XPCS CONTIN

June 2, 2017

Contents

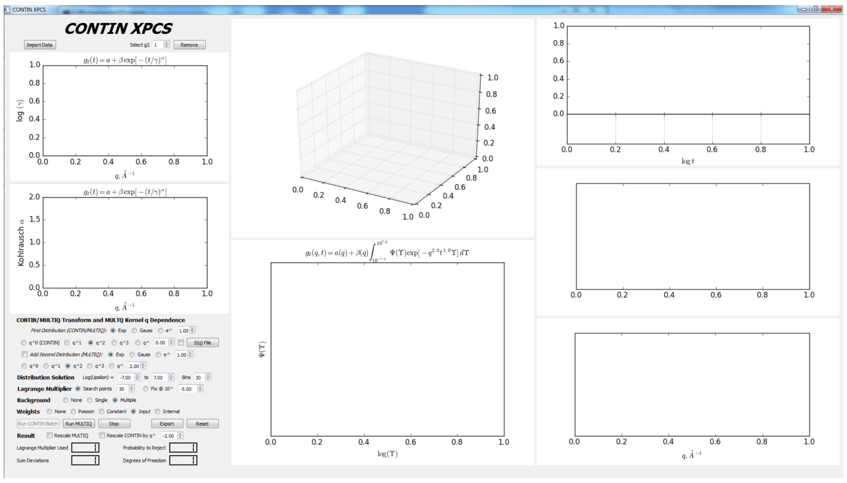
1 *Quick Start* - Install and Run Test Data

1.1 Running TestData2

This section assumes presence of the folder `Program_Distribution`.

1. **Start the program**

   Open the `Program_Distribution` folder and double click `RunXPCS_CONTIN.bat`. After a brief delay, the following GUI appears.



CONTIN/MULTIQ Transform and MULTQ Kernel q Dependence

First Distribution (CONTIN/MULTIQ): [✓ Exp] Gauss e^ 1.00

q^0 (CONTIN) q^1 [✓ q^2] q^3 q^ 0.00 S(q) File

Add Second Distribution (MULTIQ): [✓ Exp] Gauss e^ 1.00

q^0 q^1 [✓ q^2] q^3 q^ 2.00

Distribution Solution Log(Upsilon) = -7.00 to 7.00 Bins 30

Lagrange Multiplier [✓ Search points] 30 Fix @ 10^ -5.00

Background None Single [✓ Multiple]

Weights None Poisson Constant [✓ Input] Internal

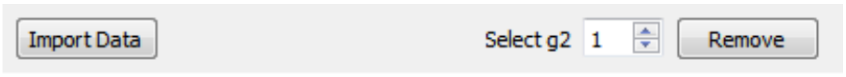
Run CONTIN Batch | Run MULTIQ | Stop | Export | Reset

Result Rescale MULTIQ Rescale CONTIN by q^ -2.00

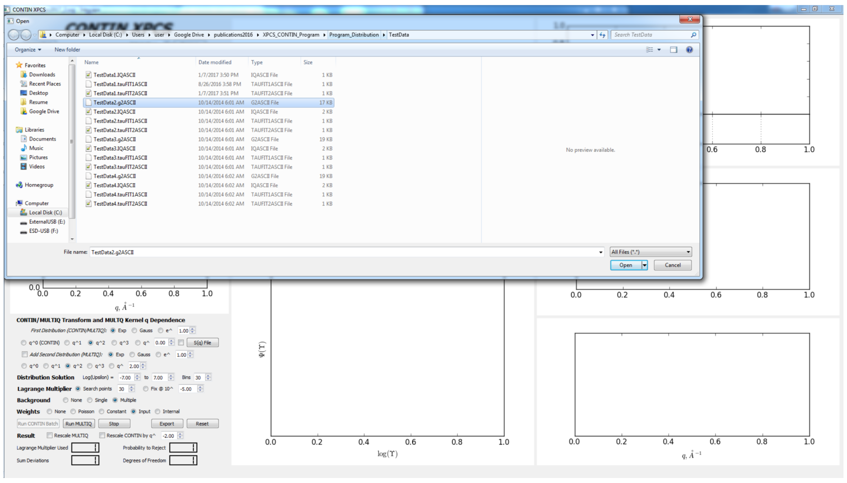
Lagrange Multiplier Used   Probability to Reject

Sum Deviations   Degrees of Freedom

2. **Load the data**



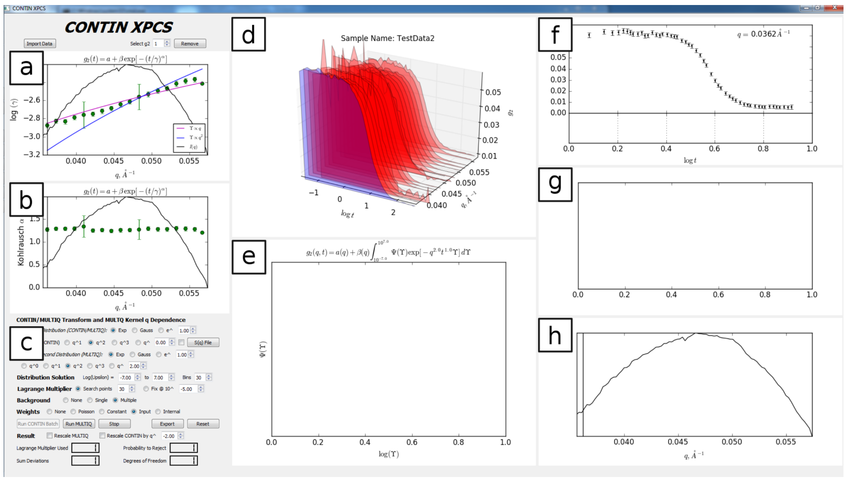
Click **Import Data**, navigate to the `TestData` folder, and open the file named `TestData2.g2ASCII`.


CONTIN XPCS
Open
Computer ▸ Local Disk (C:) ▸ Users ▸ user ▸ Google Drive ▸ publications2016 ▸ XPCS_CONTIN_Program ▸ Program_Distribution ▸ TestData
Organize
New folder
No preview available.
Open
Cancel
All Files (*.*)

Name	Date modified	Type	Size
TestData1.IQASCII	1/7/2017 3:50 PM	IQASCII File	1 KB
TestData1.tauFIT1ASCII	8/26/2016 3:58 PM	TAUFIT1ASCII File	1 KB
TestData1.tauFIT2ASCII	1/7/2017 3:51 PM	TAUFIT2ASCII File	1 KB
TestData2.g2ASCII	10/14/2014 6:01 AM	G2ASCII File	17 KB
TestData2.IQASCII	10/14/2014 6:01 AM	IQASCII File	2 KB
TestData2.tauFIT1ASCII	10/14/2014 6:01 AM	TAUFIT1ASCII File	1 KB
TestData2.tauFIT2ASCII	10/14/2014 6:01 AM	TAUFIT2ASCII File	1 KB
TestData3.g2ASCII	10/14/2014 6:01 AM	G2ASCII File	19 KB
TestData3.IQASCII	10/14/2014 6:01 AM	IQASCII File	2 KB
TestData3.tauFIT1ASCII	10/14/2014 6:01 AM	TAUFIT1ASCII File	1 KB
TestData3.tauFIT2ASCII	10/14/2014 6:01 AM	TAUFIT2ASCII File	1 KB
TestData4.g2ASCII	10/14/2014 6:02 AM	G2ASCII File	19 KB
TestData4.IQASCII	10/14/2014 6:02 AM	IQASCII File	2 KB
TestData4.tauFIT1ASCII	10/14/2014 6:02 AM	TAUFIT1ASCII File	1 KB
TestData4.tauFIT2ASCII	10/14/2014 6:02 AM	TAUFIT2ASCII File	1 KB

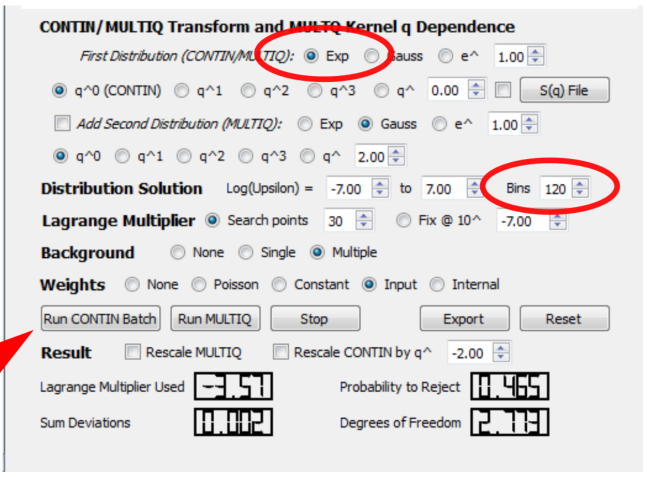
File name: TestData2.g2ASCII

Loading the data populates (a) and (b) with the Kohlrausch fitting parameters from the data set and (d) with the g_2 data. Best fit lines to q and q^2 are shown on the plot of the Kohlrausch exponential relaxation rate (a). Different g_2 (different q points) may be selected by clicking up or down the `Select g2` window. The selected q point is highlighted in blue in the 3D plot (d) and shown by points in (f).



3. **Process using `CONTIN` - Inverse Laplace Transform**

Select **q^0** for the first distribution and increase the number of bins to **120**. Pressing **Run CONTIN Batch** solves the inverse Laplace transform for each g_2 individually, giving a $\Psi(\Upsilon)$ for each g_2.



CONTIN/MULTIQ Transform and MULTIQ Kernel q Dependence

First Distribution (CONTIN/MULTIQ): [✓ Exp] Gauss e^ 1.00

[✓ q^0 (CONTIN)] q^1 q^2 q^3 q^ 0.00 S(q) File

Add Second Distribution (MULTIQ): Exp [✓ Gauss] e^ 1.00

[✓ q^0] q^1 q^2 q^3 q^ 2.00

Distribution Solution Log(Upsilon) = -7.00 to 7.00 Bins 120

Lagrange Multiplier [✓ Search points] 30 Fix @ 10^ -7.00

Background None Single [✓ Multiple]

Weights None Poisson Constant [✓ Input] Internal

Result Rescale MULTIQ Rescale CONTIN by q^ -2.00

Lagrange Multiplier Used -3.57 Probability to Reject 0.465

Sum Deviations 0.002 Degrees of Freedom 2.773

In the terminal window, the following appears

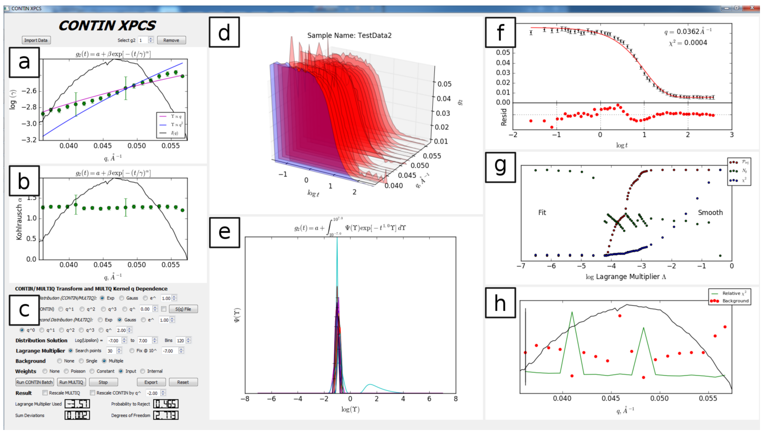
```
BATCH 1 of 18 in thread <some number>
'....COMPLETED BATCH 1 of 18 in thread <some number>
BATCH 2 of 18 in thread <some number>
'....COMPLETED BATCH 2 of 18 in thread <some number>
etc.
```

All the thread numbers should be the same - if not, increase the value of the waiting time `time.sleep(1)` at line 5588 in `XPCS_CONTIN20v11.py`, so Python will wait a little longer for the Fortran backend to finish and avoid spawning another thread.

The selections above correspond to the model shown in Equation 1, where a is the background. For `CONTIN`, selecting **Multiple** for **Background** is equivalent to **Single**; selecting **None** will set $a = 0$. The unnormalized distributions $\Psi(\Upsilon)$ will include the contrast term, and $\beta = 1$.

$$g_2(t) = a + \beta \int_{10^{-7}}^{10^{7}} \Psi(\Upsilon) \exp\left[-\Upsilon t\right] \, d\Upsilon \tag{1}$$

Fitting results in a distribution $\Psi(\Upsilon)$ for each input g_2, shown in (e); (f) shows the original data, fit, and fitting residuals for a single q point selected by the **Select g2** dialog box.

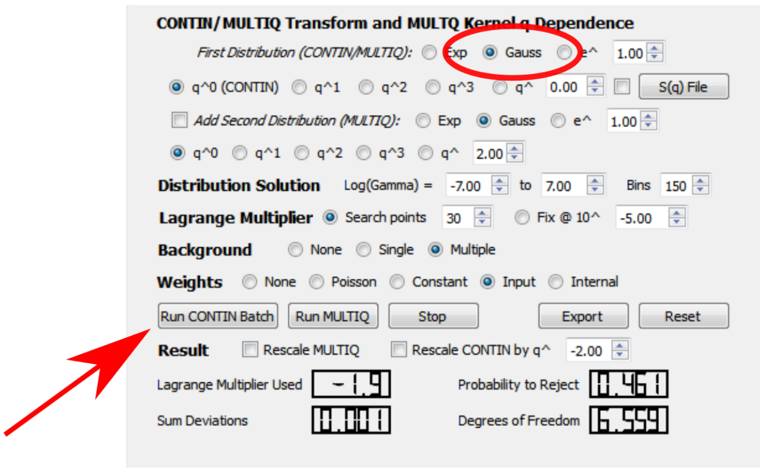


CONTIN/MULTIQ Transform and MULTQ Kernel q Dependence
Distribution (CONTIN/MULTIQ): [✓ Exp] Gauss e^ 1.00
(CONTIN) q^1 q^2 q^3 q^ 0.00 S(q) File
Second Distribution (MULTIQ): Exp [✓ Gauss] e^ 1.00
[✓ q^0] q^1 q^2 q^3 q^ 2.00
Distribution Solution Log(Upsilon) = -7.00 to 7.00 Bins 120
Lagrange Multiplier [✓ Search points] 30 Fix @ 10^ -7.00
Background None Single [✓ Multiple]
Weights None Poisson Constant [✓ Input] Internal
Run CONTIN Batch Run MULTIQ Stop Export Reset
Result Rescale MULTIQ Rescale CONTIN by q^ -2.00
Lagrange Multiplier Used -3.57 Probability to Reject 0.465
Sum Deviations 0.002 Degrees of Freedom 2.73

Inverse Laplace transformation gives a poor result, as shown by the fit to the data in (f). Though the distribution has a sharp peak, the fit decays more slowly than the data, suggesting use of an inverse Gaussian transform – the sharp peak comes from `CONTIN` try to get the fit as close as it can to the data; however, because the positivity constraint on the solution $\Psi(\Upsilon)$, faster than exponential decays cannot be inverse Laplace transformed.

4. **Process using `CONTIN` - Inverse Gaussian Transform**

   Faster than exponential dynamics suggest use of an inverse Gaussian transform. Select **Gauss** and press **Run CONTIN Batch**. The program will perform inverse Gaussian transformation on each g_2 individually, and give a series of distributions $\Psi(\Upsilon)$.



CONTIN/MULTIQ Transform and MULTQ Kernel q Dependence

First Distribution (CONTIN/MULTIQ): Exp [✓ Gauss] e^ 1.00

[✓ q^0 (CONTIN)] q^1 q^2 q^3 q^ 0.00 S(q) File

Add Second Distribution (MULTIQ): Exp [✓ Gauss] e^ 1.00

[✓ q^0] q^1 q^2 q^3 q^ 2.00

Distribution Solution Log(Gamma) = -7.00 to 7.00 Bins 150

Lagrange Multiplier [✓ Search points] 30 Fix @ 10^ -5.00

Background None Single [✓ Multiple]

Weights None Poisson Constant [✓ Input] Internal

Result Rescale MULTIQ Rescale CONTIN by q^ -2.00

Lagrange Multiplier Used -1.9

Probability to Reject 0.461

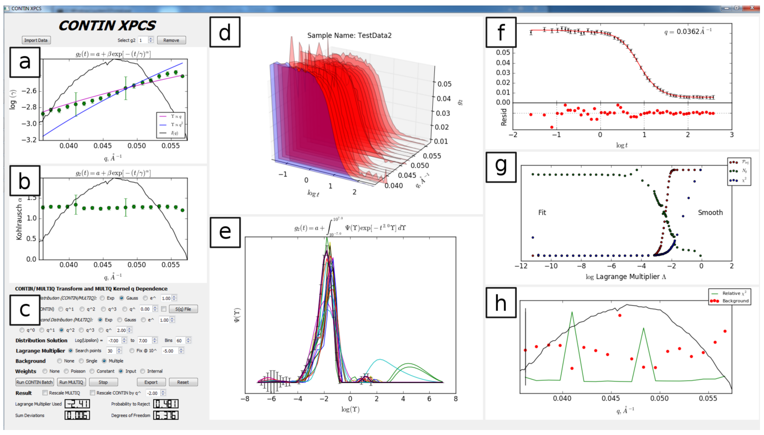
Sum Deviations 0.001

Degrees of Freedom 6.559

The selections above correspond Equation 2, where again $\beta = 1$.

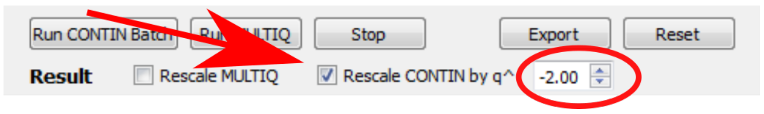
$$g_2(t) = a + \beta \int_{10^{-7}}^{10^7} \Psi(\Upsilon) \exp\left[-\Upsilon t^2\right] d\Upsilon \tag{2}$$

The result is a series of distributions of Gaussian functions (e) that gives a good fit (f) to the data.



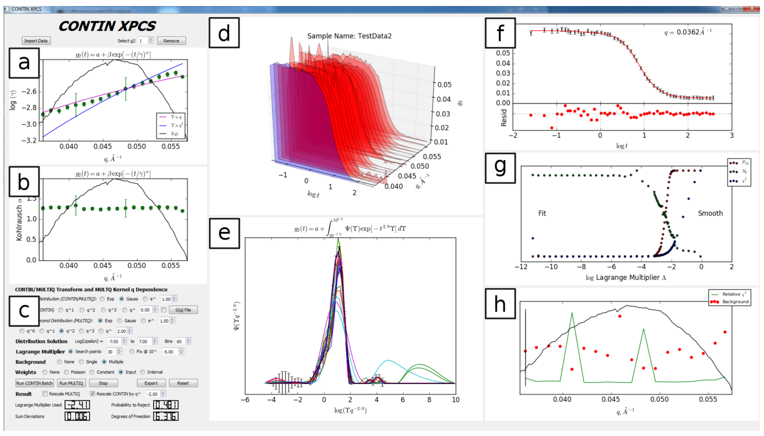
The panel (h) shows the values found for each background a as a function of q, along with the relative goodness of fit χ^2. The two spikes in χ^2 correspond to the two noisy g_2s visible in (d).

Autocorrelation functions typically have some dependence on length scale q. For classical or ballistic diffusion, we expect to find $\propto q^2$. `CONTIN XPCS` allows testing for simple q^n dependency from the sequential `CONTIN` result. Check the box **Rescale CONTIN by q^**, and enter **-2.00** in the input box.



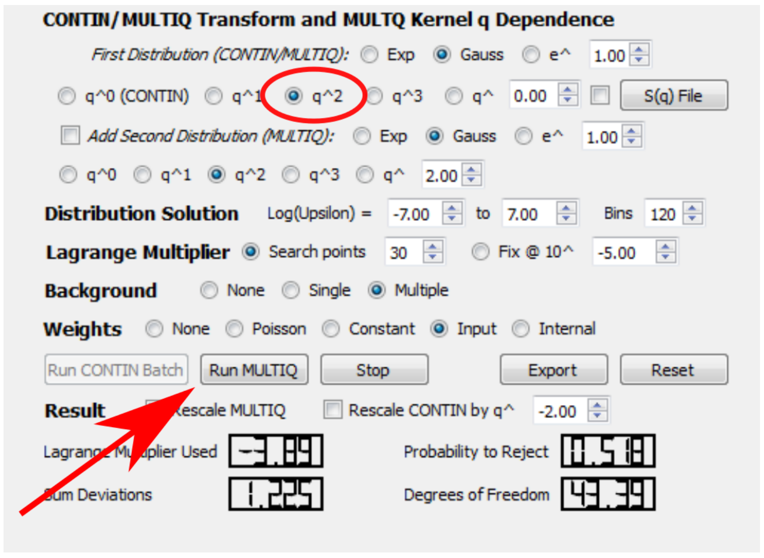
Result ☐ Rescale MULTIQ ☑ Rescale CONTIN by q^ -2.00

Checking this box rescales all the distributions in panel (e) by their corresponding measurement q value. In this case, the distributions coalesce around a single point.



5. **Process using `MULTIQ` - Inverse Gaussian Transform**

Collective `MULTIQ` analysis attempts to fit a single model to all the data sets. Click **q^2** in the first distribution and press **Run MULTIQ**.



CONTIN/MULTIQ Transform and MULTQ Kernel q Dependence

First Distribution (CONTIN/MULTIQ): Exp [✓ Gauss] e^ 1.00

q^0 (CONTIN) q^1 [✓ q^2] q^3 q^ 0.00 ☐ S(q) File

☐ *Add Second Distribution (MULTIQ):* Exp [✓ Gauss] e^ 1.00

q^0 q^1 [✓ q^2] q^3 q^ 2.00

Distribution Solution Log(Upsilon) = -7.00 to 7.00 Bins 120

Lagrange Multiplier [✓ Search points] 30 Fix @ 10^ -5.00

Background None Single [✓ Multiple]

Weights None Poisson Constant [✓ Input] Internal

Run CONTIN Batch | Run MULTIQ | Stop | Export | Reset

Result ☐ Rescale MULTIQ ☐ Rescale CONTIN by q^ -2.00

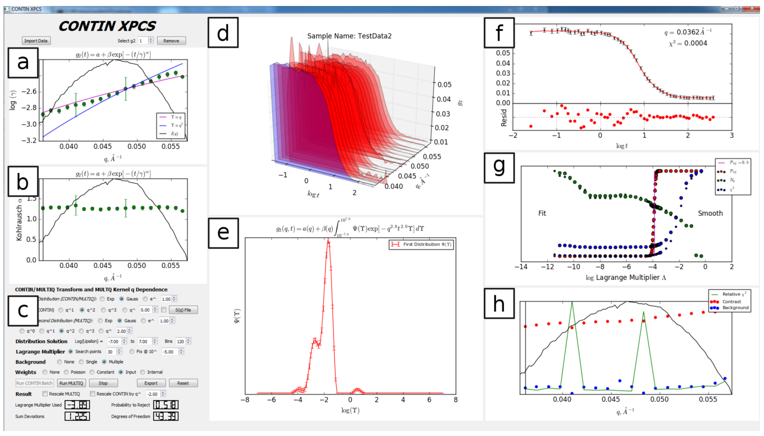
Lagrange Multiplier Used -3.89 Probability to Reject 0.518

Sum Deviations 1.225 Degrees of Freedom 43.39

Thee selections correspond to the model below, where the **Multiple** selection for **Background** corresponds to $a(q)$.

$$g_2(q,t) = a(q) + \beta(q) \int_{10^{-7}}^{10^{7}} \Psi(\Upsilon) e^{-\Upsilon q^2 t^2}\, d\Upsilon \tag{3}$$

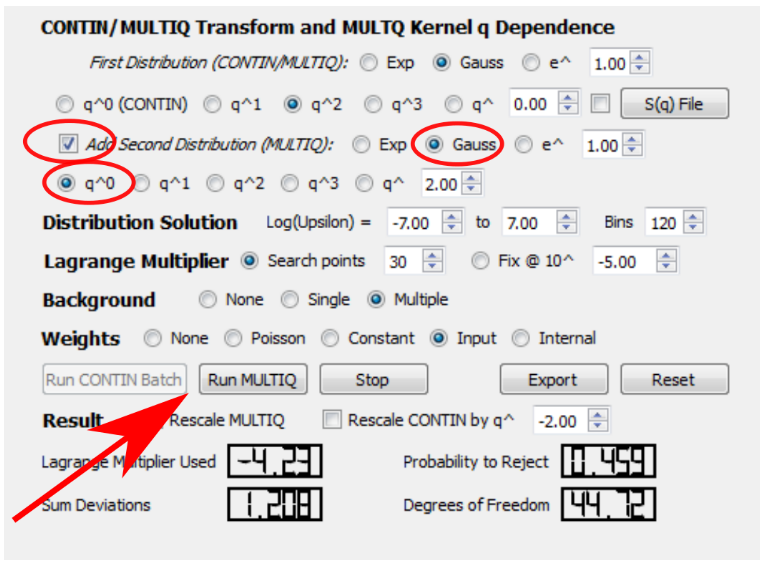
The bimodal result for $\Psi(\Upsilon)$ shown in (e) gives a good fit to the data; (f) shows the fit at the q point highlighted in panel (d). Changing the value in **Select g2** allows viewing the fit and data at other q points. Panel (g) shows the progression of degrees of freedom (number of good parameters, N_g), goodness of fit χ^2 and Provencher's rejection probability metric $\mathcal{P}_{\text{rej}}$ as a function of log Lagrange multiplier Λ. The $\mathcal{P}_{\text{rej}}$ metric defines the ideal result as that at $\mathcal{P}_{\text{rej}} = \frac{1}{2}$, shown by a vertical line in (g) Panel (h) shows the values for the relative scaling factor $\beta(q)$ and backgrounds $a(q)$ as a function of q, along with the relative χ^2 – again, the spikes in χ^2 come from the two noisy g_2s in (d).



6. **Process using `MULTIQ` - Inverse Gaussian Transform**

Collective analysis with a single distribution gave a good global fit to the data. In some cases, it may be useful to test competing hypothses, if only to gain some measure of confidence in the result. For example, if `MULTIQ` had the option of including another distribution with different q dependency – or even another type of transform – this would give some measure of the significance of the original fit. For example, one may question the degree to which the data in the last example actually requires q^2 dependency. To test this, we allow `MULTIQ` the option of using a second distribution without q dependence.

Check the box **Add Second Distribution**, click **Gauss** for the second transform and **q^0** for the second distribution q dependence.



CONTIN/MULTIQ Transform and MULTQ Kernel q Dependence

First Distribution (CONTIN/MULTIQ): Exp [✓ Gauss] e^ 1.00

q^0 (CONTIN) q^1 [✓ q^2] q^3 q^ 0.00 ☐ S(q) File

[✓ *Add Second Distribution (MULTIQ):*] Exp [✓ Gauss] e^ 1.00

[✓ q^0] q^1 q^2 q^3 q^ 2.00

Distribution Solution Log(Upsilon) = -7.00 to 7.00 Bins 120

Lagrange Multiplier [✓ Search points] 30 Fix @ 10^ -5.00

Background None Single [✓ Multiple]

Weights None Poisson Constant [✓ Input] Internal

Result ☐ Rescale MULTIQ ☐ Rescale CONTIN by q^ -2.00

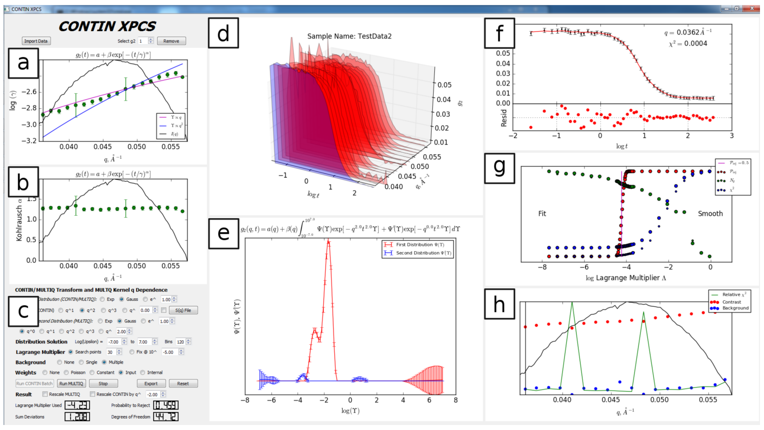
Lagrange Multiplier Used -4.23 Probability to Reject 0.459

Sum Deviations 1.208 Degrees of Freedom 44.72

These selections correspond to the model below.

$$g_2(q,t) = a + \beta \int_{10^{-7}}^{10^7} \underbrace{\Psi(\Upsilon)e^{-\Upsilon q^2 t^2}}_{1^{\text{st}}} + \underbrace{\Psi'(\Upsilon)e^{-\Upsilon q^0 t^2}}_{2^{\text{nd}}} \, d\Upsilon \qquad (4)$$

Press **Run MULTIQ**, and the program will repeat the analysis using the new model. Panel (e) now shows the result in terms of two distributions, $\Psi(\Upsilon)$ (first, red) and $\Psi'(\Upsilon)$ (second, blue). The first q^2-dependent distribution remains essentially the same as determined before, and the new q^0-dependent distribution remains essentially flat, meaning that the data do not support any dynamics that are independent of length scale.



CONTIN/MULTIQ Transform and MULTQ Kernel q Dependence

Distribution (CONTIN/MULTIQ): ○ Exp [✓ Gauss] ○ e^ 1.00

CONTIN) ○ q^1 [✓ q^2] ○ q^3 ○ q^ 0.00 ☐ S(q) File

Second Distribution (MULTIQ): ○ Exp [✓ Gauss] ○ e^ 1.00

[✓ q^0] ○ q^1 ○ q^2 ○ q^3 ○ q^ 2.00

Distribution Solution Log(Upsilon) = -7.00 to 7.00 Bins 120

Lagrange Multiplier [✓ Search points] 30 ○ Fix @ 10^ -5.00

Background ○ None ○ Single [✓ Multiple]

Weights ○ None ○ Poisson ○ Constant [✓ Input] ○ Internal

Run CONTIN Batch | Run MULTIQ | Stop | Export | Reset

Result ☐ Rescale MULTIQ ☐ Rescale CONTIN by q^ -2.00

Lagrange Multiplier Used -4.23 Probability to Reject 0.459

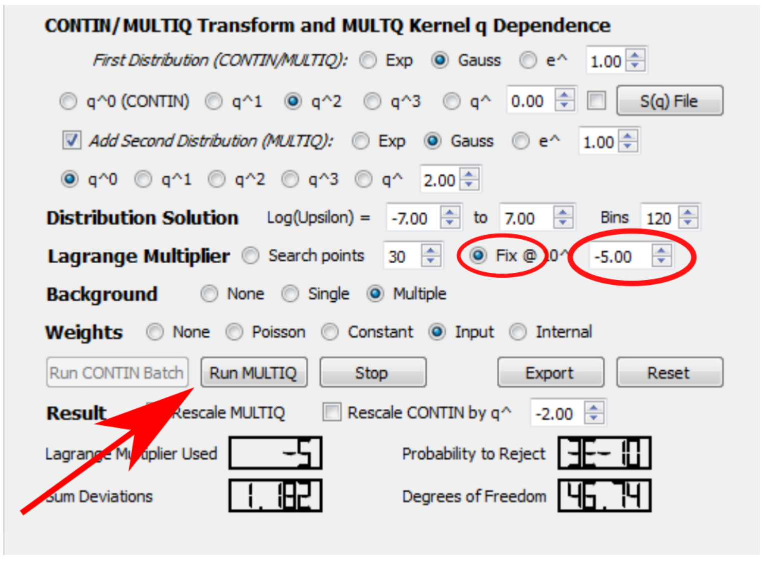
Sum Deviations 1.208 Degrees of Freedom 44.72

7. Process using `MULTIQ` - Check Lagrange Multiplier

Though we find that `CONTIN`'s automatically generally selects the most reasonable value for the Lagrange multiplier Λ, the user should verify the result by observing changes in the solution as a function of Λ.

$$V(\Lambda) = \underbrace{\chi^2}_{\text{Fit}} + \Lambda \underbrace{\int \left(\frac{d^2\Psi(\Upsilon)}{d\Upsilon^2}\right)^2 + \left(\frac{d^2\Psi'(\Upsilon)}{d\Upsilon^2}\right)^2 d\Upsilon}_{\text{Smoothness}} \qquad (5)$$

To observe the effect of Λ on the solution, select the **Fix @ 10ˆ** radio button, click down to **-5.0**. This will rerun the analysis, but, instead of searching for Λ using the rejection probability metric, the program will use the value selected. This will provide solutions for $\Psi(\Upsilon)$ and $\Psi'(\Upsilon)$ from minimization of $V\left(\Lambda = 10^{-5.0}\right)$.



CONTIN/MULTIQ Transform and MULTQ Kernel q Dependence

First Distribution (CONTIN/MULTIQ): Exp [✓ Gauss] e^ 1.00

q^0 (CONTIN) q^1 [✓ q^2] q^3 q^ 0.00 ☐ S(q) File

☑ *Add Second Distribution (MULTIQ):* Exp [✓ Gauss] e^ 1.00

[✓ q^0] q^1 q^2 q^3 q^ 2.00

Distribution Solution Log(Upsilon) = -7.00 to 7.00 Bins 120

Lagrange Multiplier Search points 30 [✓ Fix @ 10^] -5.00

Background None Single [✓ Multiple]

Weights None Poisson Constant [✓ Input] Internal

Result ☐ Rescale MULTIQ ☐ Rescale CONTIN by q^ -2.00

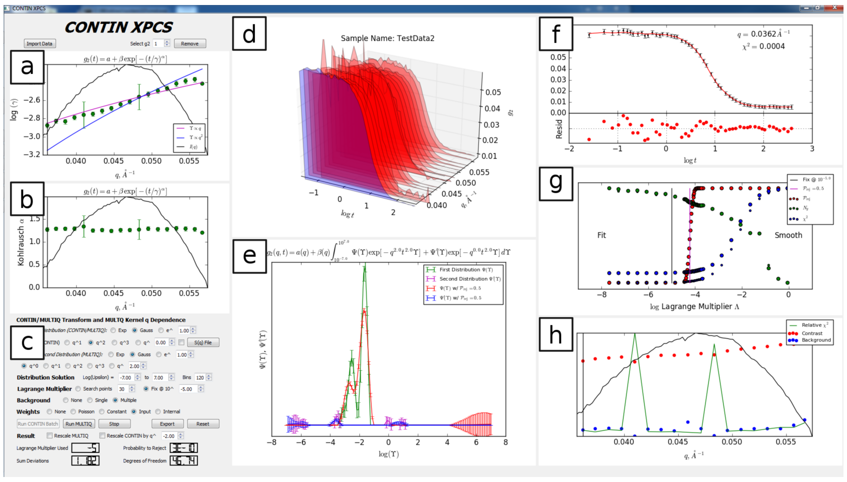
Lagrange Multiplier Used -5

Probability to Reject 3E-10

Sum Deviations 1.182

Degrees of Freedom 46.74

Press **Run MULTIQ**. The result shown in the windows (e) and (g) is shown below. The vertical black line in (g) shows the selected value of $\Lambda = 10^{-5.0}$, and the vertical magenta line shows `CONTIN`'s original selection of $\Lambda = 10^{-4.23}$. The solution window (e) compares `CONTIN`'s solution at $\Lambda = 10^{-4.23}$ (red) with that from the selection of $\Lambda = 10^{-5.0}$ (green). Referring to (g) and Equation 5, smaller values of the Lagrange multiplier cause less weighting of the smoothness constraint and more weighting of fit to the data. Consequently, the result in (e) becomes sharper (green curve) than the original (red). However, the data alone do not justify a sharper distribution – observing chi^2 in (g) shows that the smoother result in red has an only slightly increased χ^2.


CONTIN XPCS
CONTIN XPCS
Import Data
Select g2 1
Remove
a
b
c
d
e
f
g
h
$g_2(t) = a + \beta\exp[-(t/\gamma)^\alpha]$
$\log\,(\gamma)$
$\Upsilon \propto q$
$\Upsilon \propto q^2$
$I(q)$
q, Å^{-1}
Kohlrausch α
Sample Name: TestData2
$\log t$
g_2
q, Å^{-1}
$g_2(q,t) = a(q) + \beta(q)\int_{10^{-7.0}}^{10^{7.0}} \Psi(\Upsilon)\exp[-q^{2.0}t^{2.0}\Upsilon] + \Psi(\Upsilon)\exp[-q^{0.0}t^{2.0}\Upsilon]\,d\Upsilon$
First Distribution $\Psi(\Upsilon)$
Second Distribution $\Psi(\Upsilon)$
$\Psi(\Upsilon)$ w/ $P_{rej}=0.5$
$\Psi(\Upsilon)$ w/ $P_{rej}=0.5$
$\Psi(\Upsilon)$, $\Psi(\Upsilon)$
$\log(\Upsilon)$
$q = 0.0362$ Å^{-1}
$\chi^2 = 0.0004$
Resid
$\log t$
Fix @ $10^{-5.0}$
$P_{rej} = 0.5$
P_{rej}
$N_{\rm f}$
χ^2
Fit
Smooth
log Lagrange Multiplier Λ
Relative χ^2
Contrast
Background
q, Å^{-1}

CONTIN/MULTIQ Transform and MULTQ Kernel q Dependence

istribution (CONTIN/MULTIQ): Exp [✓ Gauss] e^ 1.00

ONTIN) q^1 [✓ q^2] q^3 q^ 0.00 S(q) File

cond Distribution (MULTIQ): Exp [✓ Gauss] e^ 1.00

[✓ q^0] q^1 q^2 q^3 q^ 2.00

Distribution Solution Log(Upsilon) = -7.00 to 7.00 Bins 120

Lagrange Multiplier Search points 30 [✓ Fix @ 10^] -5.00

Background None Single [✓ Multiple]

Weights None Poisson Constant [✓ Input] Internal

Run CONTIN Batch | Run MULTIQ | Stop | Export | Reset

Result Rescale MULTIQ Rescale CONTIN by q^ -2.00

Lagrange Multiplier Used	-5	Probability to Reject	3E-10
Sum Deviations	1.182	Degrees of Freedom	46.74

1.2 Running Other Test data

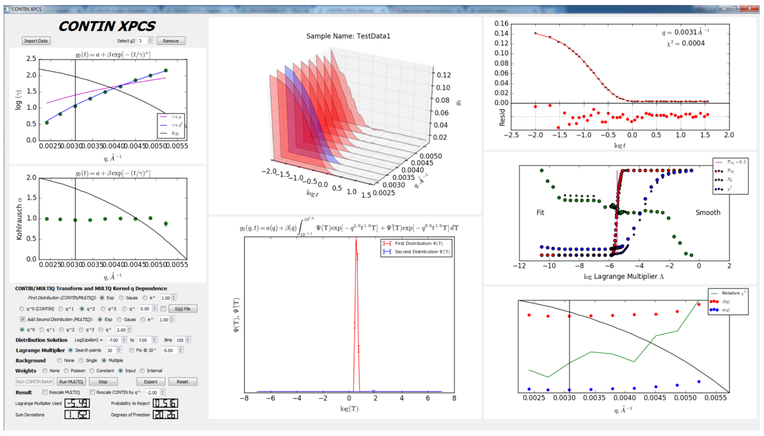
Here we illustrate analysis of the test data to generate the results shown in our publication describing the XPCS inverse transform technique.

Dilute spheres

The file `TestData.g2ASCII` contains the $g_2(q,t)$ measured from a sample of dilute monodisperse spheres. `MULTIQ` analysis according to the model below gives a single q^2 distribution of simple exponential functions.

$$g_2\left(q,t\right) = a + \beta \int \Psi(\Upsilon)e^{\text{-}\Upsilon q^2 t} + \Psi'(\Upsilon)e^{\text{-}\Upsilon q^0 t}\, d\Upsilon \qquad (6)$$

Import the `TestData.g2ASCII`, and select **Exp** and **q^2** for the first distribution and **Exp** and **q^0** for the second. Press **Run MULTIQ**, and the following result appears. `MULTIQ` assigns most all the intensity to the q^2-dependent distribution of simple exponential functions. This result, but from inverse transformation without the use of the second distribution $\Psi'(\Upsilon)$, appears in §2.1.5 and Figure 6 of our original manuscript.



CONTIN XPCS
Import Data
Select g2
Remove
$g_2(t) = a + \beta\exp[-(t/\gamma)^\alpha]$
Kohlrausch α
$q, \mathring{A}^{-1}$
Sample Name: TestData1
$\log t$
$q = 0.0031\,\mathring{A}^{-1}$
$\chi^2 = 0.0004$
Resid
Fit
Smooth
log Lagrange Multiplier Λ
$\Psi(\Upsilon), \Psi'(\Upsilon)$
$\log(\Upsilon)$
First Distribution $\Psi(\Upsilon)$
Second Distribution $\Psi'(\Upsilon)$

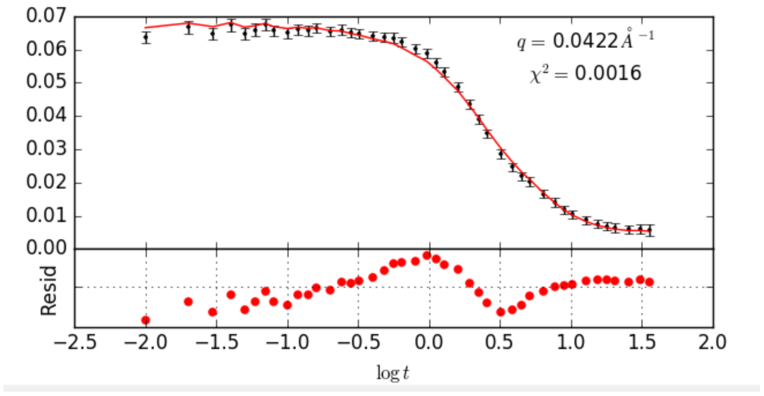
CONTIN/MULTIQ Transform and MULTQ Kernel q Dependence
First Distribution (CONTIN/MULTIQ): [✓ Exp] Gauss e^ 1.00
q^0 (CONTIN) q^1 [✓ q^2] q^3 q^ 0.00 S(q) File
[✓ Add Second Distribution (MULTIQ):] [✓ Exp] Gauss e^ 1.00
[✓ q^0] q^1 q^2 q^3 q^ 2.00
Distribution Solution Log(Upsilon) = -7.00 to 7.00 Bins 150
Lagrange Multiplier [✓ Search points] 30 Fix @ 10^ -5.00
Background None Single [✓ Multiple]
Weights None Poisson Constant [✓ Input] Internal
Run CONTIN Batch Run MULTIQ Stop Export Reset
Result Rescale MULTIQ Rescale CONTIN by q^ -2.00
Lagrange Multiplier Used -5.49 Probability to Reject 0.561
Sum Deviations 1.152 Degrees of Freedom 20.261

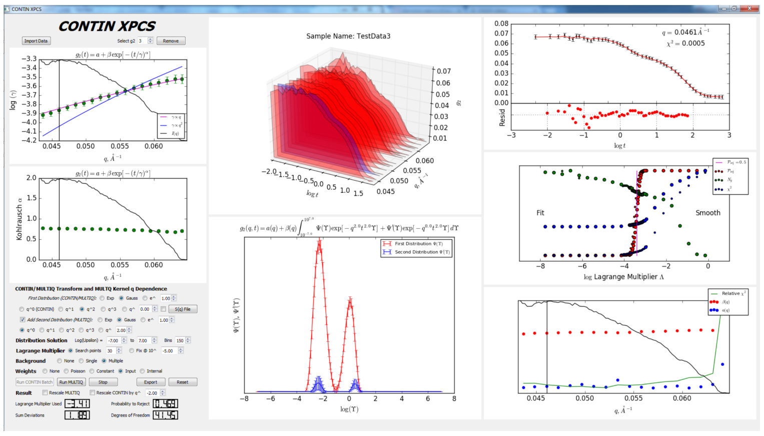
Bimodal Compressed Exponential Relaxation

The file `TestData3.g2ASCII` contains the $g_2(q,t)$ measured from a sample of a colloidal gel. The g_2s show two obvious relaxations, so the single Kohlrausch fit shown gives a misleading result of stretched exponential relaxation; using two Kohlrausch functions reveals that two compressed exponential decays. Attempting inverse Laplace transformation results in a poor fit to the data – the curvature of the data appears sharper than a Laplace transform of a sharp distribution that `CONTIN` will find in a fitting attempt. Instead, the following inverse Gaussian transform model gives a good fit to the data.

$$g_2\left(q,t\right) = a(q) + \beta(q) \int \Psi(\Upsilon)e^{\text{-}\Upsilon q^2 t^2} + \Psi'(\Upsilon)e^{\text{-}\Upsilon q^0 t^2}\, d\Upsilon \qquad (7)$$

Import the `TestData3.g2ASCII`, and select **Gauss** and **q ^ 2** for the first distribution and **Gauss** and **q ^ 0** for the second. Press **Run MULTIQ**, and the following result appears. `MULTIQ` assigns most all the intensity to the q^2-dependent bimodal distribution of Gaussian functions. This result, but without the use of the second distribution $\Psi'(\Upsilon)$, appears in §2.2.4 and Figure 10 of our original manuscript.





CONTIN/MULTIQ Transform and MULTQ Kernel q Dependence

First Distribution (CONTIN/MULTIQ): Exp [✓ Gauss] e^ 1.00

q^0 (CONTIN) q^1 [✓ q^2] q^3 q^ 0.00 S(q) File

[✓ Add Second Distribution (MULTIQ):] Exp [✓ Gauss] e^ 1.00

[✓ q^0] q^1 q^2 q^3 q^ 2.00

Distribution Solution Log(Upsilon) = -7.00 to 7.00 Bins 150

Lagrange Multiplier [✓ Search points] 30 Fix @ 10^ -5.00

Background None Single [✓ Multiple]

Weights None Poisson Constant [✓ Input] Internal

Run CONTIN Batch | Run MULTIQ | Stop | Export | Reset

Result Rescale MULTIQ Rescale CONTIN by q^ -2.00

Lagrange Multiplier Used -3.41 Probability to Reject 0.469

Sum Deviations 1.189 Degrees of Freedom 41.45

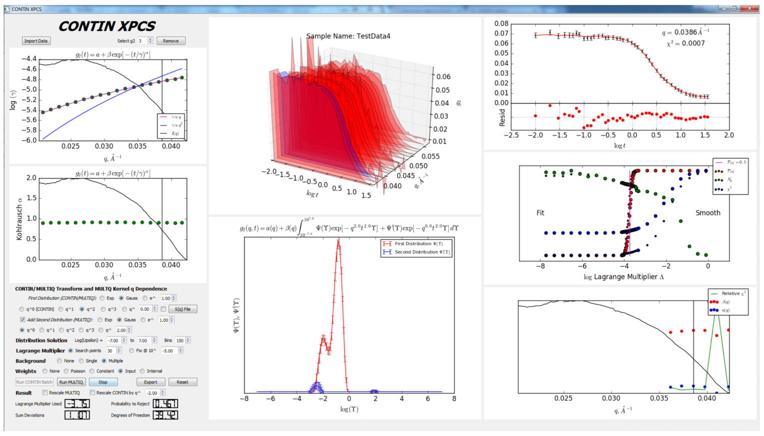
Subtle Bimodal Compressed Exponential Relaxation

The file `TestData4.g2ASCII` contains the $g_2(q,t)$ measured from a sample of a colloidal gel. Analysis with inverse Laplace transform gives a poor fit to the data, shown below.

The inverse Gaussian transform gives a good bimodal fit to the data.

$$g_2\,(q,t) = a(q) + \beta(q) \int \Psi(\Upsilon) e^{-\Upsilon q^2 t^2} + \Psi'(\Upsilon) e^{-\Upsilon q^0 t^2}\, d\Upsilon \qquad (8)$$

Import the `TestData4.g2ASCII`, and select **Gauss** and **q^2** for the first distribution and **Gauss** and **q^0** for the second. Press **Run MULTIQ**, and the following result appears. This result (but without the use of the second distribution $\Psi'(\Upsilon)$) appears in §2.3.3 and Figure 15 of our original manuscript.



CONTIN/MULTIQ Transform and MULTQ Kernel q Dependence

First Distribution (CONTIN/MULTIQ): Exp [✓ Gauss] e^ 1.00

q^0 (CONTIN) q^1 [✓ q^2] q^3 q^ 0.00 S(q) File

[✓ *Add Second Distribution (MULTIQ):*] Exp [✓ Gauss] e^ 1.00

[✓ q^0] q^1 q^2 q^3 q^ 2.00

Distribution Solution Log(Upsilon) = -7.00 to 7.00 Bins 150

Lagrange Multiplier [✓ Search points] 30 Fix @ 10^ -5.00

Background None Single [✓ Multiple]

Weights None Poisson Constant [✓ Input] Internal

Run CONTIN Batch Run MULTIQ Stop Export Reset

Result Rescale MULTIQ Rescale CONTIN by q^ -2.00

Lagrange Multiplier Used -3.75 Probability to Reject 0.467

Sum Deviations 1.101 Degrees of Freedom 39.42

2 Program Description

XPCS CONTIN uses Python to pass commands to a modified version the Fortran program CONTIN MULTIQ compiled as a standalone executable. Python writes the data and parameters to STDIN using Brandon Arnold's FortranFormat and parses the results that are passed to various `.scratch` files in the program directory. The file `terminalOutput.scratch` records the classic ASCII terminal output from CONTIN MULTIQ; this file is not used by the Python program.

Strange things happen if some other program (perhaps a backup or cloud sync) tries to read or write to files used by Python and Fortran to exchange data during the run.

2.1 Input Data Format Requirements

Autocorrelation Data

XPCS CONTIN requires an input data array with the following format.

$$\begin{bmatrix} q_1 & q_2 & q_3 & \cdots & q_n & 0 & \cdots & 0 & 0 \\ t_1 & g_2(q_1,t_1) & \sigma(q_1,t_1) & \cdots & g_2(q_{\frac{n}{2}},t_1) & \sigma(q_{\frac{n}{2}},t_1) & \cdots & g_2(q_n,t_1) & \sigma(q_n,t_1) \\ t_2 & g_2(q_1,t_2) & \sigma(q_1,t_2) & \cdots & g_2(q_{\frac{n}{2}},t_2) & \sigma(q_{\frac{n}{2}},t_2) & \cdots & g_2(q_n,t_2) & \sigma(q_n,t_2) \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ t_m & g_2(q_1,t_m) & \sigma(q_1,t_m) & \cdots & g_2(q_{\frac{n}{2}},t_m) & \sigma(q_{\frac{n}{2}},t_m) & \cdots & g_2(q_n,t_m) & \sigma(q_n,t_m) \end{bmatrix}$$

The first column contains the lag times t corresponding to all the g_2 measured, the remaining columns contain the $g_2(q,t)$ and error $\sigma(q,t)$ at each q point. The top row contains of the q points of the measurement followed by zeros. At Argonne, modification the XPCSGUI MATLAB program as shown below gives the output `.g2ASCII` file in the format required for XPCS CONTIN.

Changes for use with Argonne's XPCSGUI

In the `Ascii_from_XPCSGUI.m` MATLAB subroutine, replace

```
       end
       dlmwrite(g2_ascii_filename,[delay',g2_and_g2Error],'\t');
    end
end
```

with

```
        end
        dynamicQrev=zeros(1,2*numel(dynamicQs)+1);
        dynamicQrev(1:numel(dynamicQs))=dynamicQs';
        dlmwrite(g2_ascii_filename,vertcat(dynamicQrev,[delay',g2_and_g2Error]),'\t');
    end
end
```

This appends the q points (the "dynamic qs") at which the autocorrelations were measured as a row at the top of the array.

Optional Input Data

`XPCS CONTIN` will look for the fit file `.TAUFIT2ASCII` and the static SAXS data `.IQASCII` for plotting in the GUI corresponding to the g_2 file. These files serve only to generate plots on the GUI, so their absence has no effect on the rest of the program – the plots (a) and (b) simply remain empty.

2.2 Detailed Installation Instructions

Do not run the executable `XPCS_CONTIN.exe` - this is the backend called by Python; if called from a terminal, it will sit idle and do nothing.

Installation From Distribution

This section assumes the presence of the folder `Program_Distribution`.

The `Program_Distribution` folder contains three subfolders, `XPCS_CONTIN`, `SourceCode`, `TestData` and a batch file `RunXPCS_CONTIN.bat`. The subfolder `XPCS_CONTIN` contains the python program `XPCS_CONTIN20v11.py`, an entire distribution of Python 2.7.10 and the compiled Fortran executable `XPCS_CONTIN.exe`. This executable seems portable, as it ran successfully on a variety of different Windows 7 and 10 machines.

To run the program, double click `RunXPCS_CONTIN.bat`.

Installation From Supporting Information

This section assumes that you have Python 2.7.10 installed; other Python versions might work (but probably not Python 3.x.x). If something goes wrong, use the option above that includes a complete distribution of Python.

The supporting information contains `XPCS_CONTIN.exe` and `XPCS_CONTIN20v11.py`. Ensure the presence of these two files in the same folder, then run the python program by typing `python XPCS_CONTIN20v11.py` at the prompt in this folder.

Installation From Source Code

On Windows 7, GNU Fortran (GCC) 4.8.2 (i686-4.9.0-posix-dwarf-rt_v3-rev2) compiled the Fortran backend to `XPCS_CONTIN.exe`, the executable called by python using the following command.

```
gfortran -w -static -O3 -o XPCS_CONTIN XPCS_CONTIN14v15.f
```

Other Fortran compilers will probably work.

Ensure the presence of `XPCS_CONTIN.exe` and `XPCS_CONTIN20v11.py` in the same folder, and run the python program by typing `python XPCS_CONTIN20v11.py` at the prompt in this folder.

2.3 Modifications to CONTIN/MULTIQ

Modifications to the original CONTIN MULTIQ include increasing dimensions of the arrays, changing the kernel to accommodate arbitrary Kohlrausch exponents and q dependencies, insertion of read/write statements allowing communication between Python and Fortran, and read/write and spline functions for $S(q)$ supporting the new kernel *USERK*, shown below.

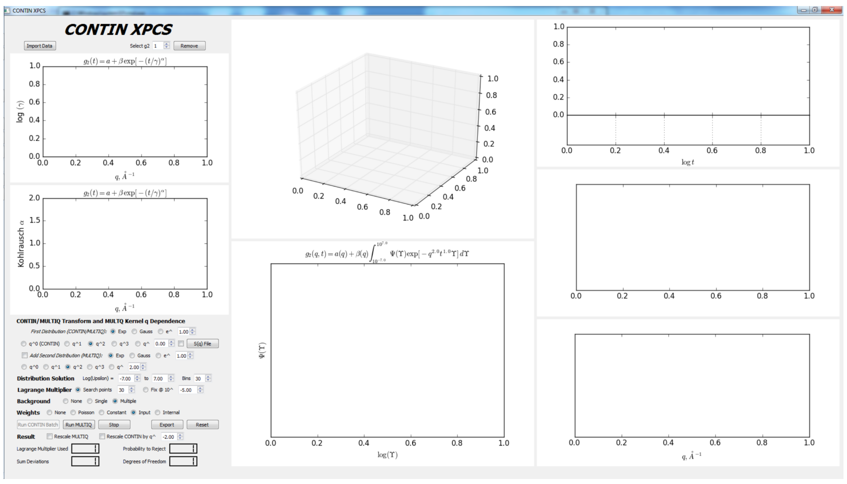
```
    FUNCTION USERK (JT,T,JG,G)                                             0643
    DOUBLE PRECISION PRECIS, RANGE                                         0644
    DOUBLE PRECISION qDepKern,qIndKern
    DOUBLE PRECISION qDepKernQDepend,qIndKernQDepend
    LOGICAL usingSqFlag
    REAL sqInputVals(2,500), sqSplineDerv(500)
    INTEGER mtotinSq
   <<< ORIGINAL CONTIN PREAMBLE >>>
    COMMON /userInput/qDepKern,qIndKern
    COMMON /userInput2/qDepKernQDepend,qIndKernQDepend
    COMMON /usingSqGrid/usingSqFlag,sqInputVals, sqSplineDerv,mtotinSq

    DATA IHOLER/1HU, 1HS, 1HE, 1HR, 1HK, 1H /                              0662
    IF (JT.GT.NY .OR. JG.GT.NG+1 .OR. MINO(JT,JG).LE.0) CALL               0663
   1 ERRMES (1,.TRUE.,IHOLER,NOUT)                                         0664

    ruser(51)=1.
    userk=0.

    DO 110 idset=1,iuser(20)
          IF (jt .LE. iuser(20+idset)) GO TO 150
110 CONTINUE
       CALL ERRMES (9, .TRUE., IHOLER, NOUT)
150 CONTINUE

    IF (luser(29) .AND. 2*jg.gt.ng) THEN
       ex=g(jg)*(t(jt)**qDepKern)*(ruser(50+idset))**qDepKernQDepend
       USERK=formf2(jt,t,jg,g)*EXP(-EX)
       RETURN
    ELSE
       EX=g(jg)*(t(jt)**qIndKern)*(ruser(50+idset))**qIndKernQDepend
          IF (usingSqFlag) THEN
                CALL splint(sqInputVals(1,1:mtotinSq),
   2 sqInputVals(2,1:mtotinSq),sqSplineDerv,mtotinSq,
   3 ruser(50+idset),sqValueFound)
                EX = EX/sqValueFound
          ENDIF
       USERK=formf2(jt,t,jg,g)*EXP(-EX)
       RETURN
    END IF
    END
```

CONTIN XPCS retains `formf2(jt,t,jg,g)=1.` for all `jt`, `t`, `jg`, `g`.

3 Detailed Operation

3.1 Starting Program

At the end of any of the three installation options, the file `XPCS_CONTIN20v11.py` exists in the same directory as the compiled Fortran executable `XPCS_CONTIN.exe`. Open a terminal window in this directory and type `python XPCS_CONTIN20v11.py`. If you used the installation option that includes a full Python distribution, the system should use this Python version (rather than any other installed on your system). After a brief delay, the following GUI should appear.


CONTIN XPCS
Import Data
Select g2 1
Remove
$g_2(t) = a + \beta\exp[-(t/\tau)^{\alpha}]$
log (·)
$q, \AA^{-1}$
Kohlrausch α
$g_2(q,t) = a(q) + \beta(q)\int_{10^{-7.0}}^{10^{7.0}} \Psi(\Upsilon)\exp[-q^{2.0}t^{1.0}\Upsilon]\,d\Upsilon$
$\Psi(\Upsilon)$
$\log(\Upsilon)$
$\log t$

CONTIN/MULTIQ Transform and MULTQ Kernel q Dependence

First Distribution (CONTIN/MULTIQ): [✓ Exp] Gauss e^ 1.00

q^0 (CONTIN) q^1 [✓ q^2] q^3 q^ 0.00 ☐ S(q) File

☐ *Add Second Distribution (MULTIQ):* [✓ Exp] Gauss e^ 1.00

q^0 q^1 [✓ q^2] q^3 q^ 2.00

Distribution Solution Log(Upsilon) = -7.00 to 7.00 Bins 30

Lagrange Multiplier [✓ Search points] 30 Fix @ 10^ -5.00

Background None Single [✓ Multiple]

Weights None Poisson Constant [✓ Input] Internal

Run CONTIN Batch | Run MULTIQ | Stop | Export | Reset

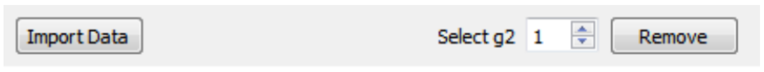
Result ☐ Rescale MULTIQ ☐ Rescale CONTIN by q^ -2.00

Lagrange Multiplier Used | Probability to Reject

Sum Deviations | Degrees of Freedom

3.2 Importing Data

Clicking the button **Import Data** opens a dialog box for selection of the file containing the autocorrelation data as an array in the format described above. Argonne uses the file extension `.g2ASCII` for this data, but `XPCS CONTIN` ignores the extension and will try to import whatever text file selected.



The **Select g2** box allows selection of a particular q point, shown in blue in the 3D plot, and also in the panel showing the fit to the data in the upper right. Selecting a g_2 and clicking **Remove** will remove one or more g_2 from the `CONTIN MULTIQ` input.

3.3 Methods of Analysis - Defining the Kernel

`CONTIN XPCS` uses one of four different inverse transform models. Batch analysis applies inverse transform analysis to each q point individually and gives a set of solutions $\Psi(\Upsilon)$, one for each individual g_2. Single distribution analysis attempts to fit all the g_2s simultaneously using a single distribution $\Psi(\Upsilon)$ with a predefined q dependence. Analysis with two distribution attempts to fit all the g_2s simultaneously using a two distributions $\Psi(\Upsilon)$ and $\Psi'(\Upsilon)$, where each one has a predefined q dependence. Finally, use of an external model, for example a structure factor $S(q)$, for q dependency allows testing of more complicated hypothesis.

$$g_2(t) = a + \beta \int \Psi(\Upsilon) e^{-\Upsilon t^{\xi}} \, d\Upsilon \tag{9}$$

$$g_2(q,t) = a + \beta \int \Psi(\Upsilon) e^{-\Upsilon q^n t^{\xi}} \, d\Upsilon \tag{10}$$

$$g_2(q,t) = a + \beta \int \Psi(\Upsilon) e^{-\Upsilon q^n t^{\xi}} + \Psi'(\Upsilon) e^{-\Upsilon q^{n'} t^{\xi'}} \, d\Upsilon \tag{11}$$

$$g_2(q,t) = a + \beta \int \Psi(\Upsilon) e^{-\Upsilon q^n t^{\xi} S(q)^{-1}} + \Psi'(\Upsilon) e^{-\Upsilon q^{n'} t^{\xi'}} \, d\Upsilon \tag{12}$$

Batch Analysis

Individual batch analysis performs an inverse transform on each individual g_2 sequentially. For this, the user must specify a value for the time exponent ξ. The time exponent should specify either a Laplace ($\xi = 1$) or Gaussian ($\xi = 2$) transform, unless a (compelling) reason exists to choose some other value. `CONTIN` converts the model into matrix form $\mathbf{b} = \mathrm{A}\mathbf{x}$ as shown below.

$$\begin{aligned}
g_2\left(t\right) &= a + \beta \int_0^{\infty} \Psi(\Upsilon) \exp\left[-\Upsilon t^{\xi}\right] \, d\Upsilon \\
g_2\left(t\right) &= \underbrace{\frac{a'}{(1+\beta')}}_{a} + \underbrace{\frac{1}{(1+\beta')}}_{\beta} \int_{\Upsilon\mathrm{Min}}^{\Upsilon\mathrm{Max}} \Psi(\Upsilon) \exp\left[-\Upsilon t^{\xi}\right] \, d\Upsilon \\
(1+\beta') g_2\left(t\right) &= a' + \int_{\Upsilon\mathrm{Min}}^{\Upsilon\mathrm{Max}} \Psi(\Upsilon) \exp\left[-\Upsilon t^{\xi}\right] \, d\Upsilon \\
g_2\left(t_j\right) &= a' - \beta' g_2\left(t_j\right) + \sum_{i=1}^{N} c_i \Psi(\Upsilon) \exp\left[-\Upsilon t^{\xi}\right]
\end{aligned}$$

$$\underbrace{\begin{bmatrix} g_2\left(t_1\right) \\ \vdots \\ g_2\left(t_M\right) \end{bmatrix}}_{\mathbf{b}} = \underbrace{\begin{bmatrix} 1 & \text{-}g_2(t_1) & c_1 e^{\text{-}\Upsilon_1 t_1^{\xi}} & \dots & c_N e^{\text{-}\Upsilon_N t_1^{\xi}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \text{-}g_2(t_M) & c_1 e^{\text{-}\Upsilon_1 t_M^{\xi}} & \dots & c_N e^{\text{-}\Upsilon_N t_M^{\xi}} \end{bmatrix}}_{\mathrm{A}} \underbrace{\begin{bmatrix} a' \\ \beta' \\ \Psi(\Upsilon_1) \\ \vdots \\ \Psi(\Upsilon_N) \end{bmatrix}}_{\mathbf{x}}$$

The figure below shows an example of batch analysis. Panel (e) shows the set of $\Psi(\Upsilon)$ recovered, and panel (h) shows the relative χ^2 and background found for each q point.

Dynamics often have well-defined dependence on the length scale of the measurement. `MULTIQ` allows collective analysis of all the data given some predefined specified q dependence either in the form of a power law q^n or incorporation of a functional model $S(q)$ resulting from de Gennes narrowing.

Single Distribution

The model for collective analysis gives a matrix equation as before, except that the $\mathbf{g_2}$ and K matrices consist of blocks. Using a global background

a and contrast β for all the measured g_2 means that the solution vector $\mathbf{S}$ has the same elements as before.

$$g_2\left(q,t\right) = a + \beta \int_0^\infty \Psi(\Upsilon) \exp\left[-\Upsilon q^n t^\xi\right] \, d\Upsilon$$

$$g_2\left(q_k,t_j\right) = a' - \beta' g_2\left(q_k,t_j\right) + \sum_{i=1}^{N} c_i \Psi(\Upsilon_i) e^{-\Upsilon_i q_k^n t_j^\xi}$$

$$\underbrace{\begin{bmatrix} g_2\left(q_1,t_1\right) \\ \vdots \\ g_2\left(q_1,t_M\right) \\ \hline \vdots \\ \hline g_2\left(q_L,t_1\right) \\ \vdots \\ g_2\left(q_L,t_M\right) \end{bmatrix}}_{\mathbf{g_2}} = \underbrace{\begin{bmatrix} \mathrm{K}_1 \\ \hline \mathrm{K}_2 \\ \hline \vdots \\ \hline \mathrm{K}_L \end{bmatrix}}_{\mathrm{K}} \underbrace{\begin{bmatrix} a' \\ \beta' \\ \Psi(\Upsilon_1) \\ \vdots \\ \Psi(\Upsilon_N) \end{bmatrix}}_{\mathbf{S}}$$

$$\mathrm{K}_k = \begin{bmatrix} 1 & \text{-}g_2(q_k,t_1) & c_1 e^{\text{-}\Upsilon_1 t_1 q_k^2} & \dots & c_N e^{\text{-}\Upsilon_N t_1 q_k^2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \text{-}g_2(q_k,t_M) & c_1 e^{\text{-}\Upsilon_1 t_M q_k^2} & \dots & c_N e^{\text{-}\Upsilon_N t_M q_k^2} \end{bmatrix}$$

Use of a different background and coherence factor for each g_2 requires modification of the kernel, with $a(q)$ and $\beta(q)$ rendered functions of q.

$$g_2\left(q,t\right) = a(q) + \beta(q) \int_0^\infty \Psi(\Upsilon) \exp\left[-\Upsilon q^n t^\xi\right] \, d\Upsilon$$

$$g_2\left(q_k,t_j\right) = a'_k - \beta'_k g_2\left(q_k,t_j\right) + \int_{\Upsilon_{\min}}^{\Upsilon_{\max}} \Psi(\Upsilon) \exp\left[-\Upsilon q_k^n t_j^\xi\right] \, d\Upsilon$$

The matrix equation has a solution vector $\mathbf{S}$ that now has blocks for $\mathbf{a}$ and $\boldsymbol{\beta}$, in addition to the distribution $\Psi(\Upsilon)$. The transform matrix K has corresponding modifications, where $\delta_{i\ell}$ is Kronecker's delta function.

$$g_2\left(q_k,t_j\right) = \sum_{\ell=1}^{N} \delta_{i\ell} a'_k + \sum_{\ell=1}^{N} \underbrace{\text{-}\delta_{k\ell} g_2\left(q_k,t_j\right)}_{\mathrm{L}_{kj\ell}} \beta'_k + \sum_{i=1}^{L} \underbrace{c_i \exp\left[-\Upsilon_i \left(\frac{q_k}{q_o}\right)^n t_j^\xi\right]}_{\mathrm{K}_{ijk}} \Psi(\Upsilon_i)$$

The following example shows application of a q-dependent inverse Gaussian transform to a set of measured XPCS data to yield a single, global distribution. The panel (h) shows the relative $\chi^2(q)$, $a(q)$, and $\beta(q)$.

$$g_2\left(q,t\right) = a(q) + \beta(q) \int_0^\infty \Psi(\Upsilon) \exp\left[-\Upsilon q^1 t^2\right] \, d\Upsilon$$

Two Distributions

Collective analysis of multiple data sets allows integration of models for different degrees of q dependency. The original implementation of

`CONTIN MULTIQ` employed two solution grids – one q^2-dependent and the other q-independent. For XCPS data analysis, `CONTIN XPCS` generalizes the model to include arbitrary values for the time exponent ξ and q-dependence for the distributions $\Psi(\Upsilon)$ and $\Psi'(\Upsilon)$.

$$g_2\left(q,t\right) = a(q)+\beta(q)\int_0^\infty \Psi(\Upsilon)\exp\left[-\Upsilon q^n t^\xi\right]+\Psi'(\Upsilon)\exp\left[-\Upsilon q^{n'} t^{\xi'}\right]\,d\Upsilon$$

As before, `CONTIN` places this in matrix form, where the solution vector has blocks for $\mathbf{a}$, $\boldsymbol{\beta}$, $\boldsymbol{\Psi}$ and $\boldsymbol{\Psi'}$.

$$g_2\left(q_k,t_j\right) = \sum_{\ell=1}^{N}\delta_{i\ell}a'_k+\sum_{\ell=1}^{N}\underbrace{-\delta_{k\ell}g_2\left(q_k,t_j\right)}_{\mathrm{L}_{kj\ell}}\beta'_k+\sum_{i=1}^{L}\underbrace{c_i\exp\left[-\Upsilon_i\left(\frac{q_k}{q_o}\right)^n t_j^\xi\right]}_{\mathrm{K}_{ijk}}\Psi(\Upsilon_i)+\sum_{i=1}^{L}\underbrace{c_i\exp\left[-\Upsilon_i\left(\frac{q_k}{q_o}\right)^{n'} t_j^{\xi'}\right]}_{\mathrm{K}'_{ijk}}\Psi'(\Upsilon_i)$$

Adding a q^0-dependent inverse Gaussian transform to the previous result serves to test the significance of the q^2 dependence.

$$g_2\left(q,t\right) = a(q)+\beta(q)\int_0^\infty \Psi(\Upsilon)\exp\left[-\Upsilon q^1 t^2\right]+\Psi'(\Upsilon)\exp\left[-\Upsilon q^0 t^2\right]\,d\Upsilon$$

Even with the option of using another distribution, `CONTIN` assigns all the intensity to the q-dependent distribution (red), and leaves the q-independent distribution flat.

Use of a model for q-dependency – $S(q)$

The first distribution has the option to include an external model for q-dependency, represented by $S(q)$ from the observation of de Gennes narrowing in XPCS. Click the **S(q) File** button to import a file containing the desired q points and the value of $S(q)$.

The imported $S(q)$ file must have the format shown below, and the range of q values must be equal to or larger than the g_2 q measurement range for internal generation of an interpolation function.

```
0.0356 0.653
0.0358 0.679
0.0360 0.711
.
.
.
0.0569 0.927
0.0571 0.881
0.0573 0.837
```

The $S(q)$ incorporates in the kernel as shown below.

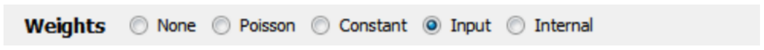
$$g_2\left(q,t\right) = a+\beta\int_0^\infty \Psi(\Upsilon)\exp\left[\frac{-\Upsilon\left(\frac{q_i}{q_o}\right)^n t^\xi}{S\left(\frac{q_i}{q_o}\right)}\right]+\Psi'(\Upsilon)\exp\left[-\Upsilon\left(\frac{q_i}{q_o}\right)^{n'} t^{\xi'}\right]\,d\Upsilon$$

$$g_2\left(q_k,t_j\right) = \sum_{\ell=1}^{N} \delta_{i\ell} a'_k + \sum_{\ell=1}^{N} \underbrace{-\delta_{k\ell} g_2\left(q_k,t_j\right)}_{\mathrm{L}_{kj\ell}} \beta'_k + \sum_{i=1}^{L} \underbrace{c_i \exp\left[\frac{-\Upsilon_i}{S\left(\frac{q_k}{q_o}\right)}\left(\frac{q_k}{q_o}\right)^{n} t_j^{\xi}\right]}_{\mathrm{K}_{ijk}} \Psi(\Upsilon_i) + \sum_{i=1}^{L} \underbrace{c_i \exp\left[-\Upsilon_i\left(\frac{q_k}{q_o}\right)^{n'} t_j^{\xi'}\right]}_{\mathrm{K}'_{ijk}} \Psi'(\Upsilon_i)$$

3.4 Other Parameters

Weights

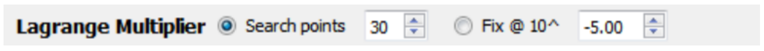
The option **Weights** controls the weighing of the input data in the inverse transformation. Usually, the most practical option weights the data by the input error, though other choices can give different results in certain cases.

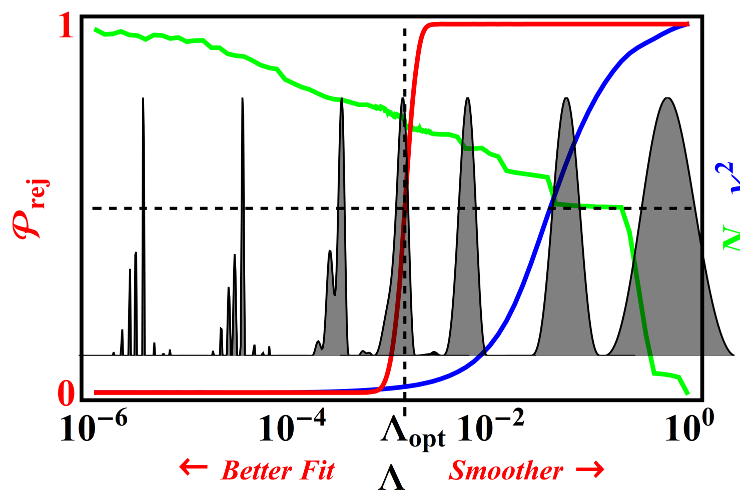

Weights None Poisson Constant [✓ Input] Internal

- `None` - Unweighted analysis
- `Poisson` - Assume variance $\propto \sqrt{g_2(q_i,t_j)}$
- `Constant` - Assume variance $\propto g_2(q_i,t_j)$
- `Input` - Data are weighted by $^1/\sigma(q_i,t_j)$.
- `Internal` - Assume variance $\propto \frac{g_2(q_i,t_j)^2+1}{4g_2(q_i,t_j)^2}$

Lagrange Multiplier

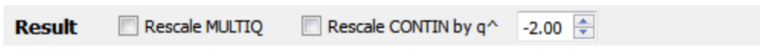
The selection of Lagrange multiplier Λ determines the solution, and likely represents the most critical part of inverse transformation. We find that the internal metric used by `CONTIN` reliably determines the most likely solution in almost all cases. In operation, `CONTIN` performs a broad search over the number of search points, then a more narrow search in the region where $\mathcal{P}_{\text{rej}}$ crosses from zero to one. Alternatively, the user may fix the value of Λ at some value to check the automatically-chosen solution. The figure below shows the search for Λ during inverse Gaussian transformation of the test data.


Lagrange Multiplier [✓ Search points] 30 Fix @ 10^ -5.00



3.5 Result

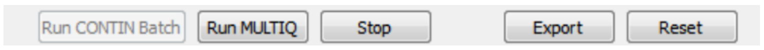
For convenience, the distributions shown initially have Υ with the absolute q dependence divided out using the first q point measured. Checking the box **Rescale MULTIQ** puts the Υ back into units of inverse time.


Result Rescale MULTIQ Rescale CONTIN by q^ -2.00

In the same way, when `CONTIN` analysis gives a series of distributions, the checkbox **Rescale CONTIN by qˆ** will shift the Υ of each distribution by the q point of the measurement according to the q value selected in the spinbox, $\Psi(\Upsilon) \to \Psi(\Upsilon q^n)$.

3.6 Export

The form of the output created by clicking **Export** depends on weather the solution resulted from clicking **Run CONTIN Batch** or **Run MULTIQ**. For **Run CONTIN Batch**, the export consists of a single file output, while **Run MULTIQ** will give two or three.



`CONTIN` Batch Export

The name of the file consists of the sample name with the file extension `.XPCSCONTINbatch`. The `TestData.XPCSCONTINbatch` file from analysis of the test data using a series if inverse transformations contains the overall parameters for the inverse transform, and a list of the `CONTIN` result for each q point. Each q point has a solution for the distribution $\Psi(\Upsilon)$ and a background, selected Lagrange multiplier, degrees of freedom, and standard deviation for each q point as shown below.

```
   Input Data File Name:   TestData
        g2s (q points) ignored in this analysis: []
        Number q points  : 18

CONTIN Batch settings:
   Solution Points:   150
   Kohlrausch Exponent:  2
   Data Weights:         Input Error


    !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
    BEGIN OUTPUT OF DISTRIBUTIONS FOUND FOR EACH Q POINT
    !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

>>> DISTRIBUTION #  1 AT q =  0.0362
   Background:                  0.0051
   log Lagrange Multiplier:     -2.1555
   Degrees of Freedom:          7.7462
   Probability to Reject:       0.4664
   Standard deviation:          0.0008
     log(Gamma)         Magnitude            Error
      --------           --------           --------
        -7.0              7.8e-07            8.06e-06
    -6.90604003          2.24e-06           2.264e-05
    -6.81208017          4.32e-06           4.221e-05
          .  .  .
          .  .  .
          .  .  .
     6.81208641            0.0                0.0
     6.90604624            0.0                0.0
     7.00000569            0.0                0.0
```

Given the discrete points (`Magnitude`) $\Psi(\Upsilon_i)$ approximating the distribution $\Psi(\Upsilon)$ at the points `log(Upsilon)`, $\widehat{\Upsilon} = \log_{10} \Upsilon$, and the background a for the solution at a particular q point, reintegration according to the simple rectangle rule below gives the fit to the data using any external program (Excel, Mathematica, etc.).

$$g_2(t) = 1 + a + (1 - a) \sum_i \Psi(\widehat{\Upsilon}_i) \exp\left[-\left(10^{\widehat{\Upsilon}_i}\right) t^{\xi}\right]$$

Run MULTIQ Export

Collective analysis of all the data sets (MULTIQ) gives two or three files upon export. The first file with extension .XPCSCONTIN contains the global parameters for the inverse transform and the global distribution(s) found without rescaling. CONTIN XPCS writes file containing the rescaled distribution with the extension .XPCSCONTINgrid1, and, if the model uses a second distribution, a second rescaled distribution file .XPCSCONTINgrid2.

A typical *.XPCSCONTIN file appears as follows.

```
   Input Data File Name:   TestData
        g2s (q points) ignored in this analysis: []
        Number q points  : 18

CONTIN MULTIQ settings
   First Distribution   q^2.0-Dependent Kohlrausch Exponent:2
   Second Distribution  q^0.0-Dependent Kohlrausch Exponent:2

   Solution Points:   150
   Background:        Multiple
   Data Weights:      Input Error

Solution:
   Lagrange Multiplier:   -3.3787
   Sum of deivations:     2.2651
   Degrees of freedom:    43.0035
   Probability to reject: 0.4803


                                !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
                                              GLOBAL DISTRIBUTION FOUND
                                !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

Gamma on the first grid is given below as Gamma /  0.036183
Gamma on the second grid is given below as Gamma / 1.0

     log(Gamma)             First Grid                Error                Second Grid              Error
      --------               --------                --------               --------               --------
        -7.0                  0.0                     0.0                  0.01110797             0.05887923
   -6.90604003                0.0                     0.0                  0.01100091             0.05836171
   -6.81208017                0.0                     0.0                  0.0108262              0.0575145
   -6.71812032                0.0                     0.0                  0.0105813              0.05632712
   -6.62416049                0.0                     0.0                  0.01026441             0.05479281
   -6.53020061                0.0                     0.0                  0.00987459             0.0529093
         .  .  .  .  .
         .  .  .  .  .
         .  .  .  .  .

              !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
                     FITTING PARAMETERS FOR EACH q POINT
              !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

         q              Background          % Contrast     Relative Deviation
      --------           --------            --------           --------
       0.0362             0.0037              3.7986             0.1096
       0.0373             0.0043              3.8818             0.1134
       0.0386             0.0043              3.9245             0.1349
       0.0398             0.004               3.9633             0.1118

              !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
                         SEARCH FOR LAGRANGE MULTIPLIER
              !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!


Lagrange Multiplier  Standard Deviation     Deg Freedom     Probability to Reject
      --------           --------            --------           --------
       -7.353             1.7694              45.7037              0.0
       -6.9517            1.7702              46.4936              0.0
       -6.5503             1.77               46.2193              0.0
```

The grid files contain the distributions with the units of the abscissa given by $\widehat{\Upsilon}$.

$$\widehat{\Upsilon}_i = \log_{10} \Upsilon_i {q_1}^n \tag{13}$$

Contents of the output file `TestData.XPCSCONTINgrid1`

```
log(Gamma)          Distribution          Error
 -8.4415                0.0                0.0
 -8.34754               0.0                0.0
 -8.25358               0.0                0.0
     .  .       .
     .  .       .
     .  .       .
 5.37059               3e-05             0.00017
 5.46455               3e-05             0.00017
 5.55851               3e-05             0.00018
```

Reintegration of the output file for $g_2(q_k, t)$ at the k^{th} q point uses `log(Upsilon)` $= \widehat{\Upsilon}_i$ and the `Distribution` $= \Psi(\widehat{\Upsilon}_i)$ from the `TestData.XPCSCONTINgrid` file using a rectangular rule and valeus for background a and the scale factors β in a ratio from the `TestData.XPCSCONTIN` file, as shown below. Usually the parameter ν_k equals unity, but occasionally requires additional *ad hoc* multiplicative scaling of the integrated result by adjusting ν_k for each q point.

$$g_2(k,t) = a_k + \frac{\nu_k \beta_{\text{last}}}{\beta_k} \sum_i \Psi(\widehat{\Upsilon}_i) \exp\left[- \left(\frac{10^{\widehat{\Upsilon}_i}}{q_1^n} \right) \left(\frac{q_k}{q_1} \right)^n t^{\xi} \right]$$

Upon checking the **Second Distribution** checkbox, the export includes the file `TestData.XPCSCONTINgrid2` containing the second distribution. The files gives Υ in terms of reduced units

$$\widehat{\Upsilon'}_i = \log_{10} \Upsilon_i {q_1}^{n'} \tag{14}$$

```
log(Gamma)          Distribution          Error
   -7.0               3e-05             0.00017
 -6.90604             3e-05             0.00017
 -6.81208             3e-05             0.00017
     .  .       .
     .  .       .
     .  .       .
 6.81209               0.0                0.0
 6.90605               0.0                0.0
 7.00001               0.0                0.0
```

Reintegration of the output file for $g_2(q_k, t)$ at the k^{th} q point using two solution distributions uses `log(Upsilon)` $= \widehat{\Upsilon}_i$ and the `Distribution` $= \Psi(\widehat{\Upsilon}_i)$ from the `.XPCSCONTINgrid1` file and `log(Upsilon)` $= \widehat{\Upsilon}_i$ and the `Distribution` $= \Psi'(\widehat{\Upsilon}_i)$ from the `.XPCSCONTINgrid2` file. Again, the background a and the contrasts β in a ratio come from the `.XPCSCONTIN` file.

$$g_2(k,t) = a_k + \frac{\nu_k \beta_{\text{last}}}{\beta_k} \left[\sum_i \Psi(\widehat{\Upsilon}_i) \exp\left[- \left(\frac{10^{\widehat{\Upsilon}_i}}{q_1^n} \right) \left(\frac{q_k}{q_1} \right)^n t^{\xi} \right] + \sum_i \Psi'(\widehat{\Upsilon}_i) \exp\left[- \left(\frac{10^{\Upsilon_i}}{q_1^{n'}} \right) \left(\frac{q_k}{q_1} \right)^{n'} t^{\xi'} \right] \right]$$