Table of Contents

License

We request that any results obtained through the use of *CC-FIT2* are accompanied by the following reference:

D. Reta and N. F. Chilton, *Phys. Chem. Chem. Phys.,* 2019, **21**, 23567.

1. Introduction

The *CC-FIT2* program is a simple tool for the *i)* fitting of AC susceptibilities using the (generalised) Debye model, *ii)* extraction of AC susceptibilities from waveform data, *iii)* fitting of DC magnetisation decays using (stretched) exponentials, *iv)* extraction of magnetic relaxation times with associated uncertainties and *v)* fitting the temperature dependence of these data accosting for uncertainties in the underlying relaxation times.

This manual does not contain a comprehensive overview of the theory of relaxation dynamics; please consult *Molecular Nanomagents* (Gatteschi, Sessoli and Villain, 2006, Oxford University Press) and the publication associated with this program (Reta and Chilton, *Phys. Chem. Chem. Phys.*, 2019, **21**, 23567).

Python library dependencies.

CC-FIT2.py has been successfully tested in Python 3.5, 3.6 and 3.7 – the program relies on the following Python libraries:

- Numpy
- Matplotlib
- Scipy
- Requests
- Argparse
- Os
- Glob
- Itertools
- Collections
- Warnings

All these are normally included with Anaconda distributions and as such the program should be ready to use without prior library installation. However, if you have your own Python installation and are missing some of the libraries, we recommend you use pip to install them.

Program's workflow.

There are two versions of *CC-FIT2*; the first is a standalone executable (CC-FIT2.exe) and the second is a Python script (CC-FIT2.py). Depending on the data to be fitted, the user can choose the most convenient version (Figure 1). However, note that CC-FIT2.py offers more functionality and it is therefore recommended. If you are not familiar with running programs in the command line, we suggest that you download Anaconda to use CC-FIT2.py (see Section 5).

For clarity, the program prints to screen the undergoing task (while running) and explicitly records the filenames and folders of the files being saved. For each module (Figure 1), the program outputs the plots of the fitted data (saved as $Name.png), a text file with the actual model optimised parameters ($Name_params.out) and a text file with the fitted data ($Name _fit.out).

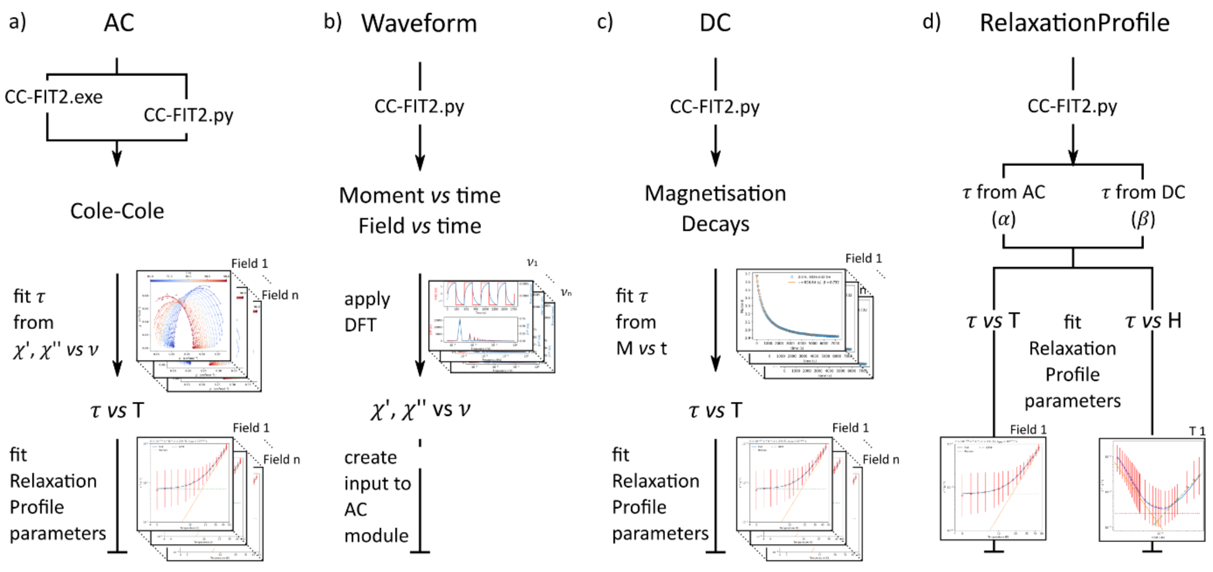


Figure 1. Workflow diagram for *AC*, *Waveform*, *DC* and *RelaxationProfile* modules (a-d), respectively. The executable version only supports the *AC* module.

Parameter error definition.

For both the fitting of the (generalised) Debye function, magnetisation decays and of the magnetic relaxation rates, the parameter errors σ_i are calculated on the basis of the Jacobian matrix obtained from the optimisation algorithm $\bar{\bar{J}}$, Equations 1 and 2.

$$\sigma_i = \sqrt{\bar{\bar{C}}_{ii}} \tag{1}$$

$$\bar{\bar{C}} = \frac{\left(\bar{\bar{J}}^T \bar{\bar{J}}\right)^{-1} RSS}{N - k} \tag{2}$$

where $\bar{\bar{C}}$ is the covariance matrix
RSS is the residual sum of squares at the minimum
N is the number of data points
k is the number of variables

2. User Guide

The standalone executable (CC-FIT2.exe) can be run by simply double-clicking it and providing the raw ac.dat file. The command line version (CC-FIT2.py) offers more functionality and can be run under four distinct modes: *AC*, *Waveform*, *DC* or *RelaxationProfile* (Figure 1). Each mode has its own arguments, detailed below.

AC module.

The input file for this option is the raw ac.dat as generated by the SQUID and only positive susceptibility values are parsed. If the data has been measured under several applied magnetic fields, the program will proceed one field at a time (Figure 1a) – however, **the program will not work if the data contains duplicates**, i.e., points measured at the same field, temperature and frequency. The program fits both the in-phase and out-of-phase susceptibilities simultaneously using either the Debye model, the generalised Debye model, or a combination of two generalised Debye functions (the user will be asked interactively to choose the model). Temperatures without a clear peak in the out-of-phase AC signal are automatically discarded as these do not allow accurate extraction of the relaxation time. Following the AC fitting, the magnetic relaxation times are plotted with their uncertainties according to the protocol in our paper (*Phys. Chem. Chem. Phys.*, 2019, **21**, 23567).

The expected arguments for this option are defined in Table 1 and can be accessed with "*python CC-FIT2.py AC -h*".

After fitting to a (Generalised) Debye model, the program will automatically output

- A plot with the χ' and χ'' components against frequency, at varying temperature.
- A plot of χ'' against χ' (Cole-Cole), at varying temperature.
- A *_fit.out text file with the fitted results, so the user can plot the data independently.
- A *_params.out text file reporting the optimised parameters and associated errors (Eqs 1 and 2). For a Generalised model, $\tau_{ln}^{ESD-up/lw}$ are the upper (up) and lower (lw) estimated standard deviations (ESD) from the log-normal (ln) distribution on τ, *i. e.*, the vertical bars shown in the relaxation profile plots.

After fitting the relaxation profile, the program will automatically output

- A plot of the fitted data (τ^{-1} *vs* T, log-log scale)
- A plot of the residuals ($\tau_{exp}^{-1} - \tau_{fit}^{-1}$ *vs* T).
- A *_fit.out text file with the fitted results, so the user can plot the data independently.
- A *_params.out text file reporting the optimised parameters and associated errors.

Table 1. Mandatory and optional argument for *CC-FIT2.py AC* module.

Mandatory args.	**Description**
filename	File containing the AC raw data, directly obtained from SQUID.
mass	Sample mass (mg)
MW	Sample molecular weight (g/mol)

Optional args.	**Description**
-h	Print help documentation
--process (Option)	Controls the flow of the program. Options: "*plot*" shows the raw data only. No fitting is performed. "*susc*" fits the AC data to extract τ only. "*all*" fits both AC data and relaxation profile. Default: "*all*".
--discard_off	Disables the option that filters out the AC fits with no peak.
--susc_vs_T	Plot and save χ', χ'' vs temperature..
--round (N)	Specifies the rounding level used to group the temperature datasets. If the data contains very closely spaced temperature points and the program struggles to separate them appropriately, use N = 1. Conversely, if the temperature was not very stable and you would like to group more approximate temperatures together, use N = 3. See section 4.2 for an example. Default: N = 2.
-- round_field (N)	Decimal rounding applied to the DC field employed in experiment. Useful to avoid artificial data separation in case there is noise in the recorded field. Default: N = 2.
--select_T	Specifies manual selection of temperatures to use in the fitting of the (generalised) Debye model. See section 4.3 for an example.
--sigma (N)	Confidence interval employed in the log-norm distribution. Default: N = 1.
--data_pointer (str)	String to locate where to start reading in the ac.dat file. This will depend on the specifics of your ac.dat file. Default: "*[Data]*".
--filter_flats	Tells the program to discard practically flat χ'' vs ν data sets.
--error_fit (N)	Optional threshold to define flat χ'' vs ν for the "-- filter_flats" option. A lower value of N is stricter as to classifying a flat profile. Default: N = 1E-06.

--verbose	Print to screen extra information on how the file is being read and statistical results.

Different SQUID magnetometers use different formats to write the results to the ac.dat file. Currently, CC-FIT2 implements a series of valid options for each property to be read (Table 2). Their relative order does not matter, but your raw ac.dat file should present one of the implemented headers for each property – note that any combination is accepted. Otherwise, CC-FIT2 will prompt an error and refer you to this manual. If that is the case, please contact us so we can add your format to the CC-FIT2 library.

Table 2. Implemented ac.dat headers. Any combination is allowed, but an implemented header **must** be present for each property indicated.

Property	**Headers**			
Field	Field (Oe)	Magnetic Field (Oe)		
Temperature	Temperature (K)			
Frequency	Wave Frequency (Hz)	AC Frequency (Hz)	Frequency (Hz)	
χ'	m' (emu)	AC X' (emu/Oe)	M' (emu)	
χ''	m" (emu)	AC X" (emu/Oe)	AC X” (emu/Oe)	M" (emu)
Drive Amplitude	Drive Amplitude (Oe)	AC Drive (Oe)	Amplitude (Oe)	

Waveform module.

This module implements the methodology developed in Phys. Chem. Chem. Phys., 2019, 21, 22302-22307 and is consistent with SuperMatlab.

The expected arguments for this option are defined in Table 3 and can be accessed with "*python CC-FIT2.py Waveform -h*".

After performing the discrete Fourier transforms, the program will automatically output

- A *$NAME_toccfit.dat* file containing the extracted AC susceptibilities. This file can be directly passed to the *AC* module.
- A plot of the extracted susceptibilities. Note this plot is not saved automatically.

Table 3. Mandatory and optional argument for *CC-FIT2.py Waveform*.

Mandatory args.	**Description**
filename(s)	SQUID file(s). It supports shell-style wildcards. For instance, use "**.dat*" to read all files with that extension. Or, say you have *Sample_10K_freq1.dat*, *Sample_12K_freq3.dat*, etc, you could use "*Sample_*K_freq*.dat*".

Optional args.	**Description**
-h	Print help documentation
--time_col (N)	Column number used to read "*Time stamp (s)*" values. Default: 2.
--temp_col (N)	Column number used to read "*Temperature (K)*" values. Default: 3.
--field_col (N)	Column number used to read "*Field (Oe)*" values. Default: 4.
--moment_col (N)	Column number used to read "*Moment (emu)*" values. Default: 5.
--field (N)	Central field value used to measure waveform data. Default: 0.
--data_pointer (str)	String used to locate data in filename(s). Default: "*[Data]*".
--field_window (N M)	Min max field values (Oe) used to define a waveform block (Figure 2). Default: -0.5 0.5
--show	Show individual Fourier transformed plots for each frequency.

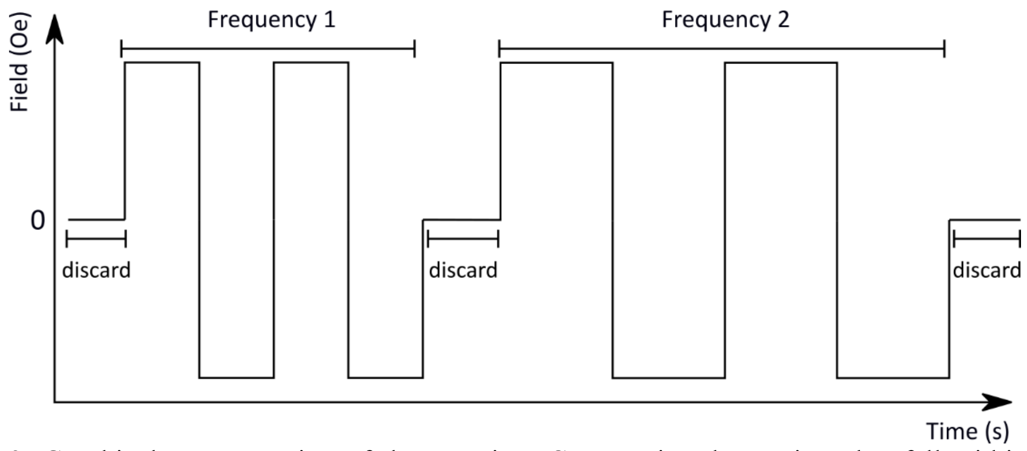


Figure 2. Graphical representation of data parsing. Consecutive data points that fall within the --*field_window* values are discarded.

DC module.

The input file for this option is **not** the raw file generated by the SQUID. See Table 4 for the expected format. The program fits the time-dependent moment of the sample using either a single or a stretched exponential function (Equation 3a and 3b). For each case, the user can specify how to treat the value of the equilibrium magnetisation (M_{eq}, see Table 5). Following the decays' fitting, the magnetic relaxation times are plotted with their uncertainties. In the same way the α parameter in the generalised Debye model can be associated with an uncertainty in τ, the β parameter in the stretched exponential function also relates to a distribution of τ and consequently to an intrinsic uncertainty on the value. These uncertainties are calculated according to the protocol in our paper (SI Section 5 in *J. Am. Chem. Soc.* 2019, **141**, 50, 19935–19940).

$$M(t) = M_{eq} + (M_0 - M_{eq})e^{(-t/\tau)} \qquad \textit{(3a)}$$

$$M(t) = M_{eq} + (M_0 - M_{eq})e^{(-t/\tau)^{\beta}} \qquad \textit{(3b)}$$

The *input_file* for this mode can contain as many fields and temperatures as wanted, but those must be arranged as follows (Table 4).

Table 4. DC input_file format. Parentheses denote variables; "Field = ", "T = ", "time" and "moment" are required as shown; [Data] indicates the section where the actual data is expected.

Field = (FIELD$_1$) Oe		
	T = (T$_1$) K	
	time	moment
	[Data]	[Data]
	T = (T$_2$) K	
	time	moment
	[Data]	[Data]
Field = (FIELD$_2$) Oe		
	T = (T$_1$) K	
	time	moment
	[Data]	[Data]

The expected arguments for this option are defined in Table 5 and can be accessed with "*python CC-FIT2.py DC -h*".

Table 5. Mandatory and optional argument for *CC-FIT2.py DC*.

Mandatory args.	**Description**
filename	File containing the DC magnetisation decays. See Table 4 for details.

Optional args.	**Description**
-h	Print help documentation
--process (Option)	Controls the flow of the program. Options: "*decays*" fits only the magnetisation decays to extract τ; "*all*" fits both the magnetisation decays and the relaxation profile. Default: "*all*".
--model (Option)	Selects the exponential function used to fit the decays. Options: "*single*" uses a single exponential function. "*stretched*" used a stretched exponential function. Default: "*single*".
--M_eq (Option)	Sets the equilibrium magnetisation value to be used. Options: "*exp*" M_eq is set by the last measured value of the data set and not used as a parameter in the fitting routine "*free*" M_eq is left free and passed as an argument to the fitting function. Use in case the equilibrium has not been reached. float M_eq is customarily defined and not used as a parameter in the fitting routine. Note that within this option, the user must provide a number. "*multiple*" Indicate if different Meq values are to be applied to different temperatures. A list of values ordered by increasing temperature will be read from " Multiple_Meq_{}Oe.txt" file in the current directory, where {} is the measured field. File format: first line with the headers T Meq. Default: "*exp*".
--guess [N_1, N_2]	τ and (β) initial guess passed to the model function. Default: 400 (0.95).
--discard (N)	Number of data points measured under a saturating magnetic field that won't be read from each magnetisation decay set. Default: N = 2.
--hide_plots	Do not show the individual magnetisation decay plots.
--save_plots	Save the individual magnetisation decay plots.
--verbose	Print to screen extra information on how the file is being read and statistical results.

RelaxationProfile module.

The input file for this option contains the temperature or field dependence of the relaxation times (τ) (see Table 7 for expected file format).

The expected arguments for this option are defined in Table 6 and can be accessed with "*python CC-FIT2.py RelaxationProfile -h*".

After fitting the relaxation profile, the program will automatically output

- A plot of the fitted data (τ^{-1} *vs* T, log-log scale)
- A plot of the residuals ($\tau_{exp}^{-1} - \tau_{fit}^{-1}$ *vs* T).
- A *_fit.out text file with the fitted results, so the user can plot the data independently.
- A *_params.out text file reporting the optimised parameters and associated errors.

Table 6. Mandatory and optional argument for *CC-FIT2.py RelaxationProfile*.

Mandatory args.	**Description**
input_file	File containing the temperature-dependent relaxation times. See Table 7 for details.
Optional args.	**Description**
-h	Print help documentation
--process (Option)	Controls the flow of the program. Options: "*tau_vs_T*" fits the temperature dependence of relaxation times τ. "*tau_vs_H*" fits the field dependence of the relaxation times τ. Default: "*tau_vs_T*".
--infield	Enables a Direct term (eq 5), relevant only if --process tau_vs_T
--sigma (N)	Confidence interval employed in the log-norm distribution. Relevant only for the τ from AC. Default: N = 1.
--verbose	Print to screen extra information on how the file is being read and statistical results.

The input_file for this mode can contain data from AC, DC and/or both and the first column can be temperature (*--process tau_vs_T*) or field (*--process tau_vs_H*). The grey columns indicate that this data is optional, as its use depends on the fitting function employed. The remaining elements are mandatory.

Table 7. RelaxationProfile input_file format.

AC			
T or Field	tau	tau_err	alpha
[Data]	[Data]	[Data]	[Data]
DC			
T or Field	tau	tau_err	beta
[Data]	[Data]	[Data]	[Data]

For the temperature dependence of the relaxation times, the following models are available:

- In the absence of an applied field, the relaxation profile can be modelled as a combination of Orbach, Raman and Quantum Tunnelling of Magnetisation terms, respectively.

$$\tau^{-1} = 10^{-\tau_0} e^{\left(-U_{eff}/T\right)} + 10^{C} T^{n} + 10^{-\tau_{QTM}} \tag{4}$$

- For in-field data, QTM is substituted by a Direct term. Note that the term A [$s^{-1}K^{-1}$] encompasses implicitly the effect of the applied field, as the employed expression does not know about it. Divide by the actual employed field to obtain [$s^{-1}K^{-1}T^{-1}$].

$$\tau^{-1} = 10^{-\tau_0} e^{\left(-U_{eff}/T\right)} + 10^{C} T^{n} + 10^{A} T \tag{5}$$

For the field dependence of the relaxation times, CC-FIT2 has the following models implemented, as derived in Phys. Rev. B 101, 174402 (2020).

- For data collected at a temperature that falls within the QTM regime in the zero-field regime, the relaxation profile can be modelled as a field-dependent QTM, plus a Raman-II and a constant term, respectively. A "*constrained*" option is also available, where the τ_{QTM} and Ct parameters are kept fixed during the fitting.

$$\tau^{-1} = \frac{10^{-\tau_{QTM}}}{1 + 10^{-\tau_{QTM(H)}} H^{p}} + 10^{C} H^{m} + 10^{Ct} \tag{6}$$

- For data collected at a temperature that falls within the Raman regime in the zero-field regime, the relaxation profile can be modelled as a Brons-Van-Vleck term weighted with a Raman-II (7) or a field-independent constant term (8). A "*constrained*" option is also available, where the Ct parameter is kept fixed during the fitting.

$$\tau^{-1} = \frac{1 + 10^{e} H^{2}}{1 + 10^{f} H^{2}} \cdot \left(10^{C} H^{m}\right) \tag{7}$$

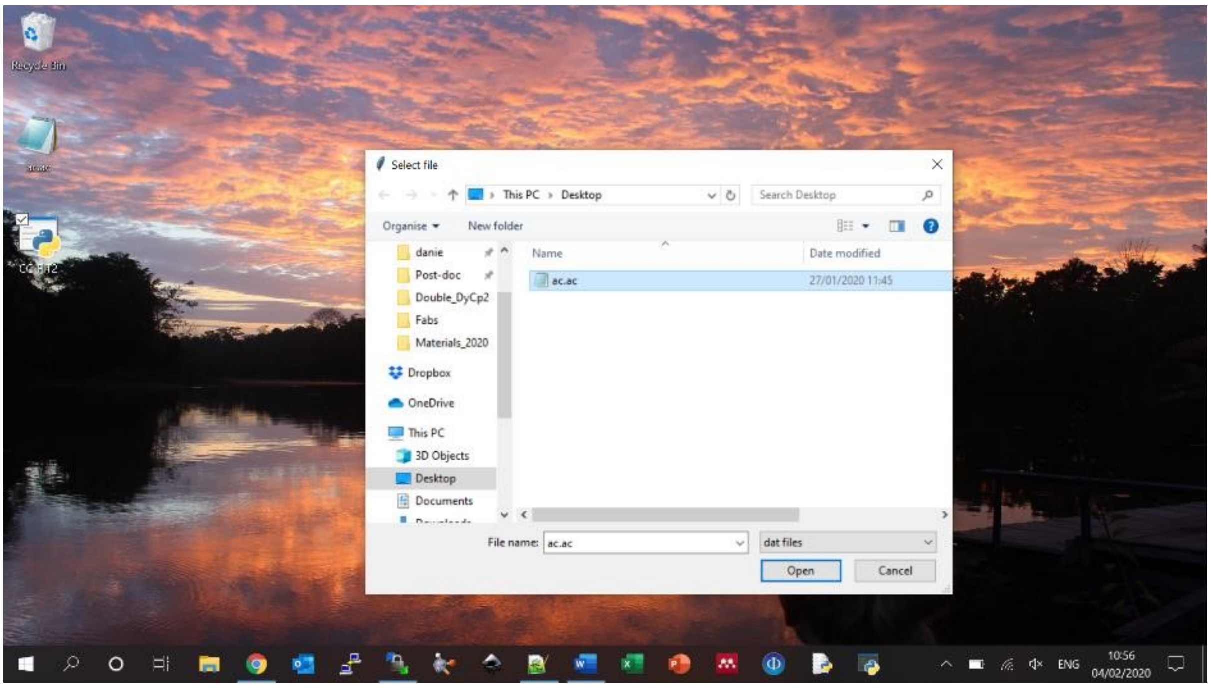
$$\tau^{-1} = \frac{1 + 10^{e} H^{2}}{1 + 10^{f} H^{2}} \cdot \left(10^{Ct}\right) \tag{8}$$

3. Examples with the executable version

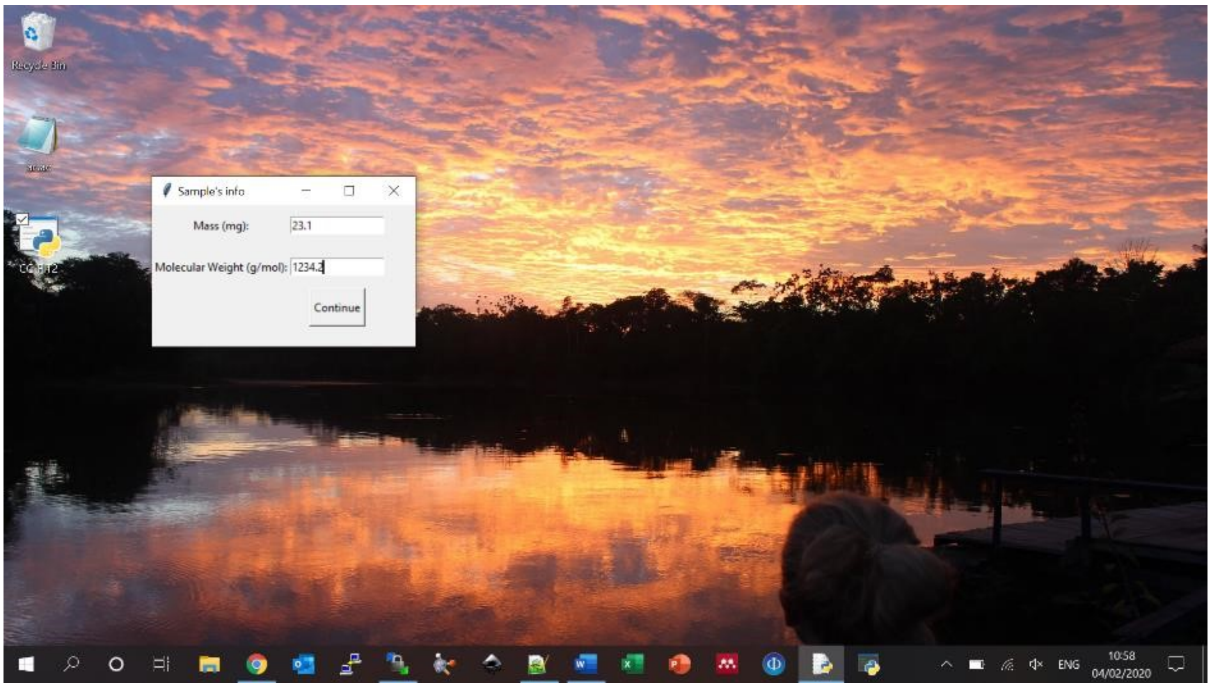
Well-behaved data.

Suppose the user has measured the ac response of a sample under zero-field, at different temperatures, using at each point a different range of wave frequencies, and wants to model the data. The steps to follow to do that with *CC-FIT2* are the following, assuming that the data is in a file named “ac.ac.dat”:

- Double click the CC-FIT.exe logo. This will prompt a “Select file” box dialog. Browse your computer to the folder where your “ac.ac.dat” is, and open it.



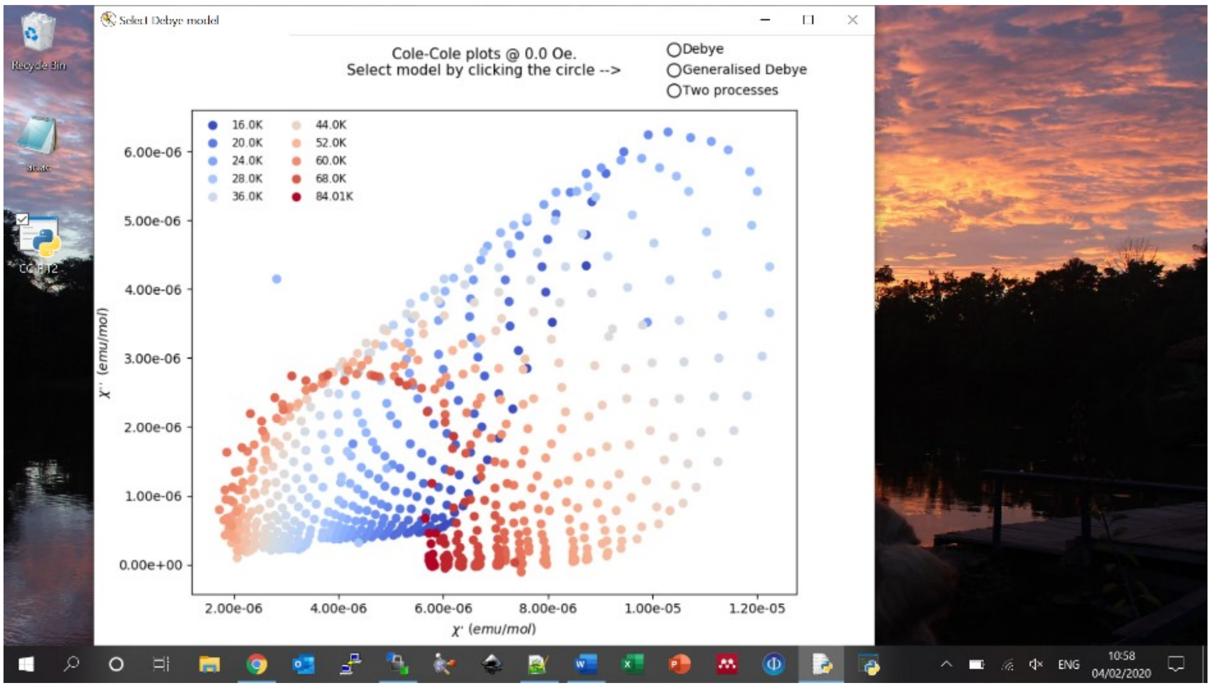
- Another box will appear asking for the mass (mg) and molecular weight (g/mol) of your sample. Type in the values and press “Continue”.


Sample's info

Mass (mg): 23.1

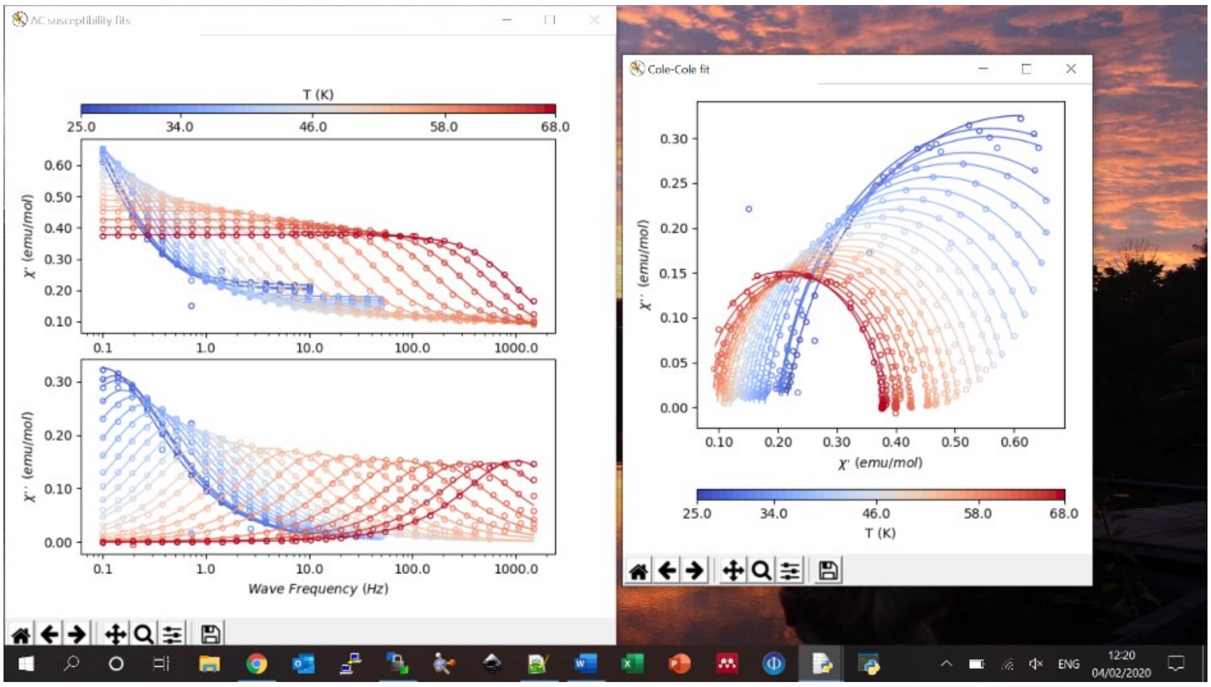
Molecular Weight (g/mol): 1234.2

- A Cole-Cole plot showing the raw data will appear. In the upper part, there are some interactive buttons that can be used to indicate what model is to be used to fit the data. Select the appropriate one.

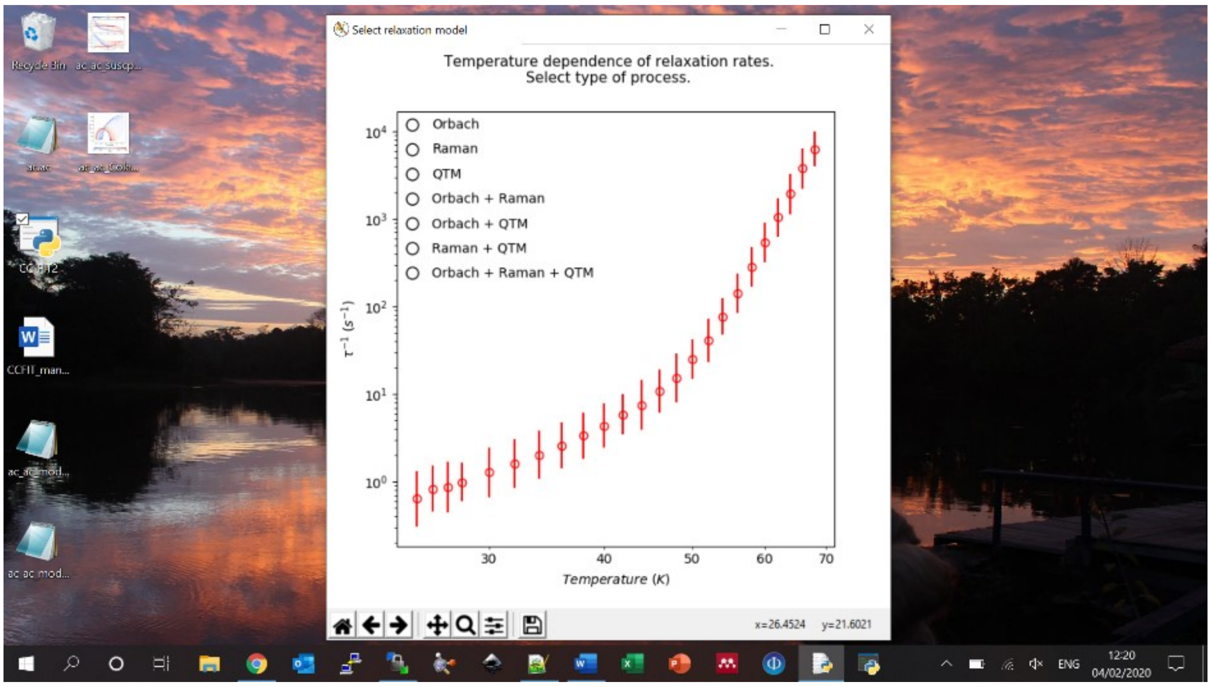

Select Debye model

Cole-Cole plots @ 0.0 Oe.
Select model by clicking the circle -->

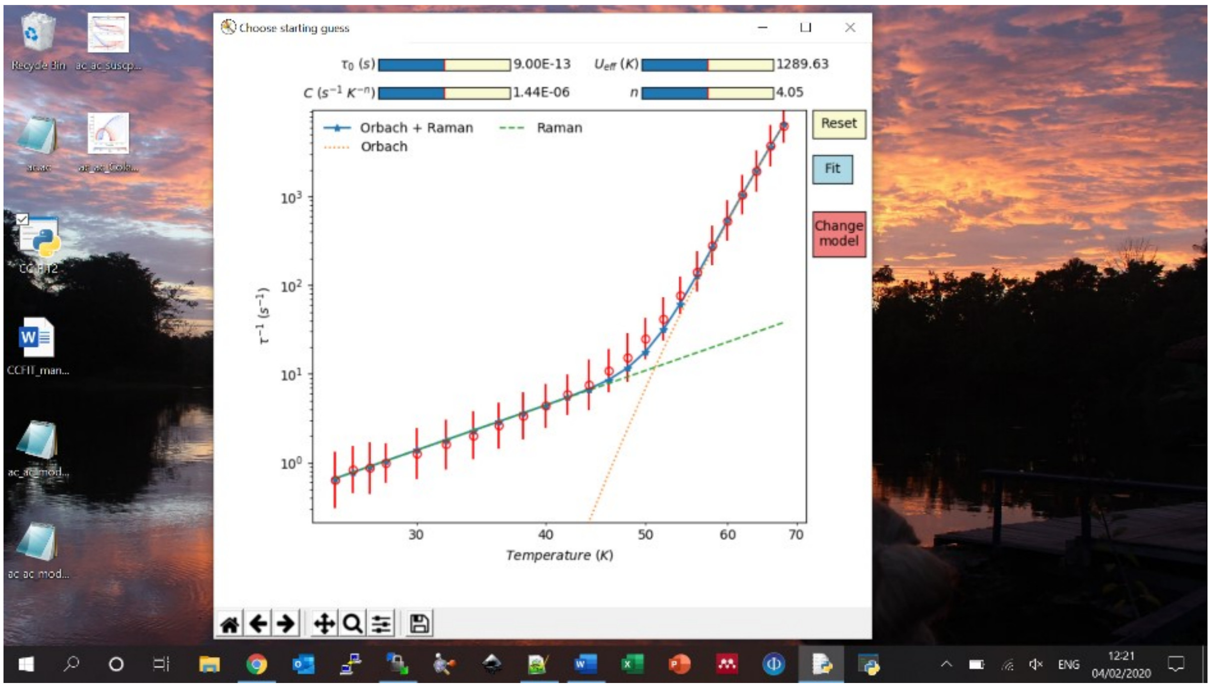
- Two new plots appear, showing the results of the fitting: the Cole-Cole plot and the susceptibility components. Note that the program discards temperature points for which a maximum is not observed; this information is only reported in the command line version (see below).



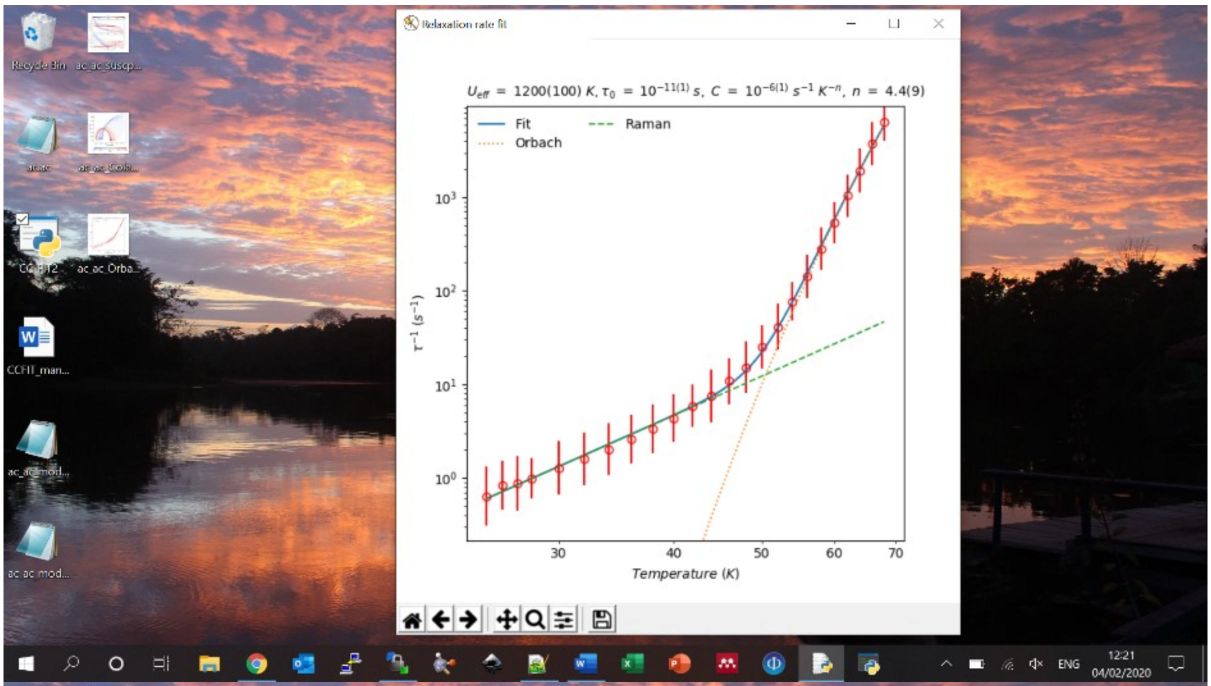
- To continue, close these windows. You do not need to save the figures, as they are automatically saved to the same folder where the data file is. Additionally, two files are created: one contains the optimal parameters of the fit for each temperature, and the other contains the fitted model functions. Once the figures are closed, a new window showing the relaxation profile of the extracted relaxation times is presented. Once again, the interactive buttons on the left allow you to choose a model to fit the temperature dependence of the data. Just click one of them:



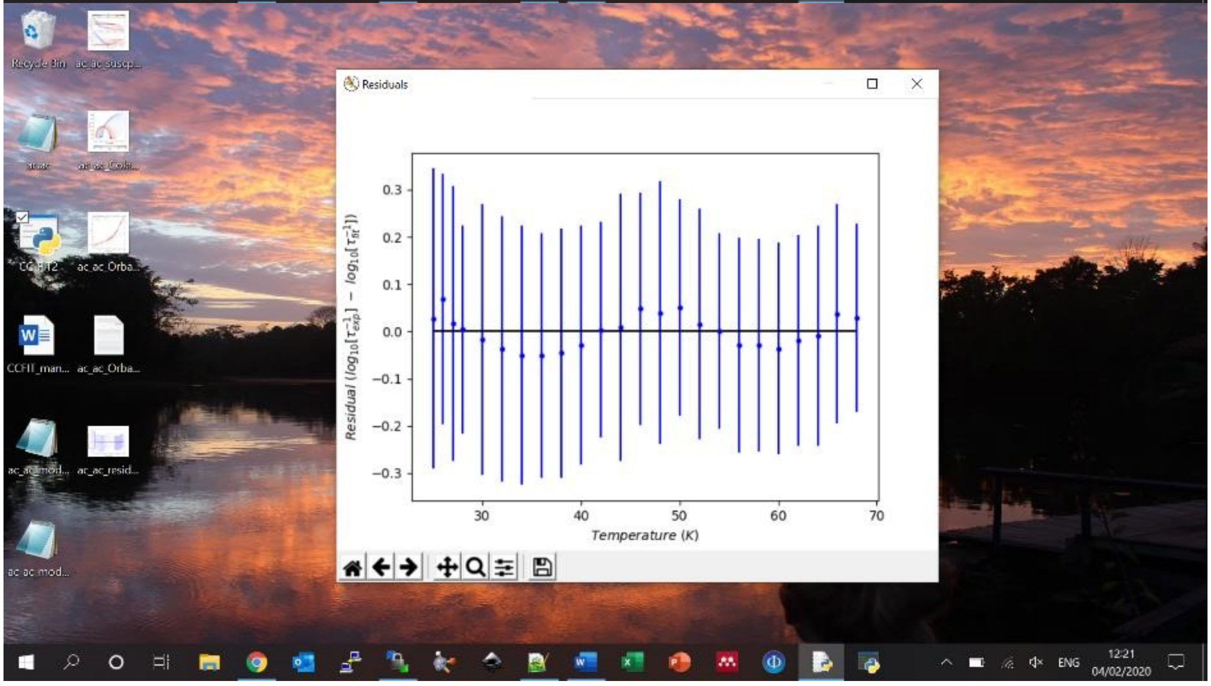
- Choosing a model function produces another window showing you an initial guess to fit the data, with interactive sliders where you can change the parameters if the initial guess is too far off:



- If you think the chosen model function is not an appropriate choice for your data, you can press the "Change model" button, taking you back to selection screen. Once you are happy with the initial guess of your chosen model function, press "Fit" to perform a fit of the relaxation profile, which yields the final results:



Relaxation rate fit
U_{eff} = 1200(100) K, τ_0 = $10^{-11(1)}$ s, C = $10^{-6(1)}$ s^{-1} K^{-n}, n = 4.4(9)
Fit
Raman
Orbach
τ^{-1} (s^{-1})
Temperature (K)



Residuals
Residual ($log_{10}[\tau_{exp}^{-1}]$ − $log_{10}[\tau_{fit}^{-1}]$)
Temperature (K)

Note that the above discussion is not limited to data collected under the same field – if the data file contains data measured under two or more external fields, the program will ask you to do the same at each external field employed.

4. Examples with the command line version

Well-behaved data.

If a user was to use the command line version instead of the executable with the same data file in section 3, they will obtain the same results. However, using the command line version becomes handy in the case of more complicated datasets, as it offers more control over the data treatment.

As indicated in Section 2, CC-FIT2.py can be used in four modes:

> python CC-FIT2.py AC <*filename*> <*mass*> <*MW*>
>
> python CC-FIT2.py AC < *filename(s)*>
>
> python CC-FIT2.py DC < *filename* >
>
> python CC-FIT2.py RelaxationProfile < *filename* >

AC data measured with very small temperature steps: "*--round*" functionality.

Sometimes the sample would present out-of-phase magnetic susceptibility only in a narrow region of temperatures, and therefore the data is collected with very small temperature steps to get as much information as possible. The program may mistakenly treat all the data as belonging to a single temperature point. This behaviour can be fixed using the --round command.

Using the -v command will print to screen what the program is doing, making it easier to track down any possible issue. For instance, for a sequence that has measured these temperatures:

[1.794347, 1.798661, 1.799994, 1.800523, 1.800567, 1.800696, 1.801384, 1.801602, 1.802653, 1.803176, 1.804246, 1.804882, 1.893796, 1.896328, 1.898486, 1.900279, 1.901154, 1.901297, 1.901564, 1.90289, 1.902917, 1.903035, 1.903924, 1.906677, 1.998204, 1.999513, 1.999818, 2.000119, 2.000131, 2.000163, 2.000182, 2.000237, 2.000276, 2.000551, 2.00121, 2.001292]

The default --round value (N = 2) will place all these 36 points in a single set of average temperature of *ca.* 1.9 K. The program will subsequently fail because there are not as many associated frequencies with this data.

However, if the program is executed with "--round 1", these temperatures are grouped into three different sets with 12 points each, corresponding to the averages of *ca.* 1.8 K, 1.9 K and 2.0 K.

Example of usage:

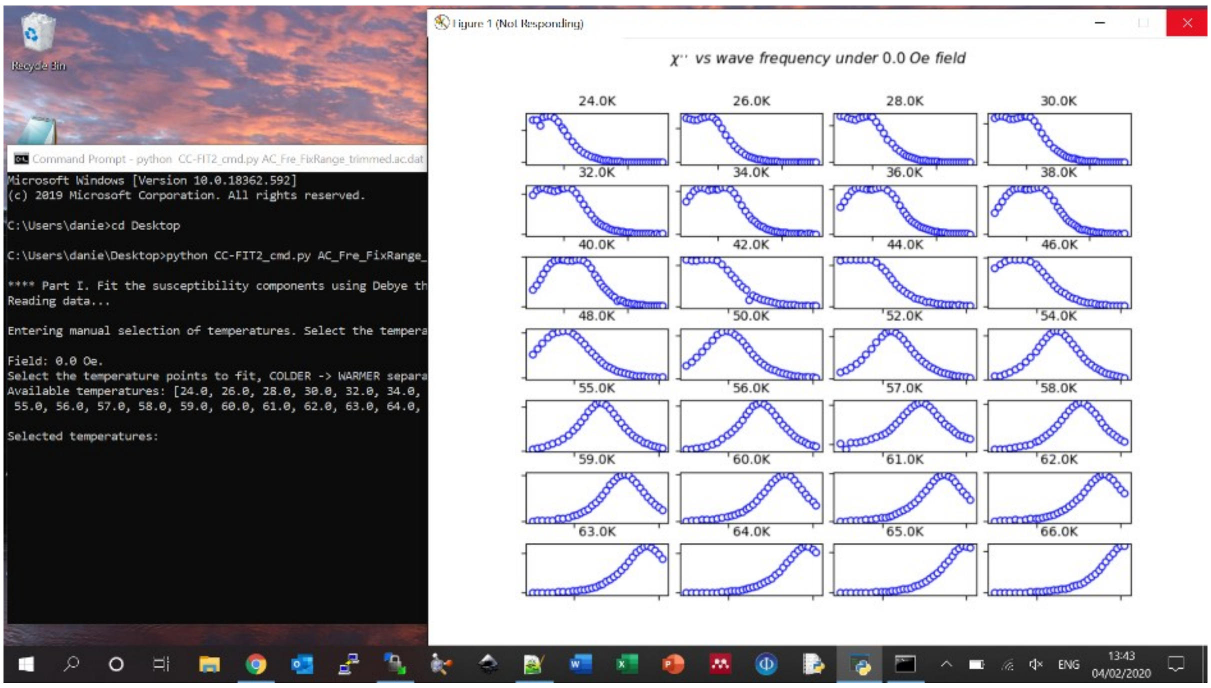
> python CC-FIT2.py AC <*filename*> <*mass*> <*MW*> --round <value:{1, 2, 3, …}>

AC data with one and two peaks in the Cole-Cole plots: "--*select_T*" functionality.

By calling the program with the --select_T option, you can access different sets of temperature ranges, which is convenient in case you cannot apply the same model function across all temperatures.

Say that you have used --select_T, then the window shown below will appear. Here, the temperature points between 26 and 40 K show two clear peaks, whereas the others only show one. The command line prompts you to indicate the temperatures you want to model, enabling to separate this single data file into two different sets; note that this would have to be repeated for the second set.

The available temperatures are printed to the command line and the user is asked to indicate which ones should be fitted. These should be given in the same format shown on the command line, *i.e.* separated by commas and in order of increasing temperature.



```
Microsoft Windows [Version 10.0.18362.592]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\danie>cd Desktop

C:\Users\danie\Desktop>python CC-FIT2_cmd.py AC_Fre_FixRange_

**** Part I. Fit the susceptibility components using Debye th
Reading data...

Entering manual selection of temperatures. Select the tempera

Field: 0.0 Oe.
Select the temperature points to fit, COLDER -> WARMER separa
Available temperatures: [24.0, 26.0, 28.0, 30.0, 32.0, 34.0,
 55.0, 56.0, 57.0, 58.0, 59.0, 60.0, 61.0, 62.0, 63.0, 64.0,

Selected temperatures:
```

Example of usage:

python CC-FIT2.py AC <*filename*> <*mass*> <*MW*> --select_T

External field was not stable during AC collection: "*--round_field*" functionality.

Sometimes the recorded external field jumps slightly around a central value, even if it is set to a fixed one. If this happens, the program will interpret that there are multiple data sets measured at different fields. To avoid this, the user can overwrite the field value read internally by the program --round_field.

Example of usage:

```
python CC-FIT2.py AC <filename> <mass> <MW> --round_field <value:{0, 1, 2, …}>
```

I want to fit the relaxation profile of τ values coming from AC and DC experiments.

Due to the limited range of AC frequencies in typical MPMS or PPMS magnetometers (typically *ca.* 0.1-10,000 Hz), the relaxation times (τ) at the lower end of temperatures are normally not accessible. A common practice is then to employ standard direct current (DC) techniques to measure magnetisation decays and extract τ at a given temperature, where the time-evolution of the sample's magnetisation is recorded. Then, these τ values can be merged to those obtained from AC to report the full relaxation profile of the sample

Example of usage:

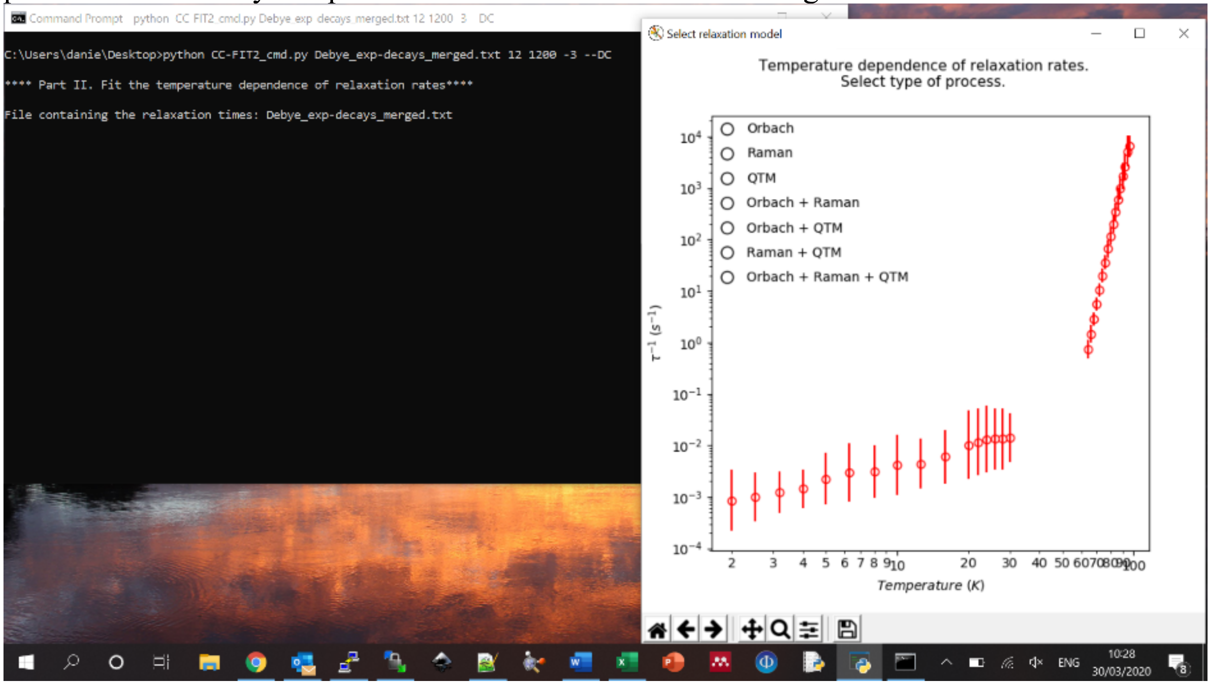
```
python CC-FIT2.py RelaxationProfile <filename>
```

For instance, using the data below:

AC			
T	tau	tau_err	alpha
64.00	1.34E+00	1.49E-01	4.68E-02
66.00	6.85E-01	4.29E-02	4.54E-02
90.00	5.84E-04	1.60E-05	8.34E-02
92.00	3.85E-04	1.44E-05	8.37E-02
94.00	1.93E-04	8.52E-06	1.11E-01
96.00	1.56E-04	1.62E-05	6.19E-02
DC			
T	tau	tau_err	beta
2.0	1.16E+03	1.49E-01	5.97E-01
2.5	1.00E+03	4.29E-02	6.66E-01
3.2	8.27E+02	1.60E-05	7.19E-01
22.0	8.64E+01	1.44E-05	5.66E-01
24.0	7.62E+01	8.52E-06	5.65E-01

Note that the presence of the columns "alpha" and "beta" depends on the model functions that have been used to fit the AC and DC data, respectively, and as such they are optional. If either of those columns are not present in the file, tau_err values will be used to define the errors on τ.

With this data, one would obtain a much more complete relaxation profile, with enough data points to statistically sample each of the Raman and Orbach regimes.



```
Command Prompt   python  CC FIT2_cmd.py Debye_exp  decays_merged.txt 12 1200  3   DC

C:\Users\danie\Desktop>python CC-FIT2_cmd.py Debye_exp-decays_merged.txt 12 1200 -3 --DC

**** Part II. Fit the temperature dependence of relaxation rates****

File containing the relaxation times: Debye_exp-decays_merged.txt
```

Select relaxation model

Temperature dependence of relaxation rates.
Select type of process.

5. Using Anaconda to run CC-FIT2.py

Installing Anaconda.

Anaconda is a distribution of Python that simplifies the usage of different modules required to run CC-FIT2. Go to the Anaconda website and download it. This tutorial might help you in the process of installing it.

Running CC-FIT2.py from Anaconda prompt.

When typing Anaconda in the Windows search bar, a Terminal will appear. You will be executing CC-FIT2_cmd.py from here. There are many ways you can make a python script callable from any directory in your computer, but they are system-dependent and a bit involved. For that reason, we recommend that you do the following, which despite not being the most efficient approach, should work.

- In your Desktop, create a folder called CCFIT and place the CC-FIT2.py file in there.
- Open the command prompt of Anaconda by typing "*Anaconda"* in the search bar.
- In the terminal that just opened, navigate to the folder where you have the data that you want to treat. To change directory in the terminal, type "*cd folder*" where folder is the name of the folder you want to change to.
- Once you are in the correct folder, type:

  "python C:\Users\$YourUserName\Desktop\CCFIT\CC-FIT2.py -h"

  Where "$YourUserName" is a variable unique to your computer and needs to be specified. For instance, in my laptop this is "C:\Users\daniel", so "$YourUserName=daniel. If everything works you should see printed to screen the available options, and then you are good to go.