# Lab Programs

Optical Properties

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

*Gainesville, 3 November 2008*

TABLE OF CONTENTS

## A. Introduction

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

This manual is for the Windows version of the programs. (Earlier versions were written for "DOS," i. e., 16-bit, EGA graphics. Beginning with NT, windows, and hardware, has become less and less compatible with that older operating system. So they 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, etc. that we've come to know and love.)

**There is a bug in all of the quick-Win programs.** Sometimes when they start, a "QuickWin Error" message box appears with a message something like

Internal Error - unexpected error
file "E:\forrtl\build\qwnbuild\qwgswc.c", line 1051."

or

Out of Memory - Quick Win Error
file "E:\forrtl\build\qwnbuild\qwgsvid.c", line 593."

There are several versions of this message. *They are due to a compiler bug and not to a bug in the program.* They seem to be more common on computers with an excessive number of fonts installed. Just click OK and proceed. The worst that can happen is that the graphs will not be annotated or that the graphics font from the graphs will be used in the dialogs. Or type control-C and start again.

### *1. List of programs*

Here are the programs:

| | | | |
|---|---|---|---|
| cmp.exe | gtfc.exe | opc.exe | rph2nk.exe |
| cmpc.exe | interp.exe | patch.exe | rph2s.exe |
| deriv.exe | interpc.exe | pfit.exe | s2rph.exe |
| dlcalc.exe | j2w.exe | pik.exe | safit.exe |
| dlfit.exe | kefit.exe | prmave.exe | t2a.exe |
| e2e.exe | kk.exe | q.exe | tcalc.exe |
| e2rph.exe | kkc.exe | ql.exe | threefit.exe |
| elphit.exe | l2rph.exe | qp.exe | tlfit.exe |
| ema.exe | mav.exe | rdrude.exe | 2c.exe |
| flmfit.exe | mflfit.exe | relfit.exe | w2l.exe |
| fts.exe | nk2rph.exe | rnm.exe | xy2yx.exe |
| gtf.exe | op.exe | rph2e.exe | z2w.exe |

### *2. Installation and use*

Put the programs in a directory on your path. I use `D:\datan\`. If this is not on the path, add it. (Google "path windows xp" for instructions on how to. First up at the time of writing is `http://www.computerhope.com/issues/ch000549.htm`)

Run the programs from a command window. In Windows NT and XP, the command window is CMD.EXE. I suggest adding a desktop link to this program. You can add links on the desktop to the data analysis programs as well, and most will accept drag/drop from windows explorer.

A better command processor is available from JP Software (`http://www.jpsoft.com/`). This program started as 4DOS for use in the 16bit DOS world. The 32 bit version (Win98/NT/XP) used to be called 4NT. The present version is TCC/LE, and is free to all.

### *3. Common features*

The programs have several things in common. All have switches that control their behavior. You can pass filenames and switches on the command line or have the program prompt you for them. 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 /xlog /a` or `qp f1006.s1 -XLOG-A` to plot the file named `f1006.s1` with autoscaling with the x-axis on a log scale.

The programs can handle long file names *except those with a space in them.* Thus, "`Data.300k.first.refl`" would be OK, and the programs will take "`refl`' as the extension (possibly to be replaced in the output file by "`RPH`" or "`SMO`." But "`Data 300k first.refl`" will be interpreted as three files. Rename it to "`Data_300k_first.refl`." A space in a file name is a perverse concept, anyway.

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

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, data fitting 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. 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.

The data are produced as follows:

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

$Y_{data}$ = sample; $Y_{ref}$ = file to modify $Y_{data}$; <action> any of `+`, `-`, `/`, and `*`.

*or*

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

$Y_{data}$ = sample; <action> = one of `1.0-`, `1.0/`, or a rescale.

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 `/X`$A, B$ or `/Y`$A, B$ 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 `/X`$A, B$ 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. |
| `/A` | —average 2 or more files. |
| `/R` | —square root of $Y_{data}$. |
| `/U` | —$1. - Y_{data}$. |
| `/I` | —Invert: $1/Y_{data}$. |
| `/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 $x \Leftarrow x \times A + B$. |
| `/S` | —sum (integrate) data. Note: Change from earlier, when it meant scale. (Use `/y` and `/x`) |
| `/2` | —use $x$-data of 2nd file. |
| `/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`

## *2. Deriv*

This program calculates the derivative of 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. |

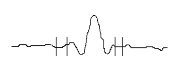
### *3. FTS*

FTS—Fourier Transform Smooth—is a program for smoothing and fringe elimination by Fourier transforming data and filtering the transform. The basic idea 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 high-frequency part of the transform, the noise can be reduced.

Fringes in the spectrum have a sinusoidal pattern, and so are at well-defined locations in the 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.

To remove fringes, click twice to left of signature and twice to right of it:



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 in, a second time where the amplitude of the transform has gotten to its high-frequency value, and twice to the right 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. If you answer '`R`' at the prompt, you can retry with the original data. If you answer '`M`,' you can continue with additional smoothing.

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

| | |
|---|---|
| `/O or /o` | —Order of polynomial to fit, e. g., `/O3` ⇒ cubic. |
| `/C`*nnn*`,`*nnn* `or /y`*nnn*`,`*nnn* | —Clip data outside range supplied. |

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.

### *4. 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. Algorithm 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 $\sigma_1$ and $\sigma_2$ and then for data files for $\mathscr{T}$, $\mathscr{R}$, and the substrate. The latter may be a `*.nk`, `*.rph`, or `*.prm` file. `*.nk` is assumed if another extension is used. The output is a `*.s12` file. (This may be converted to `*.rph` with `s2rph.exe`.)

You can provide a list of files instead of responding to the prompt. 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 or /?` | —get help. |
| `/Q` | —Quick: fewer questions /V —Verbose: more. |
| `/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)

### *5. 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

| | |
|---|---|
| `/X`*nnn*`,`*mmm* | —$x$-axis plot range = *nnn*–*mmm** |
| | —*also* the interpolation range. |
| `/X`*mmm* | —$x$-axis plot range = 0–*mmm** |
| | —*also* the interpolation range. |
| `/C`*nn* | —$x$ increment is *nn* |
| `/L` | —do interpolation Logarithmically |
| `/Y`*nnn*`,`*mmm* `or /Y`*mmm* | —$y$-axis plot range* |
| `/XL, /yl ...` | —do axis on Log scale* |
| `/N` | —No rescaling |
| `/S or /Sx` | —reScale $x$ axis data |

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 $x \Leftarrow A * x + B$.

### *6. KK and KKc*

KK—Kramers Kronig Transform—performs a Kramers-Kronig transform program for reflectance $\mathscr{R}$ *and* transmittance $\mathscr{T}$ data. KKc is the console (no graphics) version.

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 $\omega^{-\mathtt{DE}}$ up to $\omega = \mathtt{WA}$ and as $\omega^{-4}$ above $\omega = \mathtt{WA}$; 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 KK of $\mathscr{R}$, one has:

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

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

| | | | |
|---|---|---|---|
| =0 or c | ⇒ | Transmission is assumed constant to DC | |
| -1 or m | ⇒ | Metallic conductivity assumed: | $\mathscr{T} = B + C\omega^2$ |
| -2 or s | ⇒ | 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.↔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, KK 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 or /t` | —transform of Transmittance |
| `/F or /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 < *fiile* > for low frequency extrapolation |
| `/X`*nnn* `or /x`*nnn* | —film thickness (Angstroms) (for $\mathscr{T}$) |
| `/A` | —Adjust high frequency extrapolation manually |

Examples:

```
KK 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`.)

`KK 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

KK produces an `RPH` file, where the three data columns are frequency, reflectance, and phase shift on reflectance; $\omega$, $\mathscr{R}$, $\phi$. 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.

## *7. KK /T*

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

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

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

## *8. 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 multilies 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 averagethat 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 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. |
| `/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.

## *9. OP and OPc*

OP—Optical Properties—computes Optical Properties from KK-determined reflectance and phase. 24 different functions may be calculated. The program can also subtract a model dielectric functon from the data.

OPc is the console version.

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

| | |
|---|---|
| `/V`*nnn* | —unit cell Volume is *nnn* |
| `/Wp`*nnn* | —Plasma frequency is *nnn* |
| `/W`*nnn*, *mmm* | —calculate between frequencies *nnn* and *mmm* |
| `/W`*mmm* | —calculate from 0.0 to maximum frequency *mmm* |
| `/ALL or /all` | —do ALL functions (19 files created) |
| `/P<file>` | —Partial dielectric function, $\epsilon - \epsilon_{DL}$ |

The unit cell volume is needed for sum rules. The plasma frequency is used for self-energy and effective mass. Max frequency (`/Wm`*nnn*) can be used to limit file size.

The switch for "Partial dielectric function" subtracts a Drude-Lorentz model from $<\epsilon(\omega)>$ before calculation of the optical function. It either uses parameters in a `*.PRM` file produced by DLFIT or asks the user for these parameters.

Response function switches:

| | | | |
|---|---|---|---|
| `/S1 or /S` | —$\sigma_1$ | `/S2` | —$\sigma_2$ |
| `/E1 or /E` | —$\epsilon_1$ | `/E2` | —$\epsilon_2$ |
| `/N` | —refractive index, $n$ | `/K` | —extinction coefficient, $\kappa$ |
| `/A` | —Absorption coefficient, $\alpha$ | | |
| `/R` | —Reflectance, $\mathscr{R}$ | `/RPH` | —$\mathscr{R}$ is in file from KK |
| `/L` | —Loss function, $-\operatorname{Im} 1/\epsilon$ | | |
| `/M` | —effective Mass, $m^*/m$ | | |
| `/T` | —$1/\tau(\omega)$ | | |
| `/I` | —$-$Imag(self-energy), $-\Sigma_2$ | `/RE` | —REal(self-energy), $\Sigma_1$ |
| `/SU` | —SUm rule on $\sigma_1(\omega)$, $N_{eff}$ | `/SL` | —Sum r. Loss function |
| `/Wps` | —superfluid plasma frequency, $\omega_{ps}$ | | |
| `/LL` | —London Length, $\lambda_L$ | | |
| `/SK` | —SKin depth, $\delta$ | | |
| `/F` | —Free carrier: $\epsilon_1$ vs $1/\omega^2$ | | |
| `/D` | —Drude model: $1/\sigma_1$ vs $\omega^2$ | | |

Examples:

`OP F1A006 /S1`

Calculate conductivity ($\sigma_1$) 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.

## *10. PATCH*

PATCH is a program to "patch" a data file.

PATCH 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) |
| `/N` | —No rescaling |
| `/S or /Sx` | —ask about reScale of $x$-axis data |
| `/C` | —Change $x$ increment in output data |
| `/D`*n* | —Degree of fit is $n$ (0–4) |

Examples:

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

In the first, PATCH 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 $x \Leftarrow A \times x + B$.

### *11. PIK*

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 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 $mmm$ |
| `/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 |
| `/N` | —No rescaling |
| `/S or /Sx` | —ask about reScale of $x$ axis data |

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 $x \Leftarrow A \times x + B$.

### *12. 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

| | |
|---|---|
| `/X`*nnn*, *mmm* | —$x$-axis plot range = ELPHIT–*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 /L` | —both $x$ and $y$ on Rectalinear or Log 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) |
| `/Ci,j` | —data are in Cols i and j |
| `/P or /PE` | —data are '`PE`,' cols 1 and 3 |

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:

### *13. 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:

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

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

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.

## *14. T2A*

T2A—Transmittance to Alpha—computes absorption coefficient from transmittance. It can take into account the reflectance either from a file or a value of the refractive index.

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

| | |
|---|---|
| `/T`*nnn* `or /D`*nnn* | —Thickness is *nnn* |
| `/P or /PE` | —PE file |
| `/R<file>` | —Reflectance is in `<file>` |
| `/N`*nnn* | —approximate reflectance using refractive index *nnn* |

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

### *15. TwoC (2c)*

2C—Two Component—is a program makes a two-component decomposition of optical conductivity $\sigma_1(\omega)$ into Drude and midinfrared components. Algorithm 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.

## C. Fitting routines and computational routines

These routines are for fitting data to models. The two most important are DLFit and FLMFit

DLFIt fits optical conductivity data, single-bounce reflectance, dielectric function, etc. to a Drude-Lorentz model dielectric function. Other models are in TLFit, KEfit, and ELPHit.

FLMfit fits to reflectance or transmittance data of multilayer thin films on a substrate. SAfit and ThreeFit are variations.

See page 38 for the file format of the "parameter" file (`*.PRM`) produced and used by these programs.

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

## *1. DLFIT*

DLFIT—Drude-Lorentz FIT—makes a nonlinear least-square fit of optical conductivity ($\sigma_1$), dielectric constant ($\epsilon_1$), Reflectance ($\mathscr{R}$), refractive index ($n$), absorption coefficient ($\alpha$) 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: $\omega_p$, $\omega_0$, and $\gamma$. Unless you are doing optical conductivity, you will also be asked to specify $\epsilon_\infty$.

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 paramter 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 modifiy 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 $\epsilon_\infty$.

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 $\chi^2$ 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 ($\chi^2$ has to decrease by at least <tol> to continue least square fits) | | |

"Mode" specifies the way in which $\chi^2$ is calculated. ($\chi^2$ 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 $\mathrm{cm}^{-1}$. <tol> = .001; <max iterations> = 25

### *2. MFLFIT, TLFIT, and 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.

### *3. ELPHIT*

ELPHIT—ELectron-PHonon fIT—makes a nonlinear least-square fit of optical conductivity ($\sigma_1$), dielectric constant ($\epsilon_1$), Reflectance ($\mathscr{R}$), refractive index ($n$), absorption coefficient ($\alpha$) 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: $\lambda_\alpha$, $\omega_\alpha$ and $\gamma_\alpha$ ; $\omega_p$, $\omega\_0$ and $\gamma$; also $\epsilon_\infty$.

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 modifiy 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 $\epsilon_\infty$.

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 $\chi^2$ 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 ($\chi^2$ has to decrease by at least <tol> to continue least square fits) | | |

"Mode" specifies the way in which $\chi^2$ is calculated. ($\chi^2$ 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 $\mathrm{cm}^{-1}$. <tol> = .001; <max iterations> = 25

## *4. 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: $\omega_p$, $\omega_0$, and $\gamma$. You will also need to specify $\epsilon_\infty$ 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 modifiy 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 $\epsilon_\infty$. You may also write the paramter 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 $\chi^2$ 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 | | |
| `/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 ($\chi^2$ 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 $\chi^2$ is calculated. ($\chi^2$ 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 $eps_\infty$ and the thickness are in units of $\mathrm{cm}^{-1}$.

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 $\infty$. 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 $\mathrm{Real}(n(Lamax)/n(1))$.

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

## *5. 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 illuminaiton, 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 optimise 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: $\omega_p$, $\omega_0$, and $\gamma$. You will also need to specify $\epsilon_\infty$ 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 modifiy 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 $\epsilon_\infty$. You may also write the paramter 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 $\chi^2$ 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 optimise 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 ($\chi^2$ 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 $\chi^2$ is calculated. ($\chi^2$ 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 $eps_\infty$ and the thickness are in units of $\mathrm{cm}^{-1}$.

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 $\infty$. 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($n(Lamax)/n(1)$).

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

### *6. THREEFIT and 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.

## *7. PFIT*

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

The data are fitted to a polynomial in $x$. An arbitrary 160 points from the fitted curve over the data range are output.

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

| | |
|---|---|
| `/O or /o` | —Order of polynomial to fit, eg, /O3==cubic. |
| `/X`*nnn*, *mmm* `or /x`*mmm* | —Compute over x range *nnn*–*mmm* or 0–*mmm*. |
| `/CY`*nnn*, *mmm* `or /y`*nnn*, *mmm* | —Clip data to range *nnn*–*mmm* before fit. |
| `/Ci,j` | —data are in Cols i and j |

Default polynomial order: 1 (linear, with intercept).

Maximum polynomial order: 4 (quartic, with intercept).

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

Example:

```
PFit osc.str osc.fit /col3,1 /o3
```

Here, `osc.str` is the input file and `osc.fit` is the output.

## *8. DLCALC*

DLCALC—Drude-Lorentz CALC—calculates optical conductivity ($\sigma$), dielectric constant ($\epsilon$), Reflectance ($\mathscr{R}$), refractive index ($n$), absorption coefficient ($\alpha$) 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 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

| | | | |
|---|---|---|---|
| `/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 $eps(\omega)$, like $n$, $k$, etc.

Response function switches:

| | | | |
|---|---|---|---|
| `/S1 or /S` | —$\sigma_1$ | `/S2` | —$\sigma_2$ |
| `/E1 or /E` | —$\epsilon_1$ | `/E2` | —$\epsilon_2$ |
| `/N` | —refractive index, $n$ | `/K` | —extinction coefficient, $\kappa$ |
| `/A` | —Absorption coefficient, $\alpha$ | | |
| `/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` | —−Imag(self-energy), $-\Sigma_2$ | `/RE` | —REal(self-energy), $\Sigma_1$ |
| `/SU` | —SUm rule on $\sigma_1(\omega)$, $N_{\mathit{eff}}$ | `/SL` | —Sum r. Loss function |
| `/Wps` | —superfluid plasma frequency, $\omega_{ps}$ | | |
| `/LL` | —London Length, $\lambda_L$ | | |
| `/SK` | —SKin depth, $\delta$ | | |
| `/F` | —Free carrier: $\epsilon_1$ vs $1/\omega^2$ | | |
| `/D` | —Drude model: $1/\sigma_1$ vs $\omega^2$ | | |

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.

## *9. EMA*

This program uses the Bruggeman Effective Medium Approximation to calculate the dielectric function, conductivity, reflectance, refractive index, absorption coeff, 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 |
| `/F`*nnn* | — Fill fraction is *nnn* |
| `/D`*nnn* | — Depolarization factor is *nnn* |
| `/V`*nnn* | — Unit cell volume is *nnn* |
| `/Wp`*nnn* | — Plasma frequency is *nnn* |

Response function:

| | | | |
|---|---|---|---|
| `/S1 or /S` | — $\sigma_1$ | `/S2` | —$\sigma_2$ |
| `/E1 or /E` | — $\epsilon_1$ | `/E2` | —$\epsilon_2$ |
| `/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` | —$\Gamma_2$ |
| `/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($\epsilon$) | `/|S|` | —Abs($\sigma$) |
| `/TAN` | — Tan($\delta$) | `/|TAN|` | —Abs(Tan $\delta$) |

### *10. 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* |

## *11. 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. '

## D. 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 $\epsilon_1$ and $\epsilon_2$ 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` | $\omega$, Re$(1/\epsilon)$, 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, $\epsilon - \epsilon_{DL}$

Max frequency (`/Wm`*nnn*) can be used to limit file size.

The $\epsilon_\infty$ switch allows $\sigma_2$ 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. It only exists in S2RPH.

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 DLFIT or prompts for user responses. This "Partial" switch only exists in RPH2E and RPH2S.

Example:
`RPH2E F1A006`
Calculate $\epsilon_1$ and $\epsilon_2$ from file F1A006.RPH

### *8. XY2YX*

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

Switches: `/h` and `/q`.

Example:

`XY2YX  Latt.T temper.lat`

Exchange data columns in Latt.T; output in temper.lat.

### *9. E2E*

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

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

| | |
|---|---|
| `/T or /3` | —there are 3 Columns of |
| `/Ci,j` | —data are in Columns i and j |

Example:

`E2E PY2.K PY2-R.K`

Convert PY2.K to decending order.

### *10. J2W*

J2W—Jcamp to Wavenumber—converts JCAMP-DX files produced by commercial IR machines to a table of wavenumber, data values.

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, returns sets the date/time of the converted file 2 sec after that of the original JCAMP-DX file. It requires 4dos, and uses the internal "touch" function.

### *11. 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 decending 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.

### *12. Z2W*

Z2W—Zeiss to Wavenumber—is a program that converts an optical properties file from wavelength to wavenumber (nm $\rightarrow$ cm$^{-1}$) and—at the same time—percents to fractions. The program is designed for Zeiss MPM data.

Switches: `/h` and `/q`.

Example:
`Z2W PY2.K PY2-L.K`
Convert PY2.K to wavenumber.

### *13. PrmAve*

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

## E. File formats

### *1. Data files*

Data files are in ascii. They begin with a header. The last line of the header is a "/*." Then comes the data, in 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-couniting 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 KK and the first either by the user or by MAV.

### 2. *Parameter files*

DLFIT, MFLFIT, and FLMFIT 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 $\omega_p$, $\omega_0$, and $\gamma$. (In MFLFIT, the *first** set of 3 are the plasma frequency, coupling constant $\lambda$, 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 $\epsilon_\infty$ 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 and MFLFIT can also use two relaxed versions of the paramter file. In one, the third and fourth columns can be omitted; in the other, the first, third and fourth columns can be omitted.

### 3. *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.

* This is a change in version 3.1 and later; previously the last set were the MFL parameters.

A LOF might look like:
```
f1006.s1
f1100A.s1, f1100B.s1 f1100C.s1
f1150A.s1 f1150B.sig
f1200.s1
```

## *4. File extensions*

| | | |
|---|---|---|
| `ADD` | `CMP` | sum file, $A + B$ |
| `ABS` | `CMP` | Absolute value $\lvert y\rvert$ |
| `AE` | `OP` | $\lvert\epsilon\rvert$ |
| `ALP` | `OP and T2A` | Absorption coefficient, $\alpha(\omega)$ |
| `AS` | `OP` | $\lvert\sigma\rvert$ |
| `ATN` | `OP` | $\lvert\tan(\delta)\rvert$ |
| `AVE` | `CMP` | Average |
| `CLP` | `CMP` | Clip $y$ to band |
| `CMB` | `CMP` | two two-column files → 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` | $1/\sigma_1$ vs $\omega^2$ |
| `E1` | `OP` | Real diel. function, $\epsilon_1(\omega)$ |
| `E2` | `OP` | Imag. diel. function, $\epsilon_2(\omega)$ |
| `E12` | `RPH2E` | Table of frequency, $\epsilon_1(\omega)$, $\epsilon_2(\omega)$ |
| `EMA` | `EMA` | EMA calculation |
| `EYE` | `DLFIT, FLMFIT` | Results of “eyeball” fit |
| `FIT` | `DLFIT, FLMFIT` | Results of “least-square” fit |
| `FLP` | `XY2YX` | File with columns interchanged |
| `FRE` | `OP` | $\epsilon_1$ vs $1/\omega^2$ |
| `INV` | `CMP` | $1./y_{data}$ |
| `IP` | `INTERP` | Interpolated file |
| `K` | `OP` | Extinction coefficient |
| `LAM` | `OP` | London length, $\lambda_L$ |
| `LOF` | `Several programs` | List of files |
| `LOS` | `OP` | Loss function, $-\,\mathrm{Im}(1/\epsilon)$ |
| L$n$ | `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` | Effective mass ratio, $m^*/m$ |
| `MUL` | `CMP` | Product file, $A * B$ |

| | | |
|---|---|---|
| N | OP | Refractive index |
| NFR | OP | $N_{eff}$ (free elctrons) |
| NK | RPH2NK | Frequency, refractive index, extinction coefficient |
| PAT | PATCH | Data file with "patch" |
| PIK | PIK | Piece of data |
| PRM | FLMFIT, DLFIT | Parameter file |
| R | OP, DLCALC | Reflectance |
| RNM | RNM | Renormalized data |
| RPH | KK | Kramers-Kronig output |
| S1 | OP | Optical conductivity, $\sigma_1(\omega)$ |
| S2 | OP | Imag. conductivity, $\sigma_2(\omega)$ |
| S12 | RPH2S | Frequency, $\sigma_1(\omega)$, $\sigma_2(\omega)$ |
| SCA | CMP | Scaled and/or shifted file. (<file>$\times a + b$) |
| SE1 | OP | Real self-energy, $\Sigma_1(\omega)$ |
| SE2 | OP | $-$Imag. self-energy, $-\Sigma_2(\omega)$ |
| SKN | OP | Skin depth, $\delta$ |
| SLO | OP | Surface loss function |
| SMO | FTS | Fourier-transform smooth |
| SOL | OP | Sum rule on loss function |
| SQR | CMP | $\sqrt{(y)}$ |
| SSL | OP , DLCALC | Sum rule, surface loss function |
| SUM | OP | Sum rule on $\sigma_1(\omega)$, $N_{eff}(\omega)$ |
| SUM | CMP | Sum of data, $A + B$ |
| TAN | OP | $\tan(\delta)$ |
| TAU | OP | Scattering rate, $1/\tau(\omega)$ |
| WPS | OP | Superfl. plasma freq., $\omega_{ps}$ |
| XPO | CMP | $x \rightarrow x^p$ |
| YPO | CMP | $y \rightarrow y^p$ |
| 1-Y | CMP | $1.0 - y_{data}$ |

## F. Programming notes

The programs are written in Compaq visual FORTRAN and use the quick-win graphics. Most of the routines were written by Charles Porter, the rest by David Tanner.