# Zvuk – frekvenční analýza a filtrování

Jan Černocký, FIT VUT Brno

Toto cvičení se věnuje zpracování zvuku. Budeme pracovat s jednokanálovým zvukem na vzorkovací frekvenci $F_s = 44100$ Hz. Je pro Vás připraven v `music.wav`, ale můžete použít jakýkoliv jiný.

## 1 Načtení a základní analýza

```
[s, Fs] = audioread ('music.wav'); s = s'; % potrebujeme radkovy vektor
s = s(1:250000); % zkraceni, aby prehravani netrvalo dlouho.
t = (0:(length(s)-1)) / Fs;
plot (t,s);
```

Pozor, na Windowsech v CVT je lepší pužívat Matlab R2008a, protože u novějšího to vypadá na problém s licencemi na Signal Processing Toolbox. V tomto případě prosím změňte první řádku na
`[s, Fs] = wavread ('music.wav');`
Signál musí být **řádkový vektor** – ověříte pomocí `whos s`

### 1.1 Jedno spektrum

Frekvenční analýzu budeme provádět pomocí diskrétní Fourierovy transformace, která se implementuje pomocí rychlé Fourierovy transformace FFT. Zjednodušeně:

- zpracovávají $N$ vzorků signálu.
- vracejí $N$ komplexních hodnot ve spektru $X[k]$, které jsou rozmístěny od 0 až "skoro" do vzorkovací frekvence $F_s$.
- má cenu se dívat pouze na první polovinu spektra od 0 do $F_s/2$, protože ta druhá je symetrická.

Při zpracování zvuku budeme zobrazovat tzv. logaritmickou spektrální hustotu výkonu (power spectral density – PSD), která je dána jako

$$G[k] = 10 \log_{10} \frac{|X[k]|^2}{N}$$

Nemá cenu zobrazovat spektrum celého signálu, zaměříme se jen na jeden úsek. Funkce `vyber(s, kolik, odkud, Fs)` vybere úsek signálu dlouhý `kolik` sekund od `odkud` sekund, zobrazí jej a zobrazí jeho spektrum. Podívejte se dovnitř této funkce a použijte ji. Dobrá délka úseku pro analýzu je 20 ms.

```
x = vyber (s, 0.02, 2, Fs);
```

### 1.2 Spektrogram

Často nás bude zajímat průběh spektra během celého signálu. 3D obrázek, kde na vodorovné ose je čas, na svislé je frekvence a barva určuje výkon na té které frekvenci, získáme pomocí `spectrogram`. Pro naše cvičení ji obaluje funkce `myspecgram(s, Fs)`.

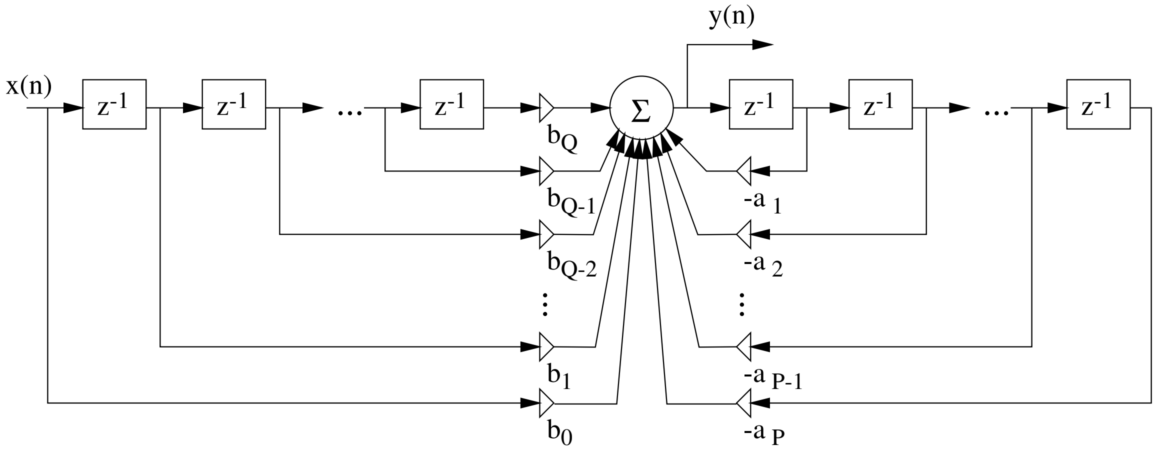
```
myspecgram(s, Fs);
```

## Úkoly

- Přečtěte si help k funkci `spectrogram` a pak zkuste uvnitř funkce `myspecgram` měnit parametry volání `spectrogramu`. Jaký je výsledek ?
- zahrajte si signál pomocí `sound(s,Fs)`

# 2 Filtrace

Číslicový filtr má základní schema:




a můžeme jej popsat **diferenční rovnicí**:

$$y[n] = \sum_{k=0}^{Q} b_k x[n-k] - \sum_{k=1}^{P} a_k y[n-k],$$

kde $b_0 \dots b_Q$ a $a_1 \dots a_P$ jsou koeficienty filtru. Filtrování pomocí diferenční rovnice implementuje matlabovská funkce `filter(b,a,x)`, kde `b` je vektor s koeficienty $b_0 \dots b_Q$ (velikost $Q+1$), `a` je vektor s koeficienty $a_0 \dots a_P$ (velikost $P+1,\ a_0 = 1$) a `x` je vektor se vstupním signálem.

Pomocí $z$-transformace můžeme diferenční rovnici převést na **přenosovou funkci**:

$$H(z) = \frac{Y(z)}{X(z)} = \frac{\sum\limits_{k=0}^{Q} b_k z^{-k}}{1 + \sum\limits_{k=1}^{P} a_k z^{-k}} = \frac{B(z)}{A(z)},$$

kde $B(z)$ je polynom řádu $Q$ a $A(z)$ je polynom řádu $P$. Z přenosové funkce snadno spočítáme frekvenční chování filtru: **komplexní kmitočtovou charakteristiku** tak, že dosadíme $z = e^{j\omega}$, kde $\omega$ je normovaná kruhová frekvence:

$$H(e^{j\omega}) = H(z)|_{z=e^{j\omega}}$$

Dosazení a výpočet provede v Matlabu funkce `freqz(b,a,N,Fs)`, kde `b` je vektor s koeficienty $b_0 \dots b_Q$ (velikost $Q+1$), `a` je vektor s koeficienty $a_0 \dots a_P$ (velikost $P+1,\ a_0 = 1$), `N` je počet bodů pro zobrazení a `Fs` je vzorkovací frekvence. Funkce generuje komplexní kmitočtovou charakteristiku od 0 do $F_s/2$.

Přenosová funkce se dá ovšem také vyjádřit pomocí nulových bodů a pólů. **Nulové body (nuly)** $n_k$ jsou body v rovině $z$, kde je polynom $z^Q + b_{Q-1}z^{Q-1} + \dots + b_1 z + b_0 = 0$ a **póly** $p_k$ jsou body v rovině $z$, kde je polynom $z^P + a_{P-1}z^{P-1} + \dots + a_1 z + a_0 = 0$. Aby byl filtr **stabilní**, musí být všechny póly uvnitř jednotkové kružnice, neboli:

$$|p_k| < 1$$

Vyděsila Vás tato teorie ? Každého na začátku vyděsí… naštěstí máme připravenou funkci `ukazmito(b,a)`, která:

- spočítá a zobrazí impulsní odezvu filtru.
- ukáže kmitočtovou charakteristiku (modul, fáze).
- ukáže póly a nuly a zjistí stabilitu filtru.

# 3 Několik příkladů filtrace

Nadefinujeme vždy filtr, podíváme se na jeho analýzu a vyfiltrujeme náš signál. Výsledky si zobrazíme a poslechneme.

Pro srovnání si do obrázků 1 a 2 dejte výběr a spektrogram originálu. Výběr a spektrogram filtrovaného signálu (obrázky 4 a 5) si pak dejte pod ně, abyste mohli srovnávat.

```
figure(1); x = vyber (s, 0.02, 2, Fs);
figure(2); myspecgram(s,Fs);
```

Začneme jednoduchou dolní propustí.

```
b = [0.2 0.2 0.2 0.2 0.2]; a = [1];
figure(3); ukazmito(b,a,Fs);
sf = filter (b,a,s);
figure(4); xf = vyber (sf, 0.02, 2, Fs);
figure(5); myspecgram(sf,Fs);
soundsc (sf,Fs);
```

Budeme pokračovat ostrou pásmovou propustí:

```
b = [1]; a = [1 -1.6808 0.998];
figure(3); ukazmito(b,a,Fs);
sf = filter (b,a,s);
figure(4); xf = vyber (sf, 0.02, 2, Fs);
figure(5); myspecgram(sf,Fs);
soundsc (sf,Fs);
```

a nakonec něco nestabilního ...

```
b = [1]; a = [1 -1.684 1.002];
figure(3); ukazmito(b,a,Fs);
sf = filter (b,a,s);
figure(4); xf = vyber (sf, 0.02, 2, Fs);
figure(5); myspecgram(sf,Fs);
soundsc (sf,Fs);
```

... moc se toho nezobrazilo. Prohlédnutí mezních hodnot `sf` nám napoví proč:

```
max(sf)
min(sf)
```

# Příklady

1. Který z filtrů je s konečnou impulsní odezvou (FIR) a který je s nekonečnou (IIR) ?

2. Pokud je to FIR, existuje nějaký vztah mezi koeficienty čitatele $b_0 \dots b_Q$ a hodnotami impulsní odezvy $h[n]$ (jsou na výstupu funkce `ukazmito` v obrázku 3) ?

# 4 Třetino-oktávový ekvalizér

Na koncertě jste jistě viděli pana zvukaře s rackem plným elektroniky. Klíčovým elementem tam byl třetino-oktávový ekvalizér, kterým se dá vyrovnat akustika místnosti, zlepšit zvuk nástrojů, atd. My si jej dnes naimplementujeme. Ekvalizér obsahuje 23 filtrů, které jsou navrženy podle článku: Christophe Couvreur: "Implementation of a One-Third-Octave Filter in Matlab Bank", k disposici na
`http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.57.5728`
Matlabové kódy k němu můžete najít na
`http://kom.aau.dk/group/06gr943/thesis/octave/`

My máme banku filtru již navrženou - koeficienty $a_i$ a $b_i$ filtru jsou uloženy v řádcích matic `Amat` a `Bmat` v souboru `filterbank.mat`. Načtení a zobrazení všech frekvenčních charakteristik (všimněte si, že používáme lepší rozlišení ve frekvenci, aby se dobře zobrazily i spodní filtry):

```
load filterbank
Hmat = zeros(23,2048); f=(0:2047) / 2048 * Fs / 2;
for i=1:23,
  H = freqz(Bmat(i,:),Amat(i,:),2048); Hmat(i,:) = abs(H);
end
figure(6); plot(f,Hmat); grid;
```

## Úkoly

1. Ověřte, že jsou filtry opravdu třetino-oktávové. Kdyby byly oktávové, byl by střed frekvenční charakteristiky filtru $i$ dvakrát větší než předcházejícího: $F_i = 2F_{i-1}$. Pokud jsou filtry třetino-oktávové, mělo by to být $F_i = \sqrt[3]{2}F_{i-1}$. Odečtěte z obrázku dvě takové frekvence a ověřte.

2. Frekvenční charakteristiky spodních filtrů nejsou moc dobře vidět. Zkuste použít `semilogx(f,Hmat)`

## 4.1 Rozklad a složení signálu

Signál zkusíme filtrovat do jednotlivých pásem a pak jej zase složit. Výsledek by měl být velmi podobný originálu. Cyklus bude pro každé pásmo ukazovat a přehrávat zvuk, pak čeká na zmáčknutí `Enter`. Pokud by Vás to štvalo, zakomentuje vše, co začíná `figure`, `soundsc` a příkaz `pause`.

```
out = zeros(size(s));
for i=1:23,
  i
  sf = filter(Bmat(i,:),Amat(i,:),s);
  figure(3); ukazmito(Bmat(i,:),Amat(i,:),Fs);
  figure(4); xf = vyber (sf, 0.02, 2, Fs);
  figure(5); myspecgram(sf,Fs);
  soundsc(sf,Fs);
  out = out + sf;
  pause;
end
soundsc(out,Fs);
```

## 4.2 Hraní s ekvalizérem

Se ekvalizérem si můžeme pohrát – třeba nechat jen basy a výšky :-) Příklad neobsahuje zobrazování, dle libosti si je doplňte.

```
cf = zeros(1,23); cf(1:4) = 1; cf(19:23) = 1;
out = zeros(size(s));
for i=1:23,
  sf = cf(i) * filter(Bmat(i,:),Amat(i,:),s);
  out = out + sf;
end
soundsc(out,Fs);
```

Na skutečném ekvalizéru jste ale zřejmě viděli nastavení v dB. Násobící koeficienty filtru se na hodnoty v dB přepočítají:

$$cf_{dB} = 20 \log_{10} cf,$$

takže zpět:

$$cf = 10^{\frac{cf_{dB}}{20}}$$

Zkusíme nastavit kmitočtovou charakteristiku do pěkného "U":

```
cfdb = [10 8 6 4 2 0 -2 -4 -6 -8 -10 -8 -6 -4 -2 0 2 4 6 8 10 10 10];
cf = 10.^(cfdb / 20)
out = zeros(size(s));
for i=1:23,
  sf = cf(i) * filter(Bmat(i,:),Amat(i,:),s);
  out = out + sf;
end
soundsc(out,Fs);
```

A nakonec můžeme zkusit něco jako wah-wah efekt - budeme modulovat střední frekvence na 3 Hz:

```
t = (0:(length(s)-1)) / Fs;
moduldB = 10 * cos (2 * pi * 3 * t);
modul = 10.^(moduldB / 20);
figure(6); plot(t,modul)
out = zeros(size(s));
for i=1:23,
  sf = filter(Bmat(i,:),Amat(i,:),s);
  if (i>=7 & i<=18)
    out = out + modul .* sf;
  else
    out = out + sf;
  end
end
soundsc(out,Fs);
```

## Úkoly

1. Líbí se Vám výsledný zvuk po prostém rozkladu a složení ? Odpovídá vybraný kus signálu (funkce `vyber`) originálu ?
2. Proč je zvuk po úpravách slabší ? Prostudujte help k funkci `soundsc`.

## 4.3 Překvapení na konec – obdélníkový impuls

Na závěr vygenerujeme obdélník o délce 1 s a zkusíme jej rozložit a opět poskládat:

```
close all; % je dobre uklidit na obrazovce ...
so = zeros (1, 3*Fs); so(Fs:2*Fs) = 1;
to = (0:(3*Fs-1)) / Fs;
figure(7); subplot(211); plot(to,so);
out = zeros(size(so));
for i=1:23,
  sf = filter(Bmat(i,:),Amat(i,:),so);
  out = out + sf;
end
figure(7); subplot(212); plot(to,out);
```

## Úkoly

1. Co se to s obdélníkem stalo ? Proč ?