6.011 Problem Set 5 Question 4

Michael Salib

October 21, 2002

I wrote all of the code for this problem set using Python with Numerical Extensions. The source code is attached.

1. (a)

$$\mu_v = \frac{1}{2}1 + \frac{1}{2}(-1) = 0$$
$$R_{vv}[m] = E[v[n+m]v[v]]$$

For $m \neq 0$,

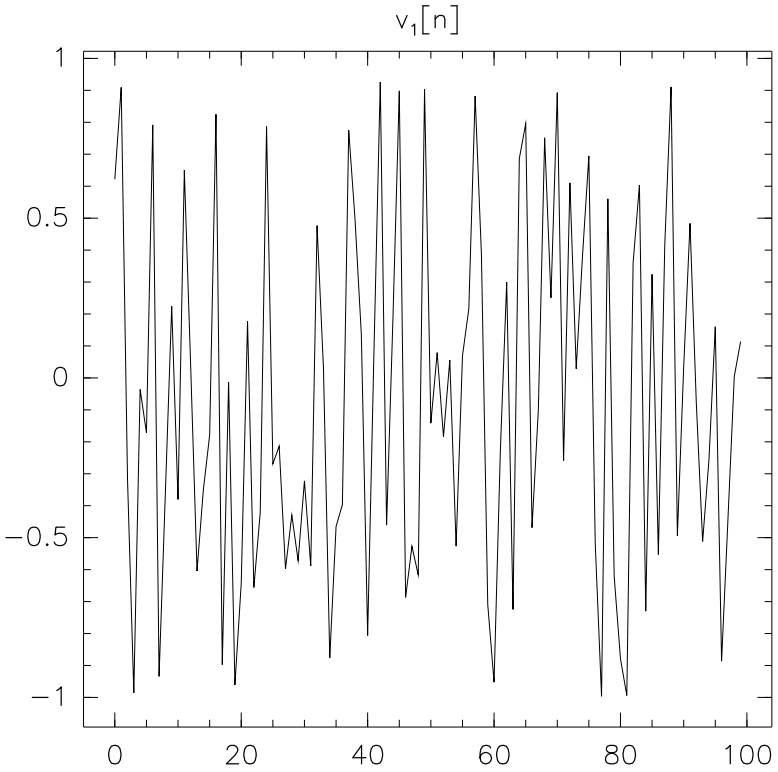
$$R_{vv}[m] = (\frac{1}{2} - \frac{1}{2})(\frac{1}{2} - \frac{1}{2}) = 0$$

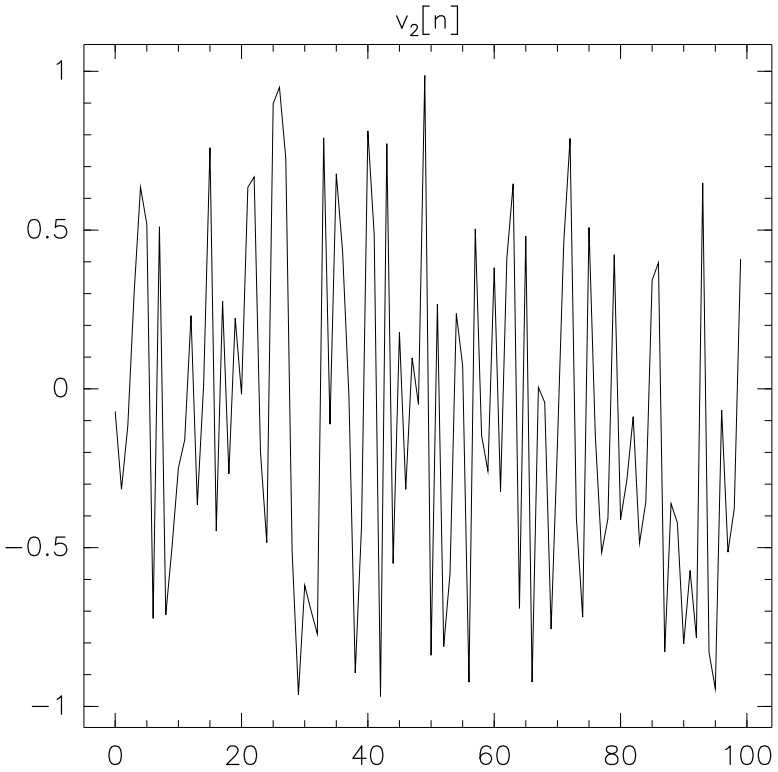
If $m = 0$,

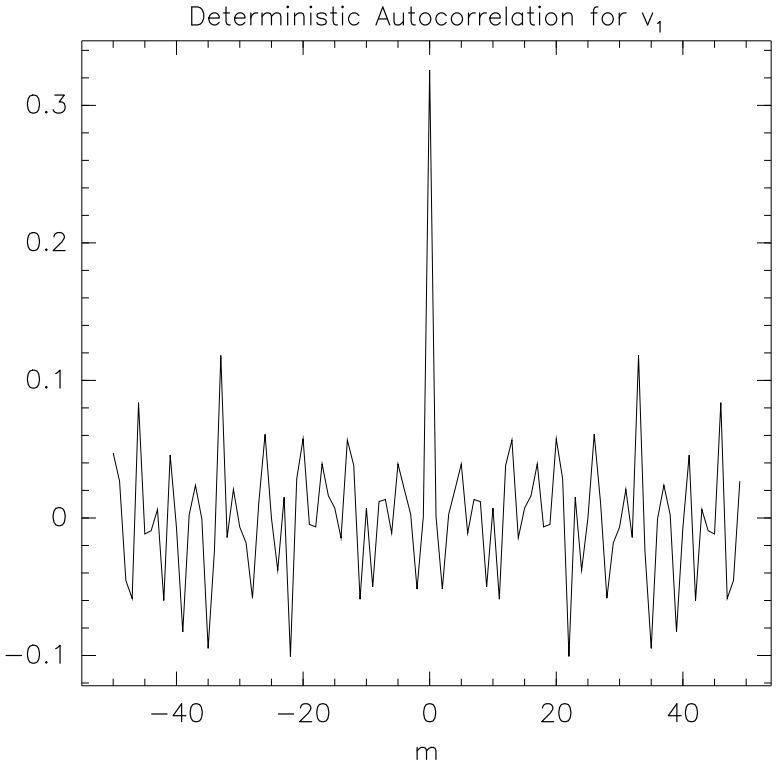
$$R_{vv}[m] = E[v^2[n]] = \frac{1}{2}1^2 + \frac{1}{2}(-1)^2 = 1$$

Thus, $R_{vv}[m] = \delta[m]$. Therefore, $S_{vv}(e^{j\omega}) = 1$.

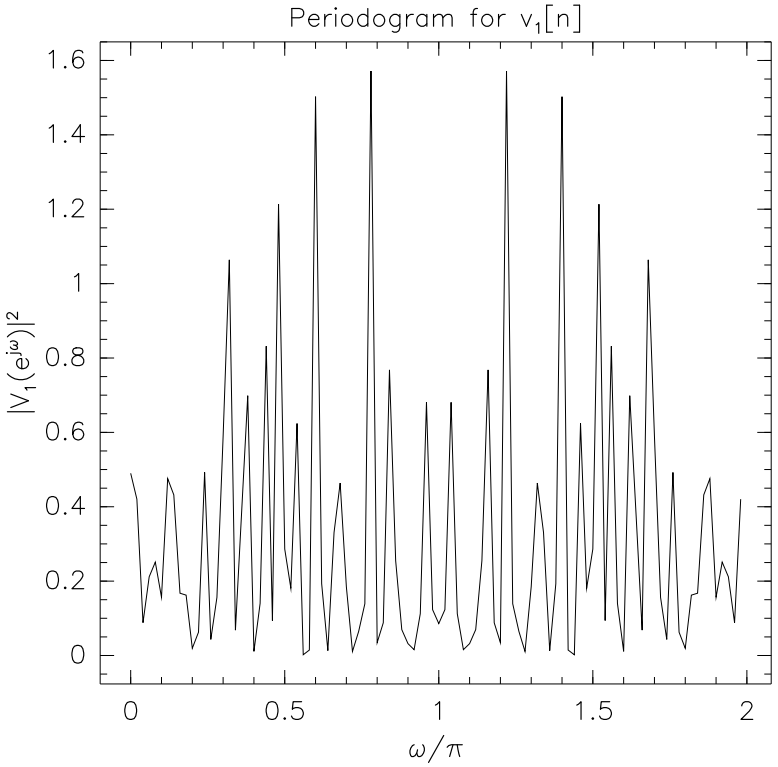
(b) For particular realizations v_1 and v_2 of the IID process V, I measured $\mu_1 = 0.06308$ and $\mu_2 = -0.03003$. These results do indicate that the process V is ergodic with respect to mean and autocorrelation since the means of each realization approach zero and the deterministic autocorrelation approaches a delta function.

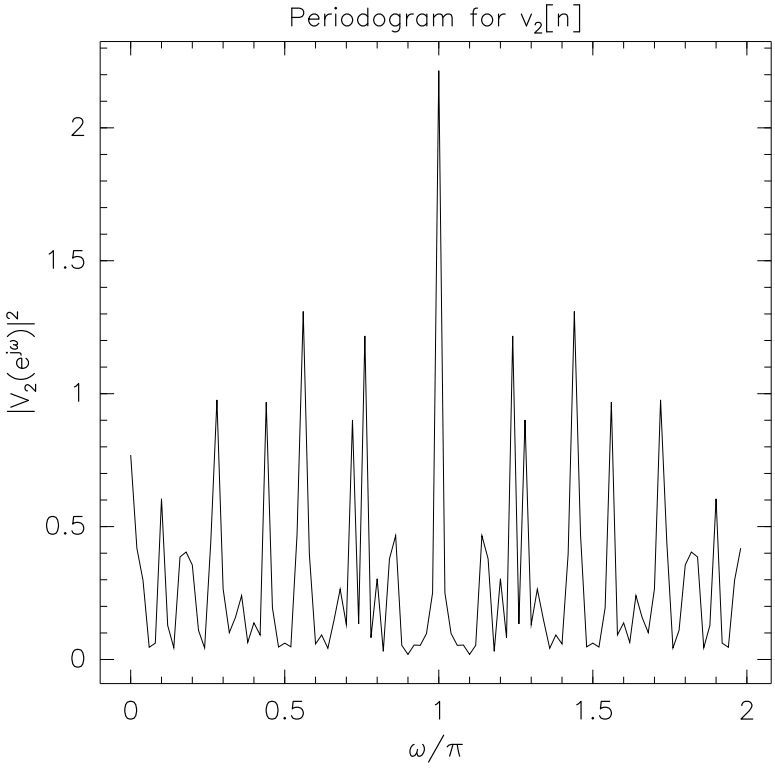

$v_1[n]$
1
0.5
0
−0.5
−1
0
20
40
60
80
100

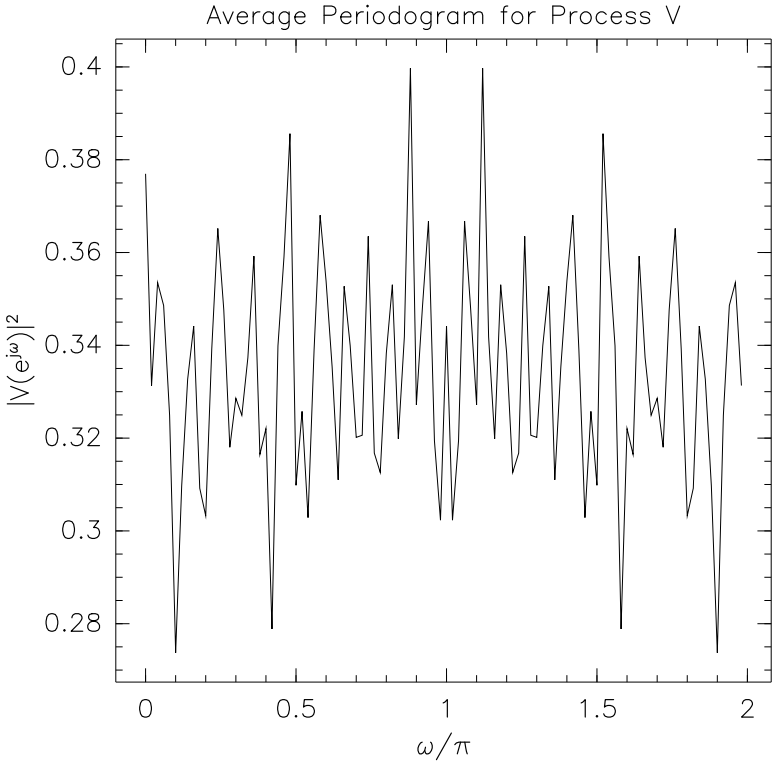

$v_2[n]$
1
0.5
0
−0.5
−1
0
20
40
60
80
100



(c) The periodograms for both realizations are symmetric about $\omega = 0$, but beyond that, they're not very similar. They both contain random peaks and valleys and have approximately the same order of magnitude.





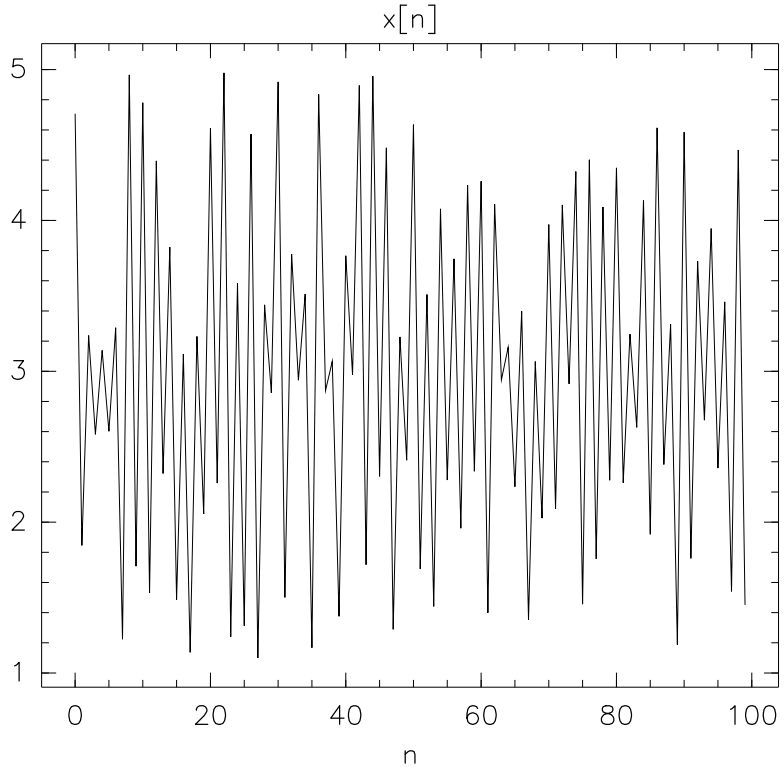


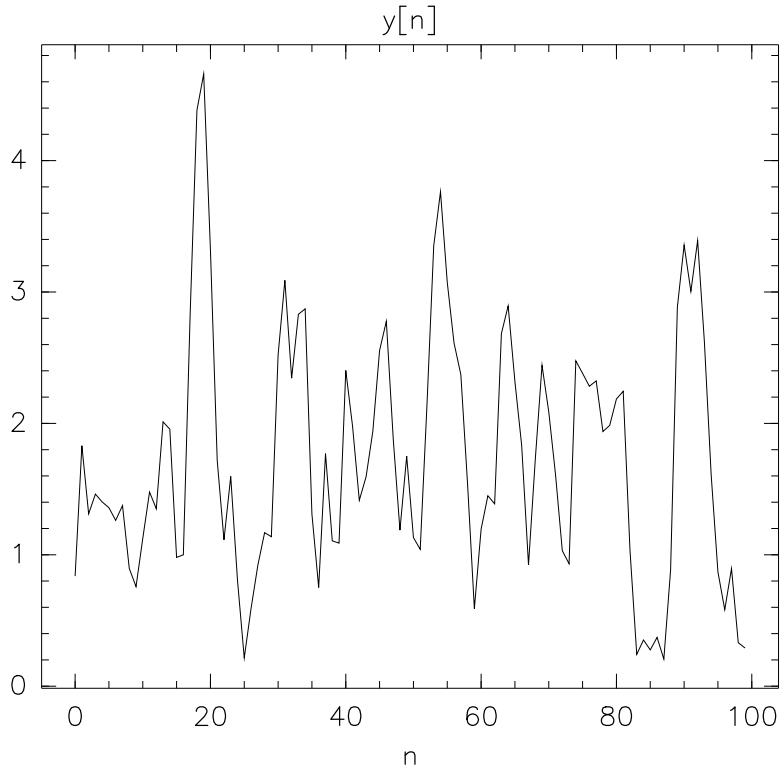
2.

$$\mu_x[n] = E[3] + E[(-1)^n] + E[v[n]] = 3 + (-1)^n + \mu_v = 3 + (-1)^n$$

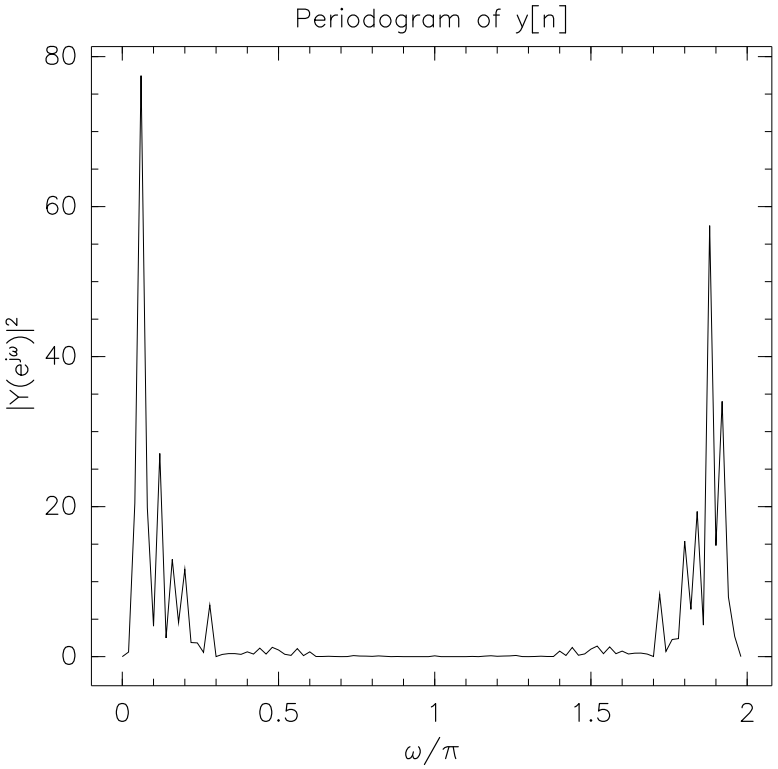
$$R_{xx}[n+m, n] = 9 + 3(-1)^n + 3(-1)^m + 3(-1)^{n+m} + R_{vv}[m]$$

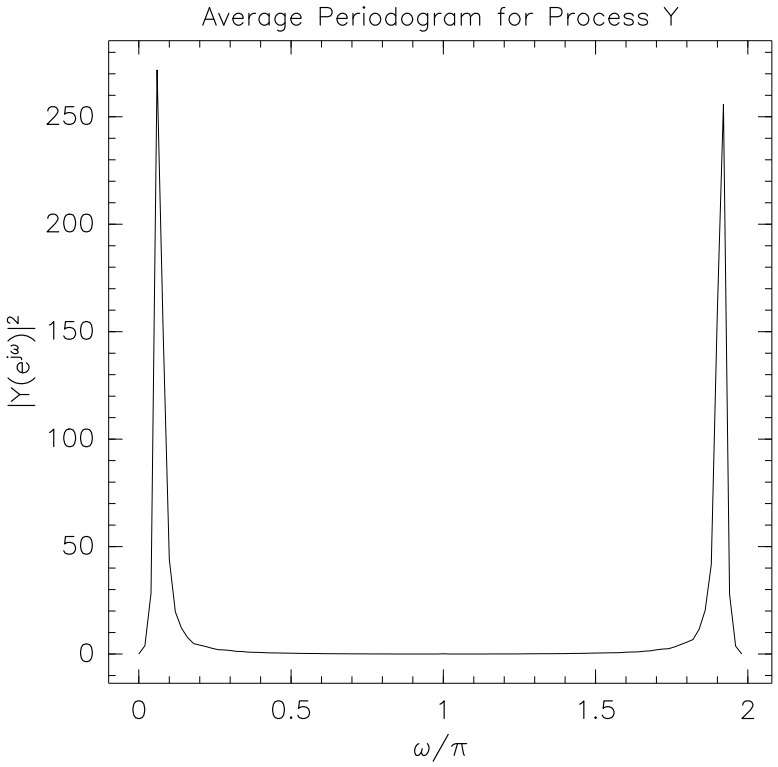
$$C_{xx}[n+m,n] = R_{vv}[m] = \delta[m]$$





$y[n]$ appears random but with a period of about 10 samples, which is consistent with the filter used to generate it. That filter looks similar to a cosine with period of about 10 cycles.





The averaged periodogram of $y[n]$ looks very similar to the theoretical power spectral density; it has two impulse like peaks as we would expect for $|H(e^{j\omega})|^2$ and is scaled by $\sigma_v^2 = R_{vv}[0] = 1$.

```
#!/usr/bin/env python
from biggles import Curve, FramedPlot
from Numeric import *
from RandomArray import uniform
from FFT import fft, inverse_fft

ARRAY_SIZE = 100

# part a.i.
def signal():
    return uniform(-1, 1,(ARRAY_SIZE,))

# part a.ii.
v1, v2 = signal(), signal()

def quickPlot(data, title = None, xlabel = None,
              ylabel = None, xdata = None):
    p = FramedPlot()
    if title:
        p.title = title
    if xlabel:
        p.xlabel = xlabel
    if ylabel:
        p.ylabel = ylabel
    if not(xdata):
        xdata =  arange(len(data))
    p.add(Curve(xdata, data))
    return p

quickPlot(v1, "$v_1[n]$").write_eps('v1.eps')
quickPlot(v2, "$v_2[n]$").write_eps('v2.eps')

print "Time averages for v1 and v2:", average(v1), average(v2)

def deterministicAutocorrelation(m, d1,d2):
    assert len(d1) == len(d2)
    if m < 0:
        return deterministicAutocorrelation(-1 * m, d2, d1)
    elif m == 0:
        x, y = d1, d2
```

```
    else:
        x = d1[m:]
        y = d2[:-1*m]
    assert len(x) == len(y)
    return  sum(x * y) / len(x)

all_m = range(-50, 50)
Rv1 = [deterministicAutocorrelation(m, v1, v1) for m in all_m]
quickPlot(Rv1, "Deterministic Autocorrelation for $v_1$",
          "m", xdata = all_m).write_eps("deterministicAutocorrelation.eps")

# a.iii.
def periodogram(data):
    return pow(absolute(fft(data)), 2) / ARRAY_SIZE

omega = arange(0, 2, 1/50.)
quickPlot(periodogram(v1), "Periodogram for $v_1[n]$", "$\omega/\pi$",
          "$|V_1(e^{j\omega})|^2$", omega).write_eps('periodogram_v1.eps')
quickPlot(periodogram(v2), "Periodogram for $v_2[n]$", "$\omega/\pi$",
          "$|V_2(e^{j\omega})|^2$", omega).write_eps('periodogram_v2.eps')

avgPeriodogram = average(pow(absolute(fft(uniform(-1,1, (200, ARRAY_SIZE)))),
                             2)) / ARRAY_SIZE
quickPlot(avgPeriodogram, "Average Periodogram for Process V$",
          "$\omega/\pi$", "$|V(e^{j\omega})|^2$",
          omega).write_eps('average_periodogram.eps')


# b

omega_half = arange(0,pi,pi/50)
omega_full = concatenate((omega_half, -1*omega_half[::-1]))
H = ((exp(2*1j*omega_full) - 1) /
     (exp(2*1j*omega_full) - 2*0.95*exp(1j*omega_full) + 0.95*0.95 + 1/25.))

x = 3 + pow(-1, arange(ARRAY_SIZE)) + signal()
y = absolute(inverse_fft(fft(x) * H))

quickPlot(x, "$x[n]$", "n").write_eps("x.eps")
quickPlot(y, "$y[n]$", "n").write_eps("y.eps")
quickPlot(pow(absolute(fft(x)*H),2)/100 , "Periodogram of $y[n]$",
          "$\omega/\pi$",
          "$|Y(e^{j\omega})|^2$", omega).write_eps("y_periodogram.eps")
```

```
avgPeriodogram = average(pow(
    absolute(H * fft(3 + pow(-1, arange(ARRAY_SIZE))
                     + uniform(-1,1,(200,100)))), 2)) / 100
quickPlot(avgPeriodogram, "Average Periodogram for Process Y$",
          "$\omega/\pi$", "$|Y(e^{j\omega})|^2$",
          omega).write_eps('y_average_periodogram.eps')
```