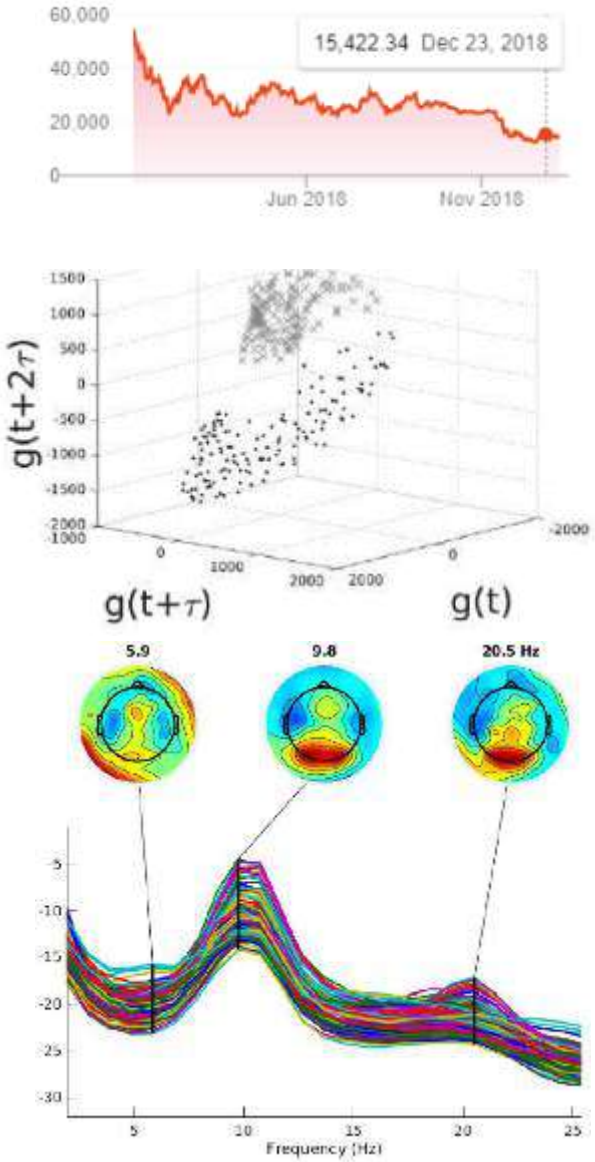


UNICAMP

Instituto de Computação

MC102: Algoritmos e Programação de Computadores (turmas 4,5,6,7)
Extra – Criando programas com Interface Gráfica

ARTHUR VALENCIO

Pós-doutorando IC/Unicamp

Fapesp CEPID NeuroMat

Campinas, 8 de Abril de 2020

>>Sobre

A linguagem Python possui uma biblioteca padrão para interface gráfica chamada **Tkinter**, porém ela é um pouco difícil de usar

Muitos desenvolvedores tendem a usar outras bibliotecas como a **PyQt**, que simplifica a escrita e adiciona funcionalidades.

Exemplos de programas utilizando PyQt são: Dropbox (serviço de armazenamento na nuvem), OpenShot (editor de vídeo para Linux), QGIS (sistema de informaçãoes geográficas gratuito), Spyder (ambiente de desenvolvimento de Python para ciência de dados) e Veusz (programa gratuito de desenvolvimento de gráficos científicos)

Outra alternativa é o **PyGTK**, que, por exemplo, está por detrás de programas como: GIMP (editor de imagens gratuito), gedit (editor de texto padrão do Linux), BitTorrent (gerenciador torrent) e Ubuntu software center

Uma biblioteca beeem mais simples, para um primeiro contato com desenvolvimento de interface gráfica, é a **PySimpleGUI**

>>PySimpleGUI: instalação

Para começar a usar as funcionalidades de construção de gráficos, digite a seguinte linha no cabeçalho do seu programa (ou no terminal python):

```
import PySimpleGUI as sg
```

Se aparecer uma mensagem de erro, é preciso instalar o pacote. Nesse caso vá ao Prompt de Comando do Windows ou Terminal do Linux/Mac e digite:

```
pip3 install pysimplegui
```

(feche e reabra o idle ou IDE de preferência após isso)

>>Exemplo de programa

```
import PySimpleGUI as sg
sg.theme('DarkAmber')

layout = [ [sg.Text('Olá Mundo!')],
           [sg.Text('Digite alguma coisa:'), sg.InputText('texto')],
           [sg.Text('Arquivo:'), sg.Input() , sg.FileBrowser()],
           [sg.OK(), sg.Cancel()]]

programa = sg.Window('Nome do Programa', layout)

while True:
        evento, valores = programa.read()
        if evento == None or evento== 'Cancel'):
                break
programa.close()
```

>>Exemplo de programa

```
import PySimpleGUI as sg
sg.theme('DarkAmber')

layout = [ [sg.Text('Olá Mundo!')],
           [sg.Text('Digite alguma coisa:'), sg.InputText('texto')],
           [sg.Text('Arquivo:'), sg.Input() , sg.FileBrowser()],
           [sg.OK(), sg.Cancel()]]

programa = sg.Window('Nome do Programa', layout)

while True:
        evento, valores = programa.read()
        if evento == None or evento== 'Cancel':
                break
programa.close()
```

Importanto PySimpleGUI e dando um apelido mais simples

Adicionando um padrão de cores para o nosso programa

>>Exemplo de programa

Aqui você cria todos os elementos do seu programa.
Cada elemento da *lista* **layout** vira uma linha na tela do programa

```
import PySimpleGUI as sg
sg.theme('DarkAmber')

layout = [ [sg.Text('Olá Mundo!')],
           [sg.Text('Digite alguma coisa:'), sg.InputText('texto')],
           [sg.Text('Arquivo:'), sg.Input() , sg.FileBrowser()],
           [sg.OK(), sg.Cancel()]]

programa = sg.Window('Nome do Programa', layout)

while True:
        evento, valores = programa.read()
        if evento == None or evento== 'Cancel':
                break
programa.close()
```

Adiciona um texto na 1ª linha

Adiciona um texto e uma caixa de input na 2ª linha

Adiciona texto, caixa de input e buscador de arquivos na 3ª linha

Adiciona botões de OK e Cancelar na 4ª linha

>>Exemplo de programa

Cria o programa, informando qual o nome que aparece da barra de título e qual a variável que tem layout com todos os itens

```
import PySimpleGUI as sg
sg.theme('DarkAmber')

layout = [ [sg.Text('Olá Mundo!')],
           [sg.Text('Digite alguma coisa:'), sg.InputText('texto')],
           [sg.Text('Arquivo:'), sg.Input() , sg.FileBrowser()],
           [sg.OK(), sg.Cancel()]]

programa = sg.Window('Nome do Programa', layout)

while True:
           evento, valores = programa.read()
           if evento == None or evento== 'Cancel':
                      break
programa.close()
```

>>Exemplo de programa

```
import PySimpleGUI as sg
sg.theme('DarkAmber')

layout = [ [sg.Text('Olá Mundo!')],
           [sg.Text('Digite alguma coisa:'), sg.InputText('texto')],
           [sg.Text('Arquivo:'), sg.Input() , sg.FileBrowser()],
           [sg.OK(), sg.Cancel()]]

programa = sg.Window('Nome do Programa', layout)

while True:
    evento, valores = programa.read()
    if evento == None or evento== 'Cancel':
        break
programa.close()
```

Roda o programa

Continuamente o programa vai ser lido

Se for fechado ou cancelado, o loop deve quebrar

E assim o programa deve terminar

>>Exemplo de programa

Este é o seu resultado:

>>Um programa estilo Windows

```
import PySimpleGUI as sg

menu_def = [['Arquivo', ['Abrir', 'Salvar', 'Sair'  ]],
            ['Editar', ['Colar', ['Especial', 'Normal', ], 'Desfazer'],],
            ['Ajuda', 'Sobre...'],]

layout = [[sg.Menu(menu_def,)],[sg.Output(size=(60, 20))]]

window = sg.Window("Meu programa", layout)

while True:
    event, values = window.read()
    if event == None or event == 'Sair':
        break
    print('Clicou em = ', event)

    if event == 'Sobre...':
        sg.popup('Sobre este programa', 'Versao 1.0')
    elif event == 'Abrir':
        filename = sg.popup_get_file('Arquivo para abrir', no_window=True)
        print(filename)
```

>>Um programa estilo Windows

```
import PySimpleGUI as sg

menu_def = [['Arquivo', ['Abrir', 'Salvar', 'Sair'  ]],
            ['Editar', ['Colar', ['Especial', 'Normal', ], 'Desfazer'],],
            ['Ajuda', 'Sobre...'],]

layout = [[sg.Menu(menu_def,)],[sg.Output(size=(60, 20))]]

window = sg.Window("Meu programa", layout)

while True:
    event, values = window.read()
    if event == None or event == 'Sair':
        break
    print('Clicou em = ', event)

    if event == 'Sobre...':
        sg.popup('Sobre este programa', 'Versao 1.0')
    elif event == 'Abrir':
        filename = sg.popup_get_file('Arquivo para abrir', no_window=True)
        print(filename)
window.close()
```

Cria o menu

Define o layout.
No caso, só tem o menu e uma caixa de texto

Cria o programa

Roda o programa

Fecha o programa caso saia do loop

>>Um programa estilo Windows

```
import PySimpleGUI as sg

menu_def = [['Arquivo', ['Abrir', 'Salvar', 'Sair'  ]],
            ['Editar', ['Colar', ['Especial', 'Normal', ], 'Desfazer'],],
            ['Ajuda', 'Sobre...'],]

layout = [[sg.Menu(menu_def,)],[sg.Output(size=(60, 20))]]

window = sg.Window("Meu programa", layout)

while True:
    event, values = window.read()
    if event == None or event == 'Sair':
        break
    print('Clicou em = ', event)

    if event == 'Sobre...':
        sg.popup('Sobre este programa', 'Versao 1.0')
    elif event == 'Abrir':
        filename = sg.popup_get_file('Arquivo para abrir', no_window=True)
        print(filename)
```

Continuamente vai lendo as ações do usuário

Sai do loop (fecha o programa) se ele sair

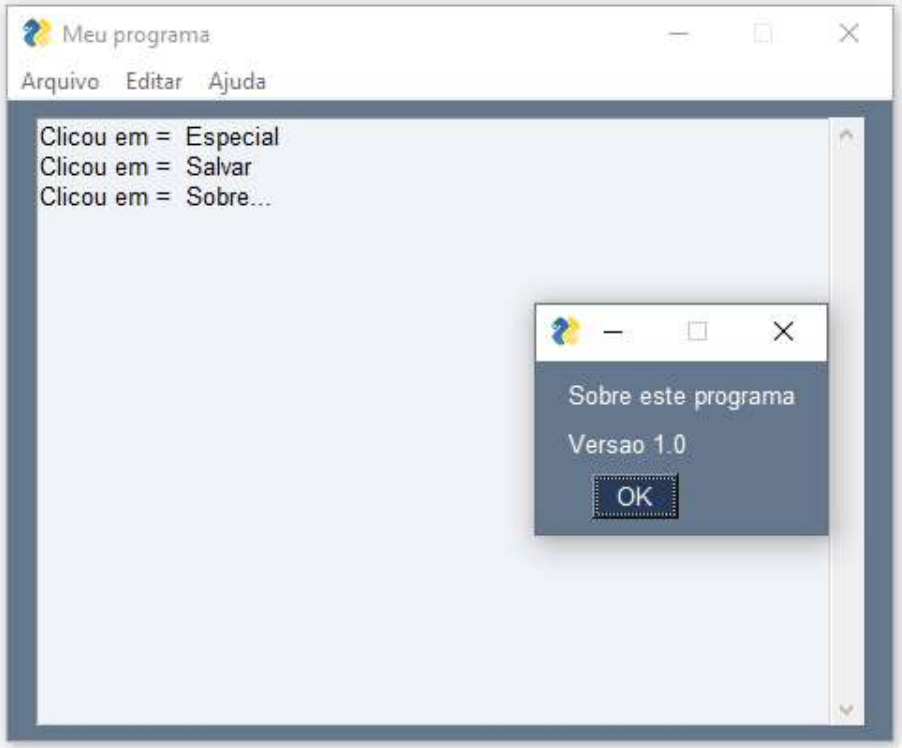
Mostra na caixa de Output a ação do usuário caso ele tenha clicado em qualquer outra opção

Se clicar em sobre, abre caixa de pop-up

Se clicar em abrir, gera o browser de arquivos e informa o endereço do arquivo selecionado

>>Um programa estilo Windows

Este é o seu resultado:



>>Um programa interativo

Vamos construir um gráfico aonde o usuário escolhe os parâmetros e a imagem é atualizada automaticamente

>>Um programa interativo

Parte 1:

- Importando bibliotecas
- Criando função para desenhar os eixos

```
import PySimpleGUI as sg
import math

Tamanho_X = 25
Tamanho_Y = 25

def desenha_eixo():
    graph.draw_line((-Tamanho_X, 0), (Tamanho_X, 0))
    graph.draw_line((0, -Tamanho_Y), (0, Tamanho_Y))
    x=-Tamanho_X
    y=-Tamanho_Y

    while x <= Tamanho_X:
        graph.draw_line((x, -1), (x, 1))
        if x != 0:
            graph.draw_text(str(x), (x, -3), color='green')
        x=x+5

    while y <= Tamanho_Y:
        graph.draw_line((-1, y), (1, y))
        if y != 0:
            graph.draw_text(str(y), (-3, y), color='blue')
        y=y+5
```

desenha os eixos x e y

desenha marcadores de valores (ticks) no eixo x

Escreve valores dos marcadores no eixo x, exceto origem

Estes marcadores aparecem só em valores de 5 em 5

idem para eixo y

>>Um programa interativo

Parte 2:
Terminada a função de contrução do eixo, agora vamos contruir o layout do programa

```
sg.theme('SystemDefault1')
```
→ Padrão de cores para o programa

```
graph = sg.Graph(canvas_size=(400, 400),
        graph_bottom_left=(-(Tamanho_X+5), -(Tamanho_Y+5)),
        graph_top_right=(Tamanho_X+5, Tamanho_Y+5),
        background_color='white',
        key='graph')
```
Cria o elemento de gráfico, de tamanho 400x400 pixels, iniciando com uma margem (5) do limite dos eixos, fundo branco

```
layout = [[sg.Text('Gráfico do seno', justification='center')],
```
→ Cria título do gráfico

```
    [graph],
```
→ Cria o gráfico

```
    [sg.Text('y = A*sin(w*x + phi)', font='COURIER 18')],
```
→ Cria texto explicando as variáveis

Cria 3 barras seletoras (slider), uma para cada variável que o usuário pode controlar

```
    [sg.Text('A'), sg.Slider((0, 20), resolution=0.1, orientation='h', enable_events=True, key='-SLIDER_A-')],
    [sg.Text('w'), sg.Slider((1, 6), resolution=0.1, orientation='h', enable_events=True, key='-SLIDER_w-')],
    [sg.Text('phi'), sg.Slider((1, 10), resolution=0.1, orientation='h', enable_events=True, key='-SLIDER_phi-')]]
```

Essas chaves serão os “nomes” com os quais você poderá utilizar/controlar os valores

Cria o programa

```
window = sg.Window('Gráfico do seno', layout)
```

>>Um programa interativo

Parte 3:
Rodar o programa

```
while True:
    event, values = window.read()
    if event == None:
        break
    graph.erase()
    desenha_eixo()
    x_anterior = None
    y_anterior = None

    x=-Tamanho_X
    while x <= Tamanho_X:
        y = float(values['-SLIDER_A-'])*math.sin(float(values['-SLIDER_w-'])*x+float(values['-SLIDER_phi-']))
        if x_anterior != None:
            graph.draw_line((x_anterior, y_anterior), (x, y), color='red')
        x_anterior = x
        y_anterior = y
        x=x+0.1
window.close()
```

Ler as ações do usuário

Se usuário fechar o programa, sair do loop

Apaga gráfico anterior

Desenha os eixos (nossa função na parte 1)

Note que as variáveis escolhidas pelos usuários são chamadas como values[key]

Para cada x ao longo do eixo, calcula y=A*sin(w*x+phi)

Desenha linha unindo ponto anterior e ponto atual, construindo, assim, o gráfico

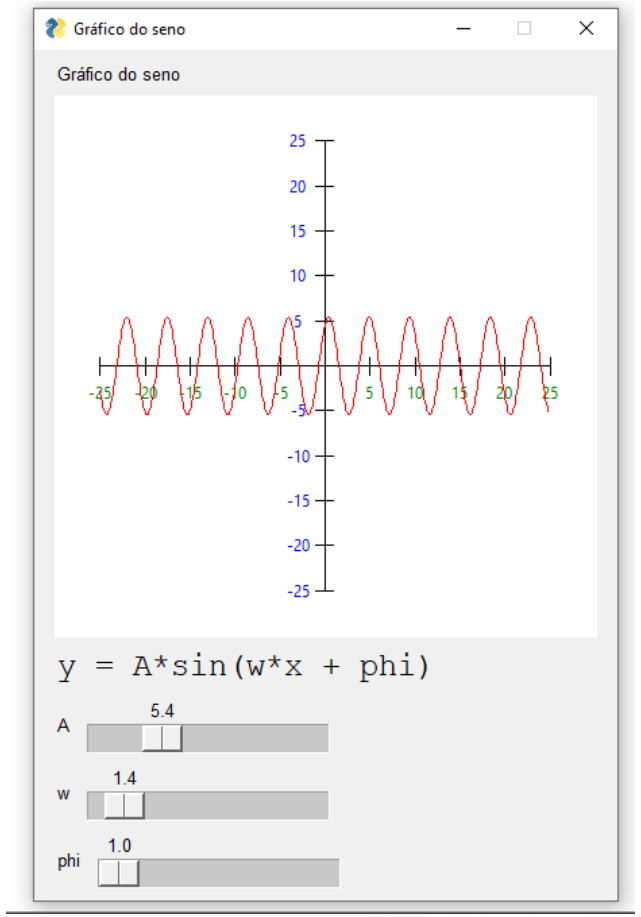
Preparando para o o próximo passo do loop: salvar os valores atuais de x e y como x_anterior e y_anterior

Incrementar x em 0.1

Finalizar programa se sair do loop

>>Um programa interativo

Este é o seu resultado:



>>Criando arquivo .exe (Windows) a partir do Python

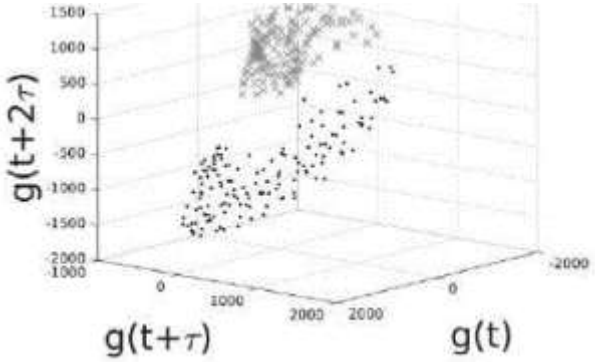
A forma mais simples é instalando o pacote **PyInstaller**. Nesse caso vá ao Prompt de Comando do Windows ou Terminal do Linux/Mac e digite (só é necessário fazer isso 1 vez):

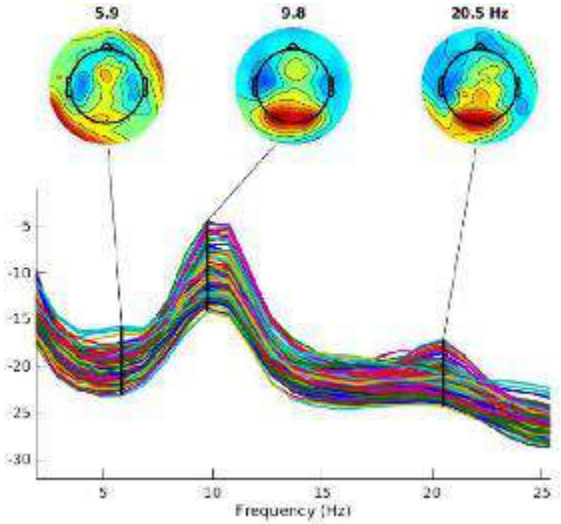
```
pip3 install PyInstaller
```

Feito isso, crie o arquivo **.exe** através do Prompt de Comando do Windows ou Terminal Linux/Mac indo na pasta aonde está o seu arquivo e digitando:

```
pyinstall -wF <nome_do_arquivo>.py
```

O arquivo **.exe** estará disponível dentro da pasta **dist** no mesmo diretório





UNICAMP

Instituto de Computação

That´s all folks!

arthur_valencio@physics.org

http://www.arthurvalencio.com/mc102

http://www.ic.unicamp.br/~mc102