Processamento de Sinal (DF-FCUL)

Novembro de 2014

Filtros FIR (V1.0)

Objectivos

Nesta aula prática introduzem-se as metodologias de projecto de filtros digitais do tipo FIR ("Finite Impulse Response"), utilizando o *Scilab* como ferramenta de auxílio ao projecto. São ilustradas as seguintes técnicas: projecto "com janela"; e projecto por amostragem da resposta na frequência. Apresenta-se também algumas rotinas "enlatadas" de projecto de filtros FIR existentes no *Scilab*.

Introdução

Os filtros FIR, ou filtros não recursivos, são descritos pela seguinte equação às diferenças que simultaneamente é uma convolução:

$$y[n] = \sum_{m=0}^{M} h[m]u[n-m] \qquad (1)$$

onde u é a entrada, y é a saída e h é a resposta ao impulso.

Na versão (1) o filtro FIR é **causal** ($h[n]$ é nula para $n < 0$), e em projectos reais impõe-se adicionalmente uma simetria nos $M+1$ coeficientes de $h[n]$ que faz com que o filtro apresente fase linear, ou seja, atraso (de grupo) constante.

Comparativamente aos filtros IIR ("Infinite Impulse Response"), a outra categoria de filtros digitais estudada em PS, os filtros FIR possuem a desvantagem de usar um número de coeficientes descritivos muito maior. No entanto, as vantagens dos FIR são tais (estabilidade incondicional, fase linear, poucos problemas de implementação no que respeita a precisão numérica e a 'overflow' na execução dos cálculos...) que levam a que os FIR sejam frequentemente seleccionados em aplicações práticas.

A equação do filtro FIR em (1) corresponde à convolução linear, o que leva a que ele seja habitualmente implementado com a FFT. O uso deste algoritmo eficiente alivia o problema decorrente do grande número de coeficientes dos FIRs práticos e das muitas operações (flops) necessárias à implementação directa da convolução. Quando o filtro é aplicado a um sinal de entrada em tempo real, recorre-se habitualmente à técnica "overlap-add" para debitar regularmente troços do sinal de saída $y[n]$, sem ser necessário esperar pela aquisição completa da entrada $u[n]$ para calcular $y[n]$ na totalidade (o que seria inviável em processamento em tempo real).

Por vezes, o filtro é especificado por uma resposta impulsiva $h[n]$ não causal, simétrica em torno de $n = 0$:

$$y[n] = \sum_{m=-P}^{P} h[m]u[n-m] \qquad (2)$$

pelo que, na aplicação prática das técnicas de cálculo da saída deverão ser feitos ajustes aos índices dos vectores. Comparando as eqs. (1) e (2), vemos que $M = 2P$, no que se refere às dimensões dos filtros[1].

[1] Infelizmente, não há normalização na notação dos filtros FIR nos textos. Aqui apresenta-se as "variedades" mais comuns.

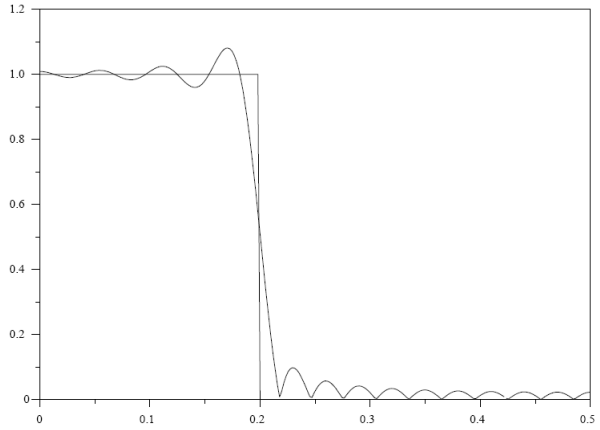

Fig. 1: Amplitude do filtro passa-baixo ideal ("parede"), $|H(e^{j\omega})|$, e aproximação $|H(e^{j\omega})|$ obtida com uma "janela rectangular" (a curva com ondulações).

Projecto de filtros FIR com janelas

Um possível projecto da resposta impulsiva do filtro FIR "real", $h[n]$, parte da relação entre esta e **a resposta na frequência pretendida**, $H_d(e^{j\omega})$. A técnica é simples:

(1) - calcula-se a *transformada de Fourier inversa* de $H_d(e^{j\omega})$ para obter a resposta impulsiva discreta:

$$h_d[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} H_d(e^{j\omega})\, e^{j\omega n} d\omega \qquad -\infty < n < \infty \qquad (3)$$

Porém, $h_d[n]$ não é causal e a sua dimensão é infinita; logo, a expressão anterior não é directamente aplicável. Por exemplo, um filtro passa-baixo ideal com 0 dB de ganho e frequência de corte normalizada ω_c (com $\omega_c < \pi$) tem por resposta impulsiva a seguinte sequência infinita (calculada por (3)):

$$h_d[n] = \frac{1}{n\pi}\sin(n\,\omega_c) \qquad -\infty < n < \infty \qquad (4)$$

(2) - para tornar a técnica prática, utiliza-se só N elementos de $h_d[n]$, com N ímpar, centrados em torno de $n = 0$ (ou esta mesma sequência transladada no tempo), o que corresponde a criar um filtro cuja resposta impulsiva é

$$h[n] = h_d[n]\, r_N[n] \qquad (5)$$

onde $r_N[n]$ é a **janela rectangular** definida por

$$r_N[n] = \begin{cases} 1, & 0 \le |n| \le (N-1)/2 \\ 0 & \text{noutros casos} \end{cases}$$

A resposta na frequência efectivamente conseguida é $H(e^{j\omega})$, a transformada de Fourier (DTFT) de $h[n]$. Na figura 1 está um exemplo em que $N = 33$ e $\omega_c = 0.2$, sendo visível a diferença entre o $|H(e^{j\omega})|$ real e o $|H_d(e^{j\omega})|$ do filtro ideal.

A existência de *ripple* (ondulação, denominada fenómeno de Gibbs) junto à transição das bandas torna esta solução impraticável em muitas aplicações. Pode ser mostrado que o *ripple* não diminui com o aumento de N: este aumento melhora a aproximação em todos os pontos, excepto na zona da

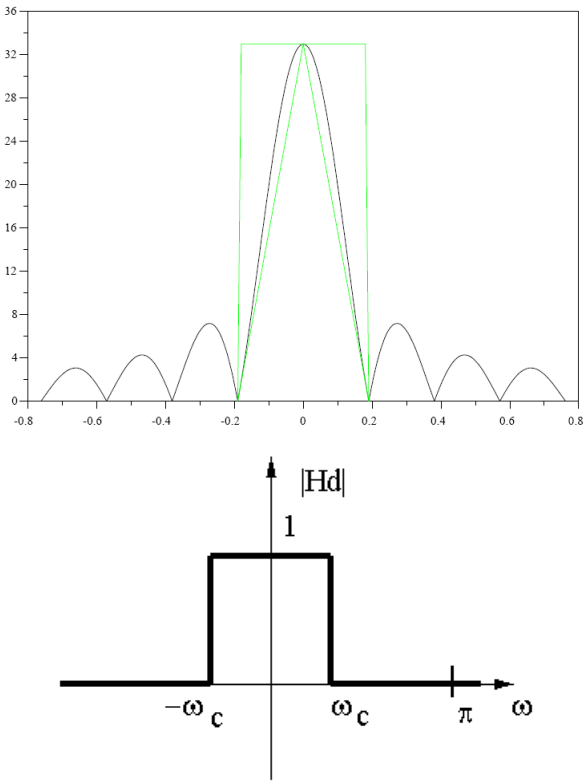

Fig. 2: Magnitudes de $R_N(e^{j\omega})$ e $H_d(e^{j\omega})$.

transição. Além do mais, quanto maior for N maiores serão os tempos de execução da convolução ou da FFT que implementam o filtro, limitando a frequência de amostragem f_S.

Como $h[n]$ é o produto de duas sequências (5), então $H(e^{j\omega})$ será a convolução das respectivas transformadas de Fourier, $H_d(e^{j\omega})$ e $\mathcal{F}\{R_N[n]\} = R_N(e^{j\omega})$:

$$H(e^{j\omega}) = \frac{1}{2\pi}\int_{-\pi}^{\pi} H_d(e^{j\psi})\, R_N(e^{j(\omega-\psi)})\, d\psi$$

As magnitudes de $R_N(e^{j\omega})$ e $H_d(e^{j\omega})$ são mostradas na figura 2. A sua convolução justifica o *ripple* observado na figura 1.

Janelas alternativas

Uma das formas de resolver o problema do *ripple* e efectivamente acabar como o "fenómeno de Gibbs", consiste em multiplicar $h_d[n]$ por uma janela não rectangular. Esta técnica denomina-se **windowing**. Há várias janelas em uso. Nas definições seguintes, as janelas existem entre $n = 0$ e $n = M$, e têm $M + 1$ pontos. São causais e possuem a propriedade de simetria relativamente a $(M + 1)/2$, valor este inteiro se M for ímpar (filtro do tipo I), ou então não inteiro, se M for par (filtro de tipo II).

Bartlett-triangular

$$w[n] = \begin{cases} 2n/M, & 0 \le n \le M/2 \\ 2 - 2n/M & M/2 < n \le M \\ 0, & \text{outros casos} \end{cases}$$

Hanning

$$w[n] = \begin{cases} [1 - \cos(2\pi n/M)]/2 & 0 \le n \le M \\ 0, & \text{outros casos} \end{cases}$$

Hamming

$$w[n] = \begin{cases} 0.54 - 0.46\,\cos(2\pi n/M) & 0 \le n \le M \\ 0, & \text{outros casos} \end{cases}$$

Blackman

$$w[n] = \begin{cases} 0.42 - 0.5\,\cos(2\pi n/M) + \\ +0.08\,\cos(4\pi n/M) & 0 \le n \le M \\ 0, & \text{outros casos} \end{cases}$$

Janela de Kaiser

A janela de Kaiser apresenta 2 graus de liberdade, β e $\alpha = M/2$, e por essa razão consegue emular e substituir qualquer das anteriores janelas.

Sendo $\alpha = M/2$ e $I_0(.)$ a função de Bessel[2] de ordem 0, a janela de Kaiser é definida por

$$w[n] = \begin{cases} \frac{I_0[\beta(1-[(n-\alpha)/\alpha]^2)^{1/2}]}{I_0(\beta)} & 0 \le n \le M \\ 0, & \text{outros casos} \end{cases}$$

No projecto com a janela de Kaiser, sendo as especificações do filtro passa-baixo

$$\Delta\omega = \omega_s - \omega_p \qquad A = -20\log\delta$$

o valor de β é calculado por (expressões empíricas, propostas por Kaiser)

$$\beta = \begin{cases} 0.1102(A - 8.7), & A > 50 \\ 0.5842(A-21)^{0.4} + 0.07886(A-21) & 21 \le A \le 50 \\ 0, & A < 21 \end{cases}$$

e M deverá satisfazer

$$M = \frac{A - 8}{2.285\,\Delta\omega}$$

Projecto por amostragem na frequência

Nesta abordagem, começa-se por considerar $\tilde{H}[k]$, a DFT da resposta impulsiva $h[n]$ que pretendemos descobrir

$$\tilde{H}[k] = H(z)|_{z=W_N^{-k}} = \sum_{n=0}^{N-1} h[n] W_N^{nk}$$

onde $W_N = e^{-j\frac{2\pi}{N}}$. A função de transferência $H(z)$ pode ser recuperada a partir das amostras $\tilde{H}[k]$

$$H(z) = \frac{1 - e^{-j\omega N}}{N} \sum_{n=0}^{N-1} \frac{\tilde{H}[k]}{1 - W_N^{-k} z^{-1}}$$

[2]Esta função é definida por $I_0(\beta) = 1 + \sum_{k=1}^{\infty}\left(\frac{(\beta/2)^k}{k!}\right)^2$.

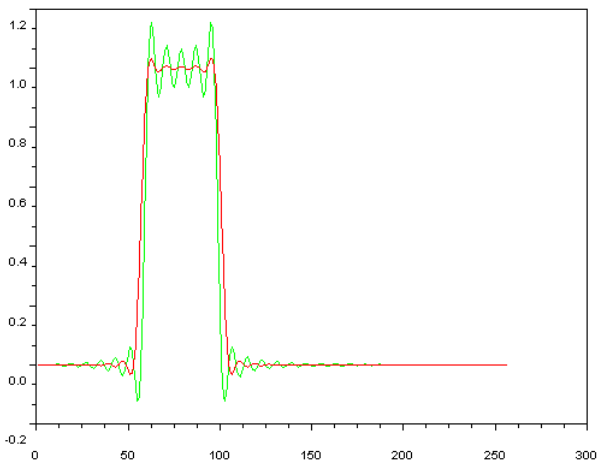

Fig. 3: Filtro passa-banda por amostragem na frequência, projectado com e sem "valor intermédio de transição".

e a respectiva resposta na frequência é

$$H(e^{j\omega}) = \frac{1-e^{-j\omega N}}{N} \sum_{n=0}^{N-1} \frac{\tilde{H}[k]}{1-W_N^{-k}e^{-j\omega}}$$

expressão que pode ser escrita como uma interpolação de sinusóides

$$H(e^{j\omega}) = \frac{1-e^{-j\omega[(N-1)/2]}}{N} \left\{ \sum_{n=0}^{N-1} \tilde{H}[k] e^{j\pi k(1-1/N)} \times \right.$$
$$\left. \times \frac{\sin[N(\omega-(2\pi/N)k)/2]}{\sin[(\omega-(2\pi/N)k)/2]} \right\}$$

Esta relação sugere uma abordagem "naive" ao projecto do filtro, i.e., fazer

$$\tilde{H}[k] = H_d(W_N^{-k}), \qquad k = 0, 1, \ldots, N-1$$

e "rezar" para que a fórmula de interpolação anterior "preencha os espaços" entre as amostras.

Na prática, isto não funciona bem. Um exemplo ilustrativo de um filtro passa-banda, com declive "abrupto", obtido por esta abordagem, é o gráfico a verde na figura 3 (com $N = 64$ amostras).

Porém, melhora-se significativamente o filtro quando não se faz uma transição abrupta entre os $\tilde{H}[k] = 0$ na(s) banda(s) de rejeição e os $\tilde{H}[k] = 1$ na(s) banda(s) de passagem do filtro, intercalando-se nesses pontos um valor numérico situado entre 0 e 1. Para ter ideia da diferença nos resultados, na figura 3 mostra-se a vermelho o projecto do mesmo filtro quando se alteraram os valores das amostras situadas nas duas transições da banda pretendida para $\tilde{H}[15] = \tilde{H}[26] = 1/2$. É visível que o filtro "vermelho" se aproxima muito mais do filtro ideal (um valor intermédio que conduz a um filtro de melhor qualidade é 0.4, em vez de 1/2).

Finalmente, deve-se ter em atenção que as técnicas apresentadas assentam em cálculos numéricos complicados que só são viáveis com o auxílio do computador. É aqui que entra o *Scilab*.

Realização de filtros

Usando as potencialidades do *Scilab*, projecte filtros FIR.

1. Projecte "manualmente" um filtro LP com frequência de corte $\omega_C = 0.15\pi$ pela técnica "da janela", variando as janelas (veja o texto e use o comando `window()`) e a dimensão (M ou N) da resposta impulsiva $h[n]$ do filtro e comparando os resultados. Comece com $h_d[n]$ em (4).

2. Compare o seu projecto anterior com projectos obtidos com os comandos `eqfir()`, `wfir()`, etc...

3. Projecte "manualmente", com a técnica da amostragem na frequência, um filtro LP com frequência de corte $\omega_C = 0.15\pi$ e um filtro BP com $\omega_0 = 0.5\pi$ e largura de banda $\Delta_B = 0.2\pi$. Vai necessitar de usar `fft()` e/ou `ifft()`. Mais uma vez, varie o número de amostras em frequência e teste o uso de "valores de transição".

4. Implemente os filtros anteriores com o comando `fsfirlin()` e compare os resultados.

Avalie as implementações no que respeita à linearidade da fase (isto é, use ondas quadradas na entrada $u[n]$ e observe a distorção na saída). Note que pode implementar os filtros FIR com `convol(h,u)`, para qualquer entrada $u[n]$.

Comandos *Scilab* para o projecto de filtros FIR

O *Scilab* tem vários comandos que servem para realizar este tipo de filtros. Reproduzimos aqui as descrições no "help" de alguns deles (não traduzidas, mas melhoradas).

- **[xm,fr]=frmag(num[,den],npts)** – calculates the magnitude of the frequency responses of FIR and IIR filters. The filter description can be one or two vectors of coefficients, one or two polynomials, or a rational polynomial. *npts* is the number of points. *fr* is the vector of frequencies where the magnitude is calculated, and depends of *npts*. (Notice that *xm* and *fr* are *outputs* of the command.)

- **[hn]=eqfir(nf,bedge,des,wate)** – is a Minimax (Chebyshev kind) approximation of a multi-band, linear phase, FIR filter. *nf* is the number of output filter points; *bedge* is a Mx2 matrix giving a pair of edges for each band; *des* is a M-vector giving the desired magnitude for each band; and *wate* is a vector of weights (usually all ones, if no band has priority).

- **wfir(ftype,forder,cfreq,wtype,fpar)** – linear-phase FIR filters, where: **ftype** is a string 'lp', 'hp', 'bp', 'sb' (filter type); **forder** is the filter order (odd for ftype='hp' or 'sb'); **cfreq** is a 2-vector of cutoff frequencies (0<cfreq(1), cfreq(2)<.5), and only cfreq(1) is used when ftype='lp' or 'hp'; **wtype** is the window type ('re', 'tr', 'hm', 'hn', 'kr', 'ch', see the *window()* command below, for details); **fpar** is a 2-vector of window parameters, and in case of Kaiser window fpar(1)>0 fpar(2)=0, and in Chebyshev window fpar(1)>0, fpar(2)<0 or fpar(1)<0, 0<fpar(2)<0.5; **wft** is a vector with the time domain coefficients $h[n]$; and **wfm** is the vector with the frequency domain response on the grid fr; **fr** is the frequency grid.

- **[x]=ffilt(ft,n,fl,fh)** - coefficients of FIR filter. Get *n* coefficients of a FIR filter. For low and high-pass filters, one cut-off frequency must be specified whose value is given in *fl*. For band-pass and stop-band filters, two cut-off frequencies must be specified for which the lower value is in *fl* and the higher value is in *fh*. *ft* is the filter type: 'lp', 'hp', 'bp', or 'sb'.

- **[hst]=fsfirlin(hd,flag)** – design of FIR, linear phase filters, using the **frequency sampling technique**.

- **window('string',n)** – computes symmetric window of various types. Dimension is *n*. 'string' is: *'re'* for rectangular; *'tr'* for triangular; *'hn'* for hanning; *'hm'* for Hamming; *'ch'* for Chebyshev; *'kr'* for Kaiser.

- **trans(hz,tr_type,frq)** or **trans(pd,zd,gd,tr_type,frq)** – low-pass to other filter transform: **hz** is the given filter transfer function; **pd** is the vector of given filter poles; **zd** is the vector of given filter zeros; **gd** is a scalar, the given filter gain; **tr_type** is a string, the type of transformation; **frq** is a 2-vector of discrete cut-off frequencies (i.e., 0<frq<0.5). For lp and hp filters, only frq(1) is used. For bp and sb filters, frq(1) is the lower cut-off frequency and frq(2) is the upper cut-off frequency; **hzt** is the transformed filter transfer function.

- **remez(nc,fg,ds,wt)** or **remezb(nc,fg,ds,wt)** – minimax approximation of a frequency domain magnitude response using Remez algorithm. The approximation takes the form h = sum[a(n)*cos(wn)]. An FIR, linear-phase filter can be obtained from the the output of the function.

Vejamos alguns exemplos.

Primeiro um simples FIR, utilizando uma rotina "enlatada":

```
// Simples Filtro FIR
hn=eqfir(33,[0 .2;.25 .35;.4 .5],
     [0 1 0],[1 1 1]);
[hm,fr]=frmag(hn,256); clf(); plot(fr,hm);
```

Agora um projecto de filtro FIR, por amostragem na frequência, com e sem amostras de transição:

```
// Amostras na frequência
hd=[zeros(1,15) ones(1,10) zeros(1,39)];
hst1=fsfirlin(hd,1);//filter
hd(15)=.5;hd(26)=.5;//samples in transition
hst2=fsfirlin(hd,1);// filter two
clf; plot(hst1,'g'); plot(hst2,'r');
```

Outros comandos

Convolução Linear

Operador: `convol(x,h)`. Realiza a convolução linear de $x[n]$ com $h[n]$.

Transformada Discreta de Fourier (DFT) e a FFT

Operadores: `fft(x)`, `ifft(X)`; extensão de vectores com `ones()` e `zeros()`.

O operador `fft()` calcula eficientemente a DFT de uma sequência utilizando o algoritmo denominado "Fast Fourier Transform" (FFT). O operador `ifft()` calcula a DFT inversa (IDFT).

A convolução linear de dois sinais de duração N_1 e N_2 é um sinal de duração $N = N_1 + N_2 - 1$. Para poder realizar convoluções lineares (as operações de processamento de sinal realizadas por SLITs discretos), utilizando a DFT/FFT, há que estender com uma sequência de zeros ("zero padding") os sinais originais, por forma que a convolução circular corresponda à convolução linear. Reveja a aula de laboratório anterior para recordar os detalhes.

Sinais de longa duração e a técnica de "overlap-add"

Na operação de convolução, $h[n]$ é um *kernel* de um filtro digital e tem dimensão limitada (no máximo, poucas centenas de pontos). Por seu lado, se a entrada $u[n]$ for um sinal que é amostrado em tempo real continuamente (sinal audio ou vídeo, sinal de um radar ou de um sonar, etc...) tem uma dimensão "infinita" e, aparentemente, só se poderia obter o sinal de saída $y[n]$ após $u[n]$ ter terminado.

Para efectuar processamento em tempo real, esta limitação tem de ser ultrapassada. Uma das técnicas usadas consiste em partir o sinal em blocos de dimensão L e, assentando no princípio da linearidade, fazer à vez a convolução destes blocos com $h[n]$ e adicionar correctamente (i.e., com a correcta translacção no tempo) os resultados parcelares para obter a solução total. Esta técnica denomina-se "overlap-add". Foi explicada nas aulas e falada nos laboratórios. Pode "reutilizá-la" para implementar filtros FIR com FFTs.