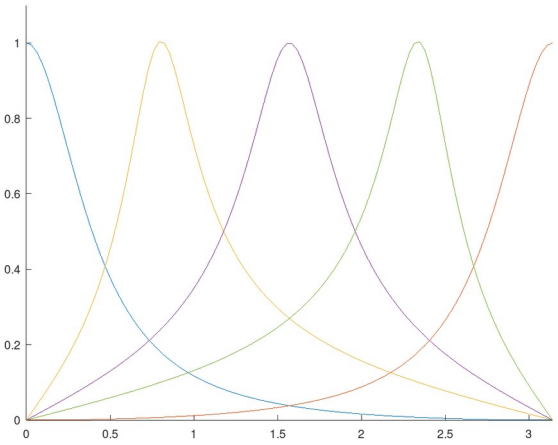
CompMus 2024 – Segundo Exercício-Programa
Equalizador Programável
Entrega da 1ª fase: 03/11/24 até 23:55
Entrega da 2ª fase: 17/11/24 até 23:55

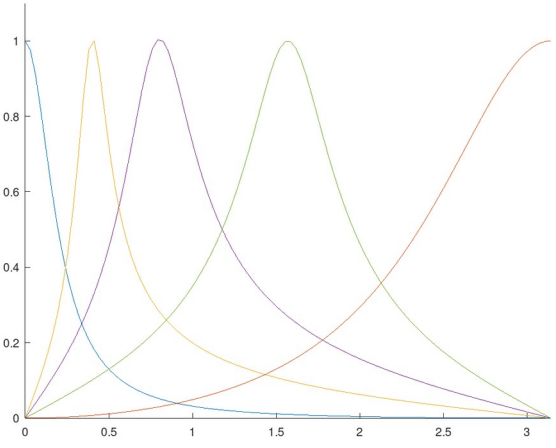
O objetivo deste trabalho é implementar um equalizador programável para sinais de áudio estéreo. A entrega está dividida em duas fases: na primeira fase faremos um equalizador com um número fixo de 5 faixas de frequências definidas por parâmetros do objeto, e na segunda fase generalizaremos a construção para um número arbitrário de faixas de frequências.

Especificação da primeira fase

Na primeira fase iremos partir de uma implementação existente `equalizador~.pd` de um equalizador de N=5 faixas, que pode ser baixado do e-disciplinas. O equalizador é implementado como um banco de N filtros passa-faixas em paralelo, sendo que o ganho de cada faixa é controlado por um slider vertical. Os filtros passa-faixas possuem 2 polos e 2 zeros, sendo que cada par de polos (complexo conjugados) é "afinado" em uma das frequências centrais, e tem sua magnitude determinada por uma condição de sobreposição das respostas em frequência do banco de filtros. A figura ao lado ilustra o banco de filtros do `equalizador~.pd` original. O patch `testeequalizador~.pd` ilustra o funcionamento do equalizador.

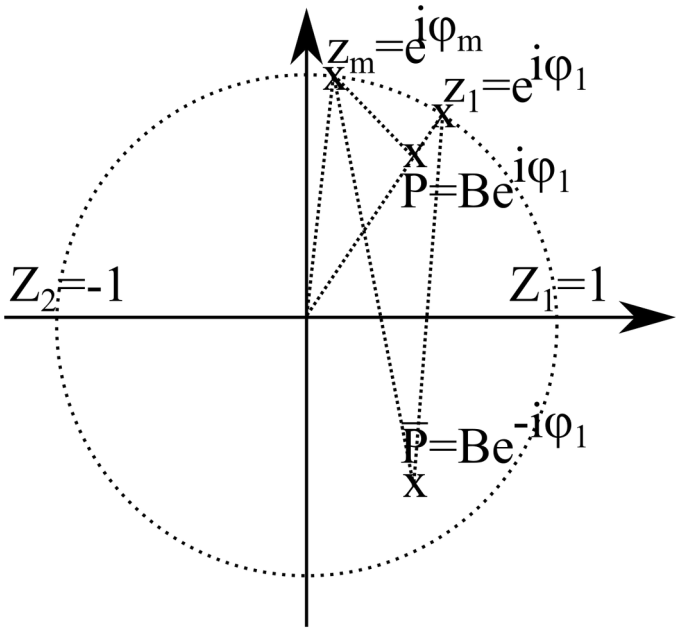


As principais diferenças entre aquele equalizador e o que deveremos entregar são: (1) o equalizador fornecido possui 5 faixas fixas, com frequências centrais de $0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}, \pi$ radianos/amostra[1] (ou equivalentemente 0, 6000, 12000, 18000 e 24000 Hz quando R=48000), ao passo que nosso equalizador possuirá 5 faixas programáveis; a figura ao lado ilustra uma possível aplicação, onde as faixas estão divididas em intervalos de oitava (razão 2:1 entre frequências centrais); (2) as posições dos polos nos filtros do `equalizador~.pd` original, que refletem as frequências centrais das 5 faixas fixas, na nova implementação deverão ser calculadas em função das frequências centrais desejadas; e (3) o equalizador fornecido possui apenas um canal, enquanto o nosso deve ser estéreo, ou seja, ter 2 **[inlet~]** e 2 **[outlet~]**. São poucas coisas para mudar em relação à implementação dada, por isso essa fase terá um prazo de entrega mais curto, a fim de mergulharmos rapidamente no contexto do EP2.



1 Não custa lembrar que cada frequência *F* em Hz corresponde ao ângulo 2πF/R, onde *R* é a taxa de amostragem.

Sua implementação deve computar as fórmulas que inicializam os parâmetros dos polos a partir das 5 frequências (em Hz, entre 0 e 24000, inclusive) informadas na criação do objeto. O exemplo da última figura corresponde ao **[equalizador~ 0 3000 6000 12000 24000]**, que corresponde às frequências angulares de $0, \frac{\pi}{8}, \frac{\pi}{4}, \frac{\pi}{2}, \pi$ (rad/amostra). A lista de parâmetros pode ser acessada dentro do patch com um **[loadbang]→|args<→[pdcontrol]→** e a partir dessa saída é possível processar a lista de frequências (em Lua ou em Pd puro) para inicializar os filtros individuais. Considere que as frequências estão em ordem crescente. Para cada frequência central F_k Hz, devemos calcular o ângulo $\varphi_k = 2\pi F_k / R$, para então determinar um dos polos $P_k = B_k e^{i\varphi_k}$ do filtro de 2 polos e 2 zeros. A magnitude B e o ângulo φ são usados para inicializar cada abstração **[2polos2zeros~ B phi]**, também fornecida com o enunciado, e que cria automaticamente um *filtro normalizado* com polos $Be^{i\varphi}$ e $Be^{-i\varphi}$, além de zeros Z_1 e Z_2 em DC e Nyquist; a própria abstração se encarrega de jogar os 2 zeros em Nyquist se os polos estão em DC, e de jogar os 2 zeros em DC se os polos estão em Nyquist, além de normalizar o filtro resultante, assim não teremos que nos preocupar com nada além da definição de um dos polos. A magnitude B de cada polo deve ser calculada de tal forma a compatibilizar a largura de banda com um dos filtros vizinhos: se um filtro de frequência (angular) φ_1 (não necessariamente o primeiro da lista) possui um filtro vizinho (à esquerda ou à direita) de frequência angular φ_2, então na frequência intermediária $\varphi_m = \frac{\varphi_1 + \varphi_2}{2}$ gostaríamos que os dois filtros tivessem ganho de $\frac{1}{2}$ (veja as duas figuras anteriores para ganhar intuição). Uma forma de determinar a magnitude B do polo do filtro de frequência φ_1 que satisfaz a condição anterior em relação à frequência φ_m pode ser deduzida da figura ao lado. Se $H(z) = a_0 \frac{(z-1)(z-(-1))}{(z-P)(z-\overline{P})}$ é a função de transferência do filtro associado ao polo $P = Be^{i\varphi_1}$, e $z_1 = e^{i\varphi_1}$ e $z_m = e^{i\varphi_m}$ são os elementos do círculo unitário representando as frequências φ_1 e φ_m, queremos garantir que $|H(z_m)| = \frac{1}{2}|H(z_1)|$, o que pode ser feito minimizando-se a discrepância $\left| |H(z_m)| - \frac{1}{2}|H(z_1)| \right|$ em função da variável $B = |P|$. Um valor aproximado da solução $B = |P|$ pode ser obtido por busca binária no intervalo [0...0.999], usando-se o código `Botimo.pd` disponível no e-disciplinas.



Note que a estratégia acima é apenas uma heurística, pois calcular B a partir de φ_1 e φ_2 não garante que o filtro em φ_2 também terá ganho de $\frac{1}{2}$ na frequência intermediária $\frac{\varphi_1 + \varphi_2}{2}$. Se tentássemos definir a magnitude do filtro em φ_2 em função de φ_1, o problema apenas mudaria de lugar, pois os outros vizinhos não teriam a mesma propriedade. Para simplificar o desenho dos filtros, consideraremos que a magnitude de cada polo será calculada em função do filtro vizinho *que estiver na direção do centro* $\frac{\pi}{2}$ *do espaço de frequências angulares*, ou seja, para $\varphi_1 < \frac{\pi}{2}$ tomaremos como vizinho o menor valor $\varphi_2 > \varphi_1$ (se existir) e para $\varphi_1 \geq \frac{\pi}{2}$ escolheremos

como vizinho o maior valor $\varphi_2 < \varphi_1$ (se existir). Quando não existir um vizinho nessas condições, usaremos a frequência extrema na direção do centro (ou seja, para $\varphi_{max} < \frac{\pi}{2}$ usaremos $\varphi_2 = \pi$ e para $\varphi_{min} \geq \frac{\pi}{2}$ usaremos $\varphi_2 = 0$.

O tratamento de sinais de áudio estéreo será feito duplicando-se o banco de filtros: o canal esquerdo deve ser tratado por uma das cópias do banco de filtros, e o canal direito pela outra. As frequências centrais e larguras de banda são rigorosamente as mesmas, os sliders são compartilhados, apenas o processamento de cada canal deve ser feito de forma independente.

Nas duas fases sua implementação deve corresponder a um único arquivo de nome `equalizador~.pd`. Você usará as abstrações **[2polos2zeros]** e **[Botimo]**, mas não deve alterá-las e nem entregá-las no e-disciplinas.

Especificação da segunda fase

Na segunda fase seu equalizador deve ser adaptável a uma quantidade N arbitrária de faixas de frequência, definidas pelo usuário a partir da criação do objeto de acordo com o número de argumentos. Assim **[equalizador~ 0 6000 12000 18000 24000]** deve gerar um equalizador idêntico ao disponibilizado originalmente, ao passo que **[equalizador~ 0 3000 6000 9000 12000 15000 18000 21000 24000]** deve produzir um equalizador com as 9 faixas de frequência indicadas.

Essa generalização depende do uso de técnicas de *metaprogramação*, também conhecida como *dynamic patching* em Pd, que vimos na Tarefa Prática 4. Como vimos, a maneira mais simples de controlar a geração dinâmica de um patch é começar a partir de um patch contendo um único subpatch **[pd meta]**, e acrescentar instruções para a criação de objetos copiando todos os seus parâmetros de inicialização a partir de versões dos mesmos objetos geradas manualmente. Lembre-se sempre também de apagar os objetos gerados automaticamente antes de salvar seu patch.

Algumas dicas e sugestões complementares a esse enunciado serão postadas no fórum durante o período de desenvolvimento. Participe compartilhando suas dúvidas por ali.

Finalmente...

- Esse trabalho é individual: conversar com os colegas para tirar dúvidas da linguagem é absolutamente normal, mas compartilhar soluções específicas e códigos não...
- Leia o enunciado mais de uma vez. É comum surgirem dúvidas que já estão respondidas no enunciado, mas que não demos atenção na primeira leitura.
- Use o fórum para tirar dúvidas!
- Entregue quantas versões parciais você quiser, antes do prazo, por precaução.
- Divirta-se programando!