Nocoes

September 22, 2021

0.1 # Noções básicas do Jupyter

0.2 Introdução

O `Jupyter` é uma aplicação web que permite compartilhar e criar documentos computacionais que envolvam códigos, simulações e seus resultados. Assim é possível documentar toda a "história" da pesquisa, através de um documento interativo que permite a organização de ideias de forma simples e direta.

Outro uso importante é a execução fracionada de código, que facilita a realização de testes breves. Permite ainda descrição de modelos e observações científicas misturando-se códigos e texto (inclusive fórmulas).

Um dos usos mais notáveis da aplicação é a possibilidade de se garantir a reprodução das observações, permitindo que outros interessados no conteúdo possam verificar, melhorar e estender as percepções.

0.3 O básico

No `Jupyter`, cada arquivo com a extensão `.ipynb` representa um novo arquivo de notas. Estes arquivos, possuem uma série de *células* que possuem um determinado tipo. Este tipo indica à aplicação como ela deve tratar o conteúdo ali contido e pode ser visto nas barras.

Abaixo vemos um célula do tipo *code*.

Note que a presente célula (este texto) possui o tipo *Markdown*, que é uma linguagem simples de marcação de texto, que nos permitiu redigir essa mensagem.

O tipo *Heading* adiciona uma célula com a notação de um cabeçalho utilizado em *Markdown*. JupyterLab

[1]:
```
x=2
```

Um tipo de célula tida como "esotérica" para muitos é o *Raw NBconvert* ou *Raw*. A suite do `ipython/jupyter` possui a opção `nbconvert` que permite transformar um arquivo de notebook `.ipynb` em outros formatos de documentos. O conteúdo desta célula terá o processamento ignorado pela ferramenta, sendo colocado *ipsis litteris* no arquivo convertido. Como exemplo, a célula subsequente contem um trecho de código `Latex` que adiciona uma equação no documento final.

O comando abaixo converte o presente notebook em um arquivo `Latex`. O comando utiliza internamente o `Pandoc`, uma suíte de conversão de documentos que utilizem de algum formato de marcação para outro.

```
ipython nbconvert --to latex Nocoes.ipynb
```

O documento fruto da conversão pode ser visto aqui. Depois de "compilado" o arquivo de saída fica assim. Podemos ver que a instrução `Latex` utilizada na célula foi fielmente disposta no documento convertido.

$$\delta = b^2 - 4ac \tag{1}$$

0.3.1 Trabalhando com as células

Existem dois modos de trabalho com as células, que são observados pela cor ao lado da célula.

0.3.2 Básico (modo de comando)

- `Enter` habilita a edição da célula selecionada
- `Esc` retorna ao modo de comando
- As setas `Cima`e `Baixo` permitem a navegação entre as células
- `Shift` + `Enter` permite executar uma célula e selecionar a célula abaixo
- `Ctrl` + `Enter` apenas executa a célula selecionada
- `Alt` + `Enter` para executar uma célula e adicionar outra célula abaixo
- Em casos de dúvida a tecla `h` mostra um painel de ajuda com essas e outras combinações

0.3.3 Outros (modo de comando)

- `dd` delete a célula corrente
- `z` desfaz algo feito no modo de comando
- `c` copia a célula corrente
- `a` adiciona uma célula nova acima
- `b` adiciona uma célula nova abaixo
- `c` copia a célula corrente

- `x` recorta a célula corrente
- `v` cola a célula abaixo
- `ke j` movimenta entre as células para cima e para baixo, respectivamente
- `p` abre um seletor dos comandos possíveis no Jupyter
- `Shift` + `cima|baixo` seleciona células
- `m` altera a célula para o tipo para *Markdown*
- `y` altera a célula para o tipo para *Code*

- `Shift` + `m` mescla as células selecionadas
- `f` abre um "localizar e substituir"

0.3.4 Modo de edição

- `Ctrl` + `[` e `Ctrl` + `]` para indentar a linha na célula
- `Shift` + `Ctrl` + `-` para dividir a célula em duas a partir do cursor.
- `Shift` + `Enter` executa a célula e vai para a próxima
- `Ctrl` + `Enter` apenas executa a célula
- `Alt` + `Enter` para executar uma célula e inserir um nova abaixo
- `Tab` habilita o autocompletar

- `Shift` + `Tab` abre um menu com as orientações do comando (docstrings)

0.3.5 Multicursor

- Segure `Alt` e selecione com o mouse as linhas que se quer editar.

0.3.6 Vamos experimentar?

[2]:
```
2+2
3+3
```

[2]:
```
6
```

[3]:
```
print('Testando o Jupyter')
```

```
Testando o Jupyter
```

[4]:
```
import numpy as np
```

[5]:
```
teste = 2
```

[6]:
```
teste
```

[6]:
```
2
```

0.3.7 Um pouco sobre a exportação de documentos.

[7]:
```
!jupyter nbconvert --output-dir="./docs/" --to latex Nocoes.ipynb
```

```
[NbConvertApp] Converting notebook Nocoes.ipynb to latex
[NbConvertApp] Writing 31025 bytes to docs/Nocoes.tex
```

[8]:
```
!jupyter nbconvert --output-dir="./docs/" --to html Nocoes.ipynb
```

```
[NbConvertApp] Converting notebook Nocoes.ipynb to html
[NbConvertApp] Writing 577847 bytes to docs/Nocoes.html
```

[9]:
```
!jupyter nbconvert --output-dir="./docs/" --to slides Slide.ipynb
```

```
[NbConvertApp] Converting notebook Slide.ipynb to slides
[NbConvertApp] Writing 575555 bytes to docs/Slide.slides.html
```

0.3.8 Recurso adicional (executar outro notebook)

Forma diferente de se separar e organizar códigos muito grandes. > %run ./imported.ipynb

[10]:
```
funcao('Oiiii')
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
/tmp/ipykernel_407/1064730495.py in <module>
----> 1 funcao('Oiiii')

NameError: name 'funcao' is not defined
```

[ ]:
```
%run ./imported.ipynb
funcao('Oiiii'), z
```

0.4 Continua em ...

- Noções básicas e de navegação do Jupyter
  - Markdown e outras marcações
  - Exemplos