Aliasing

We want to show that sampling a relatively high frequency function gives exactly the same results as sampling a lower frequency function, if the high frequency is higher than the Nyquist frequency $\omega_{\rm Ny} = \frac{\pi}{\Delta t}$.

First, we import a Matlab-like set of commands,

In [44]:
```
from pylab import *
```

I want to use a period of 20 s and 50 sampling points (hence $\Delta t = h = 0.4$ s)

In [45]:
```
Tp = 20.0
N  = 50
step = Tp/N
```

I compute the fundamental frequency of the Fourier Series associated wit the period and the corresponding Nyquist frequency

In [46]:
```
dw = 2*pi/Tp
wny  = dw*N/2
```

For comparison, we want to plot our functions also with a high sampling rate, so that we create the illusion of plotting a continuous function, so we say

In [47]:
```
M =2000
```

The function linspace generates a vector with a start and a stop value, with *that many* points in it (remember that the number of intervals is the number of points *minus* one),

In [48]:
```
t_n=linspace(0.0,Tp,N+1)
t_m=linspace(0.0,Tp,M+1)
t_m, t_n
```

Out[48]:
```
(array([  0.00000000e+00,   1.00000000e-02,   2.00000000e-02, ...,
          1.99800000e+01,   1.99900000e+01,   2.00000000e+01]),
 array([  0. ,   0.4,   0.8,   1.2,   1.6,   2. ,   2.4,   2.8,   3.2,
          3.6,   4. ,   4.4,   4.8,   5.2,   5.6,   6. ,   6.4,   6.8,
          7.2,   7.6,   8. ,   8.4,   8.8,   9.2,   9.6,  10. ,  10.4,
         10.8,  11.2,  11.6,  12. ,  12.4,  12.8,  13.2,  13.6,  14. ,
         14.4,  14.8,  15.2,  15.6,  16. ,  16.4,  16.8,  17.2,  17.6,
         18. ,  18.4,  18.8,  19.2,  19.6,  20. ]))
```

The functions that we want to sample and plot are

$$\cos(+31\Delta\omega t) \quad \text{and} \quad \cos(-19\Delta\omega t).$$

Note that $31 - N = -19$.

In the following, hs and ls mean high and low sampling frequency, while hf and lf mean high cosine frequency and low one.

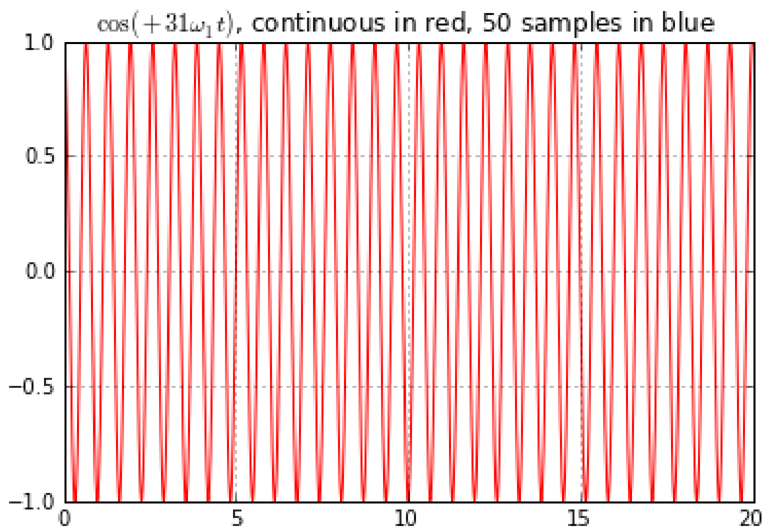
In [49]:
```
c_hs_hf = cos(+31*dw*t_m)
```

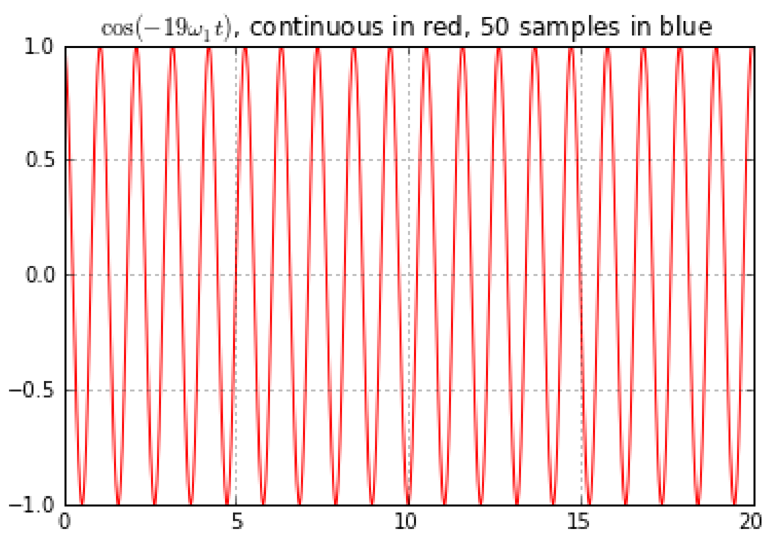
```
c_hs_lf = cos(-19*dw*t_m)

c_ls_hf = cos(+31*dw*t_n)
c_ls_lf = cos(-19*dw*t_n)
```

First, we plot the highly sampled fuctions

In [50]:
```
figure(1);plot(t_m,c_hs_hf,'-r')
grid()
title(r'$\cos(+31\omega_1t)$, continuous in red, 50 samples in blue')
figure(2);plot(t_m,c_hs_lf,'-r')
grid()
title(r'$\cos(-19\omega_1 t)$, continuous in red, 50 samples in blue')
```

Out[50]: <matplotlib.text.Text at 0x7f18d1ad0410>



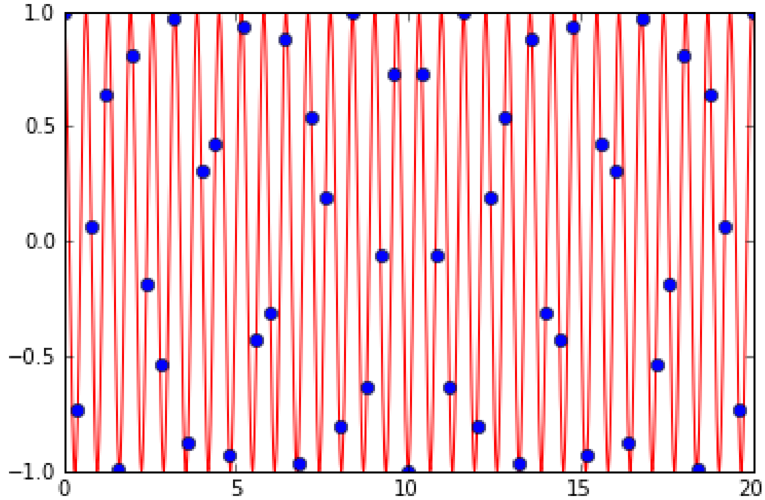


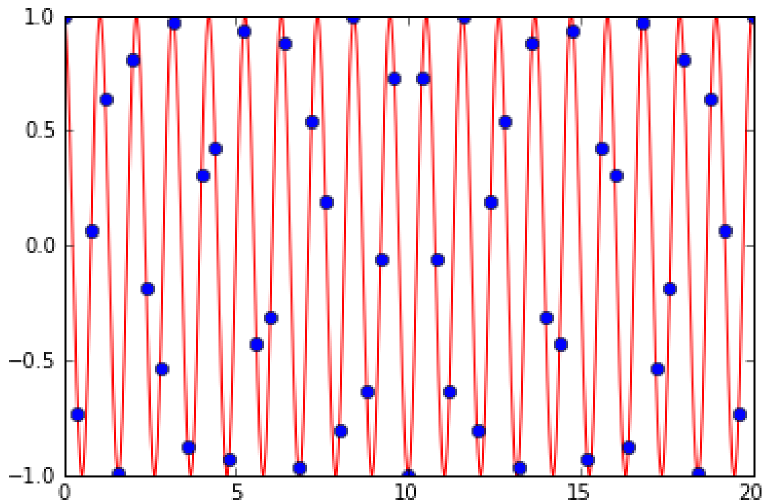
It is apparent that the two functions are different.

Then, we place a blue dot for every sample that was taken with a low sampling.

```
In [51]: figure(1) ; plot(t_m,c_hs_hf,'-r',t_n,c_ls_hf,'ob')
         figure(2) ; plot(t_m,c_hs_lf,'-r',t_n,c_ls_lf,'ob')
```

```
Out[51]: [<matplotlib.lines.Line2D at 0x7f18d1a9dcd0>,
          <matplotlib.lines.Line2D at 0x7f18d1a9d8d0>]
```





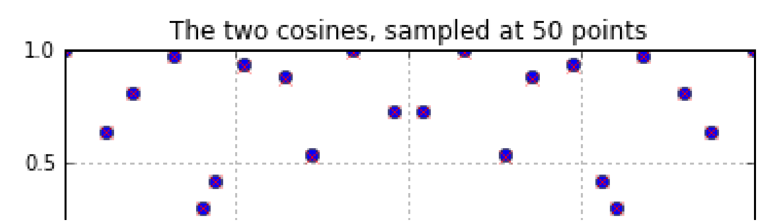
If you look at the patterns of the dots they seem, at least, very similar.

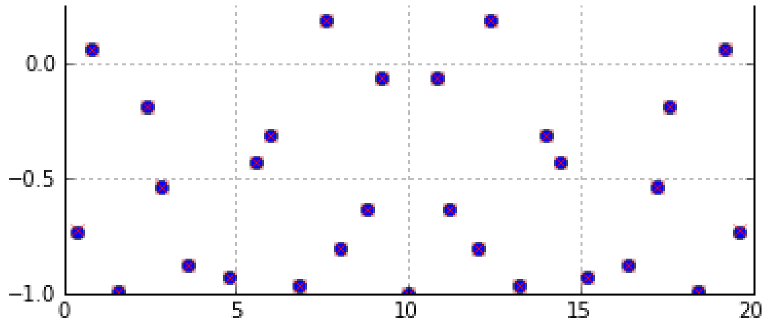
What happens is aliasing!

It's time to plot *only* the low sampling rate functions, introducing a little shift to make the plot clearer:

```
In [52]: figure(3) ; grid()
         title('The two cosines, sampled at 50 points')
         figure(3) ; plot(t_n,c_ls_hf,'ob',t_n,c_ls_lf*0.998,'xr')
```

```
Out[52]: [<matplotlib.lines.Line2D at 0x7f18d15c67d0>,
          <matplotlib.lines.Line2D at 0x7f18d1b840d0>]
```

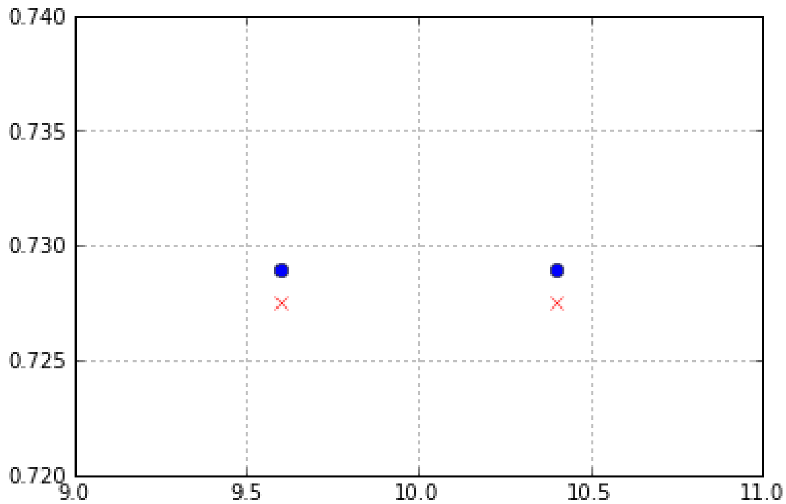




You cannot see the red crosses because they're too close to the blue dots... let's try zooming into a detail (remember, red crossed are scaled at 998 per mil).

In [53]:
```
axis([9.,11.,0.72,0.74]); grid()
plot(t_n,c_ls_hf,'ob',t_n,c_ls_lf*0.998,'xr')
```

Out[53]:
```
[<matplotlib.lines.Line2D at 0x7f18d1ba0810>,
 <matplotlib.lines.Line2D at 0x7f18d238e610>]
```



In [53]: