INSTITUTO
UNIVERSITÁRIO
DE LISBOA

Data Science na Modelação e Previsão de Séries Económico-financeiras: das Metodologias Clássicas ao Deep Learning

Filipe Roberto de Jesus Ramos

Doutoramento em Gestão, na especialidade em Métodos Quantitativos

Orientadoras:
Doutora Diana Elisabeta Aldea Mendes, Professora Associada, ISCTE – Instituto Universitário de Lisboa

Doutora Anabela Ribeiro da Costa, Professora Auxiliar, ISCTE – Instituto Universitário de Lisboa

Fevereiro, 2021

BUSINESS
SCHOOL

Departamento de Métodos Quantitativos de Gestão e Economia

Data Science na Modelação e Previsão de Séries Económico-financeiras: das Metodologias Clássicas ao Deep Learning

Filipe Roberto de Jesus Ramos

Doutoramento em Gestão, na especialidade em Métodos Quantitativos

Júri:

Doutora Elizabeth A. Reis, Professora Catedrática, ISCTE – Instituto Universitário de Lisboa (Presidente)

Doutora Andreia T. Dionísio, Professora Associada, Escola de Ciências Sociais – Universidade de Évora

Doutor Fernando M. Gonçalves, Professor Auxiliar, ISEG Lisbon School of Economics and Management – Universidade de Lisboa

Doutor Manuel A. Ferreira, Professor Catedrático, ISCTE – Instituto Universitário de Lisboa

Doutora Diana E. Mendes, Professora Associada, ISCTE – Instituto Universitário de Lisboa

Fevereiro, 2021

O limite do potencial da máquina é definido pelo pensamento do Homem

FLR

AGRADECIMENTOS

The words of the prophets are written on the subway walls
And tenement halls
And whispered in the sound of silence
The Sound of Silence – Simon & Garfunkel

É no som de um silêncio e com o tão simples, mas tão cheio de tudo, sentimento de gratidão que escrevo estas palavras. Porém, mais importante do que escrevê-las numa página, é sentir o eco silencioso que cada uma delas fez ao ser gravada naquela 'parede' onde cada um é profeta e da qual jamais serão apagadas – a memória.

Iniciando pela parte académica, às minhas orientadoras, Professora Anabela Costa e Professora Diana Mendes, agradeço as sugestões construtivas, o apoio e as palavras de incentivo na fase inicial deste trabalho. Ao Professor José Dias afirmo que não será esquecido o cuidado tido nos momentos complicados que marcaram o iniciar deste percurso. À Professora Elizabeth Reis agradeço o ter sido efetivamente ativa para que a submissão deste trabalho se concretizasse. Semelhante agradecimento expresso a outros colegas do ISCTE que se fizeram sentir presentes neste percurso, trazendo a mensagem de que não estamos sós. Não listando nesta página todos os nomes, é neles que agora penso, fazendo questão de, pessoalmente e em momento oportuno, lhes dizer o quão grato estou.

Expresso igual gratidão a alguns professores que me acompanham há anos: Professoras(es) Amélia Fonseca (FCUL), Ana Sá (FCT-UNL), Fernando Gonçalves (ISEG), Filipa Carvalho (ISEG), Gracinda Gomes (FCUL), Helena Santos (FCT-UNL), Isabel Ferreirim (FCUL), Júlia Carvalho (FCT-UNL), Mário Branco (FCUL), Purificação Coelho (FCUL) e Teresa Alpuim (FCUL).

Na esfera pessoal, deixo especial agradecimento aos meus amigos em geral, destacando o apoio incondicional de alguns: Beatriz L., Cidália A., Conceição F., David S., Fernando S., Lúcia V., Mário C., Paula P., Sónia M. e a sempre presente Tina. Mais, quero agradecer aos igualmente especiais "Faltas cá tu… ou não!", "Grupinho do café", "Holanda sempre a fundo" e aos que estão sempre aqui 'ao lado', "Patita&Companhia". Tudo seria bem mais complicado sem os nossos momentos.

Expresso especial gratidão a duas pessoas que, durante a escalada ingreme e árdua, me traziam a coragem e aquela calma que apenas se sente em 'planícies' serenas. Alexandra e Didier, agradeço-vos por estarem TÃO presentes, recordando as longas horas passadas naquela janela *Skype*.

Finalmente, um sentido agradecimento à minha família, lembrando quem sempre esteve comigo, me ensinou a interpretar o mundo, me criticou com o intento da construção, me deu alento nos momentos de desânimo e cansaço, … enfim, que se mostraram com infinita paciência e um apoio incondicional. Vocês são os meus mestres neste que é o doutoramento mais valioso – o "Saber Viver".

Mãe, …

A todos, o meu sincero OBRIGADO por me fazerem acreditar que este era o caminho.

RESUMO

A articulação de técnicas/ferramentas estatísticas, matemáticas e computacionais, no processo de análise, modelação e previsão de séries temporais, manifesta-se um claro suporte de apoio à tomada de decisão. O constante desafio na procura de previsões acuradas tem levado os investigadores à melhoraria das técnicas já existentes e a investir na procura de metodologias alternativas. Especificamente para séries económico-financeiras, a aplicação de metodologias baseadas em Inteligência Artificial, em particular de *Deep Learning*, tem sido apontada com uma opção promissora.

Neste estudo faz-se uma comparação crítica dos resultados obtidos por aplicação de metodologias clássicas de previsão (nomeadamente modelos autorregressivos e de alisamento exponencial) e de *Deep Learning* (mediante a implementação de algumas arquiteturas redes neuronais). O estudo empírico foi sustentando em quatro séries económico-financeiras distintas: *Consumer Price Index for All Urban Consumers: All Items in U.S. City Average* (CPIAUCSL)*; Vehicle-Miles Travelled* (VMT)*; Portuguese Stock Index 20* (PSI 20) e *Standard & Poor's 500 Exchange-Traded Fund* (SPY). A análise comparativa é feita tendo por base a qualidade preditiva e o custo computacional associado a cada um dos modelos de previsão.

Reconhecidas vantagens na aplicação das metodologias de *Deep Learning*, são discutidas possíveis alterações procurando melhorar a qualidade preditiva e reduzir o tempo de execução computacional. As alterações introduzidas em modelos de redes neuronais revelaram-se promissoras na redução do tempo computacional e nos valores da métrica de erro de previsão usada. Este sucesso é sobretudo evidente em séries que apresentam dinâmicas 'irregulares', como são exemplo as séries financeiras.

Palavras–chave: *Data Science*; *Deep Learning*; Séries Temporais; Modelos ARMA, ETS e DNN; Previsão; Erro de Previsão

Classificação JEL: C01, C02, C10, C22, C45, C53, C58, C60, G17, M20

ABSTRACT

The articulation of statistical, mathematical and computational techniques/tools, in the process of analysis, modelling and forecasting time series, manifests clear support for decision making. The constant challenge in the quest for the most accurate results possible has led researchers not only to improve the existing techniques, but also to invest in the search for alternative methodologies. Specifically, for economic and financial series, the application of methodologies based on Artificial Intelligence, in particular Deep Learning, has been pointed out as a promising option.

This study makes a critical comparison of the results obtained by applying classical forecasting methodologies (namely autoregressive models and exponential smoothing) and Deep Learning (through the implementation of some neural network architectures). The empirical study focused on four economic-financial series with different characteristics: Consumer Price Index for All Urban Consumers: All Items in U.S. City Average (CPIAUCSL); Vehicle-Miles Travelled (VMT); Portuguese Stock Index 20 (PSI 20) and Standard & Poor's 500 Exchange-Traded Fund (SPY). The comparative analysis is made based on both predictive quality and computational cost associated with each of the forecasting models.

Recognized the advantages in the application of Deep Learning methodologies, we discuss some changes to introduce in the existing models to improve their predictive quality while reducing computational execution time. The changes introduced in neural network models proved to be promising in reducing the associated computational time but and the values of the error metric used. This success is especially evident in series with 'irregular' dynamics, as is the case with financial series.

Keywords: Data Science; Deep Learning; Time Series; ARMA, ETS and DNN Models; Forecasting; Prediction Error

JEL Classification: C01, C02, C10, C22, C45, C53, C58, C60, G17, M20

ÍNDICE GERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

LISTA DE ABREVIATURAS

ADF	Teste de raiz unitária de *Dickey-Fuller* aumentado (*Augmented Dickey-Fuller*)
AIC	Critério de informação de Akaike (*Akaike Information Criterion*)
AR	Autorregressivo
$AR(p)$	Modelo autorregressivo de ordem p
ARIMA	Média móvel autorregressiva integrada (*Autoregressive Integrated Moving Average*)
$ARIMA\ (p,d,q)$	Modelo autorregressivo de médias móveis de ordens p e q de uma série integrada de ordem d
ARMA	Média móvel autorregressiva (*Autoregressive Moving Average*)
$\mathrm{ARMA}(p,q)$	Modelo autorregressivo de médias móveis de ordens p e q
BIC	Critério de informação de bayesiano (*Bayesian Information Criterion*)
CPI	Índice de preços ao consumidor (*Consumer Price Index*)
CPIAUCSL	*Consumer Price Index for All Urban Consumers*
CUSUM	Soma cumulativa (*Cumulative Sum*)
DETS	Alisamento exponencial duplo (*Double Exponential Smoothing*)
DF	Teste de raiz unitária de *Dickey-Fuller*
DNN	Redes neuronais com aprendizagem profunda (Deep Neural Networks)
DSP	Processo estacionário em diferenças (*Difference Stationary Process*)
EMA	Média móvel exponencial (Exponential Moving Average)
EWMA	Média móvel exponencial ponderada (*E*xponentially Weighted Moving Average)
ETS	Alisamento exponencial (*Exponential Smoothing*)
ETF	*Exchange-Traded Fund*
FAC	Função de autocorrelação
FACV	Função de autocovariância
HQIC	Critério de informação de Hannan–Quinn (*Hannan–Quinn Information Criterion*)
IA	Inteligência Artificial
iid	Independentes e identicamente distribuídos
KPSS	Teste de raiz unitária de *Kwiatkowski-Phillips-Schmidt-Shin*
LR	Rácio de verosimilhança (*Likelihood Ratio*)
LSTM	Long Short-Term Memory
MA	Média móvel (*Moving Average*)
$MA(q)$	Modelo de médias móveis de ordem q
MLP	Perceptrão multicamada (*Multilayer Perceptron*)
N	Distribuição normal

OLS	Mínimos quadrados ordinários (*Ordinary Least Squares*)
PACP	Função de autocorrelação parcial
PSI 20	*Portuguese Stock Index 20*
RNA	Rede neuronal artificial
RNN	*Recurrent Neural Network*
SARIMA	Média móvel sazonal autorregressiva integrada (*Seasonal Autoregressive Integrated Moving Average*)
SETS	Alisamento exponencial simples (*Single Exponential Smoothing*)
TETS	Alisamento exponencial triplo (*Triple Exponential Smoothing*)
SMA	Média móvel simples (*Simple Moving Average*)
SPY	*Standard & Poor's 500 Exchange-Traded Fund*
TSP	Processo estacionário em tendência (*Trend Stationary Process*)
VMT	*Vehicle-Miles Travelled*
WMA	Média móvel pderada (Weighted Moving Average)
WN	Ruído branco (*White Noise*)

LISTA DE SÍMBOLOS

b_k	*Bias* aplicado ao neurónio k
Cov	Operador covariância
CT_t	Componente do ciclo de tendência numa série temporal
$\mathcal{C}$	Função de custo
$\mathbb{E}$	Operador valor esperado
$f(y\|\Theta_i)$	Função de verosimilhança
$\mathcal{F}_S$	Força de sazonalidade de uma série temporal
$\mathcal{F}_T$	Força de tendência de uma série temporal
k	Número de parâmetros no modelo
k	Número de desfasamentos (*lags*)
ℓ	Número de camadas ocultas da rede neuronal (profundidade da rede)
L	Operador de desfasamento (*lag*)
$\mathcal{L}(\theta)$	Valor da função de máxima verosimilhança do parâmetro θ
ln	Função logaritmo neperiano
Log	Função logaritmo
ℓ_t	Nível (ou valor suavizado) da série no instante t
m	Frequência da sazonalidade
m_i	Número de neurónios na camada i
max	Função máximo
n	Dimensão de uma amostra ou número de observações de uma série
R^2	Coeficiente de determinação
R_t	Componente residual numa série temporal
s_t	Estimativa da sazonalidade no instante t
S_t	Componente sazonal numa série temporal
T	Intervalo de tempo
t	Instante no tempo
$\mathbb{T}_\lambda$	Operador transformação de Box-Cox
t_t	Estimativa da tendência no instante t
u_t	Erro do modelo de regressão linear, no instante t
Var	Operador variância
$\boldsymbol{W}$	Vetor de parametrização de um neurónio
w_{kj}	Peso sináptico da ligação j pertencente ao neurónio k
$\boldsymbol{X}$	Vetor de entrada num neurónio
x_{it}	variável exógena i num modelo de regressão linear, no instante t

$\boldsymbol{Y}$	Vetor de saída de uma rede neuronal
y_k	Sinal de saída do neurónio k
y_t	variável endógena num modelo de regressão linear, no instante t
α	Coeficiente de presença de raiz unitária; Constante de momento; Coeficiente de ponderação do EWMA; Coeficiente de alisamento exponencial para o nível.
β	Coeficiente de regressão; Coeficiente de alisamento exponencial para a tendência
γ	Coeficiente de alisamento exponencial para a sazonalidade
γ_k	Função de autocovariância entre os valores nos instantes t e $t+k$
δ	Coeficiente de amortecimento no alisamento exponencial
$\mathcal{S}_{\text{AIC}}$	Valor da estatística AIC
$\mathcal{S}_{\text{BIC}}$	Valor da estatística BIC
$\mathcal{S}_{\text{HQIC}}$	Valor da estatística HQIC
$\delta_i(t)$	Gradiente local do neurónio i no instante t
Δ	Operador diferença
ε_t	Componente residual de um modelo quando os erros são iid
η	Taxa de aprendizagem
$\theta_q(L)$	Polinómio de médias móveis de ordem q
ϑ	Parâmetro de decisão da função de ativação
λ	Parâmetro da transformação de *Box-Cox*
μ	Média aritmética
$\hat{\mu}$	Estimador para a média
v_t	Potencial de ativação
ξ_t	Passeio aleatório
ρ_k	Coeficiente de autocorrelação linear entre os valores nos instantes t e $t+k$
$\phi_p(L)$	Polinómio autorregressivo de ordem p
$\hat{\sigma}^2$	Estimador para a variância
σ^2	Variância
τ	Estatística do teste de raiz unitária
φ	Coeficiente de correlação parcial; função de ativação
$\{y_t\}_{t\in T}$	Série temporal y no instante t pertencente ao intervalo de tempo T
$\mathbf{1}_t$	Função Heavyside
∞	Símbolo de infinito
$\in$	Símbolo para “pertence ao conjunto”
$\sim$	Símbolo para “segue uma distribuição estatística”
H_0	Símbolo para “hipótese nula”
H_1	Símbolo para “hipótese alternativa”
∂	Símbolo de derivada parcial
$\mathbb{R}$	Conjunto dos números reais
$\forall$	Símbolo para “qualquer que seja”

INTRODUÇÃO

Face à crescente globalização da economia, assiste-se a uma conjuntura cada vez mais competitiva e, consequentemente, a uma forte concorrência entre organizações. Deter conhecimento científico e o domínio de técnicas e métodos de previsão pode ser determinante para o sucesso. Assim, face à forte aplicabilidade em contextos reais, não será de estranhar o crescente interesse em temáticas como a de modelação e previsão, muito em particular nas áreas da Economia, Gestão e Finanças.

Certos desses factos, estudar o fenómeno de previsão e os vários aspetos a ele inerentes constitui um desafio para o conhecimento científico. Este interesse tem levado inúmeros investigadores a desenvolverem estudos que visam não só melhorar e compreender modelos já existentes, como desenvolver novos modelos testando/procurando padrões de aplicabilidade. É precisamente nesta linha que surge a nossa investigação que, tendo por base um referencial teórico assente em investigações/trabalhos anteriores, aponta para novas perspetivas sobre o tema "Modelação e Previsão de Séries Económico-Financeiras", procurando confrontar metodologias clássicas com técnicas baseadas em *Deep Learning*. A necessidade de uma abordagem proveniente de *Data Science* está justificada não só pela importância e necessidade de extrair informação real do vasto banco de dados existente, como pelo facto desta área contemplar o conhecimento de diversas disciplinas requeridas à consecução desta investigação: Matemática, Estatística e Computação, articuladas aqui com as áreas da Economia, Gestão e Finanças.

Feito este enquadramento, na perspetiva de articular a nossa área científica com o tema em estudo, as linhas subsequentes da introdução serão dedicadas à contextualização e apresentação dos objetivos e das questões de investigação que estão na base do trabalho que nos propomos desenvolver.

(A) CONTEXTUALIZAÇÃO DO TRABALHO E MOTIVAÇÃO

Enquadrado nos domínios de *Machine Learning* e *Data Science*, passando pela implementação de métodos econométricos no estudo de séries temporais, o tema central da investigação incide na modelação e (sobretudo) previsão de séries temporais económico-

financeiras, confrontando metodologias clássicas (com ênfase nas famílias de modelos autorregressivos e de alisamento exponencial) com metodologias de aprendizagem profunda baseadas em redes neuronais artificiais.

Apesar da facilidade de implementação e rapidez na execução computacional das metodologias clássicas, na literatura científica têm-lhes sido apontadas algumas limitações, nomeadamente a dificuldade em lidar com alterações fora dos padrões e em captar as informações realmente relevantes sobre os dados, com vista a efetuar previsões. Com efeito, na era da informação, e beneficiando do forte desenvolvimento computacional, o desafio a que nos propomos passa por explorar as vantagens da implementação computacional de metodologias de *Machine Learning*, com capacidade para "aprender" e extrair informação consistente dos dados brutos.

Deste modo, é nossa motivação investir no levantamento de um referencial teórico consistente e traçar um alinhamento dos principais aspetos a considerar na modelação e previsão de séries temporais, bem como compilar e desenhar um conjunto de rotinas computacionais organizadas, automatizadas e robustas que permitam um estudo "consistente". Aliás, decidimos investir nesta parte por percebermos que: (i) a execução computacional envolve um dispêndio de tempo e um conhecimento computacional consideráveis e (ii) existem algumas lacunas na existência de *scripts*, já devidamente estruturados, que permitam um estudo computacional relativamente automatizado, de fácil implementação e aplicáveis a séries temporais de natureza diversa. Estamos certos de que será uma mais-valia para a comunidade científica.

Ainda, alinhados com uma tendência de investigação orientada para a aplicabilidade em contexto real, será dada ênfase à vertente empírica, no sentido de aplicar/testar as rotinas desenvolvidas a dados reais, para avaliar, até que ponto, as metodologias de *Machine Learning* são, ou não, uma boa 'ferramenta' de previsão. Para que as conclusões possam ser mais contributivas para desenvolvimentos futuros, será nossa preocupação testar essa hipótese em séries com características diferenciadas. Para o efeito serão utilizadas quatro séries temporais: (1) *Consumer Price Index for All Urban Consumers: All Items in U.S. City Average* (CPIAUCSL); (2) *Vehicle-Miles Travelled* (VMT); (3) *Portuguese Stock Index 20* (PSI 20) e (4) *Standard & Poor's 500 Exchange-Traded Fund* (SPY). A escolha destas séries prendeu-se, sobretudo, com o facto de apresentarem comportamentos e características distintas, em particular no que respeita a tendência, sazonalidade, quebras de estrutura, entre outras.

Com efeito, é igualmente uma motivação encontrar padrões que nos permitam identificar a existência de 'melhores' modelos/metodologias para a modelação e previsão (combinando a

qualidade preditiva com o custo computacional) de séries temporais, tendo por base as suas características.

Acreditamos, desta forma, que o desenvolvimento deste projeto de investigação contribuirá de forma positiva para o debate científico e abrirá perspetivas a desenvolvimentos futuros sobre o tema, na medida em que os padrões e as ligações a encontrar poderão constituir ensaios, com utilização prática, no estudo de séries temporais.

(B) OBJETIVOS E QUESTÕES DE INVESTIGAÇÃO

Feita a contextualização e o enquadramento da nossa investigação, bem como uma retrospetiva genérica de aspetos que estão na base da nossa motivação, importa clarificar os objetivos específicos e as questões nela implícitas. Assim, dentro do tema central definimos os objetivos específicos de investigação a seguir identificados.

1. Sistematizar um referencial teórico essencial ao estudo de séries temporais e de tópicos intrínsecos ao *Machine Learning*.

2. Analisar e caracterizar as séries temporais: representações gráficas, decomposição, quebras de estrutura, medidas descritivas e testes estatísticos (por exemplo, normalidade, estacionariedade, independência).

3. Determinar, de entre as metodologias clássicas, modelos de previsão que se ajustem aos dados em análise e implementar as respetivas rotinas computacionais (robustas e automatizadas).

4. Implementar/testar metodologias de *Machine Learning* baseadas em arquiteturas de *Deep Neural Networks* (DNN) no processo de modelação e previsão em séries temporais, envolvendo a exploração e construção de rotinas computacionais (completas e automatizadas).

5. Testar potencialidades dos modelos de previsão desenvolvidos (baseados em arquiteturas de *Deep Neural Networks*), numa perspetiva de minimização do erro de previsão.

6. Avaliar possíveis alterações a introduzir nas rotinas computacionais e/ou nas arquiteturas de *Deep Neural Networks* (eventualmente numa abordagem híbrida, com a combinação de diferentes metodologias) com o intuito de minimizar, simultaneamente, o erro de previsão e o custo computacional.

7. Pesquisar padrões das séries que permitam identificar os 'melhores' modelos de previsão (aliando melhor qualidade preditiva com menor custo computacional) consoante as características/natureza dos dados.

Enumerados os objetivos que balizam a nossa investigação, identificamos em seguida algumas questões para as quais procuraremos obter respostas.

(A) Face a uma análise cuidada das séries temporais em estudo, que particularidades estão presentes nas mesmas e em que medida poderão condicionar e ser um obstáculo ao processo de modelação e previsão?

(B) De entre as metodologias clássicas, que modelos se ajustam melhor aos dados? Serão esses modelos um suporte para cenários de previsão?

(C) Serão as metodologias de *Machine Learning* (em particular baseadas em arquiteturas de DNN) uma alternativa viável no processo de modelação e previsão, tendo em conta o erro de previsão e o custo computacional?

(D) Que implementações/alterações poderão ser feitas, numa perspetiva de inovação, com vista quer à obtenção de modelos ajustados e com boa qualidade preditiva, quer ao desenvolvimento de métodos computacionalmente mais eficientes?

(E) Existirão padrões que permitam selecionar as melhores metodologias/modelos de previsão, de acordo com as características/natureza dos dados?

Dada a preocupação no articular dos objetivos com as questões de investigação traçadas, estamos em crer que, com a obtenção de respostas concisas para cada uma das questões alcançaremos os objetivos definidos. Deste modo, é nossa ambição que o desenvolvimento deste projeto seja estruturado, coeso e alinhado com a trajetória de investigação, para que possa

não só contribuir para sistematizar a informação teórica implícita nos tópicos em estudo, como abrir novas perspetivas sobre a "Modelação e Previsão de Séries Económico-Financeiras", devido sobretudo ao confronto de metodologias e à introdução de modificações a modelos já existentes.

(C) ESTRUTURA DO TRABALHO

No que respeita à estrutura, após esta Introdução, o trabalho está organizado em 5 capítulos, complementados com uma Conclusão e vários Anexos.

O Capítulo 1 é dedicado a um enquadramento que permite balizar a nossa investigação nos domínios da área científica da Gestão. Assim, numa fase inicial, é importante enquadrar os desafios atuais, no que respeita à modelação e previsão de séries temporais, tendo em conta um referencial teórico *Applied to Bussiness*.

Os Capítulos 2 e 3 respeitam ao enquadramento teórico-matemático. O Capítulo 2 é dedicado aos aspetos principais inerentes ao estudo de séries temporais, sustentado na abordagem clássica sobre o tema. Após uma breve introdução aos aspetos genéricos afetos à modelação em Econometria (Sec. 2.1), é feita uma revisão sucinta de conceitos fundamentais no estudo de séries temporais (Sec. 2.2). Dentro dos modelos de regressão linear (clássicos), são abordadas as famílias de modelos autorregressivos (e de médias móveis) e de alisamento exponencial (Sec. 2.4), fazendo-se referência aos critérios de seleção (Sec. 2.5) e validação (Sec. 2.6) dos modelos.

Traçada uma perspetiva mais 'clássica' sobre a análise de séries temporais, no Capítulo 3 é apresentada uma revisão dos aspetos teóricos e metodológicos subjacentes ao estudo de redes neuronais artificiais. O capítulo inicia com uma introdução onde são introduzidos conceitos chave, como Inteligência Artificial, *Machine Learning*, *Deep Learning*, *Data Science*, entre outros (Sec. 3.1). Em seguida, são tecidas considerações teóricas gerais sobre redes neuronais: partindo de um enquadramento histórico (Sec. 3.2) e da descrição formal dos conceitos básicos envolvidos no desenho de redes neuronais artificiais (Sec. 3.3), revêm-se os seus processos de aprendizagem e formaliza-se um algoritmos de treino (Sec. 3.4) aplicável a algumas arquiteturas atuais de redes que servirão de base a parte do estudo empírico desenvolvido (Sec. 3.5). Finalmente, explica-se como são medidos o desempenho e capacidade da rede por validação do treino e avaliação do modelo produzido (Sec. 3.6).

Os dois capítulos seguintes (4 e 5) apresentam a parte prática do nosso estudo. No Capítulo 4 expõem-se os procedimentos metodológicos adotados na análise de dados e na

implementação computacional, recorrendo à linguagem de programação *Python*. Aqui, além da apresentação das séries temporais que vão servir da base ao estudo empírico (Sec. 4.1), são explicadas as metodologias seguidas na investigação fazendo a ponte para a parte da implementação computacional (Sec.4.3). Ainda, procurando complementar as considerações teóricas dos capítulos anteriores, são fundamentados e descritos procedimentos relativos ao pré-processamento e transformação dos dados (Sec. 4.2) e ao estabelecimento de métricas para avaliar o erro de previsão (Sec. 4.4).

O Capítulo 5 é dedicado na íntegra à apresentação de resultados. Numa parte inicial é feita uma análise exploratória das séries temporais em estudo (Sec. 5.1), seguindo-se a apresentação de resultados relativos à modelação e previsão de cada uma das séries pelos diferentes métodos (Sec. 5.2). O capítulo encerra com uma secção dedicada à comparação e discussão final de resultados obtidos por cada um dos métodos (Sec. 5.3). Todos os *outputs* apresentados ao longo desta secção, embora sujeitos a algum tratamento gráfico, foram obtidos da implementação computacional feita em *Python*.

Na última parte – Conclusão – são apresentadas e sintetizadas as principais conclusões deste estudo, procurando-se estabelecer uma relação crítica entre os resultados observados e as questões de investigação estabelecidas. São ainda feitas conjeturas relativamente a possíveis limitações, bem como aspetos a desenvolver em estudos ulteriores, partindo do trabalho aqui apresentado e discutido.

Finalmente, apresentam-se informações suplementares ou complementares ao trabalho desenvolvido em seis Anexos. No Anexo A damos conta da produção científica resultante deste estudo, seja a apresentada em conferências, seja a publicada *online*. Os Anexos B, C e D vêm complementar a informação relativa, respetivamente, aos Capítulos 2, 3 e 4. Já os Anexos E e F contêm informação relativa à parte empírica do nosso estudo, complementando o Anexo E a análise exploratória das séries temporais em estudo e o Anexo F a sua modelação e previsão.

De referir, finalmente, que a numeração das equações, figuras e tabelas é feita de acordo com o Capítulo/Anexo a que reportam. A título de exemplo, uma equação, figura ou tabela que surja no Capítulo 2 é iniciada com a marca 2.__, enquanto uma equação, figura ou tabela que surja no Anexo A é iniciada com a numeração A.__.

1. MODELAÇÃO E PREVISÃO *APPLIED TO BUSINESS*

Numa sociedade cada vez mais informatizada e sobrecarregada com um grande volume de dados, tornou-se fundamental extrair e analisar a informação que lhe está subjacente. Em particular, agentes económicos, grupos de investidores e governos têm manifestado uma necessidade crescente de análise de dados económico-financeiros fidedignos cujos resultados alimentem sistemas de apoio à decisão cada vez mais sofisticados. É neste contexto que a previsão, em particular na área afeta aos mercados financeiros (ganhos e perdas, valores futuros, risco), se assume como uma ferramenta essencial que tem impulsionado a investigação científica sobre este tema, a avaliar pelo elevado número de publicações das últimas décadas.

Hang (2019) refere-se à importância da previsão como ferramenta, destacando-a como indispensável na criação de vantagem competitiva em qualquer empresa suportando, assim, um planeamento proativo e a tomada de decisões de produção, negócios, financiamentos, investimentos, entre outros. Assim, a capacidade de previsão de lucros de uma empresa, por exemplo, permite-lhe traçar estratégias eficazes de curto médio e longo prazo para desenvolver soluções apropriadas, capazes de aumentar a sua competitividade e manter o seu desenvolvimento sustentável num mercado cada vez mais global.

Contudo, para que as técnicas de previsão sejam uma ferramenta efetiva, impõem-se que sejam realmente compreendidas. Na reflexão feita em Wilson e Spralls (2018) é analisada a perceção dos profissionais da área do *business* sobre a utilidade das técnicas de previsão. Os autores salientam a relevância de incluir a previsão como tema a abordar nos currículos académicos e de o fazer o mais próximo possível de cenários reais, referindo que atualmente ainda são as metodologias clássicas (modelos de médias móveis, tendências de regressão linear e alisamento exponencial) as mais reconhecidas e utilizadas pelos profissionais.

Neste sentido, não só no campo da investigação, mas também no campo da ação o trabalho afeto à previsão é destacado pela sua pertinência e importância, onde a teoria econométrica e a prática devem estar alinhadas. Por exemplo, Ramsey & Kmenta (1980) discutiam, já há algumas décadas, a existência de lacunas entre a teoria econométrica e a prática, sublinhando que, frequentemente, a teoria ignora problemas processuais difíceis enfrentados por economistas e profissionais afetos à área do *business*.

Apesar do desfasamento entre a teoria e a sua aplicação prática, a modelação e previsão de séries temporais tem-se mantido um tema de interesse para os investigadores, decorrente da potencial aplicação em contexto real. Aqui, não só se constata uma exploração das metodologias clássicas associadas, no sentido de identificar problemas e possíveis limitações, como, em paralelo, se destaca um claro investimento no desenvolvimento de metodologias alternativas. São exemplos as ferramentas nos domínios da "Inteligência Artificial", que, segundo Cavalcante *et al.* (2016), muito têm contribuído para avanços no que respeita à análise preditiva, nomeadamente em séries temporais económico-financeiras.

Porém, e sobretudo em séries financeiras (como as associadas a índices relativos aos mercados de valores), a constante instabilidade económica vivida à escala mundial, o surgir de crises com impactos globais, bem como outros fenómenos desencadeadores de incerteza e especulação, provocam comportamentos nos dados históricos que constituem um obstáculo aos processos de modelação e previsão (Chatfield, 2016). Algumas destas questões já são discutidas na literatura há décadas, salientando-se a dificuldade dos modelos de previsão clássicos, como os autorregressivos, propostos por Box e Jenkins (1976), em lidarem com eventos que saiam dos padrões. São ainda de destacar os impactos significativos que a não-estacionariedade (Libanio, 2005), a presença de quebras de estrutura/mudanças de regime, não linearidades ou assimetrias têm no estudo destas séries.

Especificando, períodos de forte especulação ou crises económicas/financeiras originam frequentemente grandes oscilações que saem dos padrões, podendo causar mudanças no comportamento da série, designadas por quebras de estrutura (Valentinyi-endr, 2004), o que, matematicamente, se traduz em perturbações nos parâmetros (dos modelos) conduzindo a um aumento do erro de previsão. Sobre isso, Pesaran e Timmerman (2004) sublinham que esta instabilidade estrutural pode ter um impacto significativo na *performance* preditiva e que muitos dos modelos econométricos de previsão têm um fraco desempenho em torno dos pontos de quebra.

Os obstáculos referidos motivaram alterações na linha de investigação nas últimas décadas. Se anteriormente as metodologias seguidas eram dominadas por modelos lineares, nomeadamente os autorregressivos, de médias móveis e de alisamento exponencial (muitas vezes designados por modelos clássicos), nas últimas décadas tem-se assistido a uma mudança nos paradigmas de abordagem, surgindo vários trabalhos que exploram modelos não lineares (Kirchgässner & Wolters, 2007). Estes permitem um melhor ajustamento a séries com fortes oscilações e com comportamentos que saem de um padrão (Clements, Franses, & Swanson,

2004) e a dados relativos a mercados financeiros, como é referido nos trabalhos desenvolvidos por McMillan (2005), Avramov e Chordia (2006) e McMillan (2007).

Além do supramencionado, na era da informação surgiram outros obstáculos relacionados com *Big Data* (devido à quantidade massiva de dados pouco estruturados), nomeadamente pelo facto de a capacidade preditiva continuar a ser limitada pela aplicação de técnicas estatísticas clássicas (que têm dificuldade em lidar com a dimensão, dinâmica e complexidade inerente a estes dados), pela falta de eficiência dos algoritmos de *data mining* existentes, pelas limitações de *software* e *hardware* disponíveis e pela falta de profissionais habilitados para trabalhar em *Big Data* (Hassani & Silva, 2015).

Com efeito, resultante dos desafios lançados por este campo da ação, muito em particular aqueles com que os profissionais afetos às áreas económico-financeiras se deparam, a investigação científica tem assumido um papel preponderante na procura de soluções. Isso, aliado ao grande progresso computacional observado nos últimos anos, tem levado a um interesse crescente da comunidade científica pela utilização de técnicas de *Data Science* e *Machine Learning*, nomeadamente as baseadas em redes neuronais com aprendizagem profunda (Zhang, Guo, & Wang, 2017), para o desenvolvimento e aplicação de metodologias alternativas de previsão.

A exploração de metodologias baseadas em redes neuronais (não lineares), muito debatida na década de 90, como nos dão conta Patuwo, Zhang e Hu (1998) e abandonada por limitações computacionais, ressurge em vários trabalhos atuais, fruto de desenvolvimentos tecnológicos, como alternativa promissora na modelação e previsão de séries temporais, em particular de dados económico-financeiros. Zhang *et al.* (2017) identificam dois fatores determinantes para o sucesso destas metodologias. Por um lado, as redes neuronais são capazes de extrair recursos abstratos de dados brutos, pois combinam várias fontes de informação, processam dados heterogéneos e conseguem captar alterações dinâmicas. Por outro lado, caso se trate um grande volume de dados, é possível desenhar arquiteturas de redes neuronais mais complexas (com mais parâmetros) capazes de aprender e treinar com amostras de dimensão considerável.

Efetivamente, os desenvolvimentos computacionais das últimas duas décadas foram tão significativos que potenciaram o surgimento de inúmeros trabalhos comprovando que as metodologias baseadas em redes neuronais artificiais permitem resolver com sucesso problemas de previsão de dados não lineares. Para mais detalhes, veja-se a revisão sistemática das aplicações de redes neuronais na área da Gestão entre 1994 e 2015, publicada por Tkáč & Verner (2016), onde os vários trabalhos são classificados de acordo com o ano de publicação,

área de aplicação, tipo de rede neuronal, algoritmo de aprendizagem e método de análise de referência, conforme esquematizado na Figura 1.1, adaptada do trabalho destes autores.

FIGURA 1.1 - Aplicações de redes neuronais na área da Gestão – Resumo Gráfico

Note-se que, apesar da diversidade de áreas, a maior parte da pesquisa centra-se em problemas que envolvem dificuldades financeiras, falências, previsão de preços de ações e apoio à tomada de decisão, cuja capacidade de resposta parece suportar-se, em parte, na atual evolução tecnológica.

Precisamente, beneficiando do forte desenvolvimento computacional (com a velocidade das GPUs a aumentar significativamente), várias arquiteturas de redes recorrentes, baseadas em mecanismos capazes de aprender dependências de longo prazo, têm sido propostas na literatura, evidenciando quer a sua ampla aplicabilidade, quer o seu potencial. Todavia, apesar das várias propostas, em que algumas parecem funcionar melhor em tarefas pontuais de acordo com certas especificidades dos dados (Jozefowicz, Zaremba, & Sutskever, 2015), a verdade é que as diferenças existentes são pontuais, sendo a base arquitetural a mesma (Greff *et al.*, 2015).

Desta forma, mais do que perceber as diferenças, importa perceber o efetivo funcionamento destas arquiteturas de redes neuronais, baseadas em mecanismos de aprendizagem profunda, para, assim, se poder propostas adaptações que melhor consigam responder aos desafios e a especificidades existentes nos dados em estudo.

Em suma, estamos perante um tema dinâmico, onde os desafios passados não são os atuais nem serão, seguramente, os futuros. Assim, considera-se existir um claro espaço a novos desenvolvimentos, o justifica e enquadra a nossa investigação.

2. ESTUDO DE SÉRIES TEMPORAIS: ABORDAGEM CLÁSSICA[1]

Antecedendo a implementação dos procedimentos teórico-práticos a que nos propomos neste trabalho, é necessária uma reflexão sobre os aspetos principais inerentes ao estudo de séries temporais.

Ao longo deste capítulo, reconhecendo a brevidade com que alguns tópicos são discutidos, sugerem-se várias referências bibliográficas que não só serviram de base à abordagem como a complementam. Numa perspetiva genérica, enumeramos, desde já, algumas dessas referências bibliográficas: (1) Johston e DiNardo (2001) ou Heij *et al*. (2004) apresentam uma visão detalhada sobre os métodos e modelos econométricos em geral, evidenciando uma boa integração dos conceitos da teoria económica com as técnicas econométricas e os testes utilizados na estimação e validação dos diversos modelos; (2) em Kirchgässner e Wolters (2007) é apresentada uma análise mais focada no estudo de séries temporais, procurando estabelecer ligações entre os métodos e as aplicações, com a constante apresentação de exemplos empíricos; (3) finalmente, Hyndman e Athanasopoulos (2018) apresentam uma perspetiva mais centrada no *forecasting*, fornecendo uma introdução abrangente sobre vários métodos, uma apresentação objetiva de cada um e algumas indicações/sugestões de implementação computacional (em **R**).

2.1. INTRODUÇÃO: MODELAÇÃO EM ECONOMETRIA

Pelo facto de muitas variáveis económicas serem passíveis de quantificação, é possível, através de modelos matemáticos, caracterizar comportamentos, desenvolver estudos empíricos e fazer previsões. A base da análise econométrica reside no confronto desses modelos, os quais descrevem relações entre variáveis económicas, com a realidade.

Tal como referido por Lancaster (2004), poder-se-á dizer que a análise econométrica procura avaliar a consistência dos modelos com as evidências e, posteriormente (assumindo a existência de um modelo consistente), analisar as estruturas definidas pelo modelo, com vista a fazer inferências para a tomada de decisões económicas, por exemplo.

[1] Capítulo parcialmente adaptado de Ramos (2011).

É com base nestas premissas que será desenvolvido este trabalho, com especial ênfase na exploração de modelos econométricos que visam analisar relações entre variáveis económicas, numa perspetiva de efetuar previsões.

De uma forma geral, qualquer modelo contém 'quantidades' que são observáveis e outras que não são diretamente observáveis. Com efeito, a regressão, enquanto técnica estatística de modelação, assume a forma de um modelo probabilístico em que a variável dependente (endógena) é a soma de dois termos, um determinístico e outro estocástico (componente aleatória). Assim, a forma geral dos modelos (de regressão) econométricos lineares a implementar, para uma observação t, pode ser expressa pela equação (2.1)

$$y_t = \beta_0 + \beta_1 x_{1t} + \beta_2 x_{2t} + \cdots + \beta_k x_{kt} + u_t \qquad (2.1)$$

onde y_t representa a variável endógena e o membro da direita é soma da expressão linear (em que os x_{it}, com $i = 1, \dots k$, representam as k variáveis exógenas e os β_j, com $j = 0, \dots k$, os coeficientes de regressão a estimar) com a componente aleatória, u_t (convencionalmente designada por erro do modelo). Esta última componente, u_t, incorpora a parte resultante do facto de estarmos a considerar um processo estocástico e os efeitos de outras variáveis não representadas no modelo.

No entanto, um modelo só é válido se a componente aleatória, u_t, corresponder a um processo estocástico puro, ruído branco (*White Noise* – WN), ou seja, se toda a informação relevante estiver contida na parte determinística. Nesses casos denotaremos por ε_t a componente residual, onde os erros são independentes e identicamente distribuídos (iid) com $\varepsilon_t \sim WN(0, \sigma_\varepsilon^2)$, onde: (i) $\mathbb{E}(\varepsilon_t) = 0$; (ii) $\mathbb{E}({\varepsilon_t}^2) = \sigma_\varepsilon^2$ e (iii) $\mathbb{E}(\varepsilon_i \varepsilon_j) = 0$ com $i \neq j$.[2] Assim, na apresentação dos modelos teóricos descritos ao longo deste capítulo, quando se supuser estar perante um ruído branco, usar-se-á a notação ε_t para referir a componente residual, caso contrário, manter-se-á a notação u_t.

Porém, sendo o nosso foco de interesse a modelação/previsão em séries temporais[3], será seguida uma metodologia univariada ou extrapolativa, pelo que se despreza a possibilidade de

[2] No caso de se ter $\varepsilon_t \sim N(0, \sigma_\varepsilon^2)$ estamos perante um ruído branco gaussiano. Verificar-se $\varepsilon_t \sim WN(0, \sigma_\varepsilon^2)$, mas não se verificar $\varepsilon_t \sim N(0, \sigma_\varepsilon^2)$ poderá ser um possível problema para a componente residual. No entanto, isto não condiciona a propriedade "melhor estimador linear não enviesado" dos estimadores OLS (*Ordinary Least Squares*), mas, neste caso, os procedimentos de inferência têm apenas validades assintóticas (Johnston & DiNardo, 2001).

[3] O registo de fenómenos a variar com o tempo tem o nome de série temporal (Kitagawa, 2010), conceito que se caracteriza pelo conjunto de observações feitas em períodos consecutivos e durante um determinado intervalo de tempo (discreto ou contínuo).

existirem relações entre a variável de interesse, y_t, e outras variáveis, sendo o seu comportamento explicado, apenas, com base no seu histórico (valores passados) e nos valores correntes e passados de um erro estocástico.[4]

Com efeito, o modelo linear pode ser especificado de forma autorregressiva. Mais concretamente, considerado a série temporal y no instante t (num intervalo de tempo T), $\{y_t\}_{t\in T}$, esta pode ser explicitada como função (linear) dos valores passados e alguma perturbação, conforme descrito na equação (2.2)

$$y_t = f(y_{t-1}, y_{t-2}, \dots) + u_t \tag{2.2}$$

onde u_t não é assumido, necessariamente, como um processo de ruído branco, podendo, contudo, ser especificado como função (linear) de um processo ε com essa característica. Deste modo, considerando $u_t = g(\varepsilon_t, \varepsilon_{t-1}, \varepsilon_{t-2}, \dots)$, a equação (2.2) pode ser reescrita

$$y_t = f(y_{t-1}, y_{t-2}, \dots) + g(\varepsilon_t, \varepsilon_{t-1}, \varepsilon_{t-2}, \dots) \tag{2.3}$$

Em particular, em termos autorregressivos, uma especificação da equação anterior pode ser escrita na forma

$$y_t = c + (a_1 y_{t-1} + a_2 y_{t-2}, + \cdots) + (\varepsilon_t + b_1 \varepsilon_{t-1} + b_2 \varepsilon_{t-2} + \dots) \tag{2.4}$$

Com efeito, dentro dos modelos de regressão linear (clássicos) e numa perspetiva econométrica, além da família de modelos autorregressivos (e médias móveis), a nossa opção passou ainda pela família de modelos de alisamento exponencial, abordados nas páginas subsequentes.

2.2. FUNDAMENTAÇÃO TEÓRICA E NOÇÕES GERAIS

Precedendo a apresentação e análise dos modelos econométricos referidos (Sec. 2.4), esta secção introduz conceitos implícitos no estudo de séries temporais. Importa reforçar que, embora muitas noções possam ser generalizadas numa perspetiva multivariada, daqui em diante os temas serão apenas tratados numa perspetiva de análise univariada.[5]

[4] As séries temporais podem ser constituídas por registo de um único fenómeno ao longo do tempo (séries temporais univariadas) ou obtidas pelo registo simultâneo de dois ou mais fenómenos (séries temporais multivariadas).

[5] Para mais detalhes sobre a abordagem multivariada, veja-se Johnston e DiNardo (2001).

2.2.1. COMPONENTES DE UMA SÉRIE TEMPORAL

Para uma análise mais cuidada de séries temporais, é útil, numa fase inicial, perceber a dinâmica das mesmas em função do comportamento das suas componentes: tendência, ciclicidade, sazonalidade e aleatoriedade, onde:

- a **tendência** reflete uma evolução, a longo prazo, no sentido da monotonia da série (crescente/decrescente), podendo apresentar uma dinâmica linear ou não linear. Uma análise gráfica do comportamento da série, permite identificar claramente esta característica;
- a **ciclicidade** é traduzida por um padrão de flutuações de médio prazo, podendo ou não ser periódicas, as quais podem afetar diretamente a tendência global da série. No caso de séries temporais económico-financeiras, fatores como a expansão ou recessão da economia, poderão motivar este tipo de comportamento, os quais são identificados na literatura como 'ciclos económicos';
- a **sazonalidade** reflete-se em variações cíclicas regulares que ocorrem em períodos constantes e menores comparativamente à ciclicidade.[6] Estas variações caracterizam-se, em geral, por provocarem oscilações (subidas e descidas) nos valores da série, podendo ser aditivas (amplitudes regulares) ou multiplicativas (amplitudes crescentes/decrescente). As flutuações sazonais estão, essencialmente, relacionadas com fatores naturais (como a época do ano), mas podem também dever-se ao comportamento humano;
- a **aleatoriedade** está associada a flutuações imprevisíveis, não modeladas pelas componentes anteriores; caracteriza-se, assim, por um comportamento aleatório, sem correlações temporais (idealmente).

Assim, é comum fazer-se a decomposição da série temporal nestas componentes e estudá-las de forma isolada. Mais, sendo o nosso propósito a previsão, tal como descrito por Makridakis *et al.* (1998), analisar e extrair dinâmicas (tendência, ciclicidade e sazonalidade) é fundamental para melhor compreender o comportamento da série e aumentar a precisão das previsões.

[6] Importa distinguir que, se as flutuações apresentam uma frequência 'fixa', estas estão associadas a um comportamento sazonal; por outro lado, se as flutuações não apresentam frequência fixa, estas são consideradas cíclicas.

Em termos práticos e formais, importa referir que quando decompomos uma série temporal, é usual combinar a tendência e a ciclicidade numa única componente (ciclo de tendência), por vezes chamada apenas de tendência por simplicidade (Hyndman & Athanasopoulos, 2018). Em termos matemáticos, a decomposição da série temporal $\{y_t\}_{t\in T}$ pode ser escrita de duas perspetivas: decomposição aditiva (2.5) e decomposição multiplicativa (2.6)[7]

$$y_t = CT_t + S_t + R_t \tag{2.5}$$

$$y_t = CT_t \times S_t \times R_t \tag{2.6}$$

onde CT_t denota a componente do ciclo de tendência, S_t a componente sazonal e R_t a componente residual (aleatória).

Na prática, a decomposição da série temporal deve ser considerada de início (aquando da caracterização na mesma), pois, não só ajuda a melhorar o entendimento da série, como pode ser útil, ou mesmo necessária, no processo de modelação/previsão. A extração de componentes é feita por pré-processamento dos dados, mediante a aplicação de métodos adequados descritos na Sec. 4.2.

Mais, a decomposição de séries temporais pode, ainda, ser usada para medir a 'força da tendência' e a 'força da sazonalidade' presentes na série temporal (Wang, Smith, & Hyndman, 2006). Com efeito, tendo em conta (2.5), em dados que evidenciam uma forte tendência, a variância da soma das componentes CT_t com R_t, $Var(CT_t + R_t)$, apresenta um valor considerável e, consequentemente, o valor do quociente entre as duas variâncias $Var(R_t)$ e $Var(CT_t + R_t)$, $Var(R_t)/\,Var(CT_t + R_t)$, deverá ser relativamente pequeno. Definimos, assim, a 'força de tendência' pela equação (2.7)

$$\mathcal{F}_T = max\left(0,\ \ 1 - \frac{Var(R_t)}{Var\ (CT_t + R_t)}\right) \tag{2.7}$$

onde uma série com valor de $\mathcal{F}_T$ próximo de 1 evidencia uma forte tendência, enquanto que numa série sem tendência, o valor será 0.[8]

[7] A decomposição aditiva é, geralmente, mais apropriada se a magnitude das flutuações sazonais, ou a variação em torno do ciclo de tendências, não se alterarem ao longo da série temporal. Quando a variação no padrão sazonal, ou em torno do ciclo de tendências, evidenciar uma proporcionalidade, a decomposição multiplicativa é mais apropriada. Por exemplo, em séries temporais económico-financeiras é comum considerar a decomposição multiplicativa (Hyndman & Athanasopoulos, 2018).

[8] Como $Var(R_t)$ pode, ocasionalmente, ser maior do que $Var(CT_t + R_t)$, definimos o valor mínimo de $\mathcal{F}_T$ como zero, pelo que a medida fornecerá um valor entre 0 e 1.

De forma análoga podemos definir 'força de sazonalidade' pela equação (2.8)

$$\mathcal{F}_S = max\left(0,\ \ 1 - \frac{Var(R_t)}{Var\ (S_t + R_t)}\right) \tag{2.8}$$

onde uma série com valor de $\mathcal{F}_S$ próximo de 0 exibe pouca ou nenhuma sazonalidade, enquanto numa série com forte sazonalidade, o valor será próximo de 1.

2.2.2. ESTACIONARIEDADE

Contrapondo com estudos prévios, em que os modelos econométricos usados assumiam, à partida, que a origem dos dados provinham de processos onde a média e a variância eram constantes ao longo do tempo, não dependendo do período a que se reportavam, Granger e Newbold (1974) alertaram para o facto de, ao se estimar uma regressão de séries temporais, ainda que os valores dos coeficientes de regressão fossem significativos e que o coeficiente de determinação, R^2, fosse elevado, correr-se-ia o risco de se obter o que ficou denominado de regressão espúria. Nesses casos, apesar de o modelo ter significância estatística, não existe uma relação causa-efeito entre as variáveis, pelo que os resultados não têm significado económico. Por exemplo, considerando uma série temporal com forte tendência, existe uma regressão linear entre os valores desfasados no tempo, observando-se um valor de R^2 alto, devido à presença de tendência – autocorrelação.

Daí, conhecidos os riscos das regressões espúrias, em estudos ulteriores passou a usar-se a abordagem definida por Box e Jenkins (1976)[9] (ver Figura B.1 em anexo), procurando-se 'estabilizar' as séries (médias e variâncias constantes ao longo do tempo) antes da sua inclusão nos modelos, cuja análise se torna mais simples, dada a estabilidade dos parâmetros estimados.

No seguimento do exposto, podemos afirmar que, nos modelos com séries temporais, admite-se que as variáveis assumem valores gerados por um processo estocástico. Sendo o objetivo desenvolver e utilizar modelos capazes de interpretar e prever dados económicos e financeiros, para que as inferências estatísticas sejam válidas (pois disto dependem as propriedades dos estimadores), um dos aspetos a ser validado é a estacionariedade das séries

[9] Durante a década de 60 os professores George E. P. Box e Gwilym M. Jenkins desenvolveram diversos trabalhos sobre séries temporais. Em 1970 publicaram o livro *Time Series Analysis, Forecasting and Control* (apesar da mais citada na literatura ser a versão lançada em 1976). Deixado como sugestão bibliográfica para uma análise mais detalhada sobre o tema, o grande contributo desse trabalho foi reunir as técnicas existentes numa metodologia para construir modelos que descrevessem com precisão (e de forma parcimoniosa) o processo gerador da série temporal, originando previsões consistentes de valores futuros.

temporais utilizadas. Isto é, verificar se as mesmas seguem um processo estocástico com média e variância constantes ao longo do tempo, bem como covariâncias entre valores desfasados da série igualmente constantes.

De um modo formal, define-se série estacionária como:

Definição: *Uma série* $\{y_t\}_{t\in T}$ *diz-se fracamente estacionária*[10] *(ou estacionária em covariância) se, para* $t, t-k, t-j, t-j-k \in T$ *arbitrários:*

(i) A média (valor esperado) em qualquer momento não depende desse exato momento, sendo sempre constante, isto é

$$\mathbb{E}(y_t) = \mathbb{E}(y_{t-k}) = \mu \tag{2.9}$$

(ii) A variância dos seus valores, seja qual o período a que se reporta, for constante e finita, isto é

$$\mathbb{E}((y_t - \mu)^2) = \mathbb{E}((y_{t-k} - \mu)^2) = \sigma_y^2 < \infty \tag{2.10}$$

(iii) A covariância entre as diferentes observações não depende do período em que são tomadas, sendo constante, isto é

$$\mathbb{E}\big((y_t - \mu)(y_{t-k} - \mu)\big) = \mathbb{E}\Big(\big(y_{t-j} - \mu\big)\big(y_{t-j-k} - \mu\big)\Big) = \gamma_k \tag{2.11}$$

Assim, o ruído branco é um exemplo de uma série estacionária; já uma série que evidencie tendência ou um comportamento sazonal é não estacionária, uma vez que estas duas características afetam o valor observado, em momentos de tempo diferentes (ver Figura B.2 em anexo).

De acordo com a literatura, a ausência de estacionariedade é uma característica muito frequente em séries temporais, muito em particular em séries económico-financeiras (por não apresentarem, em geral, média e/ou variância constantes). Dada a importância da verificação desta característica no processo de modelação, a questão que se impõe é "Como estabilizar uma série não estacionária?". Os métodos de estabilização comummente referidos na literatura (Box, Jenkins, & Reinsel, 1994) são o de estacionarização em tendência (*Trend Stationary Process* – TSP) e em diferenças (*Difference Stationary Process* – DSP). Por aplicação destes processos

[10] Quando provamos a estacionariedade das variáveis, referimo-nos, em regra geral, à estacionariedade fraca. Importa referir que, sob finitude de variância, a estacionariedade implica estacionariedade fraca, mas o recíproco não é necessariamente válido. Tal apenas se verifica no caso de a distribuição ser gaussiana.

obtêm-se séries estacionárias em tendência e séries estacionárias em diferenças, como descrito matematicamente em Johnston e DiNardo (2001, sec. 7.3.3).

Centrando particular atenção em dados de natureza económico-financeira, de acordo com a perspetiva tradicional dos ciclos económicos, a tendência determinística seria determinada por fatores como a acumulação de capital e o progresso tecnológico, entre outros, pelo que estes ciclos se traduziriam em perturbações transitórias de acordo com essas tendências. Deste modo, os efeitos dos choques económicos dissipar-se-iam relativamente com o tempo (abordagem onde é aplicável o TSP).

Contudo, existem outros fenómenos (como a persistência dos choques nos ciclos económicos) para os quais é mais eficaz estabilizar as séries por recurso a processos integrados, isto é, estacionários por diferenciação (DSP), onde é possível a introdução duma tendência estocástica no modelo.[11]

Com efeito, em resposta à questão acima, em função da natureza da série, podemos identificar o procedimento mais adequado para a estabilizar. No caso do TSP é removida a tendência determinística, já no caso do DSP, diferencia-se a série.

De entre as possibilidades para aferir a estacionariedade da série temporal, podemos destacar dois métodos: (i) um julgamento subjetivo efetuado a partir de representações gráficas relativas à série temporal e/ou (ii) aplicação de testes estatísticos apropriados. Veremos algumas considerações sobre cada um destes métodos em secções seguintes.

2.2.3. OPERADOR *LAG* / OPERADOR DIFERENÇA

Para tornar operacional a relação autorregressiva descrita na equação (2.4) é necessário definir o desfasamento (*lag*) a incluir no modelo ao nível das componentes autorregressiva e residual e especificar a sua forma funcional.

Define-se o **operador *lag***[12], L, como sendo

$$Ly_t = y_{t-1}\,, \quad L^2 y_t = Ly_{t-1} = y_{t-2}\,, \quad \ldots, \tag{2.12}$$

ou, geralmente, para um certo s inteiro, temos

$$L^s y_t = y_{t-s} \tag{2.13}$$

[11] Para mais detalhes, ver Kirchgässner e Wolters (2007) ou Patterson (2011).

[12] Note-se que, algebricamente, o operador *lag* é uma aplicação linear, gozando das respetivas propriedades.

O **operador (primeira) diferença**, Δ, pode escrever-se como $1 - L$, o qual é descrito pela equação (2.14)

$$\Delta y_t = y_t - y_{t-1} = (1 - L)y_t \tag{2.14}$$

ou, ainda, considerando duas diferenças sucessivas,

$$\Delta^2 y_t = y_t - 2y_{t-1} + y_{t-2} = (1 - 2L + L^2)y_t \tag{2.15}$$

Por norma operamos sobre uma série não com o operador *lag*, mas com o polinómio operador *lag*. Assim, um polinómio simples em L será, por exemplo, $(1 - L)L$, ou seja, ΔL, vindo

$$(1 - L)Ly_t = \Delta y_{t-1} = y_{t-1} - y_{t-2} \tag{2.16}$$

ou, de outra forma, para $p = 1, \dots, t - 1$, temos[13]

$$\phi(L)y_t = y_t + \sum_{k=1}^{p} a_k\, y_{t-k} = \underbrace{y_t + a_1 y_{t-1} + a_2 y_{t-2} + \cdots + a_p y_{t-p}}_{(1+a_1L+a_2L^2+\ \dots\ +a_pL^p)\, y_t} \tag{2.17}$$

Uma operação importante com polinómios L é a da inversão. A título de exemplo, embora simples, vejamos um dos mais utilizados no estudo de séries temporais.

Considerando $\phi(L) = 1 - \varrho L$, repare-se que

$$(1 - \varrho L)(1 + \varrho L + \varrho^2 L^2 + \ \dots\ + \varrho^p L^p) = 1 - \varrho^{p+1} L^{p+1} \tag{2.18}$$

Assegurada a condição de que $|\varrho| < 1$, quando $p \to \infty$, vem que $\varrho^{p+1} L^{p+1} \to 0$. Pelo que o polinómio inverso de $\phi(L)$ é dado por um polinómio 'infinito' descrito em (2.19)

$$\phi^{-1}(L) = \frac{1}{1 - \varrho L} = 1 + \varrho L + \varrho^2 L^2 + \ \dots \tag{2.19}$$

A utilidade desta propriedade será retomada mais adiante no desenvolvimento da família de modelos autorregressivos.

[13] O mesmo raciocínio pode ser adotado para a componente ε, vindo (para $q = 1, \dots, t - 1$)

$$\theta(L)\varepsilon_t = \varepsilon_t + \sum_{k=1}^{q} b_k\, \varepsilon_{t-k} = \underbrace{\varepsilon_t + b_1\varepsilon_{t-1} + b_2\varepsilon_{t-2} + \cdots + b_q\varepsilon_{t-q}}_{(1+b_1L+b_2L^2+\ \dots\ +qL^q)\, \varepsilon_t}$$

2.2.4. FUNÇÕES DE AUTOCOVARIÂNCIA, AUTOCORRELAÇÃO E AUTOCORRELAÇÃO PARCIAL

O estudo e análise de processos estacionários pode fazer-se mediante uma análise gráfica, onde as funções de autocovariância (FACV), autocorrelação (FAC) e autocorrelação parcial (FACP) entre valores desfasados da série temporal assumem extrema importância.

Considerando estar perante uma série $\{y_t\}_{t\in T}$ já estacionária (podendo, ou não, ter de se recorrer às suas diferenças como forma de conseguir a estacionariedade),[14] pode definir-se:

- a **média** e a **variância** da série por

$$\mathbb{E}(y_t) = \mu \quad e \quad Var(y_t) = \sigma^2 \tag{2.20}$$

- a **FACV** da série por

$$\gamma_k = Cov(y_t,\, y_{t+k}) = \mathbb{E}[y_t\, y_{t+k}] - \mu_t\, \mu_{t+k}\,, \quad k \in \mathbb{Z} \tag{2.21}$$

- a **FAC** da série por

$$\rho_k = \frac{Cov(y_t,\, y_{t+k})}{\sqrt{Var(y_t)\; Var(y_{t+k})}} = \frac{\gamma_k}{\gamma_0}, \quad k \in \mathbb{Z} \tag{2.22}$$

  pelo facto de $Var(y_{t+k}) = Var(y_t) = \gamma_0$.

Observe-se que, salvo situações muito particulares, um aumento do desfasamento implica um decréscimo de γ_k e, consequentemente de ρ_k, pelo que, à medida que k aumenta, é de esperar que a capacidade de memória do processo seja limitada (conforme descrito em (2.10)). Isto é, para $|k| \to \infty$, vem $\gamma_k \to 0$ e $\rho_k \to 0$.

Já quanto à **FACP** de uma série $\{y_t\}_{t\in T}$ estacionária (considerando, sem perda de generalidade, com média nula) esta é expressa pelo coeficiente de correlação parcial, φ_{kk}, $k \in \mathbb{N}_0$, entre y_t e y_{t+k} (uma vez eliminada a dependência linear de y_t das variáveis y_{t+1}, $y_{t+2}, \dots, y_{t+k-1}$, e incorporada na componente residual, ε_t),[15] que traduz o coeficiente de ordem k (φ_{kk}) identificado na equação (2.4) do modelo de séries temporais

[14] Na abordagem feita nesta secção considerar-se-á, sem perda de generalidade, que os processos estocásticos são fracamente estacionários.

[15] Relativamente à série $\{y_t\}_{t\in T}$, quando consideradas as várias observações y_{t+1}, y_{t+2}, ..., de um modo sucinto, para $k \geq 2$ e condicionado ao conhecimento de y_{t+1}, $y_{t+2}, \dots, y_{t+k-1}$, verifica-se que y_t e y_{t+k} são independentes, e logo não correlacionadas. Este tipo de comportamento é algo que não é descrito pela FAC, mas sim captado por uma outra função – FACP.

$$y_{t+k} = \varphi_{k1} y_{t+k-1} + \varphi_{k2} y_{t+k-2} + \cdots + \varphi_{kk} y_t + \varepsilon_{t+k} \tag{2.23}$$

com $\varepsilon_t \sim N(0, \sigma_\varepsilon^2)$ e ε_{t+k} independente de $y_{t+k-j}, j \geq 1$.

Além da possibilidade de estimação, usando manipulação algébrica, as autocorrelações parciais podem ser calculadas recursivamente (com recurso ao *Algoritmo de Durbin-Levison*) por

$$\varphi_{00} = 1; \quad \varphi_{11} = \rho_1 \quad e \quad \varphi_{k+1,k+1} = \frac{\rho_{k+1} - \sum_{j=1}^{k} \varphi_{kj} \cdot \rho_{k+1-j}}{1 - \sum_{j=1}^{k} \varphi_{kj} \cdot \rho_j}, \quad k \geq 1 \tag{2.24}$$

com $\varphi_{k+1,j} = \varphi_{kj} - \varphi_{k+1,k+1} \cdot \varphi_{k,k+1-j}, \quad j = 1, 2, \ldots, k$

Sendo estas medidas/funções (em geral desconhecidas) importantes para caracterizar a série temporal, partindo dos valores amostrais, além dos estimadores comummente usados para a média, $\hat{\mu} = \frac{1}{n}\sum_{t=1}^{n} y_t$, e para a variância, $\hat{\sigma}^2 = \frac{1}{n}\sum_{t=1}^{n}(y_t - \bar{y})^2$, temos:[16]

- para **FACV**, um dos estimadores proposto é

$$\hat{\gamma}_k = \frac{1}{n}\sum_{t=1}^{n-k}[(y_t - \hat{\mu})(y_{t+k} - \hat{\mu})], \quad k \geq 0 \tag{2.25}$$

- para a **FAC**, um dos estimadores proposto é

$$\hat{\rho}_k = \frac{\hat{\gamma}_k}{\hat{\gamma}_0} = \frac{\sum_{t=1}^{n-k}[(y_t - \hat{\mu})(y_{t+k} - \hat{\mu})]}{\sum_{t=1}^{n}[(y_t - \hat{\mu})^2]}, \quad k \geq 0 \tag{2.26}$$

- já para a **FACP**, os estimadores das autocorrelações parciais podem ser obtidos usando o *Algoritmo de Durbin-Levinson*

$$\hat{\varphi}_{00} = 1; \quad \hat{\varphi}_{11} = \hat{\rho}_1 \quad \text{e} \quad \hat{\varphi}_{k+1,k+1} = \frac{\hat{\rho}_{k+1} - \sum_{j=1}^{k} \hat{\varphi}_{kj} \cdot \hat{\rho}_{k+1-j}}{1 - \sum_{j=1}^{k} \hat{\varphi}_{kj} \cdot \hat{\rho}_j}, \quad k \geq 1 \tag{2.27}$$

com $\hat{\varphi}_{k+1,j} = \hat{\varphi}_{kj} - \hat{\varphi}_{k+1,k+1} \cdot \hat{\varphi}_{k,k+1-j}, \quad j = 1, 2, \ldots, k.$

[16] Procurando garantir alguma precisão nos estimadores definidos, na prática apenas se estimam os valores da FACV, da FAC e da FACP para um número de ordens correspondente a uma certa fração do tamanho da realização obtida, n. Uma das regras usada, em termos práticos, define a estimação dos coeficientes dessas funções associados às ordens dos desfasamentos $k = 0, 1, \ldots, [n/4]$ (Box *et al.*, 1994).

Finalmente, se representarmos graficamente os coeficientes de autocorrelação em função dos *lags*, k, obtemos o correlograma da série temporal, construção que será bastante útil para inferir sobre a sua estacionariedade e a ordem do modelo autorregressivo.[17]

2.2.5. TESTES DE RAIZ UNITÁRIA/ESTACIONARIEDADE

Além da análise gráfica da série e do correlograma, a estacionariedade, ou não, de uma série temporal deve também ser avaliada por realização de testes estatísticos.

A existência de raiz unitária em séries temporais de natureza financeira é tema de ampla discussão e estudo na literatura, facto que já Libanio (2005) nos dava conta com a publicação do seu trabalho. Assim, propomo-nos de seguida identificar os tipos de não estacionariedade existentes e descrever como detetar a (não) estacionariedade de uma série temporal.

Podendo uma série temporal apresentar uma tendência determinística e/ou estocástica, a questão inicial é avaliar o(s) tipo(s) de tendência presente(s) na série. Primeiramente, a análise deve centrar-se na escolha de um dos dois processos estacionários: TSP e DSP (conforme descrito na Sec. 2.2.2). Assim, para apresentar os aspetos que se seguem e de acordo com o nosso foco de interesse (análise univariada de $\{y_t\}_{t\in T}$), considere-se o modelo geral descrito na equação (2.28)

$$y_t = \rho y_{t-1} + \beta_0 + \beta_1 t + u_t \tag{2.28}$$

onde, além da constante β_0 (*intercept* ou *drif* da série), do coeficiente de tendência determinística, β_1, o coeficiente ρ traduz a autocorrelação entre y_t e y_{t-1}.

Genericamente, temos um TSP caso $\rho < 1$ e um DSP quando $\rho = 1$. De um modo mais detalhado, podemos considerar quatro casos:

- Se $\rho = 0$, temos um processo estacionário em tendência puro (TSP puro), isto é, um processo estacionário em torno de uma tendência linear, pelo que existe tendência determinística;
- Se $\rho = 1$ e $\beta_1 = 0$, existe tendência estocástica, obtendo-se um processo estacionário em primeiras diferenças puro (DSP puro), o qual se designa por passeio aleatório, (podendo β_0 ser, ou não, nulo);

[17] O correlograma é igualmente útil para inferir sobre a ordem do modelo autorregressivo, como descrito na Sec. 2.4.1.

- Se $\rho = 1$ e $\beta_{i=0,1} \neq 0$, obtemos um passeio aleatório com *drift*[18] e tendência determinística;
- Se $0 < \rho < 1$, existe autocorrelação temporal, mas não tendência estocástica, sendo $\{y_t\}_{t\in T}$ estacionária.

Deste modo, estudar uma série quanto à estacionariedade corresponde a verificar se $\rho = 1$, isto é, testar a existência de raiz unitária (designação usual). Portanto, ao efetuar a regressão de y_t em y_{t-1}, a ideia associada aos testes de raiz unitária é verificar se o coeficiente, ρ, é estatisticamente igual a 1.

TESTE DE RAIZ UNITÁRIA		
$\underbrace{H_0: \rho = 1}$ (série não estacionária)	*vs*	$\underbrace{H_1: \rho < 1}$ (série estacionária)

Com efeito, de acordo com o teste de hipóteses (teste unilateral esquerdo com estatística-t em que $\tau = (\hat{\rho} - 1)/\hat{\sigma}_{\hat{\rho}}$), caso não se rejeite $\rho = 1$ dizemos que existe uma raiz unitária, ou seja, a série é não-estacionária.

De entre possíveis variantes dos testes de raiz unitária/estacionariedade citados na literatura, por considerarem hipóteses nulas diferentes (procurando dar maior consistência à decisão tomada), a nossa opção passou por analisar dois testes: os testes de raiz unitária de *Dickey-Fuller* (DF) / *Augmented Dickey-Fuller* (ADF) e o teste de estacionariedade de *Kwiatkowski-Phillips-Schmidt-Shin* (KPSS).[19]

2.2.5.1. *DICKEY-FULLER*

Relativamente ao teste DF, este parte do pressuposto de que os erros são iid, não apresentado assim qualquer autocorrelação. Tendo isso em consideração, $u_t \equiv \varepsilon_t$, e atendendo ao modelo definido na equação (2.28), se subtrairmos y_{t-1} em ambos os membros, obtemos (**Caso 1**: com constante e tendência)

[18] Uma das características destes processos é a persistência dos choques aleatórios (processos com memória infinita). O efeito de cada termo de erro não se dissipa ao longo do tempo, o processo guarda a informação de todos os choques sofridos até ao período corrente.

[19] Para mais detalhes sobre os testes DF/ADF veja-se Fuller (Fuller, 1996). Genericamente, para outros testes aqui citados e/ou referidos na literatura, veja-se Davidson e MacKinnon (1993) e Kirchgässner e Wolters (2007).

$$\underbrace{y_t - y_{t-1}}_{\Delta y_t} = \underbrace{(\rho - 1)}_{\alpha} y_{t-1} + \beta_0 + \beta_1 t + \varepsilon_t \tag{2.29}$$

Com efeito, temos:

TESTE DE RAIZ UNITÁRIA DE DICKEY-FULLER		
$\underbrace{H_0: \alpha = \rho - 1 = 0}_{\text{(série não estacionária)}}$	*vs*	$\underbrace{H_1: \alpha < 0}_{\text{(série estacionária)}}$

Trata-se igualmente de um teste unilateral esquerdo que se pode basear no rácio-t de α: $\tau_{ct} = \hat{\alpha}/\hat{\sigma}_{\hat{\alpha}}$. Porém, como mostraram Dickey e Fuller (*apud* Fuller, 1996) a distribuição da estatística de teste (sob H_0) não é t-Student nem Normal. A distribuição em causa é assimptótica (envolvendo os chamados processos de Wiener/movimentos Brownianos), não simétrica e com mediana e moda menores que zero, designando-se distribuição DF por ter sido tabelada por Dickey e Fuller, usando métodos de simulação (Fuller, 1996), de onde se podem posteriormente obter os valores críticos (ver Tabela 2.1).

Tal como referido por Dickey e Fuller (*apud* Fuller, 1996), nem sempre se justifica a inclusão do termo de tendência determinística na equação de teste, vindo a equação (2.28) reescrita por (**Caso 2**: com constante e sem tendência)

$$\Delta y_t = \alpha\, y_{t-1} + \beta_0 + \varepsilon_t \tag{2.30}$$

Finalmente, caso se verifique um comportamento oscilatório em torno de zero, podemos ainda retirar a constante β_0 da equação (2.30), obtendo-se (**Caso 3**: sem constante e sem tendência)

$$\Delta y_t = \alpha\, y_{t-1} + \varepsilon_t \tag{2.31}$$

Em relação ao teste de hipóteses, este mantem-se igual ao indicado no **Caso 1**, sendo que: (i) no **Caso 2** é testado com base no rácio-t de α denotado por $\tau_c = \hat{\alpha}/\hat{\sigma}_{\hat{\alpha}}$ em que a distribuição estatística se desloca para a direita; (ii) no **Caso 3** a estatística denotada por $\tau = \hat{\alpha}/\hat{\sigma}_{\hat{\alpha}}$ e cuja distribuição estatística sofre igualmente um deslocamento para a direita (e um achatamento em relação ao casos anteriores – ver Figura B.4).

Os valores críticos, para cada um dos casos, são apresentados na Tabela 2.1.

TABELA 2.1 – Valores críticos assimptóticos para os testes DF

Estatística	1%	2.5%	5%	10%
τ_{ct}	-3.96	-3.67	-3.41	-3.13
τ_c	-3.42	-3.12	-2.86	-2.57
τ	-2.58	-2.23	-1.95	-1.62

Fonte: Adaptado de Fuller, (1996, p. 642)

2.2.5.2. *AUGMENTED DICKEY-FULLER*

Repare-se que existem algumas limitações no teste DF, uma vez que o modelo assumido sob hipótese nula apenas contempla uma componente autorregressiva para o modelo. Tal pode ser insuficiente para capturar a evolução da série em função dos seus valores passados, conduzindo a que a componente residual apresente autocorrelação (repare-se que $u_t \equiv \varepsilon_t$ foi uma condição imposta à priori).

A solução passa então por introduzir mais componentes autorregressivas no modelo, de modo a evitar que a componente residual descrita na equação (2.28) apresente autocorrelação, adicionando-se assim uma correção paramétrica no teste DF (Said & Dickey, 1984). Na prática, o teste ADF distingue-se do teste DF pelo facto de introduzir na equação *lags* com o objetivo de 'branquear' os resíduos (eliminar a autocorrelação). Deste modo, nos testes ADF é considerado um 'novo' modelo que se obtém de (2.29) adicionando valores desfasados, Δy, das variáveis independentes, vindo

$$\Delta y_t = \alpha y_{t-1} + \sum_{i=1}^{k} \alpha_i \Delta y_{t-i} + \beta_0 + \beta_1 t + \varepsilon_t \qquad (2.32)$$

onde, além da constante β_0, do coeficiente de tendência determinística, β_1, e do coeficiente de presença de raiz unitária, α, o valor de k diz respeito aos desfasamentos a integrar na série, garantindo que os resíduos, ε_t, não apresentam autocorrelação.

A questão prática que se coloca, neste caso, é qual o número k (*lags*) a considerar. A estratégia será iniciar o processo com um k suficientemente grande e ir tentando simplificar a autorregressão com testes-t individuais sobre os coeficientes de desfasamento mais elevados, até se obter uma rejeição. Contudo, encontramos na bibliografia algumas sugestões mais práticas. Por exemplo, Schwert (1989), analisando esta questão e procurando verificar a existência de possíveis limitações dos teste DF/ADF face a outros então surgidos na literatura,

sugere uma regra empírica para a determinação dos *lags*, k, onde, partindo da dimensão, n, da série, k é dado pela parte inteira obtida da expressão

$$\left[12\left(\frac{n}{100}\right)^{1/4}\right] \tag{2.33}$$

2.2.5.3. *KWIATKOWSKI-PHILLIPS-SCHMIDT-SHIN*

Com a mesma finalidade, avaliar a estacionariedade em uma série temporal, em alternativa aos testes de raiz unitária DF/ADF, Kwiatkowski *et al.* (1992) propuseram o teste de estacionariedade KPSS.

TESTE DE KWIATKOWSKI-PHILLIPS-SCHMIDT-SHIN (1)
H_0: $\{y_t\}_{t\in T}$ é estacionária (em tendência) *vs* H_1: $\{y_t\}_{t\in T}$ não é estacionária

Este teste complementa os usuais testes de raiz unitária na medida em que, ao testar a hipótese nula de que uma série é estacionária em torno de uma tendência determinística, contra a hipótese de não estacionariedade, permitem uma melhor clarificação em situações dúbias (em séries cujos dados não são suficientemente informativos para garantir a existência de raiz unitária).[20]

Com efeito, assumindo que a série $\{y_t\}_{t\in T}$, com n observações, é expressa como a soma de três componentes[21], tendência determinística, passeio aleatório e componente residual (estacionária), os referidos testes colocam a teste o facto de o passeio aleatório ter variância nula.

Matematicamente, considerando (2.34)

$$y_t = \beta_1 t + \xi_t + \varepsilon_t \tag{2.34}$$

onde ε_t é estacionário e ξ_t denota o passeio aleatório, com

[20] De sublinhar que, no teste KPSS, a não rejeição de H_0 não é prova imediata de estacionariedade, mas de estacionariedade em tendência (Kwiatkowski *et al.*, 1992).

[21] Sem perda de generalidade, consideremos a abordagem sem a constante β_0.

$$\xi_t = \xi_{t-1} + \zeta_t, \quad \zeta_t \sim WN(0, \sigma_v^2) \tag{2.35}$$

podemos afirmar que no caso de $\sigma_\zeta^2 = 0$, temos $\xi_t = \xi_0$, para qualquer $t \in T$, pelo que $\{y_t\}_{t\in T}$ é estacionária.

Deste modo, o teste KPSS pode ser reescrito na forma abaixo.

TESTE DE KWIATKOWSKI-PHILLIPS-SCHMIDT-SHIN (2)		
$H_0: \sigma_\zeta^2 = 0$	*vs*	$H_1: \sigma_\zeta^2 > 0$

em que a estatística de teste é expressa em (2.36)

$$\tau_{KPSS} = \frac{1}{n^2} \sum_{t=1}^{n} \frac{{S_{\varepsilon_t}}^2}{\widehat{\sigma}_{\varepsilon_t}^2} \tag{2.36}$$

onde o numerador S_{ε_t} denota a soma parcial dos resíduos, $\sum_{i=1}^{t} \hat{\varepsilon}_i$, e o denominador $\hat{\sigma}_{\varepsilon_t}^2$ um estimador para a sua variância.

É possível provar que a estatística τ_{KPSS} tem distribuição que converge assintoticamente para um *Movimento Browniano* (Kwiatkowski *et al.*, 1992) em que os valores críticos são tabelados (ver Tabela 2.2)

TABELA 2.2 – Valores críticos assintóticos para o teste de estacionariedade KPSS

Estatística	1%	2.5%	5%	10%
τ_{KPSS} (sem qualquer tendência)	0.739	0.574	0.463	0.347
τ_{KPSS} (com algum tipo de tendência)	0.216	0.176	0.146	0.119

Fonte: Adaptado de Kwiatkowski *et al.* (1992, p. 166)

2.2.6. QUEBRAS DE ESTRUTURA

Além dos entraves no processo de modelação e previsão de séries temporais que resultam da não estacionariedade, surge outro problema quando ocorrem fortes perturbações nos parâmetros. Tal como a não estacionariedade, este fenómeno, referido na literatura como a "existência de quebras de estrutura", está muitas vezes associado a situações marcantes que

fogem completamente ao padrão (nomeadamente em séries temporais de cariz económico-financeiro), condicionando o estudo das mesmas (Valentinyi-endr, 2004).

Segundo a literatura científica, existe uma forte relevância (com possíveis consequências) em estudos empíricos onde é detetada a presença de quebras de estrutura em séries temporais. Daí, ser salientada a importância de considerar a análise deste fenómeno para evitar incorreções quer no que respeita à aceitação de modelos, quer em consequentes cenários de previsão (Hansen, 2001 e Antoch *et al.*, 2019).

> *Structural change is pervasive in economic time series relationships, and it can be quite perilous to ignore. Inferences about economic relationships can go astray, forecasts can be inaccurate, and policy recommendations can be misleading or worse*. (Hansen, 2001, p. 127)

Considera-se existir uma quebra de estrutura quando algum dos parâmetros presente no modelo sofre uma alteração (anormal) num determinado momento – data de quebra. As alterações estruturais podem apresentar-se de várias formas, como, por exemplo: (i) afetarem todos os parâmetros do modelo ou apenas parte deles; (ii) ocorrerem de forma abrupta ou gradual; (iii) ocorrerem numa data conhecida ou desconhecida; (iv) ocorrem uma única vez, ou como um fenómeno múltiplo num determinado período. Certo é que, quanto maior for o intervalo temporal dos dados em análise, maior a probabilidade da ocorrência de quebras de estrutura.

Neste seguimento, alguns estudos ulteriores envolvendo modelação e previsão têm procurado responder a desafios como testar endogenamente a presença de quebras de estrutura e construir um algoritmo que as possa localizar.[22]

Formalizando, Bai e Perron (1998) forneceram uma estrutura padrão para os modelos com quebras de estrutura. Assim, considerando o período de amostragem com n observações, sendo a data de quebra denotada por T_1, em termos de regressão[23], o modelo de quebra de estrutura completa é definido em (2.37)

[22] No nosso caso, importa deixar claro que o objetivo é o de melhor caraterizar a série temporal em estudo e, com isso averiguar a existência de quebras de estrutura. Deste modo, a preocupação maior estará em identificar e implementar testes estatísticos que nos ajudem na referida tarefa.

[23] Opta-se por usar o modelo de regressão na abordagem feita nesta secção, ao invés do modelo autorregressivo em séries temporais, para manter uma concordância com as fontes bibliográficas citadas.

$$y_t = \begin{cases} \beta_0 + \beta_1' x_t + u_t \ , & t \leq T_1 \\ \beta_0 + \beta_2' x_t + u_t \ , & t > T_1 \end{cases} \tag{2.37}$$

ou, usando a função Heaviside, $\mathbf{1}_t$, pela equação (2.38)

$$y_t = \beta_0 + \beta_1' x_t \mathbf{1}_{(t \leq T_1)} + \beta_2' x_t \mathbf{1}_{(t > T_1)} + u_t \tag{2.38}$$

Deste modo, um possível teste de hipóteses, subjacente à análise da presença de quebras de estrutura, poderá ser enunciado do seguinte modo

TESTE DE QUEBRAS DE ESTRUTURA (1)		
$\underbrace{H_0: \beta_1' = \beta_2'}_{\text{(sem quebra de estrutura)}}$	*vs*	$\underbrace{H_1: \beta_1' \neq \beta_2'}_{\text{(existência de quebra de estrutura)}}$

Alternativamente, pode averiguar-se a existência de quebra de estrutura no modelo através de uma análise à variância da componente residual, u_t. Neste caso, considerando o modelo descrito na equação (2.39)

$$y_t = \beta_0 + \beta_1 x_t + u_t \tag{2.39}$$

podemos considerar

$$Var(u_t) = \begin{cases} \sigma_1^2 \ , & t \leq T_1 \\ \sigma_2^2 \ , & t > T_1 \end{cases} \tag{2.40}$$

Pelo que, uma outra abordagem para teste de hipóteses, subjacente à análise da presença de quebras de estrutura, poderá ser enunciado do seguinte modo

TESTE DE QUEBRAS DE ESTRUTURA (2)		
$\underbrace{H_0: \sigma_1^2 = \sigma_2^2}_{\text{(sem quebra de estrutura)}}$	*vs*	$\underbrace{H_1: \sigma_1^2 \neq \sigma_2^2}_{\text{(existência de quebra de estrutura)}}$

Sendo viável qualquer uma das abordagens, surge a questão: "Qual o tipo de teste que se adequa melhor à identificação da presença de quebras de estrutura em série temporais?".

Neste item, a tarefa parece ser um pouco dificultada, uma vez que, de acordo com a literatura científica sobre o tema, e de entre uma variedade de testes propostos, não há uma escolha linear do 'melhor' a aplicar. No entanto, pode depreender-se que, nessa escolha, devem ser tidos em conta alguns fatores, tais como: (i) se o momento da quebra é conhecido ou não; (ii) se existe possibilidade de uma única ou múltiplas quebras; (iii) qual o tipo de abordagem, em termo de teste de hipóteses, que se pretende avaliar.

Assim, em função da série e do que é conhecido relativamente ao momento/tipo(s) de quebra(s), procurando sintetizar alguma informação constante na literatura científica, temos: [24]

- se o momento da quebra na média é conhecido, um dos testes sugeridos na literatura é o teste de *Chow* (Chow, 1960). Quando os erros seguem uma distribuição normal, a literatura sugere a utilização deste teste para avaliar a ausência de quebras no modelo testando a igualdade dos coeficientes de regressão em períodos distintos (H_0: $\beta'_1 = \beta'_2$);
- se a localização da quebra, k, não é conhecida, uma alternativa é o teste desenvolvido por Quandt (1960), o *Quandt Likelihood Ratio*. Primeiramente a estatística de teste *Wald* é calculada para todos os possíveis valores de k e só depois se considera o seu valor máximo, denominado por *SupWald*. Posteriormente, Andrews (1993, 2003) fez desenvolvimentos, aproximando a distribuição da estatística de teste *SupWald* e determinou os seus valores críticos. Quando a hipótese nula é rejeitada, a localização da quebra pode ser estimada através do *argmaxSupWald*.[25]

Uma outra alternativa referida na literatura é o teste proposto por Brown, Durbin e Evans (1975), denominado CUSUM[26] por recorrer à soma cumulativa (*Cumulative Sum*) de resíduos recursivos (ou à soma cumulativa de resíduos de mínimos quadrados ordinários – OLS – *Ordinary Least Squares*) para testar a existência de quebra de estrutura, por análise de variações

[24] Entre outros, veja-se por exemplo Hansen (2001), Perron (2006) (ou, mais recentemente, Casini e Perron (2018)), ou ainda Aue e Horváth (2013). Esta última referência descreve alguns dos trabalhos sobre quebras estruturais em modelos de séries temporais, invocando vários testes usados. Mais recentemente, um bom contributo em termos de revisão da literatura, é feito em Morshed *et al.* (2018), onde são evidenciadas algumas limitações do teste *SupWald*, baseadas num estudo empírico envolvendo três séries dos EUA: desemprego, crescimento da produção industrial e taxas de juros. Ainda, para uma apresentação de alguns modelos e uma avaliação sucinta de prós e contras de cada um deles, veja-se Aptech Systems, Inc. (2019).

[25] A identificação do número e localização das quebras também pode ser obtida através da seleção do modelo com base em critérios de informação. Se, numa determinada localização, k, o modelo apresentar um valor de critério de informação inferior ao do modelo sem quebras de estrutura, isso será suficiente para provar a existência de quebras.

[26] Ou uma versão adaptada *CUSUM-sq* (*CUSUM* ao quadrado).

(ao longo do tempo) na componente residual. De modo sucinto, este teste, tomando como hipótese nula a estabilidade do(s) parâmetro(s), pode ser visto como um teste à estabilidade na variância da componente residual, como descrito em (2.40).

Assim, com base na estimativa recursiva de OLS do modelo descrito em (2.39), para cada momento $1 \leq t \leq n$ (considerando omisso, sem perda de generalidade, o termo constante, β_0)

$$y_t = \beta_t x_t + u_t \quad \text{com} \quad Var(u_t) = \sigma_t^2 \tag{2.41}$$

são produzidas estimativas para os $\beta_i,\ i = 1, \dots, n$. A partir daqui a estatística do teste é calculada a partir dos resíduos *one-step-ahead* do modelo OLS recursivo, de onde se pode inferir que, se o valor de β muda no tempo (isto é, se existe $\beta_i \neq \beta_j,\ i \neq j$), a previsão não será precisa e o erro de previsão será maior que zero. Deste modo, pode considerar-se o seguinte teste de hipóteses

TESTE DE QUEBRAS DE ESTRUTURA – CUSUM

$$\underbrace{H_0: \forall\, i \neq j,\ \ \sigma_i^2 = \sigma_j^2}_{\text{(sem evidências de quebra de estrutura)}},\ \ i, j = 1, \dots n \qquad vs \qquad \underbrace{H_1: \exists\, i \neq j,\ \ \sigma_i^2 \neq \sigma_j^2}_{\text{(evidências de existência de quebra de estrutura)}},\ \ i, j = 1, \dots n$$

Numa perspetiva de escolha de teste a usar na nossa abordagem, evidenciando a atualidade desta metodologia e a sua aplicabilidade em séries temporais, Aue e Horváth (2013) mostraram que procedimentos baseados no teste de CUSUM, ainda que com eventuais adaptações, funcionam igualmente em dados que exibem dependência temporal, sendo analisadas quebras na estrutura da média e da variância. Os mesmos autores abordam ainda a estimativa de múltiplas quebras, questão de interesse no estudo de séries temporais.

Ainda, com base nos fundamentos teóricos do CUSUM, em Gustafsson (2000) é apresentada uma forma simplificada e prática de um algoritmo recursivo baseado em somas cumulativas, o algoritmo CUSUM, para a deteção de alterações bruscas. Em breves palavras, tratando-se de um algoritmo recursivo, o objetivo é identificar se ocorreu uma mudança brusca entre dois instantes e precisar qual o momento em que se regista essa mudança.[27]

[27] Para a deteção das quebras de estrutura, de entre as várias possibilidades referidas, a nossa opção em termos de implementação computacional foi esta última. No entanto, procurando não nos dispersarmos e evitar uma apresentação mais extensa, para mais detalhes sobre os fundamentos teóricos implícitos no algoritmo, veja-se Granjon (2014). Não obstante, mais adiante, serão feitas considerações mais detalhadas sobre a implementação computacional do algoritmo CUSUM.

2.3. PREVISÃO EM SÉRIES TEMPORAIS

A tarefa inerente à previsão reveste-se de forte utilidade, constituindo uma ferramenta de planeamento quaisquer que sejam as circunstâncias ou horizontes de tempo envolvidos. No que respeita ao estudo de séries temporais, em concreto, boas previsões capturam os padrões que existem nos dados históricos, sendo importante distinguir entre uma flutuação aleatória, que deve ser ignorada, e um padrão genuíno, que deve ser modelado e extrapolado (Hyndman & Athanasopoulos, 2018).

Objetivamente, duas questões determinantes serão: "Que informações históricas devem ser usadas?" e "Qual o horizonte de previsão pretendido?". É na tentativa de encontrar uma resposta para cada uma das questões enunciadas que se procurará não só entender que preocupações devem ser tidas em relação aos dados como também verificar se, de acordo com a janela de previsão pretendida, existem especificações e diferentes métodos/modelos de previsão a considerar no processo preditivo.

Em séries temporais, os métodos de previsão mais simples usam apenas informações sobre a variável a ser prevista procurando, em particular, extrapolar tendências e padrões sazonais. Esses métodos podem ser simples, como utilizar as últimas observações para fazer extrapolação, ou mais complexos, como por exemplo estabelecer modelos econométricos que, usando um volume de histórico de informação maior, permitem melhorar a capacidade preditiva por distinguirem o que é aleatório daquilo que deve ser considerado no modelo.

Deste modo, um dos requisitos dos modelos de previsão de séries temporais é que as observações passadas contenham a informação fundamental da série e que esse(s) padrão(ões) seja(m) recorrente(s) ao longo do tempo. Acontece que, tal como já referido acima, parte da eficácia dos modelos está em conseguir lidar e distinguir a parte da informação que é padrão da parte que é ruído (Makridakis *et al.*, 1998).

Nesta linha de raciocínio, os modelos de previsão em séries temporais poderão ser encarados como modelos quantitativos que, tendo por base as observações passadas da série (e no seu inter-relacionamento), procuram fornecer previsões para os momentos futuros. Ou seja, independentemente do modelo de previsão considerado e formalizando o conceito de previsão, considerando a série temporal $\{y_t\}_{t\in T}$, o objetivo é prever o valor para um período futuro $t+h$ (com $h \geq 1$), $\hat{y}_{t+h}$, supondo que são conhecidas todas as observações até ao momento t, ou seja, $y_1, y_2, \dots, y_{t-1}, y_t$.

2.4. ALGUNS MODELOS ECONOMÉTRICOS DE PREVISÃO (CLÁSSICOS)

No seguimento do exposto na secção anterior, iremos agora discutir alguns dos modelos univariados de previsão 'clássicos' comummente referidos e utilizados na literatura, dando ênfase às famílias de modelos autorregressivos de médias móveis e de alisamento exponencial.

2.4.1. MODELOS AUTORREGRESSIVOS DE MÉDIAS MÓVEIS

Os modelos ARMA (*Autoregressive Moving Average*) assumem um papel preponderante na análise de séries temporais, com particular aplicação na sua modelação e previsão (Kallas, Honeine, Francis, & Amoud, 2013).

Tal como destacado por Granger & Newbold (1974), esta família de modelos tem a capacidade de descrever os processos de geração de uma multiplicidade de séries temporais, sem necessitar de informações sobre as relações económicas subjacentes. É, assim, uma metodologia flexível, em que as previsões têm por base os valores correntes e passados, com aplicabilidade a séries com comportamento estacionário ou não estacionário[28]. Mais, vários autores, em virtude dos estudos desenvolvidos, salientam que são excelentes modelos de previsão de curto prazo, algo que procuraremos indagar no estudo em causa, em virtude da natureza das séries temporais em estudo.

Passemos a uma descrição sucinta e a uma formalização da família de modelos ARMA.

2.4.1.1. *AUTOREGRESSIVE*

Contrapondo com os modelos simples de regressão linear, os modelos AR (*Autoregressive*) caracterizam-se pelo facto das variáveis regressoras serem os valores de um processo no tempo em p instantes anteriores. Com efeito, denotando os valores do processo no tempo, $t, t-1, t-2, \dots$, por $y_t, y_{t-1}, y_{t-2}, \dots$, a equação (2.42)

$$y_t = c + \sum_{i=1}^{p} a_i y_{t-i} + \varepsilon_t \tag{2.42}$$

caracteriza o modelo AR de ordem p, $AR(p)$, onde $a_1, a_2, a_3, \dots$ são os coeficientes do modelo e ε_t é o fator de aleatoriedade que reflete a informação recebida no instante t.

[28] Embora seja requerida a estacionariedade para a sua implementação computacional (ver Sec. 4.3.2).

Sendo L o operador *lag*, representado por $L(y_t) = y_{t-1}$, o modelo $AR(p)$ pode também ser representado pela equação (2.43)

$$\phi_p(L)y_t = c + \varepsilon_t \tag{2.43}$$

em que $\phi_p(L) = 1 - \sum_{i=1}^{p} a_i L^i$.

Sobre a condição de estacionaridade de modelo $AR(p)$, esta "… é vulgarmente apresentada na literatura como as raízes do polinómio característico do operador *lag* permanecerem fora do círculo unitário …" (Johnston & DiNardo, 2001, p. 236). Assim, um modelo $AR(p)$ diz-se estacionário se todas as raízes $z = \frac{1}{a_i}$ do polinómio característico $\phi_p(z) = 1 - \sum_{i=1}^{p} a_p z^p$ estiverem situadas no exterior do círculo unitário, isto é, $|a_i| < 1$.

Desta forma, em modelos autorregressivos $AR(p)$ estacionários, assumindo-se muitas vezes (e sem perda de generalidade) que $\mathbb{E}(y_t) = 0$,[29] tendo em conta o definido nas equações (2.21) e (2.22), temos:

- a **FAC**, para $k = 1, \dots, p$, vem dada por

$$\rho_k = a_1\rho_{k-1} + a_2\rho_{k-2} + \cdots + a_p\rho_{k-p} \tag{2.44}$$

- a **variância**, como função das autocorrelações, vem dada por

$$Var(y_t) = \frac{\sigma^2}{a_1\rho_1 + a_2\rho_2 + \cdots + a_k\rho_k} \tag{2.45}$$

Em virtude disto, esperamos que a FAC empírica de uma série $AR(p)$ estacionária decresça para zero e que a FACP seja, aproximadamente, nula para ordens superiores a p.

2.4.1.2. *MOVING AVERAGE*

Os modelos MA (*Moving Average*) utilizam a média das observações ocorridas no passado para efetuar previsões. Como descrito por Murteira *et al.*, (1993), estes modelos resultam de exprimir y_t em termos de um processo aleatório, ε_t. Assim, os efeitos produzidos pelas inovações só perduram por um período curto, o que contrasta com os processos autorregressivos, em que os efeitos persistem ao longo do tempo.

[29] Em processos y_t estacionários, a média surge definida por $\mathbb{E}(y_t) = \mu = \frac{c}{1-(a_1+a_2+\cdots+a_p)}$. Todavia, é comum assumir-se, sem perda de generalidade, que $\mathbb{E}(y_t) = 0$, considerando uma translação vertical de y_t.

Em termos formais, o inverso de um processo $AR(1)$ traduz um processo MA de ordem infinita, $MA(\infty)$.[30] Assim, um processo MA de ordem q, $MA(q)$, poderá ser descrito por

$$y_t = \varepsilon_t - \sum_{j=1}^{q} b_j \varepsilon_{t-j} \tag{2.46}$$

em que ε_t é um processo aleatório e $b_1, \dots, b_q$ são constantes reais. Isto significa que um processo de médias móveis de ordem q se define, em cada instante t, como a média ponderada das últimas $q+1$ observações de um processo de ruído branco.

Sendo L o operador *lag*, representado por $L(\varepsilon_t) = \varepsilon_{t-1}$, o modelo $MA(q)$ pode também ser representado pela equação (2.47)

$$y_t = \theta_q(L)\varepsilon_t \tag{2.47}$$

onde $\theta_q(L) = 1 - \sum_{j=1}^{q} b_j L^j$ é o polinómio das médias móveis de ordem q

Observe-se que o processo $MA(q)$ é sempre estacionário[31], independentemente dos valores de $b_1, \dots, b_q$. Deste modo, sendo $\mathbb{E}(\varepsilon_t) = 0$ em que ε_t são não correlacionados:

- a **média** e a **variância** são dadas, respetivamente, por

$$\mathbb{E}(y_t) = 0 \text{ e } Var(y_t) = \sigma^2\left(1 + b_1^2 + \dots + b_q^2\right) \tag{2.48}$$

- a **FACV** (com $\gamma_0 = Var(y_t)$), vem dada por

$$\gamma_k = \begin{cases} -\sigma^2\left(b_k + b_{k+1}b_1 + \dots + b_q b_{q-k}\right), & 0 < k \leq q \\ 0 \quad , & k > q \end{cases} \tag{2.49}$$

- a **FAC** vem dada por

$$\rho_k = \begin{cases} -\dfrac{b_k + b_{k+1}b_1 + \dots + b_q b_{q-k}}{1 + b_1^2 + \dots + b_q^2}, & k \leq q \\ 0 \quad , & k > q \end{cases} \tag{2.50}$$

[30] Reciprocamente, o inverso de um processo $MA(1)$ é um processo $AR(\infty)$ (ver Johnston e DiNardo (2001)).

[31] Em virtude de um processo MA poder ser invertido originando um processo AR, embora não se imponha nenhuma condição para a estacionariedade dos modelos MA, existem algumas condições a impor ao modelo, conhecidas como "condições de invertibilidade". Para mais detalhes, veja-se Johnston e DiNardo (2001).

Em virtude do exposto, num processo estacionário $MA(q)$, os q primeiros coeficientes de autocorrelação são não nulos, sendo os restantes iguais a zero, pelo que a FAC é nula a partir da ordem q. Já os coeficientes de autocorrelação parcial decaem de forma amortecida e gradual para zero; ou seja, a FACP tem decrescimento exponencial amortecido para zero.

2.4.1.3. *AUTOREGRESSIVE MOVING AVERAGE*

O modelo ARMA (*Autoregressive Moving Average*)[32] resulta de uma combinação do modelo AR com o processo MA, tal como se depreende da equação (2.51) que o caracteriza

$$y_t = c + \underbrace{\sum_{i=1}^{p} a_i y_{t-i}}_{\substack{(1-\phi_p(L))\, y_t \\ \downarrow \\ componente\ AR}} + \underbrace{\varepsilon_t - \sum_{j=1}^{q} b_j \varepsilon_{t-j}}_{\substack{\theta_q(L)\varepsilon_t \\ \downarrow \\ componente\ MA}} \tag{2.51}$$

a qual pode ser expressa, em função dos operadores *lag*, pela equação (2.52)

$$\phi_p(L) y_t = c + \theta_q(L)\varepsilon_t \tag{2.52}$$

A estacionariedade e a invertibilidade do modelo exigem que as raízes de $\phi_p(L)$ e $\theta_q(L)$ se situem fora do círculo unitário. Estas condições permitem que o processo seja expresso quer como um processo AR, quer como um processo MA, ambos de ordem infinita, pelo que, considerando nulo o termo constante, c, obtemos

$$\theta_q^{-1}(L)\phi_p(L) y_t = \varepsilon_t \quad \text{ou} \quad y_t = \phi_p^{-1}(L)\theta_q(L)\varepsilon_t \tag{2.53}$$

Para processos ARMA estacionários de ordens p e q, ARMA(p, q), tem-se:

- a **média** é dada por:

$$\mathbb{E}(y_t) = \frac{c}{1 - a_1 - a_2 - \cdots - a_p} \tag{2.54}$$

- a **FAC** apresenta combinações de comportamentos dos modelos $AR(p)$ e $MA(q)$. No entanto, quando estamos a considerar *lags* inferiores a q, a FAC é idêntica à de um

[32] Em 1926, Yule introduziu os processos AR para o estudo de séries temporais, mas só em 1937 foram apresentados, por Slutsky, os processos MA. Em 1938, Wold juntou os resultados de Yule e Slutsky e apresentou a primeira versão do modelo ARMA para modelar séries temporais estacionárias.

modelo $AR(p)$, com valores decrescendo para zero, com taxa de decréscimo dada pelos parâmetros do modelo AR.[33]

Face ao exposto, se até é relativamente simples identificar modelos AR e MA puros (em virtude do decréscimo brusco para zero da FAC ou da FACP), no caso dos modelos ARMA o processo não é tão imediato, uma vez que nenhuma das funções apresenta um decréscimo brusco para zero. A Tabela 2.3 procura sintetizar os 'padrões' de correlação esperados para cada um dos três processos, sendo feitas algumas considerações complementares e mais detalhadas no Anexo B.2.

TABELA 2.3 – Comportamento da FAC e da FACP para modelos AR, MA e ARMA

	Padrões de correlação	
Processo	Função de autocorrelação (FAC)	Função de autocorrelação parcial (FACP)
AR(p)	*Infinita*: Decresce para zero (exponencialmente ou segundo uma sinusoidal 'amortecida').	*Finita*: Decresce abruptamente para zero a partir de *lag* p.
MA(q)	*Finita*: Decresce abruptamente para zero a partir de *lag* q ou segundo uma sinusoidal 'amortecida'.	*Infinita*: Decresce para zero (exponencialmente ou segundo uma sinusoidal 'amortecida').
ARMA(p,q)	*Infinita*: Decresce para zero exponencialmente ou segundo uma sinusoidal 'amortecida' após o *lag* $p-q$.	*Infinita*: Decresce para zero (exponencialmente ou segundo uma sinusoidal 'amortecida').

Fonte: Adaptado de Johnston & DiNardo (2001, p. 240)

Todos estes modelos permitem efetuar previsões a partir de dados observados. Contudo, apresentam algumas limitações quando se deparam com séries temporais com comportamentos de tendência, sazonalidade ou aleatoriedade que conduzam à existência de assimetrias, fortes volatilidades ou eventuais quebras de estrutura. Tais características podem ser indício da não estacionariedade da série, constituindo uma limitação à implementação destes modelos.

[33] Ver Johnston & DiNardo (2001, p. 239).

2.4.1.4. *AUTOREGRESSIVE INTEGRATED MOVING AVERAGE*

Os modelos ARIMA (*Autoregressive Integrated Moving Average*) surgem no sentido de fazer face à limitação supramencionada de não estacionariedade da série.

Uma das alternativas consiste num pré-processamento (transformação) dos dados que permita 'remover' algumas características, em particular por aplicação de diferenças sucessivas, donde surgem dois conceitos importantes: séries integradas e ordem de integração. Uma série temporal é integrada de ordem $d \in \mathbb{N}$, e escrevemos $y_t \sim I(d)$, se a série é não estacionária em nível, mas pode ser transformada numa série estacionária após d diferenças sucessivas, pelo que a ordem de integração corresponde ao menor número de vezes que a série necessita de ser diferenciada para se obter uma série estacionária.[34]

Uma vez conseguida a estacionariedade da série por este procedimento, pode então aplicar-se um modelo ARMA, obtendo-se uma nova versão designada de modelo ARIMA. Assim, define-se o modelo $ARIMA\ (p, d, q)$, onde d representa o número de diferenças necessárias para tornar a série estacionária, p representa a ordem da componente AR e q representa a ordem da componente MA.

Os modelos ARIMA representam, em teoria, a classe geral dos modelos utilizados na previsão de uma série temporal que não sendo estacionária em nível pode tornar-se estacionária por diferenças sucessivas.

Formalizando, dada uma série não-estacionária, y_t, define-se a série das diferenças de ordem d, w_t, por (2.55)

$$w_t = \Delta^d y_t = (1 - L)^d y_t \tag{2.55}$$

a qual é estacionária após *d* diferenças, permitindo a aplicação de um modelo ARMA (p, q).

Usando os operadores *lag*, tal como definidos anteriormente, podemos combinar toda a informação numa única equação (2.56), a qual caracteriza os modelos $ARIMA\ (p, d, q)$[35]

[34] Por exemplo, se a série precisar ser diferenciada uma vez para atingir a estacionariedade, escrevemos $y_t \sim I(1)$. Observe-se que, se y_t não necessitar de ser diferenciada nenhuma vez para se tornar estacionária (embora possa ser não estacionária, como no caso de TS), poder-se-á escrever $y_t \sim I(0)$, significando que ela é integrada de ordem zero.

[35] Podemos descrever todos os modelos vistos anteriormente (AR, MA e ARMA) utilizando, apenas, a nomenclatura ARIMA, em que (i) $ARIMA(p, 0,0) = AR(p)$; (ii) $ARIMA(0,0, q) = MA(q)$; (iii) $ARIMA(p, 0, q) = ARMA(p, q)$.

$$\underbrace{\left(1-\sum_{i=1}^{p} a_i L^i\right)}_{\substack{\phi_p(L) \\ \downarrow \\ \text{"componente AR"}}} \underbrace{(1-L)^d}_{\substack{\downarrow \\ d\ diferenças}} y_t = c + \underbrace{\left(1-\sum_{j=1}^{q} b_j L^j\right)}_{\substack{\theta_q(L) \\ \downarrow \\ \text{"componente MA"}}} \varepsilon_t \tag{2.56}$$

que, combinada com a equação (2.55), em w_t, pode ser reescrita de um modo mais simples[36]

$$\phi_p(L)(w_t - \mu) = \theta_q(L)\varepsilon_t \tag{2.57}$$

assumido que $\mu = \mathbb{E}(w_t)$ e $c = \mu\left(1-\sum_{i=1}^{p} a_i\right)$.

2.4.1.5. *SEASONAL AUTOREGRESSIVE INTEGRATED MOVING AVERAGE*

Em qualquer dos casos anteriores, não foi invocada explicitamente a possibilidade de presença da componente de sazonalidade nem as consequências que tal poderá ter no processo de modelação da série temporal. Contudo, os modelos ARIMA também são 'capazes' de se ajustar de uma forma robusta a dados sazonais.

Nesta linha, surgem os modelos SARIMA (*Seasonal Autoregressive Integrated Moving Average*) como forma de integrar sazonalidade nos modelos ARIMA acima apresentados. Estes modelos (na forma multiplicativa) apresentam na sua formulação duas 'partes'

$$\underbrace{(p,d,q)}_{\substack{\downarrow \\ Parte\ não\ sazonal \\ do\ modelo}} \times \underbrace{(P,D,Q)\ m}_{\substack{\downarrow \\ Parte\ sazonal \\ do\ modelo}}$$

onde m denota a frequência da sazonalidade.

Assim, mantendo a notação seguida nas secções anteriores, podemos caraterizar os modelos $SARIMA(p,d,q) \times (P,D,Q)m$ do seguinte modo

$$\underbrace{\underbrace{\left(1-\sum_{i=1}^{p} a_i L^i\right)}_{\substack{\phi_p(L) \\ \downarrow \\ \text{"componente AR"}}} \underbrace{\left(1-\sum_{i=1}^{P} A_i (L^m)^i\right)}_{\substack{\Phi_P(L^m) \\ \downarrow \\ \text{"componente AR sazonal"}}} \underbrace{(1-L)^d}_{\substack{\downarrow \\ d\ diferenças}} \underbrace{(1-L^m)^d}_{\substack{\downarrow \\ d\ diferenças \\ \text{"parte sazonal"}}} y_t}_{\substack{\downarrow \\ Componente\ do\ modelo\ AR \\ sazonal\ multiplicativo \\ AR(p)\times SAR(P)m}} = \underbrace{\underbrace{\left(1-\sum_{j=1}^{q} b_j L^j\right)}_{\substack{\theta_q(L) \\ \downarrow \\ \text{"componente MA"}}} \underbrace{\left(1-\sum_{j=1}^{Q} B_j (L^m)^j\right)}_{\substack{\Theta_Q(L) \\ \downarrow \\ \text{"componente MA sazonal"}}} \varepsilon_t}_{\substack{\downarrow \\ Componente\ do\ modelo\ MA \\ sazonal\ multiplicativo \\ MA(q)\times SMA(Q)m}}$$

[36] Esta é a parametrização usada em muitos *softwares*.

a qual, combinada com a equação acima, em w_t, pode ser reescrita de um modo mais simples

$$\phi_p(L)\Phi_P(L^m)w_t = \theta_q(L)\Theta_Q(L)\varepsilon_t \tag{2.58}$$

Para os modelos SARIMA estacionários, a dedução da FAC não é trivial, sendo extremamente difícil identificar a ordem de tais modelos através da análise do correlograma (Johnston & DiNardo, 2001). No entanto, centrando especial atenção no comportamento da parte AR do modelo, espera-se que as correlações decresçam à medida que os *lags* aumentam, verificando-se picos que vão decrescendo exponencialmente (sob os pressupostos habituais de estacionariedade) nos *lags* da forma $k \times m$, $k \in \mathbb{N}$. Por outro lado, quanto à FACP é expectável que esta tenha um decrescimento para zero, esperando-se valores cada vez mais próximos de zero, podendo surgir (igualmente) picos 'relevantes' nos nos *lags* da forma $k \times m$, $k \in \mathbb{N}$.

2.4.2. MODELOS DE ALISAMENTO EXPONENCIAL

A metodologia de alisamento exponencial, ETS (*Exponential Smoothing*), a qual se refere à modelação explicita de Erro, Tendência e Sazonalidade, embora tenha sido proposta no final da década de 1950,[37] é um clássico na modelação de séries temporais, por apresentar previsões confiáveis de rápida implementação e compreensão simples, o que se reverte numa vantagem para o mundo empresarial (Ord, 2004 e Hyndman & Athanasopoulos, 2018).

Resumidamente, os modelos ETS baseiam-se em médias ponderadas de observações anteriores, podendo os pesos decrescer, ou não, exponencialmente, à medida que as observações tendem para o passado. Ou seja, ao invés de os pesos serem considerados uniformes (caso particular de um alisamento de média móvel), há possibilidade de uma 'deterioração exponencial', podendo as observações recentes terem mais peso que as observações antigas. Assim, num dos modelos mais simples de ETS apenas se observa que quanto mais recentes forem as observações, maior será o peso associado (ou vice-versa); modelos mais avançados integram outras componentes na previsão, como tendência, ou mesmo sazonalidade. Em termos de implementação prática, estes modelos não requerem uma etapa prévia de transformação (estabilização da média e/ou variância, normalização, etc.) dos dados, como acontece no caso dos modelos ARMA, ajustando-se diretamente a dados não estacionários.

[37] Ver Brown (1959), Holt (1957) e Winters (1960). Mais recentemente, partindo do trabalho de Holt (1957), Ord (2004) fornece algum contexto em termos de retrospetiva do trabalho original, traçando observações para a prática moderna, em termos de especificação e escolha de métodos.

Segundo a bibliografa (Hyndman & Athanasopoulos, 2018), podemos considerar três tipos de modelos ETS, *Single Exponential Smoothing* (SETS), *Double Exponential Smoothing* (DETS) e *Triple Exponential Smoothing* (TETS). Sucintamente, se o método mais simples assume um comportamento estacionário, o segundo lida explicitamente com existência de uma tendência, enquanto que o terceiro adiciona à tendência a ocorrência de dinâmicas de sazonalidade.

Outra vantagem associada a esta metodologia está na identificação do(s) modelo(s) que melhor se ajusta(m) aos dados. Isto porque, depois de decomposta a séria, uma análise gráfica a cada uma das componentes (tendência e sazonalidade), permite-nos identificar com alguma fiabilidade o(s) modelo(s) mais adequado(s) (Hyndman & Athanasopoulos, 2018).

Façamos, de seguida, uma breve descrição dos três tipos de modelo citados.[38]

2.4.2.1. *SINGLE EXPONENTIAL SMOOTHING*

O modelo mais simples de alisamento exponencial é o SETS, o qual, em termos preditivos, dada uma série temporal $\{y_t\}_{t\in T}$, é descrito pela equação (2.59)

$$\hat{y}_{t+1\,|\,t} = \alpha y_t + (1-\alpha)\hat{y}_{t\,|\,t-1}\,, \quad \forall\, t \in T \tag{2.59}$$

onde $\hat{y}_{t+1\,|\,t}$ denota a previsão para o momento $t+1$ (com base na informação até ao instante $t \in T$) e $0 \leq \alpha \leq 1$ é o parâmetro de alisamento.

Uma representação alternativa deste modelo é a *component form*[39] onde, a única componente a integrar é o nível (ou valor suavizado) da série no instante, ℓ_t. A representação da *component form* do SETS integra uma equação referente à previsão (2.60) e outra referente à suavização do componente incluído no modelo (2.61)

$$\hat{y}_{t+h\,|\,t} = \ell_t\,, \quad \forall\, t \in T \tag{2.60}$$

$$\ell_t = \alpha y_t + (1-\alpha)\ell_{t-1}\,, \quad \forall\, t \in T \tag{2.61}$$

onde h denota o horizonte de previsão.

[38] Para uma revisão da literatura e análise mais detalhada sobre os modelos ETS, veja-se Chatfield *et al.* (2001), Hyndman *et al.* (2008) ou, mais recentemente, Hyndman e Athanasopoulos (2018).

[39] A *component form* do SETS não é particularmente útil, mas simplifica a integração das componentes de tendência (t_t) e sazonalidade (s_t).

No modelo descrito, o coeficiente α controla a taxa com que a influência das observações anteriores decresce exponencialmente. Assim, para valores de α próximo de 1, o modelo 'dá' mais peso a observações mais recentes e torna-se mais sensível à existência, ou não, de oscilações recentes nos dados. Contrariamente, quanto mais próximo de 0 for o α, maior será o peso 'dado' a observações mais antigas, indicando um tratamento mais uniforme, que na prática se traduz (muitas vezes) em previsões mais 'estáveis'.[40]

Observe-se, ainda, que o SETS apresenta uma função de previsão (equação (2.60)) em que o valor predito é igual ao último nível (mantendo-se as previsões constantes ao longo do tempo).

Em suma, o SETS é apropriado para séries que possuem uma trajetória aleatória em torno de uma média constante (logo não é indicado realizar previsões para uma série que apresente tendência e/ou sazonalidade), e apresenta como único hiperparâmetro:

- α – Coeficiente de alisamento para o nível

2.4.2.2. *DOUBLE EXPONENTIAL SMOOTHING*

O DETS é visto como uma extensão do SETS para previsões que apresentem tendência linear (Holt, 1957). Este método recorre a duas constantes de alisamento, α e β, com valores compreendidos entre 0 e 1, que, em termos de *component form*, além da equação de previsão (2.62) envolve duas equações de suavização, uma para o nível (2.63) e outra para a tendência (2.64)

$$\hat{y}_{t+h\,|\,t} = \ell_t + h\,t_t\,, \quad \forall\, t \in T \tag{2.62}$$

$$\ell_t = \alpha y_t + (1-\alpha)(\ell_{t-1} + t_{t-1})\,, \quad \forall\, t \in T \tag{2.63}$$

$$t_t = \beta(\ell_t - \ell_{t-1}) + (1-\beta)\,t_{t-1}\,, \quad \forall\, t \in T \tag{2.64}$$

onde, $\hat{y}_{t+1\,|\,t}$ denota a previsão para o momento $t+1$, h indica o horizonte de previsão, ℓ_t e t_t designam, respetivamente, o nível e a estimativa de tendência da série no instante t, sendo α e β os correspondentes parâmetros de alisamento.[41]

[40] Para mais detalhes sobre o valor da constante de suavização, veja-se Paul (2014).

[41] O DETS pode exibir tendências modeladas de duas formas, modelo aditivo e modelo multiplicativo, dependendo de a tendência ser linear ou exponencial, respetivamente. O DETS com tendência aditiva, que é o caso apresentado, é conhecido classicamente por *Holt's Linear Trend Model*. Não incluiremos nesta apresentação teórica os modelos

No modelo DETS, a previsão com avanço h no tempo, $\hat{y}_{t+h\,|\,t}$, além do nível da série no instante t (ℓ_t), integra h vezes o último valor estimado para a tendência ($\not{t}_t$), como descrito em (2.62). Contudo, para previsões com um horizonte temporal 'grande' a tendência pode estar a ser modelada e a afetar as previsões de modo desajustado, pelo que pode justificar-se o ajuste da tendência ao longo do tempo.

No seguimento do exposto, tal como descrito por Hyndman e Athanasopoulos (2018, Chapter 7.2),

> *... the forecasts generated by Holt's linear method displays a constant trend (increasing or decreasing) indefinitely into the future. Empirical evidence indicates that these methods tend to over-forecast, especially for longer forecast horizons. Motivated by this observation, Gardner & McKenzie (1985) introduced a parameter that "dampens" the trend to a flat line some time in the future. Methods that include a damped trend have proven to be very successful, and are arguably the most popular individual methods when forecasts are required automatically for many series.*

Com efeito, a juntar aos parâmetros de alisamento α e β, o modelo pode integrar um parâmetro de amortecimento, $0 \leq \delta \leq 1$,[42] em (2.62), (2.63) e (2.64), de onde se obtém o novo grupo de equações (2.65), (2.66) e (2.67), respetivamente

$$\hat{y}_{t+h\,|\,t} = \ell_t + (\delta + \delta^2 + \cdots + \delta^h)\,\not{t}_t\,, \quad \forall\, t \in T \tag{2.65}$$

$$\ell_t = \alpha y_t + (1-\alpha)(\ell_{t-1} + \delta\,\not{t}_{t-1})\,, \quad \forall\, t \in T \tag{2.66}$$

$$\not{t}_t = \beta(\ell_t - \ell_{t-1}) + (1-\beta)\,\delta\,\not{t}_{t-1}\,, \quad \forall\, t \in T \tag{2.67}$$

multiplicativos, pois estes não serão objeto de implementação computacional, por tenderem a apresentar maus resultados preditivos (Hyndman & Athanasopoulos, 2018).

[42] Na prática, querendo fazer uma distinção com o caso anterior (em que $\delta = 1$), considera-se δ entre 0,8 e 0,98. Para valores de δ próximos de 1, o modelo amortecido não se diferencia do modelo não amortecido, enquanto que para valores inferiores a 0,8 o amortecimento tem um efeito forte.

Em suma, o modelo DETS (com amortecimento) é apropriado para séries que possuem uma trajetória com tendência, considerando-se como hiperparâmetros:

- α – Coeficiente de alisamento para o nível
- β – Coeficiente de alisamento para a tendência
- δ – Coeficiente de amortecimento
- Tipo de amortecimento: aditivo ou multiplicativo

2.4.2.3. *TRIPLE EXPONENTIAL SMOOTHING*

Holt (1957) e Winters (1960) estenderam o DETS ao TETS[43], o qual é utilizado quando as séries temporais, além de poderem apresentar padrões com tendência linear, exibem sazonalidade. Deste modo, são aplicadas suavizações para estimar o nível, tendência e sazonalidade da série em estudo no processo de previsão, pelo que o modelo irá incluir um novo parâmetro que controla a influência da componente sazonal (s_t).

O TETS apresenta duas abordagens distintas consoante o modo como a sazonalidade é modelada: a forma multiplicativa para uma variação exponencial da sazonalidade e a forma aditiva para uma variação linear.

Em termos de adequabilidade, tal como descrito em Hyndman e Athanasopoulos (2018), o método aditivo é o mais apropriado para séries cuja amplitude de sazonalidade é independente do nível (variações sazonais aproximadamente constantes), enquanto o método multiplicativo é mais indicado sempre que a amplitude da sazonalidade varie com o nível (variações sazonais proporcionais ao nível da série).

Em qualquer dos casos (aditivo/multiplicativo), a *component form* compreende a equação da previsão (2.68)/(2.72) e três equações de suavização: uma para o nível (2.69)/(2.73), uma para a tendência (2.70)/(2.74) e outra para o componente sazonal (2.71)/(2.75), com os parâmetros de suavização (com valores entre 0 e 1, inclusive) correspondentes, α, β e γ, e usamos m para indicar a frequência da sazonalidade (número de partes do ano, por exemplo, $m = 4$ para dados trimestrais e $m = 12$ para dados mensais).

No caso do método aditivo, tem-se

[43] Modelo comummente identificado na literatura como o *Holt-Winters Exponential Smoothing Model*.

$$\hat{y}_{t+h\,|\,t} = \ell_t + h\,t_t + s_{t+h-m(k+1)}, \quad \forall\, t \in T \tag{2.68}$$

$$\ell_t = \alpha(y_t - s_{t-m}) + (1-\alpha)(\ell_{t-1} + t_{t-1}), \quad \forall\, t \in T \tag{2.69}$$

$$t_t = \beta(\ell_t - \ell_{t-1}) + (1-\beta)t_{t-1}, \quad \forall\, t \in T \tag{2.70}$$

$$s_t = \gamma(y_t - \ell_t) + (1-\gamma)s_{t-m}, \quad \forall\, t \in T \tag{2.71}$$

No caso do método multiplicativo, tem-se

$$\hat{y}_{t+h\,|\,t} = (\ell_t + h\,t_t) \times s_{t+h-m(k+1)}, \quad \forall\, t \in T \tag{2.72}$$

$$\ell_t = \alpha\left(\frac{y_t}{s_{t-m}}\right) + (1-\alpha)(\ell_{t-1} + t_{t-1}), \quad \forall\, t \in T \tag{2.73}$$

$$t_t = \beta(\ell_t - \ell_{t-1}) + (1-\beta)\,t_{t-1}, \quad \forall\, t \in T \tag{2.74}$$

$$s_t = \gamma\left(\frac{y_t}{\ell_t}\right) + (1-\gamma)\,s_{t-m}, \quad \forall\, t \in T \tag{2.75}$$

Em qualquer dos casos, além de parâmetros já caraterizados nos casos anteriores, ou acima identificados, temos:

- k é a parte inteira de $(h-1)/m$, o que garante que as estimativas dos índices de sazonalidade usados para previsão são provenientes do último ano da amostra;
- a equação do nível traduz uma média ponderada entre a observação ajustada de sazonalidade $(y_t - s_{t-m})$ e a previsão não sazonal $(\ell_{t-1} + t_{t-1})$ no instante t.

À semelhança do modelo DETS, o TETS pode igualmente integrar amortecimentos em termos de tendência, pelo que, no caso do método multiplicativo[44], as equações anteriores podem ser reescritas integrando o parâmetro δ

[44] Método que, segundo Hyndman e Athanasopoulos (2018), fornece previsões precisas e robustas para dados sazonais.

$$\hat{y}_{t+h\,|\,t} = [\,\ell_t + (\delta + \delta^2 + \cdots + \delta^h)t_t\,] \times s_{t+h-m(k+1)}, \quad \forall\, t \in T \tag{2.76}$$

$$\ell_t = \alpha\left(\frac{y_t}{s_{t-m}}\right) + (1-\alpha)(\ell_{t-1} + \delta\, t_{t-1}), \quad \forall\, t \in T \tag{2.77}$$

$$t_t = \beta(\ell_t - \ell_{t-1}) + (1-\beta)\,\delta\, t_{t-1}, \quad \forall\, t \in T \tag{2.78}$$

$$s_t = \gamma\left(\frac{y_t}{\ell_t}\right) + (1-\gamma)\, s_{t-m}, \quad \forall\, t \in T \tag{2.79}$$

Em suma, o TETS (com amortecimento) é apropriado para séries que possuem uma trajetória com tendência e sazonalidade, apresentando como hiperparâmetros:

- α – Coeficiente de alisamento para o nível
- β – Coeficiente de alisamento para a tendência
- γ – Coeficiente de alisamento para a sazonalidade
- δ – Coeficiente de amortecimento
- m – Frequência da sazonalidade
- Tipo de sazonalidade: aditiva ou multiplicativa
- Tipo de amortecimento: aditivo ou multiplicativo

2.5. Critérios de Comparação/Seleção de Modelos

No seguimento do referido anteriormente (em particular na Sec. 2.2), uma análise gráfica ao correlograma da série pode ser um bom ponto de partida para identificar qual a ordem, (p, d, q) do modelo $ARMA$ que melhor ajusta os dados. Porém, trata-se de uma seleção *ad hoc*, pelo que se torna necessário recorrer a estatísticas/critérios que nos permitam fundamentar as escolhas.

Diversos critérios têm sido propostos na literatura para a seleção do tipo/ordem dos modelos, tendo como objetivo equilibrar o risco da seleção. Por um lado, a escolha de uma ordem inferior à ajustada ao real, para evitar inconsistências na estimação dos parâmetros. Por outro lado, a escolha de uma ordem superior conduz ao incremento da variância dos estimadores e do número de hiperparâmetros a considerar. O equilíbrio é alcançado por atribuição de um custo/penalização pela exclusão/introdução de variáveis adicionais.

Na prática, os critérios baseados na otimização da função de verosimilhança são os mais utilizados, sobretudo o *Likelihood Ratio* (LR), o *Akaike Information Criterion* (AIC) e o *Bayesian Information Criterion* (BIC), aos quais iremos acrescentar o *Hannan–Quinn Information Criterion* (HQIC). Se o LR é apropriado para testar dois modelos (desde que um deles seja um caso particular do outro), qualquer dos restantes critérios comparam o ajuste dentro da amostra, medido pela variância dos resíduos, contra o número de parâmetros estimados.

Analisemos, então, cada um dos critérios acima referidos.[45]

2.5.1. *LIKELIHOOD RATIO*

No que respeita à regressão, a ideia é comparar os valores observados da variável dependente com as previsões obtidas dos modelos, com e sem uma dada variável. No caso da regressão linear, o interesse é saber o valor da soma dos quadrados dos resíduos. Um valor reduzido desta soma sugere que a variável independente é importante; caso contrário, esta não é útil na previsão da variável dependente. No caso da regressão logística, a comparação dos valores observados com os valores preditos é baseada no logaritmo da verosimilhança, requerendo o LR a estimação do modelo restrito (sem inclusão da variável no modelo) e não restrito (com a inclusão da variável no modelo).

Com efeito, denotando por θ_s e por θ_c os vectores de parâmetros com e sem inclusão da variável no modelo, respetivamente, podemos calcular o valor maximizado da função de máxima verosimilhança sem e com a variável, denotado respetivamente por $\mathcal{L}_{max}(\theta_s)$ e $\mathcal{L}_{max}(\theta_c)$.

Formalmente, estando o teste LR baseado no logaritmo da razão entre as duas verosimilhanças, isto é, na diferença entre o $log\ [\mathcal{L}_{max}(\theta_s)]$ e $log\ [\mathcal{L}_{max}(\theta_c)]$, temos

$$\mathcal{S}_{LR} = -2\ log[\mathcal{L}_{max}(\theta_s)] - 2\ log\ [\mathcal{L}_{max}(\theta_c)] \qquad (2.80)$$

Assim, para testar a ausência de relação, se o valor da estatística for superior a um valor crítico ao nível de significância desejado, rejeitamos a hipótese nula de exclusão no modelo.

[45] Para mais detalhes, veja-se Sin e White (1996) ou Claeskens e Hjort (2008).

2.5.2. *AKAIKE INFORMATION CRITERION*

A estatística subjacente ao AIC, desenvolvido por Akaike (1974), permite fazer uma seleção do(s) modelo(s) que melhor se ajusta(m) aos dados com base na qualidade de ajuste dos modelos estimados.

Contrariamente a outros critérios, a estatística AIC não fornece um teste ao modelo (no sentido usual de testar uma hipótese nula) que traduza a sua adequabilidade aos dados (em sentido absoluto). O valor da estatística referente ao AIC incorpora duas componentes, uma que valoriza a precisão do ajuste e outra que penaliza, por meio de uma função crescente, os modelos com maior número de parâmetros. Deste modo, a variação nos valores da estatística de AIC, relativa a cada um dos modelos em análise, deve-se ao logaritmo das razões de verosimilhança entre os modelos, penalizando-se os modelos pelo número de parâmetros.

Em sentido lato, sendo k o número de parâmetros do modelo e $\mathcal{L}$ o valor da função de máxima verosimilhança correspondente, o valor da estatística de AIC é dado por

$$\mathcal{S}_{AIC} = -2\,log(\mathcal{L}) + 2k \tag{2.81}$$

Em suma, o AIC avalia a distância relativa do modelo proposto ao modelo verdadeiro ou, por outras palavras, a discrepância no ajuste do modelo proposto aos dados. Com efeito, o melhor modelo é aquele que apresenta um menor valor de $\mathcal{S}_{\mathrm{AIC}}$.

2.5.3. *BAYESIAN INFORMATION CRITERION*

O BIC, por vezes também referido como o *Schwarz Information Criterion*, foi proposto por Schwarz (1978). Este critério baseia-se, em parte, na função de verosimilhança, estando relacionado com o AIC, mas incorporando um formalismo bayesiano. Se ambos os critérios permitem a comparação entre modelos, penalizando os modelos com maior número de parâmetros, no modelo bayesiano de Schwarz, essa penalização é mais rigorosa.

Basicamente, a ideia associada é partir-se de um modelo de regressão com vários *lags* e, gradualmente, ir reduzindo o número de *lags* até se encontrar aquele que minimize o valor da estatística associada ao BIC, formalmente é descrita por

$$\mathcal{S}_{BIC} = -2\,log(\mathcal{L}) + k\,log(n) \tag{2.82}$$

onde k é o número de parâmetros do modelo, n a dimensão da amostra e $\mathcal{L}$ o valor da função de máxima verosimilhança correspondente.

Observe-se que o primeiro termo da estatística do BIC avalia o ajuste do modelo e o segundo corresponde à penalização pelo aumento do número de parâmetros. Tal como no caso da estatística do AIC, quanto menor for $\mathcal{S}_{\text{BIC}}$ melhor será o modelo.

2.5.4. *HANNAN–QUINN INFORMATION CRITERION*

Alternativamente aos critérios AIC e BIC, mas com o mesmo objetivo de comparar o ajuste de modelos, Hannan & Quinn (1979) propuseram o HQIC. Segundo os autores, trata-se de uma estimativa mais consistente, pois, comparada com outros procedimentos, esta subestimará a ordem em 'diferente' grau, por não penalizar tanto como o BIC os modelos com maior número de parâmetros, mas ser mais rigorosa que o AIC.

Formalmente, sendo k o número de parâmetros do modelo, n a dimensão da amostra e $\mathcal{L}$ o valor da função de máxima verosimilhança, o valor da estatística associada ao HQIC é dado por

$$\mathcal{S}_{HQIC} = -2\,log(\mathcal{L}) + 2k\,log\,[log(n)] \tag{2.83}$$

Á semelhança do AIC e do BIC, quanto menor for o valor de $\mathcal{S}_{\text{HQIC}}$, melhor se espera que seja o modelo (no sentido de ajustamento aos dados reais).

2.5.5. COMPARAÇÃO DOS CRITÉRIOS

Face às considerações feitas em cada caso, podemos concluir que todos os critérios de informação apresentados se baseiam numa otimização de função de máxima verosimilhança. A diferença fundamental entre eles reside na forma de avaliar os modelos, em particular na penalização do número de parâmetros, conduzindo a valores de estatísticas distintos.

Com o LR, considera-se por hipótese que o modelo mais simples é o de melhor ajuste, até que, dado um nível de significância α, se observem diferenças estatísticas para um modelo mais complexo. Esta avaliação parece ser a menos usada em séries temporais.[46]

[46] Razão pela qual o LR não será implementado no nosso trabalho.

Por seu lado, utilizando-se o AIC admite-se que, do conjunto de modelos avaliados, nenhum é considerado o 'modelo verdadeiro', isto é, o que realmente descreve a relação entre a variável dependente e as variáveis explicativas, pelo que se tenta escolher aquele que minimize as divergências. Já no BIC, está implícito que existe um modelo que descreve a relação entre as variáveis envolvidas (um 'modelo verdadeiro') e o critério tenta maximizar a probabilidade de o escolher, penalizando essa escolha à medida que são introduzidos mais parâmetros. Seguindo a mesma linha de penalização, mas não tão 'dura', temos o HQIC que, embora bastante citado na literatura, parece ser pouco usado (ou menos usado) na prática (Burnham & Anderson, 2002).

Em suma, aquando da etapa fundamental de seleção do(s) melhor(es) modelo(s), importa olhar para a informação que resulta da aplicação de cada um dos três critérios (AIC, BIC e HQIC). Quando comparados os três critérios, a estatística $\mathcal{S}_{\mathrm{BIC}}$ tende a ser a mais parcimoniosa. Daí que, segundo a literatura, o *"… BIC has been widely used for model identification in time series (…) [and] be applied quite widely to any set of maximum likelihood-based models…*" (Clement, 2014, p. 216).

2.6. DIAGNÓSTICO DOS MODELOS

Embora o nosso foco de interesse seja efetuar previsões com os modelos selecionados, antecedendo essa etapa convém validá-los por análise da sua componente residual. Caso o ajuste do modelo seja adequado, os resíduos (como resultado da diferença entre o valor observado e o valor predito pelo modelo) devem ser aproximados por um processo de ruído branco.

Assim, em ternos de modelos ARIMA, supondo que o modelo se ajusta como descrito em (2.57), podemos descrever os resíduos do modelo por

$$\varepsilon_t = \theta_q^{-1}(L)\phi_p(L)(w_t - \mu) \tag{2.84}$$

Garantindo os polinómios ϕ_p e θ_q a estacionariedade e a invertibilidade, respetivamente, caso o modelo seja adequado, os resíduos devem ser iid com $\varepsilon_t \sim WN(0, \sigma_\varepsilon^2)$.

Com efeito, três aspetos a considerar na análise sobre os resíduos respeitam a: (i) média nula (com verificação de normalidade mediante o teste de *Jarque-Bera*, por exemplo); (ii) ausência de autocorrelação (mediante o teste (LM) *Breusch-Godfrey* e/ou valores das estatísticas de *Box-Pierce* ou *Ljung-Box*, por exemplo); e (iii) homoscedasticidade, por exemplo

mediante o teste (LM) *ARCH (Autoregressive Conditional Heteroskedasticity)*.[47] De acordo com a literatura, na validação de modelos de séries temporais,

> *… os três aspetos cruciais dos resíduos são as suas autocorrelações, as suas autocorrelações parciais e os valores da estatística de Box-Pierce-Ljung, que testa a significância conjunta de subconjuntos de coeficientes de autocorrelação (…). Se os resíduos se afastarem significativamente [tendo em mente que se trata de um cenário teórico ideal] de um processo de ruído branco, o modelo não é satisfatório e tem de ser reespecificado…".* (Johnston & DiNardo, 2001, p. 256)

Observe-se que, por mais ajustado que seja o modelo, as previsões irão conter um erro associado dada a diferença entre o valor real e o respetivo valor predito. Deste modo, a análise à componente residual torna-se bastante informativa, na medida em que os resíduos de um modelo permitem compreender os motivos pelos quais este não consegue prever, completamente, as flutuações futuras da série de dados (por conterem toda a informação necessária). Tal como referido por Johnston e DiNardo (2001) essa incapacidade decorre, essencialmente, de dois motivos: (1) a presença de aleatoriedade nos dados da série temporal, o que conduz a um erro motivado pelo desconhecimento de inovações futuras e/ou; (2) a má especificação do modelo, isto é, a falta de ajuste devido a diferenças entre os parâmetros reais e estimados.

Precisamente, com o intuito de não incorrer no segundo caso, a análise à componente residual do modelo torna-se fundamental para que o modelo estimado possa ser considerado um 'modelo adequado' por se ajustar aos dados históricos. Esta adequabilidade será preponderante para a qualidade preditiva do modelo.

A não validação dos requisitos referidos, conduz a uma má especificação do modelo, onde, em particular, a presença de autocorrelação nos resíduos é um dos problemas mais 'sérios', pois significa dependência temporal dos valores sucessivos dos resíduos.[48]

[47] Para alguns detalhes sobre estes testes, veja-se o Anexo B.3.

[48] Das três propriedades dos estimadores (não enviesamento, consistência e eficiência), a que fica comprometida pela presença de autocorrelação nos erros é a eficiência, isto é, de entre os estimadores consistentes (não enviesados), podem existir estimadores para os parâmetros do modelo que apresentem menor variância.

3. REDES NEURONAIS ARTIFICIAIS

No presente capítulo, procurar-se-á fazer uma descrição e análise de aspetos, quer de cariz teórico, quer em termos metodológicos, intrínsecos ao estudo de redes neuronais. Assim, será objetivo apresentar uma visão geral sobre as redes neuronais, desde um enquadramento histórico, passando pela análise de alguns aspetos teóricos e exploração de algumas arquiteturas atuais de redes que servirão de base de trabalho na parte do estudo empírico desenvolvido.

3.1. DA INTELIGÊNCIA ARTIFICIAL ÀS *DEEP NEURAL NETWORKS*

A interação "homem *vs* máquina" tem sido elemento transformador da sociedade à qual parte da comunidade científica tem dedicado particular interesse. A crescente frequência com que expressões como "Inteligência Artificial" e "*Machine Learning*" são usadas, em cada vez mais áreas do conhecimento, parece ser prova disso.

O interesse pelos domínios da Inteligência Artificial (IA), muito em particular pelo *Machine Learning* tem-se mostrado tão evidente que, cada vez mais, investigadores, profissionais, e empresas procuram compreender o que é e como usar esta aprendizagem feita pela máquina. Aliás, alguns autores destacam esta revolução trazida à aprendizagem e acrescentam que não se pode ficar indiferente à sua introdução nos mais diversos tópicos do conhecimento com a introdução de funcionalidades de *Machine Learning* (Data Science Academy, 2019).

De acordo com Sage (1990) a IA consiste no desenvolvimento de paradigmas ou algoritmos que requerem o recurso a máquinas para a realização de tarefas cognitivas para as quais os humanos são particularmente melhores. Um sistema de IA, integrando componentes de representação, raciocínio e aprendizagem, deve ser capaz de cumulativamente: (i) armazenar conhecimento; (ii) aplicar o conhecimento armazenado para resolver problemas e (iii) adquirir novo conhecimento através da experiência.

Nos domínios da IA, a Data Science Academy (2019) define o *Machine Learning* como a utilização de algoritmos para extrair informações de dados brutos e representá-los através de algum modelo matemático, o qual poderá ser utilizado para fazer inferências a partir de outros

conjuntos de dados.[49] Da multiplicidade de algoritmos existentes, o recurso a uma Rede Neuronal Artificial (RNA) tem sido um dos tópicos ao qual a comunidade científica tem dado particular interesse.

Muito embora as RNAs não sejam um tópico recente na literatura (existem referências desde a década de 1950), o facto é que, beneficiando de grandes evoluções computacionais, o tema tem reaparecido recorrentemente com uma força notável e os mecanismos de base intrínsecos têm sido objeto de consideráveis desenvolvimentos, em particular no que respeita a processos de aprendizagem profunda – *Deep Learning*.

Nos últimos anos tem-se notado uma adoção em massa do *Deep Learning*, onde vários eventos ligados a *Data Science*, IA e *Big Data*, apontam o *Deep Learning* como a principal tecnologia para criação de sistemas inteligentes. Na Figura 3.1 é esquematizada a interligação dos conceitos em causa.

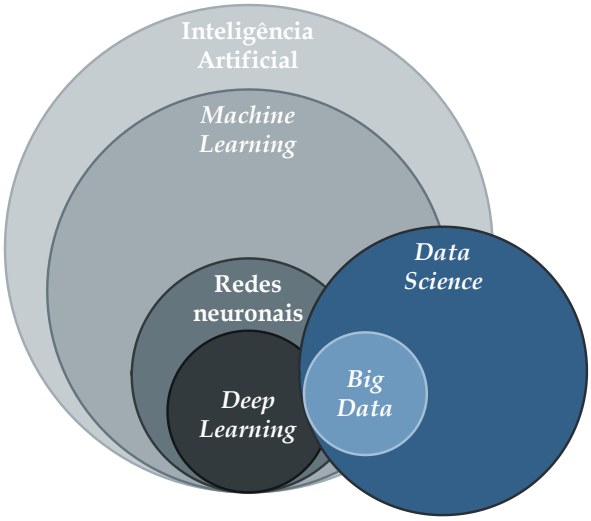


FIGURA 3.1 – *Data Science* e Inteligência Artificial: interligação de conceitos

Citando a Data Science Academy (2019, Chapter 3),

> *… Atualmente, o processamento de Big Data e a evolução da Inteligência Artificial são ambos dependentes da Aprendizagem Profunda. Com Deep Learning*

[49] Citando a Data Science Academy (2019, Chapter 1), perante um volume considerável de informação gerado em variedade e velocidade crescentes (*Big Data*), as unidades de processamento gráfico, permitindo (i) realizar operações matemáticas de forma paralela (Processamento Paralelo) e (ii) criar modelos computacionais sofisticados e com altos níveis de precisão (*Machine Learning*), formaram a tempestade perfeita que conduziu à evolução na qual nos encontramos hoje:

> *"Big Data + Processamento Paralelo + Modelos de Machine Learning = Inteligência Artificial"*

> *podemos construir sistemas inteligentes e estamo-nos aproximando da criação de uma IA totalmente autônoma. Isso vai gerar impacto em todas os segmentos da sociedade e aqueles que souberem trabalhar com a tecnologia, serão os líderes desse novo mundo que se apresenta diante de nós.*

Se nos centrarmos nos domínios de *Machine Learning*, em específico no processo de aprendizagem profundo da rede neuronal – *Deep Neural Networks* (DNN) – muito existe em comum com a mente humana.

O neurónio matemático é considerado a unidade fundamental de uma RNA. Este baseia-se no neurónio biológico, dado o paralelo que existe entre o que ocorre na mente humana e numa RNA (como descrito mais adiante na Sec. 3.3).

Na mente humana, não só é processada aprendizagem momentânea como se espera uma 'persistência' da mesma ao longo do tempo. Isto é, em cada momento ('época') da vida não começamos a aprender do zero, as nossas recordações têm um papel fundamental no processo de aprendizagem (memórias passadas ajudam-nos a compreender novas informações). Este processo é feito num ciclo contínuo de treino e aprendizagem, pesando benefícios e erros passados. Na verdade, a nossa mente não tende a armazenar toda a informação, mas vai guardando aquilo que, em algum momento, foi considerado relevante, em detrimento de 'detalhes' não considerados.

Se tivermos em conta o descrito, muito encontramos em comum com a ideia que existe por detrás das arquiteturas de redes neuronais, quer no processo de treino e aprendizagem (tema a tratar na Sec. 3.4), quer nos "mecanismos" que operam internamente e que nos permitem fazer a distinção entre algumas categorias/arquiteturas de redes (aspetos tratados na Sec. 3.5).

Atendendo à evolução reportada na literatura (retrospetiva traçada na Sec. 3.2), o aparecimento de arquiteturas alternativas é, em parte, motivado por algumas limitações que foram sendo apontadas a arquiteturas anteriores. Precisamente, a procura de respostas a limitações que vão sendo levantadas está na base de um contínuo desafio para o *Machine Learning* de RNA. Parte dos algoritmos/metodologias desenvolvidas seguem uma linha de *Deep Learning* que, na procura de respostas para os problemas, tem obrigado à reestruturação de teorias e ao desenvolvimento de mecanismos computacionais cada vez mais eficientes e alinhados com a criação de uma "inteligência artificial autónoma".

É nesta linha de permanente desafio e desenvolvimento que enquadramos as DNN, tema a desenvolver neste capítulo.

3.2. BREVE ENQUADRAMENTO HISTÓRICO

De acordo com a literatura científica, terá sido na década de 40 que surgiram os primeiros trabalhos sobre redes neuronais que, sucintamente, se caracterizaram por apresentarem um modelo matemático baseado nos neurónios biológicos, conjugando, assim, as áreas da neurofisiologia e da matemática (lógica). McCulloch e Pitts (1943 e 1947) terão sido os primeiros a apresentar trabalho de desenvolvimento computacional baseado no comportamento do cérebro humano, explorando paradigmas das redes neuronais para o reconhecimento de padrões utilizando neurónios artificiais e apresentando o neurónio booleano de McCulloch, como ficaria conhecido na literatura. Ao trabalho desenvolvido pelos autores atribui-se o nascimento das disciplinas de redes neuronais e inteligência artificial.

No final da década de 40, duas publicações importantes poderão ser consideradas como um marco no que respeita ao estudo de redes neuronais: o livro de Wiener (1948) e o de Hebb (1949), publicado um ano mais tarde. Ambas as obras têm sido consideradas uma fonte de inspiração para o desenvolvimento de modelos computacionais de sistemas adaptativos e de aprendizagem.

A década de 50 pouco trouxe de novo em termos de pesquisas de redes neuronais. Além dos estudos iniciados por Minsky (cujos desenvolvimentos seriam mais tarde publicados), no final da década e no seguimento dos citados trabalhos de McCulloch e Pitts (1943 e 1947), Rosenblatt (1958) propõe o *Perceptron* (neurónio artificial) para produzir a primeira RNA, utilizada para classificar padrões linearmente separáveis.

A década de 60 revelou-se mais ativa: Widrow e Hoff (*apud* Widrow, 1962) conceberam o algoritmo *least mean-square*, utilizando-o para formular o *Adaline* (*Adaptive Linear Element*), algoritmo reformulado por Widrow (1962) ao apresentar o *Madaline* (*multiple-adaline*) como uma das primeiras redes neuronais em camadas e treinada com múltiplos elementos adaptativos. Já no final da década, Minsky e Papert (1969) apresentam e discutem as limitações ao *Perceptron*. Como resultado destas limitações surgiu algum desinteresse, por parte dos investigadores, pelo estudo das redes neuronais artificiais, entrando-se num período que a literatura identifica como o "Inverno da IA".

Na década de 80, a ocorrência de vários eventos acabaria por potenciar um ressurgir de interesse pelo tema. Neste contexto, surgem várias publicações logo no início da década: Hopfield (1982) apresentou um modelo não linear com aprendizagem não supervisionada e Kohonen (1982) sugere um algoritmo de aprendizagem com base em mapas auto-organizáveis (utilizando uma estrutura de rede unidimensional ou bidimensional), conhecido na literatura

como *self-organization feature maps*. Esta década viria a ser ainda marcante pela apresentação de soluções para parte dos problemas levantados por Minsky e Papert (1969), com o desenvolvimento de algumas arquiteturas de redes neuronais, como as baseadas no modelo *multilayer perceptron*. Este utilizava um algoritmo com aprendizagem supervisionada, atualmente conhecido na literatura como *Backpropagation*. Foram exemplos disso os trabalhos apresentados por Rummelhart *et al.* (1986) e McClelland *et al.* (1986), por conseguirem resolver parte dos problemas de aprendizagem existentes. Segundo a literatura, estes trabalhos viriam a exercer uma influência notável na utilização da aprendizagem segundo o algoritmo *Backpropagation* (Haykin, 2009).

Se na década de 80 e início da década de 90 foram feitos avanços importantes na arquitetura das RNA (nomeadamente com o algoritmo *Backpropagation* e novos desenvolvimentos em torno das redes neuronais recorrentes), os anos seguintes ficariam marcados por algum desinteresse. O tempo necessário para obter bons resultados colocou novamente em questão a capacidade das máquinas e o poder computacional, causando um novo período de "arrefecimento" (conhecido como um segundo Inverno da IA) em termos de pesquisa em RNA e *Machine Learning*.

Outras questões levantadas na primeira metade da década de 90, como é exemplo o trabalho de Bengio, Simard, & Frasconi (1994), viriam a expor limitações das redes recorrentes (usadas em problemas de reconhecimento, produção ou previsão) na execução de tarefas em que as operações temporais presentes nas sequências de entrada/saída abrangem longos intervalos. Isto porque algoritmos de aprendizagem baseados em gradiente enfrentam maiores problemas com o aumento de dependências (temporais) a serem capturadas, por não ser retida informação por longos períodos (problema identificado na literatura como *vanishing gradient*).[50]

Contudo, em virtude do trabalho feito nos domínios da IA, em particular em metodologias de *Deep Learning*, surgem novos desenvolvimentos já na segunda metade da década de 90: Cortes e Vapnik (1995) desenvolveram o *Support Vector Network* (um sistema para mapear e reconhecer dados semelhantes) e Hochreiter e Schmidhuber (1997) apresentaram uma variação de redes recorrentes, com as chamadas unidades de *Long Short-Term Memory* (LSTM), como uma solução para o problema do *vanishing gradient*.

[50] Sucintamente, foi descoberto que as características aprendidas em treinos bastante anteriores não eram aprendidas em treinos recentes. A informação era apenas retida na aprendizagem recente, ou de "curto prazo", perdendo-se a informação de longo prazo. Este não era um problema fundamental para todas as redes neurais, apenas aquelas com métodos de aprendizagem baseados em gradientes (Data Science Academy, 2019).

O próximo passo significativo para o *Deep Learning* ocorreu em finais de 1999 inícios de 2000, quando os computadores começaram a tornar-se mais rápidos no processamento de dados e foram desenvolvidas GPUs (unidades de processamento de gráfico). Nesta fase,

> *… o poder computacional expandiu exponencialmente e o mercado viu uma "explosão" de técnicas computacionais que não eram possíveis antes disso. Foi quando o Deep Learning emergiu (…) como o principal mecanismo de construção de sistemas de Inteligência Artificial, ganhando muitas competições importantes de aprendizagem de máquina …* (Data Science Academy, 2019, Chapter 1)

Com efeito, tirando partido do gradual e notável progresso tecnológico/computacional, sobretudo pelo notável aumento da 'capacidade' das máquinas, várias arquiteturas de redes neuronais baseadas em mecanismos de aprendizagem profunda (DNN) têm sido propostas na literatura nos últimos anos, das quais se destacam as popularizadas redes LSTM.[51]

Com estas considerações, chegámos ao tópico atual e de particular interesse no nosso estudo, o qual será objeto de análise na Sec. 3.5, pelo que nos escusamos a mais detalhes neste enquadramento histórico e bibliográfico.

3.3. Do Neurónio ao Modelo de Rede Neuronal

Nesta secção será dada ênfase aos dois principais elementos que compõem uma rede neuronal: os neurónios (*perceptrons*) e as camadas (*layers*). Além de uma introdução aos conceitos, procurar-se-á fazer referência às notações a utilizar nas secções seguintes.

3.3.1. Neurónio

A unidade fundamental das redes neuronais artificiais é o neurónio artificial (*perceptron*), a qual simula um neurónio humano. Na Figura 3.2 temos uma representação simplificada do neurónio biológico e a respetiva associação com o funcionamento do neurónio artificial.

[51] Alguns detalhes interessantes, com referência a vários trabalhos, podem ainda ser consultados na publicação online *The 2010s: Our Decade of Deep Learning* disponível em http://people.idsia.ch/~juergen/2010s-our-decade-of-deep-learning.html.

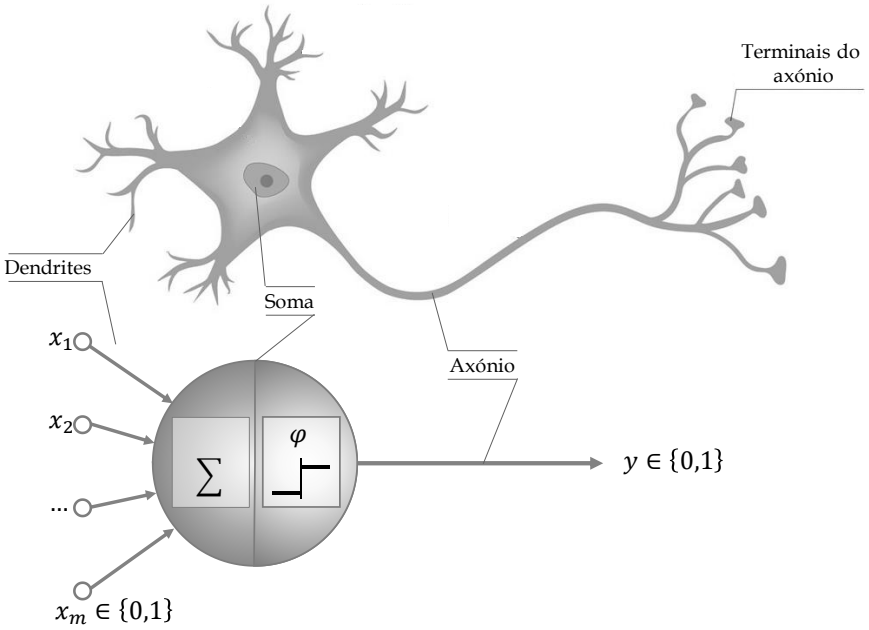


FIGURA 3.2 – Associação entre o neurónio biológico e o neurónio artificial

De forma simplificada, um neurónio biológico recebe um conjunto de sinais de entrada (dendrites), processa-o numa célula semelhante ao processador de um computador (soma) e, se a combinação de todos os sinais de entrada exceder um dado valor, produz um sinal de saída que é transmitido a outros neurónios, ligados através de um sistema semelhante a cabos (axónio, com os respetivos terminais).

Similarmente, de acordo com o primeiro modelo computacional (McCulloch & Pitts, 1943), o neurónio artificial recebe um conjunto de m sinais de entrada, $\boldsymbol{X} = (x_1, x_2, \dots x_m)$ com $x_i \in \{0, 1\}$, agrupa-o (função de transferência ou pré-ativação, Σ) e, com base numa função de ativação φ, toma uma decisão que será ou não transmitida a outros neurónios.

Formalmente, o processo pode ser descrito pelas equações

$$\Sigma(\boldsymbol{X}) = \Sigma(x_1, x_2, \dots x_m) = \sum_{i=1}^{m} x_i \tag{3.1}$$

$$y = \varphi(\boldsymbol{X}) = \begin{cases} 1\,, & se\ \ \Sigma(\boldsymbol{X}) \geq \vartheta \\ 0\,, & se\ \ \Sigma(\boldsymbol{X}) < \vartheta \end{cases} \tag{3.2}$$

onde ϑ é o parâmetro de decisão.

Este modelo permitia apenas representar funções booleanas linearmente independentes (utilizando valores de entrada e saída binários), atribuía a mesma importância relativa a todos os sinais de entrada e requeria sempre a definição do parâmetro de decisão.

Estas limitações levaram ao desenvolvimento daquele que ainda hoje é utilizado como modelo computacional de neurónio artificial (Rosenblatt, 1958 e Minsky & Papert, 1969). Com efeito, a atribuição de pesos (importâncias relativas, que no neurónio biológico são atribuídas pelas sinapses) aos valores de entrada e fornecendo uma forma de os aprender, levantou a restrição do modelo a funções booleanas (permitindo trabalhar com dados reais).

Genericamente, podemos descrever um *perceptron* k, p_k, pela soma dos seus m *inputs*, $\boldsymbol{X} = (x_1, x_2, \dots x_m)$, cada um deles ponderado por um peso (que carateriza a respetiva sinapse), w_{kj} com $j = 1, \dots, m$, e cujo *output*, y_k, é definido tendo em conta uma função de ativação, φ, e um *bias* aplicado externamente, b_k,[52] conforme ilustrado na Figura 3.3.

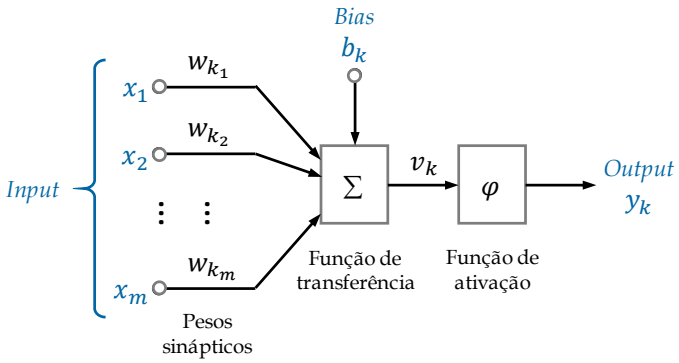


FIGURA 3.3 – Modelo não linear de um *perceptron* (1)

Em termos matemáticos, podemos escrever

$$p_k = \sum_{j=1}^{m} w_{kj} x_j = \boldsymbol{W}^T \boldsymbol{X} \tag{3.3}$$

$$y_k = \varphi(p_k + b_k) \tag{3.4}$$

onde, além dos elementos já apresentados, $\boldsymbol{W} = (w_{k1}, w_{k2}, \dots w_{km})$ representa o vetor de pesos sinápticos associados ao *perceptron* k.[53]

[52] Em termos práticos, o valor do *bias*, b_k, serve para aumentar o grau de liberdade dos ajustes dos pesos o que permite uma melhor adaptação da rede neuronal ao conhecimento por ela fornecido.

[53] O vetor $\boldsymbol{W}$ é também chamado vetor de parametrização, sendo caraterístico de cada neurónio. Tendo em conta um determinado algoritmo de treino (tema a tratar na Sec. 3.4), os valores de $\boldsymbol{W}$ são treinados de modo a minimizar o erro associado a $(y_{desejado} - y_k)$. Visto que o vetor $\boldsymbol{X}$ é dado, apenas são ajustados os pesos de cada um dos *inputs* (e também o *bias*).

O *bias*, b_k, assume-se como um parâmetro externo, pelo que, conforme representado na Figura 3.3., o potencial de ativação é dado por

$$v_k = p_k + b_k \tag{3.5}$$

ou ainda, combinado com as equações (3.3) e (3.4), podemos escrever

$$v_k = \sum_{j=0}^{m} w_{kj} x_j \tag{3.6}$$

$$y_k = \varphi(v_k) \tag{3.7}$$

em que na equação (3.6) foi adicionado uma nova sinapse, cuja entrada é $x_0 = 1$, com peso correspondente ao *bias*, isto é $w_{k0} = b_k$. Assim, podemos esquematizar o modelo do *perceptron* como apresentado na Figura 3.4.

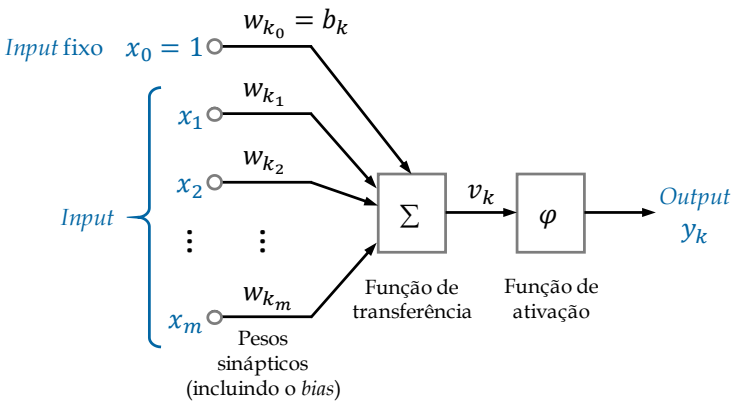


FIGURA 3.4 – Modelo não linear de um *perceptron* (2)

Outra característica de cada neurónio, que permite ajustar o resultado da soma dos *inputs* ponderados pelo respetivo peso, é a função de ativação φ. Esta atua como um filtro à saída do neurónio, podendo ser usada de modo inteligente nas camadas internas de modo a introduzir propriedades não lineares à rede.[54]

Alguns exemplos de funções de ativação típicas (com possíveis variantes) apresentam-se na Figura 3.5.

[54] Conforme descrito mais adiante, sendo a função de ativação uma característica inerente a cada camada de neurónios, numa rede neural (constituída por várias camadas) podem ser utilizadas várias funções de ativação.

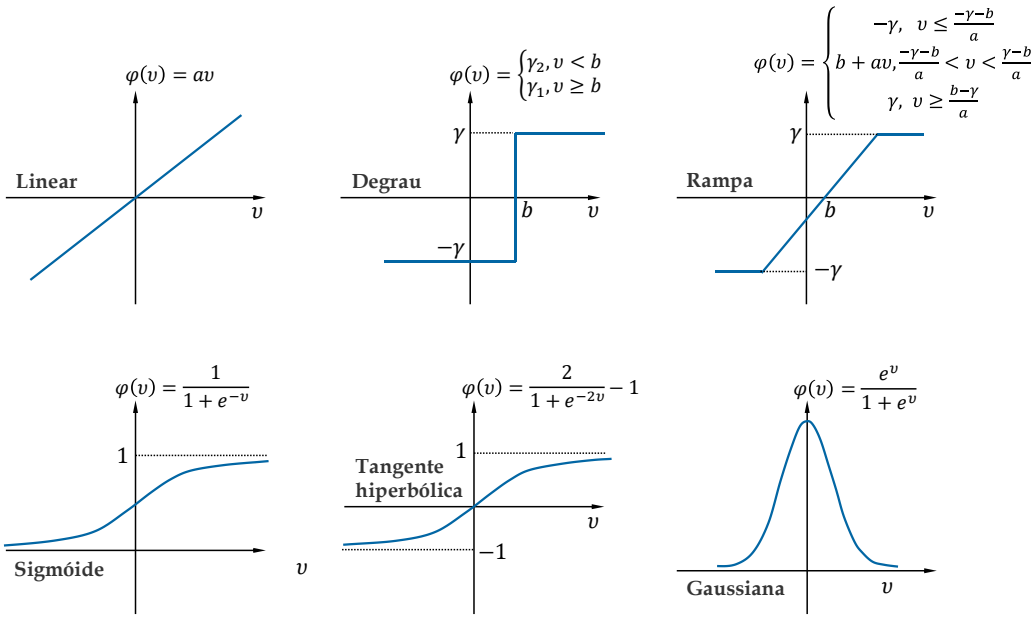


FIGURA 3.5 – Exemplos de funções de ativação (lineares e não lineares)

3.3.2. CAMADA

Os neurónios encontram-se organizados em camadas (*layers*). As ligações entre neurónios estão afetas de pesos, de tal forma que o sinal emitido por um neurónio é multiplicado pelo respetivo peso antes de dar entrada no neurónio seguinte. Cada camada é constituída por um conjunto de neurónios semelhantes, que, muito embora tenham pesos distintos, têm subjacente uma mesma função de ativação. Portanto, a camada é caracterizada pela função de ativação em uso e pelo número de neurónios que agrega.

Existem três tipos específicos de camadas a salientar (com características distintas):

- **Camada de entrada**: o objetivo desta camada é receber os valores de entrada dos atributos explanatórios para cada observação, tendo tantos neurónios quanto o número de *inputs* que queremos que o modelo tenha. A camada de entrada apresenta os padrões à rede, que comunica com uma ou mais camadas escondidas. Uma característica dos neurónios desta camada é atuarem como *passthrough*, no sentido em que não alteram os dados, deixando passar m valores (sendo m o número de neurónios). Eles apenas recebem um sinal, replicam-no e encaminham para todos os neurónios da camada seguinte. Com efeito, o vetor de pesos é um vetor com todas as entradas unitárias e a função de ativação é linear.

- **Camada escondida ou oculta**: esta camada caracteriza-se por poder ter um número arbitrário de neurónios e uma (única) função de ativação inerente. Podendo na prática existir várias camadas escondidas,[55] em cada camada são aplicadas determinadas transformações ao conjunto dos $\tilde{m}$ valores recebidos (*inputs*), gerando nova informação constituída por $\tilde{n}$ valores (*output*) para a camada seguinte.

- **Camada de saída**: esta camada tem tantos neurónios quanto o número de *outputs* desejados para o modelo (denotemos por n o número de neurónios da camada de saída).[56] Uma característica que merece especial cuidado com esta camada está relacionada com a função de ativação, a qual é escolhida tendo em conta a natureza do problema. Por exemplo: (i) no caso de previsões relativas a séries temporais, como prever valores futuros de mercado financeiros, deve ser usada a função linear; (ii) se quisermos prever uma probabilidade, deve ser usada uma sigmoide; (iii) se o objetivo for uma decisão binária, deve ser usada a função de *Heaviside*.

Assim, antecipando já o apresentado na secção seguinte, se numa camada pode existir um número variável de neurónios, numa rede neuronal, poder-se-á considerar a existência de várias camadas onde, entre a camada de entrada (composta por neurónios de entrada) e da camada de saída (composta por neurónios de saída), podem ser consideradas várias camadas escondidas intermédias (compostas por neurónios ocultos), onde se denotará por ℓ o número de camadas ocultas, ou profundidade da rede.

3.3.3. Rede Neuronal

Partindo dos conceitos de "neurónio" e "camada" apresentados, bem como das ligações que se estabelecem entre eles, podemos considerar a construção de uma rede neuronal.

Fazendo um paralelo com a parte biológica, uma rede neuronal artificial (RNA) pode ser vista como um conjunto de algoritmos computacionais, com uma arquitetura semelhante à do cérebro humano, desenhado para reconhecer padrões numéricos extraídos de dados reais (sejam

[55] Na verdade, a existência de várias camadas escondidas motiva, por si só, a existência de diferentes arquiteturas de redes neuronais (conforme descrito na secção seguinte).

[56] Com efeito, o *output* final por ser considerado um vetor $\boldsymbol{Y}$ com n elementos, $\boldsymbol{Y} = \{y_1, y_2, \dots y_n\}$.

eles imagens, sons, texto ou séries temporais) e com capacidade de aprendizagem (*Machine Learning*), ditada em parte ao se considerarem mais camadas ocultas na rede.

Esquematicamente, a rede neuronal pode representar-se como um sistema constituído por camadas de neurónios, as quais se encontram unidas por ligações direcionais (dendrites e sinapses). A título exemplificativo (e sem perda de generalidade), consideremos a Figura 3.6, onde ilustra uma rede neuronal considerando, além das camadas de entrada e de saída (com dois e um neurónio, respetivamente), duas camadas intermédias (escondidas) com quatro neurónios cada. Neste caso, cada neurónio de uma camada está ligado a todos os nós (neurónios) da camada adjacente seguinte, o que permite classificar a rede como completamente interligada. Caso tal não aconteça (algumas das possíveis ligações direcionais estiverem ausentes), a rede designa-se por parcialmente interligada.[57]

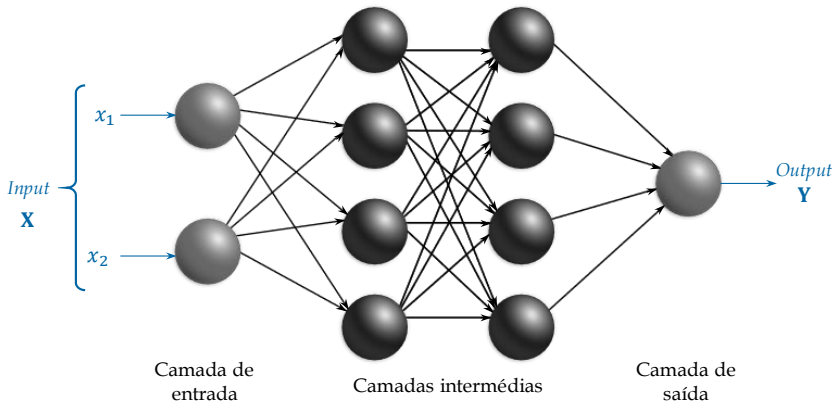


FIGURA 3.6 – Grafo de uma rede neuronal com duas camadas ocultas

Generalizando, a estrutura da rede neuronal fica completamente caraterizada pelo número de camadas, pelas unidades elementares que integram cada camada (número de neurónios e funções de ativação), pelas ligações (pesos) que se estabelecem entre elas e por um *bias* aplicado externamente.

Para finalizar, sobre o número de camadas e neurónios que integram a rede, importa referir que este deve ser suficientemente grande para captar a dinâmica do modelo, mas não tão grande que prejudique a capacidade de generalização da RNA. A utilização de um número razoável de camadas e de neurónios leva a que a rede não armazene toda a informação da aprendizagem, mas consiga generalizá-la de forma a evitar *overfitting*, fenómeno que corre quando os pesos

[57] Existem várias tipologias/arquiteturas de redes neuronais. No Anexo C.1 apresentam-se alguns grafos ilustrativos de redes neuronais.

levam o sistema a aprender demasiados detalhes sobre os dados em vez de compreender a sua 'estrutura global'.[58]

3.4. TREINO E APRENDIZAGEM DA REDE NEURONAL

Ao longo da Sec. 3.3 foi dado ênfase ao *design* de uma rede neuronal e conceitos básicos envolvidos. Outro tópico intrínseco às redes neuronais diz respeito ao processo de *Machine Learning* e, como tal, nesta secção será dada atenção aos aspetos relacionados com o treino e aprendizagem da rede neuronal.

3.4.1. PARADIGMAS DE APRENDIZAGEM

Designa-se por processo de aprendizagem de uma rede neuronal uma metodologia de cálculo dos parâmetros da rede, ou pesos sinápticos, $\boldsymbol{W}$, que conduzam a um comportamento pré-estabelecido do sistema para uma dada classe de padrões de entrada (*inputs*). O conjunto de regras pré-estabelecidas que permite a solução de um problema específico designa-se por algoritmo de aprendizagem, já o modelo de interação da rede com o ambiente onde opera define o paradigma de aprendizagem.

Assim, o processo de aprendizagem de uma RNA depende da topologia da rede, do algoritmo de treino utilizado, do paradigma de aprendizagem e de um conjunto de padrões, designado por conjunto de treino. Após o treino, o comportamento e desempenho da rede pode ser aferido recorrendo a um subconjunto independente de padrões de teste.

Os paradigmas de aprendizagem de RNAs, ilustrados esquematicamente na Figura 3.7, podem ser classificados em três grandes classes: aprendizagem supervisionada, não supervisionada e por reforço.[59]

No paradigma de aprendizagem supervisionada, o objetivo do processo de treino é a determinação dos parâmetros da rede de modo que, para cada padrão de entrada, seja obtida uma resposta tão próxima quanto possível de uma saída desejada. Os parâmetros são otimizados de forma iterativa, tendo em conta o padrão de entrada e o sinal de erro (diferença entre a saída

[58] Adiante serão feitas considerações mais consistentes sobre o fenómeno de *overfitting*.

[59] É comum encontrar na literatura ainda uma quarta opção designada de "aprendizagem semi-supervisionada" que não é mais do que um misto dos aspetos implícitos na supervisionada e não supervisionada.

desejada e a resposta real da rede). Este paradigma de aprendizagem é utilizado em problemas de classificação e regressão e recorre geralmente a um algoritmo de treino de retropropagação do sinal de erro (*Backpropagation*), proposto por Rumelhart *et al.* (1986) e descrito com mais detalhe na Sec. 3.4.3.

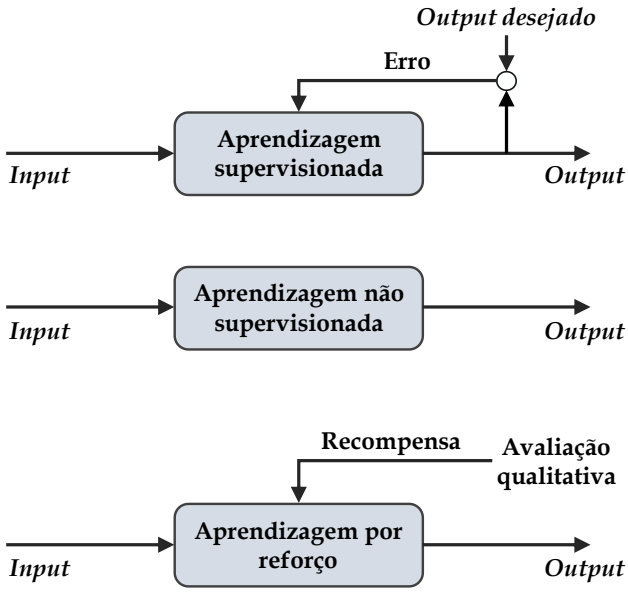


FIGURA 3.7 – Estruturas básicas dos três paradigmas de aprendizagem

Na aprendizagem não supervisionada, ou auto-organizada, não são apresentadas à rede as saídas desejadas para treino. Em vez disso, é fornecido um conjunto de regras de adaptação de pesos (algoritmos de treino) que devem contribuir para extrair características estatisticamente significativas, ou realizar uma transformação útil dos dados de entrada. Deste modo, a rede interatua diretamente com o ambiente e, uma vez ajustada às regularidades estatísticas do sinal de entrada, desenvolve a capacidade de criar representações internas para codificar a entrada e reclassificar a saída sem recorrer a supervisão externa (Becker & Plumbley, 1996). A adaptação dos pesos da rede é realizada em função das características dos padrões de entrada e do algoritmo de treino utilizado, o qual tem associado um determinado critério de desempenho que deverá ser otimizado no decurso do processo de aprendizagem. Este tipo de aprendizagem é utilizado sobretudo em problemas de *clustering* e associação, frequentemente como método de pré-processamento de dados, recorrendo a uma grande variedade de algoritmos de treino.[60]

[60] São de salientar o algoritmo Hebbiano (Hebb, 1949), os mapas topológicos de Kohonen (Kohonen, 1982) e a aprendizagem competitiva (Rumelhart & Zipser, 1985), entre outros.

Na aprendizagem por reforço a rede recebe apenas indicações imprecisas sobre a saída desejada, sendo o mapeamento da entrada-saída feito gradualmente através da interação com o ambiente, havendo apenas uma avaliação qualitativa (sem medição do erro) da saída e recebendo a rede um sinal de reforço heurístico (Barto, Sutton, & Anderson, 1983).

3.4.2. DESCIDA DO GRADIENTE E FUNÇÃO DE CUSTO

Centrando particular atenção no paradigma de aprendizagem supervisionado (o nosso caso de interesse), o processo recorre a regras de cálculo iterativo que ajustam os pesos da rede neuronal até estabilizar a diferença entre a resposta obtida e um resultado pré-definido. Sendo $d_i(t)$ e $y_i(t)$, respetivamente, os sinais de saída desejado e o obtido pelo neurónio i (da camada de saída) no instante t, é possível definir o sinal de erro por

$$e_i(t) = d_i(t) - y_i(t) \tag{3.8}$$

Os sinais de erro são convertidos em números reais por uma função de erro, ou função de custo, $\mathcal{C}(\boldsymbol{W})$, sendo os pesos e *bias* da rede neuronal ajustados pela sua minimização. Algumas funções de custo comuns em problemas de regressão e classificação utilizando redes neuronais são as seguintes (Bishop, 1995)

Soma dos quadrados dos erros $$\sum_{i=1}^{m} e_i^2 \tag{3.9}$$

Erro de Minkowski $$\sum_{i=1}^{m} |e_i|^R \, , com \; R \in \mathbb{N} \tag{3.10}$$

Erro médio quadrático $$\frac{1}{m}\sum_{i=1}^{m} e_i^2 \tag{3.11}$$

Cross-entropy $$-\sum_{i=1}^{m} [t_i \; log \; y_i + (1 - t_i) \; log \; y_i] \tag{3.12}$$

O problema de minimização de $\mathcal{C}(\boldsymbol{W})$ só pode ser resolvido analiticamente (utilizando uma matriz pseudo-inversa) quando a função de custo for a soma dos quadrados dos erros e no caso

de a função de ativação ser linear. Nos restantes casos, a solução pode ser encontrada recorrendo ao gradiente[61] da superfície de erro (hipersuperfície definida pela função de custo num espaço de dimensão $n+1$, sendo n a dimensão de $\boldsymbol{W}$)

$$\boldsymbol{\nabla}\mathcal{C}(\boldsymbol{W}) = \frac{\partial \mathcal{C}(\boldsymbol{W})}{\partial \boldsymbol{W}} \tag{3.13}$$

A minimização da função de custo é então conseguida de forma iterativa, por modificação dos pesos em cada iteração na direção oposta ao gradiente (vetor que aponta na direção do crescimento da função de custo) por aplicação de um fator, η, que controla a taxa de aprendizagem[62] em cada iteração. Assim, a regra de minimização da função de custo por descida do gradiente determina que os pesos em cada iteração sejam adaptados por

$$\boldsymbol{W}(t+1) = \boldsymbol{W}(t) - \eta\, \boldsymbol{\nabla}\mathcal{C}(\boldsymbol{W})|_{w=w(t)} \tag{3.14}$$

3.4.3. ALGORITMO *BACKPROPAGATION*

O algoritmo de retropropagação do erro (*Backpropagation*) descrito por Rumelhart *et al.* (1986) é o método de aprendizagem mais utilizado no treino supervisionado de redes neuronais. Na sua essência, este algoritmo traduz-se na minimização da função de erro por aplicação de uma técnica de gradiente, como acabámos de ver no parágrafo anterior. Contudo, atendendo à não linearidade da função de ativação e ao elevado número de parâmetros, típicos em aplicações de RNAs, o processo de treino pode tornar-se excessivamente lento, independentemente da taxa de aprendizagem. Por outro lado, a possibilidade de ocorrência de mínimos locais na função de erro (onde o gradiente se anula), pode causar convergência para um conjunto de pesos que não corresponda ao mínimo global.

O treino supervisionado da rede com este algoritmo ocorre num número finito de iterações, denominadas "épocas de treino", envolvendo uma sequência de dois passos: a propagação do sinal funcional, dado pela equação (3.7), e a retropropagação do erro. No primeiro passo, é apresentado um sinal à camada de entrada que se propaga através da rede, camada por camada,

[61] Define-se o operador gradiente por $\nabla = \left[\frac{\partial}{\partial w_1}, \frac{\partial}{\partial w_2}, \ldots, \frac{\partial}{\partial w_m}\right]^{\mathrm{T}}$.

[62] Se a taxa de aprendizagem for muito pequena, a aprendizagem pode tornar-se muito lenta, pois a trajetória definida por $\boldsymbol{W}(t)$ é muito suave. Se, pelo contrário, a taxa de aprendizagem for demasiado elevada, $\boldsymbol{W}(t)$ pode seguir uma trajetória com grandes oscilações, levando à divergência dos pesos (Jordan, 2018).

com um conjunto fixo de pesos, até se produzir o conjunto de saída (resposta real). No segundo, a saída é comparada com a resposta desejada e a diferença é utilizada para ajustar os pesos, sendo o sinal de erro propagado para trás através da rede, contendo informação a utilizar na próxima iteração (Haykin, 2009).

Em cada época, são apresentados ao algoritmo, um a um, exemplos de um conjunto de treino da rede. A atualização dos pesos pode ser feita no final da época, após terem sido apresentados todos os exemplos do conjunto de treino, ou sequencialmente, cada vez que um exemplo é apresentado ao algoritmo de aprendizagem. O processo de atualização dos pesos sinápticos é repetido época após época até que seja atingido um determinado critério de paragem.

O critério de paragem deve ser selecionado judiciosamente para evitar o sobre ajustamento da rede aos dados de treino (*overfitting*), de tal forma que os erros sejam perfeitamente corrigidos para um exemplo do conjunto de treino. Isso inibiria a capacidade de generalização da rede neuronal para encontrar uma solução válida para o conjunto de todos os dados de entrada. Um dos métodos utilizados para evitar o *overfitting* consiste em dividir o conjunto de treino em dois subconjuntos, um dos quais, denominado conjunto de validação, será utilizado para avaliar a qualidade do treino e a capacidade de generalização no final de cada época. O algoritmo termina quando o erro aumenta sucessivamente por um número definido de iterações, sendo a melhor solução a que apresentou menor erro durante o processo de treino. Este método, designado por validação cruzada (*cross-validation*), será discutido em detalhe na Sec. 3.6.

Em termos computacionais, o algoritmo de *Backpropagation* apresenta uma série de passos, para cada conjunto de treino, os quais se descrevem em seguida e estão esquematizados na Figura 3.8.

1. **Inicialização**

   O objetivo da inicialização é obter um conjunto aleatório de pesos iniciais que permitam obter um gradiente da superfície de erro entre 0 e 1, havendo vários métodos de inicialização, como o de "*Xavier*" (Glorot & Bengio, 2010) ou o de He (He, Zhang, Ren, & Sun, 2015). Na ausência de qualquer informação prévia, selecionam-se os pesos sinápticos, e os parâmetros de decisão, ϑ, de uma distribuição uniforme de média nula e variância tal que o desvio-padrão dos potenciais de ativação, v_k, dos neurónios da camada de entrada se encontre na transição entre as regiões linear e saturada da função de ativação sigmóide (ver Figura 3.5).

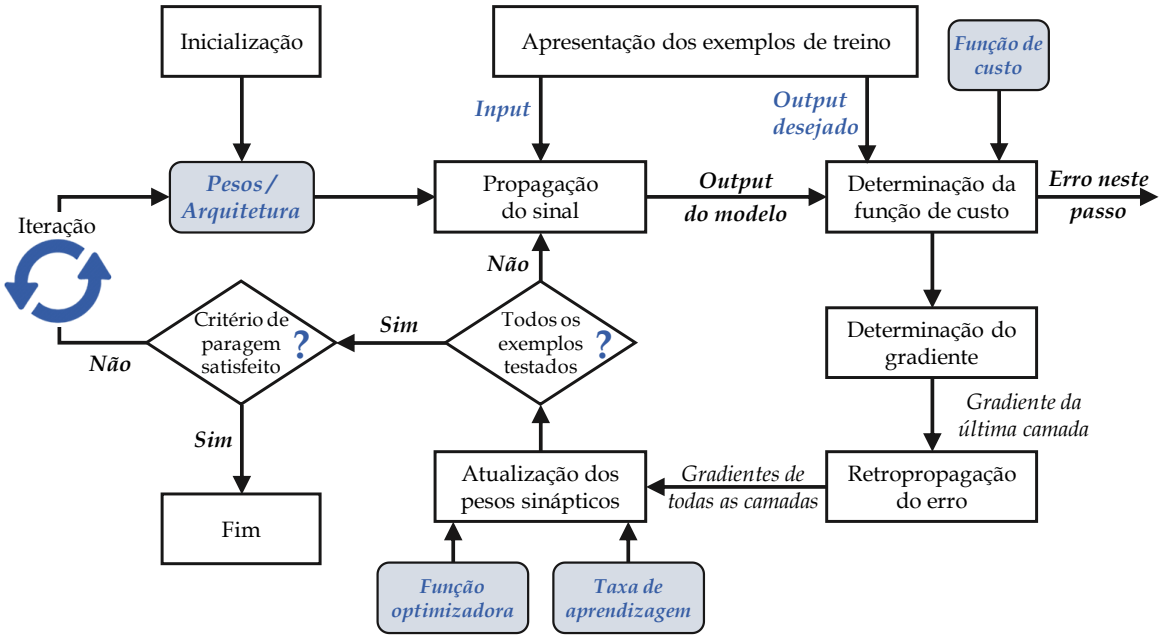


FIGURA 3.8 – Passos computacionais do algoritmo de *Backpropagation*

2. Apresentação dos exemplos de treino

É apresentada à rede uma época de exemplos de treino, contendo cada exemplo um vetor $\boldsymbol{X}(t)$ aplicado à camada de entrada e o vetor resposta desejada na camada de saída, $\boldsymbol{D}(t)$. Os exemplos são apresentados um a um, seguindo uma ordem pré-definida e, para cada exemplo, é realizada a sequência de passos 3 a 6.

3. Propagação do sinal

Para um dado exemplo são calculados os potenciais de ativação e os sinais funcionais da rede, prosseguindo para a frente, camada por camada. De acordo com as equações (3.6) e (3.7), para o neurónio k da camada l, o potencial de ativação e o sinal de saída são, respetivamente,

$$v_k^l(t) = \sum_{j=0}^{m_l} w_{kj}^l(t) y_j^{l-1}(t) \tag{3.15}$$

$$y_k^l = \varphi_k\left(v_k^l(t)\right) \tag{3.16}$$

onde $w_{kj}^l(t)$ é o peso sináptico que o neurónio k recebeu do neurónio j (da camada anterior) na iteração t, $y_j^{l-1}(t)$ é o sinal de saída da camada anterior nessa iteração e φ_k a função de ativação. Para $j = 0$, temos $y_0^{l-1}(t) = 1$ e $w_{k0}^l(t) = b_k^l(t)$ é o bias aplicado ao neurónio.

Se o neurónio pertencer à primeira camada oculta (i.e., $l = 1$), o sinal de saída da camada anterior é o j-ésimo elemento do vetor de entrada $\boldsymbol{X}(t)$, isto é $y_j(t) = x_j(t)$. Quando o neurónio pertencer à última camada oculta ($l = \ell$), obtém-se o k-ésimo elemento do vetor de saída dessa iteração, $y_k(t) = y_k^{\ell}$.

4. Determinação da Função de custo

Com base nos vetores resposta desejada, $\boldsymbol{D}(t)$, e saída obtida, $\boldsymbol{Y}(t)$, determina-se o sinal de erro, sendo os seus elementos dados pela equação (3.8), e o valor instantâneo da função de custo, $\mathcal{C}(t)$,[63] utilizando uma das funções descritas nas equações (3.9) a (3.12). Utilizando, por exemplo, a soma dos quadrados dos erros como função de custo, teríamos

$$e_k(t) = d_k(t) - y_k(t) \tag{3.17}$$

$$\mathcal{C}(t) = \frac{1}{2}\sum_{k=1}^{m_\ell} e_k(t)^2 \tag{3.18}$$

5. Determinação do gradiente

Por aplicação da regra da cadeia,[64] e derivando parcialmente as quatro equações anteriores em ordem à variável independente correspondente, é possível calcular o gradiente identificado na equação (3.13) em relação aos pesos da camada escondida para a camada de saída

$$\frac{\partial \mathcal{C}(t)}{\partial w_{lk}(t)} = -\sum_k e_k(t)\varphi_k'\big(v_k(t)\big)y_k(t) \tag{3.19}$$

[63] Obviamente, o valor da função de custo depende de todos os pesos sinápticos da rede, utilizando-se a notação $\mathcal{C}(t)$ por simplicidade. Se a atualização dos pesos for feita apenas no final da época, deve utilizar-se antes o valor médio da função de custo, $\mathcal{C}_{médio}(t) = \frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{m_\ell} e_k(t)^2$, para os N exemplos utilizados nessa época.

[64] A função de custo depende do sinal de erro, que depende do sinal de saída, que depende dos potenciais de ativação, que dependem dos pesos sinápticos. Então, por aplicação da regra da cadeia, podemos escrever $\frac{\partial \mathcal{C}(t)}{\partial w_{kj}(t)} = \frac{\partial \mathcal{C}(t)}{\partial e_k(t)}\frac{\partial e_k(t)}{\partial y_k(t)}\frac{\partial y_k(t)}{\partial v_k(t)}\frac{\partial v_k(t)}{\partial w_{kj}(t)}$, sendo $\frac{\partial \mathcal{C}(t)}{\partial e_k(t)} = e_k(t)$ por derivação da equação (3.18), $\frac{\partial e_k(t)}{\partial y_k(t)} = -1$ por derivação da equação (3.17), $\frac{\partial y_k(t)}{\partial v_k(t)} = \varphi_k'\big(v_k(t)\big)$ por derivação da equação (3.16), e $\frac{\partial v_k(t)}{\partial w_{kj}(t)} = y_k(t)$ por derivação da equação (3.15).

6. Retropropagação do erro

O erro de saída é distribuído para trás, através das componentes do gradiente local da superfície de erro, definidas pelo produto da derivada do sinal de saída pelo sinal de erro (para neurónios da camada de saída ℓ), ou pelo produto da derivada da saída pela soma ponderada dos gradientes locais calculados para os neurónios da próxima camada oculta (para neurónios das camadas ocultas l), respetivamente temos

$$\delta_k^l(t) = \frac{\partial \mathcal{C}(t)}{\partial v_k^l(t)} = \begin{cases} \varphi_k'\left(v_k^\ell(t)\right) e_k^\ell(t) & (camada\ de\ saída) \\ \\ \varphi_k'\left(v_k^l(t)\right) \sum_{j=0}^{m_l} w_{kj}^l(t)\delta_j^{l-1}(t) & (camadas\ ocultas) \end{cases} \tag{3.20}$$

7. Atualização dos pesos sinápticos

Os pesos sinápticos de cada camada oculta l são atualizados tendo em conta a taxa de aprendizagem, η, de acordo com a regra dos gradientes locais generalizada

$$w_{kj}^l(t+1) = w_{kj}^l(t) + \alpha\left[w_{kj}^l(t-1)\right] + \eta\delta_k^l(t)y_j^{l-1}(t) \tag{3.21}$$

derivada da equação (3.14) por inclusão de um termo de momento, definido por

$$\alpha\left[\mathrm{w}_{\mathrm{kj}}^{\mathrm{l}}(\mathrm{t}-1)\right], \quad \text{com } \alpha > 0$$

que permite aumentar a taxa de aprendizagem sem oscilações significativas na trajetória de $\boldsymbol{W}(t)$.

8. Iteração

Repetem-se os passos 2 a 7 para novas épocas de treino até que seja satisfeito o critério de paragem.

Uma vez estabelecidos os passos do algoritmo *Backpropagation*, rapidamente se depreende que o treino de uma RNA não é tarefa simples. Mais, face à existência de estruturas de rede mais complexas, pode haver necessidade de adaptações deste algoritmo à estrutura em causa, conforme se descreve na secção seguinte.

3.5. ALGUMAS ARQUITETURAS DE REDES NEURONAIS

A escolha da estrutura da rede neuronal (definida em Sec. 3.3.3), também designada de arquitetura ou topologia da rede, determina fortemente os resultados obtidos, pelo que a sua definição é preponderante na implementação de uma rede neuronal.

De acordo com a literatura, podemos identificar três classes genéricas de arquiteturas de redes neuronal: redes *feedforward* simples (uma camada oculta), redes *feedforward* multicamada e redes recorrentes (Haykin, 2009). Os neurónios que integram as camadas ocultas intervêm entre as camadas de entrada e de saída de forma útil para a aprendizagem da rede.

> *By adding one or more hidden layers, the network is enabled to extract higher-order statistics from its input. In a rather loose sense, the network acquires a global perspective despite its local connectivity, due to the extra set of synaptic connections and the extra dimension of neural interactions…* (Churchland e Sejnowski *apud* Haykin, 2009, p. 22).

Sem formalismo matemático, na Figura 3.6 apresentou-se um grafo de uma rede *feedforward* multicamada (com duas camadas ocultas). Quanto às redes recorrentes, estas distinguem-se por apresentarem ligações de realimentação. Numa das modalidades possíveis,[65] essa realimentação ocorre nas camadas ocultas da rede, de tal modo que cada neurónio é realimentado pelo seu próprio *output*, como ilustrado na Figura 3.9.

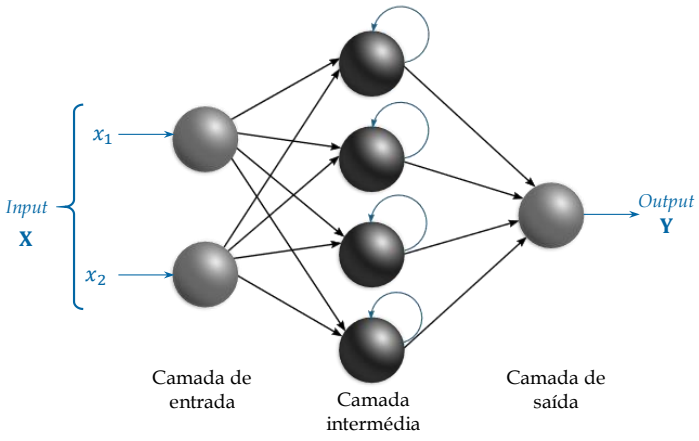


FIGURA 3.9 – Grafo de uma rede neuronal recorrente com uma camada oculta

[65] Dentro das redes recorrentes, existem diversas arquiteturas possíveis, podendo existir casos onde as camadas subsequentes geram um *input* de realimentação para as camadas anteriores. O caso descrito (sem perda de generalidade) é aquele em que o *output* gerado por cada neurónio das camadas intermédias realimenta o próprio neurónio (aprendizagem armazenada e utilizada no treino seguinte).

As ligações de realimentação têm um impacto profundo na capacidade de aprendizagem da rede e no seu desempenho, de onde poderá resultar um comportamento dinâmico não linear, admitindo que a rede neuronal contenha unidades não lineares (Haykin, 2009).

Face às arquiteturas referidas, o nosso foco de interesse recaiu naquelas que envolvem aprendizagem profunda – *Deep Neural Networks* – DNN. A complexidade deste tipo de arquitetura apresenta algumas particularidades que podem ser tidas como limitações à *priori*. Por um lado, o poder e/ou tempo computacional exigidos, visto que é sempre possível adicionar mais camadas ocultas, mais neurónios, ou mesmo criar arquiteturas mais complexas que consigam reter memória (usufruindo da capacidade de cada neurónio poder ser realimentado, no futuro, com o seu último *output*). Por outro lado, o facto de a rede se comportar como uma *black-box*, já que o processo de modelação/treino da rede acaba por estar sujeito a alguma aleatoriedade, algo que não se consegue captar nem controlar.[66]

As DNN podem ser vistas como componentes das aplicações de *Machine Learning*, envolvendo algoritmos para reforço da aprendizagem (conforme descrito em parte na Sec. 3.4). Exemplos de arquiteturas de DNN são os três casos que nos propomos desenvolver nas secções seguintes e sobre os quais nos debruçámos em termos de implementação e aplicação: (1) *Multilayer Perceptron* (MLP) (2) *Recurrent Neural Networks* (RNN) e (3) *Long Short-Term Memory* (LSTM). Será dada particular ênfase à descrição de arquiteturas LSTM, sendo estas um caso particular das RNN (e que mereceu maior atenção em termos de estudo computacional neste trabalho).

3.5.1. *MULTILAYER PERCEPTRON*

A arquitetura *Multilayer Perceptron* (MLP) foi a que esteve, em parte, na base da exploração feita na Sec. 3.3.3. Aliás, a Figura 3.6 ilustra perfeitamente uma arquitetura MLP com duas camadas ocultas.[67] Estas camadas são o verdadeiro mecanismo computacional do MLP, arquitetura que, segundo alguns autores, poderá ser vista como um dos primeiros *steps* no que

[66] No desenhar da nossa metodologia e na implementação empírica, procuraremos ter em atenção estes dois aspetos (tempo computacional e existência de uma *black-box*) para tornar mais sustentada a análise.

[67] Importa clarificar que uma rede MLP poderá apenas conter uma camada oculta, nesse caso não será adequado considerar a rede como uma DNN, uma vez que não existe algo de profundo na aprendizagem. Uma MLP com uma única camada oculta é comumente referida na literatura como *vanilla* MLP.

respeita às DNN que (apesar da sua simplicidade), por conterem várias camadas ocultas, evidenciam já uma estrutura de *Deep Learning* (Data Science Academy, 2019).

As arquiteturas MLP têm particular aplicação em problemas cujo objetivo passa pelo treino de um conjunto de dados de entrada, procurando captar a informação relevante (por exemplo, tendências), para conseguir uma generalização. O treino desta arquitetura poderá ser feito com recurso ao algoritmo *Backpropagation* descrito na Sec. 3.4, sendo os parâmetros da rede, pesos e *bias*, ajustados por minimização do erro[68] (valor médio do gradiente da função de custo). Contudo, em termos práticos, o término do processo de aprendizagem é muitas vezes ditado pelo "número de épocas de treino" decidido pelo investigador (um dos hiperparâmetros[69] da rede).

Sobre a aplicação algoritmo *Backpropagation* às arquiteturas MLP, importa notar que, como referido anteriormente, o gradiente da superfície de erro tem regiões não convexas sendo, muitas vezes, impossível escapar à convergência para um ponto de mínimo local, ao invés do mínimo global. Algumas sugestões de adaptações do algoritmo base, como o *Mini-batch Gradient Descedent*, ou mesmo o caso do algoritmo ADAM, introduzido por Kingma e Ba (2015), têm sido sugeridas na literatura para fazer face a este e a outros problemas (Haykin, 2009).

Embora a arquitetura MLP seja muito utilizada, esta apresenta menor robustez face a outras arquiteturas pois cada neurónio recebe apenas como *input* os dados de treino, não registando aprendizagem ao longo do tempo. Tal facto pode ser uma limitação quando os dados apresentam alguma sequência e/ou dependência temporal, limitação essa que se torna evidente no caso de séries temporais.[70] Em arquiteturas mais robustas, além da memória resultante de cada época de treino, existe um *input* de aprendizagem direto em algumas unidades resultante do próprio *output* gerado numa fase de treino anterior, dando ao próprio neurónio uma aprendizagem sequencial (e com o tempo).

[68] As várias métricas de avaliação do erro serão descritas na Sec. 4.4.

[69] Os hiperparâmetros da rede são as variáveis que determinam a estrutura da rede (número de camadas, número de neurónios por camada e funções de ativação) e as variáveis que determinam como a rede é treinada (por exemplo a taxa de aprendizagem e o número de épocas de treino).

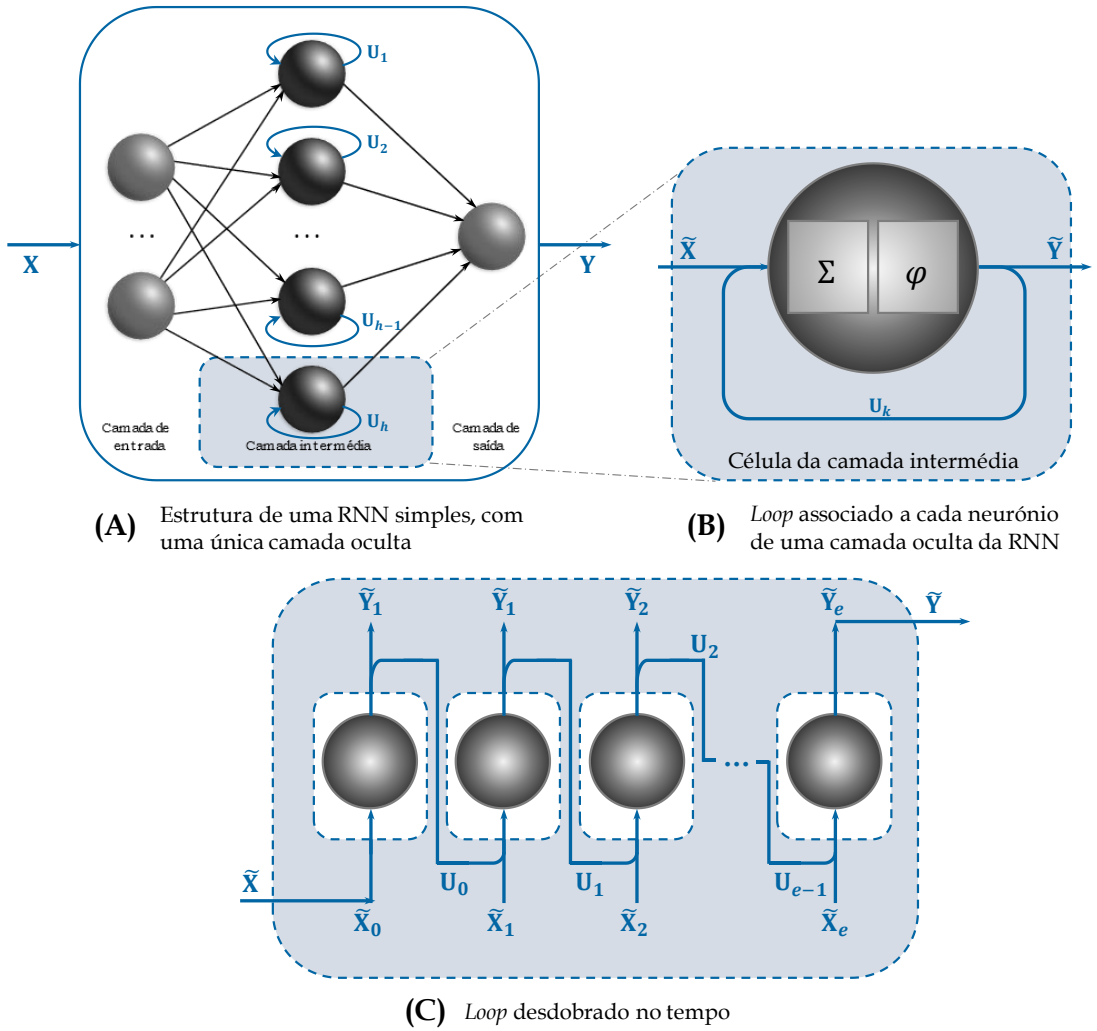
[70] Esta limitação será objeto de particular reflexão neste estudo. Como tal, destacamos aqui esse facto para que melhor se compreenda algumas das opções/implementações metodológicas feitas em termos de estudo empírico (Sec. 5.2.4).

3.5.2. *RECURRENT NEURAL NETWORKS*

As arquiteturas de *Recurrent Neural Networks* (RNN) caracterizam-se por fazer face à limitação referida, na medida em que contemplam uma dupla aprendizagem: a que resulta do ciclo de treino (seguindo os mecanismos gerais descritos paras as redes MLP) e uma "autoaprendizagem". Esta arquitetura, embora possa também ter um número arbitrário de camadas ocultas, difere das redes *feedforward* na medida em que inclui um *feedback loop*, onde o neurónio utiliza o seu último *output* como *input* no instante futuro (ou seja, na época de treino seguinte), sendo as matrizes de pesos partilhadas ao longo do tempo.

Pela componente dinâmica que esta arquitetura de redes integra em relação ao tempo (conseguindo fornecer memória à rede), as RNN mostram-se promissoras na modelação de dados temporais, como é o caso da previsão de séries temporais.

Para formalizar algumas ideias, vejamos a Figura 3.10 (A).



(A) Estrutura de uma RNN simples, com uma única camada oculta

(B) *Loop* associado a cada neurónio de uma camada oculta da RNN

(C) *Loop* desdobrado no tempo

FIGURA 3.10 – Arquitetura de uma RNN

Sem perda de generalidade, considere-se uma rede neuronal com uma camada oculta ($\ell = 1$) com h neurónios, um vetor de *input* de m entradas, $\boldsymbol{X}$, da qual se obtém um certo vetor de *output*, $\boldsymbol{Y}$, com apenas um elemento ($n = 1$). A novidade surge nas ligações de realimentação em que o vetor de *output*, $\boldsymbol{U}_k$, de cada neurónio $k = 1, .., h$ da camada oculta vai realimentar o próprio na iteração (época de treino) seguinte – mecanismo de "autoaprendizagem".

Centrando particular atenção no que opera em cada neurónio, podemos considerar que as RNN são redes com *loops*, permitindo que as informações persistam em cada momento do treino. Assim, um *loop* permite que a informação processada num dado neurónio k seja guardada e transferida de uma iteração para a próxima: $\boldsymbol{U}_k$ (ver Figura 3.10 (B)[71]).

Com efeito, cada neurónio da camada oculta da RNN pode ser considerado como "várias cópias" passando uma mensagem para si próprio em cada época de treino. Desta forma, estes neurónios apresentam uma estrutura "desdobrada" no tempo, conforme esquematizado na Figura 3.10 (C).[72]

Assim, considerando e o número de épocas de treino, em cada iteração $t = 1, \dots, e$, o neurónio será alimentado por um *input*, $\widetilde{\boldsymbol{X}}_t$, cujo *output* não só vai alimentar a camada seguinte da rede, $\widetilde{\boldsymbol{Y}}_t$, como será armazenado e incluído como *input* de aprendizagem na próxima época de treino, $\boldsymbol{U}_t$, vetor definido na equação (3.22)[73]

$$\boldsymbol{U}_t = \begin{cases} f\left(\widetilde{\boldsymbol{X}}_t, \boldsymbol{U}_{t-1}\right), & t = 1, \dots, e \\ f\left(\widetilde{\boldsymbol{X}}_t\right) \quad , & t = 0 \end{cases} \tag{3.22}$$

Embora exista uma base geral, é possível identificar alguns detalhes distintos em algumas arquiteturas de RNN, e que permite considerar quatro tipos principais de RNN (Haykin, 2009). Procurando não tornar extensa esta apresentação, optámos por apresentar a ideia do caso mais geral e que vem no seguimento do MLP – *Recurrent Multilayer Perceptron* (RMLP), sendo que os outros se obtêm considerando algumas adaptações.

Tal como sucede nas redes MLP, uma RNN poderá conter tantas camadas ocultas quanto se queira, sendo reconhecidas vantagens quando estas possuem mais do que uma camada oculta

[71] Refira-se que, por simplicidade, não se encontra representado em cada neurónio o *input* referente ao *bias*.

[72] Será desta característica das RNN que se depreende tratar-se de uma arquitetura que dá melhor resposta a problemas que envolvam dados sequenciais, como é o caso das séries temporais.

[73] Salvaguarda-se que o vetor $\boldsymbol{U}_t$, obtido de cada neurónio, é afeto de outro *input* referente ao valor do *bias*.

"…*for the same reasons that static multilayer perceptrons are often more effective and parsimonious than those using a single hidden layer*…" (Haykin, 2009, p. 794).

A estrutura arquitetural de uma RMLP tem por base os procedimentos descritos nas redes MLP, onde, em cada neurónio das camadas ocultas, opera o mecanismo de realimentação acima exposto.

Em termos de treino da rede, o algoritmo de *Backpropagation* descrito na Sec. 3.4 pode ser generalizado para aplicar em RNNs (em específico nas RMLPs). Esta generalização, proposta em (Pineda, 1987), é normalmente designada por retropropagação no tempo (*Backpropagation Through Time*-BPTT). O algoritmo baseia-se igualmente num gradiente descendente (procurando minimizar uma função de custo) e é passível de ser implementado por épocas de treino.

Conceptualmente, o BPTT pode ser derivado por transformação da sequência temporal de operações da rede numa sequência espacial, por desdobramento da operação temporal em camadas, uma por cada passo de tempo, contendo todas o mesmo número de neurónios e os mesmos pesos da camada não desdobrada. Para cada passo de tempo (camada l da rede desdobrada) de uma época de treino $[t_0; t_1]$ a ligação sináptica entre o neurónio k da camada l e o neurónio j da camada $l + 1$ é uma cópia da ligação sináptica entre o neurónio k e o neurónio j de duas camadas adjacentes da rede recorrente.

Numa época de treino, o sinal é propagado, os erros são calculados para cada passo de tempo e acumulados, tal como descrito nos passos 3 a 5 do algoritmo de *Backpropagation*. No final da época, é realizado um único passo de retropropagação do erro, para calcular os valores dos gradientes locais, a rede neuronal recorrente é reposta e os pesos atualizados por

$$w_{kj}(t+1) = w_{kj}(t) + \eta \sum_{t=t_{0+1}}^{t_1} \delta_k(t) x_j(t-1) \tag{3.23}$$

O problema principal deste algoritmo é ser muito exigente em termos computacionais (custo computacional elevado). Em cada época, é necessária a apresentação à rede, simultaneamente, de cada sequência de entrada e da respetiva sequência de saída desejada. Para grandes sequências ou sequências de dimensão desconhecida a implementação torna-se impraticável.

Apesar das arquiteturas RNN possuírem 'memória temporal', esta é de curto prazo. Esta é uma limitação apontada a estas arquiteturas, uma vez que a rede terá dificuldade em ir

transportando a informação relativa a etapas anteriores para todas as etapas posteriores, sobretudo quando consideradas muitas épocas de treino.

Efetivamente, durante o passo de *Backpropagation*, as RNN são propicias a serem afetadas pelo problema da dissipação do gradiente (*vanishing gradient*). Caso o valor do gradiente se torne muito baixo, a contribuição da iteração para a aprendizagem torna-se diminuta e, assim, com o avançar das épocas de treino, a rede acaba por receber uma pequena atualização e deixa de aprender. Deste modo, como as camadas não registam aprendizagem, as RNN tendem a esquecer o que foi visto em sequências de treino anteriores (principalmente em épocas de treino mais antigas), daí se dizer que estas possuem apenas uma memoria de curto prazo (Data Science Academy, 2019).

Para fazer face a esta limitação, foram introduzidas variantes de redes recorrentes capazes de armazenar memória ao longo das épocas de treino, conforme a descrita na secção seguinte.

3.5.3. LONG SHORT-TERM MEMORY

As arquiteturas *Long Short-Term Memory* (LSTM), tidas como redes de memória de longo prazo, são um tipo especial de RNN capazes de "aprender" dependências de longo prazo. Acrescendo alguns procedimentos internos aos mecanismos gerais de aprendizagem descritos nas RNN, estas 'novas' arquiteturas de rede neuronal conseguem selecionar informação, preservando não só as memórias antigas consideradas mais importantes (as que mais contribuíram para reduzir o erro), como também eliminar outras que se mostraram menos relevantes.

Além de outras potencialidades, no caso específico do estudo de séries temporais, esta arquitetura apresenta melhor desempenho que as RNN (ou as MLP) na captura de dependências de longo prazo (Data Science Academy, 2019). Deste modo, são reconhecidas vantagens aquando da modelação e previsão de séries temporais pela dupla capacidade da rede: (1) "lembrar" aprendizagens importante mais antigas (ou mesmo o ponto de partida), algo que nas RNN padrão vai sendo esquecido (apenas há *input* de aprendizagens recentes); (2) "esquecer" a informação considerada não relevante para aprendizagem.

Em termos metodológicos e topológicos, as redes LSTM possuem uma estrutura em cadeia que contém diferentes blocos de memória, designados por células (correspondentes a um neurónio da camada oculta). Um dos conceitos chave desta arquitetura é precisamente o "estado da célula" que, em cada momento do treino, t, atua como uma via de transporte de informações

relevantes ao longo do processamento da sequência – "memória" da rede – desde o *input* recebido, $\boldsymbol{C}_{t-1}$, ao *output* enviado, $\boldsymbol{C}_t$, (linha horizontal que percorre toda a cadeia de células, envolvendo apenas operações matemáticas elementares – ver Figura 3.11). Este fluxo de informação, ao ser (re)alimentado e (re)atualizado várias vezes em cada iteração do treino, é responsável por acoplar informações de longo prazo (resultantes de etapas anteriores) e permitir uma atualização da sua importância, removendo ou adicionando informações ao estado atual da célula.

Este processo de atualização permanente é regulado por estruturas internas de cada célula, chamadas *gates* (outro dos conceitos centrais dos LSTM), as quais são responsáveis por "manipular" a informação a passar ao estado da célula (permitindo que seja ou não transmitida). Ou seja, estas *gates* são precisamente os mecanismos que regulam o fluxo de informações que alimentam o estado da célula, permitindo à rede aprender que informação relevante deve reter e o que deve esquecer durante o treino. Conforme esquematizado na Figura 3.11 e explicado em seguida, há três tipos de *gates* distintas (*Forget Gate*, *Input Gate* e *Output Gate*), cada uma com um objetivo específico.

Como a diferença relativamente às RNN está nas operações que ocorrem em cada célula, partindo dos mecanismos descritos anteriormente, centremos atenção ao que, passo a passo, opera em termos matemáticos numa célula LSTM e qual o objetivo particular de cada *gate*.

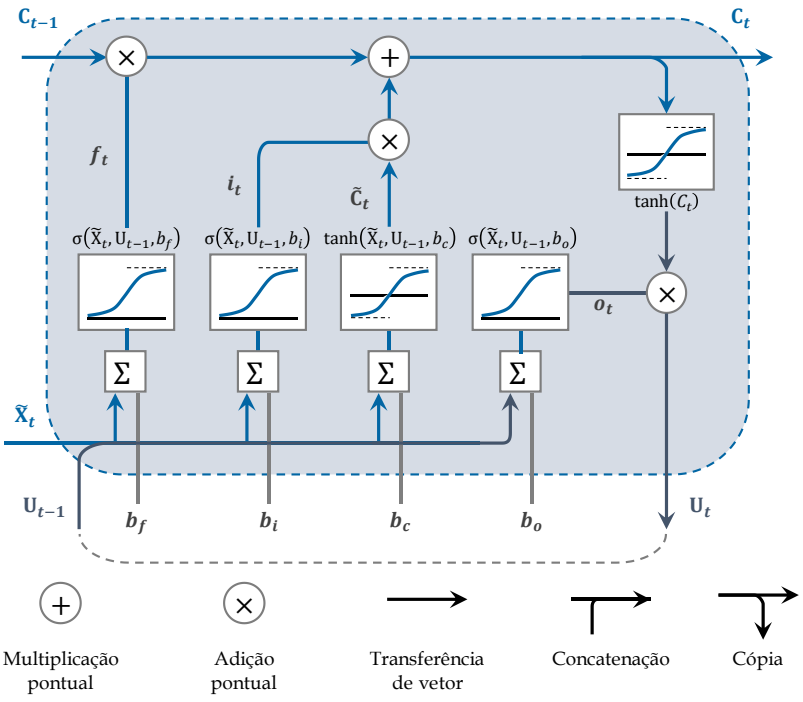


FIGURA 3.11 – Estrutura de uma célula LSTM

Especificamente:

- **1º Passo**

  O objetivo deste passo é remover, do estado da célula, a informação que deixou de ser útil, decidindo-se que informações devem ser descartadas ou mantidas (função da *Forget Gate*). Essa decisão é tomada por uma "rede neuronal sigmóide"[74]. Assim, conforme esquematizado na Figura 3.12 e num dado momento de interação t, a *Forget Gate* é alimentada pelo vetor de *inputs* atual, $\widetilde{\boldsymbol{X}}_t$, um vetor de *output* da própria célula na iteração anterior, $\boldsymbol{U}_{t-1}$, designado na arquitetura LSTM de "estado oculto" e o *bias*, b_f.

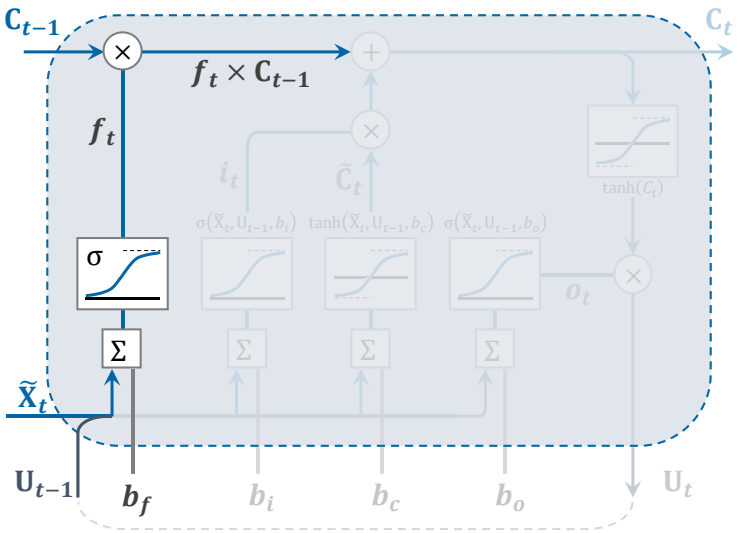


FIGURA 3.12 – Operações e fluxos de informação na célula LSTM (*Forget Gate*)

Após as operações clássicas envolvidas na rede neuronal (como a multiplicação por uma matriz de pesos, $\boldsymbol{W}_f$, seguida da adição do *bias*), o valor passa por uma função sigmóide (σ) sendo gerado o *output* da *Forget Gate*, f_t, um escalar entre 0 e 1, conforme descrito em (3.24).

$$f_t = \sigma\left[\boldsymbol{W}_f \cdot \left(\widetilde{\boldsymbol{X}}_t, \boldsymbol{U}_{t-1}\right) + b_f\right], \qquad t = 1, \ldots, e \tag{3.24}$$

O valor de f_t vai alimentar o estado de célula, sendo multiplicado por cada entrada do vetor *input*, $\boldsymbol{C}_{t-1}$. Com isso, é atualizado o estado da célula para $\boldsymbol{C}_t = f_t \times \boldsymbol{C}_{t-1}$. Em

[74] Esta rede devolve valores entre 0 e 1. Essa transformação é útil para atualizar ou esquecer os dados. No estado da célula, com o produto por 0 consegue-se que a informação seja eliminada ou "esquecida"; com a multiplicação por 1 consegue-se que a informação permaneça, ou seja "mantida". Deste modo, a rede pode aprender que dados são irrelevantes (e podem ser esquecidos) e quais são importantes (e devem ser mantidos).

particular, nos valores extremos de f_t (0 ou 1), temos: (i) se o output for $f_t = 0$, a informação é totalmente esquecida; (ii) se o output for $f_t = 1$, a informação é totalmente retida para uso futuro. Para os restantes valores de $0 < f_t < 1$, a informação é parcialmente retida/esquecida.

- **2º Passo**

  Adicionar nova informação útil ao estado da célula é o objetivo desta fase a qual é executada em três partes (conforme esquematizado na Figura 3.13).

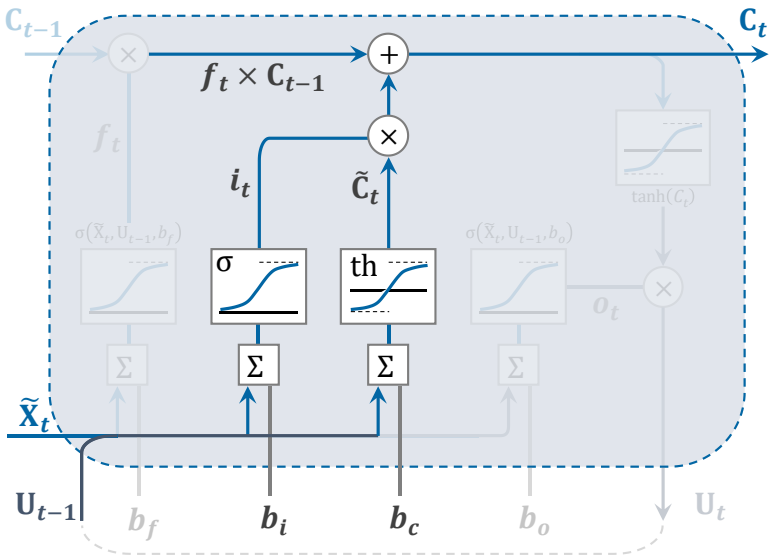


FIGURA 3.13 – Operações e fluxos de informação na célula LSTM (*Input Gate*)

Primeiramente (função da *Input Gate*), usando as entradas $\tilde{\boldsymbol{X}}_t$ e $\boldsymbol{U}_{t-1}$ (e o *bias* respetivo, b_i), a informação é regulada usando a rede sigmóide que filtra os valores a serem lembrados e atualizados, de forma similar ao descrito no 1º Passo, com o *output* i_t descrito em (3.25).

$$i_t = \sigma\left[\boldsymbol{W}_i \cdot \left(\tilde{\boldsymbol{X}}_t, \boldsymbol{U}_{t-1}\right) + b_i\right], \qquad t = 1, \ldots, e \tag{3.25}$$

Em seguida, uma "rede neuronal tangente hiperbólica"[75], igualmente alimentada por $\tilde{\boldsymbol{X}}_t$ e $\boldsymbol{U}_{t-1}$ (e o *bias* respetivo, b_c), é responsável pela criação de um vetor de "novos valores

[75] A rede tangente hiperbólica é usada para ajudar a regular os valores que circulam na rede, dado que reduz os valores ao intervalo entre -1 e 1. Com isto evita-se que alguns valores se possam tornar "explosivos", uma vez que estes estão sujeitos a diversas operações matemáticas ao longo da rede.

candidatos", $\widetilde{\boldsymbol{C}}_t$, com entradas de -1 a 1, que poderão ser adicionados ao estado da célula, conforme (3.26).

$$\widetilde{\boldsymbol{C}}_t = tanh\left[\boldsymbol{W}_c \cdot \left(\widetilde{\boldsymbol{X}}_t,\ \boldsymbol{U}_{t-1}\right) + b_c\right], \qquad t = 1, \dots, e \tag{3.26}$$

Finalmente os valores desse vetor, $\widetilde{\boldsymbol{C}}_t$, e os valores regulados, i_t, são multiplicados para obter o vetor de informações úteis, $i_t \times \widetilde{\boldsymbol{C}}_t$, a ser adicionado ao estado da célula, definindo-se o estado "final", em cada momento de iteração t, pela equação (3.27).

$$\boldsymbol{C}_t = f_t \times \boldsymbol{C}_{t-1} + i_t \times \widetilde{\boldsymbol{C}}_t, \qquad t = 1, \dots, e \tag{3.27}$$

- **3º Passo**

  Extrair informações úteis do estado da célula atual para serem apresentadas à própria célula na próxima iteração é o objetivo deste último passo (descrito na Figura 3.14).

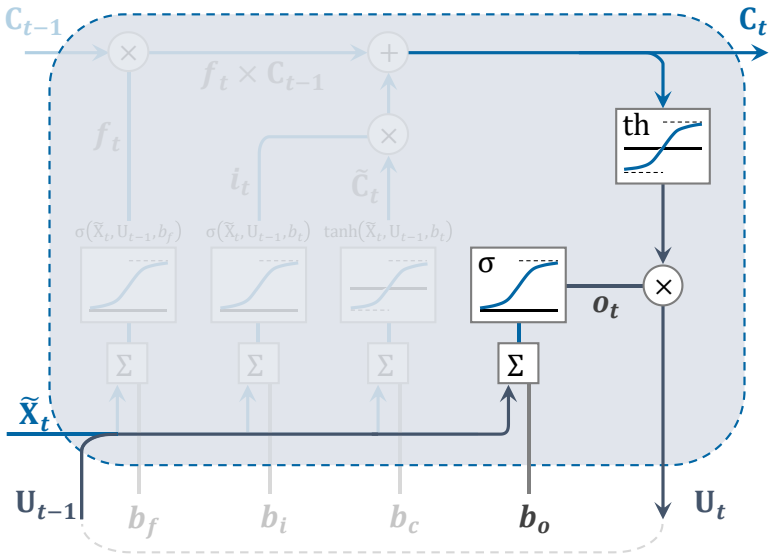


FIGURA 3.14 – Operações e fluxos de informação na célula LSTM (*Output Gate*)

Primeiramente (função da *Output Gate*) a informação é regulada usando a função sigmóide (à semelhança do descrito no 1º Passo), sendo gerado o *output* da *Output Gate*, o_t, descrito em (3.28).

$$o_t = \sigma\left[\boldsymbol{W}_o \cdot \left(\widetilde{\boldsymbol{X}}_t,\ \boldsymbol{U}_{t-1}\right) + b_o\right], \qquad t = 1, \dots, e \tag{3.28}$$

Em seguida, ao vetor de estado da célula gerado no 2º Passo, $\boldsymbol{C}_t$, é aplicada a função tanh, gerando um novo vetor com entradas entre -1 e 1. Finalmente, os valores do novo vetor, $\tanh(\boldsymbol{C}_t)$, e os valores regulados, o_t, são multiplicados para serem guardados como *input* (3.29) a alimentar o processo na próxima iteração,

$$\boldsymbol{U}_t = o_t \times tanh(\boldsymbol{C}_t)\,, \qquad t = 1, \dots, e \tag{3.29}$$

Em suma, as etapas acima descrevem os mecanismos implícitos numa célula LSTM genérica, onde as *gates* ocupam papel preponderante na "atualização" de informação. Sucintamente, a *Forget Gate* decide o que é relevante para evitar as etapas anteriores, a *Input Gate* decide as informações relevantes a adicionar a partir da etapa atual e, por fim, a *Output Gate* determina qual deve ser o próximo estado oculto a incluir como *input* na nova iteração.

Importa, finalmente, destacar que têm surgido na literatura várias variantes de redes LSTM, onde, em particular, o trabalho de Koutník *et al.* (2014) apresenta uma abordagem para lidar com dependências de longo prazo. Contudo, em análises comparativas das variantes mais populares das redes LSTM, Jozefowicz *et al.* (2015) reconhecem que algumas funcionam melhor em tarefas pontuais, enquanto Greff *et al.* (2015) destacam que, na sua globalidade, são praticamente iguais.

3.5.4. Breve Síntese Comparativa

Numa rede neuronal, MLP, entre as camadas de entrada (que recebe o sinal de *input*) e de saída (que devolve o sinal de *output*) existe um número arbitrário de camadas ocultas que, computacionalmente, constituem o elemento potenciador de *Deep Learning* (mediante um número arbitrário de épocas de treino nas quais operam dois fluxos de sinal unidirecionais: um de *feedforward* e um de *backpropagation*).

Já nas RNN a informação persiste, sendo a aprendizagem memorizada no tempo, a curto prazo. Assim, enquanto as redes MLP produzem um modelo estático dos dados, as RNN produzem um modelo dinâmico como resultado do armazenamento de informação. Nas épocas de treino subsequentes a informação guardada em memória da época anterior é combinada com o *input* de dados. O novo *output* gerado não só é enviado num fluxo *feedforward* para os neurónios das camadas seguintes, como é utilizado no passo seguinte.

A topografia LSTM, com a ajuda de *gates*, consegue um armazenamento de memória a longo prazo a qual é utilizada num processo contínuo de aprendizagem. Mais, esta arquitetura de rede consegue distinguir, nos eventos passados, a informação irrelevante da que se mostrou importante, armazenando-a em *gates* distintas (*forget gate* e *input gate*, respetivamente). Este armazenamento seletivo de informação permite à rede recorrente ir aprendendo continuamente ao longo das épocas de treino.

Ou seja, as diferenças no processo de aprendizagem destas três arquiteturas ocorrem essencialmente ao nível do fluxo de informação dentro de cada neurónio das camadas ocultas (célula), como se esquematiza na Figura 3.15.

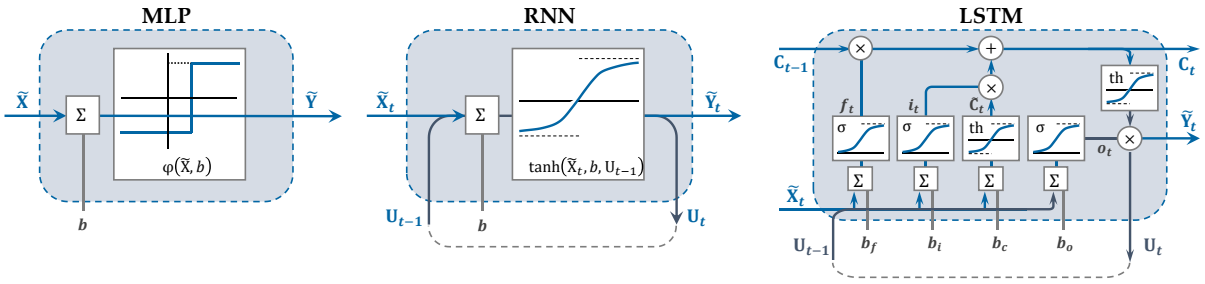


FIGURA 3.15 – Comparação de uma célula oculta MLP, RNN e LSTM

Os aspetos descritos justificam o sucesso das arquiteturas RNN no processamento de dados que, de certa foram, apresentem uma natureza sequencial, como é o caso das séries temporais. As redes LSTM, devido à sua 'capacidade de memória de longo prazo', resolvem ainda o problema de dissipação do gradiente (problema comum em redes recorrentes devido à memória de curto prazo), apesar de poderem apresentar um elevado custo computacional pelo desdobramento temporal-espacial de cada camada oculta.

3.6. DO TREINO DA REDE À AVALIAÇÃO DE MODELOS

Conforme referido na Sec. 3.4, o treino de uma RNA (qualquer que seja a sua arquitetura), é executado recorrendo a um algoritmo de aprendizagem que utiliza um conjunto de dados de treino para ajustar os pesos sinápticos da rede. O objetivo é obter um modelo analítico capaz de solucionar problemas (como, por exemplo, reconhecer padrões ou fazer previsões) a partir de novos conjuntos de dados.

Após o treino, a avaliação do modelo (desempenho e capacidade de generalização da rede) pode ser feita recorrendo a um subconjunto independente de dados de teste.

No caso particular do algoritmo *Backpropagation*,

> *The essence of backpropagation learning is to encode an input/output mapping (represented by a set of labeled examples) into the synaptic weights and thresholds of a multilayer perceptron. The hope is that the network becomes well trained so that it learns enough about the past to generalize to the future. From such a perspective, the learning process amounts to a choice of network parameterization for a given set of data. More specifically, we may view the network selection problem as choosing, within a set of candidate model structures parameterizations, the "best" one according to a certain criterion.*(Haykin, 2009, p. 171)

Neste contexto, o treino da rede neuronal para otimizar o valor dos parâmetros (pesos e *bias*) depende muito da seleção dos hiperparâmetros da rede. Acontece que algumas arquiteturas de DNN (em particular as mais poderosas) apresentam um grande número de hiperparâmetros e quanto menos restrições forem impostas, maior é a probabilidade de ocorrer um ajustamento de 'excessiva qualidade' da rede para os dados de treino. Por outro lado, o próprio critério de paragem do treino, se for demasiado exigente, também pode causar um sobre ajustamento aos dados de treino. Este fenómeno, comummente designado de *overfitting*, pode limitar a capacidade de generalização da rede a outros conjuntos de dados. Em termos de modelação, o *overfitting* leva a que o modelo estimado apresente um ajuste excelente aos dados de treino, por otimização dos pesos e *bias* da rede, mas falhe na capacidade preditiva por o ajustamento a novas observações ser fraco. Desta forma, poder-se-á dizer o modelo estimado, embora tenha "aprendido" bem com os dados de treino, é incapaz de generalizar os resultados para o conjunto de dados de teste que não foram utilizados no processo de treino[76].

Existem vários métodos para evitar este problema, sendo um dos mais utilizados a avaliação da capacidade de generalização da rede durante o próprio treino (Hastie, Trevor, Tibshirani,

[76] A ocorrência de *overfitting* pode ter duas origens: a seleção dos hiperparâmetros da rede e/ou a determinação do critério de paragem do treino. No primeiro caso, o fenómeno pode ocorrer, por exemplo, se for utilizada uma rede neuronal com um número excessivo de camadas ocultas ou de neurónios por camada. No segundo caso, pode ocorrer, por exemplo, se for permitido um número excessivo de iterações. Para mais detalhes veja-se Brownlee (2017).

Robert, Friedman, 2009). Esta técnica designada por validação cruzada, ou *cross-validation*, como é conhecida na literatura científica (Arlot & Celisse, 2010), é muito utilizada em problemas de classificação/regressão, de modelação e previsão.

> *... when evaluating different settings ("hyperparameters") for estimators (...) there is still a risk of overfitting on the training set because the parameters can be tweaked until the estimator performs optimally. This way, knowledge about the training set can "leak" into the model and evaluation metrics no longer report on generalization performance. To solve this problem, yet another part of the dataset [obtida da amostra de treino] can be held out as a so-called "validation set": training proceeds on the training set, after which evaluation is done on the validation set, and when the experiment seems to be successful, final evaluation can be done on the test set.* (Scikit-learn, n.d., Chapter 3)

De modo sucinto, o treino e a avaliação do modelo utilizando a técnica de *cross-validation* são feitos começando por dividir os dados em análise em dois grandes conjuntos: a amostra de treino (para treinar a rede) e amostra de teste (para avaliar o modelo). De seguida, a amostra de treino é dividida em dois subconjuntos disjuntos: um destinado à otimização dos parâmetros do modelo, denominado conjunto de treino, e outro destinado à avaliação da qualidade do treino e da capacidade de generalização da rede no final de cada época de treino, denominado conjunto de validação.

Globalmente, podemos identificar três subconjuntos de dados: dados de treino, dados de validação e dados de teste (designações que utilizaremos daqui em diante), cada um com uma missão e objetivo específico.[77] Especificamente, temos:

- Dados de treino (usualmente integram 60% do total dos dados em análise) – conjunto de dados usados para treinar o modelo, os quais podem ser recolhidos, ou não, sequencialmente ao longo do tempo;
- Dados de validação (cerca de 20% do total dos dados em análise) – conjunto de dados usados para validação do modelo obtido com bom desempenho utilizando os dados de treino. Este conjunto de dados permite uma avaliação imparcial da adequação do modelo

[77] Existem na literatura outras designações atribuídas aos conjuntos/subconjuntos identificados e diferentes abordagens quanto à forma como o conjunto global de dados deverá ser dividido. Em particular para a amostra de treino, veja-se Kearns (1996), por exemplo.

(com possível ajuste dos parâmetros/hiperparâmetros do modelo), mediante o cálculo de uma métrica de erro.

- Dados de teste (cerca de 20% do total dos dados em análise) – conjunto de dados complemente novo para o modelo selecionado, que permite uma avaliação final do modelo mediante o cálculo de uma métrica de erro.

Pela forma como os dados são divididos nos três subconjuntos, e pelos procedimentos internos ajustados em cada caso é possível distinguir diferentes metodologias de *cross-validation*. Contudo, regra geral, todas seguem iterativamente os mesmos procedimentos, esquematizados na Figura 3.16.

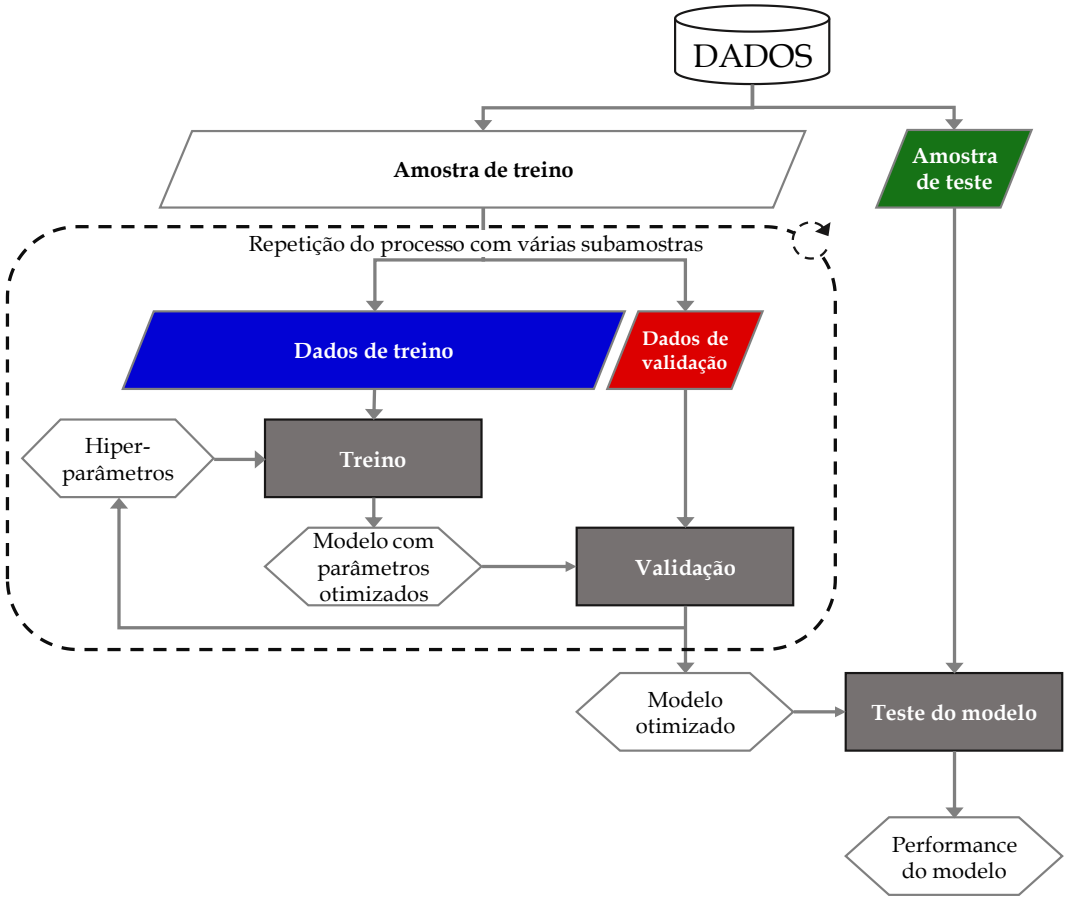


FIGURA 3.16 – Fluxograma relativo à *Cross-validation*

Tal como esquematizado, podem ser enumeradas as seguintes etapas:

(i) Dividir o conjunto de dados em 3 subconjuntos (treino, validação e teste);

(ii) Treinar o modelo no conjunto de dados de treino (obtendo o ajuste ótimo dos parâmetros com base num algoritmo de otimização);

(iii) Validar o modelo no conjunto de dados de validação (procurando a otimização 'manual' dos hiperparâmetros da rede);

(iv) Repetir as etapas **(*ii*)** e **(*iii*)** para diferentes subconjuntos de dados (de treino e validação), para que seja estimado/selecionado um "modelo ótimo"

(v) Avaliar o "modelo ótimo" no conjunto de dados de teste, para concluir sobre a qualidade e *performance* do modelo estimado/selecionado.

A parte matemática inerente ao treino (seleção e posterior validação) do modelo, tem em si uma "beleza" em termos de estrutura e sequência lógica tentadora de apresentação neste trabalho.[78] Ao invés, consideramos mais vantajoso guardar este espaço para refletir sobre: (i) algumas metodologias de *cross-validation*, sugeridas na literatura e usadas nas DNN e (ii) alguns problemas subjacentes à implementação da *cross-validation*, nomeadamente quando o conjunto de dados se trata de uma série temporal. De entre as metodologias de *cross-validation* citadas na literatura, iremos analisar três: *Forward Chaining*, *k-Fold* e *Group k-Fold*.[79]

3.6.1. *FORWARD CHAINING*

A *Forward Chaining* é das metodologias de *cross-validation* mais utilizadas quando é necessário ter em conta a ordem sequencial dos dados em estudo. Neste procedimento é considerada uma divisão sequencial do conjunto de dados em dados de treino, dados de validação e dados de teste. Isto é, o primeiro subconjunto é constituído por observações que ocorreram imediatamente antes das observações que integram o subconjunto de dados de validação que, por sua vez, ocorreram imediatamente antes das observações que integram o subconjunto de dados de teste.

O diagrama seguinte (Figura 3.17) ilustra uma série de seis *sets* de treino, validação e teste que vai percorrendo gradualmente toda a amostra de dados, onde as observações azuis formam os conjuntos de treino (para otimização dos parâmetros do modelo), as observações vermelhas

[78] Para uma análise mais cuidada sobre a parte matemática subjacente ao processo de *cross-validation*, veja-se Haykin (2009, sec. 4.13).

[79] Encontramos na literatura vários métodos de *cross-validation*, onde não só as designações não são unanimes, como os mecanismos internos apresentam ligeiras diferenças. Assim importa sublinhar que a visão apresentada nesta secção acaba por ter uma perspetiva própria. Tendo por base a literatura, a preocupação será descrever as metodologias envolvidas neste estudo e os mecanismos intrínsecos a cada uma.

formam o conjunto de validação (para ajuste dos hiperparâmetros da rede) e , finalmente, as observações a verde formam o conjunto de teste (para avaliação da qualidade do modelo).[80]

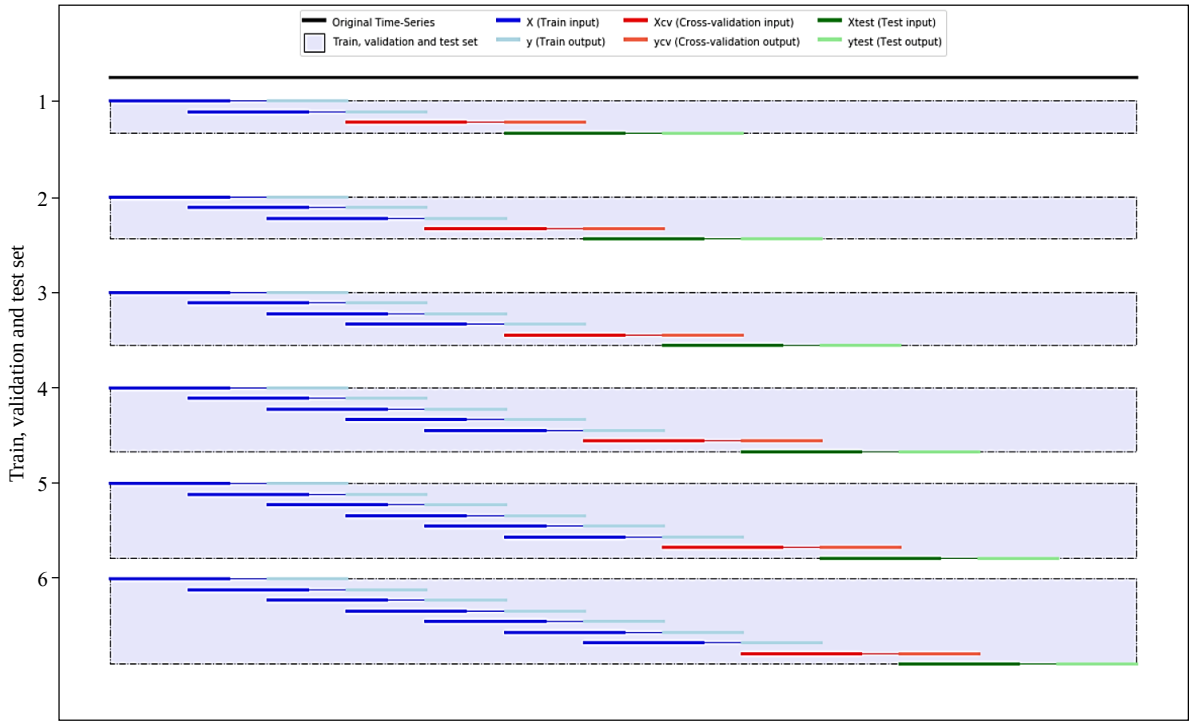


FIGURA 3.17 – *Cross-validation: Forward Chaining*

O afinar da escolha do modelo (ajuste dos parâmetros/hiperparâmetros) é feito com base na precisão dos valores destacados a vermelho no *output* da validação (erro de validação). Definido um modelo, a avaliação final é igualmente feita com base na precisão de previsões, mas agora dos valores relativos ao *output* de teste (valores destacados a verde e que nunca foram vistos antes pelo modelo – erro de teste).

Em qualquer dos casos, a precisão é avaliada a partir de uma métrica de erro (tema analisado mais adiante – Sec. 4.4) que, mediante a acumulação dos erros de validação e dos erros de teste resultantes de cada *set*, pode passar pelo cálculo da respetiva média (do valores absolutos ou dos quadrados, em função da métrica escolhida).

É desejável que não existam grandes diferenças entre os dois valores (erros médios de validação e de teste) e, naturalmente, quanto menores forem melhor será o modelo em termos

[80] Observe-se que não sendo aconselhável obter uma previsão confiável com base num pequeno conjunto de treino, no primeiro *set* a imagem ilustra logo a preocupação de que as primeiras observações que integram o subconjunto de treino devam ser mais (com dois subconjuntos de dados) comparativamente aos subconjuntos de validação e treino.

de qualidade preditiva, podendo falar-se em erro de generalização associado ao modelo. Contudo, em casos em que o erro médio de validação seja muitíssimo baixo e bem menor que o valor do erro médio de teste, será de suspeitar da ocorrência de *overfitting*.

Para finalizar, importa referir que a avaliação do modelo pode depender muito das observações que integram cada um dos três subconjuntos de dados e, portanto, a avaliação pode ser significativamente diferente dependendo do modo como a divisão foi feita. Para minimizar esta subjetividade destacamos a importância de serem considerados vários *sets* (de treino, validação e teste). Ao ser feita uma avaliação com base numa média obtida de vários *sets*, esta reverte-se de uma maior robustez. Assim, deve ser encontrado um equilíbrio entre o número de observações que deve integrar os vários subconjuntos e o tempo computacional intrínseco a esta etapa.[81]

3.6.2. *K-FOLD*

Um segundo processo de *cross-validation*, designado *k-Fold*, apresenta uma forma alternativa para a seleção e posterior avaliação do modelo, sendo especialmente indicado para dados iid. Nesta metodologia, o conjunto de dados é dividido em k subconjuntos com igual dimensão.[82] Cada vez que um dos subconjuntos é arbitrariamente usado como conjunto de teste, outros $k-2$ subconjuntos constituem o conjunto de treino e o outro constitui o conjunto de validação. Assim, como ilustrado na Figura 3.18, e usando a mesma notação de cores da figura anterior, o modelo é treinado com todos os subconjuntos exceto dois, um é usado para validação e o outro como conjunto de teste para avaliar o modelo.

À semelhança do descrito no caso anterior a validação e o teste são feitos com base nas precisões dos respetivos *outputs*, sendo o desempenho do modelo avaliado pela média dos respetivos erros (de validação e de teste) obtidos sobre todos os *sets*.

A vantagem desta metodologia relativamente à *Forward Chaining* assenta na forma como os dados são divididos, uma vez que cada subconjunto de dados será também um conjunto de teste (num dado *set*). Para finalizar, importa referir que a variação das estimativas relativas aos

[81] Existe um parâmetro relativo à *cross-validation* que pode ser ajustado e que respeita à possibilidade de se ir dando "saltos" na amostra e como tal não integrar todo o conjunto de dados disponível nesta etapa.

[82] Nesta divisão pode ocorrer que o número total de observações, n, não seja divisível por k. Nesse caso, o último subconjunto não contém um número suficiente de dados para constituir um subconjunto a considerar no processo, pelo que uma possível solução será não incluir estes dados no processo.

erros resultantes será tão mais reduzida quanto maior for o valor de *k*, sendo a desvantagem desta metodologia o facto de requerer uma quantidade excessiva de cálculos à medida que aumenta o número de subconjuntos considerado.

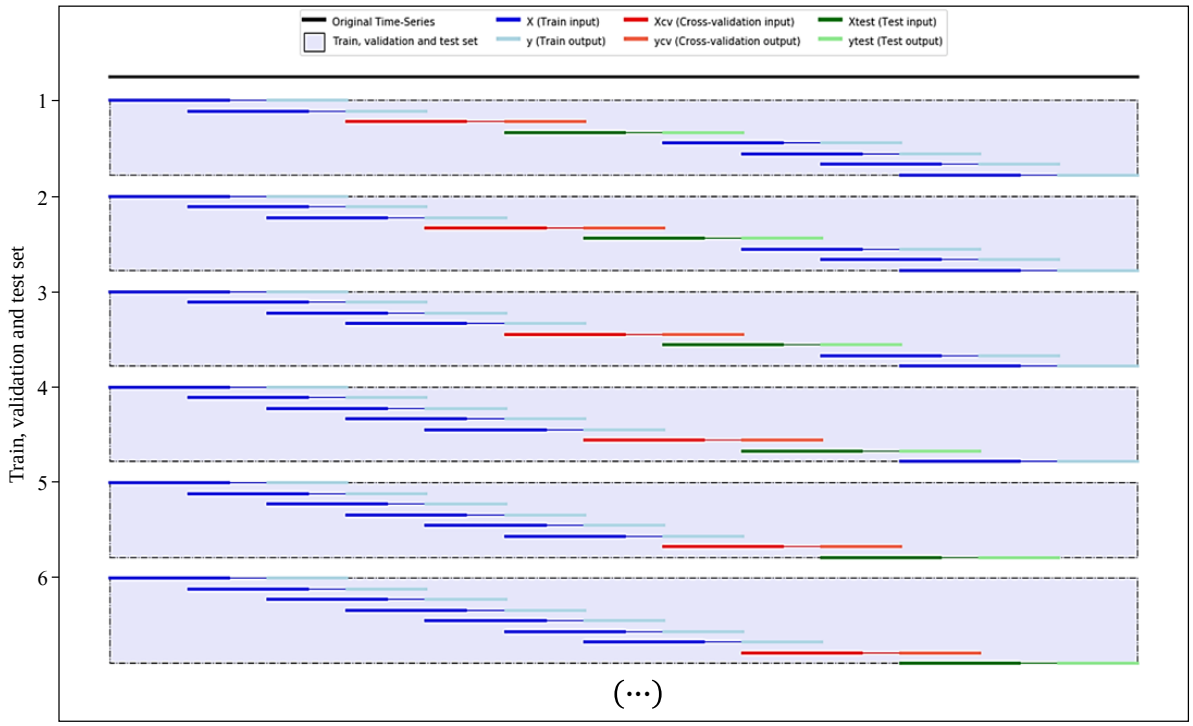


FIGURA 3.18 – *Cross-validation: k-Fold*

3.6.3. *GROUP K-FOLD*

Uma variante da metodologia *k-Fold*, é usualmente designada de *Group k-Fold*.[83] Esta metodologia apresenta alguma flexibilidade perante a dependência (temporal) nos dados da amostra. Em cada *set*, os dados são considerados de forma sequencial e, após cada treino, segue-se a validação e posterior avaliação, podendo tal ocorrer várias vezes num mesmo *set*, como ilustrado na Figura 3.19.

O conjunto de dados é dividido em *k* subconjuntos de igual dimensão. Porém, contrapondo o que acontece no caso *k-Fold* (onde $k-2$ subconjuntos constituem o subconjunto de dados de treino e os restantes dois correspondem aos subconjuntos de validação e de teste), neste caso é definido um parâmetro que dita que, por cada número fixo de subconjuntos de treino, é

[83] Em Scikit-learn (n.d., Chapter 3) são apresentadas outras variantes, como *Repeated k-Fold*, *Leave One Out* (mais indicadas para dados *iid*), ou outras alternativas usando amostragem estratificada (*Stratified k-Fold*).

considerado sequencialmente um subconjunto de validação e um subconjunto de teste. Assim, em cada *set*, pode haver um número diferente de "validações" e "testes", sendo a precisão avaliada segundo o cálculo de uma média (erro de validação do *set* e erro de teste do *set*).

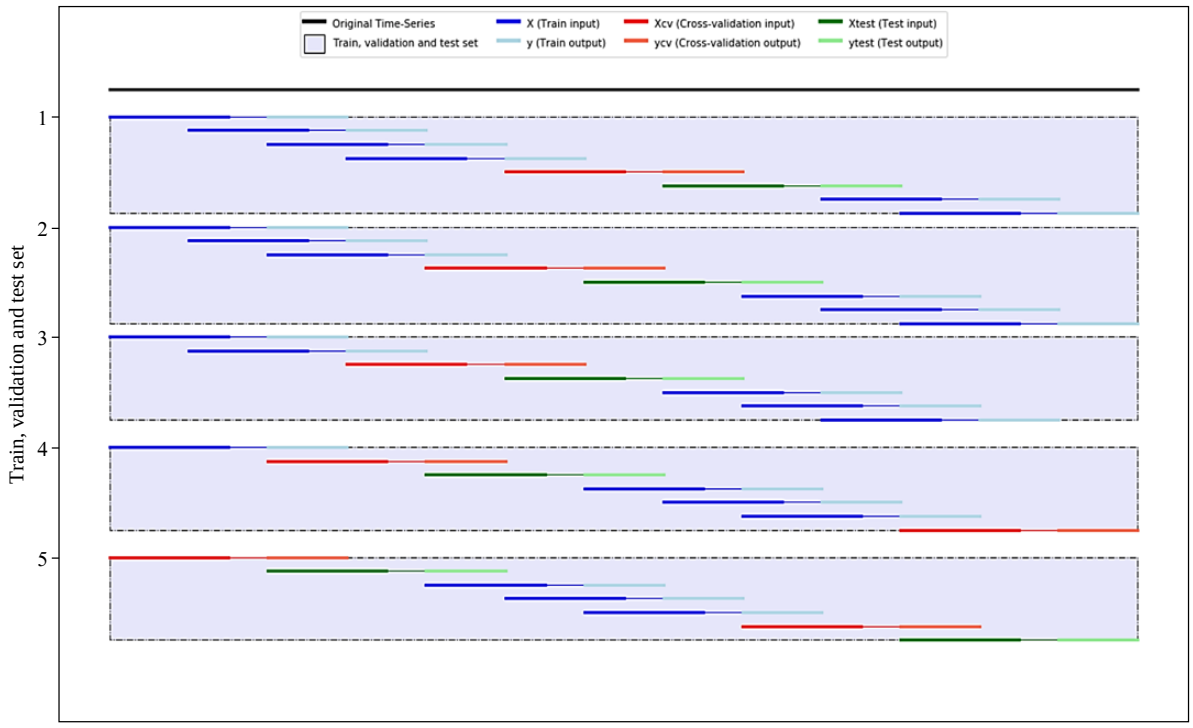


FIGURA 3.19 – *Cross-validation: Group k-Fold*

À semelhança do descrito nos casos anteriores, a validação e o teste são feitos com base nas precisões dos respetivos *outputs*, sendo o desempenho do modelo avaliado pela média dos respetivos erros (erro de validação e erro de teste) obtidos sobre todos os *sets*.

Numa nota final sobre esta metodologia queremos, ainda, destacar que uma grande vantagem relativamente à *k-Fold*, está numa maior rapidez computacional.

3.6.4. CROSS-VALIDATION EM SÉRIES TEMPORAIS

A natureza dos dados em estudo deve ser tida em conta na escolha de metodologia de *cross-validation* a usar, em paralelo com as vantagens e desvantagens de cada uma delas.

Procurando sintetizar alguma informação, apresenta-se na tabela seguinte (Tabela 3.1) uma síntese comparativa para as três metodologias acima descritas: *Forward Chaining*, *k-Fold* e *Group k-Fold.*

TABELA 3.1 – Análise comparativa de metodologias de *cross-validation*

	Vantagens	Desvantagens
Forward Chaining	• Adequabilidade a dados de natureza sequencial • Execução e tempo computacional (moderado/baixo)	• Nem todos os *sets* têm em conta a totalidade do conjunto de dados • Alguma variabilidade no desempenho (entre *sets*)
k-Fold	• Todos os *sets* têm em conta a (quase) totalidade do conjunto de dados • Não é expectável grande variabilidade no desempenho (entre *sets*)	• Não indicada para dados de natureza sequencial (requer que os dados sejam iid) • Execução e tempo computacional (elevada)
Group k-Fold	• Todos os *sets* têm em conta a (quase) totalidade do conjunto de dados • Não é expectável grande variabilidade no desempenho (entre *sets*) • Execução e tempo computacional (baixo)	• Aplicabilidade preferencial para dados iid (embora exista alguma flexibilidade para dados de natureza sequencial)

Face a estas considerações, quando o objetivo passa pelo estudo de séries temporais, apesar de ter reconhecidas desvantagens, a metodologia *Forward Chaining* parece ser a que apresenta uma maior adequabilidade por respeitar a ordem sequencial dos dados, os quais são (em geral) caracterizados por uma correlação temporal.[84] Isto porque, embora dados iid seja uma suposição comum em *Machine Learning*, isso raramente se verifica em séries temporais, pelo que as outras metodologias apresentadas podem apresentar limitações. Como é referido em Scikit-learn (n.d., Chapter 3),

> *…if one knows that the samples have been generated using a time-dependent process, it's safer to use a time-series aware cross-validation scheme. (…) Classical cross-validation techniques such as k-Fold (…) assume the samples are*

[84] Por essa razão, muito autores identificam esta metodologia como *Time Series Split*.

> *independent and identically distributed, and would result in unreasonable correlation between training and testing instances (yielding poor estimates of generalization error) on time series data.*

Contudo, havendo uma alternativa que permitisse minimizar a presença de autocorrelação nos dados, a metodologia *Group k-Fold* parece ser a que reúne maior vantagem, sobretudo pelo facto de: (i) em cada *set*, ser tida em conta a (quase) totalidade dos dados, o que se reverte de vantagens em termos de variabilidade no desempenho e (ii) o tempo de execução computacional não ser excessivo (comparando com o *k-Fold*).

Finalizando, importa sublinhar a importância dos factos apresentados, dado que a implementação da *cross-validation*, bem como a identificação da metodologia mais adequada, reverte-se de extrema importância no assegurar da estimação e escolha adequada do modelo a usar para fazer previsões, já que, numa fase intermédia do estudo, podemos avaliar o seu desempenho e a sua capacidade de generalização face ao conjunto de dados.

4. CONSIDERAÇÕES METODOLÓGICAS

Tal como já referido (Sec. 2.1), numa perspetiva de modelação/previsão de séries temporais, o nosso estudo incide numa análise univariada que passa pela exploração/implementação computacional de modelos com o objetivo de prever valores futuros de variáveis extrapolando o seu comportamento passado.

Neste capítulo, além de serem apresentadas as séries que servirão de base à parte empírica do nosso estudo, procurar-se-á: (1) explicar os procedimentos metodológicos adotados na investigação (particularmente os de implementação computacional); (2) acrescentar algumas considerações teóricas complementares aos capítulos anteriores (nomeadamente ao nível do pré-processamento dos dados e das métricas usadas para avaliar o erro de previsão).

4.1. BASES DE DADOS (SÉRIES TEMPORAIS)

Para a concretização da parte empírica da investigação foram consideradas quatro séries temporais com padrões distintos.

No contexto dos mercados financeiro, tendo em conta os estudos primários feitos com a série de dados históricos relativos ao principal índice da bolsa portuguesa, o PSI 20 (Ramos, Costa, Mendes, & Mendes, 2018), e com o correspondente índice norte-americano, o SPY (Costa, Ramos, Mendes, & Mendes, 2019), mantivemos estas duas séries até ao fim na nossa investigação.

Contudo, procurando enriquecer o estudo (e dar resposta a alguns dos objetivos propostos) com a implementação das metodologias descritas nos capítulos anteriores (2 e 3), impunha-se a escolha de outras séries temporais que, à primeira vista, apresentassem comportamentos distintos de tendência e sazonalidade.

Desta forma, de uma pesquisa em bases de dados disponíveis (atualizadas diariamente) e consideradas fiáveis, a nossa opção recaiu sobre quatro séries temporais: (1) *Consumer Price Index for All Urban Consumers: All Items in U.S. City Average* (CPIAUCSL); (2) *Vehicle-Miles Travelled* (VMT); (3) *Portuguese Stock Index 20* (PSI 20) e (4) *Standard & Poor's 500 Exchange-Traded Fund* (SPY).

Sumariamente:

- **CPIAUCSL** é o índice de preços ao consumidor norte-americano, que avalia a variação mensal média no preço de bens e serviços pagos pelos consumidores urbanos, incluindo cerca de 88% da população (mais detalhes ver Anexo D.1);

- **VMT** reflete a evolução mensal do volume de tráfego/milhas viajadas nos Estados Unidos, através de valores criados anexando os dados mensais recentes de *Traffic Volume Trends* (dados da *Federal Highway Administration*)[85] aos dados históricos mensais de *Vehicle Miles Traveled*;

- **PSI 20** diz respeito ao principal índice de referência do mercado de capitais português que agrega as 20 maiores empresas cotadas na *Euronext Lisboa*, refletindo a evolução dos preços das respetivas ações;

- **SPY** é um fundo de investimento cuja carteira de ativos inclui as ações das empresas incluídas no cálculo do índice *Standard & Poor's* 500 (S&P 500), tratando-se assim de um *Exchange-Traded Fund* (ETF) que replica o referido índice. O valor de uma unidade de participação do fundo corresponde a 1/10 do valor do índice S&P 500, que é composto por quinhentos ativos (selecionados pela sua representatividade e ponderados pelo seu valor de mercado) negociados nas duas principais bolsas norte-americanas: NYSE (*New York Stock Exchange*) e NASDAQ (*National Association of Securities Dealers Automated Quotations*).

Procurando usar dados disponíveis a toda a comunidade científica, na eventualidade de estudos comparativos posteriores, procurámos que as quatro séries temporais referidas fossem lidas diretamente do *Yahoo Finance* e do *FRED® Economic Data*, usando a biblioteca `pandas_datareader` (*Python*). A grande vantagem de utilizar esta biblioteca está no facto

[85] A tendências de volume de tráfego é um relatório mensal baseado na contagem de circulação de veículos por hora. Os dados são recolhidos em cerca 5000 locais de contagem contínua de viaturas, sendo depois usados para estimar a variação percentual do trânsito no mês atual relativamente ao período homólogo do ano anterior. As estimativas são reajustadas anualmente para corresponderem às *Vehicle Miles Travelled* a partir do *Highway Performance Monitoring System* e são atualizadas continuamente com dados adicionais (informação disponível em https://www.fhwa.dot.gov/policyinformation/travel_monitoring/tvt.cfm).

de (a princípio) os dados estarem 'tratados', isto é, não haver valores em falta, valores duplicados ou erros de formatação, entre outros.[86]

4.2. PRÉ-PROCESSAMENTO E TRANSFORMAÇÃO DOS DADOS

Antecedendo a aplicação de algumas metodologias relativas à modelação e previsão em séries temporais (como as exploradas neste trabalho na implementação dos modelos ARMA, ETS ou DNN) devem ser tidos em conta alguns procedimentos de pré-processamento e/ou transformação efetiva dos dados, por serem um requisito (como a estabilização da série, nos modelos ARMA) ou por existirem benefícios reconhecidos na sua execução (como a normalização, nos modelos DNN). Vejamos alguns exemplos.

4.2.1. SUAVIZAÇÃO

Como mencionado anteriormente, algumas séries temporais são bastante voláteis e, como tal, estão sujeitas a uma quantidade considerável de 'ruído' e flutuações, algo que dificulta a tarefa de previsão.

O alisamento das observações passadas é uma das estratégias empregues na prática para separar/eliminar o 'ruído', facilitando a identificação de ciclos de tendência(s), o que se mostra adequado em situações onde se pretende efetuar previsões.[87] Alguns dos algoritmos comummente referidos na literatura são baseados em *Moving Average* (MA), seja simples (*Simple Moving Average* – SMA), ponderada (*Weighted Moving Average* – WMA), exponencial (*Exponential Moving Average* – EMA), ou uma combinação destas duas últimas vertentes (*Exponentially Weighted Moving Average* – EWMA).

De entre as quatro hipóteses para a suavização de dados, alguma bibliografia aponta para o facto de, em séries onde existam 'mudanças' de regime (o que acontece em três dos casos estudados, incluindo as constantes oscilações no caso da série VMT), a suavização segundo os

[86] No entanto, deve ser feita sempre uma verificação às bases de dados. Por exemplo, neste processo de verificação, constou-se que a série relativa ao PSI 20, lida do *Yahoo Finance*, apresentava longos períodos com ausência de dados, além de que só havia informação disponível desde finais de 2010. A propósito deste facto, com vista a utilizar uma base de dados fidedigna, a opção passou por exportar a informação em formato '*.csv*' e lê-la diretamente do ficheiro exportado.

[87] Por exemplo, no caso dos modelos DNN, este tipo de procedimento é comumente empregue para acelerar o treino da rede neuronal.

EMA reage mais rapidamente e com melhor ajuste comparativamente aos SMA. Já os EWMA partem dos WMA, na medida em que se pode dar maior peso a dados mais recentes, mas diferem pelo facto de não descartarem informação à medida que o tempo passa. Assim, os EWMA permitem que cada observação passada se torne progressivamente menos significativa, mas ainda assim seja incluída (Kingsman, 2004). Pelos factos expostos, a nossa opção para suavizar os dados, foi a implementação computacional do EWMA.

Matematicamente, conforme descrito em (4.1), o EWMA baseia-se na obtenção do valor da média móvel no instante $t \in T$, S_t, a partir do valor da série observado nesse instante, y_t, e do valor anterior da própria média móvel, S_{t-1}, em que o coeficiente de ponderação, $0 < \alpha < 1$, determina o peso a dar às informações passada e presente[88]

$$S_t = \begin{cases} y_1 & , \ t = 1 \\ \alpha y_t + (1-\alpha) S_{t-1} & , \ t > 1 \end{cases} \tag{4.1}$$

4.2.2. Estabilização das Séries Temporais

Uma série temporal não estacionária pode ser sujeita a um processo de estabilização, mediante transformações adequadas, podendo resultar numa nova série estacionária (tema já discutido na Sec. 2.2.2). Daremos conta nesta secção de técnicas que nos permitem estabilizar a média e a variância da série temporal. Em termos práticos, quando necessário aplicar os dois procedimentos, deve começar-se por estabilizar a variância/covariância e (só) em seguida estabilizar a média.[89]

4.2.2.1. Estabilização da Variância

Dada uma série temporal $\{y_t\}_{t\in T}$, supondo-se conhecidas a média e variância (ambas finitas), $\mathbb{E}(\mathrm{y_t}) = \mu_t$ e $Var(y_t) = \sigma_t^2, \ \ \forall\, t \in T$, em que a variância da série é função da sua média, isto é, para uma certa constante $k \in \mathbb{R}^+$ e função h: $\mathbb{R} \to \mathbb{R}^+$

[88] Para valores de α pequenos, a maior contribuição virá dos valores passados, o que implica uma suavização forte. Já para valores significativos de α, a maior contribuição virá do valor atual e a suavização será mínima.

[89] Isto porque a aplicação do operador diferença, comumente usado para estabilizar a média, pode fazer aumentar a variância de uma série e tornar mais pesada a sua estrutura e autocorrelação.

$$\sigma_t^2 = k\, h(\mu_t)\ ,\ \forall\, t \in T \tag{4.2}$$

o objetivo será encontrar uma transformação, $\mathbb{T}$, tal que

$$Var[\mathbb{T}(y_t)] \cong c,\ c \in \mathbb{R}^+ \tag{4.3}$$

Numa primeira abordagem, uma forma utilizada habitualmente para estabilizar a variância de séries temporais consiste na transformação logarítmica natural dos dados

$$\mathbb{T}(y_t) = log\ y_t \tag{4.4}$$

obrigando apenas a que os valores da série sejam positivos.[90]

Um potencial entrave à aplicação desta metodologia é o facto de a transformação requer que a variância seja função da média. Ora, na prática, mesmo que exista uma relação funcional entre a média e a variância da série, esta é geralmente desconhecida.

Alternativamente, quando a variância não é função da média ou se desconhece a relação funcional entre as duas, mas se conhece uma representação da sua evolução no tempo (por exemplo), o método mais utilizado para estabilizar a variância da série consiste em recorrer à família de transformações de *Box-Cox*. Assumindo $\{y_t\}_{t\in T}$ de valores positivos, a transformação de *Box-Cox* de parâmetro $\lambda \in \mathbb{R}$, usualmente em $[-1, 1]$, é descrita por

$$\mathbb{T}(y_t) = \begin{cases} \dfrac{y_t^{\lambda} - 1}{\lambda}\ , & \lambda \neq 0 \\ log\ y_t\ , & \lambda = 0 \end{cases} \tag{4.5}$$

onde transformação logarítmica é introduzida para $\lambda = 0$ de modo a que a família de transformações $\mathbb{T}$¸ seja contínua.[91]

Em termos de implementação computacional a transformação em causa foi executada com recursos à função `boxcox` da biblioteca `sciPy` (*Python*).[92]

[90] Note-se que, na prática, tal não constitui qualquer entrave, uma vez que é sempre possível adicionar uma constante positiva aos valores observados de modo a torná-los positivos.

[91] De acordo com a relação entre o valor da média e da variância, podemos particularizar ainda as comummente designadas de transformações de raiz quadrada, $T(y_t) = \sqrt{y_t}$, ou do valor inverso, $T(y_t) = 1/y_t$.

[92] Ver https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.boxcox.html.

4.2.2.2. ESTABILIZAÇÃO DA MÉDIA

Dada uma série temporal $\{y_t\}_{t\in T}$, considere-se a série $\{z_t\}_{t\in T}$ resultante da transformação de *Box-Cox* (estabilização da variância) descrita acima. Em séries de natureza financeira, é comum que a nova série, $\{z_t\}_{t\in T}$, possua uma tendência que deve ser removida de modo a estabilizar a média da série. Como faz sentido considerar que a tendência é aleatória podendo, em cada instante, ser influenciada pela sua evolução estocástica no passado, uma possibilidade para a remover passa pela aplicação de um filtro linear da forma

$$\sum_{j=0}^{\infty} \alpha_j \, z_{t-j} \ , \quad \forall \, t \in T \tag{4.6}$$

com $\sum_{j=0}^{\infty}|\alpha_j|$ finito.

Exemplo deste tipo de operadores é o caso dos operadores de diferença, simples ou com sazonalidade (de ordem s), respetivamente definidos por

$$\Delta z_t = z_t - z_{t-1} \quad \text{e} \quad \Delta_s z_t = z_t - z_{t-s} \ , \quad \forall \, t \in T \tag{4.7}$$

Sobre o operador diferença[93] interessa observar que permite remover uma vasta gama de tendências de séries temporais, podendo ser necessário aplicá-lo mais do que uma vez. No entanto, apenas o número de vezes estritamente necessário para estabilizar a média da série.

Para finalizar, sublinhe-se que as transformações dos dados originais num novo conjunto de dados (que servirá de base a todo o trabalho experimental/computacional) têm de ser revertidas aquando da discussão e interpretação dos resultados.

4.2.3. NORMALIZAÇÃO

O processo de normalização não é mais que um redimensionamento/reescalonamento dos dados originais. Além dos benefícios no que respeita ao processo de previsão (Brownlee, 2017), algumas metodologias, em particular as que passam pela implementação das arquiteturas de redes neuronais, requerem mesmo a normalização/padronização dos dados de forma a acelerar a convergência na etapa de treino.[94]

[93] Tema a que já nos referimos na Sec. 2.2.3, pelo que nos escusamos a maiores desenvolvimentos.

[94] Embora a normalização e a padronização sejam úteis quando os dados têm escalas diferentes, a normalização é a técnica a aplicar quando se desconhece a distribuição dos dados (os algoritmos utilizados não fazem suposições

Quando claramente identificado o conjunto de dados históricos, como no estudo de séries temporais, onde se obtém facilmente algumas das estatísticas descritivas básicas (como o mínimo, o máximo, a média e o desvio-padrão, por exemplo), uma alternativa implementada que permite a **normalização** dos dados (nessa abordagem, os dados são escalonados para um intervalo fixo, geralmente entre 0 e 1) é descrita pela equação

$$z_t = \frac{y_t - y_{min}}{y_{max} - y_{min}}, \quad \forall\, t \in T \tag{4.8}$$

onde y_{max} e y_{min} denotam, respetivamente, o máximo e o mínimo dos valores observados da série $\{y_t\}_{t\in T}$.[95]

Se além da normalização for pretendida ou requerida a normalização padronizada (**padronização/estandardização**) dos dados, isto é, o redimensionamento da distribuição de valores para que a média seja 0 e o desvio-padrão seja 1 (distribuição normal padrão), a implementação computacional passa a ser descrita pela equação,

$$z_t = \frac{y_t - \bar{y}}{s}, \quad \forall\, t \in T \tag{4.9}$$

onde $\bar{y}$ denota a média e s o desvio-padrão da série temporal.

Em termos de implementação computacional, à semelhança do descrito na secção anterior, sublinhe-se a necessidade de reverter as transformações efetuadas aquando da etapa de discussão e interpretação dos resultados.

Para finalizar este tópico, importa observar que, perante uma série com um passado com fortes irregularidades e onde o presente (dados recentes) apresenta uma monotonia acentuada (crescente ou decrescente), estimar/identificar os valores reais que possam vir a corresponder aos extremos do intervalo padronizado pode ser difícil e a normalização pode não ser o melhor método para o problema. Como tal, este pré-processamento de dados deverá ser utilizado com alguma precaução e testado, averiguando-se as suas vantagens.

sobre essa distribuição), enquanto a padronização (técnica mais eficaz) é aplicável se a distribuição for Gaussiana (os algoritmos usados pressupões que os dados tenham distribuição Gaussiana).

[95] Dada uma série temporal, $\{y_t\}_{t\in T}$, a utilização dos "log-retorno", que opera com a diferença entre os dados logaritmizados, $z_t = log\, y_t - log\, y_{t-1}$, $\forall\, t \in T$, é também uma opção frequentemente usada.

4.3. IMPLEMENTAÇÃO COMPUTACIONAL

Na investigação em curso requer-se uma forte componente computacional que, na procura de modelar os dados reais e efetuar previsões, se exige que apresente uma estrutura robusta. Para o efeito, além da eficácia, apresentar um conjunto rotinas completo (não sendo necessário recorrer a procedimentos externos) e o mais simplificado possível em termos de processamento, foi um desafio. Neste sentido, com vista a futuras utilizações, pensou-se a construção de um código de leitura simples o que, em alguns casos, levou ao desenvolvimento de algumas rotinas/funções de forma a torná-lo mais compreensível e compacto.

Para que tal processo fosse conseguido com sucesso, fez-se um levantamento prévio de que *softwares* e/ou linguagens de programação poderiam ser usadas. Embora numa fase inicial da nossa investigação tenha sido utilizada a linguagem de programação e ambiente de *software* ***R***, derivado dos fortes desenvolvimentos computacionais recentes e potencialidades da linguagem de programação *Python*, optámos por uma mudança e por fazer toda a implementação computacional usando esta última linguagem de programação.

Com efeito, e em linhas gerais, utilizou-se o ambiente computacional *Jupyter Notebook*[96], sendo a sua escolha para a linguagem de programação *Python* (em específico a versão 3.7.3) justificada por: (i) vasta informação disponível que facilita a aprendizagem e compreensão; (ii) simplicidade de utilização; (iii) grande 'comunidade' utilizadora, o que não só significa fácil acesso a soluções para problemas encontrados, mas também suporte a um vasto número de bibliotecas abertas.

Na verdade, estas bibliotecas otimizadas são a principal razão para o uso desta linguagem visto que, não só não nos poupam tempo para implementar os modelos e testes estatísticos, como também nos permitem focar na otimização do código e respetiva interpretação. Alguns exemplos de bibliotecas são: `numpy` (que permite, de modo eficiente, tratar dados matemáticos); `pandas` (que permite estruturar e analisar dados de modo mais simples); `statsmodels` (que permite a implementação de diversos testes estatísticos); `matplotlib` (associado a ferramentas de visualização) e, finalmente, `tensorflow` (que permite, de modo rápido e eficiente, implementar redes neuronais).

[96] O Projeto *Jupyter* (https://jupyter.org/), cujo nome deriva do facto de, inicialmente, suportar as linguagens de programação: *Julia*, *Python* e ***R***, foi criado para desenvolver *open-source* e serviços para computação interativa em dezenas de linguagens de programação. Atualmente suporta mais de 40 linguagens de programação. Esta *framework* possibilita a criação de documentos, *Jupyter Notebooks*, o que, de modo fácil, permite converter o código para HTML ou PDF e suporta um vasto número de bibliotecas abertas e otimizadas.

Procurando estruturar o raciocínio, pensou-se esta implementação dividida em vários *notebooks*:

- *ExploratoryDataAnalysis.ipynb*, onde constam a parte respeitante à análise exploratória (técnicas descritivas e inferenciais) das séries temporais em estudo;
- *ARIMA.ipynb*, que inclui a parte respeitante à implementação dos modelos ARIMA (incluindo a análise à estacionariedade da série e validação do modelo);
- *SARIMA.ipynb*, onde é feita a implementação dos modelos SARIMA. Não só por envolver uma estrutura um pouco mais complexa, e com um custo (tempo) computacional considerável, como também por analisar com detalhe a componente de sazonalidade da série, foi projetado separadamente dos restantes modelos pertencentes à família de modelos ARMA;
- *ExponenTialSmoothing.ipynb*, relativo à implementação da família de modelos ETS;
- *DeepNeuralNetwork.ipynb*, que inclui as rotinas referentes aos modelos DNN
- *DNN_OurApproach.ipynb*, onde, partindo do *notebook DeepNeuralNetwork.ipynb*, é apresentada uma abordagem própria resultante de várias experiências no sentido de obter um modelo de DNN 'melhorado'.

Todas as rotinas computacionais foram originalmente implementadas para este trabalho e poderão ser consultadas em Lopes e Ramos (2020).

Sobre o trabalho desenvolvido, importa referir que, beneficiando do desenvolvimento computacional acima referido, procurámos compilar, numa estrutura única, informação e procedimentos já existentes, melhorando a sua eficácia, automatização e facilidade de implementação. Este foi precisamente um outro desafio a que nos propusemos já no decorrer da investigação, para sucesso do qual houve necessidade de recorrer a bibliografia técnica[97] determinante para a concretização de um trabalho interdisciplinar, conjugando diferentes valências: de um lado, competências matemáticas e estatísticas, do outro, competências informáticas e computacionais.

Vejamos algumas considerações relevantes, nomeadamente no que diz respeito à análise exploratória das séries e à implementação dos modelos ARIMA, ETS e DNN.

[97] Por exemplo Müller & Guido (2016), Brownlee (2017), Chollet (2017) ou, mais recentemente, Raschka & Mirajalili (2019) e Ravichandiran (2019).

4.3.1. ANÁLISE EXPLORATÓRIA DAS SÉRIES TEMPORAIS

No que respeita ao estudo empírico, a etapa inicial passa pela análise e caracterização das séries temporais. Essa análise envolve, especificamente, o recurso a representações gráficas, a decomposição das séries, a análise de existência de quebras de estrutura, bem como a determinação de medidas descritivas e a implementação de alguns testes estatísticos (por exemplo, normalidade, estacionariedade e independência).

Em termos de implementação computacional, os procedimentos seguidos são os que constam no *notebook* "*ExploratoryDataAnalysis.ipynb*", onde entre outras bibliotecas *Python* usadas, destacamos `matplotlib`, `statsmodels` e `scipy`.

Além da análise descritiva das séries temporais, com a determinação de medidas descritivas e implementação de testes de hipóteses (usando classes das bibliotecas acima referidas), destacamos alguns detalhes no que respeita a outros procedimentos de análise exploratória utilizados.

Decomposição das Séries Temporais

No seguimento do descrito na Sec. 2.2.1 e na Sec. 2.4.2, para a implementação de alguns métodos de previsão, em particular os ETS, é útil reconhecer a presença de comportamentos de tendência e de sazonalidade e de qualquer "ruido" presente nas observações.

Tal como referido em Hyndman e Athanasopoulos (2018), existem diversas alternativas para isolar cada componente da série (Sec. 2.2.1), sendo uma delas a designada "decomposição clássica", envolvendo a metodologia de médias móveis.[98] Apesar das fragilidades apontadas a esta abordagem de decomposição (do ponto de vista estatístico e de objetividade), a literatura aponta alguns sucessos nessa prática, como referido em fontes já citadas (Sec. 2.4.2) que envolvem previsão utilizando modelos de ETS.

No nosso trabalho considerámos a decomposição aditiva e multiplicativa da série temporal. Recorremos à metodologia de médias móveis para extrair a sazonalidade, S_t, e ao *Hodrick–Prescott filter*[99] para remover a componente ciclo de tendência, CT_t, a partir dos dados 'brutos'

[98] Além da "decomposição clássica", são propostas na literatura outras alternativas, tais como a *X11 Decomposition* e a SEATS, ambas discutidas com detalhe em Dagum e Bianconcini (2016); ou ainda a STL desenvolvida por Cleveland *et al.* (1990).

[99] Para mais detalhes, ver Hodrick e Prescott (1997).

utilizando, respetivamente, as funções `seasonal_decompose` e `hpfilter` da biblioteca `statsmodels` (*Python*).[100] Relativamente ao ajuste da sensibilidade da tendência, este foi definido segundo um parâmetro, λ, seguindo as indicações metodológicas referidas em Ravn e Uhlig (2002), com valores deduzidos e testados por nós para cada uma das séries em estudo (em função da frequência dos dados).

Existência de Quebras de Estrutura

Tal como referido na Sec. 2.2.6, existem várias possibilidades para estudar a existência de quebras de estrutura/mudanças abruptas de regime numa série temporal. Depois de testar algumas delas, optámos pelo recurso ao algoritmo CUSUM.

Relativamente à implementação em *Python* foi usada a função `detect_cusum` desenvolvida por Duarte e Watanabe (2018) e onde pode ser consultada informação relativa ao algoritmo, parâmetros envolvidos e uma explicação sobre as representações gráficas obtidas da implementação computacional.

Estacionariedade (abordagem gráfica e testes de hipóteses)

Na Sec. 2.2 (em particular nas secções 2.2.2, 2.2.4 e 2.2.5) foram feitas várias considerações sobre a forma como podemos estudar a estacionariedade de uma série temporal (abordagem gráfica e com recursos a testes de hipóteses).

Assim, além da implementação em *Python* dos testes hipóteses referidos na Sec. 2.2.5, fornecida na biblioteca `statsmodels` por meio das classes `adfuller` e `kpss`, neste trabalho a análise de estacionariedade baseou-se também na representação gráfica *Rolling mean* e *Rolling std* (complementarmente à da própria série e correspondente correlograma).

Os valores de *Rolling mean* traduzem uma série de médias de subconjuntos da série original (completa). Em termos analíticos, dada uma série temporal $\{y_t\}_{t\in T}$ e uma dimensão fixa, k, para os subconjuntos, o cálculo desta medida não é mais do que um cálculo de médias, onde, cada subconjunto é obtido excluindo o primeiro elemento do subconjunto anterior e incluindo o próximo valor da série, conforme descrito abaixo (para o caso de médias móveis simples, ou aritméticas)

[100] Ver https://www.statsmodels.org/stable/generated/statsmodels.tsa.seasonal.seasonal_decompose.html e https://www.statsmodels.org/stable/generated/statsmodels.tsa.filters.hp_filter.hpfilter.html.

$$\bar{y}_{1,k} = \frac{1}{k}\sum_{i=1}^{k} y_i \, , \ldots , \bar{y}_{t-k+1,t} = \frac{1}{k}\sum_{i=t-k+1}^{t} y_i \tag{4.10}$$

A série de médias móveis permite suavizar a série de dados original (retirar o ruído de flutuações mais fortes) formando um indicador de tendência.

Os valores de *Rolling std* descrevem a variabilidade dos dados em relação à *Rolling mean*, traduzindo a tendência da volatilidade da série temporal. Quanto maior for k, maior é o período da tendência analisada.

Outros testes de hipóteses

Em termos de análise inferencial, incluímos ainda no nosso estudo testes de hipóteses relacionados com o estudo da normalidade (Teste de *Jarque-Bera*, Teste de *Skewness* e Teste de *Kurtosis*), cuja implementação em *Python* é fornecida na biblioteca `scipy`, e da independência dos dados (Teste BDS), cuja implementação em *Python* é fornecida na biblioteca `statsmodels`.[101]

4.3.2. Modelos Autoagressivos de Médias Móveis

No que se refere ao processo de modelação e previsão, em termos de metodologias clássicas (lineares), deu-se particular atenção à família dos modelos autorregressivos (ARMA/ARIMA/SARIMA), cuja implementação computacional (*notebooks ARIMA.ipynb e SARIMA.ipynb*) é fornecida na biblioteca `statsmodels` por meio da classe `ARIMA`.

De modo sucinto, como numa série temporal os dados são registados sequencialmente ao longo do tempo, estes podem apresentar autocorrelação no tempo. Assim, em termos metodológicos foi seguida a metodologia de *Box&Jenkins* (a que já nos referimos anteriormente e descrita na Figura B.1), genericamente encontrada na literatura para os modelos ARIMA, visando captar a autocorrelação entre os valores observados da série temporal, para realizar previsões futuras.

Uma das etapas implícitas no processo de modelação e previsão em séries temporais respeita à seleção do modelo a usar. Alguns autores sugerem uma análise do correlograma da

[101] Para mais detalhes sobre estes testes, veja-se o Anexo B.3.

série, complementada com os valores obtidos dos critérios de informação, como são exemplo os critérios AIC, BIC e HQIC analisados na Sec. 2.5.

Complementando com as técnicas acima referidas, para que a escolha do modelo ARMA fosse consistente, pensou-se a construção de uma grelha de procura para identificar os 'melhores' modelos de entre um número considerável de possibilidades.

Assim, mediante a construção um algoritmo próprio (apresentado na Tabela 4.1) [102], foi desenhada uma função cíclica (ARIMA *Grid Search*) que, partindo de um conjunto de possíveis modelos, vai analisando modelo a modelo (em função dos hiperparâmetros p, d e q que o caracterizam) e estabelecendo uma hierarquia em função dos valores das estatísticas resultantes dos critérios AIC, BIC e HQIC para cada modelo.

TABELA 4.1 – Algoritmo relativo à função ARIMA

ALGORITMO 1: ARIMA *Grid Search*

```
Require: dataset to train ARIMA
Require: p_range,q_range,d_range to iterate ARIMA
Require: flag specify is dataset is log
Initialize variables
for p in p_range do
  for q in q_range do
    for d in d_range do
      Define model ARIMA(p,q,d)
      Fit model to dataset
      if model parameters converge then
        Extract AIC, BIC and HQIC
        if dataset is log then
          Get e^Prediction in-sample
          Get e^Forecast out-of-sample
        else
          Get Prediction in-sample
          Get Forecast out-of-sample
        end if
      end if
    end for
  end for
end for
```

[102] Ao longo deste trabalho, optou-se por apresentar todas as estruturas de algoritmos em Inglês, dada ser a língua usada na programação computacional.

A implementação computacional, em *Python*, do algoritmo apresentado pode ser consultada no Anexo D.2.

Assim, seguindo a metodologia de *Box&Jenkins*, os *notebooks ARIMA.ipynb* e *SARIMA.ipyn* foram estruturados em seis etapas: (i) importação dos dados; (ii) preparação dos dados; (iii) identificação dos modelos candidatos e seleção do modelo; (iv) estimação; (v) validação (diagnóstico) e (vi) previsão. De modo a apresentar os procedimentos seguidos em cada etapa, veja-se a Figura 4.1.

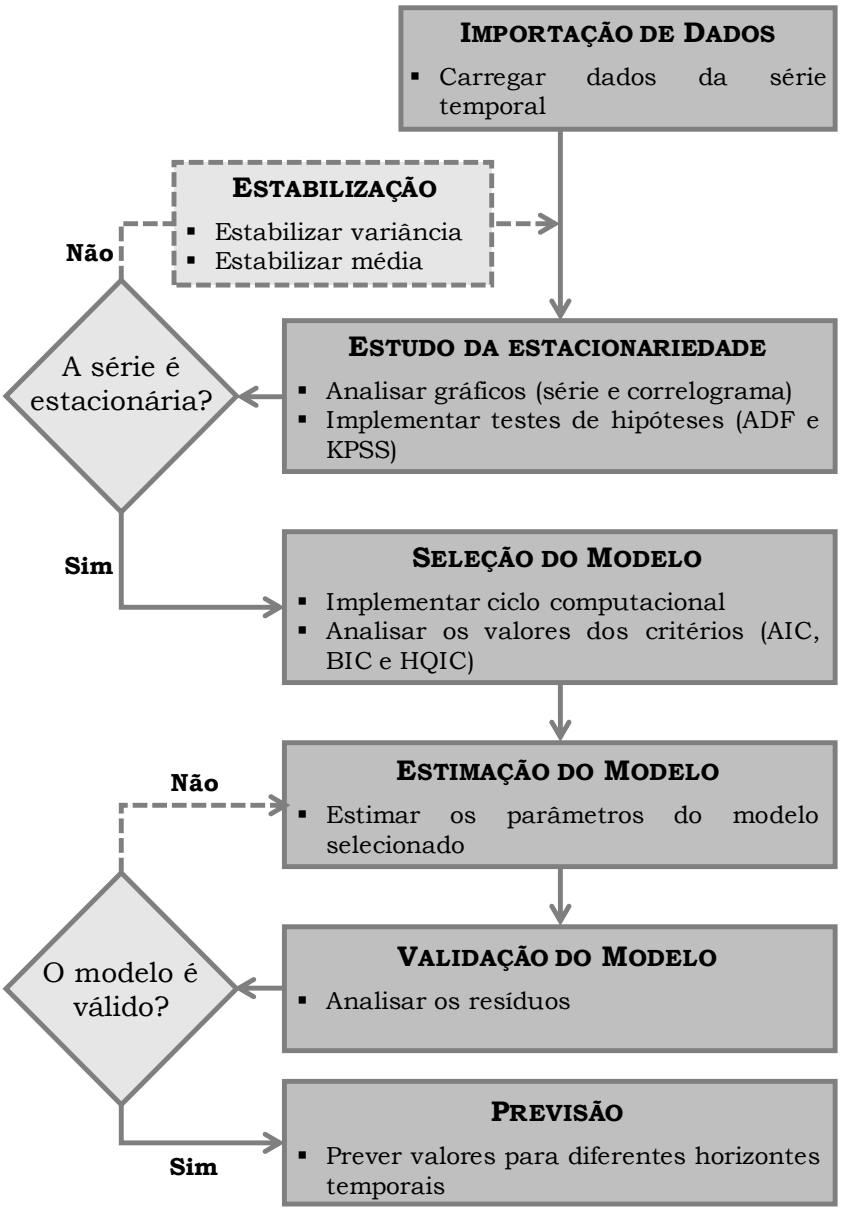


FIGURA 4.1 – Metodologia de implementação computacional dos modelos ARMA/ARIMA

4.3.3. MODELOS DE ALISAMENTO EXPONENCIAL

No que respeita à parte computacional em *Python*, a implementação dos modelos ETS (*notebook ExponenTialSmoothing.ipynb*) é fornecida na biblioteca `statsmodels` por meio das classes `SimpleExpSmoothing`, `Holt` e `ExponentialSmoothing`. Vejamos alguns detalhes e considerações sobre a implementação feita.[103]

Todos os parâmetros implícitos nos modelos ETS podem ser especificados explicitamente, pelo que o desafio está em conseguir defini-los de modo a obter boas previsões, face às várias combinações possíveis.

No seguimento do exposto na Sec. 2.4.2 e das considerações feitas, em termos de implementação computacional, consideraram-se neste trabalho 9 modelos ETS, como se descreve na Tabela 4.2, em que cada modelo é rotulado por um par de letras (T, S), tendo de acordo com as componentes tendência (T) e Sazonalidade (S).[104]

TABELA 4.2 – Classificação dos modelos ETS

	Componente sazonal (S)		
Componente de tendência (T)	Nenhuma (N)	Modelo Aditivo (A)	Modelo Multiplicativo (M)
Nenhuma (N)	(N,N)	(N,A)	(N,M)
Modelo Aditivo (A)	(A,N)	(A,A)	(A,M)
Modelo Aditivo amortecido (A_d)	(A_d,N)	(A_d,A)	(A_d,M)

Fonte: Adaptado de Hyndman & Athanasopoulos (2018, sec. 7.4)

De acordo com o exposto na Sec. 2.4.2, podemos identificar três grandes grupos de modelos, com:

- *Single Exponential Smoothing*: (N,N)
- *Double Exponential Smoothing*: (A,N) e (A_d,N)
- *Triple Exponential Smoothing*: (N,A), (A,A), (A_d,A), (N,M), (A,M) e (A_d,M)

[103] A implementação computacional é de autoria própria, seguindo algumas considerações apresentadas em Brownle (2018) e Hyndman & Athanasopoulos (2018).

[104] A classificação apresentada teve por base o descrito em Hyndman & Athanasopoulos (2018).

Para cada uma das séries, procurando evitar uma escolha *ad hoc* do melhor modelo para efetuar cenários de previsão, a qual poderia ser feita mediante análise gráfica das principais componentes da série (tendência e sazonalidade) e do modo como elas entram no método de suavização (por exemplo, de forma aditiva, amortecida ou multiplicativa), optou-se por considerar uma forma mais robusta analisando-se, para cada série, todos os modelos em paralelo com a informação obtida a partir dos critérios AIC e BIC.

Para os modelos em que existe convergência, tal como sugerido por Hyndman e Athanasopoulos (2018), uma forma robusta e objetiva de obter o valor dos hiperparâmetros numéricos será estimá-los a partir dos dados históricos, numa perspetiva de minimizar o erro.

Com efeito, para alcançar os objetivos traçados, para cada um dos nove modelos identificados na Tabela 4.2, foi desenhada uma grelha de procura cuja implementação computacional permita, simultaneamente, verificar a convergência do modelo e otimizar os hiperparâmetros do modelo.

Dada a implementação ser bastante idêntica (apenas mudam o número de hiperparâmetros a otimizar), a título de exemplo, apresenta-se na Tabela 4.3 a estrutura do algoritmo que explicita a forma como o ciclo foi desenhado para o modelo ETS(A_d, M), informação que é complementada no Anexo D.3, com a respetiva implementação em *Python*.

TABELA 4.3 – Algoritmo relativo ao modelo ETS(A_d, M)

ALGORITMO 2: Modelo ETS(A_d, M)

```
Require: Dataset to train ETS
Define model ETS(Ad, M)
Fit model to dataset
if Model parameters converge then
  Get Forecast out-of-sample
  if Forecast data is a number then
    Plot ETS forecast and dataset
  end if
end if
```

Para o caso dos modelos ETS, embora não exista uma metodologia 'teórica', foi desenhada uma sequência de procedimentos estruturados em seis etapas: (i) importação dos dados; (ii) análise da série temporal; (iii) identificação dos modelos candidatos e seleção do modelo; (iv) estimação do modelo; (v) avaliação (qualitativa e quantitativa) do modelo e (vi) previsão.

Metodologicamente, na Figura 4.2 estão apresentadas as etapas relativas à implementação computacional dos modelos ETS.

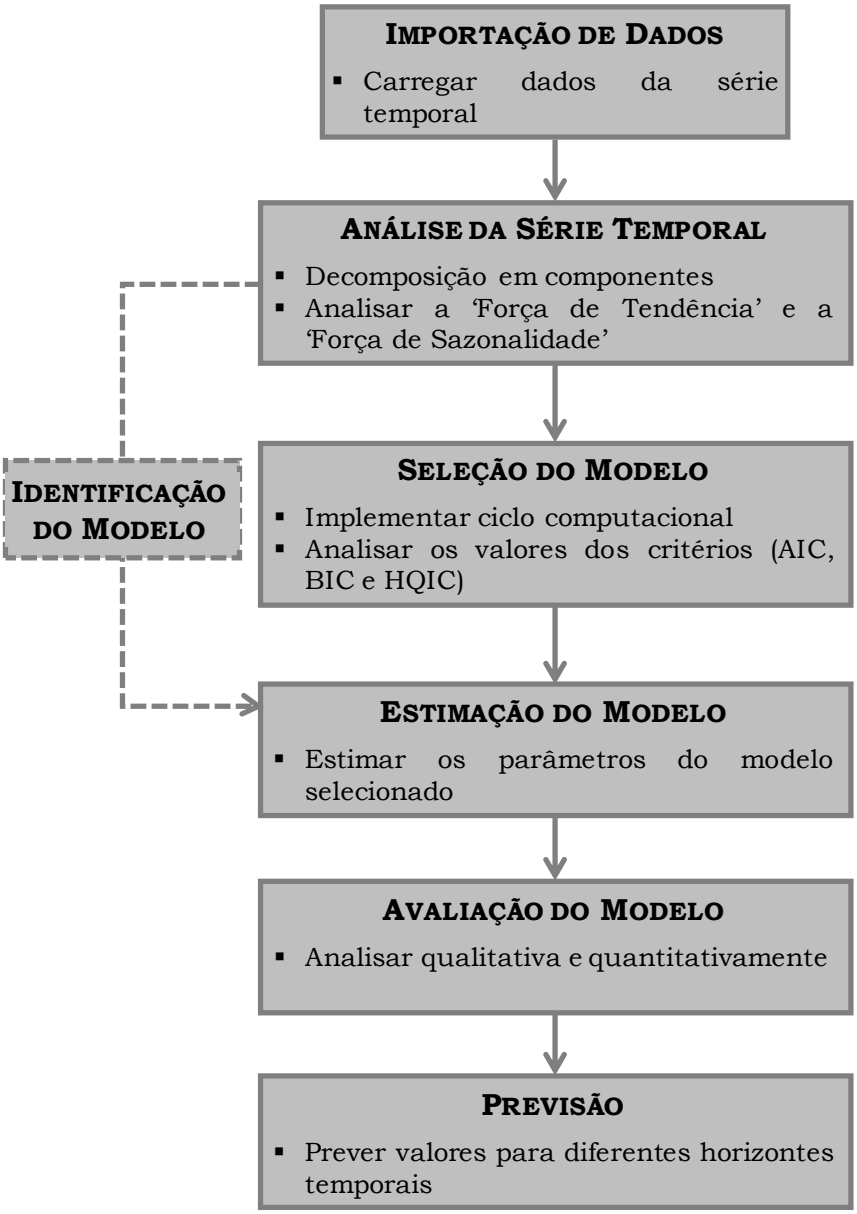


FIGURA 4.2 – Metodologia de implementação computacional dos modelos ETS

4.3.4. MODELOS DE REDES NEURONAIS

A parte respeitante à implementação computacional das redes neuronais foi um dos principais desafios não só na escrita das rotinas, como (e sobretudo) no desenho de um *script* claro para o utilizador e, ao mesmo tempo, robusto computacionalmente.

Em *Python*, a implementação destes modelos (*notebooks DeepNeuralNetwork.ipynb* e *DNN_OurApproach.ipynb*) é fornecida na biblioteca `tensorflow` por meio das classes `Dense`, `SimpleRNN` e `LSTM`.

Em termos de arquiteturas de redes, foram implementadas três arquiteturas (MLP, RNN e LSTM) que, em termos de *modus operandi*, embora existam algumas diferenças, a base arquitetural é comum. Por esse motivo, escusamo-nos a uma referência separada a cada uma das arquiteturas, centrando particular atenção na clarificação de aspetos metodológicos, nomeadamente quanto às opções que podem ser feitas ao longo do processo de modelação.

Para os modelos em causa, o código foi construído tendo por base alguns aspetos fundamentais que passam, não só, pela parte referente ao *input* a dar à rede (incluindo a abordagem às diferentes possibilidades em termos de pré-processamento dos dados, como a suavização, a normalização ou a estandardização); como também pela definição da metodologia de *cross-validation* para validação do modelo e respetiva definição dos vários hiperparâmetros da rede neuronal (número de camadas da rede, número de neurónios por camada, número de épocas de treino, funções de ativação, algoritmos de otimização, entre outros).

Face à multiplicidade de combinações possíveis, não existindo uma metodologia prévia de estudo (como a metodologia de *Box&Jenkins* para os modelos ARIMA, por exemplo), nem sendo viável uma apresentação sistemática semelhante à feito nos modelos ETS (com a identificação dos modelos possíveis), exigiu-se a definição de uma metodologia própria (assente numa planificação e estruturação prévia), seguida de um trabalho metódico em termos de estudo computacional, desenvolvido no sentido de encontrar combinações conducentes a modelos de redes neuronais com boa *performance* preditiva.

Num primeiro momento do nosso estudo, partido de uma *baseline* resultante da revisão da literatura[105] e das sugestões aí encontradas (nomeadamente sobre que transformações iniciais de dados poderão ajudar no processo de modelação, ou que funções de ativação e que algoritmos de otimização usar), apresenta-se na figura seguinte (Figura 4.3) um esquema das etapas contempladas no *notebook* computacional *DeepNeuralNetwork.ipynb*.

Por forma a dar conta de algumas das opções e/ou hiperparâmetros envolvidos, evitando uma apresentação pesada, optámos por ilustrar algumas etapas com exemplos de escolhas envolvidas.

[105] Nomeadamente em Brownlee (2017).

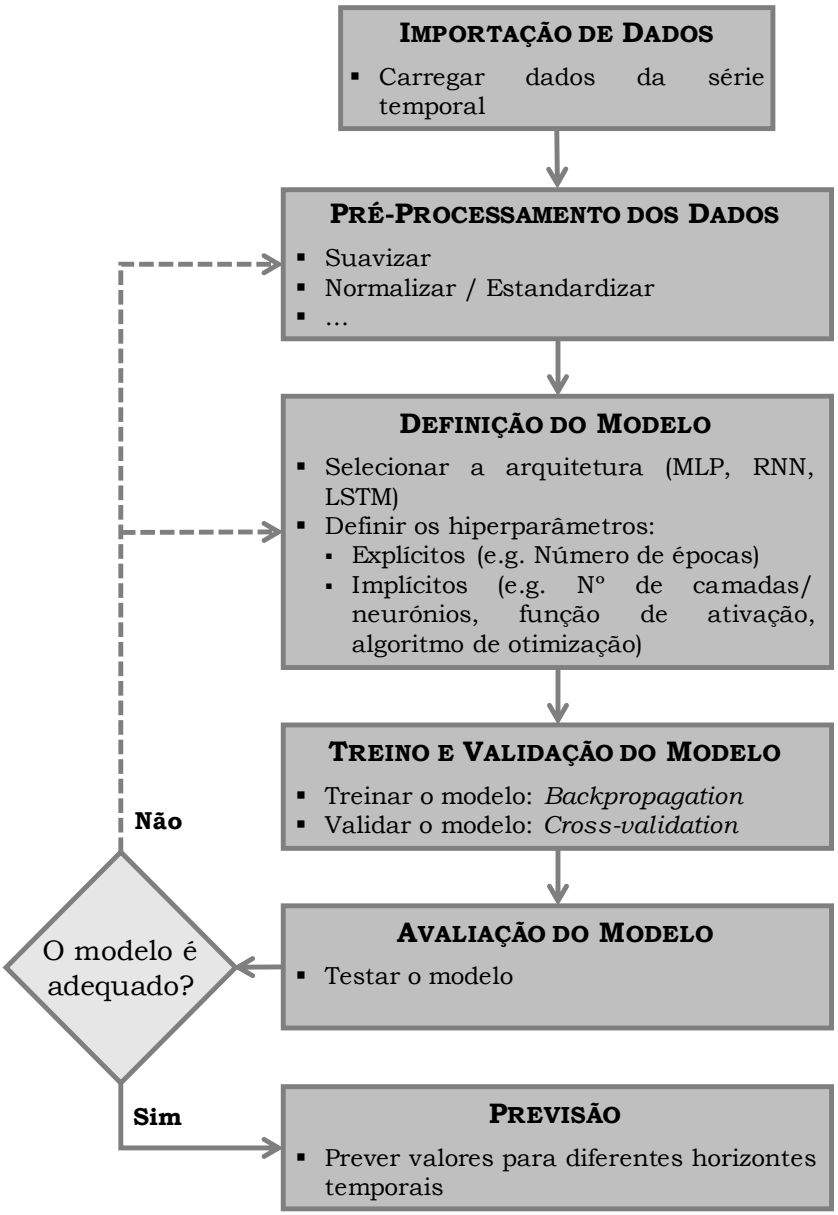


FIGURA 4.3 – Metodologia de implementação computacional dos modelos de DNN

Sobre os procedimentos metodológicos adotados nos modelos DNN, será importante referir ainda que, na etapa referente à previsão, para fazer face a uma aleatoriedade subjacente à existência da *black-box* (a que nos referimos na Sec. 3.5), procurámos dar alguma consistência aos nossos resultados.

Deste modo, dado que de cada *run* do código os valores obtidos não são exatamente os mesmos (contrariamente à exatidão subjacente aos modelos ARIMA ou ETS), considerou-se adequado pensar uma alternativa que evitasse a apresentação de conclusões falaciosas resultantes de um *run* aleatório com erros de previsão muito baixos/altos.

Para o efeito, definido o modelo de rede, foram feitas (em ciclo) várias simulações de previsão, e considerado um intervalo aparado dos valores preditos. Isto é, do total dos cenários de previsão obtidos, no geral, foram excluídas as 5% previsões que apresentavam melhores e piores resultados, apresentando-se os resultados de previsão (Sec. 5.2.3) identificando o intervalo de variação (valor mínimo e máximo aparado). Com isto procurámos dar uma maior fiabilidade às previsões obtidas.

Ainda, dada a importância da etapa referente à *cross-validation*, no contexto deste trabalho foi implementada de raiz uma biblioteca em *Python* que permitisse:

1. dividir o conjunto de dados de três formas possíveis, considerando: (i) apenas dados de treino; (ii) dados de treino e dados de validação; ou (iii) dados de treino, dados de validação e dados de teste;
2. dividir automaticamente os dados por listas de *inputs/outputs* a serem utilizadas pela rede neuronal, depois de fixado o número de *inputs* a fornecer (camada de entrada) e o número de *outputs* a obter (camada de saída);
3. implementar metodologias de *cross-validation* para series temporais: *Forward Chaining*; *k-Fold*; *Group k-Fold.*

A biblioteca em causa está em *open-source* e disponível a toda a comunidade científica em Lopes e Ramos (2019), para que possam utilizá-la e adaptá-la de acordo com os interesses particulares. A título de exemplo, apresenta-se na Tabela 4.4 a estrutura do algoritmo construído para a metodologia de *cross-validation* habitualmente usada em series temporais, *Forward Chaining*, considerando o conjunto de dados divididos em dados de treino, dados de validação e dados de teste. [106]

Não só pela inovação teórica, a biblioteca implementada constitui, sem dúvida, um dos pontos fortes do trabalho desenvolvido pela importância e utilidade, em termos práticos, que acabaria por vir a ter neste trabalho. Por um lado, a sua construção permitiu-nos perceber a dinâmica de *cross-validation* implícita no caso de séries temporais e como é que os dados são usados para efeitos de treino da rede neuronal e validação ou teste do modelo resultante. Por outro lado, a partir de um esquema de funcionalidade desenhado por nós, foi possível explorar vários cenários (numa perspetiva de *Data Science*) de acordo com os nossos interesses.

[106] Os detalhes em termos de implementação em *Python* são apresentados no Anexo D.4.

TABELA 4.4 – Algoritmo relativo à metodologia de *cross-validation Forward Chaining* (com dados de treino, dados de validação e dados de teste)

ALGORITMO 3: *Forward Chaining* (com dados de treino, dados de validação e dados de teste)

Require: `Dataset to train`
Require: `numInputs for Neural Network`
Require: `numOutputs for Neural Network`
Require: `numJumps in dataset for Neural Network`

$j = 2$
// Iterate through all train/val/test splits
while `Forever` **do**
  $start\_ix$, end_ix, $startCv\_ix$, $endCv\_ix$, $startTest\_ix$, $endTest\_ix = 0$
  $i = 0$
  // Iterate until ix of individual training set is smaller than ix of cv set
  while $i < j$ **do**
    // Training Data
    $start\_ix = i \times numJumps$
    $end\_ix = start\_ix + numInputs$
    X = dataset from $start\_ix$ to end_ix
    Y = dataset from end_ix to $end\_ix + numOutputs$
    Increment i
    // Once test data crosses time series length return
    if $(end\_ix + numInputs + numInputs + numOutputs) > len(dataset)$ **then**
      break
    end if
    // Cross Validation Data
    $startCv\_ix = end\_ix$
    $endCv\_ix = startCv\_ix + numInputs$
    Xcv = dataset from $startCv\_ix$ to $endCv\_ix$
    Ycv = dataset from $endCv\_ix$ to $endCv\_ix + numOutputs$
    // Test Data
    $startTest\_ix = endCv\_ix$
    $endTest\_ix = startTest\_ix + numInputs$
    Xtest = dataset from $startTest\_ix$ to $endTest\_ix$
    Ytest = dataset from $endTest\_ix$ to $endCv\_ix + numOutputs$
    // Add Train/Val/Test sets
    Append pair (X, Y) to train set
    Append pair (Xcv, Ycv) to val set
    Append pair $(Xtest, Ytest)$ to test set
    Increment j
    // dataset provided does not has size enough to split into sets
    if $len(X) == 0$ **or** $len(Xcv) == 0$ **or** $len(Xtest) == 0$ **then**
      break
    end if
  end while
end while

A título ilustrativo do exposto, podemos referir dois exemplos.

Um primeiro exemplo prende-se com testes feitos a alguns hiperparâmetros. Em particular, para o hiperparâmetro *numJumps* (número de saltos nos dados) foram testadas várias possibilidades no sentido de diminuir o tempo implícito no treino da rede, salvaguardando a retenção de informação relevante. Após vários testes, conclui-se que se obtinham melhores resultados quando se considerava nulo este parâmetros, ou seja, se utilizavam sequencialmente todos os dados, sem saltos (sobretudo quando usada a metodologia *Forward Chaining*).

Um segundo exemplo respeita à opção referida em **1.**, a qual abriu perspetivas para explorar outras possibilidades na divisão dos dados. Esta opção, combinada com a utilização da metodologia de *cross-validation Group k-Fold*, acaba por ser uma das particularidades do "Modelo de Redes Neuronais Modificado" (descrito na Sec. 5.2.4). Em linhas gerais, este modelo distingue-se por uma abordagem pessoal à implementação da metodologia de *cross-validation Group k-Fold* a uma série estacionária cujos dados da série foram divididos apenas em dados de treino e de validação (mais detalhes serão descritos na Sec. 5.3.4.). A estrutura do algoritmo é apresentada na Tabela 4.5 e os detalhes da respetiva implementação em *Python* podem ser consultados no Anexo D.5.

4.4. *FORECASTING* E METODOLOGIA DE AVALIAÇÃO: MÉTRICAS (ERRO)

O propósito (principal) do ajustamento da série por modelos econométricos é projetá-la para além do período da amostra. Associado a este objetivo surge o conceito de 'erro de previsão'. Pretendemos selecionar e desenvolver modelos em que estes erros sejam os menores possíveis, daí que este conceito assuma um papel preponderante no desenvolver da investigação e na análise dos valores preditos.

Nas projeções ou previsões há dois tipos de erros: (i) aleatórios, devido ao desconhecimento de variações futuras cujos fatores não são contemplados no modelo; e (ii) sistemáticos, cometidos consistentemente devido, por exemplo, à seleção de relações matemáticas incorretas entre as variáveis, ou a diferenças entre os parâmetros verdadeiros e as suas estimativas. Ambos contribuem para o erro de previsão e o melhor modelo será aquele que permite minimizar as medidas do erro sistemático (Hamilton, 1994).

No que se segue, considerando a série $\{y_t\}_{t\in T}$, vamos assumir que dispomos das observações passadas para os períodos $1, \dots, t$ e que todas as previsões feitas estão condicionadas à informação disponível em t.

TABELA 4.5 – Algoritmo relativo à metodologia de *cross-validation Group k-Fold* (com dados de treino e dados de validação)

ALGORITMO 4: *Group k-Fold* (com dados de treino e dados de validação)

Require: `Dataset to train`
Require: `numInputs for Neural Network`
Require: `numOutputs for Neural Network`
Require: `numJumps in dataset for Neural Network`

```
Set j = 0
// Iterate through 5 train/val splits
while j < 5 do
    start_ix, end_ix, startCv_ix, endCv_ix, startTest_ix, endTest_ix = 0
    Set i = 0
    Set n = 0 // Number of numJumps
    while Forever do
        if (i + 1 + j)%(5)! = 0 then
            // Training Data
            start_ix = endCv_ix + n × numJumps
            end_ix = start_ix + numInputs
            Increment n
            // Leave train/val split loop once train data crosses dataset length
            if end_ix + numOutputs > len(dataset) − 1 then
                break
            end if
            X = dataset from start_ix to end_ix
            Y = dataset from end_ix to end_ix + numOutputs
        else
            // Cross-Validation Data
            startCv_ix = end_ix
            endCv_ix = startCv_ix + numInputs
            Set n = 0
            // Leave train/val split loop once train data crosses dataset length
            if endCv_ix + numOutputs > len(dataset) then
                break
            end if
            Xcv = dataset from startCv_ix to endCv_ix
            Ycv = dataset from endCv_ix to endCv_ix + numOutputs
        end if
        Increment i
    end while
    // Add Train/Val sets
    Append pair (X, Y) to train set
    Append pair (Xcv, Ycv) to val set
    Increment j
end while
// dataset provided does not has size enough to split into sets
if len(X) == 0 or len(Xcv) == 0 or len(Xtest) == 0 then
    return
end if
```

Assim, sendo y_{t+h} o valor desconhecido no período futuro $t+h$ e $\hat{y}_{t+h}$ a sua previsão obtida com base na informação disponível até ao momento t, o erro de previsão corresponde à diferença entre esses dois valores

$$e_{t+h} = y_{t+h} - \hat{y}_{t+h} \tag{4.11}$$

Sendo o propósito avaliar o desempenho global de um determinado modelo de previsão, torna-se essencial analisar os valores de indicadores capazes de traduzir, em termos numéricos, a medida de erro. De entre as métricas de erro comummente referidas na bibliografia, destacamos as seguintes, onde s denota o número de previsões a efetuar (janela de previsão):

Mean Error (ME)

$$ME = \frac{\sum_{i=1}^{s} e_i}{s} \tag{4.12}$$

Mean Absolute Error (MAE)

$$MAE = \frac{\sum_{i=1}^{s} |e_i|}{s} \tag{4.13}$$

Mean Absolute Percentage Error (MAPE)

$$MAPE = \frac{\sum_{i=1}^{s} \left| \frac{y_{t+i} - \hat{y}_{t+i}}{y_{t+i}} \right|}{s} \times 100 \tag{4.14}$$

Mean Squared Error (MSE)

$$MSE = \frac{\sum_{i=1}^{s} e_i^2}{s} \tag{4.15}$$

Root mean squared error (RMSE)

$$RMSE = \sqrt{MSE} \tag{4.16}$$

Conseguimos identificar trabalhos científicos na literatura que recorrem a todas estas medidas, pelo que qualquer uma poderia ser opção. Contudo, a utilização do MAE e do MAPE parece ser a mais frequente, possivelmente pelo MSE e o RMSE serem obtidos a partir de uma escala diferente da original (quadrado), sendo mais sensíveis a erros elevados (Willmott & Matsuura, 2005).

Quando comparamos as duas métricas, MAE e MAPE, verificamos que o MAE é uma 'grandeza dimensional' por ser expresso na unidade de medida dos dados, enquanto o MAPE é uma 'grandeza adimensional' por avaliar a dimensão do erro em termos percentuais. Esta característica apresenta vantagens, não só pela simplificação da interpretação, mas também por permitir comparar os erros de séries de dados com unidades de medida distintas. Importa, contudo, salvaguardar que o MAPE não pode ser usado quando existem valores nulos para y_{t+i} (deve ter-se em conta a natureza dos dados), além de ser uma medida sensível, que não deve ser utilizada quando se trabalha com poucos dados históricos.[107]

[107] Note-se que, por o valor real, y_{t+i}, estar no denominador da equação, o MAPE irá assumir valores extremos no caso deste apresentar um valor muito baixo ou muito alto. Esta sensibilidade de escala torna o MAPE ineficiente como uma medida de erro para um número reduzido de dados, sendo o MAE mais vantajoso nesses casos.

5. ESTUDO EMPÍRICO

Traçada uma perspetiva teórica, bem como os procedimentos metodológicos seguidos em termos computacionais, e feita uma apresentação sucinta das séries temporais em estudo, no presente capítulo, após a sua análise exploratória (Sec. 5.1), passar-se-á ao processo afeto à modelação e previsão das series temporais apresentadas na Sec. 4.1, envolvendo a apresentação dos modelos selecionados e respetivas previsões (Sec. 5.2) e a posterior comparação e discussão dos resultados (Sec. 5.3).

Refira-se ainda que todos os *outputs* apresentados foram obtidos da implementação computacional feita em *Python*, embora tenham sido sujeitos a um tratamento gráfico.[108]

5.1. ANÁLISE EXPLORATÓRIA DAS SÉRIES TEMPORAIS

Em termos de análise exploratória de dados, destacamos: (i) a análise gráfica de cada uma das séries, visando compreender as suas características; (ii) a determinação de algumas medidas descritivas (de tendência central, não central e de variabilidade); (iii) a implementação de testes de hipóteses para validar parte das conjeturas estabelecidas.

5.1.1. SÉRIE CPIAUCSL

A primeira série temporal, CPIAUCSL[109], respeita a dados mensais do índice de preços ao consumidor norte-americano relativos ao período entre 01-01-1947 e 01-09-2019, num total de 873 observações, cuja representação gráfica se encontra na Figura 5.1. Complementando, vejam-se em anexo as representações gráficas relativas à decomposição da série em componentes aditiva e multiplicativa (Figura E.1) e à aplicação do algoritmo CUSUM para deteção de alterações bruscas (Figura E.2). Das representações gráficas da série, depreende-se

[108] Refira-se que, por uma questão de utilização posterior das imagens apresentadas neste trabalho em publicação científica, opta-se por apresentar a legendagem em Inglês.

[109] Dados importados diretamente de https://fred.stlouisfed.org/ - código 'CPIAUCSL'.

uma tendência, sem presença de comportamento sazonal e a inexistência de quebras de estrutura.

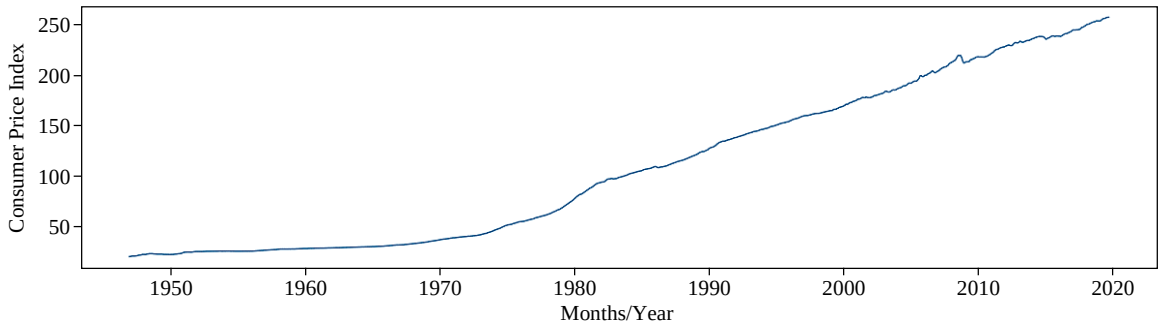


FIGURA 5.1 – Série CPIAUCSL: Representação gráfica

Sujeita a validação, por aplicação de testes de raiz unitária/estacionariedade, da representação gráfica depreende-se também a não estacionariedade da série, facto que é reforçado pela representação gráfica do *Rolling mean* e *Rolling std* (Figura 5.2), onde é visível a variação considerável do desvio-padrão e o aumento da média com o tempo (características de uma série não estacionária).

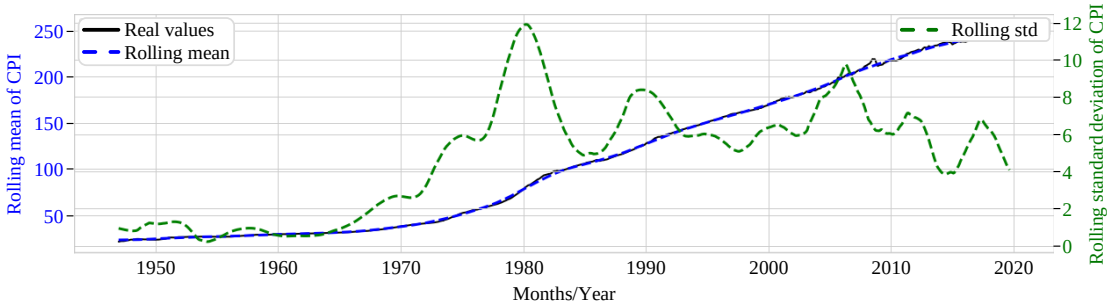


FIGURA 5.2 – Série CPIAUCSL: *Rolling mean* e *Rolling std*

As principais estatísticas descritivas, apresentadas na Tabela 5.1, são complementadas com as representações gráficas, apresentadas em anexo, da função de distribuição (Figura E.3) e da função cumulativa de distribuição (Figura E.4).

TABELA 5.1 – Série CPIAUCSL: Estatísticas descritivas

Count	*Mean*	*Std*	*Min*	*Q1*	*Q2*	*Q3*	*Max*	*Kurtosis*	*Skewness*
873	109.60	77.37	21.48	31.31	99.20	177.40	256.36	-1.308	0.397

Igualmente sujeita a validação (por aplicação de testes de hipóteses adequados), depreende-se dos valores dos coeficientes de curtose (*Kurtosis*) e de assimetria (*Skewness*) a não normalidade da série (série platicúrtica e assimétrica).

As conjeturas feitas podem ser validadas (com as respetivas probabilidades de erro) com base nos testes de hipóteses respetivos (ver Tabela 5.2).

TABELA 5.2 – Série CPIAUCSL: Testes de normalidade, estacionariedade e independência

	Testes de Normalidade			**Testes de Raiz Unitária/ Estacionariedade**		**Teste de Independência**
	Kurtosis	*Skewness*	*Jarque-Bera*	ADF	KPSS	BDS (Dim. 2 – Dim. 6)
statistic	-44.8588	4.6632	85.1703	1.8468	3.9743	155.0103 – 275.3729
p-value	0.0000*	0.0000*	0.0000*	0.9984	----------	0.0000*

*Rejeita-se H_0 para os níveis de significância de 1%, 5% e 10%

Tendo em conta o apresentado na Tabela 5.2, verificamos que, para qualquer nível de significância, é rejeitada a hipótese nula de normalidade, pelo que existem evidências estatisticamente significavas para admitir a não normalidade da distribuição de valores da série CPIAUCSL. Verifica-se ainda a não estacionariedade da série, uma vez que não há evidência, para qualquer nível de significância usual, para rejeitar a hipótese nula referente ao teste ADF (existência de raiz unitária) e a estatística de teste referente ao teste KPSS apresenta um valor superior aos valores críticos. Quanto ao teste de independência (teste BDS), para qualquer nível de significância, é rejeitada a hipótese nula (os dados são iid), pelo que se poderá deduzir a existência de algum tipo de dependência nos dados (neste caso é visível uma dependência linear).

5.1.2. SÉRIE VMT

Os dados relativos à evolução mensal do volume de tráfego nos Estados Unidos, expresso em milhões de milhas (sem ajuste sazonal), série VMT[110], obtidos mensalmente no período de 01-01-1970 a 01-09-2019, num total de 597 observações, encontram-se representados na Figura 5.3, sendo apresentadas em anexo a decomposição aditiva e multiplicativa (Figura E.7) e a

[110] Dados diretamente importados de https://fred.stlouisfed.org/ - código 'TRFVOLUSM227NFWA'

análise de quebras de estrutura (Figura E.8). Destas representações gráficas depreende-se a existência de uma tendência linear crescente, evidências claras de comportamento sazonal (ao longo do ano) e um aumento da variabilidade ao longo dos anos sem quebras de estrutura evidentes.

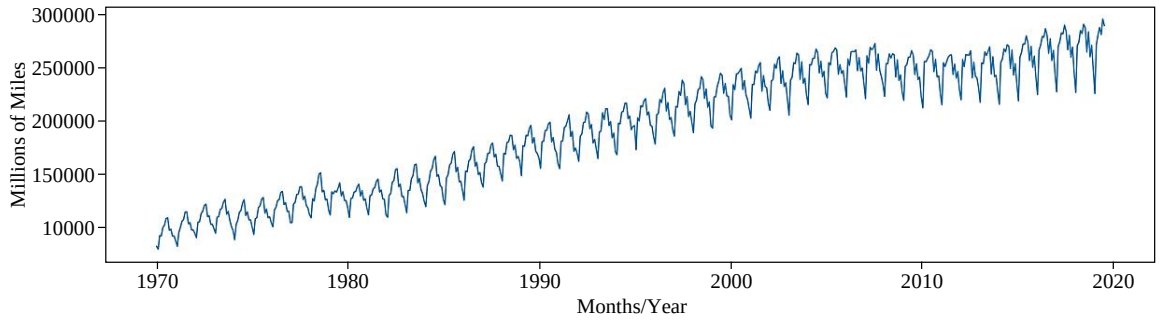


FIGURA 5.3 – Série VMT: Representação gráfica

A não estacionariedade da série (a validar pelo teste de raiz unitária) pode igualmente ser observada pelo aumento no tempo do valor médio – *Rolling mean* –e pela variação considerável do desvio-padrão – *Rolling std* – evidenciados na Figura E.9, em anexo.

O comportamento sazonal pode ser igualmente depreendido da representação gráfica dos *box-plots* mensais agrupados, apresentados na Figura 5.4, ilustrando a distribuição dos valores analisados de milhões de milhas da série para cada mês do ano. A avaliar não só pelos valores mínimos e máximos observados, como também pelos valores dos quartis, verifica-se que o volume de tráfego é alvo de flutuações ao longo dos meses do ano, cujos valores nos meses de julho e agosto são claramente superiores quando comparados com os valores observados nos meses de janeiro e fevereiro.

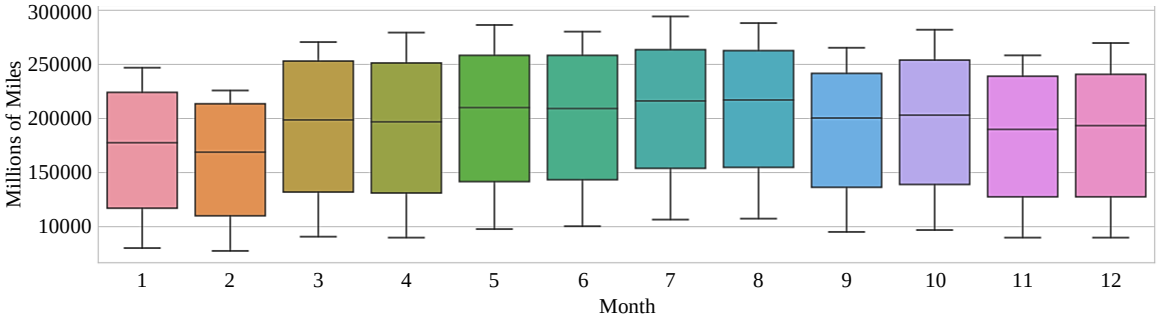


FIGURA 5.4 – Série VMT: Representação gráfica dos *box-plots* mensais

Finalmente, tendo em conta a distribuição dos dados, apresentada na Figura 5.5, facilmente se infere a não normalidade da distribuição, já que o histograma da distribuição evidencia uma série bimodal e platicúrtica.

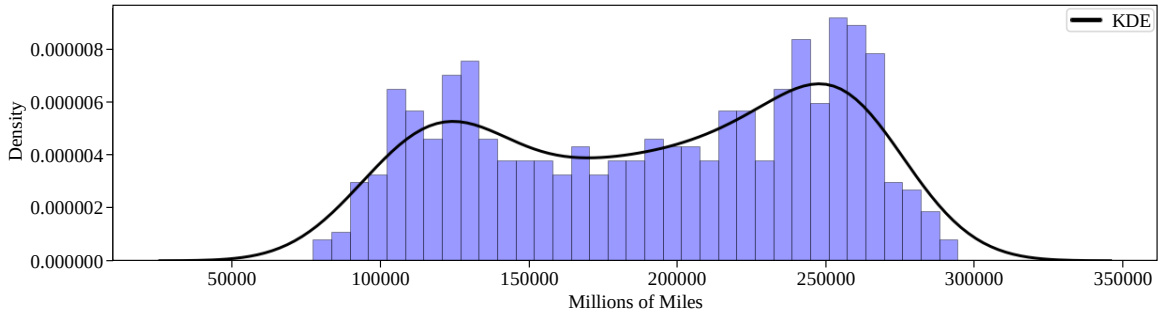


FIGURA 5.5 – Série VMT: Histograma (com curva de densidade)

A reforçar as considerações anteriores, veja-se o valor da curtose apresentado na Tabela 5.3, em conjunto com outras das estatísticas descritivas relativas à série VMT, onde a grande diferença entre os valores mínimo (77442 milhões) e máximo (294336 milhões) retrata a evolução (aumento) do volume de milhas viajadas nos Estados Unidos entre os anos de 1970 (ano onde foi verificado o valor mínimo) e 2019 (ano de registo do valor máximo).

TABELA 5.3 – Série VMT: Estatísticas descritivas

Count	*Mean*	*Std*	*Min*	*Q1*	*Q2*	*Q3*	*Max*	*Kurtosis*	*Skewness*
596	191500.82	58210.74	77442	134397	198003	244714	294336	-1.332	-0.171

A proximidade dos valores da média e da mediana, bem como a localização dos quartis (veja-se a representação da função cumulativa da distribuição, na Figura E.10 em anexo), são concordantes com o valor do coeficiente de assimetria, evidenciando alguma simetria na distribuição dos dados.

Os resultados dos testes de hipóteses sobre a normalidade e a estacionariedade dos dados da série VMT são apresentados na Tabela 5.4.

No que respeita à normalidade, apesar do *p-value* referente ao teste *Skewness* conduzir à não rejeição da hipótese nula (distribuição simétrica) para os níveis de significância de 1% e 5%, o teste de *Jarque-Bera* indica a existência de evidências estatisticamente significativas para considerar que os dados da série VMT não seguem uma distribuição aproximadamente normal.

TABELA 5.4 – Série VMT: Testes de normalidade, estacionariedade e independência

	Testes de Normalidade			Testes de Raiz Unitária/ Estacionariedade		Teste de Independência
	Kurtosis	*Skewness*	*Jarque-Bera*	ADF	KPSS	BDS (Dim. 2 – Dim. 6)
statistic	-40.3412	-1.7099	46.9367	-1.2577	3.0569	122.8501 – 210.5026
p-value	0.0000*	0.0873**	0.0000*	0.6483	----------	0.0000*

*Rejeita-se H_0 para os níveis de significância de 1%, 5% e 10%
**Não se Rejeita H_0 para os níveis de significância de 1% e 5%

Relativamente à estacionariedade, conclui-se a não estacionariedade da série, uma vez que, para qualquer nível de significância usual, não há evidência para rejeitar a existência de uma raiz unitária (teste ADF), além de o valor da estatística referente ao teste KPSS a ser superior aos valores críticos. Quanto à independência, para qualquer nível de significância, é rejeitada a hipótese nula do teste BDS, pelo que, à semelhança da série anterior, se poderá deduzir a existência algum tipo de dependência nos dados.

5.1.3. SÉRIE PSI 20

No contexto de séries financeiras, a primeira série temporal em análise diz respeito ao principal índice de referência do mercado de capitais português (PSI 20)[111]. Embora o histórico de valores esteja disponível desde 31-12-1992, ao atualizar os dados relativamente aos apresentados em Ramos *et al.* (2018), optámos por considerar só os valores da série após a entrada em vigor do euro, abrangendo assim o período de 02-01-2002 a 27-09-2019.

Os dados em estudo referem-se aos valores diários (*business days*) de fecho, num total de 4534 observações, cuja representação gráfica se apresenta na Figura 5.6.

FIGURA 5.6 – Série PSI 20: Representação gráfica

[111] Dados obtidos de https://stooq.pl/.

Complementando com a decomposição da série, apresentada na Figura E.13 em anexo, observa-se um padrão não linear com evidências claras de regimes distintos marcados, essencialmente, pela forte quebra registada em 2008, seguindo-se um período com alguma instabilidade, nomeadamente decorrente dos fenómenos que remontam a 2011 e 2014.

Especificamente, decorrente da instabilidade nos mercados (desde a crise de 2008) e do consequente contágio à dívida soberana e à redução das classificações de *rating*, Portugal é encaminhado para um resgate financeiro pedido em abril de 2011, ao qual se seguiu um ano de profunda recessão dos mercados. Em meados de 2012 seguiu-se um período de valorização do mercado acionista, ainda que com volatilidades. Em 2014, em virtude de acontecimentos externos e, sobretudo, internos, como a fragilidade de parte das instituições bancarias (por exemplo, já depois da exclusão dos títulos do ESFG do PSI 20, em agosto de 2014 as ações do BES foram excluídas a valer zero euros), o índice bolsista português é arrastado para quedas graduais e acentuadas.

Por aplicação do algoritmo CUSUM, identificamos na Figura 5.7 vários pontos de quebra de estrutura ao longo de 2008 e ocorrências pontuais em 2011 e 2014 (destacados a vermelho na parte superior), correspondendo a momentos em que as somas cumulativas das variações, negativas ou positivas, alcançam um valor limite (‘linha de alarme’ traçada a vermelho na parte inferior).

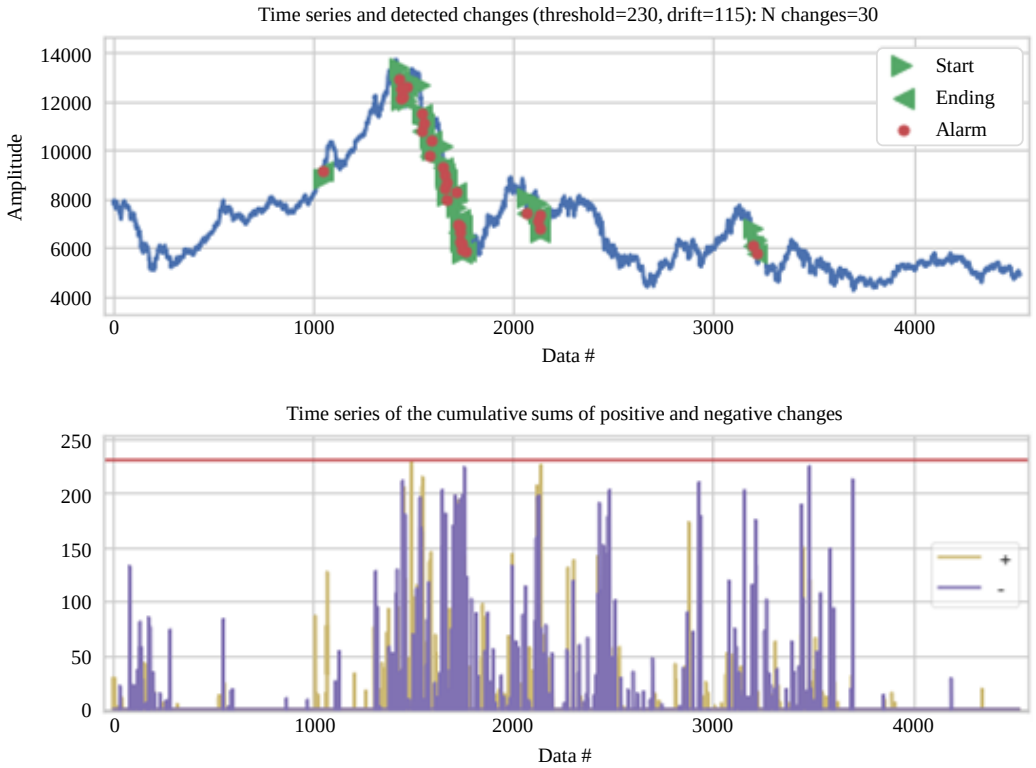


FIGURA 5.7 – Série PSI 20: Mudanças/quebras estruturais

A elevada instabilidade observada pode também ser analisada, em termos anuais, por análise da representação gráfica dos *box-plots* anuais (Figura 5.8), onde se observa uma amplitude amostral considerável no ano de 2008 (correspondendo a uma variação de mais de 7000 unidades) e uma amplitude interquartil de destaque (cerca de 2000 unidades) nos anos de 2009, 2011 e 2014, contrapondo com os restantes anos em análise. Tal facto reforça a existência de variabilidade considerável nos dados (quer entre os valores máximos e mínimos observados, quer em termos de concentração de observações), evidenciando a existência de alguma oscilação.

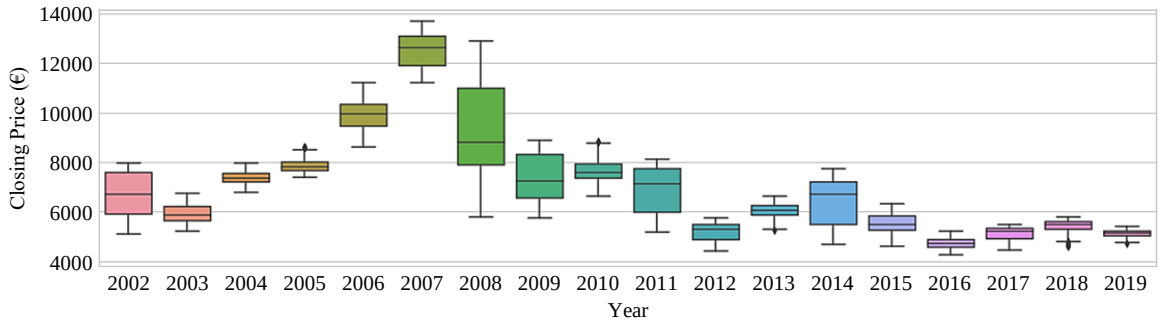


FIGURA 5.8 – Série PSI 20: Representação gráfica dos *box-plots* anuais

Estas grandes flutuações estão obviamente associadas a uma ausência de estacionariedade dos dados, como se pode também observar na Figura E.14, em anexo, pelas grandes variações do *Rolling mean* e do *Rolling std*. A não normalidade dos dados da série é facilmente depreendida do histograma com a distribuição dos dados (Figura E.15 em anexo), bimodal com assimetria positiva, bem como pelos valores da curtose e simetria apresentados na Tabela 5.5, conjuntamente com outras medidas descritivas.

Destaque ainda para o valor histórico máximo, verificado antes da crise de 2008, a destacar-se dos valores das restantes medidas estatísticas (media e quartis), valores que nunca mais o índice bolsista português conseguiu alguma aproximação.

TABELA 5.5 – Série PSI 20: Estatísticas descritivas

Count	*Mean*	*Std*	*Min*	*Q1*	*Q2*	*Q3*	*Max*	*Kurtosis*	*Skewness*
4534	6946.40	2063.53	4260.13	5372.01	6284.98	7771.66	13702.03	1.217	1.293

Na Tabela 5.6 são apresentados os resultados dos testes de hipóteses referentes às caraterísticas observadas, bem como à independência dos dados.

TABELA 5.6 – Série PSI 20: Testes de normalidade, estacionariedade e independência

	Testes de Normalidade			**Testes de Raiz Unitária/ Estacionariedade**		**Teste de Independência**
	Kurtosis	*Skewness*	*Jarque-Bera*	ADF	KPSS	BDS (Dim. 2 – Dim. 6)
statistic	3.2793	17.8311	495.5003	-2.3407	2.7392	11.0889 – 20.6448
p-value	0.0010*	0.0000*	0.0000*	0.1592	----------	0.0000*

*Rejeita-se H_0 para os níveis de significância de 1%, 5% e 10%

Como espectável, rejeita-se a normalidade para qualquer nível de significância por qualquer dos testes (*Kurtosis*, *Skewness* e *Jarque-Bera*). Dos valores apresentados na tabela, existem ainda evidências estatisticamente significativas para admitir a não estacionariedade da série (não sendo rejeitada a hipótese nula referente ao teste ADF e apresentando a estatística de teste referente ao teste KPSS um valor superior aos valores críticos de referência) e para admitir a existência de algum tipo de dependência nos dados (pela rejeição da hipótese nula referente ao teste BDS).

5.1.4. SÉRIE SPY

Por fim, e igualmente no contexto dos mercados financeiros, a outra série temporal em análise diz respeito aos valores de fecho relativos ao fundo de investimento (ETF) que replica o índice americano *Standard & Poor's* 500 – SPY[112].

No seguimento do trabalho de Costa *et al.* (2019), integrámos no presente estudo todos os dados disponibilizados no *Yahoo Finance*. Desta forma, os dados em análise são relativos aos valores diários (*business days*) de fecho no período entre 29-01-1993 e 27-09-2019, totalizando 6715 observações.

Os dados históricos relativos à série SPY apresenta-se na Figura 5.9.

[112] Dados diretamente importados de https://finance.yahoo.com/- código 'SPY'

FIGURA 5.9 – Série SPY: Representação gráfica

Complementarmente, apresentam-se em anexo os resultados da decomposição da série (Figura E.19). Das representações gráficas, depreende-se um padrão não linear com alguns regimes distintos, sendo o mais evidente marcado pela crise de 2008 (não excluindo a mudança verificada com os acontecimentos de 11 de setembro de 2001). Esta quebra é destacada a vermelho na Figura E.20 (em anexo) e também visível na representação dos *box-plots* anuais agrupados (Figura 5.10). Nesta última, é evidente como a amplitude amostral e interquartil do *box-plot* relativo ao ano de 2008 é consideravelmente elevada. Tal facto retrata a ocorrência de um fenómeno de quebra (neste caso negativa), onde o valor mínimo desceu abaixo dos mínimos registados a partir de finais da década de 90.

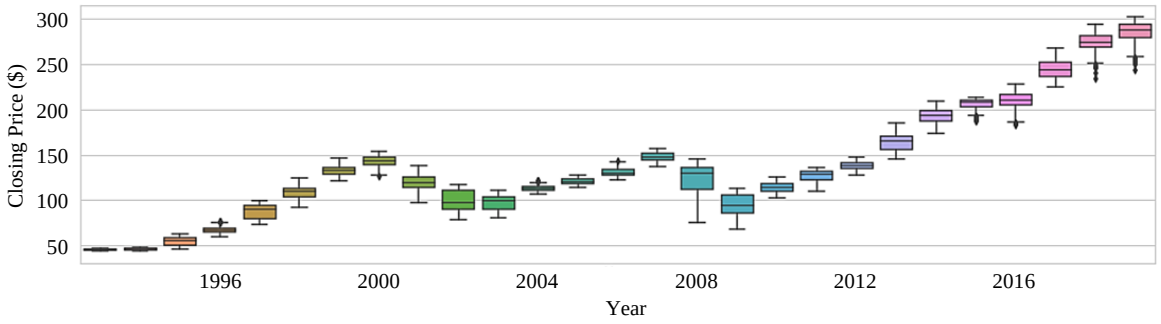


FIGURA 5.10 – Série SPY: Representação gráfica dos *box-plots* anuais

Procurando enquadrar alguns dos factos destacados, em termos de contexto histórico, após a instabilidade política e económica que sucedeu aos atentados de 11 de setembro, com impactos negativos no índice bolsista, só a partir de 2004 se observa uma conjuntura económica favorável. Já nos finais de 2007 verifica-se nova instabilidade nos mercados financeiros resultante da *crise do subprime*. Esta crise financeira, tornada pública em fevereiro de 2008 e resultante da quebra de instituições de crédito dos Estados Unidos, conduziu a uma situação de insolvência de vários bancos, o que constituiu um prenúncio da crise económica à escala

mundial verificada nesse ano, com várias economias a entrar em recessão. O quase colapso do sistema financeiro à escala mundial teve consequências bastante negativas, com grandes instituições financeiras, como a *Bear Sterns* e a *Lehman Brothers*, a entrar em rutura. Depois da queda acentuada em 2008, só em 2010 os mercados mostrariam sinais de recuperação, algo que se mantém numa tendência globalmente crescente.

Tal como as restantes séries em estudo, a SPY é uma série temporal não estacionária, com uma distribuição de dados não normal e evidenciando dependência temporal. A ausência de estacionariedade pode-se depreender da Figura E.21 em anexo, pelo aumento médio dos valores (*Rolling mean*) e variação considerável na dispersão (*Rolling std*) no tempo. A ausência de normalidade é facilmente observada no histograma apresentado em anexo, na Figura E.22, que evidencia uma distribuição multimodal e assimétrica.

As principais estatísticas descritivas são apresentadas na Tabela 5.7, em que o máximo histórico que respeita ao período recente (finais de 2019) se destaca das restantes medidas estatísticas, as quais evidenciam uma maior proximidade aos mínimos históricos verificados no período inicial em análise (1993).

TABELA 5.7 – Série SPY: Estatísticas descritivas

Count	*Mean*	*Std*	*Min*	*Q1*	*Q2*	*Q3*	*Max*	*Kurtosis*	*Skewness*
6715	135.37	61.33	43.41	97.49	124.00	153.74	302.01	0.191	0.810

Embora os valores numéricos dos quartis indiquem alguma simetria no centro da distribuição, tanto o coeficiente de assimetria como a representação da função cumulativa da distribuição (Figura 5.11) mostram que ela é, globalmente, assimétrica.

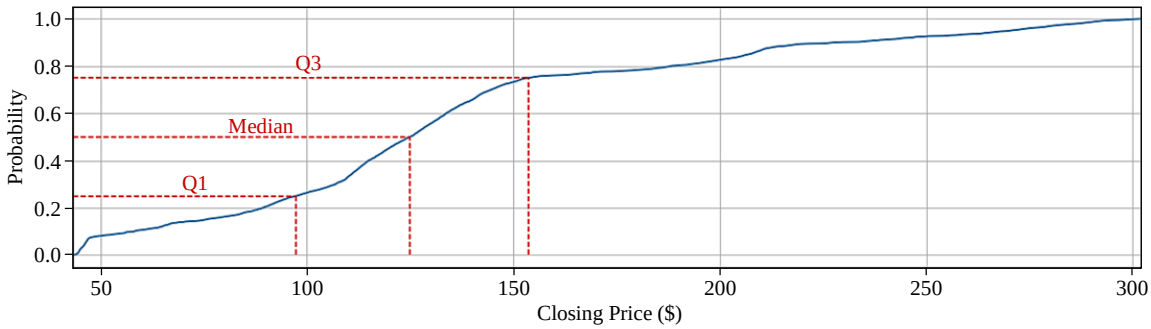


FIGURA 5.11 – Série SPY: Função cumulativa de distribuição

As observações acerca da distribuição de dados podem ser validadas pelos testes de hipóteses correspondentes, cujos resultados se apresentam na Tabela 5.8.

TABELA 5.8 – Série SPY: Testes de normalidade, estacionariedade e independência

	Testes de Normalidade			**Testes de Raiz Unitária/ Estacionariedade**		**Teste de Independência**
	Kurtosis	*Skewness*	*Jarque-Bera*	ADF	KPSS	BDS (Dim. 2 – Dim. 6)
statistic	2.9758	23.9444	743.7061	0.5406	13.3670	14.9813 – 30.9223
p-value	0.0029*	0.0000*	0.0000*	0.9861	----------	0.0000*

*Rejeita-se H_0 para os níveis de significância de 1%, 5% e 10%

Como esperado, para qualquer nível de significância, são rejeitadas a normalidade, a estacionariedade e a independência dos dados relativos à série SPY. Concretamente, existem evidências estatisticamente significavas para: (i) admitir a não normalidade da distribuição de valores da série (com a rejeição da hipótese em qualquer um dos três testes referentes à normalidade); (ii) assumir a não estacionariedade da série (por não ser rejeitada a hipótese nula referente ao teste ADF e pelo valor da estatística referente ao teste KPSS ser superior aos valores críticos de referência) e (iii) inferir sobre não independência, dada a rejeição da hipótese nula de que os dados são iid (teste BDS).

5.2. MODELAÇÃO E PREVISÃO DAS SÉRIES TEMPORAIS

Seguindo os procedimentos metodológicos descritos na Sec. 4.3, apresentam-se nas páginas subsequentes os resultados da implementação computacional, não só numa perspetiva de modelação, mas também de previsão. Para uma apresentação mais clara, dado que os procedimentos requeridos/adotados são bastantes distintos, optámos por apresentar os resultados em função da família de modelos em causa: ARMA, ETS e ANN.

5.2.1. MODELOS AUTORREGRESSIVOS DE MÉDIAS MÓVEIS

A família de modelos autorregressivos é apropriada para descrever séries temporais estacionárias. Deste modo, e como nenhuma das séries em estudo é estacionária, os dados foram pré-processados, seguindo os procedimentos de estabilização descritos na Sec. 4.2.2. Para tal,

cada uma das séries de dados originais foi transformada em novas séries: logaritmizadas, em diferenças e em diferenças de logaritmos (para estabilização da média e da variância).

A representação gráfica das séries transformadas, apresentada no Anexo F.1, permite verificar que o melhor procedimento para estabilizar as séries em estudo consiste em transformá-las em diferenças de logaritmos.

As séries DifLog(CPIAUCSL), DifLog(VMT), DifLog(PSI 20) e DifLog(SPY) apresentam-se na Figura 5.12, onde é possível depreender a obtenção padrões estacionários pela transformação.

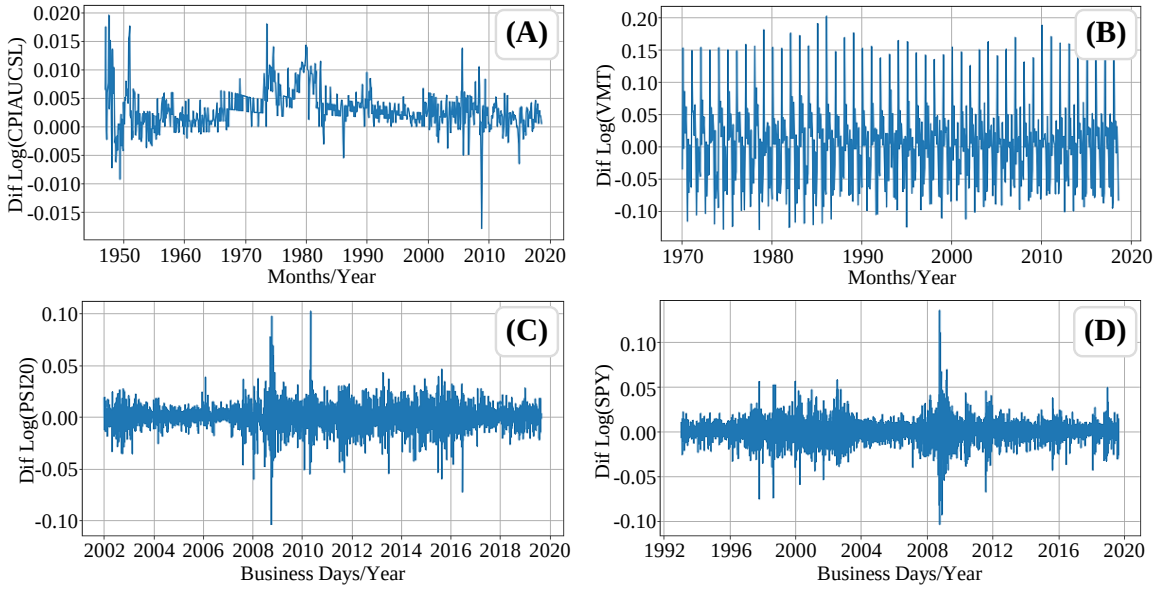


FIGURA 5.12 – Representação gráfica das séries em diferenças de logaritmos: **(A)** DifLog(CPIAUCSL); **(B)** DifLog(VMT); **(C)** DifLog(PSI 20); **(D)** DifLog(SPY)

A representação gráfica da FAC e da FACP em função dos *lags* é também útil para inferir sobre a estacionariedade das séries temporais. Nos correlogramas apresentados em anexo,

verificamos uma certa instabilidade na série CPIAUCSL (Figura F.5), que não ocorre para as séries PSI 20 (Figura F.7) e SPY (Figura F.8), em que os valores da FAC decaem abruptamente para zero. Observe-se o pormenor do correlograma da série VMT (Figura F.6) que confirma o seu comportamento sazonal, com picos nos *lags* 6, 12, 18, … , oscilando entre valores negativos e positivos.

Particularmente, e para efeito de comparação conjunta das quatro séries temporais em estudo, apresenta-se na Figura 5.13 a FAC de cada uma das quatro séries em diferenças de logaritmos, da qual se depreendem alguns dos factos acima referidos.

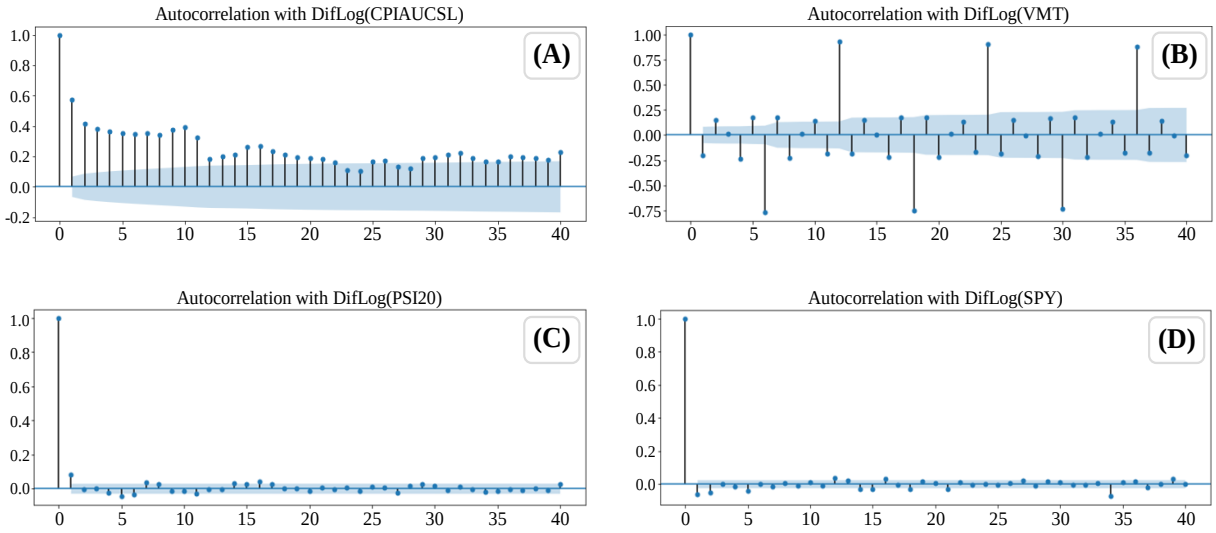


FIGURA 5.13 – Correlogramas das séries em diferenças de logaritmos: **(A)** DifLog(CPIAUCSL), **(B)** DifLog(VMT), **(C)** DifLog(PSI 20), **(D)** DifLog(SPY)

De forma mais objetiva, a estacionariedade pode ser inferida da realização dos testes ADF e KPSS, apresentando-se na Tabela 5.9 os resultados destes testes para as séries em diferenças e em diferenças de logaritmos.[113]

TABELA 5.9 – Testes de Hipóteses ADF e KPSS para as séries em diferenças e diferenças de logaritmos

RESULTADOS RELATIVOS ÀS SÉRIES EM DIFERENÇAS			
Séries Dif	ADF		KPSS
	statistic	*p-value*	*statistic*
CPIAUCSL	-3.4741	0.0087*	1.2827*
VMT	-4.8396	0.0000*	0.0418
PSI 20	-14.4348	0.0000*	0.0939
SPY	-20.1409	0.0000*	0.2234

RESULTADOS RELATIVOS ÀS SÉRIES EM DIFERENÇAS DE LOGARITMOS			
Séries DifLog	ADF		KPSS
	statistic	*p-value*	*statistic*
CPIAUCSL	-4.2883	0.0005*	0.4135**
VMT	-4.7485	0.0001*	0.1216
PSI 20	-15.8199	0.0000*	0.0797
SPY	-14.7617	0.0000*	0.1275

*Rejeita-se H_0 para os níveis de significância de 1%, 5% e 10%
** Rejeita-se H_0 para o nível de significância de 10%

Da análise dos valores obtidos, poderemos considerar válidas as suposições referentes à estacionariedade de cada uma das séries em (primeiras) diferenças e diferenças de logaritmos

[113] Note-se a não estacionariedade das séries em nível verificada anteriormente (ver Tabelas Tabela 5.2, Tabela 5.4, Tabela 5.6 e Tabela 5.8).

uma vez que: (i) é rejeitada a hipótese nula de existência de raiz unitária do teste ADF (para qualquer nível de significância); (ii) o valor da estatística referente ao teste KPSS é inferior aos valores críticos (à exceção da série CPIAUCSL), pelo que não é rejeitada a hipótese nula referente à estacionariedade das séries. Daqui podemos concluir que estamos perante séries integradas de primeira ordem, $I(1)$.[114]

O desafio que agora se coloca é verificar quais as ordens p e q, referentes respetivamente às componentes AR e MA, que melhor se ajustam ao dados, para assim encontrarmos, de entre a família de modelos ARMA, o(s) modelo(s) $ARIMA(p,1,q)$ a usar no processo de previsão. Repare-se, no que respeita à série VMT, dadas as evidências de um comportamento sazonal, fará sentido modelar a série com um modelo SARIMA.

Para simplificação da abordagem e apresentação de resultados, daqui em diante nomearemos as famílias de modelos autorregressivos no seu geral como ARMA.

Embora a análise dos correlogramas referentes às funções FAC e FACP (Anexo F.2) nos possa dar uma ideia para as ordens do modelo a considerar em cada caso (conforme Tabela 2.3), trata-se de análise subjetiva *ad hoc*.

Assim, como exposto na Sec. 4.3.2, o ciclo computacional foi implementado e, de entre os vários modelos analisados[115], apuraram-se os cinco melhores segundo cada um dos três critérios de seleção e, desses, foram selecionados os seis modelos que reuniam maior consenso nos três critérios, apresentados na Figura 5.14.

Sendo um dos objetivos avaliar o 'ajustamento' global da família dos modelos ARMA, considerámos útil analisar a qualidade preditiva dos seis modelos, na tentativa de identificar qual, ou quais, se destacavam.

Para o efeito, foram obtidas as estimativas de previsão *out-of-sample* produzidas, por cada modelo: (i) no caso das séries CPIAUCSL e VMT, foram obtidas previsões para o período de 1 ano (12 observações); (ii) no caso das séries PSI 20 e SPY, foram obtidas previsões para o período aproximado de 1 mês (21 *working days*).

[114] Assumimos a estacionariedade da série CPIAUCSL em diferenças logaritmizadas por ser rejeitada a hipótese nula de estacionariedade do teste KPSS para 10% de significância, bem como a existência de raiz unitária pelo teste ADF.

[115] Foram analisados os casos $ARIMA(p,1,q)$ com $p,q \in \{1,2,3,4,5,6\}$ ou, no caso da série VMT, os casos $SARIMA(p,d,q) \times (P,D,Q)_{12}$ com $p,q,P,Q \in \{1,2,3,4,5,6\}$ e $d,D \in \{1,2\}$.

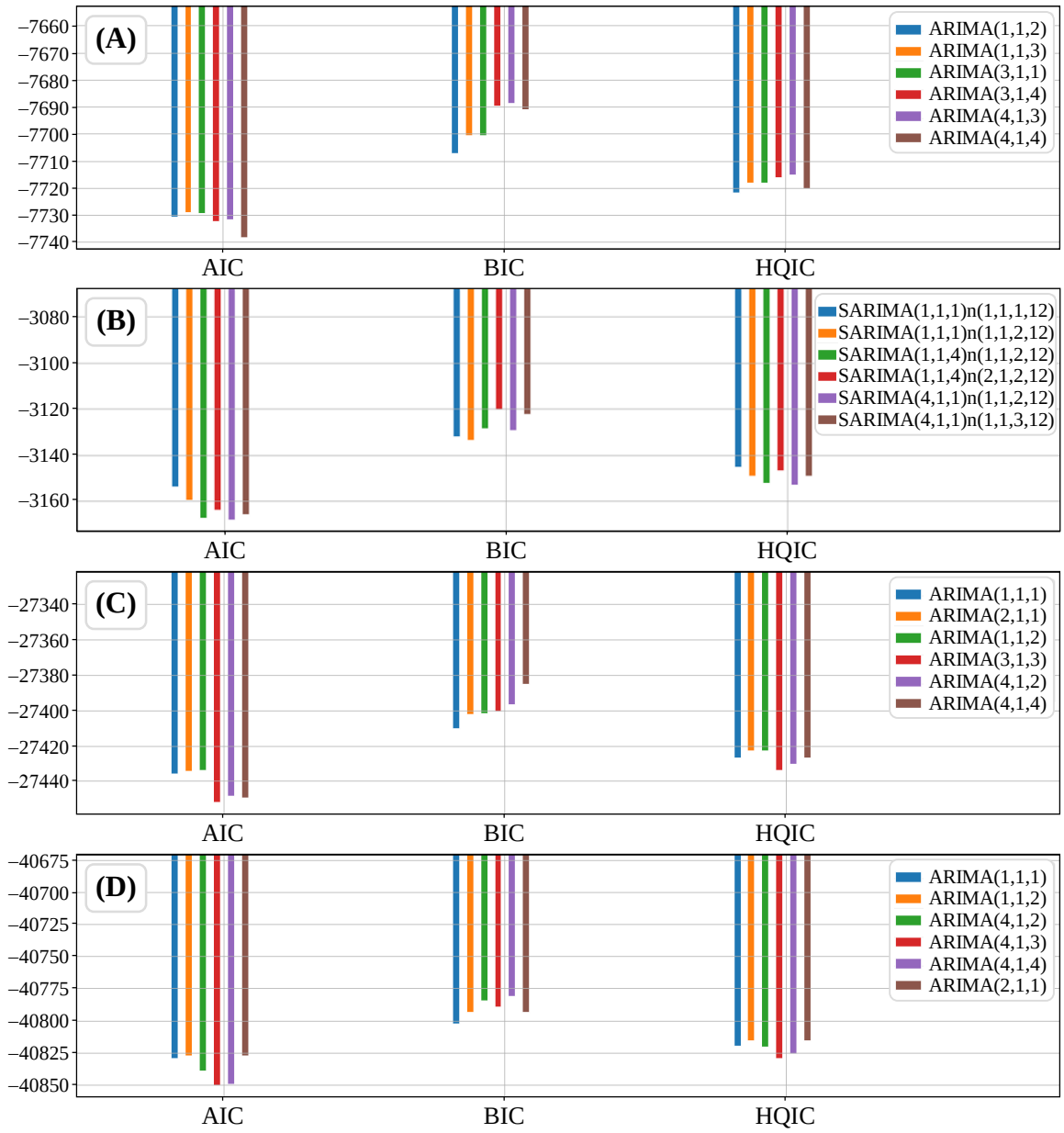


FIGURA 5.14 – Critérios de Informação para modelos ARMA das séries: **(A)** CPIAUCSL, **(B)** VMT, **(C)** PS I20, **(D)** SPY

Na Figura 5.15 encontram-se as respetivas representações gráficas (para cada série), onde, além da comparação dos valores preditos (*forecasting*) com os que efetivamente foram observados, considerámos útil representar (com uma janela temporal das últimas 15 observações) os valores ajustados pelo modelo (*fitting*), donde se pode aferir a adequabilidade de cada modelo aos dados reais (previsões *in-sample*).

Globalmente, verificamos que os seis modelos selecionados apresentam previsões muito similares para todas as séries, já que as linhas de previsão (a tracejado) estão relativamente sobrepostas em todos os casos.

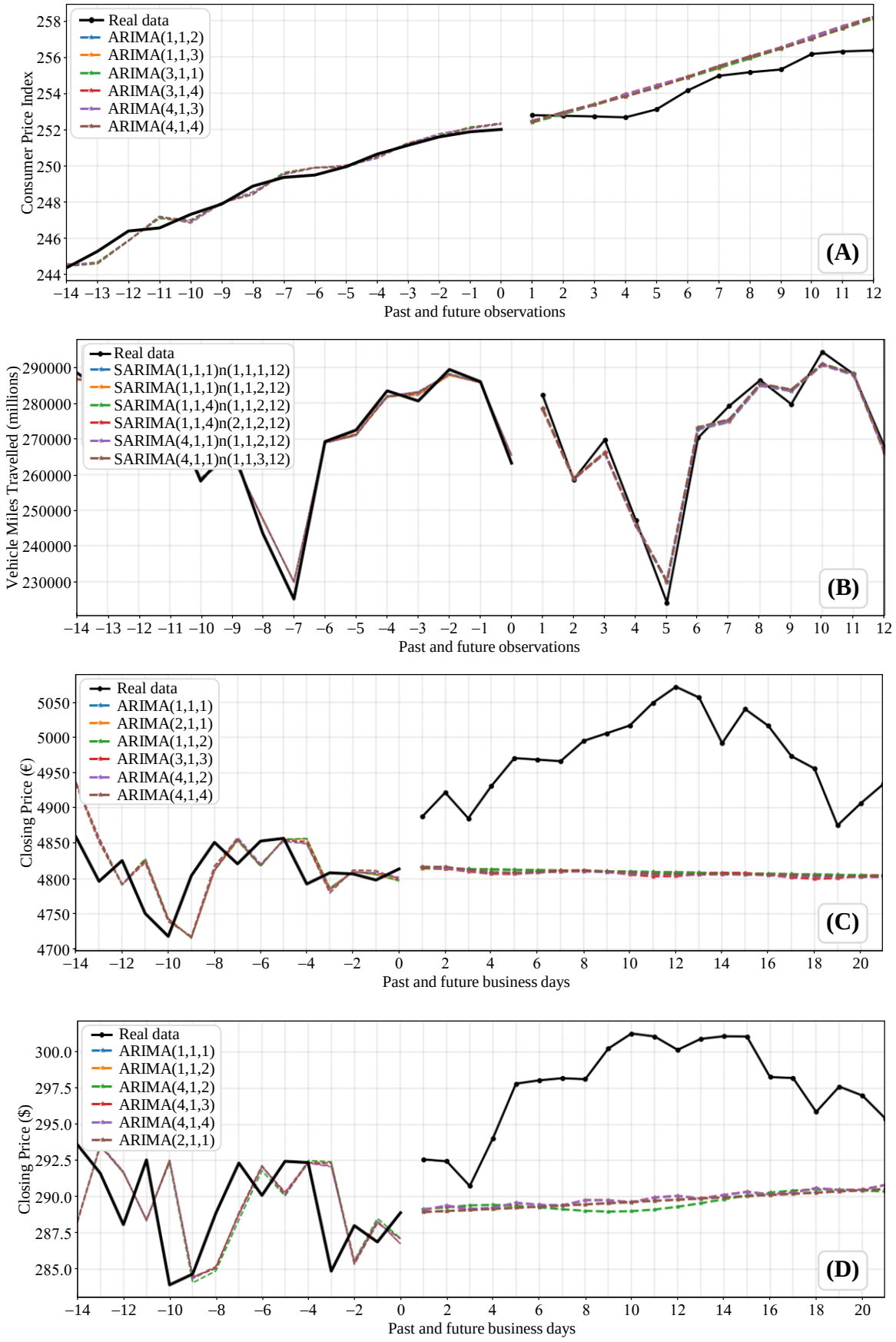


FIGURA 5.15 – Modelos ARMA (*fitting* e *forecasting*) das séries: **(A)** CPIAUCSL; **(B)** VMT; **(C)** PSI 20; **(D)** SPY

Em termo de bom ajustamento, é de destacar os modelos SARIMA no caso da série VMT (as linhas referentes aos cenários de previsão acompanham bastante bem a linha dos dados reais). O mesmo já não acontece para as séries PSI 20 e SPY, onde os modelos autorregressivos não evidenciam uma boa qualidade em termos de ajustamento aos dados reais.[116]

Para selecionar o modelo que, globalmente, melhor se ajusta aos dados de cada uma das séries, e uma vez que os seis modelos selecionados (em cada caso) apresentaram um ajuste semelhante, a análise gráfica foi complementada com a comparação dos valores de erro associados à obtenção de previsões *out-of-sample* para diferentes períodos (de acordo com a frequência dos dados). Isto porque o modelo com melhor qualidade preditiva para uma janela temporal poderá não ser o mesmo quando considerada uma janela de previsão diferente.

Como tal, relativamente às previsões *out-of-sample*, no caso das séries com dados mensais foram obtidas previsões para um mês, um trimestre e um ano. Já para o caso das séries financeiras, com dados diários, foram obtidas previsões para um dia, uma semana (5 dias) e um mês (21 dias).

Para comparação das várias estimativas obtidas, considerou-se o cálculo do MAPE, numa tentativa de uniformizar a escala de valores para avaliar e comparar a qualidade de ajustamento entre as séries com valores de diferentes ordens de grandeza, como exposto na Sec. 4.4. Pela análise desses valores, verificámos não existirem diferenças significativas entre os modelos de previsão para cada série, como seria de esperar pela semelhança gráfica da qualidade dos ajustamentos.

Assim, a seleção do modelo considerado globalmente "o melhor", foi feita pela combinação de três aspetos importantes:

1. os valores obtidos pelos critérios de informação, que são apresentados na Figura 5.14;
2. a performance não desadequada da componente residual, quer pela aproximação a um ruido branco (como se pode ver na Figura 5.16 para os modelos selecionados), quer pela sua distribuição aproximadamente normal de média nula e variância constante, ausência de autocorrelação e baixos valores da estatística de Box-Pierce-Ljung (como se pode ver nos resultados apresentados no Anexo F.4)[117];

[116] Uma discussão mais detalhada dos resultados será feita na Sec. 5.3.

[117] Além da análise gráfica dos resíduos de cada um dos modelos (que pelo facto de se identificarem com um ruído branco, embora com alguns picos em determinados casos, é um bom indicador), importa sublinhar que a tarefa inerente à validação do modelo não se ficou por aqui. Tendo em conta o exposto na Sec. 2.6, foi feita o diagnóstico aos modelos em causa mediante uma análise mais detalhada à componente residual. Contudo, procurando não tornar esta análise ainda mais extensa, optámos por apresentar esse trabalho em anexo (F.4Anexo F.4) e apenas para o modelo selecionado para cada uma das séries, embora se tenham seguido os mesmos procedimentos para outros modelos.

3. a boa *performance* em termos preditivos, quer de acordo com a Figura 5.15, quer pelos valores do MAPE obtidos para diferentes horizontes temporais de previsão, apresentados na Tabela 5.10 para os modelos selecionados.

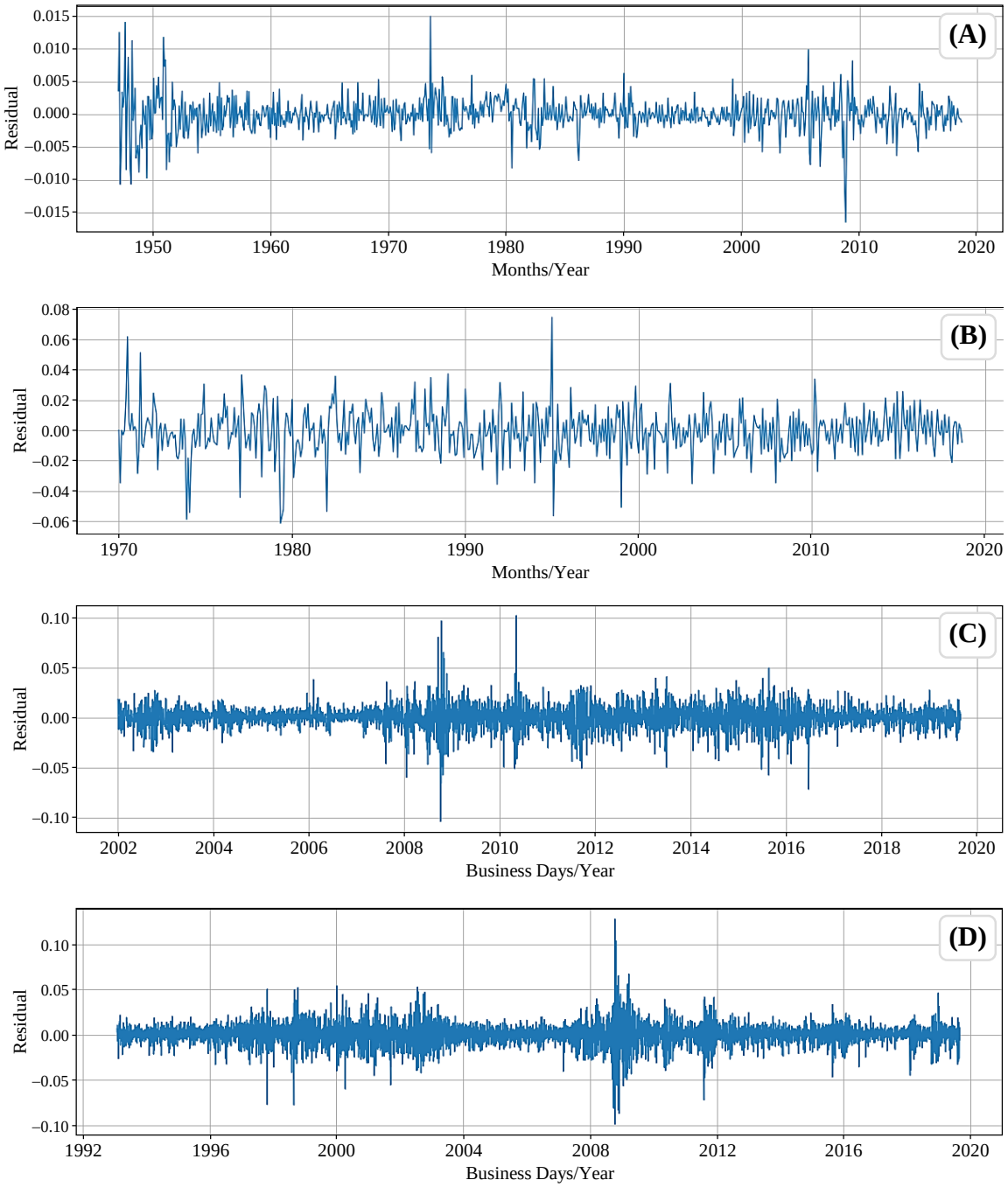


FIGURA 5.16 – Resíduos dos modelos ARMA selecionados para as séries: **(A)** CPIAUCSL – $ARIMA(3,1,4)$; **(B)** VMT – $SARIMA(4,1,1)n(1,1,3,12)$; **(C)** PSI 20 – $ARIMA(4,1,4)$; **(D)** SPY – $ARIMA(4,1,2)$

TABELA 5.10 – Previsões segundo os modelos ARMA (MAPE)

(Série)	**CPIAUCSL**			**VMT**			**PSI 20**			**SPY**		
(Modelo)	ARIMA(3,1,4)			SARIMA(4,1,1) × (1,1,3) ($Tend$: não / $m = 12$)			ARIMA(4,1,4)			ARIMA(4,1,2)		
	1 Mês	1 Trim.	1 Ano	1 Mês	1 Trim.	1 Ano	1 Dia	1 Sem.	1 Mês	1 Dia	1 Sem.	1 Mês
MAPE	0.12%	0.15%	0.37%	1.3%	0.9%	0.98%	1.45%	2.16%	3.29%	1.18%	0.91%	2.63%

Face aos valores do MAPE observados, vemos que à medida que a janela de previsão aumenta os erros de previsão tendem a aumentar, o que nos leva a concluir que os modelos perdem qualidade preditiva com o aumento do horizonte temporal. Como exceção temos o caso dos modelos SARIMA ajustados a séries com sazonalidade que, pela capacidade para captar comportamentos sazonais, conseguem manter (ou mesmo melhorar) a qualidade do ajustamento à medida que o horizonte de previsão aumenta. Para finalizar, note-se que em séries associadas a fenómenos com maior aleatoriedade e incerteza (como é o caso dos mercados financeiros – PSI 20 e SPY) a qualidade da previsão piora significativamente (de 1.45% para 3.29% no caso do PSI 20 e de 1.18% para 2.63% no caso do SPY).

5.2.2. MODELOS DE ALISAMENTO EXPONENCIAL

No que respeita à modelação e previsão usando a metodologia ETS, uma vantagem é o facto de não ser necessária a etapa prévia de transformação dos dados. Contudo, sendo a presença de tendência e de sazonalidade fatores determinantes na escolha dos modelos, além da análise gráfica da decomposição das séries (ver Figura E.1, Figura E.7, Figura E.13 e Figura E.19) considerámos útil o cálculo da 'força de tendência' ($\mathcal{F}_T$) e da 'força de sazonalidade' ($\mathcal{F}_S$), cujos valores se apresentam na Tabela 5.11.

TABELA 5.11 – Valores da 'força de tendência' e da 'força de sazonalidade'

(Série)	**CPIAUCSL**		**VMT**		**PSI 20**		**SPY**	
	$\mathcal{F}_T$	$\mathcal{F}_S$	$\mathcal{F}_T$	$\mathcal{F}_S$	$\mathcal{F}_T$	$\mathcal{F}_S$	$\mathcal{F}_T$	$\mathcal{F}_S$
Modelo Aditivo	1	0	0.9957	0.9199	0.9994	0	0.9998	0
Modelo Multiplicativo	-----	0	-----	0.9304	-----	0	-----	0

Conforme descrito na Sec. 2.2.1, os valores observados evidenciam que qualquer uma das séries apresenta uma forte tendência (valores $\mathcal{F}_T$ iguais ou próximos de 1), ao passo que apenas a série VMT exibe uma forte sazonalidade (valor de $\mathcal{F}_S$ próximo de 1).

Seguindo os procedimentos de implementação computacional descritos na Sec. 4.3.3, avançou-se para a estimação dos modelos. Nesta etapa, para determinar qual o modelo ETS mais apropriado a cada série, além da informação relativa à presença de tendência e sazonalidade, podemos igualmente recorrer aos critérios enunciados na Sec. 2.5, tendo neste caso sido utilizados os critérios de informação AIC e BIC.

Com efeito, no seguimento do ciclo computacional implementado, de entre os nove modelos possíveis (ver Tabela 4.2), foram estimados os parâmetros 'ótimos' para os casos em que era verificada a convergência do modelo e estimado o valor para os dois critérios de informação referidos (ver Anexo F.5).

Seguindo a mesma lógica adotada para os modelos ARMA, de entre os modelos ETS mais adequados (de acordo com os critérios de informação AIC e BIC), na Figura 5.17 apresentam-se o *fitting* e as previsões *out-of-sample* obtidas (*forecasting*) com os modelos ETS selecionados para cada série. Tal como anteriormente, as estimativas de previsão *out-of-sample* respeitam ao período de 1 ano para as séries CPIAUCSL e VMT e ao período aproximado de 1 mês para as séries PSI 20 e SPY.

À semelhança dos modelos ARMA, globalmente destaca-se o bom ajuste dos modelos no caso das séries CPIAUCSL e VMT e um ajuste não tão bem conseguido para as séries PSI 20 e SPY. Particularizando, no caso da série CPIAUCSL, os modelos aditivos com componente de tendência – $ETS(A,N)$ e $ETS(A,M)$ –adequam-se relativamente bem, ao contrário dos modelos aditivos amortecidos – $ETS(Ad,N)$. Já para a série VMT verificamos que qualquer um dos modelos selecionados se adequa globalmente bem aos dados reais considerando as componentes de tendência e de sazonalidade, seja esta última da forma aditiva – $ETS(A,A)$, $ETS(Ad,A)$ – ou multiplicativa – $ETS(A,M)$ e $ETS(Ad,M)$. Finalmente, para as séries financeiras, PSI 20 e SPY, verificamos que os modelos ETS não apresentam uma boa qualidade preditiva. Para qualquer destas duas séries, os valores previstos (*out-of-sample*) não estão próximos dos valores reais nem acompanham as suas flutuações para nenhum dos modelos ETS analisados.

Dos modelos ETS estimados foi selecionado aquele considerado globalmente mais adequado para cada série, onde se teve em conta:

1. os valores obtidos pelos critérios de informação, apresentados na Figura F.11 do Anexo F.5;
2. a capacidade preditiva, observável na Figura 5.17;
3. a representação gráfica dos resíduos, que se identifica globalmente com um ruído branco, embora existam alguns picos que fogem ao padrão, nomeadamente nas séries CPIAUCSL e VMT, como se pode ver na Figura 5.18.

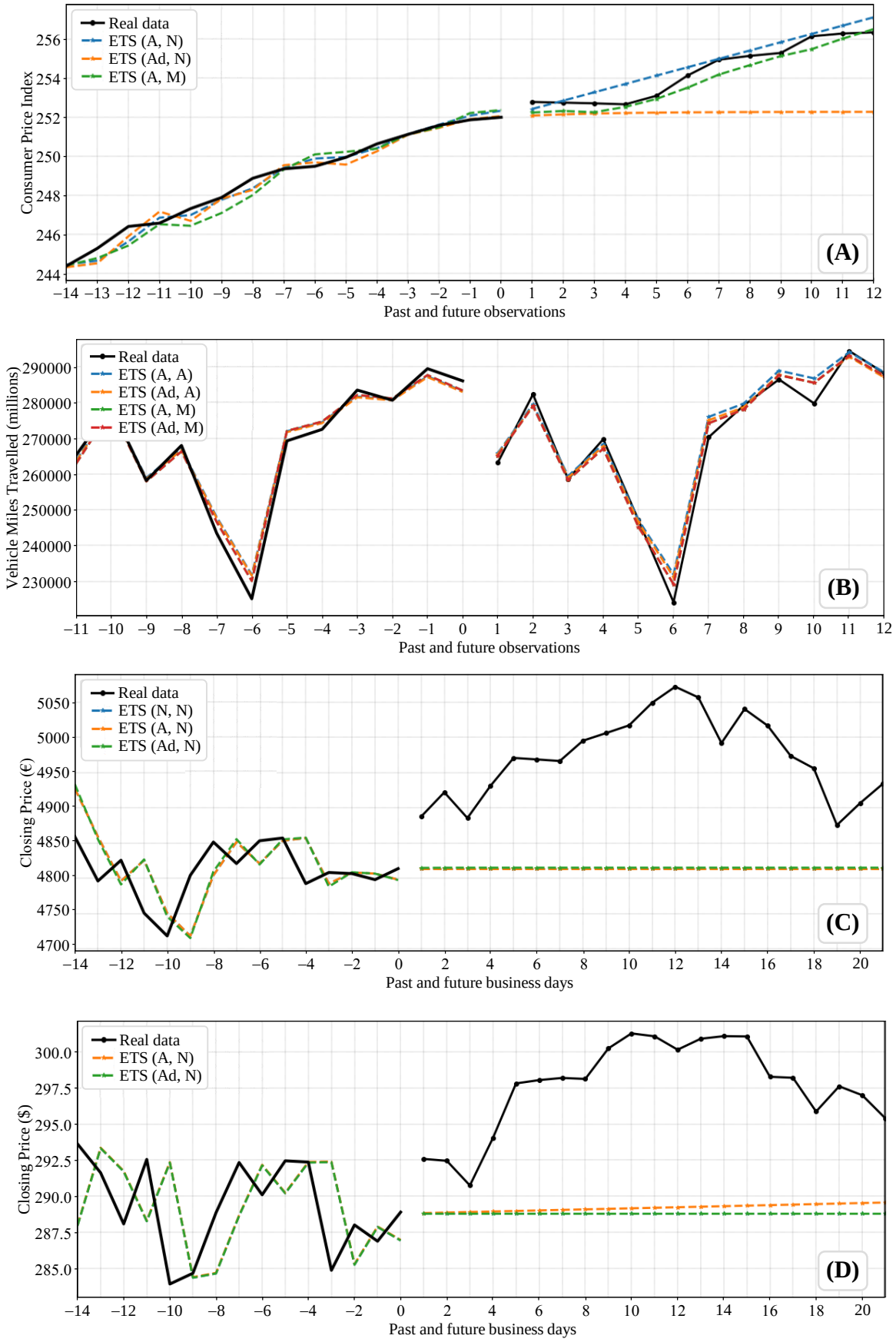


FIGURA 5.17 – Modelos ETS (*fitting* e *forecasting*) das séries: **(A)** CPIAUCSL; **(B)** VMT; **(C)** PSI 20; **(D)** SPY

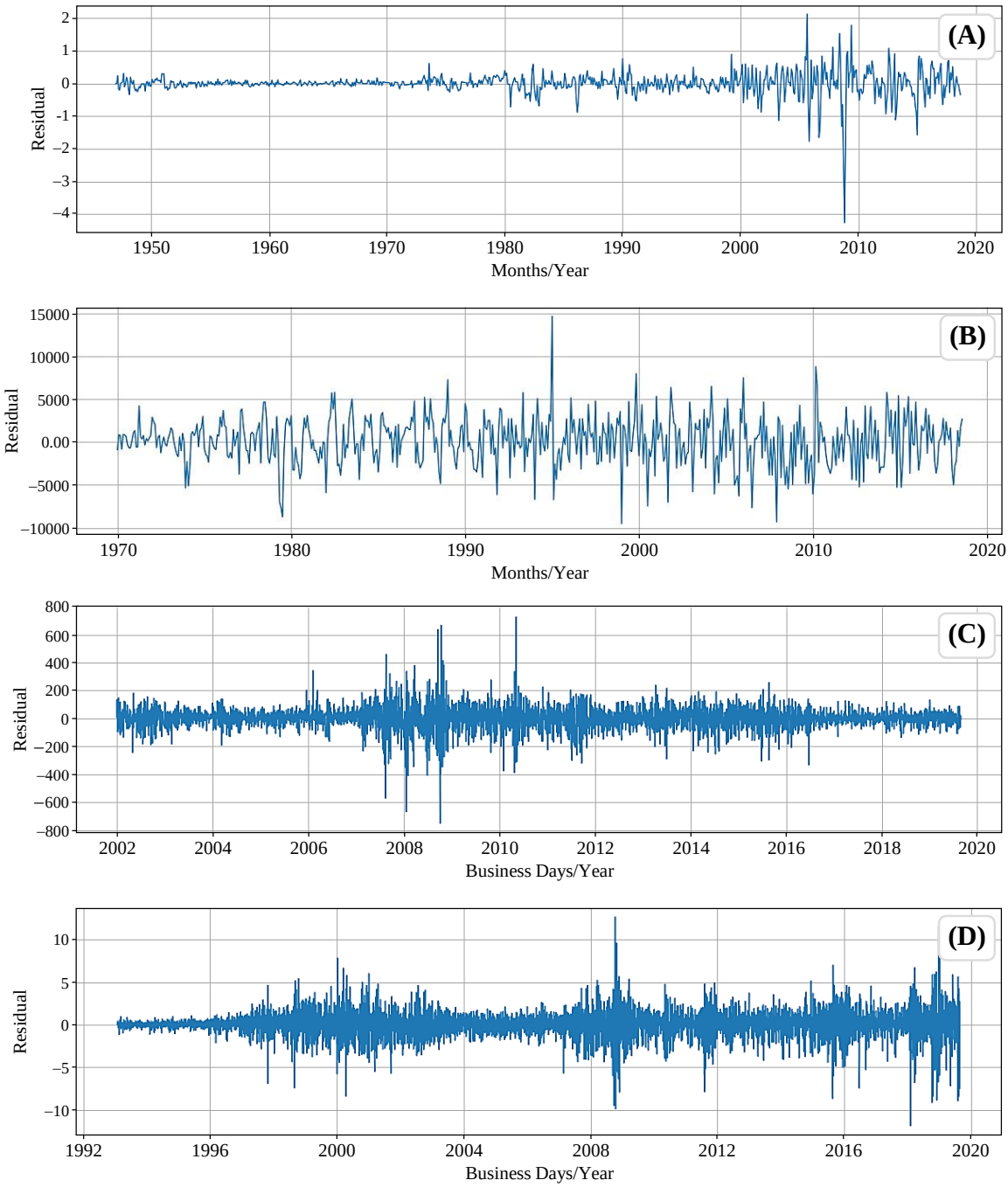


FIGURA 5.18 – Resíduos de modelos ETS selecionados das séries: **(A)** CPIAUCSL – $ETS(A,N)$; **(B)** VMT – $ETS(Ad,M)$; **(C)** PSI 20 – $ETS(Ad,N)$; **(D)** SPY – $ETS(A,N)$

Na Tabela 5.12, além de se identificar claramente o modelo selecionado (identificando os valores ótimos obtidos para cada um dos parâmetros, conforme explicitado na Sec. 2.4.2 e na Sec. 4.3.3) são apresentados os valores do MAPE considerando os diferentes horizontes temporais explicitados anteriormente (um mês, um trimestre ou um ano para as séries mensais e um dia, uma semana ou um mês para as séries diárias).

TABELA 5.12 – Previsões segundo os modelos ETS (MAPE)

(Série)	**CPIAUCSL** (A, N)			**VMT** (A_d, M)			**PSI 20** (A_d, N)			**SPY** (A, N)		
(Modelo)	$\alpha = 1\,,\ \beta = 0.12$			$\alpha = 0.37,\ \beta = 0,\ \delta = 1$ $\gamma = 0.33,\ m = 12$			$\alpha = 1, \beta = 0.29, \delta = 0.24$			$\alpha = 0.95\,,\ \beta = 0$		
	1 Mês	1 Trim.	1 Ano	1 Mês	1 Trim.	1 Ano	1 Dia	1 Sem.	1 Mês	1 Dia	1 Sem.	1 Mês
MAPE	0.14%	0.14%	0.19%	0.74%	0.75%	0.92%	1.49%	2.12%	3.15%	1.28%	1.57%	2.83%

Face aos valores observados, verificamos que os modelos ETS tendem a perder alguma qualidade preditiva à medida que a janela de previsão aumenta (os valores do MAPE tendem a aumentar). Particularizando é de destacar o bom ajuste do modelo ETS estimado para o caso da série CPIAUCSL, sem um aumento considerável do MAPE com o aumento da janela de previsão (valores entre os 0.14% e os 0.19%), algo que também acontece com a série sazonal VMT, apesar de apresentar valores de MAPE superiores (entre os 0.74% e os 0.92%). Já quanto às PSI 20 e SPY a qualidade do ajustamento piora, nomeadamente considerando a janela de previsão a um mês (3.15% e 2.83%, respetivamente). Ou seja, perante fenómenos com maior aleatoriedade e incerteza, a qualidade da previsão dos modelos ETS diminui consideravelmente.

5.2.3. MODELOS DE REDES NEURONAIS

O processo de utilização de redes neuronais para modelação e previsão das séries temporais, centrado na exploração de algumas arquiteturas de DNN (MLP, RNN e LSTM), exigiu um trabalho muito regrado de exploração de vários cenários, sobretudo pelas múltiplas escolhas e combinações possíveis para os hiperparâmetros envolvidos. Não sendo viável a apresentação detalhada de todos os ensaios efetuados, enumeramos de seguida algumas escolhas resultantes da exploração feita, tendo por base as etapas implícitas no processo (descritas na Sec. 4.3.4), antes da apresentação efetiva de resultados (finais).

Inputs

- Em termos do *input* a fornecer, ponderámos entre dar à rede toda a informação disponível de dados históricos ou fazer um corte de alguns dados mais antigos. Do estudo feito, verificámos que não havia vantagens em incluir informação muito antiga, não só pelo grande incremento dos tempos computacionais como pelo aumento dos erros de *cross-validation* (erro de validação e erro de teste), sobretudo quando o conjunto de dados mais antigos não traduz a dinâmica atual da série.

Pré-processamento dos dados

- Verificámos que o alisamento exponencial prévio dos dados históricos pode ser benéfico na obtenção de melhores resultados nos erros de *cross-validation*.
- Quanto à possibilidade de normalização ou padronização, concluímos que, em parte dos casos, os melhores resultados (erros de *cross-validation* menores) foram obtidos quando considerado, cumulativamente, um alisamento exponencial inicial dos dados históricos seguido da sua normalização.
- No caso particular de séries que apresentem um comportamento sazonal, o referido nos itens anteriores não se aplica, porque um alisamento dos dados acaba por 'dissipar' a sazonalidade, não permitindo que a rede aprenda esta característica.

Arquiteturas e respetivos hiperparâmetros

- Foram analisados os resultados provenientes da implementação de três arquiteturas de redes neuronais: MLP, RNN e LSTM. Aqui o fator 'tempo computacional'[118] é sem dúvida o mais marcante, estando a falar-se de valores que podem ir da ordem de segundos/minutos (MLP) para horas, ou até dias (RNN e LSTM).
- Tendo em conta o tempo computacional e os resultados obtidos, verifica-se que: (i) não existe uma qualidade muito superior das arquiteturas RNN em relação às MLP; (ii) as RNN perdem, em muito, em termos de qualidade de ajustamento para as redes LSTM (igualmente recorrentes, mas com 'memória longa').[119]
- O número de camadas ocultas é uma escolha bastante subjetiva, que deve ser sensível ao número de dados utilizados no treino e à natureza dos mesmos. Do estudo feito, verificámos que a escolha de entre três e cinco camadas ocultas, por um lado não compromete a aprendizagem da rede e, por outro lado, evita *overfitting*.
- O número de neurónios por camada oculta é, igualmente, uma escolha subjetiva. Esta deve, em primeiro lugar, ser flexível em relação ao número de *inputs* a dar à rede. Do estudo feito, verificámos que a primeira camada oculta e a última camada oculta não deverão ter um número muito diferente, respetivamente, do número de *inputs* e *outputs* da rede. Mais, a opção de um número maior de neurónios na(s) camada(s) oculta(s) central(is) mostrou-se a melhor (por exemplo, considerando uma rede com 4 camadas

[118] O estudo computacional foi feito numa máquina com Intel i7 CPU - 2.4GHz.

[119] Pelo exposto, não vimos vantagem na apresentação e discussão dos valores do MAPE obtidos para as RNN, uma vez que estes modelos não se mostraram muito superiores aos MLP e são inferiores aos LSTM.

ocultas, com 84 *inputs* e 21 *outputs*, nesta linha de raciocínio, uma distribuição de neurónios por camada oculta seria 100, 120, 80, 40).

- A escolha de um número de neurónios não muito elevado para a primeira camada oculta, como mencionado no item anterior, parece permitir que a rede capte a dinâmica essencial dos dados e o treino incida sobre essa dinâmica. Ao invés, quando considerado um número excessivo de neurónios nessa camada, qualquer "perturbação" é captada pela rede e o treino não se mostra tão eficaz (sendo os erros de *cross-validation* mais elevados).
- Os restantes hiperparâmetros foram ajustados consoante o modelo (de acordo com a arquitetura de redes usada), a série de dados em causa e tendo por base sugestões referidas na literatura (nomeadamente em Brownlee (2017)).

Treino da rede, *Cross-validation* e avaliação dos modelos

- No que respeita ao treino da rede, recorreu-se ao algoritmo de otimização ADAM, face às potencialidades que lhe são apontadas (Kingma & Ba, 2015).
- Sendo esta uma etapa importante no processo de seleção do modelo, foram feitos vários ensaios e testadas diferentes metodologias (Sec. 3.6). Os ensaios feitos vieram confirmar as considerações feitas na Sec. 3.6.4, na medida em que a metodologia *Forward Chaining* se mostrou a adequada.
- Para avaliar os modelos, não havendo necessidade de efetuar comparações com outros dados (nomeadamente entre séries), não se considerou necessário recorrer ao MAPE nesta fase, pelo que se usou o MAE como métrica de análise de erro.
- O critério de paragem foi definido pelo número de épocas de treino, tendo-se fixado entre 150 a 200 de acordo com a natureza dos dados e com a arquitetura de redes (MLP ou LSTM).

Previsão

A partir do estudo computacional feito, definido, treinado, validado e avaliado o modelo para cada uma das séries, seguiu-se a mesma lógica de análise à descrita para os modelos ARMA e ETS. Na Figura 5.19 (referente à arquitetura MLP) e Figura 5.20 (referente à arquitetura LSTM) apresentam-se, para cada caso: (1) a janela de dados usada para treino (zona sombreada); (2) as previsões *in-sample* (linha a laranja) que nos dão conta do *fitting* de cada modelo e (3) finalmente, ajustado o modelo, a partir da amostra de dados destacada a azul, são apresentados os dados das previsões *out-of-sample* (linha a azul), a qual pode ser comparada com os dados reais (linha a preto).

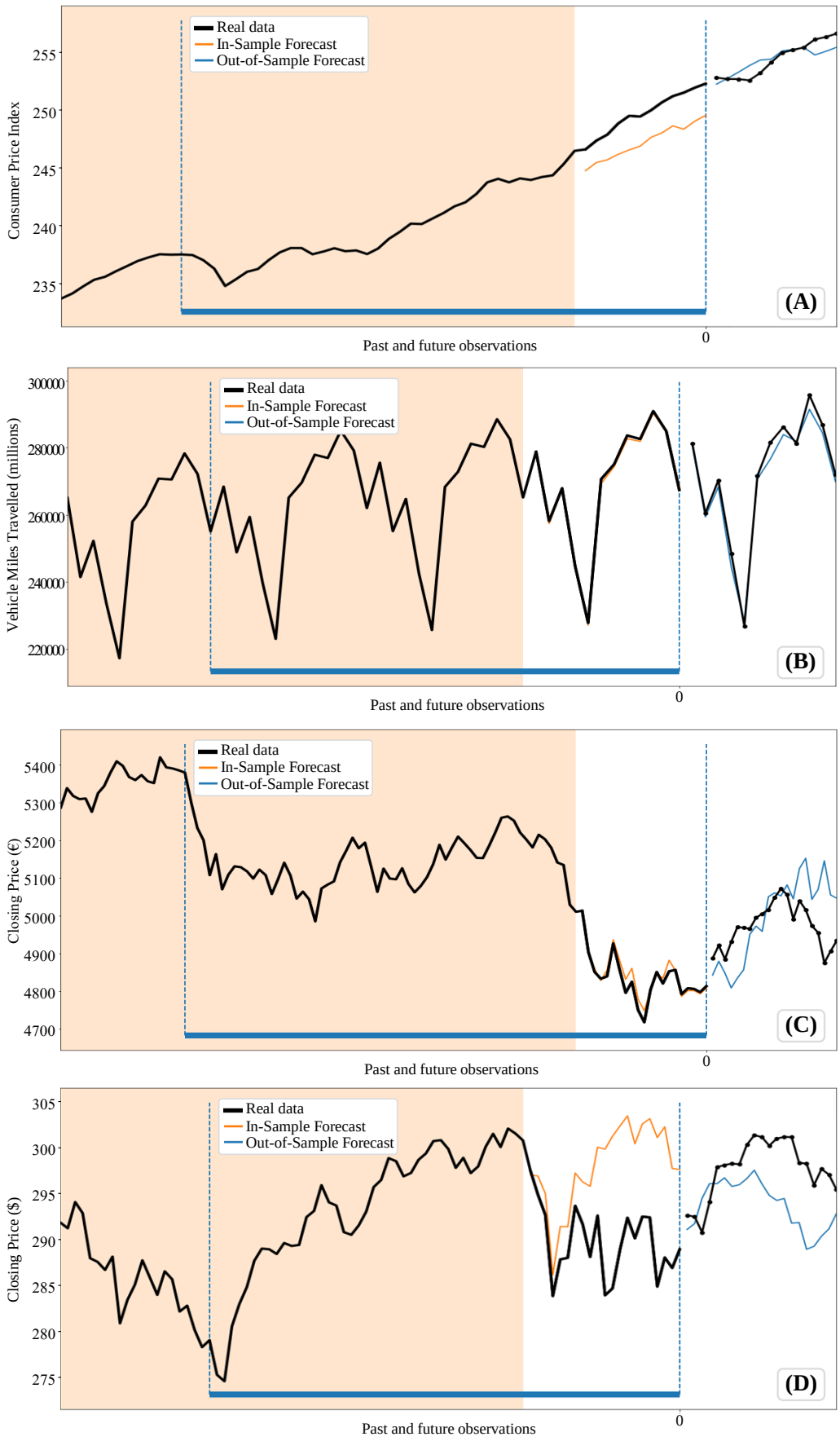


FIGURA 5.19 – Modelos DNN – MLP (*fitting* e *forecasting*) das séries: **(A)** CPIAUCSL; **(B)** VMT; **(C)** PSI 20; **(D)** SPY

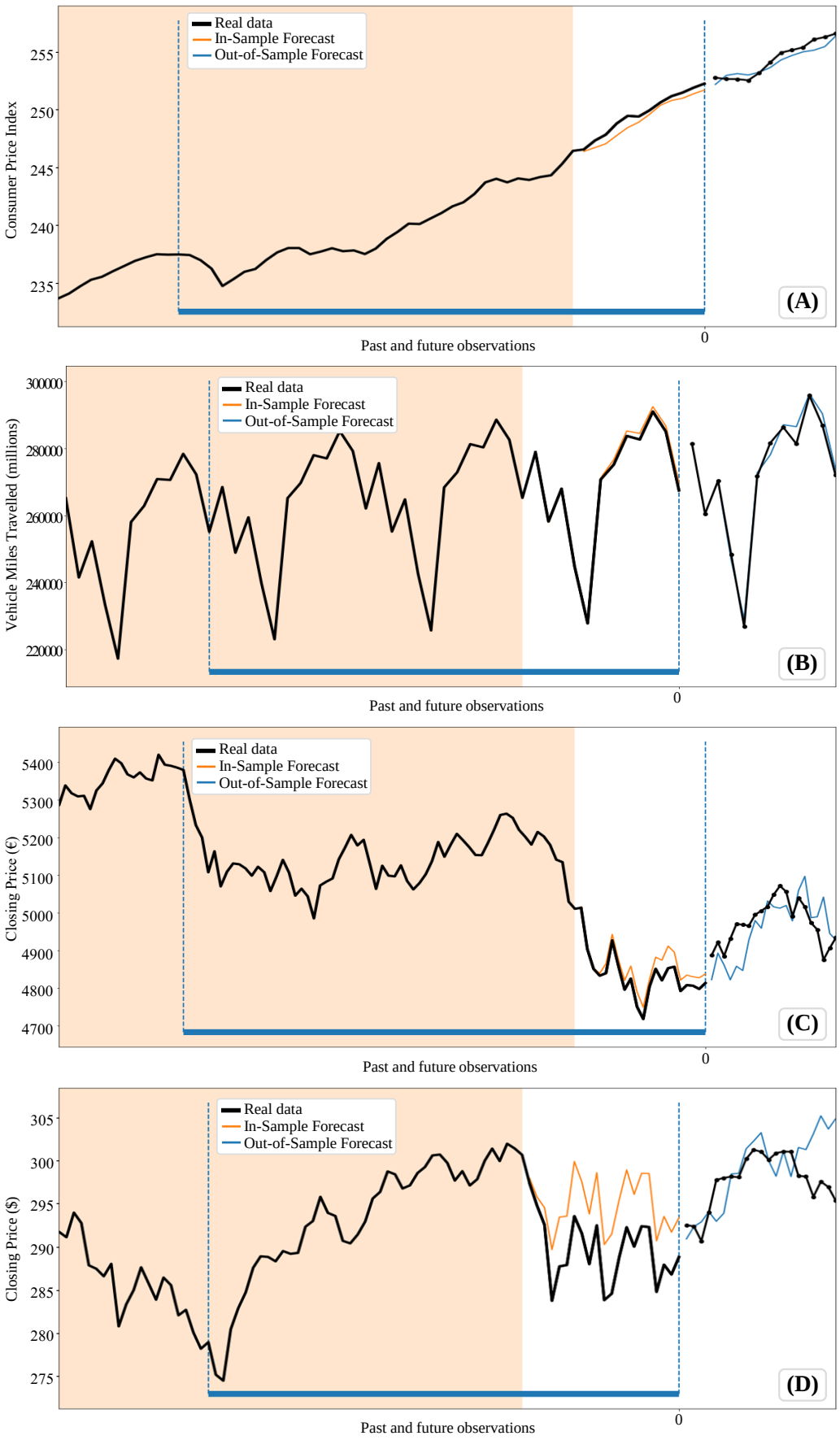


FIGURA 5.20 – Modelos DNN – LSTM (*fitting* e *forecasting*) das séries: **(A)** CPIAUCSL; **(B)** VMT; **(C)** PSI 20; **(D)** SPY

Da análise às quatro representações gráficas da Figura 5.19 (referente à arquitetura MLP), em termos globais, verificamos em qualquer um dos casos existe um ajuste razoável da linha de previsão à flutuação global dos dados reais. No caso das séries CPIAUCSL e VMT a linha de previsão (quer em termos de *fitting* quer em termos de *forecasting*) acompanha bastante bem a linha respeitante aos dados reais. Nas séries PSI 20 e SPY, embora o ajuste não seja tão bom, vemos que a linha de previsão acompanha, em geral, a tendência das séries.

Se nas arquiteturas MLP já é possível falar numa qualidade de ajuste razoável, veja-se a diferença na capacidade dos modelos LSTM (Figura 5.20) para captarem a dinâmica de flutuações nas séries PSI 20 e SPY. Estes modelos, além de manterem/melhorarem a qualidade preditiva para as séries CPIAUCSL e VMT (face às arquiteturas MLP), conseguem melhorar significativamente a qualidade de ajustamento nas séries PSI 20 e SPY, facto que fica visível quando se compara a linha de previsão *out-of-sample* (linha a azul) com a de dados reais (a preto). A capacidade de 'memória longa' da arquitetura LSTM parece ser aqui determinante em conseguir captar alguma da dinâmica nas flutuações dos dados das duas séries financeiras.

Na Tabela 5.13 e na Tabela 5.14 são apresentados os intervalos de valores do MAPE (limite inferior e limite superior do intervalo aparados em 5%) para as arquiteturas MLP e LSTM, respetivamente, considerando os mesmos horizontes de previsão explicitados utilizados nos modelos clássicos.

TABELA 5.13 – Previsões segundo os modelos de DNN – MLP (MAPE)*

CPIAUCSL			**VMT**			**PSI 20**			**SPY**		
MLP A			*MLP B*			*MLP C*			*MLP D*		
1 Mês	1 Trim.	1 Ano	1 Mês	1 Trim.	1 Ano	1 Dia	1 Sem.	1 Mês	1 Dia	1 Sem.	1 Mês
0.13% – 0.26%	0.13% – 0.27%	0.18% – 0.29%	0.21% – 0.38%	0.27% – 0.51%	0.48% – 0.82%	0.83% – 1.39%	1.31% – 1.84%	1.57% – 2.01%	0.55% – 0.94%	0.60% – 1.03%	1.02% – 1.82%

(*) Valores mínimos e máximos (aparados em 5%) obtidos num total de 60 *runs*

TABELA 5.14 – Previsões segundo os modelos de DNN – LSTM (MAPE)*

CPIAUCSL			**VMT**			**PSI 20**			**SPY**		
LSTM A			*LSTM B*			*LSTM C*			*LSTM D*		
1 Mês	1 Trim.	1 Ano	1 Mês	1 Trim.	1 Ano	1 Dia	1 Sem.	1 Mês	1 Dia	1 Sem.	1 Mês
0.11% – 0.21%	0.12% – 0.19%	0.16% – 0.24%	0.04% – 0.10%	0.16% – 0.25%	0.35% – 0.71%	0.87% – 1.41%	1.19% – 1.63%	0.97% – 1.52%	0.48% – 0.83%	0.50% – 0.96%	0.79% – 1.12%

(*) Valores mínimos e máximos (aparados em 5%) obtidos num total de 20 *runs*

Face aos valores observados, é possível verificar que, embora se observe uma diminuição expectável da qualidade preditiva com o aumento da janela de previsão, existe uma maior estabilidade dos valores preditos quando usada a arquitetura LSTM. Comparativamente aos modelos MLP, esta arquitetura não só produz valores mais estáveis em termos de previsão (veja-se a diminuição da amplitude dos intervalos de previsão comparativamente aos modelos MLP), como o aumento dos valores do MAPE não se torna tão significativos à medida que aumenta o horizonte de previsão. Por exemplo, vejam-se os valores a um dia comparativamente aos valores a um mês para a série PSI 20 (considerando os extremos à direita dos intervalos apresentados, de 1.39% para 2.01% para o modelo MLP e de 1.41% para 1.52%, no caso do modelo LSTM). Uma exceção a esta observação é série VMT, cujo modelo LSTM consegue, quase na perfeição, captar a dinâmica de sazonalidade ao curto prazo (erro inferior a 0.10%), capacidade que se vai perdendo com o aumento da janela de previsão (erro que poderá ultrapassar os 0.70%).

5.2.4. UM MODELO DE REDES NEURONAIS MODIFICADO

No sentido de ir mais além e abrir novas perspetivas, em termos de *Data Science* e *Deep Learning*, resultado da reflexão sobre os modelos usados e dos dados obtidos nas seções anteriores, foram pensadas possíveis alterações a introduzir nos modelos de redes neuronais.

Do longo trabalho de investigação feito, sendo inviável apresentar todas as tentativas realizadas, optamos por apresentar algumas considerações/conclusões, procurando, em alguns casos, fundamentar raciocínios e procedimentos.

- Certo é que quanto maior for a quantidade de dados de *input* a dar à rede, maior será o tempo computacional necessário (mais dados a processar). Alinhado com o já observado anteriormente, fomos mais longe nesta exploração, de onde se constatou que ao fornecer um histórico de dados mais antigos, não só estamos a aumentar o tempo computacional inerente ao processamento, como não foram observados ganhos pelo fornecimento dessas informações. Mais, alinhados com o referencial teórico que chama à atenção para o impacto das quebras e/ou mudanças de regime nos dados no processo de modelação (Sec. 2.2.6), comprovámos que, quando a série temporal apresenta padrões diferentes dos verificados no conjunto de dados mais recentes, esse histórico de dados pode mesmo ter um impacto negativo no treino da rede e, consequentemente, na qualidade do modelo;

- Na Sec. 4.2, relativamente ao pré-processamento dos dados iniciais, foram referidas algumas possíveis transformações. Reconhecidos os benefícios destas transformações (nomeadamente a aceleração da convergência na etapa de treino), além das combinações já testadas (envolvendo suavização e normalização), foram analisadas outras possibilidades. Uma delas, inspirada na importância da estacionariedade de uma série temporal nos modelos ARMA, envolveu avaliar a possível existência de vantagens em fornecer à rede uma série estacionária como *input*. Aqui, além de uma redução do tempo computacional, verificámos ganhos consideráveis no *fitting* do modelo obtido, algo que se viria a refletir na qualidade preditiva;
- Na separação usual dos dados em amostra de treino (dados de treino e de validação) e amostra de teste, pela metodologia *Forward Chaining*, os dados mais recentes apenas são utilizados para teste (em nenhum momento foram usados para treino da rede). Como tal, não estamos a dar a conhecer à rede a dinâmica mais atual nem a permitir que esta contribua para o ajuste dos hiperparâmetros e a otimização dos parâmetros do modelo. Por outro lado, verificámos que as distribuições dos erros de validação e de teste eram muito idênticas, ou seja, o erro de validação replicava o erro de teste (a título de exemplo para a série VMT, veja-se na Figura 5.21)[120], havendo informação repetida e cuja análise não trazia mais-valia ao processo de validação dos modelos. Assim, considerou-se a possibilidade de não ter uma amostra de teste e abdicar, no processo de *cross-validation*, da etapa referente ao teste do modelo, sendo o custo computacional substancialmente inferior. Como tal, tomámos todos os dados para amostra de treino, do qual resultam apenas os erros de treino e de "validação cruzada".
- Face à importância da etapa de *cross-validation* na definição, estimação e validação de um modelo, a escolha de metodologias de *cross-validation* mais robustas foi uma hipótese levantada no sentido de poder contribuir para estimação/escolha de um modelo mais adequado. Acontece que, de acordo com o referido na Sec. 3.6 (em particular a Sec. 3.6.4), são apontados algumas limitações à implementação de metodologias mais robustas, como o caso da *Group k-Fold*. Porém, e retrocedendo ao ponto anterior, fornecer uma série estacionária como *input* à rede permite eliminar a limitação apontada à referida metodologia, uma vez que, eliminando a presença de autocorrelação nos dados,

[120] Tal facto torna-se tão mais evidente quanto maior o número de *runs*, mas igualmente válido para poucos, como se pode observar na Figura F.17, em anexo.

torna-se viável implementar a metodologia *Group k-Fold*, que não só é mais robusta (ver Sec. 3.6.4), como implica um tempo computacional consideravelmente inferior. Assim, rompendo com a implementação clássica da metodologia *Forward Chaining* para séries temporais, considerámos na nossa abordagem a utilização da metodologia *Group k-Fold* para uma série de dados estacionária, tomando todos os dados para amostra de treino (como referido acima).

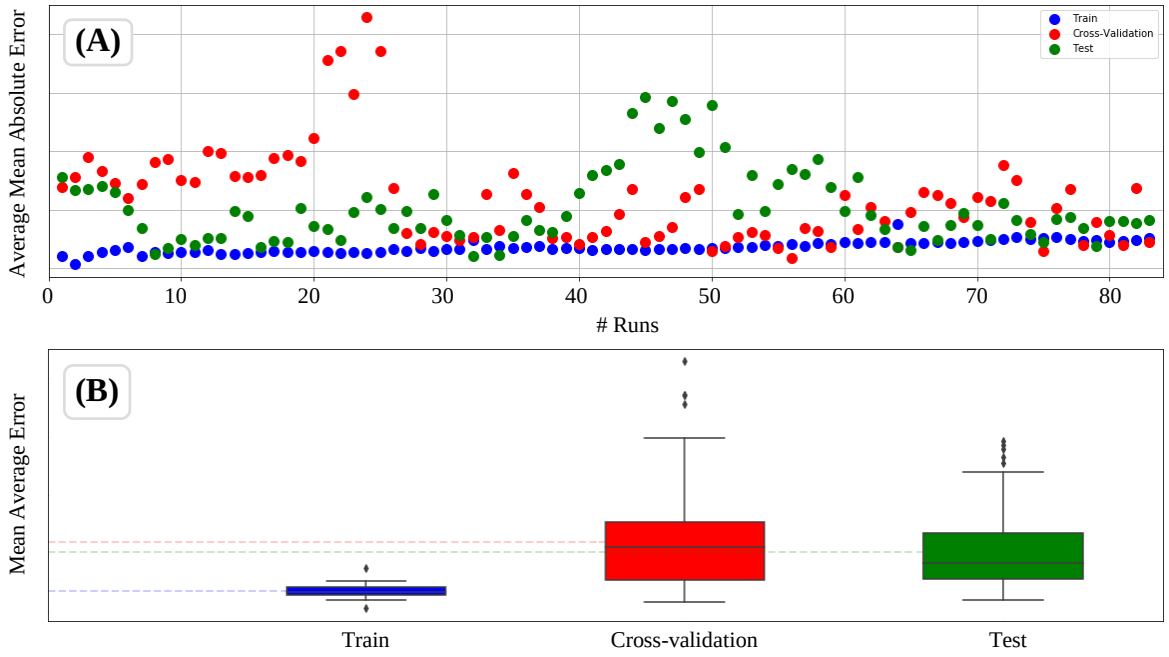


FIGURA 5.21 – Comparação das distribuições dos erros de treino, *cross-validation* e teste (modelo DNN – MLP_Our para a série VMT)

- O poder das arquiteturas LSTM face às arquiteturas MLP foi tópico em análise na Sec. 3.5 (em particular na Sec. 3.5.4), onde se destacou a 'capacidade de memória de longo prazo' das redes LSTM. Porém, foi igualmente referido que estas arquiteturas apresentam um elevado custo computacional, algo que é visto como uma desvantagem. Portanto, seguindo a direção traçada nos itens anteriores, ao dar como *input* à rede uma série temporal estacionária, caracterizada pela ausência de autocorrelação, verificámos que a 'memória temporal' da arquitetura LSTM não produz resultados substancialmente diferentes dos obtidos por implementação da arquitetura MLP, que tem um custo computacional consideravelmente inferior.

Tendo por base o descrito nos pontos anteriores, como resultado (final) do estudo computacional feito, apresentam-se de seguida alguns *outputs* considerados relevantes para a nossa discussão.[121]

Além da implementação matemática da parte referente ao pré-processamento dos dados para a estacionariedade das séries (algo a reverter numa fase final para a obtenção das previsões), um dos aspetos chave que caracterizam a nossa abordagem, na implementação da arquitetura MLP, é a parte referente ao processo de *cross-validation*. Neste sentido, foi nossa preocupação avaliar, até que ponto, esta abordagem poderia conduzir ao treino de um modelo que apresentasse erros de *cross-validation* desajustados, quando comparados com os erros de treino.

O caso que mereceu maior atenção foi o da série VMT. Neste caso houve um maior cuidado em averiguar a existência de algumas limitações desta abordagem relativamente à capacidade da rede em captar a real dinâmica relativa à sazonalidade da série VMT. Conforme a Figura 5.22, embora o valor médio dos dois tipos de erro em analise seja muito próximo, a dispersão do valor dos erros de *cross-validation* é bastante considerável (veja-se a amplitude amostral e interquartil verificadas no *box-plot* a vermelho para o caso da série VMT, comparativamente ao *box-plot* a azul referente aos erros de treino).

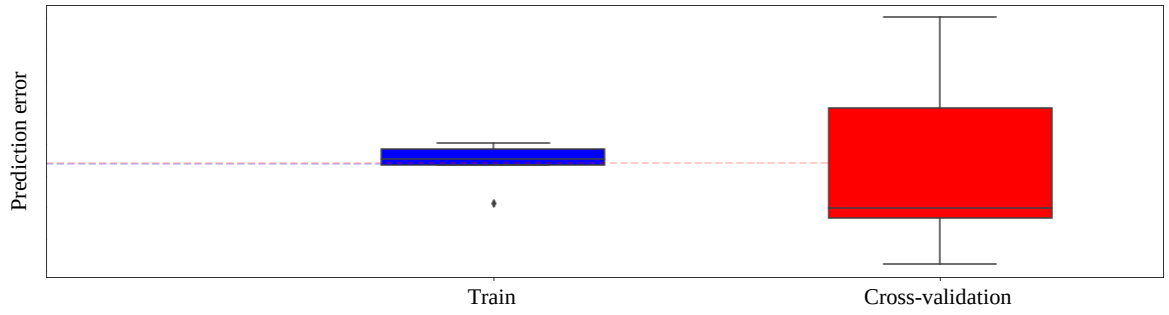


FIGURA 5.22 – Comparação dos erros de treino e erro de *cross-validation* dos modelos DNN – MLP_Our para a série VMT

Em termos preditivos, atendendo às quatro representações gráficas da Figura 5.23, verificamos que, em termos globais existe um ajuste bastante bom da linha de previsão à flutuação dos dados reais. Mais do que a proximidade entre os valores reais e preditos, é de

[121] À semelhança da abordagem gráfica feita para os modelos DNN apresentados na Sec. 5.2.3, a informação relativa aos modelos em discussão nesta secção é complementada no Anexo F.7.

destacar o excelente acompanhamento em termos de monotonia da linha de previsão relativamente à linha dos dados reais.

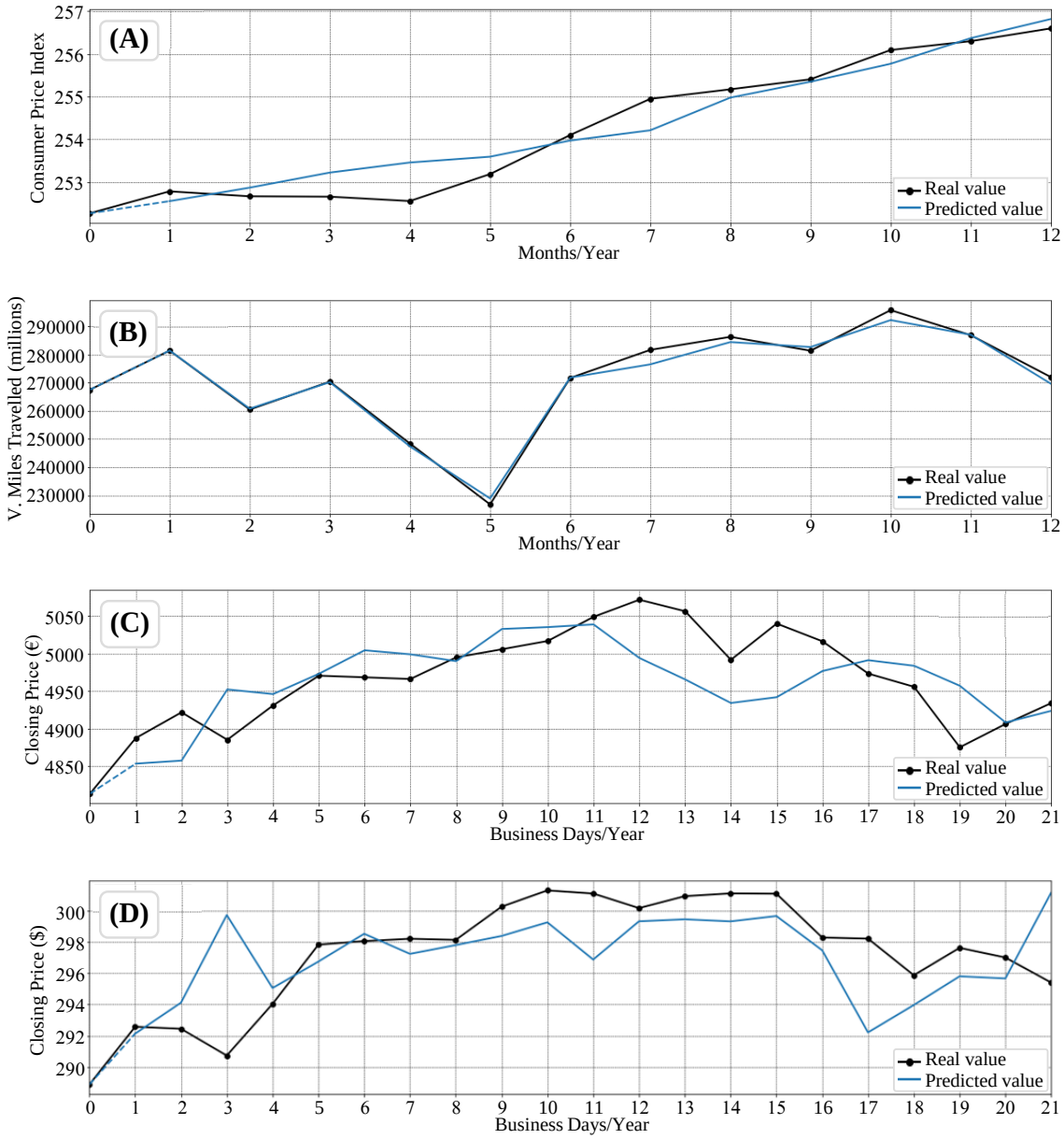


FIGURA 5.23 – Modelos DNN – MLP_ Our (*forecasting*) das séries: **(A)** CPIAUCSL; **(B)** VMT; **(C)** PSI 20; **(D)** SPY

A complementar a informação gráfica, na Tabela 5.15 são apresentados os intervalos de valores do MAPE (limite inferior e limite superior do intervalo aparados em 5%) para cada uma das quatro séries e para os diferentes horizontes de previsão em análise.

Dos valores observados, é possível verificar uma certa estabilidade das previsões resultantes do modelo selecionado, apesar do expectável aumento do erro com o aumento da

janela de previsão. No pior cenário, o relativo à série PSI 20, a amplitude do intervalo de previsão ronda os 0.40%. Este indicador torna-se bastante positivo quando aliado ao facto dos resultados obtidos da nossa abordagem sobre os dados e sobre os modelos conduzirem a valores de erro baixos. De destacar as previsões quase perfeitas para as séries CPIAUCSL e VMT para a previsão a um mês em que, mesmo na pior previsão (aparada), o valor é inferior a 0.1%. Ainda, alinhado com o referido acima, será de realçar alguma amplitude nos valores do MAPE para a série VMT, em que o valor máximo dos intervalos apresentados varia de 0.08% para 0.62%, um aumento de 0.54% da previsão a um mês para a previsão a um ano (valor de amplitude acima do verificado nas restantes séries).

TABELA 5.15 – Previsões segundo os modelos de DNN – MLP_ Our (MAPE)*

CPIAUCSL			**VMT**			**PSI 20**			**SPY**		
MLP_Our A			*MLP_Our B*			*MLP_Our C*			*MLP_Our D*		
1 Mês	1 Trim.	1 Ano	1 Mês	1 Trim.	1 Ano	1 Dia	1 Sem.	1 Mês	1 Dia	1 Sem.	1 Mês
0.03% – 0.09%	0.09% – 0.15%	0.11% – 0.16%	0.01% – 0.08%	0.04% – 0.12%	0.31% – 0.62%	0.39% – 0.92%	0.51% – 0.89%	0.71% – 1.02%	0.11% – 0.65%	0.48% – 0.93%	0.61% – 0.87%

(*) Valores mínimos e máximos (aparados em 5%) obtidos num total de 60 *runs*

5.3. COMPARAÇÃO E DISCUSSÃO DE RESULTADOS

Sem entrar numa repetição de ideias face ao já exposto nas secções anteriores, importa terminar a apresentação de resultados fazendo uma retrospetiva comparativa dos resultados obtidos com os cinco modelos implementados. Para uma apresentação mais objetiva optámos por fazer essa comparação separadamente, tendo em conta as características de cada uma das séries analisadas. Assim, complementando os resultados apresentados anteriormente, que por si só poderiam constituir a base da análise, considerámos vantajosa uma representação gráfica conjunta dos valores reais e preditos pelos cinco tipos de modelo estudados (ARMA, ETS, MLP, LSTM, MLP_Our) e dos respetivos erros de previsão para cada uma das séries estudadas. Os resultados apresentam-se da Figura 5.24 à Figura 5.27.

Da análise à Figura 5.24, relativa à série CPIAUCSL, podemos observar que:

- Dentre os modelos clássicos, verifica-se que ambos conseguem captar a tendência da série, apresentando os modelos ETS um ajuste ligeiramente melhor, com valores de erro inferiores aos dos modelos ARMA;

- Relativamente aos modelos de redes neuronais, os modelos LSTM e MLP_Our são os que apresentam melhor qualidade preditiva, tendo o segundo um custo computacional bastante inferior;
- Globalmente, será adequado dizer que, combinando a qualidade preditiva e o tempo de execução computacional, para a série CPIAUCSL não se justifica a implementação de metodologias de *Deep Learning*. Em casos como este (séries com um padrão claro de tendência, sem presença de comportamento sazonal e ausência de quebras de estrutura), os modelos clássicos ETS (com tendência e sem sazonalidade) são uma boa opção: são de implementação computacional simples e evidenciam uma boa qualidade preditiva.

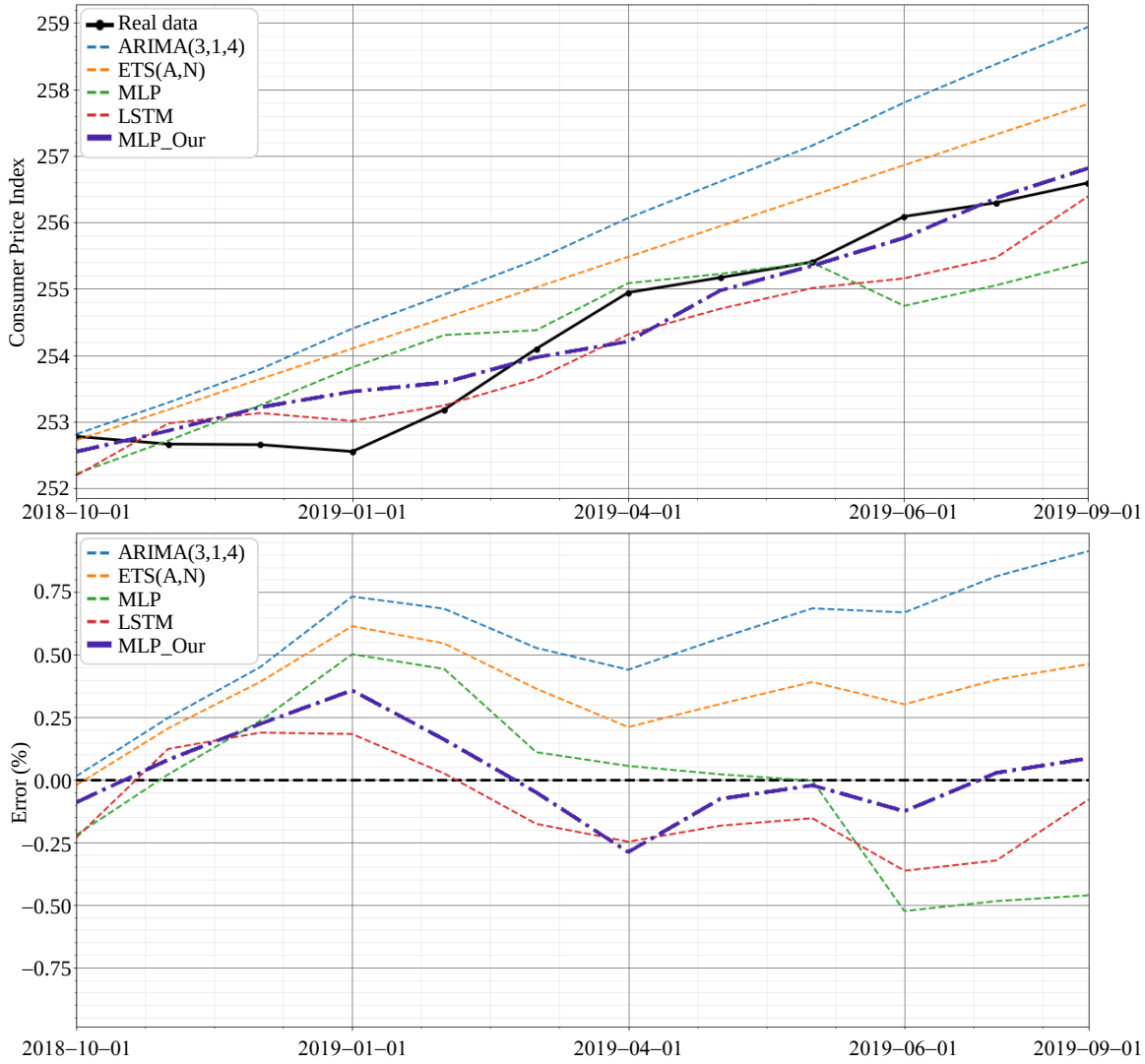


FIGURA 5.24 – Série CPIAUCSL: valores reais, valores preditos e respetivos erros

Relativamente à série VMT, da análise à Figura 5.25 é possível observar que:

- Qualquer dos modelos (clássicos ou de redes neuronais) consegue captar a dinâmica sazonal presente na série;
- Embora existam diferenças pontuais, os modelos clássicos apresentam *performances* muito idênticas. Quer os modelos autorregressivos, na vertente sazonal (modelos SARIMA), quer os modelos ETS (com componente de tendência e sazonalidade) são uma boa opção em termos preditivos. Contudo, importa referir que, pelas múltiplas combinações possíveis relativas à ordem dos modelos SARIMA, o tempo de execução computacional é elevado.
- Dos modelos de redes neuronais, destaca-se a qualidade preditiva dos modelos LSTM e MLP_Our a curto prazo, qualidade que se perde um pouco com o aumento da janela de previsão. Porém, é no modelo MLP_Our que os erros de previsão são, em média, mais baixos;

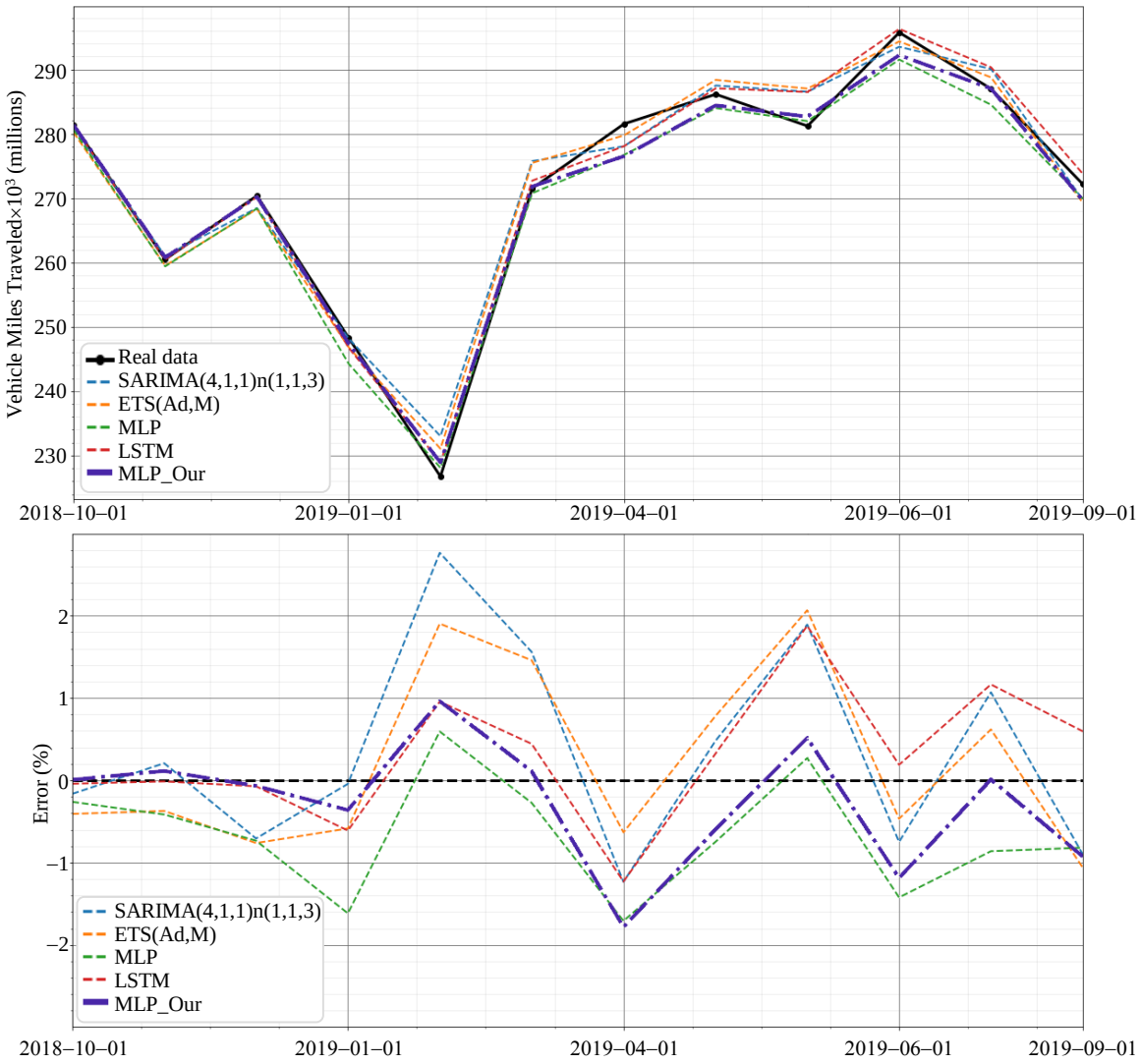


FIGURA 5.25 – Série VMT: valores reais, valores preditos e respetivos erros

- Globalmente, para a série VMT, justifica-se apontar os modelos de redes neuronais, com destaque para o MLP_Our como a melhor opção quando queremos previsões a curto prazo. Contudo, essa qualidade vai perdendo terreno para as metodologias clássicas à medida que aumenta o horizonte de previsão onde, se destacam os modelos ETS (com componente de tendência e sazonalidade) cujo tempo de execução computacional será, igualmente, um ponto a favor, uma vez que estes modelos são, sem dúvida, os de execução computacional mais fácil.

Para finalizar, na Figura 5.26 e Figura 5.27 são apresentadas as previsões e os erros de previsão paras as séries financeiras PSI 20 e SPY, respetivamente. Como estas apresentam características semelhantes, apesar da série PSI 20 ser marcada por uma maior instabilidade, optámos por fazer a comparação e discussão de resultados em simultâneo.

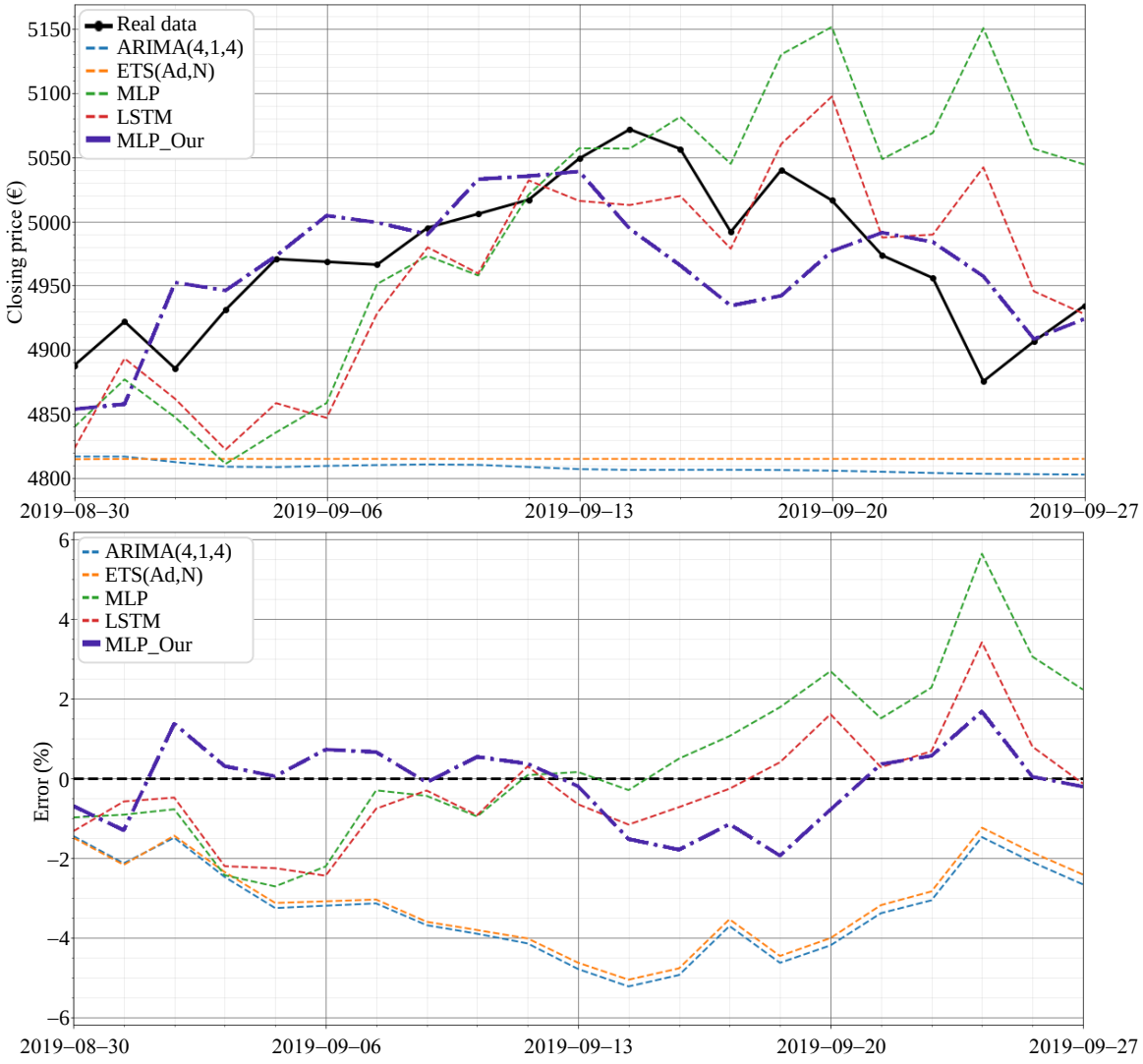


FIGURA 5.26 – Série PSI 20: valores reais, valores preditos e respetivos erros

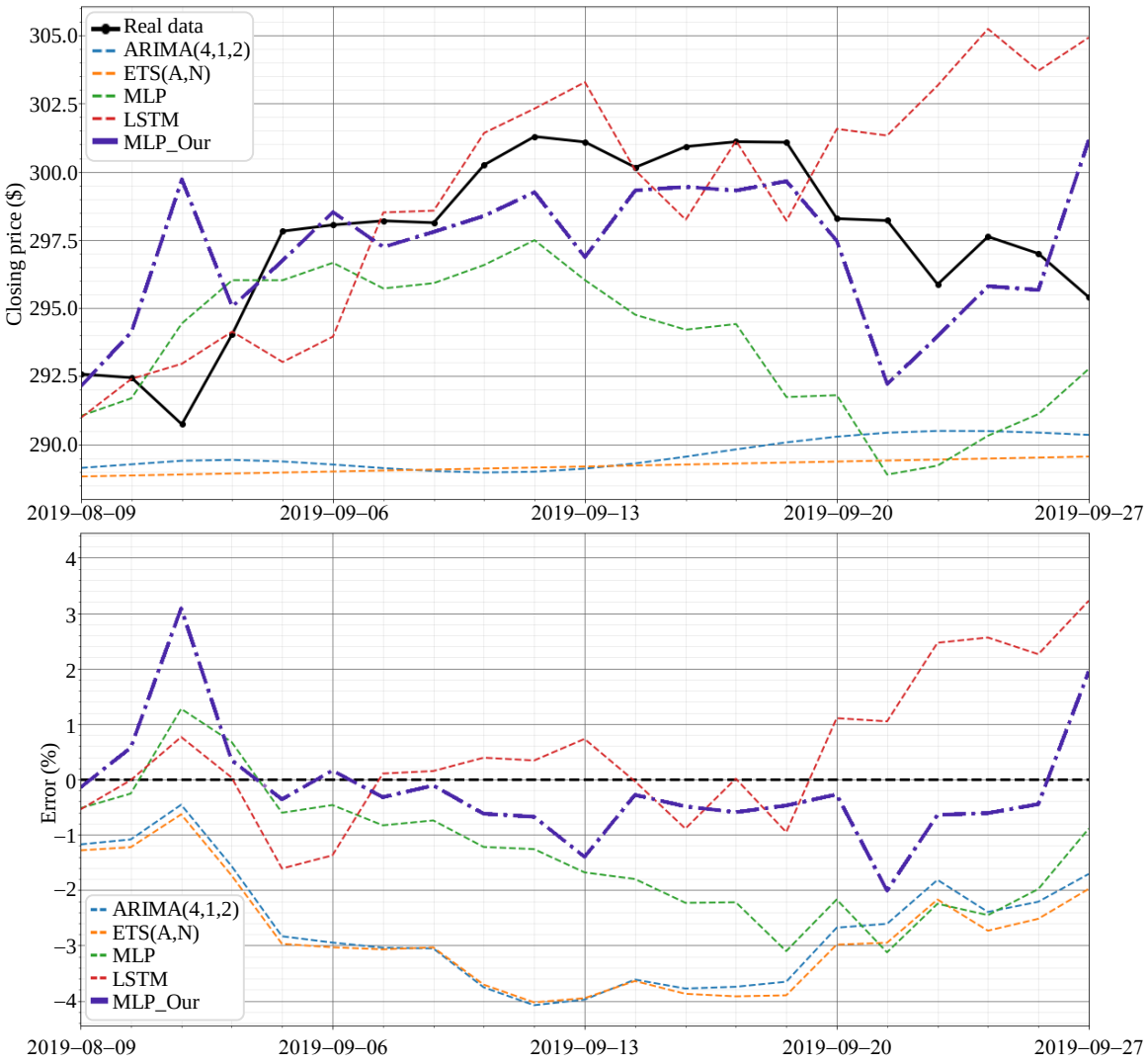


FIGURA 5.27 – Série SPY: valores reais, valores preditos e respetivos erros

Da análise às representações gráficas em causa, é possível observar que:

- Nem todos os modelos se mostram adequados para prever as séries financeiras. Em ambas, podemos concluir claramente que as metodologias clássicas não são uma boa opção. Apesar dos modelos clássicos representados serem considerados os 'melhores' na respetiva família (ARMA e ETS), a dinâmica de flutuações das séries não é minimamente captada por estes modelos, o que resulta em erros de previsão consideráveis. Qualquer dos modelos de redes neuronais apresenta uma qualidade de ajustamento superior aos modelos clássicos;
- Dos modelos de redes neuronais, contrariamente ao verificado nas séries CPIAUCSL e VMT, consegue-se depreender um desempenho bastante inferior dos modelos MLP comparativamente aos modelos LSTM e MLP_Our. Tal torna-se ainda mais evidente à medida que aumenta a janela de previsão, onde os erros de previsão resultantes dos

modelos MLP chegam a ser superiores aos obtidos das metodologias clássicas. Ou seja, na sua versão usual, a arquitetura MLP não é uma boa opção (a ausência de 'memória' não lhe permite aprender a real dinâmica dos dados quando estes apresentam flutuações consideráveis);

- A qualidade dos modelos obtidos da nossa abordagem à implementação das arquiteturas MLP (modelos MLP_Our) destacou-se nestes casos, muito embora os modelos LSTM estejam globalmente próximos. Os modelos MLP_Our não só conseguem captar a dinâmica de flutuações das séries, como apresentam uma qualidade preditiva muito boa (com erro médio, para previsões a 21 dias, abaixo dos 0.80%). Mais, note-se que sendo a série PSI 20 a que apresenta maiores oscilações, é nesta que a qualidade preditiva dos modelos MLP_Our se destaca;
- Globalmente, para as duas séries financeiras, justifica-se a utilização dos modelos de redes neuronais, com claro destaque para o MLP_Our. Apesar dos modelos LSTM não se mostrarem desadequados, a grande diferença está no custo computacional associado, sendo o tempo de execução completamente diferente (na ordem dos minutos/poucas horas para algumas horas/dias), com clara vantagem para os modelos MLP_Our.

CONCLUSÃO

Decorrente das limitações apontadas na literatura às metodologias clássicas na capacidade de modelação e previsão de séries temporais, nomeadamente a sua incapacidade de lidar com alterações que saem fora dos padrões e captar a informação relevante, foi nossa motivação explorar metodologias alternativas para a modelação e previsão em séries temporais económico-financeiras. A nossa escolha passou pela implementação de metodologias de *Deep Learning*, no sentido de avaliar a sua capacidade de 'aprendizagem' ao extrair dos dados a informação relevante, independentemente de estes apresentarem uma dinâmica de flutuações mais, ou menos, complexa. Do trabalho desenvolvido, sintetizemos algumas das principais contribuições conseguidas, não obstante, reconheceremos limitações que podem abrir portas a desenvolvimentos futuros.

I. CONTRIBUIÇÕES REALIZADAS

Alinhados com a nossa motivação inicial e fazendo uma retrospetiva geral pelo trabalho desenvolvido, estamos certos de que os objetivos a que nos propusemos nesta investigação foram alcançados com sucesso, de onde destacamos: (1) a sistematização de um referencial teórico essencial no estudo de séries temporais e de aspetos fundamentais intrínsecos ao *Machine Learning*; (2) a compilação de rotinas computacionais (relativamente automatizadas) aplicadas à modelação e previsão de séries temporais, explorando novas técnicas e/ou desenvolvendo rotinas mais eficientes, beneficiando aqui do atual desenvolvimento computacional; (3) a pesquisa de padrões que permitam identificar quais as metodologias mais adequadas e quais os modelos com melhor qualidade preditiva (confrontando os modelos clássicos ARMA e ETS com os modelos DNN), mediante a utilização das rotinas desenvolvidas em séries temporais com características diferentes.

A delimitar a concretização dos objetivos a que nos propusemos esteve a procura de respostas às questões de investigação referidas na Introdução. Neste sentido, procurando sistematizar ideias e factos discutidos ao longo deste trabalho (nomeadamente no Capítulo 5), serve parte desta secção para apresentar respostas a estas questões, abaixo transcritas.

(A) Face a uma análise cuidada das séries temporais em estudo, que particularidades estão presentes nas mesmas e em que medida poderão condicionar e ser um obstáculo ao processo de modelação e previsão?

Indo ao encontro da literatura, a existência de fenómenos que provocam fortes perturbações no conjunto de dados históricos pode condicionar a qualidade preditiva dos modelos econométricos. Deste modo, para que os objetivos traçados nesta investigação fossem concretizados, impunha-se a análise de séries onde se verificassem a presença de determinadas características. Com base em estudos anteriores, a escolha de séries financeiras foi a hipótese a considerar (séries PSI 20 e SPY). As oscilações verificadas nos mercados, causando fortes volatilidades nos dados históricos, conduzem a particularidades nestas séries temporais, como a não estacionariedade, a presença de um padrão não linear (e não sazonal), a existência de quebras de estrutura ou mudanças de regime, entre outras. Todos estes os fenómenos podem, em certa medida, ser um obstáculo à previsão.

Acresce referir que, para poder efetuar comparações e elaborar uma discussão mais concisa sobre estes aspetos, impunha-se a escolha de outras séries onde estas características não estivessem presentes. Para o efeito, foram consideradas as séries CPIAUCSL (com um claro padrão de tendência) e VMT (com evidência clara de sazonalidade).

(B) De entre as metodologias clássicas, que modelos se ajustam melhor aos dados? Serão esses modelos um suporte para cenários de previsão?

No que respeita a metodologias clássicas, além dos modelos ARMA, considerámos a aplicação das metodologias de alisamento exponencial (modelos ETS) na modelação e previsão das séries temporais. Do estudo empírico feito, os resultados estão alinhados com a literatura científica ao evidenciarem uma clara inadequabilidade destas duas metodologias em séries que apresentem os comportamentos identificados em **(A)**. Desta forma, o nosso estudo vem reforçar que, em séries com comportamentos mais irregulares (que não apresentam padrões definidos de tendência e/ou sazonalidade, por exemplo) as metodologias clássicas não exibem boa qualidade preditiva.

Já no caso de séries com um padrão claro de tendência e/ou sazonalidade, como as séries CPIAUCSL e VMT, os modelos em causa parecem conseguir captar alguma da dinâmica dos dados e apresentar cenários preditivos relativamente ajustados.

Apesar da qualidade preditiva das duas metodologias não ser muito diferente (com alguma vantagem para os modelos ETS), são de destacar algumas desvantagens na aplicação dos modelos ARMA. Primeiramente, pelo facto de nos modelos autorregressivos serem possíveis várias combinações na determinação da ordem do modelo (sobretudo na vertente dos modelos SARIMA), o tempo computacional é consideravelmente superior comparativamente ao necessário para os modelos ETS. Além disso, os cuidados a ter na aplicação dos modelos ARMA, em termos de análise inicial e pré-processamento dos dados (nomeadamente o estudo da estacionariedade), são outra desvantagem relativamente aos modelos ETS, que não requerem cuidados em termos de pré-processamento dos dados.

Assim, no que respeita às metodologias clássicas, são reconhecidas vantagens na aplicação dos modelos ETS, comparativamente aos ARMA, em séries que apresentam um claro padrão em termos de tendência e/ou sazonalidade.

(C) Serão as metodologias de *Machine Learning* (em particular baseadas em arquiteturas de DNN) uma alternativa viável no processo de modelação e previsão, tendo em conta o erro de previsão e o custo computacional?

Esta é talvez uma das questões-chave da nossa investigação, onde se pretende avaliar a capacidade preditiva das metodologias de *Machine Learning* fundamentada nos resultados obtidos dos modelos de redes neuronais MLP e LSTM.

Em virtude dos resultados observados, julgamos ser pertinente abordar esta questão de duas perspetivas distintas: por um lado a qualidade preditiva por outro o tempo computacional implícito.

Em termos de qualidade preditiva, quando comparados com os resultados obtidos pelas metodologias clássicas, as arquiteturas MLP apenas representam um ganho substancial nas séries marcadas por alguma instabilidade (como as referentes aos mercados financeiros), enquanto a aplicação de arquiteturas LSTM permite uma redução do erro de previsão (bem como intervalos de previsão mais estáveis) em todas as séries analisadas. Podemos afirmar que, de facto, a capacidade de memória (longa) implícita nas arquiteturas LSTM permite uma melhor aprendizagem dos dados da série e, com isso, uma melhoria da qualidade preditiva do modelo.

Acontece que, além da vertente académica desta investigação é nosso propósito a sua aplicabilidade e exequibilidade em contexto real. Aqui não poderemos deixar de reconhecer a desvantagem da aplicabilidade destas arquiteturas: o tempo implícito na execução e na obtenção de resultados é significativamente superior. Ou seja, apresar das arquiteturas LSTM

evidenciarem um forte potencial preditivo, o tempo de execução computacional, em máquinas 'usuais', é da ordem de várias horas/dias, algo que, num contexto prático-real, não pode ser descurado.

Deste modo poderemos assumir aqui um "sim" e um "não" como resposta a esta questão. Sim, sem dúvidas as metodologias de *Machine Learning* são promissoras em termos de qualidade preditiva. Por outro lado, estas metodologias requerem máquinas com poder computacional elevado, para diminuir o tempo de execução, o que é uma limitação.

(D) Que implementações/alterações poderão ser feitas, numa perspetiva de inovação, com vista quer à obtenção de modelos ajustados e com boa qualidade preditiva, quer ao desenvolvimento de métodos computacionalmente mais eficientes?

Face às limitações destacadas em **(C)**, e reconhecidas a vantagens na implementação das metodologias de *Machine Learning*, foi nestas arquiteturas que decidimos investir a nossa habilidade na identificação de aspetos passíveis de alterações. Além da introdução de modificações 'teóricas' em termos de implementação metodológica, com vista a obter erros de previsão menores, foi nossa preocupação encontrar estratégias que permitissem baixar o custo computacional.

Neste sentido, face à enorme diferença de tempos de execução das arquiteturas MLP e LSTM (em detrimento das segundas), fizemos vários ensaios no sentido de introduzir modificações que permitissem melhorar o desempenho das primeiras.

A vantagem das redes LSTM está na longa memória 'temporal' na aprendizagem dos dados. Contudo, se a série temporal for estacionária (em particular, caracterizada pela ausência de autocorrelação) não se justifica a necessidade de 'memória' pelo que o recurso a uma arquitetura LSTM não terá vantagens significativas face à arquitetura MLP, sobretudo quando tido em conta o custo computacional associado.

Por outro lado, sendo a série estacionária, a aplicabilidade de metodologias de *cross-validation* mais robustas, como a *Group k-Fold*, torna-se viável e permite reduzir, ainda mais, o tempo de execução computacional.

Ainda, e é aqui que entra o olhar humano sobre os dados, analisar com detalhe qual a janela de dados a fornecer como *input* de treino, dando a conhecer à rede as observações mais recentes em detrimento do histórico mais longínquo (sobretudo quando estes dados apresentam uma dinâmica diferente da atual), permite não só melhorar a qualidade preditiva como acelerar o processo de previsão (quantos mais dados, maior é o tempo de execução).

Estas alterações permitiram a criação de uma abordagem própria às arquiteturas MLP que não só conduziu a resultados melhores em termos de erro de previsão (comparativamente aos restantes modelos analisados e independentemente da série de dados), como reduziu consideravelmente o tempo computacional (da ordem de dias para minutos).

Em suma, as alterações introduzidas na nossa abordagem mostraram-se bastante promissoras, passando pela articulação da inteligência artificial com a inteligência humana, combinando o poder computacional, numa perspetiva de aprendizagem feita pela máquina, com o nosso olhar crítico sobre os dados e sobre os modelos.

(E) Existirão padrões que permitam selecionar as melhores metodologias/modelos de previsão, de acordo com as características/natureza dos dados?

Sem dúvida que sim, não se justifica utilizar metodologias complexas e de implementação morosa, quando metodologias clássicas, de implementação bem mais simples, conduzem à obtenção de modelos com boa qualidade preditiva.

De um modo geral, se a série apresenta uma dinâmica simples, com clara evidência de tendência e/ou sazonalidade, as metodologias clássicas (muito em particular as metodologias de alisamento exponencial) mostram-se suficientes por apresentarem previsões relativamente ajustadas. Os modelos ETS não só são de fácil compreensão do ponto de vista teórico e de baixo custo de execução computacional, como não exigem um grande trabalho em termos de análise prévia e de pré-processamento de dados (apenas a decomposição da série e a determinação da "força de tendência" e "força de sazonalidade"). Não obstante, sobretudo para previsões com horizonte de curto prazo, os modelos resultantes da nossa abordagem à arquitetura MLP trazem alguns benefícios em termos de erro de previsão, apesar da complexidade de implementação e do custo computacional serem superiores.

Porém, a grande diferença está no estudo de séries que apresentam padrões bastante irregulares e com as características referidas em **(A)**, como o caso das séries financeiras. Nesses casos, apesar de estarmos perante um custo computacional e com uma complexidade de implementação muito superiores, a nossa abordagem à arquitetura MLP exibe uma qualidade preditiva francamente superior à dos modelos clássicos (ARMA e ETS), já que o valor do MAPE reduz mais de $\frac{1}{3}$ dos valores obtidos para esses casos, justificando-se então a implementação de modelos de redes neuronais.

Em suma, face ao cuidado tido na redução do tempo computacional necessário na nossa abordagem à arquitetura MLP, sem dúvida que, em séries com comportamentos atípicos, a aplicabilidade desta abordagem é efetivamente uma boa opção.

II. LIMITAÇÕES E DESENVOLVIMENTOS FUTUROS

Certos do contributo da nossa investigação para o debate científico relacionado com a modelação e previsão de séries temporais, reconhecemos, todavia, algumas limitações que, associadas a novas ideias, poderão abrir perspetivas para desenvolvimentos futuros.

Num olhar transversal pelo trabalho concretizado, estamos cientes da existência de outras perspetivas de análise. No entanto, traçamos a nossa linha de investigação, tendo tido o cuidado de justificar as opções tomadas.

Sobre algumas dessas opções, uma primeira observação vai para o tipo de dados escolhidos: apesar de termos diversificado e enriquecido a parte empírica com a análise de séries com características distintas, poder-se-ia integrar, no nosso estudo, uma série temporal com um volume significativo de dados, uma vez que não foi devidamente explorado o manuseamento de um grande volume de dados (afeto à atual realidade de *Big Data*).

Já no processo de modelação, estando conscientes de que investimos mais esforços na exploração dos modelos de DNN, o mesmo não foi feito nas metodologias clássicas. Por exemplo, poder-se-ia também ter investido na exploração de vantagens resultantes da seleção/corte nos dados (nomeadamente quando a série apresenta regimes diferentes), ou ainda na procura de outras estratégias de seleção dos 'melhores' modelos.

Ainda no processo de modelação, estamos conscientes de alguma aleatoriedade nos valores de previsão obtidos pelos modelos DNN, devido à rede se comportar como uma *black-box*. Nestes modelos há sempre uma componente que não goza da mesma objetividade de modelação dos modelos ARMA e ETS, a qual não se consegue captar nem controlar. Daí que a análise tenha sido feita pelo intervalo do erro associado à previsão e não por um modelo em concreto, como seria desejável.

Por fim, relativamente ao estabelecimento de algumas conjeturas sobre "Quais os melhores modelos?", reconhecemos algumas limitações por estarmos perante conclusões resultantes da análise de apenas quatro séries temporais. No entanto, não seria comportável introduzir mais séries neste estudo, pela extensão em que se poderia incorrer. Como tal, confiantes nas conclusões apresentadas, não pretendemos fazer nenhum tipo de inferência formal, mas sim apontar diretrizes que poderão servir de base a trabalhos futuros.

Perspetivados na sempre possível e desejada melhoria do trabalho realizado, em sintonia com o acreditar na importância, utilidade e pertinência do estudo desenvolvido, conseguimos identificar os seguintes pontos de partida para desenvolvimentos futuros.

- Em primeiro lugar, face à atualidade das temáticas relacionadas com *Big Data*, seria importante incorporar neste tipo de investigação a análise de dados em massa e perceber como manusear um grande volume de dados.
- Compreendido o impacto que a etapa de validação cruzada poderá ter na seleção dos modelos, propomo-nos investir na melhoria das rotinas computacionais por implementação de uma metodologia de *cross-validation* nos modelos clássicos, no sentido de obter informação mais consistente e que melhor sustente a seleção dos modelos.
- Mantendo esta linha de investigação, será igualmente nosso objetivo dar continuidade à exploração das metodologias *Deep Learning*. Por um lado, partindo do trabalho feito, será importante melhorar as rotinas computacionais desenvolvidas, tornando-as ainda mais eficazes e automatizadas na otimização dos hiperparâmetros, nunca desvalorizando o custo computacional inerente. Por outro lado, estando alinhados com a literatura recente, tal como destaca uma equipa de pesquisa internacional da TU Wien (Viena), IST (Áustria) e MIT (EUA) que desenvolveu um novo sistema de IA, "menos neurónios, mais inteligência", baseado no cérebro de animais minúsculos (Lechner *et al.*, 2020). Há ainda muito espaço para melhorias no que respeita ao *Deep Learning*, onde reduzir massivamente a complexidade e melhorar a interpretabilidade dos modelos de rede neural, bem como reduzir tempo computacional inerente, se perspetiva como algo desafiante para a comunidade associada à IA. Neste sentido, será nosso objetivo investir não só na procura de outras transformações a introduzir nos modelos já usados, como procurar testar a aplicabilidade de novas arquiteturas de redes neuronais na modelação e previsão de séries temporais (com particular interesse pela área económico-financeira).
- Ainda em termos de perspetivas futuras, decorrente do trabalho desenvolvido em Ramos (2011) e do interesse pessoal, com ele suscitado, pela utilização dos pressupostos afetos ao paradigma da Estatística Bayesiana (onde se reconheceram vantagens na sua aplicação ao estudo da Cointegração em séries temporais), surge a ideia da aplicação destes pressupostos aos modelos de redes neuronais. Quando analisada a literatura, vemos que o tema foi objeto de vários trabalhos na primeira década de 2000 e, beneficiando do atual desenvolvimento computacional, parece ter recuperado nos últimos anos, como são exemplo os trabalhos de Kocadağli e Aşikgil (2014), Kobayashi e Shirayama (2017) ou Jang e Lee (2019).

Em suma, apesar de identificadas algumas limitações, reconhecemos que o trabalho feito não só se reverte de uma mais-valia pela compilação teórica de informação, como pelo contributo que a implementação computacional feita e o estudo empírico desenvolvido poderão ter para a comunidade científica que se dedica ao estudo afeto à modelação e previsão em séries temporais. Com isto, acreditamos que este trabalho seja um ponto de partida para desenvolvimentos futuros, onde a articulação entre o poder da inteligência artificial, numa perspetiva de *Machine Learning*, e o potencial da inteligência humana, no olhar sobre os dados, serão chave de sucesso. É nesta combinação que podemos afirmar que o limite do potencial da máquina é definido pelo pensamento do Homem.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] Akaike, H. (1974). A New Look at the Statistical Model Identification. In *Selected Papers of Hirotugu Akaike* (pp. 215–222). New York: Springer.

[2] Andrews, D. W. K. (1993). Tests for Parameter Instability and Structural Change With Unknown Change Point. *Econometrica*, *61*(4), 821.

[3] Andrews, D. W. K. (2003). Tests for Parameter Instability and Structural Change with Unknown Change Point: A Corrigendum. *Econometrica*, *71*(1), 395–397.

[4] Antoch, J., Hanousek, J., Horváth, L., Hušková, M., & Wang, S. (2019). Structural breaks in panel data: Large number of panels and short length time series. *Econometric Reviews*, *38*(7), 828–855.

[5] Aptech Systems Inc. (2019). GAUSS. Retrieved October 28, 2019, from https://www.aptech.com/

[6] Arlot, S., & Celisse, A. (2010). A survey of cross-validation procedures for model selection. *Statistics Surveys*, *4*, 40–79.

[7] Aue, A., & Horváth, L. (2013, January). Structural breaks in time series. *Journal of Time Series Analysis*.

[8] Avramov, D., & Chordia, T. (2006). Predicting stock returns. *Journal of Financial Economics*, *82*(2), 387–415.

[9] Bai, J., & Perron, P. (1998). Estimating and Testing Linear Models with Multiple Structural Changes. *Econometrica*, *66*(1), 47.

[10] Barto, A. G., Sutton, R. S., & Anderson, C. W. (1983). Neuronlike Adaptive Elements That Can Solve Difficult Learning Control Problems. *IEEE Transactions on Systems, Man and Cybernetics*, *SMC-13*(5), 834–846.

[11] Becker, S., & Plumbley, M. (1996). Unsupervised neural network learning procedures for feature extraction and classification. *Applied Intelligence*, *6*(3), 185–203.

[12] Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning Long-Term Dependencies with Gradient Descent is Difficult. *IEEE Transactions on Neural Networks*, *5*(2), 157–166.

[13] Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*. Oxford University Press.

[14] Box, G., & Jenkins, G. (1976). *Time Series Analysis, Forecasting and Control*. San Francisco: Holden-Day.

[15] Box, G., Jenkins, G. M., & Reinsel, G. (1994). *Time Series Analysis: Forecasting & Control* (3rd ed.). New Jersey: Prentice Hall.

[16] Brock, W. A., & de Lima, P. J. F. (1996). 11 Nonlinear time series, complexity theory, and finance. *Handbook of Statistics*.

[17] Broock, W. A., Scheinkman, J. A., Dechert, W. D., & LeBaron, B. (1996). A test for independence based on the correlation dimension. *Econometric Reviews*, *15*(3), 197–235.

[18] Brown, R. G. (1959). *Statistical Forecasting for Inventory Control*. New York: McGraw-Hill.

[19] Brown, R. L., Durbin, J., & Evans, J. M. (1975). Techniques for Testing the Constancy of Regression Relationships over Time. *Journal of the Royal Statistical Society. Series B (Methodological)*. WileyRoyal Statistical Society.

[20] Brownlee, J. (2017). *Introduction to Time Series Forecasting With Python*. Machine Learning Master Pty. Retrieved from https://machinelearningmastery.com/introduction-to-time-series-forecasting-with-python/

[21] Brownlee, J. (2018). A Gentle Introduction to Exponential Smoothing for Time Series Forecasting in Python. In *Time Series*. Retrieved from https://machinelearningmastery.com/exponential-smoothing-for-time-series-forecasting-in-python/

[22] Burnham, K. P., & Anderson, D. R. (2002). *Model selection and multimodel inference : a practical information-theoretic approach* (2nd ed.). Springer.

[23] Casini, A., & Perron, P. (2018). Structural Breaks in Time Series. Retrieved from http://arxiv.org/abs/1805.03807

[24] Cavalcante, R. C., Brasileiro, R. C., Souza, V. L. F., Nobrega, J. P., & Oliveira, A. L. I. (2016). Computational Intelligence and Financial Markets: A Survey and Future Directions. *Expert Systems with Applications*, *55*, 194–211.

[25] Chatfield, C. (2016). *The Analysis of Time Series: an introduction* (6th ed.). Chapman and Hall/CRC.

[26] Chatfield, C., Koehler, A. B., Ord, J. K., & Snyder, R. D. (2001). A New Look at Models for Exponential Smoothing. *Journal of the Royal Statistical Society. Series D (The Statistician)*. WileyRoyal Statistical Society.

[27] Chollet, F. (2017). *Deep Learning with Python* (1st ed.). New York: Manning Publications.

[28] Chow, G. C. (1960). Tests of Equality Between Sets of Coefficients in Two Linear Regressions. *Econometrica*, *28*(3), 591.

[29] Claeskens, G., & Hjort, N. L. (2008). *Model selection and model averaging*. *Model Selection and Model Averaging*. Cambridge University Press.

[30] Clement, E. P. (2014). Using Normalized Bayesian Information Criterion (Bic) to Improve Box - Jenkins Model Building. *American Journal of Mathematics and Statistics*, *4*(5), 214–221.

[31] Clements, M., Franses, P., & Swanson, N. (2004). Forecasting economic and financial time-series with non-linear models. *International Journal of Forecasting*, *20*(2), 169–183.

[32] Cleveland, R. B., Cleveland, W. S., McRae, J. E., & Terpenning, I. J. (1990). STL: A seasonal-trend decomposition procedure based on loess. *Journal of Official Statistics*, *6*(1), 3–73.

[33] Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, *20*(3), 273–297.

[34] Costa, A., Ramos, F., Mendes, D., & Mendes, V. (2019). Forecasting financial time series using deep learning techniques. In *IO 2019 - XX Congresso da APDIO 2019*. Instituto Politécnico de Tomar - Tomar.

[35] D'Agostino, R. B., Belanger, A., & D'Agostino Jr, R. B. (1990). A suggestion for using powerful and informative tests of normality. *American Statistician*, *44*(4), 316–321.

[36] Dagum, E. B., & Bianconcini, S. (2016). *Seasonal adjustment methods and real time trend-cycle estimation* (1st ed.). Springer.

[37] Data Science Academy. (2019). *Deep Learning Book*. Retrieved from http://deeplearningbook.com.br/

[38] Davidson, R., & MacKinnon, J. G. (1993). Estimation and Inference in Econometrics. *OUP Catalogue*.

[39] Duarte, M., & Watanabe, R. N. (2018). Notes on Scientific Computing for Biomechanics and Motor Control. GitHub. Retrieved from https://github.com/BMClab/BMC

[40] Engle, R. F. (1982). Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation. *Econometrica*, *50*(4), 987.

[41] FRED - Federal Reserve Bank of St. Louis. (n.d.). U.S. Bureau of Labor Statistics: Consumer Price Index for All Urban Consumers: All Items in U.S. City Average (CPIAUCSL). Retrieved October 31, 2019, from

https://fred.stlouisfed.org/series/CPIAUCSL

[42] Fuller, W. A. (1996). *Introduction to statistical time series*. John Wiley & Sons.

[43] Glorot, X., & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *JMLR W\&CP: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS 2010)* (pp. 249–256). Sardinia: JMLR Workshop and Conference Proceedings.

[44] Granger, C.W.J. ; Newbold, P. (1974). Spurious regressions in econometrics. *Journal of Econometrics*, *2*(2), 111–120.

[45] Granjon, P. (2014). *The CuSum algorithm - a small review*. Retrieved from https://hal.archives-ouvertes.fr/hal-00914697

[46] Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., & Schmidhuber, J. (2015). LSTM: A Search Space Odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, *28*(10), 2222–2232.

[47] Gustafsson, F. (2000). *Adaptive Filtering and Change Detection*. New York: Wiley.

[48] Hamilton, J. D. (1994). *Time series analysis*. *Econometric Theory*. New Jersy: Princeton University Press.

[49] Hang, N. T. (2019). Research on a number of applicable forecasting techniques in economic analysis, supporting enterprises to decide management. *World Scientific News*, *119*, 52–67.

[50] Hannan, E. J., & Quinn, B. G. (1979). The Determination of the Order of an Autoregression. *Journal of the Royal Statistical Society. Series B (Methodological)*. WileyRoyal Statistical Society.

[51] Hansen, B. (2001). The New Econometrics of Structural Change: Dating Breaks in U.S. Labour Productivity. *Journal of Economic Perspectives*, *15*(4), 117–128.

[52] Hassani, H., & Silva, E. S. (2015). Forecasting with Big Data: A Review. *Annals of Data Science*, *2*(1), 5–19.

[53] Hastie, Trevor, Tibshirani, Robert, Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. *Springer series in statistics*.

[54] Haykin, S. (2009). *Neural Networks and Learning Machines* (3rd ed.). Upper Saddle River, NJ: Pearson.

[55] He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In *2015 IEEE International Conference on Computer Vision (ICCV)* (pp. 1026–1034).

[56] Hebb, D. O. (1949). *The organization of behavior: a neuropsychological theory*. New

York: Wiley and Sons.

[57] Heij, C., Boer, P. de, Franses, P. H., Kloek, T., & Dijk, H. K. van. (2004). *Econometric Methods with Applications in Business and Economics*. Oxford University Press.

[58] Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735–1780.

[59] Hodrick, R. J., & Prescott, E. C. (1997). Postwar U.S. Business Cycles: An Empirical Investigation. *Journal of Money, Credit and Banking*, *29*(1), 1–16.

[60] Holt, C. (1957). *Forecasting seasonals and trends by exponentially weighted moving averages*. Pittsburgh Pa.: Carnegie Institute of Technology Graduate school of Industrial Administration.

[61] Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences of the United States of America*, *79*(8), 2554–2558.

[62] Hyndman, R. J., & Athanasopoulos, G. (2018). *Forecasting: Principles and Practice* (2nd ed.). Melbourne, Austrália: OTexts. Retrieved from https://otexts.com/fpp2/

[63] Hyndman, R. J., Koehler, A. B., Ord, J. K., & Snyder, R. D. (2008). *Forecasting with Exponential Smoothing: The State Space Approach*. *Springer Series in Statistics*. Springer Berlin Heidelberg.

[64] Jang, H., & Lee, J. (2019). Generative Bayesian neural network model for risk-neutral pricing of American index options. *Quantitative Finance*, *19*(4), 587–603.

[65] Jarque, C. M., & Bera, A. K. (1980). Efficient tests for normality, homoscedasticity and serial independence of regression residuals. *Economics Letters*, *6*(3), 255–259.

[66] Johnston, J., & DiNardo, J. (2001). *Métodos Econométricos* (4th ed.). (M. Hiil, F. Ferrão, & R. Menezes, Trads). McGraw-Hill.

[67] Jordan, J. (2018). Setting the learning rate of your neural network. Retrieved from https://www.jeremyjordan.me/nn-learning-rate/

[68] Jozefowicz, R., Zaremba, W., & Sutskever, I. (2015). An Empirical Exploration of Recurrent Network Architectures. In *ICML - International Conference on Machine Learning*.

[69] Kallas, M., Honeine, P., Francis, C., & Amoud, H. (2013). Kernel autoregressive models using Yule-Walker equations. *Signal Processing*, *93*(11), 3053–3061.

[70] Kearns, M. J. (1996). A Bound on the Error of Cross Validation Using the Approximation and Estimation Rates, with Consequences for the Training-Test Split. In *Advances in Neural Information Processing Systems* (Vol. 8, pp. 183–189).

Cambridge, MA: MIT Press.

[71] Kingma, D. P., & Ba, J. L. (2015). Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. International Conference on Learning Representations, ICLR.

[72] Kingsman, J. (2004). The investment funds and technical analysis. In *Sugar Trading Manual* (pp. 15a – 19). Elsevier.

[73] Kirchgässner, G., & Wolters, J. (2007). *Introduction to Modern Time Series Analysis* (Springer). Berlin.

[74] Kitagawa, G. (2010). *Introduction to time series modeling*. CRC Press.

[75] Kobayashi, S., & Shirayama, S. (2017). Time Series Forecasting with Multiple Deep Learners: Selection from a Bayesian Network. *Journal of Data Analysis and Information Processing*, *05*(03), 115–130.

[76] Kocadağli, O., & Aşikgil, B. (2014). Nonlinear time series forecasting with Bayesian neural networks. *Expert Systems with Applications*, *41*(15), 6596–6610.

[77] Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, *43*(1), 59–69.

[78] Koutník, J., Greff, K., Gomez, F., & Schmidhuber, J. (2014). A Clockwork RNN. *31st International Conference on Machine Learning, ICML 2014*, *5*, 3881–3889.

[79] Kwiatkowski, D., Phillips, P. C. B., Schmidt, P., & Shin, Y. (1992). Testing the null hypothesis of stationarity against the alternative of a unit root. How sure are we that economic time series have a unit root? *Journal of Econometrics*, *54*(1–3), 159–178.

[80] Lancaster, T. (2004). *An introduction to modern Bayesian econometrics*. Blackwell Pub.

[81] Lechner, M., Hasani, R., Amini, A., Henzinger, T. A., Rus, D., & Grosu, R. (2020). Neural circuit policies enabling auditable autonomy. *Nature Machine Intelligence*, *2*(10), 642–652.

[82] Libanio, G. A. (2005). Unit roots in macroeconomic time series: theory, implications, and evidence. *Nova Economia*, *15*(3), 145–176.

[83] Ljung, G. M., & Box, G. E. P. (1978). On a measure of lack of fit in time series models. *Biometrika*, *65*(2), 297–303.

[84] Lopes, D., & Ramos, F. (2019). Time Series Cross-Validation. Retrieved from https://github.com/DidierRLopes/TimeSeriesCrossValidation

[85] Lopes, D., & Ramos, F. (2020). Univariate Time Series Forecast. Retrieved from https://github.com/DidierRLopes/UnivariateTimeSeriesForecast

[86] Makridakis, S. G., Wheelwright, S. C., & Hyndman, R. J. (1998). *Forecasting: Methods*

and Applications (3rd ed.). New York, USA: Wiley.

[87] McClelland, J. L., Rumelhart, D. E., & Group, P. research. (1986). *Parallel distributed processing : Explorations in the microstructure of cognition (I and II)*. Cambridge, MA: MIT Press.

[88] McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, *5*(4), 115–133.

[89] McCulloch, W. S., & Pitts, W. (1947). How we know universals the perception of auditory and visual forms. *The Bulletin of Mathematical Biophysics*, *9*(3), 127–147.

[90] McMillan, D. G. (2005). Non-linear dynamics in international stock market returns. *Review of Financial Economics*, *14*(1), 81–91.

[91] McMillan, D. G. (2007). Non-linear forecasting of stock returns: Does volume help? *International Journal of Forecasting*, *23*(1), 115–126.

[92] Minsky, M., & Papert, S. (1969). *Perceptrons; an introduction to computational geometry*. MIT Press.

[93] Morshed, A. A., Andreou, E., & Boldea, O. (2018). Structural Break Tests Robust to Regression Misspecification. *Econometrics*, *6*(2), 1–39.

[94] Müller, A. C., & Guido, S. (2016). *Introduction to Machine Learning with Python: A Guide for Data Scientists*. O'Reilly Media, Inc.

[95] Murteira, B., Mullër, D., & Turkman, K. (1993). *Análise de Sucessões Cronológicas*. Mcgraw Hill.

[96] Ord, K. (2004). Charles Holt's report on exponentially weighted moving averages: an introduction and appreciation. *International Journal of Forecasting*, *20*(1), 1–3.

[97] Patterson, K. (2011). Why Distinguish Between Trend Stationary and Difference Stationary Processes? In *Unit Root Tests in Time Series* (pp. 38–67). Palgrave Macmillan, London.

[98] Patuwo, E., Zhang, P., & Hu, M. (1998). Forecasting With Artificial Neural Networks: The State of the Art. *International Journal of Forecasting*, *14*, 35–62.

[99] Paul, S. K. (2014). Determination of Exponential Smoothing Constant to Minimize Mean Square Error and Mean Absolute Deviation. *Global Journal of Research In Engineering*, *11*(3).

[100] Perron, P. (2006). Dealing with Structural Breaks. In *Palgrave handbook of econometrics* (Vol. 1, pp. 278--352).

[101] Pesaran, M. H., & Timmermann, A. (2004). How costly is it to ignore breaks when forecasting the direction of a time series? *International Journal of Forecasting*, *20*(3),

411–425.

[102] Pineda, F. (1987). Generalization of Back propagation to Recurrent and Higher Order Neural Networks. *Undefined*.

[103] Project Jupyter. (n.d.). Retrieved December 1, 2019, from https://jupyter.org/about

[104] Quandt, R. E. (1960). Tests of the Hypothesis That a Linear Regression System Obeys Two Separate Regimes. *Journal of the American Statistical Association*, *55*(290), 324–330.

[105] Ramos, F. (2011). *Cointegração, Modelos VAR e BVAR: Estudo comparativo entre a abordagem Clássica e Bayesiana no contexto dos Mercados Financeiros Europeus*. Faculdade de Ciências da Universidade de Lisboa / Instituto Superior de Ciências do Trabalho e da Empresa.

[106] Ramos, F., Costa, A., Mendes, D., & Mendes, V. (2018). Forecasting financial time series: a comparative study. In *JOCLAD 2018, XXIV Jornadas de Classificação e Análise de Dados*. Escola Naval – Alfeite.

[107] Ramsey, J. B., & Kmenta, J. an. (1980). Problems and Issues in Evaluating Econometric Models. In J. A. N. KMENTA & J. B. RAMSEY (Eds.), *Evaluation of Econometric Models* (pp. 1–11). Academic Press.

[108] Raschka, S., & Mirajalili, V. (2019). *Python machine learning: Machine learning and deep learning with Python, scikit-learn, and TensorFlow 2*. Packt Publishing Ltd.

[109] Ravichandiran, S. (2019). *Hands-On Deep Learning Algorithms with Python: Master deep learning algorithms with extensive math by implementing them using TensorFlow*. Packt Publishing Ltd.

[110] Ravn, M. O., & Uhlig, H. (2002). On adjusting the Hodrick-Prescott filter for the frequency of observations. *Review of Economics and Statistics*, *84*(2), 371–376.

[111] Rosenblatt, F. (1958). The Perceptron: A Probabilistic Model for Information Storage and Organization in The Brain. *Psychological Review*, 65--386.

[112] Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, *323*(6088), 533–536.

[113] Rumelhart, D. E., & Zipser, D. (1985). Feature Discovery by Competitive Learning. *Cognitive Science*, *9*(1), 75–112.

[114] Sage, A. P. (Ed.). (1990). *Concise encyclopedia of information processing in systems & organizations*. New York: Pergamon Press.

[115] Said, S. E., & Dickey, D. A. (1984). Testing for Unit Roots in Autoregressive-Moving Average Models of Unknown Order. *Biometrika*, *71*(3), 599.

[116] Schwarz, G. (1978). Estimating the Dimension of a Model. *The Annals of Statistics*, *6*(2), 461–464.

[117] Schwert, G. W. (1989). Why Does Stock Market Volatility Change Over Time? *The Journal of Finance*, *44*(5), 1115–1153.

[118] Scikit-learn. (n.d.). Scikit-learn: Machine Learning in Python. Retrieved February 17, 2020, from https://scikit-learn.org/stable/

[119] Sin, C. Y., & White, H. (1996). Information criteria for selecting possibly misspecified parametric models. *Journal of Econometrics*, *71*(1–2), 207–225.

[120] Thadewald, T., & Büning, H. (2007). Jarque-Bera test and its competitors for testing normality - A power comparison. *Journal of Applied Statistics*, *34*(1), 87–105.

[121] Tkáč, M., & Verner, R. (2016). Artificial neural networks in business: Two decades of research. *Applied Soft Computing*, *38*, 788–804.

[122] Valentinyi-endr, M. (2004). Structural breaks and financial risk management. *Magyar Nemzeti Bank Working Paper*.

[123] Wang, X., Smith, K., & Hyndman, R. (2006). Characteristic-based clustering for time series data. *Data Mining and Knowledge Discovery*, *13*(3), 335–364.

[124] Widrow, B. (1962). Generalization and information storage in network of adaline "neurons." In M. D. Yovits, G. T. Jacobi, & G. D. Goldstein (Eds.), *Self-Organizing Systems* (pp. 435–461). Washington D.C.: Spartan Books.

[125] Wiener, N. (1948). *Cybernetics or Control and Communication in the Animal and the Machine* (1st ed.). New York: John Wiley & Sons.

[126] Willmott, C., & Matsuura, K. (2005). Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. *Climate Research*, *30*(1), 79–82.

[127] Wilson, J. H., & Spralls III, S. A. (2018). What do business professionals say about forecasting in the marketing curriculum? *International Journal of Business, Marketing, & Decision Science*, *11*(1), 1–20.

[128] Winters, P. R. (1960). Forecasting Sales by Exponentially Weighted Moving Averages. *Management Science*, *6*(3), 324–342.

[129] Zhang, Y., Guo, Q., & Wang, J. (2017). Big data analysis using neural networks, *49*, 9–18.

ANEXO A

A.1. POSTER APRESENTADO NA JOCLAD 2018

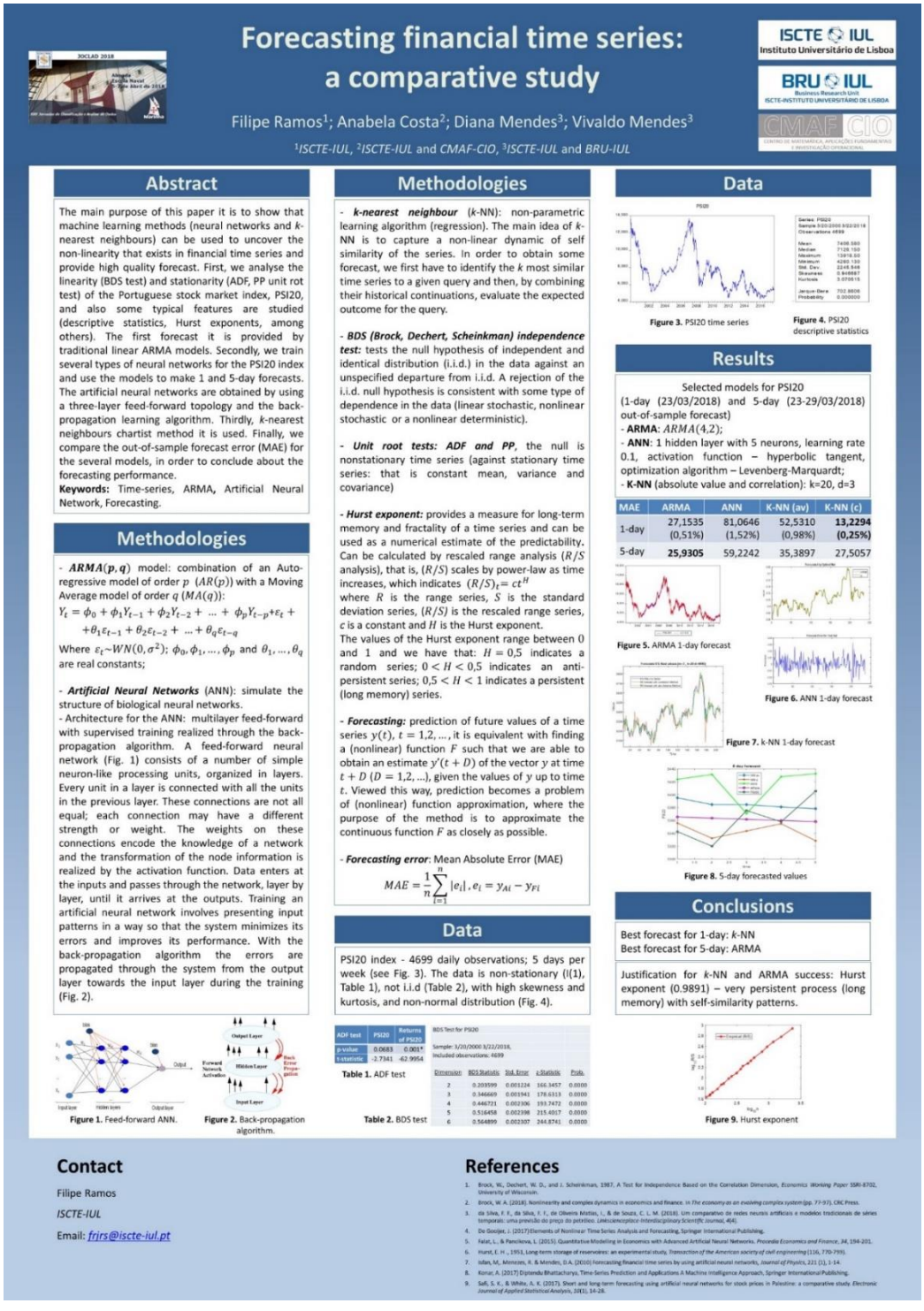


FIGURA A.1 – POSTER APRESENTADO NA JOCLAD 2018

A.2. RESUMO DO TRABALHO APRESENTADO NO XIX CONGRESSO DA ASSOCIAÇÃO PORTUGUESA DE INVESTIGAÇÃO OPERACIONAL

The main purpose of this paper it is to show that machine learning methods (neural networks and *k*-nearest neighbours) can be used to uncover the non-linearity that exists in financial time series and provide high quality forecast. First, we analyse the linearity (BDS test) and stationarity (ADF, PP unit rot test) of the Portuguese stock market index, PSI 20, and also some typical features are studied (descriptive statistics, Hurst exponents, among others). The first forecast it is provided by traditional linear ARMA models. Secondly, we train several types of neural networks for the PSI 20 index and use the models to make 1 and 5-day forecasts. The artificial neural networks are obtained by using a three-layer feed-forward topology and the back-propagation learning algorithm. Thirdly, *k*-nearest neighbours chartist method it is used. Finally, we compare the out-of-sample forecast error (MAE) for the several models, in order to conclude about the forecasting performance.

Keywords: Time-series, ARMA**,** Artificial Neural Network, Forecasting.

A.3. RESUMO DO TRABALHO APRESENTADO NO XX CONGRESSO DA ASSOCIAÇÃO PORTUGUESA DE INVESTIGAÇÃO OPERACIONAL

In this article, we propose the use of recurrent neural networks based on long short-term memory (LSTM) architecture in order to forecast financial time series. Artificial neural networks have proven to be efficient in forecasting financial time series. In particular, recurrent neural networks have been able to store past inputs to produce the currently desired output, which justifies their application in financial time series prediction. First, we study the main features of the Standard and Poor's 500 index, S&P500, such as the linearity, stationarity, descriptive statistics, Hurst exponents, among others. Secondly, we train several types of recurrent neural networks for the S&P500 index and use the models to make short-term forecasts. Finally, we compare the out-of-sample forecast error (MAE) for the employed models, in order to conclude about the forecasting performance.

Keywords: Time-series, Recurrent Neural Network, LSTM, Forecasting.

ANEXO B

B.1. ESTUDO DE SÉRIES TEMPORAIS: INFORMAÇÕES COMPLEMENTARES

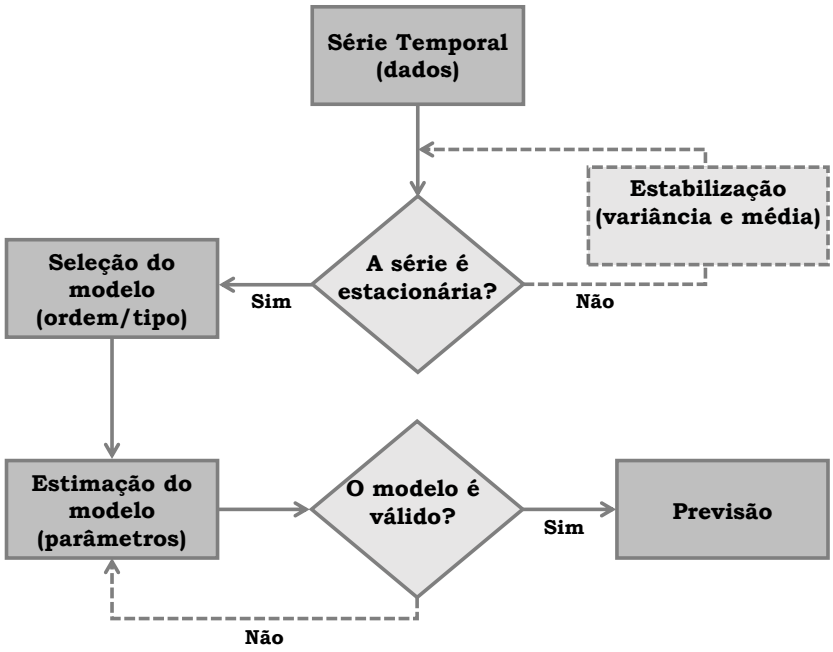


FIGURA B.1 – Ciclo iterativo de construção de modelos: Metodologia de *Box & Jenkins*

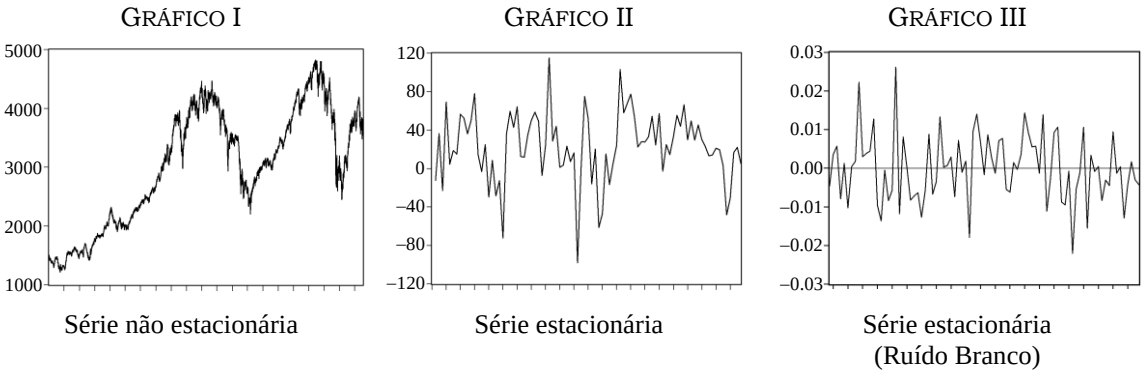


FIGURA B.2 – Exemplos de séries estacionárias e não estacionárias

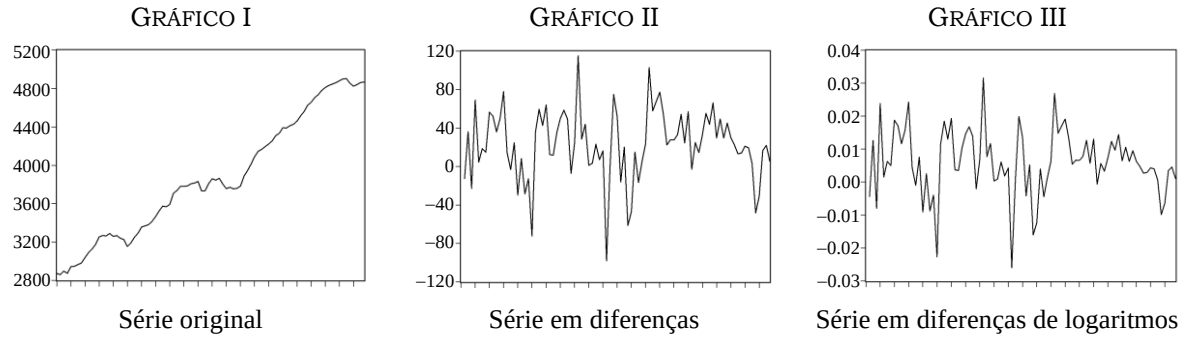


FIGURA B.3 – Processos de transformação de uma série com vista à estacionariedade

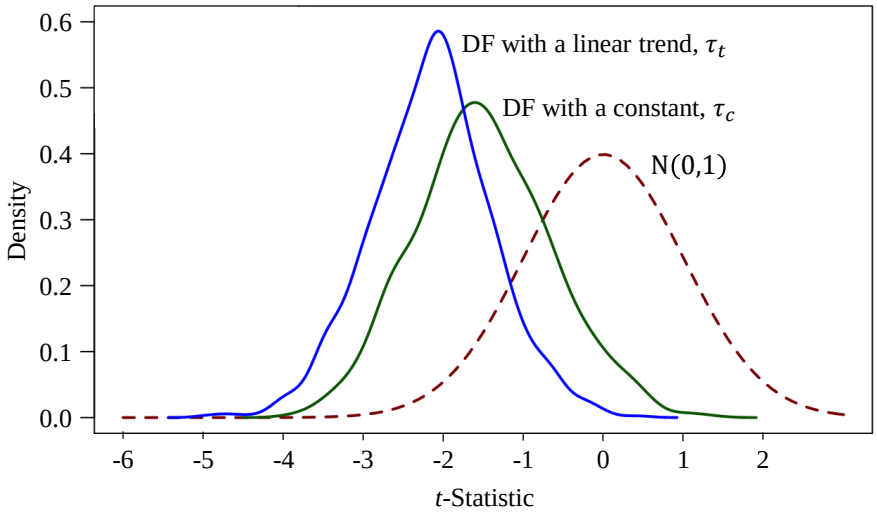


FIGURA B.4 – Distribuições *Dickey-Fuller* (τ_t e τ_c) e comparação com a $N(0,1)$

B.2. ANÁLISE DE UM CORRELOGRAMA: INFORMAÇÕES COMPLEMENTARES

Definidas as funções de autocorrelação (FAC) e autocorrelação parcial (FACP) para uma série estacionária (Sec. 2.2.4.), ao longo da Sec. 2.4 referimo-nos. a como, a partir do seu comportamento, poderemos depreender não só alguns aspetos sobre a natureza dos dados, como perceber se estamos perante uma série modelada por um modelo AR, um modelo MA, ou uma combinação de ambos – modelo ARMA. Mais, é ainda possível, a partir da análise do correlograma de uma série estacionária, conjeturar sobre qual a ordem, p e q, do modelo $ARMA(p,q)$. Contudo, muito embora existam alguns 'padrões', conforme síntese apresentada na Tabela 2.3., no seguimento do exposto ao longo do Capítulo 2, acrescenta-se alguma informação nomeadamente quanto à interpretação das autocorrelações e autocorrelações parciais.

Em termos gráficos, quando representados os coeficientes de autocorrelação,

$$r_k = \frac{\sum_{t=k+1}^{n}(x_t - \bar{x})(x_{t-k} - \bar{x})}{\sum_{t=1}^{n}(x_t - \bar{x})^2}$$

onde $\bar{x} = \sum_{t=1}^{n} x_t/n$, em função das amplitudes do seu respetivo desfasamento temporal (*lag*) k, obtemos o correlograma correspondente à FAC da série temporal. Por seu lado, caso o gráfico expresse os coeficientes de correlação parcial (correlação entre uma variável e um desfasamento temporal dela mesma que não é explicada por correlações com desfasamentos de ordem inferior) entre as observações da série em função das amplitudes do respetivo desfasamento temporal, obtemos o correlograma para FACP. Assim, considerada uma série temporal, $\{y_t\}_{t\in T}$, a autocorrelação no $lag\ 1$ corresponde à correlação entre y_t e y_{t-1}, que presumivelmente será a correlação entre y_{t-1} e y_{t-2} e assim sucessivamente.

Em termos genéricos, sobre a natureza dos dados, temos:

em processos semelhantes a um ruído branco (processo puramente aleatório), os valores sucessivos não guardam correlação entre si, sendo de esperar valores nulos (ou próximos disso) dos coeficientes de autocorrelação. Deste modo, com elevada probabilidade espera-se que que todas as autocorrelações sejam nulas, pelo que é esperado que, no correlograma, as barras de autocorrelação estejam próximas de zero, não havendo nenhuma que saia fora do intervalo de confiança.

em processos não estacionários, como séries temporais com tendência, os valores de r_k não decaem para zero (exceto, eventualmente, para desfasamentos grandes). Tal deve-se ao facto de poder existir um conjunto considerável de valores consecutivos (agrupados) acima ou abaixo da média geral. Nestes casos, pouca ou nenhuma informação pode ser extraída porque a que a tendência dominará as outras características (daí a análise das autocorrelações só ter algum significado em séries estacionárias).

Adicionalmente e em virtude do exposto, consideremos na análise subsequente estar perante uma série temporal, $\{y_t\}_{t\in T}$, estacionária.

Supondo que, num momento t, uma dada observação y_t está correlacionada com y_{t-1} e, igualmente, y_{t-1} está correlacionada com y_{t-2}, é expectável encontrar correlação também entre y_t e y_{t-2}, pelo que a correlação esperada no $lag\ 2$ será o quadrado da correlação observada no $lag\ 1$, propagando-se a correlação no $lag\ 1$ para o $lag\ 2$ e assim sucessivamente para $lags$ de ordem superior. Nestes casos, obtemos um correlograma onde, para a FAC, as autocorrelações são significativas para vários $lags$ (dada a propagação da autocorrelação), mas onde a FACP tem um pico significativo apenas no $lag\ 1$, sendo (praticamente) nula nos $lags$ seguintes. Em termos de modelos autorregressivos, a situação descrita indica que um modelo $AR(1)$ deve ser considerado.

Este raciocínio poderá ser generalizado para um $lag\ k$, onde a autocorrelação parcial no $lag\ k$ indica o coeficiente $AR(k)$. Pelo que, por meio da análise da FACP podemos determinar quantos termos autoagressivos devem ser considerados para explicar o padrão de autocorrelação numa série temporal estacionária: se a autocorrelação parcial for significativa no $lag\ k$ e cair abruptamente para zero em $lags$ de ordem superior, isso sugere considerar-se um modelo $AR(k)$.

Porém, considerada uma série estacionária, muito embora um padrão de correlação possa (em princípio) ser removido adicionando termos autorregressivos à equação do modelo, esta nem sempre é a melhor solução para explicar o padrão de autocorrelação. Por vezes, é mais eficiente adicionar termos MA (desfasamentos nos erros de previsão), ou seja, o padrão de autocorrelação poder ser explicado mais facilmente adicionando termos MA em vez de se adicionarem termos AR à série temporal. Nesses cacos, a FAC acaba por desempenhar um papel para os termos MA idêntico ao que a FACP desempenha para os termos AR.

Com efeito, seguindo a mesma linha de raciocínio acima, sendo a autocorrelação significativa num dado $lag\ k$, decaindo abruptamente para zero em $lags$ de ordem superior, tal indica-nos que devem ser usados (exatamente) k termos MA na equação de previsão, ajustando-se assim um modelo $MA(k)$.

Acontece que, na prática, o(s) melhor(es) modelos(s) resultam, muitas vezes, de uma combinação das componentes AR e MA, obtendo-se modelos mistos – modelos ARMA. Na estimação da respetiva ordem destes modelos mistos, importa referir que não deve ser seguida uma abordagem 'regressiva', consistindo em ir retirando termos AR ou MA ao modelo, mas sim uma abordagem "progressiva", isto é, ir adicionando termos conforme indicado pela aparência do correlograma respeitante à FAC e/ou à FACP.

Ainda, na escolha de modelos mistos, devem ser considerados alguns cuidados, uma vez que é possível que um termo AR cancele o efeito de e um termo MA e vice versa. Procurando clarificar tal situação, suponhamos que, para uma série temporal estacionária, apesar da 'melhor' opção ser um modelo $ARMA(p, q)$, para certos $p, q \in \mathbb{N}_0$, foi considerado um modelo $ARMA(p+1, q+1)$. Neste caso, os termos adicionados (um na componente AR e outro na componente MA) apesar de parecerem significativos no modelo, internamente podem estar a funcionar um contra o outro (o processo de estimativa de parâmetros pode obrigar a um número considerável de iterações até convergir). Assim, quando considerado um modelo com termos AR e MA simultaneamente, sugere-se considerar a possibilidade de um modelo com um termo AR e um termo MA a menos (particularmente se as estimativas de parâmetro no modelo original requerem várias iterações até convergir).

B.3. TESTES ESTATÍSTICOS USADOS NO ESTUDO DE SÉRIES TEMPORAIS: INFORMAÇÕES COMPLEMENTARES

- **Teste de Normalidade de *Jarque-Bera***

Existem diversos testes para testar a normalidade dos dados de uma série temporal ou da distribuição dos erros de um modelo de previsão dessa série, que avaliam o ajustamento dos dados a uma distribuição normal unidimensional. Um dos mais utilizados em econometria, com reconhecidas vantagens sobre outros, é o teste de *Jarque-Bera* (Thadewald & Büning, 2007). Trata-se de um teste assintótico em que é testado H_0: "os dados ajustam-se a uma distribuição normal" contra H_1: "os dados ajustam-se a uma distribuição não-normal da família de Pearson".

O procedimento do teste consiste em calcular os valores de assimetria e curtose amostrais e utilizar a seguinte estatística de teste (Jarque & Bera, 1980)

$$JB = \frac{n}{6}\left(S_k + \frac{(K_u - 3)^2}{4}\right)$$

onde:

- S_k é o coeficiente amostral de simetria;
- K_u é o coeficiente amostral de curtose;
- n é a dimensão da amostra.

Sob a hipótese nula de normalidade, se a dimensão da amostra for suficientemente grande (>2000), a estatística JB segue assintoticamente uma distribuição qui-quadrado com 2 graus de liberdade. Assim, rejeita-se a normalidade com nível de significância α se $JB > \chi^2_{1-\alpha,2}$.

- **Teste de *Skewness* e Teste de *Kurtosis* (Normalidade)**

Os testes de *Skewness* e de *Kurtosis* testam se a inclinação da curva relativa à distribuição dos dados é idêntica à distribuição normal, testando se os valores para a *Skewness* e para a *Kurtosis* são os "normais". Assim, no caso do Teste de *Skewness* é testado H_0: "a assimetria da população, de onde a amostra foi retirada, é a mesma de uma distribuição normal" contra H_1: "a assimetria da população, de onde a amostra foi retirada, não é a mesma de uma distribuição normal". Já no caso do Teste da *Kurtosis* é testado H_0: "o achatamento da população, de onde a amostra foi retirada, é o mesmo de uma distribuição normal" contra H_1: "o achatamento da população, de onde a amostra foi retirada, não é o mesmo de uma distribuição normal". Em cada um dos casos, o teste recebe como *input* os dados da amostra e devolve o valor *z-score* calculado para o teste e a respetiva probabilidade de significância (*p-value*). Para mais detalhes sobre os referidos testes, veja-se D'Agostino *et al.* (1990).

- **Independência (Teste de BDS)**

Para averiguar a independência temporal de uma série, é frequente utilizar do teste de Brock, Dechert e Scheinkmen – Teste BDS (Broock, Scheinkman, Dechert, & LeBaron, 1996). Este é um teste não paramétrico que testa a dependência em série e a estrutura não linear de uma série temporal, e tem como base o integral de correlação, testando a hipótese nula da série ser iid. O teste baseia-se no conceito de correlação espacial da teoria do caos e a estatística teste é descrita por Brock & de Lima (1996)

$$W_m(e) = \sqrt{n}\frac{C_m(e) - \left(C_1(e)\right)^m}{\hat{\sigma}_m(e)}$$

onde:

- $\hat{\sigma}_m(e)$ é uma estimativa do desvio-padrão sob a hipótese nula;
- $C_m(e)$ é o integral de correlação, que mede o número de pares possíveis de pontos sequenciais mais próximos do que uma distância e num espaço de dimensão m;
- n é a dimensão da amostra.
- $W_m^n(\varepsilon)$ tende para uma distribuição normal padrão, $N(0,1)$, quando n tende para infinito.

Se os valores da estatística teste forem superiores ao valor crítico de rejeição a série apresenta dependência linear nos seus termos.

- **Autocorrelação (teste de Ljung-Box)**

O teste de Ljung-Box permite averiguar se alguma das m autocorrelações dos resíduos estimados de uma série temporal é diferente de zero, testando H_0: "os resíduos são iid" (i.e. as correlações são nulas na população de onde foi extraída a amostra, portanto qualquer correlação observada resulta da aleatoriedade do processo de amostragem) contra H_1: "os resíduos não são iid, exibindo autocorrelação". Isto equivale a testar a nulidade de uma sequência de m valores iniciais da função de autocorrelação (FAC).

Ou seja, para testar $H_0: \rho_1 = \rho_2 = \cdots = \rho_m = 0$ contra $H_1: \exists\, \rho_k \neq 0,\ k = 1,2,\dots,m$ utiliza-se a estatística de teste (Ljung & Box, 1978)

$$LB = n(n+2)\sum_{k=1}^{m}\frac{\hat{\rho}_k{}^2}{n-k}$$

onde:

- $\hat{\rho}_k = \frac{Cov(y_t, y_{t+k})}{\sqrt{Var(y_t)\ Var(y_{t+k})}}$ é o coeficiente de correlação entre valores da série desfasados por k períodos (é possível demonstrar que $\hat{\rho}_k \sim N(0,1/T)$);
- m é a duração do desfasamento;
- n é a dimensão da amostra.

Para dimensões elevadas, a distribuição assintótica da estatística LB é qui-quadrado com m graus de liberdade. Quanto menor for o valor observado da estatística de teste melhor é o ajustamento dos dados ao modelo selecionado e, portanto, rejeitamos a hipótese de ausência de autocorrelação apenas se $LB > \chi^2_{1-\alpha,m}$, com um nível de significância α.

- **Heteroscedasticidade (teste ARCH-LM)**

Para analisar a presença de heteroscedasticidade condicional autoregressiva (*Autoregressive Conditional Heteroscedasticity*, ARCH) nas séries financeiras, é habitual recorrer-se ao multiplicador de Lagrange (*Lagrange Multiplier*, LM) desenvolvido por Engle (1982), denominado teste ARCH-LM.

O teste baseia-se em que se os resíduos do modelo forem heteroscedásticos, os seus quadrados apresentam autocorrelação. Assim, se o modelo de regressão linear dos quadrados dos resíduos não for significativo, os resíduos serão homoscedásticos.

$$u_t^2 = \sum_{i=1}^{r} \gamma_i\, u_{t-1}^2 + \varepsilon_t, \quad \varepsilon_t \sim iid\ (0, \sigma_\varepsilon)$$

A hipótese nula $H_0: \gamma_1 = \gamma_2 = \cdots = \gamma_r = 0$ pode ser testada contra a hipótese alternativa $H_1: \exists\ \gamma_i \neq 0,\ i = 1,2,\ldots,r$, através da estatística de teste nR^2 (produto da dimensão da amostra pelo coeficiente de determinação do modelo de regressão linear) que apresenta uma distribuição qui-quadrado com r graus de liberdade. Se o seu valor for superior ao valor crítico (para um dado nível de significância, α), $nR^2 > \chi^2_{1-\alpha,r}$, então a hipótese ausência de autocorrelação deve ser rejeitada e os resíduos apresentam heteroscedasticidade.

ANEXO C

C.1. TIPOLOGIAS/ARQUITETURAS DE REDES NEURONAIS

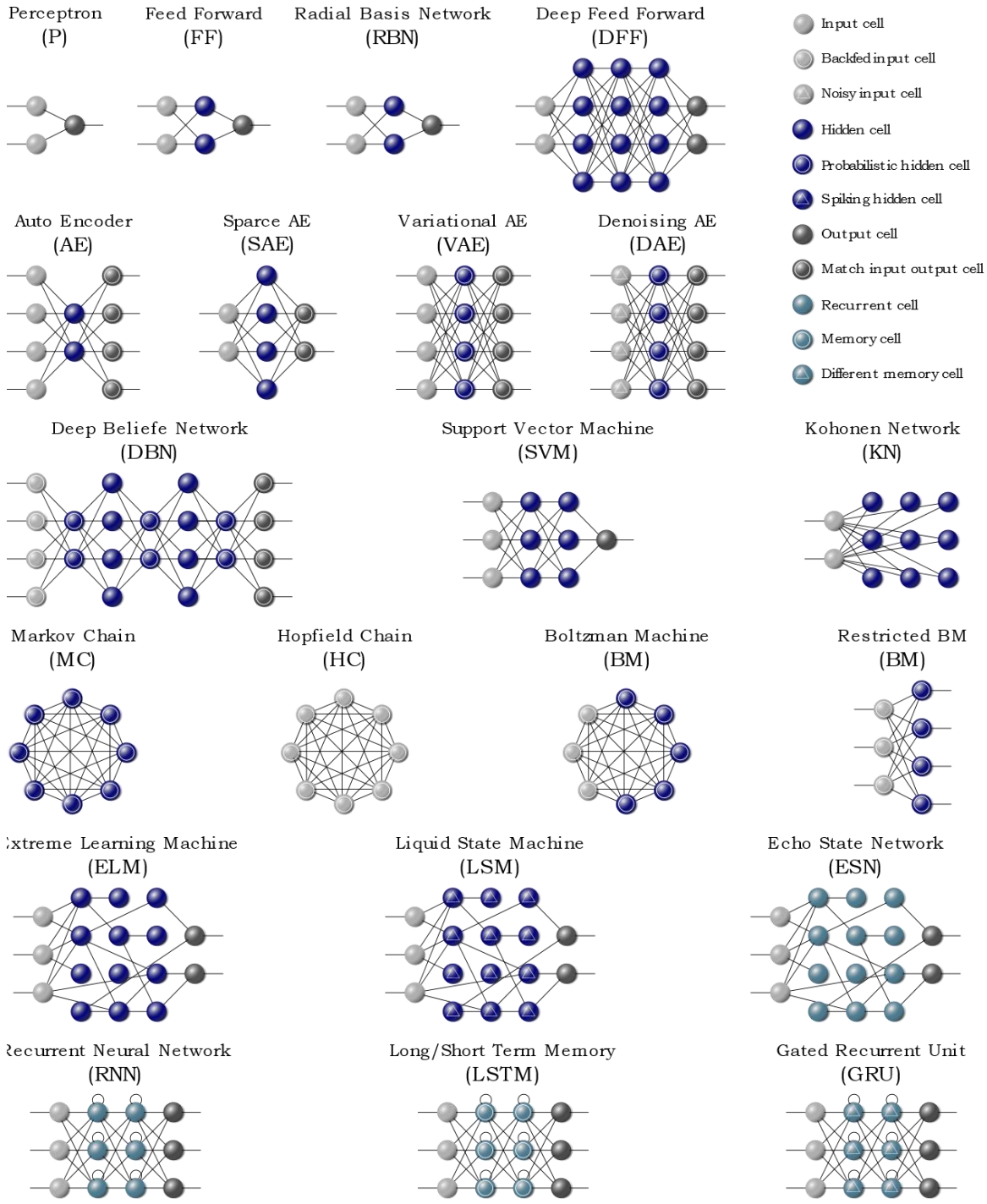


FIGURA C.1 – Grafos ilustrativos de exemplos de redes neuronais

ANEXO D

D.1. SÉRIE CPIAUCSL: INFORMAÇÕES[122]

Unidades: Índice "$1982 - 1984 = 100$" (ajustado sazonalmente)

Frequência: Mensal

O CPIAUCSL é uma medida da variação mensal média no preço de bens e serviços pagos pelos consumidores urbanos entre dois períodos. Também pode representar os hábitos de compra dos consumidores urbanos. Este índice inclui cerca de 88% da população total (incluindo elementos da população ativa, desempregados, aposentados e outros).

O CPI (*Consumer Price Index*) baseia-se nos preços de alimentos, roupas, alojamento e combustíveis, transporte, taxas de serviço (por exemplo, serviço de água e esgotos) e impostos sobre vendas. Os preços são recolhidos mensalmente em cerca de 4.000 agregados familiares e aproximadamente 26.000 estabelecimentos de venda em 87 áreas urbanas. Para calcular o índice, as variações de preço são calculadas com pesos que representam sua importância nos gastos de um determinado grupo. O índice mede as variações de preço (como uma variação percentual) a partir de uma data de referência predeterminada. Além do índice não ajustado original distribuído, o *Bureau of Labor Statistics* também divulga um índice com ajuste sazonal. A série não ajustada reflete todos os fatores que podem influenciar uma mudança nos preços. No entanto, pode ser muito útil observar o CPI com ajuste sazonal, que remove os efeitos de mudanças sazonais, como o clima.

O CPI pode ser usado para reconhecer períodos de inflação e deflação. Aumentos significativos no CPI em pouco tempo podem indicar um período de inflação e reduções significativas no CPI em pouco tempo podem indicar um período de deflação. No entanto, como o CPI inclui preços voláteis, como alimentos e petróleo, pode não ser uma medida confiável dos períodos inflacionários e deflacionários.

Finalmente, observe-se que o CPI é uma medida estatística vulnerável ao erro amostral, pois baseia-se numa amostra de preços e não na média completa.

[122] A informação constante neste Anexo foi obtida de FRED - Federal Reserve Bank of St. Louis (n.d.)

D.2. FUNÇÃO CÍCLICA PARA A SELEÇÃO DE MODELOS ARMA

```
def ARIMA_grid(dataset, arimaCfg, isLog):
    df_IC = pd.DataFrame( columns=['ARIMA(p, d, q)', 'AIC', 'BIC', 'HQIC'])
    df_IC = df_IC.set_index('ARIMA(p, d, q)')
    dict_arimaFit  = {}
    dict_arimaPred = {}
    modelNum = 0;
    totalModelNum = len(arimaCfg.p_range)*len(arimaCfg.d_range)*len(arimaCfg.q_range)
    for p in arimaCfg.p_range:
        for d in arimaCfg.d_range:
            for q in arimaCfg.q_range:
                order = (p,d,q)
                modelNum+=1;
                print("%d/%d " % (modelNum, totalModelNum))
                try:
                    model = ARIMA(dataset, order=order);
                    model_fit = model.fit(disp=0);
                    if (~np.isnan(model_fit.mle_retvals.get('fopt'))):

                        model_str = 'ARIMA' + str(order)
                        df_IC = df_IC.append(pd.DataFrame({'AIC':model_fit.aic, 'BIC':mo
del_fit.bic, \
                                                           'HQIC':model_fit.hqic}, index
=[model_str]), ignore_index=False)
                        if (isLog):
                            dict_arimaFit[model_str]  = np.e**(model_fit.predict(typ='le
vels'));
                            dict_arimaPred[model_str] = np.e**(model_fit.forecast(steps=
stepsToForecast[-1])[0]);
                        else:
                            dict_arimaFit[model_str]  = model_fit.predict(typ='levels');
                            dict_arimaPred[model_str] = model_fit.forecast(steps=stepsTo
Forecast[-1])[0];
                except:
                    continue
    return dict_arimaFit, dict_arimaPred, df_IC
```

D.3. ESTIMAÇÃO E SELEÇÃO DE MODELOS ETS: UM EXEMPLO (A_d,M)

```
model_TS_AdM = ExponentialSmoothing(df_train.values,
                                    trend='add',
                                    damped=True,
                                    seasonal='mul',
                                    seasonal_periods=seasonal_periods)
TS_AdM = model_TS_AdM.fit(smoothing_level=None,
                          smoothing_slope=None,
                          damping_slope=None,
                          smoothing_seasonal=None)
if (TS AdM.mle retvals.success):
    TS_AdM_forecast = TS_AdM.forecast(stepsToForecast[-1])

    if (~np.isnan(TS_AdM_forecast).any()):
        TS_print.append(True);
        title = 'TS (Ad, M) Holt-Winters damped method with ' \
            + r'$\alpha$' + ' = ' + str(round(TS AdM.params['smoothing level'],2)) \
            + r', $\beta$' + ' = ' + str(round(TS_AdM.params['smoothing_slope'],2)) \
            + r', $\zeta$' + ' = ' + str(round(TS_AdM.params['damping_slope'],2)) \
            + r', $\gamma$' + ' = ' + str(round(TS_AdM.params['smoothing_seasonal'],2))\
            + ', m = ' + str(model_TS_AdM.seasonal_periods);

        ETS_fit_plot(train = df_train.values,
                     fit = TS_AdM.fittedvalues,
                     title = title)
    else:
        TS_print.append(False);
        print ('RuntimeWarning: invalid value encountered in double_scalars.')
else:
    TS_print.append(False);
    print ('ConvergenceWarning: Optimization failed to converge.')
```

D.4. FUNÇÃO *CROSS-VALIDATION*: FORWARD CHAINING

```
def split_train_val_test_forwardChaining(sequence, numInputs, numOutputs, numJumps):
    """ Returns sets to train, cross-validate and test a model using forward chaining
    Parameters:
        sequence (array)  : Full training dataset
        numInputs (int)   : Number of inputs X and Xcv used at each training
        numOutputs (int)  : Number of outputs y and ycv used at each training
        numJumps (int)    : Number of sequence samples to be ignored between (X,y) sets
    Returns:
        X (2D array)      : Array of numInputs arrays used for training
        y (2D array)      : Array of numOutputs arrays used for training
        Xcv (2D array)    : Array of numInputs arrays used for cross-validation
        ycv (2D array)    : Array of numOutputs arrays used for cross-validation
        Xtest (2D array)  : Array of numInputs arrays used for testing
        ytest (2D array)  : Array of numOutputs arrays used for testing
    """
    X, y, Xcv, ycv, Xtest, ytest = dict(), dict(), dict(), dict(), dict(), dict()
    j=2; # Tracks index of CV set at each train/val/test split
    # Iterate through all train/val/test splits
    while 1:
        start_ix=0; end_ix=0; startCv_ix=0; endCv_ix=0; startTest_ix=0; endTest_ix=0;
        X_it, y_it, Xcv_it, ycv_it, Xtest_it, ytest_it = list(), list(), list(), list(),
list(), list()
        i=0; # Index of individual training set at each train/val/test split

        # Iterate until index of individual training set is smaller than index of cv set
        while (i < j):
            ## TRAINING DATA
            start_ix = numJumps*i;
            end_ix = start_ix + numInputs;

            seq_x = sequence[start_ix:end_ix]
            X_it.append(seq_x)
            seq_y = sequence[end_ix:end_ix+numOutputs]
            y_it.append(seq_y)

            i+=1;

        # Once test data crosses time series length return
        if ((((end_ix+numInputs)+numInputs)+numOutputs) > (len(sequence))):
            break

        ## CROSS-VALIDATION DATA
        startCv_ix = end_ix;
        endCv_ix = end_ix + numInputs;

        seq_xcv = sequence[startCv_ix:endCv_ix]
        Xcv_it.append(seq_xcv)
        seq_ycv = sequence[endCv_ix:endCv_ix+numOutputs]
        ycv_it.append(seq_ycv)
        ## TEST DATA
        startTest_ix = endCv_ix;
        endTest_ix = endCv_ix + numInputs;

        seq_xtest = sequence[startTest_ix:endTest_ix]
        Xtest_it.append(seq_xtest)
        seq_ytest = sequence[endTest_ix:endTest_ix+numOutputs]
        ytest_it.append(seq_ytest)
        ## Add another train/val/test split
        X[j-2] = np.array(X_it)
        y[j-2] = np.array(y_it)
        Xcv[j-2] = np.array(Xcv_it)
        ycv[j-2] = np.array(ycv_it)
        Xtest[j-2] = np.array(Xtest_it)
        ytest[j-2] = np.array(ytest_it)

        j+=1;

    if (len(X)==0 or len(Xcv)==0 or len(Xtest)==0):
        print("The sequence provided does not has size enough to populate the return arr
ays")

    return X, y, Xcv, ycv, Xtest, ytest
```

D.5. FUNÇÃO *CROSS-VALIDATION*: GROUP K-FOLD

```
def split_train_val_groupKFold(sequence, numInputs, numOutputs, numJumps):
    """ Returns sets to train and cross-validate a model using group K-Fold

    Parameters:
        sequence (array)  : Full training dataset
        numInputs (int)   : Number of inputs X and Xcv used at each training
        numOutputs (int)  : Number of outputs y and ycv used at each training
        numJumps (int)    : Number of sequence samples to be ignored between (X,y) sets
    Returns:
        X (2D array)    : Array of numInputs arrays used for training
        y (2D array)    : Array of numOutputs arrays used for training
        Xcv (2D array)  : Array of numInputs arrays used for cross-validation
        ycv (2D array)  : Array of numOutputs arrays used for cross-validation

    """

    X, y, Xcv, ycv = dict(), dict(), dict(), dict()

    # Iterate through 5 train/val splits
    for j in np.arange(5):
        start_ix=0; end_ix=0; startCv_ix=0; endCv_ix=0;
        X_it, y_it, Xcv_it, ycv_it = list(), list(), list(), list()
        i=0; # Index of individual training set at each train/val split
        n=0; # Number of numJumps

        while 1:
            if ((i+1+j)%(5) != 0):
                # TRAINING DATA
                start_ix = endCv_ix + numJumps*n;
                end_ix = start_ix + numInputs;
                n+=1;

               #Leave train/val split loop once training data crosses time series length
                if end_ix+numOutputs > len(sequence)-1:
                    break

                seq_x = sequence[start_ix:end_ix]
                X_it.append(seq_x)
                seq_y = sequence[end_ix:end_ix+numOutputs]
                y_it.append(seq_y)
            else:
                # CROSS-VALIDATION DATA
                startCv_ix = end_ix;
                endCv ix = end ix + numInputs;
                n=0;

                # Once val data crosses time series length return
                if ((endCv_ix+numOutputs) > len(sequence)):
                    break

                seq_xcv = sequence[startCv_ix:endCv_ix]
                Xcv_it.append(seq_xcv)
                seq_ycv = sequence[endCv_ix:endCv_ix+numOutputs]
                ycv_it.append(seq_ycv)

            i+=1;

        ## Add another train/val split
        X[j] = np.array(X_it)
        y[j] = np.array(y_it)
        Xcv[j] = np.array(Xcv_it)
        ycv[j] = np.array(ycv_it)

    if (len(X)==0 or len(Xcv)==0):
        print("The sequence provided does not has size enough to populate the return arr
ays")

    return X, y, Xcv, ycv
```

ANEXO E

E.1. SÉRIE CPIAUCSL: INFORMAÇÕES COMPLEMENTARES

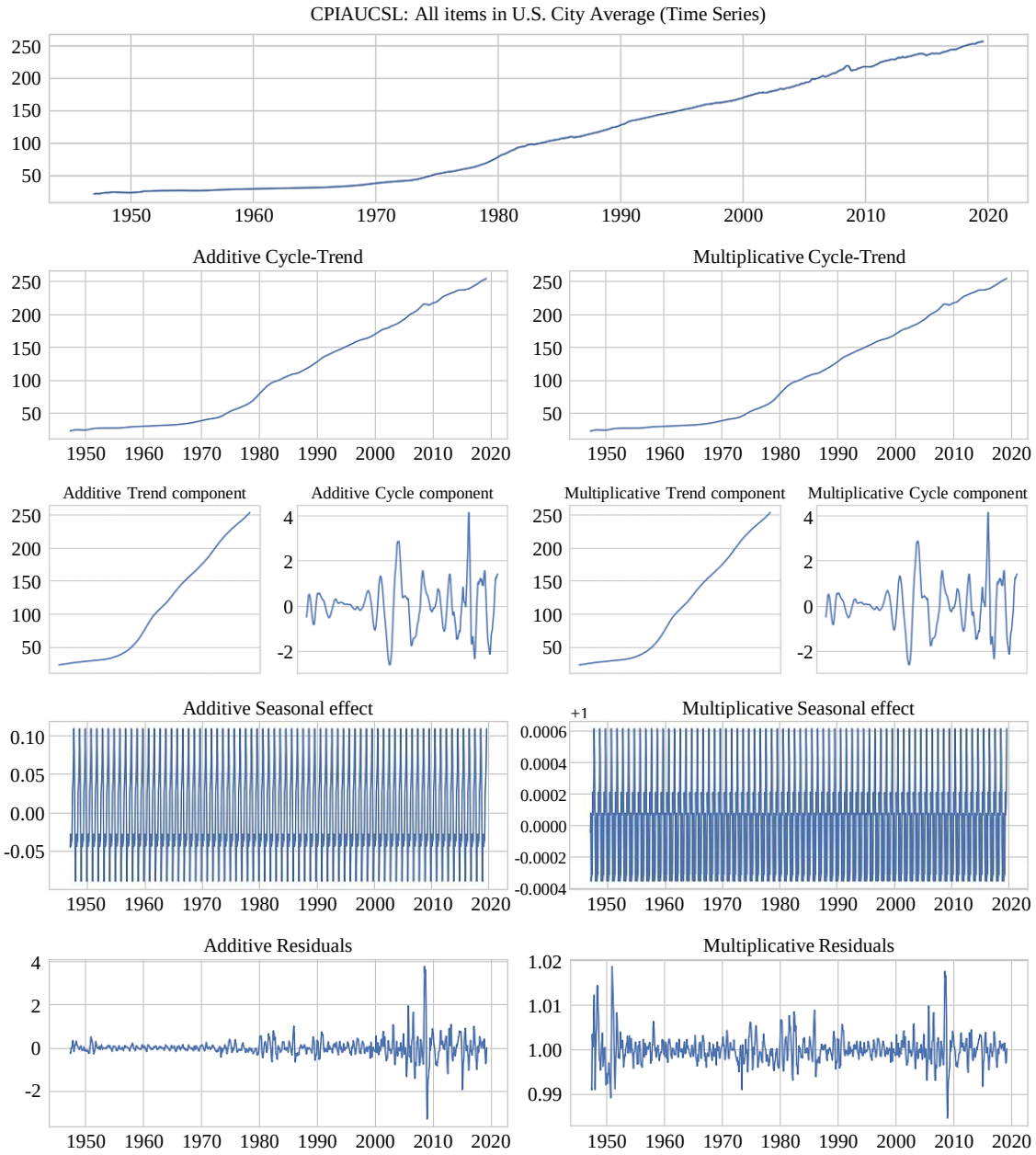


FIGURA E.1 – Série CPIAUCSL: Decomposição (aditiva e multiplicativa)

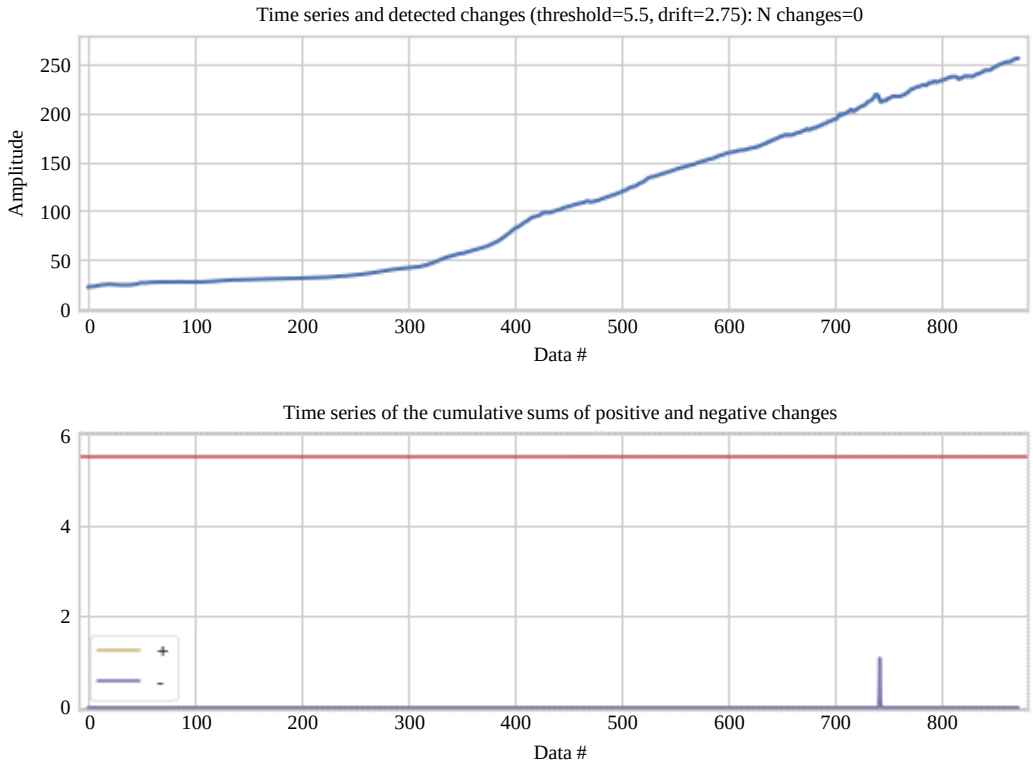


FIGURA E.2 – Série CPIAUCSL: Mudanças/quebras estruturais

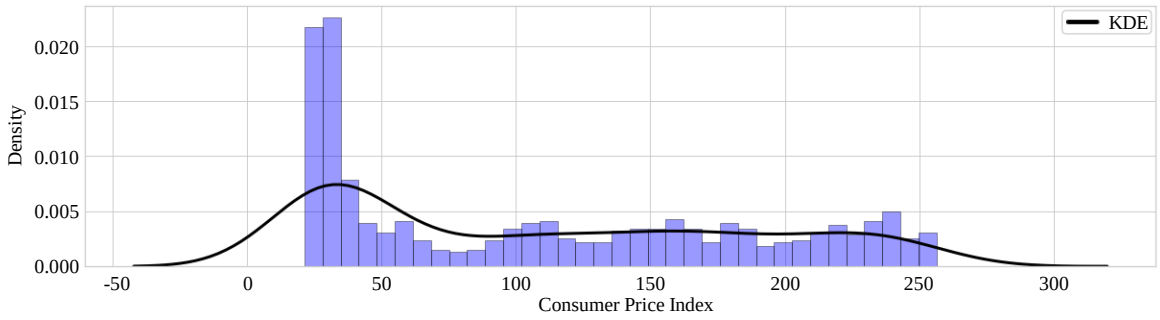


FIGURA E.3 – Série CPIAUCSL: Histograma (com curva de densidade)

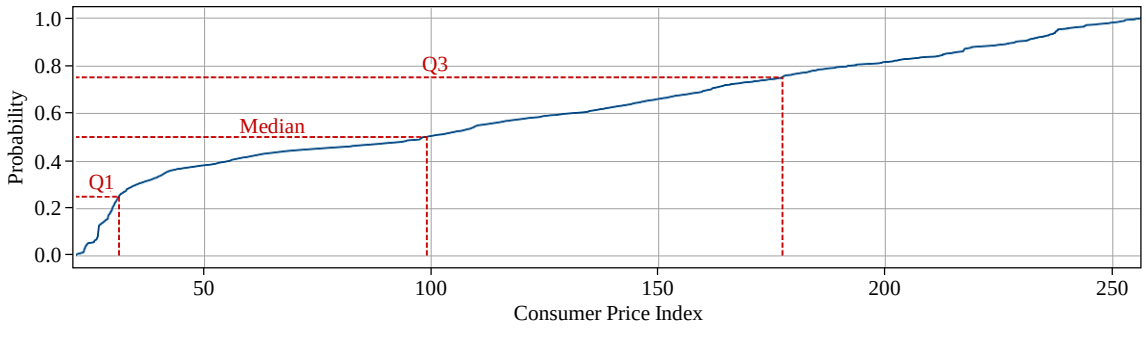


FIGURA E.4 – Série CPIAUCSL: Função cumulativa de distribuição

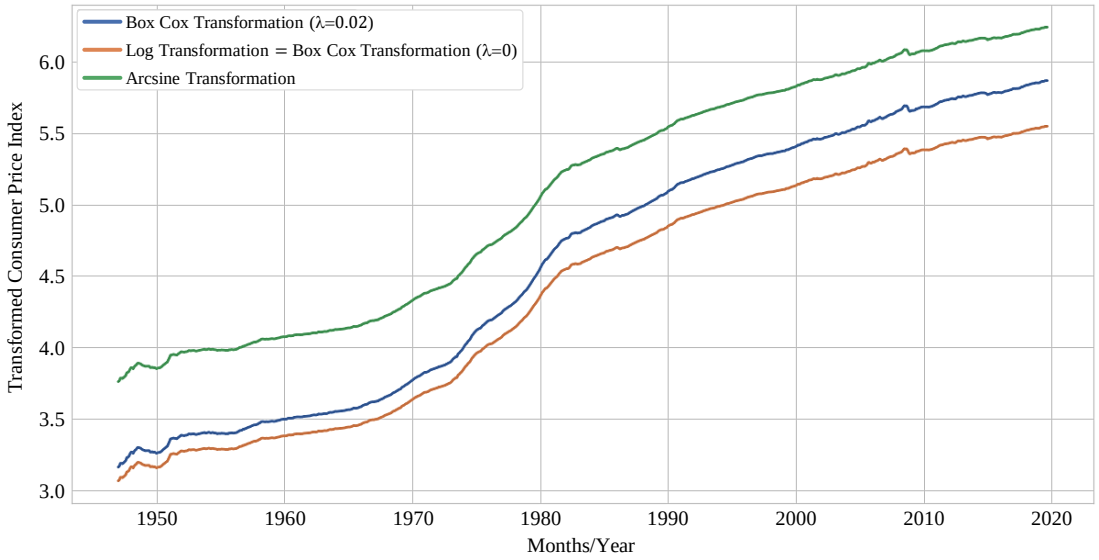


FIGURA E.5 – Série CPIAUCSL: Transformações (*Box-Cox* e *Arcsin*)

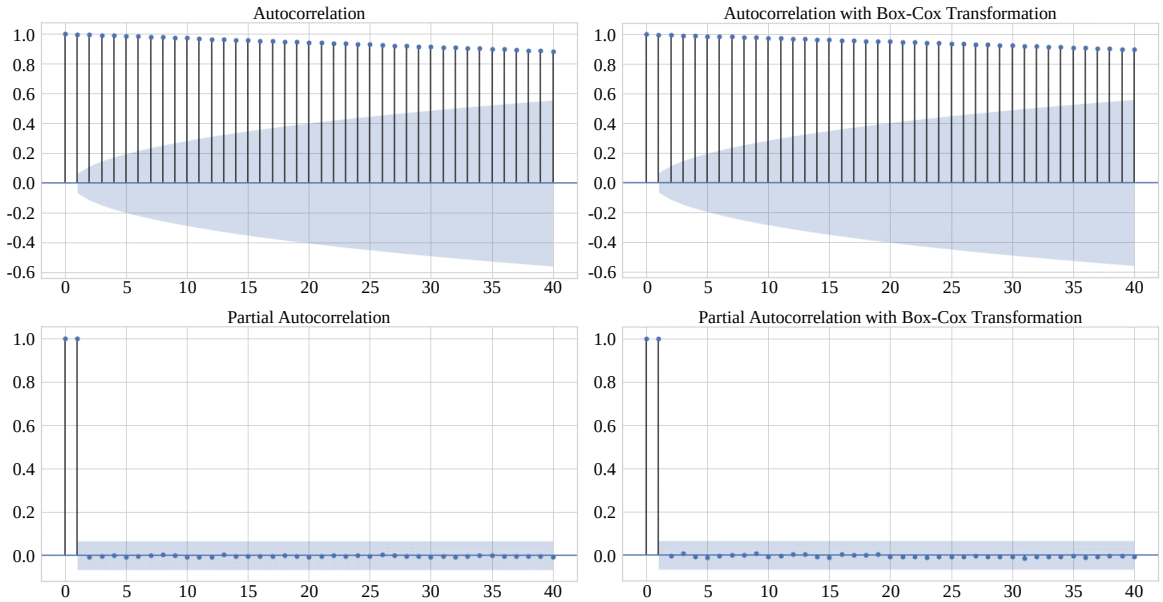


FIGURA E.6 – Série CPIAUCSL: Correlograma

E.2. SÉRIE VMT: INFORMAÇÕES COMPLEMENTARES

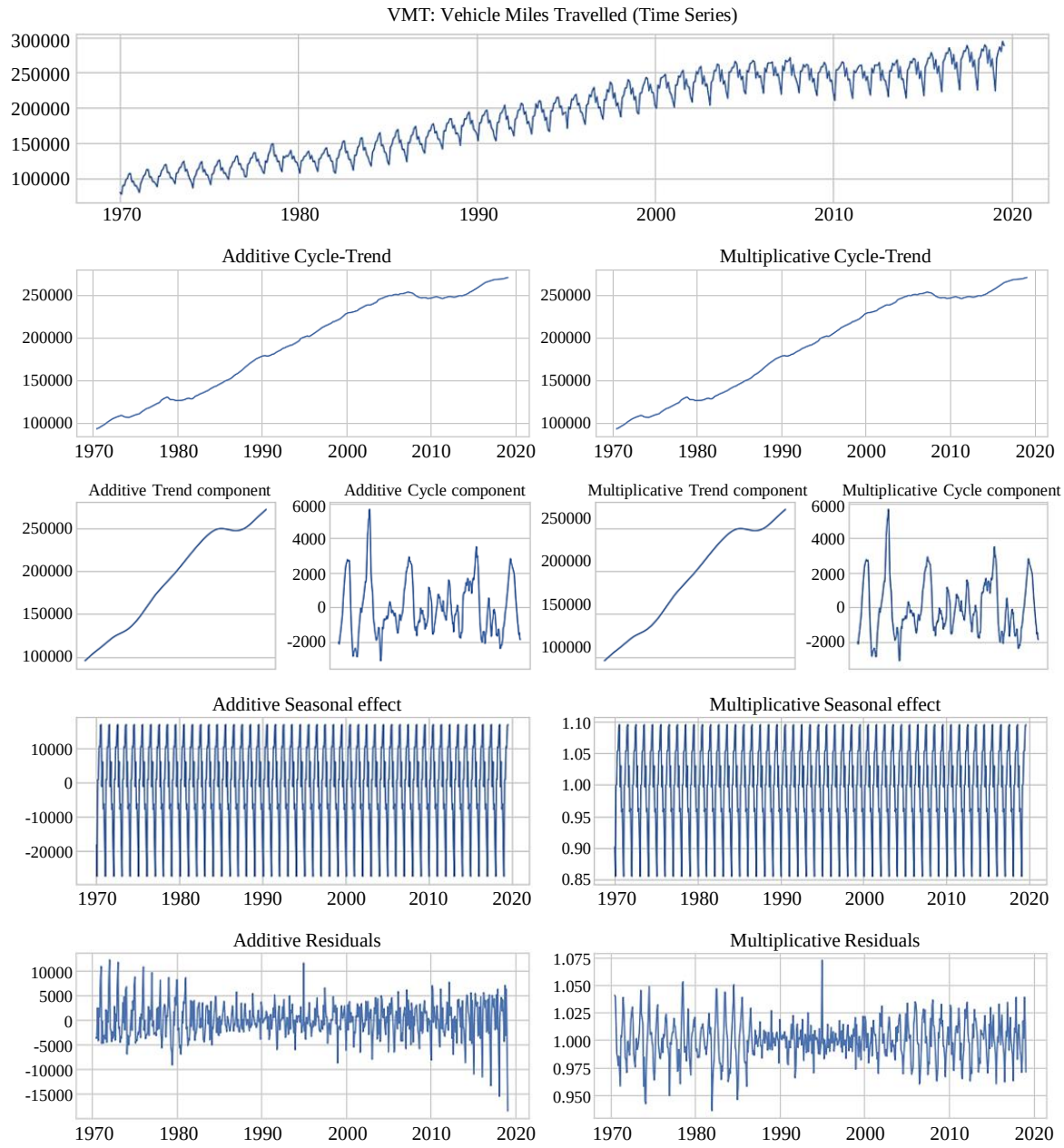


FIGURA E.7 – Série VMT: Decomposição (aditiva e multiplicativa)

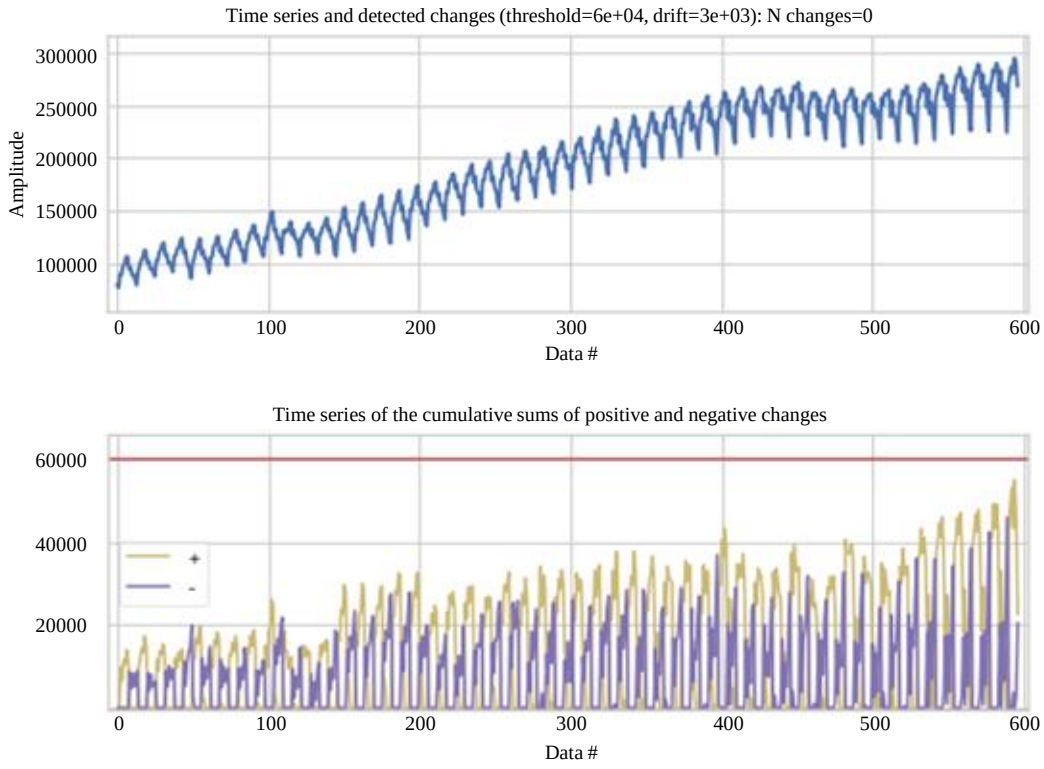


FIGURA E.8 – Série VMT: Mudanças/quebras estruturais

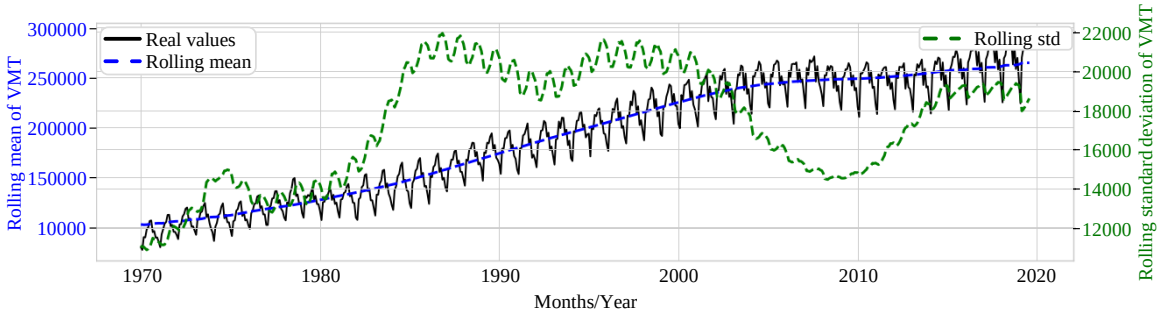


FIGURA E.9 – Série VMT: *Rolling mean* e *Rolling std*

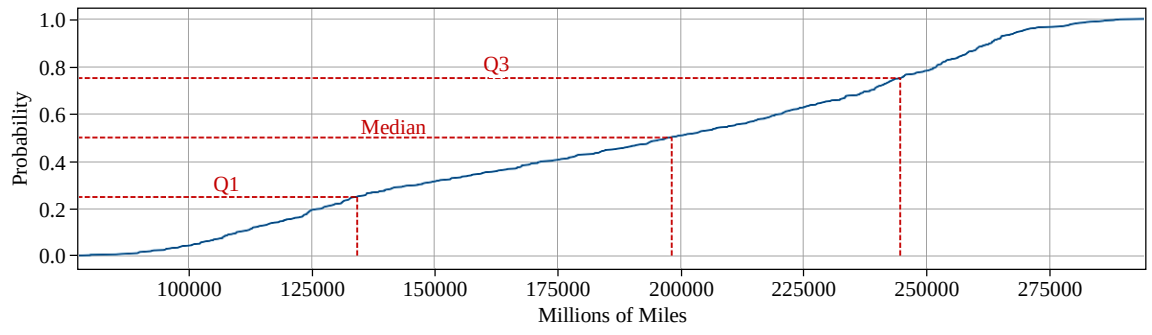


FIGURA E.10 – Série VMT: Função cumulativa de distribuição

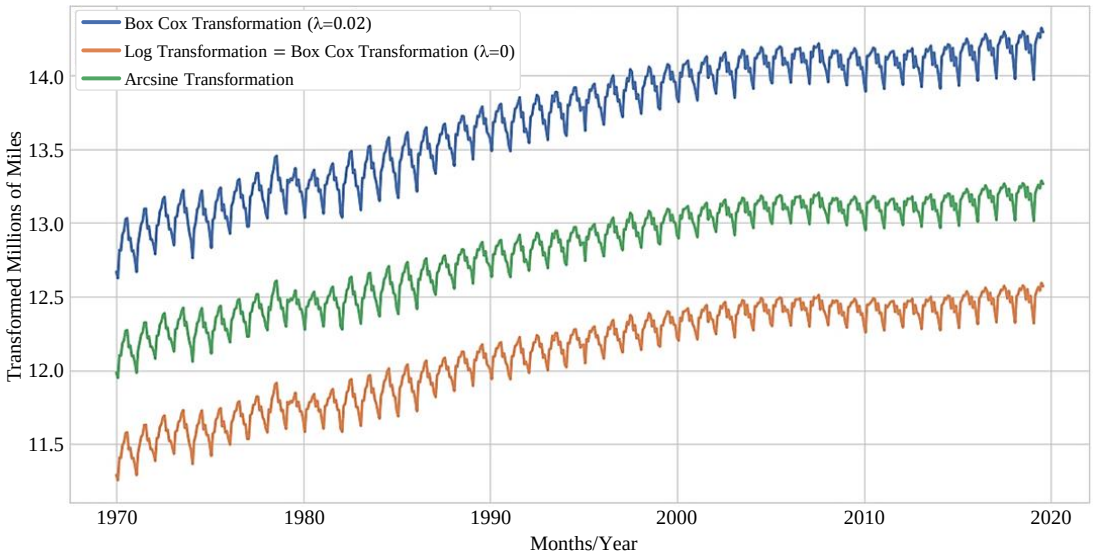


FIGURA E.11 – Série VMT: Transformações (*Box-Cox* e *Arcsin*)

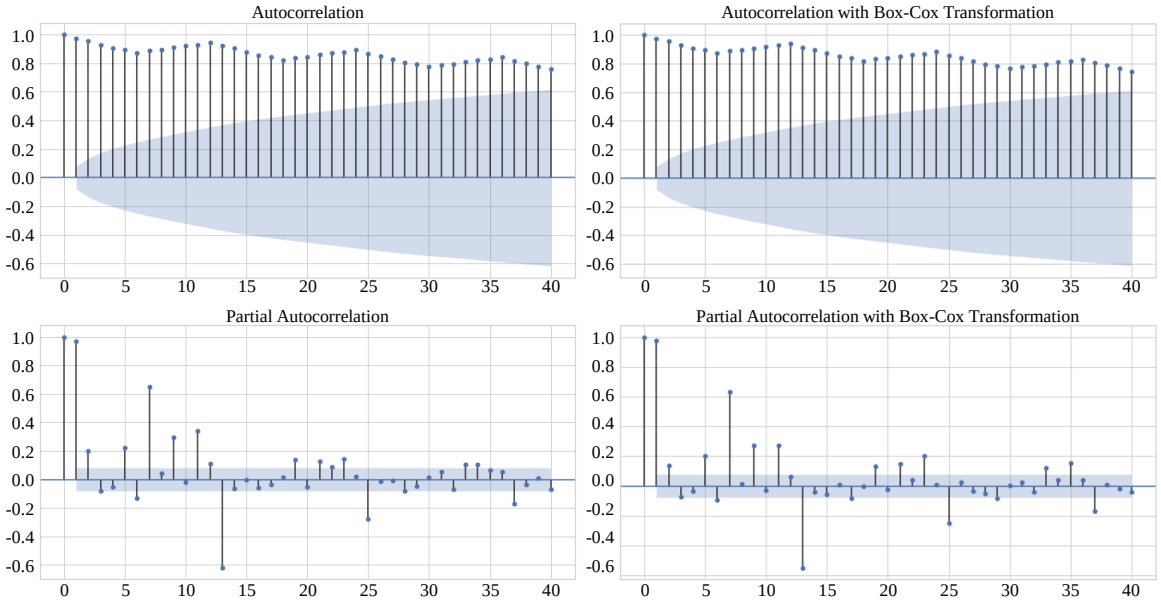


FIGURA E.12 – Série VMT: Correlograma

E.3. SÉRIE PSI 20: INFORMAÇÕES COMPLEMENTARES

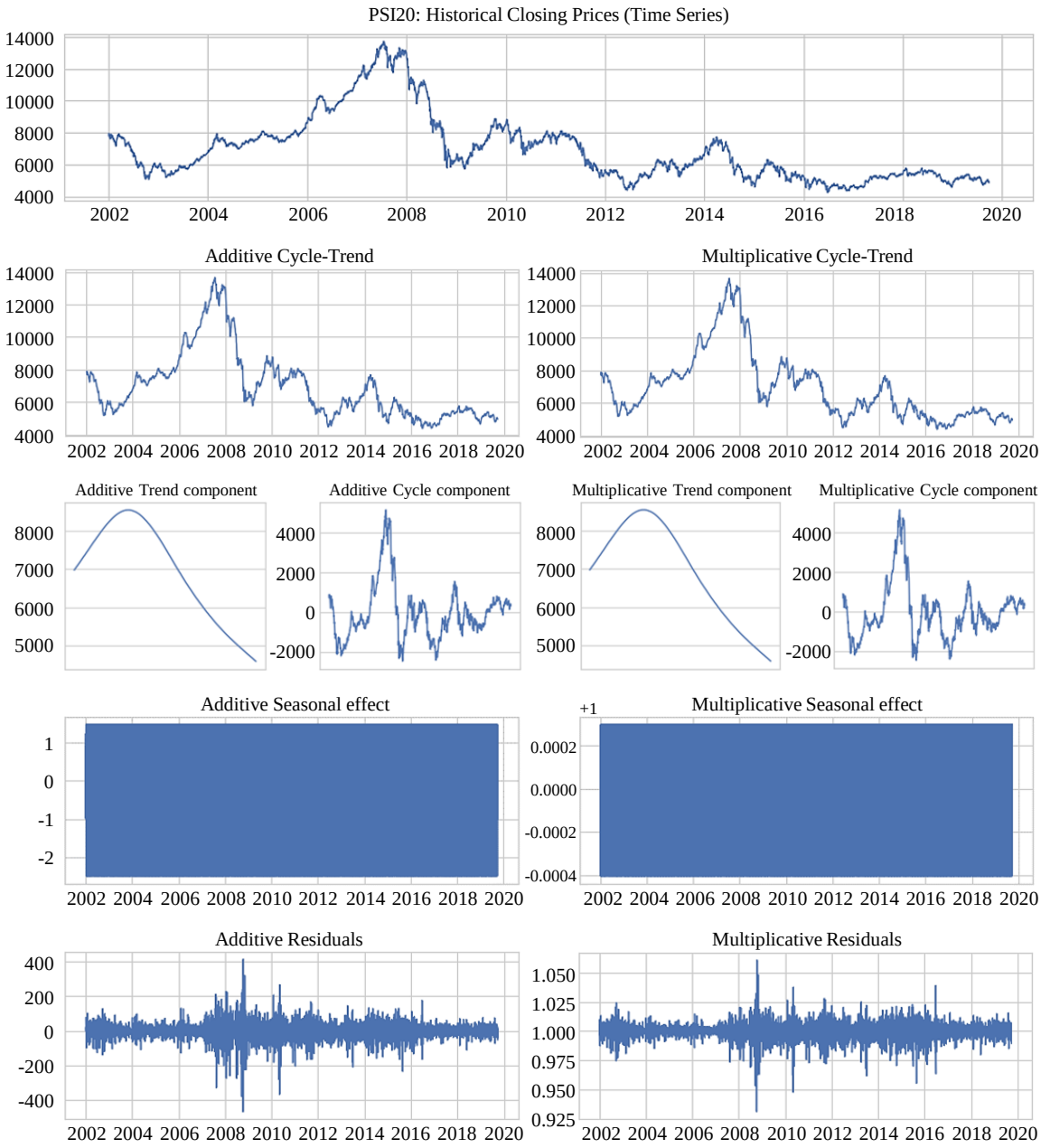


FIGURA E.13 – Série PSI 20: Decomposição (aditiva e multiplicativa)

FIGURA E.14 – Série PSI 20: *Rolling mean* e *Rolling std*

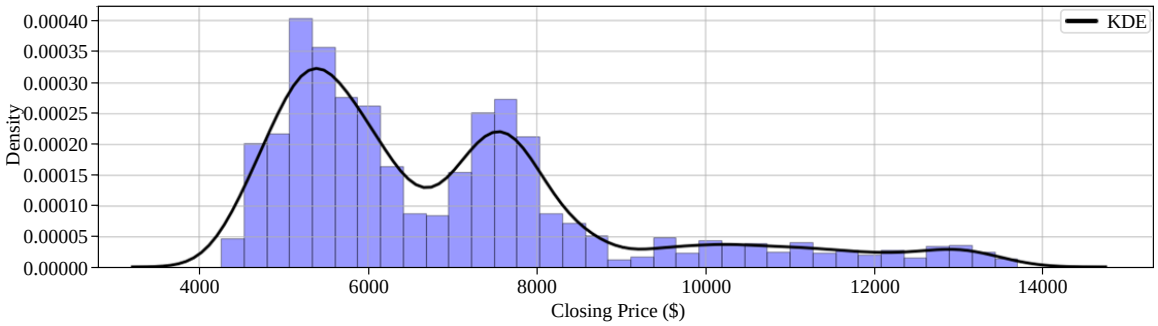


FIGURA E.15 – Série PSI 20: Histograma (com curva de densidade)

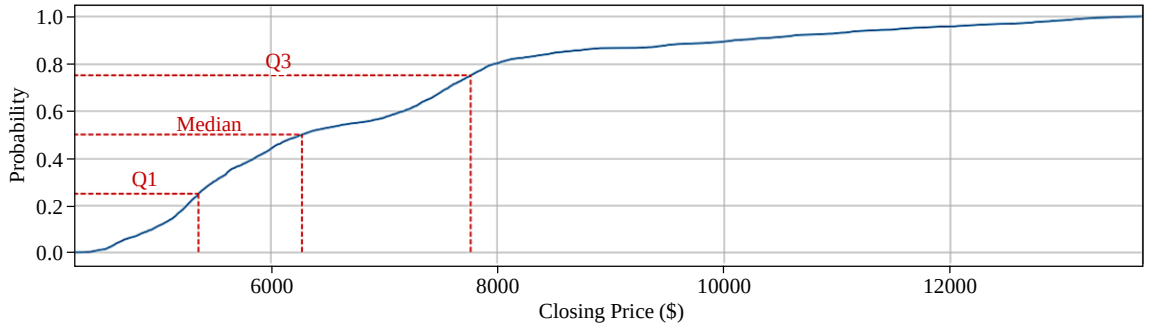


FIGURA E.16 – Série PSI 20: Função cumulativa de distribuição

FIGURA E.17 – Série PSI 20: Transformações (*Box-Cox* e *Arcsin*)

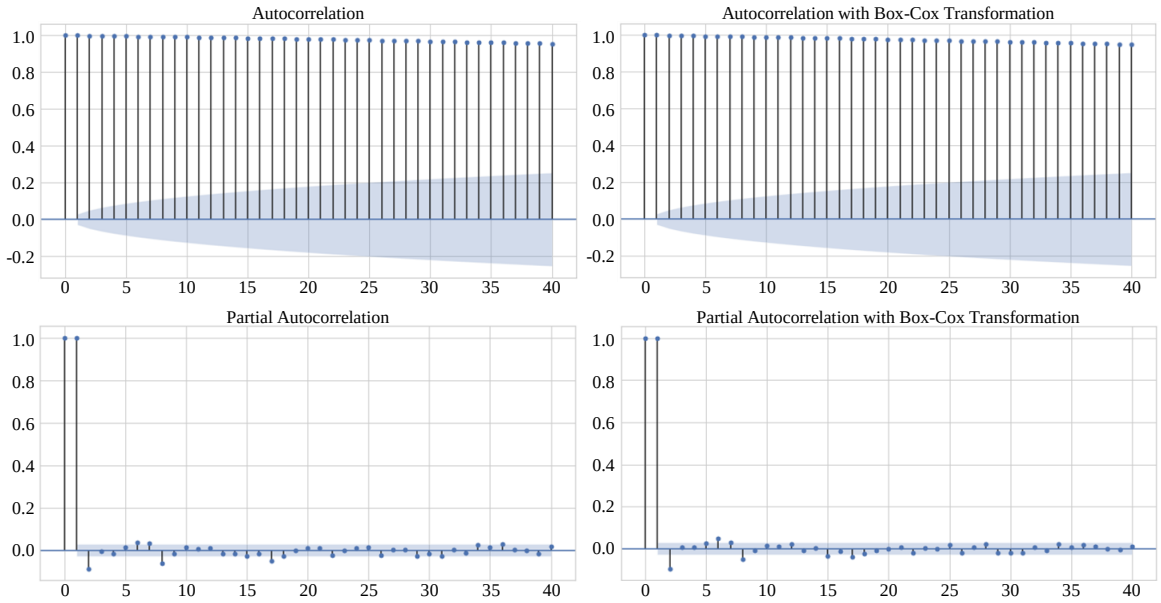


FIGURA E.18 – Série PSI 20: Correlograma

E.4. SÉRIE SPY: INFORMAÇÕES COMPLEMENTARES

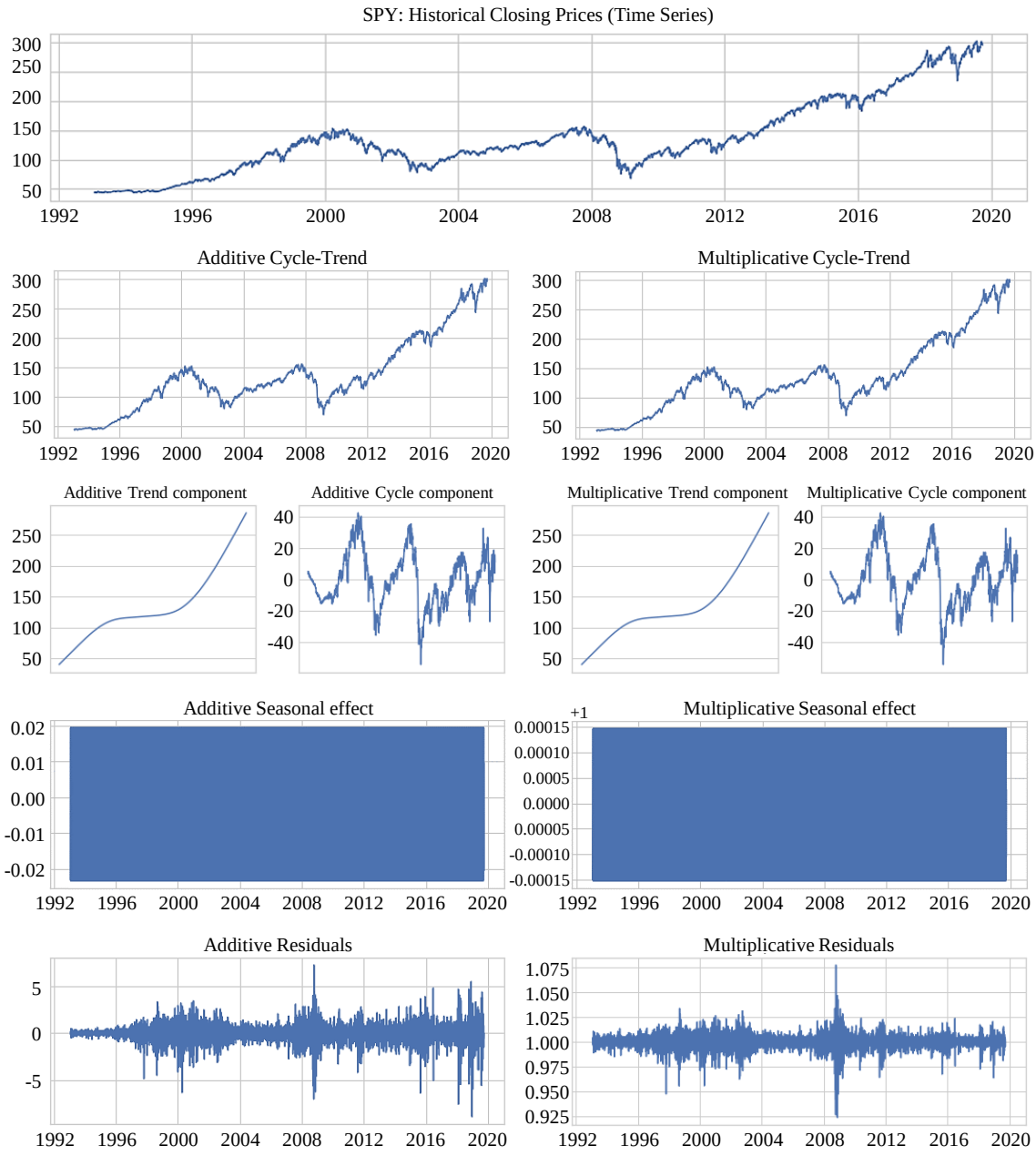


FIGURA E.19 – Série SPY: Decomposição (aditiva e multiplicativa)

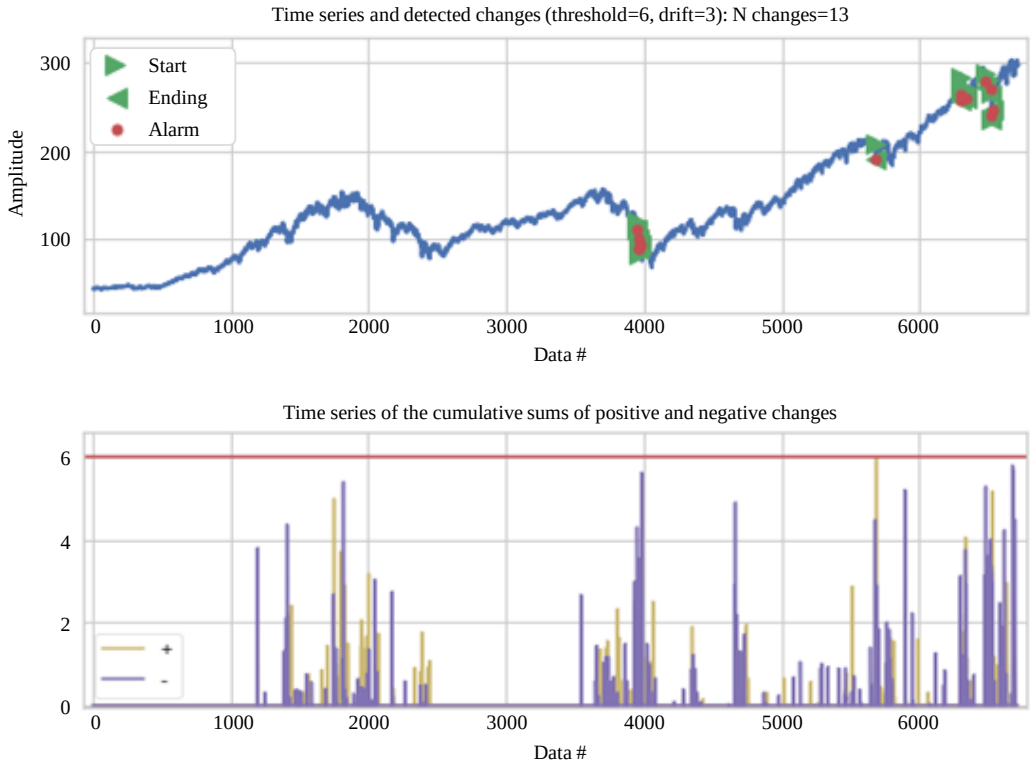


FIGURA E.20 – Série SPY: Mudanças/quebras estruturais

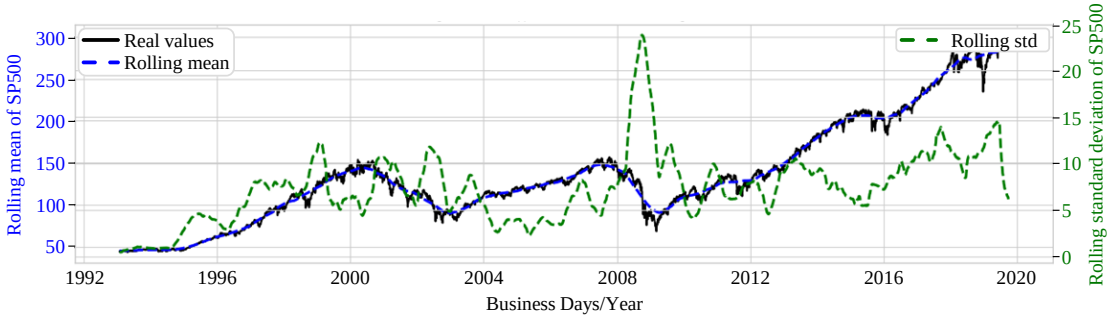


FIGURA E.21 – Série SPY: *Rolling mean* e *Rolling std*

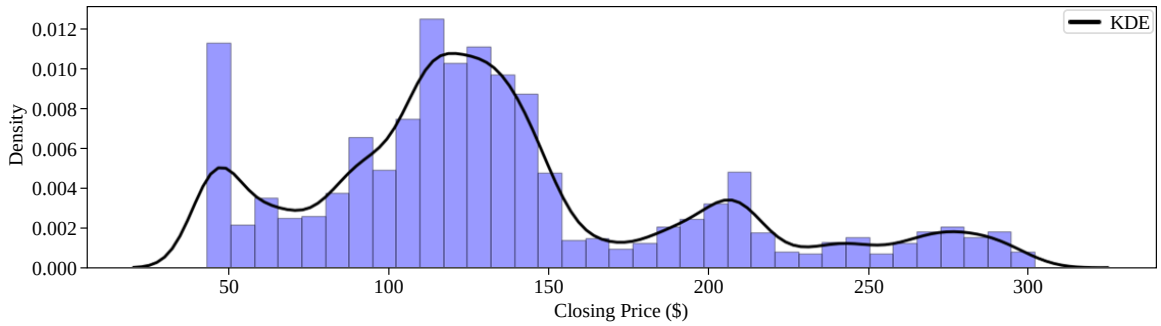


FIGURA E.22 – Série SPY: Histograma (com curva de densidade)

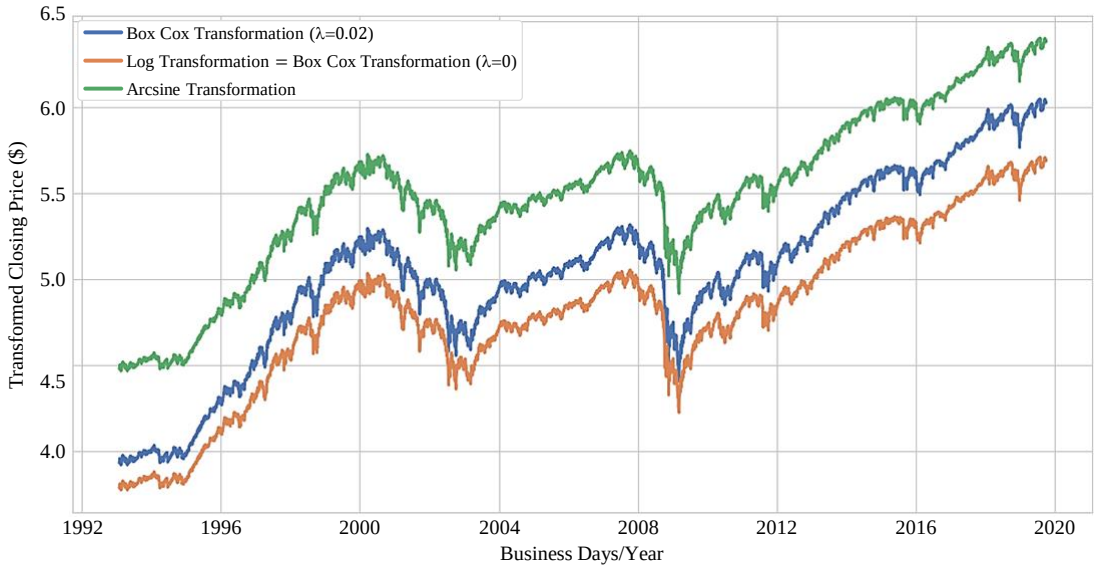


FIGURA E.23 – Série SPY: Transformações (*Box-Cox* e *Arcsin*)

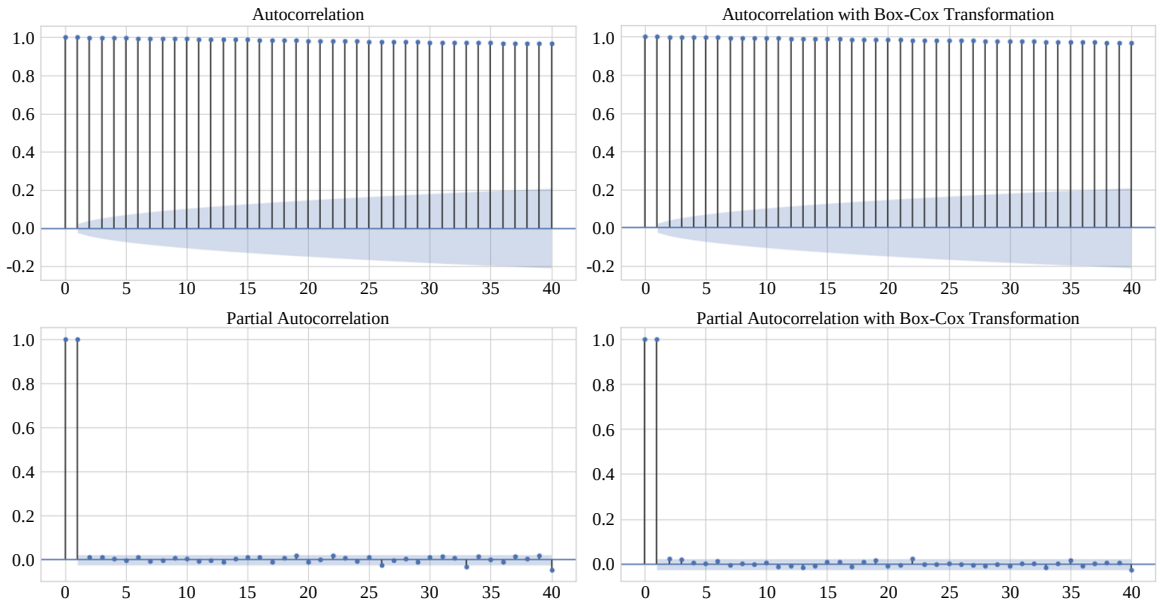


FIGURA E.24 – Série SPY: Correlograma

ANEXO F

F.1. ESTABILIZAÇÃO DAS SÉRIES TEMPORAIS

Designação: Série em nível – NAME; Série logaritmizada – Log(NAME); Série em diferenças –Dif(NAME); Série em diferenças de logaritmos –DifLog(NAME)

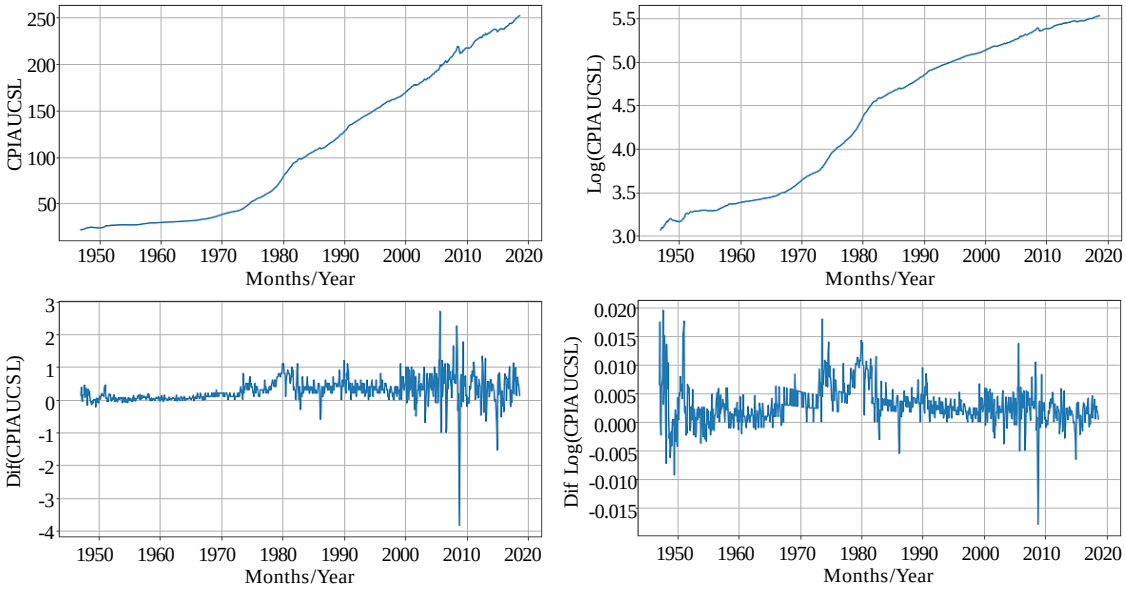


FIGURA F.1 – Estabilização da série CPIAUCSL

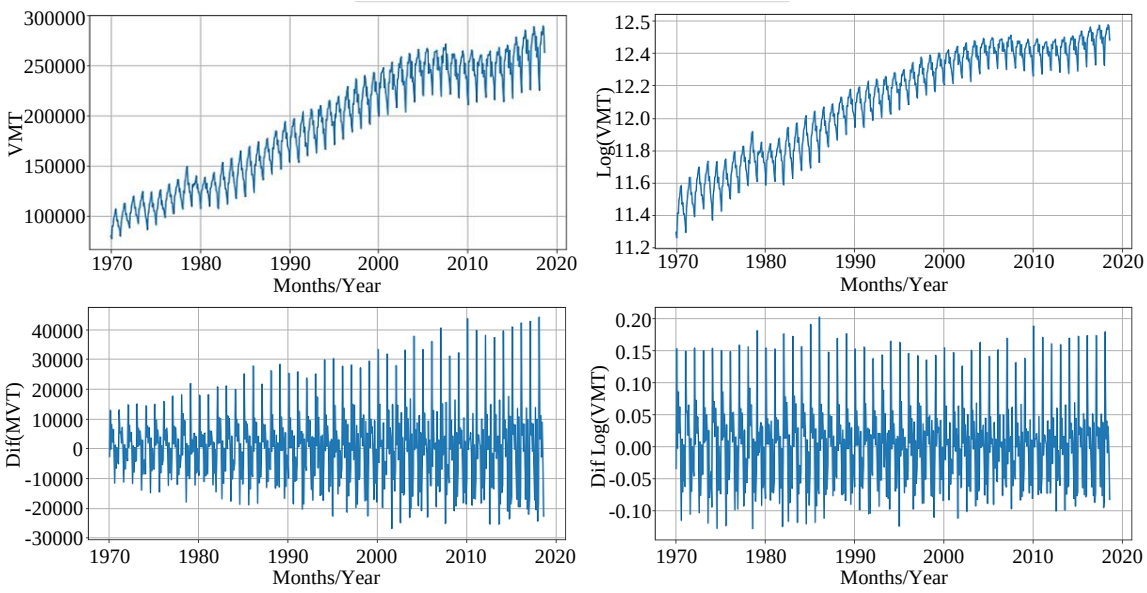


FIGURA F.2 – Estabilização da série VMT

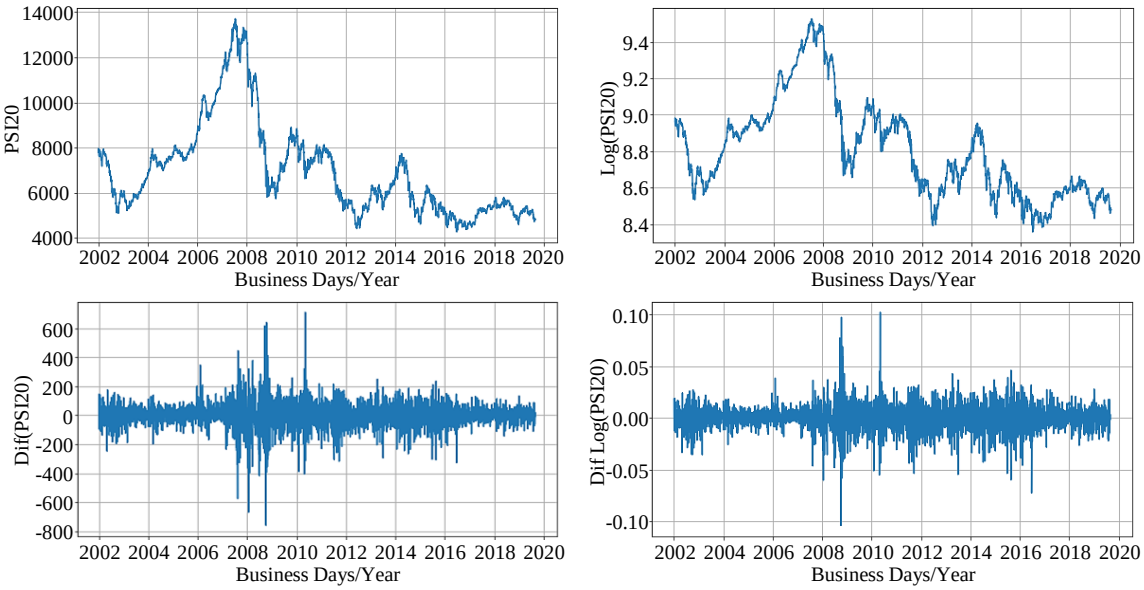


FIGURA F.3 – Estabilização da série PSI 20

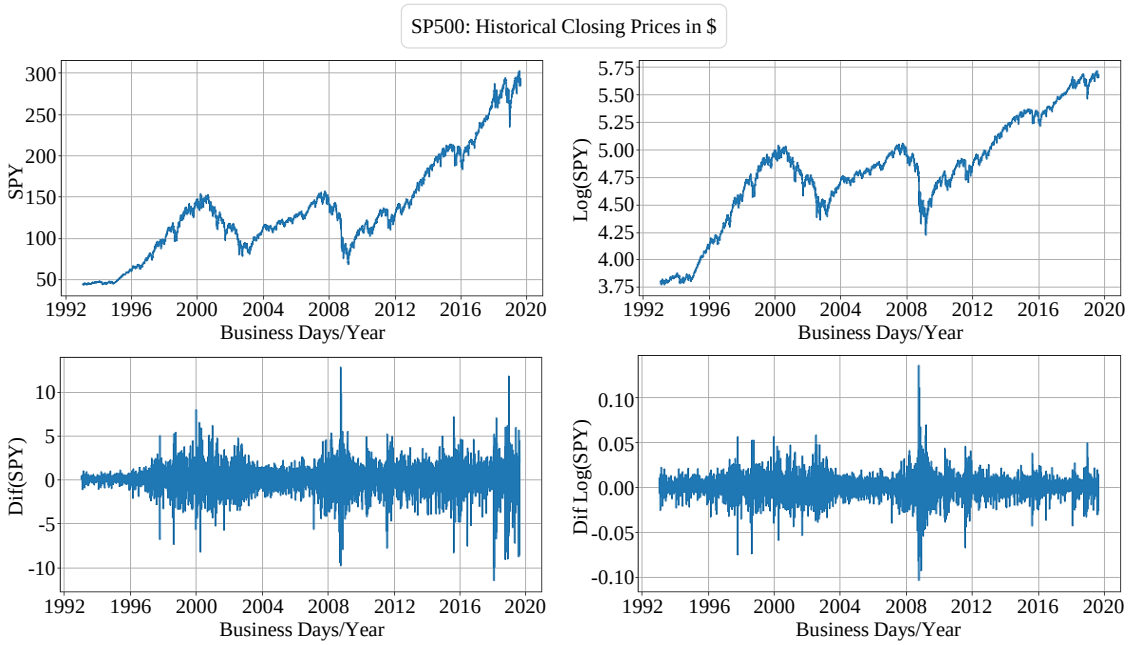


FIGURA F.4 – Estabilização da série SPY

F.2. CORRELOGRAMAS DAS SÉRIES TEMPORAIS

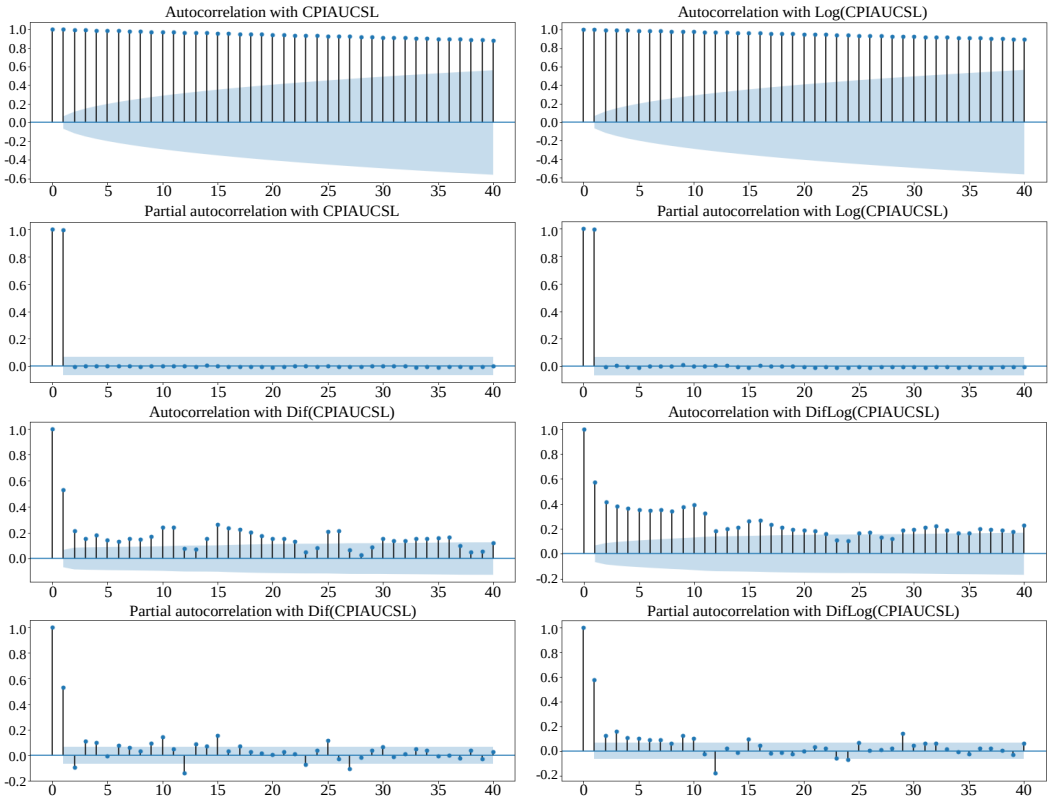


FIGURA F.5 – Correlogramas da série CPIAUCSL (em nível e estabilizada)

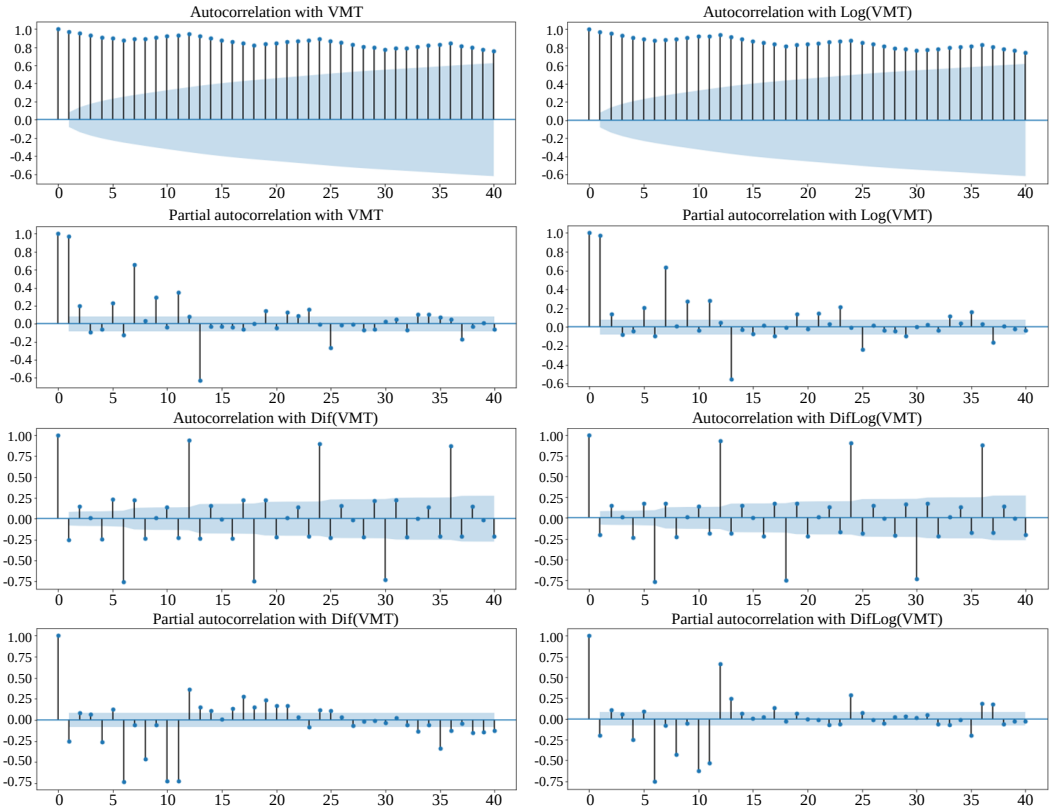


FIGURA F.6 – Correlogramas da série VMT (em nível e estabilizada)

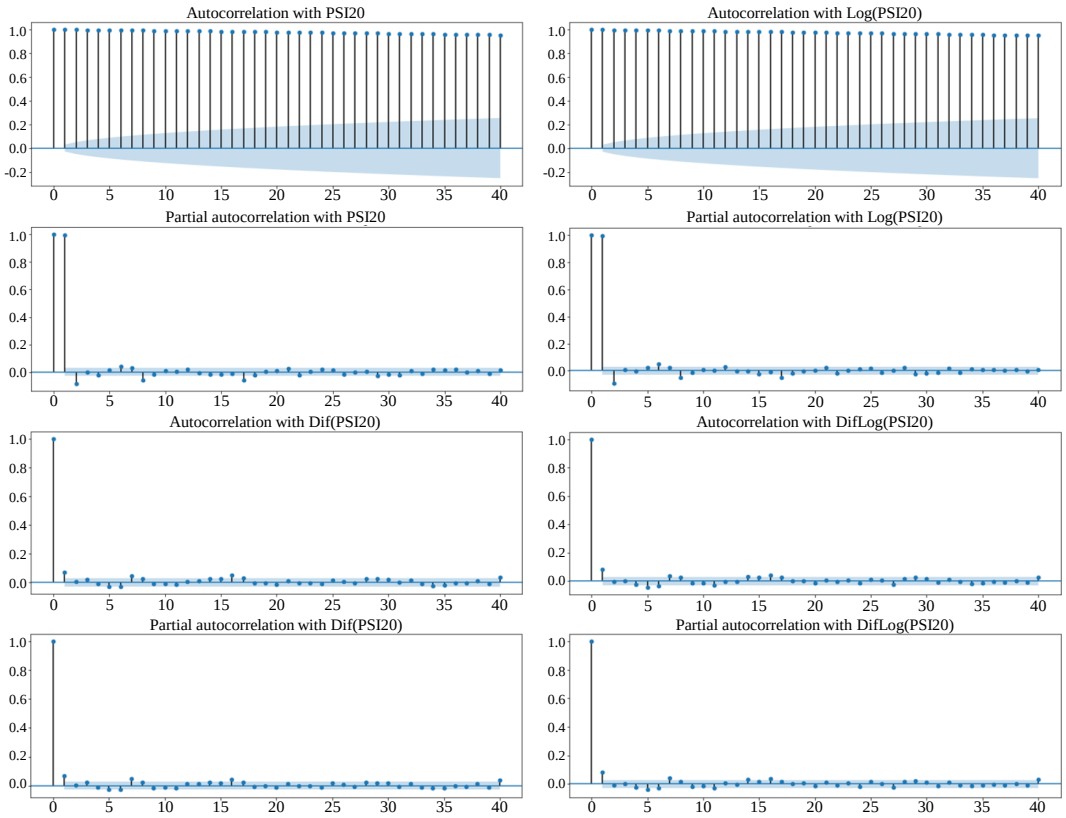


FIGURA F.7 – Correlogramas da série PSI 20 (em nível e estabilizada)

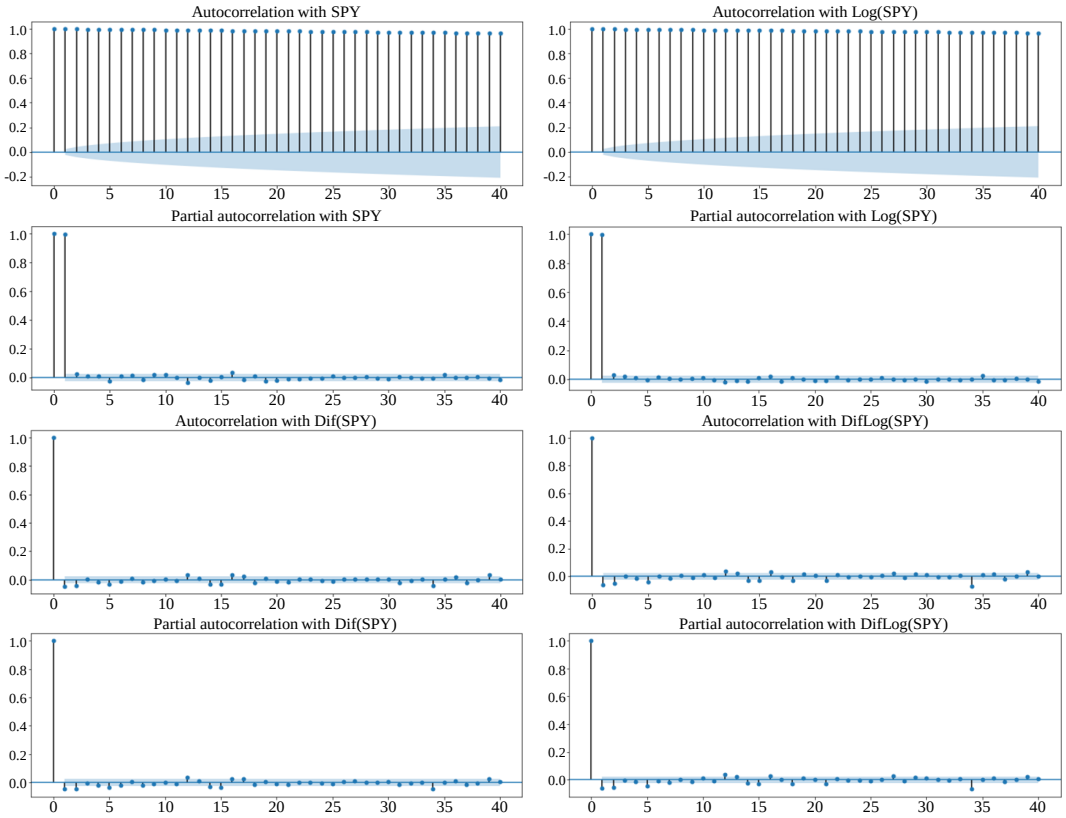


FIGURA F.8 – Correlogramas da série SPY (em nível e estabilizada)

F.3. MODELOS ARMA: SELEÇÃO E ESTIMAÇÃO

TABELA F.1 – Modelo ARMA estimado: série CPIAUCSL

Dep. Variable:		----------		**No. Observations:**		860
Model:		ARIMA (3, 1, 4)		**Log Likelihood**		3875.153
Date:		----------		**AIC**		-7732.306
Time:		----------		**BIC**		-7689.493
Sample:		----------		**HQIC**		-7715.916
	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
const	0.0029	0.001	5.077	0.000	0.002	0.004
ar.L1	0.0682	0.029	2.384	0.017	0.012	0.124
ar.L2	-0.0917	0.026	-3.516	0.000	-0.143	-0.041
ar.L3	0.9210	0.026	36.085	0.000	0.871	0.971
ma.L1	0.3588	0.045	8.018	0.000	0.271	0.447
ma.L2	0.2518	0.038	6.573	0.000	0.177	0.327
ma.L3	-0.7476	0.037	-20.033	0.000	-0.821	-0.674
ma.L4	-0.1992	0.039	-5.155	0.000	-0.275	-0.123

TABELA F.2 – Modelo ARMA estimado: série VMT

Dep. Variable:		----------		**No. Observations:**		585
Model:		SARIMA (4, 1, 1)x(1, 1, 3, 12)		**Log Likelihood**		1592.959
Date:		----------		**AIC**		-3165.919
Time:		----------		**BIC**		-3122.427
Sample:		----------		**HQIC**		-3148.952
	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
ar.L1	0.3429	0.080	4.270	0.000	0.186	0.500
ar.L2	0.1112	0.049	2.278	0.023	0.016	0.207
ar.L3	0.0480	0.051	0.950	0.342	-0.051	0.147
ar.L4	-0.1740	0.048	-3.639	0.000	-0.268	-0.080
ma.L1	-0.7606	0.073	-10.354	0.000	-0.905	-0.617
ar.S.L12	-0.3251	0.271	-1.200	0.230	-0.856	0.206
ma.S.L12	-0.3038	0.268	-1.135	0.256	-0.828	0.221
ma.S.L24	-0.4395	0.152	-2.884	0.004	-0.738	-0.141
ma.S.L36	0.0686	0.074	0.923	0.356	-0.077	0.214
sigma2	0.0002	1e-05	21.838	0.000	0.000	0.000

TABELA F.3 – Modelo ARMA estimado: série PSI 20

Dep. Variable:		----------		**No. Observations:**		4512
Model:		ARIMA (4, 1, 4)		**Log Likelihood**		13734.594
Date:		----------		**AIC**		-27449.189
Time:		----------		**BIC**		-27385.044
Sample:		----------		**HQIC**		-27426.590
	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
const	-0.0001	0.000	-0.599	0.549	-0.000	0.000
ar.L1	0.8286	0.174	4.754	0.000	0.487	1.170
ar.L2	-1.1549	0.186	-6.196	0.000	-1.520	-0.790
ar.L3	0.5461	0.184	2.971	0.003	0.186	0.906
ar.L4	-0.5130	0.151	-3.408	0.001	-0.808	-0.218
ma.L1	-0.7494	0.178	-4.220	0.000	-1.098	-0.401
ma.L2	1.0903	0.185	5.882	0.000	0.727	1.454
ma.L3	-0.4553	0.183	-2.493	0.013	-0.813	-0.097
ma.L4	0.4567	0.158	2.898	0.004	0.148	0.766

TABELA F.4 – Modelo ARMA estimado: série S&P500

Dep. Variable:		----------		**No. Observations:**		6693
Model:		ARIMA (4, 1, 2)		**Log Likelihood**		20427.377
Date:		----------		**AIC**		-40838.754
Time:		----------		**BIC**		-40784.284
Sample:		----------		**HQIC**		-40819.942
	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
const	0.0003	0.000	2.327	0.020	4.43e-05	0.001
ar.L1	1.7155	0.025	68.357	0.000	1.666	1.765
ar.L2	-0.9019	0.031	-29.488	0.000	-0.962	-0.842
ar.L3	0.0295	0.024	1.211	0.226	-0.018	0.077
ar.L4	-0.0502	0.013	-3.981	0.000	-0.075	-0.025
ma.L1	-1.7827	0.022	-81.401	0.000	-1.826	-1.740
ma.L2	0.9616	0.020	47.764	0.000	0.922	1.001

F.4. MODELOS ARMA: ANÁLISE DA COMPONENTE RESIDUAL

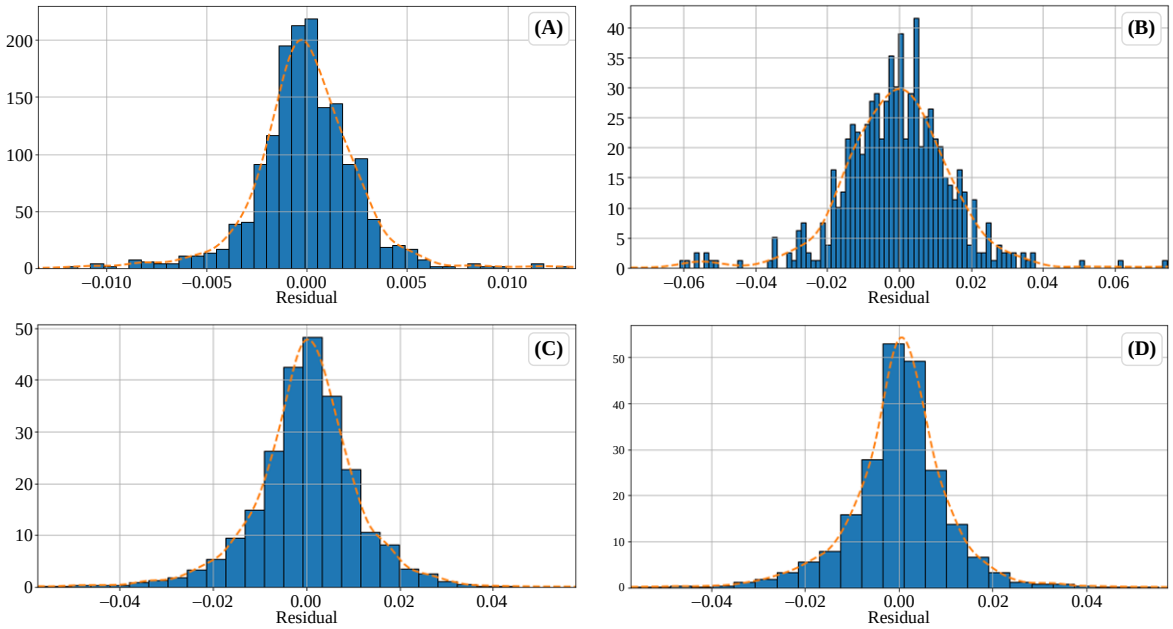


FIGURA F.9 – Histogramas dos resíduos de modelos ARMA selecionados: **(A)** CPIAUCSL–$ARIMA(3,1,4)$; **(B)** VMT–$SARIMA(4,1,1)n(1,1,3,12)$; **(C)** PSI 20–$ARIMA(4,1,4)$; **(D)** SPY–$ARIMA(4,1,2)$

TABELA F.5 – Média e Variância da componente residual dos modelos ARMA

Série	CPIAUCSL	VMT	PSI 20	SPY
Média	0.0000	0.0007	0.0000	0.0000
Variância	0.0000	0.0002	0.0001	0.0001

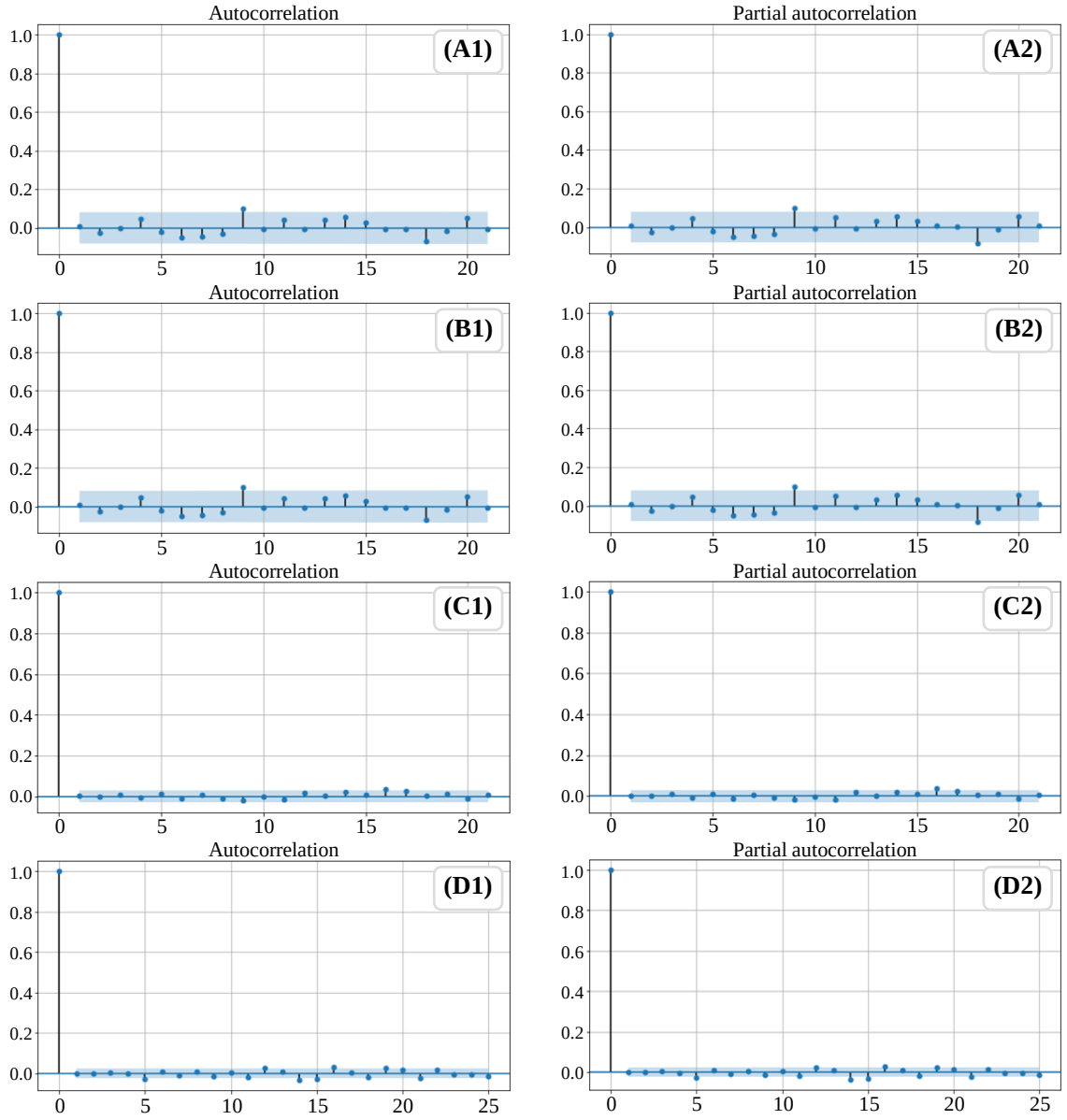


FIGURA F.10 – Correlogramas dos resíduos de modelos ARMA selecionados: **(A)** CPIAUCSL–$ARIMA(3,1,4)$; **(B)** VMT–$SARIMA(4,1,1)n(1,1,3,12)$; **(C)** PSI 20–$ARIMA(4,1,4)$; **(D)** SPY–$ARIMA(4,1,2)$

TABELA F.6 – Estudo da autocorrelação dos resíduos dos modelos ARMA: Estatística Box-Pierce-Ljung

Série	**CPIAUCSL**	**VMT**	**PSI 20**	**SPY**
statistic	0.0090	0.0419	0.0320	0.0004
p-value	0.9243	0.8379	0.8581	0.9834

Validação dos modelos ARMA

Tendo em conta o exposto na Sec. 2.6, para qualquer uma das séries temporais em estudo, temos:

- Resíduos com distribuição em "forma de sino" centrada em zero, com média e variância nulas;
- Correlograma dos resíduos (com representação gráfica das autocorrelações e autocorrelações parciais) semelhantes ao de um ruído branco, isto é, sem autocorrelações e autocorrelações parciais significativas (todas bastante próximas de zero);
- Valores da estatística de Box-Pierce-Ljung baixos para os resíduos de cada uma das séries, pelo que (com a respetiva probabilidade de erro) não se rejeita a hipótese nula de inexistência de autocorrelação dos resíduos para qualquer nível de significância, ou seja, a série dos resíduos, ao longo do tempo, é aleatória e independente.

Deste modo, podemos considerar adequados cada um dos quatro modelos selecionados (um para cada série temporal em estudo).

F.5. MODELOS ETS: SELEÇÃO

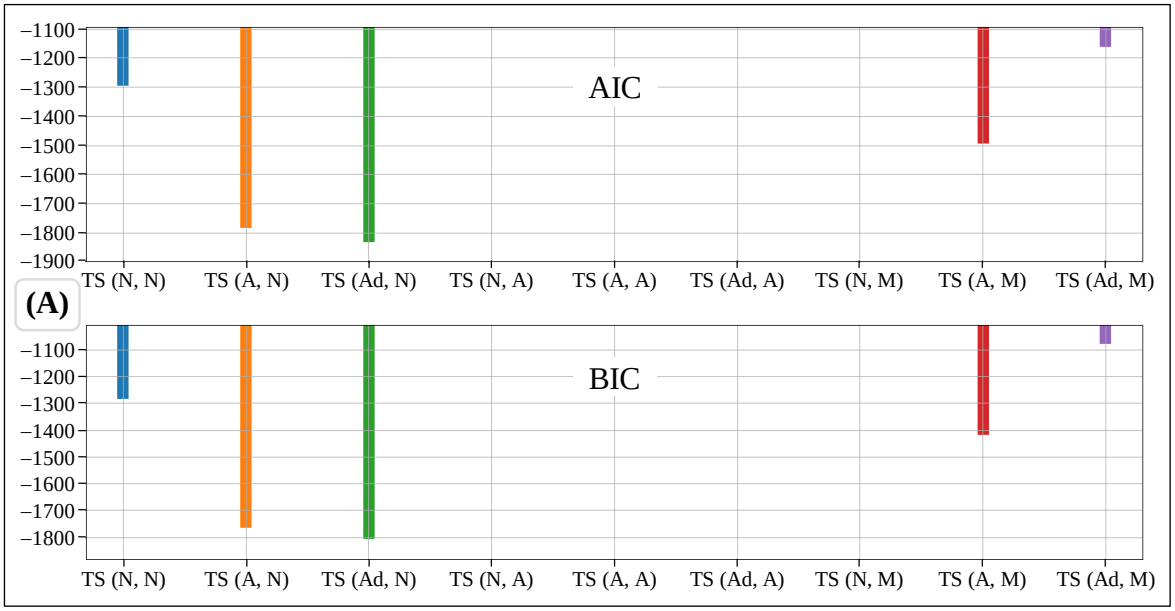


FIGURA F.11 – Critérios de Informação para modelos ETS da série CPIAUCSL

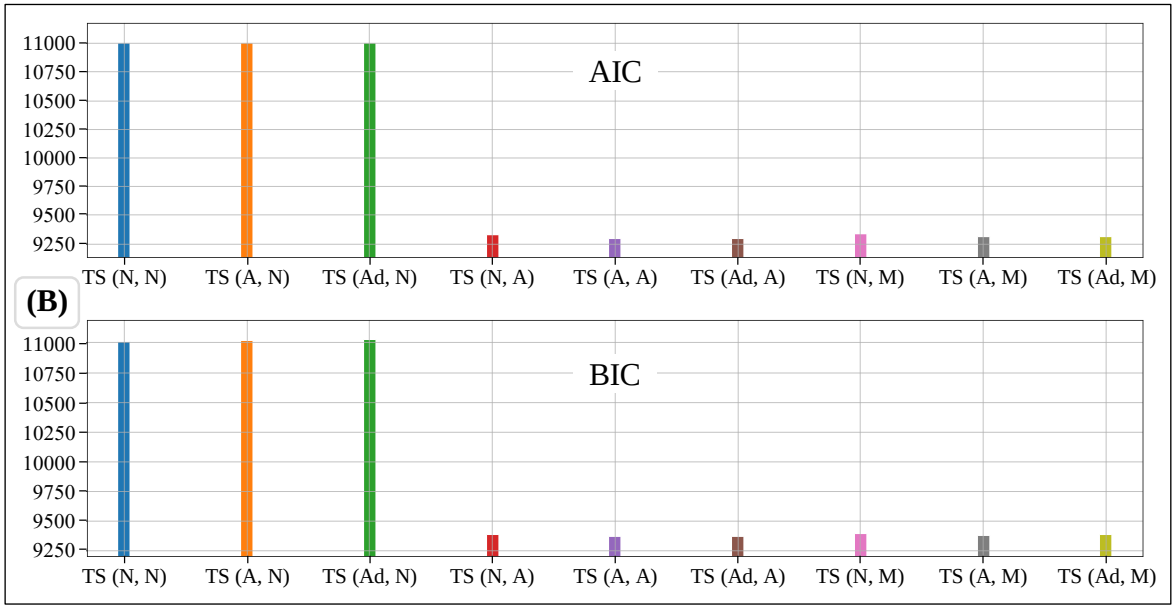


FIGURA F.12 – Critérios de Informação para modelos ETS da série VMT

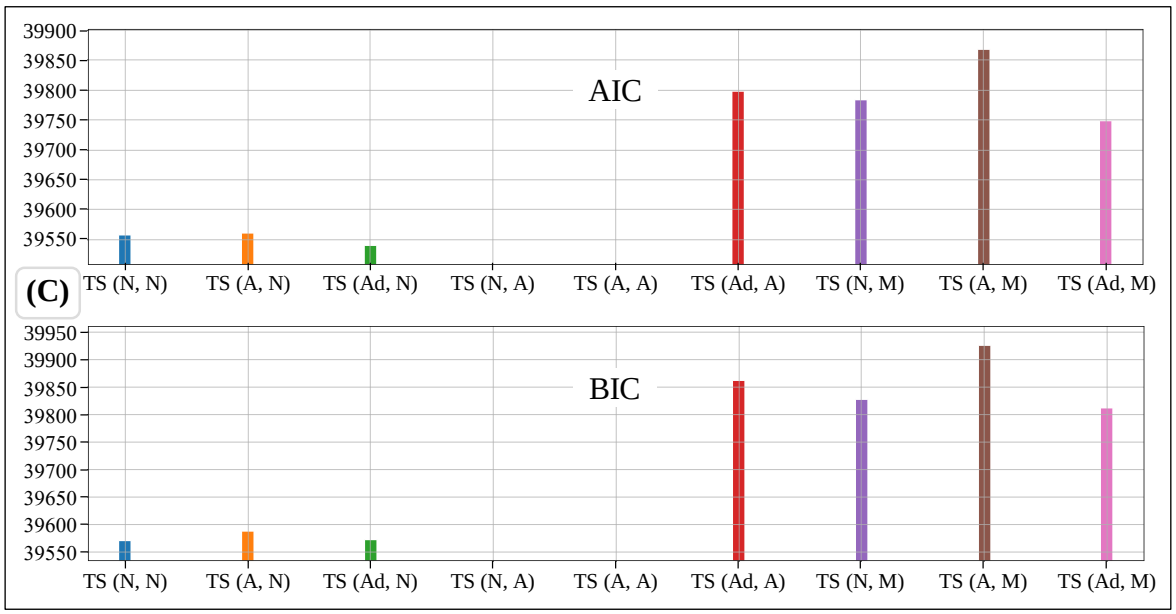


FIGURA F.13 – Critérios de Informação para modelos ETS da série PSI 20

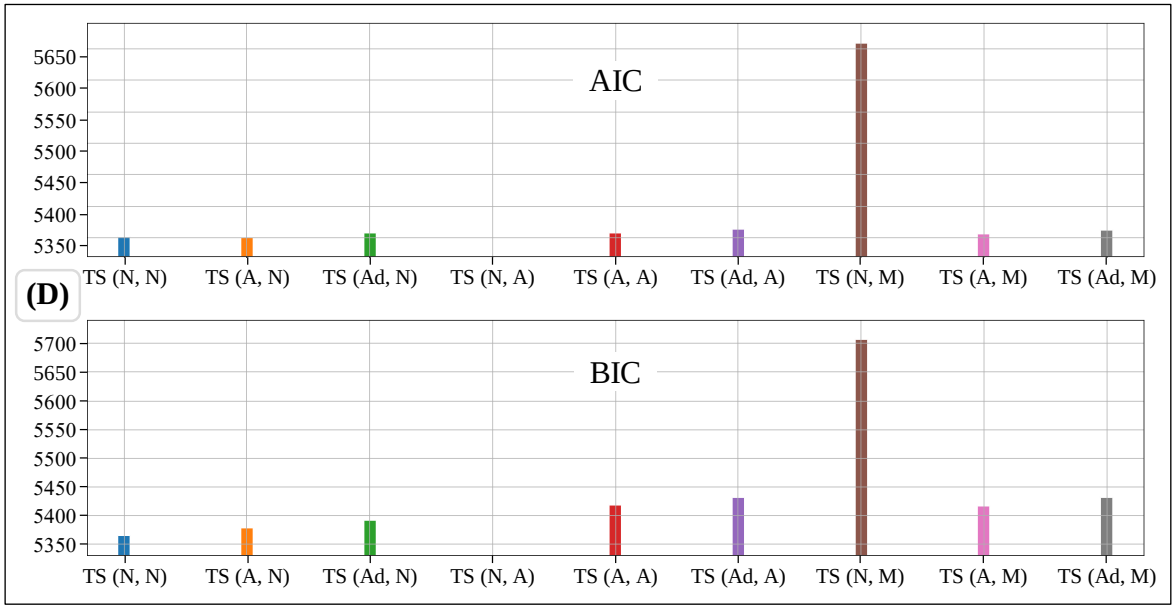


FIGURA F.14 – Critérios de Informação para modelos ETS da série SPY

F.6. MODELOS ETS: INFORMAÇÕES DA COMPONENTE RESIDUAL

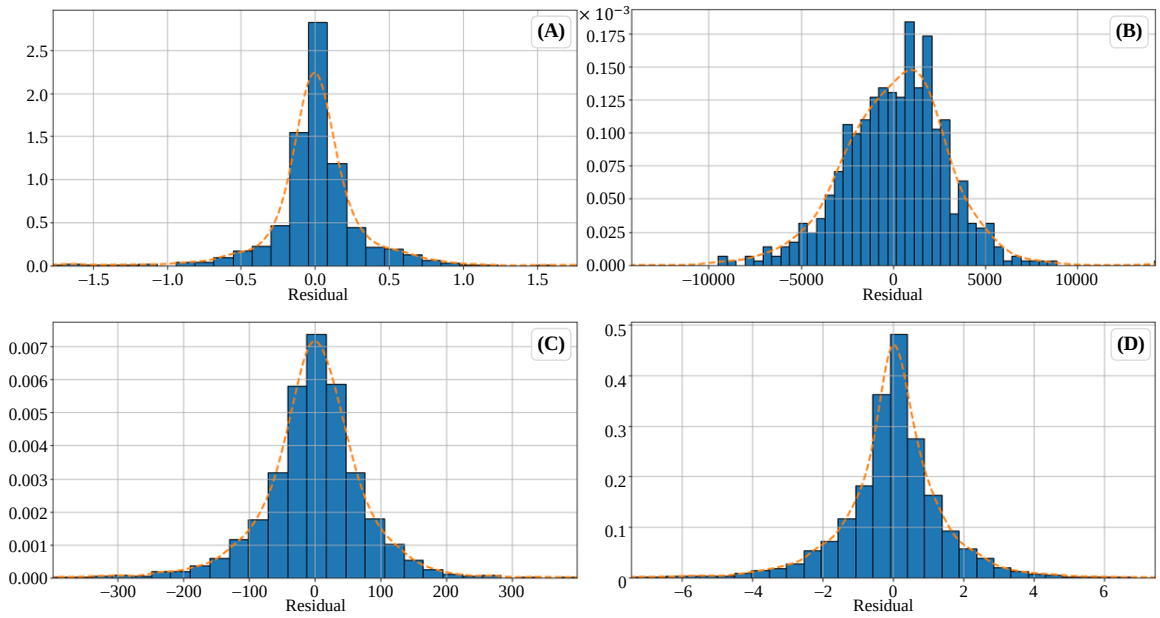


FIGURA F.15 – Histogramas dos resíduos de modelos ETS selecionados: **(A)** CPIAUCSL–$ETS(A,N)$; **(B)** VMT–$ETS(Ad,M)$; **(C)** PSI 20–$ETS(Ad,N)$; **(D)** SPY–$ETS(A,N)$

TABELA F.7 – Média e Variância da componente residual dos modelos ETS

Série	**CPIAUCSL**	**VMT**	**PSI 20**	**SPY**
Média	0.0027	129.046	-0.6062	0.0000
Variância	0.1249	7950660	6369.3	2.2214

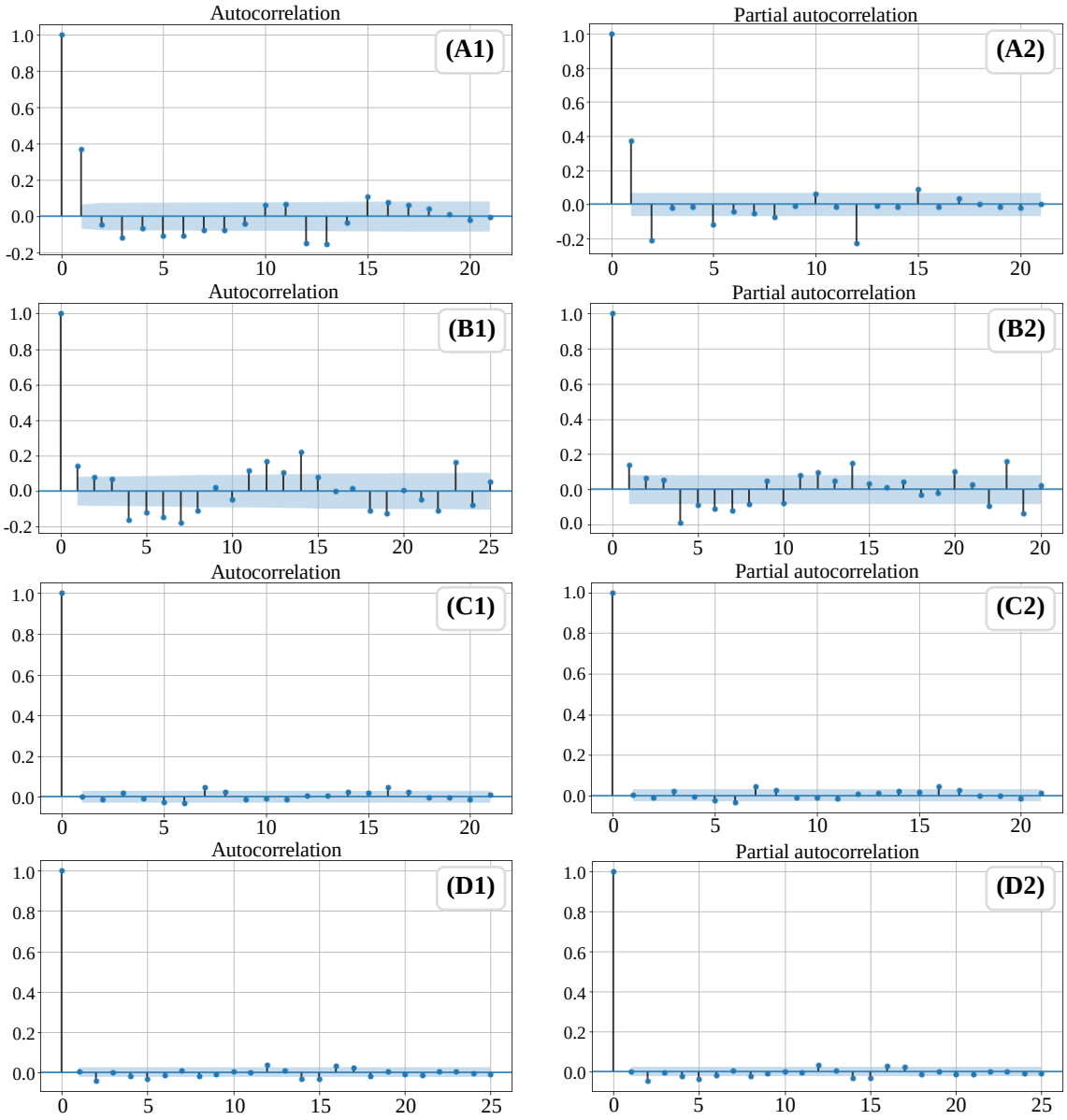


FIGURA F.16 – Correlograma dos resíduos de modelos ETS selecionados: **(A)** CPIAUCSL–$ETS(A,N)$; **(B)** VMT–$ETS(Ad,M)$; **(C)** PSI 20–$ETS(Ad,N)$; **(D)** SPY–$ETS(A,N)$

TABELA F.8 – Estudo da autocorrelação nos resíduos dos modelos ETS: Estatística Box-Pierce-Ljung

Série	**CPIAUCSL**	**VMT**	**PSI 20**	**SPY**
statistic	117.7643	11.6220	0.0070	0.0351
p-value	0.0000*	0.0000*	0.9334	0.8514

*Rejeita-se H_0 para os níveis de significância de 1%, 5% e 10%

F.7. MODELOS DNN: INFORMAÇÕES COMPLEMENTARES

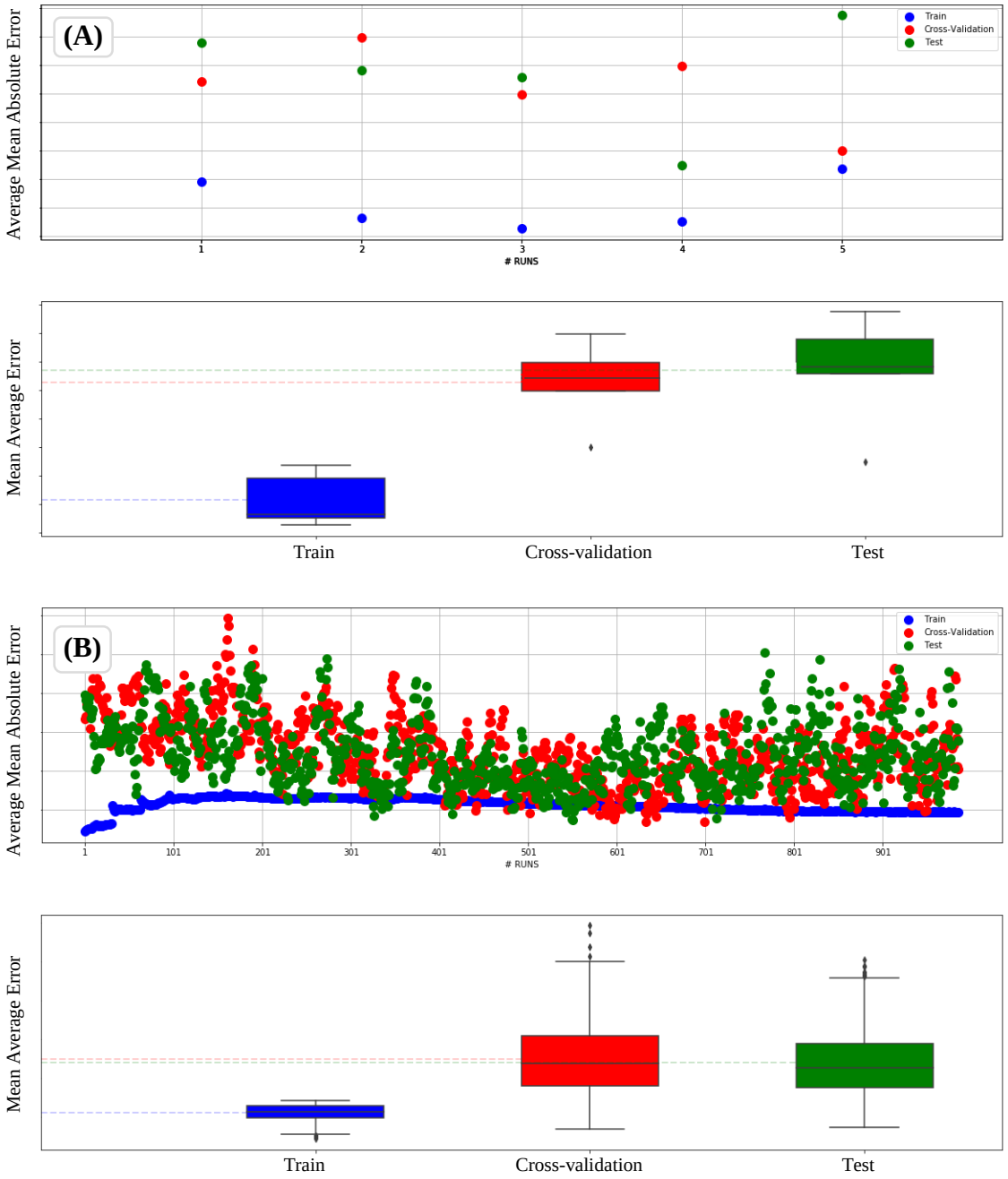


FIGURA F.17 – Comparação das distribuições dos erros de treino, cross-validation e teste (modelo DNN – MLP_Our para a série VMT): **(A)** 5 *runs*; **(B)** 1000 *runs*

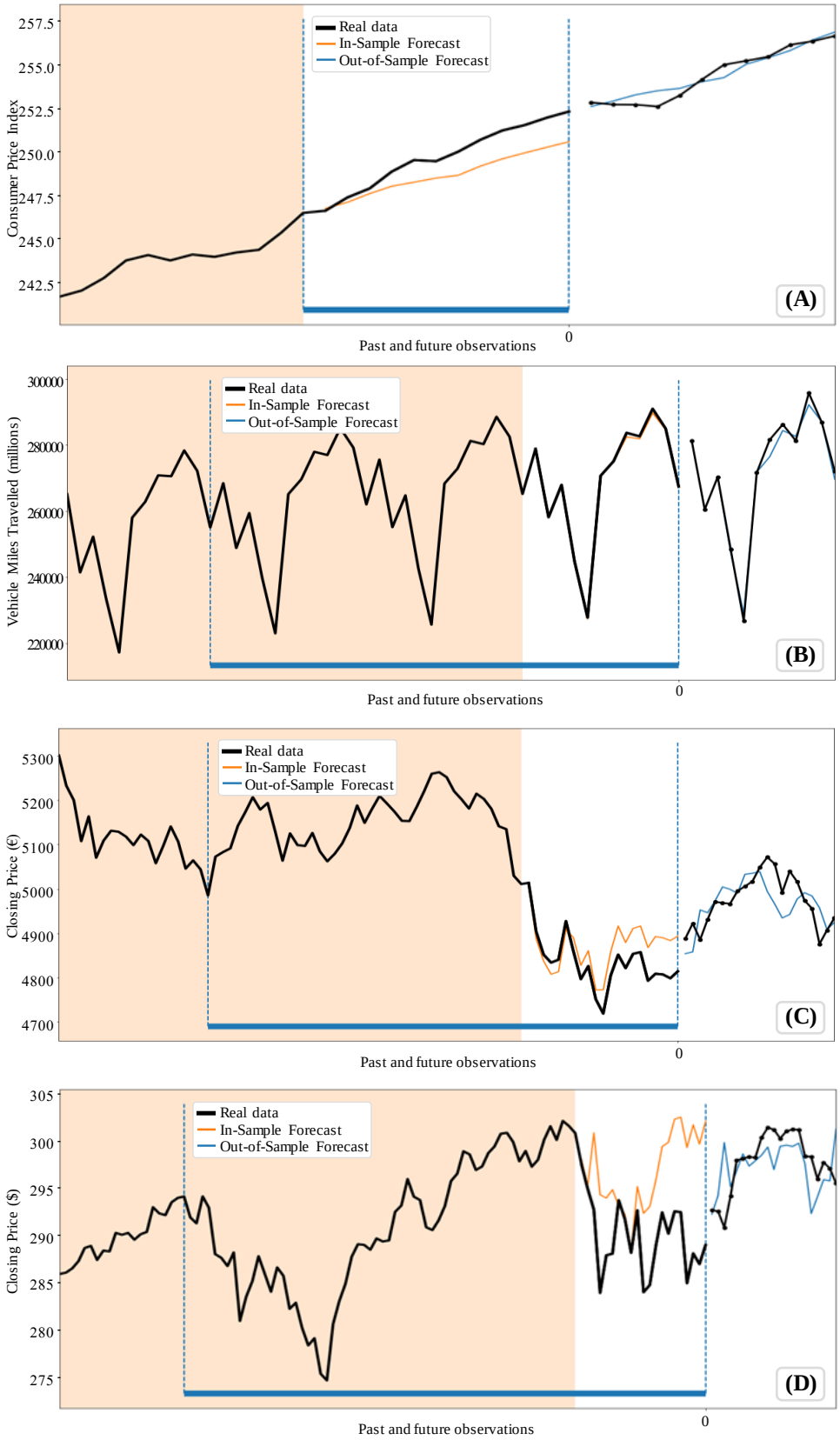


FIGURA F.18 –Modelos DNN – MLP (*fitting* e *forecasting*) das séries: **(A)** CPIAUCSL; **(B)** VMT; **(C)** PSI 20; **(D)** SPY