

Generation and Analysis of Random Processes

Lecture 17

Spring 2002

Outline

We will first develop a method to construct a discrete random process with an arbitrary power spectrum.

We will then analyze the spectra using the periodogram and corlogram methods.

Lecture 17

1

Filter Design

The random process will be created by passing a white noise sequence through a discrete filter.

A FIR filter will be used.

The design uses inverse transformation of the desired frequency response.

The filter is designed in the frequency domain and implemented as a difference equation.

Lecture 17

2

Design Procedure

1. Describe the desired frequency response $H_d(f)$ at N points corresponding to the frequencies $f_k = k/2N$, $0 \leq k \leq N - 1$. These points are in the normalized frequency interval $0 \leq f < 0.5$.
2. Compute the an impulse response sequence $h_1(t_n)$ at N points, $t_n = n$, $0 \leq n \leq N - 1$.
3. Shift the impulse response by $M/2$ and select the first $M + 1$ points of the shifted sequence. Call the new sequence h_2 .
4. Transform the new sequence to obtain the frequency response $H_{fft}(f)$. Compare to $H_d(f)$.

Lecture 17

3

Example

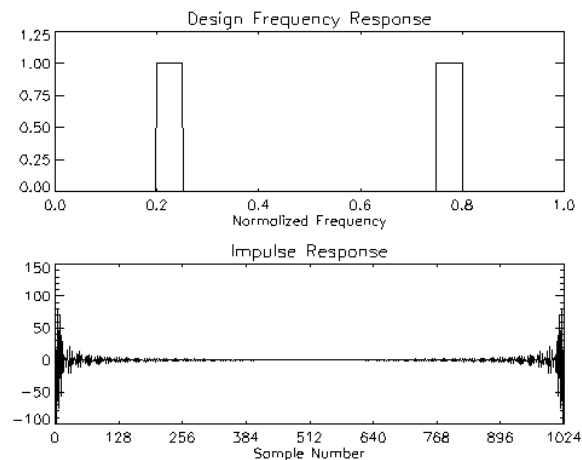
Design a bandpass filter such that

$$H_d(f) = \begin{cases} 1, & 0.2 < f < 0.25 \\ 0, & \text{elsewhere} \end{cases}$$

Step 1: Construct a response function over N frequency points.
Here we will use $N = 512$.

```
N=512
flow=0.2 & fhigh=0.25 ;Edge frequencies
fpass=[flow*2*N,fhigh*2*N] ;Edge frequency indexes
Hd=fltarr(N) & Hd[fpass[0]:fpass[1]]=1.0
Hd=[Hd,0,Reverse(Hd[0:N-2])] ;Symmetrical response function
```

$H_d(f)$ and $h_1(n)$



Example (continued)

Step 2: Compute the impulse response using the FFT.

```
h=Float(FFT(Hd,/Inverse))
```

The impulse response is shown on the preceding page.

Note that both the frequency response and the impulse response are symmetric about their midpoints.

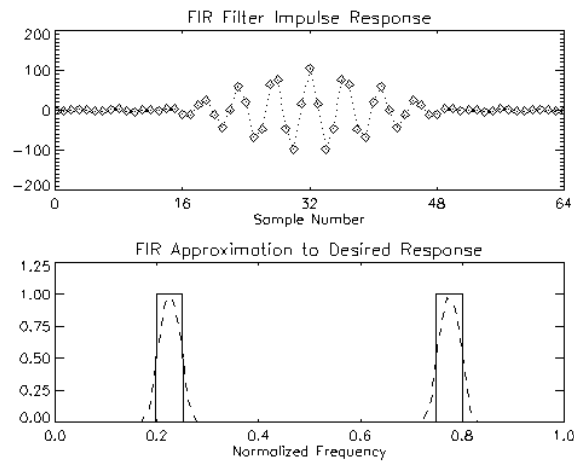
Example (continued)

The next step is to shorten the impulse response to a reasonable length. The chosen length $M + 1$ is a design parameter.

Step 3: Shift h by $M/2$ and contour with a window function to reduce aliasing. Save the result in an array of length $M + 1$.

```
M=64
h=Shift(h,M/2)
w=0.54-0.46*cos(2*!pi*findgen(M+1)/M)
h=h[0:M]*w
;Plot the impulse response
plot,h,xtickv=[0,M/4,M/2,3*M/4,M],xticks=4,psym=-4,$
linestyle=1,yrange=[-150,150],$
title='FIR Filter Impulse Response',xtitle='Sample Number'
```

$h(n)$ and $H_{fft}(f)$



Example (continued)

Step 4: The frequency response H_{fft} shown on the preceding page can be computed by taking the inverse transform of the impulse response. (The graph of H_{fft} has been rescaled to account for different number of points in H_d and H_{fft} .)

```
;Plot the frequency response
f=findgen(2*N)/2/N
plot,f,Hd,ytickv=findgen(6)/4,yticks=5,$
    title='FIR Approximation to Desired Response',$
    xtitle='Normalized Frequency'

Hfft=abs(fft(h))
fst=findgen(n_elements(h))/n_elements(h)
oplot,fst,Hfft*N_Elements(Hfft)/N_Elements(Hd),linestyle=2
```

Random Process Synthesis

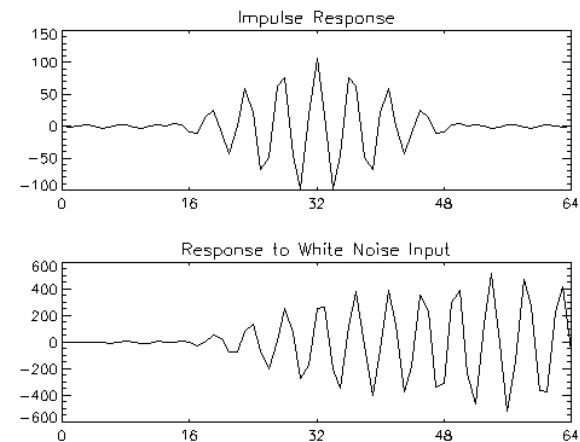
Step 5: Generate white noise and pass it through the system.

Use the program `y=filter(b,a,x)` with `b=h` and `a=1`.

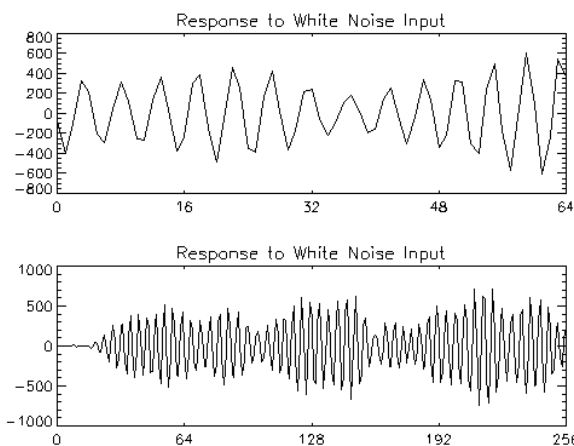
```
x=randomn(seed,1024)
y=filter(h,1,x)
```

Shown below are plots of the response to white noise. The effect of the system impulse response is evident. The noise characteristics are those of a narrowband random process.

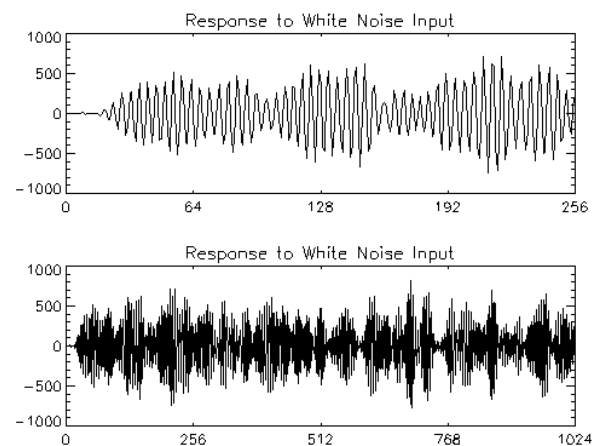
Response to White Noise Input



Response to White Noise Input



Response to White Noise Input



Random Process Generation (continued)

Step 6: Investigate the autocorrelation function $R_{yy}(n)$

We can compute R_{yy} by convolving y with the reversed y sequence.

We can also compute R_{yy} by the cross-correlation theorem:

$$R_{yy}(\tau) = \iint_{-\infty}^{\infty} R_{xx}(\tau + \alpha - \beta)h(\alpha)h(\beta)d\alpha d\beta$$

With $R_{xx}(\tau) = \sigma_x^2\delta(\tau)$,

$$\begin{aligned} R_{yy}(\tau) &= \sigma_x^2 \iint_{-\infty}^{\infty} \delta(\tau + \alpha - \beta)h(\alpha)h(\beta)d\alpha d\beta \\ &= \sigma_x^2 \int_{-\infty}^{\infty} h(\alpha)h(\tau + \alpha)d\alpha = \sigma_x^2 R_{hh}(\tau) \end{aligned}$$

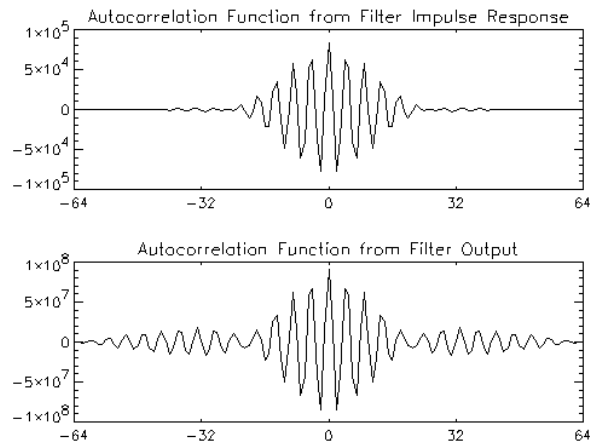
$R_{hh}(\tau)$ can be computed by convolving h with the reversed version of itself.

R_{yy} Calculations

```
rh=convolve(h,reverse(h))
rx=findgen(n_elements(rh))-n_elements(rh)/2
plot,rx,rh,xtickv=[-64,-32,0,32,64],xticks=4,$
    title='Autocorrelation Function from Filter Impulse Response'
ry=convolve(y,reverse(y))
ny=n_elements(y)
nh=n_elements(h)
ry=ry[ny-nh:ny+nh-2]
plot,rx,ry,xtickv=[-64,-32,0,32,64],xticks=4,$
    title='Autocorrelation Function from Filter Output'
```

The graphs, shown below, differ only by a scale factor.

Comparison of R_{yy} Calculations



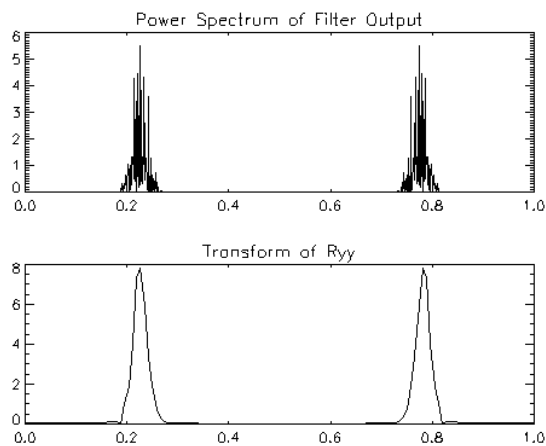
Example (continued)

Step 7: Calculate the output power spectrum using the Wiener-Khinchine Theorem and compare to the expected result.

```
Sry=abs(fft(ry))/n_elements(ry)
Sy=Abs(FFT(y))^2/varx/N
f=findgen(2*N)/2/N
fr=findgen(n_elements(rh))/(n_elements(rh)-1)
plot,f,Sy,title='Power Spectrum of Filter Output'
plot,fr,Sry/(2.0*N)^2*n_elements(ry),title='Transform of Ryy'
```

The spectral curves are compared on the next slide.

Power Spectrum Calculations



Periodogram Analysis

Periodogram estimation of the power spectral density is done by transforming and averaging many short sections of the random process. The program PERIODOGRAM.PRO is used to estimate the spectrum with

```
D=128 ;The size of the analysis window
NS=64 ;The shift of the analysis window
sp=PERIODOGRAM(Y,Y,D,NS,WINDOW=1) ;Hamming window
```

Correlogram Analysis

Correlogram estimation of the power spectral density is done by transforming the autocorrelation sequence of the random process. The program `CORRELOGRAMPSD.PRO` is used to estimate the spectrum.

`LAG=128 ;The maximum lag to be used.`

`NF=64 ;Number of frequency points to calculate.`

`sc=CORRELOGRAMPSD(Y,Y,LAG,NF)`

Power Spectrum Comparison

