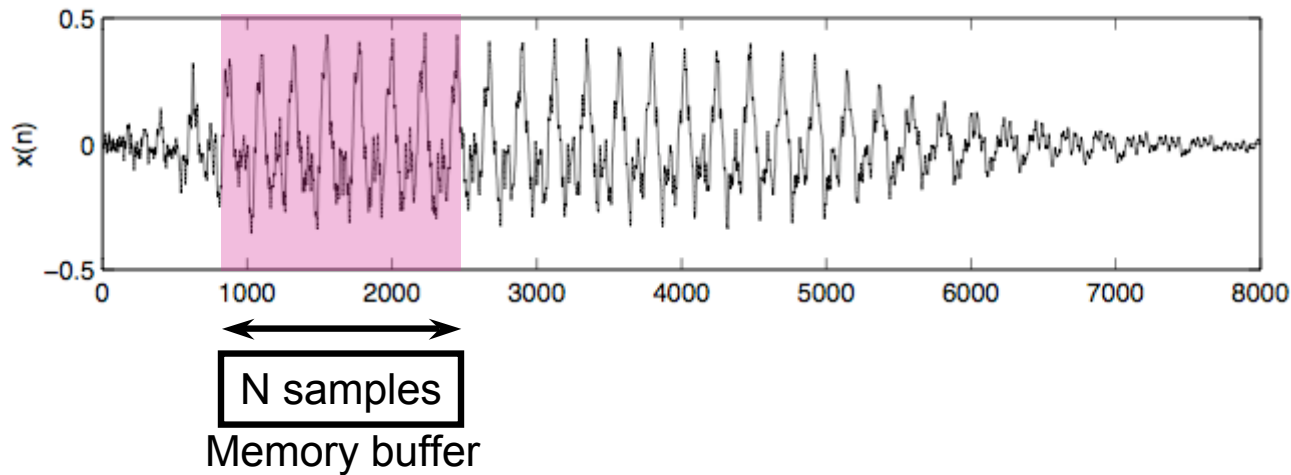


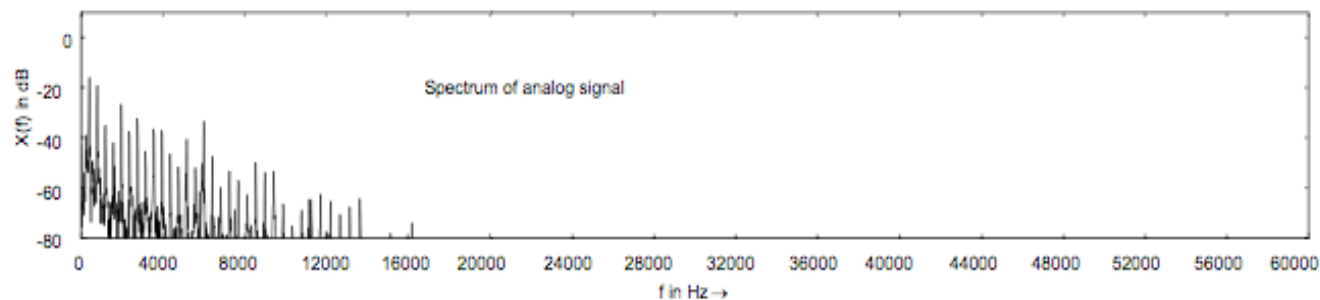
Digital Signal Processing

Juan P Bello

Block processing and spectrum



- For Block processing, signal data is sent to a buffer and processed as a block. The buffer is then filled with new data.
- A common example is spectral analysis using the DFT.
- The spectrum of a signal's segment shows the energy distribution across the frequency range



Discrete Fourier Transform

- The spectrum of a digital signal, $x(n)$, can be calculated as:

$$X(k) = DFT[x(n)] = \sum_{n=0}^{N-1} x(n)e^{-j2\pi nk/N}$$

$$k = 0, 1, \dots, N - 1$$

- The resulting N samples $X(k)$ are complex-valued:

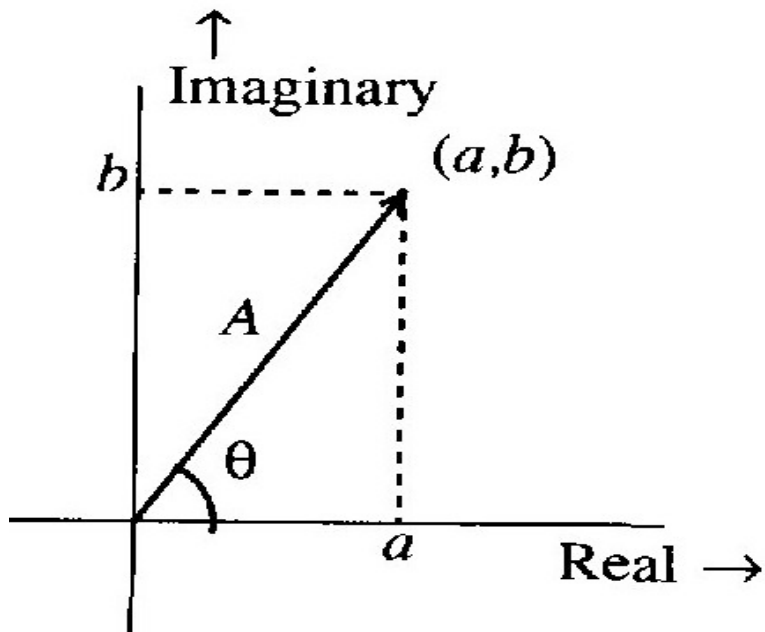
$$X(k) = X_R(k) + jX_I(k)$$

$$|X(k)| = \sqrt{X_R^2(k) + X_I^2(k)}$$

$$\varphi(k) = \arctan \frac{X_I(k)}{X_R(k)}$$

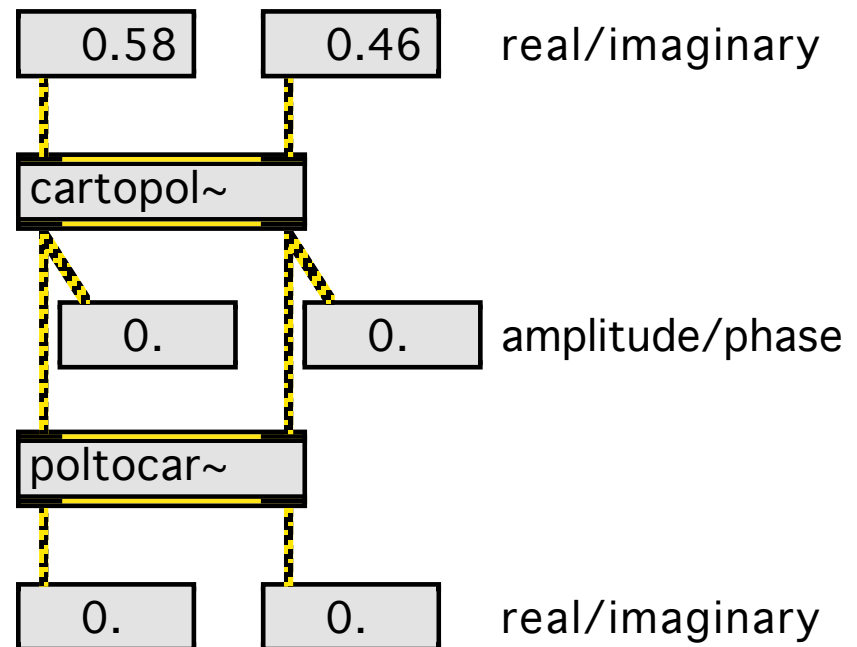
$$k = 0, 1, \dots, N - 1$$

MSP: Cartesian to Polar



$$A = \sqrt{re^2 + im^2}$$

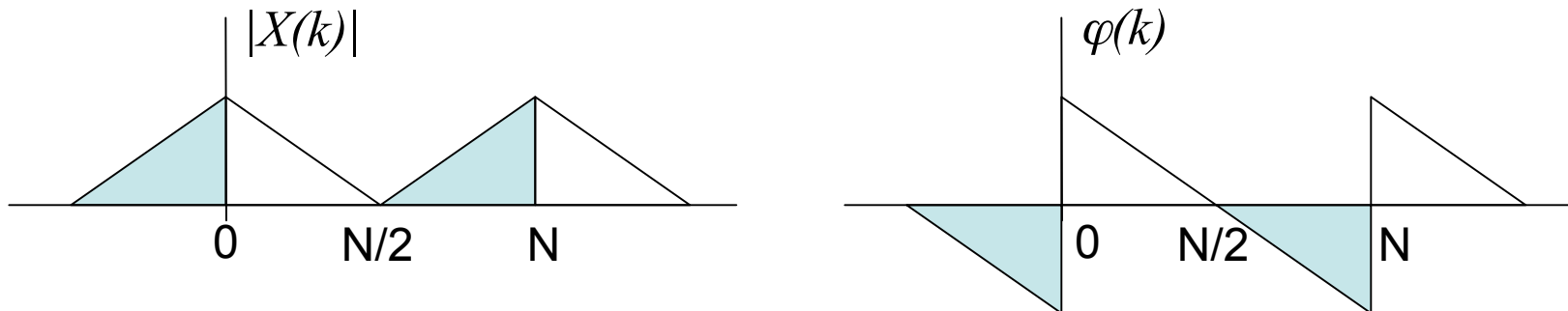
$$\theta = \tan^{-1} \frac{im}{re}$$



Discrete Fourier Transform

- The resulting spectrum is composed of N equidistant frequency points from 0 to $(N-1)f_s/N$ Hz in steps of f_s/N
- If the N samples $x(n)$ are real-valued (as in the case of audio signals) then the N DFT samples can be defined in terms of conjugate pairs of the form:

$$X(k) = X^*(N - k)$$



- That means that the DFT of a real-valued signal $x(n)$ is half-redundant. The complete information is obtained by looking at $X(k)$, $k = 0, 1, \dots, N/2$ (frequencies up to $f_s/2$)

Inverse DFT (IDFT)

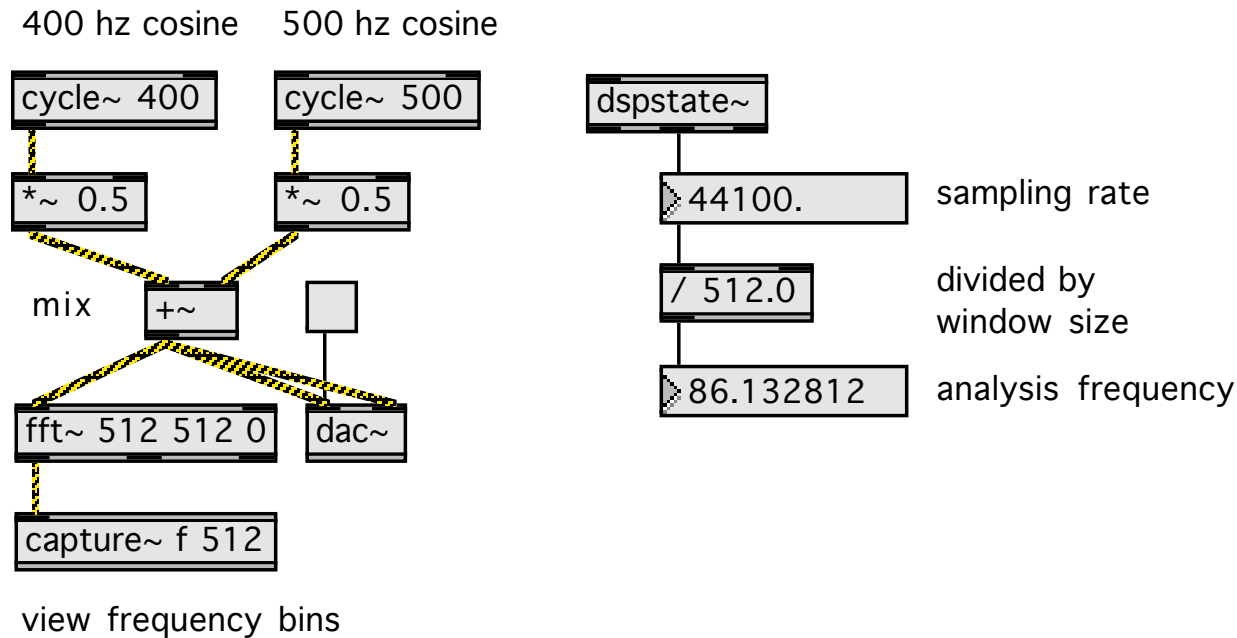
- The IDFT allows for the transformation of spectra in discrete frequency to signal in discrete time.
- It can be calculated as follows:

$$x(n) = IDFT[X(k)] = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{-j2\pi nk/N}$$

$$n = 0, 1, \dots, N - 1$$

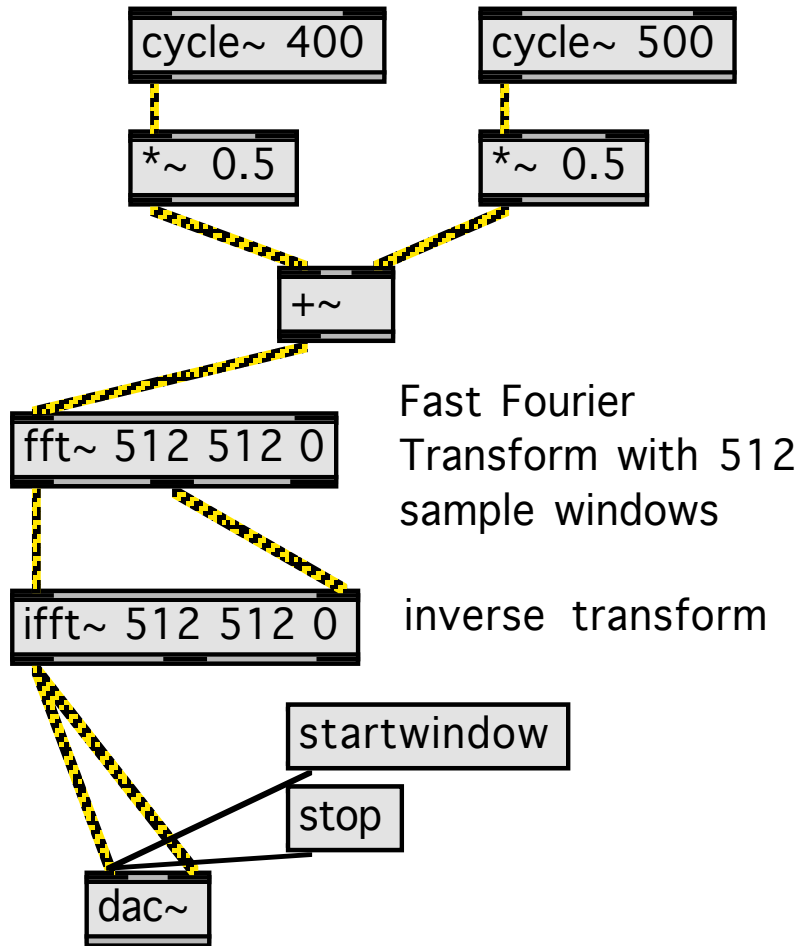
- The fast version of the DFT is known as the Fast Fourier Transform (FFT) and its inverse as the IFFT. The FFT is an algorithm to compute the DFT, usually $O(N^2)$ operations long, in $O(N \log N)$ operations
- Furthermore, there are a number of tricks to express the IDFT in terms of the FFT
- The FFT is so fast that even time-domain operations, like convolution, can be performed faster using FFT and IFFT instead.

MSP: Fast Fourier Transform



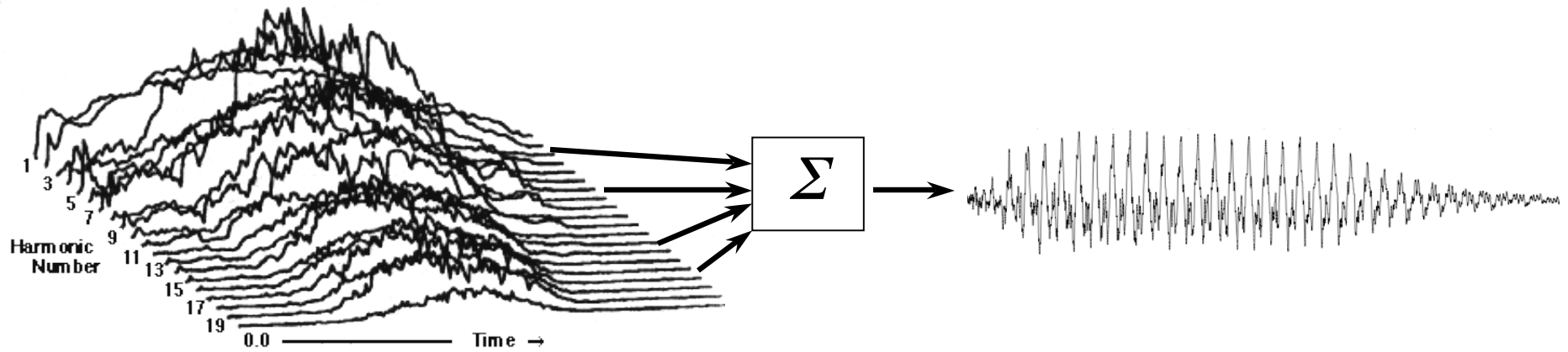
- The only constraint on the Fast Fourier Transform implementation is that the window size must be a power of two (e.g. 512, 1024, 4096)
- The MSP object `fft~` executes the fast fourier transform on the input signal.

MSP: Inverse Fourier Transform

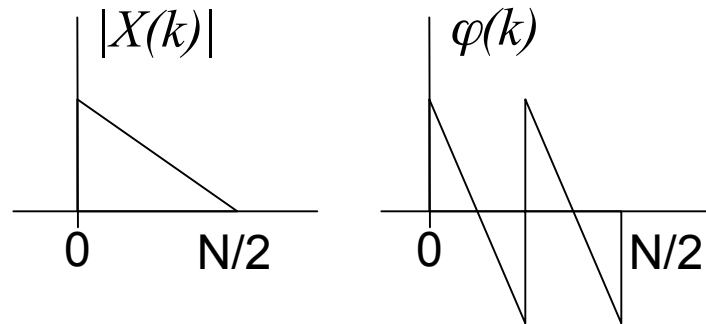


- The ifft~ object expects buffers of real and imaginary numbers identical to those output by fft~.
- This patch simply echoes signal input to fft~.

What is Fourier saying?

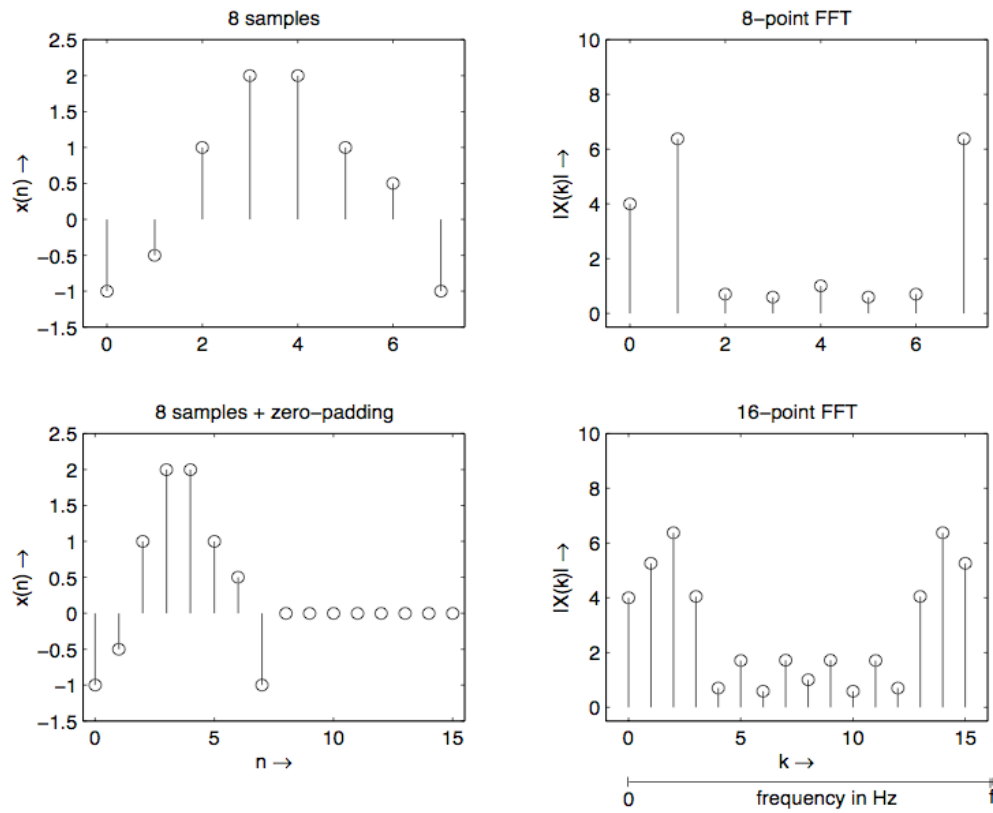


- Any periodic sound can be described by the summation of a number of sinusoids with time-varying amplitudes and phases
- Thus a complex spectrum is just a snapshot of those sinusoids' parameters



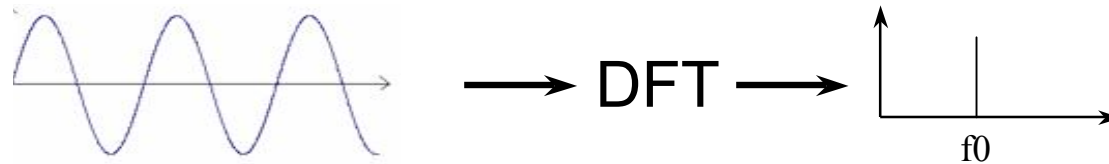
Frequency resolution

- As we now know, the frequency resolution is $\Delta f = f_s/N$.
- It can be seen that to increase resolution we need to increase N
- However that implies a loss of temporal resolution
- A possible solution is to zero-pad, i.e. to add zero-valued samples until we reach the desired N-length.

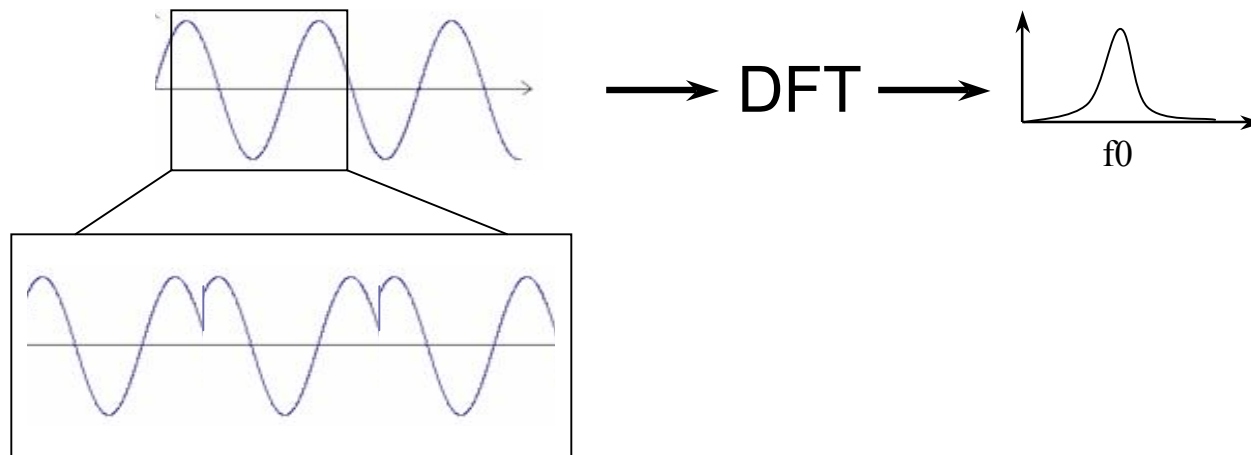


Leaking

- In theory the DFT of a sinusoid shows one spectral line at f_0



- In practice, unless we perform f_0 -synchronous analysis, there are discontinuities (sharp changes) at the segment boundaries that introduce some noise. Thus the spectral line around f_0 is smeared.
- This is known as spectral leaking

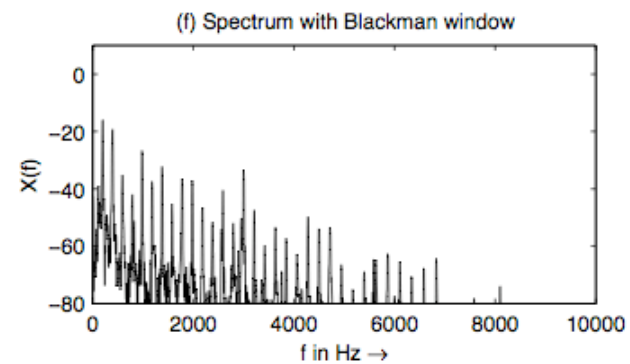
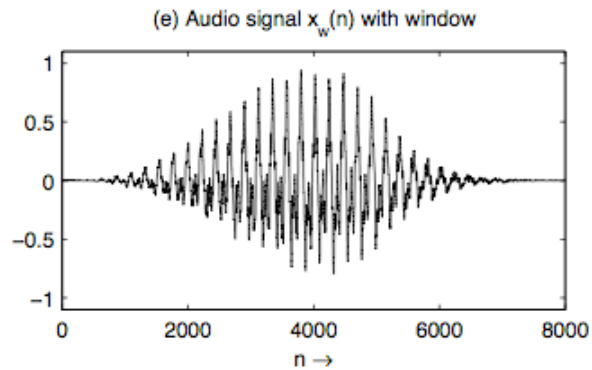
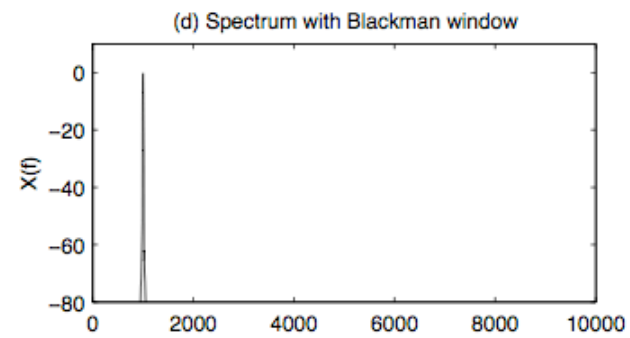
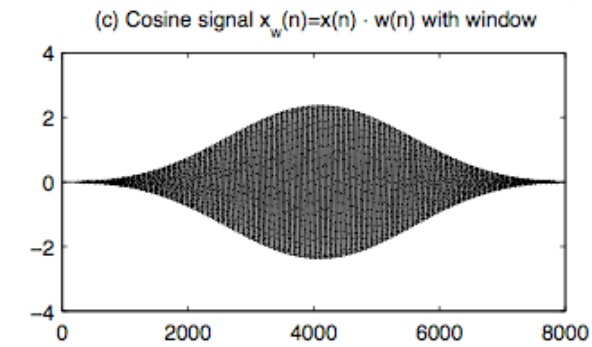
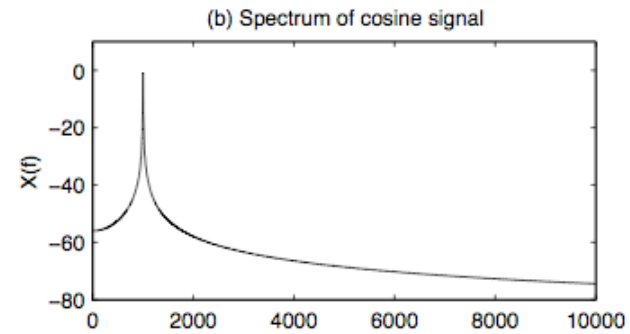
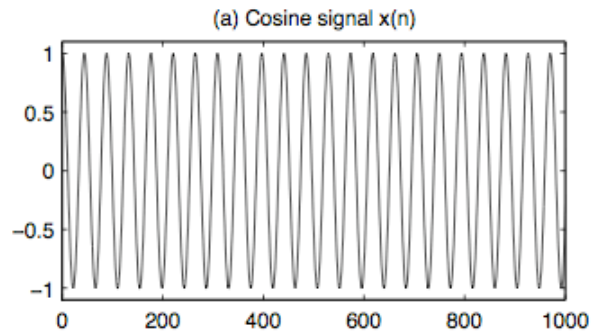


Windowing

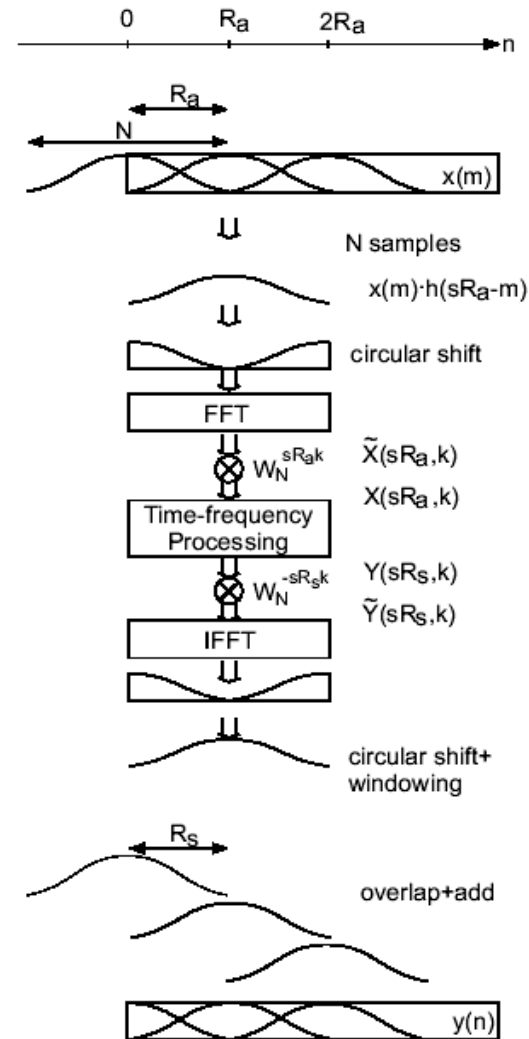
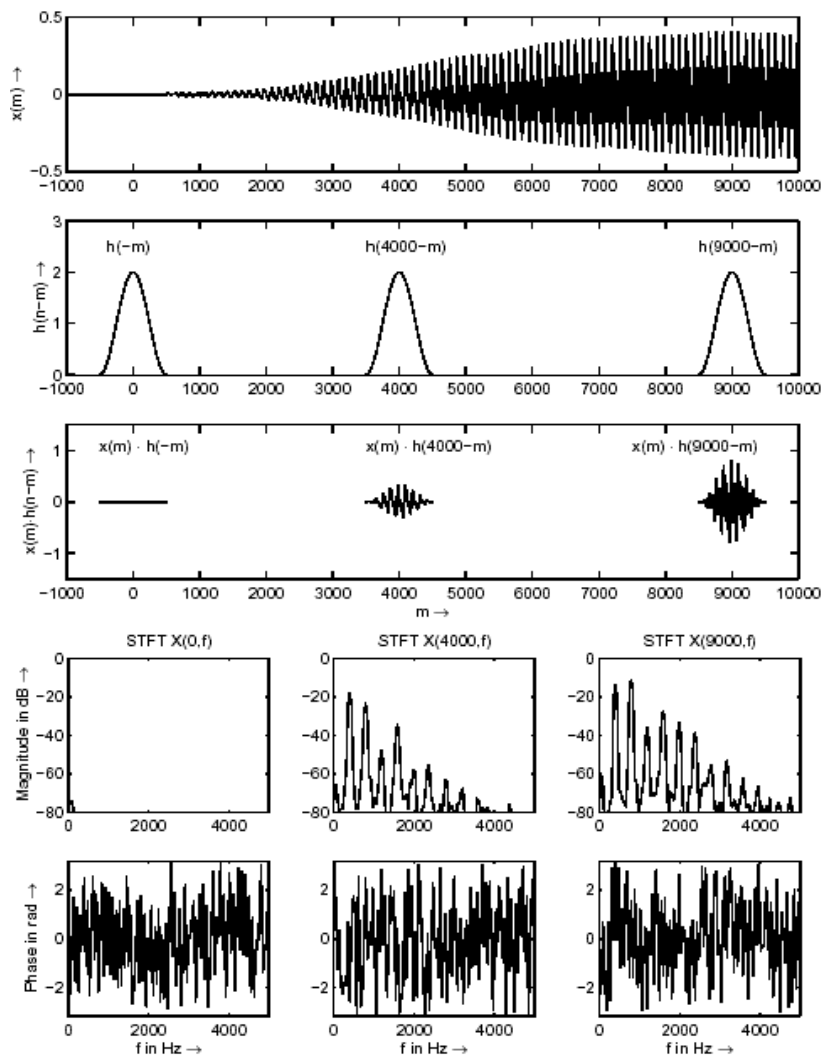
- Segmenting is equivalent to multiplying the signal by a N-length rectangular window of unitary amplitude.
- Multiplication in time-domain is equivalent to convolution in the frequency domain
- The transform of a rectangular window is a Sinc function ($\sin(x)/x$).
- We can have $N = kT_0$, where k is a positive integer, thus eliminating the discontinuities.
- Alternatively we can use a window that smoothly reduces the signal to zero at the boundaries
- Possible examples include Hamming (w_H), Blackman (w_B), Hanning, Triangular, Gaussian and Kaiser-Bessel windows.

$$\begin{aligned}w_B(n) &= 0.42 - 0.5 \cos(2\pi n/N) + 0.08 \cos(4\pi n/N), \\w_H(n) &= 0.54 - 0.46 \cos(2\pi n/N) \\n &= 0, 1, \dots, N - 1.\end{aligned}$$

Windowing

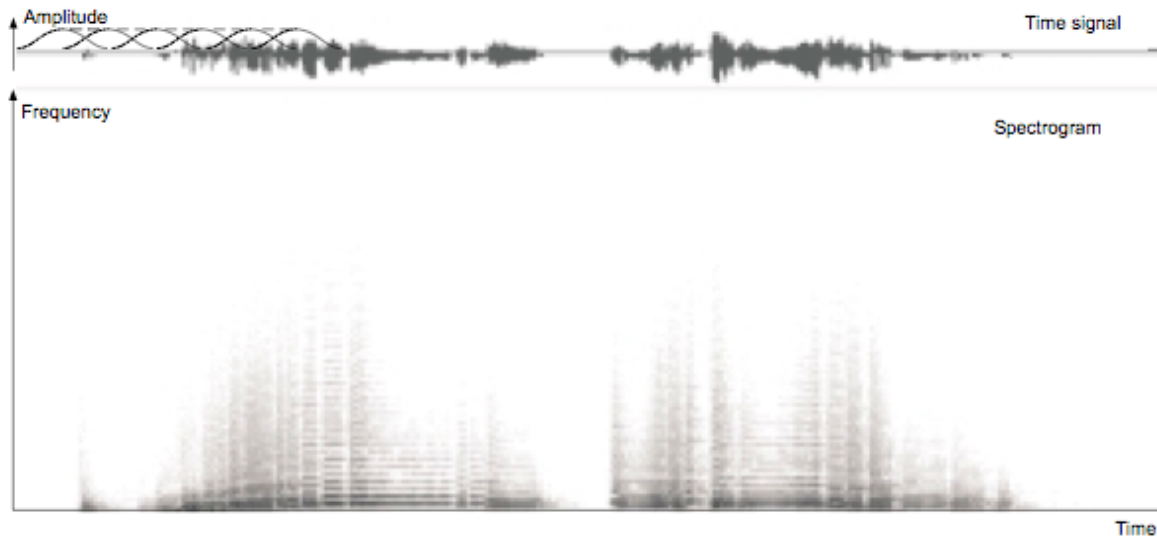


Short-time Fourier Transform

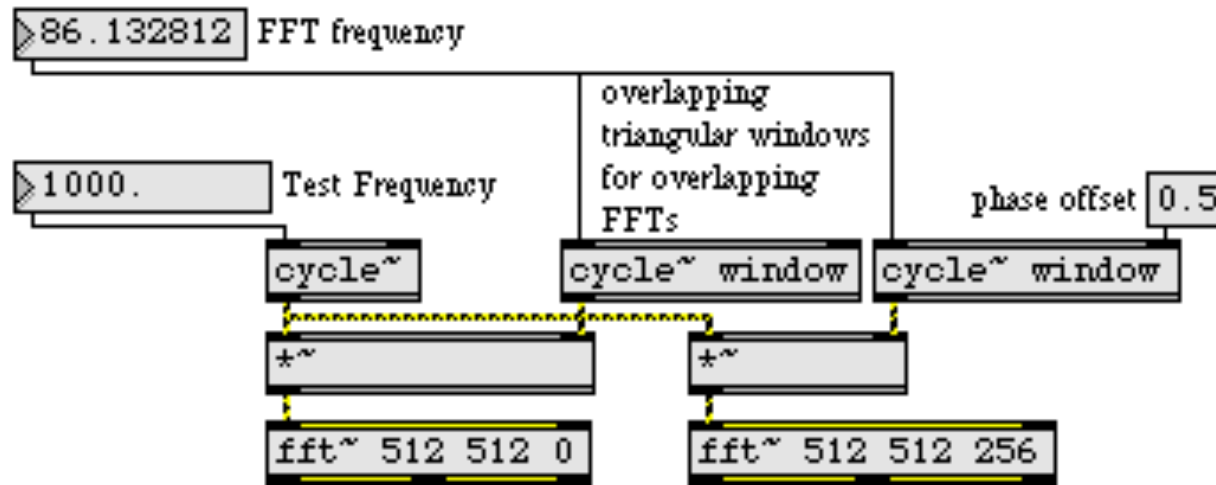


Time-frequency representation

- The Short-time Fourier Transform (STFT)
- Independent DFTs are calculated on windowed segments
- The segments usually overlap to compensate for the loss of temporal resolution
- Produces a spectrogram (or phasogram)



MSP: the STFT

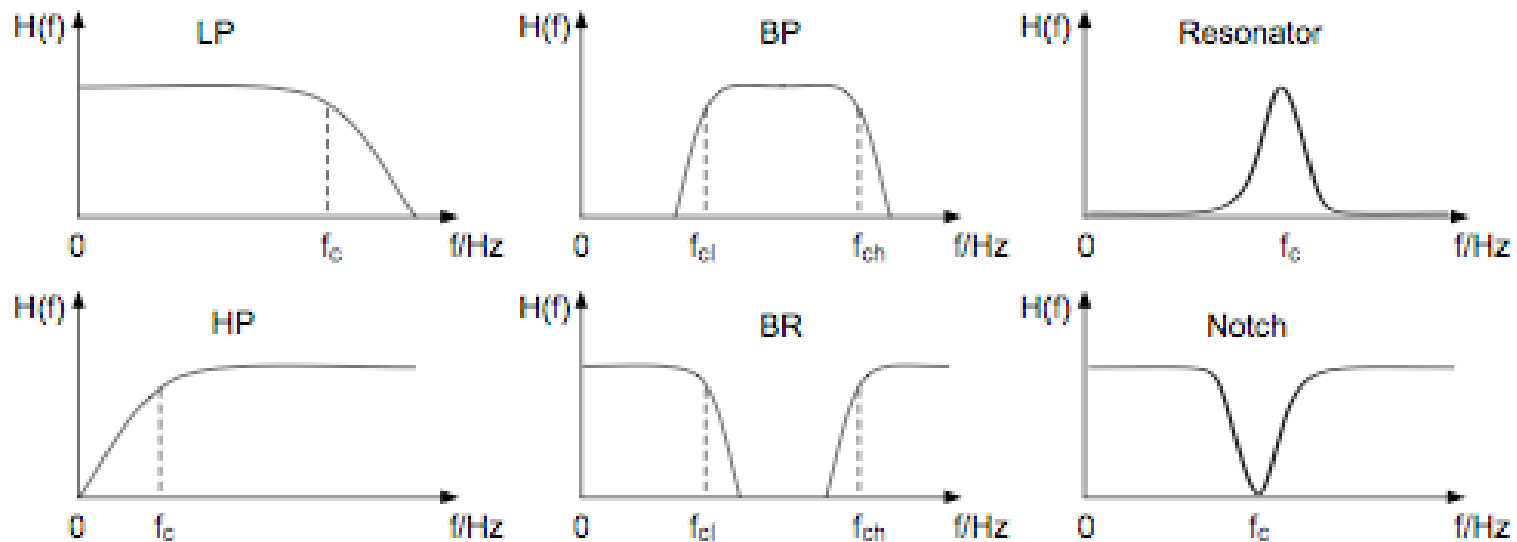


One FFT is taken 256 samples later than the other

- The arguments to `fft~` are window size, FFT period (number of samples between ffts), and phase (offset from beginning of period when fft is performed). This patch applies triangular windows to two overlapping fourier analyses.

Filters

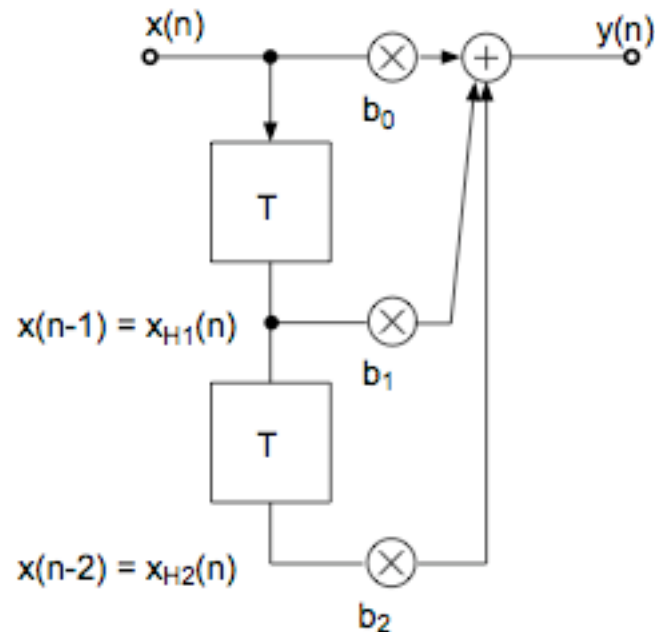
- Any process that selects a few elements from a large set of data
- In digital signal processing they are usually defined by their ability to reject, attenuate, retain or emphasize signal components
- According to this, the usual types of filters fall into one of these categories: low-pass, high-pass, band-pass, band-reject, resonator, notch and all-pass filter.
- Other complex filters can be defined as combinations of these.



- How are they implemented as digital systems?

FIR filters

- A FIR filter is a system with a finite impulse response $h(n)$
- A basic FIR filter can be described as a feed-forward delay line of the form:



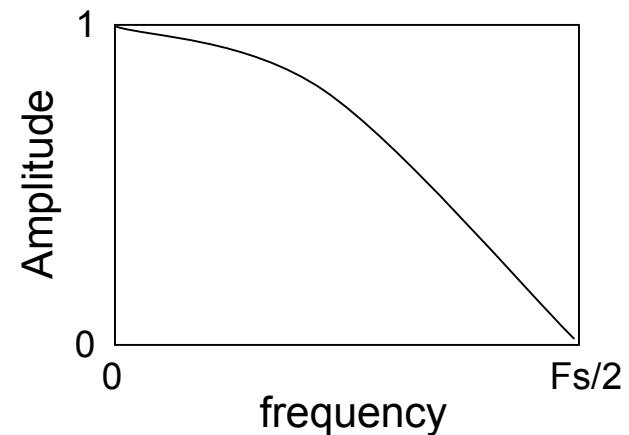
- Such that its transfer function is:

$$H(z) = b_0 + b_1 z^{-1} + b_2 z^{-2}$$

Simple FIR filter implementations

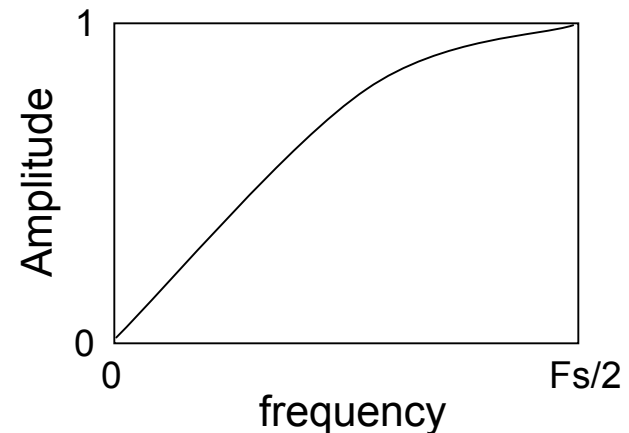
- Low-pass filter (averaging filter):

$$y(n) = \frac{x(n) + x(n-1)}{2}$$



- High-pass filter (differentiator):

$$y(n) = \frac{x(n) - x(n-1)}{2}$$



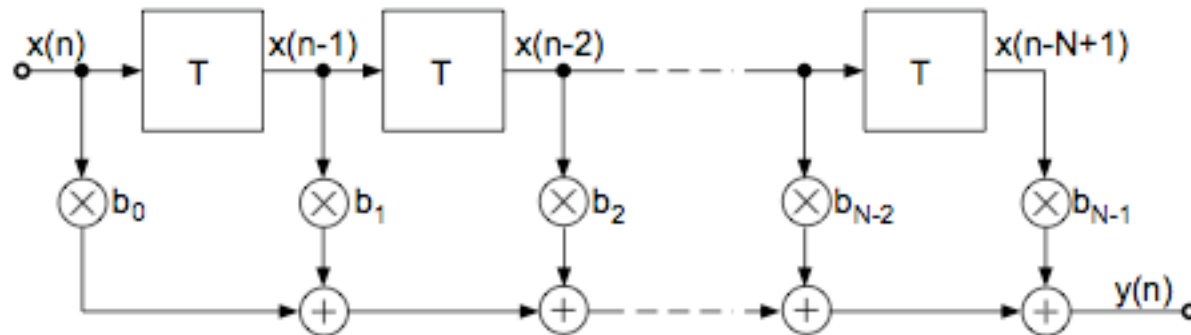
FIR filters

- More generally, a FIR filter of $N-1$ delay elements is described by the difference equation:

$$y(n) = \sum_{k=0}^{N-1} b_k x(n-k)$$

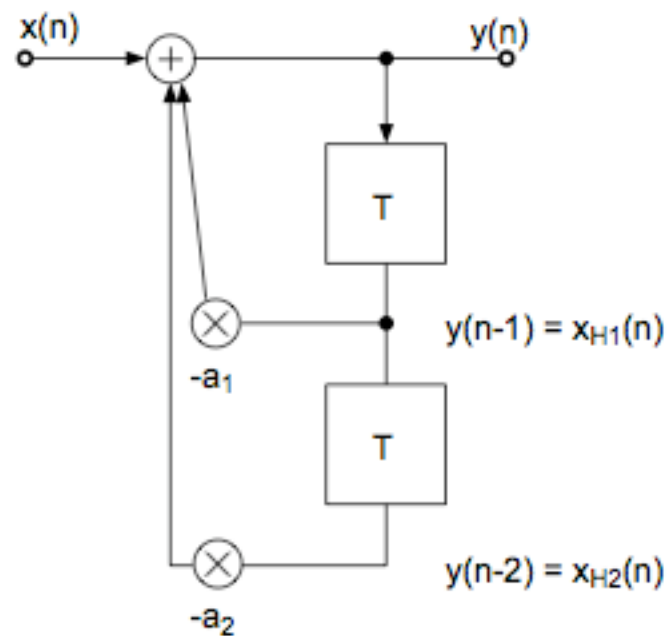
- And the transfer function:

$$H(z) = \sum_{k=0}^{N-1} b_k z^{-k}$$



IIR filters

- On the other hand, an IIR filter is a system including feed-back delay lines and thus has an infinite impulse response $h(n)$:



- Such that its transfer function is:

$$H(z) = \frac{1}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

Simple IIR filter implementations

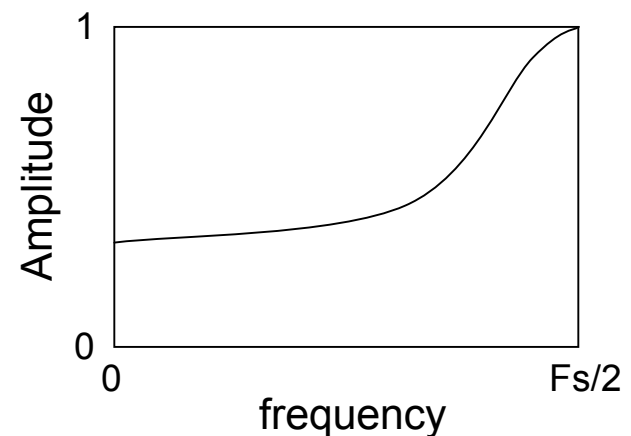
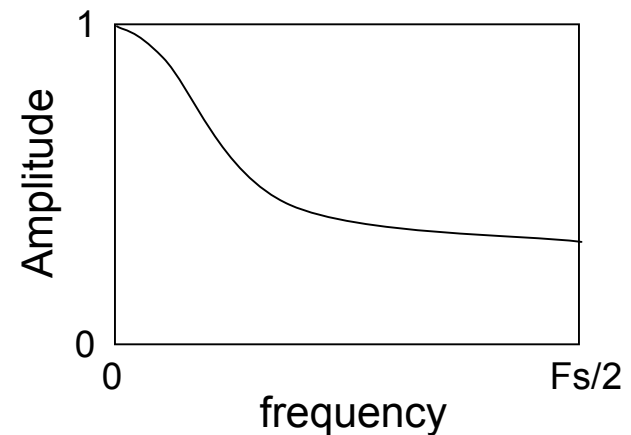
- Simple IIR filters can behave like infinitely long FIR filters
- Recursive low-pass filter (Exponential time average):

$$y(n) = b_0x(n) + a_1y(n - 1)$$

- Increasing b reduces the cutoff frequency
- However $b < 1$, otherwise is unstable
- Recursive high-pass filter:

$$y(n) = b_0x(n) - a_1y(n - 1)$$

- Increasing b increases the cutoff frequency

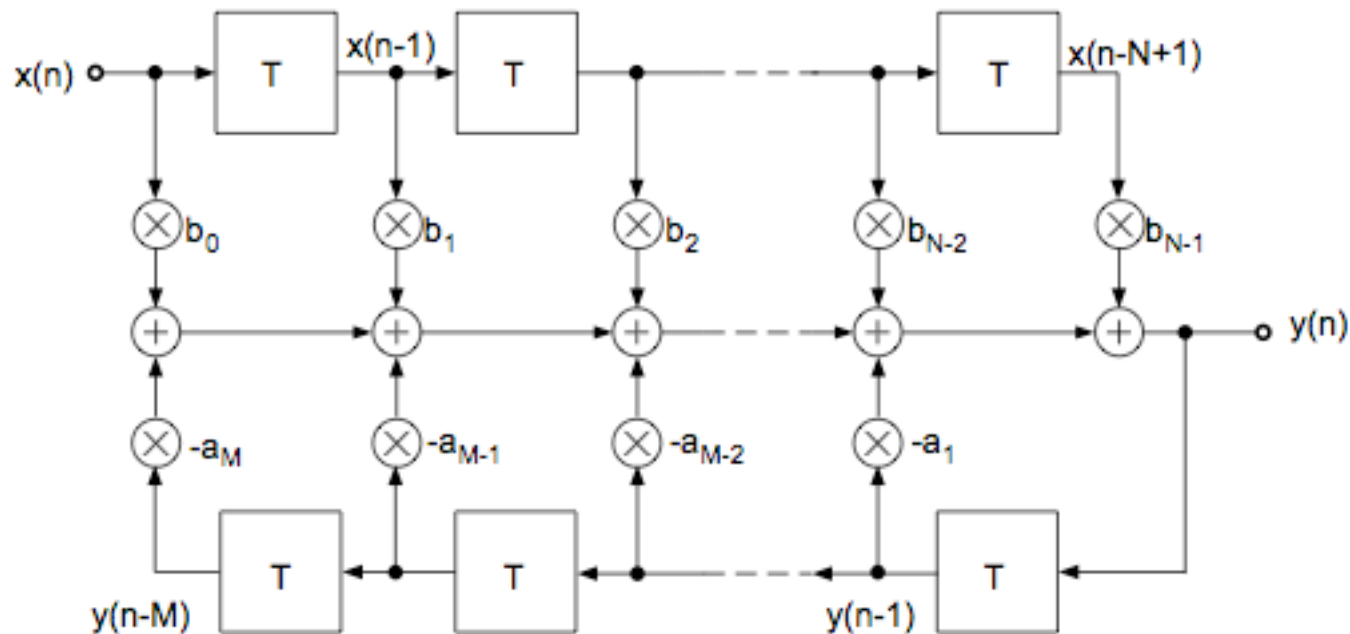


IIR filters

- In general, an IIR filter can include both feed-forward and feedback delay lines, such that:

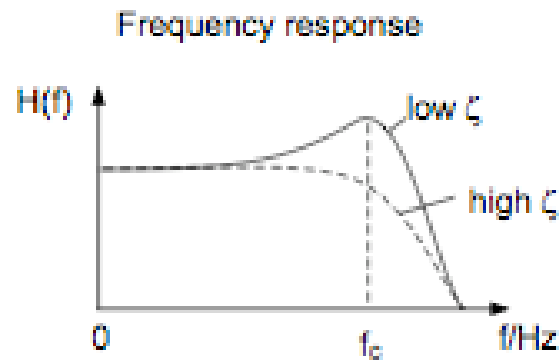
$$y(n) = \sum_{k=0}^{N-1} b_k x(n-k) - \sum_{k=1}^M a_k y(n-k)$$

$$H(z) = \frac{\sum_{k=0}^{N-1} b_k z^{-k}}{1 + \sum_{k=1}^M a_k z^{-k}}$$



Canonical 2nd order structure

- It is common practice to design high-order common filters out of series of low-order filters (e.g. 2nd order canonical filter)
- E.g. we can design a LPF using the bilinear transform method, from known cut-off (f_c) and sampling (f_s) frequencies and damping factor (ζ)



$$C = 1/[\tan(\pi f_c / f_s)]$$

$$b_0 = 1/(1 + 2\zeta C + C^2)$$

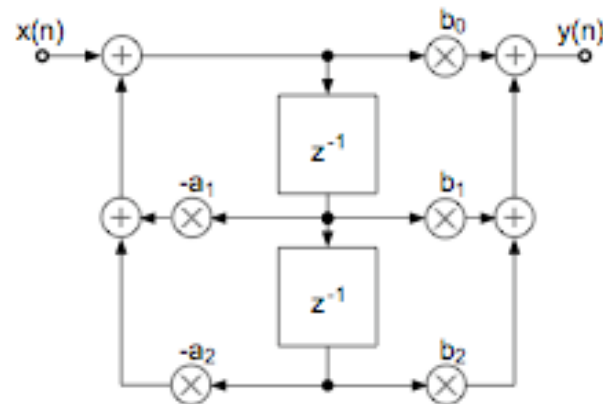
$$b_1 = 2b_0$$

$$b_2 = b_0$$

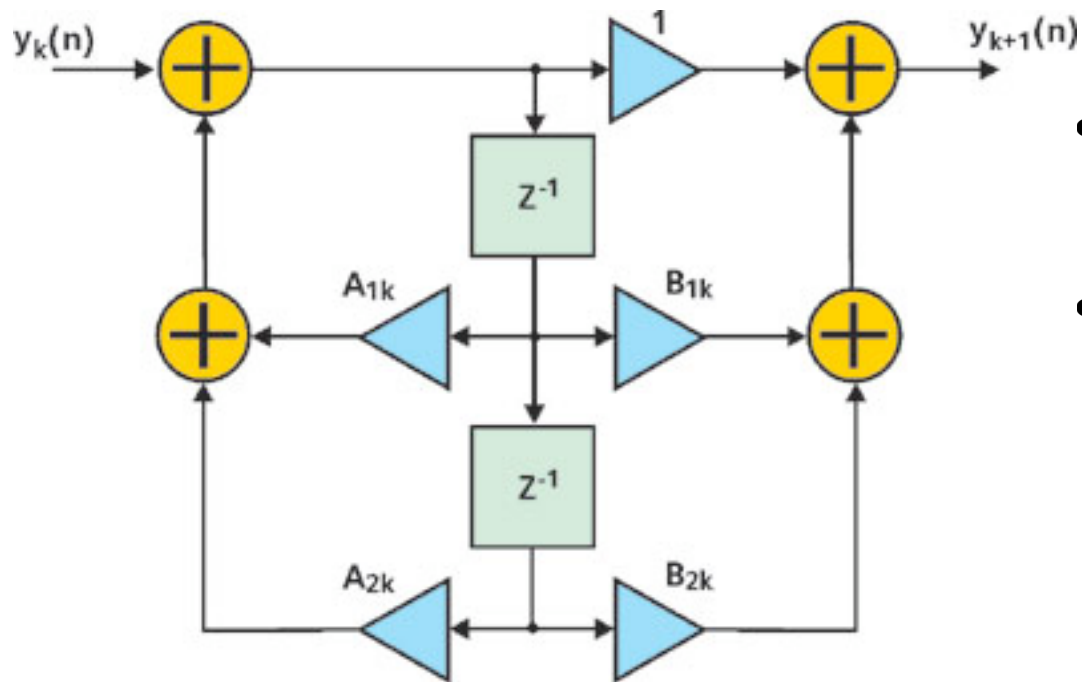
$$a_1 = 2b_0(1 - C^2)$$

$$a_2 = b_0(1 + 2\zeta C + C^2)$$

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

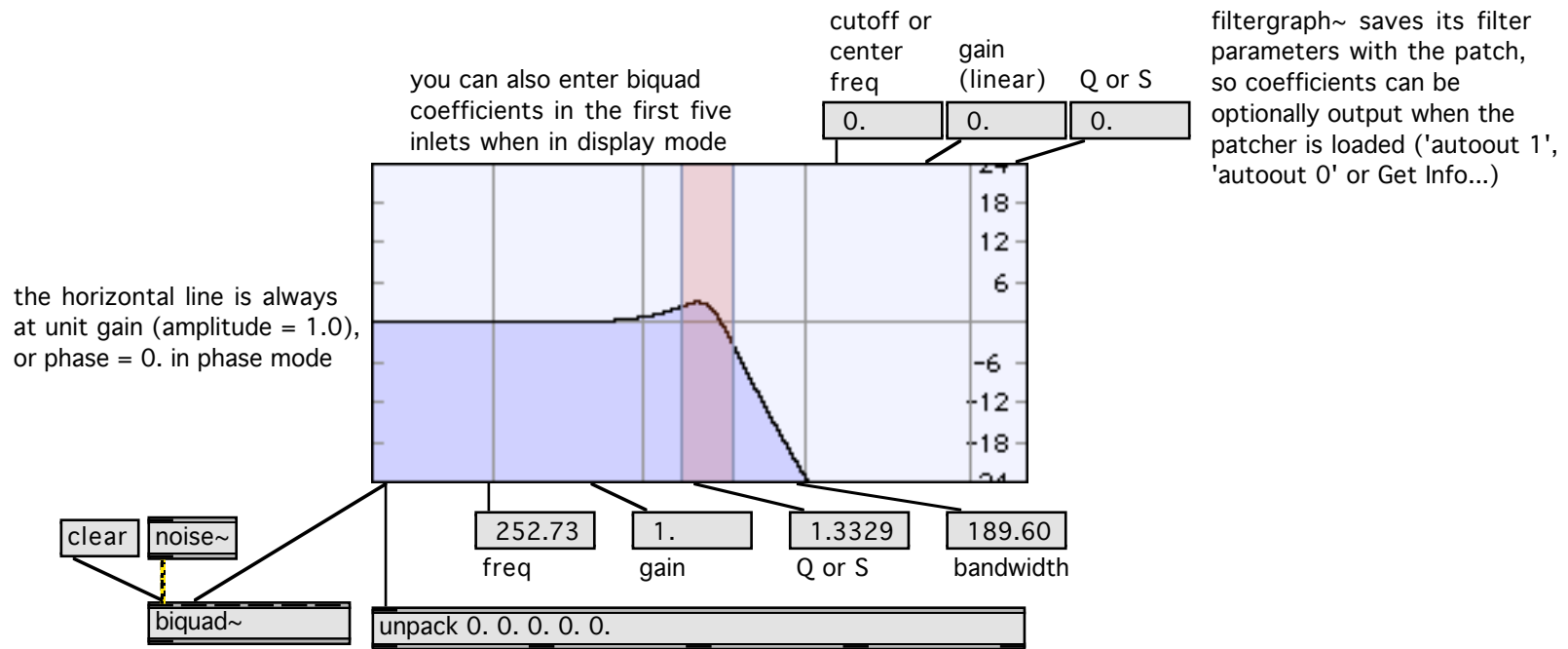


MSP: Biquad~ Filters



- Biquad~ implements a two-pole, two-zero filter.
- There are six inlets, one for the signal to be filtered and five for the coefficients (amplitudes) shown in the signal diagram.

MSP: Filtergraph~



- The filtergraph~ object provides a graphic interface to generate coefficients for biquad~.

FIR and IIR compared

- FIR:
- They are stable, as they have no feedback.
- To achieve a steep cutoff they require more arithmetic operations and memory than equivalent IIR (more costly).
- Easy to design with linear phase response.

- IIR
- Can produce exponentially sharp cutoffs and boosts using less computations than FIR (thanks to the feedback loop).
- Because of their inherent recursion, they can be unstable and are prone to roundoff errors.
- Suffer from phase distortion (due to non-linear phase response) and ringing (unwanted oscillations triggered by transients. As a result transients are smeared over time).

Useful References

- Zölzer, U. (Ed). “DAFX: Digital Audio Effects”. John Wiley and Sons (2002)
 - Chapter 1: Zölzer, U. “Introduction”.
 - Chapter 2: Dutilleux, P. and Zölzer, U. “Filters”
- Smith, J.O. “Introduction to Digital Filters”, September 2005 Draft,
<http://ccrma.stanford.edu/~jos/filters05/>
- Roads, C. “The Computer Music Tutorial”. MIT Press (1996)