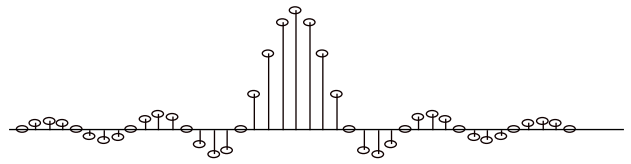
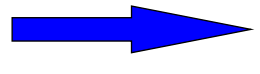
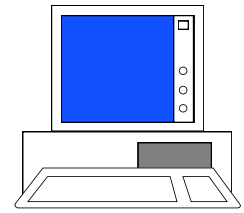


Introduction to Digital Signal Processing

Dr. Surya Singh, Prof. Brian Lovell & Dr. Paul Pounds

Lecture #6

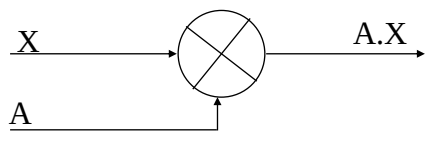




The Three Basic Elements of Linear Processing

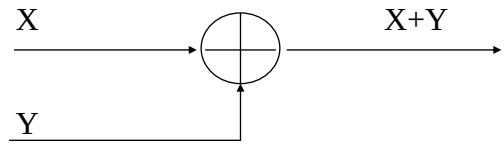
Storage; (Delay, Register)

Scaling; (Weighting, Product, Multiplication)



The Three Basic Elements of Linear Processing

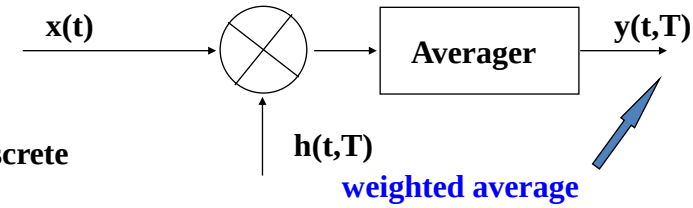
Summation; (Addition, Accumulate)



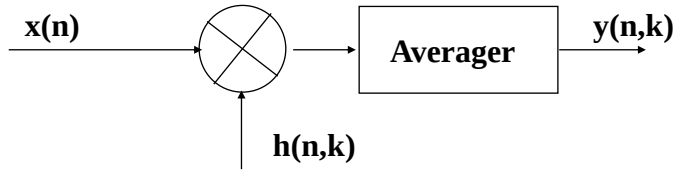
With these three operations we can perform virtually all signal processing. More complicated operations such as convolution, integration, filtering, Fourier transforms etc. are constructed from these simple primitives.

Basic Signal Processing

Continuous

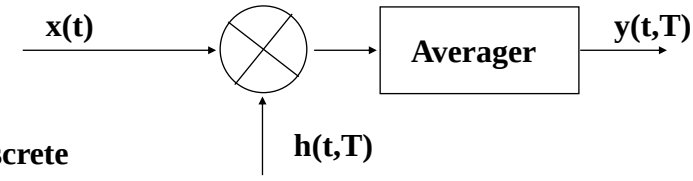


Discrete

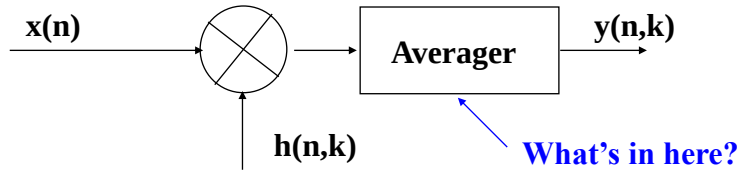


Basic Signal Processing

Continuous

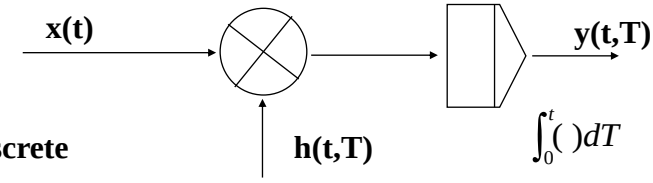


Discrete

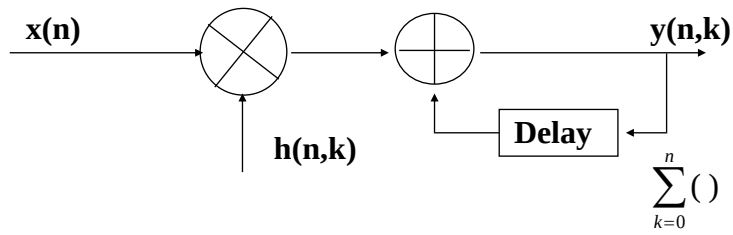


Basic Signal Processing

Continuous



Discrete



Another View

e.g. convolution

x(n) = 1 2 3 4 5
h(n) = 3 2 1

x(k)	**0 0 1 2 3 4 5**	**0 0 1 2 3 4 5**	**0 0 1 2 3 4 5**	
h(n,k)	**1 2 3 0 0 0 0**	**0 1 2 3 0 0 0**	**0 0 1 2 3 0 0**	**h(n-k)**
y(n,k)	**3**	**2 6**	**1 4 9**	
y(n)	**3**	**8**	**14**	

Sum over all k

Notice the gain

Notation

- There is some notational confusion in the preceding description.
  - When we say y(n), do we mean the value of y at sample n (a scalar), or do we mean the ordered set of y values for all n (a vector or function)?
- Bad notation has crept into DSP and it is hard to eradicate. Matrix notation is often a clearer way of expressing data manipulation.

Matrix Formulation of Convolution

$$\mathbf{y} = \mathbf{H}\mathbf{x}$$

$$\begin{bmatrix} 3 \\ 8 \\ 14 \\ 20 \\ 26 \\ 14 \\ 5 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 2 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 2 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 \end{bmatrix} . \begin{bmatrix} 0 \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 0 \\ 0 \end{bmatrix}$$

Toeplitz Matrix

Typical Linear Processors

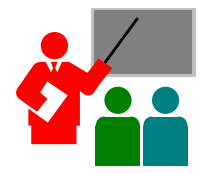
- Convolution h(n,k)=h(n-k)
- Cross Correlation h(n,k)=h(n+k)
- Auto Correlation h(n,k)=x(k-n)
- Cosine Transform h(n,k)= $\cos\left(\frac{2\pi}{N}nk\right)$
- Sine Transform h(n,k)= $\sin\left(\frac{2\pi}{N}nk\right)$
- Fourier Transform h(n,k)= $\exp\left(j\frac{2\pi}{N}nk\right)$

Typical Linear Processors

- We also have
  - Heterodyne — sinusoid (complex)
  - Window — window function
  - Shade — weighting function
  - Scramble — pseudo-random sequence
- These operations are simple cross-multiplications (".*" in Matlab) of the data with particular functions.

Demonstration

- Now that the administration is over, it's time for a quick demonstration
- The Matlab demonstration qzero.m is a script which shows the effect of quantisation noise on a speech signal.



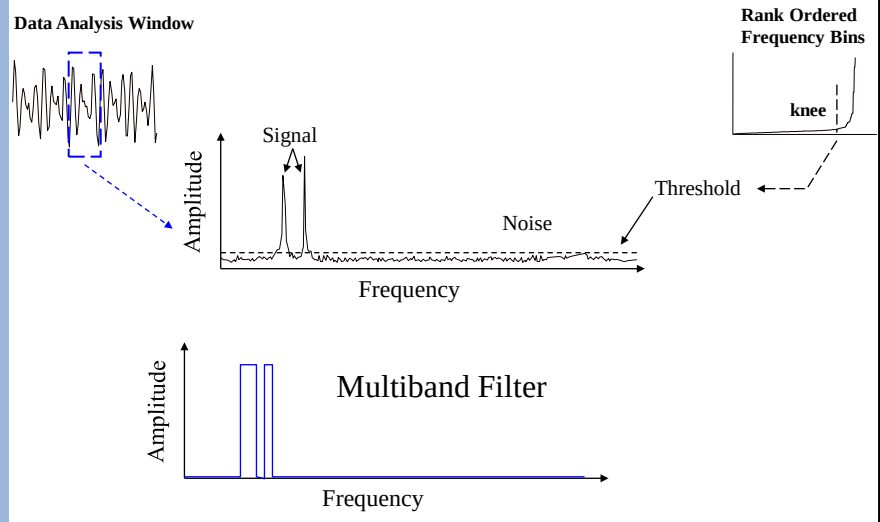
Application

Noisy

Clean

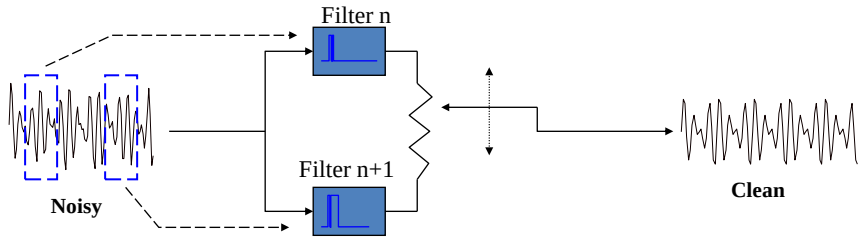
- This is demonstration of using adaptive filtering based on the DFT to remove noise from an audio recording.
- The method is based on analysing the music every few milliseconds with the DFT; rank ordering the frequency bins and thresholding to separate signal from noise; and then designing a filter to remove the noise.
- Interpolation is used to move smoothly from one filter to another.

Adaptive Filtering



Filter Interpolation

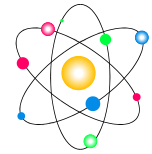
We “ping-pong” smoothly between the filter impulse responses to provide smooth filter transitions.



Introducing the Lecturer

Brian Lovell
School of ITEE
The University of Queensland 4072
Tel- (07) 33654134
Email lovell@itee.uq.edu.au
http://www.itee.uq.edu.au/~lovell

Quote

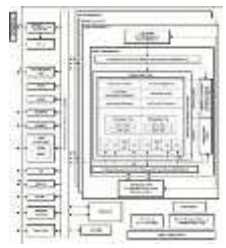


"Now that you have learnt about adding, minussing, multiplying and dividing, you can do any sum. Even an atomic scientist only really uses these four operations."

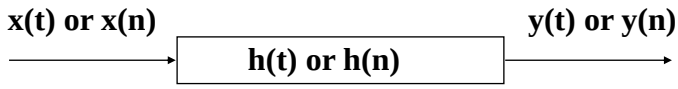
— Mrs S. Boal, Grade 1, Oakleigh Primary School, 1968

DSP Operations

- Virtually all DSP is point-by-point multiply and sum; e.g., filter, correlation, DFT
- Hence the most common DSP operation is MACD (multiply,accumulate, and delay) which is a single vector instruction on DSP chips (e.g. TMS series).



Sinusoids and Linear Systems



If $x(t) = A\cos(\omega_0 t + \theta_0)$

or $x(n) = A\cos(\omega_0 nt + \theta_0)$

then in steady state

$$y(t) = AC(\omega_0)\cos(\omega_0 t + \theta_0 + \theta(\omega_0))$$
$$y(n) = AC(\omega_0 T)\cos(\omega_0 nt + \theta_0 + \theta(\omega_0 T))$$

Sinusoids and Linear Systems

$$y(t) = AC(\omega_0)\cos(\omega_0 t + \theta_0 + \theta(\omega_0))$$
$$y(n) = AC(\omega_0 T)\cos(\omega_0 nt + \theta_0 + \theta(\omega_0 T))$$

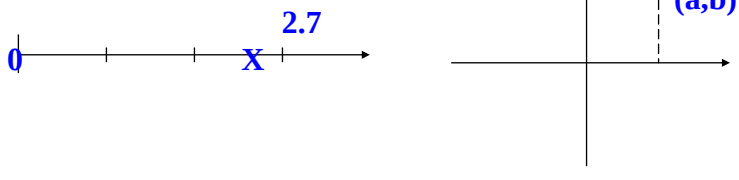
- The pair of numbers $C(\omega_0)$ and $\theta(\omega_0)$ are the complex gain of the system at the frequency ω_0 . They are respectively, the magnitude response and the phase response at the frequency ω_0 .

Why Use Sinusoids?

- Why probe system with sinusoids?
- Sinusoids are eigenfunctions of linear systems???
- What the hell does that mean?
- Sinusoid in implies sinusoid out
- Only need to know phase and magnitude (two parameters) to fully describe output rather than whole waveform
  - sine + sine = sine
  - derivative of sine = sine (phase shifted - cos)
  - integral of sine = sine (-cos)
- Sinusoids maintain orthogonality after sampling (not true of most orthogonal sets)

Notation

- The use of complex numbers to describe signals can be very confusing; it is often better to think of the data in terms of ordered pairs.
- j is not the square root of -1
  - Offers no insight, confuses people
- Isomorphism between number, point on line,
  vector, ordered pair



Notation

- Damage control (a,0) = a; (0,b)=jb
- Consider j as an operator
  - a = (a,0) ⇨ (0,a) = ja
- Need inverse operator $j^{-1} = \frac{1}{j}$
- Note division is not really possible

$$\frac{1}{a+jb} = \frac{a-jb}{(a+jb)(a-jb)} = \frac{a-jb}{a^2+b^2}$$

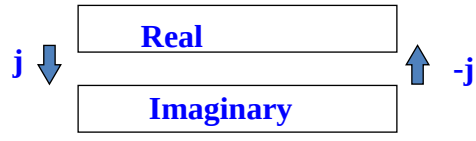
poor notation

- So is j the square root of -1?

  No, both j1 and -j1 are the roots

Notation

- Multiplication by j moves a data sample from one register (real) to the other register(imaginary). Another multiplication by j moves it back again with a sign change.



Application to Sinusoids

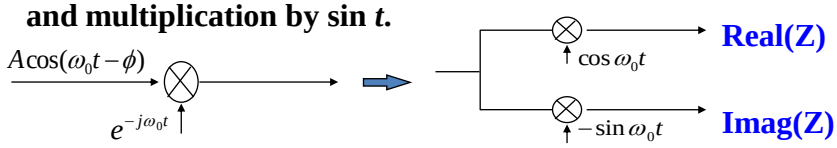
Represent ordered pair as vector length R at angle theta

$$(a,b) = a + jb = R(\cos\theta + j\sin\theta)$$

Use Euler's formula

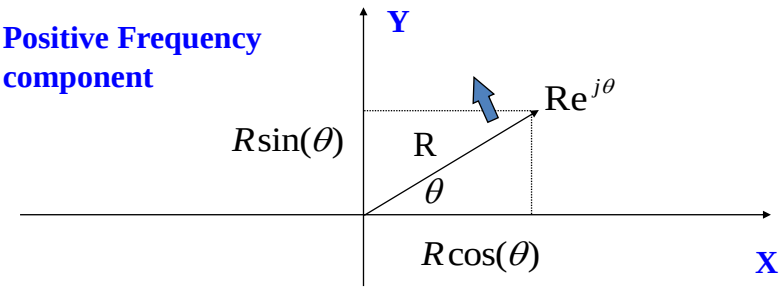
$$e^{jt} = \cos t + j\sin t$$

Multiplication by e^{jt} is equivalent to multiplication by cos *t* and multiplication by sin *t*.



Complex Numbers and Phasors

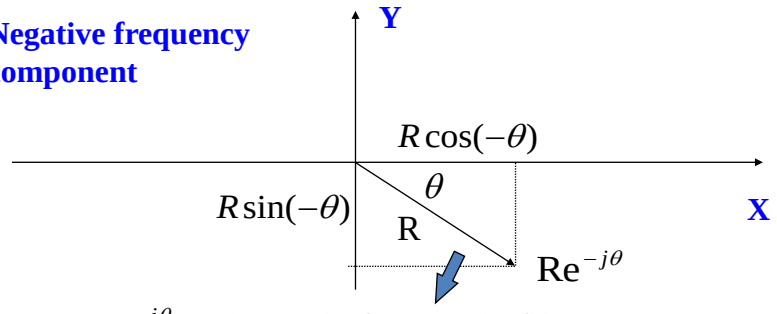
Positive Frequency component



$$\begin{aligned} Re^{j\theta} &= (R\cos\theta, R\sin\theta) \\ &= R\cos\theta + jR\sin\theta \\ &= R(\cos\theta + j\sin\theta) \end{aligned}$$

Complex Numbers and Phasors

Negative frequency component

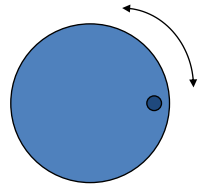


$$\begin{aligned}\mathrm{Re}^{-j\theta} &= (R\cos(-\theta), R\sin(-\theta)) \\ &= R\cos(-\theta) + jR\sin(-\theta) \\ &= R(\cos\theta - j\sin\theta)\end{aligned}$$

Positive and Negative Frequencies

- Frequency is the derivative of phase not 1/period = repetition rate.
- Hence both positive and negative frequencies are possible.
- Compare
  - velocity vs speed
  - frequency vs repetition rate

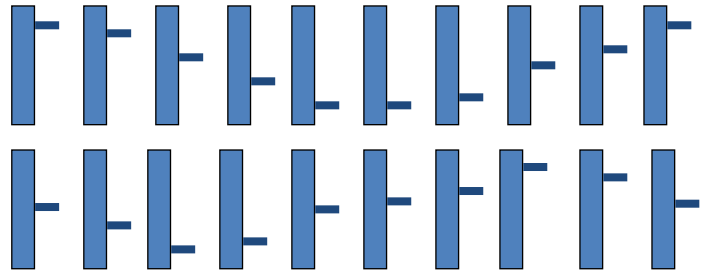
Rotating wheel and peg



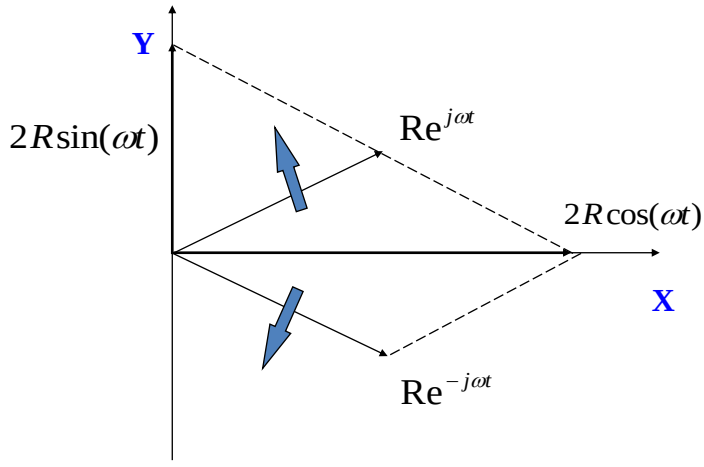
Need both top and front view to determine rotation

Top View

Front View



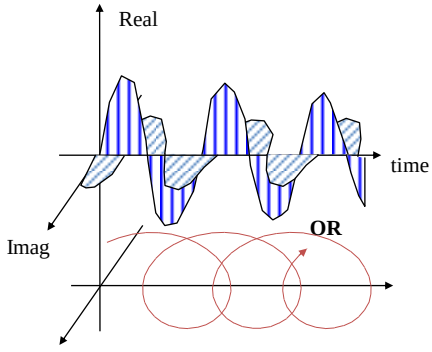
Euler's Form: Sum and Difference of Rotating Phasors



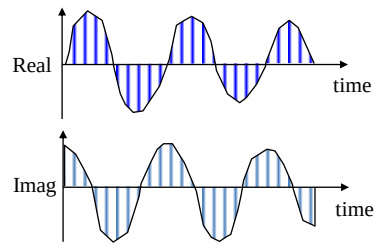
Alternate Representations for Complex Data

- Three Dimensional
- Cartesian;two graphs; real and imaginary
- Polar;two graphs; magnitude and phase; Bode
- Locus of complex values; Nyquist or Argand

Graph Types

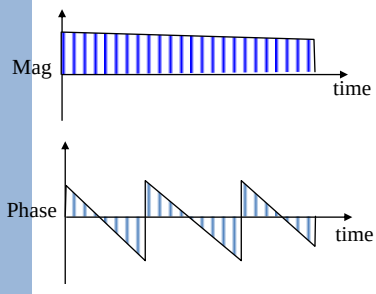


Three Dimensional

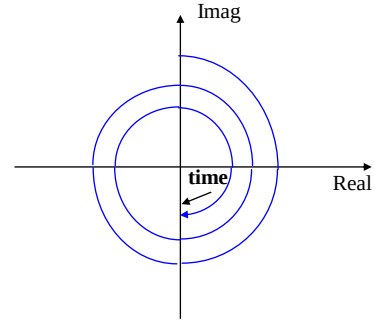


Cartesian

Graph Types



Polar; Bode



Nyquist; Argand

Heterodyning

Given

$$A\cos(\omega_0 t-\phi)=\frac{A}{2}e^{+j(\omega_0 t-\phi)}+\frac{A}{2}e^{-j(\omega_0 t-\phi)}$$

$$=\frac{A}{2}e^{-j\phi}e^{+j\omega_0 t}+\frac{A}{2}e^{+j\phi}e^{-j\omega_0 t}$$

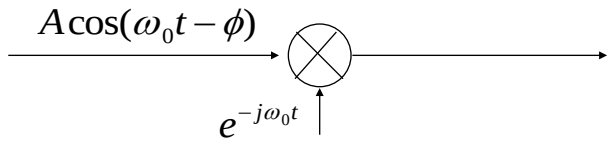
with ω_0 known, determine *A* and ϕ, be clever but use a DC voltmeter.

Heterodyning

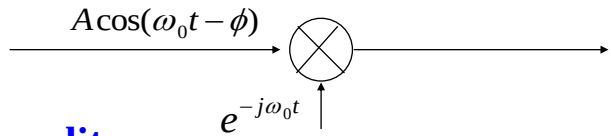
Solution: Stop one of the phasors from rotating!

Form

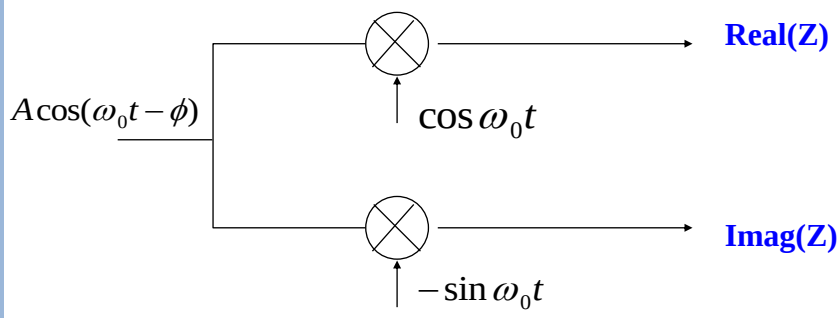
$$A\cos(\omega_0 t-\phi)e^{-j\omega_0 t}=\frac{A}{2}e^{-j\phi}+\frac{A}{2}e^{+j\phi}e^{-j2\omega_0 t}$$



Heterodyning



in reality

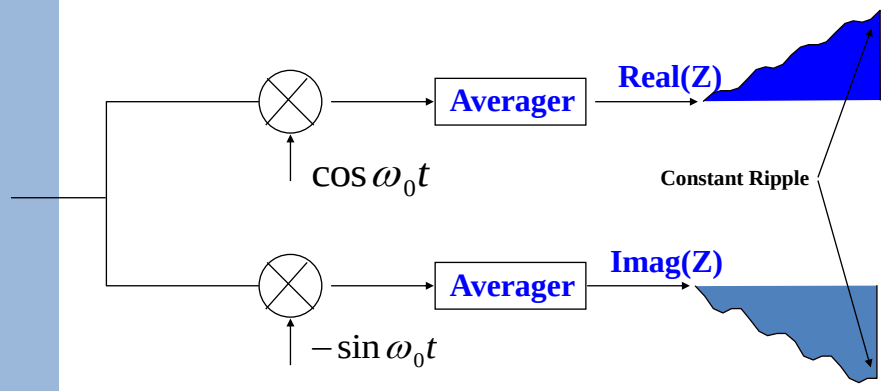


Heterodyning

- Hetero = Different; Dyne = Move
  - Move signal of interest (to DC in this case)
- Sometimes called homodyning
  Homo = Same; Dyne = Move
  - Move signal of interest to DC
- Output is DC plus double frequency component
- Integrate output and plot result
- Output is correct at multiples of the period T
- Divide integral by the time to get average DC output
  - correct at multiples of T

Averager

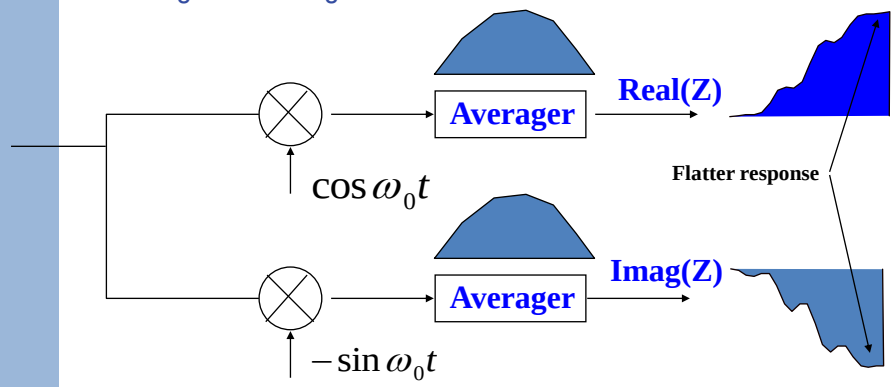
- Averager is a low pass filter which may (or may not) completely reject a second signal.



Averager

hetdemo
hetdemo2

- Filtering can be improved by using a window function to form a weighted average



Fourier Series

- What we have produced is a processor to calculate one coefficient of the complex Fourier Series
- Fourier Series Coefficients = Heterodyne and average over observation interval T

$$C_k = \frac{1}{T}\int_0^T h(t)e^{-j\frac{2\pi}{T}kt}\,dt$$

Fourier Transform

- If we change the limits of integration to the entire real line, remove the division by T, and make the frequency variable continuous, we get the Fourier Transform

$$C(\omega) = \int_{-\infty}^{+\infty} h(t) e^{-j\omega t} dt$$