1 2 9 0 FACULDADE DE CIÊNCIAS E TECNOLOGIA UNIVERSIDADE DE COIMBRA

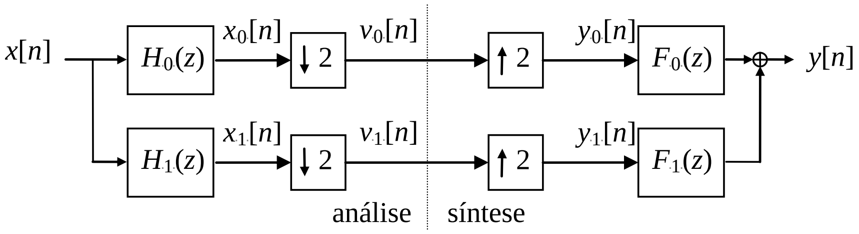
DEPARTAMENTO DE ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES

Processamento Digital de Sinal

Trabalho Prático Nº 4 – Banco de Filtros de 2 Canais de Reconstrução Perfeita

Descrição

Neste trabalho pretende-se implementar um banco de filtros de dois canais, maximamente decimado e de reconstrução perfeita (PR) com filtros QMF ("Quadrature Mirror Filters"). O esquema a implementar é o da figura seguinte, onde a relação entre os filtros deriva da solução paraunitária.



O filtro análise $H_0(z)$, FIR, de comprimento N=36, passa-baixo, com resposta a impulso $h_0[n]$, tem os seguintes coeficientes (não tem fase linear):

```
h0=[...
 0.074088731801707  0.313398921217245  0.596411460394921  0.572232242836783  0.136800277183280 -0.263693327291677,...
-0.188880560852887  0.128670638896525  0.161110885341112 -0.070664426843748 -0.127338395900939  0.044484718467773,...
 0.098090567102992 -0.031988886996564 -0.074005520243534  0.025406142155816  0.054360760452705 -0.021225452713612,...
-0.038501762462150  0.017864117805954  0.025969901555583 -0.014690181914019 -0.016410874668589  0.011563575682919,...
 0.009473939050820 -0.008573333083088 -0.004770298813289  0.005885767808813  0.001872030114699 -0.003660062664404,...
-0.000326048733191  0.001992796646105 -0.000296972271980 -0.000990559147496  0.000720924535220 -0.000170429382245];
```

Trabalho Prático

1. Construção do banco de filtros PR-QMF de 2 canais.

a) Copie os coeficientes $h_0[n]$ fornecidos (código Matlab) e coloque-os no script. Construa depois as respostas a impulso dos filtros $H_1(z)$, $F_0(z)$ e $F_1(z)$ da seguinte forma:

1) Defina a variável `h0til` com coeficientes por ordem inversa dos de `h0`;
2) Construa `h1` por troca de sinal dos coeficientes de `h0til` de ordem par (n=0,2,4,...,N-2);
3) Construa `f0` e `f1` com os valores de `h0` e `h1` por ordem inversa,

As respostas pretendidas correspondem às seguintes definições:

$$h_1[n] = (-1)^{n+1} h_0[N-1-n] = (-1)^{n+1} \tilde{h}_0[n]$$
$$f_0[n] = h_0[N-1-n] = \tilde{h}_0[n] \qquad , \; n = 0, \cdots, N-1 .$$
$$f_1[n] = h_1[N-1-n] = \tilde{h}_1[n]$$

b) Calcule a resposta em frequência dos 4 filtros usando DFTs de 1024 amostras (em vez de `freqz`). Defina um vetor de frequências de 513 pontos de forma a fazer um plot do módulo das respostas em frequência dos filtros de análise, no intervalo [0,π] (frequência em radianos), [0,1] (frequência normalizada) ou [0, f_s/2] (frequência em Hertz). Use letras maiúsculas para as respostas. Verifique que os pares de respostas apresentam simetria de espelho em quadratura (QMF).

c) Calcule a resposta em frequência global do banco de filtros sabendo que a função de transferência é $H(z) = \frac{1}{2}\left[H_0(z)F_0(z) + H_1(z)F_1(z)\right]$.

Verifique que o ganho é constante e que a fase é linear. Para tal faça o plot da fase de $H(e^{j\omega})$, com `unwrap(angle(H))/pi`, e determine (exatamente) o declive da reta de fase. Diga depois qual vai ser o atraso (constante), em amostras, provocado pelo sistema.

d) Calcule a resposta a impulso do sistema global (por transformada de z inversa da equação dada na alínea anterior; use o comando `conv`). Determine o ganho e atraso do banco de filtros. Para verificar que a reconstrução é perfeita, faça um plot desta resposta a impulso e verifique que é apenas um impulso. De seguida, faça um plot da resposta sem a amostra central (impulso) de forma a verificar que os valores obtidos são nulos (com 11 algarismos significativos corretos).

e) Determine a autocorrelação (determinística) de $h_0[n]$, $r[n]$, que corresponde à função $R(z) = H_0(z)H_0(z^{-1})$. Use o comando `r=conv(h0,h0til)`, mas tome em atenção que a amostra de índice 0 se situa a meio do vetor gerado: `plot(-(N-1):N-1,r)`. Verifique que esta autocorrelação é nula nas amostras pares, exceto em n=0, e que, portanto, se trata da solução paraunitária.

2. Implementação do banco de filtros de 2 canais

a) Implemente uma função em Matlab que, dadas duas respostas a impulso `h0` e `h1` bem como uma entrada `x`, retorna as duas saídas de um banco de filtros de 2 canais a serem transmitidas/armazenadas, `v0` e `v1` (ver figura inicial). Implemente também uma função para fazer a síntese. As funções devem ter os seguintes protótipos, e devem cumprir o que está enunciado no texto de ajuda (que se obtém fazendo `help Analysis2` e `help Synthesis2`).

```
function [v0,v1] = Analysis2(h0,h1,x)
%Aplica x[n] ao banco de filtros de 2 canais com respostas a impulso
%h0 e h1 e decima as saídas por 2. Retorna estas saídas, v0 e v1, com metade
%do comprimento de x. A entrada x deve ser um vetor coluna com comprimento par.

function y = Synthesis2(f0,f1,v0,v1)
%Expande v0[n] e v1[n] por 2 e aplica-os ao banco de filtros de síntese de 2
%canais com respostas a impulso f0 e f1.
%As entradas v0 e v1 têm de ter o mesmo comprimento.
%Sintetiza a saída, y, num vetor coluna de comprimento duplo de v0 e v1.
```

Deve testar (dentro das rotinas) as condições definidas na ajuda, e deve emitir mensagens de erro quando estas não forem verificadas. Deve usar o comando `filter` (e não `conv`, pois `conv` gera sinal de comprimento `Nx+Nh-1`) para que entrada e saída tenham o mesmo comprimento.

b) Use as funções para analisar e sintetizar um sinal de áudio com os filtros calculados. Meça o máximo erro de reconstrução e comente.

No final da aula entregue o script que produziu, na "Submissão de Trabalhos" do Nónio.

Apêndice
Representação de números em vírgula flutuante (*floating point*) com 64 bits: double.

Processamento Digital de Sinal

Trabalho Prático Nº 4 – Banco de Filtros de 2 Canais de Reconstrução Perfeita

Os números em formato IEEE com precisão `double` têm a seguinte representação binária:

```
|1| 11 |    52    |
|s|exp |mantissa|
```

(Nº de bits: total 64)
(sinal, s (1 bit), expoente, exp (11 bits) e mantissa 52 bits)

sendo o número dado por: $x = (1-2s)\times 2^{\exp-1023}\times\left(1+\sum_{n=1}^{52} b_n 2^{-n}\right)$,

onde s é sinal do número (1 bit: 1=negativo; 0=positivo); exp é o expoente base 2 com excesso 1023, e `mantissa` é o valor fracionário na forma <`1,mantissa`> com bits b_n e onde o um à esquerda da vírgula é assumido. Assim, o número 1,0 tem representação em hexadecimal: `num2hex(1)=` `3FF0000000000000` (s=0; exp=0x3FF=1023 e mantissa=0). O próximo número é `3FF0000000000001`, com um bit a 1 na última posição da mantissa, representando 2^{-52}, sendo então o número $1+2^{-52}$. Logo a diferença para 1 é `eps`=2^{-52} =2.22E-16. Este facto indica que não podemos ter precisão maior que cerca de 15 algarismos significativos na representação com números em formato *double* de 64 bits.

Verifique o seguinte resultado e explique porque não é zero:

```
log(sqrt(2)^2)/log(2) - 1 %devia ser zero...
num2hex(log(sqrt(2)^2)/log(2)) %verificar qual o bit da mantissa responsável.
```

Use esta ajuda: `help eps` e o tópico `Floating-Point Numbers`.

O número máximo nesta representação é um número com expoente 2046=1023+1023 (pois `exp`=2047 está reservado para `Inf` e `NaN`), e com a mantissa completamente preenchida de 1's, a que corresponde o valor (ver abaixo) de $2-2^{-52}$:

```
dec2hex(2046) %7FE
hex2num('7FEFFFFFFFFFFFFF')%1.797693134862316e+308 =  2^1023*(2-2^-52)
realmax                    %1.797693134862316e+308
num2hex(inf) %7FF0000000000000
num2hex(nan) %FFF8000000000000
```

A mantissa completamente preenchida corresponde a: $\sum_{n=0}^{52} 2^{-n} = \frac{1-2^{-53}}{1-\frac{1}{2}} = 2-2^{-52} \approx 2$. Assim, o maior número é `realmax=` $2^{1023}\times\left(2-2^{-52}\right) = 2^{1024}-2^{971} \approx 2^{1024}$. Mas o valor 2^{1024} já é representado como infinito (`Inf = 7FF0000000000000`).

Da mesma forma, o número positivo mais pequeno que se pode representar tem expoente de base 2 igual a –1022 (`exp` = 1 = –1022+1023) e mantissa 0:

```
2^(-1022) % 2.225073858507201e-308
realmin   % 2.225073858507201e-308
```

O valor `exp=0` é reservado para representar o zero (0.0) que tem todos os 64 bits a zero, o que é muito conveniente em inicialização de *arrays* com valores a zero em memória (8 bytes a zero por cada double).