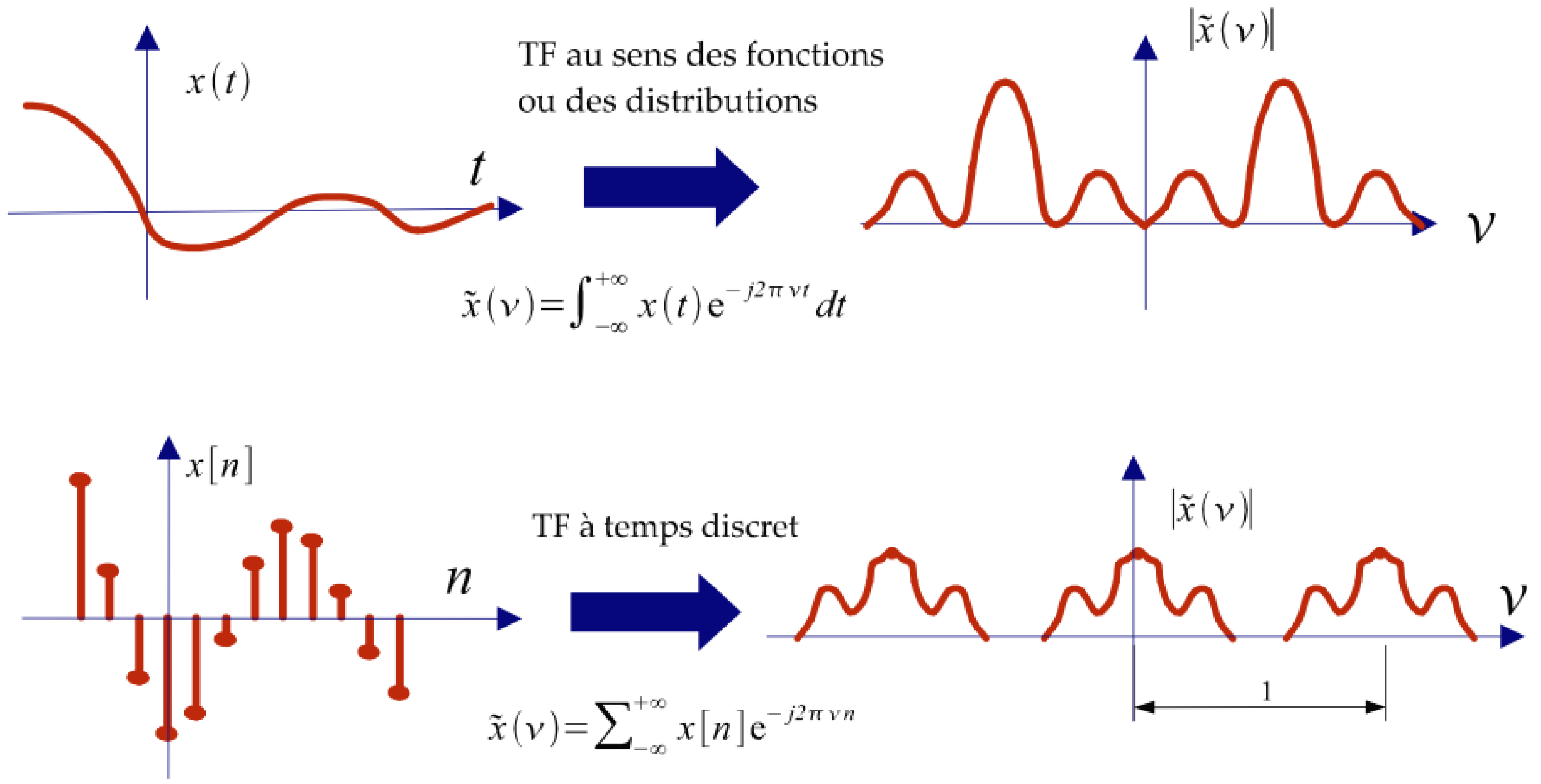
INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech
LEnsE Laboratoire d'Enseignement Expérimental

ONIP-1

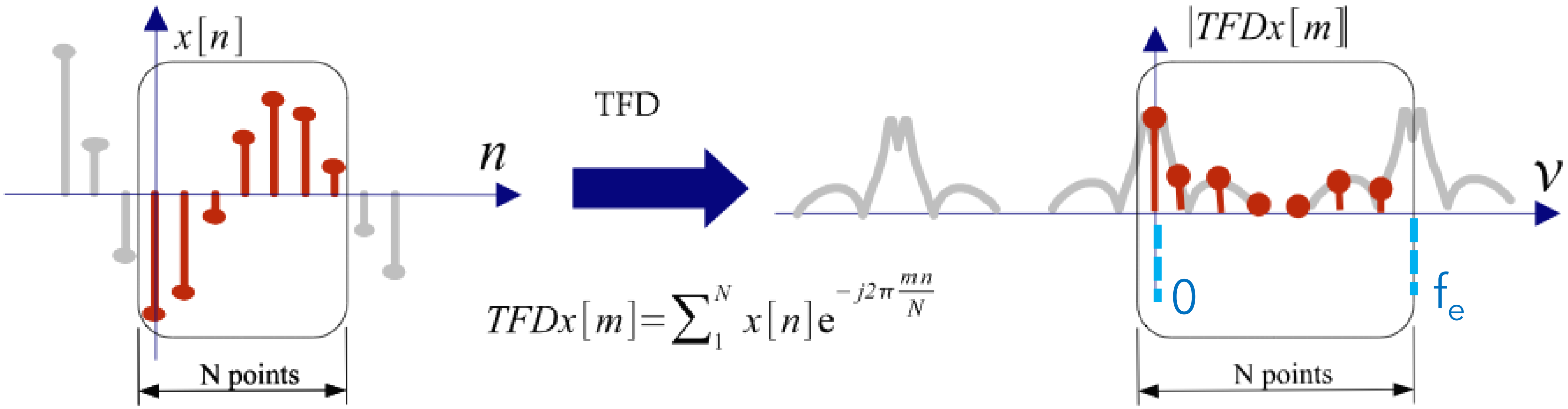
FFT et structure d'un script

Outils Numériques / Semestre 5
Institut d'Optique / B3_1

Rappel sur la Transformée de Fourier



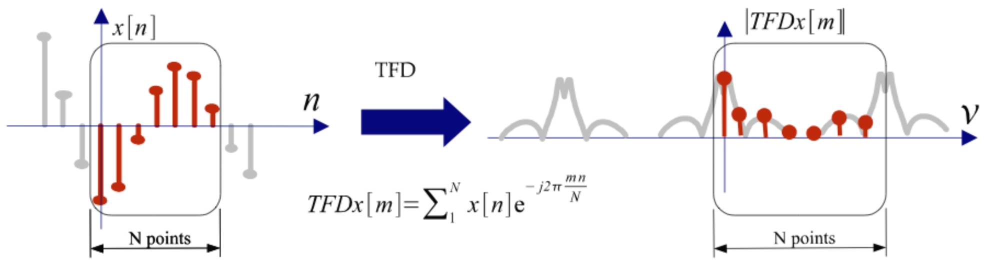
Rappel sur la FFT



numpy .fft.fft .fft.fftshift

$$A_k = \sum_{m=0}^{n-1} a_m \exp\left\{-2\pi i \frac{mk}{n}\right\} \qquad k = 0, \ldots, n-1.$$

Rappel sur la FFT



numpy .fft.fft .fft.fftshift

$$A_k = \sum_{m=0}^{n-1} a_m \exp\left\{-2\pi i \frac{mk}{n}\right\} \qquad k = 0, \ldots, n-1.$$

If **A = fft(a, n)**, then **A[0]** contains the zero-frequency term
Then **A[1:n/2]** contains the positive-frequency terms,
and **A[n/2+1:]** contains the negative-frequency terms
For an **even number** of input points, **A[n/2]** represents
both positive and negative Nyquist frequency,
For an **odd number** of input points, **A[(n-1)/2]** contains the largest positive frequency,
while **A[(n+1)/2]** contains the largest negative frequency.

Utilisation des fonctions

```
def sinus(t, A, f):
	return A*np.sin(2*np.pi*f*t)

time_vect = np.linspace(0, 1, 1001)
```

- **Mémoire préservée**

```
TF = np.fft.fft(sinus(time_vect, 1, 10))
plt.figure()
plt.plot(time_vect, sinus(time_vect, 1, 10))
```

- **Temps de calcul optimal**

```
sig = sinus(time_vect, 1, 10)
TF = np.fft.fft(sig)
plt.figure()
plt.plot(time_vect, sig)
```

Fonctions / Paramètres optionnels

```
def sinus(t, A=1, f=100):
	return A*np.sin(2*np.pi*f*t)

time_vect = np.linspace(0, 1, 101)
```

```
A1 = sinus(time_vect)
A2 = sinus(time_vect, A=10)
A3 = sinus(time_vect, A=10, f=200)
```

Fonctions / Paramètres optionnels

```
def sinus(t, A=1, f=100):
	return A*np.sin(2*np.pi*f*t)

time_vect = np.linspace(0, 1, 101)
```

```
A1 = sinus(time_vect)
A2 = sinus(time_vect, A=10)
A3 = sinus(time_vect, A=10, f=200)
```

⚠ ***PHYSIQUE***

Te = 1/101 s ≈ 10ms

f = 100 Hz
T ≈ 10ms

Critère de Shanonn-Nyquist non respecté

Fonctions / Paramètres optionnels

```
def sinus(t, A=1, f=100):
	if(isinstance(t, np.ndarray)):
		Te = t[0] - t[1]
		if(1/Te < 2*f):
			print('Shannon sampling frequency warning !!')
	return A*np.sin(2*np.pi*f*t)

time_vect = np.linspace(0, 1, 101)
```

```
A1 = sinus(time_vect)
A2 = sinus(time_vect, A=10)
A3 = sinus(time_vect, A=10, f=200)
```

PHYSIQUE

Te = 1/101 s ≈ 10ms

```
Shannon sampling frequency warning !!
```

INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech

LEnsE Laboratoire d'Enseignement Expérimental