP do processo MA(1) simulado, $\theta = -0.7$

A FAC apresenta decaimento rápido para zero enquanto a FACP é mais lento e de forma exponencial, dado θ ser negativo.

No caso de um processo $MA(2)$ tem-se:

$$X_t = \varepsilon_t - \theta_1 \varepsilon_{t-1} - \theta_2 \varepsilon_{t-2} \quad \text{ou} \quad X_t = \theta_2(B)\varepsilon_t, \tag{2.34}$$

$$\theta_2(B) = 1 - \theta_1 B - \theta_2 B^2,$$

com FAC dada por

$$\rho_1 = \frac{-\theta_1(1-\theta_2)}{1+\theta_1^2+\theta_2^2}, \tag{2.35}$$

$$\rho_2 = \frac{-\theta_2}{1+\theta_1^2+\theta_2^2},$$

$$\rho_k = 0, \quad k \geq 3,$$

e FACP por

$$\phi_{11} = \rho_1 \tag{2.36}$$

$$\phi_{22} = \frac{\rho_2 - \rho_1^2}{1 - \rho_1^2},$$

$$\phi_{33} = \frac{\rho_1^3 - \rho_1\rho_2(2 - \rho_2)}{1 - \rho_2^2 - 2\rho_1^2(1 - \rho_2)},$$

...

A simulação, seguindo a lógica dos casos anteriores, de um processo MA(2), com o cronograma, a FAC e FACP de um processo MA(2) é realizada por,

```

set.seed(1)                                # fixa a semente
TETHA1<- -0.7                              # valor de tetha1
TETHA2<- -0.5                              # valor de tetha2
w <- rnorm(250)                             # geração do ruído branco
for (t in 3:250) x[t] <- w[t] + TETHA1 * w[t - 1] + TETHA2 * w[t - 2] # ciclo que constrói a série
plot(x, type = "l", xlab= "Tempo", main=" MA(2)") # cronograma
acf(x, main= "Função de Autocorrelação")      # FAC
pacf(x,main= "Função de Autocorrelação Parcial") # FACP

```

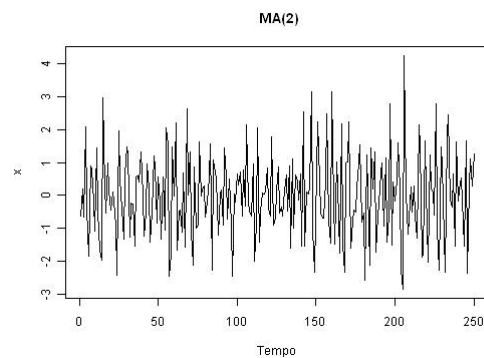


Figura 15: Cronograma do processo MA(2) simulado, $\theta_1 = -0.7, \theta_2 = -0.5$

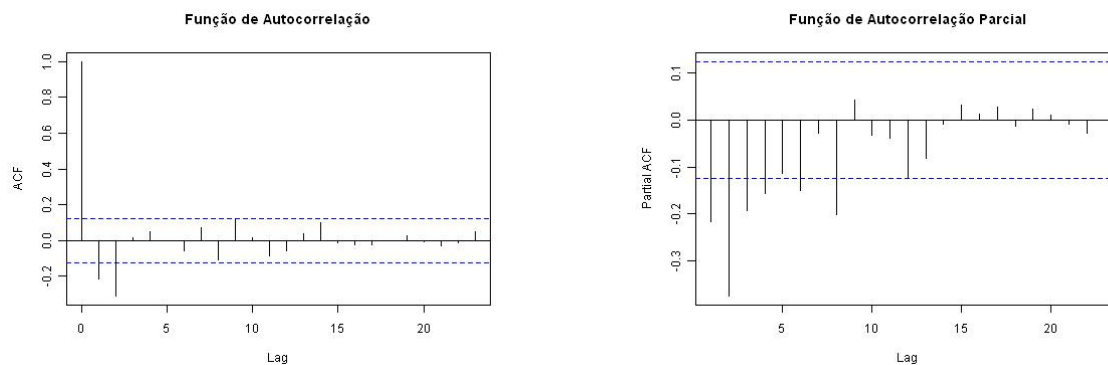


Figura 16: FAC e FACP do processo MA(2) simulado, $\theta_1 = -0.7, \theta_2 = -0.5$

Como se observa a FAC decai rapidamente para zero a partir do *lag* 2, enquanto a FACP decai lentamente de forma exponencial para zero.

Semelhanças comportamentais das FAC e FACP de AR e MA.

Em resumo, a FAC dos processos AR(p) comporta-se como a FACP dos processos MA(q). Ou seja, tipicamente decai gradualmente para zero; a FACP dos processos AR(p) comporta-se como a FAC dos processos MA(q), decaindo bruscamente para zero.

2.4.6.5 Processos Mistos Autoregressivos e Médias Móveis

Os processos mistos autoregressivos e médias móveis surgem como modelos parcimoniosos (com poucos parâmetros, por vezes conseguem-se encontrar modelos do tipo ARMA que quando modelados só com AR ou MA têm excesso de parâmetros).

O processo X_t , de média nula, diz-se ARMA(p,q) quando satisfaz a equação às diferenças estocástica,

$$X_t - \phi_1 X_{t-1} - \dots - \phi_p X_{t-p} = \varepsilon_t - \theta_1 \varepsilon_{t-1} - \dots - \theta_q \varepsilon_{t-q}, \tag{2.37}$$

ou,

$$\phi_p(B)X_t = \theta_q(B)\varepsilon_t,$$

com

$$\begin{aligned} \phi(B) &= 1 - \phi_1 B - \dots - \phi_p B^p \\ \theta(B) &= 1 - \theta_1 B - \dots - \theta_q B^q. \end{aligned}$$

A FAC de um processo ARMA(p,q) satisfaz o seguinte sistema de equações,

$$\rho_k = \phi_1 \rho_{k-1} + \dots + \phi_p \rho_{k-p}, \quad k \geq q+1. \tag{2.38}$$

A equação às diferenças (2.38) da FAC dos processos ARMA(p,q) decai gradualmente para zero a partir de $k = q+1$, ou seja, comporta-se como um

processo autoregressivo AR(p), onde os valores de ρ_1, \dots, ρ_p dependem dos parâmetros das duas componentes e funcionam como condições iniciais do padrão autoregressivo.

O facto de o processo ARMA(p,q) conter também o processo MA(q) como caso especial, faz com que a FACP seja também uma mistura de decaimento exponencial e/ou sinusóide amortecida.

As condições de estacionariedade e de invertibilidade são tais que as raízes de $\phi(B) = 0$ e de $\theta(B) = 0$ estejam fora do círculo unitário, respectivamente.

Os processos ARMA(1,1) são um caso particular dos ARMA(p,q), quando $p=q=1$, e a sua expressão tem a forma,

$$X_t - \phi X_{t-1} = \varepsilon_t - \theta \varepsilon_{t-1}, \quad (2.39)$$

ou

$$(1 - \phi B)X_t = (1 - \theta B)\varepsilon_t,$$

e possuem uma FAC da forma,

$$\rho_k = \begin{cases} 1 & , \quad k = 0, \\ \frac{(\phi_1 - \theta_1)(1 - \phi_1\theta_1)}{1 + \theta_1^2 - 2\phi_1\theta_1} & , \quad k = 1, \\ \phi_1\rho_{k-1} & , \quad k \geq 2, \end{cases}$$

Para simular um processo deste tipo, utiliza-se a mesma lógica dos casos anteriores, desta vez, incluem-se um parâmetro do tipo autoregressivo “PHI” e um parâmetro do tipo média móvel, “TETHA”, tudo o resto no que ao código e gráficos diz respeito, é análogo,

```
set.seed(1) # fixa a semente
PHI<- 0.7 # valor de phi1
TETHA<- 0.5 # valor de tetha1
x <- w <- rnorm(250) # inicialização da série e geração do ruído branco
for (t in 2:250) x[t] <-PHI * x[t - 1] + w[t] +TETHA * w[t - 1] # ciclo que constrói a série
plot(x, type = "l", xlab= "Tempo", main=" ARMA(1,1)") # cronograma
acf(x, main= "Função de Autocorrelação") # FAC
pacf(x,main= "Função de Autocorrelação Parcial") # FACP
```

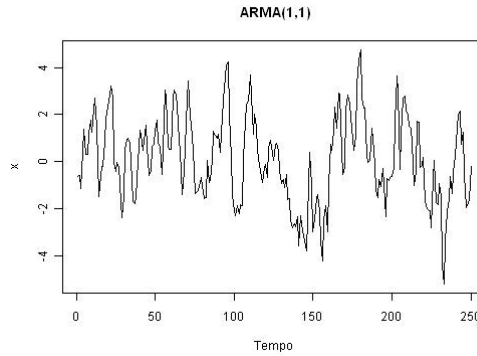


Figura 17: Cronograma do processo ARMA(1,1) simulado, $\phi = 0.7, \theta = 0.5$

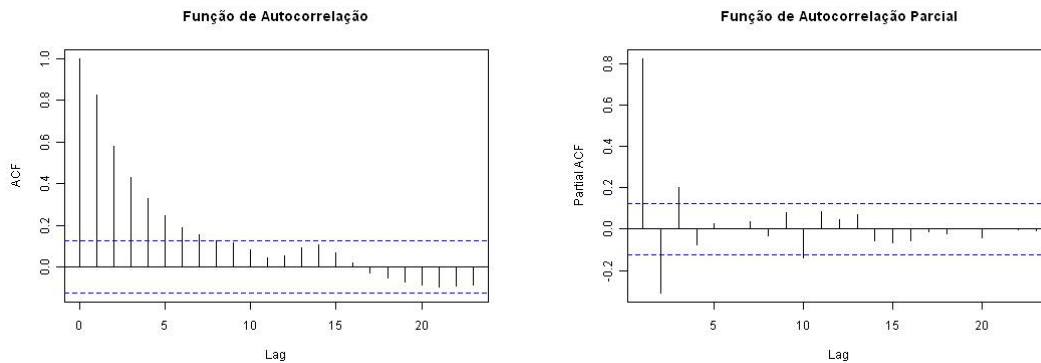


Figura 18: FAC e FACP do processo ARMA(1,1) simulado, $\phi = 0.7, \theta = 0.5$

Através da figura 18 é notório que a FAC e FACP evidenciam um comportamento que é uma mistura dos comportamentos destas funções nos casos autorregressivos e médias móveis, pois a FAC tem um decaimento lento para zero e a FACP um decaimento gradual, igualmente, para zero.

De seguida, apresentam-se resumidamente as principais características dos processos ARMA(p,q).

Embora não se tenham abordado as condições de invertibilidade e estacionariedade em todos os processos apresentados até ao momento, estas condições têm um papel fundamental quer na sua definição, quer no comportamento destes e, conseqüentemente, na sua análise. Alguns dos resultados exibidos no quadro 2.1 já foram abordados. Na bibliografia estão analisados de forma exaustiva. O objectivo deste trabalho tal como já se referiu,

consiste na aplicação do **R** na análise de séries temporais. A informação contida neste quadro e a já apresentada não dispensa a consulta de por exemplo, *Box et al [1976]*, e de *Brockwell et. Al [1991]*.

	AR(p)	MA(q)	ARMA(p,q)
Modelo em termos dos valores anteriores de X_t	$\phi_p(B)X_t = \varepsilon_t$ Série finita em X_t	$[\theta_q(B)]^{-1}X_t = \varepsilon_t$ Série infinita em X_t	$[\theta_q(B)]^{-1}\phi_p(B)X_t = \varepsilon_t$ Série infinita em X_t
Modelo em termos dos valores anteriores de ε_t	$X_t = [\phi_p(B)]^{-1}\varepsilon_t$ Série infinita em ε_t	$X_t = \theta_q(B)\varepsilon_t$ Série finita em ε_t	$X_t = [\phi_p(B)]^{-1}\theta_q(B)\varepsilon_t$ Série infinita em ε_t
Condições de estacionaridade	Raízes de $\phi_p(B) = 0$ fora do círculo unitário	Sempre estacionários	Raízes de $\phi_p(B) = 0$ fora do círculo unitário
Condições de invertibilidade	Sempre invertíveis	Raízes de $\theta_q(B) = 0$ fora do círculo unitário	Raízes de $\theta_q(B) = 0$ fora do círculo unitário
FAC	Decaimento exponencial e/ou sinusoidal para zero	Decaimento brusco para zero a partir de $k = q + 1$	Decaimento exponencial e/ou sinusoidal para zero
FACP	Decaimento brusco para zero a partir de $k = p + 1$	Decaimento exponencial e/ou sinusoidal para zero	Decaimento exponencial e/ou sinusoidal para zero

Quadro 2:1 Comparação dos vários tipos de processos ARMA(p,q)

2.5 Processos Estocásticos Não Estacionários e Lineares

A maioria das aplicações de sucessões cronológicas, particularmente as que surgem de áreas económicas, ambientais, e outras, são, por norma, não estacionárias. No que respeita à classe de processos estacionários até à 2ª ordem, um processo pode ser não estacionário por a média, μ_t , e/ou a variância, σ_t^2 , serem funções do tempo e não constantes.

Nas figuras 19 – 21 representam-se as sucessões simuladas (retiradas de *Muller et al.[1990 (pp.78-79)]*) de três tipos de processos não estacionários.



Um processo que é estacionário em média não é necessariamente estacionário em variância e covariância. No entanto, um processo não estacionário em média também não é estacionário em variância e covariância.

Para ultrapassar o problema de não estacionariedade, em média e em variância-covariância, pode muitas vezes recorrer-se a transformações que estabilizem a média e/ou variância e convertam uma sucessão cronológica não estacionária numa sucessão estacionária. Esta técnica permite que se possa realizar a inferência estatística, que não era possível no caso de não estacionariedade.

2.5.1 Não estacionariedade em média e sazonalidade

Tal como se ilustrou nas figuras anteriores existem três tipos clássicos de não estacionariedade. A não estacionariedade em:

- média
- variância
- sazonalidade

A estabilização da média através de filtros lineares está detalhada em *Muller et al.* [1990, (pp.81-86)]. Neste ponto apresentam-se, resumidamente, dois operadores que permitem realizar a estabilização da média, o operador diferença simples e o operador diferença sazonal.

De um modo geral aplica-se o operador diferença simples para **estabilizar a média**.

Operador Diferença. Seja ∇ o operador diferença tal que,

$$\nabla X_t = X_t - X_{t-1} = (1 - B)X_t, \quad (2.49)$$

designado operador diferença simples.

Este operador pode generalizar-se para várias ordens do seguinte modo:

$$\begin{aligned} \nabla^2 X_t &= \nabla(\nabla X_t) = (1 - B)^2 X_t, \\ &= (1 - 2B + B^2)X_t, \\ &= X_t - 2X_{t-1} + X_{t-2}, \end{aligned} \quad (2.50)$$

tal que, qualquer inteiro $d \geq 0$,

$$\nabla^d X_t = (1 - B)^d X_t, \quad (2.51)$$

onde d é a ordem das diferenças ou da diferenciação.

Os processos não estacionários em média que por meio de diferenciação podem transformar-se em processos estacionários, chamam-se, processos não estacionários homogêneos.

O operador diferença Sazonal é aplicado em casos em que há manifesta sazonalidade, ou em casos que se suspeita que exista, e retira a sazonalidade da série em análise.

Operador Diferença Sazonal. De igual modo, define-se o operador diferença sazonal simples, ∇_S , por

$$\nabla_S X_t = X_t - X_{t-S} = (1 - B^S)X_t. \quad (2.52)$$

Para qualquer inteiro $D \geq 0$,

$$\nabla_S^D X_t = (1 - B^S)^D X_t, \quad (2.53)$$

onde D é a ordem das diferenças ou da diferenciação sazonal.

Como caso particular tem-se ∇_S^2 tal que

$$\begin{aligned} \nabla_S^2 X_t &= \nabla_S(\nabla_S X_t) = (1 - B^S)^2 X_t, \\ &= (1 - 2B^S + B^{2S})X_t = X_t - 2X_{t-S} + X_{t-2S}. \end{aligned} \quad (2.54)$$

As condições de aplicação da transformação para **estabilização de variância** estão explicadas e deduzidas em *Muller et al.* (pp.86-91) e *Wei* (pp.77-83)

Nem todos os problemas de não estacionaridade se podem ultrapassar com diferenciação. Muitas sucessões cronológicas são estacionárias em média, no entanto não o são em variância. Para solucionar este problema é necessário aplicar uma transformação adequada de forma a estabilizar a variância.

Geralmente, a estabilização de variância pode ser obtida através de uma transformação paramétrica sugerida por *Box e Cox* [1964],

$$T(X_t) = X_t^{(\lambda)} = \frac{X_t^\lambda - 1}{\lambda}, \text{ se } \lambda \neq 0 \quad (2.55)$$

$$= \ln X_t, \text{ se } \lambda = 0.$$

O parâmetro λ é escolhido no intervalo $[-1,1]$ e os valores mais correntes correspondem às seguintes transformações:

Valores de λ	Transformação
-1.0	$1/X_t$
-0.5	$1/\sqrt{X_t}$
0.0	$\ln X_t$
0.5	$\sqrt{X_t}$
1.0	X_t

A transformação de variância é a primeira que se aplica à série. Ao se aplicar esta transformação, regra geral, também se está a realizar uma redução de escala. Caso os dados transformados apresentem manifesta sazonalidade, aplica-se a diferenciação sazonal. Por último, e caso os dados não sejam estacionários em média, é aplicada a diferenciação simples que estabiliza a média. Note-se que tanto a diferenciação sazonal como a simples poderão ter de ser aplicadas mais do que uma vez.

Caso os dados sejam nulos ou a aplicação de diferenças produza valores negativos, deve aplicar-se uma translação aos dados iniciais para que assumam valores positivos.

A aplicação de uma diferença simples de ordem 1 resulta na perda de uma observação, enquanto a aplicação de uma diferença simples de ordem 2 resulta na perda de 2 observações e assim sucessivamente. A aplicação de uma diferença sazonal de ordem 1, de período 12, ou seja, anual, resulta na perda de 12 observações. Donde resulta que se deve ter cuidado na aplicação dos vários tipos de diferenciação.

No início deste capítulo, quando se introduziu o **R** como ferramenta de trabalho, mencionou-se que o **R** possui livrarias, livrarias essas que também contêm dados. Ao contrário do que se tem vindo a mostrar até agora, não se vai realizar nenhuma geração de processos recorrendo ao **R**, mas sim utilizar dados da livraria “TSA”. Esta livraria possui um comando muito importante que representa graficamente a transformação de *Box-Cox*, indica o coeficiente da transformação e o seu respectivo intervalo de confiança.

O *package* “TSA”, foi escrito por *Kung-Sik Chan*, co-autor do livro “*Time Series Analysis With Applications in R*”, tem uma série (*electricity*) que será utilizada para mostrar a aplicação prática a casos de não estacionariedade.

Como já foi referido, há que instalar o *package* e carrega-lo no *workspace*

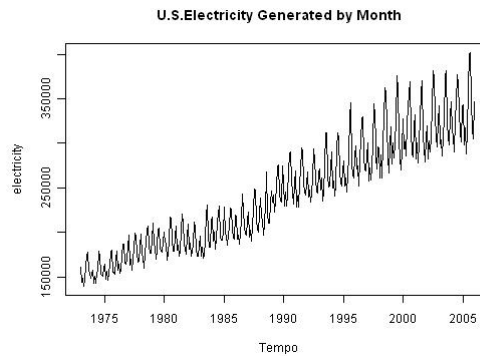
```
install.packages("TSA")      # download do package
library(TSA)                 # carregar package(livraria) no workspace
```

ao que se segue a leitura dos dados.

```
data(electricity)           # ler o dados da serie electricity
```

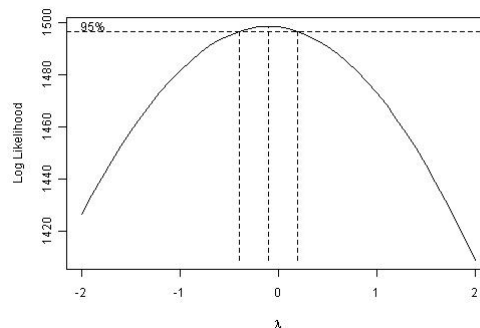
O primeiro passo que se deve fazer na análise de quaisquer dados, é representá-los graficamente. As séries temporais não são excepção.

```
plot(electricity, xlab=" Tempo", main=" U.S.Electricity Generated by Month") # cronograma dos dados
```

Figura 22 : Cronograma de *Electricity*

Como a série é notoriamente não estacionária em variância, tem de se aplicar uma transformação. O comando `BoxCox.ar()` que encontra a melhor transformação também está na livreria “TSA”.

```
electr <-BoxCox.ar (electricity)      # analisar qual a melhor transformação a aplicar
```

Figura 23 : Transformação de Box Cox com intervalos de confiança para λ

O valor zero que corresponde ao logaritmo está dentro do intervalo de confiança para a transformação a aplicar e que se pode verificar por,

```
> (electr$ci)      # intervalo de confiança do valor de lambda da transformação a aplicar
[1] -0.4 0.2
```

Onde -0.4 é o limite inferior e 0.2 é o limite superior do intervalo de confiança para o coeficiente da transformação. Vamos optar por começar a aplicar uma transformação com $\lambda = 0$.

```
plot(log(electricity))      # cronograma da serie depois de aplicada uma estabilização de variância - logaritmica
```

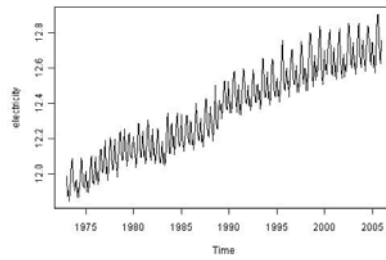


Figura 24 : Cronograma do log de *electricity*

Como se pode verificar pela figura 24, ainda há que operar a estabilização da média, através da diferença do logaritmo da série,

```
plot(diff(log(electricity))) # cronograma com diferenciação depois de aplicado o logaritmo
```

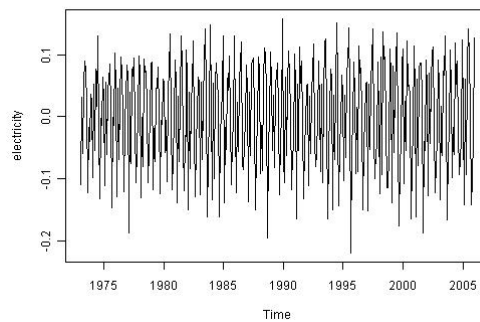


Figura 25 : Cronograma da diferença de log de *electricity*

Aparentemente, pela visualização da figura, a sucessão é estacionária em média. Não se avançará na análise deste conjunto de dados, pois já se mostrou o que se pretendia, a aplicação de transformação para estabilizar variância e aplicação de diferenças.

2.6 Modelação ARIMA de Sucessões Cronológicas

Para alcançar um “bom modelo” *Box e Jenkins* [1970] propuseram, tal como já referido, uma metodologia em três etapas:

Identificação ⇒ estimação ⇒ avaliação do diagnóstico

2.6.1 Identificação do Modelo

A identificação do modelo não é mais do que a utilização de metodologias por forma a transformar a sucessão, de modo a estabilizar a variância, neutralizar a tendência, a eliminar movimentos de carácter estritamente periódico e proceder também à escolha adequada de valores para d , S , D e para p, q . As diversas técnicas utilizadas nesta fase não têm grande precisão e muitas delas dependem de métodos gráficos, nesta fase é de crucial importância a experiência do analista para fazer uma correcta identificação do modelo.

Note-se, que também, existe a possibilidade de não ser necessária a aplicação de qualquer transformação.

2.6.1.1 Passos na identificação do modelo

Passo 1 Consiste na representação gráfica dos dados, o chamado cronograma. Através de uma análise cuidada do cronograma, geralmente fica-se com uma boa ideia do comportamento geral da sucessão. Isto é, se a sucessão tem tendência, sazonalidade, *outliers*, variância não constante, etc.

Resumidamente, as transformações mais utilizadas nesta fase são:

(1) a transformação de *Box-Cox* que permite estabilizar a variância;

(2) o operador diferenciação simples de potência d ,

$$\nabla^d = (1 - B)^d,$$

que neutraliza a tendência. Normalmente, $d \leq 2$ é suficiente para se alcançar esse objectivo.

(3) o operador diferença sazonal de potência D ,

$$\nabla_s^D = (1 - B^S)^D,$$

que permite eliminar movimentos estritamente periódicos da sucessão.

Em geral $D=1$ é suficiente para se atingir este objectivo.

Passo 2 Calcular e examinar as FAC e FACP estimadas da sucessão original para confirmar a necessidade de se proceder à estabilização da série. Algumas regras a ter em consideração são:

- (i) Se a FAC amostral decai lentamente para zero e se a FACP se anula depois do “lag” 1, esse facto indica que a sucessão deve ser diferenciada.
- (ii) Uma não estacionarização em média e uma convergência lenta para zero sobre os “lags” múltiplos de 12 sugere a aplicação de diferenciação sazonal de grau 1
- (iii) De uma forma mais geral, para remover a estacionaridade possivelmente poder-se-á considerar um grau de diferenciação superior. No entanto, tal como já referido, na maior parte dos casos tem-se d igual a 1 ou 2.
- (iv) Alguns autores argumentam que as consequências de diferenciação desnecessária são menos prejudiciais que não se diferenciar um número suficiente de vezes. No entanto há que ter em conta que um número excessivo de diferenciações provoca a perda de dados.

Passo 3 Calcular e examinar as FAC e FACP estimadas, depois de se efectuarem as transformações necessárias de forma a identificar as ordens de p, q

. Para se seleccionar correctamente o modelo deve-se ter um conhecimento profundo das FAC e FACP teóricas dos modelos mais usuais. A identificação é baseada essencialmente na comparação das FAC e FACP estimadas com as dos modelos teóricos.

Na identificação inicial não se deve prestar particular atenção a pequenos pormenores, mas sim ao comportamento geral, uma vez que se pode refinar o modelo numa fase posterior de avaliação do diagnóstico.

Existem três grandes classes de processos estacionários lineares – AR, MA e ARMA – com características bem distintas em termos da FAC e FACP. No

Quadro 2.1 já foram introduzidas de forma resumida, as características das FAC e FACP teóricas dos processos mais comuns.

2.7 Estimação

A etapa que se segue à da identificação é a da estimação dos parâmetros desconhecidos,

$$(\phi_1, \dots, \phi_p), \quad (\theta_1, \dots, \theta_q), \quad \sigma_\varepsilon^2 = v(\varepsilon_t),$$

do modelo ARMA(p,q),

$$X_t^* - \phi_1 X_{t-1}^* - \dots - \phi_p X_{t-p}^* = \varepsilon_t - \theta_1 \varepsilon_{t-1} - \dots - \theta_q \varepsilon_{t-q},$$

onde $X_t^* = X_t - \mu$ e X_t ($t=1,2,\dots,N$) são observações correspondentes à sucessão cronológica por hipótese já estacionarizada.

Uma vez identificado o modelo, as estimativas dos parâmetros são calculadas sem dificuldade pelo **R** ou por qualquer outro *package* estatístico. Os principais aspectos da teoria da estimação para os diferentes modelos, são constituídos por três classes de métodos mais utilizadas: máxima verosimilhança, mínimos quadrados e momentos.

[1] Estimação dos parâmetros nos modelos autoregressivos. Sob a hipótese de normalidade do ruído branco, isto é, admitindo que os (ε_t) formam uma sucessão de variáveis aleatórias i.i.d. $N(0, \sigma_\varepsilon^2)$, o processo AR(p), (X_t) , é gaussiano. Supondo, sem perda de generalidade, que $\mu=0$, pode provar-se (Priestley [1981]) que o logaritmo da verosimilhança é dado, a menos de uma constante aditiva, pela expressão

$$L(\phi_1, \dots, \phi_p) = -\frac{N}{2} \ln \sigma_\varepsilon^2 + \frac{1}{2} \ln |V_p| - \frac{S^*(\phi_1, \dots, \phi_p)}{2\sigma_\varepsilon^2}, \quad (2.56)$$

onde $\sigma_\varepsilon^2 V_p^{-1}$ designa a matriz de autocovariâncias de (X_1, \dots, X_p) e

$$S^*(\phi_1, \dots, \phi_p) = \sum_{i=1}^p \sum_{j=1}^p v_{ij} X_i X_j + \sum_{t=p+1}^N (X_t - \phi_1 X_{t-1} - \dots - \phi_p X_{t-p})^2, \quad (2.57)$$

com v_{ij} elemento genérico de V_p .

Uma vez que a matriz V_p depende dos parâmetros (ϕ_1, \dots, ϕ_p) , as derivadas de $\ln|V_p|$ são difíceis de obter, resultando que os estimadores de máxima verosimilhança são soluções de um sistema de equações não lineares. Contudo, demonstra-se (Box & Jenkins [1970]) que para N suficientemente grande o termo $\ln|V_p|$ é dominado por $S^*(\phi_1, \dots, \phi_p)$ podendo desprezar-se na expressão (2.56). Deste modo obtêm-se aproximações aos estimadores de máxima verosimilhança minimizando $S^*(\phi_1, \dots, \phi_p)$. Uma segunda aproximação pode efectuar-se quando p é pequeno comparado com N e consiste em tomar,

$$L(\phi_1, \dots, \phi_p) \approx \text{constante} - \frac{S(\phi_1, \dots, \phi_p)}{2\sigma_\varepsilon^2}, \quad (2.58)$$

com,

$$S(\phi_1, \dots, \phi_p) = \sum_{t=p+1}^N (X_t - \phi_1 X_{t-1} - \dots - \phi_p X_{t-p})^2, \quad (2.59)$$

saindo os estimadores da minimização de (2.59). Esta segunda aproximação corresponde ao método de máxima verosimilhança condicional em que os valores (X_1, \dots, X_p) se consideram dados e em que, portanto, não se considera a respectiva distribuição conjunta.

Interpretando formalmente o processo AR(p) como um modelo de regressão linear múltipla de X_t sobre X_{t-1}, \dots, X_{t-p} ,

$$X_t = \phi_1 X_{t-1} + \dots + \phi_p X_{t-p} + \varepsilon_t,$$

os estimadores dos parâmetros podem obter-se pelo método dos mínimos quadrados, ou seja, por minimização de (2.59),

$$S(\phi_1, \dots, \phi_p) = \sum_{t=p+1}^N (X_t - \phi_1 X_{t-1} - \dots - \phi_p X_{t-p})^2,$$

procedimento que corresponde à segunda aproximação referida anteriormente.

O método dos momentos pode ser aplicado substituindo nas relações de *Yule-Walker* as autocorrelações teóricas ρ_k pelas estimadas $\hat{\rho}_k$ e resolvendo em relação a (ϕ_1, \dots, ϕ_p) o sistema de equações resultante,

$$\begin{aligned} \hat{\rho}_1 &= \phi_1 + \phi_2 \hat{\rho}_1 + \dots + \phi_p \hat{\rho}_{p-1}, \\ \hat{\rho}_2 &= \phi_1 \hat{\rho}_1 + \phi_2 + \dots + \phi_p \hat{\rho}_{p-2}, \\ &\dots \\ \hat{\rho}_p &= \phi_1 \hat{\rho}_{p-1} + \phi_2 \hat{\rho}_{p-2} + \dots + \phi_p. \end{aligned}$$

Obtêm-se, assim, os chamados estimadores de *Yule-Walker* que, para N suficientemente grande, não diferem substancialmente dos obtidos pelo método dos mínimos quadrados.

Finalmente, a teoria da regressão linear múltipla permite estabelecer o seguinte estimador de σ_ε^2 ,

$$\hat{\sigma}_\varepsilon^2 = \frac{S(\phi_1, \dots, \phi_p)}{N - 2p}, \tag{2.60}$$

onde o denominador é obtido a partir da expressão geral: número de observações utilizadas $(N - p)$ menos o número de parâmetros estimados p .

[2] Estimação dos parâmetros nos modelos médias móveis. O método dos momentos, tal como acabou de descrever-se, aplicado aos processos de médias móveis, permite, através das correspondentes equações de *Yule-Walker*, determinar estimadores para os respectivos parâmetros, no entanto, conduz, em geral, a estimadores não eficientes.

Box e *Jenkins* [1970] desenvolveram uma metodologia de máxima verosimilhança condicional para os processos MA(q),

$$X_t = \varepsilon_t - \theta_1 \varepsilon_{t-1} - \dots - \theta_q \varepsilon_{t-q},$$

que consiste em fixar os valores iniciais,

$$\varepsilon_0 = \varepsilon_{-1} = \varepsilon_{-2} = \dots = \varepsilon_{-(q-1)} = 0.$$

O logaritmo da verosimilhança assume a forma,

$$L(\theta_1, \dots, \theta_q) = \text{constante} - \frac{1}{2\sigma_\varepsilon^2} \sum_{t=1}^N \varepsilon_t^2, \quad (2.61)$$

Ensaando valores de $(\theta_1, \dots, \theta_q)$, compatíveis com a invertibilidade do processo MA(q), podem pesquisar-se numericamente os valores que maximizam a verosimilhança (2.61) ou, equivalentemente,

$$S(\theta_1, \dots, \theta_q) = \sum_{t=1}^N \varepsilon_t^2, \quad (2.62)$$

Assim, com o auxílio de meios informáticos, obtêm-se os chamados estimadores da máxima verosimilhança condicional que são também estimadores dos mínimos quadrados condicionais.

Estimadores de máxima verosimilhança não condicionais podem também estabelecer-se para N suficientemente grande. Estimativas aproximadas resultam da minimização, ainda por cálculo numérico, de,

$$S^{**}(\theta_1, \dots, \theta_q) = \sum_{t=-(q-1)}^N \hat{\varepsilon}_t^2. \quad (2.63)$$

O estimador da máxima verosimilhança condicional de σ_ε^2 é dado por,

$$\sigma_\varepsilon^2 = \frac{1}{N} S(\hat{\theta}_1, \dots, \hat{\theta}_q), \quad (2.64)$$

em que o resultado geral dos modelos lineares usado em (2.61) não pode agora aplicar-se. No entanto, autores como *Jenkins* e *Watts* [1968], por analogia com (2.61), preferem adoptar em alternativa,

$$\tilde{\sigma}_\varepsilon^2 = \frac{1}{N-q} S(\hat{\theta}_1, \dots, \hat{\theta}_q). \quad (2.65)$$

[3] Estimação dos parâmetros nos modelos mistos. Continuando a manter a hipótese anterior sobre o ruído branco, *Box* e *Jenkins* [1970] desenvolveram uma metodologia semelhante à descrita no número anterior fixando como condições iniciais o seguinte conjunto de valores,

$$\varepsilon_1 = \varepsilon_2 = \dots = \varepsilon_p = 0 = \varepsilon_0 = \varepsilon_{-1} = \varepsilon_{-2} = \dots = \varepsilon_{-(q-p-1)} = 0, \quad q > p + 1,$$

o que permite escrever,

$$\begin{aligned}\varepsilon_{p+1} &= X_{p+1} - \phi_1 X_p - \dots - \phi_p X_1, \\ \varepsilon_{p+2} &= X_{p+2} - \phi_1 X_{p+1} - \dots - \phi_p X_2 + \theta_1 \varepsilon_{p+1}, \\ &\dots \\ \varepsilon_{p+q} &= X_{p+q} - \phi_1 X_{p+q-1} - \dots - \phi_p X_q + \theta_1 \varepsilon_{p+q-1} + \dots + \theta_{q-1} \varepsilon_{p+1}, \\ \varepsilon_t &= X_t - \phi_1 X_{t-1} - \dots - \phi_p X_{t-p} + \theta_1 \varepsilon_{t-1} + \dots + \theta_q \varepsilon_{t-q}, \quad p+q < t \leq N.\end{aligned}$$

Calculando a soma de quadrados,

$$S(\phi_1, \dots, \phi_p, \theta_1, \dots, \theta_q) = \sum_{t=p+1}^N \varepsilon_t^2, \quad (2.66)$$

e procedendo por cálculo numérico à respectiva minimização, obtêm-se os estimadores da máxima verosimilhança condicional. Para não alargar o estudo não se referem os métodos de estimação da máxima verosimilhança não condicional.

O estimador do parâmetro σ_ε^2 assume agora a forma,

$$\sigma_\varepsilon^2 = \frac{1}{N - 2p - q} S(\hat{\phi}_1, \dots, \hat{\phi}_p, \hat{\theta}_1, \dots, \hat{\theta}_q), \quad (2.67)$$

onde o denominador resulta de se terem usado apenas $N - p$ observações e estimado $p + q$ parâmetros.

[4] Estimação não Linear. Nos parágrafos anteriores, os métodos da máxima verosimilhança (condicional ou não condicional) conduziram sempre à minimização de uma soma de quadrados de resíduos que, excepção feita para os processos AR, é não linear nos parâmetros e, conseqüentemente, apenas pode fazer-se por técnicas de pesquisa iterativa.

Os métodos de estimação não linear inicializam-se com a proposta de valores iniciais para os parâmetros, eventualmente obtidos através do método dos momentos. Em seguida caminha-se iterativamente em direcção a valores dos parâmetros conducentes a uma menor soma dos quadrados dos resíduos,

comparativamente com a soma associada aos valores iniciais. O processo iterativo deve continuar até se atingir convergência à luz de algum critério previamente estabelecido.

[5] Propriedades assintóticas dos estimadores da máxima verosimilhança. Considere-se um processo ARMA(p,q), estacionário e invertível, $\phi(B)X_t = \theta(B)\varepsilon_t$. Designe-se por,

$$\alpha = (\phi, \theta), \quad \phi = (\phi_1, \dots, \phi_p), \quad \theta = (\theta_1, \dots, \theta_q),$$

o vector dos parâmetros e por

$$\hat{\alpha} = (\hat{\phi}, \hat{\theta})$$

o correspondente vector dos estimadores da máxima verosimilhança. Admitindo ruído branco gaussiano $N(0, \sigma_\varepsilon^2)$, Brockwell e Davis [1987] provam o seguinte resultados assintótico¹,

$$\sqrt{N}(\hat{\alpha} - \alpha) \overset{\cdot}{\sim} N_{p+q}(0, V(\alpha)), \quad (2.68)$$

onde para $p \geq 1$ e $q \geq 1$ a matriz de covariâncias assintótica vem dada por,

$$V(\alpha) = \sigma_\varepsilon^2 \begin{bmatrix} E\{U_t, U_t'\} & E\{U_t, V_t'\} \\ E\{V_t, U_t'\} & E\{V_t, V_t'\} \end{bmatrix}^{-1}, \quad (2.69)$$

em que $U_t = (U_t, \dots, U_{t+1-p})'$, $V_t = (V_t, \dots, V_{t+1-q})'$ e $\{U_t\}$, $\{V_t\}$ são os processos autoregressivos,

$$\phi(B)U_t = \varepsilon_t, \quad \theta(B)V_t = \varepsilon_t.$$

Para $p = 0$,

$$V(\alpha) = \sigma_\varepsilon^2 [E\{V_t, V_t'\}]^{-1},$$

e para $q = 0$,

$$V(\alpha) = \sigma_\varepsilon^2 [E\{U_t, U_t'\}]^{-1}.$$

¹ $N_r(\mu, \Sigma)$ representa a distribuição Gaussiana r-dimensional com vector de médias $\mu = (\mu_1, \mu_2, \dots, \mu_r)$ e matriz de variâncias-covariâncias Σ

Brockwell e Davis [1987] referem que os estimadores dos mínimos quadrados possuem as mesmas propriedades assintóticas dos da máxima verosimilhança, sendo conseqüentemente válidos para aqueles estimadores os resultados anteriormente estabelecidos.

Tendo em conta a distribuição assintótica dos estimadores e designando por $v_{ii}^2(\hat{\alpha})$ o i -ésimo elemento da diagonal principal da matriz $V(\hat{\alpha})$ e por $\alpha = (\phi, \theta)$ o parâmetro genérico de um processo ARMA(p, q), pode concluir-se que,

$$\frac{\sqrt{N}(\hat{\alpha}_i - \alpha_i)}{v_{ii}(\hat{\alpha})} \sim t_{(N-p-q)}, \quad (2.70)$$

onde $t_{(N-p-q)}$ representa a distribuição de Student com $(N-p-q)$ graus de liberdade.

Nos estudos realizados sobre o **R** concluiu-se que existem duas funções de estimação de parâmetros, uma para processos autoregressivos, *ar()*, e outra para processos ARMA, *arima()*.

Simule-se então o processo AR(2), $X_t = 0.2 \times X_{t-1} + 0.6 \times X_{t-2} + \varepsilon_t$

```
set.seed(1)           # fixa a semente
PHI1 <- 0.2           # valor de phi1
PHI2 <- 0.6           # valor de phi2
x <- w <- rnorm(250)  # inicialização da série e geração do ruído branco gaussiano
                      # com 250 observações

for (t in 3:250) x[t] <- PHI1 * x[t - 1] + PHI2 * x[t - 2] + w[t]      # ciclo que constrói a série
```

A função *ar()* permite estimar os parâmetros através de 4 métodos, *Yule-Walker* (*yw*), *Burg*, *Maximum-Likelihood Estimation* (*mle*) e *Ordinary Least Squares* (*ols*) e que se podem realizar da seguinte forma,

```
(x.ar.yw <- ar(x, method = "yw"))      # Estimação pelo método de Yule - Walker
Call:
ar(x = x, method = "yw")
Coefficients:
 1          2
0.2017     0.5916
```

```
Order selected 2 sigma^2 estimated as 0.9353
```

```
(x.ar.burg <- ar(x, method="burg"))          # Estimação pelo método de Burg
```

```
Call:
```

```
ar(x = x, method = "burg")
```

```
Coefficients:
```

```
  1          2
0.2014      0.5924
```

```
Order selected 2 sigma^2 estimated as 0.9225
```

```
(x.ar.mle <- ar(x, method="mle"))          # Estimação pelo método de Máxima Verosimilhança
```

```
Call:
```

```
ar(x = x, method = "mle")
```

```
Coefficients:
```

```
  1          2
0.2005      0.5885
```

```
Order selected 2 sigma^2 estimated as 0.9225
```

```
(x.ar.mle <- ar(x, method="ols"))          # Estimação pelo método de Mínimos Quadrados
```

```
Call:
```

```
ar(x = x, method = "ols")
```

```
Coefficients:
```

```
  1          2
0.2019      0.5922
```

```
Intercept: 9.394e-05 (0.06118)
```

```
Order selected 2 sigma^2 estimated as 0.9283
```

Ou por

```
(ar.yw <- ar(x))
```

```
(ar.burg <- ar(x))
```

```
(ar.mle <- ar(x))
```

```
(ar.mle <- ar(x))
```

Que é uma forma mais fácil de escrever, os resultados são exactamente os mesmos.

Simule-se um processo ARMA(1,1), $X_t = 0.2 \times X_{t-1} + \varepsilon_t - 0.6 \times \varepsilon_{t-1}$

```
set.seed(1)          # fixa a semente
```

```
PHI<- 0.2           # valor de phi
```



```

THETA <- 0.6          # valor de tetha
x <- w <- norm(250)   # inicialização da série e geração do ruído branco gaussiano
                    # com 250 observações
for (t in 3:250) x[t] <- PHI * x[t - 1] + w[t] + THETA * w[t-1]      # ciclo que constrói a série

```

A estimação com recurso à função *arima()* permite utilizar 3 métodos de estimação, *Conditional Sum of Squares – Maximum Likelihood* (CSS-ML), *Maximum Likelihood* (ML) e *Conditional Sum of Squares* (CSS),

```

(x.arma.css.ml <- arima(x,order=c(1,0,1), method ="CSS-ML", include.mean=FALSE)) # soma condicional
                                                    # de quadrados
                                                    # por máxima
                                                    # verosimilhança

Call:
arima(x = x, order = c(1, 0, 1), include.mean = FALSE, method = "CSS-ML")
Coefficients:
    ar1      ma1
 0.1071  0.6544
s.e. 0.0990  0.0830
sigma^2 estimated as 0.9217: log likelihood = -344.9, aic = 695.8

(x.arma.ml <- arima(x,order=c(1,0,1), method ="ML", include.mean=FALSE))      # maxima verosimilhança

Call:
arima(x = x, order = c(1, 0, 1), include.mean = FALSE, method = "ML")

Coefficients:
    ar1      ma1
 0.1071  0.6544
s.e. 0.0990  0.0830

sigma^2 estimated as 0.9217: log likelihood = -344.9, aic = 695.8

(x.arma.css <- arima(x,order=c(1,0,1), method ="CSS", include.mean=FALSE)) # soma condicional de
quadrados

Call:
arima(x = x, order = c(1, 0, 1), include.mean = FALSE, method = "CSS")

Coefficients:
    ar1      ma1
 0.1025  0.6594
s.e. 0.0993  0.0825

sigma^2 estimated as 0.9228: part log likelihood = -344.7

```

2.8 – Avaliação do Diagnóstico

Identificado um modelo e estimados os respectivos parâmetros passa-se à fase de avaliação do diagnóstico. Nesta fase começa-se por analisar a qualidade estatística do modelo estimado. Um modelo que não satisfaça os seus pressupostos deve ser rejeitado, e então deve repetir-se o ciclo,

Identificação \Rightarrow estimação \Rightarrow avaliação do diagnóstico,

até se encontrar um modelo satisfatório para descrever a sucessão em causa. No caso de existirem diversos modelos de boa qualidade à luz dos seus pressupostos, põe-se o problema da escolha do melhor modelo. A resolução desta situação pode ser efectuada através de alguns critérios, de entre os quais se salienta o critério AIC.

2.8.1 Avaliação da qualidade estatística do(s) modelo(s) ajustado(s)

Supondo que se ajustou um modelo ARMA(p,q) a uma dada série temporal com parâmetros, $(\phi_1, \phi_2, \dots, \phi_p, \theta_1, \theta_2, \dots, \theta_q)$. Por simplicidade, designe-se por β_i cada um destes parâmetros.

[1] Significância estatística dos parâmetros. Tendo em conta o princípio da parcimónia, deve-se verificar se um parâmetro é significativamente diferente de zero. Os parâmetros estatisticamente nulos deverão ser eliminados.

Para que um dado parâmetro seja significativo para o modelo, ou não, realiza-se o seguinte teste de hipóteses:

$$H_0 : \beta_i = 0 \quad \text{vs} \quad H_1 : \beta_i \neq 0,$$

Prova-se que a estatística de teste é

$$T = \sqrt{N} \frac{\hat{\beta}_i}{\sqrt{\hat{\text{Var}}[\hat{\beta}_i]}} \sim t_{N-p-q}$$

Como regra de decisão rejeita-se H_0 , para um nível de significância α , sempre que $|T| > t_{N-p-q, \alpha/2}$, com $t_{N-p-q, \alpha/2}$ o quantil de probabilidade $1 - \frac{\alpha}{2}$ da distribuição t-Student, com N-p-q graus de liberdade.

A não rejeição da hipótese nula, para um nível de significância α , implica a eliminação do parâmetro em teste do modelo.

Para mostrar como se testa a significância dos parâmetros simula-se um processo AR(1).

```
set.seed(1)                # fixa a semente
PHI<- 0.2                 # valor de phi1
x <- w <- rnorm(1000)     # inicialização da série e geração do ruído branco
                           # gaussiano com 1000 observações
for (t in 2:1000) x[t] <-PHI * x[t - 1] + w[t]      # ciclo que constrói a série
```

Para realizar a estimação dos parâmetros utiliza-se o comando *arima()* e para testar a significância o comando *confint()*, que integra a livreria *tseries*,

```
x.ar <- arima(x,order=c(3,0,0), include.mean=TRUE) # estimação dos parâmetros do modelo
                                                # com 3 coeficientes autoregressivos
                                                # e com constante
x.ar                                           # coeficientes estimados
Call:
arima(x = x, order = c(3, 0, 0), include.mean = TRUE)

Coefficients:
      ar1      ar2      ar3      intercept
      0.1581   -0.0276   -0.0335   -0.0143
s.e. 0.0316    0.0320    0.0316    0.0361

sigma^2 estimated as 1.065: log likelihood = -1450.6, aic = 2911.19
```

No exemplo de análise de caso prático mais adiante está explicado de forma mais completa o comando *arima()*. Ao utilizar o comando *arima()* especificou-se

que a ordem dos coeficientes autoregressivos seria exactamente 3, (ϕ_1, ϕ_2, ϕ_3) , e que incluía constante. Para aferir se são mesmo necessários os 3 coeficientes autoregressivos e a constante, utiliza-se então o comando *confint()*, mas antes há que carregar a livreria *tseries*

```
install.packages("tseries")          # download da livreria
library(tseries)                    # carrega a livreria no workspace
confint(x.ar)                        # devolve o intervalo a 95% dos parâmetros estimados
      2.5 %          97.5 %          # limites do intervalo de confiança
ar1    0.09616415    0.22003401
ar2   -0.09031343    0.03512212
ar3   -0.09549552    0.02846150
intercept -0.08516880  0.05651832
```

Como regra prática: se o zero está incluído no intervalo de confiança de um parâmetro, então esse parâmetro elimina-se do modelo, pela rejeição da hipótese nula. Que é o que acontece com ϕ_2 , ϕ_3 e a constante. Voltam-se então a estimar os parâmetros do modelo,

```
(x.ar1 <- arima(x,order=c(1,0,0), include.mean=FALSE));(confint(x.ar1))
Call:
arima(x = x, order = c(1, 0, 0), include.mean = FALSE)
Coefficients:
      ar1
      0.1543
s.e. 0.0312
sigma^2 estimated as 1.068: log likelihood = -1451.77, aic = 2907.54
      2.5 %          97.5 %
ar1    0.09310185    0.2155717
```

Estima-se assim um modelo com um coeficiente autoregressivo.

[2] Estacionaridade e invertibilidade. Se os parâmetros estimados se encontram na vizinhança das regiões de não estacionaridade ou de não invertibilidade o modelo correspondente deve ser rejeitado. Por exemplo, num modelo AR(1) de parâmetro estimado $\hat{\phi}_1 = 0.92$ poderá ser rejeitado, pois o parâmetro está muito próximo da região de não estacionariedade e o modelo torna-se instável.

Nota:

Um factor AR não invertível normalmente indica falta de diferenciação, enquanto um factor MA não invertível indica que possivelmente há excesso de diferenciação. Um factor MA pode também indicar presença de um factor determinístico.

[3] Redundância.

Quando o modelo,

$$\phi(B)X_t = \theta(B)\varepsilon_t,$$

é idêntico ao modelo

$$(1 - \alpha B)\phi(B)X_t = (1 - \alpha B)\theta(B)\varepsilon_t,$$

e diz-se que o factor $(1 - \alpha B)$ é redundante.

Exemplos:

(i) Um exemplo de redundância muito comum encontra-se disfarçada de

processo ARMA(2,1),

$$(1 - 1.3B + 0.4B^2)X_t = (1 - 0.5B)\varepsilon_t,$$

é equivalente a um processo AR(1),

$$(1 - 0.8B)X_t = \varepsilon_t,$$

Uma vez que $(1 - 1.3B + 0.4B^2)X_t = (1 - 0.5B)(1 - 0.8B)X_t$.

(ii) Uma situação prática do tipo,

$$(1 - 0.54B)X_t = (1 - 0.5B)\varepsilon_t,$$

dá lugar a um factor quase redundante.

Um modelo que apresente factores redundantes ou quase redundantes, em geral, conduz a problemas no processo de estimação dos seus parâmetros. Donde a redundância deverá ser eliminada.

[4] Correlação. Encontrando-se parâmetros estimados fortemente correlacionados, considera-se o modelo estimado de má qualidade. Será um modelo pouco estável. De facto, uma ligeira modificação num dos valores

estimados implica uma correspondente variação nos parâmetros que lhe estão correlacionados. Como regra prática, devemos rejeitar um modelo desde que a correlação entre dois parâmetros seja, em módulo, superior a 0.7 [Muller p.125 (1990)].

2.8.2 Análise dos resíduos – Aplicação de testes estatísticos.

A qualidade do ajustamento de um modelo estimado pode medir-se através da análise dos correspondentes resíduos. Considere-se o processo ARMA(p,q),

$$\phi(B)X_t = \mu + \theta(B)\varepsilon_t,$$

e sejam $\hat{\beta} = (\hat{\mu}, \hat{\phi}_1, \hat{\phi}_2, \dots, \hat{\phi}_p, \hat{\theta}_1, \hat{\theta}_2, \dots, \hat{\theta}_q)$ o vector estimador dos parâmetros do processo.

Os resíduos são estimados por,

$$\hat{\varepsilon}_t = \hat{\theta}^{-1}(B)\hat{\phi}(B)(X_t - \hat{\mu})$$

onde $\hat{\phi}(B) = (1 - \hat{\phi}_1 B - \dots - \hat{\phi}_p B^p)$ é o estimador do polinómio autoregressivo e $\hat{\theta}(B) = (1 - \hat{\theta}_1 B - \dots - \hat{\theta}_q B^q)$ é o estimador do polinómio média móveis.

Num modelo bem ajustado a uma sucessão cronológica os correspondentes resíduos, de acordo com as condições dos modelos teóricos, deverão verificar as seguintes propriedades:

- (i) $\{\hat{\varepsilon}_t\}$ deverá ter um comportamento análogo a um ruído branco,
- (ii) Não se rejeitar a hipótese de $\{\hat{\varepsilon}_t\}$, constituírem uma sequência de v.a.'s aleatórias gaussianas.

Para verificar (ii) efectua-se o teste de normalidade de *Kolmogorov-Smirnov* com correcção de *lilliefors* (livraria *nortest*), caso a amostra tenha dimensão inferior a 50 aplica-se o teste de *Shapiro – Wilk*,

Para verificar (i) realizam-se testes estatísticos sobre a FAC e FACP. O ruído branco tem como característica a sua variância ser constante e a FAC e FACP não serem nulas em todos os *lags* não nulos.

Retomando os dados simulados para mostrar a significância dos parâmetros, os resíduos resultantes da estimação ficam guardados no objecto `x.ar1$res`.

Podem aplicar-se todas as funções com vista à obtenção de estatísticas descritivas, testes de normalidade e representação gráfica da seguinte forma,

```
summary(x.ar1$res[-1])      # estatísticas descritivas
plot(x.ar1$res[-1], type = "l", xlab= "Tempo", main=" Resíduos") # cronograma
acf(x.ar1$res[-1], main= "Função de Autocorrelação")           # FAC
pacf(x.ar1$res[-1],main= "Função de Autocorrelação Parcial")  # FACP
```

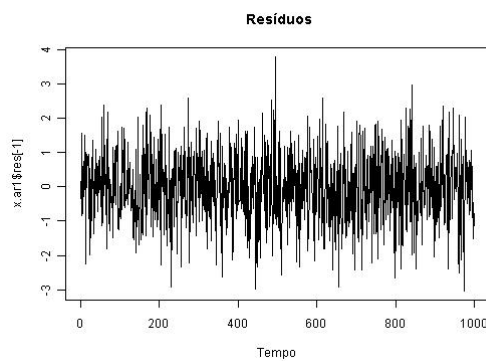


Figura 26: Cronograma dos resíduos

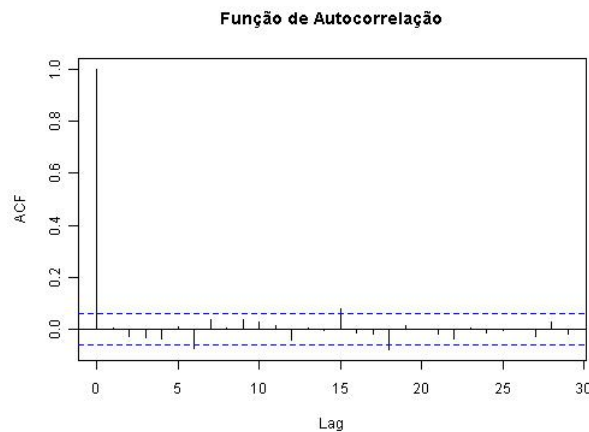


Figura 27: FAC dos resíduos

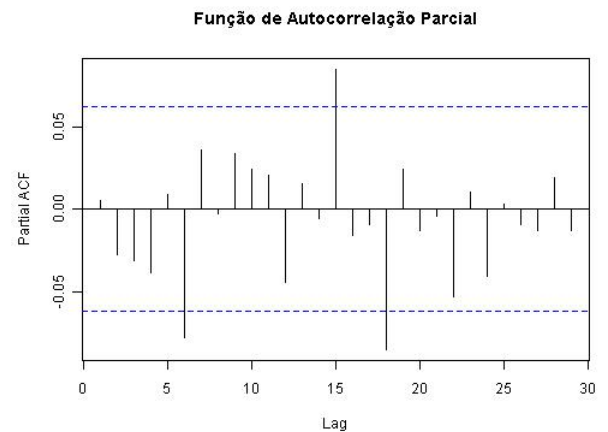


Figura 28: FACP dos resíduos

Tanto o cronograma como a FAC e FACP têm comportamentos que nos levam a pensar tratar-se de ruído branco, note-se que na FACP temos 3 lags fora das bandas de confiança não evidencia que não estamos perante ruído branco.

Depois da análise visual, passa-se à aplicação de testes,

```
install.packages("nortest") # download da livraria nortest
library(nortest)           # carregar a livraria no workspace
```

```

lillie.test(x.ar1$res[-1])          # teste de Kolmogorov-Smirnov com correcção de Lilliefors
  Lilliefors (Kolmogorov-Smirnov) normality test
data: x.ar1$res[-1]
D = 0.016, p-value = 0.7718

shapiro.test(x.ar1$res[-1])        # utilizado em amostras inferiores a 50
  Shapiro-Wilk normality test

data: x.ar1$res[-1]
W = 0.9988, p-value = 0.759

```

O teste de *Shapiro-Wilk* está representado só a título exemplificativo, pois não se aplica neste caso, dado que o número de resíduos é superior a 50 observações. O teste de *Kolmogorov-Smirnov* mostra que os resíduos resultantes do ajustamento efectuado constituem ruído branco, pois têm um *p-value* igual a 0,7718.

2.8.3 Testes estatísticos sobre a FAC e FACP.

[1] Teste de *Bartlett*. O teste de *Bartlett* testa a nulidade das FAC dos resíduos estimados, em cada *lag* k . O teste é,

$$H_0 : \rho_\varepsilon(k) = 0 \quad \text{vs} \quad H_1 : \rho_\varepsilon(k) \neq 0, \text{ para cada } k \in \mathbb{N}.$$

Prova-se que a estatística de teste é a seguinte

$$\sqrt{N} \hat{\rho}_\varepsilon(k) \sim^a N(0,1).$$

Assim, define-se, aproximadamente, a região crítica, ao nível de 5%,

$$|\hat{\rho}_\varepsilon(k)| \geq \frac{2}{\sqrt{N}}.$$

Nota: o **R** não realiza este teste.

[2] Teste de *Jenkins e Daniels* [1956]. Estes autores demonstraram, que a FACP residual estimada tem, sob a hipótese H_0 , distribuição assintótica normal

com média igual a zero e desvio padrão aproximadamente igual a $\frac{1}{\sqrt{N}}$.

A região crítica é análoga à do teste de *Bartlett*.

O teste de hipóteses é o seguinte,

$$H_0 : \phi_\varepsilon(k, k) = 0 \quad \text{vs} \quad H_1 : \phi_\varepsilon(k, k) \neq 0 \quad \text{para cada } k \in \mathbb{N}.$$

Prova-se que a estatística de teste é a seguinte,

$$\sqrt{N} \hat{\phi}_\varepsilon(k, k) \sim^a N(0, 1),$$

sendo igualmente a região crítica análoga à do teste de *Bartlett*.

Nota: o **R** também não realiza este teste.

[3] Teste de Box e Pierce. Este teste permite avaliar o conjunto de todos os valores de $\hat{\rho}_k$, $k=1, \dots, M$ das FAC de um ruído branco, isto é, testa a nulidade de uma sequência de M valores iniciais da FAC.

O teste é o seguinte,

$$H_0 : \rho_\varepsilon(1) = \rho_\varepsilon(2) = \dots = \rho_\varepsilon(M) = 0 \quad \text{vs} \quad H_1 : \exists \rho_\varepsilon(k) \neq 0, \quad \text{para } k = 1, 2, \dots, M$$

onde a estatística de teste é,

$$Q = N \sum_{k=1}^M [\hat{\rho}_\varepsilon(i)^2] \sim^a \chi_{M-p-q, \alpha/2},$$

sendo $\chi_{M-p-q, \alpha/2}$ o quantil de probabilidade de $1 - \frac{\alpha}{2}$ da distribuição Qui-

Quadrado, com $M - p - q$ graus de liberdade.

O teste aplica-se da seguinte forma,

```
Box.test(x.ar1$res[-1], lag = 1, type = "Box-Pierce")      # aplica o teste de Box-Pierce aos resíduos da variável x.ar1
Box-Pierce test
data: x.ar1$res[-1]
X-squared = 0.0262, df = 1, p-value = 0.8714
```

A aplicação deste teste não evidencia que se rejeite a não correlação dos resíduos

[4] Teste de Ljung-Box [1978]. Este teste é uma versão melhorada do teste anterior, e utiliza a estatística de teste,

$$\hat{Q} = N(N-2) \sum_{k=1}^M \frac{1}{N-k} \left[\hat{\rho}_\varepsilon(k) \right]^2 \sim^a \chi_{M-p-q, \alpha/2}^2, \quad (6.22)$$

que converge, mas mais rapidamente, para uma distribuição χ^2 do que a anterior. O teste é o análogo ao anterior.

```
Box.test(x.ar1$res[-1], lag = 1, type = "Ljung-Box") # aplica o teste de Ljung-Box aos resíduos da variável x.ar1
x-Ljung test

data: x.ar1$res[-1]
X-squared = 0.0263, df = 1, p-value = 0.8712
```

A aplicação deste teste não evidencia que se rejeite a normalidade dos resíduos

[5] Teste de Kendal e Stuart. Estes autores demonstraram, para uma sucessão de N variáveis aleatórias i.i.d., o seguinte resultado assintótico,

$$\hat{\rho}_k \dot{\sim} N\left(-\frac{1}{\sqrt{N}}, \frac{1}{\sqrt{N}}\right). \quad (6.21)$$

Esta propriedade genérica pode utilizar-se para testar a nulidade da FAC residual para cada k , $H_0: \rho_k(\varepsilon) = 0$, definindo para $\rho_k(\varepsilon)$ os limites críticos ao nível de significância de 5% (aproximadamente),

$$-\frac{1}{N} \pm \frac{2}{\sqrt{N}}.$$

Se 95% dos valores de $\hat{\rho}_k$ estiverem compreendidos nesta região serão considerados estatisticamente nulos, não se rejeitando a hipótese de que os resíduos têm comportamento de ruído branco.

Note-se, igualmente que o **R** não realiza este teste.

2.8.4 Critérios de Selecção de Modelos

Na análise de sucessões cronológicas podem existir vários modelos, que se adequam na representação dos conjuntos de dados e que passaram nos testes

anteriores. Assim, foram desenvolvidos vários critérios de selecção de modo a escolher o modelo a adoptar.

Para um dado conjunto de dados, e quando existem vários modelos adequados, o critério de selecção é normalmente baseado nas estatísticas sumárias dos resíduos calculados a partir do modelo ajustado, ou a partir dos erros de previsão produzidos por esse mesmo modelo.

[1] Critério Akaike. Assume-se que um modelo com m parâmetros está ajustado aos dados. Para avaliar a qualidade de ajustamento do modelo, Akaike [1973,1974] introduziu um critério de informação, AIC, (*Akaike's Information Criterion*) que é definido por,

$$AIC(m) = -2 \ln[\text{máxima verosimilhança}] + 2m, \quad (6.24)$$

em que m é dado pelo valor que minimiza o valor de $AIC(m)$.

Através de cálculos e fazendo uso das propriedades e características dos modelos ARMA, conclui-se que

$$AIC(m) = N \ln \sigma_e^2 + 2m, \quad (6.26)$$

A ordem óptima do modelo é escolhida pelo valor de m , que é função de p e q , de forma que $AIC(m)$ seja mínimo, ver Muller (p.135), Wei (p.153).

No caso de se estimar com a função *arima()*, o AIC é um objecto da variável onde se guarda a estimação, neste caso da variável *x.ar1*.

x.ar\$aic	# AIC do modelo que se rejeitou por excesso de parâmetros
[1] 2911.19	
x.ar1\$aic	# AIC do modelo
[1] 2907.536	

Mesmo tendo rejeitado o primeiro modelo testado por ter excesso de parâmetros, optou-se por mostrar o AIC, pois à luz deste critério o segundo modelo testado apresenta um valor mais baixo de AIC. Caso se tivesse falhado nos testes de significância de parâmetros, recorrendo ao AIC ficava-se a saber que o melhor modelo seria o que só tem um coeficiente autoregressivo.

[2] Critério BIC. Mais recentemente, *Akaike* (1978,1979) desenvolveu uma extensão bayesiana do critério anterior, definindo para um modelo com m parâmetros e N observações,

$$BIC(m) = N \ln \hat{\sigma}_\varepsilon^2 - (N - m) \ln \left(1 - \frac{m}{N}\right) + m \ln N + m \ln \left[\frac{1}{m} \left(\frac{\hat{\sigma}_x^2}{\hat{\sigma}_\varepsilon^2} - 1 \right) \right], \quad (6.27)$$

onde

$\hat{\sigma}_\varepsilon^2$ é o estimador da máxima verosimilhança de σ_ε^2

$\hat{\sigma}_x^2$ é o estimador da máxima verosimilhança σ_x^2 .

Quando m é pequeno relativamente a N pode utilizar-se a aproximação,

$$\left[-(N - m) \ln \left(1 - \frac{m}{N}\right) \right] \approx m,$$

obtendo a expressão ligeiramente mais simples,

$$BIC(m) = N \ln \sigma_\varepsilon^2 + m(1 + \ln N) + m \ln \left[\frac{1}{m} \left(\frac{\hat{\sigma}_x^2}{\hat{\sigma}_\varepsilon^2} - 1 \right) \right], \quad (6.28)$$

[3] Critério SBC de Schwartz's. *Schwartz* [1978] sugeriu o seguinte critério bayesiano para a selecção de modelos, que tem sido designado SBC (*Schwartz's Bayesian Criterion*),

$$SBC(m) = N \ln \hat{\sigma}_\varepsilon^2 + m \ln m, \quad (6.29)$$

onde $\hat{\sigma}_\varepsilon^2$, m e N continuam a ter a mesma designação que nos critérios anteriores.

2.9 Previsão

Um dos principais objectivos da análise de sucessões cronológicas é a previsão estatística, isto é, a previsão de comportamentos futuros de uma sucessão tendo em conta o seu conhecimento até ao instante t .

Considere-se uma sucessão cronológica com observações disponíveis até ao instante t , $\{X_t, X_{t-1}, X_{t-2}, \dots\}$, e se pretende com base nestas observações

prever o valor no momento $t + m$, ($m > 0$), X_{t+m} . Designando por $X_t(m)$ o preditor de X_{t+m} , tem-se,

$$X_t(m) = f(X_t, X_{t-1}, X_{t-2}, \dots), \tag{2.71}$$

ou seja, considera-se o preditor função dos valores observados da sucessão cronológica. O instante t é a origem de previsão e m o horizonte de previsão.

A escolha da função de previsão (2.71) vai efectuar-se de acordo com o critério de minimização do erro quadrático médio entre X_{t+m} e $X_t(m)$. Ou seja,

$$E\left[\left(X_{t+m} - X_t(m)\right)^2\right], \tag{2.72}$$

Prova-se que o melhor preditor de X_{t+m} em termos de erro quadrático médio é a esperança condicional,

$$X_t(m) = E\left[X_{t+m} \mid X_t, X_{t-1}, X_{t-2}, \dots\right]. \tag{2.73}$$

Na prática, em geral, assume-se, uma das seguintes situações:

- (i) $X_{t+m}, X_t, X_{t-1}, \dots$ possuem distribuição conjunta Normal. Neste caso a esperança condicional torna-se de fácil cálculo e o melhor preditor $X_t(m)$ é função linear de $X_t, X_{t-1}, X_{t-2}, \dots$.
- (ii) Admite-se que o preditor tem a forma de funções lineares de $X_t, X_{t-1}, X_{t-2}, \dots$,

$$X_t(m) = \alpha_0 + \alpha_1 X_t + \alpha_2 X_{t-1} + \dots$$

e procuram-se os valores de $\alpha_0, \alpha_1, \alpha_2, \dots$ que minimizam

$$E\left[X_{t+m} - f(X_t, X_{t-1}, X_{t-2}, \dots)\right]^2.$$

2.9.1 Previsão de Processos Não Estacionários

Considere-se um processo não estacionário ARIMA(p,d,q),

$$\phi(B)(1 - B)^d X_t = \theta(B)\varepsilon_t, \tag{2.74}$$

onde as raízes do polinómio $\phi(B)$ são em módulo estritamente superiores à unidade. Este processo pode escrever-se na forma alternativa,

$$(1 - \psi_1^* B - \dots - \psi_{p+d}^* B^{p+d}) X_t = \theta(B) \varepsilon_t, \quad (2.75)$$

onde,

$$(1 - \psi_1^* B - \dots - \psi_{p+d}^* B^{p+d}) = \phi(B)(1 - B)^d,$$

representa o operador autoregressivo não estacionário de grau $p+d$. Utilizando esta notação pode-se reescrever (2.74),

$$X_t = \psi_1^* X_{t-1} + \dots + \psi_{p+d}^* X_{t-p-d} + \varepsilon_t - \theta_1 \varepsilon_{t-1} - \dots - \theta_q \varepsilon_{t-q}.$$

Então, para todo o $m \geq 1$,

$$X_{t+m} = \psi_1^* X_{t+m-1} + \dots + \psi_{p+d}^* X_{t+m-p-d} + \varepsilon_{t+m} - \theta_1 \varepsilon_{t+m-1} - \dots - \theta_q \varepsilon_{t+m-q},$$

Nestas condições, o preditor é,

$$X_t(1) = \psi_1^* X_t + \dots + \psi_{p+d}^* X_{t+1-p-d} - \theta_1 \varepsilon_t - \dots - \theta_q \varepsilon_{t+1-q}. \quad (2.76)$$

Tendo por erro de previsão,

$$e_t(1) = X_{t+1} - X_t(1) = \varepsilon_{t+1},$$

Tanto o preditor como o erro de previsão calculam-se recursivamente.

2.9.2 Actualização das Previsões

A representação em médias móveis permite estabelecer o preditor de X_{t+m+1} como,

$$X_{t+1}(m) = \sum_{j=0}^{\infty} \psi_{m+1+j} \varepsilon_{t-j}.$$

Quando a observação X_{t+1} estiver disponível pode obter-se uma melhor previsão de X_{t+m+1} através de $X_t(m+1)$ do seguinte modo,

$$X_{t+1}(m) - X_t(m+1) = \sum_{j=0}^{\infty} \psi_{m+j} \varepsilon_{t+1-j} - \sum_{j=0}^{\infty} \psi_{m+1+j} \varepsilon_{t-j} = \psi_m \varepsilon_{t+1},$$

onde,

$$X_{t+1}(m) = \sum_{j=0}^{\infty} \psi_{m+j} \varepsilon_{t+1-j},$$

donde,

$$X_{t+1}(m) = X_t(m+1) + \psi_m \varepsilon_{t+1}, \quad (2.77)$$

com ε_{t+1} valor observado uma vez que,

$$\varepsilon_{t+1} = X_{t+1} - X_t(1) = e_{t+1}(1).$$

Obtém-se assim uma actualização ou reavaliação da previsão inicial para o horizonte $m+1$ adicionando a essa previsão um valor proporcional ao erro de previsão a um passo, $e_{t+1}(1)$, em que o coeficiente de proporcionalidade é ψ_m .

2.9.3 Intervalos de Confiança para Previsões

O erro de previsão em m passos, nos modelos ARMA estacionários, onde (ε_t) é ruído branco, é dado por,

$$e_t(m) = X_{t+m} - X_t(m) = \sum_{j=0}^{m-1} \psi_j \varepsilon_{t+m-j},$$

e, sabendo que,

$$E[e_t(m)] = 0, \quad V[e_t(m)] = \sigma_\varepsilon^2 \sum_{j=0}^{m-1} \psi_j^2$$

e admitindo que os (ε_t) têm distribuição Normal, pode concluir-se ser,

$$e_t(m) \sim N\left(0, \sigma_\varepsilon^2 \sum_{j=0}^{m-1} \psi_j^2\right). \quad (2.78)$$

Donde se pode construir um intervalo de confiança a 95% para a previsão do valor futuro, X_{t+m} , do modo usual,

$$X_t(m) \pm 1.96\sigma_\varepsilon \sqrt{\sum_{j=0}^{m-1} \psi_j^2}. \quad (2.79)$$

Repare-se que os limites inferior e superior do intervalo tendem a estabilizar-se à medida que o horizonte de previsão for aumentando.

Utilizando a mesma série simulada em 2.8 (p.67), utiliza-se código de **R** para mostrar como realizar alguns passos de previsão, a título exemplificativo.

Convém representar graficamente a série original e o resultado do ajustamento encontrado,

```

plot(x, type = "l", col = 1, main = "Cronograma")           # cronograma série gerada
z <- numeric()                                           # definir a variável z para construir ciclo
for (t in 2:1000) z[t] <- x.ar1$ar * x[t - 1]             # série de valores ajustados
points(z, col= 2, type = "o")                            # adiciona a série ajustada ao cronograma como pontos
legend("topright", legend = c(" Série Simulada", "Série Ajustada"), col= c(1, 2), lty=c(1,1)) #legenda

```

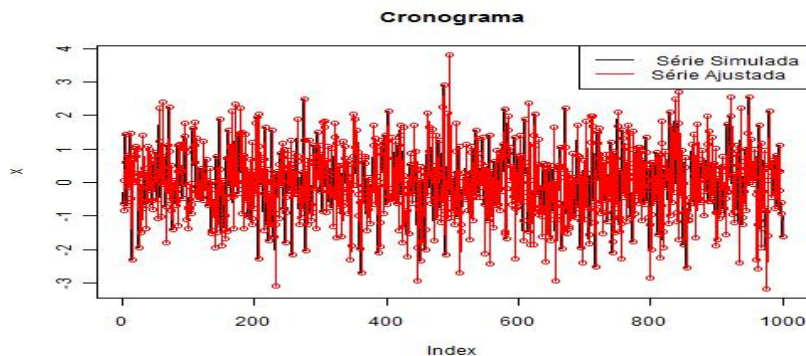


Figura 29: Cronograma da série gerada com a série ajustada representada por pontos

Como se pode verificar a série ajustada acompanha muito bem a série gerada, do ponto de vista da representação gráfica, o modelo encontrado parece ser adequado.

A previsão é realizada pelo comando *predict()*,

```

x.fore = predict(x.ar1, n.ahead = 50)                   # previsão com horizonte de previsão igual a 50
summary(x.fore)                                        # informação sumária sobre o objecto x.fore que é constituído pelos
  Length Class Mode                                     # valores de previsão e respectivos valores de desvio-padrão
pred 50  ts  numeric                                  # 50 valores de previsão do tipo time series e numéricos
se 50  ts  numeric                                  # 50 valores do standard error do tipo time series e numericos
x.fore$pred                                           # objecto que tem os valores de previsão
x.fore$se                                             # objecto que tem o desvio-padrão de cada valor de previsão

```

os limites de previsão associados são calculados por,

```

U=x.fore$pred + 2* x.fore$se                          # limite superior de previsão
L=x.fore$pred - 2* x.fore$se                          # Limite inferior de previsão

```

Para que se possam representar as previsões, e os respectivos limites de forma adequada, realizam-se cálculos complementares para estabelecer os limites da janela gráfica onde se representa a previsão. Se tal não se realizar, por vezes, a representação não permite visualizar correctamente o pretendido. Para tal encontram-se os valores máximos e mínimos entre a série e os valores dos intervalos de previsão,

```

min.x=min(x,L)                                       # cálculo entre o mínimo da série e o mínimo do intervalo de confiança inferior

```



```
max.x=max(x,U) # cálculo entre o máximo da série e o máximo do intervalo de confiança superior
```

Finalmente o gráfico de previsão,

```
x<- ts(x) # é objecto do tipo numeric tem de ser ts (time series) senão o comando seguinte
# não funciona
ts.plot(x, x.fore$pred, main= "Previsão", col=c(1,2), ylim=c(min.x, max.x)) # serie original mais previsão
lines(U, col= "blue", lty = "dashed") # limite confiança superior
lines(L, col= "blue", lty = "dashed") # limite confiança inferior
legend("topright", legend = c(" Valores Observados", "Previsão", "Limites"),col= c(1,2, 4), lty=c(1,1,1))
```

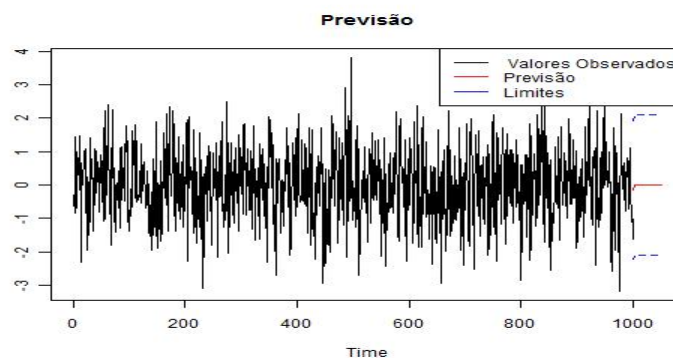


Figura 30: Série gerada com previsão e respectivos intervalos de confiança

2.9.4 Previsão de Sucessões Logaritmizadas

A logaritmização dos valores observados é uma das transformações mais usuais para estacionarizar sucessões cronológicas e permite, quer a linearização de tendências exponenciais, quer a estabilização de variâncias. No entanto, na generalidade dos casos, está-se mais interessado nas previsões das observações em termos de escala inicial do que nos valores dos logaritmos.

A previsão com horizonte m da sucessão original está relacionada com a previsão da sucessão transformada do modo seguinte,

$$X_t(m) = \exp \left[Y_t(m) + \frac{1}{2} V[e_t(m)] \right], \quad (2.80)$$

em que $Y = \ln X_t$ e $e_t(m)$ representa o erro de previsão em m passos da sucessão logaritmizada.

2.10 Análise de dados Reais

Neste ponto analisa-se uma sucessão de dados reais. Esta colecção de dados diz respeito à taxa de mortalidade de mulheres em Portugal no intervalo etário do 50 aos 54 anos, por ano de ocorrência do óbito.

Os dados foram retirados da *human mortality database* (<http://www.mortality.org/>) e referem-se ao período de 1940 a 2006.

O primeiro passo para se poder trabalhar dados em **R** é a leitura de dados a partir da fonte. Neste caso, os dados estão num ficheiro em formato *txt* ("F50_54.txt") dentro da pasta ("My Documents"). Como os dados são referentes a um determinado período, depois de carregados no *workspace* e atribuídos a uma variável ("Dados"), tem de se definir o início dos dados e a sua frequência, neste caso o início é o ano de 1940 e a frequência será um, pois os dados são anuais. Só depois desta fase preparatória os dados estão em condições de serem representados graficamente.

```
Dados<- read.csv("My Documents/F50_54.txt",header=TRUE)      # ler o ficheiro
y<- ts(Dados, start=1940, freq=1)      # transforma-se y num objecto "time series" com inicio em 1940 e frequência 1
plot(y, main= " Taxa de Mortalidade, em Portugal, \n de mulheres no intervalo 50-54 anos \n por ano do óbito",
ylab="Taxa", +xlab="Ano")
```

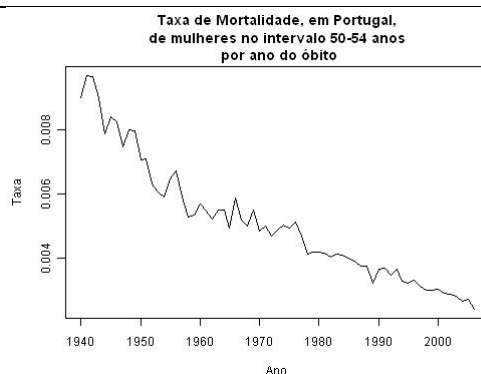


Figura 31: Taxa de mortalidade feminina entre os 50 e 54 anos

Como se pode ver pela figura acima, os dados não são estacionários, pelo que se tem de proceder de acordo com as já mencionadas transformações. A

primeira a aplicar será a transformação de *Box-Cox* para a estabilização da variância através da rotina *BoxCox.ar()*, incluída na livreria "TSA",

```
y.box.cox <- BoxCox.ar(y)      # atribuição dos resultados do comando a uma variável, para que se possam
                                # visualizar
                                # os objectos que este comando produz
```

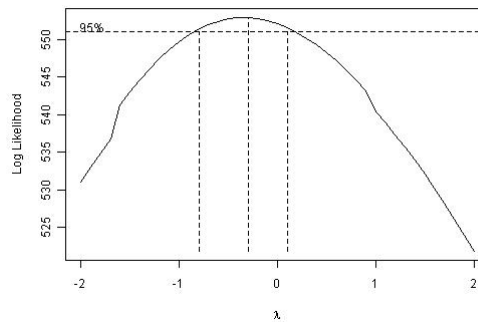


Figura 32: gráfico de *output* da função *Box.cox.ar()*

A figura acima representa a transformação de *Box-Cox* aplicada aos dados em estudo. Note-se que existe um valor que maximiza a parábola e outros dois que delimitam o intervalo de estimação a 95%.

Os valores de λ que correspondem aos limites do intervalo de confiança a 95% visualizam-se da seguinte forma,

```
> (y.box.cox$ci)                # intervalo de confiança a 95% C.I. da transformação para o valor de λ
[1] -0.8 0.1                     # output com -0.8 limite inferior e 0.1 limite superior
```

Pelo que se pode concluir que a transformação logarítmica pode ser aplicada aos dados, dado que $0 \in [-0.8; 0.1]$.

Para aplicar a transformação, opta-se por criar uma nova variável e por a representar graficamente,

```
x <- log(y)
plot(x, main= " logaritmo da Tx de Mort.")
```

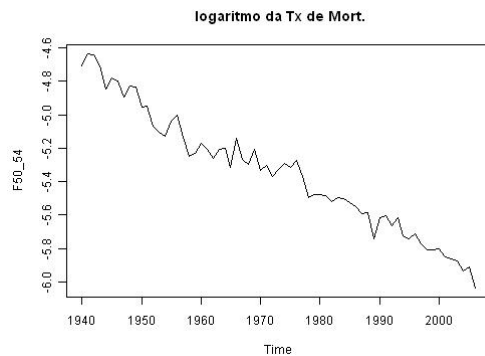


Figura 33: Cronograma da série transformada pelo logaritmo

Como se pode observar, a série transformada não é estacionária em média. Outro modo visual de verificar a não estacionariedade em média é através da inspeção das FAC e FACP, donde se constata que a transformação aplicada não é suficiente para obter a estacionariedade da série em análise.

```
acf(x,main= " ACF do log. da Tx Mort. ")      # FAC do logaritmo da série
pacf(x,main= " PACF do log. da Tx Mort. ")    # FACP do logaritmo da serie
```

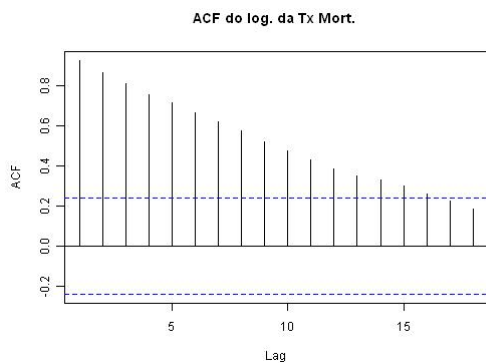


Figura 34: FAC da série transformada pelo logaritmo

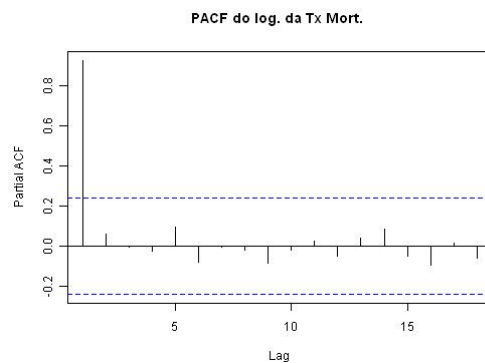


Figura 35: FACP da série transformada pelo logaritmo

O decaimento lento da FAC revela o que já se referiu, que é necessário a aplicação de diferença simples para a estabilização da média.

Começa-se por aplicar uma diferença simples através da criação de uma nova variável, *x.dif*, e repete-se o processo usual. Ou seja, com a representação do cronograma, seguido da FAC e FACP, mas agora com o intuito de identificar o modelo através dos *lags* significativos tanto da FAC como da FACP, isto no caso de apenas uma diferenciação ser suficiente para a estabilização da série.

```
x.dif <-diff(x)                                # nova variável que é a diferença do logaritmo da série
                                              # original

plot(x.dif, main= " diferenciação do logaritmo da Tx de Mort.") # cronograma da diferença do log da série original
abline(h=mean(x.dif), col = "red")             # representação da media no cronograma

acf(x.dif,main= " ACF da dif. log. da Tx Mort. ")      # FAC de x.dif
pacf(x.dif,main= " PACF da dif. do log. da Tx Mort. ") # FACP de x.dif
```

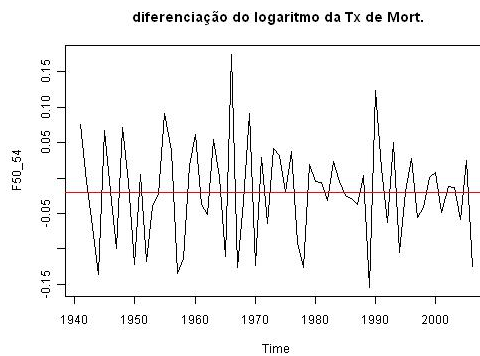


Figura 36: Diferenciação do logaritmo de taxa de mortalidade

Aparentemente a série é estacionária em média, dado que todos os seus valores estão em torno da linha a vermelho, que representa a média da série diferenciada, após aplicação de logaritmo,

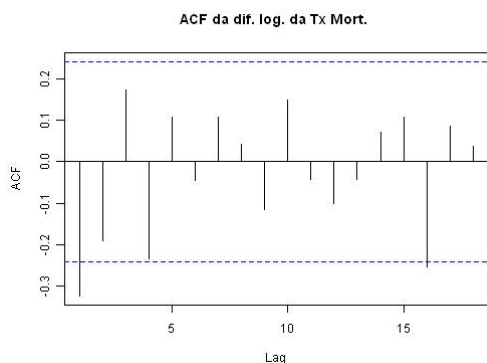


Figura 37: FAC da diferença da série transformada pelo logaritmo

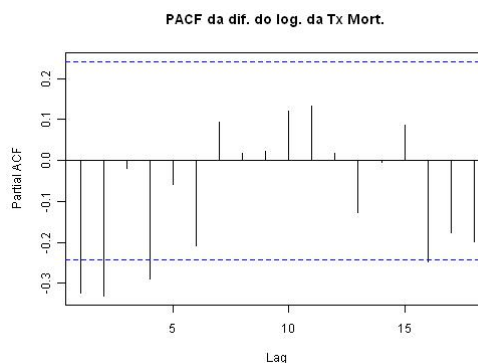


Figura 38: FACP da diferença da série transformada pelo logaritmo

Analisando a FAC e FACP podem equacionar-se dois cenários possíveis,

- Por um lado, o rápido decaimento para zero da FAC e uma FACP que se anula a partir da ordem 4, podendo-se equacionar o ajustamento de um modelo AR(4);
- Por outro lado, também se poderá equacionar ser a FACP a ter um decaimento rápido para zero e uma FAC a anular-se a partir do lag 4, estando neste caso em questão o ajustamento de um modelo MA(4) (embora se tenha analisado este cenário, não foi incluído no trabalho).

A estimação dos parâmetros para o primeiro cenário referido é um modelo AR(4).

Por conveniência de interpretação de código, considere-se uma nova variável z como sendo o resultado da aplicação de uma diferença simples à variável x ,

```
z<- x.dif
(z.fit.ar4<-arima(z, order=c(4,0,0),fixed=c(0,0,0,NA,NA),include.mean = TRUE, method = "CSS-ML"))
Coefficients:
  ar1 ar2 ar3  ar4 intercept
  0  0  0 -0.2528 -0.0196
s.e.  0  0  0  0.1228  0.0066
```

$\text{order}=\text{c}(\text{ar}, \text{dif}, \text{ma}) \rightarrow$ vector em que se estabelece a ordem dos coeficientes,

- ar – número de ordem de coeficientes auto-regressivos (neste caso concreto, 4)
- dif – ordem de diferenciação simples (neste caso concreto, zero, porque se diferenciou antes, a razão deste procedimento está mais adiante).
- ma – número de ordem dos coeficiente médias móveis (neste caso concreto, zero).

$\text{fixed}=\text{c}(\text{ar1}, \text{ar2}, \text{ar3}, \text{ar4}, \text{média}) \rightarrow$ a função *arima()* vai estimar os coeficientes de um processo AR(4) em que se inclui a constante. Como na avaliação do cenário que se realizou, se está a considerar que só o coeficiente de ordem 4 não é nulo, então:

- fixam-se os coeficientes ar1, ar2 e ar3 como sendo zero para que a função não os estime,
- NA, na posição 4, significa que se pretende que a função estime o coeficiente ar4,
- NA, na posição 5, significa que se pretende que a função estime o valor para a constante da série,
- a ordem dos parâmetros a incluir, é sempre, primeiro todos os do tipo AR, seguidos pelos do tipo MA,
- a constante a incluir no modelo será sempre a ordem a seguir ao último parâmetro, neste caso a ordem 5.

Além destas condições, para que o **R** estime a constante através do comando *arima()*, tem de se utilizar a série diferenciada, explicitar que a ordem de diferenciação seja nula, na prática em vez de *arima(x, order=c(4,1,0))* utiliza-se

arima(x.dif, order=c(4,0,0)) pois, caso contrário, o **R** não estimará a constante no modelo.

Note que os coeficientes ar1, ar2, ar3 são nulos, como resultado da sua fixação em zero em “fixed=c(**0,0,0**,NA,NA)” e o coeficiente ar4 = -0.2528, resultado de se fixar em “fixed=c(0,0,0,**NA**,NA)”, a constante da série surge devido a se ter fixado em “fixed=c(0,0,0,NA,**NA**)”.

O coeficiente ar4 e a média são guardados no objecto “z.fit.ar4\$coef”,

```
> z.fit.ar4$coef
      ar1      ar2      ar3      ar4      intercept
0.0000000 0.0000000 0.0000000 -0.2527890 -0.01961503
```

que se podem visualizar individualmente invocando a sua posição no vector,

```
> z.fit.ar4$coef[4]
      ar4
-0.252789
> z.fit.ar4$coef[5]
      intercept
-0.01961503
```

Antes de continuar a análise é conveniente que se analisem as FAC e FACP dos resíduos resultantes do ajustamento. Caso existam correlações fora dos limites de confiança, deve proceder-se a novo ajustamento de modelo, incluindo no modelo parâmetros nos *lags* que estão fora da região de confiança, **com a devida cautela**, pois 95% das correlações devem estar dentro dos limites de confiança, o que implica que 5% podem estar fora. Tem de se ter especial atenção aos *lags* que podem ser indicadores de sazonalidade, 3, 4, 6, 12, 24. Por exemplo, em 24 *lags* se existirem correlações fora dos limites de confiança só nos *lags* 12 e 24, embora tal se verifique só em 2 *lags*, é quase seguro que se esteja perante um caso de sazonalidade.

```
acf(z.fit.ar4$res[-1], main = "FAC Resíduos", cex.main =0.5)
pacf(z.fit.ar4$res[-1], main = "FACP Resíduos", cex.main =0.5)
```

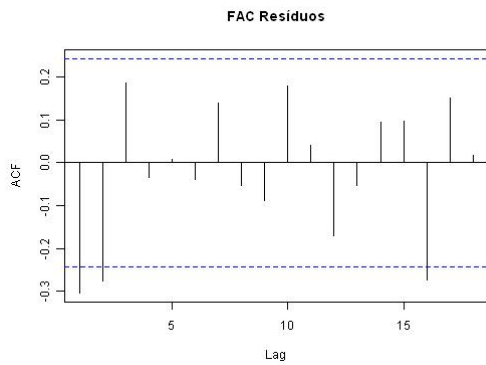


Figura 39: FAC dos resíduos do modelo estimado

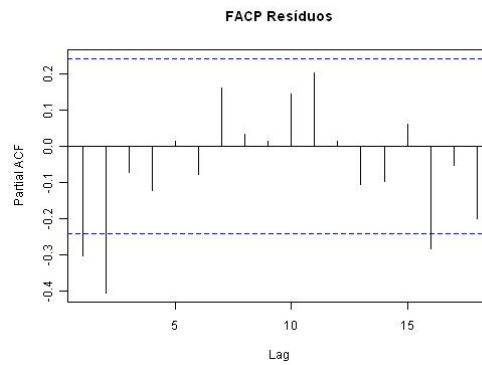


Figura 40: FACP dos resíduos do modelo estimado

Existem resíduos fora da região de confiança, tanto na FAC como a FACP nos *lags* 1 e 2, o que significa que não estamos perante um modelo adequado, há que voltar a tentar novo modelo. Valores fora das bandas de confiança nos *lags* 1 e 2 normalmente indicam que se devem tentar modelos com coeficientes nesses *lags*.

Aqui a experiência é muito importante, apesar de se tentar identificar o modelo baseado nas FAC e FACP teóricas, em que tudo indicava para parâmetros no *lag* 4, a FAC e FACP dos resíduos depois dessa tentativa apontam para o *lags* 1 e 2.

Teste-se então como se comportam os resíduos quando se aplica um modelo AR(1) e MA(1) para ver se se obtêm melhores resultados.

Estimação para AR(1)

```
(z.fit.ar1<-arima(z, order=c(1,0,0),fixed=c(NA,NA),include.mean = TRUE, method = "CSS-ML"))
```

Call:

```
arima(x = z, order = c(1, 0, 0), include.mean = TRUE, fixed = c(NA, NA), method = "CSS-ML")
```

Coefficients:

```
   ar1 intercept
-0.3404 -0.020
s.e. 0.1187  0.006
```

sigma² estimated as 0.004184: log likelihood = 87.01, aic = -170.02

Estimação para MA(1)

```
(z.fit.ma1<-arima(z, order=c(0,0,1),fixed=c(NA,NA),include.mean = TRUE, method = "CSS-ML"))
```


Call:

```
arima(x = z, order = c(0, 0, 1), include.mean = TRUE, fixed = c(NA, NA), method = "CSS-ML")
```

Coefficients:

```
      ma1 intercept
      -0.6557 -0.0198
s.e.  0.0984  0.0026
```

sigma^2 estimated as 0.003596: log likelihood = 91.79, aic = -179.59

Tomando em conta os valores do critério AIC (o comando `arima()` calcula o AIC para cada um dos modelos estimados independentemente de o modelo ser adequado ou não) de cada umas das estimações, o modelo mais adequado é o MA(1), pois é o modelo para o qual o valor do AIC é menor.

A fase seguinte da análise é a representação gráfica das FAC e FACP para cada um dos modelos.

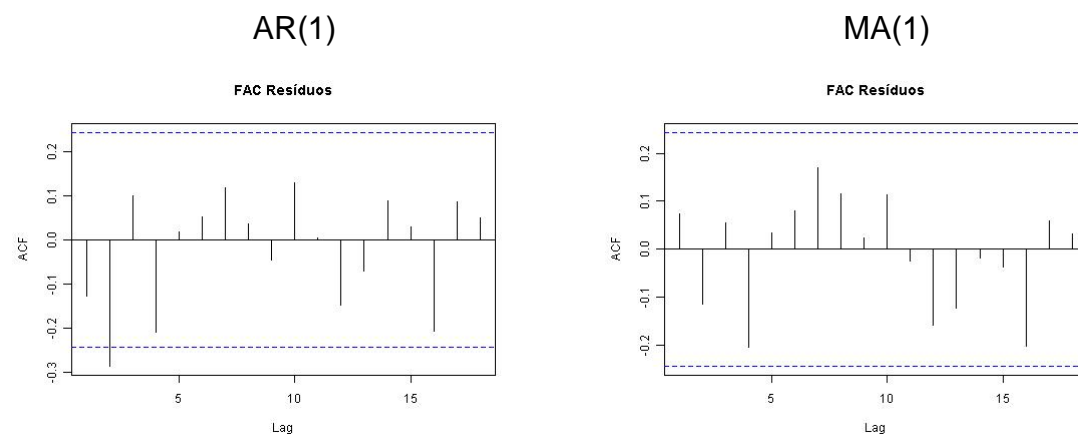
Modelo AR(1)

```
acf(z.fit.ar1$res[-1], main = "FAC Resíduos", cex.main = 0.5)
pacf(z.fit.ar1$res[-1], main = "FACP Resíduos", cex.main = 0.5)
```

Modelo MA(1)

```
acf(z.fit.ma1$res[-1], main = "FAC Resíduos", cex.main = 0.5)
pacf(z.fit.ma1$res[-1], main = "FACP Resíduos", cex.main = 0.5)
```

Comparação das FAC e FACP dos dois modelos estimados



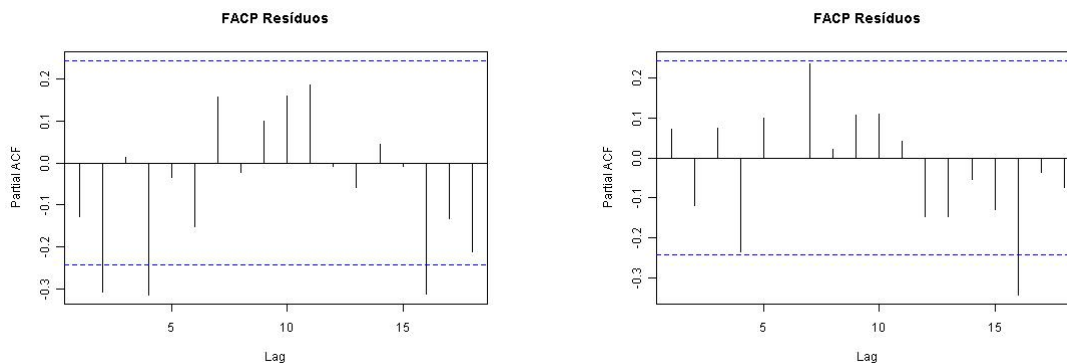


Figura 41: FAC e FACP dos resíduos dos dois modelos estimados

A análise das FAC e FACP mostra que o modelo que melhor se adequa é o MA(1), tal como já se tinha confirmado pelo valor do AIC. Note-se que no *lag* 16 existe uma correlação fora das bandas de confiança, mas não é um *lag* que possa ser influenciado por sazonalidade e não é obrigatório que todas as correlações devam estar dentro das bandas de confiança, há a possibilidade de 5% estarem fora das bandas.

Uma vez estimados os parâmetros deve sempre verificar-se sobre a sua significância, que neste caso são θ e a constante. Para tal utiliza-se o comando `confint()`, que dá os intervalos de confiança dos parâmetros. Se o intervalo de confiança de cada parâmetro não contiver o valor zero, então os parâmetros não se retiram do modelo.

```
> confint(z.fit.ma1)
                2.5 %          97.5 %
ma1            -0.84856330    -0.46287350
intercept      -0.02494944    -0.01464926
```

Como o zero não pertence aos intervalos de confiança de cada um dos parâmetros, então mantêm-se os parâmetros no modelo, neste caso, a constante e θ .

Analisada a significância dos parâmetros, há que testar os resíduos. Os testes que se irão utilizar estão incluídos na livreria *nortest*, que terá de se carregar no *workspace*.

```
> library(nortest)
> lillie.test(z.fit.ma1$res[-1])
```

```

Lilliefors (Kolmogorov-Smirnov) normality test

data: z.fit.ma1$res[-1]
D = 0.0691, p-value = 0.6184

> Box.test(z.fit.ma1$res[-1], lag = 1, type = "Box-Pierce")

Box-Pierce test

data: z.fit.ma1$res[-1]
X-squared = 0.3454, df = 1, p-value = 0.5567

> Box.test(z.fit.ma1$res[-1], lag = 1, type = "Ljung-Box")

Box-Ljung test

data: z.fit.ma1$res[-1]
X-squared = 0.3616, df = 1, p-value = 0.5476

> t.test(z.fit.ma1$res[-1])

One Sample t-test

data: z.fit.ma1$res[-1]
t = -0.0778, df = 64, p-value = 0.9382
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
-0.01546694 0.01430718
sample estimates:
mean of x
-0.000579879

```

Os diferentes métodos aplicados, revelam *p-values* superiores a 5% pelo que não rejeitamos a hipótese nula, motivo pelo qual não se rejeita a normalidade de os resíduos constituírem um ruído branco gaussiano. Desta forma o modelo ajustado, depois de aplicados os testes de significância aos parâmetros e de normalidade aos resíduos, revela-se adequado.

A avaliação da qualidade dos resíduos, que aparece com mais frequência na literatura, é através das representações das FAC, FACP, *Box-Plot*, *PP-Plot* e histograma com ajustamento normal, são métodos gráficos de análise e decisão. Além dos métodos gráficos são também utilizados os testes paramétricos, a utilização dos dois tipos de análise e decisão complementam-se, pois o objectivo é o mesmo: validar os pressupostos dos resíduos. Os diferentes tipos de

gráficos, bem como o cálculo das estatísticas de teste e *p-values* estão nos anexos II e III.

Segue-se o gráfico de previsão com as respectivas bandas de confiança

```
fit=arima(z, order=c(1,1,1), xreg=1:66)
fore=predict(fit, 15, newxreg=(66:81))
fore$pred
fore$se
U=fore$pred + 2* fore$se
U
L=fore$pred - 2* fore$se
L
minx=min(x,L)
maxx=max(x,U)
minx
maxx

ts.plot(z,fore$pred, col=1:2, main= " previsão")
lines(U, col= "blue", lty = "dashed")
lines(L, col= "blue", lty = "dashed")
```

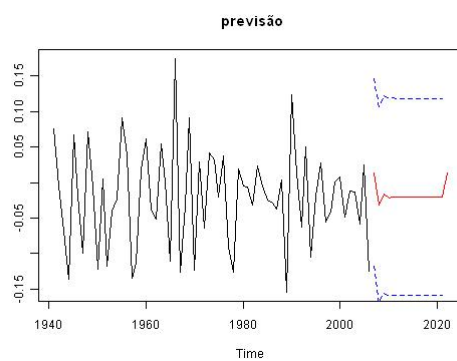


Figura 42: Previsão e respectivos limites de confiança

Capítulo 3

Processos Autoregressivos de valores inteiros não negativos de ordem 1

3.1 A operação *Thinning*

A modelação de séries de contagem, estacionárias, envolvendo somente números inteiros, não negativos, não pode ser realizada através dos processos usuais do tipo ARMA, dado estes assumirem ruídos gaussianos.

Para tal, foi estudada uma operação que substituísse a operação multiplicação dos modelos ARMA por uma em que fosse possível operar um escalar com uma variável aleatória inteira, cujo resultado fosse ainda uma variável aleatória inteira. Tal operador foi proposto por *Steutel & Van Harn* (1979) e é denominado *thinning*.

Não existe tradução da expressão *thinning* para português, pelo que será utilizado no termo original.

A operação *thinning*, usualmente representada por $*$, utiliza-se em variáveis de contagem, sempre que num conjunto de elementos cada um é seleccionado (ou eliminado) com uma certa probabilidade.

Em analogia à aplicação da distribuição binomial, pode-se considerar uma urna com bolas pretas e bolas brancas, a probabilidade de retirar uma bola branca é α , ($\alpha \in [0,1]$). $\alpha * X$ representa o número de bolas brancas retiradas com reposição em X extracções. Isto é, a operação *thinning* sobre uma variável aleatória inteira não negativa e define-se por,

$$\alpha * X = \sum_{i=1}^X Y_i, \quad (3.1)$$

onde $\{Y_i\}$ é uma sequência de variáveis aleatórias independentes e identicamente distribuídas (i.i.d) de Bernoulli com parâmetro α ($P(Y_i = 1) = \alpha$), independentes de X . A sequência Y_1, Y_2, \dots designa-se por série de contagem de $\alpha * X$. Note-se que dado X , $\alpha * X$ tem distribuição binomial de parâmetros (X, α) . Por este motivo também é usual chamar à operação binomial *thinning*.

3.2 Propriedades da Operação *Thinning*

A operação *thinning* tem propriedades que são elencadas no Lema seguinte [Silva, (p.32), (2005)].

LEMA 3.1. (Propriedades da operação *thinning*)

Sejam X, Y e Z variáveis aleatórias inteiras não negativas e $\alpha, \beta \in [0, 1]$. Então,

$$0 * Y = 0$$

$$1 * Y = Y$$

$$\alpha * (\beta * Y) \stackrel{d}{=} (\alpha\beta) * Y, \text{ onde } \stackrel{d}{=} \text{ representa igualdade em distribuição.}$$

Se as séries $\alpha * Y$ e $\beta * Z$ são independentes e independentes de X, Y e Z então

$$E[\alpha * Y] = \alpha E[Y]$$

$$E[(\alpha * Y)^2] = \alpha^2 E[Y^2] + \alpha(1 - \alpha) E[Y]$$

$$E[X(\alpha * Y)] = \alpha E[XY]$$

$$E[(\alpha * Y)(\beta * Z)] = \alpha\beta E[YZ]$$

$$E[(\alpha * Y)^3] = \alpha^3 E[Y^3] + 3\alpha^2(1 - \alpha) E[Y^2] + \alpha(1 - \alpha)(1 - 2\alpha) E[Y]$$

$$E[X(\alpha * Y)^2] = \alpha^2 E[XY^2] + \alpha(1 - \alpha) E[XY]$$

$$E[XY(\beta * Z)] = \beta E[XYZ]$$

$$E[X((\alpha * Y)(\beta * Z))] = \alpha\beta E[XYZ].$$

A demonstração deste Lema sai fora do âmbito do trabalho, em Silva (p.179) encontram-se as referências aos autores que realizaram a demonstração deste Lema.

3.3 Modelos autoregressivos inteiros de ordem 1

Os processos com estrutura autoregressiva de ordem 1, de valores inteiros não negativos (abreviadamente INAR(1)), acrónimo de **first-order INteger-valued AutoRegressive**, foram introduzidos por *McKenzie* [1985, 1988] (e estudados por diversos autores, como por exemplo, *Al-Osh* e *Alzaid* [1987]) e são baseados na operação binomial *thinning* definida por *Steutel* e *Van Harn*. *Du* e *Li* [1991] e *Latour* [1998] generalizaram estes modelos até à ordem p .

O desenvolvimento destes modelos deve-se à crescente necessidade de realizar contagens de séries que assumem valores inteiros não negativos, como por exemplo o número de nados vivos por concelho, número de óbitos por concelho, número de utilizadores ligados a um servidor de internet por hora, número de hóspedes por dia num hotel, etc.

Definição (*McKenzie* (1987))

Um processo $\{X_t\}$ diz-se um processo INAR(1), se satisfaz a equação,

$$X_t = \alpha * X_{t-1} + \varepsilon_t, \quad (3.2)$$

onde $\alpha \in [0,1]$ e ε_t é uma sequência de variáveis aleatórias independentes e identicamente distribuídas, independentes de $\{X_t\}$, com média μ_ε e variância finita σ_ε^2 .

Para que o processo INAR(1) acima definido seja estacionário $\alpha \in (0,1)$ (*Al-Osh* e *Alzaid* (1987)). Da definição resulta imediatamente que $\{X_t\}$ é um processo de *Markov* e que ε_t e X_s são independentes para $s < t$. Estes autores mostraram que a distribuição marginal do modelo INAR(1) definido por (3.2) pode ser escrita em termos da sequência de inovações $\{\varepsilon_t\}$

$$X_t = \sum_{j=0}^{t-1} \alpha^j * \varepsilon_{t-j}.$$

Uma realização deste processo X_t pode ser interpretada como a soma de duas componentes: uma constituída pelos elementos que sobrevivem a X_{t-1} , cada um com probabilidade α , e outra pelos elementos, ε_t , que entram no sistema no intervalo de tempo $(t-1, t]$, as chamadas inovações do processo.

3.4 Momentos até à segunda ordem do modelo INAR(1)

Considere-se um processo estocástico $\{X_t\}$ satisfazendo a equação (3.2). Uma vez que $\{X_t\}$ é uma sucessão de variáveis aleatórias dependentes, é particularmente importante a descrição dos momentos pelo menos até à segunda ordem. *Al-Osh* e *Alzaid* [1987] obtiveram as expressões para a média, covariância, autocovariância e função de autocorrelação do processo INAR(1). Assim, aplicando o valor esperado e a variância a ambos os membros de (3.2) e atendendo às propriedades da operação *thinning* tem-se,

$$E(X_t) = \alpha E(X_{t-1}) + \mu_\varepsilon$$

$$V(X_t) = \alpha^2 V(X_{t-1}) + \alpha(1-\alpha)E(X_{t-1}) + \sigma_\varepsilon^2,$$

em que μ_ε e σ_ε^2 representam a média e a variância da sequência de inovações $\{\varepsilon_t\}$.

Supondo que o processo é estacionário, $E(X_t) = E(X_{t-1}) = \dots = E(X_0)$ e $V(X_t) = V(X_{t-1}) = \dots = V(X_0)$, *Al-Osh* e *Alzaid* [1987] deduziram as expressões seguintes para a média e variância,

$$E(X_t) = \frac{\mu_\varepsilon}{1-\alpha}$$

$$V(X_t) = \frac{\alpha\mu_\varepsilon + \sigma_\varepsilon^2}{1-\alpha^2}$$

Utilizando as mesmas propriedades, estes autores encontraram uma expressão para a função de autocovariância, γ_k , para cada k inteiro não negativo,

$$\begin{aligned} \gamma_k &= \text{cov}(X_t, X_{t+k}) \\ &= E(X_t, X_{t+k}) - E(X_t)E(X_{t+k}) \end{aligned}$$

$$\begin{aligned}
&= E\left[X_t(\alpha * X_{t+k-1}\varepsilon_{t+k})\right] - E(X_t)E(\alpha * X_{t+k-1} + \varepsilon_{t+k}) \\
&= \alpha E(X_t X_{t+k-1}) + E(X_t \varepsilon_{t+k}) - \alpha E(X_t)E(X_{t+k-1}) + E(X_t)E(\varepsilon_{t+k}) \\
&= \alpha \gamma_{k-1} + \text{cov}(X_t, \varepsilon_{t+k})
\end{aligned}$$

Recordando que X_t e ε_{t+k} são independentes para $k > 0$, então

$$\gamma_k = \alpha \gamma_{k-1}.$$

Repetindo este raciocínio conclui-se que

$$\gamma_k = \alpha^k \gamma_0, \quad k > 0 \quad (3.3)$$

Por fim calcula-se a função de autocorrelação

$$\begin{aligned}
\rho_k &= \alpha \rho_{k-1} \\
&= \dots \\
&= \alpha^k, \quad k > 0
\end{aligned} \quad (3.4)$$

Das relações (3.3) e (3.4) resultam as funções de autocovariância, γ_k , e autocorrelação, ρ_k no lag k ,

$$\gamma_k = \alpha^{|k|} \gamma_0, \quad k \in \mathbb{Z} \quad (3.5)$$

$$\rho_k = \alpha^{|k|}, \quad k \in \mathbb{Z} \quad (3.6)$$

A função de autocorrelação parcial é análoga à dos processos AR(p), pois, por definição resulta de (2.28), o caso particular de ordem 1 aplicado ao modelo INAR tem por expressão

$$\phi_{11} = \alpha, \quad \phi_{kk} = 0, \quad k \geq 2. \quad (3.7)$$

NOTA

No que respeita à estimação, os estimadores dos momentos de segunda ordem do modelo INAR(1) obtém-se de modo análogo aos correspondentes estimadores de um processo AR(1).

Em regra geral, os diversos autores consideram que $\varepsilon_t \sim P(\lambda)$, deste modo

prova-se que $X_t \sim P\left(\frac{\mu}{1-\alpha}\right)$, consultar, por exemplo, Silva, (p.35)).

3.5 Identificação da ordem

O processo de modelação sugerido por *Box e Jenkins*, apresentado no capítulo 2 e constituído pelas etapas de identificação, estimação e avaliação de diagnóstico, será adaptado também aos modelos INAR.

A identificação consiste, essencialmente, na identificação da ordem do modelo porque a selecção de um modelo INAR depende apenas da natureza dos dados. Os modelos INAR são adequados quando as observações são constituídas por valores inteiros não negativos, nomeadamente, quando os valores são reduzidos.

A ordem do modelo é usualmente identificada através da análise da função de autocorrelação e da função de autocorrelação estimadas e dos critérios de selecção BIC e AIC, como já foi referido no capítulo 2.

3.5.1 Critérios de selecção de modelos.

Os critérios de AIC e BIC dependem da função de verosimilhança da amostra em análise. *Chandler* [1996] aproxima a função de verosimilhança pela função do critério de *Whittle* (consultar Silva (p.183-186)) e pode ser interpretada com uma função de quasi-verosimilhança¹. Deste modo, define-se AIC (*Akaike* [1974]), quando o critério é utilizado para estimar $\theta = (\alpha_1, \alpha_2, \dots, \alpha_p, \lambda)$, por

$$AIC(\theta) = -2L(\theta) + 2k$$

Em que $k = p + 1$ é o número de parâmetros estimados e $L(\theta)$ é o logaritmo de quasi-verosimilhança. A regra de escolha do melhor modelo é idêntica à referida no capítulo 2, ou seja, o menor valor de AIC.

Akaike [1978] propôs outro critério para determinar a ordem do modelo, designado por BIC, também já referido no capítulo 2 e que agora é definido por,

$$BIC(\theta) = -2L(\theta) + k(1 + \log N)$$

¹ A função de quasi-verosimilhança, foi introduzida por Robert Wedderburn [1974] e utiliza-se quando não é possível utilizar a função de verosimilhança e tem por base a relação entre μ e $Var(Y)$, quando não se supõe que os dados têm uma função de distribuição específica, consultar, por exemplo, Agresti (p.280).

Onde $k = p+1$ é o número de parâmetros estimados, $L(\theta)$ é o logaritmo da função de quasi-verosimilhança e N é o número de dados em análise.

3.5.2 Estimação de parâmetros do modelo.

Um dos métodos de estimação dos parâmetros do modelo é o método dos momentos não condicionados, também conhecido por estimadores de *Yule-Walker*.

Estimadores de Yule-Walker

Considere-se uma equação às diferenças de *Yule-Walker*, a substituição de ρ_k pelo seu valor estimado, $\hat{\rho}_k$, $k = 1, \dots, p$, então

$$\hat{\rho}_k = \hat{\alpha}_1 \hat{\rho}_{k-1} + \hat{\alpha}_2 \hat{\rho}_{k-2} + \dots + \hat{\alpha}_p \hat{\rho}_{k-p}, \quad k = 1, \dots, p \quad (3.8)$$

Como já foi referido, os estimadores de Yule-Walker obtêm-se resolvendo o sistema de p equações em ordem a $\hat{\alpha}_1, \hat{\alpha}_2, \dots, \hat{\alpha}_p$ (Du & Li (1991)).

Considere-se um processo INAR(1), $X_t = \alpha * X_{t-1} + \varepsilon_t$. De (3.6), $\rho_k = \alpha^{|k|} \Rightarrow \rho_1 = \alpha$, pelo que o estimador de Yule-Walker do parâmetro α é $\hat{\rho}_1$.

Dado que, $E(X_0) = \frac{\mu_\varepsilon}{1-\alpha}$ (Al-Osh e Alzaid [1987]), então $\mu_z = (1-\alpha)E[X_t]$,

Como $\mu_z = \lambda$, por se tratar de uma distribuição de Poisson, $E[X_t] = \bar{X}$, então

$$\hat{\lambda} = (1-\hat{\alpha})\bar{X}.$$

Estimadores dos mínimos quadrados condicionais

Para o modelo INAR(1), Al-Osh & Alzaid (1987) obtiveram os estimadores dos mínimos quadrados condicionais para α e λ

$$\hat{\alpha} = \frac{\sum X_t X_{t-1} - \frac{\sum X_t \sum X_{t-1}}{N}}{\sum X_{t-1}^2 - \frac{(\sum X_{t-1})^2}{N}}$$

e

$$\hat{\lambda} = \frac{1}{N} \left(\sum X_t - \hat{\alpha} \sum X_{t-1} \right),$$

onde em todas as somas, t toma valores de 1 a N .

Observação:

Note-se que tal como nos processos AR(1) também nos processos INAR(1) os estimadores de *Yule-Walker* coincidem com os estimadores de máxima verosimilhança de (α, λ) .

3.5.3 Validação do modelo

A validação do modelo realiza-se da forma usual aplicada aos modelos ARMA através respectiva aplicação de testes estatísticos aos resíduos e parâmetros estimados. Os resíduos seguem uma distribuição de *Poisson*, que são aproximadamente normais, de forma assintótica, quando n é suficientemente grande.

3.5.4 Previsão

Dada uma amostra X_1, \dots, X_n , os pressupostos para a realização de previsão são os mesmos utilizados nos modelos ARMA no capítulo 2, onde se tem por critério de medida da qualidade de aproximação entre os valores exacto e previsto o erro quadrático médio, definido por

$$eqm = E \left[\left(X_{N+m} - X_N(m) \right)^2 \right],$$

onde $X_N(m)$ é o predictor de X_{N+m} no instante $N+m$.

Procura-se, assim, a expressão de $X_N(m)$ que minimize o erro quadrático médio, que já se viu ser a esperança condicional,

$$\hat{X}_N(m) = E \left[X_{N+m} \mid X_1, \dots, X_N \right].$$

Du & Li (1991) sugerem como predictor estimado

$$\hat{X}_N(m) = \sum_{i=1}^p \alpha_i \hat{X}_N(m-i) + \lambda, \quad m \in \mathbb{N}.$$

3.6 Simulação de um processo INAR(1) em R

Considere-se o processo INAR(1) definido por (3.2), $X_t = \alpha * X_{t-1} + \varepsilon_t$.

Por conveniência de notação e adaptação ao código de **R** que se vai construir, considere-se (3.2) representado por $Y_t = \alpha * Y_{t-1} + Z_t$, com $Z_t \sim P(\lambda)$ e $\alpha * Y[t]_{\lfloor Y[t] \rfloor} \sim \text{Bin}(Y[t], \alpha)$.

Utiliza-se Y_t em vez de X_t porque depois de construído o processo, devido ao procedimento de *Burn In* realiza-se uma definição de variável excluindo algumas observações iniciais geradas.

Os parâmetros de inicialização do processo são

Dim <- 1500	# dimensão da amostra a gerar
BurnIn <- 500	# número de parâmetros iniciais a excluir pelo método de Burn in
Alfa <- 0.4	# parâmetro da distribuição Binomial
Lambda <- 1	# parâmetro da distribuição Poisson

A geração de inovações com distribuição de Poisson de parâmetro, λ , realiza-se do seguinte modo

Z <- rpois(Dim, Lambda)	# geração de inovações de Poisson
-------------------------	-----------------------------------

Para melhor identificar os gráficos, vão-se construir expressões com nomes e valores para os títulos dos gráficos,

names1 <- c('Histograma', 'Cronograma', 'de inovações com dist. de Poisson de parâmetro', 'lambda')	# nomes
vals1 <- round(c(Alfa, Lambda), 3)	# valores a utilizar arredondados a 3 décimas
express1 <- bquote (.(as.name(names1[1])) ~"	# construção de nome para o histograma
"~.(as.name(names1[3]))~"	
"~.(as.name(names1[4]))=.(vals1[2])"	
express2 <- bquote (.(as.name(names1[2])) ~"	# construção de nome para cronograma
"~.(as.name(names1[3])) ~"	
"~.(as.name(names1[4]))=.(vals1[2])"	

A representação do histograma das inovações de parâmetro λ , é

hist(Z, main=(express1))	# histograma das inovações de Poisson de parâmetro λ
--------------------------	--------------------------------------------------------------

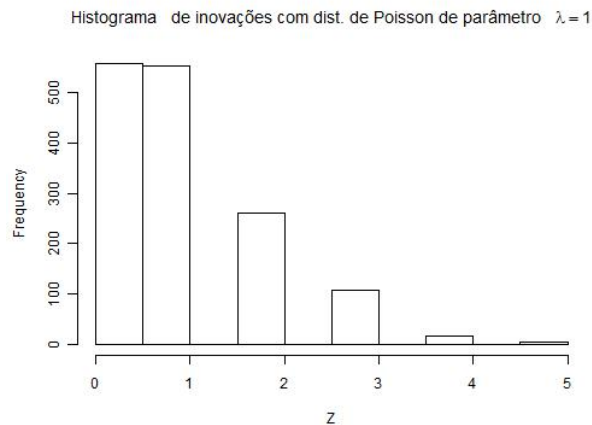


Figura 43: Histograma de inovações com distribuição de Poisson de parâmetro $\lambda = 1$

Cujo cronograma se obtém,

```
plot.ts(Z, main=(express2) # cronograma das inovações de Poisson de parâmetro  $\lambda$ 
```

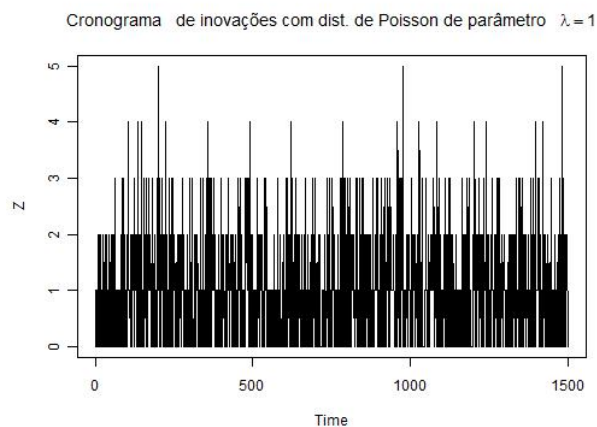


Figura 44: Cronograma de inovações com distribuição de Poisson de parâmetro $\lambda = 1$

Para continuar o processo têm de se iniciar as variáveis a incluir no ciclo de construção do processo INAR(1)

```
x <-numeric() # inicialização de variável
w <-numeric() # inicialização de variável
y <-numeric() # inicialização de variável
```

A construção do processo em si é realizada por

```
y[1] <- Z[1] # inicialização de y
for (t in 2:Dim) y[t] <- rbinom(1, y[t - 1], Alfa) + Z[t] # ciclo que constrói o processo.
```

De forma a eliminar possíveis variações no arranque das simulações é usual eliminarem-se algumas das primeira observações, num procedimento denominado de *Burn In*, que se introduz da seguinte forma,

```
x <- y[BurnIn:(Dim-1)] # eliminação das primeiras 500 observações
# a variável x é agora o processo INAR(1), da forma  $X_t = \alpha * X_{t-1} + Z_t$ 
x<-ts(x) # transformação do objecto x de numeric para ts
```

A fase seguinte é a da estimação dos parâmetros , α e λ , pelo método dos momentos,

```
Alfa.est<-acf(x, plot = FALSE)[1] # o  $\alpha$  estimado é o coeficiente de ordem 1 da FAC
# que é um objecto do tipo acf, plot = FALSE, é para que não
# seja apresentado o gráfico da FAC

Alfa.est <- Alfa.est$acf[1] # tem de se operar uma mudança de variável para que o objecto seja do tipo
# numeric, e como tal permita realizar operações de multiplicação necessárias para
# prosseguir com a análise

Lambda.est <- (1-Alfa.est)*mean(x) # estimação de  $\lambda$ 

(Alfa.est) # coeficiente estimado para  $\alpha$  , o valor utilizado na simulação foi 0.4
[1] 0.4012204

(Lambda.est) # coeficiente estimado para  $\lambda$  , o valor utilizado na simulação foi 1
[1] 0.942479
```

As estatísticas descritivas da série gerada e os coeficientes estimados de λ, α podem-se visualizar numa matriz,

```
M.Descriptives_Res<-matrix(c(Lambda.est, # Definição da Matriz
Alfa.est,
mean(Res),
sd(Res),
var(Res)), 5,1)
colnames (M.Descriptives_Res) <- c(" Valor ") # nomes colunas
rownames (M.Descriptives_Res) <- c("Lamba Estimado","Alfa Estimado","Média da Série", "Desvio-padrão da Série",
"Variância da Série")
print(round(M.Descriptives_Res, digits=3))
Valor
Lamba Estimado 0.942
Alfa Estimado 0.401
Média da Série -0.003
Desvio-padrão da Série 1.144
Variância da Série 1.308
```

Para obter os resíduos, tem de se calcular o ajustamento entre a série original e a série obtida através dos valores estimados de α e λ ,

```
ajus<-numeric() # definição da variável ajus como numérica
```

```

for (j in 2:(Dim-BurnIn)) ajus[j] <- Alfa.est * x[j-1] + Lambda.est      # ciclo que calcula os valores do ajustamento
Res <- numeric()                                                    # definição da variável Res como numérica
for (i in 2:(Dim-BurnIn)) Res[i] <- x[i] - ajus[i]                  # ciclo que calcula os resíduos
Res <- Res[2:(Dim-BurnIn)]                                          # eliminação da primeira observação dos resíduos

```

Uma vez realizados os cálculos necessários, é altura de se realizarem as várias representações gráficas necessárias. Utilizam-se nomes e valores na construção dos títulos dos gráficos,

```

names <- c('alpha','lambda', 'INAR(1)', 'Simulado','Histograma','FAC', 'FACP', 'Ajustamento','Resíduos', 'QQ plot', ' Caixa de Bigodes')
vals <- round( c( Alfa, Lambda,Alfa.est,Lambda.est),3)

```

O código utilizado para representar o cronograma e o histograma é,

```

# Cronograma
expr1 <- bquote  (.(as.name(names[3])) ~~~~"
                "~.(as.name(names[4])) ~~~~"
                "~.(as.name(names[1]))==.(vals[1])~~~~"
                "~.(as.name(names[2]))==.(vals[2])")
plot(x, type = "l",main = expr1, col = "blue", cex.main =1)
abline(h=mean(x), col ="2")
abline(h=mean(x)+ 1.96 * sd(x), col ="3")
legend("topright", legend = c(" Série Simulada", " Média", " Desvio-Padrão"), col= c(4, 2,3), lty=c(1,1))

# Histograma
expr2 <- bquote  (.(as.name(names[5])) ~~~~"
                "~.(as.name(names[3])) ~~~~"
                "~.(as.name(names[1]))==.(vals[1])~~~~"
                "~.(as.name(names[2]))==.(vals[2])")
hist(x,main = expr2)

```

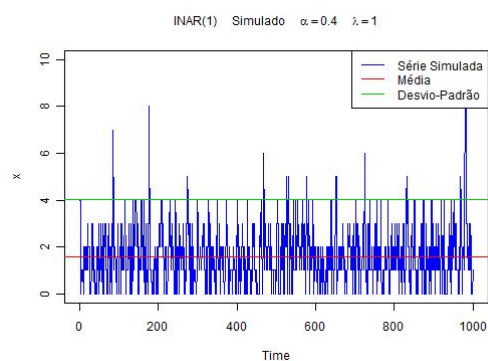


Figura 45: Cronograma de um processo INAR(1) simulado com $\alpha = 0.4$ $\lambda = 1$

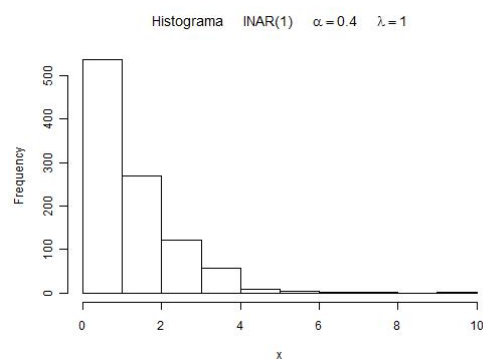


Figura 46: Histograma de um processo INAR(1) simulado com $\alpha = 0.4$ $\lambda = 1$

As FAC e FACP do processos são realizadas por,

```
# Função de Autocorrelação
expr3 <- bquote  (.(as.name(names[6])) ~~~~"
  "~.(as.name(names[3])) ~~~~"
  "~.(as.name(names[1]))==.(vals[1])~~~~"
  "~.(as.name(names[2]))==.(vals[2])")
acf(x, main = expr3)

# Função de Autocorrelação Parcial
expr4 <- bquote  (.(as.name(names[7])) ~~~~"
  "~.(as.name(names[3])) ~~~~"
  "~.(as.name(names[1]))==.(vals[1])~~~~"
  "~.(as.name(names[2]))==.(vals[2])")
pacf(x, main = expr4)
```

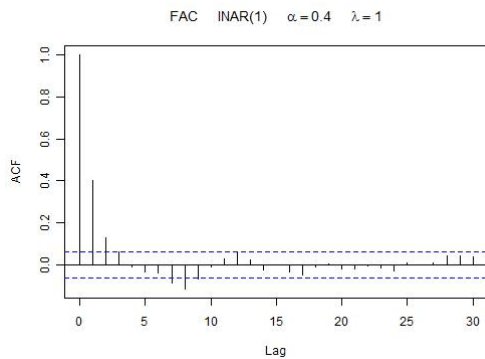


Figura 47: FAC de um processo INAR(1)
simulado com $\alpha = 0.4$, $\lambda = 1$

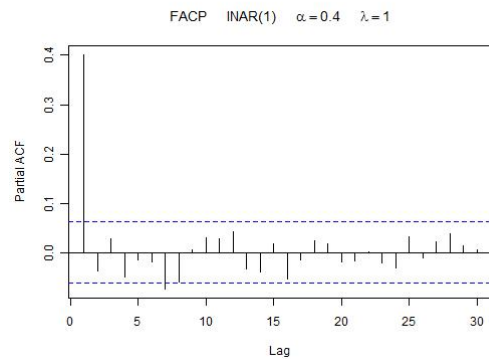


Figura 48: FACP de um processo INAR(1)
simulado com $\alpha = 0.4$, $\lambda = 1$

O código para a visualização do ajustamento é,

```
expr5 <- bquote  (.(as.name(names[8])) ~~~~"
  "~.(as.name(names[3])) ~~~~"
  "~tilde.(as.name(names[1]))==.(vals[3])~~~~"
  "~tilde.(as.name(names[2]))==.(vals[4])")
plot(x, main=expr5)
lines(ajus, col="red")
legend("topright", legend = c(" Série Simulada", " Ajustamento"), col= c(1, 2), lty=c(1,1))
```

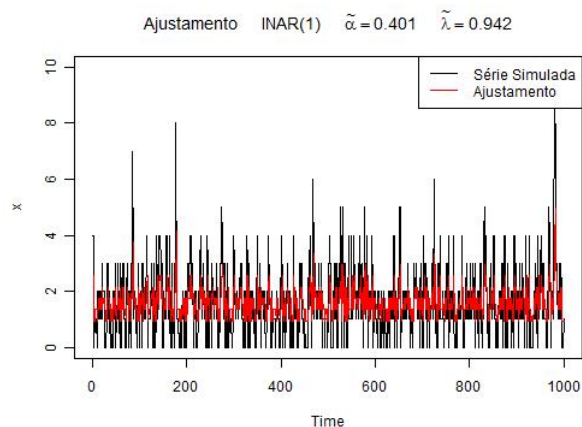


Figura 49: Ajustamento um processo INAR(1) com

$\tilde{\alpha} = 0.401$, $\tilde{\lambda} = 0.942$ estimados e simulado com $\alpha = 0.4$, $\lambda = 1$

A visualização dos resíduos e seu histograma são realizados por,

```
# Plot de Resíduos
expr6 <- bquote  (.(as.name(names[9])) ~~~~"
                 "~.(as.name(names[3])) ~~~~"
                 "~-tilde(.(as.name(names[1])))=.(vals[3])~~~~"
                 "~-tilde(.(as.name(names[2])))=.(vals[4])")
plot(Res, main=expr6)
abline(h=mean(Res), col="2")
abline(h=mean(Res)+ 1.96 * sd(Res), col="4")
abline(h=mean(Res)-1.96 * sd(Res), col="4")
legend("topright", legend = c(" Resíduos", " Média", " Desvio-Padrão" ), col= c(1, 2,4), lty=c(1,1,1))

# Histograma de Resíduos
expr7 <- bquote  (.(as.name(names[9])) ~~~~"
                 "~.(as.name(names[3])) ~~~~"
                 "~.(as.name(names[5])) ~~~~"
                 "~-tilde(.(as.name(names[1])))=.(vals[3])~~~~"
                 "~-tilde(.(as.name(names[2])))=.(vals[4])")
hist(Res, prob=TRUE, main=expr7)
lines(density(Res), col= "2")
```

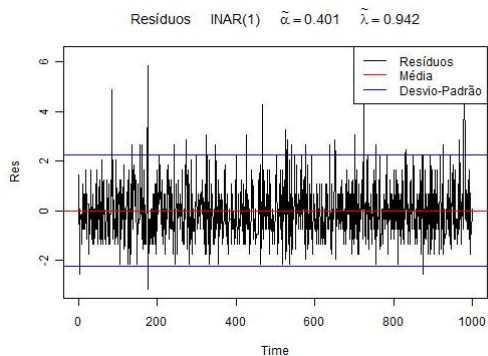


Figura 50: Cronograma dos resíduos do processo resultantes da estimação de $\tilde{\alpha} = 0.401$, $\tilde{\lambda} = 0.942$

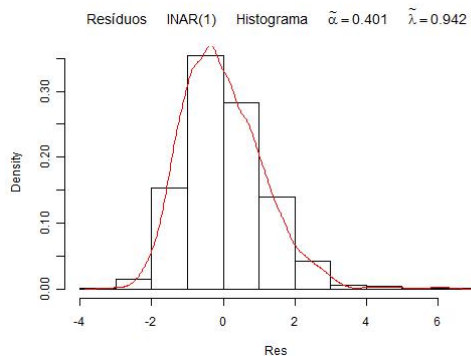


Figura 51: Histograma dos resíduos do processo resultantes da estimação de $\tilde{\alpha} = 0.401$, $\tilde{\lambda} = 0.942$

Que têm FAC e FACP,

```
# Função de Autocorrelação de resíduos
expr8 <- bquote (.(as.name(names[9])) ~~~~"
              "~.(as.name(names[3])) ~~~~"
              "~.(as.name(names[6])) ~~~~"
              "~tilde(.as.name(names[1]))==.(vals[3])~~~~"
              "~tilde(.as.name(names[2]))==.(vals[4])
acf(Res, main=expr8)

# Função de Autocorrelação parcial de resíduos
expr9 <- bquote (.(as.name(names[9])) ~~~~"
              "~.(as.name(names[3])) ~~~~"
              "~.(as.name(names[7])) ~~~~"
              "~tilde(.as.name(names[1]))==.(vals[3])~~~~"
              "~tilde(.as.name(names[2]))==.(vals[4])
pacf(Res, main=expr9)
```

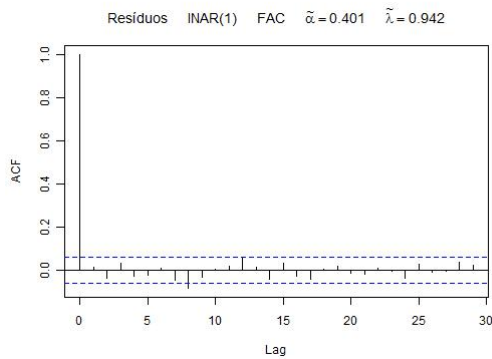


Figura 52: FAC dos resíduos do processo resultantes da estimação de $\tilde{\alpha} = 0.401$, $\tilde{\lambda} = 0.942$

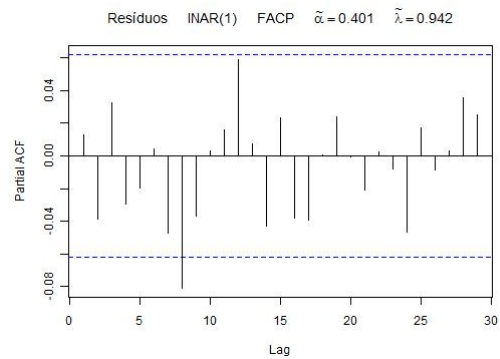


Figura 53: FACP dos resíduos do processo resultantes da estimação de $\tilde{\alpha} = 0.401$, $\tilde{\lambda} = 0.942$

Falta a aplicação de testes aos resíduos resultantes do ajustamento efectuado,

```
library(nortest)
Box.test(Res, lag = 1, type = "Box-Pierce")
Box.test(Res, lag = 1, type = "Ljung-Box")
t.test(Res)
```

cujos resultados se condensam na tabela seguinte

Box-Pierce test	Box-Ljung test	One Sample t-test
X-squared = 0.1584, df = 1, p-value = 0.6907	X-squared = 0.1588, df = 1, p-value = 0.6902	t = -0.0735, df = 998, p-value = 0.9414

O p -value de cada um dos testes realizados não evidencia a rejeição da hipótese nula de que os resíduos se comportam como um ruído branco.

A previsão pode realizar-se, no entanto não é possível estimar os limites de confiança da previsão, uma vez que se desconhece a sua distribuição de probabilidade. O código para a previsão é a implementação em **R** do resultado de *Brannas* [1994] em que o preditor h passos adiante para o modelo de *Poisson* é dado pela expressão

$$\hat{X}_{T+h|T} = \alpha^h \left[X_T - \frac{\lambda}{1-\alpha} \right] + \frac{\lambda}{1-\alpha}$$

```
h <- 200 # definição do horizonte de previsão
m <- numeric() # estabelecimento da variável índice do ciclo
for (m in 1: h) Forecast[m] <- ((Alfa.est)^(m))*((x[length(x)])-((Lambda.est)/(1-Alfa.est)))+(Lambda.est/(1-Alfa.est))
Forecast <- ts(Forecast, start = length(x)+1) # transformar os valores estimados em ts
ts.plot(x, Forecast, main= "Previsão", col=c(1,2)) # representação gráfica da previsão
legend("topright", legend = c(" Valores Observados", "Previsão"),col= c(1,2), lty=c(1,1))
```

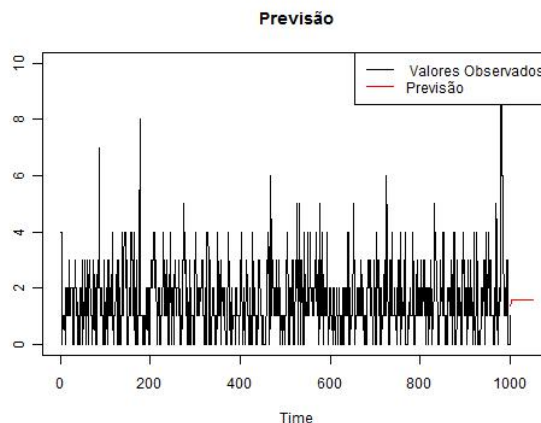


Figura 54: Previsão do processo com estimação de $\tilde{\alpha} = 0.401$, $\tilde{\lambda} = 0.942$

3.7 Análise de uma série de inteiros não negativos

Os dados reais para análise e modelação são os registos do número de óbitos, por Município (antes designado por Concelho), de Portugal Continental, que abrangem o período de 1 de Janeiro de 1980 a 31 de Dezembro de 2007. Os dados foram tratados de forma a poderem ser agrupados por outras divisões

geográficas de Portugal Continental, o número de divisões administrativas no Continente está resumida no quadro seguinte:

Divisão Administrativa	Número
Município (exemplo: Borba)	278
Distrito (exemplo: Évora)	18
NUT3 (exemplo: Alentejo Central)	30
NUT2 (exemplo: Alentejo)	5
<small>NUT: Nomenclatura de Unidade Territorial para fins estatísticos</small>	

Quadro 3.1: Divisões Administrativas de Portugal Continental

Os dados originais estavam divididos em dois ficheiros, que tiveram de ser tratados e integrados numa única base de dados. Devido a discrepâncias de identificação de Municípios das Regiões Autónomas dos Açores e Madeira, foram eliminados esses registos, ficando para análise só os dados relativos ao Continente.

Para se ter uma ideia do volume de dados tratados, no quadro seguinte apresentam-se, de forma resumida, o número de registos em análise por Divisão Administrativa.

Divisão Administrativa	Número de Registos
Município	1 317 663 (em coluna)
Distrito (exemplo: Évora)	10 227 (em tabela _(18,10227))
NUT3 (exemplo: Alentejo Central)	10 227 (em tabela _(30,10227))
NUT2 (exemplo: Alentejo)	10 227 (em tabela _(5,10227))

Quadro 3.2: Número de registos por Divisão Administrativa

Em todas as agregações de dados realizadas existem *missing values*, com exceção da agregação realizada para NUT2.

A primeira verificação a realizar nos diferentes apuramentos de dados, foi certificar que existiam 10 227 registos, que é o número de dias entre a data de início e de fim dos dados. Tal não acontecia quando se apuravam Municípios, Distritos e NUT3, pois há dias em que não ocorreram óbitos, o que produzia séries de dimensões diferentes.

Um apuramento para o Município de Évora tem 7 874 registos, o que significa que em 2 353 dias, durante o período em estudo, não há ocorrência de óbitos. Enquanto um apuramento para o Distrito de Évora revela a existência de 10 189 registos, o que revela que em 38 dias não houve ocorrência de óbitos. Ao nível de NUT3 o apuramento para Trás-os-Montes tem 10 223 registos, pelo que em 4 dias não há registo de óbitos.

A construção de tabelas cruzadas de todos os Municípios, Distritos e NUT's permitiu ultrapassar esse problema, e assim apurar as séries com *missing values*, que nos apuramentos individuais não existiam.

Na análise de séries temporais não pode haver *missing values*, têm de ser tratados. O R através da livraria *TimeSeries* permite a interpolação linear de *missing values* através do comando,

```
y<-interpNA(y, method="linear") # interpolação linear da variável y
```

que, no entanto, não se mostrou eficaz para as séries utilizadas, pois produzia valores que não eram utilizáveis em comandos como, por exemplo, `acf()` e `pacf()`. Não foi possível identificar a causa do problema. De forma a preencher os *missing values* utilizou-se o SPSS com interpolação linear das observações adjacentes.

Como o preenchimento de *missing values* pode, de alguma forma, provocar alguma variação nos dados, optou-se por utilizar nas análises os dados onde não existiam *missing values*, ou seja, os dados agregados por NUT2.

A análise da totalidade dos dados fica reservada para trabalho futuro, dado o seu volume e o tempo que essas análises consumirão.

No quadro 3.3 apresentam-se o número de séries possíveis de estudo, provenientes dos dados existentes.

As análises realizadas e a metodologia aplicada ao R apresentada, até ao presente momento são para séries individuais, com o desenvolvimento de código passo a passo. Para estes dados reais é necessário o desenvolvimento de código de R através de rotinas e ciclos bem testados e experimentados, através

de funções, para que sejam possíveis as análises e apresentação de resultados, de outra forma, é tarefa demasiadamente ...

	Número de Séries
Municípios	278
Distritos	18
NUT3	30
NUT2	5
Portugal Continental	1
Total	332

Quadro 3.3: Número de séries por Divisão Administrativa

Embora o código tenha sido desenvolvido numa óptica passo a passo, houve a preocupação de utilizar vectores com nomes, valores e construção de expressões para que mais tarde fossem integradas em rotinas de processamento o mais automatizado possível, com as devidas adaptações necessárias, o código foi pensado e estruturado com esse objectivo.

3.7.1 Óbitos por NUT2

A primeira tarefa é a leitura dos dados para o workspace do R,

```
Dados <- read.table("obitos_Nuts_2.csv", header=TRUE, sep=";", fill=TRUE) # atribuir dados a uma variável
```

Segue-se a representação gráfica dos dados. Os dados são constituídos por uma tabela com 6 colunas (data, Norte, Centro, LVT, Alentejo e Algarve) em que a primeira coluna tem efeito de controlo e não se vai ser alvo de representação gráfica. As colunas da posição 2 à 6 são as que contêm os dados, e que vão ser analisadas e representadas graficamente.

Para tratar grandes volumes de informação como é o caso, a abordagem mais adequada é através de processos recursivos, sem que sejam realizadas as

tarefas uma a uma, deste modo, apresentam-se algumas técnicas de estabelecimento de nomes para utilização em ciclo *for*,

```
Dados[0,] # visualiza cabeçalhos dos dados carregados
[1] data Norte Centro LVT Alentejo Algarve #
Names <-names(Dados) # cria variável com os cabeçalhos lidos do ficheiro
Names # visualiza dados da variável Names
[1] "data" "Norte" "Centro" "LVT" "Alentejo" "Algarve"
t <- Dados # mudança de variável para tornar a construção de ciclos
# mais fácil
names1 <- c('Cron', 'Histograma', 'FAC', 'FACP', 'de Óbitos', 'de dif.') # cria vector com nomes para títulos de gráficos
```

O cronograma, histograma, função de autocorrelação e função de autocorrelação parcial de cada uma das séries são realizadas via ciclo *for*. O resultado do código que se segue produz um ficheiro *pdf*, o que permite a visualização dos gráficos dos dados depois de realizar as operações necessárias. No entanto, de acordo com o que se tem vindo a explicar neste trabalho, serão apresentadas as figuras tal como se se estivesse a efectuar a análise passo a passo.

```
pdf(file="lista.pdf") # cria ficheiro pdf com o cronograma, histograma, FAC e FACP das 5 séries em análise
par(mfrow = c(2,2)) # cria 4 figuras por página, ou seja, o cronograma, histograma, FAC e FACP de
# cada uma das séries ficam na mesma página
i <-numeric() # inicialização do índice do ciclo
for (i in 2:length(Names)) {Nomes_coluna <- # inicialização do ciclo
j <-1 # inicialização de contador para names1
plot.ts(t[,i], main = paste(names1[j],names1[5],"- ",Names[i])) # cronograma
j <-j+1 # contador para names1
hist(t[,i], main = paste(names1[j],names1[5],"- ",Names[i]) ) # histograma
j <-j+1 # contador para names1
acf(t[,i], main = paste(names1[j],names1[5],"- ",Names[i])) # FAC
j <-j+1 # contador para names1
pacf(t[,i], main = paste(names1[j],names1[5],"- ",Names[i])) # FACP
j <-j+1 # contador para names1
i <- i+1 # contador do ciclo for
} # fim do ciclo for
dev.off() # fecha pdf

shell ("lista.pdf") # abre pdf para visualização dos resultados
# produzidos pelo ciclo for
```

Se seguida mostram-se os gráficos produzidos pelo código acima reproduzido,

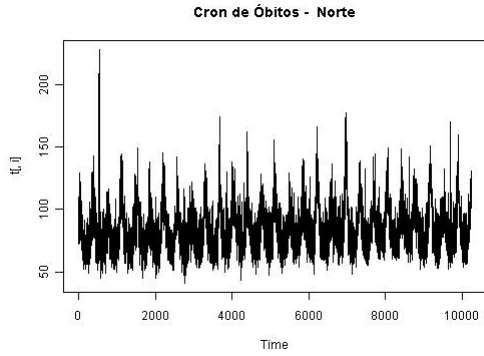


Figura 55: Cronograma de óbitos – Região Norte

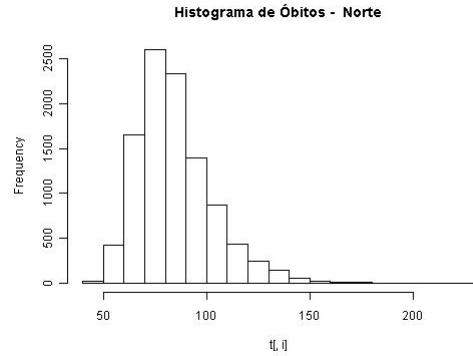


Figura 56: Histograma de óbitos – Região Norte

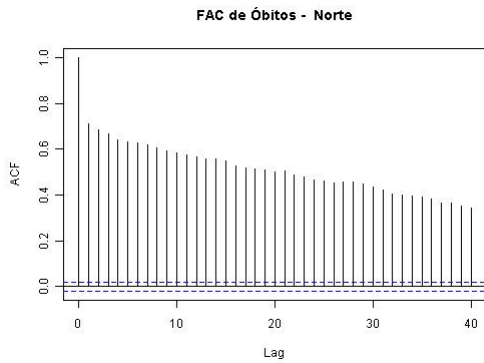


Figura 57: FAC de óbitos – Região Norte

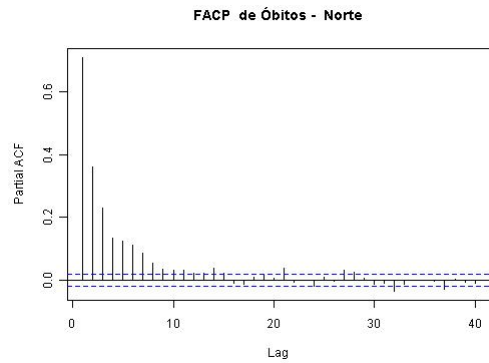


Figura 58: FACP de óbitos – Região Norte

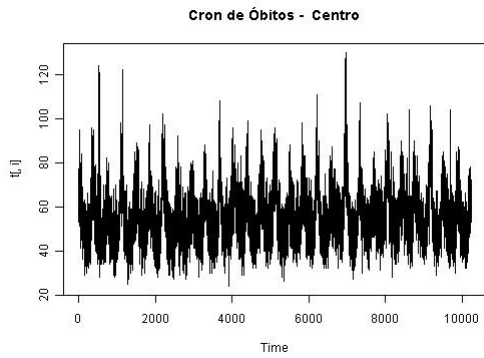


Figura 59: Cronograma de óbitos – Região Centro

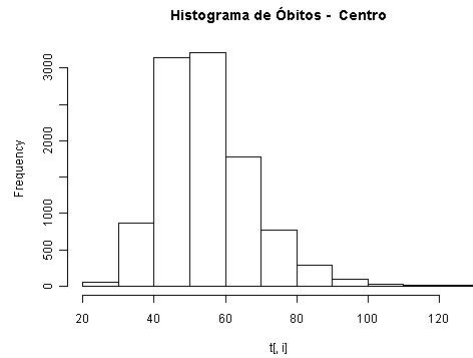


Figura 60: Histograma de óbitos – Região Centro

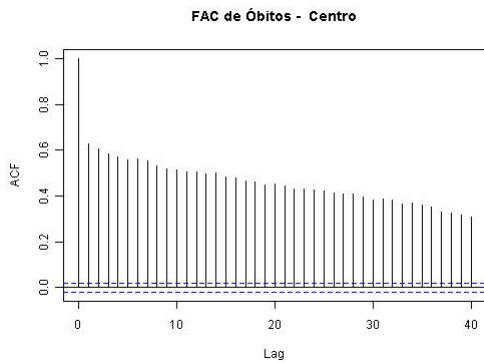


Figura 61: FAC de óbitos – Região Centro

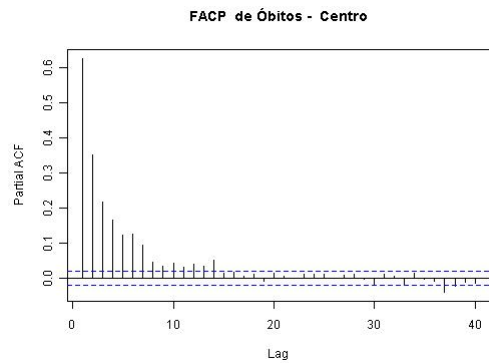


Figura 62: FACP de óbitos – Região Centro

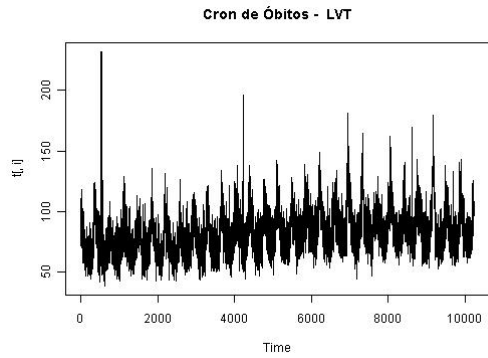


Figura 63: Cronograma de óbitos – Região Lisboa e Vale do Tejo

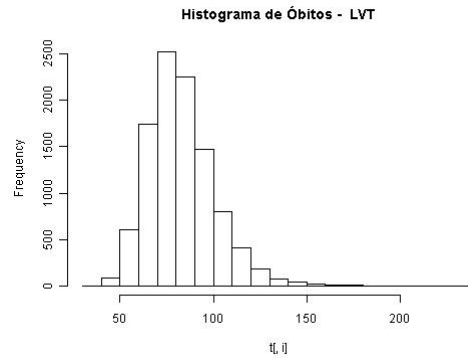


Figura 64: Histograma de óbitos – Região Lisboa e Vale do Tejo

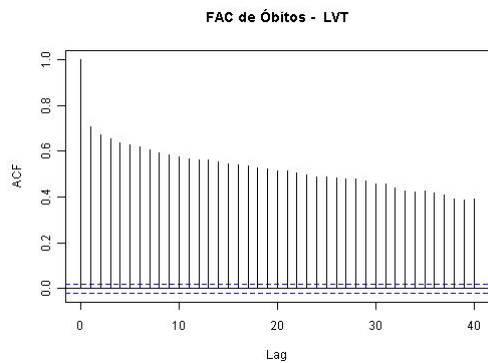


Figura 65: FAC de óbitos – Região Lisboa e Vale do Tejo

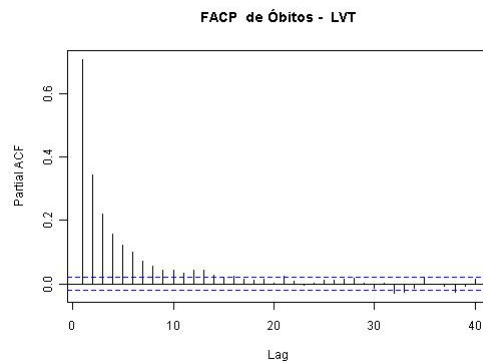


Figura 66: FACP de óbitos – Região Lisboa e Vale do Tejo

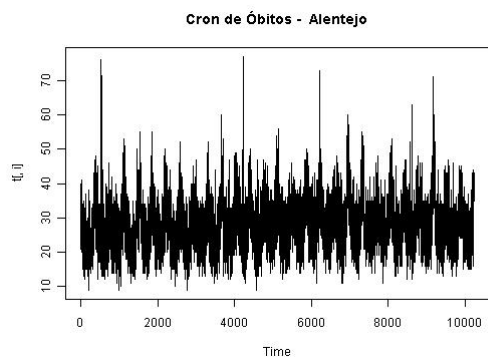


Figura 67: Cronograma de óbitos – Região do Alentejo

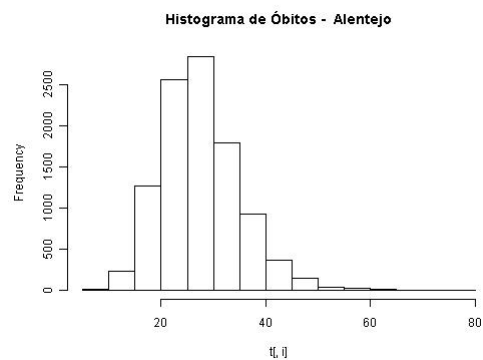


Figura 68: Histograma de óbitos – Região do Alentejo

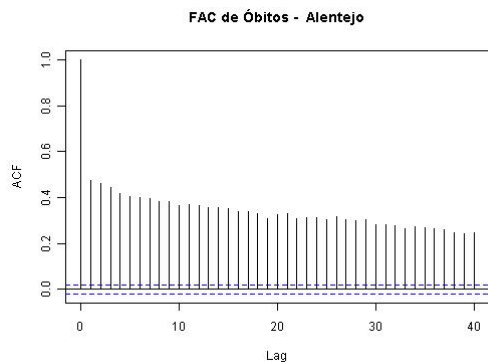


Figura 69: FAC de óbitos – Região do Alentejo

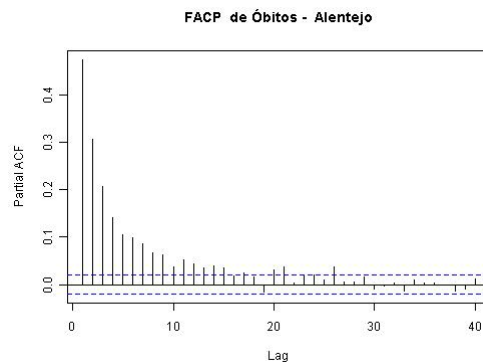


Figura 70: FACP de óbitos – Região do Alentejo

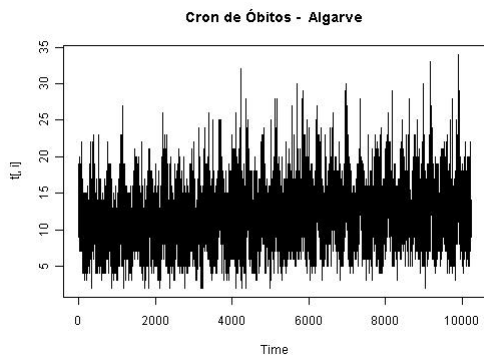


Figura 71: Cronograma de óbitos – Região do Algarve

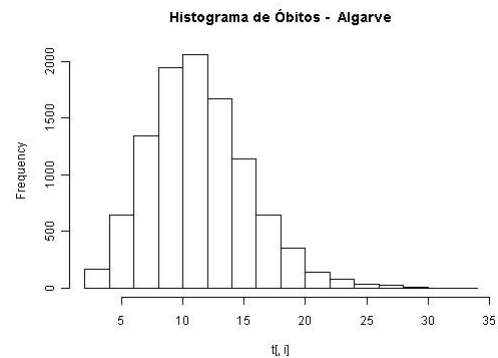


Figura 72: Histograma de óbitos – Região do Algarve

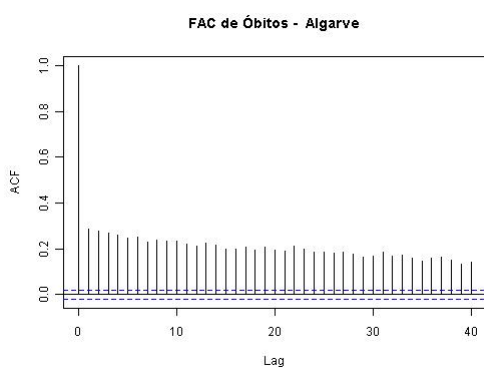


Figura 73: FAC de óbitos – Região do Algarve

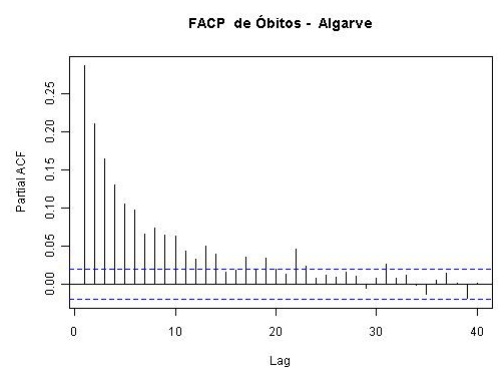


Figura 74: FACP de óbitos – Região do Algarve

O Cronograma, o histograma, a FAC e FACP de cada uma das séries denotam comportamentos semelhantes, onde se destacam os comportamentos das FAC e FACP como sendo não estacionárias em média e/ou variância.

Como nenhuma das séries é estacionária, vai agora apresentar-se código para aplicar uma das transformações aconselhadas nestes casos, a diferenciação de ordem 1.

```
s <- Dados # cria variável
s <- ts(Dados) # transforma s num objecto do tipo ts
t <- diff(s) # t é a variável com aplicação de diferenças de
# ordem 1
```

Após aplicação da diferença de ordem 1 na tentativa de estacionarizar as séries, passa-se à fase do código para se realizar a representação gráfica, que é análogo ao anterior,

```

pdf(file="lista4.pdf") # cria pdf com os resultados dos 4 tipos de gráficos das 5 séries
par(mfrow = c(2,2)) # cria 4 figuras por página, ou seja, o cronograma, histograma, FAC e
# FACP de cada uma das séries ficam na mesma página

i <- numeric() # inicialização do índice do ciclo
for (i in 2:length(Nomes)) {Nomes_coluna <- # inicialização do ciclo
j <- 1 # inicialização de contador para names1
plot.ts(t[,i], main = paste(names1[j],names1[6],names1[5],"-",Names[i])) # cronograma
j <- j+1 # contador para names1
hist(t[,i], main = paste(names1[j],names1[6],names1[5],"-",Names[i]) ) # histograma
j <- j+1 # contador para names1
acf(t[,i], main = paste(names1[j],names1[6],names1[5],"-",Names[i])) # FAC
j <- j+1 # contador para names1
pacf(t[,i], main = paste(names1[j],names1[6],names1[5],"-",Names[i])) # FACP
j <- j+1 # contador para names1
i <- i+1 # contador do ciclo for
} # fim do ciclo for
dev.off() # fecha pdf
shell ("lista.pdf") # abre pdf para visualização dos resultados
# produzidos pelo ciclo for

```

Com os resultados que de seguida se apresentam,

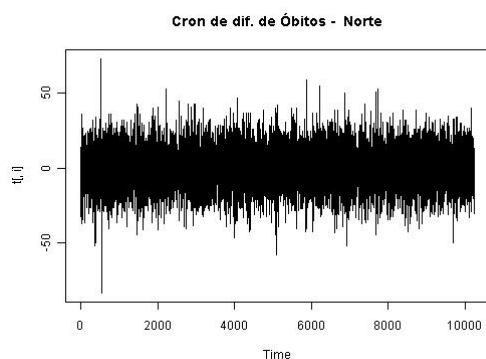


Figura 75: Cronograma da diferença de ordem 1 de óbitos – Região Norte

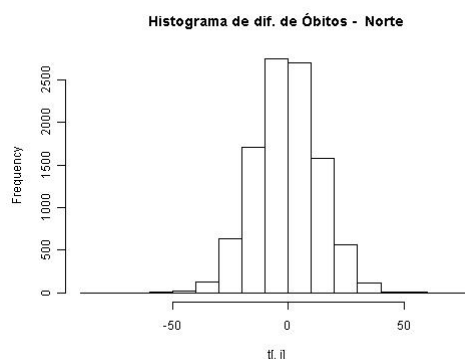


Figura 76: Histograma da diferença de ordem 1 de óbitos – Região Norte

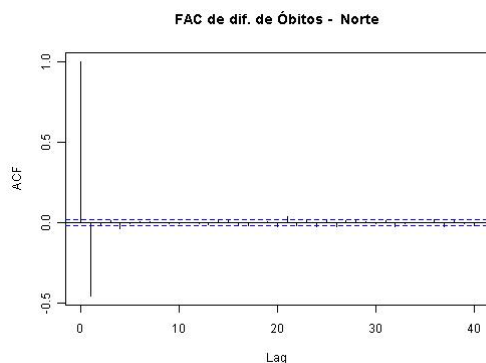


Figura 77: FAC da diferença de ordem 1 de óbitos – Região Norte

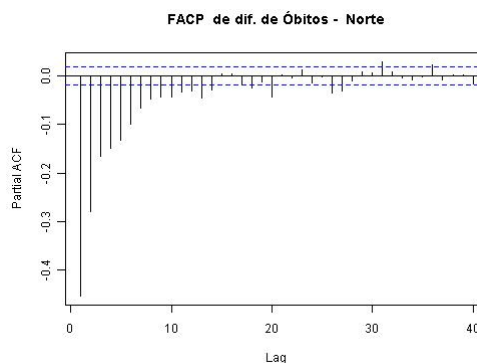


Figura 78: FACP da diferença de ordem 1 de óbitos – Região Norte

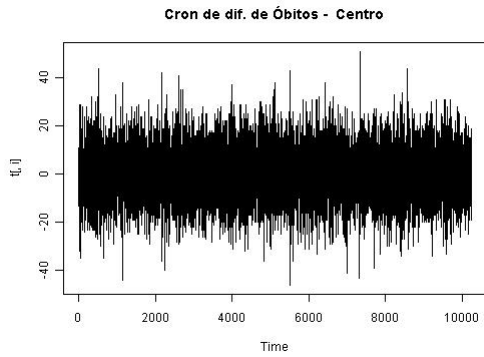


Figura 79: Cronograma da diferença de ordem 1 de óbitos – Região Centro

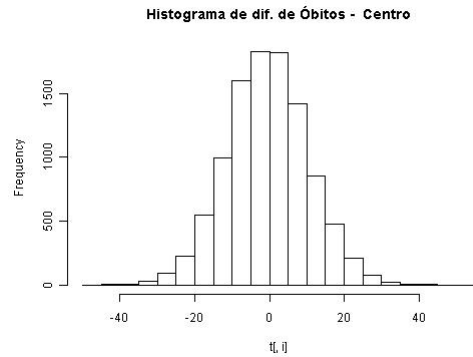


Figura 80: Histograma da diferença de ordem 1 de óbitos – Região Centro

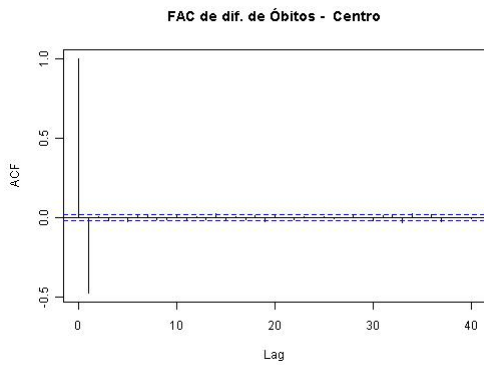


Figura 81: FAC da diferença de ordem 1 de óbitos – Região Centro

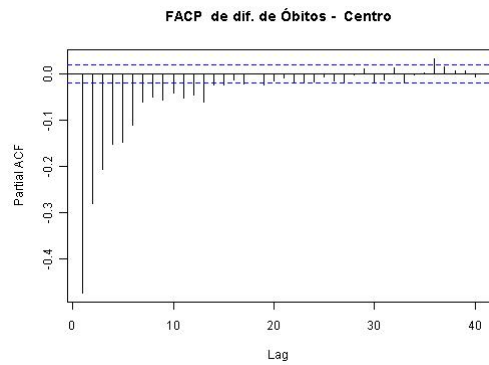


Figura 82: FACP da diferença de ordem 1 de óbitos – Região Centro

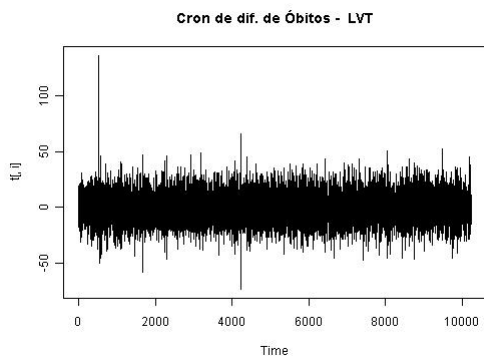


Figura 83: Cronograma da diferença de ordem 1 de óbitos – Região de Lisboa e Vale do Tejo

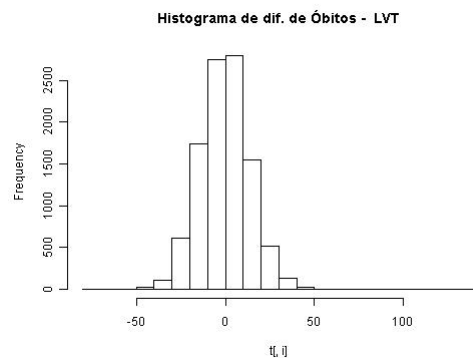


Figura 84: Histograma da diferença de ordem 1 de óbitos – Região de Lisboa e Vale do Tejo

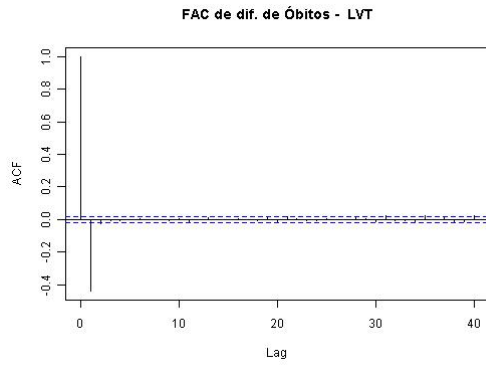


Figura 85: FAC da diferença de ordem 1 de óbitos – Região de Lisboa e Vale do Tejo

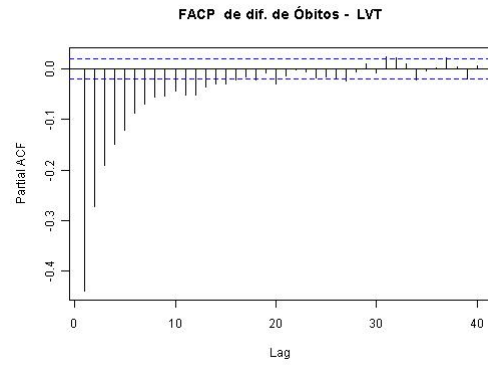


Figura 86: FACP da diferença de ordem 1 de óbitos – Região de Lisboa e Vale do Tejo

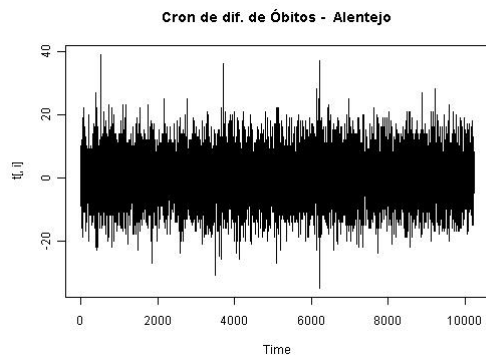


Figura 87: Cronograma da diferença de ordem 1 de óbitos – Região do Alentejo

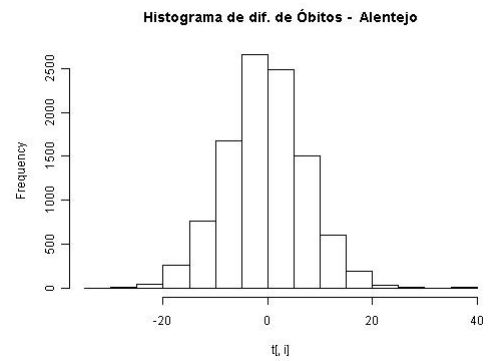


Figura 88: da diferença de ordem 1 de óbitos – Região do Alentejo

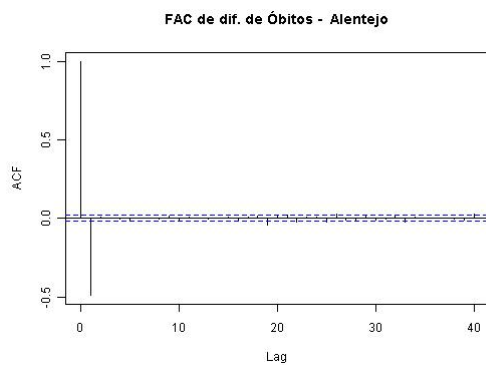


Figura 89: FAC da diferença de ordem 1 de óbitos – Região do Alentejo

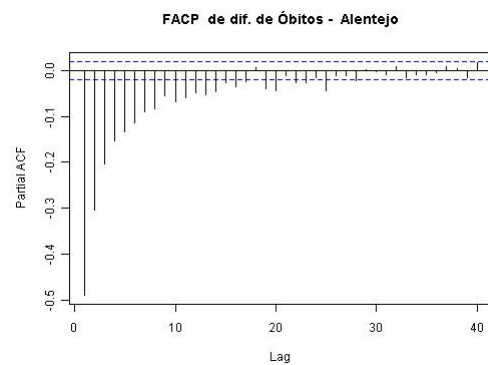


Figura 90: FACP da diferença de ordem 1 de óbitos – Região do Alentejo

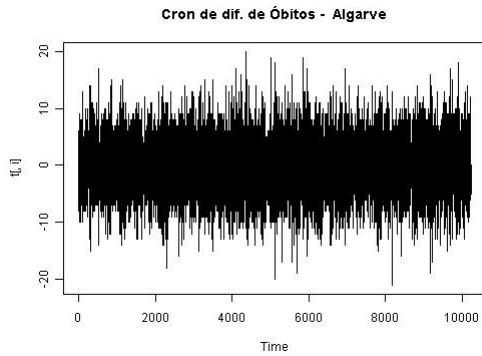


Figura 91: Cronograma da diferença de ordem 1 de óbitos – Região do Algarve

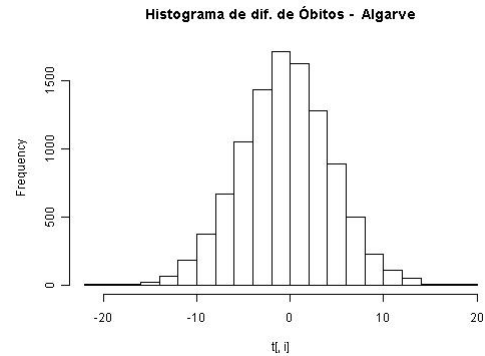


Figura 92: Histograma da diferença de ordem 1 de óbitos – Região do Algarve

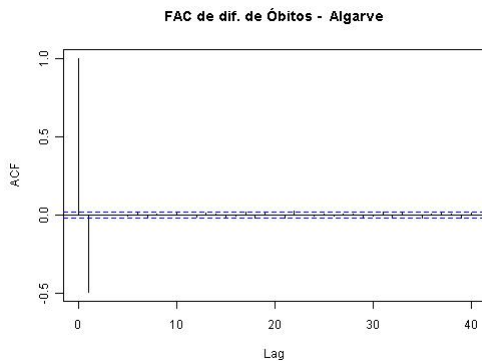


Figura 93: FAC da diferença de ordem 1 de óbitos – Região do Algarve

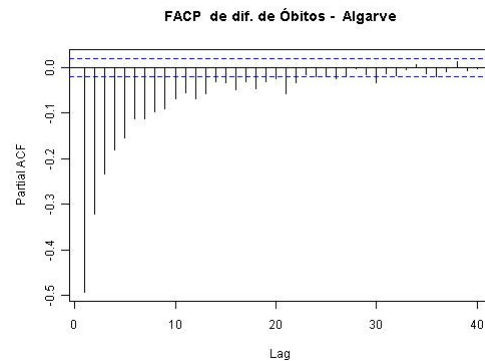


Figura 94: FACP da diferença de ordem 1 de óbitos – Região do Algarve

Todas as 5 séries têm comportamentos idênticos ao nível dos 4 gráficos. Mesmo ignorando que cada uma das séries possa deixar de ser inteira positiva, o que não se revelaria um problema, dado que se podiam aplicar mudanças de escala. Contudo, a FAC e FACP de cada uma das séries transformadas conduzem-nos a um processo do tipo MA(1). O que inviabiliza a aplicação de modelos do tipo INAR(1) a estas séries.

Foram aplicadas as mesmas técnicas a séries resultantes de outro tipo de agregação de dados, que não por NUT2, e os resultados foram semelhantes.

Capítulo 4

Conclusão

As motivações iniciais que nos levam a abraçar causas, projectos, etc, com o decorrer do tempo, o conseqüente amadurecimento provocado pelas diferentes realidades diárias, os inevitáveis choques exógenos obrigam a que se avalie, decida, reajuste, reformule e se siga em frente. Dificilmente o percurso original traçado, é o percurso percorrido, a paisagem, essa está em constante movimento, muda ao sabor das estações do ano, e por mais que se tente, em dois instantes separados por um determinado intervalo de tempo¹ não se repete.

O objectivo inicial deste trabalho consistia em abordar processos do tipo AR e por analogia prosseguir para processos INAR(1). A bagagem transportada desde o início é a motivação numa motorização de 3.0 litros, alimentada a R. As sempre presentes oscilações dos preços das matérias primas, ora provocadas por especulações financeiras, nem sempre possíveis de análise através de séries temporais, com modelos *Arch*, *Garch*, aliadas à desmesurada ganância humana de poder, fizeram com que existissem períodos, de não laboração, ora por falta de entrega de matérias primas, ora por greve dos trabalhadores instigados por sindicatos aversos à mudança, férreos defensores dos direitos adquiridos, num mundo em verdadeira mudança a escalas nunca vistas e em latitudes desconhecidas. Pelo caminho teve de se consumir alguma bagagem para manter o ânimo e possibilitar o prosseguimento da viagem.

O passeio pelos modelos de *Box* e *Jenkins*, foi assim, mais demorado, o combustível consumido foi muito mais do que o que inicialmente estava previsto, a introdução de código, devidamente comentado, e os diferentes casos analisados consumiram tempo e recursos que não estavam inicialmente previstos, a derrapagem no orçamento inicial é desta forma justificada. Os anexos são uma pequena parte palpável destas contingências do percurso, embora não reflectam a sua totalidade.

A derrapagem financeira no orçamento foi necessária e acabou por se tornar vital para o desenvolvimento do trabalho, serviu para desenvolver conhecimento e formas de aplicação, que, quer se queira quer não, era inevitável, ou não seria possível concluir o que se pretendia, chegar à simulação de modelos INAR(1).

Por desenvolver e implementar ficaram alguns métodos de estimação, de execução mais difícil e trabalhosa, que irão demorar o seu tempo a desenvolver, pois é necessário assimilar e treinar técnicas necessárias para métodos de mínimos quadrados e máxima verosimilhança, por exemplo, sob pena de um motor sobrealimentado poder queimar a cabeça do motor.

A continuação da viagem inclui a análise das séries de inteiros com as agregações possíveis, e já referidas anteriormente, numa primeira fase utilizar-se-à a interpolação linear do *SPSS*, e recorrendo aos arredondamentos que o *Excel* realiza. Terão, obrigatoriamente, de se realizar ajustes e calibrações no código, para que os automatismos funcionem, de outra forma não será possível prosseguir viagem, sob o risco de se ficar preso num ciclo sem fim.

Outro dos desafios a atingir, ainda com longo caminho a percorrer, é a generalização, através dos vários métodos de estimação, de rotinas para ordem p . Até ao momento não se conhece a sua existência.

A viagem continua, tem de se abdicar de muita coisa, trabalho árduo é essencial. A perseverança é a chave, a adaptabilidade a ambientes, culturas e o *brainstorming* com outros viajantes indicarão o melhor caminho para continuar...

¹ Em linguagem matemática – “num dado intervalo de tempo fixo”

Anexos

Anexo I

Exemplos clássicos: Ruído Branco e Passeio Aleatório

Neste anexo, através de dois exemplos clássicos, utiliza-se o R para exemplificar de forma bastante simples como executar alguns comandos elementares, bem como os primeiros passos a realizar numa situação conhecida de não estacionariedade - passeio aleatório.

Ruído Branco – *White noise* – Processo Estacionário

Para simular um ruído branco, geram-se 250 valores de uma variável aleatória i.i.d, o que é equivalente a gerar uma sucessão de ruído branco com dimensão de 250 observações.

```
set.seed(1) # fixa-se a semente de geração de números aleatórios
w <- rnorm(250) # 250 números gerados por uma distribuição normal
plot(w, type = "l") # representação gráfica
abline(h= mean(w), col = "red") # representação da média (a vermelho)
```

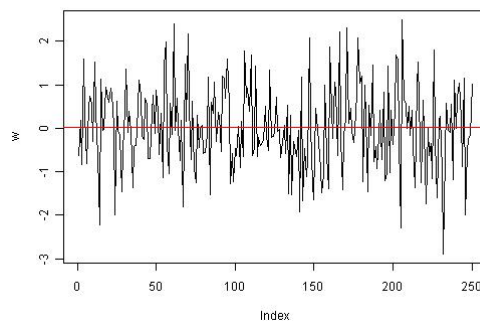


Figura A.I.1 – representação de um ruído branco

As FAC e FACP são geradas pelo código seguinte,

```
par(mfrow = c(2,1)) # divisão da janela gráfica em 2 linhas por 1 coluna
acf(w) # função de autocorrelação
pacf(w) # função de autocorrelação parcial
```

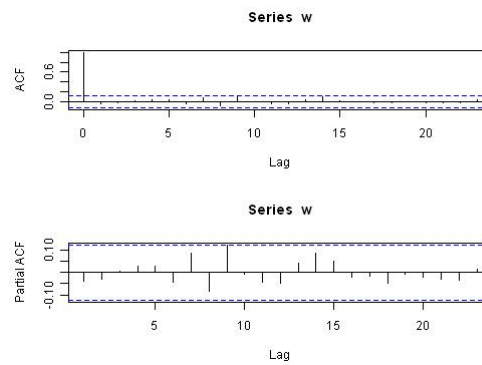


Figura A.1.2 – Função de autocorrelação (FAC) e função de autocorrelação parcial (FACP) do ruído branco
O ruído branco é um processo estacionário, a sua FAC e FACP são típicas de processos estacionários.

Passeio Aleatório – *Random Walks* – Processo não estacionário

Seja $\{X_t\}$ uma sucessão cronológica. Então $\{X_t\}$ é um passeio aleatório se

$$X_t = X_{t-1} + W_t,$$

onde $\{W_t\}$ é uma sucessão de ruído branco. Realizando substituições recursivas, dado que $X_{t-1} = X_{t-2} + W_{t-1}$, e $X_{t-2} = X_{t-3} + W_{t-2}$, e assim sucessivamente, no que é conhecido pela utilização do operador de atraso, obtem-se:

$$X_t = W_t + W_{t-1} + W_{t-2} + \dots$$

Na prática, a sucessão não será infinita, mas será iniciada algures no tempo no instante $t = 1$. Assim,

$$X_t = W_1 + W_2 + \dots + W_t$$

Vamos então agora simular um passeio aleatório para X ,

```
set.seed(2)                # ao fixar a semente garante-se que se podem reproduzir os valores
x <- w <- rnorm(1000)      # x e w são v.a. de uma dist. Normal de média nula e desvio-padrão 1
for (t in 2:1000) x[t] <- x[t - 1] + w[t]    #  $X_t = X_{t-1} + \varepsilon_t$ 
plot(x, type = "l", xlab = "Tempo", ylab = "Passeio Aleatório", main = "Trajectória de Passeio Aleatório")
```

A segunda linha de comando coloca ruído branco em w e simultaneamente inicializa x através de um ciclo (t inicia-se em 2, porque 1 é utilizado na observação de ordem menos 1 na construção do ciclo).

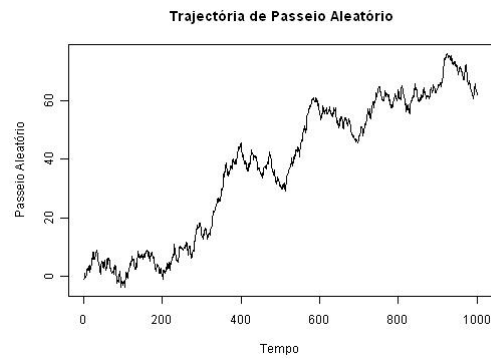


Figura A.I.3 – Gráfico da simulação de um passeio aleatório.

A sucessão exibe uma clara tendência crescente, tal é puramente estocástico e deve-se à elevada correlação (é um exemplo típico apresentado na literatura de processo não estacionário em tendência).

As FAC e FACP obtêm-se da seguinte forma.

```
par(mfrow = c(2,1)) # divisão do gráfico em 2 linhas por uma coluna
acf(x, main= "Função de Autocorrelação") # função de autocorrelação
pacf(x,main= "Função de Autocorrelação Parcial") # função de autocorrelação parcial
```

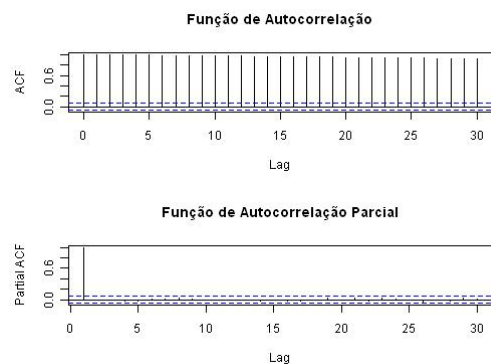


Figura A.I.4 FAC e FACP do passeio aleatório gerado.

Neste caso nem era necessário representar a FACP, pois através do cronograma da figura (A.I.3) e da FAC identifica-se a elevada correlação entre observações. O decaimento muito lento que se vê na FAC é identificativo da elevada correlação.

É necessário estabilizar uma sucessão com este comportamento, pelo que se aplica a diferenciação

```
plot(diff(x), type = "l", xlab= "Tempo", ylab="Passeio Aleatório", main=" Cronograma após diferenciação de ordem 1")
acf(diff(x), main= "Função de Autocorrelação")
```

Note-se que estamos a aplicar diferenciação à sucessão, $\mathit{diff}(x)$, e só depois é que se aplicam os comandos para realizar o cronograma, $\mathit{plot}(\mathit{diff}(x))$, e a função de autocorreção, $\mathit{acf}(\mathit{diff}(x))$.

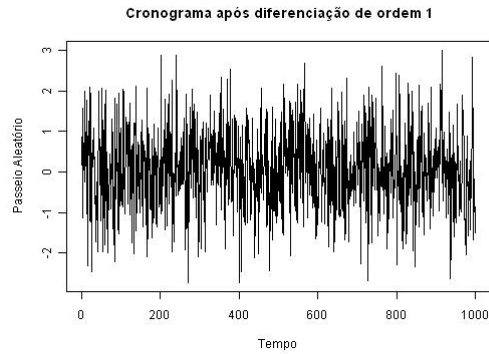


Figura A.I.5 Cronograma passeio aleatório gerado após aplicação de diferença de primeira ordem.

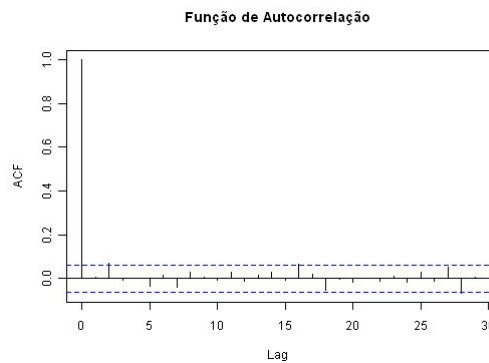


Figura A.I.6 FAC do passeio aleatório gerado após aplicação da diferenciação de primeira ordem.

Como se pode verificar na figura A.I.5 não existe um padrão óbvio no cronograma. Em relação ao correlograma (representação da FAC, fig A.I.6) existem dois valores estatisticamente significativos. Estes valores significativos podem ser ignorados, porque em termos de magnitude são pequenos e espera-se que cerca de 5% dos valores sejam significativos, mesmo quando todos os outros são nulos ou quase nulos. Assim, e tal como esperado, há evidência estatística de que se trata de um passeio aleatório, pois é resultado conhecido que a diferenciação de um passeio aleatório é ruído branco.

Anexo II

Comandos para simular e estimar um processo

Como o **R** é desenvolvido em comunidade (cada utilizador é um programador em potência) funções que já fazem parte de determinado *package*, são incluídas e melhoradas em novos *packages*, o que leva a que em alguns casos exista mais do que um comando para o mesmo resultado, até incluídos no mesmo *package*. Assim, existem dois comandos que nos permitem estimar realizações de sucessões cronológicas no **R**:

- I. **ar** → permite somente estimar processos, auto-regressivos.
- II. **arima** → permite estimar processos ARIMA, o que inclui, portanto, a estimação de processos autoregressivos e de médias móveis.

Em cada um dos casos seguintes, sabemos de antemão o que estamos a gerar, e como tal, não vamos fazer a representação das FAC e FACP para intuir sobre a ordem do modelo passando logo à fase da estimação.

Simulação de AR(1) com estimação pela função `ar()`

Quando se pretendem gerar números aleatórios, o R permite que se fixe a semente geradora. Tal permite que sempre que se repita o processo se possa repetir a mesma análise. Quando a semente não é fixada, cada geração escolhe aleatoriamente uma semente, facto que não permite voltar a repetir o processo com os mesmo valores. Em todos os casos apresentados, de aqui em diante, vamos sempre fixar a semente.

Vamos gerar o seguinte processo.

$$X_t = 0.4 \times X_{t-1} + \varepsilon_t$$

```
set.seed(1)           # estabelece-se o valor de arranque da seed
n<-1000              # estabelece-se a dim da amostra
x <- w <- rnorm(n)    # inicialização do processo
?rnorm()             # saber as funcionalidades de rnorm
for (t in 2:n) x[t] <- 0.4 * x[t - 1] + w[t] # ciclo para construir o Processo AR(1)
plot(x, type = "l", main=(expression(AR(1)~--phi[1]==+.4)), col = "blue", cex.main =1)
```

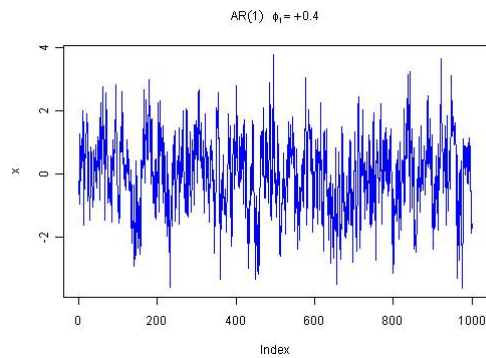


Figura A.II.1 Cronograma da série gerada - AR(1)

A estimativa de ϕ_1 será $\tilde{\phi}_1 = 0.3472$ como se pode ver no código abaixo

```
> x.ar.yw <- ar(x, method = "yw")      # estimação de parâmetro - método de Yule Walker
>?ar()                                # ver opções e funcionalidades de ar
> x.ar.yw$order                        # ordem do parâmetro estimado - método Yule-Walker
[1] 1                                  # a ordem de phi é 1
> x.ar.yw$ar                           # valor do coeficiente estimado - método Yule-Walker
[1] 0.3472397                          # valor estimado de phi
```

Simulação de AR(2) com estimação pela função ar()

Vamos gerar o seguinte processo

$$X_t = 0.4 \times X_{t-1} + 0.3 \times X_{t-2} + \varepsilon_t$$

```
set.seed(1)                            # Estabelece-se o valor de arranque da seed
x <- w <- rnorm()                       # inicialização do processo
for (t in 3:n) x[t] <- 0.4 * x[t-1] + 0.3 * x[t-2] + w[t] # ciclo para construir o Processo AR(2)
plot(x, type = "l", main=(expression(AR(1)~phi[1]==+.4~phi[2]==+.3)), col = "blue", cex.main = 1)
```

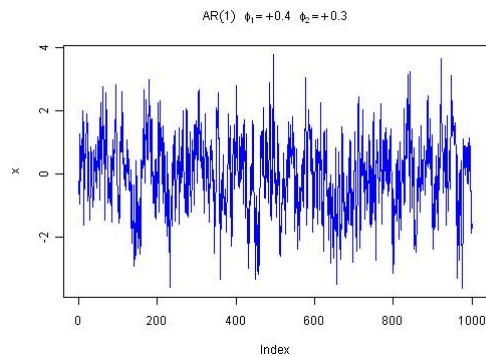


Figura A.II.2 Cronograma da série gerada - AR(2)

A estimação de parâmetros é igual ao caso anterior e $\tilde{\phi}_1 = 0.3565$, $\tilde{\phi}_2 = 0.2752$,

```
> x.ar.yw <- ar(x, method = "yw")      # estimação de parâmetro - método de Yule Walker
> x.ar.yw$order                        # ordem do parâmetro estimado - método Yule-Walker
```

```
[1] 2 # tal como esperado, 2 parâmetros, phi1 e phi2
> x.ar.yw$ar # Valores dos coeficientes estimados - método Yule-Walker
[1] 0.3565668 0.2752541
```

Simulação de AR(1) com estimação pela função arima() e simulação pela função arima.sim()

```
set.seed(1) # fixa semente
x1 = arima.sim(list(order=c(1,0,0), ar=.6), n=100) # simula processo com coeficiente ar=1, diferença=0, ma=0 e phi = 0.6
?arima.sim() # ver opções de arima.sim()
plot(x1, main=(expression(AR(1)~phi==+.6))) # cronograma
```

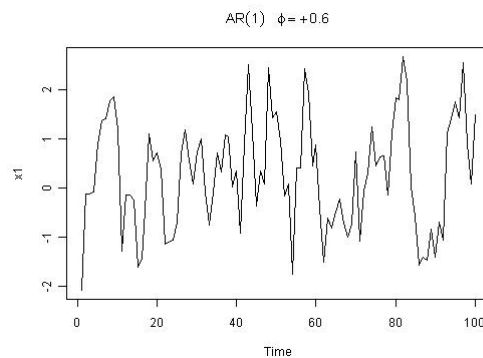


Figura A.II.3 Cronograma da série gerada - AR(1)

A estimativa do parâmetro ϕ_1 é $\tilde{\phi}_1 = 0.5924$, como se pode ver abaixo:

```
> (x1.fit = arima(x1, order = c(1, 0, 0))) # ajustamento do modelo e resultados
```

Call:

```
arima(x = x1, order = c(1, 0, 0))
```

Coefficients:

```
ar1 intercept
0.5924 0.2674
s.e. 0.0828 0.2178
```

```
sigma^2 estimated as 0.8103: log likelihood = -131.59, aic = 269.18
```

O processo gerado não inclui constante e o processo de estimação calcula-a. Para verificar se esta constante fica no modelo deve-se testar através da função confint() da livreria *Tseries*.

```
> (confint(x1.fit))
2.5 % 97.5 %
ar1 0.4301336 0.7547300
intercept -0.1594400 0.6942619
```

Como o intervalo de confiança estimado para a constante inclui o zero, a constante deve ser retirada do modelo, o que faz todo o sentido, pois a simulação realizada não incluiu constante. O comando correcto para a estimação deve então incluir que não se pretende estimar a constante. Nos exemplos seguintes utiliza-se o comando para não estimar a constante.

```
(x2.fit = arima(x1, order = c(1, 0, 0), include.mean=FALSE))
```

Call:

```
arima(x = x1, order = c(1, 0, 0), include.mean = FALSE)
```

Coefficients:

```
ar1
0.6178
s.e. 0.0804
```

```
sigma^2 estimated as 0.8213: log likelihood = -132.29, aic = 268.58
```

Simulação de AR(2) com estimação pela função `arima()` e simulação pela função `arima.sim()`

```
set.seed(1)
```

```
x3 = arima.sim(list(order=c(2,0,0), ar=c(1,-.9)), n=100) #phi1=1 e phi2=0.9
```

```
plot(x3, main=(expression(AR(2)~phi[1]==1~phi[2]==-.9)))
```

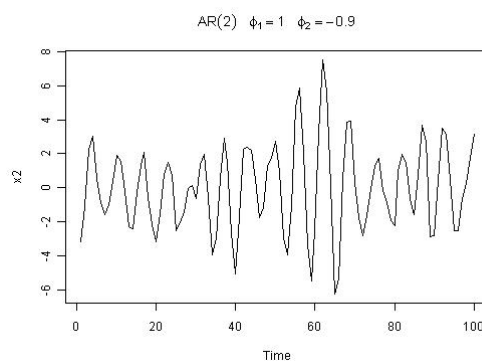


Figura A.II.4 Cronograma da série gerada - AR(2)

```
> (x3.fit = arima(x3, order = c(2, 0, 0), include.mean=FALSE)) # ajustamento do modelo e resultados
```

Call:

```
arima(x = x3, order = c(2, 0, 0), include.mean = FALSE)
```

Coefficients:

```
ar1 ar2
```

```

0.9815 -0.8812
s.e. 0.0463 0.0441

sigma^2 estimated as 1.036: log likelihood = -145.31, aic = 296.61

```

Simulação de $ma(1)$ com estimação pela função `arima()` e simulação pela função `arima.sim()`

```

set.seed(1)
x = arima.sim(list(order=c(0,0,1), ma=0.8), n=100)
plot(x, main=(expression(MA(1)~--theta[1]==0.8)))

```

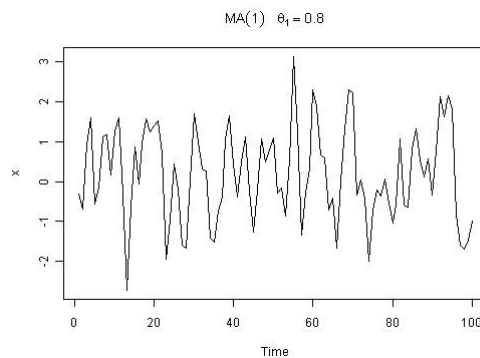


Figura A.II.5 Cronograma da série gerada - MA(1)

```

> (x.fit = arima(x, order = c(0, 0, 1), include.mean=FALSE))

Call:
arima(x = x, order = c(0, 0, 1), include.mean = FALSE)

Coefficients:
    ma1
  0.7973
s.e. 0.0658

sigma^2 estimated as 0.808: log likelihood = -131.74, aic = 267.47

```

Simulação de ARIMA(1,0,1) ou seja ARMA(1,1) com estimação pela função `arima()` e simulação pela função `arima.sim()`

```

set.seed(1)
x = arima.sim(list(order=c(1,0,1), ar=.9, ma=-.5), n=100)
plot(x, main=(expression(ARIMA(1,0,1)~--phi==.9~--theta==-.5)))

```

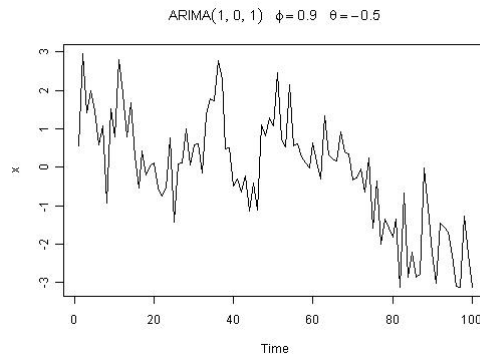


Figura A.II.6 Cronograma da série gerada - ARMA(1,1)

```
> (x.fit = arima(x, order = c(1, 0, 1), include.mean=F)) # fit the model and print the results
```

Call:

```
arima(x = x, order = c(1, 0, 1), include.mean = F)
```

Coefficients:

```
   ar1   ma1
 0.9575 -0.5488
s.e. 0.0371 0.0987
```

sigma^2 estimated as 0.8426: log likelihood = -134.01, aic = 274.02

Simulação de ARIMA(1,1,1) com estimação pela função arima() e simulação pela função arima.sim()

```
set.seed(1)
```

```
x = arima.sim(list(order=c(1,1,1), ar=.9, ma=-.5), n=200)
```

```
plot(x, main=(expression(ARIMA(1,1,1)~phi==.9~theta==-.5)))
```

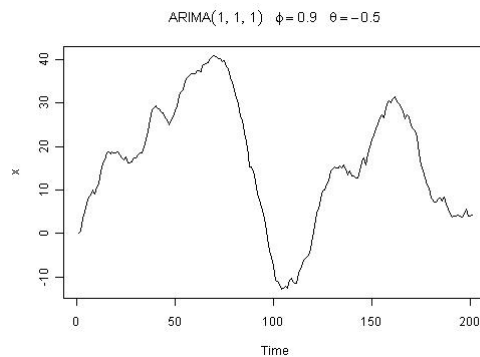


Figura A.II.7 Cronograma da série gerada - ARIMA(1,1,1)

```
> (x.fit = arima(x, order = c(1, 1, 1)))
```

Call:

```
arima(x = x, order = c(1, 1, 1))
```

Coefficients:

ar1	ma1
0.9230	-0.5609
s.e. 0.0341	0.0698

sigma^2 estimated as 0.9752: log likelihood = -281.69, aic = 569.39

Anexo III

Análise completa de AR(1) em R

III.1 Análise completa de AR(1) de média nula

Considere-se um processo AR(1) de média e desvio-padrão zero,

$$X_t = \phi_1 X_{t-1} + \varepsilon_t.$$

No quadro seguinte, estabelece-se um valor de início para a geração de números aleatórios, a dimensão da amostra é atribuída à variável n , neste caso 1000, o valor de ϕ_1 é atribuído à variável **PHI** e x e w são inicializados por uma distribuição normal de n números aleatórios de média nula e variância igual a um.

```
set.seed(1)           # estabelece-se valor de arranque da seed
n= 1000              # estabelece-se o nº da amostra
PHI <- 0.4           # definição do coeficiente AR como variável
x <- w <- rnorm(n)   # inicialização do processo
class(x)             # saber a que classe de objectos pertence o objecto x
[1] "numeric"        # resultado do comando anterior
x =ts(x)             # transforma-se o objecto x num objecto do tipo "time series"
class(x)             # saber a que classe de objectos pertence o objecto x
[1] "ts"            # mostra que x é um objecto do tipo "time series"
for (t in 2:n) x[t] <- PHI * x[t - 1] + w[t] # geração do processo AR em que PHI é definido na terceira linha
x                    # mostra os dados do processo AR(1)
x[1:3];x[998:1000]   # mostra as observações de ordem 1,2,3,998,999 e 1000
[1] -0.6264538 -0.0669382 -0.8624039 # observações de ordem 1,2,3
[1] -0.7847459 -1.7966500 -1.4159782 # observações de ordem 998,999 e 1000
```

O comando **class(x)** é de importância extrema, permite inquirir o **R** sobre a classe de objectos a que cada uma das variáveis pertence, no caso das sucessões cronológicas, as variáveis têm de ser obrigatoriamente da classe "ts" (*Time Series*) ou não será possível aplicar os diferentes comandos de *time series*.

A produção de código, independentemente da linguagem de programação utilizada, deve ser realizada de forma o mais estruturada possível, e devidamente comentada, para evitar dúvidas interpretativas, pois à medida que vai aumentando o número de linhas de código, aumenta a dificuldade em interpretar o que está escrito e surgem

com frequência dúvidas e propagam-se pequenos erros que inviabilizam o correcto desenvolvimento dos trabalhos. A quantidade significativa de gráficos produzidos suscitou a necessidade de os identificar correctamente para que não existissem dúvidas, para tal era necessário ter títulos bem explícitos.

Depois de várias experiências, infrutíferas, para colocar títulos nos diferentes gráficos gerados, pois não se conseguia inserir simbologia matemática, optou-se por criar variáveis especificamente para esse fim, a lógica utilizada é através de duas variáveis em formato de vector, *names* onde se guardam nomes e *vals* onde se guardam valores. A construção dos títulos é posteriormente assegurada à custa das funções *bquote* e *as.names*. O código utilizado não é de fácil percepção, resumidamente:

names → é formado por um vector coluna com 7 posições, de 1 a 7

vals → é formado por um vector coluna com 2 posições

expr1 → é a variável utilizada que é invocada no título de cada um dos gráficos

Assim,

```
expr1 <- bquote  ((.as.name(names[1])) ~~~~"
                 "~.(as.name(names[2]))[1]==.(vals[2]))
```

Tem como resultado:

'Simulação de Processo AR(1)' → posição um de *names*

'phi' + 1 subscrito + PHI → posição 2 de *names* (ϕ_1) que será igual à posição 2 de *vals* que é 0.4 definido no início do código, $\phi_1 = 0.4$.

(Para perceber melhor, ver título da figura A.III.1)

O código seguinte vai produzir as figuras A.III.1 a A.III.6,

```
names <- c('Simulação de Processo AR(1)', 'phi', 'Histograma', 'QQ plot', 'Caixa de Bigodes', 'FAC', 'FACP')
vals <- c("", PHI)
expr1 <- bquote  ((.as.name(names[1])) ~~~~"
                 "~.(as.name(names[2]))[1]==.(vals[2]))
expr2 <- bquote  ((.as.name(names[2]))[1]==.(vals[2]))
expr3 <- bquote  ((.as.name(names[3])) ~~~~"
                 "~.(as.name(names[2]))[1]==.(vals[2]))
expr4 <- bquote  ((.as.name(names[4])) ~~~~"
                 "~.(as.name(names[2]))[1]==.(vals[2]))
expr5 <- bquote  ((.as.name(names[5])) ~~~~"
                 "~.(as.name(names[2]))[1]==.(vals[2]))
```

```

expr6 <- bquote (.(as.name(names[6]))~~~~"
                "~.(as.name(names[2]))[1]==.(vals[2])
expr7 <- bquote (.(as.name(names[7]))~~~~"
                "~.(as.name(names[2]))[1]==.(vals[2])
expr8 <- bquote (.(as.name(names[3])) )                # fim dos nomes para títulos

plot(x, type = "l",main = expr1, col = "blue", cex.main =1)      # cronograma
abline(h=mean(x), col = "red")                                  # representação da média
legend("topright", legend = c(" Série", "Média"),col= c(4, 2), lty=c(1,1),cex=0.8)

hist(x, prob=TRUE, main = expr3,cex.main =1)                   # histograma
lines(density(x), col= "2")                                    # densidade normal no histograma
qqnorm(x,main = expr4, col= "4",cex.main =1) # normal Q-Q plot # gráfico QQ-plot
qqline(x, col = "2")                                          # linha de ajustamento normal em QQ-plot

boxplot(x,main = expr5,cex.main =1)                             # caixa de bigodes
acf(x, xlim =c(2,36),main = expr6, cex.main =1)               # FAC
pacf(x,xlim =c(2,36), main = expr7, cex.main =1)              # FACP

```

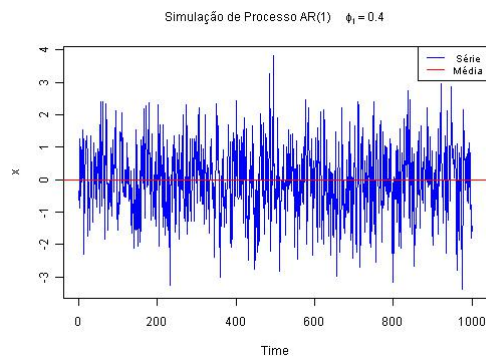


Figura A.III.1: Cronograma do processo AR(1) gerado

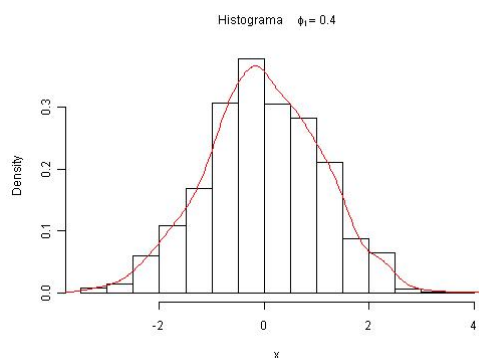


Figura A.III.2: Histograma do processo gerado com ajustamento de curva normal

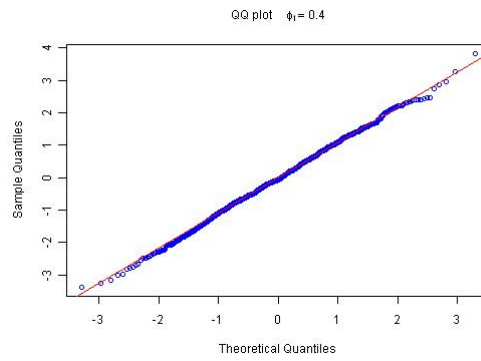


Figura A.III.3: Representação do processo gerado em papel de probabilidades (QQ plot) com ajustamento normal

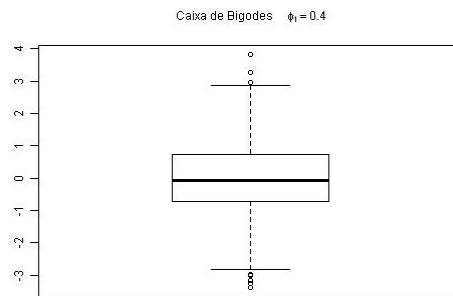


Figura A.III.4: Caixa de bigodes do processo gerado

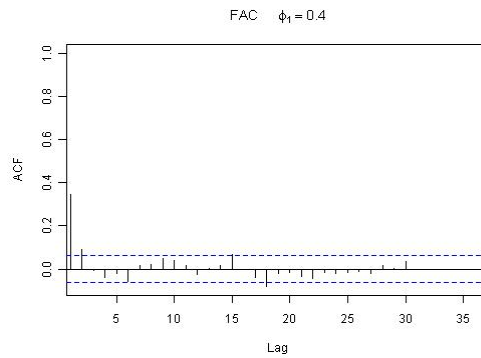


Figura A. III.5: FAC do processo gerado

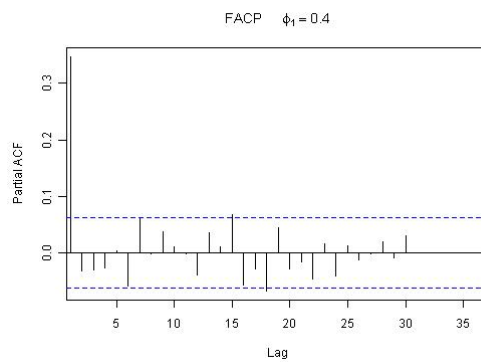


Figura A.III.6: FACP do processo gerado

Através da representação da FAC e FACP, verifica-se que o modelo adequado para se ajustar à série será AR(1), pois denotam comportamentos de acordo com os modelos teóricos de FAC e FACP de um processo AR(1).

Uma vez realizada a representação gráfica genérica, é lugar para se mostrar como se calculam algumas das estatísticas descritivas.

De modo a tornar a leitura mais simples, vamos apresentar os resultados em forma matricial. O código é o seguinte:

```
> M.Descriptives.x<-matrix(c(mean(x),sd(x), var(x)), 1,3) # definição da matriz e seus valores
> colnames (M.Descriptives.x) <- c(" Mean", " Standart Deviation", " Variance") # nomes colunas -3
> rownames (M.Descriptives.x) <- c("coeficients") # nomes da linhas
> M.Descriptives.x # apresentação de resultados
      Mean  Standart Deviation  Variance
coeficients -0.01846958      1.101999  1.214401
>
> print(round(M.Descriptives.x, digits=5)) # apresentação de resultados com 5 casas decimais
      Mean  Standart Deviation  Variance
Coefficients -0.01847      1.102  1.2144
```

A média é -.01847, muito perto de zero e o desvio-padrão (1.102) também é quase o que se estabeleceu na geração da sucessão.

Vamos passar à fase de estimação dos parâmetros do processo. Ou seja, os coeficientes do processo AR(1) - ϕ_1 .

```
x.ar.burg <- ar(x, method = "burg") # estimação de parâmetro - método de Burg
x.ar.yw <- ar(x, method = "yw") # estimação de parâmetro - método de Yule Walker
x.ar.mle <- ar(x, method = "mle") # estimação de parâmetro - método de MLE
```

Uma vez estimados os parâmetros, vamos verificar se a ordem do resultado da estimação pelos vários parâmetros é 1, pois o processo gerado é um AR(1).

```
x.ar.burg$order # ordem do parâmetro estimado - método Burg
[1] 1 # ordem do parâmetro estimado - 1
> x.ar.yw$order # ordem do parâmetro estimado - método Yule-Walker
[1] 1 # ordem do parâmetro estimado - 1
> x.ar.mle$order # ordem do parâmetro estimado - método Burg MLE
[1] 1 # ordem do parâmetro estimado - 1
```

Em todos os casos a ordem do parâmetro estimado é 1.

Vamos então ver qual é o valor estimado, $\tilde{\phi}_1$, para ϕ_1 por cada um dos métodos

```
x.ar.burg$ar      # Valor do coeficiente estimado - método Burg
[1] 0.3475724
> x.ar.yw$ar      # Valor do coeficiente estimado - método Yule-Walker
[1] 0.3472397
> x.ar.mle$ar     # Valor do coeficiente estimado - método MLE
[1] 0.3475711
```

O valor de ϕ_1 utilizado na geração do processo foi 0.4, e todos os métodos estimam ϕ_1 como 0.347, variando somente a partir da quarta casa decimal. O facto de o valor estimado não ser “muito preciso” prende-se com o valor da amostra ser só de 1000 observações, à medida que se aumentar o valor de n (quando se estabelece a dimensão da amostra), mais próximo de 0,4 será o valor estimado. Optou-se por utilizar só 1000 observações porque as representações gráficas ficam mais nítidas com 1000 observações do que por exemplo com 10 000 observações, mas para simular amostras com uma dimensão maior basta alterar o valor de n no início do código apresentado neste exemplo e correr os comandos subsequentes.

NOTA:

Caso em que se geram 10 000 observações

```
set.seed(1)      # Estabelece-se valor de arranque da seed
n= 10000         # Estabelece-se o nº da amostra
PHI <- 0.4      # definição do coeficiente AR como variável
x <- w <- rnorm(n) # inicialização do processo
x =ts(x)        # transformarr o objecto x num objecto do tipo "time series"
for (t in 2:n) x[t] <- PHI * x[t - 1] + w[t] # geração do processo AR em que PHI é definido na terceira linha
x               # mostra os dados do processo AR(1)
```

os resultados serão:

```
> x.ar.burg$ar      # Valor do coeficiente estimado - método Burg
[1] 0.4128502
> x.ar.yw$ar        # Valor do coeficiente estimado - método Yule-Walker
[1] 0.4128343
> x.ar.mle$ar       # Valor do coeficiente estimado - método MLE
[1] 0.4128251
```

Que são valores muito mais próximos de 0.4.

Voltando ao exemplo com 1000 observações, as estatísticas descritivas serão representadas matricialmente, para facilitar a visualização e tornar as comparações mais simples,

```
> M.Descriptives_Res<-matrix(c(x.ar.burg$ar,x.ar.yw$ar, x.ar.ols$ar, x.ar.mle$ar,
+ mean(x.ar.burg$res[-1]), mean(x.ar.yw$res[-1]), mean(x.ar.ols$res[-1]), mean(x.ar.burg$res[-1]),
+ sd(x.ar.burg$res[-1]), sd(x.ar.yw$res[-1]), sd(x.ar.ols$res[-1]), sd(x.ar.mle$res[-1]),
+ var(x.ar.burg$res[-1]), var(x.ar.yw$res[-1]),var(x.ar.ols$res[-1]), var(x.ar.mle$res[-1]))
+ , 4,4) # matriz 4 por 4 com coeficiente estimado, média, desvio-padrão e variância.
>
> colnames (M.Descriptives_Res) <- c(" Estim. Coef.", " Mean", " Standart Deviation", " Variance") # nomes coluna
> rownames (M.Descriptives_Res) <- c("Burg - ", "yw - ", "OLS - ", "MLE - ") # nomes linha
>
> M.Descriptives_Res # apresentação de resultados sob a forma de matriz
      Estim. Coef.      Mean      Standart Deviation      Variance
Burg - 0.3475724 1.223711e-04 1.033653 1.068438
yw - 0.3472397 1.228366e-04 1.033653 1.068438
OLS - 0.3477995 2.310143e-18 1.033653 1.068438
MLE - 0.3475711 1.223711e-04 1.033653 1.068438
>
> print(round(M.Descriptives_Res, digits=5)) # Arredondar resultados a 5 casas decimais
      Estim. Coef.      Mean      Standart Deviation      Variance
Burg - 0.34757 0.00012 1.03365 1.06844
yw - 0.34724 0.00012 1.03365 1.06844
OLS - 0.34780 0.00000 1.03365 1.06844
MLE - 0.34757 0.00012 1.03365 1.06844
```

A representação matricial mostra que, independentemente do método de estimação, os resultados são todos muito semelhantes.

Os intervalos de confiança para os coeficientes estimados por cada um dos métodos são construídos da seguinte forma:

```
> x.ar.burg$ar + c(-1.96,1.96) * sqrt(x.ar.burg$asy.var) # calculo para intervalo de confiança – método de Burg
[1] 0.2894561 0.4056888
> x.ar.yw$ar + c(-1.96,1.96) * sqrt(x.ar.yw$asy.var) # calculo para intervalo de confiança – método de Yule- Walker
[1] 0.2890575 0.4054219
> x.ar.mle$ar + c(-1.96,1.96) * sqrt(x.ar.mle$asy.var) # calculo para i. c. – método de Max. Verosimilhança
[1] 0.2894547 0.4056874
```

em formato matricial,

```
> M.I.C.P.Est<-matrix(c(x.ar.burg$ar,x.ar.yw$ar, x.ar.mle$ar,
+ x.ar.burg$ar-1.96*sqrt(x.ar.burg$asy.var), x.ar.yw$ar-1.96*sqrt(x.ar.yw$asy.var),x.ar.mle$ar -1.96 * sqrt(x.ar.mle$asy.var),
```

```

+ x.ar.burg$ar + 1.96 * sqrt(x.ar.burg$asy.var),x.ar.yw$ar + 1.96 * sqrt(x.ar.yw$asy.var),x.ar.mle$ar + 1.96 *
sqrt(x.ar.mle$asy.var)),
+ 3,3) # definição dos parâmetros constituintes da matriz
>
> colnames (M.I.C.P.Est) <- c(" Estim. Coef.", " Low. Lim. C.I.", " Sup. Lim. C.I.") # nomes para as colunas
> rownames (M.I.C.P.Est) <- c("Burg - ", "yw - ", "MLE - ") # Nomes para as linhas
> M.I.C.P.Est # apresentação de resultados em forma de matriz
      Estim. Coef.  Low. Lim. C.I.  Sup. Lim. C.I.
Burg -    0.3475724    0.2894561    0.4056888
yw -      0.3472397    0.2890575    0.4054219
MLE -     0.3475711    0.2894547    0.4056874
>
> round(M.I.C.P.Est, digits=3) # apresentação dos resultados com 3 casas decimais
      Estim. Coef.  Low. Lim. C.I.  Sup. Lim. C.I.
Burg -     0.348    0.289    0.406
yw -       0.347    0.289    0.405
MLE -     0.348    0.289    0.406

```

Todos os coeficientes estimados se encontram dentro dos respectivos intervalos de confiança. Mais uma vez, é claro que os resultados são quase idênticos para todos os métodos.

O valor da estatística de teste e os *p-values* de cada um dos métodos são calculados, respectivamente, da seguinte forma:

```

Est.Teste.x.ar.burg <- ((x.ar.burg$ar)/(sqrt(x.ar.burg$asy.var)))
Est.Teste.x.ar.yw <- ((x.ar.yw$ar)/(sqrt(x.ar.yw$asy.var)))
Est.Teste.x.ar.mle <- ((x.ar.mle$ar)/(sqrt(x.ar.mle$asy.var)))

```

```

p.value.x.ar.burg <- 2*(1 - pnorm(abs(Est.Teste.x.ar.burg), lower.tail = TRUE, log.p = FALSE))
p.value.x.ar.yw <- 2*(1 - pnorm(abs(Est.Teste.x.ar.yw), lower.tail = TRUE, log.p = FALSE))
p.value.x.ar.mle <- 2*(1 - pnorm(abs(Est.Teste.x.ar.mle), lower.tail = TRUE, log.p = FALSE))

```

Estes valores podem-se apresentar igualmente na forma matricial,

```

> M.Est.Teste.P.Value <- matrix(c(x.ar.burg$ar,x.ar.yw$ar,x.ar.mle$ar,
+ Est.Teste.x.ar.burg,Est.Teste.x.ar.yw,Est.Teste.x.ar.mle,
+ p.value.x.ar.burg ,p.value.x.ar.yw,p.value.x.ar.mle), 3,3)
> colnames (M.Est.Teste.P.Value) <- c(" Estim. Coef", " Est. de Teste", " P-Value")
> rownames (M.Est.Teste.P.Value) <- c("Burg - ", "yw - ", "mle - ")
> M.Est.Teste.P.Value
      Estim. Coef  Est. de Teste  P-Value
Burg -    0.3475724    11.72204     0
yw -      0.3472397    11.69756     0
mle -     0.3475711    11.72199     0
>

```



```
> round(M.Est.Teste.P.Value, digits=3)
      Estim. Coef  Est. de Teste  P-Value
Burg -      0.348      11.722      0
yw -       0.347      11.698      0
mle -      0.348      11.722      0
```

Mais uma vez os resultados produzidos por cada um dos métodos são quase idênticos, porque é um processo AR(1) de média nula, tal deve-se ao facto de para processos AR(1) os estimadores de *Yule-Walker* e Máxima Verosimilhança coincidirem.

Vamos representar e analisar os resíduos produzidos por cada um dos métodos utilizados na estimação de ϕ_1 .

Código para construção de títulos de gráficos

```
names1 <- c('Resíduos - ','phi','Burg', 'Yule Walker','Mínimos Quadrados', 'Máxima Verosimilhança', 'Histograma - Resíduos -
Método de')
vals1 <- c("",round((x.ar.burg$res), digits=3),round((x.ar.yw$res), digits=3),round((x.ar.ols$res),
digits=3),round((x.ar.mle$res),digits=3))
METH.expr1 <- bquote (. (as.name(names1[1]))~"
                    "~.(as.name(names1[3]))~"
                    "-~tilde(. (as.name(names1[2])))[1]==.(vals1[2]))
METH.expr2 <- bquote (. (as.name(names1[1])) ~"
                    "~.(as.name(names1[4]))~"
                    "-~tilde(. (as.name(names1[2])))[1]==.(vals1[3]))
METH.expr3 <- bquote (. (as.name(names1[1]))~"
                    "~.(as.name(names1[5]))~"
                    "-~tilde(. (as.name(names1[2])))[1]==.(vals1[4]))
METH.expr4 <- bquote (. (as.name(names1[1]))~"
                    "~.(as.name(names1[6]))~"
                    "-~tilde(. (as.name(names1[2])))[1]==.(vals1[5]))
METH.expr5 <- bquote (. (as.name(names1[7]))~"
                    "~.(as.name(names1[3]))~"
                    "-~tilde(. (as.name(names1[2])))[1]==.(vals1[2]))
```

Código para representação de gráfico dos resíduos pelo método de *Burg*

```
plot(x.ar.burg$res, main=METH.expr1,cex.main =0.8) # cronograma dos resíduos
abline(h= mean(x.ar.burg$res[-1]), col = 2) # media dos resíduos
abline(h=mean(x.ar.burg$res[-1])+ 1.96 * sd(x.ar.burg$res[-1]), col = 4) # lim. Sup. do desvio padrão dos resíduos
abline(h=mean(x.ar.burg$res[-1])- 1.96 * sd(x.ar.burg$res[-1]), col = 4) # lim. INF. do desvio padrão dos resíduos
legend("topright", legend = c("Resíduos", "Média","I.C."),col=c(1, 2, 4), lty=c(1,1,1),cex=0.7) # legenda
```

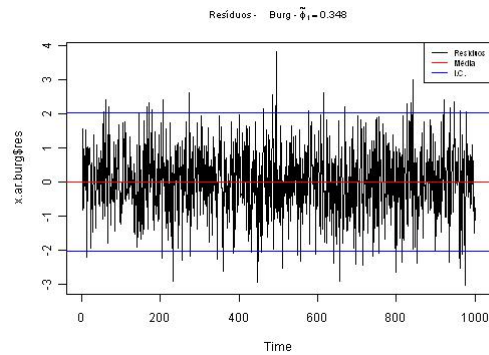


Figura A.III.7: Resíduos produzidos pela estimativa utilizando o método de *Burg*

Código para representação de gráfico dos resíduos pelo método de *Yule-Walker*

```
plot(x.ar.yw$res, main=METH.expr2,cex.main =.8)
abline(h= mean(x.ar.yw$res[-1]), col = 2)
abline(h=mean(x.ar.yw$res[-1])+ 1.96 * sd(x.ar.yw$res[-1]), col = 4)
abline(h=mean(x.ar.yw$res[-1])- 1.96 * sd(x.ar.yw$res[-1]), col = 4)
legend("topright", legend = c(" Resíduos", "Média", "I.C."),col= c(1, 2, 4), lty=c(1,1,1),cex=0.7)
```

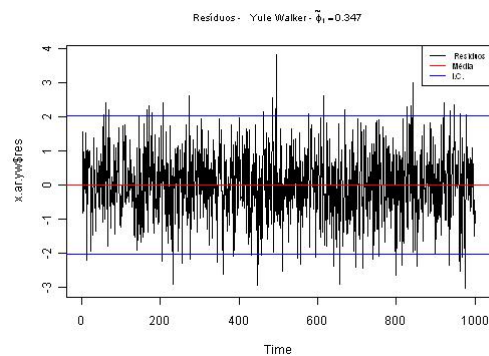


Figura A.III.8: Resíduos produzidos pela estimativa utilizando o método de *yule-walker*

Código para representação de gráfico dos resíduos pelo método de *MLE* – Máxima Verosimilhança

```
plot(x.ar.mle$res, main=METH.expr4,cex.main =.8)
abline(h= mean(x.ar.mle$res[-1]), col = 2)
abline(h=mean(x.ar.mle$res[-1])+ 1.96 * sd(x.ar.mle$res[-1]), col = 4)
abline(h=mean(x.ar.mle$res[-1])- 1.96 * sd(x.ar.mle$res[-1]), col = 4)
legend("topright", legend = c(" Resíduos", "Média", "I.C."),col= c(1, 2, 4), lty=c(1,1,1),cex=0.7)
```

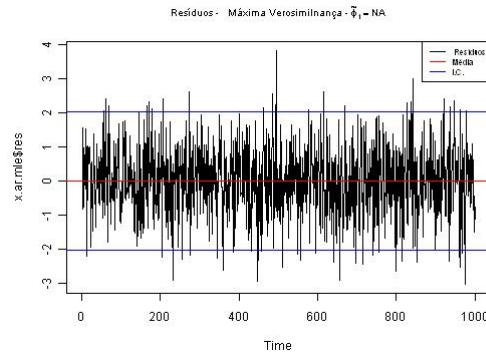


Figura A.III.9: Resíduos produzidos pela estimativa utilizando o método de Máxima Verosimilhança

Na validação de um modelo, tem de se analisar a significância dos parâmetros, e as FAC e FACP dos resíduos devem ter um comportamento semelhante ao das FAC e FACP de um ruído branco.

Os resíduos resultantes da estimação realizada por cada um dos métodos têm média que se pode considerar como sendo nula, e desvio-padrão e variância que também se podem considerar como sendo igual a um. A sua representação gráfica tem as características de um ruído branco.

Vamos passar à representação dos histogramas dos resíduos resultado das estimações por cada um dos métodos utilizados.

O código para produzir os nomes dos gráficos é o seguinte:

```
names2 <- c( 'Histograma Resíduos - ', 'phi', 'Burg', 'Yule Walker', 'OLS', 'MLE' )
vals2 <- c( "", round((x.ar.burg$ar), digits=3), round((x.ar.yw$ar), digits=3), round((x.ar.ols$ar),
digits=3), round((x.ar.mle$ar), digits=3) )
METH.HIST.expr1 <- bquote (.(as.name(names2[1]))~"
~.(as.name(names2[3]))~"
~tilde(.(as.name(names2[2])))[1]==.(vals2[2]))
METH.HIST.expr2 <- bquote (.(as.name(names2[1])) ~"
~.(as.name(names2[4]))~"
~tilde(.(as.name(names2[2])))[1]==.(vals2[3]))
METH.HIST.expr3 <- bquote (.(as.name(names2[1]))~"
~.(as.name(names2[5]))~"
~tilde(.(as.name(names2[2])))[1]==.(vals2[4]))
METH.HIST.expr4 <- bquote (.(as.name(names1[1]))~"
~.(as.name(names2[6]))~"
~tilde(.(as.name(names2[2])))[1]==.(vals2[5]))
METH.HIST.expr5 <- bquote (.(as.name(names2[1]))~"
~.(as.name(names2[3]))~"
~tilde(.(as.name(names2[2])))[1]==.(vals2[2]))
```

O código para produzir o histograma dos resíduos é o seguinte – método de *Burg*

```
hist(x.ar.burg$res[-1],prob=TRUE, main=METH.HIST.expr1,cex.main =0.8)
lines(density(x.ar.burg$res[-1]), col= 2)
```

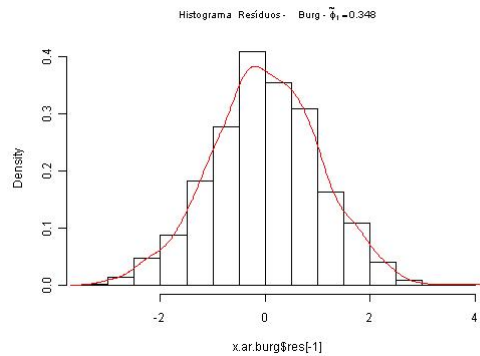


Figura A.III.10: Histograma dos resíduos – método de *Burg*

```
hist(x.ar.yw$res[-1], prob=TRUE,main=METH.HIST.expr2,cex.main =0.8)
lines(density(x.ar.yw$res[-1]), col= 2)
```

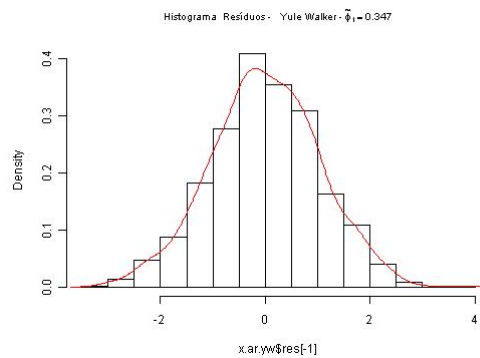


Figura A.III.11: Histograma dos resíduos – método de *Yule-Walker*

```
hist(x.ar.mle$res[-1],prob=TRUE, main=METH.HIST.expr4 ,cex.main =0.8)
lines(density(x.ar.mle$res[-1]), col= 2)
```

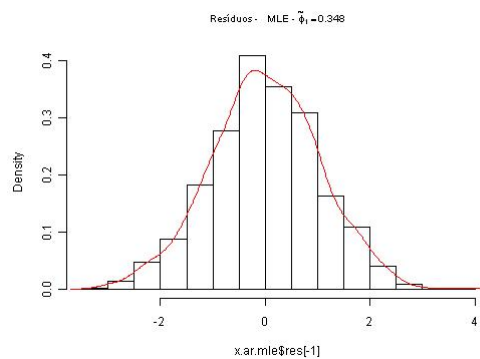


Figura A.III.12: Histograma dos resíduos – método de máxima verosimilhança

Os histogramas dos resíduos calculados pelos diferentes métodos apresentam todas características semelhantes e ajustamentos normais que nos levam a pensar que são normais.

O código para apresentar os nomes dos Box-plot é o seguinte

```
names3 <- c( 'BOX - Plot Resíduos -', 'phi', 'Burg', 'Yule Walker', 'OLS', 'MLE')
vals3      <-      c("", round((x.ar.burg$ar),      digits=3), round((x.ar.yw$ar),      digits=3), round((x.ar.ols$ar),
digits=3), round((x.ar.mle$ar), digits=3))
METH.BP.expr1 <- bquote  (.(as.name(names3[1]))~"
                        "~.(as.name(names3[3]))~"
                        "-~tilde(. (as.name(names3[2]))) [1]==.(vals3[2]))
METH.BP.expr2 <- bquote  (.(as.name(names3[1])) ~"
                        "~.(as.name(names3[4]))~"
                        "-~tilde(. (as.name(names3[2]))) [1]==.(vals3[3]))
METH.BP.expr3 <- bquote  (.(as.name(names3[1]))~"
                        "~.(as.name(names3[5]))~"
                        "-~tilde(. (as.name(names3[2]))) [1]==.(vals3[4]))
METH.BP.expr4 <- bquote  (.(as.name(names3[1]))~"
                        "~.(as.name(names3[6]))~"
                        "-~tilde(. (as.name(names3[2]))) [1]==.(vals3[5]))
METH.BP.expr5 <- bquote  (.(as.name(names3[1]))~"
                        "~.(as.name(names3[3]))~"
                        "-~tilde(. (as.name(names3[2]))) [1]==.(vals3[2]))
```

Código para a caixa de bigodes – método de *Burg*

```
boxplot(x.ar.burg$res[-1], main=METH.BP.expr1, cex.main =0.8)
```

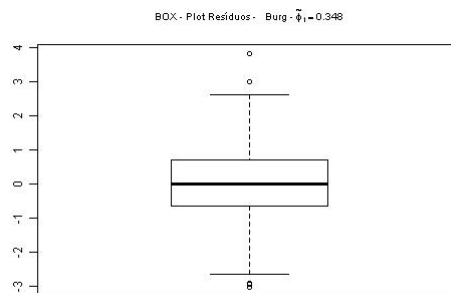


Figura A.III.13: Box-Plot dos resíduos – método de *Burg*

Código para a caixa de bigodes – método de *Yule-Walker*

```
boxplot(x.ar.yw$res[-1], main=METH.BP.expr2, cex.main =0.8)
```

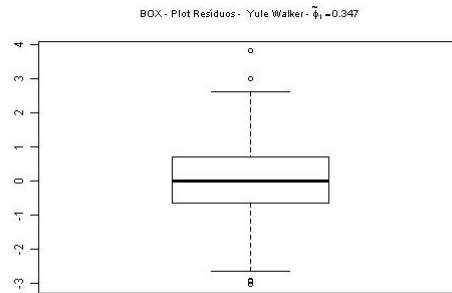


Figura A.III.14: Box-Plot dos resíduos – método de *Yule-Walker*

Código para a caixa de bigodes – método de máxima verosimilhança

```
boxplot(x.ar.mle$res[-1], main=METH.BP.expr4, cex.main =0.8)
```

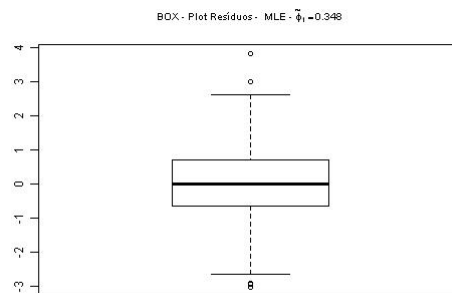


Figura A.III.15: Box-Plot dos resíduos – método de máxima verosimilhança

Breve Análise dos Box-plot (caixas de bigodes) do Resíduos

Apesar de existirem uns outliers moderados, os testes de normalidade dos resíduos não evidenciam que se deve rejeitar a normalidade

Código para construir nomes de QQ-plots por cada um dos métodos utilizados

```
names4 <- c( 'QQ - Plot Resíduos - ', 'phi', 'Burg', 'Yule Walker', 'OLS', 'MLE')
vals4      <-      c("", round((x.ar.burg$ar),          digits=3), round((x.ar.yw$ar),          digits=3), round((x.ar.ols$ar),
digits=3), round((x.ar.mle$ar), digits=3))
METH.QQP.expr1 <- bquote  (.(as.name(names4[1]))~"
                           "~.(as.name(names4[3]))~"
                           "-~tilde(.(as.name(names4[2])))[1]==.(vals4[2]))
METH.QQP.expr2 <- bquote  (.(as.name(names4[1])) ~"
                           "~.(as.name(names4[4]))~"
                           "-~tilde(.(as.name(names4[2])))[1]==.(vals4[3]))
METH.QQP.expr3 <- bquote  (.(as.name(names4[1]))~"
                           "~.(as.name(names4[5]))~"
```

```

-~tilde(.as.name(names4[2]))[1]==.(vals4[4])
METH.QQP.expr4 <- bquote (.as.name(names4[1]))~"
                    "~.(as.name(names4[6]))~"
-~tilde(.as.name(names4[2]))[1]==.(vals4[5])
METH.QQP.expr5 <- bquote (.as.name(names4[1]))~"
                    "~.(as.name(names4[3]))~"
-~tilde(.as.name(names4[2]))[1]==.(vals4[2])

```

Código para o QQ-plot dos resíduos – método de *Burg*

```

qqnorm(x.ar.burg$res[-1], main = METH.QQP.expr1, col= "blue", cex.main =0.8) # normal Q-Q plot
qqline(x.ar.burg$res[-1], col = "red")

```

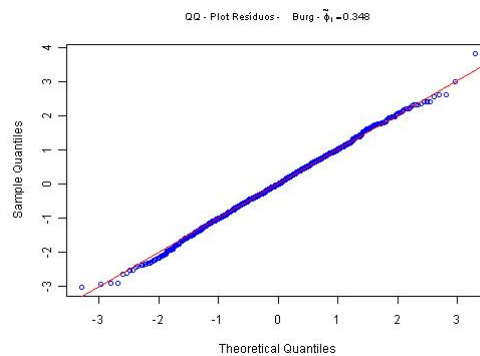


Figura A.III.16: QQ-Plot dos resíduos método de *Burg*

Código para o QQ-plot dos resíduos – método de *Yule-Walker*

```

qqnorm(x.ar.yw$res[-1], main = METH.QQP.expr2, col= "blue", cex.main =0.8) # normal Q-Q plot
qqline(x.ar.yw$res[-1], col = "red")

```

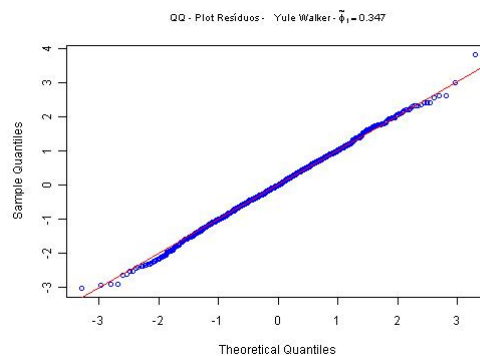


Figura A.III.17: QQ-Plot dos resíduos - método de *Yule-Walker*

Código para o QQ-plot dos resíduos – método de método de máxima verosimilhança

```

qqnorm(x.ar.mle$res[-1], main = METH.QQP.expr4, col= "blue", cex.main =0.8) # normal Q-Q plot

```

```
qqline(x.ar.mle$res[-1], col = "red")
```

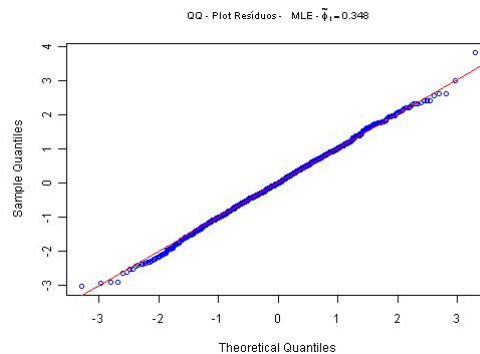


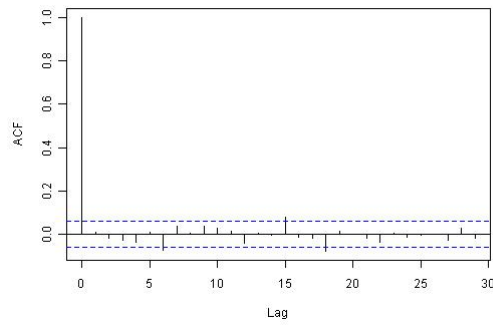
Figura a.III.18: QQ-Plot dos resíduos – método de máxima verosimilhança

Código que constrói os nomes para as FAC

```
names5 <- c('FAC Resíduos -','phi','Burg', 'Yule Walker','OLS', 'MLE')
vals5 <- c("",round((x.ar.burg$ar), digits=3),round((x.ar.yw$ar), digits=3),round((x.ar.ols$ar),
digits=3),round((x.ar.mle$ar),digits=3))
METH.FAC.expr1 <- bquote (.(as.name(names5[1]))~"
                        "~.(as.name(names5[3]))~"
                        "-~tilde(. (as.name(names5[2])))[1]==.(vals5[2]))
METH.FAC.expr2 <- bquote (.(as.name(names5[1])) ~"
                        "~.(as.name(names5[4]))~"
                        "-~tilde(. (as.name(names5[2])))[1]==.(vals5[3]))
METH.FAC.expr3 <- bquote (.(as.name(names5[1]))~"
                        "~.(as.name(names5[5]))~"
                        "-~tilde(. (as.name(names5[2])))[1]==.(vals5[4]))
METH.FAC.expr4 <- bquote (.(as.name(names5[1]))~"
                        "~.(as.name(names5[6]))~"
                        "-~tilde(. (as.name(names5[2])))[1]==.(vals5[5]))
METH.FAC.expr5 <- bquote (.(as.name(names5[1]))~"
                        "~.(as.name(names5[3]))~"
                        "-~tilde(. (as.name(names5[2])))[1]==.(vals5[2]))
```

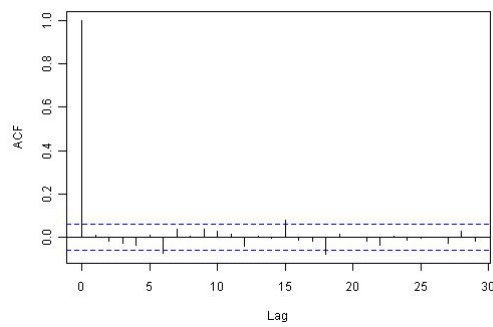
Código para representar FAC dos resíduos – método de *Burg*

```
acf(x.ar.burg$res[-1],main = METH.FAC.expr1, cex.main =0.5)
```


Figura A.III.19: FAC dos resíduos - método de *Burg*

Código para representar FAC dos resíduos – método de *Yule-Walker*

```
acf(x.ar.yw$res[-1],main = METH.FAC.expr2, cex.main =0.5)
```

Figura A.III.20: FAC dos resíduos método de *Yule-Walker*

```
acf(x.ar.mle$res[-1],main = METH.FAC.expr4, cex.main =0.5)
```

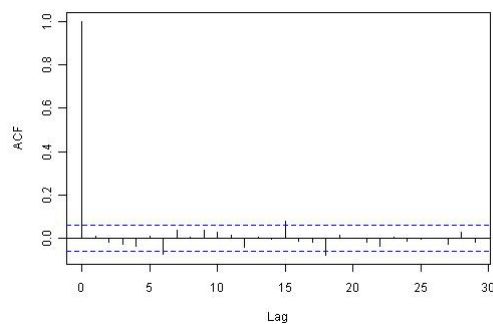


Figura A.III.21: FAC dos resíduos método de Máxima Verossimilhança

Código para a construção de nomes para FACP dos resíduos

```
names6 <- c('FACP Resíduos -','phi','Burg', 'Yule Walker','OLS', 'MLE')
```

```

vals6 <- c("",round((x.ar.burg$ar), digits=3),round((x.ar.yw$ar), digits=3),round((x.ar.ols$ar),
digits=3),round((x.ar.mle$ar),digits=3))
METH.FACP.expr1 <- bquote (.(as.name(names6[1]))~"
                        "~.(as.name(names6[3]))~"
                        "-~tilde(.as.name(names6[2]))[1]==.(vals6[2]))
METH.FACP.expr2 <- bquote (.(as.name(names6[1])) ~"
                        "~.(as.name(names6[4]))~"
                        "-~tilde(.as.name(names6[2]))[1]==.(vals6[3]))
METH.FACP.expr3 <- bquote (.(as.name(names6[1]))~"
                        "~.(as.name(names6[5]))~"
                        "-~tilde(.as.name(names6[2]))[1]==.(vals6[4]))
METH.FACP.expr4 <- bquote (.(as.name(names6[1]))~"
                        "~.(as.name(names6[6]))~"
                        "-~tilde(.as.name(names6[2]))[1]==.(vals6[5]))
METH.FACP.expr5 <- bquote (.(as.name(names6[1]))~"
                        "~.(as.name(names6[3]))~"
                        "-~tilde(.as.name(names6[2]))[1]==.(vals6[2]))

```

```
pacf(x.ar.burg$res[-1], main = METH.FACP.expr1, cex.main =0.5)
```

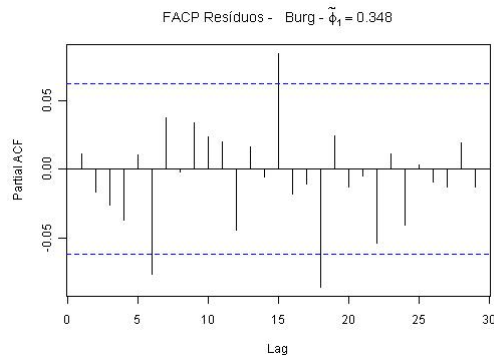


Figura A.III.22: FACP dos resíduos

```
pacf(x.ar.yw$res[-1], main = METH.FACP.expr2, cex.main =0.5)
```

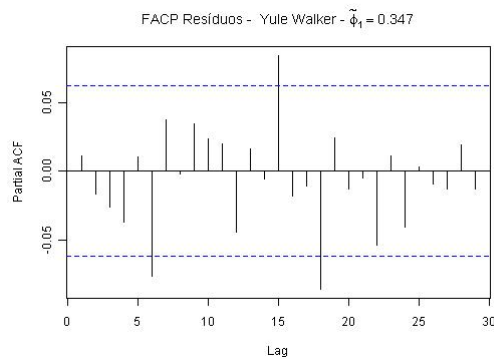


Figura A.III.23: FACP dos resíduos

```
pacf(x.ar.mle$res[-1], main = METH.FACP.expr4, cex.main =0.5)
```

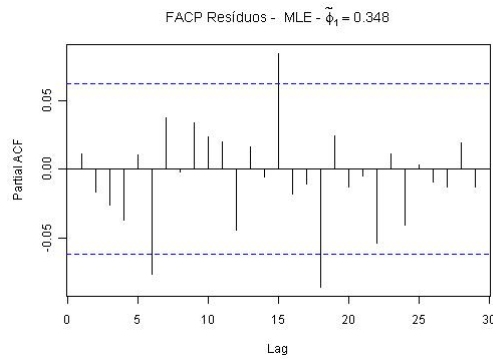


Figura A.III.24: FACP dos resíduos

Uma vez visto o comportamento dos resíduos através dos diferentes gráficos, vamos aplicar alguns testes estatísticos aos resíduos resultantes dos ajustamentos provenientes de cada um dos métodos de estimação utilizados.

Tem de instalar a livreria “*nortest*”, o código é o seguinte

```
install.packages("nortest") # download da livreria nortest
library(nortest)           # carregar a livreria no workspace
```

O código para aplicar o *Lilliefors (Kolmogorov-Smirnov) normality test* é o seguinte

```
lillie.test(x.ar.burg$res[-1])
lillie.test(x.ar.yw$res[-1])
lillie.test(x.ar.mle$res[-1])
```

cujos resultados se colaram no quadro seguinte

Resíduos	x.ar.burg\$res[-1]	x.ar.yw\$res[-1]	x.ar.mle\$res[-1]
D	0.0168	0.0166	0.0168
p-value	0.7096	0.7224	0.7097

Mais uma vez a diferença de resultados não é expressiva, porque se simulou um processo AR(1) de média nula (os estimadores coincidem). Os valores dos *p-values* e as estatísticas de teste de cada um modelos fazem com que não se rejeitem os parâmetros no modelo.

O Teste de Box-Pierce aplica-se da seguinte forma,

```
Box.test(x.ar.burg$res[-1], lag = 1, type = "Box-Pierce")
Box.test(x.ar.yw$res[-1], lag = 1, type = "Box-Pierce")
Box.test(x.ar.mle$res[-1], lag = 1, type = "Box-Pierce")
```

De modo a facilitar a interpretação colocaram-se os valores na tabela seguinte,

Resíduos	x.ar.burg\$res[-1]	x.ar.yw\$res[-1]	x.ar.mle\$res[-1]
X-squared	0.1204	0.1275	0.1204
df	1	1	1
p-value	0.7286	0.721	0.7286

Os *p-values* indicam que o parâmetro, em cada um dos métodos deve ficar no modelo.

O teste de t realiza-se da seguinte forma,

```
> t.test(x.ar.burg$res[-1])
```

One Sample t-test

data: x.ar.burg\$res[-1]

t = 0.0037, df = 998, p-value = 0.997

alternative hypothesis: true mean is not equal to 0

95 percent confidence interval:

-0.06405279 0.06429754

sample estimates:

mean of x

0.0001223711

```
> t.test(x.ar.yw$res[-1])
```

One Sample t-test

data: x.ar.yw\$res[-1]

t = 0.0038, df = 998, p-value = 0.997

alternative hypothesis: true mean is not equal to 0

95 percent confidence interval:

-0.06405234 0.06429801

sample estimates:

mean of x

0.0001228366

```
> t.test(x.ar.mle$res[-1])
```

One Sample t-test

data: x.ar.mle\$res[-1]

t = 0.0038, df = 998, p-value = 0.997

alternative hypothesis: true mean is not equal to 0

95 percent confidence interval:

-0.06405140 0.06429893

sample estimates:

mean of x

0.0001237623

Os *p-values* indicam que se trata de ruído branco.

O teste de Normalidade de Shapiro Wilks aplica-se da seguinte forma,

```
shapiro.test(x.ar.burg$res[-1])
shapiro.test(x.ar.yw$res[-1])
shapiro.test(x.ar.mle$res[-1])
```

com os resultados apresentados nas tabela seguinte

Resíduos	x.ar.burg\$res[-1]	x.ar.yw\$res[-1]	x.ar.mle\$res[-1]
w	0.9988	0.9988	0.9988
p-value	0.7783	0.7783	0.7782

Este teste não se devia aplicar neste caso, está aqui só a título exemplificativo, só se utiliza quando $n < 50$.

Código para construir nomes para títulos de gráficos de serie simulada versus série ajustada por cada um dos métodos apresentados

```
names7 <- c( 'Série gerada vs. ajustada ', 'phi', 'Burg', 'Yule Walker', 'OLS', 'MLE', 'Previsão')
vals7 <- c(PHI, round((x.ar.burg$ar), digits=3), round((x.ar.yw$ar), digits=3), round((x.ar.ols$ar),
digits=3), round((x.ar.mle$ar), digits=3))
PREV.expr1 <- bquote (.(as.name(names7[1]))~"
                      "~.(as.name(names7[3]))~"
                      "-~tilde(. (as.name(names7[2])))[1]==.(vals7[2])~"
                      "-//-~.(as.name(names7[2]))[1]==.(vals7[1]))
PREV.expr2 <- bquote (.(as.name(names7[1])) ~"
                      "~.(as.name(names7[4]))~"
                      "-~tilde(. (as.name(names7[2])))[1]==.(vals7[3])~"
                      "-//-~.(as.name(names7[2]))[1]==.(vals7[1]))
PREV.expr3 <- bquote (.(as.name(names7[1]))~"
                      "~.(as.name(names7[5]))~"
                      "-~tilde(. (as.name(names7[2])))[1]==.(vals7[4])~"
                      "-//-~.(as.name(names7[2]))[1]==.(vals7[1]))
PREV.expr4 <- bquote (.(as.name(names7[1]))~"
                      "~.(as.name(names7[6]))~"
                      "-~tilde(. (as.name(names7[2])))[1]==.(vals7[5])~"
                      "-//-~.(as.name(names7[2]))[1]==.(vals7[1]))
PREV.expr5 <- bquote (.(as.name(names7[7]))~"
                      "~.(as.name(names7[3]))~"
                      "-~tilde(. (as.name(names7[2])))[1]==.(vals7[2])~"
                      "-//-~.(as.name(names7[2]))[1]==.(vals7[1]))
PREV.expr6 <- bquote (.(as.name(names7[7])) ~"
```

```

"~.(as.name(names7[4]))~"
"-~tilde(.as.name(names7[2]))[1]==.(vals7[3])~"
"-//~-".(as.name(names7[2]))[1]==.(vals7[1])
PREV.expr7 <- bquote (.(as.name(names7[7]))~"
"~.(as.name(names7[5]))~"
"-~tilde(.as.name(names7[2]))[1]==.(vals7[4])~"
"-//~-".(as.name(names7[2]))[1]==.(vals7[1])
PREV.expr8 <- bquote (.(as.name(names7[7]))~"
"~.(as.name(names7[6]))~"
"-~tilde(.as.name(names7[2]))[1]==.(vals7[5])~"
"-//~-".(as.name(names7[2]))[1]==.(vals7[1])

```

Código para serie simulada versus série ajustada por cada um dos métodos apresentados

```

plot(x, type = "l", main=PREV.expr1, col = 1)
z <- numeric()
for (t in 2:n) z[t] <- x.ar.burg$ar * x[t - 1]
points(z, col= 2, type = "o", )
legend("topright", legend = c(" Série Simulada", "Série Ajustada"), col= c(1, 2), lty=c(1,1))

```

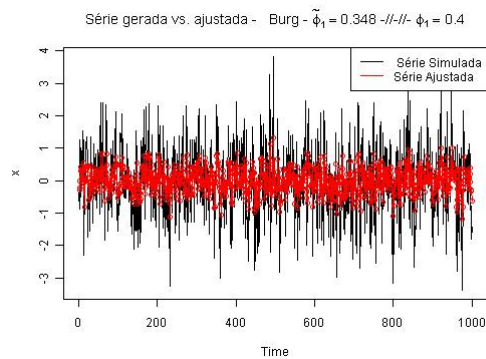


Figura A.III.25: simulada vs. Série ajustada

```

plot(x, type = "l", main=PREV.expr2, col = 1)
z <- numeric()
for (t in 2:n) z[t] <- x.ar.yw$ar * x[t - 1]
points(z, col= 2, type = "o", main=(expression(AR(1)~--~phi==PHI)))
legend("topright", legend = c(" Série Simulada", "Série Ajustada"), col= c(1, 2), lty=c(1,1))

```

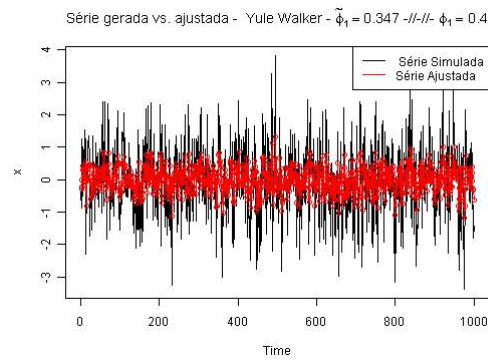


Figura A.III.26: simulada vs. Série ajustada

```
plot(x, type = "l", main=PREV.expr4, col = 1)
z <- numeric()
for (t in 2:n) z[t] <- x.ar.mle$ar * x[t - 1]
points(z, col= 2, type = "o", main=(expression(AR(1)~--~phi==PHI)))
legend("topright", legend = c(" Série Simulada", "Série Ajustada"), col= c(1, 2), lty=c(1,1))
```

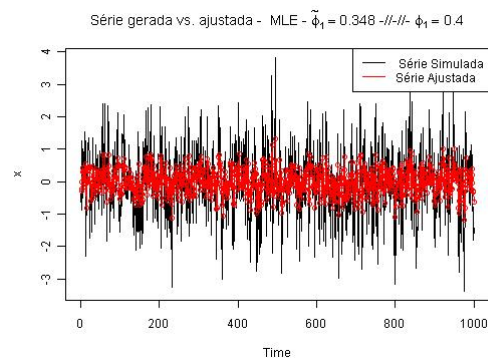


Figura A.III.27: simulada vs. Série ajustada

Código para cálculo de limites de imagem e valores de previsão

```
> x.burg.fore = predict(x.ar.burg, n.ahead = 50)
> summary(x.burg.fore)
  Length Class Mode
pred 50  ts  numeric
se 50  ts  numeric
> x.burg.fore$pred
Time Series:
Start = 1001
End = 1050
Frequency = 1
[1] -0.50420507 -0.18729785 -0.07714964 -0.03886515 -0.02555852 -0.02093350
[7] -0.01932597 -0.01876724 -0.01857304 -0.01850554 -0.01848208 -0.01847393
[13] -0.01847109 -0.01847011 -0.01846977 -0.01846965 -0.01846961 -0.01846959
[19] -0.01846959 -0.01846959 -0.01846958 -0.01846958 -0.01846958 -0.01846958
[25] -0.01846958 -0.01846958 -0.01846958 -0.01846958 -0.01846958 -0.01846958
[31] -0.01846958 -0.01846958 -0.01846958 -0.01846958 -0.01846958 -0.01846958
```

```
[37] -0.01846958 -0.01846958 -0.01846958 -0.01846958 -0.01846958 -0.01846958
[43] -0.01846958 -0.01846958 -0.01846958 -0.01846958 -0.01846958 -0.01846958
[49] -0.01846958 -0.01846958
> x.burg.fore$se
Time Series:
Start = 1001
End = 1050
Frequency = 1
[1] 1.032776 1.093381 1.100476 1.101330 1.101433 1.101446 1.101447 1.101448
[9] 1.101448 1.101448 1.101448 1.101448 1.101448 1.101448 1.101448 1.101448
[17] 1.101448 1.101448 1.101448 1.101448 1.101448 1.101448 1.101448 1.101448
[25] 1.101448 1.101448 1.101448 1.101448 1.101448 1.101448 1.101448 1.101448
[33] 1.101448 1.101448 1.101448 1.101448 1.101448 1.101448 1.101448 1.101448
[41] 1.101448 1.101448 1.101448 1.101448 1.101448 1.101448 1.101448 1.101448
[49] 1.101448 1.101448
> U.burg=x.burg.fore$pred + 2* x.burg.fore$se
> U.burg
Time Series:
Start = 1001
End = 1050
Frequency = 1
[1] 1.561347 1.999464 2.123803 2.163796 2.177308 2.181958 2.183569 2.184128
[9] 2.184322 2.184390 2.184413 2.184421 2.184424 2.184425 2.184426 2.184426
[17] 2.184426 2.184426 2.184426 2.184426 2.184426 2.184426 2.184426 2.184426
[25] 2.184426 2.184426 2.184426 2.184426 2.184426 2.184426 2.184426 2.184426
[33] 2.184426 2.184426 2.184426 2.184426 2.184426 2.184426 2.184426 2.184426
[41] 2.184426 2.184426 2.184426 2.184426 2.184426 2.184426 2.184426 2.184426
[49] 2.184426 2.184426
> L.burg=x.burg.fore$pred - 2* x.burg.fore$se
> L.burg
Time Series:
Start = 1001
End = 1050
Frequency = 1
[1] -2.569757 -2.374059 -2.278102 -2.241526 -2.228426 -2.223825 -2.222221
[8] -2.221663 -2.221468 -2.221401 -2.221377 -2.221369 -2.221366 -2.221365
[15] -2.221365 -2.221365 -2.221365 -2.221365 -2.221365 -2.221365 -2.221365
[22] -2.221365 -2.221365 -2.221365 -2.221365 -2.221365 -2.221365 -2.221365
[29] -2.221365 -2.221365 -2.221365 -2.221365 -2.221365 -2.221365 -2.221365
[36] -2.221365 -2.221365 -2.221365 -2.221365 -2.221365 -2.221365 -2.221365
[43] -2.221365 -2.221365 -2.221365 -2.221365 -2.221365 -2.221365 -2.221365
[50] -2.221365
> minx.burg=min(x,L.burg)
> maxx.burg=max(x,U.burg)
> minx.burg
[1] -3.368315
> maxx.burg
[1] 3.827546
```

Código para representar graficamente


```

x<-ts(x)
ts.plot(x, x.burg.fore$pred, main=PREV.expr5, col=1:2, ylim=c(minx.burg, maxx.burg)) # só funciona de passar x a ts, feito nas
linhas acima
lines(U.burg, col="blue", lty="dashed")
lines(L.burg, col="blue", lty="dashed")
legend("topright", legend = c("Valores Observados", "Previsão", "Limites"), col= c(1,2, 4), lty=c(1,1,1))

```

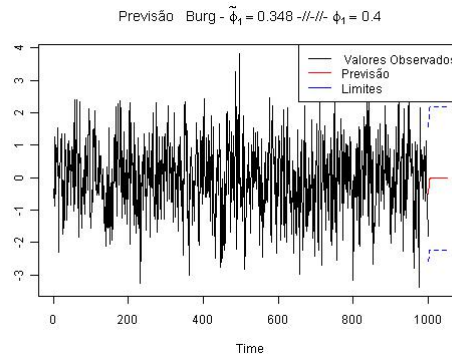


Figura A.III.28: Previsão

Código para cálculo de limites de imagem e valores de previsão

```

> x.yw.fore = predict(x.ar.yw, n.ahead = 50)
> summary(x.yw.fore)
  Length Class Mode
pred 50  ts  numeric
se 50  ts  numeric
> x.yw.fore$pred
Time Series:
Start = 1001
End = 1050
Frequency = 1
 [1] -0.50374009 -0.18697478 -0.07698128 -0.03878717 -0.02552466 -0.02091939
 [7] -0.01932025 -0.01876497 -0.01857215 -0.01850520 -0.01848195 -0.01847388
[13] -0.01847108 -0.01847010 -0.01846976 -0.01846965 -0.01846961 -0.01846959
[19] -0.01846959 -0.01846959 -0.01846958 -0.01846958 -0.01846958 -0.01846958
[25] -0.01846958 -0.01846958 -0.01846958 -0.01846958 -0.01846958 -0.01846958
[31] -0.01846958 -0.01846958 -0.01846958 -0.01846958 -0.01846958 -0.01846958
[37] -0.01846958 -0.01846958 -0.01846958 -0.01846958 -0.01846958 -0.01846958
[43] -0.01846958 -0.01846958 -0.01846958 -0.01846958 -0.01846958 -0.01846958
[49] -0.01846958 -0.01846958
> x.yw.fore$se
Time Series:
Start = 1001
End = 1050
Frequency = 1
 [1] 1.033946 1.094507 1.101584 1.102434 1.102537 1.102549 1.102551 1.102551
 [9] 1.102551 1.102551 1.102551 1.102551 1.102551 1.102551 1.102551 1.102551
[17] 1.102551 1.102551 1.102551 1.102551 1.102551 1.102551 1.102551 1.102551
[25] 1.102551 1.102551 1.102551 1.102551 1.102551 1.102551 1.102551 1.102551
[33] 1.102551 1.102551 1.102551 1.102551 1.102551 1.102551 1.102551 1.102551
[41] 1.102551 1.102551 1.102551 1.102551 1.102551 1.102551 1.102551 1.102551

```

```

[49] 1.102551 1.102551
> U.yw=x.yw.fore$pred + 2* x.yw.fore$se
> U.yw
Time Series:
Start = 1001
End = 1050
Frequency = 1
[1] 1.564152 2.002039 2.126187 2.166081 2.179549 2.184179 2.185781 2.186337
[9] 2.186529 2.186596 2.186620 2.186628 2.186630 2.186631 2.186632 2.186632
[17] 2.186632 2.186632 2.186632 2.186632 2.186632 2.186632 2.186632 2.186632
[25] 2.186632 2.186632 2.186632 2.186632 2.186632 2.186632 2.186632 2.186632
[33] 2.186632 2.186632 2.186632 2.186632 2.186632 2.186632 2.186632 2.186632
[41] 2.186632 2.186632 2.186632 2.186632 2.186632 2.186632 2.186632 2.186632
[49] 2.186632 2.186632
> L.yw=x.yw.fore$pred - 2* x.yw.fore$se
> L.yw
Time Series:
Start = 1001
End = 1050
Frequency = 1
[1] -2.571632 -2.375988 -2.280149 -2.243656 -2.230598 -2.226018 -2.224421
[8] -2.223866 -2.223674 -2.223607 -2.223583 -2.223575 -2.223573 -2.223572
[15] -2.223571 -2.223571 -2.223571 -2.223571 -2.223571 -2.223571 -2.223571
[22] -2.223571 -2.223571 -2.223571 -2.223571 -2.223571 -2.223571 -2.223571
[29] -2.223571 -2.223571 -2.223571 -2.223571 -2.223571 -2.223571 -2.223571
[36] -2.223571 -2.223571 -2.223571 -2.223571 -2.223571 -2.223571 -2.223571
[43] -2.223571 -2.223571 -2.223571 -2.223571 -2.223571 -2.223571 -2.223571
[50] -2.223571
> minx.yw=min(x,L.yw)
> maxx.yw=max(x,U.yw)
> minx.yw
[1] -3.368315
> maxx.yw
[1] 3.827546

```

Código para representar graficamente

```

ts.plot(x, x.yw.fore$pred,main=PREV.expr6, col=1:2, ylim=c(minx.yw, maxx.yw)) # só funciona de passar x a ts, feito nas
linhas acima
lines(U.yw, col= "blue", lty = "dashed")
lines(L.yw, col= "blue", lty = "dashed")
legend("topright", legend = c(" Valores Observados", "Previsão", "Limites"),col= c(1,2, 4), lty=c(1,1,1))

```

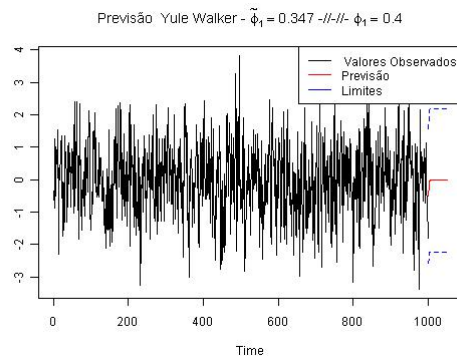


Figura A.II.29: Previsão

Código para cálculo de limites de imagem e valores de previsão

```
> x.mle.fore = predict(x.ar.mle, n.ahead = 50)
> summary(x.mle.fore)
  Length Class Mode
pred 50  ts  numeric
se 50  ts  numeric
> x.mle.fore$pred
Time Series:
Start = 1001
End = 1050
Frequency = 1
 [1] -0.50420452 -0.18729837 -0.07715097 -0.03886693 -0.02556050 -0.02093557
 [7] -0.01932808 -0.01876936 -0.01857517 -0.01850767 -0.01848421 -0.01847606
[13] -0.01847322 -0.01847224 -0.01847190 -0.01847178 -0.01847174 -0.01847172
[19] -0.01847172 -0.01847171 -0.01847171 -0.01847171 -0.01847171 -0.01847171
[25] -0.01847171 -0.01847171 -0.01847171 -0.01847171 -0.01847171 -0.01847171
[31] -0.01847171 -0.01847171 -0.01847171 -0.01847171 -0.01847171 -0.01847171
[37] -0.01847171 -0.01847171 -0.01847171 -0.01847171 -0.01847171 -0.01847171
[43] -0.01847171 -0.01847171 -0.01847171 -0.01847171 -0.01847171 -0.01847171
[49] -0.01847171 -0.01847171
> x.mle.fore$se
Time Series:
Start = 1001
End = 1050
Frequency = 1
 [1] 1.032776 1.093380 1.100476 1.101330 1.101433 1.101445 1.101447 1.101447
 [9] 1.101447 1.101447 1.101447 1.101447 1.101447 1.101447 1.101447 1.101447
[17] 1.101447 1.101447 1.101447 1.101447 1.101447 1.101447 1.101447 1.101447
[25] 1.101447 1.101447 1.101447 1.101447 1.101447 1.101447 1.101447 1.101447
[33] 1.101447 1.101447 1.101447 1.101447 1.101447 1.101447 1.101447 1.101447
[41] 1.101447 1.101447 1.101447 1.101447 1.101447 1.101447 1.101447 1.101447
[49] 1.101447 1.101447
> U.mle=x.mle.fore$pred + 2* x.mle.fore$se
> U.mle
Time Series:
Start = 1001
End = 1050
```

```

Frequency = 1
[1] 1.561347 1.999462 2.123800 2.163793 2.177305 2.181955 2.183566 2.184125
[9] 2.184319 2.184386 2.184410 2.184418 2.184421 2.184422 2.184422 2.184422
[17] 2.184422 2.184422 2.184422 2.184422 2.184422 2.184422 2.184422 2.184422
[25] 2.184422 2.184422 2.184422 2.184422 2.184422 2.184422 2.184422 2.184422
[33] 2.184422 2.184422 2.184422 2.184422 2.184422 2.184422 2.184422 2.184422
[41] 2.184422 2.184422 2.184422 2.184422 2.184422 2.184422 2.184422 2.184422
[49] 2.184422 2.184422
> L.mle=x.mle.fore$pred - 2* x.mle.fore$se
> L.mle
Time Series:
Start = 1001
End = 1050
Frequency = 1
[1] -2.569756 -2.374059 -2.278102 -2.241526 -2.228426 -2.223826 -2.222222
[8] -2.221663 -2.221469 -2.221402 -2.221378 -2.221370 -2.221367 -2.221366
[15] -2.221366 -2.221366 -2.221366 -2.221366 -2.221366 -2.221366 -2.221366
[22] -2.221366 -2.221366 -2.221366 -2.221366 -2.221366 -2.221366 -2.221366
[29] -2.221366 -2.221366 -2.221366 -2.221366 -2.221366 -2.221366 -2.221366
[36] -2.221366 -2.221366 -2.221366 -2.221366 -2.221366 -2.221366 -2.221366
[43] -2.221366 -2.221366 -2.221366 -2.221366 -2.221366 -2.221366 -2.221366
[50] -2.221366
> minx.mle=min(x,L.mle)
> maxx.mle=max(x,U.mle)
> minx.mle
[1] -3.368315
> maxx.mle
[1] 3.827546

```

Código para representar graficamente

```

ts.plot(x, x.mle.fore$pred,main=PREV.expr8, col=1:2, ylim=c(minx.mle, maxx.mle)) # só funciona de passar x a ts, feito nas
linhas acima
lines(U.yw, col= "blue", lty = "dashed")
lines(L.yw, col= "blue", lty = "dashed")
legend("topright", legend = c("Valores Observados", "Previsão", "Limites"),col= c(1,2, 4), lty=c(1,1,1))

```

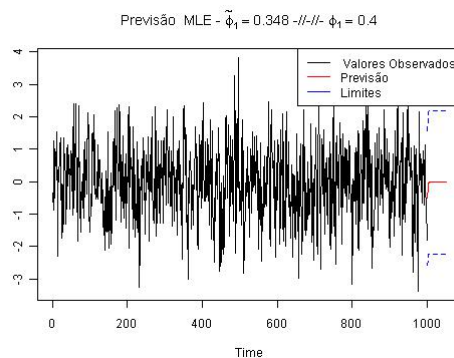


Figura A.III.30: Previsão

Anexo IV

Geração e análise de um processo AR(1) em R com média e variância não nulas pelo método de Yule-Walker.

Considere-se o processo AR(1) da forma,

$$X_t = \mu + \phi_1 X_{t-1} + Z_t$$

Com

$$\begin{cases} \mu = 5 \\ \phi = 0.4 \\ \text{var}(Z) = 1 \end{cases} \text{ e } \begin{cases} Z \sim N(0, \sqrt{\text{var}(Z)}) \\ X \sim N\left(\frac{\mu}{1-\phi}, \sqrt{\text{var}(X)}\right) \end{cases}$$

Os parâmetros de inicialização deste processo são diferentes do que se utilizou no Anexo III. Normalmente os processos têm média, ao contrário do que se simulou no Anexo III. A programação é em tudo idêntica, mas desta vez utiliza-se só a estimação de Yule-Walker.

Neste exemplo utiliza-se também o que é denominado de *burn in*, que não é mais do que inicializar o processo com um determinado número de observações. Ao se iniciar a análise eliminam-se as observações iniciais eliminando assim, possíveis problemas de aleatoriedade derivados dos primeiros valores gerados e trabalha-se com valores mais estáveis.

O código para gerar o processo é,

```
set.seed(1) # estabelece-se o valor de arranque da seed
m=2000 # estabelece-se o valor da amostra inicial
PHI <- 0.4 # estabelece-se o valor de phi
Miu <- 5 # média da série
Var.z <- 1 # variância da distribuição normal para gerar
med.x1 <- (Miu/(1-PHI)) # estabelecimento da média
Var.x <- (Var.z/(1-PHI^2)) # Variância da distribuição normal para gerar
z <- rnorm(m,mean=0,sd=sqrt(Var.z)) # geração do processo aleatório Z
n= 1000 # estabelece-se o nº da amostra depois do burn In
x <- rnorm(1,mean=med.x1,sd=sqrt(Var.x)) # estabelece-se o valor inicial x
```

```

t<-2                                # Estabelece-se o valor inicial de t
for (t in 2:m) x[t] <- Miu + PHI * x[t - 1] + z[t]    #geração do processo
x                                     # visualizam-se as observações (2000)
x<-x[(m-n):m]                         # retiram-se as primeiras 1000 observações do processo
                                     # no que se denomina processo de burn-in, que tem por
                                     # objectivo eliminar possíveis influências espúrias de
                                     # geração
x<-ts(x)                               # transformar x de objecto "numeric" em "ts"
x[1:3];x[998:1000]                   # mostra as observações de ordem 1,2,3,998,999 e 1000
[1] 6.917355 8.901907 9.672695        # observações de ordem 1,2,3
[1] 9.512209 8.005757 9.206536        # observações de ordem 998,999 e 1000

```

O código para títulos de cronograma, histograma, QQ-plot, boxplot, FAC e FAC,

```

names <- c('Simulação de Processo AR(1)', 'phi', 'Histograma', 'QQ plot', 'Caixa de Bigodes', 'FAC', 'FACP')
vals <- c("", PHI)
expr1 <- bquote  (.(as.name(names[1])) ~~~~"
                 "~.(as.name(names[2]))[1]==.(vals[2]))

expr2 <- bquote  (.(as.name(names[2]))[1]==.(vals[2]))

expr3 <- bquote  (.(as.name(names[3])) ~~~~"
                 "~.(as.name(names[2]))[1]==.(vals[2]))
expr4 <- bquote  (.(as.name(names[4])) ~~~~"
                 "~.(as.name(names[2]))[1]==.(vals[2]))
expr5 <- bquote  (.(as.name(names[5])) ~~~~"
                 "~.(as.name(names[2]))[1]==.(vals[2]))
expr6 <- bquote  (.(as.name(names[6])) ~~~~"
                 "~.(as.name(names[2]))[1]==.(vals[2]))

expr7 <- bquote  (.(as.name(names[7])) ~~~~"
                 "~.(as.name(names[2]))[1]==.(vals[2]))

expr8 <- bquote  (.(as.name(names[3])) )

```

O código para cronograma, histograma, QQ-plot, box plot FAC e FAC,

```

plot(x, type = "l", main = expr1, col = "blue", cex.main = 1)      # cronograma
abline(h=mean(x), col = "red")                                     # representação da média
legend("topright", legend = c(" Série", "Média"), col= c(4, 2), lty=c(1,1), cex=0.8)

hist(x, prob=TRUE, main = expr3, cex.main = 1)                   # histograma
lines(density(x), col= "2")                                       # densidade normal no histograma
qqnorm(x, main = expr4, col= "4", cex.main = 1) # normal Q-Q plot #Gráfico QQ-plot
qqline(x, col = "2")                                              # linha de ajustamento normal em QQ-plot

boxplot(x, main = expr5, cex.main = 1)                            # Caixa de bigodes
acf(x, xlim =c(2,36), main = expr6, cex.main = 1)                #FAC

```

```
pacf(x,xlim =c(2,36), main = expr7, cex.main =1)
```

```
#FACP
```

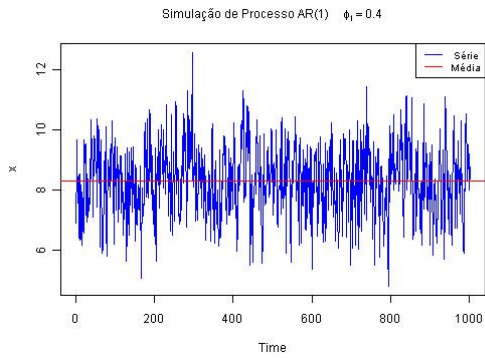


Figura A.IV.1: Cronograma do processo AR(1) gerado

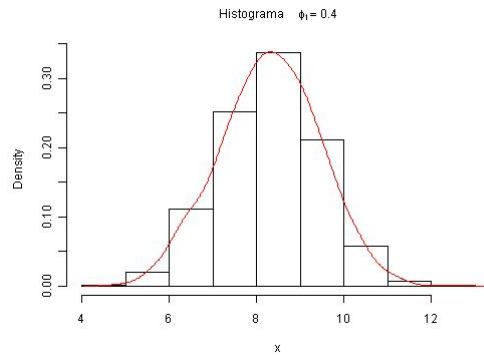


Figura A.IV.2: Histograma do processo gerado com ajustamento de curva normal

O cronograma e o histograma mostram que se trata de um processo de média não nula, e superior a 8.

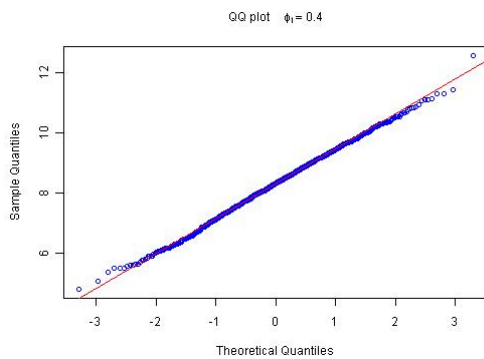


Figura A.IV.3: representação do processo gerado em papel de probabilidades (QQ plot) com ajustamento normal

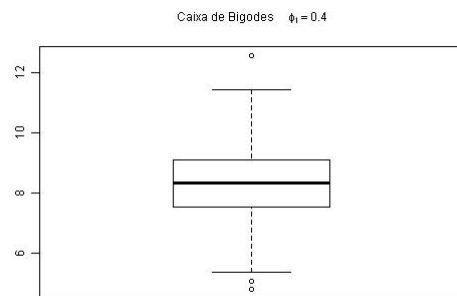


Figura A.IV.4: Caixa de bigodes do processo gerado

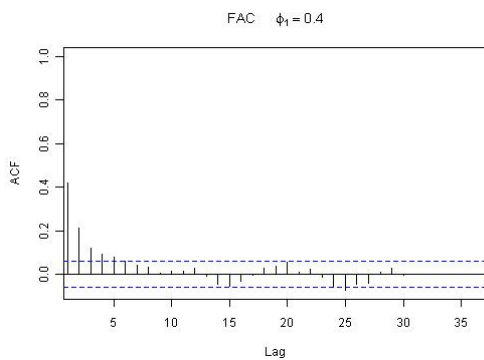


Figura A.IV.5: FAC do processo gerado

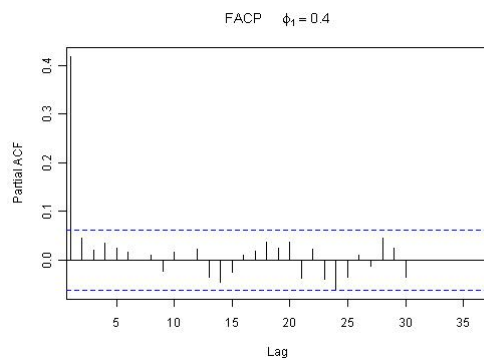


Figura A.IV.6: FACP do processo gerado

Comportamento típico de um AR(1), onde a FAC decai de forma exponencial para zero e FACP a ter o lag 1 fora dos limites e a decair bruscamente para zero a partir do lag 1.

Uma vez realizada a representação gráfica usual (histograma, pp-plot, qqplot, FAC e FACP) é lugar para se mostrar como se obtêm algumas das estatísticas descritivas.

De modo a tornar a leitura mais simples, apresentam-se os resultados em forma matricial. O código é o seguinte:

```
> M.Descriptives.x<-matrix(c(mean(x),sd(x), var(x)), 1,3) # Definição da matriz e seus valores
> colnames (M.Descriptives.x) <- c(" Mean"," Standart Deviation", " Variance") # nomes Colunas -3
> rownames (M.Descriptives.x) <- c("coeficients") # nomes da linhas
> M.Descriptives.x # apresentação de resultados
      Mean  Standart Deviation  Variance
coeficients 8.303875      1.144927  1.310858
> print(round(M.Descriptives.x, digits=5)) # apresentação de resultados com 5 casas decimais
      Mean  Standart Deviation  Variance
coeficients 8.30387      1.14493  1.31086
```

A média é 8.30387, que é um valor aproximado do valor teórico esperado,

$$\mu_x = \frac{\mu}{1-\phi} = \frac{5}{1-0.4} = 8.(3), \text{ pois a constante de inicialização foi 5 e } \phi=0.4, \text{ e o desvio-}$$

padrão (1.14493) .

A fase de estimação dos parâmetros do processo, os coeficientes do processo AR(1) - ϕ_1 e μ .

```
x.ar.yw <- ar(x,demean=TRUE, method = "yw") # estimação de parâmetro - método de Yule Walker
```

Uma vez estimados os parâmetros, vamos verificar se a ordem do resultado da estimação pelos vários parâmetros é 1, pois o processo gerado é um AR(1).

```
> x.ar.yw$order # ordem do parâmetro estimado - método Yule-Walker
[1] 1 # ordem do parâmetro estimado - 1
```

Em todos os casos a ordem do parâmetro estimado é 1.

O valor estimado, $\tilde{\phi}_1$, para ϕ_1

```
> x.ar.yw$ar          # valor do coeficiente estimado - método Yule-Walker
[1] 0.4178466
```

O valor de ϕ_1 utilizado na geração do processo foi 0.4, e o valor estimado é 0.4178466, valor mais próximo do utilizado na geração, tal deve-se ao procedimento de *burn in*.

Estimação de μ , é feito à “mão” pois o R estima a constante como sendo a média.

```
> (est.miu<-(x.ar.yw$x.mean*(1-x.ar.yw$ar))) # estimação de MIU
[1] 4.834129
```

Estatísticas Descritivas do estimador de ϕ , em formato matricial

```
M.Descriptives_Res<-matrix(c(x.ar.yw$ar,
                             mean(x.ar.yw$res[-1]),
                             sd(x.ar.yw$res[-1]),
                             var(x.ar.yw$res[-1])), 1,4)
                             # Definição da Matriz

colnames(M.Descriptives_Res) <- c(" Estim. Coef.", " Mean", " Standart Deviation", " Variance") # nomes colunas
rownames(M.Descriptives_Res) <- c("yw - ") # nomes linhas
M.Descriptives_Res # matriz

  Estim. Coef.      Mean      Standart Deviation      Variance
yw -    0.4178466  0.001414418      1.039781      1.081144

print(round(M.Descriptives_Res, digits=5)) # matriz com valores arredondados a 5 casas decimais
  Estim. Coef.      Mean      Standart Deviation      Variance
yw -    0.41785  0.00141      1.03978      1.08114
```

Os intervalos de confiança para os coeficientes estimados por cada um dos métodos são :

```
M.I.C.P.Est<-matrix(c(x.ar.yw$ar,
                     x.ar.yw$ar-1.96*sqrt(x.ar.yw$asy.var),
                     x.ar.yw$ar + 1.96 * sqrt(x.ar.yw$asy.var)),
                     1,3)
                     #definição da matriz

colnames(M.I.C.P.Est) <- c(" Estim. Coef.", " Low. Lim. C.I.", " Sup. Lim. C.I.") #nomes das colunas
rownames(M.I.C.P.Est) <- c("yw - ") # nomes das linhas
M.I.C.P.Est # matriz de resultados

  Estim. Coef.      Low. Lim. C.I.      Sup. Lim. C.I.
yw -    0.4178466      0.3615079      0.4741853

round(M.I.C.P.Est, digits=3) # matriz de resultados, arredondados a 3 decimas
  Estim. Coef.      Low. Lim. C.I.      Sup. Lim. C.I.
yw -    0.418      0.362      0.474
```

O coeficiente estimado está dentro do intervalo de confiança, intervalo esse que não inclui o zero, pelo que se inclui o coeficiente no modelo.

O valor da estatística de teste de phi calcula-se da seguinte forma,

```
> (Est.Teste.x.ar.yw <- ((x.ar.yw$ar)/(sqrt(x.ar.yw$asy.var)))) # Valor da estatística de teste
[1]
[1,] 14.53671
```

Donde resulta o seguinte *p-value* para a estatística de teste de phi

```
> (p.value.x.ar.yw <- 2*(1 - pnorm(abs(Est.Teste.x.ar.yw), lower.tail = TRUE, log.p = FALSE)))
[1]
[1,] 0
```

Como o *p-value* é zero, o parâmetro não é retirado do modelo, que se pode apresentar em forma matricial,

```
M.Est.Teste.P.Value <- matrix(c(x.ar.yw$ar,                               # definição da matriz
                               Est.Teste.x.ar.yw,
                               p.value.x.ar.yw), 1,3)
colnames(M.Est.Teste.P.Value) <- c(" Estim. Coef", " Est. de Teste", " P-Value") # nomes das colunas
rownames(M.Est.Teste.P.Value) <- c("yw - ") #nomes das linhas
M.Est.Teste.P.Value #matriz de resultados
      Estim. Coef  Est. de Teste  P-Value
yw -    0.4178466    14.53671      0

round(M.Est.Teste.P.Value, digits=3) # matriz de resultados arredondados
      Estim. Coef  Est. de Teste  P-Value
yw -    0.418     14.537      0
```

Teste à significância da constante, através da média da série

O R não permite realizar o teste de significância da constante, como tal realiza-se um *t.test* para testar a nulidade da média da série. Ou seja,

$$\begin{cases} H_0 : \mu_x = 0 & \text{Caso se rejeite } H_0 \Rightarrow \mu_x \neq 0 \\ H_1 : \mu_x \neq 0 & \text{Caso não se rejeite } H_0 \Rightarrow \mu_x = 0 \end{cases}$$

Num processo AR(1) $\mu_x = E[X_t] = \frac{\mu}{1-\phi}$, e se $\mu_x = 0 \Leftrightarrow \mu = 0$, pois não se conhece o erro padrão constante.

A média da série é

```
> mean(x)
[1] 8.303875
```

e o teste aplica-se da seguinte forma,

```
> t.test(x)
One Sample t-test
data: x
t = 229.4669, df = 1000, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 8.232862 8.374887
sample estimates:
mean of x
8.303875
```

Vamos representar e analisar os resíduos. Caso o modelo seleccionado se adequê, os resíduos deverão verificar ter um comportamento semelhante a um ruído branco.

Código para construção do título dos resíduos

```
names1 <- c('Resíduos - ','phi','Burg', 'Yule Walker','Mínimos Quadrados', 'Máxima Verosimilnança', 'Histograma - Resíduos -
Método de')

vals1 <- c("", "", round((x.ar.yw$ar), digits=3), "")

METH.expr2 <- bquote      (.as.name(names1[1])) ~"
                        "~.(as.name(names1[4]))~"
                        "~tilde(.as.name(names1[2]))[1]==.(vals1[3])
```

Estamos a utilizar os mesmos nomes utilizados no anexo III, mas recorrendo só ao método de *Yule-Walker*. Estamos assim a não utilizar algumas posições definidas na variável *names*.

Código para realizar gráfico do resíduos

```
plot(x.ar.yw$res, main=METH.expr2,cex.main =.8)
abline(h= mean(x.ar.yw$res[-1]), col = 2)
abline(h=mean(x.ar.yw$res[-1])+ 1.96 * sd(x.ar.yw$res[-1]), col = 4)
abline(h=mean(x.ar.yw$res[-1])- 1.96 * sd(x.ar.yw$res[-1]), col = 4)
legend("topright", legend = c(" Resíduos", "Média", "I.C."),col= c(1, 2, 4), lty=c(1,1,1),cex=0.7)
```

Código para construção do título do histograma

```
names2 <- c('Histograma Resíduos - ','phi','Burg', 'Yule Walker','OLS', 'MLE')

vals2 <- c("", "", round((x.ar.yw$ar), digits=3), "", "")
METH.HIST.expr2 <- bquote (.as.name(names2[1])) ~"
                        "~.(as.name(names2[4]))~"
                        "~tilde(.as.name(names2[2]))[1]==.(vals2[3])
```

Código para representar o histograma

```
hist(x.ar.yw$res[-1], prob=TRUE,main=METH.HIST.expr2,cex.main =0.8)
lines(density(x.ar.yw$res[-1]), col= 2)
```

Código para construir título do Box-Plot

```
names3 <- c( 'BOX - Plot Resíduos - ', 'phi', 'Burg', 'Yule Walker', 'OLS', 'MLE')

vals3 <- c("", "", round((x.ar.yw$ar), digits=3), "", "")
METH.BP.expr2 <- bquote  (.(as.name(names3[1])) ~"
                        "~.(as.name(names3[4]))~"
                        "-~tilde(.(as.name(names3[2])))[1]==.(vals3[3])
```

Código para representar o Box-Plot

```
boxplot(x.ar.yw$res[-1], main=METH.BP.expr2, cex.main =0.8)
```

Código para construir o título do QQ-Plot

```
names4 <- c( 'QQ - Plot Resíduos - ', 'phi', 'Burg', 'Yule Walker', 'OLS', 'MLE')
vals4 <- c("", "", round((x.ar.yw$ar), digits=3), "", "")
METH.QQP.expr2 <- bquote  (.(as.name(names4[1])) ~"
                        "~.(as.name(names4[4]))~"
                        "-~tilde(.(as.name(names4[2])))[1]==.(vals4[3])
```

Código para representar o QQ-Plot

```
qqnorm(x.ar.yw$res[-1], main = METH.QQP.expr2, col= "blue",cex.main =0.8) # normal Q-Q plot
qqline(x.ar.yw$res[-1], col = "red")
```

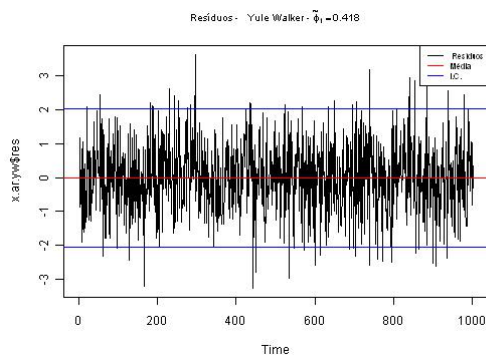


Figura A.IV.7: FAC do processo gerado

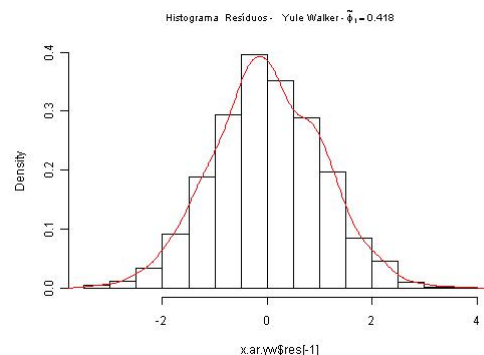


Figura A.IV.8: FACP do processo gerado

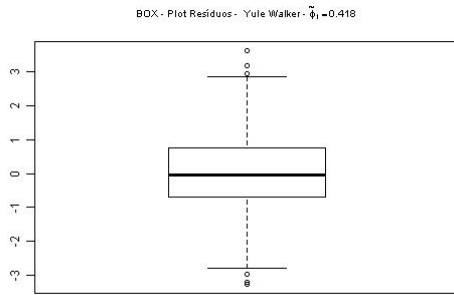


Figura A.IV.9: QQ-Plot do processo gerado

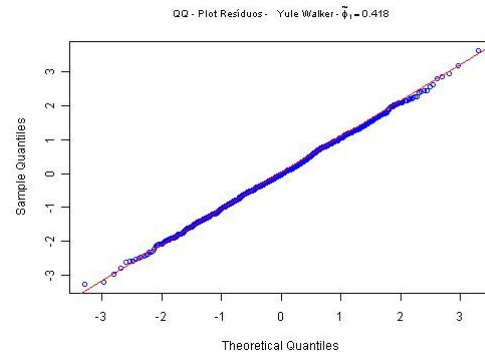


Figura A.IV.10: PP-Plot do processo gerado

Pelas figuras Figura A.IV.7 a Figura A.IV.10 verifica-se que os resíduos têm um comportamento semelhante ao de um ruído branco.

Código que constrói o título para a FAC

```
names5 <- c('FAC Resíduos -','phi','Burg', 'Yule Walker','OLS', 'MLE')
vals5 <- c("", "", round((x.ar.yw$ar), digits=3), "", "")
METH.FAC.expr2 <- bquote (.(as.name(names5[1])) ~"
                        "~.(as.name(names5[4]))~"
                        "~tilde(.(as.name(names5[2])))[1]==.(vals5[3]))
```

Código que representa a FAC

```
acf(x.ar.yw$res[-1], main = METH.FAC.expr2, cex.main =0.5)
```

Código que constrói o título para a FACP

```
names6 <- c('FACP Resíduos -','phi','Burg', 'Yule Walker','OLS', 'MLE')
vals6 <- c("", "", round((x.ar.yw$ar), digits=3), "", "")
METH.FACP.expr2 <- bquote (.(as.name(names6[1])) ~"
                        "~.(as.name(names6[4]))~"
                        "~tilde(.(as.name(names6[2])))[1]==.(vals6[3]))
```

Código que representa a FACP

```
pacf(x.ar.yw$res[-1], main = METH.FACP.expr2, cex.main =0.5)
```

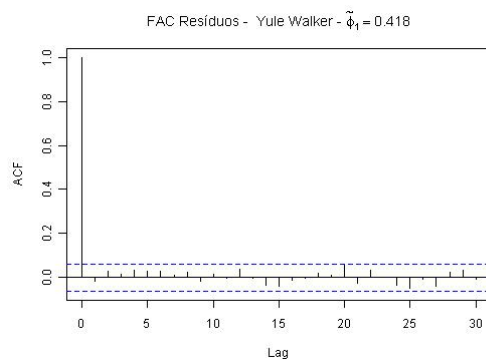


Figura A.IV.11: FAC dos resíduos

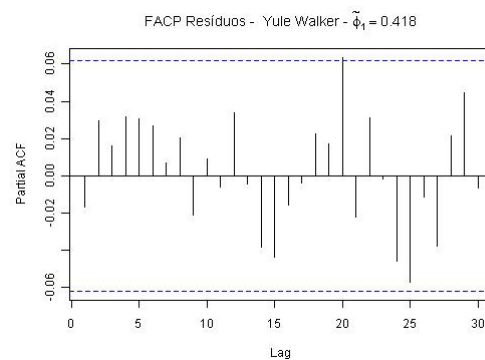


Figura A.IV.12: FACP dos resíduos

Pela visualização das FAC e FACP dos resíduos, não se rejeita a hipótese de os resíduos constituírem ruído branco, o que se pode verificar analiticamente pela realização de testes de normalidade.

Testes de Normalidade

Os testes de normalidade efectuam-se recorrendo à livreria “nortest”, que se instala da seguinte forma,

```
install.packages("nortest")      # download da livreria nortest
library(nortest)                # carregar a livreria no workspace
```

O código para aplicar o *Lilliefors (Kolmogorov-Smirnov) normality test* é o seguinte

```
> lillie.test(x.ar.yw$res[-1])
      Lilliefors (Kolmogorov-Smirnov) normality test
data:  x.ar.yw$res[-1]
D = 0.0182, p-value = 0.5882
```

O valor de *p-value* é superior a 5%, os resíduos são normais.

O que se pode confirmar através do Teste de Box-Pierce

```
> Box.test(x.ar.yw$res[-1], lag = 1, type = "Box-Pierce")
      Box-Pierce test
data:  x.ar.yw$res[-1]
X-squared = 0.2855, df = 1, p-value = 0.5931

> Box.test(x.ar.yw$res[-1], lag = 1, type = "Ljung-Box")
      Box-Ljung test
data:  x.ar.yw$res[-1]
X-squared = 0.2863, df = 1, p-value = 0.5926
```

Apesar de existirem outliers moderados na figura A:IV.7, pelos testes de *Kolmogorov* com a correcção de *Lilliefors* e o teste de *Box-Pierce* obtêm-se *p-values*, que não mostram evidência estatística para que os resíduos não sejam normais.

Dado não se ter rejeitado a normalidade dos resíduos, passemos à realização do teste de *t* para testar a nulidade da média dos resíduos.

```
> t.test(x.ar.yw$res[-1])
      One Sample t-test
data:  x.ar.yw$res[-1]
```

```
t = 0.043, df = 999, p-value = 0.9657
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
-0.06310883 0.06593767
sample estimates:
mean of x
0.001414418
```

O teste t, aplicado aos resíduos apresenta um valor p do teste (0.97), o que permite concluir que não há evidência estatística de que a média dos resíduos não seja nula. O valor p é muito próximo de 1 o que permite atestar quase certamente a validade da hipótese nula. Repare-se que para uma confiança de 95%, o erro de estimativa relativamente a 0 do valor médio dos resíduos é inferior a 7%.

Código para construir nomes para título de gráfico de serie simulada versus série ajustada e título de gráfico de previsão

```
names7 <- c('Série gerada vs. ajustada ','phi','Burg', 'Yule Walker','OLS', 'MLE', 'Previsão')

vals7 <- c(PHI,"",round((x.ar.yw$ar), digits=3),"", "")
PREV.expr2 <- bquote      (.(as.name(names7[1])) ~"
                          "~.(as.name(names7[4]))~"
                          "-~tilde(. (as.name(names7[2])))[1]==.(vals7[3])~"
                          "-//-"~.(as.name(names7[2]))[1]==.(vals7[1]))
PREV.expr6 <- bquote      (.(as.name(names7[7])) ~"
                          "~.(as.name(names7[4]))~"
                          "-~tilde(. (as.name(names7[2])))[1]==.(vals7[3])~"
                          "-//-"~.(as.name(names7[2]))[1]==.(vals7[1]))
```

Código para representar serie simulada versus série ajustada

```
plot(x, type = "l", main=PREV.expr2, col = 1)
z <- numeric()
for (t in 2:1000) z[t] <- est.miu + x.ar.yw$ar * x[t - 1]
points(z, col= 2, type = "o", main=(expression(AR(1)~--phi==PHI)))
```

Código para cálculo de limites de janela gráfica de Previsão

```
x.yw.fore = predict(x.ar.yw, n.ahead = 50)
summary(x.yw.fore)
x.yw.fore$pred
x.yw.fore$se
U.yw=x.yw.fore$pred + 2* x.yw.fore$se
U.yw
L.yw=x.yw.fore$pred - 2* x.yw.fore$se
```

```
L.yw
minx.yw=min(x,L.yw)
maxx.yw=max(x,U.yw)
minx.yw
maxx.yw
```

Código para representar gráfico de Previsão

```
ts.plot(x, x.yw.fore$pred,main=PREV.expr6, col=1:2, ylim=c(minx.yw, maxx.yw)) # só funciona de passar x a ts, feito nas
#linhas acima

lines(U.yw, col= "blue", lty = "dashed")
lines(L.yw, col= "blue", lty = "dashed")
legend("topright", legend = c("Valores Observados", "Previsão", "Limites"),col= c(1,2, 4), lty=c(1,1,1))
```

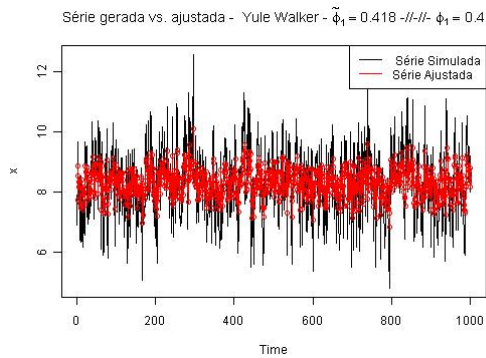


Figura A.IV.13: série gerada vs. Serie ajustada

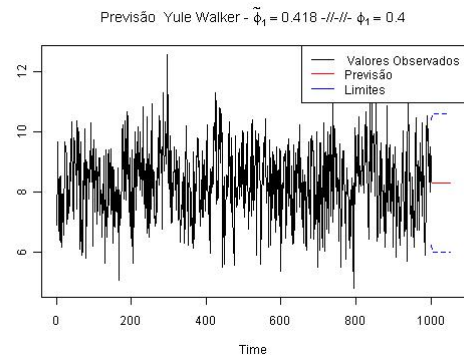


Figura A.IV.14: Previsão

Anexo V

Resumo de comandos mais importantes

Neste anexo apresentam-se de forma sucinta alguns dos comandos do R.

Alguns são específicos de livrarias necessárias para a análise de séries cronológicas, outros são de âmbito geral. A ordem pela qual são apresentados segue a lógica de aplicação utilizada ao longo deste trabalho.

O WORKSPACE

O Workspace é onde são guardados todos os objectos do R que estão disponíveis para utilização. Quando se inicia uma sessão poderão existir conjuntos de dados e livrarias carregadas no workspace, pelo que se deve verificar o que está disponível. Para se ter um controlo eficaz das análises, da atribuição de nomes a objectos, etc devem utilizar-se os seguintes comandos:

```
ls()           # lista os objectos carregados no workspace

library()     # Mostra todas a livrarias carregadas no workspace

data()        # Mostra todos os conjuntos de dados de todas a
              #livrarias carregadas no workspace

rm(list = ls(all = TRUE)) # Limpa todos os objectos do workspace

options(digits=3) # estabelece o número de casa decimais em 3

;             # Permite executar mais do que um comando por linha desde
              # que separados por ;
```

Caminho por defeito

A leitura de ficheiros de dados, e a gravação de ficheiros de output é realizada para uma pasta definida por defeito, é possível alterar a pasta de leitura e escrita por defeito.

```
getwd()       # Mostra o caminho absoluto da pasta por defeito
              # (current working directory of the R process)
```

```
# utilizada para ler e escrever ficheiros

list.files()      # lista todos os ficheiros e pastas na working directory

setwd()          # Permite estabelecer uma working directory

dir.create()     # Cria pasta na working directory

class()          # identifica o tipo de objecto com que se está a
                 # trabalhar, se é numérico, data frame, série temporal
                 # etc.
```

Livrarias necessárias que não estão incluídas na versão base de instalação do R.

Alguns comandos específicos para a realização de análise de sucessões cronológicas estão incluídos em livrarias que não estão incluídas na versão base, que é instalada por defeito quando se instala o R. Para que se possam utilizar deve-se instalar a livraria necessária, de seguida mostra-se como instalar as livrarias utilizadas aos longo do trabalho.

```
install.packages("stats")  # Realiza o download da livraria "stats"
library(stats)             # carrega a livraria no workspace

install.packages("TSA")    # Realiza o download da livraria "TSA"
library(TSA)               # carrega a livraria no workspace

install.packages("nortest") #Realiza o download da livraria "nortest"
library(nortest)          # carrega a livraria no workspace

install.packages("tseries") #Realiza o download da livraria "tseries"
library(tseries)         # carrega a livraria no workspace

install.packages("lmtest") #Realiza o download da livraria "lmtest"
library(lmtest)          # carrega a livraria no workspace

install.packages("RODBC")  # Realiza o download da livraria "RODBC"
library(RODBC)            # carrega a livraria no workspace
```

Ler e escrever Ficheiros

```
read.table      # Lê um ficheiro de dados do tipo table para o workspace

read.csv()     # Lê ficheiro do tipo csv

attach()       # no caso de os dados estarem em tabela, resultado do
```

```

# comando anterior torna possível ler o nome das colunas
# como variáveis

detach()      # volta a colocar os dados como table (frame)

write(x, file="Teste.txt",ncolumns=1)      # Escreve um ficheiro de texto
                                             # com o nome Teste.txt

shell("Teste.txt") # Abre o ficheiro Teste.txt com o notepad.

shell.exec("http://www.dmat.uevora.pt/ensino/mmead/") # abre a página de
                                                       #internet do MMEAD

Sys.Date()    # Devolve a data do sistema

source()      # Carrega no workspace um ficheiro de script

pdf()         # Cria um ficheiro pdf (utilizado em conjunto com linha
# abaixo
dev.off()     # fecha a criação de pdf

jpeg()        # cria um ficheiro do tipo jpeg
dev.off()     # fecha o processo de criar um ficheiro do tipo jpeg

```

- ler um ficheiro de Excel (tem de se ter carregada a livraria RODBC)

```

Ligacao <- odbcConnectExcel("R_EXCEL.xls") # estabelece a ligação
                                           # entre o R o Ficheiro de
                                           # Excel

sqlTables(Ligacao)      # Verifica o conteúdo do ficheiro de excel

sh1 <-sqlFetch(Ligacao, "Sheet1") # atribui os dados à variável sh1

attach(sh1)             # permite utilizar o nome de cada coluna como
# se fosse uma variável

```
- ler um ficheiro de Access (tem de se ter carregada a livraria RODBC)

```

LigaAccess <-odbcConnectAccess("bd.mdb") # estabelece a ligação
                                           # entre o R o Ficheiro de
                                           # Access

sqlTables(LigaAccess)  # Verifica o conteúdo do ficheiro de Access

Dados <- sqlQuery (LigaAccess, "select tabela.campo1,tabela.campo2
from tabela")         # executa uma query e guarda na variável
# Dados o campo1 e campo2 de tabela

attach(Dados)         # permite utilizar campo1 e campo2 como
# variáveis

```

```
mean()           # Média da variável
var()            # Variância com denominador n-1
sd()            # Desvio-Padrão
cov()           # Covariância com denominador n-1
cor()           # Correlação

summary()       # apresenta o Mínimo, 1º, 2º e 3º quartil, a
                # média e o máximo
```

Geração de Números Aleatórios e construção de processos

```
set.seed(1)     # fixa a semente de geração de números aleatórios
                # de forma a que a que a simulação possa ser
                # reproduzida

seq()           # gera uma sequência

rnorm()         # simula um ruído branco gaussiano

diff()          # diferencia uma série cronológica

polyroot()     # Extrai raízes de um polinómio

x <- w <- rnorm(1000) # atribui à variável x e w 1000 números aleatórios
                    # gerados por uma distribuição Normal(0,1).

for (t in 2:1000) x[t] <- PHI * x[t - 1] + w[t] # utilização de um ciclo
                                                # for para construir um
                                                # processo AR(1).

for (t in 2:1000) x[t] <- w[t] + THETA * w[t-1] # utilização de um
                                                # ciclo for para
                                                # construir um
                                                # processo MA(1).

BoxCox.ar      # Determina a transformação de Box - Cox mais apropriada
                # para os dados

arima.sim()    # função integrada na livraria "stats" que simula
                # processo do tipo arima

ts(w)          # transforma w num objecto do tipo time series
```

Representação Gráfica

```
par(mfrow = c(3,2)) # divisão da janela gráfica em 3 linhas por 2
                    # colunas, onde em cada célula fica um objecto
                    # (gráfico)

plot(x)         # representa graficamente x
abline(h=0, col = "red") # representa uma linha horizontal de cor vermelha
                        # no gráfico resultante de plot(x)
points()       # acrescenta pontos ao plot
```

```

hist(x, prob=TRUE, 12, col = "red") # representa um histograma
lines(density(x), col= "blue")      # acrescenta curva de densidade normal
                                     # ao histograma anterior, em cor azul
abline(v= mean(x), col = "blue")    # representa a média numa linha
                                     # vertical no histograma
points()                             # acrescenta pontos ao plot

qqnorm(x, col= "blue")               # Normal QQ plot
qqline(x, col = "red")               # adiciona linha de regressão ao QQ - plot

acf(x)                               # representa a função de autocorrelação

pacf(x)                              # representa a função de autocorrelação parcial

win.graph()                          # abre uma nova janela de gráficos

ts.plot(y)                           # representa o cronograma de um objecto y quando
                                     # este não é do tipo série temporal

y <- ts(u, start = c(1996, 1), freq = 12) # define que o objecto u é uma
                                           # série temporal com inicio no
                                           # mês de Janeiro de 1996 e com
                                           # frequência 12, ou seja, define
                                           # que os dados têm inicio em
                                           # Janeiro de 1996 e são mensais.

start(y)                             # Inicio da série
end(y)                                # Fim da série
frequency(y)                          # Frequência da Série
time(y)                               # Mostra dados da série no tempo

```

Estimação

```

ar()                                  # estimação de somente para processos autoregressivos
arma)                                 # estimação para processos ar, ma, arima

```

Intervalos de confiança para estimadores

```

confint()                             # indica o intervalo de confiança a 95% para
                                     # cada um dos coeficientes estimados

```

Testes aos Resíduos

```

lillie.test()                         # Teste de kolmogorov-Smirnov com correcção de Lilliefors
Box.test()                             # Teste de Box- Pierce

```


Bibliografia

- Adler**, Joseph (2010): “ *R in a Nutshell*”, O’Reilly.
- Agresti**, Alan (2007): “ *An Introduction to Categorical Data Analysis*”, Wiley
- Alzaid**, A. A., Al-Osh, M (1990):” An integer-Valued pth-order Autoregressive Structure (INAR(p)) Process”, *J. Appl. Prob.* 27, 314-324
- Brannas**, Kurt (1994):” Estimation and Testing in Integer-Valued AR(1) Models, *Umea Economic Studies*, 335, 1994.
- Braun**, W. John, **Murdoch**, J. Duncan (2007): *A First Course in Statistical Programming with R*, Cambridge University Press.
- Box**, G. E. P. e **Jenkins**, G. M. (1976):” *Time Series Analysis, Forecasting and Control*”, 2^a. Ed., Holden-Day.
- Brockwell**, Peter J. e **Davis**, Richard A. (1991): “*Time Series: Theory and Methods*”, 2nd. Ed., Springer Verlag.
- Brockwell**, Peter J. e **Davis**, Richard A. (2002): “*Introduction to Time Series and Forecasting*”, Second Edition, Springer.
- Cowpertwait**, Paul S. P., **Metcalf**, Andrew V. (2009): “*Introductory Time Series with R*”, Springer.
- Crawley**, Michael J. (2007) : “*The R Book*”, Wiley.
- Cryer**, Jonathan D., **Chan**, Kung-Sik (2008): “ *Time Series Analysis with Applications in R*”, Second Edition, Springer.
- Dalgaard**, Peter (2008): “*Introductory Statistics with R*”, Second Edition, Springer
- Feller**, William(1970): “*An Introduction to Probability Theory and Its Applications, Volume I*, 3rd Edition (Revised Printing)”, John Wiley & Sons.
- Gomes**, Dulce (2005):” *Cadeira de Séries Temporais*”, Universidade de Évora.
- Harvey**, Andrew C. (1993): “*Time Series Models*”, 2nd. Ed., Harvester Wheatsheaf.
- Kendall**, Maurice e **Ord**, J. Keith (1990): “*Time Series*”, Edward Arnold.
- Latour**, Alain, (1998): “ Existence and Stochastic Structure of a Non-Negative Integer Valued Autoregressive Process”, *Journal of Time Series Analysis* Vol. 19, No. 4

- Matloff**, Norman, (2008): "*R for Programmers*", University of California, Davis.
- McKenzie**, Ed, (1985): "*Some simple models for discrete variate time series*", Water Resources Bulletin, 21, 645 – 650.
- Montgomery**, Douglas C., **Johnson** A. Lynwood e Gardiner, John S. (1990): "*Forecasting & Time Series Analysis*", 2nd. Ed., McGraw-Hill.
- Montgomery**, Douglas C., **Runger**, George C., (1994): "*Applied Statistics and Probability for Engineers*", John Wiley & Sons, Inc.
- Murteira**, Bento J. F (1990): "*Probabilidades e Estatística, Volume I, 2ª Edição*", McGraw-Hill de Portugal.
- Murteira**, Bento J. F (1990): "*Probabilidades e Estatística, Volume II, 2ª Edição*", McGraw-Hill de Portugal.
- Murteira**, Bento J. F, **Muller**, Daniel A. e Turkman, K. Feridum (1993): "*Análise de Sucessões Cronológicas*", McGraw-Hill de Portugal.
- Pindyck**, Robert S., **Rubinfeld**, Daniel L. (1988) : "*Econometric Models and Economic Forecasts*", Second Edition, McGraw-Hill.
- Pitman**, Jim (1995): "*Probability*", Springer – Verlag.
- Silva**, Isabel Maria Marques da, (2005): "*Contributions to the analysis of discrete-valued time series*", Tese submetida à Faculdade de Ciências da Universidade do Porto para obtenção do grau de Doutor em Matemática Aplicada, Departamento de Matemática Aplicada, Faculdade de Ciências da Universidade do Porto.
- Silva**, Nélia, Pereira, Isabel, Silva, M. Eduarda (2009)", *REVSTAT – Statistical Journal* Volume 7, Number 1, April 2009, 119-134.
- Shumway**, Robert H., **Stoffer**, David S. (2006): "*Time Series Analysis And Its Applications with R Examples*", Second Edition, Springer
- Tiago de Oliveira**, J.(1990): "*Probabilidades e Estatística, Volume I*", McGraw-Hill de Portugal.
- Tiago de Oliveira**, J.(1990): "*Probabilidades e Estatística, Volume II*", McGraw-Hill de Portugal.
- Torgo**, Luís, (2006): "*Introdução à Programação em R*", Faculdade de Economia, Universidade do Porto.
- Tsai**, Ruey S. (2005): "*Analysis of Financial Time Series*", Second Edition, Wiley.
- Wei**, William W. S. (1994): "*Time Series Analysis, Univariate and Multivariate Methods*", Addison-Wesley.