DSP & Digital Filters

Mike Brookes

▷ 1: Introduction
Organization
Signals
Processing
Syllabus
Sequences
Time Scaling
z-Transform
Region of Convergence
z-Transform examples
Rational z-Transforms
Rational example
Inverse z-Transform
MATLAB routines
Summary

1: Introduction

Organization

1: Introduction
▷ Organization
Signals
Processing
Syllabus
Sequences
Time Scaling
z-Transform
Region of Convergence
z-Transform examples
Rational z-Transforms
Rational example
Inverse z-Transform
MATLAB routines
Summary

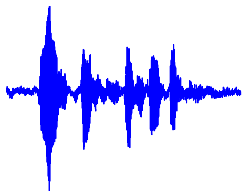
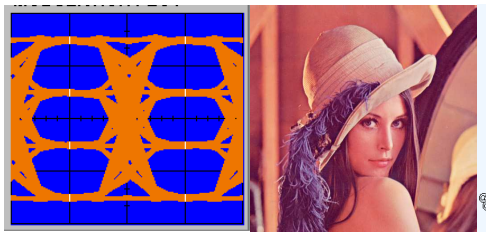
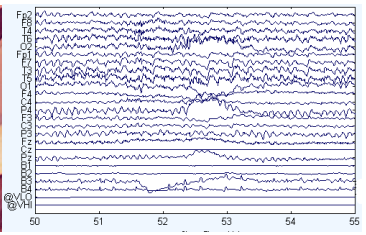
- 18 lectures: feel free to ask questions
- Textbooks:
  - (a) Mitra "Digital Signal Processing" ISBN:0071289461 £41 covers most of the course except for some of the multirate stuff
  - (b) Harris "Multirate Signal Processing" ISBN:0137009054 £49 covers multirate material in more detail but less rigour than Mitra
- Lecture slides available via Blackboard or on my website: http://www.ee.ic.ac.uk/hp/staff/dmb/courses/dspdf/dspdf.htm
  - quite dense - ensure you understand each line
  - email me if you don't understand or don't agree with anything
- Prerequisites: 3rd year DSP - attend lectures if dubious
- Exam + Formula Sheet (past exam papers + solutions on website)
- Problems: Mitra textbook contains many problems at the end of each chapter and also MATLAB exercises

Signals

1: Introduction
Organization
▷ Signals
Processing
Syllabus
Sequences
Time Scaling
z-Transform
Region of Convergence
z-Transform examples
Rational z-Transforms
Rational example
Inverse z-Transform
MATLAB routines
Summary

- A signal is a numerical quantity that is a function of one or more independent variables such as time or position.
- Real-world signals are analog and vary continuously and take continuous values.
- Digital signals are sampled at discrete times and are quantized to a finite number of discrete values
- We will mostly consider one-dimensionsal real-valued signals with regular sample instants; except in a few places, we will ignore the quantization.
  - Extension to multiple dimensions and complex-valued signals is straighforward in many cases.

Examples:

Processing

1: Introduction
Organization
Signals
▷ Processing
Syllabus
Sequences
Time Scaling
z-Transform
Region of Convergence
z-Transform examples
Rational z-Transforms
Rational example
Inverse z-Transform
MATLAB routines
Summary

- Aims to "improve" a signal in some way or extract some information from it
- Examples:
  - Modulation/demodulation
  - Coding and decoding
  - Interference rejection and noise suppression
  - Signal detection, feature extraction
- We are concerned with linear, time-invariant processing

Syllabus

1: Introduction
Organization
Signals
Processing
▷ Syllabus
Sequences
Time Scaling
z-Transform
Region of Convergence
z-Transform examples
Rational z-Transforms
Rational example
Inverse z-Transform
MATLAB routines
Summary

Main topics:

- Introduction/Revision
- Transforms
- Discrete Time Systems
- Filter Design
  - FIR Filter Design
  - IIR Filter Design
- Multirate systems
  - Multirate Fundamentals
  - Multirate Filters
  - Subband processing

Sequences

1: Introduction
Organization
Signals
Processing
Syllabus
▷ Sequences
Time Scaling
z-Transform
Region of Convergence
z-Transform examples
Rational z-Transforms
Rational example
Inverse z-Transform
MATLAB routines
Summary

We denote the n^{th} sample of a signal as $x[n]$ where $-\infty < n < +\infty$ and the entire sequence as $\{x[n]\}$ although we will often omit the braces.

Special sequences:

- Unit step: $u[n] = \begin{cases} 1 & n \geq 0 \\ 0 & \text{otherwise} \end{cases}$
- Unit impulse: $\delta[n] = \begin{cases} 1 & n = 0 \\ 0 & \text{otherwise} \end{cases}$
- Condition: $\delta_{\text{condition}}[n] = \begin{cases} 1 & \text{condition is true} \\ 0 & \text{otherwise} \end{cases}$ (e.g. $u[n] = \delta_{n \geq 0}$)
- Right-sided: $x[n] = 0$ for $n < N_{min}$
- Left-sided: $x[n] = 0$ for $n > N_{max}$
- Finite length: $x[n] = 0$ for $n \notin [N_{min}, N_{max}]$
- Causal: $x[n] = 0$ for $n < 0$, Anticausal: $x[n] = 0$ for $n > 0$
- Finite Energy: $\sum_{n=-\infty}^{\infty} |x[n]|^2 < \infty$ (e.g. $x[n] = n^{-1}u[n-1]$)
- Absolutely Summable: $\sum_{n=-\infty}^{\infty} |x[n]| < \infty \Rightarrow$ Finite energy

Time Scaling

1: Introduction
Organization
Signals
Processing
Syllabus
Sequences
▷ Time Scaling
z-Transform
Region of Convergence
z-Transform examples
Rational z-Transforms
Rational example
Inverse z-Transform
MATLAB routines
Summary

For sampled signals, the n^{th} sample is at time $t = nT = \frac{n}{f_s}$ where $f_s = \frac{1}{T}$ is the sample frequency.

We usually scale time so that $f_s = 1$: divide all "real" frequencies and angular frequencies by f_s and divide all "real" times by T.

- To scale back to real-world values: multiply all *times* by T and all *frequencies* and *angular frequencies* by $T^{-1} = f_s$.
- We use Ω for "real" angular frequencies and ω for normalized angular frequency. The units of ω are "radians per sample".

Energy of sampled signal, $x[n]$, equals $\sum x^2[n]$

- Multiply by T to get energy of continuous signal, $\int x^2(t)dt$, provided there is no aliasing.

Power of $\{x[n]\}$ is the average of $x^2[n]$ in "energy per sample"

- same value as the power of $x(t)$ in "energy per second" provided there is no aliasing.

Warning: Several MATLAB routines scale time so that $f_s = 2$ Hz. Weird, non-standard and irritating.

z-Transform

1: Introduction
Organization
Signals
Processing
Syllabus
Sequences
Time Scaling
▷ z-Transform
Region of Convergence
z-Transform examples
Rational z-Transforms
Rational example
Inverse z-Transform
MATLAB routines
Summary

The z-transform converts a sequence, $\{x[n]\}$, into a function, $X(z)$, of an arbitrary complex-valued variable z.

Why do it?

- Complex functions are easier to manipulate than sequences
- Useful operations on sequences correspond to simple operations on the z-transform:
  - addition, multiplication, scalar multiplication, time-shift, convolution
- Definition: $X(z) = \sum_{n=-\infty}^{+\infty} x[n]z^{-n}$

Region of Convergence

1: Introduction
Organization
Signals
Processing
Syllabus
Sequences
Time Scaling
z-Transform
▷ Region of Convergence
z-Transform examples
Rational z-Transforms
Rational example
Inverse z-Transform
MATLAB routines
Summary

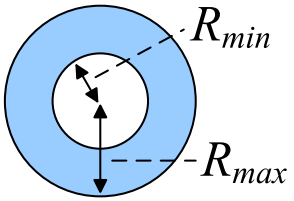
The set of z for which $X(z)$ converges is its *Region of Convergence* (ROC).

Complex analysis $\Rightarrow$: the ROC of a power series (if it exists at all) is always an annular region of the form $0 \leq R_{min} < |z| < R_{max} \leq \infty$.

$X(z)$ will always converge absolutely inside the ROC and may converge on some, all, or none of the boundary.

- "converge absolutely" $\Leftrightarrow \sum_{n=-\infty}^{+\infty} |x[n]z^{-n}| < \infty$

- finite length $\Leftrightarrow R_{min} = 0,\ R_{max} = \infty$
  - ROC may included either, both or none of 0 and ∞
- absolutely summable $\Leftrightarrow X(z)$ converges for $|z| = 1$.
- right-sided & $|x[n]| < A \times B^n \ \Rightarrow R_{max} = \infty$
  - + causal $\Rightarrow X(\infty)$ converges
- left-sided & $|x[n]| < A \times B^{-n} \ \Rightarrow R_{min} = 0$
  - + anticausal $\Rightarrow X(0)$ converges



[Convergence Properties]

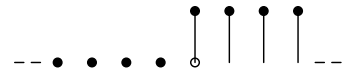
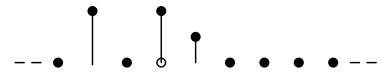
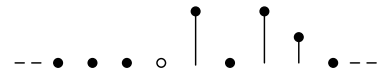
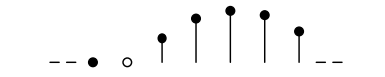
Null Region of Convergence:
It is possible to define a sequence, $x[n]$, whose z-transform never converges (i.e. the ROC is null). An example is $x[n] \equiv 1$. The z-transfom is $X(z) = \sum z^{-n}$ and it is clear that this fails to converge for any real value of z.

Convergence for $x[n]$ causal:
If $x[n]$ is causal with $|x[n]| < A \times B^n$ for some A and B, then $|X(z)| = \left|\sum_{n=0}^{\infty} x[n]z^{-n}\right| \leq \sum_{n=0}^{\infty} \left|x[n]z^{-n}\right|$ and so, for $|z| = R \geq B$, $|X(z)| \leq \sum_{n=0}^{\infty} AB^n R^{-n} = \frac{A}{1-BR^{-1}} < \infty$.

Convergence for $x[n]$ right-sided:
If $x[n]$ is right-sided with $|x[n]| < A \times B^n$ for some A and B and $x[n] = 0$ for $n < N$, then $y[n] = x[n-N]$ is causal with $|y[n]| < A \times B^{n+N} = AB^N \times B^n$. Hence, from the previous result, we known that $Y(z)$ converges for $|z| \geq B$. The z-transform, $X(z)$, is given by $X(z) = z^N Y(z)$ so $X(z)$ will converge for any $B \leq |z| < \infty$ since $\left|z^N\right| < \infty$ for $|z|$ in this range.

z-Transform examples

The sample at $n = 0$ is indicated by an open circle.

$u[n]$		$\frac{1}{1-z^{-1}}$	$1 < \|z\| \leq \infty$
$x[n]$		$2z^2 + 2 + z^{-1}$	$0 < \|z\| < \infty$
$x[n-3]$		$z^{-3}\left(2z^2 + 2 + z^{-1}\right)$	$0 < \|z\| \leq \infty$
$\alpha^n u[n]_{\alpha=0.8}$		$\frac{1}{1-\alpha z^{-1}}$	$\alpha < \|z\| \leq \infty$
$-\alpha^n u[-n-1]$		$\frac{1}{1-\alpha z^{-1}}$	$0 \leq \|z\| < \alpha$
$nu[n]$		$\frac{z^{-1}}{1-2z^{-1}+z^{-2}}$	$1 < \|z\| \leq \infty$
$\sin(\omega n)u[n]_{\omega=0.5}$		$\frac{z^{-1}\sin(\omega)}{1-2z^{-1}\cos(\omega)+z^{-2}}$	$1 < \|z\| \leq \infty$
$\cos(\omega n)u[n]_{\omega=0.5}$		$\frac{1-z^{-1}\cos(\omega)}{1-2z^{-1}\cos(\omega)+z^{-2}}$	$1 < \|z\| \leq \infty$

Note: Examples 4 and 5 have the same z-transform but different ROCs.

Geometric Progression: $\sum_{n=q}^{r} \alpha^n z^{-n} = \frac{\alpha^q z^{-q} - \alpha^{r+1} z^{-r-1}}{1-\alpha z^{-1}}$

Rational z-Transforms

1: Introduction
Organization
Signals
Processing
Syllabus
Sequences
Time Scaling
z-Transform
Region of Convergence
z-Transform examples
Rational z-Transforms
Rational example
Inverse z-Transform
MATLAB routines
Summary

Most z-transforms that we will meet are rational polynomials with real coefficients, usually one polynomial in z^{-1} divided by another.

$$G(z) = g\frac{\prod_{m=1}^{M}\left(1-z_m z^{-1}\right)}{\prod_{k=1}^{K}(1-p_k z^{-1})} = gz^{K-M}\frac{\prod_{m=1}^{M}(z-z_m)}{\prod_{k-1}^{K}(z-p_k)}$$

Completely defined by the poles, zeros and gain.

The absolute values of the poles define the ROCs:

$\exists R+1$ different ROCs

where R is the number of distinct pole magnitudes.

Note: There are $K-M$ zeros or $M-K$ poles at $z=0$ (easy to overlook)

Rational example

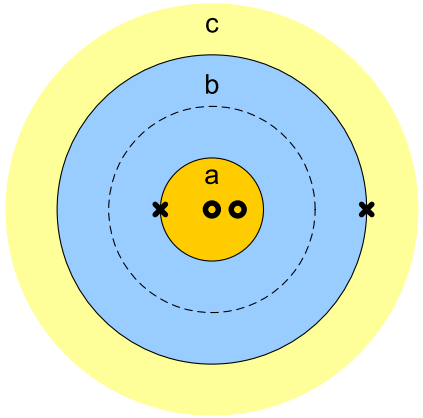
1: Introduction
Organization
Signals
Processing
Syllabus
Sequences
Time Scaling
z-Transform
Region of Convergence
z-Transform examples
Rational z-Transforms
▷ Rational example
Inverse z-Transform
MATLAB routines
Summary

$$G(z) = \frac{8-2z^{-1}}{4-4z^{-1}-3z^{-2}}$$

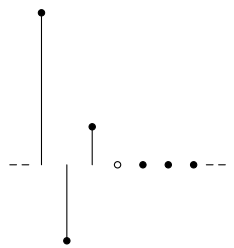
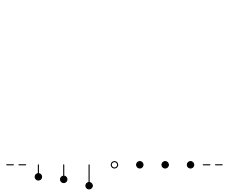
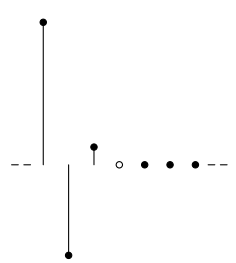
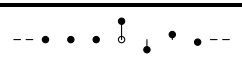
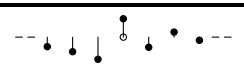
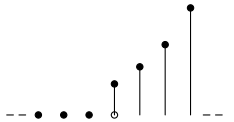
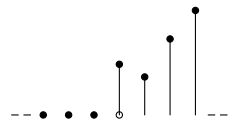
Poles/Zeros: $G(z) = \frac{2z(z-0.25))}{(z+0.5)(z-1.5)}$

$\Rightarrow$ Poles at $z = \{-0.5, +1.5)\}$,

Zeros at $z = \{0, +0.25\}$



Partial Fractions: $G(z) = \frac{0.75}{1+0.5z^{-1}} + \frac{1.25}{1-1.5z^{-1}}$

ROC	ROC	$\frac{0.75}{1+0.5z^{-1}}$	$\frac{1.25}{1-1.5z^{-1}}$	$G(z)$
a	$0 \leq \lvert z \rvert < 0.5$			
b	$0.5 < \lvert z \rvert < 1.5$			
c	$1.5 < \lvert z \rvert \leq \infty$			

Inverse z-Transform

1: Introduction
Organization
Signals
Processing
Syllabus
Sequences
Time Scaling
z-Transform
Region of Convergence
z-Transform examples
Rational z-Transforms
Rational example
▷ Inverse z-Transform
MATLAB routines
Summary

$g[n] = \frac{1}{2\pi j} \oint G(z) z^{n-1} dz$ where the integral is anti-clockwise around a circle within the ROC, $z = Re^{j\theta}$.

Proof:

$$\frac{1}{2\pi j} \oint G(z) z^{n-1} dz = \frac{1}{2\pi j} \oint \left(\sum_{m=-\infty}^{\infty} g[m] z^{-m} \right) z^{n-1} dz$$

$$\stackrel{\text{(i)}}{=} \sum_{m=-\infty}^{\infty} g[m] \frac{1}{2\pi j} \oint z^{n-m-1} dz$$

$$\stackrel{\text{(ii)}}{=} \sum_{m=-\infty}^{\infty} g[m] \delta[n-m] = g[n]$$

(i) depends on the circle with radius R lying within the ROC

(ii) Cauchy's theorem: $\frac{1}{2\pi j} \oint z^{k-1} dz = \delta[k]$ for $z = Re^{j\theta}$ anti-clockwise.

$$\frac{dz}{d\theta} = jRe^{j\theta} \Rightarrow \frac{1}{2\pi j} \oint z^{k-1} dz = \frac{1}{2\pi j} \int_{\theta=0}^{2\pi} R^{k-1} e^{j(k-1)\theta} \times jRe^{j\theta} d\theta$$

$$= \frac{R^k}{2\pi} \int_{\theta=0}^{2\pi} e^{jk\theta} d\theta$$

$$= R^k \delta(k) = \delta(k) \qquad [R^0 = 1]$$

In practice use a combination of partial fractions and table of z-transforms.

MATLAB routines

1: Introduction
Organization
Signals
Processing
Syllabus
Sequences
Time Scaling
z-Transform
Region of Convergence
z-Transform examples
Rational z-Transforms
Rational example
Inverse z-Transform
▷ MATLAB routines
Summary

tf2zp,zp2tf	$\frac{b(z^{-1})}{a(z^{-1})} \leftrightarrow \{z_m, p_k, g\}$
residuez	$\frac{b(z^{-1})}{a(z^{-1})} \rightarrow \sum_k \frac{r_k}{1-p_k z^{-1}}$
tf2sos,sos2tf	$\frac{b(z^{-1})}{a(z^{-1})} \leftrightarrow \prod_l \frac{b_{0,l}+b_{1,l}z^{-1}+b_{2,l}z^{-2}}{1+a_{1,l}z^{-1}+a_{2,l}z^{-2}}$
zp2sos,sos2zp	$\{z_m, p_k, g\} \leftrightarrow \prod_l \frac{b_{0,l}+b_{1,l}z^{-1}+b_{2,l}z^{-2}}{1+a_{\in 1,l}z^{-1}+a_{2,l}z^{-2}}$
zp2ss,ss2zp	$\{z_m, p_k, g\} \leftrightarrow \begin{cases} x' = Ax + Bu \\ y = Cx + Du \end{cases}$
tf2ss,ss2tf	$\frac{b(z^{-1})}{a(z^{-1})} \leftrightarrow \begin{cases} x' = Ax + Bu \\ y = Cx + Du \end{cases}$

Summary

1: Introduction
Organization
Signals
Processing
Syllabus
Sequences
Time Scaling
z-Transform
Region of Convergence
z-Transform examples
Rational z-Transforms
Rational example
Inverse z-Transform
MATLAB routines
▷ Summary

- Time scaling: assume $f_s = 1$ so $-\pi < \omega \leq \pi$
- z-transform: $X(z) = \sum_{n=-\infty}^{+\infty} x[n]^{-n}$
- ROC: $0 \leq R_{min} < |z| < R_{max} \leq \infty$
  - Causal: $\infty \in$ ROC
  - Absolutely summable: $|z| = 1 \in$ ROC
- Inverse z-transform: $g[n] = \frac{1}{2\pi j} \oint G(z) z^{n-1} dz$
  - Not unique unless ROC is specified
  - Use partial fractions and/or a table

For further details see Mitra:1 & 6.

2: Three Different Fourier Transforms
Fourier Transforms
Convergence of DTFT
DTFT Properties
DFT Properties
Symmetries
Parseval's Theorem
Convolution
Sampling Process
Zero-Padding
Phase Unwrapping
Uncertainty principle
Summary
MATLAB routines

2: Three Different Fourier Transforms

Fourier Transforms

2: Three Different Fourier Transforms
▷ Fourier Transforms
Convergence of DTFT
DTFT Properties
DFT Properties
Symmetries
Parseval's Theorem
Convolution
Sampling Process
Zero-Padding
Phase Unwrapping
Uncertainty principle
Summary
MATLAB routines

Three different Fourier Transforms:

- CTFT (Continuous-Time Fourier Transform): $x(t) \to X(j\Omega)$
- DTFT (Discrete-Time Fourier Transform): $x[n] \to X(e^{j\omega})$
- DFT a.k.a. FFT (Discrete Fourier Transform): $x[n] \to X[k]$

	Forward Transform	Inverse Transform
CTFT	$X(j\Omega) = \int_{-\infty}^{\infty} x(t)e^{-j\Omega t}dt$	$x(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} X(j\Omega)e^{j\Omega t}d\Omega$
DTFT	$X(e^{j\omega}) = \sum_{-\infty}^{\infty} x[n]e^{-j\omega n}$	$x[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} X(e^{j\omega})e^{j\omega n}d\omega$
DFT	$X[k] = \sum_{0}^{N-1} x[n]e^{-j2\pi\frac{kn}{N}}$	$x[n] = \frac{1}{N}\sum_{0}^{N-1} X[k]e^{j2\pi\frac{kn}{N}}$

We use Ω for "real" and $\omega = \Omega T$ for "normalized" angular frequency.
Nyquist frequency is at $\Omega_{\text{Nyq}} = 2\pi\frac{f_s}{2} = \frac{\pi}{T}$ and $\omega_{\text{Nyq}} = \pi$.

For "power signals" (energy $\propto$ duration), CTFT & DTFT are unbounded.
Fix this by normalizing:

$$X(j\Omega) = \lim_{A\to\infty} \frac{1}{2A}\int_{-A}^{A} x(t)e^{-j\Omega t}dt$$
$$X(e^{j\omega}) = \lim_{A\to\infty} \frac{1}{2A+1}\sum_{-A}^{A} x[n]e^{-j\omega n}$$

Convergence of DTFT

2: Three Different Fourier Transforms
Fourier Transforms
▷ Convergence of DTFT
DTFT Properties
DFT Properties
Symmetries
Parseval's Theorem
Convolution
Sampling Process
Zero-Padding
Phase Unwrapping
Uncertainty principle
Summary
MATLAB routines

DTFT: $X(e^{j\omega}) = \sum_{-\infty}^{\infty} x[n]e^{-j\omega n}$ does not converge for all $x[n]$.

Consider the finite sum: $X_K(e^{j\omega}) = \sum_{-K}^{K} x[n]e^{-j\omega n}$

Strong Convergence:

$x[n]$ absolutely summable $\Rightarrow$ $X(e^{j\omega})$ converges uniformly

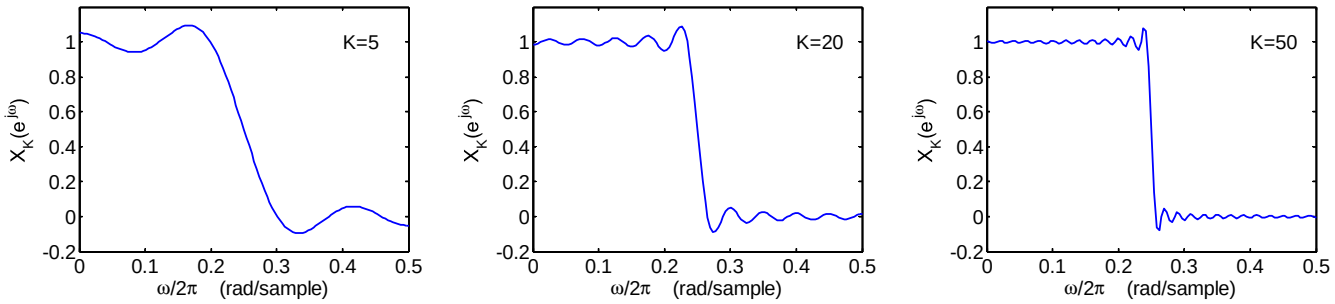
$\sum_{-\infty}^{\infty} |x[n]| < \infty \Rightarrow \sup_{\omega} \left|X(e^{j\omega}) - X_K(e^{j\omega})\right| \xrightarrow[K\to\infty]{} 0$

Weaker convergence:

$x[n]$ finite energy $\Rightarrow$ $X(e^{j\omega})$ converges in mean square

$\sum_{-\infty}^{\infty} |x[n]|^2 < \infty \Rightarrow \frac{1}{2\pi}\int_{-\pi}^{\pi} \left|X(e^{j\omega}) - X_K(e^{j\omega})\right|^2 d\omega \xrightarrow[K\to\infty]{} 0$

Example: $x[n] = \frac{\sin 0.5\pi n}{\pi n}$



Gibbs phenomenon:

Converges at each ω as $K \to \infty$ but peak error does not get smaller.

[DTFT Convergence Proofs]

(1) Strong Convergence:

[these proofs are not examinable]

We are given that $\sum_{-\infty}^{\infty} |x[n]| < \infty \Rightarrow \forall \epsilon > 0, \exists\, N \text{ such that } \sum_{|n|>N} |x[n]| < \epsilon$

For $K \geq N$, $\sup_\omega \left|X(e^{j\omega}) - X_K(e^{j\omega})\right| = \sup_\omega \left|\sum_{|n|>K} x[n]e^{-j\omega n}\right|$

$$\leq \sup_\omega \left(\sum_{|n|>K} \left|x[n]e^{-j\omega n}\right|\right) = \sum_{|n|>K} |x[n]| < \epsilon$$

(2) Weak Convergence:

We are given that $\sum_{-\infty}^{\infty} |x[n]|^2 < \infty \Rightarrow \forall \epsilon > 0, \exists\, N \text{ such that } \sum_{|n|>N} |x[n]|^2 < \epsilon$

Define $y^{[K]}[n] = \begin{cases} 0 & |n| \leq K \\ x[n] & |n| > K \end{cases}$ so that its DTFT is, $Y^{[K]}(e^{j\omega}) = \sum_{-\infty}^{\infty} y^{[K]}[n]e^{-j\omega n}$

We see that $X(e^{j\omega}) - X_K(e^{j\omega}) = \sum_{-\infty}^{\infty} x[n]e^{-j\omega n} - \sum_{-K}^{K} x[n]e^{-j\omega n}$

$$= \sum_{|n|>K} x[n]e^{-j\omega n} = \sum_{-\infty}^{\infty} y^{[K]}[n]e^{-j\omega n} = Y^{[K]}(e^{j\omega})$$

From Parseval's theorem, $\sum_{-\infty}^{\infty} \left|y^{[K]}[n]\right|^2 = \frac{1}{2\pi}\int_{-\pi}^{\pi} \left|Y^{[K]}(e^{j\omega})\right|^2 d\omega$

$$= \frac{1}{2\pi}\int_{-\pi}^{\pi} \left|X(e^{j\omega}) - X_K(e^{j\omega})\right|^2 d\omega$$

Hence for $K \geq N$, $\frac{1}{2\pi}\int_{-\pi}^{\pi} \left|X(e^{j\omega}) - X_K(e^{j\omega})\right|^2 d\omega = \sum_{-\infty}^{\infty} \left|y^{[K]}[n]\right|^2 = \sum_{|n|>N} |x[n]|^2 < \epsilon$

DTFT Properties

2: Three Different Fourier Transforms
Fourier Transforms
Convergence of DTFT
▷ DTFT Properties
DFT Properties
Symmetries
Parseval's Theorem
Convolution
Sampling Process
Zero-Padding
Phase Unwrapping
Uncertainty principle
Summary
MATLAB routines

DTFT: $X(e^{j\omega}) = \sum_{-\infty}^{\infty} x[n]e^{-j\omega n}$

- DTFT is periodic in ω: $X(e^{j(\omega+2m\pi)}) = X(e^{j\omega})$ for integer m.
- DTFT is the z-Transform evaluated at the point $e^{j\omega}$:
  $X(z) = \sum_{-\infty}^{\infty} x[n]z^{-n}$
  DTFT converges iff the ROC includes $|z| = 1$.
- DTFT is the same as the CTFT of a signal comprising impulses at the sample times (Dirac δ functions) of appropriate heights:
  $x_\delta(t) = \sum x[n]\delta(t - nT) = x(t) \times \sum_{-\infty}^{\infty} \delta(t - nT)$
  Equivalent to multiplying a continuous $x(t)$ by an impulse train.

Proof: $X(e^{j\omega}) = \sum_{-\infty}^{\infty} x[n]e^{-j\omega n}$

$$\sum_{n=-\infty}^{\infty} x[n] \int_{-\infty}^{\infty} \delta(t - nT)e^{-j\omega\frac{t}{T}}dt$$

$$\overset{\text{(i)}}{=} \int_{-\infty}^{\infty} \sum_{n=-\infty}^{\infty} x[n]\delta(t - nT)e^{-j\omega\frac{t}{T}}dt$$

$$\overset{\text{(ii)}}{=} \int_{-\infty}^{\infty} x_\delta(t)e^{-j\Omega t}dt$$

(i) OK if $\sum_{-\infty}^{\infty} |x[n]| < \infty$. (ii) use $\omega = \Omega T$.

DFT Properties

2: Three Different Fourier Transforms
Fourier Transforms
Convergence of DTFT
DTFT Properties
▷ DFT Properties
Symmetries
Parseval's Theorem
Convolution
Sampling Process
Zero-Padding
Phase Unwrapping
Uncertainty principle
Summary
MATLAB routines

DFT: $X[k] = \sum_0^{N-1} x[n] e^{-j2\pi \frac{kn}{N}}$

DTFT: $X(e^{j\omega}) = \sum_{-\infty}^{\infty} x[n] e^{-j\omega n}$

Case 1: $x[n] = 0$ for $n \notin [0,\ N-1]$

DFT is the same as DTFT at $\omega_k = \frac{2\pi}{N} k$.

The $\{\omega_k\}$ are uniformly spaced from $\omega = 0$ to $\omega = 2\pi \frac{N-1}{N}$.
DFT is the z-Transform evaluated at N equally spaced points around the unit circle beginning at $z = 1$.

Case 2: $x[n]$ is periodic with period N

DFT equals the normalized DTFT

$$X[k] = \lim_{K\to\infty} \frac{N}{2K+1} \times X_K(e^{j\omega_k})$$

where $X_K(e^{j\omega}) = \sum_{-K}^{K} x[n] e^{-j\omega n}$

[Proof of Case 2]

We want to show that if $x[n] = x[n+N]$ (i.e. $x[n]$ is periodic with period N) then

$$\lim_{K\to\infty} \frac{N}{2K+1} \times X_K(e^{j\omega_k}) \triangleq \lim_{K\to\infty} \frac{N}{2K+1} \times \sum_{-K}^{K} x[n]e^{-j\omega_k n} = X[k]$$

where $\omega_k = \frac{2\pi}{N}k$. We assume that $x[n]$ is bounded with $|x[n]| < B$.

We first note that the summand is periodic:

$$x[n+N]e^{-j\omega_k(n+N)} = x[n]e^{-j\omega_k n}e^{-jk\frac{2\pi}{N}N} = x[n]e^{-j\omega_k n}e^{-j2\pi k} = x[n]e^{-j\omega_k n}.$$

We now define M and R so that $2K+1 = MN+R$ where $0 \le R < N$ (i.e. MN is the largest multiple of N that is $\le 2K+1$). We can now write

$$\frac{N}{2K+1} \times \sum_{-K}^{K} x[n]e^{-j\omega_k n} = \frac{N}{MN+R} \times \sum_{-K}^{K-R} x[n]e^{-j\omega_k n} + \frac{N}{MN+R} \times \sum_{K-R+1}^{K} x[n]e^{-j\omega n}$$

The first sum contains MN consecutive terms of a periodic summand and so equals M times the sum over one period. The second sum contains R bounded terms and so its magnitude is $< RB < NB$.

So $\frac{N}{2K+1} \times \sum_{-K}^{K} x[n]e^{-j\omega_k n} = \frac{MN}{MN+R} \times \sum_{0}^{N-1} x[n]e^{-j\omega_k n} + P = \frac{1}{1+\frac{R}{MN}} \times X[k] + P$

where $|P| < \frac{N}{MN+R} \times NB \le \frac{N}{MN+0} \times NB = \frac{NB}{M}$.

As $M \to \infty$, $|P| \to 0$ and $\frac{1}{1+\frac{R}{MN}} \to 1$ so the whole expression tends to $X[k]$.

Symmetries

2: Three Different Fourier Transforms
Fourier Transforms
Convergence of DTFT
DTFT Properties
DFT Properties
▷ Symmetries
Parseval's Theorem
Convolution
Sampling Process
Zero-Padding
Phase Unwrapping
Uncertainty principle
Summary
MATLAB routines

If $x[n]$ has a special property then $X(e^{j\omega})$and $X[k]$ will have corresponding properties as shown in the table (and vice versa):

One domain	Other domain
Discrete	Periodic
Symmetric	Symmetric
Antisymmetric	Antisymmetric
Real	Conjugate Symmetric
Imaginary	Conjugate Antisymmetric
Real + Symmetric	Real + Symmetric
Real + Antisymmetric	Imaginary + Antisymmetric

Symmetric: $x[n] = x[-n]$

$$X(e^{j\omega}) = X(e^{-j\omega})$$

$$X[k] = X[(-k)_{\text{mod } N}] = X[N-k] \text{ for } k > 0$$

Conjugate Symmetric: $x[n] = x^*[-n]$

Conjugate Antisymmetric: $x[n] = -x^*[-n]$

Parseval's Theorem

2: Three Different Fourier Transforms
Fourier Transforms
Convergence of DTFT
DTFT Properties
DFT Properties
Symmetries
▷ Parseval's Theorem
Convolution
Sampling Process
Zero-Padding
Phase Unwrapping
Uncertainty principle
Summary
MATLAB routines

Fourier transforms preserve "energy"

CTFT $\int |x(t)|^2 \, dt = \frac{1}{2\pi} \int |X(j\Omega)|^2 \, d\Omega$

DTFT $\sum_{-\infty}^{\infty} |x[n]|^2 = \frac{1}{2\pi} \int_{-\pi}^{\pi} \left|X(e^{j\omega})\right|^2 d\omega$

DFT $\sum_{0}^{N-1} |x[n]|^2 = \frac{1}{N} \sum_{0}^{N-1} |X[k]|^2$

More generally, they actually preserve complex inner products:

$$\sum_{0}^{N-1} x[n]y^*[n] = \frac{1}{N} \sum_{0}^{N-1} X[k]Y^*[k]$$

Unitary matrix viewpoint for DFT:

If we regard $\mathbf{x}$ and $\mathbf{X}$ as vectors, then $\mathbf{X} = \mathbf{F}\mathbf{x}$ where $\mathbf{F}$ is a symmetric matrix defined by $f_{k+1,n+1} = e^{-j2\pi \frac{kn}{N}}$.

The inverse DFT matrix is $\mathbf{F}^{-1} = \frac{1}{N}\mathbf{F}^H$
equivalently, $\mathbf{G} = \frac{1}{\sqrt{N}}\mathbf{F}$ is a unitary matrix with $\mathbf{G}^H\mathbf{G} = \mathbf{I}$.

Convolution

2: Three Different Fourier Transforms
Fourier Transforms
Convergence of DTFT
DTFT Properties
DFT Properties
Symmetries
Parseval's Theorem
▷ Convolution
Sampling Process
Zero-Padding
Phase Unwrapping
Uncertainty principle
Summary
MATLAB routines

DTFT: Convolution $\rightarrow$ Product

$$x[n] = g[n] * h[n] = \sum_{k=-\infty}^{\infty} g[k]h[n-k]$$
$$\Rightarrow X(e^{j\omega}) = G(e^{j\omega})H(e^{j\omega})$$

DFT: Circular convolution$\rightarrow$ Product

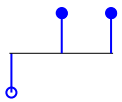
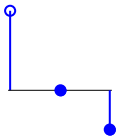
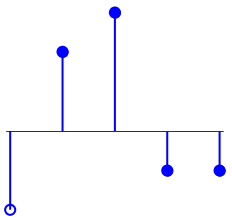
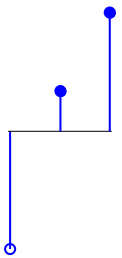
$$x[n] = g[n] \circledast_N h[n] = \sum_{k=0}^{N-1} g[k]h[(n-k)_{\text{mod}N}]$$
$$\Rightarrow X[k] = G[k]H[k]$$

DTFT: Product$\rightarrow$ Circular Convolution $\div 2\pi$

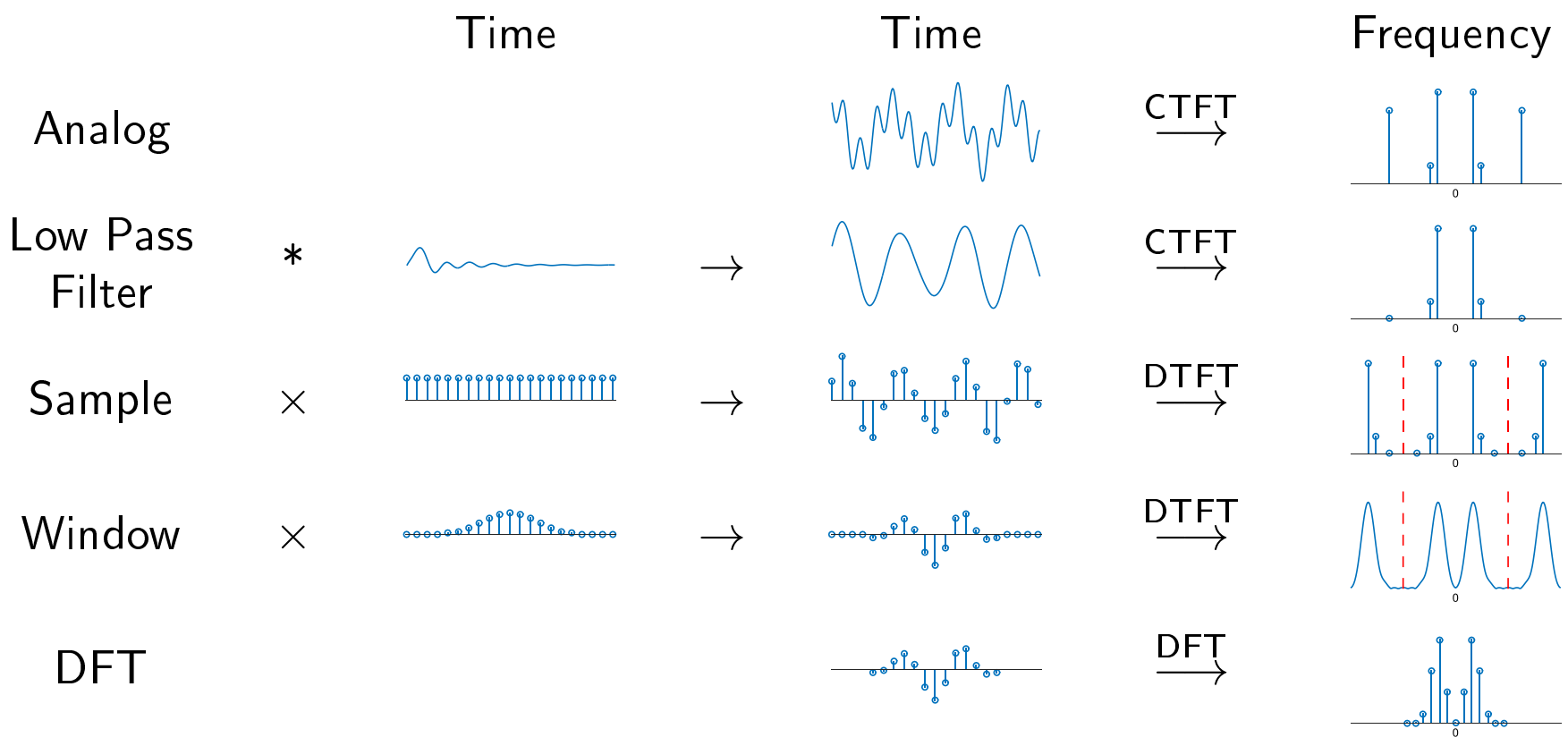
$$y[n] = g[n]h[n]$$
$$\Rightarrow Y(e^{j\omega}) = \frac{1}{2\pi} G(e^{j\omega}) \circledast_\pi H(e^{j\omega}) = \frac{1}{2\pi} \int_{-\pi}^{\pi} G(e^{j\theta})H(e^{j(\omega-\theta)})d\theta$$

DFT: Product$\rightarrow$ Circular Convolution $\div N$

$$y[n] = g[n]h[n]$$
$$\Rightarrow Y[k] = \frac{1}{N} G[k] \circledast_N H[k]$$

$g[n]$:  $h[n]$:  $g[n] * h[n]$:  $g[n] \circledast_3 h[n]$ 

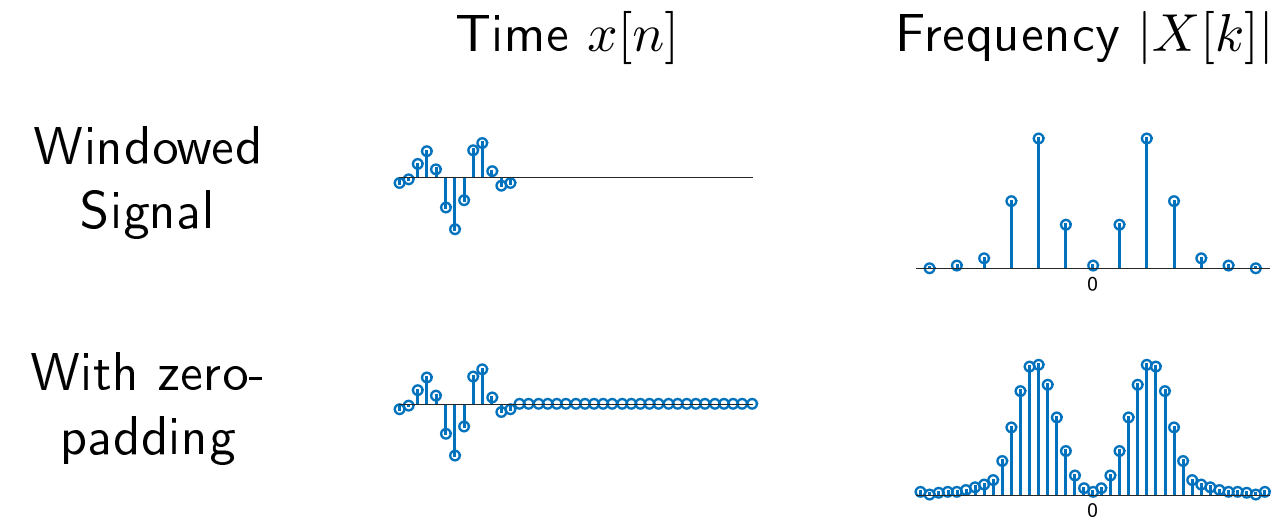
Sampling Process



Zero-Padding

2: Three Different Fourier Transforms
Fourier Transforms
Convergence of DTFT
DTFT Properties
DFT Properties
Symmetries
Parseval's Theorem
Convolution
Sampling Process
▷ Zero-Padding
Phase Unwrapping
Uncertainty principle
Summary
MATLAB routines

Zero padding means added extra zeros onto the end of $x[n]$ before performing the DFT.

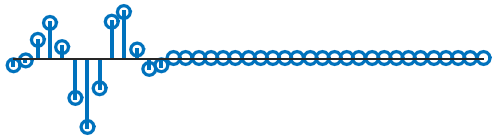


- Zero-padding causes the DFT to evaluate the DTFT at more values of ω_k. Denser frequency samples.
- Width of the peaks remains constant: determined by the length and shape of the window.
- Smoother graph but increased frequency resolution is an illusion.

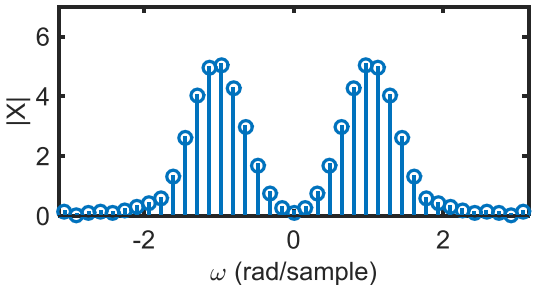
Phase Unwrapping

2: Three Different Fourier Transforms
Fourier Transforms
Convergence of DTFT
DTFT Properties
DFT Properties
Symmetries
Parseval's Theorem
Convolution
Sampling Process
Zero-Padding
▷ Phase Unwrapping
Uncertainty principle
Summary
MATLAB routines

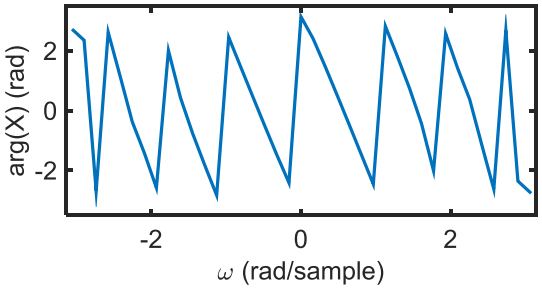
Phase of a DTFT is only defined to within an integer multiple of 2π.



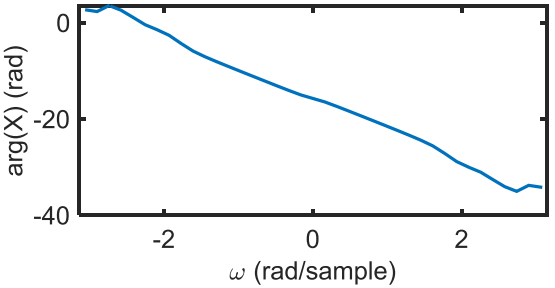
$x[n]$



$|X[k]|$



$\angle X[k]$



$\angle X[k]$ unwrapped

Phase unwrapping adds multiples of 2π onto each $\angle X[k]$ to make the phase as continuous as possible.

Uncertainty principle

2: Three Different Fourier Transforms
Fourier Transforms
Convergence of DTFT
DTFT Properties
DFT Properties
Symmetries
Parseval's Theorem
Convolution
Sampling Process
Zero-Padding
Phase Unwrapping
▷ Uncertainty principle
Summary
MATLAB routines

CTFT uncertainty principle: $\left(\frac{\int t^2|x(t)|^2dt}{\int|x(t)|^2dt}\right)^{\frac{1}{2}}\left(\frac{\int \omega^2|X(j\omega)|^2d\omega}{\int|X(j\omega)|^2d\omega}\right)^{\frac{1}{2}} \geq \frac{1}{2}$

The first term measures the "width" of $x(t)$ around $t=0$.

It is like σ if $|x(t)|^2$ was a zero-mean probability distribution.

The second term is similarly the "width" of $X(j\omega)$ in frequency.

A signal cannot be concentrated in both time and frequency.

Proof Outline:

Assume $\int |x(t)|^2\,dt = 1 \Rightarrow \int |X(j\omega)|^2\,d\omega = 2\pi$ [Parseval]

Set $v(t) = \frac{dx}{dt} \Rightarrow V(j\omega) = j\omega X(j\omega)$ [by parts]

Now $\int tx\frac{dx}{dt}dt = \frac{1}{2}tx^2(t)\Big|_{t=-\infty}^{\infty} - \int \frac{1}{2}x^2dt = 0 - \frac{1}{2}$ [by parts]

So $\frac{1}{4} = \left|\int tx\frac{dx}{dt}dt\right|^2 \leq \left(\int t^2x^2dt\right)\left(\int\left|\frac{dx}{dt}\right|^2dt\right)$ [Schwartz]

$= \left(\int t^2x^2dt\right)\left(\int |v(t)|^2\,dt\right) = \left(\int t^2x^2dt\right)\left(\frac{1}{2\pi}\int |V(j\omega)|^2\,d\omega\right)$

$= \left(\int t^2x^2dt\right)\left(\frac{1}{2\pi}\int \omega^2\,|X(j\omega)|^2\,d\omega\right)$

No exact equivalent for DTFT/DFT but a similar effect is true

[Uncertainty Principle Proof Steps]

(1) Suppose $v(t) = \frac{dx}{dt}$. Then integrating the CTFT definition by parts w.r.t. t gives
$X(j\Omega) = \int_{-\infty}^{\infty} x(t)e^{-j\Omega t}dt = \left[\frac{-1}{j\Omega}x(t)e^{-j\Omega t}\right]_{-\infty}^{\infty} + \frac{1}{j\Omega}\int_{-\infty}^{\infty}\frac{dx(t)}{dt}e^{-j\Omega t}dt = 0 + \frac{1}{j\Omega}V(j\Omega)$

(2) Since $\frac{d}{dt}\left(\frac{1}{2}x^2\right) = x\frac{dx}{dt}$, we can apply integration by parts to get
$\int_{-\infty}^{\infty} tx\frac{dx}{dt}dt = \left[t \times \frac{1}{2}x^2\right]_{t=-\infty}^{\infty} - \int_{-\infty}^{\infty}\frac{dt}{dt} \times \frac{1}{2}x^2 dt = -\frac{1}{2}\int_{-\infty}^{\infty} x^2 dt = -\frac{1}{2} \times 1 = -\frac{1}{2}$
It follows that $\left|\int_{-\infty}^{\infty} tx\frac{dx}{dt}dt\right|^2 = \left(-\frac{1}{2}\right)^2 = \frac{1}{4}$ which we will use below.

(3) The Cauchy-Schwarz inequality is that in a complex inner product space $|\mathbf{u}\cdot\mathbf{v}|^2 \leq (\mathbf{u}\cdot\mathbf{u})(\mathbf{v}\cdot\mathbf{v})$. For the inner-product space of real-valued square-integrable functions, this becomes $\left|\int_{-\infty}^{\infty} u(t)v(t)dt\right|^2 \leq \int_{-\infty}^{\infty} u^2(t)dt \times \int_{-\infty}^{\infty} v^2(t)dt$. We apply this with $u(t) = tx(t)$ and $v(t) = \frac{dx(t)}{dt}$ to get
$$\frac{1}{4} = \left|\int_{-\infty}^{\infty} tx\frac{dx}{dt}dt\right|^2 \leq \left(\int t^2x^2dt\right)\left(\int\left(\frac{dx}{dt}\right)^2 dt\right) = \left(\int t^2x^2dt\right)\left(\int v^2(t)dt\right)$$

(4) From Parseval's theorem for the CTFT, $\int v^2(t)dt = \frac{1}{2\pi}\int |V(j\Omega|^2\, d\Omega$. From step (1), we can substitute $V(j\Omega) = j\Omega X(j\Omega)$ to obtain $\int v^2(t)dt = \frac{1}{2\pi}\int \Omega^2\, |X(j\Omega|^2\, d\Omega$. Making this substitution in (3) gives
$$\frac{1}{4} \leq \left(\int t^2x^2dt\right)\left(\int v^2(t)dt\right) = \left(\int t^2x^2dt\right)\left(\frac{1}{2\pi}\int \omega^2\, |X(j\Omega|^2\, d\Omega\right)$$

Summary

2: Three Different Fourier Transforms
Fourier Transforms
Convergence of DTFT
DTFT Properties
DFT Properties
Symmetries
Parseval's Theorem
Convolution
Sampling Process
Zero-Padding
Phase Unwrapping
Uncertainty principle
▷ Summary
MATLAB routines

- Three types: CTFT, DTFT, DFT
  - DTFT = CTFT of continuous signal $\times$ impulse train
  - DFT = DTFT of periodic or finite support signal
    - DFT is a scaled unitary transform
- DTFT: Convolution $\rightarrow$ Product; Product $\rightarrow$ Circular Convolution
- DFT: Product $\leftrightarrow$ Circular Convolution
- DFT: Zero Padding $\rightarrow$ Denser freq sampling but same resolution
- Phase is only defined to within a multiple of 2π.
- Whenever you integrate over frequency you need a scale factor
  - $\frac{1}{2\pi}$ for CTFT and DTFT or $\frac{1}{N}$ for DFT
  - e.g. Inverse transform, Parseval, frequency domain convolution

For further details see Mitra: 3 & 5.

MATLAB routines

2: Three Different Fourier Transforms
Fourier Transforms
Convergence of DTFT
DTFT Properties
DFT Properties
Symmetries
Parseval's Theorem
Convolution
Sampling Process
Zero-Padding
Phase Unwrapping
Uncertainty principle
Summary
▷ MATLAB routines

fft, ifft	DFT with optional zero-padding
fftshift	swap the two halves of a vector
conv	convolution or polynomial multiplication (not circular)
$x[n] \circledast y[n]$	real(ifft(fft(x).*fft(y)))
unwrap	remove 2π jumps from phase spectrum

3: Discrete Cosine Transform
DFT Problems
DCT +
Basis Functions
DCT of sine wave
DCT Properties
Energy Conservation
Energy Compaction
Frame-based coding
Lapped Transform +
MDCT (Modified DCT)
MDCT Basis Elements
Summary
MATLAB routines

3: Discrete Cosine Transform

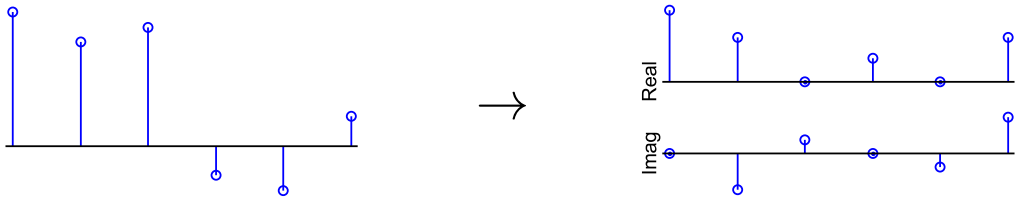
DFT Problems

3: Discrete Cosine Transform
▷ DFT Problems
DCT +
Basis Functions
DCT of sine wave
DCT Properties
Energy Conservation
Energy Compaction
Frame-based coding
Lapped Transform +
MDCT (Modified DCT)
MDCT Basis Elements
Summary
MATLAB routines

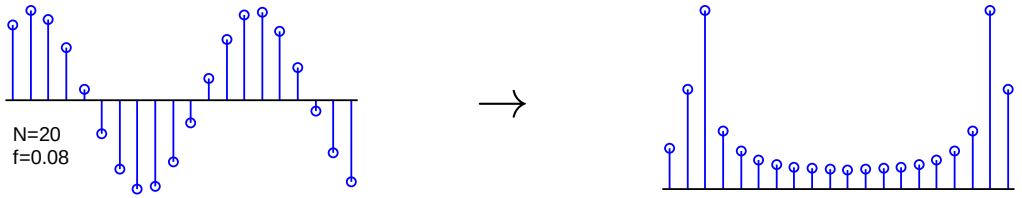
For processing 1-D or 2-D signals (especially coding), a common method is to divide the signal into "frames" and then apply an invertible transform to each frame that compresses the information into few coefficients.

The DFT has some problems when used for this purpose:

- N real $x[n] \leftrightarrow N$ complex $X[k]$: 2 real, $\frac{N}{2} - 1$ conjugate pairs



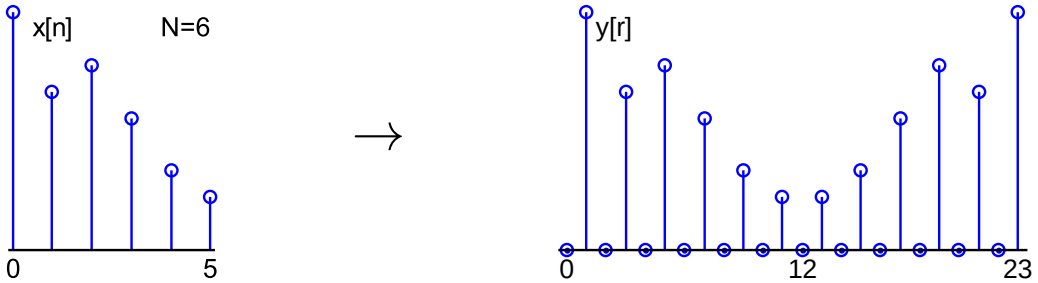
- DFT $\propto$ the DTFT of a periodic signal formed by replicating $x[n]$.
  $\Rightarrow$ Spurious frequency components from boundary discontinuity.



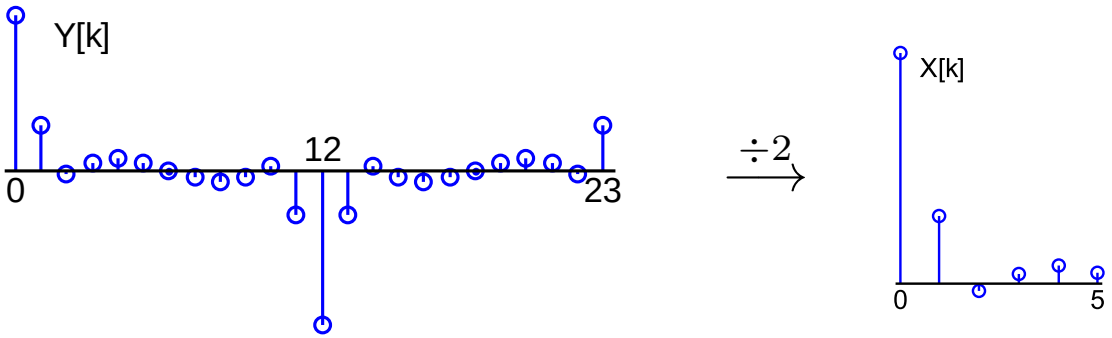
The Discrete Cosine Transform (DCT) overcomes these problems.

3: Discrete Cosine Transform
DFT Problems
▷ DCT +
Basis Functions
DCT of sine wave
DCT Properties
Energy Conservation
Energy Compaction
Frame-based coding
Lapped Transform +
MDCT (Modified DCT)
MDCT Basis Elements
Summary
MATLAB routines

To form the Discrete Cosine Transform (DCT), replicate $x[0:N-1]$ but in reverse order and insert a zero between each pair of samples:



Take the DFT of length $4N$ real, symmetric, odd-sample-only sequence.
Result is real, symmetric and anti-periodic: only need first N values



Forward DCT: $X_C[k] = \sum_{n=0}^{N-1} x[n] \cos \frac{2\pi(2n+1)k}{4N}$ for $k = 0 : N-1$

Inverse DCT: $x[n] = \frac{1}{N} X[0] + \frac{2}{N} \sum_{k=1}^{N-1} X[k] \cos \frac{2\pi(2n+1)k}{4N}$

DCT formula derivation

This proof is not examinable.

We want to show that $X_C[k] = \sum_{n=0}^{N-1} x[n] \cos \frac{2\pi(2n+1)k}{4N}$ is equivalent to replicating $x[n]$ in reverse order, inserting alternate zeros, taking DFT, dividing by 2 and keeping first N values:

Replicating + zero insertion gives $y[r] = \begin{cases} 0 & r \text{ even} \\ x\left[\frac{r-1}{2}\right] & r \text{ odd}, 1 \le r \le 2N-1 \\ x\left[\frac{4N-1-r}{2}\right] & r \text{ odd}, 2N+1 \le r \le 4N-1 \end{cases}$

$Y_F[k] = \sum_{r=0}^{4N-1} y[r] W_{4N}^{kr} \overset{\text{(i)}}{=} \sum_{n=0}^{2N-1} y[2n+1] W_{4N}^{(2n+1)k}$ where $W_a^b = e^{-j\frac{2\pi b}{a}}$

$\overset{\text{(ii)}}{=} \sum_{n=0}^{N-1} y[2n+1] W_{4N}^{(2n+1)k} + \sum_{m=0}^{N-1} y[4N-2m-1] W_{4N}^{(4N-2m-1)k}$

$\overset{\text{(iii)}}{=} \sum_{n=0}^{N-1} x[n] W_{4N}^{(2n+1)k} + \sum_{m=0}^{N-1} x[m] W_{4N}^{-(2m+1)k}$

$= 2\sum_{n=0}^{N-1} x[n] \cos \frac{2\pi(2n+1)k}{4N} = 2X_C[k]$

(i) odd r only: $r = 2n+1$

(ii) reverse order for $n \ge N$: $m = 2N-1-n$

(iii) substitute y definition & $W_{4N}^{4Nk} = e^{-j2\pi\frac{4Nk}{4N}} \equiv 1$

IDCT formula derivation

This proof is not examinable.

We want to show that $x[n] = \frac{1}{N}X[0] + \frac{2}{N}\sum_{k=1}^{N-1} X[k]\cos\frac{2\pi(2n+1)k}{4N}$

Since $Y[k] = 2X[k]$ we can write $y[r] = \frac{1}{4N}\sum_{k=0}^{4N-1} Y[k]W_{4N}^{-rk} = \frac{1}{2N}\sum_{k=0}^{4N-1} X[k]W_{4N}^{-rk}$

So we can write,

$x[n] = y[2n+1] = \frac{1}{2N}\sum_{k=0}^{4N-1} X[k]W_{4N}^{-(2n+1)k}$ where $W_a^b = e^{-j\frac{2\pi b}{a}}$

$\overset{\text{(i)}}{=} \frac{1}{2N}\sum_{k=0}^{2N-1} X[k]W_{4N}^{-(2n+1)k} - \frac{1}{2N}\sum_{l=0}^{2N-1} X[l]W_{4N}^{-(2n+1)(l+2N)}$

$\overset{\text{(ii)}}{=} \frac{1}{N}\sum_{k=0}^{2N-1} X[k]W_{4N}^{-(2n+1)k}$

$\overset{\text{(iii)}}{=} \frac{1}{N}X[0] + \frac{1}{N}\sum_{k=1}^{N-1} X[k]W_{4N}^{-(2n+1)k}$
$+\frac{1}{N}X[N]W_{4N}^{-(2n+1)N} + \frac{1}{N}\sum_{r=1}^{N-1} X[2N-r]W_{4N}^{-(2n+1)(2N-r)}$

$\overset{\text{(iv)}}{=} \frac{1}{N}X[0] + \frac{1}{N}\sum_{k=1}^{N-1} X[k]W_{4N}^{-(2n+1)k} + \frac{1}{N}\sum_{r=1}^{N-1} -X[r]W_{4N}^{(2n+1)r+2N}$

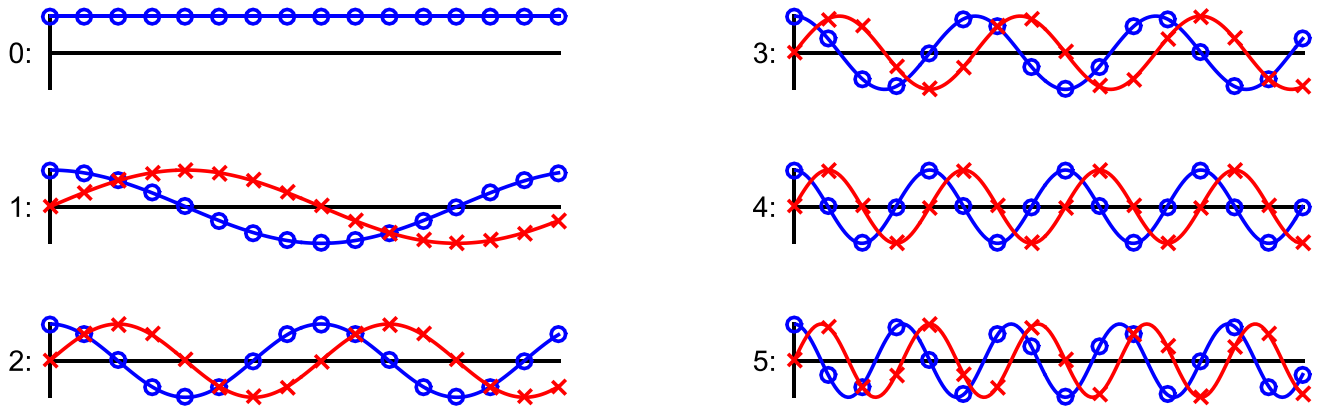
$= \frac{1}{N}X[0] + \frac{2}{N}\sum_{k=1}^{N-1} X[k]\cos\frac{2\pi(2n+1)k}{4N}$

Notes: (i) $k = l + 2N$ for $k \geq 2N$ and $X[k+2N] = -X[k]$
(ii) $\frac{(2n+1)(l+2N)}{4N} = \frac{(2n+1)l}{4N} + n + \frac{1}{2}$ and $e^{j2\pi\left(n+\frac{1}{2}\right)} = -1$
(iii) $k = 2N - r$ for $k > N$
(iv) $X[N] = 0$ and $X[2N-r] = -X[r]$

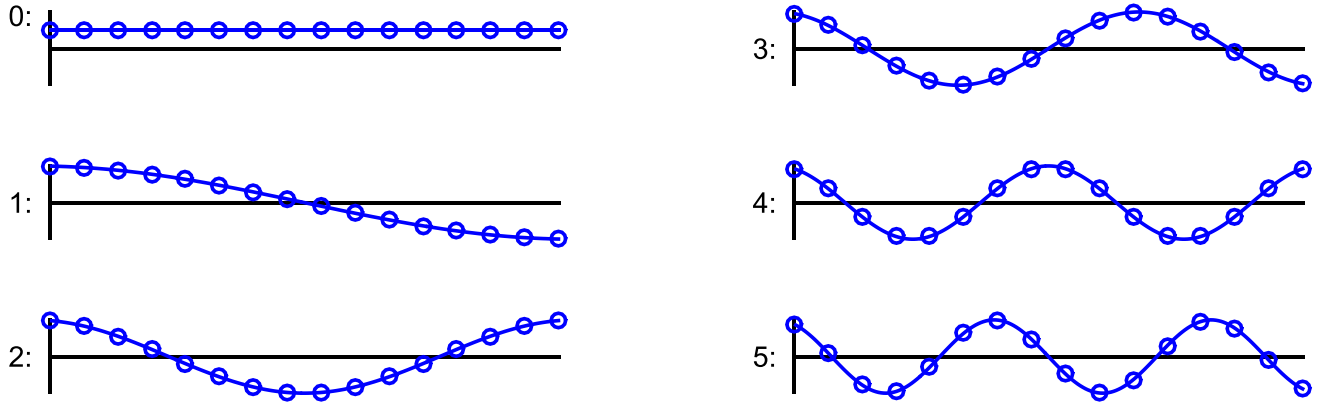
Basis Functions

3: Discrete Cosine Transform
DFT Problems
DCT +
▷ Basis Functions
DCT of sine wave
DCT Properties
Energy Conservation
Energy Compaction
Frame-based coding
Lapped Transform +
MDCT (Modified DCT)
MDCT Basis Elements
Summary
MATLAB routines

DFT basis functions: $x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j2\pi \frac{kn}{N}}$



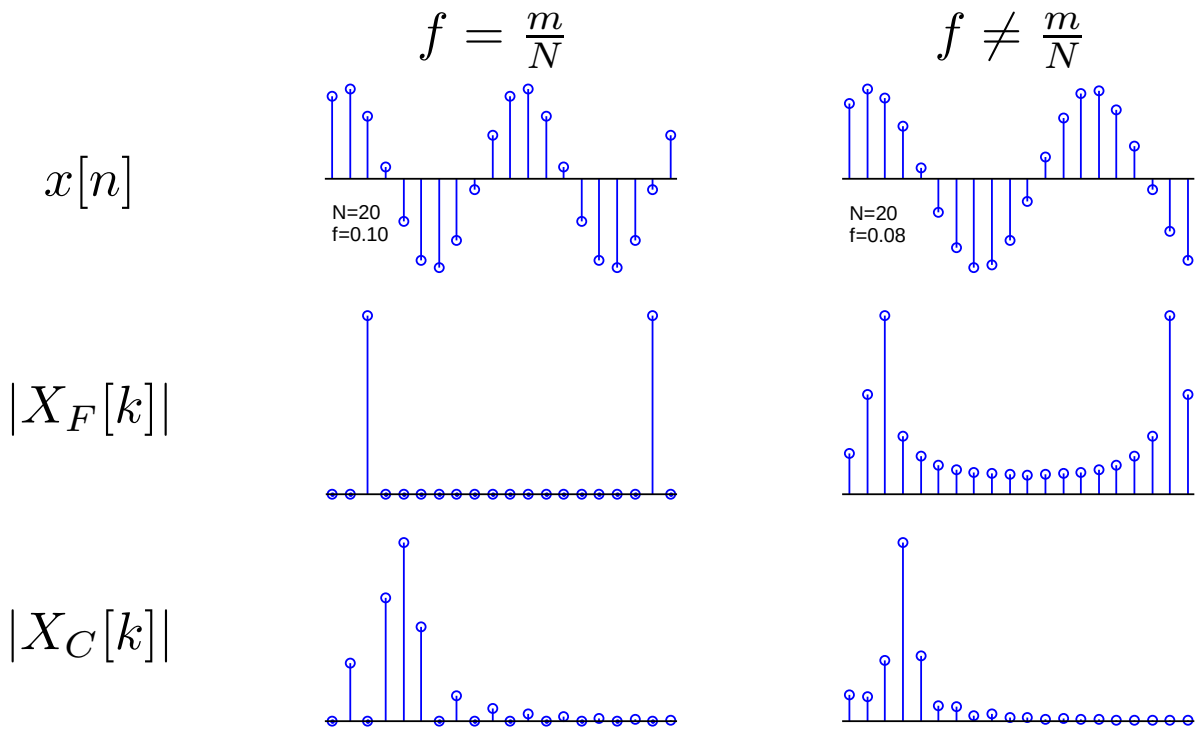
DCT basis functions: $x[n] = \frac{1}{N} X[0] + \frac{2}{N} \sum_{k=1}^{N-1} X[k] \cos \frac{2\pi(2n+1)k}{4N}$



DCT of sine wave

3: Discrete Cosine Transform
DFT Problems
DCT +
Basis Functions
▷ DCT of sine wave
DCT Properties
Energy Conservation
Energy Compaction
Frame-based coding
Lapped Transform +
MDCT (Modified DCT)
MDCT Basis Elements
Summary
MATLAB routines

DCT: $X_C[k] = \sum_{n=0}^{N-1} x[n] \cos \frac{2\pi(2n+1)k}{4N}$



DFT: Real→Complex; Freq range $[0, 1]$; Poorly localized unless $f = \frac{m}{N}$; $|X_F[k]| \propto k^{-1}$ for $Nf < k \ll \frac{N}{2}$

DCT: Real→Real; Freq range $[0, 0.5]$; Well localized $\forall f$; $|X_C[k]| \propto k^{-2}$ for $2Nf < k < N$

DCT Properties

3: Discrete Cosine Transform
DFT Problems
DCT +
Basis Functions
DCT of sine wave
▷ DCT Properties
Energy Conservation
Energy Compaction
Frame-based coding
Lapped Transform +
MDCT (Modified DCT)
MDCT Basis Elements
Summary
MATLAB routines

Definition: $X[k] = \sum_{n=0}^{N-1} x[n] \cos \frac{2\pi(2n+1)k}{4N}$

- Linear: $\alpha x[n] + \beta y[n] \rightarrow \alpha X[k] + \beta Y[k]$
- "Convolution$\longleftrightarrow$Multiplication" property of DFT does not hold ☹
- Symmetric: $X[-k] = X[k]$ since $\cos -\alpha k = \cos +\alpha k$
- Anti-periodic: $X[k+2N] = -X[k]$ because:
  - $2\pi(2n+1)(k+2N) = 2\pi(2n+1)k + 8\pi Nn + 4N\pi$
  - $\cos(\theta + \pi) = -\cos\theta$

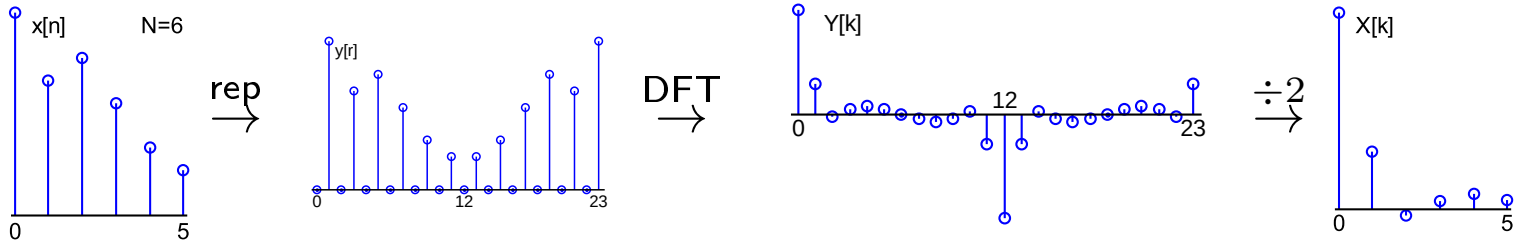
  $\Rightarrow X[N] = 0$ since $X[N] = X[-N] = -X[-N+2N]$
- Periodic: $X[k+4N] = -X[k+2N] = X[k]$

Energy Conservation

3: Discrete Cosine Transform
DFT Problems
DCT +
Basis Functions
DCT of sine wave
DCT Properties
Energy Conservation
Energy Compaction
Frame-based coding
Lapped Transform +
MDCT (Modified DCT)
MDCT Basis Elements
Summary
MATLAB routines

DCT: $X[k] = \sum_{n=0}^{N-1} x[n] \cos \frac{2\pi(2n+1)k}{4N}$

IDCT: $x[n] = \frac{1}{N} X[0] + \frac{2}{N} \sum_{k=1}^{N-1} X[k] \cos \frac{2\pi(2n+1)k}{4N}$



Energy: $E = \sum_{n=0}^{N-1} |x[n]|^2 = \frac{1}{N} |X[0]|^2 + \frac{2}{N} \sum_{n=1}^{N-1} |X[n]|^2$

In diagram above: $E \rightarrow 2E \rightarrow 8NE \rightarrow \approx 0.5NE$

Orthogonal DCT (preserves energy: $\sum |x[n]|^2 = \sum |X[n]|^2$)

$$\text{ODCT: } X[k] = \begin{cases} \sqrt{\frac{1}{N}} \sum_{n=0}^{N-1} x[n] & k = 0 \\ \sqrt{\frac{2}{N}} \sum_{n=0}^{N-1} x[n] \cos \frac{2\pi(2n+1)k}{4N} & k \neq 0 \end{cases}$$

IODCT: $x[n] = \sqrt{\frac{1}{N}} X[0] + \sqrt{\frac{2}{N}} \sum_{k=1}^{N-1} X[k] \cos \frac{2\pi(2n+1)k}{4N}$

Note: MATLAB dct() calculates the ODCT

Energy Compaction

3: Discrete Cosine Transform
DFT Problems
DCT +
Basis Functions
DCT of sine wave
DCT Properties
Energy Conservation
Energy Compaction
Frame-based coding
Lapped Transform +
MDCT (Modified DCT)
MDCT Basis Elements
Summary
MATLAB routines

If consecutive $x[n]$ are positively correlated, DCT concentrates energy in a few $X[k]$ and decorrelates them.

Example: Markov Process: $x[n] = \rho x[n-1] + \sqrt{1-\rho^2}u[n]$

where $u[n]$ is i.i.d. unit Gaussian.

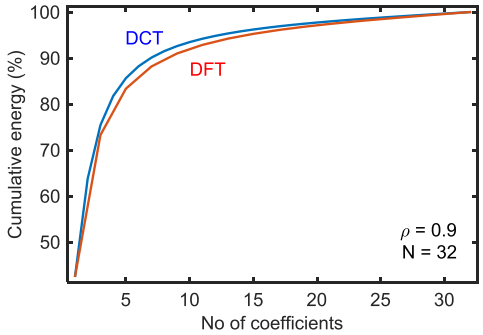
Then $\left\langle x^2[n] \right\rangle = 1$ and $\left\langle x[n]x[n-1] \right\rangle = \rho$.

Covariance of vector $\mathbf{x}$ is $\mathbf{S}_{i,j} = \left\langle \mathbf{x}\mathbf{x}^H \right\rangle_{i,j} = \rho^{|i-j|}$.

Suppose ODCT of $\mathbf{x}$ is $\mathbf{Cx}$ and DFT is $\mathbf{Fx}$.

Covariance of $\mathbf{Cx}$ is $\left\langle \mathbf{C}\mathbf{x}\mathbf{x}^H\mathbf{C}^H \right\rangle = \mathbf{C}\mathbf{S}\mathbf{C}^H$ (similarly $\mathbf{F}\mathbf{S}\mathbf{F}^H$)

Diagonal elements give mean coefficient energy.



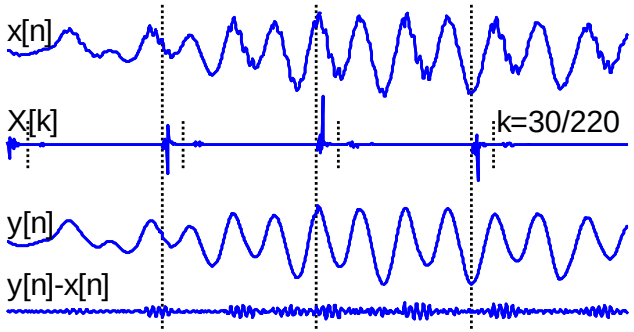
- Used in MPEG and JPEG (superseded by JPEG2000 using wavelets)
- Used in speech recognition to decorrelate spectral coeficients: DCT of log spectrum

Energy compaction good for coding (low-valued coefficients can be set to 0)
Decorrelation good for coding and for probability modelling

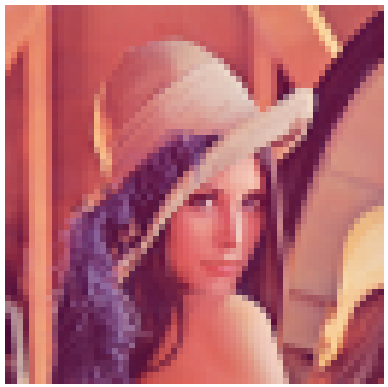
Frame-based coding

3: Discrete Cosine Transform
DFT Problems
DCT +
Basis Functions
DCT of sine wave
DCT Properties
Energy Conservation
Energy Compaction
▷ Frame-based coding
Lapped Transform +
MDCT (Modified DCT)
MDCT Basis Elements
Summary
MATLAB routines

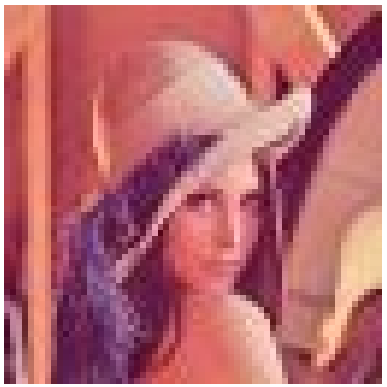
- Divide continuous signal into frames
- Apply DCT to each frame
- Encode DCT
  - e.g. keep only 30 $X[k]$
- Apply IDCT $\rightarrow y[n]$



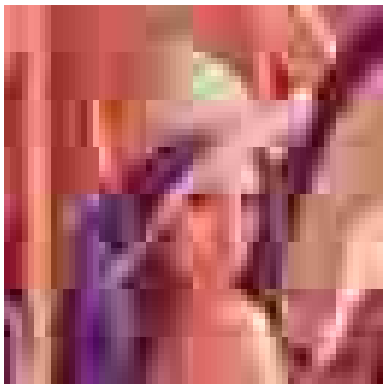
Problem: Coding may create discontinuities at frame boundaries
e.g. JPEG, MPEG use 8×8 pixel blocks



8.3 kB (PNG)



1.6 kB (JPEG)



0.5 kB (JPEG)

3: Discrete Cosine Transform
DFT Problems
DCT
Basis Functions
DCT of sine wave
DCT Properties
Energy Conservation
Energy Compaction
Frame-based coding
Lapped Transform
MDCT (Modified DCT)
MDCT Basis Elements
Summary
MATLAB routines

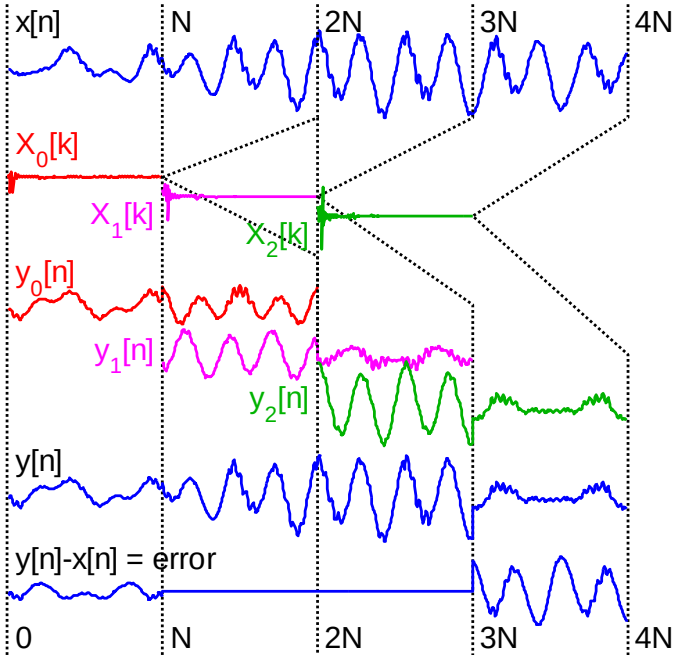
Modified Discrete Cosine Transform (MDCT): overlapping frames $2N$ long

$$x[0:2N-1] \xrightarrow{\text{MDCT}} X_0[0:N-1] \xrightarrow{\text{IMDCT}} y_0[0:2N-1]$$

$$x[N:3N-1] \xrightarrow{\text{MDCT}} X_1[N:2N-1] \xrightarrow{\text{IMDCT}} y_1[N:3N-1]$$

$$x[2N:4N-1] \xrightarrow{\text{MDCT}} X_2[2N:3N-1] \xrightarrow{\text{IMDCT}} y_2[2N:4N-1]$$

$$y[n] = y_0[n] + y_1[n] + y_2[n]$$



MDCT: $2N \to N$ coefficients, IMDCT: $N \to 2N$ samples
Add $y_i[n]$ together to get $y[n]$. Only two non-zero terms far any n.
Errors cancel exactly: Time-domain alias cancellation (TDAC)

MDCT (Modified DCT)

MDCT: $X[k] = \sum_{n=0}^{2N-1} x[n] \cos \frac{2\pi(2n+1+N)(2k+1)}{8N}$ $\quad 0 \le k < N$

IMDCT: $y[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] \cos \frac{2\pi(2n+1+N)(2k+1)}{8N}$ $\quad 0 \le n < 2N$

If $\mathbf{x}$, $\mathbf{X}$ and $\mathbf{y}$ are column vectors, then $\mathbf{X} = \mathbf{M}\mathbf{x}$ and $\mathbf{y} = \frac{1}{N}\mathbf{M}^T\mathbf{X} = \frac{1}{N}\mathbf{M}^T\mathbf{M}\mathbf{x}$

where $\mathbf{M}$ is an $N \times 2N$ matrix with $m_{k,n} = \cos \frac{2\pi(2n+1+N)(2k+1)}{8N}$.

Quasi-Orthogonality: The $2N \times 2N$matrix, $\frac{1}{N}\mathbf{M}^T\mathbf{M}$, is almost the identity:

$$\frac{1}{N}\mathbf{M}^T\mathbf{M} = \frac{1}{2}\begin{bmatrix} \mathbf{I}-\mathbf{J} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}+\mathbf{J} \end{bmatrix} \text{ with } \mathbf{I} = \begin{bmatrix} 1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 1 \end{bmatrix}, \mathbf{J} = \begin{bmatrix} 0 & \cdots & 1 \\ \vdots & \iddots & \vdots \\ 1 & \cdots & 0 \end{bmatrix}$$

When two consective $\mathbf{y}$ frames are overlapped by N samples, the second half of the first frame has thus been multiplied by $\frac{1}{2}(\mathbf{I}+\mathbf{J})$ and the first half of the second frame by $\frac{1}{2}(\mathbf{I}-\mathbf{J})$. When these $\mathbf{y}$ frames are added together, the corresponding $\mathbf{x}$ samples have been multiplied by $\frac{1}{2}(\mathbf{I}+\mathbf{J}) + \frac{1}{2}(\mathbf{I}-\mathbf{J}) = \mathbf{I}$ giving perfect reconstruction.

Normally the $2N$-long $\mathbf{x}$ and $\mathbf{y}$ frames are windowed before the MDCT and again after the IMDCT to avoid any discontinuities; if the window is symmetric and satisfies $w^2[i] + w^2[i+N] = 2$ the perfect reconstruction property is still true.

[Deriving the value of $\frac{1}{N}\mathbf{M}^T\mathbf{M}$]

This proof is not examinable.

If we define $\mathbf{A} = \frac{1}{\mathbf{N}}\mathbf{M^T M}$ with $m_{kn} = \cos\frac{2\pi(2n+1+N)(2k+1)}{8N}$, we want to show that $\mathbf{A} = \frac{1}{2}\begin{bmatrix} \mathbf{I+J} & \mathbf{0} \\ \mathbf{0} & \mathbf{I-J} \end{bmatrix}$. To avoid fractions, we write $\alpha = \frac{2\pi}{8N}$ so that $m_{kn} = \cos(\alpha(2n+1+N)(2k+1))$. Now we can say

$$
\begin{aligned}
a_{rn} &= \frac{1}{N}\sum_{k=0}^{N-1} m_{kr}m_{kn} \\
&= \frac{1}{N}\sum_{k=0}^{N-1} \cos(\alpha(2r+1+N)(2k+1))\cos(\alpha(2n+1+N)(2k+1)) \\
&= \frac{1}{2N}\sum_{k=0}^{N-1} \cos(2\alpha(r-n)(2k+1)) + \frac{1}{2N}\sum_{k=0}^{N-1} \cos(2\alpha(r+n+1+N)(2k+1))
\end{aligned}
$$

where, in the last line, we used the identity $\cos\theta\cos\phi = \frac{1}{2}\cos(\theta-\phi) + \frac{1}{2}\cos(\theta+\phi)$.

We now convert these terms to complex exponentials to sum them as geometric progressions.

$[\frac{1}{2N}\sum_{k=0}^{N-1}\cos\left(2\alpha(r-n)(2k+1)\right)]$

Converting to a the real part ($\Re$) of geometric progression (with $\alpha = \frac{2\pi}{8N}$):

$$\begin{aligned}
\frac{1}{2N}\sum_{k=0}^{N-1}\cos\left(2\alpha(r-n)(2k+1)\right) &= \frac{1}{2N}\Re\left(\sum_{k=0}^{N-1}\exp\left(j2\alpha(r-n)(2k+1)\right)\right)\\
&= \frac{1}{2N}\Re\left(\exp\left(j2\alpha(r-n)\right)\sum_{k=0}^{N-1}\exp\left(j4\alpha(r-n)k\right)\right)\\
&= \frac{1}{2N}\Re\left(\exp\left(j2\alpha(r-n)\right)\frac{1-\exp\left(j4\alpha(r-n)N\right)}{1-\exp\left(j4\alpha(r-n)\right)}\right)\\
&= \frac{1}{2N}\Re\left(\frac{1-\exp\left(j4\alpha(r-n)N\right)}{\exp\left(-j2\alpha(r-n)\right)-\exp\left(j2\alpha(r-n)\right)}\right)\\
&= \frac{1}{2N}\Re\left(\frac{1-\exp\left(j4\alpha(r-n)N\right)}{-2j\sin\left(2\alpha(r-n)\right)}\right)\\
&= \frac{1}{4N}\frac{\sin\left(4\alpha(r-n)N\right)}{\sin\left(2\alpha(r-n)\right)} = \frac{1}{4N}\frac{\sin\left((r-n)\pi\right)}{\sin\left(\frac{r-n}{2N}\pi\right)}
\end{aligned}$$

The numerator is sine of a multiple of π and is therefore 0. Therefore the whole sum is zero unless the denominator is zero or, equivalently, $(r-n)$ is a multiple of $2N$. Since $0 \le r, n < 2N$, this only happens when $r = n$ in which case the sum becomes $\frac{1}{2N}\sum_{k=0}^{N-1}\cos 0 = \frac{1}{2}$.

$[\frac{1}{2N}\sum_{k=0}^{N-1}\cos\left(2\alpha(r+n+1+N)(2k+1)\right)]$

$\frac{1}{2N}\sum_{k=0}^{N-1}\cos\left(2\alpha(r+n+1+N)(2k+1)\right)$ is the same as before with $r-n$ replaced by $r+n+1+N$.

We can therefore write

$$\frac{1}{2N}\sum_{k=0}^{N-1}\cos\left(2\alpha(r+n+1+N)(2k+1)\right) \quad = \quad \frac{1}{4N}\frac{\sin\left((r+n+1+N)\pi\right)}{\sin\left(\frac{r+n+1+N}{2N}\pi\right)}$$

The numerator is again the sine of a multiple of π and is therefore 0. Therefore the whole sum is zero unless $(r+n+1+N)$ is a multiple of $2N$. This only happens when $r+n=N-1$ or $3N-1$ since $0 \leq r,\, n < 2N$. The constraint $r+n=N-1$ corresponds to the anti-diagonal of the top left quadrant of the $\mathbf{A}$ matrix, while $r+n=3N-1$ corresponds to the anti-diagonal of the bottom right quadrant.

Writing $r+n+1+N=x$, we can use L'Hôpital's rule to evaluate $\frac{1}{4N}\frac{\sin(x\pi)}{\sin\left(\frac{x}{2N}\pi\right)}$ at $x=\{2N,\,4N\}$.

Differentiating numerator and denominator gives $\frac{1}{2}\frac{\cos(x\pi)}{\cos\left(\frac{x}{2N}\pi\right)}$ which comes to $\left\{-\frac{1}{2},\,\frac{1}{2}\right\}$ respectively at $x=\{2N,\,4N\}$.

MDCT Basis Elements

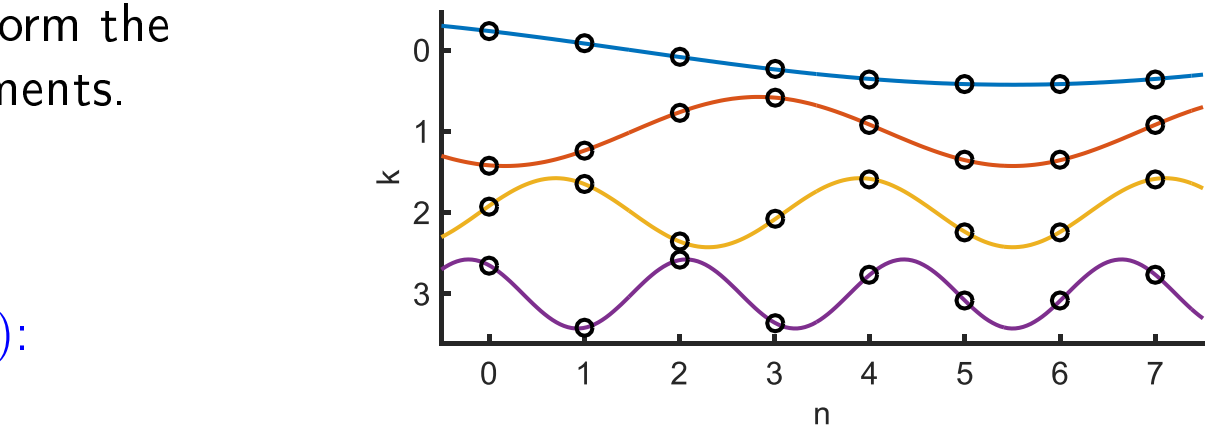
3: Discrete Cosine Transform
DFT Problems
DCT +
Basis Functions
DCT of sine wave
DCT Properties
Energy Conservation
Energy Compaction
Frame-based coding
Lapped Transform +
MDCT (Modified DCT)
▷ MDCT Basis Elements
Summary
MATLAB routines

MDCT: $X[k] = \sum_{n=0}^{2N-1} x[n] \cos \frac{2\pi(2n+1+N)(2k+1)}{8N}$ $\quad 0 \le k < N$

IMDCT: $y[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] \cos \frac{2\pi(2n+1+N)(2k+1)}{8N}$ $\quad 0 \le n < 2N$

In vector notation: $\mathbf{X} = \mathbf{M}\mathbf{x}$ and $\mathbf{y} = \frac{1}{N}\mathbf{M}^T\mathbf{X} = \frac{1}{N}\mathbf{M}^T\mathbf{M}\mathbf{x}$

The rows of $\mathbf{M}$ form the MDCT basis elements.



Example ($N = 4$):

$$\mathbf{M} = \begin{bmatrix} 0.56 & 0.20 & -0.20 & -0.56 & -0.83 & -0.98 & -0.98 & -0.83 \\ -0.98 & -0.56 & 0.56 & 0.98 & 0.20 & -0.83 & -0.83 & 0.20 \\ 0.20 & 0.83 & -0.83 & -0.20 & 0.98 & -0.56 & -0.56 & 0.98 \\ 0.83 & -0.98 & 0.98 & -0.83 & 0.56 & -0.20 & -0.20 & 0.56 \end{bmatrix}$$

The basis frequencies are $\{0.5, 1.5, 2.5, 3.5\}$ times the fundamental.

Summary

3: Discrete Cosine Transform
DFT Problems
DCT +
Basis Functions
DCT of sine wave
DCT Properties
Energy Conservation
Energy Compaction
Frame-based coding
Lapped Transform +
MDCT (Modified DCT)
MDCT Basis Elements
▷ Summary
MATLAB routines

DCT: Discrete Cosine Transform

- Equivalent to a DFT of time-shifted double-length $\left[\mathbf{x} \quad \overleftarrow{\mathbf{x}} \right]$
- Often scaled to make an orthogonal transform (ODCT)
- Better than DFT for energy compaction and decorrelation ☺
  - Energy Compaction: Most energy is in only a few coefficients
  - Decorrelation: The coefficients are uncorrelated with each other
- Nice convolution property of DFT is lost ☹

MDCT: Modified Discrete Cosine Transform

- Lapped transform: $2N \rightarrow N \rightarrow 2N$
- Aliasing errors cancel out when overlapping output frames are added
- Similar to DCT for energy compaction and decorrelation ☺
- Overlapping windowed frames can avoid edge discontinuities ☺
- Used in audio coding: MP3, WMA, AC-3, AAC, Vorbis, ATRAC

For further details see Mitra: 5.

MATLAB routines

3: Discrete Cosine Transform
DFT Problems
DCT +
Basis Functions
DCT of sine wave
DCT Properties
Energy Conservation
Energy Compaction
Frame-based coding
Lapped Transform +
MDCT (Modified DCT)
MDCT Basis Elements
Summary
▷ MATLAB routines

dct, idct	ODCT with optional zero-padding

4: Linear Time Invariant Systems
LTI Systems
Convolution Properties
BIBO Stability
Frequency Response
Causality
Convolution Complexity
Circular Convolution
Frequency-domain convolution
Overlap Add
Overlap Save
Summary
MATLAB routines

4: Linear Time Invariant Systems

LTI Systems

4: Linear Time Invariant Systems
▷ LTI Systems
Convolution Properties
BIBO Stability
Frequency Response
Causality +
Convolution Complexity
Circular Convolution
Frequency-domain convolution
Overlap Add
Overlap Save
Summary
MATLAB routines

Linear Time-invariant (LTI) systems have two properties:

Linear: $\mathscr{H}\left(\alpha u[n]+\beta v[n]\right)=\alpha\mathscr{H}\left(u[n]\right)+\beta\mathscr{H}\left(v[n]\right)$
Time Invariant: $y[n]=\mathscr{H}\left(x[n]\right)\Rightarrow y[n-r]=\mathscr{H}\left(x[n-r]\right)\forall r$

The behaviour of an LTI system is completely defined by its impulse response: $h[n]=\mathscr{H}\left(\delta[n]\right)$

Proof:

We can always write $x[n]=\sum_{r=-\infty}^{\infty}x[r]\delta[n-r]$

Hence
$$\begin{aligned}\mathscr{H}\left(x[n]\right)&=\mathscr{H}\left(\sum_{r=-\infty}^{\infty}x[r]\delta[n-r]\right)\\&=\sum_{r=-\infty}^{\infty}x[r]\mathscr{H}\left(\delta[n-r]\right)\\&=\sum_{r=-\infty}^{\infty}x[r]h[n-r]\\&=x[n]*h[n]\end{aligned}$$

Convolution Properties

4: Linear Time Invariant Systems
LTI Systems
Convolution
▷ Properties
BIBO Stability
Frequency Response
Causality +
Convolution Complexity
Circular Convolution
Frequency-domain convolution
Overlap Add
Overlap Save
Summary
MATLAB routines

Convolution: $x[n] * v[n] = \sum_{r=-\infty}^{\infty} x[r]v[n-r]$

Convolution obeys normal arithmetic rules for multiplication:

Commutative: $x[n] * v[n] = v[n] * x[n]$

Proof: $\sum_r x[r]v[n-r] \overset{\text{(i)}}{=} \sum_p x[n-p]v[p]$

(i) substitute $p = n - r$

Associative: $x[n] * (v[n] * w[n]) = (x[n] * v[n]) * w[n]$

$\Rightarrow x[n] * v[n] * w[n]$ is unambiguous

Proof: $\sum_{r,s} x[n-r]v[r-s]w[s] \overset{\text{(i)}}{=} \sum_{p,q} x[p]v[q-p]w[n-q]$

(i) substitute $p = n - r$, $q = n - s$

Distributive over $+$:

$x[n] * (\alpha v[n] + \beta w[n]) = (x[n] * \alpha v[n]) + (x[n] * \beta w[n])$

Proof: $\sum_r x[n-r]\left(\alpha v[r] + \beta w[r]\right) =$

$\alpha \sum_r x[n-r]v[r] + \beta \sum_r x[n-r]w[r]$

Identity: $x[n] * \delta[n] = x[n]$

Proof: $\sum_r \delta[r]x[n-r] \overset{\text{(i)}}{=} x[n]$ (i) all terms zero except $r = 0$.

BIBO Stability

4: Linear Time Invariant Systems
LTI Systems
Convolution Properties
▷ BIBO Stability
Frequency Response
Causality +
Convolution Complexity
Circular Convolution
Frequency-domain convolution
Overlap Add
Overlap Save
Summary
MATLAB routines

BIBO Stability: Bounded Input, $x[n] \Rightarrow$ Bounded Output, $y[n]$

The following are equivalent:

(1) An LTI system is BIBO stable

(2) $h[n]$ is absolutely summable, i.e. $\sum_{n=-\infty}^{\infty} |h[n]| < \infty$

(3) $H(z)$ region of absolute convergence includes $|z| = 1$.

Proof $(1) \Rightarrow (2)$:

Define $x[n] = \begin{cases} 1 & h[-n] \geq 0 \\ -1 & h[-n] < 0 \end{cases}$

then $y[0] = \sum x[0-n]h[n] = \sum |h[n]|$.

But $|x[n]| \leq 1 \forall n$ so BIBO $\Rightarrow y[0] = \sum |h[n]| < \infty$.

Proof $(2) \Rightarrow (1)$:

Suppose $\sum |h[n]| = S < \infty$ and $|x[n]| \leq B$ is bounded.

Then $|y[n]| = \left|\sum_{r=-\infty}^{\infty} x[n-r]h[r]\right|$

$\leq \sum_{r=-\infty}^{\infty} |x[n-r]|\,|h[r]|$

$\leq B \sum_{r=-\infty}^{\infty} |h[r]| \leq BS < \infty$

Frequency Response

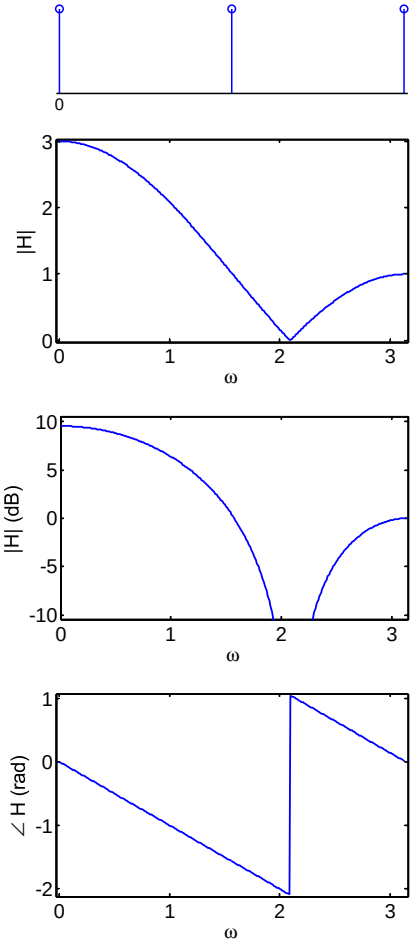
4: Linear Time Invariant Systems
LTI Systems
Convolution Properties
BIBO Stability
Frequency Response
Causality
Convolution Complexity
Circular Convolution
Frequency-domain convolution
Overlap Add
Overlap Save
Summary
MATLAB routines

For a BIBO stable system $Y(e^{j\omega}) = X(e^{j\omega})H(e^{j\omega})$
where $H(e^{j\omega})$is the DTFT of $h[n]$ i.e. $H(z)$ evaluated at $z = e^{j\omega}$.

Example: $h[n] = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$

$$H(e^{j\omega}) = 1 + e^{-j\omega} + e^{-j2\omega}$$
$$= e^{-j\omega}\left(1 + 2\cos\omega\right)$$

$$\left|H(e^{j\omega})\right| = \left|1 + 2\cos\omega\right|$$
$$\angle H(e^{j\omega}) = -\omega + \pi\frac{1-\mathrm{sgn}(1+2\cos\omega)}{2}$$



Sign change in $(1 + 2\cos\omega)$ at $\omega = 2.1$ gives
(a) gradient discontinuity in $|H(e^{j\omega})|$
(b) an abrupt phase change of $\pm\pi$.

Group delay is $-\frac{d}{d\omega}\angle H(e^{j\omega})$: gives delay of the modulation envelope at each ω.
Normally varies with ω but for a symmetric filter it is constant: in this case $+1$ samples.
Discontinuities of $\pm k\pi$ do not affect group delay.

Causality

4: Linear Time Invariant Systems
LTI Systems
Convolution Properties
BIBO Stability
Frequency Response
▷ Causality +
Convolution Complexity
Circular Convolution
Frequency-domain convolution
Overlap Add
Overlap Save
Summary
MATLAB routines

Causal System: cannot see into the future

i.e. output at time n depends only on inputs up to time n.

Formal definition:

If $v[n] = x[n]$ for $n \leq n_0$ then $\mathscr{H}(v[n]) = \mathscr{H}(x[n])$ for $n \leq n_0$.

The following are equivalent:

(1) An LTI system is causal

(2) $h[n]$ is causal $\Leftrightarrow h[n] = 0$ for $n < 0$

(3) $H(z)$ converges for $z = \infty$

Any right-sided sequence can be made causal by adding a delay.
All the systems we will deal with are causal.

Conditions on $h[n]$ and $H(z)$

Summary of conditions on $h[n]$ for LTI systems:

Causal	$\Leftrightarrow$	$h[n] = 0$ for $n < 0$
BIBO Stable	$\Leftrightarrow$	$\sum_{n=-\infty}^{\infty} \lvert h[n] \rvert < \infty$

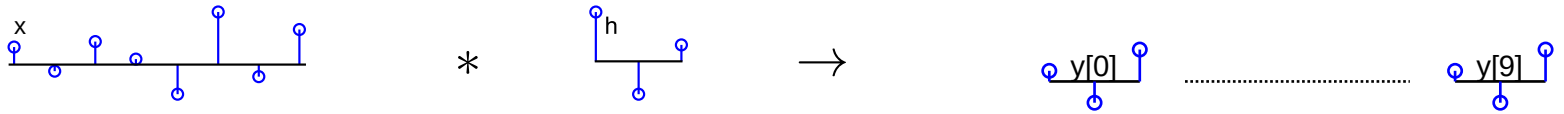
Summary of conditions on $H(z)$ for LTI systems:

Causal	$\Leftrightarrow$	$H(\infty)$ converges
BIBO Stable	$\Leftrightarrow$	$H(z)$ converges for $\lvert z \rvert = 1$
Passive	$\Leftrightarrow$	$\lvert H(z) \rvert \leq 1$ for $\lvert z \rvert = 1$
Lossless or Allpass	$\Leftrightarrow$	$\lvert H(z) \rvert = 1$ for $\lvert z \rvert = 1$

Convolution Complexity

4: Linear Time Invariant Systems
LTI Systems
Convolution Properties
BIBO Stability
Frequency Response
Causality
Convolution Complexity
Circular Convolution
Frequency-domain convolution
Overlap Add
Overlap Save
Summary
MATLAB routines

$y[n] = x[n] * h[n]$: convolve $x[0 : N-1]$ with $h[0 : M-1]$



Convolution sum:

$$y[n] = \sum_{r=0}^{M-1} h[r]x[n-r]$$

$y[n]$ is only non-zero in the range

$$0 \le n \le M + N - 2$$

Thus $y[n]$ has only

$M + N - 1$ non-zero values

$N = 8,\ M = 3$

$M + N - 1 = 10$

Algebraically:

$$x[n-r] \neq 0 \Rightarrow 0 \le n - r \le N - 1$$
$$\Rightarrow n + 1 - N \le r \le n$$

Hence: $y[n] = \sum_{r=\max(0,n+1-N)}^{\min(M-1,n))} h[r]x[n-r]$

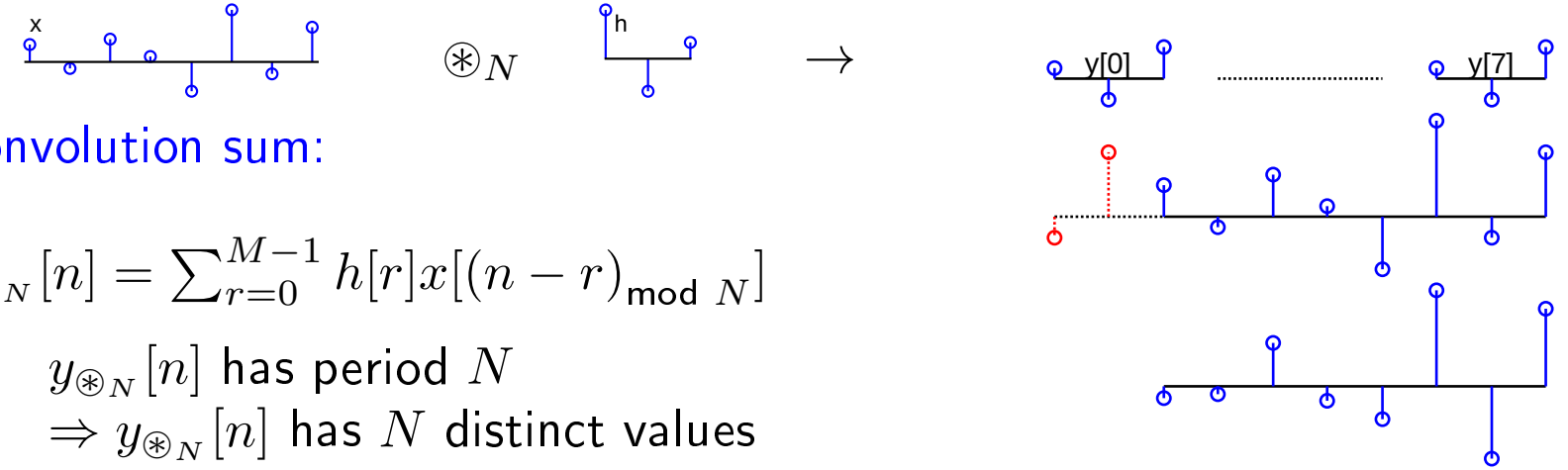
We must multiply each $h[n]$ by each $x[n]$ and add them to a total

$\Rightarrow$ total arithmetic complexity ($\times$ or $+$ operations) $\approx 2MN$

Circular Convolution

4: Linear Time Invariant Systems
LTI Systems
Convolution Properties
BIBO Stability
Frequency Response
Causality +
Convolution Complexity
▷ Circular Convolution
Frequency-domain convolution
Overlap Add
Overlap Save
Summary
MATLAB routines

$y_{\circledast}[n] = x[n] \circledast_N h[n]$: circ convolve $x[0:N-1]$ with $h[0:M-1]$



Convolution sum:

$$y_{\circledast_N}[n] = \sum_{r=0}^{M-1} h[r]x[(n-r)_{\text{mod } N}]$$

$y_{\circledast_N}[n]$ has period N
$\Rightarrow y_{\circledast_N}[n]$ has N distinct values

$N = 8,\ M = 3$

- Only the first $M-1$ values are affected by the circular repetition:
  $y_{\circledast_N}[n] = y[n]$ for $M-1 \le n \le N-1$
- If we append $M-1$ zeros (or more) onto $x[n]$, then the circular repetition has no effect at all and:
  $y_{\circledast_{N+M-1}}[n] = y[n]$ for $0 \le n \le N+M-2$

Circular convolution is a necessary evil in exchange for using the DFT

Frequency-domain convolution

4: Linear Time Invariant Systems
LTI Systems
Convolution Properties
BIBO Stability
Frequency Response
Causality +
Convolution Complexity
Circular Convolution
▷ Frequency-domain convolution
Overlap Add
Overlap Save
Summary
MATLAB routines

Idea: Use DFT to perform circular convolution - less computation

(1) Choose $L \geq M + N - 1$ (normally round up to a power of 2)

(2) Zero pad $x[n]$ and $h[n]$ to give sequences of length L: $\tilde{x}[n]$ and $\tilde{h}[n]$

(3) Use DFT: $\tilde{y}[n] = \mathcal{F}^{-1}(\tilde{X}[k]\tilde{H}[k]) = \tilde{x}[n] \circledast_L \tilde{h}[n]$

(4) $y[n] = \tilde{y}[n]$ for $0 \leq n \leq M + N - 2$.

Arithmetic Complexity:

DFT or IDFT take $4L \log_2 L$ operations if L is a power of 2
(or $16L \log_2 L$ if not).

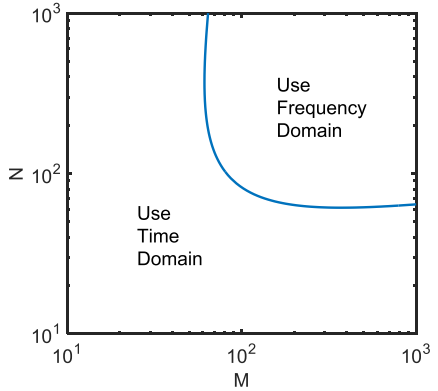
Total operations: $\approx 12L \log_2 L \approx 12\,(M + N) \log_2 (M + N)$

Beneficial if both M and N are $>\sim 70$.

Example: $M = 10^3$, $N = 10^4$:

Direct: $2MN = 2 \times 10^7$

with DFT: $= 1.8 \times 10^6$ ☺



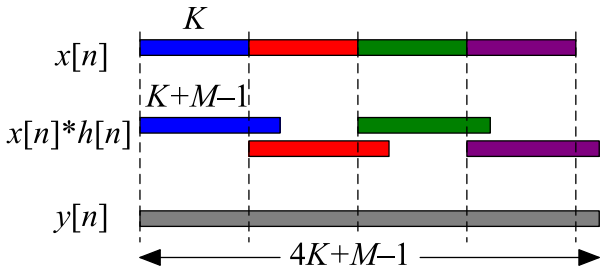
But: (a) DFT may be very long if N is large

(b) No outputs until all $x[n]$ has been input.

Overlap Add

4: Linear Time Invariant Systems
LTI Systems
Convolution Properties
BIBO Stability
Frequency Response
Causality +
Convolution Complexity
Circular Convolution
Frequency-domain convolution
▷ Overlap Add
Overlap Save
Summary
MATLAB routines

If N is very large:
(1) chop $x[n]$ into $\frac{N}{K}$ chunks of length K
(2) convolve each chunk with $h[n]$
(3) add up the results



Each output chunk is of length $K + M - 1$ and overlaps the next chunk

Operations: $\approx \frac{N}{K} \times 8\,(M+K)\log_2(M+K)$

Computational saving if $\approx 100 < M \ll K \ll N$

Example: $M = 500,\ K = 10^4,\ N = 10^7$

Direct: $2MN = 10^{10}$

single DFT: $12\,(M+N)\log_2(M+N) = 2.8 \times 10^9$

overlap-add: $\frac{N}{K} \times 8\,(M+K)\log_2(M+K) = 1.1 \times 10^9$ ☺

Other advantages:
(a) Shorter DFT
(b) Can cope with $N = \infty$
(c) Can calculate $y[0]$ as soon as $x[K-1]$ has been read:
algorithmic delay $= K - 1$ samples

Overlap Save

4: Linear Time Invariant Systems
LTI Systems
Convolution Properties
BIBO Stability
Frequency Response
Causality +
Convolution Complexity
Circular Convolution
Frequency-domain convolution
Overlap Add
▷ Overlap Save
Summary
MATLAB routines

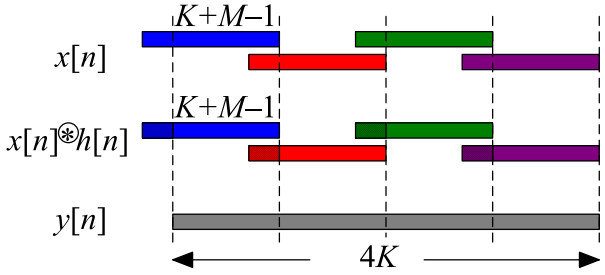
Alternative method:

(1) chop $x[n]$ into $\frac{N}{K}$ overlapping chunks of length $K+M-1$

(2) $\circledast_{K+M-1}$ each chunk with $h[n]$

(3) discard first $M-1$ from each chunk

(4) concatenate to make $y[n]$



The first $M-1$ points of each output chunk are invalid

Operations: slightly less than overlap-add because no addition needed to create $y[n]$

Advantages: same as overlap add

Strangely, rather less popular than overlap-add

Summary

4: Linear Time Invariant Systems
LTI Systems
Convolution Properties
BIBO Stability
Frequency Response
Causality +
Convolution Complexity
Circular Convolution
Frequency-domain convolution
Overlap Add
Overlap Save
▷ Summary
MATLAB routines

- LTI systems: impulse response, frequency response, group delay
- BIBO stable, Causal, Passive, Lossless systems
- Convolution and circular convolution properties
- Efficient methods for convolution
  - single DFT
  - overlap-add and overlap-save

For further details see Mitra: 4 & 5.

MATLAB routines

4: Linear Time Invariant Systems
LTI Systems
Convolution Properties
BIBO Stability
Frequency Response
Causality +
Convolution Complexity
Circular Convolution
Frequency-domain convolution
Overlap Add
Overlap Save
Summary
▷ MATLAB routines

fftfilt	Convolution using overlap add
$x[n] \circledast y[n]$	real(ifft(fft(x).*fft(y)))

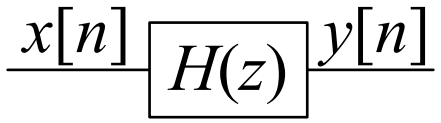
▷ 5: Filters
Difference Equations
FIR Filters
FIR Symmetries +
IIR Frequency Response
Negating z +
Cubing z +
Scaling z +
Low-pass filter +
Allpass filters +
Group Delay +
Minimum Phase +
Linear Phase Filters
Summary
MATLAB routines

5: Filters

Difference Equations

5: Filters
Difference Equations
FIR Filters
FIR Symmetries +
IIR Frequency Response
Negating z +
Cubing z +
Scaling z +
Low-pass filter +
Allpass filters +
Group Delay +
Minimum Phase +
Linear Phase Filters
Summary
MATLAB routines

Most useful LTI systems can be described by a difference equation:



$$y[n] = \sum_{r=0}^{M} b[r]x[n-r] - \sum_{r=1}^{N} a[r]y[n-r]$$

$$\Leftrightarrow \sum_{r=0}^{N} a[r]y[n-r] = \sum_{r=0}^{M} b[r]x[n-r] \qquad \text{with } a[0] = 1$$

$$\Leftrightarrow a[n] * y[n] = b[n] * x[n]$$

$$\Leftrightarrow Y(z) = \frac{B(z)}{A(z)} X(z)$$

$$\Leftrightarrow Y(e^{j\omega}) = \frac{B(e^{j\omega})}{A(e^{j\omega})} X(e^{j\omega})$$

(1) Always causal.
(2) Order of system is $\max(M, N)$, the highest r with $a[r] \neq 0$ or $b[r] \neq 0$.
(3) We assume that $a[0] = 1$; if not, divide $A(z)$ and $B(z)$ by $a[0]$.
(4) Filter is BIBO stable iff roots of $A(z)$ all lie within the unit circle.

Note negative sign in first equation.
Authors in some SP fields reverse the sign of the $a[n]$: BAD IDEA.

FIR Filters

5: Filters
Difference Equations
▷ FIR Filters
FIR Symmetries +
IIR Frequency Response
Negating z +
Cubing z +
Scaling z +
Low-pass filter +
Allpass filters +
Group Delay +
Minimum Phase +
Linear Phase Filters
Summary
MATLAB routines

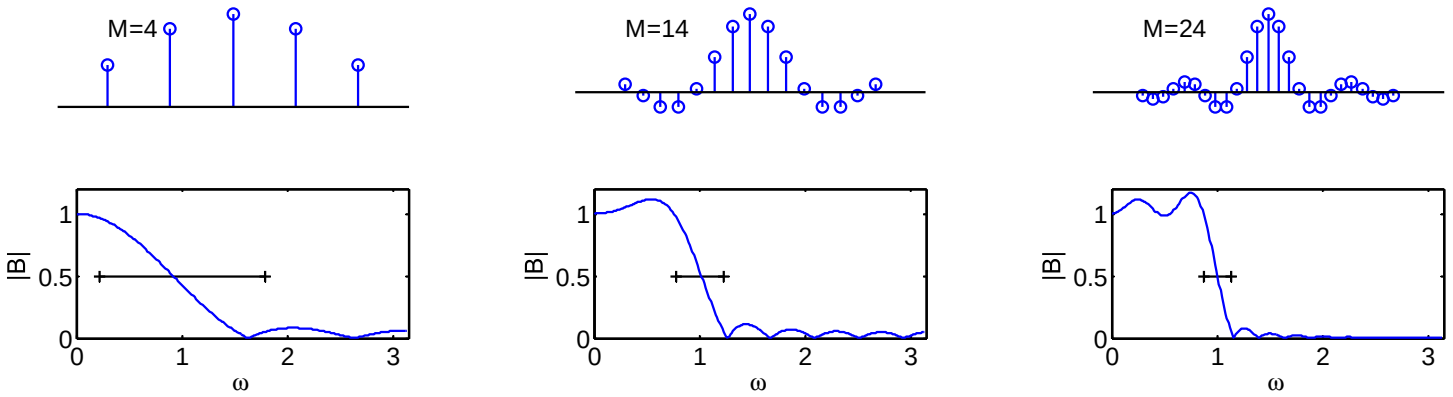
$A(z) = 1$: Finite Impulse Response (FIR) filter: $Y(z) = B(z)X(z)$.
Impulse response is $b[n]$ and is of length $M + 1$.

Frequency response is $B(e^{j\omega})$ and is the DTFT of $b[n]$.
Comprises M complex sinusoids + const:

$$b[0] + b[1]e^{-j\omega} + \cdots + b[M]e^{-jM\omega}$$

Small $M \Rightarrow$response contains only low "quefrencies"

Symmetrical $b[n] \Rightarrow H(e^{j\omega})e^{\frac{jM\omega}{2}}$ consists of $\frac{M}{2}$ cosine waves [+ const]



Rule of thumb: Fastest possible transition $\Delta\omega \geq \frac{2\pi}{M}$ (marked line)

5: Filters
Difference Equations
FIR Filters
▷ FIR Symmetries +
IIR Frequency Response
Negating z +
Cubing z +
Scaling z +
Low-pass filter +
Allpass filters +
Group Delay +
Minimum Phase +
Linear Phase Filters
Summary
MATLAB routines

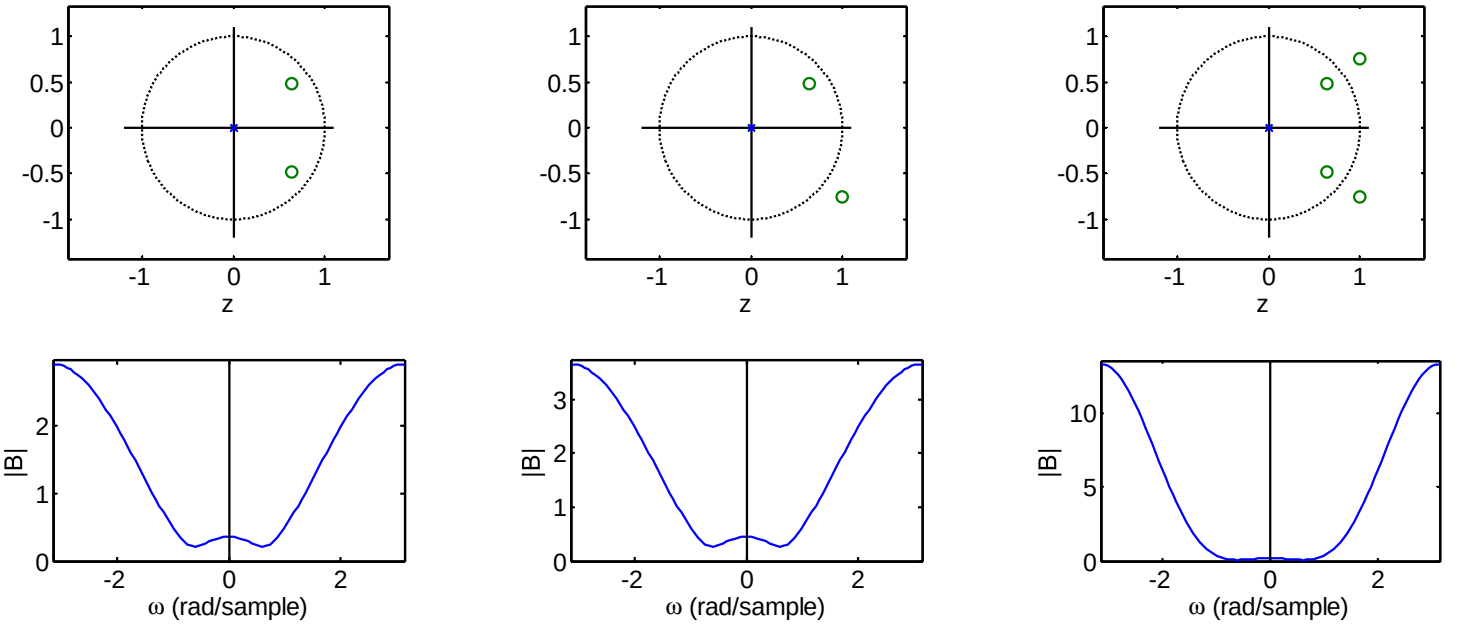
$B(e^{j\omega})$ is determined by the zeros of $z^M B(z) = \sum_{r=0}^{M} b[M-r]z^r$

Real $b[n]$ $\Rightarrow$ conjugate zero pairs: $z \Rightarrow z^*$

Symmetric: $b[n] = b[M-n]$ $\Rightarrow$ reciprocal zero pairs: $z \Rightarrow z^{-1}$

Real + Symmetric $b[n]$ $\Rightarrow$ conjugate+reciprocal groups of four or else pairs on the real axis

Real:	Symmetric:	Real + Symmetric:
$[1,\ -1.28,\ 0.64]$	$[1,\ -1.64+0.27j,\ 1]$	$[1, -3.28,\ 4.7625,\ -3.28,\ 1]$



[FIR Symmetry Proofs]

In all of the proofs below, we assume that $z = z_0$ is a root of $B(z)$ so that $B(z_0) = \sum_{r=0}^{M} b[r] z_0^{-r} = 0$ and then we prove that this implies that other values of z also satisfy $B(z) = 0$.

(1) Real $b[n]$

$$
\begin{aligned}
B(z_0^*) &= \textstyle\sum_{r=0}^{M} b[r] \left(z_0^*\right)^{-r} \\
&= \textstyle\sum_{r=0}^{M} b^*[r] \left(z_0^*\right)^{-r} && \text{since } b[r] \text{ is real} \\
&= \left(\textstyle\sum_{r=0}^{M} b[r] z_0^{-r}\right)^* && \text{take complex conjugate} \\
&= 0^* = 0 && \text{since } B(z_0) = 0
\end{aligned}
$$

(2) Symmetric: $b[n] = b[M - n]$

$$
\begin{aligned}
B(z_0^{-1}) &= \textstyle\sum_{r=0}^{M} b[r] z_0^{r} \\
&= \textstyle\sum_{n=0}^{M} b[M-n] z_0^{M-n} && \text{substitute } r = M - n \\
&= z_0^M \textstyle\sum_{n=0}^{M} b[M-n] z_0^{-n} && \text{take out } z_0^M \text{ factor} \\
&= z_0^M \textstyle\sum_{n=0}^{M} b[n] z_0^{-n} && \text{since } b[M-n] = b[n] \\
&= z_0^M \times 0 = 0 && \text{since } B(z_0) = 0
\end{aligned}
$$

IIR Frequency Response

5: Filters
Difference Equations
FIR Filters
FIR Symmetries +
IIR Frequency Response
Negating z +
Cubing z +
Scaling z +
Low-pass filter +
Allpass filters +
Group Delay +
Minimum Phase +
Linear Phase Filters
Summary
MATLAB routines

Factorize $H(z) = \frac{B(z)}{A(z)} = \frac{b[0]\prod_{i=1}^{M}(1-q_i z^{-1})}{\prod_{i=1}^{N}(1-p_i z^{-1})}$

Roots of $A(z)$ and $B(z)$ are the "poles" $\{p_i\}$ and "zeros" $\{q_i\}$ of $H(z)$
Also an additional $N - M$ zeros at the origin (affect phase only)

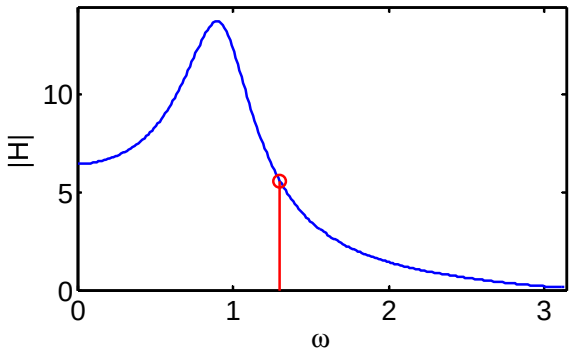
$$\left|H(e^{j\omega})\right| = \frac{|b[0]|\left|z^{-M}\right|\prod_{i=1}^{M}|z-q_i|}{|z^{-N}|\prod_{i=1}^{N}|z-p_i|} \text{ for } z = e^{j\omega}$$

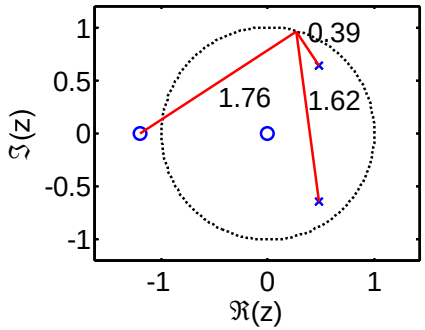
Example:

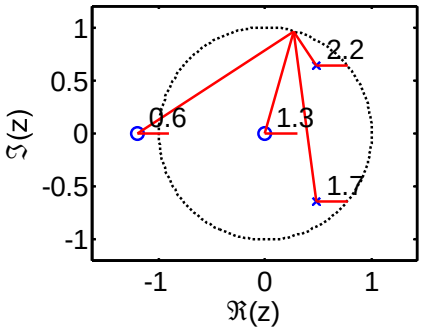
$$H(z) = \frac{2+2.4z^{-1}}{1-0.96z^{-1}+0.64z^{-2}} = \frac{2\left(1+1.2z^{-1}\right)}{(1-(0.48-0.64j)z^{-1})(1-(0.48+0.64j)z^{-1})}$$

At $\omega = 1.3$: $\left|H(e^{j\omega})\right| = \frac{2\times 1.76}{1.62\times 0.39} = 5.6$

$$\angle H(e^{j\omega}) = (0.6 + 1.3) - (1.7 + 2.2) = -1.97$$







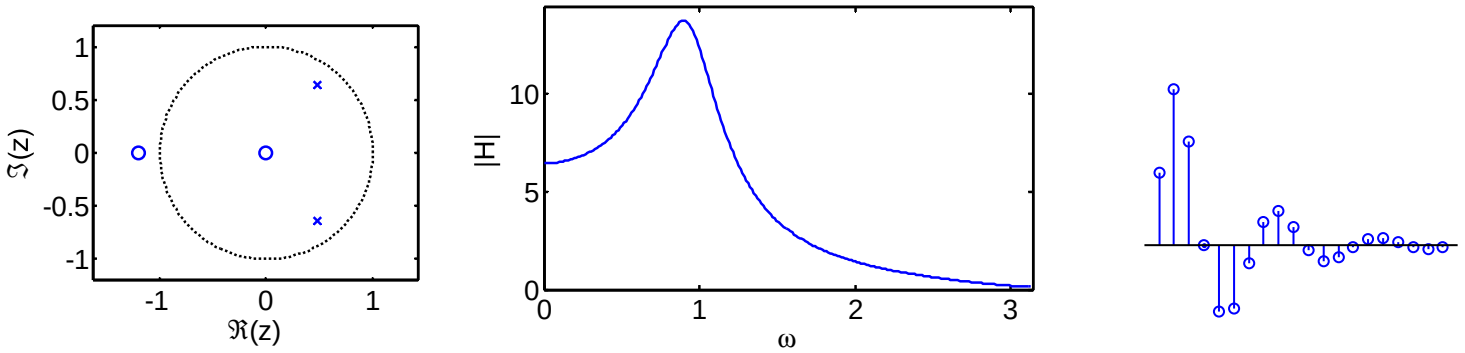
Negating z

5: Filters
Difference Equations
FIR Filters
FIR Symmetries +
IIR Frequency Response
▷ Negating z +
Cubing z +
Scaling z +
Low-pass filter +
Allpass filters +
Group Delay +
Minimum Phase +
Linear Phase Filters
Summary
MATLAB routines

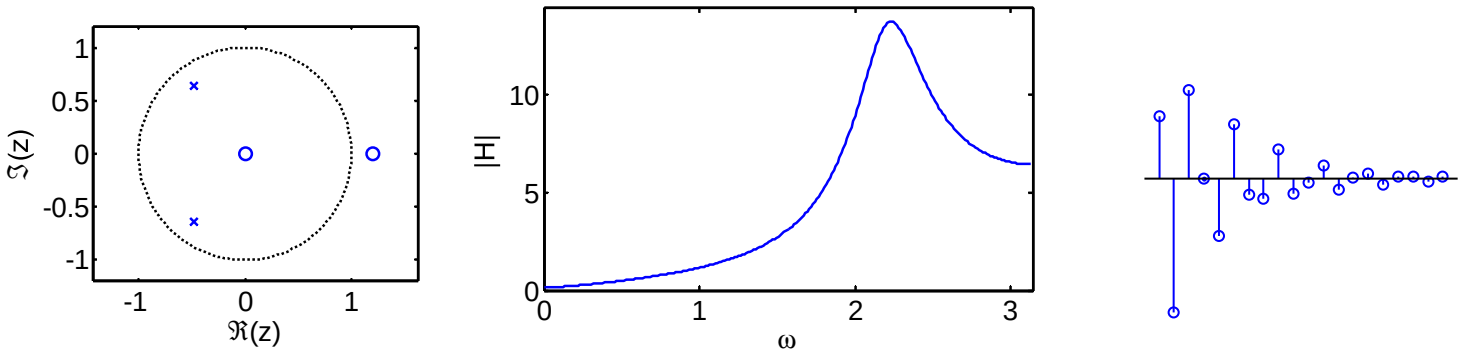
Given a filter $H(z)$ we can form a new one $H_R(z) = H(-z)$

Negate all odd powers of z, i.e. negate alternate $a[n]$ and $b[n]$

Example: $H(z) = \frac{2+2.4z^{-1}}{1-0.96z^{-1}+0.64z^{-2}}$



Negate z: $H_R(z) = \frac{2-2.4z^{-1}}{1+0.96z^{-1}+0.64z^{-2}}$ Negate odd coefficients



Pole and zero positions are negated, response is flipped and conjugated.

[Negating z]

Suppose that $H_R(z) = H(-z)$. Then $H_R(z)$ has the following two properties:

Pole and zero positions are negated

If z_0 is a zero of $H(z)$, then $H_R(-z_0) = H(z_0) = 0$ so $-z_0$ is a zero of $H_R(z)$. A similar argumnet applies to poles.

The frequency response is flipped and conjugated

The frequency response is given by $H_R(e^{j\omega}) = H(-e^{j\omega}) = H(e^{-j\pi} \times e^{j\omega}) = H(e^{j(\omega-\pi)})$. This corresponds to shifting the frequency reponse by $\pi\,\mathrm{rad/samp}$ (or, equivalently by $-\pi\,\mathrm{rad/samp}$).

If it is true that all the coefficients in $a[n]$ and $b[n]$ are real-valued (normally the case), then the response of $H(z)$ has conjugate symmetry, i.e. $H(e^{-j\omega}) = H^*(e^{j\omega})$. In this case we can write $H_R(e^{j\omega}) = H(e^{j(\omega-\pi)}) = H^*(e^{j(\pi-\omega)})$. This corresponds to a frequency response that has been reflected around $\omega = \frac{\pi}{2}$ (a.k.a. "flipped") and then conjugated.

So, the transformation of the frequency can be viewed in one of two ways: (a) it has been shifted by $\pm\pi\,\mathrm{rad/samp}$ or (b) it has been flipped around $\omega = \frac{\pi}{2}$ and then conjugated. The first interpretation is always true (even for filters with complex-valued coefficients) while the second interpretation is more intuitive but is only true if the filter coefficients are real-valued.

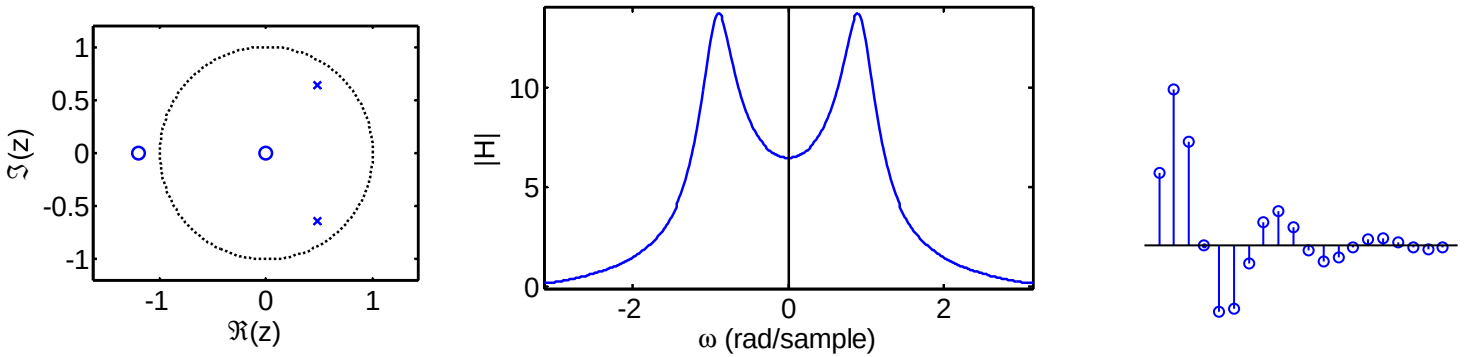
Cubing z

5: Filters
Difference Equations
FIR Filters
FIR Symmetries +
IIR Frequency Response
Negating z +
▷ Cubing z +
Scaling z +
Low-pass filter +
Allpass filters +
Group Delay +
Minimum Phase +
Linear Phase Filters
Summary
MATLAB routines

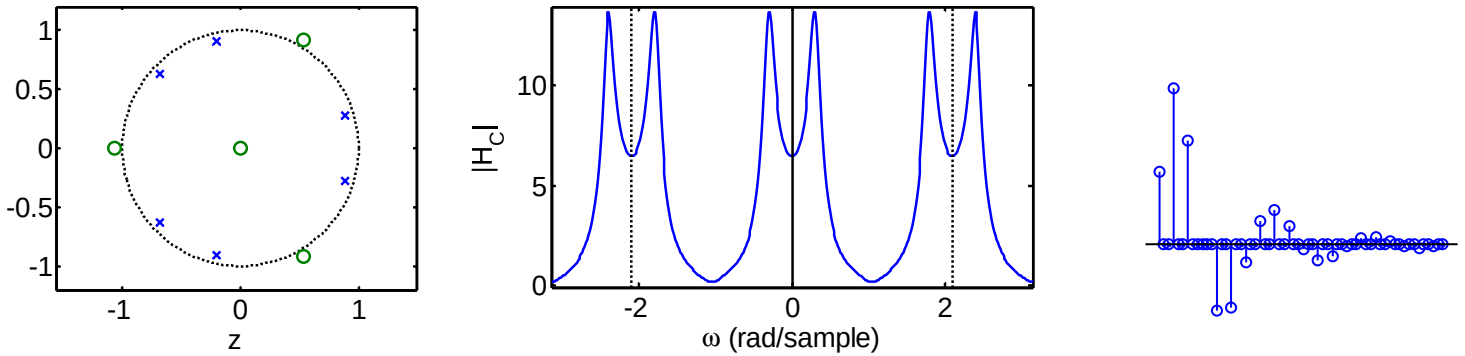
Given a filter $H(z)$ we can form a new one $H_C(z) = H(z^3)$

Insert two zeros between each $a[n]$ and $b[n]$ term

Example: $H(z) = \frac{2+2.4z^{-1}}{1-0.96z^{-1}+0.64z^{-2}}$



Cube z: $H_C(z) = \frac{2+2.4z^{-3}}{1-0.96z^{-3}+0.64z^{-6}}$ Insert 2 zeros between coefs



Pole and zero positions are replicated, magnitude response replicated.

[Cubing z]

Suppose that $H_C(z) = H(z^3)$. Then $H_C(z)$ has the following two properties:

Pole and zero positions are replicated three times

If z_0 is a zero of $H(z)$, then $H_C(\sqrt[3]{z_0}) = H(z_0) = 0$ so any cube root of z_0 is a zero of $H_C(z)$. A similar argument applies to poles. Any z_0 has three cube roots in the complex plane whose magnitudes all have the same value of $\sqrt[3]{|z_0|}$ and whose arguments are $\angle z_0 + \left\{0, \frac{2\pi}{3}, \frac{4\pi}{3}\right\}$.

The frequency response is replicated three times

The frequency response is given by $H_C(e^{j\omega}) = H(e^{j3\omega})$. This corresponds to shrinking the response horizontally by a factor of 3. Also $H_C\left(e^{j(\omega \pm \frac{2\pi}{3})}\right) = H\left(e^{j3(\omega \pm \frac{2\pi}{3})}\right) = H\left(e^{j3\omega \pm 2\pi}\right) = H_C\left(e^{j\omega}\right)$ meaning that there are three replications of the frequency response spaced $\frac{2\pi}{3}$ apart. Note that if you only look at the positive frequencies, there are three replications of the positive half of the reponse but alternate copies are flipped and conjugated (assuming the coefficients $a[n]$ and $b[n]$ are real-valued).

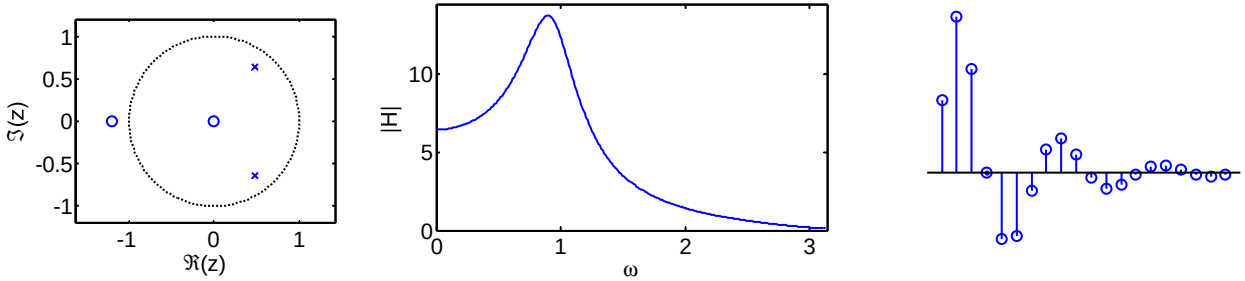
All of this carries over to raising z to any positive integer power; the number of replications is equal to the power concerned.

5: Filters
Difference Equations
FIR Filters
FIR Symmetries +
IIR Frequency Response
Negating z +
Cubing z +
▷ Scaling z +
Low-pass filter +
Allpass filters +
Group Delay +
Minimum Phase +
Linear Phase Filters
Summary
MATLAB routines

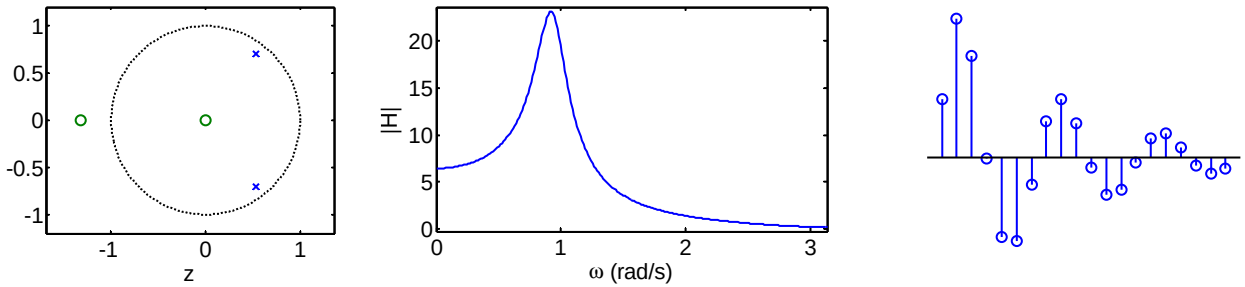
Given a filter $H(z)$ we can form a new one $H_S(z) = H(\frac{z}{\alpha})$

Multiply $a[n]$ and $b[n]$ by α^n

Example: $H(z) = \frac{2+2.4z^{-1}}{1-0.96z^{-1}+0.64z^{-2}}$



Scale z: $H_S(z) = H(\frac{z}{1.1}) = \frac{2+2.64z^{-1}}{1-1.056z^{-1}+0.7744z^{-2}}$



Pole and zero positions are multiplied by α, $\alpha > 1 \Rightarrow$peaks sharpened.

Pole at $z = p$ gives peak bandwidth $\approx 2\,|\log|p|| \approx 2\,(1 - |p|)$

For pole near unit circle, decrease bandwidth by $\approx 2 \log \alpha$

[Scaling z]

Suppose that $H_S(z) = H\left(\frac{z}{\alpha}\right)$ where α is a non-zero real number. Then $H_S(z)$ has the following two properties:

Pole and zero positions are multiplied by α

If z_0 is a zero of $H(z)$, then $H_S(\alpha z) = H(z_0) = 0$ so αz_0 is a zero of $H_S(z)$. The argument of the zero is unchanged since $\angle \alpha z_0 = \angle z_0$. The magnitude of the zero is multiplied by α. A similar argument applies to poles. If $\alpha > 1$ then the pole positions will move closer to the unit circle. If α is large enough to make any pole cross the unit circle then the filter $H_S(z)$ will be unstable.

The bandwidth of any peaks in the response are decreased by approximately $2 \log \alpha$

If $H(z)$ has a pole, p, that is near the unit circle, it results in a peak in the magnitude response at $\omega = \angle p$ whose amplitude is proportional to $\frac{1}{1-|p|}$ and whose bandwidth is approximately equal to $-2 \log |p| \approx 2\,(1 - |p|)$ (which is positive since $|p| < 1$). The corresponding pole in $H_S(z)$ is at αp, so its approximate bandwidth is now $-2 \log |\alpha p| = -2 \log |p| - 2 \log \alpha$. Thus the bandwidth has decreased by about $2 \log \alpha$.

If $\alpha > 1$ then $\log \alpha$ is positive and the peak in $H_S(z)$ will have a higher amplitude and a smaller bandwidth. If $\alpha < 1$, then $\log \alpha$ is negative and the peak will have a lower amplitude and a larger bandwidth.

5: Filters
Difference Equations
FIR Filters
FIR Symmetries +
IIR Frequency Response
Negating z +
Cubing z +
Scaling z +
▷ Low-pass filter +
Allpass filters +
Group Delay +
Minimum Phase +
Linear Phase Filters
Summary
MATLAB routines

1st order low pass filter: extremely common

$$y[n] = (1-p)x[n] + py[n-1] \Rightarrow H(z) = \frac{1-p}{1-pz^{-1}}$$

Impulse response:

$$h[n] = (1-p)p^n = (1-p)e^{-\frac{n}{\tau}}$$

where $\tau = \frac{1}{-\ln p}$ is the time constant in samples.

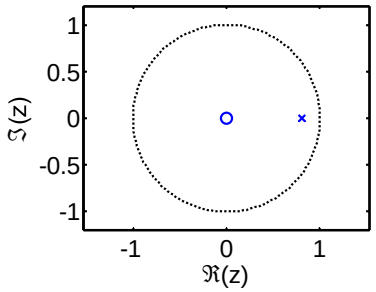
Magnitude response: $\left|H(e^{j\omega})\right| = \frac{1-p}{\sqrt{1-2p\cos\omega+p^2}}$

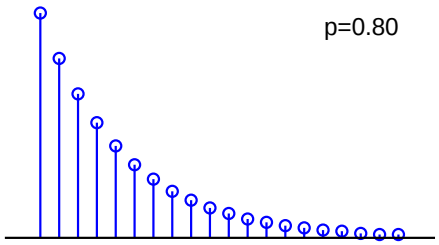
Low-pass filter with DC gain of unity.

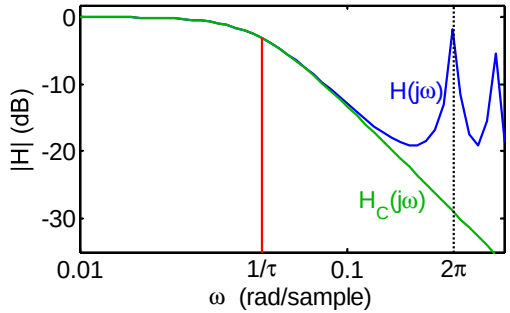
3 dB frequency is $\omega_{3dB} = \cos^{-1}\left(1 - \frac{(1-p)^2}{2p}\right) \approx 2\frac{1-p}{1+p} \approx \frac{1}{\tau}$

Compare continuous time: $H_C(j\omega) = \frac{1}{1+j\omega\tau}$

Indistinguishable for low ω but $H(e^{j\omega})$ is periodic, $H_C(j\omega)$ is not







[3 dB frequency approximation]

To find the 3dB frequency we require $|H(e^{j\omega_3})| = \sqrt{\frac{1}{2}} \Leftrightarrow |H(e^{j\omega_0})|^2 = \frac{1}{2}$.

$$\frac{(1-p)^2}{1-2p\cos\omega_3+p^2} = \frac{1}{2}$$

$$\Rightarrow 2(1-p)^2 = 1 - 2p\cos\omega_3 + p^2$$

$$\Rightarrow 2(1-p)^2 = (1-p)^2 + 2p(1-\cos\omega_3)$$

$$\Rightarrow \cos\omega_3 = 1 - \frac{(1-p)^2}{2p}$$

$$\Rightarrow \omega_3 = \cos^{-1}\left(1 - \frac{(1-p)^2}{2p}\right)$$

Expressing $\cos\omega = x$ as a Taylor series gives $x \approx 1 - \frac{\omega^2}{2} \Rightarrow \omega \approx \sqrt{2-2x}$. So replacing x by the expression in parentheses gives $\omega_3 \approx \sqrt{\frac{(1-p)^2}{p}} = \frac{1-p}{\sqrt{p}}$.

Writing $d = 1-p$ and assuming d is small, we can write $\sqrt{p} = (1-d)^{\frac{1}{2}} \approx 1 - \frac{1}{2}d = \frac{1}{2}(1+p)$. Substituting this into the previous expression gives $\omega_3 \approx 2\frac{1-p}{1+p}$.

5: Filters
Difference Equations
FIR Filters
FIR Symmetries +
IIR Frequency Response
Negating z +
Cubing z +
Scaling z +
Low-pass filter +
▷ Allpass filters +
Group Delay +
Minimum Phase +
Linear Phase Filters
Summary
MATLAB routines

If $H(z) = \frac{B(z)}{A(z)}$ with $b[n] = a^*[M-n]$ then we have an allpass filter:

$$\Rightarrow H(e^{j\omega}) = \frac{\sum_{r=0}^{M} a^*[M-r]e^{-j\omega r}}{\sum_{r=0}^{M} a[r]e^{-j\omega r}} = e^{-j\omega M}\frac{\sum_{s=0}^{M} a^*[s]e^{j\omega s}}{\sum_{r=0}^{M} a[r]e^{-j\omega r}} \qquad [s = M-r]$$

The two sums are complex conjugates $\Rightarrow$ they have the same magnitude

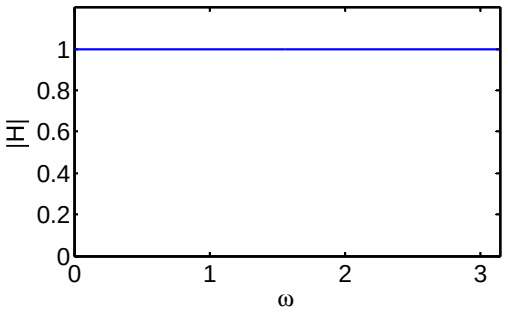
Hence $\left|H(e^{j\omega})\right| = 1 \forall\omega \Leftrightarrow$ "allpass"

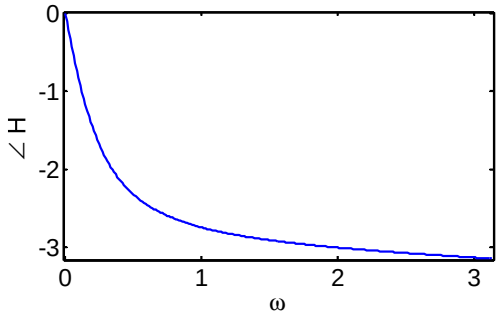
However phase is not constant: $\angle H(e^{j\omega}) = -\omega M - 2\angle A(e^{j\omega})$

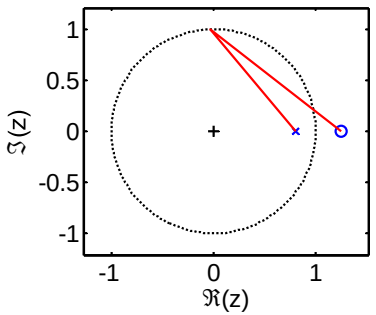
1st order allpass: $H(z) = \frac{-p+z^{-1}}{1-pz^{-1}} = -p\frac{1-p^{-1}z^{-1}}{1-pz^{-1}}$

Pole at p and zero at p^{-1}: "reflected in unit circle"

Constant distance ratio: $\left|e^{j\omega} - p\right| = |p|\left|e^{j\omega} - \frac{1}{p}\right| \forall\omega$







In an allpass filter, the zeros are the poles reflected in the unit circle.

[Allpass Filter Properties]

An allpass filter is one in which $H(z) = \frac{B(z)}{A(z)}$ with $b[n] = a^*[M-n]$. Of course, if the coefficients $a[n]$ are all real, then the conjugation has no effect and the numerator coefficients are identical to the numerator coefficients but in reverse order.

If $A(z)$ has order M, we can express the relation between $A(z)$ and $B(z)$ algebraically as $B(z) = z^{-M}\bar{A}(z^{-1})$ where the coefficients of $\bar{A}(z)$ are the conjugates of the coefficients of $A(z)$.

If the roots of $A(z)$ are p_i, then we can express $H(z)$ in factorized form as

$$H(z) = \prod_{i=1}^{M} \frac{-p_i^* + z^{-1}}{1 - p_i z^{-1}} = \prod_{i=1}^{M} \frac{1 - p_i^* z}{z - p_i}$$

We can therefore write

$$\begin{aligned} |H(z)|^2 &= \prod_{i=1}^{M} \frac{\left(1 - p_i^* z\right)\left(1 - p_i z^*\right)}{\left(z - p_i\right)\left(z^* - p_i^*\right)} = \prod_{i=1}^{M} \frac{1 - p_i z^* - p_i^* z + p_i p_i^* z z^*}{z z^* - p_i z^* - p_i^* z + p_i p_i^*} \\ &= \prod_{i=1}^{M} \left(1 + \frac{1 + p_i p_i^* z z^* - z z^* - p_i p_i^*}{z z^* - p_i z^* - p_i^* z + p_i p_i^*}\right) = \prod_{i=1}^{M} \left(1 + \frac{\left(1 - |z|^2\right)\left(1 - |p_i|^2\right)}{|z - p_i|^2}\right) \end{aligned}$$

If all the $|p_i| < 1$, then each term in the product is $\gtreqless 1$ according to whether $|z| \lesseqgtr 1$.

It follows that, provided $H(z)$ is stable, $|H(z)| \gtreqless 1$ according to whether $|z| \lesseqgtr 1$.

5: Filters
Difference Equations
FIR Filters
FIR Symmetries +
IIR Frequency Response
Negating z +
Cubing z +
Scaling z +
Low-pass filter +
Allpass filters +
▷ Group Delay +
Minimum Phase +
Linear Phase Filters
Summary
MATLAB routines

Group delay: $\tau_H(e^{j\omega}) = -\frac{d\angle H(e^{j\omega})}{d\omega}$ = delay of the modulation envelope.

Trick to get at phase: $\ln H(e^{j\omega}) = \ln\left|H(e^{j\omega})\right| + j\angle H(e^{j\omega})$

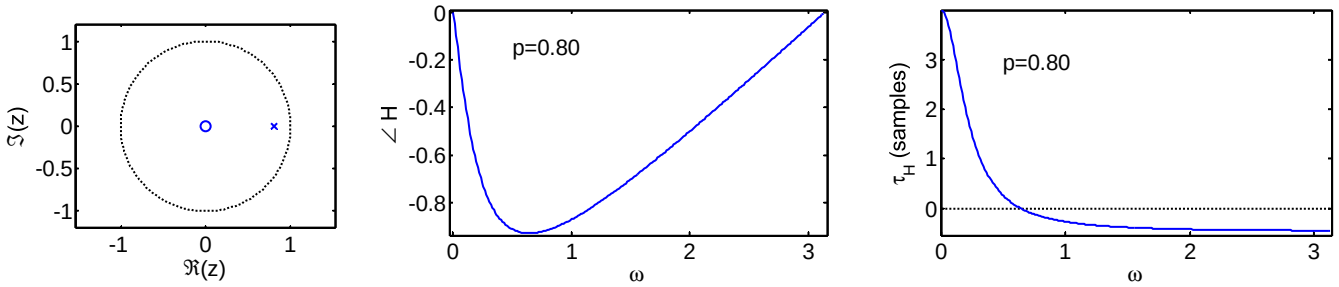
$$\tau_H = \frac{-d\left(\Im\left(\ln H(e^{j\omega})\right)\right)}{d\omega} = \Im\left(\frac{-1}{H(e^{j\omega})}\frac{dH(e^{j\omega})}{d\omega}\right) = \Re\left(\frac{-z}{H(z)}\frac{dH}{dz}\right)\Big|_{z=e^{j\omega}}$$

$$H(e^{j\omega}) = \sum_{n=0}^{\infty} h[n]e^{-jn\omega} = \mathscr{F}(h[n]) \qquad [\mathscr{F} = \text{DTFT}]$$

$$\frac{dH(e^{j\omega})}{d\omega} = \sum_{n=0}^{\infty} -jnh[n]e^{-jn\omega} = -j\mathscr{F}(nh[n])$$

$$\tau_H = \Im\left(\frac{-1}{H(e^{j\omega})}\frac{dH(e^{j\omega})}{d\omega}\right) = \Im\left(\frac{j\mathscr{F}(nh[n])}{\mathscr{F}(h[n])}\right) = \Re\left(\frac{\mathscr{F}(nh[n])}{\mathscr{F}(h[n])}\right)$$

Example: $H(z) = \frac{1}{1-pz^{-1}} \Rightarrow \tau_H = -\tau_{[1\ -p]} = -\Re\left(\frac{-pe^{-j\omega}}{1-pe^{-j\omega}}\right)$



Average group delay (over ω) = (# poles – # zeros) within the unit circle

Zeros on the unit circle count $-\frac{1}{2}$

[Group Delay Properties]

The group delay of a filter $H(z)$ at a frequency ω gives the time delay (in samples) of the envelope of a modulated sine wave at a frequency ω. It is defined as $\tau_H(e^{j\omega}) = -\frac{d\angle H(e^{j\omega})}{d\omega}$. For example, $H(z) = z^{-k}$ defines a filter that delays its input by k samples and we can calculate the group delay by evaluating

$$\tau_H(e^{j\omega}) = -\frac{d\angle H(e^{j\omega})}{d\omega} = -\frac{d}{d\omega}\left(\angle e^{-jk\omega}\right) = -\frac{d}{d\omega}\left(-k\omega\right) = k$$

which tells us that this filter has a constant group delay of k samples that is independent of ω.

The average value of τ_H equals the total change in $-\angle H(e^{j\omega})$ as ω goes from $-\pi$ to $+\pi$ divided by 2π. If you imagine an elastic string connecting a pole or zero to the point $z = e^{j\omega}$, you can see that as ω goes from $-\pi$ to $+\pi$ the string will wind once around the pole or zero if it is inside the unit circle but not if it is outside. Thus, the total change in $\angle H(e^{j\omega})$ is equal to 2π times the the difference between the number of poles and the number of zeros inside the unit circle. A zero that is exactly on the unit circle counts $\frac{1}{2}$ since there is a sudden discontinuity of π in $\angle H(e^{j\omega})$ as ω passes through the zero position.

When you multiply or divide complex numbers, their phases add or subtract, so it follows that when you multiply or divide transfer functions their group delays will add or subtract. Thus, for example, the group delay of an IIR filter, $H(z) = \frac{B(z)}{A(z)}$, is given by $\tau_H = \tau_B - \tau_A$. This means too that we can determine the group delay of a factorized transfer function by summing the group delays of the individual factors.

[Group Delay from $h[n]$ or $H(z)$]

The slide shows how to determine the group delay, τ_H, from either the impulse response, $h[n]$, or the transfer function, $H(z)$. We start by using a trick that is very common: if you want to get at the magnitude and phase of a complex number separately, you can do so by taking its natural log: $\ln\left(re^{j\theta}\right) = \ln|r| + j\theta$ or, in general, $\ln H = \ln|H| + j\angle H$. By rearranging this equation, we get $\angle H = \Im\left(\ln H\right)$ where $\Im(\,)$ denotes taking the imaginary part of a complex number. Using this, we can write

$$\tau_H = \frac{-d\left(\Im\left(\ln H(e^{j\omega})\right)\right)}{d\omega} = \Im\left(\frac{-d\left(\ln H(e^{j\omega})\right)}{d\omega}\right) = \Im\left(\frac{-1}{H(e^{j\omega})}\frac{dH(e^{j\omega})}{d\omega}\right). \tag{1}$$

By going back to the definition of the DTFT, we find that $H(e^{j\omega}) = \mathscr{F}(h[n])$ and $\frac{dH(e^{j\omega})}{d\omega} = -j\mathscr{F}(nh[n])$ where $\mathscr{F}(\,)$ denotes the DTFT. Substituting these expressions into the above equation gives us a formula for τ_H in terms of the impulse response $h[n]$.

$$\tau_H = \Re\left(\frac{\mathscr{F}(nh[n])}{\mathscr{F}(h[n])}\right) \tag{2}$$

In order to express τ_H in terms of z, we first note that if $z = e^{j\omega}$ then $\frac{dz}{d\omega} = jz$. By substituting $z = e^{j\omega}$ into equation (1), we get

$$\tau_H = \Im\left(\frac{-1}{H(z)}\frac{dH(z)}{d\omega}\right) = \Im\left(\frac{-1}{H(z)}\frac{dH(z)}{dz}\frac{dz}{d\omega}\right) = \Im\left(\frac{-jz}{H(z)}\frac{dH(z)}{dz}\right) = \Re\left(\frac{-z}{H(z)}\frac{dH(z)}{dz}\right)\Bigg|_{z=e^{j\omega}}.$$

[Group Delay Example]

As an example, suppose we want to determine the group delay of : $H(z) = \frac{1}{1-pz^{-1}}$. As noted above, if $H(z) = \frac{B(z)}{A(z)}$, then $\tau_H = \tau_B - \tau_A$. In this case $\tau_B = 0$ so $\tau_H = -\tau_{[1\ -p]}$.

Using equation (2) gives $\tau_H = -\Re\left(\frac{\mathscr{F}([0\ -p])}{\mathscr{F}([1\ -p])}\right)$ since $nh[n] = [0\ 1] \times [1\ -p]$.

Applying the definition of the DTFT, we get

$$\tau_H = -\Re\left(\frac{-pe^{-j\omega}}{1-pe^{-j\omega}}\right) = \Re\left(\frac{p}{e^{j\omega}-p}\right) = \frac{\Re\left(p\left(e^{-j\omega}-p\right)\right)}{\left(e^{j\omega}-p\right)\left(e^{-j\omega}-p\right)} = \frac{p\cos\omega - p^2}{1-2p\cos\omega+p^2}$$

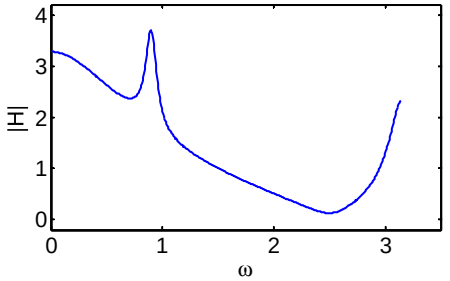
As demonstrated above, the average value of τ_H is zero for this filter because there is one pole and one zero inside the unit circle.

Minimum Phase

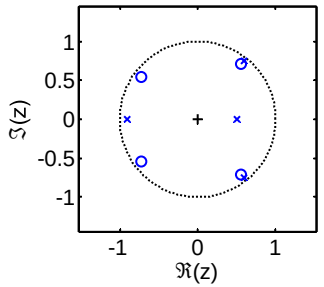
5: Filters
Difference Equations
FIR Filters
FIR Symmetries +
IIR Frequency Response
Negating z +
Cubing z +
Scaling z +
Low-pass filter +
Allpass filters +
Group Delay +
▷ Minimum Phase +
Linear Phase Filters
Summary
MATLAB routines

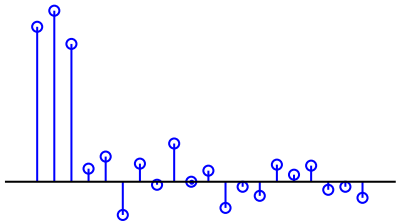
Average group delay (over ω) = (# poles – # zeros) within the unit circle

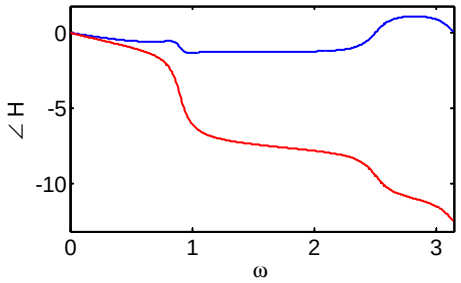
- zeros on the unit circle count $-\frac{1}{2}$

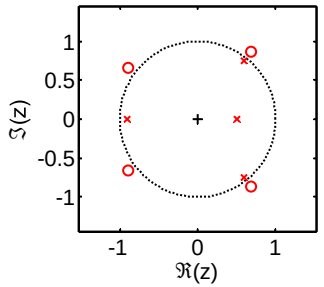


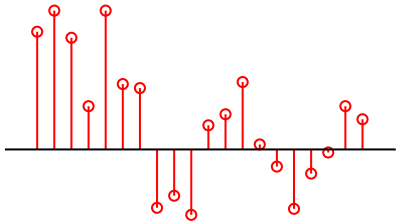
Reflecting an interior zero to the exterior multiplies $\left|H(e^{j\omega})\right|$ by a constant but increases average group delay by 1 sample.

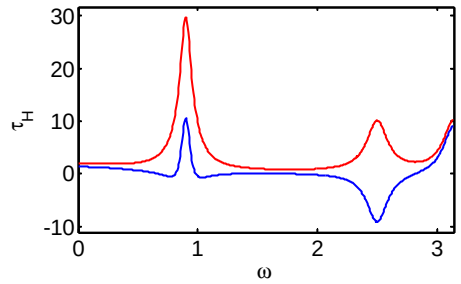












A filter with all zeros inside the unit circle is a minimum phase filter:

- Lowest possible group delay for a given magnitude response
- Energy in $h[n]$ is concentrated towards $n = 0$

[Energy Concentration Property]

This proof is not examinable

Suppose $H(z)$ has a zero inside the unit circle at $z = z_0$ so that we can write $H(z) = \left(1 - z_0 z^{-1}\right) F(z)$.
If we flip this zero outside the unit circle, we can write $G(z) = \left(z^{-1} - z_0^*\right) F(z)$ which has the same magnitude response as $H(z)$.
Taking inverse z-transforms, we can write the corresponding time domain equations:

$$h[n] = f[n] - z_0 f[n-1] \text{ and } g[n] = f[n-1] - z_0^* f[n].$$

Now, defining $f[-1] \triangleq 0$, we sum the energy in the first $K+1$ samples of the impulse response:

$$\begin{aligned}
\sum_{k=0}^{K} |h[k]|^2 &= \sum_{k=0}^{K} |f[k] - z_0 f[k-1]|^2 = \sum_{k=0}^{K} (f[k] - z_0 f[k-1]) (f[k] - z_0 f[k-1])^* \\
&= \sum_{k=0}^{K} |f[k]|^2 - z_0 f[k-1] f^*[k] - z_0^* f^*[k-1] f[k] + |z_0|^2 |f[k-1]|^2 \\
&= \sum_{k=0}^{K} |z_0|^2 |f[k]|^2 - z_0 f[k-1] f^*[k] - z_0^* f^*[k-1] f[k] + |f[k-1]|^2 \\
&\quad + \sum_{k=0}^{K} \left(1 - |z_0|^2\right) \left(|f[k]|^2 - |f[k-1]|^2\right)
\end{aligned}$$

[Energy Concentration Property (continued)]

So, repeating the previous line,

$$\begin{aligned}
\sum_{k=0}^{K} |h[k]|^2 &= \sum_{k=0}^{K} |z_0|^2 |f[k]|^2 - z_0 f[k-1] f^*[k] - z_0^* f^*[k-1] f[k] + |f[k-1]|^2 \\
&\quad + \sum_{k=0}^{K} \left(1 - |z_0|^2\right) \left(|f[k]|^2 - |f[k-1]|^2\right) \\
&= \sum_{k=0}^{K} \left(f[k-1] - z_0^* f[k]\right) \left(f[k-1] - z_0^* f[k]\right)^* + \left(1 - |z_0|^2\right) \sum_{k=0}^{K} \left(|f[k]|^2 - |f[k-1]|^2\right) \\
&= \sum_{k=0}^{K} |g[k]|^2 + \left(1 - |z_0|^2\right) \left(|f[K]|^2 - |f[-1]|^2\right) \\
&= \sum_{k=0}^{K} |g[k]|^2 + \left(1 - |z_0|^2\right) |f[K]|^2 \geq \sum_{k=0}^{K} |g[k]|^2
\end{aligned}$$

since $|z_0| < 1$ implies that $\left(1 - |z_0|^2\right) > 0$. Thus flipping a zero from inside the unit circle to outside never increases the energy in the first $K+1$ samples of the impulse response (for any K). Hence the minimum phase response is the one with the most energy in the first $K+1$ samples for any K.

Linear Phase Filters

5: Filters
Difference Equations
FIR Filters
FIR Symmetries +
IIR Frequency Response
Negating z +
Cubing z +
Scaling z +
Low-pass filter +
Allpass filters +
Group Delay +
Minimum Phase +
▷ Linear Phase Filters
Summary
MATLAB routines

The phase of a linear phase filter is: $\angle H(e^{j\omega}) = \theta_0 - \alpha\omega$

Equivalently constant group delay: $\tau_H = -\frac{d\angle H(e^{j\omega})}{d\omega} = \alpha$

A filter has linear phase iff $h[n]$ is symmetric or antisymmetric:

$$h[n] = h[M-n] \quad \forall n \text{ or else } h[n] = -h[M-n] \quad \forall n$$

M can be even ($\Rightarrow \exists$ mid point) or odd ($\Rightarrow \nexists$ mid point)

Proof $\Leftarrow$:

$$\begin{aligned} 2H(e^{j\omega}) &= \sum_0^M h[n]e^{-j\omega n} + \sum_0^M h[M-n]e^{-j\omega(M-n)} \\ &= e^{-j\omega\frac{M}{2}} \sum_0^M h[n]e^{-j\omega\left(n-\frac{M}{2}\right)} + h[M-n]e^{j\omega\left(n-\frac{M}{2}\right)} \end{aligned}$$

$h[n]$ symmetric:

$$2H(e^{j\omega}) = 2e^{-j\omega\frac{M}{2}} \sum_0^M h[n]\cos\left(n-\frac{M}{2}\right)\omega$$

$h[n]$ anti-symmetric:

$$\begin{aligned} 2H(e^{j\omega}) &= -2je^{-j\omega\frac{M}{2}} \sum_0^M h[n]\sin\left(n-\frac{M}{2}\right)\omega \\ &= 2e^{-j\left(\frac{\pi}{2}+\omega\frac{M}{2}\right)} \sum_0^M h[n]\sin\left(n-\frac{M}{2}\right)\omega \end{aligned}$$

Summary

5: Filters
Difference Equations
FIR Filters
FIR Symmetries +
IIR Frequency Response
Negating z +
Cubing z +
Scaling z +
Low-pass filter +
Allpass filters +
Group Delay +
Minimum Phase +
Linear Phase Filters
▷ Summary
MATLAB routines

- Useful filters have difference equations:
  - Freq response determined by pole/zero positions
  - $N - M$ zeros at origin (or $M - N$ poles)
  - Geometric construction of $|H(e^{j\omega})|$
    - Pole bandwidth $\approx 2\,|\log|p|| \approx 2\,(1 - |p|)$
  - Stable if poles have $|p| < 1$
- Allpass filter: $a[n] = b[M - n]$
  - Reflecting a zero in unit circle leaves $|H(e^{j\omega})|$ unchanged
- Group delay: $\tau_H\left(e^{j\omega}\right) = -\frac{d\angle H(e^{j\omega})}{d\omega}$ samples
  - Symmetrical $h[n] \Leftrightarrow \tau_H\left(e^{j\omega}\right) = \frac{M}{2} \forall \omega$
  - Average τ_H over ω = (# poles – # zeros) within the unit circle
- Minimum phase if zeros have $|q| \leq 1$
  - Lowest possible group delay for given $|H(e^{j\omega})|$
- Linear phase = Constant group Delay = symmetric/antisymmetric $h[n]$

For further details see Mitra: 6, 7.

MATLAB routines

5: Filters
Difference Equations
FIR Filters
FIR Symmetries +
IIR Frequency Response
Negating z +
Cubing z +
Scaling z +
Low-pass filter +
Allpass filters +
Group Delay +
Minimum Phase +
Linear Phase Filters
Summary
▷ MATLAB routines

filter	filter a signal
impz	Impulse response
residuez	partial fraction expansion
grpdelay	Group Delay
freqz	Calculate filter frequency response

6: Window Filter Design
Inverse DTFT
Rectangular window
Dirichlet Kernel
Window relationships
Common Windows
Order Estimation
Example Design
Frequency sampling
Summary
MATLAB routines

6: Window Filter Design

Inverse DTFT

6: Window Filter Design
▷ Inverse DTFT
Rectangular window
Dirichlet Kernel +
Window relationships
Common Windows
Order Estimation
Example Design
Frequency sampling
Summary
MATLAB routines

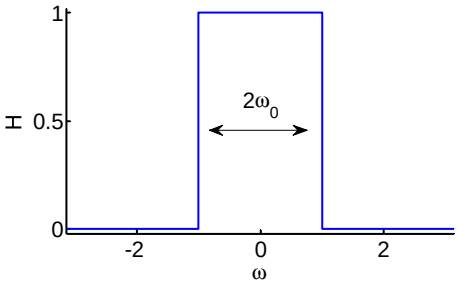
For any BIBO stable filter, $H(e^{j\omega})$ is the DTFT of $h[n]$

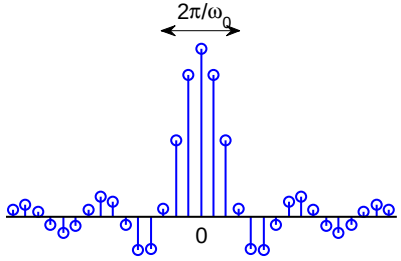
$$H(e^{j\omega}) = \sum_{-\infty}^{\infty} h[n]e^{-j\omega n} \quad \Leftrightarrow \quad h[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} H(e^{j\omega})e^{j\omega n}d\omega$$

If we know $H(e^{j\omega})$ exactly, the IDTFT gives the ideal $h[n]$

Example: Ideal Lowpass filter

$$H(e^{j\omega}) = \begin{cases} 1 & |\omega| \le \omega_0 \\ 0 & |\omega| > \omega_0 \end{cases} \quad \Leftrightarrow \quad h[n] = \frac{\sin \omega_0 n}{\pi n}$$





Note: Width in ω is $2\omega_0$, width in n is $\frac{2\pi}{\omega_0}$: product is 4π always
Sadly $h[n]$ is infinite and non-causal. Solution: multiply $h[n]$ by a window

Rectangular window

6: Window Filter Design
Inverse DTFT
▷ Rectangular window
Dirichlet Kernel +
Window relationships
Common Windows
Order Estimation
Example Design
Frequency sampling
Summary
MATLAB routines

Truncate to $\pm\frac{M}{2}$ to make finite; $h_1[n]$ is now of length $M+1$

MSE Optimality:

Define mean square error (MSE) in frequency domain

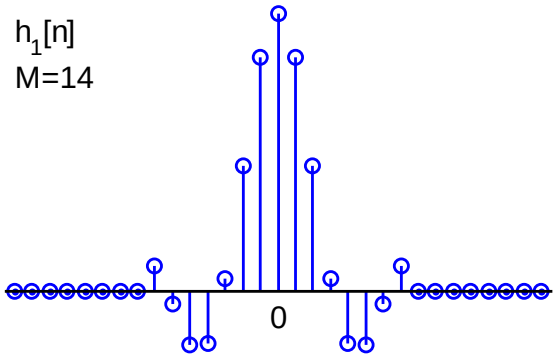
$$E = \frac{1}{2\pi}\int_{-\pi}^{\pi} \left|H(e^{j\omega}) - H_1(e^{j\omega})\right|^2 d\omega$$

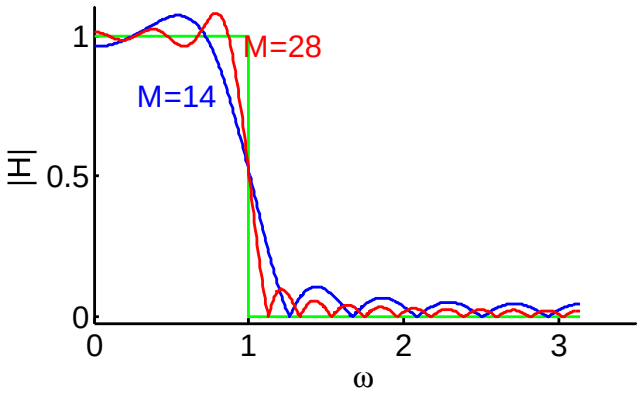
$$= \frac{1}{2\pi}\int_{-\pi}^{\pi} \left|H(e^{j\omega}) - \sum_{-\frac{M}{2}}^{\frac{M}{2}} h_1[n]e^{-j\omega n}\right|^2 d\omega$$

Minimum E is when $h_1[n] = h[n]$.

Proof: From Parseval: $E = \sum_{-\frac{M}{2}}^{\frac{M}{2}} |h[n] - h_1[n]|^2 + \sum_{|n|>\frac{M}{2}} |h[n]|^2$

However: 9% overshoot at a discontinuity even for large n.





Normal to delay by $\frac{M}{2}$ to make causal. Multiplies $H(e^{j\omega})$ by $e^{-j\frac{M}{2}\omega}$.

6: Window Filter Design
Inverse DTFT
Rectangular window
▷ Dirichlet Kernel +
Window relationships
Common Windows
Order Estimation
Example Design
Frequency sampling
Summary
MATLAB routines

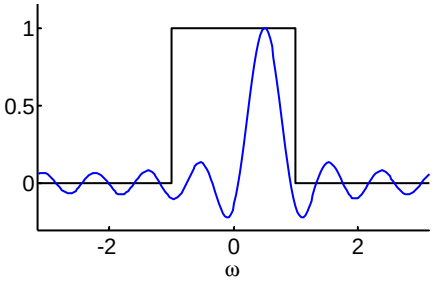
Truncation $\Leftrightarrow$ Multiply $h[n]$ by a rectangular window, $w[n] = \delta_{-\frac{M}{2} \le n \le \frac{M}{2}}$

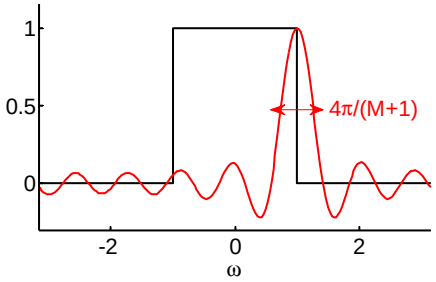
$\Leftrightarrow$ Circular Convolution $H_{M+1}(e^{j\omega}) = \frac{1}{2\pi} H(e^{j\omega}) \circledast W(e^{j\omega})$

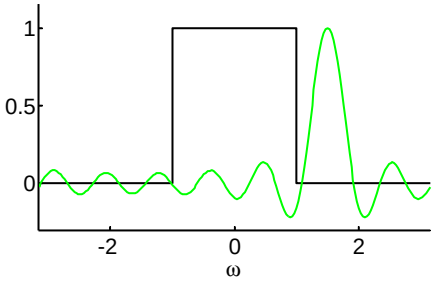
$$W(e^{j\omega}) = \sum_{-\frac{M}{2}}^{\frac{M}{2}} e^{-j\omega n} \overset{\text{(i)}}{=} 1 + 2\sum_{1}^{0.5M} \cos(n\omega) \overset{\text{(ii)}}{=} \frac{\sin 0.5(M+1)\omega}{\sin 0.5\omega}$$

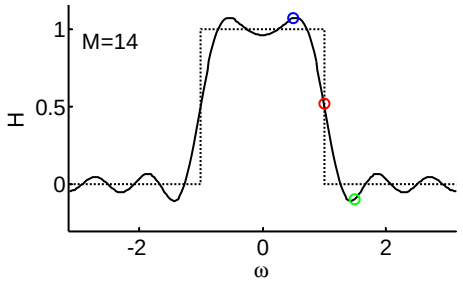
Proof: (i) $e^{-j\omega(-n)} + e^{-j\omega(+n)} = 2\cos(n\omega)$ (ii) Sum geom. progression

Effect: convolve ideal freq response with Dirichlet kernel (aliassed sinc)









Provided that $\frac{4\pi}{M+1} \ll 2\omega_0 \Leftrightarrow M+1 \gg \frac{2\pi}{\omega_0}$:

Passband ripple: $\Delta\omega \approx \frac{4\pi}{M+1}$, stopband $\frac{2\pi}{M+1}$

Transition pk-to-pk: $\Delta\omega \approx \frac{4\pi}{M+1}$

Transition Gradient: $\left.\frac{d|H|}{d\omega}\right|_{\omega=\omega_0} \approx \frac{M+1}{2\pi}$

[Dirichlet Kernel]

Other properties of $W(e^{j\omega})$:

The DTFT of a symmetric rectangular window of length $M+1$ is $W(e^{j\omega}) = \sum_{-\frac{M}{2}}^{\frac{M}{2}} e^{-j\omega n} = e^{j\omega\frac{M}{2}} \sum_{0}^{M} e^{-j\omega n} = e^{j\omega\frac{M}{2}} \frac{1-e^{-j\omega(M+1)}}{1-e^{-j\omega}} = \frac{e^{j0.5\omega(M+1)}-e^{-j0.5\omega(M+1)}}{e^{j0.5\omega}-e^{-j0.5\omega}} = \frac{\sin 0.5(M+1)\omega}{\sin 0.5\omega}$.

For small x we can approximate $\sin x \approx x$; the error is $< 1\%$ for $x < 0.25$. So, for $\omega < 0.5$, we have $W(e^{j\omega}) \approx 2\omega^{-1} \sin 0.5(M+1)\omega$.

The peak value is at $\omega = 0$ and equals $M+1$; this means that the peak gradient of $H_{M+1}(e^{j\omega})$ will be $\frac{M+1}{2\pi}$.

The minimum value of $W(e^{j\omega})$ is approximately equal to the minimuum of $2\omega^{-1} \sin 0.5(M+1)\omega$ which is when $\sin 0.5(M+1)\omega = -1$ i.e. when $\omega = \frac{1.5\pi}{0.5(M+1)} = \frac{3\pi}{M+1}$.
Hence $\min W(e^{j\omega}) \approx \min 2\omega^{-1} \sin 0.5(M+1)\omega = -\frac{M+1}{1.5\pi}$.

Passband and Stopband ripple:

The ripple in $W(e^{j\omega}) = \frac{\sin 0.5(+1)\omega}{\sin 0.5\omega}$ has a period of $\Delta\omega = \frac{2\pi}{0.5(+1)} = \frac{4\pi}{M+1}$ and this gives rise to ripple with this period in both the passband and stopband of $H_{M+1}(e^{j\omega})$.

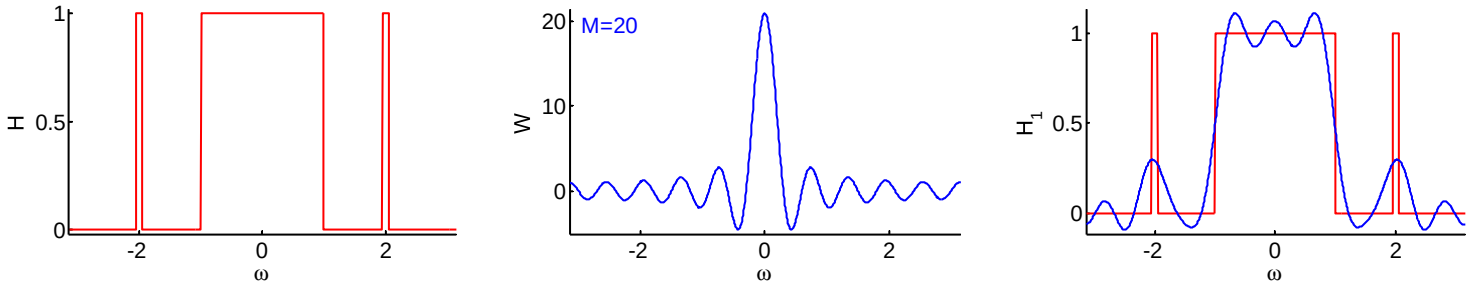
However the stopband ripple takes the value of $H_{M+1}(e^{j\omega})$ alternately positive and negative. If you plot the magnitude response, $\left|H_{M+1}(e^{j\omega})\right|$ then this ripple will be full-wave rectified and will double in frequency so its period will now be $\frac{2\pi}{M+1}$.

Window relationships

6: Window Filter Design
Inverse DTFT
Rectangular window
Dirichlet Kernel +
Window relationships
Common Windows
Order Estimation
Example Design
Frequency sampling
Summary
MATLAB routines

When you multiply an impulse response by a window $M+1$ long

$$H_{M+1}(e^{j\omega}) = \frac{1}{2\pi} H(e^{j\omega}) \circledast W(e^{j\omega})$$



(a) passband gain $\approx w[0]$; peak$\approx \frac{w[0]}{2} + \frac{0.5}{2\pi} \int_{\text{mainlobe}} W(e^{j\omega}) d\omega$

rectangular window: passband gain $= 1$; peak gain $= 1.09$

(b) transition bandwidth, $\Delta\omega$ = width of the main lobe

transition amplitude, ΔH = integral of main lobe$\div 2\pi$

rectangular window: $\Delta\omega = \frac{4\pi}{M+1}$, $\Delta H \approx 1.18$

(c) stopband gain is an integral over oscillating sidelobes of $W(e^{j\omega})$

rect window: $\left|\min H(e^{j\omega})\right| = 0.09 \ll \left|\min W(e^{j\omega})\right| = \frac{M+1}{1.5\pi}$

(d) features narrower than the main lobe will be broadened and attenuated

Common Windows

6: Window Filter Design
Inverse DTFT
Rectangular window
Dirichlet Kernel +
Window relationships
▷ Common Windows
Order Estimation
Example Design
Frequency sampling
Summary
MATLAB routines

Rectangular: $w[n] \equiv 1$

don't use

Hanning: $0.5 + 0.5c_1$

$c_k = \cos \frac{2\pi kn}{M+1}$

rapid sidelobe decay

Hamming: $0.54 + 0.46c_1$

best peak sidelobe

Blackman-Harris 3-term:

$$0.42 + 0.5c_1 + 0.08c_2$$

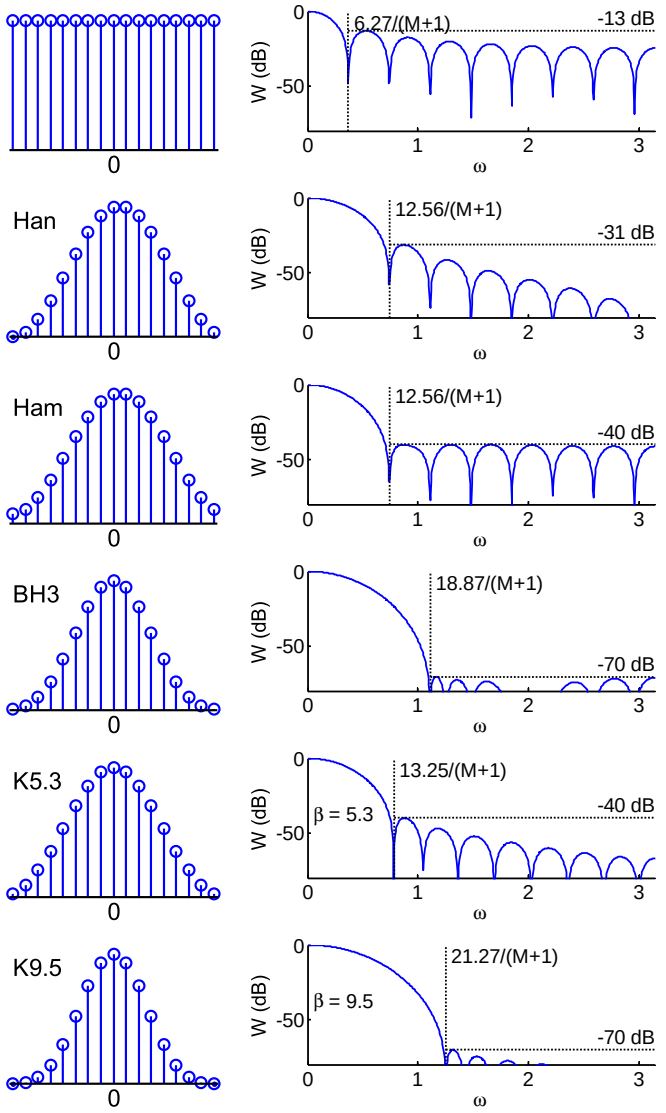
best peak sidelobe

Kaiser: $\frac{I_0\left(\beta\sqrt{1-\left(\frac{2n}{M}\right)^2}\right)}{I_0(\beta)}$

β controls width v sidelobes

Good compromise:

Width v sidelobe v decay

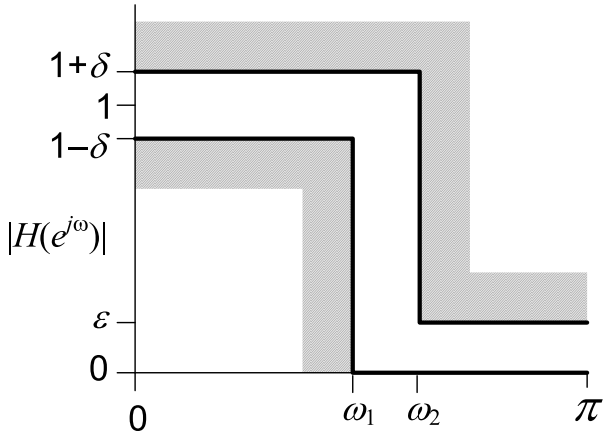


Order Estimation

6: Window Filter Design
Inverse DTFT
Rectangular window
Dirichlet Kernel +
Window relationships
Common Windows
▷ Order Estimation
Example Design
Frequency sampling
Summary
MATLAB routines

Several formulae estimate the required order of a filter, M.

E.g. for lowpass filter



Estimated order is

$$M \approx \frac{-5.6-4.3\log_{10}(\delta\epsilon)}{\omega_2-\omega_1} \approx \frac{-8-20\log_{10}\epsilon}{2.2\Delta\omega}$$

Required M increases as either the transition width, $\omega_2 - \omega_1$, or the gain tolerances δ and ϵ get smaller.
Only approximate.

Example:

Transition band: $f_1 = 1.8$ kHz, $f_2 = 2.0$ kHz, $f_s = 12$ kHz,.

$$\omega_1 = \frac{2\pi f_1}{f_s} = 0.943,\ \omega_2 = \frac{2\pi f_2}{f_s} = 1.047$$

Ripple: $20\log_{10}(1+\delta) = 0.1$ dB, $20\log_{10}\epsilon = -35$ dB

$$\delta = 10^{\frac{0.1}{20}} - 1 = 0.0116,\ \epsilon = 10^{\frac{-35}{20}} = 0.0178$$

$$M \approx \frac{-5.6-4.3\log_{10}\left(2\times10^{-4}\right)}{1.047-0.943} = \frac{10.25}{0.105} = 98 \qquad \text{or } \frac{35-8}{2.2\Delta\omega} = 117$$

Example Design

6: Window Filter Design
Inverse DTFT
Rectangular window
Dirichlet Kernel +
Window relationships
Common Windows
Order Estimation
▷ Example Design
Frequency sampling
Summary
MATLAB routines

Specifications:

Bandpass: $\omega_1 = 0.5$, $\omega_2 = 1$

Transition bandwidth: $\Delta\omega = 0.1$

Ripple: $\delta = \epsilon = 0.02$

$20 \log_{10} \epsilon = -34$ dB

$20 \log_{10} (1 + \delta) = 0.17$ dB

Order:

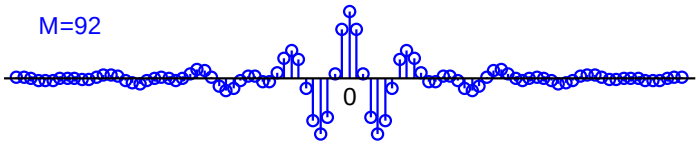
$$M \approx \frac{-5.6 - 4.3 \log_{10}(\delta\epsilon)}{\omega_2 - \omega_1} = 92$$

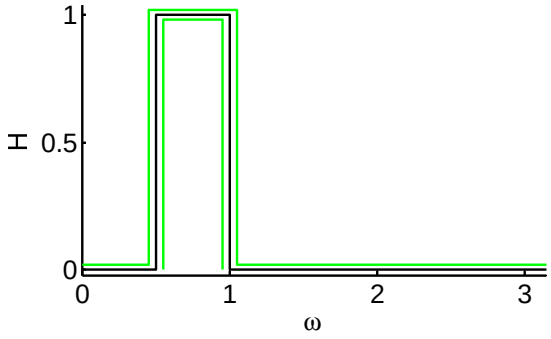
Ideal Impulse Response:

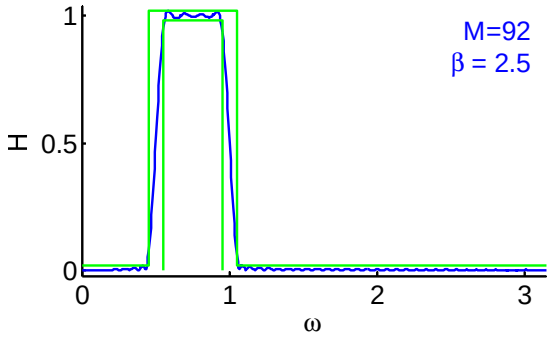
Difference of two lowpass filters

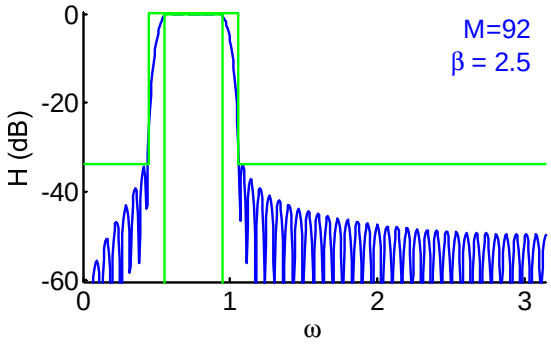
$$h[n] = \frac{\sin \omega_2 n}{\pi n} - \frac{\sin \omega_1 n}{\pi n}$$

Kaiser Window: $\beta = 2.5$









Frequency sampling

6: Window Filter Design
Inverse DTFT
Rectangular window
Dirichlet Kernel +
Window relationships
Common Windows
Order Estimation
Example Design
▷ Frequency sampling
Summary
MATLAB routines

Take $M+1$ uniform samples of $H(e^{j\omega})$; take IDFT to obtain $h[n]$

Advantage:

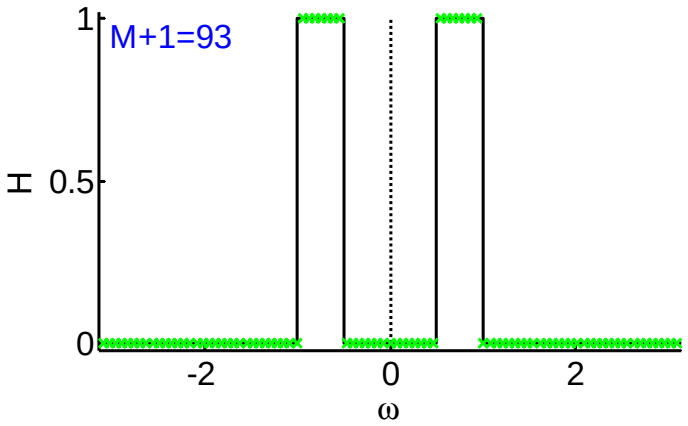
exact match at sample points

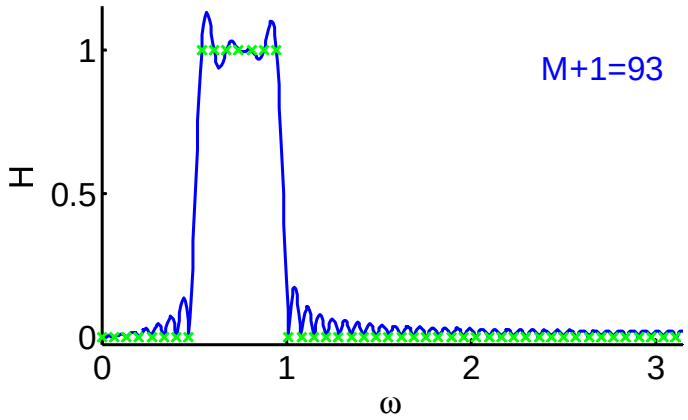
Disadvantage:

poor intermediate approximation if spectrum is varying rapidly

Solutions:

(1) make the filter transitions smooth over $\Delta\omega$ width
(2) oversample and do least squares fit (can't use IDFT)
(3) use non-uniform points with more near transition (can't use IDFT)





Summary

6: Window Filter Design
Inverse DTFT
Rectangular window
Dirichlet Kernel +
Window relationships
Common Windows
Order Estimation
Example Design
Frequency sampling
▷ Summary
MATLAB routines

- Make an FIR filter by windowing the IDTFT of the ideal response
  - Ideal lowpass has $h[n] = \frac{\sin \omega_0 n}{\pi n}$
  - Add/subtract lowpass filters to make any piecewise constant response
- Ideal filter response is $\circledast$ with the DTFT of the window
  - Rectangular window ($W(z) =$ Dirichlet kernel) has -13 dB sidelobes and is always a bad idea
  - Hamming, Blackman-Harris are good
  - Kaiser good with β trading off main lobe width v. sidelobes
- Uncertainty principle: cannot be concentrated in both time and frequency
- Frequency sampling: IDFT of uniform frequency samples: not so great

For further details see Mitra: 7, 10.

MATLAB routines

6: Window Filter Design
Inverse DTFT
Rectangular window
Dirichlet Kernel +
Window relationships
Common Windows
Order Estimation
Example Design
Frequency sampling
Summary
▷ MATLAB routines

diric(x,n)	Dirichlet kernel: $\frac{\sin 0.5nx}{\sin 0.5x}$
hanning hamming kaiser	Window functions (Note 'periodic' option)
kaiserord	Estimate required filter order and β

7: Optimal FIR filters
Optimal Filters
Alternation Theorem
Chebyshev Polynomials
Maximal Error Locations
Remez Exchange Algorithm
Determine Polynomial
Example Design
FIR Pros and Cons
Summary
MATLAB routines

7: Optimal FIR filters

Optimal Filters

7: Optimal FIR filters
▷ Optimal Filters
Alternation Theorem
Chebyshev Polynomials
Maximal Error Locations
Remez Exchange Algorithm
Determine Polynomial
Example Design
FIR Pros and Cons
Summary
MATLAB routines

We restrict ourselves to zero-phase filters of odd length $M+1$, symmetric around $h[0]$, i.e. $h[-n] = h[n]$.

$$\overline{H}(\omega) = H(e^{j\omega}) = \sum_{-\frac{M}{2}}^{\frac{M}{2}} h[n]e^{-jn\omega} = h[0] + 2\sum_{1}^{\frac{M}{2}} h[n]\cos n\omega$$

$\overline{H}(\omega)$ is real but not necessarily positive (unlike $\left|H(e^{j\omega})\right|$).

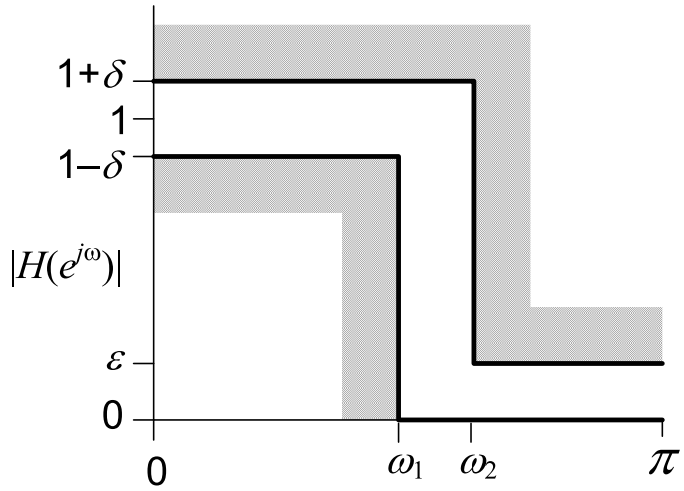
Weighted error: $e(\omega) = s(\omega)\left(\overline{H}(\omega) - d(\omega)\right)$ where $d(\omega)$ is the target.

Choose $s(\omega)$ to control the error variation with ω.

Example: lowpass filter

$$d(\omega) = \begin{cases} 1 & 0 \le \omega \le \omega_1 \\ 0 & \omega_2 \le \omega \le \pi \end{cases}$$

$$s(\omega) = \begin{cases} \delta^{-1} & 0 \le \omega \le \omega_1 \\ \epsilon^{-1} & \omega_2 \le \omega \le \pi \end{cases}$$



$e(\omega) = \pm 1$ when $\overline{H}(\omega)$ lies at the edge of the specification.

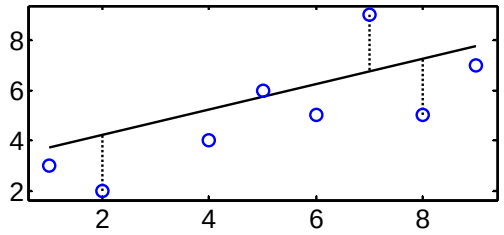
Minimax criterion: $h[n] = \arg\min_{h[n]} \max_{\omega} |e(\omega)|$: minimize max error

Alternation Theorem

7: Optimal FIR filters
Optimal Filters
Alternation Theorem
Chebyshev Polynomials
Maximal Error Locations
Remez Exchange Algorithm
Determine Polynomial
Example Design
FIR Pros and Cons
Summary
MATLAB routines

Want to find the best fit line: with the smallest maximal error.

Best fit line always attains the maximal error three times with alternate signs



Proof:

Assume the first maximal deviation from the line is negative as shown.
There must be an equally large positive deviation; or else just move the line downwards to reduce the maximal deviation.
This must be followed by another maximal negative deviation; or else you can rotate the line and reduce the deviations.

Alternation Theorem:

A polynomial fit of degree n to a set of bounded points is minimax if and only if it attains its maximal error at $n+2$ points with alternating signs.
There may be additional maximal error points.
Fitting to a continuous function is the same as to an infinite number of points.

Chebyshev Polynomials

7: Optimal FIR filters
Optimal Filters
Alternation Theorem
Chebyshev Polynomials
Maximal Error Locations
Remez Exchange Algorithm
Determine Polynomial
Example Design
FIR Pros and Cons
Summary
MATLAB routines

$$\overline{H}(\omega) = H(e^{j\omega}) = h[0] + 2\sum_{1}^{\frac{M}{2}} h[n]\cos n\omega$$

But $\cos n\omega = T_n(\cos\omega)$: Chebyshev polynomial of 1st kind

$$\cos 2\omega = 2\cos^2\omega - 1 = T_2(\cos\omega) \qquad T_2(x) = 2x^2 - 1$$
$$\cos 3\omega = 4\cos^3\omega - 3\cos\omega = T_3(\cos\omega) \qquad T_3(x) = 4x^3 - 3x$$

Recurrence Relation:

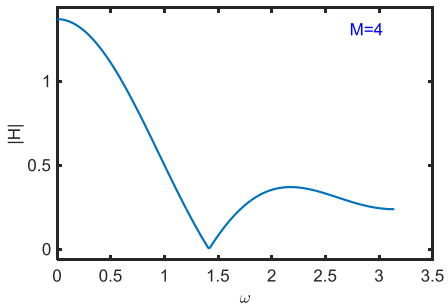
$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x) \text{ with } T_0(x) = 1,\ T_1(x) = x$$

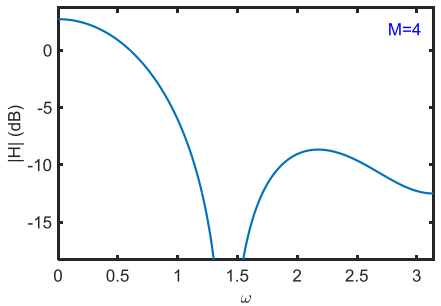
Proof: $\cos(n\omega + \omega) + \cos(n\omega - \omega) = 2\cos\omega\cos n\omega$

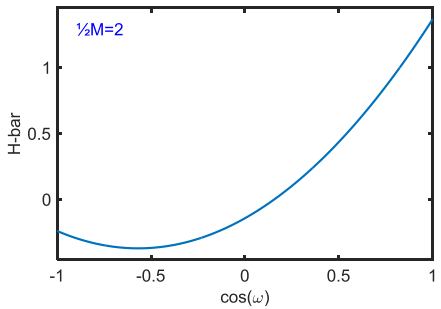
So $\overline{H}(\omega)$ is an $\frac{M}{2}$ order polynomial in $\cos\omega$: alternation theorem applies.

Example: Symmetric lowpass filter of order $M = 4$

$$H(z) = 0.1766z^2 + 0.4015z + 0.2124 + 0.4015z^{-1} + 0.1766z^{-2}$$







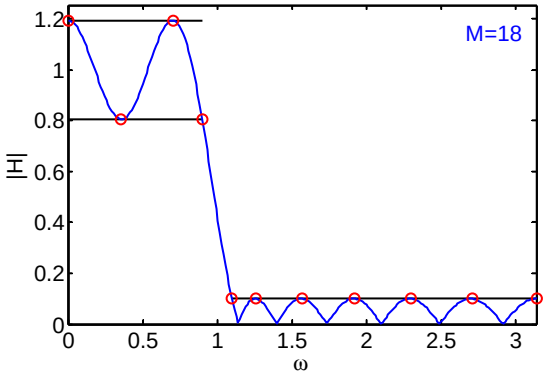
Maximal Error Locations

7: Optimal FIR filters
Optimal Filters
Alternation Theorem
Chebyshev Polynomials
▷ Maximal Error Locations
Remez Exchange Algorithm
Determine Polynomial
Example Design
FIR Pros and Cons
Summary
MATLAB routines

Maximal error locations occur either at band edges or when $\frac{d\overline{H}}{d\omega} = 0$

$$\begin{aligned}\overline{H}(\omega) &= h[0] + 2\sum_{1}^{\frac{M}{2}} h[n]\cos n\omega \\ &= P(\cos\omega)\end{aligned}$$

where $P(x)$ is a polynomial of order $\frac{M}{2}$.



$$\begin{aligned}\frac{d\overline{H}}{d\omega} &= -P'(\cos\omega)\sin\omega \\ &= 0 \text{ at } \omega = 0,\ \pi \text{ and at most } \frac{M}{2} - 1 \text{ zeros of polynomial } P'(x).\end{aligned}$$

$\therefore$ With two bands, we have at most $\frac{M}{2} + 3$ maximal error frequencies.
We require $\frac{M}{2} + 2$ of alternating signs for the optimal fit.

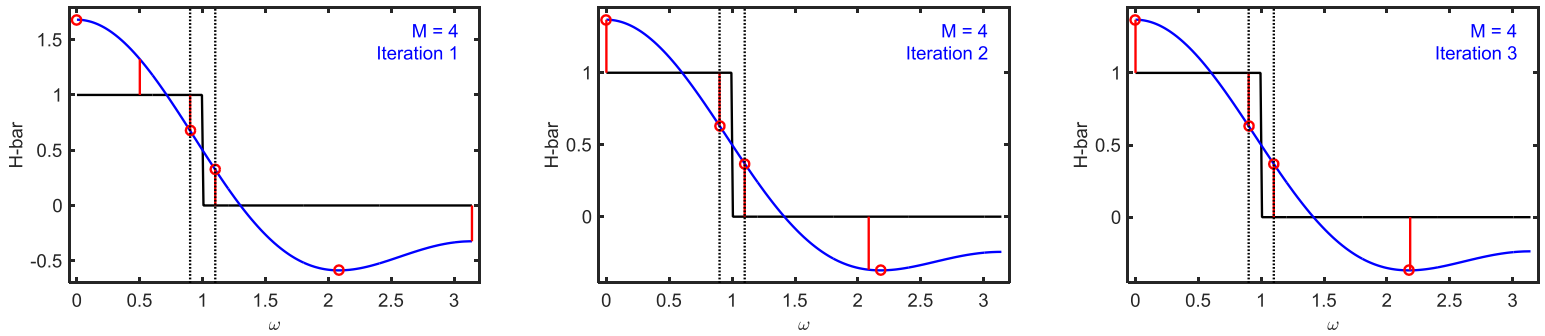
Only three possibilities exist (try them all):

(a) $\omega = 0$ + two band edges + all$\left(\frac{M}{2} - 1\right)$ zeros of $P'(x)$.
(b) $\omega = \pi$ + two band edges + all$\left(\frac{M}{2} - 1\right)$ zeros of $P'(x)$.
(c) $\omega = \{0 \text{ and } \pi\}$ + two band edges + $\left(\frac{M}{2} - 2\right)$ zeros of $P'(x)$.

Remez Exchange Algorithm

7: Optimal FIR filters
Optimal Filters
Alternation Theorem
Chebyshev Polynomials
Maximal Error Locations
▷ Remez Exchange Algorithm
Determine Polynomial
Example Design
FIR Pros and Cons
Summary
MATLAB routines

1. Guess the positions of the $\frac{M}{2}+2$ maximal error frequencies and give alternating signs to the errors (e.g. choose evenly spaced ω).
2. Determine the error magnitude, ϵ, and the $\frac{M}{2}+1$ coefficients of the polynomial that passes through the maximal error locations.
3. Find the local maxima of the error function by evaluating $e(\omega) = s(\omega)\left(\overline{H}(\omega) - d(\omega)\right)$ on a dense set of ω.
4. Update the maximal error frequencies to be an alternating subset of the local maxima + band edges + $\{0$ and/or $\pi\}$.
   If maximum error is $> \epsilon$, go back to step 2. (typically 15 iterations)
5. Evaluate $\overline{H}(\omega)$ on $M+1$ evenly spaced ω and do an IDFT to get $h[n]$.



Remex Step 2: Determine Polynomial

7: Optimal FIR filters
Optimal Filters
Alternation Theorem
Chebyshev Polynomials
Maximal Error Locations
Remez Exchange Algorithm
▷ Determine Polynomial
Example Design
FIR Pros and Cons
Summary
MATLAB routines

For each extremal frequency, ω_i for $1 \leq i \leq \frac{M}{2}+2$

$$d(\omega_i) = \overline{H}(\omega_i) + \frac{(-1)^i \epsilon}{s(\omega_i)} = h[0] + 2\sum_{n=1}^{\frac{M}{2}} h[n]\cos n\omega_i + \frac{(-1)^i \epsilon}{s(\omega_i)}$$

Method 1: (Computation time $\propto M^3$)

Solve $\frac{M}{2}+2$ equations in $\frac{M}{2}+2$ unknowns for $h[n]+\epsilon$.

In step 3, evaluate $\overline{H}(\omega) = h[0] + 2\sum_{n=1}^{\frac{M}{2}} h[n]\cos n\omega_i$

Method 2: Don't calculate $h[n]$ explicitly (Computation time $\propto M^2$)

Multiply the ω_i equation by $c_i = \prod_{j\neq i} \frac{1}{\cos\omega_i - \cos\omega_j}$ and add them:

$$\sum_{i=1}^{\frac{M}{2}+2} c_i \left(h[0] + 2\sum_{n=1}^{\frac{M}{2}} h[n]\cos n\omega + \frac{(-1)^i \epsilon}{s(\omega_i)} \right) = \sum_{i=1}^{\frac{M}{2}+2} c_i d(\omega_i)$$

All terms involving $h[n]$ sum to zero leaving

$$\sum_{i=1}^{\frac{M}{2}+2} \frac{(-1)^i c_i}{s(\omega_i)} \epsilon = \sum_{i=1}^{\frac{M}{2}+2} c_i d(\omega_i)$$

Solve for ϵ then calculate the $\overline{H}(\omega_i)$ then use Lagrange interpolation:

$$\overline{H}(\omega) = P(\cos\omega) = \sum_{i=1}^{\frac{M}{2}+2} \overline{H}(\omega_i) \prod_{j\neq i} \frac{\cos\omega - \cos\omega_j}{\cos\omega_i - \cos\omega_j}$$

$(\frac{M}{2}+1)$-polynomial going through all the $\overline{H}(\omega_i)$ [actually order $\frac{M}{2}$]

Example Design

7: Optimal FIR filters
Optimal Filters
Alternation Theorem
Chebyshev Polynomials
Maximal Error Locations
Remez Exchange Algorithm
Determine Polynomial
▷ Example Design
FIR Pros and Cons
Summary
MATLAB routines

Filter Specifications:

Bandpass $\omega = [0.5,\ 1]$, Transition widths: $\Delta\omega = 0.2$
Stopband Attenuation: -25 dB and -15 dB
Passband Ripple: ± 0.3 dB

Determine gain tolerances for each band:

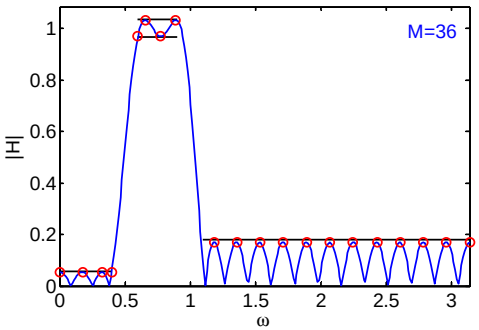
-25 dB $= 0.056$, -0.3 dB $= 1 - 0.034$, -15 dB $= 0.178$

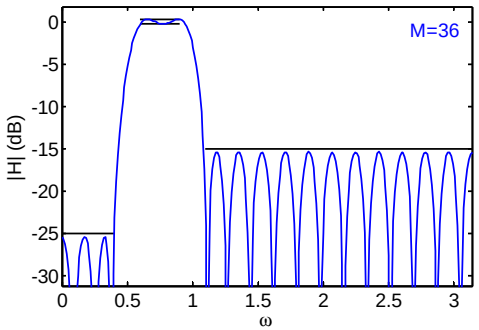
Predicted order: $M = 36$

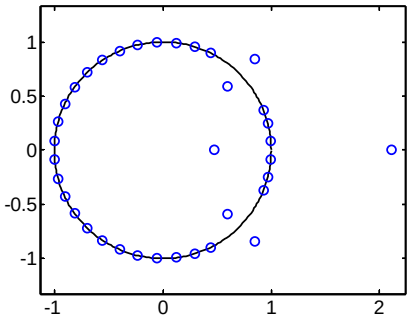
$\frac{M}{2} + 2$ extremal frequencies are distributed between the bands
Filter meets specs ☺; clearer on a decibel scale

Most zeros are on the unit circle + three reciprocal pairs

Reciprocal pairs give a linear phase shift







FIR Pros and Cons

7: Optimal FIR filters
Optimal Filters
Alternation Theorem
Chebyshev Polynomials
Maximal Error Locations
Remez Exchange Algorithm
Determine Polynomial
Example Design
▷ FIR Pros and Cons
Summary
MATLAB routines

- Can have linear phase
  - no envelope distortion, all frequencies have the same delay ☺
  - symmetric or antisymmetric: $h[n] = h[-n] \forall n$ or $-h[-n] \forall n$
  - antisymmetric filters have $H(e^{j0}) = H(e^{j\pi}) = 0$
  - symmetry means you only need $\frac{M}{2} + 1$ multiplications to implement the filter.
- Always stable ☺
- Low coefficient sensitivity ☺
- Optimal design method fast and robust ☺
- Normally needs higher order than an IIR filter ☹
  - Filter order $M \approx \frac{\mathrm{dB}_{\mathrm{atten}}}{3.5\Delta\omega}$ where $\Delta\omega$ is the most rapid transition
  - Filtering complexity $\propto M \times f_s \approx \frac{\mathrm{dB}_{\mathrm{atten}}}{3.5\Delta\omega} f_s = \frac{\mathrm{dB}_{\mathrm{atten}}}{3.5\Delta\Omega} f_s^2$ $\propto f_s^2$ for a given specification in unscaled Ω units.

Summary

7: Optimal FIR filters
Optimal Filters
Alternation Theorem
Chebyshev Polynomials
Maximal Error Locations
Remez Exchange Algorithm
Determine Polynomial
Example Design
FIR Pros and Cons
▷ Summary
MATLAB routines

Optimal Filters: minimax error criterion

- use weight function, $s(\omega)$, to allow different errors in different frequency bands
- symmetric filter has zeros on unit circle or in reciprocal pairs
- Response of symmetric filter is a polynomial in $\cos\omega$
- Alternation Theorem: $\frac{M}{2}+2$ maximal errors with alternating signs

Remez Exchange Algorithm (also known as Parks-McLellan Algorithm)

- multiple constant-gain bands separated by transition regions
- very robust, works for filters with $M > 1000$
- Efficient: computation $\propto M^2$
- can go mad in the transition regions

Modified version works on arbitrary gain function

- Does not always converge

For further details see Mitra: 10.

MATLAB routines

7: Optimal FIR filters
Optimal Filters
Alternation Theorem
Chebyshev Polynomials
Maximal Error Locations
Remez Exchange Algorithm
Determine Polynomial
Example Design
FIR Pros and Cons
Summary
▷ MATLAB routines

firpm	optimal FIR filter design
firpmord	estimate require order for firpm
cfirpm	arbitrary-response filter design
remez	[obsolete] optimal FIR filter design

8: IIR Filter Transformations
Continuous Time Filters
Bilinear Mapping
Continuous Time Filters
Mapping Poles and Zeros
Spectral Transformations
Constantinides Transformations
Impulse Invariance
Summary
MATLAB routines

8: IIR Filter Transformations

Continuous Time Filters

8: IIR Filter Transformations
Continuous Time Filters
Bilinear Mapping
Continuous Time Filters
Mapping Poles and Zeros
Spectral Transformations
Constantinides Transformations
Impulse Invariance
Summary
MATLAB routines

Classical continuous-time filters optimize tradeoff: passband ripple v stopband ripple v transition width
There are explicit formulae for pole/zero positions.

Butterworth: $\widetilde{G}^2(\Omega) = \left|\widetilde{H}(j\Omega)\right|^2 = \frac{1}{1+\Omega^{2N}}$

- Monotonic $\forall\Omega$
- $\widetilde{G}(\Omega) = 1 - \frac{1}{2}\Omega^{2N} + \frac{3}{8}\Omega^{4N} + \cdots$
  "Maximally flat": $2N-1$ derivatives are zero

Chebyshev: $\widetilde{G}^2(\Omega) = \frac{1}{1+\epsilon^2 T_N^2(\Omega)}$

where polynomial $T_N(\cos x) = \cos Nx$

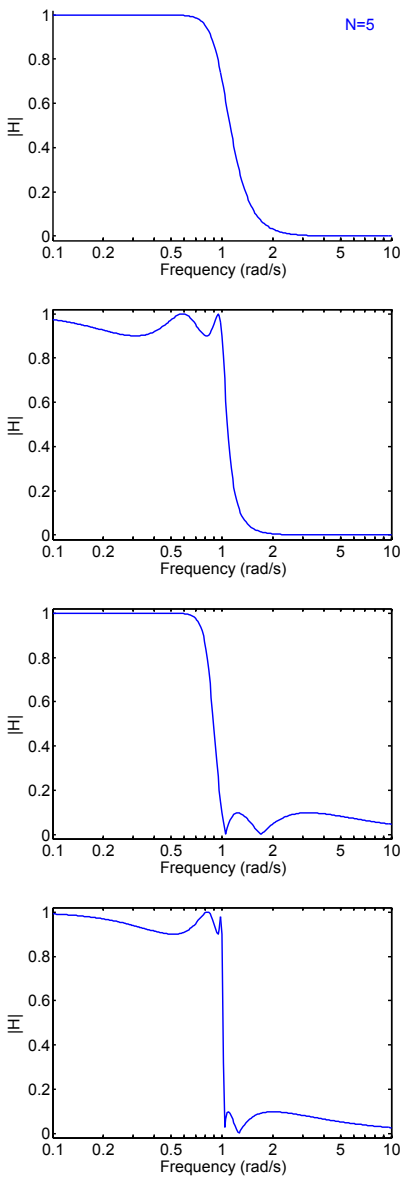
- passband equiripple + very flat at ∞

Inverse Chebyshev: $\widetilde{G}^2(\Omega) = \frac{1}{1+\left(\epsilon^2 T_N^2(\Omega^{-1})\right)^{-1}}$

- stopband equiripple + very flat at 0

Elliptic: [no nice formula]

- Very steep + equiripple in pass and stop bands



Bilinear Mapping

8: IIR Filter Transformations
Continuous Time Filters
▷ Bilinear Mapping
Continuous Time Filters
Mapping Poles and Zeros
Spectral Transformations
Constantinides Transformations
Impulse Invariance
Summary
MATLAB routines

Change variable: $z = \frac{\alpha+s}{\alpha-s} \Leftrightarrow s = \alpha\frac{z-1}{z+1}$: a one-to-one invertible mapping

- $\Re$ axis $(s) \leftrightarrow \Re$ axis (z)
- $\Im$ axis $(s) \leftrightarrow$ Unit circle (z)

  Proof: $z = e^{j\omega} \Leftrightarrow s = \alpha\frac{e^{j\omega}-1}{e^{j\omega}+1} = \alpha\frac{e^{j\frac{\omega}{2}}-e^{-j\frac{\omega}{2}}}{e^{j\frac{\omega}{2}}+e^{-j\frac{\omega}{2}}} = j\alpha\tan\frac{\omega}{2} = j\Omega$

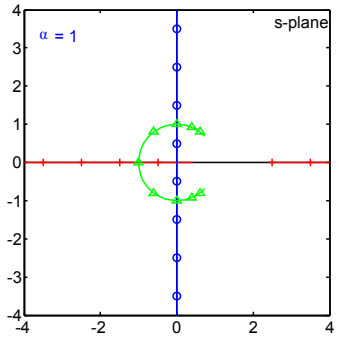
- Left half plane$(s) \leftrightarrow$inside of unit circle (z)

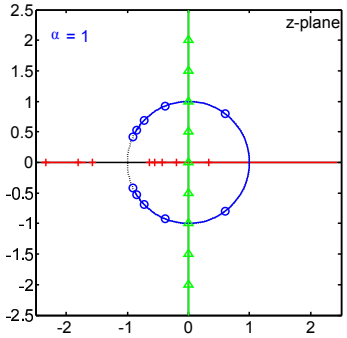
  Proof: $s = x + jy \Leftrightarrow |z|^2 = \frac{|(\alpha+x)+jy|^2}{|(\alpha-x)-jy|^2}$

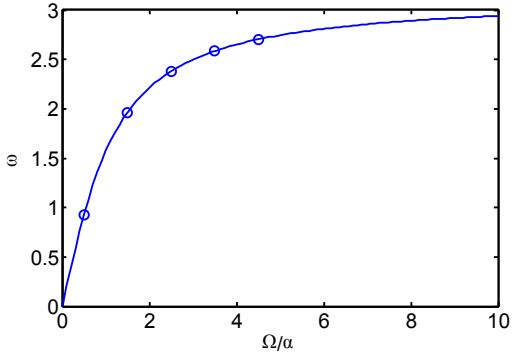
  $$= \frac{\alpha^2+2\alpha x+x^2+y^2}{\alpha^2-2\alpha x+x^2+y^2} = 1 + \frac{4\alpha x}{(\alpha-x)^2+y^2}$$

  $$x < 0 \Leftrightarrow |z| < 1$$

- Unit circle $(s) \leftrightarrow \Im$ axis (z)







Continuous Time Filters

8: IIR Filter Transformations
Continuous Time Filters
Bilinear Mapping
▷ Continuous Time Filters
Mapping Poles and Zeros
Spectral Transformations
Constantinides Transformations
Impulse Invariance
Summary
MATLAB routines

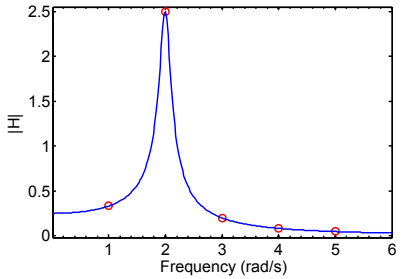
Take $\widetilde{H}(s) = \frac{1}{s^2+0.2s+4}$ and choose $\alpha = 1$

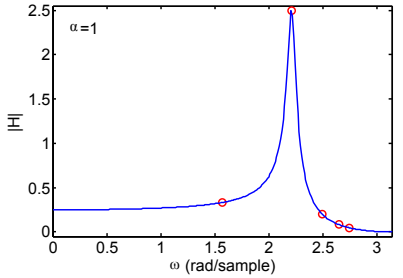
Substitute: $s = \alpha\frac{z-1}{z+1}$ [extra zeros at $z = -1$]

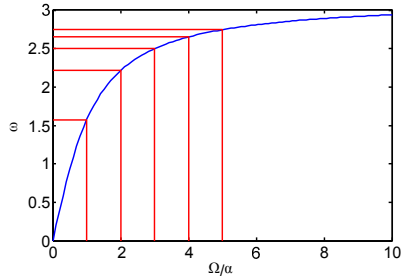
$$H(z) = \frac{1}{\left(\frac{z-1}{z+1}\right)^2+0.2\frac{z-1}{z+1}+4}$$

$$= \frac{(z+1)^2}{(z-1)^2+0.2(z-1)(z+1)+4(z+1)^2}$$

$$= \frac{z^2+2z+1}{5.2z^2+6z+4.8} = 0.19\frac{1+2z^{-1}+z^{-2}}{1+1.15z^{-1}+0.92z^{-2}}$$







Frequency response is identical (both magnitude and phase) but with a distorted frequency axis:

Frequency mapping: $\omega = 2\tan^{-1}\frac{\Omega}{\alpha}$

$$\Omega = \begin{bmatrix} \alpha & 2\alpha & 3\alpha & 4\alpha & 5\alpha \end{bmatrix}$$
$$\rightarrow \omega = \begin{bmatrix} 1.6 & 2.2 & 2.5 & 2.65 & 2.75 \end{bmatrix}$$

Choosing α: Set $\alpha = \frac{\Omega_0}{\tan\frac{1}{2}\omega_0}$ to map $\Omega_0 \rightarrow \omega_0$

Set $\alpha = 2f_s = \frac{2}{T}$ to map low frequencies to themselves

Mapping Poles and Zeros

8: IIR Filter Transformations
Continuous Time Filters
Bilinear Mapping
Continuous Time Filters
▷ Mapping Poles and Zeros
Spectral Transformations
Constantinides Transformations
Impulse Invariance
Summary
MATLAB routines

Alternative method: $\widetilde{H}(s) = \frac{1}{s^2+0.2s+4}$

Find the poles and zeros: $p_s = -0.1 \pm 2j$

Map using $z = \frac{\alpha+s}{\alpha-s} \Rightarrow p_z = -0.58 \pm 0.77j$

After the transformation we will always end up with the same number of poles as zeros:

Add extra poles or zeros at $z = -1$

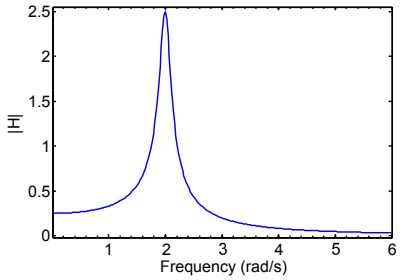
$$H(z) = g \times \frac{(1+z^{-1})^2}{(1+(0.58-0.77j)z^{-1})(1+(0.58+0.77j)z^{-1})}$$
$$= g \times \frac{1+2z^{-1}+z^{-2}}{1+1.15z^{-1}+0.92z^{-2}}$$

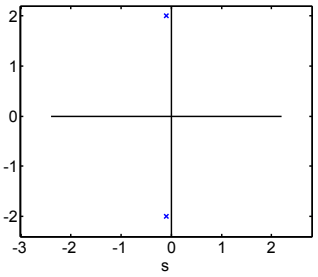
Choose overall scale factor, g, to give the same gain at any convenient pair of mapped frequencies:

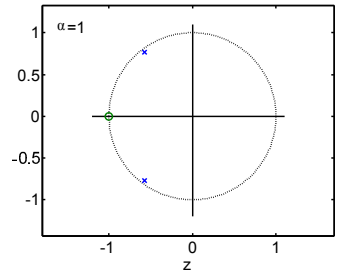
At $\Omega_0 = 0 \Rightarrow s_0 = 0 \Rightarrow \left|\widetilde{H}(s_0)\right| = 0.25$

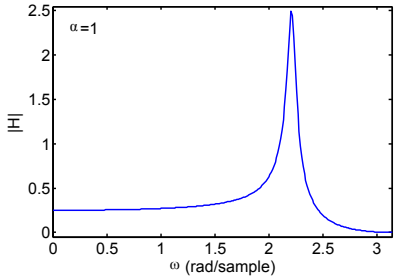
$$\Rightarrow \omega_0 = 2\tan^{-1}\frac{\Omega_0}{\alpha} = 0 \Rightarrow z_0 = e^{j\omega_0} = 1$$
$$\Rightarrow |H(z_0)| = g \times \frac{4}{3.08} = 0.25 \Rightarrow g = 0.19$$

$$H(z) = 0.19\frac{1+2z^{-1}+z^{-2}}{1+1.15z^{-1}+0.92z^{-2}}$$









Spectral Transformations

8: IIR Filter Transformations
Continuous Time Filters
Bilinear Mapping
Continuous Time Filters
Mapping Poles and Zeros
Spectral Transformations
Constantinides Transformations
Impulse Invariance
Summary
MATLAB routines

We can transform the z-plane to change the cutoff frequency by substituting

$$z = \frac{\hat{z}-\lambda}{1-\lambda\hat{z}} \Leftrightarrow \hat{z} = \frac{z+\lambda}{1+\lambda z}$$

Frequency Mapping:

If $z = e^{j\omega}$, then $\hat{z} = z\frac{1+\lambda z^{-1}}{1+\lambda z}$ has modulus 1 since the numerator and denominator are complex conjugates.

Hence the unit circle is preserved.

$$\Rightarrow e^{j\hat{\omega}} = \frac{e^{j\omega}+\lambda}{1+\lambda e^{j\omega}}$$

Some algebra gives: $\tan\frac{\omega}{2} = \left(\frac{1+\lambda}{1-\lambda}\right)\tan\frac{\hat{\omega}}{2}$

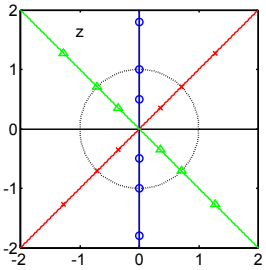
Equivalent to:

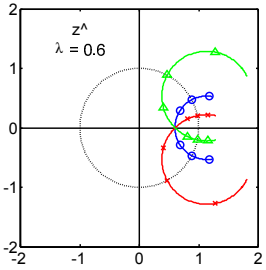
$$z \longrightarrow s = \frac{z-1}{z+1} \longrightarrow \hat{s} = \frac{1-\lambda}{1+\lambda}s \longrightarrow \hat{z} = \frac{1+\hat{s}}{1-\hat{s}}$$

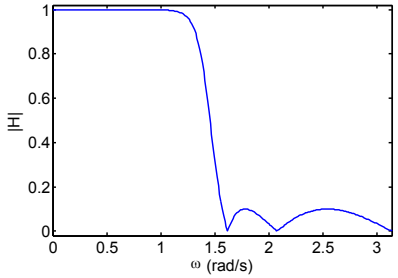
Lowpass Filter example:

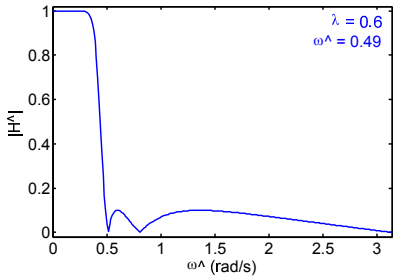
Inverse Chebyshev

$$\omega_0 = \frac{\pi}{2} = 1.57 \overset{\lambda=0.6}{\longrightarrow} \hat{\omega}_0 = 0.49$$







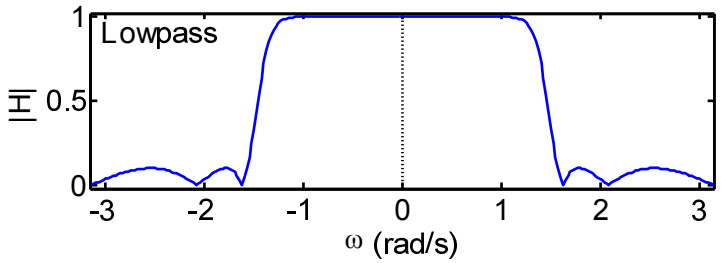


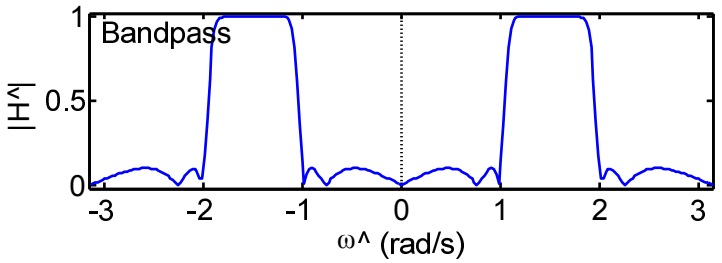
Constantinides Transformations

Transform any lowpass filter with cutoff frequency ω_0 to:

Target	Substitute	Parameters
Lowpass $\hat{\omega} < \hat{\omega}_1$	$z^{-1} = \frac{\hat{z}^{-1} - \lambda}{1 - \lambda \hat{z}^{-1}}$	$\lambda = \frac{\sin\left(\frac{\omega_0 - \hat{\omega}_1}{2}\right)}{\sin\left(\frac{\omega_0 + \hat{\omega}_1}{2}\right)}$
Highpass $\hat{\omega} > \hat{\omega}_1$	$z^{-1} = -\frac{\hat{z}^{-1} + \lambda}{1 + \lambda \hat{z}^{-1}}$	$\lambda = \frac{\cos\left(\frac{\omega_0 + \hat{\omega}_1}{2}\right)}{\cos\left(\frac{\omega_0 - \hat{\omega}_1}{2}\right)}$
Bandpass $\hat{\omega}_1 < \hat{\omega} < \hat{\omega}_2$	$z^{-1} = -\frac{(\rho-1) - 2\lambda\rho\hat{z}^{-1} + (\rho+1)\hat{z}^{-2}}{(\rho+1) - 2\lambda\rho\hat{z}^{-1} + (\rho-1)\hat{z}^{-2}}$	$\lambda = \frac{\cos\left(\frac{\hat{\omega}_2 + \hat{\omega}_1}{2}\right)}{\cos\left(\frac{\hat{\omega}_2 - \hat{\omega}_1}{2}\right)}$ $\rho = \cot\left(\frac{\hat{\omega}_2 - \hat{\omega}_1}{2}\right) \tan\left(\frac{\omega_0}{2}\right)$
Bandstop $\hat{\omega}_1 \not< \hat{\omega} \not< \hat{\omega}_2$	$z^{-1} = \frac{(1-\rho) - 2\lambda\hat{z}^{-1} + (\rho+1)\hat{z}^{-2}}{(\rho+1) - 2\lambda\hat{z}^{-1} + (1-\rho)\hat{z}^{-2}}$	$\lambda = \frac{\cos\left(\frac{\hat{\omega}_2 + \hat{\omega}_1}{2}\right)}{\cos\left(\frac{\hat{\omega}_2 - \hat{\omega}_1}{2}\right)}$ $\rho = \tan\left(\frac{\hat{\omega}_2 - \hat{\omega}_1}{2}\right) \tan\left(\frac{\omega_0}{2}\right)$

Bandpass and bandstop transformations are quadratic and so will double the order:





Impulse Invariance

8: IIR Filter Transformations
Continuous Time Filters
Bilinear Mapping
Continuous Time Filters
Mapping Poles and Zeros
Spectral Transformations
Constantinides Transformations
▷ Impulse Invariance
Summary
MATLAB routines

Bilinear transform works well for a lowpass filter but the non-linear compression of the frequency distorts any other response.

Alternative method: $\widetilde{H}(s) \overset{\mathscr{L}^{-1}}{\longrightarrow} h(t) \overset{\text{sample}}{\longrightarrow} h[n] = T \times h(nT) \overset{\mathscr{Z}}{\longrightarrow} H(z)$

Express $\widetilde{H}(s)$ as a sum of partial fractions $\widetilde{H}(s) = \sum_{i=1}^{N} \frac{g_i}{s - \tilde{p}_i}$

Impulse response is $\tilde{h}(t) = u(t) \times \sum_{i=1}^{N} g_i e^{\tilde{p}_i t}$

Digital filter $\frac{H(z)}{T} = \sum_{i=1}^{N} \frac{g_i}{1 - e^{\tilde{p}_i T} z^{-1}}$ has identical impulse response

Poles of $H(z)$ are $p_i = e^{\tilde{p}_i T}$ (where $T = \frac{1}{f_s}$ is sampling period)

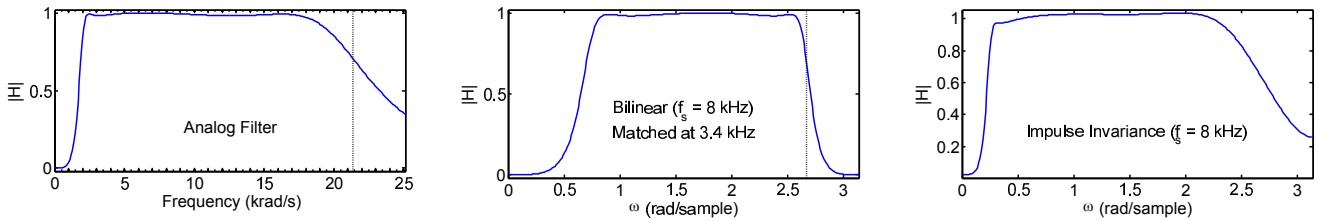
Zeros do not map in a simple way

Properties:

☺ Impulse response correct. ☺ No distortion of frequency axis.

☹ Frequency response is aliased.

Example: Standard telephone filter - 300 to 3400 Hz bandpass



Summary

8: IIR Filter Transformations
Continuous Time Filters
Bilinear Mapping
Continuous Time Filters
Mapping Poles and Zeros
Spectral Transformations
Constantinides Transformations
Impulse Invariance
▷ Summary
MATLAB routines

- Classical filters have optimal tradeoffs in continuous time domain
  - Order $\leftrightarrow$ transition width$\leftrightarrow$ pass ripple$\leftrightarrow$ stop ripple
  - Monotonic passband and/or stopband
- Bilinear mapping
  - Exact preservation of frequency response (mag + phase)
  - non-linear frequency axis distortion
  - can choose α to map $\Omega_0 \rightarrow \omega_0$ for one specific frequency
- Spectral transformations
  - lowpass $\rightarrow$ lowpass, highpass, bandpass or bandstop
  - bandpass and bandstop double the filter order
- Impulse Invariance
  - Aliassing distortion of frequency response
  - preserves frequency axis and impulse response

For further details see Mitra: 9.

MATLAB routines

8: IIR Filter Transformations
Continuous Time Filters
Bilinear Mapping
Continuous Time Filters
Mapping Poles and Zeros
Spectral Transformations
Constantinides Transformations
Impulse Invariance
Summary
▷ MATLAB routines

bilinear	Bilinear mapping
impinvar	Impulse invariance
butter butterord	Analog or digital Butterworth filter
cheby1 cheby1ord	Analog or digital Chebyshev filter
cheby2 cheby2ord	Analog or digital Inverse Chebyshev filter
ellip ellipord	Analog or digital Elliptic filter

9: Optimal IIR Design
Error choices
Linear Least Squares
Frequency Sampling
Iterative Solution
Newton-Raphson
Magnitude-only Specification
Hilbert Relations
Magnitude $\leftrightarrow$ Phase Relation
Summary
MATLAB routines

9: Optimal IIR Design

Error choices

9: Optimal IIR Design
▷ Error choices
Linear Least Squares
Frequency Sampling
Iterative Solution
Newton-Raphson
Magnitude-only Specification
Hilbert Relations
Magnitude ↔ Phase Relation
Summary
MATLAB routines

We want to find a filter $H(e^{j\omega}) = \frac{B(e^{j\omega})}{A(e^{j\omega})}$ that approximates a target response $D(\omega)$. Assume A is order N and B is order M.

Two possible error measures:

Solution Error: $E_S(\omega) = W_S(\omega)\left(\frac{B(e^{j\omega})}{A(e^{j\omega})} - D(\omega)\right)$

Equation Error: $E_E(\omega) = W_E(\omega)\left(B(e^{j\omega}) - D(\omega)A(e^{j\omega})\right)$

We may know $D(\omega)$ completely or else only $|D(\omega)|$

We minimize $\int_{-\pi}^{\pi} |E_*(\omega)|^p \, d\omega$
where $p = 2$ (least squares) or ∞ (minimax).

Weight functions $W_*(\omega)$ are chosen to control relative errors at different frequencies. $W_S(\omega) = |D(\omega)|^{-1}$ gives constant dB error.

We actually want to minimize E_S but E_E is easier because it gives rise to linear equations.

However if $W_E(\omega) = \frac{W_S(\omega)}{|A(e^{j\omega})|}$, then $|E_E(\omega)| = |E_S(\omega)|$

Linear Least Squares

9: Optimal IIR Design
Error choices
▷ Linear Least Squares
Frequency Sampling
Iterative Solution
Newton-Raphson
Magnitude-only Specification
Hilbert Relations
Magnitude ↔ Phase Relation
Summary
MATLAB routines

Overdetermined set of equations $\mathbf{Ax} = \mathbf{b}$ (#equations > #unknowns)

We want to minimize $||\mathbf{e}||^2$ where $\mathbf{e} = \mathbf{Ax} - \mathbf{b}$

$$||\mathbf{e}||^2 = \mathbf{e}^T\mathbf{e} = \left(\mathbf{x}^T\mathbf{A}^T - \mathbf{b}^T\right)(\mathbf{Ax} - \mathbf{b})$$

Differentiate with respect to $\mathbf{x}$:

$$d\left(\mathbf{e}^T\mathbf{e}\right) = d\mathbf{x}^T\mathbf{A}^T(\mathbf{Ax} - \mathbf{b}) + \left(\mathbf{x}^T\mathbf{A}^T - \mathbf{b}^T\right)\mathbf{A}d\mathbf{x}$$

[since $d(\mathbf{uv}) = d\mathbf{u}\,\mathbf{v} + \mathbf{u}\,d\mathbf{v}$]

$$= 2d\mathbf{x}^T\mathbf{A}^T(\mathbf{Ax} - \mathbf{b})$$

[since $\mathbf{u}^T\mathbf{v} = \mathbf{v}^T\mathbf{u}$]

$$= 2d\mathbf{x}^T\left(\mathbf{A}^T\mathbf{Ax} - \mathbf{A}^T\mathbf{b}\right)$$

This is zero for any $d\mathbf{x}$ iff $\mathbf{A}^T\mathbf{Ax} = \mathbf{A}^T\mathbf{b}$

Thus $||\mathbf{e}||^2$ is minimized if $\mathbf{x} = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{b}$

These are the Normal Equations ("Normal" because $\mathbf{A}^T\mathbf{e} = 0$)

The pseudoinverse $\mathbf{x} = \mathbf{A}^+\mathbf{b}$ works even if $\mathbf{A}^T\mathbf{A}$ is singular and finds the $\mathbf{x}$ with minimum $||\mathbf{x}||^2$ that minimizes $||\mathbf{e}||^2$.

This is a very widely used technique.

Frequency Sampling

9: Optimal IIR Design
Error choices
Linear Least Squares
▷ Frequency Sampling
Iterative Solution
Newton-Raphson
Magnitude-only Specification
Hilbert Relations
Magnitude ↔ Phase Relation
Summary
MATLAB routines

For every ω we want: $0 = W(\omega)\left(B(e^{j\omega}) - D(\omega)A(e^{j\omega})\right)$

$$= W(\omega)\left(\sum_{m=0}^{M} b[m]e^{-jm\omega} - D(\omega)\left(1 + \sum_{n=1}^{N} a[n]e^{-jn\omega}\right)\right)$$

$$\Rightarrow \begin{pmatrix} \mathbf{u}(\omega)^T & \mathbf{v}(\omega)^T \end{pmatrix} \begin{pmatrix} \mathbf{a} \\ \mathbf{b} \end{pmatrix} = W(\omega)D(\omega)$$

where $\mathbf{u}(\omega)^T = -W(\omega)D(\omega)\begin{bmatrix} e^{-j\omega} & e^{-j2\omega} & \cdots & e^{-jN\omega} \end{bmatrix}$

$$\mathbf{v}(\omega)^T = W(\omega)\begin{bmatrix} 1 & e^{-j\omega} & e^{-j2\omega} & \cdots & e^{-jM\omega} \end{bmatrix}$$

Choose K values of ω, $\begin{Bmatrix} \omega_1 & \cdots & \omega_K \end{Bmatrix}$ [with $K \geq \frac{M+N+1}{2}$]

$$\begin{pmatrix} \mathbf{U}^T & \mathbf{V}^T \end{pmatrix} \begin{pmatrix} \mathbf{a} \\ \mathbf{b} \end{pmatrix} = \mathbf{d}$$ [K equations, $M+N+1$ unkowns]

where $\mathbf{U} = \begin{bmatrix} \mathbf{u}(\omega_1) & \cdots & \mathbf{u}(\omega_K) \end{bmatrix}$,

$$\mathbf{V} = \begin{bmatrix} \mathbf{v}(\omega_1) & \cdots & \mathbf{v}(\omega_K) \end{bmatrix},$$

$$\mathbf{d} = \begin{bmatrix} W(\omega_1)D(\omega_1) & \cdots & W(\omega_K)D(\omega_K) \end{bmatrix}^T$$

We want to force $\mathbf{a}$ and $\mathbf{b}$ to be real; find least squares solution to

$$\begin{pmatrix} \Re\left(\mathbf{U}^T\right) & \Re\left(\mathbf{V}^T\right) \\ \Im\left(\mathbf{U}^T\right) & \Im\left(\mathbf{V}^T\right) \end{pmatrix} \begin{pmatrix} \mathbf{a} \\ \mathbf{b} \end{pmatrix} = \begin{pmatrix} \Re\left(\mathbf{d}\right) \\ \Im\left(\mathbf{d}\right) \end{pmatrix}$$

Iterative Solution

9: Optimal IIR Design
Error choices
Linear Least Squares
Frequency Sampling
▷ Iterative Solution
Newton-Raphson
Magnitude-only Specification
Hilbert Relations
Magnitude ↔ Phase Relation
Summary
MATLAB routines

Least squares solution minimizes the E_E rather than E_S.

However $E_E = E_S$ if $W_E(\omega) = \frac{W_S(\omega)}{|A(e^{j\omega})|}$.

We can use an iterative solution technique:

1. Select K frequencies $\{\omega_k\}$ (e.g. uniformly spaced)
2. Initialize $W_E(\omega_k) = W_S(\omega_k)$
3. Find least squares solution to
   $$W_E(\omega_k)\left(B(e^{j\omega_k}) - D(\omega_k)A(e^{j\omega_k})\right) = 0 \forall k$$
4. Force $A(z)$ to be stable
   Replace pole p_i by $(p_i^*)^{-1}$ whenever $|p_i| \geq 1$
5. Update weights: $W_E(\omega_k) = \frac{W_S(\omega_k)}{|A(e^{j\omega_k})|}$
6. Return to step 3 until convergence

But for faster convergence use Newton-Raphson ...

Newton-Raphson

9: Optimal IIR Design
Error choices
Linear Least Squares
Frequency Sampling
Iterative Solution
▷ Newton-Raphson
Magnitude-only Specification
Hilbert Relations
Magnitude ↔ Phase Relation
Summary
MATLAB routines

Newton: To solve $f(x) = 0$ given an initial guess x_0, we write

$$f(x) \approx f(x_0) + (x - x_0) f'(x_0) \Rightarrow x = x_0 - \frac{f(x_0)}{f'(x_0)}$$

Converges very rapidly once x_0 is close to the solution

So for each ω_k, we can write (omitting the ω and $e^{j\omega}$ arguments)

$$\begin{aligned} E_S &\approx W_S \left(\frac{B_0}{A_0} - D \right) + \frac{W_S}{A_0} (B - B_0) - \frac{W_S B_0}{A_0^2} (A - A_0) \\ &= \frac{W_S}{A_0} \left(B_0 - A_0 D + B - B_0 - \frac{B_0}{A_0} (A - 1) - \frac{B_0}{A_0} + B_0 \right) \end{aligned}$$

From which we get a linear equation for each ω_k :

$$\begin{pmatrix} \frac{B_0}{DA_0}\mathbf{u}^T & \mathbf{v}^T \end{pmatrix} \begin{pmatrix} \mathbf{a} \\ \mathbf{b} \end{pmatrix} = W \left(A_0 D + \frac{B_0}{A_0} - B_0 \right)$$

where $W = \frac{W_S}{A_0}$ and, as before, $u_n(\omega) = -W(\omega) D(\omega) e^{-jn\omega}$ for $n \in 1 : N$ and $v_m(\omega) = W(\omega) e^{-jm\omega}$ for $m \in 0 : M$.

At each iteration, calculate $A_0(e^{j\omega_k})$ and $B_0(e^{j\omega_k})$ based on $\mathbf{a}$ and $\mathbf{b}$ from the previous iteration.

Then use linear least squares to minimize the linearized E_S using the above equation replicated for each of the ω_k.

Magnitude-only Specification

9: Optimal IIR Design
Error choices
Linear Least Squares
Frequency Sampling
Iterative Solution
Newton-Raphson
▷ Magnitude-only Specification
Hilbert Relations
Magnitude ↔ Phase Relation
Summary
MATLAB routines

If the filter specification only dictates the target magnitude: $|D(\omega)|$, we need to select the target phase.

Solution:

Make an initial guess of the phase and then at each iteration update $\angle D(\omega) = \angle \frac{B(e^{j\omega})}{A(e^{j\omega})}$.

Initial Guess:

If $H(e^{j\omega})$ is causal and minimum phase then the magnitude and phase are not independent:

$$\angle H(e^{j\omega}) = -\ln \left|H(e^{j\omega})\right| \circledast \cot \tfrac{\omega}{2}$$
$$\ln \left|H(e^{j\omega})\right| = \ln |H(\infty)| + \angle H(e^{j\omega}) \circledast \cot \tfrac{\omega}{2}$$

where $\circledast$ is circular convolution and $\cot x$ is taken to be zero for $-\epsilon < x < \epsilon$ for some small value of ϵ and we take the limit as $\epsilon \to 0$.

This result is a consequence of the Hilbert Relations.

Hilbert Relations

9: Optimal IIR Design
Error choices
Linear Least Squares
Frequency Sampling
Iterative Solution
Newton-Raphson
Magnitude-only Specification
▷ Hilbert Relations
Magnitude ↔ Phase Relation
Summary
MATLAB routines

We define $t[n] = u[n-1] - u[-1-n]$

$$T(z) = \frac{z^{-1}}{1-z^{-1}} - \frac{z}{1-z} = \frac{1+z^{-1}}{1-z^{-1}}$$

$$T(e^{j\omega}) = \frac{1+e^{-j\omega}}{1-e^{-j\omega}} = \frac{e^{j\frac{\omega}{2}}+e^{-j\frac{\omega}{2}}}{e^{j\frac{\omega}{2}}-e^{-j\frac{\omega}{2}}}$$

$$= \frac{2\cos\frac{\omega}{2}}{2j\sin\frac{\omega}{2}} = -j\cot\frac{\omega}{2}$$

$h[n] \rightarrow$ even/odd parts: $h_e[n] = \frac{1}{2}\left(h[n] + h[-n]\right)$

$$h_o[n] = \frac{1}{2}\left(h[n] - h[-n]\right)$$

so $\Re\left(H(e^{j\omega})\right) = H_e(e^{j\omega})$

$$\Im\left(H(e^{j\omega})\right) = -jH_o(e^{j\omega})$$

If $h[n]$ is causal: $h_o[n] = h_e[n]t[n]$

$$h_e[n] = h[0]\delta[n] + h_o[n]t[n]$$

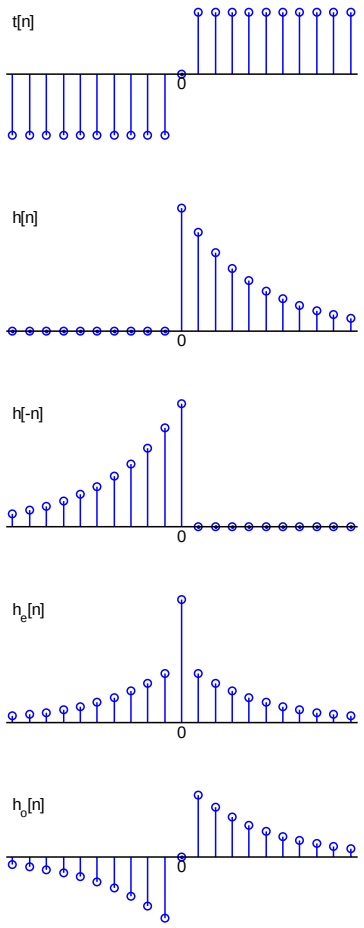
Hence, for causal $h[n]$:

$$\Im\left(H(e^{j\omega})\right) = -j\left(\Re\left(H(e^{j\omega})\right) \circledast -j\cot\frac{\omega}{2}\right)$$

$$= -\Re\left(H(e^{j\omega})\right) \circledast \cot\frac{\omega}{2}$$

$$\Re\left(H(e^{j\omega})\right) = H(\infty) + j\Im\left(H(e^{j\omega})\right) \circledast -j\cot\frac{\omega}{2}$$

$$= H(\infty) + \Im\left(H(e^{j\omega})\right) \circledast \cot\frac{\omega}{2}$$

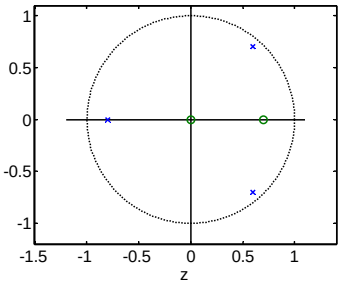


Magnitude $\leftrightarrow$ Phase Relation

9: Optimal IIR Design
Error choices
Linear Least Squares
Frequency Sampling
Iterative Solution
Newton-Raphson
Magnitude-only Specification
Hilbert Relations
Magnitude $\leftrightarrow$ Phase Relation
Summary
MATLAB routines

Given $H(z) = g\frac{\prod(1-q_m z^{-1})}{\prod(1-p_n z^{-1})}$

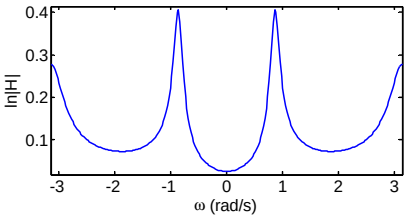
$$\begin{aligned}\ln H(z) &= \ln(g) + \sum \ln\left(1-q_m z^{-1}\right) \\ &\quad - \sum \ln\left(1-p_n z^{-1}\right) \\ &= \ln|H(z)| + j\angle H(z)\end{aligned}$$



Taylor Series:

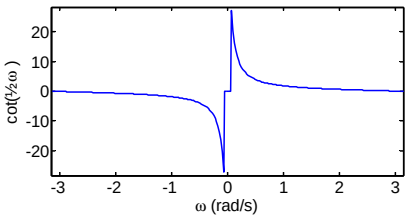
$$\ln\left(1-az^{-1}\right) = -az^{-1} - \frac{a^2}{2}z^{-2} - \frac{a^3}{3}z^{-3} - \ldots$$

causal and stable provided $|a| < 1$

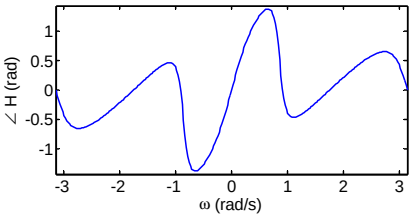


So, if $H(z)$ is minimum phase (all p_n and q_m inside unit circle) then $\ln H(z)$ is the z-transform of a stable causal sequence and:

$$\begin{aligned}\angle H(e^{j\omega}) &= -\ln\left|H(e^{j\omega})\right| \circledast \cot\tfrac{\omega}{2} \\ \ln\left|H(e^{j\omega})\right| &= \ln|g| + \angle H(e^{j\omega}) \circledast \cot\tfrac{\omega}{2}\end{aligned}$$



Example: $H(z) = \frac{10-7z^{-1}}{100-40z^{-1}-11z^{-2}+68z^{-3}}$



Note symmetric dead band in $\cot\frac{\omega}{2}$ for $|\omega| < \epsilon$

Summary

9: Optimal IIR Design
Error choices
Linear Least Squares
Frequency Sampling
Iterative Solution
Newton-Raphson
Magnitude-only Specification
Hilbert Relations
Magnitude $\leftrightarrow$ Phase Relation
▷ Summary
MATLAB routines

- Want to minimize solution error, E_S, but E_E gives linear equations:
  - $E_S(\omega) = W_S(\omega)\left(\frac{B(e^{j\omega})}{A(e^{j\omega})} - D(\omega)\right)$
  - $E_E(\omega) = W_E(\omega)\left(B(e^{j\omega}) - D(\omega)A(e^{j\omega})\right)$
  - use $W_*(\omega)$ to weight errors at different ω.
- Linear least squares: solution to overdetermined $\mathbf{Ax} = \mathbf{b}$
  - Least squares error: $\hat{\mathbf{x}} = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{b}$
- Closed form solution: least squares E_E at $\{\omega_k\}$
  - use $W_E(\omega) = \frac{W_S(\omega)}{|A(e^{j\omega})|}$ to approximate E_S
  - use Taylor series to approximate E_S better (Newton-Raphson)
- Hilbert relations
  - relate $\Re\left(H\left(e^{j\omega}\right)\right)$ and $\Im\left(H\left(e^{j\omega}\right)\right)$ for causal stable sequences
  - $\Rightarrow$ relate $\ln\left|H\left(e^{j\omega}\right)\right|$ and $\angle H\left(e^{j\omega}\right)$ for causal stable minimum phase sequences

For further details see Mitra: 9.

MATLAB routines

9: Optimal IIR Design
Error choices
Linear Least Squares
Frequency Sampling
Iterative Solution
Newton-Raphson
Magnitude-only Specification
Hilbert Relations
Magnitude $\leftrightarrow$ Phase Relation
Summary
▷ MATLAB routines

invfreqz	IIR design for complex response

10: Digital Filter Structures
Direct Forms
Transposition
State Space +
Precision Issues
Coefficient Sensitivity
Cascaded Biquads
Pole-zero Pairing/Ordering
Linear Phase
Hardware Implementation
Allpass Filters
Lattice Stage +
Example
$A(z) \leftrightarrow D(z)$
Allpass Lattice
Lattice Filter
Lattice Example
Lattice Example Numerator
Summary
MATLAB routines

10: Digital Filter Structures

Direct Forms

10: Digital Filter Structures
▷ Direct Forms
Transposition
State Space +
Precision Issues
Coefficient Sensitivity
Cascaded Biquads
Pole-zero Pairing/Ordering
Linear Phase
Hardware Implementation
Allpass Filters
Lattice Stage +
Example
$A(z) \leftrightarrow D(z)$
Allpass Lattice
Lattice Filter
Lattice Example
Lattice Example Numerator
Summary
MATLAB routines

Filter: $H(z) = \frac{B(z)}{A(z)}$ with input $x[n]$ and output $y[n]$

$$y[n] = \sum_{k=0}^{M} b[k]x[n-k] - \sum_{k=1}^{N} a[k]y[n-k]$$

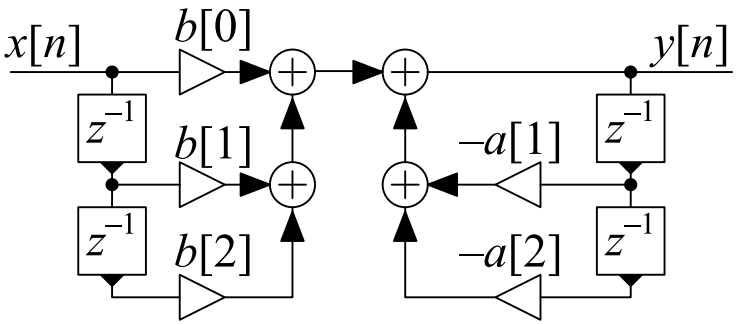
Direct forms use coefficients $a[k]$ and $b[k]$ directly

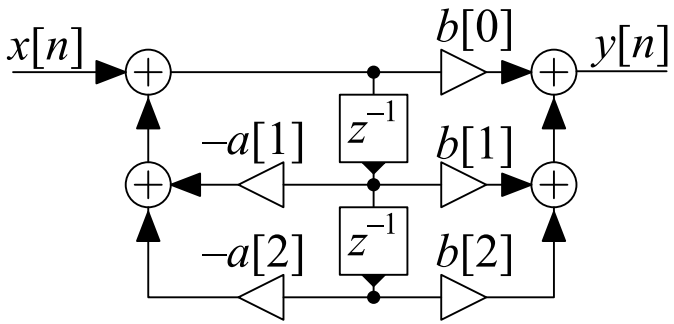
Direct Form 1:

- Direct implementation of difference equation
- Can view as $B(z)$ followed by $\frac{1}{A(z)}$

Direct Form II:

- Implements $\frac{1}{A(z)}$ followed by $B(z)$
- Saves on delays (= storage)





Transposition

10: Digital Filter Structures
Direct Forms
▷ Transposition
State Space +
Precision Issues
Coefficient Sensitivity
Cascaded Biquads
Pole-zero Pairing/Ordering
Linear Phase
Hardware Implementation
Allpass Filters
Lattice Stage +
Example
$A(z) \leftrightarrow D(z)$
Allpass Lattice
Lattice Filter
Lattice Example
Lattice Example Numerator
Summary
MATLAB routines

Can convert any block diagram into an equivalent transposed form:

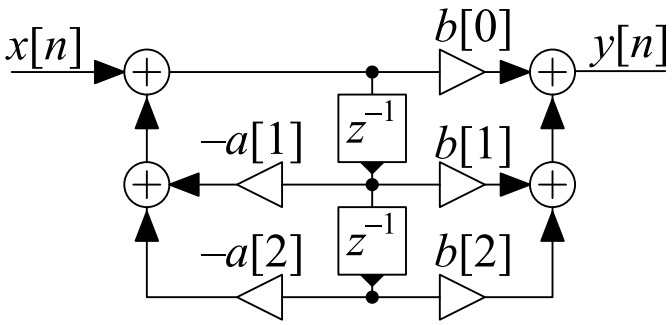
- Reverse direction of each interconnection
- Reverse direction of each multiplier
- Change junctions to adders and vice-versa
- Interchange the input and output signals

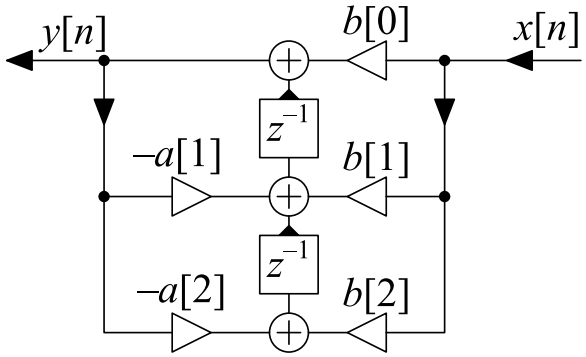
Example:

Direct form II $\rightarrow$ Direct Form II$_t$

Would normally be drawn with input on the left

Note: A valid block diagram must never have any feedback loops that don't go through a delay (z^{-1} block).





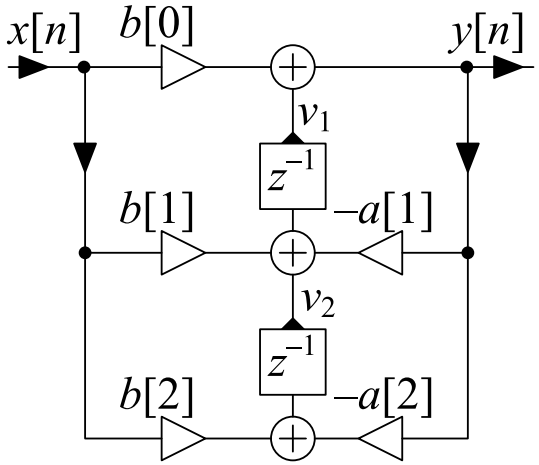
10: Digital Filter Structures
Direct Forms
Transposition
▷ State Space +
Precision Issues
Coefficient Sensitivity
Cascaded Biquads
Pole-zero Pairing/Ordering
Linear Phase
Hardware Implementation
Allpass Filters
Lattice Stage +
Example $A(z) \leftrightarrow D(z)$
Allpass Lattice
Lattice Filter
Lattice Example
Lattice Example Numerator
Summary
MATLAB routines

$\mathbf{v}[n]$ is a vector of delay element outputs

Can write: $\mathbf{v}[n+1] = \mathbf{P}\mathbf{v}[n] + \mathbf{q}x[n]$

$$y[n] = \mathbf{r}^T\mathbf{v}[n] + sx[n]$$

$\{\mathbf{P}, \mathbf{q}, \mathbf{r}^T, s\}$ is the state-space representation of the filter structure.



The transfer function is given by:

$$H(z) = \frac{B(z)}{A(z)} = \frac{\det\left(z\mathbf{I}-\mathbf{P}+\mathbf{q}\mathbf{r}^T\right)}{\det(z\mathbf{I}-\mathbf{P})} + s - 1$$

The transposed form has $\mathbf{P} \to \mathbf{P}^T$ and $\mathbf{q} \leftrightarrow \mathbf{r}$ $\Rightarrow$ same $H(z)$

Example: Direct Form II$_t$

$$\mathbf{P} = \begin{pmatrix} -a[1] & 1 \\ -a[2] & 0 \end{pmatrix} \qquad \mathbf{q} = \begin{pmatrix} b[1] - b[0]a[1] \\ b[2] - b[0]a[2] \end{pmatrix}$$

$$\mathbf{r}^T = \begin{pmatrix} 1 & 0 \end{pmatrix} \qquad s = b[0]$$

From which $H(z) = \frac{b[0]z^2+b[1]z+b[2]}{z^2+a[1]z+a[2]}$

[State-Space → Transfer Function]

[This is not examinable]

We start by proving a useful formula which shows how the determinant of a matrix, $\mathbf{A}$, changes when you add a rank-1 matrix, $\mathbf{q}\mathbf{r}^T$, onto it. The formula is known as the Matrix Determinant Lemma. For any nonsingular matrix $\mathbf{A}$ and column vectors $\mathbf{q}$ and $\mathbf{r}$, we can write

$$\begin{pmatrix} 1 & \mathbf{r}^T \\ \mathbf{0} & \mathbf{A} \end{pmatrix} \begin{pmatrix} 1+\mathbf{r}^T\mathbf{A}^{-1}\mathbf{q} & \mathbf{0}^T \\ -\mathbf{A}^{-1}\mathbf{q} & \mathbf{I} \end{pmatrix} = \begin{pmatrix} 1 & \mathbf{0}^T \\ -\mathbf{q} & \mathbf{I} \end{pmatrix} \begin{pmatrix} 1 & \mathbf{r}^T \\ \mathbf{0} & \mathbf{A}+\mathbf{q}\mathbf{r}^T \end{pmatrix}.$$

It is easy to verify this by multiplying out the matrices. We now take the determinant of both sides making use of the result that the determinant of a block triangular matrix is the product of the determinants of the blocks along the diagonal (assuming they are all square). This gives:

$$\det(\mathbf{A}) \times \left(1+\mathbf{r}^T\mathbf{A}^{-1}\mathbf{q}\right) = \det\left(\mathbf{A}+\mathbf{q}\mathbf{r}^T\right) \qquad \Rightarrow \qquad \mathbf{r}^T\mathbf{A}^{-1}\mathbf{q} = \frac{\det\left(\mathbf{A}+\mathbf{q}\mathbf{r}^T\right)}{\det(\mathbf{A})} - 1$$

Now we take the z-transform of the state space equations

$$\begin{aligned} \mathbf{v}[n+1] &= \mathbf{P}\mathbf{v}[n] + \mathbf{q}x[n] \\ y[n] &= \mathbf{r}^T\mathbf{v}[n] + sx[n] \end{aligned} \quad \underset{z-\text{transform}}{\longrightarrow} \quad \begin{aligned} z\mathbf{V} &= \mathbf{P}\mathbf{V} + \mathbf{q}X \\ Y &= \mathbf{r}^T\mathbf{V} + sX \end{aligned}$$

The upper equation gives $(z\mathbf{I}-\mathbf{P})\,\mathbf{V} = \mathbf{q}X$ from which $\mathbf{V} = (z\mathbf{I}-\mathbf{P})^{-1}\,\mathbf{q}X$ and by substituting this in the lower equation, we get $\frac{Y}{X} = \mathbf{r}^T\,(z\mathbf{I}-\mathbf{P})^{-1}\,\mathbf{q} + s \;=\; \frac{\det\left(z\mathbf{I}-\mathbf{P}+\mathbf{q}\mathbf{r}^T\right)}{\det(z\mathbf{I}-\mathbf{P})} + s - 1.$

Precision Issues

10: Digital Filter Structures
Direct Forms
Transposition
State Space +
▷ Precision Issues
Coefficient Sensitivity
Cascaded Biquads
Pole-zero Pairing/Ordering
Linear Phase
Hardware Implementation
Allpass Filters
Lattice Stage +
Example
$A(z) \leftrightarrow D(z)$
Allpass Lattice
Lattice Filter
Lattice Example
Lattice Example Numerator
Summary
MATLAB routines

If all computations were exact, it would not make any difference which of the equivalent structures was used. However ...

- Coefficient precision
  Coefficients are stored to finite precision and so are not exact.
  The filter actually implemented is therefore incorrect.
- Arithmetic precision
  Arithmetic calculations are not exact.
  - Worst case for arithmetic errors is when calculating the difference between two similar values:
    $1.23456789 - 1.23455678 = 0.00001111$: 9 s.f. $\rightarrow$ 4 s.f.

  Arithmetic errors introduce noise that is then filtered by the transfer function between the point of noise creation and the output.

Coefficient Sensitivity

10: Digital Filter Structures
Direct Forms
Transposition
State Space +
Precision Issues
Coefficient Sensitivity
Cascaded Biquads
Pole-zero Pairing/Ordering
Linear Phase
Hardware Implementation
Allpass Filters
Lattice Stage +
Example $A(z) \leftrightarrow D(z)$
Allpass Lattice
Lattice Filter
Lattice Example
Lattice Example Numerator
Summary
MATLAB routines

The roots of high order polynomials can be very sensitive to small changes in coefficient values.

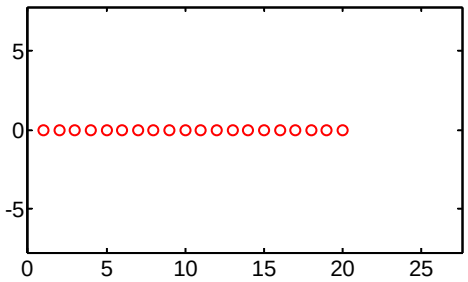
Wilkinson's polynomial: (famous example)

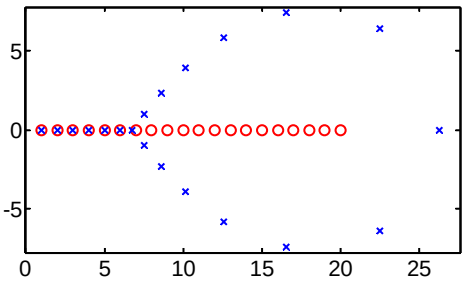
$$f(x) = \prod_{n=1}^{20} (x - n) = x^{20} - 210x^{19} + 20615x^{18} - \ldots$$

has roots well separated on the real axis.

Multiplying the coefficient of x^{19} by 1.000001 moves the roots a lot.

"Speaking for myself I regard it as the most traumatic experience in my career as a numerical analyst", James Wilkinson 1984





Moral: Avoid using direct form for filters orders over about 10.

Cascaded Biquads

10: Digital Filter Structures
Direct Forms
Transposition
State Space +
Precision Issues
Coefficient Sensitivity
▷ Cascaded Biquads
Pole-zero Pairing/Ordering
Linear Phase
Hardware Implementation
Allpass Filters
Lattice Stage +
Example
$A(z) \leftrightarrow D(z)$
Allpass Lattice
Lattice Filter
Lattice Example
Lattice Example Numerator
Summary
MATLAB routines

Avoid high order polynomials by factorizing into quadratic terms:

$$\frac{B(z)}{A(z)} = g\frac{\prod\left(1+b_{k,1}z^{-1}+b_{k,2}z^{-2}\right)}{\prod\left(1+a_{k,1}z^{-1}+a_{k,2}z^{-2}\right)} = g\prod_{k=1}^{K}\frac{1+b_{k,1}z^{-1}+b_{k,2}z^{-2}}{1+a_{k,1}z^{-1}+a_{k,2}z^{-2}}$$

where $K = \max\left(\left\lceil\frac{M}{2}\right\rceil, \left\lceil\frac{N}{2}\right\rceil\right)$.

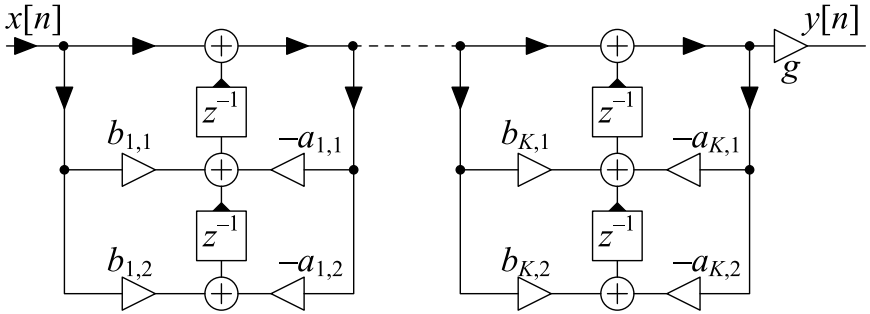
The term $\frac{1+b_{k,1}z^{-1}+b_{k,2}z^{-2}}{1+a_{k,1}z^{-1}+a_{k,2}z^{-2}}$ is a biquad (bi-quadratic section).

We need to choose:

(a) which poles to pair with which zeros in each biquad

(b) how to order the biquads

Direct Form II Transposed

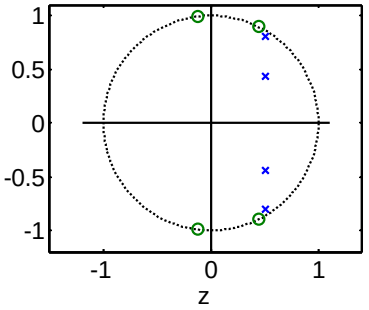


Pole-zero Pairing/Ordering

10: Digital Filter Structures
Direct Forms
Transposition
State Space +
Precision Issues
Coefficient Sensitivity
Cascaded Biquads
▷ Pole-zero Pairing/Ordering
Linear Phase
Hardware Implementation
Allpass Filters
Lattice Stage +
Example
$A(z) \leftrightarrow D(z)$
Allpass Lattice
Lattice Filter
Lattice Example
Lattice Example Numerator
Summary
MATLAB routines

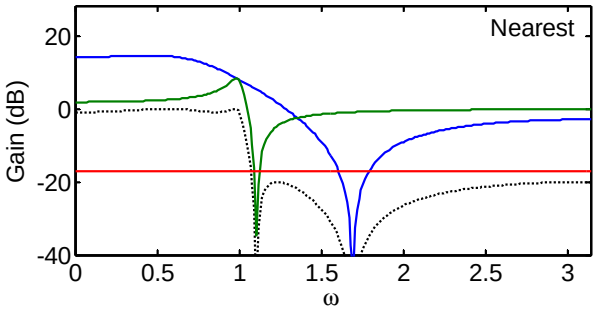
Example: Elliptic lowpass filter

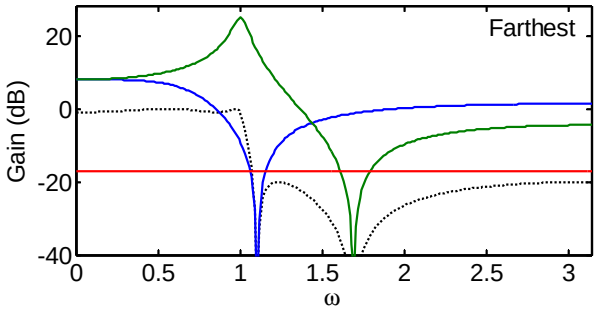
2 pole pairs and 2 zero pairs
need 2 biquads



Noise introduced in one biquad is amplified by all the subsequent ones:

- Make the peak gain of each biquad as small as possible
  - Pair poles with nearest zeros to get lowest peak gain
    begin with the pole nearest the unit circle
  - Pairing with farthest zeros gives higher peak biquad gain
- Poles near the unit circle have the highest peaks and introduce most noise so place them last in the chain





Linear Phase

10: Digital Filter Structures
Direct Forms
Transposition
State Space +
Precision Issues
Coefficient Sensitivity
Cascaded Biquads
Pole-zero Pairing/Ordering
▷ Linear Phase
Hardware Implementation
Allpass Filters
Lattice Stage +
Example
$A(z) \leftrightarrow D(z)$
Allpass Lattice
Lattice Filter
Lattice Example
Lattice Example Numerator
Summary
MATLAB routines

Implementation can take advantage of any symmetry in the coefficients.

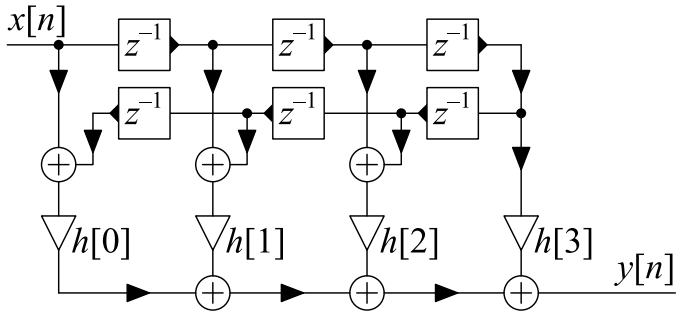
Linear phase filters are always FIR and have symmetric (or, more rarely, antisymmetric) coefficients.

$$H(z) = \sum_{m=0}^{M} h[m] z^{-m} \qquad h[M-m] = h[m]$$

$$= h\left[\frac{M}{2}\right] z^{-\frac{M}{2}} + \sum_{m=0}^{\frac{M}{2}-1} h[m]\left(z^{-m} + z^{m-M}\right) \qquad \text{[m even]}$$

For M even, we only need $\frac{M}{2}+1$ multiplies instead of $M+1$.
We still need M additions and M delays.

$M = 6$:



For M odd (no central coefficient), we only need $\frac{M+1}{2}$ multiplies.

Hardware Implementation

10: Digital Filter Structures
Direct Forms
Transposition
State Space +
Precision Issues
Coefficient Sensitivity
Cascaded Biquads
Pole-zero Pairing/Ordering
Linear Phase
Hardware Implementation
Allpass Filters
Lattice Stage +
Example
$A(z) \leftrightarrow D(z)$
Allpass Lattice
Lattice Filter
Lattice Example
Lattice Example Numerator
Summary
MATLAB routines

Software Implementation:

All that matters is the total number of multiplies and adds

Hardware Implementation:

Delay elements (z^{-1}) represent storage registers

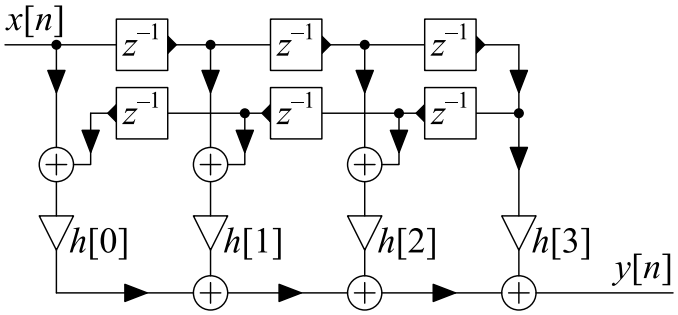
The maximum clock speed is limited by the number of sequential operations between registers

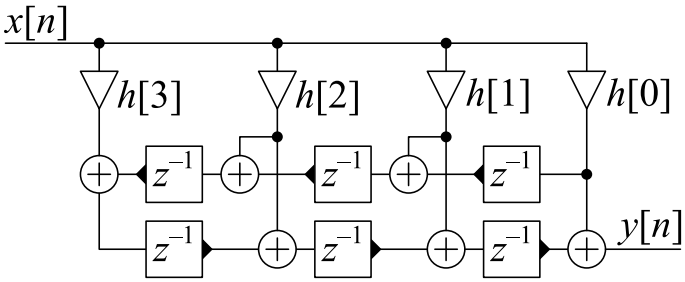
Example: Symmetric Linear Phase Filter

Direct form: Maximum sequential delay $= 4a + m$

Transpose form: Maximum sequential delay $= a + m$ ☺

a and m are the delays of adder and multiplier respectively





Allpass Filters

10: Digital Filter Structures
Direct Forms
Transposition
State Space +
Precision Issues
Coefficient Sensitivity
Cascaded Biquads
Pole-zero Pairing/Ordering
Linear Phase
Hardware Implementation
▷ Allpass Filters
Lattice Stage +
Example
$A(z) \leftrightarrow D(z)$
Allpass Lattice
Lattice Filter
Lattice Example
Lattice Example Numerator
Summary
MATLAB routines

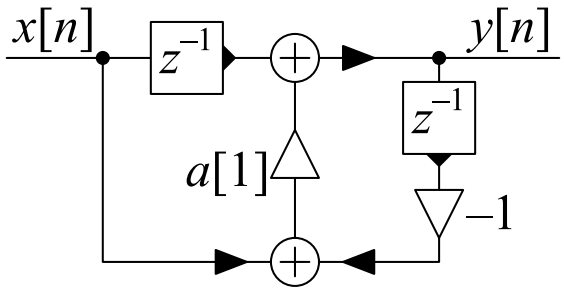
Allpass filters have mirror image numerator and denominator coefficients:

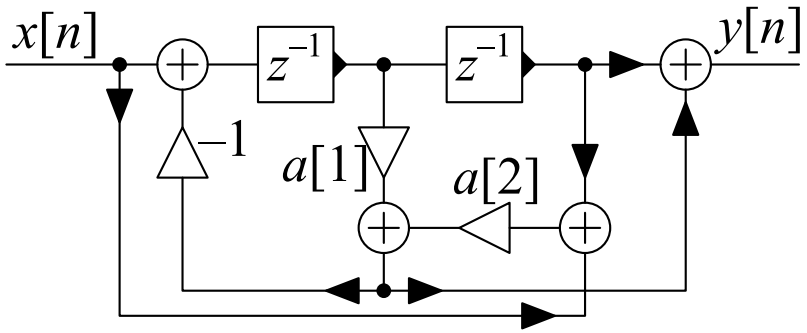
$$b[n] = a[N-n] \quad \Leftrightarrow \quad B(z) = z^{-N} A(z^{-1})$$

$$\Rightarrow \left|H(e^{j\omega})\right| \equiv 1 \forall \omega$$

There are several efficient structures, e.g.

- First Order: $H(z) = \frac{a[1]+z^{-1}}{1+a[1]z^{-1}}$
- Second Order: $H(z) = \frac{a[2]+a[1]z^{-1}+z^{-2}}{1+a[1]z^{-1}+a[2]z^{-2}}$





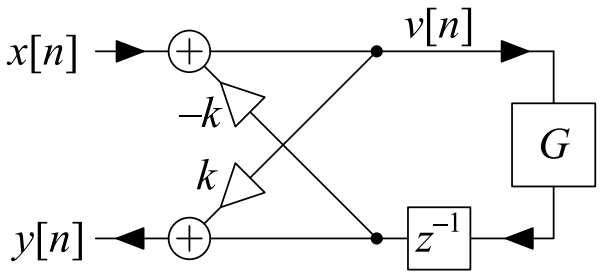
Allpass filters have a gain magnitude of 1 even with coefficient errors.

10: Digital Filter Structures
Direct Forms
Transposition
State Space +
Precision Issues
Coefficient Sensitivity
Cascaded Biquads
Pole-zero Pairing/Ordering
Linear Phase
Hardware Implementation
Allpass Filters
▷ Lattice Stage +
Example
$A(z) \leftrightarrow D(z)$
Allpass Lattice
Lattice Filter
Lattice Example
Lattice Example Numerator
Summary
MATLAB routines

Suppose G is allpass: $G(z) = \frac{z^{-N}A(z^{-1})}{A(z)}$

$$V(z) = X(z) - kGz^{-1}V(z)$$

$$\Rightarrow V(z) = \frac{1}{1+kGz^{-1}}X(z)$$



$$Y(z) = kV(z) + Gz^{-1}V(z) = \frac{k+z^{-1}G}{1+kGz^{-1}}X(z)$$

$$\frac{Y(z)}{X(z)} = \frac{kA(z)+z^{-N-1}A(z^{-1})}{A(z)+kz^{-N-1}A(z^{-1})} \triangleq \frac{z^{-(N+1)}D(z^{-1})}{D(z)}$$

Obtaining $\{d[n]\}$ from $\{a[n]\}$:

$$d[n] = \begin{cases} 1 & n = 0 \\ a[n] + ka[N+1-n] & 1 \le n \le N \\ k & n = N+1 \end{cases}$$

Obtaining $\{a[n]\}$ from $\{d[n]\}$:

$$k = d[N+1] \qquad a[n] = \frac{d[n]-kd[N+1-n]}{1-k^2}$$

If $G(z)$ is stable then $\frac{Y(z)}{X(z)}$ is stable if and only if $|k| < 1$ (see note)

[Proof of Stability Criterion]

We want to show that if $G(z)$ is a stable allpass filter then $\frac{Y(z)}{X(z)} = \frac{k+z^{-1}G(z)}{1+kz^{-1}G(z)}$ is stable if and only if $|k| < 1$.

We make use of a property of allpass filters (proved in a note in lecture 5) that if $G(z)$ is a stable allpass filter, then $|G(z)| \gtreqless 1$ according to whether $|z| \lesseqgtr 1$.

If z is a root of the denominator $1 + kz^{-1}G(z)$, then

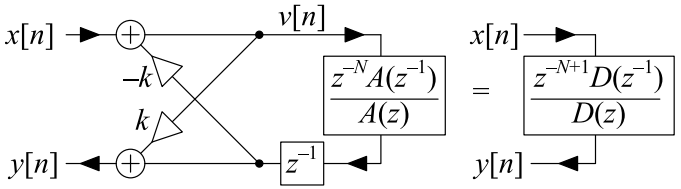
$$\begin{aligned} kz^{-1}G(z) &= -1 \\ \Rightarrow\ |k| \times |z^{-1}| \times |G(z)| &= 1 \\ \Rightarrow\ |k| &= \frac{|z|}{|G(z)|} \end{aligned}$$

It follows from the previously stated property of $G(z)$ that $|z| \lesseqgtr 1 \ \Leftrightarrow\ \frac{|z|}{|G(z)|} \lesseqgtr 1 \ \Leftrightarrow\ |k| \lesseqgtr 1$.

Example $A(z) \leftrightarrow D(z)$

10: Digital Filter Structures
Direct Forms
Transposition
State Space +
Precision Issues
Coefficient Sensitivity
Cascaded Biquads
Pole-zero Pairing/Ordering
Linear Phase
Hardware Implementation
Allpass Filters
Lattice Stage +
Example
▷ $A(z) \leftrightarrow D(z)$
Allpass Lattice
Lattice Filter
Lattice Example
Lattice Example Numerator
Summary
MATLAB routines

Suppose $N = 3$, $k = 0.5$ and
$A(z) = 1 + 4z^{-1} - 6z^{-2} + 10z^{-3}$



$A(z) \rightarrow D(z)$

	z^0	z^{-1}	z^{-2}	z^{-3}	z^{-4}
$A(z)$	1	4	−6	10	
$z^{-4}A(z^{-1})$		10	−6	4	1
$D(z) = A(z) + kz^{-4}A(z^{-1})$	1	9	−9	12	0.5

$D(z) \rightarrow A(z)$

	z^0	z^{-1}	z^{-2}	z^{-3}	z^{-4}
$D(z)$	1	9	−9	12	0.5
$k = d[N+1]$					0.5
$z^{-4}D(z^{-1})$	0.5	12	−9	9	1
$D(z) - kz^{-4}D(z^{-1})$	0.75	3	−4.5	7.5	0
$A(z) = \frac{D(z)-kz^{-4}D(z^{-1})}{1-k^2}$	1	4	−6	10	0

Allpass Lattice

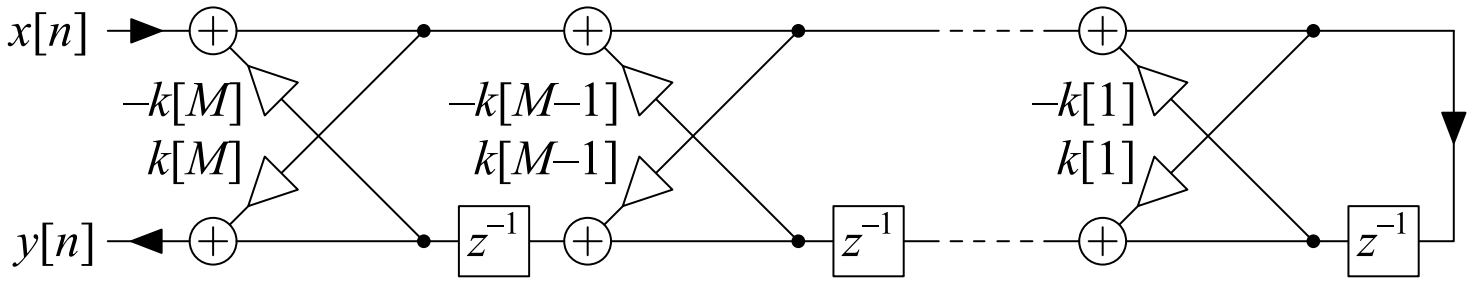
10: Digital Filter Structures
Direct Forms
Transposition
State Space +
Precision Issues
Coefficient Sensitivity
Cascaded Biquads
Pole-zero Pairing/Ordering
Linear Phase
Hardware Implementation
Allpass Filters
Lattice Stage +
Example
$A(z) \leftrightarrow D(z)$
▷ Allpass Lattice
Lattice Filter
Lattice Example
Lattice Example Numerator
Summary
MATLAB routines

We can implement any allpass filter $H(z) = \frac{z^{-M} A(z^{-1})}{A(z)}$ as a lattice filter with M stages:

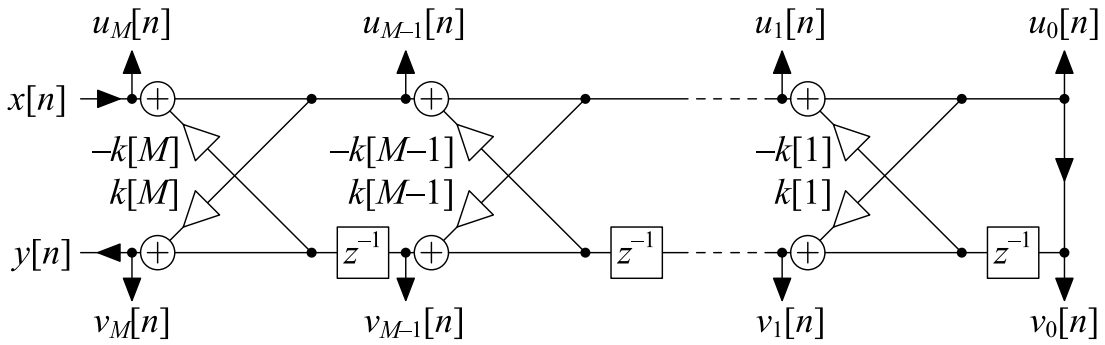
- Initialize $A_M(z) = A(z)$
- Repeat for $m = M : -1 : 1$
  - $k[m] = a_m[m]$
  - $a_{m-1}[n] = \frac{a_m[n] - k[m] a_m[m-n]}{1 - k^2[m]}$ for $0 \le n \le m-1$

    equivalently $A_{m-1}(z) = \frac{A_m(z) - k[m] z^{-m} A_m(z^{-1})}{1 - k^2[m]}$

$A(z)$ is stable iff $|k[m]| < 1$ for all m (good stability test)



Lattice Filter



Label outputs $u_m[n]$ and $v_m[n]$ and define $H_m(z) = \frac{V_m(z)}{U_m(z)} = \frac{z^{-m}A_m(z^{-1})}{A_m(z)}$

From earlier slide (slide 12):

$$\frac{U_{m-1}(z)}{U_m(z)} = \frac{1}{1+k[m]z^{-1}H_{m-1}(z)} = \frac{A_{m-1}(z)}{A_{m-1}(z)+k[m]z^{-m}A_{m-1}(z^{-1})} = \frac{A_{m-1}(z)}{A_m(z)}$$

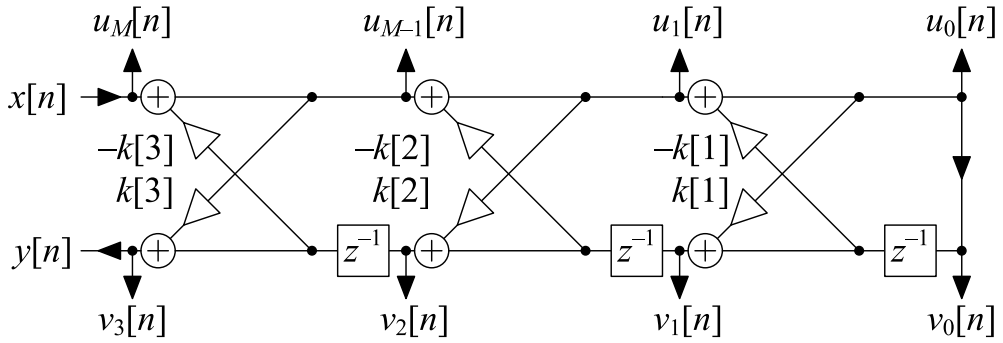
Hence:

$$\frac{U_m(z)}{X(z)} = \frac{A_m(z)}{A(z)} \qquad \text{and} \qquad \frac{V_m(z)}{X(z)} = \frac{U_m(z)}{X(z)} \times \frac{V_m(z)}{U_m(z)} = \frac{z^{-m}A_m(z^{-1})}{A(z)}$$

The numerator of $\frac{V_m(z)}{X(z)}$ is of order m so you can create any numerator of order M by summing appropriate multiples of $V_m(z)$:

$$w[n] = \sum_{m=0}^{M} c_m v_m[n] \quad \Rightarrow \quad W(z) = \frac{\sum_{m=0}^{M} c_m z^{-m} A_m(z^{-1})}{A(z)}$$

Lattice Example



$$A(z) = A_3(z) = 1 + 0.2z^{-1} - 0.23z^{-2} + 0.2z^{-3}$$

- $k[3] = 0.2 \Rightarrow a_2[\,] = \frac{[1,\ 0.2,\ -0.23] - 0.2[0.2,\ -0.23,\ 0.2]}{1 - 0.2^2} = [1,\ 0.256,\ -0.281]$
- $k[2] = -0.281 \Rightarrow a_1[\,] = \frac{[1,\ 0.256] + 0.281[-0.281,\ 0.256]}{1 - 0.281^2} = [1,\ 0.357]$
- $k[1] = 0.357 \Rightarrow a_0[\,] = 1$

$$\frac{V_0(z)}{X(z)} = \frac{1}{1+0.2z^{-1}-0.23z^{-2}+0.2z^{-3}} \qquad \frac{V_1(z)}{X(z)} = \frac{0.357+z^{-1}}{1+0.2z^{-1}-0.23z^{-2}+0.2z^{-3}}$$

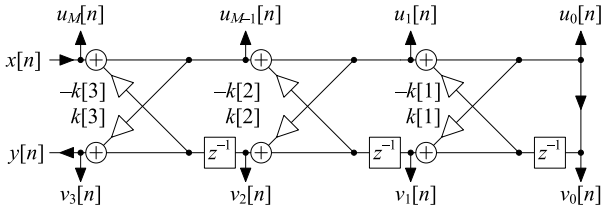
$$\frac{V_2(z)}{X(z)} = \frac{-0.281+0.256z^{-1}+z^{-2}}{1+0.2z^{-1}-0.23z^{-2}+0.2z^{-3}} \qquad \frac{V_3(z)}{X(z)} = \frac{0.2-0.23z^{-1}+0.2z^{-2}+z^{-3}}{1+0.2z^{-1}-0.23z^{-2}+0.2z^{-3}}$$

Add together multiples of $\frac{V_m(z)}{X(z)}$ to create an arbitrary $\frac{B(z)}{1+0.2z^{-1}-0.23z^{-2}+0.2z^{-3}}$

Lattice Example Numerator

Form a new output signal as $w[n] = \sum_{m=0}^{M} c_m v_m[n]$

$$W(z) = \sum_{m=0}^{M} c_m V_m(z) = \frac{B(z)}{1+0.2z^{-1}-0.23z^{-2}+0.2z^{-3}} X(z)$$



$$\frac{V_0(z)}{X(z)} = \frac{1}{1+0.2z^{-1}-0.23z^{-2}+0.2z^{-3}} \qquad \frac{V_1(z)}{X(z)} = \frac{0.357+z^{-1}}{1+0.2z^{-1}-0.23z^{-2}+0.2z^{-3}}$$

$$\frac{V_2(z)}{X(z)} = \frac{-0.281+0.256z^{-1}+z^{-2}}{1+0.2z^{-1}-0.23z^{-2}+0.2z^{-3}} \qquad \frac{V_3(z)}{X(z)} = \frac{0.2-0.23z^{-1}+0.2z^{-2}+z^{-3}}{1+0.2z^{-1}-0.23z^{-2}+0.2z^{-3}}$$

We have $\begin{pmatrix} b[0] \\ b[1] \\ b[2] \\ b[3] \end{pmatrix} = \begin{pmatrix} 1 & 0.357 & -0.281 & 0.2 \\ 0 & 1 & 0.256 & -0.23 \\ 0 & 0 & 1 & 0.2 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{pmatrix}$

Hence choose c_m as $\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} 1 & 0.357 & -0.281 & 0.2 \\ 0 & 1 & 0.256 & -0.23 \\ 0 & 0 & 1 & 0.2 \\ 0 & 0 & 0 & 1 \end{pmatrix}^{-1} \begin{pmatrix} b[0] \\ b[1] \\ b[2] \\ b[3] \end{pmatrix}$

Summary

10: Digital Filter Structures
Direct Forms
Transposition
State Space +
Precision Issues
Coefficient Sensitivity
Cascaded Biquads
Pole-zero Pairing/Ordering
Linear Phase
Hardware Implementation
Allpass Filters
Lattice Stage +
Example
$A(z) \leftrightarrow D(z)$
Allpass Lattice
Lattice Filter
Lattice Example
Lattice Example Numerator
▷ Summary
MATLAB routines

- Filter block diagrams
  - Direct forms
  - Transposition
  - State space representation
- Precision issues: coefficient error, arithmetic error
  - cascaded biquads
- Allpass filters
  - first and second order sections
- Lattice filters
  - Arbitrary allpass response
  - Arbitrary IIR response by summing intermediate outputs

For further details see Mitra: 8.

MATLAB routines

10: Digital Filter Structures
Direct Forms
Transposition
State Space +
Precision Issues
Coefficient Sensitivity
Cascaded Biquads
Pole-zero Pairing/Ordering
Linear Phase
Hardware Implementation
Allpass Filters
Lattice Stage +
Example $A(z) \leftrightarrow D(z)$
Allpass Lattice
Lattice Filter
Lattice Example
Lattice Example Numerator
Summary
▷ MATLAB routines

residuez	$\frac{b(z^{-1})}{a(z^{-1})} \rightarrow \sum_k \frac{r_k}{1-p_k z^{-1}}$
tf2sos,sos2tf	$\frac{b(z^{-1})}{a(z^{-1})} \leftrightarrow \prod_l \frac{b_{0,l}+b_{1,l}z^{-1}+b_{2,l}z^{-2}}{1+a_{1,l}z^{-1}+a_{2,l}z^{-2}}$
zp2sos,sos2zp	$\{z_m, p_k, g\} \leftrightarrow \prod_l \frac{b_{0,l}+b_{1,l}z^{-1}+b_{2,l}z^{-2}}{1+a_{\in 1,l}z^{-1}+a_{2,l}z^{-2}}$
zp2ss,ss2zp	$\{z_m, p_k, g\} \leftrightarrow \begin{cases} x' = Ax + Bu \\ y = Cx + Du \end{cases}$
tf2ss,ss2tf	$\frac{b(z^{-1})}{a(z^{-1})} \leftrightarrow \begin{cases} x' = Ax + Bu \\ y = Cx + Du \end{cases}$
poly	$\text{poly}(\mathbf{A}) = \det(z\mathbf{I}-\mathbf{A})$

11: Multirate Systems
▷ Systems
Multirate Systems
Building blocks
Resampling Cascades
Noble Identities
Noble Identities Proof
Upsampled z-transform
Downsampled z-transform
Downsampled Spectrum
Power Spectral Density +
Perfect Reconstruction
Commutators
Summary
MATLAB routines

11: Multirate Systems

Multirate Systems

11: Multirate Systems
▷ Multirate Systems
Building blocks
Resampling Cascades
Noble Identities
Noble Identities Proof
Upsampled z-transform
Downsampled z-transform
Downsampled Spectrum
Power Spectral Density +
Perfect Reconstruction
Commutators
Summary
MATLAB routines

Multirate systems include more than one sample rate

Why bother?:

- May need to change the sample rate
  e.g. Audio sample rates include 32, 44.1, 48, 96 kHz
- Can relax analog or digital filter requirements
  e.g. Audio DAC increases sample rate so that the reconstruction filter can have a more gradual cutoff
- Reduce computational complexity
  FIR filter length $\propto \frac{f_s}{\Delta f}$ where Δf is width of transition band
  Lower $f_s \Rightarrow$ shorter filter + fewer samples $\Rightarrow$computation $\propto f_s^2$

Building blocks

11: Multirate Systems
Multirate Systems
▷ Building blocks
Resampling Cascades
Noble Identities
Noble Identities Proof
Upsampled z-transform
Downsampled z-transform
Downsampled Spectrum
Power Spectral Density +
Perfect Reconstruction
Commutators
Summary
MATLAB routines

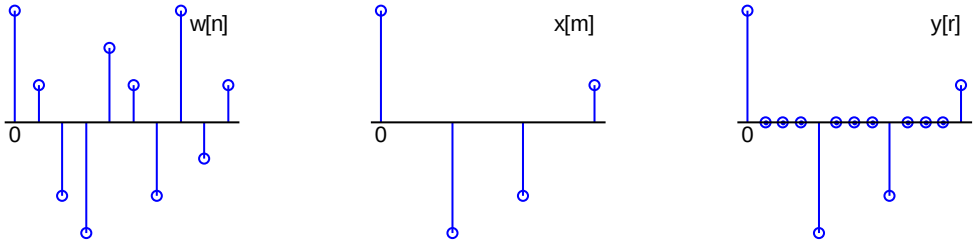
Downsample $x[n] \rightarrow \boxed{K{:}1} \rightarrow y[m]$ $\quad y[m] = x[Km]$

Upsample $u[m] \rightarrow \boxed{1{:}K} \rightarrow v[n]$ $\quad v[n] = \begin{cases} u\left[\frac{n}{K}\right] & K \mid n \\ 0 & \text{else} \end{cases}$

Example:

Downsample by 3 then upsample by 4

$w[n] \rightarrow \boxed{3{:}1} \rightarrow x[m] \rightarrow \boxed{1{:}4} \rightarrow y[r]$

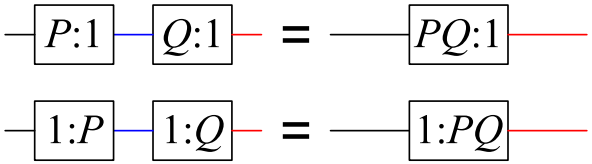


- We use different index variables (n, m, r) for different sample rates
- Use different colours for signals at different rates (sometimes)
- Synchronization: all signals have a sample at $n = 0$.

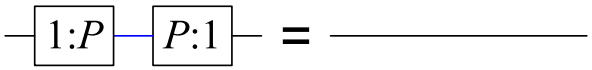
Resampling Cascades

11: Multirate Systems
Multirate Systems
Building blocks
▷ Resampling Cascades
Noble Identities
Noble Identities Proof
Upsampled z-transform
Downsampled z-transform
Downsampled Spectrum
Power Spectral Density
Perfect Reconstruction
Commutators
Summary
MATLAB routines

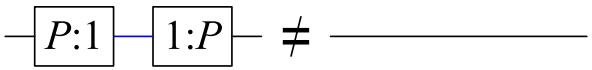
Successive downsamplers or upsamplers can be combined



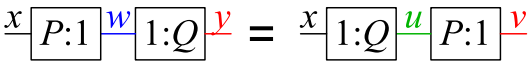
Upsampling can be exactly inverted



Downsampling destroys information permanently $\Rightarrow$ uninvertible



Resampling can be interchanged iff P and Q are coprime (surprising!)



Proof: Left side: $y[n] = w\left[\frac{1}{Q}n\right] = x\left[\frac{P}{Q}n\right]$ if $Q \mid n$ else $y[n] = 0$.

Right side: $v[n] = u\,[Pn] = x\left[\frac{P}{Q}n\right]$ if $Q \mid Pn$.

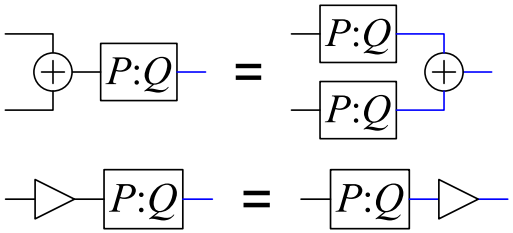
But $\{Q \mid Pn \;\Rightarrow\; Q \mid n\}$ iff P and Q are coprime.

[Note: $a \mid b$ means "a divides into b exactly"]

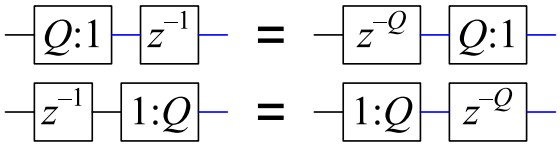
Noble Identities

11: Multirate Systems
Multirate Systems
Building blocks
Resampling Cascades
▷ Noble Identities
Noble Identities Proof
Upsampled z-transform
Downsampled z-transform
Downsampled Spectrum
Power Spectral Density +
Perfect Reconstruction
Commutators
Summary
MATLAB routines

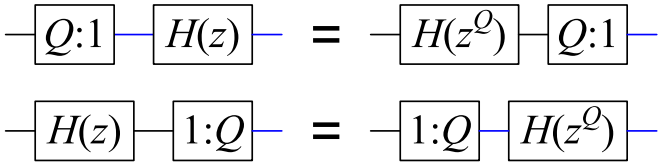
Resamplers commute with addition and multiplication



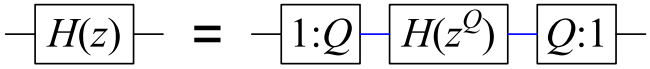
Delays must be multiplied by the resampling ratio



Noble identities:
Exchange resamplers and filters



Corrollary

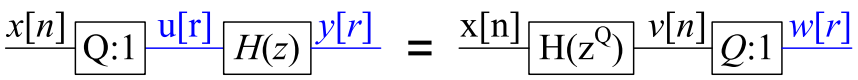


Example: $H(z) = h[0] + h[1]z^{-1} + h[2]z^{-2} + \cdots$
$H(z^3) = h[0] + h[1]z^{-3} + h[2]z^{-6} + \cdots$

Noble Identities Proof

11: Multirate Systems
Multirate Systems
Building blocks
Resampling Cascades
Noble Identities
Noble Identities
▷ Proof
Upsampled z-transform
Downsampled z-transform
Downsampled Spectrum
Power Spectral Density +
Perfect Reconstruction
Commutators
Summary
MATLAB routines

Define $h_Q[n]$ to be the impulse response of $H(z^Q)$.

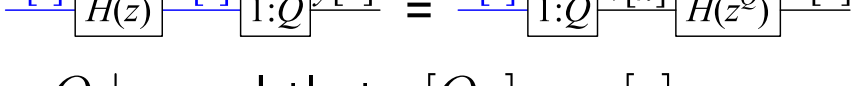


Assume that $h[r]$ is of length $M+1$ so that $h_Q[n]$ is of length $QM+1$.
We know that $h_Q[n] = 0$ except when $Q \mid n$ and that $h[r] = h_Q[Qr]$.

$$\begin{aligned} w[r] = v[Qr] &= \textstyle\sum_{s=0}^{QM} h_Q[s]x[Qr-s] \\ &= \textstyle\sum_{m=0}^{M} h_Q[Qm]x[Qr-Qm] = \sum_{m=0}^{M} h[m]x[Q(r-m)] \\ &= \textstyle\sum_{m=0}^{M} h[m]u[r-m] = y[r] \end{aligned}$$

☺

Upsampled Noble Identity:



We know that $v[n] = 0$ except when $Q \mid n$ and that $v[Qr] = x[r]$.

$$\begin{aligned} w[n] &= \textstyle\sum_{s=0}^{QM} h_Q[s]v[n-s] = \sum_{m=0}^{M} h_Q[Qm]v[n-Qm] \\ &= \textstyle\sum_{m=0}^{M} h[m]v[n-Qm] \end{aligned}$$

If $Q \nmid n$, then $v[n-Qm] = 0\ \forall m$ so $w[n] = 0 = y[n]$

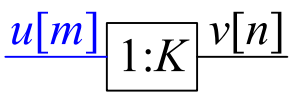
If $Q \mid n = Qr$, then $w[Qr] = \sum_{m=0}^{M} h[m]v[Qr-Qm]$
$= \sum_{m=0}^{M} h[m]x[r-m] = u[r] = y[Qr]$

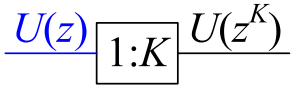
☺

Upsampled z-transform

11: Multirate Systems
Multirate Systems
Building blocks
Resampling Cascades
Noble Identities
Noble Identities Proof
Upsampled z-transform
Downsampled z-transform
Downsampled Spectrum
Power Spectral Density +
Perfect Reconstruction
Commutators
Summary
MATLAB routines

$$V(z) = \sum_n v[n] z^{-n} = \sum_{n \text{ s.t. } K|n} u[\tfrac{n}{K}] z^{-n}$$
$$= \sum_m u[m] z^{-Km} = U(z^K)$$





Spectrum: $V(e^{j\omega}) = U(e^{jK\omega})$

Spectrum is horizontally shrunk and replicated K times.

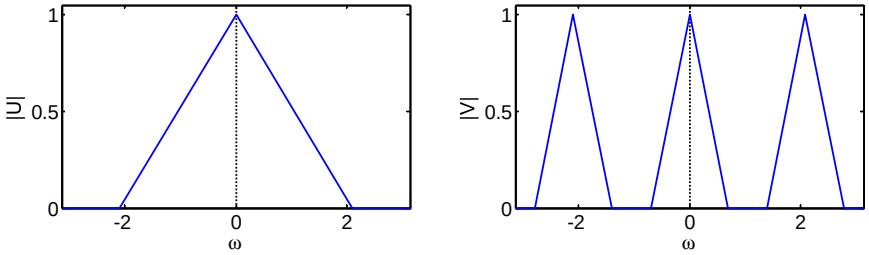
Total energy unchanged; power (= energy/sample) multiplied by $\frac{1}{K}$

Upsampling normally followed by a LP filter to remove images.

Example:

$K = 3$: three images of the original spectrum in all.

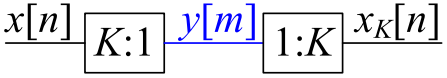
Energy unchanged: $\frac{1}{2\pi} \int \left|U(e^{j\omega})\right|^2 d\omega = \frac{1}{2\pi} \int \left|V(e^{j\omega})\right|^2 d\omega$



Downsampled z-transform

11: Multirate Systems
Multirate Systems
Building blocks
Resampling Cascades
Noble Identities
Noble Identities Proof
Upsampled z-transform
Downsampled z-transform
Downsampled Spectrum
Power Spectral Density
Perfect Reconstruction
Commutators
Summary
MATLAB routines

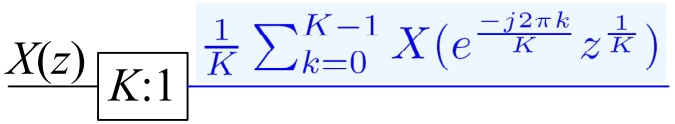
Define $c_K[n] = \delta_{K|n}[n] = \frac{1}{K}\sum_{k=0}^{K-1} e^{\frac{j2\pi kn}{K}}$



Now define $x_K[n] = \begin{cases} x[n] & K \mid n \\ 0 & K \nmid n \end{cases} = c_K[n]x[n]$

$$X_K(z) = \sum_n x_K[n]z^{-n} = \frac{1}{K}\sum_n \sum_{k=0}^{K-1} e^{\frac{j2\pi kn}{K}} x[n]z^{-n}$$
$$= \frac{1}{K}\sum_{k=0}^{K-1}\sum_n x[n]\left(e^{\frac{-j2\pi k}{K}}z\right)^{-n} = \frac{1}{K}\sum_{k=0}^{K-1} X(e^{\frac{-j2\pi k}{K}}z)$$

From previous slide:



$$X_K(z) = Y(z^K)$$
$$\Rightarrow Y(z) = X_K(z^{\frac{1}{K}}) = \frac{1}{K}\sum_{k=0}^{K-1} X(e^{\frac{-j2\pi k}{K}} z^{\frac{1}{K}})$$

Frequency Spectrum:

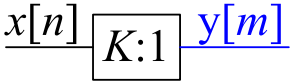
$$Y(e^{j\omega}) = \frac{1}{K}\sum_{k=0}^{K-1} X(e^{\frac{j(\omega-2\pi k)}{K}})$$
$$= \frac{1}{K}\left(X(e^{\frac{j\omega}{K}}) + X(e^{\frac{j\omega}{K}-\frac{2\pi}{K}}) + X(e^{\frac{j\omega}{K}-\frac{4\pi}{K}}) + \cdots\right)$$

Average of K aliased versions, each expanded in ω by a factor of K.
Downsampling is normally preceded by a LP filter to prevent aliasing.

Downsampled Spectrum

11: Multirate Systems
Multirate Systems
Building blocks
Resampling Cascades
Noble Identities
Noble Identities Proof
Upsampled z-transform
Downsampled z-transform
▷ Downsampled Spectrum
Power Spectral Density +
Perfect Reconstruction
Commutators
Summary
MATLAB routines

$$Y(e^{j\omega}) = \frac{1}{K}\sum_{k=0}^{K-1} X(e^{\frac{j(\omega-2\pi k)}{K}})$$

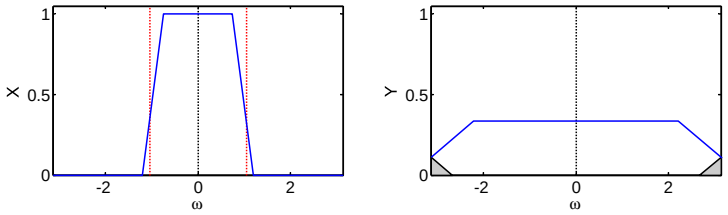


Example 1:

$K = 3$

Not quite limited to $\pm\frac{\pi}{K}$

Shaded region shows aliasing



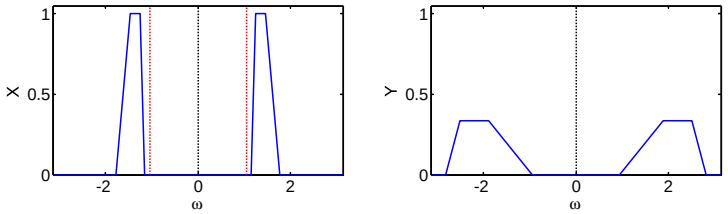
Energy decreases: $\frac{1}{2\pi}\int \left|Y(e^{j\omega})\right|^2 d\omega \approx \frac{1}{K} \times \frac{1}{2\pi}\int \left|X(e^{j\omega})\right|^2 d\omega$

Example 2:

$K = 3$

Energy all in $\frac{\pi}{K} \le |\omega| < 2\frac{\pi}{K}$

No aliasing: ☺



No aliasing: If all energy is in $r\frac{\pi}{K} \le |\omega| < (r+1)\frac{\pi}{K}$ for some integer r

Normal case ($r = 0$): If all energy in $0 \le |\omega| \le \frac{\pi}{K}$

Downsampling: Total energy multiplied by $\approx \frac{1}{K}$ ($= \frac{1}{K}$ if no aliasing)

Average power $\approx$ unchanged ($=$ energy/sample)

11: Multirate Systems
Multirate Systems
Building blocks
Resampling Cascades
Noble Identities
Noble Identities Proof
Upsampled z-transform
Downsampled z-transform
Downsampled Spectrum
▷ Power Spectral Density
Perfect Reconstruction
Commutators
Summary
MATLAB routines

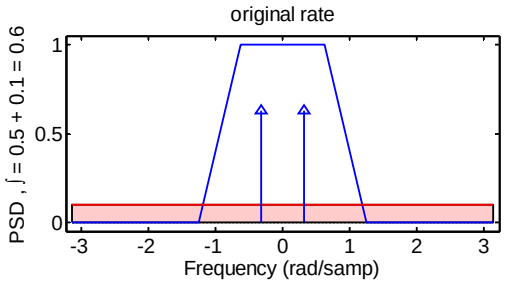
Example: Signal in $\omega \in \pm 0.4\pi$ + Tone @ $\omega = \pm 0.1\pi$ + White noise

Power = Energy/sample = Average PSD

$$= \frac{1}{2\pi} \int_{-\pi}^{\pi} \mathrm{PSD}(\omega) d\omega \ = 0.6$$

Component powers:

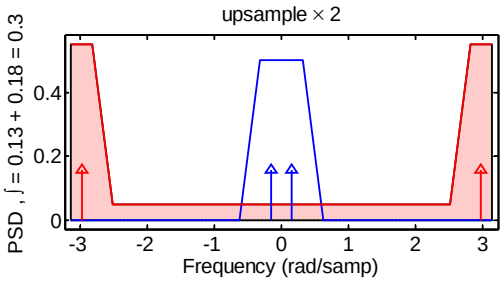
Signal $= 0.3$, Tone $= 0.2$, Noise $= 0.1$

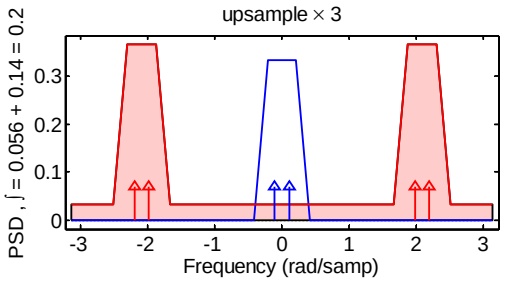


Upsampling:

Same energy per second

$\Rightarrow$ Power is $\div K$

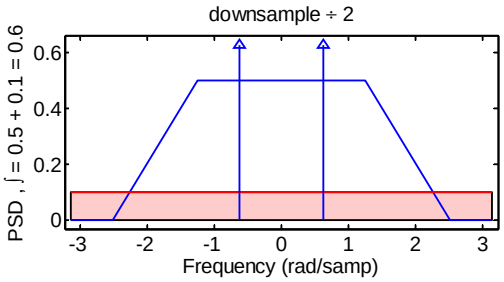


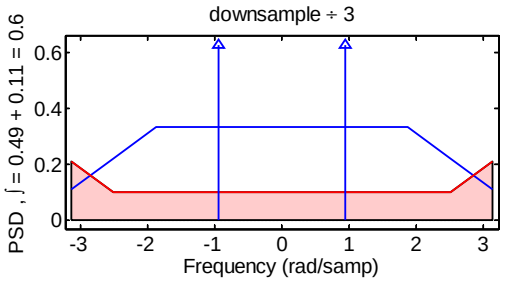


Downsampling:

Average power is unchanged.

$\exists$ aliasing in the $\div 3$ case.





[Power Spectral Density (1)]

The energy of a spectrum is $E_x = \sum_{-\infty}^{+\infty} |x[n]|^2$ and its power is $P_x = \lim_{N\to\infty} \frac{1}{2N+1} \sum_{-N}^{+N} |x[n]|^2$. The energy, E_x, is the total energy in all samples while the power, P_x, is the average energy per sample. If the finite signal $x_N[n]$ is defined as $x_N[n] = \begin{cases} x[n] & |n| \leq N \\ 0 & |n| > N \end{cases}$, then the power spectral density (PSD) is given by $S_{xx}\left(e^{j\omega}\right) = \lim_{N\to\infty} \frac{1}{2N+1} \left|X_N\left(e^{j\omega}\right)\right|^2$. From Parseval's theorem, P_x is the average value of $S_{xx}\left(e^{j\omega}\right)$ or, equivalently, $P_x = \frac{1}{2\pi} \int_{-\pi}^{\pi} S_{xx}\left(e^{j\omega}\right) d\omega$.

The signal on the previous slide has three components: (i) a signal component with a power of 0.3 and a trapezoidal PSD with a width of $\pm 0.4\pi$, (ii) a tonal component with a power of 0.2 whose PSD consists of two delta functions and (iii) a white noise component of power 0.1 whose PSD is constant at 0.1. The tonal component might arise from a time-domain waveform $\sqrt{0.4}\cos\left(0.1\pi n + \phi\right)$ where ϕ is arbitrary and does not affect the PSD.

Upsampling by K inserts additional zero-valued samples and so does not affect E_x but, since there are now K times as many samples, P_x is divided by K. The original periodic PSD is shrunk horozontally by a factor of K which means that there are now K images of the original PSD at spacings of $\Delta\omega = \frac{2\pi}{K}$. So, for example, when $K = 2$, the central trapezoidal component has a maximum height of 0.5 and a width of $\pm 0.2\pi$ and there is a second, identical, trapezoidal component shifted by $\Delta\omega = \frac{2\pi}{K} = \pi$. When K is an even number, one of the images will be centred on $\omega = \pi$ and so will wrap around from $+\pi$ to $-\pi$. The power of each image is multiplied by K^{-2} but, since there are K images, the total power is multiplied by K^{-1}. For the white noise, the images all overlap (and add in power), so the white noise PSD amplitude is multiplied by K^{-1}. Finally, the amplitudes of the delta functions are multiplied by K^{-2} so that the total power of all K images is multiplied by K^{-1}.

[Power Spectral Density (2)]

Downsampling by K deletes samples but leaves the average power of the remaining ones unchanged. Thus the total power of the downsampled spectra remains at 0.6. The downsampled PSD is the average of K shifted versions of the original PSD that have been expanded horizontally by a factor of K. The white noise component is the average of K identical expanded but attenuated versions of itself and so its PSD amplitude remains at 0.1. The power of a tonal components is unchanged and so its amplitude is also unchanged.
When downsampling by a factor of $K = 3$, the original width of the trapezoidal component expands from $\pm 0.4\pi$ to $\pm 1.2\pi$ which exceeds the $\pm\pi$ range of the graph. Thus, as ω approaches π, the PSD of the signal component is decreasing with ω but has not reached 0 at $\omega = \pi$. This portion of the trapezium wraps around to $\omega = -\pi$ and gives rise to the little triangle of additional noise in the range $-\pi < \omega < -0.8\pi$ where it adds onto the white noise component. In a similar way, the portion of the trapezium that overflows the left edge of the graph gives rise to additional noise at the right of the graph in the range $0.8\pi < \omega < \pi$.

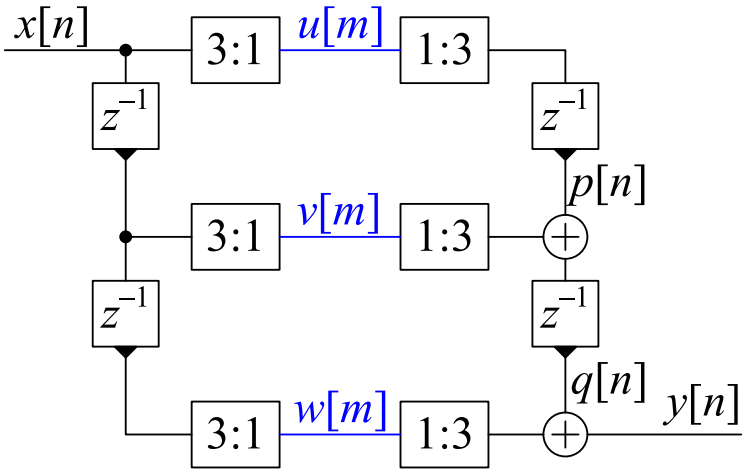
Summary of Spectral Density Changes: Width × Height (×Images)

Energy and Power Spectral Densities	Energy Spectral Density		Power Spectral Density	
	Up: $1:K$	Down: $K:1$	Up: $1:K$	Down: $K:1$
Alias-free block	$K^{-1} \times 1\,(\times K)$	$K \times K^{-2}$	$K^{-1} \times K^{-1}\,(\times K)$	$K \times K^{-1}$
Tone: $\delta(\omega - \omega_0)$	$1 \times K^{-1}\ (\times K)$	$1 \times K^{-1}$	$1 \times K^{-2}\ (\times K)$	1×1
White Noise	1×1	$1 \times K^{-1}$	$1 \times K^{-1}$	1×1
Integral $\int d\omega$	$\times 1$	$\approx \times K^{-1}$	$\times K^{-1}$	$\approx \times 1$

Perfect Reconstruction

11: Multirate Systems
Multirate Systems
Building blocks
Resampling Cascades
Noble Identities
Noble Identities Proof
Upsampled z-transform
Downsampled z-transform
Downsampled Spectrum
Power Spectral Density +
Perfect Reconstruction
Commutators
Summary
MATLAB routines

$x[n]$	cdefghijklmn
$u[m]$	c f i l
$p[n]$	-c--f--i--l
$v[m]$	b e h k
$q[n]$	-bc-ef-hi-kl
$w[m]$	a d g j
$y[n]$	abcdefghijkl



Input sequence $x[n]$ is split into three streams at $\frac{1}{3}$ the sample rate:

$$u[m] = x[3m],\ v[m] = x[3m-1],\ w[m] = x[3m-2]$$

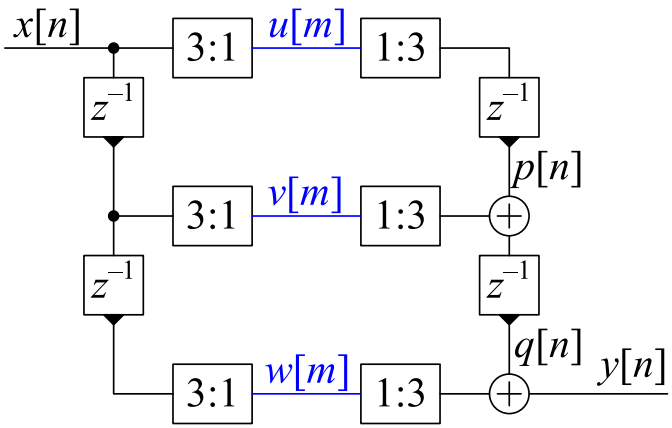
Following upsampling, the streams are aligned by the delays and then added to give:

$$y[n] = x[n-2]$$

Perfect Reconstruction: output is a delayed scaled replica of the input

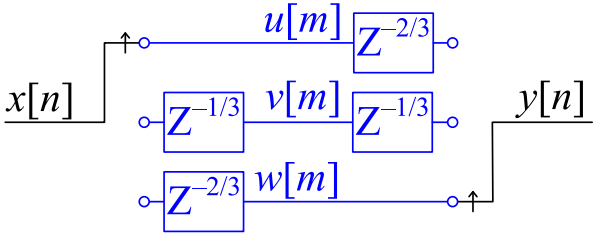
Commutators

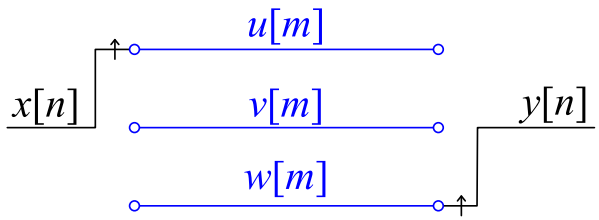
11: Multirate Systems
Multirate Systems
Building blocks
Resampling Cascades
Noble Identities
Noble Identities Proof
Upsampled z-transform
Downsampled z-transform
Downsampled Spectrum
Power Spectral Density +
Perfect Reconstruction
▷ Commutators
Summary
MATLAB routines



$x[n]$	cdefghijklmn
$u[m]$	c f i l
$v[m]$	b e h k
$w[m]$	a d g j
$v[m+\frac{1}{3}]$	e h k l
$w[m+\frac{2}{3}]$	d g j m
$y[n]$	abcdefghijkl

The combination of delays and downsamplers can be regarded as a commutator that distributes values in sequence to u, w and v.
Fractional delays, $z^{-\frac{1}{3}}$ and $z^{-\frac{2}{3}}$ are needed to synchronize the streams.
The output commutator takes values from the streams in sequence.
For clarity, we omit the fractional delays and regard each terminal, ○, as holding its value until needed. Initial commutator position has zero delay.





The commutator direction is against the direction of the z^{-1} delays.

Summary

11: Multirate Systems
Multirate Systems
Building blocks
Resampling Cascades
Noble Identities
Noble Identities Proof
Upsampled z-transform
Downsampled z-transform
Downsampled Spectrum
Power Spectral Density +
Perfect Reconstruction
Commutators
▷ Summary
MATLAB routines

- Multirate Building Blocks
  - Upsample: $X(z) \overset{1:K}{\rightarrow} X(z^K)$
    Invertible, Inserts $K-1$ zeros between samples
    Shrinks and replicates spectrum
    Follow by LP filter to remove images
  - Downsample: $X(z) \overset{K:1}{\rightarrow} \frac{1}{K}\sum_{k=0}^{K-1} X(e^{\frac{-j2\pi k}{K}} z^{\frac{1}{K}})$
    Destroys information and energy, keeps every K^{th} sample
    Expands and aliasses the spectrum
    Spectrum is the average of K aliased expanded versions
    Precede by LP filter to prevent aliases
- Equivalences
  - Noble Identities: $H(z) \longleftrightarrow H(z^K)$
  - Interchange $P:1$ and $1:Q$ iff $P$and Q coprime
- Commutators
  - Combine delays and down/up sampling

For further details see Mitra: 13.

MATLAB routines

11: Multirate Systems
Multirate Systems
Building blocks
Resampling Cascades
Noble Identities
Noble Identities Proof
Upsampled z-transform
Downsampled z-transform
Downsampled Spectrum
Power Spectral Density +
Perfect Reconstruction
Commutators
Summary
▷ MATLAB routines

resample	change sampling rate

12: Polyphase Filters
Heavy Lowpass filtering
Maximum Decimation Frequency
Polyphase decomposition
Downsampled Polyphase Filter
Polyphase Upsampler
Complete Filter
Upsampler Implementation
Downsampler Implementation
Summary

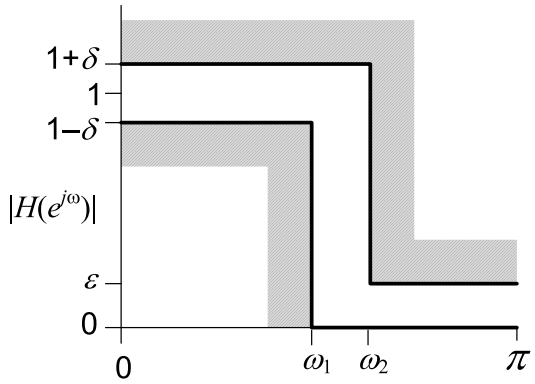
12: Polyphase Filters

Heavy Lowpass filtering

12: Polyphase Filters
- Heavy Lowpass filtering
- Maximum Decimation Frequency
- Polyphase decomposition
- Downsampled Polyphase Filter
- Polyphase Upsampler
- Complete Filter
- Upsampler Implementation
- Downsampler Implementation
- Summary

Filter Specification:

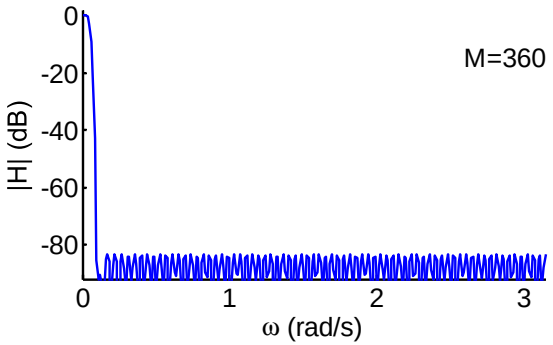
Sample Rate: 20 kHz
Passband edge: 100 Hz ($\omega_1 = 0.03$)
Stopband edge: 300 Hz ($\omega_2 = 0.09$)
Passband ripple: ± 0.05 dB ($\delta = 0.006$)
Stopband Gain: -80 dB ($\epsilon = 0.0001$)

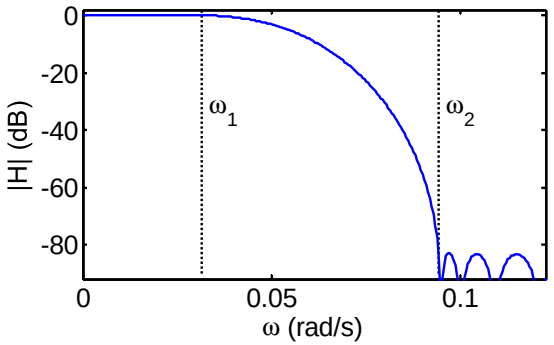


This is an extreme filter because the cutoff frequency is only 1% of the Nyquist frequency.

Symmetric FIR Filter:

Design with Remez-exchange algorithm
Order = 360

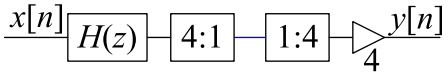


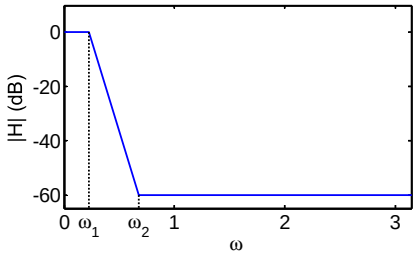


Maximum Decimation Frequency

12: Polyphase Filters
Heavy Lowpass filtering
Maximum Decimation
▷ Frequency
Polyphase decomposition
Downsampled Polyphase Filter
Polyphase Upsampler
Complete Filter
Upsampler Implementation
Downsampler Implementation
Summary

If a filter passband occupies only a small fraction of $[0, \pi]$, we can downsample then upsample without losing information.



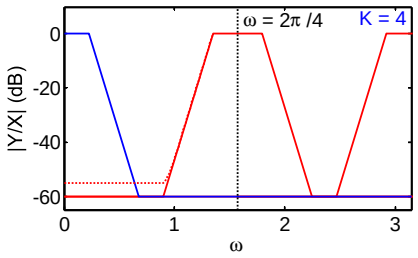


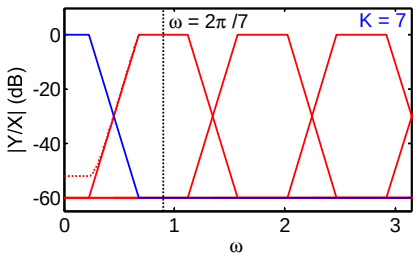
Downsample: aliased components at offsets of $\frac{2\pi}{K}$ are almost zero because of $H(z)$

Upsample: Images spaced at $\frac{2\pi}{K}$ can be removed using another low pass filter

To avoid aliasing in the passband, we need

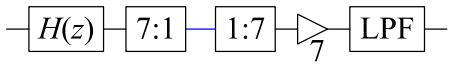
$$\frac{2\pi}{K} - \omega_2 \geq \omega_1 \quad \Rightarrow \quad K \leq \frac{2\pi}{\omega_1 + \omega_2}$$





Centre of transition band must be $\leq$ intermediate Nyquist freq, $\frac{\pi}{K}$

We must add a lowpass filter to remove the images:



Passband noise = noise floor at output of $H(z)$ plus $10\log_{10}(K-1)$ dB.

Polyphase decomposition

12: Polyphase Filters
Heavy Lowpass filtering
Maximum Decimation Frequency
Polyphase decomposition
Downsampled Polyphase Filter
Polyphase Upsampler
Complete Filter
Upsampler Implementation
Downsampler Implementation
Summary

For our filter: original Nyquist frequency $= 10$ kHz and transition band centre is at 200 Hz so we can use $K = 50$.

We will split $H(z)$ into K filters each of order $R-1$. For convenience, assume $M+1$ is a multiple of K (else zero-pad $h[n]$).

Example: $M = 399,\ K = 50 \Rightarrow R = \frac{M+1}{K} = 8$

$$H(z) = \sum_{m=0}^{M} h[m] z^{-m}$$

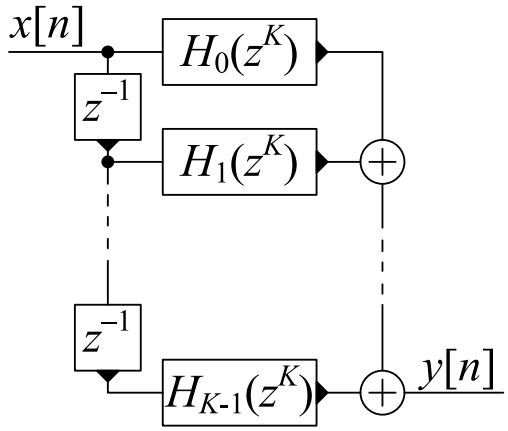
$$= \sum_{m=0}^{K-1} h[m] z^{-m} + \sum_{m=0}^{K-1} h[m+K] z^{-(m+K)} + \cdots \qquad [R \text{ terms}]$$

$$= \sum_{r=0}^{R-1} \sum_{m=0}^{K-1} h[m+Kr] z^{-m-Kr}$$

$$= \sum_{m=0}^{K-1} z^{-m} \sum_{r=0}^{R-1} h_m[r] z^{-Kr}$$

where $h_m[r] = h[m+Kr]$

$$= \sum_{m=0}^{K-1} z^{-m} H_m\left(z^K\right)$$



Example: $M = 399,\ K = 50,\ R = 8$

$h_3[r] = [h[3], h[53], \cdots, h[303], h[353]]$

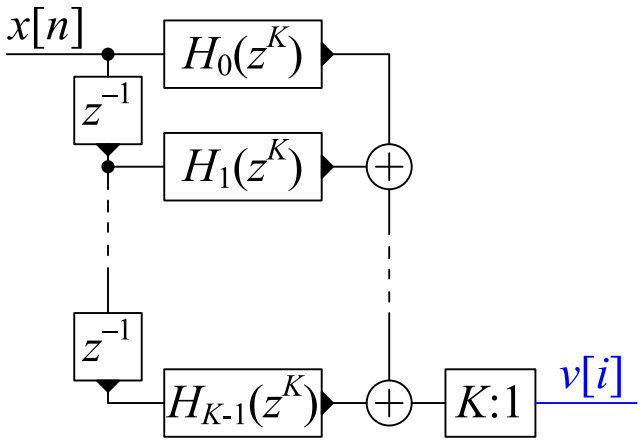
This is a polyphase implementation of the filter $H(z)$

Downsampled Polyphase Filter

12: Polyphase Filters
Heavy Lowpass filtering
Maximum Decimation Frequency
Polyphase decomposition
Downsampled Polyphase Filter
Polyphase Upsampler
Complete Filter
Upsampler Implementation
Downsampler Implementation
Summary

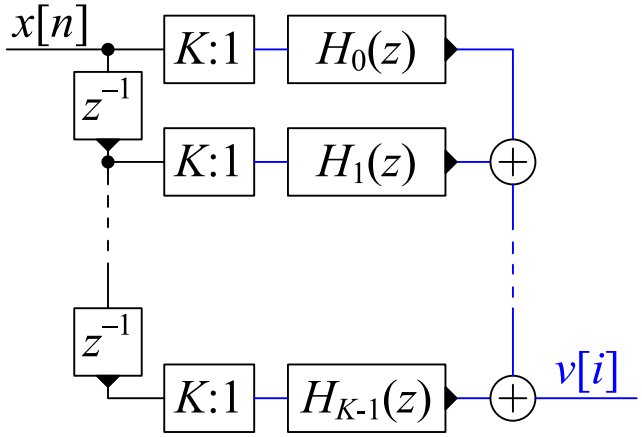
$H(z)$ is low pass so we downsample its output by K without aliasing.

The number of multiplications per input sample is $M + 1 = 400$.



Using the Noble identities, we can move the resampling back through the adders and filters. $H_m(z^K)$ turns into $H_m(z)$ at a lower sample rate.

We still perform 400 multiplications but now only once for every K input samples.



Multiplications per input sample $= 8$ (down by a factor of 50 ☺) but $v[n]$ has the wrong sample rate (☹).

Polyphase Upsampler

12: Polyphase Filters
Heavy Lowpass filtering
Maximum Decimation Frequency
Polyphase decomposition
Downsampled Polyphase Filter
▷ Polyphase Upsampler
Complete Filter
Upsampler Implementation
Downsampler Implementation
Summary

To restore sample rate: upsample and then lowpass filter to remove images

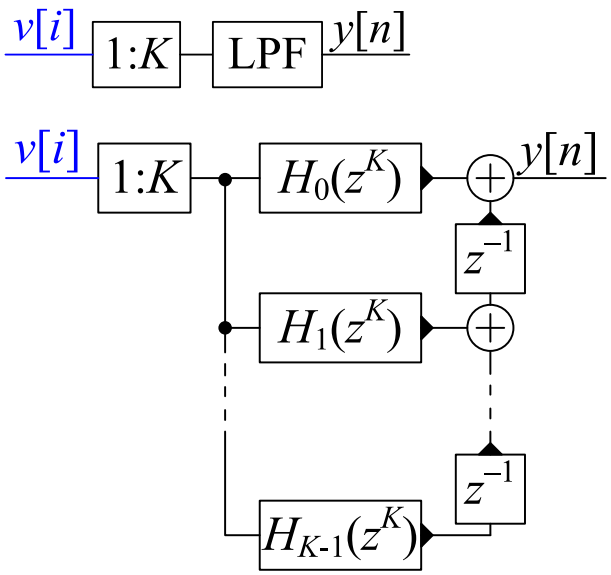
We can use the same lowpass filter, $H(z)$, in polyphase form:

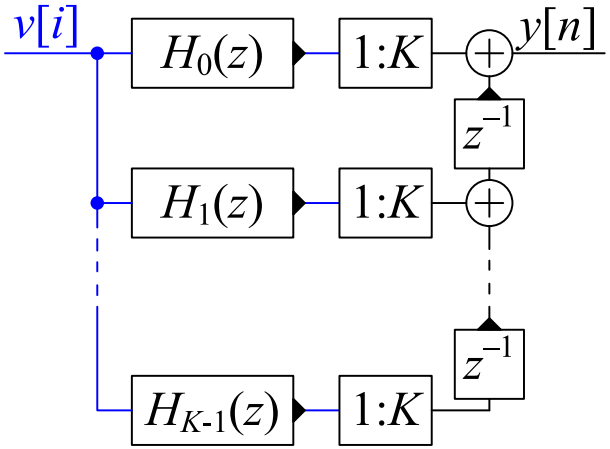
$$\sum_{m=0}^{K-1} z^{-m} \sum_{r=0}^{R-1} h_m[r] z^{-Kr}$$

This time we put the delay z^{-m} after the filters.
Multiplications per output sample = 400

Using the Noble identities, we can move the resampling forwards through the filters. $H_m(z^K)$ turns into $H_m(z)$ at a lower sample rate.

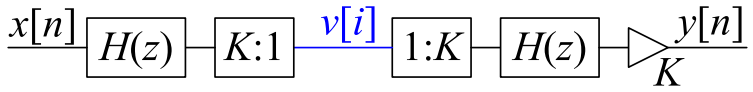
Multiplications per output sample = 8 (down by a factor of 50 ☺).





Complete Filter

12: Polyphase Filters
Heavy Lowpass filtering
Maximum Decimation Frequency
Polyphase decomposition
Downsampled Polyphase Filter
Polyphase Upsampler
▷ Complete Filter
Upsampler Implementation
Downsampler Implementation
Summary

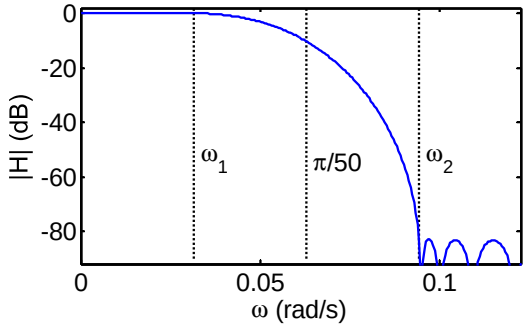


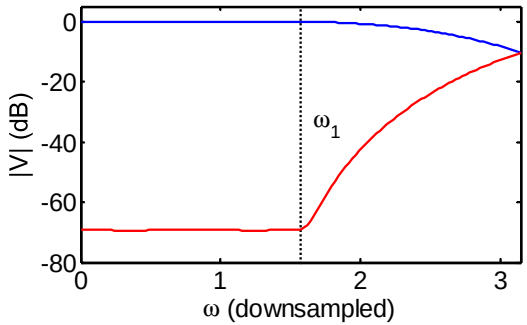
The overall system implements:
Need an extra gain of K to compensate for the downsampling energy loss.

Filtering at downsampled rate requires 16 multiplications per input sample (8 for each filter). Reduced by $\frac{K}{2}$ from the original 400.

$H(e^{j\omega})$ reaches -10 dB at the downsampler Nyquist frequency of $\frac{\pi}{K}$.
Spectral components $> \frac{\pi}{K}$ will be aliased down in frequency in $V(e^{j\omega})$.

For $V(e^{j\omega})$, passband gain (blue curve) follows the same curve as $X(e^{j\omega})$.
Noise arises from K aliased spectral intervals.
Unit white noise in $X(e^{j\omega})$ gives passband noise floor at -69 dB (red curve) even though stop band ripple is below -83 dB (due to $K-1$ aliased stopband copies).



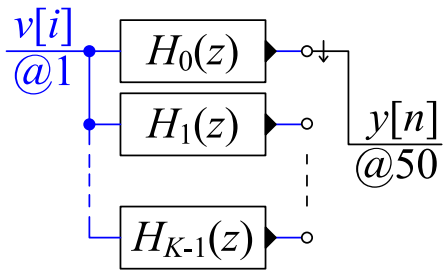


Upsampler Implementation

12: Polyphase Filters
Heavy Lowpass filtering
Maximum Decimation Frequency
Polyphase decomposition
Downsampled Polyphase Filter
Polyphase Upsampler
Complete Filter
▷ Upsampler Implementation
Downsampler Implementation
Summary

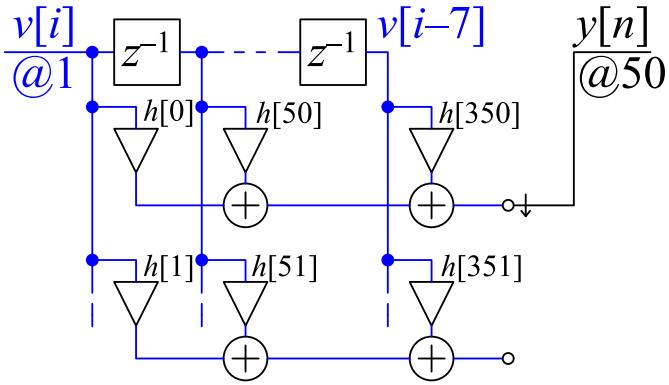
We can represent the upsampler compactly using a commutator. Sample $y[n]$ comes from $H_k(z)$ where $k = n \mod K$.

["@f" indicates the sample rate]

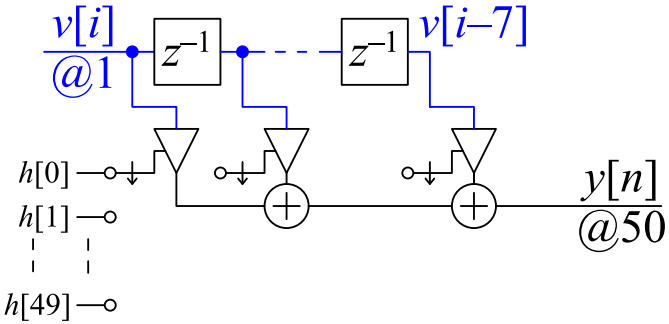


$H_0(z)$ comprises a sequence of 7 delays, 7 adders and 8 gains.

We can share the delays between all 50 filters.



We can also share the gains and adders between all 50 filters and use commutators to switch the coefficients.



We now need 7 delays, 7 adders and 8 gains for the entire filter.

Downsampler Implementation

12: Polyphase Filters
Heavy Lowpass filtering
Maximum Decimation Frequency
Polyphase decomposition
Downsampled Polyphase Filter
Polyphase Upsampler
Complete Filter
Upsampler Implementation
▷ Downsampler Implementation
Summary

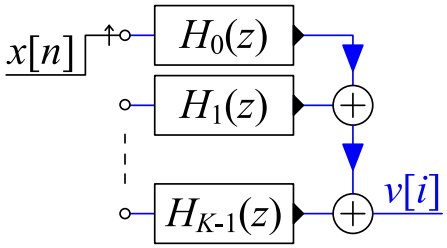
We can again use a commutator. The outputs from all 50 filters are added together to form $v[i]$.

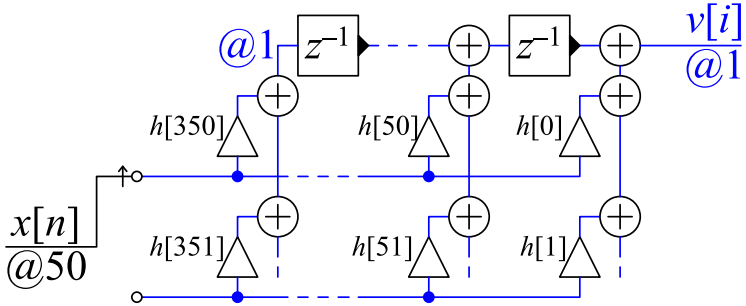
We use the transposed form of $H_m(z)$ because this will allow us to share components.

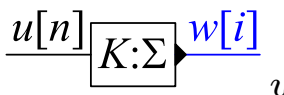
We can sum the outputs of the gain elements using an accumulator which sums blocks of K samples.

Now we can share all the components and use commutators to switch the gain coefficients.

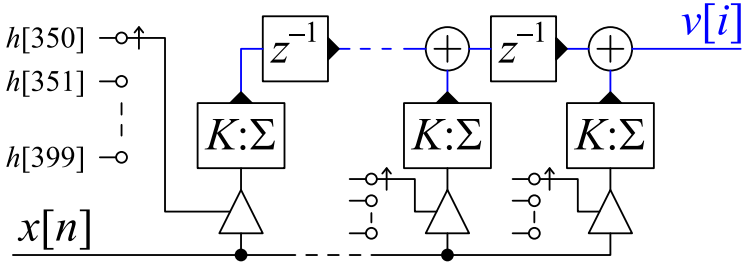
We need 7 delays, 7 adders, 8 gains and 8 accumulators in total.







$$w[i] = \sum_{r=0}^{K-1} u[Ki - r]$$



Summary

12: Polyphase Filters
Heavy Lowpass filtering
Maximum Decimation Frequency
Polyphase decomposition
Downsampled Polyphase Filter
Polyphase Upsampler
Complete Filter
Upsampler Implementation
Downsampler Implementation
▷ Summary

- Filtering should be performed at the lowest possible sample rate
  - reduce filter computation by K
  - actual saving is only $\frac{K}{2}$ because you need a second filter
  - downsampled Nyquist frequency $\geq \max(\omega_{\text{passband}}) + \frac{\Delta\omega}{2}$
- Polyphase decomposition: split $H(z)$ as $\sum_{m=0}^{K-1} z^{-m} H_m(z^K)$
  - each $H_m(z^K)$ can operate on subsampled data
  - combine the filtering and down/up sampling
- Noise floor is higher because it arises from K spectral intervals that are aliased together by the downsampling.
- Share components between the K filters
  - multiplier gain coefficients switch at the original sampling rate
  - need a new component: accumulator/downsampler $(K : \Sigma)$

For further details see Harris 5.

13: Resampling Filters
▷ Filters
Resampling
Halfband Filters
Dyadic 1:8 Upsampler
Rational Resampling
Arbitrary Resampling
+
Polynomial Approximation
Farrow Filter +
Summary
MATLAB routines

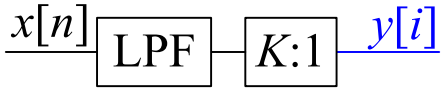
13: Resampling Filters

Resampling

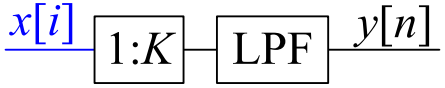
13: Resampling Filters
▷ Resampling
Halfband Filters
Dyadic 1:8 Upsampler
Rational Resampling
Arbitrary Resampling + Polynomial Approximation
Farrow Filter +
Summary
MATLAB routines

Suppose we want to change the sample rate while preserving information:
e.g. Audio 44.1 kHz$\leftrightarrow$$48$ kHz$\leftrightarrow$$96$ kHz

Downsample:
LPF to new Nyquist bandwidth: $\omega_0 = \frac{\pi}{K}$



Upsample:
LPF to old Nyquist bandwidth: $\omega_0 = \frac{\pi}{K}$



Rational ratio: $f_s \times \frac{P}{Q}$
LPF to lower of old and new Nyquist bandwidths: $\omega_0 = \frac{\pi}{\max(P,Q)}$

- Polyphase decomposition reduces computation by $K = \max(P, Q)$.
- The transition band centre should be at the Nyquist frequency, $\omega_0 = \frac{\pi}{K}$
- Filter order $M \approx \frac{d}{3.5\Delta\omega}$ where d is stopband attenuation in dB and $\Delta\omega$ is the transition bandwidth (Remez-exchange estimate).
- Fractional semi-Transition bandwidth, $\alpha = \frac{\Delta\omega}{2\omega_0}$, is typically fixed.
  e.g. $\alpha = 0.05 \quad \Rightarrow \quad M \approx \frac{dK}{7\pi\alpha} = 0.9dK$ (where $\omega_0 = \frac{\pi}{K}$)

Halfband Filters

13: Resampling Filters
Resampling
▷ Halfband Filters
Dyadic 1:8 Upsampler
Rational Resampling
Arbitrary Resampling +
Polynomial Approximation
Farrow Filter +
Summary
MATLAB routines

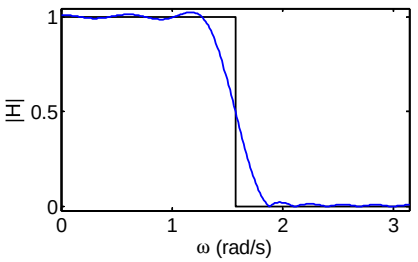
If $K = 2$ then the new Nyquist frequency is $\omega_0 = \frac{\pi}{2}$.

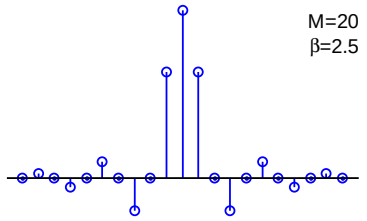
We multiply ideal response $\frac{\sin \omega_0 n}{\pi n}$ by a Kaiser window. All even numbered points are zero except $h[0] = 0.5$.

If $4 \mid M$ and we make the filter causal ($\times z^{-\frac{M}{2}}$),

$$H(z) = 0.5 z^{-\frac{M}{2}} + z^{-1} \sum_{r=0}^{\frac{M}{2}-1} h_1[r] z^{-2r}$$

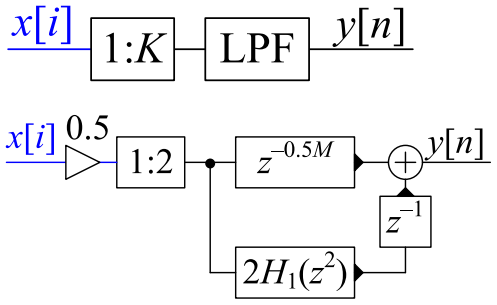
where $h_1[r] = h[2r + 1 - \frac{M}{2}]$



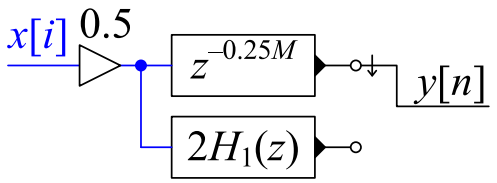


Half-band upsampler:

We interchange the filters with the 1:2 block and use the commutator notation.



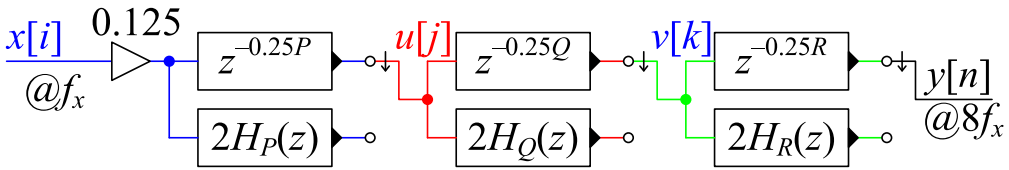
$H_1(z)$ is symmetrical with $\frac{M}{2}$ coefficients so we need $\frac{M}{4}$ multipliers in total (input gain of 0.5 can usually be absorbed elsewhere).



Computation: $\frac{M}{4}$ multiplies per input sample

Dyadic 1:8 Upsampler

13: Resampling Filters
Resampling
Halfband Filters
Dyadic 1:8 Upsampler
Rational Resampling
Arbitrary Resampling
Polynomial Approximation
Farrow Filter
Summary
MATLAB routines



Suppose $X(z)$: BW $= 0.8\pi \Leftrightarrow \alpha = 0.2$

Upsample 1:2 $\rightarrow U(z)$:

Filter $H_P(z)$ must remove image: $\Delta\omega = 0.2\pi$

For attenuation $= 60$ dB, $P \approx \frac{60}{3.5\Delta\omega} = 27.3$

Round up to a multiple of 4: $P = 28$

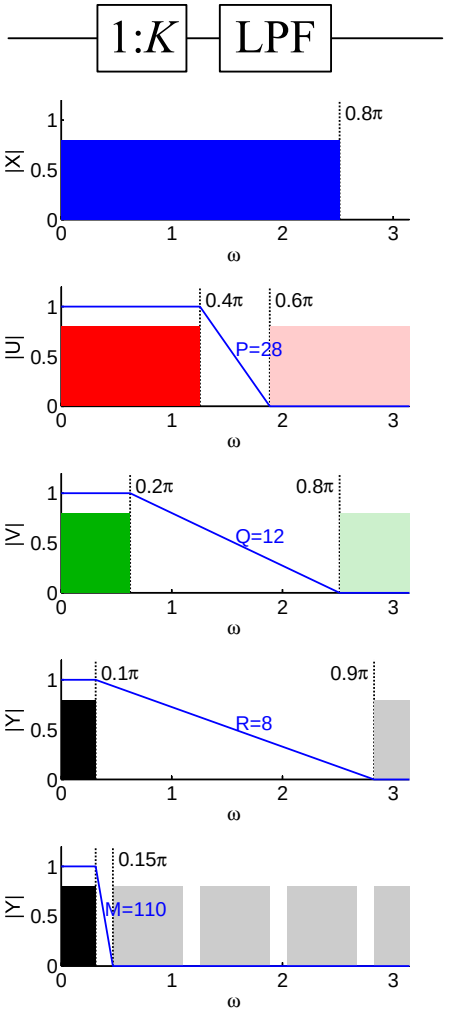
Upsample 1:2 $\rightarrow V(z)$: $\Delta\omega = 0.6\pi \Rightarrow Q = 12$

Upsample 1:2 $\rightarrow Y(z)$: $\Delta\omega = 0.8\pi \Rightarrow R = 8$

[diminishing returns + higher sample rate]

Multiplication Count:

$$\left(1 + \frac{P}{4}\right) \times f_x + \frac{Q}{4} \times 2f_x + \frac{R}{4} \times 4f_x = 22f_x$$

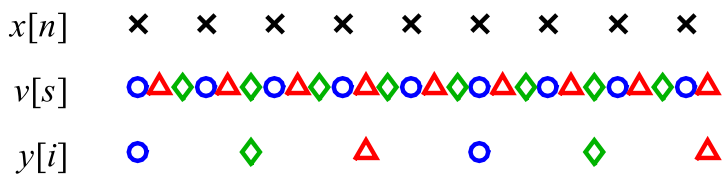


Alternative approach using direct 1:8 upsampling:

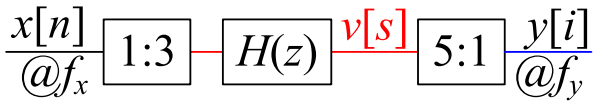
$\Delta\omega = 0.05\pi \Rightarrow M = 110 \Rightarrow 111f_x$ multiplications (using polyphase)

Rational Resampling

13: Resampling Filters
Resampling
Halfband Filters
Dyadic 1:8 Upsampler
Rational Resampling
Arbitrary Resampling + Polynomial Approximation
Farrow Filter +
Summary
MATLAB routines

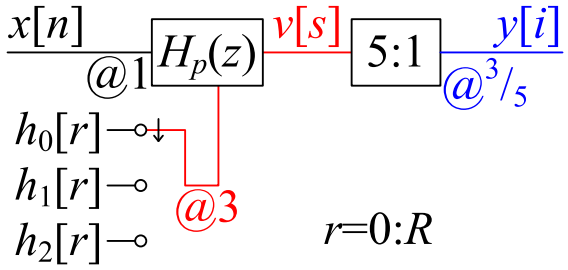


To resample by $\frac{P}{Q}$ do 1:P then LPF, then Q:1.



Resample by $\frac{P}{Q} \Rightarrow \omega_0 = \frac{\pi}{\max(P,Q)}$

$$\Delta\omega \triangleq 2\alpha\omega_0 = \frac{2\alpha\pi}{\max(P,Q)}$$



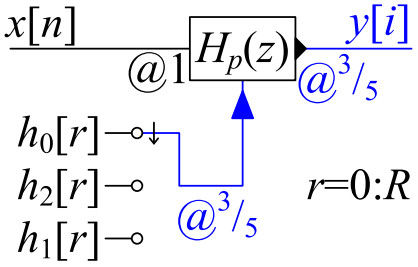
Polyphase: $H(z) = \sum_{p=0}^{P-1} z^{-p} H_p(z^P)$

Commutate coefficients:

$v[s]$ uses $H_p(z)$ with $p = s \bmod P$

Keep only every Q^{th} output:

$y[i]$ uses $H_p(z)$ with $p = Qi \bmod P$



Multiplication Count:

$$H(z)\text{: } M+1 \approx \frac{60\ [\text{dB}]}{3.5\Delta\omega} = \frac{2.7\max(P,Q)}{\alpha}$$

$$H_p(z)\text{: } R+1 = \frac{M+1}{P} = \frac{2.7}{\alpha}\max\left(1,\ \frac{Q}{P}\right)$$

$M+1$ coeficients in all

Multiplication rate: $\frac{2.7}{\alpha}\max\left(1,\ \frac{Q}{P}\right) \times f_y = \frac{2.7}{\alpha}\max\left(f_y,\ f_x\right)$

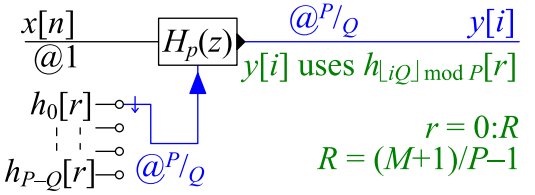
13: Resampling Filters
Resampling
Halfband Filters
Dyadic 1:8 Upsampler
Rational Resampling
▷ Arbitrary Resampling +
Polynomial Approximation
Farrow Filter +
Summary
MATLAB routines

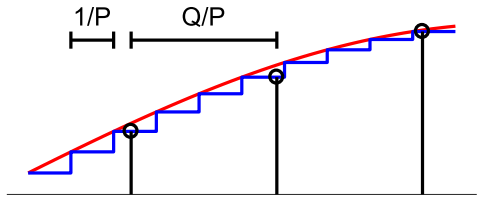
Sometimes need very large P and Q:

e.g. $\frac{44.1\text{ kHz}}{48\text{ kHz}} = \frac{147}{160}$

Multiplication rate OK: $\frac{2.7\max(f_y, f_x)}{\alpha}$

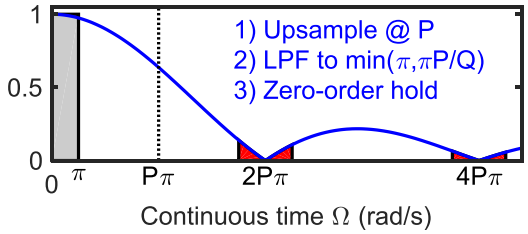
However # coefficients: $\frac{2.7\max(P,Q)}{\alpha}$





Alternatively, use any large integer P and round down to the nearest sample:

E.g. for $y[i]$ at time $i\frac{Q}{P}$ use $h_p[r]$ where $p = (\lfloor iQ \rfloor)_{\bmod P}$



Equivalent to converting to analog with zero-order hold and resampling at $f_y = \frac{P}{Q}$.

Zero-order hold convolves with rectangular $\frac{1}{P}$-wide window $\Rightarrow$ multiplies periodic spectrum by $\frac{\sin\frac{\Omega}{2P}}{\frac{\Omega}{2P}}$. Resampling aliases Ω to $\Omega_{\bmod \frac{2P\pi}{Q}}$.

Unit power component at Ω_1 gives alias components with total power:

$$\sin^2\tfrac{\Omega_1}{2P}\sum_{n=1}^{\infty}\left(\frac{2P}{2nP\pi+\Omega_1}\right)^2+\left(\frac{2P}{2nP\pi-\Omega_1}\right)^2 \approx \frac{\omega_1^2}{4P^2}\frac{2\pi^2}{6\pi^2} = \frac{\Omega_1^2}{12P^2}$$

For worst case, $\Omega_1 = \pi$, need $P = 906$ to get -60 dB ☹

[Arbitrary Resampling]

Suppose we wish to upsample by an irrational factor, $\sqrt{2} = \frac{P}{Q}$. We choose a integer value for $P \gg \frac{P}{Q}$, say $P = 25$. Conceptually, we will upsample by $P = 25$ to obtain $v[s]$ and then downsample by $Q = \frac{P}{\sqrt{2}} = 17.6\ldots$. Taking the input sample rate to be 1, the output sample number i will be at time $\frac{i}{\sqrt{2}} = \frac{iQ}{P}$ which corresponds to the sample $n' = \frac{iQ}{P}$ of $x[n]$ and to sample $s' = iQ$ of $v[s]$.
Unfortunately, s' is not an integer and so we will instead use sample $s = \lfloor s' \rfloor = \lfloor iQ \rfloor$ of $v[s]$ instead where $\lfloor\ \rfloor$ denotes the "floor" function which rounds down to the nearest integer. To calculate this, we use the sub-filter $h_p[r]$ where $p = s \mod P$. The input samples used by the filter will be the $R+1$ most recent samples of $x[n]$ namely $x[\lfloor n' \rfloor - R]$ to $x[\lfloor n' \rfloor]$.

i	$n' = iQ/P$	$s' = iQ$	$s = \lfloor s' \rfloor$	$p = s \mod P$	$\lfloor n' \rfloor - R : \lfloor n' \rfloor$
0	0	0	0	0	$-R : 0$
1	0.71	17.68	17	17	$-R : 0$
2	1.41	35.36	35	10	$1 - R : 1$
3	2.12	53.03	53	3	$2 - R : 2$
4	2.83	70.71	70	20	$2 - R : 2$
5	3.54	88.39	88	13	$3 - R : 3$

The table shows the values of everything for the first six samples of $y[i]$. Since we only use every 17^{th} or 18^{th} value of $v[s]$, the subfilter that is used, p, increases by 17 or 18 (modulo P) each time.

[Alias Components]

Ignoring the polyphase implementation, the low pass filter operates at a sample rate of P and therefore has a periodic spectrum that repeats at intervals of $2P\pi$. Therefore, considering positive frequencies only, a signal component in the passband at Ω_1 will have images at $\Omega = 2nP\pi \pm \Omega_1$ for all positive integers n.

These components are multiplied by the $\frac{\sin 0.5P^{-1}\Omega}{0.5P^{-1}\Omega}$ function and therefore have amplitudes of

$$\frac{\sin 0.5P^{-1}(2nP\pi \pm \Omega_1)}{0.5P^{-1}(2nP\pi \pm \Omega_1)} = \frac{\sin\left(n\pi \pm 0.5P^{-1}\Omega_1\right)}{\left(n\pi \pm 0.5P^{-1}\Omega_1\right)} = \frac{\sin\left(\pm 1^n 0.5P^{-1}\Omega_1\right)}{\left(n\pi \pm 0.5P^{-1}\Omega_1\right)}.$$

When we do the downsampling to an output sample rate of $\frac{P}{Q}$, these images will be aliased to frequencies $\Omega_{\text{mod } \frac{2P\pi}{Q}}$. In general, these alias frequencies will be scattered throughout the range $(0,\, \pi)$ and will result in broadband noise.

We need to sum the squared amplitudes of all these components:

$$\sum_{n=1}^{\infty} \frac{\sin^2\left(\pm 1^n 0.5P^{-1}\Omega_1\right)}{\left(n\pi \pm 0.5P^{-1}\Omega_1\right)^2} = \sin^2\left(0.5P^{-1}\Omega_1\right) \sum_{n=1}^{\infty} \frac{1}{\left(n\pi \pm 0.5P^{-1}\Omega_1\right)^2}$$

If we assume that $n\pi \gg 0.5P^{-1}\Omega_1$ and also that $\sin\left(0.5P^{-1}\Omega_1\right) \approx 0.5P^{-1}\Omega_1$, then we can approximate this sum as

$$\left(0.5P^{-1}\Omega_1\right)^2 \sum_{n=1}^{\infty} \frac{2}{(n\pi)^2} = \frac{\Omega_1^2}{4P^2} \times \frac{2}{\pi^2} \sum_{n=1}^{\infty} n^{-2}$$

The summation is a standard result and equals $\frac{\pi^2}{6}$.

So the total power of the aliased components is $\frac{\Omega_1^2}{12P^2}$.

Polynomial Approximation

13: Resampling Filters
Resampling
Halfband Filters
Dyadic 1:8 Upsampler
Rational Resampling
Arbitrary Resampling
+
Polynomial Approximation
Farrow Filter +
Summary
MATLAB routines

Suppose $P = 50$ and $H(z)$ has order $M = 249$

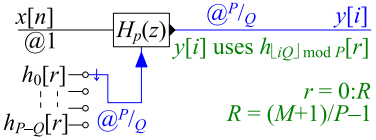
$H(z)$ is lowpass filter with $\omega_0 \approx \frac{\pi}{50}$

Split into 50 filters of length $R + 1 = \frac{M+1}{P} = 5$:

$h_p[0]$ is the first P samples of $h[m]$

$h_p[1]$ is the next P samples, etc.

$h_p[r] = h[p + rP]$

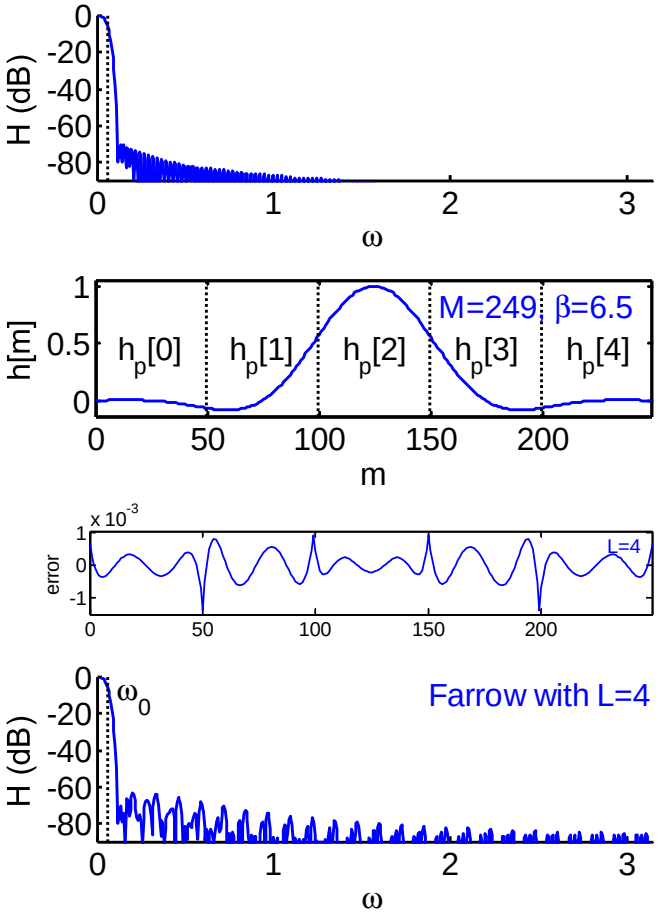


Use a polynomial of order L to approximate each segment:

$h_p[r] \approx f_r(\frac{p}{P})$ with $0 \le \frac{p}{P} < 1$

$h[m]$ is smooth, so errors are low.

E.g. error $< 10^{-3}$ for $L = 4$



- Resultant filter almost as good
- Instead of $M + 1 = 250$ coefficients we only need

$$(R + 1)(L + 1) = 25$$

where

$$R + 1 = \frac{2.7}{\alpha} \max\left(1, \frac{Q}{P}\right)$$

Farrow Filter

13: Resampling Filters
Resampling
Halfband Filters
Dyadic 1:8 Upsampler
Rational Resampling
Arbitrary Resampling
Polynomial Approximation
▷ Farrow Filter
Summary
MATLAB routines

Filter coefficients depend on fractional part of $i\frac{Q}{P}$:

$$\Delta[i] = i\frac{Q}{P} - n \text{ where } n = \left\lfloor i\frac{Q}{P} \right\rfloor$$

$$y[i] = \sum_{r=0}^{R} f_r(\Delta[i]) x[n-r]$$
where $f_r(\Delta) = \sum_{l=0}^{L} b_l[r]\Delta^l$

$$\begin{aligned} y[i] &= \sum_{r=0}^{R}\sum_{l=0}^{L} b_l[r]\Delta[i]^l x[n-r] \\ &= \sum_{l=0}^{L}\Delta[i]^l \sum_{r=0}^{R} b_l[r] x[n-r] \\ &= \sum_{l=0}^{L}\Delta[i]^l v_l[n] \end{aligned}$$
where $v_l[n] = b_l[n] * x[n]$
[like a Taylor series expansion]

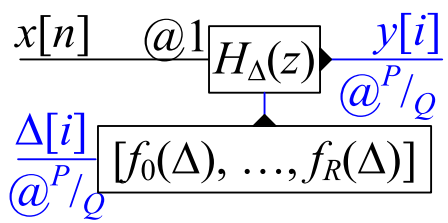
Horner's Rule:

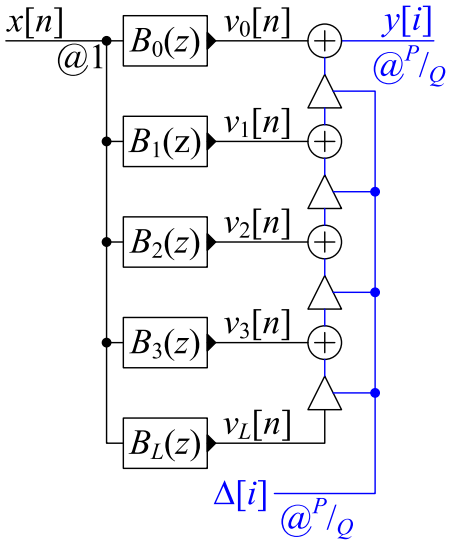
$$y[i] = v_0[n] + \Delta\left(v_1[n] + \Delta\left(v_2[n] + \Delta\left(\cdots\right)\right)\right)$$

Multiplication Rate:

Each $B_l(z)$ needs $R+1$ per input sample
Horner needs L per output sample
Total: $(L+1)(R+1)f_x + Lf_y = \frac{2.7(L+1)}{\alpha}\max\left(1, \frac{f_x}{f_y}\right) f_x + Lf_y$

$R+1 = \frac{M+1}{P} = 5$





$R+1 \approx \frac{2.7}{\alpha}\max\left(1, \frac{Q}{P}\right)$

[Farrow Filter sub-filter indexing]

We assume that the input sample rate is 1 and the output sample rate is $\frac{P}{Q}$. Output sample $y[i]$ is therefore at time $n' = \frac{iQ}{P}$ which will not normally be an integer.

Normal Resampling Method

In the normal resampling procedure, this corresponds to sample $s = iQ$ of $v[s]$ where $v[s]$ is obtained by upampling $x[n]$ by a factor of P. Using a polyphase filter to do the upsampling, we use each of the sub-filters $h_p[n]$ in turn to generate the upsampled samples $v[s]$ where $p = s \mod P$ and the filter acts on the $R + 1$most recent input samples, $x[n - R]$ to $x[n]$ where $n = \lfloor n' \rfloor]$. We can write any integer s, as the sum of an exact multiple of P and the remainder when $s \div P$ as $s = P \left\lfloor \frac{s}{P} \right\rfloor + s \mod P$. Substituting the previously defined expressions for n and p into this equation gives $iQ = Pn + p$. We can rearrange this to get $p = Pn' - Pn$ where p lies in the range $[0,\, P - 1]$ and determines which of the subfilters we will use.

Farrow Filter

In the normal method (above), the sub-filter than we use is indexed by p which lies in the range $[0,\, P-1]$. In the Farrow filter, the sub-filter that we use is instead indexed by the value of the fractional number $\Delta = \frac{p}{P}$ which always lies in the range $[0,\, 1)$. From the previous paragraph, $\Delta[i] = \frac{p}{P} = n' - n = i\frac{Q}{P} - \left\lfloor i\frac{Q}{P} \right\rfloor$ which is a function only of the output sample number, i and the resampling ratio $\frac{P}{Q}$. The advantage of this is that both P nor Q can now be non-integers.

Summary

13: Resampling Filters
Resampling
Halfband Filters
Dyadic 1:8 Upsampler
Rational Resampling
Arbitrary Resampling + Polynomial Approximation
Farrow Filter +
▷ Summary
MATLAB routines

- Transition band centre at ω_0
  - ω_0 = the lower of the old and new Nyquist frequencies
  - Transition width = $\Delta\omega = 2\alpha\omega_0$, typically $\alpha \approx 0.1$
- Factorizing resampling ratio can reduce computation
  - halfband filters very efficient (half the coefficients are zero)
- Rational resampling $\times \frac{P}{Q}$
  - # multiplies per second: $\frac{2.7}{\alpha} \max(f_y,\, f_x)$
  - # coefficients: $\frac{2.7}{\alpha} \max(P,\, Q)$
- Farrow Filter
  - approximate filter impulse response with polynomial segments
  - arbitrary, time-varying, resampling ratios
  - # multiplies per second: $\frac{2.7(L+1)}{\alpha} \max(f_y,\, f_x) \times \frac{f_x}{f_y} + L f_y$
    - ▷ $\approx (L+1)\,\frac{f_x}{f_y}$ times rational resampling case
  - # coefficients: $\frac{2.7}{\alpha} \max(P,\, Q) \times \frac{L+1}{P}$
  - coefficients are independent of f_y when upsampling

For further details see Mitra: 13 and Harris: 7, 8.

MATLAB routines

13: Resampling Filters
Resampling
Halfband Filters
Dyadic 1:8 Upsampler
Rational Resampling
Arbitrary Resampling +
Polynomial Approximation
Farrow Filter +
Summary
▷ MATLAB routines

gcd(p,q)	Find $\alpha p + \beta q = 1$ for coprime p, q
polyfit	Fit a polynomial to data
polyval	Evaluate a polynomial
upfirdn	Perform polyphase filtering
resample	Perform polyphase resampling

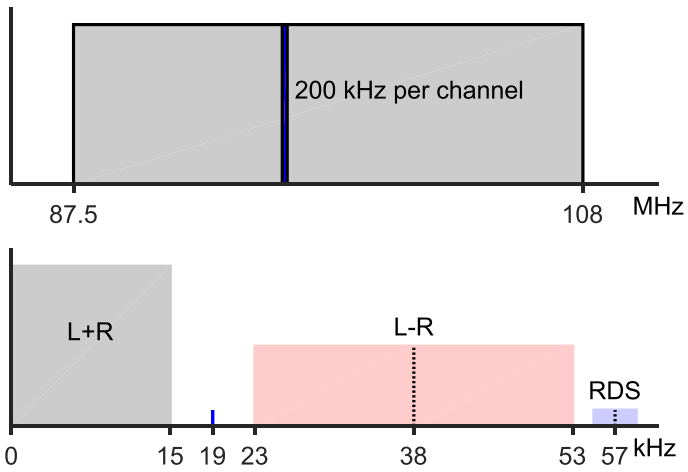
14: FM Radio Receiver
FM Radio Block Diagram
Aliased ADC
Channel Selection
Channel Selection (1)
Channel Selection (2)
Channel Selection (3)
FM Demodulator
Differentiation Filter
Pilot tone extraction + Polyphase Pilot tone
Summary

14: FM Radio Receiver

FM Radio Block Diagram

14: FM Radio Receiver
- FM Radio Block Diagram
- Aliased ADC
- Channel Selection
- Channel Selection (1)
- Channel Selection (2)
- Channel Selection (3)
- FM Demodulator
- Differentiation Filter
- Pilot tone extraction + Polyphase Pilot tone
- Summary

FM spectrum: 87.5 to $108\,\mathrm{MHz}$
Each channel: $\pm 100\,\mathrm{kHz}$

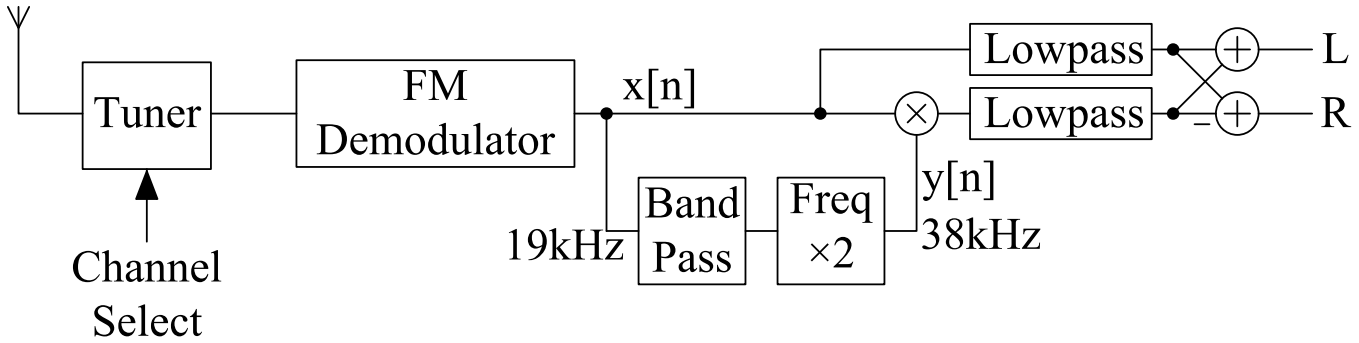


Baseband signal:

- Mono (L + R): $\pm 15\,\mathrm{kHz}$
- Pilot tone: $19\,\mathrm{kHz}$
- Stereo (L – R): $38 \pm 15\,\mathrm{kHz}$
- RDS: $57 \pm 2\,\mathrm{kHz}$

FM Modulation:

- Freq deviation: $\pm 75\,\mathrm{kHz}$



L–R signal is multiplied by $38\,\mathrm{kHz}$ to shift it to baseband

[This example is taken from Ch 13 of Harris: Multirate Signal Processing]

Aliased ADC

14: FM Radio Receiver
FM Radio Block Diagram
▷ Aliased ADC
Channel Selection
Channel Selection (1)
Channel Selection (2)
Channel Selection (3)
FM Demodulator
Differentiation Filter
Pilot tone extraction + Polyphase Pilot tone
Summary

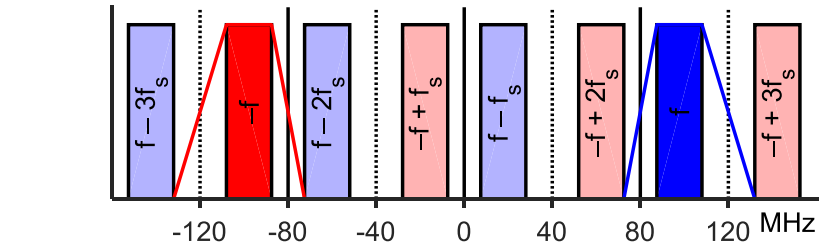
FM band: 87.5 to $108\,\mathrm{MHz}$
Normally sample at $f_s > 2f$

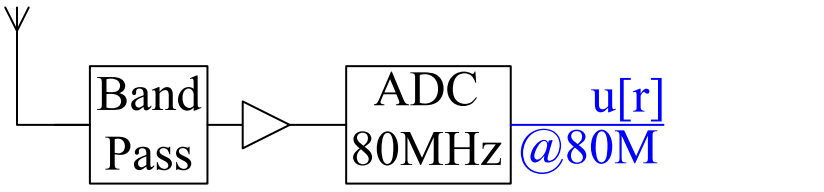
However:

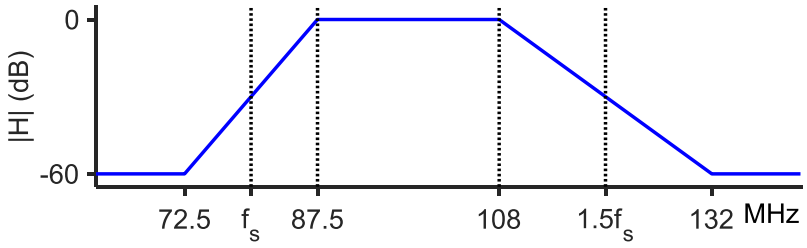
$f_s = 80\,\mathrm{MHz}$ aliases band down to $[7.5,\ 28]\,\mathrm{MHz}$.

–ve frequencies alias to $[-28,\ -7.5]\,\mathrm{MHz}$.

We must suppress other frequencies that alias to the range $\pm[7.5,\ 28]\,\mathrm{MHz}$.







Need an analogue bandpass filter to extract the FM band. Transition band mid-points are at $f_s = 80\,\mathrm{MHz}$ and $1.5f_s = 120\,\mathrm{MHz}$.

You can use an aliased analog-digital converter (ADC) provided that the target band fits entirely between two consecutive multiples of $\frac{1}{2}f_s$.
Lower ADC sample rate ☺. Image = undistorted frequency-shifted copy.

Channel Selection

14: FM Radio Receiver
FM Radio Block Diagram
Aliased ADC
▷ Channel Selection
Channel Selection (1)
Channel Selection (2)
Channel Selection (3)
FM Demodulator
Differentiation Filter
Pilot tone extraction + Polyphase Pilot tone
Summary

FM band shifted to 7.5 to $28\,\mathrm{MHz}$ (from 87.5 to $108\,\mathrm{MHz}$)

We need to select a single channel $200\,\mathrm{kHz}$ wide

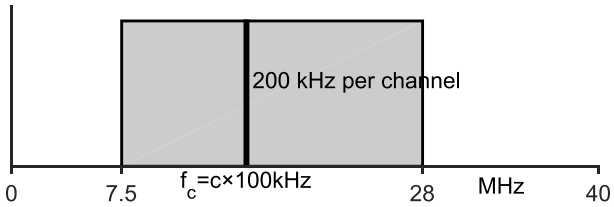
We shift selected channel to DC and then downsample to $f_s = 400\,\mathrm{kHz}$.
Assume channel centre frequency is $f_c = c \times 100\,\mathrm{kHz}$

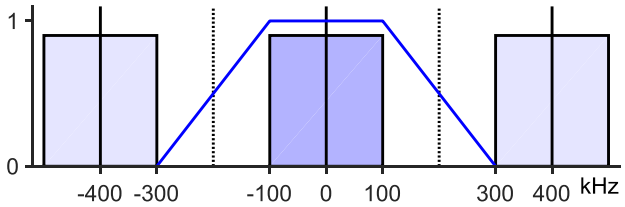
We must apply a filter before downsampling to remove unwanted images

The downsampled signal is complex since positive and negative frequencies contain different information.

We will look at three methods:

1. Freq shift, then polyphase lowpass filter
2. Polyphase bandpass complex filter
3. Polyphase bandpass real filter



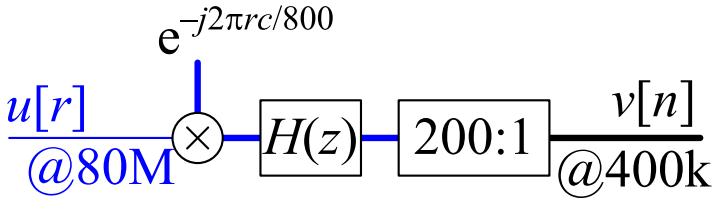


Channel Selection (1)

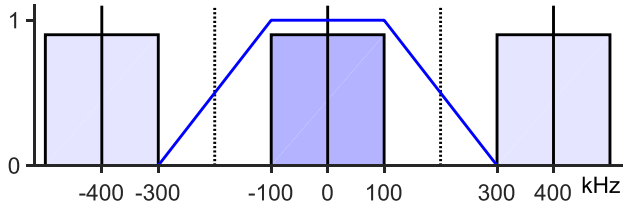
14: FM Radio Receiver
FM Radio Block Diagram
Aliased ADC
Channel Selection
Channel Selection
▷ (1)
Channel Selection (2)
Channel Selection (3)
FM Demodulator
Differentiation Filter
Pilot tone extraction + Polyphase Pilot tone
Summary

Multiply by $e^{-j2\pi r \frac{f_c}{80\,\mathrm{MHz}}}$ to shift channel at f_c to DC.

$f_c = c \times 100\,\mathrm{k} \Rightarrow \frac{f_c}{80\,\mathrm{M}} = \frac{c}{800}$



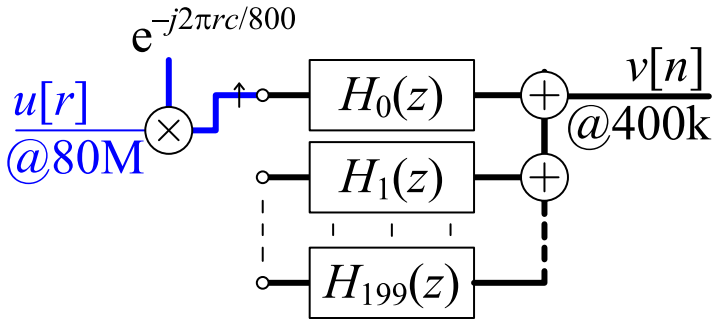
Result of multiplication is complex (thick lines on diagram)



Next, lowpass filter to $\pm 100\,\mathrm{kHz}$

$\Delta\omega = 2\pi \frac{200\,\mathrm{k}}{80\,\mathrm{M}} = 0.157$

$\Rightarrow M = \frac{60\,\mathrm{dB}}{3.5\Delta\omega} = 1091$



Finally, downsample $200:1$

Polyphase:

$H_p(z)$ has $\lceil \frac{1092}{200} \rceil = 6$ taps

Complex data $\times$ Real Coefficients (needs 2 multiplies per tap)

Multiplication Load:

$2 \times 80\,\mathrm{MHz}$ (freq shift) $+ 12 \times 80\,\mathrm{MHz}$ ($H_p(z)$) $= 14 \times 80\,\mathrm{MHz}$

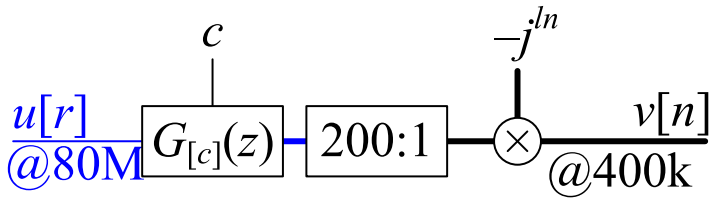
Channel Selection (2)

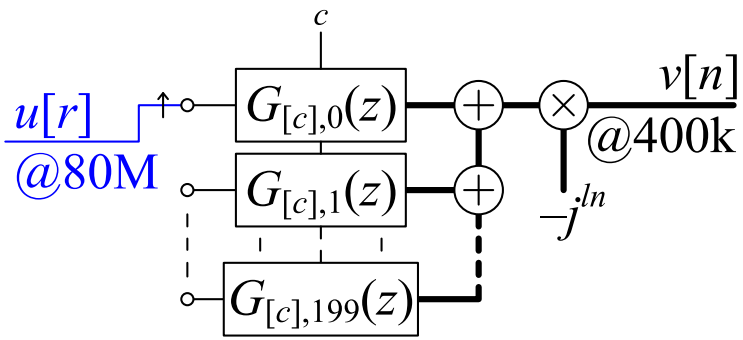
14: FM Radio Receiver
FM Radio Block Diagram
Aliased ADC
Channel Selection
Channel Selection (1)
▷ Channel Selection (2)
Channel Selection (3)
FM Demodulator
Differentiation Filter
Pilot tone extraction + Polyphase Pilot tone
Summary

Channel centre frequency $f_c = c \times 100\,\text{kHz}$ where c is an integer.

Write $c = 4k + l$

where $k = \left\lfloor \frac{c}{4} \right\rfloor$ and $l = c_{\bmod 4}$





We multiply $u[r]$ by $e^{-j2\pi r \frac{c}{800}}$, convolve with $h[m]$ and then downsample:

$$
\begin{aligned}
v[n] &= \sum_{m=0}^{M} h[m]u[200n-m]e^{-j2\pi(200n-m)\frac{c}{800}} && [r = 200n] \\
&= \sum_{m=0}^{M} h[m]e^{j2\pi\frac{mc}{800}}u[200n-m]e^{-j2\pi 200n\frac{4k+l}{800}} && [c = 4k+1] \\
&= \sum_{m=0}^{M} g_{[c]}[m]u[200n-m]e^{-j2\pi\frac{ln}{4}} && [g_{[c]}[m] \triangleq h[m]e^{j2\pi\frac{mc}{800}}] \\
&= (-j)^{ln}\sum_{m=0}^{M} g_{[c]}[m]u[200n-m] && [e^{-j2\pi\frac{ln}{4}} \text{ indep of } m]
\end{aligned}
$$

Multiplication Load for polyphase implementation:

$G_{[c],p}(z)$ has complex coefficients $\times$ real input $\Rightarrow$ 2 mults per tap

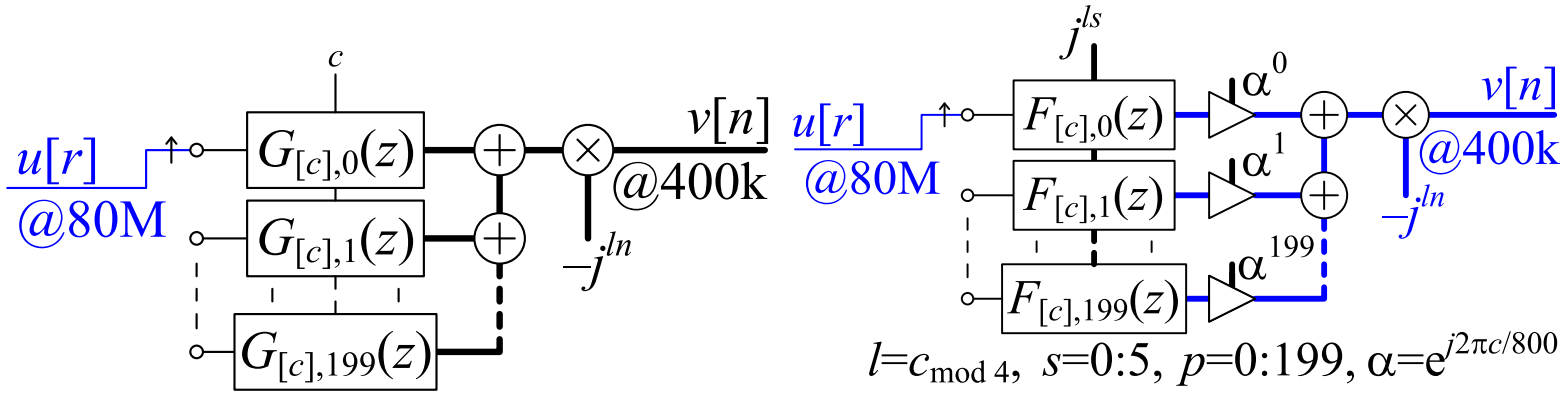
$(-j)^{ln} \in \{+1, -j, -1, +j\}$ so no actual multiplies needed

Total: $12 \times 80\,\text{MHz}$ (for $G_{[c],p}(z)$) + 0 (for $-j^{ln}$) $= 12 \times 80\,\text{MHz}$

Channel Selection (3)

14: FM Radio Receiver
FM Radio Block Diagram
Aliased ADC
Channel Selection
Channel Selection (1)
Channel Selection (2)
▷ Channel Selection (3)
FM Demodulator
Differentiation Filter
Pilot tone extraction + Polyphase Pilot tone
Summary

Channel frequency $f_c = c \times 100\,\text{kHz}$ where $c = 4k + l$ is an integer



$$g_{[c]}[m] = h[m]e^{j2\pi\frac{cm}{800}}$$

$$g_{[c],p}[s] = g_c[200s+p] = h[200s+p]e^{j2\pi\frac{c(200s+p)}{800}} \qquad \text{[polyphase]}$$

$$= h[200s+p]e^{j2\pi\frac{cs}{4}}e^{j2\pi\frac{cp}{800}} \triangleq h[200s+p]e^{j2\pi\frac{cs}{4}}\alpha^p$$

Define $f_{[c],p}[s] = h[200s+p]e^{j2\pi\frac{(4k+l)s}{4}} = j^{ls}h[200s+p]$

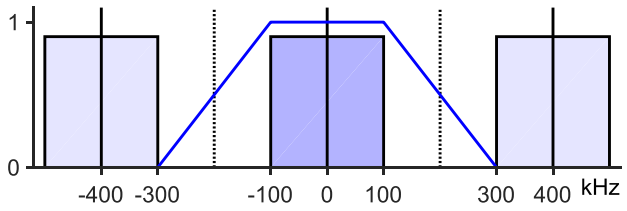
Although $f_{[c],p}[s]$ is complex it requires only one multiplication per tap because each tap is either purely real or purely imaginary.

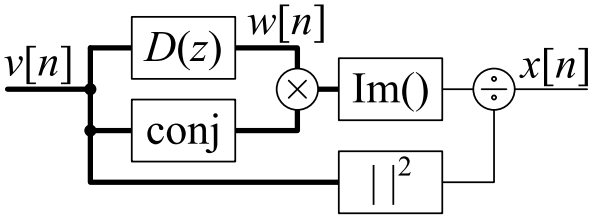
Multiplication Load:

$6 \times 80\,\text{MHz}$ $(F_p(z))$ $+ 4 \times 80\,\text{MHz}$ $(\times e^{j2\pi\frac{cp}{800}})$ $= 10 \times 80\,\text{MHz}$

FM Demodulator

14: FM Radio Receiver
FM Radio Block Diagram
Aliased ADC
Channel Selection
Channel Selection (1)
Channel Selection (2)
Channel Selection (3)
▷ FM Demodulator
Differentiation Filter
Pilot tone extraction + Polyphase Pilot tone
Summary





Complex FM signal centred at DC: $v(t) = |v(t)|e^{j\phi(t)}$

We know that $\log v = \log |v| + j\phi$

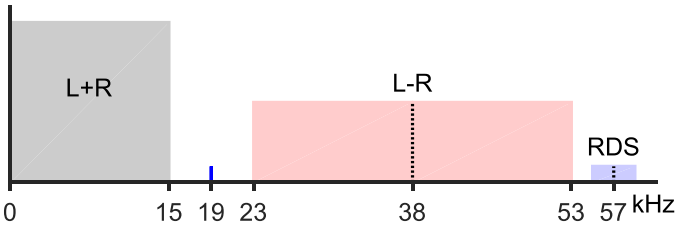
The instantaneous frequency of $v(t)$ is $\frac{d\phi}{dt}$.

We need to calculate $x(t) = \frac{d\phi}{dt} = \frac{d\Im(\log v)}{dt} = \Im\left(\frac{1}{v}\frac{dv}{dt}\right) = \frac{1}{|v|^2}\Im\left(v^*\frac{dv}{dt}\right)$

We need:

(1) Differentiation filter, $D(z)$
(2) Complex multiply, $w[n] \times v^*[n]$ (only need $\Im$ part)
(3) Real Divide by $|v|^2$

$x[n]$ is baseband signal (real):



Differentiation Filter

14: FM Radio Receiver
FM Radio Block Diagram
Aliased ADC
Channel Selection
Channel Selection (1)
Channel Selection (2)
Channel Selection (3)
FM Demodulator
Differentiation Filter
Pilot tone extraction + Polyphase Pilot tone
Summary

Window design method:

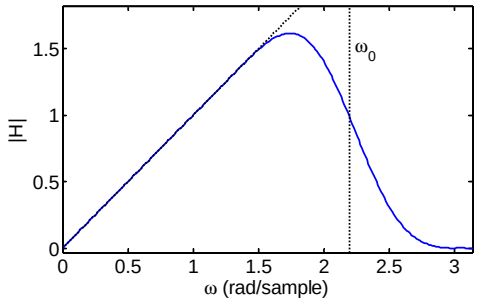
(1) calculate $d[n]$ for the ideal filter

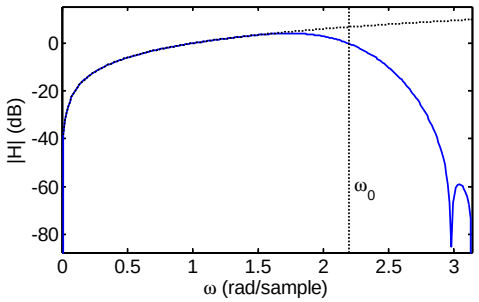
(2) multiply by a window to give finite support

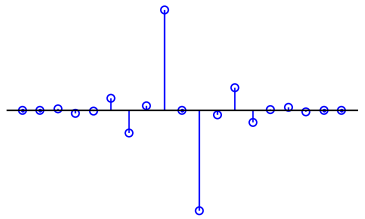
Differentiation: $\frac{d}{dt}e^{j\omega t} = j\omega e^{j\omega t} \quad \Rightarrow \quad D(e^{j\omega}) = \begin{cases} j\omega & |\omega| \le \omega_0 \\ 0 & |\omega| > \omega_0 \end{cases}$

Hence $d[n] = \frac{1}{2\pi}\int_{-\omega_0}^{\omega_0} j\omega e^{j\omega n} d\omega = \frac{j}{2\pi}\left[\frac{\omega e^{jn\omega}}{jn} - \frac{e^{jn\omega}}{j^2 n^2}\right]_{-\omega_0}^{\omega_0}$ [IDTFT]

$= \frac{n\omega_0 \cos n\omega_0 - \sin n\omega_0}{\pi n^2}$





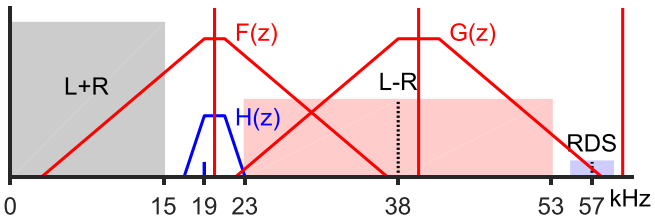


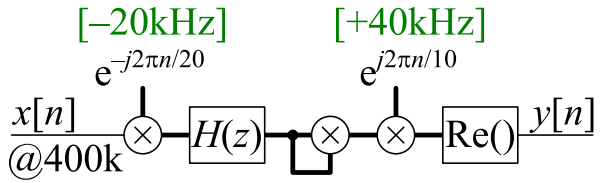
Using $M = 18$, Kaiser window, $\beta = 7$ and $\omega_0 = 2.2 = \frac{2\pi \times 140\,\text{kHz}}{400\,\text{kHz}}$:

Near perfect differentiation for $\omega \le 1.6$ ($\approx 100\,\text{kHz}$ for $f_s = 400\,\text{kHz}$)

Broad transition region allows shorter filter

14: FM Radio Receiver
FM Radio Block Diagram
Aliased ADC
Channel Selection
Channel Selection (1)
Channel Selection (2)
Channel Selection (3)
FM Demodulator
Differentiation Filter
▷ Pilot tone extraction +
Polyphase Pilot tone
Summary

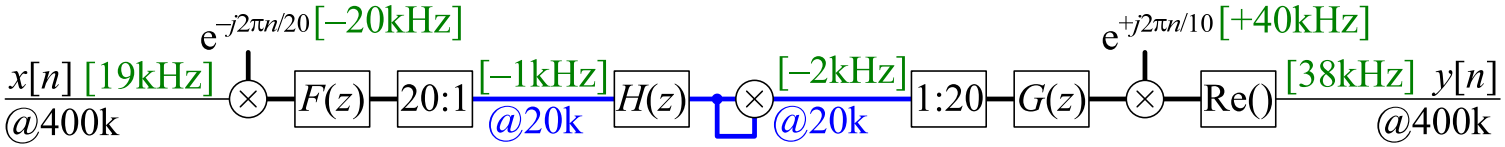




Aim: extract $19\,\mathrm{kHz}$ pilot tone, double freq $\rightarrow$ real $38\,\mathrm{kHz}$ tone.

(1) shift spectrum down by $20\,\mathrm{kHz}$: multiply by $e^{-j2\pi n\frac{20\,\mathrm{kHz}}{400\,\mathrm{kHz}}}$
(2) low pass filter to $\pm 1\,\mathrm{kHz}$ to extract complex pilot at $-1\,\mathrm{kHz}$: $H(z)$
(3) square to double frequency to $-2\,\mathrm{kHz}$ $\quad [(e^{j\omega t})^2 = e^{j2\omega t}]$
(4) shift spectrum up by $40\,\mathrm{kHz}$: multiply by $e^{+j2\pi n\frac{40\,\mathrm{kHz}}{400\,\mathrm{kHz}}}$
(5) take real part

More efficient to do low pass filtering at a low sample rate:

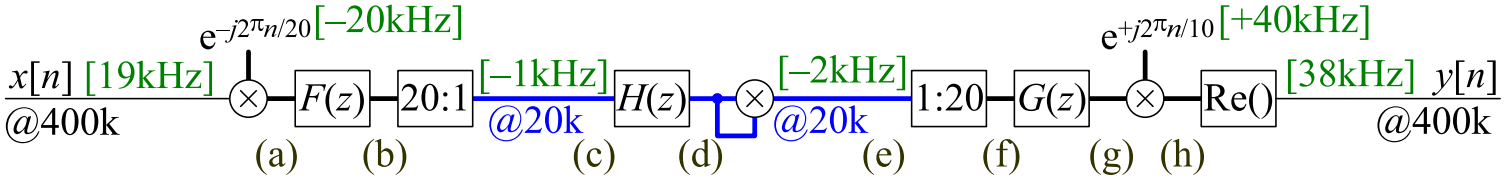


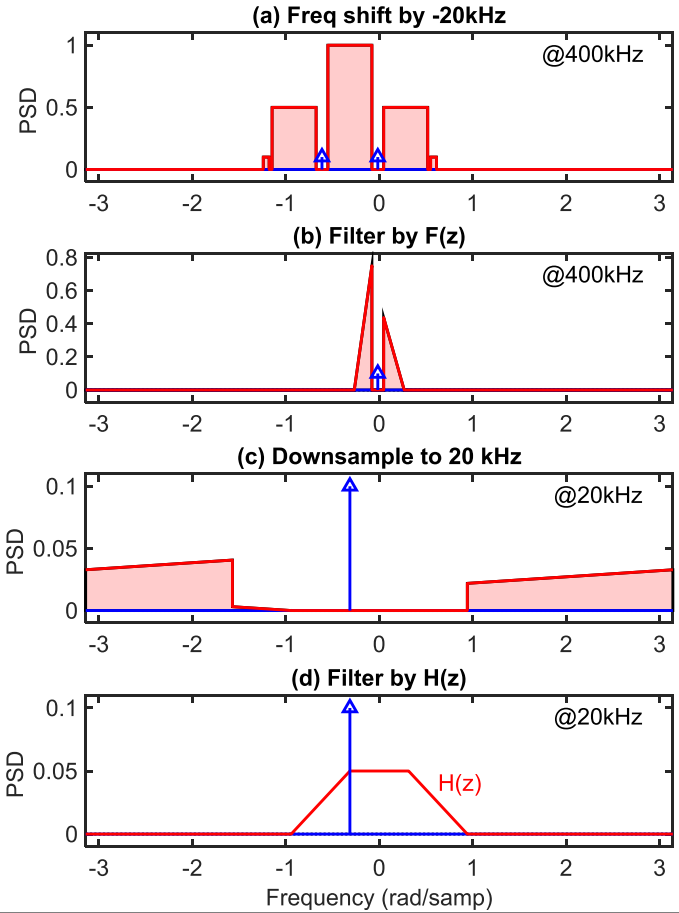
Transition bands:

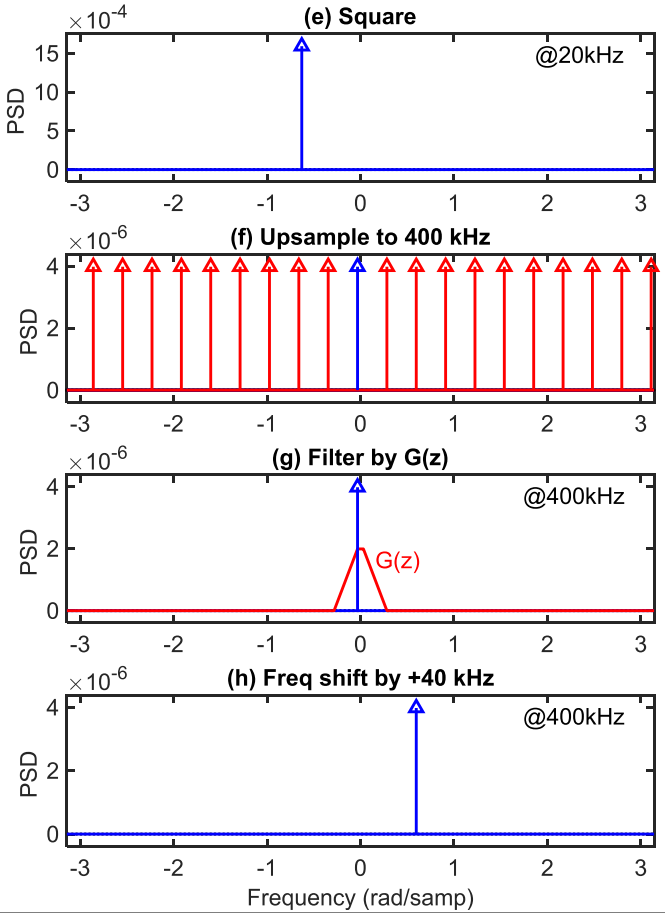
$F(z)$: $1 \rightarrow 17\,\mathrm{kHz}$, $\quad H(z)$: $1 \rightarrow 3\,\mathrm{kHz}$, $\quad G(z)$: $2 \rightarrow 18\,\mathrm{kHz}$

$\Delta\omega = 0.25 \Rightarrow M = 68$, $\quad \Delta\omega = 0.63 \Rightarrow 27$, $\quad \Delta\omega = 0.25 \Rightarrow 68$

[Pilot Tone Extraction]

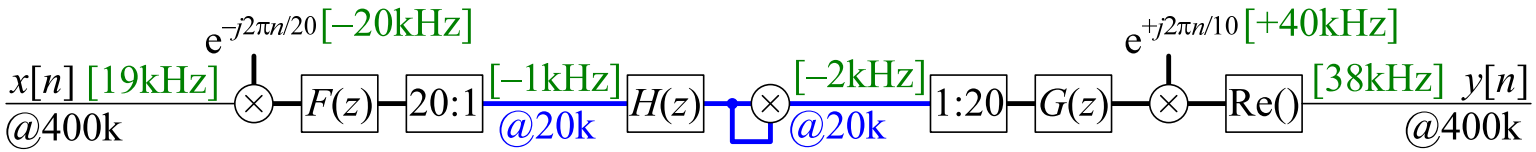






Polyphase Pilot tone

14: FM Radio Receiver
FM Radio Block Diagram
Aliased ADC
Channel Selection
Channel Selection (1)
Channel Selection (2)
Channel Selection (3)
FM Demodulator
Differentiation Filter
Pilot tone extraction
+
▷ Polyphase Pilot tone
Summary



Anti-alias filter: $F(z)$

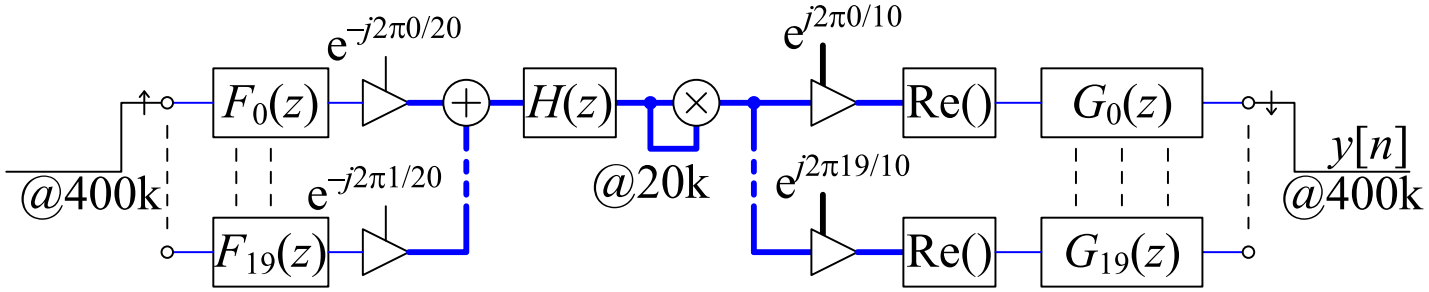
Each branch, $F_p(z)$, gets every 20^{th} sample and an identical $e^{j2\pi\frac{n}{20}}$

So $F_p(z)$ can filter a real signal and then multiply by fixed $e^{j2\pi\frac{p}{20}}$

Anti-image filter: $G(z)$

Each branch, $G_p(z)$, multiplied by identical $e^{j2\pi\frac{n}{10}}$

So $G_p(z)$ can filter a real signal



Multiplies:

F and G each: $(4+2)\times 400\,\text{kHz}$, $H+x^2$: $(2\times 28+4)\times 20\,\text{kHz}$

Total: $15\times 400\,\text{kHz}$ [Full-rate $H(z)$ needs $273\times 400\,\text{kHz}$]

Summary

14: FM Radio Receiver
FM Radio Block Diagram
Aliased ADC
Channel Selection
Channel Selection (1)
Channel Selection (2)
Channel Selection (3)
FM Demodulator
Differentiation Filter
Pilot tone extraction + Polyphase Pilot tone
▷ Summary

- Aliased ADC allows sampling below the Nyquist frequency
  - Only works because the wanted signal fits entirely within a Nyquist band image
- Polyphase filter can be combined with complex multiplications to select the desired image
  - subsequent multiplication by $-j^{ln}$ shifts by the desired multiple of $\frac{1}{4}$ sample rate
    - ▷ No actual multiplications required
- FM demodulation uses a differentiation filter to calculate $\frac{d\phi}{dt}$
- Pilot tone bandpass filter has narrow bandwidth so better done at a low sample rate
  - double the frequency of a complex tone by squaring it

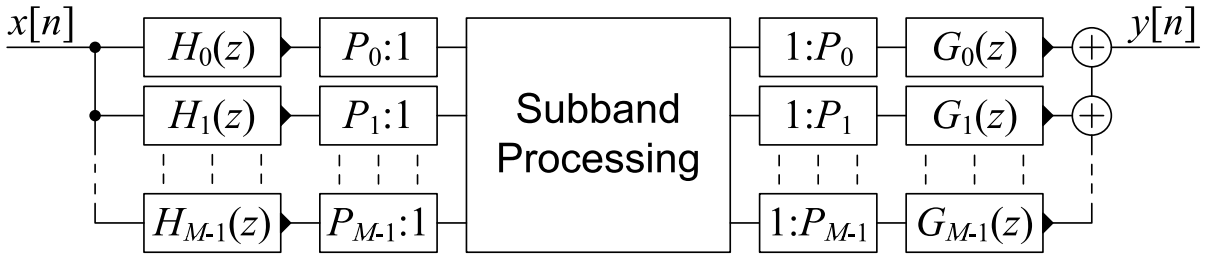
This example is taken from Harris: 13.

15: Subband Processing
Subband processing
2-band Filterbank
Perfect Reconstruction
Quadrature Mirror Filterbank (QMF)
Polyphase QMF
QMF Options
Linear Phase QMF
IIR Allpass QMF
Tree-structured filterbanks
Summary
Merry Xmas

15: Subband Processing

Subband processing

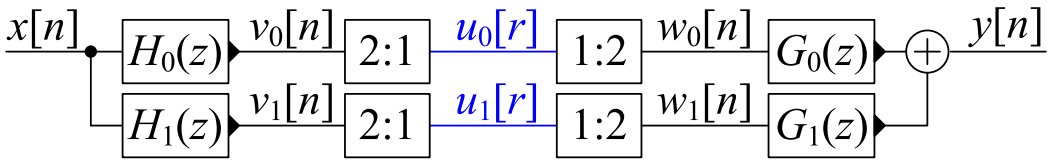
15: Subband Processing
Subband processing
2-band Filterbank
Perfect Reconstruction
Quadrature Mirror Filterbank (QMF)
Polyphase QMF
QMF Options
Linear Phase QMF
IIR Allpass QMF
Tree-structured filterbanks
Summary
Merry Xmas



- The $H_m(z)$ are bandpass *analysis filters* and divide $x[n]$ into frequency bands
- Subband processing often processes frequency bands independently
- The $G_m(z)$ are *synthesis filters* and together reconstruct the output
- The $H_m(z)$ outputs are bandlimited and so can be subsampled without loss of information
  - Sample rate multiplied overall by $\sum \frac{1}{P_i}$
    $\sum \frac{1}{P_i} = 1 \Rightarrow$ *critically sampled*: good for coding
    $\sum \frac{1}{P_i} > 1 \Rightarrow$ *oversampled*: more flexible
- Goals:
  (a) good frequency selectivity in $H_m(z)$
  (b) *perfect reconstruction*: $y[n] = x[n-d]$ if no processing
- Benefits: Lower computation, faster convergence if adaptive

2-band Filterbank

15: Subband Processing
Subband processing
▷ 2-band Filterbank
Perfect Reconstruction
Quadrature Mirror Filterbank (QMF)
Polyphase QMF
QMF Options
Linear Phase QMF
IIR Allpass QMF
Tree-structured filterbanks
Summary
Merry Xmas



$$V_m(z) = H_m(z)X(z) \qquad [m \in \{0, 1\}]$$

$$U_m(z) = \frac{1}{K}\sum_{k=0}^{K-1} V_m(e^{\frac{-j2\pi k}{K}} z^{\frac{1}{K}}) = \frac{1}{2}\left\{V_m\left(z^{\frac{1}{2}}\right) + V_m\left(-z^{\frac{1}{2}}\right)\right\}$$

$$W_m(z) = U_m(z^2) = \frac{1}{2}\{V_m(z) + V_m(-z)\} \qquad [K = 2]$$
$$= \frac{1}{2}\{H_m(z)X(z) + H_m(-z)X(-z)\}$$

$$Y(z) = \begin{bmatrix} W_0(z) & W_1(z) \end{bmatrix} \begin{bmatrix} G_0(z) \\ G_1(z) \end{bmatrix}$$

$$= \frac{1}{2}\begin{bmatrix} X(z) & X(-z) \end{bmatrix} \begin{bmatrix} H_0(z) & H_1(z) \\ H_0(-z) & H_1(-z) \end{bmatrix} \begin{bmatrix} G_0(z) \\ G_1(z) \end{bmatrix}$$

$$= \begin{bmatrix} X(z) & X(-z) \end{bmatrix} \begin{bmatrix} T(z) \\ A(z) \end{bmatrix} \qquad [X(-z)A(z) \text{ is "aliased" term}]$$

We want (a) $T(z) = \frac{1}{2}\{H_0(z)G_0(z) + H_1(z)G_1(z)\} = z^{-d}$
and (b) $A(z) = \frac{1}{2}\{H_0(-z)G_0(z) + H_1(-z)G_1(z)\} = 0$

Perfect Reconstruction

15: Subband Processing
Subband processing
2-band Filterbank
Perfect Reconstruction
Quadrature Mirror Filterbank (QMF)
Polyphase QMF
QMF Options
Linear Phase QMF
IIR Allpass QMF
Tree-structured filterbanks
Summary
Merry Xmas

For perfect reconstruction without aliasing, we require

$$\frac{1}{2}\begin{bmatrix} H_0(z) & H_1(z) \\ H_0(-z) & H_1(-z) \end{bmatrix}\begin{bmatrix} G_0(z) \\ G_1(z) \end{bmatrix} = \begin{bmatrix} z^{-d} \\ 0 \end{bmatrix}$$

Hence:

$$\begin{bmatrix} G_0(z) \\ G_1(z) \end{bmatrix} = \begin{bmatrix} H_0(z) & H_1(z) \\ H_0(-z) & H_1(-z) \end{bmatrix}^{-1}\begin{bmatrix} 2z^{-d} \\ 0 \end{bmatrix}$$

$$= \frac{2z^{-d}}{H_0(z)H_1(-z)-H_0(-z)H_1(z)}\begin{bmatrix} H_1(-z) & -H_1(z) \\ -H_0(-z) & H_0(z) \end{bmatrix}\begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$= \frac{2z^{-d}}{H_0(z)H_1(-z)-H_0(-z)H_1(z)}\begin{bmatrix} H_1(-z) \\ -H_0(-z) \end{bmatrix}$$

For all filters to be FIR, we need the denominator to be

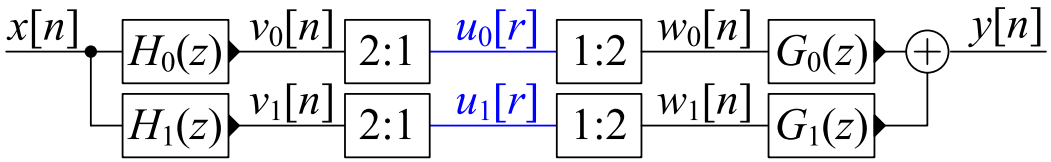
$H_0(z)H_1(-z) - H_0(-z)H_1(z) = cz^{-k}$, which implies

$$\begin{bmatrix} G_0(z) \\ G_1(z) \end{bmatrix} = \frac{2}{c}z^{k-d}\begin{bmatrix} H_1(-z) \\ -H_0(-z) \end{bmatrix} \overset{d=k}{=} \frac{2}{c}\begin{bmatrix} H_1(-z) \\ -H_0(-z) \end{bmatrix}$$

Note: c just scales $H_i(z)$ by $c^{\frac{1}{2}}$ and $G_i(z)$ by $c^{-\frac{1}{2}}$.

Quadrature Mirror Filterbank (QMF)

15: Subband Processing
Subband processing
2-band Filterbank
Perfect Reconstruction
▷ Quadrature Mirror Filterbank (QMF)
Polyphase QMF
QMF Options
Linear Phase QMF
IIR Allpass QMF
Tree-structured filterbanks
Summary
Merry Xmas



QMF satisfies:

(a) $H_0(z)$ is causal and real

(b) $H_1(z) = H_0(-z)$: i.e. $\left|H_0(e^{j\omega})\right|$ is reflected around $\omega = \frac{\pi}{2}$

(c) $G_0(z) = 2H_1(-z) = 2H_0(z)$

(d) $G_1(z) = -2H_0(-z) = -2H_1(z)$

QMF is alias-free:

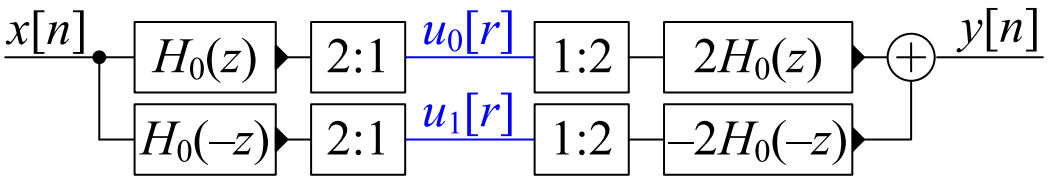
$$A(z) = \tfrac{1}{2}\{H_0(-z)G_0(z) + H_1(-z)G_1(z)\}$$
$$= \tfrac{1}{2}\{2H_1(z)H_0(z) - 2H_0(z)H_1(z)\} = 0$$

QMF Transfer Function:

$$T(z) = \tfrac{1}{2}\{H_0(z)G_0(z) + H_1(z)G_1(z)\}$$
$$= H_0^2(z) - H_1^2(z) = H_0^2(z) - H_0^2(-z)$$

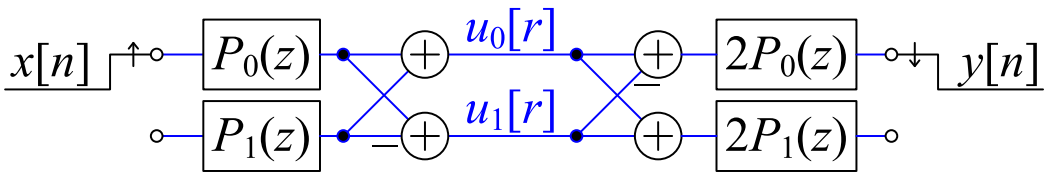
Polyphase QMF

15: Subband Processing
Subband processing
2-band Filterbank
Perfect Reconstruction
Quadrature Mirror Filterbank (QMF)
▷ Polyphase QMF
QMF Options
Linear Phase QMF
IIR Allpass QMF
Tree-structured filterbanks
Summary
Merry Xmas



Polyphase decomposition:

$$H_0(z) = P_0(z^2) + z^{-1}P_1(z^2)$$
$$H_1(z) = H_0(-z) = P_0(z^2) - z^{-1}P_1(z^2)$$
$$G_0(z) = 2H_0(z) = 2P_0(z^2) + 2z^{-1}P_1(z^2)$$
$$G_1(z) = -2H_0(-z) = -2P_0(z^2) + 2z^{-1}P_1(z^2)$$



Transfer Function:

$$T(z) = H_0^2(z) - H_1^2(z) = 4z^{-1}P_0(z^2)P_1(z^2)$$

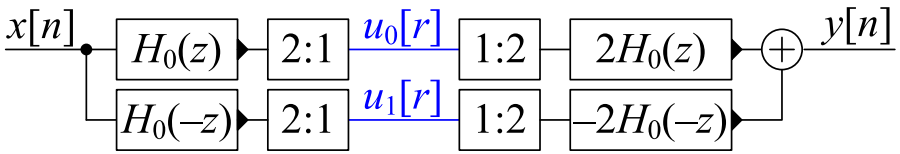
we want $T(z) = z^{-d} \Rightarrow P_0(z) = a_0 z^{-k}$, $P_1(z) = a_1 z^{k+1-d}$

$\Rightarrow H_0(z)$ has only two non-zero taps $\Rightarrow$ poor freq selectivity

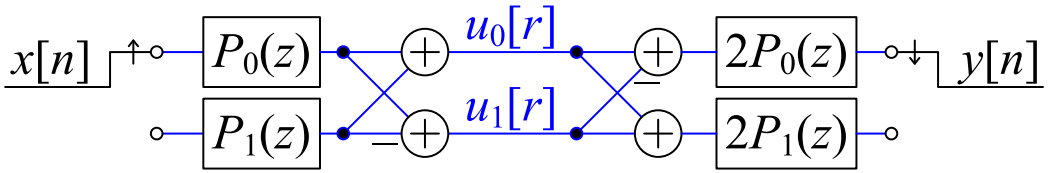
$\therefore$ Perfect reconstruction QMF filterbanks cannot have good freq selectivity

QMF Options

15: Subband Processing
Subband processing
2-band Filterbank
Perfect Reconstruction
Quadrature Mirror Filterbank (QMF)
Polyphase QMF
▷ QMF Options
Linear Phase QMF
IIR Allpass QMF
Tree-structured filterbanks
Summary
Merry Xmas



Polyphase decomposition:



$A(z) = 0 \Rightarrow$ no alias term

$T(z) = H_0^2(z) - H_1^2(z) = H_0^2(z) - H_0^2(-z) = 4z^{-1}P_0(z^2)P_1(z^2)$

Options:

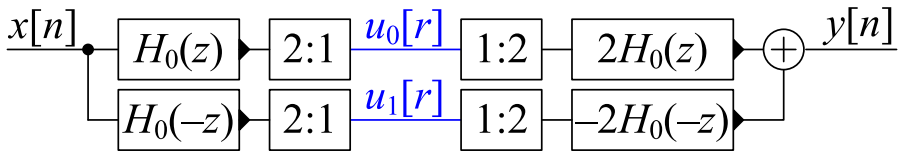
(A) Perfect Reconstruction: $T(z) = z^{-d} \Rightarrow H_0(z)$ is a bad filter.

(B) $T(z)$ is Linear Phase FIR:
$\Rightarrow$ Tradeoff: $\left|T(e^{j\omega})\right| \approx 1$ versus $H_0(z)$ stopband attenuation

(C) $T(z)$ is Allpass IIR: $H_0(z)$ can be Butterworth or Elliptic filter
$\Rightarrow$ Tradeoff: $\angle T(e^{j\omega}) \approx \tau\omega$ versus $H_0(z)$ stopband attenuation

Option (B): Linear Phase QMF

15: Subband Processing
Subband processing
2-band Filterbank
Perfect Reconstruction
Quadrature Mirror Filterbank (QMF)
Polyphase QMF
QMF Options
▷ Linear Phase QMF
IIR Allpass QMF
Tree-structured filterbanks
Summary
Merry Xmas



$T(z) \approx 1$

$H_0(z)$ order M, linear phase $\Rightarrow H_0(e^{j\omega}) = \pm e^{-j\omega\frac{M}{2}} \left|H_0(e^{j\omega})\right|$

$$\begin{aligned} T(e^{j\omega}) &= H_0^2(e^{j\omega}) - H_1^2(e^{j\omega}) = H_0^2(e^{j\omega}) - H_0^2(-e^{j\omega}) \\ &= e^{-j\omega M}\left|H_0(e^{j\omega})\right|^2 - e^{-j(\omega-\pi)M}\left|H_0(e^{j(\omega-\pi)})\right|^2 \\ &= e^{-j\omega M}\left(\left|H_0(e^{j\omega})\right|^2 - (-1)^M\left|H_0(e^{j(\pi-\omega)})\right|^2\right) \end{aligned}$$

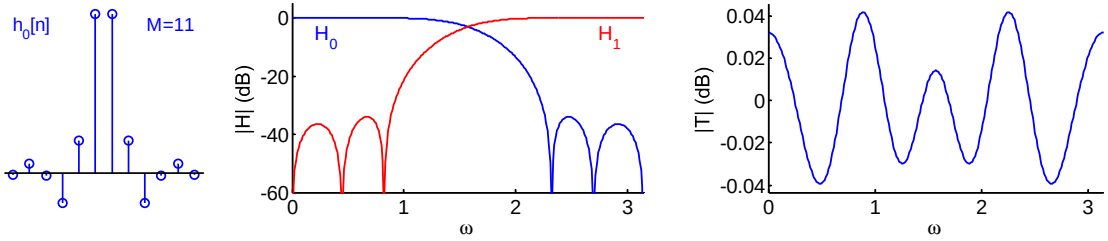
M even $\Rightarrow T(e^{j\frac{\pi}{2}}) = 0$ ☹ so choose M odd $\Rightarrow -(-1)^M = +1$

Select $h_0[n]$ by numerical iteration to minimize

$$\alpha \int_{\frac{\pi}{2}+\Delta}^{\pi} \left|H_0(e^{j\omega})\right|^2 d\omega + (1-\alpha)\int_0^{\pi}\left(\left|T(e^{j\omega})\right| - 1\right)^2 d\omega$$

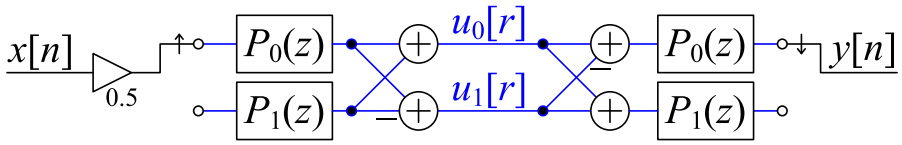
$\alpha \rightarrow$ balance between $H_0(z)$ being lowpass and $T(e^{j\omega}) \approx 1$

Johnston filter ($M = 11$):



Option (C): IIR Allpass QMF

15: Subband Processing
Subband processing
2-band Filterbank
Perfect Reconstruction
Quadrature Mirror Filterbank (QMF)
Polyphase QMF
QMF Options
Linear Phase QMF
▷ IIR Allpass QMF
Tree-structured filterbanks
Summary
Merry Xmas



$|T(z)| = 1$

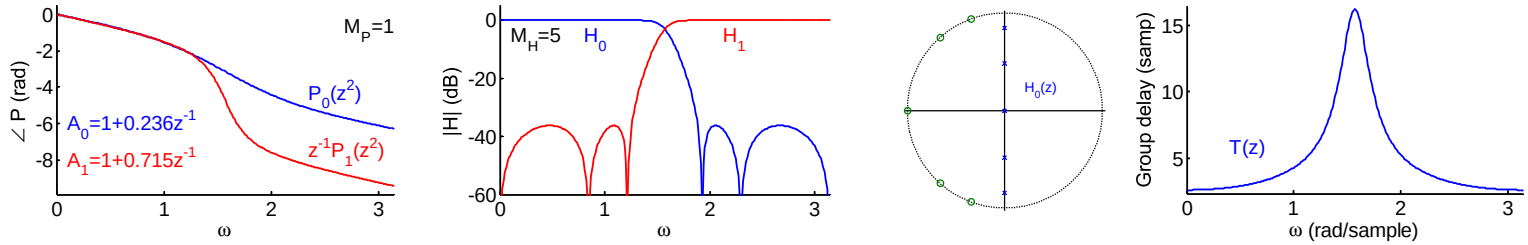
Choose $P_0(z)$ and $P_1(z)$ to be allpass IIR filters:

$$H_{0,1}(z) = \frac{1}{2}\left(P_0(z^2) \pm z^{-1}P_1(z^2)\right), \qquad G_{0,1}(z) = \pm 2H_{0,1}(z)$$

$A(z) = 0 \Rightarrow$ No aliasing

$T(z) = H_0^2 - H_1^2 = \ldots = z^{-1}P_0(z^2)P_1(z^2)$ is an allpass filter.

$H_0(z)$ can be made a Butterworth or Elliptic filter with $M_H = 4M_P + 1$:



Phase cancellation: $\angle z^{-1}P_1 = \angle P_0 + \pi$; Ripples in H_0 and H_1 cancel.

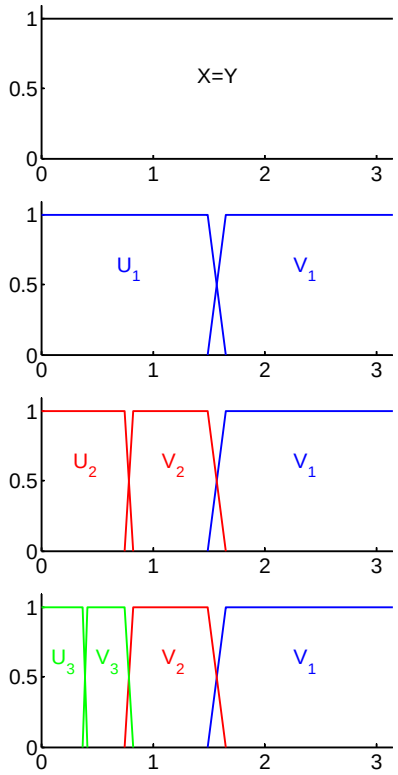
Tree-structured filterbanks

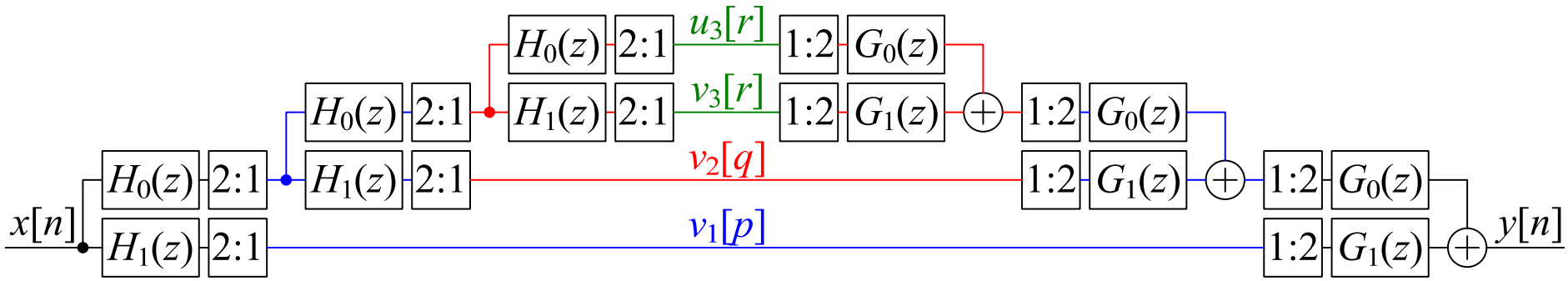
A *half-band filterbank* divides the full band into two equal halves.

You can repeat the process on either or both of the signals $u_1[p]$ and $v_1[p]$.

Dividing the lower band in half repeatedly results in an *octave band filterbank*. Each subband occupies one octave (= a factor of 2 in frequency) except the first subband.

The properties "perfect reconstruction" and "allpass" are preserved by the iteration.





Summary

15: Subband Processing
Subband processing
2-band Filterbank
Perfect Reconstruction
Quadrature Mirror Filterbank (QMF)
Polyphase QMF
QMF Options
Linear Phase QMF
IIR Allpass QMF
Tree-structured filterbanks
▷ Summary
Merry Xmas

- Half-band filterbank:
  - Reconstructed output is $T(z)X(z) + A(z)X(-z)$
  - Unwanted alias term is $A(z)X(-z)$
- Perfect reconstruction: imposes strong constraints on analysis filters $H_i(z)$ and synthesis filters $G_i(z)$.
- Quadrature Mirror Filterbank (QMF) adds an additional symmetry constraint $H_1(z) = H_0(-z)$.
  - Perfect reconstruction now impossible except for trivial case.
  - Neat polyphase implementation with $A(z) = 0$
  - Johnston filters: Linear phase with $T(z) \approx 1$
  - Allpass filters: Elliptic or Butterworth with $|T(z)| = 1$
- Can iterate to form a tree structure with equal or unequal bandwidths.

See Mitra chapter 14 (which also includes some perfect reconstruction designs).

Merry Xmas

FORMULA SHEET AVAILABLE IN EXAM

The following formulae will be available in the exam:

Where a question requires a numerical answer, it must be given as a fully evaluated decimal number and not as an unevaluated arithmetic expression.

Notation

- All signals and filter coefficients are real-valued unless explicitly noted otherwise.
- Unless otherwise specified, upper and lower case letters are used for sequences and their z-transforms respectively. The signal at a block diagram node V is $v[n]$ and its z-transform is $V(z)$.
- $x[n] = [a, b, c, d, e, f]$ means that $x[0] = a, \ldots x[5] = f$ and that $x[n] = 0$ outside this range.
- $\Re(z)$, $\Im(z)$, z^*, $|z|$ and $\angle z$ denote respectively the real part, imaginary part, complex conjugate, magnitude and argument of a complex number z.
- The expected value of x is denoted $E\{x\}$.
- In block diagrams: solid arrows denote the direction of signal flow; an open triangle denotes a gain element with the gain indicated adjacently; a "+" in a circle denotes an adder/subtractor whose inputs may be labelled "+" or "−" according to their sign; the sample rate, f, of a signal in Hz may be indicated in the form "@f".

Abbreviations

BIBO	Bounded Input, Bounded Output	IIR	Infinite Impulse Response
CTFT	Continuous-Time Fourier Transform	LTI	Linear Time-Invariant
DCT	Discrete Cosine Transform	MDCT	Modified Discrete Cosine Transform
DFT	Discrete Fourier Transform	PSD	Power Spectral Density
DTFT	Discrete-Time Fourier Transform	SNR	Signal-to-Noise Ratio
FIR	Finite Impulse Response		

Standard Sequences

- $\delta[n] = 1$ for $n = 0$ and 0 otherwise.
- $\delta_{\text{condition}}[n] = 1$ whenever "condition" is true and 0 otherwise.
- $u[n] = 1$ for $n \geq 0$ and 0 otherwise.

Geometric Progression

- $\sum_{n=0}^{r} \alpha^n z^{-n} = \frac{1-\alpha^{r+1} z^{-r-1}}{1-\alpha z^{-1}}$ provided that $\alpha z^{-1} \neq 1$.
- $\sum_{n=0}^{\infty} \alpha^n z^{-n} = \frac{1}{1-\alpha z^{-1}}$ provided that $\left|\alpha z^{-1}\right| < 1$.

Forward and Inverse Transforms

z:	$X(z) = \sum_{-\infty}^{\infty} x[n] z^{-n}$	$x[n] = \frac{1}{2\pi j} \oint X(z) z^{n-1} dz$
CTFT:	$X(j\Omega) = \int_{-\infty}^{\infty} x(t) e^{-j\Omega t} dt$	$x(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(j\Omega) e^{j\Omega t} d\Omega$
DTFT:	$X(e^{j\omega}) = \sum_{-\infty}^{\infty} x[n] e^{-j\omega n}$	$x[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} X(e^{j\omega}) e^{j\omega n} d\omega$
DFT:	$X[k] = \sum_{0}^{N-1} x[n] e^{-j2\pi \frac{kn}{N}}$	$x[n] = \frac{1}{N} \sum_{0}^{N-1} X[k] e^{j2\pi \frac{kn}{N}}$
DCT:	$X[k] = \sum_{n=0}^{N-1} x[n] \cos \frac{2\pi(2n+1)k}{4N}$	$x[n] = \frac{X[0]}{N} + \frac{2}{N} \sum_{n=1}^{N-1} X[k] \cos \frac{2\pi(2n+1)k}{4N}$
MDCT:	$X[k] = \sum_{n=0}^{2N-1} x[n] \cos \frac{2\pi(2n+1+N)(2k+1)}{8N}$	$y[n] = \frac{1}{N} \sum_{0}^{N-1} X[k] \cos \frac{2\pi(2n+1+N)(2k+1)}{8N}$

Convolution

DTFT:	$v[n] = x[n] * y[n] \triangleq \sum_{r=-\infty}^{\infty} x[r] y[n-r]$	$\Leftrightarrow$	$V\left(e^{j\omega}\right) = X\left(e^{j\omega}\right) Y\left(e^{j\omega}\right)$
	$v[n] = x[n] y[n]$	$\Leftrightarrow$	$V\left(e^{j\omega}\right) = \frac{1}{2\pi} X\left(e^{j\omega}\right) \circledast Y\left(e^{j\omega}\right) \triangleq \frac{1}{2\pi} \int_{-\pi}^{\pi} X\left(e^{j\theta}\right) Y\left(e^{j(\omega-\theta)}\right) d\theta$
DFT:	$v[n] = x[n] \circledast_N y[n] \triangleq \sum_{r=0}^{N-1} x[r] y[(n-r)_{\bmod N}]$	$\Leftrightarrow$	$V[k] = X[k] Y[k]$
	$v[n] = x[n] y[n]$	$\Leftrightarrow$	$V[k] = \frac{1}{N} X[k] \circledast_N Y[k] \triangleq \frac{1}{N} \sum_{r=0}^{N-1} X[r] Y[(k-r)_{\bmod N}]$

Group Delay

The group delay of a filter, $H(z)$, is $\tau_H(e^{j\omega}) = -\frac{d\angle H(e^{j\omega})}{d\omega} = \Re\left(\frac{-z}{H(z)} \frac{dH(z)}{dz}\right)\Big|_{z=e^{j\omega}} = \Re\left(\frac{\mathscr{F}(nh[n])}{\mathscr{F}(h[n])}\right)$ where $\mathscr{F}(\,)$ denotes the DTFT.

Order Estimation for FIR Filters

Three increasingly sophisticated formulae for estimating the minimum order of an FIR filter with unity gain passbands:

1. $M \approx \frac{a}{3.5\Delta\omega}$
2. $M \approx \frac{a-8}{2.2\Delta\omega}$
3. $M \approx \frac{a-1.2-20\log_{10} b}{4.6\Delta\omega}$

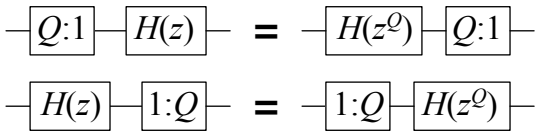
where a =stop band attenuation in dB, b = peak-to-peak passband ripple in dB and $\Delta\omega$ = width of smallest transition band in radians per sample.

z-plane Transformations

A lowpass filter, $H(z)$, with cutoff frequency ω_0 may be transformed into the filter $H(\hat{z})$ as follows:

Target $H(\hat{z})$	Substitute	Parameters
Lowpass $\hat{\omega} < \hat{\omega}_1$	$z^{-1} = \frac{\hat{z}^{-1}-\lambda}{1-\lambda\hat{z}^{-1}}$	$\lambda = \frac{\sin\left(\frac{\omega_0-\hat{\omega}_1}{2}\right)}{\sin\left(\frac{\omega_0+\hat{\omega}_1}{2}\right)}$
Highpass $\hat{\omega} > \hat{\omega}_1$	$z^{-1} = -\frac{\hat{z}^{-1}+\lambda}{1+\lambda\hat{z}^{-1}}$	$\lambda = \frac{\cos\left(\frac{\omega_0+\hat{\omega}_1}{2}\right)}{\cos\left(\frac{\omega_0-\hat{\omega}_1}{2}\right)}$
Bandpass $\hat{\omega}_1 < \hat{\omega} < \hat{\omega}_2$	$z^{-1} = -\frac{(\rho-1)-2\lambda\rho\hat{z}^{-1}+(\rho+1)\hat{z}^{-2}}{(\rho+1)-2\lambda\rho\hat{z}^{-1}+(\rho-1)\hat{z}^{-2}}$	$\lambda = \frac{\cos\left(\frac{\hat{\omega}_2+\hat{\omega}_1}{2}\right)}{\cos\left(\frac{\hat{\omega}_2-\hat{\omega}_1}{2}\right)}, \rho = \cot\left(\frac{\hat{\omega}_2-\hat{\omega}_1}{2}\right)\tan\left(\frac{\omega_0}{2}\right)$
Bandstop $\hat{\omega}_1 \not< \hat{\omega} \not< \hat{\omega}_2$	$z^{-1} = \frac{(1-\rho)-2\lambda\hat{z}^{-1}+(\rho+1)\hat{z}^{-2}}{(\rho+1)-2\lambda\hat{z}^{-1}+(1-\rho)\hat{z}^{-2}}$	$\lambda = \frac{\cos\left(\frac{\hat{\omega}_2+\hat{\omega}_1}{2}\right)}{\cos\left(\frac{\hat{\omega}_2-\hat{\omega}_1}{2}\right)}, \rho = \tan\left(\frac{\hat{\omega}_2-\hat{\omega}_1}{2}\right)\tan\left(\frac{\omega_0}{2}\right)$

Noble Identities



Multirate Spectra

Upsample: $v[n] \xrightarrow{1:Q} x[r] \quad \Rightarrow \quad x[r] = \begin{cases} v\left[\frac{r}{Q}\right] & \text{if } Q \mid r \\ 0 & \text{if } Q \nmid r \end{cases} \quad \Rightarrow \quad X(z) = V(z^Q)$

Downsample: $v[n] \xrightarrow{Q:1} y[m] \quad \Rightarrow \quad y[m] = v[Qm] \quad \Rightarrow \quad Y(z) = \frac{1}{Q}\sum_{k=0}^{Q-1} V\left(e^{\frac{-j2\pi k}{Q}} z^{\frac{1}{Q}}\right)$

Multirate Commutators

Input Commutator	Output Commutator