Processamento Digital de Sinais e Aplicações em Acústica
Soluções para Tutorial 05 - Série de Fourier

https://github.com/fchirono/Aulas_PDS_Acustica

1 Síntese de Fourier

1.2 Tarefas

1. A Figura 1 mostra como a nota A4 deve parecer quando sintetizada usando três harmônicos de amplitudes [1, 0.4, 0.2].

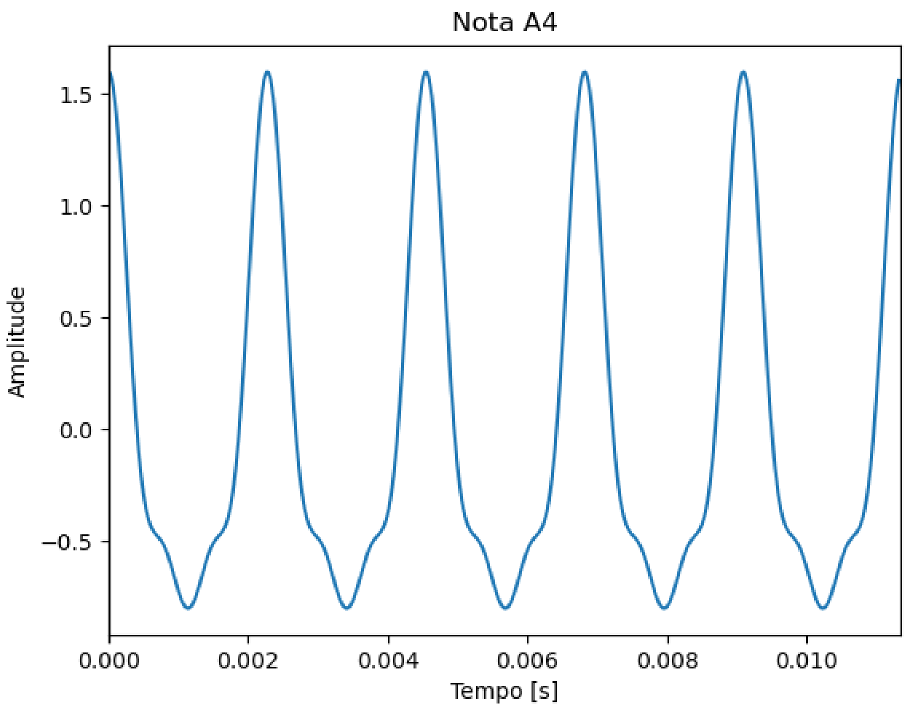
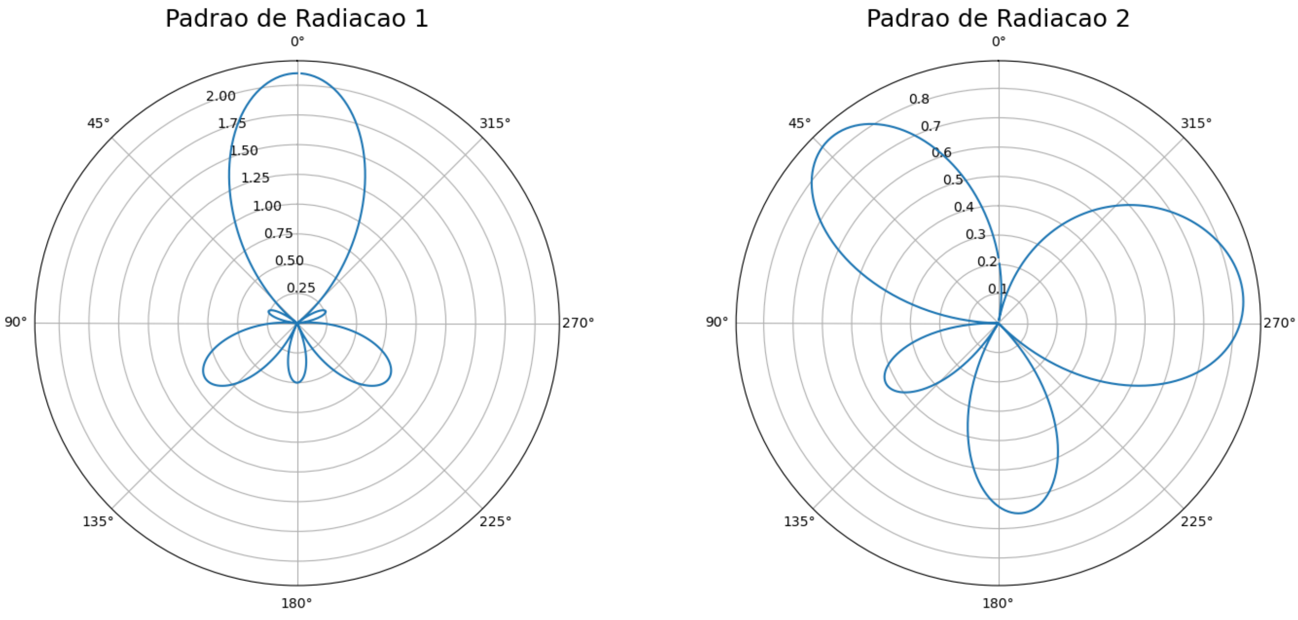


Figura 1: Nota A4 sintetizada usando três harmônicos de amplitudes [1, 0.4, 0.2].

3. O vetor `HarmAmpVect` contém os coeficientes da série de Fourier para a síntese de tom, e ao mudar seu conteúdo estamos mudando o número de harmônicos no tom e suas amplitudes relativas. Poderíamos usar o termo *conteúdo harmônico* para nos referir a essas mudanças também. De uma perspectiva musical, o conteúdo harmônico é parte do que define o *timbre* de uma nota; espera-se que você note os diferentes harmônicos mudando o timbre da melodia.

Este tipo de síntese, onde adicionamos harmônicos para mudar um som, é chamado de *síntese aditiva*; este tipo de síntese é usado no órgão eletrônico *Hammond*, por exemplo, onde o artista usa as barras deslizantes para controlar a proporção dos diferentes harmônicos.



(a) (b)

Figura 2: (a) Padrão de radiação obtido da função `PadraoRadiacao1`, e (b) padrão de radiação obtido da função `PadraoRadiacao2`.

2 Análise de Fourier

2.3 Tarefas

3. A Figura 2 mostra o valor absoluto dos padrões de radiação obtidos das funções.

4. Note que a função para determinar os coeficientes de Fourier do padrão de radiação deve receber como entrada a ordem N da série a ser calculada. Ambas essas funções de radiação foram sintetizadas usando uma série de Fourier de 3a ordem, então espera-se que sua função retorne zero para coeficientes de 4a ordem ou superiores.

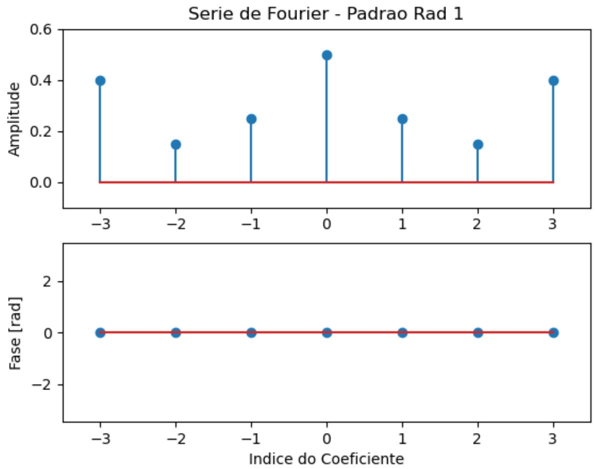
A Figura 3 mostra os coeficientes de Fourier obtidos dos padrões de radiação. Note que modificamos os limites dos eixos x e y das figuras para torná-las mais claras.

Os coeficientes que obtivemos ao calcular a série de Fourier usando 360 pontos sobre o domínio $(0, 2\pi]$ são:

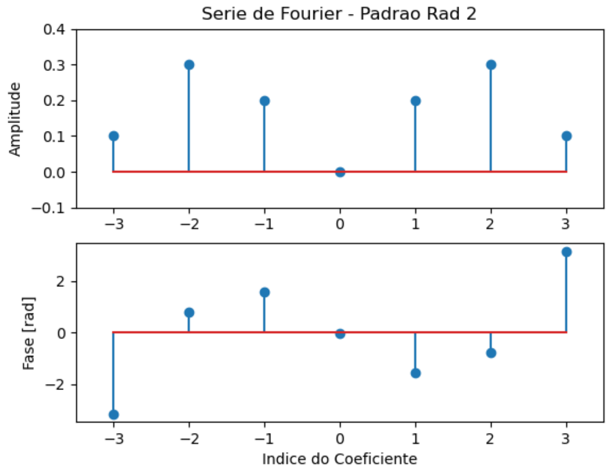
c_k	padrão de radiação 1	padrão de radiação 2
c_{-3}	0.40 -1.31561428e-16j	-1.00000000e-01 -5.06886200e-17j
c_{-2}	0.15 -1.08801856e-16j	2.12132034e-01 +2.12132034e-01j
c_{-1}	0.25 -5.88418203e-17j	8.74300632e-17 +2.00000000e-01j
c_0	0.50 -6.24500451e-19j	3.44169138e-17 -2.80641525e-18j
c_1	0.25 +5.88418203e-17j	8.79851747e-17 -2.00000000e-01j
c_2	0.15 +1.08801856e-16j	2.12132034e-01 -2.12132034e-01j
c_3	0.40 +1.31561428e-16j	-1.00000000e-01 +4.95090080e-17j

Note que números de ponto flutuante de 64 bits (como o Numpy usa para armazenar as partes real e imaginária de um número complexo) têm uma precisão de cerca de $2^{-53} \approx 1.11 \cdot 10^{-16}$; portanto, números de ponto flutuante cuja magnitude está próxima ou menor que 10^{-16} estão abaixo da precisão numérica da representação de ponto flutuante e devem ser considerados zero.

A consequência de ter um padrão de radiação simétrico em relação a 0° é que os coeficientes de Fourier são reais (até a precisão numérica - note a parte imaginária menor que a precisão numérica para os coeficientes do primeiro padrão de radiação na tabela acima).



(a)



(b)

Figura 3: (a) Coeficientes de Fourier obtidos da função `PadraoRadiacao1`, e (b) coeficientes de Fourier obtidos da função `PadraoRadiacao2`.

3 Apêndice: Gráficos Polares Avançados em Matplotlib/Pyplot

Como apêndice, vamos agora mostrar um trecho de código para demonstrar o método orientado a objetos para plotar figuras polares em Matplotlib/Pyplot; embora mais complicado, este método permite um grau muito maior de controle sobre os parâmetros do gráfico. A Figura 4 mostra o gráfico resultante para o padrão de radiação 2.

Leia o código, brinque com os parâmetros e tente entender o que cada comando faz. Comandos orientados a objetos similares também estão disponíveis para os gráficos cartesianos (isto é, não polares), então sinta-se à vontade para procurá-los e usá-los em suas figuras.

Este nivel de detalhe *não* é exigido para estes tutoriais, estamos apenas demonstrando o grau de controle que o Matplotlib lhe permite ter sobre as figuras.

```
import matplotlib.pyplot as plt

# Cria uma nova figura com um tamanho predefinido
fig_polar1 = plt.figure(figsize=(7, 7))

# Adiciona um unico "subplot" a figura, usando eixos polares
ax_polar1 = fig_polar1.add_subplot(111, polar=True)

# Plota os dados no eixo polar
plot_polar1 = ax_polar1.plot(phi, np.abs(padrao_rad1))

# Adiciona um titulo ao subplot e muda o tamanho da fonte do titulo
titulo_polar1 = ax_polar1.set_title('Padrao de Radiacao 1', size=18)

# Move o titulo ligeiramente para cima, para que nao se sobreponha ao
# indicador de 90 graus
titulo_polar1.set_y(1.09)

# Ativa o 'layout apertado' para remover alguns espacos vazios fora do
# subplot (tente desativa-lo para observar a diferenca)
fig_polar1.tight_layout()


# Igual a figura anterior - veja acima para detalhes
fig_polar2 = plt.figure(figsize=(7, 7))
ax_polar2 = fig_polar2.add_subplot(111, polar=True)
plot_polar2 = ax_polar2.plot(phi, np.abs(padrao_rad2))
titulo_polar2 = ax_polar2.set_title('Padrao de Radiacao 2', size=18)

# Define o raio maximo para o grafico polar
ax_polar2.set_rmax(1)

# Define a posicao 'zero graus' como 'Norte' (ou seja, apontando para cima)
ax_polar2.set_theta_zero_location('N')

# Define o angulo ('theta') para incrementar na direcao horaria
ax_polar2.set_theta_direction('clockwise')

# Define as localizacoes das marcacoes de grade na direcao radial e adiciona
# rotulos a elas. Note que os pontos de grade devem ser estritamente
# positivos, e eh necessario adicionar um pequeno raio positivo se voce
```

```
# quiser visualizar o marcador zero.
ax_polar2.set_rgrids([0.0001, 0.2, 0.4, 0.6, 0.8, 1.0],
                     labels=['0', '0.2', '0.4', '0.6', '0.8', '1.0'],
                     angle=-88)

# Define as localizacoes das marcacoes de grade na direcao 'theta' e adiciona
# rotulos em radianos. Esta notacao usa sintaxe LaTeX para exibir
# simbolos matematicos nas marcacoes de grade; nao se preocupe caso voce nao
# conheca LaTeX, o resultado eh puramente estetico.
#
# O comando 'frac=1.1' diz para adicionar os rotulos um pouco mais longe dos
# eixos do que o padrao, para fins de legibilidade (o padrao eh 'frac=1').
ax_polar2.set_thetagrids([0, 45, 90, 135, 180, 225, 270, 315],
                         labels=[r'$\theta = 0$', r'$+\frac{\pi}{4}$',
                                 r'$+\frac{\pi}{2}$', r'$+\frac{3\pi}{4}$',
                                 r'$\pm \pi$', r'$-\frac{3\pi}{4}$',
                                 r'$-\frac{\pi}{2}$', r'$-\frac{\pi}{4}$'],
                         size=18)

# Escreve uma linha de texto no grafico; os primeiros dois parametros sao as
# coordenadas (x, y) [ou, neste caso, as coordenadas (r, theta)] para o
# inicio da primeira letra do texto, o terceiro parametro eh a string de
# texto em si, e o quarto parametro controla o tamanho da fonte.
ax_polar2.text(-1.1*np.pi/2, 0.7, 'Magnitude', size=18)

# Move o titulo ligeiramente para cima, para que nao se sobreponha ao
# indicador 'theta = 0'
titulo_polar2.set_y(1.09)
```

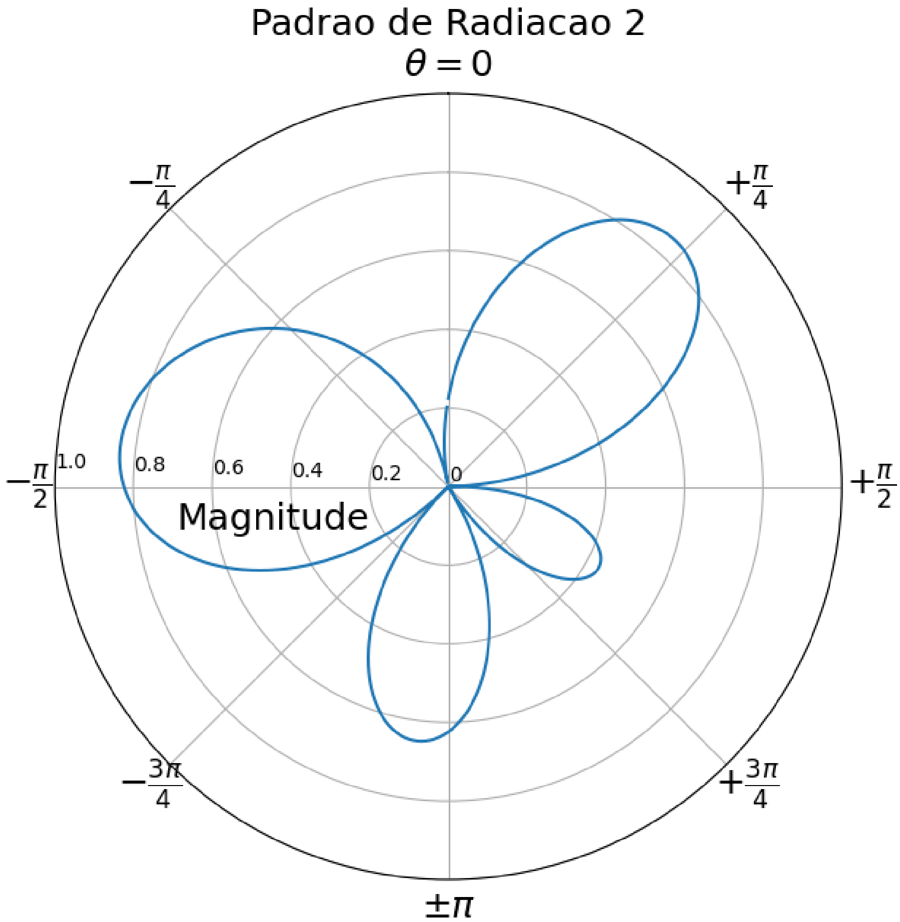


Figura 4: Gráfico alternativo do padrão de radiação 2.