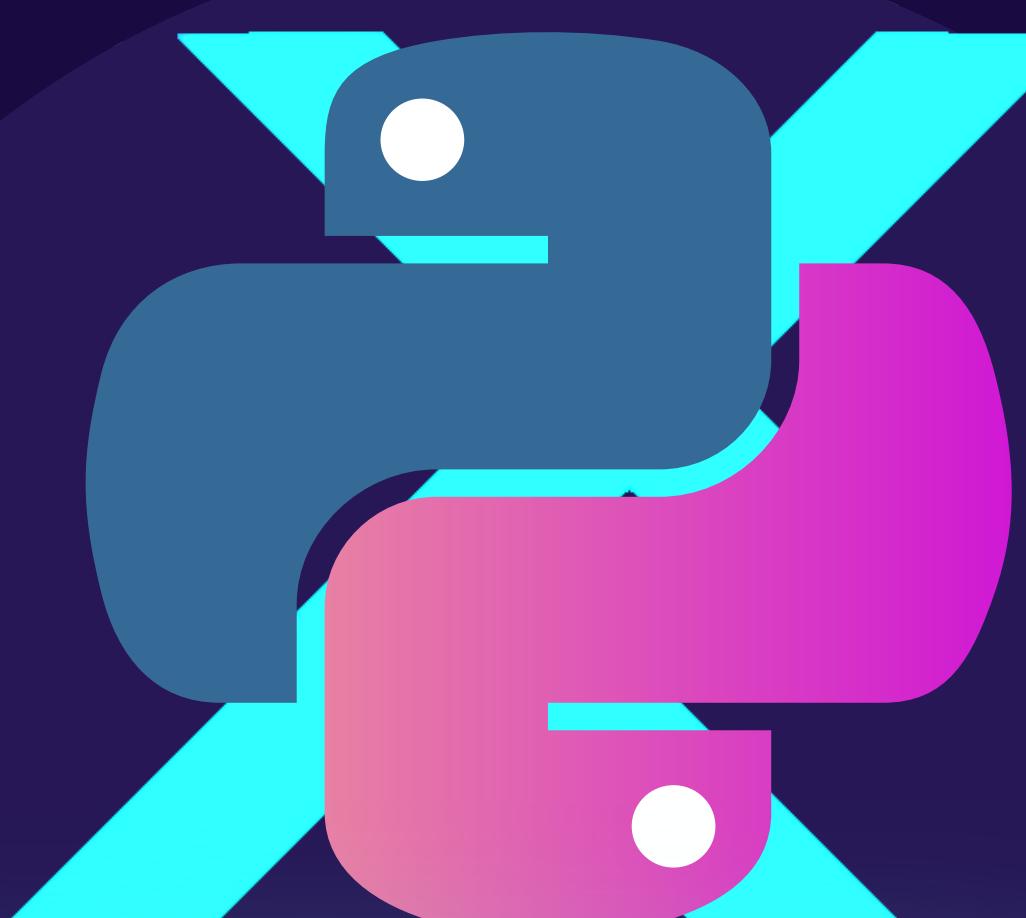
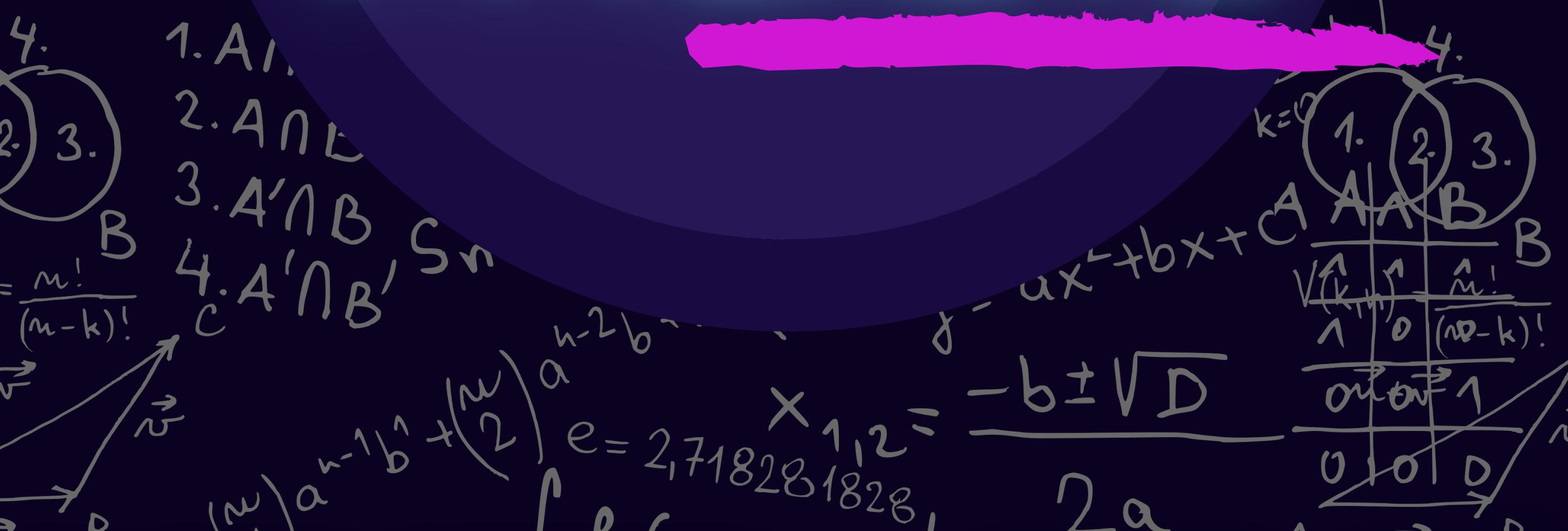


UNIVERSIDADE DE SÃO PAULO

MANUAL



PYTENCYS



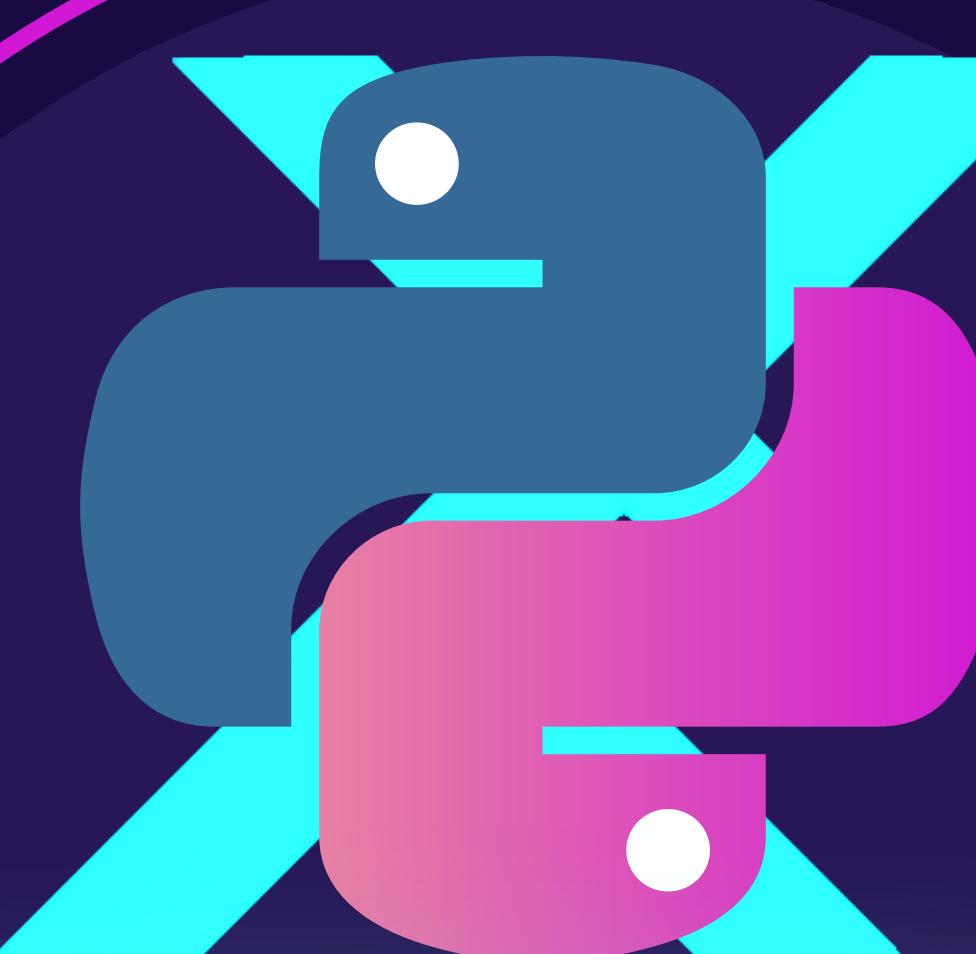
EDUARDO GIOPATTO

GABRIEL TONÉ

LUÍZ GUILHERME

TÁBATA SANTOS

PYTHON



DESENVOLVIDO

POR

Eduardo Giopatto nºUSP 11801327

Graduando em Engenharia Física

Gabriel Tomé nºUSP 11368849

Graduando em Engenharia Física

Luiz Guilherme nºUSP 11801460

Graduando em Engenharia Física

Tábata Santos nºUSP 11294681

Graduanda em Engenharia Física

TERMOS

COPYRIGHT (C) 2020 PYTENCY5

A PERMISSÃO É CONCEDIDA, GRATUITAMENTE, A QUALQUER PESSOA QUE OBTENHA UMA CÓPIA DESTE SOFTWARE E ARQUIVOS DE DOCUMENTAÇÃO ASSOCIADOS ("SOFTWARE"), PARA LIDAR COM O SOFTWARE SEM RESTRIÇÃO, INCLUINDO, SEM LIMITAÇÃO, OS DIREITOS DE USAR, COPIAR, MODIFICAR, MESCLAR, PUBLICAR, DISTRIBUIR, SUBLICENCIAR E / OU VENDER CÓPIAS DO SOFTWARE, E PERMITIR QUE AS PESSOAS A QUEM O SOFTWARE É FORNECIDO O FAÇAM, SUJEITO ÀS SEGUINTEs CONDIÇÕES:

TERMOS

COPYRIGHT (C) 2020 PYTENCY5

O AVISO DE DIREITOS AUTORAIS ACIMA E ESTE AVISO DE PERMISSÃO DEVEM SER INCLUÍDOS EM TODAS AS CÓPIAS OU PARTES SUBSTANCIAIS DO SOFTWARE.

O SOFTWARE É FORNECIDO "NO ESTADO EM QUE SE ENCONTRA", SEM QUALQUER TIPO DE GARANTIA, EXPRESSA OU IMPLÍCITA, INCLUINDO MAS NÃO SE LIMITANDO ÀS GARANTIAS DE COMERCIALIZAÇÃO, ADEQUAÇÃO A UM DETERMINADO FIM E NÃO VIOLAÇÃO. EM NENHUMA HIPÓTESE OS AUTORES OU TITULARES DOS DIREITOS AUTORAIS SERÃO RESPONSÁVEIS POR QUALQUER RECLAMAÇÃO, DANOS OU OUTRA RESPONSABILIDADE, SEJA EM UMA AÇÃO DE CONTRATO, DELITO OU DE OUTRA FORMA, DECORRENTE DE, FORA DE OU EM CONEXÃO COM O SOFTWARE OU O USO OU OUTRAS NEGOCIAÇÕES NO PROGRAMAS.

CONDIÇÕES DE USO

1. É necessário um arquivo excel com duas planilhas separadas.
2. O primeiro deverá ser a sua matriz quadrada diagonalizável
3. E a segunda matriz seu chute inicial.

$A \setminus B$
 $\frac{1}{1} \quad \frac{1}{1}$
 $\frac{0}{(n-k)!} \quad \frac{n!}{1}$
 $\frac{1}{1} \quad \frac{1}{1}$
 $\frac{0}{1} \quad \frac{0}{0}$
 $\frac{1}{1} \quad \frac{1}{1}$
 $\frac{0}{0} \quad \frac{1}{1}$
 $\frac{0}{0} \quad \frac{0}{0}$

3. $A' \cap B$, $S_n = 0^n$
 4. $A' \cap B'$, $S_n = \frac{1}{(n-1)!} a^{n-1} b + \frac{1}{(n-2)!} a^{n-2} b^2 + \dots + \frac{1}{1!} a^1 b^{n-1}$
 $y = ax^2 + bx + c$
 $-b \pm \sqrt{D}$

CÓDIGO

```

Power Method:
método das potências:
--> Descobrir o maior autovalor e o maior autovetor
de uma matriz quadrada diagonalizável
Funções:
--> método_das_potências: calcula os maiores
autovalores e autovetores de maior módulo

import numpy as np
import matriz as mat

def método_das_potências(prec):
    """
    Cálculo dos valores de maior módulo pelo método
    das potências.

    Argumentos:
    - obrigatórios:
        prec - o usuário deverá colocar a precisão que a interação
        deverá ter.

    Restrições:
    O usuário deverá colocar matrizes quadradas n dimensionais
    diagonalizáveis.
    """

    A = mat.matriz()
    print(A)
    X = mat.Interacao()
    Y = A.dot(X.T) # multiplica a matriz A pela transposta de X
    lambd = max(Y) # separa o valor de maior módulo
    X = (1/lambd)*Y # calcula o autovetor
    lambd_anterior = lambd + 1 #pra rodar
    erro = abs(lambd - lambd_anterior)
    print(Y)
    print(X)

    while erro > prec: #laço de interações
        Y = A.dot(X)
        lambd_anterior = lambd
        lambd = max(Y)
        X = (1 / lambd) * Y
        erro = abs(lambd - lambd_anterior)
        print(Y)
        print(f'autovetor:{X}')
        print(f'autovalor: {lambd}')
        print('-----')

    Lambd_final = lambd
    print(f'O autovalor final é: {Lambd_final}') #retorna o lambda final
    X_final = X
    print(f'O autovetor final é:\n {X_final}') #retorna o autovetor final

    return X_final, Lambd_final

```

CÓDIGO

```
import pandas as pd
import numpy as np

def matriz(arq = 'input_user.xlsx', sheet = 'Matriz' ):
    """
    Argumentos (opcionais):
        arq: O nome do arquivo em formato de xlsx.
        sheet: E em qual planilha se encontra
    Restrições:
        Na planilha o usuário deverá usar a partir da 1° coluna
        (B) e 2° linha (2).
    """
    df = pd.read_excel(arq, sheet_name = sheet)
    Array = df.to_numpy()
    return Array

def Interacao(arq = 'input_user.xlsx', sheet = 'Interacao' ):
    """
    Argumentos (opcionais):
        arq: O nome do arquivo em formato de xlsx.
        sheet: E em qual planilha se encontra
    Restrições:
        Na planilha o usuário deverá colocar na primeira linha
        "nomes" que representem o número e colocar os elementos da matriz
        inicial de chute em linha começando a partir da segunda.
    """
    df = pd.read_excel(arq, sheet_name = sheet)
    chute = df.to_numpy()
    return chute

import matriz as mat
import power_method as pm

def relat():
    """
    relat --> gera o relatório com a matriz de input
    o autovalor e autovetor de maior módulo.
    ** sem argumentos opcionais ou obrigatórios.
    """
    M0 = mat.matriz() #chama o módulo matriz inicial
    X0 = mat.Interacao() # chama o módulo do chute inicial
    X_final, Lambd_final = pm.metodo_das_potencias(prec = 1e-3)

    with open('arquivo.txt', 'w') as arq: #cria o arquivo; w: write
        arq.write('PYTENCCS'.center(60, '#')+'\n')
        arq.write(f'Matriz do usuário: \n{M0}')
        arq.write(f'\nChute inicial: \n{X0}\n')
        arq.write(f'O autovetor final é: \n{X_final}\n')
        arq.write(f'O autovalor final é: \n{Lambd_final}\n')
        arq.write('#'*60)
```

CÓDIGO

```
import power_method as pm
import relatorio as rl

...
''' Main.py: Será usado para chamar a matriz fornecida pelo usuário
    * biblioteca necessária:
        » power_method: biblioteca principal
    *biblioteca opcional:
        » relatorio: gera o relatório com os dados obtidos
            » na biblioteca power_method
...
pm.metodo_das_potencias(prec=1e-3)
rl.relat()
```

EXCEL

	A	B	C	D
1				
2		5	-1	7
3		-1	-1	1
4		7	1	5
5				
6				
7				
8				

	A	B	C	D
1	cx	cy	cz	
2		1	1	1
3				
4				
5				
6				

TUTORIAL

1º passo --> Escrever a matriz quadrada diagonalizável na

planilha chamada "Matriz" e desconsiderar a primeira linha da planilha

2º passo --> Escrever o chute inicial, na planilha chamada "Interacao" em linha com a mesma dimensão da matriz, ou seja, se a matriz que queremos achar seja uma com um tamanho 3x3, o chute inicial deverá ser (x,x,x).

TUTORIAL

3º passo --> no arquivo main.py

use a biblioteca "power_method.py" (obrigatório) e importe a biblioteca "relatorio.py"

caso queira um relatório geral do algoritimo, como o autovalor e o autovetor de maior módulo

4º passo --> depois de importado os módulos use as funções:

```

```
power_method.metodo_das_potencias
(prec=1e-3) #prec: precisão do
```

algoritimo

```

caso você tenha optado por gerar o relatorio use a função:

```

# TUTORIAL

`relatorio.relat()` não precisa de argumentação

5º passo --> Rode o programa e ele irá mostrar pelo terminal o resultado final. E caso tenha usado a biblioteca "relatorio.py" um arquivo txt será gerado com todos os dados.

# PYTENCYS

EDUARDO GIOPATTO  
LUÍZ GUILHERME

GABRIEL TOMÉ  
TÁBATA SANTOS