Digitalna obdelava signalov
Laboratorijske vaje

(c) Urban Burnik

Kazalo

Vaja 1
Zapis digitalnih signalov

MATLAB je odlično orodje za študij digitalne obdelave signalov, saj v svojem jeziku vključuje mnogo funkcij, ki jih potrebujemo pri tvorbi in obdelavi signalov. Z uporabo grafičnih funkcij v okolju MATLAB si bomo zlahka ogledali rezultate obdelave in s tem poglobili razumevanje prijemov, ki smo jih spoznali na predavanjih in med študijem literature.

Osnovni signali, ki jih srečujemo na področju digitalne obdelave, so enotin impulz $\delta[n]$, sinusne in eksponentne oblike signalov in njihova posplošitev na kompleksni eksponentni zapis. MATLAB omogoča zapis podatkov v obliki matrik; naši signali bodo tako stolpni vektorji končne dolžine. Bodite pozorni, da se indeksi posameznih elementov vektorja v MATLAB-u nahajajo v območju od 1 do N; literatura namreč uporablja tudi negativne in ničelni indeks, MATLAB pa jih ne dovoljuje. Upoštevajte tudi, da v okolju MATLAB `for` zank skoraj nikoli ne potrebujete.

1.1 Priprava in prikaz realnih sinusnih nihanj

Eden od osnovnih signalov je tudi sinusni signal. Splošni realni signal sinusnega poteka opisujejo parametri amplitude (A), frekvence(ω_0) in faze(Φ) v izrazu 1.1.

$$x[n] = Acos(\omega_0 n + \Phi) \tag{1.1}$$

Osnovne funkcije in operatorji, ki jih boste v okolju MATLAB potrebovali za prikaz sinusnega signala, so `sin, cos, plot` in tudi `:`. Razlago posameznih ukazov najdete v sistemu za pomoč v MATLABu. Na najbolj preprost način naloge v MATLABu lahko opravimo tako, da ukaze enega za drugim tipkamo v ukazno vrstico. Delovne spremenljivke v okolju se med izvedbo posameznih ukazov ohranijo in jih lahko uporabimo v naslednjih sklicih. Deklaracije spremenljivk seveda niso potrebne.

Matlab in tvorba signalov S pomočjo programskega paketa MATLAB narišimo signal

$$x_1[n] = sin(\frac{2\pi}{100}n), 0 \leq n \leq 100. \tag{1.2}$$

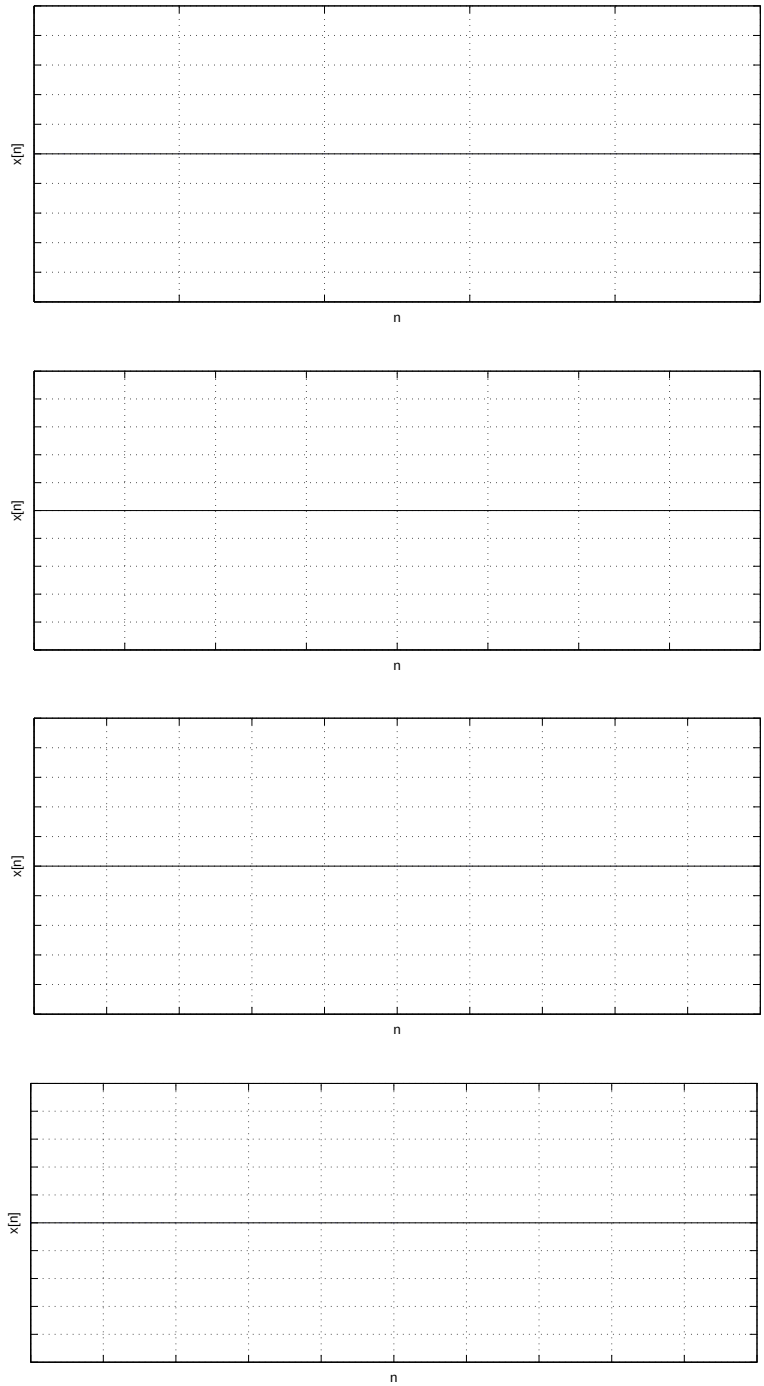
Rešitev:

```
n = 0:100;            %vektor n napolnimo z zaporednimi števili od 0 do 100
x = sin(2*n*pi/100);  %za vsak element iz vektorja n izračunamo ustrezno vrednost
plot(n, x);           %narišemo potek diskretne funkcije
```

Naloga 1 Oblikujte in narišite vsakega od naslednjih nizov. Uporabite zmožnost MATLAB-a za vektorsko računanje sinusne funkcije. V vsakem primeru naj vrednost indeksa obsega navedeno območje; oznake na grafu naj ustrezajo poteku indeksa. Za prikaz uporabite funkcijo `plot`.

$$\begin{aligned} x_1[n] &= cos(\frac{\pi}{100}n) \quad & 0 \leq n \leq 200 \\ x_2[n] &= sin(\frac{\pi}{100}n) \quad & 0 \leq n \leq 200 \\ x_3[n] &= cos(\frac{\pi}{100}n) \quad & -50 \leq n \leq 150 \\ x_4[n] &= cos(\frac{\pi}{50}n) \quad & 0 \leq n \leq 100 \end{aligned}$$

Vrišite rezultate v slike.



Pojasnite razliko med x_2 in x_3!

Primerjajte in obrazložite razlike med signaloma x_1 in x_4!

Kaj povzroči podpičje na koncu vsake vrstice?

Funkcije in skripti Kadar (in to je običajno) v okolju MATLAB določen izračun zaporedoma opravljamo nad različnimi argumenti, v ta namen lahko uporabimo ti. zaporedje ukazov (`script`) ali pa funkcijo (`function`). Tako *zaporedje ukazov* kot *funkcija* se nahajata v običajni besedilni datoteki s končnico `*.m`. Zaporedje ukazov deluje neposredno (kot npr. makro) v delovnem prostoru MATLABa in ne pozna vhodnih ali izhodnih spremenljivk. Funkcija ima lahko poljubno število argumentov, morebitne pomožne spremenljivke pa ostanejo interne.

Kot primer zaporedja ukazov vzemimo datoteko `vaja1a.m` z vsebino:

```
n = 0:100;
x = sin(n*pi/100);
plot(n, x);
```

Primer funkcije predstavlja datoteka `vaja1b.m` z vsebino:

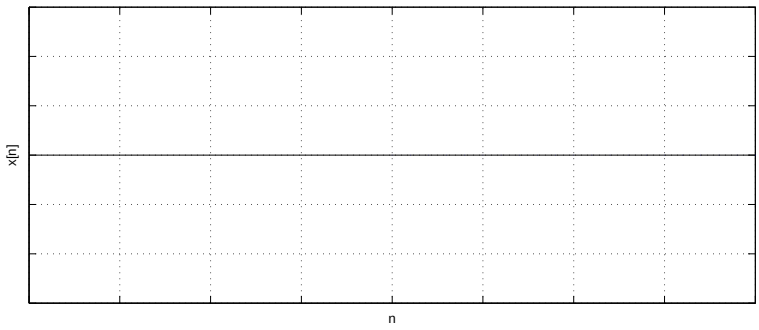
```
function [rez1, rez2]=vaja1b(vhod1, vhod2)
rez1=vhod1+vhod2;
rez2=vhod1*vhod2;
```

Iz delovnega okolja funkcije kličemo kot:

```
vaja1a
[x,y]=vaja1b(1,2)
```

Naloga 2 Napišite MATLAB-ovo funkcijo, ki oblikuje sinusoido končne dolžine. Funkcija naj ima 5 argumentov: tri za parametre sinusoide in dva za določitev začetnega in končnega indeksa n. Funkcija naj vrne stolpni vektor, ki vsebuje vrednosti sinusnega signala. Obenem naj nariše rezultat.

Preizkusite funkcijo z različnimi vhodnimi parametri. Pokažite njeno delovanje na primeru $x[n] = 2sin(\frac{\pi n}{110})$ na območju $-200 \leq n \leq 200$. Narišite rezultat! Dopolnite funkcijo iz točke 2 tako, da slednja vrne 2 argumenta: vektor indeksov in signal. Dopolnjeno funkcijo (besedilo) vpišite v prostor pod sliko signala!



1.2 Kompleksni signali

Oglejmo si še predstavitev signalov v kompleksni obliki. V resničnem svetu imajo vsi signali realne vrednosti, vendar je pogosto koristno, če oblikujemo, obdelujemo in prikazujemo pare realnih signalov v kompleksni obliki. To storimo tako, da vrednosti dveh realnih signalov združimo v par; v realno in imaginarno komponento kompleksnega števila. Za obdelavo in združevanje takšnega signala z drugimi kompleksnimi signali uporabimo pravila kompleksne aritmetike. Takšen pristop je široko v uporabi.

Kompleksni eksponentni zapis je v razredu kompleksnih signalov še posebej pomemben, ker omogoča zelo jasen in nazoren zapis sinusnih signalov. Zapis signalov v obliki *fazorjev* je razširjen na celotnem področju elektrotehnike.

Za oblikovanje in prikaz signalov boste uporabili naslednje (in njim podobne) funkcije:
`real, imag` za izračun realne in imaginarne komponente kompleksnega števila
`exp` za izračun eksponentnih funkcij
`i, j` predstavljata imaginarno število i.
`subplot` za izpis večjega števila grafov na eno samo sliko

Naloga 3 Eulerjeva enačba trdi:

$$x[n] = (z_0)^n = r^n e^{j\phi n} = r^n(cos(\phi n) + jsin(\phi n)) \tag{1.3}$$

pri čemer je $z_0 = re^{j\phi} = r\angle\phi$. Za splošen zapis poljubnega sinusnega signala potrebujemo še kompleksno konstanto, fazor $G = Ae^{j\Phi} = A\angle\Phi$:

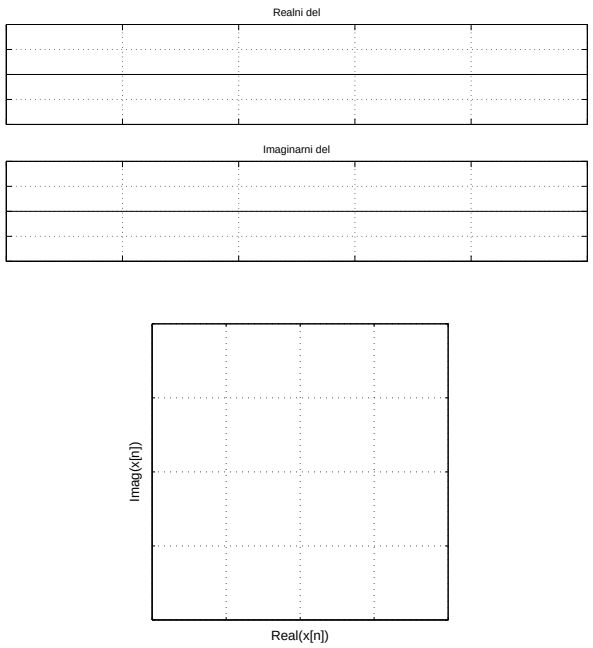
$$x = Gz_0^n = Ae^{j\Phi}r^n e^{j\phi n} = Ar^n e^{j(\phi n+\Phi)} = Ar^n[cos(\phi n + \Phi) + jsin(\phi n + \Phi)]. \tag{1.4}$$

Pretvorite signal, katerega realni del je bil enak

$$Re(x[n]) = cos(\frac{\pi}{50}n), 0 \leq n \leq 400$$

v zapis s fazorji (G, z_0)! Na sliko narišite graf istega signala tako, da oordinata grafa predstavlja realni, abscisa pa imaginarni del spremenljivke x. Na spodnjo sliko narišite realni in imaginarni del signala $x[n]$. Pomagajte si s priloženim vzorcem kode. Kaj bi morali storiti, da bi realni del spremenljivke x predstavljal sinusno funkcijo z amplitudo 2? Kaj se zgodi, če postavimo vrednost $r = 0.99$?

```
...
x=G*z_0.^n; %kompleksna eksponencialka


subplot(311)
plot(n, real(x))
title('Realni del'),xlabel('Indeks (n)')
subplot(312)
plot(n,imag(x))
title('Imaginarni del'), xlabel('Indeks (n)')
subplot(313)
plot(x)
axis('square');
```



Opozorilo: celotna vaja obravnava signale kot zvezne, čeprav je Matlab numerično orodje.

Vaja 2
Vzorčenje zveznih signalov

V nalogi se bomo seznanili z osnovnimi problemi vzorčenja časovno zveznih signalov. Ogledali si bomo problem prekrivanja frekvenc na primeru sinusnih signalov in linearno frekvenčno moduliranih signalov (ti. "chirp").

Simulacija vzorčenja z MATLABom je nekoliko težavna, saj MATLAB ne pozna zveznih signalov. Edina oblika podatkov, ki jo MATLAB lahko prikaže, je zapis diskretnih signalov v obliki vektorjev. Zato bomo morali simulirati os časovnega poteka signala z diskretnim modelom. Zavedati se moramo razlike med Δt, ki smo ga uporabili za simulacijo zveznega signala in periodo vzorčenja T_s.

Vzorčenje signala $x(t) = sin(2\pi f_o t + \Phi)$ s frekvenco $f_s = \frac{1}{T_s}$ lahko simuliramo na naslednji način:

$$x[n] = x(t)|_{t=nT_s} = x(t)|_{t=n/f_s} = sin(2\pi \frac{f_o}{f_s} n + \Phi) \tag{2.1}$$

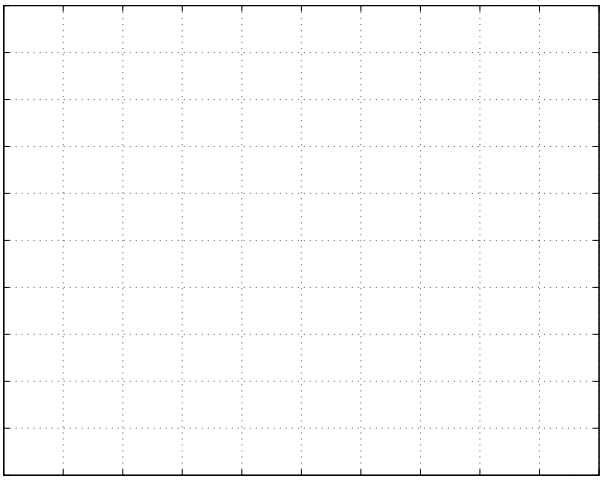
Teorem vzorčenja in njegove posledice si lahko ogledamo pri različnih kombinacijah f_0 in f_s.

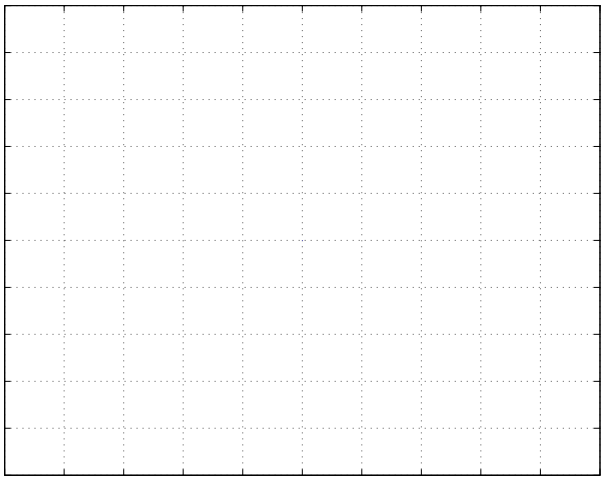
Naloga 1 Narišite sliko vzorčene sinusoide. $f_0 = 300Hz$, trajanje signala je $10ms$. Faznega zamaka ne upoštevajte ($\Phi = 0$). Frekvenca vzorčenja je $f_s = 8KHz$.

Koliko vzorcev potrebujemo?

Napišite izraz v Matlabu, ki izračuna ustrezno sinusno krivuljo!

Narišite vzorčeno sinusoido s ukazom `stem` (levo) in z ukazom `plot` (desno).

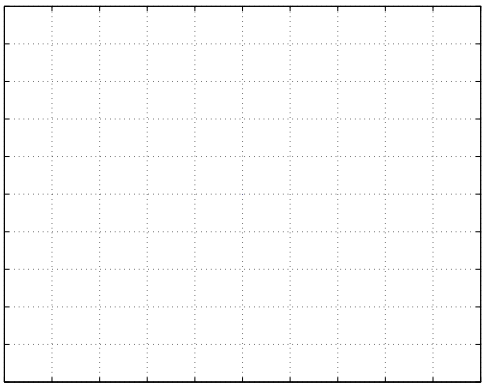


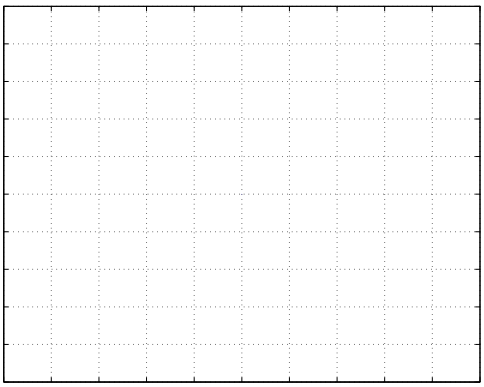


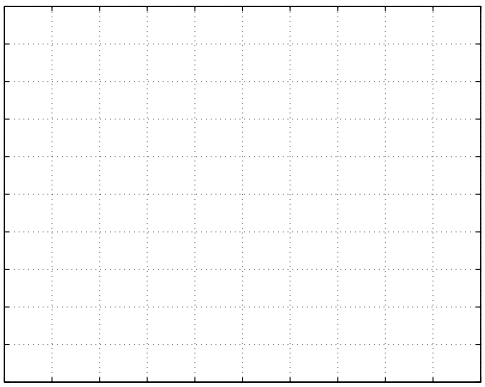
Ovojnica signala postane bolj očitna. Linearna aproksimacija nam v resnici ne daje rezultatov, ki jih predvideva teorem o vzorčenju, vendar je pogosto zadosti uporabna.

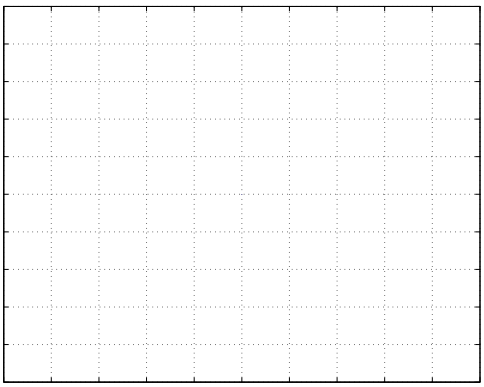
Po navedenih napotkih z uporabo funkcije `plot` prikažite tudi nekaj primerov **čistih sinusnih signalov**, npr.

- $f = 100 - 475Hz$ v korakih po $125Hz$

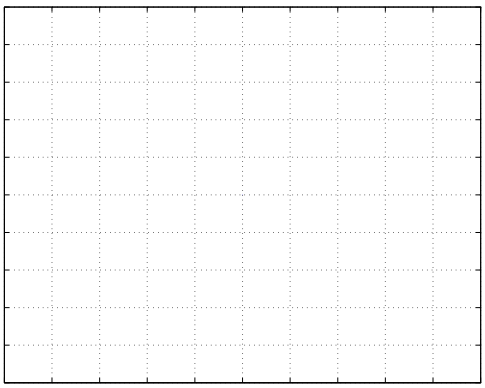


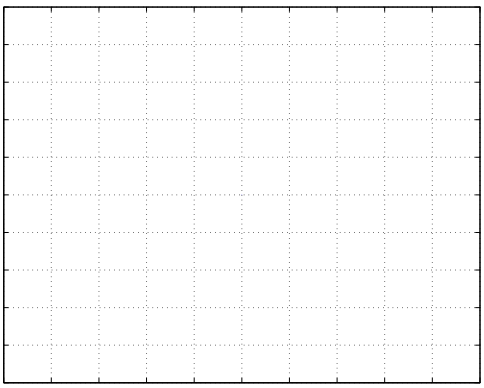


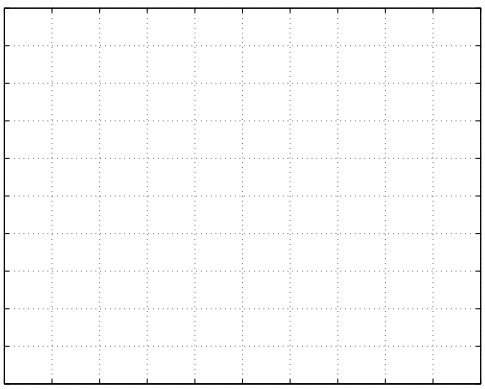


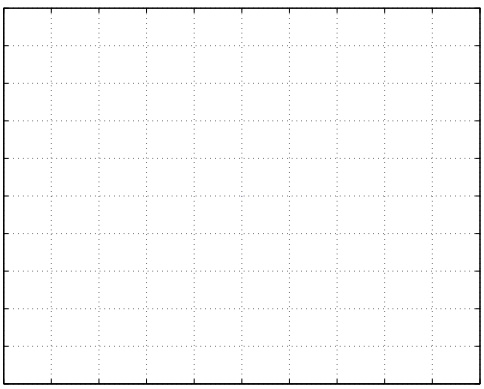


- $f = 7525 - 7900Hz$ v korakih po $125Hz$

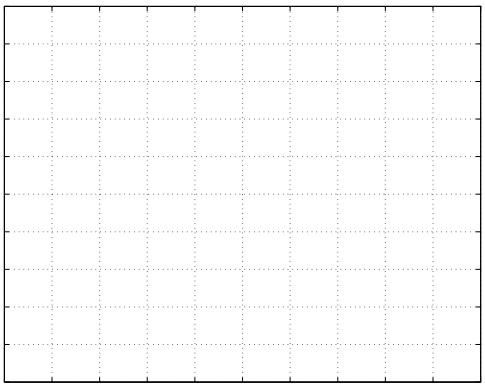
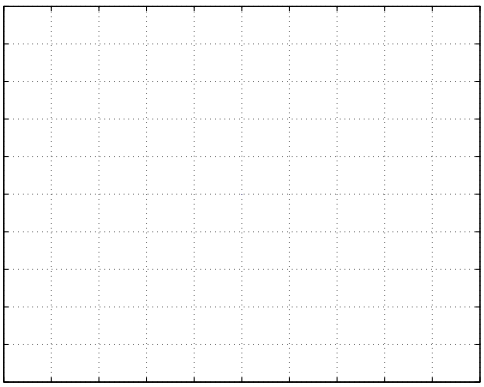
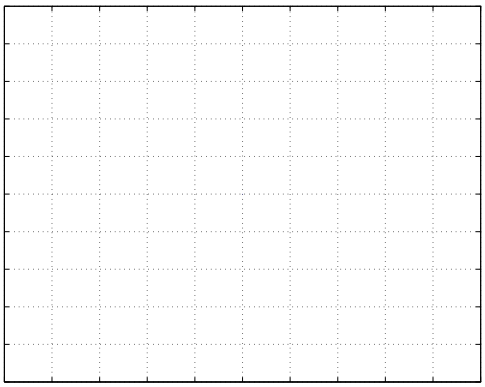
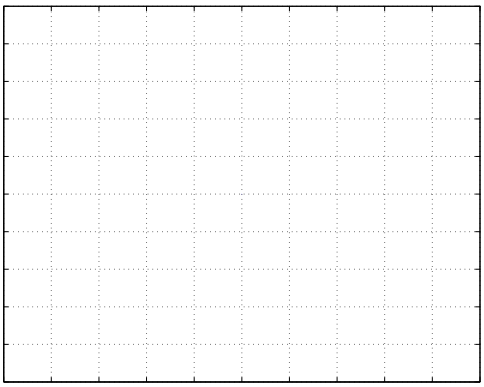








- $f = 32100 - 32475Hz$ v korakih po $125Hz$

Komentirajte!

_____

_____

Na eni sliki predstavite učinek uporabe vzorčne frekvence, ki ne ustreza Nyquistovem kriteriju. Narišite sliko vzorčene sinusoide s frekvenco $f_0 = 10kHz$, trajanje signala naj bo $2ms$. Fazni zamik naj bo enak $\Phi = 0$. Uporabite dve vzorčevalni frekvenci, $f_s1 = 40kHz$ in $f_s2 = 9500Hz$.

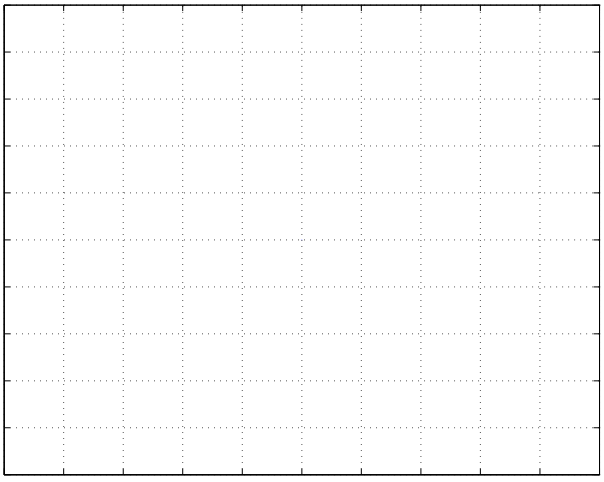
Koliko vzorcev potrebujemo glede na izbrano frekvenco vzorčenja?

`n1 = 0:`_____;

`n2 = 0:`_____;

Izračunajte vzorčena signala in narišite rezultat na eno sliko!

```
x1 = sin(n1*2*pi*f/fs1);
x2 = sin(n2*2*pi*f/fs2);
plot(n1/fs1,x1,'o',n2/fs2,x2,'-')
```



2.1 Vzorčenje linearno frekvenčno moduliranega signala

Linearno frekvenčno moduliran signal ("Chirp")je dober primer za prikaz problema prekrivanja spektra. Njegova matematična definicija je

$$c(t) = cos(\pi\mu t^2 + 2\pi f_1 t + \psi) \tag{2.2}$$

Trenutno frekvenco signala predstavlja časovni odvod argumenta kosinusne funkcije

$$f_i(t) = \mu t + f_1 \tag{2.3}$$

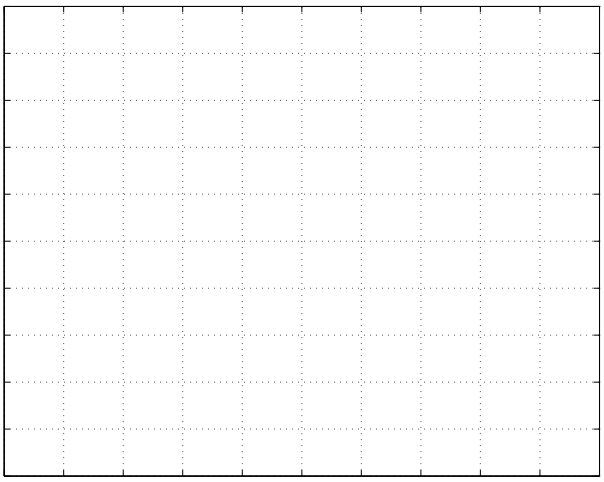
kar predstavlja linearno povečevanje frekvence glede na čas.

Naloga 2 Parametri linearno frekvenčno moduliranega signala naj bodo $f_1 = 4kHz$ in $\mu = 600kHz/s$. ψ naj bo poljuben (lahko tudi 0). Če je skupna dolžina signala enaka 50 ms, izračunajte začetno in končno frekvenco signala po enačbi 2.3.

Vzorčna frekvenca naj bo enaka $f_s = 8kHz$. Izračunajte potrebno število vzorcev!

Podajte izraz za izračun signala v Matlabu!

Narišite signal $c[n]$ v diskretnem prostoru z uporabo funkcije `plot`. Za uspešno delo boste morali zvezno definiran linearno frekvenčno moduliran signal vzorčiti po vzorcu 2.1.



Komentirajte, kje pride do prekrivanja!

Ponovite vajo tako, da dobljeni signal poslušate. Da bi slišali rezultat, bo potrebno podaljšati trajanje signala in zmanjšati parameter μ. Vzorčna frekvenca, s katero MATLAB reproducira zvok, je enaka 8192 Hz. Priredite vrednosti μ tako, da bo zvočni signal trajal vsaj 2 sekundi, pri tem pa naredil 5 prehodov prekrivanja!

Izračunajte potrebno število vzorcev!

Za poslušanje uporabite funkcijo `soundsc`!
Komentirajte vse dobljene rezultate!

Vaja 3
Diferenčne enačbe v digitalni obdelavi signalov

V digitalni obdelavi signalov imajo posebno vrednost sistemi, ki jih lahko zapišemo v obliki linearnih diferenčnih enačb s konstantnimi koeficienti. Takšne sisteme lahko uporabljamo kot model realnih sistemov, še posebej pogosto pa kot orodje za obdelavo signalov. V tem primeru jih poimenjujemo z izrazom digitalni filtri.

V današnji vaji boste spoznali značilnosti tovrstnih sistemov v časovnem in v frekvenčnem prostoru.

3.1 Diferenčna enačba in njen časovni odziv

V nalogi boste oblikovali časovni odziv filtra z **neskončnim impulznim odzivom**, ki ga predstavlja naslednja linearna diferenčna enačba s konstantnimi koeficienti:

$$\sum_{k=0}^{N_a} a_k\, y[n-k] = \sum_{l=0}^{N_b} b_l\, x[n-l] \tag{3.1}$$

V MATLAB-u diferenčne enačbe predstavimo z dvema vektorjema; prvi predstavlja koeficiente b_l, ki ustrezajo elementom x, drugi pa koeficiente povratne vezave, a_k, soležne elementom y. Običajno oblikujemo diferenčno enačbo tako, da je koeficient a_0 enak 1, tako, da ga v izpeljavi izhodnega signala

$$y[n] = -\frac{1}{a_0}\sum_{k=1}^{N_a} a_k\, y[n-k] + \frac{1}{a_0}\sum_{l=0}^{N_b} b_l\, x[n-l] \tag{3.2}$$

ni potrebno upoštevati. Še napotek za uporabo filtrov v MATLAB-u: iz zgornje enačbe je razvidno, da a_0 nikoli ne sme biti enak 0, saj sicer po preureditvi izraza izgine rezultat!

MATLAB-ova funkcija `y = filter(b,a,x)` predstavlja digitalno filtriranje signala x, kot ga definirata vektorja koeficientov a in b diferenčne enačbe 3.1. Če je x enotin impulz, bo rezultat y predstavljal impulzni odziv sita, $h[n]$. Pozor: funkcija $filter$ vrne le toliko vzorcev y, kot jih je v vhodnem vektorju x. Zato je impulzni odziv (načelno je neskončen!) "odrezan" na dolžino vhodnega vektorja x.

Naloga 1 Tvorite vektorja `b` in `a`, ki, glede na enačbo 3.1 vsebujeta ustrezne koeficiente sita, ki ga predstavlja naslednja diferenčna enačba:

$$y[n] + 0.9y[n-2] = 0.3x[n] + 0.6x[n-1] + 0.3x[n-2] \tag{3.3}$$

```
a = [_____________________________];

b = [_____________________________];
```

Dopolnite definicijo diskretnega impulza enote!

$$\delta[n] = \begin{cases} 1, & \_\_\_\_\_ \\ 0, & \_\_\_\_\_ \end{cases}$$

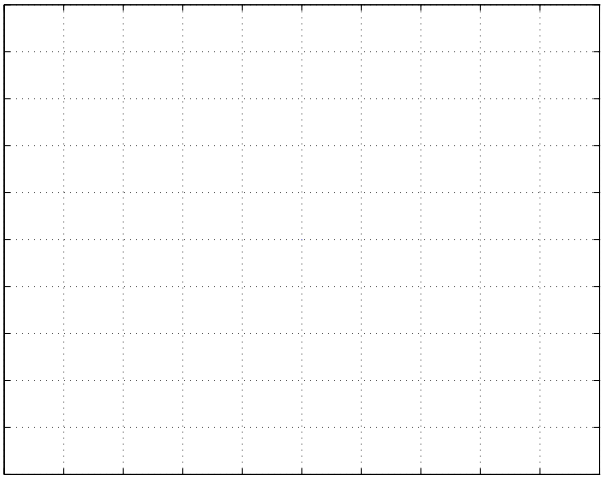
Oblikujte vektor impulza enote, `imp`, dolžine 128. Matlab pozna Boolove operatorje in jih izvaja tudi na vektorjih, zato si pri tem lahko pomagamo takole:

```
imp = (n==0);
```

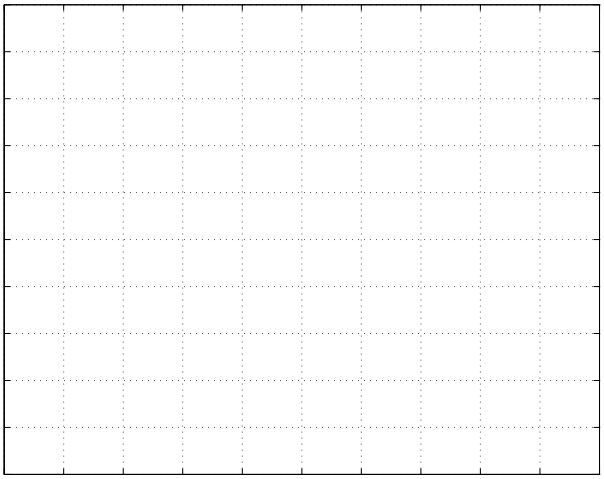
S pomočjo ukazov iz uvoda izračunajte impulzni odziv sita, ki ga podaja enačba 3.3. Upoštevajte seveda vhodni signal `imp` (napišite ukaz!).

```
y = _____________________________;
```

Z uporabo funkcije `stem` narišite impulz



in impulzni odziv sistema



v diskretnem prostoru!
Podrobnosti bodo bolje vidne, če boste narisali le prvih 20 točk signala! Verificirajte rezultat!

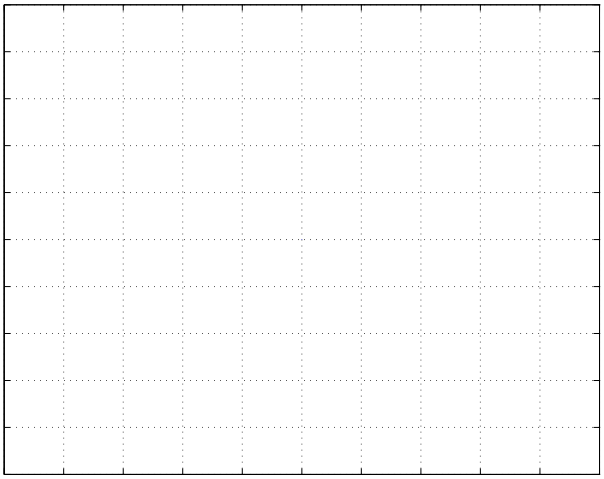
Naloga 2 Imamo naslednje diferenčne enačbe:

$$y_1[n] - 1.92\cos(\frac{\pi}{8})y_2[n-1] + 0.9216y_2[n-2] = x_2[n] + \frac{1}{2}x_2[n-1] \tag{3.4}$$

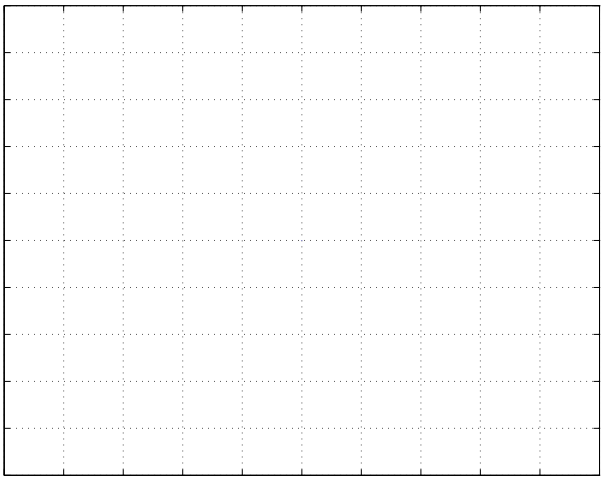
$$y_2[n] - 2\cos(\frac{\pi}{8})y_3[n-1] + y_3[n-2] = x_3[n] + \frac{1}{2}x_3[n-1] \tag{3.5}$$

Izračunajte odzive sistemov $h_1[n]$, $h_2[n]$, $h_3[n]$ in $h_4[n]$, ki jih podajajo enačbe 3.4 - 3.5, na impulz enote v območju $-10 \leq n \leq 100$!

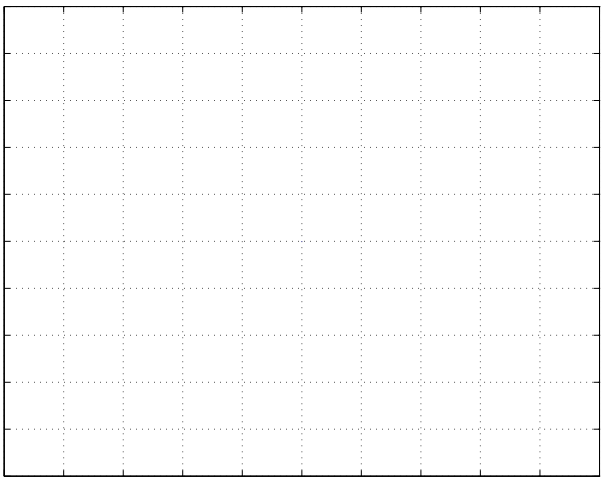
Impulz enote:



Signal $h_1[n]$:



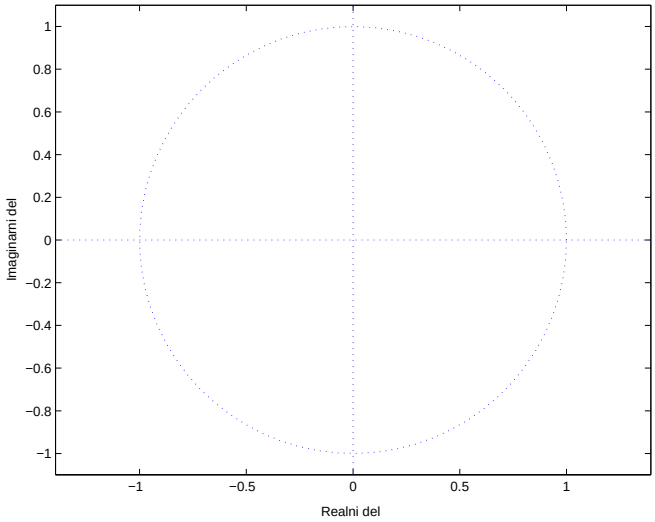
Signal $h_2[n]$:



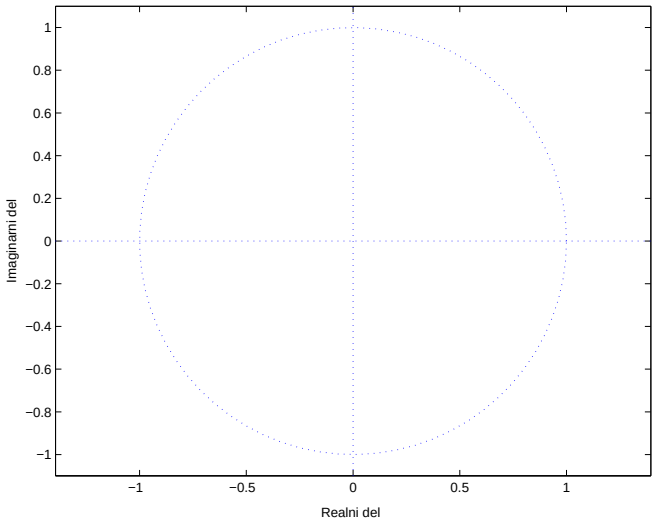
Naloga 3 Vemo, da je impulzni odziv diferenčne enačbe sestavljen iz naravnih frekvenc. Naravne frekvence določajo povratnozančni koeficienti a_k. Vsak koren (p_k^n) značilnega polinoma $A(z) = 1 + \sum_{k=1}^{N_a} a_k\, z^{-k}$ vzbuja izhodni signal oblike $p_k^n u[n]$, kjer je $u[n]$ diskretna stopnica enote. Poiščite korene (pole (poles) in ničle (zeros) značilnega polinoma diferenčnih enačb 3.4 – 3.5(pomagajte si z ukazom za iskanje korenov, `p=roots(a)` oz. `z=roots(b)`)!

Narišite lego korenov v kompleksni ravnini (`zplane(z,p)`)!

Koreni sistema 3.4 v kompleksni ravnini:



Koreni sistema 3.5 v kompleksni ravnini:



Primerjajte rezultate z odzivi sistema iz naloge 2! Kako lega korenov značilnega polinoma vpliva na odziv sistema?

Vaja 4
Diskretna Fourierova transformacija

Čeprav so lastnosti diskretne Fourierove transformacije (DFT, Discrete Fourier Transform) podobne lastnostim ostalih različic Fourierovih transformacij, lahko govorimo o pomembnih razlikah, ki jih povzroča končna narava omenjene transformacije. DFT predstavlja eno od osnovnih orodij na področju digitalne obdelave signalov. Zato nam bo koristilo, če se z njenimi lastnostmi temeljito seznanimo.

Diskretna Fourierova transformacija je preslikava vektorja z N elementi $\{x[0], x[1], ..., x[N-1]\}$, ki predstavlja signal v časovnem prostoru:

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{nk} \; za \; k = 0, 1, 2, ..., N-1 \tag{4.1}$$

pri čemer je $W_N = e^{-j2\pi/N}$.

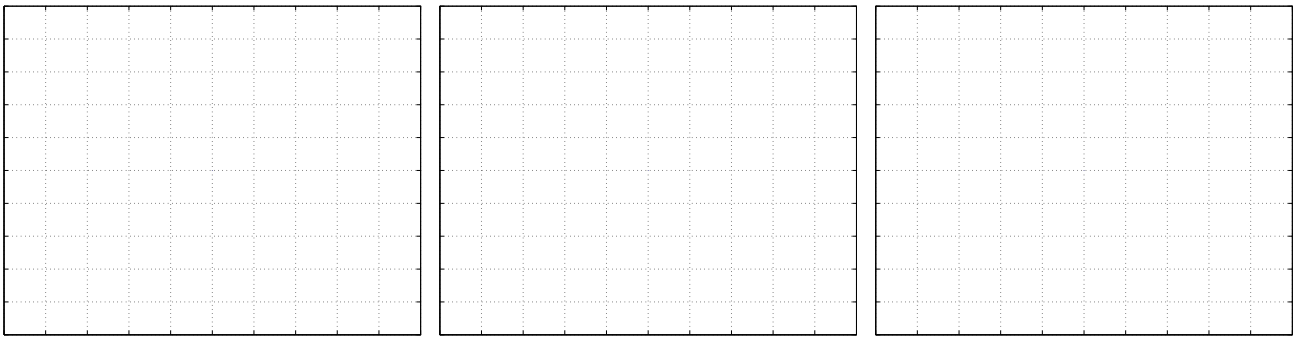
Omenimo še, kaj je razlika med kraticama DFT in FFT. FFT (Fast Fourier Transform, hitra Fourierova transformacija) je le eden od hitrih digitalnih algoritmov za izračun diskretne Fourierove transformacije. Matlab pozna zgolj ukaz `fft`; uporabljeni algoritem je uporabniku skrit in odvisen od števila vzorcev signala v časovnem prostoru. Ukaz `ifft` predstavlja inverzno transformacijo.

4.1 Izračun DFT nekaterih značilnih signalov

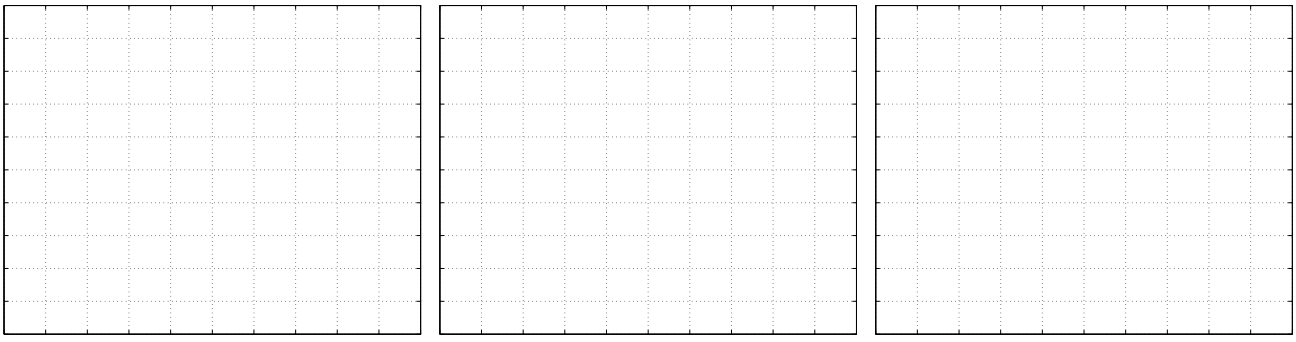
Namen naloge je, da se seznanite z obliko signalov in njihovih preslikav. Primerjajte rezultate med seboj! Za risanje uporabljajte ukaz `stem`. Rezultat DFT je v splošnem kompleksen, zato za vsak primer narišite njegovi amplitudo in fazo (`abs, angle`)!

4.1.1 Diskretna Fourierova preslikava enotinega impulza in enotine stopnice

1. Oblikujte in narišite impulz enote in njegovo DFT. Uporabite 8 točk signala!

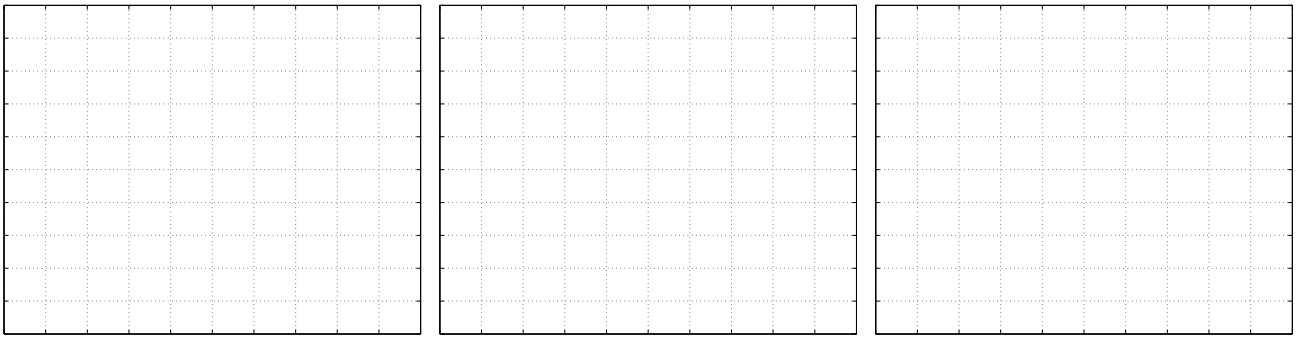


2. Oblikujte in narišite signal s samimi enicami (vse vrednosti so enake 1) in njegovo DFT. Uporabite 8 točk signala!



Primerjajte s predhodnim rezultatom! Dualnost DFT?

3. Zamik impulza: `xish=[0 0 0 1 0 0 0 0]`!

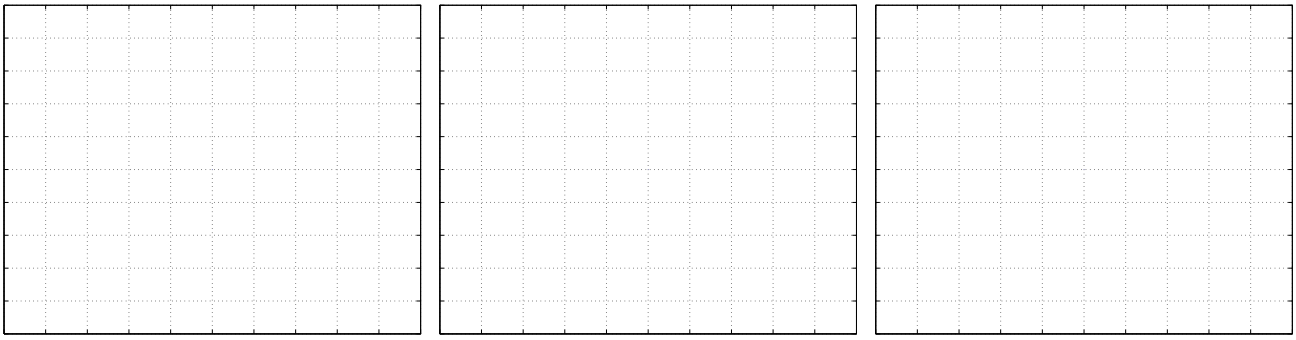


Primerjajte rezultate na prikazu faze in amplitude signala s točko 1!

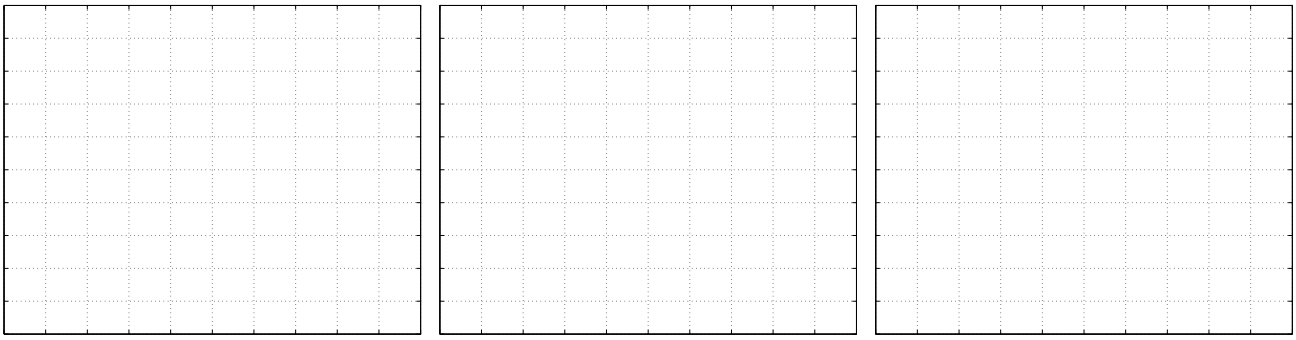
Poskusite še z drugačnimi zamiki! Ali obstoja neničelni premik enotinega impulza, pri katerem je DFT signala povsem realna? Poizkusite!

---

4. Tritočkovni pravokotni impulz: `xish=[1 1 1 0 0 0 0 0]` in njen DFT.



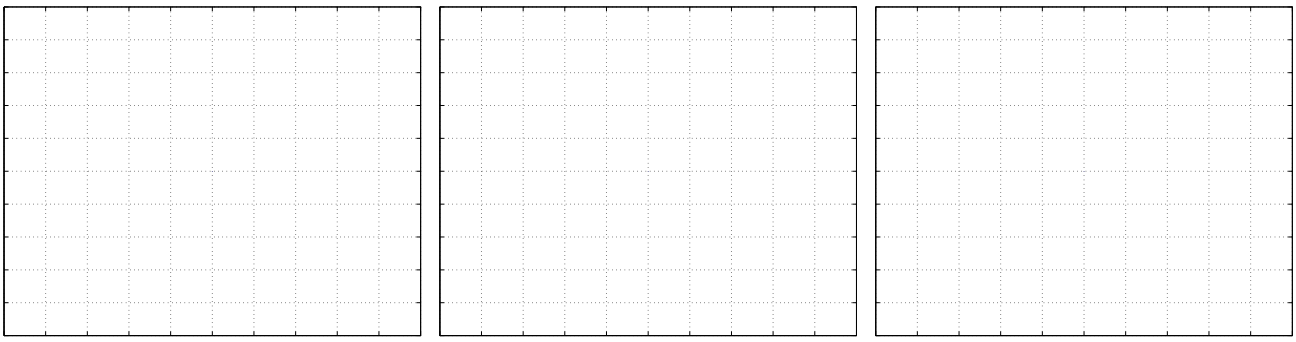
5. Simetrični impulz: `xish=[1 1 0 0 0 0 0 1]`!



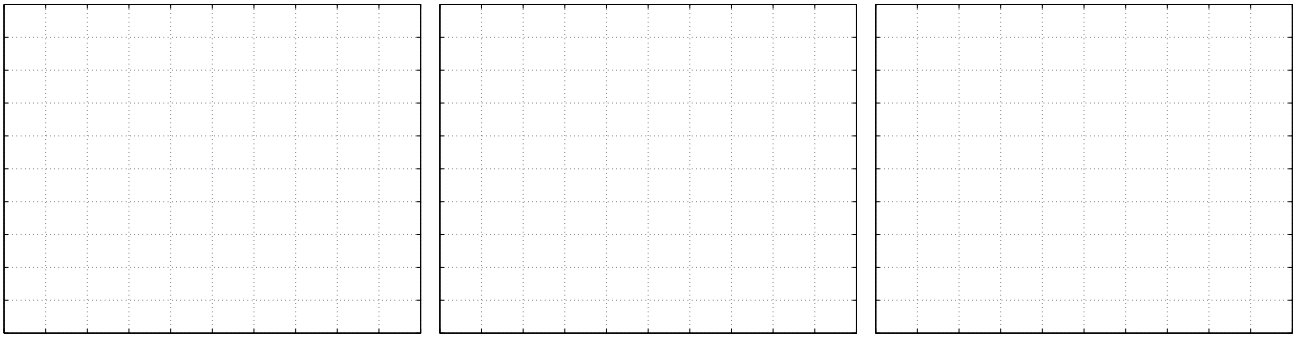
Primerjajte rezultate na prikazu faze in amplitude signala s točko 4! Kakšna je imaginarna vrednost DFT?

4.1.2 Sinusni signali

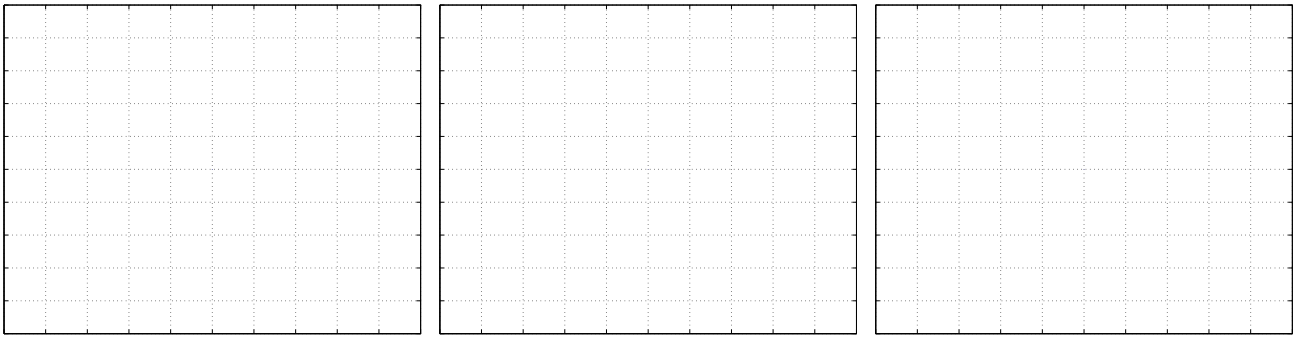
1. Izračunajte in narišite 21 točkovno DFT za signal, ki predstavlja natanko eno periodo kosinusnega signala! (Pozor: natanko 1 periodo in niti vzorca več!)



2. Ponovite točko 1 za sinusni signal

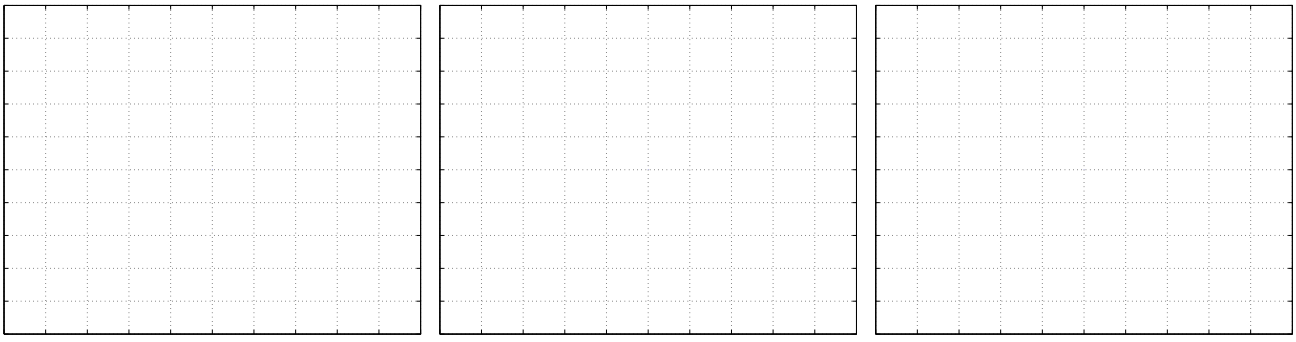


3. Uporabite sinusni signal, vendar s tremi periodami!



Iz slike razberite frekvenco signala!

4. Uporabite sinusni signal, ki obsega 3.1 periode.

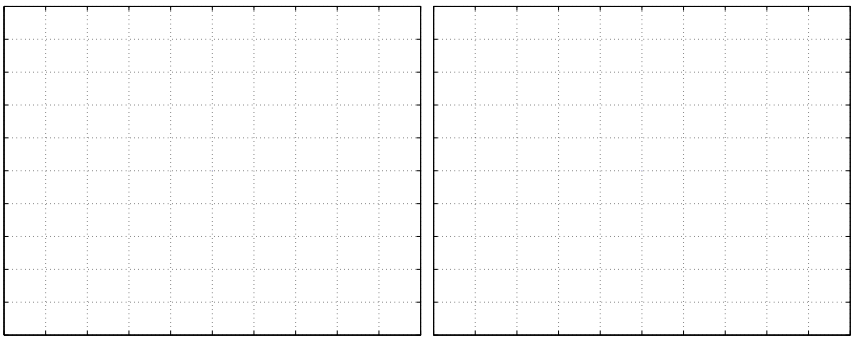


Zakaj je oblika DFT zdaj drugačna?

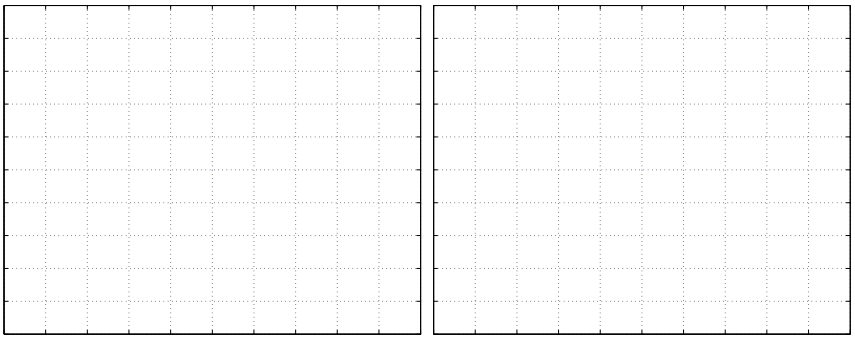
4.2 DFT, krožna in linearna konvolucija

Vemo, da sta krožna konvolucija in DFT Fourierov par. Zato krožno konvolucijo lahko izračunamo tako, da izračunamo DFT obeh signalov, rezultata zmnožimo (po komponentah!) ter izračunamo inverzni DFT dobljenega produkta.

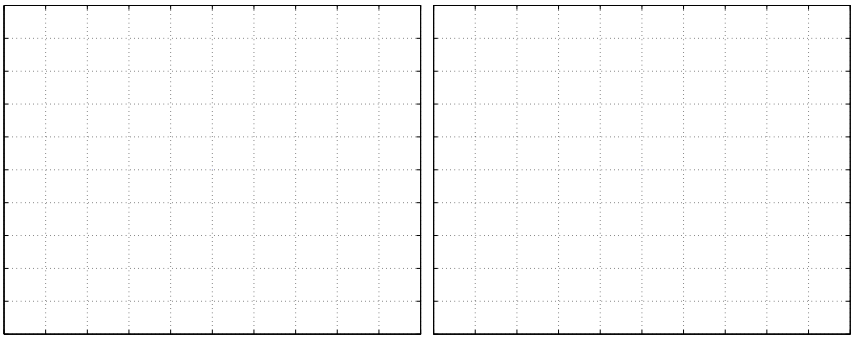
1. Vzemite signala $\mathbf{x} = [1, 1, -1, -1]$ in $\mathbf{y} = [-1, -1, 1, 1]$. Vrišite signala v okvirčka spodaj.



2. Ročno izračunajte krožno konvolucijo obeh signalov $x \circledast y$. Rezultat vrišite levo. Izračunajte DFT obeh signalov, produkt obeh transformacij in inverzno DFT. Oglejte si vse korake tudi grafično! Končni rezultat vrišite v desno okence!



3. Izračunajte linearno konvolucijo obeh signalov $x * y$. Rezultat vrišite levo. Signalna niza x in y dopolnite z ničlami tako, da boste prek DFT lahko pravilno izračunali linearno konvolucijo. Izračunajte DFT obeh dopolnjenih signalov, produkt obeh transformacij in inverzno DFT. Oglejte si vse korake tudi grafično! Rezultat vrišite v desno okence!



Vaja 5
Okenske funkcije in spektralna analiza

Na področju spektralne analize signalov kot tudi za potrebe načrtovanja digitalnih filtrov poznamo različne oblike okenskih funkcij. Oknenje je množenje signala v časovnem prostoru z okensko funkcijo; učinkuje kot uteženo rezanje signala:

$$y[n] = x[n] \cdot w[n] \;\; kjer\ je\ w[n] = 0;\ za\ n < 0, n > L - 1 \tag{5.1}$$

Oglejmo si še, kaj oknenje predstavlja v frekvenčnem prostoru signala; Fourierova preslikava v diskretnem času (DTFT) za signal, skrajšan z okensko funkcijo, je enaka periodični konvoluciji DTFT signala in DTFT okenske funkcije:

$$x[n] \overset{DTFT}{\longleftrightarrow} X(e^{j\omega}) = \sum_{n=-\infty}^{n=\infty} x[n]e^{-j\omega n} \tag{5.2}$$

$$y[n] = x[n] \cdot w[n] \overset{DTFT}{\longleftrightarrow} Y(e^{j\omega}) = \frac{1}{2\pi}\int_{-\pi}^{\pi} X(e^{j\phi})W(e^{j[\omega-\phi]})d\phi \tag{5.3}$$

5.1 Splošni napotki

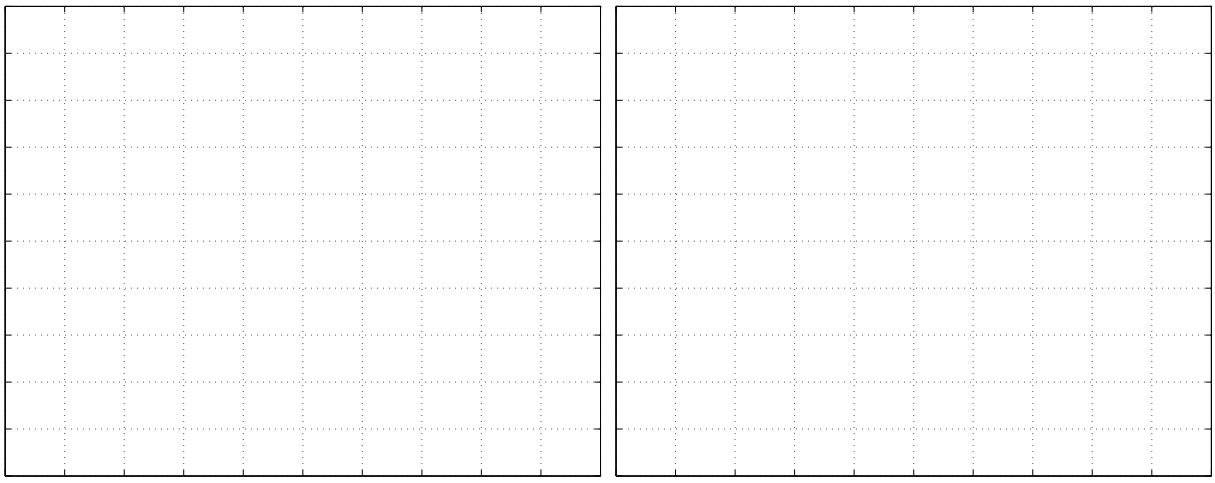
Pri izračuni DTFT okenskih funkcij si boste pomagali z vzorčenjem, saj Matlab spektra $W(e^j\omega)$ ne more izračunati v zveznem prostoru. Da bi dobili dovolj ločljiv prikaz, moramo uporabiti vsaj 4-5 krat več vzorcev, kot jih obsega okenska funkcija. Načelno to storimo tako, da okenski funkciji na koncu dodamo še ustrezno število ničel (3-4 krat toliko, kot je vzorcev $w[n]$). Matlab to stori samodejno z ukazom `W=fft(w,Nsamp)`, pri čemer `Nsamp` predstavlja ločljivost prikaza oziroma število točk prikaza spektra DTFT. Za prikaz uporabimo sorodnike funkcije `plot`; najustreznejši je **logaritemski prikaz amplitude in linearni prikaz frekvence** `semilogy`.

5.2 Značilne okenske funkcije in njihov DTFT

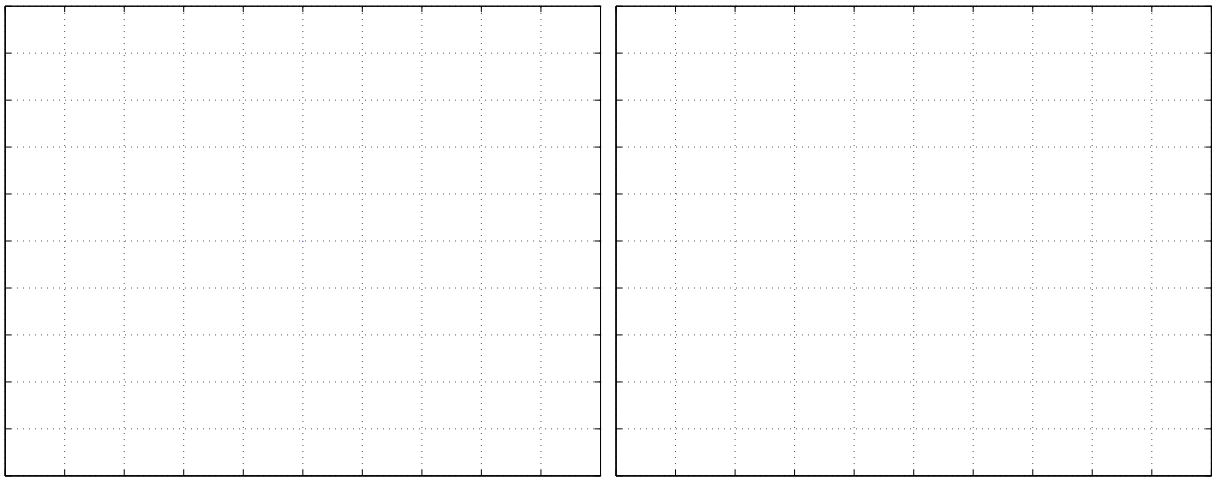
5.2.1 Pravokotno okno

Pravokotno okno je najpreprostejše okno. Kjerkoli pride do rezanja signala, govorimo o oknenju; če pri tem signala nismo "utežili" z dodatnimi funkcijami, smo pravzaprav uporabili pravokotno okno. Oglejte si DTFT pravokotnega okna in sklepajte, kakšne so posledice rezanja signala na njegov izračunani spekter.

1. Oblikujte pravokotno okno dolžine 21 in izračunajte približek njegovega spektra v logaritmičnem merilu `Nsamp = 128; boxcar; ones`. Narišite obliko okna (lahko brez uporabe Matlaba!) in njegov amplitudni spekter (`abs`)!



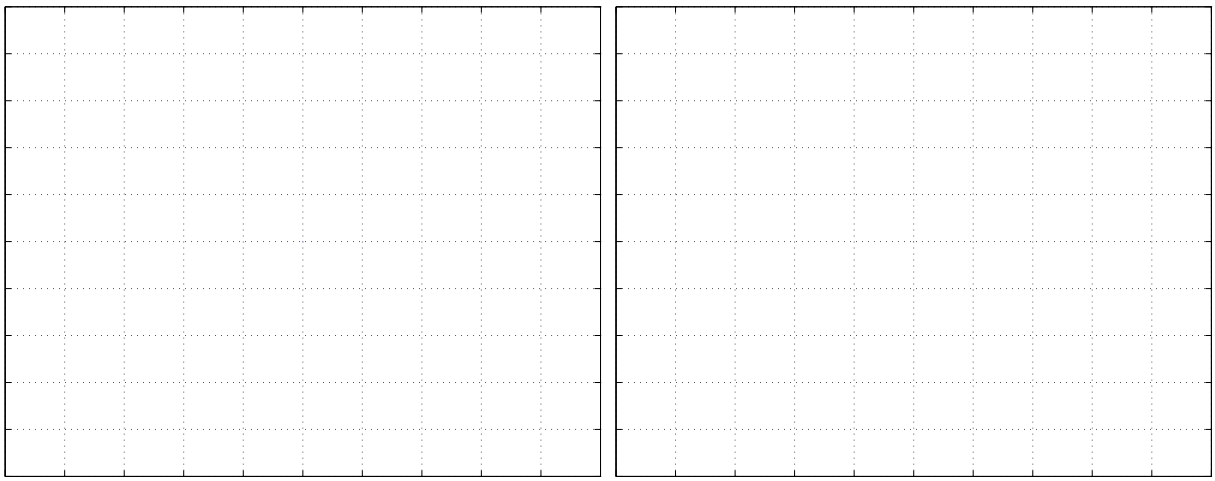
2. Ponovite vajo za različne dolžine oken: 16, 31 in 61. Narišite rezultate na en graf. Primerjajte in komentirajte (širina osnovnega snopa, velikost prvega bočnega snopa)!



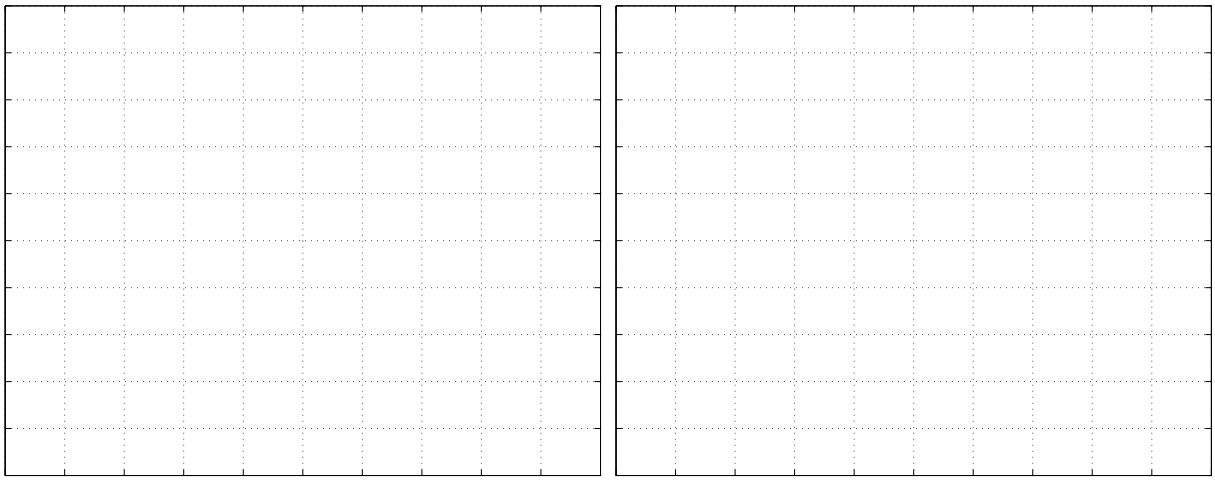
5.2.2 Trikotno okno

Okno trikotne oblike imenujemo tudi Bartlettovo okno. Matlab pozna tako funkciji `bartlett` kot tudi `triang`.

1. Oblikujte trikotno okno dolžine 11. Primerjajte funkciji `bartlett` in `triang` tako, da okni izrišete v časovnem prostoru (leva slika). Narišite DTFT v logaritmičnem merilu (desno). `Nsamp = 128, plot!`



2. Ponovite vajo za različne dolžine oken: 31 in 61. Uporabite funkcijo `triang`. Narišite rezultate na en graf, slike okenskih funkcij na levi in amplitudnega spektra na desni strani.

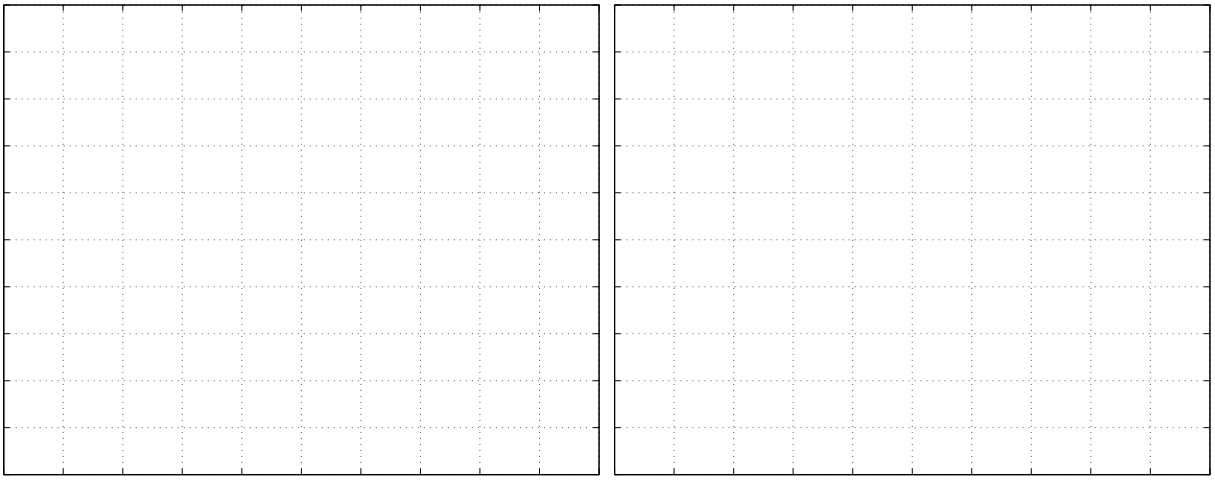


Primerjajte rezultate s pravokotnim oknom in komentirajte! V kakšnem matematičnem razmerju sta trikotno in pravokotno okno in kaj to pomeni?

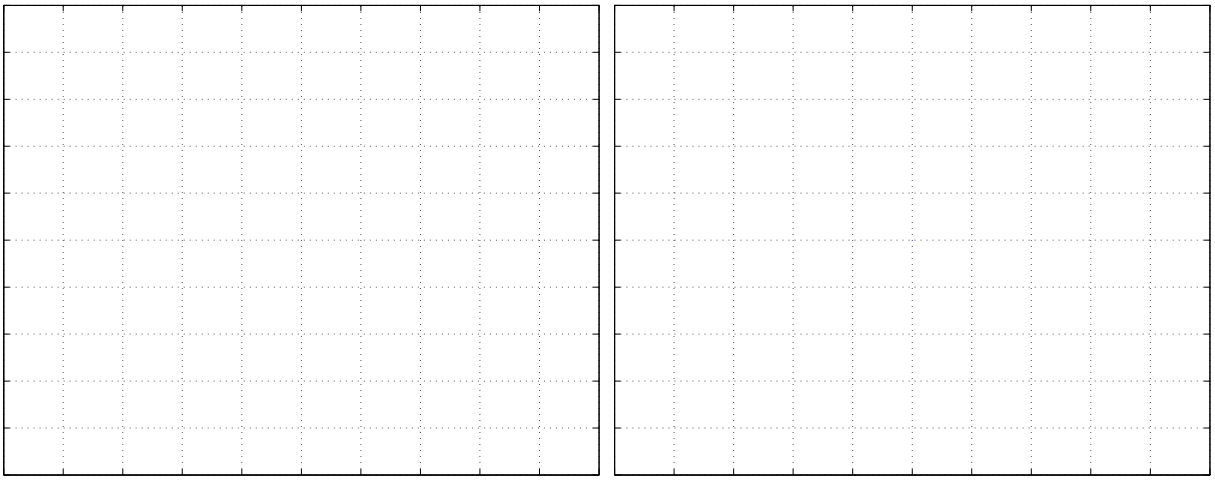
5.2.3 Ostale okenske funkcije

Poleg trikotnega in pravokotnega okna ima Matlab vgrajenih več okenskih funkcij. Različna okna so primerna za različna področja uporabe. Narišite naslednje okenske funkcije (`stem`) in njihove DTFT(`plot`) za velikost okna 31 vzorcev in primerjajte rezultate:

Hammingovo okno (`hamming`)(levo) in Hannovo okno (`hanning`) (desno)



Blackmannovo okno (levo) (`blackman`) in Dolph-Chebyshevo okno (`chebwin`) za višino bočnih nihajev 30 in 50 dB (desno).



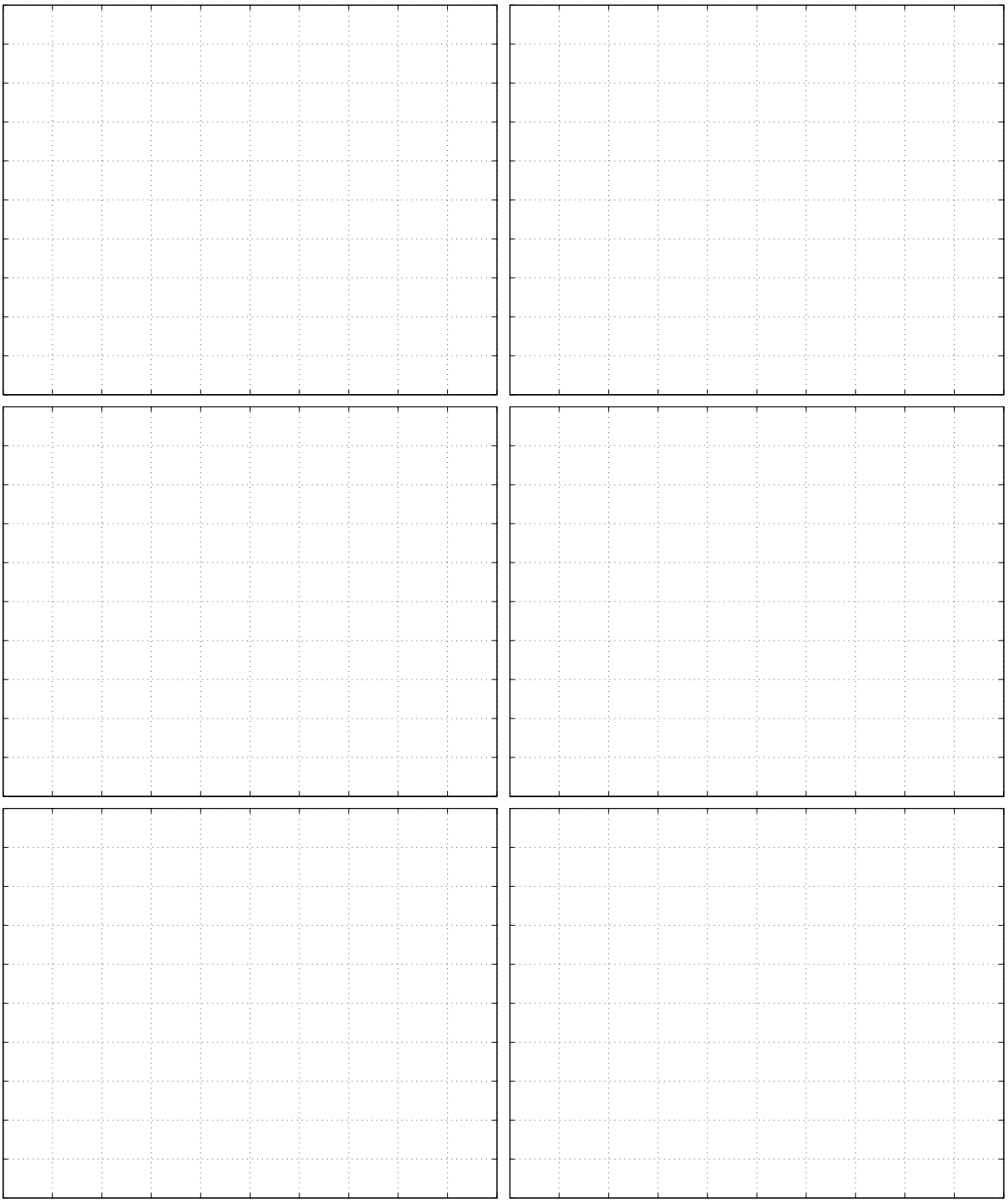
5.2.4 Vpliv oken na frekvenčno ločljivost

Za različne okenske funkcije(npr.`boxcar, bartlett, hamming,...`), narišite potek oknjenega signala in njegovega amplitudnega spektra! V ta namen pripravite oknjeni signal

```
n=0:63;
x=(sin(2*pi*n/4.1)+sin(2*pi*n/3.9))';
xw=x.*OKENSKA_FUNKCIJA(64);
```

Izračunajte amplitudni spekter oknjenega signala:

```
stem(xw)
plot(abs(fft(xw,256)));
semilogy(abs(fft(xw,256)));
```



Kaj se zgodi z obliko signala v frekvenčnem prostoru? Bodite pozorni na potek na celotnem področju kot tudi na frekvenčno ločljivost! Primerjajte učinek različnih okenskih funkcij (lahko pa tudi njihovih dimenzij) med seboj!

Vaja 6
Načrtovanje digitalnih sit s končnim impulznim odzivom

6.1 Načrtovanje nizkega sita s frekvenčnim vzorčenjem

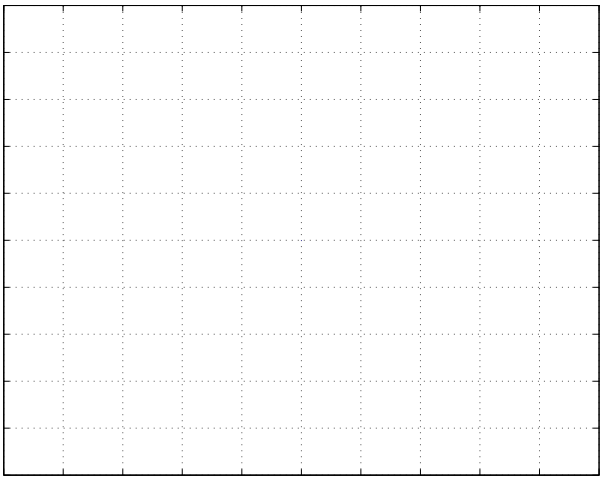
Načrtajte nizko FIR sito 23 reda z linearno fazo in s frekvenco rezanja pri $\omega_0 = 0.3\pi$ z metodo frekvenčnega vzorčenja. Predpostavite, da je vzorčna frekvenca enaka 1, kar predstavlja Nyquistovo frekvenco pri $\omega = \pi$!

Najprej oblikujte vektor vzorcev idealnega amplitudnega odziva v frekvenčnem prostoru za frekvence od 0 do π! Iz zahteve po simetričnosti sklepajte, kakšen je manjkajoči amplitudni odziv pri preostalih frekvencah v območju od π do 2π. Za uspešno načrtovanje moramo zagotoviti vzorce v celotni frekvenčni ravnini od 0 do π

Razložite, kako to dosežemo z naslednjimi ukazi v jeziku MATLAB:

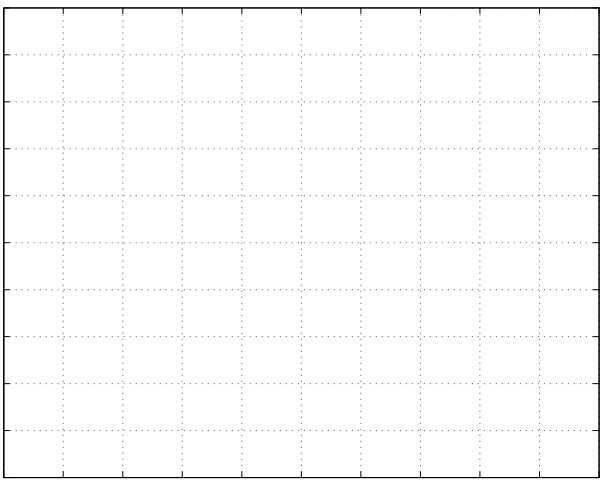
```
pass = fix(w0*L/(2*pi))+1;
if rem (L,2) == 0, s = -1; else s = 1; end;
Ad = [ones(1,pass), zeros(1,L-2*pass+1), s*ones(1, pass-1)]
```

Narišite potek `Ad` in komentirajte idealni frekvenčni odziv filtra. Čemu služi vrstica z `if` stavkom?



Amplitudni odziv filtra v frekvenčnem prostoru je zdaj pripravljen. Oblikujte še vektor faznega poteka; oba skupaj predstavljata kompleksni odziv filtra z linearno fazo. Fazni potek je določen z zahtevo, da ima filter lineani fazni potek in da je kavzalen. Podrobno preučite opisani postopek!

```
M = (L-1)/2;
k = [0:L-1];
p = exp(2*pi*j*(-M)*k/L);
```

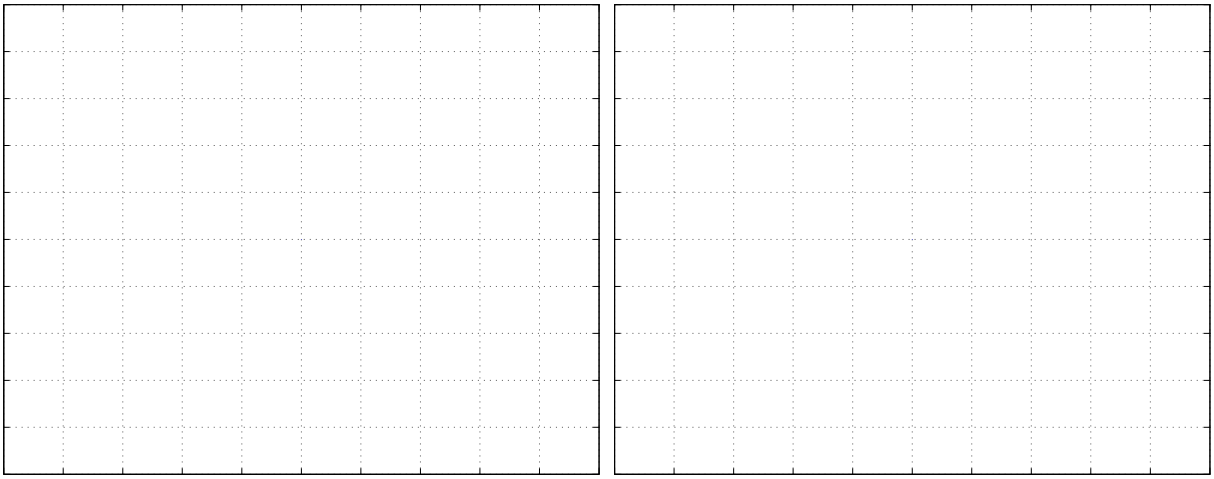


Kompleksni frekvenčni odziv filtra `H` predstavlja zmnožek amplitudnega in faznega vektorja po komponentah (`H=Ad.*p`).

Načrtajte filter z uporabo inverzne diskretne Fourierove transformacije z uporabo funkcije

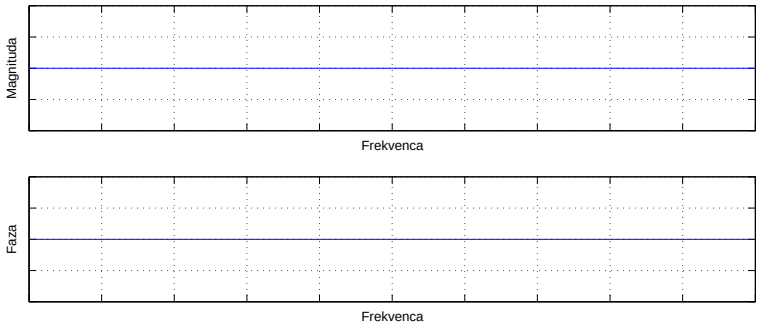
```
h = ifft(H)
```

Do numeričnih napak je dobljeni vektor `h` realen. Če imaginarne komponente niso zanemarljivo majhne, ste se zmotili pri določanju `H`. Prepričajte se z uporabo funkcij `stem(real(h)), stem(imag(h))`.

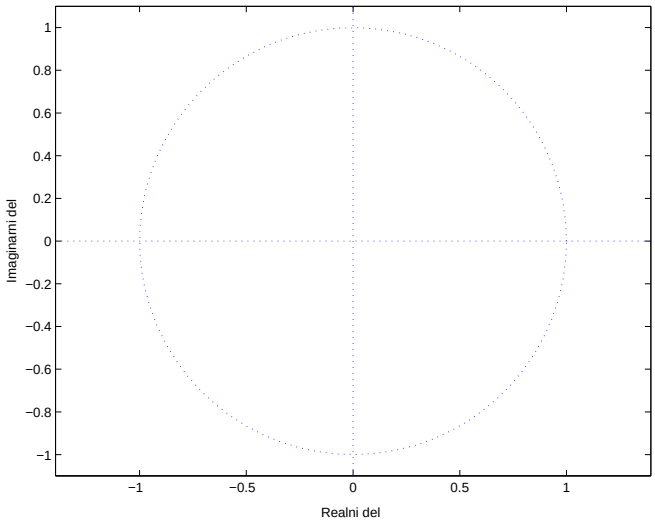


Odstranite majhne imaginarne komponente z ukazom `h=real(h)`!

Preverite delovanje sita, ki ste ga oblikovali po opisanem postopku, s funkcijo `freqz`. Narišite rezultat!

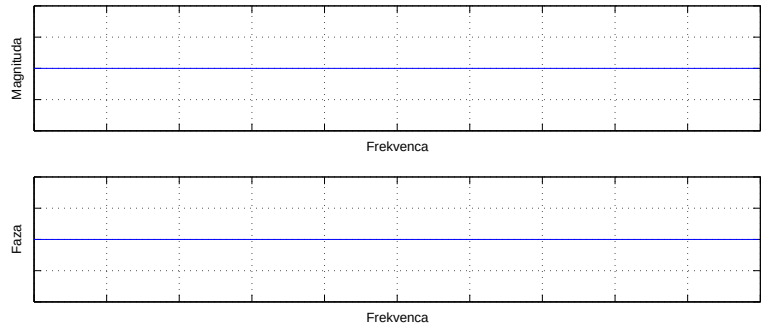


Narišite lego ničel sita v kompleksni z-ravnini z ukazom `zplane`. Povežite lego ničel z obliko frekvenčnega odziva. Povežite število ničel in dolžino filtra.



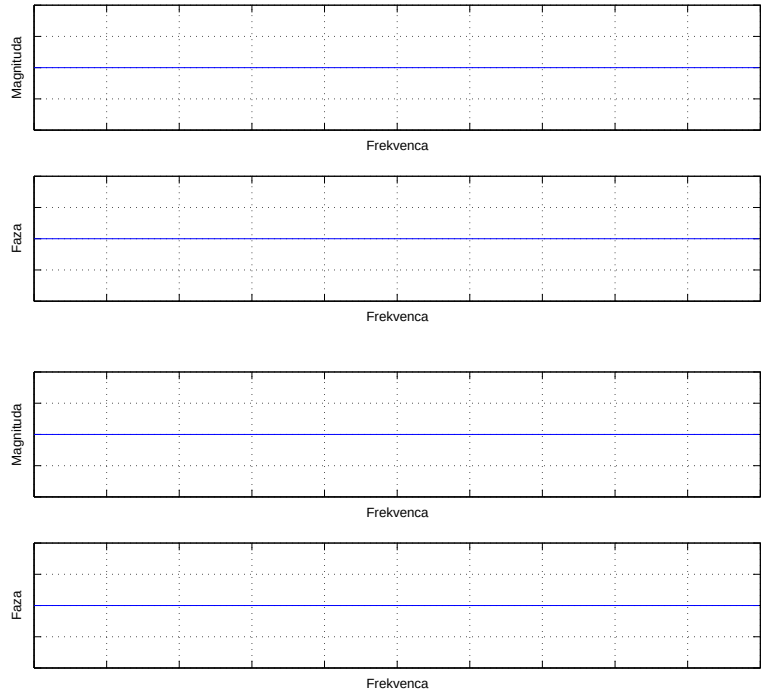
6.2 Nizko sito sode dolžine

Ponovite vajo, vendar za filter dolžine $L = 22$! Opišite razlike glede na sito lihe dolžine! Narišite odziv filtra z ukazom `freqz`! Primerjajte prenihaj v zapori s prenihajem sita z $L = 23$!



6.3 Uporaba okenskih funkcij

Z uporabo okenskih funkcij zmanjšamo učinek ti. Gibbsovega fenomena. Za filter iz prve točke uporabite okenski funkciji `triang` in `hamming`.



Primerjajte odziv filtra z odzivom iz prve točke naloge!

Dodatek: Pasovna in zaporna sita

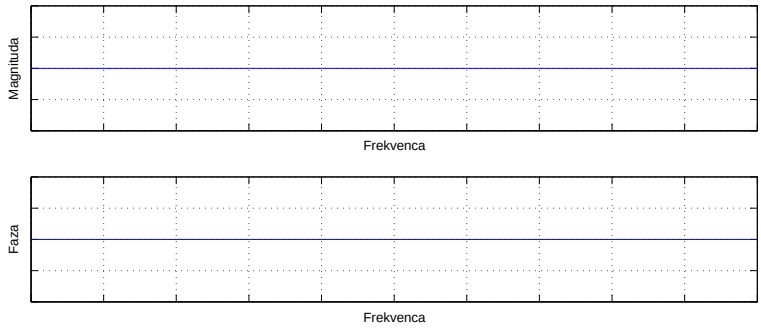
Pasovna in visoka sita lahko izpeljemo iz ustreznih nizkih sit. Zaradi linearnosti inverzne diskretne Fourierove transformacije velja:

$$H_{pp} = H_{n1} - H_{n2}$$

$$h_{pp}[n] = h_{n1}[n] - h_{n2}[n]$$

Velja tudi, da se nizko sito prezrcali v visoko sito z množenjem koeficientov z $(-1)^n$. Na kakšen način se to zgodi?

V dveh korakih načrtajte pasovno zaporno sito 31. reda z $\omega_1 = 0.2\pi$ in z $\omega_2 = 0.3\pi$.
Verificirajte rezultat z uporabo funkcije `freqz`!



Vaja 7
Filtri z neskončnim impulznim odzivom

7.1 Filter z zarezo

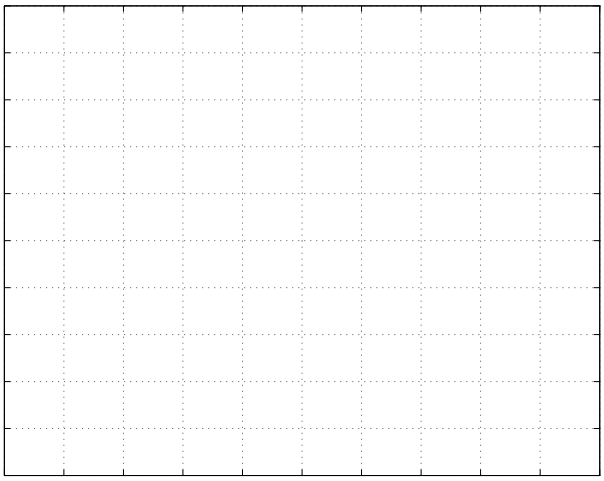
Filter z zarezo (ti."notch" filter) uporabljamo, kadar iz signala želimo izločiti točno odločeno frekvenčno komponeneto. V našem primeru bomo načrtali sistem za obdelavo signalov, ki temelji na uporabi signalnega procesorja. Uporabili ga bomo za odstranjevanje motilne frekvence 50 Hz, ki se pogosto pojavi v signalih zaradi motenj, ki jih povzročajo elektroenergetski sistemi.

1. Filter z zarezo naj ima naslednji frekvenčni odziv:

$$H(e^{j\omega}) = \frac{[1 - e^{-j(\omega-\omega_0)}][1 - e^{-j(\omega+\omega_0)}]}{[1 - re^{-j(\omega-\omega_0)}][1 - re^{-j(\omega+\omega_0)}]} \tag{7.1}$$

   Narišite amplitudni potek $H(e^{j\omega})$ v območju $0 \leq \omega \leq \pi$ za normirano frekvenco $\omega_0 = 2\pi/5$ in $r = 0.9$!

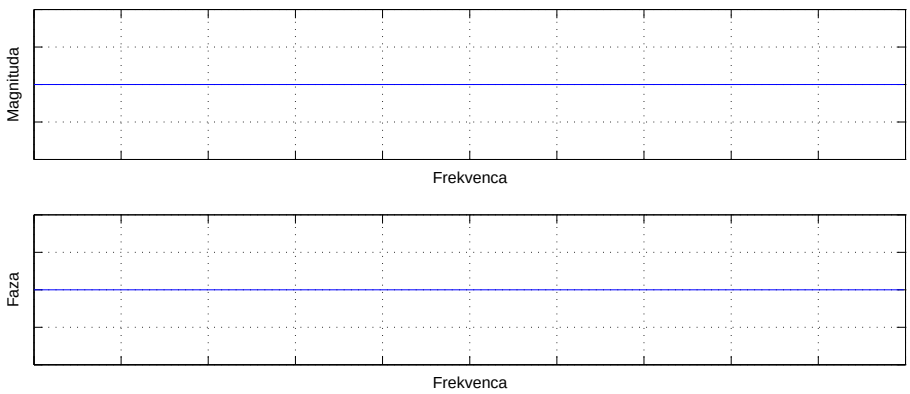
```
r=0.9
w0=2*pi/5
w=0:0.01:pi;
H=(1-exp(-j*(w-w0))).*(1-exp(-j*(w+w0)))./((1-r*exp(-j*(w-w0))).*(1-r*exp(-j*(w+w0))));
semilogy(abs(H);
```

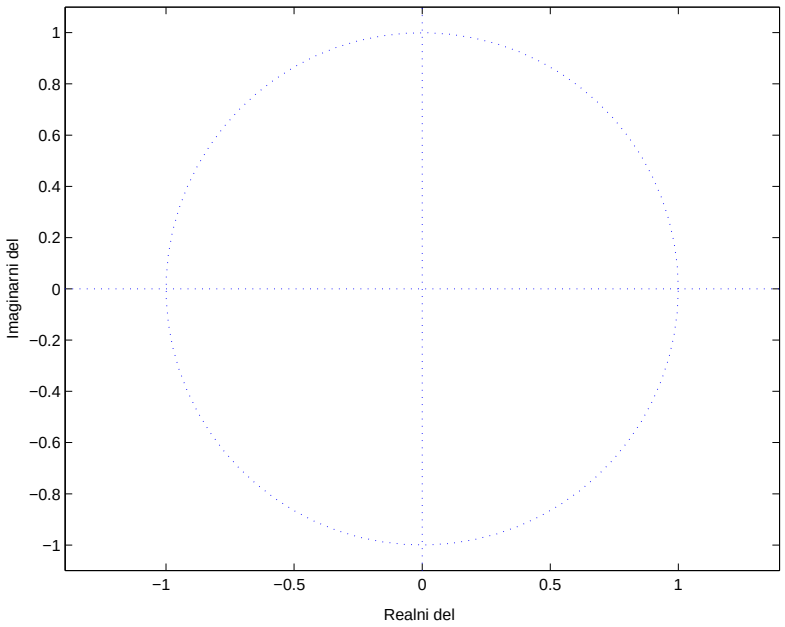


2. Izpeljite koeficiente a in b filtra $H(z) = \frac{b_0+b_1z^{-1}+b_2z^{-2}}{a_0+a_1z^{-1}+a_2z^{-2}}$ za poljuben ω_0. Pri tem upoštevajte, da je $z = e^{j\omega}$; nadomestite ustrezne člene v enačbi 7.1 in skrajšajte rezultat!

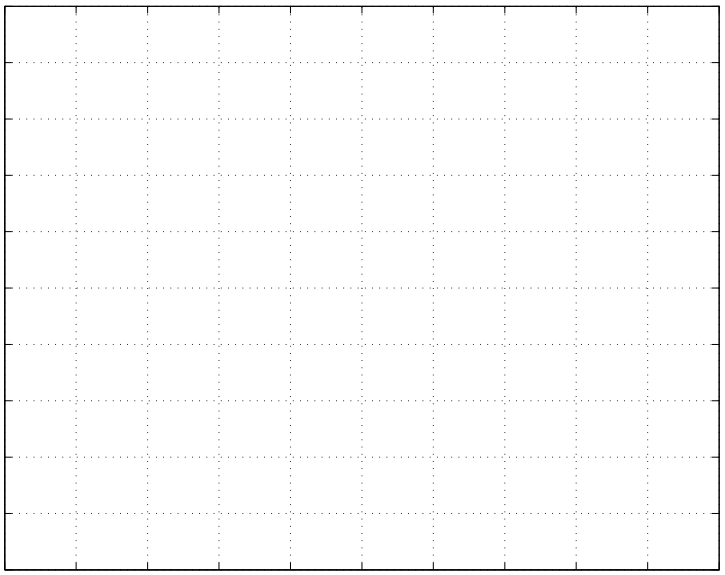
```
a = [_____________________________];

b = [_____________________________];
```

3. Izračun verificirajte tako, da uporabite za $\omega_0 = 2\pi/5$; novo prenosno funkcije narišite z ukazom `freqz`! Oblika prenosne funkcije se mora ujemati z izračunom iz prve točke naloge. Oglejte si lego korenov filtra z ukazom `zplane`.





4. Oblikujte in narišite 150 vzorcev sinusnega signala frekvence 50 Hz pri vzorčni frekvenci $f_s = 1000Hz$.

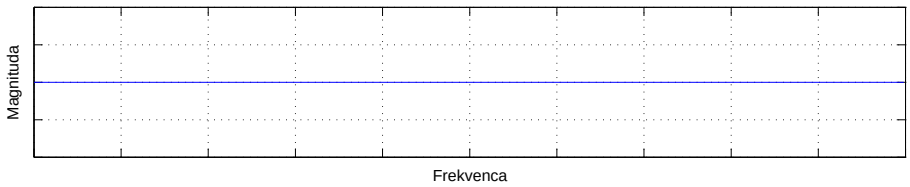


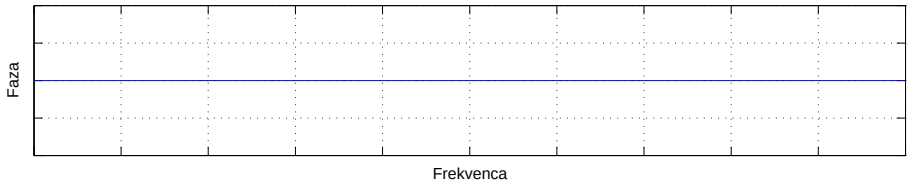
Izračunajte ustrezno **normirano frekvenco** $\omega_0 = 2\pi \frac{f}{f_s}$, nato pa še koeficiente filtra `a` in `b`, ki izloča 50 Hz motilni signal.

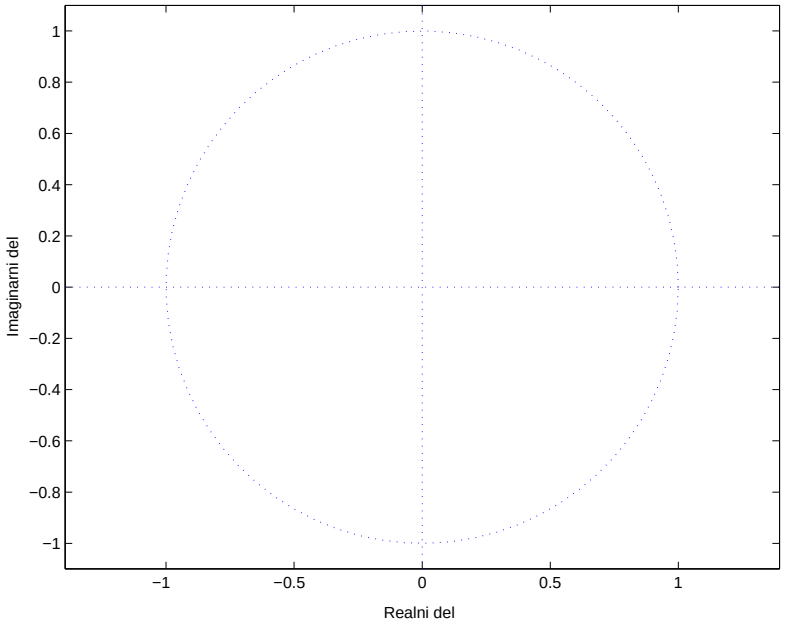
```
a = [_____________________________];
```

```
b = [_____________________________];
```

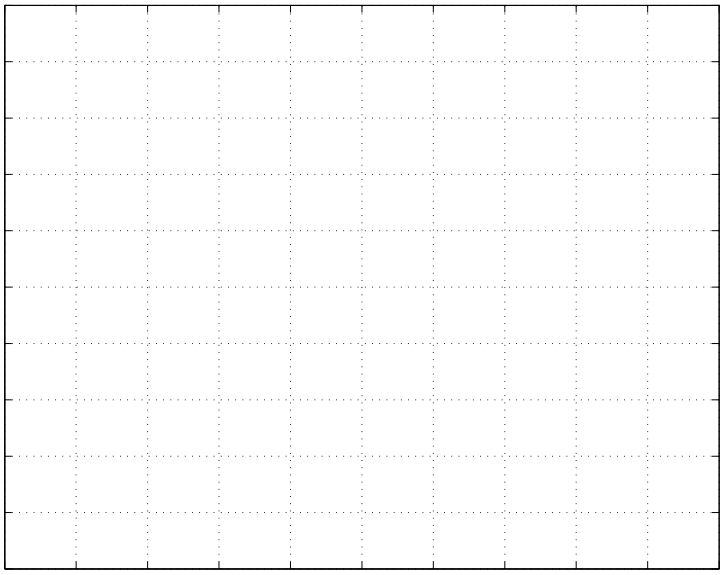
Oglejte si prenosno karakteristiko novega filtra in lego njegovih korenov!







Filtrirajte signal z ukazom `filter` in narišite rezultat! Komentirajte!



7.2 Splošno načrtovanje IIR filtrov

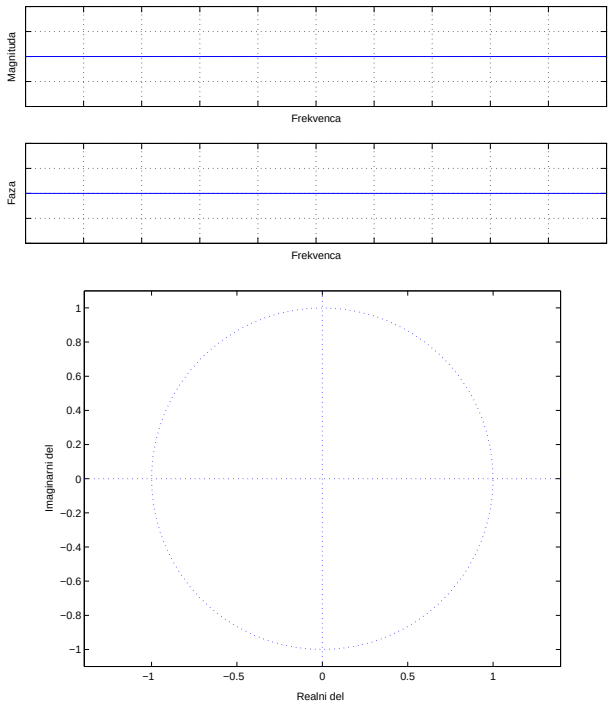
Digitalni filter z neskončnim impulznim odzivom (IIR) je najbolj splošna linearna struktura za digitalno obdelavo signalov. Njeno obnašanje predstavlja rekurzivna linearna diferenčna enačba

$$y(n) = -\sum_{k=1}^{N} a_k y[n-k] + \sum_{m=0}^{M} b_m x[n-m] \tag{7.2}$$

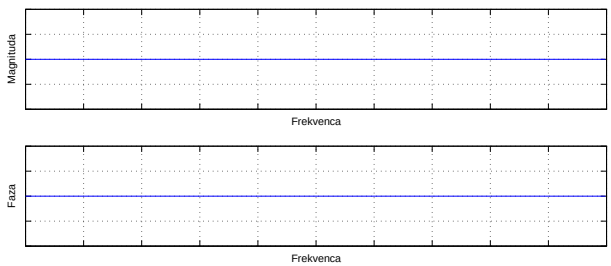
pri čemer koeficienti a_k in b_m niso funkcije časovno diskretne spremenljivke n. Poseben primer je filter s končnim impulznim odzivom, kjer je $N = 0$, uporabimo pa le predhodne vhodne vrednosti. Prednost filtrov z neskončnim impulznim odzivom v primerjavi s filtri FIR je v učinkovitosti; načrtovalske zahteve lahko izpolnimo z znatno nižjimi redi filtra. Težave pri uporabi nastopajo predvsem zaradi stabilnosti takšnih filtrov, ki postanejo še izrazitejše zaradi kvantizacije signalov. Izvedba filtrov ne omogoča linearnega faznega odziva, ki se mu lahko le približamo. Povratna vezava znotraj filtra tako ne prinaša le pozitivnih, ampak tudi neželene lastnosti. Načrtovalski problem je, kako iz specifikacije filtra v frekvenčnem prostoru poiskati optimalne uteži filtra a_k in b_m. Filter z neskončnim impulznim odzivom je diskretni ekvivalent zveznih RLC vezij oz. aktivnih RC filtrov. Tako mnogi načrtovalski postopki predvidevajo, da najprej načrtamo ustrezen filter v zveznem prostoru, tega pa nato prevedemo v ustrezno obliko v diskretnem časovnem prostoru. Pri tem se bomo danes osredotočili na pretvorbo in ne na analogno specifikacijo filtra.

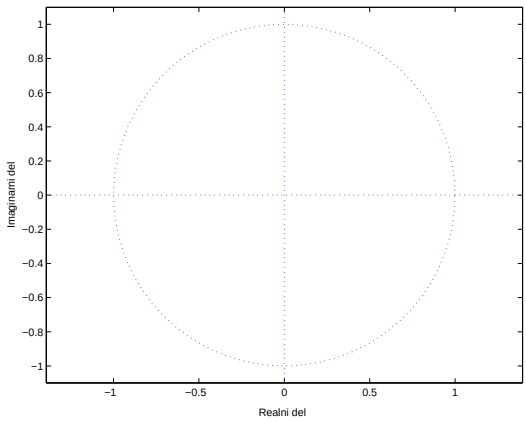
Poznamo 4 osnovne tipe IIR filtrov, ki predstavljajo 4 različne kombinacije dveh kriterijev: razvoja v Taylorjevo vrsto in maksimalne razlike med želenim in dejanskim frekvenčnim odzivom. To so Butterworthov filter, filter Chebysheva, filter Chebysheva II in eliptični filter. Načrtovalski postopki bodo temeljili na pretvorbi analognih prototipov filtrov v diskretno IIR obliko. Matlab ima na voljo neposredne ukaze za načrtovanje diskretnih IIR filtrov, to so `butter`, `cheby1`, `cheby2` in `ellip`. Za risanje odzivov v frekvenčnem prostoru uporabite funckijo `freqz`. Lego polov in ničel v z ravnini boste najlažje predstavili z ukazom `zplane`.

1. Z ukazom `butter` načrtajte Butterworthov filter 5. reda s frekvenco rezanja pri 600 Hz. Vzorčna frekvenca naj bo 2000 Hz. Narišite odziv filtra v frekvenčnem prostoru ter lego polov in ničel v z ravnini.

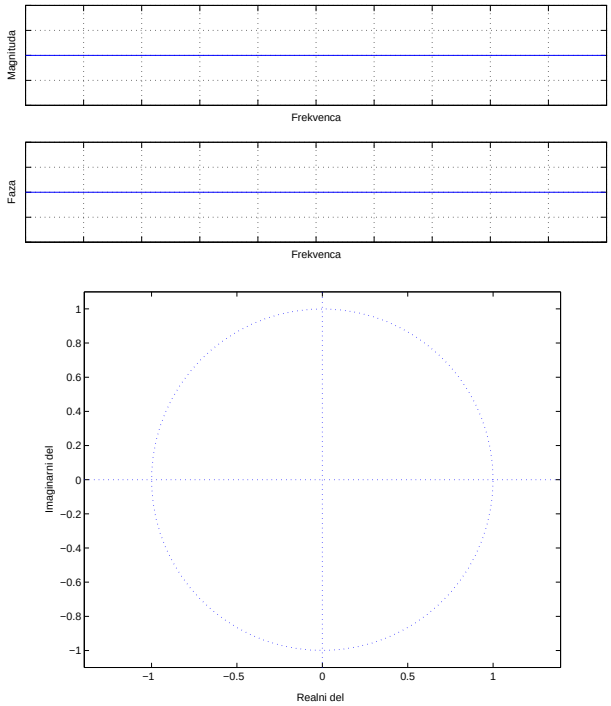


2. Načrtajte tudi filter Chebysheva enake specifikacije. Prenihaj v prepustnem pasu naj bo enak 0.5 dB.





3. Ponovite primer za eliptični filter. Prenihaj v zapornem pasu naj bo za 30dB nižji od prenihaja v prepustnem pasu filtra.



4. Načrtati moramo filter, ki deluje pri vzorčni frekvenci 2000 Hz. Prenihaj v prepustnem pasu naj bo 0.1, v zapornem pa pod 30 dB. Rob prepustnega pasu izberemo pri 280 Hz, zapornega pa pri 320 Hz. Z ukazi `buttord`, `cheb1ord`, `cheb2ord` in `ellipord` ugotovite, kakšen je minimalni red filtrov posameznega razreda, ki že ustreza specifikacijam.

Vaja 8
Dvodimenzionalni signali

8.1 Obdelava slik

V okviru vaje se bomo seznanili z osnovami obdelave dvodimenzionalnih signalov. Ogledali si bomo digitalni zapis enakomerno vzorčene črno-bele slike. Testne slike predstavljajo matrike dimenzij 256×256, intenziteto posameznih točk predstavljajo realne vrednosti v dinamičnem območju od 0 do 1.

Slika 8.1: Fotograf

Poskrbite za ustrezen prikaz slike v okolju Matlab. Napišite funkcijo za dvodimenzionalno filtriranje slike. Preizkusite delovanje funkcije. Na sliki preizkusite filter za glajenje in za ostrenje slike. Kako bi oblikovali filter, ki predstavlja tresenje roke fotografa? Oglejte si rezultate!

Naloga

1. Z domače strani predmeta (na strani http://ldos.fe.uni-lj.si) v svoj delovni direktorij prekopirajte testno sliko za izvedbo naloge.

2. Naložite sliko iz datoteke. (`load foto.mat`).

3. Matriko podatkov, ki predstavlja sliko, si lahko ogledate z ukazom `imshow`.

4. Sliki dodajte gaussov šum:

   ```
   foto_gs = imnoise(foto,'gaussian',0,0.01)
   ```

   Oglejte si novo sliko. Kako vpliva dodajanje Gaussovega šuma na sliko?

5. Sliki dodajte šum "sol in poper':

   ```
   foto_sp = imnoise(foto,'salt & pepper',0.02)
   ```

   Kakšen je šum 'sol in poper' v primerjavi z Gaussovim šumom?

6. V okolju Matlab filtre s končnim impulznim odzivom predstavljajo matrike ustreznih dimenzij. Običajno ne uporabljamo filtrov velikih dimenzij zaradi popačenj slik na robovih in zaradi izračunske zahtevnosti. Dejansko filtriranje opravimo s funkcijo `filter2`. Preizkusite delovanje nizkopasovnega Gaussovega filtra na slikah z in brez šuma!
   (`B=fspecial('gaussian',N,STD)`; N=3 ali 5, STD=0.1, 1, 10 ali 100).

7. V nekaterih primerih boljše rezultate dobimo z uporabo nelinearnih postopkov filtriranja. Na slikah uporabite še filter mediane `medfilt2`. Pomagajte si s pomočjo, ki jo nudi okolje Matlab, razložite delovanje filtra mediane in primerjajte rezultate z rezultati Gaussovega filtra.

8. Eden od filtrov za iskanje robov je Laplace-ov operator. Predstavlja ga matrika

$$B = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}.$$

   Preizkusite delovanje filtra na sliki s šumom in brez šuma. Komentirajte!

9. Kako bi filter za iskanje robov spremenili v filter, ki robove samo poudari? (Namig: seštejte rob in originalno sliko z enom samim *linearnim* operatorjem!) Kaj s tem dosežemo? Komentirajte dobljene rezultate!

10. Miselno začrtajte filter, ki predstavlja tresenje roke fotografa in komentirajte rezultate!

Dodatek: Namesto slike `foto.mat` lahko izberete slike `bact.mat` in `riz.mat`.

8.2 Zgoščevanje slik

V okviru vaje se bomo seznanili tudi z zgoščevanjem dvodimenzionalnih signalov. Testno sliko predstavlja matrika dimenzij 256×256. Razdelite sliko na bloke velikosti 8×8 slikovnih elementov (pikslov). Vsak blok posebej preslikajte v frekvenčni prostor z uporabo diskretne kosinusne preslikave.

Zgoščevanje slike boste simulirali tako, da posameznih frekvenčnih komponent v prostoru diskretne kosinusne preslikave ne boste upoštevali pri kodiranju slike - privzeli boste, da je njihova vrednost enaka 0. Elementov, katerih vrednost je enaka 0, nam na ta način ni potrebno zapisati; v tem se skriva možnost za zgoščen zapis slike. Pravi program bi moral izvesti tudi kodni zapis, s čimer bi dosegli dejansko zgoščevanje, mi pa privzamemo, da je razmerje zgoščevanja slike enako razmerju **izničeni elementi : vsi elementi bloka**. Matrika, ki poskrbi za izničevanje elementov, naj bo enaka za vse bloke v sliki. Izdelek najprej verificirajte z zgoščevanjem 1:1. Nato uporabite zgoščevanje 1:64, 1:16, 1:8, 1:4 in 1:2. Komentirajte rezultate in učinkovitost postopka za zgoščevanje.

1. Naložite sliko iz datoteke. (load foto.mat) Za prikaz uporabite znanje iz prvega dela naloge.

2. Pretvorite sliko po blokih velikosti 8×8 v frekvenčni prostor z uporabo diskretne kosinusne preslikave. Uporabite funkcijo `dct2`. Oglejte si intenzitete v DCT prostoru in pojasnite, kje so možnosti za zgoščevanje slike. Nasvet: rezultate za celotno sliko ohranite v matriki, ki dimenzijsko ustreza matriki slike. Za obdelavo slike po blokih uporabite funkcijo `blkproc`, ki jo pozna Matlab:
   `foto_DCT=blkproc(foto,[8 8],'dct2');`

3. Postavite vrednosti izbranih komponent slike po posameznih blokih na 0. V ta namen uporabite matrike, ki so že pripravljene v datoteki `mask.mat`. Naložite vrednosti iz datoteke (`load mask`! V delovnem okolju boste dobili matrike `mask01, mask02, mask04, mask08, mask16, mask64`, ki prikazujejo učinek zgoščevanja 1:1, 1:2, 1:4, 1:8, 1:16 in 1:64. Elemente v posameznem bloku izničite tako, da ustrezni blok po komponentah pomnožite z ustrezno matriko `mask`. Matrika `maskXX` ima vrednosti 1 v levem zgornjem predelu, njihovo število pa je odvisno od razmerja zgoščevanja! Učinkovito boste nalogo opravili z naslednjim ukazom (ukaz inline nadomesti potrebo za pisanje namenske funkcije za blokovno množenje po komponentah):
   `foto_DCT_K=blkproc(foto_DCT,[8 8],inline('x.*QM','x','QM'),maskXX);`

4. Z uporabo funkcije `idct2` po blokih preslikajte sliko iz frekvenčnega nazaj v krajevni prostor. Izvedba operacije je enakovredna operaciji iz točke 2 današnje naloge. Kako se posledice zgoščenega zapisa pokažejo na sliki? Komentirajte rezultate za razmerja, navedena v uvodu naloge!

Opomba: Predstavljeni način za zgoščevanje slik seveda le nakazuje enega od možnih pristopov za zmanjšanje potrebne količine podatkov, ki predstavljajo posamezno statično sliko. Resnični algoritmi za zgoščevanje slik upoštevajo velikost posameznih koeficientov in izničijo izključno majhne vrednosti glede na zahtevano stopnjo zgoščevanja.

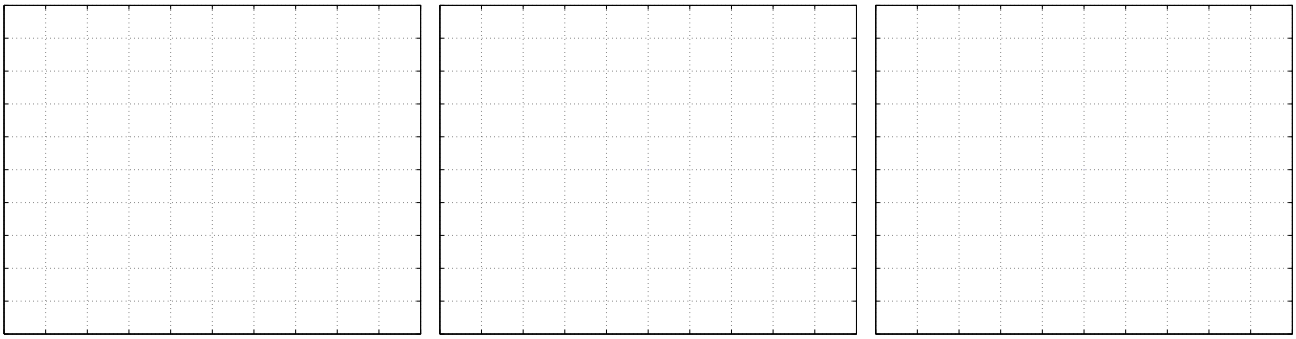
Vaja 9
Kvantizacija in digitalna obdelava signalov

V idealnih razmerah , torej v primeru, da bi parametre filtrov, vmesne rezultate in vhodne podatke obravnavali s poljubno visoko natančnostjo, bi se digitalni sistemi obnašali tako, kot to določa teorija, ki smo jo dosedaj spoznali. Žal je natančnost računanja končna, to pa vodi k bolj ali manj nepričakovanemu obnašanju sistemov; obnašanje sistemov je v tem primeru odvisno ne le od izbranih koeficientov, ampak tudi od strukturne izvedbe sistema. Omejena natančnost se z aritmetičnimi operacijami nad spremenljivkami lahko še dodatno povečuje. Omejena natančnost v digitalne sisteme vnaša določeno stopnjo nelinearnosti, ta pa lahko popolnoma spremeni obnašanje sistema. S povečano natančnostjo računanja morebitne težave sicer olajšamo, povsem odpraviti pa jih ni mogoče. Splošni pristop k problemu končne natančnosti računanja je matematično zahteven in takorekoč neizvedljiv. Ogledali si bomo le nekaj vplivov omejene natančnosti računanja.

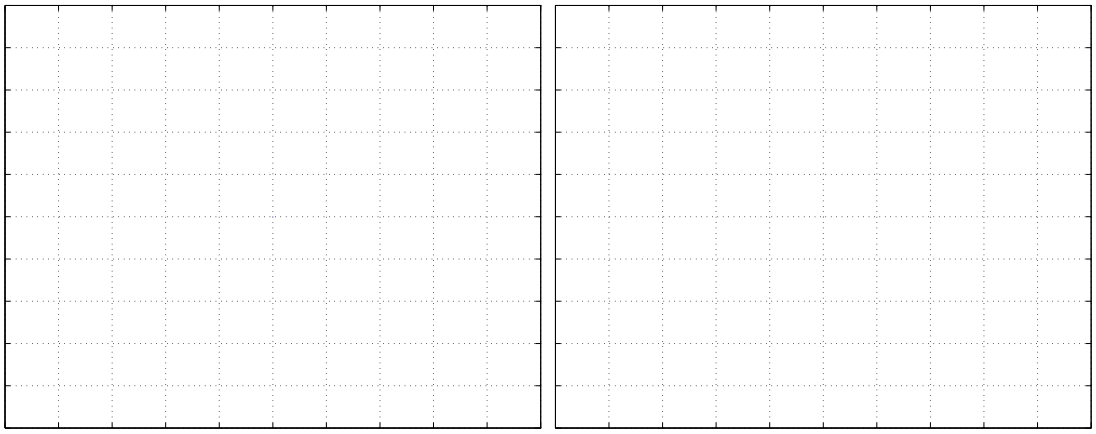
9.1 Kvantizacija signalov

1. Oglejte si in preučite program za simulacijo kvantizacije `fxquant`. Program v parametrični obliki omogoča vnos vhodnega signala in natančnost v bitih.

2. Preizkusite delovanje kvantizatorja na linearno naraščajočem zaporedju $v(n)$, ki narašča od -1.1 do 1.1 v koraku 2^{-6}. Uporabite 3 bitno natančnost. Izračunajte kvantizirani signal `vq` in kvantizacijsko napako `dq = vq - v`. Narišite signal, kvantizirani signal in kvantizacijsko napako v odvisnosti od časovnega indeksa.



Narišite še kvantizirani signal in kvantizacijsko napako v odvisnosti od signala.



3. Raziščite lastnosti zaporedja napake $e(n)$ za naslednje signale:

   - $v_1[n]$ je naključni signal uniformne porazdelitve v območju $(-1, 1)$ (ukaz `2*rand(1,8000)-1` - zakaj množimo z 2 in odštejemo 1?).
   - $v_2[n]$ je naključni signal normalne porazdelitve z varianco 1 (ukaz `randn(1,8000)`).
   - $v_3[n] = a\sin[n\omega], a = 1\omega = 2\pi/8000, n = 0 : 8000$

   Uporabite 8000 elementov zaporedja zgoraj navedenih signalov in izračunajte zaporedja napake

   $$e[n] = v[n]_Q - v[n]$$

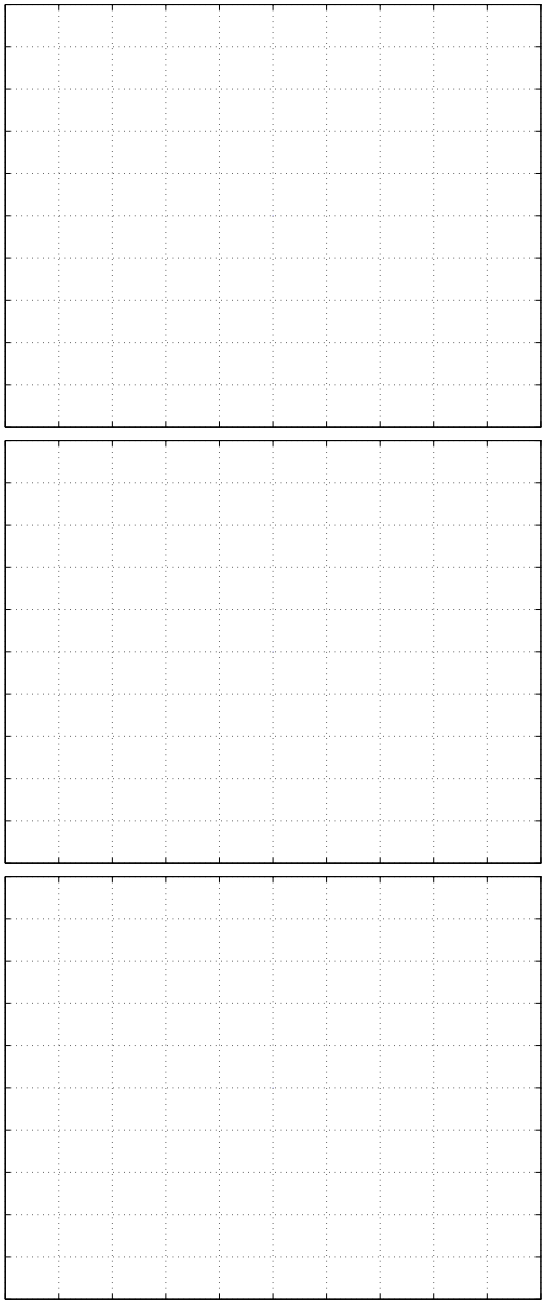
   za 8 bitno besedo z zaokrožanjem in z nasičenjem. Izračunajte srednjo vrednost in varianco (`mean, std`) za vse tri primere.

   _____

   _____

   _____

   Narišite verjetnostno gostoto $e[n]$ (`hist(e,20)`) napake kvantizacije za vse tri primere!

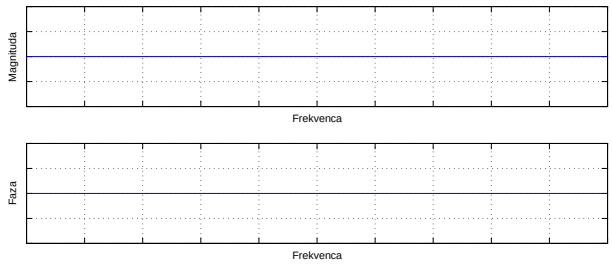


9.2 Kvantizacija koeficientov filtra

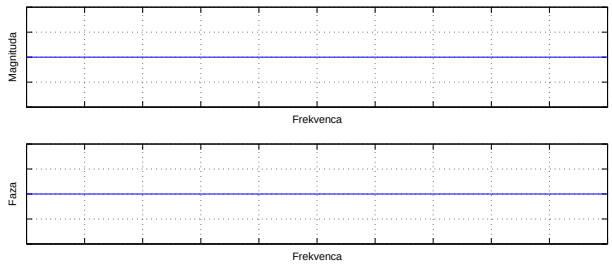
Ogledali si bomo, kako se lastnosti filtrov spremenijo, če koeficiente računamo s končno natančnostjo. Vzemimo za primer koeficiente IIR filtra. Poudariti je potrebno, da so učinki kvantizaciije koeficientov filtra močno odvisni od strukture uporabljenega filtra.

1. Izračunajte koeficiente eliptičnega filtra 5. reda s frekvenco rezanja pri normalizirani frekvenci $0,5$, s prenihajem v prepustnem pasu $0,5dB$ in s prenihajem v zapornem pasu $30dB$. Uporabite ukaz `[b,a] = ellip(5,0.5,30,0.5)`.

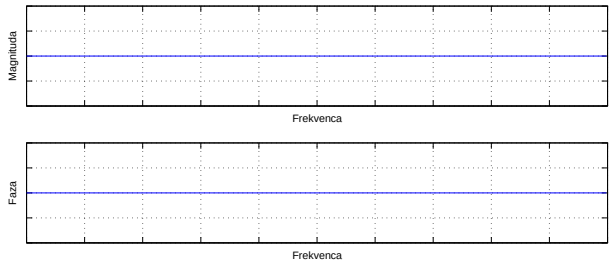
2. Za kasnejšo primerjavo izračunajte frekvenčni odziv filtra Uporabite ukaz `freqz(b,a)`.



3. Kvantizirajte koeficiente z zaokrožanjem, upoštevajte 6 bitno natančnost ($B = 6$). Zaokrožanje opravite z ukazom
   `aq=fxquant(a,6,'round','none')`
   `bq=fxquant(b,6,'round','none')`.
   Izračunajte in narišite frekvenčni odziv filtra s kvantiziranimi koeficienti.



4. Preverite učinek kvantizacije tako, da na isti graf narišete obe sliki!



9.3 μ-zakon

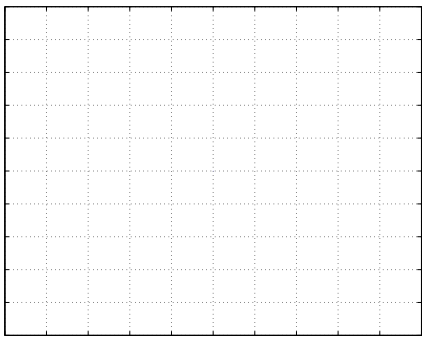
S problemom kvantizacije se ne srečujemo le na področju občutljivosti filtrov. Enakomerna kvantizaacija nam na signal vpliva neodvisno od njegove dinamike, torej neodvisno od tega, kako veliko amplitudo imajo vzorčeni signali. Ob uporabi grobe kvantizacije se tako lahko zgodi, da tihi zvoki zaradi svoje nizke amplitude po kvantizaciji popolnoma izginejo, saj je njihova amplituda manjša od koraka kvantizacije. Z uporabo zakona μ signal pred kvantizacijo priredimo tako, da bo kvantizacijska napaka sorazmerna jakosti signala.

Zgoščevanje po zakonu μ podaja naslednja enačba:

$$y[n] = X_{\max} \frac{\log\left[1 + \mu \frac{|x[n]|}{X_{\max}}\right]}{\log(1+\mu)} \cdot \operatorname{sign}(x[n]) \tag{9.1}$$

Tovrstne postopke uporabljano predvsem v digitalni telefoniji.

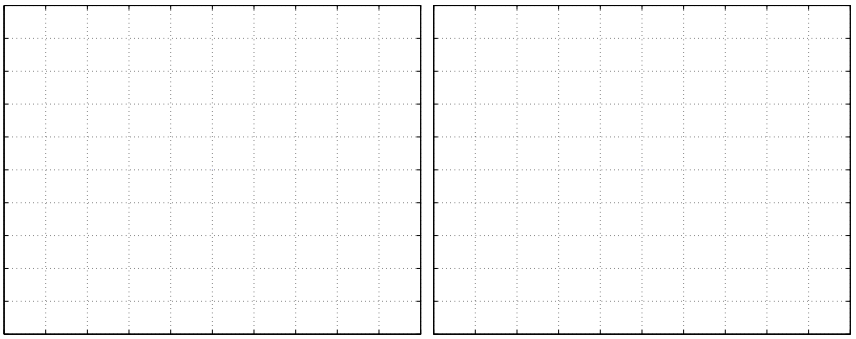
1. Oblikujte enakomerno naraščajoči vektor `[0:0.0005:1]` in narišite funkcijo μ zakona za $\mu = 100$, 255 in 500. Vse funkcije narišite na enem grafu. Uporabite funkcijo `mulaw`, ki se nahaja na strežniku LDOS. (http://ldos.fe.uni-lj.si/).



2. Iz datoteke `s5.mat` naložite vzorčen govorni signal. Narišite originalni signal in signal po μ zakonu. Primerjajte časovni potek (opišite) in skicirajte histograma signalov.

_____

_____



3. Po vzoru funkcije `mulaw` napišite inverzno funkcijo μ zakona `mulawinv`. Preverite njeno delovanje.

4. Na kvantiziranem signalu preverite učinek uporabe μ zakona. Uporabite 6 bitni kvantizator. Oglejte si učinke. Uporabite kvantizator, ki se nahaja v datoteki `fxquant.m` z naslednjimi argumenti: `BIT=6, RMODE='round', LMODE='sat'`.