ĐẠI HỌC ĐÀ NẴNG

TRƯỜNG ĐẠI HỌC BÁCH KHOA

KHOA ĐIỆN TỬ - VIỄN THÔNG

GIÁO TRÌNH

THÍ NGHIỆM XỬ LÝ SỐ TÍN HIỆU

BỘ MÔN KỸ THUẬT MÁY TÍNH

Biên soạn: Th.S THÁI VĂN TIẾN

MỤC TIÊU CỦA HỌC PHẦN

1. Mục tiêu chung.

Sinh viên hiểu được phần mềm MATLAB và vận dụng phần mềm này giải quyết các bài toán của học phần lý thuyết đã học.

2. Mục tiêu cụ thể.

- Kiến thức: Nắm vững cơ sở lý thuyết tín hiệu và hệ thống và xử lý tín hiệu 1, 2. Đồng thời được trang bị kiến thức về xử lý tín hiệu làm tiền đề cho nghiên cứu đồ án chuyên ngành cũng như đồ án tốt nghiệp trong học kỳ kế tiếp.
- Kỹ năng: Am hiểu công cụ phần mềm tính toán MATLAB.
- Thái độ: Nghiêm túc, chấp hành các quy định của phòng thí nghiệm.

3. Tiêu chí và thang đánh giá.

- Sinh viên phải tham gia đầy đủ các buổi thí nghiệm theo lịch của phòng đào tạo nếu nghỉ từ ½ số buổi thí nghiệm trở lên sẽ không được làm bài test.
- Sinh viên phải tìm hiểu và đọc trước các tài liệu của thầy hướng dẫn yêu cầu.
- Sinh viên phải hoàn tất các bài Lab của thầy hướng dẫn.
- Thang đánh giá:
    - Chuyên cần : 10%
    - Thực hành : 20%
    - Kiểm tra : 70%

BÀI 1: TÌM HIỂU PHẦN MỀM MATLAB

MABLAB, viết tắt của Matrix Labotary, là một công cụ phần mềm hỗ trợ tính toán trên ma trận. MATLAB được tích hợp trên một môi trường chung một loạt các khả năng bao gồm tính toán, hiển thị kết quả và lập trình nhằm giải quyết các vấn đề liên quan đến toán học. Các vấn đề đó bao gồm:

- Các phương trình toán học và tính toán
- Phát triển các giải thuật
- Thu thập dữ liệu
- Mô hình hoá, mô phỏng và tạo các mẫu theo thiết kế
- Phân tích, khảo sát và thể hiện dữ liệu bằng hình ảnh
- Biểu diễn các biểu đồ mang tính khoa học và tính kỹ thuật
- Phát triển với các giao diện với người sử dụng.

Ưu điểm nổi bật của MATLAB, như đã được đề cập ở trên, là khả năng tính toán, đặc biệt là những bài toán liên quan đến ma trận và vector, với thời gian ít hơn nhiều lần so với cùng một công việc tính toán trên các ngôn ngữ lập trình khác như C hay Fortran. Khả năng lập trình của MATLAB cũng rất linh hoạt, cụ thể là trong việc tạo ra các câu lệnh riêng và các hàm của riêng người sử dụng.

Hệ thống MATLAB bao gồm 5 phần chính sau:

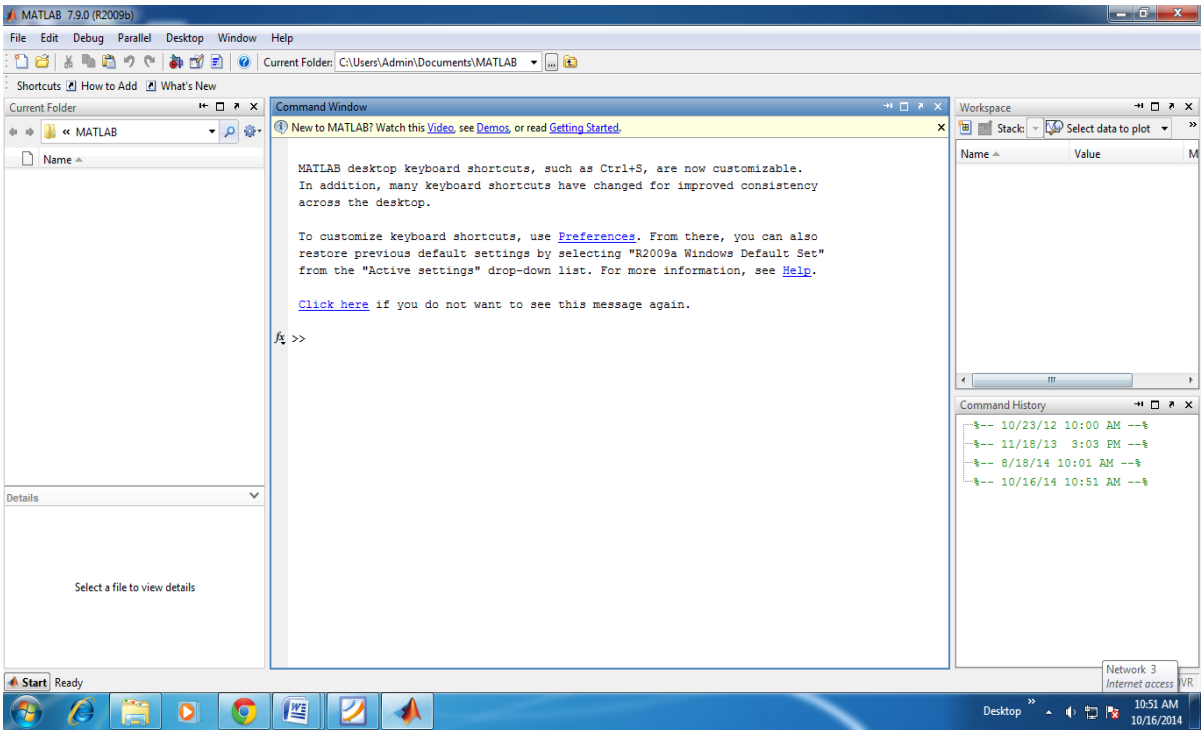
- ***Môi trường phát triển:*** là một tập hợp các công cụ, phần lớn trong chúng là các giao diện đồ hoạ, giúp người dùng sử dụng các câu lệnh và các hàm của MATLAB.
- ***Thư viện các hàm toán học:*** Là một tập hợp các hàm toán học bao gồm từ các hàm cơ bản như sin, cosin, các phép tính đại số phức đến các hàm phức tạp như tìm ma trận đảo, tìm ma trận riêng, hàm Bessel và biến đổi Fourier nhanh (Fast Fourier Transform – FFT).
- ***Ngôn ngữ lập trình:*** là một ngôn ngữ bậc cao liên quan đến ma trận và mảng. Trong MATLAB có đầy đủ những đặc trưng của một ngôn ngữ lập trình bao gồm các lệnh rẽ nhánh, các hàm, cấu trúc dữ liệu, nhập/xuất dữ liệu, và các đặc tính liên quan đến lập trình hướng đối tượng (***object-oriented programming***).
- ***Đồ hoạ:*** là một tập hợp các công cụ để biểu diễn ma trận và vector bằng đồ hoạ. Bên cạnh các công cụ ở mức thấp để thể hiện dữ liệu dạng 2 chiều và 3 chiều, xử lý hình ảnh tĩnh, ảnh động còn có các công cụ ở mức cao dùng để tạo ra các biểu diễn đồ hoạ theo ý đồ của người sử dụng cũng như tạo ra các giao diện đồ hoạ users.

- ***Các API:*** Là một thư viện cho phép người sử dụng gọi các hàm viết trên ngôn ngữ C và Fortran. Chúng bao gồm cả các công cụ cho phép gọi các hàm từ MATLAB dưới dạng liên kết động, và để đọc và ghi các tệp .MAT.

MATLAB, bên cạnh khả năng tính toán trên ma trận, đồng thời cũng là một ngôn ngữ lập trình mạnh. Các tệp chương trình của MATLAB được ghi dưới dạng đuôi .m, được gọi là M-files. Có hai loại tệp dạng đuôi .m:

- ***Tệp kịch bản (scripts):*** Loại tệp này không có các biến đầu vào và đầu ra, nó đơn thuần chỉ xử lý dữ liệu với các biến trên vùng làm việc hiện thời (***work space***) của MATLAB. Khi gõ tên tệp tại cửa sổ lệnh (***command window***), các lệnh được lưu trong nội dung của tệp lần lượt được gọi ra theo một kịch bản tuần tự từ trên xuống dưới.
- ***Tệp mô tả hàm (functions):*** Loại tệp này cần khai báo các biến đầu vào và đầu ra. Các biến được khai bên trong loại tệp này là các biến địa phương (local variables) và chỉ có phạm vi ảnh hưởng tại chính hàm số đó. Nội dung trong các tệp này nhằm mục đích tính toán các thông số đầu ra dựa trên các tham số đầu vào của hàm số. Tên của tệp loại này cần trùng với tên của hàm số được khai báo và mô tả bên trong nội dung của tệp.

Để khởi động **MATLAB**, người sử dụng có thể nháy đúp chuột vào biểu tượng MATLAB trên màn hình desktop hoặc vào menu ***Start -> All Programs -> MATLAB →R2009b → MATLAB R2009b*** từ giao diện của Windows. Sau khi MATLAB được khởi động, trên màn hình người sử dụng sẽ hiển thị lên môi trường phát triển tích hợp của MATLAB bao gồm một số cửa sổ như trong Hình 1.1.



Hình 1.1. Giao diện chính của MATLAB R2009b.

Trong đó có các cửa số quan trọng sau:

- ***Cửa sổ lệnh (Command Window):*** có chức năng thể hiện dấu nhắc để nhập vào các lệnh từ bàn phím, và hiển thị kết quả tính toán sau khi gõ một lệnh hoặc gọi một hàm.
- ***Cửa sổ các lệnh đã dùng (Command History):*** thể hiện danh mục các lệnh đã gõ hoặc các hàm đã được gọi theo các phiên làm việc.
- ***Cửa sổ thư mục hiện thời (Current Directory):*** thể hiện danh sách các tệp dạng đuôi .m đang tồn tại trong thư mục hiện thời. Để thay đổi thư mục hiện thời trên cửa sổ nhỏ nằm ngay bên trên cửa số lệnh.
- ***Vùng làm việc (Workspace):*** thể hiện danh mục tất cả các biến bao gồm: tên biến, giá trị hiện thời của biến, kiểu biến đang tồn tại ở phiên làm việc hiện tại.

Ngoài ra còn một loạt các cửa sổ khác sẽ được kích hoạt và hiển thị khi gọi một lệnh hoặc chọn một mục trong phần Menu của MATLAB. Để biết thêm về các cửa số có thể tham khảo thêm trong phần trợ giúp (Help) của MATLAB bằng cách nhấn phím F1.

Để soạn thảo một kịch bản hoặc một hàm, thực hiện chọn menu **File -> New -> M-File** hoặc nhắp chuột vào biểu tượng **New M-File** trên thanh công cụ (**Toolbar**). Trên màn hình sẽ hiển thị lên cửa sổ soạn thảo (**Editor**) có đầy đủ các chức năng soạn thảo giống như bất cứ môi trường soạn thảo của ngôn ngữ lập trình nào khác.

Để xem trợ giúp về một lệnh hay một hàm có sẵn nào đó của MATLAB, gõ lệnh help kèm theo tên của lệnh hoặc hàm từ cửa sổ lệnh của MATLAB,

ví dụ: >> **help fft**

trên cửa số lệnh sẽ đưa ra nội dung về chức năng, cú pháp cho các tham số vào/ra cho hàm thực hiện phép biến đổi **Fourier** nhanh được MATLAB đặt dưới tên FFT.

BÀI 2: TÍN HIỆU VÀ HỆ THỐNG RỜI RẠC TRONG MIỀN THỜI GIAN RỜI RẠC n

2.1. Cơ sở lý thuyết.

Xử lý số tín hiệu, về bản chất, là tìm hiểu về các phép toán và giải thuật liên quan đến các tín hiệu rời rạc và các hệ thống rời rạc. Các tín hiệu rời rạc thường được thể hiện dưới dạng một dãy số như sau:

{…, x(-3), x(-2), x(-1), x(0), x(1), x(2), x(3), …}

Tuy nhiên, MATLAB chỉ có khả năng biểu diễn một dãy số với độ dài hữu hạn. Khi đó dãy số được khai báo và lưu trữ dưới dạng vector, ví dụ:

>> x = [3, 2, -1, 7, -5]

Với cách khai báo như vậy, dãy số không thể hiện được chỉ số của các thành phần trong dãy. Vì vậy, để biểu diễn một dãy rời rạc có độ dài hữu hạn, ta cần khởi tạo và lưu

trữ chúng dưới dạng 2 vector.

Ví dụ:

>> n = [-2:2]

>> x = [3, 2, -1, 7, -5]

Với ***x*** là một dãy gồm 5 phần tử xuất phát từ **-2** đến **2** có: **x(-2) = 3, x(-1) = 2, x(0) = -1, x(1) = 7 và x(2) = -5.** Trong tất cả các bài thí nghiệm trên MATLAB của môn học này, chúng ta nên tuân theo một nguyên tắc như vậy.

Định nghĩa một số dãy cơ bản :

1) Dãy xung đơn vị:

$$\boldsymbol{\delta(n) = \begin{cases} 1 & n = 0 \\ 0 & n \neq 0 \end{cases}}$$

Dãy xung đơn vị trễ (dịch) đi $\mathbf{n_0}$ mẫu:

$$\boldsymbol{\delta(n - n_0) = \begin{cases} 1 & n = n_0 \\ 0 & n \neq n_0 \end{cases}}$$

2) Dãy nhảy đơn vị:

$$\boldsymbol{u(n) = \begin{cases} 1 & n \geq 0 \\ 0 & n < 0 \end{cases}}$$

3) Dãy hàm mũ thực: $\boldsymbol{x(n) = a^n,\ \forall n\ \ a \in R}$

4) Dãy hàm mũ phức: $\boldsymbol{x(n) = e^{(\sigma + j\omega_0)n}, \forall n}$

Với, **σ :** là độ suy giảm của tín hiệu, $\boldsymbol{\omega_0}$ là tần số góc tính theo đơn vị r**adians.**

5) ***Dãy lượng giác:*** Dãy lượng giác là dãy thể hiện tín hiệu có dạng hàm toán học là tổ hợp tuyến tính của các hàm ***sin*** và ***cosin***. Một ví dụ về dãy lượng giác như sau:

$$x(n) = \cos(\omega_0 + \theta), \forall n$$

Với, **θ** là pha ban đầu của tín hiệu.

6) ***Dãy ngẫu nhiên:*** Là dãy mà các phần tử của dãy có giá trị ngẫu nhiên. Sự phân bố ngẫu nhiên có thể được điều chỉnh là phân bố đều hay tuân theo một quy luật phân bố xác suất nào đó. Trong MATLAB có sẵn một số hàm cho phép khởi tạo ra một dãy ngẫu nhiên theo phân bố đều và theo phân bố Gauss.

7) ***Dãy tuần hoàn:*** Dãy tuần hoàn là một dãy có giá trị của các phần tử lặp lại tuần hoàn sau một số mẫu nhất định: $x(n) = x(n + mN), m \in Z$

Hệ thống rời rạc :

Tín hiệu vào được gọi là đầu vào **(*input*)** hay kích thích **(*excitation*)** của hệ thống. Tín hiệu ra được gọi là đầu ra **(*output*)** hay đáp ứng **(*response*)** của hệ thống. Trong MATLAB, hệ thống được định chung bởi khái niệm **"*filter*"**.

- ***Một hệ thống là tuyến tính bất biến*** (***Linear Time-Invariant – LTI***) nếu nó hội đủ cả hai tính chất tuyến tính (*linearity*) và bất biến theo thời gian (*time-invariance*). Tính chất tuyến tính nói lên rằng đáp ứng của hệ thống với kích thích là một tổ hợp tuyến tính các tín hiệu rời rạc sẽ bằng với tổ hợp tuyến tính của các đáp ứng, với mỗi đáp ứng này là đầu ra khi cho từng thành phần của đầu vào qua hệ thống. Tính chất bất biến theo thời gian nói lên rằng đáp ứng của hệ thống có dạng giống hệt nhau với cùng một kích thích mà không phụ thuộc vào thời điểm đưa kích thích tới đầu vào. *Trong môn học Xử lý số tín hiệu, tất cả các hệ thống được xét tới đều là tuyến tính bất biến*.

- ***Một hệ thống tuyến tính bất biến*** luôn có đáp ứng ra ***y(n)*** là tích chập (***convolution sum***) giữa đầu vào ***x(n)*** với dãy đáp ứng xung ***h(n)*** của hệ thống, là đáp ứng của hệ thống khi đưa kích thích ***δ(n)*** tới đầu vào. Tính tích chập bởi công thức:

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

- ***Một hệ thống là nhân quả*** nếu đáp ứng ra tại thời điểm hiện tại không phụ thuộc vào kích thích vào tại các thời điểm tương lai. Một hệ thống tuyến tính bất biến là nhân quả nếu đáp ứng xung thoả mãn: ***h(n) = 0 khi n < 0***

- ***Một hệ thống là ổn định*** (***BIBO Stable***) nếu với một kích thích bị chặn luôn sinh ra một đáp ứng cũng bị chặn, tức là giá trị của đáp ứng ra không tiến đến vô cùng. Một hệ thống tuyến tính bất biến là ổn định nếu đáp ứng xung thoả mãn:

$$\sum_{n=-\infty}^{\infty} |h(n)| < \infty$$

Nói chung, tất cả các hệ thống tuyến tính bất biến có thể thực hiện được, thông qua phần cứng hoặc mô tả phần mềm, đều được mô tả bởi phương trình sai phân tuyến tính hệ số hằng có dạng như sau:

$$\sum_{k=0}^{N} a_k y(n-k) = \sum_{r=0}^{M} b_r x(n-k)$$

hay có thể viết dưới dạng sau thích hợp với thể hiện mô hình sơ đồ khối của hệ thống:

$$y(n) = \sum_{r=0}^{M} b_r x(n-k) - \sum_{k=1}^{N} a_k y(n-k)$$

Trong MATLAB có hàm ***filter*** cho phép tìm dãy đáp ứng đầu ra ***y(n)*** nếu biết trước các biến đầu vào là các hệ số của phương trình sai phân, dãy $\mathbf{a_k}$ và $\mathbf{b_r}$, và kích thích đầu vào **x(n).** Chúng ta có thể dùng lệnh này để phác hoạ định dạng đầu ra của hệ thống với các tham số nêu trên.

2.2. Một số lệnh và hàm của MATLAB.

Phần này đưa ra danh mục các lệnh các hàm của MATLAB có thể sử dụng trong phần thí nghiệm này. Để biết cụ thể hơn về chức năng của hàm và cú pháp của lệnh gọi hàm, gõ lệnh **help** kèm theo tên của hàm tại cửa sổ lệnh của MATLAB.

- **zeros:** tạo một ma trận với toàn bộ các phần tử có giá trị bằng 0.
- **ones:** tạo một ma trận với toàn bộ các phần tử có giá trị bằng 1.
- **rand:** tạo một ma trận với các phần tử nhận các giá trị ngẫu nhiên được phân bố đều trong khoảng từ 0 đến 1.
- **randn:** tạo một ma trận với các phần tử nhận các giá trị ngẫu nhiên theo phân bố Gauss có giá trị trung bình bằng 0, phương sai bằng 1.
- **min:** trả về giá trị nhỏ nhất trong một ma trận.
- **max:** trả về giá trị lớn nhất trong một ma trận.
- **fliplr:** lộn ngược lại thứ tự các phần tử trong một ma trận theo hướng xuất phát từ phải qua trái trở thành từ trái qua phải.
- **plot và stem**: vẽ đồ thị của một dãy số, plot để thể hiện dạng liên tục, stem để thể hiện dạng rời rạc, thường sử dụng hàm stem để vẽ tín hiệu ở miền n.
- **conv:** trả về tích chập của 2 vector.
- **filter:** trả về đáp ứng theo thời gian của hệ thống được mô tả bởi một phương trình sai phân tuyến tính hệ số hằng.

Ngoài ra, sinh viên cần tìm hiểu một cách rất cẩn thận các phép toán trên ma trận và ***vector*** trong phần trợ giúp ***(Help)*** của MATLAB**.**

2.3. Thực hiện các bài Lab.

Lab 1: Tạo các dãy xung đơn vị và dãy nhảy đơn vị theo chương trình mẫu bằng cách gõ các dòng lệnh cho ở 2 bảng dưới đây vào cửa sổ soạn thảo (Editor) và ghi lại theo các tên tệp lần lượt là **impseq.m** và **stepseq.m**:

- ***Dãy xung đơn vị:***

```
function [x,n] = impseq(n0,n1,n2)
%Tao ra day x(n) = delta(n-n0); n1 <= n <= n2
%[x,n] = impseq(n0,n1,n2)
n = [n1:n2]; x = [(n-n0) == 0];
```

- ***Dãy nhảy đơn vị:***

```
function [x,n] = stepseq(n0,n1,n2)
%Tao ra day x(n) = u(n-n0); n1 <= n <= n2
%[x,n] = stepseq(n0,n1,n2)
n = [n1:n2]; x = [(n-n0) >= 0];
```

Lab 2: Viết chương trình tạo dãy hàm mũ thực với các tham số đầu vào và đầu ra được nhập theo câu lệnh:

[x,n] = expseq(a,n1,n2) ***Chú ý:*** tham số a có thể thực hoặc phức.

Kết quả Lab 2:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Lab 3: Viết chương trình tạo một dãy thực ngẫu nhiên xuất phát từ ***n1*** đến ***n2*** và có giá trị của biên độ theo phân bố ***Gauss*** với trung bình bằng ***0***, phương sai bằng ***1***. Các tham số đầu vào và đầu ra được nhập theo câu lệnh:

[x,n] = randnseq(n1,n2)

Kết quả Lab 3:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Lab 4: Tạo các hàm cộng 2 dãy và nhân 2 dãy với các chỉ số đầu và chỉ số cuối của hai dãy tương ứng khác nhau, hàm tạo trễ và hàm biến số n đảo theo chương trình mẫu bằng cách gõ các dòng lệnh cho ở 4 bảng dưới đây vào cửa số soạn thảo (Editor) và ghi lại theo các tên tệp lần lượt là ***sigadd.m, sigmult.m, sigshift.m,*** và ***sigfold.m***:

- ***Cộng 2 dãy:***

```
function [y,n] = sigadd(x1,n1,x2,n2)
%Thuc hien y(n) = x1(n)+x2(n)
%[y,n] = sigadd(x1,n1,x2,n2)
% y = day tong co vector chi so n
%x1 = day thu nhat co vector chi so n1
%x2 = day thu hai co vector chi so n2 (n2 co the khac n1)
n = min(min(n1),min(n2)):max(max(n1),max(n2));
y1 = zeros(1,length(n)); y2 = y1;
y1(find((n>=min(n1))&(n<=max(n1))==1)) = x1;
y2(find((n>=min(n2))&(n<=max(n2))==1)) = x2;
y = y1+y2;
```

- ***Nhân 2 dãy:***

```
function [y,n] = sigmult(x1,n1,x2,n2)
%Thuc hien y(n) = x1(n)*x2(n)
%-----------------------------------------------------
% y = day tich co vector chi so n
%x1 = day thu nhat co vector chi so n1
%x2 = day thu hai co vector chi so n2 (n2 co the khac n1)
n = min(min(n1),min(n2)):max(max(n1),max(n2));
y1 = zeros(1,length(n)); y2 = y1;
y1(find((n>=min(n1))&(n<=max(n1))==1)) = x1;
y2(find((n>=min(n2))&(n<=max(n2))==1)) = x2;
y = y1.*y2;
```

- ***Trễ (dịch):***

```
function [y,n] = sigshift(x,m,n0)
%Thuc hien y(n) = x(n-n0)
%-------------------------------------------
%[y,n] = sigshift(x,m,n0)
n = m + n0; y = x;
```

- ***Biến số n đảo:***

```
function [y,n] = sigfold(x,n)
%Thuc hien y(n) = x(-n)
%-------------------------------------------
%[y,n] = sigfold(x,n)
y = fliplr(x); n = -fliplr(n);
```

Kết quả Lab 4:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Lab 5: Vẽ đồ thị dãy $x(n) = 2\delta(n+2) - 2\delta(n-4), -5 \leq n \leq 5$ theo chương trình mẫu bằng cách gõ các dòng lệnh theo bảng dưới đây vào cửa sổ soạn thảo (Editor) và ghi lại theo tên tệp ***Lab5***.

```
n = [-5:5];
x = 2*impseq(-2,-5,5) - impseq(4,-5,5);
stem(n,x);
title('Day so theo dau bai 1.5');
xlabel('n'); ylabel('x(n)');
```

Kết quả Lab 5:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Lab 6 : Tạo hàm tính tích chập có tên ***conv_m*** thực hiện việc tính tích chập của hai dãy, mà mỗi dãy được thể hiện bởi 2 vector, một vector thể hiện chỉ số, một vector thể hiện giá trị của dãy. Ghi lại theo tên tệp ***conv_m.m***.

```
function [y,ny] = conv_m(x,nx,h,nh)
%Ham tinh tich chap da duoc sua doi danh cho xu ly so tin hieu
%[y,ny] = conv_m(x,nx,h,nh)
%[y,ny] = day ket qua
%[x,nx] = day thu nhat
%[h,nh] = day thu hai
nyb = nx(1)+nh(1); nye = nx(length(x))+nh(length(h));
ny = [nyb:nye];
y = conv(x,h);
```

Kết quả Lab 6:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Lab 7: Viết chương trình thể hiện trên đồ thị kết quả hàm tự tương quan của dãy sau:

$$y(n) - y(n-1) + 0.9y(n-2) = x(n)$$

Kết quả Lab 7:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Lab 8: Cho hệ thống mô tả bởi phương trình sai phân tuyến tính hệ số hằng như sau:

$$y(n) - y(n-1) + 0.9y(n-2) = x(n)$$

Viết chương trình sử dụng hàm *filter* của MATLAB thực hiện các công việc sau:

a. Biểu diễn bằng đồ thị hàm đáp ứng xung đơn vị của hệ thống với $-40 \leq n \leq 80$

b. Biểu diễn bằng đồ thị dãy đáp ứng của hệ thống với $\boldsymbol{-40 \leq n \leq 80}$ khi dãy đầu vào là dãy nhảy đơn vị.

Kết quả Lab 8:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

BÀI 3: TÍN HIỆU VÀ HỆ THỐNG RỜI RẠC Ở MIỀN Z, MIỀN TẦN SỐ LIÊN TỤC ω, VÀ MIỀN TẦN SỐ RỜI RẠC k

3.1. Cơ sở lý thuyết.

Tất cả các hệ thống được xét đến trong môn học Xử lý số tín hiệu đều là Hệ thống tuyến tính bất biến. Có một số cách thức để phân tích một tín hiệu thành tổ hợp tuyến tính của các tín hiệu thành phần. Trong những cách đó, lựa chọn tín hiệu thành phần là các hàm xung đơn vị tại các thời điểm khác nhau là một ví dụ điển hình. Một hệ thống tương đương với toán tử **T** tác động lên dãy ***x(n)*** tại đầu vào sẽ có dãy đáp ứng ra ***y(n)*** là:

$$y(n)=T[x(n)]=T\left[\sum_{k=-\infty}^{\infty}x(k)\delta(n-k)\right]=\sum_{k=-\infty}^{\infty}x(k)T[\delta(n-k)]=\sum_{k=-\infty}^{\infty}x(k)h(n-k)=x(n)*h(n)$$

với, $h(n)=T[\delta(n)]$ là đáp ứng xung của hệ thống.

Công cụ để thực hiện việc phân tích trên là biến đổi Fourier, một phép biến đổi biến một dãy số rời rạc theo thời gian thành một hàm số phức với biến số thực liên tục, tuần hoàn ở miền tần số.

Phép biến đổi Fourier cho dãy *số* ***x(n),*** với ***x(n)*** thoả mãn điều kiện $\sum_{n=-\infty}^{\infty}|x(n)|<\infty$

$$X(e^{j\omega})=FT[x(n)]=\sum_{n=-\infty}^{\infty}x(n)e^{-j\omega n}$$

Biến đổi Fourier ngược đối với hàm $X(e^{j\omega})$:

$$x(n)=IFT[X(e^{j\omega})]=\frac{1}{2\pi}\int_{-\pi}^{\pi}X(e^{j\omega})e^{-j\omega n}d\omega$$

$X(e^{j\omega})$: là hàm phức với biến số thực nên nó thường được thể hiện bởi 2 thành phần phổ biên độ và phổ pha dưới dạng sau đây:

$$X(e^{j\omega})=|X(e^{j\omega})|e^{j\arg[X(e^{j\omega})]}=|X(e^{j\omega})|e^{j\varphi(\omega)}$$

- $|X(e^{j\omega})|$: Là phổ biên độ của tín hiệu x(n).
- $\arg[X(e^{j\omega})]=\varphi(\omega)$: Là phổ pha của tín hiệu x(n).

Khi quan tâm đến các thành phần tần số của một tín hiệu, ta cần quan tâm đến hàm phổ biên độ và hàm phổ pha của tín hiệu đó đối với các tần số. Có hai điểm cần lưu ý đối với biểu diễn tín hiệu ở miền tần số:

- Do x(n) là rời rạc nên $\mathbf{X(e^{j\omega})}$ là hàm tuần hoàn chu kỳ 2π theo biến số $\boldsymbol{\omega}$.

- Do tính chất đối xứng của phép biến đổi Fourier nên nếu dãy ***x(n)*** là thực thì hàm $X(e^{j\omega})$ có tính chất đối xứng Hermit ***(Hermitian Symmetric)***, điều này có nghĩa phổ biên độ là một hàm thực chẵn và phổ pha là một hàm thực lẻ.

Hai tính chất trên nói lên rằng nếu **x(n)** là một dãy tín hiệu thực thì chỉ cần khảo sát hàm $X(e^{j\omega})$ trong phạm vi $\boldsymbol{\pi \le \omega \le 0}$ là đã có đầy đủ thông tin về toàn bộ hàm $X(e^{j\omega})$ với $-\infty \le \omega \le \infty$. Trên thực tế khi xem xét đồ thì phổ biên độ và phổ pha của tín hiệu, chúng ta thường thể hiện đồ thị trong một vài chu kỳ tuần hoàn.

MATLAB, cũng như mọi phần mềm hỗ trợ tính toán và các ngôn ngữ lập trình khác không có khả năng tính toán trực tiếp cũng như thể hiện một hàm số với biến số liên tục biến thiên từ -∞ đến ∞. Điều này có nghĩa MATLAB không thể trực tiếp tính $X(e^{j\omega})$ từ ***x(n).*** Tuy nhiên, nếu biết được biểu thức của hàm ảnh của tín hiệu qua phép biến đổi Fourier (***hàm phổ của tín hiệu***), ta có thể tính các giá trị của hàm phổ tín hiệu tại các điểm rời rạc trong một khoảng nào đó và thể hiện gần đúng trên đồ thị phổ biên độ và phổ pha của tín hiệu gốc.

Trong trường hợp x(n) là một dãy có chiều dài hữu hạn, ta có thể tính gần đúng $X(e^{j\omega})$ tại M+1 giá trị gần đúng trong khoảng [0,π] theo nguyên tắc sau:

- Do ***x(n)*** chỉ xác định trong một khoảng hữu hạn $\boldsymbol{0 \le n \le N-1}$ nên:

$$X\left(e^{j\omega}\right) = FT\left[x(n)\right] = \sum_{n=0}^{N-1} x(n)e^{-j\omega n}$$

- Khi lấy M+1 điểm rời rạc cách đều nhau trong khoảng [0,π], biến liên tục $\boldsymbol{\omega}$

trở thành biến rời rạc $\boldsymbol{\omega_k}$ với $\omega_k = \frac{\pi}{M}k$, k = 0, 1, ..., M.

- Giá trị của $X(e^{j\omega})$ tại các điểm rời rạc là:

$$X\left(e^{j\omega_k}\right) = \sum_{n=-\infty}^{\infty} x(n)e^{-j\frac{\pi}{M}kn}$$

- Công thức trên có thể viết dưới dạng phương trình ma trận như sau:

$$\begin{bmatrix} X\left(e^{j\omega_0}\right) \\ X\left(e^{j\omega_1}\right) \\ \vdots \\ X\left(e^{j\omega_M}\right) \end{bmatrix} = \begin{bmatrix} e^{-j\frac{\pi}{M}00} & e^{-j\frac{\pi}{M}01} & \cdots & e^{-j\frac{\pi}{M}0(N-1)} \\ e^{-j\frac{\pi}{M}10} & e^{-j\frac{\pi}{M}11} & & e^{-j\frac{\pi}{M}1(N-1)} \\ \vdots & & & \\ e^{-j\frac{\pi}{M}M0} & e^{-j\frac{\pi}{M}M1} & & e^{-j\frac{\pi}{M}M(N-1)} \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(N-1) \end{bmatrix}$$

lấy chuyển vị của cả hai vế, phương trình trên trở thành

$$\begin{bmatrix} X(0) & X(1) & \cdots & X(M) \end{bmatrix} = \begin{bmatrix} x(0) & x(1) & \cdots & x(N-1) \end{bmatrix} \begin{bmatrix} e^{-j\frac{\pi}{M}00} & e^{-j\frac{\pi}{M}10} & \cdots & e^{-j\frac{\pi}{M}(N-1)0} \\ e^{-j\frac{\pi}{M}01} & e^{-j\frac{\pi}{M}11} & \cdots & e^{-j\frac{\pi}{M}(N-11)} \\ \vdots & \vdots & & \vdots \\ e^{-j\frac{\pi}{M}0(N-1)} & e^{-j\frac{\pi}{M}1(N-1)} & \cdots & e^{-j\frac{\pi}{M}M(N-1)} \end{bmatrix}$$

Đoạn chương trình sau nhằm thực hiện việc tính giá trị của hàm $X(e^{j\omega})$ của dãy ***x(n)*** có chiều dài hữu hạn từ ***n1*** đến ***n2*** với ***M+1*** giá trị rời rạc trong khoảng [0,π]:

```
>> k = [0:M]; n=[n1:n2];
>> X = x * exp(-j*pi/M) .^ (n'*k);
```

Dù cho việc phân tích tín hiệu và hệ thống bằng phép biến đổi Fourier là thuận tiện và rất hữu ích trong rất nhiều trường hợp, công cụ này đôi khi cũng gặp một số khó khăn:

- Một số dãy tín hiệu trong thực tế ví dụ như u(n) và nu(n) là không có biến đổi Fourier, dẫn đến không phân tích được các thành phần tần số của tín hiệu.
- Đáp ứng của hệ thống trong thời gian quá độ gây bởi điều kiện đầu của hệ thống hoặc đột ngột thay đổi dạng tín hiệu dãy đầu vào là không khảo sát được bằng biến đổi Fourier.

Phép biến đổi Z cho phép chúng ta có thể giải quyết được bài toán trong các trường hợp như vậy. Định nghĩa phép biến đổi ***Z*** cho dãy số ***x(n)*** là:

$$X(z) = ZT[x(n)] = \sum_{n=-\infty}^{\infty} x(n) z^{-n}$$

X(z) là một hàm phức với biến số (độc lập) phức. Tập các giá trị z để chuỗi hàm bên tay phải của biểu thức trên hội tụ về một hàm số, hay nói một cách khác để ***X(z)*** tồn tại gọi là miền hội tụ ***RC (Region of Convergence)*** của biến đổi ***Z***. Có thể chứng tỏ được rằng, trong trường hợp tổng quát miền hội tụ của biến đổi **Z** của một dãy số nằm bên trong một hình vành khuyên $R_{x-} < z < R_{x+}$, với R_{x-} và R_{x+} là các số thực dương.

Biến đổi Z ngược đối với hàm X(z):

$$x(n) = ZT[X(z)] = \frac{1}{2\pi} \oint_C X(z) z^{-n} dz$$

Với, C là một đường cong kín lấy theo chiều ngược chiều kim đồng hồ, bao quanh gốc toạ độ và nằm hoàn toàn trong miền hội tụ của ***X(z) (RC[X(z)]).***

Trên thực tế, phương pháp được sử dụng trong hầu hết các trường hợp tìm biến đổi Z ngược của một hàm phân thức hữu tỷ ***X(z)*** là phân tích thành tổng của các phân thức đơn giản. Hàm ***residuez*** của MATLAB cho phép nhanh chóng tìm ra các điểm cực và các hệ số trong khai triển ứng với các điểm cực đó của một hàm phân thức hữu tỷ ***X(z)***.

Trong trường hợp đường tròn đơn vị nằm trong miền hội tụ của biến đổi Z thì biến đổi Fourier chính là biến đổi Z đánh giá trên đường tròn đơn vị.

Đối với một hệ thống, hàm truyền đạt H(z) của hệ thống được định nghĩa là biến đổi Z của hàm đáp ứng xung:

$$H(z) = ZT[h(n)] = \sum_{n=-\infty}^{\infty} h(n)z^{-n}$$

Hàm truyền đạt của hệ thống chính là tỷ số giữa biến đổi Z của tín hiệu đầu ra trên biến đổi Z của tín hiệu đầu vào:

$$H(z) = \frac{Y(z)}{X(z)}$$

Như ở phần trước đã đề cập tất cả các hệ thống tuyến tính bất biến có thể thực hiện được đều được mô tả bởi phương trình sai phân tuyến tính hệ số hằng. Các hệ thống này có ảnh của đáp ứng xung qua phép biến đổi Z đều có dạng phân thức hữu tỷ mà tử số và mẫu số là các đa thức theo ***z*** (hoặc z^{-1}). Các điểm không, tại đó giá trị của ***X(z)*** bằng 0, chính là các nghiệm của tử số. Các điểm cực, tại đó giá trị của ***X(z)*** tiến tới vô cùng, chính là các nghiệm của mẫu số. Sự phân bố các điểm cực và điểm không của biến đổi ***Z*** đối với một tín hiệu, hoặc hàm truyền đạt của hệ thống, quyết định đến toàn bộ các tính chất của tín hiệu hay hệ thống được xét đến.

Hai phép biến đổi nói trên, biến đổi ***Fourier*** và biến đổi ***Z***, về bản chất là biến đổi một dãy số trở thành một hàm phức với biến số thực, đối với biến đổi Fourier, và một hàm phức với biến số phức, đối với biến đổi Z. Các miền mới được xét đến là miền ω và miền Z. Đặc điểm chung của các hàm số trên hai miền mới là hàm số với biến số liên tục, do đó, MATLAB cũng như tất cả các ngôn ngữ lập trình và công cụ phần mềm hỗ trợ bằng máy tính không thể tính toán chính xác toàn bộ hàm số ảnh của các phép biến đổi nói trên, thay vì đó ta chỉ thu được kết quả gần đúng tại các điểm rời rạc.

Biến đổi Fourier rời rạc, ứng dụng trên dãy tuần hoàn và dãy có chiều dài hữu hạn là phép biến đổi cho phép máy tính tìm được chính xác mọi giá trị của hàm ảnh của phép biến đổi tại tất cả các biến của hàm số bởi hàm ảnh là hàm trên miền rời rạc, miền này gọi là miền ***k***. Công thức biến đổi ***Fourier rời rạc*** cho một dãy số ***x(n)*** có ***chiều dài hữu hạn*** từ ***0*** đến ***N-1*** được cho như sau:

$$X(k)_N = DFT[x(n)_N] = \sum_{n=0}^{N-1} x(n)e^{-j\frac{2\pi}{N}kn} = \sum_{n=0}^{N-1} x(n)W_N^{kn}$$

Công thức biến đổi Fourier rời rạc ngược đối với dãy $X(k)_N$ là:

$$x(n)_N = IDFT[X(k)_N] = \frac{1}{N}\sum_{k=0}^{N-1} X(k)e^{j\frac{2\pi}{N}kn} = \frac{1}{N}\sum_{k=0}^{N-1} X(k)W_N^{-kn}$$

Với, $W_N = e^{-j\frac{2\pi}{N}}$ dẫn đến $W_N^{kn} = e^{-j\frac{2\pi}{N}kn}$, $W_N^{-kn} = e^{j\frac{2\pi}{N}kn}$, x(n) và X(k) chỉ khác 0 trong khoảng từ 0 đến ***N-1***.

Dưới dạng ma trận các công thức trên được thể hiện:

$$[X] = [W_N][x] \text{ và } [x] = [W_N]^{-1}[X] = \frac{1}{N}[W_N]^*[X]$$

với X, x, và WN là các vector và ma trận được định nghĩa:

$$[X] = \begin{bmatrix} X(0) \\ X(1) \\ \vdots \\ X(N-1) \end{bmatrix}, [W_N] = \begin{bmatrix} W_N^{00} & W_N^{01} & \cdots & W_N^{0(N-1)} \\ W_N^{10} & W_N^{11} & \cdots & W_N^{1(N-1)} \\ \vdots & \vdots & & \vdots \\ W_N^{(N-1)0} & W_N^{(N-1)1} & \cdots & W_N^{(N-1)(N-1)} \end{bmatrix}, \text{ và } [x] = \begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(N-1) \end{bmatrix}$$

Chúng ta hoàn toàn có thể xây dựng thuật toán biến đổi ***Fourier rời rạc thuận*** và ***ngược*** một cách trực tiếp xây dựng từ công thức nhân ma trận trên, giống như thuật toán tính gần đúng của biến đổi Fourier đã được đề cập đến ở đầu phần tóm tắt lý thuyết này. Tuy nhiên, số phép tính để tính toán là rất lớn, tương đương với NxN phép nhân trên số phức và ***N(N-1)*** phép cộng trên số phức cho biến đổi Fourier rời rạc đối với dãy có độ dài là N mẫu. Năm 1965, Cooley và Turkey đã đưa ra một thuật toán rút gọn số lượng phép tính trong biến đổi Fourier đi rất nhiều. Thuật toán này được biết đến với tên gọi biến đổi Fourier nhanh (***FFT***). Ý tưởng của thuật toán này cũng có thể áp dụng cho phép tính biến đổi Fourier gần đúng trên M+1 điểm rời rạc trong khoảng [0,π].

Hàm **fft** của MATLAB cho phép thực hiện việc biến đổi Fourier rời rạc theo thuật toán biến đổi Fourier nhanh. Hàm **fft** được viết bằng ngôn ngữ máy chứ không phải bằng ngôn ngữ MATLAB nên nó quá trình thực hiện biến đổi Fourier rời rạc được tiến hành rất nhanh. Nếu N là luỹ thừa của 2, hàm **fft** sẽ giải quyết bài toán theo thuật toán cơ số 2.

3.2. Một số lệnh và hàm của MATLAB.

Phần này đưa ra danh mục các lệnh các hàm của MATLAB có thể sử dụng trong phần thí nghiệm này. Để biết cụ thể hơn về chức năng của hàm và cú pháp của lệnh gọi hàm, gõ lệnh ***help*** kèm theo tên của hàm tại cửa số lệnh của MATLAB.

- **abs, angle:** trả về các hàm thể hiện Modun và Agumen của một số phức.
- **real, imag:** trả về các hàm thể hiện phần thực và phần ảo của một số phức.
- **residuez:** trả về các điểm cực và các hệ số tương ứng với các điểm cực đó trong phân tích một hàm phân thức hữu tỷ ở miền Z thành các thành phần là các hàm phân thức đơn giản, ngược lại nếu đầu vào là danh sách các điểm cực và các hệ số, hàm ***residuez*** sẽ trả về hàm phân thức hữu tỷ ở miền Z.
- **poly:** xây dựng một đa thức từ danh sách các nghiệm của nó.

- **ztrans:** trả về biến đổi Z của một hàm số được định nghĩa theo công thức của một biểu tượng (symbol).
- **iztrans:** hàm ngược lại của hàm ztrans.
- **zplane:** thể hiển phân bố điểm cực và điểm không của một hàm phân thức hữu tỷ lên mặt phẳng Z.
- **freqz:** trả về đáp ứng tần số của một hệ thống tại một số hữu hạn các điểm rời rạc trên vòng tròn đơn vị khi biết hàm truyền đạt của nó.
- **fft:** thực hiện biến đổi Fourier rời rạc của một dãy số có độ dài hữu hạn theo thuật toán biến đổi Fourier nhanh và trả về kết quả biến đổi Fourier rời rạc của dãy số đó.
- **clock:** trả về thời gian thực hiện tại.
- **etime:** trả về thời gian tính bằng giây giữa 2 thời điểm.

3.3. Thực hiện các bài Lab.

Lab 1: Cho dãy $x(n) = 0.5^n u(n)$

a. Dựa trên định nghĩa của biến đổi Z, tìm biến đổi Z của dãy trên.

b. Kiểm chứng lại kết quả câu a bằng hàm ***ztrans***.

c. Từ kết quả trên, tìm biến đổi ***Fourier*** của ***x(n).***

d. Dùng MATLAB thể hiện trên đồ thị phổ $\boldsymbol{X(e^{j\omega})}$ tại 501 điểm rời rạc trong khoảng [0,π] theo chương trình mẫu bằng cách gõ các dòng lệnh theo bảng dưới đây vào cửa số soạn thảo (Editor) và ghi lại theo tên tệp ***Lab1_3.3***.

```
w = [0:1:500]*pi/500;
X = exp(j*w) ./ (exp(j*w)- 0.5*ones(1,501));
magX = abs(X); angX = angle(X);
realX = real(X); imagX = imag(X);
subplot(2,2,1); plot(w/pi,magX); grid;
title('Magnitude Part'); xlabel('frequency in pi units');
ylabel('Magnitude');
subplot(2,2,3); plot(w/pi,angX); grid;
title('Angle Part'); xlabel('frequency in pi units');
ylabel('Radians');
subplot(2,2,2); plot(w/pi,realX); grid;
title('Real Part'); xlabel('frequency in pi units');
ylabel('Real');
subplot(2,2,4); plot(w/pi,imagX); grid;
title('Imaginary Part'); xlabel('frequency in pi units');
ylabel('Imaginary');
```

Kết quả Lab 1:

Lab 2: Cho phổ $\mathbf{X(e^{j\omega})}$ có dạng sau: $Xe^{j\omega} = e^{-j\frac{\omega}{2}}sin3\omega$

Viết chương trình thể hiện trên đồ thị các hàm phổ biên độ, phổ pha, phần thực và phần ảo của $\mathbf{X(e^{j\omega})}$, tính tại 2001 điểm rời rạc trong khoảng [-2π,2π].

Kết quả Lab 2:

Lab 3: Cho dãy x(n) có dạng như sau: x(n) {...,-2, 0, 1, 4, 3, 1, 5, 0, 2,...}
↑

Đây là một dãy số xác định trong một khoảng hữu hạn từ -1 đến 3. Tính và thể hiện phổ của dãy ***x(n)*** tại 501 điểm rời rạc trong khoảng [0,π] theo chương trình mẫu. Gõ các dòng lệnh theo bảng dưới đây vào cửa số soạn thảo và ghi lại theo tên tệp ***Lab3_3.3***.

```
n = -1:3; x = 1:5;
k = 0:500; w = (pi/500)*k;
X = x*(exp(-j*pi/500)).^(n'*k);
magX = abs(X); angX = angle(X);
realX = real(X); imagX = imag(X);
subplot(2,2,1); plot(k/500,magX); grid;
title('Magnitude Part'); xlabel('frequency in pi units');
ylabel('Magnitude');
subplot(2,2,3); plot(k/500,angX); grid;
title('Angle Part'); xlabel('frequency in pi units');
ylabel('Radians');
subplot(2,2,2); plot(k/500,realX); grid;
title('Real Part'); xlabel('frequency in pi units');
ylabel('Real');
subplot(2,2,4); plot(k/500,imagX); grid;
title('Imaginary Part'); xlabel('frequency in pi units');
ylabel('Imaginary');
```

Gõ lệnh ***Lab3_3.3*** tại cửa sổ lệnh để chạy kịch bản nói trên và xem các đồ thị.

Kết quả Lab 3:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Lab 4: Một hàm ở miền Z được cho với công thức sau đây:

$$X(z) = \frac{z}{3z^2 - 4z + 1}$$

a. Sử dụng lệnh ***residuez*** của MATLAB, tính các điểm cực, thặng dư tại các điểm cực theo chương trình mẫu bằng cách gõ các dòng lệnh theo bảng dưới đây vào cửa số soạn thảo và ghi lại theo tên tệp ***Lab4_3.3***.

```
b = [0 1]; a = [3 -4 1];
[R,p,C] = residuez(b,a)
% -------------------------
[b a] = residuez(R,p,C)
```

Từ đó hãy viết dạng tổng các hàm phân thức đơn giản của ***X(z).***

b. Từ kết quả câu trên, viết công thức khai triển ***X(z)*** thành tổng các phân thức đơn giản, từ đó tìm biến đổi ***Z*** ngược của X(z) trên miền sao cho x(n) là một dãy nhân quả.

c. Kiểm chứng lại kết quả câu b bằng hàm ***iztrans.***

Kết quả Lab 4:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Lab 5: Cho hàm X(z) với công thức như sau:

$$X(z) = \frac{1}{(1 - 0.9z^{-1})^2(1 - 0.9z^{-1})}$$

a. Viết chương trình tính các điểm cực, thặng dư của các điểm cực của hàm X(z) trên (**Hint:** có thể dùng hàm poly của MATLAB để khôi phục lại đa thức mẫu số từ một mảng các nghiệm của đa thức - mảng các điểm cực của X(z)).

b. Từ kết quả câu trên, viết công thức khai triển X(z) thành tổng các phân thức đơn giản, từ đó tìm biến đổi Z ngược của X(z) trên miền $|z| > 0.9$.

Kết quả Lab 5:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Lab 6: Cho hệ thống nhân quả biểu diễn bởi phương trình sau:

$$y(n) - 0.9y(n-1) = x(n)$$

a. Tìm hàm truyền đạt của hệ thống.

b. Dùng lệnh **zplane** của MATLAB biểu diễn trên đồ thị mặt phẳng Z sự phân bố các điểm cực và điểm không.

```
b = [1 0]; a = [1 -0.9];
% Tim phan bo diem cuc va diem khong
subplot(1,2,1);
zplane(b,a);
title('Z plane');
% Tim dap ung tan so bang cach danh gia 200 diem roi rac
% cua H(z) tren duong tron don vi
[H, w] = freqz(b,a,200,'whole');
magH = abs(H(1:101)); phaH= angle(H(1:101));
% Ve dap ung tan so
subplot(2,2,2); plot(w(1:101)/pi,magH); grid;
title('Magnitude Response');
xlabel('frequency in pi units');
ylabel('Magnitude');
subplot(2,2,4); plot(w(1:101)/pi,phaH/pi); grid;
title('Phase Response');
xlabel('frequency in pi units');
ylabel('Phase in pi units');
```

c. Dùng lệnh **freqz** tính và biểu diễn trên đồ thị hàm đáp ứng tần số $\mathbf{H(e^{j\omega})}$ của hệ thống (bao gồm đáp ứng biên độ - tần số và đáp ứng pha - tần số) tại 200 điểm rời rạc trên đường tròn đơn vị theo chương trình mẫu bằng cách gõ các dòng lệnh theo bảng trên vào cửa số soạn thảo và ghi lại theo tên tệp ***Lab6_3_3***.

Kết quả Lab 6:

Lab 7: Cho hệ thống nhân quả biểu diễn bởi phương trình sau:

$$y(n) - 0.81y(n-2) = x(n) - x(n-2)$$

a. Viết công thức hàm truyền đạt H(z) của hệ thống.

Viết các chương trình bằng MATLAB thực hiện các công việc sau:

b. Tính vị trí các điểm cực, các hệ số trong khai triển H(z) thành tổng các phân thức đơn giản.

c. Biểu diễn phân bố điểm cực và điểm không trên mặt phẳng Z.

d. Tính và biểu diễn trên đồ thị hàm đáp ứng tần số $H(e^{j\omega})$ của hệ thống (bao gồm đáp ứng biên độ - tần số và đáp ứng pha - tần số) tại 200 điểm rời rạc trên đường tròn đơn vị.

Từ kết quả thu được ở câu b. Tìm hàm đáp ứng xung h(n) của hệ thống.

Kết quả Lab 7:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Lab 8: Tạo các hàm thực hiện việc biến đổi Fourier rời rạc thuận và Fourier rời rạc ngược theo chương trình mẫu bằng cách gõ các dòng lệnh cho ở 2 bảng dưới đây vào cửa số soạn thảo và ghi lại theo các tên tệp lần lượt là ***dft.m***, và ***idft.m***:

- **Tìm biến đổi Fourier rời rạc thuận:**

```
function [Xk] = dft(xn,N)
% Tim bien doi Fourier roi rac thuan
% ------------------------------------------
% [Xk] = dft(xn,N)
% Xk = day cac he so DFT tren doan 0<=k<=N-1
% xn = day huu han N diem
%  N = chieu dai DFT
n = [0:1:N-1];
k = [0:1:N-1];
WN = exp(-j*2*pi/N);
nk = n' * k;
WNnk = WN .^ nk;        % ma tran DFT
Xk = xn * WNnk;
```

➢ **Tìm biến đổi Fourier rời rạc ngược:**

```
function [xn] = idft(Xk,N)
% Tim bien doi Fourier roi rac nguoc
% ------------------------------------------
% [xn] = idft(Xk,N)
% xn = day co chieu dai huu han tren doan 0<=n<=N-1
% Xk = day cac he so DFT tren doan 0<=k<=N-1
%  N = chieu dai DFT
% ------------------------------------------
n = [0:1:N-1];
k = [0:1:N-1];
WN = exp(-j*2*pi/N);
nk = n' * k;
WNnk = WN .^ (-nk);        % ma tran IDFT
xn = (Xk * WNnk)/N;
```

Lab 9: Dựa trên các hàm **dft** được xây dựng ở **Lab 8**, tìm biến đổi Fourier rời rạc của dãy có chiều dài N=20:

$$x(n) = \begin{cases} 1 & 0 \le n \le 4 \\ 0 & n \text{ còn lại} \end{cases}$$

Kết quả Lab 9:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

BÀI 4: THIẾT KẾ BỘ LỌC SỐ BẰNG MATLAB

4.1. THIẾT KẾ BỘ LỌC SỐ FIR.

4.1.1. Cơ sở lý thuyết.

Về mặt lý thuyết, dựa trên đặc điểm của đáp ứng tần số, Xử lý số tín hiệu quan tâm đến 4 loại bộ lọc lý tưởng sau đây:

1) Bộ lọc thông thấp lý tưởng.

Đáp ứng biên độ - tần số : $$\left|H_d\left(e^{j\omega}\right)\right| = \begin{cases} 1, & |\omega| \le \omega_c \\ 0, & n\ còn\ lại \end{cases}$$

Khi đó đáp ứng xung của bộ lọc thông thấp lý tưởng với trễ nhóm α là:

$$h_d(n) = \frac{\omega_c}{\pi} sinc\omega_c(n-\alpha)$$

2) Bộ lọc thông cao lý tưởng.

Đáp ứng biên độ - tần số : $$\left|H_d\left(e^{j\omega}\right)\right| = \begin{cases} 1, & |\omega| \ge \omega_c \\ 0, & n\ còn\ lại \end{cases}$$

Khi đó đáp ứng xung của bộ lọc thông cao lý tưởng với trễ nhóm α là:

$$h_d(n) = \delta(n-\alpha) - \frac{\omega_c}{\pi} sinc\omega_c(n-\alpha)$$

3) Bộ lọc thông dải lý tưởng

Đáp ứng biên độ - tần số: $$\left|H_d\left(e^{j\omega}\right)\right| = \begin{cases} 1, & \omega_{c1} \le |\omega| \le \omega_{c2} \\ 0, & n\ còn\ lại \end{cases}$$

Khi đó đáp ứng xung của bộ lọc thông dải lý tưởng với trễ nhóm α là:

$$h_d(n) = \frac{\omega_{c2}}{\pi} sinc\omega_{c2}(n-\alpha) - \frac{\omega_{c1}}{\pi} sinc\omega_{c1}(n-\alpha)$$

4) Bộ lọc chắn dải lý tưởng

Đáp ứng biên độ - tần số: $$\left|H_d\left(e^{j\omega}\right)\right| = \begin{cases} 0, & \omega_{c1} \le |\omega| \le \omega_{c2} \\ 1, & n\ còn\ lại \end{cases}$$

Khi đó đáp ứng xung của bộ lọc chắn dải lý tưởng với trễ nhóm α là:

$$h_d(n) = \delta(n-\alpha) - \frac{\omega_{c2}}{\pi} sinc\omega_{c2}(n-\alpha) - \frac{\omega_{c1}}{\pi} sinc\omega_{c1}(n-\alpha)$$

Ngoài ra, bộ vi phân và bộ biến đổi **Hilbert** cũng được xem xét đến :

➢ ***Bộ vi phân lý tưởng:***

Đáp ứng tần số: $$\left|H_d(e^{j\omega})\right| = \begin{cases} j\omega, & 0 < \omega \le \pi \\ -j\omega, & -\pi < \omega \le 0 \end{cases}$$

Khi đó đáp ứng xung của bộ vi phân lý tưởng với trễ nhóm α là:

$$h_d(n) = \begin{cases} \dfrac{cos\pi(n-\alpha)}{(n-\alpha)} & ,n \neq \alpha \\ 0 & ,n = \alpha \end{cases}$$

➢ ***Bộ biến đổi Hilbert.***

Đáp ứng tần số: $$\left|H_d(e^{j\omega})\right| = \begin{cases} -j, & 0 < \omega \leq \pi \\ j, & -\pi < \omega \leq 0 \end{cases}$$

Khi đó đáp ứng xung của bộ biến đổi Hilbert lý tưởng với trễ nhóm α là:

$$h_d(n) = \begin{cases} \dfrac{2}{\pi}\dfrac{sin^2\left[\dfrac{\pi(n-\alpha)}{2}\right]}{(n-\alpha)} & ,n \neq \alpha \\ 0 & ,n = \alpha \end{cases}$$

Chúng ta có nhận xét là đáp ứng xung của các bộ lọc lý tưởng nói trên có chiều dài vô hạn, xuất phát từ chỉ số -∞ đến +∞, và không nhân quả, dẫn đến không thể thực hiện được về mặt vật lý. Khi tổng hợp bộ lọc thực tế, ta phải chấp nhận đáp ứng xung phải xuất phát từ chỉ số 0 để đáp ứng điều kiện nhân quả. Khi đó, đáp ứng tần số của bộ lọc thực tế có phần quá độ từ dải thông đến dải chắn, hoặc ngược lại, và được gọi là dải chuyển tiếp (transition band). Đồng thời phải có sự gợn sóng (ripple) ở cả dải thông và dải chắn hoặc ít nhất tại một trong hai, dải thông hoặc dải chắn. Việc thiết kế bộ lọc là quá trình tìm ra các tham số, hay dãy đáp ứng xung của bộ lọc, thoả mãn các yêu cầu chỉ tiêu kỹ thuật cho trước, cụ thể là một số hoặc tất cả các tham số tuyệt đối sau:

- Tần số cắt dải thông $\boldsymbol{\omega_p}$.
- Tần số cắt dải thông $\boldsymbol{\omega_s}$.
- Bề rộng dải quá độ $\boldsymbol{\Delta\omega}$.
- Độ gợn sóng dải thông $\boldsymbol{\delta_1}$.
- Độ gợn sóng dải chắn $\boldsymbol{\delta_2}$.

Trên thực tế, các tham số thường được cho dưới dạng tương đối tính theo đơn vị **decibels** dưới dạng sau đây:

- Độ gợn sóng dải thông theo dB, được tính bằng công thức:

$$R_p = -20log\frac{1-\boldsymbol{\delta_1}}{1+\boldsymbol{\delta_1}}[dB]$$

- Độ suy giảm dải chắn theo dB được tính bằng công thức:

$$A_s = -20log\frac{\boldsymbol{\delta_2}}{1+\boldsymbol{\delta_1}}[dB]$$

Khi xem xét hàm đáp ứng tần số của bộ lọc, chỉ cần xem xét ω trong khoảng [0,π] là đủ. Với bộ lọc FIR ta luôn đặt được điều kiện pha tuyến tính, điều này có nghĩa đáp ứng

pha - tần số là một hàm số bậc nhất theo tần số ω, tương đương với thực hiện việc trễ hàm đáp ứng xung ở miền thời gian. Hàm đáp ứng pha - tần số của bộ lọc FIR được cho dưới dạng độ lớn và pha như sau:

$$H(e^{j\omega}) = A(e^{j\omega})e^{j\theta(\omega)}$$

Với, $\theta(\omega) = \beta - \alpha\omega$; $A(e^{j\omega})$: là hàm thực; α, β : là các hằng số.

Dựa trên tính chất đối xứng hay phản đối xứng của dãy đáp ứng xung và chiều dài N của dãy đáp ứng xung, người ta phân loại bộ lọc FIR làm 4 loại và đáp ứng tần số của bộ lọc FIR cho từng loại là như sau:

- **FIR loại 1:**

$$H(e^{j\omega}) = \left[\sum_{n=0}^{\frac{N-1}{2}} a(n)cosn\omega\right] e^{-j\frac{N-1}{2}\omega}$$

Trong đó, $a(0) = h\left(\frac{N-1}{2}\right)$, $h\left(\frac{N-1}{2}\right)$ là mẫu giữa của dãy đáp ứng xung.

$a(n) = 2h\left(\frac{N-1}{2} - n\right)$, với $1 \le n \le \frac{N-1}{2}$

Dẫn đến : $A(e^{j\omega}) = \sum_{n=0}^{\frac{N-1}{2}} a(n)cosn\omega$, và $\theta(\omega) = -\frac{N-1}{2}\omega$

- **FIR loại 2:**

$$H(e^{j\omega}) = \left[\sum_{n=0}^{\frac{N}{2}} b(n)cos\left(n - \frac{1}{2}\right)\omega\right] e^{-j\frac{N-1}{2}\omega}$$

Trong đó, $b(n) = 2h\left(\frac{N}{2} - n\right)$, với $1 \le n \le \frac{N}{2}$

Dẫn đến : $A(e^{j\omega}) = \sum_{n=0}^{\frac{N}{2}} b(n)cos\left(n - \frac{1}{2}\right)\omega$, và $\theta(\omega) = -\frac{N-1}{2}\omega$

- **FIR loại 3:**

$$H(e^{j\omega}) = \left[\sum_{n=1}^{\frac{N-1}{2}} c(n)\sin n\omega\right] e^{j\left(\frac{\pi}{2} - \frac{N-1}{2}\omega\right)}$$

Trong đó, $c(n) = 2h\left(\frac{N-1}{2} - n\right)$, với $1 \le n \le \frac{N-1}{2}$

Dẫn đến : $A(e^{j\omega}) = \sum_{n=0}^{\frac{N-1}{2}} c(n)\sin n\omega$, và $\theta(\omega) = \frac{\pi}{2} - \frac{N-1}{2}\omega$

- **FIR loại 4:**

$$H(e^{j\omega}) = \left[\sum_{n=1}^{\frac{N}{2}} d(n)\sin\left(n-\frac{1}{2}\right)\omega\right] e^{j\left(\frac{\pi}{2}-\frac{N-1}{2}\omega\right)}$$

Trong đó, $\mathrm{d}(n) = 2h\left(\frac{N}{2}-n\right)$, với $1 \le n \le \frac{N}{2}$

Dẫn đến : $A(e^{j\omega}) = \sum_{n=0}^{\frac{N-1}{2}} d(n)\sin\left(n-\frac{1}{2}\right)\omega$, và $\theta(\omega) = \frac{\pi}{2} - \frac{N-1}{2}\omega$

Có các phương pháp chính sau để tổng hợp bộ lọc FIR pha tuyến tính, đó là:

- Phương pháp cửa sổ.
- Phương pháp lấy mẫu tần số.

a. <u>Phương pháp cửa sổ.</u>

Phương pháp cửa sổ là tìm ra đáp ứng xung của bộ lọc lý tưởng rồi sau đó cắt xén ở hai đầu (hay nhân với một hàm cửa số) dãy đáp ứng xung đó sao cho ta thu được một bộ lọc FIR pha tuyến tính, đồng thời là nhân quả. Điểm nhấn mạnh ở phương pháp này là tìm ra đáp ứng xung thích hợp của bộ lọc lý tưởng và lựa chọn hàm cửa số thích hợp. Về mặt lý tưởng, bộ lọc thông thấp lý tưởng pha tuyến tính có độ lợi dải thông bằng 1 và đáp ứng tần số bằng 0 trên toàn dải chắn, tức là:

$$H_d(e^{j\omega}) = \begin{cases} e^{-j\alpha\omega}, |\omega| \le \omega_c \\ 0 \quad\quad , n\ còn\ lại \end{cases}$$

Với, $\omega_c$là tần số cắt và α là trễ nhóm nên : $h_d(n) = \frac{\omega_c}{\pi} sin\omega_c(n-\alpha)$ có tính chất đối xứng tại α.

Với các bộ lọc số lý tưởng khác, bao gồm thông cao, thông dải, và chắn dải, dãy đáp ứng xung cũng có dạng tương tự như vậy và có thể suy ra từ dạng đáp ứng xung của bộ lọc thông thấp lý tưởng nói trên. Với bộ vi phân và bộ biến đổi **Hilbert**, bằng biến đổi toán học, ta cũng có được đáp ứng xung có tính chất phản đối xứng tại α.

Một số cửa số thông dụng được lựa chọn là:

- **Cửa số chữ nhật :** $w_r(n) = rect_N(n) = \begin{cases} 1, & 0 \le n \le N-1 \\ 0, & n\ còn\ lại \end{cases}$
- **Cửa sổ Bartlet (hay cửa sổ tam giác):** $w(n) = \begin{cases} \frac{2n}{N-1} & ,0 \le n \le \frac{N-1}{2} \\ 2 - \frac{2n}{N-1} & ,\frac{N-1}{2} \le n \le N-1 \\ 0 & ,\ n\ còn\ lại \end{cases}$
- **Cửa sổ Hanning:** $w(n) = \begin{cases} 0.5 - 0.5cos\left(\frac{2\pi n}{N-1}\right), 0 \le n \le N-1 \\ 0 \quad\quad\quad\quad\quad\quad\quad\quad , \quad n\ còn\ lại \end{cases}$

- **Cửa số Hamming:** $w(n) = \begin{cases} 0.54 - 0.46cos\left(\frac{2\pi n}{N-1}\right), & 0 \le n \le N-1 \\ 0 & , \quad n\ còn\ lại \end{cases}$

- **Cửa sổ Blackman: cho đến hài bậc hai:**

$$w(n) = \begin{cases} 0.42 - 0.5cos\left(\frac{2\pi n}{N-1}\right) + 0.08cos\left(\frac{4\pi n}{N-1}\right) & , \ 0 \le n \le N-1 \\ 0 & , \quad n\ còn\ lại \end{cases}$$

- **Cửa sổ Kaiser:** $w(n) = \frac{I_0\left[\beta\sqrt{1-\left(1-\frac{2n}{N-1}\right)^2}\right]}{I_0[\beta]}$, *$I_0[]$* là hàm Bessel bậc không.

b. <u>Phương pháp lấy mẫu tần số.</u>

Phương pháp này là xây dựng một bộ lọc có đáp ứng xung chiều dài N và có đáp ứng tần số xấp xỉ với đáp ứng tần số của bộ lọc lý tưởng. Cụ thể, ta có thể xét tại N mẫu rời rạc cách đều nhau trong khoảng từ 0 đến 2π, hàm đáp ứng tần số của bộ lọc thực tế bằng đúng với hàm đáp ứng xung của bộ lọc lý tưởng. Đáp ứng tần số được viết dưới dạng độ lớn và pha:

$H(e^{j\omega}) = A(e^{j\omega})e^{j\theta(\omega)}$, với $A(e^{j\omega})$ và $\theta(\omega)$ là các hàm thực.

Ảnh của ***h(n)*** qua biến đổi Fourier rời rạc cũng được viết dưới dạng độ lớn và pha:

$H(k) = A(k)e^{j\theta(k)}$, với $A(k)$ và $\theta(k)$ là các hàm thực thì độ lớn và pha của dãy ***H(k)*** sẽ được tính theo công thức:

$$A(k) = A\left(e^{j\frac{2\pi}{N}k}\right) \text{ và } \theta(k) = \theta\left(\frac{2\pi}{N}k\right)$$

Do ***H($e^{j\omega}$)*** và ***H(k)*** đều có tính chất đối xứng ***Hermit*** nên: $A(k) = A(N-k), k = 1, \dots, N-1$.

Và :

- ***Đối với bộ lọc FIR loại 1:*** $\theta(k) = \begin{cases} -\frac{N-1}{2}\left[\frac{2\pi}{N}k\right] & , \ k = 1, \dots, \frac{N-1}{2} \\ +\frac{N-1}{2}\left[\frac{2\pi}{N}(N-k)\right] & , \ k = \frac{N+1}{2}, \dots, N-1 \end{cases}$

- ***Đối với bộ lọc FIR loại 2:*** $\theta(k) = \begin{cases} -\frac{N-1}{2}\left[\frac{2\pi}{N}k\right] & , \ k = 1, \dots, \frac{N-2}{2} \\ +\frac{N-1}{2}\left[\frac{2\pi}{N}(N-k)\right] & , \ k = \frac{N}{2}, \dots, N-1 \end{cases}$

- ***Đối với bộ lọc FIR loại 3:*** $\theta(k) = \begin{cases} \frac{\pi}{2} - \frac{N-1}{2}\left[\frac{2\pi}{N}k\right] & , \ k = 1, \dots, \frac{N-1}{2} \\ -\frac{\pi}{2} + \frac{N-1}{2}\left[\frac{2\pi}{N}(N-k)\right] & , \ k = \frac{N+1}{2}, \dots, N-1 \end{cases}$

- ***Đối với bộ lọc FIR loại 4:*** $\theta(k) = \begin{cases} \frac{\pi}{2} - \frac{N-1}{2}\left[\frac{2\pi}{N}k\right] & , \ k = 1, \dots, \frac{N-2}{2} \\ -\frac{\pi}{2} + \frac{N-1}{2}\left[\frac{2\pi}{N}(N-k)\right] & , \ k = \frac{N}{2}, \dots, N-1 \end{cases}$

Nếu coi hàm sai số xấp xỉ được tính bằng độ sai lệch giữa đáp ứng tần số của bộ lọc lý tưởng với đáp ứng tần số của bộ lọc thực tế , ta có các nhận xét sau:

- Hàm sai số xấp xỉ bằng không tại các tần số được lấy mẫu
- Hàm sai số xấp xỉ tại các tần số khác phụ thuộc vào mức độ dốc hay độ biến đổi đột ngột tại tần số đó. Tại tần số có đáp ứng càng dốc, ví dụ gần biên của dải thông và dải chắn, thì có hàm sai số xấp xỉ càng lớn.

Dãy đáp ứng xung của bộ lọc được suy ra từ biến đổi Fourier rời rạc ngược của dãy các mẫu X(k): $h(n) = IDFT[X(k)]$ và hàm tìm biến đổi Fourier rời rạc ngược bằng thuật toán biến đổi Fourier nhanh có thể được áp dụng trong trường hợp này.

4.1.2. Một số lệnh và hàm của MATLAB

Phần này đưa ra danh mục các lệnh các hàm của MATLAB có thể sử dụng trong phần thí nghiệm này. Để biết cụ thể hơn về chức năng của hàm và cú pháp của lệnh gọi hàm, gõ lệnh help kèm theo tên của hàm tại cửa số lệnh của MATLAB.

- **freqz:** trả về đáp ứng tần số của một hệ thống tại một số hữu hạn các điểm rời rạc trên vòng tròn đơn vị khi biết hàm truyền đạt của nó.
- **sin, cos, sinc:** trả về các hàm toán học thể hiện các công thức lượng giác.
- **boxcar, bartlett, hanning, hamming, blackman, kaiser:** trả về các hàm cửa sổ với chiều dài cho trước.
- **firpm:** thực hiện công việc tìm ra dãy đáp ứng xung của bộ lọc tối ưu theo nghĩa Chebyshev bằng thuật toán của **Parks** và **McClellan**.

4.1.3. Thực hiện các bài Lab.

Lab 1: Tạo các hàm thể hiện độ lớn của đáp ứng tần số các bộ lọc FIR loại 1 và FIR loại 2 từ dãy đáp ứng xung của chúng theo chương trình mẫu và ghi lại theo các tên tệp lần lượt là **Hr_Type1.m** và **Hr_Type2.m**:

```
function [Hr,w,a,L] = Hr_Type1(h)
% Tinh ham do lon cua dap ung tan so Hr(w)
% bo loc FIR loai 1
% [Hr,w,a,L] = Hr_Type1(h)
% Hr = Do lon ,  w = Vector tan so trong khoang [0 pi]
%  a = Cac he so cua bo loc FIR loai 1 ,  L = Bac cua bo loc
%  h = Dap ung xung cua bo loc FIR loai 1
M  = length(h);
L  = (M-1)/2;
a  = [h(L+1) 2*h(L:-1:1)];
n  = [0:1:L];
w  = [0:1:500]'*pi/500;
Hr = cos(w*n)*a';
```

Kết quả Lab 1:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Lab 2: Cho bộ lọc FIR với đáp ứng xung như sau:

```
function [Hr,w,b,L] = Hr_Type2(h)
% Tinh ham do lon cua dap ung tan so Hr(w)
% bo loc FIR loai 2
% [Hr,w,a,L] = Hr_Type2(h)
% Hr = Do lon ,  w = Vector tan so tron khoang [0 pi]
%  b = Cac he so cua bo loc FIR loai 2 ,  L = Bac cua bo loc
%  h = Dap ung xung cua bo loc FIR loai 2
M  = length(h);
L  = M/2;
b  = 2*h(L:-1:1);
n  = [1:1:L]; n = n-0.5;
w  = [0:1:500]'*pi/500;
Hr = cos(w*n)*b';
```

a. Xác định loại của bộ lọc.

Tính và biểu diễn trên đồ thị:

b. Dãy đáp ứng xung của bộ lọc.

c. Các hệ số của bộ lọc.

d. Hàm độ lớn của đáp ứng tần số.

e. Phân bố điểm cực và điểm không theo chương trình mẫu bằng cách gõ các dòng lệnh cho ở bảng dưới đây vào cửa số soạn thảo và ghi lại theo tên tệp là **Lab2_4_1.m**.

```
h = [-4,1,-1,-2,5,6,5,-2,-1,1,-4];
M = length(h); n =0:M-1;
[Hr,w,a,L] = Hr_Type1(h);
a, L
amax = max(a)+1; amin = min(a)-1;
subplot(2,2,1); stem(n,h);
axis([-1,2*L+1,amin,amax]);
title('Impulse Response');
xlabel('n'); ylabel('h(n)');
subplot(2,2,3); stem(0:L,a);
axis([-1,2*L+1,amin,amax]);
title('a(n) coefficients');
xlabel('n'); ylabel('a(n)');
subplot(2,2,2); plot(w/pi,Hr); grid;
title('Type-1 Amplitude Response');
xlabel('frequency in pi units'); ylabel('Hr');
subplot(2,2,4); zplane(h,1);
```

Kết quả Lab 2:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Lab 3: Cho bộ lọc FIR với đáp ứng xung như sau:

$$h(n) = \{-1, 2, 3, 1, 4, -3, 2, 4, -5, 5, 1\}$$
$$\uparrow$$

a. Xác định loại của bộ lọc.

Viết chương trình tính và biểu diễn trên đồ thị:

b. Dãy đáp ứng xung của bộ lọc.

c. Các hệ số của bộ lọc.

d. Hàm độ lớn của đáp ứng tần số.

e. Phân bố điểm cực và điểm không.

Kết quả Lab 3:

Lab 4: Tạo hàm thể hiện độ dãy đáp ứng xung của bộ lọc thông thấp lý tưởng từ các tham số đầu vào là tần số cắt ω_c và chiều dài đáp ứng xung **M** theo chương trình mẫu và ghi lại theo tên tệp là **ideal_lp.m**:

```
function hd = ideal_lp(wc,M)
% Ideal LowPass filter computation
% [hd] = ideal_lp(wc,M)
% hd = ideal impluse response between 0 to M-1
% wc = cutoff frequency in radians , M = length of the ideal filter
alpha = (M-1)/2;
n = [0:1:(M-1)];
m = n - alpha + eps;
hd = sin(wc*m) ./ (pi*m);
```

Lab 5: Thiết kế bộ lọc thông thấp theo phương pháp cửa số với các tham số đầu vào:

$$\omega_p = 0.2\pi, \quad R_p = 0.25dB \; ; \omega_s = 0.3\pi, \quad A_s = 50dB$$

Chọn cửa số Hamming. Tính và biểu diễn trên đồ thị:

a. Dãy đáp ứng xung của bộ lọc lý tưởng

b. Dãy hàm cửa sổ

c. Hàm độ lớn tuyệt đối của đáp ứng tần số

d. Hàm độ lớn tương đối tính theo dB của đáp ứng tần số theo chương trình mẫu và ghi lại theo tên tệp là **Lab5_4_1.m**

```
wp = 0.2*pi; ws =0.3*pi;
tr_width = ws - wp;
M = ceil(6.6*pi/tr_width) + 1
n = [0:1:M-1];
wc = (ws+wp)/2;
hd = ideal_lp(wc,M);
w_ham = (hamming(M))';
h = hd .* w_ham;
[db,mag,pha,grd,w] = freqz_m(h,[1]);
delta_w = 2*pi/1000;
Rp = -(min(db(1:1:wp/delta_w+1)))
As = -round(max(db(ws/delta_w+1:1:501)))
subplot(2,2,1); stem(n,hd);
axis([0,M-1,-0.1,0.3]);
title('Ideal Impulse Response');
xlabel('n'); ylabel('hd(n)');
subplot(2,2,2); stem(n,w_ham);
axis([0,M-1,0,1.1]);
```

```
title('Hamming Window');
xlabel('n'); ylabel('w(n)');
subplot(2,2,3); stem(n,h);
axis([0,M-1,-0.1,0.3]);
title('Actual Impulse Response');
xlabel('n'); ylabel('h(n)');
subplot(2,2,4); plot(w/pi,db); grid;
axis([0,1,-100,10]);
title('Magnitude Response in dB');
xlabel('frequency in pi units'); ylabel('Decibels');
```

Kết quả Lab 5:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Lab 6: Thiết kế bộ lọc thông dải theo phương pháp lấy mẫu tần số với tham số đầu vào:

$$\omega_{p1} = 0.35\pi, \quad R_p = 1dB \; ; \omega_{s1} = 0.2\pi, \quad A_s = 50dB$$

$$\omega_{p2} = 0.65\pi, \quad R_p = 1dB\ ; \omega_{s2} = 0.8\pi, \quad A_s = 60dB$$

Giả sử rằng ta chọn đáp ứng xung có chiều dài 40 tương đương với lấy 40 mẫu tần số trong khoảng [0,2π). Dải thông có độ rộng là 0,3π tương đương với 7 mẫu nhận giá trị 1. Giả sử tiếp rằng quá trình tối ưu hoá chỉ ra nên chọn trên cả 2 dải chuyển tiếp 2 mẫu nhận các giá trị T1 = 0,109021 và T2 = 0,59417456. Vậy dãy mẫu được cho như sau:

$$H(k) = \{\underbrace{0,\ldots,0}_{5\ mẫu\ 0}, T_1, T_2, \underbrace{1,\ldots,1}_{7\ mẫu\ 1}, T_2, T_1, \underbrace{0,\ldots,0}_{9\ mẫu\ 0}, T_1, T_2, \underbrace{1,\ldots,1}_{7\ mẫu\ 1}, T_2, T_1, \underbrace{0,\ldots,0}_{4\ mẫu\ 0}\}$$

Viết chương trình tính và biểu diễn trên đồ thị:

a. Dãy các mẫu tần số

b. Dãy đáp ứng xung của bộ lọc thực tế

c. Hàm độ lớn tuyệt đối của đáp ứng tần số

d. Hàm độ lớn tương đối tính theo dB của đáp ứng tần số

Kết quả Lab 6:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....