



**HAL**  
open science

# Inelastic and quasielastic neutron scattering data analysis

Tilo Seydel

► **To cite this version:**

Tilo Seydel. Inelastic and quasielastic neutron scattering data analysis. École thématique. SwedNess specialized course in neutron spectroscopy 2021, Sweden. 2021. hal-03842803

**HAL Id: hal-03842803**

**<https://hal.science/hal-03842803v1>**

Submitted on 7 Nov 2022

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.


L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution - NonCommercial - NoDerivatives 4.0 International License

## Neutron Spectroscopy Inelastic and quasielastic neutron scattering

### Data analysis

1. The scattering law  $S(Q,\omega)$  and related  $I(Q,t)$  and  $G(r,t)$
2. Basic theory of inelastic neutron scattering. Different types of excitations.
3. Basic theory of quasielastic neutron scattering
4. Magnetic scattering
5. Different types of motions and how they can be distinguished
6. Different types of neutron spectrometers, including the spin-echo technique
7. Coherent and incoherent scattering of crystalline and amorphous materials
8. Examples of science, where inelastic and quasielastic neutron scattering are useful tools, including a description of the two hands-on experiments at ISIS
-  **9. Data analysis. Corrections and analysis of real experimental data**
10. Complementary molecular dynamics simulations

# (0) – Welcome



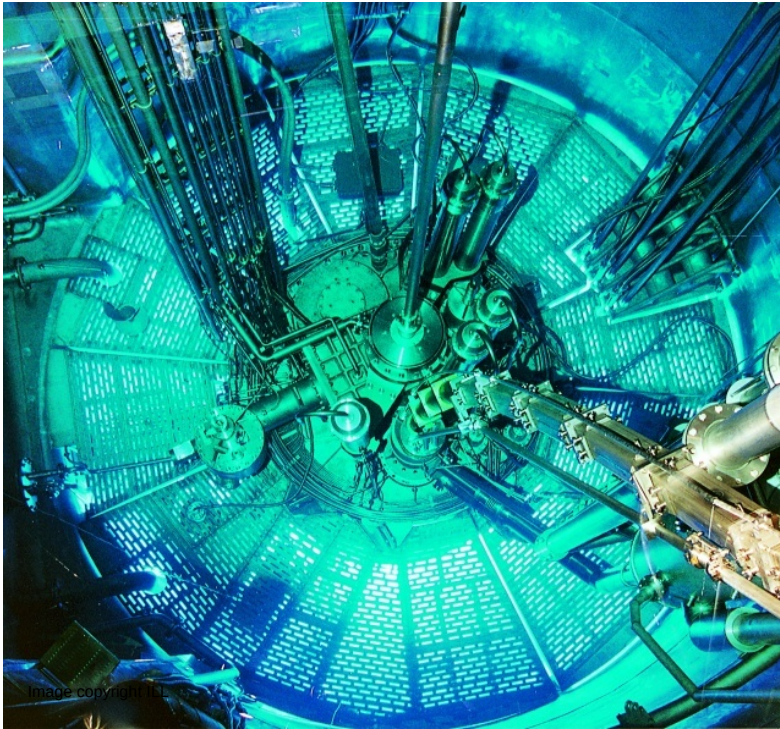
Image copyright ILL

## IN16B

“En effet au moment de la conférence de Genève [1964], on ne savait pas si une telle source coûterait 50 ou 500 millions de francs. Les ministres déclarèrent qu’il ne fallait pas s’arrêter à ce détail et qu’on verrait bien.”

B.Jacrot, Des neutrons pour la science, Les Ulis 2006

# (0) – Welcome



Brightest continuous source  
of thermal neutrons  
that can be built at reasonable cost

- Mean free path of a fission neutron  $\sim 10$  cm
  - Thermalization time  $\sim 1$  ms
  - Fission: 190 MeV per available neutron
  - ILL: 1.5 MW/l (limit for stationary system)
  - Spallation: 25 MeV per available neutron
  - Diffusion equation governs moderation
- F. Mezei, J. Neutron Research 6, 3 (1997)

ILL reactor:

- Single fuel element
- $D_2O$  moderator for thermal neutrons
- $\sim 29$  collisions from 2MeV to 1eV
- Scattering mean free path  $\sim 3$  cm
- Absorption mean free path  $\sim 3.3$  m
- Water entrance  $T \approx 30^\circ C$ , exit  $T \approx 48^\circ C$

## Data analysis

### Corrections / reduction of real experimental data

"From raw data to a complete fit, step-by-step", a few notes on Mantid, hdf and python in passing

From standard corrections to some examples of how the corrected data can be fitted and analyzed

- monitor normalization
- empty can contribution, self-shielding
- Vanadium normalization
- standard time-of-flight data reduction
- detailed balance
- resolution function (spectrometer response function)
- data fitting, maximum likelihood estimation,  $\chi^2$  minimization and Bayesian analysis
- examples, fun math



# (1) – Outline

In practice ...

[www.mantidproject.org](http://www.mantidproject.org)



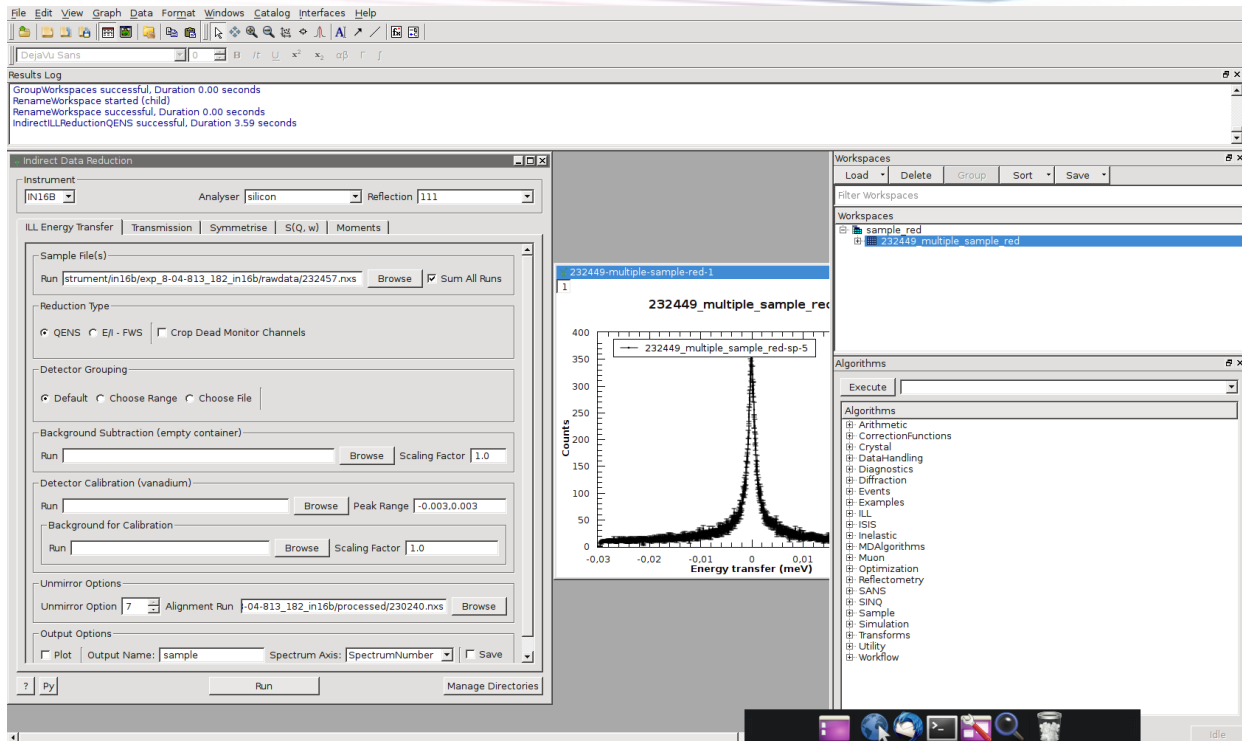
Algorithms can be used  
in python  
(no need for any “GUI”):

```
import sys
```

```
sys.path.append( '/opt/Mantid/bin/' )
```

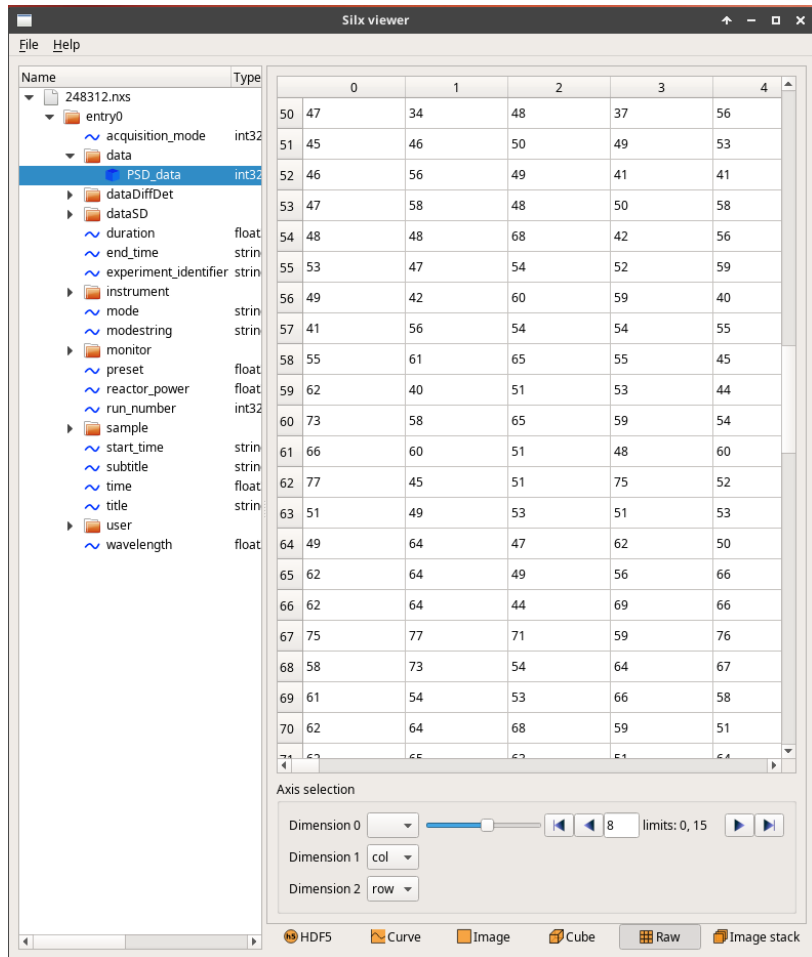
```
# then import Mantid application programming interface (API):
```

```
import mantid.simpleapi as mtd
```



# (1) – Outline

## The “hdf” data format



The screenshot shows the SiIx viewer interface. On the left is a tree view of a file named '248312.nxs'. The tree includes folders like 'entry0', 'data', 'instrument', 'monitor', 'sample', and 'user'. Under 'data', there is a folder 'PSD\_data' which is selected. The main area displays a grid of data with columns labeled 0 through 4 and rows numbered 50 through 70. The data values are integers. At the bottom, there is an 'Axis selection' panel with dropdowns for 'Dimension 0', 'Dimension 1', and 'Dimension 2', and a slider for 'Dimension 0' set to 8. The bottom status bar shows icons for HDF5, Curve, Image, Cube, Raw, and Image stack.

|    | 0  | 1  | 2  | 3  | 4  |
|----|----|----|----|----|----|
| 50 | 47 | 34 | 48 | 37 | 56 |
| 51 | 45 | 46 | 50 | 49 | 53 |
| 52 | 46 | 56 | 49 | 41 | 41 |
| 53 | 47 | 58 | 48 | 50 | 58 |
| 54 | 48 | 48 | 68 | 42 | 56 |
| 55 | 53 | 47 | 54 | 52 | 59 |
| 56 | 49 | 42 | 60 | 59 | 40 |
| 57 | 41 | 56 | 54 | 54 | 55 |
| 58 | 55 | 61 | 65 | 55 | 45 |
| 59 | 62 | 40 | 51 | 53 | 44 |
| 60 | 73 | 58 | 65 | 59 | 54 |
| 61 | 66 | 60 | 51 | 48 | 60 |
| 62 | 77 | 45 | 51 | 75 | 52 |
| 63 | 51 | 49 | 53 | 51 | 53 |
| 64 | 49 | 64 | 47 | 62 | 50 |
| 65 | 62 | 64 | 49 | 56 | 66 |
| 66 | 62 | 64 | 44 | 69 | 66 |
| 67 | 75 | 77 | 71 | 59 | 76 |
| 68 | 58 | 73 | 54 | 64 | 67 |
| 69 | 61 | 54 | 53 | 66 | 58 |
| 70 | 62 | 64 | 68 | 59 | 51 |
| 71 | 63 | 65 | 63 | 54 | 64 |

```
import h5py
```

```
def loadIN5( filename ): # read lamp-reduced IN5 data
    class out:
        pass
    f = h5py.File( filename, 'r' )
    out.hw = np.transpose( np.array( f['/entry1/data1/X/'] ) )
    out.q = np.transpose( np.array( f['/entry1/data1/Y/'] ) )
    out.sqw = np.transpose( np.array( f['/entry1/data1/DATA/'] ) )
    out.dsqw = np.transpose( np.array( f['/entry1/data1/errors/'] ) )
    return( out )
```

```
def loadMantid( filename ): # read Mantid-reduced data
    class out:
        pass
    f = h5py.File( filename, 'r' )
    out.x = np.transpose( np.array( f['/mantid_workspace_1/workspace/axis1/'] ) )
    out.q = np.transpose( np.array( f['/mantid_workspace_1/workspace/axis2/'] ) )
    out.sqw = np.transpose( np.array( f['/mantid_workspace_1/workspace/values/'] ) )
    out.dsqw = np.transpose( np.array( f['/mantid_workspace_1/workspace/errors/'] ) )
    out.hw = ( out.x[0:-1] + out.x[1:] ) / 2.0 # from Mantid bins to channel center
    return( out )
```

## (2) – Monitor normalization

Normalization to the  
time-dependent  
incident neutron flux

Errors:

$$\delta I_k = \Delta \left( \frac{N_k}{M_k} \right) = \frac{\sqrt{N_k}}{M_k} + \frac{N_k \sqrt{M_k}}{M_k^2}$$

detector

monitor

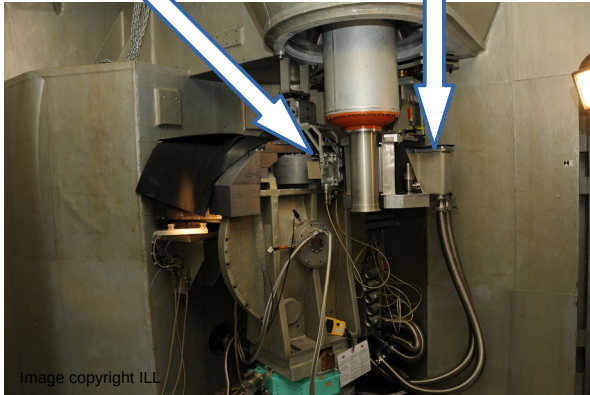
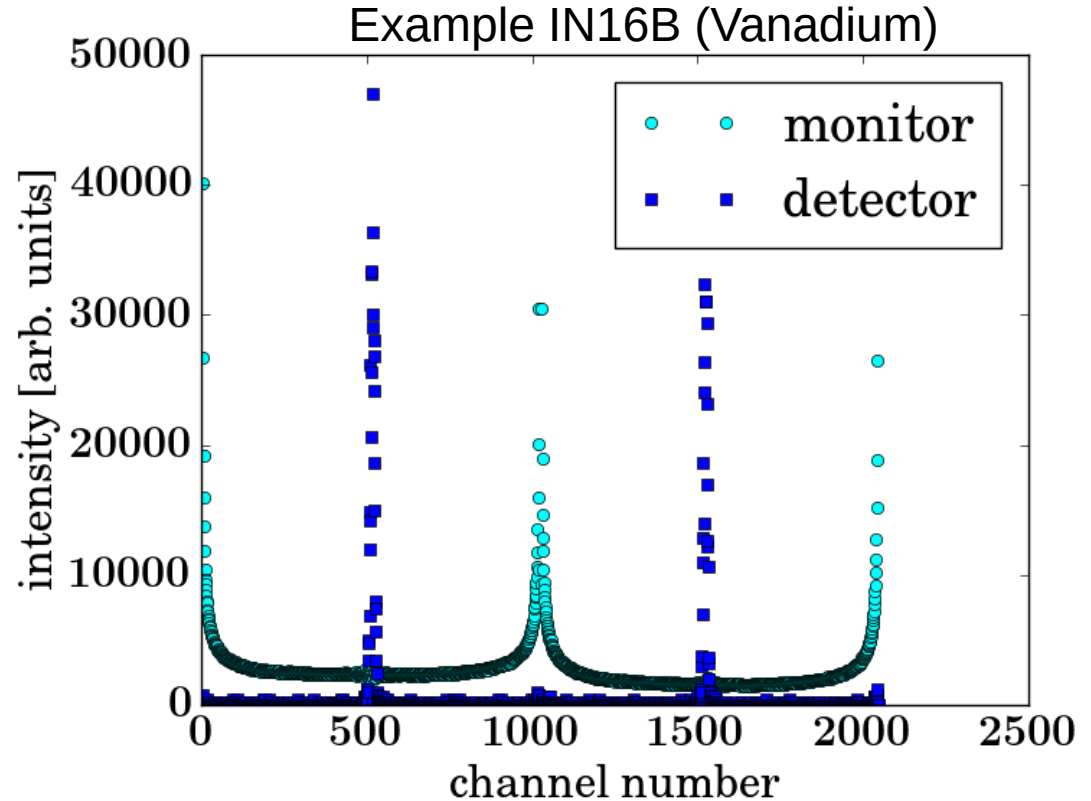
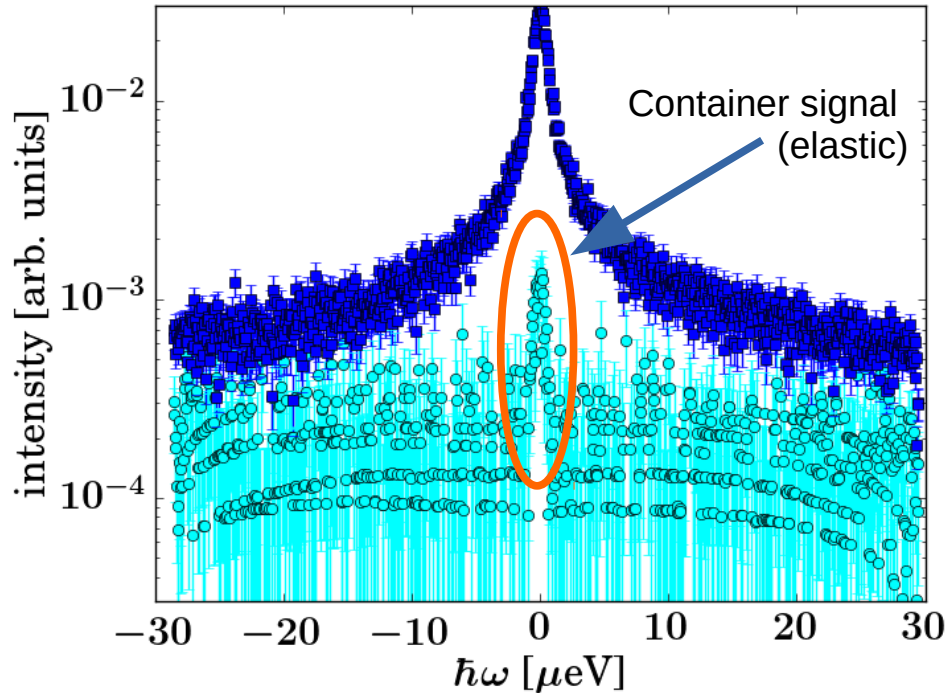


Image copyright ILL





### Sample container contribution



(example: IN16B protein in  $\text{D}_2\text{O}$ ,  $q=0.82/\text{\AA}$ ,  $T=280\text{K}$ )

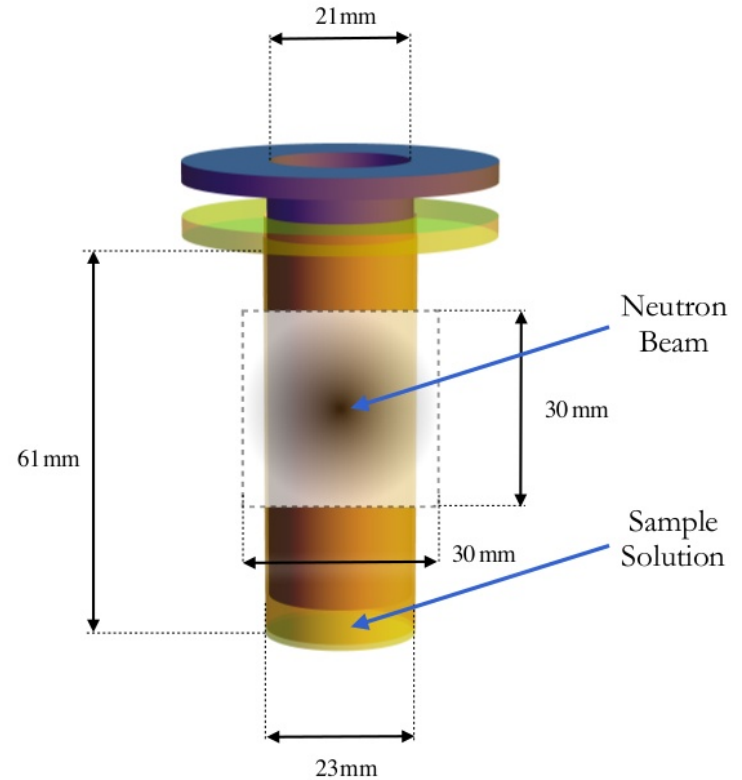


Figure from M. Hennig, PhD thesis, University of Tübingen, Germany, 2011

### (3) – Empty can contribution, self-shielding

S: Sample, C: Container,  $I_C^C$ : Container scattering

$I_{SC}^{SC}$ : Sample+Container scattering

$$I^S = \frac{1}{A_{SC}^S} \left( I_{SC}^{SC} - \frac{A_{SC}^C}{A_C^C} I_C^C \right)$$

$A_{SC}^S, A_{SC}^C, A_C^C$ :  $q$ -dependent Paalman-Pings coefficients

$$\alpha_{SC} := \frac{1}{A_{SC}^S}$$

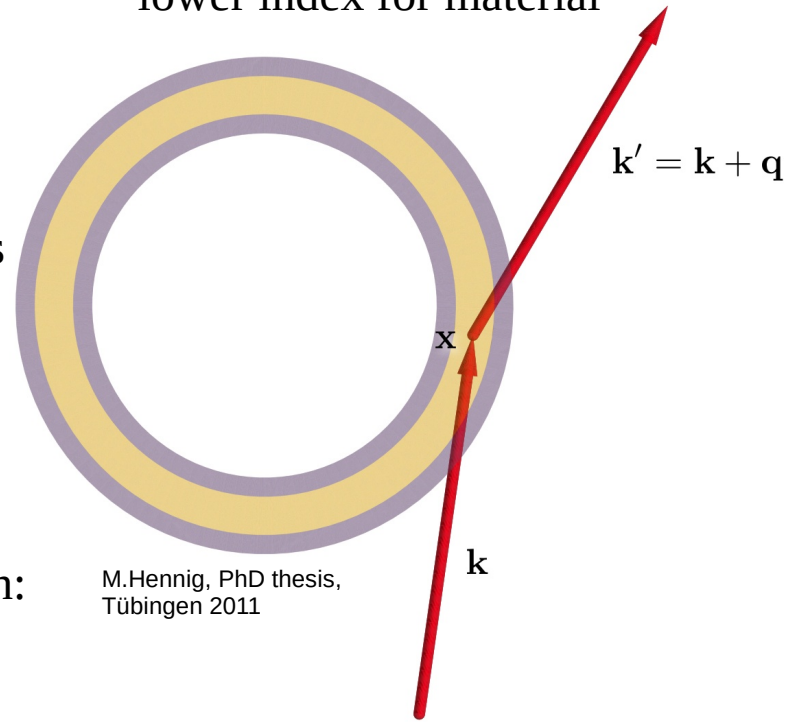
$$\beta_{SC} := \frac{1}{A_{SC}^S} \frac{A_{SC}^S}{A_C^C}$$

Attenuation of the neutron beam intensity along its path:

$$A_{\Sigma}^V(\mathbf{q}) = \frac{1}{V} \int_V \exp \left[ - \int_{\gamma(\mathbf{x})} \Sigma(\mathbf{x}') ds(\mathbf{x}') \right] d^3 \mathbf{x}$$

Volume  $V$ , path  $\gamma(\mathbf{x})$ , attenuation coefficient  $\Sigma$ , infinitesimal line element  $ds(\mathbf{x})$

$A_{XY}^X$ : upper index for volume  
lower index for material



H.Paalman, C.Pings; J.Appl.Phys. 1962

### (3) – Empty can contribution, self-shielding

S: Sample, C: Container,  $I_C^C$ : Container scattering

$I_{SC}^{SC}$ : Sample+Container scattering

$$I^S = \frac{1}{A_{SC}^S} \left( I_{SC}^{SC} - \frac{A_{SC}^C}{A_C^C} I_C^C \right)$$

$A_{SC}^S, A_{SC}^C, A_C^C$ :  $q$ -dependent Paalman-Pings coefficients

SC:  $\Sigma = \Sigma_S \Omega_S(\mathbf{x}) + \Sigma_C \Omega_C(\mathbf{x})$

$$\Omega_S(\mathbf{x}) = 1 \text{ for } \mathbf{x} \in V$$

$$\Omega_S(\mathbf{x}) = 0 \text{ for } \mathbf{x} \notin V$$

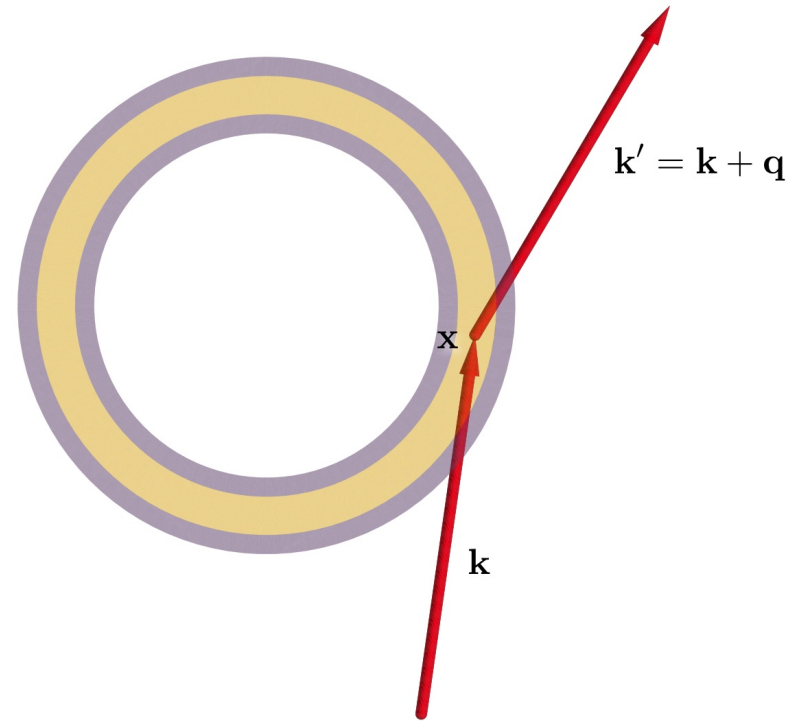
Path for neutron being scattered at  $\mathbf{x}$ :

$$\gamma(\mathbf{x}) = \mathbf{x} + t \mathbf{k} / k \quad \text{for } t \leq 0$$

$$\gamma(\mathbf{x}) = \mathbf{x} + t \mathbf{k}' / k' \quad \text{for } t > 0$$

with  $t \in [-\infty, \infty]$

$$\Sigma = \rho(\sigma_c + \sigma_i + \sigma_a)$$



### (3) – Empty can contribution, self-shielding

S: Sample, C: Container,  $I_C^C$ : Container scattering

$I_{SC}^{SC}$ : Sample+Container scattering

$$I^S = \frac{1}{A_{SC}^S} \left( I_{SC}^{SC} - \frac{A_{SC}^C}{A_C^C} I_C^C \right)$$

$A_{SC}^S, A_{SC}^C, A_C^C$ :  $q$ -dependent Paalman-Pings coefficients

All together, including detector efficiency correction:

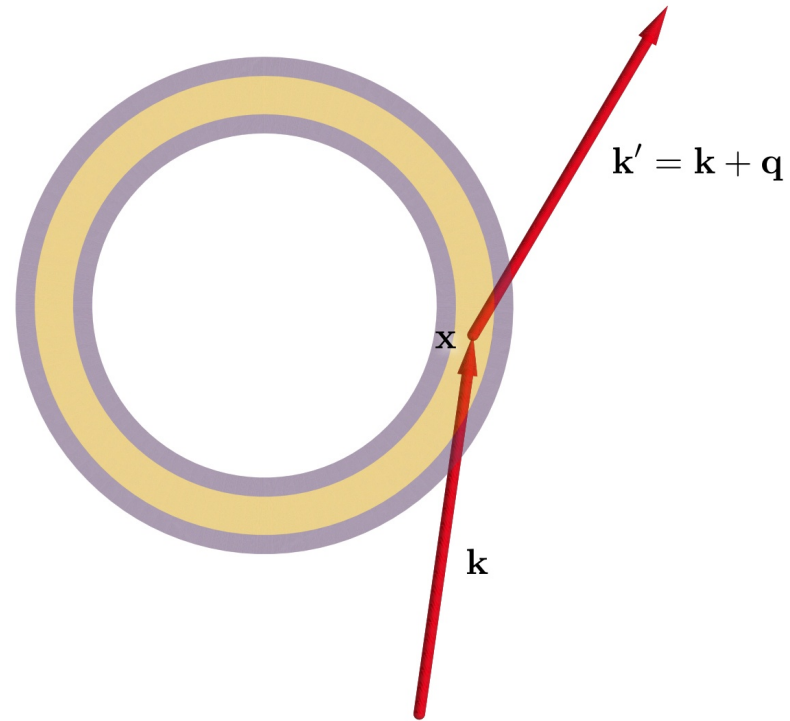
$$I^S(q) = \frac{1}{D(q)} \left( \alpha(q) I_{SC}^{SC}(q) - \beta(q) I_C^C(q) \right)$$

with

$$\alpha(q) = \frac{1}{A_{SC}^S(q)}$$

$$\beta(q) = \frac{1}{A_{SC}^S(q)} \frac{A_{SC}^S(q)}{A_C^C(q)}$$

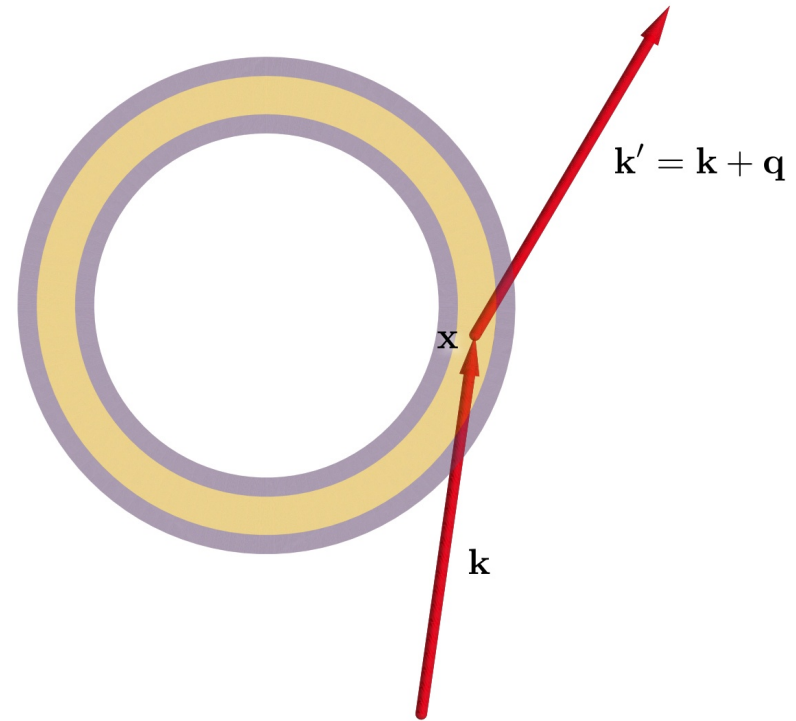
and the  $q$ -dependent detector efficiency  $D(q)$



### (3) – Empty can contribution, self-shielding

Examples ( $\lambda=6.4\text{\AA}$ ):

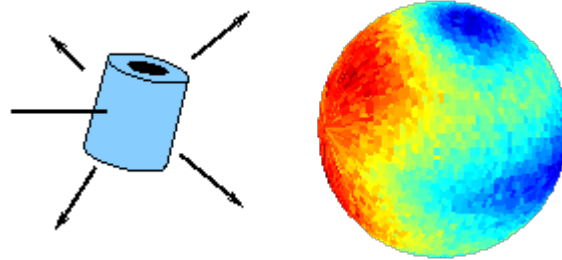
| Sample  | Density [g/cm <sup>3</sup> ] | $\Sigma$ [1/cm] (cf. slide 9) |
|---|------------------------------|-------------------------------|
| Vanadium foil   | 6.11                         | 1.673                         |
| Aluminum sample container                             | 2.7                          | 0.147                         |
| Water (D <sub>2</sub> O) with protein (BSA, 200mg/ml) | 1.174                        | 0.708                         |
| Pure D <sub>2</sub> O                                 | 1.1056                       | 0.649                         |





Limitation of the standard correction: No account for divergent beam and beam spot

Solution: Ray-tracing “Monte Carlo” simulation, e.g. in McStas



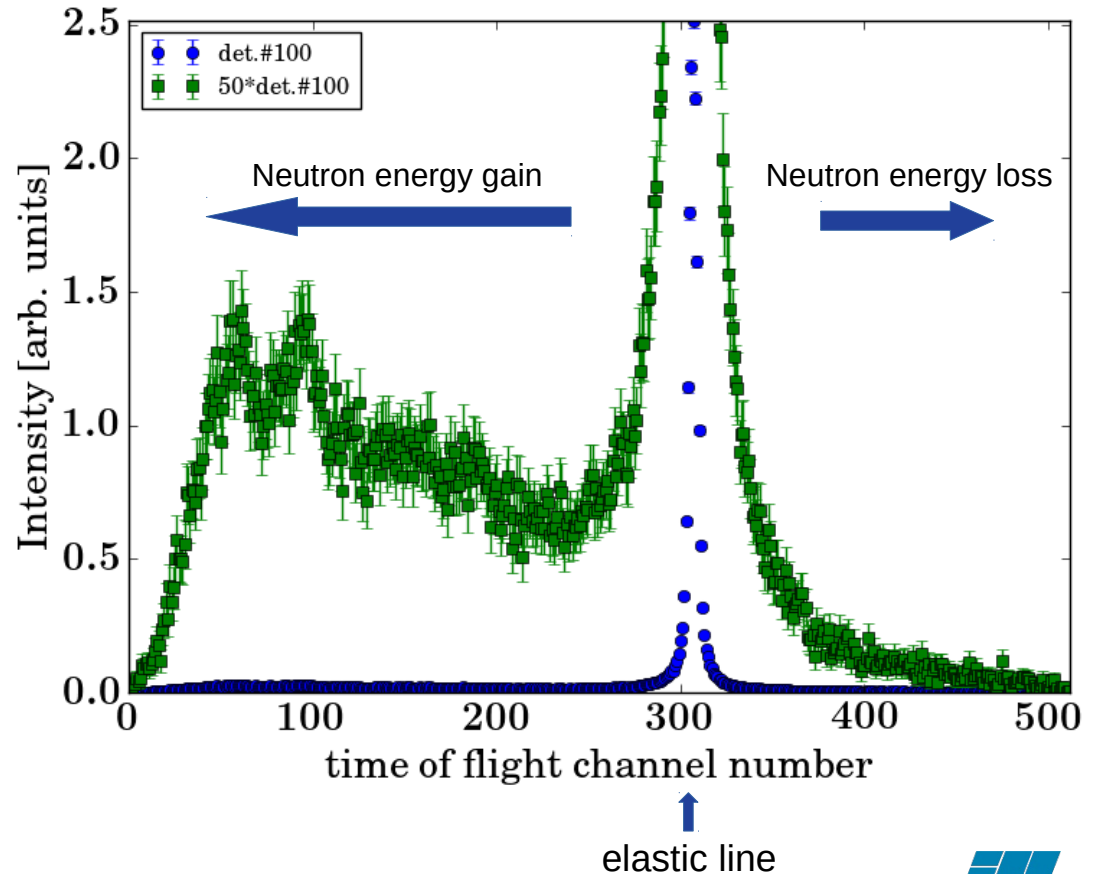
### Simulated scattering from a Vanadium cylinder

The sample is at the center of the sphere with the neutron beam coming from the left.

[www.mcstas.org](http://www.mcstas.org)

## (4) – Interpolation to $S(q,\omega)$

Example: Time-of-flight spectrum  
Individual detector on IN5  
(Cold neutron ToF spectrometer)  
Sample: Protein in  $D_2O$ ,  $T=280K$   
Container signal not subtracted



### From time-of-flight (ToF) data to $S(q, \omega)$

'Handwaving' derivation of the conversion of raw ToF data to  $S(q, \omega)$  based on differential scattering cross sections in time  $t$  and energy  $\omega$

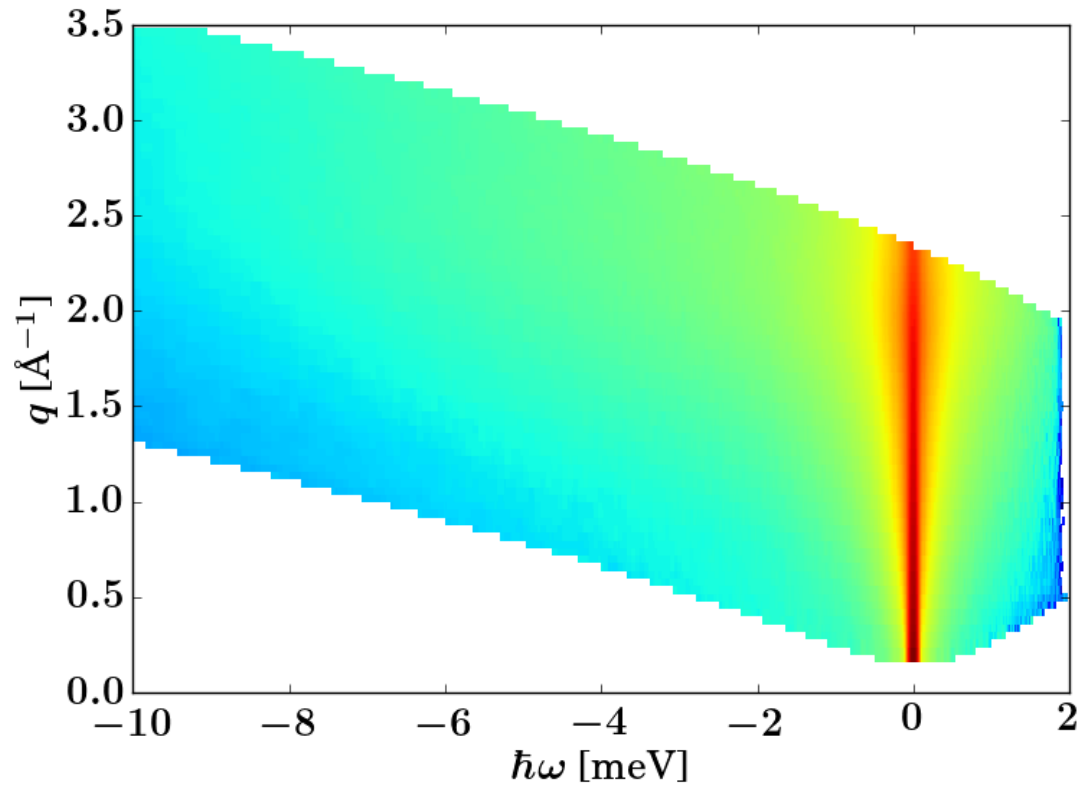
$$S(q, \omega) \propto \frac{k_i}{k_f} \frac{\partial^2 \sigma}{\partial \omega \partial \Omega} \propto \frac{k_i}{k_f} \frac{\partial t}{\partial \omega} \frac{\partial^2 \sigma}{\partial t \partial \Omega} \propto t^{-4} \frac{\partial^2 \sigma}{\partial t \partial \Omega} \quad \text{using} \quad \partial \omega = \frac{\partial \omega}{\partial t} \partial t$$

$$k_f \propto m \frac{s}{t} \quad \omega = \frac{1}{2} m v^2 = \frac{1}{2} m \left( \frac{s}{t} \right)^2 \quad \frac{\partial \omega}{\partial t} \propto t^{-3}$$

$s$ : flight path sample to detector;  $v$ : neutron velocity  
 $\partial \Omega$ : solid angle element

IN5 data interpolated to  $S(q,\omega)$

(Example: protein in  $D_2O$  → QENS)

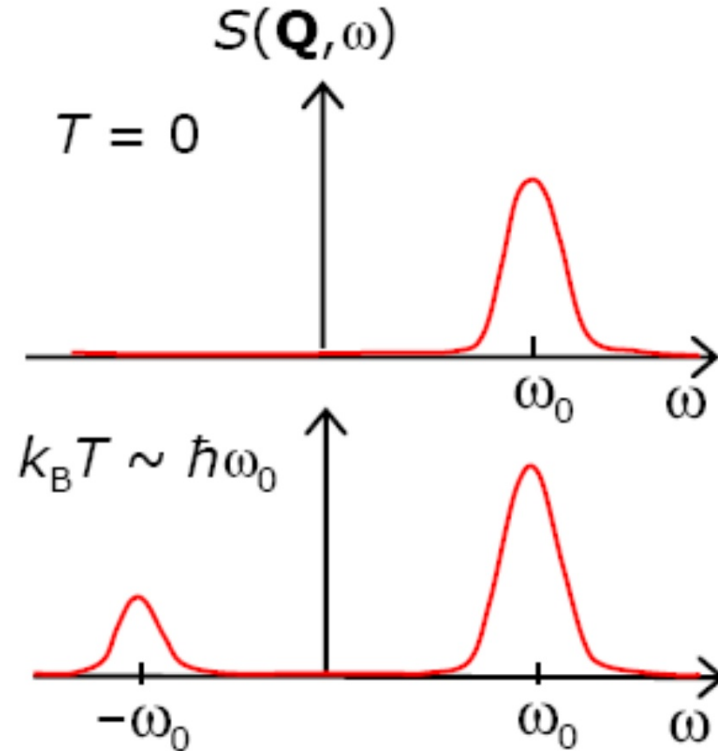


Principle of detailed balance

$$S(q, -\omega) = \exp\left(-\frac{\hbar\omega}{k_B T}\right) S(q, \omega)$$

Neutron  
energy gain

Neutron  
energy loss





The inelastic part of the spectrum – density of states

$$\frac{\partial^2 \sigma_{inc}}{\partial \Omega \partial \omega} = \frac{k_f}{k_i} b_{inc}^2 \exp\left(-\frac{\hbar \omega}{2 k_B T}\right) \tilde{S}_s(q, \omega) \quad \text{one-phonon incoherent differential cross section}$$

reduced variables:  $\alpha = \frac{\hbar^2 q^2}{2 M k_B T}$  identical atoms, mass M  $\beta = \frac{\hbar \omega}{k_B T}$

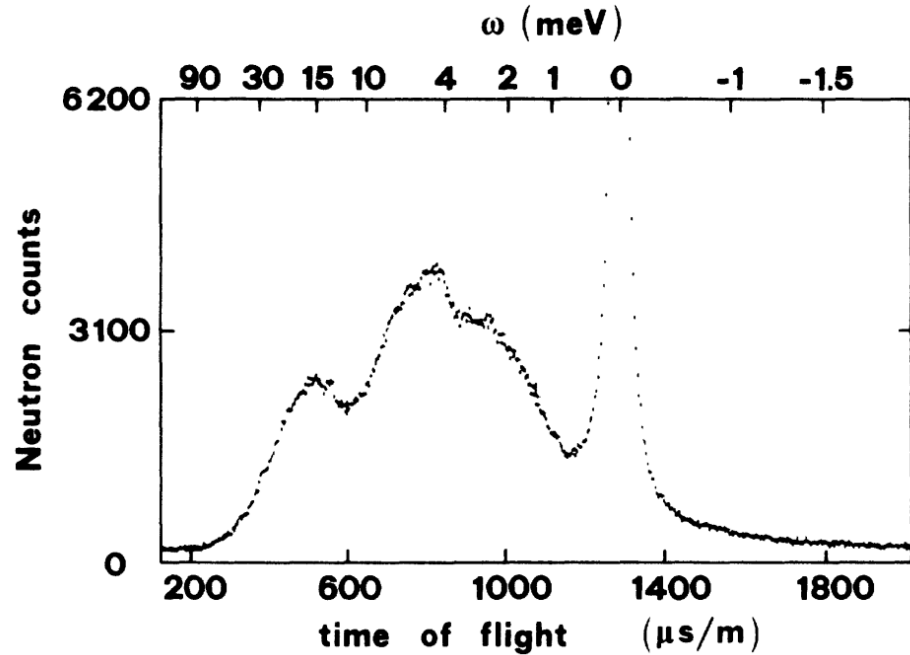
$$\tilde{S}_s(\alpha, \beta) = \exp(-q^2 \langle u^2 \rangle) \frac{\alpha}{2 \beta \sinh(\beta/2)} g(\omega) \quad \text{with the density of states } g(\omega)$$

$$P(\alpha, \beta) = 2 \beta \sinh(\beta/2) \frac{\tilde{S}_s(\alpha, \beta)}{\alpha} \quad \text{generalized frequency distribution}$$

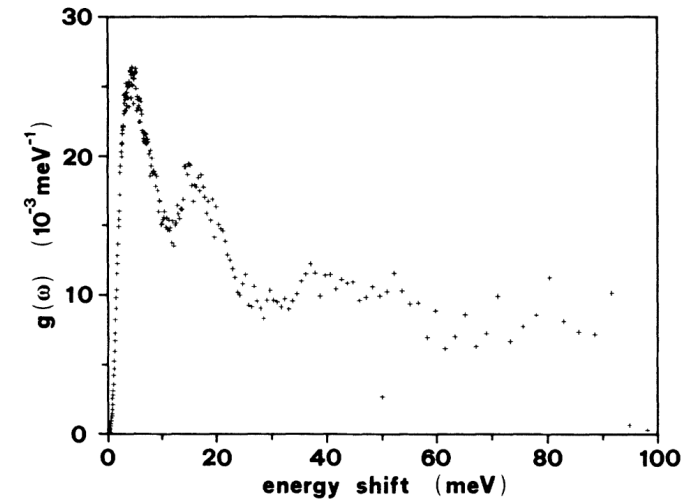
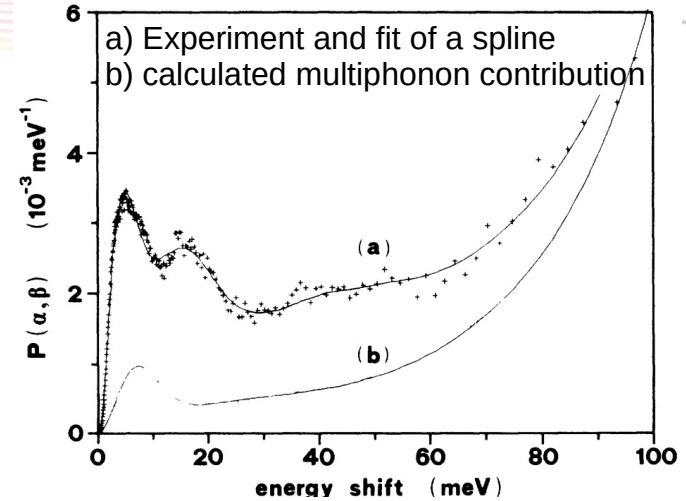
$$g(\omega) = 2 \beta \sinh(\beta/2) \lim_{\alpha \rightarrow 0} \left( \frac{\tilde{S}_s(\alpha, \beta)}{\alpha} \right) \quad \text{approximation neglecting multiphonon contributions}$$

## (4) – Interpolation to $S(q,\omega)$

The inelastic part of the spectrum – density of states



ToF spectrum on  $^{11}\text{B}$  borate glass, IN6,  $5.1\text{\AA}$ , 176K  
A.Fontana et al., Phys.Rev.B 41, 3778 (1990)

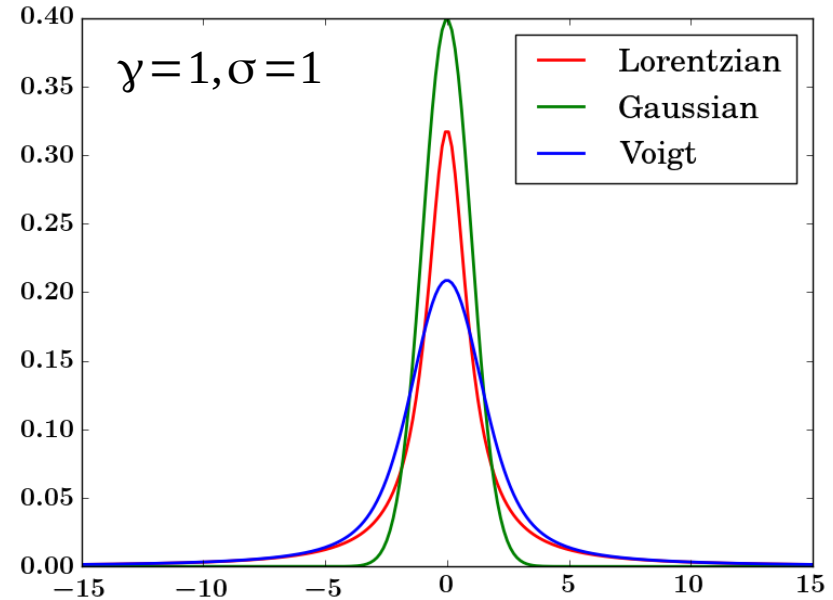


## Important functions

Gaussian: 
$$g(x; \sigma) = \frac{1}{\sigma \sqrt{2\pi}} \exp\left(-\frac{1}{2} \frac{x^2}{\sigma^2}\right)$$

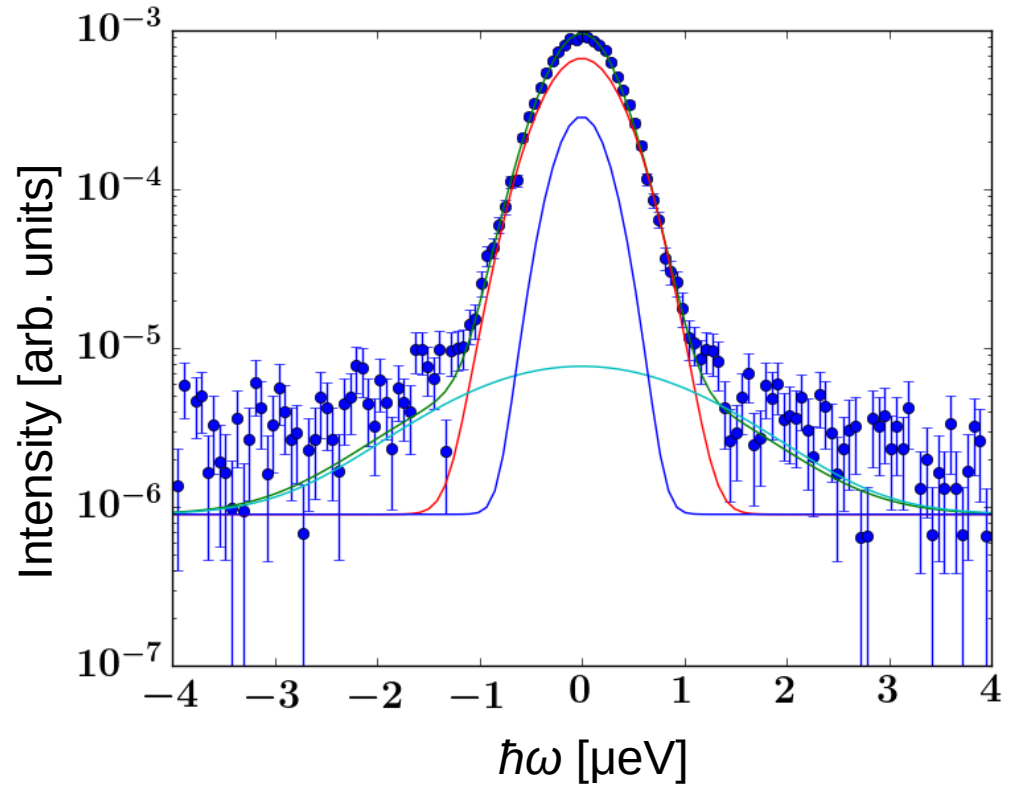
Lorentzian: 
$$l(x; \gamma) = \frac{1}{\pi} \frac{\gamma}{x^2 + \gamma^2}$$

Voigt: 
$$v(x; \sigma, \gamma) = \int_{-\infty}^{\infty} g(x'; \sigma) l(x - x'; \gamma) dx'$$



```
def V(x, sigma, gamma): # Voigt line shape
    import numpy as np
    from scipy.special import wofz # Faddeeva function exp(-z**2)*erfc(-i*z), erfc = 1-erf
    return np.real( wofz( (x + 1j*gamma) / sigma / np.sqrt(2.) ) ) / sigma / np.sqrt( 2.*np.pi )
```

Fit of the resolution function by  
a sum of Gaussians

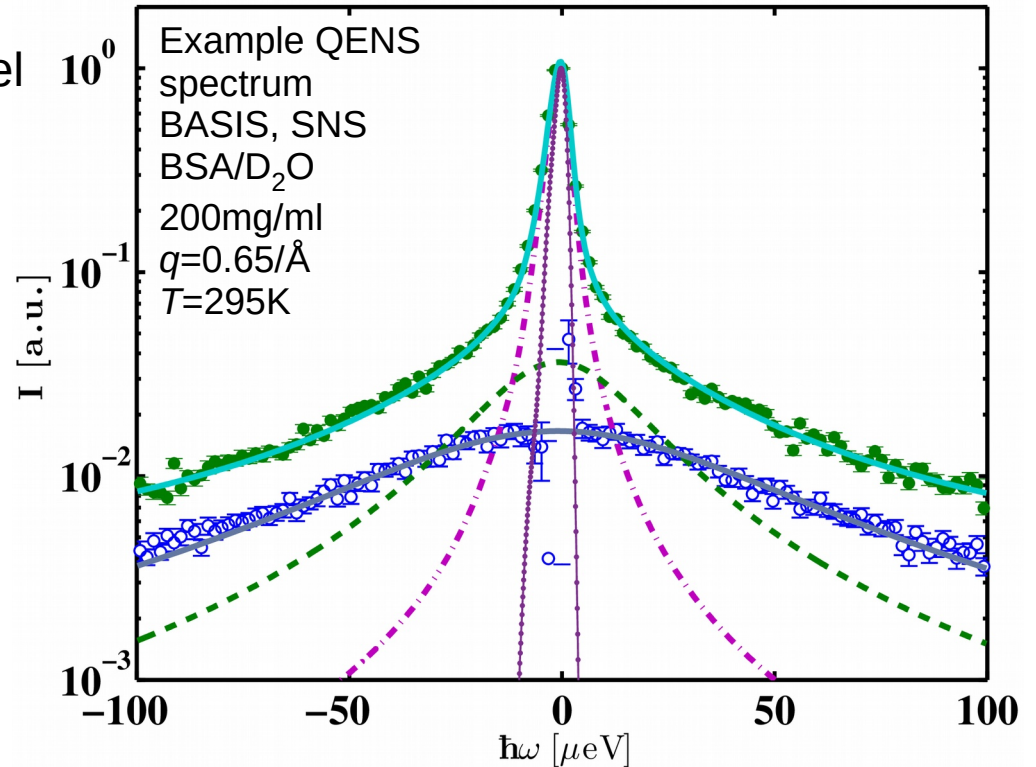


## (5) – Resolution function

Build the instrumental resolution model  
from a sum of Gaussians;

Describe the physics by Lorentzians

=> Analytical convolution



Experimental Methods in the Physical Sciences Volume 49; F.Fernandez-Alonso, D.L.Price eds.; Neutron Scattering - Applications in Biology, Chemistry, and Materials Science; Elsevier Academic Press 2017; chapter 2



## Fitting

Non-linear least squares minimization (iterative application of linear least squares fitting to a linearized form)

“A mathematical procedure for finding the best-fitting curve to a given set of points by minimizing the sum of the squares of the offsets (‘the residuals’) of the points from the curve.”

mathworld.wolfram.com

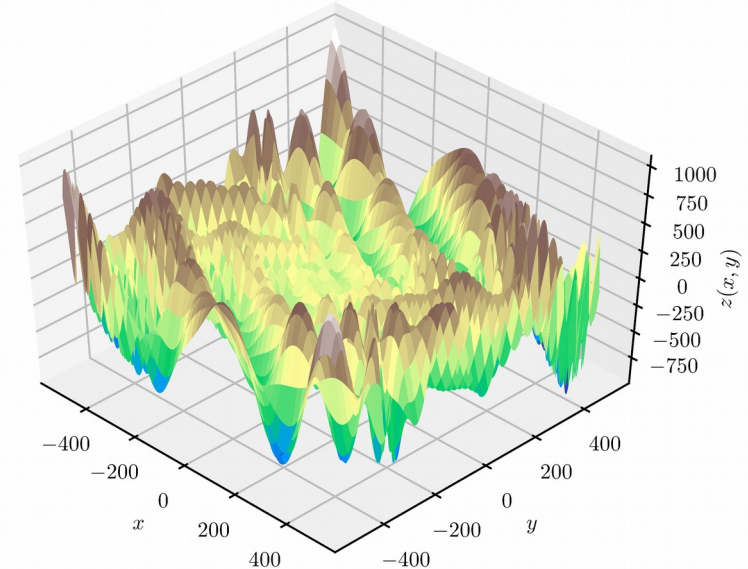
Error-weighted fits

“A measurement without error is meaningless.”

Machine learning is a kind of generalized fitting ...

“To build truly intelligent machines, teach them cause and effect.”

J.Pearl, Quanta Magazine 2018



<https://docs.scipy.org/doc/scipy/tutorial/optimize.html>

# Fitting

Useful reading (pdf via google scholar):

J. Orear: **Notes on Statistics for Physicists**

Laboratory for Nuclear Studies, Cornell University, Ithaca, NY 14853

July 28, 1982 (revised edition based on the original lecture series in 1958)

“Usually the physicist does an experiment and quotes a result  $a = a^* \pm \Delta a$ .

‘What do we mean by  $a^*$  and  $\Delta a$  ?’ and ‘What is the best way to calculate  $a^*$  and  $\Delta a$  ?’.

These are questions of extreme importance to all physicists.”

“What the physicist usually means is that the ‘probability’ of finding  $(a^* - \Delta a) < a_0 < (a^* + \Delta a)$  is 68.3 % (standard deviation).”

The more accurate term would be “**inverse probability**”.

It is common in physics to have an infinite set of hypotheses; i. e., a parameter that is a continuous variable  $f(a;x)$ .

## (6) – Non-linear least squares minimization

Likelihood function: joint probability density of a particular experimental result,  $x_1; x_2; \dots; x_N$ , assuming  $f(a;x)$  is the true normalized distribution function:

$$L(a) = \prod_{i=1}^N f(a; x_i) \quad \int f(a; x) dx = 1 \quad \Delta a = \left[ \frac{\int (a - a^*)^2 L(a) da}{\int L(a) da} \right]^{\frac{1}{2}}$$

The most probable value  $a^*$  of  $a$  is called maximum likelihood solution of  $L(a)$ .

Maximum likelihood theorem:  $N \rightarrow \infty : a^* \rightarrow a_0$   $N \rightarrow \infty : L(a) \sim \exp \left[ -\frac{h}{2} (a - a^*)^2 \right]$

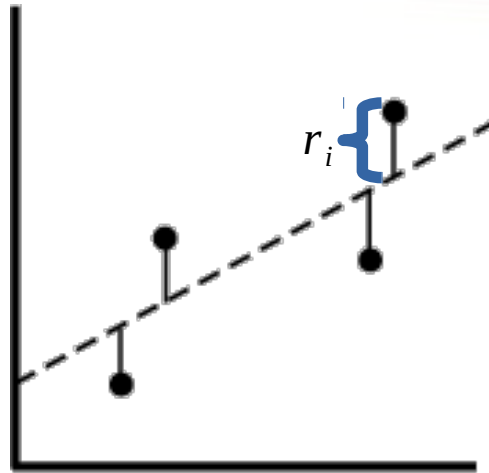
$$\Delta a = \frac{1}{\sqrt{h}} \quad w := -\frac{h}{2} (a - a^*)^2 + \text{const.} \quad \frac{\partial w}{\partial a} = -h (a - a^*)$$

Maximum likelihood error:  $-\frac{\partial^2 w}{\partial a^2} = h = \Delta a$

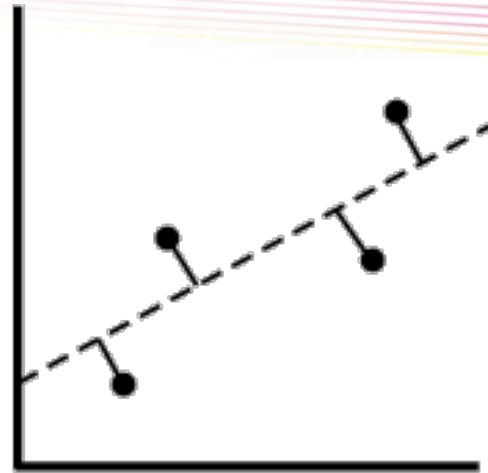
Let  $P(N,x)$  be the probability of finding  $N$  events, if the “true” count rate is  $\bar{N}$ .

$$P(N, \bar{N}) = \frac{\bar{N}^N}{N!} e^{-\bar{N}} \quad \text{Poisson distribution} \quad L(a) = \frac{a^N}{N!} e^{-a} \quad w = N \ln a - a - \ln N! \quad \frac{\partial^2 w}{\partial a^2} = \frac{-N}{a^2} \quad \Delta a = \frac{a}{\sqrt{N}}$$

## (6) – Non-linear least squares minimization



*vertical offsets*



*perpendicular offsets*

mathworld.wolfram.com

Task: Minimize sum of squared residuals (vertical offsets):

$$\chi^2 := \sum_{i=1}^m r_i^2 \quad \text{with} \quad r_i = y_i - f(x, \beta) \quad \text{for } i=1, 2, 3, \dots, m$$

The square is needed to obtain a differentiable function.

## (6) – Non-linear least squares minimization

Function  $f$  depending on  $x$  and the fit parameters  $\beta$ :  $y=f(x,\beta)$  with  $\beta=(\beta_1,\beta_2,\beta_3,\dots,\beta_n)$

Data points  $y_i$ , residuals  $r_i$ :  $r_i=y_i-f(x,\beta)$  for  $i=1,2,3,\dots,m$

Task: Minimize  $\chi^2:=\sum_{i=1}^m r_i^2$  Minimum:  $\frac{\partial \chi^2}{\partial \beta_j}=2\sum_{i=1}^m r_i \frac{\partial r_i}{\partial \beta_j}=0$  ( $j=1,\dots,n$ )

Gradient equations

Iteration:  $\beta_j \approx \beta_j^{k+1} = \beta_j^k + \Delta \beta_j$  with the vector of increments (shift vector)  $\Delta \beta$

First-order Taylor expansion about  $\beta^k$  at each iteration step:

$$\begin{aligned} f(x_i, \beta) &\approx f(x_i, \beta^k) + \sum_j \frac{\partial f(x_i, \beta)}{\partial \beta_j} (\beta_j - \beta_j^k) \\ &= f(x_i, \beta^k) + \sum_j J_{ij} \Delta \beta_j \end{aligned}$$

$$r_i = y_i - f(x_i, \beta^k) - \sum_{k=1}^n J_{ik} \Delta \beta_k \quad \text{with the Jacobian } J_{ij} = -\frac{\partial r_i}{\partial \beta_j}$$

## (6) – Non-linear least squares minimization

Residuals:  $r_i = y_i - f^k(x_i, \beta) - \sum_{k=1}^n J_{ik} \Delta \beta_k$  with the Jacobian  $J_{ij}$

The Jacobian changes during each iteration.

Iteration:  $\beta_j \approx \beta_j^{k+1} = \beta_j^k + \Delta \beta_k$

$$r_i = y_i - f(x_i, \beta) = \underbrace{(y_i - f(x_i, \beta^k))}_{\text{inserted}} + (f(x_i, \beta^k) - f(x_i, \beta)) \approx \Delta y_i - \sum_{s=1}^n J_{is} \Delta \beta_s$$

$\Delta y_i = y_i - f(x_i, \beta^k)$

Inserting the above into the gradient equations:

$$-2 \sum_{i=1}^m J_{ij} \left( \Delta y_i - \sum_{s=1}^n J_{is} \Delta \beta_s \right) = 0$$

Rearrangement to:  $\sum_{i=1}^m \sum_{s=1}^n J_{ij} J_{is} \Delta \beta_s = \sum_{i=1}^m J_{ij} \Delta y_i \quad (j=1, \dots, n)$

In matrix notation:  $(J^T J) \Delta \beta = J^T \Delta y$

## (6) – Non-linear least squares minimization

Generalize to error-weighted fits:

Error-weighted fits, diagonal weight matrix  $\mathbf{W}$

$$\chi^2 = \sum_{i=1}^m W_{ii} r_i^2$$

$$(\mathbf{J}^T \mathbf{W} \mathbf{J}) \Delta \boldsymbol{\beta} = \mathbf{J}^T \mathbf{W} \Delta \mathbf{y}$$

Close to the minimum value:

$$\chi^2 \approx \sum_i W_{ii} \left( y_i - \sum_j J_{ij} \beta_j \right)^2 \quad \text{quadratic function}$$

## (6) – Non-linear least squares minimization

### The error matrix

$$\mathbf{r} = (r_1, \dots, r_m)$$

$$\nabla^2 \chi^2 = \nabla (\nabla \mathbf{r}^2) = \nabla (2 \mathbf{r} \nabla \mathbf{r}) = 2 \nabla \mathbf{r} \nabla \mathbf{r} + 2 \mathbf{r} \nabla^2 \mathbf{r} = 2 \mathbf{J}^T \mathbf{J} + 2 \mathbf{r} \nabla^2 \mathbf{r}$$

chain rule

product rule

Hessian

Inverse of covariance matrix

Algorithms:  
Gradient-descent  
Gauss-Newton  
Levenberg-Marquart  
...

### In python:

```

popt, pcov = curve_fit(lambda x, *p: wrapper_func(x, q, r, fixed_p, len(f0), p), x, y, p0=f0, bounds=(l,u), sigma=dy )

```

↑ function handle     
↑ vector     
↑ variable length     
↑ initial guess

↓ resolution     
↓ fit parameters     
↓ signal     
↓ errors

One standard deviation: `perr = np.sqrt( np.diag( pcov ) )`

[docs.scipy.org](https://docs.scipy.org)

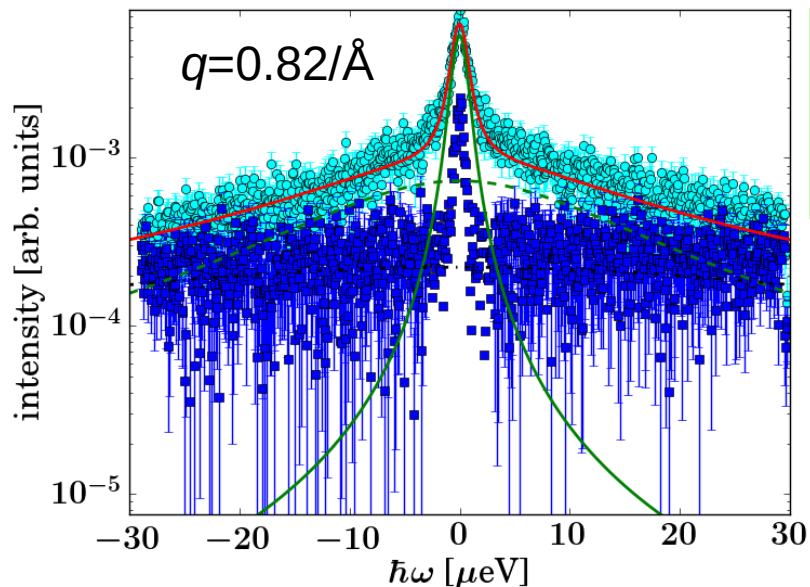
```

def wrapper_func( hw, q, r, fixed_p, N, *args ): # "wrap" a variable number of fit parameters and fixed values into the model
    a = list( args[0][:N] )
    fp = np.asarray( a )
    return model_sqw( hw, fp, q, r, fixed_p ) # call the model function

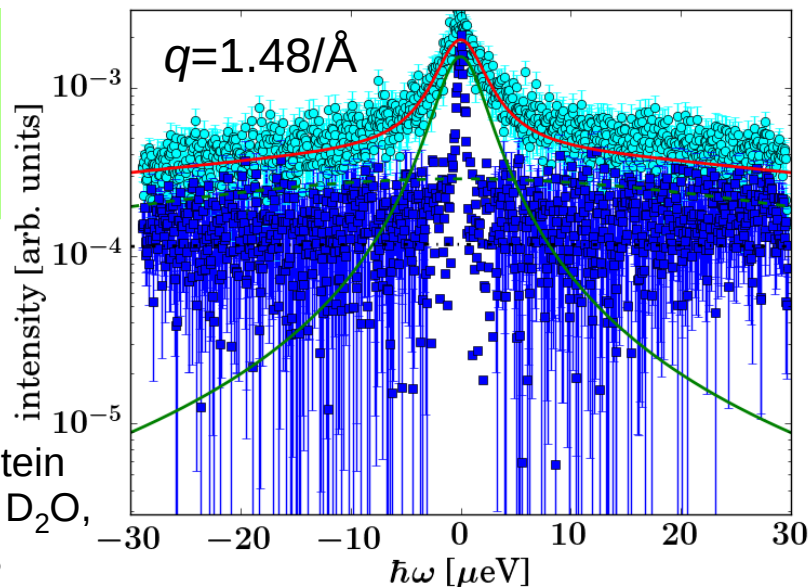
```



## (6) – Non-linear least squares minimization



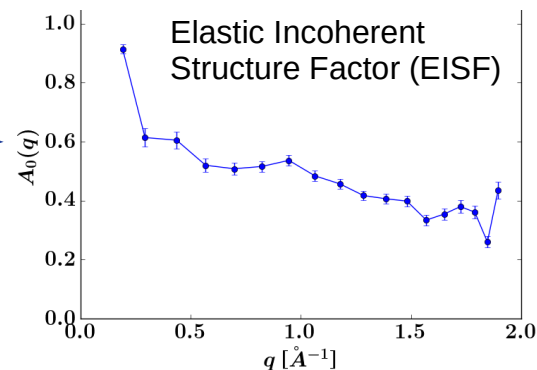
Fitting complex models



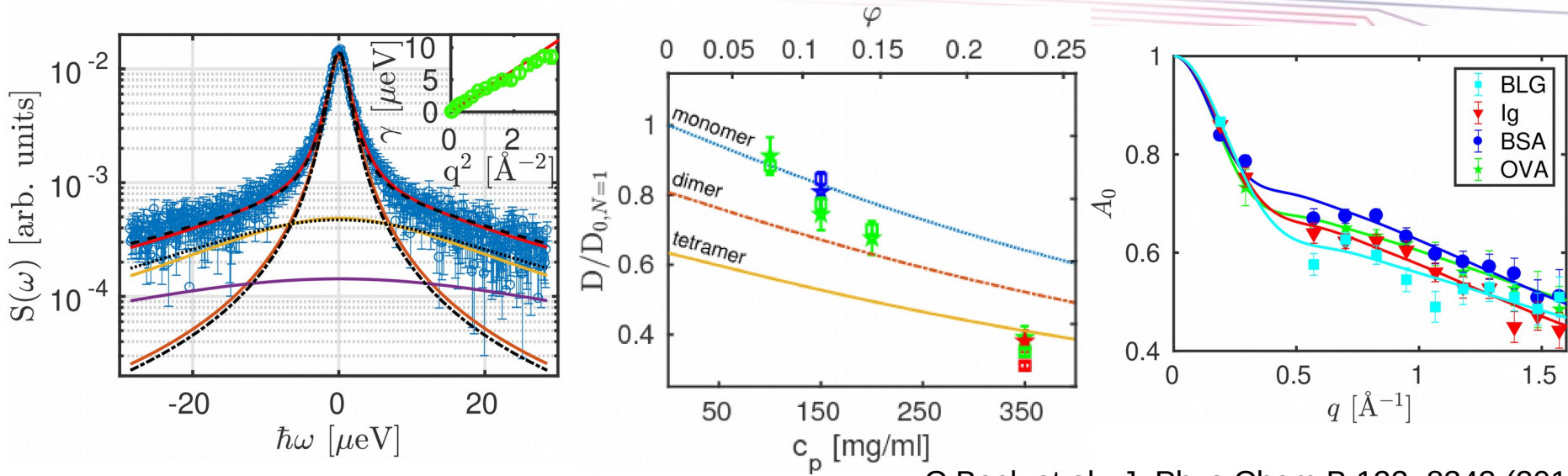
Example protein  
135mg/ml in D<sub>2</sub>O,  
280K, IN16B

$$S(q, \omega) = \mathcal{R} \otimes \{ \beta(q) [ A_0(q) \mathcal{L}(\gamma(q), \omega) + (1 - A_0(q)) \mathcal{L}(\gamma(q) + \Gamma(q), \omega) ] + \beta_{\text{D}_2\text{O}}(q) \mathcal{L}(\gamma_{\text{D}_2\text{O}}(q), \omega) \}$$

$$\Gamma(q) = \frac{D_{\text{int}} q^2}{1 + D_{\text{int}} q^2 \tau} \quad \text{“jump diffusion”}$$



## (6) – Non-linear least squares minimization



C.Beck et al., J. Phys.Chem.B 122, 8343 (2018)

$$S(q, \omega) = \mathcal{R} \otimes \{ \beta(q)[A_0(q)\mathcal{L}(\gamma(q), \omega) + (1 - A_0(q))\mathcal{L}(\gamma(q) + \Gamma(q), \omega)] + \beta_{\text{D}_2\text{O}}(q)\mathcal{L}(\gamma_{\text{D}_2\text{O}}(q), \omega) \}$$

$$\Gamma(q) = \frac{D_{\text{int}}q^2}{1 + D_{\text{int}}q^2\tau} \quad \text{“jump diffusion”}$$

## (6) – Generalization to Maximum Entropy

$P(X, Y) = P(X|Y)P(Y) = P(Y|X)P(X)$  Bayes' theorem,  $X, Y$ : random variables,  
 $P(X, Y)$ : probability of simultaneous incidence of  $X$  and  $Y$ ,  $P(.|.)$ : conditional probability

$$P(X|Y) = P(Y|X) \frac{P(X)}{P(Y)}$$

$$P(f|g^{\text{exp}}, \lambda) = P(g^{\text{exp}}|f, \lambda) \frac{P(f|\lambda)}{P(g^{\text{exp}}|\lambda)}$$

$$P(g^{\text{exp}}|f, \lambda) \propto \exp\left(-\frac{1}{2}\chi^2\right) \text{ for number of data points } N_d \rightarrow \infty$$

$$\chi^2 = \sum_{k=1}^{N_d} \frac{(g_k^{\text{exp}} - g_k^{\text{theo}}(f))^2}{\sigma_k^2}$$

$$P(f|\lambda) = \exp(-\alpha S) \quad \text{with} \quad S(h) = \sum_{i=1}^{N_p} \left( h_i - m_i - h_i \ln \left( \frac{h_i}{m_i} \right) \right) \quad \text{Shannon entropy}$$

Task: Maximize  $\exp\left(\alpha S - \frac{1}{2}\chi^2\right)$  i.e. maximize  $\alpha S - \frac{1}{2}\chi^2$  i.e. minimize  $\frac{1}{2}\chi^2 - \alpha S$

Minimal free energy  $F = U - TS$

## (6) – Generalization to Maximum Entropy

$P(X, Y) = P(X|Y)P(Y) = P(Y|X)P(X)$  Bayes' theorem,  $X, Y$ : random variables,  
 $P(X, Y)$ : probability of simultaneous incidence of  $X$  and  $Y$ ,  $P(.|.)$ : conditional probability

$$P(X|Y) = P(Y|X) \frac{P(X)}{P(Y)}$$

$$P(f|g^{\text{exp}}, \lambda) = P(g^{\text{exp}}|f, \lambda) \frac{P(f|\lambda)}{P(g^{\text{exp}}|\lambda)}$$

Prior probability function

Likelihood function:

$$P(g^{\text{exp}}|f, \lambda) \propto \exp\left(-\frac{1}{2}\chi^2\right) \text{ for number of data points } N_d \rightarrow \infty$$

“Ockham’s razor”  
(William of Ockham, early 14<sup>th</sup> century)  
“Essentia non sunt multiplicanda praeter necessitatem.”

$$\chi^2 = \sum_{k=1}^{N_d} \frac{(g_k^{\text{exp}} - g_k^{\text{theo}}(f))^2}{\sigma_k^2}$$

$h_i$ : reconstructed data  
in a suitable space  
 $m_i$ : measure on that space  
 (“prior knowledge”)

$$P(f|\lambda) = \exp(-\alpha S) \quad \text{with} \quad S(h) = \sum_{i=1}^{N_p} \left( h_i - m_i - h_i \ln \left( \frac{h_i}{m_i} \right) \right) \quad \text{Shannon entropy}$$

Task: Maximize  $\exp\left(\alpha S - \frac{1}{2}\chi^2\right)$  i.e. maximize  $\alpha S - \frac{1}{2}\chi^2$  i.e. minimize  $\frac{1}{2}\chi^2 - \alpha S$

Minimal free energy  $F = U - TS$

## (6) – Generalization to Maximum Entropy

### Example

Task: Determine an even function  $f(x)$  in the intervall  $x \in [0,1]$

Measured data: Fourier coefficients  $g_k = 2 \int_0^1 f(x) \cos(\pi k x) dx$

$$g_k = \frac{2}{N} \sum_{n=0}^{N-1} f_n \cos\left(\frac{\pi k n}{N}\right) \quad f_n = f\left(\frac{n}{N}\right)$$

$f(x) = \frac{g_0}{2} + \sum_{k=1}^{\infty} g_k \cos(\pi k x)$  Fourier reconstruction, requiring an assumption for  $k > 9$

|       |     |        |         |        |        |
|-------|-----|--------|---------|--------|--------|
| k     | 0   | 1      | 2       | 3      | 4      |
| $g_k$ | 200 | -75.31 | -107.03 | 142.26 | -19.51 |

|       |         |       |       |        |       |
|-------|---------|-------|-------|--------|-------|
| k     | 5       | 6     | 7     | 8      | 9     |
| $g_k$ | -101.78 | 92.62 | 13.58 | -86.06 | 53.69 |

Table of measured Fourier coefficients

## (6) – Generalization to Maximum Entropy

$$g_k^{\text{theo}} = \frac{2}{N} \sum_{n=0}^{N-1} f_n \cos\left(\frac{\pi k n}{N}\right)$$

$$\chi^2 = \sum_{k=0}^9 \left( \frac{g_k^{\text{exp}} - g_k^{\text{theo}}}{\sigma_k} \right)^2$$

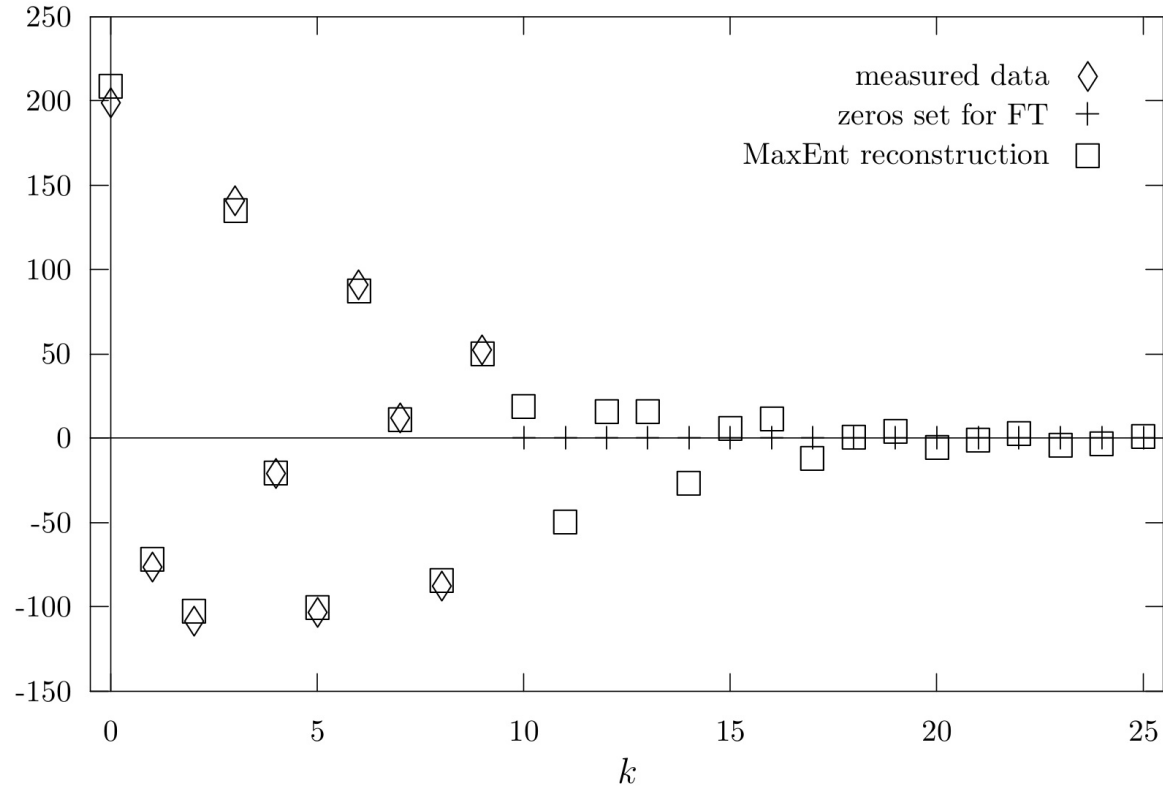
$$S = \sum_{n=0}^{N-1} \left( f_n - m_n - f_n \ln\left(\frac{f_n}{m}\right) \right)$$

$$m_n = m \quad \forall n$$

$$\chi^2 = K, \quad K=10, \quad \text{sets } \alpha$$

$$\sigma_k = 5 \quad \forall k \in \{0, \dots, 9\}$$

$$f_n \geq 0 \quad \forall n \in \{0, \dots, N-1\}$$



## (6) – Generalization to Maximum Entropy

$$g_k^{\text{theo}} = \frac{2}{N} \sum_{n=0}^{N-1} f_n \cos\left(\frac{\pi k n}{N}\right)$$

$$\chi^2 = \sum_{k=0}^9 \left( \frac{g_k^{\text{exp}} - g_k^{\text{theo}}}{\sigma_k} \right)^2$$

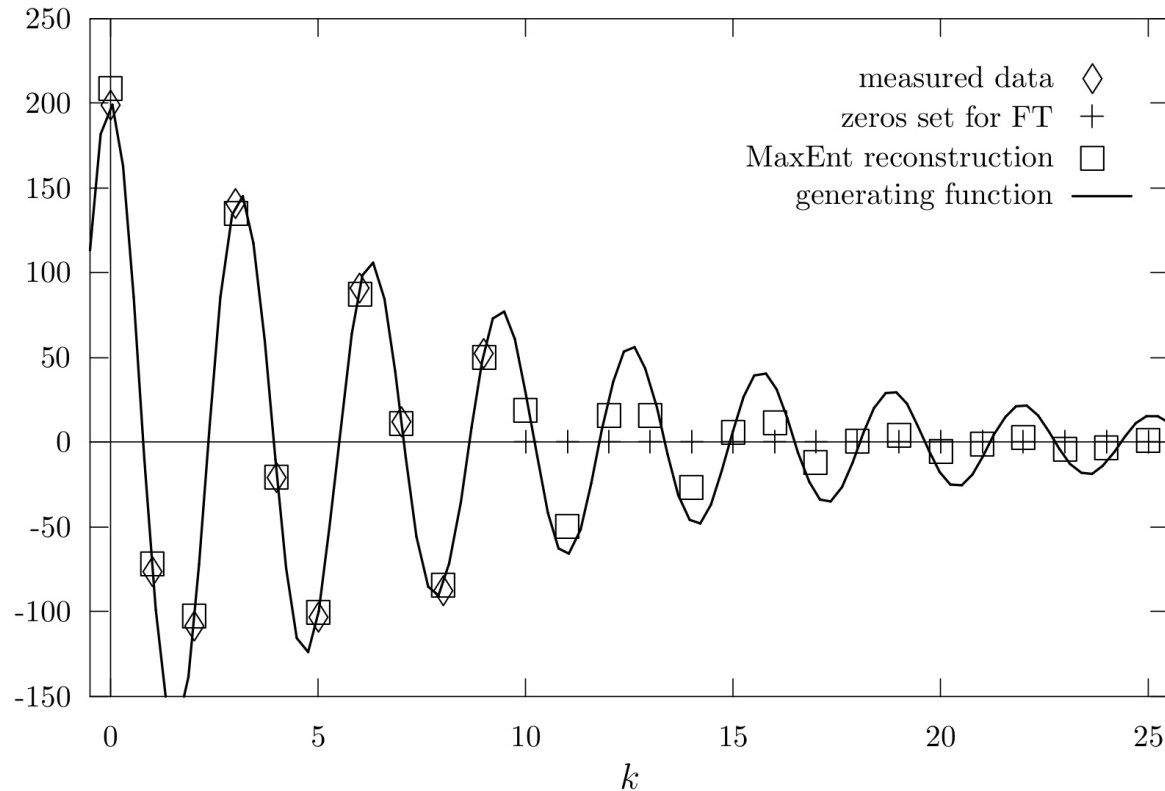
$$S = \sum_{n=0}^{N-1} \left( f_n - m_n - f_n \ln\left(\frac{f_n}{m}\right) \right)$$

$$m_n = m \quad \forall n$$

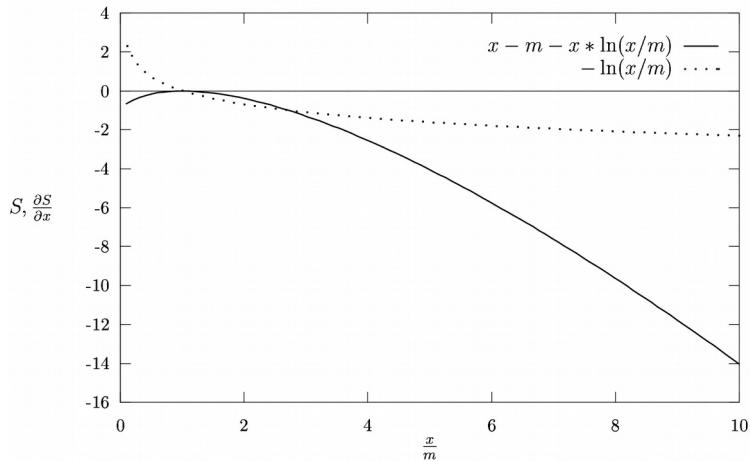
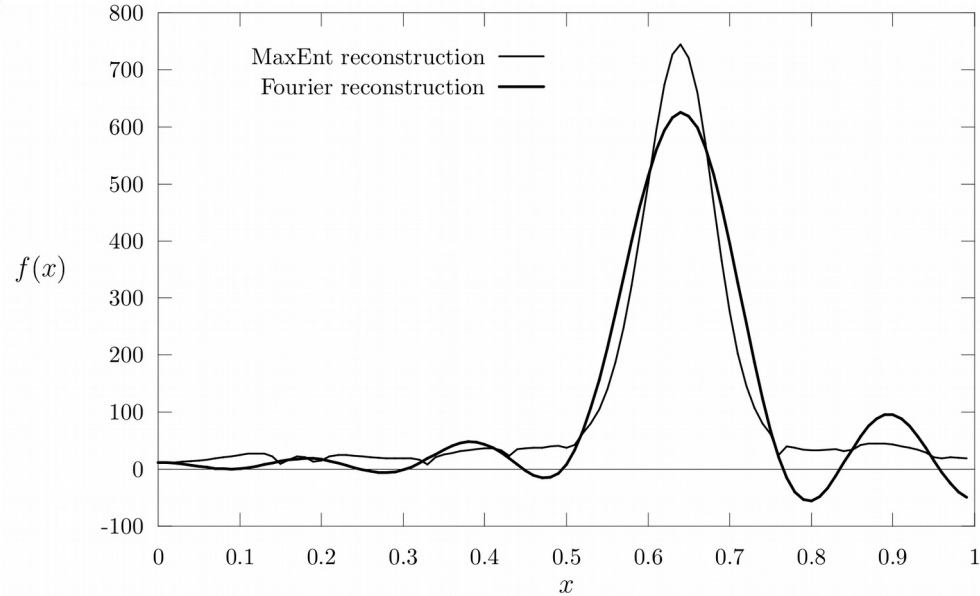
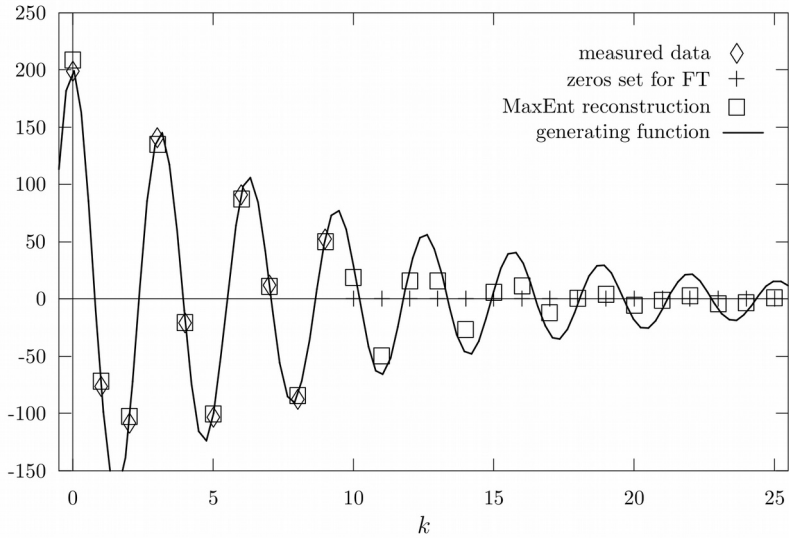
$$\chi^2 = K, \quad K=10, \quad \text{sets } \alpha$$

$$\sigma_k = 5 \quad \forall k \in \{0, \dots, 9\}$$

$$f_n \geq 0 \quad \forall n \in \{0, \dots, N-1\}$$



# (6) – Generalization to Maximum Entropy



$$g_k = 2 \int_0^1 f(x) \cos(\pi k x) dx$$

$$f(x) = \frac{g_0}{2} + \sum_{k=1}^{\infty} g_k \cos(\pi k x)$$



Diffusion equation:

$$\frac{\partial}{\partial t} G(\mathbf{r}, t) = D \nabla^2 G(\mathbf{r}, t) \qquad G(r, t) = \frac{1}{\sqrt{4Dt}} \exp\left(-\frac{r^2}{4Dt}\right)$$

$$\hat{\mathbf{D}}^\alpha G(\mathbf{r}, t) = D_\alpha \nabla^2 G(\mathbf{r}, t) \quad \leftarrow \text{Generalized “fractional” diffusion equation}$$

Integro-differential operator:

$$\hat{\mathbf{D}}^\alpha [\Phi](t) = \frac{1}{\Gamma(1-\alpha)} \int_0^t \frac{\Phi'(\xi)}{(t-\xi)^\alpha} d\xi$$

Spatial Fourier transform:

$$\frac{\partial G(q, t)}{\partial t} = -Dq^2 G(q, t) \qquad G(q, t) = \exp(-Dq^2 t)$$

Diffusion equation:

$$\frac{\partial}{\partial t} G(\mathbf{r}, t) = D \nabla^2 G(\mathbf{r}, t) \qquad G(r, t) = \frac{1}{\sqrt{4Dt}} \exp\left(-\frac{r^2}{4Dt}\right)$$

$$\hat{\mathbf{D}}^\alpha G(\mathbf{r}, t) = D_\alpha \nabla^2 G(\mathbf{r}, t)$$

Integro-differential operator:

$$\hat{\mathbf{D}}^\alpha [\Phi](t) = \frac{1}{\Gamma(1-\alpha)} \int_0^t \frac{\Phi'(\xi)}{(t-\xi)^\alpha} d\xi \quad \leftarrow \text{(by Cauchy-formula for repeated integration)}$$

Spatial Fourier transform:

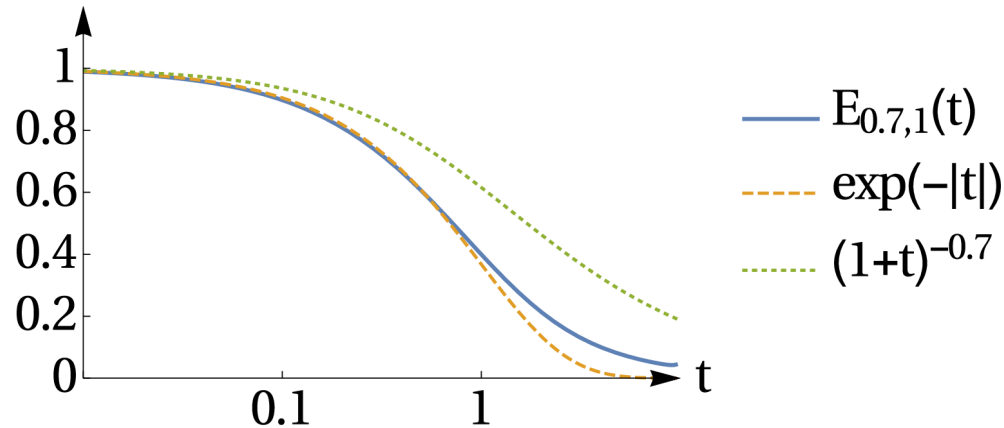
$$\hat{\mathbf{D}}^\alpha G(q, t) = -D_\alpha q^2 G(q, t) \qquad G(q, t) = E_{\alpha,1}(-D_\alpha q^2 t^\alpha)$$

## (7) – QENS and (generalized fractional) diffusion

The Mittag-Leffler function interpolates between the stretched exponential and power law:

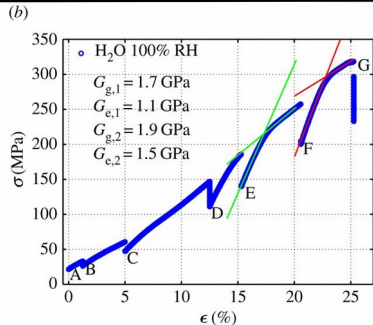
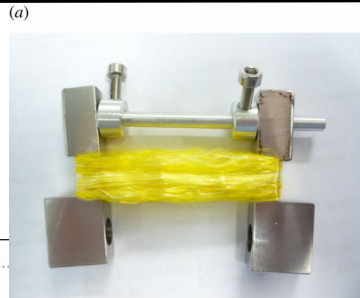
$$E_{\alpha,\beta}(z) = \sum_{n=0}^{\infty} \frac{z^n}{\Gamma(\alpha n + \beta)} \quad \text{with} \quad \Gamma(n) = (n-1)! \quad \text{and} \quad \Gamma(z) = \int_0^{\infty} x^{z-1} e^{-x} dx$$

$$\exp(z) = \sum_{n=0}^{\infty} \frac{z^n}{n!} = E_{1,1}(z), \quad \text{stretched exponential} \quad \exp(-|z|^\beta)$$

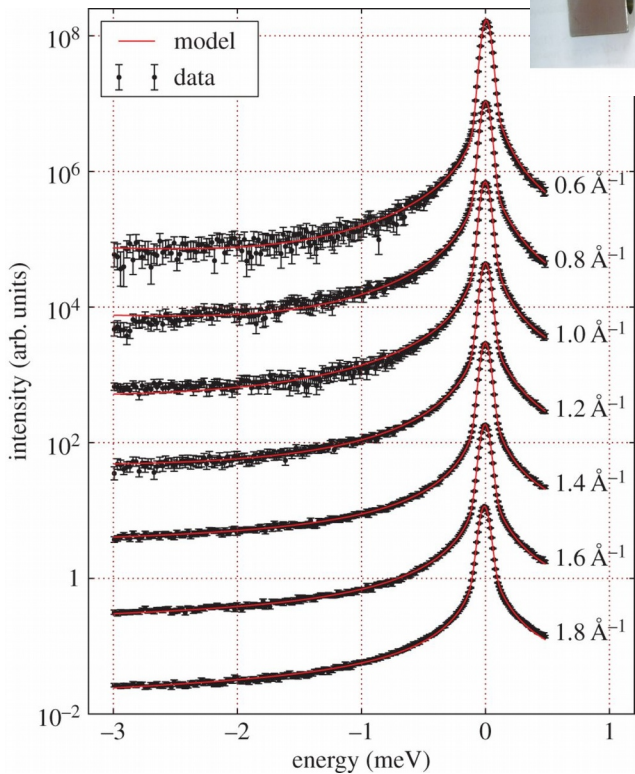


# (7) – QENS and (generalized fractional) diffusion

## Time-of-flight spectroscopy on spider silk



Tensile strain

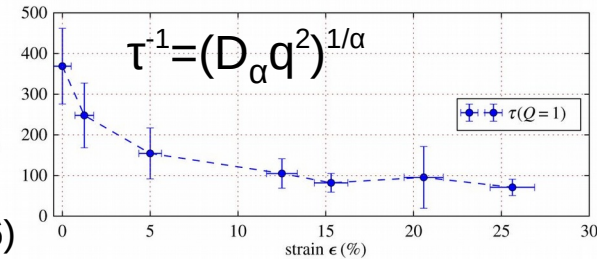
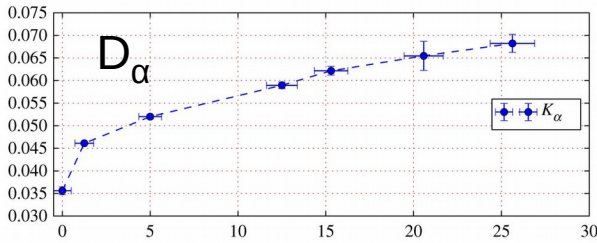
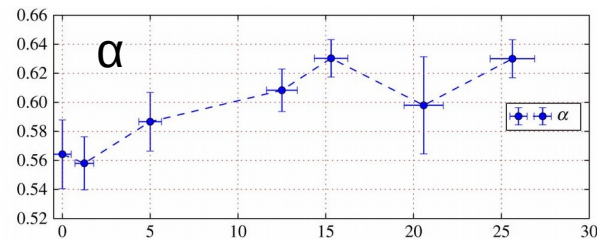


Global fit of all  $q$  at once

Increasing loss of memory

Faster mobility with strain

Increasing spectral linewidth



I.Krasnov et al.,  
 J.R.Soc.Interface 13, 20160506 (2016)

- A few concepts for data reduction and analysis
- Standard reductions / corrections
- Non-linear least squares fitting
- No universal solution for all problems
- Tools to build own analysis, Mantid, python, lamp, Dave, ....

Thank you for listening !