Dielectric Behavior of de Vries Liquid Crystals

Zach Sailer
Bachelors of Science in Physics
California Polytechnic State University
San Luis Obispo, CA

https://github.com/Zsailer/calpolythesis

July 26, 2013

Abstract

In this experiment, we use the method of dielectric spectroscopy with two De Vries liquid crystal materials, 8422[2F3] and TSiKN65, to determine the type of phase transition that occurs between the SmA-SmC phases and quantify their susceptibility in an external perturbation. In the smectic-C, we find resonance peaks which indicate an increase in the crystal's susceptibility to align with an alternating electric field. We fit our data to the Debye dispersion model and extract the relaxation strength of the crystal at different temperature. This data is used to make a Curie-Weiss plot for each sample, where the critical exponent γ is determined. Each material exhibits second order phase transitions ($\gamma \approx 1$) that can be modeled by a Landau free energy equation. This model quantifies the material's susceptibility with the constant value α. Smaller values of $|\alpha|$ indicate higher susceptibility for the material to respond to external perturbations. We determine α to be 5.255^{-1}K for the TSiKN65 compound and -1.60 $K^{-1}$for the 8422[2F3] compound.

1 Introduction

The most familiar states of matter are the solid, liquid, gas, and plasma phases. For many centuries, these were the only known states; however, in 1888 an Austrian botanical physiologist, Friedrich Reinitzer, discovered a second melting point between the solid and liquid phase in a cholesterol compound [13]. This intermediate, unique phase would later be known as the liquid crystalline phase. It has become an important concept in understanding of biological systems and development of electro-optical sciences. Such substances "flow" much like a liquid and maintain positional order like the crystalline phase. Today, the physics of these materials are used to study cellular bi-layer membranes within the human body, create fast-response digital displays, gauge temperature changes with high accuracy, and many other purposes.

They exists due to the shape and polarization of the molecules. Liquid crystals (LC's) are treated as long rod-shaped molecules, fairly rigid along their long axises, and possess a strong dipole polarization. These two characteristics cause the crystals to inherit unique positional and orientational order not seen in liquids; yet they still flow much like a liquid, without the a rigid structure seen in solids. Also, there are sub-phases within liquid crystals that demonstrate different positional, orientational, and bond-orientational order. Three phases that are commonly recognized within these crystals are the nematic, smectic A (SmA), and smectic C (SmC) phases.

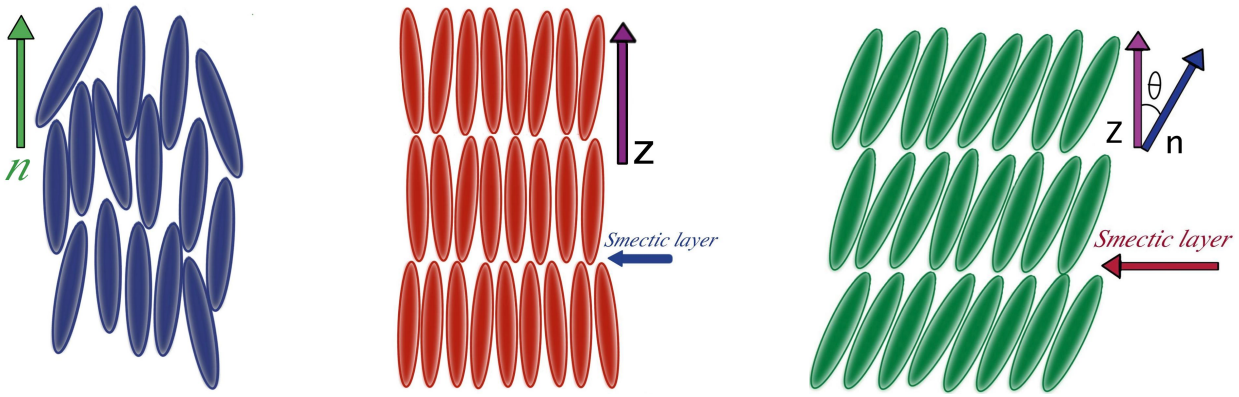


Figure 1: The illustration on the left is an example of the nematic phase of liquid crystals. All the molecules' long axises align in a common direction, while the positions of the molecules are free to move around.

1.1 Phases within the Liquid Crystal Phase

In the nematic phase, the crystals do not possess any position order, only long-range directional order. Due to their rod-like shape, the long axis of all the crystals tend to align with one another, but their position may flow in any direction. Thus, their center of masses are randomly located throughout a sample. In Figure 1, the picture on the left illustrates a typical nematic phase. There is a common direction, along the director n, that the crystals point. Notice, the molecules do not necessarily all point in the same direction. In fact, they may align in many different directions; however, the average of all the directors point in one direction. This director is a fundamental characteristic of all liquid crystals.

The smectic phases are characterized by both orientational order and *positional order*. The position of each crystal is not distributed randomly; rather, they align into layers. Again, take a look at Figure 1. The picture in the center illustrates the smectic A phase. In this phase, all the molecules axises align with a common director z like the nematic phase, and their center of masses align with one another. As a result, the sample forms distinct layers stacked on top of one another, much like a bookshelf, and the director is pointed perpendicular to these shelves.

The smectic C phase possesses similar characteristics to the smectic A phase. The molecules form layers on top of one another, and the molecules align along a similar director. However, this director is not perpendicular to the layers. Instead, a new director n with a small tilt angle θ is induced on all the molecules. This is shown on the right in Figure 1.

1.2 Tilt Cone, Polarization, and LC Cells

When the molecules are in the SmC phase and experience such a tilt, they are able to revolve along a cone formed along the z director (see Figure 2). This characteristic induces a polarization on a sample of LC's. In the following experiment, we explore this polarization and how susceptible these crystals are to changes in electric fields. The direction of such a polarization $\mathbf{P}$ for liquid crystals is $\hat{z} \times \hat{n}$. Figure 2 shows that the direction of $\mathbf{P}$ is into the page. As the molecules spins around the tilt cone, the polarization direction moves around the top of the cone like hands of a clock. If an external electric field is applied across the short axis of the liquid crystal parallel to the layers, the molecule will revolve until the polarization direction aligns with the field. For our experiments, we exploit this property by observing these materials between two conducting glass slides.

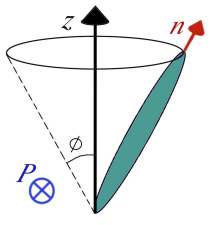


Figure 2: Crystals in the SmC phase experience a tilt, with angle θ, and are free to revolve around a cone.

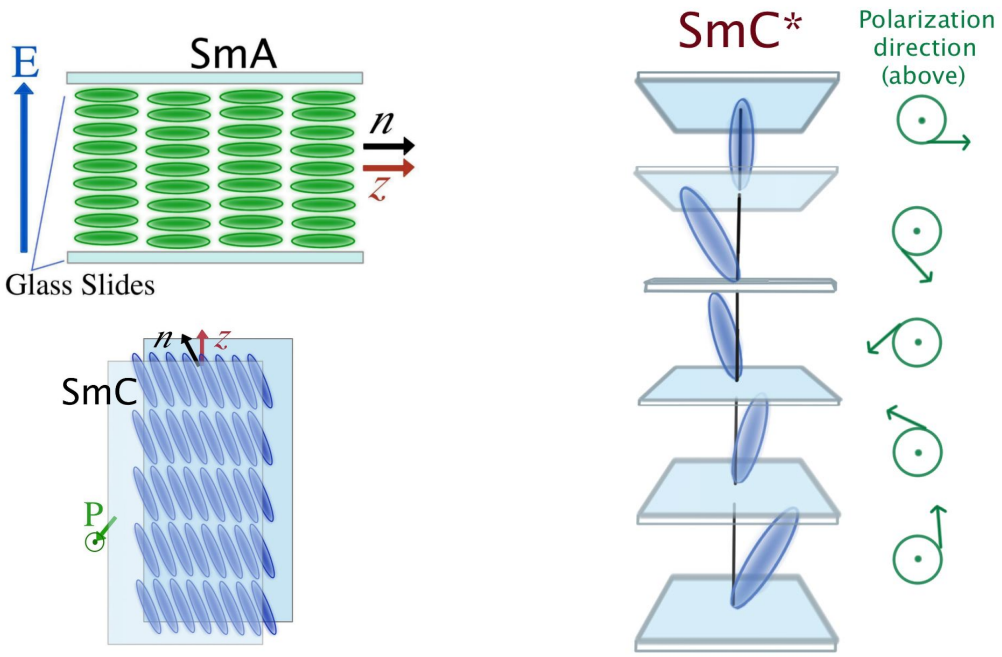


Figure 3: The capacitor consists of two plates separated by a distance, d. When a dielectric material is between the two plates (such as liquid crystals), the properties of the capacitor changes.

Recall the bookshelf analogy from before. In our liquid crystal samples, the glass slides are the sides of the book shelf. The layers are the shelves, and the crystals are the books standing upright. If the crystal is in the SmA phase, the $\hat{z}$ (perpendicular to the layers or shelves) and $\hat{n}$ directors (parallel to the orientation of the books) are parallel to each other. The direction of polarization, by definition, is the cross product of these two directors; therefore, there is no polarization in the SmA phase (See Figure 3). When the crystal melts into the SmC phase (the books lean on the shelf), the tilt induces a polarization also shown in Figure 3.

Our samples are built using two glass slides coated with indium-tin oxide for conducting electric charge at their surface. We add a separate coating of nylon material and run a brush over the surface of the slide. This "scratches" microscopic grooves onto the glass that causes the liquid crystal material to align between the slides. Micro-spacers are dropped onto the slide to create micrometer spacing between the two plates. Before the liquid crystal material is applied, we use an Ocean Optics spectrometer to measure the thickness of the cells. Two wire leads are connected to the surface of the cells using conducting epoxy. These connections allow us to apply electric fields across the slides and polarize the material [9]. Figure 8 illustrates a liquid crystal slide used in this experiment.

1.3 Ferroelectricity, Antiferroelectricity, and Chirality

The principle of **ferroelectricity** is important property of some SmC liquid crystal phases. It is analogous with ferromagnetism, which is the permanent magnetization present in materials like iron ("ferro" means iron). Ferroelectricity describes the phenomena when the bulk of a material has a permanent electric polarization without an external electric field, and all the molecules point in the same direction. This property can be reversed in the presence of an external field. The SmA phase of liquid crystals do not exhibit ferroelectric behavior. In SmC phase, the tilt of the molecules induces a polarization in each of the layers of the sample. If the layers all point in the same direction, the sample is ferroelectric. It is also possible for each layer to polarize in opposite directions, and the bulk to show

zero net polarization (a phenomena known as **antiferroelectricity**). If layers' directors do not align or show any antiferroelectric symmetry, the system is neither ferroelectric or antiferroelectric. All of the samples used in this experiment are ferroelectric.

Ferroelectricity brings the concept of symmetry to question. In several branches of science, the term **chirality** describes the characteristic of asymmetry. In liquid crystals, chirality can be used to describe individual molecules or a bulk system of LC's. If a bulk is not identical to its mirror image , the system is considered chiral [5]. If a molecule is not symmetric about its short or long axis, the molecules itself is considered chiral. Achiral systems show these kinds of symmetry. For example, a unique achiral phase is the helical structure shown in Figure 3. This type of SmC phase has no net polarization in the sample (antiferroelectric). When the layers are formed and an angle is induced in the SmC phase, each layer has a different polarization direction with an angle ϕ in the azimuthal plane. Each layer's polarization angle is slightly shifted from the surrounding layers, forming a helix-twisted structure in the bulk sample. This phase is described as an achiral antiferroelectric SmC liquid crystal [3]. Phases that demonstrate chiral structure are labeled with an asterisk (i.e. SmC*).

1.4 De Vries Materials

For our experiments, we use a special class of liquid crystal material known as De Vrie's LC's. These crystals are unique from other materials, because they do not exhibit any layer shrinkage between the SmA-SmC phase transition [5]. Recall, the SmA phase typically requires that all the crystals align along a common director that is perpendicular to it's layers. In the SmC phase, this common director tilts at an angle away from the horizontal, shortening the distance between layers. In the De Vries materials, there is no difference between the layer distances in the SmA and SmC phases. This lack of layer shrinkage is due to a unique property in their SmA phase. Unlike typical SmA LC's, De Vries materials actually experience a spontaneous tilt with random azimuthal angles in each layer. These angles cancel each other out, so the SmA phase still shows a net common director z perpendicular to its layers, like other materials. When the crystals transition into the SmC phase, they maintain this tilt, but their directors align in a common direction with angle θ from the z director. Thus, no layer shrinkage occurs during the phase transition.

1.5 Universality Classes

In this experiment, we seek to determine whether our liquid crystals fall into a particular universality class. Universality classes are groups of mathematical expressions that attempt to model the behavior of physical systems. Each expression contains the same set of "critical exponents": α the heat capacity exponent, β the magnetic order parameter exponent, γ the susceptibility exponent, and ν the correlation length. A physical material is considered part of a class when the it's properties align with the critical exponents of the model. This process of classification allows scientists to study complex, foreign materials more efficiently. If a material's behavior fits the universality class of a known substance or computer simulated system, a scientist can perform similar experiments on both.

Class Name	α	β	γ	ν	Dimension count
XY Model	-0.0146(8)	0.3485(2)	1.3177(5)	0.67155(27)	3
2d Ising Model	0	1/8	7/4	1	2
3d Ising Model	0.1096(5)	0.32653(10)	1.2373(2)	0.63012(16)	3
Mean Field (Landau) Model	0	1/2	1	1/2	2, 3

Table 1: Universality classes and their critical exponents used to describe the behavior physical substances [2].

A few common universality classes are listed in Table 1. In earlier experiments, the XY critical

exponents seemed to accurately describe the classical behavior of liquid crystals; however, later experiments shows that around the SmC-SmA phase transition, the crystals behave similar to the mean field, Landau model [14]. In this experiment, we determine the susceptibility exponent γ for our samples and compare them to the above models. In the SmC-SmA transition, we expect to see a critical exponent similar to the Landau model.

2 Experimental Design - The Theory Behind Dielectric Spectroscopy

We use the technique of dielectric spectroscopy to determine the susceptibility critical exponent. Dielectric spectroscopy is an important tool in studying the molecular dynamics and relaxation characteristics in ferroelectric LC's. This measurement uses an alternating current and impedance analyzer to measure the crystals' polarization response in an electric field. This measurement is repeated over multiple temperatures to obtain a spectrum of the susceptibility, and the data is compared with the universality models.

2.1 Review of a Capacitor

Before diving into the complex details of dielectric spectroscopy, we begin with a review of the physics of capacitors. Liquid crystal slides behave as a capacitor with dielectric material between the two plates. Recall, that the capacitor consists of two parallel conducting plates with a separation, d, between them. Each plate holds equal and opposite charge, creating an electric field between them.

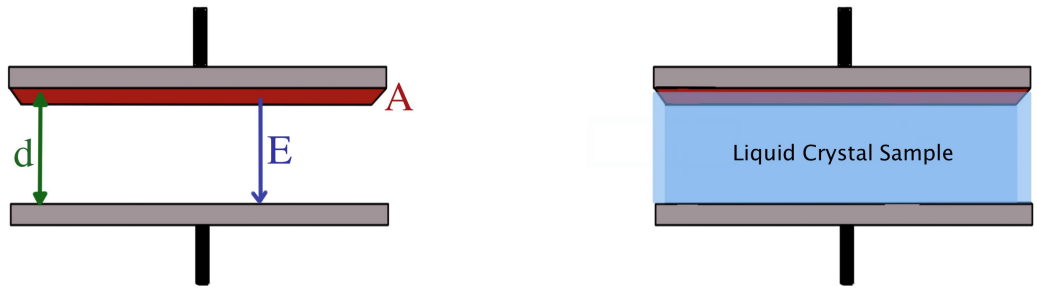


Figure 4: The capacitor consists of two plates separated by a distance, d. When a dielectric material is between the two plates (such as liquid crystals), the properties of the capacitor changes.

The ratio of the charge on each plate over the potential between them is known as the capacitance:

$$C = \frac{Q}{V}. \tag{1}$$

Assuming that the area, A, of the plates is much larger than the distance, d, between plates, the potential between the two plates is defined as the line integral of the electric field in the capacitor.

$$V = \int_0^d E dz = \int_0^d \frac{\rho}{\epsilon_0} dz = \frac{\rho d}{\epsilon_0} = \frac{Qd}{\epsilon_0 A} \tag{2}$$

where ρ is the free charge collected at the surface of the capacitor and ϵ_0 is the permittivity of free space. Therefore, the total charge on the surface of the capacitor is

$$\frac{Q}{A} = \epsilon_0 E. \tag{3}$$

This new expression for the potential can be substituted into the equation for capacitance. Thus, the capacitance is directly related to the permittivity within the capacitor (in this case, this region is a vacuum), and indirectly related to the electric field:

$$C_0 = \frac{\epsilon_0 A}{d} = \frac{Q}{Ed} \tag{4}$$

In this experiment, an alternating power source is used to observe the response of the capacitor when a potential is applied across the plates. As the potential difference between the glass slides switches direction, there is time delay before the current changes direction due to the charging at the capacitor's surface. Thus, there is lag between the applied voltage and current phase angles (shown in Figure 5) Recall that the current is defined as motion of charge over time (dQ/dt) and the potential is Q/C for a

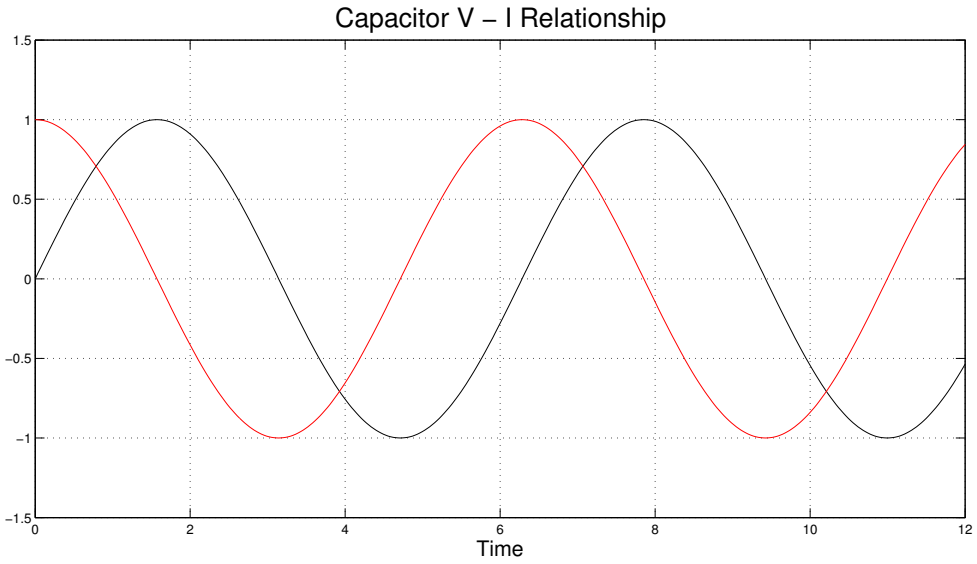


Figure 5: This shows the phase angle lag between the current and the applied potential. (The potential is the black wave while the current is the red wave.)

capacitor. Thus, we see that the current is simply the time derivative of the potential across the plates. The ratio of the potential and current is known as the impedance of the capacitor, or the opposition to current flowing across the plates when voltage is applied.

$$\begin{aligned} \frac{v_c}{i_c} &= \frac{V\sin(\omega t)}{\omega VC\cos(\omega t)} = \frac{\sin(\omega t)}{\omega C\sin(\omega t+\frac{\pi}{2})} \\ Z_{\text{capacitor}} &= \frac{1}{\omega C}e^{-j\frac{\pi}{2}} = \frac{1}{j\omega C} \end{aligned} \tag{5}$$

The impedance tells us a lot about the capacitor. It is dependent upon the frequency of the applied oscillating field; therefore, we will observe how it changes when the sample is scanned over a range of frequencies. Notice that the impedance of the capacitor is imaginary. Our samples, however, are not empty capacitors. They are comprised of two glass microscope slides with liquid crystals between them. Thus, before we begin to describe the experiment, we must review the physics of a capacitor with dielectric material between the two plates.

2.2 Capacitor with Dielectric Material

A dielectric material is an electric insulator that can be polarized in the presence of an electric field. Liquid crystals are a type of dielectric material. Electric charges do not flow through the material as

they do a conductor, rather they slightly shift in location by polarization of the molecules inside. In this case, positive charges shift in the direction of the external electric field, and negative charges away from the field. If the molecules of a material are weakly bounded, they will flip their axis of symmetry parallel to align with the direction of the field, a phenomenon known as polarization. An increase in polarization inside a capacitor will increase its surface charge. The polarization density, $\mathbf{P}$, within a capacitor is an expression for the vector field describing the number of dipole moments between the two plates.

$$\mathbf{P} = \epsilon_0 \chi \mathbf{E} \tag{6}$$

where χ is the electric susceptibility of the dielectric material, or how easily the dipoles align with the electric field. The susceptibility factor describes how sensitive the dielectric is to its environment. Temperature, density, and pressure can effect the material's response to the external electric field. Thus, this factor is the interest of dielectric spectroscopy. Equation 6 is linear under small alternating electric fields, and is related the dielectric's permittivity:

$$\epsilon_r = \chi + 1 \tag{7}$$

The total charge on the surface of capacitor's plates, altered by this polarization density caused by the dielectric, is:

$$\frac{Q}{A} = \epsilon_0 \mathbf{E} - \mathbf{P}. \tag{8}$$

Plugging in the expression for the polarization density and solving for the electric field, we can see how the electric field is effected by the dielectric (due to the susceptibility constant).

$$\mathbf{E} = \frac{Q}{A\epsilon_0(1+\chi)} = \frac{Q}{A\epsilon_0\epsilon_r} \tag{9}$$

The new capacitance of the slides is

$$C = \frac{Q}{dE} = \frac{Q}{\frac{Q}{A\epsilon_0\epsilon_r}d} = \frac{A\epsilon_0\epsilon_r}{d} = C_0\epsilon_r. \tag{10}$$

Notice, there is a new factor ϵ_r. Thus, by measuring the capacitance of our samples, we can study the polarization susceptibility of liquid crystals. We will exploit this fact through using alternating electric fields at different frequencies and observing the sample's response. After scanning across a wide range of frequencies, we expect to see a particular frequency at which the capacitance falls off (and conductance peaks).

2.3 The Theory Behind the Experiment

In this experiment, we use a HP 4192A LF Impedance Analyzer to measure the dielectric properties of our liquid crystal samples. This machine applies an AC voltage across the sample and measures the magnitude of it's impedance. Recall from earlier, the applied voltage lags the current through the capacitor by some angle. This device measures this phase angle (as well as the magnitude), allowing us to determine the real and imaginary parts of the total impedance. Figure 6 shows the imaginary versus real plane. The real part is known as the resistance of the circuit, R, while the imaginary part is known as the reactance, X. Equation 5 describes the impedance of a capacitor as imaginary; therefore, an ideal liquid crystal sample would only displace the imaginary part. However, there is a small bit of resistance in the samples due to the electrode connections across the slides. Using the data output from the impedance analyzer, the magnitude and phase angle, we can determine the values for the reactance and resistance:

$$R = |Z|\cos\theta \quad \text{and} \quad X = |Z|\sin\theta \tag{11}$$

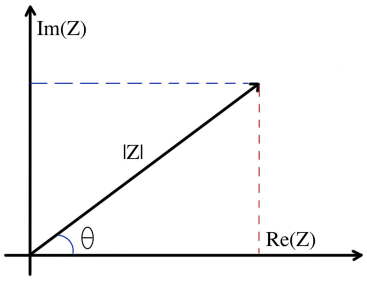


Figure 6: Imaginary Vs. Real plane for the capacitor's impedance. $|Z| = \sqrt{R^2 + X^2}$

These two expressions lead us to the complex impedance:

$$Z = R + iX. \tag{12}$$

If we take the reciprocal of this value, we get the admittance, Y, which can lead us straight to the capacitance and conductance of the sample. The complex admittance is

$$Y = G + i\omega C \tag{13}$$

where G is the conductance and C is the capacitance of the circuit (The expression ωC is known as the susceptance B). In this experiment, we measure the impedance and use it to calculate the values for G and C. These expressions are simply:

$$G = \frac{R}{R^2 + X^2} \quad \text{and} \quad C = \frac{-X}{\omega(R^2 + X^2)} \tag{14}$$

From these, we can derive the complex dielectric variable, ϵ_r, which has real and imaginary parts ($\epsilon_r = \epsilon' + i\epsilon''$). Assuming our sample behaves much like a dielectric capacitor (the premise of our entire model), the admittance will be:

$$Y_{sample} = \frac{1}{Z_{sample}} = j\omega C = \frac{A\epsilon_0 j\omega\epsilon_r}{d} = C_0 j\omega\epsilon_r. \tag{15}$$

Using this expression for the admittance, we can relate the dielectric variable, ϵ_r, directly to the capacitance and conductance.

$$\epsilon_r = \frac{G}{j\omega C_0} + \frac{j\omega C}{j\omega C_0} \tag{16}$$

We call these two values the real and imaginary parts of the dielectric variable.

$$\epsilon' = \frac{C}{C_0} \quad \text{and} \quad \epsilon'' = \frac{G}{\omega C_0}. \tag{17}$$

The behavior of these two values is the interest of this experiment. We will observe the changes of these spectrums at different frequencies and temperatures.

The Debye relaxation model describes a dielectric's relaxation response to a population of ideal dipoles exposed to an alternating external field. When the liquid crystal is in the SmC phase, it behaves similar to bulk of ideal dipoles. Thus, we will use this model to monitor and probe our data for peaks in dielectric response. It will be used to fit out experimental data and determine the susceptibility of the crystals to align with the field. The general model is:

$$\epsilon'' = \epsilon_\infty + \frac{\Delta\epsilon}{1 + [i(f/f_r)]^\beta} + g(f) \tag{18}$$

where ϵ_∞ is the permittivity at the high frequency limit, $\Delta\epsilon$ is the relaxation strength or susceptibility, f_r is the relaxation frequency, and $g(f)$ is the lower-frequency contribution [12]. This model contains complex values and can be plotted for both the real and the imaginary part against the frequency (known as a Debye plot). Figure 7 shows the general shape of this model.

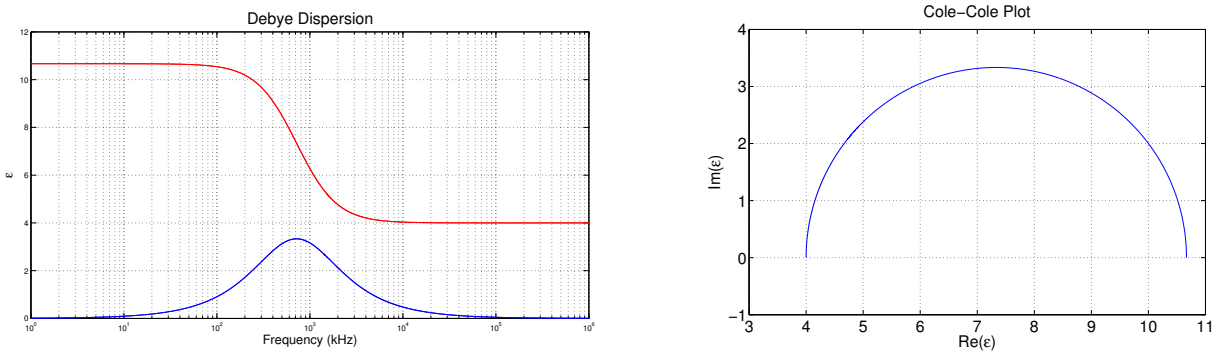


Figure 7: The left plot shows and example of a Debye dispersion. The real part (ϵ') of the model is plotted as the red line, while the imaginary part (ϵ'') is plotted as the blue line. The right plot shows a typical Cole-Cole plot of Debye dispersion.

It is important to remember our goal in this experiment. We are seeking to measure the electrical susceptibility χ_e of our liquid crystal samples. From this figure, we can see that the imaginary part of the dielectric variable shows a distinct "peak" known at a mode. Recall that this plot is related to the conductance G (or ease of current flow through the sample) of the liquid crystal cell. Thus, this peak describes a rise in conductance and shows that the crystals are more *susceptible* to align with the alternating electric field at this frequency. Ultimately, this is a plot of the samples susceptibility factor.

Using the Debye dispersion model, we measure the height of any modes, $\Delta\epsilon$, and determine the dielectric constant at that temperature. The reciprocal of this value is plotted against the temperature for each measurement, and the susceptibility critical exponent, γ, is determined used a fit with the power law model [7]:

$$(\Delta\epsilon)^{-1} = A|t|^{\gamma} + B. \tag{19}$$

2.4 What is Going on with the Liquid Crystal?

Now that we know how to make the measurements, we take a moment to explain what exactly we are measuring. More specific, what is happening inside the liquid crystal cell? As described earlier, when a material is in the SmC phase, it tilts an angle θ away from the perpendicular director z. While the average direction of all the molecules have the same director n, this director has the freedom to move anywhere on the tilt cone. Especially in an alternating electric field, the molecules will revolve back and forth to align with the field. In this experiment, we observe the response of the crystals to such a field by measuring the dielectric coefficient shown in Figure 7. The imaginary piece ϵ" of this coefficient is the susceptibility of the crystals to revolve at the frequency of the alternating field.

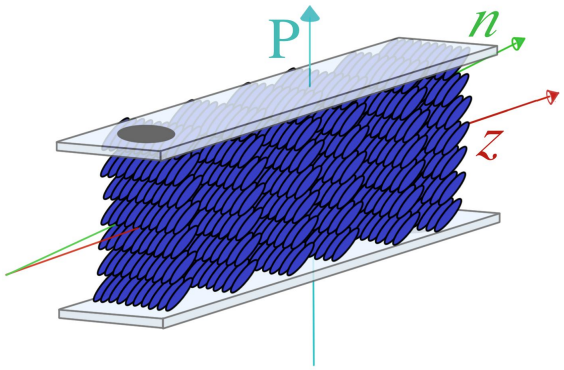


Figure 8: Illustration of a typical SmC liquid crystal sample between two conducting slides. Due to the direction of the tilt (along the horizontal plane of the cell), the polarization points vertically through the sample.

Any peaks visible on this graph describe the phenomenon when the crystals revolution rate is closest to matching the frequency of the field. This is known as the **resonance frequency** of the crystal.

If the crystal is in the SmA phase, we expect the sample to have no polarization, because the z and n director are parallel. The illustration in Figure 3 shows the alignment of the crystals with our glass slides. The crystals do not flip or align with any external electric field applied across the slides without polarization. Therefore, when observing the dielectric response of the crystal in this phase, we expect to see no resonance peak in the measurements.

If the crystal is in the SmC phase, the crystals have a polarization from the tilt cone. As a field is applied across the slides, the polarization director aligns with this field and the molecules revolve accordingly. Therefore, these type of crystals typically have high dielectric permittivity in the SmC phase due to a collective relaxation response of every crystal in the system. This phenomena is known as the Goldstone mode (GM) [5]. This is observed as a peak in the dielectric absorption over a range of frequencies. We repeat these measurements at many different temperatures to create a surface plot of the dielectric spectrum. On this surface, we look for a clear phase transition at the transition temperature for each sample we test.

3 Obtaining LC Dielectric Spectrum

In this experiment, we use two different De Vries materials with virtually no layer shrinkage. The first material is a 8422[2F3] sample. Using the Ocean Optics spectrometer, we determined the width of the sample to by 5μm. The second material is a TSiKN65 sample and the width is measured to be 5μm. We use a Hewlett-Packard Impedance Analyzer 4192A LF to apply a 0.1V oscillating current through each sample and measured the magnitude of the impedance and it's complex angle. The dielectric values are determined from these impedances using the conversion described the previous section. Using a custom National Instruments Labview program to automate the experiment, we determine these values over a range of frequencies. This technique is repeated at multiple temperatures (near the SmA-SmC transition) to obtain a 3D surface spectrum of the dielectric absorption. These surfaces for each material are shown in Figures 9

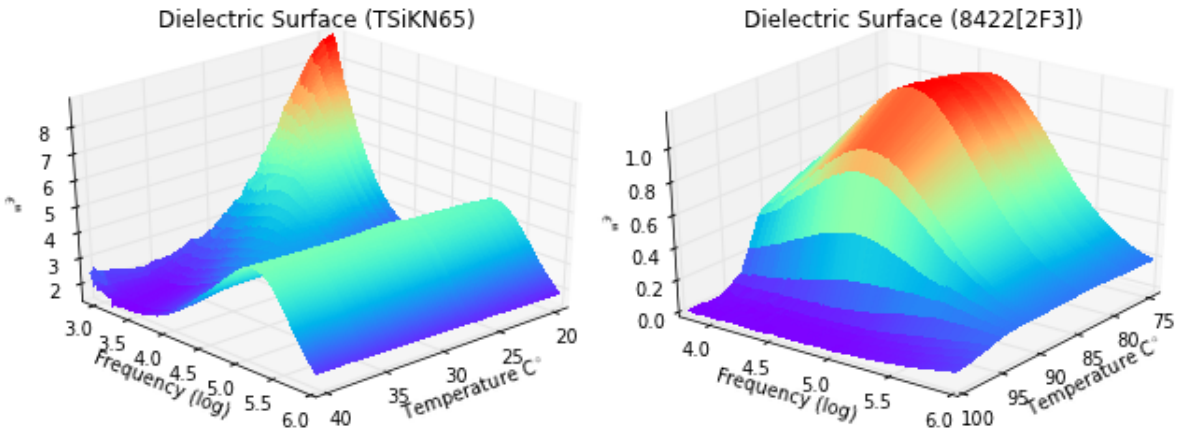


Figure 9: Dielectric surface plots for the TSiKN65 and 8422[2F3] samples

3.1 Soft-Mode Absorption

From these spectra, one can easily see the Goldstone mode that exists throughout the SmC phase, described earlier. This is heavily influenced by effects from surface anchoring and cannot be easily

quantified [1]. For this experiment, we restrict ourselves to the analysis of the mode that appears right at the transition temperature. This is known as the soft mode and results from the fluctuations of the tilt angle. A typical De Vries material exhibit large soft-mode absorptions because of the reduced layer shrinkage. In a non-De Vries material, the restoring force of tilt fluctuations should be close to the elastic energy required for layer shrinkage (induced by the tilt) at the transition temperature. This means the switching of the tilt director will require more energy than a De Vries material, and dielectric absorption will be small in this region. In the De Vries materials, the molecules are already tilted in the SmA phase and can easily align and tilt in the oscillating frequency. In general, the weaker the coupling between the layer spacing and tilt fluctuations, the higher the soft-mode response.

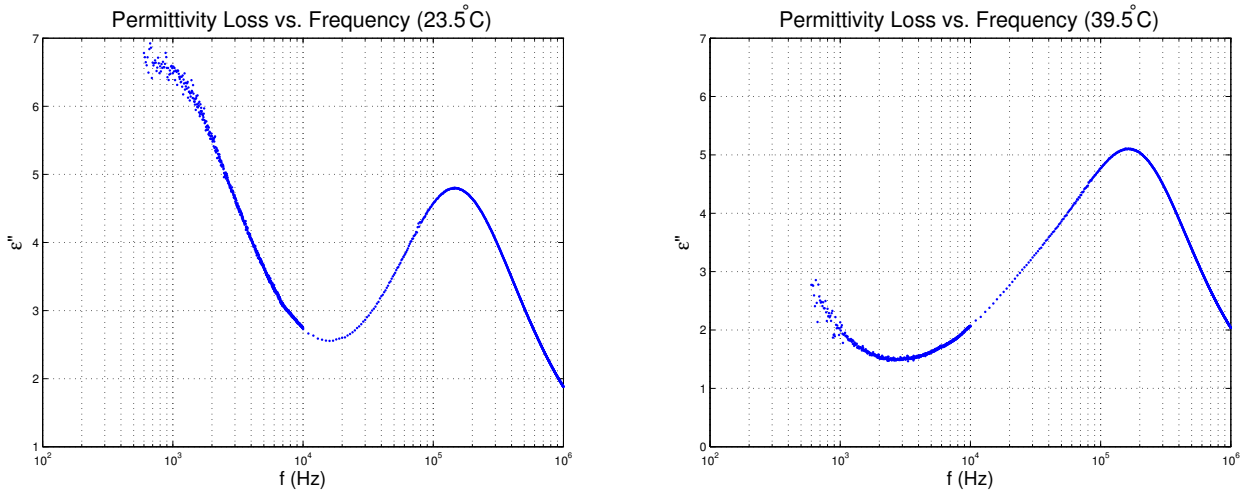


Figure 10: The left plot shows and example of a Debye dispersion. The real part (ϵ') of the model is plotted as the red line, while the imaginary part (ϵ'') is plotted as the blue line. The right plot shows a typical Cole-Cole plot of Debye dispersion.

We quantify the soft-mode data by using the Debye Equation (Equation 7) for each spectrum. We use IPython notebook and Matplotlib to plot the data and fit our model to each mode in a spectrum. The modes are characterized by their dielectric susceptibility χ, distribution parameter β, and resonant frequency. A least-squares-fit function is used to determine these values. The results for two temperatures, 23.5°C and 39.5°C of the TSiKN65, are shown in Figures 10 and 11, comparing the different behaviors for the dielectric spectra in the SmC and SmA phases. (In both phases, there is a dielectric resonance peak in the 10^5 Hz range known as the dipolar relaxation. The dipole polarization orientation is disturbed by thermal noise and there is a relaxation period where the molecules realign their polarization vectors with the external field. This peak is an intrinsic property and is ignored for this paper) [10].

3.2 Landau Theory

For the experiment, we are looking to compare this behavior to the mean-field theory for the susceptibility of these materials. One conclusion of the theory is that the susceptibility of order-parameter fluctuations (like the soft mode fluctuations) for a second-order transition point T_C diverges according to:

$$\Delta\epsilon = \chi = \frac{1}{m(T - T_C)} \tag{20}$$

From this equation, we plot the reciprocal of our χ against the $T - T_C$ and extract the constant slope m. This plot is known as a Curie-Weiss Plot. In our experiment, we expect the crystals to show linear

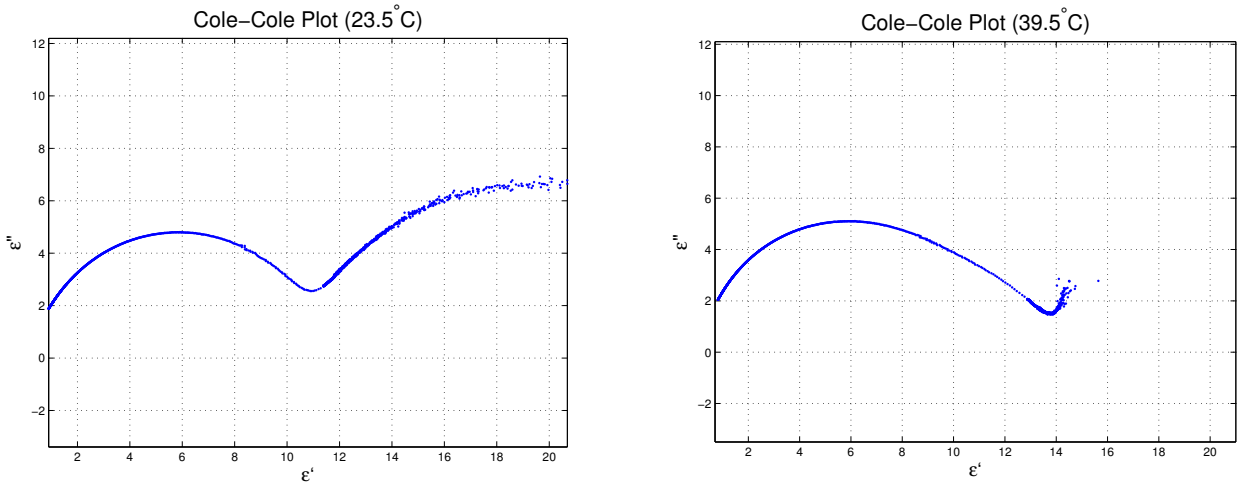


Figure 11: The left plot shows and example of a Debye dispersion. The real part (ϵ') of the model is plotted as the red line, while the imaginary part (ϵ'') is plotted as the blue line. The right plot shows a typical Cole-Cole plot of Debye dispersion.

behavior, indicating a second-order phase transition. This can be modeled by the Landau expansion of the free energy:

$$g = g_0 + \frac{1}{2}\alpha(T - T_C)\Theta^2 + \frac{1}{2\chi_\infty \epsilon_0}P^2 - C\Theta P - PE, \tag{21}$$

where Θ is the tilt angle, P is the polarization, E is the electric field applied, and g_0 is the nonsingular term corresponding the SmA phase [10]. A smaller leading Landau coefficient α broadens the temperature region where the mean-field theory is valid [4]. This coefficient measures how easily the tilt angle can fluctuate in these materials from external perturbations (like an oscillating electric field). From our conclusions earlier in this section, a larger soft-mode from small layer shrinkage leads to a smaller restoring force and smaller values of α. In the Landau model, this means the tilt-angle is easily effected by external electric fields.

If we minimize the free energy equation (Equation 21) with respect to the polarization, this gives the spontaneous polarization [1]

$$P_s = \chi_\infty \epsilon_0 C \theta_s \tag{22}$$

where χ_∞ is $\epsilon_\infty - 1$ and C is a constant that couples the tilt and the polarization. We obtain this value from optical experiments done by M. Krueger and F. Giesselmann in 2005 [4]. Therefore, we expect the slope of our Curie-Weiss plot to be

$$m_\chi = \frac{\alpha}{\epsilon_0 C^2 \chi_\infty^2}. \tag{23}$$

We solve this equation in terms of α to determine the soft-mode susceptibility Landau coefficient of the liquid crystal

$$\alpha = \frac{m_\chi m_{P\theta}^2}{\epsilon_0} \tag{24}$$

where $C^2\chi_\infty^2 = m_{P\Theta}^2$. This coefficient quantifies the dielectric susceptibility of the system in the presence of external electric perturbation.

Sample	SmA-SmC Transition (°C)	m_χ	B	γ value	α (K^{-1})	$m_{P\Theta}$
TSiNK65	26.3	0.014	-0.20	0.99	5.26	185
8422[2F3]	65	-0.0041	0.603	0.95	-1.60	185

Table 2: The results from the dielectric experiment for both samples.

4 Results and Discussion

For the TSiNK65 sample, we sweep the frequency from 100 Hz to 10^6 Hz. These measurements are repeated for temperatures between 19°C and 40°C and the $\Delta\epsilon$ is determined from our fitted Debye-model. This value is used to make the Curie-Weiss plot shown in Figure 12. We use a linear fit function to determine the slope of this data m_χ. The values for this fit are shown in Table 2. The first thing we notice is that the value for γ is near 1. This is our indication that our sample experienced a second order phase transition, which follows the mean-field universality class. We plug these values into Equation 24 to determine the value of α. In our experiment, we determine this value to be 5.255^{-1}K.

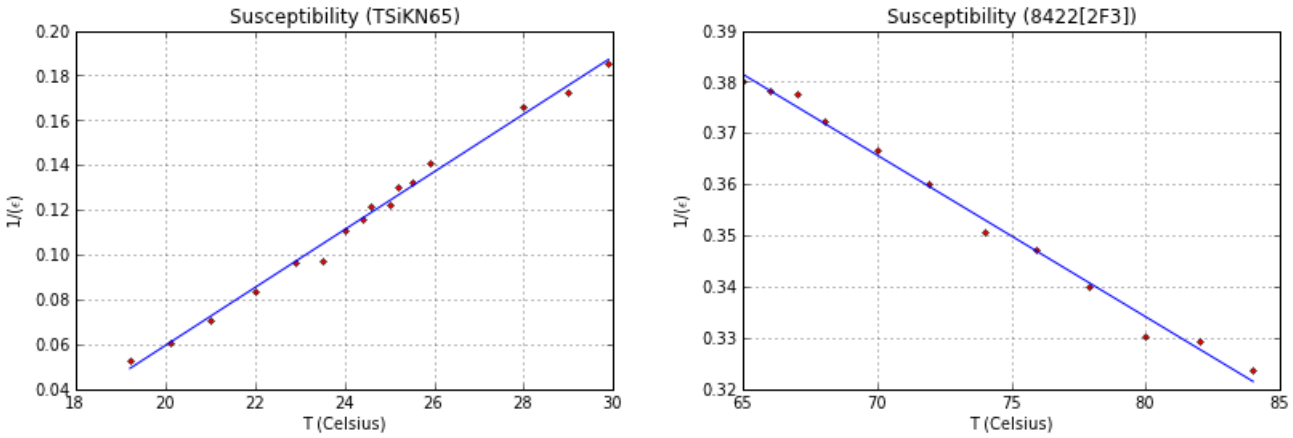


Figure 12: The Curie-Weiss plots for each of our samples. The plot on the left shows the data and fitted line for the TSiKN65 material. The plot on the right shows the data and fitted line for the 8422[2F3] material.

The Curie-Weiss plot for 8422[2F3] sample is shown in Figure 12. The frequencies are swept from 1kHz to 10^6Hz, and the temperatures range from 55°C to 100°C. We repeat the measurements and calculations to obtain the value of m_χ. The values are shown in Table 2. This sample also behaves as a second order phase transition with γ near 1. The unusual result from our experiment is that the value of m_χ is negative, causing the value of α to also be negative. Understanding this behavior requires more exploration beyond the scope of this experiment. We determine the value of α to be -1.60 K^{-1}.

The results from our experiment confirm our theory that the SmA-SmC transition is a second order phase transition and follows Landau free expansion behavior. The susceptibility coefficient α proves to be smaller for these crystals than non-De Vries materials measured in previous experiments (i.e. "Study of De Vries behavior of the smectic A and smectic C phase transition" [10]). As described earlier, a smaller coefficient implies that less energy is needed to align the material's polarization with the external field. This also implies that the restoring force for the layer shrinkage (or molecule tilt) is smaller than typical SmC liquid crystals. Because the De Vries materials do not experience layer shrinkage, it makes sense that this restoring force is small. Hence, De Vries materials will always have lower values of α. This is confirmed by our experiment.

References

[1] R. Blinc and B. Zeks. Soft modes in ferroelectric and antiferroelectric liquid crystals. *Physics Review Journal*, 1978.

[2] Massimo Campostrini, Martin Hasenbusch, Andrea Pelissetto, Paolo Rossi, and Ettore Vicari. Critical behavior of the three-dimensional xy universality class. *Physical Review B*, Vol. 63, May 2001.

[3] Deblal Das, Tanmoy Lahiri, and Tapas Pal Majumder. Theoretical investigation of helix distortion and dielectric spectrum of antiferroelectric liquid crystals. *Physica B*, February 2011.

[4] Michael Krueger and Frank Giesselmann. Dielectric spectroscopy of de vries-type smectic a - smectic c transitions. *Physical Review*, 71, 2005.

[5] P Malik, K K Raina, A Bubnov, A Chaudhary, and R Singh. Electro-optic switching and dielectric spectroscopy studies of ferroelectric liquid crystals with low and high spontaneous polarization. *Thin Solid Films*, August 2010.

[6] U Manna, Jang-Kun Song, Yu P Panarin, Atsuo Fukuda, and J K Vij. Electro-optic and dielectric study of the de vries-type smectic-a* phase exhibiting transitions to smectic-c phase. *Physical Review*, Vol. 77, 2008.

[7] Amrita Mukherjee, S S Bhattacharyaa, B K Chaudhuri, and S L Wu. Effect of polymer relaxation on dielectric spectroscopic study of polymer-ferroelectric liquid crystal composites. *Journal of Molecular Liquids*, Vol. 148, July 2009.

[8] P Perkowski. How to determine parameters of soft mode from dielectric spectroscopy performed using cells with ito electrodes. *Opto-Electronics Review*, Vol. 19, 2011.

[9] P Perkowski. Dielectric spectroscopy of liquid crystals. electrodes resistivity and connecting wires inductance influence on dielectric measurements. *Opto-Electronics Review*, Vol. 20(1):pg. 79–86, 2012.

[10] N Podoliak, V Novotna, M Glogarova, V Hamplova, M Kaspar, A Bubnov, K Kapernaum, and F Giesselmann. Study of de vries behavior of the smectic a* - smecticc* phase transition. *Phase Transitions*, Vol. 83, July 2010.

[11] S Krishna Prasad, D S Shankar Rao, S Sridevi, Jawad Naciri, and B R Ratna. Critical behavior of three organosiloxane de vries-type liquid crystals observed via the dielectric response. *Journal of Physics: Condensed Matter*, Vol. 23, February 2011.

[12] Yuji Sasaki, Kenji Ema, and Haruhiko Yao. Static and dynamic critical behaviors in de vries liquid crstals. *Liquid Crystals*, 37:571–577, May 2010.

[13] Shri Singh, Avanish Singh Parmar, and Abhilasha Singh. Phase transitions in ferroelectric liquid crystals. *Phase Transitions*, Vol. 81(9), September 2008.

[14] K Takekoshi, Y Sasaki, K Ema, H Yao, and Y Takanishi H Takezoe. Quasi-two-dimensional ising critical behavior of de vries liquid crystals observed in the heat capacity and dielectric response. *Physical Review*, Vol. 75, 2007.

5 Appendix I: Labview Program

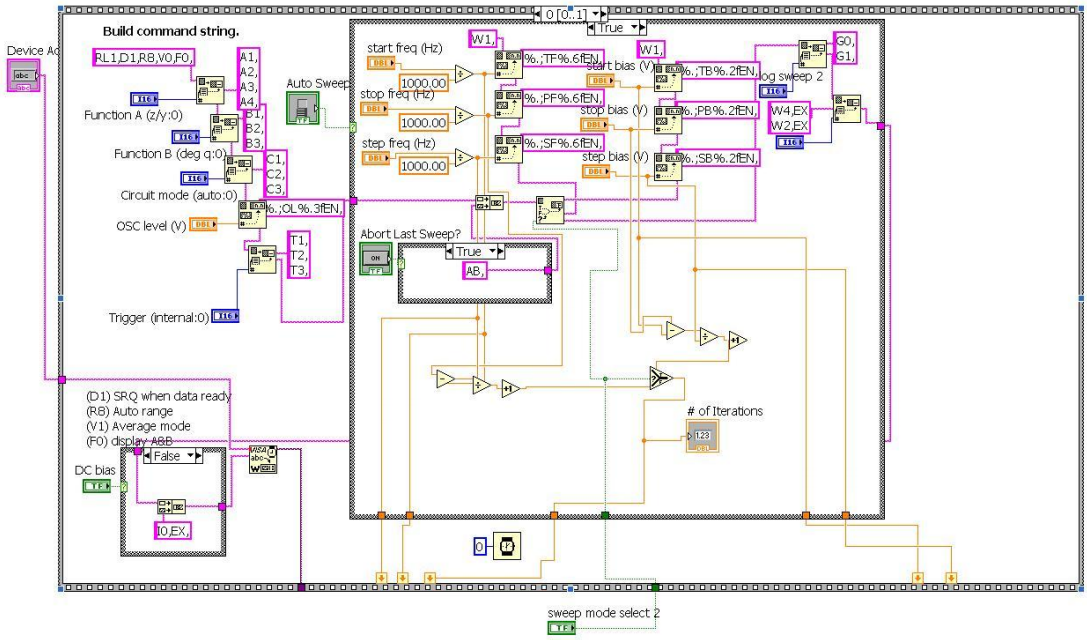


Figure 13: The Labview block diagram that describes the front panel and controls of the program.

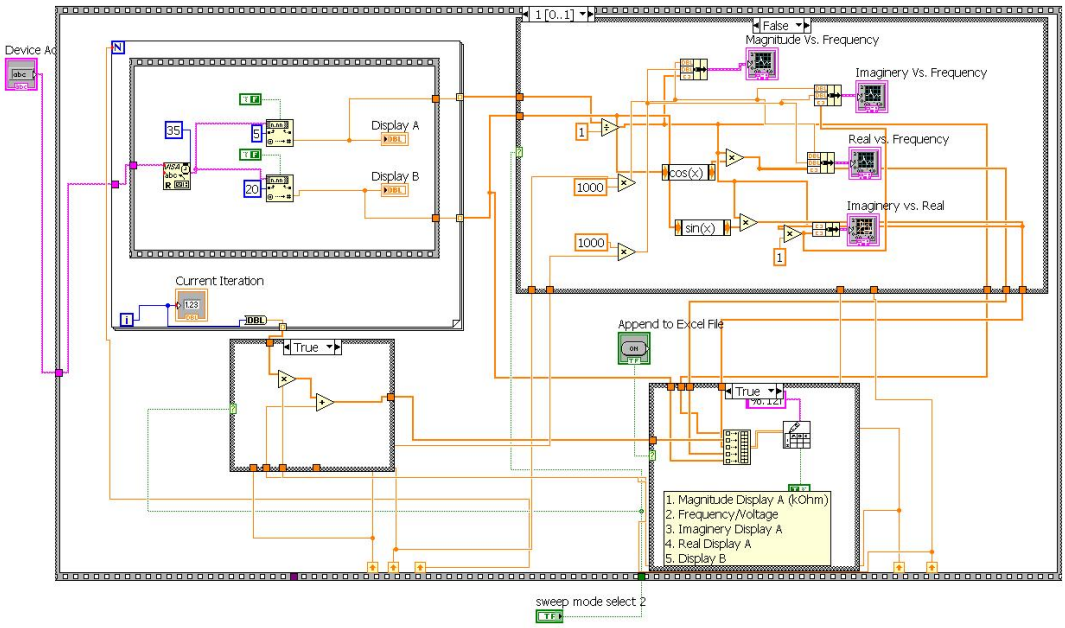


Figure 14: The Labview block diagram that describes the logging and graphing of the data from the program.

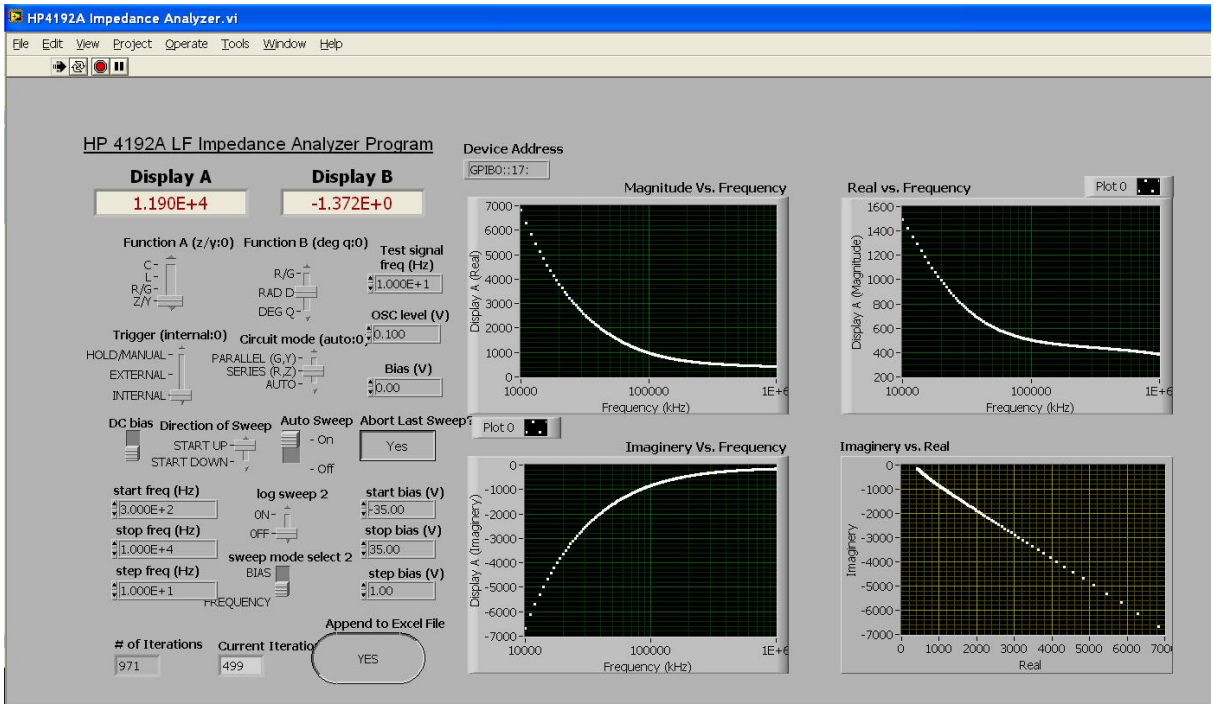


Figure 15: The Labview front panel UI.

6 Appendix II: IPython Notebooks

Full IPython notebook Analysis at the following website:
https://github.com/Zsailer/calpolythesis

Obtaining Liquid Crystal Dielectric Spectrums

In this experiment, we use two different De Vries materials with virtually no layer shrinkage. The first material is a 8422[2F3] sample. Using the Ocean Optics spectrometer, we determined the width of the sample to by 5μm. The second material is a TSiKN65 sample and the width is measured to be 5μm. We use a Hewlett-Packard Impedance Analyzer 4192A LF to apply a 0.1V oscillating current through each sample and measured the magnitude of the impedance and it's complex angle. The dielectric values are determined from these impedances using the conversion described the previous section. Using a custom National Instruments Labview program to automate the experiment, we determine these values over a range of frequencies. This technique is repeated at multiple temperatures (near the SmA-SmC transition) to obtain a 3D surface spectrum of the dielectric absorption.

Plot the 3D surface to show all data taken over all temperatures

Initial setup for IPython notebook

In [2]:
```
%pylab inline
from mpl_toolkits.mplot3d import Axes3D
import scipy.optimize as spo
```

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

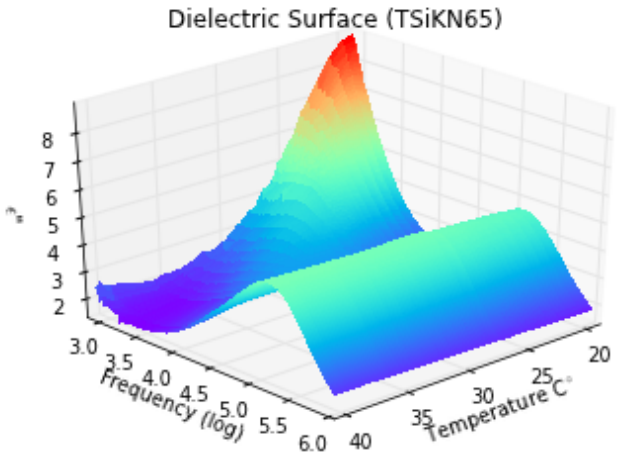
TSiKN65 Surface Plot

In [18]:
```
files = ['test01','test02','test03','test04','test05','test06','test07','test08',
        'test09','test10','test11','test12','test13','test14','test15','test16',
        'test17','test18','test19','test20','test21','test22','test23','test24',
        'test25','test26','test27','test28','test29','test30','test31','test32',
        'test33','test34','test35','test36','test37','test38','test39','test40',
        'test41','test42','test43','test44','test45','test46','test47','test48',
        'test49','test50']

n = len(files)/2
A = .01**2;
eps0 = 8.854187e-12;
d = 5e-6;
e1=[]
e2=[]
Tem = [40.5, 39.5, 36.1, 35.1, 34.1, 33, 31.9, 30.9, 29.9, 29, 28, 27, 25.9, 25.5,
25.2,
               25, 24.6, 24.4, 24, 23.5, 22.9, 22, 21, 20.1, 19.2]
temp= []
fig = plt.figure()
```

```
ax = fig.gca(projection='3d')

for i in range(0,n,1):
    test = np.loadtxt('Data03082013/'+files[i*2], dtype=float,delimiter='\t')
    m = test.T[0]
    f = test.T[1]
    a = test.T[4]
    m = m[50:]
    f = f[50:]
    a = a[50:]

    test2 = np.loadtxt('Data03082013/'+files[i*2+1], dtype=float,delimiter='\t')
    m2 = test2.T[0]
    f2 = test2.T[1]
    a2 = test2.T[4]

    magn = np.concatenate((m,m2))
    freq = np.concatenate((f,f2))
    angl = np.concatenate((a,a2))
    temp = np.repeat(Tem[i],len(freq))

    omega = 2*np.pi*freq
    G = cos(angl)/(magn)
    C = -sin(angl)/(magn)

    C0 = (A*eps0)/(d)
    e2 = G/(omega*C0)
    e1 = C/omega*C0
    temp = np.repeat(Tem[i],len(freq))
    temp = temp

    if i == 0:
        dielectric = e2
        frequency = freq
        temperature = temp
    else:
        dielectric = np.vstack((dielectric,e2))
        frequency = np.vstack((freq, frequency))
        temperature = np.vstack((temperature,temp))

Z = dielectric
X = np.log10(frequency)
Y = temperature
surf = ax.plot_surface(Y,X,Z, rstride=1, cstride=1, cmap=cm.rainbow,
        linewidth=0, antialiased=False)
ax.view_init(azim=47)
title('Dielectric Surface (TSiKN65)')
ylabel('Frequency (log)')
xlabel('Temperature C$^\circ$')
ax.set_zlabel('$\epsilon$"')
```

Out[18]: <matplotlib.text.Text at 0xe7c1470>

-



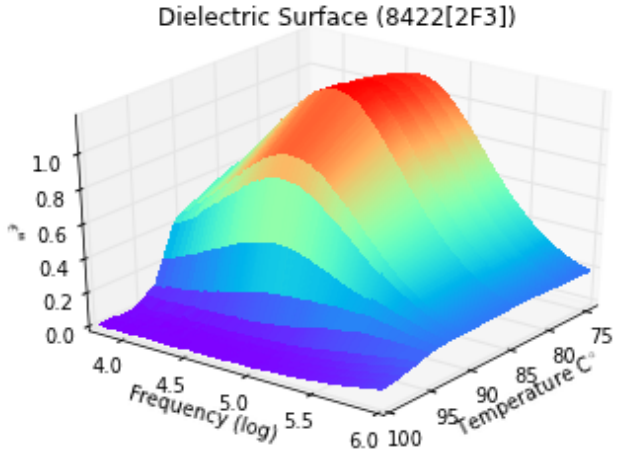
8422[2F3] Surface

In [17]:

```
files = ['test01','test02','test03','test04','test05','test06','test07','test08',
        'test09','test10','test11','test12','test13','test14','test15','test16',
        'test17','test18','test19','test20','test21','test22','test23','test24',
        'test25','test26','test27','test28','test29','test30','test31','test32']

n = len(files)/2
A = .01**2;
eps0 = 8.854187e-12;
d = 5e-6;
e1=[]
e2=[]
Tem = [100.1, 98.9, 97.9, 96.9, 96, 94.6, 93.5, 92.4, 91, 90.1, 84, 82, 80, 77.9,
               75.9, 74, 71.9, 70, 68, 67, 66, 65]
temp= []
fig = plt.figure()
ax = fig.gca(projection='3d')

for i in range(0,n,1):
    test = np.loadtxt('Data04262013/'+files[i], dtype=float,delimiter='\t')
    m = test.T[0]
    f = test.T[1]
    a = test.T[4]
    m = m[5:]
    f = f[5:]
    a = a[5:]

    magn = m
    freq = f
    angl = a
    temp = np.repeat(Tem[i],len(freq))

    omega = 2*np.pi*freq
    G = cos(angl)/(magn)
    C = -sin(angl)/(magn)
```

```
    C0 = (A*eps0)/(d)
    e2 = G/(omega*C0)
    e1 = C/omega*C0
    temp = np.repeat(Tem[i],len(freq))
    temp = temp

    if i == 0:
        dielectric = e2
        frequency = freq
        temperature = temp
    else:
        dielectric = np.vstack((dielectric,e2))
        frequency = np.vstack((freq, frequency))
        temperature = np.vstack((temperature,temp))

Z = dielectric
X = np.log10(frequency)
Y = temperature
surf = ax.plot_surface(Y,X,Z, rstride=1, cstride=1, cmap=cm.rainbow,
        linewidth=0, antialiased=False)
ax.view_init(azim=37)
title('Dielectric Surface (8422[2F3])')
ylabel('Frequency (log)')
xlabel('Temperature C$^\circ$')
ax.set_zlabel('$\epsilon$"')
```

Out[17]: <matplotlib.text.Text at 0x19414f30>



Soft Mode Absorption

From these spectra, one can easily see the Goldstone mode that exists throughout the SmC phase, described earlier. This is heavily influenced by effects from surface anchoring and cannot be easily quantified \cite{blinc}. For this experiment, we restrict ourselves to the analysis of the mode that appears right at the transition temperature. This is known as the soft mode and results from the fluctuations of the tilt angle. A typical De Vries material exhibit large soft-mode absorptions because of the reduced layer shrinkage. In a non-De Vries material, the restoring force of tilt fluctuations should be close to the elastic energy required for layer shrinkage (induced by the tilt) at the transition temperature. This means the switching of the tilt

director will require more energy than a De Vries material, and dielectric absorption will be small in this region. In the De Vries materials, the molecules are already tilted in the SmA phase and can easily align and tilt in the oscillating frequency. In general, the weaker the coupling between the layer spacing and tilt fluctuations, the higher the soft-mode response.

We quantify the soft-mode data by using the Debye Equation for each spectrum. We use IPython notebook and Matplotlib to plot the data and fit our model to each mode in a spectrum. The modes are characterized by their dielectric susceptibility χ, distribution parameter β, and resonant frequency. A least-squares-fit function is used to determine these values.

For the experiment, we are looking to compare this behavior to the mean-field theory for the susceptibility of these materials. One conclusion of the theory is that the susceptibility of order-parameter fluctuations (like the soft mode fluctuations) for a second-order transition point T_C diverges according to:

$$\Delta\epsilon = \chi = \frac{1}{m(T - T_C)}$$

From this equation, we plot the reciprocal of our χ against the $T - T_C$ and extract the constant slope m. This plot is known as a Curie-Weiss Plot. In our experiment, we expect the crystals to show linear behavior, indicating a second-order phase transition. This can be modeled by the Landau expansion of the free energy:

$$g = g_0 + \frac{1}{2}\alpha(T - T_C)\Theta^2 + \frac{1}{2\chi_\infty \epsilon_0} P^2 - C\Theta P - PE,$$

where Θ is the tilt angle, P is the polarization, E is the electric field applied, and g_0 is the nonsingular term corresponding the SmA phase \cite{devries}. A smaller leading Landau coefficient α broadens the temperature region where the mean-field theory is valid \cite{dielectric}. This coefficient measures how easily the tilt angle can fluctuate in these materials from external perturbations (like an oscillating electric field). From our conclusions earlier in this section, a larger soft-mode from small layer shrinkage leads to a smaller restoring force and smaller values of α. In the Landau model, this means the tilt-angle is easily effected by external electric fields.

If we minimize the free energy equation with respect to the polarization, this gives the spontaneous polarization \cite{blinc}
$P_s = \chi_\infty \epsilon_0 C \theta_s$

where χ_∞ is $\epsilon_\infty - 1$ and C is a constant that couples the tilt and the polarization. We obtain this value from optical experiments done by M. Krueger and F. Giesselmann in 2005 \cite{dielectric}. Therefore, we expect the slope of our Curie-Weiss plot to be

$$m_\chi = \frac{\alpha}{\epsilon_0 C^2 \chi_\infty^2}.$$

We solve this equation in terms of α to determine the soft-mode susceptibility Landau coefficient of the liquid crystal

$$\alpha = \frac{m_\chi m_{P\theta}^2}{\epsilon_0}$$

where $C^2 \chi^{2}{\infty} = m^{2}{P\Theta}$. This coefficient quantifies the dielectric susceptibility of the system in the presence of external electric perturbation.

The Code to Analyze the Data

Run the cell to load in two programs which process the HP 4192A LF impedance analyzer data recorded by the Labview Program, "Impendance Analyzer." The files are exported as ".txt" files, and the ".py" programs import the data into the IPython notebook. Using matplotlib and numpy modules, the data is parsed and filtered to create spectras for each temperature run. The second program plots all the data as a 3D surface

Each spectra is a record of the dielectric constant, ϵ'', against the frequency (in Hertz). The surface plot is this value at all temperatures and frequencies.

The impedance analyzer measures the magnitude of the complex impedance and the

Import the single temperature data to fit the curve and initialize

In [3]:
```
files = ['test35','test36']

test = np.loadtxt('Data03082013/'+files[0], dtype=float,delimiter='\t')
#magnitude
m = test.T[0]
#frequency
f = test.T[1]
#angle projected into imaginery/real plane
a = test.T[4]

test2 = np.loadtxt('Data03082013/'+files[1], dtype=float,delimiter='\t')
m2 = test2.T[0]
f2 = test2.T[1]
a2 = test2.T[4]
```

Combine the data into a 3D array

In [4]:
```
magn = np.concatenate((m,m2))
freq = np.concatenate((f,f2))
angl = np.concatenate((a,a2))
```

Conversion to Dielectric

Put the data in terms of the dielectric constant.

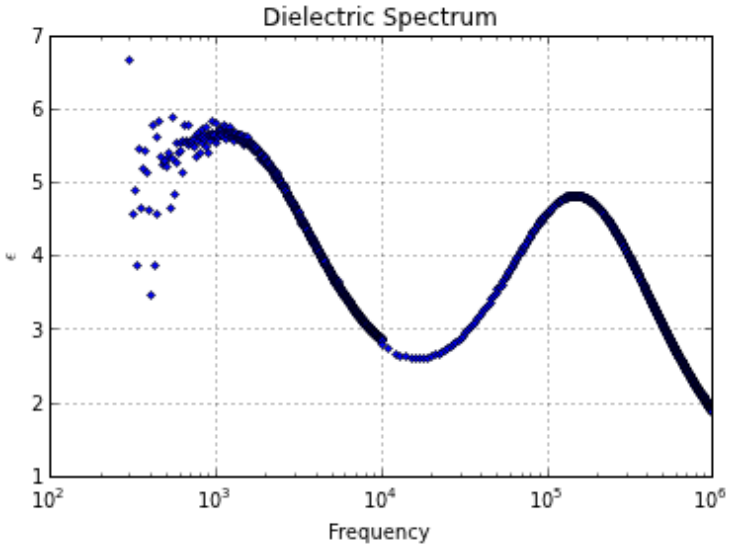
In [4]:
```
A = .01**2;
eps0 = 8.854187e-12;
d = 5e-6;
omega = 2*np.pi*freq;

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);
```

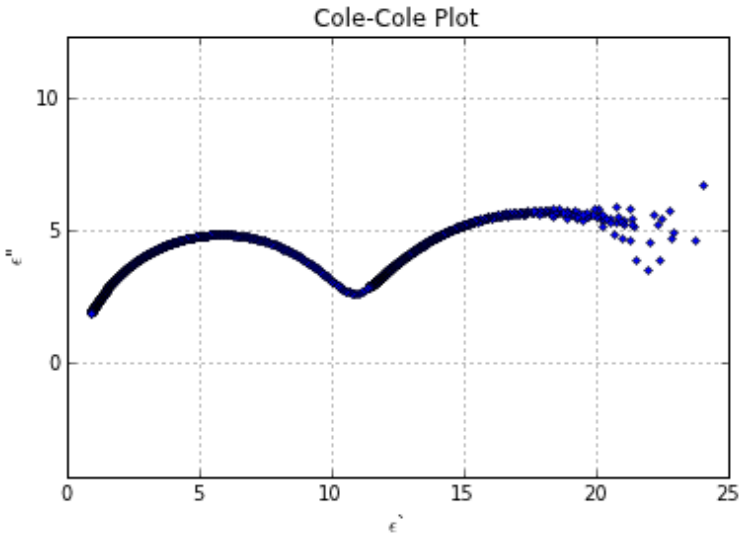
Plot the data using matplotlib

The first plot is the imaginery dielectric spectrum. The second plot is the cole-cole plot of the liquid crystal.

In [5]:
```
plt.figure(1)
semilogx(freq,e2,'.')
xlabel('Frequency')
ylabel('$\epsilon$')
title('Dielectric Spectrum')
grid()

plt.figure(3)
plot(e1,e2,'.')
xlabel('$\epsilon$`')
ylabel('$\epsilon$"')
title('Cole-Cole Plot')
grid()
axis('equal')
```

Out[5]: (0.0, 25.0, 1.0, 7.0)





Use fitted model to determine the dielectric parameters and susceptibility.

Example of one analysis process:

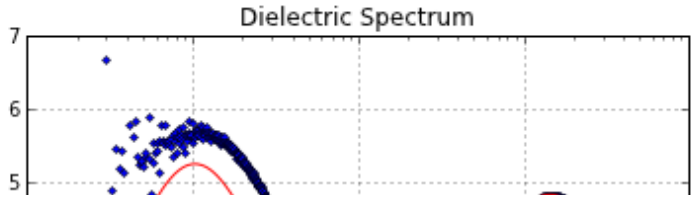
In [6]:
```
""" GUESSED VALUES for the fit of the primary peak in the dielectric data. """
E_infinity = 0;
Delta_E = 12;
Relaxation_frequency = 1000;
Beta = .9;
G_low_frequency = 0;

""" GUESSSED VALUES for the fit of the secondary peak in dielectric data. """
#E_infinity2 = 0;
Delta_E2 = 11;
Relaxation_frequency2 = 150000;
Beta2 = .9;
#G_low_frequency2 = 0;

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

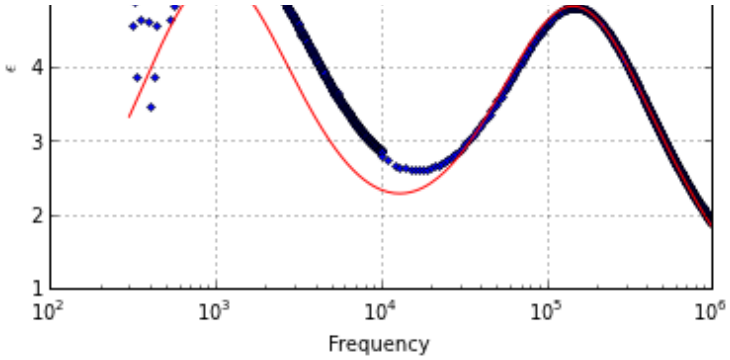
Model function to fit our data

In [7]:
```
def func(x, Einf,delE,relaxFreq,beta,g,delE2,relaxFreq2,beta2):
    z = (Einf + (delE/(1 + (1j*(x/relaxFreq))**(beta))) + g + (delE2/(1 + (1j*
(x/relaxFreq2))**(beta2))))
    return -(z.imag)
```

Display our 'guessed' model to avoid timeout in curve_fit function

In [8]:
```
guess = func(x,E_infinity, Delta_E, Relaxation_frequency, Beta, G_low_frequency,
Delta_E2, Relaxation_frequency2, Beta2)
plt.figure(1)
semilogx(freq,e2,'.')
semilogx(freq,guess,'r')
xlabel('Frequency')
ylabel('$\epsilon$')
title('Dielectric Spectrum')
grid()
```



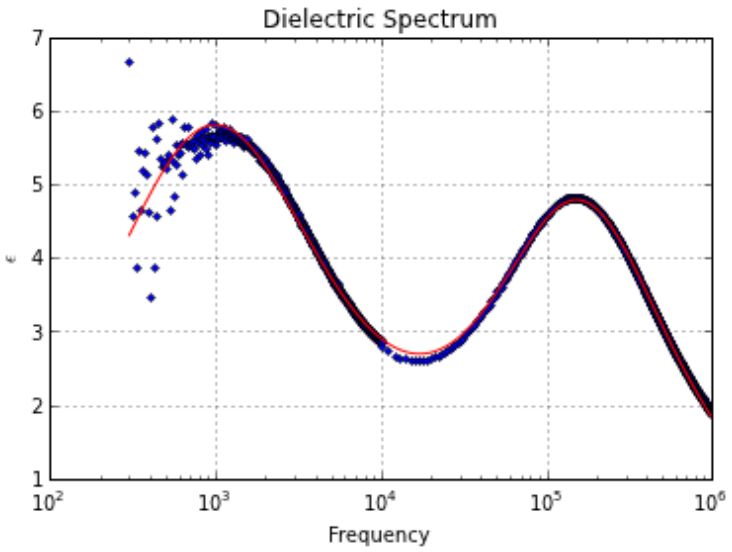


Use scipy's 'curve_fit' function to fit our model to the data.

Print out the values of the fittedmodel and plot the model over the data.

In [9]:
```
popt, pcov = spo.curve_fit(func,x,y,[E_infinity, Delta_E, Relaxation_frequency,
Beta, G_low_frequency, Delta_E2, Relaxation_frequency2, Beta2])
fittedmodel =
func(x,popt[0],popt[1],popt[2],popt[3],popt[4],popt[5],popt[6],popt[7])

def fit_results(param,values):
    for i in range(0,8,1):
        print param[i] + str(values[i])

params = ["\nRESULTS FOR PRIMARY PEAK: \n\nE_infinity: ", "Delta_E: ",
"Relaxation_frequency: ", "Beta: ", "G_low_frequency: ",
          "\nRESULTS FOR SECONDARY PEAK: \n\nE_infinity: ", "Delta_E: ",
"Relaxation_frequency: ", "Beta: ", "G_low_frequency: "];

fit_results(params,popt)

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

```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 16.1055269206
Relaxation_frequency: 961.090887303
Beta: 0.785402730347
G_low_frequency: 0.0

RESULTS FOR SECONDARY PEAK:

E_infinity: 10.2428329901
Delta_E: 162243.20224

Relaxation_frequency: 0.919287227143
```



In [ ]:

Back to top

Data Analysis of TSiKN65 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 [167]:
```
%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],filenames[1]]

    w = np.loadtxt('Data03082013/weights.txt', dtype=float,delimiter='\t')
    w = w.T
  #  w = w[100:]
    test = np.loadtxt('Data03082013/'+files[0], dtype=float,delimiter='\t')
    #magnitude
    m = test.T[0]
  #  m = m[100:]
    #frequency
    f = test.T[1]
  #  f = f[100:]
    #angle projected into imaginery/real plane
    a = test.T[4]
  #  a = a[100:]

    test2 = np.loadtxt('Data03082013/'+files[1], dtype=float,delimiter='\t')
    m2 = test2.T[0]
    f2 = test2.T[1]
    a2 = test2.T[4]

    """ Combine the data into a 3D array """
    magn = np.concatenate((m,m2))
    freq = np.concatenate((f,f2))
    angl = np.concatenate((a,a2))

    """ Put data in terms of the dielectric constant """
    A = .01**2;
    eps0 = 8.854187e-12;
    d = 3e-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];

    """ GUESSSED VALUES for the fit of the secondary peak in dielectric data. """
    E_infinity2 = guess[5];
    Delta_E2 = guess[6];
    Relaxation_frequency2 = guess[7];
    Beta2 = guess[8];
    G_low_frequency2 = guess[9];

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

    """ Model Function to fit our data """
    def func(x, Einf,delE,relaxFreq,beta,g,Einf2,delE2,relaxFreq2,beta2,g2):
        z = (Einf + (delE/(1 + (1j*(x/relaxFreq))**(beta))) + g + Einf2 + (delE2/(1 +
(1j*(x/relaxFreq2))**(beta2))) + g2)
        return -(z.imag)

    dummy = func(x,E_infinity, Delta_E, Relaxation_frequency, Beta, G_low_frequency,
E_infinity2, Delta_E2, Relaxation_frequency2, Beta2, G_low_frequency2)

    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, E_infinity2, Delta_E2, Relaxation_frequency2, Beta2,
G_low_frequency2], sigma=w, maxfev=10000)

    fittedmodel =
func(x,popt[0],popt[1],popt[2],popt[3],popt[4],popt[5],popt[6],popt[7],popt[8],popt[9])
```

```
    if bool_to_print == True:
        def fit_results(param,values):
            for i in range(0,10,1):
                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: ",
                  "\nRESULTS FOR SECONDARY 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((0,16))
        grid()

    return popt, fittedmodel
```

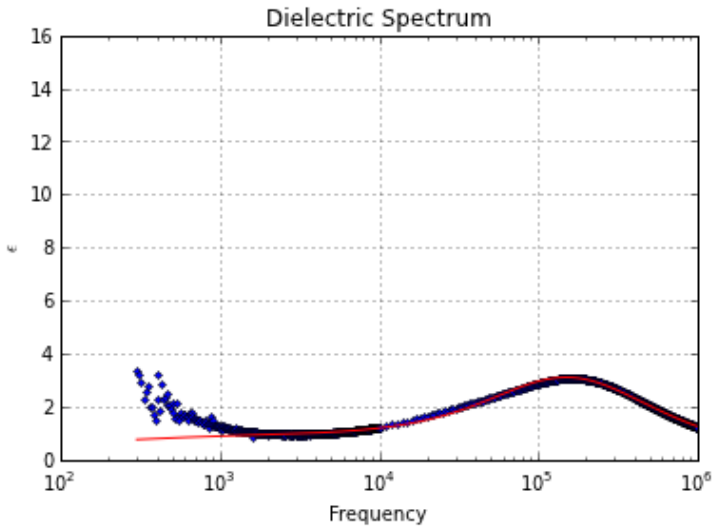
```
Welcome to pylab, a matplotlib-based Python environment [backend:
module://IPython.kernel.zmq.pylab.backend_inline].
For more information, type 'help(pylab)'.
```

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

In [168]:
```
""" GUESSED VALUES for the fit of the primary peak in the dielectric data. """
E_infinity = 0;
Delta_E = 15;
Relaxation_frequency = 2500;
Beta = .5;
G_low_frequency = 0;

""" GUESSSED VALUES for the fit of the secondary peak in dielectric data. """
E_infinity2 = 0;
Delta_E2 = 10;
Relaxation_frequency2 = 150000;
Beta2 = .9;
G_low_frequency2 = 0;

guess = [E_infinity, Delta_E, Relaxation_frequency, Beta, G_low_frequency,
E_infinity2, Delta_E2, Relaxation_frequency2, Beta2, G_low_frequency2]

popt1, fittedmodel1 = data_extract(['test01','test02'], guess, False, True, True)
```

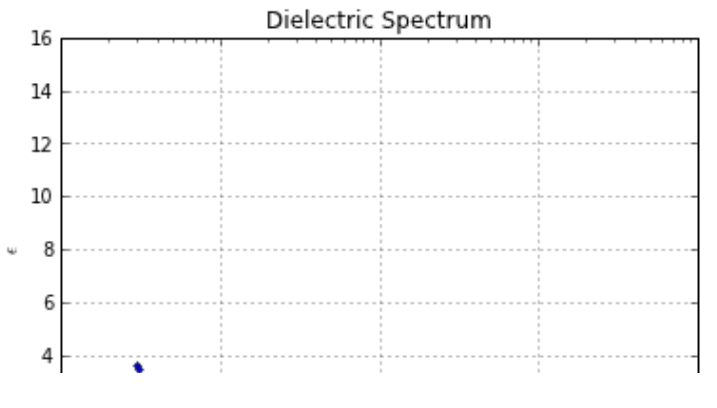
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 4.83484789508
```

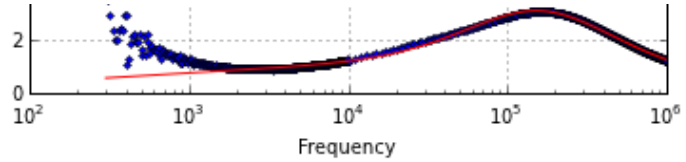
```
Relaxation_frequency: -1451.21798264
Beta: -0.421719282528
G_low_frequency: 0.0

RESULTS FOR SECONDARY PEAK:

E_infinity: 0.0
Delta_E: 6.3605015633
Relaxation_frequency: 158203.197642
Beta: 0.911964316778
G_low_frequency: 0.0
```



In [169]:
```
popt2, fittedmodel2 = data_extract(['test03','test04'], guess, False, True, True)
```

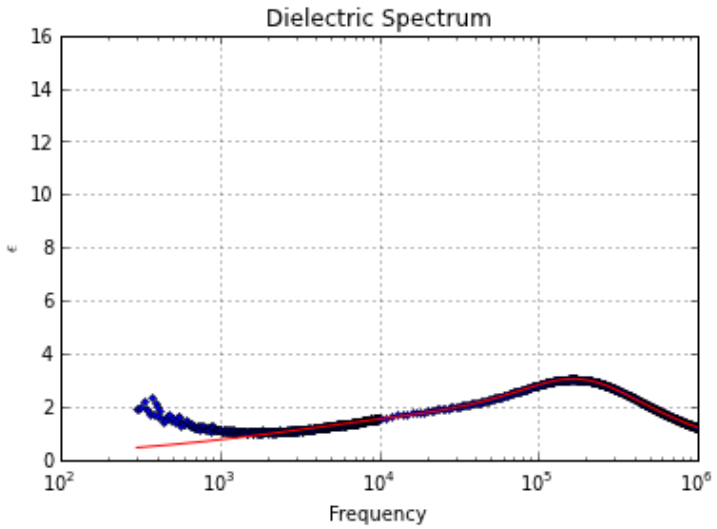
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 4.14723330416
Relaxation_frequency: 4071.36512592
Beta: 0.450793712629
G_low_frequency: 0.0

RESULTS FOR SECONDARY PEAK:

E_infinity: 0.0
Delta_E: 6.11989640352
Relaxation_frequency: 161271.846496
Beta: 0.920898335413
G_low_frequency: 0.0
```





In [170]:
```
popt3, fittedmodel3 = data_extract(['test05','test06'], guess, False, True, True)
```

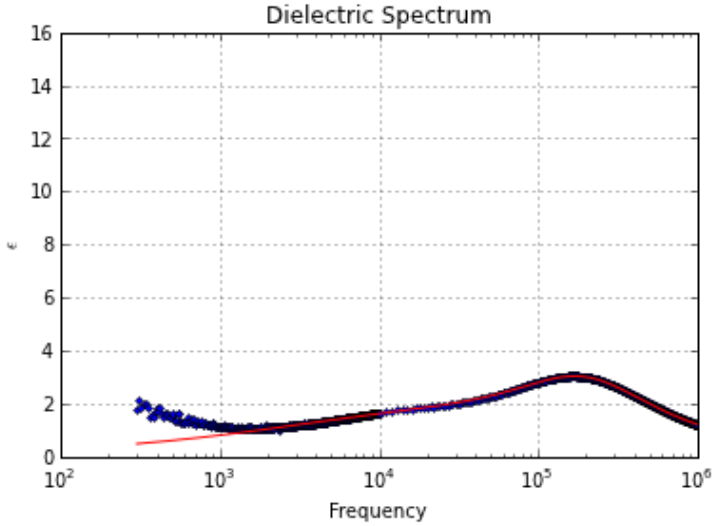
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 6.35539281559
Relaxation_frequency: 19862.0192039
Beta: 0.521108220475
G_low_frequency: 0.0

RESULTS FOR SECONDARY PEAK:

E_infinity: 0.0
Delta_E: 4.00329561371
Relaxation_frequency: 188770.167133
Beta: 1.02362490918
G_low_frequency: 0.0
```



In [171]:
```
popt4, fittedmodel4 = data_extract(['test07','test08'], guess, False, True, True)
```

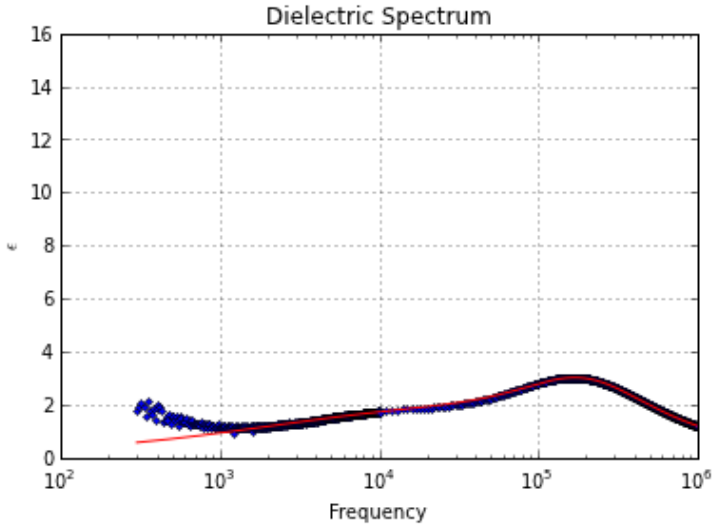
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 6.96647539597
Relaxation_frequency: 20599.3715717
Beta: 0.523842209073
G_low_frequency: 0.0

RESULTS FOR SECONDARY PEAK:

E_infinity: 0.0
Delta_E: 3.6519028785
Relaxation_frequency: 193801.66922
Beta: 1.04837737827
```

```
G_low_frequency: 0.0
```



In [172]:
```
popt5, fittedmodel5 = data_extract(['test09','test10'], guess, False, True, True)
```

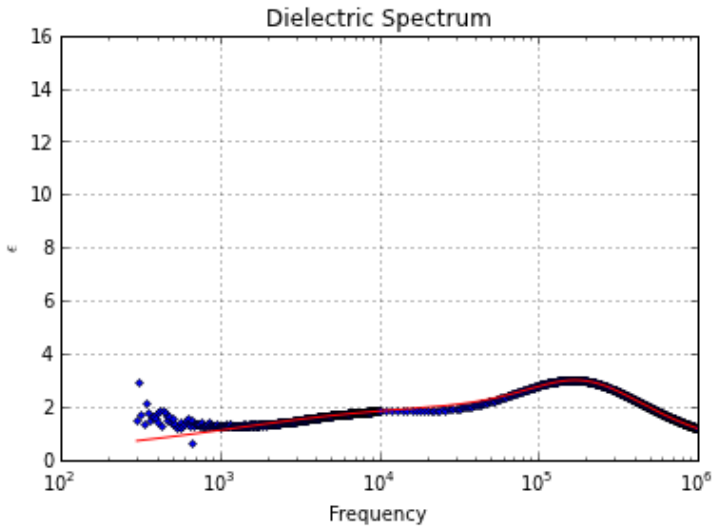
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 7.31835143472
Relaxation_frequency: 15980.8124522
Beta: 0.521151869392
G_low_frequency: 0.0

RESULTS FOR SECONDARY PEAK:

E_infinity: 0.0
Delta_E: 3.69971372194
Relaxation_frequency: 195939.540272
Beta: 1.04665821285
G_low_frequency: 0.0
```



In [173]:
```
popt6, fittedmodel6 = data_extract(['test11','test12'], guess, False, True, True)
```

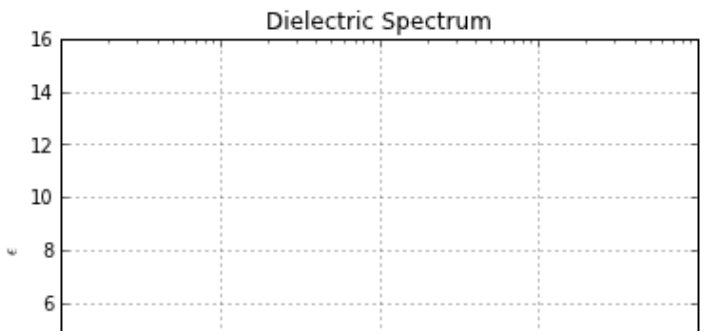
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
```

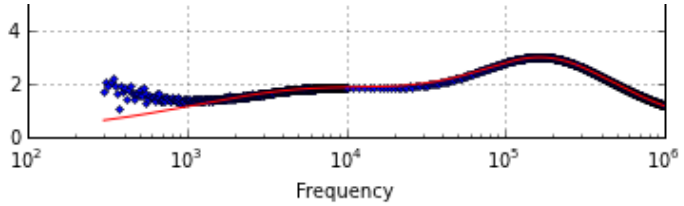
```
Delta_E: 7.85188060788
Relaxation_frequency: 11553.3064374
Beta: 0.51130424142
G_low_frequency: 0.0

RESULTS FOR SECONDARY PEAK:

E_infinity: 0.0
Delta_E: 3.7294122309
Relaxation_frequency: 196884.459867
Beta: 1.04819996081
G_low_frequency: 0.0
```



In [174]:
```
popt7, fittedmodel7 = data_extract(['test13','test14'], guess, False, True, True)
```

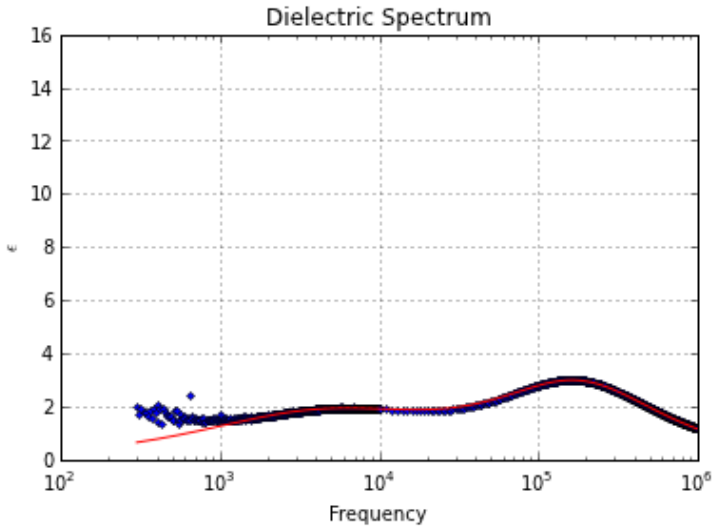
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 5.6516419668
Relaxation_frequency: 5532.25346757
Beta: 0.66331941147
G_low_frequency: 0.0

RESULTS FOR SECONDARY PEAK:

E_infinity: 0.0
Delta_E: 5.40776835736
Relaxation_frequency: 185759.82899
Beta: 0.96198871258
G_low_frequency: 0.0
```





In [175]:
```
popt8, fittedmodel8 = data_extract(['test15','test16'], guess, False, True, True)
```

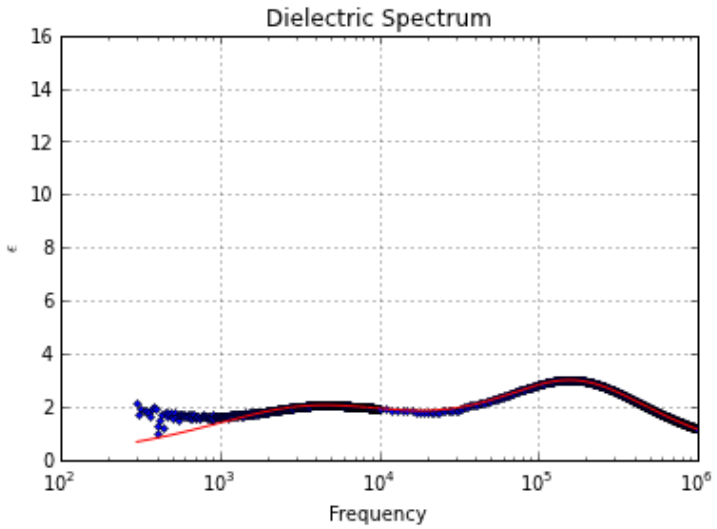
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 5.49236136069
Relaxation_frequency: 4389.63835011
Beta: 0.719688508668
G_low_frequency: 0.0

RESULTS FOR SECONDARY PEAK:

E_infinity: 0.0
Delta_E: 5.73904900033
Relaxation_frequency: 180216.427906
Beta: 0.94980055765
G_low_frequency: 0.0
```



In [176]:
```
popt9, fittedmodel9 = data_extract(['test17','test18'], guess, False, True, True)
```

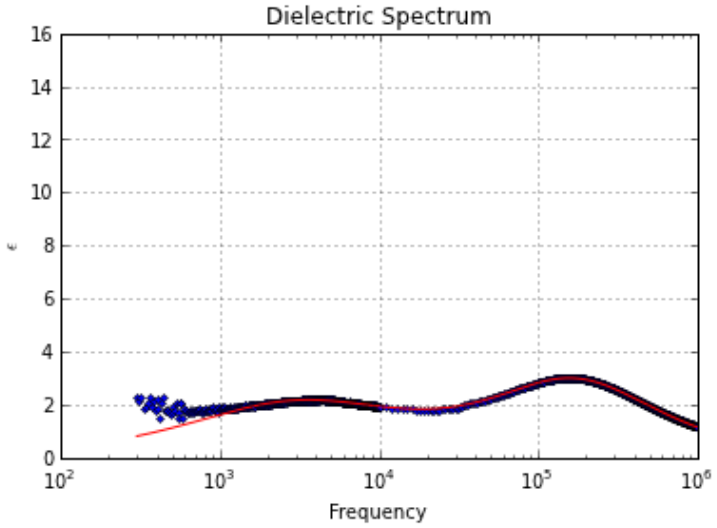
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 5.39744414112
Relaxation_frequency: 3636.26154669
Beta: 0.772699443391
G_low_frequency: 0.0

RESULTS FOR SECONDARY PEAK:

E_infinity: 0.0

Delta_E: 6.07305303797
```

```
Relaxation_frequency: 171430.642555
Beta: 0.9352542898
G_low_frequency: 0.0
```



In [177]:
```
popt10, fittedmodel10 = data_extract(['test19','test20'], guess, False, True, True)
```

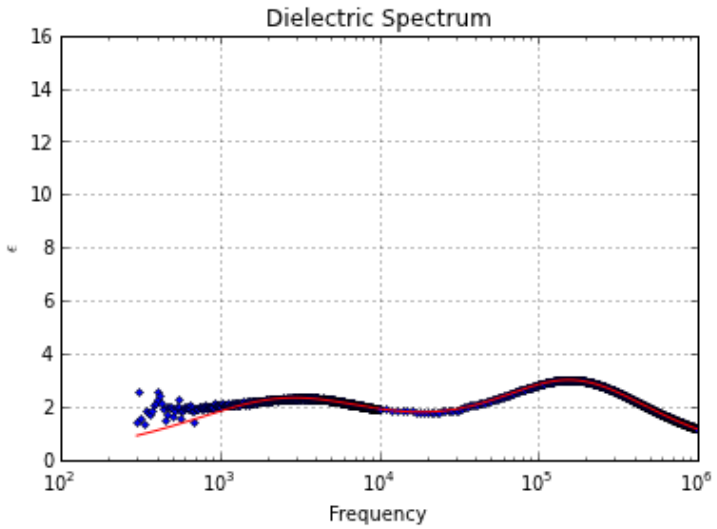
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 5.80146987443
Relaxation_frequency: 3026.42871102
Beta: 0.77620720075
G_low_frequency: 0.0

RESULTS FOR SECONDARY PEAK:

E_infinity: 0.0
Delta_E: 6.14303439883
Relaxation_frequency: 169885.392647
Beta: 0.932500124397
G_low_frequency: 0.0
```



In [178]:
```
popt11, fittedmodel11 = data_extract(['test21','test22'], guess, False, True, True)
```

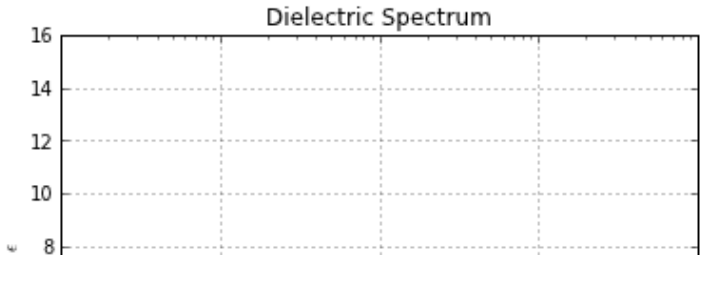
```
RESULTS FOR PRIMARY PEAK:
```

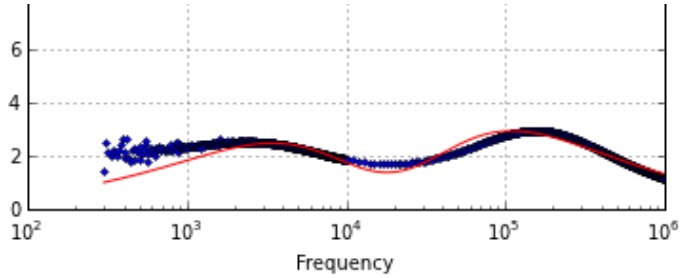
```
E_infinity: 0.0
Delta_E: 6.0155731992
Relaxation_frequency: 2602.20752991
Beta: 0.802478164365
G_low_frequency: 0.0

RESULTS FOR SECONDARY PEAK:

E_infinity: 0.0
Delta_E: 6.28064508946
Relaxation_frequency: 167125.122212
Beta: 0.92715827509
G_low_frequency: 0.0
```



In [203]:
```
popt12, fittedmodel12 = data_extract(['test23','test24'], guess, False, True, True)
```

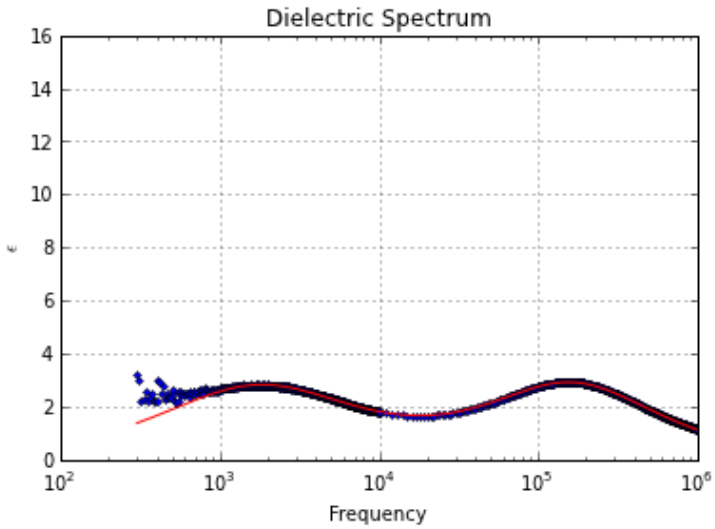
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 1.00001657305
Delta_E: -4973.29944897
Relaxation_frequency: 20390.6290499
Beta: 0.792386872048
G_low_frequency: 1.00001375054

RESULTS FOR SECONDARY PEAK:

E_infinity: 1.00001347511
Delta_E: 4986.13504084
Relaxation_frequency: 20398.6455949
Beta: 0.791300164622
G_low_frequency: 1.00001272438
```





In [180]:
```
popt13, fittedmodel13 = data_extract(['test25','test26'], guess, False, True, True)
```

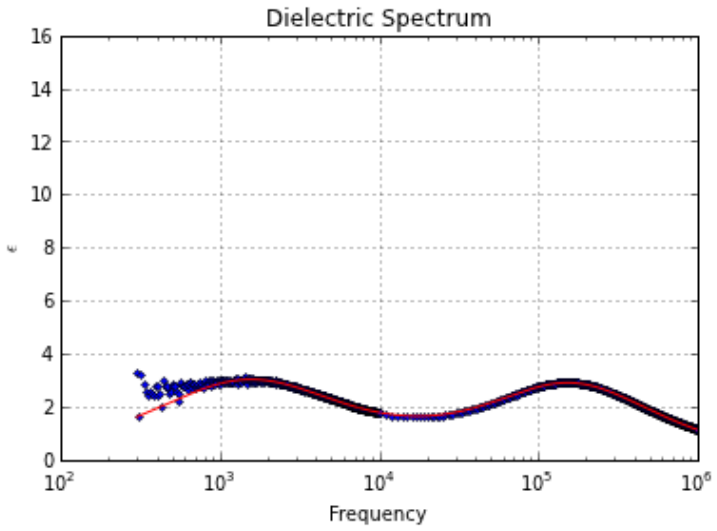
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 7.08990980696
Relaxation_frequency: 1724.32696549
Beta: 0.8357130783
G_low_frequency: 0.0

RESULTS FOR SECONDARY PEAK:

E_infinity: 0.0
Delta_E: 6.27489342192
Relaxation_frequency: 163957.678057
Beta: 0.919923648869
G_low_frequency: 0.0
```



In [181]:
```
popt14, fittedmodel14 = data_extract(['test27','test28'], guess, False, True, True)
```

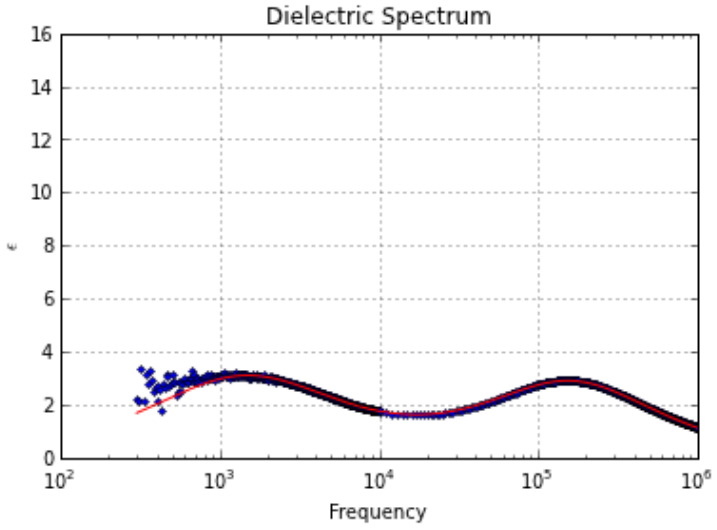
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 7.56690742034
Relaxation_frequency: 1491.87829764
Beta: 0.840289568025
G_low_frequency: 0.0

RESULTS FOR SECONDARY PEAK:

E_infinity: 0.0
```

```
Delta_E: 6.31754123994
Relaxation_frequency: 162386.339084
Beta: 0.913117910437
G_low_frequency: 0.0
```



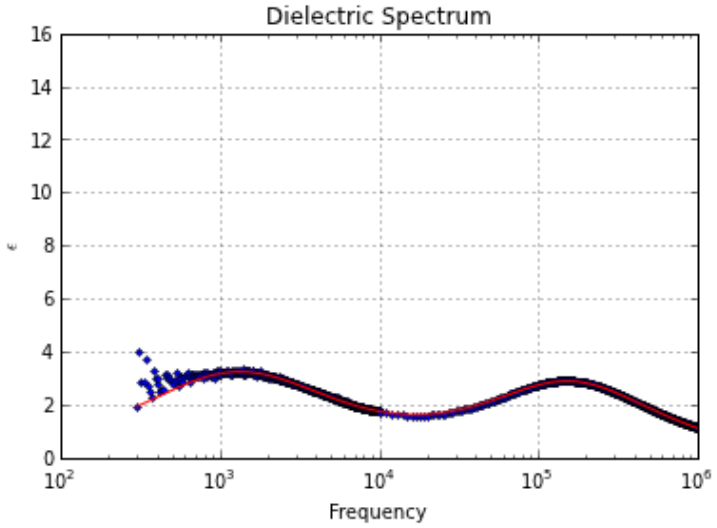
In [182]:
```
popt15, fittedmodel15 = data_extract(['test29','test30'], guess, False, True, True)
```

```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 7.69702571271
Relaxation_frequency: 1426.11365889
Beta: 0.84623533911
G_low_frequency: 0.0

RESULTS FOR SECONDARY PEAK:

E_infinity: 0.0
Delta_E: 6.34415766174
Relaxation_frequency: 161775.68699
Beta: 0.912068140767
G_low_frequency: 0.0
```



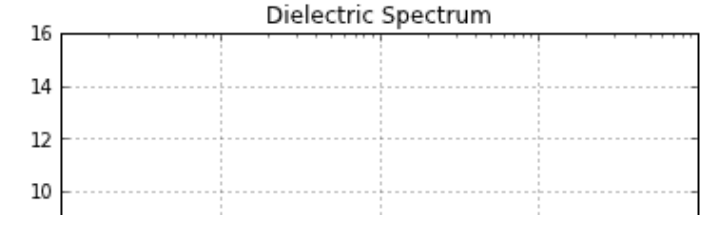
In [183]:
```
popt16, fittedmodel16 = data_extract(['test31','test32'], guess, False, True, True)
```

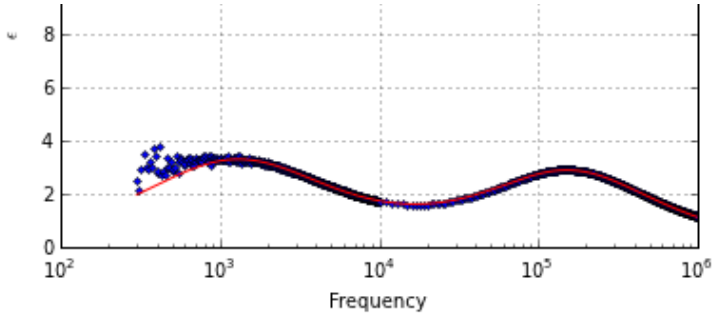
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 8.196919553
Relaxation_frequency: 1279.40845464
Beta: 0.836193795887
G_low_frequency: 0.0

RESULTS FOR SECONDARY PEAK:

E_infinity: 0.0
Delta_E: 6.3041159349
Relaxation_frequency: 160432.526906
Beta: 0.911846178456
G_low_frequency: 0.0
```



In [184]:
```
popt17, fittedmodel17 = data_extract(['test33','test34'], guess, False, True, True)
```

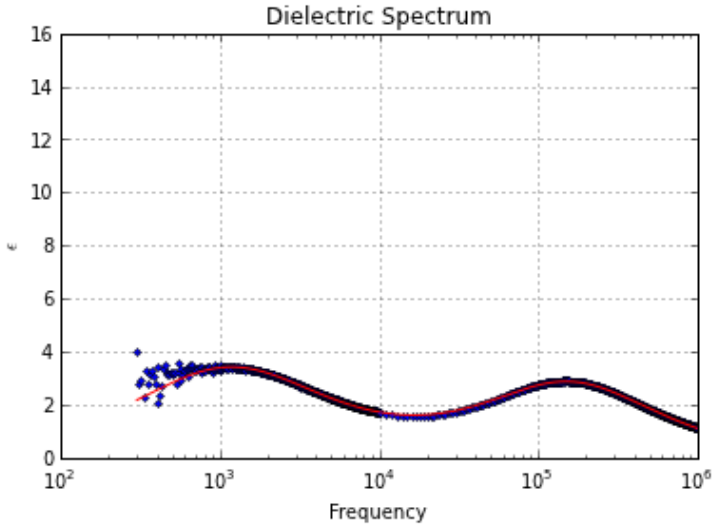
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 8.23158582709
Relaxation_frequency: 1252.61984816
Beta: 0.844877397694
G_low_frequency: 0.0

RESULTS FOR SECONDARY PEAK:

E_infinity: 0.0
Delta_E: 6.32962845714
Relaxation_frequency: 159905.222636
Beta: 0.911072474559
G_low_frequency: 0.0
```





In [185]:
```
popt18, fittedmodel18 = data_extract(['test35','test36'], guess, False, True, True)
```

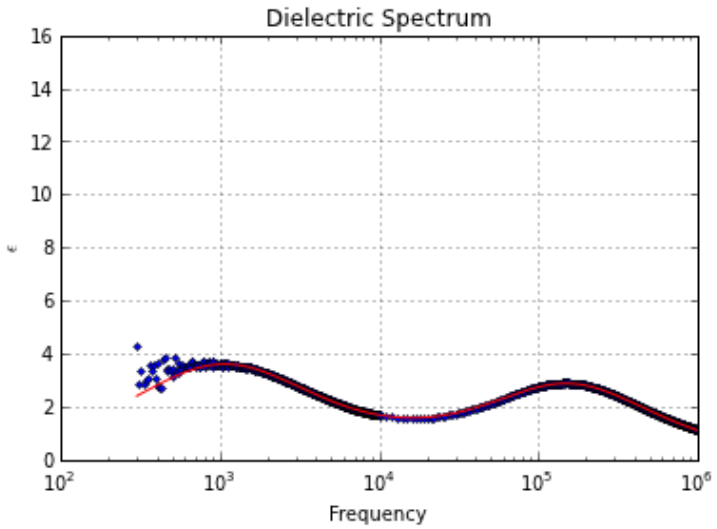
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 8.64142357285
Relaxation_frequency: 1139.74394489
Beta: 0.838668250134
G_low_frequency: 0.0

RESULTS FOR SECONDARY PEAK:

E_infinity: 0.0
Delta_E: 6.32403113073
Relaxation_frequency: 158093.878053
Beta: 0.908198135573
G_low_frequency: 0.0
```



In [186]:
```
popt19, fittedmodel19 = data_extract(['test37','test38'], guess, False, True, True)
```

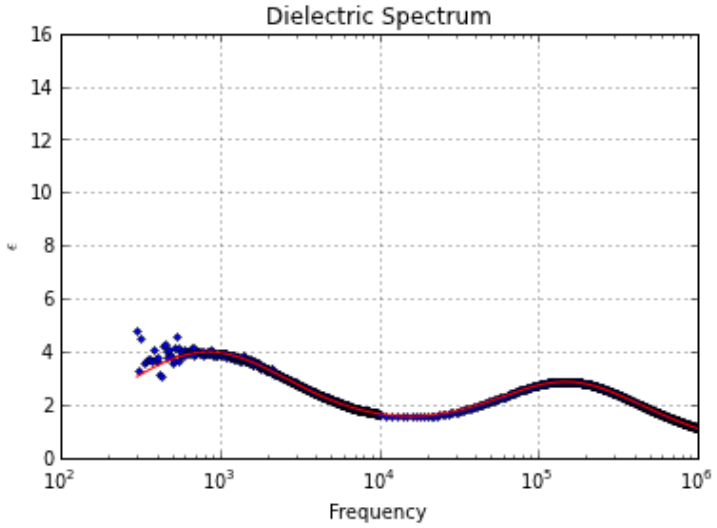
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 9.03181191628
Relaxation_frequency: 1036.30596044
Beta: 0.845716886433
G_low_frequency: 0.0

RESULTS FOR SECONDARY PEAK:
```

```
E_infinity: 0.0
Delta_E: 6.36117099108
Relaxation_frequency: 157230.914157
Beta: 0.906738713181
G_low_frequency: 0.0
```



In [187]:
```
popt20, fittedmodel20 = data_extract(['test39','test40'], guess, False, True, True)
```

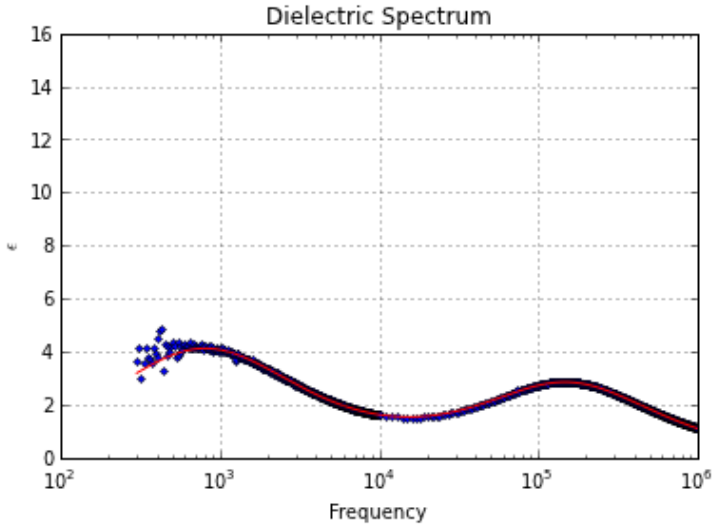
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 10.2574754304
Relaxation_frequency: 822.942258078
Beta: 0.832950325059
G_low_frequency: 0.0

RESULTS FOR SECONDARY PEAK:

E_infinity: 0.0
Delta_E: 6.35665452493
Relaxation_frequency: 156192.661725
Beta: 0.904040527225
G_low_frequency: 0.0
```



In [188]:
```
popt21, fittedmodel21 = data_extract(['test41','test42'], guess, False, True, True)
```

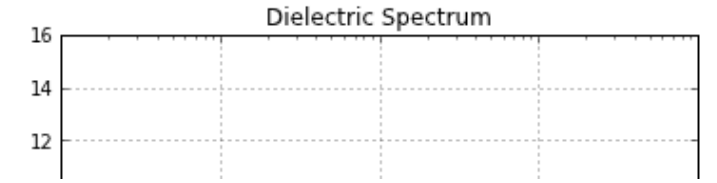
```
popt21, fittedmodel21 = data_extract(['test41','test42'], guess, False, True, True)
```

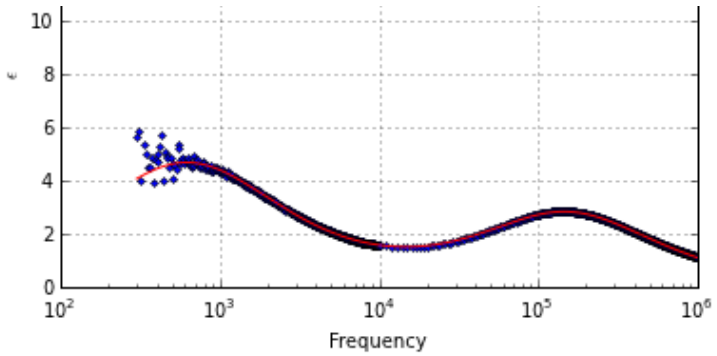
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 10.3881134949
Relaxation_frequency: 788.460717548
Beta: 0.845099869696
G_low_frequency: 0.0

RESULTS FOR SECONDARY PEAK:

E_infinity: 0.0
Delta_E: 6.38511369177
Relaxation_frequency: 154616.754588
Beta: 0.900727682047
G_low_frequency: 0.0
```



In [189]:
```
popt22, fittedmodel22 = data_extract(['test43','test44'], guess, False, True, True)
```

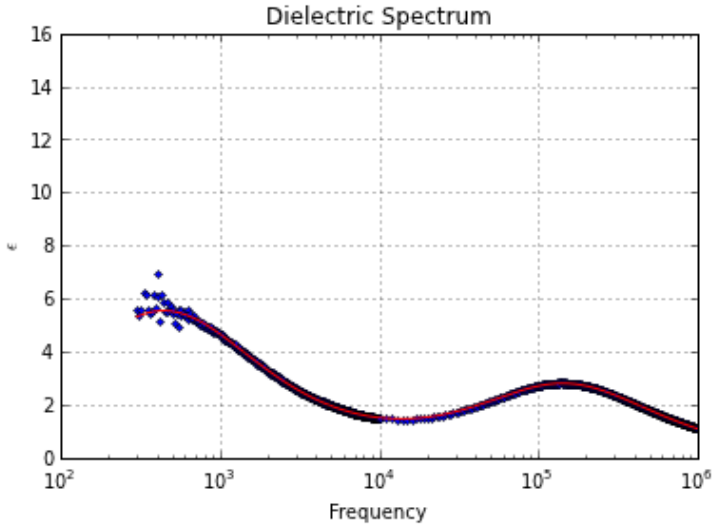
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 11.9141525757
Relaxation_frequency: 608.062391222
Beta: 0.841899302264
G_low_frequency: 0.0

RESULTS FOR SECONDARY PEAK:

E_infinity: 0.0
Delta_E: 6.40466699701
Relaxation_frequency: 152504.05802
Beta: 0.895510032412
G_low_frequency: 0.0
```





In [190]:
```
popt23, fittedmodel23 = data_extract(['test45','test46'], guess, False, True, True)
```

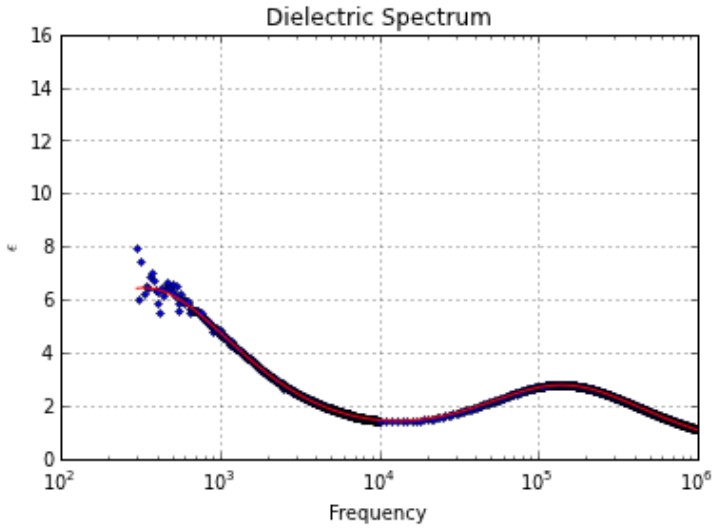
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 14.1787976274
Relaxation_frequency: 438.250740526
Beta: 0.840356693464
G_low_frequency: 0.0

RESULTS FOR SECONDARY PEAK:

E_infinity: 0.0
Delta_E: 6.44886575404
Relaxation_frequency: 149436.53564
Beta: 0.886907015655
G_low_frequency: 0.0
```



In [191]:
```
popt24, fittedmodel24 = data_extract(['test47','test48'], guess, False, True, True)
```

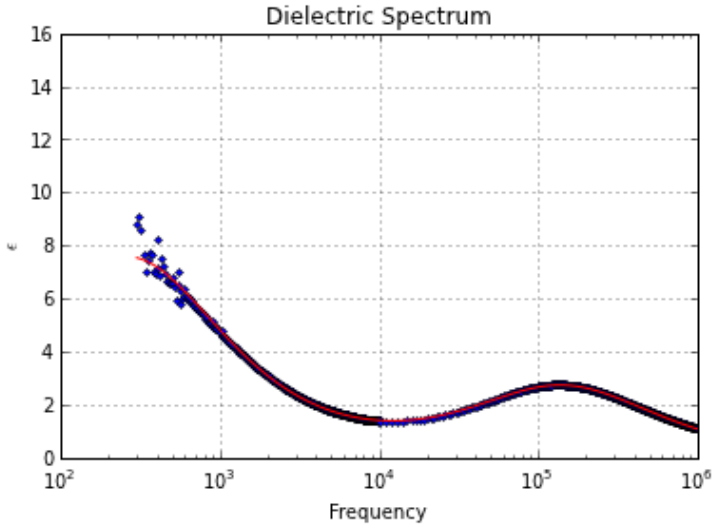
```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 16.4636981851
Relaxation_frequency: 333.175710056
Beta: 0.842160072259
G_low_frequency: 0.0

RESULTS FOR SECONDARY PEAK:
```

```
E_infinity: 0.0
Delta_E: 6.46714937195
Relaxation_frequency: 146644.776785
Beta: 0.880059703238
G_low_frequency: 0.0
```



In [192]:
```
popt25, fittedmodel25 = data_extract(['test49','test50'], guess, False, True, True)
```

```
RESULTS FOR PRIMARY PEAK:

E_infinity: 0.0
Delta_E: 18.994058014
Relaxation_frequency: -261.228077435
Beta: -0.854719377442
G_low_frequency: 0.0

RESULTS FOR SECONDARY PEAK:

E_infinity: 0.0
Delta_E: 6.51625635892
Relaxation_frequency: 143105.438608
Beta: 0.869370021032
G_low_frequency: 0.0
```

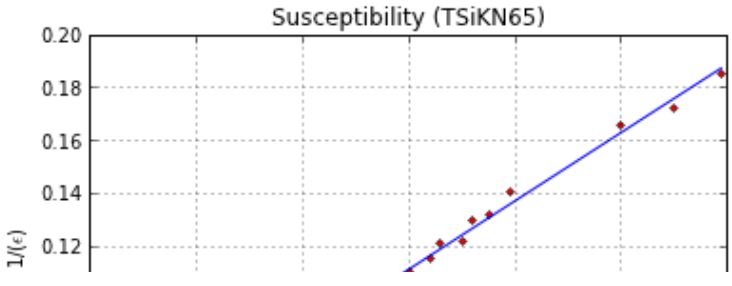


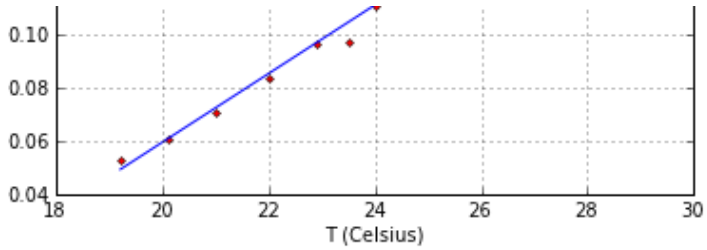
In [200]:
```
#epsilon = [popt1[1],popt2[1],popt3[1],popt4[1],popt5[1],popt6[1],popt7[1],popt8[1],
epsilon = [popt9[1],
popt10[1],popt11[1],popt12[1],popt13[1],popt14[1],popt15[1],popt16[1],popt17[1],popt18[1],

           popt19[1],popt20[1],popt21[1],popt22[1],popt23[1],popt24[1],popt25[1]]

#reson = [popt1[2],popt2[2],popt3[2],popt4[2],popt5[2],popt6[2],popt7[2],popt8[2],
reson = [popt9[2],
popt10[2],popt11[2],popt12[2],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]]

#temperature =  [40.5, 39.5, 36.1, 35.1, 34.1, 33, 31.9, 30.9,
temperature = [ 29.9, 29, 28, 27, 25.9, 25.5, 25.2,
               25, 24.6, 24.4, 24, 23.5]
#, 22.9, 22, 21, 20.1, 19.2]
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 [211]:
```
def susc_func(x,A,B,gamma):
    s = A*np.power(x,gamma) + B
    return s

susc, l = spo.curve_fit(susc_func, temperature ,epsilon, p0=[1,0,1], maxfev=10000)
print "The value of Gamma is: " + str(susc[2])
print "The value of m is: " + str(susc[0])
print "The value of B is: " + str(susc[1])
fittedmodel = susc_func(temperature, susc[0],susc[1],susc[2])

plt.figure(2)
plot(T,epsilon,'r.')
plot(T,fittedmodel,'b')
xlabel('T (Celsius)')
ylabel('1/($\epsilon$)')
title('Susceptibility (TSiKN65)')
grid()
```

```
The value of Gamma is: 0.987314830011
The value of m is: 0.0135955511085
The value of B is: -0.202210014365
```





In [195]:
```
susc2, l2 = spo.curve_fit(susc_func, temperature ,reson, p0=[.000007,-.5,3.9])
print "The value of Gamma is: " + str(susc2[2])
print "The value of m is: " + str(susc2[0])
fittedmodel = susc_func(temperature, susc2[0],susc2[1],susc2[2])
```

```
The value of Gamma is: 4.10918331399
The value of m is: 3.69058550296e-06
```

Back to top

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 """
```

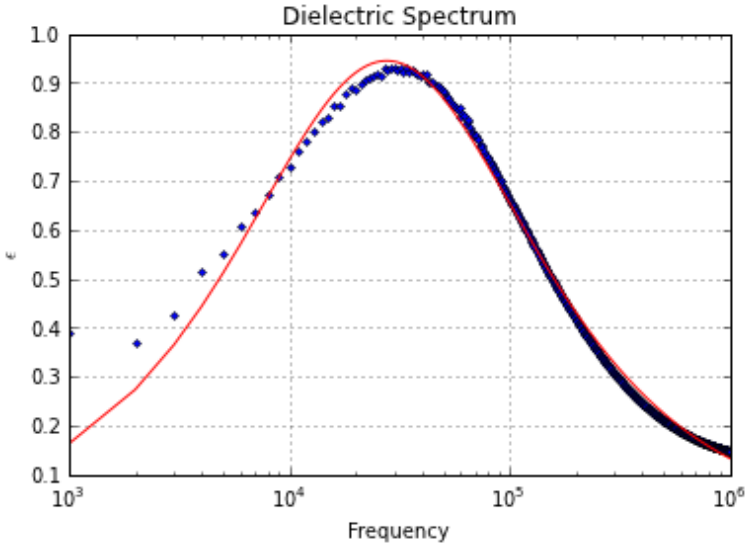
```
        plt.figure(2)
        semilogx(freq,e2,'.')
        semilogx(freq,fittedmodel,'r')
        xlabel('Frequency')
        ylabel('$\epsilon$')
        title('Dielectric Spectrum')
        #ylim((0,1))
        grid()

    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
```

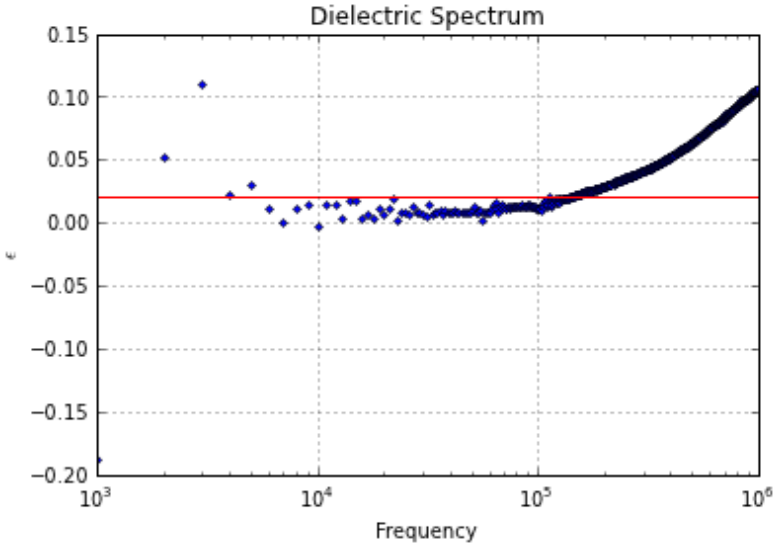


In [315]:
```
popt2, fittedmodel2 = data_extract(['test02'], guess, False, True, True)
```

```
RESULTS FOR PRIMARY PEAK:
```

RESULTS FOR PRIMARY PEAK:

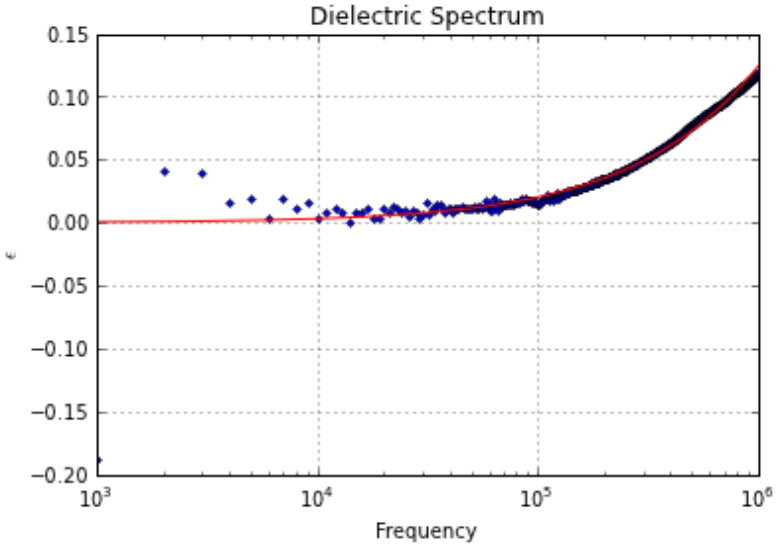
E_infinity: 0.0
Delta_E: 118.998295987
Relaxation_frequency: 46.9818041045
Beta: 0.000443506948068
G_low_frequency: 0.0



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

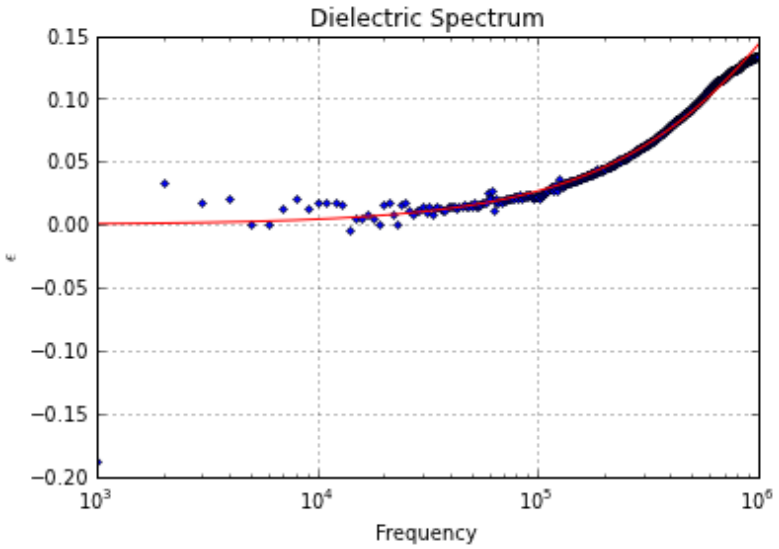


In [317]:
```
popt4, fittedmodel4 = data_extract(['test04'], guess, False, True, True)
```

RESULTS FOR PRIMARY PEAK:

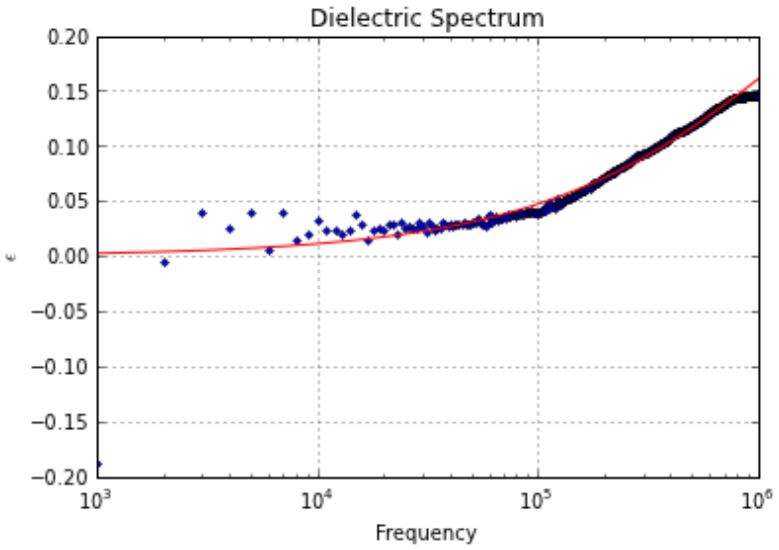
E_infinity: 0.0

```
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)
```

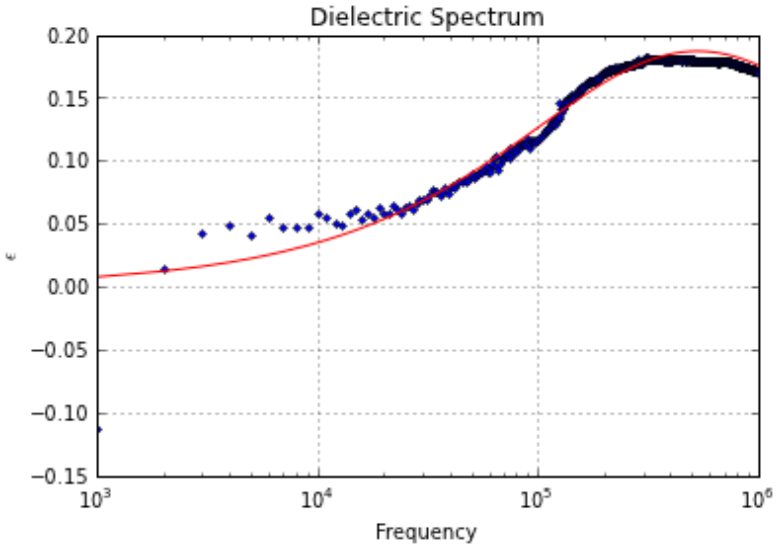
```
RESULTS FOR PRIMARY PEAK:

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)
```

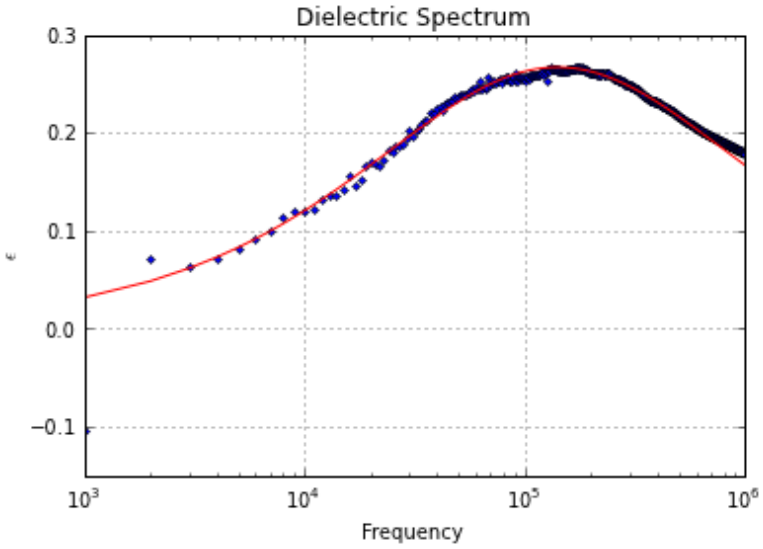
```
RESULTS FOR PRIMARY PEAK:

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

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



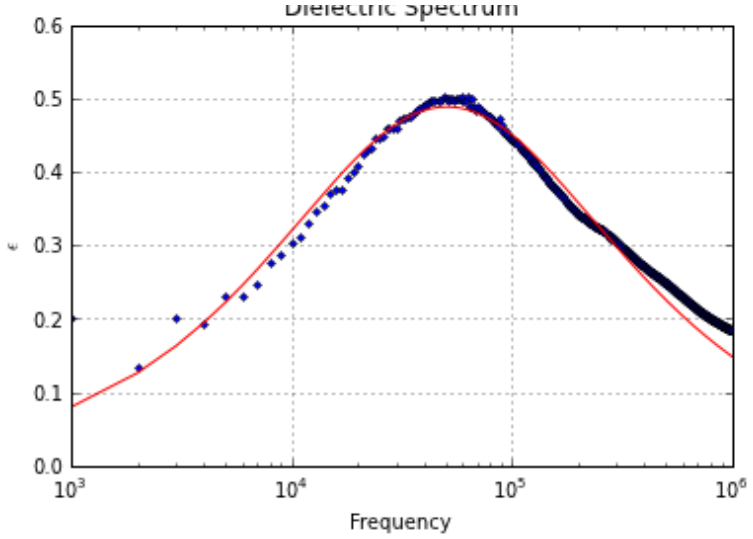
In [320]:
```
popt7, fittedmodel7 = data_extract(['test07'], guess, False, True, True)
```

```
RESULTS FOR PRIMARY PEAK:

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



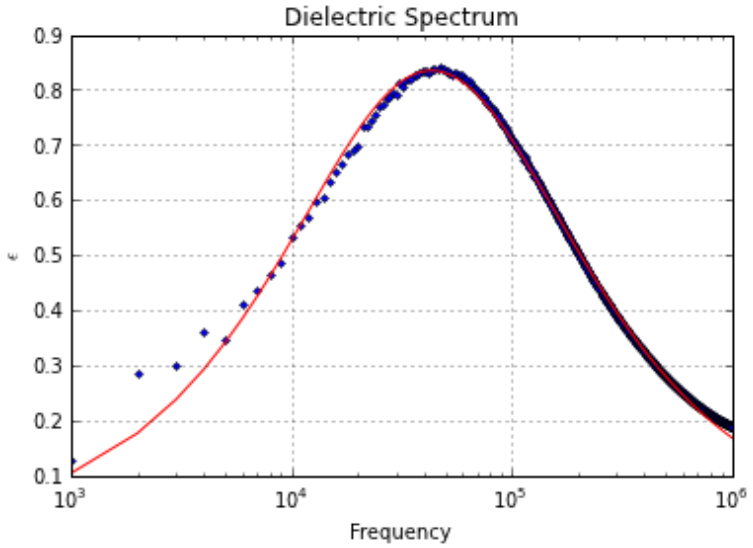
In [321]:
```
popt8, fittedmodel8 = data_extract(['test08'], guess, False, True, True)
```

```
RESULTS FOR PRIMARY PEAK:

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



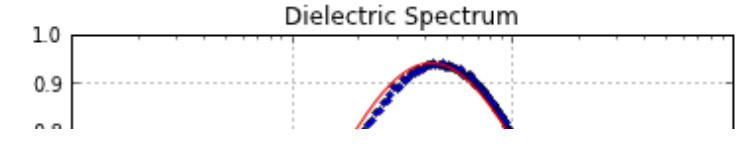
In [322]:
```
popt9, fittedmodel9 = data_extract(['test09'], guess, False, True, True)
```

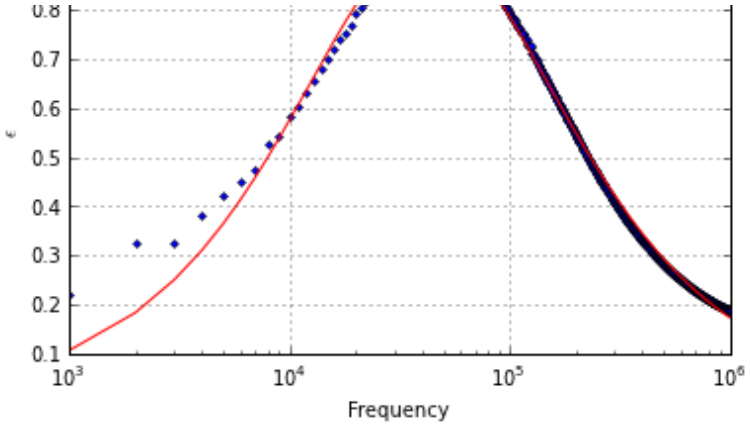
```
RESULTS FOR PRIMARY PEAK:

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)
```

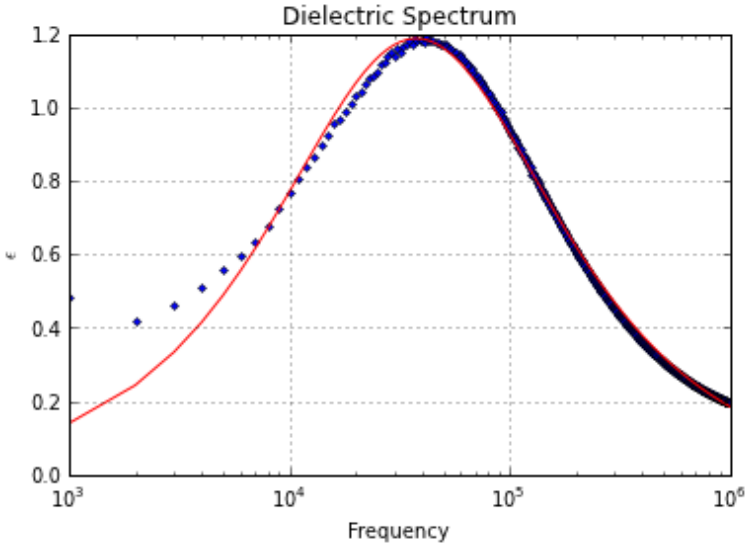
```
RESULTS FOR PRIMARY PEAK:

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





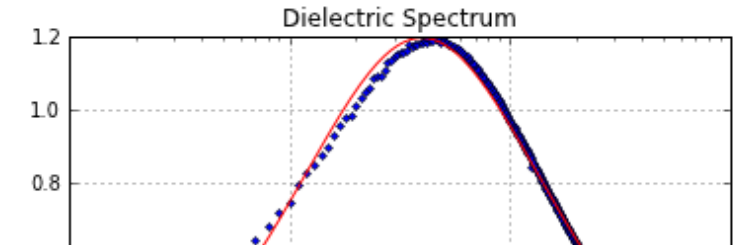
In [324]:
```
popt11, fittedmodel11 = data_extract(['test11'], guess, False, True, True)
```

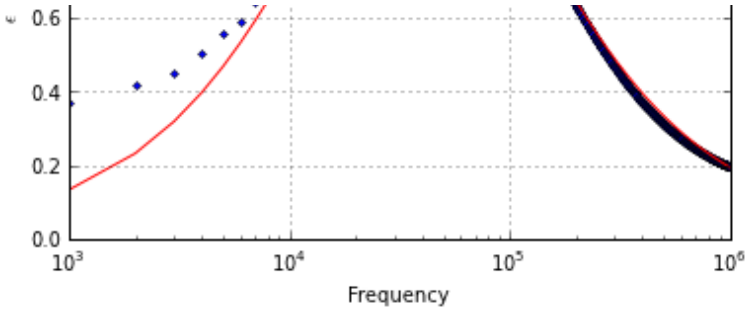
```
RESULTS FOR PRIMARY PEAK:

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)
```

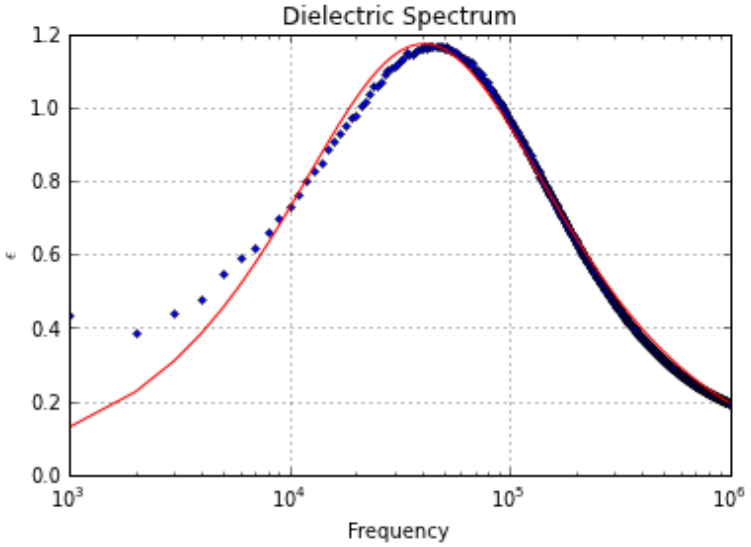
```
RESULTS FOR PRIMARY PEAK:

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





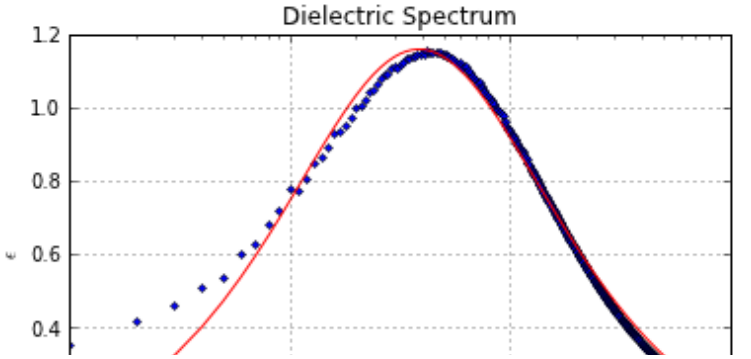
In [326]:
```
popt13, fittedmodel13 = data_extract(['test13'], guess, False, True, True)
```

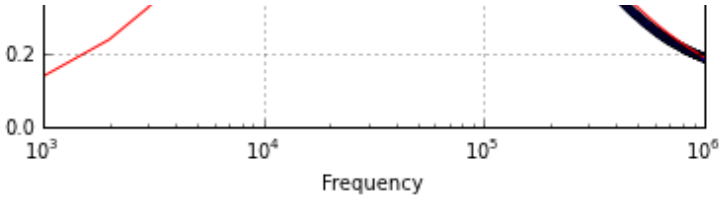
```
RESULTS FOR PRIMARY PEAK:

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)
```

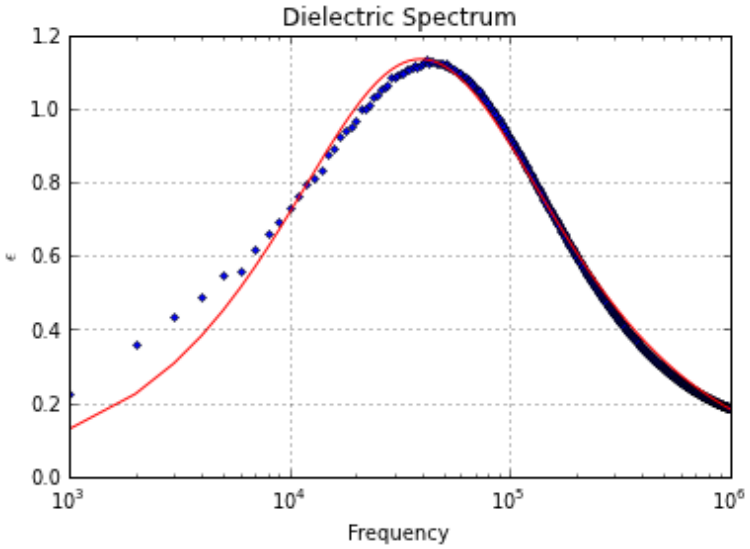
```
RESULTS FOR PRIMARY PEAK:

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





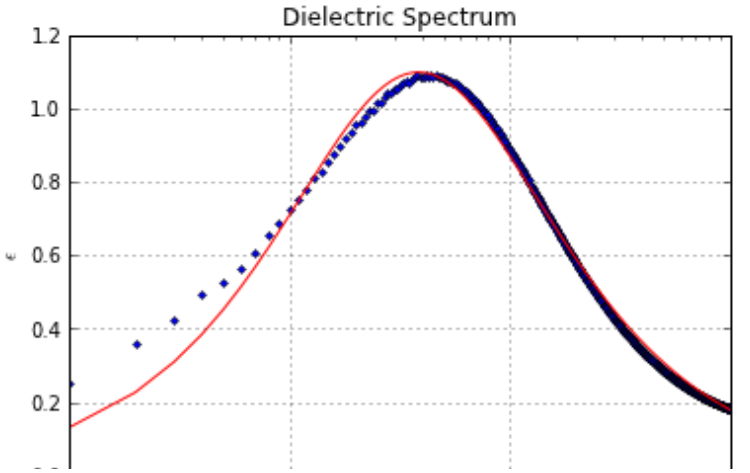
In [328]:
```
popt15, fittedmodel15 = data_extract(['test15'], guess, False, True, True)
```

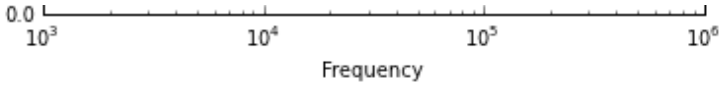
```
RESULTS FOR PRIMARY PEAK:

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)
```

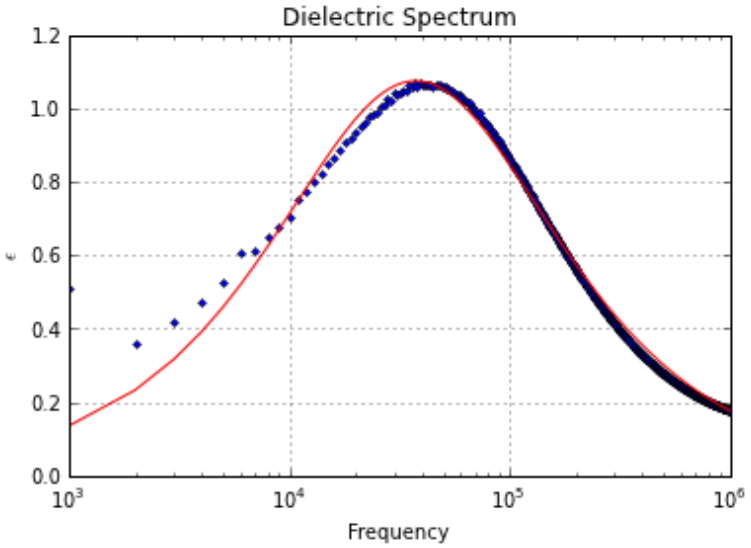
```
RESULTS FOR PRIMARY PEAK:

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





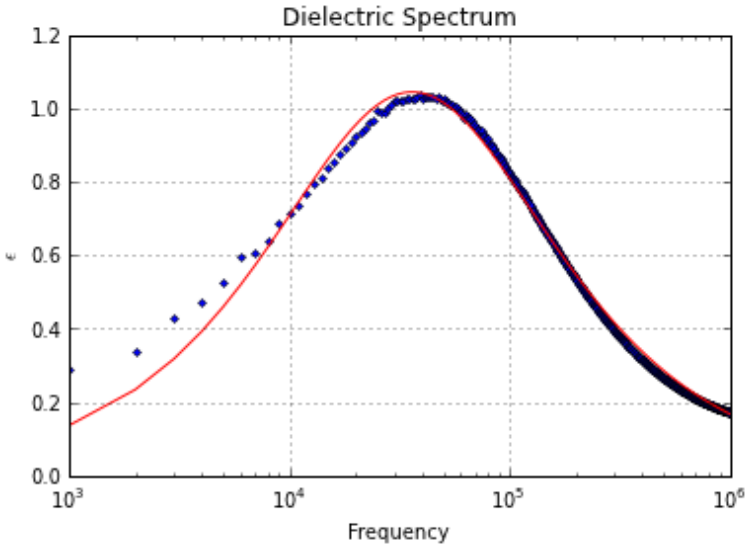
In [330]:
```
popt17, fittedmodel17 = data_extract(['test17'], guess, False, True, True)
```

```
RESULTS FOR PRIMARY PEAK:

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



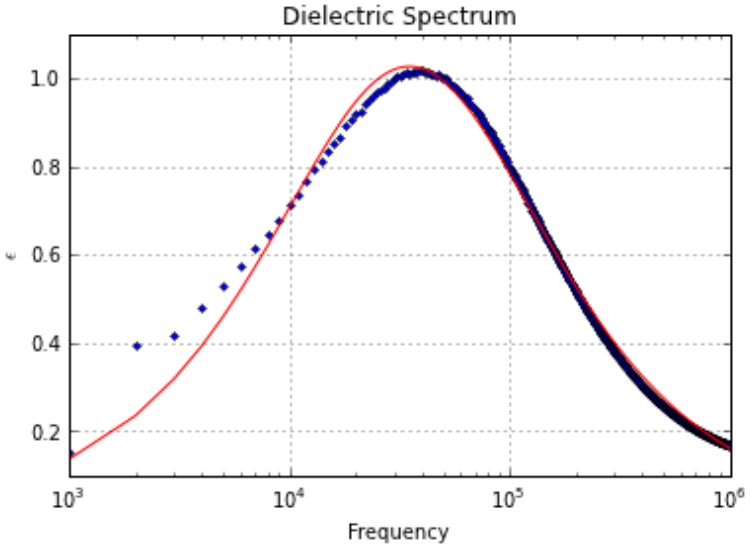
In [331]:
```
popt18, fittedmodel18 = data_extract(['test18'], guess, False, True, True)
```

```
RESULTS FOR PRIMARY PEAK:

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



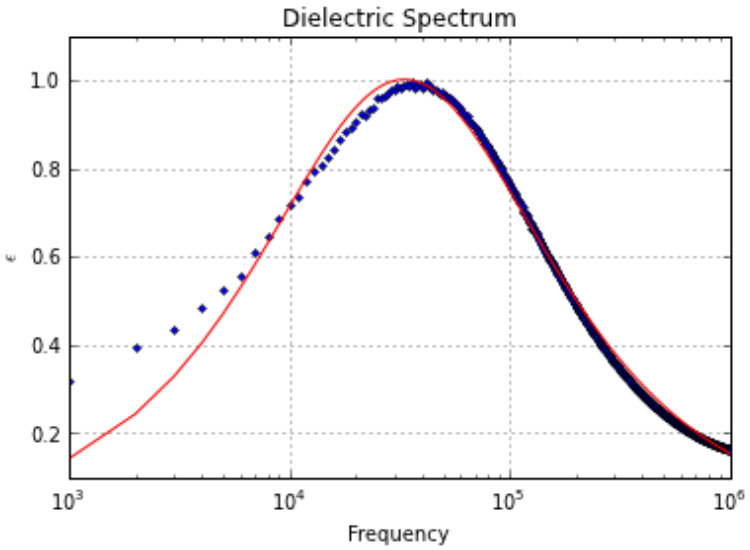
In [332]:
```
popt19, fittedmodel19 = data_extract(['test19'], guess, False, True, True)
```

```
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)
```

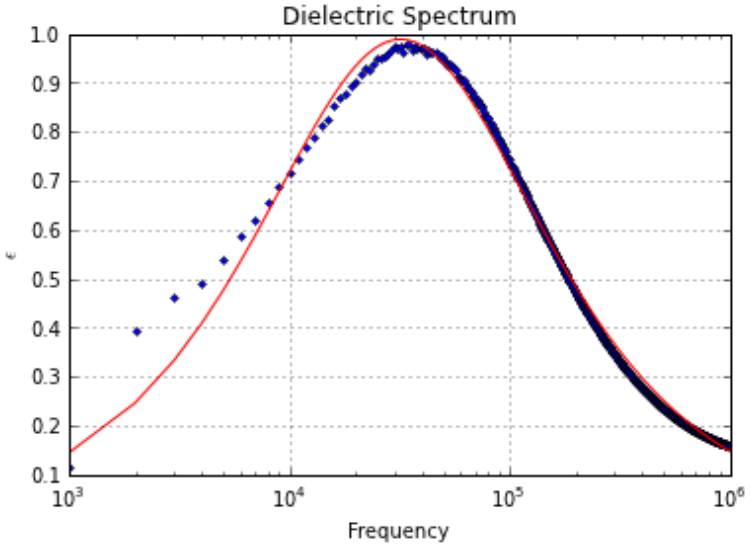
```
RESULTS FOR PRIMARY PEAK:

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)
```

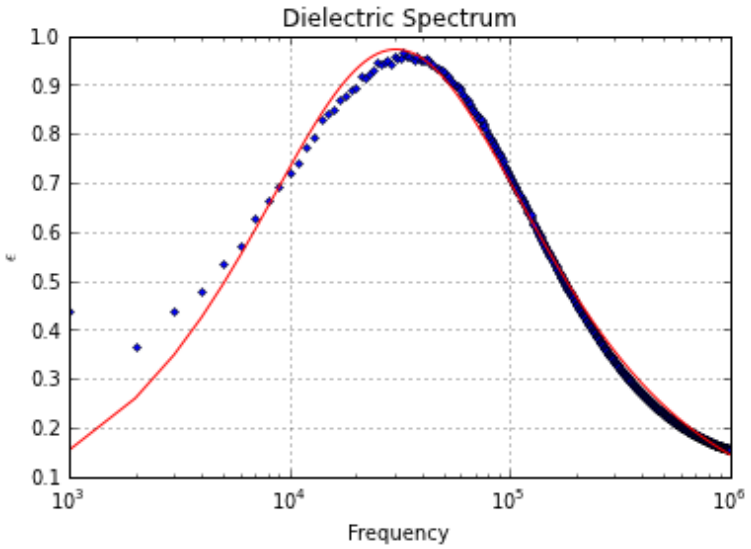
```
RESULTS FOR PRIMARY PEAK:
```

```
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)
```

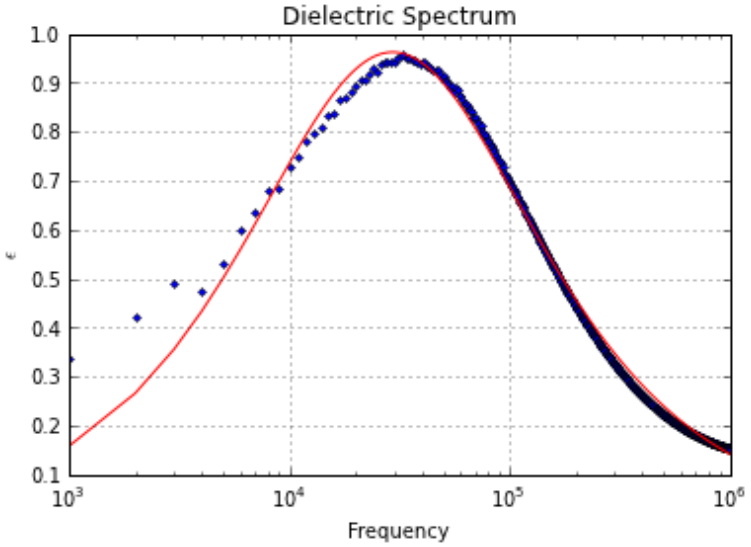
```
RESULTS FOR PRIMARY PEAK:

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)
```

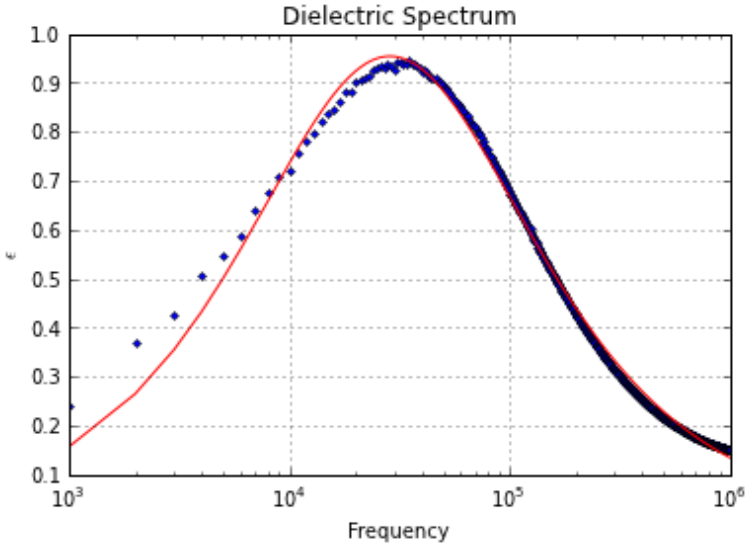
```
RESULTS FOR PRIMARY PEAK:

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)
```

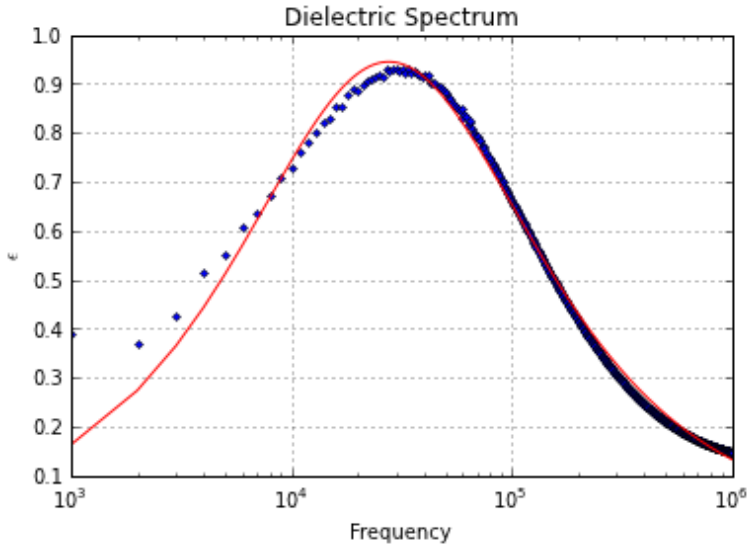
```
RESULTS FOR PRIMARY PEAK:

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)
```

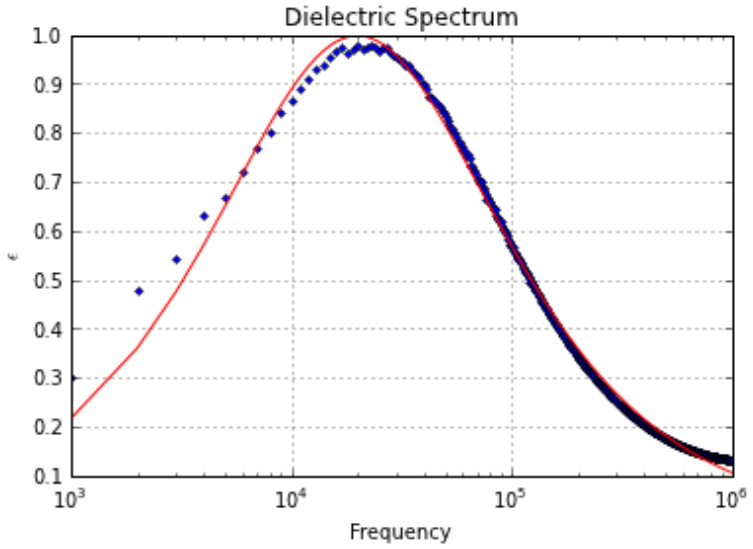
```
RESULTS FOR PRIMARY PEAK:

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

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



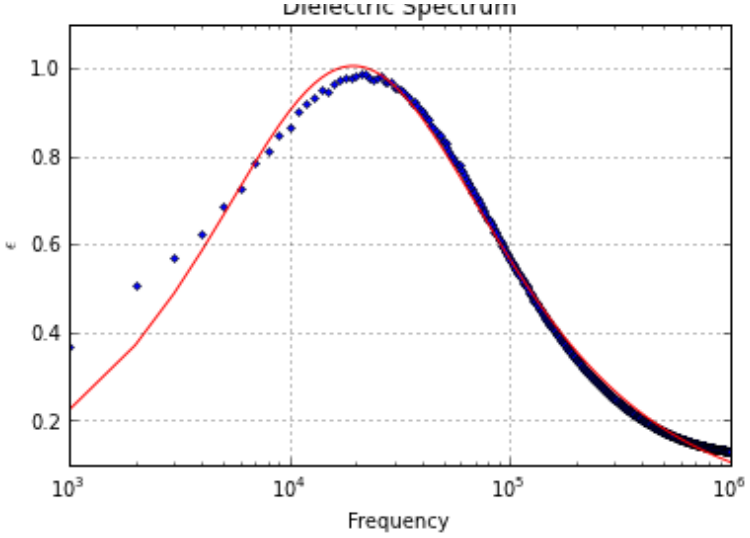
In [339]:
```
popt26, fittedmodel26 = data_extract(['test26'], guess, False, True, True)
```

```
RESULTS FOR PRIMARY PEAK:

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



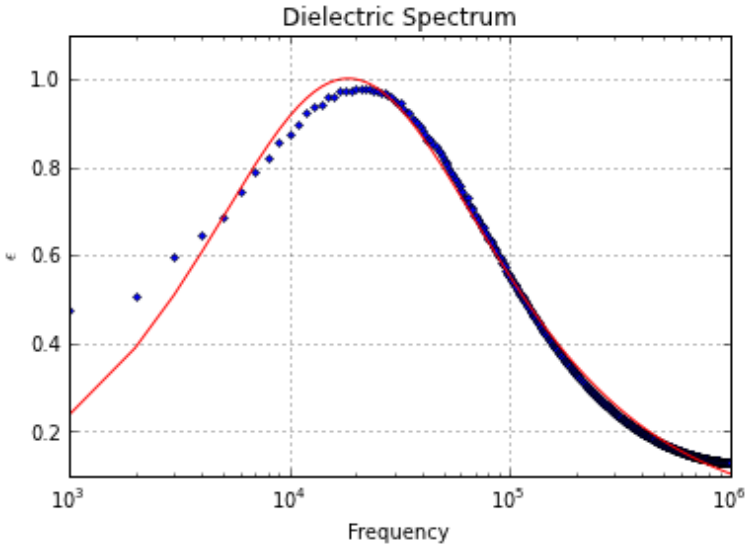
In [340]:
```
popt27, fittedmodel27 = data_extract(['test27'], guess, False, True, True)
```

```
RESULTS FOR PRIMARY PEAK:

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



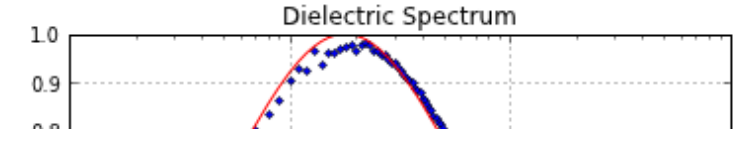
In [341]:
```
popt28, fittedmodel28 = data_extract(['test28'], guess, False, True, True)
```

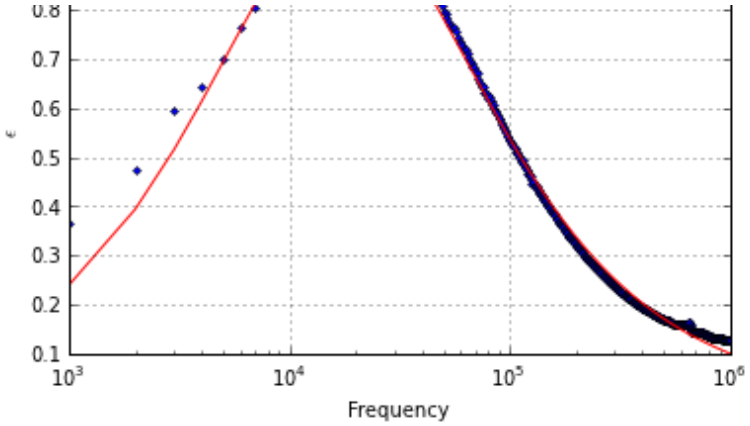
```
RESULTS FOR PRIMARY PEAK:

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)
```

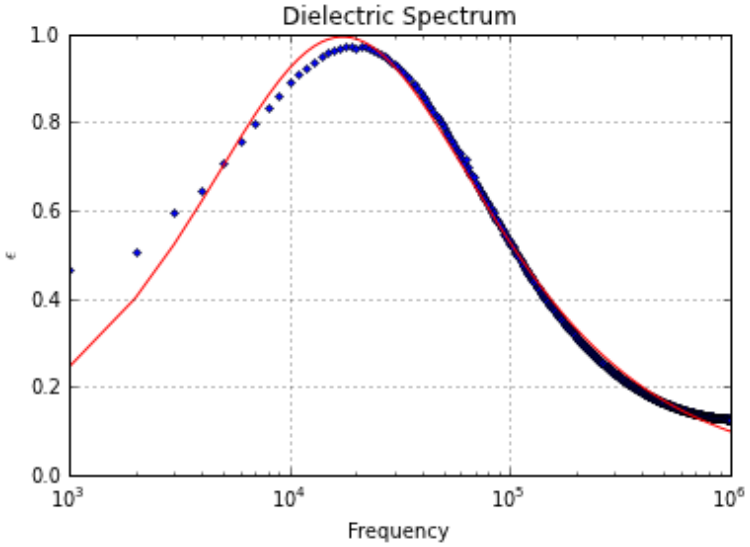
```
RESULTS FOR PRIMARY PEAK:

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





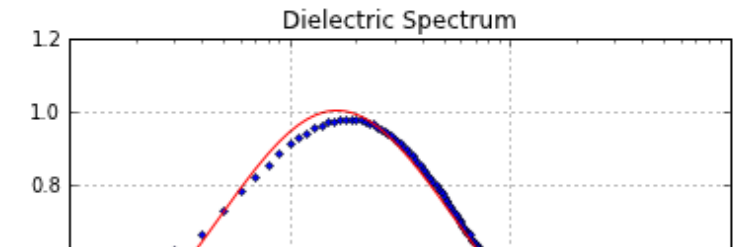
In [343]:
```
popt30, fittedmodel30 = data_extract(['test30'], guess, False, True, True)
```

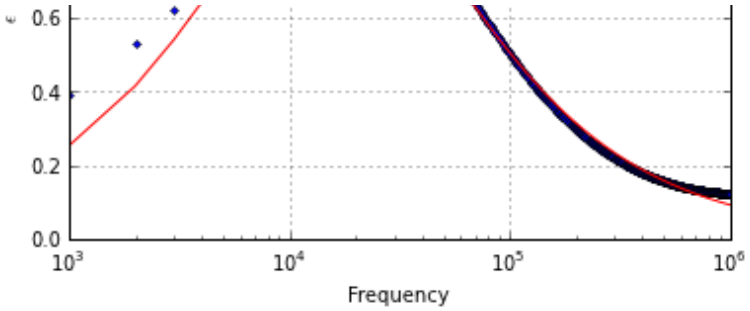
```
RESULTS FOR PRIMARY PEAK:

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)
```

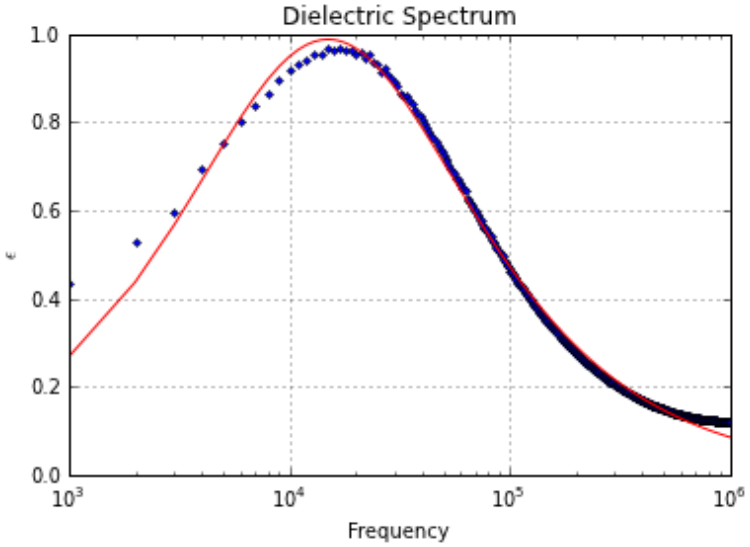
```
RESULTS FOR PRIMARY PEAK:

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
```



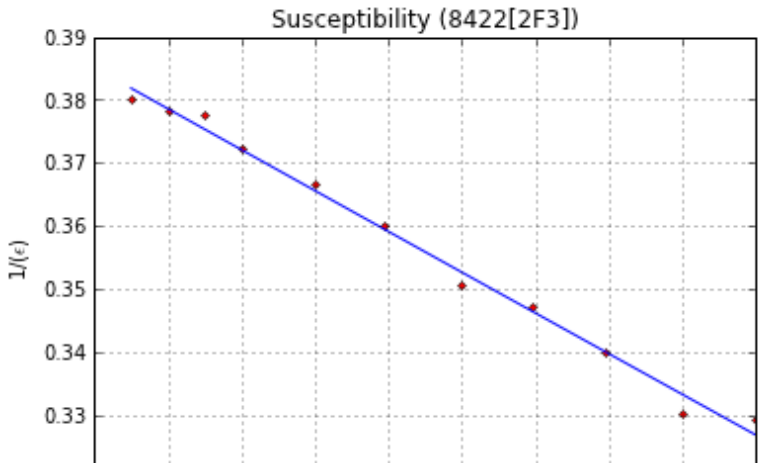
In [366]:
```
#epsilon =
[popt1[1],popt2[1],popt3[1],popt4[1],popt5[1],popt6[1],popt7[1],popt8[1],
popt9[1], popt10[1],
epsilon = [popt13[1],popt14[1],popt15[1],popt16[1],popt17[1],popt18[1],
           popt19[1],popt20[1],popt21[1],popt22[1],popt23[1]]
#,popt24[1],popt25[1]]
#,popt26[1],popt27[1],popt28[1],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

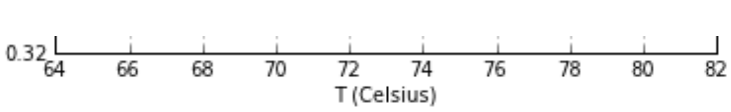
```
temperature =  [82, 80, 77.9,
               75.9, 74, 71.9, 70, 68, 67, 66, 65]
#, 64, 63]
#, 61, 60.1, 59, 58, 57, 56, 55];

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:
$(\Delta\epsilon)^{-1} = At^{\gamma} + B$

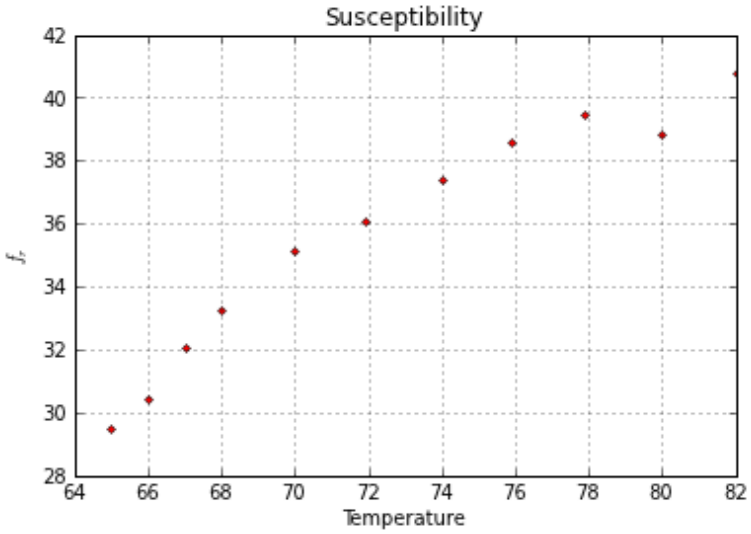
In [369]:
```
def susc_func(x,A,B,gamma):
    s = A*np.power(x,gamma) + B
    return s

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