Chương 5
MÔ HÌNH BỘ LỌC VÀ KỸ THUẬT MÔ PHỎNG

5.1. Mở đầu

Chương 5 tập trung triển khai các mô hình mô phỏng cho bộ lọc. Bộ lọc là một bộ phận quan trọng của rất nhiều phân hệ trong hệ thống tổng thể. Để mô phỏng, ta phải chuyển bộ lọc tương tự thành bộ lọc số một cách phù hợp. Có nhiều kỹ thuật khả dụng, chúng đều thực hiện xấp xỉ và đều gây ra lỗi trong kết quả mô phỏng. Chương này sẽ khai thác các kỹ thuật hữu hiệu nhất để tổng hợp và mô phỏng bộ lọc. Ngoài ra, cũng đề cập các hạn chế và nguồn lỗi.

Theo định nghĩa và phân loại, thì bộ lọc có tính chọn lọc tần số và đáp ứng xung kim thuộc loại đáp ứng xung kim hữu hạn FIR hoặc đáp ứng xung kim vô hạn IIR. Do tính chọn lọc tần số nên nó có tính nhớ, tín hiệu ra của bộ lọc tại thời điểm hiện tại được tính từ tín hiệu vào hiện thời và trước đó. Theo đó, bộ lọc cần có bộ lưu trữ, việc lưu trữ và lấy các giá trị mẫu làm tăng đáng kể mức độ tính toán trong chương trình mô phỏng, đồng nghĩa với tăng thời gian mô phỏng. Vì vậy, cần có các thuật toán sao cho giảm tải tính toán.

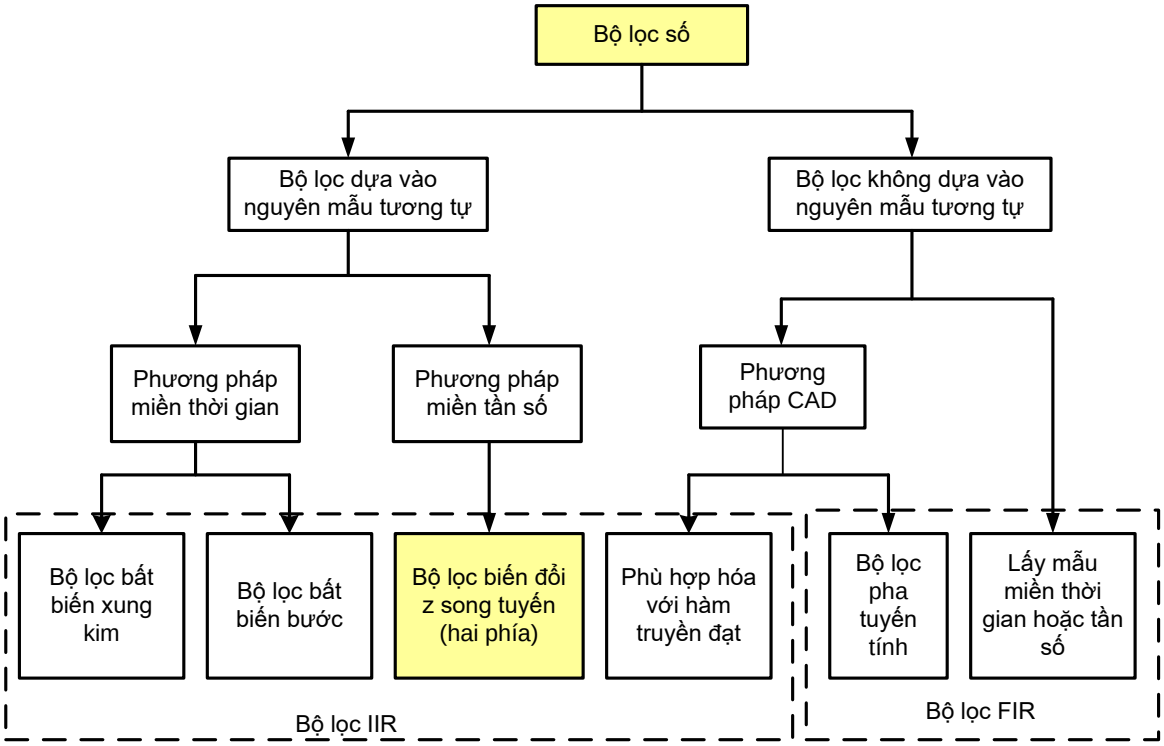
Chương này không trình bày chi tiết kỹ thuật thiết kế bộ lọc số (nhiều giáo trình đã đề cập thiết kế bộ lọc số và các kỹ thuật cơ bản đã được sử dụng trong nhiều năm) mà chỉ tóm các kỹ thuật hữu hiệu nhất và nhấn mạnh *lỗi lấy xấp xỉ* trong mô phỏng.

Tổng hợp một số kỹ thuật triển khai các bộ lọc số được minh họa ở hình 5.1. Các kỹ thuật truyền thống thường dựa trên các nguyên mẫu tương tự. Khi này, triển khai bộ lọc số trong mô hình mô phỏng bắt đầu bằng hàm truyền đạt miền s (Laplace) của bộ lọc tương tự từ đó tìm phần tử số tương đương. Sau đó, bài toán được quy về tìm bộ lọc số tương đương phù hợp với nguyên mẫu bộ lọc tương tự. Nhiều phương pháp nền tảng dựa trên đánh giá tính tương đương trong tiêu chuẩn miền thời gian hoặc tần số. Mặc dù, thường lọc theo các *đặc tính chọn lọc tần số* nhưng khi dùng tiêu chuẩn miền thời gian dẫn đến hiệu quả. Tiêu chuẩn miền thời gian được áp dụng bằng cách qui định đầu ra bộ lọc số phù hợp với đầu ra được lấy mẫu của nguyên mẫu tương tự tương ứng. Hai kỹ thuật tổng hợp nền tảng dựa trên tiêu chuẩn miền thời gian là *bộ lọc số bất biến xung kim* và *bộ lọc số bất biến bước,* trong đó: *(i) Bộ lọc số bất biến xung kim* là một thiết kế trong đó đáp ứng xung kim của bộ lọc số phù hợp với đáp ứng xung kim của nguyên mẫu tương tự được lấy mẫu; *(ii) Bộ lọc số bất biến bước,* đáp ứng *bước* của bộ lọc số phù hợp với *đáp ứng bước* của nguyên mẫu tương tự được lấy mẫu.

Nhiều thiết kế khác sử dụng "*các tín hiệu kiểm tra*" chứ không phải là các bước đơn vị hay các bước xung kim. Ta sẽ thấy, nếu nguyên mẫu tương tự và bộ lọc số là tương đương nhau trong miền thời gian thì cũng tương đương trong miền tần số, ít nhất đối với các giá trị tần số nhỏ so với tần số lấy mẫu.

Có lẽ phương pháp phổ biến nhất để ánh xạ một nguyên mẫu tương tự thành số là thông qua biến đổi z song tuyến tính (hai phía). Ở đây sử dụng biến đổi z song tuyến có hàm ý hướng vào ứng dụng, hàm ý về cấu trúc và nguyên lý hoạt động hơn là mô tả toán học thuần túy. Phương pháp tổng hợp biến đổi z-hai phía (kỹ thuật đại số hoàn toàn) được thực thi sao cho cả nguyên mẫu tương tự và bộ lọc số có cùng đáp ứng tần số (pha và độ lớn) tại các giá trị tần số thiết kế cụ thể. Ngoài ra, phương pháp biến đổi z-hai phía khử các lỗi chồng phổ ở vùng mở rộng phổ tần do tính phi tuyến.

Kỹ thuật tổng hợp dựa trên nguyên mẫu tương tự tạo ra bộ lọc số IIR (đáp ứng xung kim vô hạn). Đáp ứng xung kim được tạo ra bởi một trong các phương pháp thiết kế IIR tiêu chuẩn được cắt ngắn để tạo thành bộ lọc FIR. Có thể giảm lỗi do việc cắt ngắn xuống mức chấp nhận bằng cách tính đến số lượng lớn thành phần trong đáp ứng xung kim của bộ lọc FIR kết quả.



Hình 5.1: Phân loại bộ lọc số

Thuộc tính quan trọng của bộ lọc số là có thể triển khai bộ lọc mà không cần có nguyên mẫu tương tự. Hầu hết các bộ lọc số quan trọng thuộc loại này, là các bộ lọc cho phép xấp xỉ hóa với một đáp ứng biên độ cho trước mà vẫn duy trì được đáp ứng pha pha tuyến tính một cách hoàn hảo. Đây là bộ lọc FIR được thực hiện theo cấu trúc đường trễ rẽ nhánh TDL. Tồn tại nhiều kỹ thuật thiết kế cho các bộ lọc này. Phương pháp nền tảng nhất là khai triển đáp ứng biên độ, là tuần hoàn theo tần số lấy mẫu trong chuỗi Fourier. Các hệ số Fourier xác định đáp ứng xung kim của bộ lọc số. Có thể dùng FFT để thực hiện quá trình này. Đây là một ví dụ về *lấy mẫu tần số,* bởi lẽ đáp ứng tần số được *"được lấy mẫu"* tại nhiều điểm khác nhau trong

miền tần số. Thực hiện IFFT lên các mẫu tần này cho ta đáp ứng xung kim của bộ lọc. Lấy tích chập tín hiệu vào bộ lọc với đáp ứng xung kim (thực hiện mô hình mô phỏng của bộ lọc) cho ta tín hiệu đầu ra của bộ lọc.

Tồn tại nhiều kỹ thuật thiết kế được hỗ trợ bởi máy tính CAD để thiết kế các bộ lọc số. Trong chương này ta khai thác hai trong số chúng. Kỹ thuật đầu tiên dẫn đến bộ lọc IIR, kỹ thuật thứ hai dẫn đến bộ lọc FIR pha tuyến tính.

Có thể tham khảo phần Phụ lục 5B "Practical FIR Filter Design in MATLAB" để được rõ hơn về việc thiết kế bộ lọc FIR thực tế.

5.2. Bộ lọc IIR và FIR

Như đã đề cập, ta thường phân loại bộ lọc số theo khoảng thời gian đáp ứng xung kim (FIR hay IIR). Gắn liền với phân loại đáp ứng xung kim là thực hiện cấu trúc. Tại đây ta xét các mô hình bộ lọc khác nhau.

5.2.1. Bộ lọc IIR

Một bộ vi xử lý tín hiệu số tuyến tính (bộ lọc số) tính tín hiệu ra hiện thời $y[n]$ là tổng trọng số của N mẫu đầu ra trước $y[n-k],\ 1 \le k \le N$, mẫu đầu vào hiện thời $x[n]$ và N mẫu vào trước $x[n-k],\ 1 \le k \le N$. Nói cách khác, thuật toán để tính đầu ra hiện thời theo các đầu vào/ra trước đó là:

$$y[n] = \sum_{k=0}^{N} b_k x[n-k] - \sum_{k=1}^{N} a_k y[n-k] \tag{5.1}$$

Tồn tại nhiều thuật toán hiệu quả để thực hiện (5.1) trong chương trình mô phỏng.

Thấy rõ, nếu một trọng số nào đó trong (5.1), b_k hay a_k, $k \ge 1$ khác không thì bộ vi xử lý có tính nhớ, có tính chọn lọc tần số và được coi là một bộ lọc. *Trường hợp hệ thống thay đổi theo thời gian, thì một hoặc một vài trọng số sẽ là hàm của chỉ số n*. Hàm truyền đạt *H(z)* là biến đổi z hai phía của (5.1). Cần lưu ý rằng: *(i)* biến đổi z là phép toán tuyến tính vì vậy biến đổi của một tổng là tổng của các biến đổi; *(ii)* khi làm trễ *k* mẫu là tương đương với việc nhân với z^{-k}. Dẫn đến:

$$Y[z]\left[1 + \sum_{k=1}^{N} a_k z^{-k}\right] = X(z) \sum_{k=0}^{N} b_k z^{-k} \tag{5.2}$$

Theo đó, ta có hàm truyền đạt:

$$H[z] = \frac{Y[z]}{X(z)} = \frac{\sum_{k=0}^{N} b_k z^{-k}}{\left[1 + \sum_{k=1}^{N} a_k z^{-k}\right]} \tag{5.3}$$

Đây là dạng tổng quát của bộ lọc *tuyến tính bất biến*.

Trong các ứng dụng được xét ở đây, ta quan tâm đáp ứng xung kim và đáp ứng tần số của của bộ lọc. Đáp ứng xung kim $h[n]$ là biến đổi z ngược của hàm truyền đạt *H(z).* Tìm đáp ứng tần số bằng cách thay $z = e^{j2\pi fT}$ vào hàm truyền đạt. Nói cách khác, đáp ứng tần số là:

$$H(z)\big|_{z=e^{j2\pi fT}} = H\left(e^{j2\pi fT}\right) \quad 5.4)$$

Tìm được đáp ứng xung kim của bộ lọc số $h[n]$ bằng cách đặt $x[n] = \delta[n]$, với:

$$\delta[n] = \begin{cases} 1, & n=0 \\ 0, & n \neq 0 \end{cases} \quad (5.5)$$

vào (5.1). Do bản chất đệ qui của (5.1) (nghĩa là $y[n]$ là *hàm* của $y[n-1]$) nên ở dạng tổng quát đáp ứng xung kim $h[n]$ sẽ có khoảng thời gian vô hạn và được gọi bộ lọc đáp ứng xung kim vô hạn IIR. Cần lưu ý rằng, đáp ứng xung kim $h[n]$ là hàm rời rạc của chỉ số n, trong khi đáp ứng tần số là một hàm liên tục của biến tần số liên tục *f*.

5.2.2. Bộ lọc FIR

Có được bộ lọc FIR nếu $a_k = 0, k \geq 1$ trong (5.1) (tương đương trong (5.3)) Theo đó, hàm truyền đạt và đáp ứng xung kim của bộ lọc FIR là:

$$H[z] = \sum_{k=0}^{N} b_k z^{-k} \quad (5.6)$$

$$h[n] = \sum_{k=0}^{N} b_k .\delta[n-k] \quad (5.7)$$

khác không chỉ khi $0 \leq n \leq N$. Do đó, đáp ứng xung kim có nhiều nhất *N+1* thành phần khác không và có khoảng thời gian hữu hạn. Thuật toán để tạo chuỗi đầu ra bộ lọc $y[n]$ từ chuỗi đầu vào *x[n]* là một tổng chập:

$$y[n] = \sum_{k=0}^{N} b_k .x[n-k] = \sum_{k=0}^{N} h[k].x[n-k] \quad (5.8)$$

Suy ra trực tiếp từ (5.1) khi $a_k = 0, k > 0$

5.2.3. Tổng hợp và mô phỏng

Cần phải chỉ ra rằng, bộ lọc trong chương trình mô phỏng gồm hai phép toán rất khác biệt: *(i) Phép toán thứ nhất* là *tổng hợp*, ta phải xác định rõ các yêu cầu lọc, hàm truyền đạt của bộ lọc H(z), hàm H(z) đáp ứng các yêu cầu lọc đã định. Điều này thiết lập mô hình mô phỏng. Kết quả của quá trình tổng hợp thường được biểu diễn ở dạng 2 véc-tơ, một véc-tơ chứa các hệ số ở mẫu a_k, một véc-tơ chứa các hệ số ở tử số b_k. Hai véc-tơ này sẽ tạo hàm truyền đạt (5.3) và thuật toán tạo đầu ra bộ lọc khi biết đầu vào. Khối lượng tính toán của quá trình tổng hợp không quá lớn, thậm chí khi sử dụng thuật toán phức tạp, vì tổng hợp chỉ thực hiện duy nhất 1 lần do vậy nó nằm ngoài vòng lặp mô phỏng chính; *(ii) Phép toán thứ hai* là tính tín hiệu ra bộ lọc tại mỗi bước thời gian mô phỏng (nghĩa là tại mỗi nhịp của đồng hồ mô phỏng). Quá trình

này nhiều khi lặp lại hàng triệu lần, thậm chí là hàng tỉ lần trong chương trình mô phỏng Monte Carlo. Do vậy, khối lượng tính toán cho quá trình này phải được giảm thiểu để thời gian chạy mô phỏng hợp lý. Cấu trúc bộ lọc chuyển vị trong mục 5.3 dành cho chủ đề này.

5.3. Thực hiện bộ lọc IIR và FIR

Ta khảo sát vắn tắt cách thực thi bộ lọc số trong chương trình mô phỏng. Như đã đề cập, với mục đích giảm thiểu tải tính toán để thời gian chạy chương trình mô phỏng là nhỏ nhất.

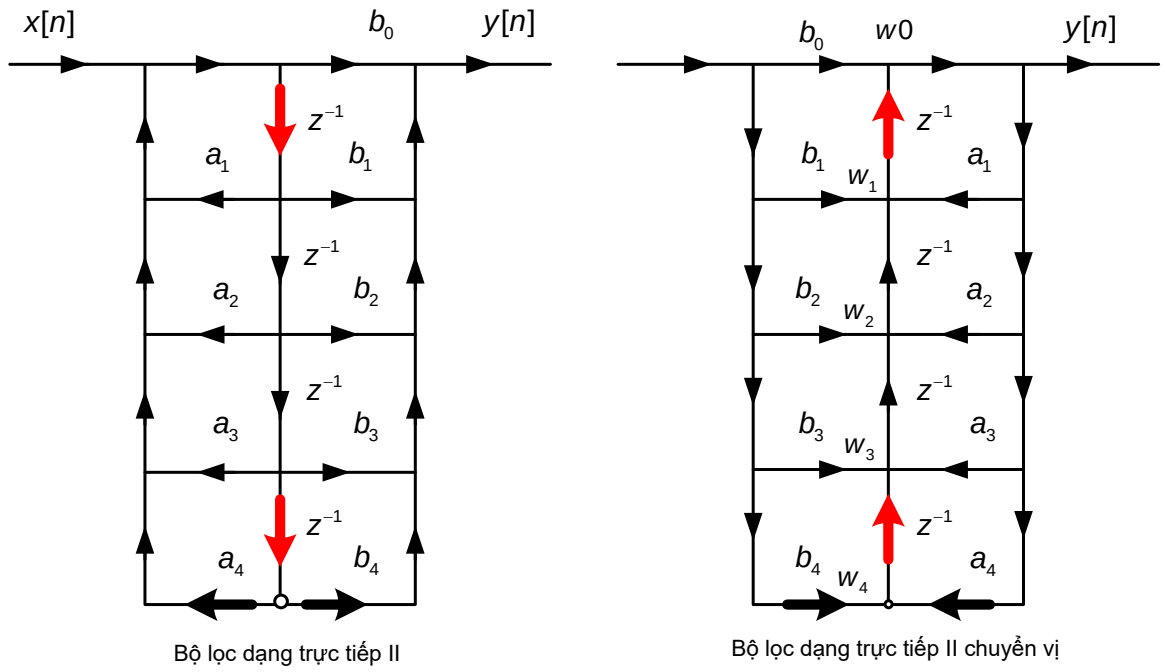
5.3.1. Thực hiện dạng trực tiếp II và dạng trực tiếp II chuyển vị

Kỹ thuật hiệu quả để thực thi bộ lọc số IIR trong mô phỏng là kiến trúc dạng trực tiếp chuyển vị II. Biểu đồ dòng tín hiệu của kiến trúc dạng trực tiếp II chuyển vị và kiến trúc dạng trực tiếp II được minh họa trong hình 5.2. Ta bắt đầu với kiến trúc dạng trực tiếp II vì nó dễ dàng thực thi phương trình sai phân (5.1). Ta nên dành thời gian để kiểm tra cả hai kiến trúc được minh họa trong hình 5.2, chúng đều thỏa mãn công thức (5.1) và (5.3).

Cấu trúc dạng trực tiếp II chuyển vị được dùng phổ biến nhất trong mô phỏng bộ lọc vì tốc độ thực hiện. Dễ dàng rút ra từ cấu trúc dạng trực tiếp II. Nguyên tắc tạo cấu trúc bộ lọc chuyển vị từ một cấu trúc bộ lọc cho trước như sau

1. Kéo lại các đồ hình luồng tín hiệu (dạng trực tiếp II) gốc duy trì kiến trúc (tất cả các liên kết vẫn duy trì các vị trí tương ứng của nó).
2. Đảo hướng luồng tín hiệu trên mỗi liên kết.
3. Gán cho liên kết mới cùng một toán hạng (nhân với hằng số, trễ,...), các toán hạng này đã được gán cho liên kết gốc.
4. Nếu muốn, lật (trái sang phải) đồ hình luồng tín hiệu mới sao cho hướng của luồng tín hiệu vào/ra phù hợp với đồ hình luồng tín hiệu gốc (Lưu ý: luồng tín hiệu thường từ trái qua phải).

Đồ hình luồng tín hiệu mới được gọi là cấu trúc dạng trực tiếp II chuyển vị (DF II) sẽ có cùng hàm truyền đạt với đồ hình luồng tín hiệu gốc.



Hình 5.2: Các cấu trúc thực thi các bộ lọc IIR

Để thấy được tính hấp dẫn của việc thực thi dạng trực tiếp II chuyển vị, ta xét bộ lọc dạng trực tiếp II (DF II) chuyển vị bậc 4 được cho ở hình 5.2 (mở rộng cho các bậc cao hơn cũng tương tự). Ta đưa mẫu tín hiệu vào $x[n]$ để tính tín hiệu ra $y[n]$. Trước tiên là tính biến trạng thái $w_j[n]$ với j = 0, 1,..., 4. Lưu ý rằng, với bộ lọc bậc 4 có 5 biến trạng thái trong công thức. Năm biến trạng thái này được cho bởi:

$$w_0[n] = w_1[n-1] + b_0 x[n] \tag{5.9}$$

$$w_1[n] = -a_1 w_0[n] + w_2[n-1] + b_1 x[n] \tag{5.10}$$

$$w_2[n] = -a_2 w_0[n] + w_3[n-1] + b_2 x[n] \tag{5.11}$$

$$w_3[n] = -a_3 w_0[n] + w_4[n-1] + b_3 x[n] \tag{5.12}$$

$$w_4[n] = -a_4 w_0[n] + b_4 x[n] \tag{5.13}$$

Xét ưu điểm tính toán của kiến trúc chuyển vị được định nghĩa bởi các phương trình trên.

Có thể tính toán các biến trạng thái như được biểu diễn bởi (5.9) đến (5.13) "trong chuỗi". Ví dụ, cho tín hiệu vào $x[n]$ thì tính được $w_0[n]$, vì $w_1[n-1]$ đã được biết từ trước thông qua vòng lặp mô phỏng. Một khi $w_0[n]$ đã biết thì tính được $w_1[n]$. Tiếp tục tính toán, ta thấy rằng $w_j[n]$ chỉ phụ thuộc vào $w_k[n]$ trong đó $k<j$. Vì vậy, quá trình tính toán mỗi biến

trạng thái chỉ yêu cầu biết các đại lượng tính toán trước. Đoạn mã chương trình Matlab thực hiện (5.9) đến (5.13) trong vòng lặp mô phỏng được cho dưới đây.

```
w1 = 0; w2 = 0; w3 = 0 w4 = 0; % Khởi tạo các biến trạng thái
for k = 1:nputs
 ...
w0 = w1 + b0*x
 w1 = -a1*w0 + w2 + b1*x;
 w2 = -a2*w0 + w3 + b2*x;
 w3 = -a3*w0 + w4 + b3*x;
 w4 = -a4*w0 + b4*x;
 y = w0;
 ...
end
```

Trong đoạn mã trên *x* và *y* là tín hiệu vào/ra bộ lọc hiện thời. Lưu ý rằng, các toán hạng trễ một mẫu được minh họa trong (5.9) tới (5.13) được thực hiện theo thứ tự trong đó các biến trạng thái được tính toán. Không cần thiết phải lưu trữ và khôi phục. Kết quả: thuật toán nhanh hơn so với các thuật toán dựa trên các cấu trúc khác, đây là lý do tại sao bộ lọc thường trình (*routine filter*) của Matlab dựa trên cấu DF II chuyển vị. Cũng lưu ý, các biến trạng thái w_1, w_2, w_3, w_4 phải được khởi tạo trước khi đi vào trong vòng lặp mô phỏng lần đầu tiên. Việc khởi tạo này tạo ra một đáp ứng tạm thời đầu ra của bộ lọc. Cụ thể, vòng lặp mô phỏng phải được thực hiện nhiều lần trước khi số liệu hữu hiệu được tập hợp từ quá trình mô phỏng. Thời gian này được coi là "*thời gian định cư*" và gấp vài lần tỉ lệ nghịch băng thông bộ lọc. Dạng véc-tơ của các tính toán trên được đề cập ngắn ngọn (xem ví dụ 5.1).

Các phương trình trạng thái cho bộ lọc số thường được biểu diễn ở dạng ma trận. Dạng của ma trận biến trạng thái rất thuận lợi khi bộ lọc có nhiều đầu vào/ra. Tuy nhiên ở đây, ta chỉ quan tâm đến bộ lọc có một đầu vào *x[n]* và một đầu ra *y[n]*. Biểu thức chung cho bộ lọc 1 đầu vào là:

$$\mathbf{W}[n] = \mathbf{F_c}\mathbf{W}[n] + \mathbf{F_d}\mathbf{W}[n-1] + \mathbf{B}\mathbf{x}[n] \tag{5.14}$$

Trong đó $\mathbf{W}[n]$ và $\mathbf{W}[n-1]$ là các véc-tơ cột $k \times 1$ thể hiện biến trạng thái trước và hiện tại, **F**c và **F**d là các ma trận hệ số và **B** là véc-tơ cột $k \times 1$ ghép đầu vào *x[n]* với các biến trạng thái, phương trình đầu ra đối với một đầu ra là:

$$\mathbf{y}[n] = C\mathbf{W}[n] \tag{5.15}$$

Trong đó **C** là véc-tơ hàng $1 \times k$.

Các phương trình (5.9) tới (5.13) được viết ở dạng ma trận là:

$$\begin{bmatrix} w_0[n] \\ w_1[n] \\ w_2[n] \\ w_3[n] \\ w_4[n] \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ -a_1 & 0 & 0 & 0 & 0 \\ -a_2 & 0 & 0 & 0 & 0 \\ -a_3 & 0 & 0 & 0 & 0 \\ -a_4 & 0 & 0 & 0 & 0 \end{bmatrix} . \begin{bmatrix} w_0[n] \\ w_1[n] \\ w_2[n] \\ w_3[n] \\ w_4[n] \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} w_0[n-1] \\ w_1[n-1] \\ w_2[n-1] \\ w_3[n-1] \\ w_4[n-1] \end{bmatrix} + \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} x[n] \tag{5.16}$$

Cho thấy rằng, có thể tính toán các trạng thái "*theo chuỗi*", vì ma trận $\mathbf{F_c}$ thiết lập quan hệ **W**j[n] và **W**k[n] có giá trị không ở trên và ở bên trên đường chéo chính. Biểu đồ dòng tín hiệu có tính chất này được gọi là biểu đồ khả tính toán. Thuật ngữ *khả tính toán* ở đây không có nghĩa là cần phải có dạng này để tính đầu ra khi cho trước cho một đầu vào. Thấy rõ, vì cấu trúc tam giác phía trên có thể được loại bỏ bằng cách đánh nhãn lại các trạng thái. Hơn nữa, thuật ngữ *khả tính toán* có nghĩa là các trạng thái có thể được tính toán lần lượt như được cho từ (5.9) đến (5.13).

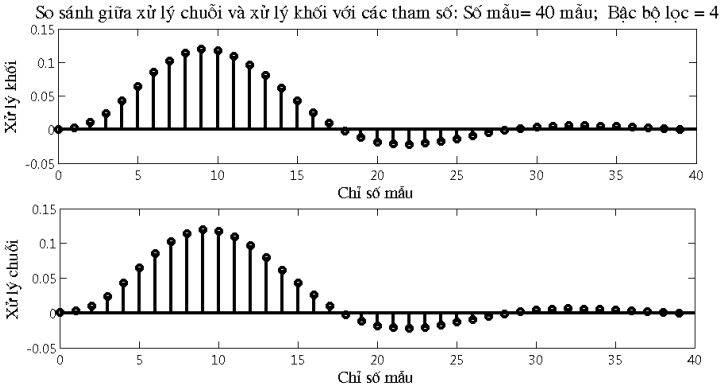
Một ưu điểm của Matlab là dễ dàng tính toán các hệ số bộ lọc a_k và b_k cho rất nhiều nguyên mẫu tương tự khác nhau. Mặc dù, dễ dàng tạo ra các hệ số bộ lọc khi dùng thường trình *filter* trong Matlab, được hướng vào xử lý khối, các bộ lọc IIR có thể được mô phỏng trên cơ sở từng mẫu bằng ba dòng lệnh của Matlab dưới đây:

```
out   =  b(1)*in + sreg(1,1)          % Tính đầu ra bộ lọc
sreg  =  in*b-out*a + sreg;           % Cập nhật các nội dung bộ ghi dịch
sreg  =  [sreg(1,2:(order + 1)),0];   % Bộ ghi dịch dịch vòng
```

Trong đó a và b được xác định ở ngoài vòng lặp mô phỏng. Tham số *areg* trong đoạn mã trình bày thanh ghi dịch có chiều dài là *order+1* trong đó *order* là bậc của bộ lọc. Ưu điểm của giải pháp này là các thường trình tổng hợp bộ lọc Matlab như ***butter*, *cheby1* và *elliptic*,** được dùng để tính toán các véc-tơ hệ số bộ lọc a và b. Phải cẩn thận khi dùng để đảm bảo rằng các véc-tơ chứa các hệ số mẫu số và tử số có cùng chiều dài. Nếu hai véc-tơ có chiều dài khác nhau, thì độn thêm số không vào véc-tơ ngắn hơn.

Ví dụ 5.1: Để minh họa kỹ thuật trên, ta xét chương trình Matlab **NVD5_filterex1.m** được cho ở Phụ lục 5A, chương trình xác định đáp ứng xung kim bộ lọc Butterworth bậc 4 sử dụng cả hai quá trình xử lý khối và chuỗi (từng mẫu một).

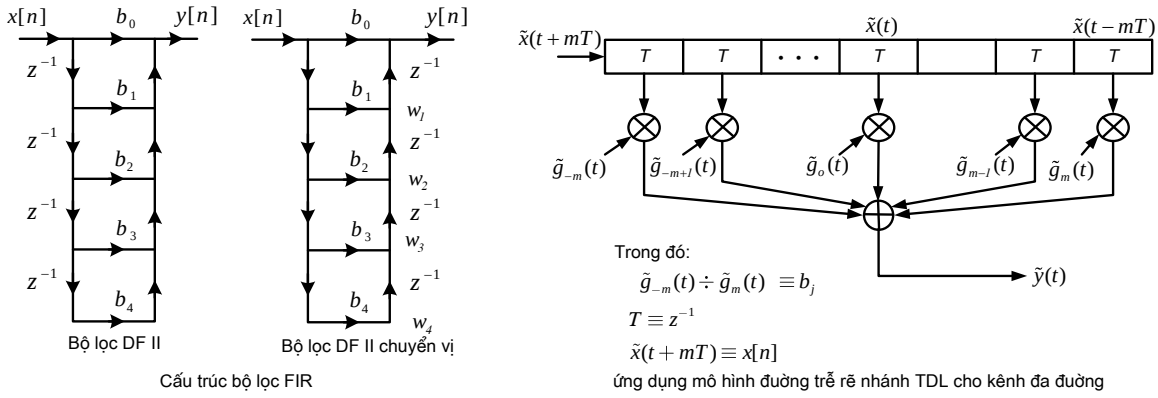
Kết quả chạy chương trình được cho ở hình 5.3. Mô phỏng cho thấy, cả hai kỹ thuật xử lý khối và xử lý nối tiếp đều cho cùng một kết quả.



Hình 5.3: So sánh giữa xử lý chuỗi và xử lý khối

5.3.2. Thực hiện bộ lọc FIR

Chiến lược thực thi các bộ lọc số FIR trực tiếp từ hình 5.2 khi đặt $a_k = 0, k \geq 1$. Hiển nhiên, ta nhận được hai cấu trúc được cho ở hình 5.4. Chúng được coi là cấu trúc đường trễ rẽ nhánh TDL bởi lẽ chúng được thực thi là các dây trễ (các phần tử trễ z-1 được nối tầng) nhân với các trọng số b_k.



Hình 5.4: Cấu trúc bộ lọc FIR và ứng dụng điển hình

5.4. Bộ lọc IIR - Kỹ thuật tổng hợp và đặc điểm

Bộ lọc đáp ứng xung kim vô hạn IIR thường được thiết kế từ nguyên mẫu tương tự (Butterworth, Chebyshev, Elliptic...), và được thực thi bằng các cấu trúc đệ quy. Hệ thống viễn thông hiện đại không dựa vào các nguyên mẫu tương tự như trước đây, ngoại trừ các tầng RF của hệ thống. Nhiều hệ thống hiện đại như: hệ thống vô tuyến mềm dùng kỹ thuật DSP để xây dựng các khối chức năng (bao gồm các bộ lọc). Trong các hệ thống dựa trên DSP, bài toán triển khai số hóa cho các nguyên mẫu tương tự là không đáng bàn vì các bộ lọc được dùng trong phần cứng vật lý đã thực sự ở dạng số. Kết quả là, kỹ thuật kinh điển để tổng hợp bộ lọc số dùng nguyên mẫu tương tự làm điểm bắt đầu đang mờ nhạt. Tuy nhiên, các thiết kế từ các

nguyên mẫu tương tự vẫn được dùng trong nhiều ứng dụng. Một số kỹ thuật thiết kế được hỗ trợ bởi máy tính đã được triển khai cho bộ lọc IIR, một trong số đó được đề cập ở đây. Vì mục đích minh họa các kỹ thuật cơ bản và các lỗi mô phỏng khi dùng các kỹ thuật này, nên ta chỉ xét các ứng dụng rất đơn giản của các kỹ thuật này.

5.4.1. Bộ lọc bất biến xung kim

Giả sử đáp ứng xung *đơn vị* bộ lọc số là đáp ứng xung kim của bộ lọc tương tự được lấy mẫu (bỏ qua tỷ số biên độ). Khi này, bộ lọc số được xem là thực hiện bất biến xung kim của bộ lọc nguyên mẫu tương tự. Nói cách khác, hàm truyền đạt của bộ lọc số $H(z)$ được định nghĩa bởi:

$$H(z) = T\mathcal{Z}\left\{\mathcal{L}^{-1}\left[H_a(s)\right]\Big|_{t=nT}\right\} \tag{5.17}$$

Trong đó $\mathcal{L}$-1(.) là biến đổi Laplace ngược, $\mathcal{Z}(.)$ là biến đổi Z, $H_a(s)$ là hàm truyền đạt của bộ lọc tương tự và T là chu kỳ lấy mẫu (là tỷ lệ biên độ).

Để minh họa kỹ thuật bất biến xung kim ta xét nguyên mẫu tương tự đơn giản nhất là:

$$H_a(s) = \frac{a}{s+a} \tag{5.18}$$

Lưu ý rằng: Đây là bộ lọc thông thấp bậc nhất có thành phần một chiều dc (f = 0) là 1.

Ví dụ 5.2: Từ (5.17) trước hết biến đổi Laplace ngược của $H_a(s)$ ta được đáp ứng xung kim:

$$\mathcal{L}^{-1}\left[H_a(s)\right] = h_a(t) = a.e^{-at}.u(t) \tag{5.19}$$

Trong đó $u(t)$ là hàm bước nhảy đơn vị, được dùng để *biểu diễn tính nhân quả của bộ lọc*. Trước khi biến đổi z đáp ứng xung kim phải được lấy mẫu tại các điểm lấy mẫu $t = nT$. Ta được:

$$\mathcal{L}^{-1}\left[H_a(s)\right]\Big|_{t=nT} = h_a[n] = a.e^{-anT}\,u[n] \tag{5.20}$$

Biến đổi z của $h_a[n]$ là:

$$\mathcal{Z}\left\{\underbrace{\mathcal{L}^{-1}\left[H_a(s)\right]\Big|_{t=nT}}_{h_a[n]}\right\} = \sum_{n=0}^{\infty} a\left[e^{-aT}\right]^n z^{-n} \tag{5.21}$$

Lấy tổng và nhân với chu kỳ lấy mẫu ta có:

$$H(z) = \frac{aT}{1 - e^{-aT}\,z^{-1}} \tag{5.22}$$

Là tương đương bất biến xung kim của (5.18). Tính cách trong miền tần số của bộ lọc này sẽ được khai thác trong ví dụ 5.5.

5.4.2. Bộ lọc bất biến bước

Bộ lọc bất biến bước thường được dùng trong mô phỏng (lý do được nêu rõ ở phần sau). Với bộ lọc bất biến bước, *đáp ứng xung đơn vị* của bộ lọc số là đáp ứng bước của bộ lọc tương tự được lấy mẫu. Vì đáp ứng bước của bộ lọc tương tự được biểu diễn trong miền tần số là $H_a(s)/s$, nên hàm truyền đạt của bộ lọc bất biến bước là:

$$\underbrace{\frac{1}{1-z^{-1}}H(z)}_{\substack{\text{Đáp ứng của bộ lọc số có}\\ \text{hàm truyền đạt H(z) đối với}\\ \text{bước đơn vị được lấy mẫu}}} = \mathcal{Z}\left\{\underbrace{\underbrace{\mathcal{L}^{-1}\left[\frac{1}{s}H_a(s)\right]}_{\substack{\text{Đáp ứng bước đơn vị của}\\ \text{bộ lọc nguyên mẫu tương tự}}}\Big|_{t=nT}}_{\substack{\text{Đáp ứng bước đơn vị của bộ lọc}\\ \text{nguyên mẫu tương tự được lấy mẫu}}}\right\} \tag{5.23}$$

Lưu ý rằng, vế bên trái của (5.23) là đáp ứng của bộ lọc số có hàm truyền đạt *H(z)* đối với bước đơn vị được lấy mẫu. Vế phải của (5.23) biểu diễn biến đổi z của đáp ứng bước đơn vị của bộ lọc nguyên mẫu tương tự được lấy mẫu.

Ví dụ 5.3: Trong ví dụ này ta lại sử dụng (5.18) làm nguyên mẫu tương tự. Đáp ứng bước của bộ lọc tương tự là:

$$\frac{1}{s}H_a(s) = \frac{a}{s(s+a)} \tag{5.24}$$

Khai triển ta có:

$$\frac{1}{s}H_a(s) = \frac{1}{s} - \frac{1}{s+a} \tag{5.25}$$

Lấy biến đổi Laplace ngược và lấy mẫu tại $t = nT$ ta có:

$$\mathcal{L}^{-1}\left[\frac{1}{s}H_a(s)\right]\Big|_{t=nT} = u[n] - e^{-anT}.u[n] \tag{5.26}$$

Và lấy biến đổi z ta có:

$$\frac{1}{1-z^{-1}}H(z) = \sum_{n=0}^{\infty}\left\{1-\left(e^{-aT}\right)^n\right\}z^{-n} \tag{5.27}$$

$$\frac{1}{1-z^{-1}}H(z) = \frac{1}{1-z^{-1}} - \frac{1}{1-e^{-aT}z^{-1}} \tag{5.28}$$

Hàm truyền đạt của bộ lọc số bất biến bước là:

$$H(z) = 1 - \frac{1-z^{-1}}{1-e^{-aT}z^{-1}} \tag{5.29}$$

Viết lại phương trình trên ở dạng chuẩn như được định nghĩa bởi (5.3) ta có:

$$H(z) = \frac{\left[1 - e^{-aT}\right] z^{-1}}{1 - e^{-aT} z^{-1}} \tag{5.30}$$

Các tính cách của bộ lọc này trong miền tần số sẽ được khai thác trong ví dụ 5.5. Kết quả sẽ được so sánh với kết quả bất biến xung kim.

5.4.3. Bộ lọc biến đổi z song tuyến tính

Bộ lọc số biến đổi z song tuyến có lẽ được sử dụng nhiều nhất trong mô phỏng bởi lẽ: *(i)* việc tổng hợp bộ lọc sử dụng kỹ thuật biến đổi z song tuyến là rất dễ thực hiện và thuần túy số học; *(ii)* bộ lọc biến đổi z song tuyến không tỏ ra chồng phổ.

Kỹ thuật tổng hợp

Kỹ thuật tổng hợp biến đổi z song tuyến ánh xạ nguyên mẫu tương tự $H_a(s)$ thành bộ lọc số bằng biến đổi số học đơn giản. Cụ thể, hàm truyền đạt của bộ lọc số, $H(z)$, được định nghĩa bởi:

$$\underset{\substack{\text{Hàm truyền đạt}\\ \text{bộ lọc số}}}{H(z)} = \underset{\substack{\text{Hàm truyền đạt}\\ \text{nguyên mẫu tương tự}}}{H_a(s)}\Bigg|_{s=C(1-z^{-1})/(1+z^{-1})} \tag{5.31}$$

Trong đó C là hằng số. Kỹ thuật để xác định C sẽ được đề cập ngắn gọn sau. Vì $H_a(s)$ được định nghĩa là tỷ số của các đa thức trong miền s nên ánh xạ số học được định nghĩa bởi (5.31) cho ta $H(z)$ ở dạng tỷ số các đa thức theo z^{-1}. Thực thi bộ lọc cũng như các phần khác đã được đề cập.

Để chứng minh hàm truyền đạt của bộ lọc số có các tính chất mong muốn, ta thay $e^{s_d T}$ (s_d là biến tần số phức đối với bộ lọc số) cho z trong định nghĩa của biến đổi z song tuyến. Ta có:

$$s = \frac{1 - e^{-s_d T}}{1 + e^{-s_d T}} \tag{5.32}$$

Thế $s = j2\pi f_a$ và $s_d = j2\pi f_d$ vào (5.32) ta có:

$$j2\pi f_a = C \frac{e^{j\pi f_d T} - e^{-j\pi f_d T}}{e^{j\pi f_d T} + e^{-j\pi f_d T}} \tag{5.33}$$

Tương đương:

$$2\pi f_a = C \tan\left(\pi f_d T\right) \tag{5.34}$$

$$C = 2\pi f_a \cot\left(\pi f_d T\right) \tag{5.35}$$

Xác định quan hệ giữa đáp ứng tần số của bộ lọc tương tự nguyên mẫu và bộ lọc số. Điều chỉnh *C* theo cách này sao cho $f_c = f_d$ với tần số lấy mẫu bất kỳ được gọi là làm méo tần trước *(prewarning)*.

Thấy rõ từ các phương trình trên, quan hệ chuyển đổi giữa f_a và f_d là phi tuyến. Trong môi trường mô phỏng thường mong muốn duy trì dạng của hàm truyền đạt cả về biên độ và pha khi ánh xạ bộ lọc tương tự vào bộ lọc số. Việc duy trì dạng hàm truyền đạt cần có quan hệ chuyển đổi giữa f_d và f_a là tuyến tính trong một khoảng càng rộng càng tốt. Vì $tan(x) \approx x$ khi x nhỏ, biến đổi xấp xỉ tuyến tính yêu cầu $\pi f_d T \le 1$, hay:

$$f_d \ll \frac{1}{\pi} f_s \tag{5.36}$$

Trong đó f_d là giải tần số xét và f_s là tần số lấy mẫu. Giá trị C sẽ làm cho dễ dàng xác định biến đổi xấp xỉ tuyến tính với $f_d \approx f_a$. Tính tuyến tính yêu cầu:

$$2\pi f_a \approx C(\pi f_d T) \tag{5.37}$$

Nếu ta cần $f_d \approx f_a$ ta có:

$$C = \frac{2}{T} = 2 f_s \tag{5.38}$$

Tuy nhiên, việc duy trì $f_d \approx f_a$ với mọi tần số xét như khoảng tần số của dải thông bộ lọc yêu cầu tần số lấy mẫu quá lớn. Dẫn đến thời gian mô phỏng không thực tế.

Ví dụ 5.4: Để minh họa phương pháp tổng hợp biến đổi z song tuyến ta trở lại nguyên mẫu tương tự bậc nhất đơn giản được xác định bởi (5.18). Theo định nghĩa, bộ lọc biến đổi z song tuyến có hàm truyền đạt như sau:

$$H(z) = \left. \frac{a}{s + a} \right|_{s = C\frac{1 - z^{-1}}{1 + z^{-1}}} \tag{5.39}$$

Dẫn đến:

$$H(z) = \frac{a(1 + z^{-1})}{C(1 - z^{-1}) + a(1 + z^{-1})} \tag{5.40}$$

Ở dạng tiêu chuẩn là:

$$H(z) = \frac{\frac{a}{C + a} + \frac{a}{C + a} z^{-1}}{1 - \frac{C - a}{C + a} z^{-1}} \tag{5.41}$$

Tính cách miền tần số của bộ lọc sẽ được khai thác trong ví dụ 5.5.

Ví dụ 5.5: Ví dụ này sẽ so sánh ba bộ lọc số trong ba ví dụ trên. Giả sử tần số lấy mẫu là 100 Hz (T = 0,01), tham số a là $2\pi(10)$ do vậy tần số 3 dB của bộ lọc tương tự là 10 Hz. Với các giá trị giả định này ta có:

$$aT = 0,628319 \tag{5.42}$$

$$e^{-aT} = 0,533488 \tag{5.43}$$

Theo đó, ta có các hàm truyền đạt cho ba loại bộ lọc số: *(i)* bất biến xung kim; *(ii)* bất biến bước; *(iii)* biến đổi z song tuyến tính (hai phía) nghĩa là:

Hàm truyền đạt của bộ lọc bất biến xung kim có được từ (5.22):

$$H_{ii}(z) = \frac{0,628319}{1 - 0,533488z^{-1}} \tag{5.44}$$

Hàm truyền đạt của bộ lọc bất biến bước có được từ (5.30):

$$H_{si}(z) = \frac{(1 - 0,533488)z^{-1}}{1 - 0,533488z^{-1}} \tag{5.45}$$

Hàm truyền đạt của bộ lọc biến đổi z hai phía được xác định như sau: Hai phiên bản của bộ lọc biến đổi z song tuyến sẽ được trình bày.

Trước hết, nếu không làm méo tần trước thì đáp ứng biên độ của nguyên mẫu tương tự và bộ lọc số sẽ giống nhau ở các tần số thấp:

$$C = \frac{2}{T} = 200 \tag{5.46}$$

Thế giá trị *C* và a vào (5.41) ta có:

$$H_{bl}(z) = \frac{0,239057 + 0,239057z^{-1}}{1 - 0,5218861z^{-1}} \tag{5.47}$$

Tại đây, giả thiết là giá trị C được lựa chọn sao cho đáp ứng tần số được phù hợp tại tần số 3 dB của bộ lọc tương tự. Khi đó:

$$f_a = f_d = \frac{a}{2\pi} \tag{5.48}$$

Vì vậy, từ (5.35) giá trị *C* là:

$$C = a\cot\left(\frac{aT}{2}\right) \tag{5.49}$$

Tương đương:

$$C = 40\pi\cot(0,2\pi) = 172,961 \tag{5.50}$$

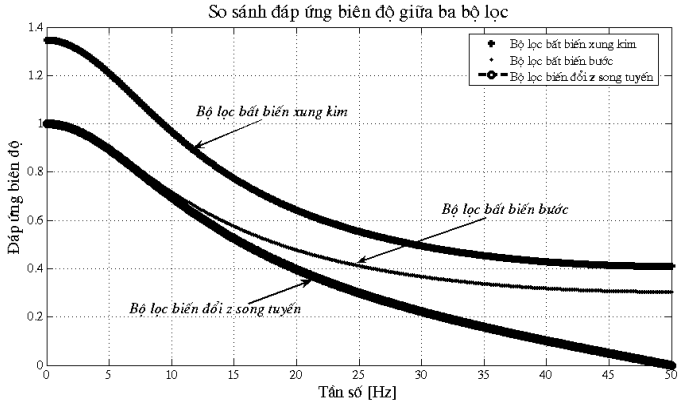
Sử dụng giá trị C trong (5.41) nhận được:

$$H_{bl}(z) = \frac{0,420808 + 0,420808z^{-1}}{1 - 0,158384z^{-1}} \tag{5.51}$$

Ta nên so sánh đáp ứng được định nghĩa bởi (5.47) và (5.51).

Mã chương trình Matlab **NVD5_threefilters.m** được cho ở Phụ lục 5A tạo đáp ứng biên độ của các bộ lọc bất biến xung kim, bất biến bước, biến đổi z song tuyến được mô tả bởi (5.47).

Kết quả được minh họa trong hình 5.5. Cần lưu ý: *(i)* bộ lọc bất biến hàm bước và bộ lọc biến đổi z song tuyến đều có cùng độ lợi bằng 1 tại dc. Độ lợi dc của bộ lọc bất biến xung lớn hơn một do chồng phổ; *(ii)* bộ lọc biến đổi z song tuyến có giá trị không tại tần số Nyquist ($f_s/2$) bởi lẽ nguyên mẫu tương tự có giá trị không tại tần số $f = \infty$ và giá trị không này được ánh xạ vào tần số Nyquist.

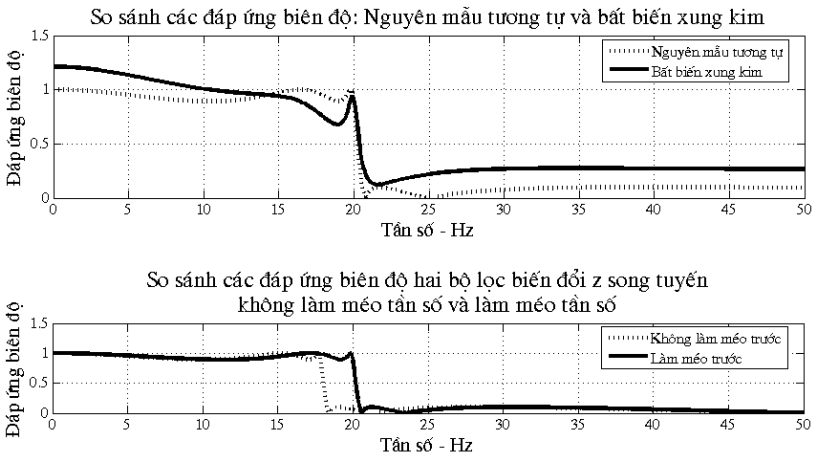


Hình 5.5: So sánh các đáp ứng biên độ

Ví dụ 5.6: Bộ lọc được triển khai trong ví dụ 5.5 dựa vào nguyên mẫu bậc một rất đơn giản và các hệ số bộ lọc: giá trị a_k và b_k được xác định không cần máy tính. Trong ví dụ này ta xét bộ lọc elip bậc 5 có độ nhấp nhô dải thông là 1 dB và suy hao nhỏ nhất của chắn dải là 20 dB. Mặc dù có thể dễ dàng xác định hệ số bộ lọc bằng giải tích ở ví dụ 5.5 nhưng do tính phức tạp của hàm truyền đạt nguyên mẫu tương tự cần nỗ lực đáng kể.

Tuy nhiên Matlab hỗ trợ nhiều kỹ thuật tổng hợp ở dạng hộp công cụ Toolbox tín hiệu và hệ thống. Sẵn có các thường trình để thiết kế bộ lọc số biến đổi z song tuyến và bất biến xung kim. Mã chương trình Matlab tổng hợp bộ lọc elip bậc 5 được cho ở file **NVD5_ellipexam.m** có trong Phụ lục 5A thực hiện bài toán này.

Lưu ý chạy chương trình **NVD5_ellipexam.m** sẽ tạo ra 4 đáp ứng tần số được minh họa ở hình 5.6 trong đó: *(i)* tạo ra đáp ứng biên độ tần số của nguyên mẫu tương tự để so sánh với đáp ứng biên độ tần số của bộ lọc số bất biến xung kim; *(ii)* tạo hai bộ lọc số biến đổi z song tuyến: một có méo tần số và một không méo tần số để minh họa ảnh hưởng của méo tần số.



Hình 5.6: So sánh các đáp ứng biên độ

Để được tường minh, ta so sánh đáp ứng biên độ của nguyên mẫu tương tự, bộ lọc số bất biến xung kim và hai bộ lọc biến đổi z song tuyến được cho ở hình 5.6. Theo đó, hình 5.6(a) so sánh đáp ứng biên độ của bộ lọc số bất biến xung kim với nguyên mẫu tương tự. Lưu ý là ngoài băng thông việc so sánh không được rõ rằng do lỗi chồng phổ, khá rõ trong các bộ lọc elip vì đáp ứng biên độ của nguyên mẫu tương tự không bị giới hạn băng do tính cách dải chắn bộ lọc. Điều này có thể thấy rõ trong đáp ứng biên độ được nâng lên trong cả dải thông và dải chắn của bộ lọc.

Hình 5.6(b) minh họa phương pháp tổng hợp biến đổi z song tuyến tốt hơn nhiều. Nếu méo tần không được dùng thì tần số cắt bị giảm từ 20 Hz đến xấp xỉ 18 Hz. Khi làm méo trước tần số thì tần số cắt của bộ lọc số đạt tới tần số cắt của nguyên mẫu tương tự dẫn đến đáp ứng biên độ gần với đáp ứng biên độ của nguyên mẫu trong hầu hết dải tần. Tuy nhiên, có một điểm khác biệt. Tại tần số Nyquist ($f_s/2 = 50Hz$) đáp ứng tần số của nguyên mẫu là 0,1 (tương ứng với suy hao 20 dB) trong khi đó đáp ứng biên độ của bộ lọc số biến đổi z song tuyến là 0 (tương ứng suy hao là ∞ dB). Bộ lọc tương tự có giá trị không tại tần số $f = \infty$ còn với biến đổi z song tuyến điểm không này được ánh xạ vào tần số Nyquist.

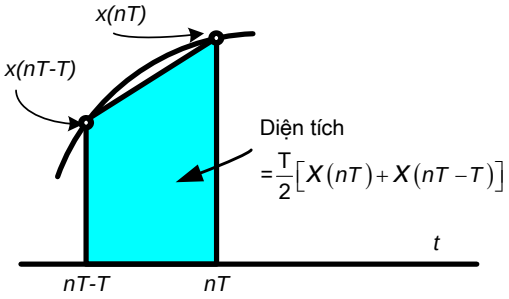
Trường hợp đặc biệt: Tích phân hình thang

Thuật toán tích phân quan trọng thường được dùng trong mô phỏng là thuật toán tích phân hình thang. Tổng quát, tích phân thời gian rời rạc được cho bởi biểu thức:

$$y[n] = y[n-1] + \Delta(n-1,n) \tag{5.52}$$

$y[n]$ và $y[n-1]$ là đầu ra bộ tích phân tại chỉ số n và n - 1, và $\Delta(n-1,n)$ là diện tích gia lượng cộng thêm vào tích phân theo gia số thời gian $(n-1)T < t \le nT$. Diện tích gia lượng được minh họa ở hình 5.7 và được cho bởi:

$$\Delta\big(n-1,n\big) = \frac{T}{2}\big(x[n] + x[n-1]\big) \tag{5.53}$$



Hình 5.7: Tích phân hình thang

Theo đó:

$$y[n]-y[n-1]=\frac{T}{2}\left(x[n]+x[n-1]\right) \tag{5.54}$$

Lấy biến đổi z hai phía (5.54) nhận được hàm truyền đạt của tích phân hình thang. Kết quả là:

$$H(z)=\frac{Y(z)}{X(z)}=\frac{T}{2}\times\frac{1+z^{-1}}{1-z^{-1}} \tag{5.55}$$

Nếu ta thiết kế thuật toán tích phân dựa trên kỹ thuật tổng hợp biến bổi z song tuyến, kết quả là:

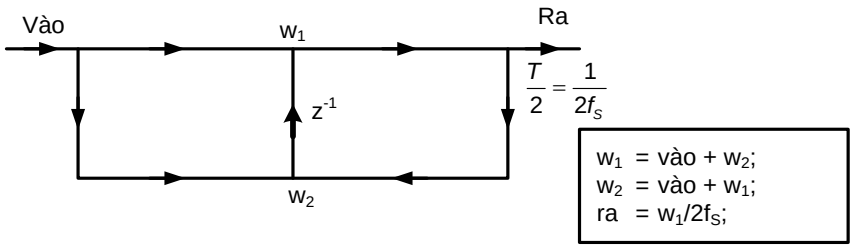
$$H(z)=H_a(s)\Big|_{s=\frac{2}{T}\times\frac{1-z^{-1}}{1+z^{-1}}} \tag{5.56}$$

Vì tích phân trong miền thời gian là tương đương với chia cho s trong miền s nên:

$$H(z)=\frac{1}{s}\Big|_{s=\frac{2}{T}\times\frac{1-z^{-1}}{1+z^{-1}}}=\frac{T}{2}\times\frac{1+z^{-1}}{1-z^{-1}} \tag{5.57}$$

Cũng giống như tích phân hình thang được biểu diễn bởi (5.55)

Đồ hình tín hiệu tương ứng để thực hiện DF II chuyển vị của bộ tích phân hình thang được minh họa ở hình 5.8. Lưu ý rằng, ngoại trừ lấy tỷ số biên độ là $T/2$ tất cả là đơn vị. Tích phân hình thang chỉ là một trong rất nhiều thuật toán được sử dụng trong các chương trình mô phỏng.



Hình 5.8: Đồ hình dòng tín hiệu cho bộ tích phân hình thang và mã Matlab

5.4.4. Thiết kế bộ lọc số IIR với sự hỗ trợ của máy tính

Vài thập niên gần đây đã triển khai nhiều kỹ thuật thiết kế bộ lọc số FIR và IIR với sự hỗ trợ của máy tính. Kỹ thuật FIR nhận được nhiều ứng dụng hơn kỹ thuật IIR nhưng kỹ thuật IIR đôi khi được sử dụng do cấu trúc nhỏ gọn (ít thành phần trễ hơn). Hai ưu điểm nổi trội thiết kế với sự hỗ trợ bởi máy tính là: *(i)* có thể thiết kế các bộ lọc số mong muốn mà không cần một nguyên mẫu tương tự; *(ii)* cho phép đạt được dung hòa, chẳng hạn giữa mức độ phức tạp (số lượng hệ số bộ lọc) và mức độ chính xác (lỗi giữa đáp ứng biên độ mong muốn và thực tế).

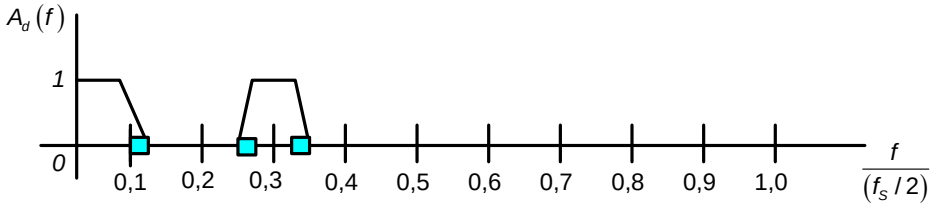
Kỹ thuật phổ biến để triển khai bộ lọc IIR là điều chỉnh hệ số bộ lọc cho đến khi đạt được đáp ứng mong muốn, phổ cập nhất là xác định đặc điểm bộ lọc mong muốn, xác định đáp ứng biên độ và bậc N của bộ lọc. Dẫn đến bài toán tối ưu bị hạn chế do bậc N là giới hạn.

Ví dụ 5.7: Ta triển khai bộ lọc định kênh để có được các tính chất như trong hình 5.9. Lưu ý: *(i)* bộ lọc cho hai kênh qua nó, một kênh được dịch về dc; *(ii)* tần số trong hình 5.9 được chuẩn hóa theo tần số Nyquist (một nửa tần số lấy mẫu). Dưới dạng tần số chuẩn hóa này $f_N = f/(f_s/2)$, đáp ứng biên độ của bộ lọc được định nghĩa là:

$$A_d(f) = \begin{cases} 1, & |f| < 0{,}1 \quad 0{,}25 < f < 0{,}35 \\ 0, & 0{,}12 < f < 0{,}23 \quad 0{,}37 < f < 1 \end{cases} \tag{5.58}$$

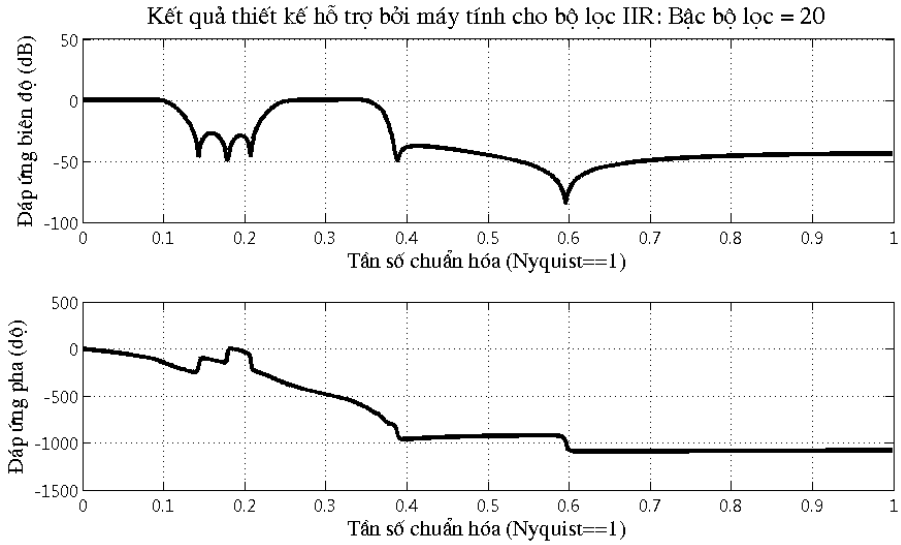
Ba băng tần không có trong biểu thức của $A_d(f)$ là các băng tần dịch, được ký hiệu vùng đen trong hình 5.9.

Mã chương trình Matlab để thiết kế bộ lọc này được cho ở file **NVD5_yw.m** trong Phụ lục 5A.



Hình 5.9: Đáp ứng biên độ mong muốn

Lưu ý rằng, nếu bộ lọc có bậc 20 được tạo ra thì cả tử số và mẫu số của đa thức *H(z)* là đa thức bậc 20 của z^{-1}. Chạy chương trình mô phỏng Matlab kết quả được cho ở hình 5.10, suy hao dải chắn là 30 dB hoặc tốt hơn. Dải thông được định dạng khá hợp lý và đáp ứng pha xấp xỉ tuyến tính trong dải thông. Tăng bậc bộ lọc sẽ cải thiện lỗi xấp xỉ nhưng lại tăng tính phức tạp hay tăng thời gian mô phỏng.



Hình 5.10: Kết quả thiết kế IIR với sự hỗ trợ của máy tính

5.4.5. Nguồn lỗi trong bộ lọc IIR

Thấy rõ, nguồn lỗi phụ thuộc vào phương pháp tổng hợp được dùng. Đa phần các lỗi này được giảm thiểu bằng cách sử dụng tần số lấy mẫu rất lớn trong mô phỏng nhưng thời gian mô phỏng lại quá lâu đôi khi không thực sự cần thiết. Theo đó, việc chọn tần số lấy mẫu phải cân nhắc giữa tính chính xác và thời gian mô phỏng. Nguồn lỗi được tóm tắt trong bảng 5.1.

Bảng 5.1: Các nguồn lỗi bộ lọc số IIR

Kỹ thuật tổng hợp	Nguồn lỗi	Giảm thiểu lỗi
Bất biến xung kim	Chồng phổ	Chọn tần số lấy mẫu cao hơn
Bất biến bước	Chồng phổ	Chọn tần số lấy mẫu cao hơn
Biến đổi Z song tuyến (2 phía)	Méo tần số	Chọn một tần số lấy mẫu cao hơn nhiều tần số tới hạn cao nhất
Phương pháp CAD	Lỗi lấy xấp xỉ	Tăng bậc của bộ lọc hoặc dùng phương pháp tổng hợp khác phù hợp với ứng dụng hơn

5.5. Bộ lọc FIR - Kỹ thuật tổng hợp và đặc điểm

Nếu như đáp ứng xung kim của bộ lọc $h[n]$ là hữu hạn, hay nó được lấy hữu hạn bằng cách cắt bớt đáp ứng xung kim vô hạn thì đầu ra của bộ lọc trong miền thời gian rời rạc được cho bởi:

$$y(nT) = T\sum_{k=0}^{N} h(kT).x\{(n-k)T\} \tag{5.59}$$

Được viết theo DSP chuẩn là:

$$y[n] = \sum_{k=0}^{N} h[n]x[n-k] = \sum_{k=0}^{N} b_k x[n-k] \tag{5.60}$$

Hoặc ở dạng biến đổi z là:

$$Y(z) = \sum_{k=0}^{N} b_k z^{-k} X(z) \tag{5.61}$$

Phương trình (5.60) định nghĩa bộ lọc số FIR, là phiên bản rời rạc theo thời gian của tích chập đối với bộ lọc tương tự. Lưu ý rằng, (5.60) và (5.61) giống với mô hình bộ lọc được định nghĩa bởi (5.1) và (5.3) với $a_k = 0$ khi $k \geq 1$. Bộ lọc là FIR bởi lẽ trong $h[n]$ hầu như có $N + 1$ thành phần khác không. Vì $a_k = 0$ khi $k \geq 1$ nên bộ lọc FIR không có đường hồi tiếp. Phép toán tổng chập trong miền thời gian được định nghĩa trong (5.60), được mô phỏng bằng hàm *filter* trong Matlab. Nếu ta nghiên cứu hàm *filter* bằng lệnh *doc filter* từ cửa sổ làm việc của Matlab, khi này ta sẽ nhận được trình bày ngắn gọn và súc tích về định nghĩa hàm cũng như cách sử dụng và *rất dễ hiểu* thông qua các minh họa ở đó.

Bộ lọc FIR là rất hấp dẫn vì các lý do cơ bản sau

1. Không phải tất cả các bộ lọc trong viễn thông đều có thể được biểu diễn dưới dạng hàm truyền đạt trong miền biến đổi Laplace do vậy kỹ thuật IIR được mô tả trong phần trước là không thể trực tiếp áp dụng được. Hai bộ lọc quan trọng thuộc loại này là bộ lọc định dạng xung SQRC, và bộ lọc Doppler Jakes. Dễ dàng mô phỏng chúng bằng giải pháp FIR.

2. Trong nhiều mô phỏng, số liệu bộ lọc được cho ở dạng đáp ứng tần số đo đạc hoặc số liệu của đáp ứng xung kim. Rất dễ thực hiện mô phỏng các bộ lọc này bằng giải pháp FIR. Trong khi sẵn có một số kỹ thuật để làm phù hợp mô hình ARMA với số liệu đáp ứng tần số thì việc lấy xấp xỉ như vậy không mang lại kết quả mong muốn khi số liệu đáp ứng tần số là không liên tục.

3. Bằng giải pháp FIR, ta có thể xác định đáp ứng pha và biên độ tùy ý, và chúng có thể độc lập với nhau.

4. Do bộ lọc FIR không có hồi tiếp nên luôn luôn ổn định.

Mô hình mô phỏng FIR có một nhược điểm lớn là hiệu quả tính toán không bằng bộ lọc IIR. Mỗi mẫu được tạo ra theo (5.60) cần phải có N phép nhân và phép cộng. Nếu đáp ứng xung rất dài (N > 1024 điểm), thì thuật toán FIR sẽ chạy lâu hơn rất nhiều so với các thuật toán IIR thông thường. Với bộ lọc IIR số lượng các phép cộng và nhân để tạo ra một mẫu được xác định bởi bậc của bộ lọc chứ không phải là chiều dài của đáp ứng xung kim như trường hợp FIR.

Hiệu quả tính toán của mô hình FIR có thể được cải thiện bằng việc dùng DFT/FFT để tính tổng chập. Khi sử dụng DFT/FFT cần đặc biệt các lưu ý sau:

1. DFT/FFT là tuần hoàn và tạo các đầu ra chập một cách tuần hoàn hoặc vòng. Để có được các đầu ra chập tuyến tính thì phải độn các số 0 vào chuỗi đầu vào và chuỗi đáp ứng xung kim. Để có được hiệu quả tính toán cao nhất thì chiều dài của các véc tơ đã được độn phải là lũy thừa của 2 do đó tận dụng thuật toán FFT cơ số 2.

2. Nếu chuỗi đầu vào quá dài thì trước hết phân thành các khối nhỏ không chồng lấn, sau đó thực hiện tích chập cho mỗi khối đầu vào để tìm khối đầu ra tương ứng, cuối cùng xếp chồng lần lượt và cộng để tạo ra chuỗi đầu ra. Nếu khối đầu vào ngắn (nhỏ hơn 216 mẫu) thì khối đầu ra sẽ được tính toán trong một lần. Phương pháp xếp chồng và cộng của thuật toán DFT/FFT dựa trên tổng chập được thực hiện bằng hàm Matlab có sẵn **fftfit**. Ta có thể tìm hiểu hàm **fftfit** bằng lệnh **doc fftfit.**

3. Tổng chập dựa trên DFT/FFT là một quá trình xử lý khối, mẫu đầu tiên của khối đầu ra chỉ được tạo ra sau khi tích lũy đủ số mẫu đầu vào để thực hiện DFT/FFT. Do trễ này mà bộ lọc DFT/FFT thường không được sử dụng để mô phỏng bộ lọc (mà bộ lọc là một phần của vòng lặp hồi tiếp).

Thấy rõ, hiệu quả tính toán tổng chập trong miền thời gian dựa trên DFT là tỉ lệ với $\frac{log_2 N}{N}$. Nếu chiều dài của đáp ứng xung kim nhỏ hơn 128 mẫu, sẽ không có nhiều khác biệt về khối lượng tính toán giữa tổng chập trong miền thời gian và thực hiện bằng DFT/FFT.

Hiệu quả tính toán của phương pháp FIR (trong miền thời gian và DFT/FFT dựa trên tích chập) được cải thiện nếu chiều dài của đáp ứng xung kim được rút ngắn. Việc cắt giảm cũng tương đương với việc nhân đáp ứng xung kim $h[k]$, k = 0, 1, 2, 3,... với một hàm cửa sổ $w[k]$ có chu kỳ là N mẫu. Giá trị N được chọn sao cho ít nhất 98% năng lượng của $h[n]$ được chứa trong cửa sổ. Nói cách khác:

$$\sum_{k=0}^{N} |h[k]|^2 = 0{,}98 \sum_{k=0}^{\infty} |h[k]|^2 \tag{5.62}$$

Hàm của sổ đơn giản nhất là hàm cửa sổ chữ nhật nó sẽ loại bỏ các mẫu đáp ứng xung kim ứng với k > N. Nói cách khác:

$$w[k] = \begin{cases} 1, & 0 \le k \le N \\ 0, & \text{nếu khác} \end{cases} \tag{5.63}$$

Tồn tại rất nhiều hàm cửa sổ khác được sử dụng.

Việc cắt ngắn (cửa sổ hóa) trong miền thời gian tương đương với tổng chập trong miền tần số, nghĩa là $H_T(f) = H(f) \otimes W(f)$ trong đó $\otimes$ là tổng chập. Một cách lý tưởng hàm cửa sổ trong miền tần số là $W(f) = \delta(f)$ vì khi lấy tích chập với một xung kim sẽ không làm thay đổi $H(f)$. *Một xung kim trong miền tần số sẽ tương ứng với một hằng số trong miền thời gian nên nó có chu kỳ là vô hạn chứ không phải là hữu hạn. Theo đó, nếu đặc tính lý tưởng trong miền tần số thì tương ứng không lý tưởng trong miền thời gian và ngược lại*. Vì vậy, khi lựa chọn hàm cửa sổ cần phải cân nhắc cẩn thận và tính đến méo do việc cửa sổ hóa. Có thể giảm thiểu méo này bằng cách chọn hàm cửa sổ phù hợp. Các đặc tính mong muốn của hàm của sổ gồm (luôn lưu ý rằng, về mặt lý tưởng luôn mong muốn có xung kim trong miền tần số).

1. "Búp phổ chính hẹp" trong miền tần số chứa đựng hầu hết năng lượng

2. Các búp phụ nhỏ.

Hàm cửa sổ được sử dụng phổ biến nhất là hàm cửa sổ chữ nhật, cửa sổ Hamming và cửa sổ Kaiser. Khi chọn hàm cửa sổ, cần phải dung hoa giữa việc giảm thiểu méo tín hiệu với khối lượng tính toán.

5.5.1. Thiết kế từ đáp ứng biên độ

Kỹ thuật cơ bản để thiết kế bộ lọc số FIR dựa trên đáp ứng tần số (biên độ và pha) và đáp ứng xung kim đơn vị của bộ lọc, chúng là cặp biến đổi Fourier. Ta rút ra đáp ứng xung kim đơn vị bằng cách xác định rõ đáp ứng biên độ mong muốn $A(f)$ sau đó biến đổi Fourier ngược. Đáp ứng biên độ đích thường được xác định là thực chẵn vì vậy đáp ứng xung kim đơn vị cũng là thực và chẵn. Vì đáp ứng xung kim là chẵn (không nhân quả) và để thực thi hệ thống trong miền thời gian thì đáp ứng xung kim phải được cắt ngắn sao cho hữu hạn và phải được dịch thời để nó nhân quả. Việc cắt xén đáp ứng xung kim phải được thực hiện cẩn thận nếu không nó sẽ gây ra lỗi rất lớn. Lựa chọn cửa sổ phù hợp sẽ giảm ảnh hưởng các lỗi này. Dịch thời xung kim làm cho bộ lọc đáp ứng pha tuyến tính tương đương với trễ nhóm bằng dịch thời. Kỹ thuật này tạo ra bộ lọc có đáp ứng biên độ tùy ý và dịch pha tuyến tính. Nếu muốn có bộ lọc có cả đáp ứng biên độ và pha chiến lược thì phải xác định rõ hàm truyền đạt phức.

Đáp ứng biên độ của bộ lọc số là hàm liên tục của biến tần số *f*, tuần hoàn theo tần số lấy mẫu và được biểu diễn dưới dạng chuỗi Fourier. Các hệ số Fourier là rời rạc, tạo ra đáp ứng xung kim bộ lọc số mong muốn. Ta sẽ thấy trong các ví dụ dưới đây, tìm đáp ứng xung kim bằng cách biến đổi Fourier ngược lên đáp ứng tần số hoặc thực hiện IFFT lên tập các mẫu của đáp ứng biên độ. Kỹ thuật này rất cơ bản và nếu áp dụng cẩn thận nó sẽ tạo ra các bộ lọc rất tốt.

Vì đáp ứng biên độ của bộ lọc số là tuần hoàn theo tần số lấy mẫu nên nó được khai triển ở dạng chuỗi Fourier:

$$H(e^{j2\pi fT}) = \sum_{n=0}^{M-1} h[n] e^{-j2\pi nfT} \tag{5.64}$$

Trong đó $h[n]$ là hệ số Fourier, M là chiều dài của đáp ứng xung kim và $T = 1/f_s$. Lưu ý rằng, (5.64) chính xác là (5.6) và (5.4) được sử dụng để đạt được đáp ứng tần số trạng thái ổn định từ hàm truyền đạt $H(z)$. Giả sử M = 2L+1 (lý do sẽ đề cập sau). Với giả thiết này ta thế vào và thay đổi hệ số $k = n - L$ ta có:

$$H(e^{j2\pi fT}) = \sum_{k=-L}^{L} h[k+L] e^{-j2\pi(k+L)fT} \tag{5.65}$$

$$H(e^{j2\pi fT}) = e^{-j2\pi LfT} \sum_{k=-L}^{L} h[k+L].e^{-j2\pi kfT} \tag{5.66}$$

Được viết dưới dạng:

$$H(e^{2j\pi fT}) = e^{-j2\pi LfT} .H_1(e^{j2\pi fT}) \tag{5.67}$$

Trong đó:

$$H_1(e^{j2\pi fT}) = \sum_{k=-L}^{L} h_1[k]e^{-j2\pi kfT} \tag{5.68}$$

Rõ ràng $h_1[k] = h[k+L]$ là đáp ứng xung kim của bộ lọc nhân quả, được định nghĩa bởi (5.64) được dịch L mẫu. Lưu ý, bộ lọc được định nghĩa bởi (5.68) không phải là bộ lọc nhân quả. Tuy nhiên, nó rất dễ để thiết kế bằng kỹ thuật Fourier cơ bản và có thể chuyển thành nhân quả bằng cách dịch thời đáp ứng xung kim phù hợp. Đáp ứng biên độ của các bộ lọc được định nghĩa bởi $h[n]$ và $h_1[n]$ là giống nhau và hàm truyền đạt chỉ khác nhau bởi dịch pha tuyến tính như được định nghĩa bởi (5.67). Vì vậy, quy trình thiết kế dựa vào (5.68).

Giả sử bộ lọc có đáp ứng biên độ cho trước là $H_1(e^{j2\pi fT}) = A(f)$. Nhân cả hai vế của (5.68) với $exp(j2\pi mfT)$ ta có:

$$A(f).e^{j2\pi mfT} = \sum_{k=-L}^{L} h_1[k]e^{j2\pi(m-k)fT} \tag{5.69}$$

Tích phân hai phía của (5.69) trên độ rộng băng tần mô phỏng, nó là một chu kỳ của $H_1(e^{j2\pi fT})$ ta có:

$$\sum_{k=-L}^{L} h_1[k].I(m,k) = \int_{-f_s/2}^{f_s/2} A(f)e^{j2\pi mfT}\,df \tag{5.70}$$

Trong đó:

$$I(m,k) = \int_{-f_s/2}^{f_s/2} e^{j2\pi(m-k)fT}\,df \tag{5.71}$$

Tích phân, và thừa nhận $f_sT = 1$ ta được:

$$I(m,k) = \frac{1}{T}\frac{\sin\pi(m-k)}{\pi(m-k)} = \frac{1}{T}\delta(m-k) \tag{5.72}$$

Thế kết quả này vào (5.70) ta có:

$$h_1[m] = T\int_{-f_s/2}^{f_s/2} A(f)e^{j2\pi mfT}\,df, \qquad -L \le m \le L \tag{5.73}$$

Nó là phương trình thiết kế cơ bản của ta.

Ví dụ 5.8: Thiết kế bộ lọc thông thấp số, xấp xỉ với bộ lọc số lý tưởng có độ rộng băng tần là $\lambda f_N = \lambda f_s/2$, trong đó f_N là tần số Nyquist, f_s là tần số lấy mẫu, λ là tham số nằm giữa 0 và 1. Vì vây, đáp ứng biên độ mong muốn là:

$$A(f) = \begin{cases} 1, & |f| < \lambda f_s/2 \\ 0, & |f| \ge \lambda f_s/2 \end{cases} \tag{5.74}$$

Từ (5.73) ta có:

$$h_1[m] = T\int_{-\lambda f_s/2}^{\lambda f_s/2} (1)e^{j2\pi mfT}\,df \tag{5.75}$$

Thực hiện tích phân trên ta có:

$$h_1[m] = T.\frac{1}{\pi mT}.\frac{1}{j2}\left[e^{j\pi m\lambda f_s T} - e^{-j\pi m\lambda f_s T}\right] \tag{5.76}$$

Vì $f_s T = 1$ nên (5.76) được viết:

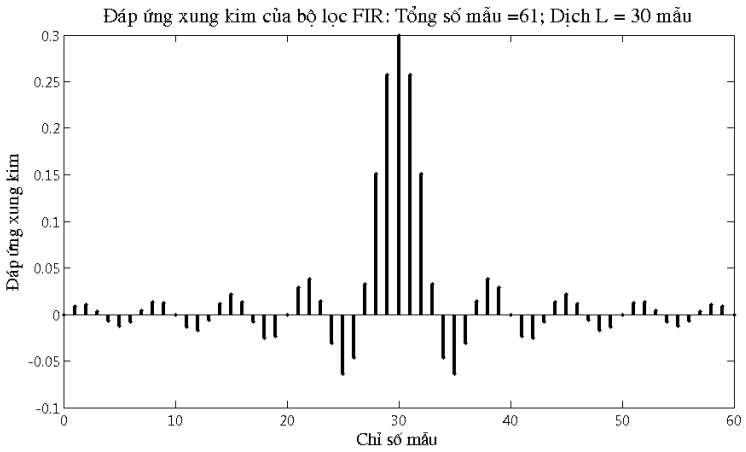
$$h_1[m] = \frac{1}{\pi m} sin\left(\pi m\lambda\right) \tag{5.77}$$

Lưu ý là $h_1[0] = h[L] = \lambda$.

Mã chương trình Matlab để khai thác kỹ thuật này được cho ở file **NVD5_FIRdesign.m** trong Phụ lục 5A.

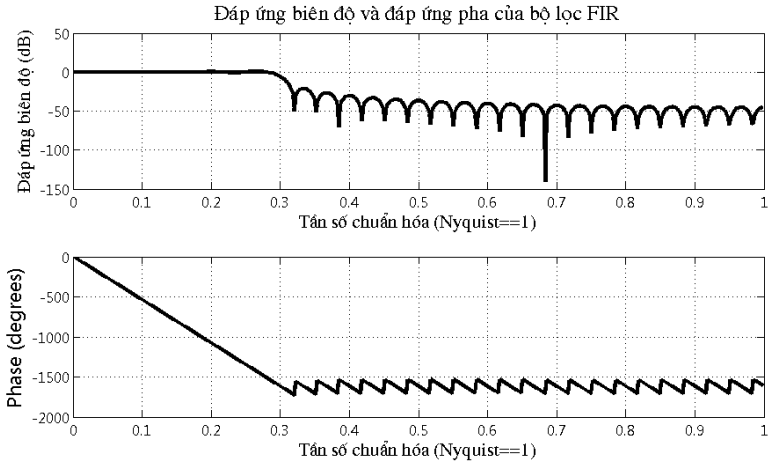
Khi nghiên cứu chương trình mô phỏng cần lưu ý: *(i)* trong chương trình này sử dụng L = 30 và λ = 0,3; *(ii)* **eps** được thêm vào chỉ số *m* để tránh tình trạng không xác định tại *m* = 0; *(iii)* Cần phải dịch *L* = 30 mẫu để làm cho xung kim là nhân quả được thực hiện theo cách Matlab đánh chỉ số véc tơ.

Đáp ứng xung kim được tạo ra bởi chương trình Matlab, kết quả được minh họa ở hình 5.11. Cần lưu ý: *(i)* $h[30] = 0,3$ đã được xác định trước; *(ii)* xung kim là hàm chẵn qua trọng số trung tâm tại $m = L = 30$. Để được tường minh, cần đọc kỹ mã chương trình Matlab, chạy từng lệnh và thay đổi các tham số cho mỗi lần chạy chương trình.



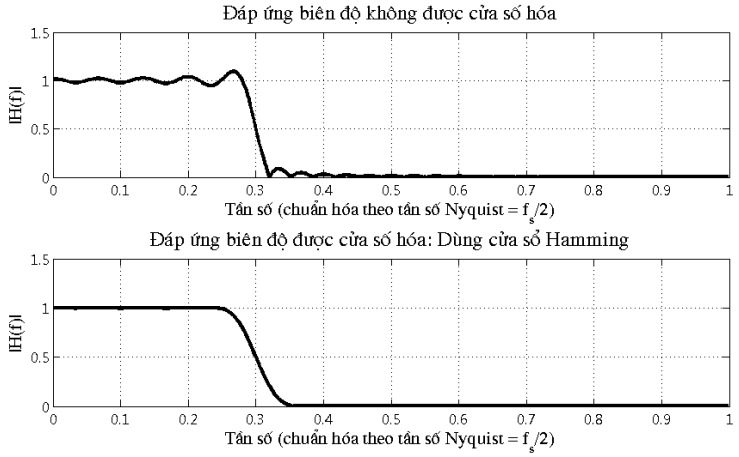
Hình 5.11: Đáp ứng xung kim của bộ lọc FIR

Đáp ứng pha và biên độ của bộ lọc FIR, được tạo ra bởi lệnh **freqz** trong chế độ vẽ mặc định (để được sáng tỏ, ta tìm hiểu hàm **freqz** dùng lệnh **doc freqz**), kết quả được minh họa ở hình 5.12. Như mong đợi, bộ lọc có pha tuyến tính vì đáp ứng xung kim là hàm chẵn qua trọng số trung tâm (xem hình 5.11). Dịch pha tuyến tính (trễ nhóm không đổi) do dịch đáp ứng xung kim một khoảng L mẫu làm cho bộ lọc có tính nhân quả (xem hình và mã chương trình Matlab). Đáp ứng pha tựa hình răng cưa trong băng chặn do đổi dấu của đáp ứng biên độ (xem hình 5.12).



Hình 5.12: Đáp ứng biên độ và pha của bộ lọc

Đáp ứng biên độ trong băng thông của bộ lọc là phẳng ở hình 5.12. Để khảo sát và trực quan hóa mô phỏng, vẽ đáp ứng biên độ lần thứ hai dùng lấy tỉ lệ tuyến tính để tránh nén biên độ do lấy tỉ lệ loga. Kết quả được cho ở nửa trên của hình 5.13. Cho thấy rằng có nhiều gợn sóng trong hình là do cắt ngắn bớt đáp ứng xung kim bằng cách nhân đáp ứng xung kim với hàm cửa sổ, tương đương với lấy tích chập đáp ứng tần số lý tưởng với $sin(\beta f)/\beta f$ trong miền tần số (Giá trị β được xác định bởi độ rộng cửa sổ). Có thể giảm độ gợn sóng do cửa sổ hóa bằng cách dùng hàm cửa sổ mịn hơn chuyển dịch từ $w[n]=1$ tới $w[n]=0$. Nhắc lại là ta mong muốn hàm cửa sổ giống hàm xung kim hơn nữa trong miền tần số.



Hình 5.13: Ảnh hưởng của cửa sổ hóa

Để được tường minh, ta chạy chương trình từng lệnh và thay đổi các giá trị khác nhau mỗi khi chạy chương trình cũng như tìm hiểu các hàm trong mã chương trình.

Hàm cửa sổ thường dùng là cửa sổ Hamming được định nghĩa bởi các trọng số:

$$w[n]=0,54+0,46\cos\left(\frac{\pi n}{L}\right),\quad -L\le n\le L \tag{5.78}$$

Lưu ý rằng, hàm cửa sổ *w[n]* phải được dịch thời L mẫu sao cho *w[0]* được trung tâm trên *h[n]*. Kết quả sử dụng bộ lọc Hamming được minh họa ở nửa dưới của hình 5.13. Lưu ý rằng, phần gợn sóng cho cả dải thông và dải chắn đều được nén khi sử dụng của sổ Hamming.

Ví dụ 5.9: Thiết kế bộ lọc số thông dải có đáp ứng biên độ của bộ lọc tương tự Butterworth pha tuyến tính. Bộ lọc tương tự Butterworth được định nghĩa bởi đáp ứng biên độ:

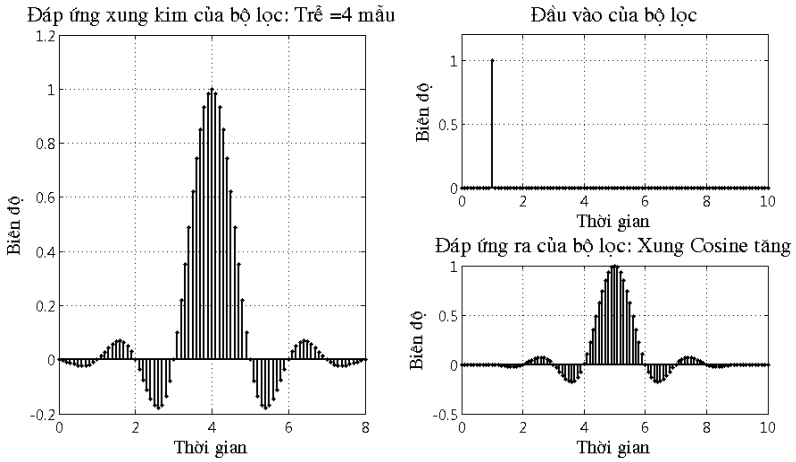
$$A(f) = \frac{1}{\sqrt{1 + (f / f_c)^n}} \tag{5.79}$$

Trong đó f_c là độ rộng băng tần, hay tần số 3dB và n là bậc của bộ lọc. Tiếp theo là lấy mẫu đáp ứng tần số, các mẫu đáp ứng tần số được cho bởi:

$$A(f_k) = \frac{1}{\sqrt{1 + (f_k / f_c)^n}} \tag{5.80}$$

Thực hiện thuật toán IFFT lên các mẫu tần số $A(f_k)$ sẽ tạo ra đáp ứng xung kim. Cần lưu ý rằng, khi thực hiện IFFT N điểm, các mẫu thời gian âm sẽ xuất hiện trong véc-tơ kết quả có dải chỉ số từ $(N/2)+1$ đến N và phải được đánh chỉ số lại để có đáp ứng xung kim. Mã chương trình Matlab tính toán các mẫu đáp ứng xung kim cho bộ lọc pha tuyến tính có đáp ứng biên độ Butterworth là file **NVD5_firbutter.m** được cho ở Phụ lục 5A.

Kết quả chạy chương trình Matlab tạo ra đáp ứng xung kim được minh họa ở hình 5.14 cho bậc = 1 và bậc = 30. Đáp ứng là một hàm mũ giảm hai phía (hai phía do đáp ứng biên độ $A(f)$ là thực). Khi bậc là 30, thì đáp ứng xung kim sẽ xấp xỉ hàm **sinc** vì bộ lọc Butterworth có bậc rất cao xấp xỉ bộ lọc lý tưởng. Do vậy, kết quả được minh họa ở hình 5.14 là đúng.



Hình 5.14: Kết quả tổng hợp bộ lọc - đáp ứng xung kim

Thực tế, trong mô phỏng mong muốn giảm thiểu khối lượng tính toán bằng cách giảm thiểu số lượng trọng số (thành phần b_k được dùng để biểu diễn đáp ứng xung kim). Khi giảm thiểu số lượng trọng số sẽ giảm thiểu số phép nhân và số lượng các tầng trong đường trễ rẽ

nhánh (thanh ghi dịch). Xuất hiện nhiều trọng số trong đáp ứng xung kim bậc = 1 là xấp xỉ bằng không và có thể được khử với mức độ ảnh hưởng không đáng kể lên đáp ứng biên độ. Điều này cũng đúng với đáp ứng xung kim bậc 30. Vì vậy, ta nên thực hiện cửa sổ hóa đáp ứng xung kim bằng hàm cửa số chữ nhật có độ rộng đủ lớn để chứa tất cả các thành phần quan trọng trong đáp ứng xung kim đó và loại bỏ các thành phần ngoài cửa sổ.

5.5.2. Thiết kế từ đáp ứng xung kim

Trong ví dụ 5.9, bộ lọc số được tổng hợp từ chỉ tiêu đặc tính đáp ứng biên độ và pha, là một ứng dụng của *lấy mẫu tần số*. Trong nhiều ứng dụng, biểu thức giải tích đáp ứng xung kim của bộ lọc số. Khi này, lấy mẫu đáp ứng xung kim cho ta thiết kế bộ lọc FIR. Thường được dùng để định dạng xung trong truyền dẫn tín hiệu và số liệu là chuỗi xung kim được phân tách về thời gian một khoảng *T*. Trường hợp nhị phân ta có:

$$d(t) = \sum_k d_k \delta(t - kT)$$
$$\text{trong đó } d_k = \begin{cases} 1, & \text{bit nhị phân } 1 \\ -1, & \text{bit nhị phân } 0 \end{cases} \tag{5.81}$$

Cho nó qua bộ lọc định dạng xung có đáp ứng xung kim là *p(t)* ta được dạng sóng:

$$x(t) = \sum_k d_k p(t - kT) \tag{5.82}$$

Sau đây sẽ xét cho hai trường hợp thông dụng.

***Ví dụ 5.10*:** Định dạng xung *p(t)* được dùng đối với truyền số liệu thường được chọn là một dạng xung thỏa mãn tính chất Nyquist ISI = 0. Một ví dụ về định dạng xung ISI = 0 là xung cosin tăng được cho trong miền tần số là:

$$P(f) = \begin{cases} T, & 0 \le |f| \le \dfrac{1-\beta}{2T} \\ \dfrac{T}{2}\left[1 + \cos\dfrac{\pi T}{\beta}\left(|f| - \dfrac{1-\beta}{2T}\right)\right], & \dfrac{1-\beta}{2T} < |f| \le \dfrac{1+\beta}{2T} \\ 0, & |f| > \dfrac{1-\beta}{2T} \end{cases} \tag{5.83}$$

T là khoảng thời gian xung hay thời gian ký hiệu. Lấy biến đổi Fourier ngược ta được dạng xung:

$$p(t) = \frac{\sin \pi t/T}{\pi t/T} \frac{\cos(\pi \beta t/T)}{1 - 4\beta^2 t^2/T^2} \tag{5.84}$$

Thấy rõ, đây là một xung không nhân quả. Cụ thể, một khi làm trễ xung đi một số nguyên lần chu kỳ ký hiệu, giả sử là *mT* và cắt ngắn xung còn *2mT*. Giá trị *m* để dung hòa giữa các yêu cầu độ chính xác và tính tiện dụng. Sau đó ta lấy *k* mẫu trên một chu kỳ ký hiệu sao

cho $T = kT_s$ với T_s là chu kỳ lấy mẫu. Thay t bằng $t\text{-}t_d = t\text{-}mT$, đặt $t = nT_S$ và $T = kT_S$ nhận được:

$$\frac{t}{T} \to \frac{nT_s - mkT_s}{kT_s} = \frac{n}{k} - m \tag{5.85}$$

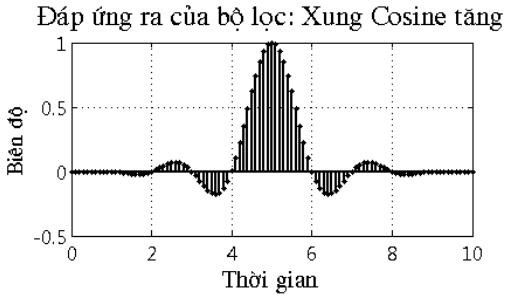
Thế vào trong công thức (5.84) cho ta chuỗi mẫu *p[n]* biểu diễn đáp ứng xung kim của bộ lọc số là:

$$p[n] = \frac{\sin\pi\left\{\frac{n}{k} - m\right\}}{\pi\left\{\frac{n}{k} - m\right\}} \times \frac{\cos\left(\pi\beta\left\{\frac{n}{k} - m\right\}\right)}{1 - 4\beta^2\left\{\frac{n}{k} - m\right\}^2}, \quad 0 \le n \le 2m \tag{5.86}$$

Để minh họa đáp ứng cosin tăng, ta đặt $x[n] = \delta[n-1]$ là đầu vào của bộ lọc có đáp ứng xung kim *p[n]*. Đầu ra là:

$$y[n] = p[n] \otimes x[n] = p[n-1] \tag{5.87}$$

Kết quả được minh họa ở hình 5.15 với các tham số $\beta = 0{,}32$, $k = 10$ mẫu trên một ký hiệu và m = 4 ký hiệu. Thấy rõ, *p[n]* bằng không tại các vị trí là bội số nguyên lần khoảng thời gian xung giả định T = 1. Mã chương trình Matlab được cho bởi file **NVD5_rcossim.m** để tạo ra hình 5.15 trong Phụ lục 5A.



Hình 5.15: Minh họa xung cosin tăng $\beta = 0{,}32$, $k = 10$ mẫu trên một ký hiệu và m = 4 ký hiệu

Trong nhiều thiết kế hệ thống, thực thi hàm truyền đạt *P(f)* là nối tầng của hai bộ lọc mỗi bộ lọc có hàm truyền đạt $\sqrt{P(f)}$. Một được dùng ở máy phát và một được dùng ở máy thu. Dẫn đến:

$$p_{SQRC}(t) = 4\beta\frac{\cos\left[(1+\beta)\pi t/T\right] + \sin\left[(1-\beta)\pi t/T\right]\left(4\beta t/T\right)^{-1}}{\pi\sqrt{T}\left[1 - 16\beta^2 t^2/T^2\right]} \tag{5.88}$$

Biểu thức (5.85) được áp dụng để thực hiện trễ và lấy mẫu. Kết quả là:

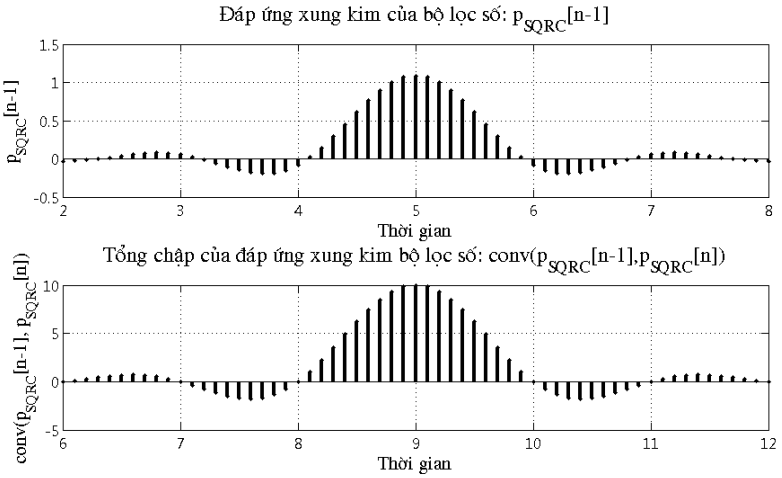
$$p_{SQRC}[n] = 4\beta\frac{\cos\left\{(1+\beta)\pi[(n/k)-m]\right\} + \sin\left\{(1-\beta)\pi[(n/k)-m]\right\} \times \left\{4\beta[(n/k)-m]\right\}^{-1}}{\pi\sqrt{T}\left\{1 - 16\beta^2\left[(n/k)-m\right]^2\right\}} \tag{5.89}$$

Hình 5.6 (phần trên) minh họa đáp ứng của bộ lọc có đáp ứng xung kim $p_{SQRC}[n]$ khi đặt vào $\delta[n-1]$. Nó là:

$$p_{SQRC}[n] \otimes \delta[n-1] = p_{SQRC}[n-1] \tag{5.90}$$

Tham số bộ lọc là $\beta = 0{,}32$, $k = 10$ mẫu trên một ký hiệu, T = 1, m = 4 ký hiệu. Lưu ý: *(i)* điểm cắt không xảy ra không chính xác tại bội số nguyên lần của T = 1. Nửa dưới của hình 5.16 minh họa kết quả lấy tích chập của chuỗi mẫu (được cho ở hình 5.16) với $p_{SQRC}[n]$. Về cơ bản nó thể hiện tích chập của $p_{SQRC}[n]$ với chính nó và tương đương với hai bội lọc SQRC trong nối tầng. *(ii)* các điểm không bây giờ rơi vào các vị trí là bội số nguyên lần chu kỳ T = 1 vì kết hợp xếp tầng của hai bộ lọc SQRC là một bộ lọc ISI = 0. Mã chương trình Matlab được cho ở file **NVD5_sqrcsim.m** trong Phụ lục 5A thực hiện tính toán các kết quả trên.

Kết quả nghiên cứu phần này được ứng dùng ở chương 18 khi đó ta mô phỏng hệ thống thông tin vệ tinh.



Hình 5.16: Xung cosin tăng căn bậc 2: ***SQRC***

5.5.3. Thực hiện mô hình mô phỏng bộ lọc FIR

Các mô hình FIR đóng vai trò trung tâm trong mô phỏng hệ thống truyền thông. Trong các phần trước, kỹ thuật thiết kế được sáng tỏ cho trường hợp hàm truyền đạt hay đáp ứng xung kim là đã được biết trước ở dạng giải tích do đó nó được lấy mẫu. Một điểm quan trọng nữa khi sử dụng mô hình mô phỏng FIR là các giai đoạn sau của quá trình thiết kế. Khi này, bộ lọc đã thực sự được thiết kế và xây dựng vì vậy đáp ứng tần số là khả dụng. Các mô hình mô phỏng FIR rất phù hợp để mô phỏng bộ lọc có đáp ứng tần số được cho ở dạng đo đạc.

Ta tóm tắt các bước quan trọng để mô phỏng bộ lọc có đáp ứng tần số tùy ý (đáp ứng tần số này được xác định ở dạng bảng số liệu đo kiểm). Muốn vậy, ta tiến hành ba bước cơ bản là: chọn tham số, lập mô hình mô phỏng và thực hiện mô phỏng bằng Matlab:

Bước 1 (chọn tham số): Chọn hai tham số quan trọng cơ bản là: tần số lấy mẫu và khoảng thời gian của đáp ứng xung kim đã được cắt. Các tham số thực thi bộ lọc FIR phải được chọn cẩn thận để giảm thiểu mức độ phức tạp tính toán đồng thời vẫn đáp ứng sự phân giải trong miền thời gian và tần số. Ta nên chọn các tham số theo trình tự sau:

- Tần số lấy mẫu f_s: $f_s > (16B \div 32B)$, B là độ rộng băng của bộ lọc. Độ phân giải thời gian (thời gian giữa các mẫu) T_s là $1/f_s$:

- Phân giải tần số, $\Delta_f = \frac{1}{NT_s} = \frac{f_S}{N}$, trong đó $N = \frac{f_S}{\Delta_f}$. Giá trị hay dùng là $\Delta_f = \left(\frac{B}{64} \div \frac{B}{32}\right)$.

- Số lượng mẫu/ký hiệu là số nguyên và là lũy thừa của 2. Giá trị nhỏ nhất là 8.

- Khoảng thời gian của đáp ứng xung kim thường từ 8 đến 6 ký hiệu.

Từ đó, dẫn đến lựa chọn N = 1024 với N là số lượng nhánh bộ lọc hay là số lượng mẫu của đáp ứng xung kim.

Bước 2 (lập mô hình): Sau khi chọn các tham số chủ đạo là f_s và N (do vậy ta có T_s và Δ_f) sau đó thực hiện tiền xử lý số liệu đáp ứng tần số và tiến hành lập mô hình mô phỏng như sau:

- *Trước tiên là tiền xử lý số liệu đáp ứng tần số*: Bước này bao gồm chuyển thông dải thành thông thấp (nếu bộ lọc là bộ lọc thông dải), lấy mẫu lại số liệu đáp ứng tần số và chuyển trễ nhóm thành pha bằng cách tích phân dữ liệu trễ nhóm. Việc chuyển thông dải thành thông thấp bằng cách đánh nhãn lại trục tần số ($f_1 = f - f_c$). Dẫn đến tạo ra một tương đương thông thấp không đối xứng liên hợp qua f = 0. Khi này, đáp ứng xung kim của bộ lọc tương đương thông thấp có thể là giá trị phức.
- *Tiếp đó là lấy mẫu lại:* Cần phải lấy mẫu lại nếu các kết quả đo phân tích mạng được cho tại các điểm tần số khác với dữ liệu được sử dụng để thực thi FIR. Chẳng hạn, giả sử có 100 mẫu đáp ứng tần số được tập hợp từ máy đo phân tích mạng và việc thực thi FIR dựa trên N = 1024 mẫu. Khi này, đáp ứng tần số phải được nội suy. Nội suy tuyến tính đơn giản thích hợp trong hầu hết các trường hợp.
- *Cuối cùng là chuyển đáp ứng trễ nhóm thành đáp ứng pha:* Một lần nữa, số liệu trễ nhóm được chuyển thành đáp ứng pha bằng cách lấy tích phân trễ nhóm theo tần số.

Bước 3 (thực thi bằng Matlab): Một khi đáp ứng tần số của phiên bản tương đương thông thấp của bộ lọc là khả dụng thì việc thực thi mô hình bộ lọc FIR trên Matlab gồm các bước sau đây:

1. Mở rộng số liệu đáp ứng tần số từ $-f_s/2$ đến $f_s/2$ và có được các giá trị mẫu của hàm truyền đạt $H(f)$, $-f_s/2 < f < f_s/2$ với $f = k\Delta_f$, $k = -N/2$ đến $N/2-1$, Δ_f là độ phân giải tần số ($\Delta_f = 1/(NT_s)$).

2. Di chuyển phần phổ âm của *H(f)* từ *N/2 + 1* tới *N* sao cho đáp ứng tần số được chứa trong $H(k\Delta_f)$, $k = 1,2,...,N$.
3. Lấy biến đổi IFFT để có được đáp ứng xung kim (lấy mẫu trong miền thời gian sẽ là $T_s = 1/(N\Delta_f)$).
4. Cửa sổ hóa hàm đáp ứng xung kim nếu cần thiết (phải chắc chắn rằng đáp ứng xung kim được "quay" chính xác và được trung tâm trong hàm cửa sổ). Chuẩn hóa đáp ứng xung kim đã được cửa sổ hóa để có năng lượng đơn vị.

Sau khi có được đáp ứng xung kim đã được cắt ngắn, mô phỏng bộ lọc trên Matlab bằng hàm *filter* với các tham số a = 1 và b = h, h là dãy đáp ứng xung kim, nghĩa là đầu ra bộ lọc được tính toán bằng dòng lệnh:

$$\text{Đầu ra} = filter(b, a, \text{đầu vào})$$

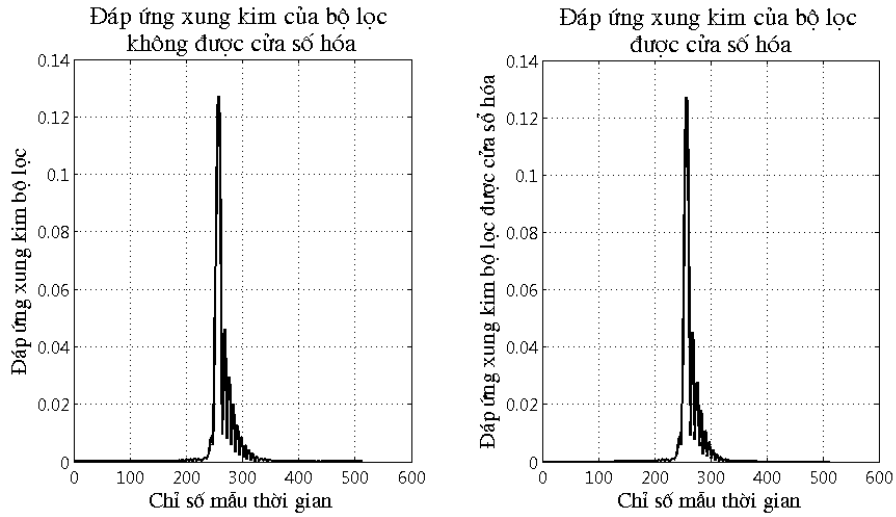
Nếu chuỗi đầu vào quá dài thì ta tính chuỗi đầu ra bằng hàm **fftfilt.m** như sau:

$$\text{Đầu ra} = fftfilt(b, \text{đầu vào}, N)$$

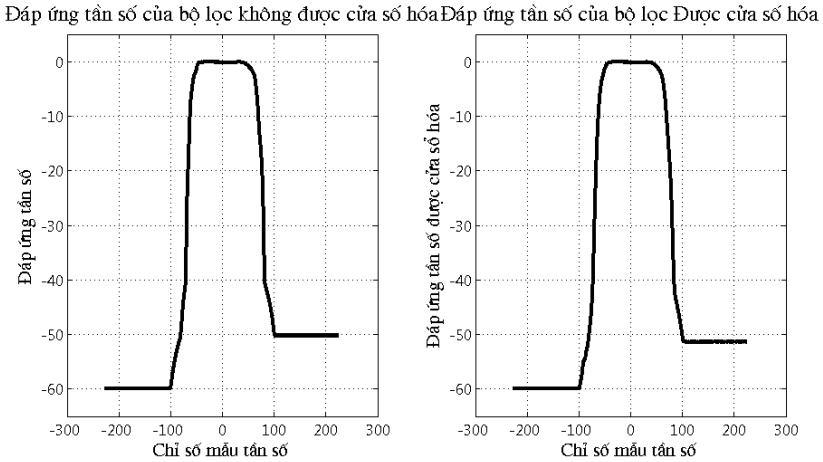
Thực thi bộ lọc theo khối bằng cách dùng phương pháp chồng lấn và cộng với kích thước khối là N như đã được đề cập.

Chương trình Matlab minh họa thực thi bài toán này được cho ở file **NVD5_FIRExample.m** trong Phụ lục 5A, cho thấy làm thế nào mô phỏng một bộ lọc định kênh thông dải 125 MHz được sử dụng trong vệ tinh. Các đặc tính dữ liệu đo được cho ở dạng các đặc tính trễ và đáp ứng biên độ (xem file **NVD5_Filter_Data.m** trong Phụ lục 5A).

Ví dụ 5.11: Chương trình Matlab để thiết kế bộ lọc FIR sử dụng kỹ thuật đã mô tả được cho ở file **NVD5_FIRExample.m** trong Phụ lục 5A. Số liệu đo được dùng để thiết kế cũng được cho ở **NVD5_Filter_Data.m** trong Phụ lục 5A. Kết quả mô phỏng được minh họa ở hình 5.17 và 5.18, với hình 5.17 minh họa đáp ứng xung kim được cửa số hóa và không được cửa sổ hóa. Đáp ứng được cửa sổ hóa (cắt ngắn) và không được cửa số hóa (không bị cắt ngắn) được vẽ độc lập bởi lẽ chúng khá giống nhau và sẽ không thể phân biệt được nếu để trong cùng một hình vẽ. Điểm khác nhau duy nhất giữa hai đáp ứng biên độ này là vùng phẳng trong phần vùng lân cận chỉ số mẫu từ 150 đến 200. Sự suy hao là lớn hơn một ít khi dùng hàm cửa sổ Hamming.



Hình 5.17: So sánh đáp xung (được cửa số hóa và không được cửa số hóa)



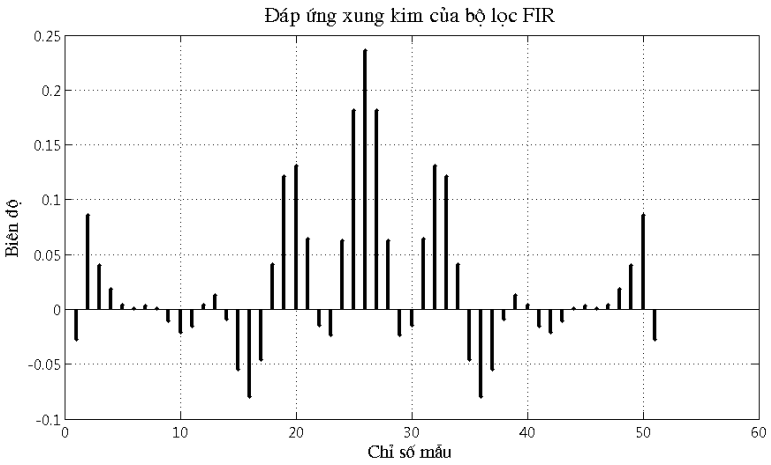
Hình 5.18: So sánh đáp ứng biên độ (được cửa sổ hóa và không được cửa sổ hóa)

5.5.4. Thiết kế bộ lọc số FIR với sự hỗ trợ của máy tính

Tính phổ cập nhất của kỹ thuật thiết kê trợ giúp máy tính CAD để thiết kế bộ lọc FIR là giải thuật Parks-McClellan, có thể áp dụng cho bộ lọc có đặc tính đáp ứng biên độ lý tưởng hằng số. Chẳng hạn áp dụng thuật toán Parks-McClellan để tổng hợp bộ lọc định kênh có đặc tính đáp ứng biên độ được cho ở hình 5.9. Thuật toán Parks - McCllellan sử dụng đa thức Chebyshev làm phù hợp với đáp ứng biên độ mong muốn và là xấp xỉ hóa đẳng gợn (độ gợn sóng như nhau) với đáp ứng biên độ mong muốn. Nghĩa là, về phương diện tỉ lệ tuyến tính, biên độ độ gợn sóng của dải thông và dải chắn là bằng nhau.

Ví dụ 5.12: Minh họa thiết kế dựa trên máy tính CAD bộ lọc FIR và thuật toán Parks-McClellan, xét đáp ứng biên độ mong muốn được cho ở hình 5.9. Chương trình Matlab để xác định xung kim của bộ lọc được cho bởi file **NVD5_pmc.m** trong Phụ lục 5A.

Lưu ý rằng, các véc-tơ xác định các điểm tần số và đáp ứng biên độ mong muốn tại các điểm tần số của chúng là chung cho cả 2 chương trình tổng hợp bộ lọc FIR và IIR.

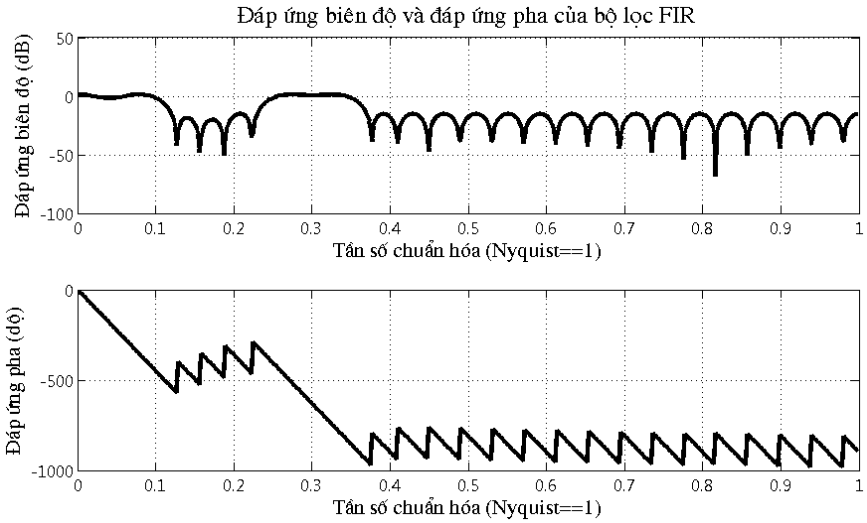


Hình 5.19: Đáp ứng xung kim của bộ lọc FIR

Kết quả chạy chương trình mô phỏng là đáp ứng xung kim được minh họa ở hình 5.19. Lưu ý rằng, có 51 thành phần trong đáp ứng xung kim, tương ứng với bậc là 50 và đáp ứng xung kim này là hàm chẵn qua thành phần trung tâm. Thực thi bộ lọc ở dạng đường trễ rẽ nhánh TDL với các giá trị được minh họa trong hình 5.19 là các trọng số nhánh. Thiết kế bộ lọc "hoạt động độc lập" hoặc liên kết với các chương trình mô phỏng sao cho quá trình thiết kế bộ lọc trở thành một phần của quá trình mô phỏng.

Đáp ứng pha và biên độ của bộ lọc số được minh họa ở hình 5.20. Lưu ý rằng, độ gợn dải thông và dải chắn là đặc điểm của quá trình thiết kế Parks - McClellan. Cho thấy, suy hao dải chắn khoảng 20 dB. Suy hao dải chắn có thể tăng bằng cách tăng tăng bậc của bộ lọc. Tăng bậc của bộ lọc cũng giảm độ gợn của dải thông. Yêu cầu về bậc của bộ lọc được xác định bởi độ gợn dải thông cho phép (độ gợn dải thông càng nhỏ thì yêu cầu bậc bộ lọc càng lớn), suy hao dải chắn (suy hao dải chắn càng lớn yêu cầu bậc bộ lọc càng lớn) và độ rộng của băng tần chuyển dịch (băng tần chuyển dịch càng hẹp yêu cầu bậc bộ lọc càng lớn). Thiết kế điển hình sử dụng quá trình Park-McClellan là một quá trình lặp trong đó các chỉ tiêu được điều chỉnh trong giới hạn cho phép với hy vọng sẽ nhận được đáp ứng xung kim có chiều dài thích hợp.

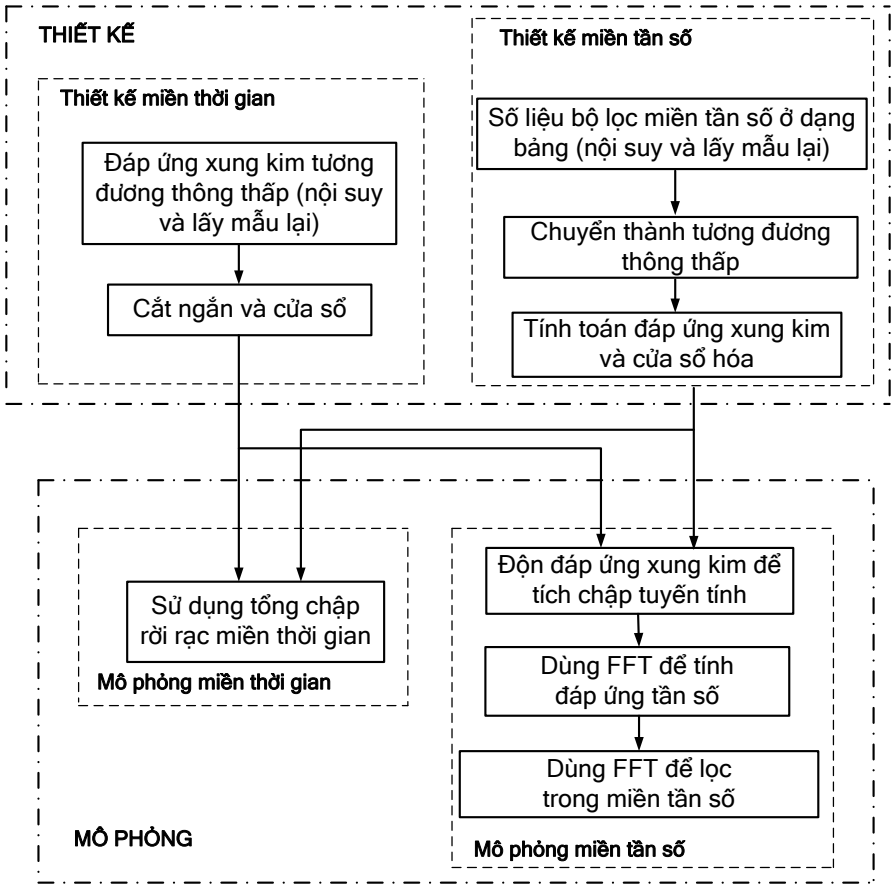
Vì đáp ứng xung kim có dạng hàm chẵn bị trễ, nên đáp ứng pha sẽ là tuyến tính như ở hình 5.20. Nhẩy 180^0 trong đáp ứng pha tương ứng với đổi dấu trong đáp ứng biên độ khi đáp ứng biên độ đi qua 0 ($-\infty$ dB). Độ dốc của đáp ứng pha, trễ nhóm của bộ lọc được xác định bởi trễ cần thiết để tạo ra đáp ứng xung kim của bộ lọc có tính nhân quả.



Hình 5.20: Đáp biên độ và pha của bộ lọc FIR

5.5.5. Bình luận về thiết kế bộ lọc FIR

Như đã đề cập, có thể thiết kế các bộ lọc FIR mà không cần có tương đương tương tự. Điều này càng có ý nghĩa bởi lẽ xu thế các hệ thống truyền thông dựa vào phần mềm (chẳng hạn: vô tuyến được định nghĩa bằng phần mềm hay mềm hóa cấu hình hệ thống...) với việc số hóa bộ lọc. Ở dạng thực thi, các ưu, nhược điểm quan trọng nhất được cho ở bảng 5.2. Hình 5.21 tóm tắt các kỹ thuật mô phỏng và thiết kế bộ lọc FIR. Lưu ý rằng, có thể thiết kế sử dụng số liệu trong miền tần số hoặc miền thời gian. Việc mô phỏng trong miền thời gian hay mô phỏng trong miền tần số tùy vào việc chọn số liệu thiết kế.



Hình 5.21: Thiết kế bộ lọc FIR và các kỹ thuật mô phỏng

Bảng 5.2: Các ưu nhược điểm của các kỹ thuật thực thi bộ lọc FIR

Thực hiện	Ưu điểm	Nhược điểm
Miền thời gian	Thực thi đơn giản	Mất nhiều thời gian với chuỗi đáp ứng xung kim dài.
Miền tần số	Xử lý nhanh Dễ dàng thiết kế bằng cách sử dụng dữ liệu đáp ứng tần số.	Gây trễ nhân tạo bằng với chiều dài khối FFT. Không thể sử dụng trong hệ thống có hồi tiếp.
Phương pháp CAD	Dẫn đến bộ lọc pha tuyến tính	Đáp ứng xung kim dài.