

**Guide to**

# **RefFIT**

*software to fit optical spectra*

**Alexey Kuzmenko**

**Program version: 1.2.44**

**Last updated: 06.06.2006 (666 version!)**

# Contents

1. Introduction .....	4
1.1. Foreword .....	4
1.2. What is RefFIT for? .....	5
1.3. About this manual .....	6
1.4. Let's keep in touch .....	7
2. Basics of optical data fitting .....	8
2.1. Non-linear modeling .....	8
2.1.1. Levenberg-Marquardt algorithm .....	9
2.1.2. Simultaneous fitting of several datasets of different types .....	12
2.1.3. Confidence limits .....	13
2.2. Modeling of the dielectric functions .....	14
2.2.1. Physical properties of the dielectric functions .....	14
2.2.2. Why modeling? .....	15
2.2.3. Formula-defined dielectric functions .....	16
2.2.4. Variational dielectric functions (KK-constrained) .....	17
2.2.5. 'Not-KK-constrained' variational dielectric functions .....	22
2.3. Differential modeling .....	23
3. Tutorial .....	25
3.1. Drude-Lorentz fitting of a reflectivity spectrum .....	25
3.2. Simultaneous fitting of several data types .....	36
3.3. Using macros .....	42
3.3.1. Writing a simple macro .....	42
3.3.2. Using cycles in a macro: the temperature dependence of phonon spectra .....	44
3.4. Using variational dielectric functions .....	51
3.4.1. Kramers-Kronig analysis of reflectivity .....	51
3.4.2. Inversion of ellipsometric data .....	51
4. Reference .....	52
4.1. System requirements .....	52
4.2. Installation and starting RefFIT .....	52
4.3. Parameters .....	53
4.4. Experimental Parameters .....	54
4.5. Models .....	55
4.5.1. Saving and loading models .....	56
4.5.2. Editing models .....	56
4.5.3. Model types .....	57
4.6. "Dielectric function" model .....	57
4.6.1. Drude-Lorentz dielectric function .....	60
4.6.2. Special dielectric functions .....	60
4.6.3. Variational dielectric function .....	65
4.7. Special models .....	68
4.7.1. Differential dielectric function .....	68
4.7.2. Reflection and transmission of a multi-layer sample .....	69
4.7.3. Ellipsometry of an orthorhombic sample .....	71

4.7.4.	Differential ellipsometry of an orthorhombic sample .....	73
4.7.5.	Ellipsometry of an orthorhombic film on an orthorhombic substrate .....	73
4.7.6.	Extended Drude .....	75
4.7.7.	Epsilon+Mu .....	76
4.7.8.	Spectral Weight .....	79
4.7.9.	Optics of a plate sample .....	81
4.7.10.	Reflectivity of a Film (Epsilon+Mu) on a Substrate .....	84
4.8.	Dataset manager .....	85
4.8.1.	Loading and unloading datasets .....	85
4.9.	Graphs .....	87
4.9.1.	Graph properties .....	88
4.9.2.	Graph contents .....	88
4.10.	Fit window .....	90
4.10.1.	Setting up the fitting task .....	90
4.10.2.	Fitting options .....	91
4.10.3.	Starting/stopping fitting process .....	93
4.11.	Macro language .....	93
4.11.1.	Writing and executing macros .....	93
4.11.2.	Macro syntax .....	94
4.11.3.	General macro commands .....	95
4.11.4.	Dataset macro commands .....	96
4.11.5.	Model macro commands .....	98
4.11.6.	Graph macro commands .....	103
4.11.7.	Experimental parameters macro commands .....	106
4.11.8.	Fitting macro commands .....	107
4.11.9.	Loop macro commands .....	108
5.	Bibliography .....	110

---

# 1. Introduction

## 1.1. Foreword

As a scientist, I am supposed to do experiments and publish articles, not to mention the teaching, so you can imagine that little time is left for programming and, especially, writing manuals. However, I had to make RefFIT because no other software was flexible and fast enough to model spectral data in the way I wanted. It represents, if you want, my idea of data fitting. Initially it was a little DOS program for a Drude-Lorentz fitting of the reflection coefficient (this is where its name comes from). Somewhat later it has grown into a sizeable Windows application intended to facilitate modeling and the extraction of the complex dielectric function from various kinds of optical experiments, such as reflectivity, transmission, ellipsometry *etc.* measured on different kinds of samples (anisotropic, layered *etc.*). The following principles have formed the basis of RefFIT: (i) it should be possible combine different types of spectra and fit them simultaneously, (ii) a user should be able to see in real time how does the fitting come about (and not to wait until the program finishes and gives birth to an output file, which yet has to be painted by a graphical package), (iii) the program should run as fast as possible (iv) there should be a macro language for routine operations.

Initially I did not bother with any written manuals. When some of my colleagues found it useful, it was possible to explain privately, how RefFIT works. However, as more people become interested in it, I realized that a manual would help a lot. It has become even more evident after I had to look into this guide a couple of times myself (the human memory is not that long!).

I have to emphasize, that RefFIT is not professional software, designed in accordance with established interface and compatibility standards, but rather a home-made tool, sharpened to solve particular class of problems by someone, who deals with these problems every day. Unfortunately, it is not ‘fool-proof’ so, please, be prepared that it may crash unexpectedly. All formulas RefFIT uses were carefully checked and tested. However I do not take ‘any legal responsibility’ for possibly overlooked mistakes and the ‘consequences, caused by those’ (please, let me know, if you find any). Some commands and conventions look rather strange because they were put as temporary solutions, but are still in there. Many improvements to RefFIT are planned for a long time but not have been done yet. Nevertheless, I dare to offer you this program already now because in spite of its ‘child diseases’ RefFIT works and nicely solves most of fitting problems that we encounter in our lab. I would be happy if you can find it useful too.

Before I start let me acknowledge those who contributed directly or indirectly to the creation of this program. The very idea of RefFIT has appeared during my graduate study in P.L.Kapitza Institute for Physical Problems (Moscow) as a result of enlightening discussions with my PhD adviser, professor E.A.Tishchenko, who was also my first teacher of optical spectroscopy. When I came a postdoc to the Optical Solid State group headed by professor Dirk van der Marel (University of Groningen and, currently, University of Geneva) I was deeply impressed by his versatile OPTPAL program [1], also designed to fit optical data, but much more powerful than RefFIT at that time. The later development of RefFIT was to a great extent

governed by a desire to combine the best OPTPAL features with a user-friendly interface. The creative science atmosphere of Dirk's group is very stimulating for me. I must specially acknowledge Patricio Mena, who helped me a lot to improve RefFIT by using it extensively in his PhD study and even writing several subroutines. And, of course, I want to thank all my colleagues who elected to use RefFIT as their computational tool and thus furthered its evolution.

## 1.2. What is RefFIT for?

One can generally say that RefFIT is designed to analyze the optical spectra of solids. Clearly, this statement has to be a bit specified, because there are myriads of thinkable optical experiments and, correspondingly, very different ways to analyze them. So which sort of data are we going to deal with and what for 'analysis' we are going to apply?

First of all, by optical *spectra*, we mean *frequency-dependent* optical quantities. An example of an optical spectrum is the frequency-dependent reflectivity  $R(\omega)$ , which can be directly measured. Another example is given by the optical conductivity  $\sigma(\omega)$ , which is usually not measured directly but derived from experimental data after some analysis (or, alternatively, it can be taken from articles of other groups, which is also a sort of measurement ☺). Of course, in addition to the light frequency there might be other experimental parameters, such as the angle of incidence, a sample thickness *etc.* Secondly, we put a common dogma, that the optical properties we deal with are determined solely by the complex *dielectric function*  $\varepsilon(\omega) = \varepsilon_1(\omega) + i\varepsilon_2(\omega)$  of the studied material. Thus, we assume that the measurable optical quantities, such as reflection, transmission, ellipsometry outputs *etc.*, are described by the textbook Fresnel equations [2].

The primary goal of spectra analysis, that RefFIT does, is to get information about the material dielectric function on the base of optical spectra. It is done by the *fitting* of these spectra using a model of the dielectric function with a set of adjustable parameters. These parameters are varied in order to obtain the best match between the experimental and calculated data points.

There are two ways to model the dielectric function. It might be either a mathematical (or physical, if you like) formula with a limited number of parameters, or a variational (also called a 'free-shape') dielectric function, which is 'allowed' to vary independently at every frequency point [4] (see sections 2.2.4 and 2.2.5). The first possibility is very familiar to many (*e.g.*, the Drude-Lorentz modeling of reflectivity). Although the second one might look uncommon, it is often implicitly present in the data analysis. For example, in spectroscopic ellipsometry  $\varepsilon_1$  and  $\varepsilon_2$  are extracted independently at every frequency from the two measured ellipsometric angles -  $\psi$  and  $\Delta$ . The same thing can be done in RefFIT by the fitting of  $\varepsilon(\omega)$  with a *not-Kramers-Kronig-constrained* variational dielectric function (section 2.2.5). Another example is the conventional Kramers-Kronig transformation of reflectivity [6] (used, by the way, in about 90% of papers on infrared spectroscopy of solids), which can be substituted in RefFIT by the fitting of  $R(\omega)$  by a *Kramers-Kronig-constrained* variational dielectric function (section 2.2.4). To make a long story short: *every analysis that RefFIT does is a fitting!*

For a reliable extraction of  $\varepsilon(\omega)$  it is often crucial to fit spectra of *different types simultaneously*. An example, when the analysis of a single quantity might be rather shaky, is the already mentioned Kramers-Kronig transformation of reflectivity. It is well known, that the experimental reflectivity uncertainties and the ambiguity of extrapolations to high and low frequencies in this case can boost the error bars of the resulting dielectric function (or, equivalently, of the optical conductivity). As a rule, it helps a lot to supplement the reflectivity measurement with the ellipsometry or transmission spectra. In RefFIT the user can freely choose the collection of datasets, which should be fitted simultaneously.

An indicative list of the built-in models includes the Drude-Lorentz model, the dielectric function of a BSC superconductor, the extended Drude model, ellipsometry, reflection and transmission of multi-layer and anisotropic samples. Of course, it is hopeless to predict all the experiments that a solid-state optical spectroscopist might wish to do. Therefore, new models are added to RefFIT from time to time in order to meet the emerging needs.

The fitting is always a try-and-error activity, which requires a lot of human efforts and intuition. Sometimes one can spend hours and even days before a proper model is found. In order to ease this procedure, RefFIT was designed in such a way that a user can see the results of his manipulations with parameters on the graphs in real time.

Once a good model is found, it might be necessary to apply it many times in the same way to different datasets, *e.g.* to study temperature dependence of spectra. It is obviously not practical to continue doing it ‘by hand’. Do not worry: the built-in macro language greatly facilitates the execution of lengthy routine calculations in RefFIT.

### **1.3. About this manual**

This manual consists of three major parts.

In Chapter 2 some theoretical aspects of the optical data fitting are considered. It starts with the description of the celebrated Levenberg-Marquardt algorithm for a non-linear modeling, which is the ‘computational engine’ of RefFIT. Then the physical constraints on the dielectric functions are discussed, which have to be met by a realistic model. We also describe in details a novel method of the so-called variational dielectric functions (RefFIT is likely the first program to use this technique). Finally, the analysis of differential (modulation) spectra is discussed.

Chapter 3 is a tutorial that shows, step-by-step, how to tackle typical fitting problems with RefFIT. The examples are selected in order to cover the most essential features of RefFIT.

Finally, in Chapter 4 you can find the full reference information about RefFIT, including the description of all its elements (datasets, models, graphs *etc.*), the exact formulas used to calculate optical quantities, the macro language and so on.

This manual is not meant to be an optics course, so I assume that the reader is familiar with optical spectroscopy. A basic acquaintance with Windows is also assumed.

The screenshots, which I used in the text, are made on my laptop, running under Windows XP. If you have another Windows version, then the appearance of some graphics on your screen might look slightly different.

## 1.4. Let's keep in touch

Please, do not hesitate to contact me if you have any problems or suggestions regarding RefFIT or this manual. Depending on the problem, I might be able to solve it quickly and send you an improved program version. It is even feasible to include extra features according to your particular needs, provided that these extras are reasonable enough to be potentially useful to other users. For instance, one can easily put new formulas for the dielectric function, new model types or new macro commands. It is clear that only the user's feedback is really able to improve the program.

If you wish to periodically receive information about the recent developments in RefFIT, such as new models included, new possibilities added and bugs fixed, you can send me an e-mail. I would appreciate it if you provide some information about your field of research and the type of optical measurements you do.

My current contact information is:

*Phone:* +41-22-3793105, *Fax:* +41-22-3796869

*E-mail:* [Alexey.Kuzmenko@physics.unige.ch](mailto:Alexey.Kuzmenko@physics.unige.ch)

University of Geneva, DPMC, Quai Ernest-Ansermet, 24, 1211 Geneva, Switzerland

---

## 2. Basics of optical data fitting

### 2.1. Non-linear modeling

The fitting or modeling of the experimental (in particular, optical) data is a process, when someone looks for a meaningful model with a set of adjustable parameters and varies them in order to get the best match of the model to experimental curves.

To find a proper model is usually a central problem, which a computer can hardly do on its own (we humans are still proud to be smarter than the machines we make!). Once a model is thought up and reasonable initial values of parameters are found, the second stage is to vary them in order to get the best fit. To continue doing it by hand is usually not practical and even doable, when there are more than 2 or 3 tangled parameters. It could be even dangerous, because a ‘reasonably looking’ match might be rather far from the numerically-the-best one. Fortunately, at this stage the fitting procedure can be very efficiently automated as described below.

Suppose, we have a set of  $N$  experimental data points  $\{x_i, y_i, \sigma_i\}$  ( $i = 1, \dots, N$ ), that we want to fit. Here  $x_i$  is the data coordinate<sup>1</sup>,  $y_i$  is the data value and  $\sigma_i$  is the data error bar. Next, we take a model, which calculates the data value  $y$  as a function of  $x$ , and a set of internal parameters  $\{p_1, p_2 \dots p_M\}$ :  $y = f(x, p_1 \dots p_M)$ .

Let us construct a so-called ‘chi-square’ functional:

$$\chi^2 \equiv \sum_{i=1}^N \left( \frac{y_i - f(x_i, p_1 \dots p_M)}{\sigma_i} \right)^2 = \chi^2(p_1, \dots p_M)$$

Equation 2-1

If we assume that all measured values  $y_i$  are *normally distributed* with standard deviations given by  $\sigma_i$ , then ‘statistically-the-best’ match would correspond to the *minimal* value of  $\chi^2$ . Thus, the modeling is essentially the minimization of the chi-square with respect to parameters. Therefore, the method itself is called the ‘least-square’ technique.

Of course, the error bars are determined not only by a statistical noise, but also by systematic inaccuracies, which are very hard to estimate and are clearly not normally distributed<sup>2</sup>. However, to move on, we suppose that they are somehow accounted for by the values  $\sigma_i$ .

When  $f(x, p_1 \dots p_M)$  is a non-linear function of parameters (which is virtually always the case in optical modeling), the so-called Levenberg-Marquardt algorithm of the chi-square minimization is indispensable. We shall discuss it in the next section.

---

<sup>1</sup> In the case of optical data  $x_i$  is usually the light frequency.

<sup>2</sup> Honestly, I do not believe there is a ‘scientific way’ to deal with systematic error bars (tell me if I am wrong!)

### 2.1.1. Levenberg-Marquardt algorithm

The Levenberg-Marquardt (LM) algorithm is based on the self-adjustable balance between the two minimizing strategies: the ‘gradient descent’ and the ‘inverse Hessian’ methods.

The ‘gradient descent’ method is simply an instinctive moving in the ‘steepest descent’ direction, which is apparently determined by the minus-gradient:

$$\beta_k \equiv -\frac{1}{2} \frac{\partial \chi^2}{\partial p_k} = \sum_{i=1}^N \frac{y_i - f(x_i, p_1, \dots, p_M)}{\sigma_i^2} \frac{\partial f}{\partial p_k}(x_i, p_1, \dots, p_M)$$

Equation 2-2

(the one-half coefficient is put to simplify the formulas). Suppose, the current parameter values are  $p_k$  ( $k=1, \dots, M$ ). To improve the fit, we can ‘shift’ the parameters  $p_k \rightarrow p_k + \delta p_k$ , where

$$\delta p_k = \text{constant} \times \beta_k$$

Equation 2-3

The absolute value of the constant we will discuss later. The ‘steepest descent’ strategy is justified, when one is far from the minimum, but it becomes extremely inefficient in the ‘plateau’ close to the minimum, especially in the multi-parameter space.

In the latter case it is much better to assume that the function to be minimized has almost parabolic shape, determined by the Hessian:

$$\alpha_{kl} \equiv \frac{1}{2} \frac{\partial^2 \chi^2}{\partial p_k \partial p_l} = \sum_{i=1}^N \frac{1}{\sigma_i^2} \left\{ \frac{\partial f}{\partial p_k}(x_i, p_1, \dots, p_M) \frac{\partial f}{\partial p_l}(x_i, p_1, \dots, p_M) - [y_i - f(x_i, p_1, \dots, p_M)] \frac{\partial^2 f}{\partial p_k \partial p_l}(x_i, p_1, \dots, p_M) \right\}$$

Equation 2-4

(the one-half here is also for the sake of simplicity). After the computing, numerically or analytically, the gradient and the Hessian for the current set of parameters, one can immediately ‘jump’ to the minimum by shifting the parameters  $p_k \rightarrow p_k + \delta p_k$ , where the displacement vector  $\delta p_k$  is determined from the linear system:

$$\sum_{i=1}^M \alpha_{kl} \delta p_l = \beta_k .$$

Equation 2-5

It was argued (see Ref. [3]), that the term in Equation 2-4, which contains the second derivative  $\frac{\partial^2 f}{\partial p_k \partial p_l}$ , is not important near the minimum and, moreover, may even destabilize the fitting process. So, instead of Equation 2-4 we shall define the  $\alpha$ -matrix simply as:

$$\alpha_{kl} \equiv \sum_{i=1}^N \frac{1}{\sigma_i^2} \frac{\partial f}{\partial p_k}(x_i, p_1, \dots, p_M) \frac{\partial f}{\partial p_l}(x_i, p_1, \dots, p_M).$$

Equation 2-6

Coming back to the ‘steepest descent’ technique, one can see that the Equation 2-3 has a problem with the unit dimensions. Let us suppose that the parameter  $p_k$  is measured in  $\text{cm}^{-1}$ . Then  $\beta_k$  has the units of  $\text{cm}$  (as the  $\chi^2$  is dimensionless) and the constant ought to have a dimension ( $\text{cm}^{-2}$  in this case). Therefore it cannot be the same for all parameters, which are generally measured in different units (seconds, Teslas *etc.*). The solution is to use the *dimensionless constant*. The only way to get rid of the dimension, is to normalize it by  $\alpha_{kk}$  :

$$\delta p_k = \frac{\text{constant}}{\alpha_{kk}} \times \beta_k.$$

Equation 2-7

There is an elegant way, due to Marquardt, to continuously ‘switch’ from one strategy to another. Let us consider a ‘diagonally-enhanced’  $\alpha$ -matrix:

$$\alpha'_{kl} = \alpha_{kl} (1 + \delta_{kl} \lambda),$$

Equation 2-8

where  $\lambda$  is a dimensionless constant, and replace  $\alpha_{kl}$  with  $\alpha'_{kl}$  in Equation 2-5:

$$\sum_{i=1}^M \alpha'_{kl} \delta p_l = \beta_k.$$

Equation 2-9

If we take  $\lambda \ll 1$ , then the displacement vector  $\delta p_k$ , obtained from Equation 2-9, is close to the one, obtained by the pure ‘inverse Hessian’ technique (Equation 2-5). However, if  $\lambda \gg 1$ , then we can almost neglect the off-diagonal elements and the solution of Equation 2-9 becomes simply

$$\delta p_k = \frac{\beta_k}{\alpha'_{kk}} = \frac{\beta_k}{\alpha_{kk} (1 + \lambda)}.$$

Equation 2-10

One can see that Equation 2-10 has the same form, as Equation 2-7. It means that, by increasing the parameter  $\lambda$ , we approach the ‘steepest descent’ limit.

Now we are ready to formulate the LM algorithm, which block diagram is shown in Figure 2-1.

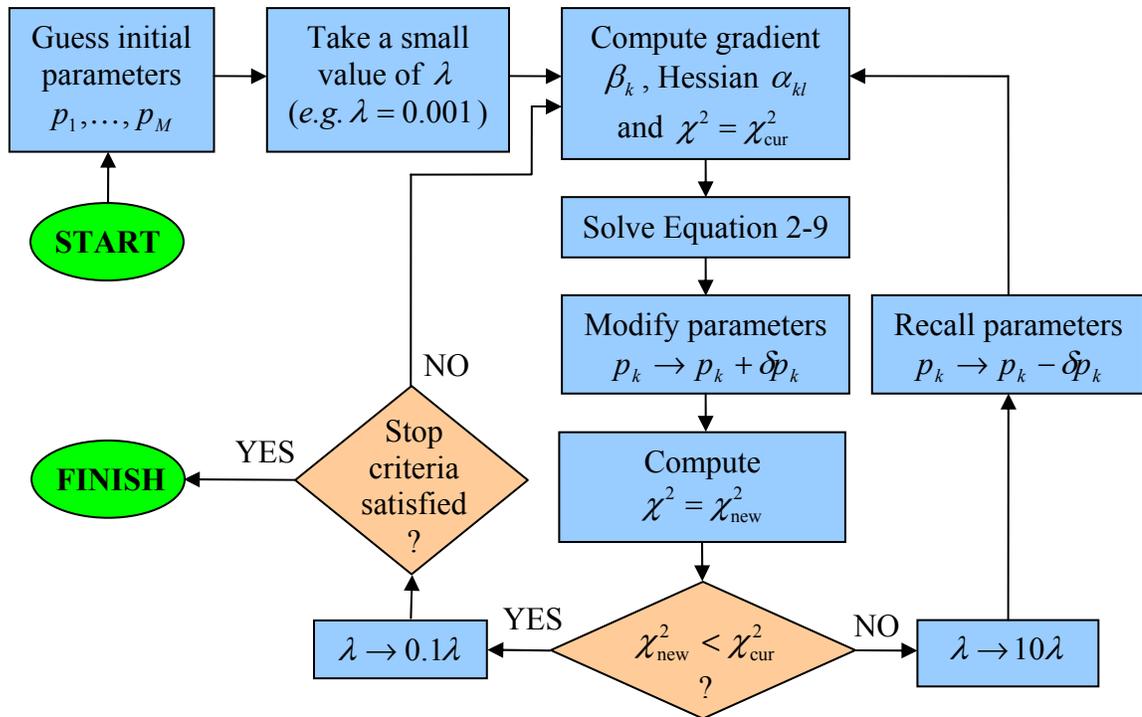


Figure 2-1. The block diagram of the Levenberg-Marquardt technique, used by RefFIT.

The minimization process is iterative. One starts with a reasonably small value of  $\lambda$ . At every successful iteration ( $\chi^2_{\text{new}} < \chi^2_{\text{cur}}$ ) it is *reduced* by a factor of 10, moving towards the ‘inverse Hessian’ regime. Otherwise it retreats to the ‘steepest descent’ regime by being *increased* by a factor of 10. When the so-called stop criteria are satisfied, the fitting process stops.

An important issue is the computation of  $\alpha_{kl}$  and  $\beta_k$ . According to Equation 2-2 and Equation 2-6, we have to calculate the first derivatives of the model function  $f(x, p_1 \dots p_M)$  with respect to parameters. In general, one can do it numerically, which is the only possibility, if the model function (but not the derivatives) is provided by an external routine. In this case much care has to be taken, as the computational errors could be rather large, especially near the minimum point. Fortunately, since we are going to use the model functions given by explicit analytical expressions, we can always calculate the derivatives *analytically*.

The stop criteria are necessary to avoid an endless iteration cycle. The most natural reason to stop is when the fitting process converges, or, formally, when the absolute reduction of the  $\chi^2$  during the last few (e.g., three) iterations does not exceed a certain threshold  $\delta\chi^2$ . It also makes sense to ensure that the fitting process does not take too long by putting certain limits on the total number of iterations or (and) the total fitting time. There could be many more criteria. For instance, the process must terminate, when an impatient user hits the “STOP” button.

It is remarkable, that the LM algorithm, in spite of its simplicity, may easily handle models that contain a huge number of parameters (of the order of thousands!). The convergence speed, *i.e.*, the number of iterations, needed to reach the minimum, is not significantly

influenced by the number of parameters. It rather depends on the adequacy of the model to the experimental data and the success of the initial approximation.

### 2.1.2. Simultaneous fitting of several datasets of different types

So far we considered the fitting of only one dataset by a single model. It is rather straightforward to extend the discussion to a case, when *several datasets of different experimental types* have to be fitted *simultaneously* with *several models*.

Let us consider  $Q$  datasets, while an  $\nu$ -th dataset ( $\nu = 1 \dots Q$ ) contains  $N_\nu$  datapoints:  $\{x_i^\nu, y_i^\nu, \sigma_i^\nu\}$  ( $i = 1 \dots N_\nu$ ). Suppose that an  $\nu$ -th dataset should be fitted by its own model  $f_\nu(x, p_1, \dots, p_M)$ . Although in this notation all models depend formally on the same set of parameters, it does not imply that every model *really* depends on all parameters. In other words, some derivatives  $\partial f_\nu / \partial p_k$  may be equal to zero by definition. It is important, however, that *different models may depend on the same parameters*.

Our goal is to fit several datasets simultaneously. For each dataset a separate chi-square term can be written:

$$\chi_\nu^2 \equiv \sum_{i=1}^{N_\nu} \left( \frac{y_i^\nu - f_\nu(x_i^\nu, p_1, \dots, p_M)}{\sigma_i^\nu} \right)^2$$

Equation 2-11

We can compose the total chi-square to be minimized:

$$\chi^2 = \sum_{\nu=1}^Q w_\nu \chi_\nu^2$$

Equation 2-12

Here  $w_\nu$  are the ‘weights’ of individual chi-square terms that have to be adjusted, as discussed below. The definitions of  $\beta_k$  (Equation 2-2) and  $\alpha_{kl}$  (Equation 2-6) should be modified accordingly:

$$\beta_k = \sum_{\nu=1}^Q w_\nu \sum_{i=1}^{N_\nu} \frac{y_i^\nu - f_\nu(x_i^\nu, p_1, \dots, p_M)}{(\sigma_i^\nu)^2} \frac{\partial f_\nu}{\partial p_k}(x_i^\nu, p_1, \dots, p_M),$$

$$\alpha_{kl} = \sum_{\nu=1}^Q w_\nu \sum_{i=1}^{N_\nu} \frac{1}{(\sigma_i^\nu)^2} \frac{\partial f_\nu}{\partial p_k}(x_i^\nu, p_1, \dots, p_M) \frac{\partial f_\nu}{\partial p_l}(x_i^\nu, p_1, \dots, p_M).$$

The remaining part of the LM algorithm goes exactly as in 2.1.1.

The weight coefficients  $w_\nu$  deserve special remarks. Rigorously speaking, they should be equal to 1, provided that the spreads of all data points are statistically independent. However, due to the *systematic* error bars, this assumption is obviously not correct. For instance, the shift of the reflectivity coefficient, caused by the reference mirror imperfection, is not very different for two spectrally close data points. The second problem is that the error bars  $\sigma_i^\nu$  are not always well known. Therefore, it is often necessary to ‘tune’ the weight coefficients in order to

achieve a proper ‘balance’ between the contributions of different data sets to the total chi-square (Equation 2-12). But how this can be done? Frankly speaking, I see no other recipe but to try different values of  $w_v$ , see the result, and listen to your own intuition when choosing the best one.

### 2.1.3. Confidence limits

When the fitting is done, it is often necessary to estimate the ‘error bars’ of the obtained parameter values  $p_k^{(0)}$ . From the statistical point of view, it is more correct to talk about the so-called ‘confidence limits’. One can intuitively define the confidence limit of a parameter  $p_k$  as the largest possible value  $\delta p_k$ , such as the shift  $p_k \rightarrow p_k^{(0)} + \delta p_k$  does not cause an ‘unrealistically’ large increase of the chi-square. An essential addition to this definition is that, after the shifting of the value of  $p_k$ , one should again minimize  $\chi^2$  with respect to all remaining parameters.

The importance of an extra minimization is clear from the following (a bit exaggerated) example. Let us consider the following model function  $f$ , which depends on the two parameters ( $p_1$  and  $p_2$ ) and does not even depend on  $x$ :  $f(p_1, p_2) = p_1 + p_2$ . Suppose, our dataset contains only one data point  $\{y = 1, \sigma = 1\}$  ( $x$  is not important here). The chi-square in the case is  $\chi^2 = (1 - p_1 - p_2)^2$ . The fitting procedure may converge, for instance, to  $p_1^{(0)} = 0.6$  and  $p_2^{(0)} = 0.4$  (in this case the fit is exact and  $\chi^2 = 0$ ). What are the confidence limits of both parameters? Obviously, there are no limits at all, because for any given number  $a$ , the combination  $p_1 = a, p_2 = 1 - a$  also provides an exact fit! However, at any fixed value of  $p_2$ , the shift of  $p_1$  will cause an increase of the  $\chi^2$ . If we now set that the largest ‘realistic’ value of  $\chi^2$  is 0.01 then formally the ‘confidence limit’ of  $p_1$  should be 0.1. The same is valid for  $p_2$ . One can say that parameters  $p_1$  and  $p_2$  are correlated.

The calculation of the confidence limits  $\delta p_k$  is relatively simple. We can ignore the deviations of  $\chi^2(p_1, \dots, p_M)$  near the minimum point  $\chi_{(0)}^2(p_1^{(0)}, \dots, p_M^{(0)})$  from the quadratic shape, given by the Hessian matrix  $\alpha_{kl}$ .

$$\chi^2 - \chi_{(0)}^2 = \sum_{k,l} \alpha_{kl} (p_k - p_k^{(0)})(p_l - p_l^{(0)}).$$

Equation 2-13

Let  $\delta \chi^2$  be the ‘maximal acceptable’ difference  $\chi^2 - \chi_{(0)}^2$  that we will discuss later. Then, after simple algebra, we can get:

$$\delta p_k = \sqrt{(\delta \chi^2)(\alpha^{-1})_{kk}},$$

Equation 2-14

where  $(\alpha^{-1})_{kl}$  is the inverse Hessian matrix also called the ‘covariance matrix’. Note, that all the diagonal elements  $(\alpha^{-1})_{kk}$  are positive because  $\chi_{(0)}^2 = \chi^2(p_1^{(0)}, \dots, p_M^{(0)})$  is a local minimum point and Equation 2-14 can always be applied.

The reasonable choice of  $\delta\chi^2$  absolute value is quite an issue. Ideally,  $\delta\chi^2$  should be defined by the condition:

$$P\left(\frac{M}{2}, \frac{\delta\chi^2}{2}\right) = p,$$

Equation 2-15

where  $M$  is the number of parameters,  $p$  is the desired confidence probability limit (typically,  $p = 0.95$ ) and  $P(a, x) \equiv \left(\int_0^x e^{-t} t^{a-1} dt\right) / \left(\int_0^\infty e^{-t} t^{a-1} dt\right)$  is the incomplete gamma-function (the derivation can be found in Ref. [3]).

However, the Equation 2-15 can be applied, if (i) the data points are statistically independent, (ii) all weight coefficients  $w_v$  are unities, and (iii) the model is *absolutely adequate* to the data. However, as was mentioned in the section 2.1.2, the existence of the systematic error bars invalidates the first two assumptions. One also needs much optimism to heavily rely on the assumption (iii). In this situation the choice of  $\delta\chi^2$  becomes rather ambiguous and, therefore, human-dependent.

Fortunately, from Equation 2-14 it follows that  $\delta\chi^2$  scales the confidence limits of all parameters *proportionally*. One can therefore reliably *compare* the error bars of different parameters, even though their absolute values might be ill-defined.

## 2.2. Modeling of the dielectric functions

The central assumption in the calculations RefFIT does is that all measurable optical quantities (such as reflectivity, penetration depth *etc.*) can be expressed in terms of the *complex* frequency-dependent dielectric function  $\varepsilon(\omega) = \varepsilon_1(\omega) + i\varepsilon_2(\omega)$  of the material under study. Therefore, the most important issue is the modeling of the dielectric function itself.

### 2.2.1. Physical properties of the dielectric functions

It is well-known from the textbooks (*e.g.*, [2]) that any realistic dielectric function ought to satisfy certain *physical* conditions.

First of all,  $\varepsilon_1(\omega) = \varepsilon_1(-\omega)$  and  $\varepsilon_2(\omega) = -\varepsilon_2(-\omega)$ , therefore it is sufficient to model the  $\varepsilon(\omega)$  for  $\omega \geq 0$  only.

Secondly,  $\varepsilon_2(\omega > 0) \geq 0$ , which means that the intensity of light *cannot increase* in the direction of propagation.

The third requirement is that at very high frequencies the optical properties of matter are the same as those of vacuum:  $\varepsilon_1(\omega \rightarrow \infty) = 1$  and  $\varepsilon_2(\omega \rightarrow \infty) = 0$ .

Finally, due to the causality principle, the real and imaginary parts are not independent, but coupled via the so-called Kramers-Kronig (KK) relation<sup>3</sup>:

$$\varepsilon_1(\omega) - 1 = \frac{2}{\pi} \int_0^{\infty} \frac{x \varepsilon_2(x) dx}{x^2 - \omega^2}.$$

Equation 2-16

Since the integrated function has a pole at  $x = \omega$ , the principal value of the integral should be taken.

The Equation 2-16 is a particularly strong constraint. It implies that, *the knowledge of the  $\varepsilon_2(\omega)$  in the whole spectral range is enough to restore the  $\varepsilon_1(\omega)$  without any extra assumptions*. Thus only one of the two functions  $\varepsilon_1(\omega)$  and  $\varepsilon_2(\omega)$  is independent.

The dielectric function  $\varepsilon(\omega)$  of a physical system is often presented a *sum* of different terms (contributions) due to the optical response of independent subsystems:

$$\varepsilon(\omega) = 1 + \varepsilon^A(\omega) + \varepsilon^B(\omega) + \dots$$

Obviously, in this case the KK relations must hold for individual contributions separately:

$$\varepsilon_1^A(\omega) = \frac{2}{\pi} \int_0^{\infty} \frac{x \varepsilon_2^A(x) dx}{x^2 - \omega^2}, \quad \varepsilon_1^B(\omega) = \frac{2}{\pi} \int_0^{\infty} \frac{x \varepsilon_2^B(x) dx}{x^2 - \omega^2} \dots$$

Equation 2-17

Note that 1 is no longer present in this formula, like in Equation 2-16.

What is the practical value of the KK relation? It is not obvious, how to use it, as experimental spectra are never measured in the whole spectral range. In many cases, if the range is very limited, it makes no sense to impose the KK relation on the dielectric function to be found. If the spectral range is relatively large, then the KK condition becomes essential, but one should be rather careful while making assumptions about the behavior of the dielectric function outside the considered spectral region. We shall discuss this issue later on.

### 2.2.2. Why modeling?

Why and when do we need to model the dielectric functions? One can speculate much on that, but I would distinguish the two main possibilities.

Firstly, one can get some physically meaningful characteristics of the material under study (for example, the plasma frequency, the fundamental gap or the phonon line asymmetry) directly from the experimental spectra, assuming a specific physical model (metallic Drude conductivity, semiconductor gap, Fano lineshape *etc.*). In this case the obvious recipe is to fit the data with a formula-defined function, which is derived from this model (section 2.2.3). The quality of the best match will suggest how good the model is.

<sup>3</sup> There exists another KK relation that we will not use, as it does not impose essentially new constraints

The second possibility is to extract the dielectric function  $\varepsilon(\omega)$  *itself* from the measured spectra as careful as possible without any model assumptions (the model-dependent analysis of  $\varepsilon(\omega)$  can be done afterwards). For example, we might want to obtain  $\varepsilon_1(\omega)$  and  $\varepsilon_2(\omega)$  from the measured ellipsometric quantities  $\psi(\omega)$  and  $\Delta(\omega)$ , which is rather trivial (in the isotropic case), or from reflectivity  $R(\omega)$  alone, which is not trivial, as it heavily relies on the Kramers-Kronig relations. It is remarkable, that in this case the problem can be again formulated in terms of fitting! Then, one has to find the *best* dielectric function that matches the available optical data. In other words, we have to minimize the chi-square (Equation 2-1 or Equation 2-12) in the whole Hilbert space of physically sensible dielectric functions, which formally means that we have an *infinite* number of the fitting parameters. As it is technically impossible to deal with an infinite number of parameters, one has to restrict somehow the set of considered dielectric functions to a finite-dimensional subspace<sup>4</sup>. One way to do it is to represent  $\varepsilon(\omega)$  again by a formula (section 2.2.3). The particular meaning of the formula is not important, provided that it gives a physically allowed result (see section 2.2.1) and is ‘flexible’ enough to fit all important features of the experimental spectra. However, the formula-given dielectric functions are usually too ‘stiff’ for this purpose. In this case a better solution might be to use the so-called variational dielectric functions (section 2.2.4).

### 2.2.3. Formula-defined dielectric functions

There is a variety of formulas corresponding to different models of the dielectric functions. Perhaps, the most famous example is the Drude-Lorentz (DL) model, which we will refer to many times later on:

$$\varepsilon(\omega) = \varepsilon_{\infty} + \sum_i \frac{\omega_{pi}^2}{\omega_{oi}^2 - \omega^2 - i\gamma_i\omega},$$

Equation 2-18

It describes the optical response of a set of harmonic (damped) oscillators. Here  $\varepsilon_{\infty}$  is the so-called ‘high-frequency dielectric constant’, which represents the contribution of all oscillators at very high frequencies (compared to the frequency range under consideration). The parameters  $\omega_{pi}$ ,  $\omega_{oi}$  and  $\gamma_i$  are the ‘plasma’ frequency, the transverse frequency (eigenfrequency) and the linewidth (scattering rate) respectively of the  $i$ -th Lorentz oscillator. For the Drude term, which describes to response of the unbound (free) charge carriers, the  $\omega_{oi}$  is zero.

We shall distinguish between the formulas that satisfy the KK relation and the ones that do not. Of course, the safest way is to always fit the data with the KK-compliant functions. However, non-KK functions are also often used. For instance, the formula for an asymmetric oscillator:

$$\varepsilon(\omega) = \frac{\omega_p^2 e^{i\pi\phi}}{\omega_o^2 - \omega^2 - i\gamma\omega}$$

<sup>4</sup> Moreover, it makes no sense to have more parameters than the total number of experimental points.

is useful to estimate the asymmetry of the lineshape, when the data are analyzed in the range close to  $\omega_0$ . However, this ‘asymmetric’ term may cause serious problems with the asymptotic behavior at high frequencies and optical sum rules if applied to a broad range of frequencies.

Notably, even the DL function (Equation 2-18) is, strictly speaking, not Kramers-Kronig-compatible, if  $\varepsilon_\infty \neq 1$ . This ‘problem’ is commonly ignored, because  $\varepsilon_\infty \neq 1$  can be imitated by an oscillator sitting infinitely far away above the highest considered frequency.

Every formula-given function can be differentiated analytically, which is crucial to implement the Levenberg-Marquardt minimization (section 2.1.1).

#### 2.2.4. Variational dielectric functions (KK-constrained)

The method of variational dielectric functions is relatively new [4]. This section contains the detailed description of this technique.

Let me outline a situation, where this type of functions naturally comes about. Imagine that we are given a normal-incidence reflectivity spectrum  $R(\omega)$ , measured in the (wide enough) spectral range  $[\omega_{\min}, \omega_{\max}]$ , and we have to extract the corresponding dielectric function  $\varepsilon(\omega)$ .

One solution that is known since fifties [6] is to apply a special Kramers-Kronig transformation for the logarithm of the complex reflectivity. This method is very popular since it does not require much numerical work. However, its applicability is limited to the case of the normal-incidence reflectivity from a bulk isotropic sample. It fails for instance, if the angle of incidence is large enough [7].

A more universal way is to *fit* the reflectivity using some model function, which automatically satisfies the KK relation. We also have to use a Fresnel relation between  $\varepsilon(\omega)$  and  $R(\omega)$ , which properly takes into account particular experimental conditions (the angle of incidence, a possible substrate *etc.*). If we manage to match all essential features of the measured reflection coefficient (apart from the noise) then we expect the model dielectric function to closely resemble the true dielectric function of the system. As we cannot assume *a-priori* any specific model, the only way is to take some trial function with a lot of parameters that is ‘flexible enough’ to effectively *approximate* the true dielectric function.

One can take, for instance, the DL oscillator function (Equation 2-18) and put as many terms as it is necessary to obtain a nice fit to the data. All oscillator parameters  $\omega_{pi}$ ,  $\omega_{0i}$  and  $\gamma_i$  in this case are usually adjustable. In fact, this approach is doable, even though in some ‘unlucky’ cases the number of oscillators has to be very large. The reason for the success is the *completeness* of Drude-Lorentz functions in a sense that any physical dielectric function can be approximated with an arbitrarily good accuracy by some set of Lorentzians (although this statement is intuitively clear, the rigorous proof might be complicated).

However, this way of fitting has serious disadvantages. First of all, there are no obvious criteria on how many oscillators are really necessary. Secondly, as we add more and more oscillators to the model function, it becomes less and less clear how to guess the initial values

of parameters of every newcomer. Finally, if all parameters are allowed to change, it often happens that the fitting process causes an uncontrollable divergence of some of them<sup>5</sup>. To avoid it, one has to deliberately fix some parameters. It makes the fitting procedure rather tricky and ambiguous, preventing the routinely execution.

Now we take the first step towards the variational dielectric functions. In order to improve the multi-oscillator fitting strategy, we can put oscillators at *fixed* frequencies  $\omega_{0i} = \omega_i$ , ( $i = 1, \dots, N$ ) and set their linewidths  $\gamma_i$  to an also *fixed* value, which is of the order of the distance between adjacent frequency points (for instance  $\gamma_i = (\omega_{i+1} - \omega_{i-1})/2$ ). The idea, is that the  $i$ -th oscillator is ‘locally responsible’ for the spectral weight<sup>6</sup> near  $\omega_i$  (see Figure 2-2). In this way we ensure, that our model function is able to provide enough spectral weight to any frequency region. We are left with only plasma frequencies  $\omega_{pi}$  (which determine the spectral weights) allowed to change.

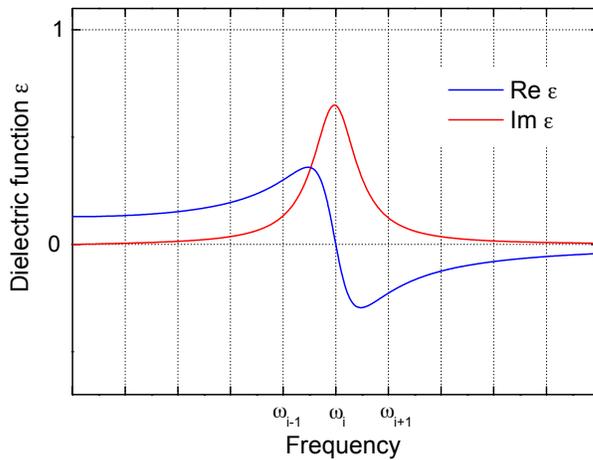


Figure 2-2 The real and imaginary parts of the Lorentz dielectric function  $\epsilon_i^{Lor}(\omega)$ , with the transverse frequency  $\omega_0 = \omega_i$  and the linewidth  $\gamma = (\omega_{i+1} - \omega_{i-1})/2$ .

The frequency mesh should cover the spectral range  $[\omega_{\min}, \omega_{\max}]$ <sup>7</sup> and should be dense enough to allow a very accurate fitting of the reflectivity spectrum. Ultimately, we can put even *one oscillator to every experimental point of the spectrum!* In this case the number of adjustable parameters is the same as the number of experimental points, so that the fitting can be in principle *exact* (of course, the data noise will be also fitted). Although we can still use the standard minimization procedure, described in section 2.1, such a fitting can be hardly referred to as ‘modeling’. I would rather call it ‘variational fitting’ or ‘free-shape fitting’ or something of the kind.

To make this variational approach really workable, we have to overcome the two major obstacles.

<sup>5</sup> Typically, it happens, when the transversal frequency goes beyond the experimental spectral range.

<sup>6</sup> By ‘spectral weight’ we mean, as usual, the integrated optical conductivity  $\int \sigma_1(x) dx \sim \int x \epsilon_2(x) dx$

<sup>7</sup> then  $\omega_1 = \omega_{\min}$  and  $\omega_N = \omega_{\max}$

The first problem is that the Lorentzian lineshape

$$\varepsilon_2^{i,Lor}(\omega) = \frac{\omega_{pi}^2 \gamma \omega}{(\omega_i^2 - \omega^2)^2 + \gamma^2 \omega^2}$$

is not very suitable to represent solely the ‘local’ spectral weight close to  $\omega_i$  because of the slowly decaying low- and high-frequency ‘tails’ (Figure 2-2). We need a more ‘localized’ function. Ideally, it should have no tails at all, *i.e.* to be non-zero only inside, let say, the small region  $[\omega_{i-1}, \omega_{i+1}]$ , adjacent to  $\omega_i$ .

A good candidate is a ‘triangular’ shape of  $\varepsilon_2(\omega)$  (Figure 2-3):

$$\varepsilon_2^{i,\wedge}(\omega) = \begin{cases} (\omega - \omega_{i-1})/(\omega_i - \omega_{i-1}) & , \quad \omega_{i-1} < \omega < \omega_i \\ (\omega_{i+1} - \omega)/(\omega_{i+1} - \omega_i) & , \quad \omega_i < \omega < \omega_{i+1} \\ 0 & , \quad otherwise \end{cases}$$

Equation 2-20

The corresponding  $\varepsilon_1(\omega)$  can be analytically obtained by applying the KK transformation (Equation 2-17):

$$\varepsilon_1^{i,\wedge}(\omega) = \frac{1}{\pi} \left[ \frac{g(\omega, \omega_{i-1})}{\omega_i - \omega_{i-1}} - \frac{(\omega_{i+1} - \omega_{i-1})g(\omega, \omega_i)}{(\omega_i - \omega_{i-1})(\omega_{i+1} - \omega_i)} + \frac{g(\omega, \omega_{i+1})}{\omega_{i+1} - \omega_i} \right],$$

Equation 2-21

where  $g(x, y) \equiv (x + y) \ln|x + y| + (x - y) \ln|x - y|$ .

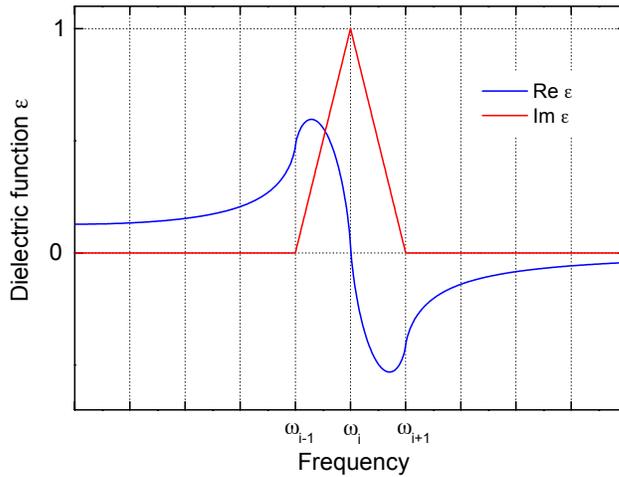


Figure 2-3. The real and imaginary parts of the ‘triangular’ dielectric function  $\varepsilon^{i,\wedge}(\omega)$ , described in the text.

Now we can construct the desired variational function as a linear superposition of triangular functions localized at all frequencies  $\omega_i$

$$\varepsilon_{\text{var}}(\omega) = \sum_{i=1}^N A_i \varepsilon^{i,\wedge}(\omega)$$

Equation 2-22

and consider coefficients  $A_i$  as *free* parameters. To ensure that  $\varepsilon_2(\omega) \geq 0$ , we require that all  $A_i \geq 0$ . It is convenient to set them to zero at the boundaries:  $A_1 = A_N = 0$ , ensuring that  $\varepsilon_2(\omega)$  vanishes at  $\omega_1$  and  $\omega_N$ <sup>8</sup>.

As is schematically shown on Figure 2-4, the imaginary part of  $\varepsilon_{\text{var}}(\omega)$  is a piecewise curve going through  $N$  points  $\{\omega_i, A_i\} (i = 1, \dots, N)$ . One can see that  $\text{Im} \varepsilon_{\text{var}}(\omega)$  in between the reference frequency points  $\omega_i$  is simply given by a linear interpolation. The  $\text{Re} \varepsilon_{\text{var}}(\omega)$  is its exact KK transform.

The free parameters of  $\varepsilon_{\text{var}}(\omega)$  are the values of  $\varepsilon_2$  at every frequency point  $\omega_i$ . There are altogether  $N - 2$  parameters. Obviously, this construction is extremely *flexible* and totally *model-independent*, just what we aimed to obtain!

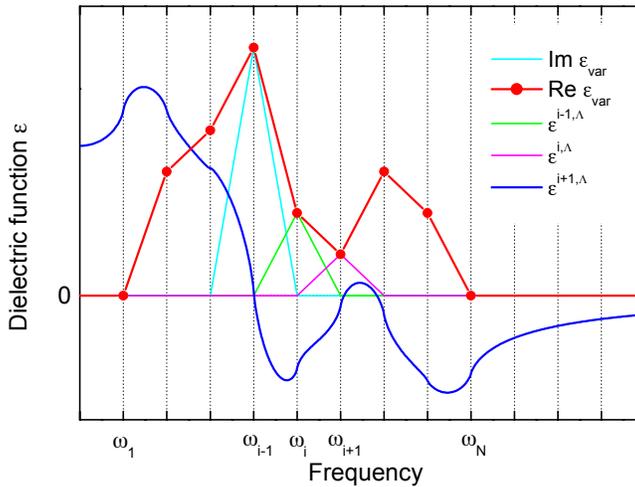


Figure 2-4. A KK-constrained variational dielectric function. The  $\text{Im} \varepsilon_{\text{var}}(\omega)$  is composed of many triangular functions  $\text{Im} \varepsilon^{i,\wedge}(\omega)$  and the  $\text{Re} \varepsilon_{\text{var}}(\omega)$  is the exact KK-transform of  $\text{Im} \varepsilon_{\text{var}}(\omega)$ .

The second problem is that the so-constructed variational dielectric function (Equation 2-22) totally ignores all spectral weight beyond the frequency range  $[\omega_{\min}, \omega_{\max}]$ . However, according to the KK relation (Equation 2-17), the non-zero  $\varepsilon_2(\omega)$  outside this region influences  $\varepsilon_1(\omega)$  inside it, and is, therefore, essential to calculate reflectivity  $R(\omega)$ , which depends on both  $\varepsilon_1(\omega)$  and  $\varepsilon_2(\omega)$ .

<sup>8</sup> a non-zero value of  $\varepsilon_2$  at the boundary would result in a discontinuity of  $\varepsilon_2$  and therefore in an unwanted divergence of  $\varepsilon_1$

The problem can be circumvented by doing the fitting in *two steps*. Initially, the spectra are fitted in a conventional way by some formula-defined dielectric function  $\varepsilon^{(0)}(\omega) = \varepsilon_{\text{mod}}(\omega)$ . If the match is reasonably good, then we can assume that  $\varepsilon_{\text{mod}}(\omega)$  has correct frequency dependence outside the considered spectral range, even though some fine details of the experimental curve are not fitted very well. Then we can fix all parameters of  $\varepsilon_{\text{mod}}(\omega)$  and *add* a variational function to it:

$$\varepsilon^{(1)}(\omega) = \varepsilon_{\text{mod}}(\omega) + \varepsilon_{\text{var}}(\omega).$$

Now the  $\varepsilon_{\text{var}}(\omega)$  acts as a *small* correction to the initial model  $\varepsilon_{\text{mod}}(\omega)$ . When we do a variational fitting of the reflectivity spectrum with  $\varepsilon^{(1)}(\omega)$ , the ‘KK influence’ of the low- and high-frequency spectral weights on  $\varepsilon_1(\omega)$  inside  $[\omega_{\text{min}}, \omega_{\text{max}}]$  is already accounted for by  $\varepsilon_{\text{mod}}(\omega)$ .

Because  $\varepsilon_{\text{var}}(\omega)$  is now added to  $\varepsilon_{\text{mod}}(\omega)$ , which is the dominant contribution to the total dielectric function, the parameters  $A_i$  are not necessarily positive (which was initially required by the condition  $\varepsilon_2(\omega) \geq 0$ ), but can be *negative* as well.

From the Equation 2-22 we can get a simple analytical formula for the first derivatives of the dielectric function  $\varepsilon_{\text{var}}(\omega)$  with respect to the parameters  $A_i$ , which are required by the Levenberg-Marquardt procedure (section 2.1.1):

$$\frac{\partial \varepsilon^{\text{var}}(\omega)}{\partial A_i} = \varepsilon^{i,\wedge}(\omega).$$

At the end we minimize the chi-square with respect to all  $N-2$  parameters, thus obtaining ‘the best’ KK-related dielectric function, which fits the reflectivity spectrum.

In the above example we discussed the fitting of a single reflectivity spectrum. As was mentioned before, in the particular case of a normal-incidence reflectivity spectrum of an isotropic sample, (almost) the same result could be obtained by the ‘conventional’ KK technique. However, an important advantage of the new method is that it can be applied to virtually *any type of optical spectra*, or a *combination of them!*

The typical number of parameters that are adjusted in the fitting by variational functions is very large – up to the number of experimental points, which might amount to *few thousands*. Although such an enormous number of parameters seem to make the fitting procedure prohibitively slow, we found that it nevertheless converges within acceptable time limits. This is yet another reason to admire, how good the LM method is!

The optimization of the mesh of anchor frequency points  $\omega_1 \dots \omega_N$  is essential for correct work of the variational fitting algorithm. On one hand, the mesh should be dense enough to enable the variational function to fit all important spectral details. On another hand,  $N$  cannot be too large. The first reason is that the calculation time is growing quickly as a function of the number of parameters to be adjusted. The second reason is that an excessive density of anchor points may give rise to numerical instabilities, most typically, fast spurious oscillations of the resulting  $\varepsilon_1$  and  $\varepsilon_2$  as functions of  $\omega$ . In other words, the variational function should not be

'too flexible'. According to our experience, numerical instabilities can often be avoided if the number of anchor frequencies is *two times smaller*, than the total number of experimental points. Each anchor point must correspond to at least one experimental point, contributing to the chi-square (Equation 2-12).

### 2.2.5. 'Not-KK-constrained' variational dielectric functions

Although the title of this section looks 'frightening', the model function developed here is in fact much easier to understand and deal with than the one of the previous section.

The variational functions that we told so far about were always 'forced' to satisfy the Kramers-Kronig relation. However, one can also consider a not KK-constrained variational function, where both  $\varepsilon_1(\omega)$  and  $\varepsilon_2(\omega)$  are 'flexible' and independent (Figure 2-5). Now it is not necessary to set  $\varepsilon_1(\omega)$  and  $\varepsilon_2(\omega)$  to zero at the boundaries  $\omega_1$  and  $\omega_2$ . If we consider a mesh of frequencies  $\omega_i$  ( $i = 1, \dots, N$ ), as in the previous section, then the dielectric function of this kind is parameterized by the  $2N$  values  $\varepsilon_1(\omega_i)$  and  $\varepsilon_2(\omega_i)$ .

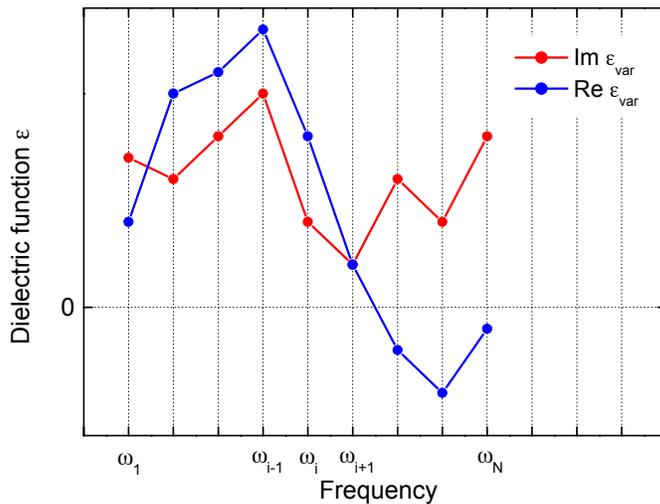


Figure 2-5 A non-KK-constrained variational dielectric function, which is composed of many triangular functions. The real and imaginary parts are completely independent.

Of course, now there is no guarantee, that the resulting  $\varepsilon_1(\omega)$  and  $\varepsilon_2(\omega)$  will be KK-related. As a result, one can get sometimes a totally unphysical result (for instance, when there is not enough data or the systematic error bars are large). On the other hand, such 'KK-relaxed' function is much easier to deal with, than a KK-constrained one, because  $\varepsilon_1(\omega)$  does not depend non-locally on the values of  $\varepsilon_2(\omega)$  in the entire range anymore.

Although it is generally important to verify that the resulting function satisfies the KK relations, it is not always necessary and even plausible. For instance, in some optical techniques two or more values are measured independently at every frequency. The well known example is the ellipsometry, where two parameters ( $\psi$  and  $\Delta$ ) are obtained. Another example is the method, where the reflectivity  $R$  and transmission  $T$  are measured on the same sample. In both techniques the  $\varepsilon_1$ ,  $\varepsilon_2$  are derived from the measured quantities at every frequency

independently, without the usage of the KK relations. If the spectral range, where the measurements are done, is too small, it is even impossible to check for the KK relations<sup>9</sup>.

In these cases the fitting of the data with a KK-relaxed variational dielectric function obviously does the right job. Even though the fitting now formally involves  $2N$  parameters, it is in fact equivalent to the ‘point-by-point’ series of  $N$  independent fitting problems which involve 2 parameters ( $\varepsilon_1(\omega_i)$  and  $\varepsilon_2(\omega_i)$ ) each.

## 2.3. Differential modeling

Suppose, we measure some optical spectra  $S(\omega)$  (for example, reflection, transmission, ellipsometry *etc.* or their combinations), from which we can obtain the dielectric function  $\varepsilon(\omega) = \varepsilon_1(\omega) + i\varepsilon_2(\omega)$  by a modeling, described in section 2.2. If the dielectric function depends on some externally controlled parameter  $P$  (most typically, temperature, but might be also pressure, magnetic field *etc.*), then it is often necessary to measure the *change* of  $\varepsilon$  due to the change of  $P$ , let say, from  $P_1$  to  $P_2$ :

$$\Delta\varepsilon(\omega) \equiv \varepsilon(\omega, P_1) - \varepsilon(\omega, P_2),$$

or, equivalently (if the changes are small), to obtain the derivative  $\partial\varepsilon/\partial P$ . Sometimes this technique is called ‘modulation spectroscopy’ because the optical properties are ‘modulated’ by an external parameter.

In principle, we can obtain the model dielectric functions  $\varepsilon_{\text{mod}}(\omega, P_1)$  and  $\varepsilon_{\text{mod}}(\omega, P_2)$  by the fitting of  $S(\omega, P_1)$  and  $S(\omega, P_2)$  separately and assume that

$$\Delta\varepsilon(\omega) \approx \Delta\varepsilon_{\text{mod}}(\omega) \equiv \varepsilon_{\text{mod}}(\omega, P_1) - \varepsilon_{\text{mod}}(\omega, P_2).$$

However, this approach completely ignores, that the error bars of the *differential* spectra

$$\Delta S(\omega) = S(\omega, P_1) - S(\omega, P_2)$$

have very little to do with the ones of the genuine spectra  $S(\omega)$ . In fact, they are usually *much smaller*, because most systematical errors are cancelled out!. As a result, the  $\Delta_{\text{mod}}\varepsilon(\omega)$  may look rather different from  $\Delta\varepsilon(\omega)$ .

If the  $\Delta S$  and  $\Delta\varepsilon$  are small, then it might be better to fit the  $\Delta S(\omega)$  directly by using a so-called *differential* model for  $\Delta\varepsilon(\omega)$ . The procedure can be done in two steps. Initially, the *base* model  $\varepsilon_{\text{base}}(\omega)$  can be deduced by the usual fitting of  $S(\omega, P_1)$ . It can be either formula-defined model (section 2.2.3), or a variational model (sections 2.2.4, 2.2.5). Apart from the values of  $\varepsilon_1(\omega)$  and  $\varepsilon_2(\omega)$ , the base model also provides the derivatives (sensitivities) of the measured spectrum with respect to the  $\varepsilon_1$  and  $\varepsilon_2$ :

<sup>9</sup> When the ellipsometry data are taken in a broad range, it *does* make sense to fit it with a KK-constrained function, because it may reveal serious systematic measurement errors, if any.

$$\alpha_1(\omega) \equiv \frac{\partial S(\omega)}{\partial \varepsilon_1(\omega)}, \quad \alpha_2(\omega) \equiv \frac{\partial S(\omega)}{\partial \varepsilon_2(\omega)}$$

Using these derivatives, the differential spectra can be calculated in the first order of a Taylor expansion:

$$\Delta S(\omega) \approx \alpha_1(\omega)\Delta\varepsilon_1(\omega) + \alpha_2(\omega)\Delta\varepsilon_2(\omega).$$

Equation 2-23

Then one can associate an extra (*differential*) model  $\varepsilon_{diff}(\omega)$  with  $\Delta\varepsilon$  and adjust its parameters in order to fit  $\Delta S(\omega)$ , using Equation 2-23. The error bars of  $\Delta S(\omega)$  have to be estimated separately. In the fitting with the differential model, the role of the base model is to provide only the (fixed) derivatives  $\alpha_1(\omega)$  and  $\alpha_2(\omega)$ .

Formally,  $\varepsilon_{diff}(\omega)$  has to be treated as a usual dielectric function, because the physical requirements to it are almost the same as the usual requirements to the dielectric functions (see 2.2.1). For instance, it is important that  $\varepsilon_{diff}(\omega)$  satisfies the Kramers-Kronig relation. The only obvious difference is that  $\Delta\varepsilon_2$  can be negative (unlike  $\varepsilon_2$ ), and *both*  $\Delta\varepsilon_1$  and  $\Delta\varepsilon_2$  must vanish at very high frequencies.

---

## 3. Tutorial

This chapter shows how to solve typical problems using RefFIT. Each section is devoted to a particular task. I suggest you to go through the sections sequentially, since the last parts do not repeat the detailed explanations, given in the beginning. It is especially important to read the first section (3.1), which deals with the most basic RefFIT objects.

Of course, before the going through this tutorial, you should install RefFIT on your computer. This simple procedure is explained in section 4.2.

### 3.1. Drude-Lorentz fitting of a reflectivity spectrum

As I already mentioned, RefFIT was created as a reflectivity fitting routine with the Drude-Lorentz dielectric function (Equation 2-18). The normal-incidence reflectivity  $R(\omega)$  is expressed via the dielectric function  $\varepsilon(\omega)$  according to Fresnel formula:

$$R = \left| \frac{1 - \sqrt{\varepsilon}}{1 + \sqrt{\varepsilon}} \right|^2.$$

It is one the most common data analysis that solid-state infrared spectroscopists do. So why don't we try it first?

In order to do it in RefFIT, one should first open a model window by choosing "Model" from the "Window" menu (Figure 3-1). It contains a particular DL model that can be used later on for plotting graphs, fitting data and so on. In this window we can add or delete the Lorentzian terms and manually edit parameters. The "Einf" field stands for  $\varepsilon_\infty$ ; "Wo", "Wp" and "G" designate the transverse frequency  $\omega_{0i}$ , the plasma frequency  $\omega_{pi}$  and the scattering rate  $\gamma_i$  respectively, measured in inverse centimeters<sup>10</sup> ( $\text{cm}^{-1}$ ). By the way, 1 eV is about 8000  $\text{cm}^{-1}$ . Let us keep for a while the initial values (Figure 3-1).

---

<sup>10</sup> Inverse centimeters are apparently invented by the infrared spectroscopists to make the life of normal people more difficult ☺.

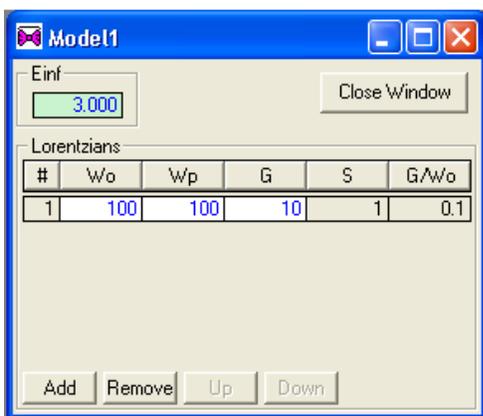


Figure 3-1 A newly created model window with default parameters.

Next we create a new graph, by choosing “Graph” from the “Window” menu (Figure 3-2). In a graph you can plot curves, generated by the models and datasets.

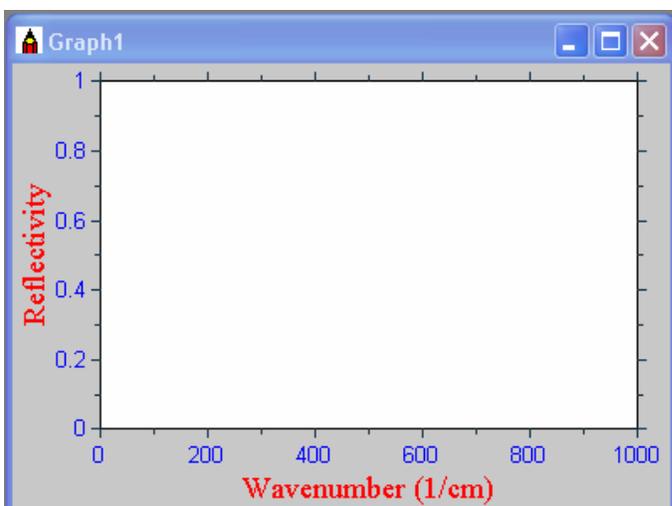


Figure 3-2 An empty graph window.

If you want to set up the graph contents, just double-click on the white area. You will get a window, as shown at Figure 3-3. Choose the first item “(Model [R] “Model1”)” from the list “Available curves” and transfer it to the list “Graph curves” by clicking the  button. The curve will appear on the graph.

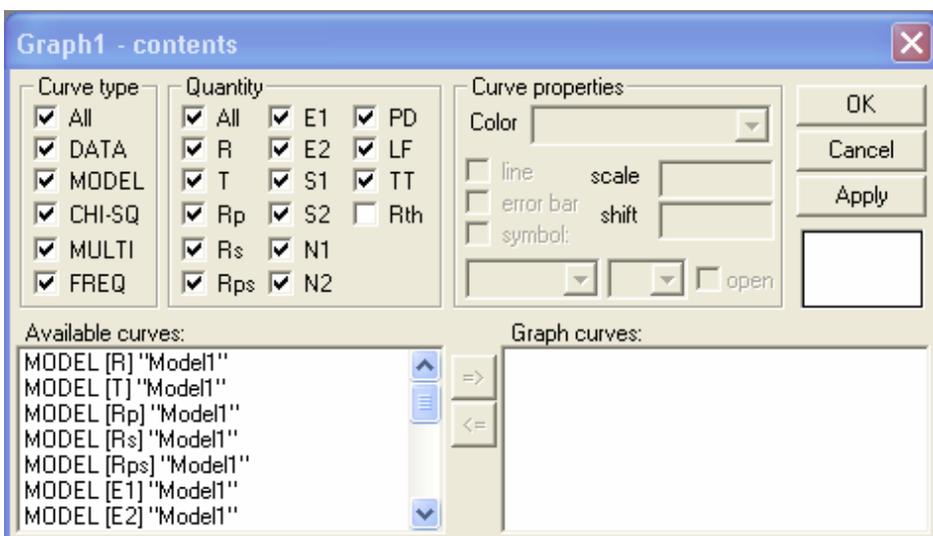


Figure 3-3 Graph contents window is used to set up the graph curves.

Now we would like to edit the model parameters. If you do not like, that the plasma frequency of the Lorentzian in the model is so small ( $100 \text{ cm}^{-1}$ ), you can change it to, let say, 1000. For that you can either double-click on the corresponding field in the “Model1” window, or set the marker to this field and start typing a new value. The graph will be immediately redrawn after you finish editing by pressing “Enter” or in any other way. You can do the same with other parameters, or add extra Lorentzian terms. Any time you change something the graphs will react accordingly.

You might have already noticed an additional window, called “Parameter control” (Figure 3-6) that appeared simultaneously with the “Model1” window. It is intended to change the characteristics of the currently selected parameter, such as its value as well as the minimal and maximal limits *etc.* On the left side you can see a trackbar, which works like a microphone volume control. Just drag it up and down in order to change the parameter value. The graph starts to show a movie, which you direct yourself!

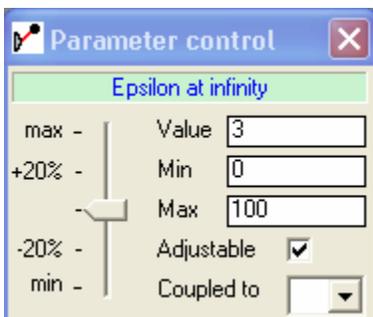


Figure 3-4 Parameter control window allows one to change the properties of a currently select parameter.

After finishing with one parameter click to another one and continue the procedure. You might even not need a keyboard to get the desired spectrum shape.

If you need to change the graph properties, such as axes scales, titles etc, you can double-click on the gray area in the “Graph1” window and enter the corresponding properties.

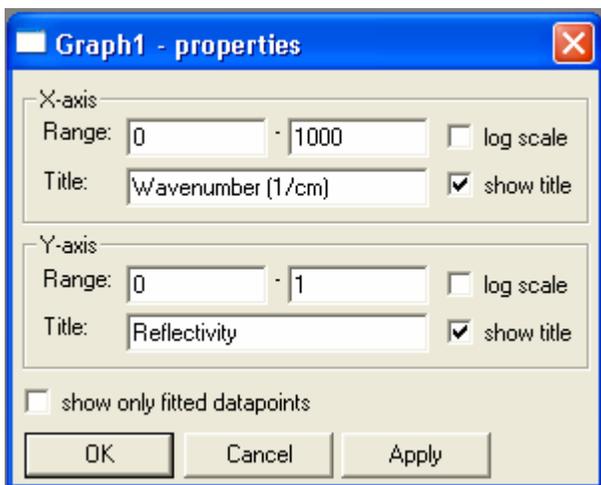


Figure 3-5 In the “Graph properties” window one can modify the axis scales and titles.

The playing around with parameters in an abstract model may start to get boring after a minute or so. To be more close to the real life let us load an experimental reflectivity curve. No doubt, you have your own beautiful spectra, but I would propose to take my own data file as a start. OK, we should open the dataset manager by clicking at “Dataset manager” item of the “Window” menu. After that we get a window, shown at Figure 3-6. It has 10 slots for datasets.

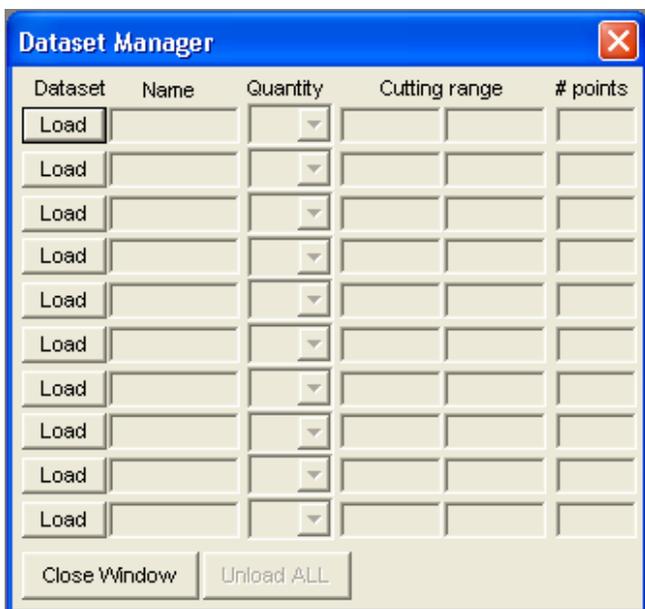


Figure 3-6 Dataset manager: no datasets are loaded yet.

The rest is straightforward. Click the  button in the first slot that will invoke the “Load Dataset” window (Figure 3-7).

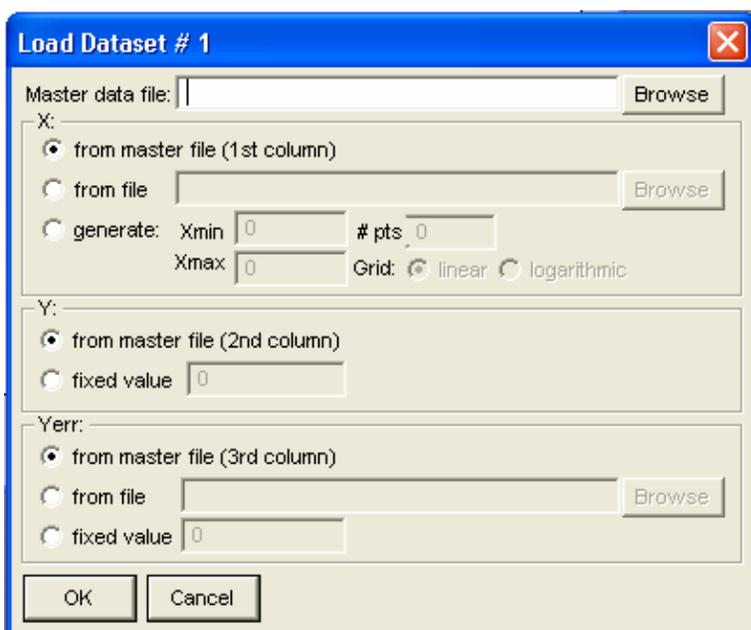


Figure 3-7

Click the **Browse** button and in the “Browse” window pick up the master file ‘R1.DAT’ from the directory “TUTORIAL\PART1”. Click the **OK** button. The dataset will be loaded now to the first slot (Figure 3-8). Note that the program assumed that this file contained reflectivity by checking the first filename character – “R”. Sometimes the data type is determined incorrectly. If this happens, then you must choose the right quantity in the “Quantity” field.

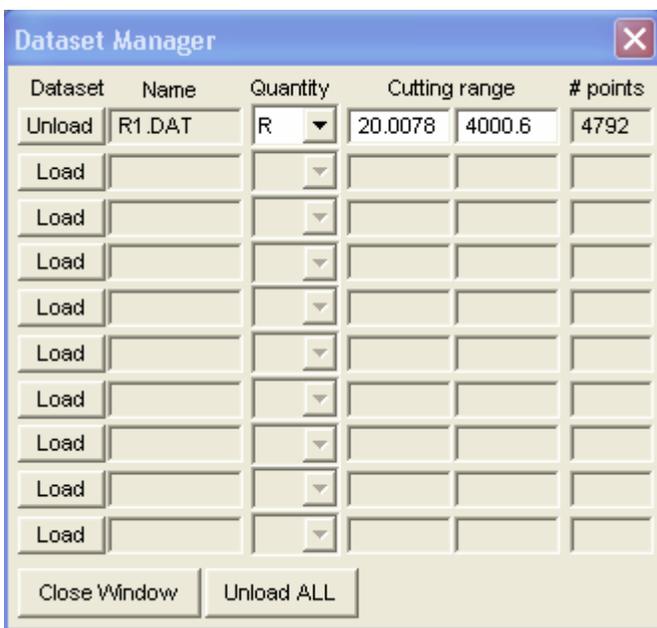


Figure 3-8 The dataset manager with a reflectivity dataset loaded into the first slot.

You may close the ‘Dataset manager’ window in order to save some space on the screen. Don’t worry; it will not cause the unloading of our dataset.

Now we would like to show this curve on the same graph. Let's go back to the "Graph1" window and double-click anywhere in the white space. You can notice, that an extra item "DATA [R] "R1.DAT"" has appeared in the "Available curves" list of the "Graph contents" window (Figure 3-9). Very nice, this is our dataset. Move it to the "Graph curves" list and click the  button. You may not like the appearance of the experimental curve. If so, you have to pop up "Graph contents" again, select "DATA [R] "R1.DAT"" in the "Graph curves" list and set the curve parameters according to your taste. Let us set the parameters as shown in Figure 3-9.

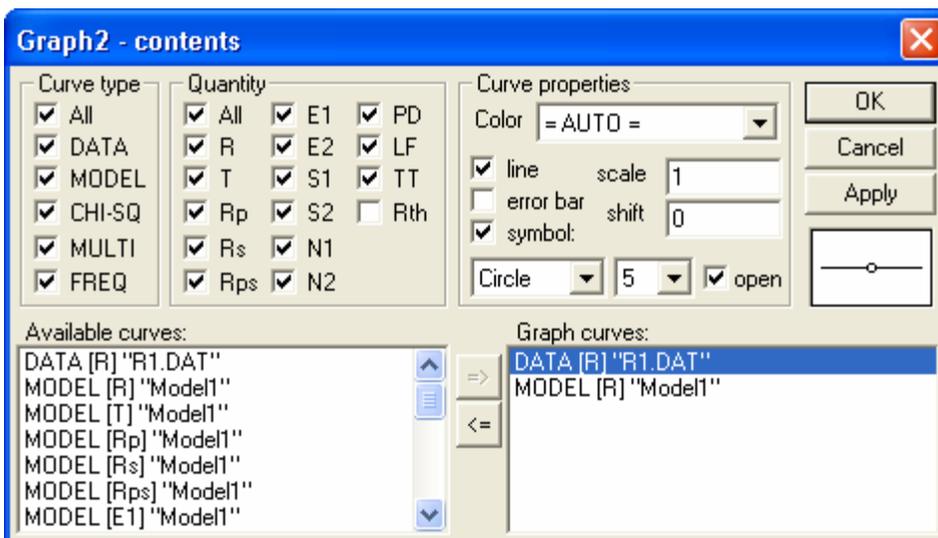


Figure 3-9 The "graph contents" window with the data and model curves selected for plotting.

Next I would suggest you to play with the model parameters manually in order to make the model curve as close as possible to the experimental one. To be specific, keep only two Lorentzians in the model. I'll see you in five minutes...

...OK, I am back. I hope, your "Graph1" looks similar to Figure 3-10.

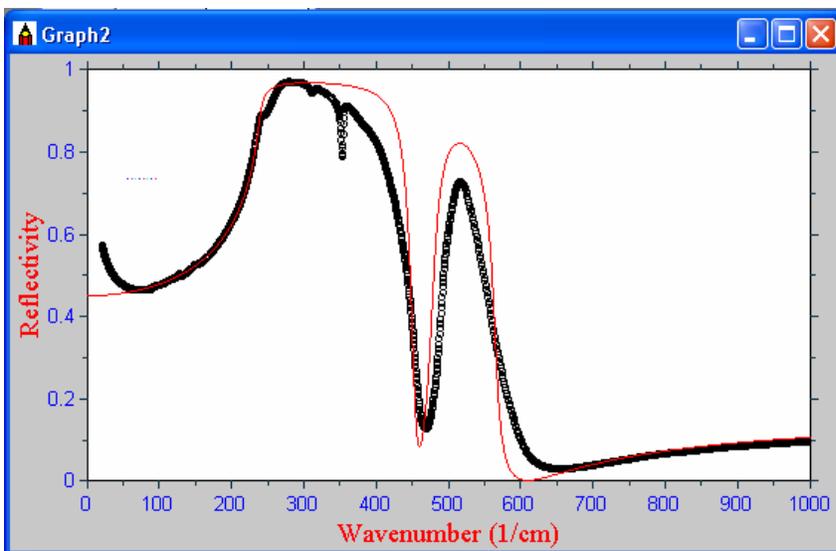


Figure 3-10 The match between the experimental and model reflectivity curves (after a manual parameter adjustment).

If you are still far from this fitting match, you can type the parameters that I obtained (Figure 3-11). You must have already figured out that these two Lorentzian terms correspond to the two strongest optical phonon modes.

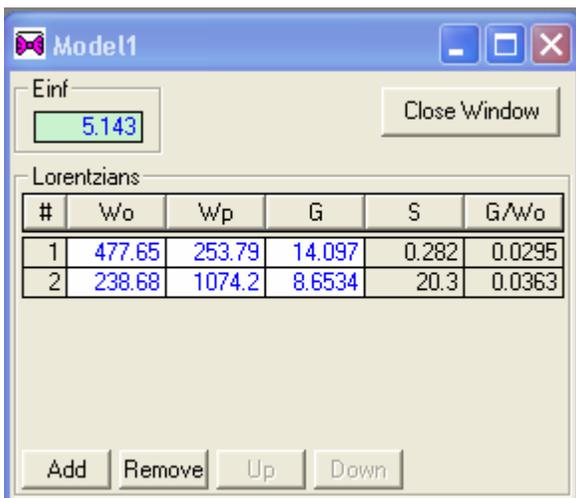


Figure 3-11 The model with the two manually adjusted Lorentzians.

So, we did a first step, and it is time to save the model. The way to do it is to activate our model window and press the **F2** on the keyboard (do not ask me, why!). You will have the possibility to choose the path and the filename. I would suggest calling it “MODEL1.RFM”<sup>11</sup>, but it is up to you. If you now do something wrong or the program crashes, you can load this saved model by pressing the **F3** button in the active model window.

Of course, we would expect from RefFIT to do an automated fitting. This is our ultimate goal, after all. Let us show the ‘Fit’ window by clicking on ‘Fit’ in the ‘Window’ menu (Figure 3-12).

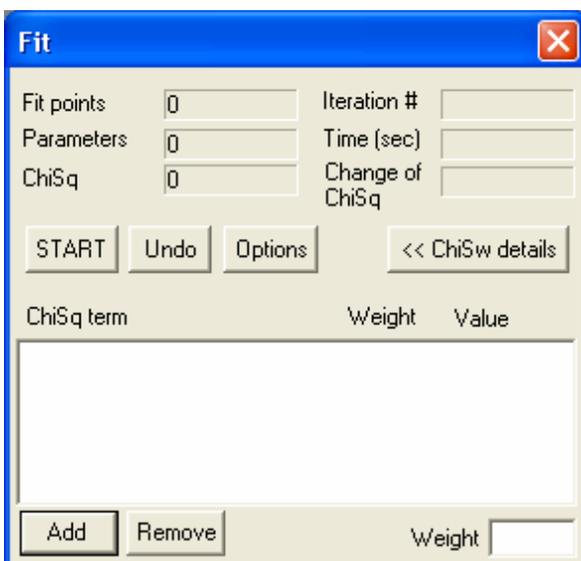


Figure 3-12 The “Fit” window. The fitting task is not specified yet.

<sup>11</sup> “RFM” is a standard extension for the RefFIT model files.

It is tempting to press immediately the **START** button, but this wouldn't work until we tell the program that we want to fit the "R1.DAT" dataset to the model "Model1". The way to do it is to add the corresponding term to the chi-square (Equation 2-12). Click the **Add** button. The window "Add ChiSq term" will show up (Figure 3-13). In the list "Available ChiSq terms" there is only one possibility: "[R] "R1.DAT" - "Model1"" (there would be much more if more models and datasets were available). Select this item and click **OK**.



Figure 3-13 Adding chi-square terms.

Now the fitting problem is well specified (Figure 3-14). The number of points in the spectrum is 4792 and the number of fitted parameters is 7. You can also see how big the chi-square is.

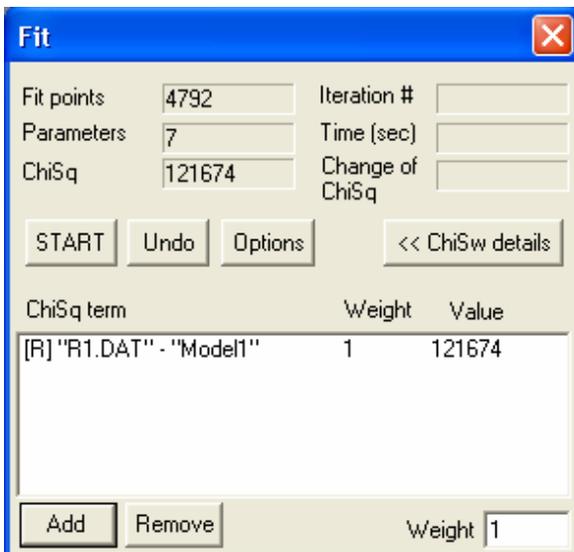


Figure 3-14 The "Fit" window after the adding of a chi-square term.

Now we can press the **START** button. You might notice that the match considerably improved (Figure 3-15) and the chi-square has decreased more than 10 times. It is a good idea save the model again, perhaps with a different name, *e.g.*, "MODEL2.RFM".

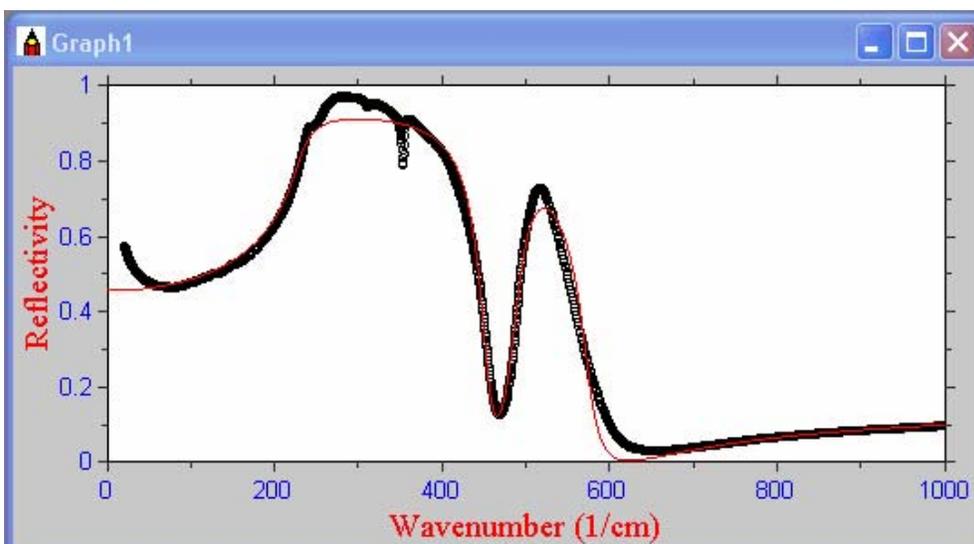


Figure 3-15 The model and experimental reflectivity curves after the automated fitting. The model contains only two oscillators.

Honestly, I am still not quite happy with the fitting quality. It looks like that our sample has a small metallic conductivity, because reflectivity is bending upwards at very low frequencies. Let us put an extra Drude peak, corresponding to the response of charge carriers. It is a special case of Lorentzian with zero transverse frequency  $\omega_0$ . What we should do is to simply add an extra Lorentzian to the model (by clicking the  button) and set  $\omega_0$  explicitly to zero. You may try to adjust  $\omega_p$  and  $\gamma$  in order to get a better match. If you find it difficult, I suggest setting both of them to  $500 \text{ cm}^{-1}$ .

Now it makes sense to start the automated fitting again. But wait! We want  $\omega_0$  of the Drude term to be always zero, while at the moment it is an adjustable parameter. There are several ways to tell RefFIT that we do not want a parameter to be varied. Perhaps, the fastest is to click on it with the right mouse button. Another way is to move the marker (light-green) to a specific parameter and press the **SPACEBAR**. The third option is to uncheck “Active in fit” in the Parameter window (Figure 3-4). The fixed parameter always looks red.

OK, we can start fitting. Wow, we’ve got an improvement (Figure 3-16)! The model now is shown at Figure 3-17. If you prefer to have oscillators to be sorted by their transverse frequency ( $\omega_0$ ), then click on the column header “ $\omega_0$ ”. It is time to save the model again.

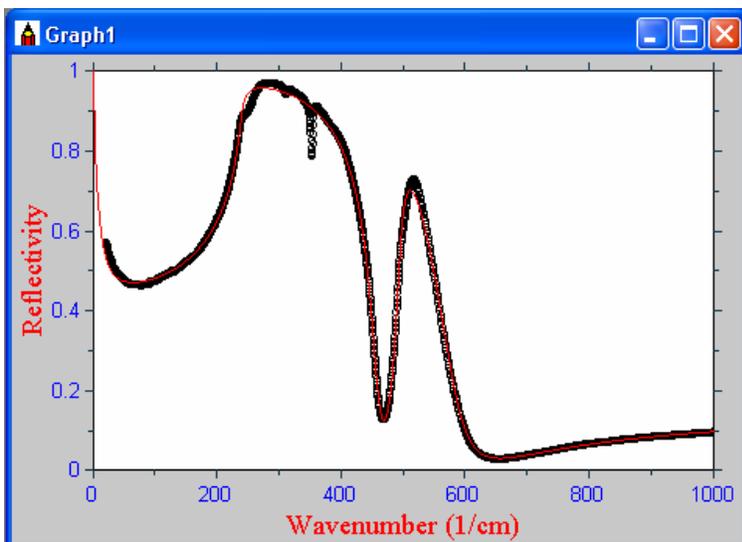


Figure 3-16 The best model and data match for the model with one Lorentz terms and one Drude term.

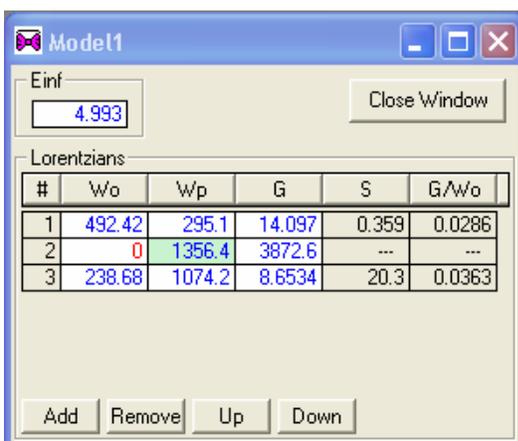


Figure 3-17 The model with one Lorentz terms and one Drude term, giving the best match to the data.

Having obtained such a fit, you might want to see the corresponding dielectric function. Let us open a new graph window (by choosing ‘Graph’ from the ‘Window’ menu). Open the “Graph properties” window by double-clicking on the gray area. Change the Y-axis range to [-20;+20] and Y-axis title to “E1,E2” (Figure 3-18).

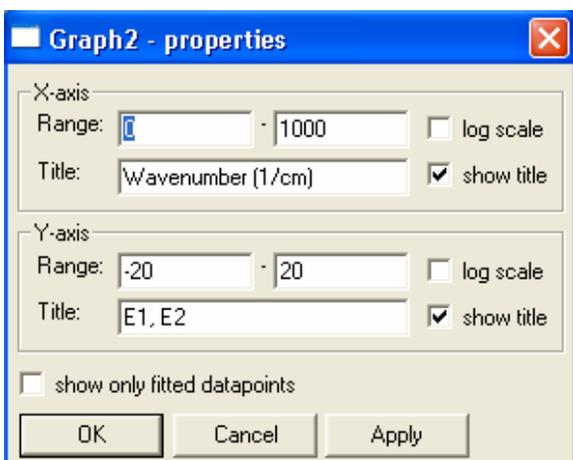


Figure 3-18 The “Graph properties” window.

Then open the “Graph contents” window by double-clicking on the white area (Figure 3-3) and move items “MODEL [E1] “Model1”” and “MODEL [E2] “Model1”” to the list “Graph curves”. Now we can see the real and imaginary parts of the dielectric function ( $\epsilon_1$  and  $\epsilon_2$ ) on the same graph (Figure 3-19). Of course, in this way you can plot also other curves, like conductivity, penetration depth, loss function *etc.*

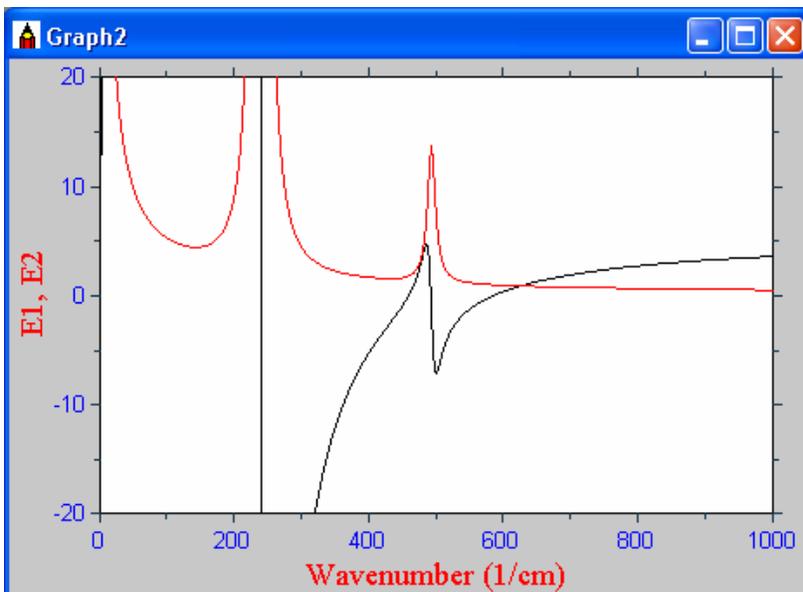


Figure 3-19 The real and imaginary part of the dielectric function, plotted on the same graph.

In order to have some outcome you may want export some curves, calculated by a model. To do this you should activate a graph window and press the **SPACEBAR**. You can then specify the file name in the “Export Graph to ASCII” window. All the model curves on the graph will be stored as separate columns in this file. At this moment it is not yet possible to specify, which set of frequencies will be used to generate the model curves. The computer will take the frequency range exactly the same as on the horizontal scale of the graph. The resolution will correspond to the resolution of your screen. Please, do not complain. It used to be a temporary solution, but I did not make a better one. However, you can export data with an adjustable set of frequencies using a macro (see section 4.11.6).

Congratulations! You have fitted a real experimental curve! You may try to improve the match by fitting, for example, the sharp phonon dip at around  $350\text{ cm}^{-1}$  or other structures.

By the way, this was the reflectivity spectrum of a high-temperature superconductor  $\text{La}_{1.85}\text{Sr}_{0.15}\text{CuO}_4$  at 100 K for electric field polarization perpendicular to the  $\text{CuO}_2$ -planes (Ref. [8]). At this temperature it was not yet superconducting. If you want to fit the reflectivity of the sample in the superconducting state, I prepared another file for you (“R2.DAT”). Hint: you will need to introduce an extra Drude term with  $\gamma = 0$ , which corresponds to the contribution of superconducting electrons.

### 3.2. Simultaneous fitting of several data types

This time we consider an example, where several spectra of *different* types are fitted *simultaneously* in order to improve the accuracy of the resulting dielectric function<sup>12</sup>. The measurements were done on a single-crystalline sample of compound  $\text{CeIrIn}_5$  at low temperature  $T = 35\text{ K}$  (see Ref. [9]). This material is a rather good metal, which results in a very high reflectivity at low frequencies.

The reflectivity  $R$  at nearly-normal incidence was measured in the far- and mid-infrared spectral range  $25 - 5500\text{ cm}^{-1}$  (data file “R.DAT”). The ellipsometric measurements provided the spectra of  $\varepsilon_1$  and  $\varepsilon_2$  from  $6000$  to  $36000\text{ cm}^{-1}$ , which includes the near-infrared, the visible and the soft ultraviolet ranges (files “E1.DAT” and “E2.DAT”). In addition, the static (*i.e.* zero-frequency) conductivity  $\sigma_{DC}$ , equal to about  $4.8 \cdot 10^4\ \Omega^{-1}\text{cm}^{-1}$ , was obtained from conventional electrical measurements (file “S1DC.DAT”, which contains only one point).

It is important to fit spectra simultaneously, since it helps to remove the uncertainty, given by the fact that in the far-infrared only one quantity (reflectivity) is measured, which is in general not enough to get both  $\varepsilon_1$  and  $\varepsilon_2$ <sup>13</sup>. Therefore, our goal is to fit all available data with the same model dielectric function. For simplicity, we shall use only the Drude-Lorentz formula (Equation 2-18).

To begin with, let us load all spectra that we are going to use, one by one. First we load the reflectivity spectrum.

Please, open the “Dataset Manager” window (Figure 3-6) and click the  button in the first slot. In the window “Load Dataset #1” (shown in Figure 3-7) click the  button on the top. Select the master file “R.DAT” from the directory “TUTORIAL\PART2”. As this file turns out to be rather big, we will generate a smaller dataset with different set of frequencies. In the “X” group box check the button “generate” and type 25 in the field “Xmin”, 5000 in the field “Xmax” and 1000 in the field “# pts”. Check the “linear” grid. It means that the dataset will contain 1000 evenly (linearly) distributed frequency points in the range 25-5000  $\text{cm}^{-1}$ . As a result, the “Load Dataset #1” should look like in Figure 3-20. Click the  button.

<sup>12</sup> This example was provided by F.P.Mena

<sup>13</sup> One can say, that the knowledge of the static conductivity and as well as high-frequency dielectric function helps to effectively ‘anchor’ the complex phase of the reflectance coefficient, which is difficult to measure in the far-infrared

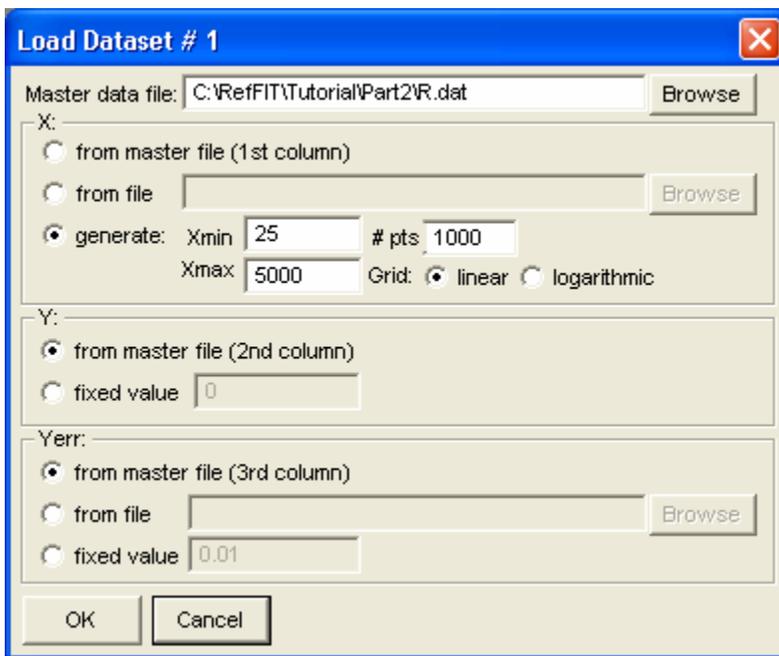


Figure 3-20 Loading the reflectivity dataset.

Load dataset “E1.DAT” to slot #2. It contains not so many frequency points, so we would like to load it as it is. Just do not check “generate” button in the “X” group box. Load files “E2.DAT” and “S1DC.DAT” in the same way. If all downloads are OK, the “Dataset Manager” window should look as in Figure 3-21.

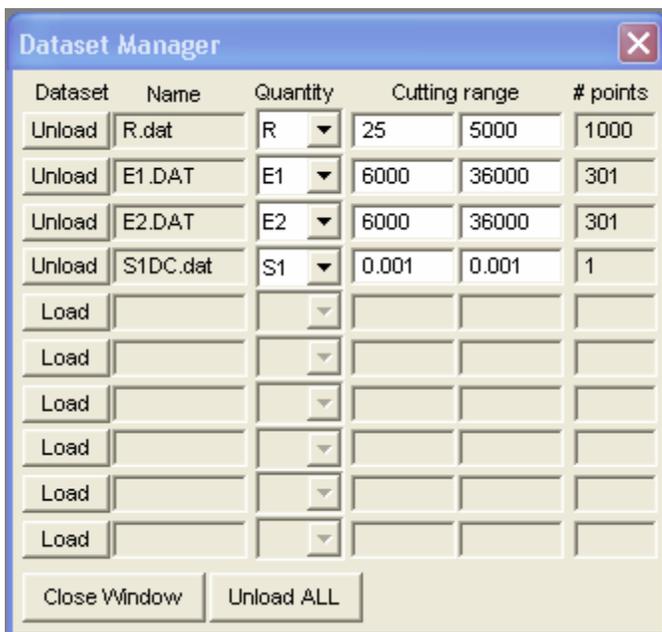


Figure 3-21 The dataset manager after all four datasets are loaded.

We will need one “Model” window to model the dielectric function of the sample. Could you create a new “Model” window? If you do not remember how, look at the first Tutorial section (3.1) or at the Reference (section 4.5). Let us postpone the editing of the model parameters until we create the graphs.

I suggest to create *three* graphs: one for the reflectivity, one for the dielectric function (both  $\epsilon_1$  and  $\epsilon_2$ ) and one for the optical conductivity. The X and Y ranges must be set according to the dataset ranges. The procedure to set the graph properties and curves was also described in section (3.1) and in the Reference (section 4.9).

Let me ask you to set the X-range of the first graph to [0; 5500] and the Y-range to [0.65; 1.0]. The default axis titles of “Wavenumber (1/cm)” and “Reflectivity” are OK here. Create experimental model reflectivity curves (“DATA [R] “R.DAT”” and “MODEL [R] “Model1”” respectively).

For the second graph set the X-range and the Y-range to [5000; 36000] and [-30; +30] correspondingly; change the Y-axis title to “Eps1, Eps2”. Create two dataset curves (“DATA [E1] “E1.DAT”” and “DATA [E2] “E2.DAT””) and two model curves (“MODEL [E1] “Model1”” and “MODEL [E2] “Model1””).

For the third graph set the X-range to [0; 1000] and set the Y-range to [0; 60000]. Change the Y-title to “Sigma1 (S/cm)”. Plot the curves “DATA [S1] “S1DC.DAT”” and “MODEL [S1] “Model1””.

Arrange the graphs on the screen in order to see them all at the same time.

Now you can try to fit the data ‘by hand’ in order to get a good starting point for the automated fitting routine. You will need more than one oscillator. I advise to use the slider in the “Parameter control” window (Figure 3-4) to change the values of the model parameter. Adjust the parameter minimum and maximum limits, if necessary (the maximum limit is only 10000 by default, which is not enough in some cases).

It may take a while before you find a reasonable fit, depending on your experience. For instance, it took me about 10 minutes to get a fit with one Drude and one Lorentz oscillator, shown in Figure 3-22. The match is not great, but it looks like an acceptable initial approximation. If you are not so happy with what you obtained, just copy my parameter values (you can load the model stored in the file “MODEL1.RFM”).

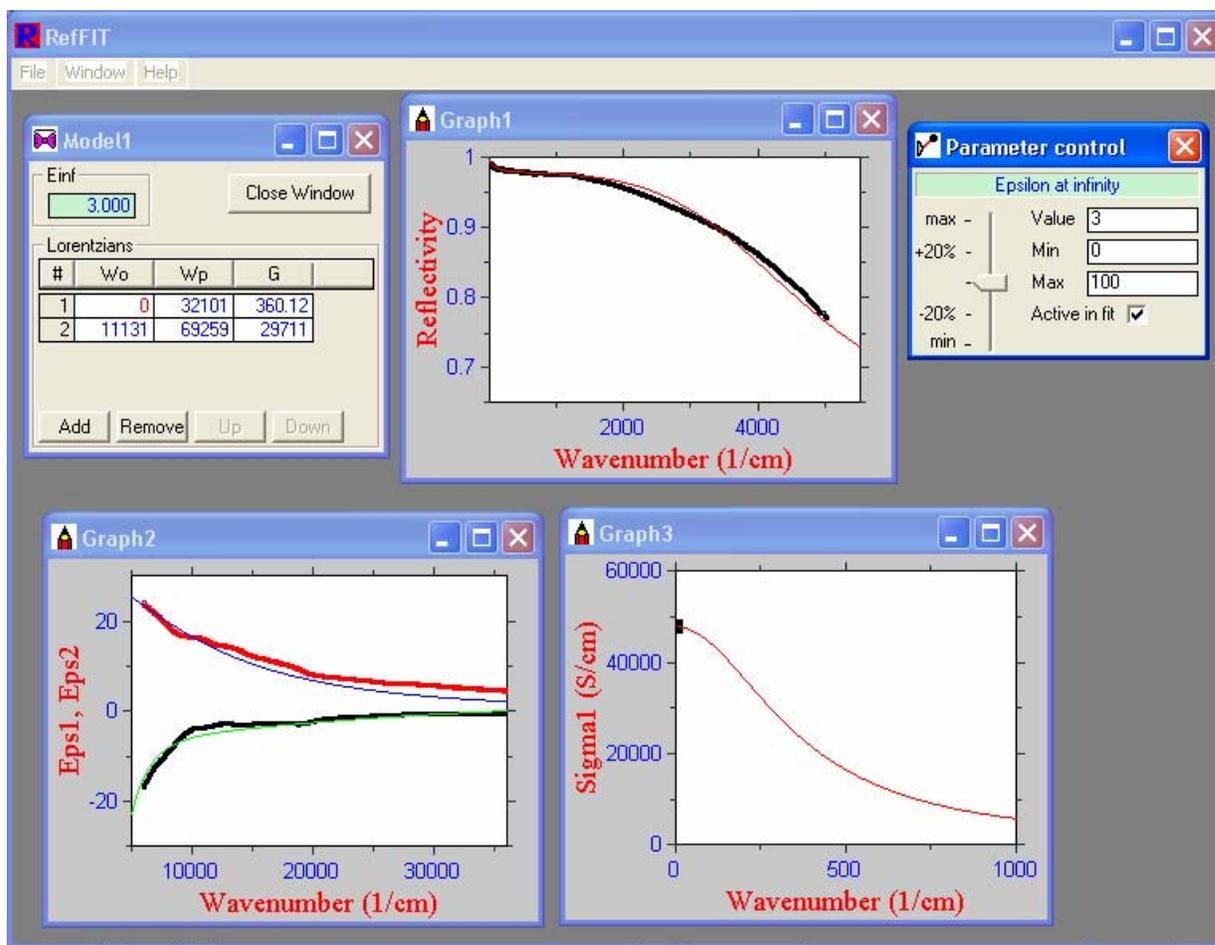


Figure 3-22 The general RefFIT view after the data fitting ‘by hand’.

OK, it is time for computer to work a little bit and refine the model parameters. We only have to set up the fitting task to tell him what exactly we want to fit.

Open the “Fit” window (Figure 3-12). We have to specify the chi-square terms  $\chi_v^2$  and their weights  $w_v$  from Equation 2-12. Click the **Add** button. In the window “Add Chi-square term(s)” (Figure 3-23) select all available terms: “[S1] “S1DC.DAT” – “Model1””, “[E2] “E2.DAT” – “Model1””, “[E1] “E1.DAT” – “Model1”” and “[R] “R.DAT” – “Model1”” and click **OK**.

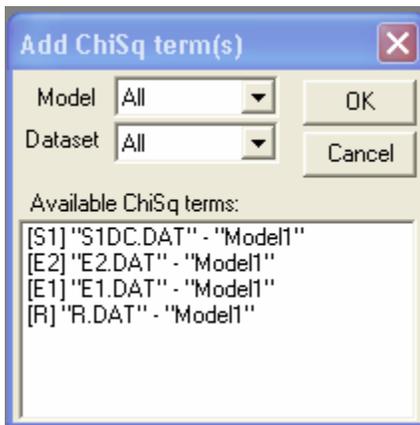


Figure 3-23 Adding the four available chi-square terms.

The weights of all chi-square terms  $w_v$  are equal to one by default. However, here we have a case, when this is not the optimal choice. The reason is (see sections 2.1.2 and 2.1.3) that we did not set the proper data error bars. RefFIT always sets to them 0.01 unless other values are explicitly specified. I would suggest you to set  $w_v$  equal to 1000 for the term “R”, 1 for the terms “E1” and “E2” and 1.0E-6 for the term “S1” (I found these values after several tries and errors). The window “Fit” should look like at Figure 3-24.

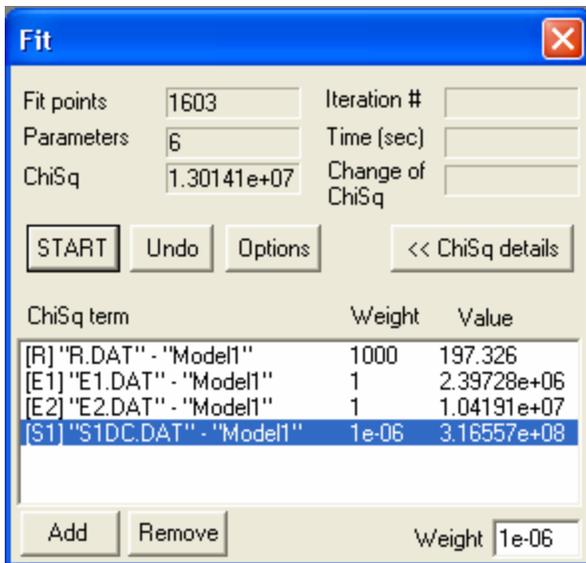


Figure 3-24 The “Fit” window. Four chi-square terms are selected.

Now you can click **START** (which is my ‘favorite’ button, by the way). It takes seconds (or even fractions of a second) to reach a better fit that you can see at Figure 3-25. Not bad for a model with only two oscillators! By the way, do not forget to save the model.

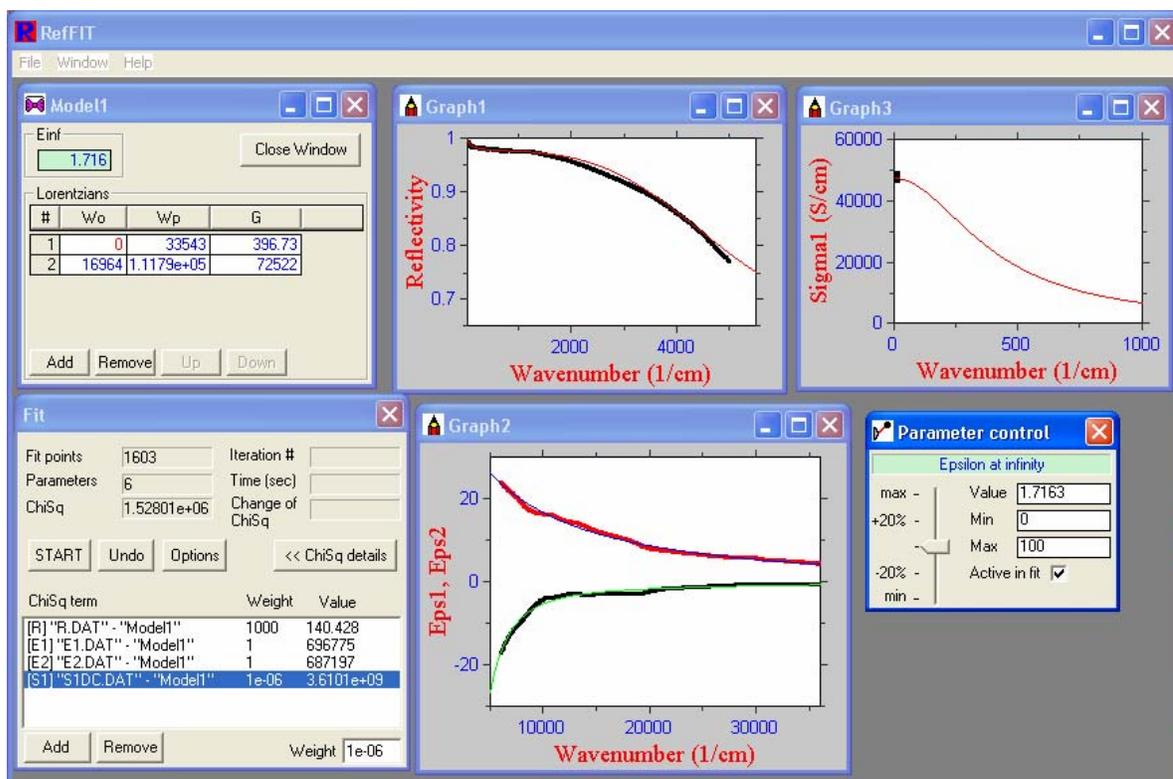


Figure 3-25 The general RefFIT view after the automated data fitting with a two-oscillator (Drude+Lorentz) model.

You may want to improve the fit by putting more oscillators. Having done it, you can compare your result with the one shown at Figure 3-26. Here 7 oscillators in the model make the match almost perfect everywhere, except for the reflectivity at about 5000  $\text{cm}^{-1}$ . I guess the problem lies in some experimental mismatch between the measured low-frequency reflectivity and the reflectivity, that corresponds to the dielectric function at high frequencies. The model, shown in Figure 3-26, can be loaded from the file "MODEL2.RFM".

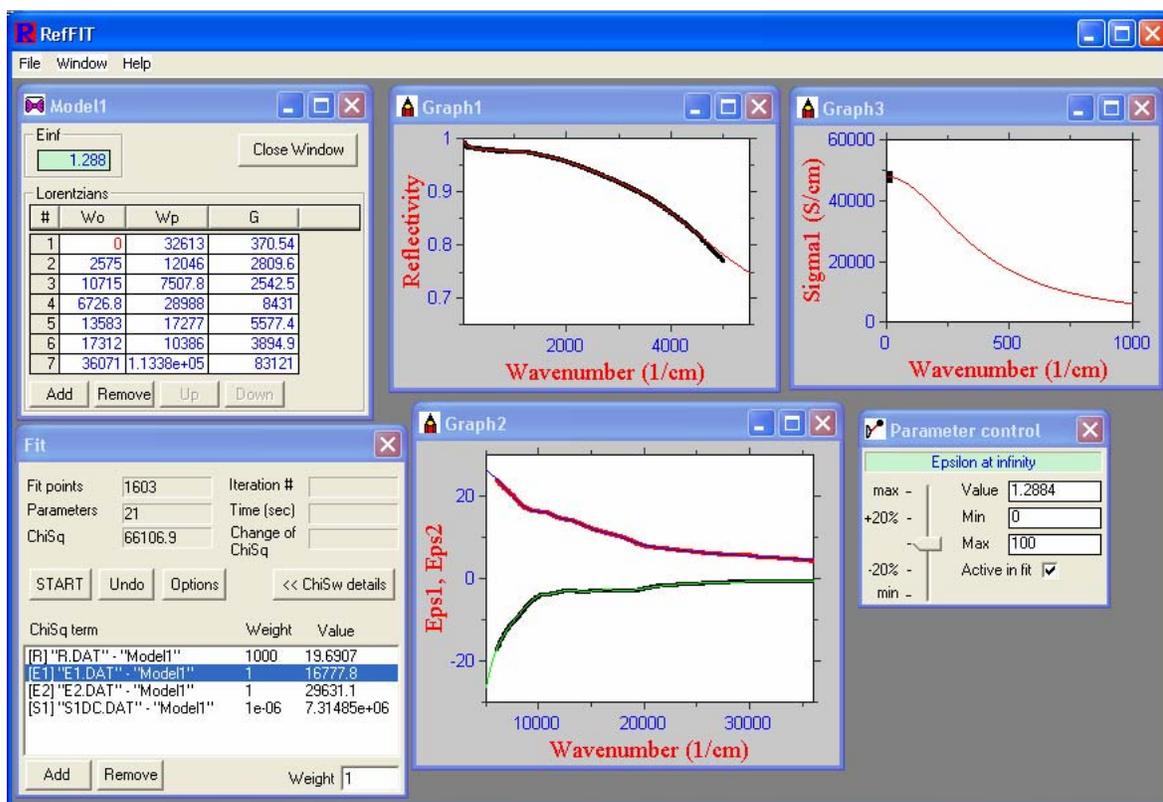


Figure 3-26 The general RefFIT view after the data fitting with a model, which contains 7 oscillators.

### 3.3. Using macros

#### 3.3.1. Writing a simple macro

All actions that we went through in section 3.2 can be programmed in a macro (see section 4.11). An obvious advantage of a macro is that it can be edited and run many times. A stored macro also helps to recollect what exactly analysis has been performed the last time.

An example of such a macro is given by the file “DEMO.RFS” in the directory “TUTORIALPART3” (I advise to put the extension “.RFS” to all macro filenames). Below I put the listing of this file. Each command corresponds to some user action. They are separated by semicolons “;”. Note, that all lines starting from “#” are ignored by the macro interpreter. You can check the syntax of particular commands in the Reference (section 4.11).

```
#####
#
#   This macro instructs RefFIT to perform the actions described
#   in the RefFIT Manual (Tutorial section 3.2)
#   All input data are taken from the directory "..\PART2"
#
#   Written by A.Kuzmenko (2004)
#
#####
#-----
# sets the dimensions and the title of the main (application) window
```

```

MainWindow(XPos = 0, YPos = 0, Width = 1024, Height = 740, Title = "macro example");

#-----
# loads four datasets

# here the frequency set is generated by RefFIT (XMethod = 3)
LoadDataset(DatasetNo = 1, Quantity = "R", MasterFile = "..\PART2\R.DAT", XMethod =
3,
    Xmin = 25.0, Xmax = 5000.0, XPts = 1000, XGrid = 2);

# here the frequency sets are taken from the master file (XMethod = 1)
LoadDataset(DatasetNo = 2, Quantity = "E1", MasterFile = "..\PART2\E1.DAT", XMethod =
1);
LoadDataset(DatasetNo = 3, Quantity = "E2", MasterFile = "..\PART2\E2.DAT", XMethod =
1);
LoadDataset(DatasetNo = 4, Quantity = "S1", MasterFile = "..\PART2\S1DC.DAT", XMethod
= 1);

#-----

# creates a model (which is AUTOMATICALLY ascribed ModelNo = 1)
NewModel(XPos = 0, YPos = 50, Width = 300, Height = 400);

# loads a saved model file into the model window
LoadModel(ModelNo = 1, File = "..\PART2\MODEL2.RFM");

#-----

# makes and sets up three graph windows

# creates a graph (which is AUTOMATICALLY ascribed GraphNo = 1)
NewGraph(XPos = 300, YPos = 50, Width = 420, Height = 350);

# sets the graph properties
GraphProperties(GraphNo = 1, Xmin = 10, Xmax = 5500.0, Xlog = 0,
    Ymin = 0.65, Ymax = 1.0, Ylog = 0, YTitle = "Reflectivity");

# plots the dataset #1
AddDataCurve(GraphNo = 1, DatasetNo = 1, nColor = 0, ShowLine = 1, ShowSymbol = 1,
    ShowError = 0, nSymbolSize = 1, nSymbolShape = 0, SymbolOpen = 1, Scale = 1.0,
Shift = 0.0);

# plots a reflectivity curve, generated by the model #1
AddModelCurve(GraphNo = 1, ModelNo = 1, Quantity = "R", nColor = 0, Scale = 1.0,
Shift = 0.0);

# creates a graph (which is ascribed GraphNo = 2)
NewGraph(XPos = 300, YPos = 400, Width = 420, Height = 330);

# and so forth...
GraphProperties(GraphNo = 2, Xmin = 5000.0, Xmax = 36000.0, Xlog = 0,
    Ymin = -30.0, Ymax = 30.0, Ylog = 0, YTitle = "Eps1, Eps2");

AddDataCurve(GraphNo = 2, DatasetNo = 2, nColor = 0, ShowLine = 1, ShowSymbol = 1,
    ShowError = 0, nSymbolSize = 1, nSymbolShape = 0, SymbolOpen = 1, Scale = 1.0,
Shift = 0.0);

AddDataCurve(GraphNo = 2, DatasetNo = 3, nColor = 0, ShowLine = 1, ShowSymbol = 1,
    ShowError = 0, nSymbolSize = 1, nSymbolShape = 0, SymbolOpen = 1, Scale = 1.0,
Shift = 0.0);

```

```

AddModelCurve(GraphNo = 2, ModelNo = 1, Quantity = "E1", nColor = 0, Scale = 1.0,
Shift = 0.0);
AddModelCurve(GraphNo = 2, ModelNo = 1, Quantity = "E2", nColor = 0, Scale = 1.0,
Shift = 0.0);

NewGraph(XPos = 650, YPos = 200, Width = 352, Height = 330);
GraphProperties(GraphNo = 3, Xmin = 0, Xmax = 1000.0, Xlog = 0,
Ymin = 0.0, Ymax = 60000, Ylog = 0, YTitle = "Sigma1 (S/cm)");
AddDataCurve(GraphNo = 3, DatasetNo = 4, nColor = 0, ShowLine = 0, ShowSymbol = 1,
ShowError = 0, nSymbolSize = 4, nSymbolShape = 0, SymbolOpen = 0, Scale = 1.0,
Shift = 0.0);
AddModelCurve(GraphNo = 3, ModelNo = 1, Quantity = "S1", nColor = 0, Scale = 1.0,
Shift = 0.0);

#-----

# opens the "Fit" window
WindowFit(Xpos = 0, YPos = 450);

# adds four chi-square terms to the fitting task, which means that all datasets are
#requested to be fitted, but does not perform fitting yet
AddChiSqTerm(DatasetNo = 1, ModelNo = 1, Weight = 1e3);
AddChiSqTerm(DatasetNo = 2, ModelNo = 1, Weight = 1.0);
AddChiSqTerm(DatasetNo = 3, ModelNo = 1, Weight = 1.0);
AddChiSqTerm(DatasetNo = 4, ModelNo = 1, Weight = 1e-6);

#-----

# starts the fitting, which is limited by 10 iterations
Fit(NumIters = 10);

#-----

```

Listing 1 The commands of the macro “DEMO.RFS” simply imitate the user actions.

How to run this macro? Technically speaking, you should provide the macro filename as a command-line parameter when you start RefFIT. As is written in section 4.11.1, there are several ways to do that. I would advise to associate the extension “.RFS” with the executable file “REFFIT.EXE”, which you can do, for instance, in the Windows Explorer program. After you have successfully done it, all macro files (including this file “DEMO.RFS”) should appear with an icon . Now double-click on the file “DEMO.RFS” and the macro will be executed.

You may want to play a bit with some commands in order to better arrange the windows on the screen, or to change the graph parameters (scales, curve appearances *etc.*). Attention: if RefFIT will find an error in the macro, it will refuse to execute it. If this happens, you can ‘locate’ the error by ‘commenting out’ the recently modified commands one-by-one (using the symbol “#”).

### 3.3.2. Using cycles in a macro: the temperature dependence of phonon spectra

Now we consider an example of a more advanced macro, which routinely fits reflectivity spectra, measured at several temperatures, with a Drude-Lorentz model.

Here we take the reflectivity spectra of the cupric oxide CuO in the far-infrared (phonon) range for the electric field parallel to the **b**-axis (Ref. [10]). This example is very interesting, because the temperature dependence of the lineshape of one phonon mode demonstrates a

remarkable ‘anomaly’, related to the antiferromagnetic ordering at about 210-230 Kelvin. The spectra were measured at temperatures 4, 100, 150, 180, 200, 210, 220, 230, 240, 250 and 300 Kelvin. They are stored as separate data files “RB003.DAT”, “RB100.DAT”, *etc.* in the directory “TUTORIAL\PART4”. What we would like to do, is to fit all spectra one-by-one with a similar model (*i.e.* with the same set of oscillators) allowing only parameter values to be varied. After that we shall plot the temperature dependence of the phonon parameters

Before the writing of a macro we have to manually find a proper model for the dielectric function. Let us first have a look at three files “RB004.DAT”, “RB150.DAT” and “RB300.DAT”. You can load these files using the Dataset Manager and plot them on the same graph (see figure Figure 3-27).

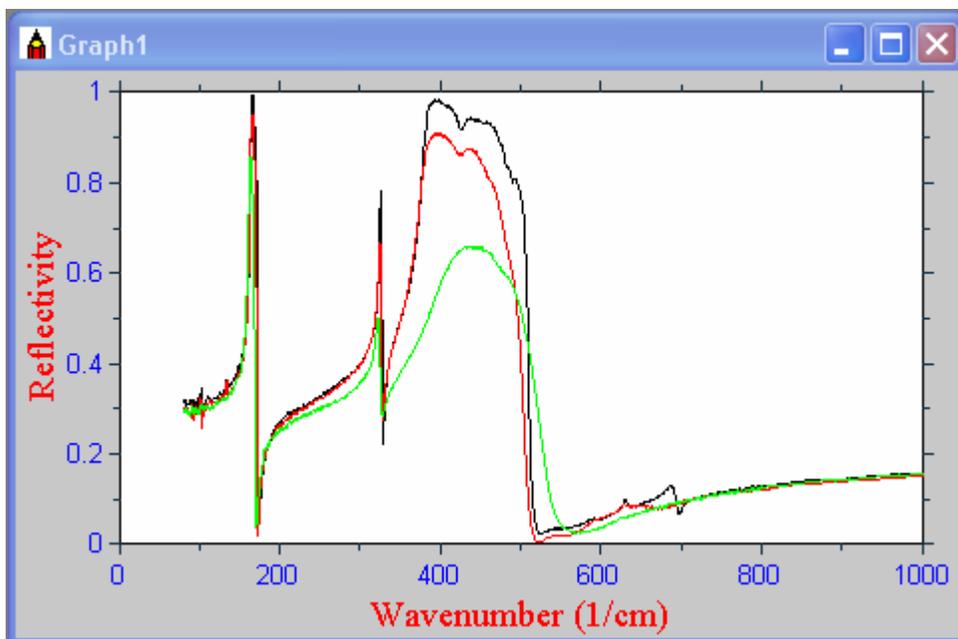


Figure 3-27 Three reflectivity spectra of CuO at 4, 150 and 300 K are plotted on the same graph.

One can see that, although the spectra look rather different, they have at least three common phonon modes (two modes at around 160, 330 and a very intense and a strongly temperature-dependent one at about  $400\text{ cm}^{-1}$ ). Few extra minor modes are present at low temperatures that we can ignore in our analysis.

I would ask you to fit the reflectivity spectrum at 4 K with a Drude-Lorentz model. You will need to put three Lorentz oscillators with all parameters free. Follow section 3.1 if you need more details on how to do that. Having obtained the fit, you can save your model (let say, as a file “MYMODEL.DAT”) and compare it with my version stored in the file “MODEL.DAT” (see Figure 3-28).

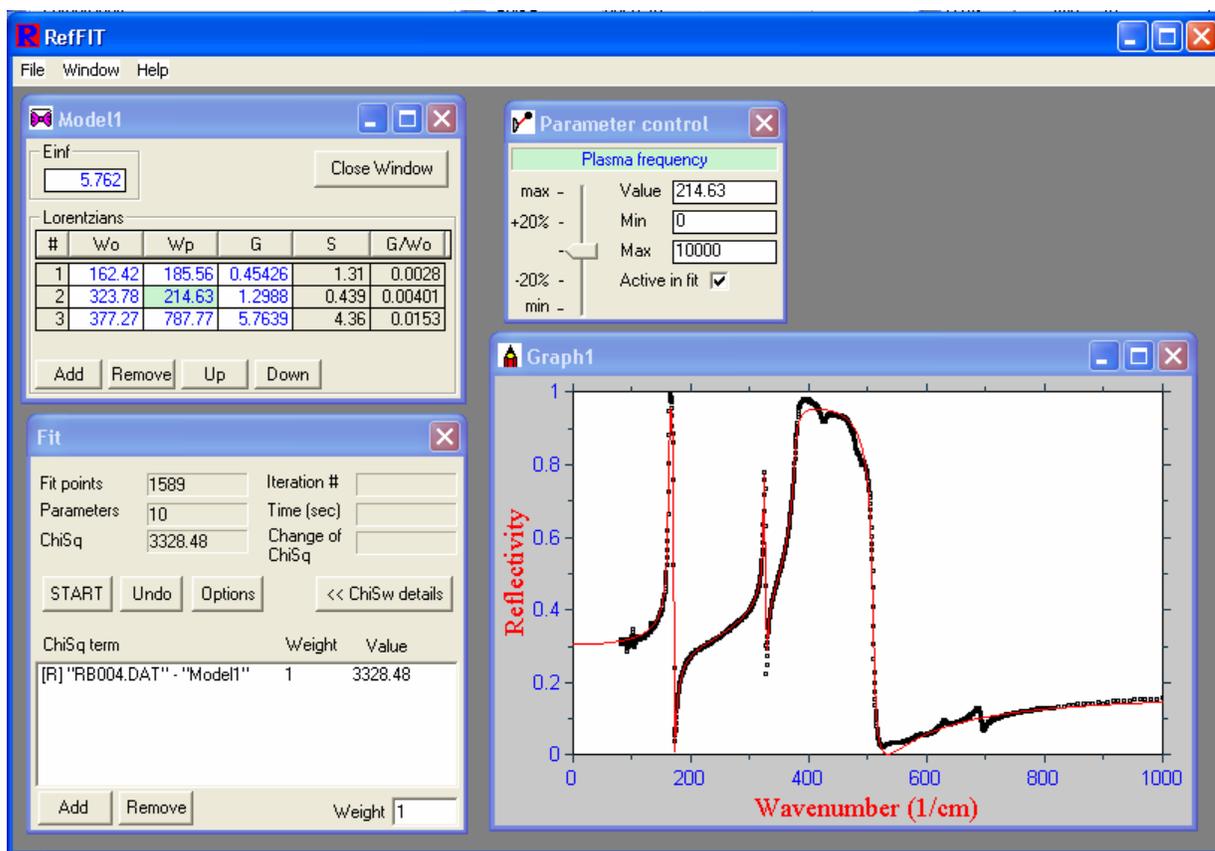


Figure 3-28 The fitting of the reflectivity spectrum at 4 K with a three-oscillator model.

Now we can proceed with a programming. A macro should first fit the spectrum at the lowest temperature (4 K); then it should fit the next spectrum (100 K), using the 4K parameter values as an initial approximation; then fit the 150 K spectrum, and so forth. On each step it must save the model parameters and export some model curves (for instance, the reflectivity or the optical conductivity) that can be plotted afterwards using any scientific graphics package (e.g., Origin).

To run a cycle we will need the text file “LOOP.DAT”, each line of which corresponds to one particular temperature (see Listing 2). More exactly, each line contains a particular version of the changeable (temperature-dependent) part of the data file name. I have already made this file for you, but you can also create it with any standard text editor.

```

004
100
150
180
200
210
220
230
240
250
300

```

Listing 2 File “LOOP.DAT” specifies the macro cycle.

The macro itself, which is called “FITALL.RFS” is shown in the Listing 3.

```

#####
#
#   This macro fits the reflectivity spectra
#   "RB004.DAT", "RB100.DAT", ... , "RB300.DAT",
#   which are measured for different temperatures, in a cycle.
#   All input data are taken from the same directory
#
#   Written by A.Kuzmenko (2004)
#
#####

#-----
# opens the main window

MainWindow(XPos = 0, YPos = 0, Width = 1024, Height = 740);

#-----
# creates a new model

NewModel(XPos = 0, YPos = 50, Width = 300, Height = 240);

#-----
# creates and sets up a couple of new graphs (for reflectivity and conductivity

NewGraph(XPos = 300, YPos = 50, Width = 720, Height = 350);
GraphProperties(GraphNo = 1, Xmin = 75.0, Xmax = 4000.0, Xlog = 1, Ymin = 0.0,
               Ymax = 1.0, Ylog = 0, YTitle = "Reflectivity");
AddModelCurve(GraphNo = 1, ModelNo = 1, Quantity = "R");

NewGraph(XPos = 300, YPos = 400, Width = 720, Height = 330);
GraphProperties(GraphNo = 2, Xmin = 75.0, Xmax = 4000.0, Xlog = 1, Ymin = 0.0,
               Ymax = 100.0, Ylog = 0, YTitle = "Conductivity");
AddModelCurve(GraphNo = 2, ModelNo = 1, Quantity = "S1");

#-----
# opens the "Fit" window

WindowFit(Xpos = 0, YPos = 450);

#-----
# loads a 'seed' model

LoadModel(ModelNo = 1, File = "MODEL.RFM");

#-----
# enters the cycle loop

BeginLoop(LoopFile = "LOOP.DAT");

#-----
# changes the title of the application

MainWindow(Title = "Fitting spectrum at T = %1 K");

#-----
# loads a new experimental reflectivity curve

LoadDataset(DatasetNo = 1, Quantity = "R", MasterFile = "RB%1.DAT",
            XMethod = 3, Xmin = 75.0, Xmax = 2000.0, XPts = 1000, XGrid = 2);

#-----
# plots the experimental reflectivity curve

```

```

AddDataCurve(GraphNo = 1, DatasetNo = 1, ShowLine = 1, ShowSymbol = 1,
             ShowError = 0, nSymbolSize = 1, nSymbolShape = 0, SymbolOpen = 1);

#-----
# adds the corresponding chi-square term

AddChiSqTerm(DatasetNo = 1, ModelNo = 1, Weight = 1.0);

#-----
# wait until all the graphs are redrawn

Wait();

#-----
# performs the fitting (maximum 10 iterations)

Fit(NumIters = 10);

#-----
# suspends the execution, allows the user to verify, that the fit is OK

Suspend();

#-----
# saves the new model parameters under a unique name

SaveModel(ModelNo = 1, File = "MOD_%1.RFM");

#-----
# export the model reflectivity and conductivity curves

ExportModelCurve(File = "RF_%1.DAT", ModelNo = 1, Quantity = "R",
                 Xmin = 1E-4, Xmax = 1E6, XPts = 1000, XGrid = 2);

ExportModelCurve(File = "S1_%1.DAT", ModelNo = 1, Quantity = "S1",
                 Xmin = 0.995e-4, Xmax = 1.005e-4, XPts = 3, XGrid = 1);

#-----
# deletes the experimental reflectivity curve

DeleteDataCurve(GraphNo = 1, DatasetNo = 1);

#-----
#deletes the chi-square term

DeleteChiSqTerm(DatasetNo = 1, ModelNo = 1);

#-----
#unloads the dataset

UnloadDataset(DatasetNo = 1);

#-----
# the end of the cycle loop
EndLoop();

#-----
# deletes the model curve

DeleteModelCurve(GraphNo = 1, ModelNo = 1, Quantity = "R");
DeleteModelCurve(GraphNo = 2, ModelNo = 1, Quantity = "S1");

#-----

```

```

# restores the original application title

MainWindow(Title = "");

#-----
# exits the application

Exit(QuitApp = 1);
#-----

```

Listing 3 Macro “FITALL.RFS” fits the reflectivity spectra at different temperatures one-by-one in a cycle.

The cycle body is marked by the two commands: `BeginLoop` and `EndLoop`. In the first command the name of the ‘loop file’ has to be specified (“LOOP.DAT”). The cycle will be repeated as many times as the number of lines in the loop file. On each stage the symbol “%1” in the file name parameter of all input-output commands (`LoadDataset`, `SaveModel`, `ExportModelCurve`) is substituted with the string in the corresponding line of the loop file (for instance, “RB%1.DAT” will be substituted by “RB004.DAT” on the first stage, by “RB100.DAT” on the second stage and so on).

It is essential, that within the cycle body the commands `LoadDataset`, `CreateDataCurve` and `AddChiSquareTerm` are followed by the corresponding ‘anti-commands’ `UnloadDataset`, `DeleteDataCurve` and `DeleteChiSquareTerm` correspondingly. Otherwise the execution of a macro may cause the program crash (sorry, that RefFIT is not ‘fool-proof’ yet!).

Now you can run the macro, as described in section 3.3.

In the beginning it will fit the spectrum at 4 K and show a dialog window asking whether you want to continue the macro execution (see Figure 4-30). This dialog is invoked by the command `Suspend` that we put after the command `Fit`. If you are not happy with the fitting quality, you can stop the macro and ‘correct’ the model manually. It is also possible to correct the model without the terminating the macro and to continue the execution later on. If you find the fitting match acceptable, then continue the program without doing anything.

In this version the macro will ask you to verify the fitting quality for every temperature. If you think it is unnecessary, then just remove or ‘comment out’ the command `Suspend`, and run the macro again. It is much faster now, isn’t it!!!!

You may have noticed already that the macro execution creates a number of new files. These are: (i) the models saved in the files “MOD\_XXX.RFM” (RefFIT format) and “MOD\_XXX.ASC” (text format), (ii) the model reflectivity curves (files “RF\_XXX.DAT”), (iii) the model optical conductivity curves (files “S1\_XXX.DAT”), (iv) the file “PAR.DAT”, where RefFIT ‘flushes’ the model parameters to every time it saves a model (see section 4.5.1).

You can use the file “PAR.DAT” to plot the temperature dependence of the parameters of the phonon modes. Let us make a graph of  $\omega_o(T)$  and  $\gamma(T)$  of the model third oscillator, which seems to change strongly when we sweep across the critical temperature. One way to do it is to import the file “PAR.DAT” to the Origin worksheet (see Figure 3-29). Then we should somehow convert the first column (which is a filename text) to the temperature (which is a

number). Here you can do it just ‘by hand’, but when you have more temperatures, it makes sense to do the conversion differently<sup>14</sup>.

	A[M]	B[M]	C[M]	D[M]	E[M]	F[M]	G[M]	H[M]	I[M]	J[M]	K[M]
1	MOD 004.RFM	1609.7	5.83499	162.44678	186.16435	0.46467	323.85505	213.26503	1.32236	377.74129	791
2	MOD 100.RFM	1850.3	5.78914	162.32155	185.99903	0.51717	323.65976	205.69588	1.67747	377.69777	786
3	MOD 150.RFM	1386.4	5.51997	162.1938	178.47565	0.65414	323.03407	208.31505	2.27591	374.39254	75
4	MOD 180.RFM	1723	5.71522	161.66311	183.22476	0.60621	322.54074	215.41295	2.47389	371.23377	763
5	MOD 200.RFM	2477.7	5.94318	161.13001	192.08962	0.40516	322.17239	216.61251	2.38965	369.57557	799
6	MOD 210.RFM	2530.1	5.94207	161.14432	190.70468	0.45529	322.09172	201.59845	2.32802	373.40588	820
7	MOD 220.RFM	1296.8	5.88907	160.95698	189.2166	0.54555	321.97144	194.04946	2.65966	382.55563	838
8	MOD 230.RFM	1173.4	5.97981	160.92678	185.61479	0.79155	322.00261	189.91199	3.0885	394.22091	843
9	MOD 240.RFM	1219.4	5.95939	160.95814	180.1451	0.9905	322.05775	180.17829	3.31204	400.65612	832
10	MOD 250.RFM	888.73	5.95169	160.77954	178.8755	1.04186	322.13687	169.54357	3.32133	404.59513	830
11	MOD 300.RFM	809.63	5.76879	160.49033	168.65334	1.47299	321.65576	158.50887	3.94541	409.78589	807

Figure 3-29 File “PAR.DAT”, which contains the temperature dependence of the phonon parameter, can be imported into Origin.

The desired parameters are stored in the 10-th and 12-th columns respectively (the meaning of columns is described in section 4.5.1). You can plot the graphs, as is shown in Figure 3-30. One can see that the phonon mode softens and narrows dramatically below the transition temperature (210-230 K). This is perhaps the strongest phonon anomaly I ever seen!

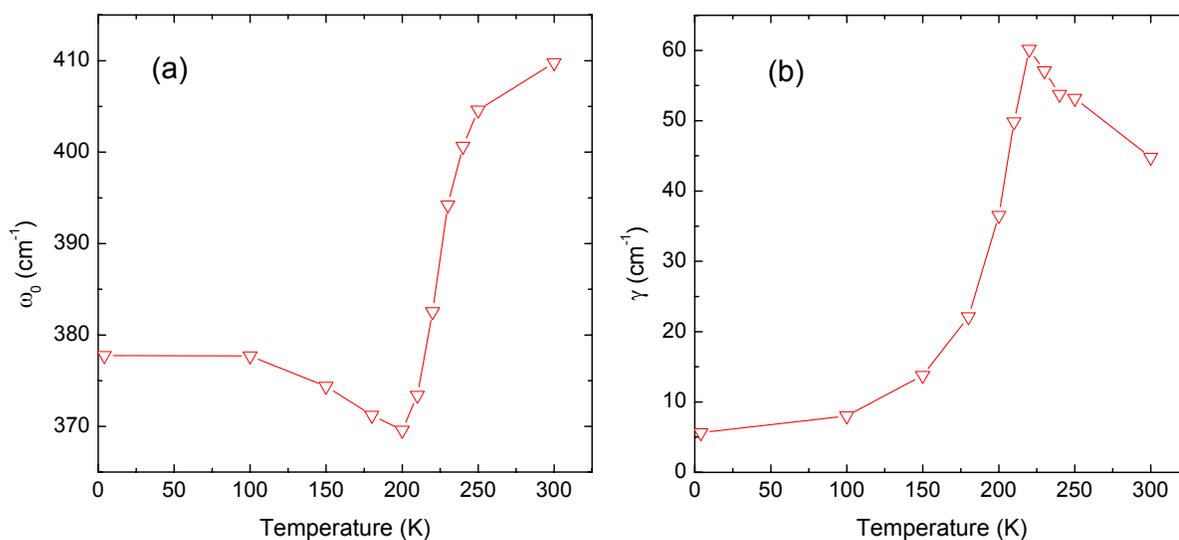


Figure 3-30 The temperature dependence of a phonon mode, stored in the file “PAR.DAT” after the macro execution.

<sup>14</sup> I know one ‘trick’: to import the file “LOOP.DAT” to the first column of this worksheet.

## 3.4. Using variational dielectric functions

### 3.4.1. Kramers-Kronig analysis of reflectivity

The Drude-Lorentz fitting of reflectivity spectrum with a limited number of oscillators (see section 3.1) gives artificially smooth resulting spectra of  $\varepsilon_1(\omega)$  and  $\varepsilon_2(\omega)$  (see, for example, Figure 3-19). Some small, but potentially important spectral details are always lost in this approach. On another hand, the well-known Kramers-Kronig analysis of reflectivity [6], which is, in principle, model-independent, preserves all spectral information, contained in the original reflectivity curve.

With RefFIT, the KK analysis of spectra can also be done. However, the approach, used in this program, is rather different from the conventional one [6]. Instead of applying the KK transformation to the reflectivity directly, it can *fit* the reflectivity with a KK-constrained variational dielectric function, abbreviated ‘VDF’ (see section 2.2.4). In Ref. [4] it was shown, that both approaches give almost identical results, when applied to a normal-incidence reflectivity.

We are going to apply the KK analysis to the same reflectivity spectrum as we fitted in section 3.1. We shall also get use of the best Drude-Lorentz reflectivity fit, obtained in that part of the Tutorial.

```
...
>>> To be finished <<<
...
```

All the operations, that we just performed manually, I also put to the macro “KK.rfs”, located in the directory “TUTORIAL\PART5”. A special command `VarDielFunc` (see section 4.11.5) was used to manipulate VDFs.

### 3.4.2. Inversion of ellipsometric data

In this example we are going to use VDFs without the Kramers-Kronig constraint.

```
...
>>> To be finished <<<
```

---

## 4. Reference

### 4.1. System requirements

ReFFIT can be run under all Microsoft Windows versions starting from Windows 95. Although it is difficult to formulate specific computer requirements, one can advise to have at least 128 Mb of memory and processor faster than 1 GHz. The point is that some complicated fitting tasks claim a lot of memory and are computationally rather challenging, especially if the variational dielectric functions are involved (see sections 2.2.4, 2.2.5, 4.6.3). Also, a high processor speed is necessary to update the screen in real time when a user is manually ‘playing with’ model parameters.

### 4.2. Installation and starting RefFIT

Although there is no setup program, the installation of RefFIT is very easy. One should manually create a directory, *e.g.* “D:\REFFIT”, and copy to it all the files from the distribution package (including “REFFIT.EXE”), keeping the directory structure. No extra files, such as dynamical libraries, have to be installed on the system.

To start RefFIT, one should just run file “REFFIT.EXE”. It can be done, for instance, by double-clicking on the program icon in the Explorer program. To facilitate the program call one can create a shortcut to “REFFIT.EXE” on the Windows taskbar. When using a macro, “REFFIT.EXE” should be called with a macro name as a command line parameter. More details about macros can be found in section 4.11.

Right after starting, the application window looks rather ascetic (Figure 4-1). All actions are done using the menus or the toolbar. The menu “File” is intended to Save and Load the RefFIT projects<sup>15</sup>. Via the “Window” menu all RefFIT windows (models, graphs *etc.*) are created. Finally, from the “Help” menu one can access the “RefFIT manual” (this file basically)<sup>16</sup> and get the “About” information.

As usual, one can choose “Exit” in the “File” menu or click the  button in order to terminate the application.

---

<sup>15</sup> This option is not realized yet! However, one can ‘save’ the project by writing it in the form of a macro.

<sup>16</sup> It is supposed that your system can read the pdf-files (Adobe Reader or Adobe Acrobat must be already installed).

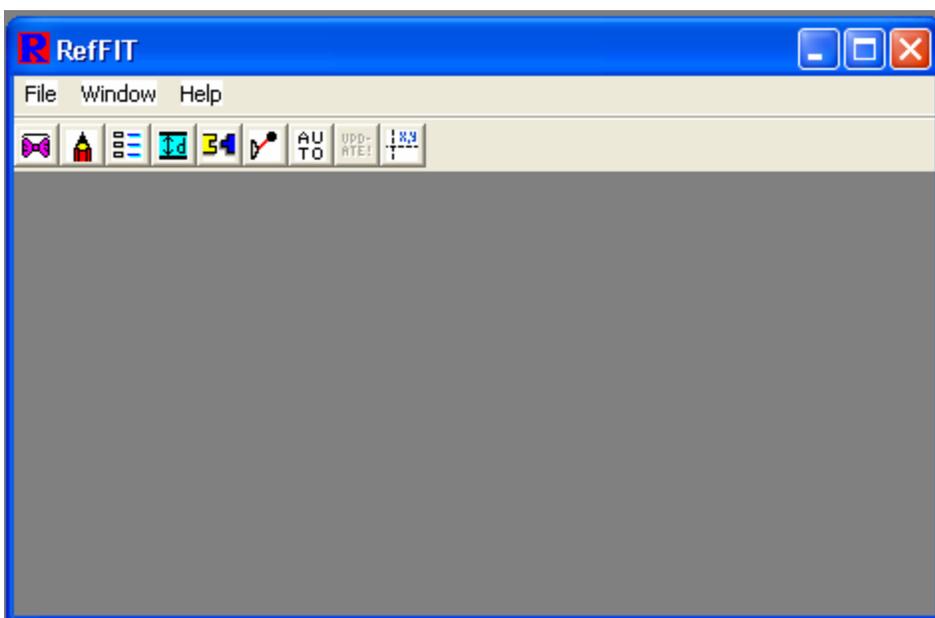


Figure 4-1 This is how RefFIT looks like when it starts. No internal objects are created yet.

### 4.3. Parameters

A parameter in RefFIT is a number, which is used as an input for model formulas to calculate optical quantities, and can be varied (manually or automatically) in order to improve the match of the model and experimental spectra. The examples of parameters are Lorentzian transverse frequency, and experimental angle of incidence. Parameter values are displayed and can be manually edited either in the “Experimental parameters” window (Figure 4-3) or “Model” windows (Figure 4-4).

The value is not the only characteristic of a parameter. Another essential property is the switch, which determines whether or not a parameter is allowed to be varied in the automated fitting (adjustable) or not. There are also maximal and minimal values (limits) that parameters may have<sup>17</sup>.

To access these extra characteristics a separate window is provided, which is called “Parameter control” (Figure 4-2). It contains the full information about the parameter, which is currently selected. To see the characteristics of another parameter, one should select it by a mouse click or by moving the marker to it with the arrow keys. This window also contains a trackbar, which allows you to change the parameter value in a comfortable way by dragging it up and down. Note that all the other windows in RefFIT will be updated in real time.

<sup>17</sup> Although the limits can be set, the fitting routine is not yet able to take them into account

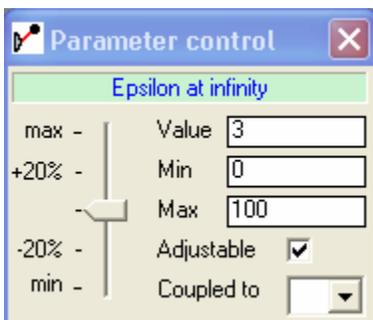


Figure 4-2 The “Parameter control” windows contains the full information about the selected parameter.

The activity switch can be also toggled in the “Model” window by clicking right mouse button on the parameter, or pressing **SPACE** when the parameter is selected.

There is a possibility to ‘couple’ two or more parameters together. The coupled parameters are treated as a single one. In particular they always have the same values and other characteristics. The parameters may belong to different models. In order to couple parameters together, one has to associate each of them with the same ‘coupling variable’ using the “Parameter Control” window. There can be several coupling variables, allowing several sets of coupled parameters. One has to do it, for example, if two ellipsometry experiments are done on the same sample. In this case each experiment should be described by a separate model, but the sample thickness has to be the same. In the model window, the coupled parameters are displayed with a bold font.

#### 4.4. Experimental Parameters

There is a set of parameters, which are not directly related to the model dielectric functions, but rather to the experimental conditions. For instance, the reflectivity coefficient depends on the angle of incidence, even though the dielectric function does not. This kind of parameters is collected in a separate window called “Experimental parameters” (Figure 4-3). It can be displayed by choosing the corresponding item in the “Window” menu.

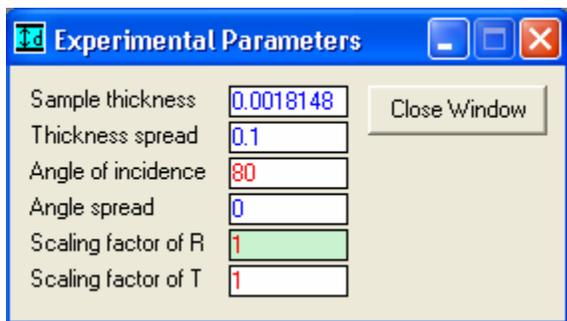


Figure 4-3 The “Experimental parameters” window lists all parameters, related to the experimental conditions.

One should note that experimental parameters apply to the output quantities of the models of the Dielectric Function type only (see section 4.6). In the case of other model types all the parameters, necessary to calculate model outputs, are contained in the model window.

The description of all the experimental parameters is given in Table 4-1.

Parameter name	Units	Description
----------------	-------	-------------

Sample thickness	cm	Thickness of the sample, used to calculate the coefficient of transmission with the interference (“T”) and without the interference “TT”.
Thickness spread	-	The relative spread of the sample thickness. Used to model the effect of non-parallel (wedged) sample. Applies to the quantities “T” and “TT”.
Angle of incidence	degrees	The angle of incidence. The normal incidence corresponds to 0°, the totally grazing incidence – 90°. Applies only to “Rp”, “Rs” and “Rps”.
Angle spread	degrees	The absolute spread of angles of incidence. Used to model the effect of non-parallel beam. Applies to “Rp”, “Rs” and “Rps”.
Scaling factor of R	-	Used to scale the calculated value of reflectivity by a constant value. Has no particular physical meaning, but may help to simulate the systematic error bars. Applies only to ‘R’, ‘Rp’ and ‘Rs’.
Scaling factor of T	-	The same as, scaling factor of R, but applied to the transmission coefficient (“T” and “TT”).

Table 4-1 The description of the experimental parameters.

## 4.5. Models

Models are the central objects in the RefFIT program. One can define a model as a set of parameters, which are used to calculate a set of optical quantities, for example reflectivity, conductivity, loss function *etc.*, according to certain sets of formulas. Each model is represented by a separate window, which looks like Figure 4-4.

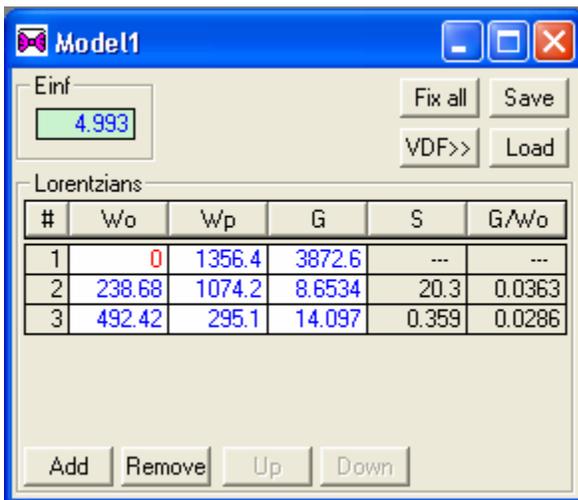


Figure 4-4 An example of the “Dielectric function” model window.

In order to create a new model one should choose the “Model” command in the “Window” menu, or click on the  button from the toolbar. One can open as many as 100 models at the same time. They will automatically receive names “Model1”, “Model2” *etc.*

To close the existing model window you can either click either  or  button. When the Model window is closed then the model itself is destroyed as an object.

The model windows can be resized.

### 4.5.1. Saving and loading models

A Model can be stored in a separate file. To do that, one should activate the corresponding window and press , or alternatively, the **F2** button. In the “Save Model as” window that appears one can specify the file path (directory) and the file name. The convention is that the model files have the extension “RFM”, *e.g.* “MYMODEL.RFM”.

The model file stores information about the parameters in the internal RefFIT format, which cannot be viewed or accessed by other programs. However, every time a model is saved, an extra file is created automatically, which lists all parameter values and error bars in the text format, that can be opened, *e.g.* by the Notepad program. This file has the same name as the model file, but the extension “ASC”. For instance, after saving a model under file name “MOD12.RFM”, a text file “MOD12.ASC” is created. Note, that error bars are not calculated, unless the corresponding fitting option is specified (section 4.10.2).

There is another ‘side’ action, which accompanies model saving. Namely, a new line is added to a file called “PAR.DAT”, which lies in the current directory. This line has the following format “ModelFileName Chi-Square Einf Wo[1] Wp[1] G[1] Wo[2] Wp[2] G[2] Wo[3] Wp[3] G[3] ...”, where ModelFileName is the filename of the model saved, Chi-Square is the current value of the fitting chi-square, Einf is the value of  $\epsilon_{\infty}$ , Wo[1], Wp[1] and G[1] are the values parameter  $\omega_0$ ,  $\omega_p$  and  $\gamma$  from the first row in the Lorentzian table, Wo[2], Wp[2] and G[2] are the parameters from the second row and so on. File “PAR.DAT” is very useful, when, for example, the spectra for different temperatures are fitted in a cycle and for each temperature the model is saved under a different name. Then one can easily plot the temperature dependence of model parameters, such as plasma frequency or conductivity gap, using any standard scientific graphics package (see an example in section 3.3.2).

A saved model can be loaded into an existing model window. One should create a new model window or activate the existing one and press , or, alternatively, the **F3** button. After that the model file to be loaded has to be selected. It is a good idea to save models after each modification (*e.g.*, fitting run) to avoid the potential loss of data.

### 4.5.2. Editing models

The rows in the ‘Lorentzian’ section can be added and deleted by the buttons  and , or, alternatively, by the **Insert** and **Delete** keys. A row can be moved up and down by clicking  and . They can also be sorted in different ways by mouse clicking on the column headers. For instance, in order to sort the Lorentzians by increasing transversal frequencies one should click on the first column header “Wo”.

### 4.5.3. Model types

The meanings of parameters and output optical spectra depend on the model type. At the moment there exist several types of models. The most common one is the “Dielectric Function” type (see section 4.6). Each newly created model is assumed to be of this type. In this case the meaning of the parameters exactly corresponds to the text labels in the Model window.

Originally, the “Dielectric Function” was the only type of models in RefFIT, but recently other types were introduced (see section 4.7). Due to technical reasons, it was easier to keep the same appearance of the Model window for other model types and use special ‘tricks’ in order to explain RefFIT that the model type is not “Dielectric Function”, but something else. According to the convention, the way to change the model type is to put a negative integer number (code) into the parameter “Einf” and make this parameter fixed (look red on the screen). The code -1 corresponds to the extra (special) model type 1, code -2 corresponds to the extra model type 2, *etc.* In this case “Einf” is not treated as a parameter but just as model type identifier and the text labels in the Model window do not have original meanings anymore. Particular meaning of parameters in the “Lorenzian” section depends on the model type<sup>18</sup>.

Each model has a set of output quantities, which depends on the model type. Different quantities are designated by abbreviations. For instance, reflectivity is referred to as “R”, the real part of conductivity – “S1”, penetration depth – “PD” *etc.* These abbreviations are meaningful only for the Dielectric Function model. For other types the meaning of the output quantities is different, although the same abbreviations are used.

The next sections describe different model types.

### 4.6. “Dielectric function” model

In the “Dielectric Function” model the complex dielectric function  $\varepsilon = \varepsilon_1 + i\varepsilon_2$  is calculated as a sum (linear superposition) of different terms, *e.g.*, Drude, Lorentz, or others, using the parameters residing in the “Model” window. The dielectric function has only one component, thus this model is applicable either to the isotropic sample or to the special experiment geometry, when only one component of the dielectric tensor is involved<sup>19</sup>.

All the output optical quantities are derived from the dielectric function. In some cases extra parameters are used from the “Experimental parameters” set. For example, the transmission of a thin sample depends on sample thickness.

Table 4-2 lists all output quantities of the “Dielectric Function” model. The formulas in this table refer to the complex refraction index  $N = n_1 + in_2 = \sqrt{\varepsilon}$ , complex reflectivity  $r = r_1 + ir_2 = \frac{1 - N}{1 + N}$  and complex optical conductivity  $\sigma = \sigma_1 + i\sigma_2 = \frac{\omega\varepsilon}{4\pi}$ .

Quantity abbreviation	Description	Calculation formula(s)	Units	Experimental parameters used
-----------------------	-------------	------------------------	-------	------------------------------

<sup>18</sup> In the future RefFIT versions, a more elegant way to distinguish model types will surely be found.

<sup>19</sup> Some special models can combine several isotropic dielectric functions to form a tensor.

“R”	Normal-incidence reflectivity of semi-infinite sample	$\alpha_R  r ^2$	-	Scaling factor of $R$ ( $\alpha_R$ )
“Rph”	Complex phase of the normal-incidence reflectivity of a semi-infinite sample	$\arg(r) \equiv \arctan(\text{Im}(r)/\text{Re}(r))$	-	-
“T”	Normal-incidence transmission of a finite-thickness sample, considering multiple (Fabry-Perot) internal reflections and a possible thickness spread	$\alpha_T \langle T \rangle_{average}$ , where $\langle T \rangle_{average} = \frac{1}{N+1} \sum_{i=0}^N T(d_i)$ , $d_i = d \left( 1 + s_d \frac{i - N/2}{N} \right)$ , ( $N = 10$ ), $T(d) = \left  \frac{(1-r^2)t(d)}{1-r^2t(d)} \right ^2$ , $t(d) = \exp\left( i \frac{\omega}{c} \sqrt{\epsilon} d \right)$	-	sample thickness ( $d$ ), thickness spread ( $s_d$ ), scaling factor of $T$ ( $\alpha_T$ )
“TT”	Normal-incidence transmission of a finite-thickness sample, ignoring multiple (Fabry-Perot) internal reflections but considering a possible thickness spread	$\alpha_T \langle T \rangle_{average}$ , where $\langle T \rangle_{average} = \frac{1}{N+1} \sum_{i=0}^N T(d_i)$ , $d_i = d \left( 1 + s_d \frac{i - N/2}{N} \right)$ , ( $N = 10$ ), $T(d) =  1 - r^2  t(d)^2$ , $t(d) = \exp\left( i \frac{\omega}{c} \sqrt{\epsilon} d \right)$	-	sample thickness ( $d$ ), thickness spread ( $s_d$ ), scaling factor of $T$ ( $\alpha_T$ )
“Rp”	Grazing-incidence reflectivity (p-polarization)	$\alpha_R \langle R_p \rangle_{average}$ , where $\langle R_p \rangle_{average} = \frac{1}{N+1} \sum_{i=0}^N R_p(\theta_i)$ , $\theta_i = \theta \left( 1 + s_\theta \frac{i - N/2}{N} \right)$ , ( $N = 10$ ), $R_p(\theta) =  r_p(\theta) ^2$ ,	-	angle of incidence ( $\theta$ ), angle spread ( $s_\theta$ ), scaling factor of $R$ ( $\alpha_R$ )

		$r_p(\theta) = \frac{\varepsilon \cos \theta - \sqrt{\varepsilon - \sin^2 \theta}}{\varepsilon \cos \theta + \sqrt{\varepsilon - \sin^2 \theta}}$		
“Rs”	Grazing-incidence reflectivity (s-polarization)	$\alpha_R \langle R_s \rangle_{average}$ , where $\langle R_s \rangle_{average} = \frac{1}{N+1} \sum_{i=0}^N R_s(\theta_i)$ , $\theta_i = \theta \left( 1 + s_\theta \frac{i - N/2}{N} \right), (N = 10),$ $R_s(\theta) =  r_s(\theta) ^2$ $r_s(\theta) = \frac{\cos \theta - \sqrt{\varepsilon - \sin^2 \theta}}{\cos \theta + \sqrt{\varepsilon - \sin^2 \theta}}$	-	angle of incidence ( $\theta$ ), angle spread ( $s_\theta$ ), scaling factor of $R$ ( $\alpha_R$ )
“Rps”	Ratio between Rp and Rs	$\langle R_{ps} \rangle_{average}$ , where $\langle R_{ps} \rangle_{average} = \frac{1}{N+1} \sum_{i=0}^N R_{ps}(\theta_i)$ , $\theta_i = \theta \left( 1 + s_\theta \frac{i - N/2}{N} \right), (N = 10),$ $R_{ps}(\theta) =  r_{ps}(\theta) ^2$ $r_{ps}(\theta) = \frac{r_p}{r_s} = \frac{\sin^2 \theta - \cos \theta \sqrt{\varepsilon - \sin^2 \theta}}{\sin^2 \theta + \cos \theta \sqrt{\varepsilon - \sin^2 \theta}}$	-	angle of incidence ( $\theta$ ), angle spread ( $s_\theta$ )
“E1”	The real part of the dielectric function	$\varepsilon_1$	-	-
“E2”	The imaginary part of the dielectric function	$\varepsilon_2$	-	-
“S1”	The real part of the optical conductivity in practical units	$\sigma_1 \times \frac{\pi}{15}$	$\Omega^{-1} \text{cm}^{-1}$	-
“S2”	The imaginary part of the optical conductivity in practical units	$\sigma_2 \times \frac{\pi}{15}$	$\Omega^{-1} \text{cm}^{-1}$	-

“N1”	The real part of the refraction index	$n_1$	-	-
“N2”	The imaginary part of the refraction index (extinction coefficient)	$n_2$	-	-
“PD”	Penetration depth	$\frac{1}{2\pi n_2 \omega}$	cm	-
“LF”	Loss function	$\text{Im}\left(-\frac{1}{\varepsilon}\right) = \frac{\varepsilon_2}{\varepsilon_1^2 + \varepsilon_2^2}$	-	-

Table 4-2 The output quantities of the “Dielectric Function” model.

### 4.6.1. Drude-Lorentz dielectric function

The physical meaning of the Drude-Lorentz (DL) dielectric function and its parameters is described in section 2.2.3.

The dimensionless  $\varepsilon_\infty$  is stored in the “Einf” field. Each row in the “Lorentzians” table corresponds to one oscillator. The transverse frequency  $\omega_0$  is taken from the column “Wo” (measured in  $\text{cm}^{-1}$ ), plasma frequency  $\omega_p$  – from column “Wp” (measured in  $\text{cm}^{-1}$ ), and linewidth  $\gamma$  (measured in  $\text{cm}^{-1}$ ) – from column “G”. The light frequency  $\omega$  is measured in  $\text{cm}^{-1}$  as well.

The Drude term is a particular case of the Lorentz term, with  $\omega_0 = 0$ . In this case  $\omega_0$  must be fixed (not active in fit).

### 4.6.2. Special dielectric functions

In addition to the standard Drude-Lorentz model RefFIT has a number of extra built-in formulas for the dielectric function. In this section we will describe all of them.

The parameters of the extra (special) dielectric functions can be stored in the “Lorentzians” table of the model window, using a ‘trick’, similar to the one intended to distinguish the Dielectric Function model from other (special) models (section 4.5).

By default, RefFIT assumes that each row in the “Lorentzians” table corresponds to a Lorentz oscillator. However, if the “Wo” column contains a *negative integer value*, e.g., -10 (it should be also fixed!), RefFIT considers the parameters in the “Wp” and “G” columns as the ones of a *special* dielectric function. The particular type of a dielectric function is determined by the number in the “Wo” column. Some special functions depend on more than two parameters, so that two or more rows are necessary to store them.

The list of the built-in special dielectric function formulas is given in Table 4-3. The selection might look strange at a first glance. The point is that it is not meant to be a comprehensive collection of functions for fitting. It is rather a set of some special formulas that

ReffIT users, including myself, have ever employed in their data analysis. As was mentioned in the Introduction, the list is being continuously extended and everyone can ask me to include an extra function that he (or she) would like to have. The light frequency  $\omega$  is assumed to be in  $\text{cm}^{-1}$ . Note, that all the formulas give a dielectric function that satisfies Kramers-Kronig relations (see section 2.2.1), unless the opposite is explicitly mentioned.

Dielectric function	Wo (code)	Wp	G
Formula for the dielectric function of a non-Fermi liquid (Ref.[11], formula (15))	-1	$\omega_p$ [ $\text{cm}^{-1}$ ]	$\gamma_1$ [ $\text{cm}^{-1}$ ]
$\varepsilon = -\frac{\omega_p^2}{\omega(\omega + i\gamma_1)^{2\alpha}(\omega + i\gamma_2)^{1-2\alpha}}$	-2	$\alpha$	$\gamma_2$ [ $\text{cm}^{-1}$ ]
Formula for the c-axis dielectric function of cuprates with the body-centered tetragonal structure liquids (Ref.[11], formula (14))	-3	$\sigma_0$ [ $\Omega^{-1}\text{cm}^{-1}$ ]	-
$\varepsilon = (60i/\omega)2\sigma_0[(1+2\Omega^2)(1-8\Omega^4-8\Omega^2)+16\Omega^3(1+\Omega^2)^{3/2}],$ where $\Omega = \sqrt{(\gamma_1 - i\omega)/\gamma_2}$	-4	$\gamma_1$ [ $\text{cm}^{-1}$ ]	$\gamma_2$ [ $\text{cm}^{-1}$ ]
Formula for the ab-plane dielectric function of cuprates (Ref.[11], formula (3))	-5	$\sigma_0$ [ $\Omega^{-1}\text{cm}^{-1}$ ]	-
$\varepsilon = (60i/\omega)\sigma_0 \frac{\gamma_1}{\sqrt{\gamma_1 - i\omega}\sqrt{\gamma_1 + \gamma_2 - i\omega}},$	-6	$\gamma_1$ [ $\text{cm}^{-1}$ ]	$\gamma_2$ [ $\text{cm}^{-1}$ ]
The dielectric function of a weak-coupling s-wave BCS superconductor with arbitrary scattering rate (Ref. [12]). Parameter $t$ is the reduced temperature $t = T/T_c$ . The formulas of Ref.[12] are taken as they are, with $\tau = 1/\gamma$ , except for the temperature dependence of the gap, which in this case is	-7	$\omega_p$ [ $\text{cm}^{-1}$ ]	$\gamma$ [ $\text{cm}^{-1}$ ]
$\Delta(t) = \Delta_0 \sqrt{\cos \frac{\pi}{2} t}.$ <i>Note: the computational code is taken from Ref..[12].</i>	-8	$t$	$\Delta_0$ [ $\text{cm}^{-1}$ ]
Drude (not Lorentz!) term with arbitrary sign of the spectral weight	-9	$A$ [ $\text{cm}^{-2}$ ]	$\gamma$ [ $\text{cm}^{-1}$ ]
$\varepsilon = -\frac{A}{\omega(\omega + i\gamma)}$ <i>Note: although a negative <math>A</math> would have a little physical meaning for the usual dielectric function, it can be used for the differential dielectric function (see section 4.7.1).</i>			
Constant (frequency independent) dielectric function	-10	$\varepsilon_1$	$\varepsilon_2$
$\varepsilon = \varepsilon_1 + i\varepsilon_2$ <i>Note: this function is simple but obviously NOT Kramers-Kronig friendly, so be careful!</i>			
“Gapped” Drude function:	-20	$\omega_p$ [ $\text{cm}^{-1}$ ]	$\gamma$ [ $\text{cm}^{-1}$ ]

$\varepsilon = \varepsilon_1 + i \frac{4\pi\sigma_1(\omega)}{\omega} + A\delta(\omega), \text{ if } f_{\text{SW}} = 1$ $\varepsilon = \varepsilon_1 + i \frac{4\pi\sigma_1(\omega)}{\omega}, \text{ if } f_{\text{SW}} = 0.$ <p>Here</p> $\sigma_1(\omega) = \Gamma(\omega)\sigma_{1D}(\omega)$ is the “gapped” conductivity, $\varepsilon_1(\omega) = 1 + 8 \int_0^\infty \frac{\sigma_1(\omega')d\omega'}{\omega'^2 - \omega^2}$ is the KK transform of $\sigma_1$ , $\Gamma(\omega) = \xi_0 + (1 - \xi_0) \frac{1}{2} \left( 1 + \tanh\left(\frac{\omega - 2\Delta}{2\delta_\Delta}\right) \right),$ is the smoothed step function, $\sigma_{1D}(\omega) = \frac{\omega_p^2 \gamma}{4\pi(\gamma^2 + \omega^2)}$ is the Drude conductivity.	-21	$\Delta$ [cm <sup>-1</sup> ]	$\delta_\Delta$ [cm <sup>-1</sup> ]
	-22	$\xi_0$	$f_{\text{SW}}$
<p>Here <math>A</math> is the spectral weight added at zero frequency to compensate the loss of spectral weight due to the gap (if <math>f_{\text{SW}} = 1</math>). In other words, <math>A</math> is adjusted in order to keep the total spectral weight as if it there were no gap (e.g., <math>\omega_p^2/8</math>).</p> <p><i>Note:</i> This formula has no microscopic justification. It is just a handy way to introduce the conductivity gap of a controllable shape to the Drude dielectric while keeping the KK relations.</p> <p>The shape depends on the gap value <math>\Delta</math>, the gap spread <math>\delta_\Delta</math> and the filling factor <math>\xi_0</math> (0 for the full gap, 1 for no gap). A similar dielectric (but without the KK transformation) was used in Ref.[13].</p>	-25	$A$ [cm <sup>-2</sup> ]	$\gamma$ [cm <sup>-1</sup> ]
<p>Lorentz term with arbitrary sign of the spectral weight</p> $\varepsilon = \frac{A}{\omega_0^2 - \omega^2 - i\gamma\omega}$ <p><i>Note:</i> although a negative value of <math>A</math> would have little physical meaning for the usual dielectric function, it can be used for the differential dielectric function (see section 4.7.1).</p>	-26	$\omega_0$ [cm <sup>-1</sup> ]	-
<p>The difference between two Lorentzians with the same spectral weight and linewidth, but different frequencies</p> $\varepsilon = \frac{A}{\omega_1^2 - \omega^2 - i\gamma\omega} - \frac{A}{\omega_2^2 - \omega^2 - i\gamma\omega}$ <p><i>Note:</i> this term allows one to model the shift of the oscillator frequency in the differential dielectric function (see section 4.7.1).</p>	-27	$\omega_1$ [cm <sup>-1</sup> ]	$\omega_2$ [cm <sup>-1</sup> ]
	-28	$A$ [cm <sup>-2</sup> ]	$\gamma$ [cm <sup>-1</sup> ]
The same function as for codes (-25) and (-26), just in case one	-29	$A$	$\gamma$

needs more terms of this type.		[cm <sup>-2</sup> ]	[cm <sup>-1</sup> ]
	-30	$\omega_0$ [cm <sup>-1</sup> ]	-
The same function as for codes (-25) and (-26), just in case one needs more terms of this type.	-35	$A$ [cm <sup>-2</sup> ]	$\gamma$ [cm <sup>-1</sup> ]
	-36	$\omega_0$ [cm <sup>-1</sup> ]	-
The same function as for codes (-25) and (-26), just in case one needs more terms of this type.	-37	$A$ [cm <sup>-2</sup> ]	$\gamma$ [cm <sup>-1</sup> ]
	-38	$\omega_0$ [cm <sup>-1</sup> ]	-
The same function as for codes (-25) and (-26), just in case one needs more terms of this type.	-39	$A$ [cm <sup>-2</sup> ]	$\gamma$ [cm <sup>-1</sup> ]
	-40	$\omega_0$ [cm <sup>-1</sup> ]	-
Kramers-Kronig, sum-rule consistent power-law function: $\varepsilon = -\frac{\omega_p^2}{\omega(\omega + i\gamma_1)} F\left(1, 1 - \alpha, 2 - \alpha, \frac{\gamma_2 - \gamma_1}{i\omega - \gamma_1}\right)$ , where $F(a, b, c, z)$ is the hypergeometric function. It shows the powerlaw behavior for $\gamma_1 \ll \omega \ll \gamma_2$ : $\sigma(\omega) \sim (-i\omega)^{-\alpha}$	-33	$\omega_p$ [cm <sup>-1</sup> ]	$\gamma_1$ [cm <sup>-1</sup> ]
	-34	$\alpha$	$\gamma_2$ [cm <sup>-1</sup> ]
Dielectric function of electrons, which scatter on some bosons (for example, phonons) and impurities [16]. $\varepsilon = -\frac{\omega_p^2}{\omega(\omega + iM(\omega, T))}$ , where $\omega_p$ - plasma frequency, $M(\omega, T) = \gamma_{imp} - 2i \int_0^\infty d\Omega B(\Omega) K\left(\frac{\omega}{2\pi T}, \frac{\Omega}{2\pi T}\right)$ - “memory” function, $K(x, y) = \frac{i}{y} + \left\{ \frac{y-x}{x} [\psi(1-ix+iy) - \psi(1+iy)] \right\} - \{y \rightarrow -y\}$ , $\psi(x)$ - digamma function, $\gamma_{imp}$ - impurity scattering rate, $T$ - temperature, $B(\Omega)$ - electron-boson coupling function (in the case of the phonons – transport Eliashberg function $\alpha^2 F(\Omega)$ )  The function $B(\Omega)$ is taken from the dataset, which has a	-41	$\omega_p$ [cm <sup>-1</sup> ]	$\gamma_{imp}$ [cm <sup>-1</sup> ]
	-42	$T$ [K]	#B
	-43	$N_{nodes}$	-

<p>number #B.</p> <p><math>N_{nodes}</math> tells the program how to do the integration. If <math>N_{nodes} = 0</math> then all datapoints of <math>B(\Omega)</math> will be used. If <math>N_{nodes} &gt; 0</math> then the given <math>B(\Omega)</math> will be substituted by a set of <math>N_{nodes}</math> evenly spaced delta-functions, each of those represents the integrated <math>B(\Omega)</math> in the respective frequency region. Note, that the performance of the calculations depends strongly on <math>N_{nodes}</math>, therefore it makes sense to take the minimal possible <math>N_{nodes}</math>, which gives results, close to the case of large <math>N_{nodes}</math>.</p>			
<p>The same function as for codes (-41), (-42) and (-43), just in case one needs more terms of this type.</p>	-44	$\omega_p$ [cm <sup>-1</sup> ]	$\gamma_{imp}$ [cm <sup>-1</sup> ]
	-45	$T$ [K]	#B
	-46	$N_{nodes}$	-
<p>Conductivity Kubo formula in the Marginal Fermi-Liquid theory with a high-frequency cut-off of the boson spectrum.</p> $\varepsilon(\omega) = \frac{\omega_p^2}{2\omega^2} \int_{-\infty}^{+\infty} \left[ \tanh \frac{\beta(y+\omega)}{2} - \tanh \frac{\beta y}{2} \right] \frac{dy}{\Sigma(y+\omega) - \Sigma^*(y) - \omega}$ <p>with the self-energy <math>\Sigma(\omega) = \Sigma'(\omega) + i\Sigma''(\omega)</math>:</p> $\Sigma'(\omega) = \frac{\lambda}{2} [g(\omega + \pi T) + g(\omega - \pi T) - g(\omega + \omega_c) - g(\omega - \omega_c)],$ $\Sigma''(\omega) = -\frac{\lambda\pi}{2} [\theta( \omega  - \pi T) \min( \omega , \omega_c) + \theta(\pi T -  \omega )\pi T] - \gamma_{imp},$ <p>where</p> <p><math>g(x) \equiv x \ln x </math>, <math>\theta(x)</math> - Heaviside step function,  <math>\omega_p</math> - plasma frequency, <math>\gamma_{imp}</math> - impurity scattering rate,  <math>T</math> - temperature, <math>\beta \equiv 1/T</math>,  <math>\lambda</math> - coupling constant, <math>\omega_c</math> - high-frequency cutoff.</p> <p>If Flag=0, then the above (Kramers-Kronig-consistent) expression for the self-energy is used. If Flag=1, then the ‘original’, (not Kramers-Kronig consistent) expression is used:</p> $\Sigma(\omega) = \lambda\omega \ln \frac{x}{\omega_c} - i \frac{\pi\lambda}{2} x, \text{ where } x = \max( \omega , \pi T).$	-51	$\omega_p$ [cm <sup>-1</sup> ]	$\gamma_{imp}$ [cm <sup>-1</sup> ]
	-52	$\lambda$	$\omega_c$ [cm <sup>-1</sup> ]
	-53	$T$ [K]	Flag
	-54	C1	C2

<p><math>C_1</math> and <math>C_2</math> are technical parameters which tell the program how to perform the integration. The integral limits are taken from <math>-C_1T - \omega</math> to <math>C_1T</math> with a step of <math>T/C_2</math>. Increasing <math>C_1</math> and <math>C_2</math> will increase the accuracy but slow down the calculation. The recommended values are <math>C_1 = 10</math>, <math>C_2 = 2</math>.</p>			
<p>The factorized (TO-LO) formula for the dielectric function:</p> $\varepsilon = \varepsilon_\infty \times \frac{\prod_{i=1}^{n_{LO}} (\Omega_{i,LO}^2 - \omega^2 - i\gamma_{i,LO}\omega)}{\prod_{i=1}^{n_{TO}} (\Omega_{i,TO}^2 - \omega^2 - i\gamma_{i,TO}\omega)}, \text{ where}$ <p><math>\varepsilon_\infty</math> is the high-frequency dielectric constant (note: it is different from the one in the Drude-Lorentz model!),  <math>\Omega_{i,TO}</math>, <math>\Omega_{i,LO}</math> are the transversal (longitudinal) frequencies,  <math>\gamma_{i,TO}</math>, <math>\gamma_{i,LO}</math> are the transversal (longitudinal) scattering rates.  The advantage of this formula with respect to the DL model is that it allows different scattering rates for the TO and LO modes.  ReffFIT determines <math>n_{TO}</math> as the total number of rows with the code (-1001) and <math>n_{LO}</math> as the total number of rows with the code (-1002). There can be as many rows with codes (-1001) and (-1002) as necessary.</p> <p>In principle, physically sensible are only such models, where <math>n_{TO} = n_{LO}</math>, and the TO and LO frequencies are alternated: <math>\Omega_{1,TO} \leq \Omega_{1,LO} \leq \Omega_{2,TO} \leq \Omega_{2,LO} \leq \dots</math>. Even then, some unphysical artifacts, like a negative value of <math>\varepsilon_2</math>, may happen, if improper scattering rates are chosen. More discussion of this model can be found, for instance in Ref. 14.  Be careful: ReffFIT blindly feeds the parameters to the TO-LO formula. It is the user's responsibility to provide the right initial parameters and to check that they are still OK after the fitting!</p>	-1000	$\varepsilon_\infty$	-
	-1001	$\Omega_{i,TO}$ [cm <sup>-1</sup> ]	$\gamma_{i,TO}$ [cm <sup>-1</sup> ]
	-1002	$\Omega_{i,LO}$ [cm <sup>-1</sup> ]	$\gamma_{i,LO}$ [cm <sup>-1</sup> ]

Table 4-3 Special dielectric functions.

### 4.6.3. Variational dielectric function

Variational dielectric functions (VDFs) cannot be described by a single mathematical formula. Instead, the values of the dielectric function *at a mesh of the anchor frequency points*  $\omega_1 \dots \omega_N$  are considered as independent parameters. The theory behind this approach is given in sections 2.2.4 and 2.2.5.

A VDF can be *added* to the total model dielectric function. The user can switch it on and off, or can make it adjustable (active in fit), or fixed. The anchor frequency set  $\omega_1 \dots \omega_N$  is specified when the VDF is initialized.

There exist two types of VDFs: the KK-constrained (Kramers-Kronig-constrained) ones and the not KK-constrained ones. In the not-KK-constrained regime both  $\epsilon_1$  and  $\epsilon_2$  are treated as independently varied parameters at every frequency point (see section 2.2.5). In the KK regime only  $\epsilon_2$  is treated as an independent function, while  $\epsilon_1$  is given by the KK-transform of  $\epsilon_2$  (see section 2.2.4). In the KK mode  $\epsilon_2$  is set to zero at the edges of the spectral range, in order to avoid a discontinuity of  $\epsilon_2$ , which causes divergent singularities in  $\epsilon_1$ . Thus, a KK-constrained VDF has  $N - 2$  parameters, while a non-KK-constrained one has  $2N$  parameters.

The VDF controls are collected in the ‘Variational Diel. Function’ button group of the model window, which can be made visible or invisible by clicking the **VDF>>** button (see Figure 4-5).

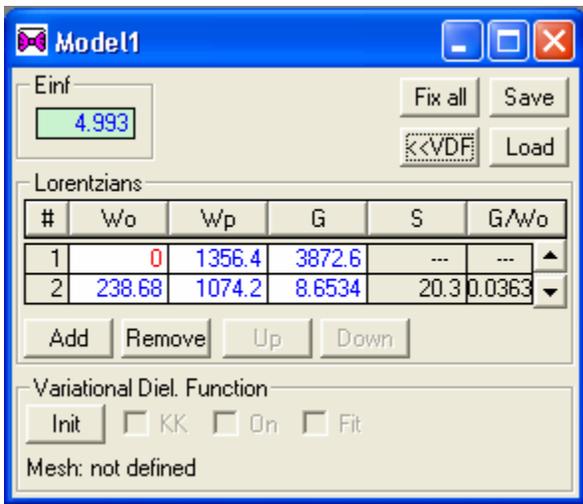


Figure 4-5 The “Dielectric function” model window with the VDF controls visible (the anchor mesh is not defined yet).

When a new model is created, the VDF of this model is deactivated and its anchor mesh is not defined. In order to initialize VDF and set its mesh, one has to press the **Init** button (or press **F6**). The ‘Initialize VDF’ window will show up (Figure 4-8).

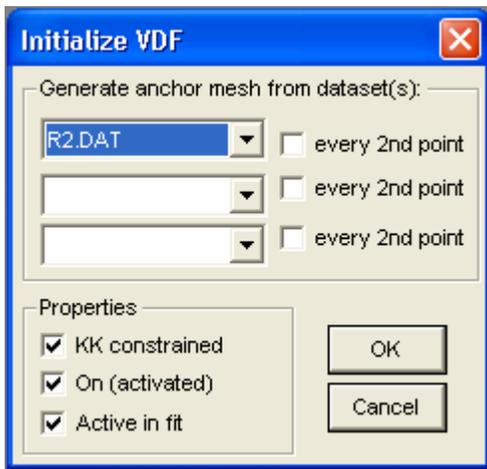


Figure 4-6 The “Initialize VDF” window.

The anchor mesh can be generated on the base of one or more (up to three) loaded datasets (see Section 4.8). If ‘every 2nd point’ button is not checked, then all frequency points from the datasets will be used as anchor points of the VDF. Otherwise, every 2<sup>nd</sup> data point will be used. The ‘every 2nd point’ may be sometimes helpful to avoid some numerical instabilities of the fitting routine (typically, fast spurious oscillations of the resulting  $\varepsilon_1$  and  $\varepsilon_2$ ). The VDF properties - KK constraint, on/off and fitting activity - can also be set here.

When VDF is initialized, the information about the anchor mesh is indicated (Figure 4-7). A VDF can be switched between KK and non-KK-constrained regimes by checking/unchecking the ‘KK’ button (or by pressing **F5**). It can be switched on and off (without removing it from the memory) by checking/unchecking the ‘On’ button (or by pressing **F4**). It can be made active in fit (adjustable) or fixed with the ‘Fit’ button (or by pressing **F7**).

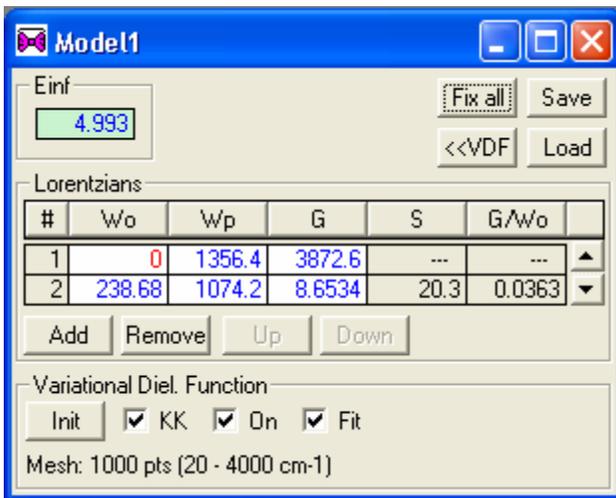


Figure 4-7 The “Dielectric function” model window with the VDF controls visible (the anchor mesh defined).

A VDF is not saved together with the main model (when using button **F2**). However, this shortcoming is easily bypassed by using macros to generate VDFs ‘on-fly’ or by exporting the model  $\varepsilon_1$  and  $\varepsilon_2$  to a file.

It worth mentioning that to handle VDF in the KK-constrained mode requires a lot of computational work and may significantly slow down the program. Therefore one should avoid the fitting of unnecessary large datasets. On an average modern PC (at the moment of writing this manual it is Pentium 1.5-2.5 GHz) RefFIT manages easily with  $N \sim 1000$ . Working with larger datasets would require some patience.

## 4.7. Special models

As was mentioned in section 4.5, special models are distinguished from the “Dielectric Function” model by a ‘code’, which is stored in the parameter “Einf”. The table “Lorentzians” is used to store the model parameters, whose meaning depends solely on the model type. Also, the output quantities are different from the ones of the “Dielectric Function” model (even though the same abbreviations are used, as in Table 4-2). Special models often contain links to other models.

Table 4-4 lists special models and the corresponding codes. The detailed description of special models is presented in the subsequent sections.

“Einf” (Code)	Special model
-1	Differential dielectric function (section 4.7.1)
-2	Reflection and transmission of a multi-layer sample (section 4.7.2)
-3	Ellipsometry of an orthorhombic sample (section 4.7.3)
-4	Ellipsometry of an orthorhombic sample (differential model) (section 4.7.4)
-5	Ellipsometry of an orthorhombic film on an orthorhombic substrate (section 4.7.5)
-6	Extended Drude (section 4.7.6)
-9	Epsilon+Mu (section 4.7.7)
-10	Spectral Weight (section 4.7.8)
-13	Optics of a plate sample (section 4.7.9)
-18	Optics of a plate sample (section 4.7.10)

Table 4-4 Special models.

### 4.7.1. Differential dielectric function

This model is intended to fit *differential* spectra of any quantity that can be generated by the “Dielectric function” model (section 4.6). The theoretical description of differential spectra is presented in section 2.3.

A model of the “Differential dielectric function” type combines the two models of the “Dielectric function” type. The first dielectric function is treated as a *base* model and the second one – as a *differential* model. The description of the parameters is given in Table 4-5.

#	Wo	Wp	G
1	Base model #	-	-
2	Differential model #	-	-

Table 4-5 The meaning of the parameters of the “Differential dielectric function” model.

The output quantities of this model are the differential analogs of the quantities, generated by the “Dielectric function” model (see Table 4-2). For instance, the quantity “S1” is the differential conductivity.

The Figure 4-8 presents an example of the “Differential dielectric function” model. It is assumed here that the base DF is stored in the “Model2” and the differential DF is in “Model3”. Note that all parameters of the “Model1” are fixed. The output quantity “R” of the “Model1” will correspond to the differential reflectivity  $\Delta R$ , “T” will correspond to the differential transmission  $\Delta T$  and so forth.



Figure 4-8 An example of the “Differential dielectric function” model.

Usually the fitting of differential spectra is done in two steps. On the first step the base model is adjusted to fit the base spectra. On the second step all the parameters of the base model are fixed and the differential model is adjusted to fit the differential spectra. Note that the internal parameters of the base model should be fixed on the second step.

#### 4.7.2. Reflection and transmission of a multi-layer sample

This model has been designed to fit the normal-incidence reflection and transmission spectra of samples that consist of an arbitrary number of layers (Figure 4-9). Each layer is supposed to be isotropic and described by its own complex dielectric function  $\epsilon$ , thickness  $d$  and thickness spread  $\delta_d$ .

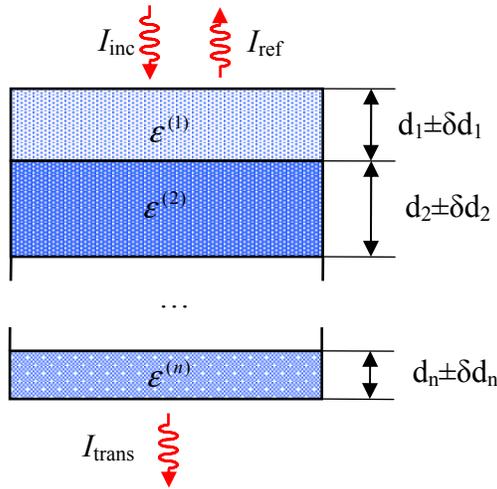


Figure 4-9 The experimental configuration of the model “Reflection and transmission of a multi-layer sample”.

The sample reflectivity is defined as  $R = I_{ref} / I_{inc}$ , transmission is defined as  $T = I_{trans} / I_{inc}$ , where  $I_{inc}$  is the intensity of the incident light,  $I_{ref}$  - intensity of the reflected light and  $I_{trans}$  - intensity of the transmitted light. The exact formulas for  $R$  and  $T$  are rather cumbersome; they can be found, for instance, in Ref. [15].

The parameters of this model are described in Table 4-6. The number of layers is given by the number of rows in the “Lorentzians” table. Each row corresponds to a particular layer.

#	Wo	Wp	G
1	DF model (layer #1)	Thickness d (layer #1) [cm]	Relative thickness spread $\delta d/d$ (layer #1)
2	DF model (layer #2)	Thickness d (layer #2) [cm]	Relative thickness spread $\delta d/d$ (layer #2)
...	...	...	...
N	DF model (layer #n)	Thickness d (layer #n) [cm]	Relative thickness spread $\delta d/d$ (layer #n)

Table 4-6 The parameters of the model “Reflection and transmission of a multi-layer sample”.

An example of this of kind of model window is shown in Table 4-5. Here a two-layer system (film-substrate) is modeled. The top layer (film) is described by the dielectric function from “Model2”; it has a thickness of  $1.76e-6$  cm (17.6 nm) and relative thickness spread of 1%. The substrate is described by the dielectric function from “Model3” and it is 0.057 cm (570 micron) thick with a relative spread of 3%. Note that the film thickness is treated as a variable parameter (marked blue).

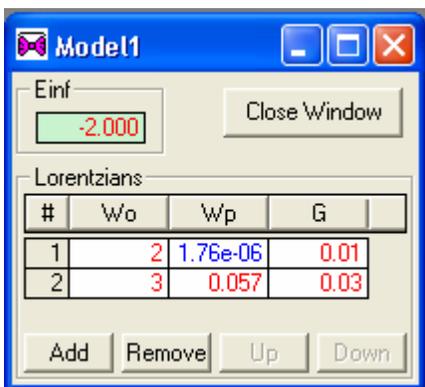


Figure 4-10 An example of the model “Reflection and transmission of a multi-layer sample”.

Table 4-7 describes the meaning of the output quantities.

It saves a lot of computational time to set the thickness spread to zero unless it is really necessary to model a wedged sample.

Quantity abbreviation	Description
“R”	$R =  r ^2$
“T”	$T =  t ^2$
“E1”	$\text{Re } r$
“E2”	$\text{Im } r$
“S1”	$\text{Re } t$
“S2”	$\text{Im } t$

Table 4-7 The output quantities of the model “Reflection and transmission of a multi-layer sample”.

### 4.7.3. Ellipsometry of an orthorhombic sample

This model is designed to fit the ellipsometric measurements done on an anisotropic semi-infinite sample. Of course, it can be also used (as the simplest case) for an isotropic sample. The experimental geometry, including the orientation of principal dielectric tensor axes, is shown in Figure 4-11.

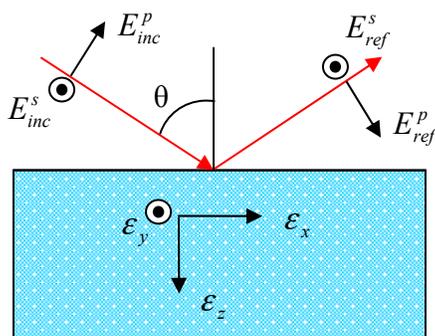


Figure 4-11 The experimental configuration of the model “Ellipsometry of an orthorhombic sample”.

As this manual is not an ellipsometry textbook, we present only the formulas used. The complex reflectivities for the p- and s-polarized light are:

$$r_p \equiv \frac{E_{ref}^p}{E_{inc}^p} = \frac{\sqrt{1 - \frac{\sin^2 \theta}{\epsilon_z}} - \sqrt{\epsilon_x} \cos \theta}{\sqrt{1 - \frac{\sin^2 \theta}{\epsilon_z}} + \sqrt{\epsilon_x} \cos \theta} \quad \text{and} \quad r_s \equiv \frac{E_{ref}^s}{E_{inc}^s} = \frac{\cos \theta - \sqrt{\epsilon_y - \sin^2 \theta}}{\cos \theta + \sqrt{\epsilon_y - \sin^2 \theta}}.$$

In ellipsometry the complex ratio  $\rho = r_p / r_s$  is measured, commonly expressed in terms of the two real parameters  $\psi$  and  $\Delta$ :  $\rho = \tan \psi e^{i\Delta}$ . The inversion of this formula gives  $\psi = \arctan|\rho|$  and  $\Delta = \arg \rho$ .

The pseudo-dielectric function is defined as:

$$\epsilon_{pseudo} = \sin^2 \theta \left[ 1 + \tan^2 \theta \left( \frac{1 + \rho}{1 - \rho} \right)^2 \right].$$

In the case of an isotropic sample the  $\epsilon_{pseudo}$  coincides with the true dielectric function  $\epsilon$ .

One of the possible ellipsometry configurations involves a fixed polarizer (at angle  $P$ ) and a rotating analyzer (without a retarder). In this case the Fourier coefficients of the detector signal as functions of the analyzer angle are:

$$\alpha = \frac{\tan^2 \psi - \tan^2 P}{\tan^2 \psi + \tan^2 P}, \quad \beta = \frac{2 \tan \psi \cos \Delta \tan P}{\tan^2 \psi + \tan^2 P}.$$

The description of the model parameters is given in Table 4-8.

#	Wo	Wp	G
1	DF model for $\epsilon_x$	$\theta$ [degrees]	P [degrees]
2	DF model for $\epsilon_y$	-	-
3	DF model for $\epsilon_z$	-	-

Table 4-8 The parameters of the model “Ellipsometry of an orthorhombic sample”.

An example of such a model is shown in Figure 4-12. It is assumed that the “Model2” contains  $\epsilon_x$ , the “Model3” contains  $\epsilon_y$  and “Model4” contains  $\epsilon_z$ . The angle of incidence is  $80^\circ$  and the polarizer angle is  $45^\circ$ .

To imitate the isotropic sample, one should assign the dielectric components  $\epsilon_x$ ,  $\epsilon_y$  and  $\epsilon_z$  may to the same model. For a uniaxial crystal two of the three components should refer to the same “Dielectric function” model.



Figure 4-12 An example of the model “Ellipsometry of an orthorhombic sample”.

Table 4-9 describes the meaning of the output quantities. Note again, that quantity abbreviations here have no meaning whatsoever.

Quantity abbreviation	Description
“E1”	$\psi$ [degrees]
“E2”	$\Delta$ [degrees]
“S1”	$\alpha$
“S2”	$\beta$
“N1”	$\text{Re } \varepsilon_{pseudo}$
“N2”	$\text{Im } \varepsilon_{pseudo}$

Table 4-9 The output quantities of the model “Ellipsometry of an orthorhombic sample”.

#### 4.7.4. Differential ellipsometry of an orthorhombic sample

This model is the ‘differential’ extension of the model “Ellipsometry of an orthorhombic sample”. The details about differential models are given in section 2.3.

The description of the model parameters is given in Table 4-8.

#	Wo	Wp	G
1	The base model for $\varepsilon_x$	The differential model for $\varepsilon_x$	$\theta$ [degrees]
2	The base model for $\varepsilon_y$	The differential model for $\varepsilon_y$	$P$ [degrees]
3	The base model for $\varepsilon_z$	The differential model for $\varepsilon_z$	-

Table 4-10 The parameters of the model “Differential ellipsometry of an orthorhombic sample”.

The output quantities are the differential versions of quantities listed in Table 4-9.

#### 4.7.5. Ellipsometry of an orthorhombic film on an orthorhombic substrate

This model is an extension of the ellipsometry to the double-layer system, like a film on a substrate (Figure 4-13). The symmetry of both film and substrate dielectric tensors can be as low as orthorhombic. The substrate is treated as a semi-infinite one.

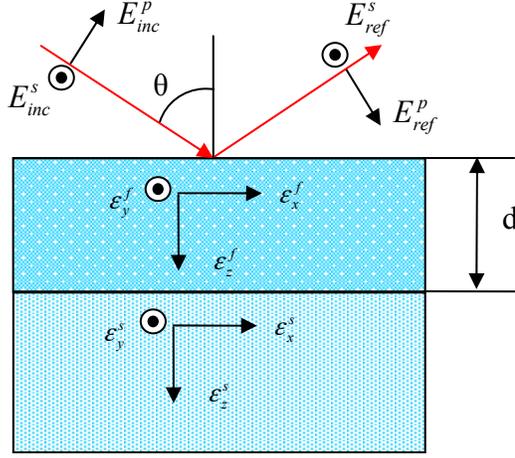


Figure 4-13 The experimental configuration of the model “Ellipsometry of an orthorhombic film on an orthorhombic substrate”.

All the formulas are the same as for the model “Ellipsometry of an orthorhombic Sample” (section 4.7.3), except the ones for  $r_p$  and  $r_s$ , which are a bit more lengthy:

$$r_p = \frac{r_p^{of} + t_p r_p^{fs}}{1 + t_p r_p^{of} r_p^{fs}}, \quad r_s = \frac{r_s^{of} + t_s r_s^{fs}}{1 + t_s r_s^{of} r_s^{fs}}, \quad \text{where}$$

$$r_p^{of} = \frac{\sqrt{1 - \frac{\sin^2 \theta}{\epsilon_z^f}} - \sqrt{\epsilon_x^f} \cos \theta}{\sqrt{1 - \frac{\sin^2 \theta}{\epsilon_z^f}} + \sqrt{\epsilon_x^f} \cos \theta}, \quad r_s^{of} = \frac{\cos \theta - \sqrt{\epsilon_y^f - \sin^2 \theta}}{\cos \theta + \sqrt{\epsilon_y^f - \sin^2 \theta}},$$

$$r_p^{fs} = \frac{\sqrt{\epsilon_x^f} \sqrt{1 - \frac{\sin^2 \theta}{\epsilon_z^s}} - \sqrt{\epsilon_x^s} \sqrt{1 - \frac{\sin^2 \theta}{\epsilon_z^f}}}{\sqrt{\epsilon_x^f} \sqrt{1 - \frac{\sin^2 \theta}{\epsilon_z^s}} + \sqrt{\epsilon_x^s} \sqrt{1 - \frac{\sin^2 \theta}{\epsilon_z^f}}}, \quad r_s^{fs} = \frac{\sqrt{\epsilon_y^f - \sin^2 \theta} - \sqrt{\epsilon_y^s - \sin^2 \theta}}{\sqrt{\epsilon_y^f - \sin^2 \theta} + \sqrt{\epsilon_y^s - \sin^2 \theta}}$$

$$t_p = \exp\left\{4\pi i \omega d \sqrt{\epsilon_x^f} \sqrt{1 - \frac{\sin^2 \theta}{\epsilon_z^f}}\right\}, \quad t_s = \exp\left\{4\pi i \omega d \sqrt{\epsilon_y^f - \sin^2 \theta}\right\}, \quad d \text{ is the thickness of}$$

the top layer (measured in cm),  $\omega$  is the light frequency (measured in  $\text{cm}^{-1}$ ).

The description of the model parameters is given in Table 4-11.

#	Wo	Wp	G
1	DF model for $\epsilon_x^f$	$\theta$ [degrees]	P [degrees]

2	The model for $\epsilon_y^f$	$d$ [cm]	-
3	The model for $\epsilon_z^f$	-	-
4	The model for $\epsilon_x^s$	-	-
5	The model for $\epsilon_y^s$	-	-
6	The model for $\epsilon_z^s$	-	-

Table 4-11 The parameters of the model “Ellipsometry of an orthorhombic film on an orthorhombic substrate”.

An example of such a model is shown in Figure 4-14. It is assumed that the “Model2”, “Model3” and “Model4” describe the  $x$ ,  $y$  and  $z$  components respectively of the film dielectric tensor. The substrate is uniaxial with the symmetry axis  $z$ , normal to the sample surface. The  $x$ - and  $y$ -components of the substrate dielectric tensor are described by the “Model5”, the  $z$ -component is given by the “Model6”. The angle of incidence is  $70^\circ$ , the angle of polarizer is  $45^\circ$  and the film thickness  $0.000532$  cm (5.32 micron).

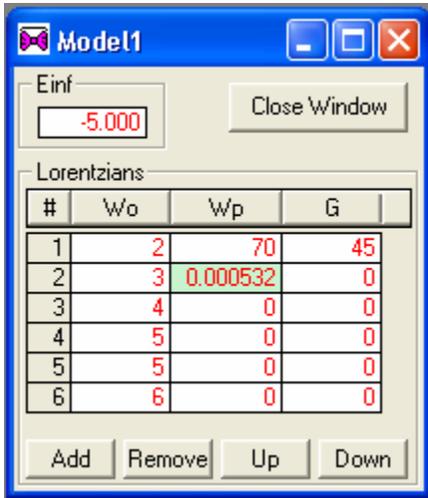


Figure 4-14 An example of the model “Ellipsometry of an orthorhombic film on an orthorhombic substrate”.

#### 4.7.6. Extended Drude

This model takes as an input a dielectric function  $\epsilon(\omega)$  given by some dielectric function model and converts it to the frequency-dependent scattering rate  $1/\tau(\omega)$  and the normalized frequency-dependent effective mass  $m^*(\omega)/m$  according to the formulas:

$$\frac{1}{\tau(\omega)} = -\frac{\omega_p^2}{\omega} \operatorname{Im} \left( \frac{1}{\epsilon(\omega) - \epsilon^\infty} \right),$$

$$\frac{m^*(\omega)}{m} = -\frac{\omega_p^2}{\omega^2} \operatorname{Re} \left( \frac{1}{\epsilon(\omega) - \epsilon^\infty} \right).$$

Here  $\omega_p$  and  $\varepsilon^\infty$  are considered as parameters of the extended Drude model (note, that they have nothing to do with the parameters of the model, which calculates  $\varepsilon(\omega)$ ). In addition another ‘scattering rate’  $1/\tau^*(\omega)$  is calculated:

$$\frac{1}{\tau^*(\omega)} = \frac{\omega \operatorname{Im}\left(\frac{1}{\varepsilon(\omega) - \varepsilon^\infty}\right)}{\operatorname{Re}\left(\frac{1}{\varepsilon(\omega) - \varepsilon^\infty}\right)} = \frac{1/\tau(\omega)}{m^*(\omega)/m}.$$

Note, that  $1/\tau^*(\omega)$  does not depend on the parameter  $\omega_p$ .

The description of the model parameters is given in Table 4-12.

#	Wo	Wp	G
1	The model for $\varepsilon(\omega)$	$\omega_p$ [ $\text{cm}^{-1}$ ]	$\varepsilon^\infty$

Table 4-12 The model parameters of the model “Extended Drude”.

Table 4-13 describes the meaning of the output quantities.

Quantity abbreviation	Description
“R”	$1/\tau(\omega)$ [ $\text{cm}^{-1}$ ]
“T”	$1/\tau^*(\omega)$ [ $\text{cm}^{-1}$ ]
“Rp”	$m^*(\omega)/m$

Table 4-13 The output quantities of the model “Extended Drude”.

Figure 4-15 gives an example of the extended Drude model. The complex value of  $\varepsilon(\omega)$  is taken from “Model 2”,  $\omega_p = 12000 \text{ cm}^{-1}$  and  $\varepsilon^\infty = 3.6$ .

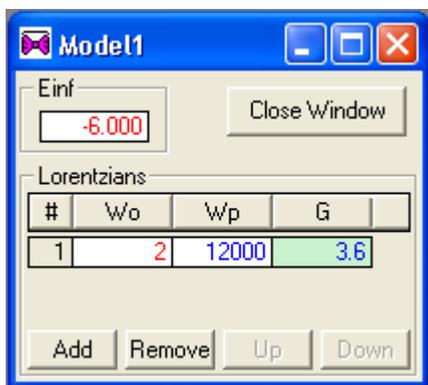


Figure 4-15 An example of the model “Extended Drude”

#### 4.7.7. Epsilon+Mu

This model calculates optical properties of a medium, which has not only the dielectric function  $\varepsilon(\omega) \neq 1$ , but also an arbitrary the magnetic permeability  $\mu(\omega) \neq 1$ .

The description of the model parameters is given in Table 4-12.

#	Wo	Wp	G
1	The model for $\epsilon(\omega)$	The model for $\mu(\omega)$	-

Table 4-14 The model parameters of the model “Epsilon+Mu”.

Even though the model for  $\mu(\omega)$  is formally of the type “Dielectric function”, its output is assumed to be the magnetic permeability.

Table 4-15 lists all output quantities of the “Epsilon+Mu” model. Note that they have basically the same physical meaning as they do in the case of the “Dielectric function” model (Table 4-2).

Quantity abbreviation	Description	Calculation formula(s)	Units	Experimental parameters used
“R”	Normal-incidence reflectivity of semi-infinite sample	$\alpha_R  r ^2$ , where $r = \frac{\sqrt{\mu} - \sqrt{\epsilon}}{\sqrt{\mu} + \sqrt{\epsilon}}$	-	Scaling factor of $R$ ( $\alpha_R$ )
“Rph”	Complex phase of the normal-incidence reflectivity of a semi-infinite sample	$\arg(r) \equiv \arctan(\text{Im}(r)/\text{Re}(r))$ , where $r = \frac{\sqrt{\mu} - \sqrt{\epsilon}}{\sqrt{\mu} + \sqrt{\epsilon}}$	-	-
“T”	Normal-incidence transmission of a finite-thickness sample, considering multiple (Fabry-Perot) internal reflections and a possible thickness spread	$\alpha_T \langle T \rangle_{average}$ , where $\langle T \rangle_{average} = \frac{1}{N+1} \sum_{i=0}^N T(d_i)$ , $d_i = d \left( 1 + s_d \frac{i - N/2}{N} \right)$ , ( $N = 10$ ), $T(d) = \left  \frac{(1-r^2)t(d)}{1-r^2t^2(d)} \right ^2$ , $r = \frac{\sqrt{\mu} - \sqrt{\epsilon}}{\sqrt{\mu} + \sqrt{\epsilon}}$ $t(d) = \exp\left( i \frac{\omega}{c} \sqrt{\epsilon \mu} d \right)$	-	sample thickness ( $d$ ), thickness spread ( $s_d$ ), scaling factor of $T$ ( $\alpha_T$ )
“TT”	Normal-incidence transmission of a finite-thickness sample, ignoring	$\alpha_T \langle T \rangle_{average}$ , where $\langle T \rangle_{average} = \frac{1}{N+1} \sum_{i=0}^N T(d_i)$ ,	-	sample thickness ( $d$ ), thickness spread ( $s_d$ ), scaling factor of $T$ ( $\alpha_T$ )

	multiple (Fabry-Perot) internal reflections but considering a possible thickness spread	$d_i = d \left( 1 + s_d \frac{i - N/2}{N} \right), (N = 10),$ $T(d) =  (1 - r^2)t(d) ^2,$ $r = \frac{\sqrt{\mu} - \sqrt{\epsilon}}{\sqrt{\mu} + \sqrt{\epsilon}}$ $t(d) = \exp \left( i \frac{\omega}{c} \sqrt{\epsilon \mu} d \right)$		
“Rp”	Grazing-incidence reflectivity (p-polarization)	$\alpha_R \langle R_p \rangle_{average}, \text{ where}$ $\langle R_p \rangle_{average} = \frac{1}{N+1} \sum_{i=0}^N R_p(\theta_i),$ $\theta_i = \theta \left( 1 + s_\theta \frac{i - N/2}{N} \right), (N = 10),$ $R_p(\theta) =  r_p(\theta) ^2,$ $r_p(\theta) = \frac{\sqrt{\epsilon} \cos \theta - \sqrt{\mu - \frac{\sin^2 \theta}{\epsilon}}}{\sqrt{\epsilon} \cos \theta + \sqrt{\mu - \frac{\sin^2 \theta}{\epsilon}}}$	-	angle of incidence ( $\theta$ ), angle spread ( $s_\theta$ ), scaling factor of $R$ ( $\alpha_R$ )
“Rs”	Grazing-incidence reflectivity (s-polarization)	$\alpha_R \langle R_s \rangle_{average}, \text{ where}$ $\langle R_s \rangle_{average} = \frac{1}{N+1} \sum_{i=0}^N R_s(\theta_i),$ $\theta_i = \theta \left( 1 + s_\theta \frac{i - N/2}{N} \right), (N = 10),$ $R_s(\theta) =  r_s(\theta) ^2$ $r_s(\theta) = \frac{\sqrt{\mu} \cos \theta - \sqrt{\epsilon - \frac{\sin^2 \theta}{\mu}}}{\sqrt{\mu} \cos \theta + \sqrt{\epsilon - \frac{\sin^2 \theta}{\mu}}}$	-	angle of incidence ( $\theta$ ), angle spread ( $s_\theta$ ), scaling factor of $R$ ( $\alpha_R$ )
“Rps”	Ratio between Rp and Rs	$\langle R_{ps} \rangle_{average}, \text{ where}$ $\langle R_{ps} \rangle_{average} = \frac{1}{N+1} \sum_{i=0}^N R_{ps}(\theta_i),$	-	angle of incidence ( $\theta$ ), angle spread ( $s_\theta$ )

		$\theta_i = \theta \left( 1 + s_\theta \frac{i - N/2}{N} \right), (N = 10),$ $R_{ps}(\theta) =  r_{ps}(\theta) ^2$ $r_{ps}(\theta) = \frac{r_p}{r_s} =$ $\frac{(1 - \epsilon\mu) \sin^2 \theta - (\mu - \epsilon) \cos \theta \sqrt{\epsilon\mu - \sin^2 \theta}}{(1 - \epsilon\mu) \sin^2 \theta + (\mu - \epsilon) \cos \theta \sqrt{\epsilon\mu - \sin^2 \theta}}$		
“N1”	The real part of the refraction index	$\text{Re} \sqrt{\epsilon\mu}$	-	-
“N2”	The imaginary part of the refraction index (extinction coefficient)	$\text{Im} \sqrt{\epsilon\mu}$	-	-
“PD”	Penetration depth	$\frac{1}{2\pi \text{Im} \sqrt{\epsilon\mu\omega}}$	cm	-

Table 4-15 The output quantities of the “Epsilon+Mu” model.

Figure 4-15 gives an example of the model “Epsilon+Mu”. The complex value of  $\epsilon(\omega)$  is taken from “Model 1”, The complex value of  $\mu(\omega)$  is taken from “Model 2”.

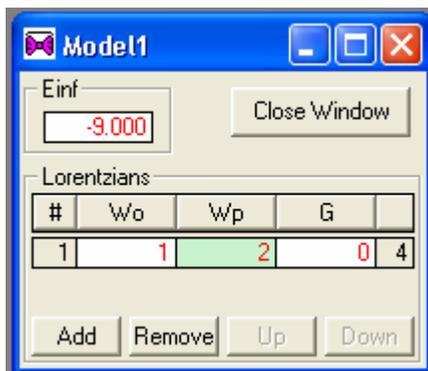


Figure 4-16 An example of the model “Epsilon+Mu”

### 4.7.8. Spectral Weight

This model takes as an input a dielectric function  $\epsilon(\omega)$  given by some dielectric function model and calculates the integrated spectral weight:

$$SW(\omega) = \int_0^{\omega} \sigma_1(x) dx = \frac{1}{60} \int_0^{\omega} x \epsilon_2(x) dx,$$

where  $\sigma_1$  is measured in  $\Omega^{-1}\text{cm}^{-1}$ ,  $\omega$  is measured in  $\text{cm}^{-1}$ . It also calculates the related quantities: the effective plasma frequency  $\omega_p(\omega)$  and effective number of carriers  $N_{eff}(\omega)$ .

The calculations are done analytically. At the moment, it works correctly only for the usual Lorentz function (see section 4.6.1), Lorentz-derived special functions (codes -25, -26, etc, see section 4.6.2) and variational dielectric function (see section 4.6.3). If other special dielectric functions are used, then the returned spectral weight is set to zero.

#	Wo	Wp	G
1	The model for $\varepsilon(\omega)$	The unit cell volume $V_{\text{cell}} [\text{cm}^3]$	--

Table 4-16 The model parameters of the model ‘‘Spectral Weight’’.

Table 4-13 describes the meaning of the output quantities.

Quantity abbreviation	Description
‘‘R’’	SW [ $\Omega^{-1}\text{cm}^{-2}$ ]
‘‘T’’	$\omega_p = \sqrt{\frac{120}{\pi} SW} [\text{cm}^{-1}]$
‘‘Rp’’	$N_{eff} = C * SW * Vc$ [dimensionless], where $C = 4.2568 \cdot 10^{14}$ , $V_{\text{cell}}[\text{cm}^3]$ is the unit cell volume.

Table 4-17 The output quantities of the model ‘‘Spectral Weight’’.

Figure 4-15 gives an example of this special model. The value of  $\varepsilon(\omega)$  is taken from ‘‘Modell’’’. The unit cell volume is  $2.89 \cdot 10^{-23} \text{cm}^3$ .

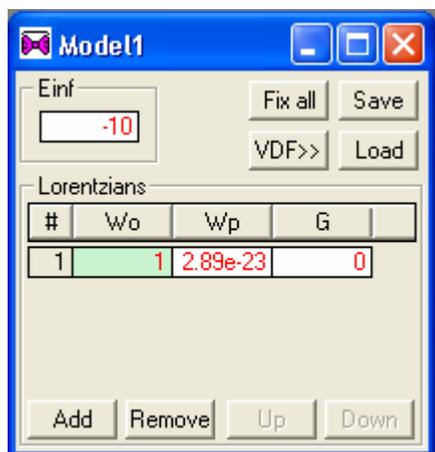


Figure 4-17 An example of the model ‘‘Spectral Weight’’

### 4.7.9. Optics of a plate sample

This model can calculate reflectivity, transmission and ellipsometric parameters (in both reflection and transmission regimes) of a plate sample. The material is assumed to be optically isotropic. The angle of incidence and the angle of polarization of the incident light can be set. In contrast to other models (such as special models 2 and 5) this model offers a number of possibilities to treat the multiple internal reflections (Fabry-Perot effect) and the degree of optical phase coherence between different internal rays. The experimental geometry is shown in Figure 4-11.

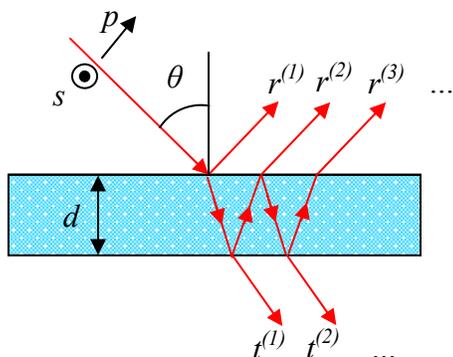


Figure 4-18 The experimental configuration of the model “Ellipsometry of an orthorhombic sample”.

Below we present the formulas used by this model.

Bulk complex reflectivities ( $p$ , and  $s$  refer to the p- and s-polarized light):

$$r_p = \frac{\sqrt{\varepsilon - \sin^2 \theta} - \varepsilon \cos \theta}{\sqrt{\varepsilon - \sin^2 \theta} + \varepsilon \cos \theta}, \quad r_s = \frac{\cos \theta - \sqrt{\varepsilon - \sin^2 \theta}}{\cos \theta + \sqrt{\varepsilon - \sin^2 \theta}},$$

where  $\varepsilon$  is the complex dielectric constant,  $\theta$  is the angle of incidence.

Complex transmission:

$$\tau = \exp\left\{2\pi i \omega d \sqrt{\varepsilon - \sin^2 \theta}\right\},$$

where  $d$  is the sample thickness (measured in cm),  $\omega$  is the light frequency (measured in  $\text{cm}^{-1}$ ).

Complex reflectivities of individual rays:

$$r_{p,s}^{(n)} = \begin{cases} r_{p,s} & , n = 1 \\ -r_{p,s} \tau^2 (1 - r_{p,s}^2) (r_{p,s}^2 \tau^2)^{n-2} & , n > 1 \end{cases}$$

The total complex reflectivity (all rays added with some weighting coefficients  $\alpha_n$ ):

$$\tilde{r}_{p,s} = \sum_{n=1}^{\infty} \alpha_n r_{p,s}^{(n)}$$

Equation 4-1

The power coefficients, which correspond to the full phase coherence between different rays:

$$A_{pp} = |\tilde{r}_p|^2, \quad A_{ss} = |\tilde{r}_s|^2, \quad A_{ps} = \frac{\tilde{r}_p \tilde{r}_s^* + \tilde{r}_s \tilde{r}_p^*}{2}.$$

Equation 4-2

The auxiliary coefficients, which correspond to the absence of any coherence between different rays:

$$B_{pp} = \sum_{n=1}^{\infty} \alpha_n |r_p^{(n)}|^2, \quad B_{ss} = \sum_{n=1}^{\infty} \alpha_n |r_s^{(n)}|^2, \quad B_{ps} = \sum_{n=1}^{\infty} \alpha_n \frac{r_p^{(n)} r_s^{(n)*} + r_s^{(n)} r_p^{(n)*}}{2}$$

Equation 4-3

One can roughly introduce a partial coherence ( $0 \leq \gamma \leq 1$ ):

$$C_{pp} = \gamma B_{pp} + (1 - \gamma) A_{pp},$$

$$C_{ss} = \gamma B_{ss} + (1 - \gamma) A_{ss},$$

$$C_{ps} = \gamma B_{ps} + (1 - \gamma) A_{ps},$$

The measured (intensity) reflectivity:

$R = s(C_{pp} \cos^2 P + C_{ss} \sin^2 P)$ , where  $P$  is the polarizer angle ( $=0^\circ$  for the p-polarized light,  $=90^\circ$  for the s-polarized light),  $s$  is a scaling coefficient (which should normally be 1, but is introduced only in order to compensate a possible uncertainty of the absolute value of  $R$ ).

Ellipsometric parameters:

$$\tan \psi = \sqrt{\frac{C_{pp}}{C_{ss}}}, \quad \cos \Delta = \frac{C_{ps}}{\sqrt{C_{pp} C_{ss}}},$$

A possible ellipsometry configuration involves a fixed polarizer (set at angle  $P$ ) and a rotating analyzer (without a retarder). In this case the Fourier coefficients of the detector signal as functions of the analyzer angle are:

$$\alpha = \frac{\tan^2 \psi - \tan^2 P}{\tan^2 \psi + \tan^2 P}, \quad \beta = \frac{2 \tan \psi \cos \Delta \tan P}{\tan^2 \psi + \tan^2 P}.$$

For the transmission rays, we have:

$t_{p,s}^{(n)} = \tau(1 - r_{p,s}^2)(r_{p,s}^2 \tau^2)^{n-1}$ , ( $n \geq 1$ ), and then the Equation 4-1, Equation 4-2 and Equation 2-9 are valid with the substitution  $r \rightarrow t$ .

In addition, the model calculates the phase shift (divided by  $2\pi$ ) due to the double pass through the sample:

$\varphi = 2\omega d \sqrt{\varepsilon - \sin^2 \theta} + \varphi_0$ . For the Fabry-Perot interference maxima in transmission,  $\varphi$  should be an integer, for the minima – half integer. Thus, if the Fabry-Perot pattern is well seen, one can determine  $\varphi$  from it.

The description of the model input parameters is given in Table 4-8.

#	Wo	Wp	G
1	DF model for $\varepsilon$	$\theta$ [degrees]	$P$ [degrees]
2	$d$ [ $\text{cm}^{-1}$ ]	$\gamma$	RayNo
3	ReflTran	$\varphi_0$	$s$

Table 4-18 The parameters of the model “Optics of a plate sample”.

RayNo is an integer number, which selects the rays which are detected. If RayNo = 0, then all rays are assumed to equally reach the detector (all  $\alpha_n = 1$ ). If RayNo = 1, 2, etc., then only this ray reaches the detector, while the others miss it ( $\alpha_{n=\text{RayNo}} = 1, \alpha_{n \neq \text{RayNo}} = 0$ ).

ReflTran chooses between reflection and transmission. If ReflTran = 1(0), then all output properties (intensity and ellipsometry) refer to reflection(transmission).

An example of such a model is shown in Figure 4-19. It is assumed that the “Model2” contains  $\varepsilon$ . The angle of incidence is  $60^\circ$  and the polarizer angle is  $45^\circ$ , the sample thickness is 0.001 cm (10 microns), the degree of coherence  $\gamma=0.7$ , the RayNo=0 (all rays are detected), ReflTran is 1 (reflectivity),  $\varphi_0=7$  and the scaling coefficient  $s=1.03$ , .

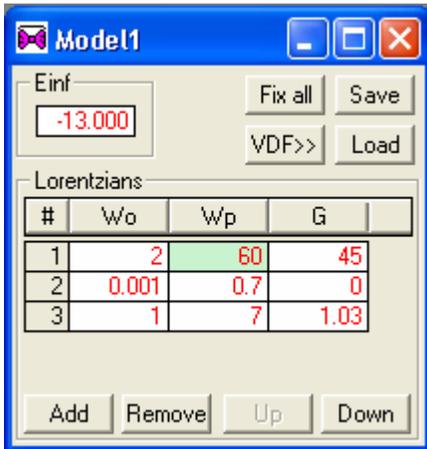


Figure 4-19 An example of the model “Optics of a plate sample”.

Table 4-9 describes the meaning of the output quantities. Note, that the quantity abbreviations have no original meanings.

Quantity abbreviation	Description
“R”	Reflection $R$ , or transmission $T$ , depending on ReflTran
“RPS”	$\varphi$
“E1”	$\psi$ [degrees]
“E2”	$\Delta$ [degrees]
“S1”	$\alpha$
“S2”	$\beta$

Table 4-19 The output quantities of the model “Optics of a plate sample”.

#### 4.7.10. Reflectivity of a Film (Epsilon+Mu) on a Substrate

This model calculates the normal-incidence reflectivity  $R$  of a film of thickness  $d$ , which is characterized by a dielectric function  $\varepsilon$  and a magnetic susceptibility  $\mu$  on a semi-infinite substrate, characterized by a dielectric function  $\varepsilon_s$ . There is a possibility to roughly set the degree of coherence  $\gamma$  between multiply reflected rays in order to suppress the Fabry-Perot interference.

$$R = \gamma R_c + (1 - \gamma) R_i, \text{ where}$$

$$R_c = \left| r_1 + \frac{(1 - r_1^2) r_2 \tau^2}{1 + r_1 r_2 \tau^2} \right|^2 - \text{the reflectivity in the case of full coherence (the amplitudes are added) and}$$

$$R_i = |r_1|^2 + \frac{|(1 - r_1^2) r_2 \tau^2|^2}{1 + |r_1 r_2 \tau^2|^2} - \text{the reflectivity in the case of no coherence (the intensities are added)}$$

$$\text{and } r_1 = \frac{\sqrt{\mu} - \sqrt{\varepsilon}}{\sqrt{\mu} + \sqrt{\varepsilon}}, r_2 = \frac{\sqrt{\varepsilon} - \sqrt{\varepsilon_s \mu}}{\sqrt{\varepsilon} + \sqrt{\varepsilon_s \mu}}, \tau = \exp \left[ i \frac{\omega}{c} \sqrt{\varepsilon \mu} d \right].$$

The description of the model input parameters is given in Table 4-8.

#	Wo	Wp	G
1	DF model for $\varepsilon$	DF model for $\mu$	$d$ [cm <sup>-1</sup> ]
2	DF model for $\varepsilon_s$	$\gamma$	-

Table 4-20 The parameters of the model “Reflectivity of a film (Epsilon+Mu) on a substrate”.

An example of such a model is shown in Figure 4-19. It is assumed that the “Model1” contains  $\epsilon$ , “Model2” contains  $\mu$ , “Model3” contains  $\epsilon_s$ . The sample thickness is 0.012 cm, the degree of coherence  $\gamma=0.3$ .

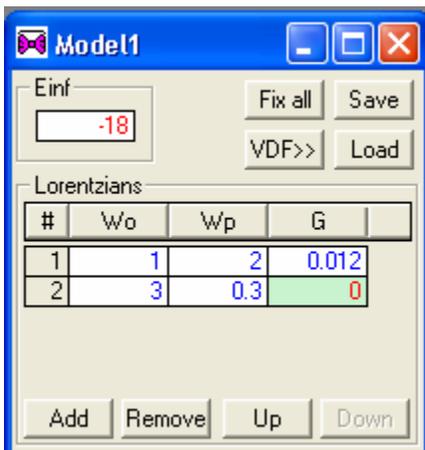


Figure 4-20 An example of the model “Reflectivity of a film (Epsilon+Mu) on a substrate”.

Table 4-9 describes the meaning of the output quantities (there is only one actually).

Quantity abbreviation	Description
“R”	Reflection $R$

Table 4-21 The output quantities of the model “Reflectivity of a film (Epsilon+Mu) on a substrate”.

## 4.8. Dataset manager

The dataset manager is used to load, assign, store and unload experimental datasets. To activate this window one should choose the “Dataset manager” item from the “Window” menu (Figure 3-8). Using this window one can load up to 10 independent datasets at the same time. One can store more of them, using the macro language (see section 4.11.4).

To close the “Dataset Manager” window you can either click the  or  button. Closing this window helps to save the screen space but does not cause unloading of datasets.

### 4.8.1. Loading and unloading datasets

In order to load a dataset to a certain slot, one should click the  button. An extra window will show up, where you can specify the way the data should be loaded to the RefFIT memory. Each dataset is a table, which consists of three columns: X, Y and Yerr. The X is the light frequency (in  $\text{cm}^{-1}$ ), the Y is the measured optical quantity (e.g., reflectivity) and the Yerr is the error bar.

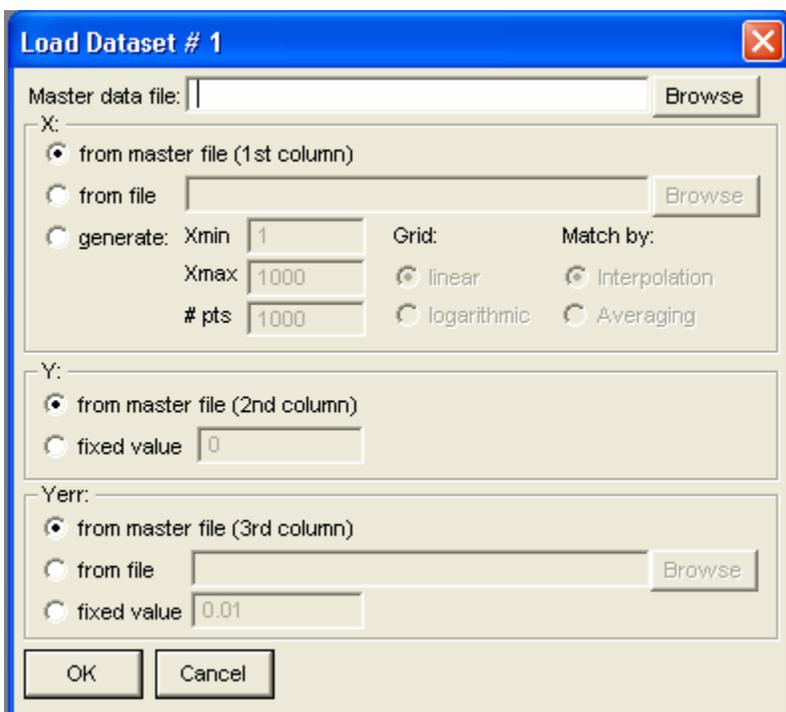


Figure 4-21 The dataset manager window.

One should always specify the master data file. It must contain at least two columns: X and Y. Optionally it may have third column Yerr. The fastest way is to take X, Y and Yerr from the master file. If Yerr is missing, a default value (=0.01) will be put.

The data file should be a plain (ANSI) text file. Do not use special text encodings, like UNICODE. The columns in the data file have to be separated by the space character “ ” or the tabulation character. Do not use commas, semicolons *etc.*

Sometimes, it is convenient to have different frequency points in the dataset. For instance, if you load reflection and transmission file, which are measured with different frequency points, you might still want to have two datasets with the same frequency points. In this case there two options. Firstly, you may provide an extra file with only one column – X, which will serve as a source of frequency points. Secondly, you may generate your own grid of frequency points by specifying the interval in the fields “Xmin” and “Xmax”, the number of points “#pts” and the grid type (linear or logarithmic). For a linear grid all the frequencies are equally spaced. For a logarithmic grid the logarithms of frequencies are equally spaced. The latter grid is useful, if you have more structures in the low-frequency part of your spectrum than in the high-frequency part. The conversion from the original frequency grid to the new one is done by a linear interpolation method.

If the frequency points are generated externally (from a file, or generated) then there are two possibilities to match the Y and Yerr values for the new frequency set with the one in the master data file: by interpolation, or by averaging. You should choose the corresponding option. Interpolation means that Y and Yerr are taken from the linear interpolation between the two closest points in the original (master) dataset. Averaging means that all original points falling into the interval, which belongs to the given new frequency point, are averaged to generate the new values of Y or Yerr. Averaging reduces the noise of the original file.

In most cases the Y-column should be taken from the master file. However, there is a possibility to set Y to a fixed (constant) value.

Ideally, the error bars should be provided in the master file. Proper error bars always improve the fitting reliability. There is an option to load error bars from a separate file, or set it to a fixed (constant) value.

Having chosen the correct loading options click the  button. Once you have loaded the data set you must specify its type. The dataset types should correspond to one of the model output types (see Table 4-2). You can choose the type from the list in the “Quantity” column. By default, the program determines the data type looking at the first letters of the file name. For instance, RefFIT will suggest type “PD” (Penetration depth) for the filename “PD\_T300K.DAT”.

There is an extra possibility to cut the spectral range to a subset that will be used for fitting. To do this you may specify the limits of the “Cutting range”. The corresponding number of frequency points will be highlighted in the field “# points”.

To unload a dataset click the  button. To unload all the datasets at once, click the  button in the lower part of the window.

## 4.9. Graphs

Graphs in RefFIT serve to visualize the model and experimental data and update their changes *in real time*. Each graph is shown in a separate window (Figure 4-22). A new graph can be created by choosing “Graph” in the “Window” menu. Closing the graph window destroys the graph as an object.

One can specify graph properties, such as axis scale, title *etc.*, and graph contents, such as the set of curves and their appearance on the screen.

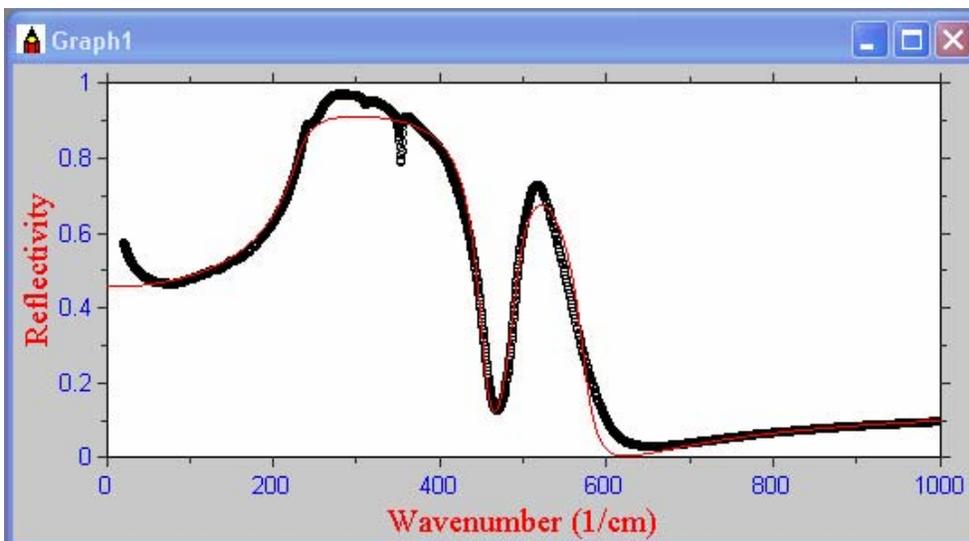


Figure 4-22 The “Graph” window.

### 4.9.1. Graph properties

To set the graph properties, one should double-click on the gray (outer) area of the graph (Figure 4-22). The window “Graph properties” will pop up (Figure 4-23). Here one can change the properties of the “X” and “Y” axes. It is only necessary to set the range. The tick step will be determined automatically according to the graph size. The default value for the “Y” axis is “Reflectivity” which should be changed if other spectra will be plotted (*e.g.*, conductivity) in order to avoid confusions. This, however, does not influence the fitting procedure. One can save a bit of the graph space by not showing axis titles.

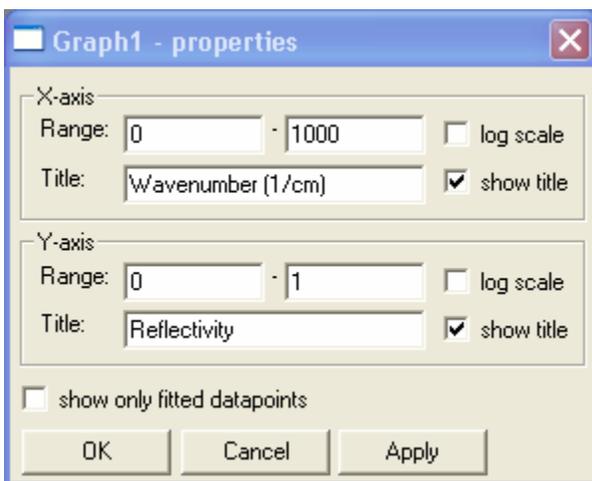


Figure 4-23 The “Graph properties” window

By default, all the loaded dataset points are plotted. However, one can show only the data points which are used for fitting (specified in the “cutting range” in the dataset manager) by checking the “show only fitted data points” option.

### 4.9.2. Graph contents

To set the graph contents, one should double-click on the white (inner) area of the graph (Figure 4-22). In the window “Graph contents” (Figure 4-24) one can select the curves to show in the graph and specify their appearance.

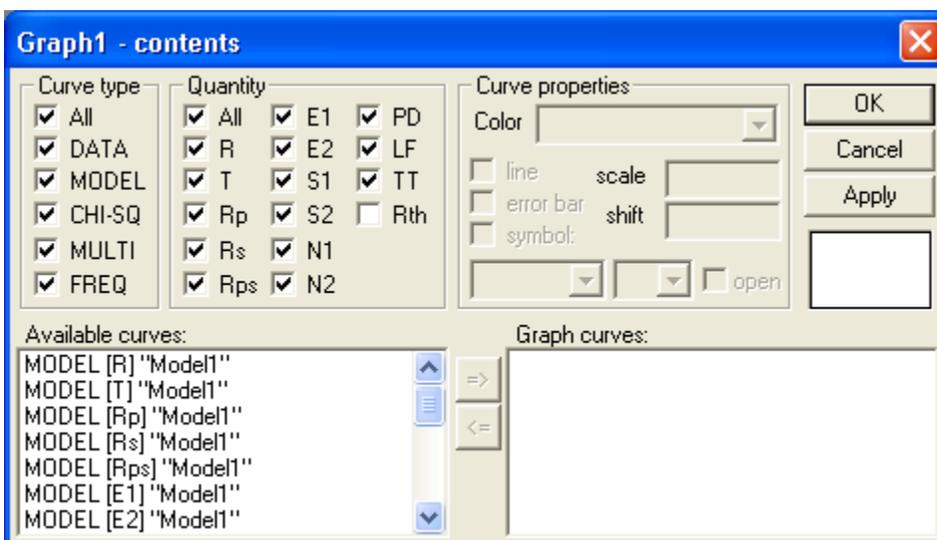


Figure 4-24 The “Graph contents” window.

It is possible to plot five types of curves: “DATA”, “MODEL”, “CHI-SQ”, “MULTI” and “FREQ”.

The curves of the “DATA” type represent experimental spectra, loaded as datasets (see section 4.8). They can be plotted either as symbols of different shape and size, as lines, as error bars, or as a combination of these.

The “MODEL” type corresponds to model output quantities (transmission, conductivity *etc.*). Model curves are always continuous.

A curve of the “CHI-SQ” type shows the square of the difference between experimental and model spectra (chi-square). It is used to estimate the fitting match. It is this value, which is minimized in the fitting procedure (see section 2.1.2).

The “MULTI” type is used to show the contribution of all Lorentz oscillators, included in the model, separately. Obviously, it applies only to the Drude-Lorentz model type and only to the *additive* quantities, such as the outputs “E1”, “E2”, “S1” and “S2” of the “Dielectric function” model.

The “FREQ” is a special curve type, which indicates the transverse frequencies of the Lorentz oscillators in a form of dashed vertical lines.

In order to specify the curves to be plotted one should first check the corresponding buttons in the groups “Curve type” and “Quantity”. The full list of available curves, according to these buttons will be shown in the “Available curves” section. The next step is to select some curves from this list (using the mouse, possibly in combination with the **CTRL** and **SHIFT** keys) and move them to the list “Graph curves” by clicking the  $\Rightarrow$  button. In order to remove curves from the “Graph curves” list one should select them and click the  $\Leftarrow$  button.

The graph curves appearance can be adjusted using the controls from the ‘Curve properties’ part. One should select one curve and change its properties, then switch to another one and repeat the procedure and so on. It is possible to select a group of curves and modify their properties. Each curve can be shifted or/and scaled, if necessary, in order to be better seen on the graph.

To apply the specified graph contents curves one should press button  or . In the latter case the changes will be applied without closing the “Graph Contents” window.

## 4.10. Fit window

The “Fit” window is activated by the “Fit” item from the “Window” menu (Figure 4-25). It is used to set the fitting task, launch/stop the automated fitting routine and monitor the fitting progress. The closing of this window does not change the fitting task (section 4.10.1) and fitting options (section 4.10.2).



Figure 4-25 The “Fit” window.

### 4.10.1. Setting up the fitting task

The posing of the fitting task is essentially the setting the chi-square terms to be minimized (see section 2.1, Equation 2-11, Equation 2-12). Each chi-square term is specified by a pair “Dataset-Model”. By including such a pair you tell RefFIT which model has to be fitted to which dataset.

It is important, that one can select *many chi-square terms simultaneously*, which allows fitting several spectra of different type at the same time. There is a full freedom to specify any combinations of Dataset-Model pairs!

Initially there are no chi-square terms in the fitting task (Figure 4-25). To add them, one should press the  button. In the window “Add ChiSq terms” (Figure 4-26) one can select Available ChiSq terms and click the  (using field “Model” and “Dataset” helps to narrow the list). After that the new terms will show up in the “Fit” window (Figure 4-27).



Figure 4-26 The “Add ChiSq term(s)” window.

To remove chi-square terms one should select them in the list and press the **Remove** button (Figure 4-27). By default, all terms have the same weight  $w$  equal to 1. To change the weight, one should select the corresponding term(s) and type the new weight in the “Weight” field.

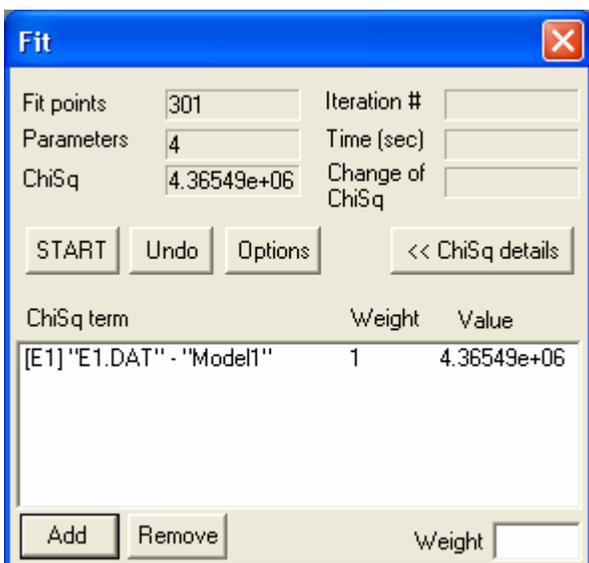


Figure 4-27 The “Fit” window with one chi-square term selected.

The values of the individual chi-square terms and the total chi-square value (considering term weights) are always seen in the “Fit” window. They are updated in real time as the model parameters are changed manually, or automatically.

The number of fit points and parameters is derived on the base of the chi-square terms. Only the points in the cutting range of the Dataset are considered by the fitting routine.

#### 4.10.2. Fitting options

There is a possibility to set up the technical parameters, which regulate the flow of the fitting process. To do this one should click the **Options** button in the “Fit” window (Figure 4-27). The dialog window “Fitting options” will show up (Figure 4-28).

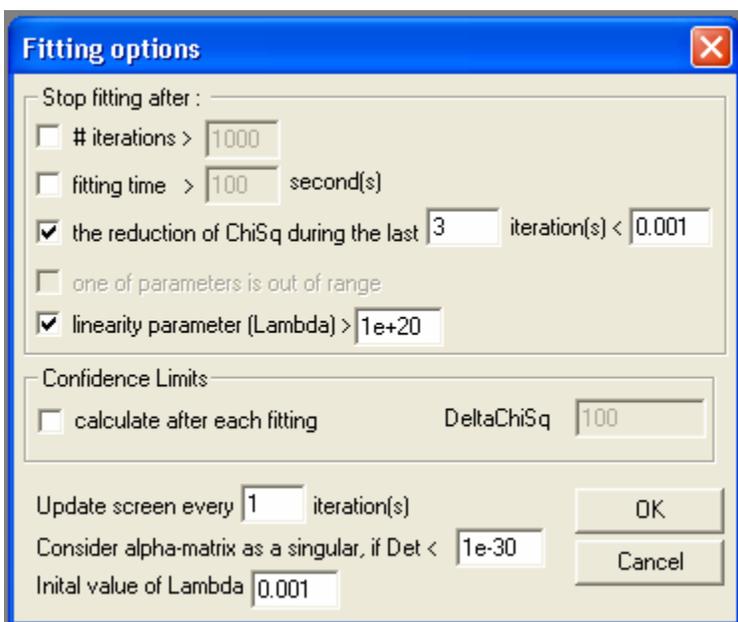


Figure 4-28 The “Fitting options” window.

To use these options requires the knowledge of the Levenberg-Marquardt algorithm, described in section 2.1.1. However, in most cases the default settings work reasonably well.

The first set of parameters (“Stop fitting after”) define the criteria to terminate the fitting process. The following criteria can be applied:

- (i) a certain maximum number of iterations is exceeded;
- (ii) the fitting takes too much time;
- (iii) the process seems to converge, *i.e.*, the total chi-square did not vary significantly during the last few iterations;
- (iv) one of the parameters goes out of range<sup>20</sup>;
- (v) the linearity parameter  $\lambda$  becomes too large.

The group “Confidence limits” deals with the calculation of the parameter confidence limits, or error bars (see section 2.1.3). One can force RefFIT to calculate them after each fitting iteration. This may slightly slow down the fitting process. The value in the “DeltaChiSq” corresponds to  $\delta\chi^2$  in Equation 2-14.

By default, all spectra and parameters are updated on the screen after each iteration. In order to save some time it is possible to let the program to update the screen not so often by changing the field “Update screen every ... iteration(s)”<sup>21</sup>.

At each iteration a linear equation system with the  $\alpha$ -matrix is solved (Equation 2-9). It becomes impossible when the determinant of the matrix is getting too small. This happens, for instance, if one of the active parameters does not affect the value of chi-square. In this case the

<sup>20</sup> This option is not realized yet

<sup>21</sup> The screen update time is usually only a minor fraction of the total computing activity.

iterations stop. One can specify the minimal allowed value of the determinant, which is by default  $10^{-30}$ .

The initial value of  $\lambda$  can also be modified (the default value is 0.001).

### 4.10.3. Starting/stopping fitting process

To start the automated fitting one should click the  button (Figure 4-27). Only a very limited set of actions is allowed when the fit is running (*e.g.*, one can resize the windows). The process can be stopped by the same button, which is now looks as . It may take some time before the program really stops, because it must complete the current iteration.

During the fitting process the following quantities are displayed: the current value of this chi-square, its absolute change compared to the initial value, the number of iterations and elapsed fitting time.

In order to recover the previous parameter values before the last fitting started, one should click the  buttons<sup>22</sup>. It is only possible right after the fitting run.

## 4.11. Macro language

Once you have found a good data fit, you may need to do the same task many times, for instance, to fit the same kind of spectra for different temperatures. Each case, however, involves an appreciable number of actions, such as opening a model, loading data, adding chi-square terms and so forth. Imagine, how many mouse clicks have to be done to fit spectra for, let's say, 10 temperatures! And what about 300 temperatures?!!!

To facilitate the execution of routine operation the macro scripts are indispensable. Another advantage of scripts is the possibility to modify them in order to tackle new problems, which are similar to the ones already solved. Also, a script may serve as a sort of logbook, which helps to recollect, how exactly a particular data analysis has been performed a long time ago.

### 4.11.1. Writing and executing macros

Macro scripts are stored as text files, which can be created and modified using any text editor such as Windows Notepad. Make sure, that the file is saved as a usual, or ANSI, text (and not, for example, as UNICODE text). The convention is (although it is not obligatory) to use the extension 'RFS' to macro files, for instance "MACRO1.RFS".

Once a macro is created, it can be executed at the stage when RefFIT starts. One should call the RefFIT program supplemented with the macro file name, for example "D:\REFFIT\REFFIT.EXE MACRO1.RFS". In this case the executable file "REFFIT.EXE" should be in the directory "D:\REFFIT" and the macro file "MACRO1.RFS" should be in the currently selected directory.

---

<sup>22</sup> this option is not realized yet

There are several ways to launch the RefFIT program with a script name. The most universal one (though not the most handy) is to create a batch file, called, *e.g.*, “MACRO1.BAT”, which contains only one command line “D:\REFFIT\REFFIT.EXE MACRO1.RFS”. It can be done using the same editor program, like Notepad (Figure 4-29). The execution of “MACRO1.BAT” (*e.g.*, by clicking on it in the Windows Explorer) will have the desired effect.

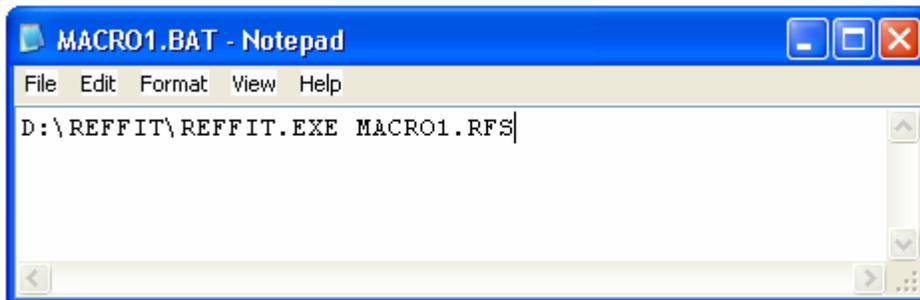


Figure 4-29 The batch file to start macro “MACRO1.RFS”.

Scripts can be called more comfortably by just double-clicking on the icons of their files. For that one should associate the file extension “RFS” with the executable “REFFIT.EXE”. It can be done, for instance, in the standard Windows Explorer program by clicking on a macro file with the right mouse button, selecting item “Open With”→”Choose program” and then specifying the actual path to “REFFIT.EXE”.

#### 4.11.2. Macro syntax

The RefFIT macro syntax is extremely simple. Macro scripts consist of commands, separated by semicolon “;”. It is a good idea (although not imperative) to write each command on a separate line. Commands correspond to actions done by the mouse or the keyboard in the application windows.

The command syntax is as follows: “CommandName(Par1 = Value1, Par2 = Value2, ...);” The CommandName identifies the command. “Par1”, “Par2” *etc.* are the names of parameters; “Value1”, “Value2” *etc.* are their values. The parameters should be separated by commas “,”. The set of parameters depends, of course, on the command. In some cases certain parameters might be omitted. Even if no parameters are set in the command, the command name must be followed by empty brackets “()”. The sequence of parameters in brackets does not matter.

The parameters can be of integer, double or text type. The integer parameter value can be any integer number, *e.g.*, 10. The double parameter value can be any double (floating point) number, *e.g.*, 1.23E-4. The text parameter value should be a text string, enclosed in parentheses “...”, *e.g.*, “CONDUCTIVITY”.

Any command preceded by the “#” character is ignored by the macro interpreter. This can be used to put text comments into the macro, or temporarily ‘comment out’ some commands.

Before a macro is actually executed, it is first translated by the macro interpreter. If the interpreter finds any kind of error, the execution does not start. Unfortunately, at this moment the interpreter is not programmed to make any error report. If a macro does not run, the easiest

way to localize the error is to comment out potentially erroneous commands one by one and try the macro again.

### 4.11.3. General macro commands

*Command name:* **MainWindow**

*Description:* Moves, resizes and renames the main application window.

*Parameters:*

Name	Type	Description	What if omitted
XPos	Integer	Horizontal position of the left side (in screen pixels)	Window is neither moved nor resized
YPos	Integer	Vertical position of the top side (in screen pixels)	Window is neither moved nor resized
Width	Integer	Window width (in screen pixels)	The window is moved but the width is not changed
Height	Integer	Window height (in screen pixels)	The window is moved but the height is not changed
Title	Text	Window title	The window title is not changed

*Command name:* **wait**

*Description:* makes sure that the screen (*i.e.*, parameters and model curves) is fully updated to reflect the last parameter changes and then waits the specified amount of time.

*Note:* this is a ‘technical’ command, which has absolutely no influence on the data and fitting process.

*Parameters:*

Name	Type	Description	What if omitted
Time	Integer	The extra time to wait (in milliseconds)	Assumed to be 0

*Command name:* **Suspend**

*Description:* suspends the macro execution and shows a “Suspend” window (Figure 4-30), which allows a user to choose whether to continue or stop macro execution.

*Note:* It is useful, when the macro works in a cycle and the fitting match has to be visually verified on each step of the cycle.

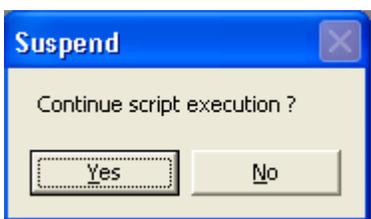


Figure 4-30 The “Suspend” window.

*Parameters:* no parameters

*Command name:* **Exit**

*Description:* terminates the execution of macro without going further.

*Note:* this command is useful during the ‘debugging’ of the macro.

*Parameters:*

Name	Type	Description	What if omitted
QuitApp	Integer	If QuitApp $\neq$ 0, then the application will be terminated also	Assumed to be 0

#### 4.11.4. Dataset macro commands

*Command name:* **DatasetManager**

*Description:* shows the “Dataset Manager” window on the screen and sets its position

*Parameters:*

Name	Type	Description	What if omitted
XPos	Integer	Horizontal position of the left side (in screen pixels)	The position is not changed
YPos	Text	Vertical position of the top side (in screen pixels)	The position is not changed

*Command name:* **LoadDataset**

*Description:* loads a dataset

*Note:* This command has the same effect as manual dataset loading (section 4.8). Even though not more than 10 datasets can be loaded manually in the dataset manager, up to 100 datasets can be loaded with this command.

*Parameters:*

Name	Type	Description	What if omitted
DatasetNo	Integer	Dataset number, corresponding to the slot number in the Dataset Manager. Starts from 1.	Assumed to be 1

Quantity	Text	Abbreviation of the quantity, which is ascribed to this dataset, e.g. “R”, “S1” etc. (see Table 4-2);	Assumed to be “R” ( <i>i.e.</i> reflectivity)
MasterFile	Text	The name of the master file (including file path, if the file is not in the same directory, as the macro file), <i>i.e.</i> “R.DAT”, or “C:\MYDATA\E2.DAT”, or “..\SPECTRA\N1.DAT”	Does not load any dataset
XMethod	Integer	Specifies the algorithm used to generate column X in the database. =1: X is taken from the master file (1 <sup>st</sup> column) =2: X is taken from the extra file, according to parameter XFile =3: X is generated, according to parameters Xmin, Xmax, XPts and XGrid	Assumed to be 1
XFile	Text	The name of the extra file, used to generate column X (including file path, if the file is not in the same directory, as the macro file). Only applies when XMethod=2.	No action is taken (if XMethod=2).
XMin	Double	The minimum value of the generated X Only applies, when XMethod=3.	Assumed to be 1.0
XMax	Double	The maximum generated X value. Only applies when XMethod=3.	Assumed to be 1000.0
XPts	Integer	The number of the generated X values. Only applies when XMethod=3.	Assumed to be 1000
XGrid	Integer	Specifies the grid type of the generated X values =1: linear grid (equally spaced values) =2: logarithmic grid	Assumed to be 1
Match	Integer	Specifies the way to match the values of Y and Yerr to the ones from the original dataset =1: interpolation =2: averaging	Assumed to be 1
YMethod	Integer	Specifies the algorithm used to generate column Y in the dataset. =1: Y is taken from the master file (2 <sup>nd</sup> column) =2: Y is set to a constant value, according to parameter YVal	Assumed to be 1
YVal	Double	Specifies the constant value to be set to column Y. Only applies when	Assumed to be 0.0

		YMethod=2.	
YErrMethod	Integer	Specifies the algorithm used to generate column YErr in the dataset. =1: YErr is taken from the master file (3 <sup>rd</sup> column) =2: YErr is taken from the extra file, according to parameter YErrFile =3: YErr is set to a constant value, according to parameter YErrVal	Assumed to be 0.01
YErrFile	Text	The name of the extra file, used to generate column YErr (including file path, if the file is not in the same directory, as macro file). Only applies when YErrMethod=2.	Sets YErr to be 0.01 (constant value)
YErrVal	Double	Specifies the constant value to be set to column YErr. Only applies when YErrMethod=3.	Assumed to be 0.01

*Command name:* **UnloadDataset**

*Description:* unloads a dataset.

*Parameters:*

Name	Type	Description	What if omitted
DatasetNo	Integer	Dataset number, which corresponds to the slot number in the Dataset Manager. Starts from 1.	Assumed to be 1

*Command name:* **CutDataset**

*Description:* specifies the cutting range of the dataset, which is considered by the fitting routine.

*Parameters:*

Name	Type	Description	What if omitted
DatasetNo	Integer	Dataset number, which corresponds to the slot number in the Dataset Manager. Starts from 1.	Assumed to be 1
XMin	Double	The minimum value of the cutting range	Xmin is not changed
XMax	Double	The maximum value of the cutting range	Xmax is not changed

#### 4.11.5. Model macro commands

*Command name:* **NewModel**

*Description:* Creates a new model; moves and resizes the model window.

*Note:* when a new model is created, it is *automatically* ascribed an identity number, which starts from 1 and increments for every new model. It can be seen in the window title, e.g., model #5 has title “Model5”. This number is used by other commands to identify this model.

*Parameters:*

Name	Type	Description	What if omitted
XPos	Integer	Horizontal position of the left side (in screen pixels)	The default value is assumed
YPos	Integer	Vertical position of the top side (in screen pixels)	The default value is assumed
Width	Integer	Window width (in screen pixels)	The default value is assumed
Height	Integer	Window height (in screen pixels)	The default value is assumed
VarDielFuncSeen	Integer	= 0: the control group “Variational Diel. Function” is not seen = 1: it is seen	0 is assumed

*Command name:* **SaveModel**

*Description:* Saves the specified model to a file.

*Note:* This command is equivalent to using the **F2** button in the “Model” window (section 4.5).

*Parameters:*

Name	Type	Description	What if omitted
ModelNo	Integer	The identity number of the model to be saved	Assumed to be 1
File	Text	The name of the model file (including file path, if the file is not in the same directory, as macro file), <i>i.e.</i> “MODEL1.RFM”, or “C:\MYDATA\MODEL1.RFM”, or “..\MODELS\MODEL2.RFM”	No action is taken

*Command name:* **LoadModel**

*Description:* Loads a model from a file to the specified (already created) model.

*Note:* This command has the same effect as using **F3** button in the window ‘Model’ (section 4.5). It does NOT create a new model in RefFIT, so the model with the specified identity number has to be already created with the NewModel command.

*Parameters:*

Name	Type	Description	What if omitted
ModelNo	Integer	Model identity number	Assumed to be 1

File	Text	The name of the model file (including file path, if the file is not in the same directory, as macro file), <i>i.e.</i> “MODEL1.RFM”, or “C:\MYDATA\MODEL1.RFM”, or “..\MODELS\MODEL2.RFM”	No action is taken
------	------	---	--------------------

*Command name:* **SetModelParam**

*Description:* Sets the value and the fitting activity of the specified model parameter

*Note:* this command affects only one model parameter

*Parameters:*

Name	Type	Description	What if omitted
ModelNo	Integer	Model identity number	No action is taken
ParamName	Text	The name of the parameter. “Einf” - epsilon at infinity; “Wo” – transverse frequency; “Wp” – plasma frequency; “G” – linewidth.	No action is taken
LorNo	Integer	The number of the Lorentzian (starting from 1). It is ignored, if ParamName=“Einf”	No action is taken in case ParamName = “Wo”, “Wp” or “G”
Value	Double	The new value	The current value of the parameter is unchanged
FitActive	Integer	=0 – not active in fitting (fixed) =1 – active in fitting (free)	The current fitting activity of the parameter is unchanged
CoupledTo	Integer	The coupling variable =0 – no coupling	The coupling variable is unchanged

*Command name:* **CopyModelParam**

*Description:* Makes a (destination) parameter to be the exact copy of another (source) parameter.

*Note:* it copies ALL characteristics of the parameter, including minimum and maximum limits (not only the value and the fitting activity).

*Parameters:*

Name	Type	Description	What if omitted
DestModelNo	Integer	Identity number of the destination model	No action is taken
DestParamName	Text	The name of the destination parameter. “Einf” - epsilon at infinity; “Wo” – transverse frequency;	No action is taken

		“Wp” – plasma frequency; “G” – linewidth.	
DestLorNo	Integer	The number of the destination Lorentian (starting from 1). It is ignored, if DestParamName=“Einf”	No action is taken in case DestParamName = ”Wo”, “Wp” or “G”
SourceModelNo	Integer	Identity number of the source model	No action is taken
SourceParamName	Text	The name of the source parameter. “Einf” - epsilon at infinity; “Wo” – transverse frequency; “Wp” – plasma frequency; “G” – linewidth.	No action is taken
SourceLorNo	Integer	The number of the source Lorentian (starting from 1). It is ignored, if SourceParamName=“Einf”	No action is taken in case SourceParamName = ”Wo”, “Wp” or “G”

**Command name: FixAllModelParams**

*Description:* Makes *all* the parameters of the specified model not active in fitting (fixed).

*Note:* it is nice to use this command when one has to adjust a model using one set of data and then to use it for other purposes without fitting. For example, when extracting optical properties of a thin film, it is common to obtain first the dielectric function of the substrate from an independent measurement and then to use it as known (fixed) while analyzing spectra measured on the ‘film-substrate’ system.

*Parameters:*

Name	Type	Description	What if omitted
ModelNo	Integer	Model identity number	No action is taken

**Command name: VarDielFunc**

*Description:* Controls the variational dielectric functions (VDFs) (see sections 2.2.4, 2.2.5 and 4.6.3). It can initialize the anchor mesh on the base of up to three source datasets, and/or change the properties of the VDF.

*Note:* If none of the three source datasets is specified, then the mesh is not initialized (only the VDF properties are affected).

*Parameters:*

Name	Type	Description	What if omitted
ModelNo	Integer	Identity number of the Model, which hosts the VDF	No action is taken
Source1DatasetNo	Integer	Dataset number, which is used as a 1 <sup>st</sup> source of anchor points	The 1 <sup>st</sup> source of anchor points is not used

Source1Every2nd	Integer	= 1, if every second point from the 1 <sup>st</sup> source should be used = 0, if all points from the 1 <sup>st</sup> source should be used	Assumed to be 0
Source2DatasetNo	Integer	Dataset number, which is used as a 2 <sup>nd</sup> source of anchor points	The 2 <sup>nd</sup> source of anchor points is not used.
Source2Every2nd	Integer	= 1, if every second point from the 2 <sup>nd</sup> source should be used = 0, if all points from the 2 <sup>nd</sup> source should be used	Assumed to be 0.
Source3DatasetNo	Integer	Dataset number, which is used as a 3 <sup>rd</sup> source of anchor points	The 3 <sup>rd</sup> source of anchor points is not used.
Source3Every2nd	Integer	= 1, if every second point from the 3 <sup>rd</sup> source should be used = 0, if all points from the 3 <sup>rd</sup> source should be used	Assumed to be 0.
KK	Integer	= 1, KK-constrained = 0, not KK-constrained	Does not change the KK regime, if no sources of anchor points are specified.  Assumed to be 1, if sources of anchor points are specified.
On	Integer	= 1, On (activated) = 0, Off (deactivated)	Does not change the On/off status, if no sources of anchor points are specified.  Assumed to be 1, if sources of anchor points are specified..
FitActive	Integer	= 1, active in fit (adjustable) = 0, not active in fit (fixed)	Does not change the 'Active in fit' status, if no sources of anchor points are specified.  Assumed to be 1, if sources of anchor points are specified.

Command name: **FreeShape**

*Description:* Emulates the pressing of buttons **F4**, **F5**, **F6** and **F7** in the model window. Note: these buttons are used to set up the so called variational, or ‘free-shape’, dielectric function (see section 4.6.3).

*Note:* this command is obsolete, and is retained for the compatibility with the previously written macros only. It is strongly advised to use a more powerful command VarDielFunc.

*Parameters:*

Name	Type	Description	What if omitted
ModelNo	Integer	Model identity number	No action is taken
Key	Text	Specifies the button to be emulated: “F4”, “F5”, “F6” or “F7”	No action is taken

#### 4.11.6. Graph macro commands

*Command name:* **NewGraph**

*Description:* Creates a new graph; moves and resizes the graph window.

*Note:* when a new graph is created, it is *automatically* ascribed an identity number, which starts from 1 and increments for every new graph. It can be seen in the window title, e.g. graph #5 has title “Graph5”. This number is used by other commands to identify this graph.

*Parameters:*

Name	Type	Description	What if omitted
XPos	Integer	Horizontal position of the left side (in screen pixels)	The default value is assumed
YPos	Integer	Vertical position of the top side (in screen pixels)	The default value is assumed
Width	Integer	Window width (in screen pixels)	The default value is assumed
Height	Integer	Window height (in screen pixels)	The default value is assumed

*Command name:* **GraphProperties**

*Description:* Sets the graph properties.

*Parameters:*

Name	Type	Description	What if omitted
GraphNo	Integer	Graph identity number	No action is taken
XMin	Double	Minimal X value	XMin is unchanged
XMax	Double	Maximal X value	XMax is unchanged
XLog	Integer	=0: linear X scale =1: logarithmic X scale	XLog is unchanged
XTitle	Text	X title	XTitle is unchanged
YMin	Double	Minimal Y value	YMin is unchanged

YMax	Double	Maximal Y value	YMax is unchanged
YLog	Integer	=0: linear Y scale =1: logarithmic Y scale	YLog is unchanged
YTitle	Text	Y title	YTitle is unchanged
ShowFitted	Integer	=0: show all datapoints =1: show fitted datapoints only	ShowFitted is unchanged (=0 by default)

*Command name:* **AddDataCurve**

*Description:* Adds a data curve to the graph.

*Parameters:*

Name	Type	Description	What if omitted
GraphNo	Integer	Graph identity number	No action is taken
DatasetNo	Integer	Number of the dataset to be plotted	No action is taken
nColor	Integer	The curve color =0: AUTO (automatic incrementing) =1: Black =2: Red =3: Green =4: Blue =5: Cyan =6: Magenta =7: Yellow =8: Gray =9: Grass =10: Sky =11: Orange =12: Chalk =13: Brown =14: Sea =15: Gold =16: Khaki	Assumed to be 0
ShowLine	Integer	=0: not to show line =1: to show line	Assumed to be 1
ShowSymbol	Integer	=0: not to show symbol =1: to show symbol	Assumed to be 1
ShowError	Integer	=0: not to show error bar =1: to show error bar	Assumed to be 1
nSymbolSize	Integer	Symbol size (only for ShowSymbol =1 ) =0: 1 pixel =1: 3 pixels =2: 5 pixels =3: 7 pixels	Assumed to be 2

		=4: 9 pixels =5: 11 pixels =6: 13 pixels =7: 15 pixels	
nSymbolShape	Integer	Symbol shape (only for ShowSymbol =1 ) =0: square =1: circle =2: up triangle =3: down triangle =4: diamond	Assumed to be 0
SymbolOpen	Integer	(only for ShowSymbol =1 ) =0: closed symbol =1: open symbol	Assumed to be 0
Scale	Double	Curve scale	Assumed to be 1.0 (not scaled)
Shift	Double	Curve shift	Assumed to be 0.0 (not shifted)

*Command name:* **AddModelCurve**

*Description:* Adds a model curve to the graph.

*Parameters:*

Name	Type	Description	What if omitted
GraphNo	Integer	Graph identity number	No action is taken
ModelNo	Integer	Number of the model to be plotted	No action is taken
Quantity	Text	Abbreviation of the quantity to be plotted, e.g. "R", "S1" <i>etc.</i> (see Table 4-2);	Assumed to be "R" ( <i>i.e.</i> reflectivity)
nColor	Integer	The curve color (the same as for command AddDataCurve)	Assumed to be 0
Scale	Double	Curve scale	Assumed to be 1.0 (not scaled)
Shift	Double	Curve shift	Assumed to be 0.0 (not shifted)

*Command name:* **DeleteDataCurve**

*Description:* Deletes data curve from the graph.

*Parameters:*

Name	Type	Description	What if omitted
GraphNo	Integer	Graph identity number	No action is taken
DatasetNo	Integer	Number of the dataset, which gives the	No action is taken

		curve to be deleted	
--	--	---------------------	--

*Command name:* **DeleteModelCurve**

*Description:* Deletes model curve from the graph.

*Parameters:*

Name	Type	Description	What if omitted
GraphNo	Integer	Graph identity number	No action is taken
ModelNo	Integer	Number of the model, which gives the curve to be deleted	No action is taken
Quantity	Text	Abbreviation of the quantity, which corresponds to the curve to be deleted	No action is taken

*Command name:* **ExportModelCurve**

*Description:* exports the spectrum of the specified model quantity to a file

*Parameters:*

Name	Type	Description	What if omitted
File	Text	The name of the file (including file path, if the file is not in the same directory, as macro file), <i>i.e.</i> "E1.DAT", or "C:\MYDATA\E1.DAT", or "..\EPS\E2.DAT"	No action is taken
ModelNo	Integer	The model number	No action is taken
Quantity	Text	Abbreviation of the quantity to be exported	No action is taken
XMin	Double	Minimal X (frequency) value	No action is taken
XMax	Double	Maximal X value	No action is taken
XPts	Number	Number of points to be exported	No action is taken
XGrid	Number	=1: linear grid =2: logarithmic grid	Assumed to be 1

#### 4.11.7. Experimental parameters macro commands

*Command name:* **ExpParams**

*Description:* shows the "Experimental parameters" window on the screen and sets its position

*Parameters:*

Name	Type	Description	What if omitted
XPos	Integer	Horizontal position of the left side (in screen pixels)	The position is not changed
YPos	Text	Vertical position of the top side (in	The position is not

		screen pixels)	changed
--	--	----------------	---------

*Command name:* **SetExpParam**

*Description:* Sets the value and the fitting activity of the specified experimental parameter

*Parameters:*

Name	Type	Description	What if omitted
ParamName	Text	The name of the parameter. The possible names are: "Sample thickness" "Thickness spread" "Angle of incidence" "Angle spread" "Scaling factor of R" "Scaling factor of T"	No action is taken
Value	Double	The new value	The current value of the parameter is unchanged
FitActive	Integer	=0 – not active in fitting (fixed) =1 – active in fitting (free)	The current fitting activity of the parameter is unchanged
CoupledTo	Integer	The coupling variable =0 – no coupling	The coupling variable is unchanged

#### 4.11.8. Fitting macro commands

*Command name:* **WindowFit**

*Description:* shows the "Fit" window on the screen and sets its position.

*Note:* this command is not necessary to do a fit.

*Parameters:*

Name	Type	Description	What if omitted
XPos	Integer	Horizontal position of the left side (in screen pixels)	The position is not changed
YPos	Text	Vertical position of the top side (in screen pixels)	The position is not changed

*Command name:* **AddChiSqTerm**

*Description:* adds the specified chi-square term to the fitting task

*Note:* the chi-square term is defined by the pair Dataset-Model. The right quantity should be ascribed to the dataset when loading!

*Parameters:*

Name	Type	Description	What if omitted
------	------	-------------	-----------------

DatasetNo	Integer	Number of the dataset	No action is taken
ModelNo	Integer	Number of the model	No action is taken
Weight	Double	The weight of the term	Assumed to be 1.0

*Command name:* **DeleteChiSqTerm**

*Description:* delete the specified chi-square term from the fitting task

*Note:* the chi-square term is defined by the pair dataset-model.

*Parameters:*

Name	Type	Description	What if omitted
DatasetNo	Integer	Number of the dataset	No action is taken
ModelNo	Integer	Number of the model	No action is taken

*Command name:* **Fit**

*Description:* starts the fitting process

*Note:* there is no command to stop the fitting process. It will be stopped on the base of the termination criteria (see section 4.10.2).

*Parameters:*

Name	Type	Description	What if omitted
NumIters	Integer	The maximum number of iterations that will be performed	No limit on the number of iterations is imposed

#### 4.11.9. Loop macro commands

The two loop commands allow to perform calculations in a cycle, for instance, to do the same data fitting for several temperature (or magnetic field) values. Command **BeginLoop** marks the beginning of the cycle; command **EndLoop** marks the end of the cycle. They always come in pairs. All commands enclosed between **BeginLoop** and **EndLoop** are executed in a cycle.

The command **BeginLoop** has one text parameter “LoopFile”, which contains the name (with a path, if necessary) of the so-called loop file. The loop file instructs RefFIT on how to run the cycle. It is a text file, which has the same number of lines, as the desired number of cycle steps. Each line contains one or more words separated by space. The first word is the first cycle variable, the second word is the second variable and so forth.

The cycle variables are used to modify parameters of the commands in the cycle. If a text parameter contains combination “%1”, then it will be substituted by the first cycle variable. The combination “%2” is substituted by the second cycle variable *etc.* For instance, if the first cycle variable is “250”, then text string “C:\MYDATA\REFL\_T%1.DAT” will become “C:\MYDATA\REFL\_T250.DAT”. Such a substitution, for example, in the File parameter of the LoadDataset command will cause the loading of a different dataset on each cycle step.

One can also use combinations %1, %2 *etc.* (without parentheses) to set values of integer and double parameters. For instance, in a command `SetModelParam(ModelNo = %1, ParamName = "Wp", LorNo = 1, Value = %2)` the values of parameters `ModelNo` and `Value` are set by the cycle variables.

---

## 5. Bibliography

1. *OPTPAL* as well as many other useful FORTRAN optical programs by Dirk van der Marel can be downloaded from [http://optics.unige.ch/vdm/marel\\_files/fortran\\_codes.html](http://optics.unige.ch/vdm/marel_files/fortran_codes.html).
2. *Principles of optics*, M.Born, E.Wolf, Pergamon Press (1968).
3. *Numerical recipes*, W.H.Press, B.P.Flannery, S.A.Teukolsky and W.T.Vetterling, Cambridge Univ. Press (1986).
4. *Kramers-Kronig-constrained variational analysis of optical data*, A.B.Kuzmenko, Rev. Sci. Instrum. **76**, 083108 (2005).
5. J.W. van der Eb, A.B.Kuzmenko and D. Van der Marel, Phys. Rev. Lett. **86**, 3407 (2001).
6. F.C.Jahoda, Phys.Rev. **107**, 1261 (1957).
7. H.Somal, PhD Thesis, University of Groningen (1999).
8. A.B.Kuzmenko, N.Tombros, H.J.A.Molegraaf, M.Grüninger, D. van der Marel and S.Uchida, Phys. Rev. Lett. **91**, 037004 (2003).
9. F.P.Mena, to be published.
10. A.B. Kuzmenko, D. van der Marel, P. J. M. van Bentum, E. A. Tishchenko, C. C.Presura, and A. A. Bush, Phys. Rev. B, **63**, 094303 (2001).
11. D. van der Marel, Phys. Rev. B **60**, R765 (1999).
12. W.Zimmermann, E.H.Brandt, M.Bauer, E.Seider and L.Genzel, Physica C **183**, 99 (1991).
13. C.C.Homes, T.Timusk, D.A.Bonn, R.Liang and W.Hardy, Physica C **254**, 265 (1995).
14. F.Gervais and B.Piriou, J. Phys.C: Solid State Phys. **7**, 2374 (1974).
15. O.S.Heavens, "Thin Film Physics", Ed. Methuen, London (1970).
16. O.V.Dolgov and S.V.Shulga, J. Supercond. **8**, 611 (1995).