ELECTRONOTES
1016 Hanshaw Road
Ithaca, NY 14850

APPLICATION NOTE NO. 412

May 25, 2014

MORE FUN WITH RED NOISE

INTRODUCTION

In AN-384 ("Fun with Red Noise") [1] we discussed issues with so-called "Red Noise" signals (also called brown noise, random walk, or "drunkard's walk") that are the result of integrating white noise signals. So there is a unique relationship between any instance of white noise and a corresponding instance of red noise, with some obvious refinements relating to the "constant of integration" familiar since Calculus 101.

Red noise really is more "fun" than white. The reason I think is that we can without much difficulty show a "typical example of white noise". Anyone who has had occasion to write or discuss white noise probably realizes that when you work with what you call a random signal, you really should not select from several or many to illustrate a "typical" signal. You use the first one you get, or perhaps decide ahead of time to use the 17^{th} example, and when you get it, that's it. I guess you could plead special exceptions. For example, when you display a run of coin flips, what if you got a length 25 sequence of 24 heads and a terminating isolated tail at the end? You could spend a few paragraphs apologizing or just make a new run. We don't expect a run of 24 heads to occur. In fact, and this is the point, we have almost no problem obtaining an honest typical white noise.

The same is NOT true of red noise. We can honestly calculate and plot one, but we may fear that the reader will get the wrong idea unless an array of possibilities is presented, and this we saw in AN-384 and will continue to see below.

There are two additional points to be made here. (1) While it is possible to simply integrate white noise (AN-384 as partly redone below) it is also possible to calculate the FFT of a white signal, shape it to fall off as 1/frequency, and take the inverse FFT of the result. (2) It is the case that while red noises wonder about with many shapes, more than one might at first expect will have their extreme values relatively close to the ends. I think this is easily understood – but we need to observe that it is true.

INTEGRATED WHITE NOISE

Recalling here from AN-384, we can generate red noise easily, digitally, by integrating a white noise sequence (a random number generator). All we need to do in add up the sequence. Fig. 1 shows the absurdly simple digital filter integrator that we will use here. So it is very similar to adding up heads and tails, except here there is no requirement that the white noise is binary (heads/tails being binary). We generally would use a uniform or Gaussian distribution for the input. So a white noise algorithm and a few lines of code do what we need.

Here is the Matlab code: PROGRAM 1

```
% Red Noise Test   redtest1.m
% m signals
S=zeros(1,501);    % correction to AN-384 (does not change results of AN-384)
for m=1:10000      % more signals than AN-384

  sw=2*(rand(1,1020)-0.5);  % white, uniform

   % red filter - pole at z=+1, (integrator)
     sr(1)=sw(1);
     for k=2:1000
       sr(k)=sr(k-1)+sw(k);
     end

    %  FFT analysis and sum
       SR=abs(fft(sr));
       SR=SR(1:501);
       S=S+SR;
end

figure(1)
plot([0:500],S)
figure(2)
loglog([0:500],S)
grid
axis('square')
figure(2)
```

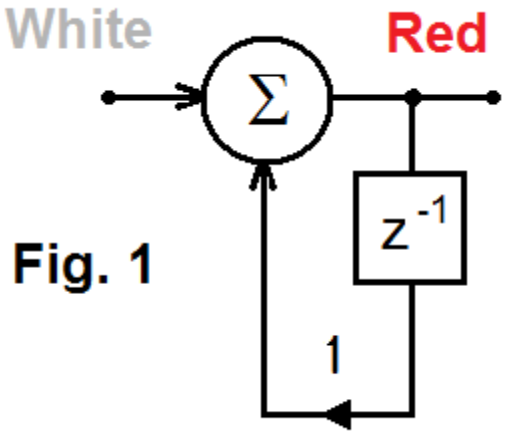
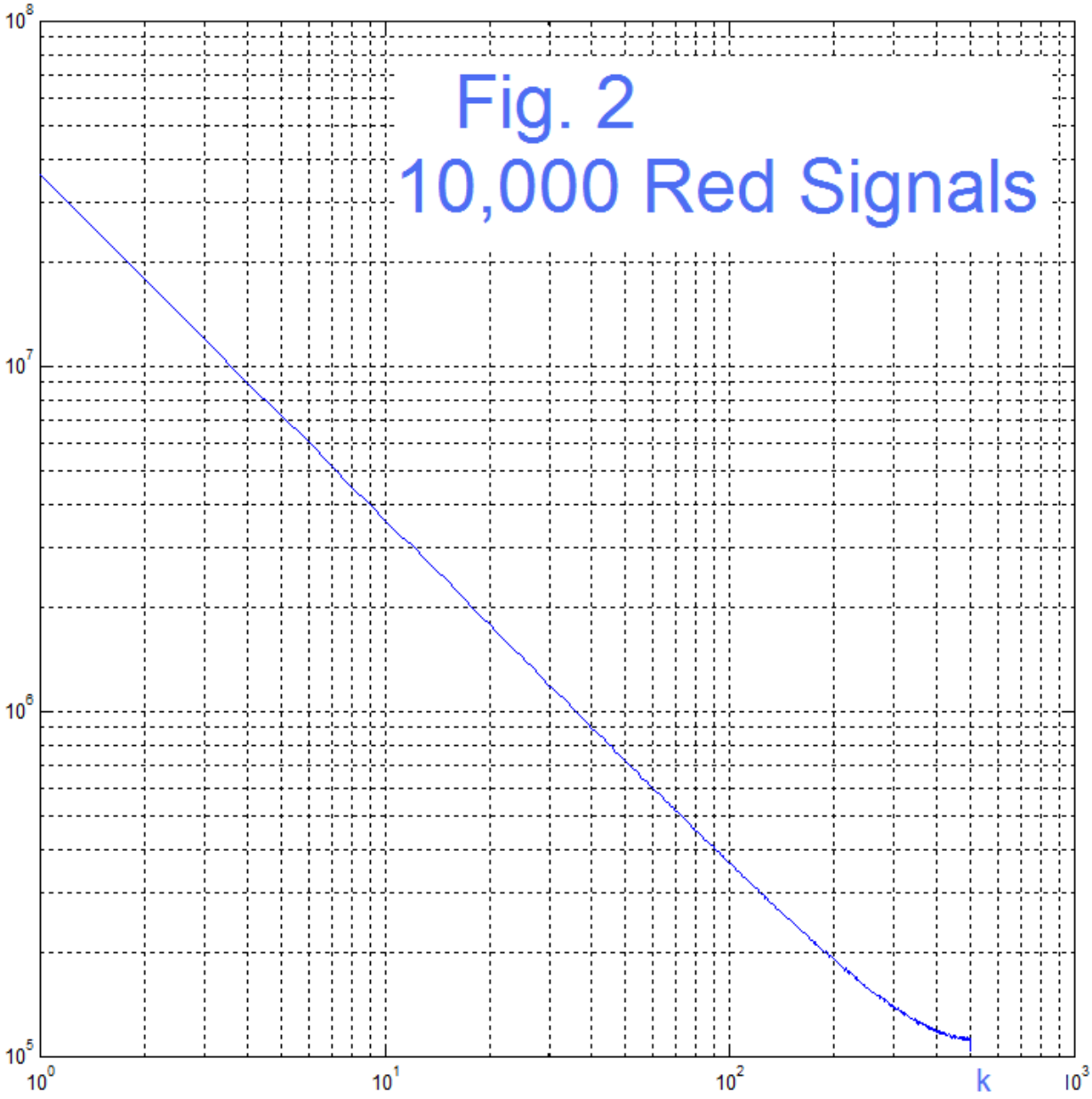


Fig. 1

The Matlab code generates 10,000 red signals of length 1000, calculates the FFT's of each signal, computes the magnitudes, and sums them. The result (Fig. 2) shows that the spectra of the red signals average to 1/f, the roll-off we expect from the integrator. This is a 45° angle on the log-log plot. The same spectral shape, in the form of a frequency response of the discrete integrator is available (Fig. 3) by the Matlab lines:

```
H =0.5*abs(freqz(1, [ 1 -1],500))       % arbitrary scaling
loglog([0:0.001:0.499,H)
grid
axis('square')
```



Thus it is easy enough to generate a red noise signal – but quite difficult to select one example and call it “typical”. For the most part, we will generate groups of 10 signals, each as an unselected (as is) sequence, as a means of illustrating a variety. Nonetheless, the signal in Fig. 4 is offered to show one red signal. We indicate on the figure that it is very long, 160,000 samples, far far more than we can expect to resolve in a plot. Indeed, had we chosen just the first 40,000 samples, we would have a far different impression of what a red signal might look like, assuming it had a preference for a certain (positive) polarity. It is probably a surprise to see that it comes back down, going below zero by nearly the same amount it went positive, then up again, and so on. Indeed, quite a show. We should keep in mind the ever-increasing amplitudes of lower frequencies as an explanation.

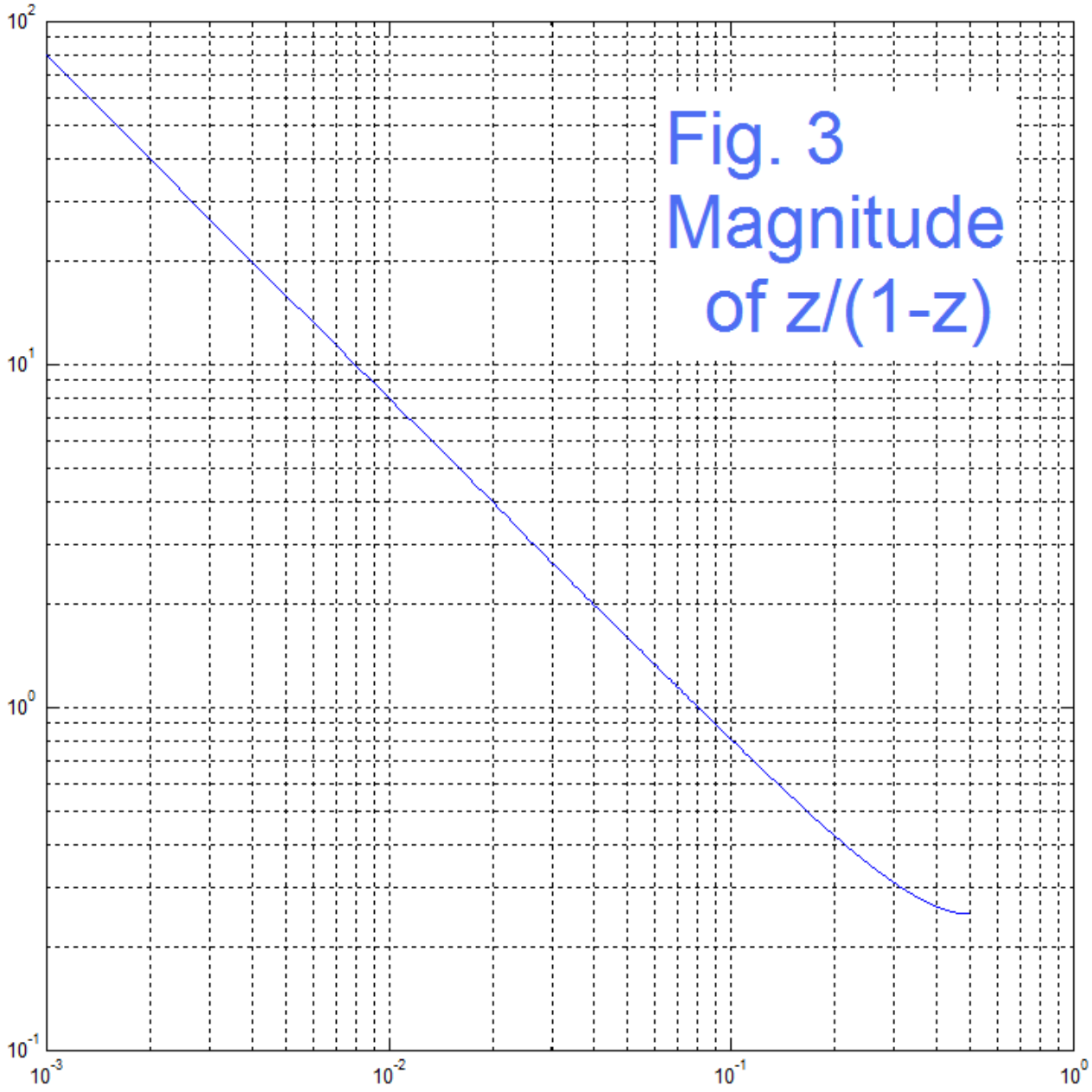


Fig. 3
Magnitude
of z/(1-z)
10^2
10^1
10^0
10^{-1}
10^{-3}
10^{-2}
10^{-1}
10^0

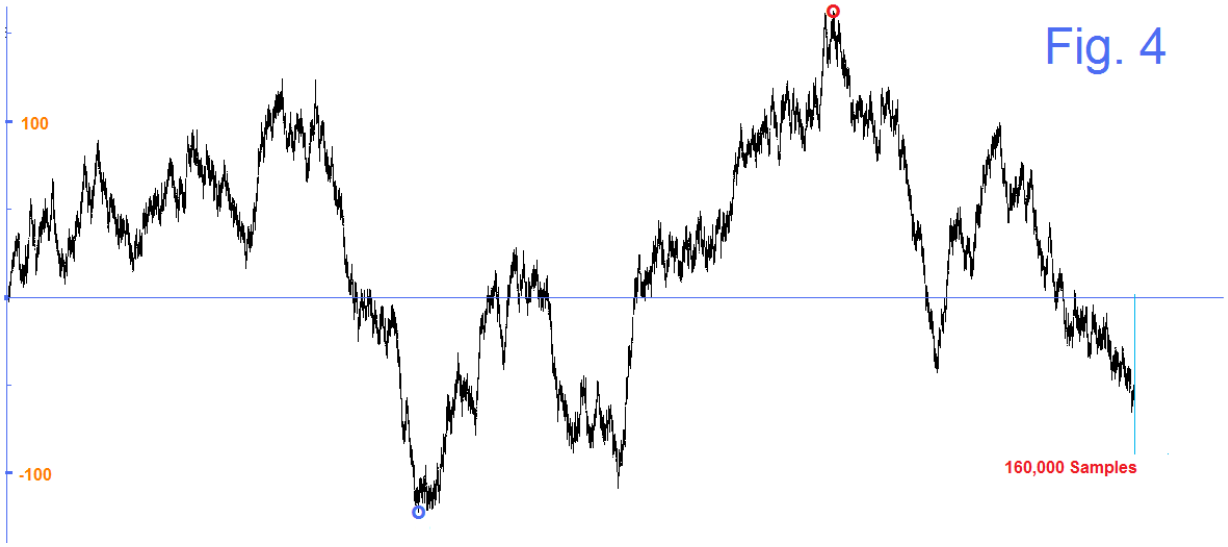


Fig. 4
100
-100
160,000 Samples

While the example of Fig. 4 started at zero, the starting point is arbitrary (because it is an integration). In general, we want to look at red noise signals that were either started at some random value, or which are selected well into the integration process. Some may well be all positive (for the selected region observed), all negative, or have several or

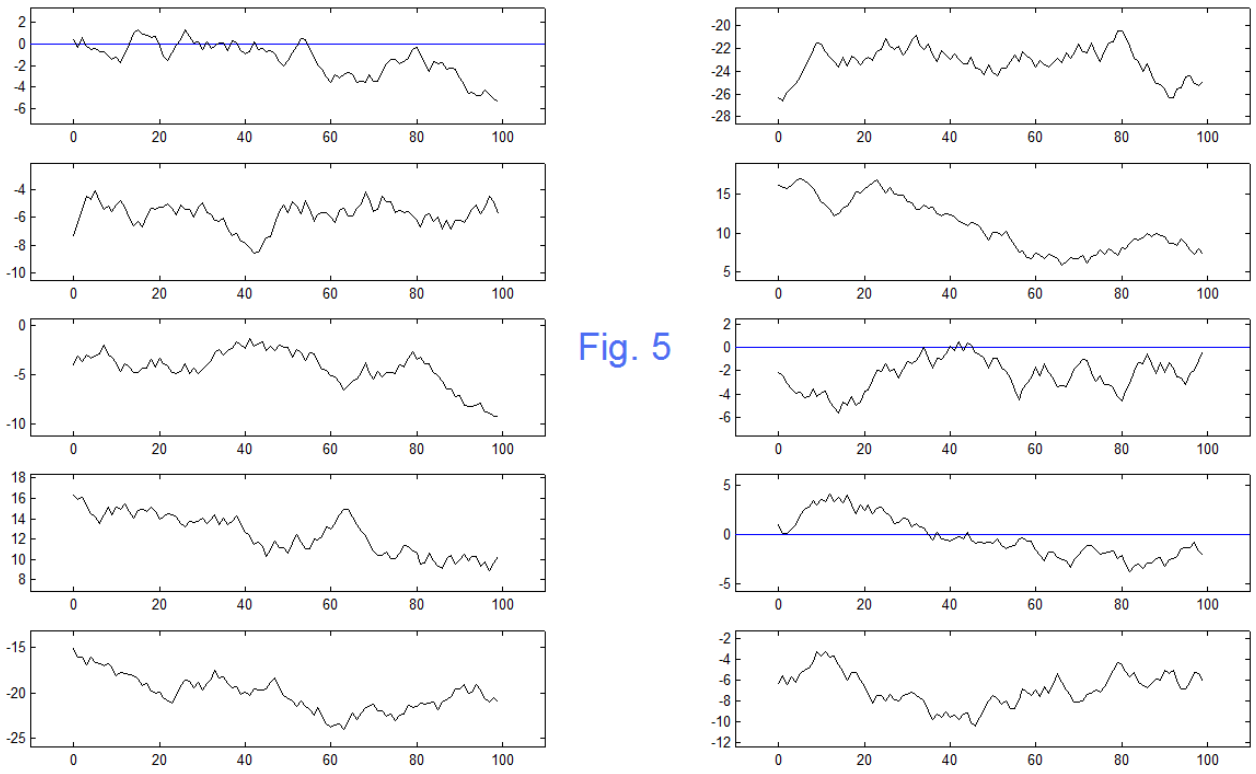


Fig. 5

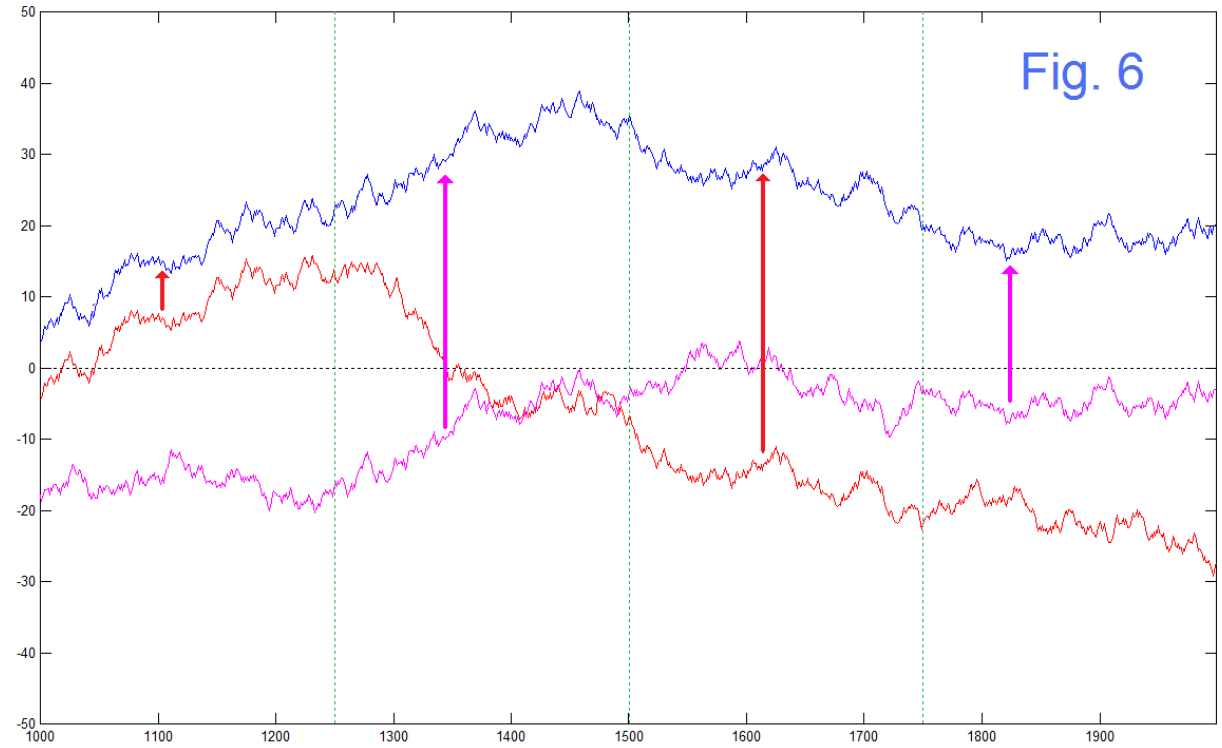


Fig. 6

many zero crossings. (In Fig. 5, a blue axis is included in the three examples that do cross zero). The 100 samples for each of the ten examples are actually samples 500-599 of sequences started at zero. Note the variability.

Fig. 6 is an entertaining case where we have two red noise signals (red and magenta) generated by separate white noise sequences. A third, perfectly good sequence (blue) is generated by jumping from one white noise to the other every 250 samples. This is equivalent to chopping out a segment of the red or magenta sequences and splicing it to the blue, matching the endpoint of the so-far constructed blue segment. The arrows show the apparent displacements of the segments. This is of course not the same as one would obtain if we were to splice the segments without matching the ends.

FILTERING WITH THE FFT

We mentioned above that we were interested in generating red noise in the spectral domain. This is of interest beyond offering an alternative to the integration, as it is more obvious how weighting an FFT could allow a highly customized spectral shape. The following code snippet shows the general idea for generating length 1000 red noises.

```
% fft method

k=0:500;
rk=1./k;                    % reciprocal
fil=[rk  rk(500:-1:2)];     % “filter”
fil(1)=0;                    % throw out infinity at 0
%
x=2*(rand(1,1000)-0.5);   % white
X=fft(x);
XF=X.*fil;
xr=500*real(ifft(XF));
```

We are about to run away and look at a second interesting topic (positions of maxima and minima) so for the moment we will just plot a selection of 10 red noise signals (Fig. 7) as generated by the FFT process, similar to those from the integration process in Fig. 5. Later we will again use the FFT generation process.

POSITIONS OF MAXIMA AND MINIMA

Fig. 4 was our length-160,000 red noise signal. If you look closely at the plot, we see for one thing, a maximum sample (red) and a minimum sample (blue) have been circled. This simple location and marking is easy enough to understand. We note in passing as well that the local extremes here are quite large (over 100) so the “balance” between positive and negative contributions can conspire to keep much (most) or the curve well

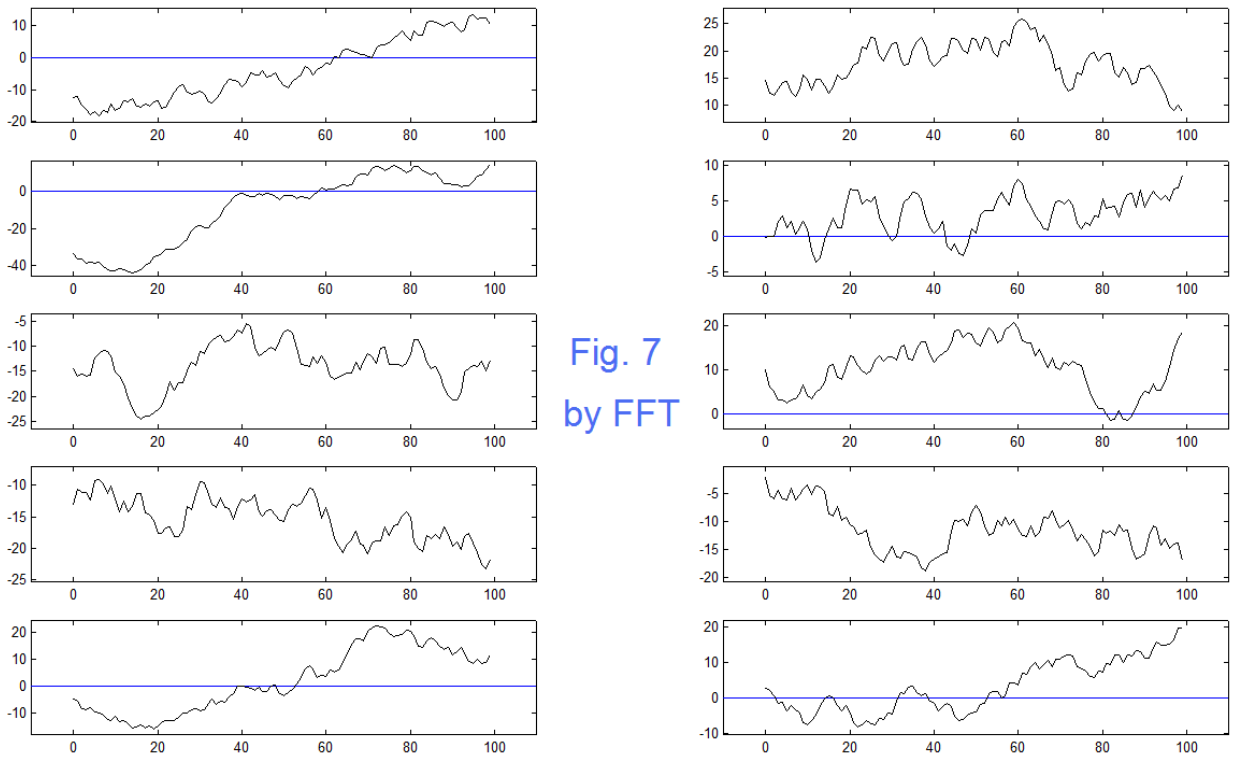


Fig. 7 by FFT

away from zero. Here we are integrating Gaussian noise, but it is similar to a coin-flip random walk. So the imbalance is something like 100-200 heads (or tails) in 40,000 or so trial flips. A lot - but not that surprising. But then it turns around and heads not only to zero but to large negative values. This is perhaps more of a surprise. We kind of have to believe that the signal really has (eventually) larger and larger amounts of lower and lower frequencies. Quite fascinating.

So we are left to speculate about where (at what time instances) we expect maxima and minima to occur. Likely we suspect that since the signals are based on random noise that it would be equally likely that they could occur anywhere. In fact, this is true for white noise. And, unless one has had occasion to look (with amusement) at a lot of red noise signals, we might not appreciate that the finite duration red signals, due to low frequency trends, will have an predilection for maxima/minima relatively close to the ends. Examine the curves of Fig. 5 and Fig. 7 to get the general idea. Recently [2] on a blog posting it was noted that extreme values of certain time sequences (said there to be autocorrelated) tend to occur near the ends. As we say, if we think of red-signals (or many other if not all non-white signals) as being correlated, this is immediately plausible. If we had to guess the next value of a white noise sequence, we would guess 0 (or whatever the mean is known to be). If we had to guess the next value of a red noise sequence, we would guess the current value. So an intuitive explanation is at hand. It is not so easy however to clearly quantify this tendency, nor is it obvious (to me at least) how one would calculate a theoretical imbalance. Arbitrarily we will just choose to say that a maximum or minimum is "close to an end" if it is within 10% or an end, and is not close to a end if it is in the middle 80%. This is what we will use here.

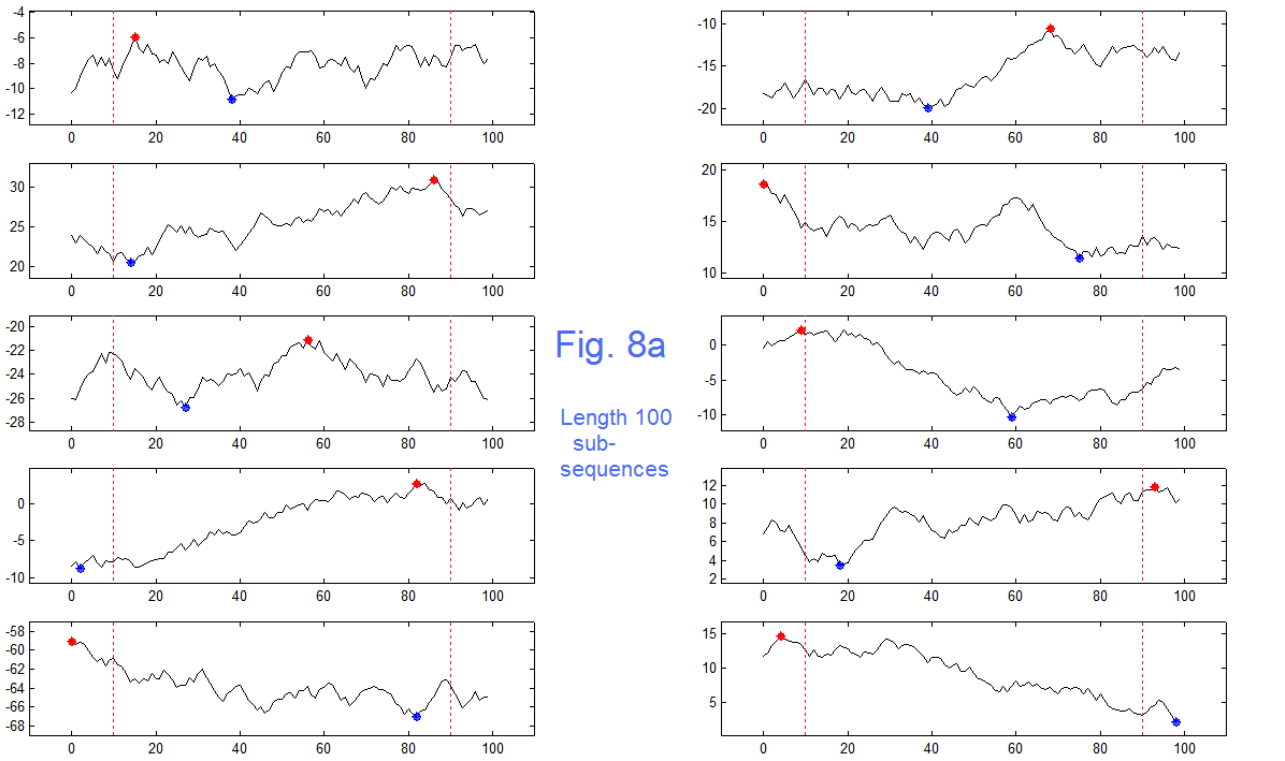


Fig. 8a

Length 100 sub-sequences

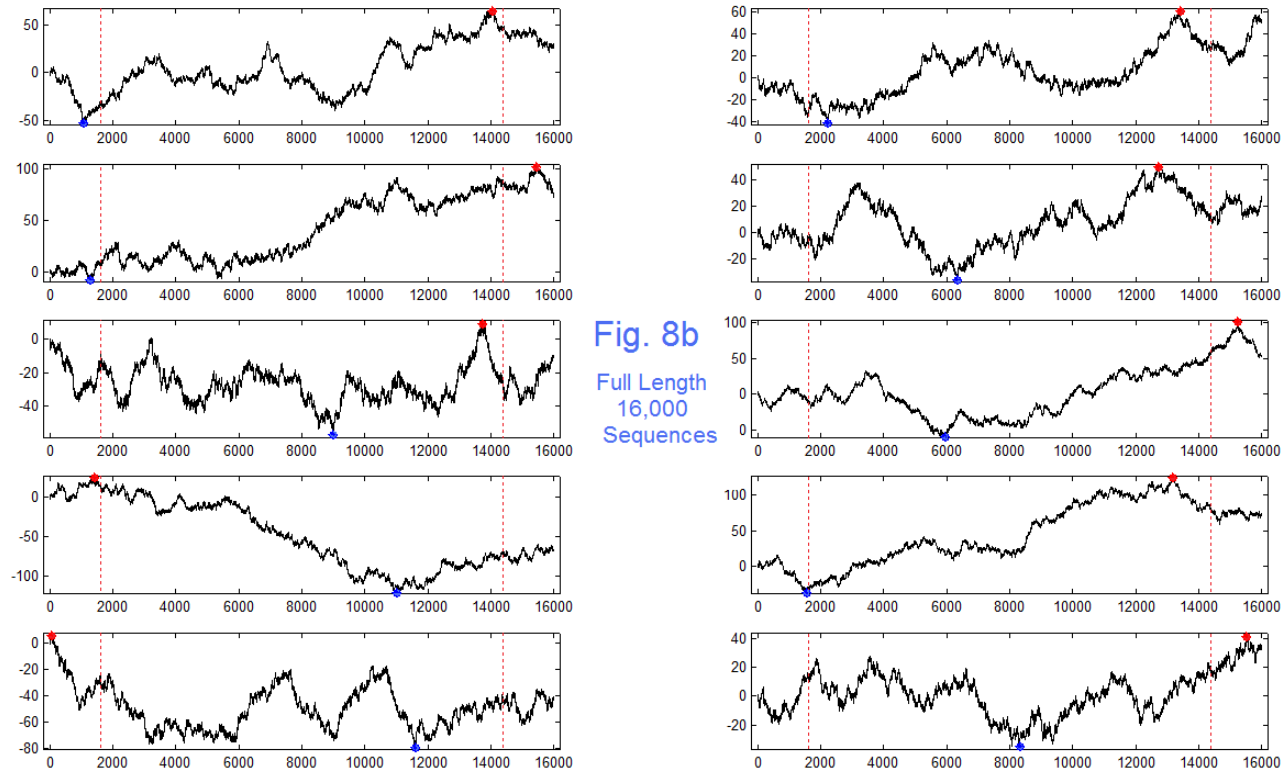


Fig. 8b

Full Length 16,000 Sequences

In Fig. 8a and Fig. 8b, we see 10 examples of red noise signals generated by the integration procedure. Fig. 8b shows sequences that are 16,000 samples long while the ones in Fig. 8a are just 100 long, being samples 4000-4099 of corresponding sequences (same subplot position) in Fig. 8b. Here we again (as in Fig. 4) mark the maximum (red) and the minimum (blue). In addition we have vertical red dashed lines to indicate the first 10% and the last 10%. Thus we can more easily judge a particular maximum or minimum to be close to an end. Each run has a maximum and a minimum of course, giving two opportunities for a max/min to be close to end. So each of the 10 runs is scored at 0, 1, or 2. Fig. 8a has seven such occurrences, or 7/20 = 35%. By chance we might have expected 20%.

Shortly we will do many more trials and use automatic counting, in a test that makes use of the FFT method. For the moment we note that the corresponding length-16,000 cases (Fig. 8b) have a similar result, 8 possibilities out of 20 or 40% of min/max near the ends. This was done in case there was reason to suppose that the result depended on the length, which it apparently does not. Roughly speaking, based on what we have plotted here, we seem to have twice as many cases as the expected 20% with white noise.

SPECTRAL SHAPES: WHITE, RED, OTHERS

At this point, we have two remaining things to look at. First, we can consider some additional shapes other than just white or red. Second, we understand pretty well the fact that for white noise 20% of the min/max fall in a end range totaling 20%. We also got something like 40% for red. We want to pin these down a bit more by making a very large umber or trials.

Our trial Matlab code used here is shown below, and it uses the FFT method from the snippet on page (6) and then runs 1,000,000 trials and counts up min/max occurrences close to the ends. A few things about the program may be noted. First, instead of forming the FFT "filter" as 1/k we form it is $1/k^a$. This makes the a=0 case just white noise, and the FFT←→IFFT is really unnecessary for this case (we do it anyway). The a=1 cases is red noise. We will try two other cases, a=1/2, a form of "pink" noise and a=2 which is very highly correlated and which we will call here "Dark Brown".

PROGRAM 2

```
% redex11
% fft method, length 1000, test max/min to ends
endguys=0
a=1          % red
%  form filter
k=0:500;
k(1)=0.000001;
rk=1./(k.^a);
fil=[rk  rk(500:-1:2)];
fil(1)=0;
```

```
N=1000000     % trials
for mmm=1:N
x=2*(rand(1,1000)-0.5);
X=fft(x);
XF=X.*fil;
xr=500*real(ifft(XF));
xrtest=xr(500:599);     % select sub-sequence

[xrtestmax,ixmax]=max(xrtest);
if ixmax<11; endguys = endguys+1; end
if ixmax>90; endguys = endguys+1; end
[xrtestmin,ixmin]=min(xrtest);
if ixmin<11; endguys = endguys+1; end
if ixmin>90; endguys = endguys+1; end

end

endguys
endguyspercent=endguys/(2*N)
```

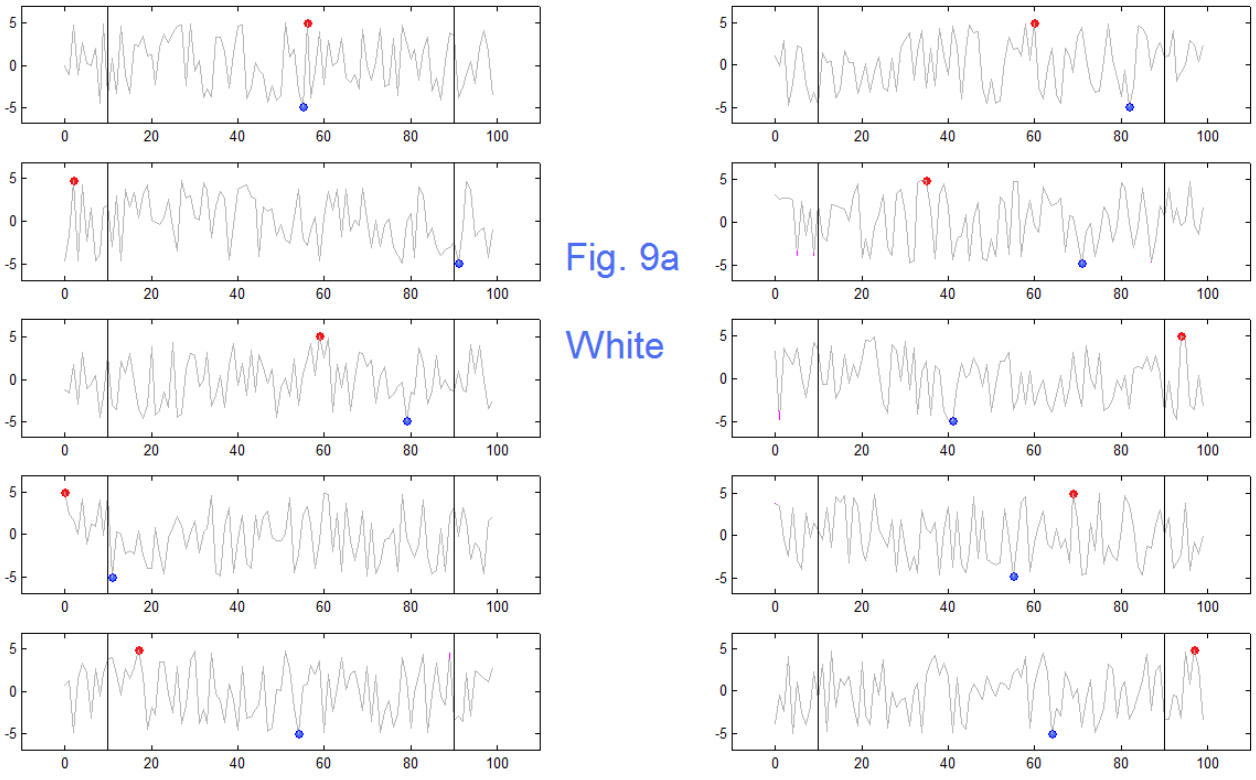


Fig. 9a White

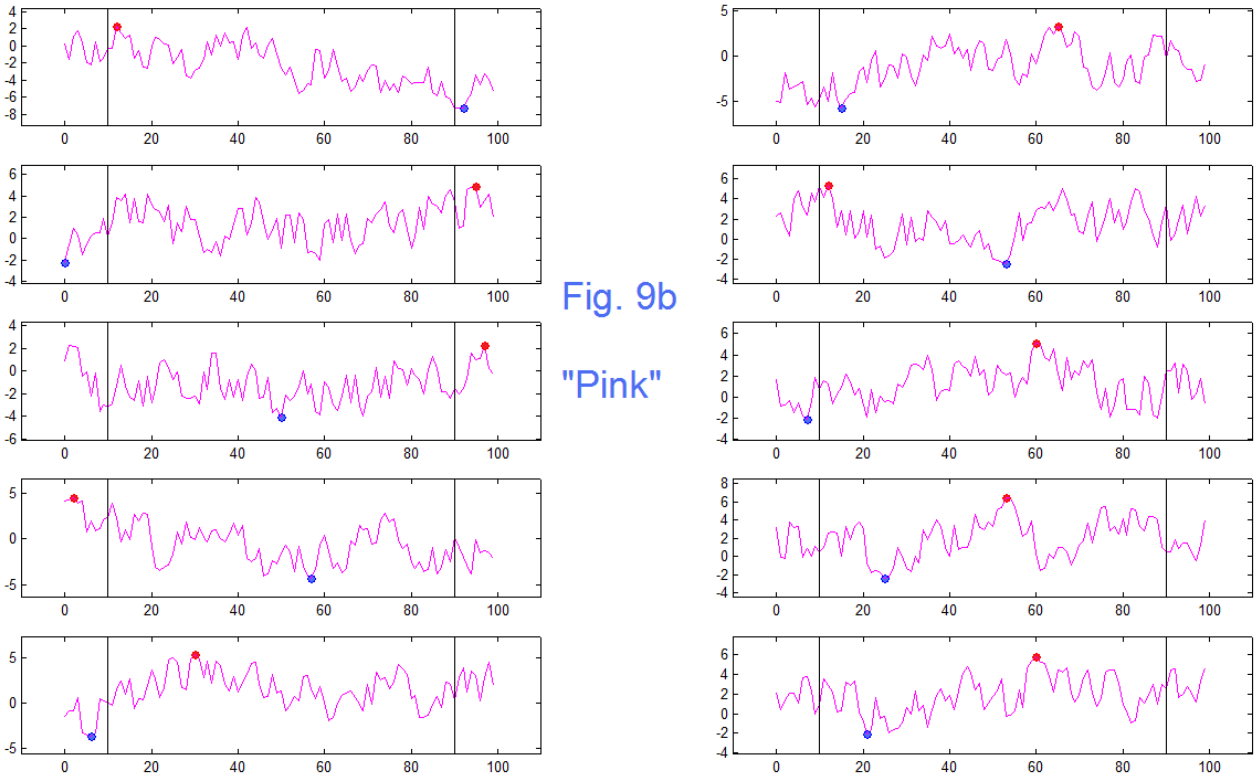


Fig. 9b

"Pink"

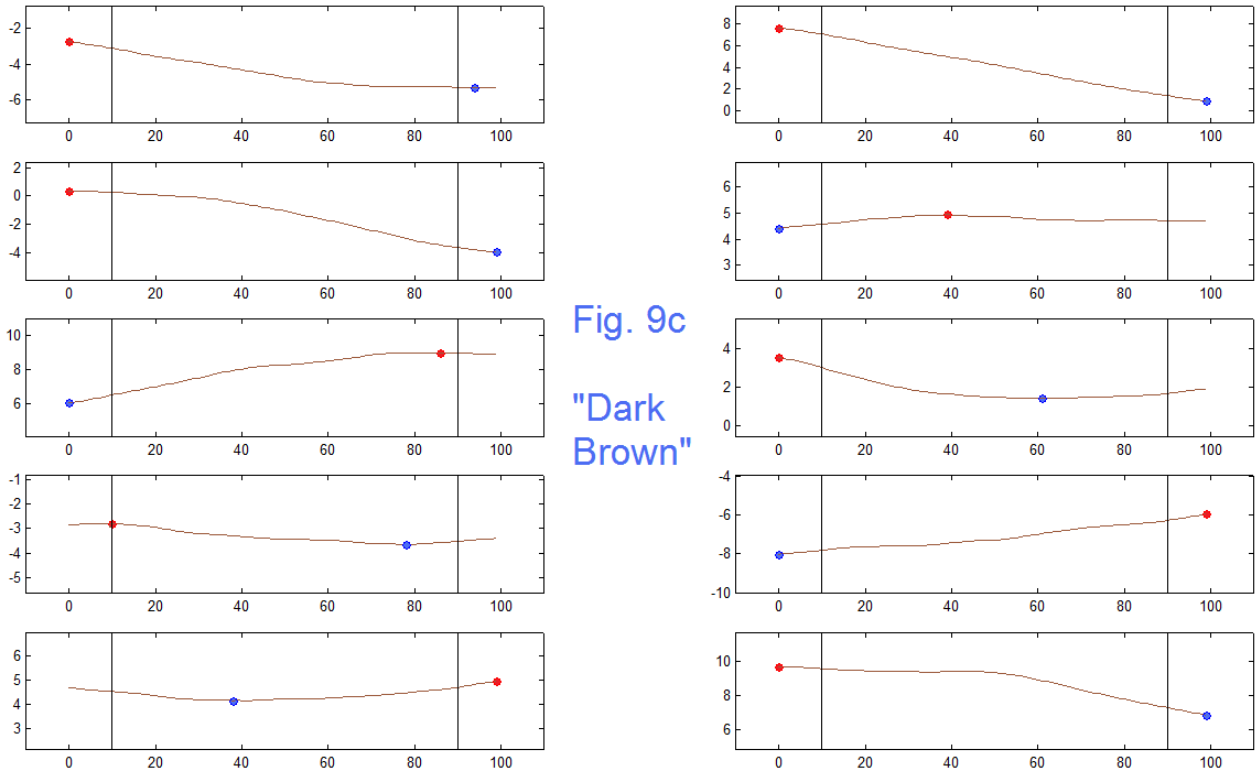


Fig. 9c

"Dark Brown"

Fig. 9a shows the white noise case. As we said, it is easy to believe that just about any of the white noises is "typical" but we need 10 here to show a very rough idea of the occurrences of min/max in the end regions. This particular case has in fact exactly 4 extreme values in the end regions. Since there are 10 trials, and since each trial has two possible instances of extremals near the ends, we indeed kind of expected 20 x 20% or four of them. Other runs (100 trials) gave from 1 to 12, with an average of 4.29. The computer eventually runs a million trials.

Fig. 9b is a "Pink" noise case which we would expect to be somewhere between white and red. The 10 trials here have in fact 7 extremal values in the endpoint region, but we learn very little from 10 trials. Below we will be doing a million of these too. It is probably evident that if we compare Fig. 9b to the red cases (compare Fig. 5, 7, and 8a) we see the pink cases seems to have a bit more "fuzz" on it. Pretty much an expected result.

The third new (non-red) case here uses the $1/k^2$ roll-off which is pretty drastic as seen in Fig. 9c. (Keep in mind that we are skipping the red examples between Fig. 9b and Fig. 9c). These are little more than slightly curved titled lines for the most part. Now most all the extreme values are in the end regions, often at the very ends. The computer counted 14 of the extremals in the end regions (70% in this very limited trial). This too will get 1,000,000 trials.

So – here is the data for a million trials (two million possibilities actually, max and min).

COLOR	a	% in End Regions
White	0	19.96
Pink	1/2	27.05
Red	1	40.78
Dark Borwn	2	80.76

Nothing here is much of a surprise. Unfortunately, except for the white noise case, we don't have much of an idea if there is a theoretical basis for these experimental results.

REFERENCES

[1] B. Hutchins, "Fun with Red Noise", Electronotes Application Note no. 384, September 1, 2012
http://electronotes.netfirms.com/AN384.pdf

[2] W. Eschenbach, "Extreme Times", Watts Up With That blog, April 24, 2014
http://wattsupwiththat.com/2014/04/24/extreme-times/