Data Analysis of 8422[2f3] Material

This file extracts the $\Delta\epsilon$ value from the fitted model I use on my data. The inverse of this value is plotted against the temperature for each spectrum at that given temperature. From this, we fit a quadratic formula to determine the value of the susceptibility critical exponent, γ.

The cell below defines a function used to extract the data. It returns the parameter values determined by our model and scipy's "curve_fit" function, a weighted least-squares fit.

In [313]:
```
%pylab inline
from mpl_toolkits.mplot3d import Axes3D
import scipy.optimize as spo

def data_extract(filenames, guess, bool_to_guess, bool_to_print, bool_to_plot):

    files = [filenames[0]]

    w = np.loadtxt('Data04262013/weights.txt', dtype=float,delimiter='\t')
    w = w.T
  #  w = w[100:]
    test = np.loadtxt('Data04262013/'+files[0], dtype=float,delimiter='\t')

    m = test.T[0]

    f = test.T[1]

    a = test.T[4]


    """ Combine the data into a 3D array """
    magn = m
    freq = f
    angl = a

    """ Put data in terms of the dielectric constant """
    A = .01**2;
    eps0 = 8.854187e-12;
    d = 5e-6;
    omega = 2*np.pi*freq;

    G = cos(angl)/(magn);
    C = -sin(angl)/(magn);

    C0 = (A*eps0)/(d);

    """ e2 is the epsilon double primed or the imaginery part of the dielectric
```

```
data
        e1 is the epsilong primed or real part of the dielectric data """

    e2 = G/(omega*C0);
    e1 = C/(omega*C0);

    """ GUESSED VALUES for the fit of the primary peak in the dielectric data. """
    E_infinity = guess[0];
    Delta_E = guess[1];
    Relaxation_frequency = guess[2];
    Beta = guess[3];
    G_low_frequency = guess[4];

    """ Defining the data as x,y """
    x = freq;
    y = e2;

    """ Model Function to fit our data """
    def func(x, Einf,delE,relaxFreq,beta,g):
        z = (Einf + (delE/(1 + (1j*(x/relaxFreq))**(beta))) + g)
        return -(z.imag)

    dummy = func(x,E_infinity, Delta_E, Relaxation_frequency, Beta,
G_low_frequency)

    if bool_to_guess == True:
        plt.figure(1)
        semilogx(freq,dummy,'-r')
        semilogx(freq,e2,'.')
        xlabel('Frequency')
        ylabel('$\epsilon$')
        title('Dielectric Spectrum')
        grid()

    popt, pcov = spo.curve_fit(func,x,y,p0=[E_infinity, Delta_E,
Relaxation_frequency, Beta, G_low_frequency], sigma=w)

    fittedmodel = func(x,popt[0],popt[1],popt[2],popt[3],popt[4])

    if bool_to_print == True:
        def fit_results(param,values):
            for i in range(0,5,1):
                print param[i] + str(values[i])

        """ Print out the values of the fit """
        params = ["\nRESULTS FOR PRIMARY PEAK: \n\nE_infinity: ", "Delta_E: ",
"Relaxation_frequency: ", "Beta: ", "G_low_frequency: "];
        fit_results(params,popt)

    if bool_to_plot == True:
        """ Plot the fitted model """
```

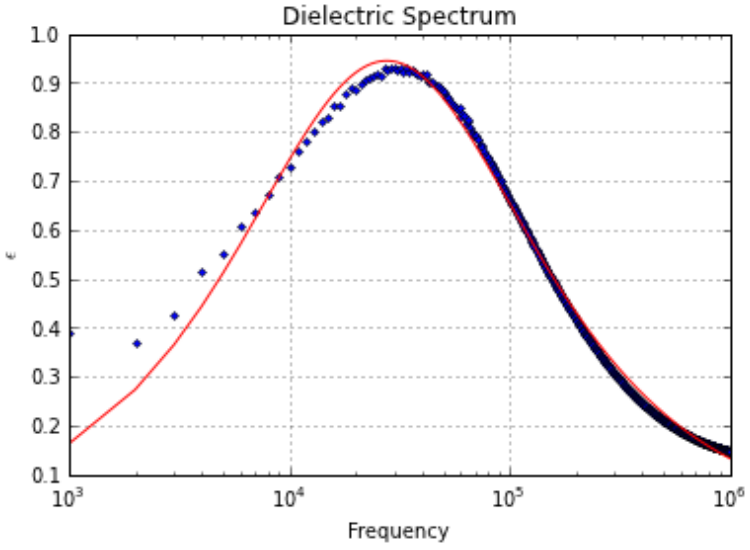
```
        plt.figure(2)
        semilogx(freq,e2,'.')
        semilogx(freq,fittedmodel,'r')
        xlabel('Frequency')
        ylabel('$\epsilon$')
        title('Dielectric Spectrum')
        #ylim((0,1))
        grid()

    return popt, fittedmodel
```

```
Welcome to pylab, a matplotlib-based Python environment [backend:
module://IPython.kernel.zmq.pylab.backend_inline].
For more information, type 'help(pylab)'.
```

Call the function above on each spectrum and return the fit values.

In [314]:
```
""" GUESSED VALUES for the fit of the primary peak in the dielectric data. """
E_infinity = 0;
Delta_E = 1;
Relaxation_frequency = 1*10**4.5;
Beta = .2;
G_low_frequency = 0;

guess = [E_infinity, Delta_E, Relaxation_frequency, Beta, G_low_frequency]

popt1, fittedmodel1 = data_extract(['test25'], guess, False, True, True)
```

```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 2.58441980306
Relaxation_frequency: 27717.1947545
Beta: 0.803753072946
G_low_frequency: 0.0
```

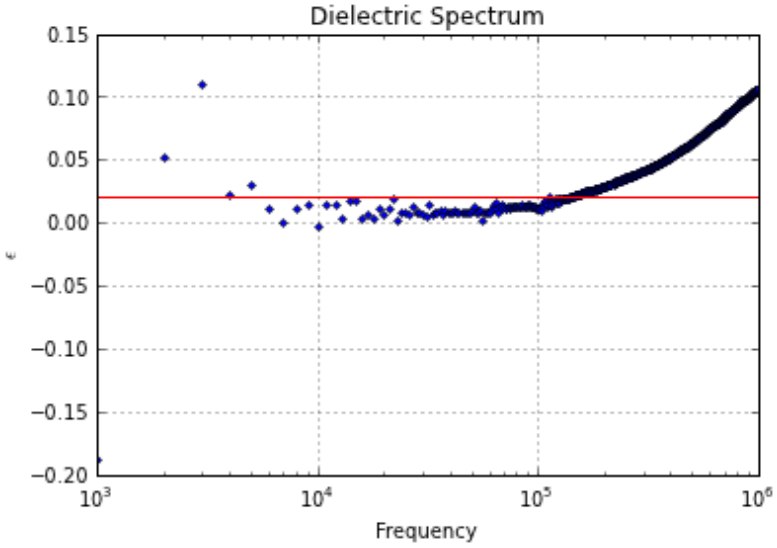


In [315]:
```
popt2, fittedmodel2 = data_extract(['test02'], guess, False, True, True)
```

```
RESULTS FOR PRIMARY PEAK:
```

RESULTS FOR PRIMARY PEAK:

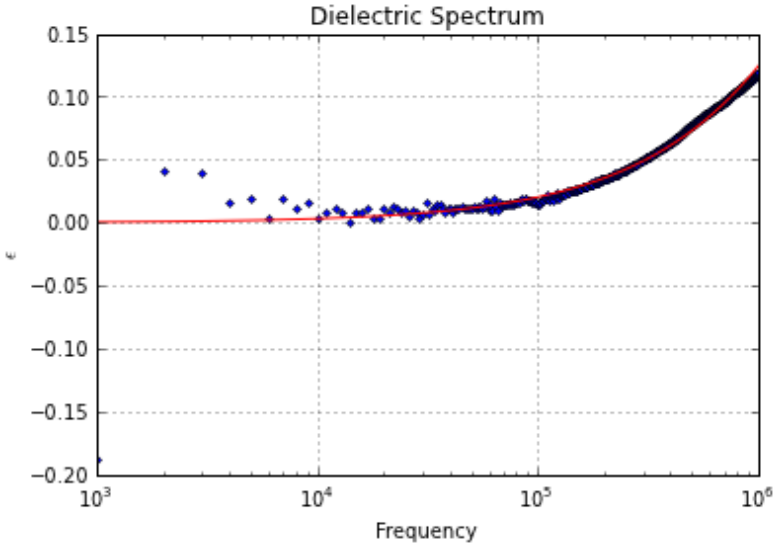
E_infinity: 0.0
Delta_E: 118.998295987
Relaxation_frequency: 46.9818041045
Beta: 0.000443506948068
G_low_frequency: 0.0



In [316]:
```
popt3, fittedmodel3 = data_extract(['test03'], guess, False, True, True)
```

RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 1.31195187701
Relaxation_frequency: 15773513.5511
Beta: 0.813380336756
G_low_frequency: 0.0

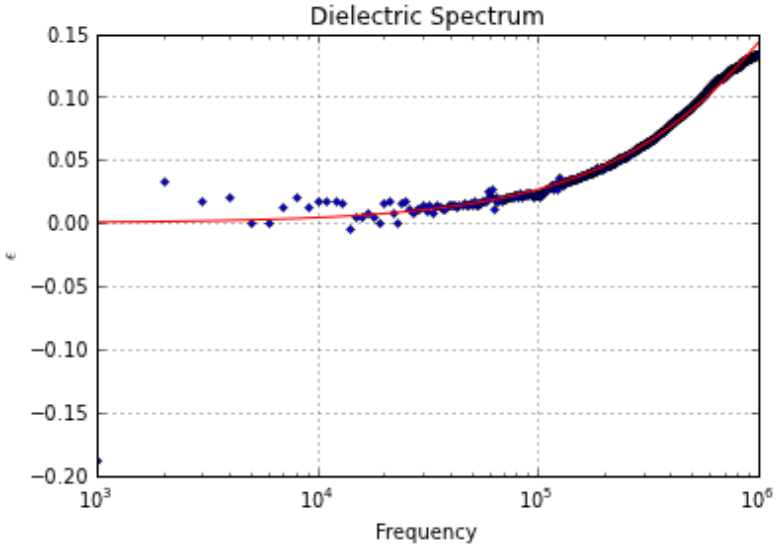


In [317]:
```
popt4, fittedmodel4 = data_extract(['test04'], guess, False, True, True)
```

RESULTS FOR PRIMARY PEAK:

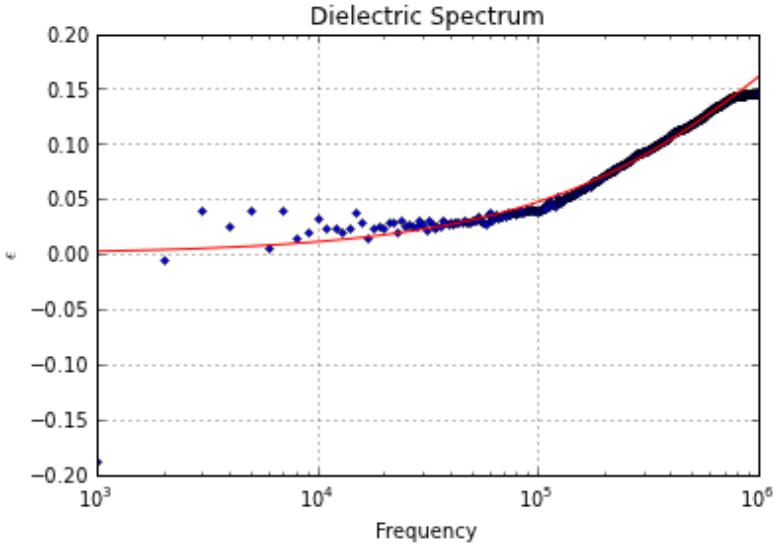
E_infinity: 0.0

```
Delta_E: 0.730048608396
Relaxation_frequency: 5635316.59632
Beta: 0.80281311
G_low_frequency: 0.0
```



In [318]:
```
popt5, fittedmodel5 = data_extract(['test05'], guess, False, True, True)
```

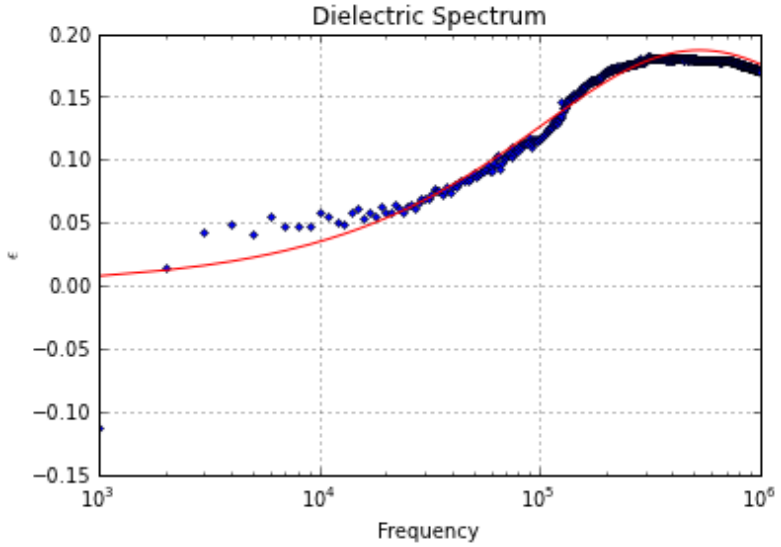
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 0.873121630824
Relaxation_frequency: 6227005.35118
Beta: 0.649344824043
G_low_frequency: 0.0
```



In [319]:
```
popt6, fittedmodel6 = data_extract(['test06'], guess, False, True, True)
```

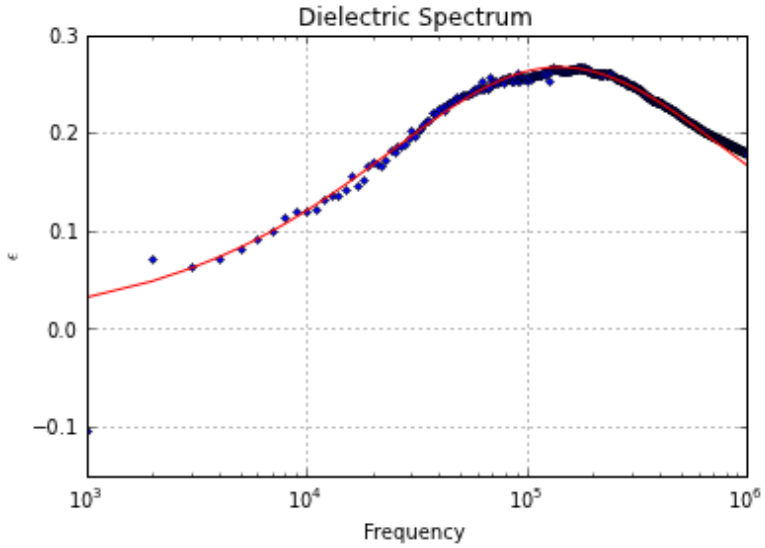
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 0.632729189059
Relaxation_frequency: 530954.625369
```

```
Beta: 0.679510773013
G_low_frequency: 0.0
```



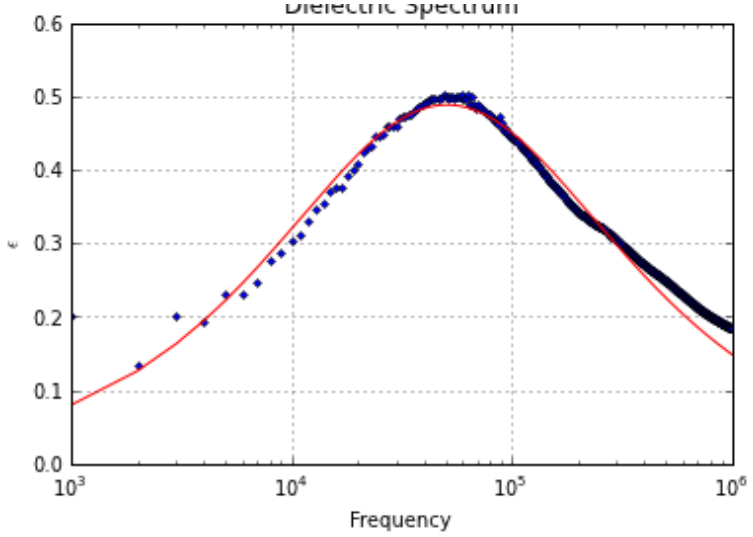
In [320]:
```
popt7, fittedmodel7 = data_extract(['test07'], guess, False, True, True)
```

```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 0.957961341696
Relaxation_frequency: 141030.038655
Beta: 0.647194272047
G_low_frequency: 0.0
```



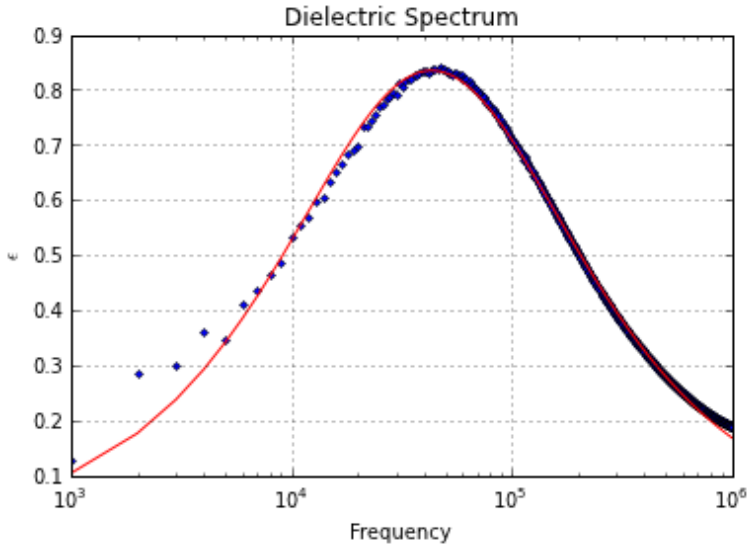
In [321]:
```
popt8, fittedmodel8 = data_extract(['test08'], guess, False, True, True)
```

```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 1.55022063362
Relaxation_frequency: 50916.7508087
Beta: 0.715710277668
G_low_frequency: 0.0
```



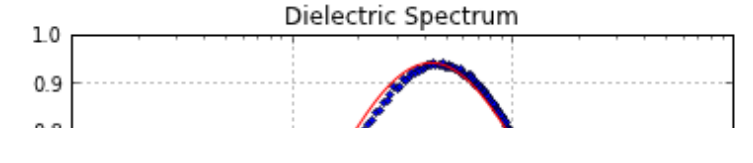
In [322]:
```
popt9, fittedmodel9 = data_extract(['test09'], guess, False, True, True)
```

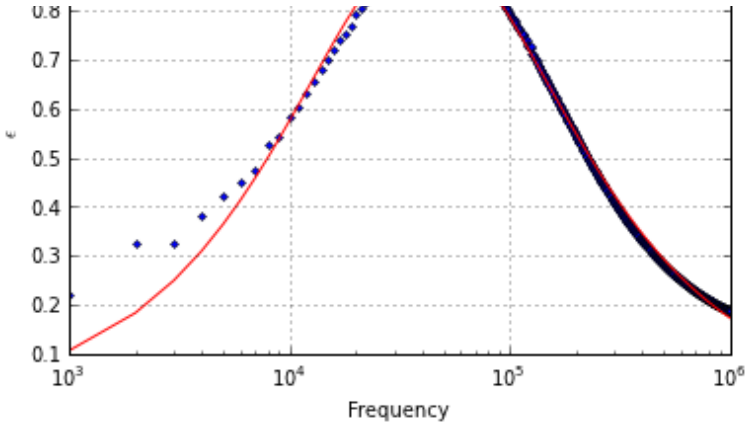
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 2.30235861059
Relaxation_frequency: 43386.87195
Beta: 0.798974165756
G_low_frequency: 0.0
```



In [323]:
```
popt10, fittedmodel10 = data_extract(['test10'], guess, False, True, True)
```

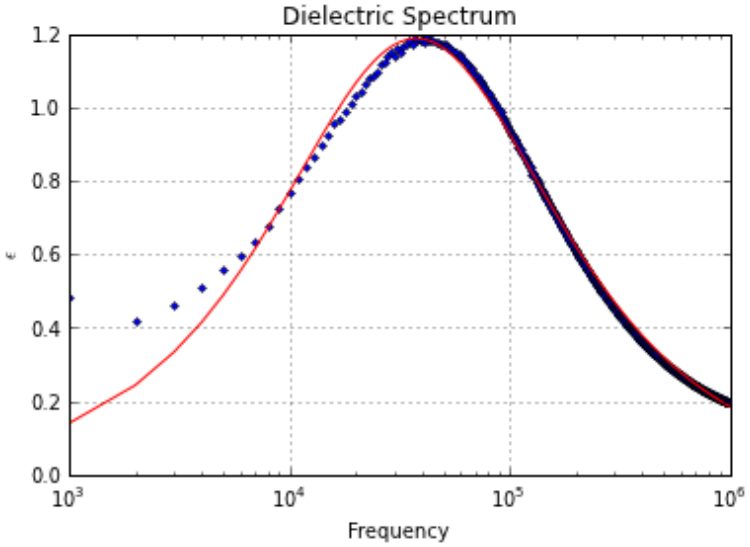
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 2.49590947114
Relaxation_frequency: 43119.6797734
Beta: 0.822190463857
G_low_frequency: 0.0
```





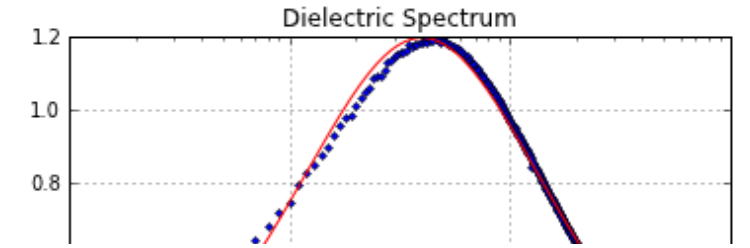
In [324]:
```
popt11, fittedmodel11 = data_extract(['test11'], guess, False, True, True)
```

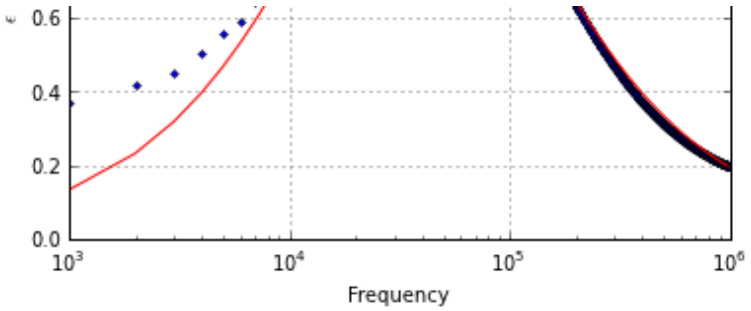
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 3.07963435544
Relaxation_frequency: 37701.9062584
Beta: 0.836152309831
G_low_frequency: 0.0
```



In [325]:
```
popt12, fittedmodel12 = data_extract(['test12'], guess, False, True, True)
```

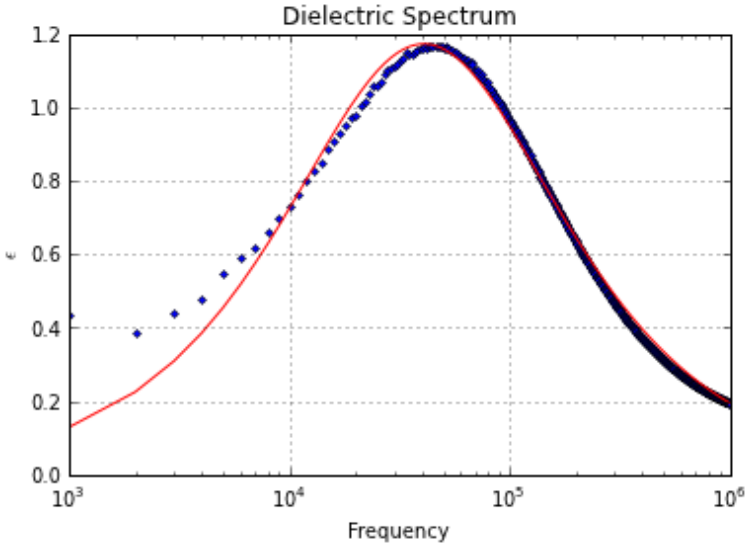
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 3.0883271675
Relaxation_frequency: 39969.5393543
Beta: 0.838111524492
G_low_frequency: 0.0
```





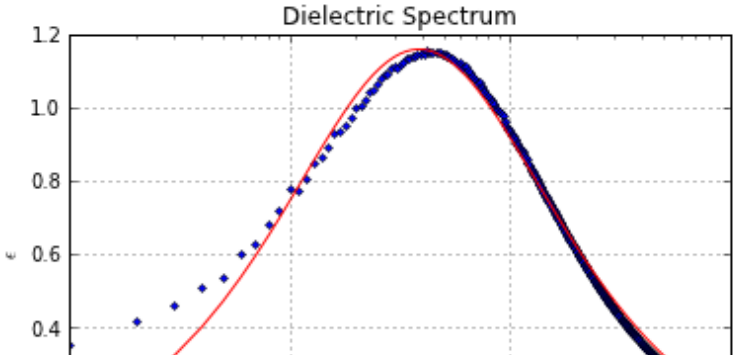
In [326]:
```
popt13, fittedmodel13 = data_extract(['test13'], guess, False, True, True)
```

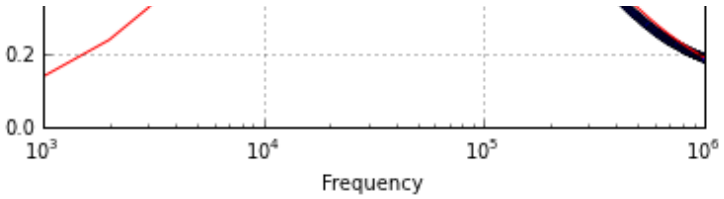
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 3.03728312763
Relaxation_frequency: 40805.9221535
Beta: 0.836971142844
G_low_frequency: 0.0
```



In [327]:
```
popt14, fittedmodel14 = data_extract(['test14'], guess, False, True, True)
```

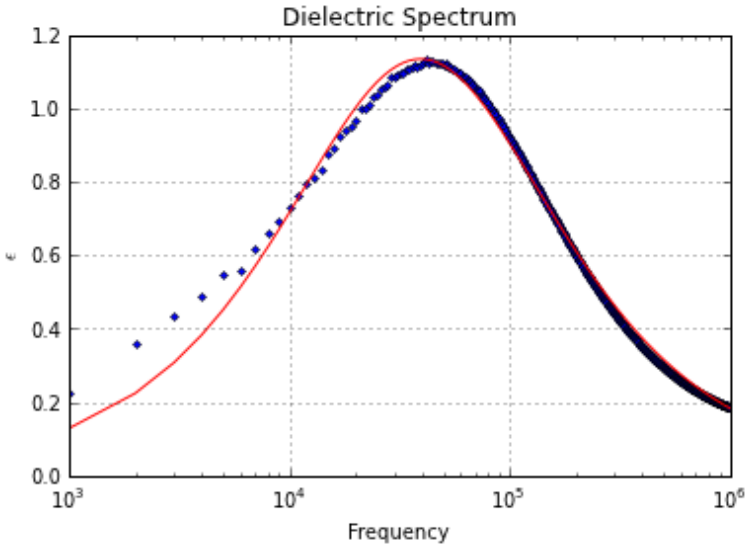
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 3.02688569403
Relaxation_frequency: 38808.9967403
Beta: 0.83131101337
G_low_frequency: 0.0
```





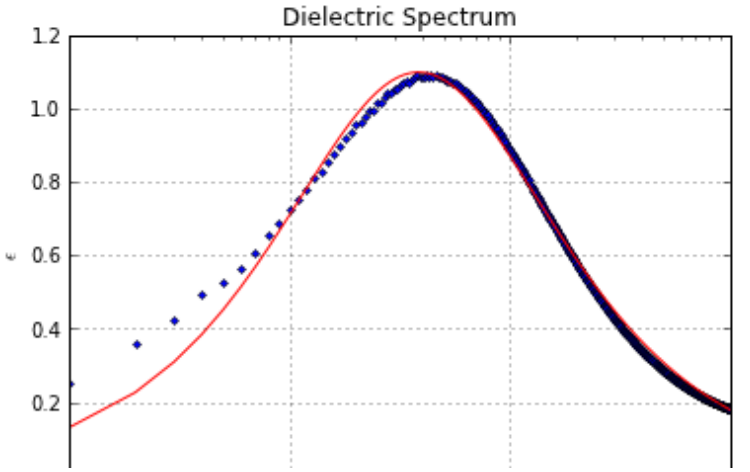
In [328]:
```
popt15, fittedmodel15 = data_extract(['test15'], guess, False, True, True)
```

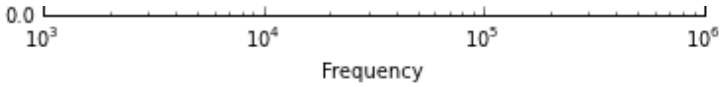
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 2.94107477538
Relaxation_frequency: 39452.0305128
Beta: 0.836327682136
G_low_frequency: 0.0
```



In [329]:
```
popt16, fittedmodel16 = data_extract(['test16'], guess, False, True, True)
```

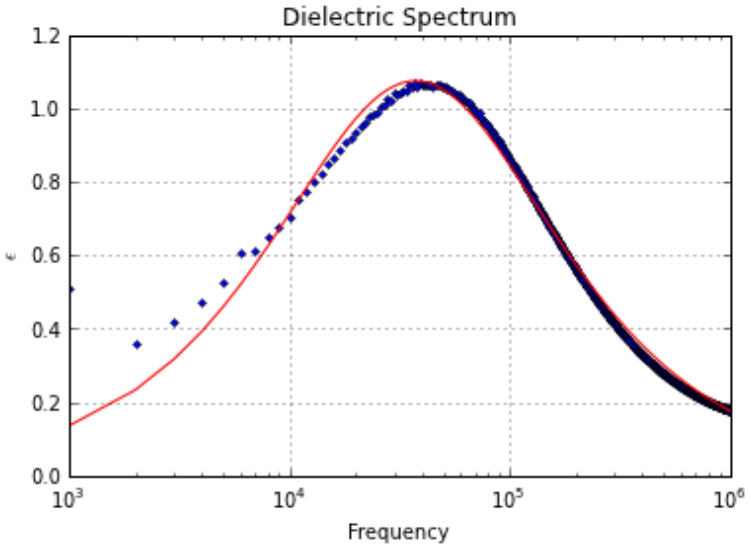
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 2.8790302444
Relaxation_frequency: 38606.8371397
Beta: 0.82951751516
G_low_frequency: 0.0
```





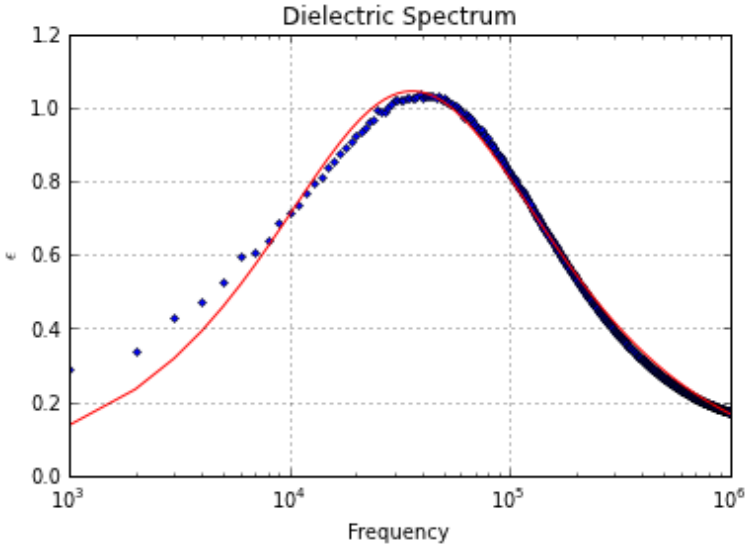
In [330]:
```
popt17, fittedmodel17 = data_extract(['test17'], guess, False, True, True)
```

```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 2.85049641429
Relaxation_frequency: 37419.2982883
Beta: 0.822363632645
G_low_frequency: 0.0
```



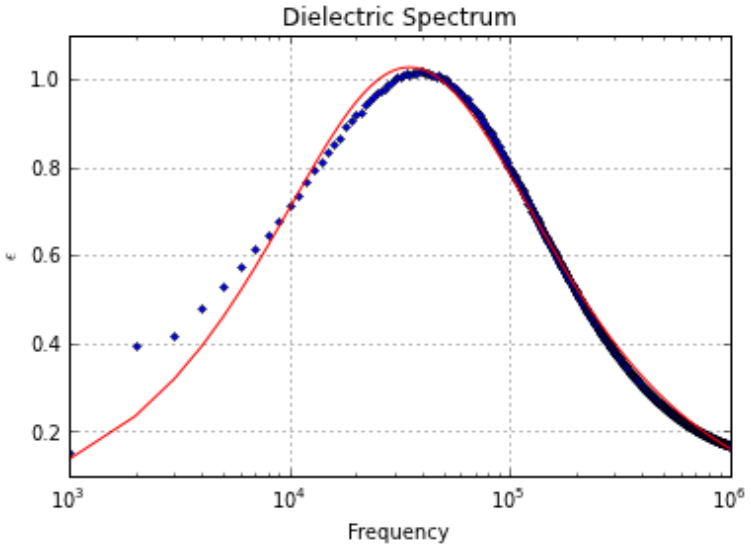
In [331]:
```
popt18, fittedmodel18 = data_extract(['test18'], guess, False, True, True)
```

```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 2.77644304289
Relaxation_frequency: 36095.2876941
Beta: 0.820978897872
G_low_frequency: 0.0
```



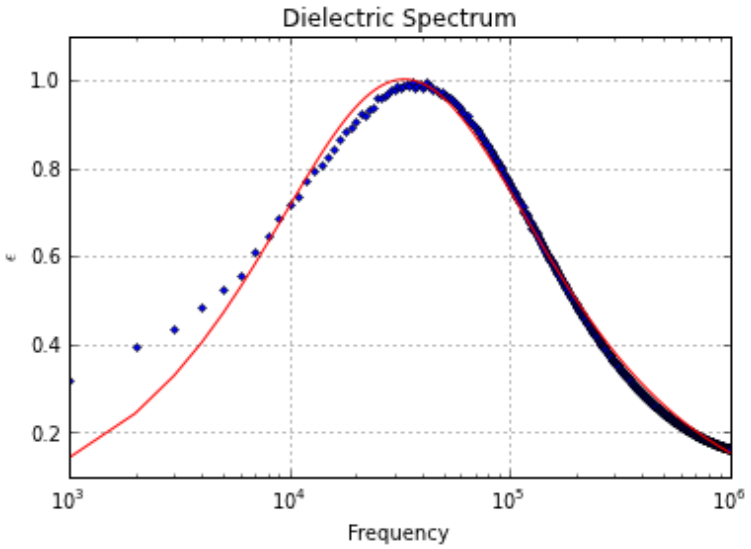
In [332]:
```
popt19, fittedmodel19 = data_extract(['test19'], guess, False, True, True)
```

```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 2.72838975027
Relaxation_frequency: 35120.8270513
Beta: 0.821059486263
G_low_frequency: 0.0
```



In [333]:
```
popt20, fittedmodel20 = data_extract(['test20'], guess, False, True, True)
```

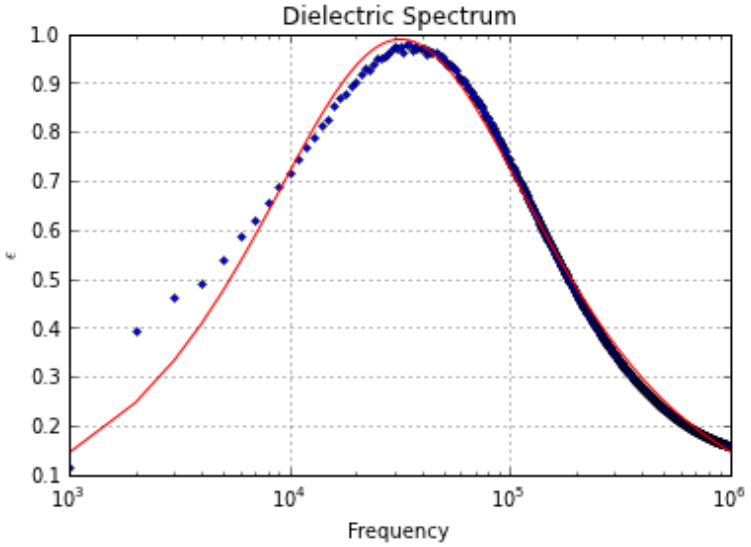
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 2.68612833537
Relaxation_frequency: 33276.7804138
Beta: 0.815503859188
G_low_frequency: 0.0
```



In [334]:
```
popt21, fittedmodel21 = data_extract(['test21'], guess, False, True, True)
```

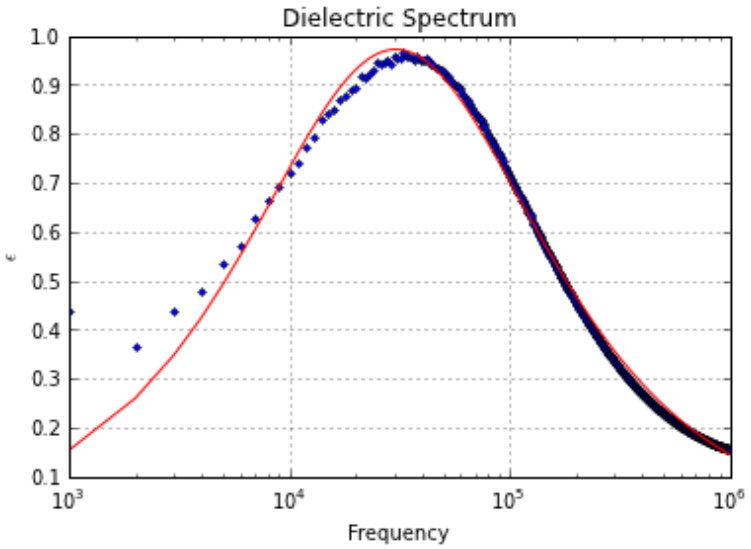
```
RESULTS FOR PRIMARY PEAK:
```

```
E_infinity: 0.0
Delta_E: 2.64880270719
Relaxation_frequency: 32055.3324892
Beta: 0.816308640105
G_low_frequency: 0.0
```



In [335]:
```
popt22, fittedmodel22 = data_extract(['test22'], guess, False, True, True)
```

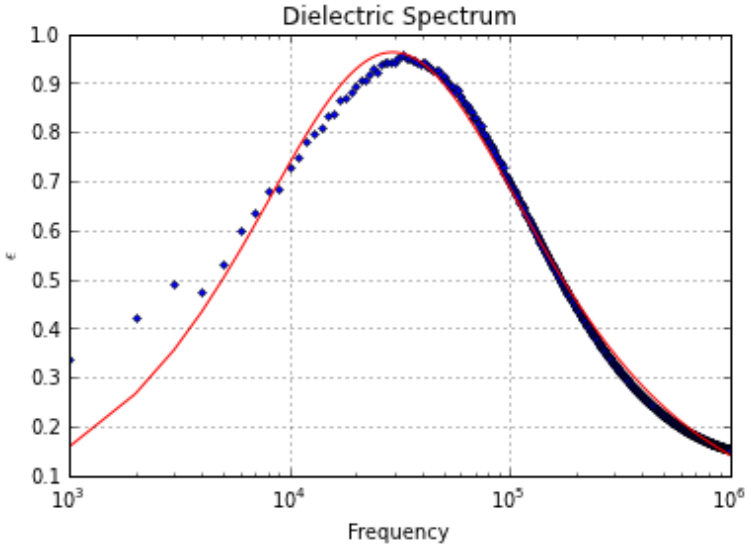
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 2.64442411481
Relaxation_frequency: 30452.0441881
Beta: 0.807145351461
G_low_frequency: 0.0
```



In [336]:
```
popt23, fittedmodel23 = data_extract(['test23'], guess, False, True, True)
```

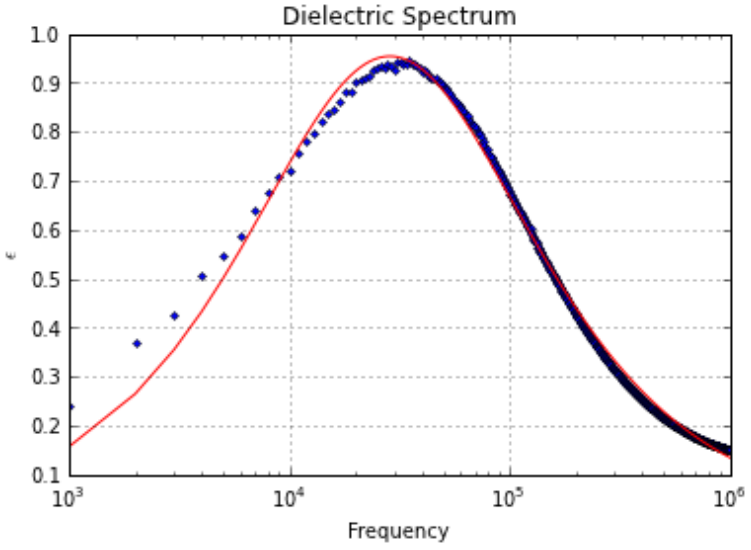
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
```

```
Delta_E: 2.63093306154
Relaxation_frequency: 29490.1770938
Beta: 0.804183920622
G_low_frequency: 0.0
```



In [337]:
```
popt24, fittedmodel24 = data_extract(['test24'], guess, False, True, True)
```

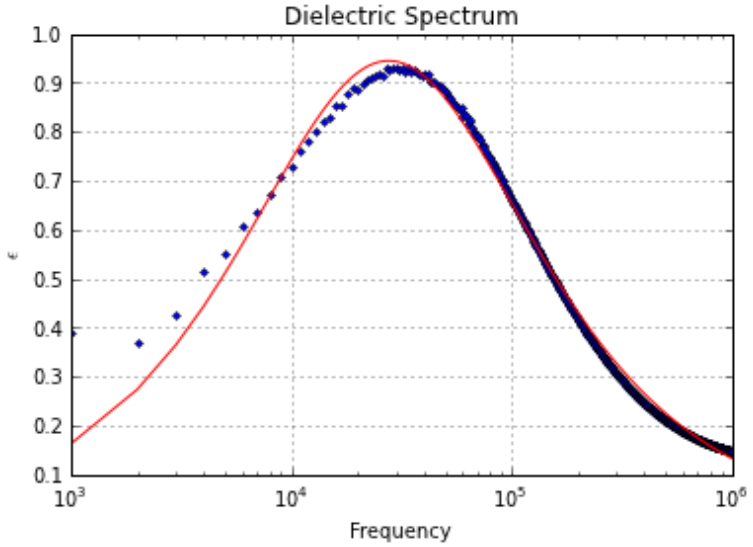
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 2.58637551532
Relaxation_frequency: 28665.0039269
Beta: 0.8092442889
G_low_frequency: 0.0
```



In [338]:
```
popt25, fittedmodel25 = data_extract(['test25'], guess, False, True, True)
```

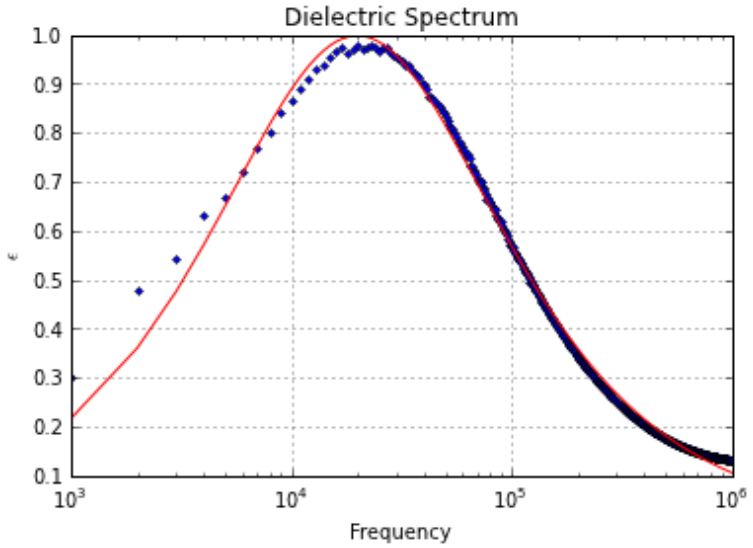
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 2.58441980306
Relaxation_frequency: 27717.1947545
```

```
Beta: 0.803753072946
G_low_frequency: 0.0
```



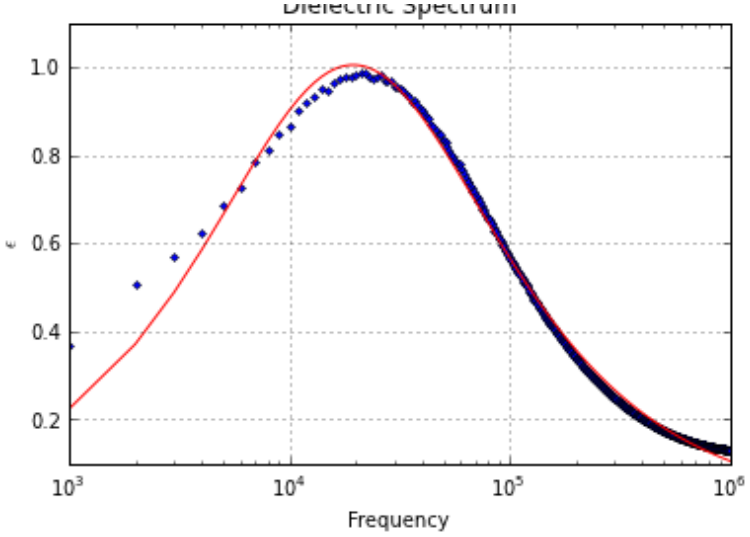
In [339]:
```
popt26, fittedmodel26 = data_extract(['test26'], guess, False, True, True)
```

```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 2.70559435333
Relaxation_frequency: 19942.3162844
Beta: 0.809259522622
G_low_frequency: 0.0
```



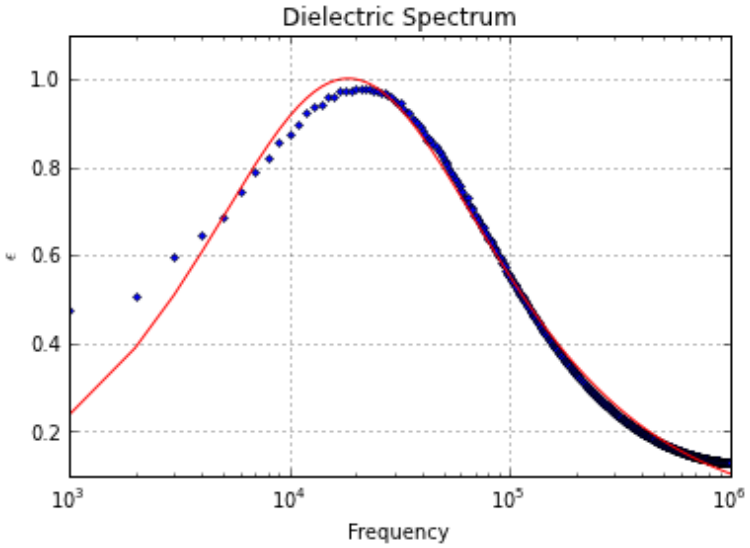
In [340]:
```
popt27, fittedmodel27 = data_extract(['test27'], guess, False, True, True)
```

```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 2.73452338911
Relaxation_frequency: 19507.3717494
Beta: 0.806585263929
G_low_frequency: 0.0
```



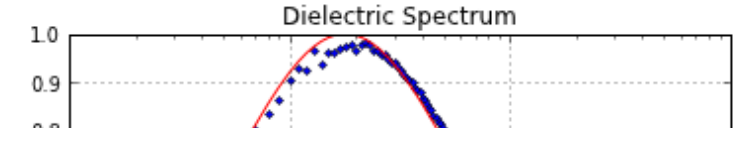
In [341]:
```
popt28, fittedmodel28 = data_extract(['test28'], guess, False, True, True)
```

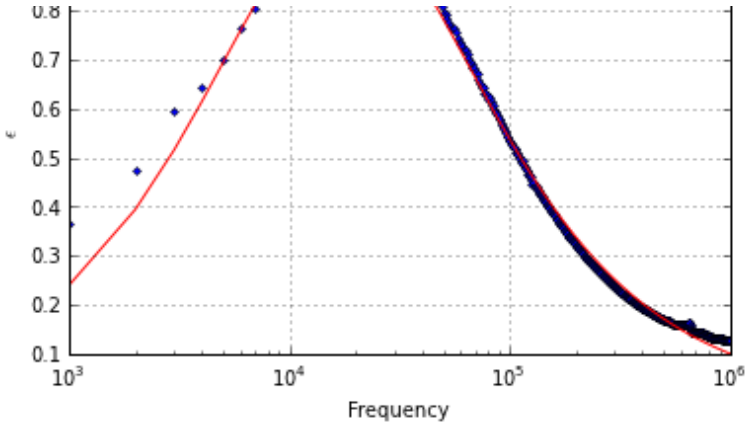
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 2.75321645729
Relaxation_frequency: 18513.3274851
Beta: 0.800223072693
G_low_frequency: 0.0
```



In [342]:
```
popt29, fittedmodel29 = data_extract(['test29'], guess, False, True, True)
```

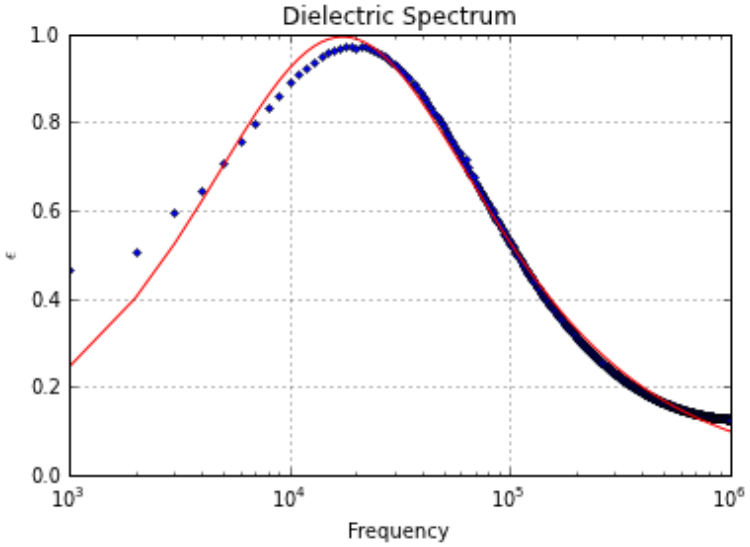
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 2.73203474441
Relaxation_frequency: 17924.3305289
Beta: 0.804260598216
G_low_frequency: 0.0
```





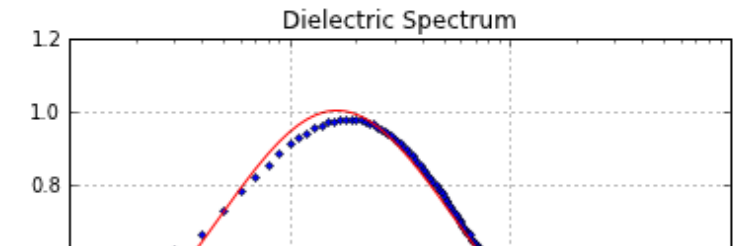
In [343]:
```
popt30, fittedmodel30 = data_extract(['test30'], guess, False, True, True)
```

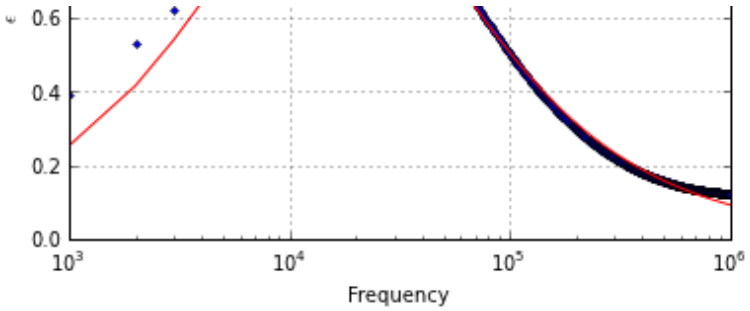
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 2.72273975919
Relaxation_frequency: 17486.5629194
Beta: 0.802354031285
G_low_frequency: 0.0
```



In [344]:
```
popt31, fittedmodel31 = data_extract(['test31'], guess, False, True, True)
```

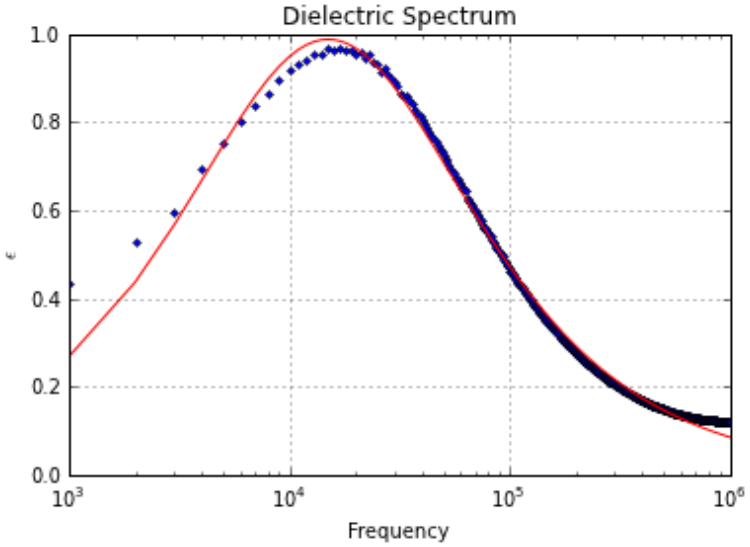
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 2.72215514045
Relaxation_frequency: 16428.8027229
Beta: 0.807773340741
G_low_frequency: 0.0
```





In [345]:
```
popt32, fittedmodel32 = data_extract(['test32'], guess, False, True, True)
```

```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 2.68860937209
Relaxation_frequency: 15007.0627274
Beta: 0.80646496647
G_low_frequency: 0.0
```



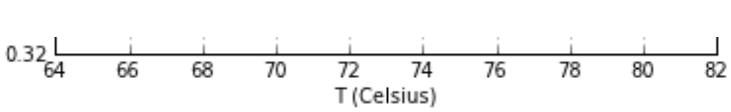
In [366]:
```
#epsilon =
[popt1[1],popt2[1],popt3[1],popt4[1],popt5[1],popt6[1],popt7[1],popt8[1],
popt9[1], popt10[1],
epsilon = [popt13[1],popt14[1],popt15[1],popt16[1],popt17[1],popt18[1],
           popt19[1],popt20[1],popt21[1],popt22[1],popt23[1]]
#,popt24[1],popt25[1]]
#,popt26[1],popt27[1],popt28[1],
#            popt29[1],popt30[1],popt31[1],popt32[1]]

#reson = [popt1[2],popt2[2],popt3[2],popt4[2],popt5[2],popt6[2],popt7[2],popt8[2],
popt9[2], popt10[2],
reson = [popt13[2],popt14[2],popt15[2],popt16[2],popt17[2],popt18[2],
           popt19[2],popt20[2],popt21[2],popt22[2],popt23[2]]
#,popt24[2],popt25[2]]
#,popt26[2],popt27[2],popt28[2],
#            popt29[2],popt30[2],popt31[2],popt32[2]]

#temperature = [100.1, 98.9, 97.9, 96.9, 96, 94.6, 93.5, 92.4, 91, 90.1,
```

```
temperature =  [82, 80, 77.9,
               75.9, 74, 71.9, 70, 68, 67, 66, 65]
#, 64, 63]
#, 61, 60.1, 59, 58, 57, 56, 55];

T = temperature
for i in range(0,len(epsilon),1):
    epsilon[i]=epsilon[i]**(-1)
    T[i] = T[i]
    reson[i] = reson[i]/1000
```

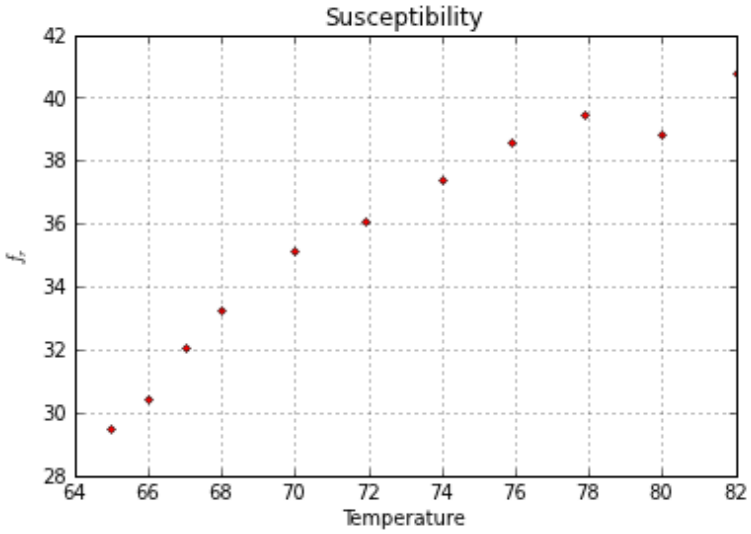
Determining the susceptibility critical exponent of the liquid crystal by fitting to the power law equation:
$(\Delta\epsilon)^{-1} = At^{\gamma} + B$

In [369]:
```
def susc_func(x,A,B,gamma):
    s = A*np.power(x,gamma) + B
    return s

susc2, l2 = spo.curve_fit(susc_func, temperature ,epsilon, p0=[-.0033,.58,1],
maxfev=10000)
print "The value of B is: " + str(susc2[1])
print "The value of m is: " + str(susc2[0])
print "The value of gamma is: " + str(susc2[2])

fittedmodel = susc_func(temperature, susc2[0],susc2[1],susc2[2])
#,susc2[2])
#guess = susc_func(T,-.0033,.58,1)

plt.figure(2)
plot(T,epsilon,'r.')
#plot(T,guess,'b')
plot(T,fittedmodel,'b')
xlabel('T (Celsius)')
ylabel('1/($\epsilon$)')
title('Susceptibility (8422[2F3])')
grid()
```

```
The value of B is: 0.603348063493
The value of m is: -0.00413386084396
The value of gamma is: 0.953729413215
```



In [370]:
```
plt.figure(1)
plot(T,reson,'r.')
xlabel('Temperature')
ylabel('$f_{r}$')
title('Susceptibility')
grid()
```



In [368]:
```
eps0 = 8.854187e-12
mP = 185*10**-9/.01**2
m = -1*susc2[0]
alpha = (m*mP**2)/eps0/1000
print alpha
```

```
1.59790376445
```

Back to top