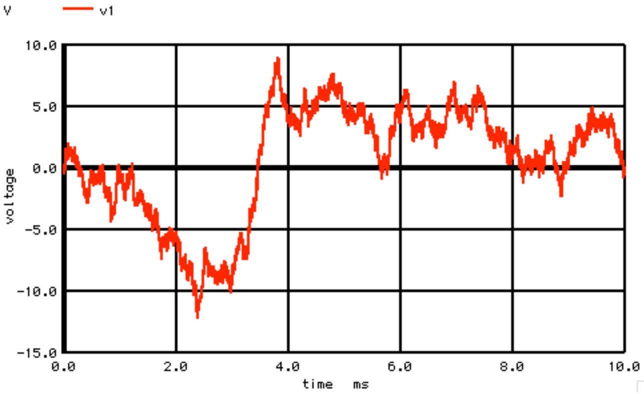
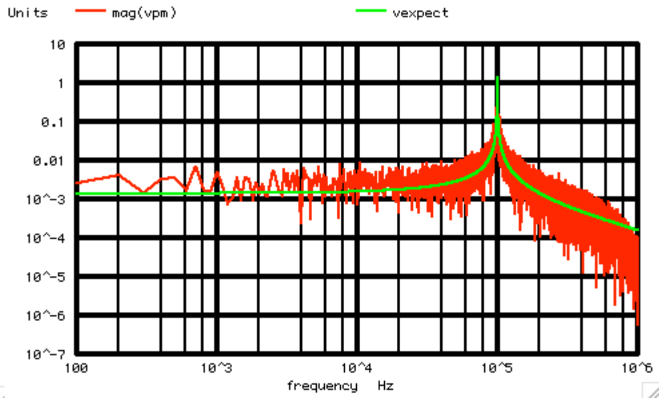
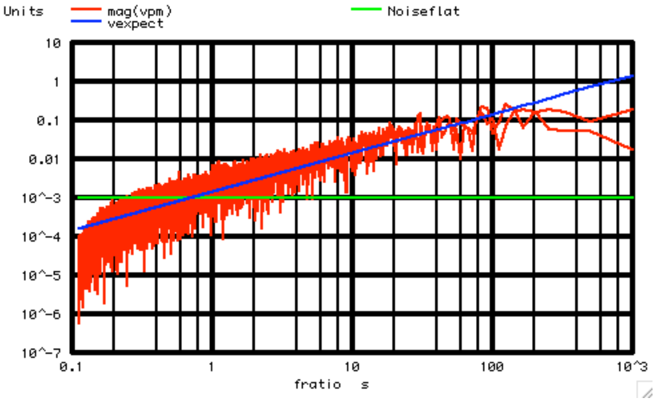
```
=====================MAPPING_PHASE_NOISE=====================================

1)  Oscillator phase noise is an accumlation of period timing tolerance error.
2)  A +/- .1_pk  radian Phase Modulation maps to two -26dB sidebands
3)  A +/- .1_rms radian Phase Modulation maps to two -23dB sidebands
4)  Flat Randomness Phase Modulation is spread out over all FFT frequency bins.
     Flat_Expect_per100Hz = .1*.5*sqrt(2)/sqrt(5000)   = 1m
5)  Limited accumulated phase rms approaches SD*sqrt(Num_periods)/sqrt(2*4)
    V1_Expect_rms_10ms    = 0.1*0.707*sqrt(10000)/2    = 3.5
6)  Apply Limited accumulated phase Modulation to the Carrier
    Vpm                   = cos(2*PI*100k*V(VTime)+ V(V1))
7)  Define Fratio          = Freq_Carrier_Hz/mag(Frequency_Hz-Freq_Carrier_Hz)
    Fratio                  define the level of the accumulation process
    Fratio                = 100k/mag(frequency-100k)
8)  When Fratio = 1         two periods of carrier should have sqrt(2) the normal noise
9)  Expected_spectrum     = Noiseflat*fratio*sqrt(2)
10) Plot Real_spectrum      Expected_spectrum  Flat_Expected vs Fratio
```





The accumulated randomness for 0.1_rms radians added up to around 3.5_rms over 10k samples at 1us each. The spectrum of Phase Modulation using the accumulated randomness shows the expected curve. It is possible to predict this curve.

```
=======================Want_10000_1us_steps===============================
Total_Period_s =       0.01
Bin_Resolutio_Hz =     100
Sample_Period_s =      1E-06
Nyquist_Hz =           500000
Total_Bins =           5000
=======================Create_PWL_array_and_Index_and_Plot==============
=======================Add_.1Vrms_Noise_to_PWL_array====================
=======================Adjust_Endpoint===================================
=======================Find_Ave_Rms_V1===================================
RMS_level_Expect       0.1*0.707*sqrt(10000)/2
RMS_level_Expect       3.535
RMS_level_RM           3.22762
=======================Install_the_PWL_array============================
=======================FFT_and_Plot_VPM===================================
FFT_BandWidth_Hz=      1E+06
FFT_resolution_Hz=     100
Flat_Noise             .1*.5*sqrt(2)/sqrt(5000)
Flat_Noise_Expect      0.001
fratio                 100k/mag(frequency-100k)
vexpect                Noiseflat*fratio*sqrt(2)
=======================done===============================================
```



The definition of Fratio (which can be thought of as an accumulation factor) is this.

```
Fratio          = Freq_Carrier_Hz/mag(Frequency_Hz-Freq_Carrier_Hz)
```

When the real spectrum is plotted versus Fratio, a very linear relationship is present. When Fratio is at one, think of the noise as consisting of two time periods. And each time period has the normal (flatband) tolerance. Now the noise is being integrated, so a Fratio of 10 should have 10 times that value, and so on. In this case the noise is really limited accumulated randomness. So at some level of Fration the noise flattens out.

Note that the accumulation process is integrating the noise floor such that a factor of 10 increase in time results in an factor of 10 increase in noise floor. But the bandwidth of this noise floor is 10 time less. So the full RMS of accumulated noise over a factor of 10 mover time should be..

Increase_In_Gain/Decrease_In_Bandwidth = 10/sqrt(10) = sqrt(10)

Limited Accumlation noise for N samples appears to follow this equation.

LimitAcc_Noise_rms   = RMS*0.707*sqrt(N)/2

```
===================MacSpiceCode====================================================
MAPPING_PHASE_NOISE
*=========Create_Signal==================
VTime     VTime  0      DC     0    PWL(    0      0    1      1)
Vfreq1    Vfreq1 0      DC     2
V1        V1     0      DC     0
BMOD      VMOD   0      V      = cos(6.2831853*2000*V(VTime))
BPM       VPM    0      V      = 1*cos(6.2831853*100k*V(VTime)+1*V(V1))
BCOS      VCOS   0      V      = 1*cos(6.2831853*100k*V(VTime))

.control
*TRAN            TSTEP  TSTOP  TSTART TMAX   ?UIC?
echo             "=======================Want_10000_1us_steps================================"
let n =          10000
let tstep =      1us
let period_t =   n*tstep
let Bin_Hz =     1/period_t
let nyquist =    .5/tstep
let binsTotal=   nyquist/Bin_Hz
echo             "Total_Period_s =        $&period_t"
echo             "Bin_Resolutio_Hz =      $&Bin_Hz"
echo             "Sample_Period_s =       $&tstep"
echo             "Nyquist_Hz =            $&nyquist"
echo             "Total_Bins =            $&binsTotal"
echo             "=======================Create_PWL_array_and_Index_and_Plot=============="
let pwl_1 =      vector(2*n)*tstep*0.5
let ii =         vector(2*$&n)
echo             "=======================Add_.1Vrms_Noise_to_PWL_array====================="
let n2 =         n-1
let              pwl_1[0] = 0
let index =      1
repeat           $&n2
let vnoise =     .1414*(rnd(127)+rnd(127)+rnd(127)+rnd(127)+rnd(127)+rnd(127)+rnd(127)+rnd(127)-507.5)/102.879
let              pwl_1[1+2*index] = pwl_1[-1+2*index] + vnoise
let index =      index + 1
end
echo             "=======================Adjust_Endpoint====================================="
let endpt =      pwl_1[19999]
let index =      1
repeat           $&n2
let              pwl_1[1+2*index] = pwl_1[1+2*index] -1*endpt*index/10000
let index =      index + 1
end
let endpt =      pwl_1[1999]
echo             "=======================Find_Ave_Rms_V1====================================="
let averVal =    mean(pwl_1)
let noisAC =     pwl_1 - averVal
let RmsVal =     1*sqrt(mean(noisAC* noisAC))
let rms_exp =    0.1*.707*sqrt(10000)/2
echo             "RMS_level_Expect        0.1*0.707*sqrt(10000)/2  "
echo             "RMS_level_Expect        $&rms_exp  "
echo             "RMS_level_RM            $&RmsVal  "
unlet            averVal
unlet            RmsVal
echo             "=======================Install_the_PWL_array============================"
alter            @v1[pwl] = pwl_1
tran             .1u    10m     0       .1u
set              pensize = 2
plot             v1
echo             "=======================FFT_and_Plot_VPM====================================="
linearize
let              FFT_BandWidth_Hz =      1meg
let              FFT_resolution_Hz =     100
echo             "FFT_BandWidth_Hz=      $&FFT_BandWidth_Hz"
echo             "FFT_resolution_Hz=     $&FFT_resolution_Hz"
set              specwindow=             "rectangular"
spec             $&FFT_resolution_Hz     $&FFT_BandWidth_Hz   $&FFT_resolution_Hz     v(vpm)
let              Noiseflat =             .1*.5*sqrt(2)/sqrt(5000)
echo             "Flat_Noise             .1*.5*sqrt(2)/sqrt(5000) "
echo             "Flat_Noise_Expect      $&Noiseflat  "
let              fratio =                100k/mag(frequency-100k)
echo             "fratio                 100k/mag(frequency-100k)"
echo             "vexpect                Noiseflat*fratio*sqrt(2)"
let              vexpect =               Noiseflat*fratio*sqrt(2)
plot             mag(vpm) Noiseflat      vexpect  vs fratio loglog
plot             mag(vpm) vexpect        loglog
```

```
echo            "=======================done================================================="
.endc
.end

4.4.11_12.12PM
dsauersanjose@aol.com
Don Sauer
```