ESTUDO DE PROCESSOS DE FILTRAGEM DE CURVAS DE LUZ OBTIDAS COM OS SATÉLITES COROT E KEPLER

Guilherme Samuel de Souza Barbosa [1]; Roberto Bertoldo Menezes [2]

[1] Aluno de Iniciação Científica da Escola de Engenharia Mauá (EEM/CEUN-IMT);
[2] Professor da Escola de Engenharia Mauá (EEM/CEUN-IMT).

Resumo. *Curvas de luz são dados astronômicos utilizados para o estudo da variabilidade do brilho das estrelas, o que pode resultar na descoberta de exoplanetas e de binárias eclipsantes, devido aos eclipses gerados por esses corpos e detectados nas curvas de luz. No entanto, um problema conhecido das curvas de luz é a presença de ruído de alta frequência, que pode dificultar tanto a detecção dos eclipses quanto a modelagem das curvas de luz visando a determinação de parâmetros dos exoplanetas. Dessa forma, nesse trabalho, usando dados obtidos com o telescópio espacial CoRoT, avaliamos quais as melhores técnicas de filtragem, capazes de remover o ruído de alta frequência, sem comprometer as futuras análises a serem aplicadas, envolvendo exoplanetas. Ao final, verificamos que os filtros de Butterworth e de Bessel, com ordens e frequências de corte específicas, foram os que proporcionaram as maiores melhorias (superiores a 30%) na precisão dos parâmetros determinados para os exoplanetas, sem comprometer as curvas de luz correspondentes.*

Introdução

Curvas de luz são dados astronômicos frequentemente utilizados para o estudo da variabilidade do brilho das estrelas e consistem essencialmente em gráficos do brilho observado das estrelas em função do tempo. Uma análise detalhada desses dados permite detectar, entre outros fenômenos, eclipses, que ocorrem quando um corpo transita entre a estrela observada e o observador, resultando em uma diminuição do brilho estelar, detectado nas curvas de luz. Esses eclipses podem ser causados por exoplanetas, planetas que orbitam estrelas além do Sol, ou por binárias eclipsantes, par de estrelas que orbitam o centro de massa em comum. No caso de curvas de luz contendo exoplanetas, uma modelagem teórica pode ser aplicada visando a determinação de alguns de seus parâmetros, como os raios e as distâncias em relação às estrelas que orbitam (e.g. Mandel & Agol, 2002).

Observações feitas com telescópios espaciais forneceram curvas de luz para um número consideravelmente elevado de estrelas, o que resultou na identificação de milhares de exoplanetas e de binárias eclipsantes. Os telescópios espaciais CoRoT (Convection, Rotation and planetary Transits – Auvergne et al. 2009), lançado em 27 de dezembro de 2006, e o Kepler (Koch et al. 2010), lançado em 7 de março de 2009, que já não estão mais ativos, possuíam como objetivo em comum para a ciência a procura por exoplanetas. Juntos, os telescópios observaram e produziram curvas de luz de mais de 360000 estrelas. Ao final, o satélite CoRoT detectou 34 exoplanetas e 2269 binárias eclipsantes (Deleuil, 2018), enquanto que o satélite Kepler detectou 2335 exoplanetas e mais de 2400 binárias eclipsantes (Slawson, 2011; LaCourse, 2015).

Um problema conhecido das curvas de luz está associado ao ruído de alta frequência, que pode dificultar tanto a detecção de eclipses causados por exoplanetas ou por binárias eclipsantes quanto a modelagem teórica desses dados. Por conta disso, métodos de filtragem são comumente aplicados, visando a remoção desse ruído. Estudos preliminares realizados com curvas de luz revelaram que a filtragem de Butterworth, por exemplo, com ordem e frequência de corte específicas, resulta em incertezas cerca de 40% menores para os parâmetros dos exoplanetas, que permanecem compatíveis aos originais. Dessa forma, o objetivo principal desse trabalho foi obter estimativas mais detalhadas quanto a eficácia de diferentes técnicas de filtragem aplicadas às curvas de luz.

Material e Métodos

Coletamos curvas de luz, geradas pelo satélite CoRoT, contendo exoplanetas e pré-processamos essas séries temporais, reamostrando as curvas e mantendo apenas aquelas que possuíam uma determinada quantidade de observações. À essas curvas de luz pré-processadas, aplicamos diferentes filtros passa-baixas, o que gerou um novo conjunto de dados. Em seguida, computamos certos parâmetros dos exoplanetas das curvas não-filtradas e das filtradas, de modo que foi possível comparar os resultados obtidos e avaliar a eficácia da filtragem em melhorar a qualidade dos dados, sem comprometer os mesmos.

Aquisição e pré-processamento dos dados

Os dados do CoRoT foram obtidos a partir do *website CoRoT IAS Archive (http://idoc-corot.ias.u-psud.fr/)*, do qual foi feito o *download* das 34 curvas de luz contendo exoplanetas. Os dados adquiridos foram encontrados no formado *FITS* (*Flexible Image Transport System)*, bastante utilizado na astronomia, que corresponde a um arquivo com múltiplas extensões, cada uma das quais contendo informações a respeito daquele corpo de estudo. No caso das curvas de luz, foram encontradas as extensões: *PRIMARY*, *BAR*, *BARFILL* e *SYSTEMATIC* e, como estamos interessados em trabalhar apenas com as séries temporais propriamente ditas, mantivemos apenas a extensão *BARFILL*, pois contém informações a respeito das datas nas quais a amostra foi obtida (coluna DATEBARTT), do brilho estelar mensurado (coluna WHITEFLUXSYS) e uma coluna contendo *flags* de *status* da curva, que foi descartada. Informações detalhadas a respeito de cada extensão que o arquivo *FITS* armazena sobre curvas de luz obtidas pelo CoRoT pode ser encontrado na literatura (S. Chaintreuil, A. Deru1, F. Baudin *et. al*. 2016).

Posteriormente, foi utilizada a linguagem Python para transformar esses dados em *DataFrames* – estrutura de dados oriunda do pacote *Pandas*, contendo as colunas correspondentes à data da observação e ao valor do fluxo de luz branca medido. Em seguida, cada curva de luz foi salva em arquivos *CSV*s, pois julgamos os mais adequados e simples para serem operados.

Após essa transformação nos dados "brutos", observamos que havia uma curva de luz com poucos pontos de observação, isto é, menos de 584 valores de fluxo medidos, o que resultou em uma curva com pouca quantidade de eclipses, e isso potencialmente atrapalharia a modelagem teórica a ser aplicada futuramente. Dessa forma, optou-se por excluir essa curva e manter apenas as 33 curvas com mais de 584 pontos.

Além disso, verificou-se que havia curvas de luz com frequências de amostragem diferentes e, consequentemente, tamanhos diferentes. Visando, posteriormente, aplicar técnicas de *machine learning* a esses dados a fim de avaliar o efeito dos diferentes processos de filtragem em sua classificação, foi feita a sua reamostragem para o número mediano de pontos dos dados restantes: 15050. A reamostragem foi feita utilizando o método *resample* do pacote *Scipy.signal*, da linguagem Python, que implementa o método de Fourier para reamostragem. O método de Fourier para reamostrar um sinal X, de tamanho N, se divide em dois casos: aumentar ou diminui a amostragem. Para aumentar, o método consiste em transformar o sinal real para o domínio de frequências e adicionar $N/2$ zeros no seu fim, ou seja, aplicar o *zero padding* ao sinal. Depois é tomada a transformada inversa de Fourier do sinal reamostrado; para diminuir a amostragem, o método também transforma o sinal real para o domínio de frequências e então exclui o segundo e o terceiro grupo de $N/4$ elementos (que corresponde à metade com as componentes de maior frequência). Em seguida, o sinal é transformado para o domínio temporal.

Assim sendo, os dados pré-processados foram salvos em arquivos *CSV*s e então iniciaram-se os procedimentos de filtragem.

Filtragem das curvas de luz

A aplicação de filtros em sinais tem como objetivo reduzir uma certa parcela de ruído. No caso das curvas de luz, sabe-se que esse ruído é de alta frequência e, para o amortecermos, deve-se aplicar filtros passa-baixa. Um filtro passa-baixa, em termos de eletrônica, corresponde a um circuito que permite a passagem de baixas frequências e atenua a amplitude das frequências maiores que uma constante determinada, a chamada frequência de corte. Sobre esse circuito eletrônico, pode-se computar sua função de transferência, isto é, a relação matemática entre a entrada e a saída desse sistema físico, e essa relação será utilizada para definir as diferentes técnicas de filtragens aqui utilizadas.

Nesse trabalho, foram estudados os filtros passa-baixa Ideal, Butterworth, gaussiano, Bessel e mediano. Os quatro primeiros filtros são aplicados a partir do seguinte procedimento: primeiramente, é calculada a transformada de Fourier dos dados tratados, passando-os para o domínio de frequências. Em seguida, a transformada de Fourier é multiplicada pelo filtro a ser utilizado. Após isso, é calculada a transformada de Fourier inversa do resultado, da qual é tomada apenas a parte real. Dessa forma, temos que a equação geral para filtragem no domínio de frequências é dada por

$$g(x) = \{real[\mathcal{F}^{-1}(\, F(u)H(u)\,)]\} \tag{1}$$

em que $g(x)$ é o sinal de saída (filtrado), $\mathcal{F}^{-1}$ é a transformada inversa de Fourier, $F(u)$ é a transformada de Fourier do sinal de entrada $f(x)$ e $H(u)$ é a função de transferência discreta do filtro. $F(u)$, $H(u)$ e $g(x)$ são *arrays* de tamanho M, o mesmo do sinal original $f(x)$. Portanto, filtrar no domínio de frequências consiste essencialmente em modificar a transformada de Fourier do sinal de entrada, nesse caso das curvas de luz.

Para computar a transformada de Fourier dos dados, utilizamos o algoritmo *Fast Fourier Transform* (FFT), visando converter um sinal digital do domínio temporal para o domínio de frequências. No entanto, a FFT é baseada no método de dobramento sucessivo (Gonzales e Woods, 1992), ou seja, assume que o sinal de entrada é periódico e que o número de amostras é inteiro e potência de 2, mas a segunda premissa não é um requisito geral. A primeira conjectura, porém, é bastante relevante para que não haja distorções em $g(x)$. Coletar X amostras de um sinal, o que o CoRoT realizou, é equivalente a fatiar um pedaço finito de tempo, entre T_1 e T_2 de um sinal real, de domínio $-\infty$ a $+\infty$. Existe a possibilidade de que essa janela represente perfeitamente as oscilações do sinal original, que é conter os períodos completos de todas as faixas de frequência, o que resulta em uma FFT sem distorções, mas é estatisticamente improvável que isso ocorra. Nesse caso, temos períodos incompletos que vão introduzir distorções ao sinal que, em geral, são encontrados em suas "bordas".

A razão pela qual essas distorções são comumente encontradas nas "bordas" do sinal é justificada pela forma como a FFT opera: seus cálculos recursivos implicam em unir o final do sinal com o seu início (pela premissa da FFT, temos períodos completos), e é nessa conexão que as deformidades são introduzidas. Então, para reduzir esse efeito, foi aplicado o método de expansão das "bordas" das curvas de luz. Isso é, dado uma curva de luz A, de tamanho k, o procedimento consiste em adicionar N pontos antes do primeiro termo, com o valor do primeiro termo, e depois do último termo, com o valor do último termo, de A. Finalmente, teremos uma curva $\bar{A}$, de tamanho $(k + 2N)$, com as bordas expandidas. Esse método resolve o problema descrito pois a estratégia aqui adotada foi: calcula-se a FFT da curva de luz com a borda expandida, realiza-se os demais procedimentos e, quando a curva estiver no domínio temporal de novo, retira-se essas bordas potencialmente distorcidas.

Outro ponto de atenção ao se aplicar a equação (1) é o produto $H(u)F(u)$, dado pela multiplicação de arranjos matriciais entre as funções em questão. Para exemplificar o funcionamento dessa operação, considere os seguintes vetores $4x1$:

$$[a_1 \;\; a_2 \;\; a_3 \;\; a_4] \;\; e \;\; [b_1 \;\; b_2 \;\; b_3 \;\; b_4] \tag{2}$$

O produto de arranjos matriciais é dado por:

$$[a_1 \;\; a_2 \;\; a_3 \;\; a_4][b_1 \;\; b_2 \;\; b_3 \;\; b_4] = [a_1 b_1 \;\; a_2 b_2 \;\; a_3 b_3 \;\; a_4 b_4] \quad (3)$$

Contudo, o produto de duas funções no domínio de frequências, $H(u)F(u)$, implica, pelo teorema da convolução, na convolução dessas funções no domínio temporal. O teorema da convolução unidimensional, no caso discreto, nos diz que:

$$\mathcal{F}\{f \star g\} = k.\mathcal{F}\{f\}.\mathcal{F}\{h\} \quad (4)$$

onde $\mathcal{F}$ é a transformada de Fourier, k é uma constante e $(f \star h)$ corresponde à convolução das funções f e h. Portanto, a transformada de Fourier do produto de duas funções no domínio espacial corresponde ao produto das transformadas dessas funções no domínio das frequências. Em termos matemáticos, isso é:

$$f(t) \star h(t) \Leftrightarrow F(u)H(u). \quad (5)$$

A flecha dupla representa que a expressão do lado direito é obtida a partir da transformada de Fourier da expressão da esquerda, assim como a expressão da esquerda é obtida a partir da transformada inversa de Fourier da expressão da direita.

Assim, ao se utilizar tanto o produto $H(u)F(u)$, no domínio das frequências, quanto o seu respectivo no domínio temporal, dado pelo teorema da convolução, deve-se tomar conta da periodicidade implícita ao sinal de entrada na expressão da transformada de Fourier para sinais discretos. Nesse caso, estaríamos aplicando a convolução de duas funções teoricamente periódicas, o que implica no resultado da operação ser igualmente periódico, porém, como explicado anteriormente, é estatisticamente improvável. Dessa forma, para remover essa periodicidade implícita ao sinal, que gera o conhecido efeito de borda, ou *wraparound error*, aplica-se o *zero padding* a $f(x)$.

O procedimento do *zero padding* consiste em adicionar zeros ao final de $f(x)$, antes de ser calculada sua transformada de Fourier. A quantidade de zeros a ser adicionada corresponde ao tamanho do vetor de entrada, ou seja, dado que $f(x)$ tenha tamanho k, ao se aplicar o procedimento, passa a possuir tamanho $2k$. Essa estratégia está matematicamente comprovada na obra desenvolvida por E. Oran Brigham (1988).

Tendo aplicado o *zero padding* às curvas de luz, o próximo passo é tentar simplificar a definição da função de transferência do filtro, $H(u)$. Baseando-se em propriedades de simetria de funções, nota-se que a especificação de $H(u)$ é bem simplificada quando se trabalha com funções simétricas ao seu centro, e isso requer que a transformada de Fourier dessa função também esteja centralizada. A centralização é demonstrada pela propriedade de translação e rotação da transformada de Fourier:

$$f(x)e^{j2\pi(u_0 x/M)} \Leftrightarrow F(u - u_0), \quad (6)$$

ou seja, multiplicar a transformada de Fourier pela exponencial da equação (6) desloca a origem da transformada para u_0. Tomando $u_0 = M/2$, a parcela exponencial se torna $e^{j\pi x} = (-1)^x$, estritamente porque x é um número inteiro. Então, pela equação (6), temos:

$$f(x)(-1)^x \Leftrightarrow F(u - M/2) \quad (7)$$

Portanto, por (7), a multiplicação do sinal de entrada $f(x)$ por $(-1)^x$, onde x corresponde a cada índice de $f(x)$, implica no deslocamento dos dados de tal forma que $F(0)$ está no centro do intervalo $[0, M-1]$, que é o tamanho de $f(x)$, como desejado.

Agora temos um sinal tratado e resistente a possíveis distorções nas transformações realizadas a partir de agora. A próxima etapa para prosseguirmos com a filtragem é decompor o sinal original temporal $f(x)$ em suas componentes de frequências, ou seja, calcular a transformada de Fourier de $f(x)$, escrita nesse trabalho como $F(u)$. A transformada de

Fourier de uma função temporal $f(t)$ é dada pela integral no domínio da função vezes uma exponencial, e representada por:

$$F(\omega) = \mathcal{F}\{f(t)\} \triangleq \int_{-\infty}^{+\infty} f(t)\, e^{-j\omega t}\, dt, \tag{8}$$

onde ω é a frequência complexa resultado da transformada.

No entanto, estamos trabalhando com sinais discretos e, dessa forma, precisamos discretizar a equação (8) de forma que possamos computar o espectro de frequências de $f(x)$. Assim, a transformada de Fourier de tempo discreto (DTFT) é dada pela somatória:

$$X(\omega) = \sum_{n=-\infty}^{\infty} x[n]\, e^{-j\omega n}, \tag{9}$$

onde $x[n]$ é o vetor de entrada (sinal digital de entrada).

Agora, deve-se realizar o produto de arranjos matriciais entre $F(u)$ e $H(u)$, ou seja, a filtragem propriamente dita. Para definirmos as funções de transferências discretas dos filtros utilizados, tomamos como referência a literatura de Gonzalez e Woods (1992). Dessa forma, a definição matemática de um filtro passa-baixa ideal (ILPF) é: sendo D_0 uma constante positiva, definida pelo usuário, e $D(u)$ a distância entre um ponto u até o centro do espectro de frequência do sinal de entrada, o ILPF permite passar todas as componentes de frequências em um círculo de raio D_0 a partir da origem do espectro e remove todas as componentes fora desse círculo. Dessa forma, pode ser escrito como:

$$H(u) = \begin{cases} 1, & se\ D(u) \leq D_0, \\ 0, & se\ D(u) > D_0 \end{cases} \tag{10}$$

sendo que $D(u)$ é definido por

$$D(u) = (u - P/2), \tag{11}$$

onde P é o tamanho do vetor original após o procedimento do *zero padding* e da expansão de bordas e o ponto de transição entre $H(u) = 1$ e $H(u) = 0$ é chamado de frequência de corte. Dessa forma, o filtro Ideal possui apenas um parâmetro livre, sua frequência de corte D_0.

O filtro gaussiano, assim como o ILPF, possui apenas um parâmetro livre, que é a frequência de corte da filtragem (que basicamente dá a largura do filtro). A função de transferência discreta unidimensional do filtro gaussiano passa-baixa (GLPF) é dada por

$$H(u) = e^{-\frac{D^2(u)}{2D_0^2}}, \tag{12}$$

onde $D(u)$ está definido em (11) e D_0 é uma constante positiva (frequência de corte).

Os filtros de Butterworth e Bessel, por sua vez, possuem dois parâmetros livres: a frequência de corte e a ordem de filtragem. A função de transferência discreta unidimensional $H(u)$ do filtro de Butterworth passa-baixa de ordem de filtragem n e frequência de corte D_0, é definida como

$$H(u) = \frac{1}{1 + [D(u)/D_0]^{2n}} \tag{13}$$

e a função de transferência discreta unidimensional $H(s)$ do filtro de Bessel passa-baixa de ordem de filtragem n e frequência de corte ω_0, é

$$H(s) = \frac{\theta_n(0)}{\theta_n(s/\omega_0)}, \tag{14}$$

onde,

$$\theta_n(s) = \sum_{k=0}^{n} a_k \, s^k \tag{15}$$

e

$$a_k = \frac{(2n-k)!}{2^{n-k} k! \, (n-k)!} \tag{16}$$

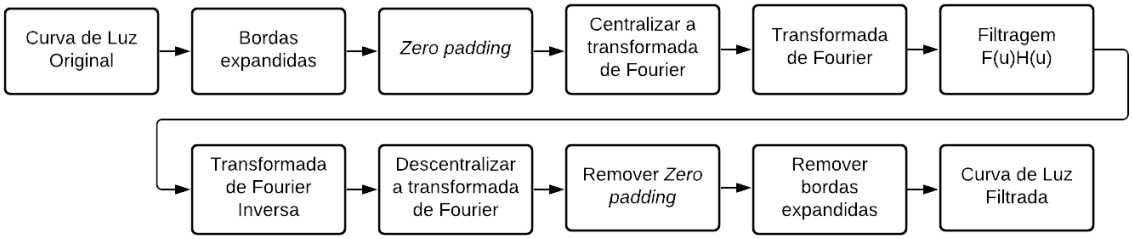
O filtro mediano, por outro lado, é aplicado de forma consideravelmente diferente, na qual cada valor dos dados filtrados corresponde a uma mediana de um grupo n de valores adjacentes nos dados originais. Este é o único filtro que não é aplicado no domínio de frequências e possui apenas o número n de valores adjacentes como parâmetro livre.

Nesse trabalho, foi realizada a filtragem de todo o conjunto de curvas de luz contendo exoplanetas (33 curvas) com essas cinco técnicas de filtragem e, em relação aos seus respectivos parâmetros livres, foram utilizadas frequências de cortes de 0.1 a 0.9 Nyquist, com passo de 0.1 Nyquist (sendo que 1 Nyquist corresponde à metade da frequência de amostragem do sinal discreto), ordens de filtragens de 1 a 6, com passo de 1 e números de valores adjacentes (para a filtragem mediana): 3, 5, 7, 9 e 11.

Agora, tendo definidas todas as técnicas aqui utilizadas e usando a linguagem Python, realizamos o produto $F(u)H(u)$ para os filtros passa-baixa ideal, gaussiano, de Butterworth e Bessel e utilizamos o método *medfilt* do pacote *Scipy.signal*, que, dado um *array* e o número de vizinhos a se considerar, retorna o *array* filtrado pelo filtro mediano.

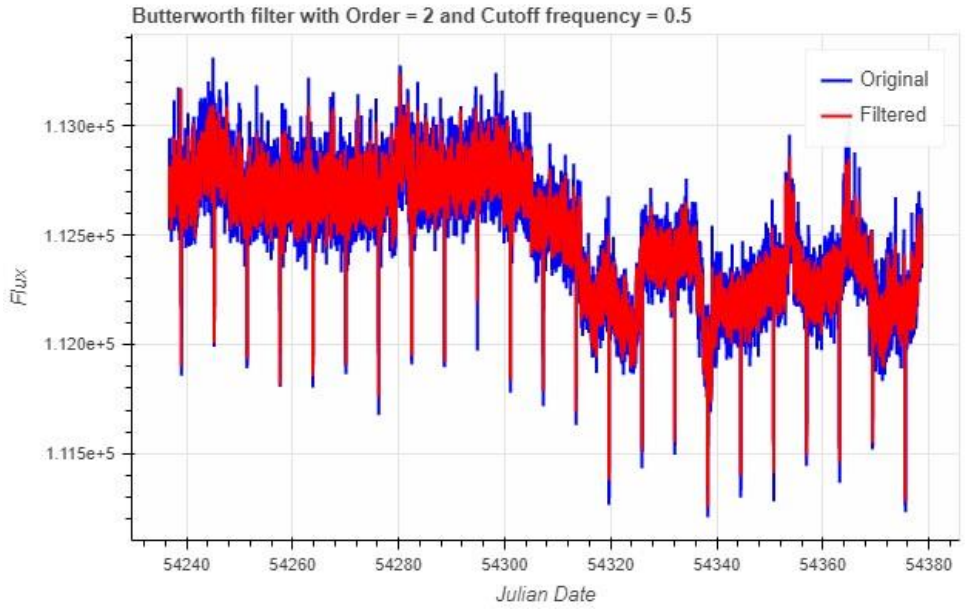
Para prosseguirmos, basta desfazer os procedimentos realizados antes da filtragem propriamente dita. Da operação $F(u)H(u)$, aplicamos a transformada de Fourier inversa nesse produto, que corresponde a curva de luz filtrada, e tomamos apenas a sua parte real. Em seguida, removemos o *zero padding* apenas recortando a segunda metade da curva de luz, que era composta por zeros. Após, é realizada a remoção das bordas expandidas e, para isso, basta remover o número escolhido de expansão, definido como N, e remover N valores antes e depois do resultado obtido. Finalmente, basta multiplicar a curva por $(-1)^x$, onde x corresponde a cada índice do vetor da curva de luz. As etapas para se realizar a filtragem das curvas de luz no domínio de frequências é mostrada na Figura 1.

Figura 1 – Fluxograma das etapas da filtragem no domínio de frequências



Assim, aplicamos essas etapas para as 33 curvas de luz contendo exoplanetas e obtivemos todo o conjunto de curvas filtradas com as diferentes técnicas aqui propostas. Um exemplo de curva de luz filtrada é apresentado a seguir. A Figura 2 representa a curva de luz de CoRoT-ID 0101086161, filtrada com o filtro de Butterworth, de ordem 2 e frequência de corte de 0.5 Nyquist.

Figura 2 - Exemplo de curva de luz filtrada



Modelagem teórica das curvas de luz contendo exoplanetas

Para avaliar a eficácia de cada um dos métodos de filtragem em análises de exoplanetas, foi aplicada uma modelagem teórica às curvas de luz, utilizando-se o formalismo proposto por Mandel e Agol (2008). Essa modelagem tem o propósito de determinar os seguintes parâmetros, e suas incertezas, dos exoplanetas orbitando as estrelas envolvidas: $a/R*$ (onde a é a distância do exoplaneta à estrela que orbita e $R*$ é o raio da estrela), $R_p/R*$ (onde R_p é o raio do exoplaneta), b (que corresponde ao parâmetro de impacto da passagem do exoplaneta na frente da estrela) e P (que corresponde ao período orbital do planeta). Não obstante, a aplicação dessa modelagem não pode ser feita sobre as curvas filtradas de forma direta, isto é, deve-se aplicar uma transformação às curvas de luz antes de sua aplicação.

O formalismo foi aplicado nos eclipses médios das curvas de luz, ou na o *phase-folded light curve*. O procedimento para se computar o eclipse médio se baseia nas seguintes etapas: primeiro, devem ser mapeados todos os pontos centrais dos eclipses de uma curva de luz, o que deve resultar em um vetor de índices que representa a localidade desses eclipses, chamado de *positions*. Em sequência, estima-se a largura média dos eclipses de uma curva de luz, denotado *width*. Dessa forma, para calcular o eclipse médio, basta somar todos as regiões da curva que contêm eclipses, ou seja, que estão em um intervalo [*positions[eclipse]-width, positions[eclipse]+width*], onde o índice *eclipse* é um número inteiro que possui como valor máximo o tamanho de *positions*, e dividir esse *array* pela quantidade de eclipses observados. Um fator importante a se considerar é que tanto o mapeamento dos índices das localidades dos eclipses quanto a determinação da largura média dos eclipses de uma curva de luz foram feitos manualmente, ou seja, não houve o uso de nenhuma técnica automática para tal.

Desse modo, foi determinado o vetor *positions* e o *float width* para cada curva de luz e os resultados foram armazenados em uma espécie de tabela. Logo após, foi aplicado o algoritmo *phase-fold* ao conjunto das curvas de luz filtradas e não filtradas, o que resultou em curvas (filtradas e não filtradas) prontas para terem seus parâmetros (descritos acima) determinados.

A modelagem teórica das curvas de luz contendo exoplanetas foi implementada em linguagem Python, e possui como entrada o eclipse médio da curva observada (para calcular o erro entre a curva simulada e a real), o parâmetro de impacto *b*, a razão entre o raio do exoplaneta e o raio da estrela, agora chamado de *p* (era $R_p/R*$), o período orbital *period* e a razão entre a distância do exoplaneta à estrela que orbita e o raio da estrela, agora chamado de *adivR* (era $a/R*$). O algoritmo apresenta como saídas o eclipse médio simulado e o valor do

qui-quadrado χ^2. Uma simulação que representa bem o eclipse médio da curva original apresenta χ^2 baixo, o que foi observado em todos os casos estudados.

Com a possibilidade de criar eclipses médios artificiais, a partir de 4 parâmetros das curvas de luz, e de se obter o χ^2 dessa curva artificial em relação a uma curva de referência, criamos o método *simulate.values()*, de um pacote de criação própria, que implementa a modelagem descrita acima. Esse método recebe como argumentos de entrada uma *grid* com valores de cada parâmetro respectivo às curvas de luz (*b, p, adivR* e *period)* e o eclipse médio observado, do qual deseja-se estimar os 4 parâmetros, e tem como saída os parâmetros que correspondem verdadeiramente à curva observada. Tais parâmetros correspondem à simulação que resultou no menor χ^2, ou seja, que obteve maior aproximação à curva original.

Assim, a partir desse momento, temos uma ferramenta capaz de determinar os 4 parâmetros de uma curva de luz contendo eclipses causados por exoplanetas. Contudo, a cada parâmetro é atrelado sua respectiva incerteza e infere-se que a essa incerteza tem origem, dentre outras fontes, do ruído presente nesses dados. Dentro do método *simulate.values()* possuímos, implicitamente, uma tabela de resultados para cada simulação, contendo os valores dos parâmetros testados e o χ^2 calculado em cada simulação. Para calcular as incertezas de cada parâmetro, foram, primeiramente, selecionados todos os casos desse parâmetro em que o $\chi^2 - \chi^2_{mínimo} < 1$, que corresponde a 1 desvio padrão. Sobre esses dados, foi calculado um erro dado por

$$e = \pm t \times \frac{\mathrm{s}}{\sqrt{\mathrm{n}}}, \tag{17}$$

onde s corresponde ao desvio padrão amostral, n ao tamanho da amostra e t é obtido a partir de uma consulta na tabela *t-student*, sendo uma função de uma dada confiança e de $\phi = n - 1$ (graus de liberdade). Adotamos uma confiança de 99%, ou seja, foi possível afirmar com 99% de certeza que o parâmetro escolhido realmente corresponde ao valor se fizermos simulações para uma grade de parâmetros infinita, que resultaria em um valor populacional. Para fazer a consulta na tabela, foi utilizado o método *ppf()* do pacote *Scipy.stats* onde, dado o valor da confiança e o tamanho da amostra, é retornado o valor de t correspondente.

Definido o processo para se calcular as incertezas de cada parâmetro, aplicamos o método *simulate.values()* para todos os eclipses médios das curvas de luz não filtradas e filtradas. Dessa forma, foi possível mensurar o quanto as diferentes técnicas de filtragens adotadas impactaram tanto na determinação dos parâmetros, que devem permanecer compatíveis aos originais, quanto nas incertezas atreladas a esses parâmetros.

Resultados e Discussão

A aplicação do método *simulate.values()* gerou uma tabela que apresenta os valores computados de cada parâmetro das curvas de luz, para cada técnica de filtragem aqui estudada, com suas respectivas incertezas. Para determinarmos quais foram as técnicas que resultaram em melhores estimativas dos parâmetros, primeiro, foram selecionados apenas os resultados cujos parâmetros das curvas filtradas permaneceram compatíveis com os das curvas originais, isto é, estavam dentro de um intervalo de $\pm 3\,\sigma$ em relação ao parâmetro original, o que corresponde a 99,7 % de abrangência. Em seguida, excluímos os resultados cuja incerteza da curva filtrada foi maior do que a da curva original. Após, computamos o percentual de redução, PR, de incertezas para cada parâmetro para cada curva de luz, dado por

$$PR = 100 \times \frac{incertezas\ com\ filtragem - \ incertezas\ sem\ filtragem}{incertezas\ sem\ filtragem}. \tag{17}$$

Finalmente, ordenamos essa tabela de resultados do maior para o menor PR, para cada curva e para cada parâmetro. Assim, observamos que a filtragem de Butterworth, com ordem variando

entre 1 e 2 e frequências de corte variando de 0.4 até 0.6 Nyquist, resultou em incertezas cerca de 35,13 % menores em relação às curvas originais, seguida pela filtragem de Bessel, com ordem variando entre 5 e 6 e frequências de corte de 0.4 até 0.6 Nyquist, que resultou em incertezas cerca de 31,18 % menores.

Ademais, nesse trabalho iniciou-se a construção do pacote *IMT-LightCurve*, disponibilizado em github.com/Guilherme-SSB/IMT-LightCurve-Library. Essa biblioteca tem como objetivo facilitar o processamento e a análise de curvas de luz e, para isso, disponibiliza diversos procedimentos comentados nesse estudo. Por exemplo, todas as técnicas de filtragens foram implementadas como métodos da classe *LightCurve* e o *script* de simulação das curvas de luz, visando a determinação de seus parâmetros, faz parte da classe *Simulate*.

Conclusões

Esse trabalhou confirmou a hipótese inicial de que o ruído de alta frequência das curvas de luz dificultava a identificação dos eclipses, assim como a determinação de certos parâmetros dos exoplanetas, detectados nesses dados. Foi examinado e confirmado que a utilização de técnicas de filtragem resulta, em geral, em melhores estimativas quanto à determinação dos parâmetros dos exoplanetas. Contudo, é necessário aplicar essas técnicas com cautela. Verificou-se que o filtro de Butterworth, juntamente com o filtro de Bessel, com ordens e frequências de corte específicas, foram os que proporcionaram as maiores melhorias (superiores a 30 %) na precisão dos parâmetros determinados para os exoplanetas. Por outro lado, o filtro Ideal resultou, majoritariamente, em incertezas maiores para esses parâmetros, ou seja, em piores estimativas.

Nota-se que os resultados foram obtidos a partir do estudo de dados passados, do satélite CoRoT, mas podem ser utilizados em estudos futuros contendo análise de exoplanetas, gerados, por exemplo, pelo telescópio espacial PLATO. Dessa forma, as decorrências desse estudo farão parte de uma futura proposta de pipeline completa de aquisição, processamento, análise e classificação inteligente das curvas de luz.

Referências Bibliográficas

Auvergne, M., Bodin, P., Boisnard, L., Buey, J. -T., Chaintreuil, S., el at. 2009. Automated classification of variable stars in the asteroseismology program of the Kepler space mission.

Deleuil, M., Airgrain, S., Moutou C., Cabrera, J., Bouchy, F., et al. 2018. Planets, candidates, and binaries from CoRoT/Exoplanet programme.

E. Oran Brigham. 1988. The fast Fourier transform and its applications.

Gonzales, R. e Woods, R. 1992. Digital Image Processing.

Koch, D. G., Borucki, W. J., Basri, G., Batalha, N. M., Brown, T. M., et al. 2010. Kepler mission design, realized photometric performance, and early science.

Mandel, K & Agol, E. 2002. Analytic Light Curves for Planetary Transit Searches.

Slawson, R. W., Prsa, A., Welsh, W. F., Orosz, J. A., Rucker, M., et al. 2011. VizieR Online Data Catolog: Kepler Mission. II. Eclipsing binaries in DR2.