ĐẠI HỌC QUỐC GIA TP HỒ CHÍ MINH

ĐẠI HỌC BÁCH KHOA

BÀI TẬP LỚN

XỬ LÝ SỐ TÍN HIỆU

CHỦ ĐỀ: THIẾT KẾ BỘ LỌC SỐ

GVHD: PGS. TS Hà Hoàng Kha

Lớp L02 – Nhóm 19 – HK241

Sinh viên thực hiện	Mã số sinh viên	Đóng góp
Trần Hoàng Kiên	2211736	100%
Nguyễn Đức Phát	2212518	100%
Phùng Hoàng Hữu Nhân	2212378	100%

Thành phố Hồ Chí Minh – 2024

I. Giới thiệu đề tài

Bộ lọc số là một thành phần cơ bản trong xử lý tín hiệu số (Digital Signal Processing - DSP), được sử dụng để chỉnh sửa hoặc loại bỏ các thành phần không mong muốn trong tín hiệu số hoặc để cải thiện các đặc tính mong muốn. Các bộ lọc số là hệ thống dùng làm biến dạng sự phân bố tần số của các thành phần của một tín hiệu theo các chỉ tiêu đã cho. Bộ lọc số hoạt động trên tín hiệu rời rạc bằng cách áp dụng các phép toán học thông qua thuật toán. Bộ lọc số đóng vai trò quan trọng trong nhiều lĩnh vực, từ viễn thông, xử lý âm thanh, xử lý ảnh, đến y sinh và tự động hóa. Đề tài thiết kế bộ lọc số không chỉ cung cấp kiến thức lý thuyết mà còn trang bị cho sinh viên, kỹ sư những kỹ năng thực tiễn.

Lọc là một trong những hoạt động xử lý tín hiệu quan trọng. Một bộ lọc tương tự hoạt động trên các tín hiệu liên tục và thường được thực hiện với các linh kiện như khuếch đại thuật toán, các điện trở và các tụ điện. Một bộ lọc số hoạt động trên tín hiệu thời gian rời rạc và có thể thực hiện với một bộ xử lý số tín hiệu như họ TMS320C6x. Quá trình lọc bao gồm sử dụng một bộ biến đổi A/D để nhận tín hiệu vào, xử lý các mẫu vào rồi gửi kết quả ra thông qua một bộ biến đổi D/A. Các bộ lọc số có rất nhiều ưu điểm so với các bộ lọc tương tự. Các ưu điểm này bao gồm độ tin cậy cao hơn, độ chính xác cao hơn và ít nhạy với nhiệt độ và tuổi đời. Các đặc tính lọc như tần số trung tâm, băng thông và loại bộ lọc có thể thay đổi dễ dàng.

Dưới đây là cách thiết kế một bộ lọc số sử dụng MATLAB.

II. Lý thuyết

1. Khái niệm bộ lọc số

Bộ lọc số là một hệ thống dùng để làm biến dạng sự phân bố tần số của các thành phần của một tín hiệu theo các chỉ tiêu đã cho. Các mạch lọc số cho tín hiệu số có phổ nằm trong 1 dải tần số nhất định đi qua và không cho tín hiệu có phổ nằm ngoài dải tần số đó đi qua. Các thao tác của xử lý dùng để làm biến dạng sự phân bố tần số của các thành phần của một tín hiệu theo các chỉ tiêu đã cho nhờ một hệ thống số được gọi là việc lọc số.

Bộ lọc số thường được mô tả bằng phương trình vi sai rời rạc hoặc hàm truyền H(z), biểu diễn mối quan hệ giữa đầu vào x[n] và đầu ra y[n] của tín hiệu:

$$y[n] = h[n] * x[n] = \sum_{k=0}^{N} h(k)x(n-k)$$

- Thiết kế bộ lọc: là xây dựng hàm truyền thỏa đáp ứng tần số cho trước.
- Thiết kế bộ lọc với đáp ứng xung hữu hạn FIR: đầu ra là vector đáp ứng xung h = [h0 , h1 , h2 , …. ,hN]
- Thiết kế bộ lọc với đáp ứng xung vô hạn IIR: đầu ra là các vector hệ số tử số và mẫu số của hàm truyền b = [b0 , b1 , …, bN] và a = [1, a1 , a2 ,…, aN]

❖ **Bộ lọc FIR:**

Ưu điểm:

- Đặc tuyến pha tuyến tính.
- Độ ổn định cao(do không có các cực).

Nhược điểm:

- Để có đáp ứng tần số tốt cần chiều dài bộ lọc N lớn → Gia tăng chi phí tính toán.

❖ **Bộ lọc IIR:**

Ưu điểm:

- Chi phí tính toán thấp.

- Thực hiện hiệu quả theo kiểu cascade các mạch bậc 2 (Second-order sections) .

Khuyết điểm:

- Có sự bất ổn định do quá trình lượng tử hóa các hệ số có thể đẩy các cực ra ngoài vòng tròn đơn vị.
- Không thể đạt pha tuyến tính trên toàn khoảng Nyquist.

2. Phân loại bộ lọc lý tưởng dựa trên đáp ứng tần số

Dựa vào đáp ứng tần số, có thể chia bộ lọc ra làm các loại sau:

- Bộ lọc thông thấp LPF (Low Pass Filter)
- Bộ lọc thông cao HPF (High Pass Filter)
- Bộ lọc thông dải BPF (Band Pass Filter)
- Bộ lọc chắn dải BSF (Band Stop Filter)

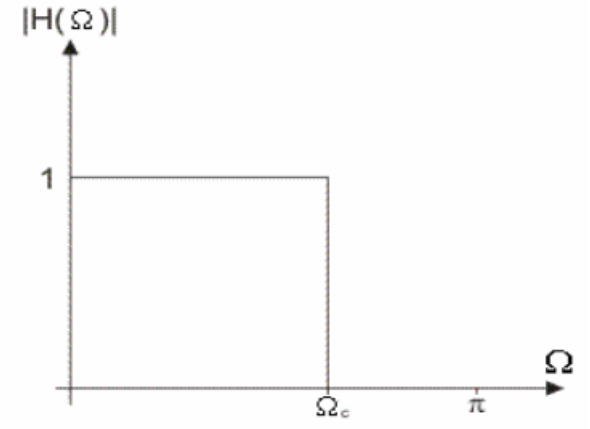
❖ **Bộ lọc lý tưởng:**

Bộ lọc thông thấp lý tưởng: Trong thiết kế bộ lọc FIR thông thấp, mục tiêu là tạo ra một bộ lọc có đáp ứng tần số truyền qua các tần số thấp và chặn các tần số cao hơn tần số cắt f_c.

Đáp ứng tần số: Một bộ lọc thông thấp lý tưởng có đáp ứng tần số lý tưởng $H(e^{j\omega})$, được định nghĩa trong miền tần số như sau:

$$|H(e^{j\omega})| = \begin{cases} 1 & -\omega_c \le \omega \le \omega_c \\ 0 & \omega \text{ còn lại} \end{cases}$$

ω_c: là tần số cắt



Để tìm được đáp ứng xung $h_d(n)$ trong miền thời gian từ đáp ứng tần số $H(e^{j\omega})$ trong miền tần số, ta dùng biến đổi Fourier ngược.

Biến đổi Fourier ngược $H(e^{j\omega})$ cho ra hàm đáp ứng xung $h_d(n)$ như sau:

$$h_d(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} H(e^{j\omega})e^{j\omega n}d\omega = \frac{1}{2\pi}\int_{-\omega_c}^{\omega_c} e^{j\omega n}d\omega$$
$$= \frac{1}{2\pi}\left(\frac{e^{j\omega n}}{jn}\right)\Big|_{-\omega_c}^{\omega_c} = \frac{\sin(\omega_c n)}{\pi n}$$

Tương đương với:

$$h_d(n) = \int_{-\infty}^{\infty} H(f)e^{j2\pi fn}df = \int_{-f_c}^{f_c} e^{j2\pi fn}df$$
$$= \left(\frac{e^{j2\pi fn}}{j2\pi n}\right)\Big|_{-f_c}^{f_c} = \frac{\sin(2\pi f_c n)}{\pi n}$$

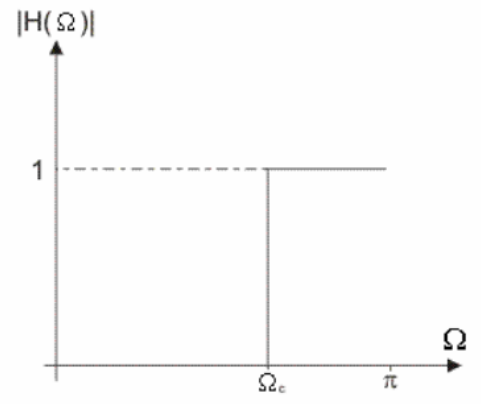
Trường hợp khi n = 0, công thức trên có dạng $\frac{0}{0}$, là một trường hợp không xác định. Sử dụng giới hạn của hàm sin khi n → 0, ta có: $h_d(0) = \frac{\omega_c}{\pi} = 2f_c$

Đáp ứng xung bộ lọc thông thấp lý tưởng: $$h_d(n) = \begin{cases} \frac{\sin(\omega_c n)}{\pi n}, & n \neq 0 \\ \frac{\omega_c}{\pi}, & n = 0 \end{cases}$$

Bộ lọc thông cao lý tưởng: Bộ lọc FIR thông cao có nhiệm vụ loại bỏ các thành phần tần số thấp trong tín hiệu và chỉ giữ lại các tần số cao hơn ngưỡng cắt xác định.

Đáp ứng tần số:

$$\left|H\left(e^{j\omega}\right)\right| = \begin{cases} 1 & |\omega| \geq \omega_c \\ 0 & |\omega| \leq \omega_c \end{cases}$$



Biến đổi Fourier ngược $H\left(e^{j\omega}\right)$ cho ra hàm đáp ứng xung $h_d(n)$ như sau:

$$h_d(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} H\left(e^{j\omega}\right)e^{j\omega n}d\omega = \frac{1}{2\pi}\left(\int_{-\pi}^{-\omega_c} e^{j\omega n}d\omega - \int_{\pi}^{\omega_c} e^{j\omega n}d\omega\right)$$

$$= \frac{1}{2\pi}\left(\frac{e^{-j\omega_c n} - e^{-j\pi n}}{jn} + \frac{e^{j\pi n} - e^{j\omega_c n}}{jn}\right) = \frac{\sin(\pi n) - \sin(\omega_c n)}{\pi n}$$

Khi $n = 0$, ta có: $h_d(0) = 1 - \frac{\omega_c}{\pi}$

Đáp ứng xung bộ lọc thông cao lý tưởng: $h_d(n) = \begin{cases} \frac{-\sin(\omega_c n)}{\pi n}, & n \neq 0 \\ 1 - \frac{\omega_c}{\pi}, & n = 0 \end{cases}$

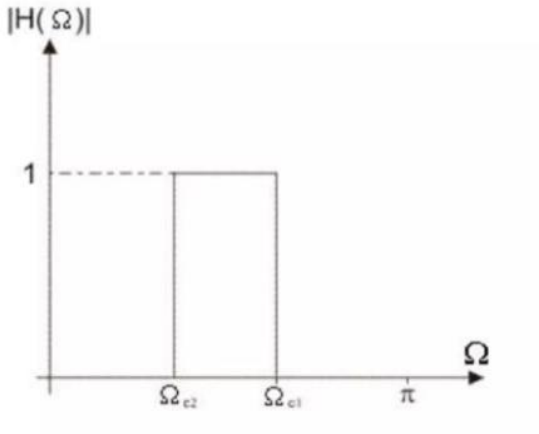
Tuy nhiên, nếu bộ lọc có chiều dài hữu hạn N, ta sẽ **cắt xén** đáp ứng xung lý tưởng này tại các chỉ số mẫu từ $\frac{-N-1}{2}$ đến $\frac{N-1}{2}$, và điều chỉnh hàm *sinc* sao cho nó trở thành một bộ lọc FIR hữu hạn bậc.

Để giảm độ dài bộ lọc và làm cho nó khả thi trong thực tế, người ta thường áp dụng một **hàm cửa sổ** (như cửa sổ Hamming, Hanning, hoặc Kaiser) để làm trơn đáp ứng xung lý tưởng này.

Bộ lọc thông giải: Bộ lọc cho phép các tần số nằm trong một dải tần số nhất định (giữa hai tần số cắt) đi qua, trong khi các tần số nằm ngoài dải này sẽ bị loại bỏ.

Bộ lọc này có *hai* tần số cắt:

- Tần số cắt dưới f_L (hoặc ω_L)
- Tần số cắt trên f_H (hoặc ω_H)



Đáp ứng tần số:

$$\left|H\left(e^{j\omega}\right)\right| = \begin{cases} 1 & \omega_{c2} \le |\omega| \le \omega_{c1} \\ 0 & |\omega| \le \omega_{c2},\ \ |\omega| \ge \omega_{c1} \end{cases}$$

Biến đổi Fourier ngược $H\left(e^{j\omega}\right)$ cho ra hàm đáp ứng xung $h_d(n)$ như sau:

$$\begin{aligned} h_d(n) &= \frac{1}{2\pi}\int_{-\pi}^{\pi} H\left(e^{j\omega}\right)e^{j\omega n}d\omega \\ &= \frac{1}{2\pi}\int_{-\omega_{c1}}^{\omega_{c1}} e^{j\omega n}d\omega - \frac{1}{2\pi}\int_{-\omega_{c2}}^{\omega_{c2}} e^{j\omega n}d\omega \\ &= \frac{1}{\pi n}\left(\sin(\omega_{c1}n) - \sin(\omega_{c2}n)\right) \end{aligned}$$

Khi $n = 0$, ta có: $h_d(0) = \frac{\omega_{c1}-\omega_{c2}}{\pi}$

Đáp ứng xung bộ lọc thông thấp lý tưởng:

$$h_d(n) = \begin{cases} \dfrac{\sin(\omega_{c1}n) - \sin(\omega_{c2}n)}{\pi n}, & n \ne 0 \\ \dfrac{\omega_{c1} - \omega_{c2}}{\pi}, & n = 0 \end{cases}$$

Đáp ứng xung của bộ lọc lý tưởng có độ dài vô hạn và không nhân quả.

Bộ lọc FIR luôn ổn định do độ dài $L[h(n)] = N$. N là số lẻ: để đảm bảo tính đối xứng của đáp ứng xung và giảm sai số trong quá trình thiết kế bộ lọc.

Nếu $h(n)$ không nhân quả, dịch $h(n)$ sang phải n_0 đơn vị thành $h(n - n_0)$, nhưng đáp ứng biên độ vẫn không đổi.

❖ Bộ lọc thực tế

Bộ lọc với đáp ứng xung hữu hạn (FIR)

Từ đáp ứng tần số mong muốn $H_d(\omega)$ với các chỉ tiêu tương ứng, ta lấy biến đổi Fourier ngược để có đáp ứng xung $h_d(n)$:

$$h_d(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} H\left(e^{j\omega}\right)e^{j\omega n}d\omega\,, -\infty < n \le +\infty$$

Nói chung, $h_d(n)$ thu được sẽ có chiều dài vô hạn và không nhân quả, ta không thể thực hiện được trong thực tế. Vì vậy, hệ thống phải được điều chỉnh lại thành nhân quả và buộc phải hạn chế chiều dài của $h_d(n)$.

Phương pháp đơn giản là dịch $h_d(n)$ đi n_0 đơn vị $\rightarrow h_d(n - n_0)$: nhân quả.

Phương pháp cửa sổ: Giới hạn số mẫu của hàm $h_d(n)$ bằng cách nhân với một hàm cửa sổ (windowing) $\rightarrow h(n) = h_d(n - n_0).w(n)$: ổn định.

Trong đó: $w(n)$ có chiều dài hữu hạn là $N - n_0$ và đối xứng chẵn quanh điểm giữa.

Các loại cửa sổ thông dụng:

- **Cửa sổ chữ nhật:** $W_R(\mathrm{n}) = \begin{cases} 1 & 0 \le \mathrm{n} \le \mathrm{N}-1 \\ 0 & n \text{ còn lại} \end{cases}$

- **Cửa sổ tam giác (Bartlett):** $W_T(\mathrm{n}) = \begin{cases} \frac{2n}{\mathrm{N}-1} & 0 \le \mathrm{n} \le \frac{\mathrm{N}-1}{2} \\ \frac{2-2n}{\mathrm{N}-1} & \frac{\mathrm{N}-1}{2} \le n \le \mathrm{N}-1 \\ 0 & n \text{ còn lại} \end{cases}$

- **Cửa sổ Hanning:** $W_{Han}(\mathrm{n}) = \begin{cases} 0{,}5 - 0{,}5\cos\left(\frac{2\pi n}{N-1}\right) & 0 \le \mathrm{n} \le \mathrm{N}-1 \\ 0 & n \text{ còn lại} \end{cases}$

- **Cửa sổ Hamming:**

$$W_{Ham}(\mathrm{n}) = \begin{cases} 0{,}54 - 0{,}46\cos\left(\frac{2\pi n}{N-1}\right) & 0 \le \mathrm{n} \le \mathrm{N}-1 \\ 0 & n \text{ còn lại} \end{cases}$$

- **Cửa sổ Blackman:**

$$W_B(\mathrm{n}) = \begin{cases} 0{,}42 - 0{,}5\cos\left(\frac{2\pi n}{N-1}\right) + 0{,}08\cos\left(\frac{4\pi n}{N-1}\right) & 0 \le \mathrm{n} \le \mathrm{N}-1 \\ 0 & n \text{ còn lại} \end{cases}$$

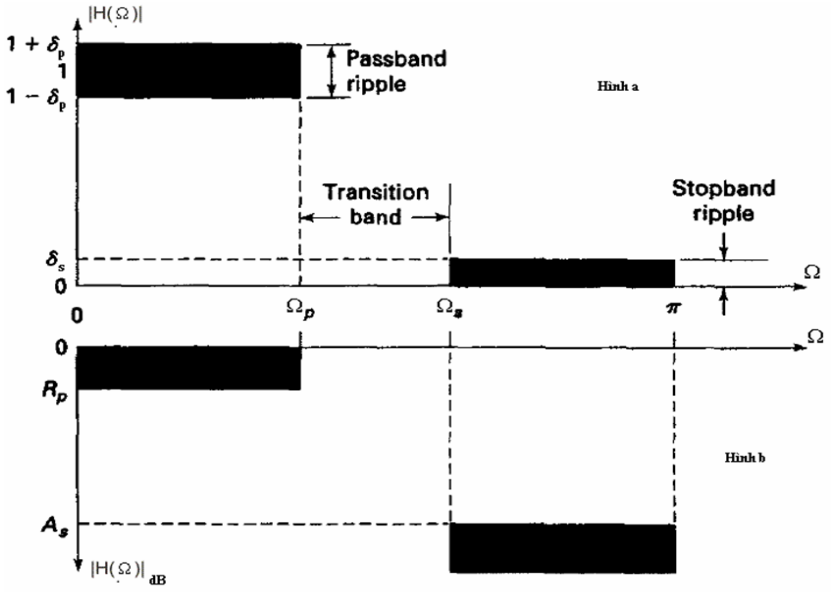
- **Cửa sổ Kaiser:**

$$W_k(\mathrm{n}) = \begin{cases} \frac{\mathrm{I}_0.\left[\beta.\left(\frac{n-1}{2}\right).\sqrt{1-\left(\frac{2n}{N-1}-1\right)^2}\right]}{\mathrm{I}_0.\beta.\left(\frac{n-1}{2}\right)} & 0 \le \mathrm{n} \le \mathrm{N}-1 \\ 0 & n \ne 1 \end{cases}$$

Chọn β: [4; 9] $\quad \mathrm{I}_0 = 1 + \sum_{k=1}^{\infty}\left[\frac{1}{k!}.\left(\frac{x}{2}\right)^k\right]^2$

3. Các đặc tả bộ lọc số

- Các tham số của bộ lọc: dải thông, dải chắn, dải chuyển tiếp, độ gợn dải thông, suy hao dải chắn.



- Xét bộ lọc thông thấp:

  δ_P : độ lệch dải thông

  δ_S : độ lệch dải chắn

  Độ gợn dải thông [dB] : $R_P = -20\log(1-\delta_P)$ *Đặc tả tuyệt đối (H.a)*

  Suy hao dải chắn [dB] : $A_S = -20\log\delta_S$ *Đặc tả tương đối (H.b)*

4. Các bước để thiết kế bộ lọc số

Quá trình thiết kế bộ lọc số gồm 3 bước:

- *Xác định các đặc tả của bộ lọc:* Tùy theo yêu cầu ứng dụng, ở bước này cần tiến hành xác định các đặc tả của bộ lọc: ω_p, ω_s, v.v.
- *Xác định giá trị các hệ số của bộ lọc:* Sau khi đã có đặc tả của bộ lọc, sử dụng các phương pháp thiết kế khác nhau: phương pháp dùng cửa sổ, phương pháp lấy mẫu tần số, phương pháp thiết kế tối ưu, v.v. để xác định các hệ số của bộ lọc h(n), $0 \le n \le N$.
- *Thực hiện mạch lọc:* Trên cơ sở đã có được các hệ số của bộ lọc, vấn đề thiết kế chỉ còn là việc lựa chọn sơ đồ thực hiện (dạng trực tiếp, dạng chính tắc) → xây dựng giải thuật tương ứng → viết chương trình → cài đặt.

→ Quá trình này có thể được thực hiện bằng phần cứng hoặc phần mềm.

III. Code MATLAB thiết kế bộ lọc

1. Các chương trình thiết kế bộ lọc

1.1. Code bộ lọc thông thấp

```
function h = designFIRLowpass(Fs, Fc, N)
% Hàm thiết kế bộ lọc FIR thông thấp sử dụng cửa sổ Hamming
%
% Đầu vào:
% Fs - Tần số lấy mẫu (Hz)
% Fc - Tần số cắt (Hz)
% N - Chiều dài của bộ lọc (số mẫu), phải là số lẻ
%
% Đầu ra:
% h - Hệ số bộ lọc FIR thông thấp
% Kiểm tra điều kiện chiều dài N là số lẻ
if mod(N, 2) == 0
error('Chiều dài N của bộ lọc phải là số lẻ.');
end
% Chuẩn hóa tần số cắt
fc_norm = Fc / (Fs / 2);
% Tính toán đáp ứng xung lý tưởng h_d[n]
n = 0:N-1;
n_middle = (N-1) / 2;
hd = sin(2 * pi * fc_norm * (n - n_middle)) ./ (pi * (n - n_middle));
hd(n_middle + 1) = 2 * fc_norm; % Xử lý tại vị trí 0 để tránh chia cho 0
% Tạo hàm cửa sổ Hamming
w = 0.54 - 0.46 * cos(2 * pi * n / (N - 1));
% Nhân đáp ứng xung lý tưởng với hàm cửa sổ
h = hd .* w;
% Hiển thị đáp ứng tần số của bộ lọc
fvtool(h, 1);
title('Đáp ứng tần số của bộ lọc FIR thông thấp');
end
```

1.2. Code bộ lọc thông cao:

```
function h = designFIRHighpass(Fs, Fc, N)
% Hàm thiết kế bộ lọc FIR thông cao sử dụng cửa sổ Hamming
%
% Đầu vào:
% Fs - Tần số lấy mẫu (Hz)
% Fc - Tần số cắt (Hz)
% N - Chiều dài của bộ lọc (số mẫu), phải là số lẻ
%
% Đầu ra:
% h - Hệ số bộ lọc FIR thông cao
% Kiểm tra điều kiện chiều dài N là số lẻ
if mod(N, 2) == 0
error('Chiều dài N của bộ lọc phải là số lẻ.');
end
% Chuẩn hóa tần số cắt
fc_norm = Fc / (Fs / 2);
% Tính toán đáp ứng xung lý tưởng h_d[n]
```

```
n = 0:N-1;
n_middle = (N-1) / 2;
hd_lowpass = sin(2 * pi * fc_norm * (n - n_middle)) ./ (pi * (n - n_middle));
hd_lowpass(n_middle + 1) = 2 * fc_norm; % Xử lý tại vị trí 0
hd = -hd_lowpass; % Đáp ứng xung cho thông cao
hd(n_middle + 1) = 1 - 2 * fc_norm; % Chỉnh giá trị tại tâm
% Tạo hàm cửa sổ Hamming
w = 0.54 - 0.46 * cos(2 * pi * n / (N - 1));
% Nhân đáp ứng xung lý tưởng với hàm cửa sổ
h = hd .* w;
% Hiển thị đáp ứng tần số của bộ lọc
fvtool(h, 1);
title('Đáp ứng tần số của bộ lọc FIR thông cao');
end
```

1.3. Code bộ lọc thông dải

```
function h = designFIRBandpass(Fs, Fc1, Fc2, N)
% Hàm thiết kế bộ lọc FIR thông dải sử dụng cửa sổ Hamming
%
% Đầu vào:
% Fs - Tần số lấy mẫu (Hz)
% Fc1 - Tần số cắt thấp (Hz)
% Fc2 - Tần số cắt cao (Hz)
% N - Chiều dài của bộ lọc (số mẫu), phải là số lẻ
%
% Đầu ra:
% h - Hệ số bộ lọc FIR thông dải
% Kiểm tra điều kiện chiều dài N là số lẻ
if mod(N, 2) == 0
error('Chiều dài N của bộ lọc phải là số lẻ.');
end
% Chuẩn hóa tần số cắt
fc1_norm = Fc1 / (Fs / 2);
fc2_norm = Fc2 / (Fs / 2);
% Tính toán đáp ứng xung lý tưởng h_d[n]
n = 0:N-1;
n_middle = (N-1) / 2;
% Đáp ứng xung cho thông thấp với Fc2
hd_lowpass2 = sin(2 * pi * fc2_norm * (n - n_middle)) ./ (pi * (n - n_middle));
hd_lowpass2(n_middle + 1) = 2 * fc2_norm;
% Đáp ứng xung cho thông thấp với Fc1
hd_lowpass1 = sin(2 * pi * fc1_norm * (n - n_middle)) ./ (pi * (n - n_middle));
hd_lowpass1(n_middle + 1) = 2 * fc1_norm;
% Đáp ứng xung cho thông dải
hd = hd_lowpass2 - hd_lowpass1;
% Tạo hàm cửa sổ Hamming
w = 0.54 - 0.46 * cos(2 * pi * n / (N - 1));
% Nhân đáp ứng xung lý tưởng với hàm cửa sổ
h = hd .* w;
% Hiển thị đáp ứng tần số của bộ lọc
fvtool(h, 1);
title('Đáp ứng tần số của bộ lọc FIR thông dải');
end
```

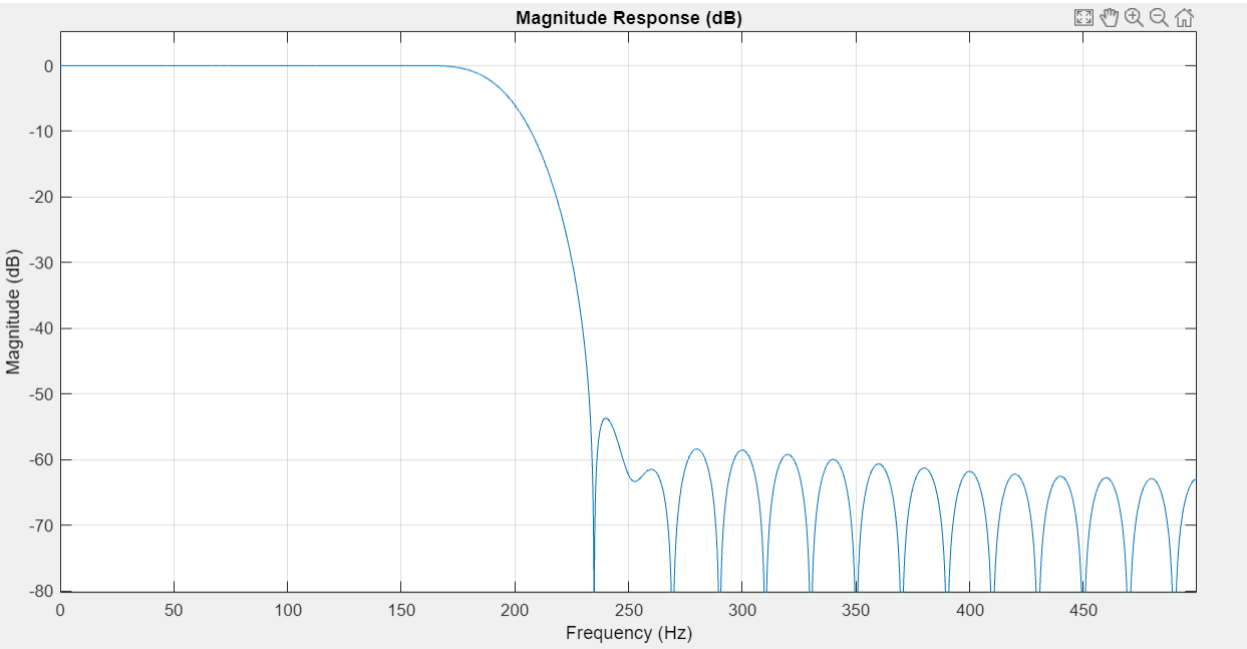
1.4. Code hàm áp dụng bộ lọc

```
function y = applyFIRFilter(h, x)
% Hàm thực hiện phép lọc FIR thông qua tích chập (convolution)
%
% Đầu vào:
% h - Hệ số của bộ lọc FIR
% x - Tín hiệu đầu vào
%
% Đầu ra:
% y - Tín hiệu đầu ra sau khi lọc
% Thực hiện phép tích chập giữa tín hiệu và hệ số bộ lọc
y_conv = conv(x, h, 'full'); % Tích chập đầy đủ
% Để đầu ra có cùng độ dài với tín hiệu gốc:
% Cắt bớt giá trị dư thừa ở phần đầu và phần cuối
filter_delay = (length(h) - 1) / 2; % Độ trễ của bộ lọc
y = y_conv(filter_delay + 1:end - filter_delay); % Loại bỏ biên dư thừa
end
```

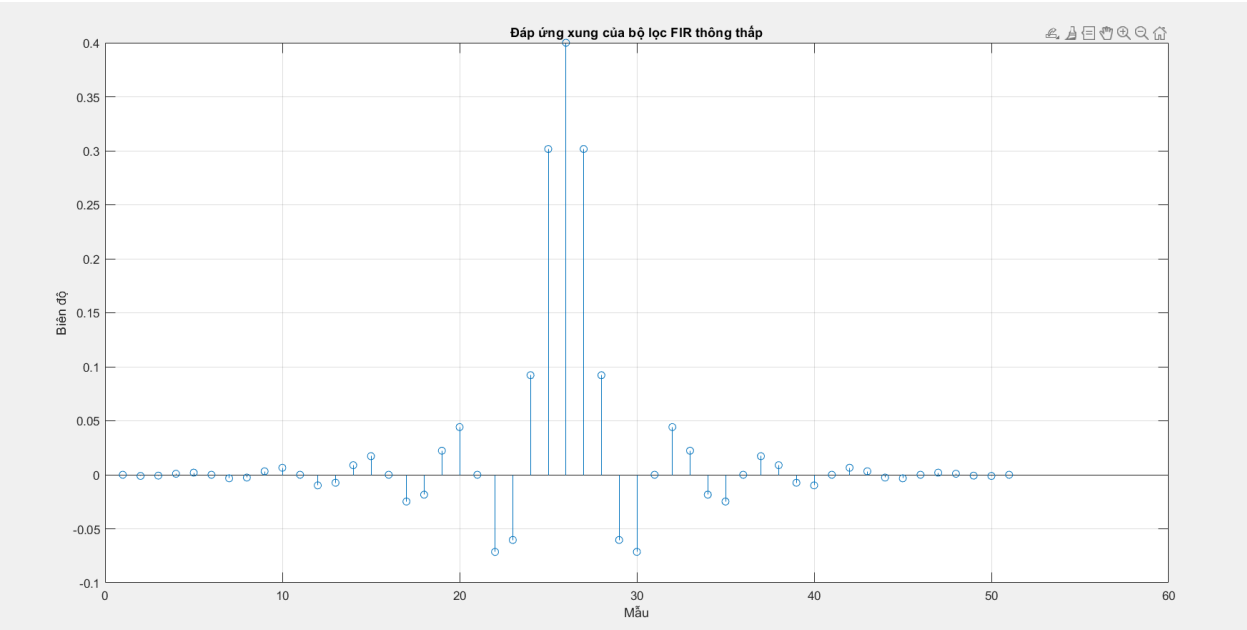
2. Áp dụng bộ lọc

2.1. Bộ lọc thông thấp

```
Fs = 1000;
Fc = 100;
N = 51;
h = designFIRLowpass(Fs, Fc, N);
% Hiển thị đáp ứng xung
figure;
stem(h);
title('Đáp ứng xung của bộ lọc FIR thông thấp');
xlabel('Mẫu');
ylabel('Biên độ');
grid on;
```



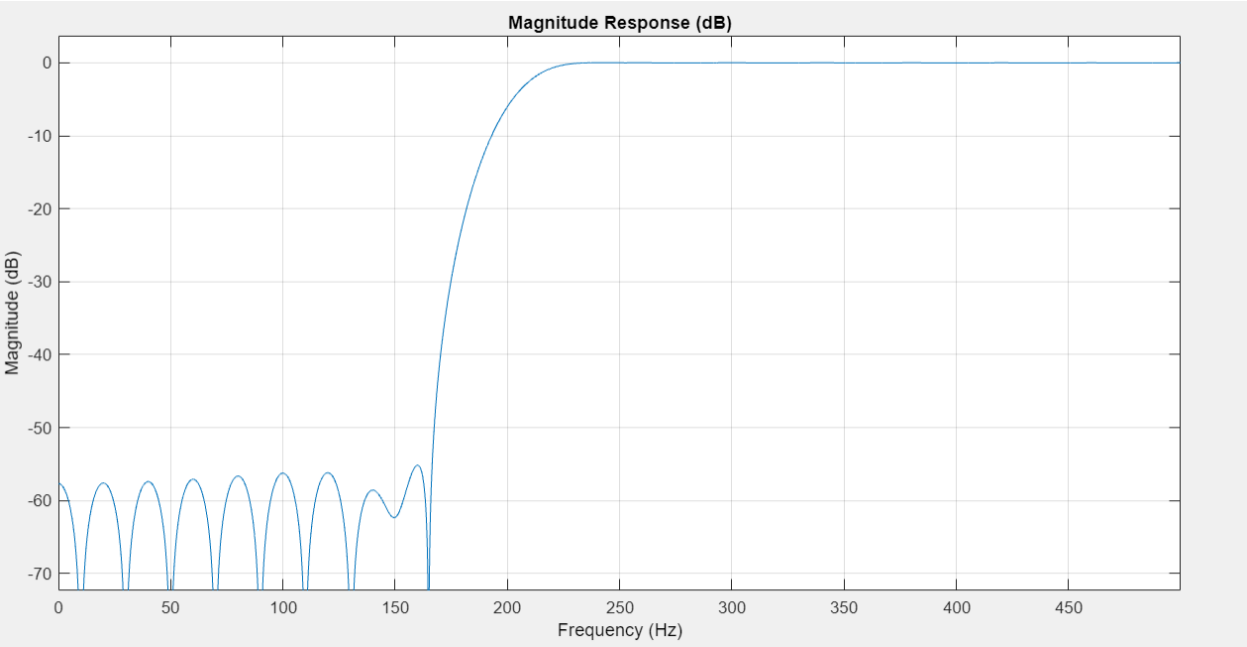
Đáp ứng tần số của bộ lọc



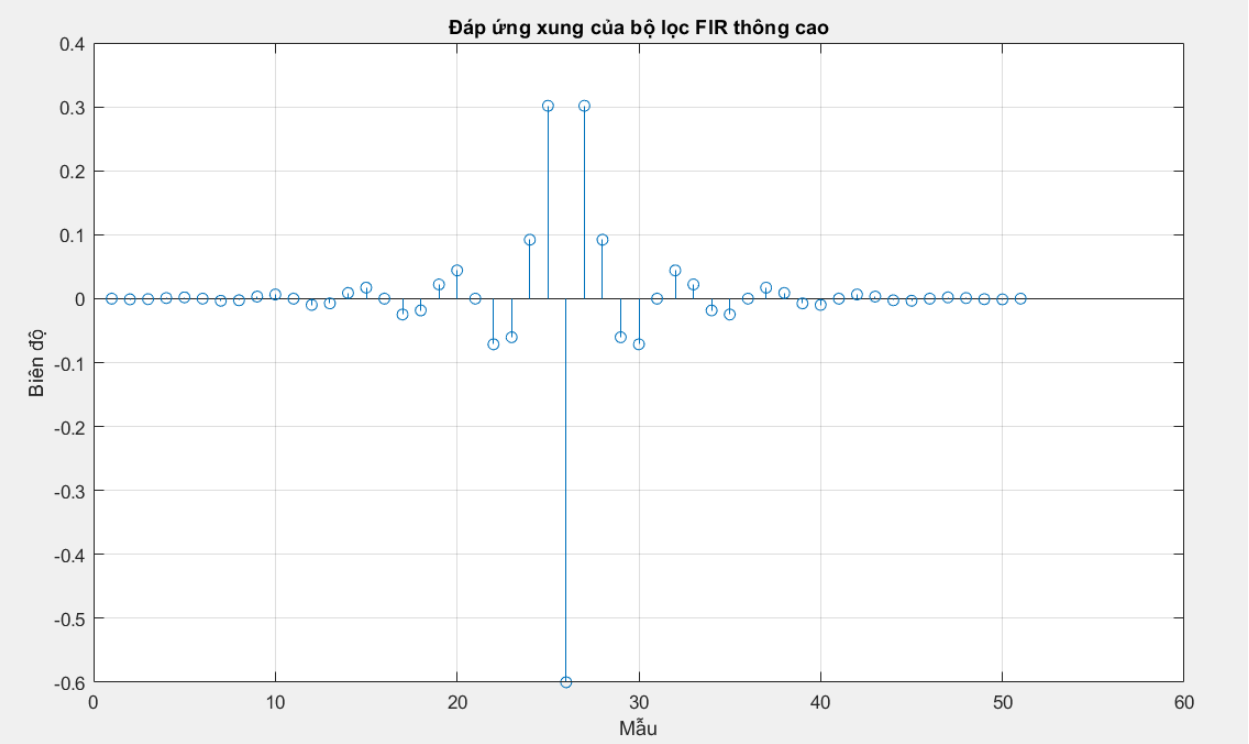
Đáp ứng xung của bộ lọc

2.2. Bộ lọc thông cao

```
Fs = 1000;
Fc = 400;
N = 51;
h = designFIRHighpass(Fs, Fc, N);
% Hiển thị đáp ứng xung
figure;
stem(h);
title('Đáp ứng xung của bộ lọc FIR thông cao');
xlabel('Mẫu');
ylabel('Biên độ');
grid on;
```



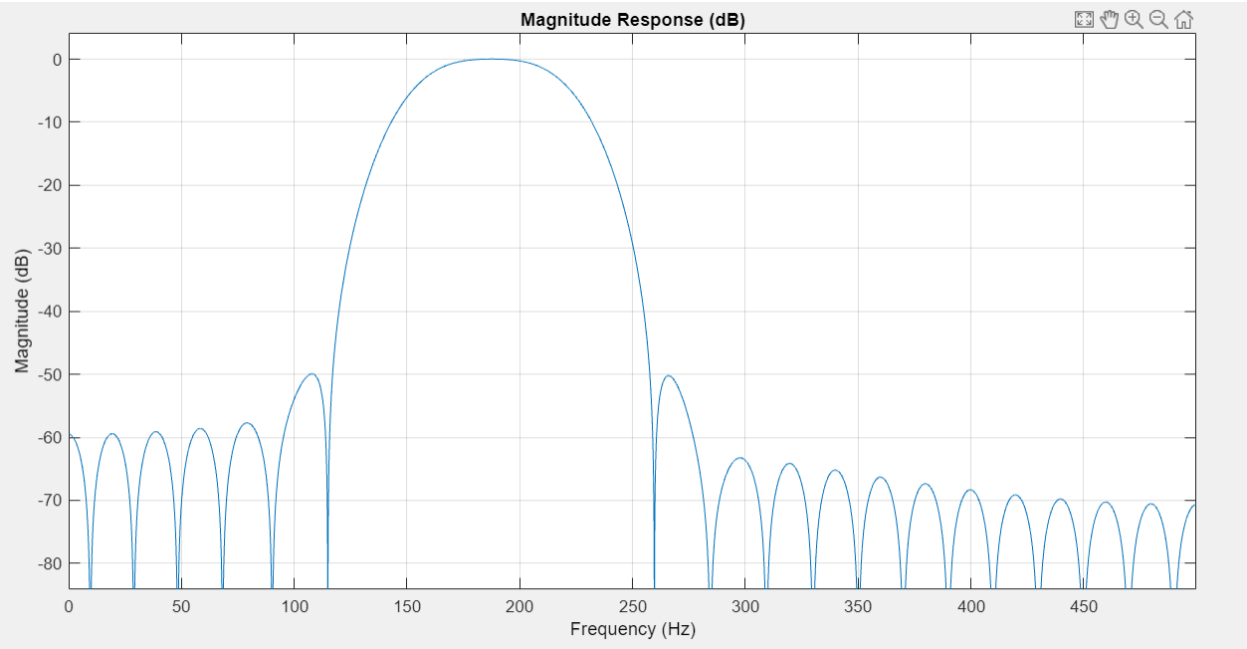
Đáp ứng tần số của bộ lọc



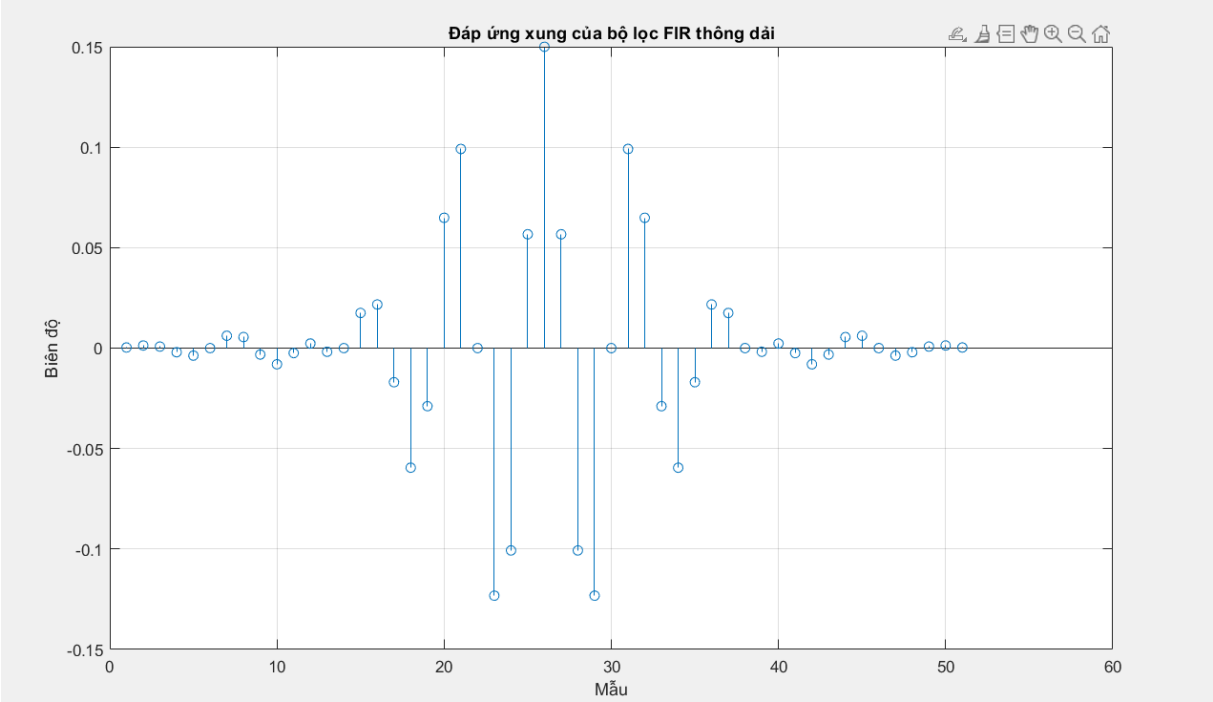
Đáp ứng xung của bộ lọc

2.3. Bộ lọc thông dải

```
% Thông số thiết kế
Fs = 8000;                  % Tần số lấy mẫu (Hz)
Fc1 = 600;                  % Tần số cắt thấp (Hz)
Fc2 = 900;                  % Tần số cắt cao (Hz)
N = 51;                     % Bậc bộ lọc (lẻ)
% Gọi hàm thiết kế
h = designFIRBandpass(Fs, Fc1, Fc2, N);
% Hiển thị đáp ứng xung
figure;
stem(h);
title('Đáp ứng xung của bộ lọc FIR thông dải');
xlabel('Mẫu');
ylabel('Biên độ');
grid on;
```



Đáp ứng xung của bộ lọc

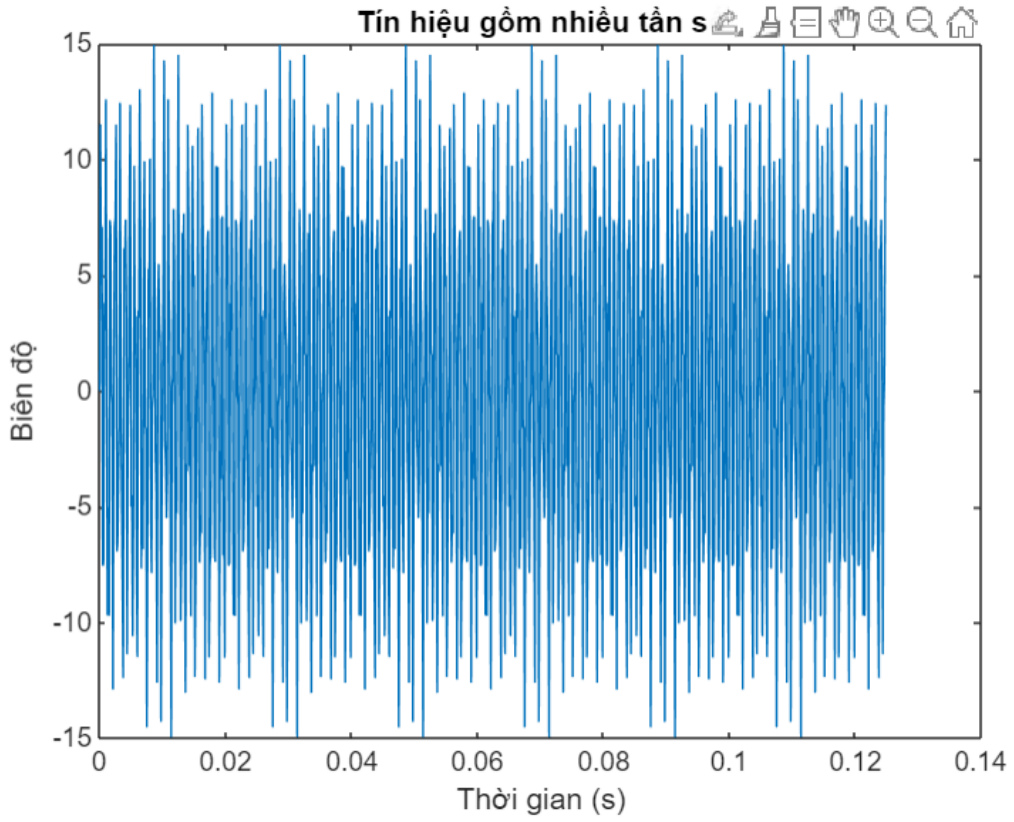


Đáp ứng tần số của bộ lọc

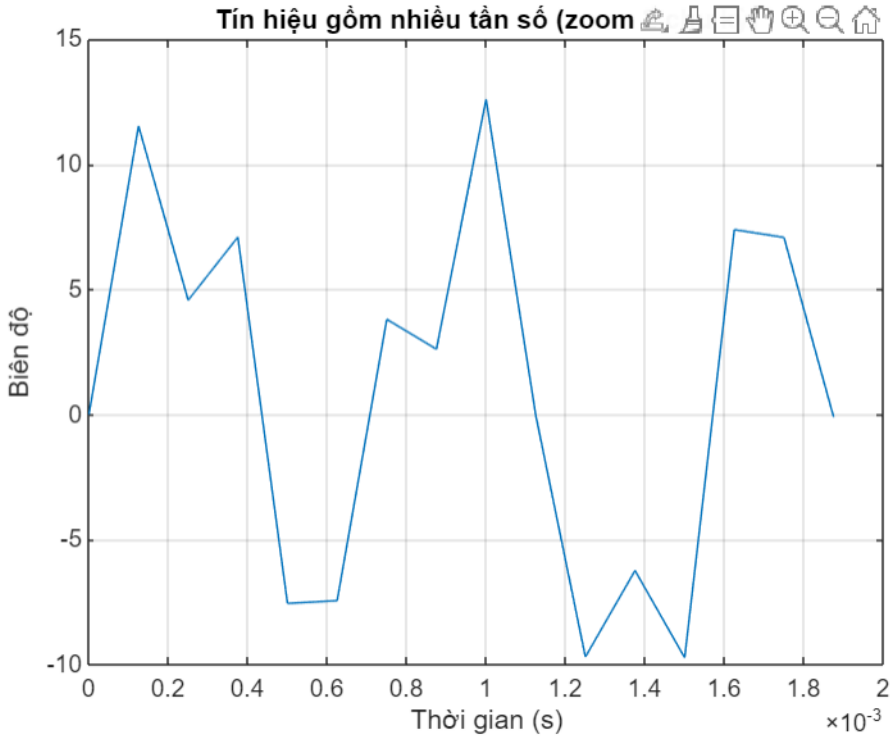
IV. Ứng dụng:

❖ TẠO TÍN HIỆU ÂM THANH VỚI 3 TẦN SỐ KHÁC NHAU:

```
% Thông số cơ bản
Fs = 8000; % Tần số lấy mẫu (Hz)
duration = 5; % Thời gian tín hiệu (giây)
t = 0:1/Fs:duration-1/Fs; % Thời gian mẫu
% Tạo tín hiệu với nhiều tần số
f1 = 500; % Tần số 500 Hz
f2 = 900; % Tần số 900 Hz
f3 = 1300; % Tần số 1300 Hz
signal = 2*sin(2*pi*f1*t) + 5*sin(7*pi*f2*t) + 9*sin(2*pi*f3*t);
% Lưu tín hiệu vào tệp âm thanh
audiowrite('multi_frequency_signal.wav', signal, Fs);
% Vẽ tín hiệu để kiểm tra
figure;
plot(t(1:1000), signal(1:1000)); % Hiển thị 1000 mẫu đầu tiên
title('Tín hiệu gồm nhiều tần số');
xlabel('Thời gian (s)');
ylabel('Biên độ');
[data, fs] = audioread("C:\Users\Phat\Documents\MATLAB\multi_frequency_signal.wav")%
Đường dẫn
```



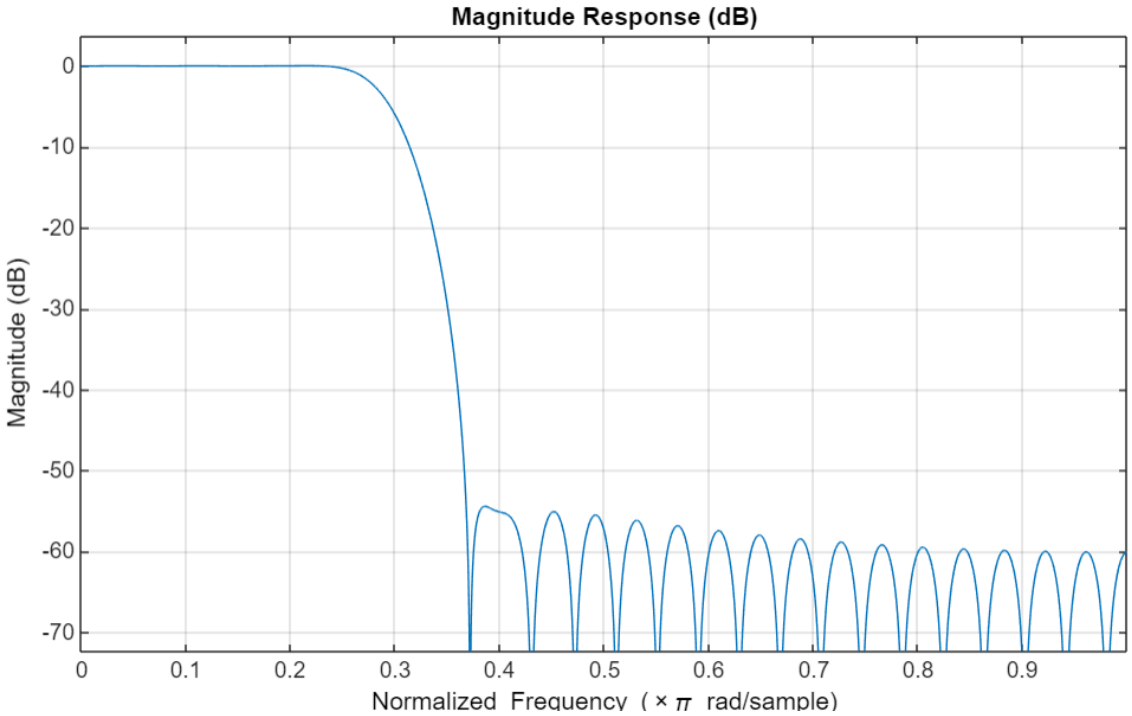

```
% Hiển thị tín hiệu trong khoảng 0.002 giây (tương ứng 2 chu kỳ của tần số 1000 Hz)
figure;
plot(t(1:round(0.002*Fs)), signal(1:round(0.002*Fs)));
title('Tín hiệu gồm nhiều tần số (zoom 2 chu kỳ)');
xlabel('Thời gian (s)');
ylabel('Biên độ');
grid on;
```

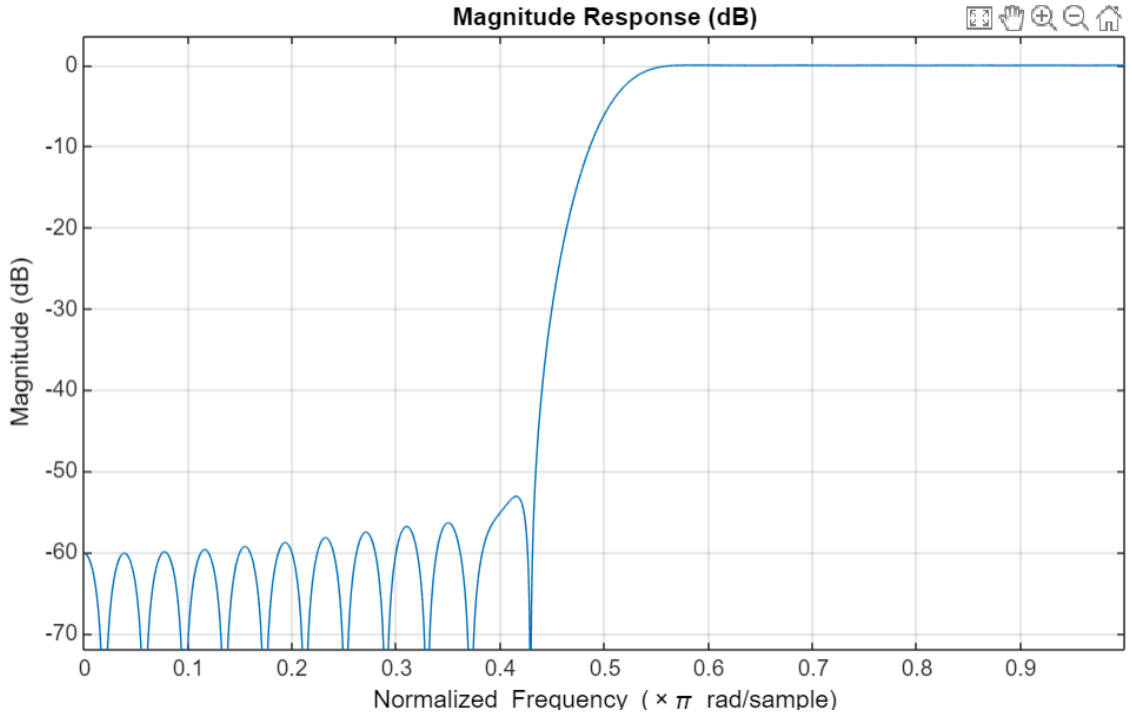


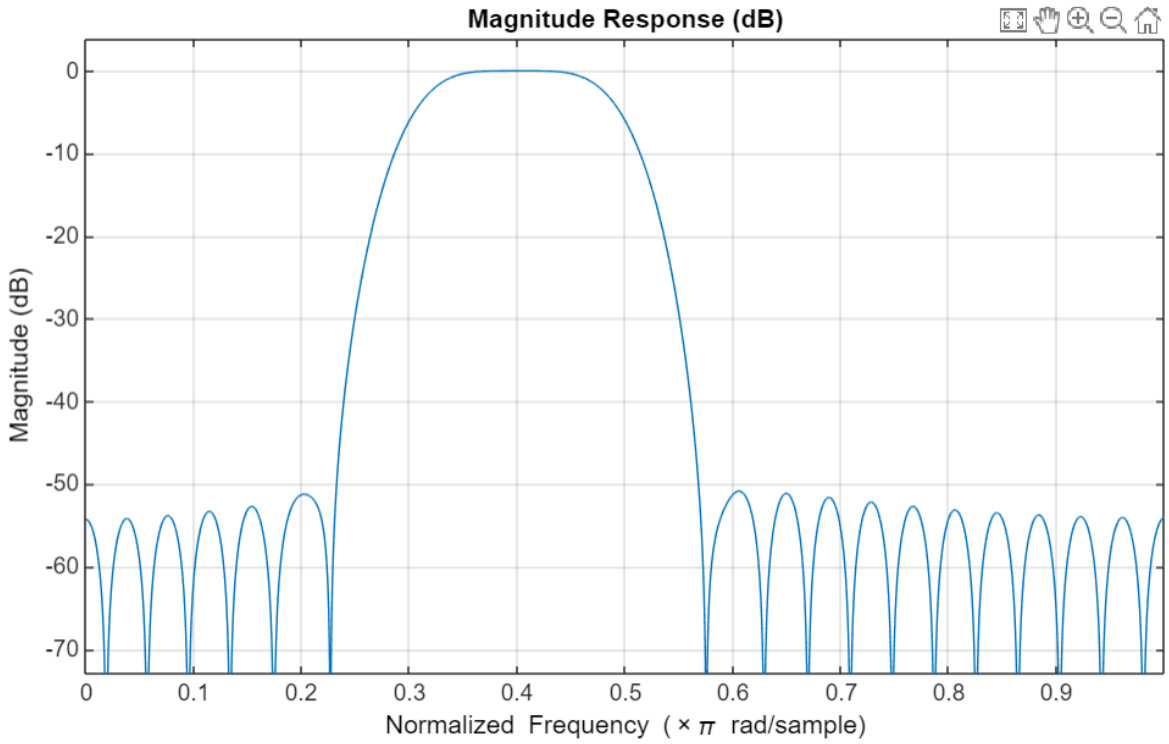
❖ THIẾT KẾ ÁP DỤNG BỘ LỌC CHO TÍN HIỆU ÂM THANH TRÊN:

```
% --- Thông số cơ bản ---
Fs = 8000; % Tần số lấy mẫu (Hz)
duration = 5; % Thời gian tín hiệu (giây)
t = 0:1/Fs:duration-1/Fs; % Trục thời gian
% --- Tạo tín hiệu âm thanh ---
f1 = 500; % Tần số 500 Hz
f2 = 900; % Tần số 900 Hz
f3 = 1300; % Tần số 1300 Hz
signal = 2*sin(2*pi*f1*t) + 5*sin(7*pi*f2*t) + 9*sin(2*pi*f3*t);
% --- Thiết kế bộ lọc ---
% Bộ lọc thông thấp
Fc_lowpass = 600; % Tần số cắt 600 Hz
N_lowpass = 51; % Độ dài bộ lọc
h_lowpass = designFIRLowpass(Fs, Fc_lowpass, N_lowpass);
% Bộ lọc thông cao
Fc_highpass = 1000; % Tần số cắt 1000 Hz
N_highpass = 51;
h_highpass = designFIRHighpass(Fs, Fc_highpass, N_highpass);
% Bộ lọc thông dải
Fc1_bandpass = 600; % Tần số cắt thấp
Fc2_bandpass = 1000; % Tần số cắt cao
N_bandpass = 51;
h_bandpass = designFIRBandpass(Fs, Fc1_bandpass, Fc2_bandpass, N_bandpass);
% --- Áp dụng bộ lọc ---
```

```
output_lowpass = applyFIRFilter(h_lowpass, signal);
output_highpass = applyFIRFilter(h_highpass, signal);
output_bandpass = applyFIRFilter(h_bandpass, signal);
% --- Lưu tín hiệu sau khi lọc thành tệp âm thanh ---
audiowrite('filtered_lowpass.wav', output_lowpass, Fs);
audiowrite('filtered_highpass.wav', output_highpass, Fs);
audiowrite('filtered_bandpass.wav', output_bandpass, Fs);
% --- Xác định khoảng 2 chu kỳ để vẽ ---
f_min = f1; % Tần số thấp nhất trong tín hiệu
T_min = 1 / f_min; % Chu kỳ (giây)
num_samples = round(2 * T_min * Fs * 5); % Số mẫu tương ứng với 2 chu kỳ
% Lấy tín hiệu 2 chu kỳ
t_plot = t(1:num_samples);
signal_plot = signal(1:num_samples);
output_lowpass_plot = output_lowpass(1:num_samples);
output_highpass_plot = output_highpass(1:num_samples);
output_bandpass_plot = output_bandpass(1:num_samples);
% --- Vẽ tín hiệu ---
figure;
% Tín hiệu gốc
subplot(4, 1, 1);
plot(t_plot, signal_plot);
title('Tín hiệu gốc (trước khi lọc)');
xlabel('Thời gian (s)');
ylabel('Biên độ');
% Tín hiệu sau khi lọc thông thấp
subplot(4, 1, 2);
plot(t_plot, output_lowpass_plot);
title('Tín hiệu sau khi lọc FIR thông thấp');
xlabel('Thời gian (s)');
ylabel('Biên độ');
% Tín hiệu sau khi lọc thông cao
subplot(4, 1, 3);
plot(t_plot, output_highpass_plot);
title('Tín hiệu sau khi lọc FIR thông cao');
xlabel('Thời gian (s)');
ylabel('Biên độ');
% Tín hiệu sau khi lọc thông dải
subplot(4, 1, 4);
plot(t_plot, output_bandpass_plot);
title('Tín hiệu sau khi lọc FIR thông dải');
xlabel('Thời gian (s)');
ylabel('Biên độ');
```


Magnitude Response (dB)
Magnitude (dB)
0
-10
-20
-30
-40
-50
-60
-70
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
Normalized Frequency (× π rad/sample)


Magnitude Response (dB)
Magnitude (dB)
0
-10
-20
-30
-40
-50
-60
-70
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
Normalized Frequency (× π rad/sample)



❖ TÍN HIỆU ÂM THANH SAU KHI ĐI QUA CÁC BỘ LỌC: