

# AGG0012 – Problemas Integrados em Ciências da Terra II

Diferenciação numérica  
Victor Sacek

# Lidando com arrays do numpy

```
In []: x = np.arange(0.,100.,10.)
```

```
In []: print(x)
[  0.  10.  20.  30.  40.  50.  60.  70.  80.  90.]
```

```
In []: x[0]
Out[]: 0.0
```

```
In []: x[5]
Out[]: 50.0
```

```
In []: x[9]
Out[]: 90.0
```

```
In []: x[10]
```

---

```
IndexError                                Traceback (most recent call last)
<ipython-input-32-7644c71c757c> in <module>()
----> 1 x[10]
```

```
IndexError: index 10 is out of bounds for axis 0 with size 10
```

# Lidando com arrays do numpy

```
In []: x = np.arange(0.,100.,10.)
```

```
In []: print(x)
[  0.  10.  20.  30.  40.  50.  60.  70.  80.  90.]
```

```
In []: x[0]
Out[]: 0.0           primeiro valor do vetor x
```

```
In []: x[5]
Out[]: 50.0          sexto valor do vetor x
```

```
In []: x[9]
Out[]: 90.0          décimo valor do vetor x
```

```
In []: x[10]         não existe o décimo primeiro valor!!!!
```

---

```
IndexError                                Traceback (most recent call last)
<ipython-input-32-7644c71c757c> in <module>()
----> 1 x[10]
```

```
IndexError: index 10 is out of bounds for axis 0 with size 10
```

# Lidando com arrays do numpy

```
In []: np.size(x)  
Out[]: 10
```

```
In []: x[-1]  
Out[]: 90.0
```

```
In []: x[-2]  
Out[]: 80.0
```

```
In []: x[-3]  
Out[]: 70.0
```

# Lidando com arrays do numpy

In []: np.size(x)    “tamanho” do vetor x (número de elementos)  
Out []: 10

In []: x[-1]  
Out []: 90.0    último valor do vetor x

In []: x[-2]  
Out []: 80.0    penúltimo valor do vetor x

In []: x[-3]  
Out []: 70.0    anti-penúltimo valor do vetor x

# Lidando com arrays do numpy

```
In []: np.size(x)  "tamanho" do vetor x (número de elementos)  
Out[]: 10
```

```
In []: x[-1]  
Out[]: 90.0  último valor do vetor x
```

```
In []: x[-2]  
Out[]: 80.0  penúltimo valor do vetor x
```

```
In []: x[-3]  
Out[]: 70.0  anti-penúltimo valor do vetor x
```

```
valores de x:      [  0.  10.  20.  30.  40.  50.  60.  70.  80.  90.]
```

```
índices de x:      [  0   1   2   3   4   5   6   7   8   9  ]
```

```
índices inversos de x: [-10  -9  -8  -7  -6  -5  -4  -3  -2  -1  ]
```

# Lidando com arrays do numpy

In []: print(x[0:3])    todos os elementos do vetor x com índices de 0 até < 3  
[ 0. 10. 20.]

In []: print(x[0:7])    todos os elementos do vetor x com índices de 0 até < 7  
[ 0. 10. 20. 30. 40. 50. 60.]

In []: print(x[0:7:2])    elementos do vetor x com índices de 0 até < 7,  
[ 0. 20. 40. 60.]    “pulando” de 2 em 2.

In []: print(x[0:7:3])    elementos do vetor x com índices de 0 até < 7,  
[ 0. 30. 60.]    “pulando” de 3 em 3.

In []: print(x[2:7:3])    elementos do vetor x com índices de 2 até < 7,  
[ 20. 50.]    “pulando” de 3 em 3.

# Lidando com arrays do numpy

<pre>In []: print(x[5:]) [ 50.  60.  70.  80.  90.]</pre>	elementos do vetor x com índices de 5 até o final
<pre>In []: print(x[:5]) [  0.  10.  20.  30.  40.]</pre>	elementos do vetor x com índices do início até < 5
<pre>In []: print(x[5::2]) [ 50.  70.  90.]</pre>	elementos do vetor x com índices de 5 até o final “pulando” de 2 em 2
<pre>In []: print(x[:5:2]) [  0.  20.  40.]</pre>	elementos do vetor x com índices do início até < 5 “pulando” de 2 em 2



# Número de medalhas olímpicas brasileiras

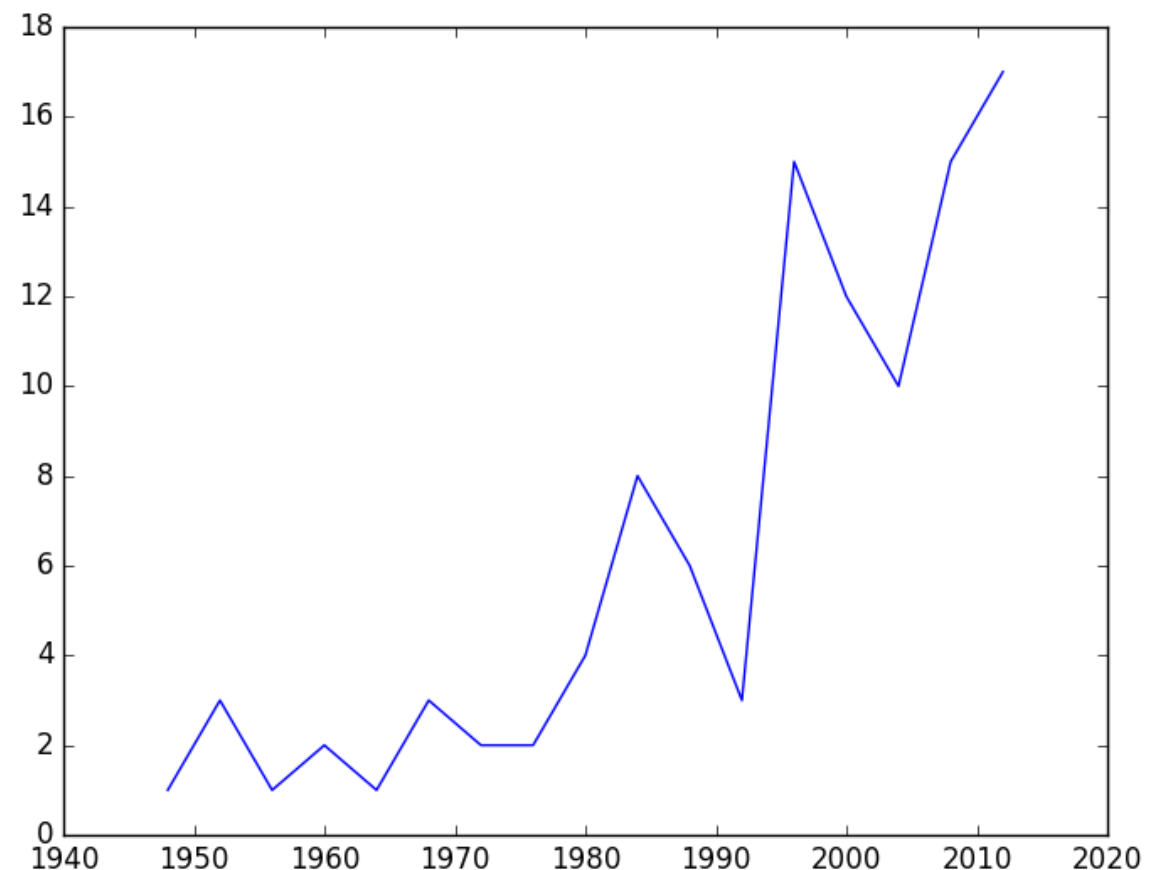
```
In []: a = np.array([1, 3, 1, 2, 1, 3, 2, 2, 4, 8, 6, 3, 15, 12, 10, 15, 17])
```

```
In []: ano = np.arange(1948,2016,4)
```

```
In []: print(ano)
```

```
[1948 1952 1956 1960 1964 1968 1972 1976 1980 1984 1988 1992 1996 2000 2004  
2008 2012]
```

```
In []: plt.plot(ano,a)
```



# Variação do nº de medalhas em olimpíadas consecutivas

```
In []: print(a)  
[ 1  3  1  2  1  3  2  2  4  8  6  3 15 12 10 15 17]
```

# Variação do nº de medalhas em olimpíadas consecutivas

```
In []: print(a)  
[ 1  3  1  2  1  3  2  2  4  8  6  3 15 12 10 15 17]
```

```
In []: print(a[1:])  
[ 3  1  2  1  3  2  2  4  8  6  3 15 12 10 15 17]
```

# Variação do nº de medalhas em olimpíadas consecutivas

```
In []: print(a)  
[ 1  3  1  2  1  3  2  2  4  8  6  3 15 12 10 15 17]
```

```
In []: print(a[1:])  
[ 3  1  2  1  3  2  2  4  8  6  3 15 12 10 15 17]
```

```
In []: print(a[:-1])  
[ 1  3  1  2  1  3  2  2  4  8  6  3 15 12 10 15]
```

# Variação do nº de medalhas em olimpíadas consecutivas

```
In []: print(a)  
[ 1  3  1  2  1  3  2  2  4  8  6  3 15 12 10 15 17]
```

```
In []: print(a[1:])  
[ 3  1  2  1  3  2  2  4  8  6  3 15 12 10 15 17]
```

```
In []: print(a[:-1])  
[ 1  3  1  2  1  3  2  2  4  8  6  3 15 12 10 15]
```

```
In [89]: print(a[1:]-a[:-1])  
[ 2 -2  1 -1  2 -1  0  2  4 -2 -3 12 -3 -2  5  2]
```

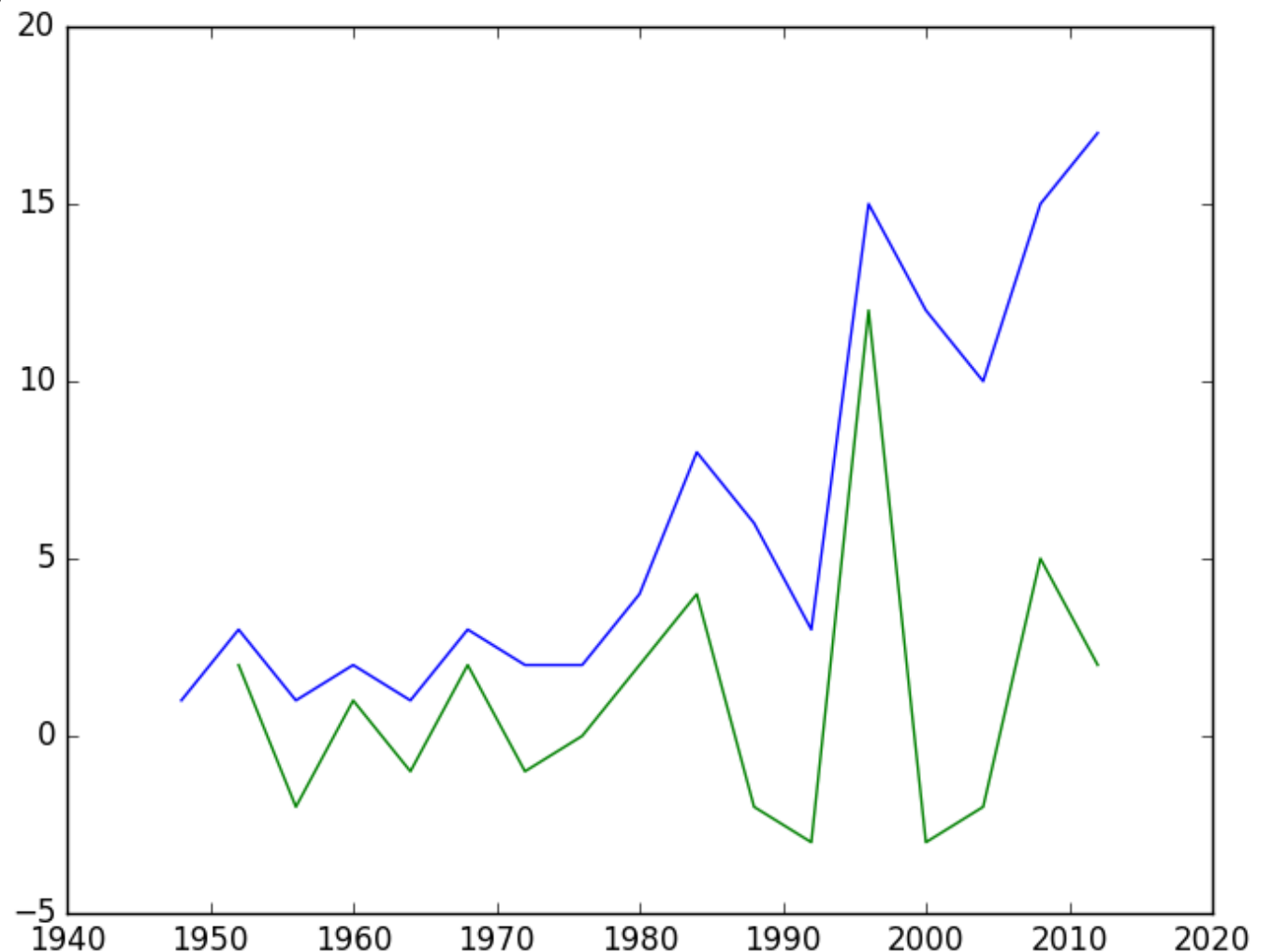
# Variação do nº de medalhas em olimpíadas consecutivas

```
In []: da = a[1:]-a[:-1]
```

```
In []: print(da)
```

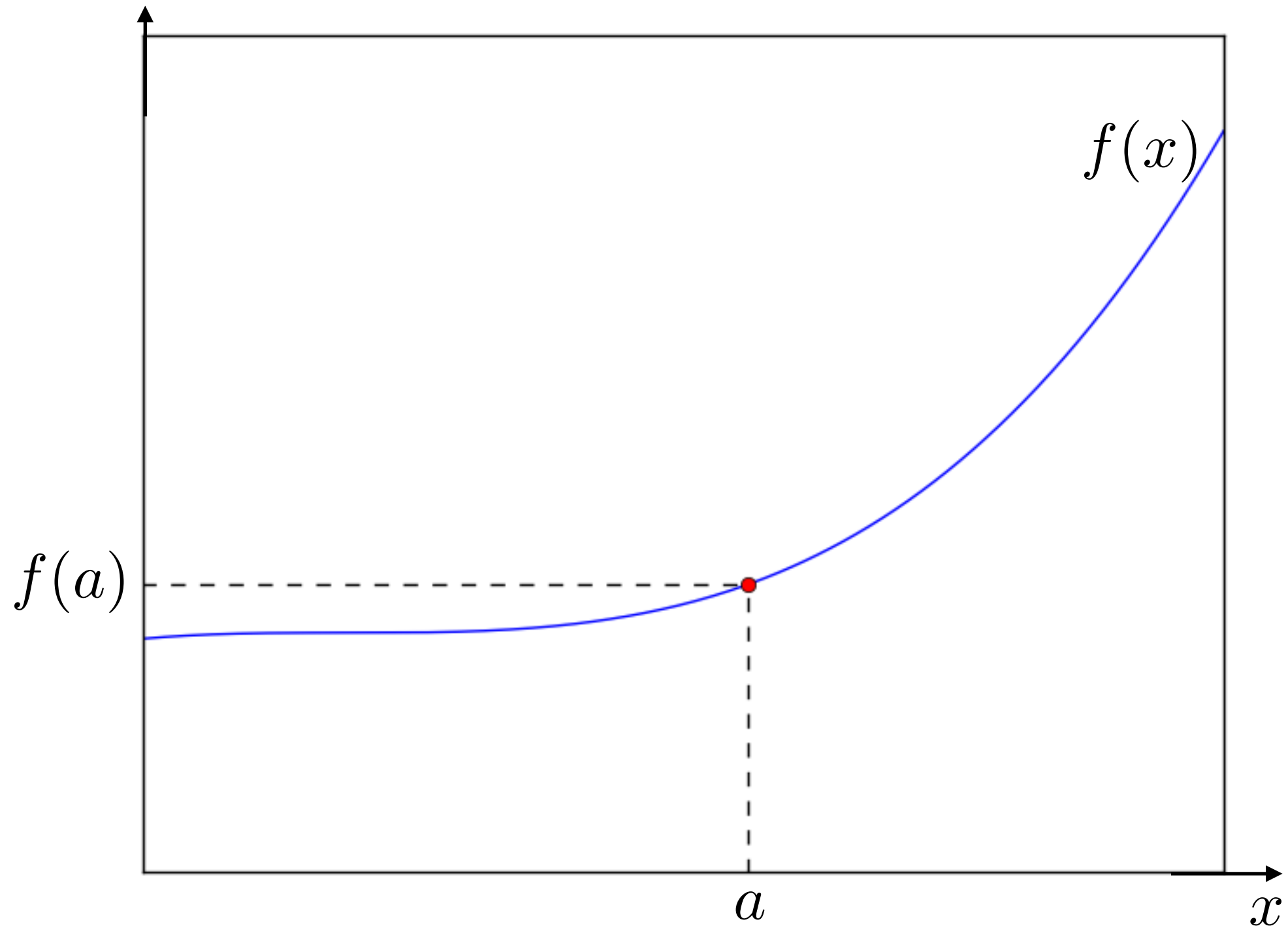
```
[ 2 -2  1 -1  2 -1  0  2  4 -2 -3 12 -3 -2  5  2]
```

```
In []: plt.plot(ano[1:],da)
```



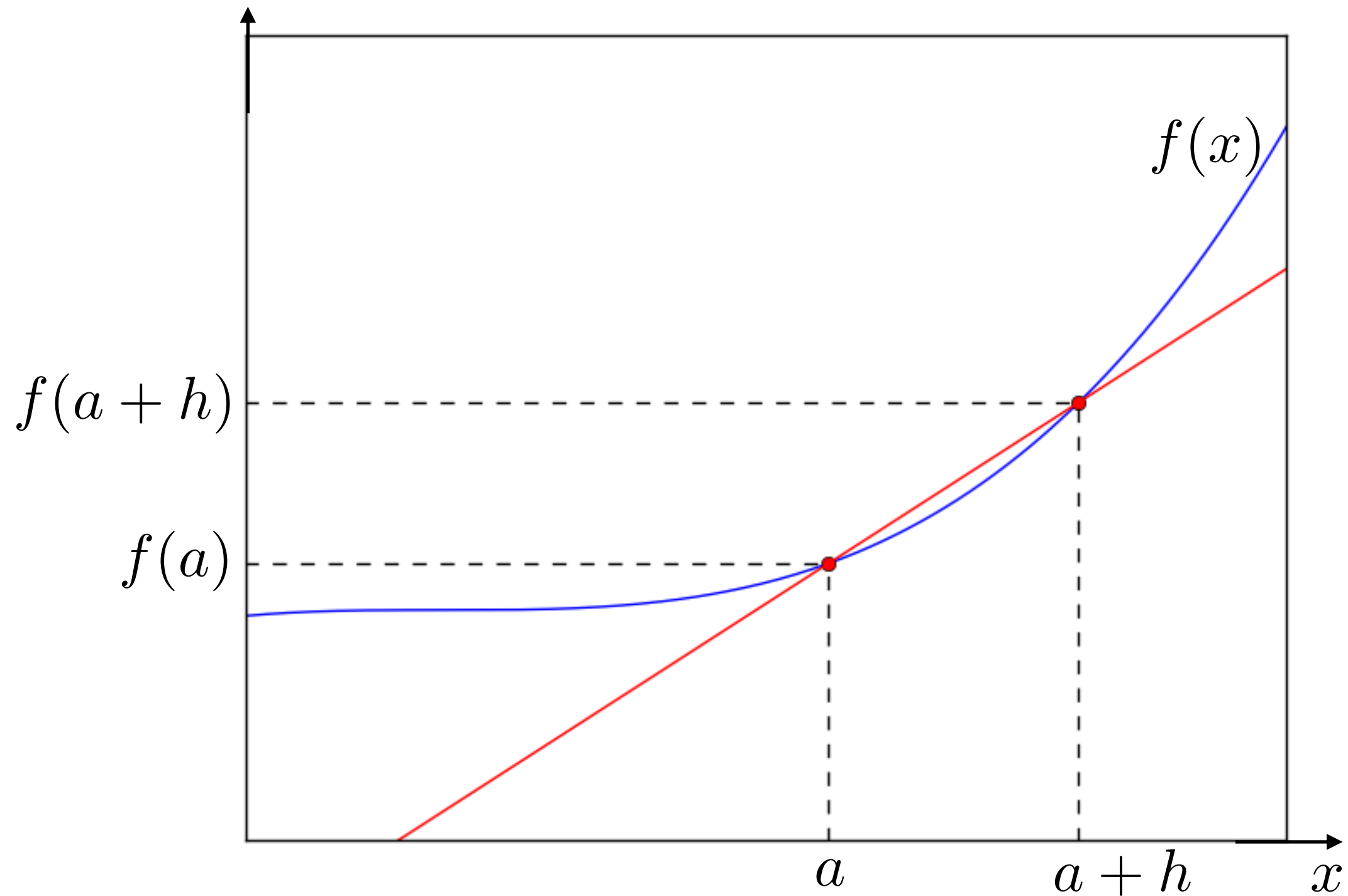
# Derivada

$$\frac{df(a)}{dx} = \lim_{h \rightarrow 0} \frac{f(a+h) - f(a)}{h}$$



# Derivada

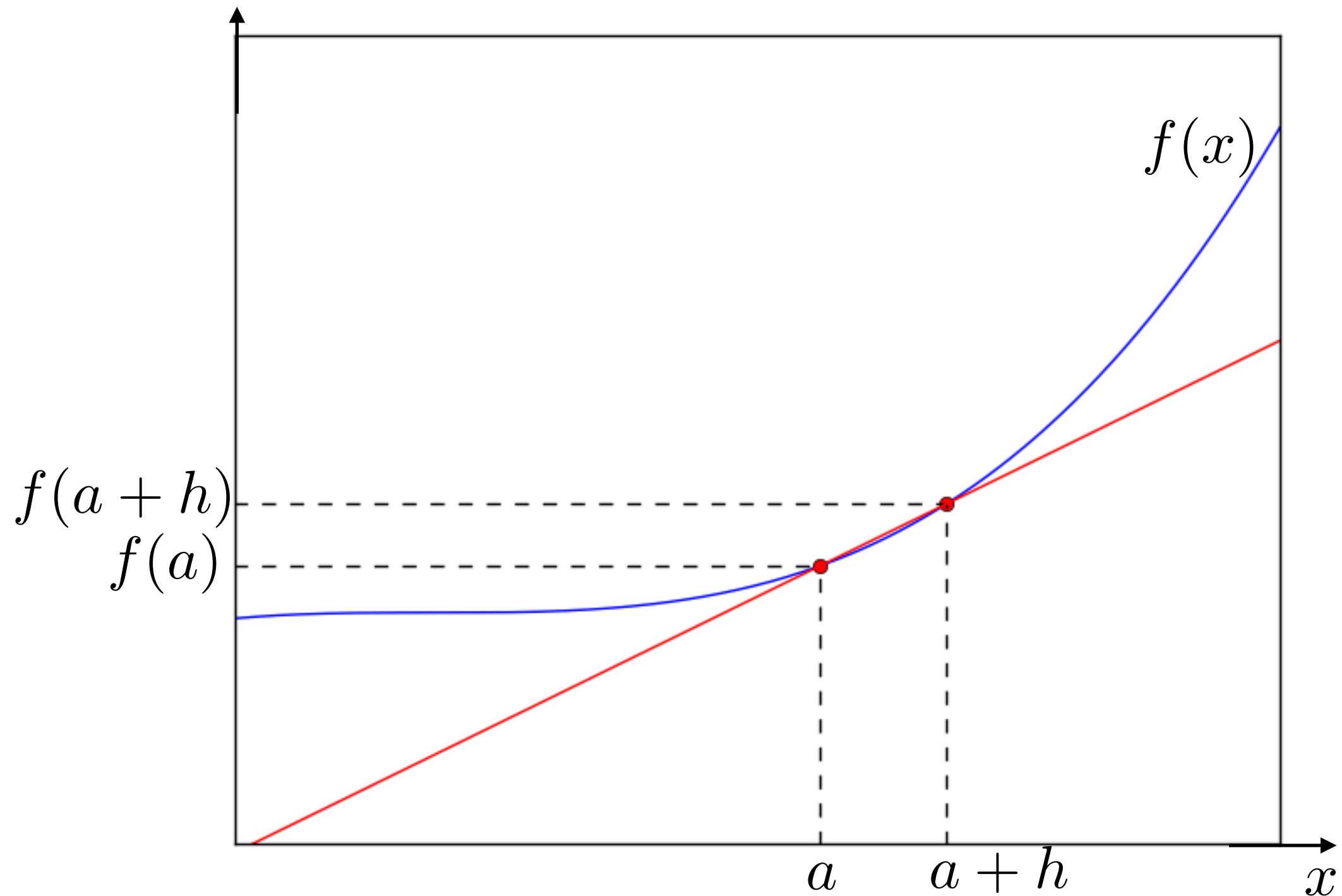
$$\frac{df(a)}{dx} = \lim_{h \rightarrow 0} \frac{f(a+h) - f(a)}{h}$$





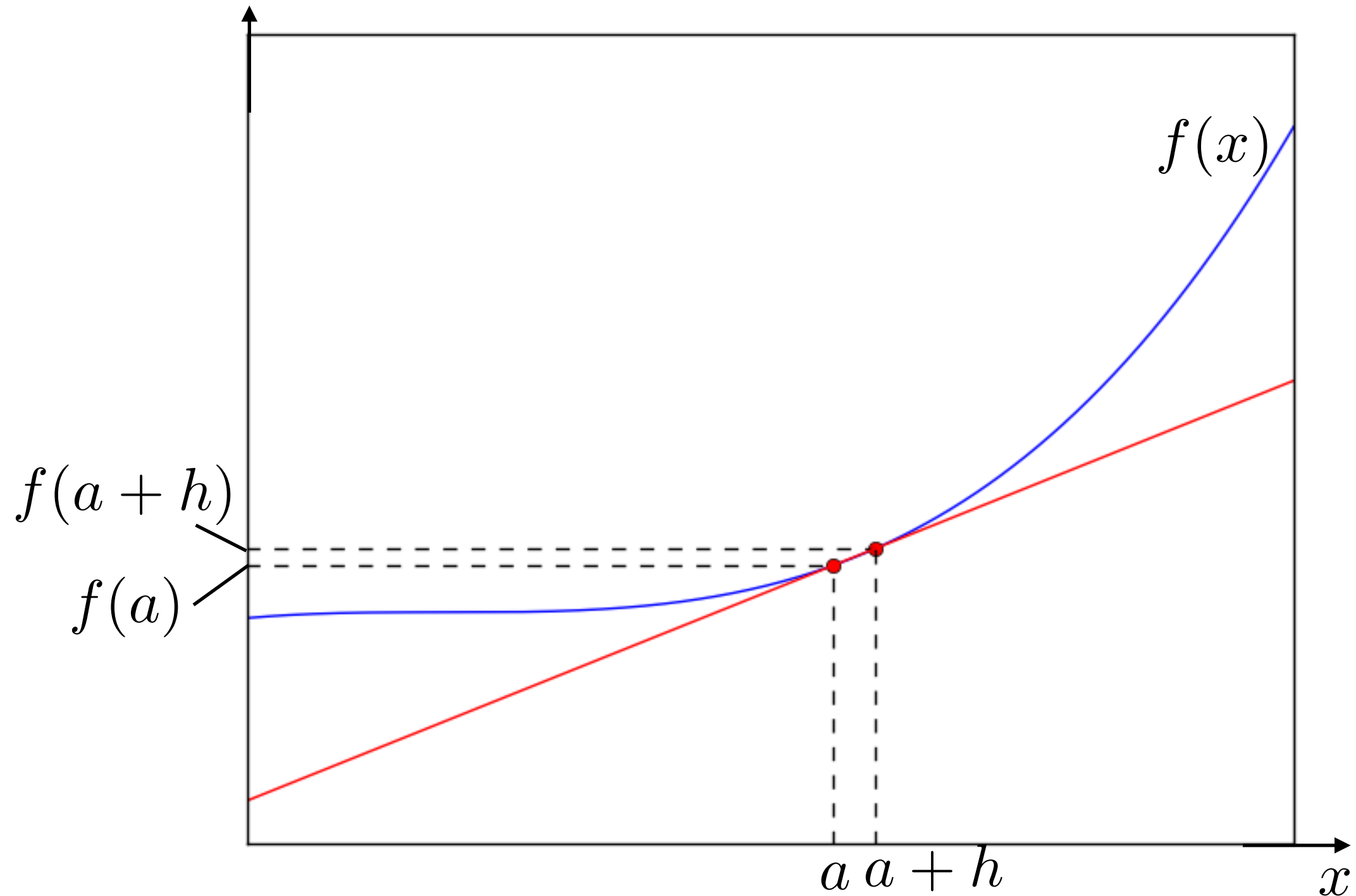
# Derivada

$$\frac{df(a)}{dx} = \lim_{h \rightarrow 0} \frac{f(a+h) - f(a)}{h}$$



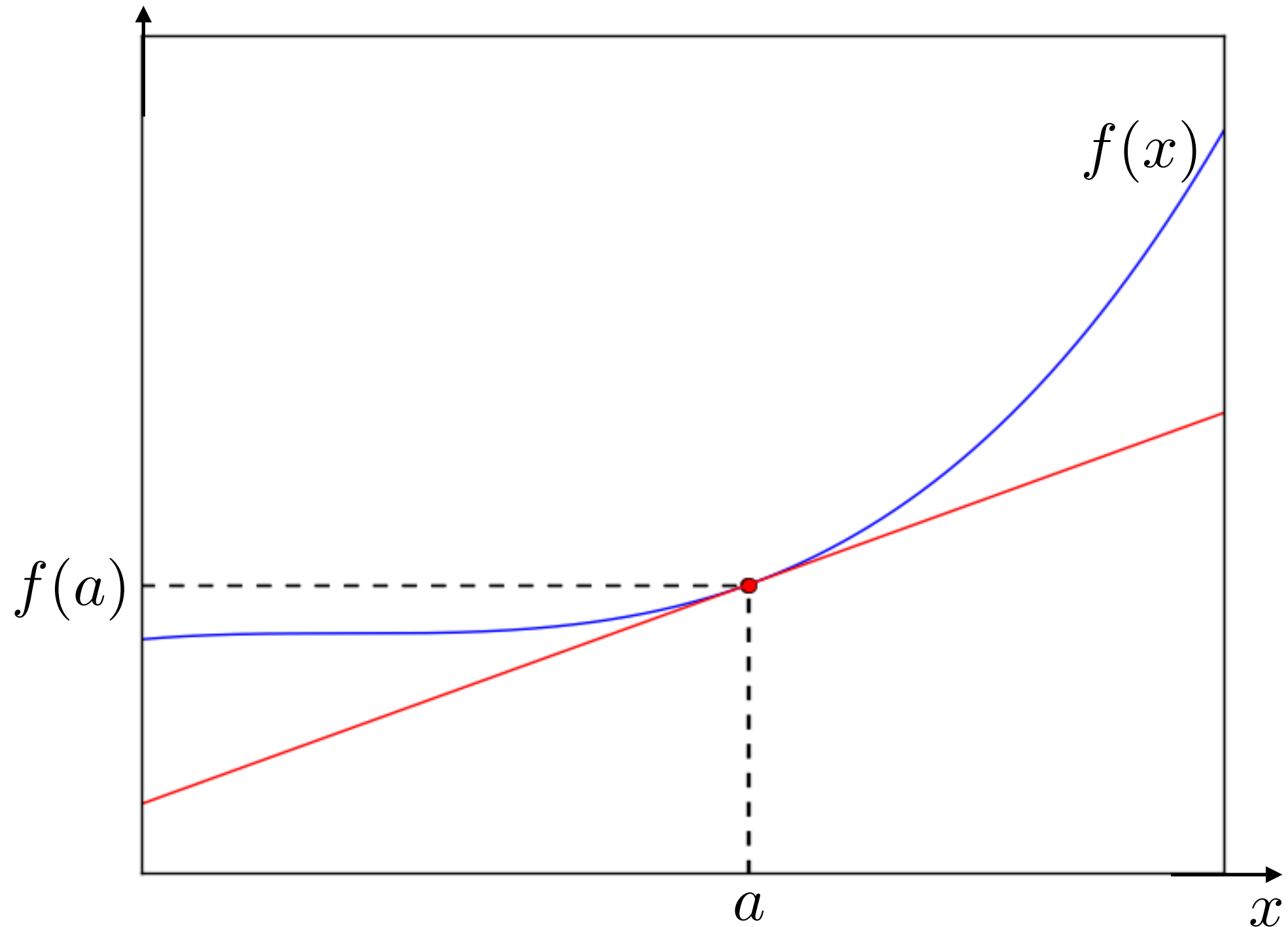
# Derivada

$$\frac{df(a)}{dx} = \lim_{h \rightarrow 0} \frac{f(a+h) - f(a)}{h}$$

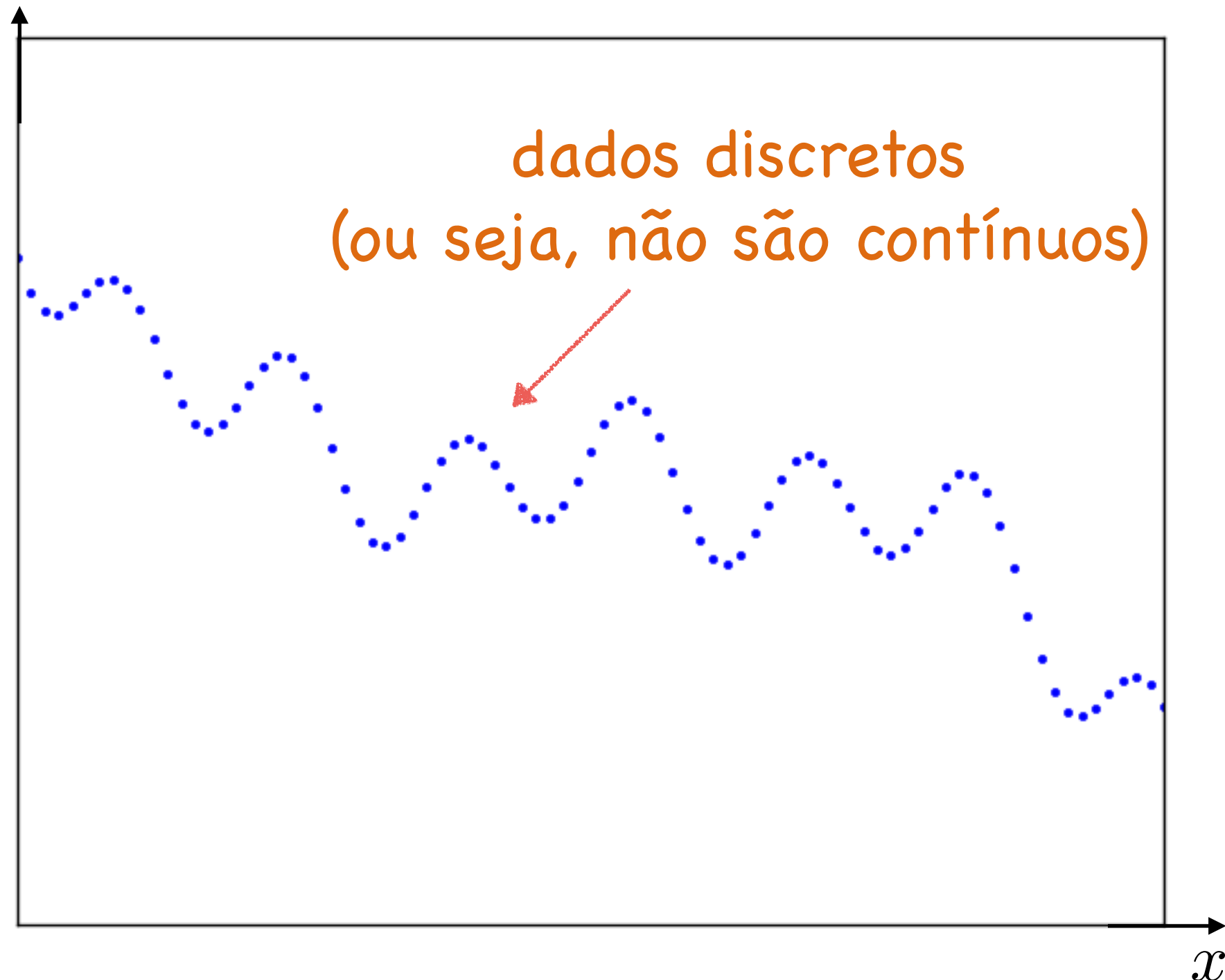


# Derivada

$$\frac{df(a)}{dx} = \lim_{h \rightarrow 0} \frac{f(a+h) - f(a)}{h}$$



Mas e se não conhecermos  
um expressão para  $f(x)$ ?



# Derivação numérica

$$\frac{df(a)}{dx} = \lim_{h \rightarrow 0} \frac{f(a+h) - f(a)}{h}$$

$$\frac{df(a)}{dx} \approx \frac{f(a+h) - f(a)}{h} = \frac{\Delta f}{\Delta x}$$



este  $h$  não vai para zero,  
mas é finito

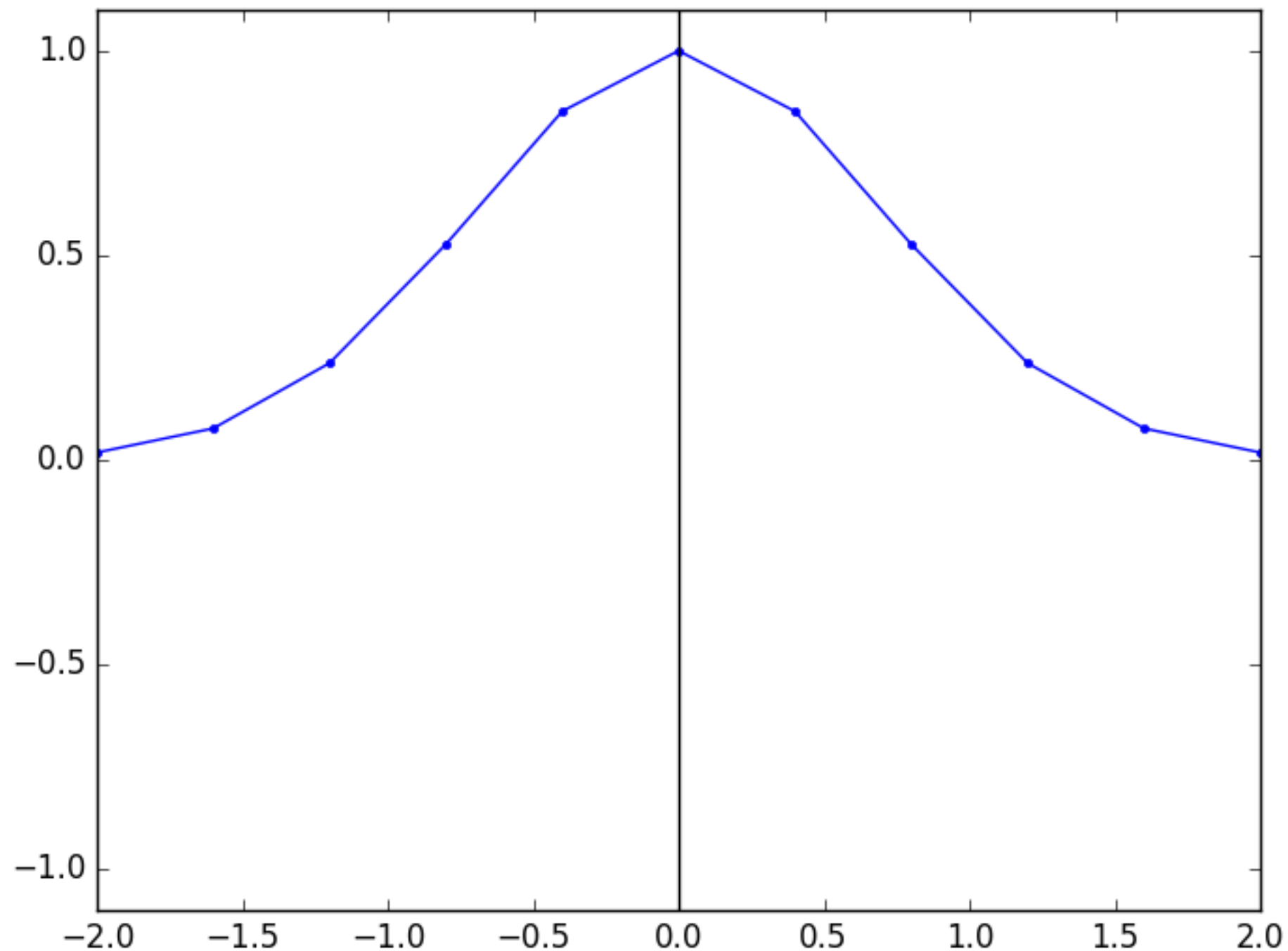
OK, mas como escrevemos  
 $\frac{\Delta f}{\Delta x}$  no Python?

Da mesma forma que fizemos com a variação  
do número de medalhas!

$\Delta f$                       `df = f[1:] - f[:-1]`

$\frac{\Delta f}{\Delta x}$                       `df/dx`

Mas em que ponto está sendo calculada a derivada numérica?



# Exemplo

```
dx = 0.4
```

```
x = np.arange(-2, 2.2, dx)
```

```
f = np.exp(-x*x)
```

```
plt.close()
```

```
plt.ion()
```

```
plt.plot(x, f, ".-")
```

```
plt.ylim([-1.1, 1.1])
```

```
plt.plot([0, 0], [-2, 2], "k")
```



# Exemplo

`df = f[1:] - f[:-1]`    diferença entre valores consecutivos

`plt.plot(x[:-1], df/dx, ".-")`

vetor x sem  
o último ponto

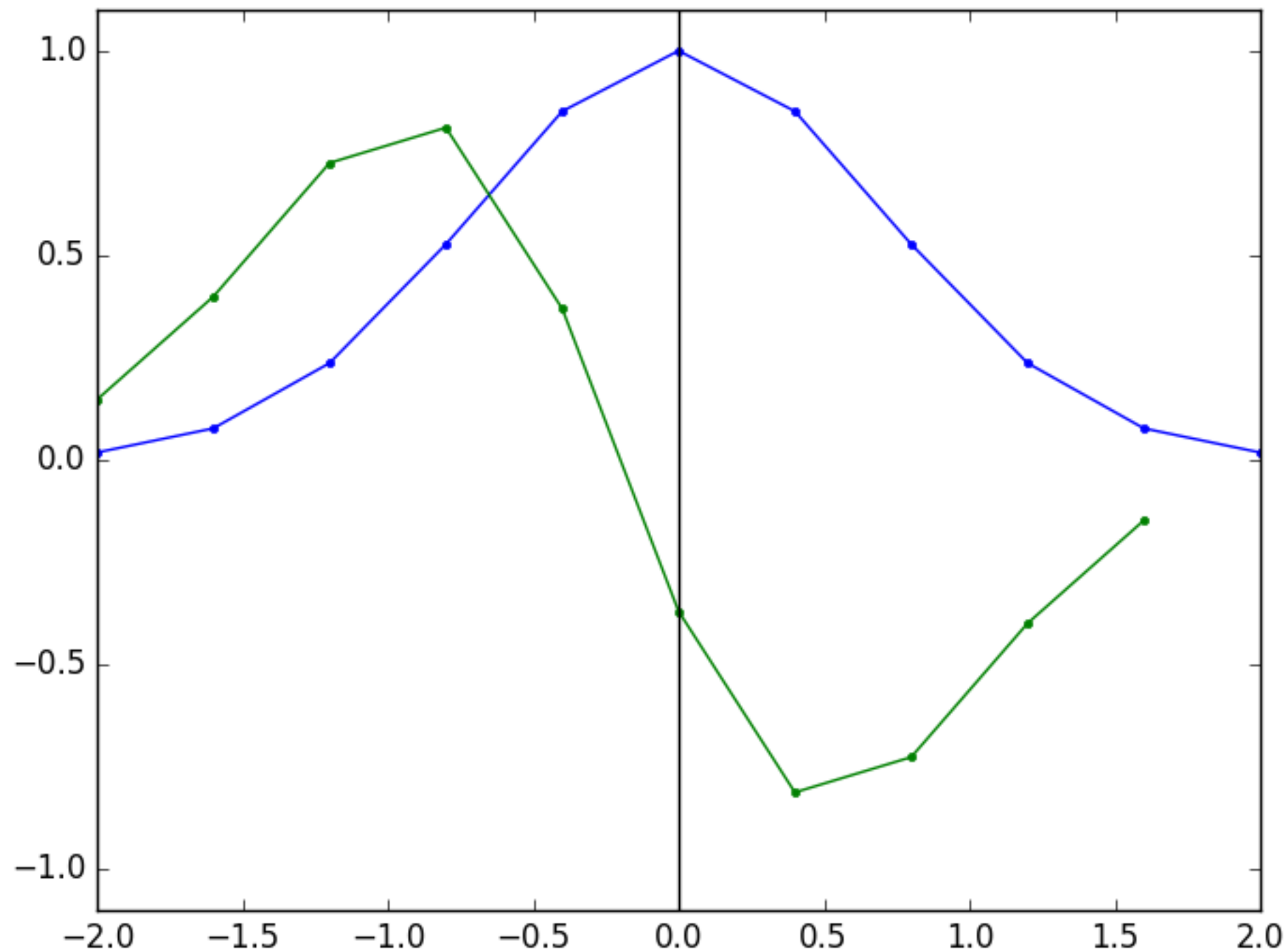


derivada  
numérica



# Mas em que ponto está sendo calculada a derivada numérica?

Se for no ponto da esquerda:



# Exemplo

`df = f[1:] - f[:-1]`    diferença entre valores consecutivos

`plt.plot(x[1:], df/dx, ".-")`

vetor x sem  
o primeiro ponto

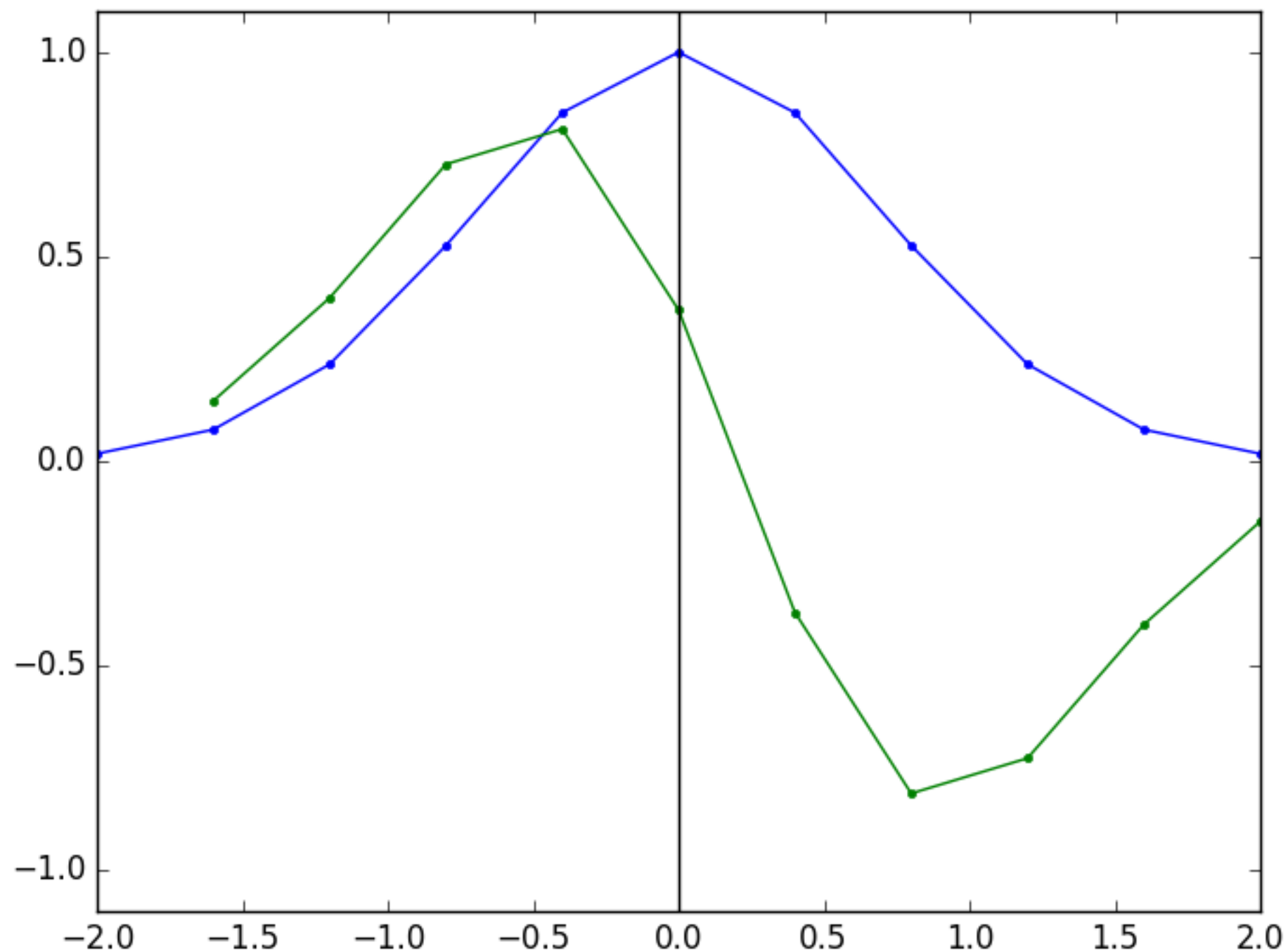


derivada  
numérica



# Mas em que ponto está sendo calculada a derivada numérica?

Se for no ponto da direita:



# Exemplo

`df = f[1:] - f[:-1]`      diferença entre valores consecutivos

`plt.plot(x[:-1] + dx/2, df/dx, ".-")`

vetor x sem  
o último ponto



derivada  
numérica

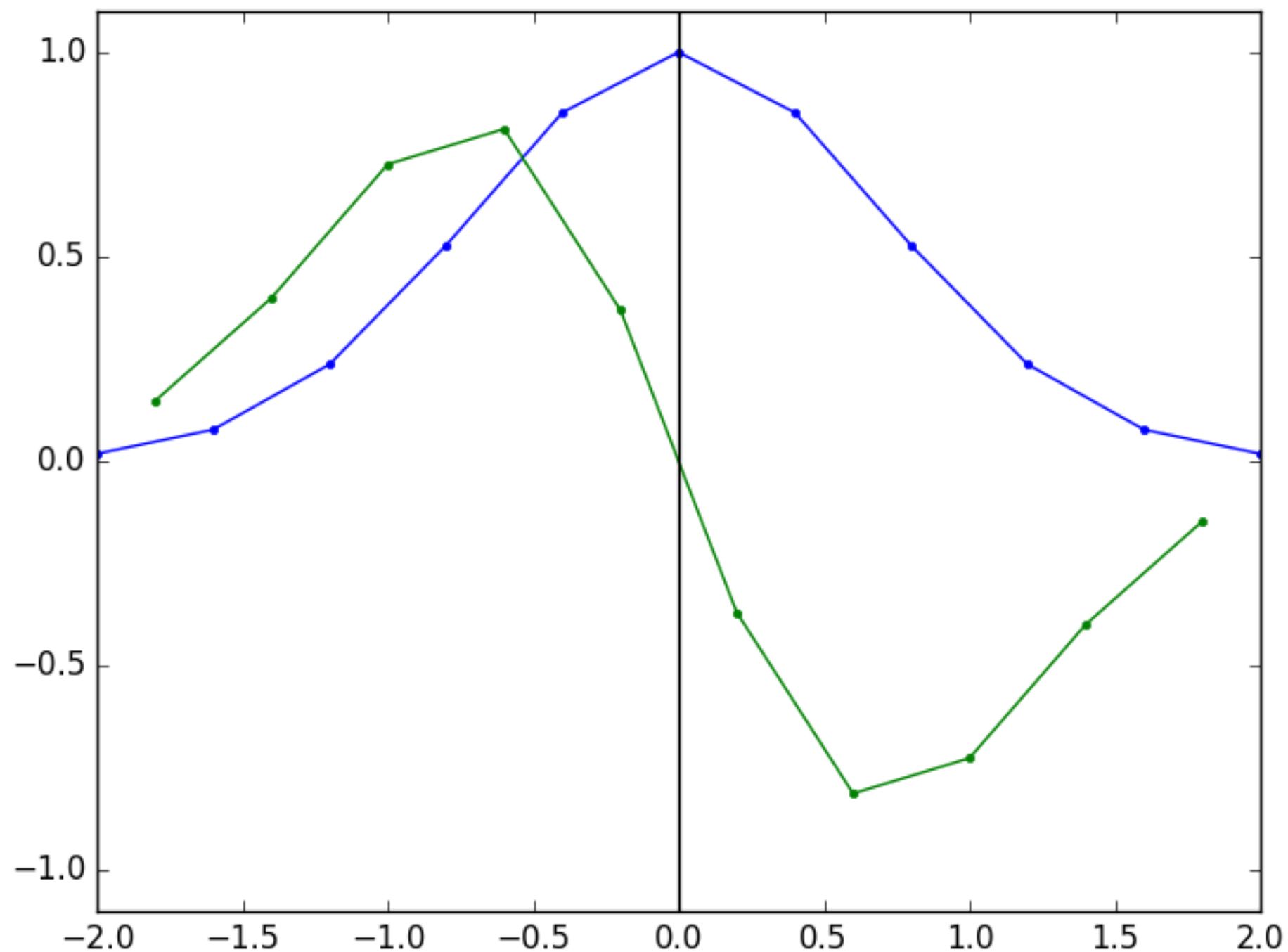


metade do  
espaçamento



# Mas em que ponto está sendo calculada a derivada numérica?

Se for no ponto médio



# Exercício

- Calcule a derivada numérica da função  $f(x) = \sin(x)$
- Compare com a derivada analítica:  $f'(x) = \cos(x)$
- Faça o teste com três valores diferentes para  $\Delta x$
- Envie o script python no e-disciplinas.