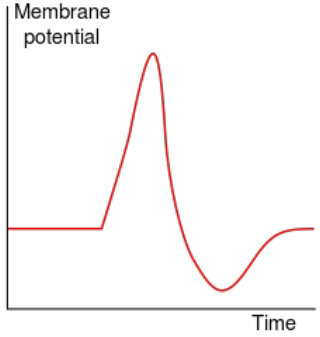
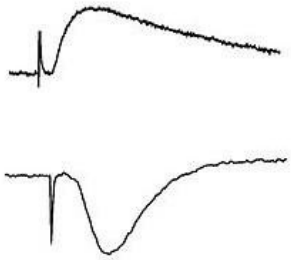
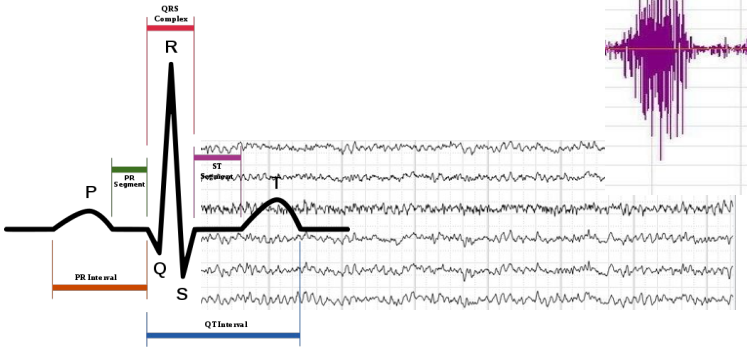
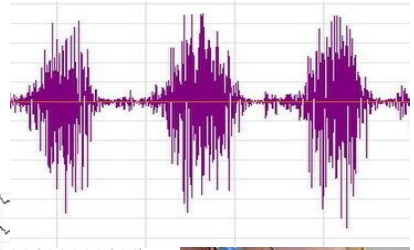
Procesando señales electrofisiológicas con Python

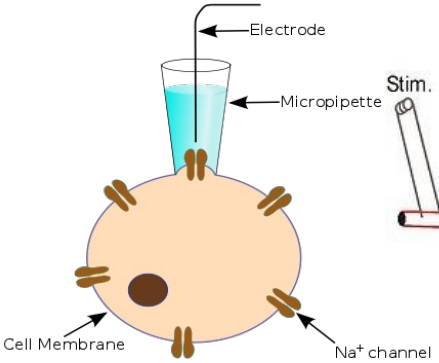
Alejandro Weinstein (@ajweinstein), Escuela de Ingeniería Biomédica de la Universidad de Valparaíso
Oscar Valencia, Escuela de Kinesiología de la Universidad de los Andes

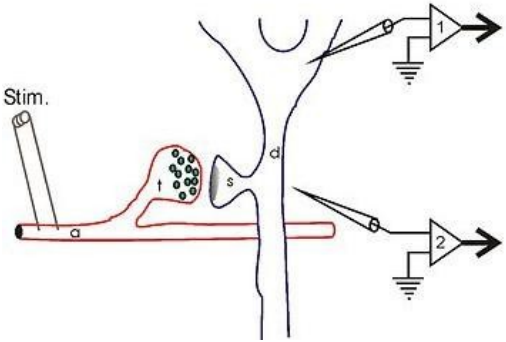


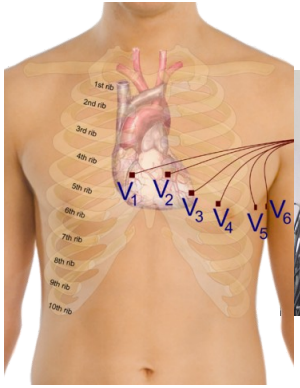


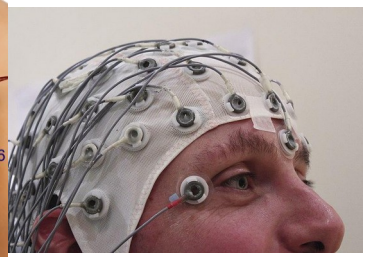


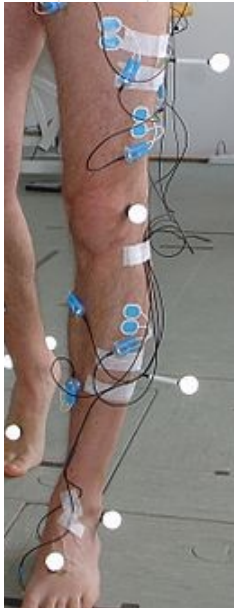




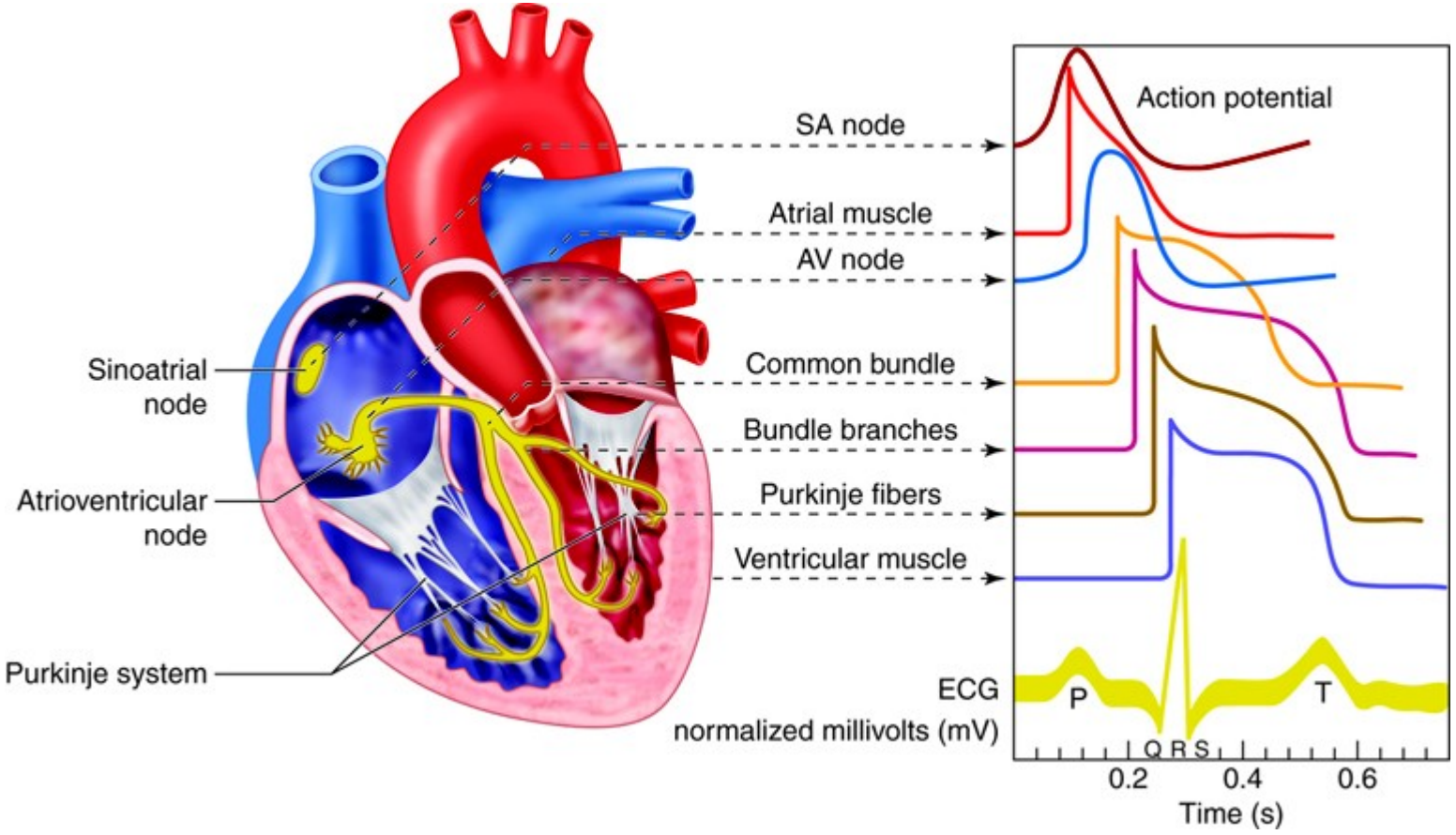




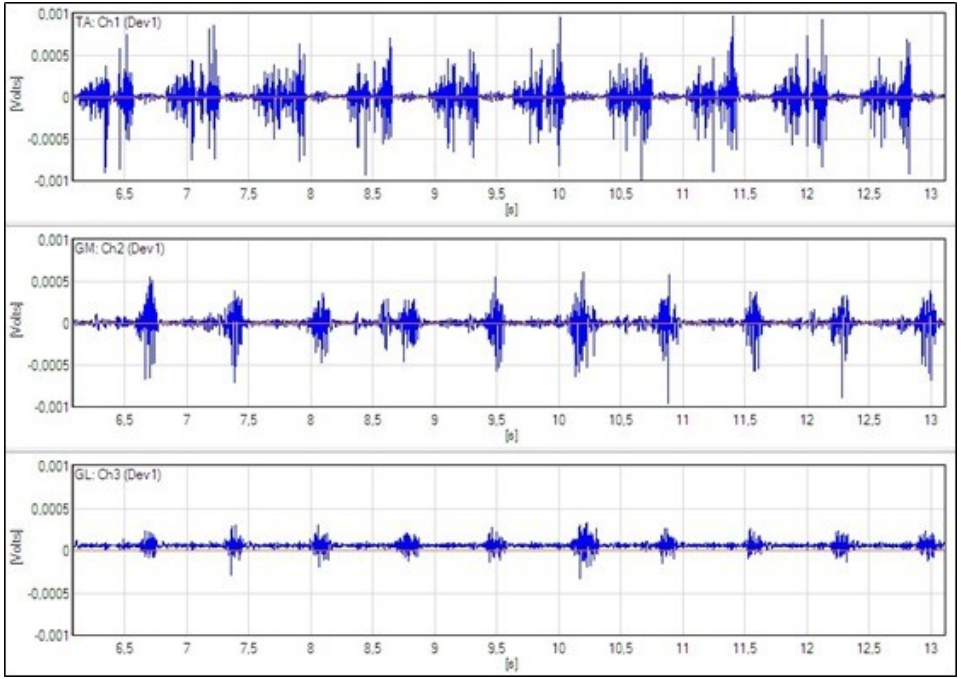




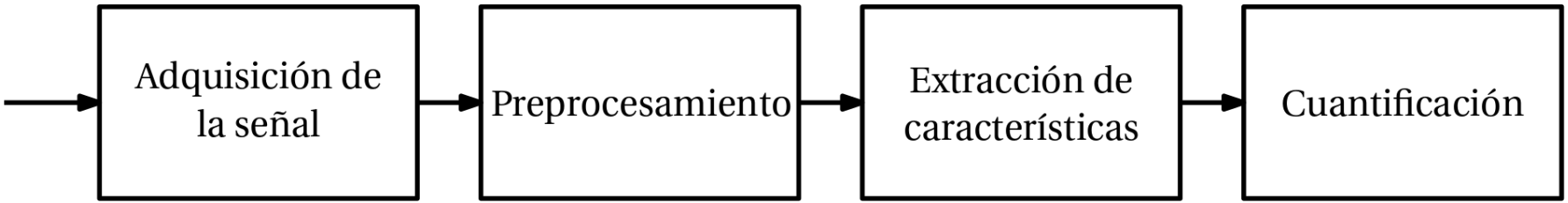
Escala



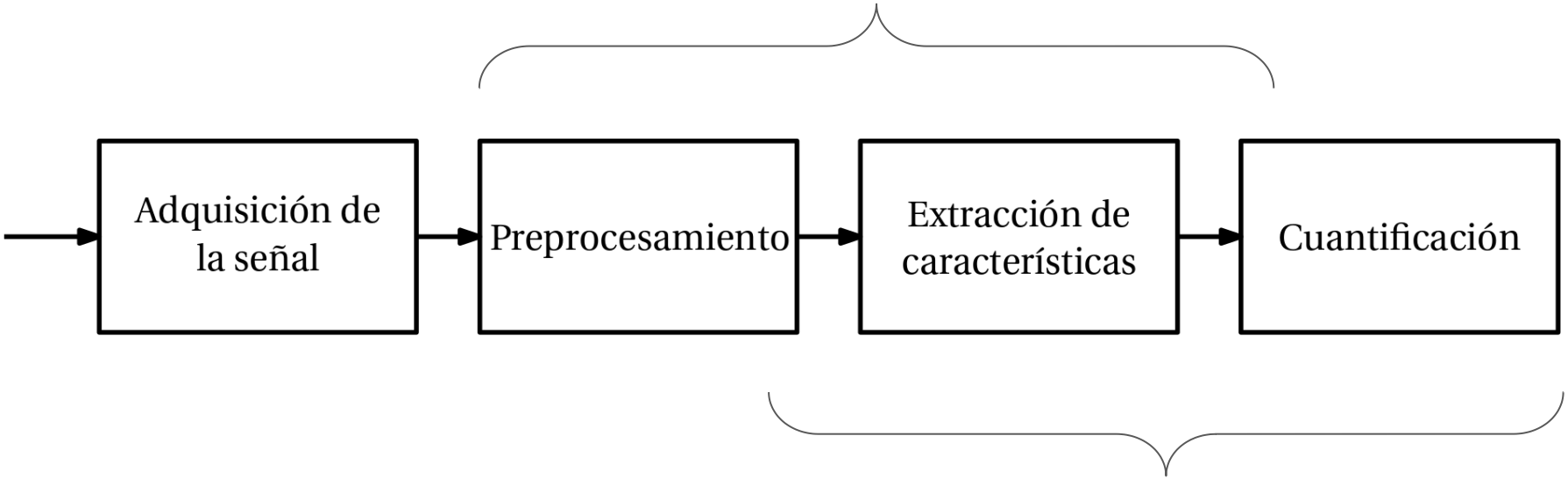
https://commons.wikimedia.org/wiki/File:Anatomy_of_the_Heart.jpg



Valencia, Oscar, et al. "Análisis comparativo de la actividad electromiográfica en miembro inferior entre técnicas de antepié y retropié en corredores amateur." *Retos* 38.38 (2020): 271-275.


Adquisición de la señal
Preprocesamiento
Extracción de características
Cuantificación

Stack de computación científica en Python:
Numpy + SciPy + Matplotlib

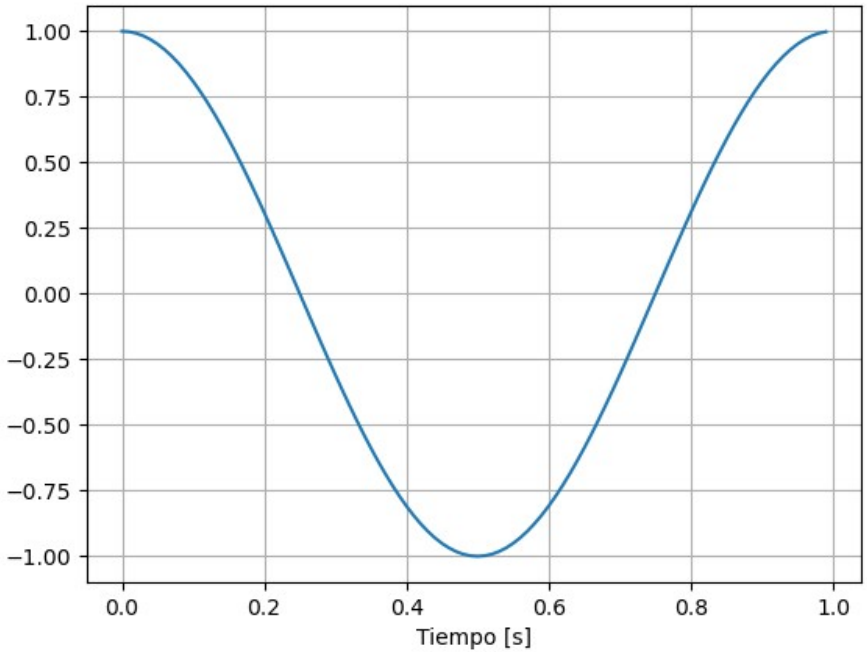


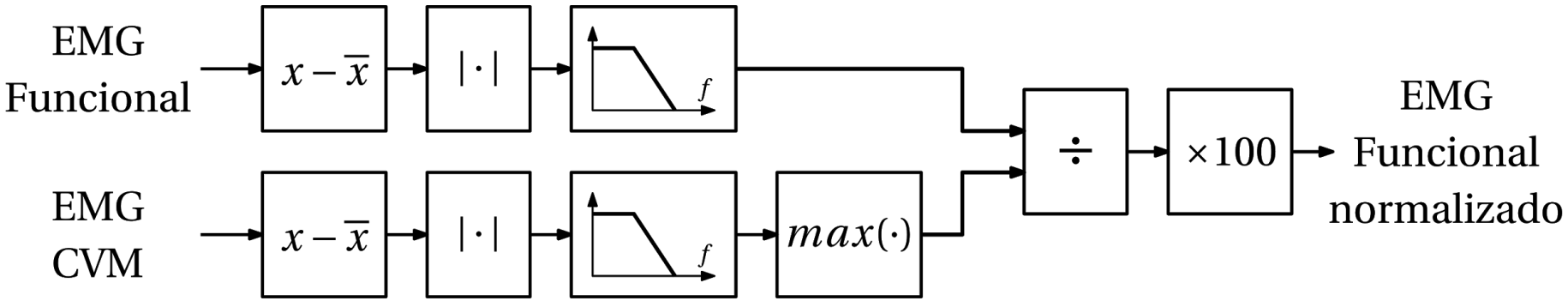
Bibliotecas ad hoc para el tipo de señal:
MNE-Python -> EEG ; pyHRV -> ECG

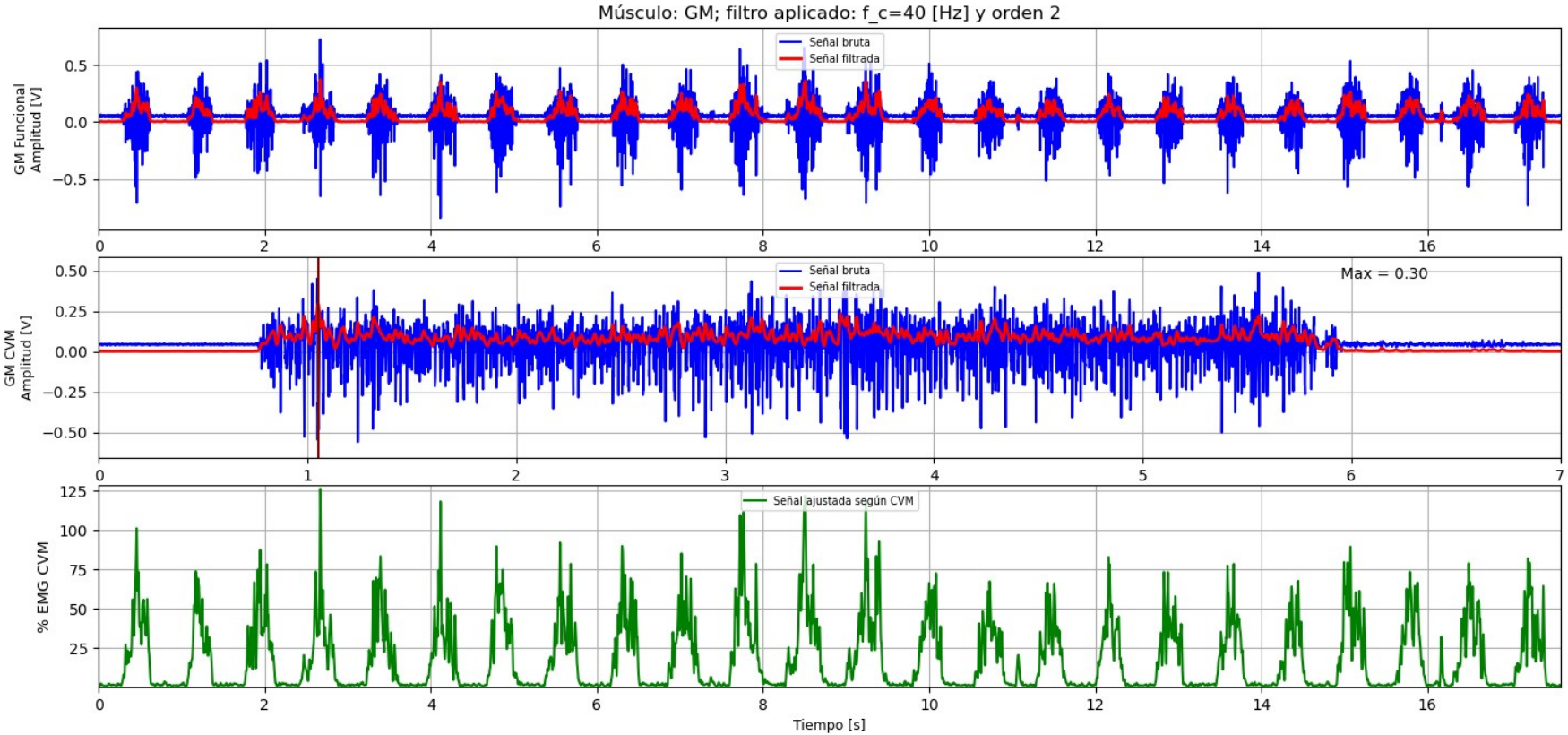
```
In [16]: import numpy as np

In [17]: a = np.array([1, 2, 3])

In [18]: a[1] = 20

In [19]: a * 2
Out[19]: array([ 2, 40,  6])

In [20]: b = np.array([4, 5, 6])

In [21]: a + b
Out[21]: array([ 5, 25,  9])
```

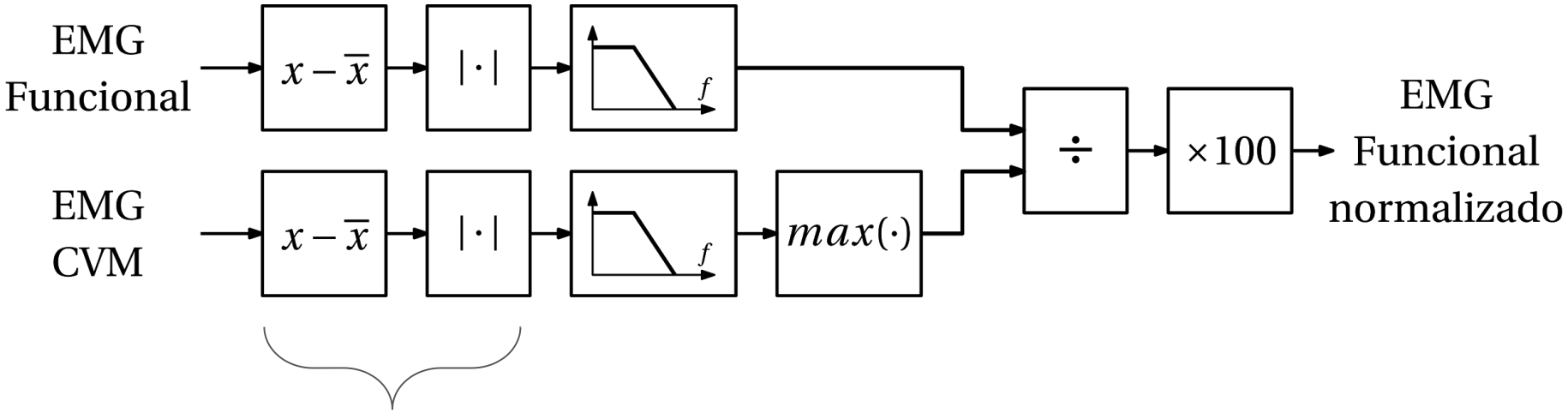
```
In [22]: c = np.array([7, 8])
In [23]: a + c
Traceback (most recent call last):
  File "<ipython-input-23-e81e582b6fa9>",
line 1, in <module>
    a + c
ValueError: operands could not be
broadcast together with shapes (3,) (2,)
```

```
import numpy as np

import matplotlib.pyplot as plt

t = np.arange(0, 1, 0.01)

y = np.cos(2 * np.pi * t)

plt.plot(t, y)

plt.xlabel('Tiempo [s]')

plt.grid()
```

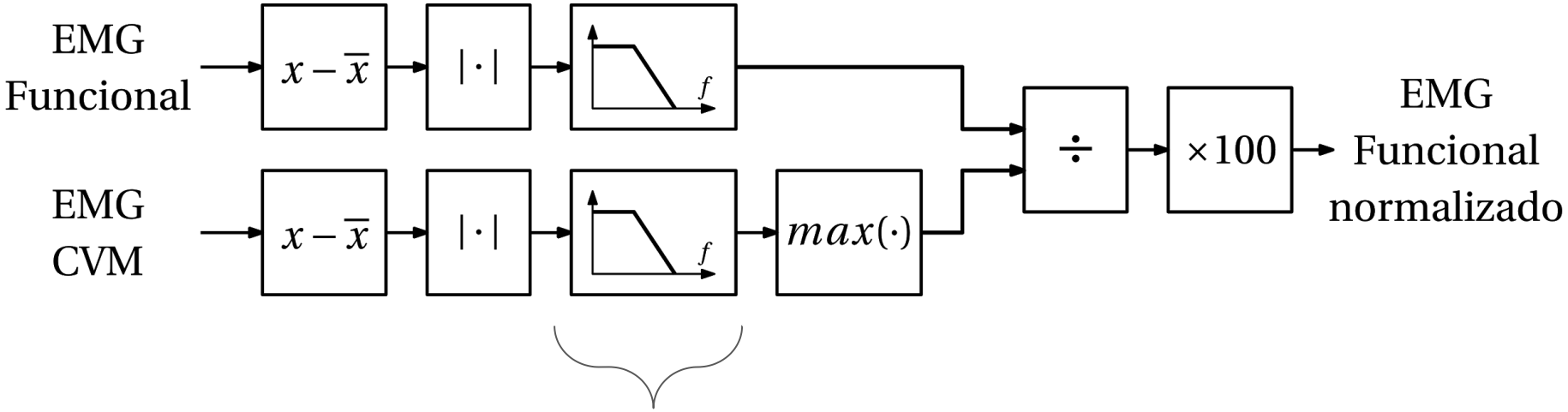



EMG
Funcional
EMG
CVM
$x - \overline{x}$
$|\cdot|$
f
$max(\cdot)$
÷
×100
EMG
Funcional
normalizado

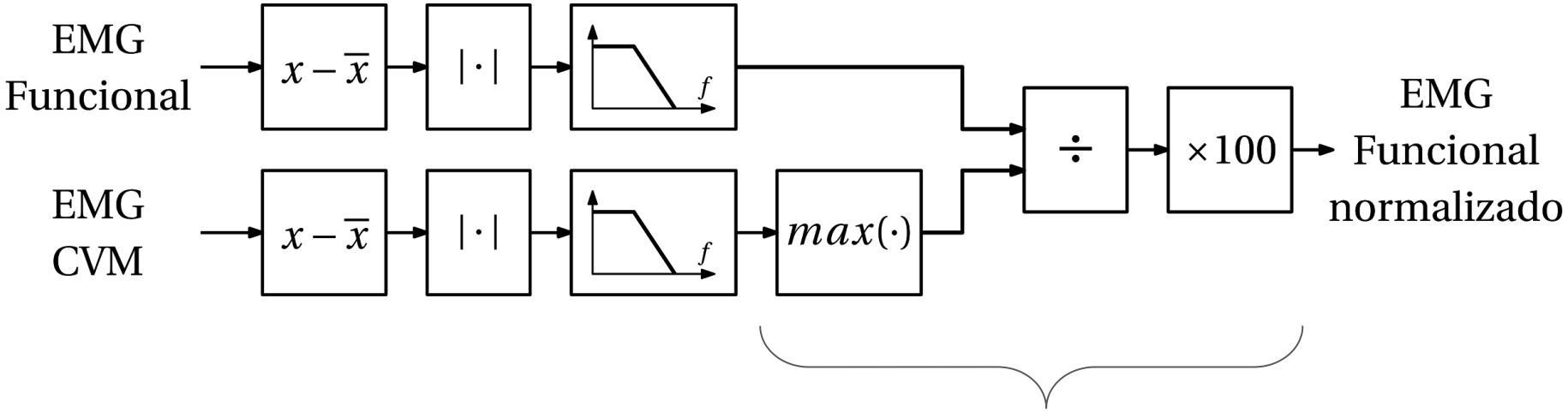

Músculo: GM; filtro aplicado: f_c=40 [Hz] y orden 2
Señal bruta
Señal filtrada
GM Funcional
Amplitud [V]
Señal bruta
Señal filtrada
Max = 0.30
GM CVM
Amplitud [V]
Señal ajustada según CVM
% EMG CVM
Tiempo [s]




```
emg_fun_env = abs(emg_fun - np.mean(emg_fun))

emg_cvm_env = abs(emg_cvm - np.mean(emg_cvm))
```




```
b, a = butter(int(filt_ord), (int(fc)/(fs/2)), btype = 'low')

emg_fun_env_f = filtfilt(b, a, emg_fun_env)

emg_cvm_env_f = filtfilt(b, a, emg_cvm_env)
```




```
emg_cvm_I = np.max(emg_cvm_env_f)

emg_fun_norm = (emg_fun_env_f / emg_cvm_I) * 100
```

Ejemplo completo en:

https://github.com/aweinstein/emg_cvm_normalization