Lab Programs

Optical Properties

Department of Physics, University of Florida, Gainesville, FL 32611-8440, USA

Gainesville, 9 April 2020

TABLE OF CONTENTS

A. Introduction

This document describes the data analysis programs used in our lab. Much of what is here is on their help screens,* but there are some additional notes found here.

This manual is for the Windows versions of the programs. (Earlier versions were written for "DOS," i. e., 16-bit, EGA graphics. Beginning about the time of Windows 98 SP2 and Windows NT, both the Windows operating system and computer hardware became less and less compatible with the older graphics modes. So the programs have been ported over to Windows. They are not true Windows applications, though. They lack all of the message boxes, drop-down menus, radio buttons, tool tips, animated paper clips, etc. that we've come to know and love.)

The ones that do graphics are compiled as "QuickWin" applications. They show a white screen and make 16 color plots. Some use the mouse. Others, text only, are "Windows console" programs. They are intended to be run from a command window (cmd.exe, 4nt.exe, Take Command, etc.) but can be run from Windows Explorer.

1. Installation and use

Put the programs in a directory on your path. I use `D:\datan\`. If the directory you choose is not on the path, add it. (Google "path windows `<ver>`," where `<ver>` is xp or vista or 10 or ... for instructions on how to set the path.)

Run the programs from a command window. Nowadays, the command window is CMD.EXE. I suggest adding a desktop link to this program. A better command processor is available from JP Software (`http://www.jpsoft.com/`). This program started as 4DOS for use in the 16 bit DOS world. The 32 bit version (Win98/NT/XP) used to be called 4NT. The present version is Take Command. A version called TCC/LE is free to all. Versions with even more features are available at a modest price.

You can add links on the desktop to the data analysis programs as well, and most will accept drag/drop from Windows Explorer.

2. List of "QuickWin" programs

These programs make plots of the data during their operation.

2c.exe	ema.exe	kk.exe	ql.exe
avg.exe	fft.exe	mav.exe	qp.exe
bits.exe	ftgram.exe	mend.exe	ranu.exe
bridge.exe	fts.exe	op.exe	rnm.exe
cmp.exe	gtf.exe	pfit.exe	tcalc.exe
dfc.exe	ifft.exe	pik.exe	tff.exe
dff.exe	inject.exe	pik2.exe	
dlcalc.exe	interp.exe	q.exe	

---

* Type `<prog>` /h or answer ? to the first prompt.

3. List of "Windows console" programs

These programs do text-only input and output.

addtime.exe	gtfc.exe	prm2par.exe	t2approx.exe
aph2ri.exe	interpc.exe	rdrude.exe	t2k.exe
bins.exe	j2w.exe	ri2aph.exe	w2l.exe
cmpc.exe	kkc.exe	rph2e.exe	xro.exe
deriv.exe	l2rph.exe	rph2nk.exe	xy2y.exe
e2e.exe	nk2rph.exe	rph2s.exe	xy2yx.exe
e2rph.exe	opc.exe	s2rph.exe	
getdat.exe	pinkify.exe	t2a.exe	

z2w.exe

4. Obsolete programs

Some programs are no longer maintained, either because they have been supplanted by newer or better programs, or because they just are not used any more. Examples in the first category are DLfit.exe (Drude-Lorentz fit), MFLfit.exe, KEfit.exe, and TLfit.exe, which made fits of optical data to the Drude-Lorentz model, the Marginal Fermi Liquid mode, etc. The functionality of all of these (plus some more) is included in dff.exe (Dielectric Function Fit). SaFit and PrmAve are examples of the second category.

Here are the depreciated or obsolete programs, with the replacement in parentheses:

dlcalc.exe (dfc)	kkold.exe (kk)	safit.exe (unused)
dlfit.exe (dff)	kknohi.exe (none)	sel-ftran.exe (fft)
elphit.exe (dff)	mflfit.exe (dff)	threefit.exe (unused)
flmfit.exe (tff)	prmave.exe (unused)	tlfit.exe (dff)
ftran.exe (fft)	relfit.exe (unused)	
kefit.exe (dff)	reseq.exe (e2e)	

5. Common features

The programs have several things in common. All have switches that control their behavior. You can pass file names and switches on the command line or have the program prompt you for them. Filenames should be separated by spaces. Switches may be introduced with '/' or '-', have optional spaces between them, and may be in upper or lower case. Thus, for example, you can say either `qp f1006.s1 f1100.s1 /xlog /a` or `QP f1006.s1 f1100.s1 -XLOG-A` to plot the files named `f1006.s1` and `f1100.s1` with autoscaling and with the x-axis on a log scale.

The programs can handle long file names. File names with a space in them need to be delimited with ". Thus, `Data.300k.first.refl` is OK, and the programs will take "`refl`' as the extension (possibly to be replaced in the output file by "`RPH`" or "`SMO`"). `"Data 300k first.refl"` is also fine, as long as the quotes are used.

Case in names is preserved. (Earlier versions changed them to uppercase.)

The data can be in multiple columns even if only two (x, y values) are to be used. You will be prompted for the column number for x and y. These can be passed on the command line as well, by saying either `/Ci,j` or `/COLi,j` to read columns i and j.

The programs have a help screen. It is displayed if you type the `/h` or `/?` switch (e. g., `pik /?`), or by saying `?` to the first prompt.

Another useful switch is `/q` for "quick." "Quick" bypasses most of the questions, using instead default values. Some of the programs (depending on the whim of the programmer) default to "quick" mode; ordinary mode is regained by using the "verbose" switch, `/v`. A few have three states, "quick" == `/q`, "usual" (no switch), and "verbose" == `/v`. The help screen lists the switches that change how many questions are asked. If you say `/v` when that is not an option you will get a message `?-<program>-W-Ignoring unknown command! /V`.

The programs can be divided into data analysis programs, data fitting programs, spectral analysis programs, and utilities for converting data files. They are briefly described on the following pages.

B. Data analysis programs

These programs manipulate data from our spectrometers. They merge, average, smooth, KK transform, and carry out calculations on the data.

1. AVG

AVG—AVeraGe—averages two or more data files. It reads the files and writes out their average. The default extension is `AVG`.

$$Y_{out} = \frac{1}{N} \sum_{n=1}^{N} Y_n.$$

Files are entered on the command line *or* in response to the "Names of input data files?" prompt:

```
Data1  Data2  ... DataN
```

NB: all on a single line! Switches: `/h`, `/q`, `/v`, and

`/x or /y`	— rescale data, with prompt for scaling factors
`/y`A or `/y`A, B	— rescale data as $y \Leftarrow y \times A + B$.
`/x`A or `/x`A, B	— rescale data as $x \Leftarrow x \times A + B$.
`/Ci,j`	— read Cols i, j.
`/O`	— Omit duplicate X values (default).
`/D`	— include Duplicate x values. (Was `/E`!)
`/RPH`	— data are in an `RPH` file.

2. Bridge

Bridge contains the bridging routines from KK and allows you to see the results of all the possible choices there. You give Bridge the names of two file which do not overlap in frequency (or x values) and Bridge will calculate a power law or cubic spline function that fills the gap. It can compute power laws in frequency or wavelength (1/frequency). Specifically, it will use one of 3 methods of bridging the gap between the two data regions:

1. A power law in ω: $Y = A + B\omega + C\omega^2 + D\omega^3 + \ldots$ (Say `W#` at the prompt.)
2. A power law in $1/\omega$: $Y = A + B/\omega + C/\omega^2 + D/\omega^3 + \ldots$ (Say `L#` at the prompt.)
3. A cubic spline can also be used. (Not recommended, but sometimes it works.)

The power series end point runs from one $(A + B\omega)$ to nine $(A + \ldots + J\omega^9)$.

Bridge first shows the bridge function for all the power series in ω then for all the series in $1/\omega$ and finally the cubic spline. It then asks you which you prefer and uses that preference to bridge and combine the two data sets.

Switches: `/h`, `/q`, `/v`, and

`/F<file>`	— Use `<file>` for high freq extrapolation
`/BW<num>`	— Bridge with $Y = A + B\omega + C\omega^2 + \ldots$ (up to `num`).
`/BL<num>`	— Bridge with $Y = A + B/\omega + C/\omega^2 + \ldots$ (up to `num`).
`/BS`	— Bridge with cubic spline.
`/W or /G`	— Widen Gap using mouse to select start and end of new gap.
`/xA or /xA, B`	— plot range is 0–A or B–A.
`/yA or /yA, B`	— plot range is 0–A or B–A.
`/Ci,j`	— read Cols i, j.

3. CMP and CMPc

CMP—CoMPute—is a program to "compute" or "compare" data files: It will calculate sum, difference, product, or ratio; it also will scale either the x or y-data. Finally it will calculate the inverse of the y-data, or subtract it from unity.

CMPc is a "console" version. It is CMP without any graphics.

CMP can take the actions in the following list. (The first shown is the answer you give to the action prompt and the second the command-line switch.)

+:	Add y-values of the two files.	(/+)
-:	Subtract y-values of two files.	(/-)
*:	Multiply y-values of two files.	(/*)
/:	Divide y-values of two files.	(//)
&:	Concatenate two files, making a three-column file x, y_1, y_2.	(/&)
INT:	Integrate one file.	(/S)
SQR:	Take square root of one file, $y_o = \sqrt{y_i}$.	(/R)
1-:	Subtract one file from 1, $y_o = 1 - y_i$.	(/U)
1/:	Invert one file, $y_o = 1/y_i$.	(/I)
LOG:	Take log of one file, $y_o = \log_{10}(y_i)$.	(/L)
10u:	Exponentiate one file, $y_o = 10^{y_i}$.	(/E)
XSQ:	Raise x values to a power, $x_o = x_i^n$.	(/X**n)
YSQ:	Raise y values to a power, $y_o = y_i^n$.	(/Y**n)
ABS:	Calculate absolute value of a single file, $y_o = |y_i|$.	(/ABS or /[)
CLP:	Clip y in a single file, $A \leq y_o \leq B$.	(/CLA,B)
SCA:	Scale y in a single file, $y_o A y_i + B$.	(/SCA)

The data are produced as follows:

Y_1 <action> $Y_2 \Rightarrow Y_o$

Y_1 = file 1; Y_2 = file 2. <action> +, -, /, * etc.

or

<action> $Y_{data} \Rightarrow Y_{out}$

Y_{data} = sample; <action> = 1.0 -, 1.0/, rescale, etc.

Files entered on the command line in the sequence: `data, ref, out` or `data, out`. CMP will then do its thing with `data` and (if needed) `ref` and produce `out`. If not enough file names are entered, you will be prompted for the ones that are needed.

If only one file name is entered and if one or both of the /XA, B or /YA, B switches have been entered, CMP will assume that you want to scale this file. Similarly if you respond to the "second file" prompt with a `<cr>`, CMP will assume that you want to scale the first file. You will be prompted for the scaling/shifting factors for the y-data. (Note that the only way to scale the x-data is to use the /XA, B or the /X switches.)

Switches: /h, /q, /v, and

/+	— add two files.
/-	— subtract the two files (default if no <action> entered).
/*	— multiply them.
//	— divide.

/&	— combine in 3 columns.
/S	— sum (integrate) data. Note: Change from versions 3 and earlier, when it meant scale. (Use /y and /x for that.)
/R	— square root: $\sqrt{Y_{data}}$.
/U	— subtract from unity: $1. - Y_{data}$.
/I	— Invert: $1/Y_{data}$.
/L	— Log: $\log_{10}(Y_{data})$.
/E	— Exponentiate: $10^{(Y_{data})}$.
/X**n	— Raise to power: $(X_{data})^n$
/Y**n	— Raise to power: $(Y_{data})^n$
/ABS	— Absolute value: $|Y_{data}|$.
/CL$A, B$	— Clip data to range A to B.
/x or /y	— rescale data, with prompt for scaling factors
/yA or /yA, B	— rescale data as $y \Leftarrow y \times A + B$.
/xA or /xA, B	— rescale data as $x \Leftarrow x \times A + B$.
/2	— use x-data of 2nd file for output x values.
/Ci,j	— read Cols i, j.
/O	— Omit duplicate X values (default).
/D	— Include Duplicate x values. (Was /E!)
/RPH	— data are in an RPH file.

Examples:

CMP /- Mydata.mrg Myphon.on Myout.dat
     Mydata.mrg $-$ Myphon.on $\Rightarrow$ Myout.dat
CMP Mydata.mrg Myref.mrg //
     Mydata.mrg / Myref.mrg $\Rightarrow$ Mydata.div

Note the following default extensions for CMP's output files.

    ADD: +
    DIF: $-$
    MUL: *
    DIV: /
    CMB: &
    SUM: integral
    INV: I i.e., 1/
    1-Y: U i.e., 1$-$
    L10: LOG
    10U: 10^n
    SQY: $\sqrt{}$
    XPO: x^n
    YPO: y^n
    ABS: $|y|$
   CLP: clipped data
    SCA: scaled data

4. *FTS*

FTS—Fourier Transform Smooth—is a program for fringe elimination and smoothing. It works by Fourier transforming data and filtering the transform. You do fringe elimination (if necessary) first and then noise reduction.

When FTS reads data. it first removes broad trends by subtracting a polynomial fit. The data (now with average value zero) are then Fourier transformed from frequency to time domain.

Fringe elimination: Fringes have a sinusoidal pattern in the frequency spectrum, and so are at well-defined locations in the time-domain transform. By zeroing the fringe signature, the fringe pattern can be reduced without much affecting the rest of the spectrum.

You will be asked for four breakpoints, which you enter by clicking the mouse. The program multiplies the data by a function which goes to zero as a quadratic between the first two breakpoints and returns to 1 between the next two. Thus, to remove fringes, click twice to left of signature and twice to right of it:
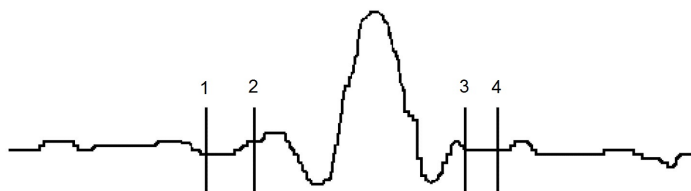
In general the first and second clicks should be separated by a few to a few tens of points. It helps if the second click is near a place where the time-domain data pass through zero. The third click (which also ideally will be near a zero crossing) and fourth should be to the right and separated by distances similar to clicks 1 and 2.

The program suppresses the spike, Fourier transforms back to the frequency domain, and displays the result. You can accept and save this result, retry the process, or apply further treatment. If you answer 'R' at the prompt, FTS will return to the data before the most recent cycle. If you answer 'M,' you can continue with removal of other fringe frequencies or to proceed to smoothing. A '<cr>' proceeds to the save-me dialog.

Noise elimination The basic idea of Fourier-transform smoothing is that the noise components have a white frequency spectrum, whereas the data are more slowly varying. Thus, by Fourier-transforming the file and reducing the amplitude of the large-time part of the transform, the noise can be reduced. In principal this approach is supposed to be less subjective than other methods.

If you click outside the frame, the breakpoints are moved to the end of the transform data. To apply an overall smooth to the data, click once about 10% of the way from zero to the end, a second time where the amplitude of the transform has gotten to its large-time value, and twice outside the right edge of the frame.

After the transform has been filtered, the inverse transform is done and the smoothed file is plotted. You can accept this result, try again from the start, or apply further smoothing.

8

If you answer 'R' at the prompt, FTS will return to the data before the most recent cycle. If you answer 'M,' you can continue with additional smoothing. A '<cr>' proceeds to the save-me dialog.

You may want to save the data after fringe elimination and then read it back in for smoothing.

Switches: /h, /q, and

/O	— Order of polynomial to fit, e. g., /O3 $\Rightarrow$ cubic.
/C*nnn*,*nnn* or /y*nnn*,*nnn*	— Clip data outside range supplied.
/COLi,j	— read Cols i, j.

Example:

FTS s114.bru s114.ftc /c0,1 /q

Here, s114.bru is the input file and s114.ftc is the output. /c0,1: Data clipped to window 0–1 before FT. /q: First plot is skipped; no rescaling is done.

Important notes:

1. Smooth individual files rather than merged or combined spectra.

2. Too much smoothing is not a Good Thing; the data look unnatural in that case.

5. GTF and GTFc

GTF—Glover-Tinkham Film or Glover-Tinkham Frontside—extracts the conductivity of a thin film on a thick substrate from the transmittance and reflectance. The algorithm is the one by Glover and Tinkham. This program assumes that the reflectance was measured on the film side of the film/substrate combination.

GTF is the graphics version; GTFc the windows console version.

You will be asked for the file to output σ_1 and σ_2 and then for data files for $\mathscr{T}$, $\mathscr{R}$, and the substrate. The latter may be a `*.nk`, `*.rph`, `*.prm` or `*.ain` file. `*.ain` is assumed if no extension is given. The output is a `*.s12` file. (This may be converted to `*.rph` with `s2rph.exe`.) If in reply to the prompt for substrate file, you enter no file name (just hit 'Enter'), you will be asked for a substrate refractive index n. The program then takes $n = $ constant and $\alpha = 0$.

You can provide a list of files instead of responding to the many prompts. The LOF must have the same name as the `s12` file file and extension 'LOF'. Then start the program with `GTF <lof-file-name>`. In the LOF file, you put the names of the $\mathscr{T}$, $\mathscr{R}$, and substrate files on the first 3 lines followed by film thickness (Angstroms) and substrate thickness (cm) on the next 2.

Switches: `/h`, `/q`, `/v`, and
- `/T`*nnn* — Film thickness in Angstroms is *nnn*.
- `/S`*nnn* — Substrate thickness in cm is *nnn*.
- `/W`*nnn* — Stop at frequency *nnn*.

A list of files would be in `MYDATA.LOF`, which might look like

```
f1006.tra
f1006.ref
sub.prm
2500 (film thickness in Angstroms)
0.07 (substrate thickness in cm)
```

6. INTERP and INTERPc

INTERP—INTERPolate—allows data to be interpolated, either on a finer or coarser scale than originally. The interpolation can be done on either a logarithmic or linear scale. INTERP can also scale data. The default is to ask for scale factors for the Y-data, but command line switches modify this.

INTERPc has the graphics removed.

Switches: `/h`, `/q`, and

/Xnnn,mmm	— x-axis plot range = nnn–mmm
	— *also* the interpolation range.
/Xmmm	— x-axis plot range = 0–mmm
	— *also* the interpolation range.
/Ynnn,mmm or /Ymmm	— y-axis plot range
/XL,	— do x axis on Log scale (also interpolation)
/YL,	— do y axis on Log scale
/Dnnn	— Delta x is nnn
/G	— do interpolation loGarithmically
/S	— reScale y axis data
/Sx	— reScale x and y axis data
/N	— No rescaling
/Ci,j	— Data in Columns i, j.

Examples:

`INTERP f1006.s1 f1006c.s1 /x0,5000 /log /y0,3000`

    Interpolate `f1006.s1` logarithmically, using the X-values in `f1006c.s1`.

`INTERP f1006.s1 -y3000 -q -s`

    Here, the x-axis is autoscaled; y-axis is from 0 to 3000. The user is asked for output file and for A, B for $x \Leftarrow A * x + B$.

7. KK

KK—Kramers-Kronig transform—performs a Kramers-Kronig transform for (single-bounce) reflectance $\mathscr{R}$ *and* (slab) transmittance $\mathscr{T}$ data. KK uses the graphics window to show you some of the intermediate steps.

You will need to specify the extrapolations to be used outside the data. In 2012 these were changed significantly.

At high frequencies, you must use a `<compound name>.XRO` file. These are generated by the xro program described on page 37. Examples would be `Ag.xro` or `YBa2Cu3O6.93.xro`. The xro files extend from 80,660 cm^{-1} (10 eV) to 241,980,000 cm^{-1} (30,000 eV). Above this range, the data are extrapolated as $\mathscr{R} \sim \omega^{-4}$ as appropriate for completely free electrons.

Unless your data extend to 80,600 cm^{-1} or so (unlikely if you do not have vacuum ultraviolet instrumentation) there will be a gap. Optical data typically stop between 20,000 and 40,000 cm^{-1}, or 2.5 and 5 eV. There are 4 methods of bridging the gap between your measured data and the `.xro` data:

1. A power law in $1/\omega$: $\mathscr{R} = A + B/\omega + C/\omega^2 + D/\omega^3 + \dots$ (Say `L#` at the prompt.)
2. A power law in ω: $\mathscr{R} = A + B\omega + C\omega^2 + D\omega^3 + \dots$ (Say `W#` at the prompt.)
3. A cubic spline can also be used. (Not recommended, but sometimes it works.)
4. No bridge. This is equivalent to a straight line connection.

For the two power laws you are shown a plot of $\mathscr{R}$ in the bridge region, and asked to select the points used in the fit. You also specify the order (the highest power of ω used in the fit).* The order is 3 in the above equations. The earlier extrapolation (in KKold) used $1/\omega^D$ with D in the range 0 to 4. Thus the power law in $1/\omega$ is probably the best choice. The default is to use $1/\omega$.

Sometimes the x-ray (xro) calculation produces a rather high reflectance in the 80,000–300,000 cm-1 region. The reflectance is "high" if it is well above the measured data in the visible/uv. The bridge will have a positive slope, and can contribute to a negative $\sigma_1(\omega)$ in the KK results. To work around this, use the `/W` or `/G` option. Before asking about the fit over the bridge region, KK will then show the high-frequency end of the data and the low end of the x-ray-extension reflectance. You will be asked to click twice to select the new start and end of the gap. Generally you will (1) click just right of the first data point (keeping all the data†) and (2) in the xro data where it has fallen below the measured data. This click pushes the start of the x-ray extension to higher energies and uses a power law over the wider gap.

In verbose mode (`\v`), KK will ask if you want to widen the gap. Verbose mode also asks if you want to keep (include) the low- and high-frequency extrapolated data (reflectance and phase). These options are also available with `/KL` and `/KH` switches. If you do keep the extrapolations in the output file, be sure to remove the extrapolation data before making plots for publication!

---

* If the order is too large, the fitting routine produces a singular matrix. In this case, KK reduces the order by 1 and tries again. You are limited to order ≤ 9.

† You may of course click somewhere inside your data if you want to trim what seem like unreliable uv data.

There are several low frequency extrapolations possible, controlled by a parameter `COND`. For KK of $\mathscr{R}$, one has:

>0	$\Rightarrow$	`COND` is the known DC conductivity, in $\Omega^{-1}\text{cm}^{-1}$
=0 or i	$\Rightarrow$	Reflectance is assumed constant to DC
-1 or m	$\Rightarrow$	Metallic conductivity assumed: $\qquad\qquad \mathscr{R} = 1 - A\omega^{1/2}$
-2 or t	$\Rightarrow$	Two-fluid model: $\qquad\qquad\qquad\quad \mathscr{R} = 1 - A\omega^2$
-3 or l	$\Rightarrow$	Marginal Fermi Liquid: $\qquad\qquad \mathscr{R} = 1 - A\omega^1$
-4 or s	$\Rightarrow$	Superconducting: $\qquad\qquad\qquad \mathscr{R} = 1 - A\omega^4$
-5 or f	$\Rightarrow$	Use a parameter file. (You will be asked for the name of the `*.PAR` file.)

The use of a parameter file (-5, f, or `/P<file>`) is strongly recommended. KK will use the parameters to calculate the low frequency reflectance or transmittance.[*]

You also need to specify the file names for input or output. The file order on command line is: Data, X-ray, Parameter, Output. KK will construct the output name from the data file name, with extension `.rph` or `.tph` if the name is not specified.

The data can be scaled or shifted and are clipped to 0.↔1. range.[†]

Switches: `/h`, `/q`, and

`/O<file>`	— Use `<file>` (.RPH) for output
`/F<file>`	— Use `<file>` (.XRO) for high freq extrapolation
`/BL<number>`	— Bridge with $\mathscr{R} = A + B/\omega + C/\omega^2 + \ldots$ up to `<number>`
`/BW<number>`	— Bridge with $\mathscr{R} = A + B\omega + C\omega^2 + \ldots$ up to `<number>`
`/BS`	— Bridge with cubic spline (which sometimes does not work...)
`/BN`	— No bridge (equivalent to a straight line)
`/W or /G`	— Widen Gap using mouse to select highest data and lowest .xro points to keep
`/C`*nn*	— low frequency extrapolation using $\sigma_{dc} = nnn$
`/I or /C0`	— Reflectance is assumed constant to DC (default)
`/C or /C-1`	— Metallic conductivity assumed: $\mathscr{R} = 1 - A\omega^{1/2}$
`/L or /C-2`	— London (two-fluid) model: $\mathscr{R} = 1 - A\omega^2$
`/M or /C-3`	— Marginal-Fermi Liquid: $\mathscr{R} = 1 - A\omega^1$
`/S or /C-4`	— Superconducting: $\mathscr{R} = 1 - A\omega^4$
`/P<file>`	— Use `<file>` (.PAR or .PRM) for low frequency extrapolation (preferred)
`/KH`	— Keep high frequency extrapolation points
`/KL`	— Keep low frequency extrapolation points
`/PI`	— Subtract π from the phase data
`/T`	— transform of Transmittance
`/TPH`	— transform Transmittance and write *.TPH file
`/X`*nnn* or `/x`*nnn*	— film thickness (Angstroms) (for $\mathscr{T}$)
`/Y`*nnn,mmm*	— Scale data as $Y \leftarrow nnn * (Y - mmm)$

---

[*] If the fit is not good at low frequencies, you may want to pick out the data up to say 500 cm^{-1} and optimize the parameters for a good low-frequency fit.

[†] The actual clipping algorithm allows the data to be 1.0 as long as it is 1.0 at all lower frequencies; otherwise 0.999999 is the largest value allowed.

```
    /COLi,j                    — Data in Columns i, j.
```
Examples:
```
KK F1A100.rnm  YBa2Cu3O6.93.xro /t /c-1 -q
```
    Transform `f1a100.rnm`, a transmittance file. Use built-in defaults, except extend low-frequency region as metal. Use `YBa2Cu3O6.93.xro` at high frequencies.
```
KK F1A100.RAW  /FYBa2Cu3O6.93 /PYBa2Cu3O6.93.PAR /BL4 /KH
```
```
KK F1A100.RAW  YBa2Cu3O6.93 YBa2Cu3O6.93.PAR /BL4 /KH
```
    Transform `f1a100.raw`. The low frequencies are extrapolated using the parameters in `YBa2Cu3O6.93.PAR`. Use `YBa2Cu3O6.93.xro` in the x-ray region. Bridge with $\mathscr{R} = A + B/\omega + C/\omega^2 + D/\omega^3 + E/\omega^4$. Keep the high-frequency extrapolation data.*

KK produces an `RPH` file, where the three data columns are frequency, reflectance, and phase shift on reflectance; ω, $\mathscr{R}$, ϕ. From these, the complex refractive index is

$$N(\omega) = \frac{1 + \sqrt{\mathscr{R}(\omega)}e^{i\phi(\omega)}}{1 - \sqrt{\mathscr{R}(\omega)}e^{i\phi(\omega)}}.$$

The program OP can calculate a wide variety of optical constants from the `RPH` file. See pages 18–30 for the list and formulas.

KK uses a phase that is consistent with writing $re^{i\phi} = (N - 1)/(N + 1)$. The "true" phase (defined as the relation between the incident field and the reflected field, evaluated at the surface) would write $re^{i\phi'} = (1 - N)/(1 + N)$. You can get this phase by saying `KK/PI`.

---

* If you use `/KH` be sure to remove the extrapolation data before making plots for publication! Ditto for `/KL`.

8. *KK /T*

KK with the `/T` switch does the Kramers-Kronig analysis of transmittance.

Most of the comments above apply to transmittance.

KK /T produces an `RPH` file, where the three data columns are frequency, reflectance, and phase shift on reflectance; ω, $\mathscr{R}$, ϕ. From these, the complex refractive index is

$$N = \frac{1 + \sqrt{\mathscr{R}}\,e^{i\phi}}{1 - \sqrt{\mathscr{R}}\,e^{i\phi}}.$$

Note that the values of $\mathscr{R}$ and ϕ are those of the *same* material measured as single-bounce reflectance.

The bridging and x-ray (`*.xro`) high frequency extrapolations are as in the case of $\mathscr{R}$. KK /T assumes coherent transmission through a slab.* Above the x-ray range, the data are extrapolated as $\mathscr{T} \sim 1 - C\omega^{-2}$ as appropriate for completely free electrons.

There are several low frequency extrapolations possible, controlled by a parameter `COND`. For KK of $\mathscr{T}$, one has:

=0 or c $\Rightarrow$	Transmission is assumed constant to DC	
-1 or m $\Rightarrow$	Metallic conductivity assumed:	$\mathscr{T} = B + C\omega^2$
-2 or s $\Rightarrow$	Superconducting:	$\mathscr{T} = A\omega^2$

9. *KKc*

KKc—Kramers-Kronig transform "console version"—performs a Kramers-Kronig transform for (single-bounce) reflectance $\mathscr{R}$ *and* (slab) transmittance $\mathscr{T}$ data. KKc has no graphics.

---

* Otherwise the phase would not be meaningful.

10. MAV

MAV—Merge and AVerage—This program merges and/or averages several blocks of x, y data to produce one data set. It is intended for spectral data, where the experiment measures in several overlapping ranges. For example, you may obtain `file1` over the range 1–100, `file2` over 90–180, `file3` and `file4` over 150–350, etc.. MAV will combine these data. It compares in the overlap region the quantities $\int dx\,$`file1` and $\int dx\,$`file2` and multiplies the data in `file2` by a constant so as to make the two integrals equal. Then, the two data sets are combined using a weighted average that starts with 100% `file1` and ends with 100% `file2`.

MAV *will* combine non-overlapping ranges, but it doesn't work so well. It will merely equate the first point of the second-range data with the last point of the first-range data.

In use, you select the region where data are combined by clicking with the mouse. Click at each end of the data. The overlap region is the data between the left and right mouse clicks.

After the merge is calculated, the program will ask "Is this OK." If you don't like the result, say 'N', and enter the scaling factor for the second-range data.

To average files, enter the file names on a single line. Merging and averaging can be intermixed. In other words, you can average a bunch of files taken over identical x ranges, and then merge the average with data from other x ranges.

At the end, MAV prints a list of the scaling factors used (and their inverses) and gives you a chance to make a final scaling of the merged data. This gives you a chance to set the scaling factor for your "best" data to 1.

You will be asked for the names of the output file and at least two data files. You can provide a list of files instead of responding to the prompt. The LOF must have the same name as the output file and extension 'LOF.' Then start the program with `MAV <lof-file-name>`.

Switches: `/h`, `/q`, and

`/C`	— Cumulative: Initial scale is previous.
`/S`	— Find average and std dev of a set of files.
`/Xnnn,mmm or /Xmmm`	— x-axis plot range for the *first* two files.
`/Ynnn,mmm or /Ymmm`	— y-axis plot range.
`/COLi,j`	— Data in Columns i, j.
`/P or /PE`	— data are 'PE' Columns 1 and 3.

Example:

`MAV MYDATA.MRG`

A list of files would be in `MYDATA.LOF`, which might look like

`Y04A.bru`
`Y03A.bru,y03b.bru, y03C.bru`
`Y02A.bru y02b.bru`
`Y01A.bru`

This will merge four frequency regions, averaging data sets A, B, and C in region 2 and data sets A and B in region 3.

11. MEND

MEND is a program to "MEND" a data file.

MEND used to be named PATCH, but Windows 7 invokes the "Do you want this program to make changes?" dialogue any time a program named `patch.exe` is started. No matter what it does! What a bunch of marooons.

MEND replaces sections of data with a power-law fit. Data are selected with the mouse. You select first the range to be replaced; then, the range to be fitted. The second range should include the first.

The program can also scale data. The default is to ask for scale factors for the y-data, but command line switches modify this.

Switches: `/h`, `/q`, and

`/X`nnn, mmm	— x-axis plot range $= nnn$–mmm
`/X`mmm	— x-axis plot range $= 0$–mmm
`/Y`nnn, mmm `or /Y`mmm	— y-axis plot range
`/XL, /yl ...`	— do axis on Log scale
`/XA, /ya, /A ...`	— Autoscale axis (A:both)
`/S or /Sx`	— ask about reScale of x-axis data
`/N`	— No rescaling
`/D`n	— Degree of fit is n (0–4)
`/C`	— Change x increment in output data
`/COLi,j`	— Data in Columns i, j.

Examples:

```
Mend f1006.s1 f1006c.s1 /x0,5000 /xlog /y0,3000
Mend f1006.s1 -xauto-y3000 -q -s -d3
```

In the first, MEND plots file `f1006.s1` semilog, puts patched data in `f1006c.s1`. In the second, the x-axis is autoscaled y-axis is from 0 to 3000. A cubic fit is used for the patch. User is asked for output file and for A, B for $x \Leftarrow A \times x + B$.

Patch.bat

Patch.bat is a batch file which calls Mend. If you are used to typing patch, this will avoid file-not-found errors.

12. OP and OPc

OP—Optical Properties—computes Optical Properties from KK-determined reflectance and phase. 38 different functions may be calculated. The program can also subtract a model dielectric function from the data.

OPc is the windows console version.

Switches: `/h`, `/q`, and

/V*nnn*	— unit cell Volume is *nnn*
/Wp*nnn*	— Plasma frequency is *nnn*
/Inf*nnn*	— ϵ_∞ is *nnn*
/W*nnn, mmm*	— calculate between frequencies *nnn* and *mmm*
/Wm*nnn*	— calculate from 0.0 to maximum frequency *nnn*
/ALL or /all	— do ALL functions (38 files created)
/P<file>	— Partial dielectric function, $\epsilon - \epsilon_{\text{model}}$

The unit cell volume is needed for sum rules. The plasma frequency and ϵ_∞ are used for self-energy, effective mass etc. The max frequency (`/Wmnnn`) can be used to limit file size.

The switch for "Partial dielectric function" subtracts a model contribution to the dielectric function from $<\epsilon(\omega)>$ before calculation of the optical function. It uses parameters in a `*.PAR` file produced by DFF.

Response function switches:

/S1 or /S	— σ_1	/S2	— σ_2
/E1 or /E	— ϵ_1	/E2	— ϵ_2
/N	— refractive index, n	/K	— extinction coefficient, κ
/A	— Absorption coefficient, α	/SK	— SKin depth, δ
/L	— Loss function, $-\operatorname{Im} 1/\epsilon$	/LS	— Surface loss function, $-\operatorname{Im} 1/(\epsilon - 1)$
/RH	— Optical resistivity ρ	/RH2	— Imaginary ρ
/SU	— SUm rule on $\sigma_1(\omega)$, N_{eff}		
/SL	— Sum rule Loss function	/SF	— Sum rule surFace loss function
/I	— $-\operatorname{Imag}$(self-energy), $-\Sigma_2$	/RE	— REal(self-energy), Σ_1
/G	— Γ_1	/G2	— Γ_2
/T	— $1/\tau(\omega)$	/M	— effective Mass, m^*/m
/Wps	— superfluid plasma frequency, ω_{ps}		
/LL	— London Length, λ_L	/NFR	— delta-fn weight
/F	— Free carrier: ϵ_1 vs $1/\omega^2$		
/D	— Drude model: $1/\sigma_1$ vs ω^2		
/E	— Abs(Epsilon)	/S	— Abs(Sigma)
/TAN	— Tan(delta)	/TAN	— Abs(Tan(delta))
/R	— Reflectance, $\mathscr{R}$	/RPH	— $\mathscr{R}$ is in file from KK
/TI	— Slab T(incoherent)	/TC	— Slab T(coherent)
/RI	— Slab R(incoherent)	/RC	— Slab R(coherent)

Examples:
`OP F1A006 /S1`

Calculate conductivity (σ_1) from file `F1A006.RPH`

```
OP F1A006 -sum -v162.1 -q
```
> Calculate Sum rule from file `F1A006.RPH`; output in `F1A006.SUM`. Unit cell volume = 162.1 Å^3. Be quick.

OP uses as input an `RPH` file, containing in 3 columns the frequency (in cm^{-1}), the power reflectance $\mathscr{R}$, and the phase shift on reflection ϕ. This file is produced either by KK or, for simulated data, by DFC.

The complex (amplitude) reflectivity is $\sqrt{\mathscr{R}}\,e^{i\phi}$. The complex refractive index is calculated from the reflectivity:

$$N = \frac{1 + \sqrt{\mathscr{R}}\,e^{i\phi}}{1 - \sqrt{\mathscr{R}}\,e^{i\phi}}. \tag{1}$$

and the complex dielectric function is

$$\epsilon = \epsilon_1 + i\epsilon_2 = N^2. \tag{2}$$

A number of the optical functions depend on $\Delta\epsilon$, the dielectric function with something subtracted. If there is a single low energy band, with all the high frequency behavior lumped into ϵ_∞ then

$$\Delta\epsilon = \epsilon - \epsilon_\infty. \tag{3}$$

More generally, one may want to subtract off the contributions of phonons, interband transitions, etc. In this case, $\Delta\epsilon$ is the Partial dielectric function, $\epsilon - \epsilon_{model}$ that you invoke with /P<file>. OP will prompt you for ϵ_∞ if it is needed and not known.

Here are the functions that OP calculates. The frequencies are in wavenumbers (cm^{-1}), with $\omega[\text{ cm}^{-1}] = f[\text{Hz}]/c = \omega[\text{rad/s}]/2\pi c$. The computed results are in SI, mostly.

Optical "constants"

1. `Sigma_1`

The complex conductivity and dielectric function are related. In cgs the conductivity and frequency are both measured in sec^{-1} and the relation is

$$\epsilon = 1 + \frac{4\pi i\sigma}{\omega}, \tag{4}$$

whereas in SI the conductivity is in $\Omega^{-1}\text{cm}^{-1}$, and it is

$$\epsilon = 1 + \frac{i\sigma}{\omega\epsilon_0}. \tag{5}$$

with ϵ_0 the dielectric function of empty space.

In lab units, frequencies are in cm^{-1} and conductivities are in Ω^{-1}cm^{-1}, and the relation is

$$\epsilon = 1 + \frac{60i\sigma}{\omega}, \tag{6}$$

The real part of the optical conductivity (in Ω^{-1}cm^{-1}) is then

$$\sigma_1 = \frac{\omega\epsilon_2}{60}, \tag{7}$$

with ω in cm^{-1}. This equation* can be compared to the expression in cgs:

$$\sigma_1 = \frac{\omega\epsilon_2}{4\pi}, \tag{8}$$

with ω in sec^{-1} and σ_1 also in sec^{-1}. The cgs conductivity is 9×10^{11} times larger than the SI conductivity.

2. **Sigma_2**

The imaginary part of the optical conductivity (in Ω^{-1}cm^{-1}) is

$$\sigma_2 = -\frac{\omega(\epsilon_1 - 1)}{60}. \tag{9}$$

If you are using the /P (partial) option, then OP calculates

$$\sigma_2 = -\frac{\omega(\Delta\epsilon_1)}{60}. \tag{10}$$

3. **Epsilon_1**

$$\epsilon_1 = \mathrm{Re}(\epsilon) \tag{11}$$

4. **Epsilon_2**

$$\epsilon_2 = \mathrm{Im}(\epsilon) \tag{12}$$

5. **Refr Index**

N is calculated from reflectance and phase via the inversion of

$$\sqrt{\mathscr{R}}\,e^{i\phi} = \frac{N-1}{N+1} \tag{13}$$

and

$$n = \mathrm{Re}(N) \tag{14}$$

---

* The same factor of 60 appears if you calculate σ_{dc} from the plasma frequency and relaxation rate,

$$\sigma_{dc} = \frac{\omega_p^2}{60(1/\tau)}$$

with ω_p and $1/\tau$ in cm^{-1} and σ_{dc} in Ω^{-1}cm^{-1}. In reality, $60 = 377\ \Omega/2\pi$.

6. Ext Coeff.

The imaginary part of N is

$$\kappa = \text{Im}(N) \tag{15}$$

7. Alpha

The absorption coefficient (in cm^{-1}) is

$$\alpha = 4\pi\omega\kappa, \tag{16}$$

correct for ω in cm^{-1}. In cgs, this is

$$\alpha = 2\omega\kappa/c. \tag{17}$$

where ω is in $\sec^{-1}$.

8. Skin depth

In the skin effect the electric field goes as $e^{-x/\delta}$ but of course the wave governed by a refractive index N has a field which decays as $e^{-\kappa\omega x/c}$ (ω in rad/s). Hence:

$$\delta = \frac{10^8}{2\pi\omega\kappa}, \tag{18}$$

with ω in cm^{-1}. The factor of 10^8 makes the units of $\delta(\omega)$ in Å.

9. Loss Fnctn

The energy loss function governs the transmission of fast electrons through the material. These depend on the longitudinal response of the dielectric. It is typically written as $-\text{Im}(1/\epsilon)$; here we use L.

$$L = \frac{\epsilon_2}{(\epsilon_1^2 + \epsilon_2^2)} = -\text{Im}\frac{1}{\epsilon}. \tag{19}$$

10. Surface Loss Fnctn

The "surface" energy loss function is

$$L_s = \frac{\epsilon_2}{((\epsilon_1 - 1)^2 + \epsilon_2^2)} = -\text{Im}\frac{1}{\epsilon - 1}. \tag{20}$$

21

Sum rules

11. Sumrule S_1

This is the most important sum rule function calculated by OP. The f-sum rule for solids can be written

$$\int_0^\infty \sigma_1(\omega')d\omega' = \frac{\omega_{p,tot}^2}{8},$$ (21)

where $\omega_{p,tot} = \sqrt{4\pi n_{tot}e^2/m}$ is the plasma frequency for all the electrons in the solid. We may write a partial sum rule (for the conduction band, say) as

$$\int_0^\omega \sigma_1(\omega')d\omega' = \frac{\pi ne^2}{2m^*},$$ (22)

where m^* is the average effective mass of the band. If the upper limit is above the free-carrier contribution and below the onset of the interband transitions, the right hand side becomes $\omega_p^2/8$ with ω_p the conduction band plasma frequency.

Next, write $n = N_{eff}/V_c$ and let m be the free electron mass. The density is in the number of effective electrons per unit cell volume or formula volume. It generally better to consider the latter: V is the volume of the formula unit, typically V_{cell}/Z as specified by the structural paper. Z is the number of formula units in the unit cell. For example, NaCl is fcc with a lattice constant of 5.64 Å. The unit cell volume is 179.4 Å^3 but $Z = 4$, so $V = 44.8$ Å^3. Note that if you know the density you can always calculate the volume: $V = M/\rho$ with ρ the density and M the mass of the atom in the element or atoms in the compound.

Then, OP calculates

$$N_{eff}\frac{m}{m^*} = \frac{2mV_c}{\pi e^2}\int_0^\omega \sigma_1(\omega')d\omega'$$ (23)

with m the mass of an electron and m^* the effective mass of the charge carriers.

12. Sumrule LossFn

We may write a partial sum rule (for the conduction band, say) of the loss function as

$$\frac{1}{4\pi}\int_0^\omega \omega' L(\omega')d\omega' = \frac{\pi ne^2}{2m^*},$$ (24)

where m^* is the average effective mass of the band. If the upper limit is above the free-carrier contribution and below the onset of the interband transitions, the right hand side becomes $\omega_p^2/8$ with ω_p the conduction band plasma frequency. (The factor of $1/4\pi$ is the same as appears in $\omega\epsilon_2/4\pi = \sigma_1$.) Thus,

$$N_{eff}\frac{m}{m^*} = \frac{mV_c}{2\pi^2 e^2}\int_0^\omega \omega' L(\omega')d\omega'$$ (25)

13. Sumrule SurfaceLossFn

Same as Eqs. 24 and 25, but for L_s.

Frequency-dependent scattering, self-energy, and effective mass functions

The dielectric function of perfectly free (noninteracting) carriers with density n, charge $\pm e$ and mass m is given by

$$\epsilon(\omega) = \epsilon_\infty - \frac{\omega_p^2}{\omega^2} \tag{26}$$

with ω_p, the plasma frequency (in cgs)

$$\omega_p = \sqrt{\frac{4\pi n e^2}{m}}. \tag{27}$$

If now we allow scattering of free carriers from impurities, defects, surfaces, and lattice oscillations,* with a mean free time τ or scattering rate $1/\tau$, independent of the frequency, the dielectric function has the Drude form:

$$\epsilon(\omega) = \epsilon_\infty - \frac{\omega_p^2}{\omega^2 + i\omega/\tau} \tag{28}$$

Now, if we turn on interactions, the scattering rate becomes a function of frequency (and probably also of temperature). However, to allow the scattering rate, $1/\tau$ to vary with frequency and remain real quantity would cause the dielectric function (and conductivity) to violate the Kramers-Kronig relation. One may accommodate this either by considering the scattering rate to be a complex function or by also having the effective mass m^* to vary with frequency.

Theorists have devised a number of such functions: the complex self-energy function Σ,[†] the complex scattering (or memory) function Γ, a complex scattering rate $1/\tau^*$ and effective mass m^*, and a complex scattering rate $1/\tau^*$ and mass enhancement factor λ. OP can take the KK-derived dielectric function and compute each and all of these quantities.

14. -Imag self-energy (one-particle)

In some theories (such as the marginal Fermi liquid) the imaginary part of the self energy Σ plays the role of a scattering rate, with the dielectric function written as

$$\epsilon(\omega) = \epsilon_\infty - \frac{\omega_p^2}{\omega[\omega - \Sigma(\omega)]}. \tag{29}$$

with ω_p the (unrenormalized)[‡] plasma frequency and ϵ_∞ the limiting high frequency value of the dielectric function. The imaginary part of Σ is related to the quasiparticle lifetime through $1/\tau^*(\omega) = -\operatorname{Im}\Sigma(\omega)(m_b/m^*(\omega))$ with m^* the effective mass and m_b the band mass.

---

* In principal, coupling to phonons scatters the charge carriers by either absorption or emission of a phonon. In turn, these processes lead to a frequency and temperature-dependent scattering rate. For this discussion, the frequency dependence is ignored, as are the emission processes. Thus, we consider only the high-temperature limit.

† There are actually two of these.

‡ Unrenormalized means "what it would be if there were no interactions."

OP computes $-\operatorname{Im}\Sigma$ via

$$-\operatorname{Im}\Sigma(\omega) = -\frac{\omega_p^2}{\omega}\operatorname{Im}\frac{1}{\Delta\epsilon}. \tag{30}$$

with $\Delta\epsilon = \epsilon - \epsilon_\infty$. You must enter the plasma frequency and ϵ_∞. Or, use the `partial` (`/P<file>`) option to subtract a model dielectric function. Subtraction removes optical phonons and interband transitions which may occur.

15. Real self-energy (one-particle)

The real part of Σ is related to the effective mass m^* of the interacting carriers by

$$m^*(\omega)/m_b = 1 - \operatorname{Re}\Sigma(\omega)/\omega. \tag{31}$$

Hence, OP computes

$$\operatorname{Re}\Sigma(\omega) = \frac{\omega_p^2}{\omega}\operatorname{Re}\frac{1}{\Delta\epsilon} + \omega. \tag{32}$$

16. -Imag self-energy (two-particle)

A slightly different approach recognizes that optical excitations are created in pairs (electron and hole) and that the self-energy should be a two-particle process. Still, the imaginary part of the self energy Σ plays the role of a scattering rate, with the dielectric function written as

$$\epsilon(\omega) = \epsilon_\infty - \frac{\omega_p^2}{\omega[\omega - 2\Sigma(\omega/2)]}. \tag{33}$$

with ω_p the (unrenormalized) plasma frequency and ϵ_∞ the limiting high frequency value. The factors of 2 occur because the excitations are created in pairs (electron-hole pairs or quasiparticle pairs). (These factors do not appear in all theoretical treatments. Indeed the "one-particle" versions are much more commonly seen.) The imaginary part of Σ is related to the quasiparticle lifetime through $1/\tau^*(\omega) = -2\operatorname{Im}\Sigma(\omega/2)(m_b/m^*(\omega))$ with m^* the effective mass and m_b the band mass.

OP computes $-\operatorname{Im}\Sigma$ via

$$-\operatorname{Im}\Sigma = -\frac{\omega_p^2}{2\omega}\operatorname{Im}\frac{1}{\Delta\epsilon}. \tag{34}$$

with $\Delta\epsilon = \epsilon - \epsilon_\infty$. You must enter the plasma frequency and ϵ_∞.

After computing the right hand side of the above equation, OP divides the frequencies by 2, so that the dielectric function value at 400 cm^{-1} (for example) generates the $-\operatorname{Im}\Sigma$ value at 200 cm^{-1}.

Note that optical phonons and interband transitions may occur; OP allows you to subtract these from ϵ via its `/P<file>` capability.

17. Real self-energy (two-particle)

The real part of Σ is related to the effective mass m^* of the interacting carriers by

$$m^*(\omega)/m_b = 1 - 2 \operatorname{Re}\Sigma(\omega/2)/\omega. \tag{35}$$

Hence, OP computes

$$\operatorname{Re}\Sigma(\omega/2) = \frac{\omega_p^2}{2\omega} \operatorname{Re}\frac{1}{\Delta\epsilon} + \omega/2. \tag{36}$$

After computing the right hand side of the above equation, OP divides the frequencies by 2, so that the dielectric function value at 400 cm^{-1} (for example) generates the $\operatorname{Re}\Sigma$ value at 200 cm^{-1}.

18. Gamma_1

The frequency dependent and complex damping is often written in an extended Drude model by replacing $1/\tau$ by a "memory function" Γ. We write

$$\epsilon = \epsilon_\infty - \frac{\omega_p^2}{\omega(\omega + i\Gamma)} \tag{37}$$

with $\Gamma = \Gamma_1 + i\Gamma_2$.

$$\Gamma_1 = -\frac{\omega_p^2}{\omega} \operatorname{Im}\frac{1}{\Delta\epsilon}. \tag{38}$$

Note that $\Gamma_1 = -\Sigma_2$ (one-particle).

19. Gamma_2

The imaginary part of Γ is

$$\Gamma_2 = \frac{\omega_p^2}{\omega} \operatorname{Re}\frac{1}{\Delta\epsilon} + \omega \tag{39}$$

Note that $\Gamma_2 = +\Sigma_1$ (one-particle).

20. tau$^\wedge$-1*

$$1/\tau = -\omega\frac{\operatorname{Im}(\Delta\epsilon)}{\operatorname{Re}(\Delta\epsilon)} \tag{40}$$

21. m*/m

$$\frac{m^*}{m} = -\frac{\omega_p^2}{\omega^2} \operatorname{Re}(\frac{1}{\Delta\epsilon}) \tag{41}$$

25

22. Mass enhancement factor $\lambda(\omega)$

The effective mass may be written in terms of a mass enhancement factor $\lambda(\omega)$:

$$\frac{m^*}{m} = 1 + \lambda(\omega). \tag{42}$$

Hence

$$\lambda(\omega) = -\frac{\omega_p^2}{\omega^2}\operatorname{Re}(\frac{1}{\Delta\epsilon}) - 1 \tag{43}$$

Specialized functions, mostly for superfluids

23. Superfluid plasma freq

A superfluid has a dielectric function that is

$$\epsilon = \epsilon_\infty - \frac{\omega_{ps}^2}{\omega^2} \tag{44}$$

with ω_{ps} the superfluid plasma frequency. OP calculates

$$\omega_{ps} = \sqrt{\omega^2|\operatorname{Re}(\Delta\epsilon)|} \tag{45}$$

where the absolute value is used instead of a minus sign to avoid numerical problems. Be sure that ϵ is negative when you invoke this function!

If ω_{ps} is (nearly) constant in frequency, the behavior is that of a superfluid.

24. N_eff (free-el)

Again, writing

$$\epsilon = \epsilon_\infty - \frac{\omega_{ps}^2}{\omega^2} \tag{46}$$

with ω_{ps} the superfluid plasma frequency. OP then writes $\omega_{ps}^2 = 4\pi n_s(\omega)e^2/m$ so that:

$$n_s(\omega) = \omega^2 V_c C|\operatorname{Re}(\Delta\epsilon)| \tag{47}$$

where C is a numerical constant.

$n_s(\omega)$ should be nearly constant as a function of frequency for the concept to be valid.

25. London penetra. depth

The dielectric function in 23 is that of a London superconductor, so one may also calculate λ_L as

$$\lambda_L = \frac{10^8}{2\pi\omega\sqrt{|\operatorname{Re}(\Delta\epsilon)|}} \tag{48}$$

with ω in cm^{-1}. The factor of 10^8 makes the units of $\lambda_L(\omega)$ in Å.

$\lambda_L(\omega)$ should be nearly constant as a function of frequency for the concept to be valid.

Other specialized functions

26. `Epsilon_1 vs 1/w^2`

If

$$\epsilon = \epsilon_\infty - \frac{\omega_{ps}^2}{\omega^2} \tag{49}$$

then a plot of ϵ vs $1/\omega^2$ should be a straight line, with slope $-\omega_{ps}^2$. So this is equivalent to function 23.

27. `1/Sigma_1 vs w^2`

If the dielectric function is Drude like, then

$$\sigma_1(\omega) = \frac{\sigma_{dc}}{1 + \omega^2 \tau^2} \tag{50}$$

so you can make a plot of $1/\sigma_1$ vs ω^2. The plot goes like is

$$\frac{1}{\sigma_1(\omega)} = \frac{1}{\sigma_{dc}} + \frac{\tau^2}{\sigma_{dc}}\omega^2$$

This plot should be a straight line with intercept $1/\sigma_{dc}$ and slope τ^2/σ_{dc}. If needed, σ_{dc} can be read off the plot of $\sigma_1(\omega)$ assuming that the data extend well below $1/\tau$. Otherwise one can write the slope as $4\pi\tau/\omega_p^2$ and get ω_p from the plot of ϵ_1 vs. $1/\omega^2$.

28. `Resistivity (real)`

The optical resistivity is just the inverse of the optical conductivity, so the real part is

$$\rho_1 = \text{Re}(\frac{1}{\sigma}). \tag{51}$$

Because σ is complex, $\rho_1 = \sigma_1/(\sigma_1^2 + \sigma_2^2)$.

29. `Resistivity (imag)`

The imaginary part if the optical resistivity is

$$\rho_2 = \text{Im}(\frac{1}{\sigma}). \tag{52}$$

30. `Abs(eps)`

There are examples where the magnitude of the complex dielectric function $|\epsilon|$ displays power law behavior. Hence OP calculates

$$|\epsilon| = \sqrt{\epsilon_1^2 + \epsilon_2^2} \tag{53}$$

Note that this should be the entire function, so $\epsilon\infty$ is not automatically subtracted. If you want to compute $\Delta\epsilon$, use the `/P` option.

31. **Abs(sigma)**

This is

$$|\sigma| = \sqrt{\sigma_1^2 + \sigma_2^2} \tag{54}$$

32. **Tan(delta)**

The loss tangent is commonly used in the analysis of RF and microwave experiments. It is

$$\tan\delta = \frac{\epsilon_2}{\epsilon_1}. \tag{55}$$

This calculation uses the entire function, so ϵ_∞ is not automatically subtracted. If you want to compute $\Delta\epsilon$, use the /P option.

33. **Abs(Tan(delta))**

If you want this to be always positive, use

$$\tan\delta_{abs} = \left|\frac{\epsilon_2}{\epsilon_1}\right|. \tag{56}$$

Transmission and reflection

OP uses the complex dielectric function (and the thickness, if needed) to calculate reflectance and transmittance. For a sample of finite thickness, you have a choice to treat it as "thick" (i.e., incoherent) or "thin" (coherent). In the first case, OP adds intensities of multiple internal reflections while in the second OP adds amplitudes. The second case allows for interference, with fringes in the transmittance or reflectance.

34. **Reflectance**

The power reflectance (single-bounce reflectance) is

$$\mathscr{R} = \left|\frac{N-1}{N+1}\right|^2. \tag{57}$$

OP actually computes this rather than just reading the values from the .RPH file.

35. **T slab (thick)**

Add intensities.

OP computes

$$\mathscr{T} = \frac{(1-\mathscr{R}_{sb})^2\, e^{-\alpha d}}{1 - \mathscr{R}_{sb}^2\, e^{-2\alpha d}} \tag{58}$$

with d the thickness, α the absorption coefficient, and $\mathscr{R}_{sb}$ the single bounce reflectance. $\mathscr{R}_{sb} = [(N-1)/(N+1)]^2$.

36. T slab (fringes)

Add amplitudes.

To compute the coherent transmittance, OP first calculates the phase gain on passing the sample (one way),

$$\delta = 2\pi\omega N d, \tag{59}$$

the amplitude transmitted *into* the slab, from vacuum,

$$t_1 = \frac{2}{N+1}, \tag{60}$$

the amplitude transmitted *out of* the slab,

$$t_2 = \frac{2N}{N+1}, \tag{61}$$

and the amplitude reflection at the interfaces, incident from within

$$r_{in} = \frac{N-1}{N+1}. \tag{62}$$

Then the formula for the transmission coefficient (amplitude) is

$$t = \frac{t_1 t_2 e^{i\delta}}{1 - r_{in}^2 e^{2i\delta}} \tag{63}$$

and to get the transmittance, OP calculates

$$\mathscr{T} = t t^* \tag{64}$$

This is the Airy formula.

37. R slab (thick)

Add intensities.

OP computes

$$\mathscr{R} = \mathscr{R}_{sb} + \frac{\mathscr{R}_{sb}(1 - \mathscr{R}_{sb})^2 \, e^{-2\alpha d}}{1 - \mathscr{R}_{sb}^2 \, e^{-2\alpha d}} \tag{65}$$

with d the thickness, α the absorption coefficient, and $\mathscr{R}_{sb}$ the single bounce reflectance. $\mathscr{R}_{sb} = [(N-1)/(N+1)]^2$.

38. R slab (fringes)

Add amplitudes.

29

To compute the coherent reflectance, OP first calculates the phase gain on passing the sample (one way),

$$\delta = 2\pi\omega N d, \tag{66}$$

the amplitude transmitted *into* the slab, from vacuum,

$$t_1 = \frac{2}{N+1}, \tag{67}$$

the amplitude transmitted *out of* the slab,

$$t_2 = \frac{2N}{N+1}, \tag{68}$$

the amplitude reflection at the interfaces, incident from within

$$r_{in} = \frac{N-1}{N+1}, \tag{69}$$

and the front-surface reflectance

$$r_{fs} = \frac{1-N}{1+N}. \tag{70}$$

Here, the sign matters; these definitions make r_{fs} negative and r_{in} positive if $N > 1$.

Then the formula for the reflection coefficient (amplitude) is

$$r = r_{fs} + \frac{t_1 r_{in} t_2 e^{2i\delta}}{1 - r_{in}^2 e^{2i\delta}}. \tag{71}$$

Finally, to get the reflectance, OP calculates

$$\mathscr{R} = rr^*\tag{72}$$

13. PIK and PIK2

PIK—PIcK—Is a program to pick sections out of a data file.

Data are selected with the mouse. The output contains all data *between* the left and right mouse clicks. The program can also scale data. The default is to ask for scale factors for the y-data, but command line switches modify this.

Switches: `/h`, `/q`, and

`/P`nnn,mmm	— Pick data between nnn and mmm
`/S or /Sx`	— ask about reScale of x axis data
`/N`	— No rescaling
`/X`nnn,mmm	— x-axis plot range $= nnn$–mmm
`/X`mmm	— x-axis plot range $= 0$–mmm
`/Y`nnn,mmm `or /Y`mmm	— y-axis plot range
`/XL, /yl ...`	— do plot on Log scale
`/XA, /ya...`	— Autoscale axis
`/Ci,j`	— data are in Cols i and j

(Pik2 does not have the `/S` and `/N` options.)

The first file name on command line is the input; second is the output.

Examples:
```
PIK f1006.s1 f1006c.s1 /x0,5000 /xlog /y0,3000
PIK f1006.s1 -xauto-y3000 -q -s
```
In the first, PIK plots file `f1006.s1` semilog on scales given, puts picked data in `f1006c.s1`. In the second, the x-axis is autoscaled while the y-axis will be plotted from 0 to 3000. User is asked for output file and for A, B for $x \Leftarrow A \times x + B$.

PIK2 is a version which is designed for large data sets. After the plot you can Expand, Compress, Shift or shift Back the plot by typing 'E,' 'C,' 'S,' or 'B. S shifts the viewpoint to the right (higher x values) and B shifts to the left. The program loops to allow multiple selections. It works from left to right, starting each subsequent plot from the end of the previous selection.

Click the x box at the top of the screen to end it.

14. QP, Q, and QL

QP—QuickPlot—makes a plot on PC screen. Q is equivalent to `QP /q`. QL makes the plot with lines, rather than dots at each point.

After files on the command line are plotted, you can tell QP to plot more files, change scales, print the plot, or exit. You change plot scales by answering '/'. All files read so far will be replotted. A 'p' gives a screen dump. Exit by answering a `<cr>` to the next-file prompt.

Switches: `/h`, `/q`, and

`/N /X`nnn,mmm	— x-axis plot range = nnn–mmm
`/X`mmm	— x-axis plot range = 0–mmm
`/Y`nnn,mmm `or /Y`mmm	— y-axis plot range
`/XR(ect) or /Yr`	— do axis on Rectilinear scale
`/XL(og) or /Yl`	— do axis on Log scale
`/R`	— both x and y on Rectilinear scales
`/XA(uto) or /Ya`	— Autoscale axis
`/A`	— Autoscale both x and y axes
`/Sx or /Sy`	— ask about factor to reScale data
`/O`	— omit duplicate data points
`/E`	— read every point, even duplicates (default)
`/P or /PE`	— data are 'PE,' cols 1 and 3
`/Ci,j`	— data are in Cols i and j

Examples:

`qp f1006.s1 f1100.s1 /x0,5000 /xlog /y0,3000`

   Plot two files in semilog format on scales given:

`qp f1006.s1 f1100.s1 -xauto-y3000`

   Autoscale x-axis, plot y-axis from 0 to 3000:

`qp s647.44 s647.45 s647.46 /pe /sy`

   Plot 3 files, which are in PE format (data in columns 1, 3); rescale y-data:

15. RNM

RNM—ReNorMalize—is a program written to correct reflectance data for systematic errors from surface scattering, sample area, and reference reflectance.

RNM corrects the reflectance data by dividing the data by the reflectance of a coated sample and then multiplying by a file representing the reflectance of the coating. It can also be used to correct for the non-unity reflectance of a reference mirror. More generally, RNM will multiply/divide any data files.

The data are scaled as follows:

$$R_{out} = \frac{R_{data}}{R_{coated}} \times R_{ref} \tag{73}$$

Here R_{data} = sample; R_{coated} = coated sample; R_{ref} = coating or reference mirror.

Switches: `/h`, `/q`, and

`/Al`	— ref is in `RAL.DAT`
`/Ag`	— ref is in `RAG.DAT`
`/N`	— No coated sample data
`/T`	— Transmission: no Ref file
`/R`	— Repeat RNM on additional data
`/Ci,j`	— data are in Cols i and j

Example:

`RNM Mydata.mrg Mycoat.mrg Ral.dat Mydata.rnm`

   (Mydata.mrg / Mycoat.mrg) ∗ Ral.dat ⇒ Mydata.rnm

`RNM /N Mydata.mrg Ral.dat`

   Mydata.mrg ∗ Ral.dat ⇒ output; user prompted for file name.

`RNM Mydata.mrg Myblank.mrg /T /q`

   (Mydata.mrg / Mycoat.mrg) ⇒ Mydata.rnm; `/q` makes output name equal to name of first data file.

16. T2A

T2A—Transmittance to Alpha—computes absorption coefficient from transmittance. It is intended to be used for thick samples, not thin films. Here "thick" means $d \gg \lambda$. (d is thickness and λ is the wavelength.) It can take into account the (single-bounce) reflectance either from a file or a value of the refractive index.

The absorption coefficient α is calculated from the measured transmittance $\mathscr{T}$ by inverting

$$\mathscr{T} = \frac{(1 - \mathscr{R}_{sb})^2 e^{-\alpha d}}{1 - \mathscr{R}_{sb}^2 e^{-2\alpha d}} \tag{74}$$

with d the thickness and $\mathscr{R}_{sb}$ the single-bounce reflectance. This equation is quadratic in $e^{\alpha d}$ with one positive root. The solution is

$$e^{\alpha d} = \frac{(1 - \mathscr{R}_{sb})^2}{\mathscr{T}} \left[1/2 + \sqrt{1/4 + \frac{\mathscr{R}_{sb}^2 \mathscr{T}^2}{(1 - \mathscr{R}_{sb})^4}} \right] \tag{75}$$

T2A can also calculate the extinction coefficient κ, related to α by $\alpha = 2\omega\kappa/c$.

If you do not have data for $\mathscr{R}_{sb}$, give a value for the refractive index n. Then T2A will use $\mathscr{R}_{sb} \approx [(n-1)/(n+1)]^2$. The approximation is typically OK, because for a transparent sample, $n \gg \kappa$.

Switches: `/h`, `/q`, and

`/T`*nnn* or `/D`*nnn*	— Thickness is *nnn*
`/R<file>`	— Reflectance (single-bounce) is in `<file>`
`/N`*nnn*	— approximate reflectance using refractive index *nnn*
`/K or /E`	— calculate κ (Extinction coefficient)
`/P or /PE`	— PE file
`/Ci,j`	— Data are in columns i and j

Example: `T2A MYDATA /T0.12 /Rrefl.dat`

17. T2Approx

T2Approx—Transmittance to Alpha—computes absorption coefficient from transmittance. It can take into account in an approximate way the reflectance either from a file or a value of the refractive index. It considers $\mathscr{R}_{sb}$ in Eq. 74 to be small, so $\mathscr{R}_{sb}^2$ is really small. Then the transmittance $\mathscr{T}$ is $\mathscr{T} \approx (1 - \mathscr{R}_{sb})^2 e^{-\alpha d}$ and $e^{\alpha d} \approx (1 - \mathscr{R}_{sb})^2 / \mathscr{T}$

Switches: `/h`, `/q`, and

`/T`*nnn* or `/D`*nnn*	— Thickness is *nnn*
`/R<file>`	— Reflectance is in `<file>`
`/N`*nnn*	— approximate reflectance using refractive index *nnn*
`/K or /E`	— calculate κ (Extinction coef)
`/P or /PE`	— PE file
`/Ci,j`	— Data are in columns i and j

Example: `T2Approx MYDATA /T0.12 /Rrefl.dat`

18. T2K

T2K—Transmittance to Kappa—computes extinction coefficient from transmittance. It is intended to be used for thick samples, not thin films. Here "thick" means $d \gg \lambda$. (d is thickness and λ is the wavelength.) It can take into account the (single-bounce) reflectance either from a file or a value of the refractive index.

The extinction coefficient κ is calculated from the measured transmittance $\mathscr{T}$ by inverting

$$\mathscr{T} = \frac{(1 - \mathscr{R}_{sb})^2 e^{-\frac{2\omega}{c}\kappa d}}{1 - \mathscr{R}_{sb}^2 e^{-\frac{4\omega}{c}\kappa d}} \tag{76}$$

with d the thickness and $\mathscr{R}_{sb}$ the single-bounce reflectance. This equation is quadratic in $e^{\frac{2\omega}{c}\kappa d}$ with one positive root. The solution is

$$e^{\frac{2\omega}{c}\kappa d} = \frac{(1 - \mathscr{R}_{sb})^2}{\mathscr{T}} \left[\frac{1}{2} + \sqrt{\frac{1}{4} + \frac{\mathscr{R}_{sb}^2 \mathscr{T}^2}{(1 - \mathscr{R}_{sb})^4}} \right] \tag{77}$$

If you do not have data for $\mathscr{R}_{sb}$, give a value for the refractive index n. Then T2K will use $\mathscr{R}_{sb} \approx [(n-1)/(n+1)]^2$. The approximation is typically OK, because for a transparent sample, $n \gg \kappa$.

Switches: `/h`, `/q`, and
`/T`*nnn* or `/D`*nnn*	— Thickness is *nnn*
`/R<file>`	— Reflectance (single-bounce) is in `<file>`
`/N`*nnn*	— approximate reflectance using refractive index *nnn*
`/P` or `/PE`	— PE file
`/Ci,j`	— Data are in columns i and j

Example: `T2K MYDATA /T0.12 /Rrefl.dat`

19. TwoC (2c)

2C—Two Component—is a program makes a two-component decomposition of optical conductivity $\sigma_1(\omega)$ into Drude and midinfrared components. The algorithm was invented by Danilo Romero.

You will be asked first for the file to output the MIR average and then for at least two data files. First is the lowest T conductivity, which is used as the first estimate of the MIR conductivity. The rest are the ones to be decomposed.

To average files, enter the file names on a single line.

You can provide a list of files instead of responding to the prompt. The LOF must have the same name as the MIR average file and extension 'LOF.'

Switches: /h, /q, and

    /T*nnn*   — Tolerance for convergence *nnn*.

    /W*nnn*   — carry out Drude fits to frequency *nnn*.

A list of files would be in MYDATA.LOF, which might look like

```
f1006.s1
f1100A.s1, f1100B.s1 f1100C.s1
f1150A.s1 f1150B.sig
f1200.s1
```

2C will carry out four analyses, averaging A, B, and C in the second and data sets A and B in the third.

20. XRO

XRO—X-ray optics—calculates the reflectance, reflectance phase, thin-film transmittance, conductivty, dielectric function, and complex refractive index at X-ray wavelengths.

The scattering functions f_1 and f_2 for atomic constituents are those of Henke (1993),[*] with new KK transforms for f_1. The data are embedded in the `xro.exe` file; auxiliary files are not needed.

You will need to specify the formula, e. g., `YBa2Cu3O6.93`. Fractional subscripts are fine. Case matters: cobalt (Co) is not carbon monoxide (CO)! XRO does not do very complicated parsing of formulae, so a material with a formula of $(La_{0.5}Pr_{0.5})_{0.67}Ca_{0.33}MnO_3$ must be reported as `La0.335Pr0.335Ca0.33MnO3`.

You may either enter the formula on command line, as in `XRO YBa2Cu3O6.93`, or use the `/FYBa2Cu3O6.93` switch. The output file will then be `YBa2Cu3O6.93.XRO`, except in the following case. If you enter a non-formula on the command line, that will be the name of the output file and you will need to use the `/F` switch or answer the prompt for the formula. (Be careful here! YBCO is shorthand for the 123 superconductor, but it also is yttrium boron carbon oxide. Lower case, e. g., ybco, is always safe.)

You may also report the formula as `YBa_2Cu_3O_6.93`. (But *not* as `YBa_2Cu_3O_{6.93}`!) Of course, then your file will be named `YBa_2Cu_3O_6.93.XRO` which requires a heavy use of the shift key.

You will also need to specify either V, the volume per formula unit in cubic Angstroms, or the density, in grams/cc. Here, V is the volume of the formula unit, typically V_{cell}/Z as specified by the structural paper. Z is the number of formula units in the unit cell. For example, NaCl is fcc with a lattice constant of 5.64 Å. The unit cell volume is 179.4 Å^3 but $Z = 4$, so $V = 44.8$ Å^3. Note that density always works. You get the same xro data using the density of 2.165 g/cm^3 for NaCl or Na$_4$Cl$_4$. The volumes V would be 44.8 and 179.4, respectively.

XRO calculates the dielectric function using the original atomic scattering factor data developed by Henke.[*] Some information about this is at `http://henke.lbl.gov/optical_constants/`.[†]

The dielectric function[‡] is

$$\epsilon = 1 - \sum_j \frac{4\pi n_j e^2}{m\omega^2}(f_1^j - if_2^j) \tag{78}$$

where the sum runs over atoms j at number density n_j and with complex scattering factor

---

[*] B.L. Henke, E.M. Gullikson, and J.C Davis, "X-ray Interactions: Photoabsorption, Scattering, Transmission, and Reflection at E=50–30,000 eV, Z=1–92," *Atomic Data and Nuclear Data Tables* **54**, 181–342 (1993).

[†] XRO uses the first version of the scattering data because it is sampled at identical photon energies for each atom; the newer version is not.

[‡] Henke—and the website above—write an equation for the refractive index $N = 1 - (nr_0\lambda^2/2\pi)(f_1 + if_2)$ with $r_0 = e^2/mc^2$ the classical radius of the electron and $\lambda = 2\pi c/\omega$ the wavelength. This is clearly an expansion of $N = \sqrt{\epsilon}$.

f^j. Note that this has the right limiting high-frequency behavior, because $f_1^j \to Z^j$ (with Z^j here the atomic number)* and $f_2 \to 0$, so that $\epsilon \to 1 - \sum_j 4\pi n_j Z^j e^2/m\omega^2$

The refractive index is $N = \sqrt{\epsilon}$ and the reflectance is calculated from the usual equation, Eq. 57. The transmittance is calculated from Eqs. 59–64.

Examples:
```
XRO YBa2Cu3O6.93 /v186
XRO silver /fAg /r10.49 /P
```

In the first, XRO calculates for ybco, using a cell volume of 186 Å^3. In the second the reflectance and the phase are computed for Ag, based on a density of 10.49 g/cm^3. The output file is silver.XRP.

Switches:

/F<formula>	— Chemical formula
/Vnnn	— Volume/formula unit is nnn
/Dnnn /Rnnn	— Density is nnn
/E	— Energy in eV rather than cm^{-1}
/P	— Write also the phase; 3 columns data
/S1	— Write conductivity, $\sigma_1(\omega)$
/E1	— Write dielectric const, $\epsilon_1(\omega)$
/I	— Write Index (n and κ) rather than $\mathscr{R}$ and (with \P) ϕ
/T	— Write Transmittance, $\mathscr{T}$
/Xnnn	— For $\mathscr{T}$: the thickness (in Å) is nnn
/F	— Write f_1 and f_2 (for the *first* atom in the formula)
/q	— Fewer questions

Switches may be introduced with '/' or '-', have optional spaces between them, and may be in upper or lower case.

---

* There is a small relativistic correction.

C. Fitting routines and computational routines

These routines are for fitting data to models. The two most important are DFF and TFF DFF fits optical conductivity data, single-bounce reflectance, dielectric function, etc. to a Drude-Lorentz model dielectric function. TFF fits to reflectance or transmittance data of multilayer thin films on a substrate.

See page 75 for the file format of the "parameter" file (`*.PAR`) produced and used by these programs. (This is the preferred parameter format. The old `*.PRM` format is usable if you only want Drude-Lorentz models. It is described on page 77)

This section also lists computational programs. For example, DLC uses the results of DFF to calculate a variety of optical functions, similar to OP for KK results. RDrude calculates reflectance within a Drude model.

1. DFF

DFF—Dielectric Function FIT—makes a nonlinear least-square fit to optical data. The data may be any of a number of functions. The formulas used are the same as used in OP, listed on pages 19–30.

1. Optical conductivity, σ_1, (Eq. 8).
2. Imaginary optical conductivity, σ_2, (Eq. 9).
3. Real dielectric function, ϵ_1, (Eq. 11).
4. Imaginary dielectric function, ϵ_2, (Eq. 12).
5. Refractive index, n, (Eq. 14).
6. Extinction coefficient, κ, (Eq. 15).
7. Absorption coefficient, α, (Eq. 16).
8. Loss function, $-\operatorname{Im}(1/\epsilon)$, (Eq. 19).
9. Reflectance, $\mathscr{R}$, (Eq. 57).
10. $\mathscr{T}$ of a slab (thick) (Eq. 58).
11. $\mathscr{T}$ of a slab (fringes) (Eq. 64).
12. $\mathscr{R}$ of a slab (thick) (Eq. 65).
13. $\mathscr{R}$ of a slab (fringes) (Eq. 72).

DFF can also be used to calculate these functions.

Several optical models are available, as discussed below. Most of the models calculate the complex dielectric function,

$$\epsilon(\omega) = \epsilon_1(\omega) + \frac{4\pi i}{\omega}\sigma_1(\omega) \tag{79}$$

where ϵ_1 is the real part and $\epsilon_2 = 4\pi\sigma_1/\omega$ is the imaginary part. The functions in the above list all may be computed once $\epsilon(\omega)$ is known.

a. High frequency value

The limiting high frequency value of the dielectric function should be unity, but the measurements rarely achieve this frequency, so DFF allows for an ϵ_∞. This is a purely real number, typically between 1 and 4 if data extend to the uv.

In the following formulas, I'll include ϵ_∞ in every formula, though it only comes in once, of course.

b. Lorentz model

The Lorentz model (or Drude-Lorentz) dielectric function is the dielectric function of a classical harmonic oscillator or of a lifetime broadened atomic transition.

$$\epsilon = \epsilon_\infty + \frac{\omega_p^2}{\omega_0^2 - \omega^2 - i\omega\gamma} \tag{80}$$

where ω_p is the "plasma frequency," ω_g the resonant frequency, and γ the damping constant, equal to the inverse lifetime.

In solids, the Lorentzian is used for optically-active phonons (vibrational modes) and for interband transitions. It is a pretty good model for the former, but not so good for the latter as the widths of these transitions are governed by electronic band structure rather than damping.

There can of course be many* such transitions, so one can write:

$$\epsilon = \epsilon_\infty + \sum_{k=1}^{N} \frac{\omega_{pk}^2}{\omega_k^2 - \omega^2 - i\omega\gamma_{ek}} + \sum_{j=1}^{M} \frac{\Omega_{pj}^2}{\omega_j^2 - \omega^2 - i\omega\Gamma_j} \tag{81}$$

Here, the electronic and vibrational oscillators are explicitly separated as a reminder that they represent different physics, although the functional forms are the same.

The optical conductivity of the Lorentz oscillator is

$$\sigma_1 = \frac{\omega_p^2\omega^2\gamma}{4\pi[(\omega_0^2 - \omega^2)^2 + \omega^2\gamma^2]} \tag{82}$$

The parameters are entered in the sequence ω_p, ω_0, and γ.

c. Drude model

If the resonant frequency in the Lorentz model is put to zero, one has the Drude dielectric function:

$$\epsilon = \epsilon_\infty - \frac{\omega_{pD}^2}{\omega^2 + i\omega\gamma} \tag{83}$$

where now $\omega_{pD} = \sqrt{4\pi n e^2/m}$ is the Drude plasma frequency and $\gamma = 1/\tau$ with τ the mean free time between collisions.† With these definitions, the optical conductivity becomes

$$\sigma_1 = \frac{ne^2\tau/m}{1 + \omega^2\tau^2} \tag{84}$$

DFF uses the same formulas for Drude and Lorentz models. In the case of the former, you set the middle parameter (ω_0) to zero. DFF is smart enough to leave that at zero and not to adjust during the least squares optimization.

---

* Up to $N + M = 666$.

† With frequencies, including γ, in cm^{-1}, τ would be in cm. To convert to sec, you divide by $2\pi c$. The factor of 2π is *required*. It accounts for the fact that frequencies go from cm-1 to radians/sec by multiplying by $2\pi c$.

d. Rice EMV model

This model considers a set of phonons interacting with an electronic continuum. The N phonons are described by a set of coupling constants λ_k, resonant frequencies ω_k, and damping Γ_k. The phonons appear in a a "phonon propagator" D_0:

$$D_0 = \sum_{k=1}^{N} \frac{\lambda_k \omega_k^2}{\omega_k^2 - \omega^2 - i\omega\Gamma_k} \tag{85}$$

which then enters a Lorentz dielectric function as:

$$\epsilon = \epsilon_\infty + \frac{\omega_p^2}{\omega_0^2(1 - D_0) - \omega^2 - i\omega\gamma} \tag{86}$$

The EMV parameters are entered as λ_k, ω_k, and Γ_k. In addition, there must be at least one Lorentz oscillator, with parameters ω_p, ω_0, and γ. DFF will use the *first* Lorentz oscillator in the list as the one coupled to the phonons.

e. Product dielectric function

If one factors Eq. 80, puts it on a common denominator, and makes a few notation changes, the dielectric function can be written

$$\epsilon = \epsilon_\infty \frac{\omega_L^2 - \omega^2 - i\omega\gamma_L}{\omega_T^2 - \omega^2 - i\omega\gamma_T}, \tag{87}$$

where $\omega_T = \omega_0$ is the transverse optical frequency (a pole in the response function), $\omega_L = \sqrt{\omega_p^2/\epsilon_\infty + \omega_0^2}$ is the longitudinal optical frequency (a zero in the response function), and the two damping values γ_T and γ_L are both equal to γ, but here are written as different quantities, to give 4 parameters to the oscillator.

Of course there could be many[*] oscillators, so DFF uses a product form:

$$\epsilon = \epsilon_\infty \prod_{j=1}^{N} \frac{\omega_{Lj}^2 - \omega^2 - i\omega\gamma_{Lj}}{\omega_{Tj}^2 - \omega^2 - i\omega\gamma_{Tj}}. \tag{88}$$

The parameters are entered in the order ω_{Lj}, γ_{Lj}, ω_{Tj}, γ_{Tj}.

The imaginary part of Eq. 87 is

$$\epsilon_2 = \omega\epsilon_\infty \left[\frac{\omega_L^2\gamma_T - \omega_T^2\gamma_L + \omega^2(\gamma_L - \gamma_T)}{(\omega_T^2 - \omega^2)^2 + \omega^2\gamma_T^2} \right]. \tag{89}$$

Now, ϵ_2 cannot be negative, so we require

$$\gamma_T \leq \gamma_L \leq \gamma_T \frac{\omega_L^2}{\omega_T^2} \tag{90}$$

and

$$\omega_L \geq \omega_T. \tag{91}$$

DFF enforces obedience to Eqs. 90 and 91.

---

[*] Up to $N = 499$.

41

f. Indirect bandgap (~Tauc Lorentz)

If you look at Eq. 82 you will see that the conductivity (and hence the absorption) falls as ω^2 for frequencies well below the resonance (so long as γ is not too big). Actual solids have a much more rapid decrease near their gaps, with the optical penetration length changing from kilometers to micrometers over a few thousand wavenumbers. DFF addresses this by using a modified Tauc-Lorentz dielectric function.[*] One knows that the absorption for an indirect bandgap semiconductor (or in some cases amorphous semiconductors) just above the gap follows:

$$\alpha(\omega) \sim (\omega - \omega_g)^2, \tag{92}$$

where w_g is the indirect bandgap.

This behavior must stop eventually to maintain finite oscillator strength; experimentally the absorption has a maximum at a frequency somewhere above the gap and falls above this maximum. The Tauc-Lorentz model was written to account for this behavior. The imaginary part of the dielectric function can be written:

$$\epsilon_2 = \begin{cases} 0 & \omega \leq \omega_g \\ C \left(\dfrac{\omega - \omega_g}{\omega} \right)^2 \dfrac{\omega_0^2 \omega \gamma}{(\omega_0^2 - \omega^2)^2 + \omega^2 \gamma^2} & \omega \geq \omega_g. \end{cases} \tag{93}$$

ϵ_2 is zero up to ω_g and has a maximum at ω_0. In Tauc-Lorentz, there are four parameters, ω_g, ω_0, γ, and C. DFF removes one of these, making $\omega_0 \simeq 1.45 \omega_g$. In addition, the coefficient C is adjusted so that

$$\int_0^\infty d\omega' \sigma_1(\omega') = \frac{\omega_p^2}{8} \tag{94}$$

(The oscillator strength in Eq. 93 is *not* independent of γ the way the Drude or Lorentz functions are.) Hence the adjustable parameters are the gap, ω_g, the width, γ, and ω_p representing the integrated oscillator strength.

The parameters are entered in the sequence ω_p, ω_g, and γ.

g. Direct bandgap

DFF follows the same approach for a direct bandgap semiconductor, using a modified Tauc-Lorentz dielectric function. One knows that the absorption for a direct bandgap semiconductor just above the gap follows:

$$\alpha(\omega) \sim (\omega - \omega_g)^{1/2}, \tag{95}$$

where w_g is the direct bandgap.

---

[*] See also the next subsection for the function for the case of a direct bandgap.

This behavior must also stop eventually to maintain finite oscillator strength; experimentally the absorption has a maximum at a frequency above gap and falls above this maximum. The imaginary part of the direct bandgap dielectric function is written:

$$\epsilon_2 = \begin{cases} 0 & \omega \leq \omega_g \\ C\left(\dfrac{\omega - \omega_g}{\omega}\right)^{1/2} \dfrac{\omega_0^2 \omega \gamma}{(\omega_0^2 - \omega^2)^2 + \omega^2 \gamma^2} & \omega \geq \omega_g. \end{cases} \tag{96}$$

ϵ_2 is zero up to ω_g and has a maximum at ω_0. DFF sets $\omega_0 \simeq 1.225\omega_g$. In addition, the coefficient C is adjusted so that

$$\int_0^\infty d\omega' \sigma_1(\omega') = \frac{\omega_p^2}{8} \tag{97}$$

(The oscillator strength in Eq. 96 is *not* independent of γ the way the Drude or Lorentz functions are.) Hence the adjustable parameters are the gap, ω_g, the width, γ, and ω_p representing the integrated oscillator strength.

The parameters are entered in the sequence ω_p, ω_g, and γ.

h. Marginal Fermi liquid

The MFL dielectric function is given in Eq. 29.

$$\epsilon(\omega) = \epsilon_\infty - \frac{\omega_p^2}{\omega[\omega - \Sigma(\omega)]}. \tag{98}$$

where ω_p is a "bare" plasma frequency and $\Sigma(\omega)$ is the quasiparticle self-energy. The MFL theory has the imaginary part of the self-energy linear in ω up to a cutoff frequency; above this frequency it is taken as constant. Hence, below the cutoff,

$$\Sigma = 2\lambda\omega \ln\left(\frac{2\pi T - i\omega}{\omega_c}\right) - 2\pi^2 i\lambda T \tag{99}$$

Above the cutoff, this becomes

$$\Sigma = 2\lambda\omega_c \ln\left(\frac{2\pi T - i\omega_c}{\omega_c}\right) - 2\pi^2 i\lambda T \tag{100}$$

The three parameters are entered as ω_p, λ, and γ. DFF also will need the temperature, in Kelvin.

i. Fermi liquid

The Fermi liquid dielectric function is

$$\epsilon(\omega) = \epsilon_\infty - \frac{\omega_p^2}{\omega[\omega + i\gamma(\omega)]}. \tag{101}$$

where ω_p is a "bare" plasma frequency and $\gamma(\omega)$ is a frequency-dependent damping. The Fermi liquid theory has the imaginary part of the self-energy quadratic in ω. This would

43

continue up to a cutoff frequency of order the Fermi energy. Here, this is assumed to be above the highest measured frequency.

$$\gamma = \gamma_0 + \left(\frac{\omega}{\omega_{FL}}\right)^2.$$ (102)

The three parameters are entered as ω_p, γ_0, and ω_{FL}.

Note that the model is incomplete in two ways. First, the cutoff ought to be in the parameter set. Second, and more serious, Kramers-Kronig requires the frequency-dependent damping to be a complex function. Alternatively, the mass component of ω_p should also be frequency dependent.

j. Kivelson-Emery model

This model is based on phase separation (stripes), driven by the lower energy of a spin singlet but frustrated by long-range coulomb interaction. The theory gives the following expression for the optical conductivity:

$$\sigma_1 = e^2 A \frac{\chi_2(\omega, T)}{\omega} + (e^*)^2 \omega \chi_2(\omega, T)$$ (103)

where e^* is the charges of dipolar fluctuations (e is the charge of the mobile carriers), A is related to exchange, and χ_2 is

$$\chi_2 = c \tanh\left(\frac{\hbar\omega}{2kT}\right) \frac{\Gamma}{\Gamma^2 + \omega^2},$$ (104)

where c us the concentration of dipolar fluctuations, T is the temperature, and Γ is the Kondo scale of the problem, ~0.1 eV (1000 K).

You can only fit this model to the optical conductivity σ_1. The parameters are A, e^*, and Γ. DFF also need to know the temperature.

k. DFF usage

DFF will prompt you for the parameters needed, in groups of three or four. For the Lorentz (and Drude) models these are ω_p, ω_0, and γ. Unless you are doing optical conductivity, you will also be asked to specify ϵ_∞.

You first provide the parameters (or a file containing them) and are given a chance to change them. This "eyeball fitting" phase of the program can be looped as many times as you wish, producing a plot of the adjusted response function each time. You may also write the parameter file and the calculated function to disk.

During the eyeball fit part, the program replots the fitted calculation after *each* set of 3 parameters is entered, i. e., for one oscillator, makes a plot of the result, and then asks about this oscillator immediately again. This allows you to modify the strength, center frequency, and linewidth for each oscillator until you are happy with it. A `<cr>` moves you to the next oscillator in turn, and so on, and finally to ϵ_∞.

Answering `L` to the prompt about what to do (`E, L,` or `Q`) causes the program to carry out least square adjustment of parameters you specify. You are asked about the mode of the fit, the maximum number of times to loop through the χ^2 minimization process, and the criterion for stopping the fit. Then you are given a chance to "fix" or "free" the parameters. The parameters for each oscillator are presented, and you enter "0" to fix them; "1" to free them up. (In the list where they are presented, parameters that are fixed are shown in parentheses. If you want no change, you can enter a "/" or "`<cr>`".)

Switches: `/h, /q,` and

`/S`	— data are $\sigma_1(\omega)$	`/S2`	— data are σ_2
`/E`	— data are $\epsilon_1(\omega)$	`/E2`	— data are ϵ_2
`/R`	— data are Reflectance	`/RPH`	— in KK file
`/L`	— data are Loss function		
`/A`	— data are Absorption coefficient		
`/N`	— data are refractive index	`/K`	— extinction coeff.
`/TI`	— Slab $\mathcal{T}$(incoherent)	`/TC`	— Slab $\mathcal{T}$(coherent)
`/RI`	— Slab $\mathcal{R}$(incoherent)	`/RC`	— Slab $\mathcal{R}$(coherent)
`/TEnnn`	— Temperature is nnn, in Kelvin		
`/Dnnn`	— Thickness is nnn in Angstrom units		
`/Pnnn,nnn`	— partial fit, to data between the two frequencies given		
`/ST`	— Read Std dev file		
`/PRM`	— Use old ".PRM" parameter file		
`/Z`	— Do not ask about fix/free parameters		
`/Mn`	— Mode: 1 = std dev; 0 = none; −1 = 1/data		
`/Fn`	— Fraction of parameter used for initial step		
`/Innn`	— maximum number of Iterations in least square fit		
`/Tnnn`	— nnn is Tolerance (χ^2 has to decrease by at least <tol> to continue least square fits)		
`/Ci,j`	— Data in Columns i, j.		

"Mode" specifies the way in which χ^2 is calculated. (χ^2 is what the program minimizes.) If <mode> = 1, the actual standard deviation of the data is used; you will be prompted for a file containing the standard deviations. If <mode> = 0 each data point is assumed to have the same standard deviation. If <mode> = -1, the standard deviation is set to 1/data.

DFF uses the GRIDLS routine from the book by Bevington.

Example:
`DFF F1A006.S1 /S F1A006.PRM /T.001 /I25 /P120,990`
  Fit file `F1A006.S1`, which is $\sigma_1(\omega)$, using `F1A006.PRM` over 120–990 cm^{-1}. <tol> = .001; <max iterations> = 25

2. TFF

TFF—Thin Film Fit—fits reflectance, $\mathscr{R}$, transmittance, $\mathscr{T}$, or absorbance ($\mathcal{A} = \infty - \mathscr{R}$) for thin film multilayers using a variety of models for dielectric functions of the layers. It can also be used to calculate these functions.

The dielectric function models are the same as used by DFF, described on pp. 39–44. TFF will prompt you for the parameters needed. An example is the Drude-Lorentz model, for which it needs parameters in groups of three: ω_p, ω_0, and γ. You will also need to specify ϵ_∞ and the thickness.

You first provide the parameters (or a file containing them) for each layer, and are given a chance to change them. This "eyeball fitting" phase of the program can be looped as many times as you wish: the program replots the fitted calculation after *each* set of parameters is entered, starting with ϵ_∞ and thickness, and then oscillators. TFF makes a plot of the result, and then asks about this oscillator immediately again. This allows you to modify the strength, center frequency, and linewidth for each oscillator until you are happy with it. A <cr> moves you to the next oscillator in turn, and so on. You may also write the parameter file and the calculated function to disk.

Answering L to the prompt about what to do (E, L, or Q) causes the program to carry out least square adjustment of parameters you specify. You are asked about the mode of the fit, the maximum number of times to loop through the χ^2 minimization process, and the criterion for stopping the fit. Then you are given a chance to "fix" or "free" the parameters. The parameters for each oscillator are presented layer by layer, and you enter "0" to fix them; "1" to free them up. (In the list where they are presented, parameters that are fixed are shown in parentheses. If you want no change, you can enter a "/" or "<cr>".)

Switches: /h, /q, and

/R	— data are Reflectance	/RPH	— in KK file
/K	— bacKside Reflectance		
/T	— data are Transmittance		
/A	— data are Absorabance, $\mathcal{A} = 1 - \mathscr{R}$		
/Ln	— n Layers in sample		
/S [/Sn]	— Substrate is thick [not thick]'with opaque substrate		
/B [/Bn]	— Back of substrate [not] seen in reflectance		
/D	— there is an .RPH file for the substrate		
/Mn	— Mode: 1 = std dev; 0 = none; −1 = 1/data		
/Fn	— Fraction of parameter used for initial step		
/Innn	— maximum number of Iterations in least square fit		
/Onnn	— nnn is tOlerance (χ^2 has to decrease by at least <tol> to continue least square fits)		
/Xnnn,mmm	— x-axis plot range = nnn–mmm		
/Xmmm	— x-axis plot range = 0–mmm		
/Ynnn,mmm	— y-axis plot range		
/XL, /yl ...	— do axis on Log scale		
/Ci,j	— Data in Columns i, j		

"Mode" specifies the way in which χ^2 is calculated. (χ^2 is what the program minimizes.)

If <mode> = 1, the actual standard deviation of the data is used; you will be prompted for a file containing the standard deviations. If <mode> = 0 each data point is assumed to have the same standard deviation. If <mode> = -1, the standard deviation is set to 1/data.

Note: The film thicknesses is passed in Angstrom units (even for the thick substrates) to avoid format problems in the I/O routines of TFF. Note further that all parameters except eps_∞ and the thickness are in units of cm^{-1}.

Note 2: The program asks about each layer in sequence, beginning from the top and ending with the substrate (or bottom layer. If there are L_a layers, then $N_s = 1 + L_a$ is the number of interfaces. Moreover, there are $2 + L_a$ distinct media, because the first and last extend to ∞. Here is a picture:

```
|\|   /|\   \ |
-------->--<----|-|-/-|-\--\|---->------
R       |\|/  |  \  |    T
Incident  La =   1  2    3     final
Ns =    1 2   3      4
N (arg)    1      2  3    4        5
```
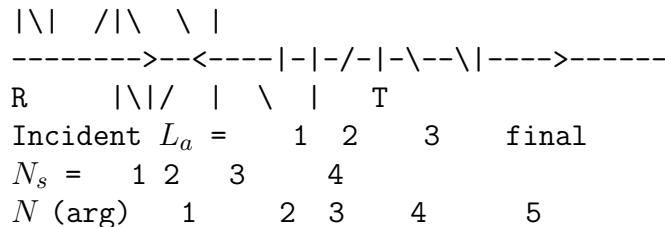
Note 3: `REFTR2`, the subroutine for the thin film transmission, does NOT account for any differences between incident and final medium. If these have different refractive indices, then $\mathscr{T}$ must be multiplied by $\mathrm{Real}(n(La^{\max})/n(1))$.

TFF uses a modification of the GRIDLS routine from the book by Bevington.

Examples:
`TFF F1A006.DAT   F1A006.PRM`
     Fit file `F1A006.DAT`, using `F1A006.PRM` as the parameter file.
`TFF F1A006.DAT /T.001 /I25`
     Fit file `F1A006.DAT`, <tol> = .001; <max iterations> = 25

3. DFC

DFC—Dielectric Function Calculate—calculates a variety of optical functions, such as optical conductivity (σ_1), dielectric constant (ϵ_1), and many more quantities. It also can calculate the function for each component of the model, where that is appropriate. It calculates (from models) all of the optical functions that OP calculates from KK results. See pages 19–30 for the equations governing these functions.

The program uses a number of models (Drude-Lorentz, interband, marginal Fermi liquid, etc.) for the complex dielectric function. The models are described on pages 39–44.

DFC gets its parameters from a `*.PAR` file produced by DFF or TFF (or the oldstyle `*.PRM` file produced by DLFIT/FLMFIT). It will either ask you for the frequencies at which to do the calculation or use the frequencies in a data file you specify.

Switches: `/h`, `/q`, and

`/IN`	— Do the INdividual components
`/V`*nnn*	— unit cell Volume is *nnn*
`/Wp`*nnn*	— plasma frequency is *nnn*
`/RPHin`	— Frequencies come from KK file
`/PI`	— Subtract π from the phase data
`/X`*nnn*`,`*mmm*	— *x*-axis plot range = *nnn*–*mmm*
`/X`*mmm*	— *x*-axis plot range = 0–*mmm*
`/Y`*nnn*`,`*mmm*	— *y*-axis plot range
`/XL, /yl ...`	— do axis on Log scale `/XR` — linear scale
`/Ci,j`	— Data in Columns i, j

The unit cell volume is used for sum rules; the plasma frequency for self-energy and effective mass. *Note:* In calculating the 'components', $\epsilon_\infty = 1.0$ except for $\epsilon_1(\omega)$ itself where $\epsilon_\infty = 0$.

Response function switches:

`/S1 or /S`	— σ_1	`/S2`	— σ_2
`/E1 or /E`	— ϵ_1	`/E2`	— ϵ_2
`/N`	— refractive index, n	`/K`	— extinction coefficient, κ
`/A`	— Absorption coefficient, α		
`/R`	— Reflectance, $\mathscr{R}$		
`/L`	— Loss function, $-\operatorname{Im} 1/\epsilon$		
`/M`	— effective Mass, m^*/m		
`/T`	— $1/\tau(\omega)$		
`/I`	— $-\mathrm{Imag}$(self-energy), $-\Sigma_2$	`/RE`	— REal(self-energy), Σ_1
`/SU`	— SUm rule on $\sigma_1(\omega)$, N_{eff}	`/SL`	— Sum r. Loss function
`/Wps`	— superfluid plasma frequency, ω_{ps}		
`/LL`	— London Length, λ_L		
`/SK`	— SKin depth, δ		
`/F`	— Free carrier: ϵ_1 vs $1/\omega^2$		
`/D`	— Drude model: $1/\sigma_1$ vs ω^2		
`/RPHout, /3`	— Refl and phase; write `*.RPH` file		

Example:

```
DFC F1A006.S1 /S F1A006.PRM
```

Calculate $\sigma_1(\omega)$, using `F1A006.PRM` at the frequencies in `F1A006.S1`.

Produce `F1A006.DLC`, `F1A006.DRU`, `F1A006.L1`, etc.

4. EMA

This program uses the Bruggeman Effective Medium Approximation to calculate the dielectric function, conductivity, reflectance, refractive index, absorption coefficient, loss function, and other quantities for a 2-component mixture. The program uses a Drude-Lorentz model for the complex dielectric function for each component.

Examples:

EMA met.prm ins.prm

Calculate ema using met.prm ins.prm

EMA F1A006.S1

Calculate fit to F1A006.S1; parameter file names will be requested by the program.

Switches: `/h`, `/q`, and

`/Xnnn,mmm (/Y)`	— x(y) range = nnn–mmm
`/NPnnn`	— NP points
`/XR(/Yr)[/XL..]`	— Do axis on rectilinear [log] scale
`/Ci,j`	— Data are in columns i and j
`/RPHin (/RPHout/3)`	— Read (write) 3 col *.rph file
`/Fnnn`	— Fill fraction is *nnn*
`/Dnnn`	— Depolarization factor is *nnn*
`/Vnnn`	— Unit cell volume is *nnn*
`/Wpnnn`	— Plasma frequency is *nnn*

Response function:

`/S1 or /S`	— σ_1	`/S2`	— σ_2
`/E1 or /E`	— ϵ_1	`/E2`	— ϵ_2
`/R`	—Reflectance		
`/RPH or /3`	— Reflectance and phase		
`/N`	— Refr Index	`/K`	— Extinction coeff.
`/A`	— Absorption coeff.	`/L`	— Loss function
`/I`	— Imag(self-energy)	`/RE`	— Real(self-energy)
`/G or /G1`	— $\Gamma_1(\omega)$	`/G2`	— Γ_2
`/M`	— Effective mass m^*/m	`/T`	— $1/\tau(\omega)$
`/SU`	— Sum rule: $\sigma_1(\omega)$	`/SL`	— Loss function
`/Wps`	— Superfl. plasma freq.	`/LL`	— London length
`/SK`	— Skin depth		
`/|E|`	— Abs(ϵ)	`/|S|`	— Abs(σ)
`/TAN`	— Tan(δ)	`/|TAN|`	— Abs(Tan δ)

5. PFIT

PFit—Power series fit—fits data to a power series.

The data are fitted to a polynomial in x. Pfit outputs 4000 points (an arbitrarily chosen number) from the fitted curve over the data range.

Switches (must be passed on the command line): `/h`, `/q`, and

`/O`	— Order of polynomial to fit, eg, /O3==cubic.
`/Z`	— (Almost) suppress zero offset of x data.
`/L`	— Fit log(x) vs. log(y), for power law fit. (Order is 1.)
`/OM`	— Omit dupicate data points in input file.
`/X`nnn, mmm `or /x`mmm	— Compute over x range nnn–mmm or 0–mmm.
`/Ci,j`	— data are in Cols i and j

Default polynomial order: 2 (quadratic, with intercept: $y = a_0 + b_0 x + c_0 x^2$).

Maximum polynomial order: 8 (octal, with intercept).

To omit the intercept, put *two* 0,0 entries at start of file.

The zero suppress option moves the data array until the first data point is almost at $x = 0$. (It is at $x = x(2) - x(1)$.) This option does not work if you use the two 0,0 entries to omit the intercept. It is also switched off if $x(1) = 0$.)

Example:
`PFit osc.str osc.fit /col3,1 /o3`
Here, `osc.str` is the input file and `osc.fit` is the output.

6. RDrude

Calculate the reflectance for Drude model.

Switches: `/h` and

`/F`*nnn*`,`*nnn*	— Frequency range *nnn* to *mmm*
`/D`*nnn*	— Frequency interval is *nnn*
`/C`*nnn*	— DC Conductivity is *nnn*
`/P`*nnn*	— Plasma Frequency is *nnn*
`/E`*nnn*	— Epsilon_infity is *nnn*

7. TCalc

TCalc calculates optical data. It can compute the dielectric function, conductivity, bulk reflectance, and several other quantities. It does this at 2 or more temperatures, making assumptions about $\gamma(T)$.

The program uses a Drude-Lorentz model for the complex dielectric function.

Switches: `/h`, `/q`, and

`/X`*nnn*`,`*mmm*	— x-axis plot range $= nnn$–mmm
`/X`*mmm*	— x-axis plot range $= 0$–mmm
`/Y`*nnn*`,`*mmm*	— y-axis plot range
`/XL, /yl ...`	— do axis on Log scale `/XR` — linear scale
`/V`*nnn*	— unit cell Volume is *nnn*
`/Wp`*nnn*	— plasma frequency is *nnn*
`/RPHin`	— Frequencies come from KK file

The unit cell volume is used for sum rules; the plasma frequency for self-energy and effective mass. *Note:* In calculating the 'components', $\epsilon_\infty = 1.0$ except for $\epsilon_1(\omega)$ itself where $\epsilon_\infty = 0$. This is an issue for quantities that are nonlinear functions of $eps(\omega)$.

Response function switches:

`/S1 or /S`	— σ_1	`/S2`	— σ_2
`/E1 or /E`	— ϵ_1	`/E2`	— ϵ_2
`/N`	— refractive index, n	`/K`	— extinction coefficient, κ
`/A`	— Absorption coefficient, α		
`/R`	— Reflectance, $\mathscr{R}$		
`/RPHout, /3`	— Refl and phase; write `*.RPH` file		
`/L`	— Loss function, $-\operatorname{Im} 1/\epsilon$		
`/M`	— effective Mass, m^*/m		
`/T`	— $1/\tau(\omega)$		
`/I`	— $-\mathrm{Imag}$(self-energy), $-\Sigma_2$	`/RE`	— REal(self-energy), Σ_1
`/SU`	— SUm rule on $\sigma_1(\omega)$, N_{eff}	`/SL`	— Sum r. Loss function
`/Wps`	— superfluid plasma frequency, ω_{ps}		
`/LL`	— London Length, λ_L		
`/SK`	— SKin depth, δ		
`/F`	— Free carrier: ϵ_1 vs $1/\omega^2$		
`/D`	— Drude model: $1/\sigma_1$ vs ω^2		

D. Spectral analysis

These spectral analysis programs do Fourier transforms. There are some test programs included.

1. *FFT*

`FFT` calculates Fourier transforms of a time series.

`FFT` calculates the transform of the input data, extended to a power of two. It has only two apodization/truncation functions: rectangle (boxcar) and Hamming. It will do up to 4,194,304 points (at 0 zerofill).

Switches: `/h`, `/q`, and

`/A<char>`	— Window (apodization) function.
`/AB or /AR`	→ Boxcar == Rectangle.
`/AH`	→ Hamming window (default!).
`/Z#`	— Zerofill factor (default $1 \rightarrow$ none).
`/Dnnn`	— Sampling interval is *nnn* sec (for 1 col data).
`/Fnnn`	— Sampling rate is *nnn* Hz (for 1 col data).
`/P`	— Compute Power spectrum.
`/aph or /ph`	— write amplitude and phase.
`/Ci,j`	— Data in Columns i, j.

Examples:'

`FFT s114.bru s114.ftr /c1,2`

    `s114.bru` is the input file and `s114.ftr` is the output.

    `/c1,2` → data are in columns 1 and 2.

2. *FTgram*

`FTGram` calculates Fourier transforms of subsets of a time series. It will do up to 4,194,304 points (at 0 zerofill). It produces many subspectra at (of course) lower resolution than if the entire time series is used. It also produces a list of maxima in the subspectra as 'Time,' 'Peak frequency,' and 'N sigma height.'

Switches: `/h`, `/q`, and

`/U#`	— number of sUbspectra. [Should be odd!]
`/V#`	— degree of oVerlap in time series (default 1).
`/S#`	— peaks are # Sigma (default 8).
`/A<char>`	— Window (apodization) function.
`/AB or /AR`	→ Boxcar == Rectangle.
`/AH`	→ Hamming window (default!).
`/Z#`	— Zerofill factor (default 2).
`/Dnnn`	— Sampling interval is *nnn* sec (for 1 col data).
`/Fnnn`	— Sampling rate is *nnn* Hz (for 1 col data).
`/P`	— Compute Power spectrum.
`/N, /NS, /NOS`	— Don't write spectra.
`/NP, /NOP`	— Don't write frequencies of peaks.
`/Ci,j`	— Data in Columns i, j.

3. IFFT

IFFT program calculates the inverse Fourier transform of a data set. It will do up to 4,194,304 points (at 0 zerofill).

Switches: `/h`, `/q`, and

`/A<char>`	— Window (apodization) function.
`/AB or /AR`	→ Boxcar == Rectangle.
`/AH`	→ Hamming window (default!).
`/Z#`	— Zerofill factor (default 2).

Example:

`IFFT s114.ftr s114.tr /c1,2`

　　`s114.ftr` is the input file and `s114.ttr` is the output.
　　/c1,2 → data are in columns 1 and 2.

4. *Bits*

Bits simulates N bits data resolution. It scales floating-point data to a range of (say) -8–+7, converts the floating point data to an integer, and then converts back, rescaling the data into (approximately) the same range of values it had before being quantized. Note that one bit is reserved for the sign, so the range -8–+7 (16 distinct values) represents 4-bit data.

The program underfills the window to avoid clipping, with the result that the area under the rms value of the bit-reduced data is smaller than in the original data.

Switches: `/h`, `/q`, `/v`, and

`/Nnn`	— N-bit data
`/Xnnn,mmm`	— x-axis plot range = nnn–mmm
`/Xmmm`	— x-axis plot range = 0–mmm
`/S`	— reScale y data
`/Sx`	— reScale x data

Example:

`Bits Data.rnu /N8 /a2`

Here, Bits reduces data.rnu to 8 bit (+/- 4 range) data.

5. *Inject*

This program injects a sine wave into data. The program can also scale data. The default is to ask for scale factors for the Y-data, but command line switches modify this.

The first file name on command line is input; second is output.

Switches: `/h`, `/q`, `/v` and

`/F`*nn*	— Frequency is *nn* Hz.
`/D`*nn*	— Doppler chirps the frequency by *nn*/sec.
`/P`*nn*	— Phase is *nn*.
`/s<file>`	— Stored phase in ¡file¿.
`/A`*nn*	— Amplitude is *nn*.
`/X`*nnn*,*mmm*	— x-axis plot range = *nnn*–*mmm*
`/X`*mmm*	— x-axis plot range = 0–*mmm*
`/N`	— No rescaling
`/Sy`	— reScale y data
`/Sx`	— reScale x data

Example:'
`Inject Data.rnu /F12340  /a2`
   Here, Inject injects into `data.rnu`, at f = 12,340 Hz.
   The amplitude is 2.0.

6. *Pinkify*

Gaussian noise has desirable properties: well defined spectrum, bounded, stationary... Many noise sources are none of the above, such as for example $1/f$ noise. Pink noise is white at high frequencies and $1/f$ at low frequencies. Pinkify (used in conjunction with `FFT` and `IFFT`) will convert a time series made up of Gaussian noise to noise with a pink spectrum. To do this you have to generate the Gaussian time series, Fourier transform to get a frequency spectrum, convert to pink noise with Pinkify, inject any signals you want to inject, and then compute the inverse Fourier transform. The result is a time series with pink noise. Note that the phase is unchanged by Pinkify.

In sum: Ranu → FFT to amplitude → Pinkify, as

$$Y(f) = Y(f) * [1 + (\frac{f_0}{f})^a],$$

(phase unchanged) → IFFT to time series.

Switches: `/h`, `/q`, `/v` and

    `/fnnn`   — $1/f$ below frequency *nnnn*

    `/a`α — α in $1/f^\alpha$

7. *Ranu*

This program generates a N-point series of random numbers. The maximum number of output points is 4,200,000. The numbers are Gaussian. The default average = 0 and default standard deviation 1.

Ranu simulates a noisy time series, and you can specify the (equally spaced) sampling rate for the time with the `/f` option. The default is 100 kHz (10 μsec).

Switches: `/h`, `/q`, `/v`, `\ci,j` and

`/N`*nnn*	— Number of points is *nnn*
`/F`*nnn*	— Sample Frequency is *nnn*
`/A`*nnn*	— Make average *nnn*
`/S`*nnn*	— Make standard deviation *nnn*
`/R`*nnn*	— Random seed is *nnn*
`/X`*nnn,mmm*	— x-axis plot range = *nnn*–*mmm*
`/X`*mmm*	— x-axis plot range = 0–*mmm*

Example:

`RaNu Data.rnu /n10240 /f50000 /r123 /v /sx`

Here, RaNu calculates 10240 points, at time intervals of 20 microseconds (1/50000) using 123 as the seed of the generator. User is asked for output file and for A, B for $x \leftarrow A * x + B$.

E. Utilities

There are a number of small programs for converting among presentations of complex optical data, converting from wavenumber to wavelength, converting JCAMP/DX files, etc.. These are described in the following.

1. RPH2E

2. RPH2NK

3. RPH2S

4. E2RPH

5. NK2RPH

6. S2RPH

7. L2RPH

These programs all convert among presentations of complex optical data. For example, RPH2E gives ϵ_1 and ϵ_2 from Kramers-Kronig-determined reflectance and phase. Some programs can also subtract a model dielectric function from the data.

Here are the programs and what they do:

Program	Input file	Output file	Conversion
RPH2E	*.RPH	*.E12	$\omega, \mathscr{R}, \phi \rightarrow \omega, \epsilon_1, \epsilon_2$
RPH2NK	*.RPH	*.NK	$\omega, \mathscr{R}, \phi \rightarrow \omega, n, \kappa$
RPH2S	*.RPH	*.S12	$\omega, \mathscr{R}, \phi \rightarrow \omega, \sigma_1, \sigma_2$
E2RPH	*.E12	*.RPH	$\omega, \epsilon_1, \epsilon_2 \rightarrow \omega, \mathscr{R}, \phi$
NK2RPH	*.NK	*.RPH	$\omega, n, \kappa \rightarrow \omega, \mathscr{R}, \phi$
S2RPH	*.S12	*.RPH	$\omega, \sigma_1, \sigma_2 \rightarrow \omega, \mathscr{R}, \phi$
L2RPH	*.L12	*.RPH	$\omega, \mathrm{Re}(1/\epsilon), \mathrm{Im}(1/\epsilon) \rightarrow \omega, \mathscr{R}, \phi$

Switches: `/h`, `/v`, and

`/Wm`*nnn*	— calculate up to max frequency *nnn*
`/E`*nnn*	— $\epsilon_\infty = nnn$ (default = 1.0)
`/P<file>`	— Partial dielectric function, $\epsilon - \epsilon_{DL}$
`/PI`	— Subtract π from the phase data

The ϵ_∞ switch (only in S2RPH) allows σ_2 to be corrected for a high-frequency slope (equivalent to $\epsilon_1(\omega) \rightarrow \epsilon_1(\omega) - 1.0 + \epsilon_\infty$) before converting to reflectance and phase.

The "Partial dielectric function" option allows you to subtract a Drude-Lorentz model from $<\epsilon(\omega)>$ before the calculation of optical function. It either uses parameters in a *.PRM file from DFF or prompts for user responses. This "Partial" switch only exists in RPH2E and RPH2S. And, it only works with oldstyle *.PRM parameter files. Use OP for more general subtraction.

The "Pi" switch subtracts π from the phase. It only exists in L2RPH, E2RPH, S2RPH, and NK2RPH.

Example:

```
RPH2E F1A006
```
    Calculate ϵ_1 and ϵ_2 from file F1A006.RPH

8. *Deriv*

This program calculates the derivative of data in a file. It is a Windows "console" program.

Switches: **/h**, **/q**, **/v** and

    /Ci,j   — data are in Columns i and j.
    /P or /PE   — data are 'PE' Columns 1 and 3.

9. *E2E*

E2E (end to end) converts optical constants files from ascending to descending order. (or vice versa.) This also works for 3 column data.

Switches: **/h**, **/q**, and

    /T or /3   — there are 3 Columns of data to swap
    /Ci,j   — data are in Columns i and j

Example:
```
E2E PY2.K PY2-R.K
```
    Convert PY2.K to descending order.

10. *GetDat*

GetDat (Get Data) picks data from multi-column data and writes out 2 or 3 of them.

Switches: **/h**, **/q**, and

    /T or /3   —3 Columns output data
    /Ci,j(,k)   — data are in Columns i and j (and k if 3)
    /R   — Input is in descending order (hi to lo)
    /SY$A,B$ **or /SY**A — rescale y data as $Y \leftarrow A * Y + B$
    /SX$A,B$ **or /SX**A — rescale x data

Example:
```
GetDat Widedat.txt TwoOf.dat
```
    Extract two columns from Widedat.txt.

11. *J2W*

J2W—Jcamp to Wavenumber—converts JCAMP-DX files produced by commercial IR machines to a table of wavenumber, data values. It works for Bruker, Niclolet, and a set of test files from the web.

Switches: **/h**, **/q**, and

    /D   — Write fixDate batch file. (Sets **/Q** also.)

Example:
```
J2W 27T.DX1 27T.JX1
```
    Convert 27T.DX1 to wavenumber, data. Result in 27T.JX1

Note: The FixDate batch file, if run, sets the date of the converted file to that of the original JCAMP-DX file. It requires 4NT, and uses the internal "touch" function.

12. PRM2PAR

PRM2PAR converts oldstyle parameter files (`.PRM`) to newstyle (`.PAR`) format.

PRM2PAR can also generate a new `*.par` file, for the Drude-Lorentz model. Use the `/K` switch, and PRM2PAR will prompt you for the parameters in groups of three: ω_p, ω_0 and γ. You will also need to specify ϵ_∞.

Examples:
```
rm2Par F1A006.PRM F1A006.PAR
```
    Convert parameter file F1A006.PRM to F1A006.PAR'
```
Prm2PAR/q F1A006
```
    Convert parameter file F1A006.PRM to F1A006.PAR'

13. W2L

W2L—Wavenumber to Lambda—is a program which converts an optical properties file between wavenumber and wavelength ($\mathrm{cm}^{-1} \leftrightarrow \mathrm{nm}/\mu\mathrm{m}$).

Switches: `/h`, `/q`, and

`/M or /U`	— Wavelength in microns.
`/V`	— Go from wavelength to wavenumber.
`/R`	— Input is in descending order (i. e., high to low).
`/Sy`A or `/Sy`A, B	— rescale data as $y \Leftarrow y \times A + B$.
`/Sx`A or `/Sx`A, B	— rescale data as $x \Leftarrow x \times A + B$.
`/3 or /T`	— Three-column data (like `*.RPH`).
`/Ci,j`	— data are in Columns i and j.

Example:
```
W2L PY2.K PY2-L.K
```
    Convert PY2.K to wavelength.

14. XY2YX

XY2YX is a program which exchanges columns in a 2 column data file.

Switches: `/h`, `/q`, `/Ci,j`, `/PE`.

Example:
```
XY2YX  Latt.T temper.lat
```
    Exchange data columns in `Latt.T`; output in `temper.lat`.

15. *Z2W*

Z2W—Zeiss to Wavenumber—is a program that converts an optical properties file from wavelength to wavenumber (nm $\rightarrow$ cm^{-1}) and—by default—percents to fractions. The program is designed for Zeiss MPM data. It also tidies up the header, removing unneeded lines and adding the standard "/*" at the end.

Switches: `/h`, `/q`, and

> `/N` — Do not convert percents to fractions.

Example:

`Z2W PY2.K PY2-L.K`

> Convert PY2.K to wavenumber.

F. Obsolete and depreciated programs

The notes below describe programs that are no longer maintained. Most have been replaced with better ones, different enough to require a new name. See the list on page 2 for the replacements.

1. DLFIT

DLFIT—Drude-Lorentz FIT—makes a nonlinear least-square fit of optical conductivity (σ_1), dielectric constant (ϵ_1), Reflectance ($\mathscr{R}$), refractive index (n), absorption coefficient (α) to the Drude-Lorentz model. It can also be used to calculate these functions.

The program uses a Drude-Lorentz model for the complex dielectric function. It will prompt you for the parameters needed, in groups of three: ω_p, ω_0, and γ. Unless you are doing optical conductivity, you will also be asked to specify ϵ_∞.

You first provide the parameters (or a file containing them) and are given a chance to change them. This "eyeball fitting" phase of the program can be looped as many times as you wish, producing a plot of the adjusted response function each time. You may also write the parameter file and the calculated function to disk.

During the eyeball fit part, the program replots the fitted calculation after *each* set of 3 parameters is entered, i.e., for one oscillator, makes a plot of the result, and then asks about this oscillator immediately again. This allows you to modify the strength, center frequency, and linewidth for each oscillator until you are happy with it. A `<cr>` moves you to the next oscillator in turn, and so on, and finally to ϵ_∞.

Answering `L` to the prompt about what to do (`E, L,` or `Q`) causes the program to carry out least square adjustment of parameters you specify. You are asked about the mode of the fit, the maximum number of times to loop through the χ^2 minimization process, and the criterion for stopping the fit. Then you are given a chance to "fix" or "free" the parameters. The parameters for each oscillator are presented, and you enter "`0`" to fix them; "`1`" to free them up. (In the list where they are presented, parameters that are fixed are shown in parentheses. If you want no change, you can enter a "`/`" or "`<cr>`".)

Switches: `/h`, `/q`, and

`/S`	— data are $\sigma_1(\omega)$
`/E`	— data are $\epsilon_1(\omega)$
`/R`	— data are Reflectance `/RPH` — in KK file
`/L`	— data are Loss function
`/N`	— data are refractive index
`/A`	— data are Absorption coefficient
`/P`*nnn*`,`*nnn*	— fit to data between the two frequencies given
`/M`*n*	— Mode: 1 = std dev; 0 = none; −1 = 1/data
`/F`*n*	— Fraction of parameter used for initial step
`/I`*nnn*	— maximum number of Iterations in least square fit
`/T`*nnn*	— *nnn* is Tolerance (χ^2 has to decrease by at least <tol> to continue least square fits)

"Mode" specifies the way in which χ^2 is calculated. (χ^2 is what the program minimizes.) If <mode> = 1, the actual standard deviation of the data is used; you will be prompted for a file containing the standard deviations. If <mode> = 0 each data point is assumed to have the same standard deviation. If <mode> = -1, the standard deviation is set to 1/data.

DLFIT uses the GRIDLS routine from the book by Bevington.

Example:
`DLFIT F1A006.S1 /S F1A006.PRM /T.001 /I25 /P120,990`
Fit file `F1A006.S1`, which is $\sigma_1(\omega)$, using `F1A006.PRM` over 120–990 cm^{-1}. <tol> = .001; <max iterations> = 25

2. ELPHIT

ELPHIT—ELectron-PHonon fIT—makes a nonlinear least-square fit of optical conductivity (σ_1), dielectric constant (ϵ_1), Reflectance ($\mathscr{R}$), refractive index (n), absorption coefficient (α) to the Drude-Lorentz model. It can also be used to calculate these functions.

The program uses the Rice model of coupling a set of phonons to electrons. It will prompt you for the parameters needed, in groups of three: λ_α, ω_α and γ_α ; ω_p, ω_0 and γ; also ϵ_∞.

You first provide the parameters (or a file containing them) and are given a chance to change them. This "eyeball fitting" phase of the program can be looped as many times as you wish; the program replots the fitted calculation after *each* set of 3 parameters is entered, i. e., for one oscillator, it makes a plot of the result, and then asks about this oscillator immediately again. This allows you to modify the strength, center frequency, and linewidth for each oscillator until you are happy with it. A `<cr>` moves you to the next oscillator in turn, and so on, and finally to ϵ_∞.

Answering `L` to the prompt about what to do (`E, L, or Q`) causes the program to carry out least square adjustment of parameters you specify. You are asked about the mode of the fit, the maximum number of times to loop through the χ^2 minimization process, and the criterion for stopping the fit. Then you are given a chance to "fix" or "free" the parameters. The parameters for each oscillator are presented, and you enter "0" to fix them; "1" to free them up. (In the list where they are presented, parameters that are fixed are shown in parentheses. If you want no change, you can enter a "/" or "`<cr>`".)

Switches: `/h`, `/q`, and

`/S`	— data are $\sigma_1(\omega)$		
`/E`	— data are $\epsilon_1(\omega)$	`/E2`	— data are $\epsilon_2(\omega)$
`/R`	— data are Reflectance	`/RPH`	— in KK file
`/L`	— data are Loss function		
`/N`	— data are refractive index		
`/A`	— data are Absorption coefficient		
`/P`*nnn*`,`*nnn*	— fit to data between the two frequencies given		
`/M`*n*	— Mode: $1 = $ std dev; $0 = $ none; $-1 = 1/$data		
`/F`*n*	— Fraction of parameter used for initial step		
`/I`*nnn*	— maximum number of Iterations in least square fit		

/T*nnn*   — *nnn* is Tolerance (χ^2 has to decrease by at
least \<tol\> to continue least square fits)

"Mode" specifies the way in which χ^2 is calculated. (χ^2 is what the program minimizes.) If \<mode\> = 1, the actual standard deviation of the data is used; you will be prompted for a file containing the standard deviations. If \<mode\> = 0 each data point is assumed to have the same standard deviation. If \<mode\> = -1, the standard deviation is set to 1/data.

Example:
ELPHIT F1A006.S1 /S F1A006.PRM /T.001 /I25 /P120,990
     Fit file F1A006.S1, which is $\sigma_1(\omega)$, using F1A006.PRM over 120–990 cm^{-1}. \<tol\> = .001; \<max iterations\> = 25

3. MFLFIT

4. TLFIT

5. KEFIT

Similar programs are MFLFIT, TLFIT, and KEFIT, which respectively fit the marginal Fermi liquid formula, the product (coupled oscillator) model, and the Kivelson-Emery model.

6. FLMFIT

FLMFIT—FiLM FIT—fits reflectance, $\mathscr{R}$, or transmittance, $\mathscr{T}$, for thin film multilayers to the Drude-Lorentz model. It can also be used to calculate these functions.

The program uses a Drude-Lorentz model for the complex dielectric function of each layer in the film. It will prompt you for the parameters needed, in groups of three: ω_p, ω_0, and γ. You will also need to specify ϵ_∞ and the thickness.

You first provide the parameters (or a file containing them) for each layer, and are given a chance to change them. This "eyeball fitting" phase of the program can be looped as many times as you wish: the program replots the fitted calculation after *each* set of 3 parameters is entered, i.e., for one oscillator, makes a plot of the result, and then asks about this oscillator immediately again. This allows you to modify the strength, center frequency, and linewidth for each oscillator until you are happy with it. A \<cr\> moves you to the next oscillator in turn, and so on, and finally to ϵ_∞. You may also write the parameter file and the calculated function to disk.

Answering L to the prompt about what to do (E, L, or Q) causes the program to carry out least square adjustment of parameters you specify. You are asked about the mode of the fit, the maximum number of times to loop through the χ^2 minimization process, and the criterion for stopping the fit. Then you are given a chance to "fix" or "free" the parameters. The parameters for each oscillator are presented layer by layer, and you enter "0" to fix them; "1" to free them up. (In the list where they are presented, parameters that are fixed are shown in parentheses. If you want no change, you can enter a "/" or "\<cr\>".)

Switches: /h, /q, and
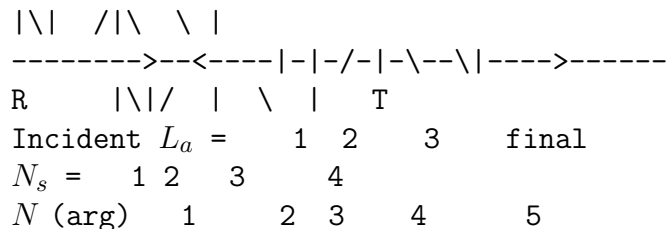   /R   — data are Reflectance   /RPH   — in KK file
     /T   — data are Transmittance

```
/B                   — Back of substrate seen in reflectance
/D                   — there is an .RPH file for the substrate
/Mn                  — Mode:  1 = std dev; 0 = none; −1 = 1/data
/Fn                  — Fraction of parameter used for initial step
/Innn                — maximum number of Iterations in least square fit
/Onnn                — nnn is tOlerance ($\chi^2$ has to decrease by at
                       least <tol> to continue least square fits)
/Xnnn,mmm            — x-axis plot range = nnn–mmm
/Xmmm                — x-axis plot range = 0–mmm
/Ynnn,mmm            — y-axis plot range
/XL, /yl ...         — do axis on Log scale
```

"Mode" specifies the way in which χ^2 is calculated. (χ^2 is what the program minimizes.) If <mode> = 1, the actual standard deviation of the data is used; you will be prompted for a file containing the standard deviations. If <mode> = 0 each data point is assumed to have the same standard deviation. If <mode> = -1, the standard deviation is set to 1/data.

Note: The film thicknesses is passed in Angstrom units (even for the thick substrates) to avoid format problems in the I/O routines of FLMFit. Note further that all parameters except eps_∞ and the thickness are in units of cm^{-1}.

Note 2: The program asks about each layer in sequence, beginning from the top and ending with the substrate (or bottom layer. If there are L_a layers, then $N_s = 1 + L_a$ is the number of interfaces. Moreover, there are $2 + L_a$ distinct media, because the first and last extend to ∞. Here is a picture:

```
|\|  /|\  \ |
-------->--<----|-|-/-|-\--\|---->------
R      |\|/  |  \  |   T
Incident $L_a$ =   1  2    3     final
$N_s$ =   1 2   3      4
$N$ (arg)   1      2  3    4       5
```

Note 3: REFTR2, the subroutine for the thin film transmission, does NOT account for any differences between incident and final medium. If these have different refractive indices, then $\mathscr{T}$ must be multiplied by Real($n(Lamax)/n(1)$).

FLMFit uses a modification of the GRIDLS routine from the book by Bevington.

Examples:
```
FLMFIT F1A006.DAT  F1A006.PRM
```
    Fit file F1A006.DAT, using F1A006.PRM as the parameter file.
```
FLMFIT F1A006.DAT /T.001 /I25
```
    Fit file F1A006.DAT, <tol> = .001; <max iterations> = 25

7. *RelFit*

RelFit—Relative FIT—fits the relative reflectance, $\mathscr{R}(x)/\mathscr{R}_0$, or transmittance, $\mathscr{T}(x)/\mathscr{T}_0$, for thin film multilayers using the Drude-Lorentz model. Here x is some parameter (gate voltage, electrochemical potential, light illumination, perhaps temperature) which changes the reflectance or transmittance from the reference value $\mathscr{R}_0$. RelFit can also be used to calculate these functions.

The program takes one of the layers as the "active" layer, and adjusts the parameters for that layer to optimize the results for the relative response.

The program uses a Drude-Lorentz model for the complex dielectric function of each layer in the film. It needs you to supply the parameters, in groups of three: ω_p, ω_0, and γ. You will also need to specify ϵ_∞ and the thickness for each layer.

For the inactive layers and also for the active layer response that is used as the background, the parameters are input in a `*.prm` file. The active layer parameters for the data, e.g., $\mathscr{R}(x)$, can be input as a file or in response to prompts. This "eyeball fitting" phase of the program can be looped as many times as you wish: the program replots the fitted calculation after *each* set of 3 parameters is entered, i.e., for one oscillator, makes a plot of the result, and then asks about this oscillator immediately again. This allows you to modify the strength, center frequency, and linewidth for each oscillator until you are happy with it. A `<cr>` moves you to the next oscillator in turn, and so on, and finally to ϵ_∞. You may also write the parameter file and the calculated function to disk.

Answering `L` to the prompt about what to do (`E, L,` or `Q`) causes the program to carry out least square adjustment of parameters you specify. You are asked about the mode of the fit, the maximum number of times to loop through the χ^2 minimization process, and the criterion for stopping the fit. Then you are given a chance to "fix" or "free" the parameters. The parameters for each oscillator are presented layer by layer, and you enter "`0`" to fix them; "`1`" to free them up. (In the list where they are presented, parameters that are fixed are shown in parentheses. If you want no change, you can enter a "`/`" or "`<cr>`".)

Only the parameters for the "active" layer are adjusted! Use `flmfit.exe` to optimize the parameters for the inactive layers and for the background conditions of the active layer.

Many experiments give $\Delta\mathscr{T}/\mathscr{T}$ rather than $\mathscr{T}(x)/\mathscr{T}_0$. But $\Delta\mathscr{T}/\mathscr{T} = (\mathscr{T}(x) - \mathscr{T}_0)/\mathscr{T}_0$ and this is the same as $\Delta\mathscr{T}/\mathscr{T} = (\mathscr{T}(x)/\mathscr{T}_0) - 1$. So compute (using `CMP.exe`) $(\Delta\mathscr{T}/\mathscr{T}) + 1 = \mathscr{T}(x)/\mathscr{T}_0$, and proceed.

Switches: `/h`, `/q`, and

`/R`	— data are Reflectance	`/RPH`	— in KK file
`/T`	— data are Transmittance		
`/A`*nnn*	— Layer *nnn* is the active layer		
`/B`	— Back of substrate seen in reflectance		
`/D`	— there is an `.RPH` file for the substrate		
`/M`*n*	— Mode: 1 = std dev; 0 = none; −1 = 1/data		
`/F`*n*	— Fraction of parameter used for initial step		
`/I`*nnn*	— maximum number of Iterations in least square fit		
`/O`*nnn*	— *nnn* is tOlerance (χ^2 has to decrease by at		

least <tol> to continue least square fits)

/X*nnn*,*mmm*	— x-axis plot range = *nnn*–*mmm*
/X*mmm*	— x-axis plot range = 0–*mmm*
/Y*nnn*,*mmm*	— y-axis plot range
/XL, /yl ...	— do axis on Log scale

"Mode" specifies the way in which χ^2 is calculated. (χ^2 is what the program minimizes.) If <mode> = 1, the actual standard deviation of the data is used; you will be prompted for a file containing the standard deviations. If <mode> = 0 each data point is assumed to have the same standard deviation. If <mode> = -1, the standard deviation is set to 1/data.

Note: The film thicknesses is passed in Angstrom units (even for the thick substrates) to avoid format problems in the I/O routines of RelFit. Note further that all parameters except eps_∞ and the thickness are in units of cm^{-1}.

Note 2: The program asks about each layer in sequence, beginning from the top and ending with the substrate (or bottom layer. If there are L_a layers, then $N_s = 1 + L_a$ is the number of interfaces. Moreover, there are $2 + L_a$ distinct media, because the first and last extend to ∞. Here is a picture:

```
|\|   /|\   \ |
-------->--<----|-|-/-|-\--\|---->------
R      |\|/  |  \  |    T
Incident La =   1  2    3     final
Ns =    1 2   3     4
N (arg)   1     2  3    4      5
```
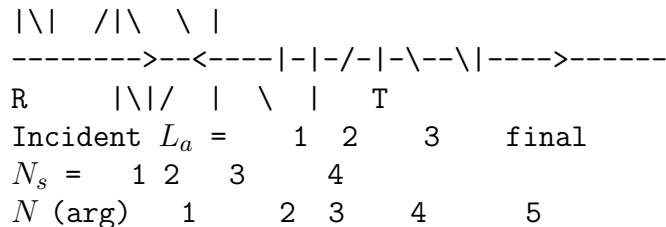
Note 3: REFTR2, the subroutine for the thin film transmission, does NOT account for any differences between incident and final medium. If these have different refractive indices, then $\mathscr{T}$ must be multiplied by Real($n(Lamax)/n(1)$).

RELFIT uses a modification of the GRIDLS routine from the book by Bevington.

Examples:
RELFIT F1A006.DAT F1A006.PRM
    Fit file F1A006.DAT, using F1A006.PRM as the parameter file.
RELFIT F1A006.DAT /T.001 /I25
    Fit file F1A006.DAT, <tol> = .001; <max iterations> = 25

8. THREEFIT

9. SAFIT

Related programs are THREEFIT and SAFIT. ThreeFit displays transmission, reflection, and backside reflectance in three plots. The fits to all are shown, but the least-squares routine fits to the transmittance.

SAfit is a modification for fitting to a thin film on sapphire, taking into account the anisotropy of the substrate.

10. DLCALC

DLCALC—Drude-Lorentz CALC—calculates optical conductivity (σ), dielectric constant (ϵ), Reflectance ($\mathscr{R}$), refractive index (n), absorption coefficient (α) and several other quantities. It also can calculate the function for each component of the model.

The program uses a Drude-Lorentz model for the complex dielectric function. It gets its parameters from a *.PRM file produced by DLFIT or RelFit. It will either ask you for the frequencies at which to do the calculation or use the frequencies in a data file you specify.

Switches: /h, /q, and

/X*nnn*,*mmm*	— x-axis plot range $= nnn$–mmm		
/X*mmm*	— x-axis plot range $= 0$–mmm		
/Y*nnn*,*mmm*	— y-axis plot range		
/XL, /yl ...	— do axis on Log scale	/XR	— linear scale
/V*nnn*	— unit cell Volume is nnn		
/Wp*nnn*	— plasma frequency is nnn		
/RPHin	— Frequencies come from KK file		

The unit cell volume is used for sum rules; the plasma frequency for self-energy and effective mass. *Note:* In calculating the 'components', $\epsilon_\infty = 1.0$ except for $\epsilon_1(\omega)$ itself where $\epsilon_\infty = 0$. This is an issue for quantities that are nonlinear functions of $eps(\omega)$.

Response function switches:

/S1 or /S	— σ_1	/S2	— σ_2
/E1 or /E	— ϵ_1	/E2	— ϵ_2
/N	— refractive index, n	/K	— extinction coefficient, κ
/A	— Absorption coefficient, α		
/R	— Reflectance, $\mathscr{R}$		
/RPHout, /3	— Refl and phase; write *.RPH file		
/L	— Loss function, $-\operatorname{Im} 1/\epsilon$		
/M	— effective Mass, m^*/m		
/T	— $1/\tau(\omega)$		
/I	— $-\operatorname{Imag}$(self-energy), $-\Sigma_2$	/RE	— REal(self-energy), Σ_1
/SU	— SUm rule on $\sigma_1(\omega)$, N_{eff}	/SL	— Sum r. Loss function
/Wps	— superfluid plasma frequency, ω_{ps}		
/LL	— London Length, λ_L		
/SK	— SKin depth, δ		
/F	— Free carrier: ϵ_1 vs $1/\omega^2$		
/D	— Drude model: $1/\sigma_1$ vs ω^2		

Example:

`DLCALC F1A006.S1 /S F1A006.PRM`

Calculate $\sigma_1(\omega)$, using `F1A006.PRM` at the frequencies in `F1A006.S1`.
Produce `F1A006.DLC`, `F1A006.DRU`, `F1A006.L1`, etc.

11. PrmAve

This program averages parameter files from least-squares fitting of optical data.

12. Reseq

This program converts optical constants files from ascending to descending order. (or vice versa.) Use E2E.

Example:
RESEQ PY2.K PY2-R.K $\rightarrow$ Convert PY2.K to descending order.

13. KKold

KKold—Kramers Kronig Transform old version—performs a Kramers-Kronig transform of reflectance $\mathscr{R}$ *and* transmittance $\mathscr{T}$ data. Power-law high-frequency extrapolations are used. Use KK or KKc to take advantage of the x-ray optics high frequency extrapolations.

You will need to specify the extrapolations to be used outside the data. Two parameters control the high frequency behavior: the exponent DE and the crossover frequency WA. Above the maximum frequency in the data file, the reflectance is extrapolated $\omega^{-\text{DE}}$ up to $\omega = $ WA and as ω^{-4} above $\omega = $ WA; transmittance is $1 - \mathscr{R}$. Typical values of DE are 0–2. The book by Wooten recommends a value near 2 but it seems that something between 0.5 and 1.0 works better.

There are several low frequency extrapolations possible, controlled by a parameter COND. For transform of $\mathscr{R}$, one has:

>0	$\Rightarrow$ COND is the known DC conductivity, in $\Omega^{-1}\text{cm}^{-1}$	
=0 or c $\Rightarrow$	Reflectance is assumed constant to DC	
-1 or m $\Rightarrow$	Metallic conductivity assumed:	$\mathscr{R} = 1 - A\omega^{1/2}$
-2 or t $\Rightarrow$	Two-fluid model:	$R = 1 - A\omega^2$
-3 or l $\Rightarrow$	Marginal Fermi Liquid:	$R = 1 - A\omega^1$
-4 or s $\Rightarrow$	Superconducting:	$R = 1 - A\omega^4$
-5 or f $\Rightarrow$	Use a parameter file. (You will be asked for the name of a *.PRM file.)	

For transform of $\mathscr{T}$, one has:

=0 or c $\Rightarrow$	Transmission is assumed constant to DC	
-1 or m $\Rightarrow$	Metallic conductivity assumed:	$\mathscr{T} = B + C\omega^2$
-2 or s $\Rightarrow$	Superconducting:	$T = A\omega^2$

You also need to specify the file names for input or output. The data can be scaled or shifted and are clipped to 0.$\leftrightarrow$1. range. (The actual clipping algorithm allows the data to be 1.0 as long as it is 1.0 at all lower frequencies; otherwise 0.999999 is the largest value allowed.)

After the transform, KKold can plot the optical conductivity and give you a chance to change WA and DE and observe the effect the changes have on the conductivity, if you use the /A switch.

Switches: /h, /q, and

/T	— transform of Transmittance
/TPH	— Transmittance. Write *.TPH file
/F	— ask about Factor, zero to multiply, shift data
/Y*nnn*,*mmm*	— Scale data as $Y \leftarrow nnn * (Y - mmm)$
/W*nnn* or /WA*nnn*	— free-electron behavior starts at WA
/D*nnn* or /DE*nnn*	— exponent in interband region is DE

`/C`*nn* or `/COND`*nn*	— low frequency extrapolation according to `COND`
`/P`$< file >$	— Use $< file >$ for low frequency extrapolation
`/X`*nnn* or `/x`*nnn*	— film thickness (Angstroms) (for $\mathscr{T}$)
`/A`	— Adjust high frequency extrapolation manually
`/Y`*nnn,mmm*	— Scale data as $Y \leftarrow nnn * (Y - mmm)$
`/COL`*i,j*	— Data in Columns i, j.

Examples:

`KKold F1A006.RAW /WA1E6 /D2.2 /C-1 /A`

Transform `f1a006.raw`. Free electron region is 1,000,000 cm^{-1}; $\mathscr{R} = \omega^{-2.2}$ in intermediate region; extend low region as metal. Plot $\sigma_1(\omega)$; allow user to change `DE` and `WA`. (It is more effective to change `DE`.)

`KKold F1A100.rnm -q-c-1`

Transform `f1a100.rnm`. Use built-in defaults, except extend low-frequency region as metal.

Defaults:

`WA` $= 1,000,000$ cm^{-1}

`DE` $= 1.0$

`COND` $= 0$

KKold produces an `RPH` file, where the three data columns are frequency, reflectance, and phase shift on reflectance; ω, $\mathscr{R}$, ϕ. From these, the complex refractive index is $N = 1 + \sqrt{\mathscr{R}}\,e^{i\phi}/1 - \sqrt{\mathscr{R}}\,e^{i\phi}$. The program OP can calculate a wide variety of optical constants from the `RPH` file.

14. KKold /T

KKold with the `/T` switch does the Kramers-Kronig analysis of transmittance.

KKold /T produces an `RPH` file, where the three data columns are frequency, reflectance, and phase shift on reflectance; ω, $\mathscr{R}$, ϕ. From these, the complex refractive index is $N = 1 + \sqrt{\mathscr{R}}\,e^{i\phi}/1 - \sqrt{\mathscr{R}}\,e^{i\phi}$.

Note that the values of $\mathscr{R}$ and ϕ are those of the *same* material measured as single-bounce reflectance.

15. KKnohi

KKnohi—Kramers-Kronig transform—performs a Kramers-Kronig transform for (single-bounce) reflectance $\mathscr{R}$ *and* (slab) transmittance $\mathscr{T}$ data. It does the integral of the data and of the low-frequency extrapolation but does no high-frequency extrapolation.

G. File formats

1. Data files

Data files are in ascii. They begin with a header. The last line of the header is a "/*." Then come the data, in ascii lines, one line per x-y or x-y-z... entry. There are no requirements on locations of the data columns ("free-format").

Most programs will write information to the header. They typically also transfer all non-redundant, nonblank lines from the header of the input file to the header of the output file. The exceptions are the line that tells how many data points are in the file (which is updated in case the number of points is modified) and the column headers, which are changed by programs such as OP and T2A.

The data columns are separated with one or more spaces, or with a comma. If there are only two columns, they are taken as x, y pairs. Otherwise, the user in generally prompted for which columns to read. (If the /RPH switch is used, the first and second columns are taken as x, y; if /PE the first and third.

In a single column file, the x values are the point number, starting with $x = 1$ and continuing to $x = $ NP. You can also answer 0 to the Column no. for X-data? prompt to have the x axis be the point number. (This won't work for two column files. You can use an editor to put a third dummy column in the *first* row of data to trick the column-counting mechanism.)

The file is terminated with the DOS end of file [EOF] marker.

Here is an example:

```
     YBACUO FILM 100K
 0.100000E+07    0.00      -1.00     1.00     0.00
 Freq    Sigma_1
 646 data points
 /*
 91.13    6086.82
 92.04    6095.79
 92.96    6046.8
 93.89    6011.65
 94.83    5981.28
 95.78    5949.78
 96.74    5906.65
 97.71    5849.56
 ...
```

The file was written by OP; OP also wrote the last two header lines, and the *. The second line was written by KKold and the first either by the user or by MAV.

2. The LOF file

Several programs can use a "List of Files" in order to simplify the entry of file names. It contains the names of files in the order in which they will be read. If there are two file names on one line, these will be averaged after they are read in.

A LOF might look like:

```
f1006.s1
f1100A.s1, f1100B.s1 f1100C.s1
f1150A.s1 f1150B.sig
f1200.s1
```

3. Parameter files

DFF, TFF, and DFC write and read parameter (`*.PAR`) files. Here is an example.

```
YBACUO FILM 100K
636 pts. Chisq =     3.921D+03  N*Chisq =      2.447D+06
26 parameters
Ifix,  Parm,   Delta,   Sigma
/*
[Temperature]
100.

[Marginal Fermi Liquid]
1  10000.    0.1        0.
2  1.        0.1        0.
3  1500.     0.1        0.

[Drude-Lorentz]
1  2909.084    290.9084    0.3
3  279.6382    27.9638     0.02
5  189.5232    18.9523     0.05

0  7199.863    719.9863    0.4
0  657.5309    65.7531     0.06
7  1013.892    101.3892    0.2

0  14323.75    1432.375    0.9
0  2126.9      212.69      0.3
0  5562.322    556.2322    0.9
...
2  54329.97    79.307      20.
4  78662.61    29.7171     10.
6  56049.03    589.6533    70.

[Einfinity]
8  1.2345    0.1        0.01
```

The `*.PAR` file starts with a header, some of which comes from the data file being fitted. DFF or TFF also write some information in the header, such as the χ^2. The header ends with a `/*`.

The data are organized in blocks, in (`[...]`), telling the program which of the models the parameters in the block are intended for. Here are the blocks understood by the programs:

1. [`Temperature`]. "TEM' or "TMP" are OK.*
2. [`Thickness (AA)`]. (in Å units). "THI" or "THK" are OK.
3. [`Substrate Thickness (cm)`] (in cm). "SUB" or "DCM" are OK.
4. [`Einfinity`]. "EIN," "EPS," or "INF" are OK
5. [`Drude-Lorentz`]. "DRU†" or "LOR" are OK.
6. [`Product DLF`]. "'PRO" or "LST" are OK.
7. [`Direct band gap`]. "DIR" or "DBG" are OK.
8. [`Indirect band gap`]. "IND" or "IBG" are OK.
9. [`Marginal Fermi Liquid`]. "MAR" or "MFL" are OK.
10. [`Rice EMV`]. "RIC" or "EMV" are OK.
11. [`Kivelson-Emery`] (for DFF and DFC, not TFF). "KIV" or "K-E" are OK .

Moreover, case does not matter: [`LOR`] and [`lor`] are equivalent.

The last header line (`Ifix, Parm, Delta, Sigma`) identifies the four columns for each parameter. The first column is an integer which controls the order in which the fitting routine varies the parameter. A zero entry means not to adjust this parameter. The second column is the parameter; the third the step by which the parameter initially is varied; and the last is supposedly an estimate of the uncertainty of the parameter.

See pages 39–44 for the order of the parameters.

It is OK (and helpful) to leave a blank line between the three or four parameters that represent a single term in $\epsilon(\omega)$. The blank lines are not required.

If you are typing up a parameter file for the first time you can simplify to a single column format for the parameters. Here is an example.

```
Silver
/*
[Drude]
70000
0
300

[Eps infty]
4.5
```

---

* Strictly speaking, the temperature is not a parameter; it is not adjusted to optimize the fit. It is required by some models, however.

† Note that although it says "Drude" you must enter three parameters, with the second one—which would be the resonant frequency in the Lorentz model—equal to zero.

4. Oldstyle Parameter files

DLFit, FLMFit, and cousins write and read parameter (*.PRM or *.MFL) files. Here is an example.

```
         19          6
 1     10242.1600     8720.3090         .1E+02
 0          .0000          .0000         .0
 3         1.3191         1.0843         .9E-03

 4      2671.4760       43.1593         .3
 5       264.8839        1.6720         .2E-01
 6       146.6709        3.8451         .4E-01
...
18      6353.1720      250.0000          .2E+02

 0         1.0000          .1000         .0
YBACUO FILM 6K
DLFIT parameters
for least-sq fit to data in f1006.s1
637 pts. Chisq =    8.269D+03  N*Chisq =     5.127D+06
```

The first line has the number of parameters and the number of oscillators in the dielectric function model. Then come the individual parameters, one per line in the order ω_p, ω_0, and γ. (In MFLFIT, the *first** set of 3 are the plasma frequency, coupling constant λ, and the cutoff frequency for the conduction carriers.) There is an (optional) blank line between the three parameters that describe each oscillator. At the end of the parameter list file are lines for ϵ_∞ and (if for FLMFit) film thickness.

In each line, the first column is an integer which controls the order in which the fitting routine varies the parameter. A zero entry means not to adjust this parameter. The second column is the parameter, the third the step by which the parameter initially is varied, and the last is supposedly an estimate of the uncertainty of the parameter.

After the last parameter is a "trailer" containing information from the header of the file that was fitted and the statistics of the fit. This is written by the program but does not need to be entered if you are typing the file.

DLFIT can also use two relaxed versions of the parameter file. In one, the third and fourth columns can be omitted; in the other, the first, third and fourth columns can be omitted.

5. File extensions

ADD	CMP	sum file, $A + B$
ABS	CMP	Absolute value $|y|$
AE	OP, DFC	$|\epsilon|$
ALP	OP, DFC, T2A	Absorption coefficient, $\alpha(\omega)$

---

* This is a change. Prior to version 3.1 the last set were the MFL parameters.

AS	OP, DFC	$|\sigma|$
ATN	OP, DFC	$|\tan(\delta)|$
AVE	CMP	Average
BRG	Bridge	Two data sets joined by a bridge
CLP	CMP	Clip y to band
CMB	CMP	two two-column files $\rightarrow$ one 3 column
D	DERIV	Derivative
DIF	CMP	Comparison file, $A - B$
DIV	CMP	Comparison file, A/B
DLC	DLCALC	Results from Drude-Lorentz calculation
DRU	2C, DLCALC	Drude conductivity
DRU	RDRUDE	Drude reflectance
DSL	OP, DFC	$1/\sigma_1$ vs ω^2
E1	OP, DFC	Real diel. function, $\epsilon_1(\omega)$
E2	OP, DFC	Imag. diel. function, $\epsilon_2(\omega)$
E12	RPH2E	Table of frequency, $\epsilon_1(\omega)$, $\epsilon_2(\omega)$
EMA	EMA	EMA calculation
EYE	DFF, TFF	Results of "eyeball" fit
F	J2W	Output file translated from JCAMP.
FDX	J2W	Output file translated from JCAMP.
FIT	DLFIT, FLMFIT	Results of "least-square" fit
FLP	XY2YX	File with columns interchanged
FRE	OP, DFC	ϵ_1 vs $1/\omega^2$
FTP	FTS, FTgram	Fourier transform of time series
G1	OP, DFC	Real memory function, $\gamma_1(\omega)$
G2	OP, DFC	Imag. memory function, $\gamma_2(\omega)$
INV	CMP	$1./y_{data}$
IP	INTERP	Interpolated file
K	OP, DFC, T2K	Extinction coefficient
LAM	OP, DFC	mass enhancement factor, λ
LOF	Several programs	List of files
LON	OP, DFC	London length, λ_L
LOS	OP, DFC	Loss function, $-\operatorname{Im}(1/\epsilon)$
Ln	DLCALC	Lorentzian n response fn
MFL	MFLFIT	Parameter file
MIA	2C	Midinfrared average
MIR	2C	Midinfrared conductivity
MRG	MAV	Merge and average file
MST	OP, DFC	Effective mass ratio, m^*/m
MUL	CMP	Product file, $A * B$
N	OP, DFC	Refractive index
NFR	OP	N_{eff} (free electrons)
NK	RPH2NK	Frequency, refractive index, extinction coefficient
PAR	DFF, TFF	Parameter file

PAT	MEND	Data file with "patch"
PIK	PIK	Piece of data
PKS	FTgram	List of peaks from Fourier analysis
PKY	Pinkify	File with added pink noise
PRM	FLMFIT, DLFIT	Parameter file
R	OP, DFC	Reflectance
RFP	OP , DFC	Coherent (Fabry-Perot) reflectance
RHO	OP, DFC	Optical resistivity ρ_1
RH2	OP, DFC	Imaginary optical resistivity ρ_2
RNM	RNM	Renormalized data
RNU	Ranu	Random-number time series
RPH	KK	Kramers-Kronig output(reflectance, phase)
RSB	OP , DFC	Incoherent slab reflectance
S1	OP, DFC	Optical conductivity, $\sigma_1(\omega)$
S2	OP, DFC	Imag. conductivity, $\sigma_2(\omega)$
S12	RPH2S	Frequency, $\sigma_1(\omega)$, $\sigma_2(\omega)$
SCA	CMP	Scaled and/or shifted file. ($\texttt{<file>} \times a + b$)
SE1	OP, DFC	Real self-energy, $\Sigma_1(\omega)$
SE2	OP, DFC	$-$Imag. self-energy, $-\Sigma_2(\omega)$
SKN	OP, DFC	Skin depth, δ
SLO	OP, DFC	Surface loss function
SMO	FTS	Fourier-transform smooth
SOL	OP	Sum rule on loss function
SQR	CMP	$\sqrt{(y)}$
SSL	OP , DFC	Sum rule, surface loss function
SUM	OP, DFC	Sum rule on $\sigma_1(\omega)$, $N_{eff}(\omega)$
SUM	CMP	Sum of data, $A + B$
TAN	OP, DFC	$\tan(\delta)$
TAU	OP, DFC	Scattering rate, $1/\tau(\omega)$
TPH	KK	Kramers-Kronig output (transmittance, phase) (obsolete)
TRA	OP , DFC	Incoherent slab transmittance
TFP	OP , DFC	Coherent (Fabry-Perot) transmittance
WPS	OP, DFC	Superfl. plasma freq., ω_{ps}
XPO	CMP	$x \to x^p$
XRO	XRO	X-ray reflectance
XRP	XRO	X-ray reflectance and phase
XTR	XRO	X-ray transmittance
YPO	CMP	$y \to y^p$
ZCR	Tcalc	Zero crossing list
1-Y	CMP	$1.0 - y_{data}$

H. Programming notes

The programs are written in Intel visual FORTRAN and use the quick-win graphics. The Kramers-Kronig (reflectance) code was written by Claus Jacobsen. The least-squares minimization uses a routine by Bevington. Most of the other important routines were written by Charles Porter; the rest is by David Tanner.

If you find errors or if things crash suddenly, send an email message to David Tanner at `tanner@phys.ufl.edu`. Describe the problem, send screen shots, and—if you can—send also an example of the input files that were used.