ELECTRONOTES
1016 Hanshaw Road
Ithaca, NY 14850

APPLICATION NOTE NO. 378

March 2012

A PROGRAM FOR A FLAT WHITE NOISE

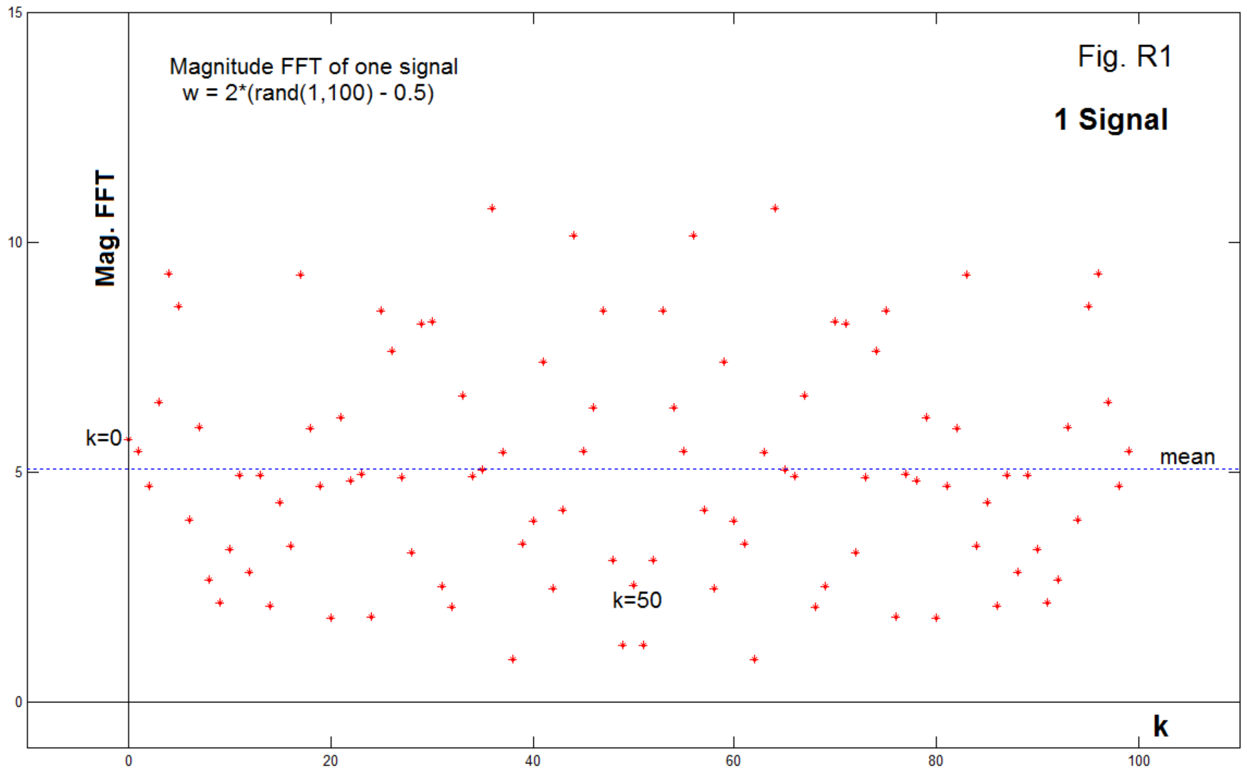
As noted in a recent Electronotes issue (EN#208):

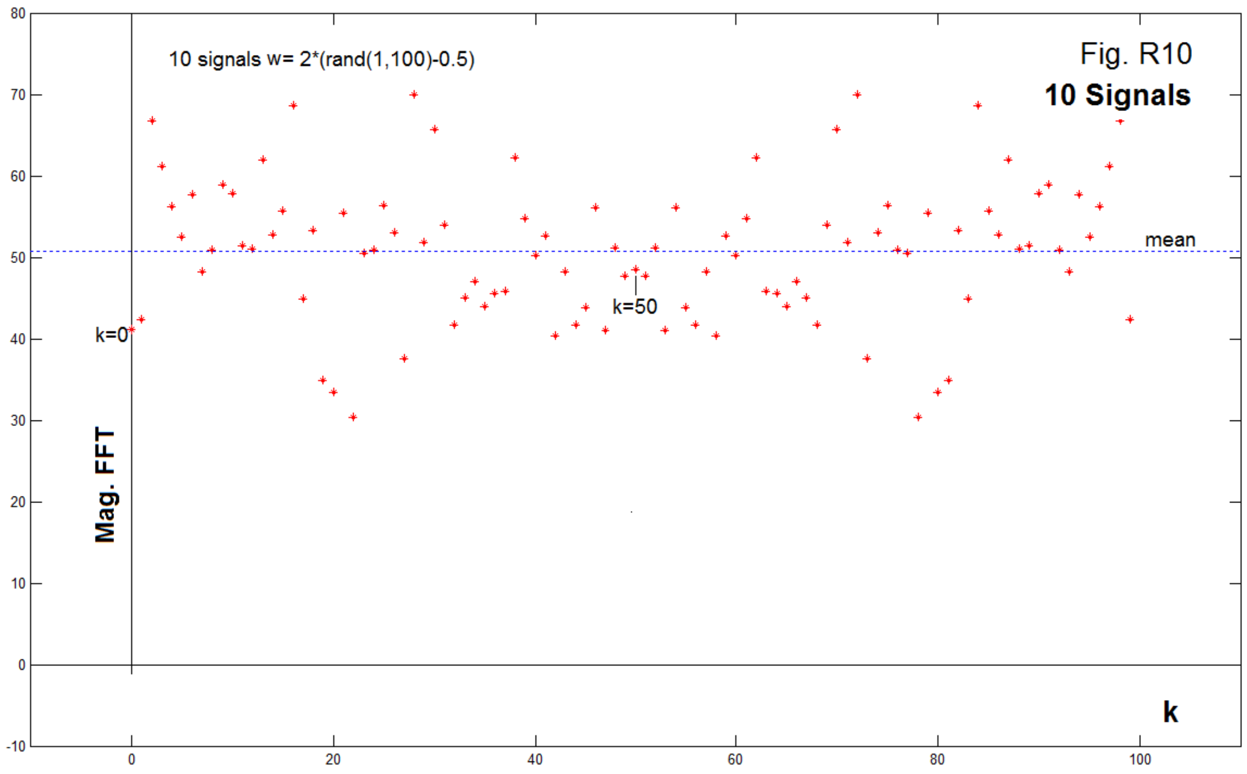
http://electronotes.netfirms.com/EN208.pdf

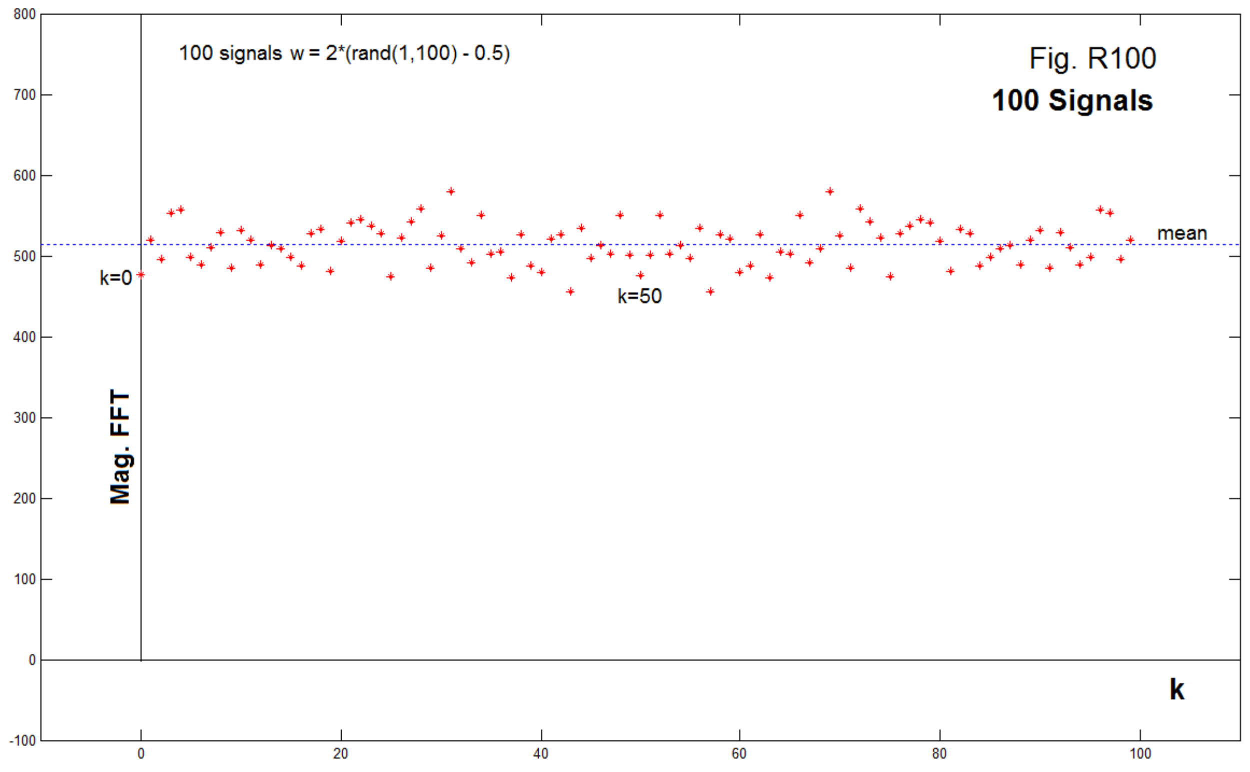
what would seem to be an obvious procedure to demonstrate the "flat spectrum" of a random sequence contained a surprise. The original idea was that, first of all, we do not expect any single FFT (magnitude) of a white noise sequence to be flat. We would however expect an average of magnitudes of many FFT's to be flatter and flatter as more examples are dawn from the ensemble. Our finding was that this is almost true. For a length N random sequence, with the exception of two frequencies, 0 (dc) and $f_s/2$ (half the sampling frequency), the latter in the case of an even length sequence. These two cases show a dip below flat that amounted to $(2)^{3/2}/\pi = 0.9003163$. That is, the FFT magnitudes for k=0 and for k=N/2 (for even N) are only 90% of those for all other frequencies. The EN#208 offered an explanation of the dip in terms of a Rayleigh vs. a "folded normal" distribution, which will not be repeated here. Further in EN#208, we showed that the average spectrum can be leveled by adding appropriate correcting components at k=0 and at k=N/2 (N even). The corrections were found experimentally, and no "derivation" was offered. Here we want to refine this correction and offer a Matlab function, ***randf***, that is flat.

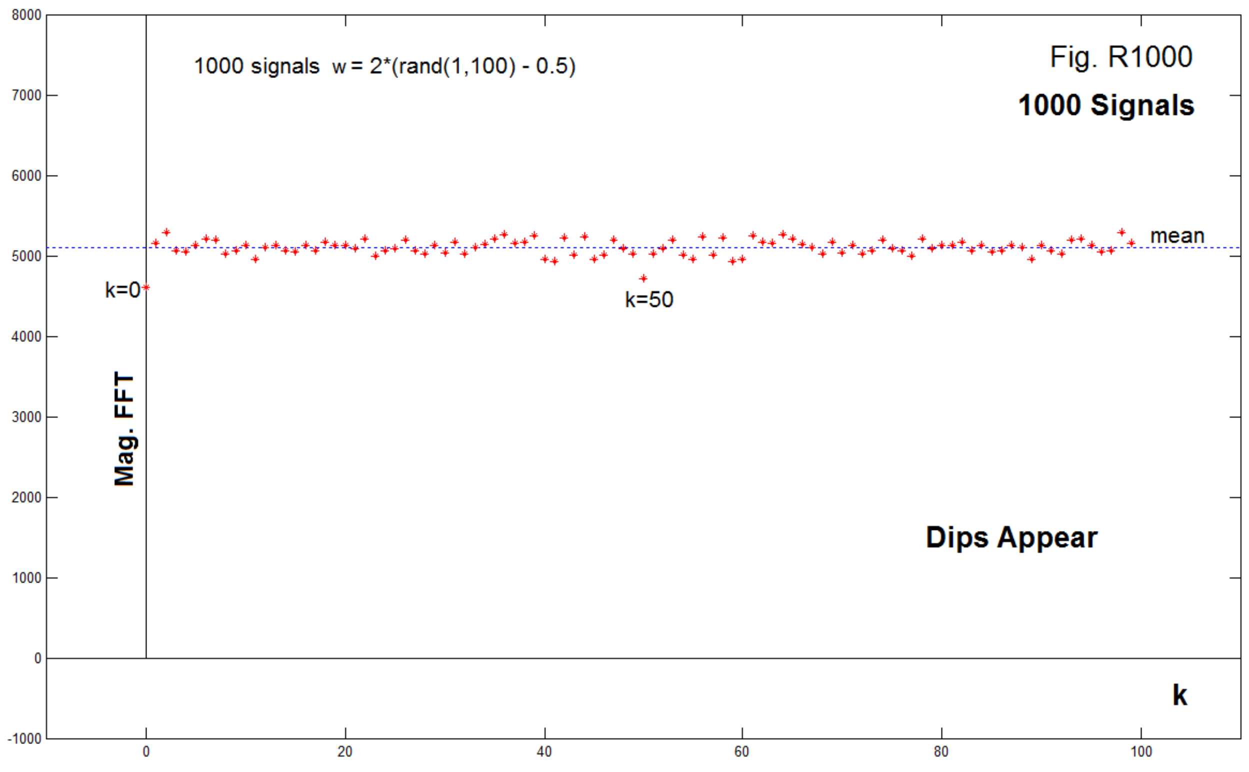
To review, the finding was that, as we anticipate, the FFT magnitude of any one random sequence is not in any way seen to be obviously flat (Fig. R1). By the time we average 10 FFT's, we see some flattening (Fig. R10) and this continues through additional Factors of 10 (Fig. R100, Fig. R1000, Fig. R10000, Fig. R100000) and with a million in the average (Fig. R1000000) we see a flat result with the two dips to 90% as mentioned. This is the result previously reported in EN#208.

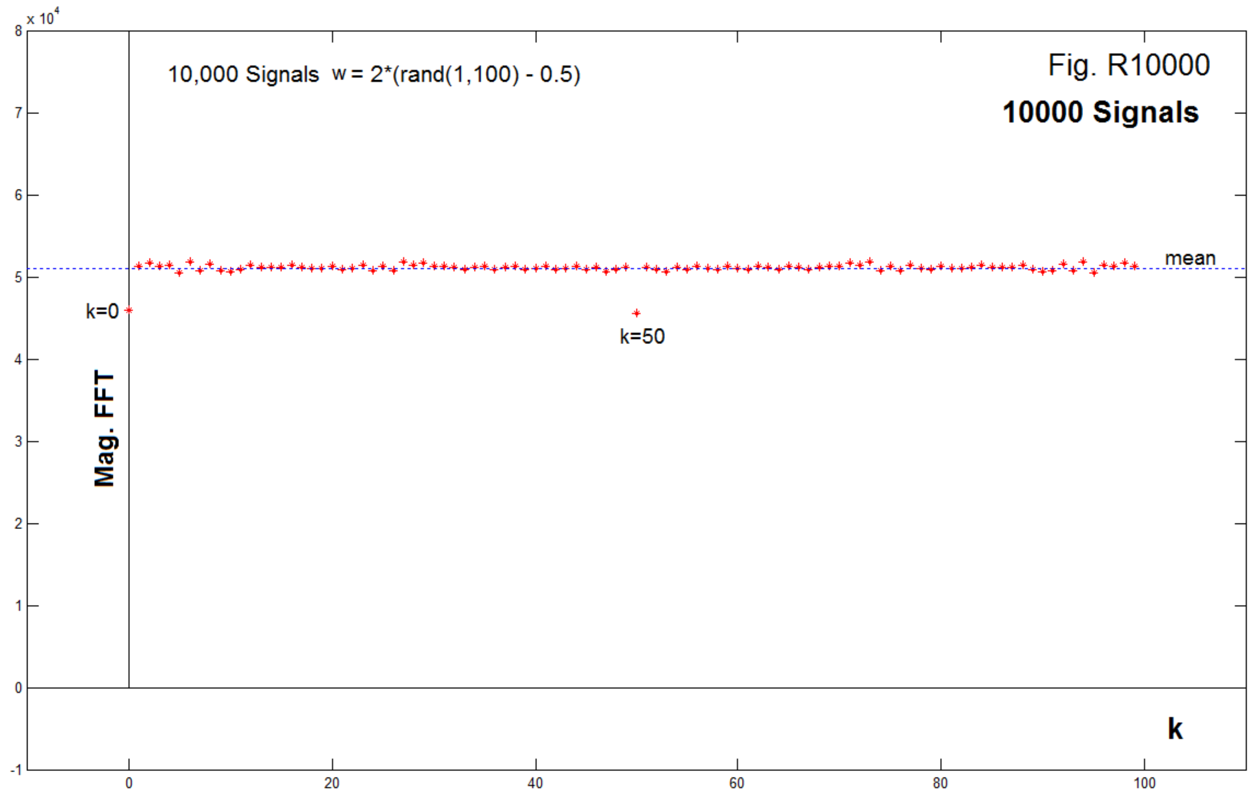
In addition to finding that the dips at k=0 and k=N/2 (N even) are very real, and finding an theoretical explanation, a proposed solution to the problem was offered. In as much as there were one or two spectral points that were low, it made sense to try to add spectral energy at these two frequencies. That is, we wanted to add some dc bias for k=0 and an alternating sequence (N even).

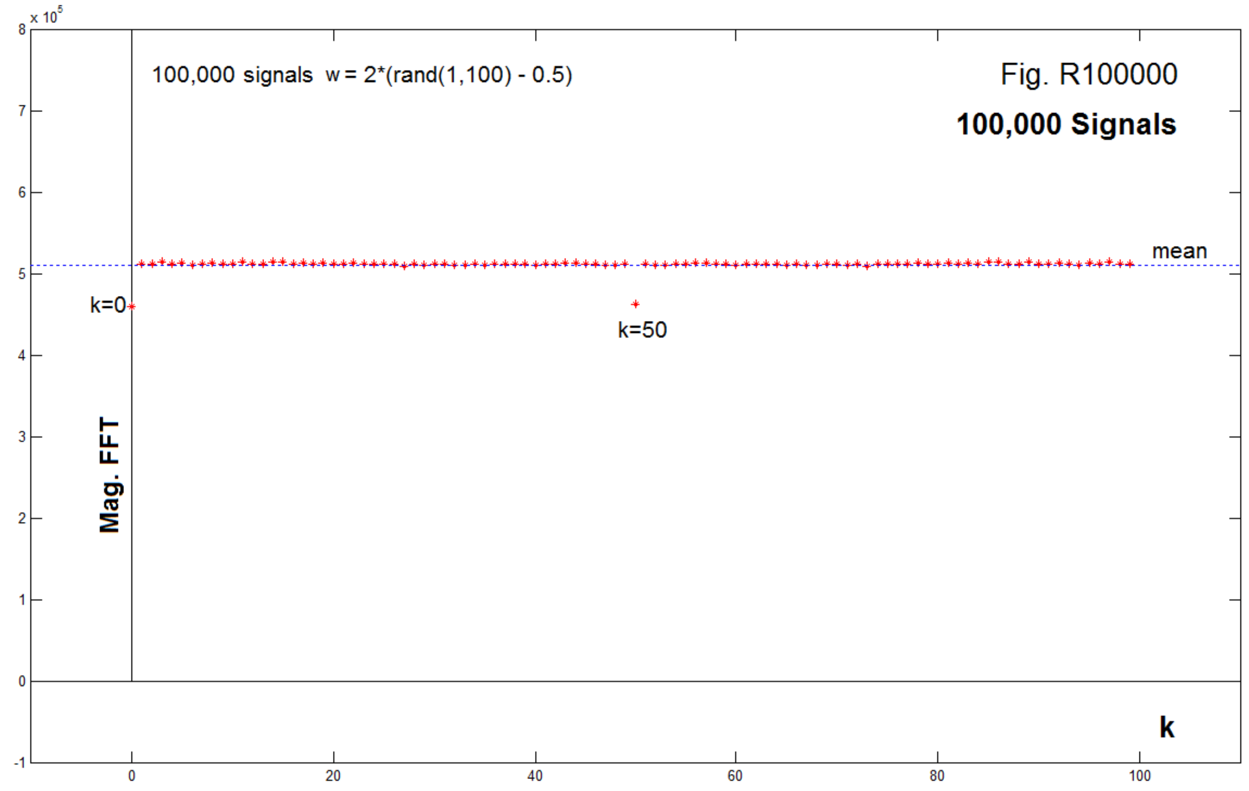

Magnitude FFT of one signal
w = 2*(rand(1,100) - 0.5)
Fig. R1
1 Signal
Mag. FFT
k=0
mean
k=50
k
15
10
5
0
0
20
40
60
80
100

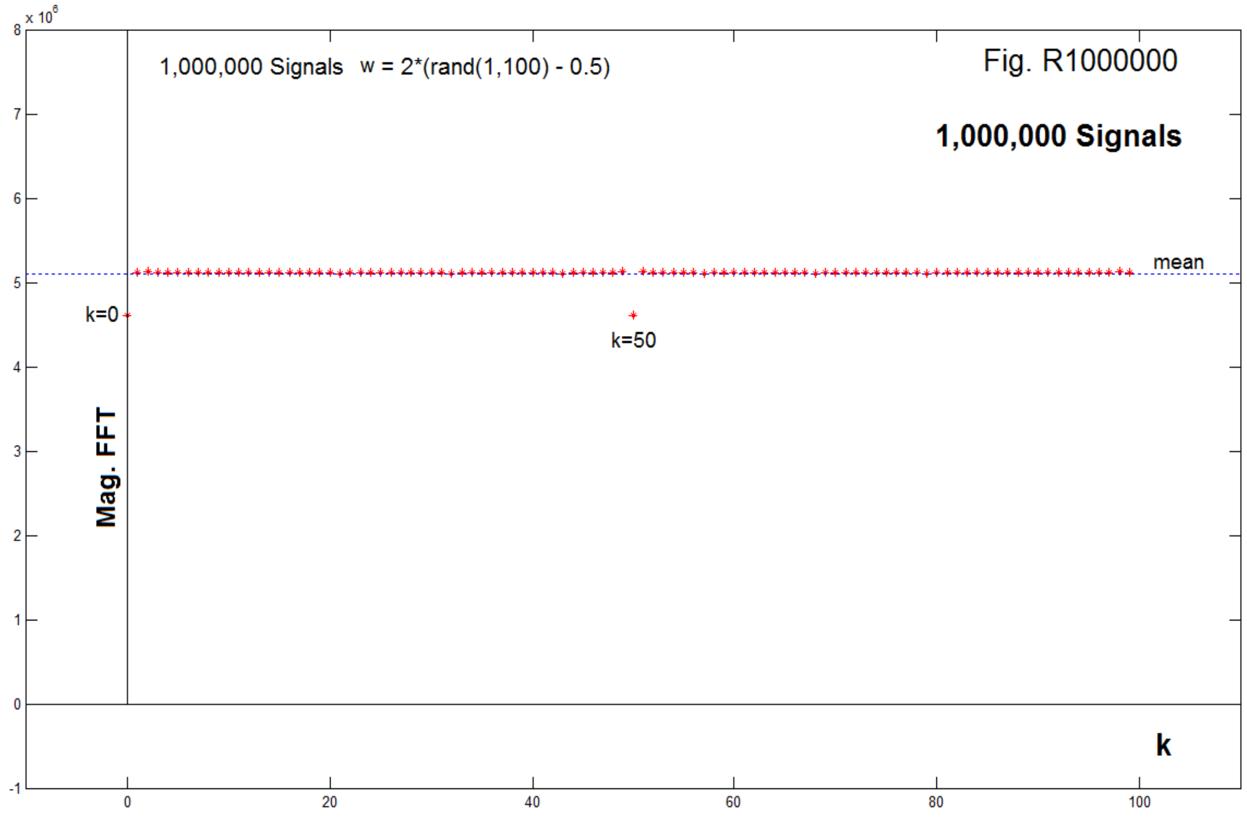

10 signals w= 2*(rand(1,100)-0.5)
Fig. R10
10 Signals
Mag. FFT
k=0
mean
k=50
k
80
70
60
50
40
30
20
10
0
-10
0
20
40
60
80
100


100 signals w = 2*(rand(1,100) - 0.5)
Fig. R100
100 Signals
mean
k=0
k=50
Mag. FFT
k
800
700
600
500
400
300
200
100
0
-100
0
20
40
60
80
100


1000 signals w = 2*(rand(1,100) - 0.5)
Fig. R1000
1000 Signals
mean
k=0
k=50
Mag. FFT
Dips Appear
k
8000
7000
6000
5000
4000
3000
2000
1000
0
-1000
0
20
40
60
80
100


x 10^4
10,000 Signals w = 2*(rand(1,100) - 0.5)
Fig. R10000
10000 Signals
mean
k=0
k=50
Mag. FFT
k
8
7
6
5
4
3
2
1
0
-1
0
20
40
60
80
100


x 10^5
100,000 signals w = 2*(rand(1,100) - 0.5)
Fig. R100000
100,000 Signals
mean
k=0
k=50
Mag. FFT
k
8
7
6
5
4
3
2
1
0
-1
0
20
40
60
80
100



The figures above were obtained by taking random sequences by the familiar **UNMODIFIED** Matlab command line:

```
w = 2*(rand(1,100) – 0.5)
```

In EN#208 we modified the random signal generator by adding a constant and an alternating series

```
n=0:99
sq = (-1).^n
w = 2*(rand(1,100) – 0.5 – 0.0139) + 2*0.0139*sq
```

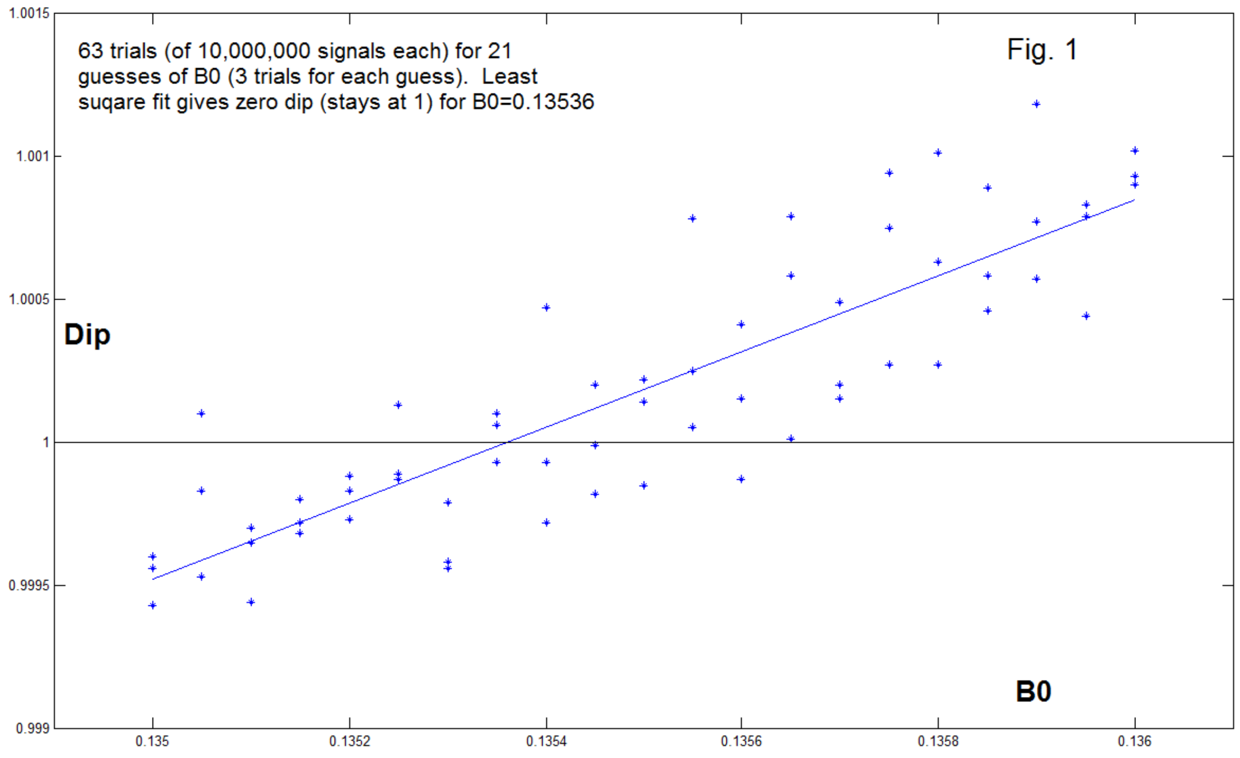
This seemed to work. However it was not clear where the parameter 0.0139 came from – except by trial and error. We further determined that the parameter was 0.139/√N where N was the length of the sequence (N = 100 here). In particular, no simple relationship to more fundamental constants was found, it did not seem to be anything we could calculate, and error limits on what it might be experimentally were not established. Here we will look at this a bit more. The approach remains experimental.

For this note, we decided to take a closer look at what our one free parameter really is. This was a matter of running a lot more random trials. This was done with a program where the random sequences to be tested were calculated as:

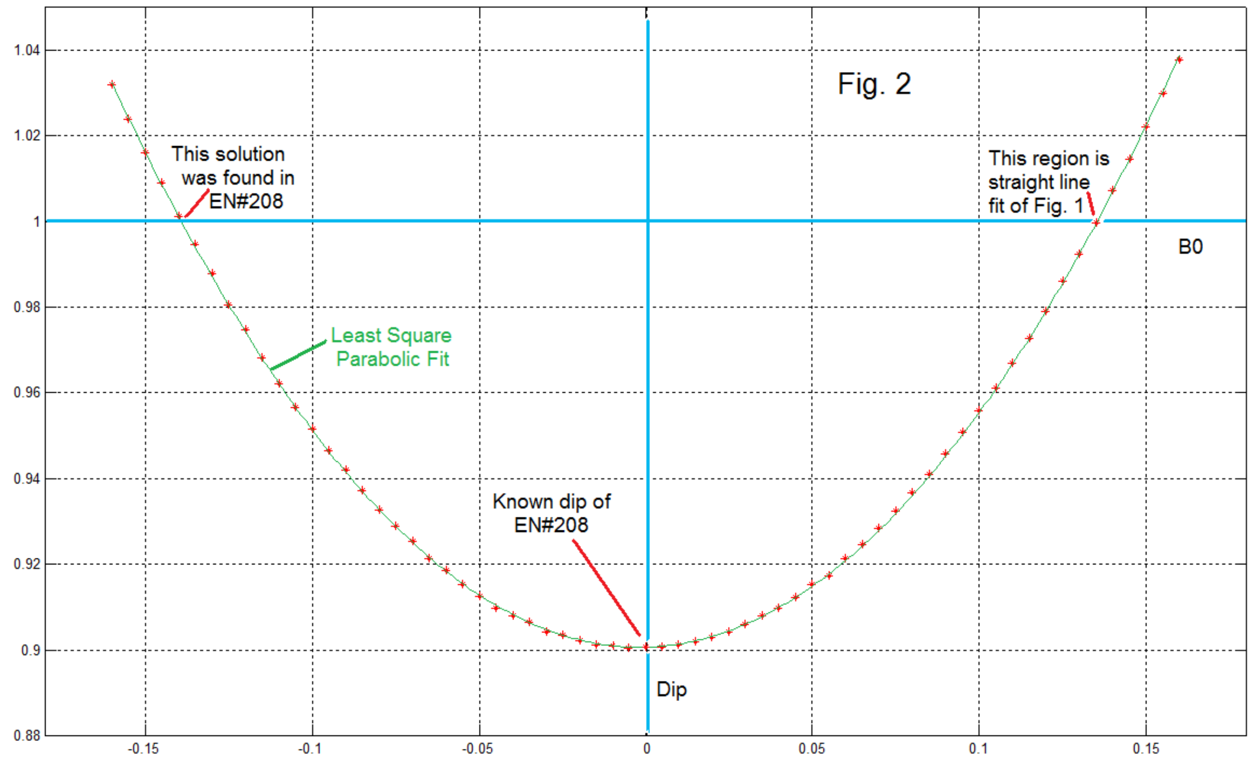
B = B0/sqrt(N)

w = 2*(rand(1,N) – 0.5 + B) +even*2*B*sq

In this case, we expected the cancellation of the dips to occur for a value of B0 = -0.139 (or thereabout) based on EN#208. However, through a failure to pay careful attention (!), positive value around 0.139 were tested. With a large number (63, each of 10,000,000 random signals) of trials, a least-square linear fit was found (Fig. 1) and from that, we found B0 to be 0.13536. Each of the data points represents about 10,000,000 averaged trials. Total computer time about ½ a day. The additional two decimal places were perhaps reasonably justified. The surprise was that when we looked back to EN#208, that number was negative – this one was positive, and they were not the exact same magnitude. So switching the sign was a fortunate error. There was more going on here.



Further, we now had evidence that the shape of the curve of the Dip, as a function of the parameter B0, is probably something more like a parabola. We know that the Dip is close to 1 at the two values -0.139 and + 0.135, but we also know that when B0=0 the dip is about 0.9. Thus encouraged, we can calculate a wider range of B0, not just where we approach a Dip of 1 (as in Fig. 1) and these 65 trials are plotted (red stars) in



Negative B0	Positive B0
-0.139 Fig. 5 of EN#208	0.13536 Fig. 1 Here
-0.13922 Fig. 2 Here	0.13535 Fig. 2 Here

Fig. 2, along with a least squared fit parabola (green line). Again, each point is 10,000,000 trials, and the total computation is about ½ day. From Fig. 2, we get both the positive and the negative solutions. The parabola, if it is a parabola, is slightly skewed to the left as shown. A summary of the numerical values is shown in the table just below Fig. 2.

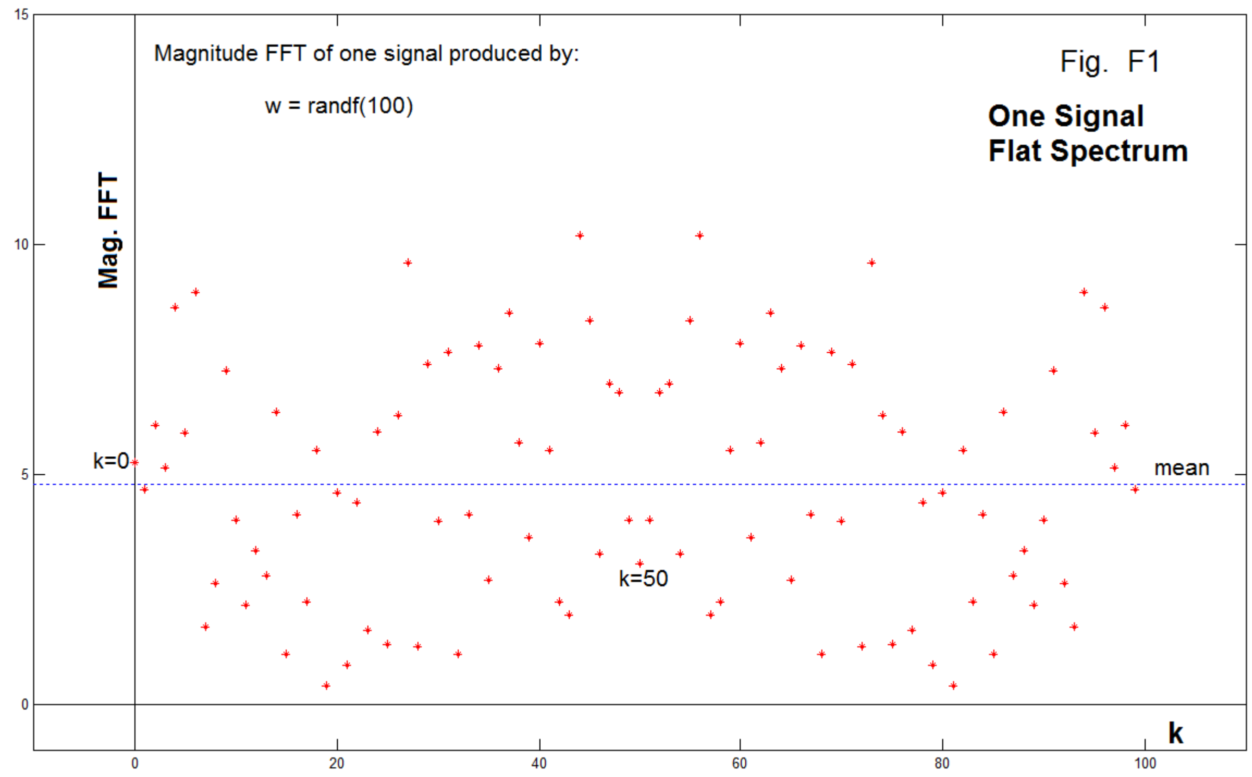
This is what we know. We know enough to write a new random function. There is much more we don't know.

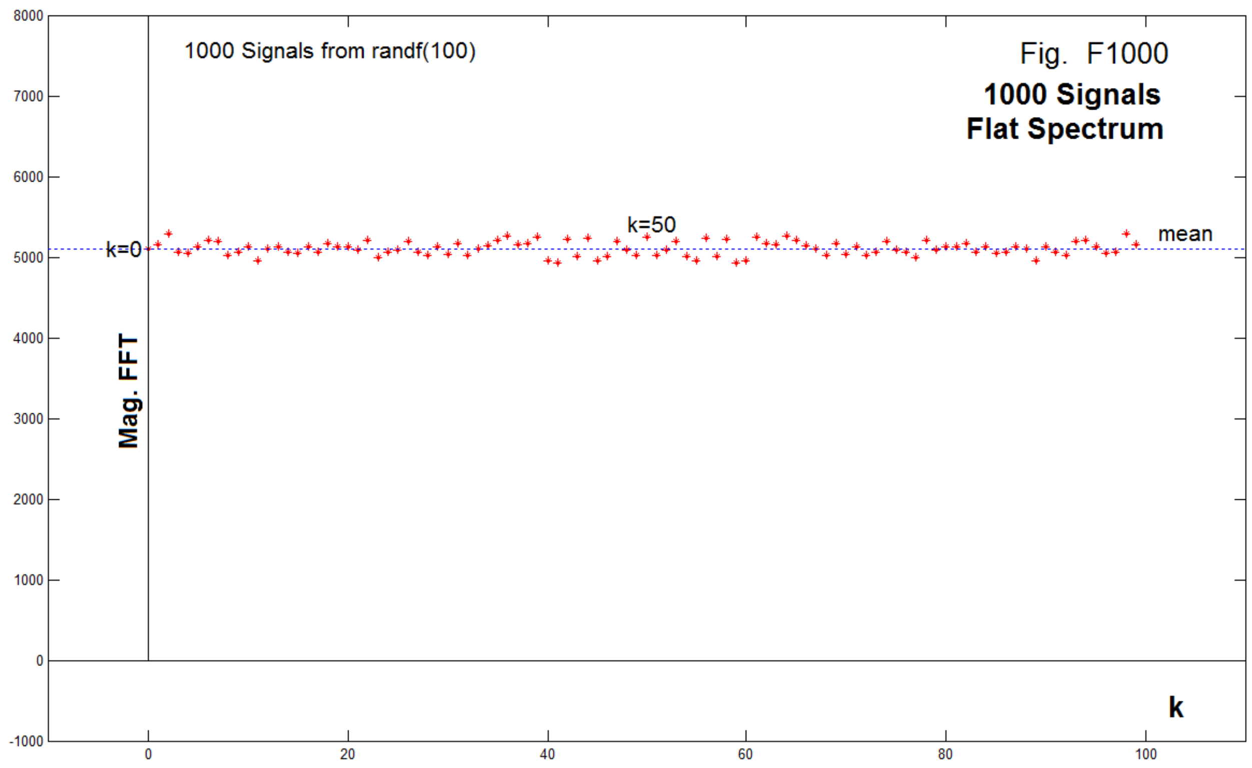
Here is some code:

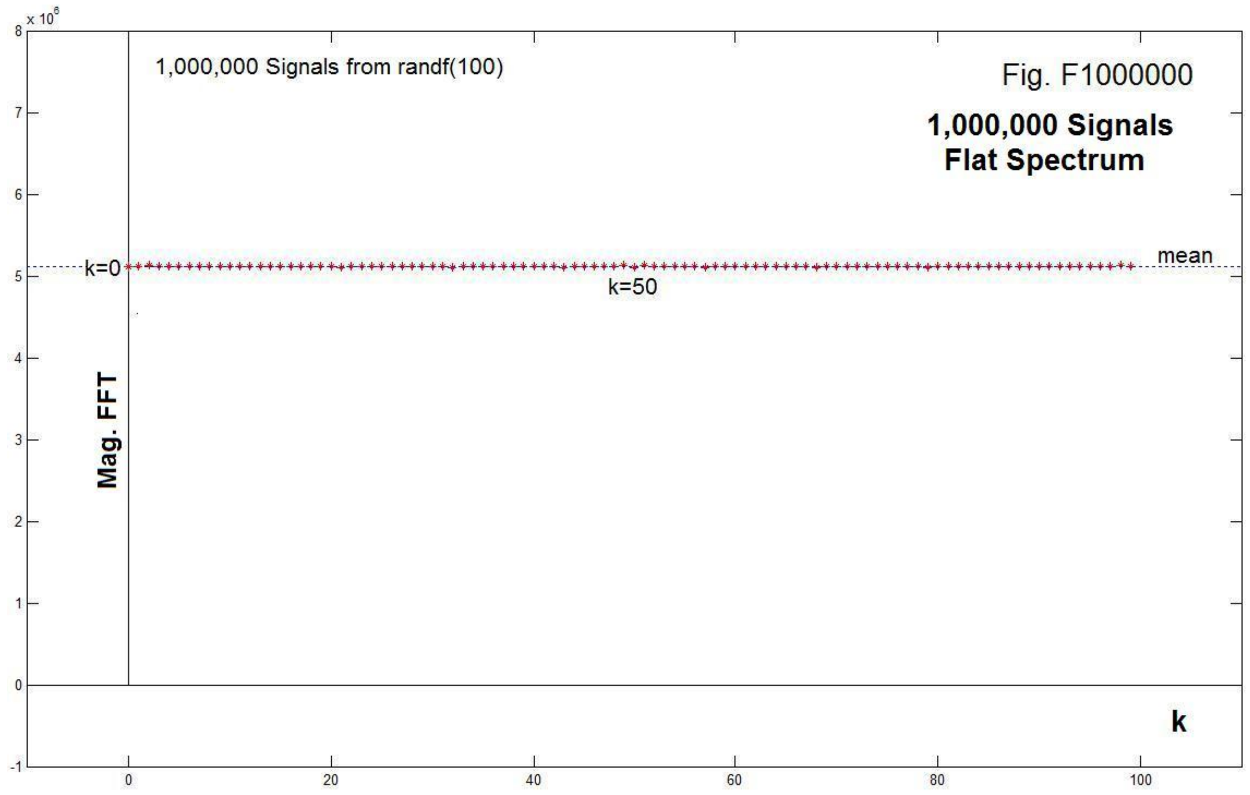
```
% randf
% produces a length N random signal with a uniform distribution
% close to -1 and +1 with DC and AC "bias" such that the DFT
% spectrum is not reduced by 90% at k=0 and k=N/2 (N even).
function x=randf(N)

B=0.13536/sqrt(N);
even=0;
sq=(-1).^(0:N-1);
if N/2==floor(N/2)
   even=1;
end
      x=2*(rand(1,N) -0.5 + B) + even*2*B*sq;
```

This function ***randf*** is not a plug-in replacement for ***rand***. It is defined to replace the usual 2*(rand(1,N) – ½). Putting this into the program that produced the "R" figures results in an analogous sequence, of which we only want to print a few here.

Fig. F1 is very much like Fig. R1 in that neither shows any degree of flattening or anything special at k=0 and k=50. Looking at Fig. R1000 (less so Fig. R100), we see that the dips at k=0 and k=50 are appearing. In comparison, we see nothing special happening in Fig. F1000. Finally, we compare Fig. F1000000 to Fig. R1000000, and we see what is essentially absolutely flat in Fig. F1000000 while the dips at k=0 and k=50 were well established in Fig. R1000000. To the accuracy of plotting in these figures, somewhat fewer that the five digits used in the programs would likely be sufficient.


Magnitude FFT of one signal produced by:
w = randf(100)
Fig. F1
One Signal
Flat Spectrum
Mag. FFT
k=0
mean
k=50
k
15
10
5
0
0
20
40
60
80
100


1000 Signals from randf(100)
Fig. F1000
1000 Signals
Flat Spectrum
k=50
mean
k=0
Mag. FFT
k
8000
7000
6000
5000
4000
3000
2000
1000
0
-1000
0
20
40
60
80
100



SUCCESS ? Some Answers, More Questions.

In as much as we set out to achieve a program for a flat white noise, we have succeeded. But we kind of had this practical tool at the end of the EN#208 presentation. Here we have made a few steps toward a better understanding of what is going on – but have just as certainly found new questions. The fact that the parameter we need to flatten the spectrum is found to be double valued, is interesting. But we don't know how to calculate it (them) from more fundamental considerations. This may be a parabolic object as in Fig. 2 (despite appearing skewed – which may be a numerical consequence), but we neither know how to derive an equation for the parabola, or why it should come out that way. Lots of projects like this!