



KANDÓ KÁLMÁN

VILLAMOSMÉRNÖKI KAR

---

Kún Gergely – Tóth Ádám – Fekete Zoltán

# Híradástechnika a gyakorlatban

ÓE-KVK 2131

Budapest, 2015

# Tartalomjegyzék

1	Bevezető.....	4
2	MatLab gyakorlati bevezető.....	5
2.1	Változók kezelése .....	5
2.2	Az „m” fájlok használata .....	8
2.3	Függvények használata .....	11
2.4	Számítási példa mátrixokkal.....	13
3	Analóg és digitális modulációk.....	21
3.1	Modulációs eljárások .....	21
3.2	Analóg modulációs eljárások áttekintése .....	21
3.3	Feladatok .....	24
3.4	Digitális modulációs eljárások áttekintése .....	27
3.5	Feladatok .....	30
4	Képek kezelése a MatLab-ban .....	33
4.1	Színes képek .....	33
4.2	Szürkeárnyaltos képek .....	35
5	Digitális jelek .....	43
5.1	Mintavételezés, a mintavételi tétel .....	43
5.2	Kvantálás .....	48
6	Jelek spektrális felbontása, Fourier transzformáció .....	56
6.1	Általános bevezető.....	56
6.2	Fourier sor alakjai .....	59
6.3	Feladatok .....	63
7	Lineáris torzítás hatásainak vizsgálata.....	69
7.1	Feladatok .....	70
8	Diszkrét idejű hálózatok szimulációja .....	72
8.1	Elméleti áttekintés .....	72
8.2	Feladatok .....	74
9	FIR szűrők tervezése.....	81

9.1	Elméleti áttekintés .....	81
9.2	Feladatok .....	88
10	IIR szűrők tervezése.....	93
10.1	Elméleti áttekintés .....	93
10.2	Feladatok .....	94
11	Forrásjegyzék.....	98

# 1 Bevezető

Ez a jegyzet a Híradástechnika című tantárgyakhoz készült, útmutatást és kiegészítést ad az elméleti órán elhangzottak gyakorlatban való alkalmazásához. A jegyzet által összefoglalt elméleti ismeretek és gyakorlati példák a tantárgyakhoz kapcsolódó laboratóriumi órákon is hasznosnak bizonyulhatnak, de semmiképpen sem pótolja az előadást és a gyakorlatokon elhangzottakat.

A jegyzet a tárgyalt példákat MatLab környezetben demonstrálja, mellékelt kódok futtatásával, módosításával világít rá a későbbi tanulmányokban is fontos összefüggésekre és ismeretekre.

A jegyzet első fejezete általános áttekintést ad a MatLab-ban használt gyakori parancsok, számítási módok, az „m” szkriptek és a függvények használatáról, mely ismeretek elengedhetetlenek a későbbi fejezetekben található példák megoldásaihoz.

Külön köszönetet szeretnénk mondani a könyv szakmai lektorának, Dr. Wühl Tibornak, aki sok szakmai észrevétellel, alkotó kritikájával tette jobbá ezt a jegyzetet.

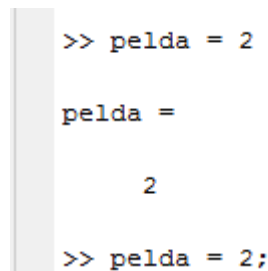
## 2 MatLab gyakorlati bevezető

A MatLab-ot a mérnöki gyakorlatban számos tervezési és szimulációs feladat esetében kiválóan alkalmazhatjuk. Használatának elsajátításával nemcsak ezt a szoftvert, de számos, a MatLab kezelő felületéhez hasonló, matematikai programot is képesek leszünk használni (például a Linux rendszerek alá ingyenesen elérhető FreeMath vagy Octave).

### 2.1 Változók kezelése

A MatLab az adatokat mátrix formában tárolja. Adatsorokat vagy egyetlen értéket tartalmazó változókat is megadhatunk, de ezeket mind egy NxM méretű mátrixban fogja tárolni (N és M természetesen lehet akár 1 is). A mátrix celláiban tárolt értékek alapján különböző változó formátumokat kezel. Ezen változó típusok hasonlatosak a magas szintű programnyelvekben megismert változó típusokhoz. Egy változó létrehozásakor viszont nem kell törődnünk a típusválasztással, azt a program magától elvégzi az értékadásakor. Fontos megjegyezni, hogy az értékadás sorrendje balról jobbra történik, vagyis először a feltölteni kívánt változó nevét kell megadni, majd azt az értéket, amit a változóban tárolni szeretnénk. Ez lehet egy direkt érték megadása, vagy más változóból áttöltött, esetleg függvénnyel létrehozott adat. Az értékadás műveletét (értékadás operátor), az „=” jelzi.

<VÁLTOZÓ> = <ÉRTÉK>



```
>> pelda = 2  
  
pelda =  
  
2  
  
>> pelda = 2;
```

2-1. ábra. Értékadás példa

A „>>” a parancssori kurzor jele, a sorvégi „;” pedig egy lezáró karakter. Ha nem tesszük ki egy sor végére, akkor az értékadás – vagyis a változó létrehozása – után a MatLab a Command Window-ban rögtön meg is jeleníti a művelet végeredményét, ami hasznos lehet, ha gyorsan ellenőrizni szeretnénk egy művelet végeredményét. Azonban ha csak egy gyors értékadást szeretnénk, például egy későbbi művelethez viszünk be adatot, akkor zavaró lehet a folyamatos listázás. Nagyobb mátrixok, vagy sorvektorok feltöltésekor felesleges is lehet az adatok megjelenítése, ilyenkor hasznos kitenni a pontosvesszőt. Természetesen a Workspace ablakban bármikor ellenőrizhetjük a változók értékét.

A változók neveit nem szabad számmal kezdeni, nem tartalmazhatnak speciális karaktereket és ékezetes betűket. A MatLab különbséget tesz kis- és nagybetűk között. Tehát a

```
>> d = 3;
```

és a

```
>> D = 4;
```

műveletek két különböző változót hoznak létre.

### 2.1.1 Műveletek és változók a parancssorban

A hálózatszámítási műveletek között sűrűn használjuk a párhuzamos ellenállás értékek meghatározásához – a reciprokösszeg számolására – a szakmában egyezményesen használt „replusz” műveletet, mely egy összetett művelet.

Oldjuk meg először a parancssorban a műveletet két párhuzamosan kapcsolt 20  $\Omega$ -os ellenállás eredőjére:

```
>> a=20  
  
a =  
  
    20  
  
>> b=20  
  
b =  
  
    20  
  
>> parhuzamos = (a*b)/(a+b)  
  
parhuzamos =  
  
    10
```

2-2. ábra. Replusz művelet Command Window-ban

Mint látjuk az értékadások után a replusz művelet eredményét a `parhuzamos` nevű változóba mentettük. Megtehetjük persze azt is, hogy nem adunk változó nevet, csupán a műveletet hajtjuk végre. Ilyenkor a MatLab egy alapértelmezett `ans` nevű változóba menti a művelet eredményét. Ezt a változót folyamatosan frissíti, tehát egy újabb művelettel felülírásra kerül!

```
>> (a*b)/(a+b)

ans =

    10

>> a+b

ans =

    40
```

**2-3. ábra. Az ans változó használata**

Ha tehát egy művelet eredményét el akarjuk menteni, akkor egy újabb változóba kell áttöltenünk. A két ellenállás összege pont a soros eredőjük, mentsük el e szerint:

```
>> soros = ans

soros =

    40
```

**2-4. ábra. Az ans változó mentése**

Vegyük észre, hogy egy karaktersor a MatLab-ban vagy változó vagy függvényként kerül értelmezésre. Ha valamilyen karaktersort írunk értékadásra, akkor a MatLab megkeresi a kérdéses változót. Változóinkat érdemes beszédes névvel ellátni, hogy mindig tudjuk, hogy milyen értéket tárolunk bennük.

Változóinknak természetesen adhatunk értéként karakterláncot is. Ez az úgynevezett sztring. Az előző példában a `soros` változó az `ans` nevű változó értékét vette fel. Ha a `soros` változóba – most a példa kedvéért – az „ans” karakterláncot akarjuk elmenteni, akkor ehhez az `ans` karakterláncot ’ ’ (aposztrófok) közé kell tennünk.

```
>> soros='ans'

soros =

ans
```

**2-5. ábra. Karakterlánc értékadása**

## 2.2 Az „m” fájlok használata

Az „m” fájl egy egyszerű parancssorozat, vagyis szkript, a parancssorba (Command Window) írt utasítások elmentett sorozata. Ezeket az „m” fájlokat később egy parancs használatával tudunk előhívni. A jegyzetben bemutatott gyakorlati példák egy részét is „m” fájlban valósítottuk meg.

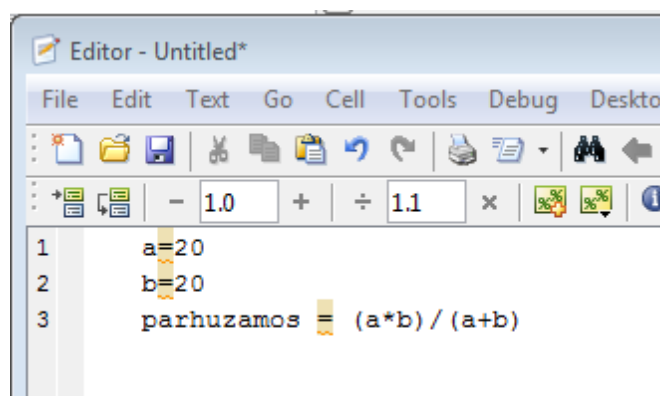
Az „m” fájlokat felhasználhatjuk például egy kívánt ábra előhívásához, vagy hosszabb számítási sor eltárolására. Nézzünk ez utóbbira egy példát!

### 2.2.1 Parancssori műveletekből „m” fájl készítése

Eddigi műveleteinket a Command Window-ban végeztük. Gyűjtsük össze az elvégzett műveleteket egy szkriptbe! A CTRL-N, vagy a File → Script lépéssorral tudunk egy Editor ablakot nyitni, amelyben megszerkeszthetjük parancsainkat.

Gyűjtsük össze az parancsokat! Megtehetjük ezt a History ablakból vagy a Command Window-ból is kimásolva, vagy akár beírhatjuk kézzel is.

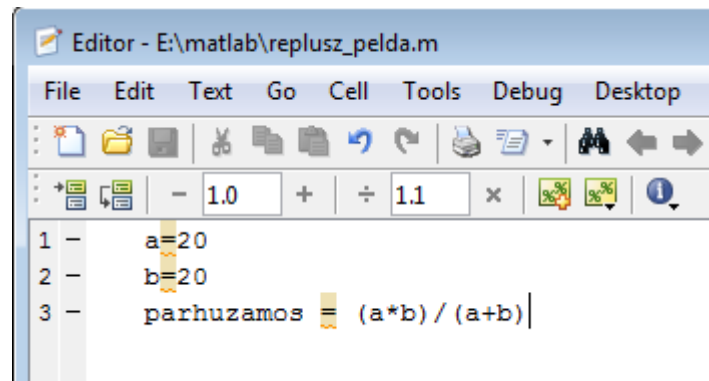
A parancsfájlunk tartalma ezután a következő:



2-6. ábra. Editor ablak mentés előtt

A parancsfájl futtatásához, a jelenleg Untitled nevű utasítássorunkat, először el kell menteni. (File → Save). A mentésnél figyeljünk arra, hogy amennyiben azonnal használni szeretnénk a fájlunkat, akkor az aktuális Current Folder mappába mentsük, vagy mentés után mozgassuk át oda, hiszen a MatLab csak így fogja a neve alapján megtalálni a szkriptet. Mentés után az ablak tetején megváltozik a név és eltűnik a változtatást jelző \* is a név mellől.





2-7. ábra. Editor ablak mentés után

M-fájlt egy egyszerű szövegszerkesztőben is írhatunk, csupán arra kell figyelni, hogy a fájl kiterjesztése végül „m” legyen.

### 2.2.2 „m” fájl futtatása

Futtathatjuk rögtön a Debug → Run (illetve a kód változtatása esetén Save and Run) művelettel, az F5 gyorsbillentyűvel vagy értelemszerűen a parancssorban a fájl nevének begépelésével (elég az „m” kiterjesztés nélkül).

```
>> replusz_pelda

a =

    20

b =

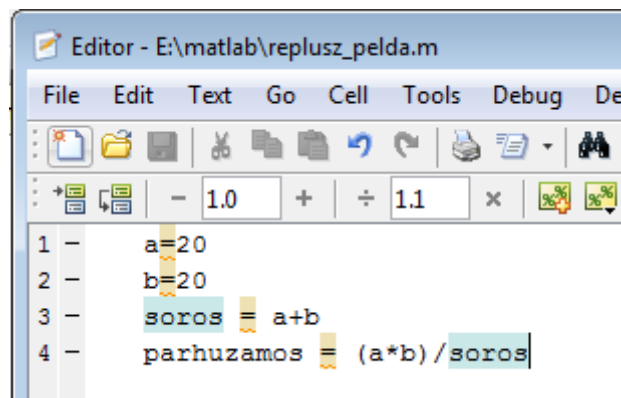
    20

parhuzamos =

    10
```

2-8. ábra. Szkript futtatása

Felmerülhet bennünk, hogy a replusz művelet számításakor tulajdonképpen az ellenállások soros eredőjét is meghatározzuk, hasznos lehet tehát ennek a kiírása is. Módosítsuk a kódot úgy, hogy felhasználjuk a részeredményt is!



2-9. ábra. Replusz és soros

Futassuk újra a szkriptet!

```
>> replusz_pelda

a =

    20

b =

    20

soros =

    40

parhuzamos =

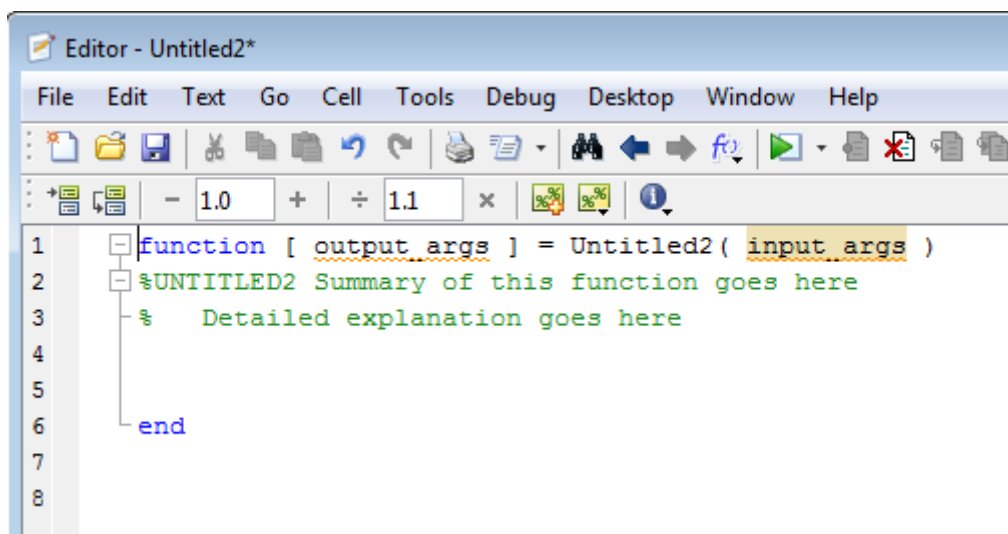
    10
```

2-10. ábra. Párhuzamos és soros értékek számítása

A szkript tehát azonos értékekkel lefutott. Ha másik ellenállásértékre szeretnénk futtatni, akkor azt az Editorban kell átírnunk és újrafuttatnunk. Ez nem túl kényelmes módja az új számolásnak – főleg hosszabb műveletsoroknál – kényelmesebb lenne, ha már a parancssor meghívásakor tudnánk megmondani, hogy milyen értékeket használjon a számoláshoz. Ehhez azonban szkriptünket függvényé kell alakítani, hiszen így tudunk bemeneti adatokat adni a műveletekhez. Az új „m” fájl típus, amivel meg kell ismerkednünk ehhez a függvény, vagyis `function`.

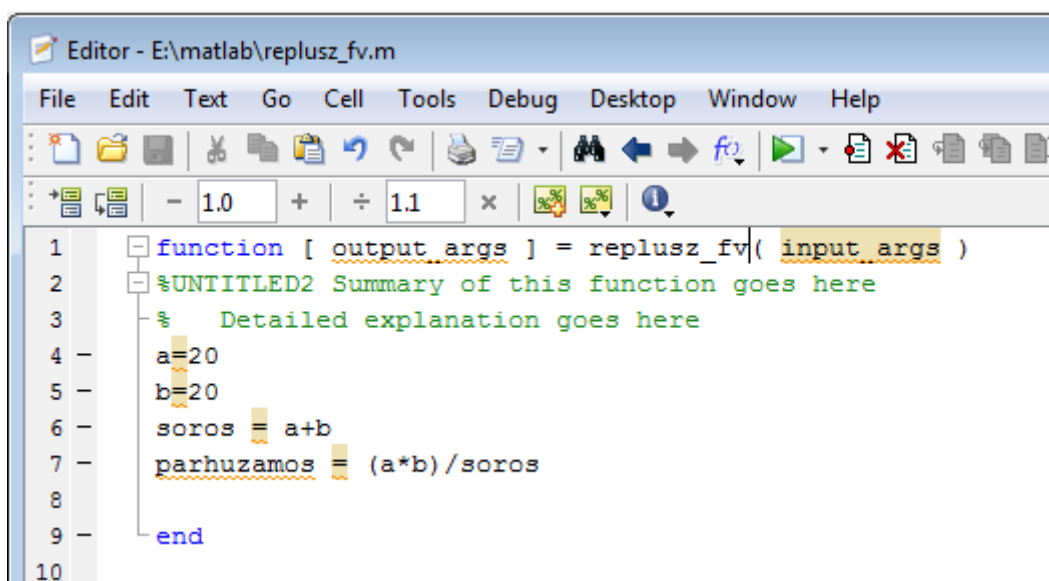
## 2.3 Függvények használata

Új függvényt a File → New → Function műveletsorral tudunk nyitni.



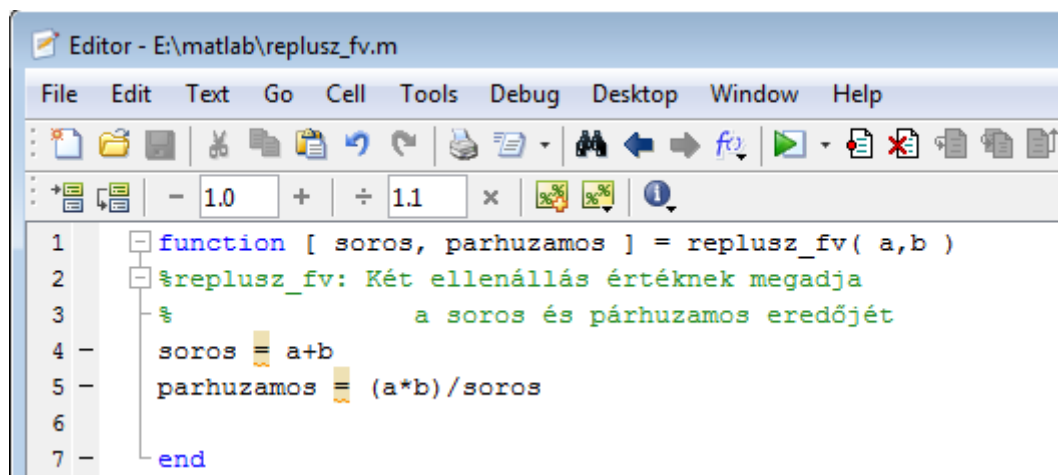
2-11. ábra. Function üresen

Ez egy alapértelmezetten strukturált szerkezetet ad meg nekünk, amit egyébként egy szkriptből is ki tudunk alakítani, ha úgy döntünk, hogy „függvényesítjük”. A `function` jelöli, hogy függvény típusú „m” fájlt írunk. Ez után a `[]` között a függvény kimeneti értékeit kell megadnunk, ha vannak. Alapértelmezetten `Untitled` a neve a függvénynek ez után `()`-ben kell megadni a bemeneti paramétereket. A komment rész után kell elhelyezni a függvény magját, ami akár lehet az előző szkriptünk is. Első lépésként ezt tegyük.



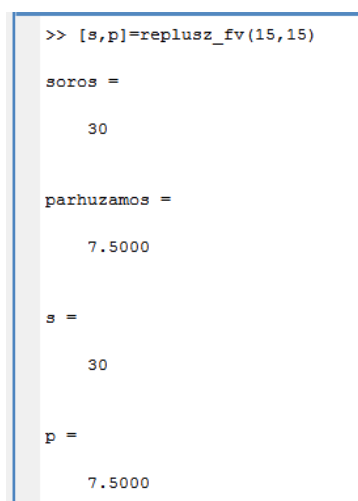
2-12. ábra. Function-ba illesztett szkript

A 4. és 5. sorban lévő értékadás természetesen felesleges, hiszen a függvény meghívásakor akarjuk majd megadni ezeket az értékeket. Ezeket tehát írjuk át az `input_args` helyére. Ezenkívül jelöljük ki a kimeneti változókat is, amelyek természetesen a soros és a párhuzamos változónevek lesznek.



2-13. ábra. Function módosítva, futtatásra készen

Az elkészült kód már kész a futtatásra. Meghívásához a Command Windowban a fájlnev után kerek zárójelben kell megadnunk a bemeneti paramétereket.



2-14. ábra. Replusz függvény futtatása

Látható, hogy futtatás után a két kért összeg kiírásra került, és a megadott s és p változóba is elmentésre kerültek. A függvény újabb meghívásakor viszont ezek a változók felülíródnak!

A bemeneti változókat természetesen megadhatjuk más változók értékeivel is, amelyeket a függvény meghívása előtt számolunk ki.

```

>> c=24+32
c =
    56
>> d=40+51
d =
    91
>> [s,p]=replusz_fv(c,d)
soros =
    147
parhuzamos =
    34.6667
s =
    147
p =
    34.6667

```

**2-15. ábra. Függvény hívás változókkal**

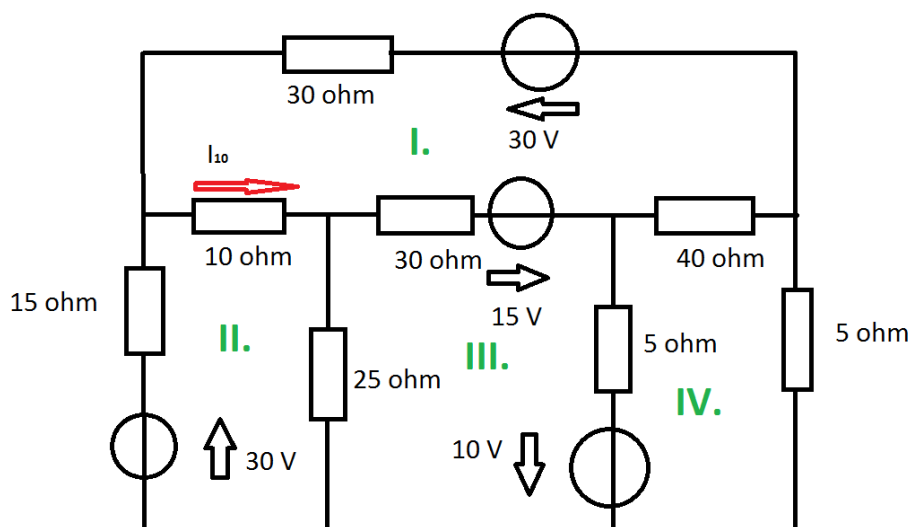
Az így futtatott függvény  $c$  és  $d$  változókból számolta értékeinket.

Zárásképpen foglaljuk össze a szkript és a függvény közti különbségeket:

- A szkript eltárolt utasítás sor, meghatározott értékekkel fut le. Ezeket szerkesztéssel tudjuk változtatni.
- A function függvény, tehát a meghívásakor megadott változókat felhasználva hajtja végre a parancsokat, magja lehet egy előzetesen megírt szkript is.

## 2.4 Számítási példa mátrixokkal

A mátrixokkal való számításokat egy villamosságtan példán keresztül szemléltetjük. Keressük meg az alábbi kapcsolásban a jelzett  $I_{10}$  áramot!



2-16. ábra. Hurok áramok meghatározása.

A római számmal jelzett hurok hurokárait az óramutató járásával megegyező irányban értelmezve  $I_{10} = II. - I.$  és a kapcsolásra az alábbi egyenletrendszer írható fel:

$$\begin{aligned}
 I.* (110) - II.* (10) - III.* (30) - IV.* (40) - 30 &= 0 \\
 -I.* (10) + II.* (50) - III.* (25) - IV.* (0) + 30 &= 0 \\
 -I.* (30) - II.* (25) + III.* (60) - IV.* (5) + 25 &= 0 \\
 -I.* (40) - II.* (0) - III.* (5) + IV.* (50) - 10 &= 0
 \end{aligned}$$

Az egyenletrendszert mátrixformában így írhatjuk fel:

$$\begin{bmatrix} 110 & -10 & -30 & -40 \\ -10 & 50 & -25 & 0 \\ -30 & -25 & 60 & -5 \\ -40 & 0 & -5 & 50 \end{bmatrix} * \begin{bmatrix} I. \\ II. \\ III. \\ IV. \end{bmatrix} = \begin{bmatrix} 30 \\ -30 \\ -25 \\ 10 \end{bmatrix}$$

Cramer szabállyal történő megoldáshoz szükségünk lesz az együtthatók 4x4-es mátrixára (Együtthato), továbbá annak az első és a második oszlopának módosításával előállított mátrixokra:

$$\text{Első oszlop módosítva} = \begin{vmatrix} 30 & -10 & -30 & -40 \\ -30 & 50 & -25 & 0 \\ -25 & -25 & 60 & -5 \\ 10 & 0 & -5 & 50 \end{vmatrix},$$

$$\text{Második oszlop módosítva} = \begin{vmatrix} 110 & 30 & -30 & -40 \\ -10 & -30 & -25 & 0 \\ -30 & -25 & 60 & -5 \\ -40 & 10 & -5 & 50 \end{vmatrix}$$

### 2.4.1 Mátrixok deklarálása

A mátrixokat szakirodalomban nagybetűvel jelöljük, de most akár használhatunk beszédes mátrix neveket is. Az elemeket [ ] közé írjuk, sorok elemeit vesszővel, a sorokat pontosvesszővel határoljuk el.

```
>> Egyutthato=[110,-10,-30, -40;-10,50,-25,0;-30,-25,60,-5;-40,0,-5,50]

Egyutthato =

    110    -10    -30    -40
    -10     50    -25     0
    -30    -25     60    -5
    -40     0     -5     50
```

2-17. ábra. Együttható mátrix bevitele a parancssorban

Javasolt, hogy a parancs végi pontosvessző karaktert hagyjuk el, hogy tudjuk ellenőrizni a bevitt adatok helyességét. Ellenőrzésként nézzük meg, hogy a mátrix főátlójában valóban csak pozitív értékek szerepelnek-e!

```
>> Atlo=diag(Egyutthato)

Atlo =

    110
     50
     60
     50
```

2-18. ábra. Főátló értékei

Készítsük elő a Cramer-szabály alkalmazásához a két módosított mátrixot is.

```
>> Elsomatrix=[30,-10,-30, -40;-30,50,-25,0;-25,-25,60,-5;10,0,-5,50]

Elsomatrix =

     30    -10    -30    -40
    -30     50    -25     0
    -25    -25     60    -5
     10     0     -5     50
```

2-19. ábra. Első hurokáramhoz tartozó mátrix bevitele

```
>> Masodikmatrix=[110,30,-30, -40;-10,-30,-25,0;-30,-25,60,-5;-40,10,-5,50]

Masodikmatrix =

    110     30    -30    -40
    -10    -30    -25     0
    -30    -25     60    -5
    -40     10     -5    50
```

2-20. ábra. Második hurokáramhoz tartozó mátrix bevitele

A Cramer-szabály alapján a két determináns hányadosával megkapjuk a hurokáramokat.

```
>> FOdeterminans = det(Egyutthato)

FOdeterminans =

    5.1275e+006
```

2-21. ábra. Együttható oldal determináns meghatározása

A determinánsokat a `det` paranccsal számoltathatjuk ki.

```
>> elsodeterminans = det(Elsomatrix)

elsodeterminans =

   -4.2500e+004
```

2-22. ábra. Első hurokáram mátrix determinánsának meghatározása

```
>> masodikdeterminans = det(Masodikmatrix)

masodikdeterminans =

   -5230000
```

2-23. ábra. Második hurokáram determinánsának meghatározása

A determinánsok segítségével meghatározzuk a hurokáramokat és kiszámoljuk a keresett áramot.



```
>> ElsoHurok=elsodeterminans/FOdeterminans

ElsoHurok =

    -0.0083

>> MasodikHurok=masodikdeterminans/FOdeterminans

MasodikHurok =

    -1.0200

>> Ag_aram=MasodikHurok-ElsoHurok

Ag_aram =

    -1.0117
```

2-24. ábra. Hurokáramok és a keresett ágáram meghatározása.

Megadhatjuk pusztán a mátrixművelet felhasználásával is az eredményt, de ilyenkor nem kapjuk meg a részeredményeket.

```
>> Ag_aram_v2= (det(Masodikmatrix)-det(Elsomatrix))/det(Egyutthato)

Ag_aram_v2 =

    -1.0117
```

2-25. ábra. Keresett ágáram mátrixműveletekkel

A keresett ágáramunk tehát 1,0117 A, viszont ellentétes irányú a feltételezett áramiránnyal, de kérdés, hogy jó-e az eredmény. Az ellenőrzéshez felhasználhatjuk a másik totális hálózatanalízis módszert a csomóponti potenciálok módszerét, de ezt a tisztelt hallgatóra bízunk. Most egy elegánsabb megoldási módszerrel ellenőrizzük a számolást.

Ehhez először hozzuk létre az eredmény oldali mátrixot is.

```
>> Eredmeny = [30,-30,-25,10] '

Eredmeny =

     30
    -30
    -25
     10
```

2-26. ábra. Eredmény oldali mátrix bevitele

A mátrix elemeit jelző [] után látható felső egyszeres aposztróf transzponálást jelent, hogy a vektorunk oszlopvektorként kerüljön eltárolásra.

Most már jöhet hurok áramok közvetlen gyors kiszámítása.

```
>> Hurok_aramok=inv(Egyutthato)*Eredmeny

Hurok_aramok =

    -0.0083
    -1.0200
    -0.8367
     0.1097
```

2-27. ábra. Hurokáramok meghatározása inverz mátrixszal

Ezzel a sorral egy lépésben az összes hurokáramot meg tudtuk határozni. Látható, hogy a válasz egy oszlopvektor, mely sorrendben tartalmazza a hurokáramok értékeit. Emlékezve az előző számolásra láthatjuk, hogy az első és a második hurokáramot jól határoztuk meg korábban. Vélhetően jó lesz a keresett ágáramunk is. Kiszámolhatjuk kimásolva is az értékeket, de helyette tanuljuk meg, hogyan tudunk hivatkozni egy mátrix elemeire.

#### 2.4.2 Hivatkozás egy mátrix elemeire

```
>> Ag_aram_v3=Hurok_aramok(2,1)- Hurok_aramok(1,1)

Ag_aram_v3 =

    -1.0117
```

2-28. ábra. Ágáram meghatározása mátrix elemeire hivatkozással

Egy mátrix elemeire az alábbi szintaxissal tudunk hivatkozni:

MÁTRIX(keresett elem sora, keresett elem oszlopa)

Ezt felhasználhatjuk értékadásra is. Ha elhibázzuk egy mátrix bevitelét, de csak egyetlen elemet akarunk megváltoztatni, akkor azt megtehetjük ugyanezzel a művelettel.

```

>> Proba=Hurok_aramok

Proba =

    -0.0083
    -1.0200
    -0.8367
     0.1097

>> Proba(4,1)=0

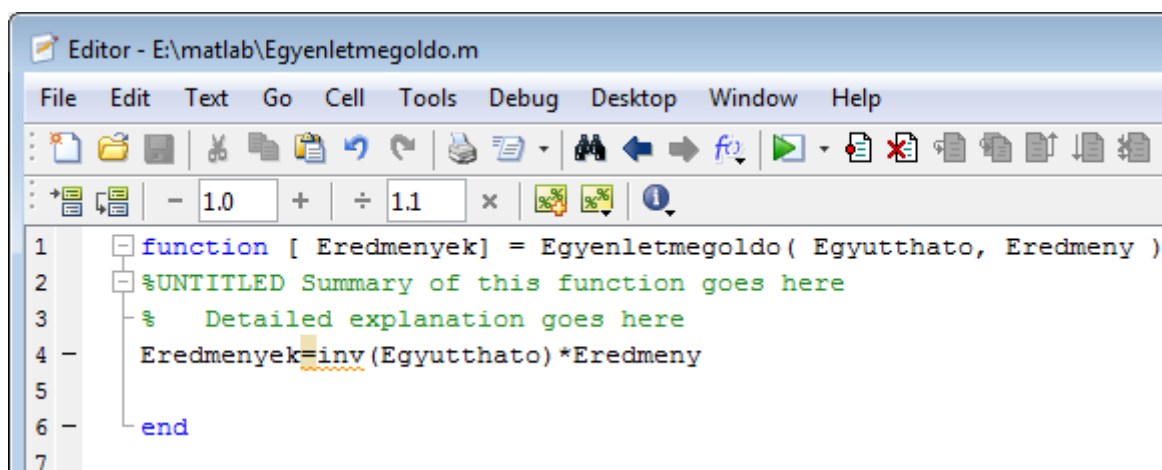
Proba =

    -0.0083
    -1.0200
    -0.8367
         0

```

2-29. ábra. Mátrix módosítása

Az ágáramok meghatározására használt ellenőrző művelet, melyben inverz mátrixszal számoltunk alkalmas lehet a megoldási művelet függvénné alakítására is, hiszen a bevitt mátrixokból közvetlenül tudtuk számolni a keresett hurokáramokat. Vagy akár a csomóponti potenciálokat, vagy bármilyen lineáris egyenletrendszer megoldását.



2-30. ábra. Egyenletmegoldó függvény

Futtatása pedig az alábbi eredményt adja:

```
>> Egyenletmegoldo(Egyutthato, Eredmeny);

Eredmenyek =

    -0.0083
    -1.0200
    -0.8367
     0.1097
```

2-31. ábra. Egyenletmegoldó függvény futtatása

### 2.4.3 Példákban használt parancsok listája

```
pelda = 2
pelda = 2;
d=3
D=4
a=20
b=20
parhuzamos = (a*b)/(a+b)
(a*b)/(a+b)
a+b
soros = ans
soros='ans'
replusz(15,15)
replusz_fv(10,10)
c=24+32
d=40+51
c=24+32
d=40+51
replusz(c,d);
Egyutthato=[110,-10,-30, -40;-10,50,-25,0;-30,-25,60,-5;-40,0,-5,50]
Atlo=diag(Egyutthato)
Elsomatrix=[30,-10,-30, -40;-30,50,-25,0;-25,-25,60,-5;10,0,-5,50]
Masodikmatrix=[110,30,-30, -40;-10,-30,-25,0;-30,-25,60,-5;-40,10,-5,50]
FOdeterminans = det(Egyutthato)
elsodeterminans = det(Elsomatrix)
masodikdeterminans = det(Masodikmatrix)
ElsoHurok=elsodeterminans/FOdeterminans
MasodikHurok=masodikdeterminans/FOdeterminans
Ag_aram=MasodikHurok-ElsoHurok
Ag_aram_v=(masodikdeterminans-elsodeterminans)/FOdeterminans
Ag_aram_v2=(det(Masodikmatrix)-det(Elsomatrix))/det(Egyutthato)
Eredmeny = [30,-30,-25,10]'
Hurok_aramok=inv(Egyutthato)*Eredmeny
Ag_aram_v3=Hurok_aramok(2,1)- Hurok_aramok(1,1)
Proba=Ag_aramok
Proba(4,1)=0
Egyenletmegoldo(Egyutthato, Eredmeny);
```

### 3 Analóg és digitális modulációk

Modulációs eljárást a híradástechnika számos területén alkalmazunk akkor, ha a rendelkezésünkre álló átviteli közegen az átvinni kívánt információt hordozó jel nem képes, vagy csak nagy csillapítással, torzulással tudja eljuttatni az egyik ponttól a másikig. Ekkor választunk egy olyan jelet (vivőjelet), amelyet a rendelkezésre álló átviteli közeg az elvárásainknak megfelelően képes továbbítani. Az információ továbbításához a vivő jel valamely jellemzőjét (esetleg jellemzőit) az információt hordozó jellel (moduláló jellel) megváltoztatjuk. A fenti eljárást modulációnak nevezzük. A modulált jelet (vivőjel és moduláló jel segítségével előállított jel) eljuttatjuk az átviteli közegen keresztül a célpontig, majd leválasztjuk arról az információt hordozó jelet. A leválasztást nevezzük demodulációnak.

#### 3.1 Modulációs eljárások

A modulációs eljárásokat megkülönböztetjük aszerint, hogy a vivő jel mely jellemzője fogja az információt eljuttatni egyik ponttól a másikig. Ezek alapján a következő analóg és digitális modulációs eljárásokat különböztetjük meg:

- Amplitúdó modulációs eljárások
- Szög modulációs eljárások
- Kevert modulációs eljárások

#### 3.2 Analóg modulációs eljárások áttekintése

Analóg modulációs eljárásoknak nevezzük mindazon modulációs módszereket, melyekben az alkalmazott vivő- és moduláló jel értékkészlete és értelmezési tartománya folytonos, vagyis a vivő és a moduláló jel analóg.

##### 3.2.1 Amplitúdó moduláció

Abban az esetben, ha az információ hordozó paraméter a vivő jel amplitúdója, akkor a modulációs eljárást amplitúdó modulációnak (röviden AM-nek) nevezzük.

Amplitúdó moduláció meghatározása matematikailag, ha a vivő jel:

$$u_v(t) = U_v \cdot \sin(2\pi f_v t),$$

és a moduláló jel:  $u_m(t)$ , akkor a modulált jel:

$$u_{md}(t) = (U_v + U_v \cdot k \cdot u_m(t) / u_{mmax}) \cdot \sin(2\pi f_v t),$$

ahol  $k$  –modulációs mélység,  $u_{\text{mmax}}$  – moduláló jel amplitúdójának maximuma,  $U_v$  – a vivő amplitúdója,  $f_v$  – a vivő frekvenciája

A gyakorlatban az amplitúdó modulált jelet (mint modulációs terméket) gyakran szűrésnek vetjük alá (frekvencia spektrum bizonyos elemeit, esetleg azok bizonyos részeit elnyomva). A jellemző spektrumkép alapján a megszürt AM jeleket a következő elnevezésekkel láthatjuk el:

- Dupla oldalsávós AM jel: AM-DSB (DSB = Double Side Band)
- Elnyomott vivőjű dupla oldalsávós AM jel: AM-DSB/SC (SC = Supressed Carrier)
- Dupla oldalsávós AM jel, átvitt vivővel: AM-DSB/TC (TC = Transmitted Carrier)
- Egyoldalsávós AM jel: AM-SSB (SSB = Single Side Band)
- Elnyomott vivőjű csonka oldalsávós AM jel: AM-SSB/SC

Technikai kényszerűségi és spektrumgazdálkodási ok miatt került bevezetésre. Használata abban az esetben indokolt, ha a vivő komponens és az oldalsávi komponens alsó határfrekvenciás összetevője túl közel van egymáshoz, így az szűréssel nem választható le (végtelen meredekségű szűrőre lenne szükség, ami technikailag kivitelezhetetlen). Ekkor az elnyomott oldalsáv vivőhöz közeli komponenseit fokozatosan csillapítják, majd nyomják el, valamint arányosan csillapítják a megmaradó oldalsáv vivőhöz közeli komponenseit is.

Az eljárást AM-VSB (VSB = Vestigial Side Band) –nek nevezik és az analóg TV technikában alkalmazzák.

Gyakran jelölik azt is, hogy mely oldalsáv van jelen a frekvenciaspektrumban:

- USB = Upper Side Band illetve
- LSB = Lower Side Band)

### 3.2.2 Szögmodulációk (PM és FM)

Szögmodulációnak nevezzük azon modulációs eljárásokat, mely során a vivőjelet leíró függvény argumentumát (szögét) módosítja a moduláló jel. A szögmoduláció lehet frekvencia moduláció, ekkor közvetlenül a vivő jel frekvenciáját módosítja a moduláló jel (FM) és lehet fázis moduláció, ekkor pedig a vivő jel fázisszögét módosítja a moduláló jel (PM).

### 3.2.3 Analóg modulációk kezelése a MatLab-ban

A modulációs módszerek és eljárások megismerésére napjainkban sok módszer nyújt lehetőséget. Modulált jelet előállíthatunk laboratóriumi jelgenerátorokkal (szignál-

generátorokkal) és vizsgálhatjuk azokat oszcilloszkóppal az időtartományban, valamint spektrum-analizátorral a frekvenciatartományban.

A digitális számítógépek lehetőséget biztosítanak a szimulációs vizsgálatokra. Szimulációs módszerrel hatékonyan, gyorsan ismerhetjük meg a modulációs eljárásokat és azok tulajdonságait. Ebben az esetben nélkülözhetetlen a vizsgálni kívánt modulációs eljárás matematikai leírásának az ismerete.

Az analóg modulációs eljárások vizsgálatát a MatLab felhasználásával elvégezhetjük a „modulate” és a „demod” függvények segítségével. A fenti két függvény, a MatLab Signal Processing Toolbox függvénye.

A „modulate” függvény paraméterezése a következő:

```
>>Y=modulate(X,fc,fs,METHOD,OPT);
```

ahol X – a moduláló jel, fc – a vivő jel frekvenciája és fs – a mintavételi frekvencia. (Nem szabad elfelejtenünk, hogy bár analóg jelet reprezentálunk de digitális rendszerrel dolgozunk, ezért az analóg jelek vizsgálatánál is csak azok mintái állnak rendelkezésre!)  $fs > 2*(fc+BW)$ -nek a mintavételi törvény értelmében teljesülni kell, ahol a BW a modulált jel sávszélességét jelenti. METHOD – a modulációs módszer ('am', 'amdsb-sc', 'amdsb-tc', 'amssb', 'fm', 'pm', 'qam', stb.). OPT – opcionális paraméterek egyes modulációs eljárások esetén.

A modulációk elemzéséhez feltétlen szükség van a jelek spektrumának (frekvencia tartományi) megjelenítésére is. A MatLab-bal gyors Fourier transzformációt tudunk elvégezni az fft függvény segítségével. Az alábbi, drawfft nevű, függvény előállítja a vizsgálatokhoz szükséges, megfelelő paraméterekkel rendelkező, spektrumot, mely a kiszámított komplex spektrumnak csak a pozitív frekvenciatartományát, és annak abszolút értékét (azaz a jelek amplitúdóját) ábrázolja.

A drawfft függvény használatához két paraméter megadása szükséges, a vizsgált jel sorvektora, és a mintavételi frekvencia. Pl.:

```
drawfft(x1,fs);
```

### A drawfft függvény kódja:

```
function drawfft(x,fs)
N=length(x);
sp1=(fft(x,N))/N; % Komplex spektrum számítása
sp3=sp1(1:N/2); % Csak a pozitív frekvenciák
sp3(2:N/2)=sp3(2:N/2)*2; % Valós amplitúdók
df=fs/N; % FFT felbontása
ff2=0:df:(fs/2)-df; % Frekvencia skála számítása
```

```
figure; % új ablak nyitása
stem(ff2,abs(sp3),'.'); % spektrum vonalak
axis([0,3e4,0,5]); % skálázás
grid;
xlabel('frekvencia [Hz]');
ylabel('Amplitúdó');
title('Spektrum (abszolút érték)');
```

### 3.3 Feladatok

A modulációk vizsgálatához, mivel a MatLab csak mintavételezett számmintákon képes dolgozni, meg kell adnunk az analóg jelek feldolgozásához a mintavétellel kapcsolatos paramétereket:

1. *Adjuk ki a következő parancsokat!*

```
N=10000;
tw=5e-3;
fs=N/tw;
td=tw/N;
t=0:td:tw-t; %időskála
```

Az egyes változók jelentései a következők:

- N minták száma
- tw időablak [s], azaz hány másodperc hosszú legyen a mintánk
- fs mintavételi frekvencia [Hz]
- td mintavételi időköz

Mint látható az első két paraméter megadásából az fs és a td is kiszámítható.

2. *Jegyezzük le a fenti paraméterek értékeit, és az összefüggéseket is!*
3. *Mekkora a mintavételezési frekvencia? Legfeljebb mekkora frekvenciájú jeleket vizsgálhatunk így?*
4. *Ellenőrizzük, hogy az időtengelyt reprezentáló t sorvektor értékei valóban 0-tól td lépközzel növekednek (Workspace ablakban a változó nevén duplán kattintsunk az értékek megjelenítéséhez)?*
5. *Hány elemből áll a t vektor?*

Az alábbi táblázat szerint fogunk előállítani két, eltérő fázisú és frekvenciájú periodikus jelet, amiket a későbbiekben, mint moduláló jelet használunk.



**3-1. táblázat. Moduláló periodikus jelek paraméterei**

változónév	amplitúdó	frekvencia	kezdőfázis
x1	2	1000Hz	0
x2	1	1600Hz	$-\pi/4$

6. Adjuk ki az alábbi parancsokat az x1 létrehozatalához!

```
x1=2*cos(2*pi*1000*t+0);
```

7. Hozzuk létre hasonlóan az x2 jelet is!

8. Hozzunk létre egy x változót, mely az x1 és az x2 összege

9. Jelenítsük meg a létrehozott jeleket (x1, x2, x) időtartományban (plot)!

A frekvenciatartománybeli ábrákon a Data Cursor segítségével tudjuk gyorsan és pontosan leolvasni a komponensek értékeit. Több spektrumvonal esetén az Alt billentyű lenyomva tartása mellett lehet a spektrum vonalakra elhelyezni több szövegdobozt.

10. Jelenítsük meg az x1, az x2, majd az x jel spektrumát a drawfft függvény használatával!  
Pl.:

```
drawfft(x1,fs)
```

11. Jelöljük be és olvassuk le a megjelenített spektrum ábrákon az egyes komponensek értékeit a Data Cursor segítségével!

A moduláló jel előállítása és vizsgálata után végezzük el a modulációt az alábbi lépésekben. Először adjuk meg a vivő jel frekvenciáját (fc), amit most 20000 Hz-re választunk. (Bár a gyakorlatban a vivő jel frekvenciája ennél jóval magasabb értékű, esetünkben az ábrázolás skálázásának egyszerűsítése végett választottuk ezt az alacsony értéket.)

12. Hozzunk létre egy fc=20000 változót!

13. Moduláljuk az fc vivőt az x1 moduláló jel használatával az alábbi parancs segítségével!

```
yx1amdsbtc=modulate(x1,fc,fs,'amdsb-tc');
```

14. Ábrázoljuk a létrehozott jelet az idő- és a frekvencia függvényében, jegyezzük le, hogy milyen komponensek milyen frekvenciákon jelentek meg!

```
plot(t, yx1amdsbtc)
drawfft(yx1amdsbtc, fs)
```

15. *Demoduláljuk jeliünket (yx1amdsbtc) a demod függvénnyel!*

```
x1amdsbtc=demod(yx1amdsbtc, fc, fs, 'amdsb-tc');
```

16. *Jelenítsük meg idő- és frekvencia tartományban a jelet! Ellenőrizzük, hogy a demodulált alapsávi jel megegyezik-e az eredetivel! Az eltéréseket jegyezze le (pl. DC eltolás, eltérő amplitúdó)!*

17. *Moduláljunk az x1 jellel más vivő frekvenciával is (a frekvencia változtatásakor ügyeljen a mintavételi törvény betartására!)*

```
fc1=50000; %[Hz]
yx1amdsbtc2=modulate(x1, fc1, fs, 'amdsb-tc');
```

18. *Ábrázoljuk a létrehozott jelet az idő- és a frekvencia függvényében, jegyezzük le, hogy milyen komponensek milyen frekvenciákon jelentek meg!*

19. *Végezzük el a fenti modulációt és a demodulációt az x2 jelre is!*

```
yxamdsbtc=modulate(x2, fc, fs, 'amdsb-tc');
```

20. *Ábrázoljuk a létrehozott modulált és alapsávi jelet az idő- és a frekvencia függvényében, jegyezzük le, hogy milyen komponensek milyen frekvenciákon jelentek meg!*

A MatLab Súgójában, vagy a parancsablakba beírva a help modulate parancsot, megtalálhatjuk, hogy milyen AM típusokat ismer a modulate és a demod függvény: AMDSB-TC, AMDSB-SC, AMSSB.

21. *Végezzük el az x2 jellel a modulációt és a demodulációt az eddig nem használt AM típusokkal, ügyeljünk a megkülönböztethető változónevekre!*

22. *Az időtartományi és a spektrumábrák alapján jegyezzük le az egyes moduláció típusok közötti különbségeket!*

A következőkben vizsgáljuk meg a szögmodulációkat (fázis és frekvenciamoduláció – FM, PM)!

23. *Hozzuk létre az x1 moduláló jel használatával az FM és a PM modulált jeleket!*

```
y1fm=modulate(x1,fc,fs,'fm');
y1pm=modulate(x1,fc,fs,'pm');
```

#### 24. *Ábrázoljuk és jellemezzük a modulált jeleket időtartományban!*

Az analóg kvadratúra amplitúdó modulációval (QAM) két egymástól független jelet tudunk egy vivőn továbbítani. Használjuk ehhez az x1, x2 jeleket!

#### 25. *Hozzuk létre az analóg QAM jelet!*

```
yqam=modulate(x1,fc,fs,'qam',x2);
```

#### 26. *Vizsgáljuk meg a modulált jelet idő- és frekvenciatartományban egyaránt!*

#### 27. *Demoduláljuk a QAM modulált jelet!*

```
[x1dem, x2dem]=demod(yqam,fc,fs,'qam');
```

#### 28. *Ellenőrizzük, hogy a két jel az eredeti alapsávi jelekkel megegyezik-e (időtartományban és frekvenciatartományban)!*

### 3.4 Digitális modulációs eljárások áttekintése

Digitális modulációs eljárásoknak, vagy billentyűzésnek, nevezzük mindazon modulációs módszereket, melyekben az alkalmazott moduláló jel értékkészlete és értelmezési tartománya diszkrét, azaz digitális, a vivő jel pedig szinuszos.

#### 3.4.1 Amplitúdó billentyűzés ASK (Amplitude Shift Keying)

Amplitúdó billentyűzés esetén a vivő jel szinuszos, a moduláló jel pedig digitális (értékkészlete '0' vagy '1'). A moduláló jel jelen esetben a vivő jel amplitúdóját változtatja ('kapcsolgatja'). Az így előállított jel (modulált jel) teljesítményszintje folyamatosan ingadozó, mivel a logikai '0'-hoz A0, a logikai '1'-hez pedig A1 amplitúdó tartozik.

$$u_{ASK}(t)=A*\sin(2*\pi*f+\varphi), \quad (2.3)$$

ahol az A az ASK jel pillanatnyi amplitúdója (A0 vagy A1), az f a vivő jel frekvenciája, a  $\varphi$  pedig a vivőjel kezdőfázisa. A fenti modulációs eljárás során A0 vagy akár A1 (egyik a kettő közül) lehet nulla. Ekkor valójában a digitális moduláló jel nem tesz mást, mint a vivőjelet be-ki kapcsolgatja, attól függően, hogy '0' vagy '1' értékű bit kerül átvitelre.

Az ASK önmagában történő használata a távközlő rendszereknél nem elterjedt.

### 3.4.2 Frekvencia billentyűzés FSK (Frequency Shift Keying)

Frekvencia billentyűzés esetén a vivő jel szinuszos, a moduláló jel pedig digitális (értékkészlete '0' vagy '1'). A moduláló jel jelen esetben a vivő jel frekvenciáját ( $f_p$ ) változtatja, például a logikai '0'-hoz  $f_0$ , míg a logikai '1'-hez  $f_1$  tartozik.

$$u_{FSK}(t) = A \cdot \sin(2 \cdot \pi \cdot f_p \cdot t + \varphi), \quad (2.4)$$

ahol  $A$  az FSK jel amplitúdója,  $f_p$  a vivő jel pillanatnyi frekvenciája ( $f_0$  vagy  $f_1$ ),  $\varphi$  pedig a vivőjel kezdőfázisa.

Nézzük meg példaként a V.21 ajánlás szerinti FSK jellemzőket!

- 300 bit/s sebességű duplex átvitelt tesz lehetővé a beszédcélú távközlő hálózaton.
- Az „adás” és a „vétel” irányt a beszédsávban két csatorna kijelölésével oldják meg.
- A hívó berendezés szemszögéből az adási frekvenciák (1-es csatorna):
  - 980 Hz, ami logikai '1'-nek felel meg
  - 1180 Hz, ami pedig logikai '0'-t jelenti.
- A hívó berendezés szemszögéből a vételi frekvenciák (2-es csatorna):
  - 1650 Hz, ami logikai '1'-nek felel meg
  - 1850 Hz, ami pedig logikai '0'-t jelenti.

Az FSK általában a kis sebességű digitális átviteli rendszerek kapcsán elterjedt eljárás. A beszédcélú távközlő csatornán nagy megbízhatóságú (fizikai réteg szinten) átviteli eljárás.

### 3.4.3 Fázis billentyűzés PSK (Phase Shift Keying)

Fázis billentyűzés esetén a vivő jel szinuszos, a moduláló jel pedig digitális (értékkészlete '0' vagy '1'). A moduláló jel jelen esetben a vivő jel fázisát változtatja.

$$u_{PSK}(t) = A \cdot \sin(2 \cdot \pi \cdot f_p \cdot t + \varphi), \quad (2.5)$$

ahol az „ $A$ ” a PSK jel amplitúdója, az  $f$  a vivő jel frekvenciája, a  $\varphi$  pedig a vivőjel pillanatnyi fázisa ( $\varphi_0$  vagy  $\varphi_1$ ).

A PSK modulációs eljárást tovább csoportosítva, megkülönböztetünk úgynevezett állapot (PSK) és differenciális modulációt (DPSK).

- Állapot moduláció (PSK) esetén a kezdő fázis értékhez viszonyítottan változtatjuk a vivőjel fázisát. Ebben az esetben úgy teszünk, mintha egy fix fázisú referencia jelet

vennénk viszonyításnak. Pl. 1-es értéknek a 180 fok fázis felel meg, míg 0-ásnak a 0 fok. (BPSK)

- Differenciális modulációról beszélünk abban az esetben, ha az előző jelelemet vesszük referenciának és a következő jelelem fázisát ehhez képest határozzuk meg. Azaz, ha az átvinni szándékozott bit 1, akkor 180 fok fázist tolunk, ha 0 nem változtatjuk a fázist. (DBPSK)

A PSK moduláció során gyakran több fázisállapotot is kihasználunk, ezért a digitális jelfolyamban az egymás után következő biteket csoportosítva (2, 3, 4 .... x bit) – szimbólumokat képezve – végezzük a modulálást.

Két bit esetén (egy jelelem 2 bit információt hordoz) 4 állapot lehetséges, azaz négy fázisszöget kell definiálnunk (QPSK vagy 4PSK):

**3-2. táblázat. QPSK fázisok**

bitek	fázis / fázisváltozás
00	0°
01	90°
10	270°
11	180°

Három bit esetén (egy jelelem 3 bit információt hordoz) 8 állapot lehetséges, azaz nyolc fázisszöget kell definiálnunk.

**3-3. táblázat. 8PSK fázisok**

bitek	fázis / fázisváltozás
000	45°
001	0°
010	90°
011	135°
100	270°
101	315°
110	225°
111	180°

### 3.4.4 Digitális kevert modulációs eljárások

Mostanában különösen elterjedtek a kombinált modulációs eljárások. Az eljárás lényege az, hogy a vivő jel nemcsak egy, hanem több jellemzőjét is megváltoztatja a moduláló jel, illetve a digitális moduláló jelfolyamból csoportosított bitsorozat.

Abban az esetben, ha a vivő jel fázisát és amplitúdóját is megváltoztatjuk a modulációs eljárás esetén, akkor az eljárást kvadratúra amplitúdó modulációnak nevezzük. Ekkor a vivő amplitúdója és fázisa is információt hordozó fizikai jellemző. A QAM (Quadrature Amplitude Modulation) nagy adatátviteli sebesség elérését teszi lehetővé.

$$u_{QAM}(t) = A \cdot \sin(2 \cdot \pi \cdot f \cdot t + \varphi), \quad (2.6)$$

ahol „A” a QAM jel pillanatnyi amplitúdója ( $A_0, A_1 \dots A_n$ ),  $f$  a vivő jel frekvenciája,  $\varphi$  pedig a vivőjel pillanatnyi fázisa ( $\varphi_0$  vagy  $\varphi_1 \dots \varphi_n$ ).

### 3.4.5 Digitális modulációk kezelése MatLab-ban

A digitális modulációs eljárások vizsgálatához a MatLab a következő beépített függvényeket tartalmazza (Signal Processing Toolbox), melyek használata mellett frekvencia tartománybeli reprezentációval is szemléltethetőek a modulációk:

- fskmod,
- fskdemod,
- pskmod,
- pskdemod.

## 3.5 Feladatok

Vizsgáljuk meg a BPSK (2PSK) jellemzőit! Fázisbilleentyűzés esetén az egyes szimbólumoknak (biteknek) különböző fázishelyzeteket feleltetünk meg. BPSK esetén 2 féle fázishelyzet van: pl. 0 és  $\pi$  rad.

Digitális jelek átvitele esetén a modulált jelet nem időben szokás ábrázolni, hanem úgynevezett konstellációs ábrán. Ez egy komplex sík, melyben a pontok (komplex számok) abszolút értéke adja meg az adott szimbólumhoz tartozó vivő jel amplitúdó értékét (ez PSK esetén állandó, egyenlő pl. 1-gyel) és az argumentuma a fázis helyzetet.

1. *Definiáljuk a BPSK billeentyűzéshez tartozó fázisállapotokat a komplex síkon! Exponenciális alak használatával igen egyszerű egység hosszú, adott fázishelyzetű (0,  $\pi$  rad) pontokat megadni:*

```
konst_bpsk = exp(j*[0, pi]);
```

2. *Ábrázoljuk komplex síkon a pontokat! Írjuk mellé a pontokhoz rendelhető szimbólumok (itt most bitek) értékét!*

```
scatterplot(konst_bpsk,1,0,'o');
text(real(konst_bpsk),imag(konst_bpsk)+0.1,dec2bin([0 1]));
```

3. *Mekkorák az egyes szimbólumok közötti szöghelyeltérések.*
4. *Egy szimbólumidő alatt mennyi bitet lehet átvinni ezzel a billentyűzés típussal?*
5. *Állítsuk elő a QPSK (4PSK) konstellációs ábráját az előző példa alapján!*

```
konst_qpsk = exp(j*[0, pi/2, 3*pi/2, pi]);
scatterplot(konst_qpsk,1,0,'o');
text(real(konst_qpsk),imag(konst_qpsk)+0.1,dec2bin([0 1 2 3]));
```

6. *Mekkorák az egyes szimbólumok közötti szöghelyeltérések.*
7. *Egy szimbólumidő alatt mennyi bitet lehet átvinni ezzel a billentyűzés típussal?*
8. *Állítsuk elő a 8PSK konstellációs ábráját az előző példák alapján!*
9. *Mekkorák az egyes szimbólumok közötti szöghelyeltérések.*
10. *Egy szimbólumidő alatt mennyi bitet lehet átvinni ezzel a billentyűzés típussal?*

A következőkben nézzük meg mi történik, ha 8PSK-val véletlen generált szimbólumokat továbbítunk zajos csatornán! Ehhez az alábbi MatLab kódot futassuk, célszerű „m” szkriptet készíteni a kódból:

```
numphase=8; % Fázisok száma, nPSK
SNR = 25; % jel/zaj viszony (dB)
konst_psk = exp(j*[pi/numphase:2*pi/numphase:(2*numphase-1)*pi/numphase]).'; % psk állapotok a
komplex síkon
num=2000; % továbbítandó szimbólumok száma (min kb. 10)
x=randint(num,1,numphase)+1; % véletlen generált szimbólumsorozat (pl. 8 szintű: 1..8)
y = konst_psk(x); % psk billentyűzés
n = randn(size(y))+j*randn(size(y)); % Gauss
ny = y + n*std(y)/(std(n)*10^(SNR/20)); % zaj keverése a modulált jelhez SNR szerint
h1 = scatterplot(ny,1,0,'.'); % zajos modulált jel
hold on
scatterplot(konst_psk,1,0,'g*', h1); % eredeti fázisállapotok
axis([-2 2 -2 2]); grid;
title('Zajos csatornán továbbított jel konstellációs ábrája');
hold off;
```

11. *Értelmezzük a megjelenő konstellációs ábrát!*
12. *Futtassuk a kódot egyre romló csatorna viszonyokat (SNR = 25, 20, 15, 10) modellezve. Minden futtatás előtt módosítsuk az SNR változó értékét!*

13. *Mikor mondhatjuk, hogy hibátlan az átvitel?*
14. *Mekkora SNR értéknél van a hibátlan átvitel határa?*
15. *Változtassuk meg az állapotok számát 4-re, majd 8-ra és futassuk a kódot!*
16. *Az állapotok száma és a hibátlan átvitelhez szükséges minimális SNR hogyan függ össze?*

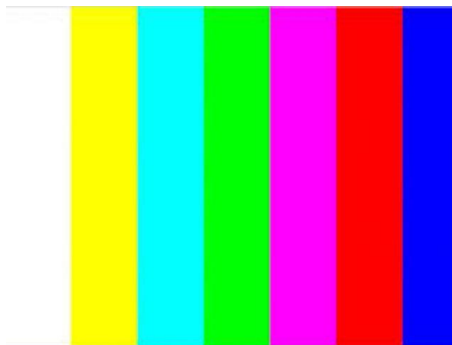


## 4 Képek kezelése a MatLab-ban

Egy digitális kép tökéletes példa egy olyan adathalmazra amelyet MatLab-ban érdemes és érdekes feldolgozni, mivel a képeket alkotó pixelek mátrixot alkotnak. A fekete-fehér vagy helyesebben fogalmazva szürkeárnyaltos képek világosságértékeket, a színes képek pedig a színösszetevők értékeit tartalmazzák egy 3 dimenziós NxM-es tömbben, vagyis mátrixban.

### 4.1 Színes képek

Vizsgálatunkban vegyünk egy egyszerű képet, mégpedig a videótechnikában használt színcsík ábrát. Ez tartalmazza az additív színkeverés alapszíneit (piros, zöld és kék) valamint a szubsztraktív színkeverés alapszíneit (cián, sárga és magenta) is. Az általunk érzékelt színek ezekből az alapszínekből keverhetők ki.



4-1. ábra. Színcsík: color\_bar2.jpg

A kép feldolgozásához először be kell importálnunk a képet a MatLab-ba. Ezt megtehetjük a Current Folder ablakban a fájlra kétszer klikkelve az egér bal gombjával, majd a felugró párbeszédablakot az OK megnyomásával bezárva. A műveletet a parancssorban is végrehajthatjuk:

```
>> color_bar2=imread('color_bar2.jpg');
```

A létrejött változó egy 480x640 méretű 3 dimenziós mátrix, vagyis egy tömb. A képünk eredetileg 640x480 felbontású, de a képek tárolására használt úgynevezett indexált tömbben a sorok és az oszlopok hivatkozási sorrendje felcserélésre került. Tehát egy NxM felbontású kép MxN méretű tömbben kerül tárolásra. Az egyes rétegek mátrixai egy adott színkomponens értékeit tartalmazzák. A színkomponensek attól függenek, hogy milyen színtérben határozzuk meg a színeket. A legelterjedtebb színterek az `rgb` és `yuv` a fotó- vagy videótechnikában, `CYMK` a nyomdatechnikában illetve a szürkeárnyalat vagyis `gray` színtér a szürkeárnyaltos képeknél.

Mostani képünk `rgb` színterű vagyis a tömb első rétege (mátrixa) az `r` – red, vagyis piros értéket, a második a `g` – green, vagyis zöld, illetve a harmadik a `b` – blue, vagyis kék értékeket tartalmazza. A rétegeket sokszor csatornáknak is nevezzük.

A Workspace ablakban megfigyelhetjük, hogy a mátrixok értékeit `uint8` formátumban tárolja a program. Ez unsigned, tehát előjel nélküli 8 bites integer értékeket jelent. A képeink színmélysége 8 bites, vagyis 256 árnyalatot különböztetünk meg benne.

Az előző fejezetben megtanult műveletek a tömbökre is vonatkoznak, tehát megtehetjük, hogy egy új tömböt egy előző segítségével töltünk fel.

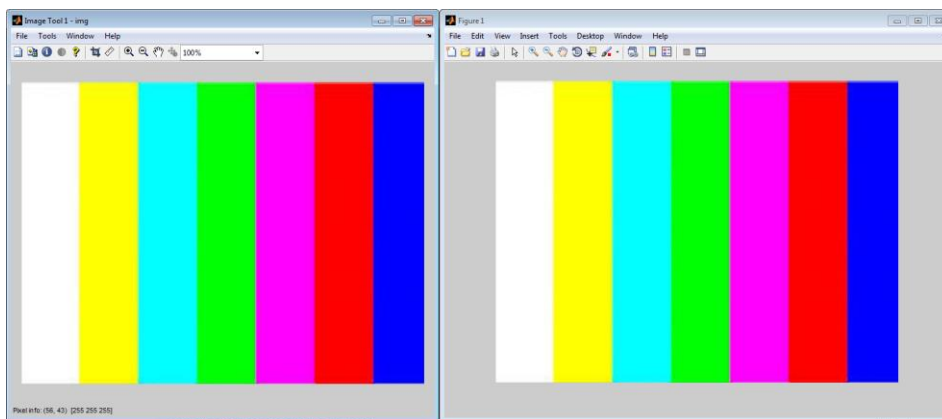
```
>> img=color_bar2;
```

A létrejött új tömb paramétereiben és adatiban is megegyezik az eredeti tömbbel.

Képek megjelenítésére számos utasítás közül válogathatunk. A két leghasznosabb az `imtool` és az `imshow` parancsok. Az `imshow` a kép egyszerű megjelenítésére szolgál, a képből kiolvastva határozza meg a színtérképet, az `imtool` használatával a kurzor mellett az aktuális pozícionál lévő pixelek értékeit is kiírja a MatLab.

```
>> imtool(img)
```

```
>> imshow(img)
```



**4-2. ábra. Az `imtool` és `imshow` utasításokra felugró ablakok.**

A példák során az `imshow` utasítást fogjuk alkalmazni a gyors megjelenítéshez, a tisztelt olvasóra bízunk, hogy felfedezze az `imtool` használatát.

Adott tehát egy színes kép, melynek szeretnénk megnézni a világosság eloszlás függvényét, vagyis világosság hisztogramját, mely vízszintes tengelyén az ábrázolt színárnyalatok, függőleges tengelyén pedig azok gyakorisága látható abszolút vagy normalizált skálán. Az erre szolgáló parancs az `imhist` utasítás.

```
>> imhist(img)
```

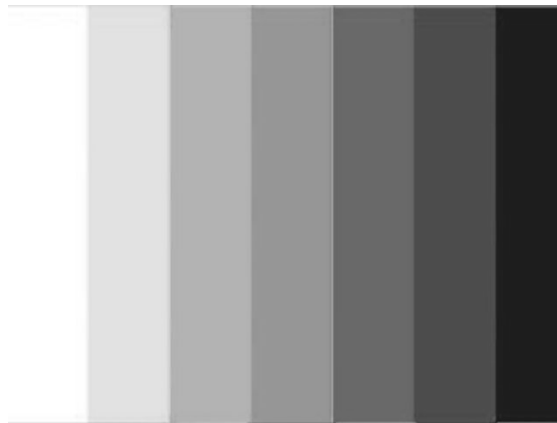
A parancs kiadása után a MatLab hibát jelez. Ez az utasítás ugyanis nem képes kettőnél több dimenziós adathalmaz megjelenítésére. A hisztogram kirajzolásához először szürkeárnyalatossá kell alakítani a képünket.

```
>> im_gray = rgb2gray(img);
```

## 4.2 Szürkeárnyaltos képek

Az újonnan létrejött `im_gray` változó méretét tekintve megegyezik az eredeti képpel (480x640) de már nem három, hanem egy dimenziós. A kép színtelepeit világosságértékekre számoltuk át.

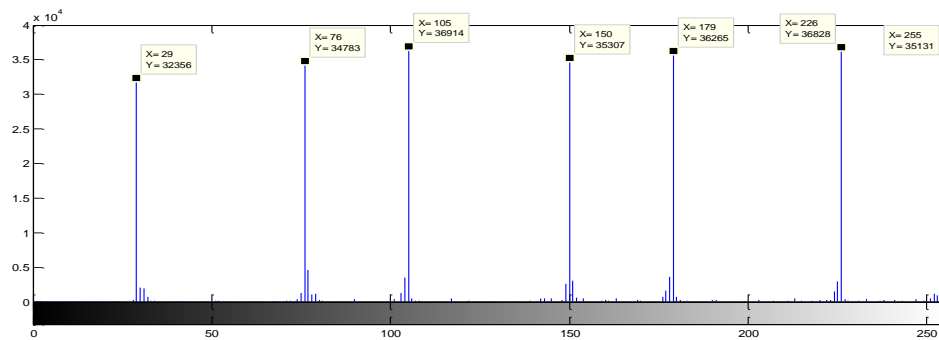
```
>> imshow(im_gray)
```



4-3. ábra. Az `im_gray` szürkeárnyaltos kép

Most már meg tudjuk jeleníteni a hisztogramot, hiszen ez egy kétdimenziós tömb.

```
>> imhist(im_gray)
```



**4-4. ábra. Az im\_gray hisztogramja**

A megjelenő ábrán jól láthatóan hét nagy vonal rajzolódott ki, illetve ezek mellett kisebb vonalak. A hét nagy vonal a színcsíkot alkotó hét színhez tartozó világosságértéket reprezentálja. A kisebb vonalak a színek határainál létrejövő színátmenetekhez tartoznak melyek az eredeti kép tömörítéséből (jpeg formátum) adódnak, a továbbiakban ezekről eltekintünk.

A hét világosságértékünk a következő:

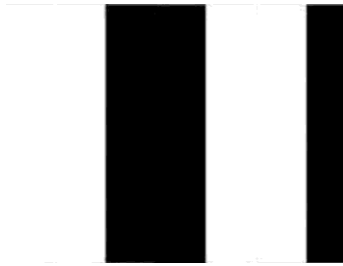
29  
76  
105  
150  
179  
226  
255

A következőkben ezeket szeretnénk azonosítani, ehhez a képet először csatornákra kell bontani. Térjünk tehát vissza az eredeti `img` tömbünkhöz és daraboljuk fel.

```
>> red = img(:,:,1);  
>> green = img(:,:,2);  
>> blue = img(:,:,3);
```

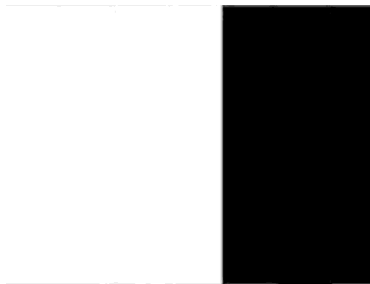
A létrejött `red`, `green` és `blue` tömbök az eredeti `img` tömb szétválasztott mátrixai.

```
>> imshow(red)
```



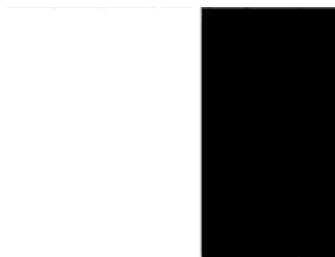
**4-5. ábra. Piros csatorna**

```
>>imshow(green)
```



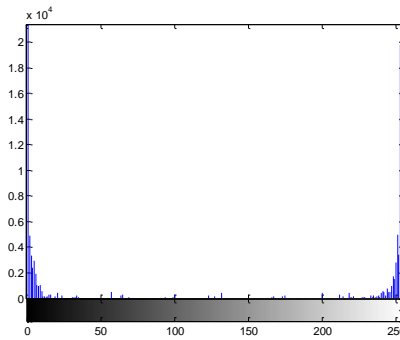
**4-6. ábra. Zöld csatorna**

```
>>imshow(blue)
```



**4-7. ábra. Kék csatorna**

Mint észrevehető ezek a képek szürkeárnyaltosak, hiszen a piros, zöld és kék színértékeit tartalmazzák, amik önmagukban csupán világosságértékeket adnak meg. Ha megnézzük hisztogramjukat azonos ábrát kapunk.



**4-8. ábra. Fekete-fehér kép hisztogramja**

A képen csupán fekete és fehér képpontok vannak, így a színek világosságértékének meghatározásához további műveletekre van szükségünk.

Módosítsuk a képeket, hogy a csatornákat újra színesnek lássuk. Ehhez először szükségünk lesz egy csupa nullából álló mátrixra, mellyel újra egyesítve a csatornák mátrixait újra 3 dimenziós tömböt kapunk.

```
>> a = zeros(size(img, 1), size(img, 2));
```

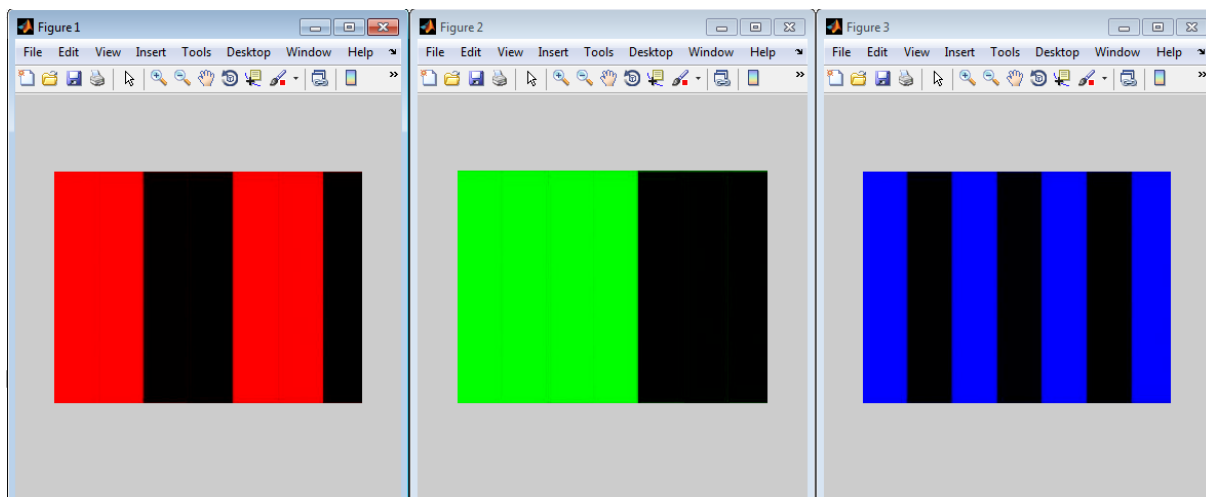
A létrejött a tömb méretében megegyezik az `img` mátrixsal, benne csupa 0 érték van.

Ezt kell összeraknom a `red`, `green` és `blue` mátrixokkal, hogy újra háromdimenziós, vagyis színes képként megjeleníthető tömböt kapjunk.

```
>> only_red = cat(3, red, a, a);
>> only_green = cat(3, a, green, a);
>> only_blue = cat(3, a, a, blue);
```

A létrehozott új tömböket már színes képként tudjuk megjeleníteni, hiszen a megfelelő tömbpozícióba kerültek a csatornák.

```
>> imshow (only_red)
>> imshow (only_green)
>> imshow (only_blue)
```



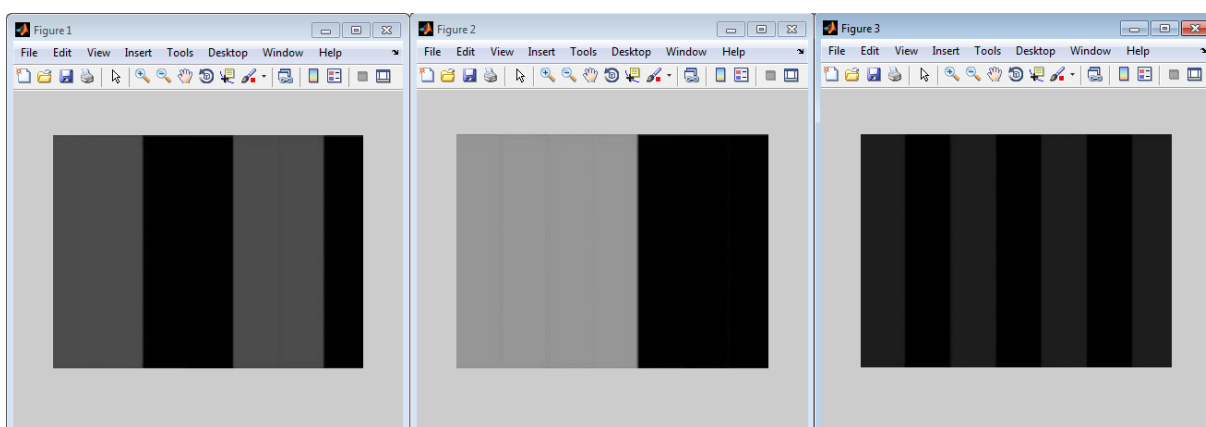
**4-9. ábra. Csatornák színesben megjelenítve**

Ezekből a képekből újra szürkeárnyaltos képet kell csinálnunk, hogy megkapjuk a megfelelő világosságértékeket.

```
>> jrk_gray = rgb2gray(only_red);
>> jgk_gray = rgb2gray(only_green);
>> jbk_gray = rgb2gray(only_blue);
```

Jelenítsük meg az új képeket!

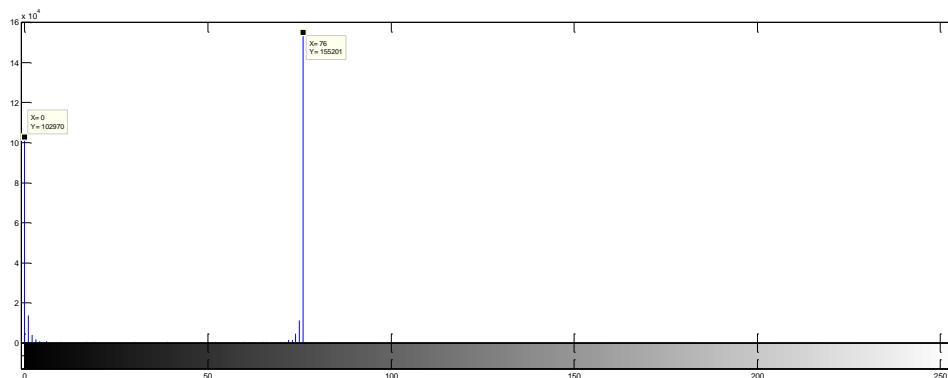
```
>> imshow (jrk_gray );
>> imshow (jgk_gray );
>> imshow (jbk_gray );
```



**4-10. ábra. RGB csatornák szürkeárnyalatban**

Látszódik, hogy a színek helyén az általuk képviselt világosság értékek vannak. Ezeket már tudjuk hisztogramon ábrázolni.

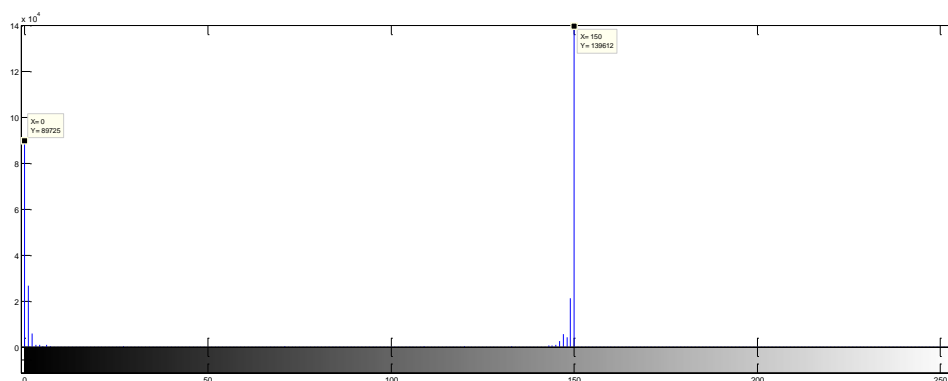
```
>> imhist (jrk_gray )
```



**4-11. ábra. Piros csatorna hisztogramja**

A piros csatorna hisztogramján a 0 szürkeérték a fekete szín, a 76-os érték a piros színnek megfelelő szürkeérték.

```
>> imhist (jgk_gray )
```

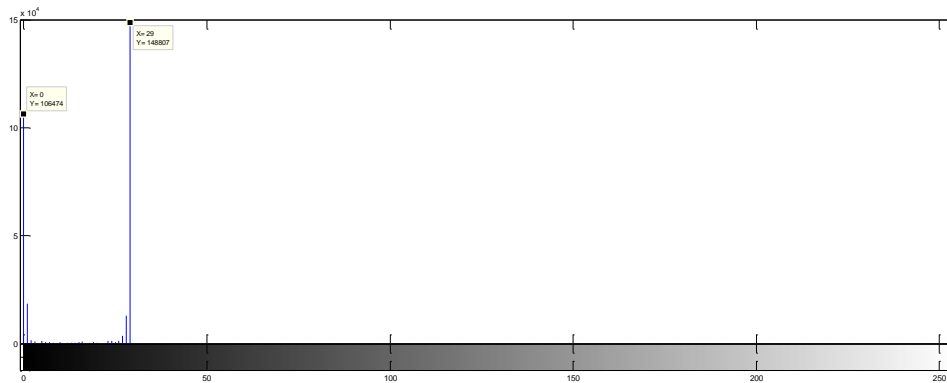


**4-12. ábra. Zöld csatorna hisztogramja**

A zöld szín szürkeértéke 150.

```
>> imhist (jbk_gray )
```





**4-13. ábra. Kék csatorna hisztogramja**

A kék színek megfelelő szürkeérték 29.

Máris azonosítottunk három értéket az eredeti hisztogramunkon.

29	<i>Kék</i>
76	<i>Piros</i>
105	
150	<i>Zöld</i>
179	
226	
255	

Már csak négy érték hiányzik. Ezekhez szükségünk lesz a szubsztraktív színpaletta színeinek létrehozására, melyet legegyszerűbben úgy tudunk létrehozni, hogy az eredeti `img` tömbből kivonjuk a piros, a zöld és a kék csatorna színeit.

```
>> Cr=img-only_red;
>> Cb=img-only_blue;
>> Cg=img-only_green;
```

Az új színes képeket szürkeárnyalatossá alakítjuk.

```
>> Cr_gray = rgb2gray(Cr);
>> Cg_gray = rgb2gray(Cg);
>> Cb_gray = rgb2gray(Cb);
```

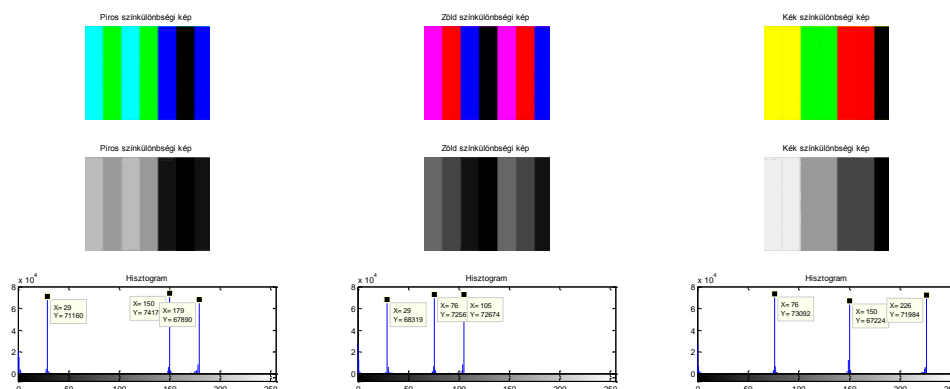
És ezeket is megjelenítjük hisztogramban. Azonban, hogy kicsit áttekinthetőbb legyen a megjelenő ábra, helyezzük egymás alá a különbségi színeket tartalmazó képeket és hisztogramjukat a `subplot` utasítás segítségével.

```
>> figure, title(' Színkülönbségi csatorna szürkeértékei')
subplot(3,3,1), imshow(Cr), title('Piros színkülönbségi kép')
```

```

subplot(3,3,2), imshow(Cg), title('Zöld színkülönbségi kép')
subplot(3,3,3), imshow(Cb), title('Kék színkülönbségi kép')
subplot(3,3,4), imshow(Cr_gray), title('Piros színkülönbségi kép')
subplot(3,3,5), imshow(Cg_gray), title('Zöld színkülönbségi kép')
subplot(3,3,6), imshow(Cb_gray), title('Kék színkülönbségi kép')
subplot(3,3,7), imhist(Cr_gray), title('Hisztogram')
subplot(3,3,8), imhist(Cg_gray), title('Hisztogram')
subplot(3,3,9), imhist(Cb_gray), title('Hisztogram')

```



**4-14. ábra. Színkülönbségi képek és hisztogramjaik**

Az újonnan megjelenő hisztogramokból az eddig ismert értékek segítségével meg tudjuk állapítani, a hiányzó értékeket. Az értékeket 255-tel elosztva pedig megkapjuk százalékos arányban a Fehér, mint 1-es – vagy ha úgy szemléletesebb 100% – értékhez képest mekkora világosság értéket képviselnek az egyes színek.

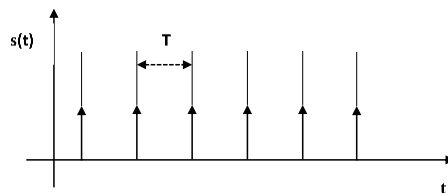
0	<i>Fekete</i>	0	0%
29	<i>Kék</i>	0,11	11%
76	<i>Piros</i>	0,29	29%
105	<i>Magenta</i>	0,41	41%
150	<i>Zöld</i>	0,58	58%
179	<i>Cián</i>	0,70	70%
226	<i>Sárga</i>	0,88	88%
255	<i>Fehér</i>	1	100%

## 5 Digitális jelek

Digitális rendszerekben csak véges mennyiségű adatokkal dolgozhatunk, viszont egy tetszőlegesen változó analóg jel csak végtelen sok adattal írható le pontosan. Belátható, hogy ha kellő sűrűséggel veszünk mintát az analóg jelből, akkor a mintákból jó közelítéssel helyreállítható lesz az eredeti jelünk.

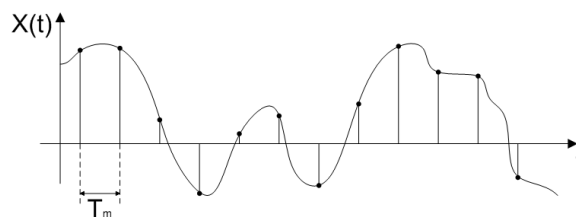
### 5.1 Mintavételezés, a mintavételi tétel

Első példaként tekintsünk egy olyan rendszert, amely végtelen felbontással képes amplitúdó mintákat ábrázolni, azaz a vizsgált sávkorlátozott analóg jelünket csupán mintavételezzük, és még nem kvantáljuk! Az ideális mintavételező függvény egy mintavételi időközönként ( $T_m$ ) érkező periodikus Dirac sorozat (a mintavételi frekvencia  $f_s=1/T_m$ ).



5-1. ábra.  $T_m$  periodusú Dirac sorozat

A Dirac impulzusokkal megszorozva a vizsgált jel időfüggvényét, a kapott mintavett jel egy  $n \cdot T_m$  időpillanatokban érvényes Dirac sorozat lesz, amelyek területe hordozza az adott időpillanatbeli jelminta értékét.



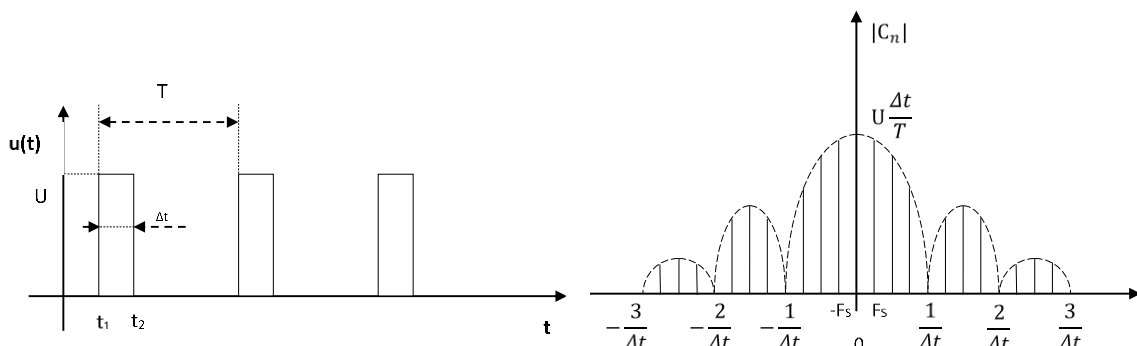
5-2. ábra. Eredeti és a mintavett jel

A mintavett jelet megvizsgálva láthatjuk, hogy a mintavett jel csak az adott időpillanatokban hordozza az eredeti jel értékeit, az intervallumok közé eső időpillanatokban első ránézésre nem hordoz információt.

#### 5.1.1 Mintavett jel spektruma

Először nézzük meg egy periodikus négyszögjel spektrumát: a spektrumkép vonalas, a spektrumvonalak burkolója a  $\sin(x)/x$  függvény és a négyszög impulzus szélessége (kitöltési tényezője,  $\Delta t$ ) határozza meg, hogy a  $\sin(x)/x$  függvény első zérushelye milyen messze esik az

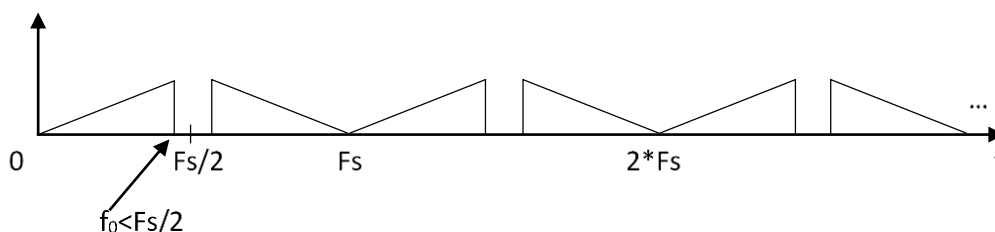
y tengelytől (ezt a távolságot, frekvenciában, nevezzük a négyyszögjel konvencionális sávszélességének – az ábrán  $1/\Delta t$ ). A négyyszögjel frekvenciája ( $F_s=1/T$ ) határozza meg a spektrumvonalak helyét és a vonalak távolságát.



5-3. ábra. Egy négyyszögjel és annak vonalas spektruma

Térjünk vissza a Dirac impulzussorozathoz, ami végtelen kicsi kitöltési tényezőjű periodikus négyyszögjelnek feleltethető meg, így ennek spektruma a  $\sin(x)/x$  burkolójú vonalas spektrumból levezethető, úgy hogy  $\Delta t$  értékével tartsunk 0-ba! A négyyszögjel spektruma erre a következőképp módosul. Az  $1/\Delta t$ -nél lévő zérushely a végtelenbe tart, ezzel létrehozva egy konstans magasságú, vonalas spektrumot, ahol a vonalak  $F_s$  távolságra vannak egymástól.

Ezek után vegyük elő a vizsgált (sávkorlátozott) jelünket és szorozzuk össze a Dirac impulzus sorozattal.

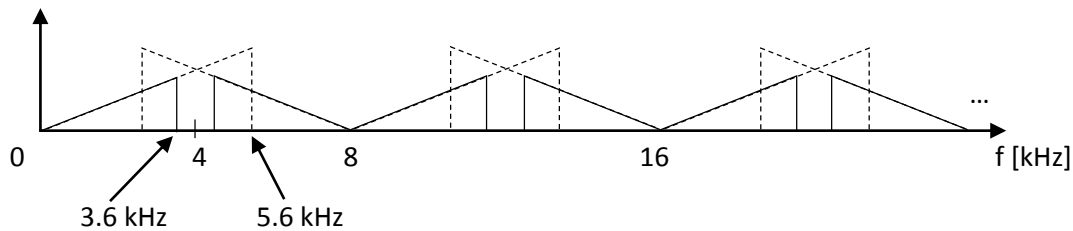


5-4. ábra. A mintavételezett periodikus spektrum

Spektrumképünk  $F_s$  szerint periodikussá válik a frekvencia tengely mentén. Az ábrán a vizsgált jel sávszélessége  $f_0$  volt, a mintavételi frekvencia  $F_s$ . Vizsgáljuk meg az első  $F_s$  körüli spektrumképet, mire emlékeztet? Az AM-DSB/SC modulált jel spektrumképe jelent meg előttünk, annyi különbséggel, hogy a frekvencia tengelyen  $F_s$  gyakorisággal ismétlődik, mintha végtelen sok vivőre ültettük volna jelünket.

Mi történik, ha a mintavételezett jel sávszélességét, illetve a mintavételi frekvenciát változtatjuk? A két értéket a mintavételi törvénnyel összhangban változtathatjuk csak: azaz

$F_s > 2 \cdot f_0$  különben a spektrumok átlapolódnak, aminek eredményeképpen az eredeti jelet visszaállítani nem lehet. Például egy  $f_s = 8$  kHz-es mintavételi frekvencia mellett egy 3.6 kHz-es sávkorlátozott jel mintavételezésével a mintavételezési törvényt nem sértjük meg, viszont, ha a sávkorlát pl. 5.6 kHz-re nő, az alábbi ábrán látható a spektrumok átlapolódása. Ennek eredményeképpen a visszaállított jelben az eredetivel keveredve megjelennek a 4 kHz alatti szaggatott vonallal jelzett frekvenciakomponensek is.



5-5. ábra. Átlapolódás a spektrumban

### 5.1.2 Feladatok

Vizsgáljuk meg a fenti eseteket egy szinuszos jelen, az alias.m fájlban található függvény segítségével! Először hívjuk meg az alias függvényt paraméterek nélkül, ekkor a mintavételi frekvencia  $f_s = 8$  kHz és a mintavételezett szinuszos jel  $f_0 = 500$  Hz.

```
>> alias
```

A megjelenő ábrán fent az eredeti jel a mintavételi pontokkal. Alul a mintavett jel elvi spektrumának 0 és  $f_s \cdot 3$  értékek között ábrázolt része látható. Az amplitúdó értékeket egységnyire állítottuk, vizsgálatainkban csak a frekvencia tengelyen felvett pozíciók a fontosak. Szaggatott vonalakkal ábrázoltuk a mintavételi frekvencia egész számú többszöröseit (ezek csak segédvonalak, nem frekvencia komponensek!), míg folytonos vonallal a mintavételezett jelhez tartozó frekvencia komponenseket. A színek a mintavételezésből adódó, periodicitásból származó, „összetartozó” spektrumvonalakat jelzik.

1. Értékeljük a kapott ábrákat, vizsgáljuk meg a mintavételezett jelünk spektrumát, használjuk hozzá a data cursort! Milyen frekvenciákon találjuk az ábrázolt komponenseket?
2. Egészítsük ki a az alábbi képleteket, hogy hogyan számíthatók ki  $f_s$  és  $f_0$  segítségével az  $n$ -dik (mintavételezésből adódó) komponensek frekvenciái:

kék komponensek:  $f_{\text{alsó}} = 1 \cdot f_s - f_0$ ,  $f_{\text{felső}} = 1 \cdot f_s + f_0$

sárga komponensek:  $f_{\text{alsó}} = \dots \cdot f_s - f_0$ ,  $f_{\text{felső}} = \dots \cdot f_s + f_0$

3. Vizsgáljuk meg  $f_0=3000$  Hz esetén a mintavételezett jel frekvenciáját. Számítsuk ki először az egyes komponensek értékeit, majd ellenőrizzük az értékeket a kapott ábráról. Hívjuk meg az alias függvényt az alábbi módon:

```
>> alias(3000)
```

4. Az alapértelmezett 8 kHz-es mintavételi frekvencia esetén, mekkora a legnagyobb frekvencia, amit a Shannon mintavételezési törvény értelmében mintavételezhetünk?

A digitális jel visszaállításakor (analóggá alakításakor) a mintavett spektrum 0 és  $f_s/2$  tartományba eső komponensek maradnak csak meg. Az eddigi feladatokban a mintavételi törvényt betartottuk, az eredeti jelünket teljes mértékben vissza lehetett állítani a rögzített mintákból. A következő feladatokban vizsgáljuk meg, ha nem tartjuk be ezt a szabályt, azaz átlapolódást, más nevén „aliasing”-ot idézünk elő.

5. Mintavételezzünk a továbbra is 8 kHz-es mintavételi frekvencia mellett  $f_0=5000$  Hz frekvenciájú szinusz jelet, így megsértjük a mintavételi törvényt! Számítsuk ki először, hogy milyen komponenseket kapunk a mintavételezett spektrumban!
6. Ellenőrizzük az alias használatával az eredményeket! Figyeljük meg a kapott ábrán az eredeti jelen, hogy a mintavételi pontok a korábbiakhoz képest mennyivel ritkábbak.

Állítsunk elő most olyan helyzetet mellyel durván megsértjük a mintavételi függvényt, használjunk ehhez 7900 Hz-es szinusz jelet! Ebben az esetben az időtartományi ábráról (felső) is könnyen leolvasható, hogy a mintákból milyen frekvenciájú szinusz állítható vissza.

7. Hívjuk meg az alias függvényt 7900 Hz-es paraméterrel! Olvassuk le a mintákból kirajzolódó szinusz periódusidejét a felső ábráról, majd számoljuk ki ebből a frekvenciáját, vessük össze a spektrumábrán található komponenssel (a  $0...f_s/2$  tartományban).
8. Minimálisan elvileg mekkora mintavételi frekvenciával lehet megfelelően mintavételezni 7900 Hz-es jelet?
9. Ellenőrizzük az alias függvény következő formájú meghívásával,  $f_s$ -t helyettesítsük a kiszámolt értékkel, majd többféle  $f_s$ -sel is nézzük meg az eredményt!

```
>> alias(7900, fs)
```

### 5.1.3 Az alias függvény

```
function alias(f0, fs)
    % alapértelmezett értékek, ha 1 vagy 0 paraméterrel hívjuk meg a függvényt
    if nargin < 2, fs = 8000; end
    if nargin < 1, f0 = 500; end
    fi=0;
    Ts=1/fs; % mintavételi időköz
    t1=[0:.000001:.01];
    x1=sin(2*pi*f0*t1+fi); % eredeti jel reprezentációja 1 MHz-es mintavétellel
    t2=[0:Ts:.01];
    x2=sin(2*pi*f0*t2+fi); % mintavett jel

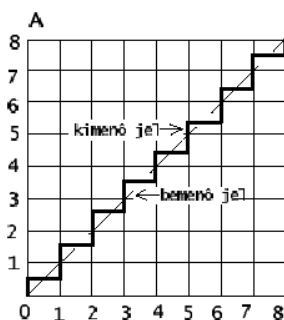
    subplot(2,1,1);
    plot(t1,x1); hold on
    plot(t2,x2,'ro');
    hold off
    title(['mintavételi frekvencia, fs=',int2str(fs),' Hz; analóg jel frekvenciája, f0=',int2str(f0),' Hz']);
    xlabel('idő [s]')
    bars=[];
    for i=0:5
        bars(i+1,1)=i*fs-f0;
        bars(i+1,2)=i*fs;
        bars(i+1,3)=i*fs+f0;
    end

    subplot(2,1,2);
    mycolor=['r' 'g' 'b' 'y' 'm' 'k'];
    stem(bars(1,2),1,mycolor(mod(1,6)+1),'Marker','none','LineStyle','--'); hold on
    stem(fs/2,1,'black','Marker','none','LineStyle','--');
    stem(bars(1,3),1,mycolor(mod(1,6)+1),'o');
    for i=2:length(bars)
        stem(bars(i,1),1,mycolor(mod(i,6)+1),'o');
        stem(bars(i,2),1,mycolor(mod(i,6)+1),'Marker','none','LineStyle','--');
        stem(bars(i,3),1,mycolor(mod(i,6)+1),'o');
    end
    title('A mintavételezett jel spektruma (eredeti jel: zöld)');
    xlabel('frekvencia, [Hz]')
    axis([-1 fs*3+1 0 2]);
    hold off
```

Ne felejtsük el, hogy a MatLab diszkrét értékekkel dolgozik, így az eredeti jelnek hívott jelünk is csak egy mintavételezett jel. Ennek értékeit a pontos kirajzolás érdekében 1 MHz-es mintavételi frekvenciával vettük fel, és folytonosság érzékeltetése miatt a pontjait a plot függvénnyel összekötve ábráztuk.

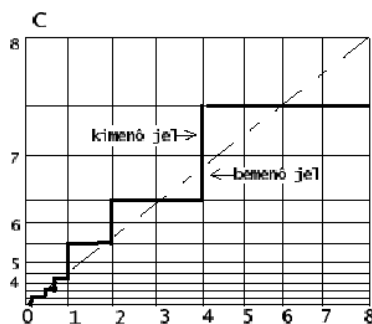
## 5.2 Kvantálás

Kvantáláskor a bemeneti jelünkhöz, mely tetszőleges értékeket vehet fel, rendelünk a kvantálási lépcsőknek megfelelő diszkrét értékeket. Az ábra szerinti kvantálási lépcső alapján, pl. ha a bemeneti jelünk 3.11 (a vízszintes tengelyen), akkor kimenő jelnek 3.5-et kapunk, míg 3.79-es értékhez szintén 3.5-et kapunk. Mivel itt a lépcsők nagysága azonos, ezt lineáris kvantálásnak nevezzük.



5-6. ábra. Lineáris kvantálási görbe

Nemlineáris kvantálás esetén a kvantálási lépcsők magassága nem azonos. A leggyakrabban használt nemlineáris kvantálás a logaritmikus görbét közelítő „A kvantálási karakterisztika”:



5-7. ábra. Nemlineáris kvantálási görbe

Ezzel a kvantálási karakterisztikával jobb eredménnyel tudunk olyan jeleket kvantálni, amelyek sok kis amplitúdójú jelet tartalmaznak (pl. beszéd). Itt a kisebb amplitúdójú jelszintek ábrázolása nagyobb felbontással történik, míg a nagy amplitúdójú jelek esetén elnagyoltabb a kvantálási szint hozzárendelés. Pl.:

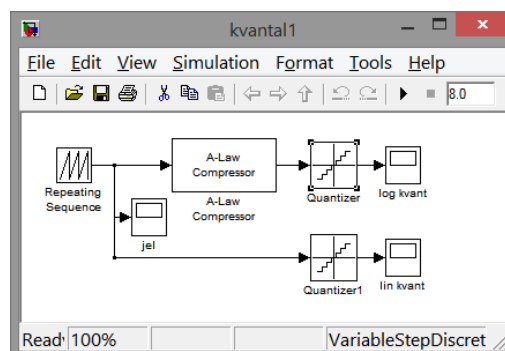
- 1.5-es bemeneti jelszinthez látható, hogy 5.5 tartozik,
- 2.5-hez 6.3,



- 5.5 és 6.5 esetén mindkét esetben 7.4-et kapunk, holott a bemeneti jelek közötti különbség azonos volt.

### 5.2.1 Feladatok

Vizsgáljuk meg a lineáris és az A karakterisztika működését szimulációval A MatLab SimuLink felületén! Nyissuk meg a kvantal1.mdl fájlt! A megnyitáskor a workspace-en létrejön egy quantstep=8 változó mely a kvantálási lépcsők számát adja meg pozitív és negatív amplitúdókra.



5-8. ábra. Lineáris és logaritmikus kvantálás a SimuLinkban

Az ablak alatt megjelenő 3 ablak az azonos nevekkkel jelölt szkópok képeit mutatja. Ha véletlenül bezárjuk őket az adott szkópra duplán kattintva az ablak ismét megjelenik. A szkópok ábráit PrintScreen-nel menthetjük el legegyszerűbben. Az ablak skálázását pedig automatikusan elvégeztethetjük, ha abban jobb klikk után az Autoscale pontra kattintunk.

1. Az ablak eszköztárában találunk egy lejátszás (play, ►) gombot, mellyel a szimuláció indítható, futtassuk!

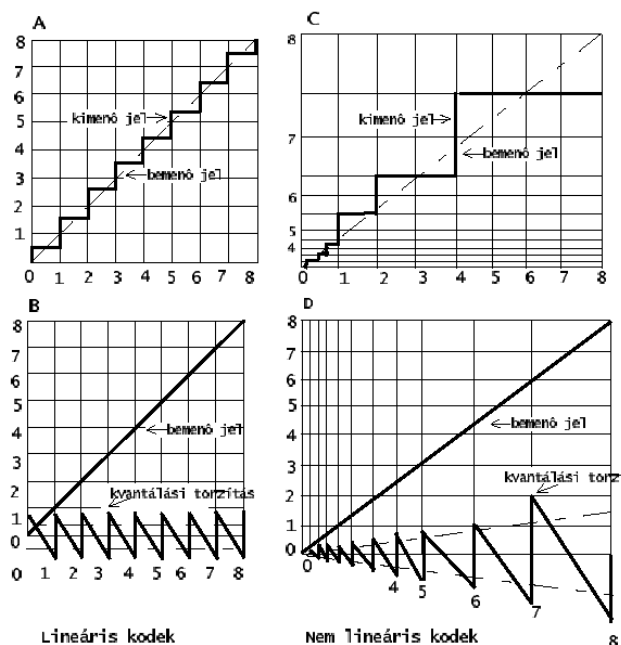
Az alsó három ablakban megjelennek az egyes jelalakok. Az első ('jel') maga a generált kvázi fűrészel látható, míg a másik kettőn a lineáris, illetve A karakterisztikás kvantált jel. Az eredeti beállítás szerint a nulla szinttel együtt 8-8 kvantálási lépcső adott, a bemenő jel  $-1$  és  $+1$  között változik egyenletesen, a kvantálási tartomány pedig szintén  $-1$  és  $+1$  közé esik. Ha nagyítunk az ábrákon (vagy teljes képernyőn vizsgáljuk) láthatóvá válnak jobban a kvantálási lépcsők!

2. Vizsgáljuk meg, hogy kis amplitúdók esetén valóban az „A” karakterisztika felbontása jobb-e, azaz használ több lépcsőt a bemenő jel ábrázolására! Változtassuk meg a bemenő jel paraméterét úgy, hogy a kvantálási tartomány 1 értékéhez képest annak a maximális értéke csupán 0.1 legyen. Kattintsunk duplán a Repeating Sequence dobozon, majd az

*Output values mezőben a  $[-1 \ 1]$ -et írjuk át  $[-0.1 \ 0.1]$ -re! Értékeljük a futtatás utáni eredményeket, számoljuk össze, hogy összesen hány kvantálási lépcsőt használ az egyik és másik módszer!*

3. *Melyik kvantálás ábrázolja több lépcsővel a kis amplitúdójú jeleket?*
4. *Vizsgáljuk meg nagy amplitúdók esetén, hogy ott valóban milyenek a felbontások! Változtassuk a bemenő jelet úgy, hogy végig relatíve magas legyen az amplitúdó, most legyen az Output values mezőben pl.  $[0.5 \ 1]$ . Értékeljük a kapott ábrákat!*

## 5.2.2 Kvantálási zaj és jel zaj viszony lineáris és logaritmikus kvantálás esetén



5-9. ábra. Kvantálási zajok

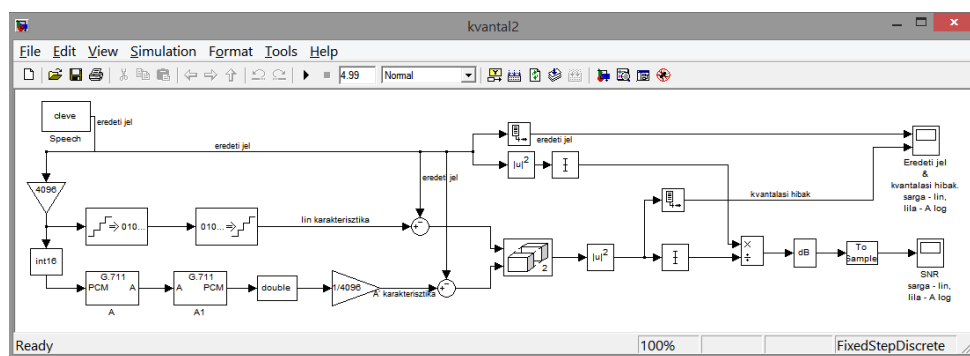
A kvantálási görbék ábráit kiegészítettük egy felfelé futó fűrészjel (bemenő jel) kvantálásakor keletkező hibákkal, ez látható a B és a D ábrán! A lineáris kvantálásnak (B) köszönhetően a kvantálási torzítás a jel teljes dinamikatartományában egy állandó érték alatt marad (kvantálási lépcső magasságának fele). Ebből az következik, hogy amennyiben a kvantálandó jel szintje magas, a jel-zaj arány relatíve magas lesz, míg alacsony jelszint esetén ez az arány igencsak alacsony lesz. Lineáris kvantálás elsősorban szórakoztató elektronikában, audió- és videótechnikában használatos.

Logaritmikus kvantálás esetén (D) a változó nagyságú kvantálási lépcsőknek köszönhetően kicsi jelszint esetén a kvantálási zaj aránya relatíve kicsi marad, míg nagy jelszint esetén relatíve nagy lesz (nagyobb jelszintek esetén a kvantálási zaj mértéke meghaladja a lineáris kvantálás zaját is!). A relatív hibaarány (jel-zaj viszony) viszont a tartomány nagy

részen állandó értéken tartható. Gyakorlati szempontból az emberi beszéd és az ezzel kapcsolatos emberi beszéd érzékeléshez illeszkedő a logaritmikus kvantálás. A beszédátvitelnél körülbelül 30dB jel-zaj viszonyt az ember jó minőségűnek mond, következésképpen érdemes alacsony jelszinten finomabb felbontást alkalmazni kvantáláskor, viszont nagyobb jelszintek esetén felesleges a jel-zaj viszony javítása. Az állandónak tekinthető jel-zaj viszony a beszédkódolás esetén előnyös, így kevés bitszámon nagy dinamikatartomány érhető el.

### 5.2.3 Feladatok

Indítsuk el a kvantal2.mdl SimuLink szimulációt. A megjelenő ablakban egy rögzített beszédminta kvantálási torzítását fogjuk megvizsgálni. A jelet lineáris és „A” karakterisztikás kvantálóval kvantáljuk majd kódoljuk, utána visszaállítjuk. Az eredeti és a visszaállított jeleket egymásból kivonva meghatározhatjuk az egyes eljárások esetén érvényes hiba jelet, azaz zajt, majd ezt az eredeti jelhez viszonyítva számíthatjuk a relatív hiba arányt, a jel-zaj viszonyt. Futtassuk a szimulációt!



5-10. ábra. Jel-zaj viszony értékek számítása lin és log kvantálás esetén

Két szkóp képe jelenik meg az ablak alatt (ha nem kattintsunk duplán az szimulációs ablak jobb oldalán a két szkópra). Az első ablak felső ábráján az eredeti beszédjelet láthatjuk. Az alsón a kétféle kvantálás után helyreállított jel és az eredeti különbségét, azaz magát a kvantálási zajt. A másik ábrán pedig a szimuláció ideje alatt mérhető jel zaj viszonyt. Sárga szín a lineáris, míg a lila a logaritmikus („A” karakterisztika) kvantáló jeleit mutatja.

Vizsgáljuk meg a kapott ábrákat lineáris kvantálás tulajdonságainak szempontjából (sárga görbék)!

1. Milyen mértékű a kvantálási zaj kicsi és nagy beszéd-amplitúdók esetén? Olvassuk le a hibák maximum értékeit hozzávetőlegesen az ábráról legalább két, különböző bemeneti amplitúdó esetén!
2. Hogyan alakul a relatív hibaarány (SNR) a leolvasott értékeknél?

3. Az eredeti jel amplitúdó szempontjából milyen jelszinteknél magasabb, illetve alacsonyabb a jel zaj viszony értéke?

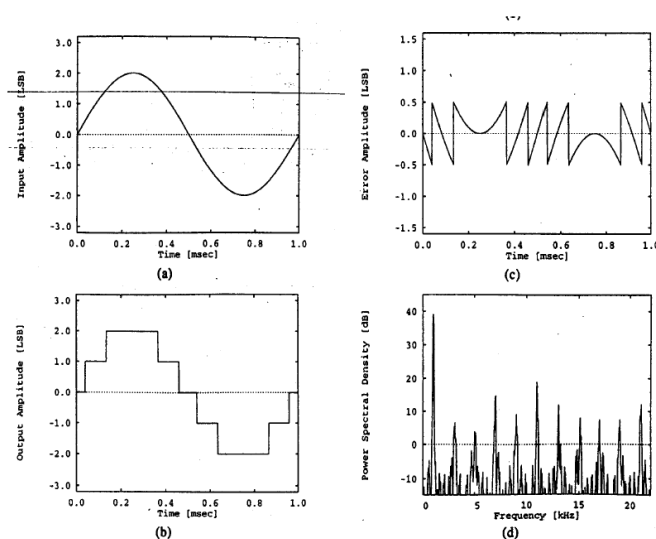
Most vizsgáljuk meg a kapott ábrákat logaritmikus kvantálás tulajdonságainak szempontjából (lila görbék)!

4. Milyen mértékű a kvantálási zaj kicsi és nagy beszéd-amplitúdók esetén? Olvassuk le a hibák maximum értékeit hozzávetőlegesen az ábráról az előző feladatban kiválasztott időpontokban!
5. Hogyan alakul a relatív hibaarány (SNR) a leolvasott értékeknél?
6. Az eredeti jel amplitúdó szempontjából mennyire függ a jel zaj viszony értéke a magasabb, illetve alacsonyabb bemenő jelszintektől?

#### 5.2.4 Dithering alkalmazása

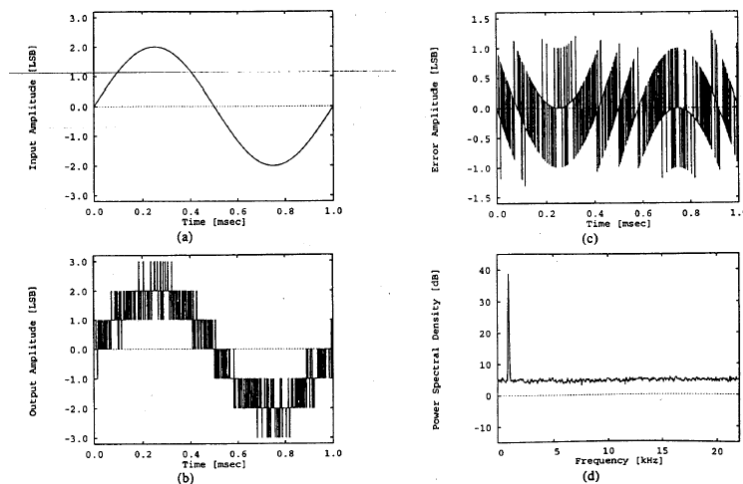
Kvantálási zaj tulajdonságait az előző feladatban megvizsgáltuk. Kvantálási torzítás jelensége akkor fordul elő, ha a kódolandó jel olyan kicsi, hogy az összemérhető a kvantálási lépcsők méretével. Ekkor a kvantálás után kapott függvényünk nagyon „lépcsős” lesz. Ebben az esetben jelünk spektrumában a hasznos jel teljesítményével összemérhető torzítási komponensek is megjelennek.

Amennyiben a kódolandó jel a példa szerinti szinuszos jel, úgy diszkrét csúcsokat láthatunk a spektrumképen. A visszaállított jel esetében ezek a komponensek igen zavaróak lesznek. Kiszűrésükhöz nagyon precíz szűrőre lenne szükség.



5-11. ábra. Kis amplitúdójú jelek esetén megjelenő torzítási komponensek

Amennyiben a kódolandó, vagy magához a kódolt jelhez a kvantálási lépcsővel összemérhető amplitúdójú zajt keverünk – ez a Dither zaj – akkor a helyreállítás során a fehér zaj szint ugyan kissé megnő, de a sokkal kellemetlenebb hangzást keltő torzítási komponensek eltűnnek:



5-12. ábra. Kis amplitúdójú jelek esetén Dither zaj alkalmazása

### 5.2.5 Feladatok

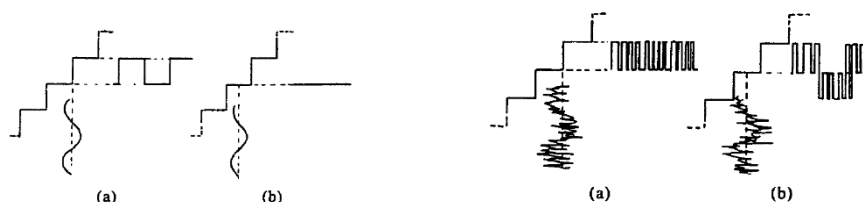
Nyissuk meg a dither1.mdl fájlt. A szimulációban egy szinuszos jelet vizsgálunk, melynek amplitúdója 3, frekvenciája 1000 Hz. A kvantálási tartomány amplitúdóban 16, kvantálónk 16 szintet különböztet meg, 4 bitre kódol. Jelünk a 16 szintből igen keveset „használ majd ki”.

1. Ellenőrizzük, hogy a Gauss zajgenerátor utáni „erősítő” értéke 0 legyen, azaz ne adjunk zajt az eredeti jelhez. Futtassuk a szimulációt. Értelmezzük az ábrákat, kövessük végig a folyamat különböző fázisait!
2. A megjelenő FFT ablakban láthatjuk a dekódolt jel spektrumát. Mely komponens a hasznos jelünk?
3. Melyek a torzítás miatt megjelenő komponensek?
4. Olvassuk le és jegyezzük le hozzávetőlegesen az első 4 komponens frekvenciáját és amplitúdóját a dB skálán!
5. Adjunk a kódolandó jelünkhöz Dither zajt! A Gaussian Noise Generator után található erősítő értékét írjuk át 0-ról 1-re! Futtassuk a szimulációt! A megjelenő FFT ablakban milyen változást láthatunk?
6. Mely komponens a hasznos jelünk?
7. Mekkora a zaj szintje?

8. *Változtassunk a kódolandó (eredeti) jel amplitúdóján úgy, hogy jobban kihasználjuk a kvantálási tartományt! Az eredeti 3-ról növeljük meg 15-re az amplitúdót (a kódoló kvantálási tartománya 16 amplitúdóig terjed!).*
9. *Futassuk a szimulációt Ditherzaj nélkül, majd azzal! A spektrumképen nézzük meg, hogy nagyobb amplitúdójú jel esetén érdemes-e használni a Dither zajt?*

### 5.2.6 Dithering alkalmazása 2

Előfordulhat, hogy kódolandó jelünk annyira kicsiny, hogy változása kisebb a kvantálási lépcsőknél. Ha a jel a kvantáláskor éppen egy lépcső határára esik, akkor visszaállításkor is lesz nyoma (a), de ha a teljes tartománya egy kvantálási lépcsőbe esik, akkor a kódolás után folytonos 0-át kapunk.



5-13. ábra. Kis amplitúdójú jel kvantálása

Dither zaj alkalmazásával, egy kis hozzáadott fehér zaj elviselése mellett ez is megoldható. Amennyiben egy lépcsőnél kisebb amplitúdójú jelhez adunk Dithert, akkor az általa keltett zajból fülünk átlagoló képességének köszönhetően kihalljuk a hasznos jelet. A második ábrán látható, hogy a b) esetben már nem azonosan 0 a kvantálás után a jelünk. Ezzel a módszerrel akár a lépcső méreténél tízszer kisebb amplitúdójú jeleket is meghallhatunk, úgy hogy a PCM kódszavak hosszát (a kódolás bitszámát) nem kellett növelnünk. Az átlagolás után előtűnik a hasznos jel.

### 5.2.7 Feladatok

Nyissuk meg a dither2.mdl fájlt! A kódolandó jel egy szinuszos jel lesz, amit úgy paraméterezünk, hogy épp egy kvantálási lépcső tartományába essen. A kvantálási tartományunk 16 szintű. A szinusz jel amplitúdója 0.9 és +1 offsettel rendelkezik, hogy ne essen bele az  $y=0$ -nál lévő kvantálási lépcsőbe.

1. *Ellenőrizzük, hogy a Gauss zajgenerátor utáni „erősítő” értéke 0 legyen, azaz ne adjunk zajt az eredeti jelhez. Futtassuk a szimulációt! Látható, hogy a kódolón nem jutott át jel.*

2. *Kapcsoljuk be a Dither zajgenerátort az utána lévő erősítő értékének 1-re állításával! Értékeljük, hogy mit kapunk a rendszer kimenetén!*
3. *Változtassuk a bemenő jel amplitúdóját nagyobbra (pl. 2). Ebben az esetben van haszna a Dither zaj alkalmazásának?*

## 6 Jelek spektrális felbontása, Fourier transzformáció

A jelek spektrális, vagy frekvencia tartománybeli leírása, spektrális tulajdonságaik a villamosmérnöki gyakorlatban nélkülözhetetlen ismeretek. Spektrumvizsgálatokra úgynevezett spektrumanalizátor műszerek állnak rendelkezésünkre, melyek frekvencia tartományban rajzolják fel az egyes összetevők amplitúdó értékét. Spektrális összetevők az időtartományban megfigyelt (pl. oszcilloszkópos méréssel) eredmények alapján is számolhatók.

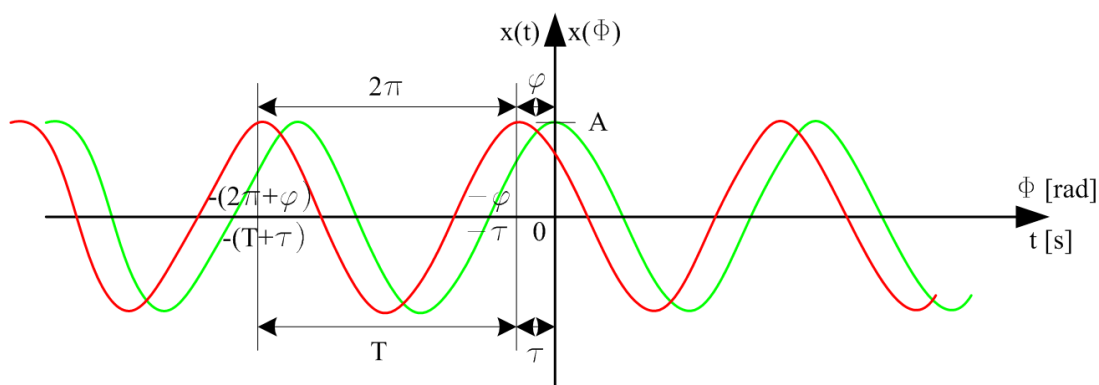
### 6.1 Általános bevezető

Mint ismeretes, egy  $f(t)$  periodikus időfüggvény felbontható egy konstans tag, továbbá véges, illetve végtelen sok szinusz, koszinusz időfüggő összetevő összegére. A periodikus jel  $T$  periódusidejének reciproka adja az alapharmonikus frekvenciáját, a további összetevők (felharmonikusok) frekvenciája ezen alapharmonikus egész számú többszörösei. A konstans tag nem más, mint a jelfolyam DC összetevője (0 frekvenciás tag), melyre a váltakozó komponensek rásuperponálódnak.

Mielőtt rátérnénk a Fourier sor matematikai leírására, tekintsük át a harmonikus jelek leírását, amely csupán ismétlésként szolgál. Alapnak tekintjük a tisztán páros cos rezgést (zöld). A sin jel ehhez képest 90 fokkal késik.

$$\sin \varphi = \cos \left( \varphi - \frac{\pi}{2} \right)$$

Egy általános harmonikus rezgést látunk példaképpen (piros), amely a cos-hoz képest siet. A fáziseltérést radiánban, vagy időben is kifejezhetjük.



6-1. ábra. Általános harmonikus rezgés.

A frekvencia:  $f = \frac{1}{T} \text{ [Hz]}$

A körfrekvencia:  $\omega = \frac{2\pi}{T} \left[ \frac{\text{rad}}{\text{s}} \right]$

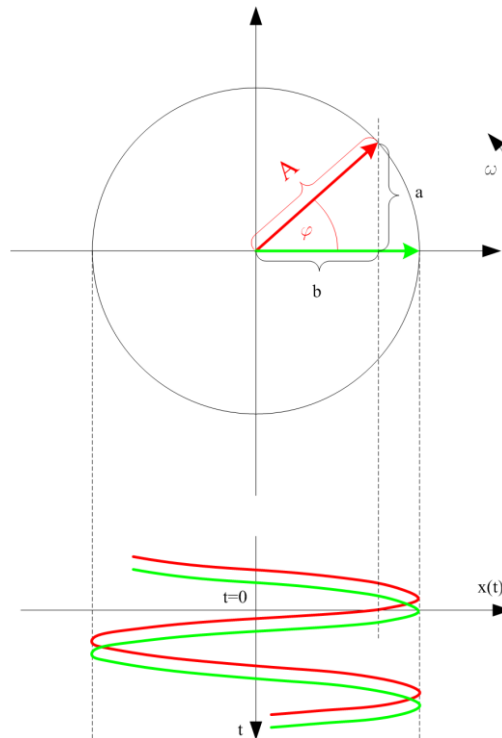


Az általános harmonikus rezgés (piros) matematikailag:

$$x(t) = A \cos(\omega t + \varphi) = A \cos\{\omega(t + \tau)\}$$

A negatív kezdőfázis fáziskésést, a pozitív fázissietést jelent az alapjel 0 fázisához képest.

Leírhatjuk ugyanezt a jelet egy  $\omega$  szögsebességgel forgó vektor vetületével is:



6-2. ábra. Forgó vektoros leírás

$$\sin \varphi = \frac{a}{A} \rightarrow a = A \sin \varphi$$

$$\cos \varphi = \frac{b}{A} \rightarrow b = A \cos \varphi$$

$$A = \sqrt{a^2 + b^2} \quad \text{és} \quad \operatorname{tg} \varphi = \frac{a}{b} \rightarrow \operatorname{arc} \operatorname{tg} \frac{a}{b} = \varphi$$

Ha ismerjük, hogy

$$\cos(\alpha + \beta) = \cos \alpha \cos \beta - \sin \alpha \sin \beta,$$

akkor

$$x(t) = A \cos(\omega t + \varphi) = A \cos(\omega t) \cos \varphi - A \sin(\omega t) \sin \varphi = b \cos(\omega t) - a \sin(\omega t)$$

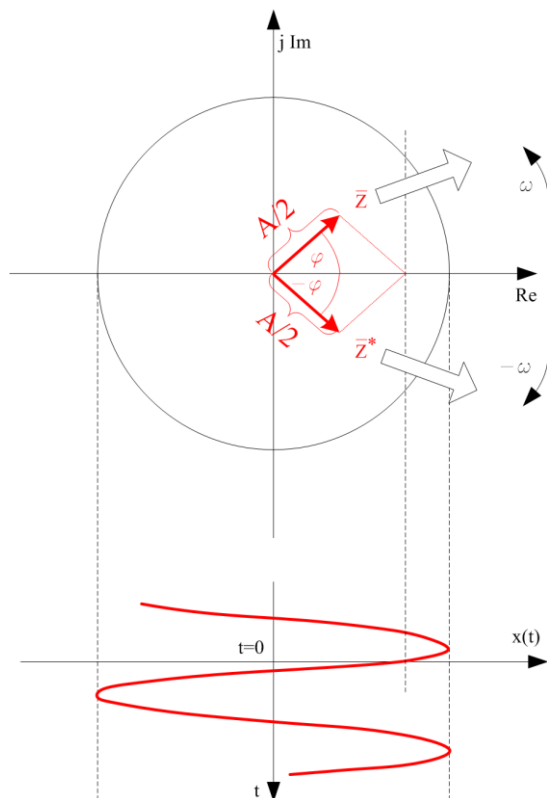
Tulajdonképp a jelünket két azonos frekvenciájú tiszta cos és sin jelek előjeles összegeként írtuk le. A kezdőfázist és az amplitúdót az „a” és a „b” együtthatók határozzák meg.

Számunkra a legizgalmasabb leírási mód viszont a komplex forgó vektorok eredőjével történő megadás. Euler-féle összefüggések összegéből és különbségéből következik:

$$\cos x = \frac{e^{jx} + e^{-jx}}{2} \quad \sin x = \frac{e^{jx} - e^{-jx}}{2j},$$

majd alkalmazva:

$$x(t) = A \cos(\omega t + \varphi) = A \frac{e^{j(\omega t + \varphi)} + e^{-j(\omega t + \varphi)}}{2} = \frac{A}{2} e^{j\omega t} e^{j\varphi} + \frac{A}{2} e^{-j\omega t} e^{-j\varphi}$$



6-3. ábra. Harmonikus jel komplex forgó vektorokkal

Bevezetjük a komplex amplitúdók fogalmát, amely az amplitúdó és a kezdőfázis információt hordozza:

$$\bar{A} = A e^{j\varphi} \quad \bar{A}^* = A e^{-j\varphi}$$

Ezek után felírhatjuk:

$$A \cos(\omega t + \varphi) = \frac{\bar{A}}{2} e^{j\omega t} + \frac{\bar{A}^*}{2} e^{-j\omega t}$$

Tehát a valós harmonikus rezgésünk leírható két, ellentétes irányba, de azonos szögsebességgel forgó, azonos nagyságú komplex vektor összegeként. Az összeg mindig valós érték, hiszen a két vektor minden időpillanatban komplex konjugált vektor-párt alkot.

Az elforgásért természetesen az exponens tag időben változó komplex hatványa a felelős. Ez az időben pozitív és negatív irányban növekvő additív fázist kölcsönöz a komplex amplitúdó

vektoroknak. Figyeljük meg, hogy a vektorok abszolút értéke csak fél amplitúdóval szerepelnek, hiszen az összegzés után kapjuk meg a teljes amplitúdót, a vektorok egyező állása esetén.

A negatív frekvencia értelmezése: az ellentétes körüljárási irányba forgó vektor körfrekvenciája. Ne ijedjünk meg tehát ennek hallatán! Ez egy matematikai leírási mód.

## 6.2 Fourier sor alakjai

Ezek után nézzük meg, hogy ha egy tetszőleges periodikus jel a Fourier sora szerint több (véges vagy végtelen sok) harmonikus rezgést tartalmaz, akkor a sor maga hogyan írható fel, igazodván az előbbi alakokhoz.

Amikor egy periodikus jelet Fourier összetevőire bontunk, akkor már át is tértünk időtartományról frekvenciatartományra. Az összetevők frekvenciatartomány megjelenítése adja a jel spektrumát. A spektrum ismerete nagyon fontos a műszaki gyakorlatban, mivel a számításaink túlnyomó többségét sokkal egyszerűbb frekvenciatartományban elvégezni, mint időtartományban. A periodikus jelek spektruma diszkrét és vonalas.

### 6.2.1 Fourier sor 1. alak:

$$x(t) = \sum_{k=0}^{\infty} B_k \cos(k\omega_0 t + \varphi_k)$$

Az alapharmonikus frekvenciája a periódus idő reciproka:

$$\Delta f = \frac{1}{T_0} = f_0$$

- Az egyes spektrumvonalak is ilyen távolságban vannak egymástól.
- Az egyen-összetevő nagysága:  $B_0$
- Az egyen-összetevő negatív előjelét felfoghatjuk egy 180 fokos fázistolásnak is.
- A cos együtthatója az amplitúdó. A kezdőfázis szöge radiánban értendő!

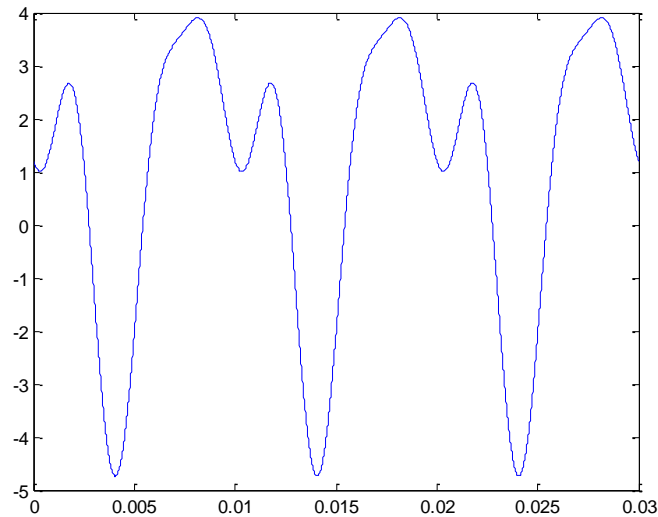
### Vegyünk egy egyszerű példát:

Az alábbi periodikus jel egy alapharmonikust és két felharmonikust tartalmaz.

$$x(t) = 1 + 3 \cos\left(2\pi * 1 * 100t + \frac{\pi}{4}\right) + \\ + 2 \cos\left(2\pi * 2 * 100t - \frac{3\pi}{4}\right) +$$

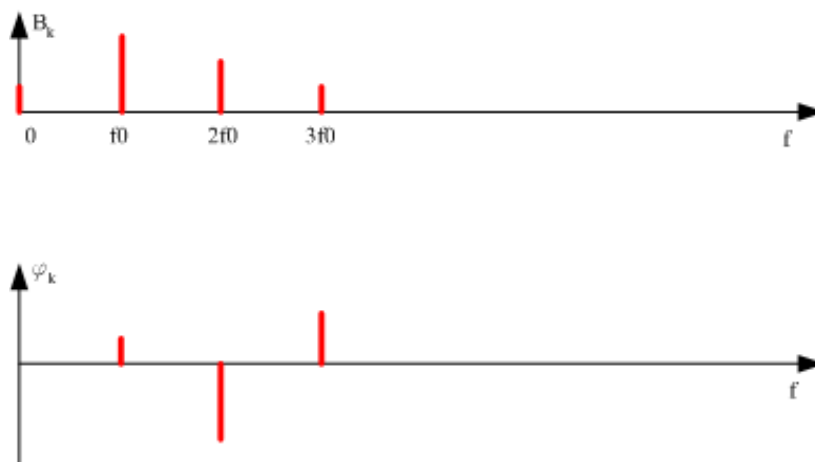
$$+ 1 \cos\left(2\pi * 3 * 100t + \frac{2\pi}{3}\right) [V]$$

Az időfüggvény látható a következő ábrán:



6-4. ábra. Időtartományi kép

A spektrumkép, amelyben az egyes összetevők amplitúdói és kezdőfázisai vannak ábrázolva a frekvencia függvényében:



6-5. ábra. Spektrumkép, abszolút érték és fázis

## 6.2.2 Fourier sor 2. alak

Ez a leírási mód a forgó vektorok vetületével történő megadáson alapul.

$$x(t) = \sum_{k=0}^{\infty} b_k \cos(\omega_k t) - \sum_{k=1}^{\infty} a_k \sin(\omega_k t)$$

$$\omega_k = 2\pi f_k; \quad \omega_k = k\omega_0; \quad f_k = kf_0$$

Az együtthatók kiszámításának módszere (részleteket ld. a matematika tárgyban):

$$b_0 = \frac{1}{T_0} \int_{-\frac{T_0}{2}}^{\frac{T_0}{2}} x(t) dt$$

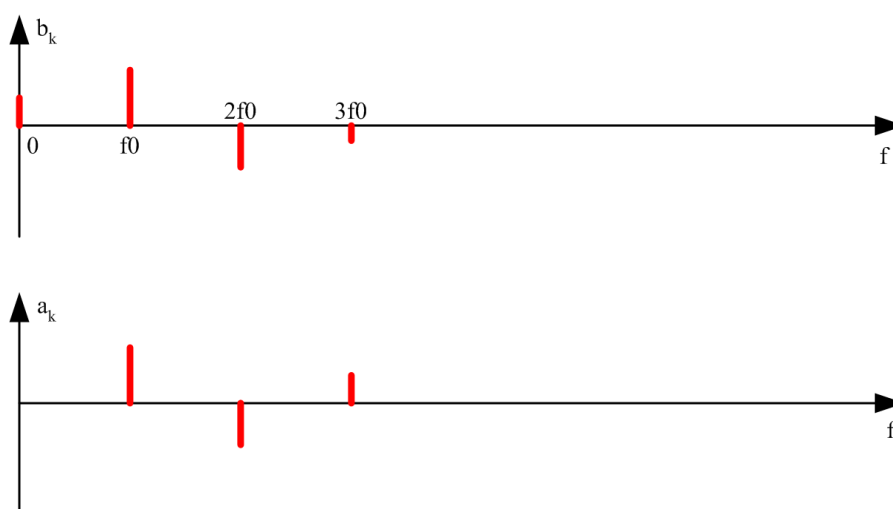
$$b_k = \frac{2}{T_0} \int_{-\frac{T_0}{2}}^{\frac{T_0}{2}} x(t) \cos(k\omega_0 t) dt$$

$$a_k = -\frac{2}{T_0} \int_{-\frac{T_0}{2}}^{\frac{T_0}{2}} x(t) \sin(k\omega_0 t) dt$$

Az előző példa ebben a leírási módban:

$$\begin{aligned} x(t) = & 1 + 2.12 \cos(2\pi * 1 * 100t) - 2.12 \sin(2\pi * 1 * 100t) - \\ & - 1.41 \cos(2\pi * 2 * 100t) + 1.41 \sin(2\pi * 2 * 100t) - \\ & - 0.5 \cos(2\pi * 3 * 100t) - 0.87 \sin(2\pi * 3 * 100t) \quad [V] \end{aligned}$$

Spektrumkép:



6-6. ábra. Spektrum ábra

### 6.2.3 Fourier sor 3. alak

Ez a leírási mód pedig a komplex forgó vektorok összegén alapul.

$$\begin{aligned} x(t) &= b_0 + \sum_{k=1}^{\infty} b_k \cos(\omega_k t) - \sum_{k=1}^{\infty} a_k \sin(\omega_k t) = \\ &= b_0 + \sum_{k=1}^{\infty} b_k \frac{e^{j\omega_k t} + e^{-j\omega_k t}}{2} - \sum_{k=1}^{\infty} a_k \frac{e^{j\omega_k t} - e^{-j\omega_k t}}{2j} = \\ &= b_0 + \sum_{k=1}^{\infty} \frac{b_k + ja_k}{2} e^{jk\omega_0 t} + \sum_{k=1}^{\infty} \frac{b_k - ja_k}{2} e^{-jk\omega_0 t} = \end{aligned}$$

$$= b_0 + \sum_{k=1}^{\infty} \bar{C}_k e^{jk\omega_0 t} + \sum_{k=1}^{\infty} \bar{C}_{-k} e^{-jk\omega_0 t}$$

$$|\bar{C}_k| = |\bar{C}_{-k}|; \quad \arg(\bar{C}_k) = -\arg(\bar{C}_{-k})$$

azaz a  $C_k$  és  $C_{-k}$  értékek egymás konjugáltjai.

Felismerhetjük, hogy a pozitív és a negatív frekvenciás tagok együtthatói:

$$\bar{C}_k = \frac{\bar{A}_k}{2}; \quad \bar{C}_{-k} = \frac{\bar{A}_k^*}{2}$$

Majd összerendezve felírhatjuk, talán a legfontosabb alakot:

$$x(t) = \sum_{k=-\infty}^{\infty} \bar{C}_k e^{jk\omega_0 t}$$

Az összetevők kiszámításának módszere:

$$\bar{C}_k = \frac{1}{T_0} \int_{-\frac{T_0}{2}}^{\frac{T_0}{2}} x(t) e^{-jk\omega_0 t} dt$$

Ebben a felírási módban benne van az egyen-összetevő és a különböző frekvenciáknak megfelelő komplex amplitúdó vektorok fele ( $C_k$ , amplitúdó és kezdőfázis információval), amelyek az exponens időfüggő kitevőjének köszönhetően más-más szögsebességgel, ellentétes irányokban forgó komplex konjugált vektor-párokat alkotnak.

Az előbbi példa ebben a formában:

$$x(t) = \sum_{k=-3}^3 \bar{C}_k e^{jk2\pi 100t} [V]$$

Azaz:

$$C_0 = 1$$

$$C_1 = 1.06 + j1.06$$

$$C_{-1} = 1.06 - j1.06$$

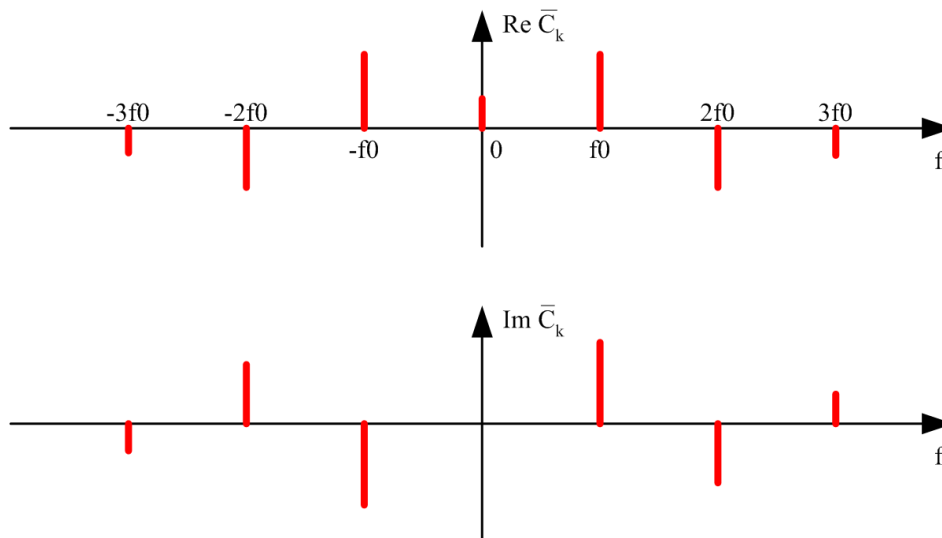
$$C_2 = -0.71 - j0.71$$

$$C_{-2} = -0.71 + j0.71$$

$$C_3 = -0.25 + j0.44$$

$$C_{-3} = -0.25 - j0.44$$

A spektrumkép a pozitív és negatív frekvenciákon is értelmezett:



6-7. ábra. 3. alak szerinti spektrumkép

A  $\text{Re}$  függvény mindig tengelyesen (páros függvény), az  $\text{Im}$  pedig középpontosan (páratlan) szimmetrikus. Így teljesülhet, hogy a megfelelő összetartozó pozitív és negatív frekvenciás tagok komplex konjugált párokat alkotnak. Ügyelni kell arra, hogy ezek a vektorok egyenként, a DC-t kivéve, csak fél amplitúdó hosszúságúak.

Ennek a spektrumábrázolásnak az alternatívája lehet, ha a komplex együtthatók abszolút értéke és a fázisa (argumentuma) kerül megjelenítésre.

#### 6.2.4 Átszámítási módok

A spektrumok különböző leírási formái átalakíthatók egymásba:

$$B_0 = |b_0| = |\bar{C}_0|$$

$$B_k = \sqrt{a_k^2 + b_k^2} = 2|\bar{C}_k|$$

$$\varphi_k = \arctan \frac{a_k}{b_k} = \arctan \frac{\text{Im}(\bar{C}_k)}{\text{Re}(\bar{C}_k)}$$

### 6.3 Feladatok

A spektrum vizsgálatokat MatLab szkriptek segítségével végezzük el. Nem szabad azonban elfelejteni, hogy a Fourier analízis számítógépes környezetben (amely lényegében diszkrét mintáiban tud kezelni idő és frekvencia függvényt egyaránt) sok buktatót rejt a gyakorlatlan felhasználó számára. Kövessük az utasításokat pontosan, így garantálható, hogy az eredmények szemléletesen és pontosan jelennek meg.

Vizsgáljunk meg néhány tipikus periodikus jelet! Hívjuk segítségül a `fourier_1.m...fourier_4.m` file-okat, amelyek képesek fogadni tetszőleges időfüggvényt és

képesek azt, valamint annak spektrumait megjeleníteni. A programok szemléletesen ábrázolják a komplex spektrum együtthatóinak vektorait is a komplex síkon.

A vizsgálatok minden pontjánál tudni kell, hogy az időfüggvények megadása egy  $t_f=5\text{ms}$ -os időintervallumban történik 256 függvényérték felvételével. A spektrum 200 Hz-enként ábrázol összetevőt 25,6 kHz-ig, így mindig figyelniünk kell, hogy az alapharmonikus 200 Hz egész számú többszöröse legyen.

### 6.3.1 Különböző jelalakok spektrumainak vizsgálata

1. *Nyissuk meg a fourier\_1.m programot, majd futtassuk!*
2. *Dokumentáljuk az időfüggvényt és a kapott spektrumképeket! Milyen leírási módoknak felelnek meg?*
3. *Változtassuk az időfüggvényt sin-ra fázistolással ( $-\pi/2$ ).*
4. *Hogyan változnak a spektrumképek?*
5. *Hogyan változik a vektorábra?*
6. *Figyeljük meg a komplex spektrum Re és Im részének páros ill. páratlan függvény jellegét!*
7. *Változtassunk a jel paraméterein (DC, Amplitúdó, frekvencia, kezdőfázis) és figyeljük meg a változásokat! Vigyázat! A frekvencia 200 Hz egész számú többszöröse legyen 25,6 kHz alatt!*

```
%Fourier analízis
%N: minták száma; fs: mintavételi frekvencia [Hz]; Tf: időablak [s]; td:
%mintavételi időköz 19.53e-6[s]; df: spektrum felbontása 200[Hz]
%A: amplitúdó; fi: fázis; DC: egyenfeszültség komponens; f: frekvencia
%Az időfüggvény megadása
N=256; fs=51.2e3; tf=5e-3; df=fs/N; td=tf/N;
t=0:td:tf-td; %időskála
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Vizsgált jel megadása
A=10; fi=0; DC=3; f=2000;
x=DC+A*cos((2*pi*f*t)+fi); %időfüggvény definíció
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
plot(t,x);
axis([0,5e-3,-20,20]);
grid;
xlabel('t [s]');
ylabel('U [V]');
title('Vizsgálandó jel időfüggvénye');
%Komplex spektrum számítása (Fourier sor 3. alak)
spl=(fft(x,256))/256; %FFT
sp2=fftshift(spl); %FFT igazítás
ffl=-fs/2:df:(fs/2)-df; %Frekvencia skála
figure;
subplot(2,1,1); stem(ffl,real(sp2),'.');
axis([-3e4,3e4,-10,10]);
grid;
xlabel('f [Hz]');
ylabel('Re(Ck)');
title('Komplex spektrum valós komponensei');
subplot(2,1,2); stem(ffl,imag(sp2),'.');
axis([-3e4,3e4,-10,10]);
grid;
xlabel('f [Hz]');
```



```

ylabel('Im(Ck)');
title('Komplex spektrum képzetes komponensei');
%Amplitúdó és fázisspektrum számítás (Fourier sor 1. alak)
sp3=sp1(1:128); %Csak a pozitív frekvenciák
sp3(2:128)=sp3(2:128)*2; %Valós amplitúdók negatív frekvenciák nélkül, DC nem szorzódik
ff2=0:df:(fs/2)-df; %Frekvencia skála
figure;
subplot(2,1,1); stem(ff2,abs(sp3),'.');
axis([0,3e4,0,15]);
grid;
xlabel('f [Hz]');
ylabel('Bk [V]');
title('Spektrum komponensek amplitúdója');
ang1=[];
for i=1:128;
    if abs(sp3(i))>0.0001;
        ang1(i)=angle(sp3(i));
    else
        ang1(i)=0;
    end
end
subplot(2,1,2); stem(ff2,ang1,'.');
axis([0,3e4,-pi,pi]);
grid;
xlabel('f [Hz]');
ylabel('fi [rad]');
title('Spektrum komponensek fázisa');
%Komplex spektrum vektoros ábrázolása
figure;
compass(sp2);
title('Komplex spektrum vektoros ábrázolása');

```

8. Cseréljük ki a „Vizsgált jel megadása” utáni jeldefiníciókat az alábbiakra, futtassuk a programot, majd válaszoljunk az alábbi kérdésekre!

Négyszög jel:

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Vizsgált jel megadása
A=10; f=400; %alapharmonikus
A1=4*A/pi
x=A1*cos(2*pi*f*t);
for felh=1:30 %felharmonikusok
    x=x+(A1*(-1)^felh)*(1/((2*felh)+1))*cos(2*pi*((2*felh)+1)*f*t);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

Háromszög jel:

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Vizsgált jel megadása
A=10; f=400; %alapharmonikus
A1=8*A/(pi^2)
x=A1*sin(2*pi*f*t);
for felh=1:30 %felharmonikusok
    x=x+(A1*(-1)^felh)*(1/((2*felh)+1)^2))*sin(2*pi*((2*felh)+1)*f*t);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

Fűrész jel:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Vizsgált jel megadása
A=10; f=400; %alapharmonikus
A1=-2*A/pi
x=A1*sin(2*pi*f*t);
for felh=1:60 %felharmonikusok
    x=x+(A1*(1/(felh+1))*sin(2*pi*(felh+1)*f*t));
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

9. *Azonosítsuk az alapharmonikusokat és a felharmonikusokat!*

Tisztán páros időfüggvény tisztán páros (cos) komponenseket tartalmaz, ilyenkor a spektrum csak valós komponenseket ábrázol. Tisztán páratlan időfüggvény csak tisztán páratlan komponenseket tartalmaz (sin, azaz 90 fokkal eltolt cos), ilyenkor a spektrum csak képzetes komponenseket ábrázol.

10. *Csökkentsük a spektrális komponensek számát drasztikusan az időfüggvény megadásánál (for ciklus számának változtatásával)!*

11. *Hogyan változtatja ez a jel alakját?*

12. *Hány komponensre lenne szükség az ideális jelalak előállításához?*

13. *Hogyan befolyásolja a jelalakot, ha viszonylagosan kicsi amplitúdójú komponenseket veszünk ki a spektrumból, és hogyan, ha a nagyokkal tesszük ugyanezt?*

Markerrel ellenőrizhetők a grafikonok függvényértékei. Használjuk a zoom funkciót! Vegyük figyelembe, hogy a fázisspektrumban a  $-\pi$  és a  $+\pi$  fázistolás ekvivalens, hisz ugyanazt az eredményt adja. A program ezeket látszólag véletlenszerűen fel is cseréli egymással. Figyeljük meg, hogy a spektrum abszolút értékének ábrázolásánál a negatív frekvenciás tagokat nem ábrázoltuk. Ebben az esetben a pozitív frekvenciás tagok abszolút értékét felszoroztuk kettővel, így kapván meg a tényleges amplitúdó értékeket.

### 6.3.2 Komplex Fourier sor értelmezése

Egy DC összetevő, egy alapharmonikus és két felharmonikus összegeként előállt általános periodikus jel alkotja a vizsgálat tárgyát.

1. *Futtassuk a mellékelt fourier\_5.m programot! Az előtűnő animációt érdemes teljes képernyőn vizsgálni.*

Látható, ahogyan a komplex spektrum együtthatói felveszik a kezdő állapotukat ( $C_k$ ). Egy harmonikus rezgéshez két komplex konjugált, fél-fél amplitúdójú vektor tartozik. A három harmonikushoz tehát hat vektor dukál, plusz egy a reális tengelyre eső DC komponens. A komplex vektorok összege pirossal jelölt. Mivel a vektorpárok minden időpillanatban konjugált párokat alkotnak, így az összegük szükségképpen minden időpillanatban a reális tengelyre esik. Ahogy az idő telik, úgy a vektorpárok a saját szögsebességükkel elfordulnak. A pozitív körüljárási irány az óramutató járásával ellentétes. A negatív frekvenciás tagok pedig ellentétesen indulnak el, mert az idővel a negatív argumentum növekszik. Az első felharmonikusnak kétszer, a másodiknak háromszor akkora az  $\omega$  szögsebessége, mint az alapharmonikusé. Ez is jól követhető az animáción. Az időtengelyen ábrázolva az eredő valós vektor nagyságát, visszarajzolódik a kiinduláskor megadott időfüggvény!

2. *Értelmezzük és dokumentáljuk az időfüggvényt, a spektrumképeket, és az animációt!*
3. *Az előbbieken említett korlátok között változtassunk a bemenő jelen, és követhetjük a változásokat!*

### 6.3.3 Négyyszögimpulzus spektrumának vizsgálata

Egy periodikus négyyszögjel spektruma vonalas, és egy  $\sin(x)/x$  burkoló határozza meg a spektrum vonalak nagyságát. E miatt a jel spektruma végtelen kiterjedésű, de az alkalmazások java részénél elég a fentebbi spektrumösszetevőket csak az első zérus helyig figyelembe venni. Ezt a pontot hívjuk konvencionális sávszélességnek és ez a négyyszögjel kitöltési tényezőjétől függ, az impulzus szélességéből számítható, azaz  $1/\Delta t$ .

Ezek után vizsgálódjunk. Használjuk a korábbi kódot, de a négyyszögjelet a lentebb megadott paraméterek alapján állítsuk elő, vagy használjuk a mellékelt .mat fájlokat!

1. *A bemenő jelsorozat legyen egy 400Hz-es,  $1/4$  kitöltési tényezőjű négyyszög impulzus sorozat. Ennek időfüggvény-mintái az `x_1.mat` file-ban található. Töltsük be a Workspace-en, és futtassuk ezzel a `fourier_6.m` programot.*
2. *Azonosítsuk be az alapharmonikust és a felharmonikusokat!*
3. *Keressük meg a képzeletbeli  $\sin(x)/x$  burkoló első zérushelyét! Használjuk a zoom funkciót!*
4. *Dokumentáljuk az eredményeket, különösen a konvencionális sávszélességet!*

Itt a komplex spektrum abszolút értéke és fázisa is ábrázolásra kerül, hogy a kapott eredmények összevethetők legyenek.

5. *Csökkentsük az impulzusok szélességét a felére, azaz 1/8-ra! Az időfüggvény minták az x\_2.mat file-ban találhatóak. Töltsük be, és ezzel futtassuk a programot!*
6. *Nézzük meg, hogy mi a következménye a sávszélességre vonatkozólag.*
7. *Vessük össze az előzőleg használt jel eredményével!*
8. *Most a 400Hz-es, 1/4 kitöltési tényezőjű négyszög impulzus sorozathoz képest növeljük az impulzus szélességét a kétszeresére. Így 1/2 kitöltési tényezőt kaptunk. Az időfüggvény minták az x\_3.mat file-ban találhatóak. Töltsük be, és ezzel futtassuk a programot!*
9. *Nézzük meg, hogy mi a következménye a sávszélességre vonatkozólag. Vessük össze a korábbi eredményekkel!*

Láthatjuk, hogy a sávszélesség a felére csökkent. Az 1/2 kitöltési tényezőjű négyszögjel sorozatnak csak a páratlan harmonikusai léteznek, mivel a  $\sin(x)/x$  burkológörbe zérushelyei épp a páros harmonikusokra esnek.

10. *Állítsunk be az impulzusok szélességét ismét 1/8-ra, de növeljük a periódusidőt a kétszeresére ( $f=200$  Hz), azaz távolítsuk el egymástól az impulzusokat. Az időfüggvény minták az x\_4.mat file-ban találhatóak. Töltsük be, és ezzel futtassuk a programot!*
11. *Nézzük meg, hogy mi a következménye a sávszélességre vonatkozólag.*
12. *Vessük össze a korábbi eredményekkel 2. pont és a számpélda eredményével!*

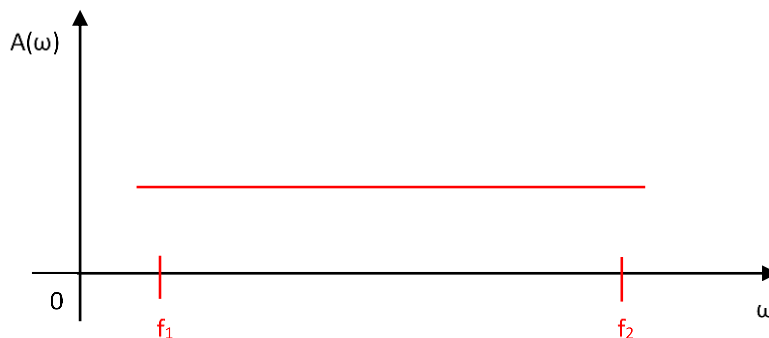
Láthatjuk, hogy a sávszélesség nem változott, viszont a spektrumvonalak „besűrűsödtek”, hiszen az alapharmonikus frekvenciája lecsökkent épp a felére. A komponensek abszolút értékei viszont lecsökkentek. Látjuk tehát, hogy valóban a sávszélesség az impulzusok szélességétől függ, de a periódusidőtől nem. Extrém esetben, az impulzusokat végtelenül eltávolítva egymástól egy magában álló impulzust kapunk, amelyre az lesz a jellemző, hogy a spektrális sávszélesség nem változik, de a spektrum végtelenül besűrűsödik és a komponensek egyenként végtelenül picinyek lesznek, azaz a spektrum folytonossá válik.

13. *Most a 400Hz-es, 1/8-ad kitöltési tényezőjű jelünket késleltessük 62,5 $\mu$ s-ot. A minták az x\_5.mat file-ban találhatóak. Töltsük be, és ezzel futtassuk a programot!*
14. *Figyeljük meg, hogy az amplitúdó spektrum nem változott!*
15. *Milyen változás tapasztalható a fázisspektrumban, hogyan változik a késleltetés nélküli állapothoz képest?*

## 7 Lineáris torzítás hatásainak vizsgálata

A lineáris torzítatlanság feltételei a következők, amelyeknek az adott rendszer hasznos sávjában kell teljesülniük (a rendszer hasznos sávja az a véges sávtartomány a frekvencia tengelyen, ahol a kezelt jelünk található):

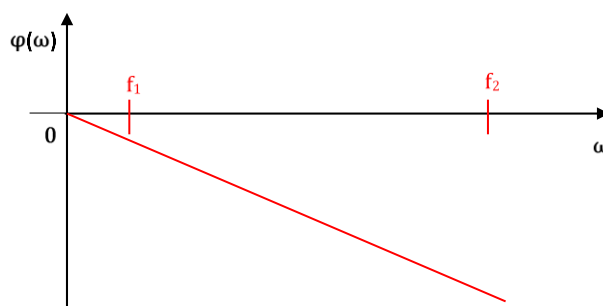
1. Az átviteli karakterisztika abszolút értéke legyen konstans. Ez alapján a rendszerünk az átvitt jelkomponensek minden frekvenciájú komponensét azonos mértékben csillapítja vagy erősíti.



7-1. ábra. Amplitúdó karakterisztika

2. A fáziskarakterisztika 0-ból induló lineáris egyenes legyen. Az a fontos, hogy akár pozitív, akár negatív a fázistolása a hálózatnak, az egyes frekvencia-összetevők azonos idővel tolódjanak el, hogy a kimeneten összegződve ugyanazt a jelalakot írják le, mint a bemeneten volt. Például, ha egy rendszer fázist késleltet, akkor a torzítatlanság úgy teljesül, ha minden spektrumösszetevő időkésleltetése ugyanaz  $-\tau'$ .

$$y(t) = A_y * \cos \{ \omega (t - \tau') + \varphi_x \}$$
$$-\omega \tau' = \varphi(\omega)$$



7-2. ábra. Lineáris fáziskarakterisztika

3. A csoportfutási időnek konstansnak kell lennie, ami a fáziskarakterisztika feltételéből következik, mivel a csoportfutási idő a fáziskarakterisztika meredeksége, azaz deriváltja.

$$\tau(\omega) = \frac{d\varphi(\omega)}{d\omega}$$

## 7.1 Feladatok

Adott egy periodikus PCM-AMI jel időfüggvénye a jel Fourier sorának első 32 tagjának összegével. Mivel csak az első 32 tagot vesszük figyelembe az ideális végtelen helyett, az impulzusok nem ideálisak. Fourier sor összetevői:

$$\begin{aligned} x(t) = & +[2/\pi] * \sin(2\pi ft + \varphi_1) & +0 & +[2/3\pi] * \sin(6\pi ft + \varphi_3) & +0 \\ & -[2/5\pi] * \sin(10\pi ft + \varphi_5) & +0 & -[2/7\pi] * \sin(14\pi ft + \varphi_7) & +0 \\ & +[2/9\pi] * \sin(18\pi ft + \varphi_9) & +0 & +[2/11\pi] * \sin(22\pi ft + \varphi_{11}) & +0 \\ & -[2/13\pi] * \sin(26\pi ft + \varphi_5) & +0 & -[2/15\pi] * \sin(30\pi ft + \varphi_7) & +0 \\ & +[2/17\pi] * \sin(34\pi ft + \varphi_9) & +0 & +[2/19\pi] * \sin(38\pi ft + \varphi_{11}) & +0 \\ & -[2/21\pi] * \sin(42\pi ft + \varphi_5) & +0 & -[2/23\pi] * \sin(46\pi ft + \varphi_7) & +0 \\ & +[2/25\pi] * \sin(50\pi ft + \varphi_9) & +0 & +[2/27\pi] * \sin(54\pi ft + \varphi_{11}) & +0 \\ & -[2/29\pi] * \sin(58\pi ft + \varphi_5) & +0 & -[2/31\pi] * \sin(62\pi ft + \varphi_7) & +0 \end{aligned}$$

Ha a jel áthalad egy rendszeren, ott a  $K(j\omega)$  átviteli karakterisztikának megfelelően lineáris torzítást szenvedhet. Torzításmentes esetben biztosítható az alakhű jelátvitel, még ha a jel erősítést (csillapítást), vagy késleltetést szenvedett is. Az alábbiakban a lineáris torzítás hatását vizsgáljuk.

1. *Rajzoltassuk ki a jel időfüggvényét! Határozzuk meg az alapharmonikust és a felharmonikusokat! Alapesetben az eredő PCM-AMI jel tisztán páros időfüggvény. A Fourier sor összetevőinek fázisai rendre 0.*
2. *Vizsgáljuk meg, hogy milyen hatással van a torzítatlanságra, ha pl. kétszeresére megváltoztatjuk az összes komponens amplitúdóját. Utána állítsuk vissza az eredeti jelet!*
3. *Vizsgáljuk meg, hogy milyen hatással van a torzítatlanságra, ha pl. kétszeresére változtatjuk a legnagyobb frekvenciás komponens amplitúdóját.*

4. *Próbáljuk meg megváltoztatni csak az alapharmonikus amplitúdóját ugyanilyen mértékben. Mit tapasztalunk? Utána állítsuk vissza az eredeti jelet!*
5. *Változtassuk meg az alapharmonikus, majd a legmagasabb frekvenciás komponens fázisát, külön-külön, ugyanakkora mértékben. Minden változtatás után állítsuk vissza az eredeti állapotot. Írjuk le tapasztalatokat!*
6. *Most módosítsuk úgy az időfüggvényt, hogy a jel késleltetése 0 helyett  $T/4$  legyen (ahol  $f=n \times 1/T$ ). Ehhez minden komponens fázisát a következőképpen kell módosítani:*

$$\varphi_n = -n\pi/2 \quad (p_1 \dots \varphi_9 = 0 \rightarrow \varphi_9' = -9\pi/2)$$

7. *Határozzuk meg az új fázisokat (minden komponens esetén más fázistolást kell kapnunk a lineáris fáziskarakterisztika szerint, ami konstans késleltetés mellett lineáris torzítatlanságot eredményez). Rajzoljuk fel az időfüggvényt és figyeljük meg az eredő jel késleltetését (a nullfázisú állapothoz viszonyítva).*

### 7.1.1 Programkód

```
clear;
f=1000;
fi= [0 0 0 0 0 0 0 0 0 0 0 0 0 0];
%fi= [-pi/2 -3*pi/2 -5*pi/2 -7*pi/2 -9*pi/2 -11*pi/2 -13*pi/2 -15*pi/2 -17*pi/2 -19*pi/2 -
21*pi/2 -23*pi/2 -25*pi/2 -27*pi/2 -29*pi/2 -31*pi/2];
A=[2/pi 2/(3*pi) -2/(5*pi) -2/(7*pi) 2/(9*pi) 2/(11*pi) -2/(13*pi) -2/(15*pi) 2/(17*pi)
2/(19*pi) -2/(21*pi) -2/(23*pi) 2/(25*pi) 2/(27*pi) -2/(29*pi) -2/(31*pi)];
%A=[4/pi 4/(3*pi) -4/(5*pi) -4/(7*pi) 4/(9*pi) 4/(11*pi) -4/(13*pi) -4/(15*pi) 4/(17*pi)
4/(19*pi) -4/(21*pi) -4/(23*pi) 4/(25*pi) 4/(27*pi) -4/(29*pi) -4/(31*pi)];
t1=[0:0.000001:0.0025];
for a=1:1:2500
t(a)=t1(a);
x1=A(1)*cos(2*pi*f*t+fi(1))+A(2)*cos(6*pi*f*t+fi(2))+A(3)*cos(10*pi*f*t+fi(3))+A(4)*cos(14*pi*
f*t+fi(4));
x2=A(5)*cos(18*pi*f*t+fi(5))+A(6)*cos(22*pi*f*t+fi(6))+A(7)*cos(26*pi*f*t+fi(7))+A(8)*cos(30*pi
f*t+fi(8));
x3=A(9)*cos(34*pi*f*t+fi(9))+A(10)*cos(38*pi*f*t+fi(10))+A(11)*cos(42*pi*f*t+fi(11))+A(12)*cos
(46*pi*f*t+fi(12));
x4=A(13)*cos(50*pi*f*t+fi(13))+A(14)*cos(54*pi*f*t+fi(14))+A(15)*cos(58*pi*f*t+fi(15))+A(16)*c
os(62*pi*f*t+fi(16));
x=x1+x2+x3+x4;
end
plot(t, x);
```

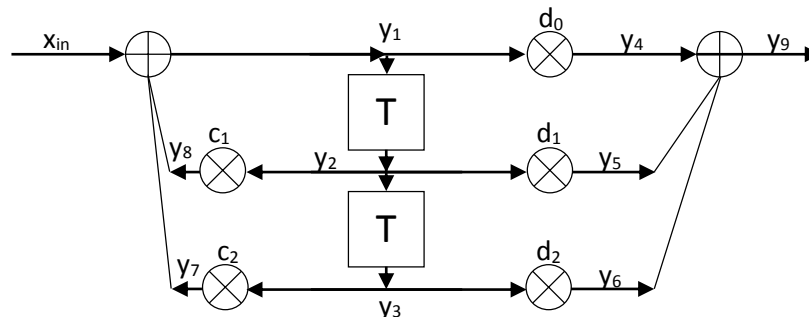
## 8 Diszkrét idejű hálózatok szimulációja

A diszkrét idejű hálózatok szimulációja az egyre nagyobb teret hódító digitális jelfeldolgozásnak (DSP) köszönhetően legalább annyira fontos, mint a folytonos idejű rendszereké. Ebben a fejezetben e témakört vizsgáljuk.

### 8.1 Elméleti áttekintés

Lineáris, időben invariáns diszkrét idejű rendszereket gyakran az úgynevezett állapotegyenleteivel írjuk le. Állapotváltozókat általában az együtemnyi késleltetők bemenetén vagy kimenetén vesszük fel. Vizsgálataink szempontjából előnyösebbnek tűnik, ha az összes építőelem kimenetén megjelenő számértékeket minden  $k$ -adik ütemre meghatározzuk. Tehát az állapotváltozós analízistől eltérően a hálózatot a csomópontok jeleivel írjuk le, azaz csomóponti analízist végzünk, ez, a hálózat mélyebb vizsgálatát teszi lehetővé.

Csomópont a hálózat bármely pontján felvehető. A 8-1. ábra példáján az analízishez kilenc csomópontot vettünk fel. Ezek közül az ❶ összevont összegző - elágazó csomópont, a ❷ és ❸ elágazó, míg a ❹ összegző csomópont. Ezenkívül a szorzások hatásának vizsgálatához a konstans szorzók kimenetén vettük még fel ❺❻❼❼❼❸ csomópontot.



8-1. ábra. Másodfokú alaptag (IIR szűrő)

Az összes csomópontra felírva az egyenleteket:

$$y1(k) = xin(k) + y7(k) + y8(k)$$

$$y2(k) = y1(k-1)$$

$$y3(k) = y2(k-1)$$

$$y4(k) = d0 \cdot y1(k)$$

$$y5(k) = d1 \cdot y2(k)$$

$$y6(k) = d2 \cdot y3(k)$$

$$y7(k) = c2 \cdot y3(k)$$



$$y_8(k) = c_1 \cdot y_2(k)$$

$$y_9(k) = y_{out}(k) = y_4(k) + y_5(k) + y_6(k).$$

Ahol  $y_i(k)$  az  $i$ -edik csomópont  $k$ -adik ütembeli jele. Majd az egyenleteket a következő szemléletes formába rendezhetjük:

$$\begin{aligned} y_1(k) - y_7(k) - y_8(k) &= x_{in}(k) \\ y_2(k) &= y_1(k-1) \\ y_3(k) &= y_2(k-1) \\ -d_0 \cdot y_1(k) + y_4(k) &= 0 \\ -d_1 \cdot y_2(k) + y_5(k) &= 0 \\ -d_2 \cdot y_3(k) + y_6(k) &= 0 \\ -c_2 \cdot y_3(k) + y_7(k) &= 0 \\ -c_1 \cdot y_2(k) + y_8(k) &= 0 \\ -y_4(k) - y_5(k) - y_6(k) + y_9(k) &= 0. \end{aligned}$$

Az egyenletrendszert mátrix alakban felírva:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -d_0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -d_1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -d_2 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & -c_2 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & -c_1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 & -1 & -1 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} y_1(k) \\ y_2(k) \\ y_3(k) \\ y_4(k) \\ y_5(k) \\ y_6(k) \\ y_7(k) \\ y_8(k) \\ y_9(k) \end{bmatrix} = \begin{bmatrix} x_{in}(k) \\ y_1(k-1) \\ y_2(k-1) \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Vagyis az  $\underline{\underline{M}} \cdot y = g$  alakot kapjuk.

A  $k$ -adik ütemben az összes csomóponton megjelenő számérték az

$$\underline{\underline{M}}^{-1} \cdot g = y$$

alapján számolható, ahol  $\underline{\underline{M}}^{-1}$  az  $\underline{\underline{M}}$  mátrix inverze.

Az  $\underline{M}$  mátrix felírása igen egyszerűen mechanizálható a következő szerint:

- A főátlóba mindig 1 kerül.
- A mátrix összes többi elemét megkapjuk, ha minden sorhoz és oszlophoz egy-egy csomópontot rendelünk, majd a jelfolyamgráfon megnézzük, hogy a jelfolyam-nyíllal ellentétes irányban haladva mely csomópontból melyikbe tudunk közvetlenül eljutni.

Ha valamely csomópontból valamelyikbe közvetlenül nem tudunk eljutni – a fent említett módon – akkor a mátrix ezen sor-oszlop pozíciójába 0-át, ha pedig közvetlenül el tudunk jutni, akkor ide annak a konstans szorzónak a mínusz egyszeresét írjuk, melyen keresztül eljutottunk egyik csomópontból a másikba.

A fentiek alapján – jelen példában – az  $M$  mátrix első sora:

$$1 \ 0 \ 0 \ 0 \ 0 \ 0 \ -1 \ -1 \ 0,$$

vagyis a főátlóba 1 került, a 7. és a 8. oszlop pozícióba -1, mivel a jelfolyam-nyíllal ellentétes irányban haladva a 7. és 8. csomópontba 1 konstans szorzón keresztül juthatunk.

A  $g$  oszlopvektor elemeit a gerjesztő jelek adják, amik a csomópontok bemenő jelei. Gerjesztő jelnek a  $k$ -adik ütemben a  $k$ -adik bemenőjel  $\{x_{in}(k)\}$ , valamint a letárolt minták  $\{y_{n(k-1)}\}$  számítanak.

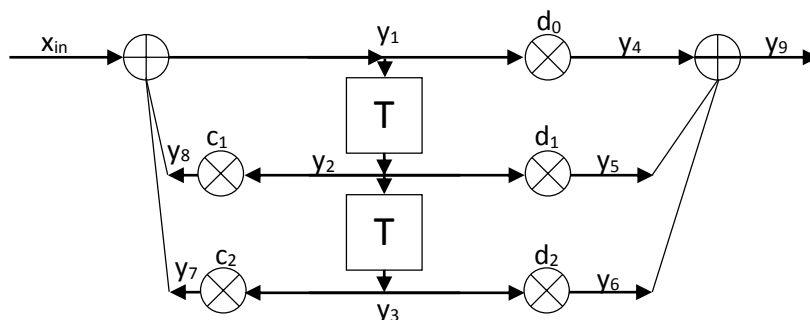
Jelen példában az "1." csomópont forrása az  $x_{in}(k)$ , a "2." csomóponté az  $y_1(k-1)$ , a "3." csomóponté pedig  $y_2(k-1)$ . A többi csomópontra nézve 0 adódik, így a  $g$  többi eleme nulla.

A fentiekben bemutatott lineáris analízis az átviteli jellemzők meghatározásán kívül alkalmas a túlsordulási pontok kimutatására. Minden ütemben meghatározhatjuk az összes csomópont jelét, így ha a valamely számérték abszolút értéke a megengedett fölé növekszik (ábrázolási tartományon kívül esik), akkor ez kijelezhető. A jelfeldolgozó hálózatunk kimenő mintáit az  $y_9$  vektor fogja tartalmazni.

## 8.2 Feladatok

### 8.2.1 Szűrőanalízis állapotváltozókkal

Nyissuk meg az *iir1.m* MatLab programot, mely elvégzi a következő jelfeldolgozó hálózat időtartomány analízisét:



8-2. ábra. Másodfokú IIR struktúra

A szorzó konstansok a következők:

$$c1 = 0.98650792405655952$$

$$c2 = -0.44679329164515174;$$

$$d0 = 1$$

$$d1 = 2$$

$$d2 = 1$$

Gerjesztő jelnek alkalmazzunk Kronecker-deltát (Dirac – delta mintavételes megfelelője, melyet, ha egy vektorban tárolunk, akkor a vektor összes eleme nulla kivéve az első elemet, ami egy.)

Minden csomóponthoz tartozó k-adik mintát a csomóponthoz rendelt vektorban tároljuk.

Az analízist végezzük el 8000 ütemre! A vizsgálathoz írjunk ciklust, az eredményeket és részeredményeket csomópontokhoz tartozó vektorokban tároljuk. Valójában s súlysorozat diszkrét Fourier transzformációja (DFT ill. FFT) adja az átviteli karakterisztika diszkrét mintáit. Ne feledjük, hogy az átviteli karakterisztika folytonos függvény, így a mintákat összekötve ez a függvény jól közelíthető.

Az időtartomány analízist, ha a fenti formában végezzük el, akkor a jelfeldolgozó hálózat összes megjelölt (felvett) csomópontján előálló eredményeket meg tudjuk vizsgálni. A módszer (mint azt az elméleti összefoglalóban áttekintettük) számítógéppel hatékonyan automatizálható, akár grafikusam megrajzolt jelfolyam-diagramból is.

## Programkód

```
%Másodfokú IIR struktúrájú aluláteresztő szűrő analízis
%vizsgált ütemszám:
N=8000;
%előre csatolás szorzó konstansok
d0=1;d1=2;d2=1;
%viSSzacsatolás szorzó konstansok
```

```

c1=0.98650792405655952;
c2=-0.44679329164515174;
%IIR hálózatot leíró mátrix
M=[1, 0, 0, 0, 0, 0, -1, -1, 0;
    0, 1, 0, 0, 0, 0, 0, 0, 0;
    0, 0, 1, 0, 0, 0, 0, 0, 0;
    -d0, 0, 0, 1, 0, 0, 0, 0, 0;
    0, -d1, 0, 0, 1, 0, 0, 0, 0;
    0, 0, -d2, 0, 0, 1, 0, 0, 0;
    0, 0, -c2, 0, 0, 0, 1, 0, 0;
    0, -c1, 0, 0, 0, 0, 0, 1, 0;
    0, 0, 0, -1, -1, -1, 0, 0, 1];
Mi=inv(M);
%csomópontok eredményeit tároló vektorok előállítása és nullázása
y1=zeros(1,N);
y2=zeros(1,N);
y3=zeros(1,N);
y4=zeros(1,N);
y5=zeros(1,N);
y6=zeros(1,N);
y7=zeros(1,N);
y8=zeros(1,N);
y9=zeros(1,N);

%k-adik ütem kimeneti eredményeit átmenetileg tároló vektor definíciója:
Y=zeros(1,9);

%Gerjesztő jel definíció:
Xin=zeros(1,N);
Xin(2)=1; %Kronecker delta

for k=2:N; %ütemező ciklus
    g=[Xin(k), y1(k-1), y2(k-1), 0, 0, 0, 0, 0, 0]';

    Y=Mi*g; %egyenletrendszer megoldás a k-adik ütemben

    %megoldások eltárolása a csomópontokhoz rendelt vektorokba:
    y1(k)=Y(1);
    y2(k)=Y(2);
    y3(k)=Y(3);
    y4(k)=Y(4);
    y5(k)=Y(5);
    y6(k)=Y(6);
    y7(k)=Y(7);
    y8(k)=Y(8);
    y9(k)=Y(9);
end;

```

1. A programkódból keressük ki, hány lépést (ütemet) fogunk a kód futtatásakor kiszámoltatni!
2. Futtassuk a programot!
3. A programkódban melyik változó felel meg a bemenetnek? Írjuk le a jegyzőkönyvbe ennek a vektornak az első 10 elemét!
4. Melyik változó felel meg a kimenetnek?
5. Jelenítsük meg a kimenet értékeit a plot függvénnyel. Zoom segítségével keressük meg hányadik ütemig van látható kitérés a kimeneten?
6. A program futtatása után jelenítsük meg mind a kilenc változó értékeit egy ábrában a plot függvénnyel.

7. Melyik változó veszi fel a legnagyobb szélsőértékeket és mennyi az?
8. A mátrix alakban felírt megoldás (első mintaanalízis) esetén, mivel minden felvett csomópont minden ütemben számított értékét eltároltuk, ezért hatékonyan tudjuk ellenőrizni a DSP áramkör esetleges „túlvezérléseit”, túlsordulásait az időtartományban. Az egyes csomópontokban előálló eredmények ellenőrzésével meghatározhatjuk a feldolgozó hálózat azon pontjait, amelyek a leggyengébbek a kivezérelhetőség szempontjából!
9. Jelenítsük meg a kimenet impulzusválaszának Fourier transzformáltját az alábbi paranccsal:  
  

```
plot(20*log10(abs(fft(y9))));grid;title('FFT');
```
10. Milyen jellegű szűrőt valósít meg a rendszer? (0..4000-ig tartó  $x$  tengely szerinti tartományt vegyük alapul!)

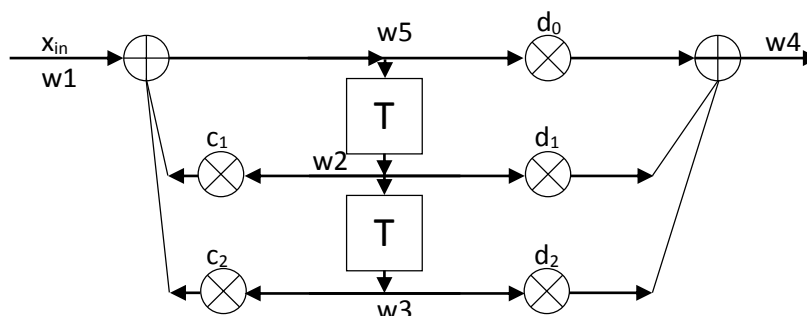
### 8.2.2 Szűrőanalízis redukált számú állapotváltozóval

Az előző módszer hátránya, a sok „felesleges” művelet elvégzése, ami a tényleges realizálásnál nem megengedhető, hiszen ott a rendszernek valós időben kell működnie és a kimeneti minták kiszámítására mindössze  $1/f_m$  idő áll rendelkezésünkre.

A megvalósításnál tehát arra kell törekednünk, hogy az egyenletrendszernek a lehető legegyszerűbb alakját oldjuk meg!

A MatLab program készítésénél is követhetjük ezt az irányelvet, mely nagyon hasznos lehet DSP program hibakeresésnél. Sok esetben egy DSP program működik ugyan, de nem a várt eredményeket hozza! Ehhez elegendő egy előjel elírás.

A következő kódban (iir2.m) csupán öt változót vettünk fel ( $w1..w5$ ), melyek az alábbi ábrán a jelölt csomópontok értékeit tartalmazzák.



8-3. ábra. Redukált számú állapotváltozó felvétele

## Programkód:

```
%Másodfokú IIR szűrő analízise 2
%vizsgált ütemszám:
N=8000;
%előre csatolás szorzó konstansok
d0=1;d1=2;d2=1;
%visszacsatolás szorzó konstansok
c1=0.98650792405655952;
c2=-0.44679329164515174;
%c1=0.99;
%c2=-0.45;

%=====
%állapotváltozók és átmeneti táruk

%bemeneti vektor
w1=zeros(1,N); %N elemű sorvektor
w1(1)=1; %Kronecker delta gerjesztés (vektor első eleme 1, többi 0)

%tárolók
w5=zeros(1,N); %átmeneti tároló - 1-es csomópont
w2=zeros(1,N); %első IIR tároló - 2-es csomópont
w3=zeros(1,N); %második IIR tároló - 3-as csomópont
w4=zeros(1,N); %kimenet - 4-es csomópont

for k=1:N

w5(k)=w1(k)+w2(k)*c1+w3(k)*c2; %átmeneti adat (ütem végén ezt kell betárolni)
w4(k)=w2(k)*d1+w5(k)*d0+w3(k)*d2; %kimeneti minta az n-edik ütemben
                                     %9-es csomópont („kimenet”)

%betárolás a következő ütemhez
w3(k+1)=w2(k);
w2(k+1)=w5(k);
%-----
end;
```

1. A programkódból keressük ki, hány lépést (ütemet) fogunk a kód futtatásakor kiszámoltatni!
2. Futtassuk a programot!
3. A programkódban melyik változó felel meg a bemenetnek? Írjuk le a jegyzőkönyvbe ennek a vektornak az első 10 elemét!
4. Melyik változó felel meg a kimenetnek? Az előző feladatban ezt melyik változó reprezentálta?
5. Jelenítsük meg a kimenet értékeit a plot függvénnnyel. Zoom segítségével keressük meg hányadik ütemig van látható kitérés a kimeneten?
6. Tapasztaltunk különbséget a két kód futtatásának eredményei között?

### 8.2.3 FFT ábra felskálázása

Kiindulásképpen az FFT ábra x tengelyén a minták számát látjuk csak (8000), mivel az FFT N db időtartományi mintából (most N=8000 súlysorozat minta) N db frekvenciatartományi

mintát számol. A rendszerben jelenleg nincs megadva a mintavételi frekvencia, nem tudni, hogy egy állapot mennyi ideig tart.

Legyen a mintavételi frekvencia, **fs=48 kHz**. Ekkor a mintavételi idő  $td=1/fs$ . Az FFT ábrán az x tengelyen 1 egység,  $df=fs/N$  frekvenciát jelent (6 Hz).

1. *Hozzuk létre az alábbi változókat a megadott értékekkel:*

```
fs=48e3;  
df=fs/N;  
scale=1:df:df*N;
```

2. *Rajzoltassuk ki ismét az FFT ábráját az alábbi paranccsal, így már frekvenciaskálázott ábrát kapva!*

```
plot(scale, 20*log10(abs(fft(w4)))); grid; title('súlysorozat    Fourier  
transzformáltja');
```

3. *Állapítsuk meg, hogy milyen frekvencia érték felett csillapít a szűrő? Olvassuk le a szűrő átvitelét 100, 10000, 20000 Hz-en!*

Ne feledjük, hogy a frekvencia-átviteli karakterisztikát ugyan 6 Hz-enként, diszkrét pontokon számítjuk, de maga a függvény minden frekvencián értelmezhető, folytonos függvény, ezért a pontokat összeköthetjük, jól közelítve a valóságos görbét.

#### 8.2.4 Paraméter érzékenység

Vizsgáljuk meg a szűrő szorzó konstansok paraméter érzékenységét! Ez a vizsgálat azért fontos, mert a leendő megvalósításban a bitszám mindig adott. Alacsonyabb bitszám esetén a kiszámított konstansok nem ábrázolhatók pontosan, kérdés, hogy ennek hatása mennyire rontja el a szűrő karakterisztikáját. Ha találunk olyan konstansokat, melyek értékére kevésbé érzékeny a rendszer, akkor ezek értékét kettő hatványainak választva jelentős sebességnövekedést érhetünk el bizonyos szorzás vagy osztás műveletnél.

1. *Először a programkód elején található c1, c2 változók értékeit kerekítsük! A kódban található kommentezett sorokat használjuk megfelelően ehhez!*
2. *Vizsgáljuk meg a kimenet értékeit, hogy történt-e jelentős változás (max érték, lecsengés hossza)? Vizsgáljuk meg a karakterisztika menetét is!*

```
plot(scale, 20*log10(abs(fft(w4)))); grid; title('FFT');
```

3. *Vizsgáljuk meg, hogy mennyire gerjedékeny a hálózatunk! változtassuk a visszacsatoló ágak konstansainak értékét ( $c_1$ ,  $c_2$ ) úgy, hogy gerjedjen be a hálózat. Mi lesz ennek a következménye?*

### 8.2.5 Szinuszos vizsgálat

Vizsgáljuk meg az áramkört szinuszos gerjesztő jellel is.

1. *A  $w1=zeros(1,N)$ ;  $w1(1)=1$ ; sorokat kommentezzük ki, és adjunk meg egy 100 Hz-es szinuszos jelet bemenő jelként:*

```
td=1/fs;  
t=0:td:(N-1)*td;  
A=1; fi=0; DC=0; f=100;  
w1=DC+A*sin(2*pi*f*t+fi);      %szinusz
```

2. *Az időtartományi bemenő és kimenő jelet vizsgálva határozzuk meg, hogy mennyi az erősítés, csillapítás 100, 10000, 20000 Hz-en. Vessük össze az eredményt a korábbi feladat megfelelő pontjával!*

Fentebbi mérési feladatban a frekvencia-átviteli karakterisztika ábráján a 0 dB-hez tartozó frekvenciát meghatároztuk már.

3. *Adjuk a bemenetre a leolvasott frekvenciájú egységnyi amplitúdójú jelet! A kimeneten ugye szintén egységnyi amplitúdójú a szinusz jel?*
4. *Próbáljuk ki SimuLinkban a rendszer működésének vizsgálatát! Nyissuk meg a iirsimu.mdl fájlt. Hozzunk létre egy simin nevű sorvektort 8000 elemmel, tele 0-kal, majd az első elemét cseréljük ki 1-re (Kronecker delta). Indítsuk a szimulációt az eredmény a simout nevű változóba kerül. Hasonlítsuk össze a kimeneten kapott eredményeket a mérés megfelelő feladatainak eredményeivel! A szükséges parancsok:*

```
N=8000;  
simin=zeros(1,N);  
simin(1)=1;
```



## 9 FIR szűrők tervezése

A FIR (Finite Impulse Response) struktúra, a DSP direkt struktúrák egyik alap építőelemének tekinthető. Strukturális stabilitása és kellemes fázisjellemzői miatt bizonyos szakterületeken nélkülözhetetlen jelfeldolgozó építőelem.

### 9.1 Elméleti áttekintés

Diszkrét idejű rendszerek esetén a  $H(z)$  transzferfüggvény segítségével teremtünk kapcsolatot a rendszer bemenete és kimenete között. Z-transzformáció esetén az összefüggés így  $H(z)=Y(z)/X(z)$ . A  $H(z)$  racionális törtfüggvény akkor, ha a rendszerünk lineáris, időinvariáns és kauzális. A kauzalitás megléte szükséges a gyakorlatban való megvalósításhoz, ugyanis olyan bemeneti jel, ami még meg sem jelent az adott időpillanatban nem vehet részt a kimenet értékének alakításában. Azaz adott  $n$ -hez tartozó kimeneti értékek meghatározásához, csak az aktuális és az azt megelőző  $n$ -ekhez tartozó bemeneti értékeket, valamint a már korábban kiszámított kimeneti értékeket vehetjük figyelembe. Az egyes értékeket természetesen súlyozhatjuk is különböző konstans tényezőkkel – ez még belefér a lineáris műveletek halmazába. A kauzalitás miatt tehát nem vehetjük figyelembe a későbbi  $n$ -ekhez tartozó bemeneti értékeket, valamint a még ki nem számolt kimeneti értékeket. A fentiek alapján a kimenet  $n$ -dik értéke a következő formában írható fel:

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

ahol  $b_m$  és  $a_k$  konstans szorzó tényezőket takar. Látható, hogy  $y(n)$  meghatározásához  $(M+1)$  darab bemeneti és  $N$  darab kimeneti értéket használtunk fel. Ez egy differencia egyenlet. Áttérve „ $z$ ” tartományba

$$Y(z) = \sum_{m=0}^M b_m X(z) z^{-m} + \sum_{k=1}^N a_k Y(z) z^{-k}$$

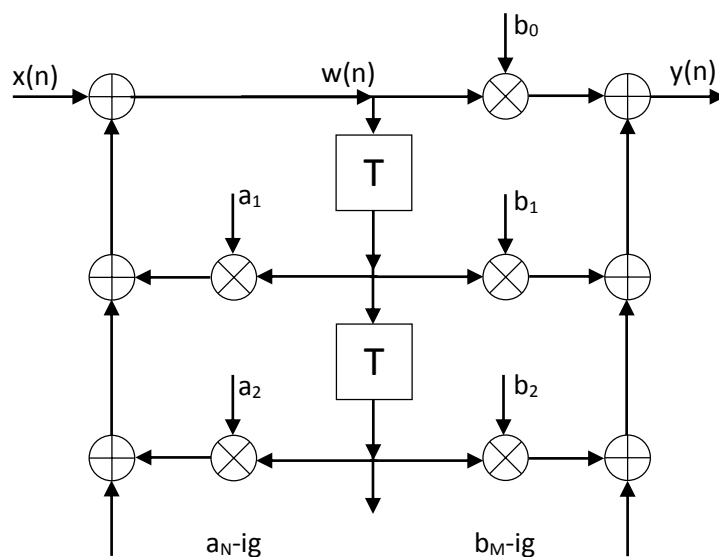
adódik.  $X(z)$  és  $Y(z)$  kiemelhető az összegzésből, mivel futó index nincs bennük jelen. Az egyenletrendezés után a következő törtfüggvény áll elő:

$$H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{m=0}^M b_m z^{-m}}{1 - \sum_{k=1}^N a_k z^{-k}} = \frac{B(z)}{A(z)}$$

A transzferfüggvény ez esetben tehát két polinom hányadosa, amely nagyon megkönnyíti a vele való tervezési lépéseket.

**$B(z)$  gyökei zérusoknak,  $A(z)$  gyökei pólusoknak hívjuk.**

Szűrőtervezés során ilyen típusú transzfer függvényekkel találkozunk. Egy tipikus megvalósítási topológiája az ilyen jellegű rendszereknek pl. a **2. típusú direkt forma**:



9-1. ábra. 2. típusú direkt forma (IIR)

Ha az összes  $a_k$  együttható 0 értékű, azaz a nevező 1-gyel egyenlő, akkor **nem rekurzív struktúráról** beszélünk, és ennek impulzusválasz függvénye **véges**  $n$ -ben. Ezeket a rendszereket nevezzük **FIR (Finite Impulse Response) szűrőknek** (ezek tervezésével foglalkozik jelen mérés is). Az ilyen topológiában nincs visszacsatolás, mivel az aktuális  $y(n)$  meghatározásában nem használunk fel megelőző kimeneti értékeket, az csak a bemenettől (annak jelenlegi és korábbi értékeitől) függ. A fentiekből még az a tulajdonság is következik, hogy a FIR szűrők sosem gerjednek be.

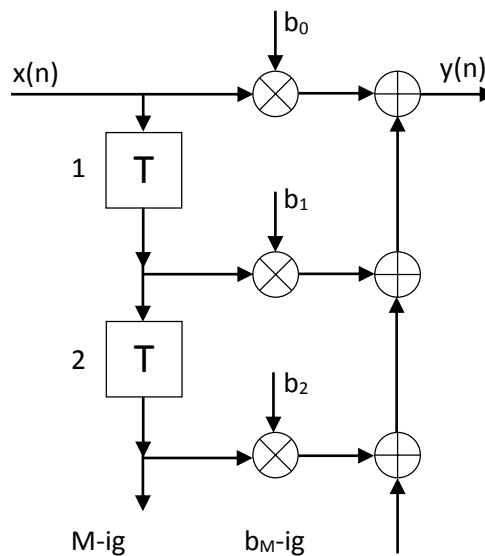
(Ha akár csak egy  $a_k$  együttható is nem 0, abban az esetben a rendszerben megjelenik a visszacsatolás, a rendszer impulzusválasza  $n$ -ben végtelen lesz – ezeket nevezzük IIR (Infinite Impulse Response) szűrőknek.)

### 9.1.1 FIR szűrők tulajdonságai

A FIR szűrők transzfer függvénye a fent leírtak alapján a következő egyszerűbb formában írható fel:

$$H(z) = \sum_{m=0}^M b_m z^{-m},$$

és az általános esetben felrajzolt topológia a következő formára egyszerűsíthető (felfedezhető, hogy csupán a visszacsatolásos ágakat hagytuk el):



9-2. ábra. FIR szűrő megvalósítása 2. típusú direkt struktúrával

A fenti ábra szerinti FIR struktúra fokszáma  $N=M+1$ . A transzferfüggvény felírásából pedig látható, hogy ebben az esetben a kiindulásul vett racionális törtfüggvény nevezője eggyel egyenlő, azaz a **FIR szűrőknek csak zérusai vannak, pólusai nincsenek.**

A pólusok hiánya miatt a FIR szűrőknek két fontos tulajdonságuk van:

- nem gerjednek be,
- fázisválasz függvényük lineáris, illetve konstans 90 fokra szuperponálódó lineáris.

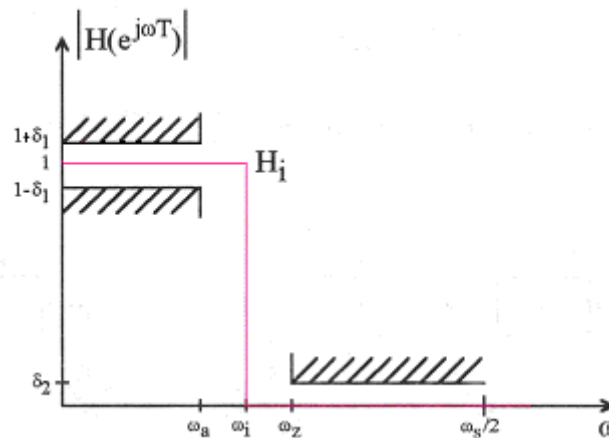
### 9.1.2 FIR szűrők tervezése

Három alapvető módszer használatos a FIR szűrők tervezésekor, mivel a hagyományos szűrőtervezési közelítések a pólusok mellett zérusokat is generálnak:

- ablakozás
- frekvencia mintavételezés
- optimális tervezés

Tervezés során elsősorban arra keressük a választ, hogy egy adott követelmény kielégítéséhez mekkora foksám és együttthatók tartoznak, máshogy fogalmazva, milyen a szűrő impulzusválasza.

Egy aluláteresztő szűrőre megfogalmazott amplitúdóválasz követelmény látható a következő ábrán (itt a fázisválaszra lineáris előírást teszünk):



9-3. ábra. Aluláteresztő szűrő amplitúdóválaszának specifikációja

Az ábrán látható  $H_i$  a FIR szűrő ideális amplitúdóválasza (színessel). Tervezéskor ezt fogjuk a követelményeknek megfelelően bizonyos mértékben megközelíteni.

Az áteresztősávban a frekvenciaválasz függvénynek ki kell elégítenie a

$$1 - \delta_1 \leq |H(e^{j\omega T})| \leq 1 + \delta_1, \quad |\omega| \leq \omega_a$$

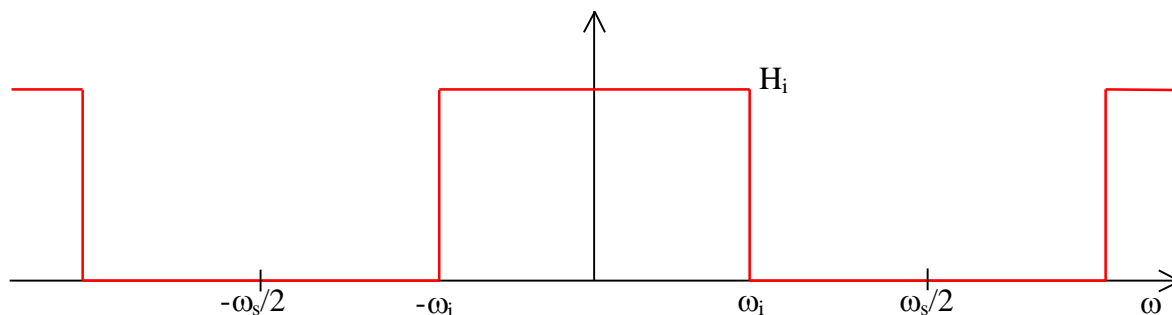
egyenlőtlenséget. A zárósávban pedig

$$|H(e^{j\omega T})| \leq \delta_2, \quad \omega_z \leq |\omega| \leq \omega_s/2$$

kell, hogy teljesüljön. A gyakorlatban  $\delta_1$  helyett sokszor  $A_a$ -t használunk, ami logaritmikus skálán a 0 dB-hez mért maximális eltérést adja meg pozitív-negatív irányban, míg  $\delta_2$  helyett pedig a zárósávi elnyomást használjuk, ami:

$$A_z = -20 \lg(\delta_2).$$

Az  $\omega_a$  és  $\omega_z$  között lévő átmeneti tartományban nem teszünk külön előírást, de azt azért elvárjuk, hogy  $|H(e^{j\omega T})|$  értéke itt is a  $[0,1]$  határok között legyen. Az együttható valós értékei következtében csak a  $[0, \omega_s/2]$  frekvenciatartományt tudjuk kézben tartani a tervezésnél. A függőleges tengelyre tükrözve ezt a tartományt kapjuk meg a  $[-\omega_s/2, 0]$  tartományt. Az így kapott  $[-\omega_s/2, \omega_s/2]$  sáv pedig mindkét irányban végtelenszer ismétlődik.



9-4. ábra. Az  $\omega$  tengelyen periodikus  $H_i$  impulzusválasz függvény

### 9.1.3 FIR szűrőtervezés ablakozással

A FIR szűrőtervezés legegyszerűbb megvalósítási módja az ablakozás használata, amely annyit jelent, hogy az ideális amplitúdó válaszhoz tartozó, végtelen hosszú impulzusválasz sorozatot egy ablakozó sorozattal lecsonkítjuk  $N$  hosszúságúra. Első lépésként az ideális aluláteresztő szűrő amplitúdó karakterisztikából kiindulva határozzuk meg az ehhez tartozó impulzusválasz sorozatot, azaz az együtthatókat:

$$h_i(n) = \frac{T}{2\pi} \int_{-\omega_s/2}^{\omega_s/2} H_i(e^{j\omega T}) e^{j\omega nT} d\omega \quad ,$$

A  $h_i(n)$  értékek azok az együtthatók, amik a 9-2. ábrán látható  $b_i$  konstansszorzókat adják, bár nem közvetlenül, mivel a  $h_i(n)$  értékeket még súlyozni fogjuk.

Az egyenlet kiszámításához a diszkrét idejű rendszerek alapjai ismertetésekor használt összefüggések alapján a  $H(e^{j\omega T})$ -t  $h(n)$  z transzformáltjának egységsugarú körön való kiértékelésével kapjuk meg, és ez visszafelé is alkalmazható. Kihasználva, hogy  $H_i(e^{j\omega T})$  a  $[-\omega_s/2, \omega_s/2]$  tartományban (itt kell az integrált elvégezni) csak  $[-\omega_i, \omega_i]$  tartományban vesz fel nem 0 értéket így az integrálás határait módosíthatjuk  $-\omega_i$  és  $\omega_i$ -re. A  $[-\omega_i, \omega_i]$  tartományban viszont  $H_i(e^{j\omega T})=1$  (4. ábra), így ezt is helyettesíthetjük egy 1-es számmal:

$$h_i(n) = \frac{T}{2\pi} \int_{-\omega_i}^{\omega_i} 1 e^{j\omega nT} d\omega = \frac{\sin(\omega_i nT)}{\pi n}$$

Az integrálást elvégezve kapjuk a szinuszos összefüggést.

A probléma ezen a ponton az, hogy az ideális súlysorozat (ami a szögletes 9-4. ábrabeli  $H_i$  függvényt kiadná) végtelen sok elemből áll (végtelen sok  $h_i$ -t kellene kiszámolni), más szóval csak végtelen foksámú szűrővel lenne megvalósítható. Gyakorlati megvalósítás csak véges foksám esetén lehetséges, így kénytelenek vagyunk az együtthatók számát véges számúra csökkenteni, legyen ez  $N$ .

$$-\frac{(N-1)}{2} \leq n \leq \frac{(N-1)}{2}$$

A korlátokat 0 körül szimmetrikusra kell választani, mert ez biztosítja a lineáris fázismenetet. Kauzalitás megtartásához, viszont nem használhatunk 0-nál kisebb indexű elemeket, ezért a fenti eredményt el kell tolni, úgy, hogy csak a pozitív tartományba essen. (Jobbra (N-1)/2-vel, így a -(N-1)/2 indexű elem épp a 0-hoz esik, a negatív tartományban pedig nem lesz értékünk.)

$$h_i(n) = \frac{\sin\left[\omega_i T \left(n - \frac{N-1}{2}\right)\right]}{\pi\left(n - \frac{N-1}{2}\right)}, \quad 0 \leq n \leq N-1.$$

Az együtthatók számításakor így N db (véges!) együtthatót kapunk, amik egy véges fokszerű FIR szűrőt határoznak majd meg. Páratlan N esetén a hányados nulla per nulla jellegű lesz, amit L'Hospital szabály alkalmazásával tudunk számítani.

Felüláteresztő szűrő esetén N csak páratlan lehet!

Korábbiakban láttuk, hogy pl. egy négyszögjelet harmonikus komponensekre bontunk, akkor végtelen sok komponens összege adja vissza az eredeti négyszögjelünket, ha csak véges számú komponenst használunk az eredeti jel helyett kisebb-nagyobb mértékben oszcilláló jelet kaphatunk csak.

Hasonlóan alakul ez itt is. Az impulzusválasz komponenseinek csonkolása miatt az egyenes ideális amplitúdómenet ingadozni fog, melynek mértékét egyrészt az alkalmazott fokszerű határozza meg.

A különböző tervezési módszerek alkalmazásával tudjuk ezt az ingadozást olyan határok között tartani, hogy a feladatban megadott feltételeknek megfelelő szűrőket tudjunk tervezni.

Eljutottunk tehát oda, hogy elkészült az ideális karakterisztikát kielégítő együttható sorozat N hosszúságúra való csonkítása. A példákban előforduló szűrőkre igaz az, hogy a középső együtthatóktól bármely irányba távolodva, jellegében csökkenő nagyságú együtthatókat kapunk a nevező n függése miatt. Fizikailag is kielégítő ez, hiszen korábbiak alapján a szűrő csoportfutási- vagy késleltetési ideje:

$$D(\omega) = \frac{M}{2}T = \frac{N-1}{2}T,$$

amely megegyezik a kauzalitáshoz szükséges eltolással. A FIR szűrő tehát egy olyan súlyozott összegképző operáció, ahol távolodva attól a ponttól, amelyre a válasz képződik, egyre kisebb súllyal esnek latba a bemeneti értékek. A legkézenfekvőbb megoldás az lenne, ha ezt, a csak hosszában csonkított sorozatot tekintetnénk a szűrő együtthatóinak. Ahogyan

hamarosan kiderül ez azonban nem elég, értékeiben is meg kell változtatnunk a sorozat elemeit. Általánosságban tehát a szűrő impulzusválasz sorozatát a

$$h(n) = h_i(n)w(n)$$

összefüggés alapján nyerjük, tehát az ideális karakterisztikából kapott sorozat tagjait mintánként beszorozzuk egy ablakozó sorozat elemeivel.

Összegezve: véges mintaszámmal megvalósíthatatlan ideális, vagyis átmeneti tartomány- és áteresztősáv ingadozás nélküli, nulla zárósávi átvitelrel bíró karakterisztikából az ablakozó sorozat segítségével kapjuk meg a követelményeket kielégítő fizikailag megvalósítható hálózatot.

Az **ablakozó sorozatok szerepe** nagyon hasonló a diszkrét Fourier transzformációnál tárgyaltakhoz. **Időtartományban vizsgálva** egyrészt **megszüntetik** az impulzusválasz **sorozat két szélén**, az ideális együtthatókban még meglévő **ugrásokat**, másrészt pedig az **együtthatók közepe** felé haladva tovább **növelik ezek szerepét** a súlyozott összegben. Frekvenciatartományban szemléletesen felfoghatjuk az ablakozás hatását úgy, mint az ideális karakterisztikában lévő ugrások helyére szuperponálódó, adott ablakozó sorozathoz tartozó ekvivalens átviteli függvényt. A főnyalábjának szélessége az átmeneti tartomány szélességét, az első oldalnyaláb maximuma pedig az áteresztősávi ingadozás értékét határozza meg. Mivel az ekvivalens átviteli függvények szimmetrikusak, e tervezési módszer használatakor  $\delta_1 = \delta_2$  vagyis nem kaphatunk különböző értéket a zárósávi elnyomásra és az áteresztősávi ingadozásra.

Négyszögletes ablakozó sorozatot használva (amikor az ablakozó együtthatók mind 1-ek), a Gibbs oszcillációként emlegetett jelenséghez jutunk. Ilyenkor az  $N$  fókuszam nagyságától függetlenül 9% ingadozás van az áteresztő és a zárósávban egyaránt.  $N$  növelésével csak a hullámok szélessége csökken.  $\delta_1 = 0.09$ ,  $A = 20 \log 0.09 = -20.91$  dB. Az átmeneti tartomány szélessége pedig  $0.9 \omega_s / (N-1)$ . Ezekkel a jellemzőkkel komoly követelményű szűrő nem tervezhető. Gyakorlati megvalósításoknál az ablakozó függvények használata mindenképpen szükséges. Az ablakozófüggvények (pl Hanning, Hanning, Blackman) problémája, hogy a fókuszam növelésével ugyan az átmeneti tartomány szélessége csökkenthető, de  $\delta_1$ ,  $\delta_2$  értéke adott ablak esetén konstans. Ezt a problémát Kaiser ablakozó sorozata oldja meg, mely rugalmasan képes ezen a paraméteren is változtatni.

## 9.2 Feladatok

$$h_i(n) = \frac{\sin\left[\omega_i T \left(n - \frac{N-1}{2}\right)\right]}{\pi\left(n - \frac{N-1}{2}\right)}, \quad 0 \leq n \leq N-1.$$

1. Számítsuk ki az alábbi parancsok használatával egy 25-öd egy 50-ed és egy 100-adrendű szűrő koeficienseit a fenti képlet alapján!  $h_n$  számítása  $\text{sinc}(x) = \sin(x)/x$  függvény felhasználásával történik úgy, hogy a frekvencia értékeket normalizálva helyettesítjük (1 egység =  $f_s/2$ ).

N=25, 50, 100;

```
N=25;
n=[0:N-1];
fs=48000;
fi=9600;
hn25=fi/(fs/2)*sinc(fi/(fs/2)*(n-(N-1)/2));
fvtool(hn25,1);
```

Az fvtool (Filter Visualization Tool) a  $H(z)$  racionális törtfüggvény paramétereit várja (első paraméter a számláló, második a nevező – FIR szűrőknél ez 1!)

2. Az fvtool ablak megjelenésekor hasonlítsuk össze a kapott amplitúdó átviteli ábrákat, majd kattintsunk jobb egérgombbal az y tengely Magnitude (dB) feliratán, majd válasszuk a csak Magnitude lehetőséget. Ezt az ábrát is hasonlítsuk össze a 3 esetben, hogyan változik az ingadozás (Gibbs oszcilláció) mértéke, továbbá az átmeneti tartomány meredeksége?
3. Alkalmazzunk az ablakozást a kiszámított  $h_n$  értékekre mind a 3 esetben! A látványosság kedvéért alkalmazzuk a Hamming ablakot! (Kipróbálható a Kaiser ablakkal is, de ott az oszcilláció csökkentésének hatása ezek a paraméterek mellett kisebb.)

```
hnw25=hn25.*hamming(N)';
fvtool(hnw25);
```

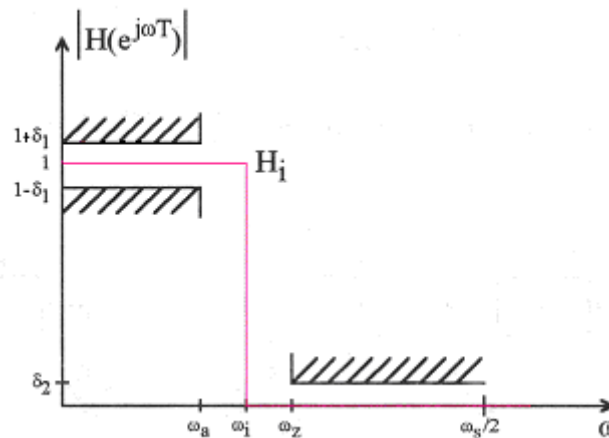
Vizsgáljuk meg egy aluláteresztő szűrő esetében a megvalósítás foksámát az egyes követelmények függvényében egy MatLab demo segítségével!



4. Adjuk ki a parancssorba a `filterdemo` parancsot! Állítsuk be a következő paramétereket!

Mintavételi frekvencia,  $F_{\text{samp}}=2$  kHz (Vigyázat a demo kezelése kissé nehézkes, az értékeket egér használatával is be lehet állítani az ábrában, vagy ha közvetlenül írjuk át őket, akkor egérrel kattintsunk egy másik szövegmezőbe annak érvényesítéséhez!)

5. Állítsuk be a felső lenyíló menüben a KAISER ablakozó sorozatot!



9-5. ábra. Szűrőkövetelmény

$$\omega_a/2\pi=f_{\text{pass}}=500 \text{ Hz}$$

$$\omega_z/2\pi=f_{\text{stop}}=600 \text{ Hz}$$

$$R_{\text{pass}}=3 \text{ dB}$$

$$R_{\text{stop}}=A_z=50 \text{ dB}$$

$R_{\text{pass}}$  és  $R_{\text{stop}}$  értékek az amplitúdóválasz tűrésértékeit határozzák meg dB-ben, ( $\delta_1$ ,  $\delta_2$  helyett használja a MatLab)

6. Jegyezzük le, hogy hányad fokú szűrőt számolt ki a program a megadott paraméterekhez!
7. Változtassunk a paramétereken, úgy hogy csökkentsük az átmeneti tartomány szélességét - a szűrő meredekségét - a kezdeti 100 Hz-ről a felére! Legyen  $f_{\text{pass}}=500$  Hz,  $f_{\text{stop}}=550$  Hz. Mi változik?
8. Változtassunk az  $R_{\text{pass}}$  és  $R_{\text{stop}}$  értékeken, egyszerre csak az egyiken! Hogyan befolyásolják a szűrő fokszámát a különböző változtatások?
9. Mi történik, ha valamelyik beállításnál a szűrő fokszámának meghatározását átállítjuk „kézire” és az előbb automatikusan kiszámított érték alapján beírunk kb. harmad akkora értéket! Teljesíti a szűrő a megadott követelményeket?

10. *Állítsuk át a szűrő típusát most FIRPM-re. Ez egy optimális megvalósítást számoló változat. Állítsuk vissza a szűrő paramétereit az első feladat értékeire! A KAISER ablakozó számítósos típushoz képest mekkora fokszámmra van szüksége ennek a változatnak egy ugyanolyan követelményű szűrő megvalósításához?*

### 9.2.1 Fdatool (Filter Design and Analysis Tool) használata

1. *Írjuk be a parancssorba az `fdatool` parancsot!*

Nézzük meg a következő felüáteresztő szűrő megtervezését ennek a programnak a segítségével! A megjelenő ablak bal oldalán ellenőrizzük, hogy a Design filter gomb legyen benyomva (legalsó)! A felső ikonsoron pedig a Filter Specification gomb (középtájon).

2. *Megfelelő helyeken adjuk meg a következő paramétereket!*

- Felüáteresztő szűrő
- FIR struktúra, ablakozás alkalmazással
- Legkisebb fokszámú megoldást keressük (A MatLab „rend”-et használ - order)
- Az ablak típusa Kaiser legyen (nyomjuk meg bátran a View gombot az ablakfüggvény megjelenítéséhez, a Scale Passband maradjon bepipálva)
- Majd adjuk meg az alábbi paramétereket!

$$f_s = 13 \text{ kHz}; f_z = 2 \text{ kHz}; f_a = 3 \text{ kHz};$$

$$A_z \geq 66\text{dB}; A_a \leq 1 \text{ dB}$$

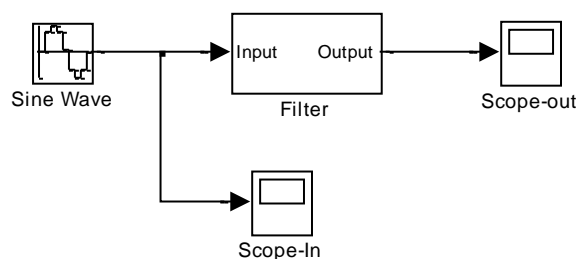
3. *Kattintsunk a Design Filter gombra! Ekkor az eszköztárban a Magnitude response gomb lesz aktív – megjelenítve az amplitúdó választ a rendszernek.*
4. *Nézzük meg a Magnitude vagy a Magnitude Squared ábráját is (Jobb gomb a Magnitude-on) – látszik az 1-es feladathoz hasonló oszcilláció?*
5. *Váltsunk át normalizált frekvenciaskálára (jobb gomb a Frequency-n). Hogyan kell a két skála között átszámítani az értékeket?*
- 6.
7. *Az ablak felső részén lévő eszköztárban nézzük végig az egyes gombok funkcióit!*
8. *Milyen jellegű a fázismenete a szűrőnek?*
9. *Milyen a csoportfutási idő?*
10. *Nézzük meg a Kronecker deltára és az egységugrásra adott válaszokat!*
11. *A pólus-zérus ábrán, hogyan helyezkednek el a zérusok?*

12. Mennyi a koefficiensek száma? Mi a köze ezek számának és a szűrő fokszámának egymáshoz?
13. Nézzük meg az ablak bal felső részén lévő információkat! Mit tudhatunk meg ezekből a szűrőnkéről?
14. Valósítsuk meg a szűrőt a bal oldalon lévő ikonok közül a Realize modellel! Itt minden beállítást hagyhatunk alapértelmezettnek, nyomjuk meg a gombot! A megvalósított szűrőnk a SimuLinkba jelenik meg egy blokkként.

### 9.2.2 Szűrő analízis a SimuLinkban

A SimuLinkban vizsgáljuk meg a szűrő viselkedését szinusz jelekkel!

Építsük össze az alábbi hálózatot! A Filter a megvalósított szűrőnk. A többi elemet az eszköztárban a Library Browser gombra kattintva (vagy View menü | Library browser) tudjuk előkeresni, a megjelenő ablakban. Keressük itt meg a legelső SimuLink nevű csoportot, majd a Sinks és a Sources alcsoportokon belül megtaláljuk a megfelelő elemeket. A szinusz jel magadásakor ügyeljünk a frekvencia helyes megadására, amit érdemes  $2\pi \cdot f$  alakban beírni, így nem kell radiánba átszámolni minden értéket. A mintavételi időt pedig a mintavételi frekvenciával is megadhatjuk:  $1/13000$  beírásával. (Sine type: Time based, Time (t): Use simulation time) formában megadni és akkor nem kell számolni.



9-6. ábra. SimuLink szűrővizsgálat

A szkópokon a MatLab alapértelmezésként csak az utolsó 5000 értéket hagyja megjelenítve a szimuláció befejezése után. Ha a teljes képet meg akarjuk jeleníteni, kattintsunk a szkóp ablak eszköztárában a második (Parameters) gombra, majd válasszuk a Data History fület. A Limit Data... sorból szedjük ki a pipát. A műveletet a másik szkópra is meg kell ismételni.

1. *Három jellegzetes frekvenciát válasszunk, hogy ellenőrizzük a szűrő viselkedését, a záró, az áteresztő és az átmeneti tartományban (pl 1500, 3500, 2500 Hz)! A szkóp képeket mentjük el és számoljuk ki a bemenet és a kimenet amplitúdó változásából, hogy mekkora volt a csillapítás decibelben!*

(Bemenet amplitúdó 1, kimenet 0.9.  $0.9/1=0.9$  – ebből  $20 \cdot \log(0.9)=-0.91\text{dB}$ )

## 10 IIR szűrők tervezése

Az IIR (Infinite Impulse Response) struktúrákkal viszonylag alacsony fokszámmal jó hatásfokkal tudunk szűrési feladatokat elvégezni. A struktúra tartalmaz visszacsatolást, ezért könnyen instabillá, gerjedékennyé válhat. Ebben a fejezetben az IIR fontosabb jellemzőit ismertetjük.

### 10.1 Elméleti áttekintés

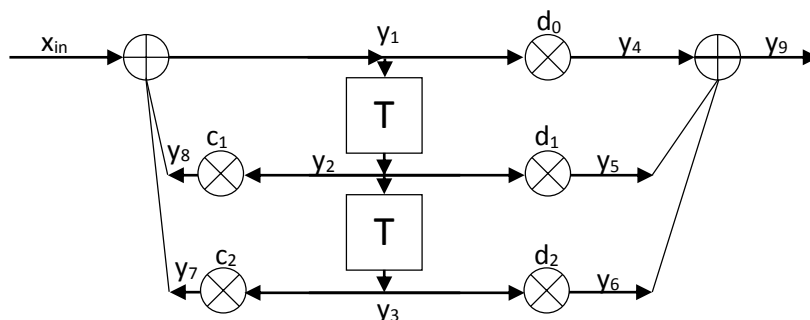
Az IIR szűrők olyan speciális kauzális, időinvariáns, lineáris operációk, melyek transzferfüggvénye:

$$H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{m=0}^M b_m z^{-m}}{1 - \sum_{k=1}^N a_k z^{-k}} = \frac{B(z)}{A(z)}$$

Emlékezzünk vissza a FIR szűrőkre, melyek transzferfüggvényében az  $a_k$  együtthatók minden  $k$ -ra 0-k voltak. Mivel itt a transzferfüggvény mind számlálójában, mind nevezőjében nem 0 együtthatók szerepelnek a szűrőnek pólusai és zérusai is lesznek egyaránt.

A FIR szűrők tervezési módszerétől eltérően az IIR szűrők tervezési lépései máshogy alakulnak. Röviden összefoglalva a következő lépéseket kell végrehajtani.

- követelmények definiálása
- frekvencia előtorzítás (erre a lépésre azért van szükség, mert a későbbiekben következő hagyományos – folytonos idejű – szűrőtervezés esetén a diszkrét rendszerre megfogalmazott frekvencia követelményeket át kell konvertálni a folytonos időbe.)
- ha szükséges: típus transzformáció (a tervezés mindig aluláteresztő szűrő tervezést jelent, amennyiben más típusú szűrőt akarunk tervezni, a követelményeket át kell transzformálni aluláteresztővé.)
- hagyományos szűrőtervezés (ebben a lépésben egy átranszformált követelményeknek megfelelő analóg aluláteresztő szűrőt kell megtervezni, azaz meghatározni a transzferfüggvényének gyökeit. A meghatározáshoz többféle közelítés használható, melyekből ebben a mérésben a Butterworth, Chebysev, inverz Chebysev és Cauer közelítések tulajdonságait vizsgáljuk meg.)
- gyökök visszatranszformálása és típustranzformáció inverze egy lépésben (itt térünk vissza ismét diszkrét időtartományba)
- realizálás, másodfokú tagokból felépített kaszkád vagy speciális kaszkád struktúrával.



Ilyen tagokból kapcsolhatunk össze sorba  $n$  db-ot a 2. típusú kaszkád forma létrehozásához.

Az IIR szűrők esetén a visszacsatolás jelenléte miatt ügyelni kell a rendszer stabilitására is. Stabil a szűrő, ha a pólusai az egység sugarú körön belül helyezkednek el a komplex síkon.

## 10.2 Feladatok

### 10.2.1 FIR és IIR szűrők összehasonlítása

Hasonlítsuk össze az IIR és a FIR szűrők amplitúdó karakterisztikáinak tulajdonságait a MatLab `filtdemo` programjával a lentebb található táblázat kitöltésével! Induljunk ki az alapértelmezett paraméterekből (Fsamp=2000, Fpass=500, Fstop=600, Rpass=3, Rstop=50).

1. *Elsőként jelenítsük meg a FIRPM (optimális), majd a Kaiser ablakozott FIR szűrő amplitúdó karakterisztikáját!*
2. *Jellemezzük a két FIR és a különböző típusú IIR szűrők (Butterworth, Chebysev (Cheby1), inverz Chebysev (Cheby2), Cauer (Ellip)) amplitúdó karakterisztikáját az áteresztő-, a záró- (mennyire egyenletes) és az átmeneti tartományban (mennyire meredek és hogyan változik a meredekség). Figyeljük a szűrő fokszámának alakulását is!*
3. *Milyen különbség látszik a követelmények beállításával kapcsolatban (zöld vonal!) az áteresztő tartományban a FIR és az IIR szűrők esetében?*

IIR szűrő tervezés az fdatool (Filter Design and Analysis Tool) használatával.  
Tervezzünk **aluláteresztő** szűrőt a **Butterworth** közelítés használatával!

1. *Az FDATool ablak bal oldalán ellenőrizzük, hogy a Design filter gomb legyen benyomva! A felső ikonsoron pedig a Filter Specification gomb (középtájon). Ezután a megfelelő helyeken adjuk meg a szükséges paramétereket!*

- Legkisebb foksámú megoldást keressük.
- Az Options mezőben állítsuk be passband-ot.
- Majd adjuk meg az alábbi paramétereket!

$$f_s = 24 \text{ kHz};$$

$$f_{\text{áteresztő}} = 7 \text{ kHz};$$

$$f_{\text{záró}} = 7.5 \text{ kHz};$$

$$A_{\text{záró}} \geq 66 \text{ dB};$$

$$A_{\text{áteresztő}} \leq 1 \text{ dB}$$

2. *Kattintsunk a Design Filter gombra! Ekkor a Magnitude response gomb lesz aktív – megjelenítve az amplitúdó válaszát a rendszernek.*
3. *Nagyítsuk ki kb. a 6 és 8 kHz közötti és a 0–100 dB részét az ábrának! Ellenőrizzük, hogy a követelményben megadott frekvenciáknál megfelelő-e az amplitúdó menet! Becsüljük meg, mekkora meredekséggel törik le a görbe (dB/kHz mérték alapján)?*
4. *Milyen a görbe jellege az áteresztő-, az átmeneti- és a zárótartományban? (Nagyítsunk jól rá a görbe megfelelő részletére!)*
5. *Nézzük meg a fázis válaszát a rendszernek! Jellemezzük a görbe alakját! A fázis válasz görbéjének alapján hogyan alakulhat a csoportfutási idő frekvencia függése? Ellenőrizzük le, hogy jól gondoltuk-e, és írjuk le a tapasztalatokat!*
6. *Nézzük meg az impulzusválaszát a szűrőnek. Visszaemlékezve a FIR szűrők hasonló ábráira, mit mondhatunk azokhoz képest ennek az IIR szűrőnek a késleltetéséről?*
7. *Mentsük el a csoportfutási idő ábráját! Hozzunk létre ugyanilyen követelményekkel egy FIR Kaiser ablakos szűrőt, és hasonlítsuk össze a csoportfutási idő ábrákat! (Vessünk közben egy pillantást a rendszer Order paraméterére is – mekkora ez az IIR megvalósításhoz képest?) Hányad fokúak ezek a szűrők?*
8. *Nézzük meg a pólus-zérus ábrát! Hol helyezkednek el a komplex síkon a pólusok és a zérusok?*
9. *Végül a szűrőtervet mentsük el butter.fda néven!*
10. *Hozzunk létre egy Chebysev (Cheby1) közelítéses szűrőt, ugyanezekkel a követelményekkel! Az Options mezőben maradjon a passband beállítás!*

11. *Vizsgáljuk meg az amplitúdó válaszát a rendszernek a 3 tartományban külön-külön felnagyítva!*
12. *A Butterworth szűrőhöz képest milyen az amplitúdómenet az áteresztő tartományban?*
13. *Milyen az átmeneti tartomány áteresztősségi oldalán a meredekség (dB/kHz)? (Nagyítsunk rá kb. a 6.8 .. 7.4 kHz-es tartományra!)*
14. *Hogyan változott a szűrő fokszáma ezzel a közelítéssel?*
15. *Nézzük meg az impulzusválaszát a rendszernek, majd a csoportfutási időt. Hogyan változott a rendszer késleltetése, mondjunk még az ábráról leolvasva egy kb. átlagértéket!*
16. *Nézzük meg a pólus-zérus ábrát! Hol helyezkednek el a komplex síkon a pólusok és zérusok?*
17. *Mentsük el a szűrőt cheby1.fda néven!*
18. *Hozzunk létre egy inverz Chebysev (Cheby2) közelítéses szűrőt, ugyanezekkel a követelményekkel! Az Options mezőben maradjon a passband beállítás!*
19. *Vizsgáljuk meg az amplitúdó válaszát a rendszernek a 3 tartományban külön-külön felnagyítva!*
20. *A Chebysev szűrőhöz képest milyen az amplitúdómenet az áteresztő tartományban? Jobban hasonlít ez a szakasz a Butterworth szűrőéhez?*
21. *Milyen az átmeneti tartományban átlagosan a meredekség (dB/kHz)?*
22. *Milyen a görbe a zárótartományban?*
23. *Nézzük meg a pólus-zérus ábrát! Hol helyezkednek el (most) a komplex síkon a pólusok és zérusok?*
24. *Hozzunk létre egy Cauer (Elliptic) közelítéses szűrőt, ugyanezekkel a követelményekkel! Az Options mezőben maradjon a passband beállítás!*
25. *Vizsgáljuk meg az amplitúdó válaszát a rendszernek a 3 tartományban külön-külön felnagyítva!*
26. *Milyen különbségeket látunk a korábbi szűrőkhöz képest?*
27. *Nézzük meg a pólus-zérus ábrát! Hol helyezkednek el (most) a komplex síkon a pólusok és zérusok?*
28. *Ezzel a közelítéssel hányad fokú szűrőt lehetett tervezni? A többi közelítéssel összehasonlítva fokszám alapján melyik szűrőtípus a „legjobb”?*
29. *Viszont nézzük meg a fázis válaszát a rendszernek! A nemlinearitás ennél a közelítésnél a legerősebb – mi ennek a következménye?*
30. *Valósítsuk meg a Butterworth és a Chebysev szűrőt SimuLinkban!*



31. *A SimuLinkban vizsgáljuk meg az egyik szűrő viselkedését szinusz jelekkel egy-egy jellemző pontban!*
32. *Az átmeneti tartomány áteresztőszávhoz közel eső részében a Chebysev szűrő meredeksége a vizsgálataink alapján meredekebbnek bizonyult, mint a Butterworth-é! Válasszunk ki két megfelelő frekvenciájú jelet, és a kimenetek megjelenítésével demonstráljuk a különbséget!*

## 11 Forrásjegyzék

- [1] Simán István, „*Digitális jelfeldolgozás alapjai*”, OE KVK 1170
- [2] Egri Tamás, „*Hírközlés*”, OE KVK 2000
- [3] Kármán József, „*Híradástechnika II.*”, előadás prezentációk
- [4] Lukács György, Wühl Tibor, „*Híradástechnika I.*”, OE KVK 2090
- [5] Wühl Tibor, „*Bevezetés a MatLab használatába*”, OE KVK 2071
- [6] Wühl Tibor, „*Hullámdigitális jelfeldolgozás alapjai*”, OE KVK 2073
- [7] Géher Károly, „*Híradástechnika*”, Műszaki könyvkiadó
- [8] Mathworks website, <http://www.mathworks.com>