Universidade do Minho
Licenciatura em Física

Tarefa 9

Professores:	Cristiana Neto, Pedro Oliveira, Vitor Alves
Disciplina:	Linguagens para Computação Numérica
Tema:	Gráficos
Data:	Abril de 2022

1 Introdução

Na aula anterior abordamos uma biblioteca para manipulação e visualização de dados em formato de tabela. Neste módulo vamos introduzir a biblioteca `matplotlib` para visualização de dados em forma de gráficos. Muitas vezes, estas duas bibliotecas são usadas em conjunto.

Tal como temos vindo a fazer, o primeiro passo para a utilização deste biblioteca é a sua importação. Nós queremos utiliza-la para criar gráficos. Portanto, teremos de dar indicação ao Python para importar da biblioteca `matplotlib` a parte de criar gráficos (`pyplot`).

```
import matplotlib.pyplot as plt
```

Após a importação da biblioteca, é necessário criar listas com os dados que queremos utilizar no nosso gráfico.

1.1 Tipos de Gráficos

Existem vários tipos de gráficos que podem ser construidos. Alguns destes serão mostrados de seguida.

Para cada exemplo que se segue, considere o DataFrame sobre a taxa de natalidade e mortalidade da aula anterior.

In [1]:
```
import pandas

população = pandas.DataFrame({
    'Ano': [ 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018 ],
    'Natalidade': [ 9.2, 8.5, 7.9, 7.9, 8.3, 8.4, 8.4, 8.5 ],
    'Mortalidade': [ 9.7, 10.2, 10.2, 10.1, 10.5, 10.7, 10.7, 11.0 ]
})
```

1.1.1 Gráfico de Dispersão

Os gráficos de dispersão `plt.scatter()` são uma boa abordagem para um gráfico inicial, quando ainda não se conhecem bem os dados.

Vamos mostrar num gráfico os valores da Taxa Bruta de Mortalidade, para cada um dos anos. Para o eixo dos x, escolhemos a coluna Ano. No eixo dos y, colocamos a taxa bruta de mortalidade.

```
import matplotlib.pyplot as plt

x = população.Ano
y = população.Mortalidade
plt.scatter( x, y)
```

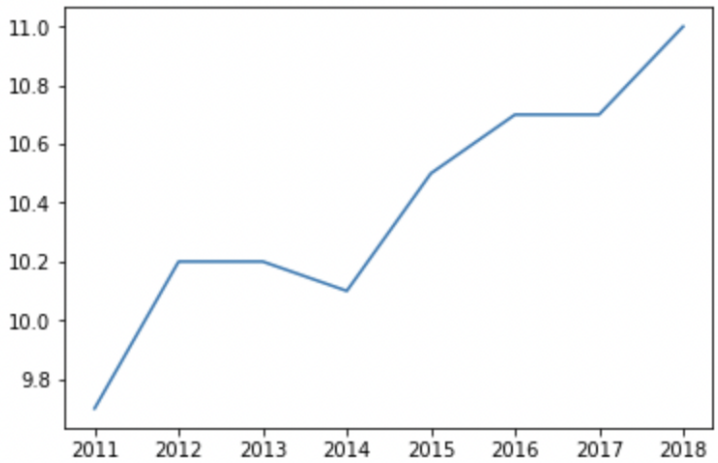
```
<matplotlib.collections.PathCollection at 0x7f3eb1eb1c90>
```

1.1.2 Gráfico de Linha

O método `plt.plot` cria um gráfico de linhas. Podem ser apresentadas um ou mais linhas.

```
plt.plot( população.Ano, população.Mortalidade)
```

```
[<matplotlib.lines.Line2D at 0x7f3eb0c16810>]
```

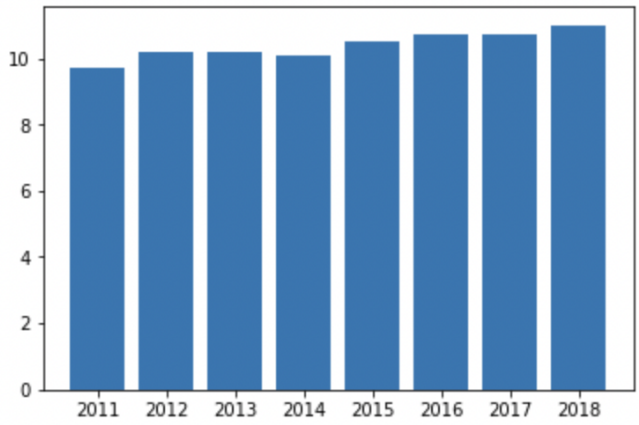


1.1.3 Gráfico de Barras

Um gráfico de barras representa os dados com barras retangulares com comprimentos e alturas proporcionais aos valores que representam e pode ser criado através do método `plt.bar`.

```
plt.bar( população.Ano, população.Mortalidade)
```

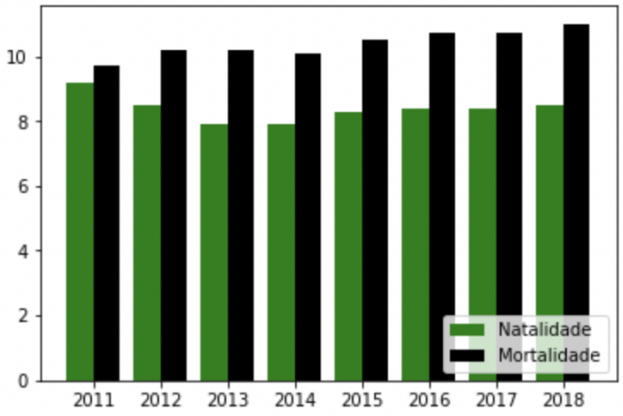
```
<BarContainer object of 8 artists>
```



Podemos ainda dar um passo à frente e juntar as duas séries, adicionando também uma pequena legenda através do método `plt.lengend`.

```
plt.bar( população.Ano-0.2, população.Natalidade, width=0.4, color="green")
plt.bar( população.Ano+0.2, população.Mortalidade, width=0.4, color="black")
plt.legend([ 'Natalidade', 'Mortalidade'], loc='lower right')
```

```
<matplotlib.legend.Legend at 0x7f3eafdd8bd0>
```

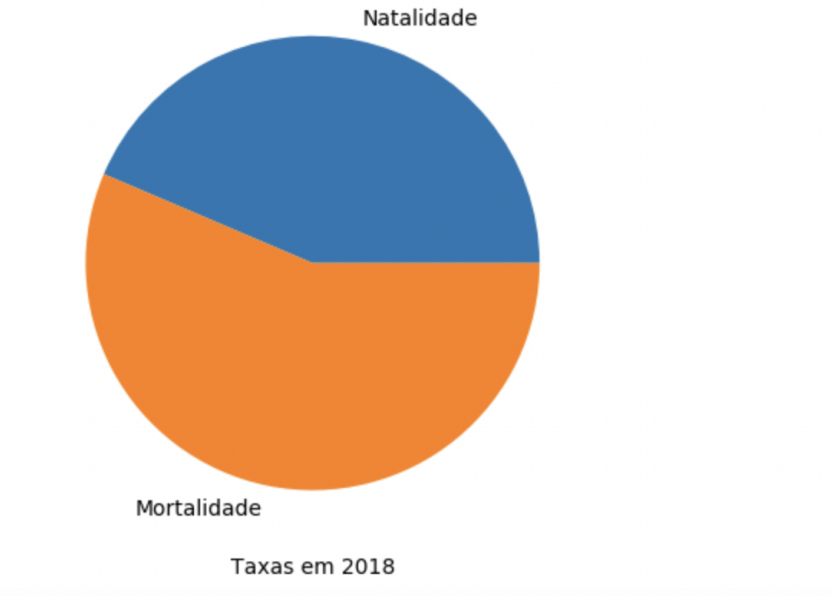


1.1.4 Gráfico de Setores ou Gráfico Circular

Estão gráficos são muito apropriados para se perceber a proporção entre variáveis e podem ser criados através do método `plt.pie`. No exemplo seguinte, é mostrada a diferença ente a taxa de natalidade e mortalidade em 2018.

```
colunas = ['Natalidade', 'Mortalidade']
valores = população[ população.Ano == 2018][ colunas ].values
plt.pie(valores[0], labels=colunas)
plt.xlabel("Taxas em 2018")
```

```
Text(0.5, 0, 'Taxas em 2018')
```



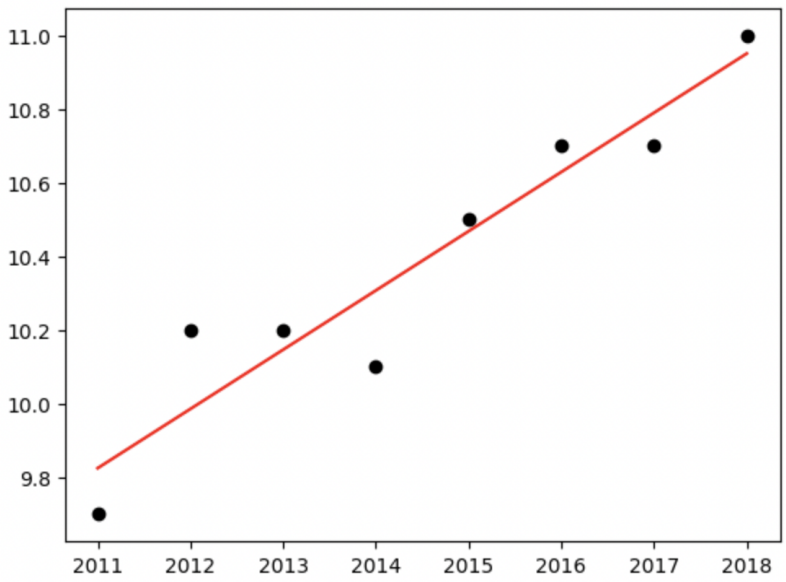
Outro exemplo em que se usa o parâmetro `explode` para destacar um dos elementos.

1.1.5 Gráfico com Duas Séries

No gráfico seguinte, juntamos duas visualizações. O eixo dos x é o mesmo. No eixo dos y, mostramos os valores da taxa de mortalidade e a relação linear entre o ano e a taxa de mortalidade bruta. A regressão linear está desenhada a vermelho, para se ver melhor.

```
plt.scatter( x, y, color="black")
plt.plot(x, m * x + b, color="red")
```

```
[<matplotlib.lines.Line2D at 0x7f3eafa74910>]
```

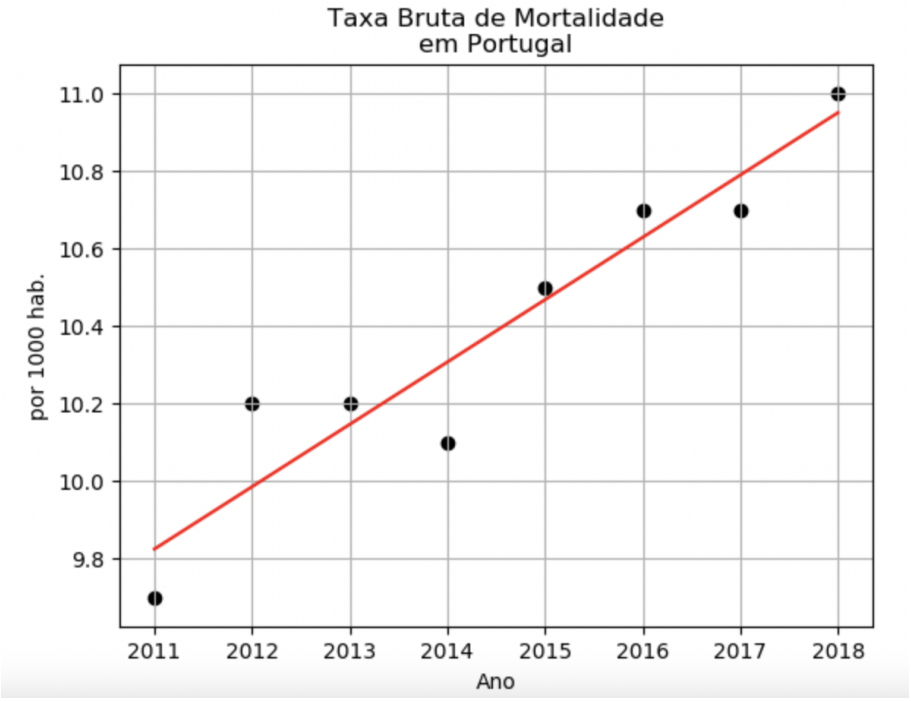


O gráfico apresentado, permite ver uma tendência clara de aumento da Taxa Bruta de Mortalidade.

Vamos melhorar o gráfico anterior, com um título em cada um dos eixos e com uma grelha.

```
plt.scatter( x, y, color="black")
plt.plot(x, m * x + b, color="red")

plt.xlabel("Ano")
plt.ylabel("por 1000 hab.")
plt.title("Taxa Bruta de Mortalidade\nem Portugal")
plt.grid()
```



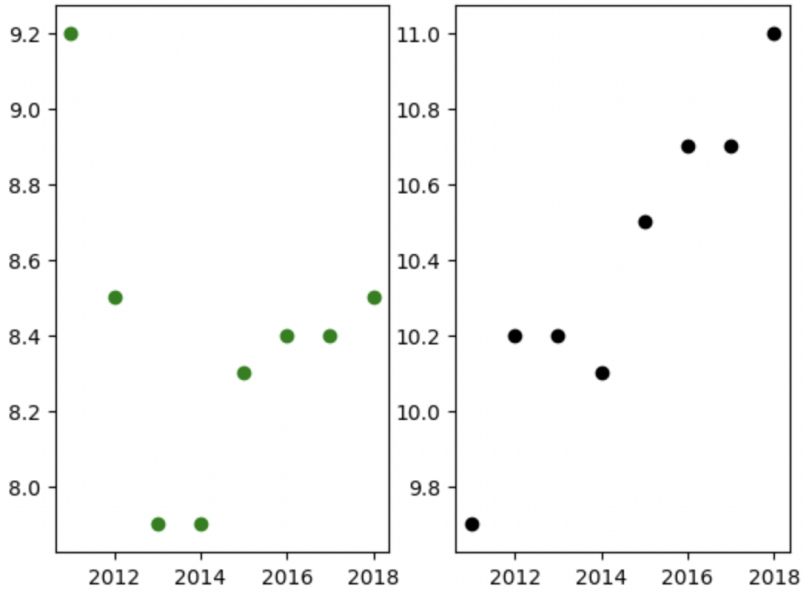
1.2 Composição de Gráficos

No exemplo seguinte, vamos colocar na mesma figura dois gráficos. Vamos ter dois subgráficos, dispostos numa linha, com duas colunas.

Repare que se usa o método `plt.subplot()` com três argumentos: os dois primeiros referem-se à geometria dos subgráficos: 1 linha, 2 colunas; o último (1 ou 2), refere-se ao índice do gráfico que vou desenhar.

```
plt.subplot(1, 2, 1)
plt.scatter( população.Ano, população.Natalidade, color="green")
plt.subplot(1, 2, 2)
plt.scatter( população.Ano, população.Mortalidade, color="black")
```

```
<matplotlib.collections.PathCollection at 0x7f3e92810150>
```



De forma análoga, se se quiserem juntar quatro gráficos na mesma figura, podemos criar uma geometria de duas linhas por duas colunas (2x2), como é ilustrado na figura seguinte.

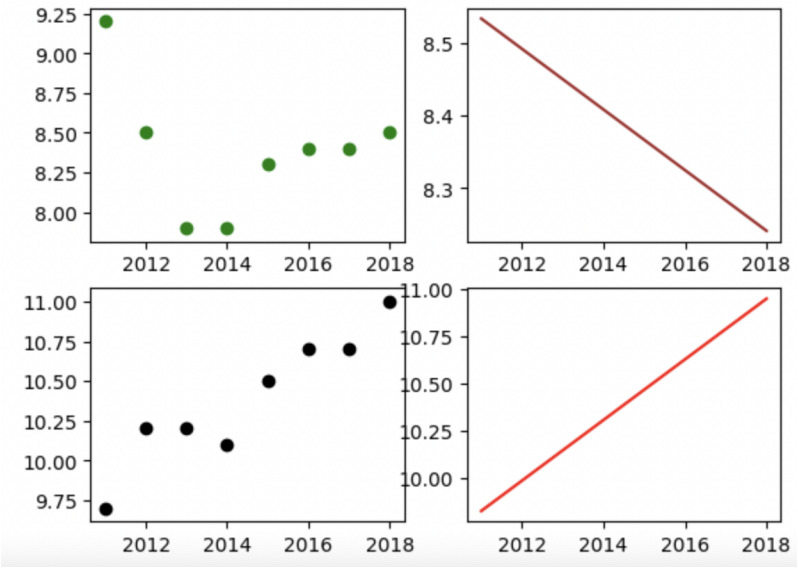
```
fig = plt.figure()
ax_natal = fig.add_subplot(2, 2, 1)
ax_natal.scatter( população.Ano, população.Natalidade, color="green")

ax_regressao_n = fig.add_subplot(2, 2, 2)
ax_regressao_n.plot(x, m_n * x + b_n, color="brown")

ax_mortal = fig.add_subplot(2, 2, 3)
ax_mortal.scatter( população.Ano, população.Mortalidade, color="black")

ax_regressao = fig.add_subplot(2, 2, 4)
ax_regressao.plot(x, m * x + b, color="red")
```

```
[<matplotlib.lines.Line2D at 0x7f3e92583790>]
```



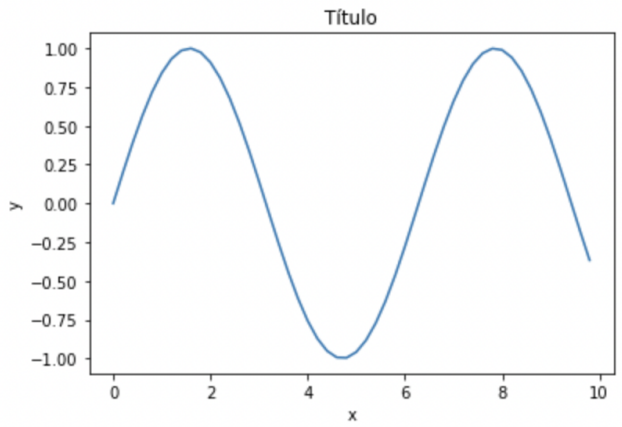
Como é possível verificar, este último exemplo apresenta uma notação ligeiramente diferente do que temos vindo a ver. Esta notação torna mais fácil e mais clara a manipulação dos diferentes objectos da figura. Passa a ter objectos que representam a figura em si (*fig*), e cada um dos seus *Axes* (`ax_natal, ax_regressao_n, ax_mortal, ax_regressao`).

O objeto *Figure* criado mantém o registo de um ou mais objetos *Axes*, um conjunto de elementos (títulos, legendas, etc) e a tela (canvas) onde tudo é desenhado. Uma figura pode conter qualquer número de *Axes*, mas para que ela seja útil, pelo menos um é necessário.

O objeto *Axe* é o que pode ser de fato pensado como "um gráfico", sendo a região da imagem que contém o espaço de dados. Cada objeto *Axes* pode pertencer a apenas uma *Figure* e contém dois (ou três, em gráficos 3D) objetos do tipo *Axis* que controlam os limites dos dados (eixos). Cada *Axes* tem um título (definido via `set_title()`), um rótulo do eixo x (definido via `set_xlabel()`) e um rótulo do eixo y (`set_ylabel()`), como demonstrado na figura seguinte:

```
import numpy as np

x = np.arange(0, 10, 0.2)
y = np.sin(x)
fig, ax = plt.subplots()
ax.plot(x, y)
ax.set_title('Título')
ax.set_xlabel('x')
ax.set_ylabel('y')
plt.show()
```



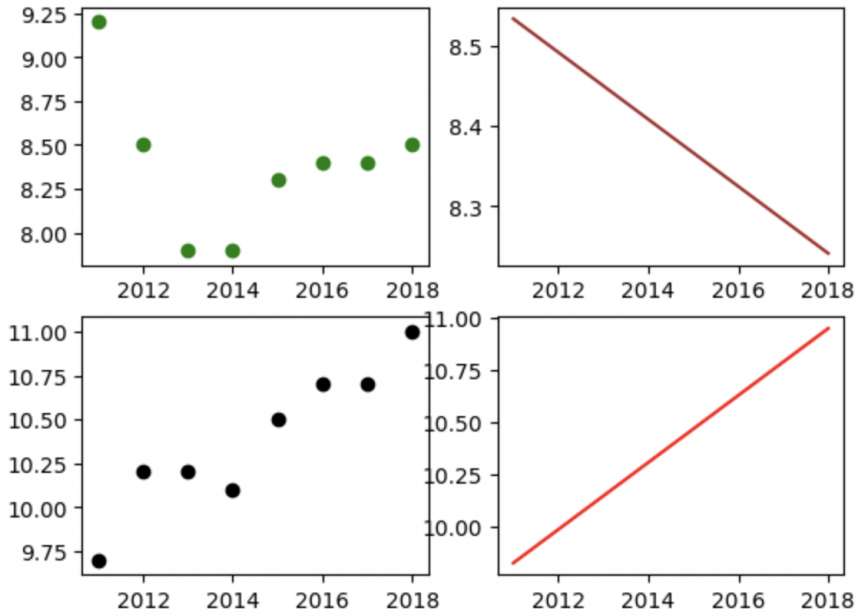
À medida que for construíndo gráficos mais complexos, verá que esta notação torna mais claro o manuseamento das diferentes componentes do gráfico.

Podemos usar uma outra forma ainda mais compacta, que cria os mesmos objectos, e que é a **forma mais indicada**. O exemplo anterior e o próximo, já usam esta forma compacta.

```
fig = plt.figure()
fig, ((ax_natal, ax_regressao_n), (ax_mortal, ax_regressao)) = plt.subplots(2, 2)
ax_natal.scatter( população.Ano, população.Natalidade, color="green")
ax_regressao_n.plot(x, m_n * x + b_n, color="brown")
ax_mortal.scatter( população.Ano, população.Mortalidade, color="black")
ax_regressao.plot(x, m * x + b, color="red")
```

```
[<matplotlib.lines.Line2D at 0x7f06b2fbae90>]

<Figure size 640x480 with 0 Axes>
```



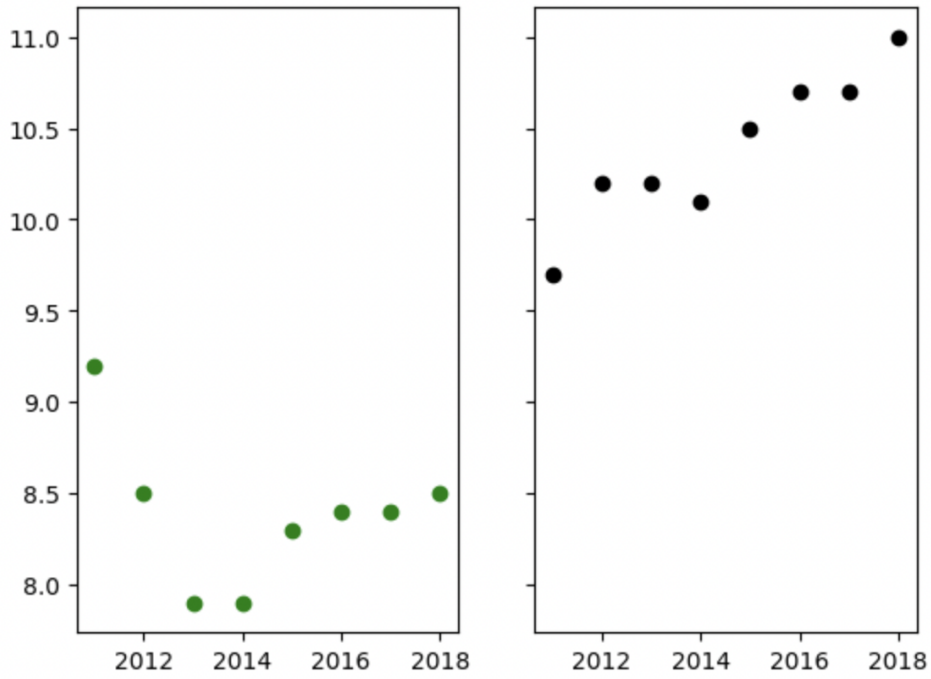
1.2.1 Partilhar eixos

Repare que até aqui temos juntado gráficos na mesma figura, mas sempre independentes. Neste caso, pode-se melhorar a informação passada pelos gráficos, usando os mesmos valores em y. Ou seja, tornando o eixo dos y comum aos dois gráficos da figura.

```
fig, (ax_natal, ax_mortal) = plt.subplots(1, 2, sharey=True)

ax_natal.scatter( população.Ano, população.Natalidade, color="green")
ax_mortal.scatter( população.Ano, população.Mortalidade, color="black")
```

```
<matplotlib.collections.PathCollection at 0x7f3e923463d0>
```



1.3 Formatação dos Gráficos

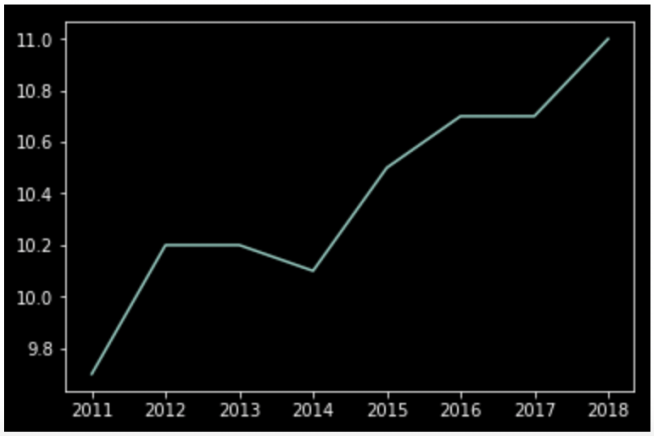
Podemos fazer inúmeros ajustes nos nossos gráficos, como já vimos em alguns dos exeplos anteriores: colocar título, legenda, nomear os eixos x e y, adicionar uma grade, escolher o tipo de traço, as cores envolvidas, entre outros.

- Adicionar título:
  `plt.title("O meu gráfico")`
- Nomear os eixos:
  `plt.xlabel("Anos")`
  `plt.ylabel("Taxa")`
- Adicionar uma grade (grid) ao fundo do gráfico:
  `plt.grid()`

A apresentação do gráfico pode ser personalizada. A maneira mais fácil é escolher uma dos múltiplos estilos existentes.

```
plt.style.use('dark_background')
plt.plot( população.Ano, população.Mortalidade)
```

```
[<matplotlib.lines.Line2D at 0x7f3eafcc74d0>]
```



Pode-se também alterar as propriedades da figura que é criada, quando nada é dito. No exemplo seguinte, vamos explicitamente criar uma figura com o dobro da largura, para acomodar melhor os dois gráficos lado a lado.

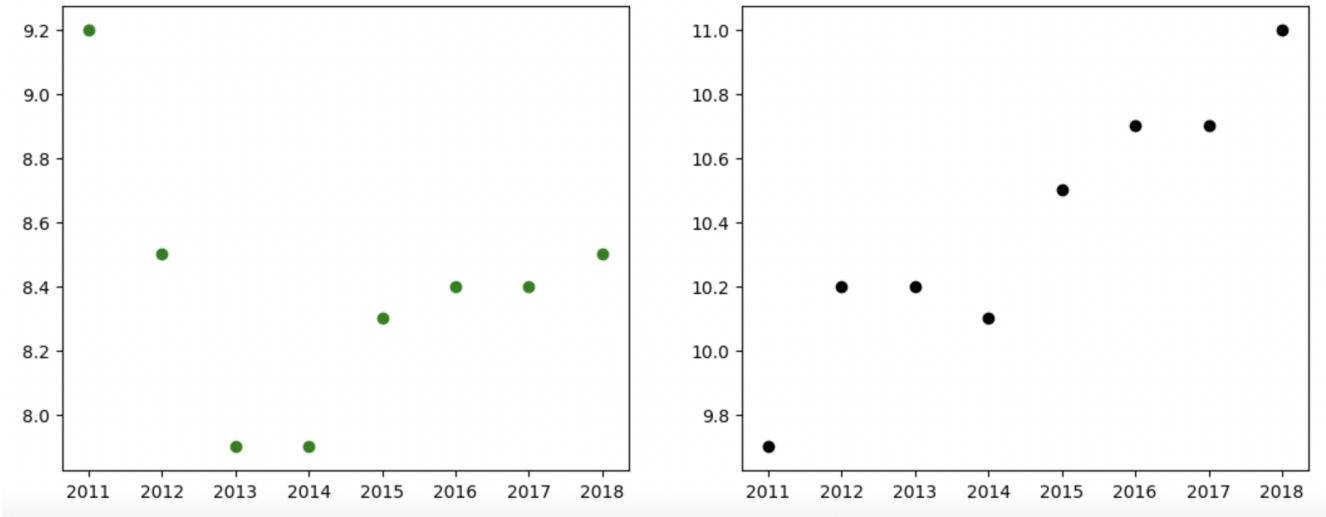
As dimensões predefinidas são: 6.4"por 4.8"polegadas (sendo 1"igual a 25.4 mm). Ou seja, em mm, um gráfico tem as dimensões 162.56 mm de largura por 121.92 mm de altura.

Vamos duplicar a largura predefinida, para 12.8".

```
fig, (ax_natal, ax_mortal) = plt.subplots(1, 2, figsize=( 12.8, 4.8))

ax_natal.scatter( população.Ano, população.Natalidade, color="green")
ax_mortal.scatter( população.Ano, população.Mortalidade, color="black")
```

```
<matplotlib.collections.PathCollection at 0x7f3e928eb250>
```



2 Exercícios

Com esta ficha de trabalho pretende-se apresentar alguns problemas exemplificativos da utilização do pandas.

1. Utilizando o DataFrame da taxa de natalidade e mortalidade fornecido anteriormente, apresente o gráfico de dispersão da Taxa Bruta de Natalidade. Escolha ainda um dos

estilos disponíveis, diferente do exemplo dado anteriormente, e aplique-o ao gráfico. Antes de iniciar os próximos exercícios, retorne para o estilo clássico.

2. Sabendo que o FC Vizela, até à data, conseguiu os seguintes resultados na Primeira Liga:

   - 7 vitórias
   - 11 empates
   - 13 derrotas

   Faça um gráfico de setores que represente a proporção entre vitórias, empates e derrotas, dando destaque às vitórias.

3. O ficheiro `clientes.csv` apresenta alguns dados sobre os clientes de uma empresa (salário, idade, etc...). Carregue estes dados através do pandas, analize os dados que estão presentes e elabore os seguintes gráficos:

   - Gráfico de barras com o número de clientes por país. Ajuste os eixos, colocando: eixo do y - valores entre 0 e 7000, separados de 1000 em 1000 e tamanho da letra a 12; eixo do x - palavras com uma rotação de 45º e tamanho da letra a 12. O título da iamgem deve ser 'Número de Clientes' com tamanho de letra 14.
   - Crie um histograma com o saldo dos clientes com 12 caixas (*bins*) e com valores entre 25000 e 225000. Este deve estar ainda com a cor cinza escuro e com o título 'Distribuição no Saldo (25000 - 225000)' com tamanho de letra 14.
   - Selecine uma *sample* de 200 casos e crie um gráfico de disperção com 2 *plots*: um com a pontuação de crédito por idade na França e outro na Alemanha. O título da imagem deve ser 'França vs Alemanha' com tamanho de letra 14 e uma legenda no canto inferior esquerdo com tamanho de letra 12.
   - Crie um gráfico de barras com 2 *plots* na mesma linha mas em duas colunas distintas que partilham o eixo do y. A figura deve ter 8 de largura e 5 de altura. Um dos *subplots* deve conter o número de clientes por país e outro o número de clientes por número de produtos, ambos com os repetivos títulos ('Países' e 'Número de Produtos') com tamanho de letra 12.
   - Histogramas 2D (`hist2d`) permitem visualizar as distribuições de um par de variáveis, proporcionando uma visão geral de como os valores dessas variáveis mudam juntos. Crie um histograma 2D a informação da pontuação de crédito por idade, com título 'Pontuação de Crédito vs Idade' com tamanho de letra 15.

4. Tendo em conta as seguintes fórmulas relativas ao lançamento de um projétil, onde $g = 9.8$, resolva as próximas alíneas:

$$x = v_0 t \cos(\theta),$$
$$y = v_0 t \sin(\theta) - \frac{1}{2} g t^2$$

   (a) Apresente numa imagem as trajetórias de um projétil lançado a partir do solo com uma velocidade inicial de 30 m/s para ângulos entre 30º e 60º (de 5º em 5º). Ou seja, o gráfico deve mostrar as posições x e y do projétil. Atenção: a imagem deve conter uma legenda que não sobreponha as linhas do gráfico.

   (b) Apresente numa só imagem 2 plots distintos, um com as posições x e outro com as posições y do mesmo projétil da alínea anterior, desta vez lançado de uma altura de 20 metros, ao longo do tempo. Atenção: o eixo do y não deve conter valores negativos.

5. O momento de um objeto com massa m e velocidade v é definido de forma diferente na física clássica e na física relativista:

$$p_{clas} = m \cdot v$$
$$p_{rel} = m \cdot v \cdot \gamma, \qquad \gamma = \frac{1}{\sqrt{1 - \frac{v^2}{c^2}}}$$

A velocidade da luz é considerada $c \approx 3 \times 10^8$ m/s.

Considere m = 7 kg, construa um único gráfico com as duas funções, com as velocidades igualmente espaçadas no intervalo $v \in [0c, 0.9c] = [0$ m/s, 2.7×10^8 m/s$]$. Mostre a grelha do gráfico, coloque as respetivas *labels* nos eixos, mostre a legenda no canto inferior direito.

6. A lei de Planck descreve quanta energia um corpo negro (geralmente uma estrela) emite em diferentes comprimentos de onda de radiação eletromagnética. A lei é dada como:

$$B(\lambda) = \frac{2hc^2}{\lambda^5} \frac{1}{\mathrm{e}^{\frac{hc}{\lambda k T}} - 1}$$

onde T é a temperatura da estrela, $h = 6.62 \times 10^{-34}$ J s é a constante de Planck, $k = 1.38 \times 10^{-23} J/K$ é a constante de Boltzmann, e a velocidade da luz ainda é $c \approx 3 \times 10^8$ m/s. Os gráficos que resultam desta função são chamados de *curvas de Planck*, e são uma aparição comum nos livros de física.

(a) Use a temperatura do sol, $T = 5800$ K, e trace $B(\lambda)$ para os comprimentos de onda no intervalo $\lambda \in [10$ nm, 3000 nm$]$. Note que estes comprimentos de onda estão em nanometros, enquanto que a função utiliza comprimentos de onda em metros. A linha deve ser amarela.

(b) Inclua a curva de Planck para a estrela Alpha Centauri A ($T = 2000$ K) no mesmo gráfico que o sol, com cores distintas. Estas devem partilhar o eixo do x, mas, por uma questão de escala, devem ter eixos y diferentes. Coloque o eixo do y da mesma cor que a respetiva linha.

(c) A lei de de Wien diz-nos o comprimento de onda onde podemos encontrar o pico da curva de Planck é dado a partir da seguinte equação:

$$\lambda_{max} = \frac{b}{T}$$

onde $b = 2.9 \times 10^{-3} K \times m$. Expanda o gráfico de b) adicionando duas linhas verticais tracejadas em $x = \lambda_{max}$ para cada uma das duas estrelas, de modo a confirmar que as linhas correspondem ao pico das curvas. Dica: pode fazer linhas verticais com a função `axvline(x= ...)`.