Un filtre passe-bande conserve uniquement les fréquences comprises entre deux valeurs :une fréquence de coupure basse f_low et une fréquence de coupure haute f_high. Toutes les fréquences en dehors de cette plage sont supprimées.

```
import numpy as np
import matplotlib.pyplot as plt

# 1. Paramètres du signal
fs = 1000  # Fréquence d'échantillonnage (Hz)
T = 1      # Durée du signal (s)
t = np.linspace(0, T, fs, endpoint=False)  # Axe temporel

# 2. Génération du signal périodique
freqs = [50, 150, 300]  # Fréquences en Hz
amplitudes = [1, 0.5, 0.2]
signal = sum(A * np.sin(2 * np.pi * f * t) for A, f in
zip(amplitudes, freqs))

# 3. Calcul du spectre du signal (FFT)
fft_signal = np.fft.fft(signal)
freqs_fft = np.fft.fftfreq(len(signal), 1/fs)

# 4. Définition des filtres
def filtre_passe_bande(signal_fft, freqs, f_low, f_high):
    """Conserve uniquement les fréquences entre f_low et f_high"""
    filtré = np.zeros_like(signal_fft)  # Initialise avec des zéros
    masque = (np.abs(freqs) >= f_low) & (np.abs(freqs) <= f_high)  #
Conserve les fréquences dans la bande
    filtré[masque] = signal_fft[masque]  # Applique le filtre
    return filtré

# 5. Application du filtre passe-bande
f_low = 100   # Fréquence de coupure basse (Hz)
f_high = 250  # Fréquence de coupure haute (Hz)
fft_passe_bande = filtre_passe_bande(fft_signal, freqs_fft, f_low,
f_high)

# 6. Reconstruction du signal filtré
signal_passe_bande = np.fft.ifft(fft_passe_bande).real

# 7. Affichage des résultats
fig, axs = plt.subplots(3, 2, figsize=(12, 8))

# Signal initial
axs[0, 0].plot(t, signal)
axs[0, 0].set_title("Signal initial")
axs[0, 1].stem(freqs_fft[:fs//2], np.abs(fft_signal[:fs//2]),
use_line_collection=True)
axs[0, 1].set_title("Spectre du signal initial")

# Signal passe-bande
axs[1, 0].plot(t, signal_passe_bande, 'm')
axs[1, 0].set_title(f"Signal après filtre passe-bande ({f_low}-
{f_high} Hz)")
axs[1, 1].stem(freqs_fft[:fs//2], np.abs(fft_passe_bande[:fs//2]),
use_line_collection=True, linefmt='m', markerfmt='mo')
axs[1, 1].set_title(f"Spectre après filtre passe-bande ({f_low}-
{f_high} Hz)")
```

```
for ax in axs.flat:
    ax.set_xlabel("Temps (s)" if ax in axs[:, 0] else "Fréquence
(Hz)")

plt.tight_layout()
plt.show()
```