==========PWL_Spectrum_Noise_Scale==========

Because noise is often formated in terms of a RMS value, this can cause some confusion when viewing a PWL random signal on a spectrum plot.

Charles D.H. Williams (the man behind MacSpice) has written a spice file which can generate a Piece Wise Linear Noise source. This spice file usually is installed in the MacSpice folder of the Document folder.

Calling out the file name and sample rate and total period in a MacSpice console will create the file.

==========Create_PWL_Noise_File==========

```
MacSpice 2 -> rndsrc .5m   1
PWL_File.inc has been created in the MacSpice folder
MacSpice 3 ->
```

==========

In this case the sample rate will be .5ms and the total time is 1 second.

The Full code for this spice file is shown below. Except for what is red, this is Charles code.

=========="rndsrc"_File_Needs_To_Be_In_MacSpice_Folder======

```
* rndsrc -- by CDHW -- writes a gausian random voltage source
*
* Note: see also the frontend command 'compose'
*
.control
begin
    setplot new
 set outfile = "PWL_File.inc"
 if ($argc = 2)
  let step = $argv[1]
  let duration =  $argv[2]
 else
  echo "usage  - rndsrc timestep duration"
  echo "effect - gaussian source written to file -- $outfile"
  unset outfile
  goto done
 endif

 set parity = true
 let time = 0
 echo "VpwlT OUT 0 PWL(" > $outfile

 while time < duration
  let time = time + step
  if $parity
    let X1 = (1+rnd(32768))/32769
    let X2 = rnd(32768)/32768*8*atan(1)
    let Vnoise = sqrt(-2*ln(X1))*cos(X2)
    set parity = false
  else
    let Vnoise = 1.3*sqrt(-2*ln(X1))*sin(X2)
    set parity = true
  endif
  echo "+ $&time $&Vnoise" >> $outfile
 end
 echo "+ )" >> $outfile
 unset outfile parity
 label done
 destroy
end

echo    "PWL_File.inc has been created in the MacSpice folder"

.endc
```

==========="PWL_File.inc"_in_MacSpice_Folder================

The output file is in this format...

```
VpwlT OUT 0 PWL(
+ 0.0005   1.04178
+ 0.001    0.772328
+ 0.0015 -0.276687
+ 0.002  -0.314243
+ 0.0025   0.412586
+ 0.003  -0.619019
+ 0.0035   1.81929
.....
+ 0.9995 -0.432622
+ 1      -0.0988927
+ 1.0005 -0.726563
+ )
```

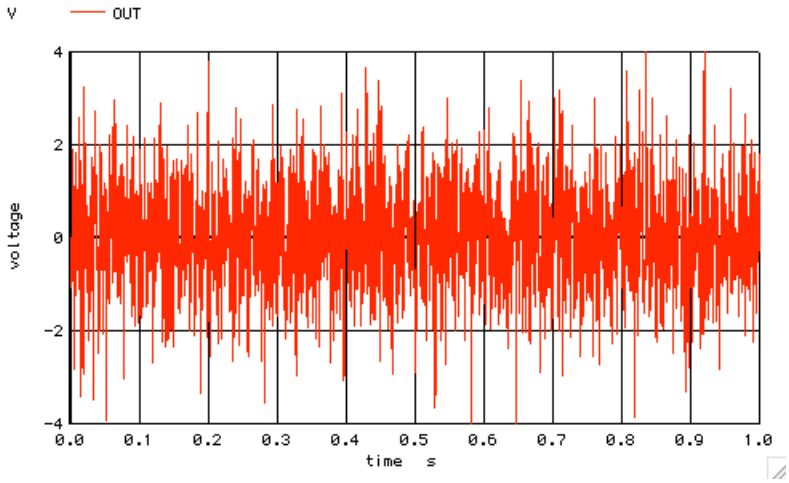
==========PWL_Noise_Simulation=========================

An include statement can import this random noise waveform into a transient simulation.

==========Simulate_Noise_in_Transient=========

```
PWL_Noise_1VRMS_@1KHz
* www.idea2ic.com
* dsauersanjose@aol.com  1/17/10   replace(OPT-SPACE)=>SPACE
*    OUT     Rload
*      _____/\  __          need to have a file called
*    _|_     \/  |         "rndsrc" in the "MacSpice"
*   /   \        |         folder inside "Doucments"
*  /VpwlT\       |
*  \     /      _|_        First typing in a MacSpice
*   \___/       ///        window "rndsrc .5m 1"
*     |         Gnd
*     |                    Then run this file
*    _|_
*    ///
*    Gnd
* timestep = .5m      means 1KHz bandwidth
* duration = 1        means 1Hz resolution
* type into a         MacSpice window =>  "rndsrc .5m 1"
* it will generate    PWL_File.inc in this format..
* VpwlT OUT 0         PWL( + 0.0005 0.988835 +.....
*==========Circuit_Netlist=======================
.include    PWL_File.inc
Rload       OUT     0         1k
*TRAN       TSTEP   TSTOP     TSTART  TMAX        ?UIC?
.tran       .05m    1         0       .05m        UIC
*==========Run_Transient=========================
.control
run
plot        OUT     ylimit -3 +3
*================================================================
```

==========Eye_Ball_Noise================



RMS and standard deviation in this case are the same thing. It is convenient to set this noise to 1V rms. Ploting the noise shows that the standard deviation is about +/- 1V. Another eye ball test is that the noise should be within +/-3V for 99.9% of the time.

==========RMS_The_Noise==========

Why eyeball the RMS value when it can be directly calculated.

```
*==========Find_RMS_Input=======================================
let num =   length(out)-1
let i  =    0
let vpwr =  0
repeat      $&num
let i =     i +1
let vpwr =  vpwr  + (mag(OUT[i])*mag(OUT[i]))/num
end
let vrms1 = sqrt(vpwr)
echo        INPUT RMS =  $&vrms1
*================================================================
```

INPUT RMS = 1.00587

A little tweeking was done to "rndsrc" to set the result to be close to 1Vrms

==========Now_Do_The_Spectrum==========

```
*==========Find_Spectrum=================================
linearize
set         specwindow =    "rectangular"
*SPEC       FSTART  FSTOP    FSTEP   VECTOR
spec        1       10k      1       v(OUT)
*================================================================
```

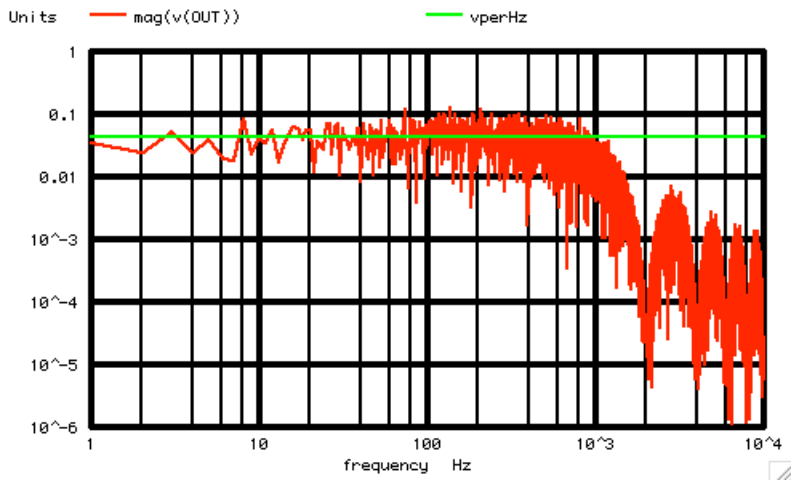
The total test time was set to 1sec and the transient is set to run at .05ms. This set the min and max frequencies to 1Hz and 10KHz.

The Noise signal it self has a sample rate of .5msec. This means the noise bandwidth should be 1KHz.

It might be convenient to average up all the noise in the spectrum output and plot it with the spectrum output.

```
*==========Find_Spectrum_RMS=================================
let num =   length(out)-1
let i  =    0
```

```
let vpwr =  0
repeat      $&num
let i =     i +1
let vpwr =  vpwr  + mag(OUT[i])* mag(OUT[i])
end
let vrms2 = sqrt(vpwr)
echo        SPECTRUM RMS =  $&vrms2
*==========View_Spectrum=========================================
let         BandW = 1000
let         vperHz = vrms2/sqrt(BandW)
set         pensize       = 2
plot        mag(v(OUT)) vperHz  loglog
*=================================================================
```



SPECTRUM RMS = 1.36041

The average of the noise appears to plot nicely on the spectrum output. But the RMS value is about 3dB higher than 1.

==========The_Surprise==========

MacSpice is formating a unity sine waves as unity in the spectrum output. But a unity sine wave has an RMS value of .707vrms. A 1.414 magnitude sine wave would have the same power as a 1Vrms noise signal. Macspice is not plotting noise as a RMS value. It is plotting both noise and a sine wave in a magnitude format.

==========Full_Netlist_For_Copy_Paste==========

```
PWL_Noise_1VRMS_@1KHz
* www.idea2ic.com
* dsauersanjose@aol.com  1/17/10   replace(OPT-SPACE)=>SPACE
*    OUT     Rload
*        _____/\__          need to have a file called
*     _|_      \/  |         "rndsrc" in the "MacSpice"
*    /   \         |         folder inside "Doucments"
*   /VpwlT\        |
*   \     /       _|_        First typing in a MacSpice
*    \___/       ///        window "rndsrc .5m 1"
*                Gnd
*     |
*    _|_                    Then run this file
*    ///
*    Gnd
* timestep = .5m     means 1KHz bandwidth
* duration = 1       means 1Hz resolution
* type into a        MacSpice window =>  "rndsrc .5m 1"
* it will generate   PWL_File.inc in this format..
* VpwlT OUT 0        PWL( + 0.0005 0.988835 +.....
*==========Circuit_Netlist=======================
.include    PWL_File.inc
Rload       OUT     0       1k
*TRAN       TSTEP   TSTOP   TSTART  TMAX        ?UIC?
.tran      .05m     1       0       .05m        UIC
*==========Run_Transient=========================
.control
run
set         pensize =  1
```

```
plot        OUT     ylimit -4 +4
*===========Find_RMS_Input=========================
let num =   length(out)-1
let i  =    0
let vpwr =  0
repeat      $&num
let i =     i +1
let vpwr =  vpwr  + (mag(OUT[i])*mag(OUT[i]))/num
end
let vrms1 = sqrt(vpwr)
echo        INPUT RMS =  $&vrms1
*===========Find_Spectrum=========================
linearize
set         specwindow =    "rectangular"
*SPEC       FSTART  FSTOP    FSTEP   VECTOR
spec        1       10k      1       v(OUT)
*===========Find_Spectrum_RMS=========================
let num =   length(out)-1
let i  =    0
let vpwr =  0
repeat      $&num
let i =     i +1
let vpwr =  vpwr  + mag(OUT[i])* mag(OUT[i])
end
let vrms2 = sqrt(vpwr)
echo        SPECTRUM RMS =  $&vrms2
*===========View_Spectrum=========================
let         BandW = 1000
let         vperHz = vrms2/sqrt(BandW)
set         pensize       = 2
plot        mag(v(OUT)) vperHz  loglog
.endc
.end
```

rndsrc .5m 1

2.12.10_2.39PM

dsauersanjose@aol.com
Don Sauer
http://www.idea2ic.com/