دانشگاه سیرجند
دانشکده مهندسی
برق و کامپیوتر

دستور کار پیشنهادی

# آزمایشگاه پردازش سیگنال‌های دیجیتال

Digital Signal Processing Laboratory

(DSP Lab)

۱۳۹۷-۹۸

# فهرست مطالب

# آزمایش ۱

## تحلیل سیگنال‌های زمان‌گسسته در حوزه‌ی زمان

### هدف آزمایش

در این آزمایش، طریقه‌ی پیاده سازی تعدادی از سیگنال‌های گسسته‌ی پایه را در نرم افزار Matlab خواهیم آموخت و سپس عملیات ابتدایی را بر روی این سیگنال‌ها اعمال خواهیم کرد. علاوه بر این، با کاربرد تعدادی از فرمان‌های پایه ای متلب و طرز اعمال آن‌ها بر روی مسائل ساده‌ی پردازش سیگنال دیجیتال آشنا خواهیم شد.

### فرمان‌های مورد نیاز در این آزمایش

در ادامه، با برخی از فرمان‌های متلب که در آزمایش ۱ مورد استفاده قرار خواهد گرفت، آشنا می‌شویم:

عملگرها و کاراکترهای خاص : Operators and special characters:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| % | ; | / | * | - | + | . | : |

ماتریس‌های اولیه و به کار بردن ماتریس: Elementary matrices and matrix manipulation:

| | | | | | |
|---|---|---|---|---|---|
| i | ones | Pi | rand | randn | zeros |

توابع اولیه: Elementary functions:

| | | | |
|---|---|---|---|
| cos | exp | imag | real |

تحلیل داده‌ها: Data analysis:

| |
|---|
| sum |

ابزار ترسیم دو بعدی: Two dimensional graphics:

| | | | | |
|---|---|---|---|---|
| axis | grid | legend | plot | stairs |
| stem | title | xlabel | ylabel | |

ابزار ترسیم عمومی: General purpose graphics functions:

| | |
|---|---|
| clf | subplot |

جعبه ابزار پردازش سیگنال: Signal processing toolbox:

| | |
|---|---|
| sawtooth | square |

## آزمایش ۱-۱ : تابع ضربه و تابع پله‌ی واحد

دو نمونه از مهم‌ترین سیگنال‌های پایه ای در پردازش سیگنال ، سیگنال ضربه و سیگنال پله‌ی واحد می‌باشند. سیگنال ضربه u[n] با طول N را در متلب می‌توانیم به صورت زیر ایجاد کنیم.

```
U=[1    zeros (1,N-1)]
```

همچنین، سیگنال پله‌ی واحد به صورت زیر ایجاد می‌شود:

```
S=[ ones(1,N) ]
```

در صورتی که بخواهیم در سیگنال با طولN ، تاخیر به اندازه‌ی M سپمل ایجاد کنیم (M<N)، میتوانیم به مقدار تاخیر، صفر به ابتدای سیگنال اضافه کنیم. به عنوان مثال سیگنال ضربه با تاخیر M:

```
Ud=[ zeros(1,M)    1    zeros(1,N-M-1) ]
```

در این آزمایش، به منظور ایجاد تابع ضربه و رسم آن، برنامه‌ی P1-1 را از فایل ضمیمه اجرا کرده و تمرین‌های زیر را انجام دهید.

تمرین ۱: هر کدام از دستورهای clf, axis, title, xlabel, ylabel چه کاری را انجام می‌دهند؟

تمرین ۲: برنامه‌ی آزمایش را به نحوی تغییر دهید تا تابع ضربه با تاخیر ۱۱ سمپل را نمایش دهد.

تمرین۳: برنامه‌ی آزمایش را به نحوی تغییر دهید تا تابع پله‌ی واحد را نمایش دهد.

تمرین۴: برنامه ای بنویسید که خروجی آن، تابع پله‌ی واحد با تاخیر ۷ سمپل باشد.

## آزمایش ۱-۲: سیگنال‌های نمایی

یک نمونه‌ی دیگر از سیگنال‌های پایه‌ای مهم، سیگنال‌های نمایی هستند که در متلب، میتوان با استفاده از دستور exp و یا با به توان رساندن از طریق عملگرهای ^. آن‌ها ر ایجاد کرد. به عنوان نمونه :

$e^a$= exp (a)

$5^a$= 5. ^a

برنامه‌ی P1-2 برای ایجاد رشته‌ی نمایی مختلط نوشته شده است. آن را اجرا کرده و تمرین‌های زیر را انجام دهید.

تمرین ۵: کدام پارامتر، نرخ رشد و یا نزول این رشته را کنترل می‌کند؟ با کدام پارامتر می‌توان دامنه‌ی این سیگنال را تغییر داد؟

تمرین۶: اگر پارامتر c را به مقدار (1/12)+(pi/6)*i تغییر دهیم، چه تغییری در نتیجه حاصل خواهد شد؟

تمرین۷: نقش عملگرهای real و imag و نقش فرمان subplot در برنامه چیست؟

برنامه‌ی P1-3 را به منظور ایجاد سیگنال نمایی حقیقی اجرا کنید.

تمرین۸: نرخ رشد و نزول و دامنه‌ی این سیگنال توسط چه متغیرهایی تغییر می‌کند؟

تمرین۹: تفاوت بین عملگر ^ با .^ در چیست؟

تمرین۱۰: پارامتر a را به عددی کمتر از یک تغییر دهید، چه تغییری در سیگنال حاصل می‌شود؟ برنامه را به ازای a=0.9 و k=20 اجرا کنید.

تمرین ۱۱: طول سیگنال چند است و چگونه میتوان آن را تغییر داد؟

تمرین ۱۲: انرژی سیگنال را یک بار در حالت اولیه‌ی تمرین P1-3 و دفعه‌ی دیگر بعد از اعمال تغییرات تمرین ۱۰ محاسبه کنید. چه تفاوتی در انرژی سیگنال ایجاد می‌شود؟ چرا ؟

نکته: برای محاسبه‌ی انرژی سیگنال حقیقی گسسته S در متلب می‌توان از عبارت sum(s.*s) کمک گرفت.

## آزمایش ۱-۳: سیگنال‌های سینوسی

یکی دیگر از سیگنال‌هایی که در مهندسی برق کاربرد فراوانی دارد، سیگنال سینوسی است. این نوع سیگنال می‌تواند در حالت کلی به صورت زیر نمایش داده شود:

$$x[n] = Acos(\omega_0 n + \varphi)$$

با استفاده از عملگرهای مثلثاتی cos و sin در متلب، می‌توان این توابع را ایجاد کرد.

برنامه‌ی P1-4 را اجرا کنید.

تمرین۱۲: فرکانس و دوره‌ی تناوب سیگنال را مشخص کنید. توسط چه پارامترهایی در برنامه می‌توان فرکانس، فاز و دامنه-ی سیگنال را تغییر داد؟

تمرین۱۳: طول سیگنال چند است و چگونه می‌توان آن را تغییر داد؟

تمرین۱۳: توان متوسط سیگنال را محاسبه کنید.

تمرین۱۴: نقش فرمان های grid و axis را بیان کنید.

تمرین ۱۵: با استفاده از این برنامه، سیگنال‌های سینوسی جدید یک بار با فرکانس ۰,۹ و سپس با فرکانس ۱,۱ ایجاد کرده و مشاهده کنید. چه تفاوتی در شکل آن‌ها ایجاد می‌شود؟

تمرین۱۶: یک برنامه بنویسید که سیگنال سینوسی با طول ۵۰، فرکانس ۰,۰۸ ، دامنه‌ی ۲,۵ و فاز ۹۰ درجه ایجاد کرده و آن را رسم کند. دوره‌ی تناوب این سیگنال چند است؟

تمرین ۱۷: فرمان stem را یک دفعه با فرمان plot و دفعه‌ی دیگر با فرمان stairs جایگزین کنید. شکل نمایش داده شده را در هر یک از این حالت ها با هم مقایسه کنید.

## آزمایش ۱-۴: سیگنال‌های تصادفی

به منظور ایجاد سیگنال تصادفی با طول N و توزیع یکنواخت در بازه‌ی (0,1) ، می‌توانیم از فرمان زیر در متاب استفاده کنیم:

```
x=rand(1,N);
```

همچنین، دنباله‌ی تصادفی با طول N و با توزیع نرمال ( دارای میانگین صفر و واریانس یک) با استفاده از فرمان زیر، قابل ایجاد است:

```
x=randn(1,N);
```

تمرین۱۸: برنامه‌ای بنویسید که سیگنال تصادفی با طول ۱۰۰ با توزیع یکنواخت در بازه‌ی [-2,2] ایجاد کرده و آن را نمایش دهد.

تمرین۱۹: برنامه‌ای بنویسید که سیگنال تصادفی گوسی با طول ۷۵ با توزیع نرمال و میانگین صفر و واریانس ۳ ایجاد کند و آن را نمایش دهد.

تمرین ۲۰: برنامه ای بنویسید که جمله‌هایی پنج سمپلی از سیگنال تصادفی سینوسی زیر با طول ۳۱ را ایجاد کرده و نشان دهد.

$$\{X[n]\} = \{Acos(\omega_0 n + \varphi)\}$$

در حالی که $A$ و $\varphi$ به صورت آماری مستقل بوده و دارای توزیع یکنواخت در بازه‌ی $0 \leq A \leq 4$ و $0 \leq \varphi \leq 2\pi$ می‌باشند.

## آزمایش ۱-۵: عملگرهای ساده روی سیگنال‌ها، هموار سازی سیگنال

تا به اینجا با نحوه‌ی ایجاد انواع سیگنال‌های پایه‌ای آشنا شده‌اید. در این بخش، با اجرای عملگرهای پایه‌ای از قبیل جمع، ضرب اسکالر، معکوس زمانی، تاخیر و ضرب نقطه‌ای بر روی سیگنال‌های گسسته آشنا خواهید شد. به عنوان یک مثال معروف از کاربرد پردازش سیگنال دیجیتال، هموار سازی سیگنال را مورد بررسی قرار خواهیم داد.

فرض کنید s[n] یک سیگنال گسسته و d[n] نویز تصادفی باشد و سیگنال x[n] از جمع شدن نویز با سیگنال اصلی ایجاد شود. به منظور هموار کردن سیگنال و تخمین قابل قبول از سیگنال s[n]، از رابطه‌ی زیر استفاده می‌شود:

$$y[n] = \frac{1}{3}(x[n-1] + x[n] + x[n+1])$$

تمرین ۲۱: برنامه‌ی P1-5 را اجرا کنید. فرم سیگنال s[n] و d[n] در این برنامه چگونه است؟

تمرین ۲۱: چرا در برنامه برای جمع کردن سیگنال با نویز، از عبارت x=s+d استفاده نشده است؟

تمرین۲۲: بین سیگنال‌های x، x1، x2 و x3 چه رابطه‌ای بر قرار است؟

تمرین ۲۳: نقش فرمان legend در برنامه چیست؟

**آزمایش ۱-۶: ترکیب سیگنال‌ها، مدولاسیون دامنه**

با ترکیب سیگنال‌های پایه و استفاده از عملگرهای ساده، می‌توان سیگنال‌های پیچیده‌تری به وجود آورد. به عنوان مثال، در این آزمایش، می‌خواهیم مدولاسیون دامنه (AM) را توسط سیگنال سینوسی با فرکانس بالا $x_H[n]=\cos(w_H n)$ برای سیگنال فرکانس پایین $x_L[n]=\cos(w_L n)$ انجام دهیم. سیگنال مدوله شده‌ی نهایی به صورت زیر است:

$$y[n] = A(1 + m.x_L[n])x_L[n] = A\big(1 + m.cos(w_L n)\big)cos(w_H n)$$

که در آن، m اندیس مدولاسیون است و باید به گونه‌ای باشد که عبارت $(1 + m.x_L[n])$ به ازای همه‌ی n ها، مثبت باشد. برای ایجاد مدولاسیون دامنه، برنامه‌ی P1-6 را اجرا کنید.

تمرین۲۴: این برنامه را به ازای مقادیر مختلف اندیس مدولاسیون، فرکانس حامل و فرکانس سیگنال اصلی اجرا و نتیجه را تحلیل کنید.

تمرین ۲۵: تفاوت عملگرهای ریاضی * و *. را بیان کنید.

از آنجایی که فرکانس لحظه‌ای در هر سیگنال سینوسی برابر است با مشتق فاز آن نسبت به زمان، برای ایجاد سیگنال‌هایی که فرکانس آن‌ها با زمان به صورت خطی تغییر می‌کند، نیاز است تا آرگومان آن‌ها تابعی از درجه‌ی دو نسبت به زمان باشد.

تمرین ۲۶: برنامه‌ی P1-7 را اجرا کنید. کمترین و بیشترین فرکانس این سیگنال چند است؟

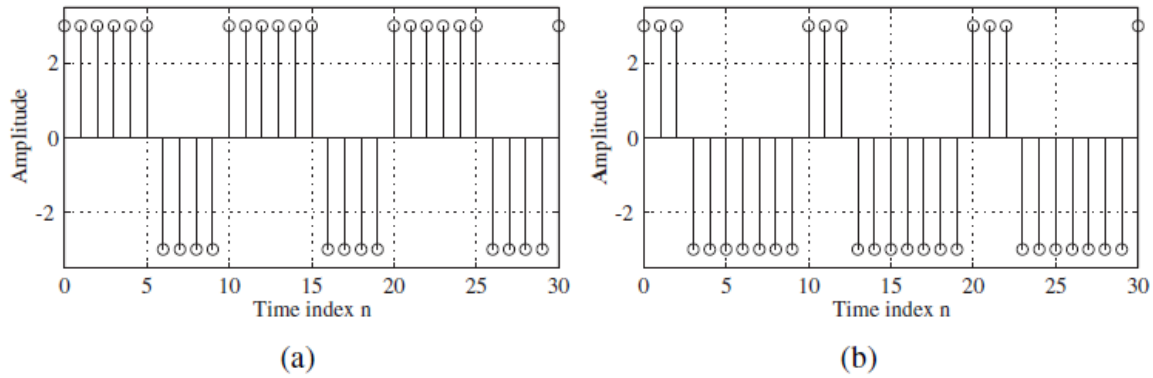
تمرین ۲۷: برنامه را طوری تغییر دهید تا سیگنال سینوسی با فرکانس متغیر با مینیمم ۰,۱ و ماکزیمم ۰,۳ ایجاد کند.

**آزمایش ۱-۶: اطلاعات مربوط به workspace**

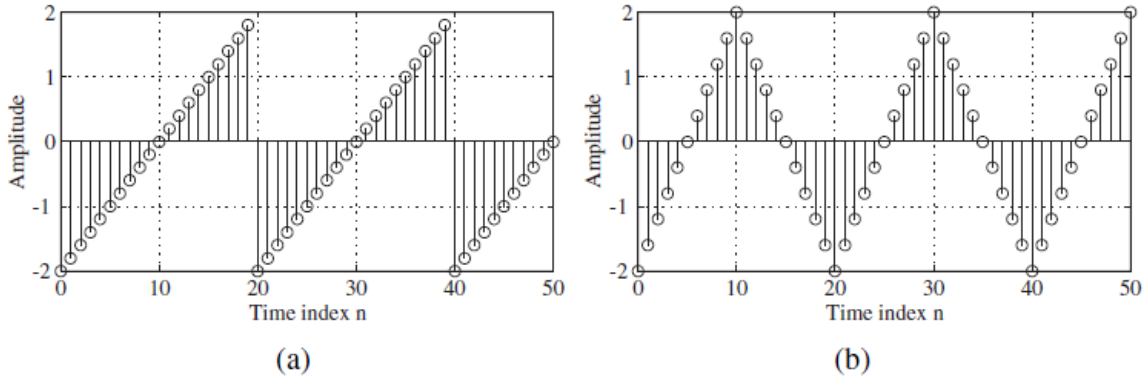
تمرین ۲۸: فرمان‌های who و whos برای نمایش اطلاعات مربوط به متغیرهای ذخیره شده در برنامه استفاده می‌شود. این دو فرمان را جداگانه اجرا کنید. چه تفاوتی با هم دارند؟

## آزمایش ۱-۷: شکل موج های دیگر (شکل موج مربعی و دندانه اره‌ای)

تمرین ۲۹: با استفاده از توابع square و sawtooth ، برنامه ای بنوسید که خروجی آن، نمودارهای زیر باشد.




شکل ۱. شکل موج مربعی




شکل ۲. شکل موج دندانه اره‌ای

# آزمایش ۲

## تحلیل سیستم‌های زمان‌گسسته در حوزه‌ی زمان

### هدف آزمایش

در مبحث تجزیه و تحلیل، بررسی خواص سیستم‌ها از لحاظ خطی بودن، پایداری، تغییر ناپذیزی با زمان و ... امری پرکاربرد است. در این آزمایش، برخی از خواص سیستم‌های گسسته را از طریق نرم افزار متلب، بررسی خواهیم کرد.

### فرمان‌های مورد نیاز در این آزمایش

در ادامه، با برخی از فرمان‌های متلب که در آزمایش ۲ مورد استفاده قرار خواهد گرفت، آشنا می‌شویم:

فرمان‌های عمومی: General purpose commands

| |
|---|
| disp |

عملگرها و کاراکترهای خاص : Operators and special characters:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| : | . | + | - | * | / | ; | % |
| < | | | | | | | |

ساختارهای زبانی و اشکال زدایی : Language constructs and debugging:

| | | | | |
|---|---|---|---|---|
| break | end | for | If | input |

ماتریس‌های اولیه و به کار بردن ماتریس: Elementary matrices and matrix manipulation:

| | | |
|---|---|---|
| ones | pi | zeros |

توابع اولیه: Elementary functions:

| | |
|---|---|
| abs | cos |

توابع چندجمله‌ای و تعاملی : Polynomial and interpolation functions:

| |
|---|
| conv |

ابزار ترسیم دو بعدی: Two dimensional graphics:

| | | | | |
|---|---|---|---|---|
| axis | plot | stem | title | xlabel |
| ylabel | | | | |

ابزار ترسیم عمومی: General purpose graphics functions:

| | |
|---|---|
| clf | subplot |

توابع مربوط به کاراکترهای رشته‌ای: Character string functions:

| |
|---|
| Num2str |

جعبه ابزار پردازش سیگنال: Signal processing toolbox:

| | |
|---|---|
| filter | impz |

**آزمایش ۲-۱: بررسی سیستم تغییر میانگین**

سیستم زیر را که در آزمایش قبل برای هموار سازی استفاده شد، در نظر بگیرید.

$$y[n] = \frac{1}{3}(x[n-1] + x[n] + x[n+1]) \quad (۱-۲)$$

آیا این سیستم LTI است ؟ آیا این سیستم علی است ؟

تعمیم این رابطه به صورت فیلتر FIR نوشته می‌شود که هموار ساز M نقطه ای است و فیلتر تغییر میانگین نامیده می‌شود:

$$y[n] = \frac{1}{M}\sum_{k=0}^{M-1} x[n-k] \quad (۲-۲)$$

در این آزمایش، از این فیلتر برای حذف مولفه‌های فرکانس بالای سیگنال ورودی استفاده می‌کنیم.

تمرین۱: برنامه‌ی P2-1 را به ازای M=2 اجرا کنید. و تصویر سیگنال ورودی به فیلتر و خروجی از آن را مشاهده کنید. کدام مولفه از سیگنال ورودی توسط فیلتر حذف شده‌اند ؟

تمرین۲: اگر در تمرین قبل، فیلتر را به حالت $0.5[x[n] - x[n-1]]$ تغییر دهیم، تاثیرش در خروجی چه خواهد بود؟

تمرین ۳: به ازای مقادیر مختلفی از M و فرکانس سیگنال‌های ورودی، برنامه را اجرا کرده و نتایج را تحلیل کنید.

تمرین ۴: حال سیگنال ورودی را با یک سیگنال سینوسی با فرکانس افزایشی دارای طول ۱۰۱ و فرکانس ۰ تا ۰٫۵ جایگزین کنید سپس تمرین ۱ و تمرین ۲ را با این ورودی جدید پاسخ دهید.

**آزمایش ۲-۲: بررسی یک سیستم غیرخطی ساده**

سیگنال غیر خطی زیر را در نظر بگیرید:

$$y[n] = x[n]^2 - x[n-1]x[n+1] \quad (۳-۲)$$

در این آزمایش، میخواهیم با استفاده از ورودی‌های متنوع، خروجی سیستم غیر خطی بالا را بررسی کنیم.

تمرین۵: برنامه‌ی P2-2 را به ازای ورودی‌های سینوسی با فرکانس‌های مختلف اجرا کرده و خروجی را مشاهده نمایید. سیگنال خروجی چگونه با تغییر فرکانس تغییر می‌کند؟ از لحاظ تئوری نیز بررسی کنید.

تمرین ۶: خروجی این برنامه را به ازای ورودی سینوسی دارای مقدار DC به فرم $x[n] = \sin(w_0 n + k)$ بررسی کنید. سیگنال خروجی چه رابطه ای با مقدار DC سیگنال ورودی دارد ؟

## آزمایش ۲-۳: سیستم‌های خطی و غیرخطی

در این آزمایش، میخواهیم خاصیت خطی بودن را برای سیستم علی زیر بررسی کنیم:

$$y[n] - 0.4y[n-1] + 0.75y[n-2] = 2.2403x[n] + 2.4908x[n-1] + 2.2403x[n-2] \quad (۴-۲)$$

تمرین ۷: برنامه‌ی P2-3 را اجرا کنید. در این برنامه یک بار ترکیب خطی از دو ورودی به سیستم داده می‌شود و بار دیگر، خروجی حاصل از هر کدام از ورودی ها به صورا جداگانه با هم ترکیب خطی می‌شوند. این دو خروجی را با هم مقایسه کنید. آیا سیستم خطی است؟

تمرین ۸: برنامه‌ی قبل را برای سه مقدار مختلف از ثابت‌های a و b و همچنین سه مقدار وختلف از فرکانس سیگنال‌های ورودی اجرا کرده و خطی بودن سیستم را بررسی کنید. چه نتیجه‌ای میگیرید؟

تمرین۹: تا به اینجا شرایط اولیه‌ی فیلتر صفر در نظر گرفته شده بودند. حال برنامه را با شرایط اولیه‌ی غیر صفر (دلخواه) اجرا کنید. چه نتیجه ای میگیرید؟

تمرین۱۰: مشابه تمرین ۸ ، مقادیر مختلف از پارامترهای ثابت و فرکانس‌های ورودی را این بار با شرایط اولیه‌ی غیر صفر بررسی کنید.

تمرین ۱۱: سیستم زیر را جایگزین سیستم برنامه‌ی P2-3 کنید و تمرین ۷ را تکرار کنید.

$$y[n] = x[n].x[n-1] \quad (۵-۲)$$

آیا این سیستم خطی است؟

## آزمایش ۲-۴: سیستم‌های تغییر پذیر و تغیر ناپذیر با زمان

در این آزمایش، میخواهیم خاصیت تغییر پذیری و یا تغییر ناپذیری با زمان را برای سیستم بررسی کنیم. سیستم آزمایش قبلی با رابطه‌ی (۲-۴) را دوباره درنظر بگیرید. در این آزمایش، ورودی های $x[n]$ و $x[n-D]$ را به سیستم اعمال کرده و

خروجی‌های متناظر $y_1[n]$ و $y_2[n]$ را به دست می‌آوریم. سپس بررسی می‌کنیم که آیا $y_2[n+D]$ با $y_1[n]$ برابر است یا نه.

تمرین۱۲: برنامه‌ی P2-4 را اجرا کنید و خروجی های $y[n]$ و $y_a[n-10]$ را مشاهده و با هم مقایسه کنید. این دو خروجی چه رابطه‌ای با هم دارند؟ آیا این سیستم تغییر ناپذیر با زمان است ؟

تمرین۱۳: تمرین قبل را با مقادیر مختلفی از تغییر D بررسی کنید.

تمرین۱۴: تمرین ۱۲ را به ازای سه مقدار مختلف از فرکانس ورودی بررسی کنید.

تمرین۱۵: حال تمرین ۱۲ را با شرایط اولیه‌ی غیر صفر بررسی کنید. آیا در اینصورت سیستم تغییر ناپذیر با زمان است؟

تمرین ۱۶: سیستم دارای شرایط اولیه را با سه مقدار مختلف از فرکانس ورودی بررسی کنید. آیا این سیستم تغییر ناپذیر با زمان است؟

تمرین ۱۷: سیستم این برنامه را با سیستم زیر جایگزین کرده و خاصیت تغییر پذیری با زمان را بررسی کنید.

$$y[n] = nx[n] + x[n-1] \qquad (۶-۲)$$

## آزمایش ۲-۵: سیستم‌های خطی و تغییر ناپذیر با زمان (LTI)، محاسبه‌ی پاسخ فرکانسی

با استفاده از فرمان $y = impz(num, den, N)$ در متلب، میتوان N سمپل اولیه‌ی پاسخ ضربه‌ی سیستم علی و LTI را مشاهده کرد. $num$ نمایانگر بردار ضرایب صورت و $den$ ضرایب مخرج سیستم است.

تمرین۱۸: برنامه‌ی P2-5 را اجرا و پاسخ ضربه‌ی سیستم را مشاهده کنید.

تمرین ۱۹: برنامه را طوری تغییر دهید که ۴۵ سمپل اولیه از سیستم علی و LTI دارای رابطه‌ی زیر را نشان دهد.

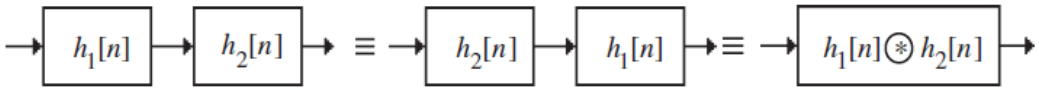
$$\begin{aligned} y[n] + 0.71y[n-1] - 0.46y[n-2] - 062y[n-3] \\ = 0.9x[n] - 0.45x[n-1] + 0.35x[n-2] + 0.002x[n-3] \end{aligned} \qquad (۷-۲)$$

تمرین ۲۰: برنامه‌ای بنویسید که پاسخ ضربه‌ی سیستم نشان داده شده در رابطه‌ی (۲-۷) را این‌بار با فرمان filter محاسبه کند و تعداد ۴۰ سمپل اولیه‌ی آن را نمایش دهد. نتایج خود را با پاسخ ضربه‌ی محاسبه شده در تمرین ۱۹ مقایسه کنید.

تمرین ۲۱: برنامه‌ای بنویسید که پاسخ پله‌ی سیستم نشان داده شده در رابطه‌ی (۲-۷) را محاسبه و ۴۰ سمپل از آن را نمایش دهد.

## آزمایش ۲-۶: سری کردن سیستم‌ها

در عمل، سیستم‌های علی و LTI مرتبه بالا را با استفاده از سری کردن سیستم‌های مرتبه پایین‌تر پیاده سازی می‌کنند. در شکل ۲-۱ اتصال سری را مشاهده می‌کنید.




شکل ۲-۱ اتصال سری سیستم‌ها

به عنوان مثال، سیستم مرتبه چهار زیر را در نظر بگیرید:

$$y[n] + 1.6y[n-1] + 2.28y[n-2] - 062y[n-3] = 0.9x[n] - 0.45x[n-1] + 0.35x[n-2] + 0.002x[n-3] \quad (۸-۲)$$

در این آزمایش، می‌خواهیم نشان دهیم که سیستم فوق را می‌توان با سری کردن دو سیستم زیر به دست آورد:

$$y1[n] + 0.9y1[n-1] + 0.8y1[n-2] = 0.3x[n] - 0.2x[n-1] + 0.4x[n-2] \quad (۹-۲)$$

$$y2[n] + 0.7y2[n-1] + 0.85y2[n-2] = 0.2y1[n] - 0.5y1[n-1] + 0.3y1[n-2] \quad (۱۰-۲)$$

تمرین ۲۲: برنامه‌ی P2-6 را اجرا کنید و بررسی کنید که آیا $y[n]$ با $y2[n]$ برابر است ؟

تمرین ۲۳: این برنامه را یک مرتبه برای ورودی سینوسی و یک مرتبه‌ی دیگر با شرایط اولیه‌ی غیر صفر اجرا کنید . چه نتیجه ای می‌گیرید؟

## آزمایش ۲-۷: کانولوشن

میدانیم که یکی از خواص سیستم‌های LTI، این است که پاسخ سیستم را میتوان با استفاده از کانوالو ورودی محدود با پاسخ ضربه‌ی سیستم به دست آورد . در این آزمایش این خاصیت را بررسی خواهیم کرد.

تمرین۲۴: برنامه‌ی P2-7 را اجرا کنید و خروجی حاصل از کانولوشن را با خروجی حاصل از فیلتر مقایسه کنید. آیا این دو سیگنال برابرند؟ دلیل اضافه کردن صفر به ورودی قبل از فیلتر کردن چیست؟

تمرین ۲۵: برنامه ای بنویسید که بردار ورودی $x[n]$ با ۱۵ عضو را با پاسخ ضربه‌ی $h[n]$ کانوالو کند. همچنین این خروجی را از طریق فیلتر کردن نیز بیابد.

## آزمایش ۲-۸: پایداری سیستم LTI:

می‌دانیم که یک سیستم LTI زمانی نسبت به ورودی محدود پایدار است که پاسخ ضربه‌ی آن محدود باشد. بنابراین شرط لازم برای پایداری سیستم LTI و IIR این است که با افزایش اندیس زمان، پاسخ ضربه به سمت صفر همگرا شود. در این آزمایش، پایداری سیستم IIR را بررسی خواهیم کرد.

تمرین ۲۶: برنامه‌ی P2-8 را اجرا کنید. این برنامه چگونه \ایداری را بررسی می‌کند؟ دلیل استفاده از دستورهای for, end و break چیست ؟

تمرین ۲۷: سیستم گسسته ای که پاسخ ضربه‌ی آن در این سیستم بررسی می‌شود چیست؟ مشابه همین برنامه، برنامه‌ای بنویسید و با آن، پایداری سیستم زیر را بررسی کنید.

$$y[n] = x[n] - 4x[n-1] + 3x[n-2] + 1.7y[n-1] - y[n-2] \quad (۹-۲)$$

# آزمایش ۳

## تحلیل سیگنال‌های زمان‌گسسته در حوزه‌ی فرکانس

### هدف آزمایش

در دو آزمایش قبل، با تعدادی از ویژگی‌های سیگنال‌ها و سیستم‌های گسسته در حوزه‌ی زمان آشنا شدیم و آن‌ها را بررسی کردیم. با تحلیل در حوزه‌ی فرکانس، ویژگی‌های مفید دیگری به دست خواهد آمد. در این آزمایش، با ورود به حوزه‌ی فرکانس، ویژگی‌های سیگنال‌های گسسته مورد بررسی قرار می‌گیرد. بدین منظور، سه نوع نمایش معروف و پرکاربرد تبدیل فوریه گسسته (DTFT) ، تبدیل فوریه‌ی سریع (FFT) و تبدیل Z (در آزمایش بعدی) به کار گرفته می‌شود و ویژگی‌های آن-ها بررسی خواهند شد.

### فرمان‌های مورد نیاز در این آزمایش

در ادامه، با برخی از فرمان‌های متلب که در آزمایش ۳ مورد استفاده قرار خواهد گرفت، آشنا می‌شویم:

فرمان‌های عمومی : General purpose commands:

| | |
|---|---|
| disp | |

عملگرها و کاراکترهای خاص : Operators and special characters:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| % | ; | / | * | - | + | . | : |
| | | ~= | .^ | ^ | .* | > | < |

ساختارهای زبانی و اشکال زدایی : Language constructs and debugging:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| % | ; | / | * | - | + | . | : |
| | | ~= | .^ | ^ | .* | > | < |

ماتریس‌های اولیه و به کار بردن ماتریس: Elementary matrices and matrix manipulation:

| | | | | |
|---|---|---|---|---|
| fliplr | i | pi | zeros | : |

توابع اولیه: Elementary functions:

| | | | | | |
|---|---|---|---|---|---|
| abs | angle | conj | exp | imag | real |
| rem | | | | | |

توابع چندجمله‌ای و تعاملی : Polynomial and interpolation functions:

| |
|---|
| conv |

ابزار ترسیم دو بعدی: Two dimensional graphics:

| | | | | |
|---|---|---|---|---|
| axis<br>xlabel | grid<br>ylabel | plot | stem | title |

ابزار ترسیم عمومی: General purpose graphics functions:

| | |
|---|---|
| clf | subplot |

توابع مربوط به کاراکترهای رشته‌ای: Character string functions:

| |
|---|
| Num2str |

آنالیز داده و توابع تبدیل فوریه: Data analysis and fourier transform functions:

| | | | |
|---|---|---|---|
| fft | ifft | max | min |

جعبه ابزار پردازش سیگنال: Signal processing toolbox:

| | | | | | |
|---|---|---|---|---|---|
| freqz | impz | residuez | Tf2zp | Zp2sos | Zp2tf |
| zplane | | | | | |

## آزمایش ۳-۱: محاسبه‌ی تبدیل فوریه گسسته DTFT

می‌دانیم که تبدیل فوریه‌ی سیگنال گسسته، تابعی پیوسته از فرکانس می‌باشد. روابط مربوط به تبدیل فوریه گسسته و معکوس آن را در ادامه مشاهده می‌کنید.

$$X(\omega) = \sum_{n=-\infty}^{n=\infty} x[n]e^{-j\omega n} \quad (۱-۳)$$

$$x[n] = \frac{1}{2\pi}\int_{\omega=-\pi}^{\omega=\pi} X(\omega)e^{j\omega n} \quad (۲-۳)$$

با توجه به روابط فوق، به دو دلیل نمی‌توانیم تبدیل فوریه گسسته را در متلب پیاده سازی کنیم. اولا در رابطه‌ی تبدیل فوریه، طول سیگنال نامحدود است اما در کامپیوتر قابل نمایش و محاسبه نیست؛ دوما تبدیل فوریه‌ی گسسته تابعی پیوسته از $\omega$ می‌باشد، اما در متلب مقادیر به صورت گسسته ذخیره و نمایش داده خواهند شد. بنابراین، به منظور محاسبه تبدیل فوریه‌ی گسسته، میتوانیم از دستور freqz استفاده کرده و در ضرایب فرکانسی گسسته، مقادیر تبدیل فوریه را مشاهده کنیم.

تمرین۱: برنامه‌ی P3-1 را اجرا کنید و بخش حقیقی، موهومی، دامنه و فاز تبدیل فوریه را جداگانه محاسبه کنید. آیا تبدیل فوریه متناوب است ؟ اگر بله دوره‌ی تناوب آن چیست؟ قسمت‌های نمایش داده شده از لحاظ تقارن چگونه‌اند؟

تمرین۲: برنامه را طوری تغییر بدهید که تبدیل فوریه‌ی سیگنال محدود زیر را نمایش دهد.

$$g[n] = [1\ \ 3\ \ 5\ \ 7\ \ 9\ \ 11\ \ 13\ \ 15\ \ 17] \qquad (۳-۳)$$

تمرین۳: چرا در این برنامه از دستور pause استفاده شده است؟ آیا می‌توانید در برنامه فاز را بر حسب درجه نشان دهید؟

## آزمایش ۳-۲: ویژگی شیف زمانی تبدیل فوریه گسسته

با ویژگی‌های تبدیل فوریه‌ی گسسته و اثبات آن‌ها در درس تجزیه و تحلیل سیستم‌ها آشنا هستیم. در اینجا تعدادی از مهم-ترین ویژگی‌های پرکاربرد تبدیل فوریه گسسته را در غالب چندین آزمایش بررسی و مشاهده خواهیم کرد. ابتدا خاصیت شیفت زمانی را بررسی خواهیم کرد.

تمرین۴: برنامه‌ی P3-2 را اجرا کنید. شیفت زمانی چه تاثیری در تبدیل فوریه سیگنال داشته است؟برنامه را بگونه ای تغییر دهید تا برای هر بعد از نمودارها، برچسب مناسب نمایش دهد.

تمرین۵: به ازای دو رشته‌ی متفاوت و دو مقدار متفاوت شیفت زمانی، خروجی ها را بررسی کنید و نتیجه را بیان کنید.

## آزمایش ۳-۳: ویژگی شیف فرکانسی تبدیل فوریه گسسته

برنامه‌ی P3-3 به منظور بررسی خاصیت شیفت فرکانسی تبدیل فوریه نوشته شده است. با توجه به آن، تمرینات زیر را انجام دهید.

تمرین۶: تغییراتی در برنامه ایجاد کنید تا برای هر بعد از نمودارها، برچسب مناسبی نشان دهد. تاثیر شیفت فرکانسی در تبدیل فوریه چیست؟ چه پارامتری در این برنامه عامل تغییر شیفت فرکانسی است؟

تمرین۷: به ازای مقادیر مختلف شبفت فرکانسی، این ویژگی را بررسی کنید.

## آزمایش ۳-۴: ویژگی کانولوشن تبدیل فوریه گسسته

عمل کانولوشن یکی از کاربردی ترین ابزارها در تحلیل سیستم های LTI است. بنابراین، دانستن رفتار حوزه‌ی فرکانس دو سیگنالی که با هم کانوالو می‌شوند، مفید خواهد بود. برنامه‌ی P3-4 به منظور بررسی این ویژگی نوشته شده است.

تمرین۸: برنامه را به گونه ای تغییر دهید تا برای هر بعد از نمودارها، برچسب مناسبی نشان دهد. با توجه به این برنامه، ویژگی کانولوشن را چگونه تحلیل می‌کنید؟

تمرین۹: این خاصیت را برای دو سیگنال دلخواه دیگر بررسی کنید.

### آزمایش ۳-۵: ویژگی ضرب تبدیل فوریه گسسته

در بسیاری از کاربردها، نیاز است تا سیگنال‌ها را در حوزه‌ی زمان در هم ضرب کنیم. تغییرات تبدیل فوریه‌ی این سیگنال‌ها، به عنوان ویژگی مدولاسیون شناخته می‌شود. برنامه‌ی P3-5 به منظور بررسی این ویژگی نوشته شده است.

تمرین۹: برنامه را به گونه ای تغییر دهید که ابعد نمودارها، برچسب مناسب داشته باشند. ویژگی مدولاسیون را چگونه ارزیابی می‌کنید؟

تمرین۱۰: ویژگی مدولاسیون را برای دو سیگنال دلخواه دیگر بررسی کنید.

### آزمایش ۳-۶: ویژگی معکوس کردن زمان تبدیل فوریه گسسته

در این آزمایش، می‌خواهیم ببینیم اگر متغیر زمان در سیگنالی معکوس شود، تبدیل فوریه‌ی آن چه تغییری خواهد کرد. برنامه‌ی P3-6 بدین منظور نوشته شده است.

تمرین۱۱: مشابه تمرین‌های اخیر، برچسب‌های مناسب را برای ابعاد ایجاد کنید. از اجرای این برنامه چه نتیجه ای می‌گیرید؟

تمرین۱۲: این ویژگی را برای دو سیگنال دلخواه با طول متفاوت بررسی کنید.

### آزمایش ۳-۷: تبدیل فوریه سریع fft

در آزمایش اول دیدیم که تبدیل فوریه‌ی گسسته، با روابط (۳-۱) و (۳-۲) در متلب قابل محاسبه نیست و باید در مقادیر گسسته ای از فرکانس، تبدیل فوریه را بیابیم. بدین منظور می‌توانیم تبدیل فوریه‌ی گسسته را در مقادیر گسسته‌ی فرکانس با فاصله‌ی مساوی در بازه‌ی $0 \le \omega \ll 2\pi$ به ازای $\omega_k = \frac{2\pi k}{N}\ ; k = 0,1,\dots,N-1$ محاسبه کنیم. روابط بدین صورت خواهد بود:

$$X[k] = \sum_{n=0}^{M-1} x[n] e^{-j2\pi kn/N}\ ;\ k = 0,1,\dots,N-1 \qquad (۳-۴)$$

$$x[n] = \sum_{k=0}^{N-1} x[n]e^{j2\pi kn/N} \; ; \; n = 0,1, \dots, M-1 \qquad (۵-۳)$$

برای رشته‌ی محدود $x[n]$، تبدیل فوریه‌ی $X[k]$ توسط دستور fft به راحتی قابل محاسبه است. در صورتی که از دستور fft(x) استفاده کنیم، تبدیل فوریه‌ی محاسبه شده $X[k]$ طولی برابر با طول $x[n]$ خواهد داشت. برای محاسبه‌ی تبدیل فوریه‌ی L نقطه ای از سیگنال ورودی با N نقطه، از دستور fft(x,L) استفاده می‌شود. در اینصورت، خروجی سیگنالی L نقطه‌ای خواهد بود طوری که $L \gg N$ است.

تبدیل فوریه‌ی معکوس $X[k]$ نیز از رابطه‌ی ifft قابل محاسبه است.

تمرین۱۳: برنامه‌ای بنویسید که تبدیل فوریه‌ی L نقطه‌ای سیگنال ورودی دلخواه $x[n]$ دارای N نقطه را $(L \gg N)$ محاسبه و رسم کند، سپس تبدیل فوریه‌ی معکوس L نقطه‌ای را نیز رسم کند. به ازای چند ورودی دلخواه و مقادیر مختلفی از L و N خروجی ها را مشاهده کنید. چه نتیجه‌ای می‌گیرید؟

## آزمایش ۳-۸: ویژگی های شیفت و کانولوشن چرخشی fft

قبل از بررسی این ویژگی‌ها، ابتدا با طریقه‌ی ایجاد توابع (functions) مختلف در متلب آشنا می‌شویم. هر تابع در متلب به صورت فایل جداگانه نوشته شده و به همان نام، ذخیره می‌شود. شمای کلی هر تابع به صورت نمایش داده شده در شکل ۳-۱ می‌باشد. همان‌گونه که در شکل مشاهده می‌کنید، خروجی‌ها، ورودی‌ها و نام تابع در سطر اول مشخص می‌شوند و عملیاتی که قرار است بر روی این ورودی‌ها انجام شود برای محاسبه‌ی خروجی، در ادامه نوشته شده و در انتها، دستور end قرار می‌گیرد.

```
function [ output_args ] = Untitled( input_args )
%UNTITLED Summary of this function goes here
%   Detailed explanation goes here


end
```

شکل ۳-۱: نمای کلی یک تابع در متلب

برای استفاده از توابع، توجه به دو نکته حائز اهمیت است:

۱- نام فایل ذخیره شده‌ی تابع باید با نام خود تابع برابر باشد. برای فراخوانی تابع در برنامه نیز، از همان نام استفاده خواهد شد.

۲- فایل ذخیره شده‌ی تابع باید در path root متلب قرار داده شود. به عنوان مثال اگر برنامه‌ی متلبی را از درون فولدر خاصی اجرا می‌کنید، فایل تابع باید داخل همان پوشه قرار داده شده باشد.

تمرین ۱۴: تابع circshift1 را مشاهده کنید. توضیح دهید که این تابع چگونه شیفت چرخشی را انجام می‌دهد؟

تمرین ۱۵: تابع circonv را مشاهده و نحوه‌ی کار آن را توضیح دهید.

تمرین ۱۶: برنامه‌ی P3-7 را اجرا کنید. به ازای مقادیر مختلفی از مقدار شیفت، خروجی را بررسی کنید. اگر مقدار شیفت از طول سیگنال بزرگتر باشد چه اتفاقی خواهد افتاد؟

تمرین ۱۷: برنامه‌ی P3-8 را اجرا کنید. با توجه به این تمرین، خاصیت شیفت چرخشی تبدیل فوریه گسسته چیست؟

تمرین۱۸: برنامه‌ی P3-9 را اجرا کرده و خاصیت کانولوشن چرخشی تبدیل فوریه گسسته را توضیح دهید.

### آزمایش ۳-۹: پیاده سازی کانولوشن خطی با استفاده از کانولوشن چرخشی

در این بخش، می‌خواهیم با استفاده از کانولوشن چرخشی. کانولوشن خطی را پیاده سازی کنیم. برنامه‌ی P3-10 بدین منظور نوشته شده است.

تمرین۱۹: برنامه P3-10 را اجرا کنید. طبق این تمرین، چه خاصیتی بین کانولوشن چرخشی و کانولوشن خطی برقرار است؟

تمرین ۲۰: به ازای دو سیگنال دلخواه با طول متفاوت برنامه را اجرا کنید. چه نتیجه ای می‌گیرید؟

تمرین۲۱: برنامه‌ای بنویسید که کانولوشن دو سیگنال را با استفاده از تبدیل فوریه‌ی تک تک آن‌ها محاسبه کند.

# آزمایش ۴

## تحلیل سیستم‌های زمان‌گسسته در حوزه‌ی فرکانس

### هدف آزمایش

می‌دانیم که رفتار هر سیستم LTI، با در دست داشتن پاسخ ضربه‌ی آن به طور کامل مشخص می‌شود. همچنین در آزمایش قبل، تبدیل فوریه‌ی پاسخ ضربه (پاسخ فرکانسی) را بررسی کردیم. در این آزمایش، ابتدا با تبدیل z پاسخ ضربه با نام تابع تبدیل آشنا می‌شویم. سپس ویژگی‌ها و انواع سیستم‌ها از نظر پاسخ فرکانسی را بررسی می‌کنیم. همچنین با محاسبه‌ی پاسخ فرکانسی و پاسخ ضربه با در دست داشتن تابع تبدیل، پایداری سیستم، ویژگی‌های متفاوت سیستم علی و LTI پایدار، بررسی نمودار صفرها و قطب‌های تابع تبدیل و بررسی پایداری با استفاده از این نمودار، آشنا خواهیم شد.

### فرمان‌های مورد نیاز در این آزمایش

در ادامه، با برخی از فرمان‌های متلب که در آزمایش ۴ مورد استفاده قرار خواهد گرفت، آشنا می‌شویم:

فرمان‌های عمومی : General purpose commands:

| |
|---|
| disp |

عملگرها و کاراکترهای خاص : Operators and special characters:

| | | | | | | |
|---|---|---|---|---|---|---|
| : | . | + | - | * | / | ; |

ساختارهای زبانی و اشکال زدایی : Language constructs and debugging:

| | |
|---|---|
| function | pause |

ماتریس‌های اولیه و به کار بردن ماتریس: Elementary matrices and matrix manipulation:

| | |
|---|---|
| fliplr | pi |

توابع اولیه: Elementary functions:

| | | | | | |
|---|---|---|---|---|---|
| abs | angle | imag | Log10 | real | Real |

ابزار ترسیم دو بعدی: Two dimensional graphics:

| | | | | |
|---|---|---|---|---|
| axis | grid | plot | stem | title |
| xlabel | ylabel | | | |

ابزار ترسیم عمومی: General purpose graphics functions:

| | |
|---|---|
| clf | subplot |

جعبه ابزار پردازش سیگنال: Signal processing toolbox:

| | | | | | |
|---|---|---|---|---|---|
| freqz | impz | residuez | Tf2zp | Zp2sos | Zp2tf |
| zplane | | | | | |

### آزمایش۴-۱: رسم قطب‌ها و صفرهای تبدیل z سیستم

میدانیم که تبدیل فوریه گسسته، همان تبدیل z روی دایره‌ی واحد است. یعنی در حالت کلی z عددی مختلط خواهد بود و در ازای $z = e^{jw}$، تبدیل z تبدیل به تبدیل فوریه گسسته خواهد شد. بنابراین، واضح است که دستور freqz تبدیل z را محاسبه می‌کند اما تنها بر روی دایره‌ی واحد.

تمرین۱: دستور zplane را در دو حالت مختلف شرح دهید. ( ورودی تابع تبدیل باشد و یا ورودی بردارهای صفر و قطب باشد).

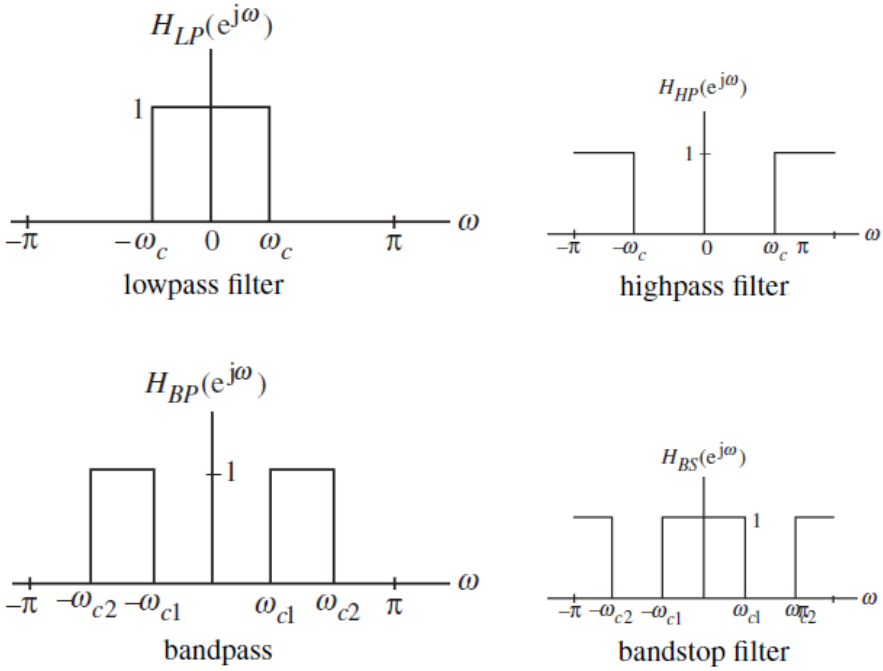
تمرین۲: برنامه‌ای بنویسید که نمودار صفرها و قطب‌های تابع تبدیل زیر را نشان دهد.

$$G(z) = \frac{2 + 5z^{-1} + 9z^{-2} + 5z^{-3} + 3z^{-4}}{5 + 45z^{-1} + 2z^{-2} + z^{-3} + z^{-4}} \quad (۱-۴)$$

تمرین۳: با استفاده از دستور tf2zp، مکان قطب ها و صفرهای این سیستم را مشخص کنید.

### آزمایش ۴-۲: بررسی ویژگی‌های سیستم ها، انواع فیلترها

یکی از مهم‌ترین کاربردهای پردازش سیگنال دیجیتال، فیلتر کردن سیگنال‌های مختلف به منظور حذف اجزای نامطلوب آن-ها است. پاسخ فرکانسی چهار فیلتر ایده‌آل را در شکل ۴-۱ مشاهده می‌کنید.




شکل ٤-١: پاسخ فرکانسی چهار فیلتر ایده آل

در عمل پیاده سازی فیلتر ایدهآل امکان پذیر نیست. بنابراین، از پاسخ ضربه‌ی محدود استفاده میکنیم . همچنین برای پیاده سازی باید فیلتر علی باشد. پاسخ ضربه‌ی تخمین زده شده‌ی فیلتر پایین گذر به صورت زیر است:

$$\hat{h}_{LP}[n] = \frac{sin\big(w_c(n - N/2)\big)}{\pi(n - N/2)} \qquad (۲-۴)$$

دراین آزمایش، این فیلتر را شبیه‌سازی می‌کنیم.

تمرین۴: برنامه‌ی P4-1 را اجرا کنید. این برنامه به منظور ایجاد و رسم پاسخ ضربه‌ی فیلتر پایین گذر نوشته شده است. طول این فیلتر چند است؟ فرکانس قطع (cutoff) فیلتر توسط چه پارامتری کنترل می‌شود؟

تمرین۵: برنامه را طوری ویرایش کنید که پاسخ ضربه‌ی فیلتر FIR پایین گذری با پاسخ ضربه‌ی نشان داده شده در معادله-ی ۴-۲ با طول ۲۰و فرکانس قطع $w_c = 0.45$ را ایجاد و رسم کند.

## آزمایش ۴-۳: محاسبه‌ی گین تابع تبدیل

معمولا گین تابع تبدیل بر حسب دسی بل محاسبه و رسم می‌شود و فرکانس قطع 3-dB آن لحاظ می‌شود. منظور محسابه‌ی گین تابع تبدیل بر حسب دسی‌بل، تابع gain در متلب نوشته شده است و از این تابع، در برنامه‌ی P4-2 برای محاسبه‌ی گین فیلتر پایین گذر moving average استفاده می‌کنیم.

تمرین۶: تابع gain را مشاهده کنید و توضیح دهید چگونه این تابع، گین را محاسبه می‌کند؟

تمرین۷: برنامه‌ی P4-2 را اجرا کنید. با توجه به گین رسم شده، فرکانس قطع این فیلتر پایین گذر چند است؟

رابطه‌ی زیر، نمایان‌گر تابع تبدیل یک فیلتر بالاگذر است.

$$H(z) = \frac{1}{M}\sum_{n=0}^{M-1}(-1)^n z^{-n} \qquad (۳-۴)$$

تمرین۸: با توجه به برنامه‌ی قبل، برنامه‌ای بنویسید که گین تابع تبدیل فیلتر بالاگذر بالا را برای M=5 برحسب دسی‌بل رسم کند. سپس از روی گین رسم شده، فرکانس قطع 3-dB را مشخص کنید.

## آزمایش ۴-۴: بررسی پایداری فیلتر

پایداری فیلترهای IIR، یکی از پارامترهای مهم در طراحی این نوع فیلترها می‌باشد. می‌دانیم زمانی فیلتر پایدار خواهد بود که تمامی قطب‌های تابع تبدیل آن درون دایره‌ی واحد قرار داشته باشند. اگرچه می‌توان با استفاده از دستور zplane، قطب ها را مشاهده و پایداری را تحقیق کرد، اما زمانی که قطب‌ها خیلی نزدیک به دایره‌ی واحد باشند، این برای بررسی پایداری مناسب و دقیق نخواهد بود. با استفاده از دستور poly2rc، میتوانیم پایداری را به صورت دقیق تر بررسی کنیم.

تمرین۹: دو سیستم با تابع تبدیل‌های زیر در نظر بگیرید.

$$H_1(z) = \frac{1}{1 - 1.848z^{-1} + 0.85z^{-2}} \qquad (۴-۴)$$

$$H_2(z) = \frac{1}{1 - 1.851z^{-1} + 0.85z^{-2}} \qquad (۵-۴)$$

برنامه‌ای بنویسید که محل صفرها و قطب‌های این دو تابع تبدیل را نشان دهد. آیا می‌توانید مشخص کنید که کدام سیستم پایدار است؟

تمرین۱۰: حال با استفاده از برنامه‌ی P4-4 پایداری دو سیستم را بررسی کنید و نتیجه را گزارش کنید.

# آزمایش ۵

# طراحی فیلترهای دیجیتال IRR

## هدف آزمایش

طراحی فیلتر دیجیتال به معنی به دست آوردن تابع تبدیلی است که پاسخ فرکانسی آن، تقریبی از پاسخ فرکانسی مطلوب باشد. پس از طراحی تابع تبدیل، آن را پیاده سازی می‌کنند. در آزمایش قبل با پیاده سازی فیلترها با در دست داشتن تابع تبدیل آشنا شدیم. اما در این آزمایش و آزمایش بعدی، با اصول طراحی این توابع تبدیل برای دستیابی به گین مشخص آشنا خواهیم شد. در این آزمایش، طراحی فیلترهای IIR را بررسی خواهیم کرد.

## فرمان‌های مورد نیاز در این آزمایش

در ادامه، با برخی از فرمان‌های متلب که در آزمایش ۵ مورد استفاده قرار خواهد گرفت، آشنا می‌شویم:

فرمان‌های عمومی : General purpose commands:

| | |
|---|---|
| disp | length |

عملگرها و کاراکترهای خاص : Operators and special characters:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| % | ; | / | * | - | + | . | : |
| | | | | == | > | ./ | .* |

ساختارهای زبانی و اشکال زدایی : Language constructs and debugging:

| | | |
|---|---|---|
| else | function | if |

ماتریس‌های اولیه و به کار بردن ماتریس: Elementary matrices and matrix manipulation:

| | | | |
|---|---|---|---|
| fliplr | nargin | pi | : |

توابع اولیه: Elementary functions:

| | | | | | |
|---|---|---|---|---|---|
| abs | ceil | cos | Log10 | sin | Sqrt |

آنالیز داده: Data analysis:

| |
|---|
| min |

ابزار ترسیم دو بعدی: Two dimensional graphics:

| | | | | |
|---|---|---|---|---|
| axis<br>ylabel | grid | plot | title | xlabel |

جعبه ابزار پردازش سیگنال: Signal processing toolbox:

| | | | | | | |
|---|---|---|---|---|---|---|
| blackman | butter | buttord | chebwin | cheb1ord | Cheb2ord | Cheby1 |
| Cheby2 | ellip | ellipord | fir1 | fir2 | firpm | firpmord |
| freqz | hanning | hamming | kaiser | | | |

معروف‌ترین روش طراحی فیلتر IIR، بر مبنای تبدیل bilinear از تابع تبدیل پیوسته نمونه است. تابع تبدیل آنالوگ معمولا یکی از چهار نوع باتروورث (Butterworth)، چبی‌شف (Chebyshev) نوع۱ ، چبی شف نوع۲، و elliptic است.

**آزمایش ۵-۱: تعیین درجه‌ی فیلتر IIR**

اولین قدم پس از انتخاب نوع فیلتر مورد نیاز، تعیین درجه‌ی تابع تبدیل آن بر اساس ویژگی‌های مورد نیاز می‌باشد. فرامین زیر در متلب به منظور تعیین درجه‌ی چهارنوع فیلتر ذکر شده، قرار داده شده است.

$$[N, W_n] = buttord(W_P, W_S, R_P, R_S),$$

$$[N, W_n] = cheb1ord(W_P, W_S, R_P, R_S),$$

$$[N, W_n] = cheb2ord(W_P, W_S, R_P, R_S),$$

$$[N, W_n] = ellipord(W_P, W_S, R_P, R_S),$$

تمرین۱: توضیح دهید که هر یک از پارامترهای فرامین بالا نشان‌گر چیست؟

تمرین۲: برنامه‌ای بنویسید که کمترین درجه‌ی تابع تبدیل فیلتر پایین گذر IIR از هر چهار نوع ذکر شده را با مشخصات زیر به دست آورد:

نرخ نمونه برداری 40kHz ، فرکانس مرز باند عبور 4kHz ، فرکانس مرز باند توقف 8kHz، ریپل باند عبور 0.5dB و حداقل میرایی باند توقف 40dB

## آزمایش ۵-۲: طراحی فیلتر IIR

بعد از انتخاب نوع فیلتر و تعیین درجه‌ی آن، باید تابع تبدیل فیلتر را محاسبه کنیم. توابع زیر در متلب به منظور محاسبه‌ی تابع تبدیل با درجه‌ی مطلوب، قرار داده شده اند:

$$[num, then] = butter(N, W_n)$$

$$[num, then] = butter(N, W_n, 'high')$$

$$[num, then] = butter(N, W_n, 'stop')$$

$$[num, then] = cheby1(N, R_P, W_n)$$

$$[num, then] = cheby1(N, R_P, W_n, 'filtertype')$$

$$[num, then] = cheby2(N, R_S, W_n)$$

$$[num, then] = cheby2(N, R_S, W_n, 'filtertype')$$

$$[num, then] = ellip(N, R_P, R_S, W_n)$$

$$[num, then] = ellip(N, R_P, R_S, W_n, 'filtertype')$$

تمرین۳: پارامترها و عملکرد هر یک از توابع بالا را توضیح دهید.

تمرین۴: برنامه‌ی P7-1 طراحی یک فیلتر باتروورث را نشان می‌دهد. آن را اجرا کنید. این برنامه چه نوع فیلتری از لحاظ باند عبور طراحی می‌کند؟ مشخصات دقیق فیلتر طراحی شده چیست؟

تمرین۵: برنامه‌ای بنویسید که یک فیلتر بالاگذر چبی شف نوع ۲ را با مشخصات زیر، طراحی کند.

نرخ نمونه برداری 3.5 Hz، فرکانس مرز باند عبور 1050Hz، فرکانس مرز باند توقف 600Hz، ریپل باند عبور 1dBو حداقل میرایی باند توقف 50dB.

# آزمایش ۶

# طراحی فیلترهای دیجیتال FIR

## هدف آزمایش

اهمیت طراحی فیلترهای دیجیتال در آزمایش قبل ذکر شد. همچنین آموختیم که به منظور طراحی، پس از انتخاب نوع فیلتر، باید با توجه به مشخصات مطلوب، درجه‌ی تابع تبدیل را تعیین کنیم و پس از آن تابع تبدیل را به دست آوریم. طراحی چهار نوع از پرکاربردترین فیلترهای دیجیتال IIR در آزمایش قبل بررسی شد و حال همین مراحل را این‌بار برای فیلترهای FIR، بررسی می‌کنیم.

## فرمان‌های مورد نیاز در این آزمایش

در ادامه، با برخی از فرمان‌های متلب که در آزمایش ۵ مورد استفاده قرار خواهد گرفت، آشنا می‌شویم:

General purpose commands: فرمان‌های عمومی :

| | |
|---|---|
| disp | length |

Operators and special characters: عملگرها و کاراکترهای خاص :

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| : | . | + | - | * | / | ; | % |
| .* | ./ | > | == | | | | |

Language constructs and debugging: ساختارهای زبانی و اشکال زدایی :

| | | |
|---|---|---|
| else | function | if |

Elementary matrices and matrix manipulation: ماتریس‌های اولیه و به کار بردن ماتریس:

| | | | |
|---|---|---|---|
| fliplr | nargin | pi | : |

Elementary functions: توابع اولیه:

| | | | | | |
|---|---|---|---|---|---|
| abs | ceil | cos | Log10 | Sin | Sqrt |

Data analysis: آنالیز داده:

| |
|---|
| min |

Two dimensional graphics: ابزار ترسیم دو بعدی:

| | | | | |
|---|---|---|---|---|
| axis<br>ylabel | grid | plot | Title | xlabel |

جعبه ابزار پردازش سیگنال: Signal processing toolbox:

| | | | | | | |
|---|---|---|---|---|---|---|
| blackman | butter | buttord | chebwin | cheb1ord | Cheb2ord | Cheby1 |
| Cheby2 | ellip | ellipord | fir1 | fir2 | Firpm | firpmord |
| freqz | hanning | hamming | kaiser | | | |

ساده ترین روش برای طراحی فیلتر FIR، این است که از پاسخ فرکانسی فیلتر مطلوب، تبدیل فوریه‌ی معکوس گرفته و سپس تعداد محدودی از نقاط آن را انتخاب کنیم. اما این کار باعث می‌شود ریپل‌هایی در پاسخ فرکانسی به وجو آید که اثر Gibbs نام دارد. برای حذف این و یا کاهش این اثر، پاسخ ضربه نامحدود را از پنجره‌ی محدود و مناسب عبور می‌دهیم. در متلب فرامینی تعبیه شده و با استفاده از آن‌ها، می‌توانیم بردار ضرایب پاسخ ضربه فیلتر FIR با مشخصات مطلوب را داشته باشیم. برخی از این فرامین در بخش‌های بعد آورده شده است.

**آزمایش ۶-۱: تعیین درجه‌ی فیلتر FIR**

در آزمایش قبل با استفاده از فرامین موجود در متلب، ابتدا درجه‌ی تابع تبدیل فیلتر IIR مطلوب و سپس خود فیلتر را به دست آوردیم. برای تعیین درجه‌ی فیلتر FIR میتوان از فرمول Kaiser استفاده کرد که فرامین آن در جعبه ابزار پردازش سیگنال به صورت زیر است:

$$[N, W_n, beta, ftype] = kaiserord(fedge, aval, dev)$$

$$[N, W_n, beta, ftype] = kaiserord(fedge, aval, dev, FT)$$

تمرین۱: مشخص کنید که هر کدام از فرامین بالا چه خروجی تولید می‌کند و پارامترهای استفاده شده در آنها را توضیح دهید.

تمرین۲: با استفاده از دستور kaiserord، درجه‌ی فیلتر FIR پایین گذر با فاز خطی و دارای مشخصات زیر را پیدا کنید. فرکانس مرز باند عبور 2kHz، فرکانس مرز باند توقف 2.5kHz، ریپل باند عبور و باند توقف $\delta_p = \delta_s = 0.005$ و فرکانس نمونه برداری 10kHz.

به عنوان یک روش دیگر برای تعیین درجه‌ی فیلتر FIR، تابع kaiord نوشه شده است و می‌توان از آن استفاده کرد.

تمرین۳: تابع kaiord را مشاهده کنید. پارامترهای ورودی و خروجی این تابع چیست؟ دلیل استفاده ازفرمان‌های ceil و nargin را در این تابع بیان کنید.

تمرین۴: درجه‌ی فیلتر با مشخصات خواسته شده در تمرین۲ را این بار با استفاده از تابع kaiord محاسبه کنید و با نتیجه‌ی به دست آمده از تمرین ۲، مقایسه کنید.

## آزمایش ۶-۲: طراحی فیلتر FIR

پس از محاسبه‌ی درجه‌ی فیلتر، با استفاده از زیر می‌توان به بردار ضرایب پاسخ ضربه‌ی فیلتر مورد نظر دست یافت:

$$b = fir1(N, W_n)$$

$$b = fir1(N, W_n, 'high')$$

$$b = fir1(N, W_n, 'stop')$$

$$b = fir1(N, W_n, taper)$$

تمرین۵: مشخص کنید که هر کدام از فرامین بالا چه خروجی تولید می‌کند و پارامترهای استفاده شده در آنها را توضیح دهید.

نکته: همان‌گونه که پیش‌تر توضیح داده شد، برای حذف اثر نوسان در فیلتر FIR از پنجره استفاده می‌کنیم. در حالت استفاده بدون taper، فرمان به صورت پیش‌فرض از پنجره‌ی همینگ hamming استفاده خواهد کرد. اما در صورت نیاز به استفاده از پنجره های دیگر، می‌توان با فرمان‌های زیر، پنجره‌های دیگری نیز ایجاد کرد.

$$taper = blackman(N) \qquad taper = hamming(N) \qquad taper = hanning(N)$$

$$taper = chebwin(N) \qquad taper = kaiser(N, beta)$$

در ادامه فرامین دیگری از این نوع فیلتر را مشاهده می‌کنید.

$$b = fir2(N, fpts, mval)$$

$$b = fir2(N, fpts, mval, mval, taper)$$

تمرین۶: مشابه تمرین قبل، خروجی و پارامترهای فرامین بالا را نیز بیان کنید.

تمرین ۷: با استفاده از دستورfir1 ، فیلتر پایین گذر FIR با فاز خطی طراحی کنید که ویژگی های زیر را داشته باشد.

فرکانس مرز باند عبور 2kHz، فرکانس مرز باند توقف 2.5kHz، ریپل باند عبور و باند توقف $\delta_p = \delta_s = 0.005$ و فرکانس نمونه برداری 10kHz.

پس از طراحی فیلتر، گین پاسخ فرکانسی و فاز آن را رسم کنید. (از درجه‌ی محاسبه شده در تمرین ۲ استفاده کنید). ضرایب فیلتر را در جدولی نمایش دهید. آیا ویژگی‌های فیلتر طراحی شده با ویژگی‌های مطلوب تطابق دارد؟ اگر تطابق ندارد درجه‌ی فیلتر را تنظیم کنید تا زمانی که مشخصات طراحی شده، منطبق بر مشخصات مطلوب باشد.

تمرین۸: تمرین قبل را این بار با هر یک از پنجره‌های hanning، blackman طراحی کنید و حاصل را با تمرین قبل مقایسه کنید.

# آزمایش ۷

## نمونه برداری از سیگنال پیوسته، افزایش و کاهش نرخ نمونه برداری

### هدف آزمایش

به طور معمول در عمل سیگنال‌های پیوسته‌ی زیادی مانند سیگنال صوت وجود دارند. اما سیستم‌های پردازش دیجیتال و یا شبیه‌سازی کامپیوتری نیازمند داده‌های گسسته می‌باشد. بدین ترتیب، نمونه برداری از سیگنال‌های پیوسته انجام می‌شود تا پردازش‌های لازم در حالت گسسته بر روی آن‌ها اعمال شود و سپس تبدیل به حالت پیوسته شود. نمونه برداری از تابع پیوسته با استفاده از یک سیگنال پریودیک صورت می‌گیرد. در این آزمایش، اثر نمونه برداری را در حوزه‌ی زمان و حوزه‌ی فرکانس بر روی سیگنال مشاهده خواهیم کرد. همچنین روش‌های بازسازی سیگنال پیوسته از روی نمونه‌های گسسته‌ی آن ضمن حفظ اطلاعات سیگنال، بررسی می‌شود.

در مبحث پردازش سیگنال گسسته، گاه نیاز است که نرخ نمونه برداری را توسط یک ضریب افزایش و یا کاهش دهیم. عمل افزایش نرخ نمونه برداری upsampling و عمل کاهش آن downsamlping نام دارد. در این آزمایش با ویژگی‌های این دو عملگر و تاثیر آن‌ها بر سیگنال در حوزه‌ی زمان و حوزه‌ی فرکانس، آشنا خواهیم شد.

فرمان‌های عمومی : General purpose commands:

| | |
|---|---|
| size | length |

عملگرها و کاراکترهای خاص : Operators and special characters:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| : | . | + | - | * | / | ; | % |
| == | ~ | & | \| | .* | | | |

ساختارهای زبانی و اشکال زدایی : Language constructs and debugging:

| | | | | |
|---|---|---|---|---|
| else | function | if | end | for |

ماتریس‌های اولیه و به کار بردن ماتریس: Elementary matrices and matrix manipulation:

| | | | |
|---|---|---|---|
| ‘ | ones | linespace | pi |

توابع اولیه: Elementary functions:

| | | | | |
|---|---|---|---|---|
| abs | cos | exp | sin | Log10 |

آنالیز داده: Data analysis:

| |
|---|
| min |

ابزار ترسیم دو بعدی: Two dimensional graphics:

| | | | | |
|---|---|---|---|---|
| axis<br>ylabel | stem | plot | Title | xlabel |

جعبه ابزار پردازش سیگنال: Signal processing toolbox:

| | | | | | | |
|---|---|---|---|---|---|---|
| freqs | freqz | sinc | hamming | Fir2 | resample | sinc |

## آزمایش ۷-۱: نمونه برداری از سیگنال سینوسی

از آنجایی که در متلب سیگنال با مقادیر پیوسته نمی‌توان ایجاد کرد، از یک سیگنال سینوسی با نرخ خیلی بالا به عنوان سیگنال پیوسته استفاده می‌کنیم. سپس از این سیگنال نمونه برداری خواهیم کرد.

تمرین۱: برنامه‌ی P5-1 را اجرا کنید. فرکانس سیگنال سینوسی و دوره‌ی نمونه برداری چند است؟ چرا از فرمان axis در این برنامه استفاده شده است؟ نقش آن را بیان کنید.

تمرین۲: به ازای نرخ‌های مختلف، نمونه برداری را انجام دهید و نتیجه را گزارش کنید.

تمرین۳: برای دو فرکانس دیگر از سیگنال سینوسی با مقادیر 3Hz و 5Hz، نمونه برداری را انجام دهید. چه تفاوتی میان سیگنال‌های نمونه برداری شده‌ی جدید است؟ توضیح دهید.

## آزمایش ۷-۲: بررسی رابطه بین نرخ نمونه برداری و فرکانس سیگنال

پس از نمونه‌برداری از سیگنال با نرخ‌های متفاوت، سیگنال پیوسته را از نمونه‌های آن بازسازی می‌کنیم. می‌دانیم که در صورتی که نرخ نمونه برداری کمتر از نرخ نایکوئیست باشد، سیگنال به طور دقیق بازسازی نخواهد نشده و دچار تخریب خواهد شد که به آن aliasing می‌گویند. در این آزمایش این موضوع را بررسی خواهیم کرد.

تمرین۴: برنامه‌ی P5-2 را اجرا کنید و خروجی را مشاهده کنید. سپس فرکانس سیگنال ورودی را به مقایر 3Hz و 5Hz تغییر داده و خروجی را مشاهده کنید. آیا سیگنال بازسازی شده تغییر کرده است؟ چرا؟

تمرین۵: به ازای فرکانس ورودی دلخواه و نرخ‌های نمونه برداری متفاوت، حد نایکوییست را تحقیق کنید.

## آزمایش ۷-۳: بررسی اثر تخریب بازسازی در حوزه‌ی فرکانس

در این آزمایش، اثر aliasing را در حوزه‌ی فرکانس بررسی خواهیم کرد.

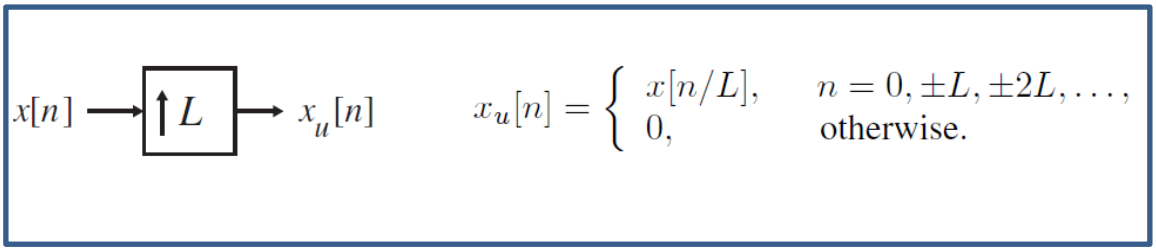
تمرین ۶: برنامه‌ی P5-3 را اجرا کنید. آیا اثر تخریب در حوزه‌ی فرکانس مشهود است؟ توضیح دهید.

تمرین۷: نرخ نمونه برداری را کم کم زیاد کرده و خروجی را مشاهده کنید. سپس نرخ نمونه برداری مناسب را به صورت تقریبی بیابید.

تمرین۸: برنامه را به ازای $x_a(t) = e^{-\pi t^2}$ اجرا کنید و اثر aliasing را بررسی کنید.

## آزمایش ۷-۴: بررسی افزایش نرخ نمونه برداری (upsampling) در حوزه‌ی زمان وفرکانس

افزایش نرخ نمونه برداری به صورت زیر انجام می‌شود:




شکل ۷-۱: نمایش عمل افزایش نرخ نمونه برداری

تمرین۹: برنامه‌ی P10-1 را اجرا کنید. عملیات افزایش نرخ چگونه و با چه مقداری انجام شده است؟

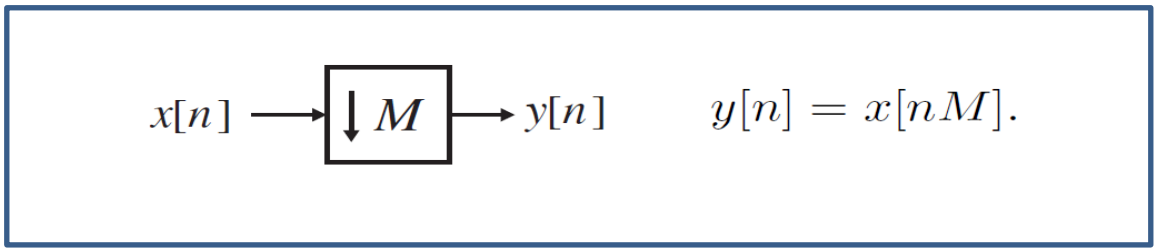
تمرین۱۰: به ازای دو مقدار دخواه از L، رابطه‌ی بین سیگنال ورودی و خروجی را تحقیق کنید.

تمرین۱۱: برنامه‌ی P10-3 را به ازای مقادیر L=2، L=3 و L=5 اجرا کرده و خروجی را مشاهده کنید. چه نتیجه ای می-گیرید؟ آیا می‌توانید رابطه‌ی بین طیف ورودی و خروجی را بیابید؟

## آزمایش ۷-۵: بررسی کاهش نرخ نمونه برداری (downsampling) در حوزه‌ی زمان وفرکانس

مشابه آزمایش قبل، این‌بار کاهش نرخ نمونه برداری را در حوزه‌ی زمان و فرکانس بررسی می‌کنیم.

کاهش نرخ نمونه برداری به صورت زیر است:




شکل ۷-۲: شکل ۷-۱: نمایش عمل کاهش نرخ نمونه برداری

تمرین۱۲: برنامه‌ی P10-2 را اجرا کنید و عملیات کاهش نرخ را توضیح دهید.

تمرین۱۳: برنامه‌ی را به ازای مقادیر مختلف M اجرا کرده و خروجی را مشاهده کنید. چه نتیجه ای می‌گیرید؟

تمرین۱۴: برنامه‌ی P10-4 را به ازای مقادیر M=2، M=3 و M=5 اجرا کرده و خروجی را مشاهده کنید. چه نتیجه ای می‌گیرید؟ آیا می‌توانید رابطه‌ی بین طیف ورودی و خروجی را بیابید؟

# آزمایش ۸

## مقدمه‌ای بر پردازش تصویر دیجیتال

### هدف آزمایش

امروزه با پیشرفت تکنولوژی و گسترش استفاده از دوربین‌های دیجیتال، حجم تصاویر ذخیره شده بر روی دستگاه‌های حافظه، روز به روز در حال افزایش است. لذا به منظور استفاده‌ی مطلوب از این تصاویر، پردازش تصویر دیجیتال اهمیت ویژه ای پیدا کرده است. هر تصویر دیجیتال در متلب، آرایه‌ای دو بعدی و یا سه بعدی (برای تصاویر رنگی) از اعداد است. در این آزمایش با خواندن تصاویر دیجیتال در متلب، عملیات‌های ابتدایی پردازش بر روی تصاویر، ترکیب و ذخیره‌ی تصاویر آشنا خواهیم شد.

فرمان‌های عمومی : General purpose commands:

| | |
|---|---|
| size | length |

عملگرها و کاراکترهای خاص : Operators and special characters:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| : | . | + | - | * | / | ; | % |
| == | ~ | & | \| | .* | | | |

ساختارهای زبانی و اشکال زدایی : Language constructs and debugging:

| | | | | |
|---|---|---|---|---|
| else | function | if | end | for |

آنالیز داده: Data analysis:

| | | |
|---|---|---|
| min | max | mean |

ابزار ترسیم دو بعدی: Two dimensional graphics:

| | | | | |
|---|---|---|---|---|
| axis | stem | plot | Title | xlabel |
| ylabel | | | | |

جعبه ابزار پردازش تصویر : image processing toolbox:

| | | | | | | |
|---|---|---|---|---|---|---|
| imread | imwrite | imfinfo | imnoise | Mat2gray | Rgb2gray | Im2uint8 |
| im2uint16 | imhist | | | | | |

## آزمایش ۸-۱- خواندن تصویر و محاسبه‌ی هیستوگرام

با استفاده از تصاویر قرار داده شده در فولدر مربوطه، تمارین زیر را انجام دهید.

تمرین۱: با استفاده از دستور imread، تصویر موجود در فولدر آزمایش به نام image1.tif را فراخوانده و در متغیر a ذخیره کنید. ابعاد آرایه‌ی تصویر چند است و نوع داده‌های آن چیست؟ تصویر a را با استفاده از دستور imshow مشاهده و سپس هیستوگرام آن را با استفاده از دستور imhist رسم کنید.

تمرین۲: مشابه تمرین قبل، همین مراحل را برای تصویر image2.tif تکرار کنید. هیستوگرام این تصویر چه تفاوتی با هیستوگرام تصویر تمرین قبل دارد؟ با توجه به تعریف هیستوگرام، این تفاوت را چگونه توجیه می‌کنید؟

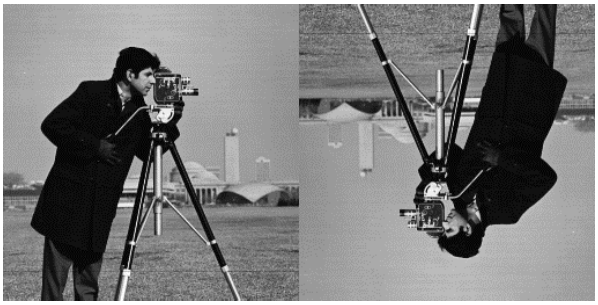
تمرین۳: تصویر onion.png را فراخوانی کرده و در متغیر a ذخیره کنید. ابعاد آرایه‌ی این تصویر چند است؟ تصویر را مشاهده کنید. پیکسل‌های مربوط به کانال‌های G،R و B را به صورت جداگانه در سه متغیر ذخیره کرده و هیستوگرام هر کدام را رسم کنید.

## آزمایش ۸-۲- اعمال تبدیل روی تصاویر

در این آزمایش، با چرخش، برش و تغییر مقیاس تصویر آشنا خواهید شد.

تمرین۴: تصویر image.tif را فراخوانی کرده، ابعاد آن را به مقدار ۱۰۰*۱۰۰ تغییر دهید و تصویر جدید را مشاهده کنید. هیستوگرام تصویر جدید را با هیستوگرام اولیه مقایسه کنید. آیا تفاوتی مشاهده می‌شود؟ چرا؟

تمرین۵: تصویر cameraman.tif را فراخوانی کنید. سپس برنامه ای بنویسید که تصویر زیر را ایجاد کرده و آن را ذخیره کند.



شکل ۸-۱ تصویر خروجی مطلوب

## آزمایش ۸-۳- تغییر تعداد سطوح خاکستری تصویر

می‌دانیم که کیفیت یک تصویر خاکستری، وابسته به تعداد سطوح خاکستری (gray levels) آن است. با افزایش این تعداد سطوح، تغییرات روشنایی میان پیکسل‌ها، نرم‌تر خواهد شد. با کاهش تعداد سطوح روشنایی، تعداد بیت کمتری برای ذخیره-سازی هر پیکسل نیاز است و با کاهش تعداد سطوح، حجم تصویر کم‌تر شده و کیفیت آن نیز کاهش می‌یابد. به عنوان مثال، در شکل زیر تصویر cameraman با ۲۵۶ سطح خاکستری و تصویر تبدیل شده‌ی آن به دو سطح را مشاهده می‌کنید. کد این برنامه تحت عنوان P3_8 در فولدر مربوط به آزمایش قرار دارد.

(الف) (ب)

شکل ۸-۲ (الف) تصویر با ۲۵۶ سطح خاکستری و (ب) تصویر تبدیل شده به ۲ سطح خاکستری

تمرین ۶: برنامه‌ی p3_8 را اجرا کنید. تصویر اولیه و تصویر تبدیل شده را به صورت جداگانه با استفاده از دستور imwrite ذخیره و حجم هر کدام را مشاهده کنید.

تمرین۷: با توجه به برنامه‌ی فوق، تابعی بنویسید که تصویر ورودی و تعداد سطوح روشنایی ( یکی از اعداد ۲،۴ و ۸) را از کاربر دریافت کند و تصویر با تعداد سطح روشنایی جدید را ذخیره کند. توجه کنید که تابع باید به گونه ای باشد تا اگر تصویر ورودی رنگی بود، ابتدا آن را به تصویر خاکستری تبدیل کند.

# آزمایش ۹

## فیلتر کردن تصاویر دیجیتال و بهبود تصاویر

اعمال فیلترهای گوناگون بر روی تصاویر دیجیتال، از اهمیت ویژه‌ای برخوردار است. دو نمونه از مهم‌ترین کاربردهای پردازش تصویر، حذف نویز از تصویر و لبه‌یابی است که در اکثر زمینه‌ها بخصوص حذف نوز از تصاویر پزشکی و تصاویر ماهواره‌ای و.. از اهمیت ویژه‌ای برخوردار می‌باشند. در این آزمایش، ابتدا بهبود کیفیت تصویر از طریق هیستوگرام آن را بررسی و پس از آشنایی با روش اعمال فیلتر بر روی تصاویر، حذف نویز و لبه یابی را با استفاده از فیلترها تمرین خواهیم کرد.

فرمان‌های عمومی : General purpose commands:

| | |
|---|---|
| size | length |

عملگرها و کاراکترهای خاص : Operators and special characters:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| : | . | + | - | * | / | ; | % |
| == | ~ | & | \| | .* | | | |

ساختارهای زبانی و اشکال زدایی : Language constructs and debugging:

| | | | | |
|---|---|---|---|---|
| else | function | if | end | for |

آنالیز داده: Data analysis:

| | | |
|---|---|---|
| min | max | mean |

ابزار ترسیم دو بعدی: Two dimensional graphics:

| | | | | |
|---|---|---|---|---|
| axis | stem | plot | Title | xlabel |
| ylabel | | | | |

جعبه ابزار پردازش تصویر : image processing toolbox:

| | | | | | | |
|---|---|---|---|---|---|---|
| imread | imwrite | imfinfo | imnoise | Mat2gray | Rgb2gray | Im2uint8 |
| im2uint16 | imhist | imfilter | histeq | Filter2 | Medfilt2 | fspecial |

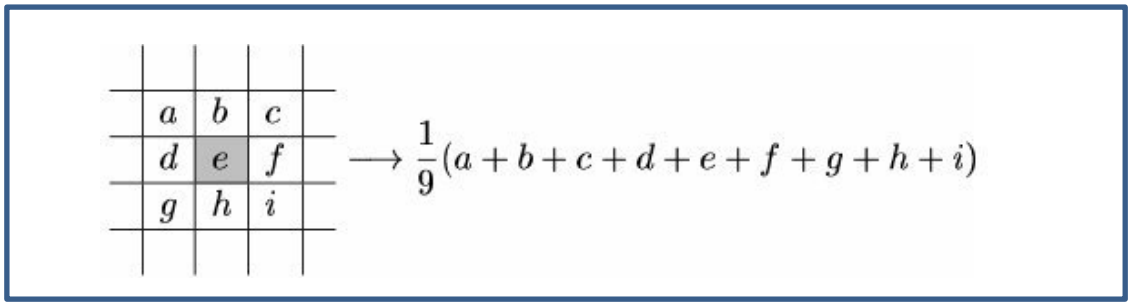
## آزمایش ۹-۱: یکنواخت سازی هیستوگرام تصویر

یکنواخت کردن هیستوگرام یک روش معمول افزایش کنتراست تصاویر است. بدین منظور می‌توان در متلب از دستور histeq استفاده کرد.

تمرین ۱: تصویر image9_1 موجود در فولدر آزمایش را فراخوانی کرده و مشاهده کنید. سپس هیستوگرام تصویر را مشاهده کنید. حال دستور histeq را بر روی تصویر اعمال کرده و خروجی و هیستوگرام آن را مشاهده کنید. آیا کیفیت تصویر بهتر شده است؟ چرا؟

## آزمایش ۹-۲: حذف نویز از تصویر

به منظور حذف نویزهای با فرکانس مکانی بالا، از فیلتر پایین گذر استفاده می‌کنیم تا تصویر اصطلاحا نرم‌تر شود. فیلتر کردن تصویر بدین صورت است که پنجره‌ای با ابعاد معین بر روی تصویر حرکت کرده و روی پیکسل‌ها تاثیر می‌گذارد. ساده‌ترین فیلتر که به منظور حذف نویز استفاده می‌شود، فیلتر میانگین است. در این نوع از فیلتر، به جای هر پیکسل در تصویر، میانگین پیسل‌های همسایگی آن قرار داده می‌شود. به عنوان مثال، عملکرد این فیلتر را برای پنجره‌ی با ابعاد سه در سه را در شکل زیر مشاهده می‌کنید.




شکل ۹-۱ فیلتر میانگین با ابعاد ۳*۳

در این آزمایش، اثر این فیلتر را بر تصویر بررسی خواهیم کرد.

تمرین۲: برنامه‌ای بنویسید که برای تصویر نمونه‌ی image9-1 موجود در فولدر آزمایش، فیلتر میانگین با ابعاد ۳*۳ را اعمال کند. در این تمرین از توابع آماده‌ی متلب استفاده نکنید و برنامه را با استفاده از حلقه بنویسید.

تمرین۳: تصویر cameraman.tif را بارگذاری کرده و با استفاده از دستور imnoise، آن را به نویز نمک و فلفل آغشته کنید، به نحوی که ۲۵٪ از پیکسل‌ها معیوب شوند. تصویر نویزی را مشاهده کنید. حال ماتریس h را با ابعاد ۳*۳ ساخته و با استفاده از دستور filter2 ، آن را به تصویر اعمال کنید. تصویر خروجی را با تصویر اولیه در یک figure نشان داده و با هم مقایسه کنید. آیا نویز به خوبی حذف شده است؟ همین تمرین را با مقادیر مختلفی از درصد نویز و ابعاد پنجره امتحان کنید و نتایج خود را بیان کنید. آیا افزایش ابعاد فیلتر میانگین برای حذف این نویز سودمند است؟ چرا؟

به منظور حذف بهتر نویز، از فیلتر میانه میتوانیم استفاده کنیم. در این فیلتر، مقادیر خیلی کم و خیلی زیاد در یک رشته از اعداد حذف خواهند شد. در متلب برای اعمال فیلتر میانه دستور medfilt2 وجود دارد.

تمرین۴: تصویر cameraman.tif را دوباره آغشته به نویز نمک و فلفل کنید و این بار فیلتر میانه را اعمال کنید. تصویر ورودی و خروجی را در یک figure رسم کنید. آیا این فیلتر به خوبی توانسته است نویز را حذف کند؟ به نظر شما چرا این فیلتر بهتر از فیلتر میانگین در حذف این نمونه از نویز عمل می‌کند؟

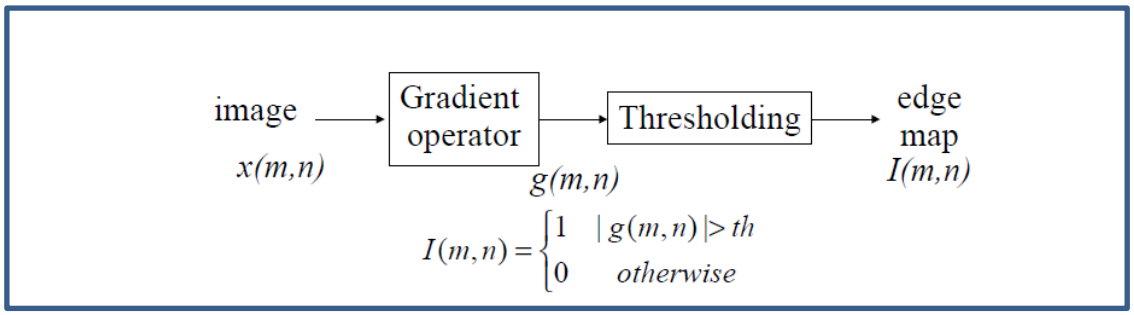
با استفاده از دستور fspecial(type) می‌توان ماتریس پنجره‌ی آماده برای فیلتر های مختلف را ایجاد کرد. تعدادی از این فیلترها را در جدول زیر مشاهده می‌کنید.

جدول ۱ چند نمونه از فیلترهای پردازش تصویر

| نوع | توضیحات |
|---|---|
| **average** | فیلتر میانگین |
| **disk** | فیلتر میانگین دایروی |
| **gaussian** | فیلتر گوسی پایین گذر |
| **laplacian** | تقریب دوبعدی عملگر لاپلاسین |
| **log** | لاپلاس فیلتر گوسی |
| **motion** | تقریب حرکت خطی دوربین |
| **prewitt** | مشخص کردن لبه |
| **sobel** | مشخص کردن لبه |

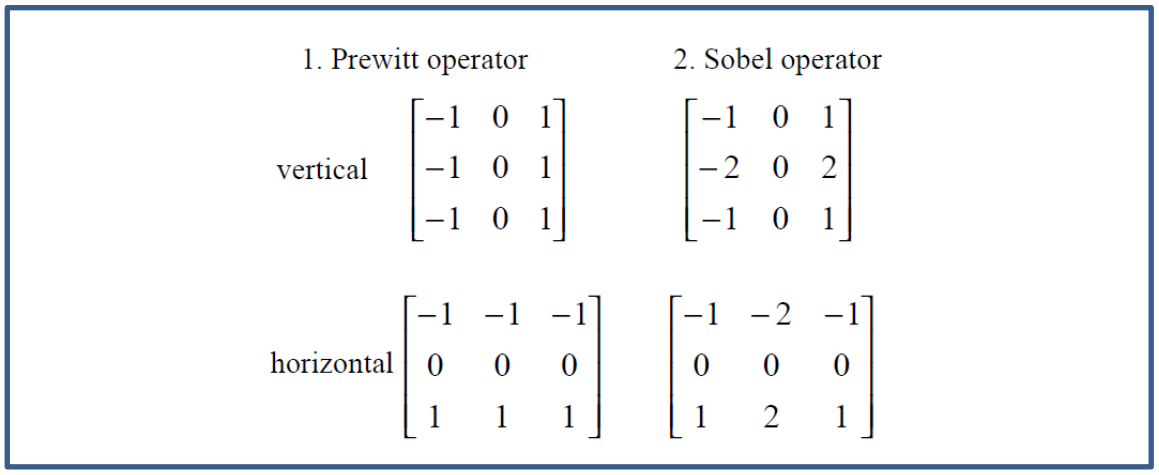
## آزمایش ۹-۳: لبه‌یابی

مشخص کردن لبه‌های اجسام درون یک تصویر، از اهمیت ویژه‌ای در پردازش تصویر برخوردار است. بخصوص در ردیابی و فشرده‌سازی کاربردهای فراوانی دارد. لبه‌ها در تصاویر دارای فرکانس‌های بالا هستند. فرکانس بالا در تصویر به معنای تغییرات شدید مقدار پیکسل‌ها در فاصله‌ی کم است. بنابراین، با محاسبه‌ی گرادیان و آستانه‌گذاری می‌توان لبه‌ها را مشخص کرد. اما در متلب فیلترهای اماده ای نسز وجود دارند که توسط آن‌ها، مکان لبه‌ها به خوبی مشخص خواهد شد.




شکل ۹-۲ منطق عملکرد لبه یابی

به منظور تعیین لبه ها در نرم افزار متلب، می‌توانیم از فیلترهای زیر استفاده کنیم.




شکل ۹-۳ فیلترهای معروف لبه‌یابی عمودی و افقی

تمرین ۵: تصویر cameraman.tif را فراخوانی کرده و لبه‌های افقی آن را با استفاده از فیلترهای sobel و prewitt آشکار کنید. کدام‌یک از دوفیلتر بهتر عمل می‌کند؟

# آزمایش ۱۰

## آشنایی با سیگنال‌های صوتی

یکی دیگر از کاربردهای پردازش سیگنال دیجیتال، پردازش سیگنال‌های صوتی و پردازش گفتار می‌باشد. پردازش صوت از دیرباز از اهمیت فروانی برخوردار بوده است، بخصوص در فشرده سازی، بهبود کیفیت صدا و حذف نویز، تقویت سیگنال صوتی و کدینگ صوت. لذا آشنایی با اصول اولیه پردازش صوت در متلب، خالی از لطف نیست. در این آزمایش، با اصول اولیه‌ی کار با سیگنال‌های صوتی در نرم افزار متلب آشنا خواهید شد.

فرمان‌های عمومی : General purpose commands:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| size | length | | | | | | |

عملگرها و کاراکترهای خاص : Operators and special characters:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| : | . | + | - | * | / | ; | % |
| == | ~ | & | \| | .* | | | |

ساختارهای زبانی و اشکال زدایی : Language constructs and debugging:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| else | function | if | end | for | | | |

آنالیز داده و توابع تبدیل فوریه: Data analysis and fourier transform functions:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| fft | ifft | max | min | | | | |

ابزار ترسیم دو بعدی: Two dimensional graphics:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| axis | stem | plot | Title | xlabel | ylabel | | |

جعبه ابزار پردازش صوت : image processing toolbox:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| audioread | audioplay | audiowrite | | | | | |

## آزمایش ۱۰-۱: ایجاد یک رشته‌ی صوتی

می‌دانیم که پس از نمونه برداری از یک سیگنال صوتی با یک فرکانس نمونه برداری مشخص در نرم افزار متلب، حاصل به صورت ی رشته‌ی برداری ذخیره می‌شود. بنابراین، هر بردار با یک ستون ( و یا دو ستون برای دو کانال صوتی) می‌تواند به عنوان یک صوت شناخته شود. دستور sound بردار را به صورت سیگنال صوتی به اسپیکر ارسال می‌کند، در حالی که فرکانس نمونه برداری به صورت پیش فرض 8192Hz در نظر گرفته شده است. فرکانس نمونه برداری دلخواه را می‌توان به عنوان ورودی به صورت sound(a,fs) وارد کرد.

تمرین ۱: بردار سینوسی زیر را در نظر بگیرید:

$$y = A.\cos(2 * pi * 10 * t) \qquad (۱-۱۰)$$

$$t = 0{:}\,0.1{:}\,100$$

با استفاده از دستور sound، به ازای دامنه‌های ۰,۱، ۱ و ۱۰ به سیگنال گوش دهید. چه تفاوتی حس می‌کنید؟ نقش دامنه‌ی سینوسی در سیگنال صوت چیست؟

تمرین۲: مشابه تمرین قبل، این بار به ازای سه فرکانس دلخواه سیگنال را پخش کنید. چه تفاوتی ایجاد خواهد شد؟

تمرین۳: فرکانس نمونه برداری را به عدد 20kHz تغییر دهید و دوباره به سیگنال گوش دهید. چه تفاوتی وجود دارد؟ چرا؟

## آزمایش ۱۰-۲: پردازش سیگنال‌های صوتی

با استفاده از دستور audioread میتوان فایل صوتی نمونه برداری شده را داخل یک بردار ذخیره کرده و سپس پردازش‌های لازم را برروی آن انجام داد. همچنین دستور audiowrite به منظور ذخیره‌ی فایل صوتی در هارددیسک تعبیه شده است.

تمرین ۴: فایل audio1.wav را بخوانید و در بردار a ذخیره کنید. فرکانس نمونه برداری آن را مشخص کنید. با استفاده از دستور audioplay به سیگنال گوش دهید. طول سیگنال چند ثانیه است؟

تمرین۵: با استفاده از دستور audioinfo تعداد کانال‌ها، فرکانس نمونه برداری و طول سیگنال صوتی را مشخص کنید.

تمرین ۶: فایل audio2.wav را خوانده و در بردار b ذخیره کنید. ابتدا فایل را گوش داده سپس هر دو بردار a و b را با استفاده از دستور subplot رسم کنید. چه تفاوتی از روی نمونه های زمانی آن‌ها مشخص است؟

تمرین۷: دو سیگنال a و b را با یکدیگر جمع کنید و سپس سیگنال ایجاد شده را ابتدا گوش داده و سپس رسم کنید. آیا از روی شکل موج سیگنال، می‌توانیم دو صدا را تفکیک کنیم؟

تمرین ۸: سیگنال a را با یک موج سینوسی با دامنه‌ی ۱ و فرکانس 1Khz جمع کنید. با این کار به سیگنال نویز اضافه کرده ایم. حال دوباره به سیگنال گوش داده و آن را رسم کنید. آیا از شکل موج، نویز و سیگنال صحبت قابل تفکیک اند؟ سیگنال آغشته به نویز را با نام audio3.wav ذخیره کنید.

تمرین۹: اندازه‌ی تبدیل فوریه‌ی سیگنال آغشته به نویز تمرین قبل را رسم کنید. آیا با مشاهده‌ی مولفه‌های فرکانسی، نویز از سیگنال قابل تفکیک است؟ چه راه حلی برای حذف نویز پیشنهاد می‌دهید؟

# آزمایش ۱۱

## فیلترینگ و بهبود کیفیت صوت

یکی از کاربردهای مهم پردازش سیگنال دیجیتال، حذف نویز از صوت می‌باشد. در آزمایش قبل با پخش سیگنال‌های صوتی، رسم طیف و رسم شکل موج آن‌ها آشنا شدید. همچنین مشاهده کردید که سیگنال آغشته به نویز در حوزه‌ی فرکانس قابل تشخیص و تفکیک است. در این آزمایش، با توجه به مطالب آموخته شده در طراحی فیلترهای دیجیتال در آزمایش‌های پیشین، سعی در طراحی فیلترهای مناسب برای حذف نویز و بهبود کیفیت سیگنال صوتی خواهیم داشت.

General purpose commands: فرمان‌های عمومی :

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| size | length | | | | | | |

Operators and special characters: عملگرها و کاراکترهای خاص :

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| : | . | + | - | * | / | ; | % |
| == | ~ | & | \| | .* | | | |

Language constructs and debugging: ساختارهای زبانی و اشکال زدایی :

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| else | function | if | end | for | | | |

Data analysis and fourier transform functions: آنالیز داده و توابع تبدیل فوریه:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| fft | ifft | max | min | | | | |

Two dimensional graphics: ابزار ترسیم دو بعدی:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| axis | stem | plot | Title | xlabel | ylabel | | |

image processing toolbox: جعبه ابزار پردازش صوت :

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| audioread | audioplay | audiowrite | | | | | |

## آزمایش ۱۱-۱: طراحی فیلتر حذف نویز

تمرین ۱: سیگنال audio1.wav را خوانده و آن را گوش دهید. شکل موج و طیف سیگنال را رسم کرده و مولفه‌های نویز را مشخص کنید. چه نوع فیلتری برای حذف این نویز لازم است؟

تمرین۲: پارامترهای فیلتر در چه محدوده‌ای باید باشند؟

تمرین۳: بک فیلتر باتروورث طراحی کنید که نویز را از این سیگنال صوتی حذف کند.با استفاده از دستور butter ضرایب صورت و مخرج را مشخص کنید. سپس با استفاده از دستور filter، سیگنال جدید را به دست آورید.

تمرین۴: سیگنال جدید را با همان فرکانس نمونه برداری قبلی گوش دهید؟ آیا اثر حذف نویز مشهود است؟ طیف سیگنال جدید را رسم کنید. آیا به خوبی مولفه‌های نویز حذف شده اند؟

تمرین۵: همین مراحل را این بار با فیلتر FIR انجام داده و نتیجه را با حالت قبل مقایسه کنید.

## آزمایش ۱۱-۲: سیگنال‌های صوتی دارای دو کانال

در آزمایش‌های قبلی دیدیم که گاها سیگنال‌های صوتی دارای دو کانال می‌باشند. لذا بردار آن‌ها دارای دو ستون است. در این آزمایش، کار با این سیگنال‌ها را تمرین خواهید کرد.

تمرین۶: تابعی بنویسید که سیگنال صوت ورودی را خوانده، دو کانال آن را از هم تفکیک کند و هر کدام را جداگانه با اسم دلخواه ذخیره کند. سپس شکل موج سیگنال اصلی و دو کانال جدا شده را رسم کند.

تمرین ۷: سیگنال audio2.wav را فراخوانی کنید و شکل موج آن را رسم کنید. این سیگنال دارای چند کانال است؟

تمرین۸ : با استفاده از تمرین قبل، برنامه ای بنویسید که دو کانال را از هم تفکیک کرده، و اثر نویز را در هر کدام حذف و سپس سیگنال نهایی را دوکاناله به عنوان خروجی دخیره کند. ( توجه کنید که ابتدا با رسم طیف هر کدام از کانال‌ها، باید فیلتر مناسب را طراحی کنید).

تمرین۹: سیگنال جدید را گوش دهید. آیا اثر نویز حذف شده است؟ شکل موج و طیف هر دو سیگنال را رسم کنید و بیان کنید که آیا نویز به طور کامل حذف شده است؟

# آزمایش ۱۲

## پیاده‌سازی مدولاسیون‌های دیجیتال

امروزه بسیاری از سیستم‌ای فرستنده و گیرنده از سیگنال‌های دیجیتال استفاده می‌کنند. ذخیره‌سازی سیگنال‌ها در رایانه‌ها به صورت دیجیتال صورت میگیرد و پردازش بر روی آن انجام می‌شود. به منظور ارسال دیتای دیجیتال در کانال مخابراتی، از مدولاسیون‌های دیجیتال استفاده می‌کنیم. مدولاسیون دیجیتال بدین معنی است که ویژگی‌های سیگنال حامل با توجه به بیت‌های ورودی تغییر می‌کند. این تغییرات می‌تواند در دامنه، فاز و یا در فرکانس سیگنال حامل اعمال شود. در این آزمایش، با طریقه‌ی ساخت تعدادی از این مدولاسیون‌ها و خواص آن‌ها آشنا خواهیم شد.

فرمان‌های عمومی : General purpose commands:

| | | |
|---|---|---|
| size | length | input |

عملگرها و کاراکترهای خاص : Operators and special characters:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| : | . | + | - | * | / | ; | % |
| == | ~ | & | \| | .* | | | |

ساختارهای زبانی و اشکال زدایی : Language constructs and debugging:

| | | | | |
|---|---|---|---|---|
| else | function | if | end | for |

آنالیز داده و توابع تبدیل فوریه: Data analysis and fourier transform functions:

| | | |
|---|---|---|
| fft | ifft | rand |

ابزار ترسیم دو بعدی: Two dimensional graphics:

| | | | | |
|---|---|---|---|---|
| axis<br>ylabel | stem | plot | Title | xlabel |

## آزمایش ۱۲-۱: مدولاسیون ASK

در این مدولاسیون، تغییرات داده‌ی ورودی بر روی دامنه‌ی سیگنال حامل اعمال می‌شود. بدین‌گونه که به ازای هرسطح ازورودی، دامنه‌ای متفاوت از سیگنال حامل اعمال می‌شود. در این آزمایش، با این نوع از مدولاسیون آشنا شده و آن را بر روی داده‌ی ورودی دلخواه، اعمال خواهید کرد.

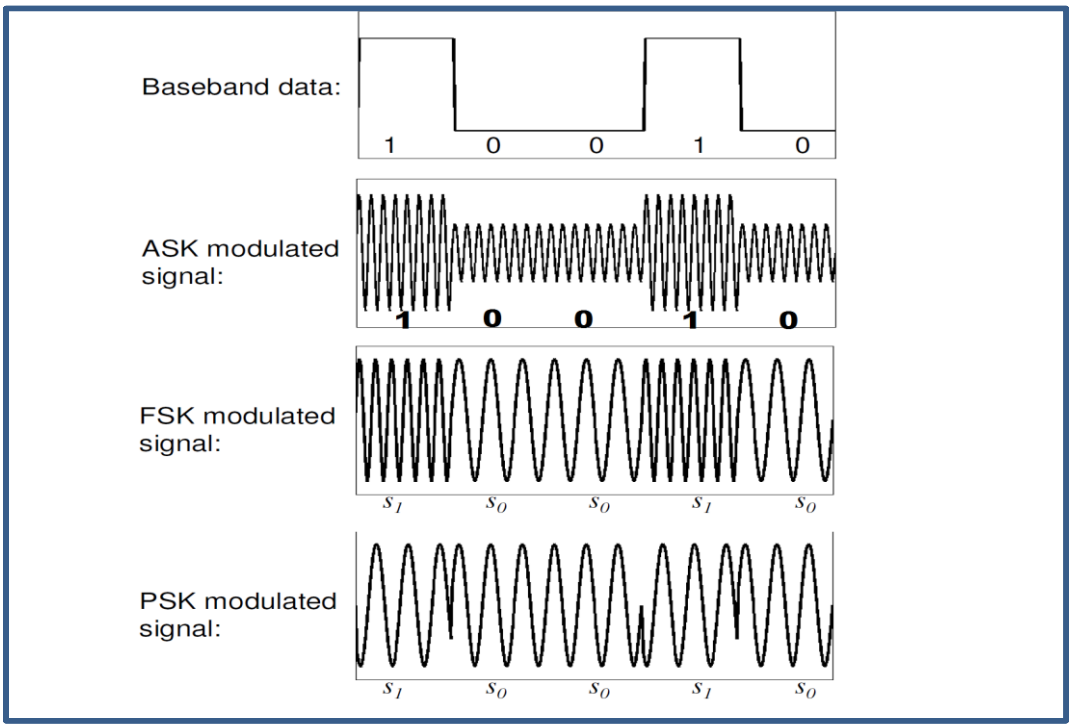
تمرین ۱: برنامه‌ی P12_1 را اجرا کنید. فرکانس حامل را ابتدا ۲۰ و فرکانس پالس ورودی را ۴ قرار دهید. با توجه به شکل رسم شده، آیا می‌توانید نحوه‌ی کار این مدولاسیون را توضیح دهید؟

تمرین۲: برنامه را به ازای فرکانس های مختلفی از سیگنال حامل اجرا کنید.

تمرین ۳: دستور input چه کاری را انجام می‌دهد؟ برنامه را به گونه‌ای تغییر دهید که دامنه‌ی سیگنال حامل نیز توسط کاربر به الگوریتم داده شود.

## آزمایش ۱۲-۲: مدولاسیون BPSK

در این نوع مدولاسیون که با نام 2PSK نیز شناخته می‌شود، با تغییر داده‌ی ورودی، فاز سیگنال حامل ارسالی عوض می‌شود. حالت BPSK حالت ساده‌از مدولاسیون PSK است که با تغییر داده، فاز حامل به اندازه‌ی ۱۸۰ درجه جابجا می‌شود. در شکل زیر، تفاوت مدولاسیون‌های ASK، FSK و PSK را مشاهده می‌کنید.




شکل ۱ نمایش مدولاسیون‌های ASK، FSK و PSK

تمرین۴: برنامه‌ی P12_2 را اجرا کنید و با توجه به شکل سیگنال مدوله شده و سیگنال ورودی، مدولاسیون را توضیح دهید.

تمرین۵: دستور setچه کاری انجام می‌دهد؟

تمرین۶: با استفاده از این برنامه و برنامه‌ی قبل، برنامه‌ای بنویسید که نوع مدولاسیون (ASK و PSK)، فرکانس حامل و فرکانس دیتا را از کاربر گرفته و مدولاسیون را انجام دهد.

## آزمایش ۱۲-۳: مدولاسیون FSK

در این نوع از مدولاسیون، تغییرات بیت‌های ورودی در فرکانس سیگنال حامل مشخص می‌شود. به گونه‌ای که به ازای بیت-های مختاف ورودی، فرکانس‌های مختلفی از سیگنال حامل ارسال می‌شود. در حالت ساده ی آن، مدولاسیون BFSK را خواهیم داشت که در ازای دو بیت مختلف ورودی (۰ یا ۱)، سیگنال حامل دو فرکانس مجزا خواهد داشت.

تمرین ۷: برنامه‌ی P12_3 را اجرا کنید. این برنامه رشته بیت ورودی را از کاربر می‌خواهد. با توجه به متن برنامه، چهار شکلی که رسم می‌شود را توضیح دهید که هر کدام مربوط به چه سیگنالی است؟

تمرین۸: با توجه به برنامه‌ی فوق، ویژگی این مدولاسیون را توصیف کنید.

تمرین ۹: حلقه‌ی for موجود در خط ششم برنامه، چه کاری انجام می‌دهد؟

تمرین۱۰: بخش آخر برنامه که نویز را به سیگنال اضافه می‌کند، چه کاری انجام می‌دهد ؟ با توجه به این بخش، مزیت مدولاسیون FSK را نسبت به دو مدولاسیون قبل بیان کنید.

تمرین ۱۱: تابعی بنویسید که مقادیر ورودی آن شامل رشته‌ی بیت ورودی، دامنه‌ی سیگنال حامل و فرکانس‌های سیگنال حامل باشد و در خروجی سیگنال مدوله شده را تحویل دهد. شکل تابع به صورت زیر باشد.

```
function [ s ] = my_FSK( x,A,f1,f2 )
```

تمرین ۱۲: با استفاده از تابع فوق، به ازای داده‌های مختلف و فرکانس‌های مختلف، مدولاسیون FSK را انجام داده و رسم کنید.