
**Modelos score-driven
não-gaussianos para séries
temporais com combinação não
linear das componentes de
tendência e sazonalidade**

Matheus Carneiro Nogueira

Relatório de Projeto Final 1 de Graduação

Centro Técnico Cienctífico - CTC
Departamento Engenharia Elétrica e
Departamento de Informática
Curso de Graduação em Engenharia de
Computação

Rio de Janeiro
Junho de 2023



Matheus Carneiro Nogueira

**Modelos score-driven não-gaussianos para
séries temporais com combinação não linear
das componentes de tendência e sazonalidade**

Relatório de Projeto Final 1 de Graduação

Relatório de Projeto Final 1 de Graduação, apresentado ao Curso de Graduação em Engenharia de Computação, do Departamento de Informática da PUC-Rio.

Orientador: Prof. Cristiano Fernandes

Rio de Janeiro
Junho de 2023

*Fazer previsões é muito difícil, especialmente
se forem sobre o futuro.*

Niels Bohr, .

Agradecimentos

Ainda a escrever

Resumo

Nogueira, Matheus; Fernandes, Cristiano. **Modelos score-driven não-gaussianos para séries temporais com combinação não linear das componentes de tendência e sazonalidade.** Rio de Janeiro, 2023. 76p. Projeto Final de Graduação – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Uma técnica comum em modelagem de séries temporais é a decomposição da série de interesse em componentes de tendência e sazonalidade. Dentro da classe de modelos Score-Driven, essa decomposição é usualmente realizada de forma aditiva, de tal modo que a série temporal de interesse é expressa como a componente de tendência somada à componente de sazonalidade. Entretanto, não é raro que, mesmo contabilizando a componente sazonal, os resíduos do modelo implementado ainda indiquem dependência sazonal não capturada pelo modelo. Dito isso, os principais objetivos desse projeto são (1) estudar se a combinação não linear das componentes de tendência e sazonalidade e (2) generalizar a variância da componente sazonal para capturar variâncias de diferentes períodos do ano, ambas com o intuito de capturar melhor a dependência sazonal da série.

Palavras-chave

Séries Temporais; Modelos Score Driven; Decomposição em Tendência e Sazonalidade.

Abstract

Nogueira, Matheus; Fernandes, Cristiano (Advisor). **Non-gaussian score driven models for time series with non-linear combination of trend and seasonality components.** Rio de Janeiro, 2023. 76p. Projeto Final de Graduação – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

A common technique in time series modeling is to decompose the time series into its trend and seasonal components. Inside the Score-Driven Models class, this decomposition is usually made in an additive form, so that the series is expressed as the sum of its trend and seasonal components. However, it is not unusual that, even with the seasonal component being considered into the model, its residuals still show signs of seasonal dependency that was not captured by the model. With that said, the main objectives of this project are (1) to study the nonlinear combination of trend and seasonality components, and (2) to generalize the variance of the seasonal component to capture variances of different periods of the year, both with the aim of better capturing the seasonal dependence of the time series.

Keywords

Time Series; Score Driven Models; Trend and Seasonality Decomposition.

Sumário

1	Introdução	13
2	Situação Atual	15
2.1	Implementação Computacional	15
3	Objetivos	30
4	Metodologia	32
4.1	Fundamentação Teórica	32
4.2	Implementação Computacional	37
5	Resultados	46
5.1	Série de Carga	46
5.2	Série de ENA	55
6	Referências bibliográficas	65
A	Desenvolvimento GAS	68
A.1	GAS gama	68
B	Sazonalidade em modelos de espaço de estados	70

Lista de figuras

Figura 2.1	Série temporal mensal de precipitação e seu histograma.	16
Figura 2.2	Tela inicial do TSL.	18
Figura 2.3	Telas de especificação do modelo e análise de resíduos.	18
Figura 2.4	Telas com o <i>fit in sample</i> do modelo GAS estimado.	19
Figura 2.5	Tela de previsão do modelo.	19
Figura 2.6	Especificação de modelo GAS estrutural no Time Series Lab State Space Edition.	21
Figura 2.7	<i>Fit in sample</i> do GAS estrutural estimado e gráfico das componentes estimadas.	21
Figura 2.8	Gráfico e FAC dos resíduos do modelo estimado.	22
Figura 2.9	Previsão 12 passos à frente do modelo estimado.	22
Figura 2.10	<i>Fit in sample</i> do modelo GAS para a série de precipitação utilizando ScoreDrivenModels.jl.	24
Figura 2.11	Resíduos e FAC dos resíduos do modelo GAS para a série de precipitação utilizando ScoreDrivenModels.jl.	24
Figura 2.12	Previsão pontual e simulações 12 passos à frente utilizando ScoreDrivenModels.jl.	24
Figura 2.13	Fit in Sample do modelo GAS(1,1) gama estimado no pacote GAS R.	26
Figura 2.14	Resíduos e FAC dos resíduos do modelo GAS(1,1) gama estimado no pacote GAS R.	26
Figura 2.15	Previsão do modelos GAS(1,1) estimado no pacote GAS R.	26
Figura 2.16	Fit in Sample do modelo GAS-CNO normal estimado no pacote UnobservedComponentsGAS.jl.	29
Figura 2.17	Resíduos e FAC dos resíduos do modelo GAS-CNO normal estimado no pacote UnobservedComponentsGAS.jl	29
Figura 2.18	Previsão do modelo GAS-CNO normal estimado no pacote UnobservedComponentsGAS.jl	29
Figura 4.1	Família de modelos ETS	36
Figura 4.2	Série Mensal de Carga	38
Figura 4.3	Série Mensal de ENA	39
Figura 4.4	Série Mensal de Vazão	39
Figura 5.1	Fit in Sample do modelo AutoARIMA para série de carga	46
Figura 5.2	Diagnósticos dos resíduos do modelo AutoARIMA para série de carga	47
Figura 5.3	Previsão 12 passos à frente do modelo AutoARIMA para série de carga	48
Figura 5.4	Fit in sample e previsão do modelo AutoARIMA para série de carga	48
Figura 5.5	Fit in Sample do modelo GAS-CNO LogNormal para série de Carga	49
Figura 5.6	Componentes estimadas do modelo GAS-CNO LogNormal para série de Carga	49

Figura 5.7 Diagnósticos dos resíduos do modelo GAS-CNO LogNormal para série de Carga	50
Figura 5.8 Previsão 12 passos à frente do modelo GAS-CNO LogNormal para série de Carga	51
Figura 5.9 Fit in sample e previsão do modelo GAS-CNO LogNormal para série de Carga	51
Figura 5.10 Fit in Sample do modelo GAS-CNO Gamma para série de Carga	52
Figura 5.11 Componentes estimadas do modelo GAS-CNO Gamma para série de Carga	52
Figura 5.12 Diagnósticos dos resíduos do modelo GAS-CNO Gamma para série de Carga	53
Figura 5.13 Previsão 12 passos à frente do modelo GAS-CNO Gamma para série de Carga	54
Figura 5.14 Fit in sample e previsão do modelo GAS-CNO Gamma para série de Carga	54
Figura 5.15 Fit in Sample do modelo AutoARIMA para série de ENA	55
Figura 5.16 Diagnósticos dos resíduos do modelo AutoARIMA para série de ENA	56
Figura 5.17 Previsão 12 passos à frente do modelo AutoARIMA para série de ENA	57
Figura 5.18 Fit in sample e previsão do modelo AutoARIMA para série de ENA	57
Figura 5.19 Fit in Sample do modelo GAS-CNO LogNormal para série de ENA	58
Figura 5.20 Componentes estimadas do modelo GAS-CNO LogNormal para série de ENA	58
Figura 5.21 Diagnósticos dos resíduos do modelo GAS-CNO LogNormal para série de ENA	59
Figura 5.22 Previsão 12 passos à frente do modelo GAS-CNO LogNormal para série de ENA	60
Figura 5.23 Fit in sample e forecast do modelo GAS-CNO LogNormal para série de ENA	60
Figura 5.24 Fit in Sample do modelo GAS-CNO Gamma para série de ENA	61
Figura 5.25 Componentes estimadas do modelo GAS-CNO Gamma para série de ENA	61
Figura 5.26 Diagnósticos dos resíduos do modelo GAS-CNO Gamma para série de ENA	62
Figura 5.27 Previsão 12 passos à frente do modelo GAS-CNO Gamma para série de ENA	63
Figura 5.28 Fit in sample e previsão do modelo GAS-CNO Gamma para série de ENA	63

Lista de tabelas

Tabela 4.1	Divisão das séries temporais em treino e teste	40
Tabela 5.1	MAPEs de treino e teste do modelo AutoARIMA para série de carga	48
Tabela 5.2	Testes de hipótese para resíduos do modelo GAS-CNO Log-Normal para série de Carga	50
Tabela 5.3	MAPEs de treino e teste do modelo GAS-CNO LogNormal para série de Carga	52
Tabela 5.4	Testes de hipótese para resíduos do modelo GAS-CNO Gamma para série de Carga	53
Tabela 5.5	MAPEs de treino e teste do modelo GAS-CNO LogNormal para série de Carga	54
Tabela 5.6	MAPEs de treino e teste para série de carga	55
Tabela 5.7	MAPEs de treino e teste do modelo AutoARIMA para série de ENA	57
Tabela 5.8	Testes de hipótese para resíduos do modelo GAS-CNO Log-Normal para série de ENA	59
Tabela 5.9	MAPEs de treino e teste do modelo GAS-CNO LogNormal para série de ENA	60
Tabela 5.10	Testes de hipótese para resíduos do modelo GAS-CNO Gamma para série de ENA	62
Tabela 5.11	MAPEs de treino e teste do modelo GAS-CNO Gamma para série de ENA	63
Tabela 5.12	MAPEs de treino e teste para série de ENA	64

Lista de algoritmos

Lista de Códigos

Código 1	Script ScoreDrivenModels.jl	23
Código 2	Script GAS package R	25
Código 3	Script UnobservedComponentsGAS.jl	28

Lista de Abreviaturas e Notação

ARMA – *Auto Regressive Moving Average*

CNO – Componentes não observáveis

FAC – Função de Autocorrelação

FACP – Função de Autocorrelação Parcial

MV - Máxima Verossimilhança

TSL – *Time Series Lab*

OD – *Observation driven models*

PD – *Parameter driven models*

y ou \vec{y} ou \underline{y} – Todos esses símbolos representam vetores

y – Letras minúsculas representam números escalares

1

Introdução

introdução

Modelos *score driven* são uma classe geral e flexível de modelos de série temporal desenvolvida simultaneamente por [Harvey 2013] e [Creal, Koopman e Lucas 2013]. Essa classe é reconhecida por diferentes nomenclaturas: *generalized autoregressive score (GAS)*, *dynamic conditional score models*, ou, simplesmente, modelos score-driven. Eles introduzem diversas vantagens de modelagem de séries temporais, descritas a seguir.

Primeiramente, essa classe de modelo traz a flexibilidade de modelar séries temporais não-gaussianas, isto é, séries cuja distribuição de probabilidade condicional não é a distribuição normal. Essa vantagem torna o modelo GAS uma poderosa ferramenta de modelagem, uma vez que há diversas situações nas quais a distribuição normal é, por definição, inadequada. Como exemplo, tome as séries de velocidade de vento, que assumem apenas valores positivos, fazendo com que a distribuição de probabilidade associada ao problema deva ser definida apenas para os reais positivos. As séries de retornos financeiros, por sua vez, possuem como fato estilizado excesso de curtose em relação à distribuição normal, tornando-a também inadequada. Além disso, séries temporais associadas à contagem, como por exemplo, quantidade de itens vendidos em uma determinada loja, necessitam de distribuições discretas. Sendo assim, a inexistência da restrição de normalidade da série a ser modelada torna os modelos GAS extremamente úteis em diversas aplicações.

para variáveis
elegíveis

Além dessa flexibilidade, os modelos GAS também permitem estimação de parâmetros variantes no tempo com diferentes processos de atualização. Isso os torna capazes de generalizar diferentes modelos clássicos de séries temporais, como, por exemplo, os modelos GARCH [Bollerslev 1986] e modelos de duração condicional [Engle e Russell 1998].

Dada a relevância dos modelos score-driven e a vasta aplicabilidade desse arcabouço de modelagem, diversas ferramentas computacionais vêm sendo desenvolvidas a fim de implementar essa classe de modelos. Podemos citar, por exemplo, o pacote *GAS* desenvolvido em R [Ardia, Boudt e Catania 2016], o pacote *PyFlux*, implementado em *Python* [Taylor 2016] e o pacote *ScoreDrivenModels* implementado em Julia [Bodin et al. 2020]. Somado a isso, existe o software livre *Time Series Lab* no qual também está implementada a classe de modelos GAS em sua versão *Time Series Lab - Score Edition* [Lit, Koopman e Harvey 2021].

Como resposta ao dinamismo com o qual os modelos GAS são estu-

dados, foi criado um repositório dedicado exclusivamente a artigos e implementação de código sobre essa classe de modelos, que pode ser acessado em <http://www.gasmmodel.com/> [Andre e Koopman].

Dito isso, para fins de organização, este relatório está estruturado da seguinte forma. No Capítulo 2 está apresentada a situação atual de modelos score-drive, tanto em termos de desenvolvimento teórico quanto aplicações práticas e desenvolvimento de ferramentas computacionais. No Capítulo 3 são apresentados os principais objetivos a serem alcançados com esse trabalho. O Capítulo 4 discorre sobre o desenvolvimento teórico realizado no projeto. Por fim, o Capítulo ?? compara o planejamento inicial do projeto com seu andamento real.

2

Situação Atual

Dada a flexibilidade trazida pela modelagem *score-driven* para séries temporais não Gaussianas, esses modelos estão sendo utilizados nas mais diversas áreas de aplicação, como finanças, varejo, energia e clima.

No contexto de finanças, eles têm se mostrado especialmente úteis para modelar a volatilidade de ativos financeiros. Por exemplo, um estudo recente de [Xu e Lien 2022] explorou o uso de modelos GAS na previsão da volatilidade de ativos de óleo e gás, comparando a qualidade de previsão com os modelos GARCH e EGARCH e constatando que, para os ativos de óleo, a modelagem GAS se mostrou superior em todos os cenários enquanto que, para ativos de gás natural, mostrou-se superior para previsões de 1 e 5 passos à frente, considerando séries diárias. Além disso, [Fuentes, Herrera e Clements 2023] utilizaram modelos *score-driven* para previsão de eventos extremos em séries de retornos financeiros, o que, dada a flexibilidade de adicionar medidas de volatilidade realizada, melhorou a acurácia da previsão quando comparada com outros modelos de *benchmark*.

Embora uma grande parte das publicações relacionadas a modelos *score driven* sejam dentro do contexto financeiro, há outras áreas que também estão utilizando essa classe de modelos de séries temporais. No varejo, [Hoeltgebaum et al. 2021] utilizaram modelos GAS com distribuição log-normal para realizar previsões de demanda diária em centros de distribuição, constatando que essa metodologia é competitiva quando comparada com *benchmarks* usuais, além de trazer a vantagem de produzir uma densidade preditiva de forma fechada.

No contexto de clima e energia, modelos *score driven* também são largamente utilizados. Por exemplo, [Blazsek e Escribano 2023] utilizaram modelos GAS para modelar e medir efeitos climáticos não lineares de longo prazo resultados de alterações em variáveis climáticas como volume de gelo global, nível de dióxido de carbono (CO₂) na atmosfera e temperatura da superfície da Antártida. Pesquisar mais trabalhos em energia e clima

2.1

Implementação Computacional

Assim como foi comentado no Capítulo 1, já existem algumas ferramentas computacionais que implementam os modelos *score-driven* para séries temporais, sejam elas pacotes das principais linguagens utilizadas para ciência de da-

Lisso
Previsão
Há
medio
de retorno
cenário
volatilidade
já
informações

dos e estatística, ou *software* específicos para modelagens de séries temporais. Nessa seção, estão apresentadas com maior profundidade essas ferramentas com o intuito de entender suas implementações, capacidades e limitações.

Somado a isso, nas seções seguintes será utilizada uma série temporal mensal de precipitação retirada do site do Centro de Previsão de Tempo e Estudos Climáticos <https://clima.cptec.inpe.br/> para exemplificar o funcionamento das ferramentas explicadas. A Figura 2.1 exibe a série e o seu histograma. Note que modelar essa série a partir de uma distribuição normal não seria adequado devido a assimetria positiva do histograma.

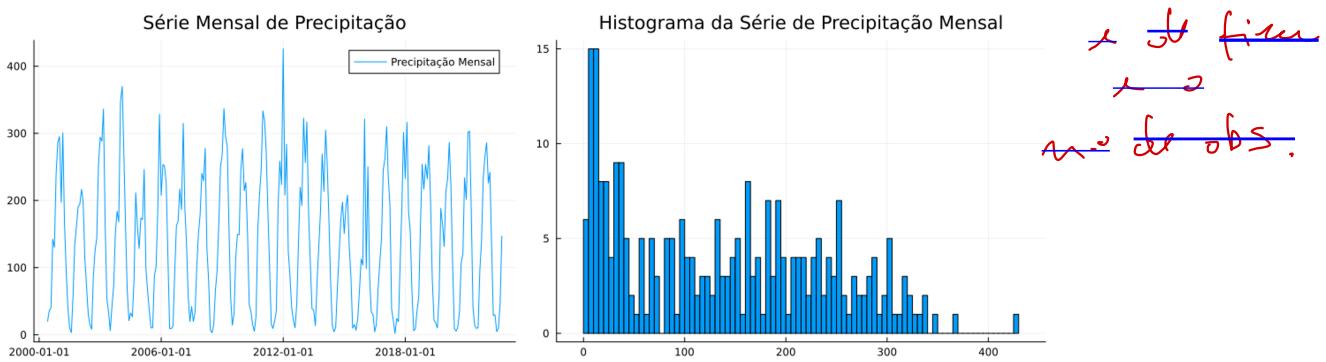


Figura 2.1: Série temporal mensal de precipitação e seu histograma.

2.1.1

TimeSeriesLab Score Edition

O TimeSeriesLab (TSL) é uma plataforma para análise, modelagem e previsão de séries temporais de forma interativa e repleta de suporte visual [Lit]. Essa ferramenta possui uma grande seleção de modelos de séries temporais, como modelos ARIMA de Box-Jenkins, modelos de suavização exponencial, modelos *score-driven* e modelos estruturais básicos. Sendo assim, tudo que o usuário precisa fazer para realizar a sua modelagem é escolher o tipo de modelo que melhor se adequa aos seus dados.

Além disso, o TSL também permite que o usuário crie seu próprio modelo de séries temporais com base em componentes dinâmicas como tendência, sazonalidade e ciclos.

Como o foco desse projeto é a modelagem *score-driven*, essa seção se limita a comentar sobre esse tipo de modelo e não traz um extenso resumo do funcionamento da ferramenta. Para fins de aprofundamento no TSL, recomenda-se a leitura do manual disponível no site da ferramenta <https://timeserieslab.com/> [Lit].

O modelo *score driven* implementado no TSL *Score Edition* é o GAS(p,q) que, assim como será visto no Capítulo 4, utiliza a formulação de um modelo

ARMA(p,q) para a função de atualização dos seus parâmetros variantes no tempo. Esse *software* possui uma segunda versão denominada *Time Series Lab State Space Edition* que mistura o GAS(p,q) com modelos estruturais. Essa versão está apresentada na seção 2.1.2. Caso o usuário queira que o TSL defina automaticamente os valores dos hiperparâmetros p e q, a ferramenta é capaz de definir esses valores com base no algoritmo de Hyndman-Khandakar, que utiliza o AIC como critério de escolha [Hyndman e Khandakar 2008].

Uma das principais qualidades do TSL é a sua usabilidade. Com interface simples e intuitiva, um usuário pode, rapidamente, estimar e avaliar diversos modelos de séries temporais sem a necessidade de conhecimento prévio sobre a teoria por detrás dos modelos ou o conhecimento de alguma linguagem de programação.

Por outro lado, uma vez sendo um *software* pronto, o TSL não possui a mesma flexibilidade de pacotes implementados para linguagens de programação como os outros que serão aqui apresentados.

A Figura 2.2 exibe a tela inicial do TSL, na qual o usuário pode visualizar a série temporal a ser modelada e realizar pré-processamentos como transformações na escala da série. Em seguida, a Figura 2.3 exibe as telas de especificação do modelo, onde o usuário define a distribuição desejada e as características dos parâmetros do modelo, e de análise dos resíduos, onde é possível ver a série de resíduos do modelo estimado, bem como duas FAC e outras visualizações importantes. Junto da análise dos resíduos, é importante visualizar o *fit in sample* do modelo, que está exibido na Figura 2.4. Por fim, a Figura 2.5 exibe a tela de previsão do modelo estimado.

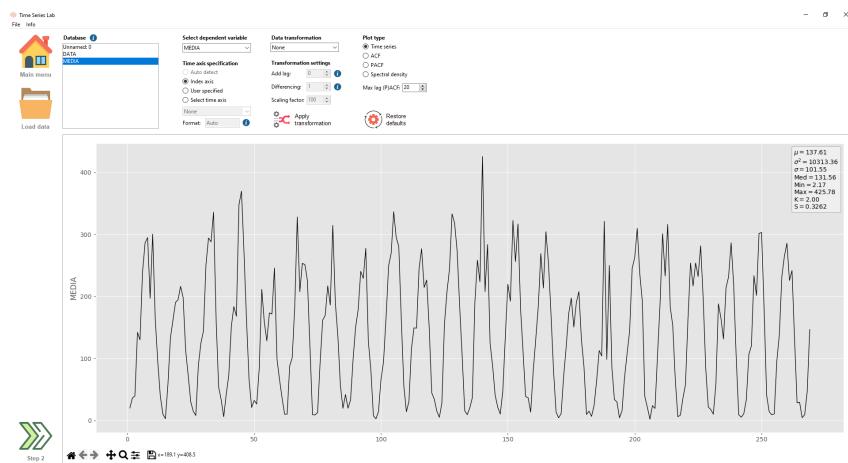


Figura 2.2: Tela inicial do TSL.



Figura 2.3: Telas de especificação do modelo e análise de resíduos.

Fazendo no TSL Score Edition, com
TSL prefer space, listam a lista.
contínuos e discos que estar disponibili-
zados.

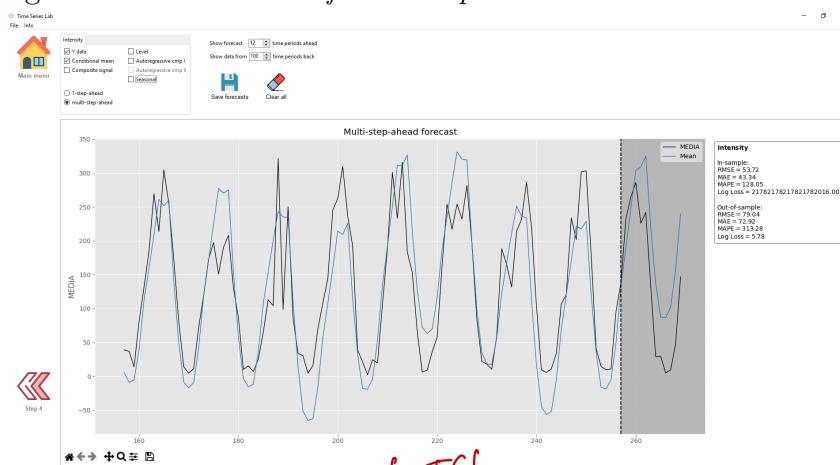
Figura 2.4: Telas com o *fit in sample* do modelo GAS estimado.

Figura 2.5: Tela de previsão do modelo.

2.1.2

TimeSeriesLab - State Space Edition

Além do *software* apresentado na seção anterior, a *TimeSeriesLab* também possui um segundo *software* que implementa modelos *score-driven*, o *Time Series Lab State Space Edition*. Como o nome sugere, o foco desse *software* é a modelagem de séries temporais a partir do arcabouço dos modelos de espaço de estado.

Nessa versão, o usuário é capaz tanto de utilizar modelos pré construídos quando criar seu próprio modelo, utilizando como base as componentes clássicas de um modelo estrutural: tendência, nível, sazonalidade e ciclos. Dentre os modelos pré construídos, estão:

- Métodos básicos: amortecimento exponencial simples (*EWMA*), *Holt Winters* e *Holt Winters Sazonal*;
- Modelos Estruturais: nível local (com tendência e com tendência e sazonalidade) e modelo estrutural básico;

- Modelos ARIMA;
- Modelos *score-driven* GAS(p,q);
- Modelos de Machine Learning: *XGBoost Regressor*.

Note que, dentro desse *software*, há modelos *score-driven* GAS(p,q). Dito isso, a utilização dessa versão é mais recomendada dado sua maior amplitude de modelagem, englobando, inclusive, a versão *TimeSeriesLab - Score Edition*.

Além desses modelos já implementados, o usuário também pode especificar seu próprio modelo a partir das componentes estruturais clássicas. Somado a elas, é possível adicionar componentes ARMA(p,q) e variáveis explicativas.

Por fim, a versão *State Space Edition* permite construir modelos *score-driven* estruturais, que combinam o modelo GAS com componentes de tendência e sazonalidade (ver seção 4.1.1). Essa é, justamente, a classe de modelos a ser focada neste projeto. A Figura 2.6 exibe a tela na qual o usuário especifica o modelo GAS estrutural desejado. Em seguida, a Figura 2.7 exibe o modelo estimado, com seu *fit in sample* e o gráfico das componentes não observáveis estimadas.

→ não possui o nível de detalhe
de implementação dos modelos
Score driven é composta por a
TSL Score Edition.

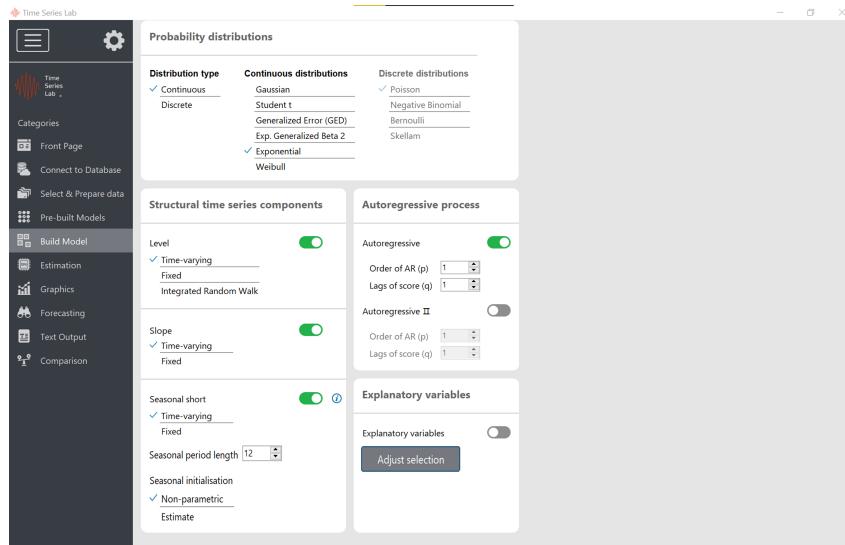


Figura 2.6: Especificação de modelo GAS estrutural no Time Series Lab State Space Edition.

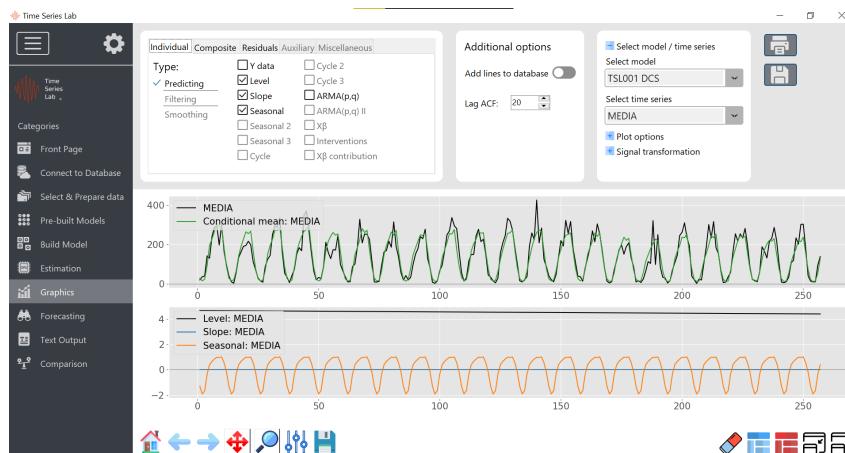


Figura 2.7: *Fit in sample* do GAS estrutural estimado e gráfico das componentes estimadas.

Como é usual, o usuário pode estudar os resíduos do modelo estimado, tanto com seu gráfico quanto com sua FAC, para avaliar a qualidade da estimação e, em seguida, seguir com a previsão *out of sample* do modelo. As Figuras 2.8 e 2.9 ilustram as janelas nas quais o usuário realiza essas operações.

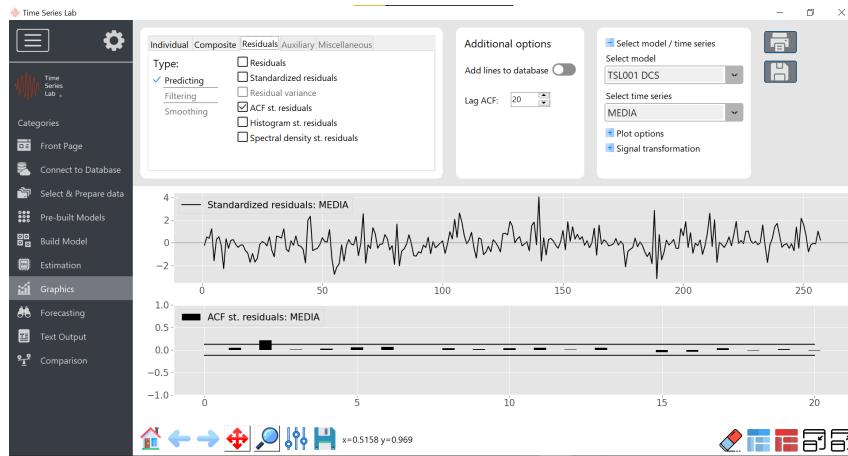


Figura 2.8: Gráfico e FAC dos resíduos do modelo estimado.

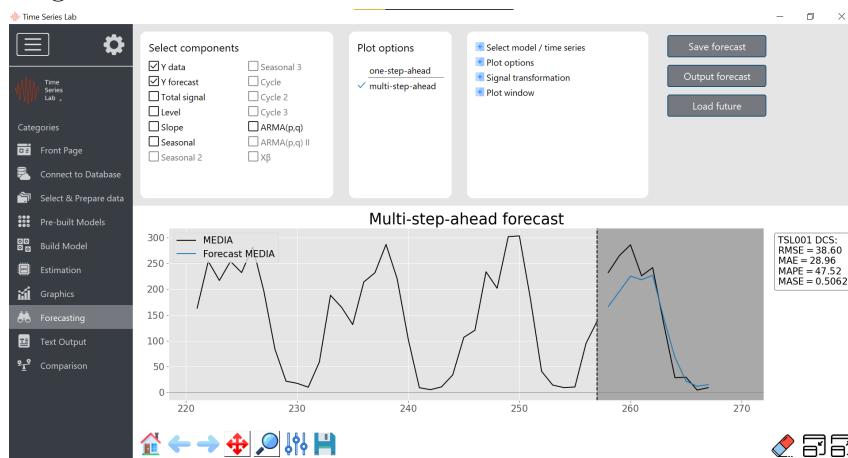


Figura 2.9: Previsão 12 passos à frente do modelo estimado.

Certamente, dentre os *softwares* aqui apresentados, o *TimeSeriesLab StateSpace Edition* é aquele que melhor combina flexibilidade e usabilidade, fornecendo uma grande gama de modelos de séries temporais com uma interface simples e intuitiva, tornando-o de mais fácil acesso que pacotes implementados em linguagens de programação. Somado a isso, uma grande qualidade do *software*, especialmente para esse projeto, é a implementação de modelos *score-driven* tanto GAS(p,q) quanto estruturais. Embora muito completo, há duas limitações no TSL: as componentes dos modelos estruturais são combinadas apenas aditivamente e a variância da componente sazonal é comum para todos os períodos sazonais. O principal objetivo deste projeto é, justamente, implementar modelos *score-driven* estruturais que supram essas duas limitações. Sobre isso, ver seções 4.1.2 e 4.1.3.

2.1.3

ScoreDrivenModels.jl - Julia

ScoreDrivenModels.jl é um pacote desenvolvido na linguagem Julia para modelar, prever e simular séries temporais a partir de modelos *score-driven* [Bodin et al. 2020]. A API do pacote permite o usuário escolher entre diversas distribuições, estruturas de lag, métodos de otimização e valores de escala. Além disso, o pacote também traz a possibilidade do usuário adicionar novas distribuições de probabilidade caso sejam necessárias.

O que é isso?

Função:

Para especificar um modelo, o usuário necessita de 4 informações: a distribuição desejada, a escala, a estrutura de *lag* (os valores de *p* e *q*) e especificar quais parâmetros serão variantes no tempo. Essas informações são passadas como parâmetros da função *ScoreDrivenModel()*. Em seguida, a função *fit!()* deve ser usada para estimar o modelo e é por meio dela que o usuário indica a série temporal a ser modelada e qual o otimizador a ser utilizado. Por fim, uma vez identificado que o modelo estimado é satisfatório, o usuário pode realizar a previsão de sua série temporal por meio da função *forecast()* que recebe como parâmetros a série temporal usada para a estimação, o modelo estimado e a quantidade de passos à frente. O código 2.1.3 abaixo apresenta um script simples para a realização das etapas descritas.

Código 1: Script ScoreDrivenModels.jl

```

1 using ScoreDrivenModels
2
3 "Código para definição e estimação do modelo GAS(1,1) LogNormal"
4 gas = ScoreDrivenModel([1, 12], [1, 12], Gamma, 0.0) #definição do
    modelo
5 ScoreDrivenModels.fit!(gas, y_fit) #estimação do modelo
6 fit_in_sample = fitted_mean(gas, y_fit) #obtenção do fit in sample
7 residuals = y_fit .- fit_in_sample #obtenção dos resíduos
8 forecast = ScoreDrivenModels.forecast(y_fit, gas, 12) #obtenção das
    previsões

```

As Figuras 2.10 e 2.11 exibem o *fit in sample* e os resíduos do modelo estimado com sua FAC, enquanto que a Figura 2.12 exibe a previsão pontual 12 passos à frente e a simulação. Todos os resultados foram obtidos a partir da execução do script 2.1.3.

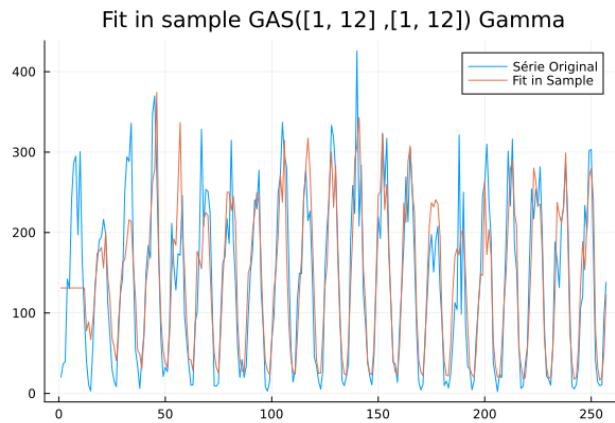


Figura 2.10: *Fit in sample* do modelo GAS para a série de precipitação utilizando ScoreDrivenModels.jl.

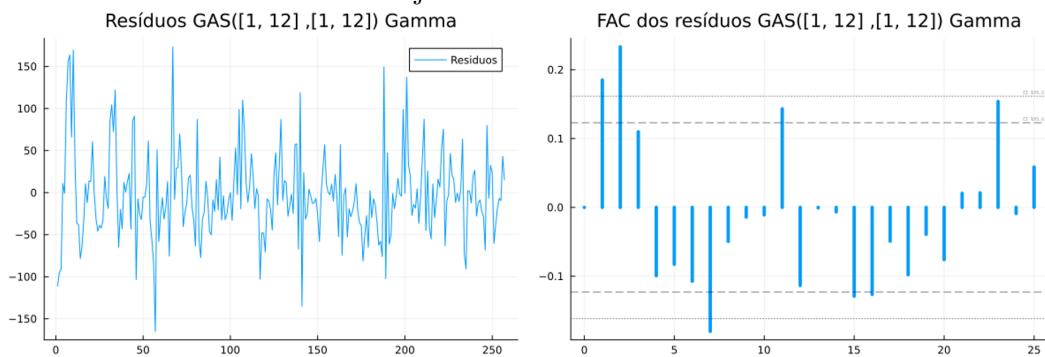


Figura 2.11: Resíduos e FAC dos resíduos do modelo GAS para a série de precipitação utilizando ScoreDrivenModels.jl.

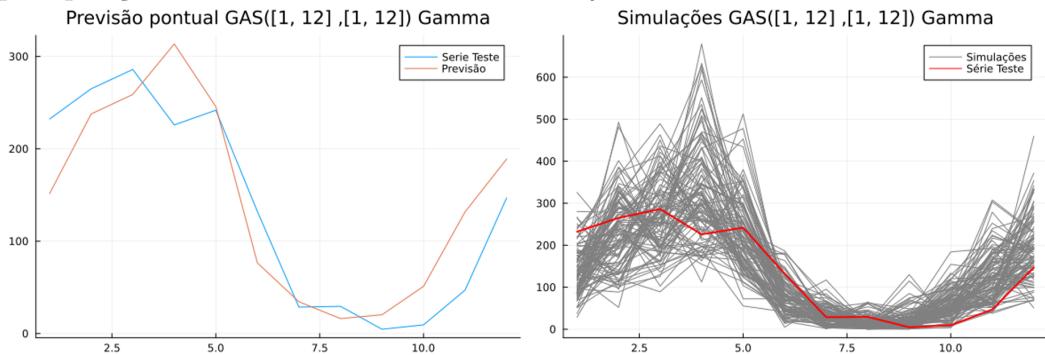


Figura 2.12: Previsão pontual e simulações 12 passos à frente utilizando ScoreDrivenModels.jl.

Algumas das principais qualidades desse pacote é sua integração com a linguagem Julia, que vem sendo cada vez mais utilizada para modelagem de problemas que exigem ferramentas de estatística, otimização, ciência de dados e áreas correlatas para a sua solução. Somado a isso, o pacote traz grande flexibilidade de modelagem, uma vez que permite ao usuário adicionar qualquer distribuição desejada. Por fim, sua sintaxe é simples e muito similar à diversos outros pacotes.

Uma desvantagem, no entanto, é o fato desse pacote implementar apenas

a dinâmica GAS(p,q) para modelos *score-driven*, não sendo possível optar pela dinâmica via componentes não observáveis.. Essa limitação impossibilita modelar a série temporal a partir de suas componentes de tendência e sazonalidade, algo que será fundamental para este projeto.

2.1.4

GAS Package - R

O pacote GAS, desenvolvido por [Ardia, Boudt e Catania 2016] na linguagem R foi o primeiro a ser implementado e, nesse sentido, foi utilizado também como referência para os pacotes apresentados. Ele permite ao usuário realizar simulação, estimação e previsão de modelos GAS(p,q) tanto univariados quanto multivariados, o que é uma grande qualidade. No entanto, o pacote apresenta limitações para modelos GAS multivariados com mais de 4 séries temporais, uma vez que ele não reporta a atualização exata dos parâmetros de atualização de correlação nesses casos. Para entender melhor essa limitação, ver [Ardia, Boudt e Catania 2016].

A funcionalidade desse pacote é muito similar aos demais apresentados, com funções usuais de um pacote de modelagem de séries temporais. O usuário possui a capacidade de especificar o modelo GAS a ser implementado. Para isso, é necessário definir a distribuição de probabilidade, o tipo de parametrização (identidade, inversa ou raiz quadrada da inversa) e quais parâmetros da distribuição definida serão variantes no tempo. A partir do modelo especificado, existem funções próprias para estimar o modelo, obter os resíduos do modelo e realizar a previsão do modelo. O script abaixo traz um exemplo de código R para realizar essas instruções. Em seguida, as Figuras 2.13 e 2.14 exibem o *fit in sample* e os resíduos do modelo com sua FAC, e a Figura 2.15 exibe a previsão 12 passos à frente.

Código 2: Script GAS package R

```

1 library(GAS)
2
3 # Especificação do modelo
4 GASSpec = UniGASSpec(Dist = "gamma", ScalingType = "Identity",
5                         GASPar = list(location = TRUE, scale = TRUE))
6 Fit      = UniGASFit(GASSpec, y) # estima o modelo
7 residuals = residuals(Fit) # obtém os resíduos do modelo estimado
8 forecast  = UniGASFor(Fit, 12) # gera a previsão 12 passos à frente

```

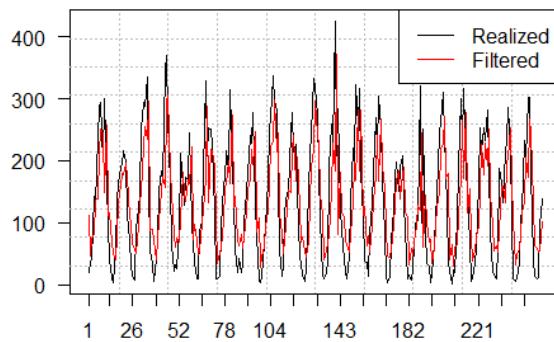


Figura 2.13: Fit in Sample do modelo GAS(1,1) gama estimado no pacote GAS R.

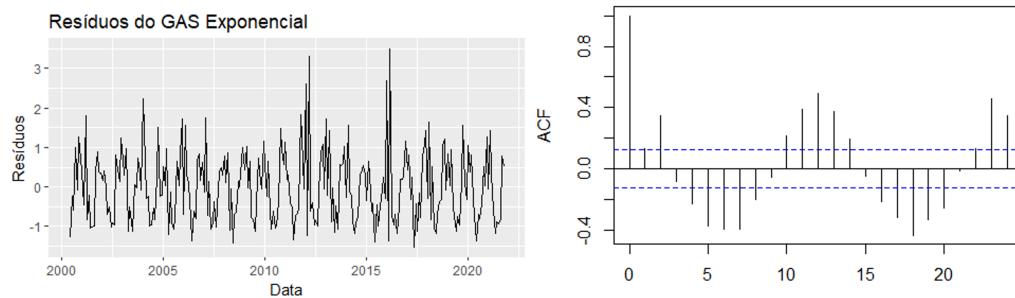


Figura 2.14: Resíduos e FAC dos resíduos do modelo GAS(1,1) gama estimado no pacote GAS R.

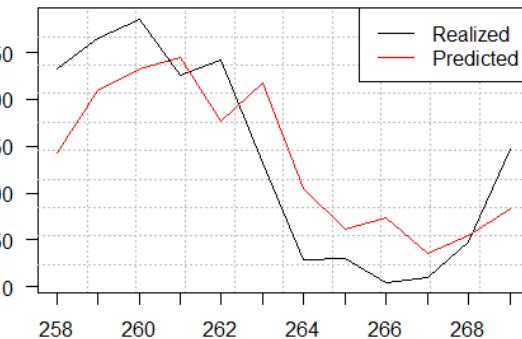


Figura 2.15: Previsão do modelos GAS(1,1) estimado no pacote GAS R.

Tal qual os outros pacotes, é possível escolher uma dentre diversas distribuições de probabilidade, o que é fundamental em termos de implementação para que o usuário possa desfrutar da flexibilidade de modelagem que a teoria dos modelos fornece. Ademais, ele também permite a escolha de diferentes métodos de otimização para a estimação do modelo.

Um diferencial deste pacote é a existência de funções prontas para realizar previsão de vários passos à frente a partir de uma janela rolante de reestimação do modelo após uma previsão de um passo à frente. Além disso, também estão implementadas funções específicas para realização de *backtests* do modelo estimado e para simulações de séries a partir de um processo GAS.

Poderíamos citar apenas duas limitações do pacote. Primeiramente, ele

GAS(1,1)?

implementa apenas a dinâmica GAS(p,q) (ver Capítulo 4), tal qual o pacote *ScoreDrivenModels.jl*. Somado a isso, o pacote está limitado a modelos com estrutura de lag (1,1), isto é, apenas implementa o GAS(1,1). Essa é uma grande desvantagem do pacote pois limita muito a modelagem das séries temporais sazonais, visto que utilizar mais *lags* da série captura melhor a estrutura de dependência linear e, inclusive, a inclusão de *lags* sazonais pode melhorar consideravelmente a estimativa dos modelos. Vale comentar que essa limitação é notada no exemplo dessa seção, uma vez que a FAC dos resíduos exibida na Figura 2.14 indica que o modelo não capturou a estrutura sazonal da série.

2.1.5

UnobservedComponentsGAS.jl - Julia

Por último, o pacote *UnobservedComponentsGAS.jl*, que está sendo desenvolvido como uma dissertação de mestrado de [Alves], no laboratório LAMPS PUC-Rio, implementa modelos GAS cuja dinâmica dos parâmetros variantes no tempo segue um modelo de componentes não observáveis.

O grande diferencial deste pacote é o fato de sua implementação ter sido inteiramente desenvolvida a partir da modelagem JuMP, que é um poderoso framework de otimização dentro da linguagem Julia. Uma das vantagens dessa implementação é a possibilidade de tratar as restrições de parâmetros positivos das distribuições não com o ferramental de funções de ligação, mas como simples restrições do problema de otimização. Isso traz simplicidade e flexibilidade para a implementação.

Até o momento, as distribuições implementadas são a distribuição $Normal(\mu, \sigma^2)$ e a distribuição $tLocationScale(\mu, \sigma^2, \nu)$, de média μ , variância σ^2 e ν graus de liberdade.

As componentes que podem ser adicionadas à dinâmica dos parâmetros variantes no tempo até o momento são:

- Random ~~Walk~~
- Random ~~Walk~~ with ~~Slope~~
- AR(p)
- Sazonalidade por funções trigonométricas

Além disso, o pacote também fornece ao usuário a opção de estimar um *auto-gas*, de forma a deixar para o otimizador escolher os melhores valores do hiperparâmetro d , da matriz de informação de Fisher e do hiperparâmetro α , que é uma penalização da otimização.

O script abaixo exemplifica a utilização deste pacote para estimar um modelo GAS-CNO Gaussiano, com média variante no tempo e variância fixa. Note que o modelo a ser estimado é um modelo cuja média segue uma dinâmica com *random walk with slope* e sazonalidade com período sazonal igual a 12. Além disso, o usuário pode utilizar a função *fit*, caso deseje especificar os valores de d e α , ou a função *auto_gas*, caso queira que o otimizador escolha os melhores valores para esses hiperparâmetros.

Código 3: Script UnobservedComponentsGAS.jl

```

1 using UnobservedComponentsGAS
2
3 "Código para definição e estimação do modelo GAS Normal"
4 dist = UnobservedComponentsGAS.NormalDistribution(missing, missing)
5 time_varying_params = [true, false]
6 random_walk = Dict(1=>false, 2=>false)
7 random_walk_slope = Dict(1=>true, 2=>false)
8 ar = Dict(1=>false, 2=>false)
9 seasonality = Dict(1=>12)
10 robust = false
11 stochastic = false
12 d = 1.0
13 num_scenarios = 500
14
15 gas_model = UnobservedComponentsGAS.GASModel(dist,
        time_varying_params, d, random_walk, random_walk_slope, ar,
        seasonality, robust, stochastic)
16
17 fitted_model = UnobservedComponentsGAS.fit(gas_model, y_fit)
18 fitted_model = UnobservedComponentsGAS.auto_gas(gas_model, y_fit,
        steps_ahead)
19
20 residuals = fitted_model.residuals
21 forecast = UnobservedComponentsGAS.predict(gas_model, fitted_model,
        y_fit, steps_ahead, num_scenarios)

```

As figuras a seguir exibem os mesmos gráficos exibidos para os demais pacotes: a figura 2.16 exibe o *fit in sample* do modelo para a série de vazão, a figura 2.17 exibe os resíduos padronizados e a FAC desses resíduos e, por fim, a figura 2.18 exibe a previsão 12 passos à frente deste modelo, em comparação com os valores reais da série temporal.

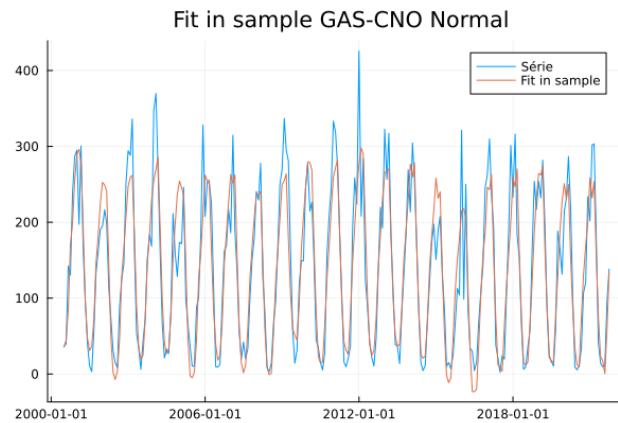


Figura 2.16: Fit in Sample do modelo GAS-CNO normal estimado no pacote UnobservedComponentsGAS.jl.

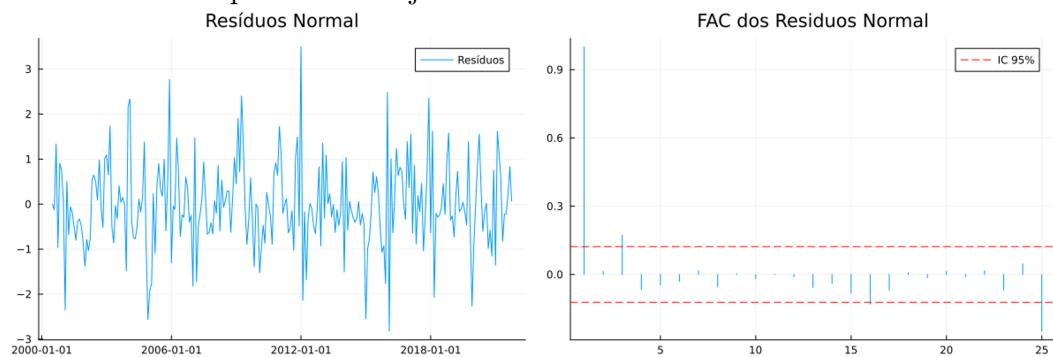


Figura 2.17: Resíduos e FAC dos resíduos do modelo GAS-CNO normal estimado no pacote UnobservedComponentsGAS.jl

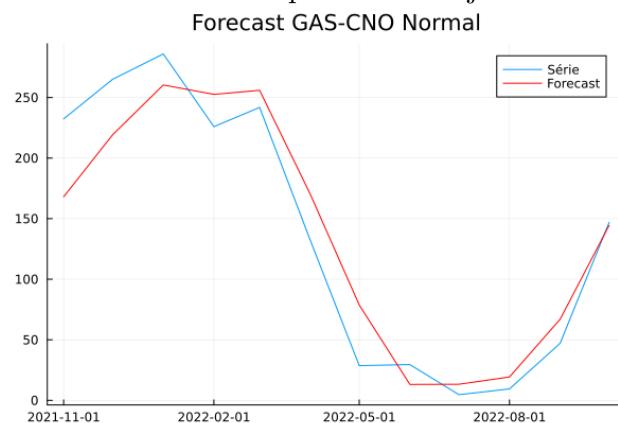


Figura 2.18: Previsão do modelo GAS-CNO normal estimado no pacote UnobservedComponentsGAS.jl

Fazendo isso agora com uma tabela comparativa
entre os softwares / linguagens de tipos

Linguagem / Pacote	Distr. Lispus novo	GAS (pg)	Comparado
TSL Score			
TSL State Space			
GAS R			
Julia			
UC Suite			

etc etc

Entendido? Olhando essa tabela
fazemos uma brev^a review sobre
as poss. de cada linguagem.

3 Objetivos

O foco principal desse projeto são os modelos score-driven para séries temporais. Dentro deste escopo, a proposta consiste em aprofundar os estudos em modelos GAS cuja distribuição de probabilidade não é Gaussiana, e cuja dinâmica dos parâmetros variantes no tempo segue um modelo de componentes não observáveis (CNO), com o intuito de modelar séries climáticas com restrições em seu domínio. Dois exemplos clássicos são séries de vazão de rio, que assumem apenas valores positivos, e séries de velocidade de vento que, além de também não assumirem valores negativos, podem apresentar muitos zeros com o aumento da frequência da série (diária ou horária). ?!

A partir desses modelos, um dos objetivos é estudar a decomposição das séries em componentes de tendência e sazonalidade, com parâmetros variantes no tempo e com combinações não lineares dessas componentes. A principal decomposição, gerada a partir do modelo de espaço de estados [Harvey 2013] e [Caivano, Harvey e Luati 2016], na qual a série é decomposta como a soma de uma componente de tendência μ_t com uma componente sazonal γ_t , acrescida de um choque aleatório normal ϵ_t pode ser expressa como:

$$y_t = \mu_t + \gamma_t + \epsilon_t \quad (3-1)$$

O exemplo de combinação não linear mais simples que pretende ser estudada é a combinação por meio da multiplicação das componentes ou a combinação multiplicativa da componente sazonal com a exponencial da componente de tendência:

$$y_t = \mu_t * \gamma_t + \epsilon_t \quad (3-2)$$

$$y_t = \mu_t + \exp(a + b\mu_t) \gamma_t + \epsilon_t \quad (3-3)$$

No entanto, ao entrar no escopo de combinações não lineares, a quantidade de diferentes maneiras de combinar as componentes de tendência e sazonalidade torna-se muito maior que a quantidade possível com apenas combinações lineares. Também poderiam ser estudados componentes ao quadrado, multiplicação dessas componentes ao quadrado, dentre muitas outras.

Além de estudar novas combinações de componentes em modelos GAS, também é um objetivo deste projeto estudar diferentes maneiras de tratar a variância da componente sazonal, de forma torná-la variante com o tempo. Para exemplificar, podemos imaginar que a variância da sazonalidade de uma série

de vazão, por exemplo, comporta-se de forma diferente a depender da estação do ano, ou em épocas de chuva e seca em determinada região.

Seja um modelo de séries temporais que decompõe a série em tendência, sazonalidade e um erro, como o modelo da equação 3-1. Como será visto no capítulo 4, associado à componente sazonal de um modelo estrutural γ_t há um choque aleatório normal $\omega_t \sim N(0, \sigma_\omega^2)$. Generalizar a variância dessa componente significa substituir σ_ω^2 por $\sigma_{\omega,t}^2$.

A fim de realizar esses experimentos, após o estudo e desenvolvimento teórico necessário, a proposta é implementar essas extensões do arcabouço de modelos GAS a partir do pacote *UnobservedComponentsGAS.jl*, escrito em Julia por [Alves]. Apenas com desenvolvimento dessa ferramenta computacional poderemos averiguar se essas alterações propostas conseguem resolver o problema da dependência sazonal não capturada pelo modelo, evidenciada pela FAC dos resíduos do modelo estimado.

Metodologia

4.1 Fundamentação Teórica

A primeira diferenciação que deve ser feita é em termos de classe de M modelos para séries temporais observation driven (OD) e parameter driven (PD), tal qual proposto em [Cox et al. 1981]. Modelos PD são aqueles cujos parâmetros variantes no tempo são função ~~apenas~~ de seus valores passados e de inovações aleatórias. Ou seja, se f_t é o vetor de parâmetros variantes no tempo e η_t é uma ruído aleatório puro, temos que:

$$f_t = g(f_{t-1}, \eta_t) \quad \text{onde } f_t \text{ é uma função contínua e diferenciável} \quad (4-1)$$

Um exemplo clássico de modelos PD são os modelos de espaço de estados ou modelos de componentes não observáveis (CNO), ~~onde, por exemplo~~

$$f_t = \phi f_{t-1} + \eta_t, \quad \eta_t \sim N(0, \sigma_\eta^2) \quad (4-2)$$

Os modelos score-driven, por sua vez, pertencem à classe OD, o que significa que eles são definidos a partir de uma distribuição condicional de probabilidade que depende dos valores passados da própria série temporal [Harvey 2022]. Em outras palavras,

$$f_{t|t-1} = g(f_{t-2}, S_t), \quad s_t = s_t(\mathbf{Y}_{t-1}) \quad (4-3)$$

onde s_t é chamado de *score* (ver seção ??) e $\mathbf{Y}_{t-1} = (y_{t-2}, \dots, y_2, y_1)$.

Sendo assim, o primeiro passo da modelagem é definir qual é a distribuição de probabilidade seguida pela série a ser modelada. Ou seja, dada uma série \mathbf{y}_t e o seu passado $\mathbf{Y}_{t-1} = (y_{t-1}, y_{t-2}, \dots, y_2, y_1)$, preciso definir uma distribuição condicional tal que

$$y_t \sim p(y_t | f_t; \theta, \mathbf{Y}_{t-1}) \quad \text{estimando } \theta \quad (4-4)$$

onde f_t é o vetor de parâmetros variantes no tempo e θ é o vetor de parâmetros fixos a serem estimados.

Vale ressaltar que a distribuição pode ser de qualquer tipo: contínua, discreta, univariada, multivariada, etc.

Uma vez definida a distribuição condicional de probabilidade, o próximo passo é definir a função de atualização dos parâmetros variantes no tempo, ou *updating mechanism*. Serão abordadas duas possibilidades: a primeira é a função de atualização seguir um processo ARMA(p,q) e a segunda é

desenvolver essa função a partir do arcabouço dos modelos observáveis (CNO). As seções ?? e 4.1.1 a seguir desenvolvem as duas abordagens.

4.1.1

Modelos score-driven - CNO

~~A~~ Outro lado, podemos especificar os modelos GAS a partir do arcabouço dos modelos de componentes não observáveis para a função de atualização de f_t . Essa formulação foi apresentada em [Harvey 2013] e [Caivano, Harvey e Luati 2016] e é especialmente útil para o problema que este trabalho vai endereçar, uma vez que, dentro desse arcabouço, as componentes de tendência e sazonalidade ficam explícitas.

Para explicitar essa formulação, devemos partir de um modelo estrutural básico onde a série temporal é decomposta em componentes de tendência linear estocástica μ_t e sazonalidade estocástica γ_t por *dummies*, como dado a seguir:

$$\begin{aligned} y_t &= \mu_t + \gamma_t + \epsilon_t, \epsilon_t \sim N(0, \sigma^2) & f_t \Rightarrow f_{t|t+1} \\ \mu_{t+1} &= \mu_t + \beta_t + \eta_t, \eta_t \sim N(0, \sigma_\eta^2) \\ \beta_{t+1} &= \beta_t + \xi_t, \xi_t \sim N(0, \sigma_\xi^2) \\ \gamma_{t+1} &= \sum_{j=0}^{s-2} \gamma_{t-j} + \omega_t, \omega_t \sim N(0, \sigma_\omega^2) \end{aligned} \quad (4-5)$$

Onde s é o período sazonal da série e $\epsilon_t, \eta_t, \xi_t, \omega_t$ são choques aleatórios normais de média zero e variância constante no tempo. Essa formulação está melhor desenvolvida no Apêndice B.

É necessário, a partir dessa formulação, alterar os choques aleatórios de cada componente, isto é, $\epsilon_t, \eta_t, \xi_t$ e ω_t para transformar esse modelo em um modelo GAS. A seguir estão os passos necessários para obter um modelo GAS estrutural, supondo que a distribuição de y_t é uma $gama(\alpha, \lambda_t/\alpha)$, com $\lambda_t > 0, \alpha > 0$, sendo o primeiro o parâmetro *shape* e o segundo *scale*.

Sabe-se que, utilizando essa parametrização, $E(y_t|f_t, \mathbf{Y}_{t-1}, \theta) = \lambda_t$ e $V(y_t|f_t, \mathbf{Y}_{t-1}, \theta) = \lambda_t^2/\alpha$. Note que decidi por apenas um parâmetro variante no tempo, o *shape*. Além disso, sabe-se que o *score* padronizado associado ao parâmetro *shape*, com $d = 1$ e *link function* logarítmica, é $\nabla_t^\lambda = \frac{y_t}{\lambda_{t-1}}$ (ver Apêndice A.1). Chamarei, a partir desse momento, esse *score* de u_t e ele fará o papel dos choques aleatórios existentes em um modelo CNO ao ser transscrito para a modelagem *score-driven*.

Agora vamos apresentar a especificação da componente sazonal γ_t através de um mecanismo um pouco mais geral do que aquele apresentado em 4-5. Diferente do que está descrito no Apêndice B, vamos começar assumindo que

o fator sazonal $\gamma_{j,t}$ da estação j no mês t , assumindo uma série mensal, possui a dinâmica de um passeio aleatório sem *drift*. Ou seja:

$$\gamma_{j,t+1} = \gamma_{j,t} + \omega_{j,t} \quad \omega_{j,t} \sim N(0, \sigma_\omega^2) \quad j = 1, \dots, s \quad (4-6)$$

Estação pode significar a própria frequência mensal da série, ou outra divisão periódica qualquer, como as próprias estações do ano.

É possível colocar a equação anterior em forma vetorial, tal que:

$$\boldsymbol{\gamma}_{t+1} = \boldsymbol{\gamma}_t + \boldsymbol{\omega}_t \quad (4-7)$$

onde $\boldsymbol{\gamma}_t = [\gamma_{1,t}, \dots, \gamma_{s,t}]^T$ e $\boldsymbol{\omega}_t = [\omega_{1,t}, \dots, \omega_{s,t}]^T$. Assim como foi feito na parametrização 3 apresentada no Apêndice B, é necessário adicionar a restrição de que a soma dos componentes sazonais de todos os j é igual a zero.

Até o momento, ainda estamos com a formulação dentro do escopo dos modelos de espaço de estados. Para alterar essa formulação para modelos *score-driven*, o primeiro passo será substituir os choque aleatórios $\boldsymbol{\omega}_t$ por $\mathbf{k}_t u_t$, obtendo uma nova equação vetorial. Note que u_t não está em negrito por não representar um vetor, e sim um escalar. A nova equação é que descreve a componente sazonal em um modelo GAS-CNO é:

$$\boldsymbol{\gamma}_{t+1|t} = \boldsymbol{\gamma}_{t|t-1} + \mathbf{k}_t u_t \quad (4-8)$$

onde u_t é o *score* definido acima e $\mathbf{k}_t = [k_{1,t}, \dots, k_{s,t}]$.

É necessário definir um conjunto de variáveis indicadoras z_t para indicar se o fator sazonal $\gamma_{j,t}$ está ou não ativo em um dado mês t . Sendo assim,

$$z_t = \begin{cases} 1, & \text{se } t = j \\ 0, & \text{se } t \neq j \end{cases} \quad (4-9)$$

A partir da definição dessas variáveis, temos que o fator sazonal de cada mês t será dado por

$$s_t = z_t \boldsymbol{\gamma}_t \quad (4-10)$$

Tal como apresentado para as componentes sazonais de modelos CNO, a soma dos fatores sazonais do modelo *score-driven* - CNO deve ser igual a zero.

Com isso, utilizando a equação 4-10 junto das equações para descrever a dinâmica da tendência ao longo do tempo, obtemos o seguinte modelo GAS-CNO com distribuição *gama*.

$$s_{t+1|t} = z_t \boldsymbol{\gamma}_{t+1|t}$$

→ isso vai

implica restrições nos

k_{jt} 's ⇒ ver anexo do

Harvey em livro.

$$\hat{y}_{t+1|t} = E[y_t | f_t, \mathbf{Y}_t, \theta] = \lambda_t = \exp(f_t) \quad (4-11)$$

$$f_{t+1|t} = \mu_{t+1|t} + s_{t+1|t} \quad (4-12)$$

$$\mu_{t+1|t} = \mu_{t|t-1} + \beta_{t|t-1} + k_1 u_t \quad (4-13)$$

$$\beta_{t+1|t} = \beta_{t|t-1} + k_2 u_t \quad (4-14)$$

$$s_{t+1|t} = \mathbf{z}_t \gamma_{t+1|t} \quad (4-15)$$

$$\gamma_{t+1|t} = \gamma_{t|t-1} + \mathbf{k}_t u_t \quad (4-16)$$

sendo $u_t = \nabla_t^\lambda$, o *score*, μ_t a componente de tendência, β_t a componente de inclinação e s_t a componente sazonal.

A principal diferença entre a formulação de um modelo de espaço de estados descrito como 4-5 para um modelo *score driven* descrito como 4-11 é o fato do primeiro possuir choques aleatórios em suas equações de componentes enquanto que o segundo substitui esses choque por uma função do *score*, tornando o modelo dependente das observações passadas da série temporal. Com isso, agora temos a formulação de um modelo *score-driven* cujos parâmetros variantes no tempo não seguem um processo ARMA, mas sim um processo CNO.

4.1.2

Combinação não linear de tendência e sazonalidade

A maneira mais usual de combinar as componentes de tendência e sazonalidade, tanto em modelos estruturais, quanto em modelos GAS-CNO, quando utilizadas distribuições gaussianas e *t-student* é a combinação aditiva:

$$y_t = \mu_t + \gamma_t + \epsilon_t$$

Essa é a combinação utilizada em todos os desenvolvimentos apresentados anteriormente.

No entanto, para distribuições não gaussianas, é comum que utilizemos alguma forma multiplicativa de combinação de tendência e sazonalidade. Suponha, por exemplo, a modelagem de uma série y_t utilizando uma distribuição gama:

$$y_t \sim Gama(\mu/\theta, \theta) \quad \mu > 0, \theta > 0$$

$$E(y_t) = \frac{\mu}{\theta} \cdot \theta = \mu$$

Como foi visto, é preciso definir a distribuição condicional associada ao

modelo:

$$p(y_t | \mathbf{Y}_{\tilde{\mathbf{t}-1}}) \sim Gama \left(\frac{\mu_{t|t-1}}{\theta_{t|t-1}}, \theta_{t|t-1} \right)$$

$$E(y_t | \mathbf{Y}_{\tilde{\mathbf{t}-1}}) = \mu_{t|t-1}$$

Como o objetivo é estimar a média dessa distribuição ao longo do tempo, podemos decompor essa média em componentes de tendência e sazonalidade. Ao mesmo tempo, para garantir que a média seja sempre positiva, tal como manda a restrição desse parâmetro, podemos utilizar uma função de ligação exponencial. Com isso, obtemos a seguinte expressão para a média:

$$\begin{aligned} \mu_{t|t-1} &= \exp(m_{t|t-1} + \gamma_{t|t-1}) \\ &= \exp(m_{t|t-1}) \cdot \exp(\gamma_{t|t-1}) \\ &= m_{t|t-1}^* \cdot \gamma_{t|t-1}^* \end{aligned}$$

Note que, com essa formulação, o modelo final obtido foi um modelo multiplicativo entre as componentes.

Como este projeto não utilizou o ferramental de funções de ligação, essa formulação não foi implementada.

As alterações propostas com o objetivo de generalizar a combinação de tendência e sazonalidade foi inspirada nas variações de ~~de~~^F modelos ETS, um outro modelo muito importante para séries temporais. A Figura 4.1, obtida de [Hyndman e Athanasopoulos 2021], exibe diversas combinações possíveis em modelos ETS. Na figura, **N** significa a ausência da componente, **A** significa combinação aditiva, **M** significa combinação multiplicativa e A_d significa tendência aditiva e amortecida.

Trend	Seasonal		
	N	A	M
N	$\hat{y}_{t+h t} = \ell_t$	$\hat{y}_{t+h t} = \ell_t + s_{t+h-m(k+1)}$	$\hat{y}_{t+h t} = \ell_t s_{t+h-m(k+1)}$
	$\ell_t = \alpha y_t + (1-\alpha)\ell_{t-1}$	$\ell_t = \alpha(y_t - s_{t-m}) + (1-\alpha)\ell_{t-1}$	$\ell_t = \alpha(y_t/s_{t-m}) + (1-\alpha)\ell_{t-1}$
A	$\hat{y}_{t+h t} = \ell_t + hb_t$	$\hat{y}_{t+h t} = \ell_t + hb_t + s_{t+h-m(k+1)}$	$\hat{y}_{t+h t} = (\ell_t + hb_t)s_{t+h-m(k+1)}$
	$\ell_t = \alpha y_t + (1-\alpha)(\ell_{t-1} + b_{t-1})$ $b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1}$	$\ell_t = \alpha(y_t - s_{t-m}) + (1-\alpha)(\ell_{t-1} + b_{t-1})$ $b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1}$	$\ell_t = \alpha(y_t/s_{t-m}) + (1-\alpha)(\ell_{t-1} + b_{t-1})$ $b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1}$
A_d	$\hat{y}_{t+h t} = \ell_t + \phi_h b_t$	$\hat{y}_{t+h t} = \ell_t + \phi_h b_t + s_{t+h-m(k+1)}$	$\hat{y}_{t+h t} = (\ell_t + \phi_h b_t)s_{t+h-m(k+1)}$
	$\ell_t = \alpha y_t + (1-\alpha)(\ell_{t-1} + \phi b_{t-1})$ $b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)\phi b_{t-1}$	$\ell_t = \alpha(y_t - s_{t-m}) + (1-\alpha)(\ell_{t-1} + \phi b_{t-1})$ $b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)\phi b_{t-1}$	$\ell_t = \alpha(y_t/s_{t-m}) + (1-\alpha)(\ell_{t-1} + \phi b_{t-1})$ $b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)\phi b_{t-1}$

Figura 4.1: Família de modelos ETS

Como as possibilidades são inúmeras, a primeira a ser testada é a com-

binação multiplicativa entre tendência e sazonalidade, expressa pelo modelo a seguir. Note que esta combinação independe da dinâmica de cada uma das componentes, que não foram especificadas neste momento.

$$\mu_t = m_t \times \gamma_t$$

onde μ_t é um parâmetro variante no tempo de uma distribuição de probabilidade, m_t é a componente de tendência e γ_t é a componente de sazonalidade.

4.1.3

Tratamento da variância da componente sazonal

O outro objetivo deste projeto é desenvolver um modelo *score-driven* generalizando a variância da componente sazonal com o intuito de capturar variâncias distintas de diferentes períodos do ano. Para exemplificar, é razoável supor que, em uma série de precipitação, a variância da quantidade de chuva nos meses de inverno é diferente daquela dos meses de verão.

COMPLETAR

4.1.4

Restrição dos γ_t para obter um amortecimento exponencial

?

4.2

Implementação Computacional

Nesta sessão, estão descritas as implementações e os testes realizados neste projeto. Na sessão 4.2.1 estão definidas as séries temporais que foram utilizadas. Em seguida, na sessão 4.2.2, apresento as distribuições de probabilidade escolhidas para modelar as séries temporais definidas. Já na sessão 4.2.3 explicito o modelo GAS utilizado nos testes, definindo quais foram as componentes não observáveis consideradas. A seguir, na sessão 4.2.6 são exibidos os *benchmarks* obtidos considerando as definições das 3 sessões anteriores que servirão de comparação para avaliar a qualidade das alterações propostas por este projeto. Por fim, as sessões ?? e ?? apresentam as formulações dos modelos testados para cada uma dessas alterações propostas.

É importante comentar que, fora os *benchmarks* da sessão 4.2.6, nenhum outro resultado é apresentado neste capítulo, dado que o objetivo seu objetivo é apenas formalizar a metodologia utilizada para os testes. Todos os resultados dos testes desenvolvidos encontram-se no capítulo 5.

4.2.1

Definição das séries temporais

Nesta seção está descrita a metodologia de testes utilizada neste projeto. Com o intuito de implementar e testar as alterações apresentadas para modelos GAS-CNO nas seções 4.1.2, 4.1.3 e 4.1.4 foram escolhidas 3 séries temporais mensais dentro do contexto de clima e energia.

A primeira é uma série temporal de carga mensal média, em MW, das regiões Sudeste e Centro Oeste do Brasil, obtida de <https://www.snirh.gov.br/hidroweb/serieshistoricas>. A figura 4.2 exibe o gráfico dessa série temporal, bem como seu histograma. Optou-se por utilizar dados dessa série a partir de janeiro de 2002 para evitar os *outliers* gerados pelo racionamento de junho e julho de 2001. Algumas características a se destacar sobre essa série é sua não estacionariedade, cada a existência de tendência, um visível aumento da variabilidade da série com o passar do tempo e, por fim, uma possível diminuição da inclinação da tendência nos últimos anos da série.

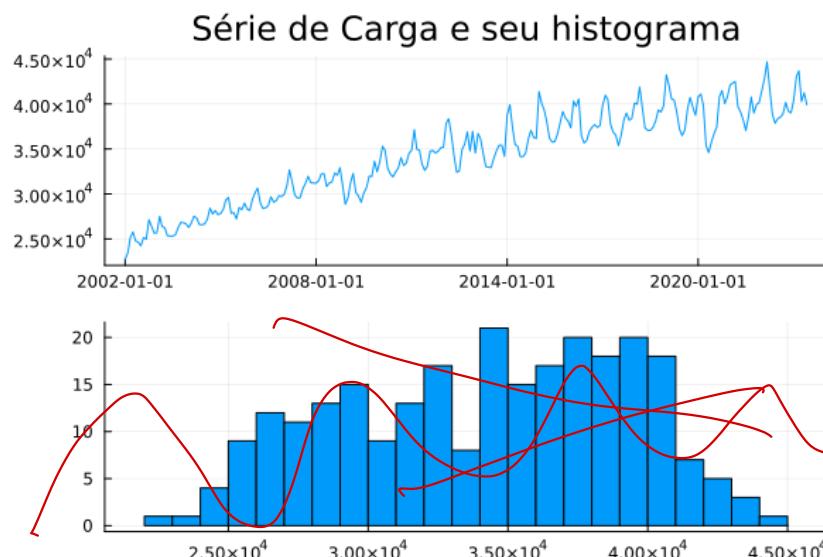


Figura 4.2: Série Mensal de Carga

Em seguida, foi escolhida uma série mensal de ENA, isto é de Energia Natural Afluente, que corresponde à quantidade de água que chega a uma certa usina ou reservatório. A série escolhida foi obtida de https://www.ons.org.br/Paginas/resultados-da-operacao/historico-da-operacao/energia_afluentes_subsistemas.aspx e corresponde a ENA Bruta do Sudeste e Centro Oeste do país. Novamente, a figura 4.3 exibe o gráfico da série e seu histograma. A primeira característica que chama a atenção sobre essa série é sue histograma indicar que não se

trata de uma distribuição normal. Outro fato importante é a série apresentar estacionariedade na média e um padrão sazonal visualmente marcante.

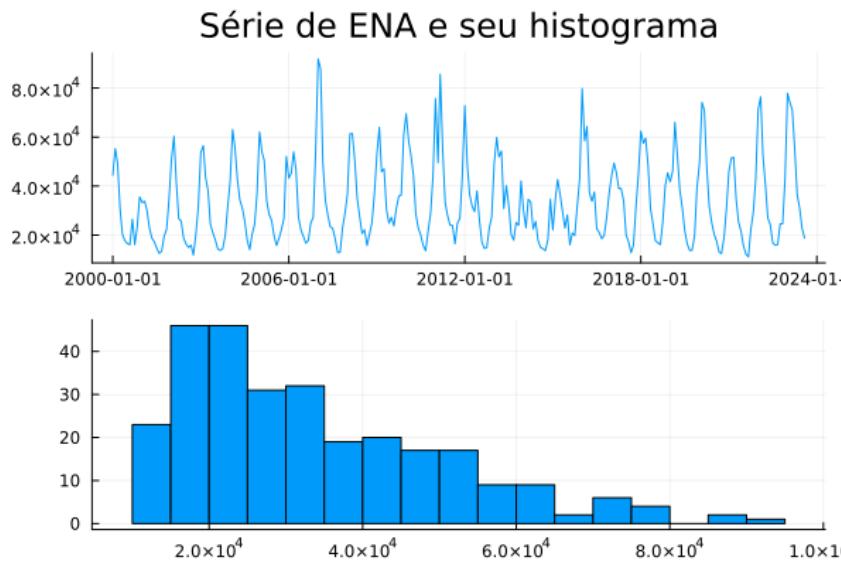


Figura 4.3: Série Mensal de ENA

Por fim, também foi escolhida uma série de vazão ~~área~~ da estação de Juazeiro da bacia do Rio São Francisco. Essa série foi obtida de <https://www.snirh.gov.br/hidroweb/serieshistoricas>. A figura 4.4 exibe o gráfico da série temporal e seu histograma. Mais uma vez, nota-se que o histograma indica que a série não segue uma distribuição normal. Além disso, dentre as 3 séries selecionadas, essa é a "menos comportada", por não apresentar sinais claros de repetição sazonal nem nenhum outro padrão repetitivo ao longo do tempo.

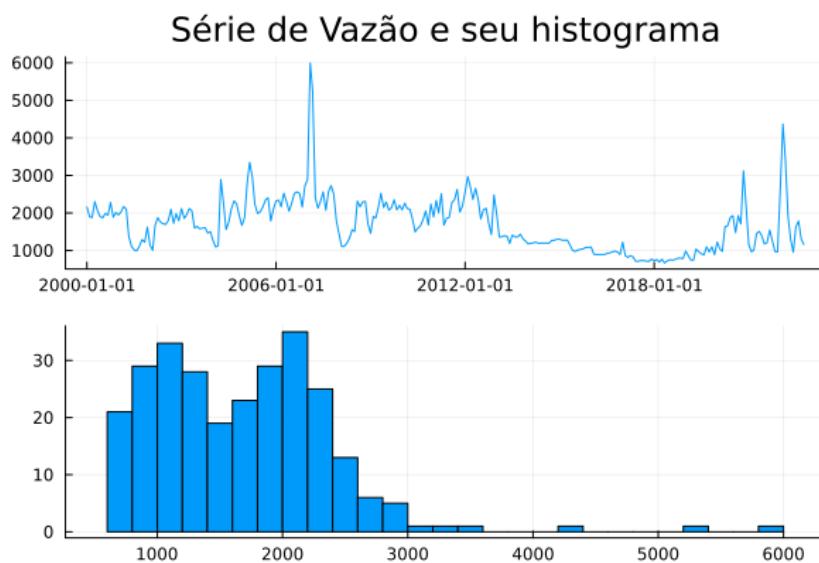


Figura 4.4: Série Mensal de Vazão

*para todos
e
não obs.*

Série	T	Treino	Teste
Carga	258	01/2002	07/2022
		a	a
		06/2022	06/2023
ENA	284	01/2000	09/2022
		a	a
		08/2022	08/2023
Vazão	272	01/2000	11/2021
		a	a
		09/2021	10/2022

Tabela 4.1: Divisão das séries temporais em treino e teste.

W Todas as séries foram separadas em períodos de treino, ou *in sample* e teste, ou *out of sample*. A tabela 4.1 exibe, para cada série temporal, seu tamanho T , as datas separadas para a estimativa dos modelos (Treino) e as datas separadas para validação do modelo (Teste). Note que, para todas as séries, foram separadas as últimas 12 observações para servirem de dados *out of sample*.

4.2.2

Definição das distribuições de probabilidade ~~a serem utilizadas~~

Após escolhidas as séries temporais a serem utilizadas nos experimentos, o próximo passo foi decidir quais distribuições seriam testadas. Foram escolhidas 3 distribuições não gaussianas: lognormal, gama e weibull.

Distribuição LogNormal

Seja Z uma variável aleatória normal padrão, $Z \sim N(0, 1)$ e μ, σ dois números reais tal que $\sigma > 0$. Então, uma variável aleatória $X = e^{\mu + \sigma Z}$ segue uma distribuição LogNormal, tal que $E[\ln(X)] = \mu$ e $V[\ln(X)] = \sigma^2$.

Dado que o pacote *UnobservedComponentsGAS* já possui a distribuição normal implementada, ao invés de implementar a distribuição LogNormal, optou-se por estimar o GAS-CNO Normal no logaritmo da série temporal. Isto é, dado uma série temporal \tilde{Y}_t , definiu-se:

$$p(\ln(y_t) | \ln(\tilde{Y}_{t-1})) \sim \text{Normal}(\mu, \sigma^2)$$

É importante ressaltar que, após estimado o modelo na escala logarítmica, com o intuito de recuperar o *fit in sample* e a previsão para a escala original, não basta exponenciar os valores estimados. O correto é seguir a expressão:

$$\hat{y}_{original} = e^{\hat{y}_{log}} e^{\frac{1}{2} \frac{SSR}{N-K}}$$

onde \hat{y}_{log} são os valores estimados na escala logarítmica, SSR é a soma dos quadrados dos resíduos do modelo estimado (*sum of squared residuals*), N é o tamanho da série estimada e K são os graus de liberdade do modelo, que equivalhe à quantidade de parâmetros estimados.

Distribuição Gama

Como a distribuição gama não estava originalmente implementada no pacote *UnobservedComponentsGAS*, sua implementação fez parte desse projeto. Para tal, escolheu-se a parametrização $X \sim Gama(\alpha, \lambda)$, tal que $E[X] = \lambda$ e $V[X] = \lambda^2/\alpha$.

Dado que essa distribuição foi implementada dentro do escopo deste projeto, aqui estão expostos alguns resultados importantes para a distribuição gama em sua parametrização α e λ . Somado a isso, o desenvolvimento completo dos resultados mostrados a seguir encontram-se no apêndice A.1

A função densidade de probabilidade e o logaritmo natural dessa função são:

$$f(y_t | \tilde{y}_{t-1}; \alpha_t, \lambda_t) = \frac{1}{\Gamma(\alpha_t)} \frac{1}{(\alpha_t^{-1} \lambda_t)^{\alpha_t}} y_t^{\alpha_t-1} e^{\frac{-\alpha_t}{\lambda_t} y_t}$$

$$\begin{aligned} \ln f(y_t | \tilde{y}_{t-1}; \alpha_t, \lambda_t) &= -\ln(\Gamma(\alpha_t)) - \alpha_t \ln(1/\alpha_t) - \alpha_t \ln(\lambda_t) + \\ &\quad + (\alpha_t + 1) \ln(y_t) - (\alpha_t / \lambda_t) y_t \end{aligned}$$

$y_t \Rightarrow \tilde{y}_{t-1}$
 \sim
 \uparrow
 bold
 sem tilde
 sem barra

O vetor score obtido para essa distribuição é:

$$\nabla_t = \begin{pmatrix} \nabla_t^\alpha \\ \nabla_t^\lambda \end{pmatrix} = \begin{pmatrix} \ln(y_t) - \frac{y_t}{\lambda_t} + \ln(\alpha_t) - \psi_1(\alpha_t) - \ln(\lambda_t) + 1 \\ \frac{\alpha_t}{\lambda_t} \left(\frac{y_t}{\lambda_t} - 1 \right) \end{pmatrix}$$

onde a função $\psi_1(\alpha_t)$ é a função *digamma* definida como $\psi_1(\alpha_t) = \frac{\Gamma'(\alpha_t)}{\Gamma(\alpha_t)}$

Por fim, a última estrutura que precisa ser definida é a matriz de informação de fisher $I_{t|t-1}$.

$$I_{t|t-1} = \begin{pmatrix} I_{t|t-1}^{\alpha, \alpha} & I_{t|t-1}^{\alpha, \lambda} \\ I_{t|t-1}^{\lambda, \alpha} & I_{t|t-1}^{\lambda, \lambda} \end{pmatrix} = \begin{pmatrix} \psi_2(\alpha_t) - \frac{1}{\alpha_t^2} & 0 \\ 0 & \frac{\alpha_t}{\lambda_t^2} \end{pmatrix}$$

onde a função $\psi_2(\alpha_t)$ é a função *trigamma* definida como $\psi_2(\alpha_t) = \frac{\psi'_1(\alpha_t)}{\psi_1(\alpha_t)}$

4.2.3

Definição das componentes não observáveis

O próximo passo na definição da metodologia do projeto é especificar as componentes não observáveis dos modelos GAS a serem implementados e testados. Embora o pacote *UnobservedComponentsGAS* forneça diferentes possibilidades de especificação de componentes, por fins de simplicidade, o modelo GAS a ser utilizado será um modelo com tendência e sazonalidade estocásticas com apenas 1 parâmetro variando no tempo. Dito isso, para as distribuições listadas anteriormente, optei pelos seguintes modelos:

$$\begin{aligned} P(y_t | \underset{\sim}{y_{t-1}}; \mu_t, \sigma^2) &\sim \text{Lognormal}(\mu_t, \sigma^2) \\ P(y_t | \underset{\sim}{y_{t-1}}; \alpha, \lambda_t) &\sim \text{Gama}(\alpha, \lambda_t) \end{aligned}$$

μ_t λ_t

A escolha de μ e λ como parâmetros variantes no tempo se deu pelo fato deles representarem as médias de suas respectivas distribuições.

Uma vez definidos os parâmetros variantes no tempo, define-se a dinâmica de atualização desses parâmetros. Essa escolha depende principalmente do comportamento da série temporal.

A seguir está descrito o modelo para a média m_t variante no tempo. Note que é exatamente o mesmo que seria definido para média λ_t .

Para a série de carga, dado seu perfil não estacionário, o parâmetro de média será modelado com um passeio aleatório com *drift* como componentes de tendência e com sazonalidade trigonométrica. As equações abaixo descrevem este modelo:

$$m_{t+1|t} = m_{t+1|t} + \delta_{t+1|t}$$

$$\mu_t = m_t + \gamma_t \quad (4-17)$$

$$m_{t+1|t} = m_{t|t-1} + b_{t|t-1} + \kappa_m s_t$$

$$b_{t+1|t} = b_{t|t-1} + \kappa_b s_t$$

$$\gamma_{t+1|t} = \sum_{i=1}^{S/2} \gamma_{i,t}$$

$$\gamma_{i,t} = \gamma_{i,t-1} \cos(2\pi i/S) + \gamma_{i,t-1}^* \sin(2\pi i/S) + \kappa_s s_t$$

$$\gamma_{i,t}^* = \gamma_{i,t-1} \sin(2\pi i/S) + \gamma_{i,t-1}^* \cos(2\pi i/S) + \kappa_s \tilde{s}_t$$

onde S é o período sazonal da série (12 para séries mensais), s_t é o vetor *score*, m_t é a componente de tendência e γ_t a componente de sazonalidade.

A série de ENA, por sua vez, não apresenta tendência, o que torna o modelo acima inadequado. Sendo assim, será mantida a componente sazonal

via funções trigonométricas, mas a componente de tendência será descrita por meio de um processo $AR(p)$. As equações abaixo descrevem este modelo:

$$\begin{aligned} \text{MST}_t &= m_{t+1|t} + \gamma_{t+1|t} \\ m_t &= m_t + \gamma_t \end{aligned} \quad (4-18)$$

$$m_{t+1|t} = \phi_0 + \sum_{i=1}^p \phi_i m_{t|t-i} + \kappa_m s_t$$

$$\gamma_{t+1|t} = \sum_{i=1}^{S/2} \gamma_{i,t}$$

$$\gamma_{i,t} = \gamma_{i,t-1} \cos(2\pi i/S) + \gamma_{i,t-1}^* \sin(2\pi i/S) + \kappa_s s_t$$

$$\gamma_{i,t}^* = \gamma_{i,t-1} \sin(2\pi i/S) + \gamma_{i,t-1}^* \cos(2\pi i/S) + \kappa_s s_t$$

S_t → escalar periódico π de período S que é o resultado da transformação de Fourier.

4.2.4

Diagnósticos dos Resíduos

Parte essencial da modelagem de séries temporais é realizar diagnósticos nos resíduos do modelo estimado, com o intuito e verificar a qualidade da estimação. De acordo com [Hyndman e Athanasopoulos 2021], os resíduos de um bom modelo de série temporal devem apresentar duas principais características: ausência de correlação e média zero. Além desses dois requisitos, também é desejável que eles sejam homocedásticos e que sejam oriundos de uma distribuição normal.

Apresentados esses objetivos, a maneira mais simples de definir o resíduo de um modelo é tal que

$$r_t = y_t - \hat{y}_t$$

onde r_t é a série de resíduos, y_t é a série temporal a ser modelada e \hat{y}_t é o valor estimado pelo modelo. Para ir além dessa definição, podemos utilizar os resíduos de pearson [Fernandes 2023], definidos como:

$$r_t = \frac{y_t - \mathbb{E}(y_t | f_t, F_t, \theta)}{\sqrt{\text{Var}(y_t | f_t, F_t, \theta)}}, \quad t = 1, \dots, T$$

onde y_t é a série temporal, $\mathbb{E}(y_t | f_t, F_t, \theta)$ é a média do modelo probabilístico e $\text{Var}(y_t | f_t, F_t, \theta)$ é a variância desse modelo.

Para compor o diagnóstico dos resíduos e verificar se as características desejadas foram obtidas, é comum realizar testes de hipóteses. Em especial, os testes de Jarque Bera, que avalia normalidade dos resíduos, o teste de Ljung-Box, que avaliar presença de autocorrelação nos resíduos e o teste ARCHLM, que investigar presença de heterocedasticidade. As hipóteses nulas destes 3 testes são:

- Teste JarqueBera: H_0 supõe resíduos normalmente distribuídos

- Teste ARHLM: H_0 supõe resíduos homocedásticos
- Teste LjungBox: H_0 supõe resíduos descorrelatados

Para fins de metodologia, será utilizado nível de significância $\alpha = 5\%$ para esses testes. Além disso, vale comentar que o tamanho da amostra dos testes pode influenciar fortemente a rejeição da hipótese nula do teste, de tal forma que um resultado, mesmo que seja estatisticamente significante, pode não ser relevante.

Para complementar o teste de hipótese para verificar a normalidade dos resíduos, é comum utilizar o QQ-Plot (*quantile-quantile plot*). Ele permite comparar os quantis dos nossos dados com os quantis de uma distribuição de probabilidade qualquer. Para fins de diagnóstico de resíduos, comparamos os quantis dos resíduos de pearson com os quantis de uma distribuição normal padrão (média 0 e desvio padrão 1).

4.2.5

Métrica de avaliação

Por fim, a última etapa da avaliação do modelo é o cálculo de uma métrica de aderência dos valores estimados ou previstos pelo modelo com os valores reais da série. Muitas são essas métricas de erro: erro quadrático médio (*MSE*), raiz quadrada do erro quadrático médio (*RMSE*), erro absoluto médio (*MAE*), dentre outros.

A métrica escolhida para avaliar a estimação e previsão do modelo neste trabalho é o erro médio percentual absoluto, ou *MAPE*, em inglês. Ele foi escolhido por sua interpretabilidade, uma vez que trata de um erro percentual. O MAPE é definido como:

$$MAPE = \frac{1}{T} \sum_{t=1}^T \left| \frac{y_t - \hat{y}_t}{y_t} \right| \cdot 100 \text{ } \% \quad , \quad y_t \neq 0$$

onde T é o tamanho da série, y_t é a série temporal e \hat{y}_t é o valor estimado ou previsto pelo modelo.

Esta métrica será utilizada em dois contextos: para avaliar a qualidade da estimação *in sample* e a qualidade da previsão *out of sample*.

4.2.6

Benchmarks

Com o intuito de avaliar e comparar os resultados obtidos a partir das alterações propostas neste trabalho para modelos GAS-CNO, foram criados alguns *benchmarks* para cada uma das séries temporais apresentadas. Primeiramente, foi estimado um modelo AutoARIMA para cada uma das séries. Em

seguida, foram estimados os modelos GAS-CNO LogNormal e Gamma com as especificações apresentadas na sessão 4.2.3. Todos os resultados desses modelos *benchmarks* estão apresentados no capítulo 5.

5

Resultados

Neste capítulo estão apresentados todos os resultados das alterações descritas nas seções do capítulo 4.

A apresentação segue o seguinte padrão: para cada modelo, são apresentados os gráficos de *fit in sample* e componentes estimadas, em seguida o diagnóstico dos resíduos, composto de gráfico, FAC, histograma, QQ-Plot e testes de hipótese. Por fim, o resultado da previsão *out of sample* 12 passos à frente. Ao final, as tabelas 5.6 e 5.12 exibem os MAPEs de treino e teste para cada série e cada modelo. Primeiro estão os resultados para a série de carga e, em seguida, para a série de ENA.

5.1 Série de Carga

5.1.1 Benchmark: AutoARIMA

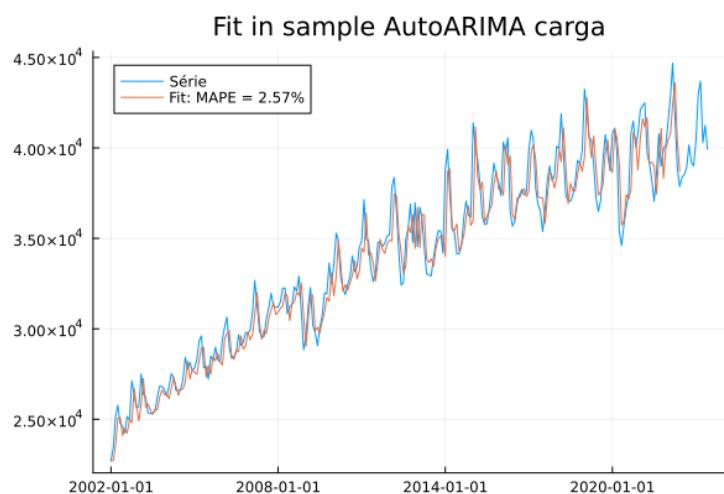


Figura 5.1: Fit in Sample do modelo AutoARIMA para série de carga

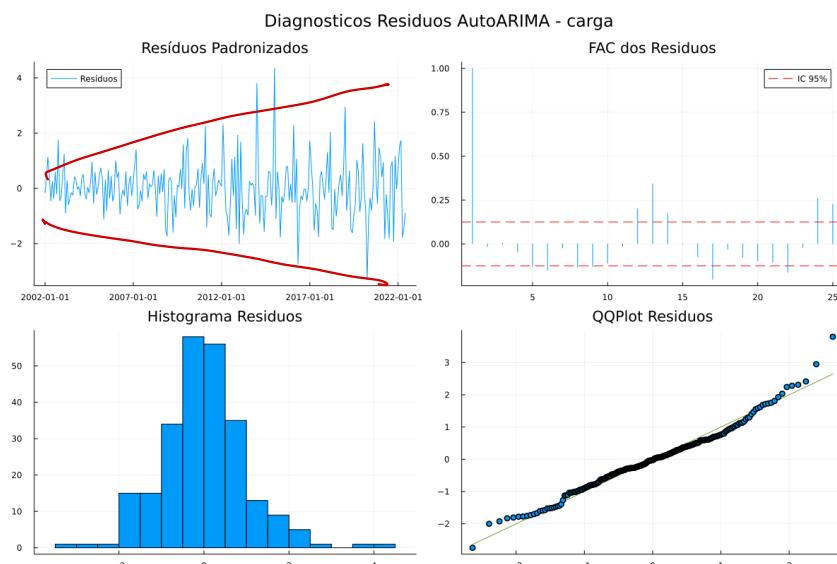


Figura 5.2: Diagnósticos dos resíduos do modelo AutoARIMA para série de carga

→ talvez o modelo ~~deveria~~
sido ajustado ao log de Série.

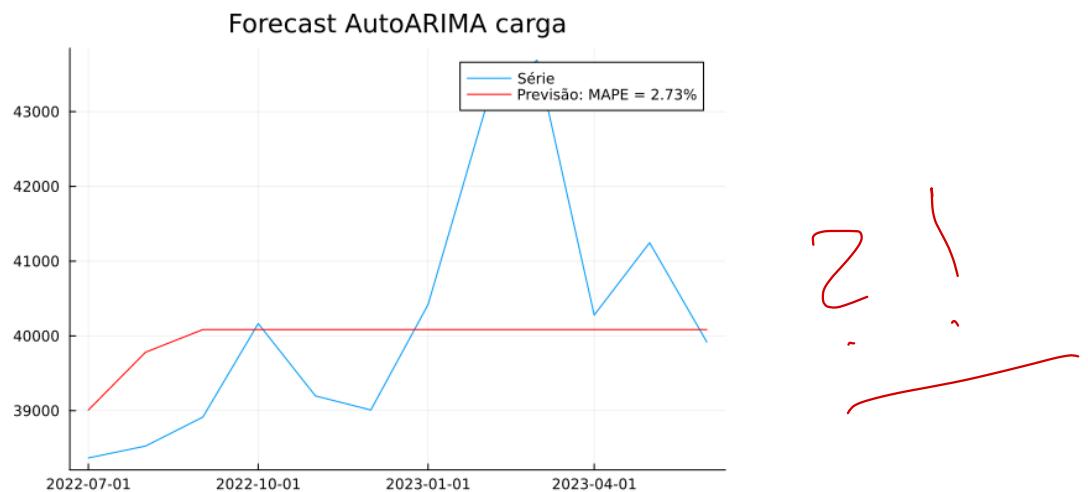


Figura 5.3: Previsão 12 passos à frente do modelo AutoARIMA para série de carga

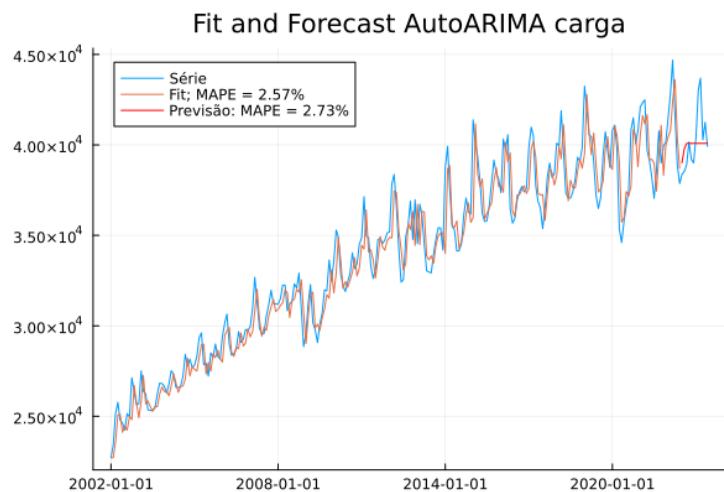


Figura 5.4: Fit in sample e previsão do modelo AutoARIMA para série de carga

MAPE Treino	MAPE Teste
2.57%	2.73%

Tabela 5.1: MAPEs de treino e teste do modelo AutoARIMA para série de carga

5.1.2

Benchmark: GAS-CNO LogNormal

As figuras 5.5 e 5.6 exibem os valores estimados para a série de carga, bem como os valores estimados para as componentes do modelo, a partir de um modelo GAS-CNO LogNormal especificado na equação 4-17.

O modelo estimado consegue, aparentemente, capturar bem a dinâmica da série *in sample*, dado que o *fit in sample* se mostra razoável, deixando

de capturar alguns vales e picos da série apenas. Quando olhamos para as componentes estimadas, podemos ver que o modelo conseguiu estimar uma tendência crescente e razoavelmente linear, bem como um perfil sazonal bem definido.

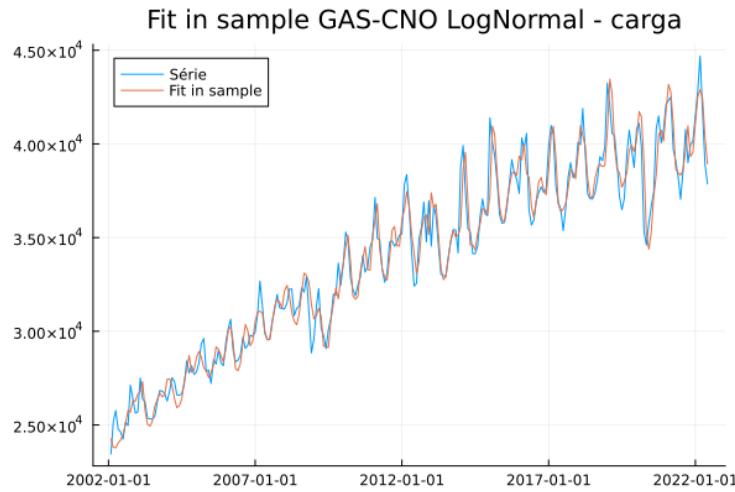


Figura 5.5: Fit in Sample do modelo GAS-CNO LogNormal para série de Carga
Componentes GAS-CNO LogNormal - carga

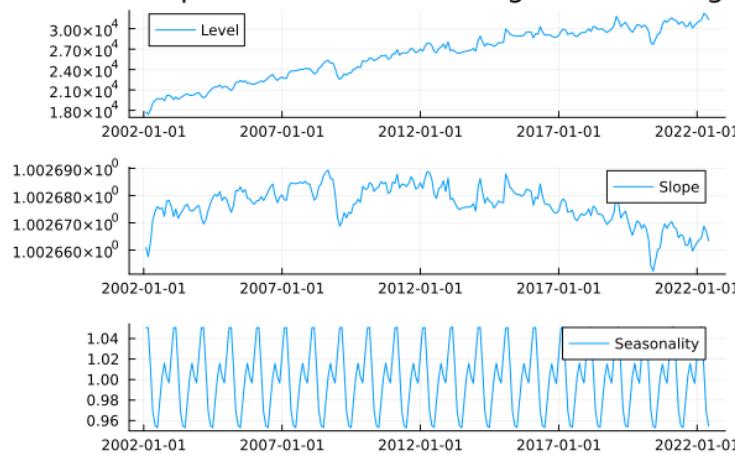


Figura 5.6: Componentes estimadas do modelo GAS-CNO LogNormal para série de Carga

Em seguida, a figura 5.7 exibe todos os gráficos relacionados aos resíduos do modelo GAS-CNO Lognormal para série de Carga. Para completar esse diagnóstico, a tabela 5.2 exibe os resultados de 3 testes de hipótese usuais para modelagem de série temporais, todos com nível de significância $\alpha = 5\%$.

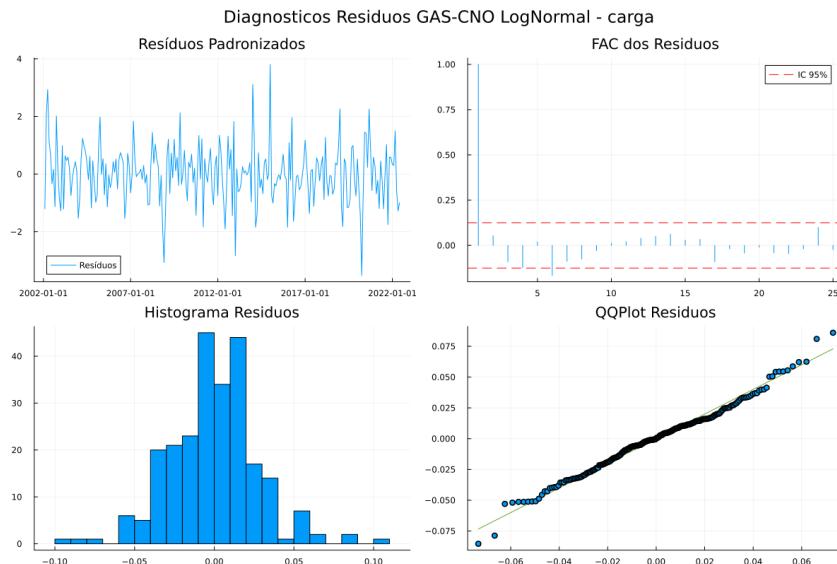


Figura 5.7: Diagnósticos dos resíduos do modelo GAS-CNO LogNormal para série de Carga

Teste	p-valor	Rejeição
Jarque Bera	9.49e-6	Rejeita H_0
ARCHLM	0.01462	Rejeita H_0
LjungBox	0.00070	Rejeita H_0

Tabela 5.2: Testes de hipótese para resíduos do modelo GAS-CNO LogNormal para série de Carga

Ao analisar o gráfico dos resíduos padronizados, nota-se que há alguns resíduos ainda próximos de 3 desvios padrão da média 0 que ainda podem ser melhorados. A FAC, por sua vez, revela que o modelo conseguir capturar praticamente toda a dependência linear da série, havendo ainda apenas um pequeno afeito no *lag* 6. Importante ressaltar que a FAC indica que a dependência sazonal foi bem capturada. Embora o teste de Ljung Box tenha rejeitado H_0 , indicando que há evidências de correlação nos resíduos, é razoável concluir que essa rejeição foi fortemente influenciada pelo tamanho da série, dado o resultado da FAC. Por fim, o teste de Jarque Bera também rejeita normalidade. Ao analisarmos o QQplot, podemos ver um pouco de efeito de cauda gorda que, certamente, influenciou no resultado deste teste contra a hipótese de normalidade.

Por fim, a figura 5.8 exibe a previsão 12 passos à frente do modelo estimado em comparação com os dados de teste separados inicialmente. Para complementar a interpretação desses resultados, a tabela 5.5 exibe os MAPEs de treino e teste para esse modelo.

O gráfico da previsão fora da amostra atesta a qualidade da estimação do modelo pois, embora ele não acerte perfeitamente a previsão pontual, a dinâmica da previsão acompanha muito bem a dinâmica da série de teste. Um MAPE de 2.36% fora da amostra corrobora com essa conclusão. Importante notar também que o MAPE de teste não piorou muito em relação ao MAPE de treino, o que indica que o modelo consegue generalizar de forma satisfatória para fora da amostra.

Mesmo com resultados razoáveis, espera-se que as alterações propostas por este projeto possam melhorar as métricas da estimação e, principalmente, da previsão.

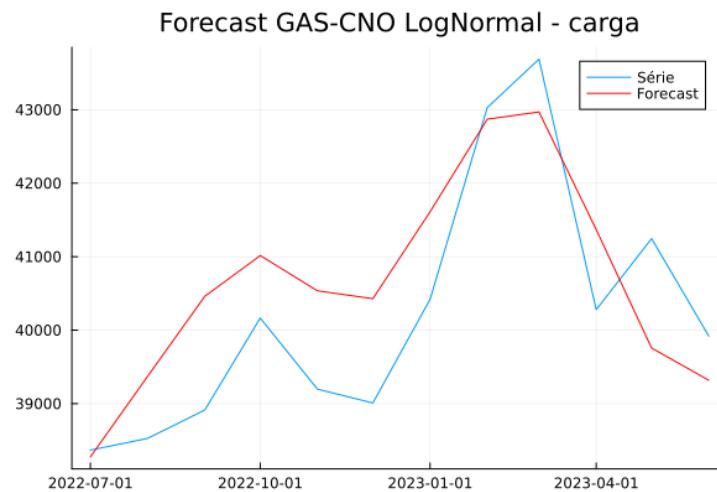


Figura 5.8: Previsão 12 passos à frente do modelo GAS-CNO LogNormal para série de Carga

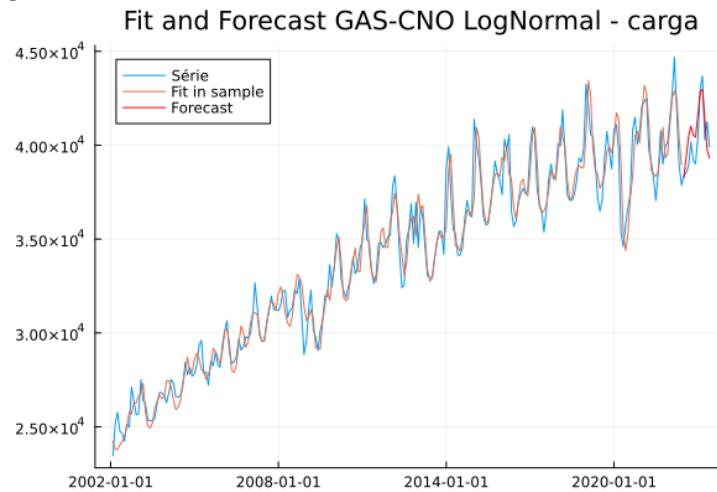


Figura 5.9: Fit in sample e previsão do modelo GAS-CNO LogNormal para série de Carga

MAPE Treino	MAPE Teste
2.07%	2.36%

Tabela 5.3: MAPEs de treino e teste do modelo GAS-CNO LogNormal para série de Carga

5.1.3

Benchmark: GAS-CNO Gamma

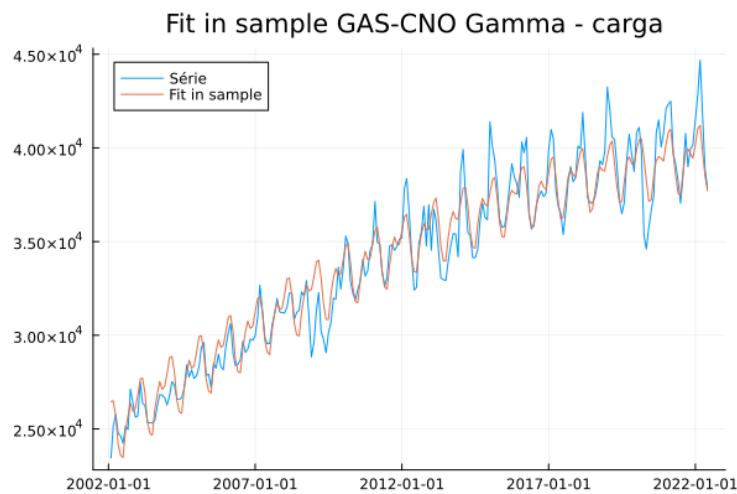


Figura 5.10: Fit in Sample do modelo GAS-CNO Gamma para série de Carga
Componentes GAS-CNO Gamma - carga

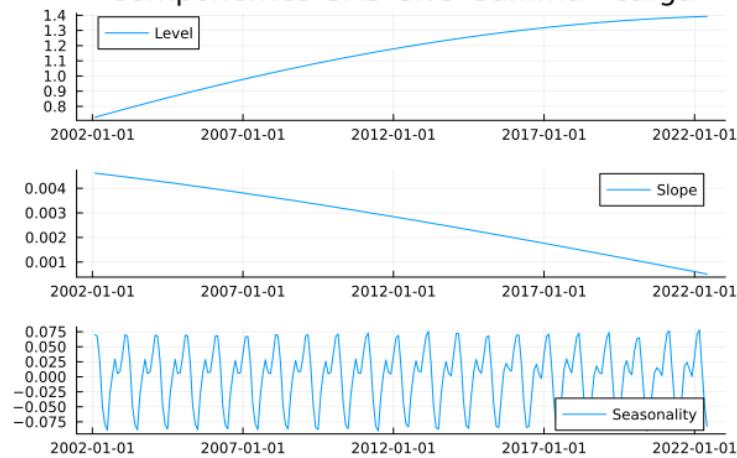


Figura 5.11: Componentes estimadas do modelo GAS-CNO Gamma para série de Carga

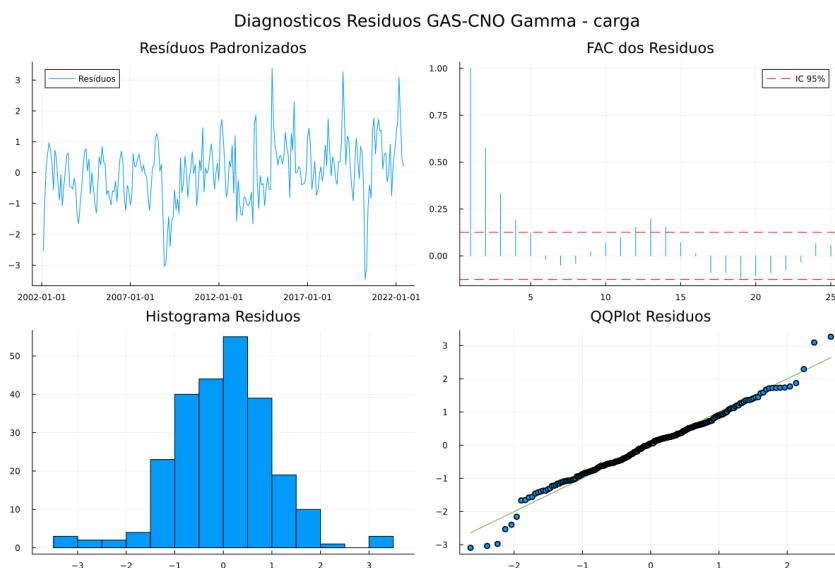


Figura 5.12: Diagnósticos dos resíduos do modelo GAS-CNO Gamma para série de Carga

Teste	p-valor	Rejeição
Jarque Bera	4.41e-6	Rejeita H_0
ARCHLM	6.02e-6	Rejeita H_0
LjungBox	4.46e-32	Rejeita H_0

Tabela 5.4: Testes de hipótese para resíduos do modelo GAS-CNO Gamma para série de Carga

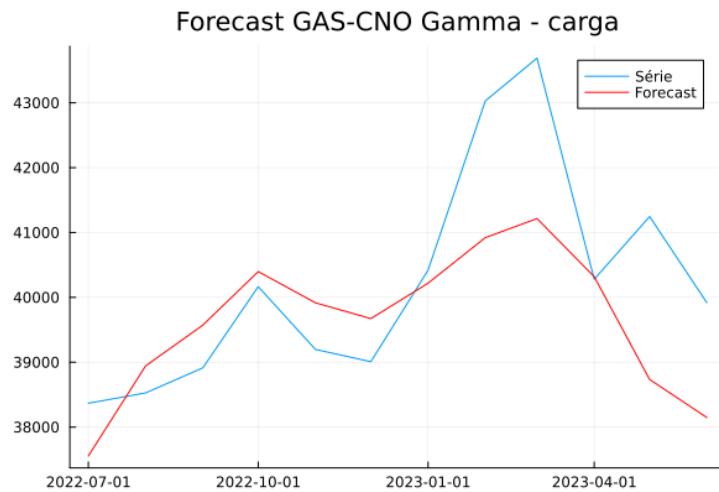


Figura 5.13: Previsão 12 passos à frente do modelo GAS-CNO Gamma para série de Carga

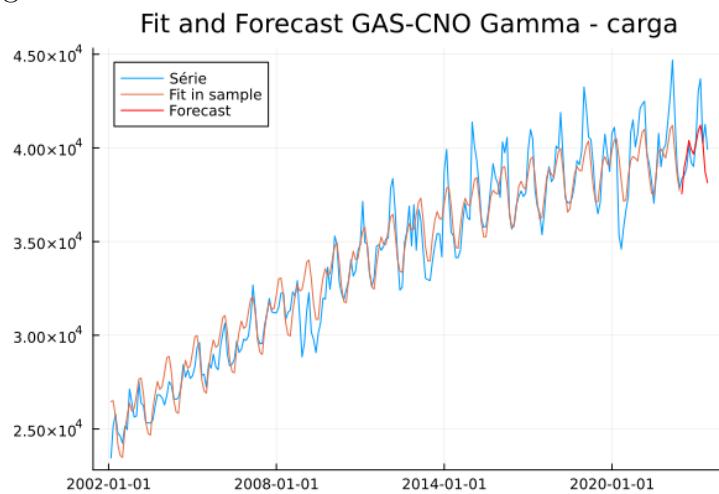


Figura 5.14: Fit in sample e previsão do modelo GAS-CNO Gamma para série de Carga

MAPE Treino	MAPE Teste
2.61%	2.56%

Tabela 5.5: MAPEs de treino e teste do modelo GAS-CNO LogNormal para série de Carga

5.1.4

GAS-CNO LogNormal com combinação multiplicativa

5.1.5

GAS-CNO Gamma com combinação multiplicativa

5.1.6

GAS-CNO LogNormal com múltiplas variâncias

5.1.7

GAS-CNO Gamma com múltiplas variâncias

5.1.8

MAPES

Modelo - Carga	MAPE Treino	MAPE Teste
AutoARIMA	2.57	2.73
GAS-CNO LN	2.07	2.36
GAS-CNO G	2.61	2.56
GAS-CNO LN Mult	mape	mape
GAS-CNO G Mult	mape	mape
GAS-CNO LN MultiVar	mape	mape
GAS-CNO G MultiVar	mape	mape

Tabela 5.6: MAPEs de treino e teste para série de carga

5.2

Série de ENA

5.2.1

Benchmark: AutoARIMA

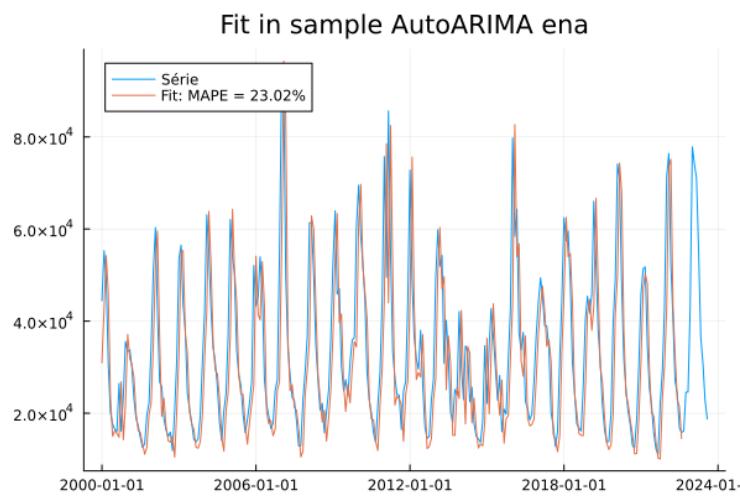


Figura 5.15: Fit in Sample do modelo AutoARIMA para série de ENA

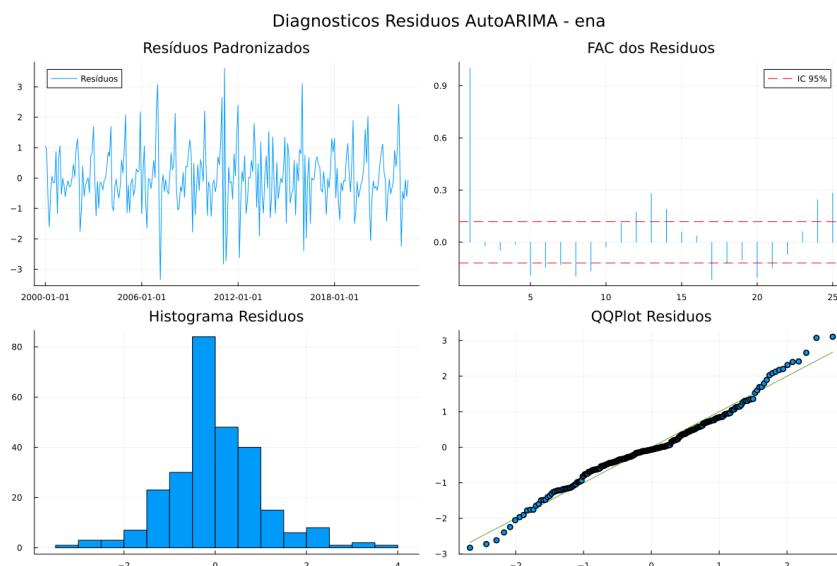


Figura 5.16: Diagnósticos dos resíduos do modelo AutoARIMA para série de ENA

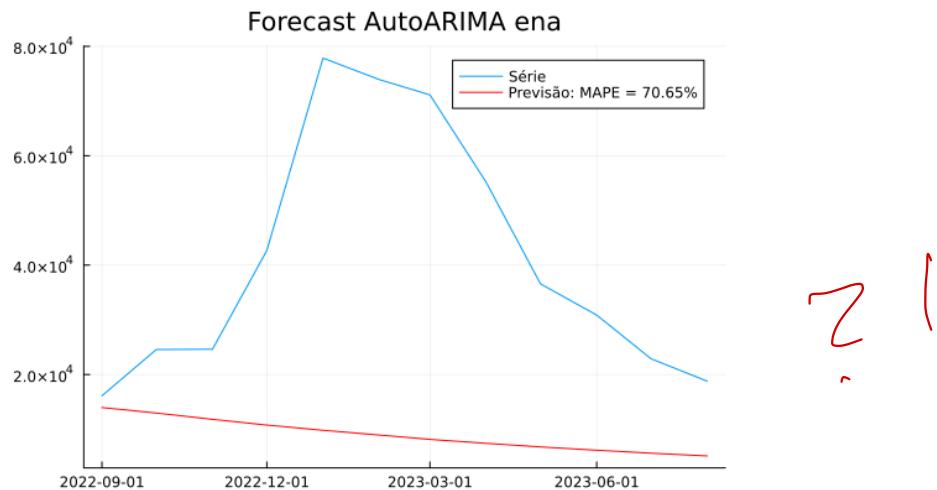


Figura 5.17: Previsão 12 passos à frente do modelo AutoARIMA para série de ENA

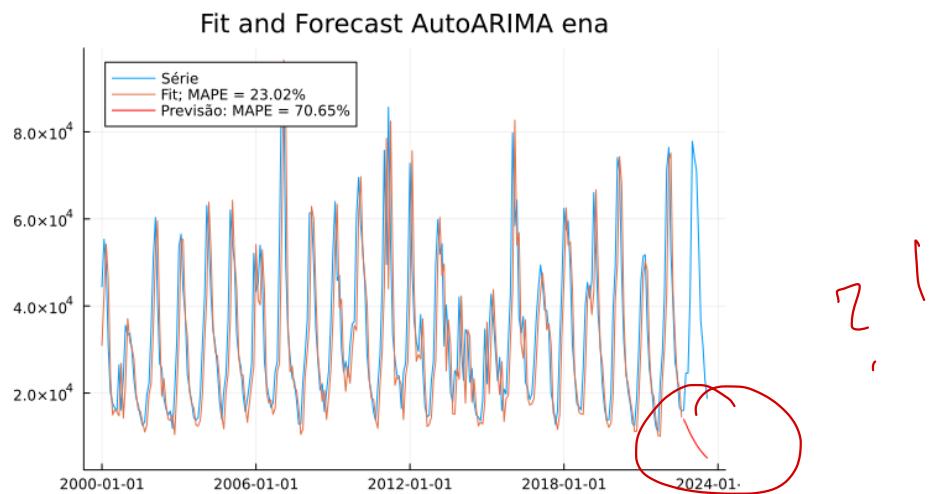


Figura 5.18: Fit in sample e previsão do modelo AutoARIMA para série de ENA

MAPE Treino	MAPE Teste
23.02%	70.65%

Tabela 5.7: MAPEs de treino e teste do modelo AutoARIMA para série de ENA

5.2.2

Benchmark: GAS-CNO LogNormal

As figuras 5.19 e 5.20 exibem os valores estimados para a série de ENA, bem como os valores estimados para as componentes do modelo.

Como a série de ENA foi modelada com um $AR(1)$, a única componente exibida é a componente sazonal. Nota-se que o modelo capturou um perfil

sazonal extremamente bem comportado e coerente com a percepção visual da série de ENA.

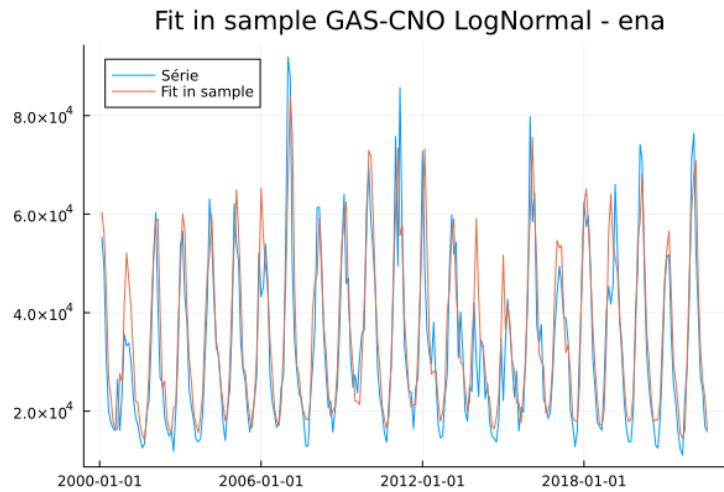


Figura 5.19: Fit in Sample do modelo GAS-CNO LogNormal para série de ENA

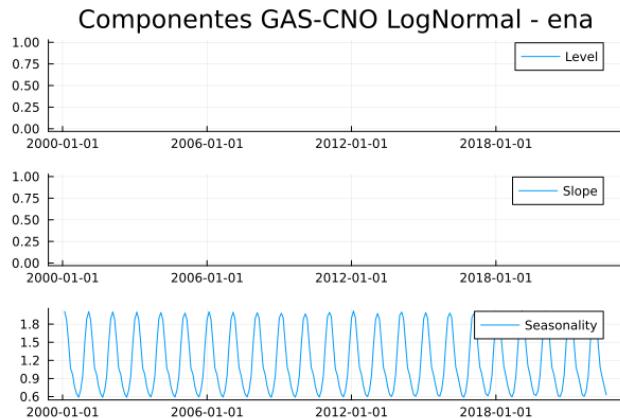


Figura 5.20: Componentes estimadas do modelo GAS-CNO LogNormal para série de ENA

A figura 5.21, junto da tabela 5.8 exibem todos os resultados dos diagnósticos dos resíduos do modelo estimado para a série de ENA. Novamente, os testes de hipótese utilizaram nível de significância de $\alpha = 5\%$.

O gráfico dos resíduos padronizados exibe poucos valores elevados, mas ainda dentro de 2 desvios padrão de distância da média. A FAC, por sua vez, revela que não há nenhuma dependência linear restante nos resíduos do modelo, mesmo que o teste de LjungBox rejeite H_0 . O resultado mais inesperado é o QQPlot que indica que os dados estão bem longe de uma distribuição normal devido à caudas mais leves que o esperado, o que também é percebido ao analisar o histograma.

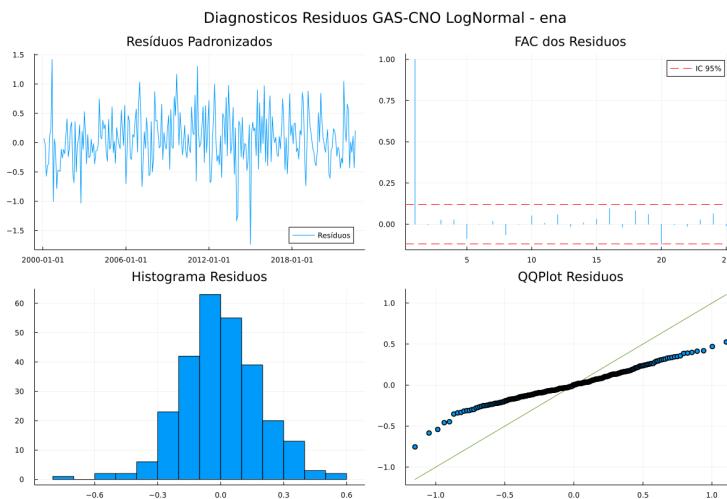


Figura 5.21: Diagnósticos dos resíduos do modelo GAS-CNO LogNormal para série de ENA

Teste	p-valor	Rejeição
Jarque Bera	0.00314	Rejeita H_0
ARCHLM	0.01058	Rejeita H_0
LjungBox	0.02786	Rejeita H_0

Tabela 5.8: Testes de hipótese para resíduos do modelo GAS-CNO LogNormal para série de ENA

Por último, a figura 5.22, junto da tabela 5.9 exibem os resultados associados à previsão fora da amostra do modelo estimado.

Embora a dinâmica da previsão se assemelhe da dinâmica da série fora da amostra, a ordem de grandeza está consideravelmente diferente, dado que a previsão não acompanhou o pico da série de teste. Note que a figura 5.23 indica que o pico da série de teste não é atípico em relação ao histórico da série, o que justificaria a previsão não ter alcançado. É curioso perceber, no entanto, que o MAPE fora da amostra foi melhor que o MAPE *in sample*. Além disso, os valores das métricas de erro estão bem maiores que os da série de carga. Sendo assim, será interessante averiguar se combinação não linear de tendência e sazonalidade, bem como múltiplas variâncias na componente sazonal conseguirão melhorar os resultados deste modelo para a série de ENA.

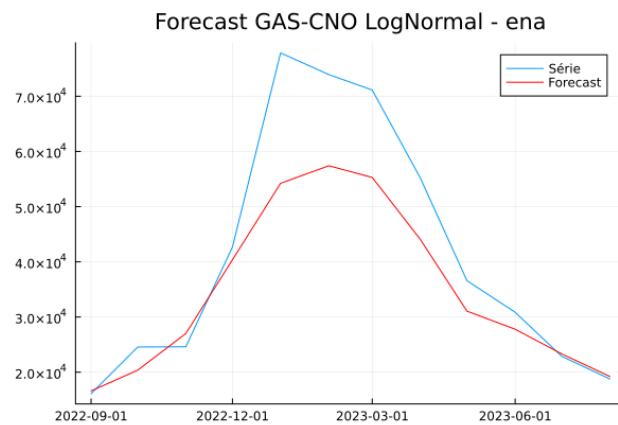


Figura 5.22: Previsão 12 passos à frente do modelo GAS-CNO LogNormal para série de ENA

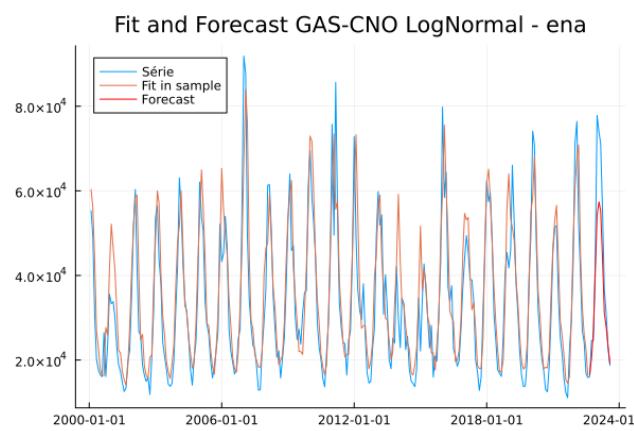


Figura 5.23: Fit in sample e forecast do modelo GAS-CNO LogNormal para série de ENA

MAPE Treino	MAPE Teste
18.57%	13.32%

Tabela 5.9: MAPEs de treino e teste do modelo GAS-CNO LogNormal para série de ENA

5.2.3

Benchmark: GAS-CNO Gamma

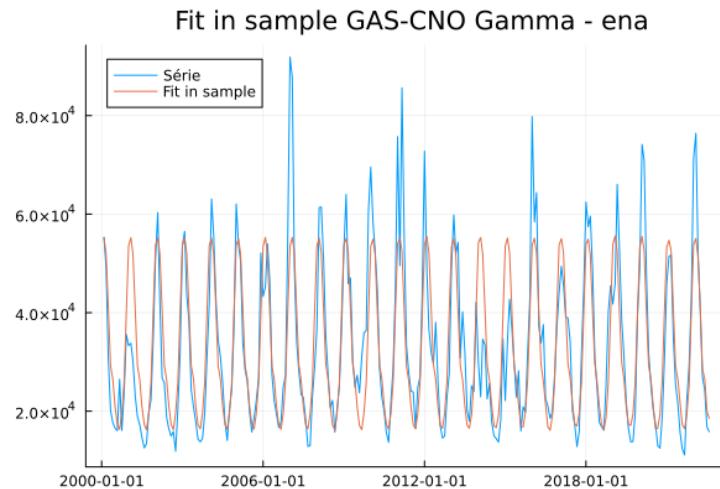


Figura 5.24: Fit in Sample do modelo GAS-CNO Gamma para série de ENA

Componentes GAS-CNO Gamma - ena

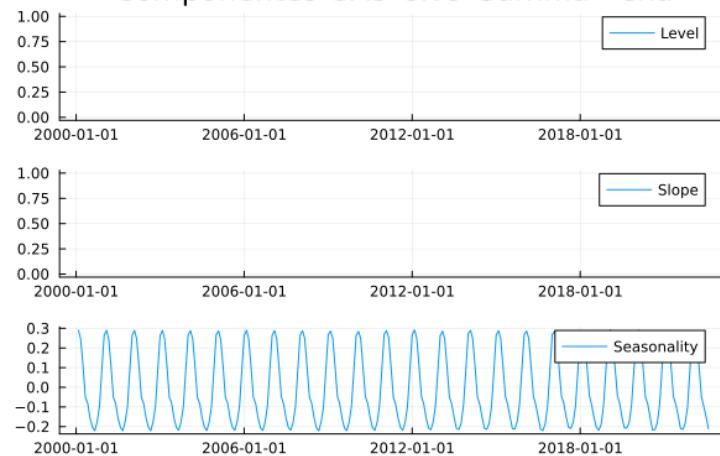


Figura 5.25: Componentes estimadas do modelo GAS-CNO Gamma para série de ENA

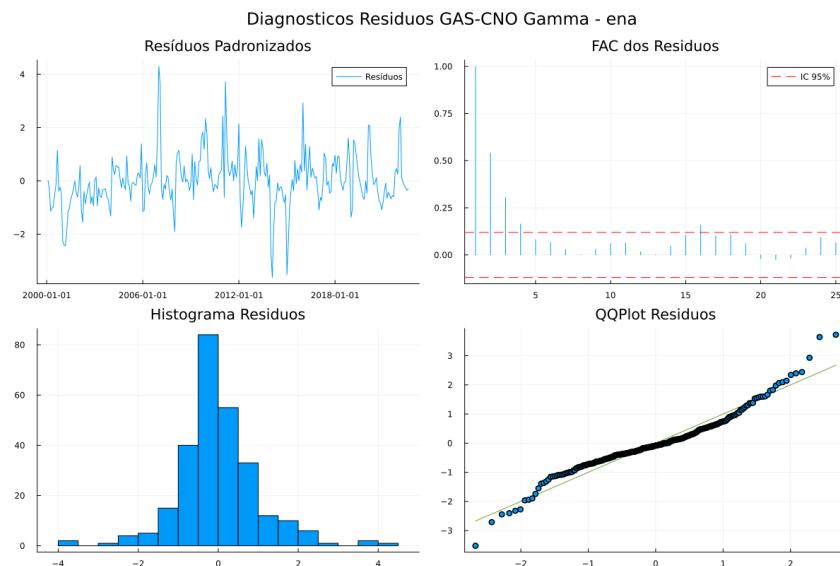


Figura 5.26: Diagnósticos dos resíduos do modelo GAS-CNO Gamma para série de ENA

Teste	p-valor	Rejeição
Jarque Bera	4.41e-27	Rejeita H_0
ARCHLM	1.42e-7	Rejeita H_0
LjungBox	1.89e-26	Rejeita H_0

Tabela 5.10: Testes de hipótese para resíduos do modelo GAS-CNO Gamma para série de ENA

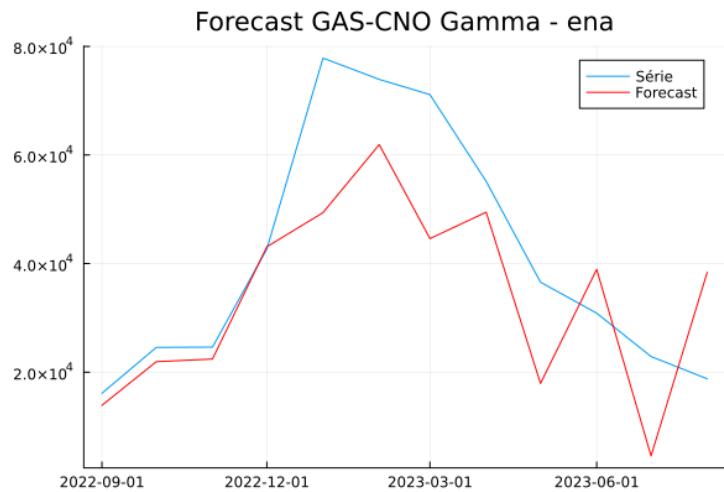


Figura 5.27: Previsão 12 passos à frente do modelo GAS-CNO Gamma para série de ENA

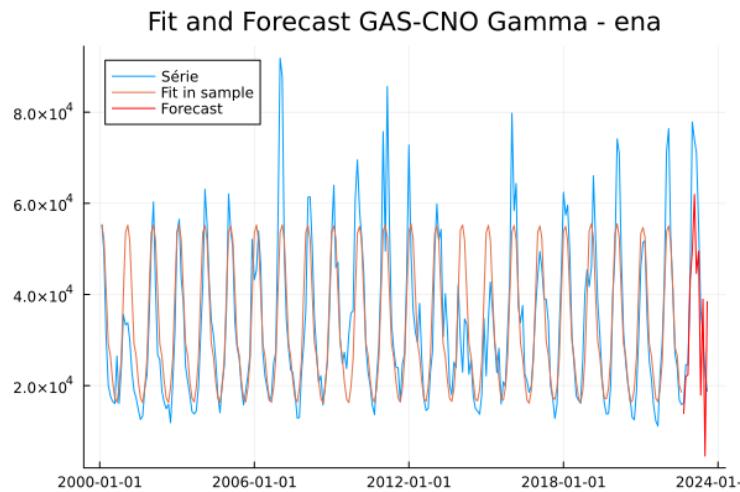


Figura 5.28: Fit in sample e previsão do modelo GAS-CNO Gamma para série de ENA

MAPE Treino	MAPE Teste
19.45%	33.05%

Tabela 5.11: MAPEs de treino e teste do modelo GAS-CNO Gamma para série de ENA

5.2.4

GAS-CNO LogNormal com combinação multiplicativa

5.2.5

GAS-CNO Gamma com combinação multiplicativa

5.2.6**GAS-CNO LogNormal com múltiplas variâncias****5.2.7****GAS-CNO Gamma com múltiplas variâncias****5.2.8****MAPES**

Modelo - ENA	MAPE Treino	MAPE Teste
AutoARIMA	23.02	70.65
GAS-CNO LN	18.57	13.32
GAS-CNO G	19.45	33.05
GAS-CNO LN Mult	mape	mape
GAS-CNO G Mult	mape	mape
GAS-CNO LN MultiVar	mape	mape
GAS-CNO G MultiVar	mape	mape

Tabela 5.12: MAPEs de treino e teste para série de ENA

→ é isso mesmo?

6

Referências bibliográficas

[Alves] ALVES, M. e. a. Unobservedcomponentsgas.jl:. Citado 2 vezes nas páginas 27 e 31.

[Andre e Koopman] ANDRE, L.; KOOPMAN, S. J. *Generalized autoregressive score models*. Disponível em: <<http://www.gasmodel.com/index.htm>>. Citado na página 14.

[Ardia, Boudt e Catania 2016] ARDIA, D.; BOUDT, K.; CATANIA, L. Generalized autoregressive score models in r: The gas package. *arXiv preprint arXiv:1609.02354*, 2016. Disponível em: <<https://cran.r-project.org/web/packages/GAS/GAS.pdf>>. Citado 2 vezes nas páginas 13 e 25.

[Blazsek e Escribano 2023] BLAZSEK, S.; ESCRIBANO, Á. Score-driven threshold ice-age models: benchmark models for long-run climate forecasts. *Energy Economics*, Elsevier, p. 106522, 2023. Citado na página 15.

[Bodin et al. 2020] BODIN, G. et al. Scoredrivenmodels.jl: a julia package for generalized autoregressive score models. *arXiv preprint arXiv:2008.05506*, 2020. Citado 2 vezes nas páginas 13 e 23.

[Bollerslev 1986] BOLLERSLEV, T. Generalized autoregressive conditional heteroskedasticity. *Journal of econometrics*, Elsevier, v. 31, n. 3, p. 307–327, 1986. Citado na página 13.

[Caivano, Harvey e Luati 2016] CAIVANO, M.; HARVEY, A.; LUATI, A. Robust time series models with trend and seasonal components. *SERIEs*, Springer, v. 7, p. 99–120, 2016. Citado 2 vezes nas páginas 30 e 33.

[Cox et al. 1981] COX, D. R. et al. Statistical analysis of time series: Some recent developments [with discussion and reply]. *Scandinavian Journal of Statistics*, JSTOR, p. 93–115, 1981. Citado na página 32.

[Creal, Koopman e Lucas 2013] CREAL, D.; KOOPMAN, S. J.; LUCAS, A. Generalized autoregressive score models with applications. *Journal of Applied Econometrics*, Wiley Online Library, v. 28, n. 5, p. 777–795, 2013. Citado na página 13.

- [Engle e Russell 1998] ENGLE, R. F.; RUSSELL, J. R. Autoregressive conditional duration: a new model for irregularly spaced transaction data. *Econometrica*, JSTOR, p. 1127–1162, 1998. Citado na página 13.
- [Fernandes 2023] FERNANDES, C. *Notas de aula – IV Sazonalidade em em Modelos Estatísticos*. [S.I.]: Departamento de Engenharia Elétrica da PUC-Rio, Grupo de Energia Elétrica do LAMPS, 2023. Citado 2 vezes nas páginas 43 e 70.
- [Fuentes, Herrera e Clements 2023] FUENTES, F.; HERRERA, R.; CLEMENTS, A. Forecasting extreme financial risk: A score-driven approach. *International Journal of Forecasting*, Elsevier, v. 39, n. 2, p. 720–735, 2023. Citado na página 15.
- [Harvey 2013] HARVEY, A. C. *Dynamic models for volatility and heavy tails: with applications to financial and economic time series*. [S.I.]: Cambridge University Press, 2013. v. 52. Citado 3 vezes nas páginas 13, 30 e 33.
- [Harvey 2022] HARVEY, A. C. Score-driven time series models. *Annual Review of Statistics and Its Application*, Annual Reviews, v. 9, p. 321–342, 2022. Citado na página 32.
- [Hoeltgebaum et al. 2021] HOELTGEBAUM, H. et al. A score-driven model of short-term demand forecasting for retail distribution centers. *Journal of Retailing*, Elsevier, v. 97, n. 4, p. 715–725, 2021. Citado na página 15.
- [Hyndman e Athanasopoulos 2021] HYNDMAN, R.; ATHANASOPOULOS, G. *Forecasting: principles and practice*. [S.I.]: OText, 2021. Citado 2 vezes nas páginas 36 e 43.
- [Hyndman e Khandakar 2008] HYNDMAN, R. J.; KHANDAKAR, Y. Automatic time series forecasting: the forecast package for r. *Journal of statistical software*, v. 27, p. 1–22, 2008. Citado na página 17.
- [Lit] LIT, R. *Time Series Lab Manual*. Disponível em: <<https://timeserieslab.com>>. Citado na página 16.
- [Lit, Koopman e Harvey 2021] LIT, R.; KOOPMAN, S.; HARVEY, A. Time series lab - dynamic score edition. 2021. Disponível em: <<https://timeserieslab.com>>. Citado na página 13.
- [Taylor 2016] TAYLOR, R. Pyflux: A time-series analysis library for python. nov. 2016. Disponível em: <<https://pyflux.readthedocs.io/en/latest/index.html>>. Citado na página 13.

[Xu e Lien 2022] XU, Y.; LIEN, D. Forecasting volatilities of oil and gas assets: A comparison of gas, garch, and egarch models. *Journal of Forecasting*, Wiley Online Library, v. 41, n. 2, p. 259–278, 2022. Citado na página 15.

A

Desenvolvimento GAS

Neste capítulo são apresentadas todas as contas relacionadas ao modelo GAS para algumas distribuições de probabilidade.

A.1 GAS gama

Seja $(y_t | \tilde{y}_{t-1}) \sim Gama(\alpha_t, \lambda_t)$, ou seja, uma série cuja distribuição é uma gama com ambos os parâmetros variantes no tempo.

Vale que $E(y_t | \tilde{y}_{t-1}) = \lambda_t$ e $V(y_t | \tilde{y}_{t-1}) = \lambda_t^2 / \alpha_t$.

Para essa parametrização escolhida, segue-se que a função densidade é

$$f(y_t | \tilde{y}_{t-1}; \alpha_t, \lambda_t) = \frac{1}{\Gamma(\alpha_t)} \frac{1}{(\alpha_t^{-1} \lambda_t)^{\alpha_t}} y_t^{\alpha_t - 1} e^{-\frac{\alpha_t}{\lambda_t} y_t}$$

Tomando o logaritmo natural da densidade, obtenho:

$$\ln f(y_t | \tilde{y}_{t-1}; \alpha_t, \lambda_t) = -\ln(\Gamma(\alpha_t)) - \alpha_t \ln(1/\alpha_t) - \alpha_t \ln(\lambda_t) + (\alpha_t + 1) \ln(y_t) - (\alpha_t / \lambda_t) y_t$$

O próximo passo é obter o vetor $\nabla_t = \begin{pmatrix} \nabla_t^\alpha \\ \nabla_t^\lambda \end{pmatrix}$

onde

$$\begin{aligned} \nabla_t^\alpha &= \frac{\partial \ln f(y_t | \tilde{y}_{t-1})}{\partial \alpha_t} = \ln(y_t) - \frac{y_t}{\lambda_t} + \ln(\alpha_t) - \psi_1(\alpha_t) - \ln(\lambda_t) + 1 \\ \nabla_t^\lambda &= \frac{\partial \ln f(y_t | \tilde{y}_{t-1})}{\partial \lambda_t} = \frac{\alpha_t}{\lambda_t} \left(\frac{y_t}{\lambda_t} - 1 \right) \end{aligned}$$

Além disso, foi definido $\psi_1(\alpha) = \frac{\Gamma'(\alpha)}{\Gamma(\alpha)}$, também conhecida como função *digamma*.

É importante ressaltar que $E_{t-1}(\nabla_t^\alpha) = E_{t-1}(\nabla_t^\lambda) = 0$

O próximo passo para a definição de um modelo GAS gama é a obtenção da matriz de informação de Fisher

$$I_{t|t-1} = \begin{pmatrix} I_{t|t-1}^{\alpha\alpha} & I_{t|t-1}^{\alpha\lambda} \\ I_{t|t-1}^{\alpha\lambda} & I_{t|t-1}^{\lambda\lambda} \end{pmatrix}$$

Onde, cada elemento dessa matriz, é definido como

$$\begin{aligned}
I_{t|t-1}^\alpha &= E_{t-1}(\nabla_t^\alpha \nabla_t'^\alpha) = E_{t-1}(\nabla_t^\alpha)^2 = -E_{t-1} \frac{\partial}{\partial \alpha_t} (\nabla_t^\alpha) \\
I_{t|t-1}^\lambda &= E_{t-1}(\nabla_t^\lambda \nabla_t'^\lambda) = E_{t-1}(\nabla_t^\lambda)^2 = -E_{t-1} \frac{\partial}{\partial \lambda} (\nabla_t^\lambda) \\
I_{t|t-1}^{\alpha,\lambda} &= E_{t-1}(\nabla_t^\alpha \nabla_t'^\lambda) = -E_{t-1} \frac{\partial}{\partial \lambda} (\nabla_t^\alpha) = -E_{t-1} \frac{\partial}{\partial \alpha} (\nabla_t^\lambda)
\end{aligned}$$

Agora, desenvolvendo cada um desses termos individualmente, obtenho

$$\begin{aligned}
I_{t|t-1}^\alpha &= E_{t-1} \frac{\partial}{\partial \alpha_t} \left[\ln(y_t) - \frac{y_t}{\lambda_t} + \ln(\alpha_t) - \psi_1(\alpha_t) - \ln(\lambda_t) + 1 \right] = \dots = \\
&= \psi_2(\alpha_t) - \frac{1}{\alpha_t} \\
I_{t|t-1}^\lambda &= E_{t-1} \frac{\partial}{\partial \lambda_t} \left[\frac{\alpha_t y_t}{\lambda_t^2} - \frac{\alpha_t}{\lambda_t} \right] = -E_{t-1} \left[\frac{-2\alpha_t y_t}{\lambda_t^3} + \frac{\alpha_t}{\lambda_t^2} \right] = \dots = \\
&= \frac{\alpha_t}{\lambda_t^2} \\
I_{t|t-1}^{\alpha,\lambda} &= -E_{t-1} \frac{\partial}{\partial \alpha_t} \left[\frac{\alpha_t y_t}{\lambda_t^2} - \frac{\alpha_t}{\lambda_t} \right] = -E_{t-1} \left[\frac{\alpha_t y_t}{\lambda_t^2} - \frac{\alpha_t}{\lambda_t} \right] = \dots = \\
&= 0
\end{aligned}$$

Portanto, a matriz de informação de Fisher $I_{t|t-1}$ para uma distribuição gama de parâmetros α_t e λ_t é

$$I_{t|t-1} = \begin{pmatrix} \psi_2(\alpha_t) - \frac{1}{\alpha_t} & 0 \\ 0 & \frac{\alpha_t}{\lambda_t^2} \end{pmatrix}$$

onde a função $\psi_2(\alpha_t)$ é a função *trigamma* definida como $\psi_2(\alpha_t) = \frac{\psi'_1(\alpha_t)}{\psi_1(\alpha_t)}$

Com isso, estão definidos todos os resultados necessários para estimar um modelo GAS, tanto com dinâmica ARMA quanto CNO, cuja distribuição condicional é $gama(\alpha_t, \lambda_t)$.

B

Sazonalidade em modelos de espaço de estados

Neste apêndice, está descrito o processo de modelagem de sazonalidade em modelos de espaço de estados, começando pelo tratamento de sazonalidade determinística até sua versão estocástica. O entendimento da modelagem da componente sazonal dentro dessa classe de modelos é fundamental para compreender a modelagem de sazonalidade em modelos *score-driven* cuja dinâmica é modelada via arcabouço CNO. A maior parte do desenvolvimento aqui apresentado encontra-se em [Fernandes 2023].

Primeiramente, é importante definir sazonalidade como a representação de flutuações periódicas associadas a eventos climáticos, como as estações do ano, eventos culturais, como feriados e festas, bem como datas administrativas, como períodos letivos. Vale comentar que a boa modelagem da sazonalidade é essencial para garantir a qualidade da estimativa de um modelo de série temporal (quando esta apresenta sazonalidade), dado o seu caráter repetitivo.

Um conceito importante a ser definido é o período sazonal, que representa o tempo em que a flutuação periódica leva para se repetir e é descrito pela letra s . Além disso, sazonalidade é definida apenas para flutuações de, no máximo, 1 ano. Para períodos maiores que este existe o conceito de ciclo que não será abordado nesse apêndice. Exemplos de períodos sazonais seriam $s = 4$ para séries trimestrais, $s = 12$ para séries mensais e $s = 52$ para séries semanais. Embora seja possível que uma série possua mais de uma componente sazonal, como é o caso de séries diárias que podem possuir sazonalidade semanal e anual, esse assunto também não será abordado nesse apêndice, dado que o escopo do projeto se limita a séries mensais.

Para começar o desenvolvimento da modelagem de sazonalidade em série temporais, suponha que uma dada série y_t possa ser decomposta em componentes de tendência μ_t e sazonalidade γ_t como se segue:

$$y_t = \mu_t + \gamma_t + \epsilon_t \quad \epsilon_t \sim N(0, \sigma^2) \quad (\text{B-1})$$

A equação B-1 descreve um modelo cuja componente sazonal se combina de forma aditiva e é o padrão para modelos lineares. Também é possível haver sazonalidade multiplicativa em modelos lineares. Para isso, basta que o modelo seja ajustado para log da série. Esse modelo seria descrito como:

$$y_t = \mu_t \cdot \gamma_t \cdot \epsilon_t \quad \epsilon_t \sim N(0, \sigma^2) \quad (\text{B-2})$$

Sabe-se que, dentro do arcabouço de modelos de espaço de estados, a

componente de tendência μ_t pode ser definida da seguinte maneira (modelo estrutural básico):

$$\begin{aligned}y_t &= \mu_t + \gamma_t + \epsilon_t, \quad \epsilon_t \sim N(0, \sigma^2) \\ \mu_{t+1|t} &= \mu_t + \beta_t + \eta_t, \quad \eta_t \sim N(0, \sigma_\eta^2) \\ \beta_{t+1|t} &= \beta_t + \xi_t, \quad \xi_t \sim N(0, \sigma_\xi^2)\end{aligned}$$

O objetivo desse apêndice é adicionar a formulação para a componente sazonal γ_t nesse modelo. Para isso, primeiramente será apresentada a componente sazonal determinística utilizando variáveis *dummies* e funções trigonométricas para, em seguida, apresentar a formulação da sazonalidade estocástica.

Sazonalidade por variáveis *dummies*

O tratamento da sazonalidade por variáveis *dummies* é o mais simples de ser implementado, além de trazer uma interpretação direta de cada coeficiente estimado.

Suponha uma série y_t estacionária trimetral, o que implica que s_4 . Poderíamos utilizar um modelo como:

$$\begin{aligned}y_t &= \beta + \gamma_1 D_{1,t} + \gamma_2 D_{2,t} + \gamma_3 D_{3,t} + \gamma_4 D_{4,t} + \epsilon_t, \quad t = 1, \dots, T \\ D_{i,t} &= 1, \quad i = t, \quad i = 1, 2, 3, 4 \\ D_{i,t} &= 0 \quad c.c.\end{aligned}$$

Note que cada variável *dummie* $D_{i,t}$ indica se a observação daquele trimestre t está no trimestre i . Embora esse modelo capture a periodicidade trimestral, ele apresenta o problema de multicolinearidade perfeita, dado que um dos regressores ($D_{4,t}$, por exemplo) pode ser obtido a partir da combinação linear dos demais.

Para resolver esse problema, existem 3 parametrizações possíveis.

A primeira baseia-se em abandonar uma das *dummies* do modelo. Suponha, por exemplo, que abandonemos a *dummie* associada ao quarto trimestre. O período que foi abandonado é chamado de período basal. Com isso, obteríamos o modelo:

$$\begin{aligned}y_t &= \beta + \gamma_1 D_{1,t} + \gamma_2 D_{2,t} + \gamma_3 D_{3,t} + \epsilon_t, \quad t = 1, \dots, T \\ D_{i,t} &= 1, \quad t = i, i + s, i + 2s, \dots, \quad i = 1, 2, 3 \\ D_{i,t} &= 0 \quad t \neq i, i + s, i + 2s, \dots\end{aligned}$$

Essa parametrização traz uma interpretação direta para os coeficientes do modelo, que advém do valor esperado de y_t para cada trimestre, como se segue

abaixo.

$$\begin{aligned} E(y_t | D_{i,t} = 0, i = 1, 2, 3) &= \beta \\ E(y_t | D_{j,t} = 1, D_{i,t} = 0, j \neq i, i = 1, 2, 3) &= \beta + \gamma_j \end{aligned}$$

Portanto, $\hat{\beta} = \bar{y}_4$, ou seja, o intercepto representa a média do trimestre basal e $\gamma_j = \bar{y}_j - \bar{y}_4$, isto é, os coeficientes sazonais são o desvio da média de cada trimestre em relação ao trimestre basal.

A segunda parametrização opta, por sua vez, em abandonar o intercepto e manter todas as *dummies* sazonais, obtendo o modelo:

$$\begin{aligned} y_t &= \delta_1 D_{1,t} + \delta_2 D_{2,t} + \delta_3 D_{3,t} + \delta_4 D_{4,t} + \epsilon_t, \quad t = 1, \dots, T \\ D_{j,t} &= 1, \quad t = j, i+s, i+2s, \dots, \quad j = 1, 2, 3 \\ D_{j,t} &= 0 \quad t \neq j, j+s, j+2s, \dots \end{aligned}$$

Novamente, a interpretação de cada coeficiente sazonal é direta, mas, dessa vez, $\hat{\delta}_j = \bar{y}_j$, que significa que a estimativa de cada coeficiente é a média daquele trimestre. Essa interpretação vem do fato de que $E(y_t | D_{j,t}) = \delta_j$.

A terceira e última parametrização opta por manter todas as *dummies* no modelo, mas, para evitar a multicolinearidade, introduz uma restrição nos coeficientes sazonais de tal forma que a soma dos fatores sazonais deve ser zero no período sazonal. Isso significa que:

$$\sum_{j=1}^s \theta_j = 0 \implies \theta_s = -\sum_{j=1}^{s-1} \theta_j$$

A partir dessa restrição, obtemos o modelo:

$$\begin{aligned} y_t &= \alpha + \sum_{j=1}^{s-1} \theta_j D_{j,t} + \epsilon_t, \quad t = 1, \dots, T \\ D_{j,t} &= 1, t = j, j+s, j+2s, \dots \\ D_{j,t} &= 0, t \neq j, j+s, j+2s, \dots \\ D_{j,t} &= -1, t = s, 2s, 3s, \dots \end{aligned}$$

Para conseguir obter uma interpretação útil dos coeficientes sazonais, note

que

$$\begin{aligned} E(y_t|D_{j,t}) &= \alpha + \theta_j \implies \theta_j = E(y_t|D_{j,t}) - \alpha, \quad j = 1, 2, 3 \\ E(y_t|D_{4,t}) &= \alpha - (\theta_1 + \theta_2 + \theta_3) = \alpha + \theta_4 \implies \theta_4 = E(y_t|D_{4,t}) - \alpha \end{aligned}$$

Dada a restrição imposta inicialmente, posso escrever que:

$$0 = \theta_1 + \theta_2 + \theta_3 + \theta_4 = \sum_{j=1}^4 E(y_t|D_{j,t}) - 4\alpha$$

Logo,

$$\alpha = \frac{1}{4} \sum_{j=1}^4 E(y_t|D_{j,t}) \implies \alpha = \frac{1}{4} \sum_{j=1}^4 \bar{y}_j$$

Portanto, se os trimestres forem balanceados, $\hat{\alpha} = \bar{y}$, que é a média da série. Com isso, $\hat{\theta}_j$ serão os desvios das médias dos trimestres em relação à média da série.

A partir da modelagem por *dummies* com a parametrização 3, podemos completar a definição do modelo estrutural básico que foi apresentado no início desse apêndice. Mantendo as componentes de tendência e nível estocástica, mas a nova componente de sazonalidade determinística, obtenho:

$$\begin{aligned} y_t &= \mu_t + \gamma_t + \epsilon_t, \quad \epsilon_t \sim N(0, \sigma^2) \\ \mu_{t+1|t} &= \mu_t + \beta_t + \eta_t, \quad \eta_t \sim N(0, \sigma_\eta^2) \\ \beta_{t+1|t} &= \beta_t + \xi_t, \quad \xi_t \sim N(0, \sigma_\xi^2) \\ \gamma_t &= - \sum_{j=1}^{s-1} \gamma_{t-j} \end{aligned}$$

Note que a componente sazonal já está escrita em uma forma recursiva, então, para modificá-la para uma versão estocástica, basta somar um choque aleatório em sua equação de estado. Com isso, finalmente obtemos o modelo estrutural básico com sazonalidade modelada a partir de variáveis *dummies*.

$$\begin{aligned} y_t &= \mu_t + \gamma_t + \epsilon_t, \quad \epsilon_t \sim N(0, \sigma^2) \\ \mu_{t+1|t} &= \mu_t + \beta_t + \eta_t, \quad \eta_t \sim N(0, \sigma_\eta^2) \\ \beta_{t+1|t} &= \beta_t + \xi_t, \quad \xi_t \sim N(0, \sigma_\xi^2) \\ \gamma_t &= - \sum_{j=1}^{s-1} \gamma_{t-j} + \omega_t, \quad \omega_t \sim N(0, \sigma_\omega^2) \end{aligned}$$

É importante para o escopo deste projeto ressaltar que, normalmente, todos os trimestres da série temporal possuem uma mesma variância da componente

sazonal. Um dos objetivos deste projeto é, justamente, generalizar essa variância de forma que diferentes trimestres, ou épocas do ano de forma geral, possuam variâncias diferentes, ou seja $\sigma_\omega^2 \rightarrow \sigma_{\omega,t}^2$

Sazonalidade por funções trigonométricas

Além de modelar a sazonalidade de uma série temporal a partir de variáveis *dummies* que indicam em qual período uma dada observação se encontra, também é possível modelar o efeito sazonal a partir de uma combinação de funções trigonométricas. Começando, novamente, pela versão determinística, podemos definir a dinâmica da componente sazonal como:

$$\gamma_t = \sum_{j=1}^{[s/2]} (\gamma_j \cos(\lambda_j t) + \gamma_j^* \sin(\lambda_j t)) \quad (\text{B-3})$$

onde $\lambda_j = 2\pi j/s, j = 1, \dots, [s/2]$ são os harmônicos, s é o período sazonal da série e $[s/2] = s/2, s$ par ou $[s/2] = (s-1)/2, s$ ímpar.

É importante ressaltar que $\gamma_j^* = 0$ se $j = [s/2]$. Além disso, diferente da sazonalidade por *dummies*, o coeficiente sazonal não mais o fator sazonal em si. Agora, os coeficientes sazonais são γ_j, γ_j^* , enquanto que o fator sazonal é γ_t .

Uma vez definida a versão determinística da sazonalidade por funções trigonométricas, falta obter sua versão estocástica. Para isso, primeiro vou obter uma forma recursiva da equação e, em seguida, adicionar um choque aleatório.

Antes de começar esse desenvolvimento, atente-se para a notação possivelmente confusa, assumindo série mensal: γ_t é a sazonalidade do mês t , (γ_j, γ_j^*) são os coeficientes trigonométricos e $(\gamma_{j,t}, \gamma_{j,t}^*)$ são os coeficientes estocásticos.

O primeiro passo é reescrever a equação B-3 como:

$$\gamma_t = \sum_{j=1}^{[s/2]} (\gamma_j \cos(\lambda_j t) + \gamma_j^* \sin(\lambda_j t)) = \sum_{j=1}^{[s/2]} \gamma_{j,t} \quad (\text{B-4})$$

Note que

$$\gamma_{j,t} = \begin{cases} \gamma_j \cos(\lambda_j t) + \gamma_j^* \sin(\lambda_j t), & \text{se } j = 1, \dots, (s/2) - 1 \\ \gamma_{s/2} \cos \lambda_{s/2} t, & \text{se } j = s/2 \end{cases} \quad (\text{B-5})$$

Para obter a relação recursiva, trocaremos t por $t+1$ na expressão acima e desenvolveremos as contas.

$$\begin{aligned}
\gamma_{j,t+1} &= \gamma_j \cos(\lambda_j(t+1)) + \gamma_j^* \sin(\lambda_j(t+1)) \\
\gamma_{j,t+1} &= \gamma_j \cos(\lambda_j t + \lambda_j) + \gamma_j^* \sin(\lambda_j t + \lambda_j) \\
\gamma_{j,t+1} &= \gamma_j [\cos(\lambda_j t) \cos(\lambda_j) - \sin(\lambda_j t) \sin(\lambda_j)] + \\
&\quad + \gamma_j^* [\sin(\lambda_j t) \cos(\lambda_j) + \cos(\lambda_j t) \sin(\lambda_j)] \\
\gamma_{j,t+1} &= \cos \lambda_j [\gamma_j \cos(\lambda_j t) + \gamma_j^* \sin(\lambda_j t)] + \\
&\quad + \sin \lambda_j [-\gamma_j \sin(\lambda_j t) + \gamma_j^* \cos(\lambda_j t)]
\end{aligned}$$

Pode-se perceber que, dentro dos colchetes, obtivemos, justamente, as expressões para $\gamma_{j,t}$ e $\gamma_{j,t}^*$. Reescrevendo a equação acima em forma vetorial obtenho:

$$\gamma_{j,t+1} = \begin{bmatrix} \cos \lambda_j & \sin \lambda_j \end{bmatrix} \begin{bmatrix} \gamma_{j,t} \\ \gamma_{j,t}^* \end{bmatrix}$$

Com essa equação, temos a recursão de $\gamma_{j,t+1}$ com $\gamma_{j,t}$, mas falta ainda obter a recursão para $\gamma_{j,t}^*$. Para isso, basta seguir o mesmo procedimento:

$$\begin{aligned}
\gamma_{j,t+1}^* &= \gamma_j \cos^*(\lambda_j(t+1)) - \gamma_j \sin(\lambda_j(t+1)) \\
\gamma_{j,t+1}^* &= -\gamma_j [\sin(\lambda_j t) \cos(\lambda_j) + \sin(\lambda_j) \cos(\lambda_j t)] + \\
&\quad + \gamma_j^* [\cos(\lambda_j t) \cos(\lambda_j) - \sin(\lambda_j t) \sin(\lambda_j)] \\
\gamma_{j,t+1}^* &= \begin{bmatrix} -\sin \lambda_j & \cos \lambda_j \end{bmatrix} \begin{bmatrix} \gamma_j \cos(\lambda_j t) + \gamma_j^* \sin(\lambda_j t) \\ \gamma_j^* \cos(\lambda_j t) + \gamma_j \sin(\lambda_j t) \end{bmatrix} \\
\gamma_{j,t+1}^* &= \begin{bmatrix} -\sin \lambda_j & \cos \lambda_j \end{bmatrix} \begin{bmatrix} \gamma_{j,t} \\ \gamma_{j,t}^* \end{bmatrix}
\end{aligned}$$

Juntando as duas equações, obtenho finalmente a relação recursiva desejada:

$$\begin{bmatrix} \gamma_{j,t+1} \\ \gamma_{j,t+1}^* \end{bmatrix} = \begin{bmatrix} \cos \lambda_j & \sin \lambda_j \\ -\sin \lambda_j & \cos \lambda_j \end{bmatrix} \begin{bmatrix} \gamma_{j,t} \\ \gamma_{j,t}^* \end{bmatrix}$$

Assim como foi o caso para o tratamento da sazonalidade via variáveis *dummies*, para tornar essa componente estocástica basta adicionar choques aleatórios na equação.

$$\begin{bmatrix} \gamma_{j,t+1} \\ \gamma_{j,t+1}^* \end{bmatrix} = \begin{bmatrix} \cos \lambda_j & \sin \lambda_j \\ -\sin \lambda_j & \cos \lambda_j \end{bmatrix} \begin{bmatrix} \gamma_{j,t} \\ \gamma_{j,t}^* \end{bmatrix} + \begin{bmatrix} \omega_{j,t} \\ \omega_{j,t}^* \end{bmatrix}$$

onde $\omega_{j,t} \sim N(0, \sigma_\omega^2)$ e $\omega_{j,t}^* \sim N(0, \sigma_\omega^2)$.

É possível simplificar o modelo utilizando o mesmo choque para todas as

componentes sazonais, de tal forma que :

$$\begin{bmatrix} \gamma_{j,t+1} \\ \gamma_{j,t+1}^* \end{bmatrix} = \begin{bmatrix} \cos \lambda_j & \sin \lambda_j \\ -\sin \lambda_j & \cos \lambda_j \end{bmatrix} \begin{bmatrix} \gamma_{j,t} \\ \gamma_{j,t}^* \end{bmatrix} + \begin{bmatrix} \omega_{j,t} \\ \omega_{j,t} \end{bmatrix}$$

Além disso, note que a variância dos choques da componente sazonal são constantes no tempo e, tal qual foi comentado para o caso do tratamento via variáveis *dummies*, um dos objetivos desse projeto é generalizar $\sigma_\omega^2 \rightarrow \sigma_{\omega,t}^2$.