



Ciências  
ULisboa

# Modelação Numérica

## Aula 2

Séries temporais e análise de Fourier

# Modelação Numérica 2023

Mon	Tue	Wed	Thu	Fri	Mon	Tue	Wed	Thu	Fri	Mon	Tue	Wed	Thu	Fri
13-Feb	14-Feb	15-Feb	16-Feb	17-Feb	20-Feb	21-Feb	22-Feb	23-Feb	24-Feb	27-Feb	28-Feb	01-Mar	02-Mar	03-Mar
		T1									T2	T3	P1-1	P1-1
06-Mar	07-Mar	08-Mar	09-Mar	10-Mar	13-Mar	14-Mar	15-Mar	16-Mar	17-Mar	20-Mar	21-Mar	22-Mar	23-Mar	24-Mar
	T4	T5	P1-2	P1-2		T6	T7	P1-3	P1-3		T8	T9	P1-4	P1-4
27-Mar	28-Mar	29-Mar	30-Mar	31-Mar	03-Apr	04-Apr	05-Apr	06-Apr	07-Apr	10-Apr	11-Apr	12-Apr	13-Apr	14-Apr
	T10	T11	P1-5	P1-5		T12	T13			E1			Proj1	
17-Apr	18-Apr	19-Apr	20-Apr	21-Apr	24-Apr	25-Apr	26-Apr	27-Apr	28-Apr	01-May	02-May	03-May	04-May	05-May
	T14	T15	P2-1	P2-1			T16	P2-2	P2-2		T17	T18	P2-3	P2-3
08-May	09-May	10-May	11-May	12-May	15-May	16-May	17-May	18-May	19-May	22-May	23-May	24-May	25-May	26-May
	T19	T20	P2-4	P2-4		T21	T22	P2-5	P2-5	E2			Proj2	

Projetos devem ser entregues (E1,E2) na semana da defesa.

Início dos Projetos: P1-1 e P2-1

Apresentação: Proj1, Proj2

Ficheiros a entregar: MN2023\_Px\_PLxx\_Gx.py, \*.pptx (num zip)

# Funções (contínuas) de uma variável independente

$$V = V(t)$$

$t$  é o tempo, mas pode ser outra variável ( $x \dots$ )

**Discretização: Amostra regular** com  $N$  pontos

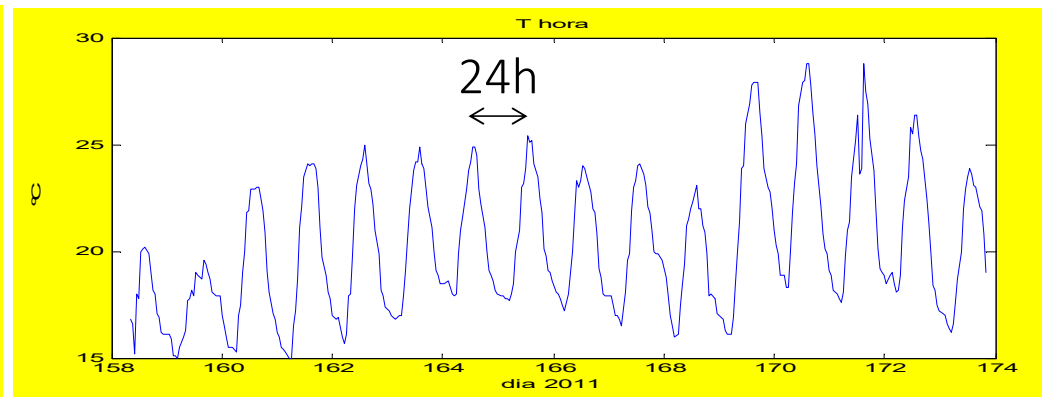
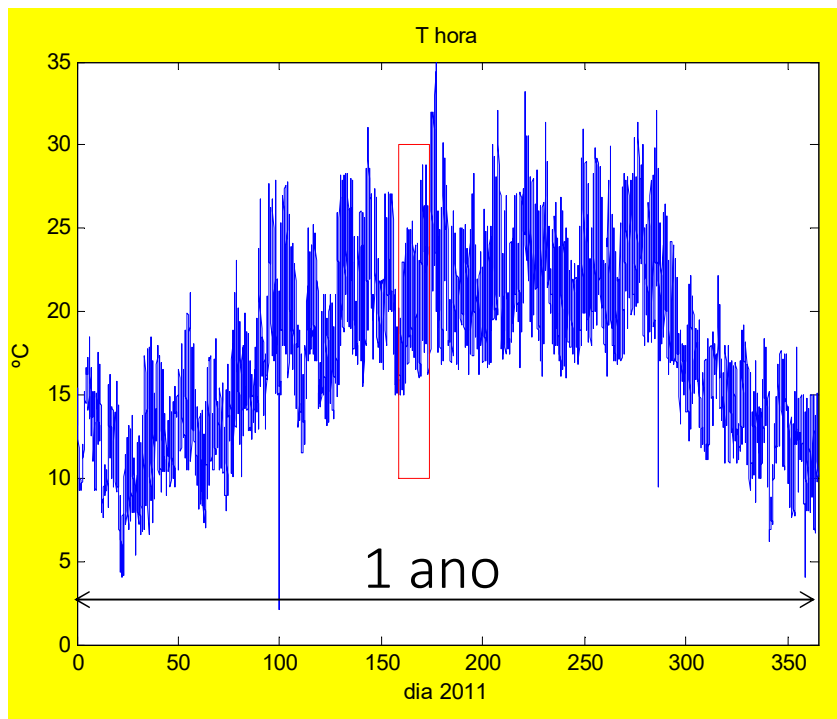
$$V_n = V(t_0 + n\Delta t), n = 0, \dots, N - 1$$

$t_0$  – fase inicial (amostra 0)

$\Delta t$  – intervalo de amostragem (step)

$V_n$  – número float (truncado a 64 bit)

# Exemplo: observações da Temperatura em Lisboa



Observações horárias são suficientes para descrever os “**ciclos**” diurno e annual. Não dizem nada sobre flutuações muito rápidas (sub-horárias).

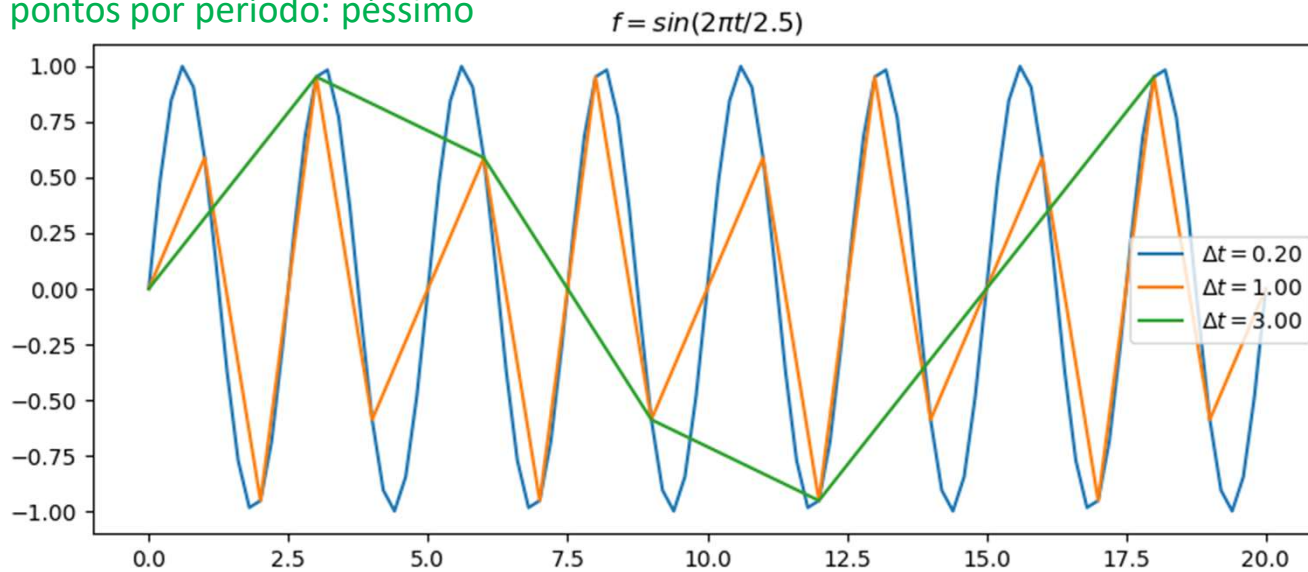
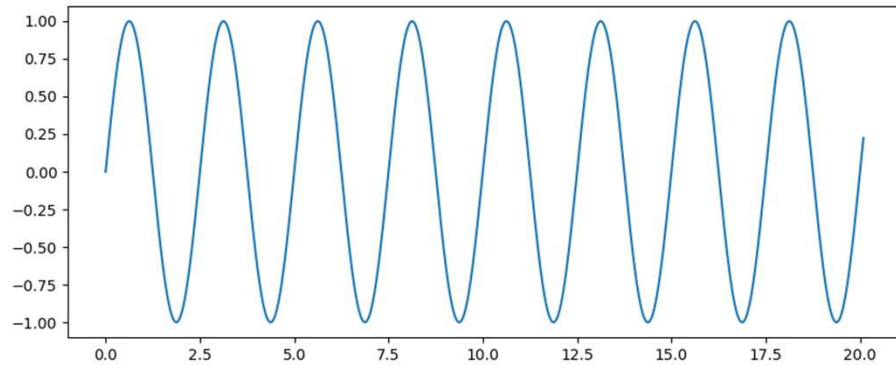
# Amostras de Sinusóides

$$t_0 = 0$$

12.5 pontos por período: razoável

2.5 pontos por período: mau

0.8 pontos por período: péssimo



# Teorema da amostragem

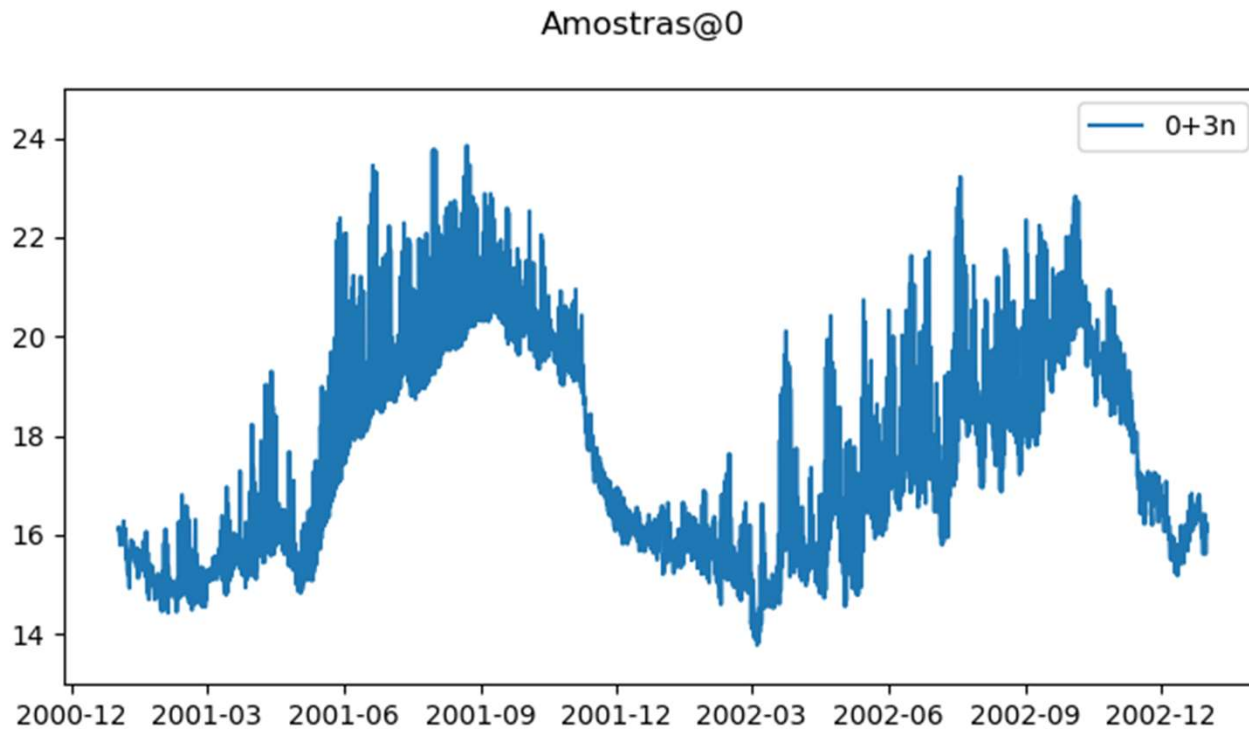
O exemplo anterior mostra que a operação de amostragem implica sempre um **erro**, mas em certos casos altera completamente a função.

No caso de uma função sinusoidal é fácil perceber que tudo depende do **Número de amostras por período**:

Com **menos de 2 amostras por período a função é falseada (*aliasing*)** sendo vista como um seno com um período muito mais longo : uma amostra só pode representar períodos  **$T \geq 2\Delta t$**

Com pouco mais de 2 amostras por período a série pode apresentar **artefactos**. A **fase** também é importante.

Várias formas de amostrar uma série: **start**  
( **$n_0$** , fase), **step** ( **$\Delta t$** )



E se a função não for uma senoide?

O **teorema de Fourier** garante que qualquer **função periódica** pode ser obtida pela soma de sinusoides:

$$f(t) = \frac{a_0}{2} + \sum_{k=1}^{\infty} \left( a_k \cos \frac{2\pi kt}{T} + b_k \sin \frac{2\pi kt}{T} \right)$$

Onde  $(a_k, b_k)$  são as **amplitudes** associadas à **harmónica**  $k$

Em geral, precisamos de **infinitas** harmónicas!



# Série de Fourier na forma complexa

Utilizando a formula de Euler ( $e^{ix} = \cos x + i \sin x$ ) pode mostrar-se que:

$$f(t) = \frac{a_0}{2} + \sum_{k=1}^{\infty} \frac{a_k - ib_k}{2} e^{\frac{i2\pi kt}{T}} + \sum_{k=1}^{\infty} \frac{a_k + ib_k}{2} e^{-\frac{i2\pi kt}{T}}$$

Ou

$$f(t) = \sum_{k=-\infty}^{\infty} c_k e^{i2\pi kt/T}$$

Com os coeficientes (complexos):

$$c_k = \frac{1}{2\pi} \int_{-T/2}^{T/2} f(t) e^{-i2\pi k/T} dt, k = 0, \pm 1, \pm 2, \dots$$

# Transformada discreta de Fourier (1)

No caso geral o teorema de Fourier não é **computável**, pois requer infinitos coeficientes, e os coeficientes são calculados por meio de um integral.

Se a função for representada exatamente com um número finito de harmónicas (**função de banda limitada**), se o intervalo de amostragem satisfizer o **teorema da amostragem**  $\left(\Delta t < \frac{T_{Min}}{2} \Leftrightarrow \Delta t < \frac{1}{2f_{Max}}\right)$  e se a **amostra for suficientemente longa** para conter um período fundamental (o mais longo), a série de Fourier é **computável** e **exata**.

# Transformada discreta de Fourier (2)

**Transformada discreta de Fourier** ( $k$  indica a frequência)

$$F_k(k) = \sum_{n=0}^{N-1} f_n e^{-2\pi i n k / N}$$

**transformada discreta inversa de Fourier** ( $n$  indica )

$$f_n(n) = \frac{1}{N} \sum_{k=0}^{N-1} F_k e^{2\pi i n k / N}$$

# Código naïve (lento)

```
import numpy as np;import matplotlib.pyplot as plt
def dFT(y): #transformada discreta de Fourier direta
    i=complex(0,1.);pi=np.pi
    N=len(y)
    z=np.zeros(y.shape,dtype=complex)
    for k in range(N):
        for j in range(N):
            z[k]=z[k]+y[j]*np.exp(-2*pi*i*j*k/N)
    return z
def iFT(y): #transformada discreta de Fourier inversa
    i=complex(0,1.);pi=np.pi
    N=len(y)
    z=np.zeros(y.shape,dtype=complex)
    for k in range(N):
        for j in range(N):
            z[k]=z[k]+y[j]*np.exp(2*pi*i*j*k/N)
    return z/N
```

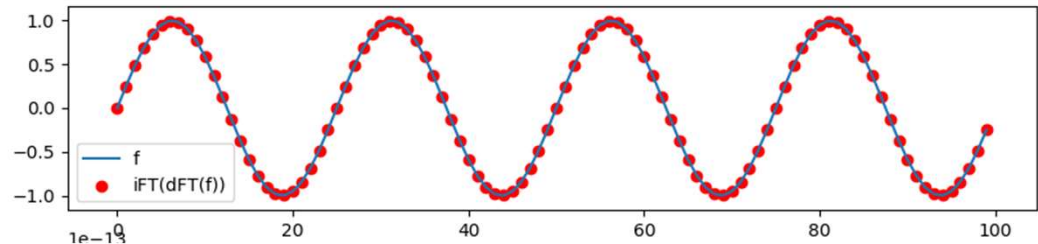
$$F_k = \sum_{n=0}^{N-1} f_n e^{-2\pi i n k / N}$$

$$f_n = \frac{1}{N} \sum_{k=0}^{N-1} F_k e^{2\pi i n k / N}$$

```

N=100;dt=1.;T=N*dt/4.;
t=np.linspace(0,dt*(N-1),N);
f=np.sin(2*np.pi*t/T)
plt.subplot(3,1,1);plt.plot(t,f,label='f')
F=dFT(f);
fNyq=1/(2*dt) #frequência de Nyquist
df=2*fNyq/(N-1) #resolução espectral
freq=np.zeros(t.shape)
freq[0:N//2+1]=np.arange(0,fNyq+df,df)
if N%2==0:
    freq[N//2+1:N]=np.arange(-fNyq+df,0,df)
else:
    freq[N//2+1:N]=np.arange(-fNyq,0,df)
left=range(N//2+1,N)
right=range(0,N//2+1)

```



```
plt.subplot(3,1,2);
plt.plot(freq[left],np.real(F[left]))
plt.plot(freq[right],np.real(F[right]))
plt.ylabel('Real(F)');plt.xlabel('f (Hz)')
plt.subplot(3,1,3);
plt.plot(freq[left],np.imag(F[left]))
plt.plot(freq[right],np.imag(F[right]));
plt.ylabel('Imag(F)'); plt.xlabel('f (Hz)')
ff=iFT(F)
plt.subplot(3,1,1);
plt.scatter(t,np.real(ff),color='red',\
            label='iFT(dFT(f))')
plt.legend()
```