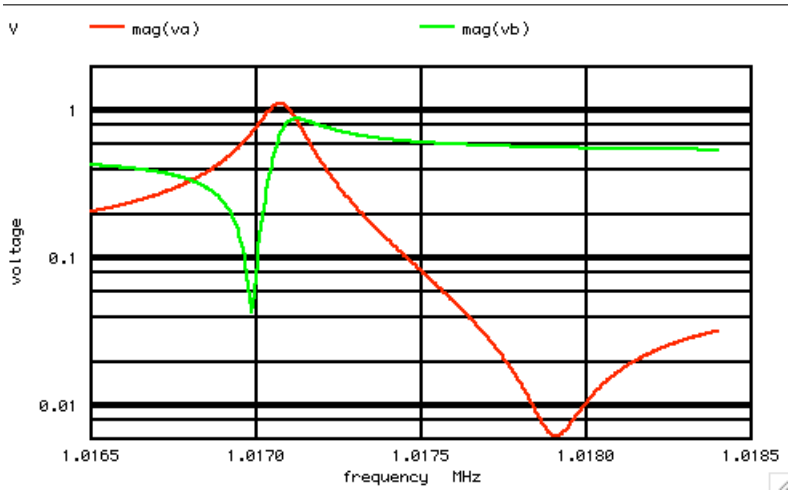
*==========Vector_Processing==========

Having working examples for everything has the advantage of not leaving any important details out. The following show how the functions that process waveforms work.

```
Wave_Function_Examples
VIN         VC      0       DC      AC      1
XCRY1       VA      VB      XCRYST
R1          VA      VC      1000k
R2          VC      VB      12.7k
C1          VA      0       16.5p
C2          VB      0       18p
.control
*AC         DECLin NUMDEC FSTART      FSTOP
ac          dec     600000 1.0165meg  1.0184meg
set         pensize = 2
```

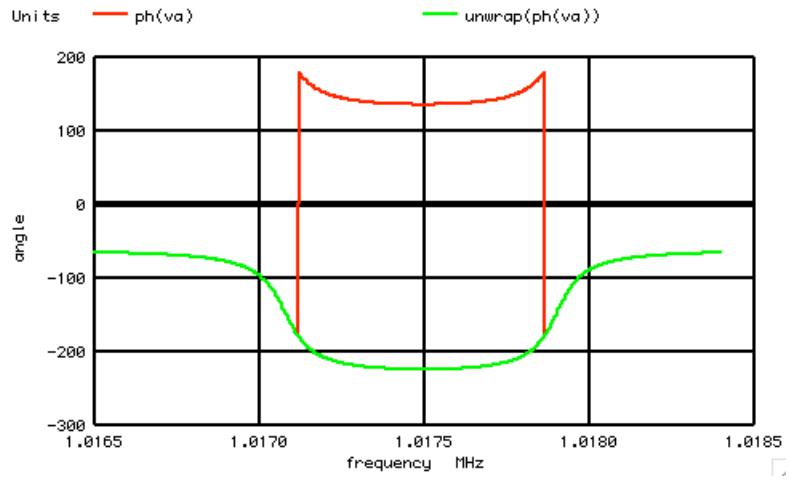
With the mag function, one can plot phase on a log scale.

```
plot       mag(va) mag(vb)     ylog
```



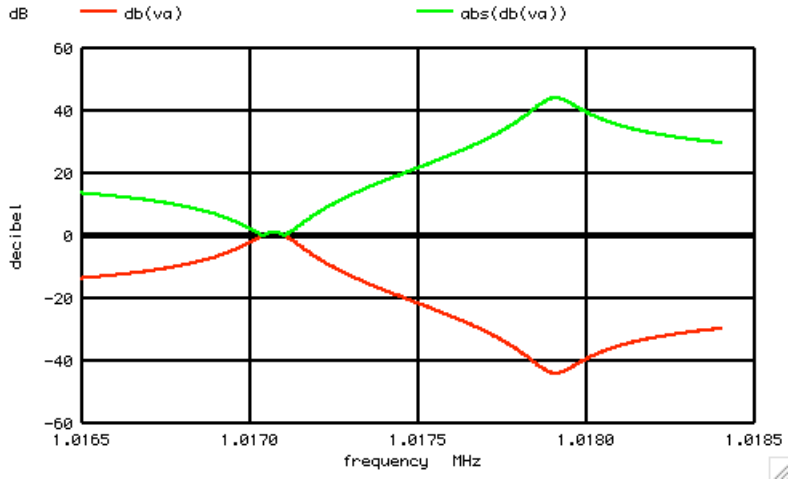
There is a handy unwrap function if one does not wish phase to automatically wrap around +/-180degrees

```
plot       ph(va)              unwrap(ph(va))          linear
```



The dB and abs function operate as expected.

```
plot       db(va)              abs(db(va))             linear
```

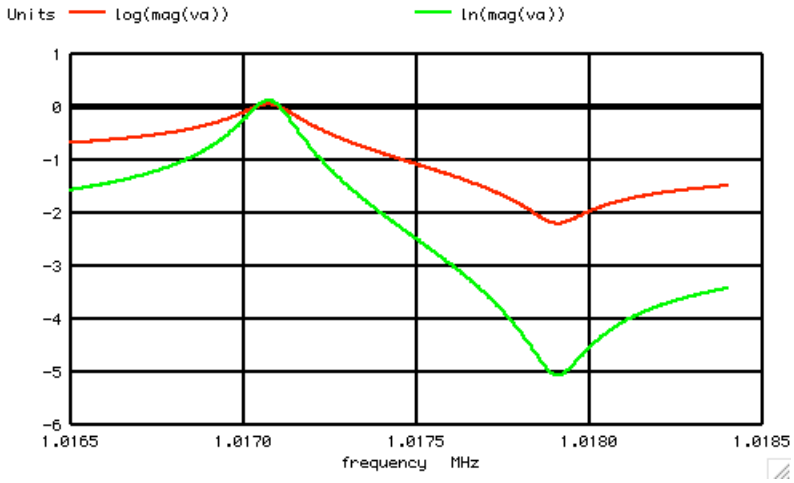


==========

The difference between the log and ln functions are just a difference in scale.

==========

plot	**log(mag(va))**	**ln(mag(va))**	**linear**

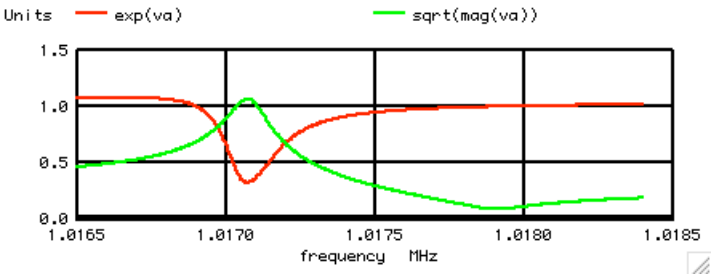


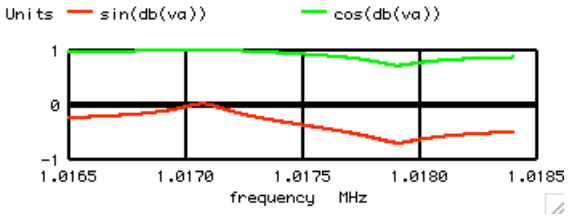
==========

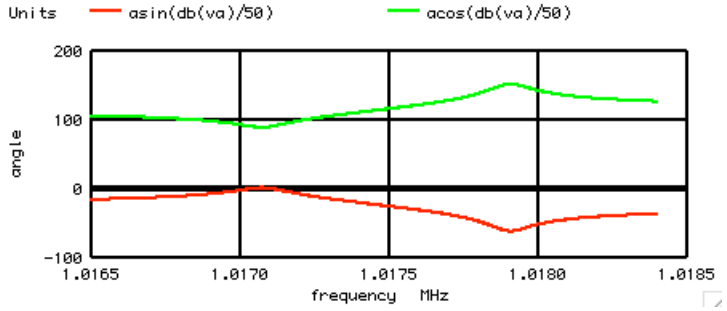
The math functions operate as expected. It looks like they may be using the degree format.

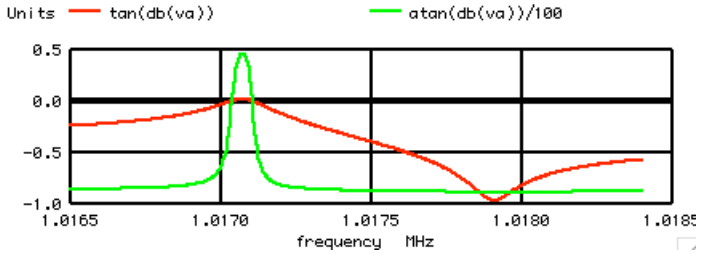
==========

plot	**exp(va)**	**sqrt(mag(va))**	**linear**
plot	**sin(db(va))**	**cos(db(va))**	**linear**
plot	**asin(db(va)/50)**	**acos(db(va)/50)**	**linear**
plot	**tan(db(va))**	**atan(db(va))/100**	**linear**







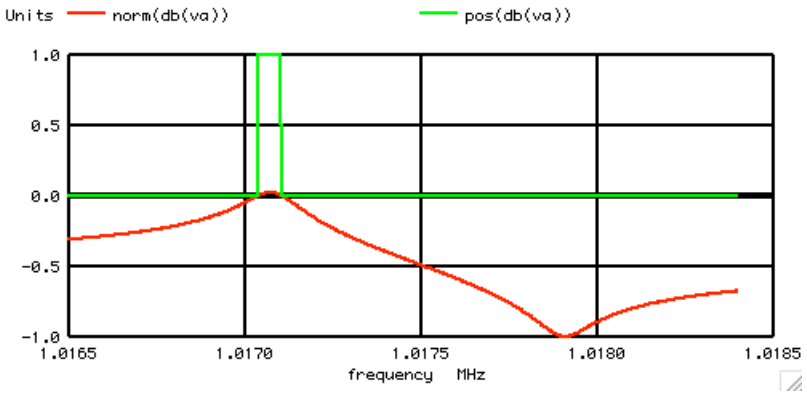


==========

The norm and pos functions scale to a +/- unity value.

==========

```
plot      norm(db(va))      pos(db(va))            linear
```

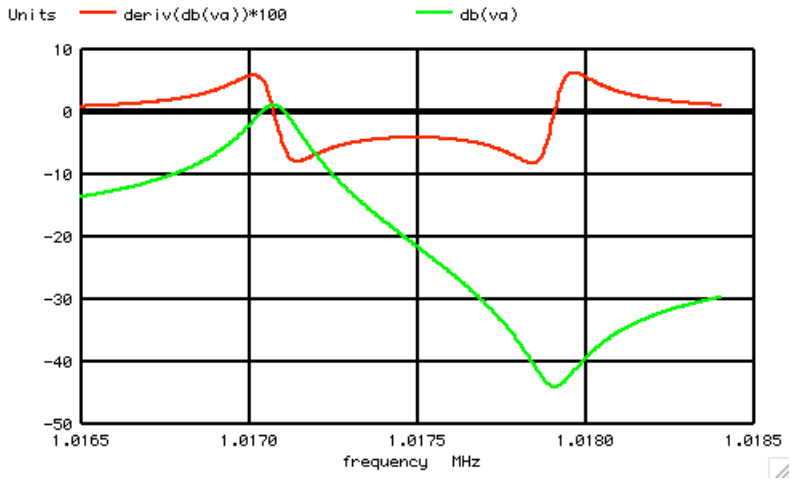


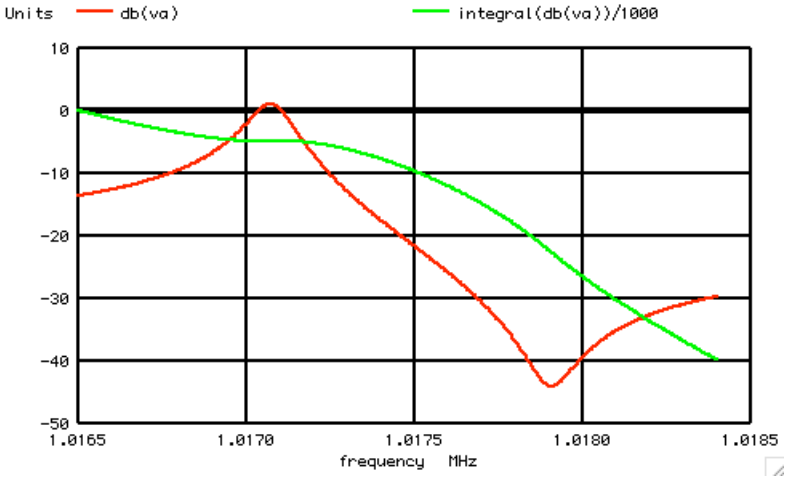
==========

The deriv and integral functions may needs some scaling to view the graphs.

==========

```
plot      deriv(db(va))*100 db(va)                 linear
plot      db(va)            integral(db(va))/1000  linear
```

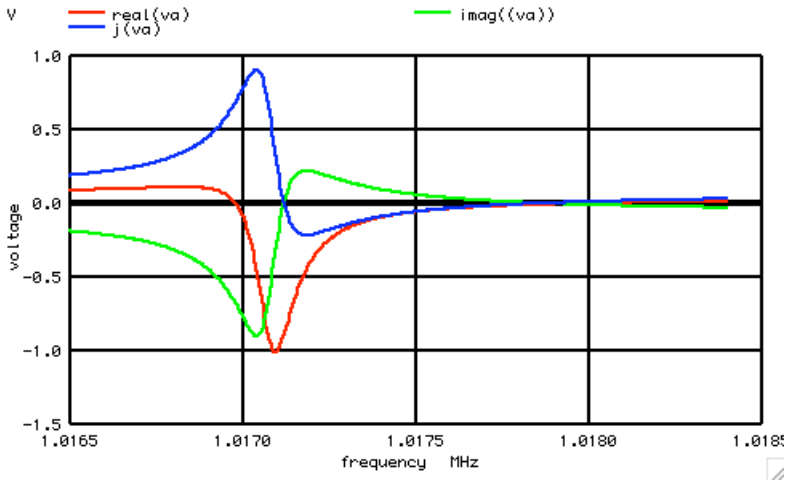




The waveforms come out as complex numbers. While it is common to use dB and ph, the real, imag can be used. Plotting with j shows that the complex values get multiplied by the sqrt(-1) and only the real value gets plotted.

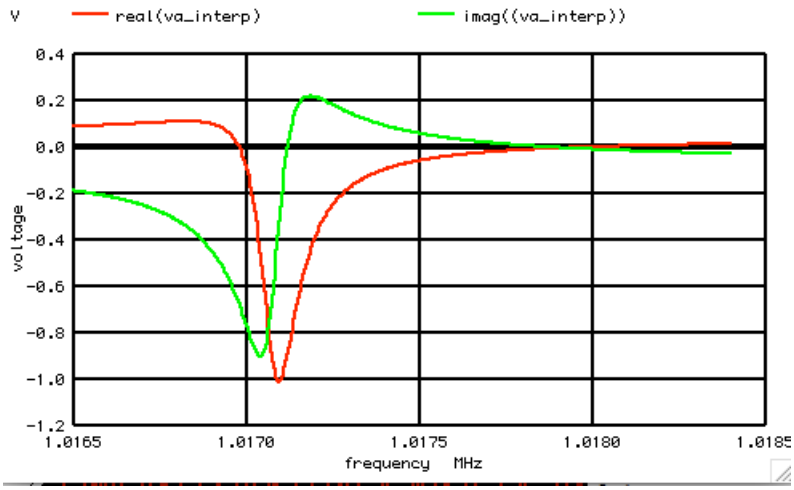
---

plot real(va) imag((va)) j(va) linear



There is an interpolate function that needs to be handled in the complex format.

```
let       va_interp =       interpolate(real(va)) +    j(interpolate(imag(va)))
plot      real(va_interp)   imag((va_interp))          linear
```



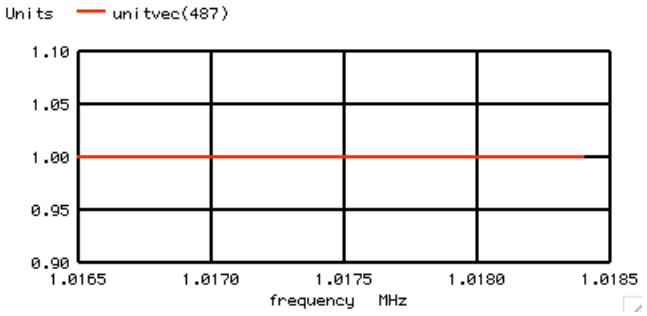
Many of the functions produce just a single value.

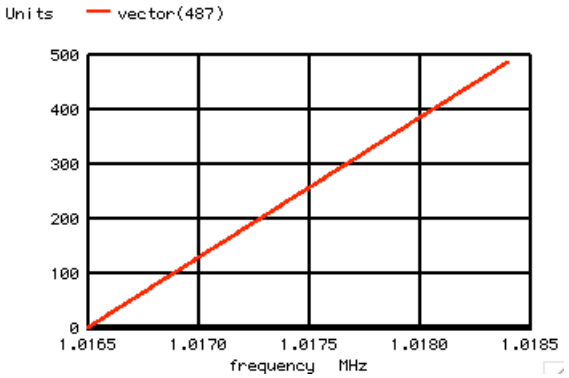
```
print     va[1] real(va[1]) imag (va[1]) j(va[1])    real(j(va[1]))
let       max_ph =          maximum(ph(va))
echo      "phase_Max  =      $&max_ph"
let       min_ph =          minimum(ph(va))
echo      "phase_Min  =      $&min_ph"
let       max_uw_ph =       maximum(unwrap(ph(va)))
echo      "UnWrap_phase_Max= $&max_uw_ph"
let       min_uw_ph =       minimum(unwrap(ph(va)))
echo      "UnWrap_phase_Min= $&min_uw_ph"
let       ave_uw_ph =       mean(unwrap(ph(va)))
echo      "UnWrap_phase_Ave= $&ave_uw_ph"
let       number =          length(va)
print     number
```

```
va[1]             = ( 8.80558e-02,-1.90259e-01 )
real(va[1])       =  8.80558e-02
imag (va[1])      = -1.90259e-01
j(va[1])          = ( 1.90259e-01, 8.80558e-02 )
real(j(va[1]))    =  1.90259e-01
phase_Max         =      179.067
phase_Min         =     -178.586
UnWrap_phase_Max = -65.1287
UnWrap_phase_Min = -224.255
UnWrap_phase_Ave = -136.067
number            =  4.87000e+02
```

The unit and vector functions now and again can come in handy.

```
plot      unitvec($&number) linear
plot      vector($&number)  linear
```



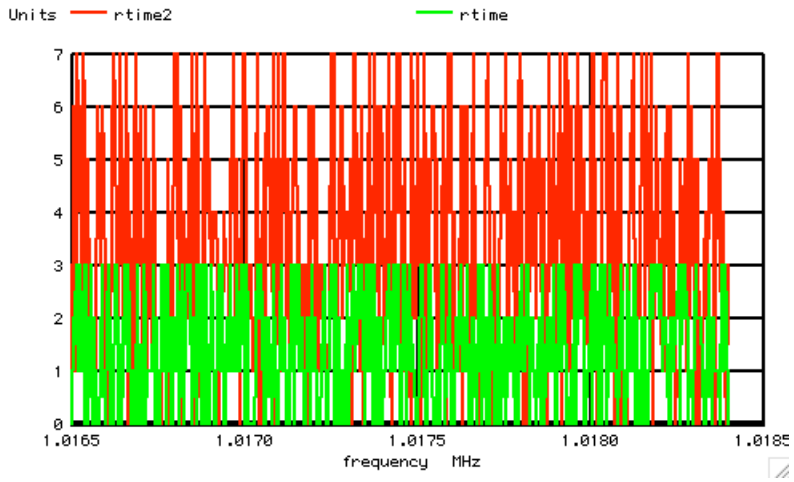


==========

It is also possible to build quantized random waveforms.

==========

```
compose   rtime  start = 0  stop = $&number            step =1
compose   rtime2 start = 0  stop = $&number            step =1
let       i = 0
repeat    $&number
let       i = i +1
let       rtime[i]  = rnd(4)
let       rtime2[i] = rnd(8)
end
plot      rtime2 rtime linear
```



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

```
Wave_Function_Examples
VIN       VC      0       DC      AC      1
XCRY1     VA      VB      XCRYST
R1        VA      VC      1000k
R2        VC      VB      12.7k
```

```
C1        VA     0      16.5p
C2        VB     0      18p
.control
*AC       DECLin NUMDEC FSTART     FSTOP
ac        dec    600000 1.0165meg  1.0184meg
set       pensize = 2
plot      mag(va)           mag(vb)                ylog
plot      ph(va)            unwrap(ph(va))         linear
plot      db(va)            abs(db(va))            linear
plot      log(mag(va))      ln(mag(va))            linear
plot      exp(va)           sqrt(mag(va))          linear
plot      sin(db(va))       cos(db(va))            linear
plot      asin(db(va)/50)   acos(db(va)/50)        linear
plot      tan(db(va))       atan(db(va))/100       linear
plot      norm(db(va))      pos(db(va))            linear
plot      deriv(db(va))*100 db(va)                 linear
plot      db(va)            integral(db(va))/1000  linear
plot      real(va)          imag((va))  j(va)          linear
let       va_interp =       interpolate(real(va)) +    j(interpolate(imag(va)))
plot      real(va_interp)   imag((va_interp))          linear
print     va[1] real(va[1]) imag (va[1]) j(va[1])    real(j(va[1]))
let       max_ph =          maximum(ph(va))
echo     "phase_Max  =      $&max_ph"
let       min_ph =          minimum(ph(va))
echo     "phase_Min  =      $&min_ph"
let       max_uw_ph =       maximum(unwrap(ph(va)))
echo     "UnWrap_phase_Max= $&max_uw_ph"
let       min_uw_ph =       minimum(unwrap(ph(va)))
echo     "UnWrap_phase_Min= $&min_uw_ph"
let       ave_uw_ph =       mean(unwrap(ph(va)))
echo     "UnWrap_phase_Ave= $&ave_uw_ph"
let       number =          length(va)
print     number
plot      unitvec($&number) linear
plot      vector($&number)  linear
compose   rtime  start = 0  stop = $&number            step =1
compose   rtime2 start = 0  stop = $&number            step =1
let       i = 0
repeat    $&number
let       i = i +1
let       rtime[i]  = rnd(4)
let       rtime2[i] = rnd(8)
end
plot      rtime2 rtime linear

.endc
*=========CRYSTAL_SubCircuit================================================
.SUBCKT   XCRYST VA     VB
 RS       VA     VCS    340
 CS       VCS    VLS    7f
 LS       VLS    VB     3.5
 CP       VA     VB     3pf
.ENDS     XCRYST
.end
```

7.12.10_10.31AM

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