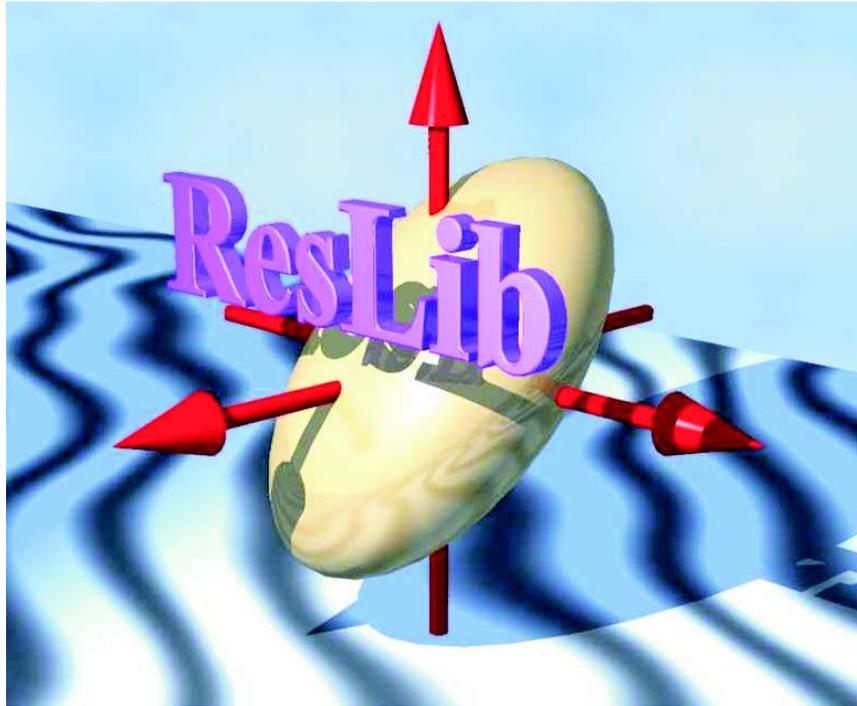


# ResLib 3.1



## 3-axis resolution library for MatLab

A. Zheludev

*Solid State Division, Oak Ridge national Laboratory, Oak Ridge, TN 37831-6393, USA.*

(Dated: October 15, 2001)

### Abstract

Features of the latest *ResLib* release include: 3-axis resolution calculations in the **Cooper-Nathans** and **Popovici** approximations with support for horizontally focusing analyzers, anisotropic sample mosaic, normalization to monitor counts, analyzer reflectivity corrections and data smoothing; numerical convolution of user-supplied cross section functions with calculated experimental resolution, including cross sections written in the **single mode or Lorentzian approximations**; fitting of parameterized model cross sections to experimental data; graphical representation of resolution ellipsoids in 2 and 3 dimensions; and crystallographic utility functions. The present manual provides a detailed description of the functions included in the *ResLib* library and lists **all the essential formulas** used in the calculations.

## Contents

<b>Introduction</b>	4
<b>Formulas for Resolution calculation</b>	5
Cooper-Nathans resolution matrix	5
Bopovici resolution matrix	6
Normalization factors	7
Normalization to source flux	7
Normalization to monitor counts	8
Analyzer reflectivity	9
Focusing analyzer	9
Sample mosaic	10
Coordinate systems	10
Smoothed data	10
<b>HsLib functions for resolution calculation</b>	12
ResMat.m	12
ResMatS.m	15
MakeExp.m	17
<b>Crystallography utility functions</b>	18
Star.m	18
Scalar.m	19
Modvec.m	19
StandardSystem.m	19
<b>Convolution with resolution function: general 4-D convolution</b>	21
ConvRes.m	22
<b>Convolution with resolution function: Single Mode Approximation and Voigt profiles</b>	25
ConvResSMA.m	26
<b>Least-squares fitting of convoluted cross sections</b>	28

AitConv.m	28
BitConvSMA.m	30
CitConvBoth.m	30
<b>Graphic representations of resolution ellipsoids</b>	<b>31</b>
ResPlot.m	31
ResPlot3D.m	32
<b>Downloads, packaging, installation and feedback</b>	<b>34</b>
File download	34
Backaging	34
Installation	36
Feedback	36
<b>Credits</b>	<b>36</b>
<b>Disclaimer</b>	<b>36</b>
<b>References</b>	<b>37</b>

## I. INTRODUCTION

Perhaps the most important aspect of analyzing inelastic neutron scattering data measured using a 3-axis spectrometer is properly taking into account the experimental resolution. Even though one is fairly confident that the Gaussian approximation to the resolution function is accurate, thanks to the Central Limit Theorem, the problem is still quite non-trivial. In general, to properly analyze experimental scans using a parameterized model cross section  $S(\mathbf{Q}, \hbar\omega, \mathbf{p})$ , one needs to 1) be able to calculate the resolution function at each data point, given the spectrometer configuration and sample parameters, 2) numerically convolute the theoretical cross section with this resolution function; and 3) fit the convoluted cross section to the data. The *ResLib* library is designed to accomplish these tasks.

## II. FORMULAS FOR RESOLUTION CALCULATION

*ResLib* is based on the Gaussian approximation for the resolution function. The measured inelastic neutron scattering intensity is related to the dynamic structure factor through:

$$\frac{d\sigma}{d\Omega dE'} \approx R_0(\mathbf{Q}_0) \int \int \int \int d^4\mathcal{Q} S(\mathcal{Q}) \exp \left[ -\frac{1}{2} (\mathcal{Q}^i - \mathcal{Q}_0^i) (\mathcal{Q}^j - \mathcal{Q}_0^j) M_{ij}(\mathbf{Q}_0) \right]. \quad (1)$$

Here  $\mathcal{Q} \equiv (Q_x, Q_y, \hbar\omega, Q_z)$  represents a 4-D vector in energy-momentum space,  $\mathbf{Q}_0 = \mathbf{k}_i - \mathbf{k}_f$  is the momentum transfer to the sample, and  $\hbar\omega_0 = E_i - E_f$  is the energy transfer. In *ResLib* wave vectors are measured in  $\text{\AA}^{-1}$ , and energy in meV. The coordinate system for components of  $\mathbf{Q}$  and  $M$  is defined by the scattering vector and the scattering plane.  $x$  is chosen along  $\mathbf{Q}_0$ ,  $z$  is perpendicular to the scattering plane (vertical) and  $y$  completes the orthogonal right-handed basis set. Note that in Eq. ?? the usual  $k_i/k_f$  scaling factor is included into  $R_0$ .

### A. Cooper-Nathans resolution matrix

In the Cooper-Nathans approximation [? ? ] it is assumed that the beam divergences on each arm of the 3-axis spectrometer are determined by the Soller collimators and the mosaic spread of monochromator and analyzer crystals. The most elegant formulas for this approximation are given in [? ]. The Soller collimators are described by a diagonal  $8 \times 8$  matrix with non-zero elements  $G_{1,1} = 8 \ln(2)\alpha_1^{-2}$ ,  $G_{2,2} = 8 \ln(2)\alpha_2^{-2}$ ,  $G_{3,3} = 8 \ln(2)\beta_1^{-2}$ ,  $G_{4,4} = 8 \ln(2)\beta_2^{-2}$ ,  $G_{5,5} = 8 \ln(2)\alpha_3^{-2}$ ,  $G_{6,6} = 8 \ln(2)\alpha_4^{-2}$ ,  $G_{7,7} = 8 \ln(2)\beta_3^{-2}$  and  $G_{8,8} = 8 \ln(2)\beta_4^{-2}$ , where  $\alpha_i$  and  $\beta_i$  are the FWHM horizontal and vertical beam divergences in the collimators, starting from the in-pile collimation.  $\eta$  and  $\eta'$ , the horizontal and vertical FWHM mosaic spreads for the monochromator and analyzer crystals, are collected into a diagonal  $4 \times 4$  matrix with non-zero elements  $F_{1,1} = 8 \ln(2)\eta_M^{-2}$ ,  $F_{2,2} = 8 \ln(2)\eta_M'^{-2}$ ,  $F_{3,3} = 8 \ln(2)\eta_A^{-2}$ , and  $F_{4,4} = 8 \ln(2)\eta_A'^{-2}$ . In addition, the following matrixes are defined.  $A$  is a  $6 \times 8$  matrix with non-zero elements  $A_{1,1} = -A_{1,2} = k_i \cot(\theta_M)/2$ ,  $A_{4,5} = -A_{4,6} = k_f \cot(\theta_A)/2$ ,  $A_{2,2} = A_{3,4} = k_i$  and  $A_{5,5} = A_{6,7} = k_f$ .  $C$  is a  $6 \times 8$  matrix with non-zero elements  $C_{1,1} = C_{1,2} = C_{3,5} = C_{3,6} = 1/2$ ,  $C_{2,3} = -C_{2,4} = 1/(2 \sin \theta_M)$  and  $C_{4,7} = -C_{4,7} = 1/(2 \sin \theta_A)$ . Note that in [? ] there is a typo in the definition of matrix  $C$ . Finally,  $B$  is defined as a  $4 \times 6$  matrix with the following non-zero elements:  $B_{1,1} = B_{2,2} \cos \phi$ ,  $B_{1,2} = -B_{2,1} = \sin \phi$ ,  $B_{1,4} = B_{2,5} = -\cos(\phi - 2\theta_S)$ ,  $B_{1,5} = -B_{2,4} = -\sin(\phi - 2\theta_S)$ ,

$B_{3,3} = -B_{3,6} = 1$ ,  $B_{4,1} = \frac{\hbar^2}{m}k_i$  and  $B_{4,4} = -\frac{\hbar^2}{m}k_f$ . In these formulas  $2\theta_M$ ,  $2\theta_S$  and  $2\theta_A$ , are the scattering angles in monochromator, sample and analyzer, respectively and  $m$  is the neutron mass. The scattering angles are calculated as:

$$\begin{aligned}\theta_M &= -\arcsin(\tau_M/(2k_i))\epsilon_M, \\ \theta_A &= -\arcsin(\tau_A/(2k_f))\epsilon_A, \\ 2\theta_S &= \arccos[(k_i^2 + k_f^2 - Q^2)/(2k_i k_f)],\end{aligned}\tag{2}$$

where  $\epsilon_M = 1$  if the sample scattering direction is opposite to the monochromator scattering directions and  $-1$  otherwise,  $\epsilon_A = 1$  if the sample scattering direction is opposite to the analyzer scattering directions and  $-1$  otherwise, and  $\tau_M = 2\pi/d_M$  and  $\tau_A = 2\pi/d_A$  are the monochromator and analyzer scattering vectors, respectively. The sample rotation angle is given by:

$$\phi = \text{ATAN2}(-k_f \sin 2\theta_S, k_i - k_f \cos 2\theta_S).\tag{3}$$

These matrixes defined, the resolution matrix is given by:

$$\tilde{M}^{-1} = BA(G + C^T FC)^{-1}A^T B^T.\tag{4}$$

The resolution matrix  $M$  is obtained from  $\tilde{M}$  by switching the 3rd and 4th rows and columns.

## B. Popovici resolution matrix

The beam divergences are influenced not only by Soller collimators, but also by the shapes and dimensions of the source, monochromator, sample, analyzer and detector [? ]. These shapes are described by the  $13 \times 13$  covariance matrix  $S^{-1}$ . The matrix is cell-diagonal and has the form:  $S^{-1} = \{\langle y_0^2 \rangle, \langle z_0^2 \rangle, S_M^{-1}, S_S^{-1}, S_A^{-1} \langle y_4^2 \rangle, \langle z_4^2 \rangle\}$ .  $\langle y_0^2 \rangle$  and  $\langle z_0^2 \rangle$  describe the spatial distribution of the source density perpendicular to the first spectrometer arm in the horizontal and vertical directions, respectively. Similarly,  $\langle y_4^2 \rangle$  and  $\langle z_4^2 \rangle$  describe the spatial distribution of the detector.  $S_M^{-1}$ ,  $S_S^{-1}$  and  $S_A^{-1}$  are the spatial distributions of the monochromator, sample and analyzer densities, respectively. For example,

$$S_S^{-1} = \begin{bmatrix} \langle x^2 \rangle & \langle xy \rangle & \langle xz \rangle \\ \langle yx \rangle & \langle y^2 \rangle & \langle yz \rangle \\ \langle zx \rangle & \langle zy \rangle & \langle z^2 \rangle \end{bmatrix},\tag{5}$$

where  $x$  bisects the incident and scattered beams on the sample,  $z$  is the vertical axis and  $y$  completes an orthogonal coordinate system.

Additional matrices are defined and contain the distances  $L_0$ ,  $L_1$ ,  $L_2$  and  $L_3$  from the source to the monochromator, from the monochromator to the sample, from the sample to the analyzer and from the analyzer to the detector, respectively. Additional parameters are the horizontal and vertical curvature radii  $\rho$  and  $\rho'$ , for the monochromator and analyzer crystals.  $T$  is a  $4 \times 13$  matrix with the following non-zero elements:  $T_{1,1} = -1/(2L_0)$ ,  $T_{1,3} = \cos \theta_M(1/L_1 - 1/L_0)/2$ ,  $T_{1,4} = \sin \theta_M(1/L_0 + 1/L_1 - 2/(\rho_M \sin \theta_M))/2$ ,  $T_{1,6} = \sin \theta_S/(2L_1)$ ,  $T_{1,7} = \cos \theta_S/(2L_1)$ ,  $T_{2,2} = -1/(2L_0 \sin \theta_M)$ ,  $T_{2,5} = (1/L_0 + 1/L_1 - 2 \sin \theta_M/\rho'_M)/(2 \sin \theta_M)$ ,  $T_{2,8} = -1/(2L_1 \sin \theta_M)$ ,  $T_{3,6} = \sin \theta_S/(2L_2)$ ,  $T_{3,7} = -\cos \theta_S/(2L_2)$ ,  $T_{3,9} = \cos \theta_A(1/L_3 - 1/L_2)/2$ ,  $T_{3,10} = \sin \theta_A(1/L_2 + 1/L_3 - 2/(\rho_A \sin \theta_A))/2$ ,  $T_{3,12} = 1/(2L_3)$ ,  $T_{4,8} = -1/(2L_2 \sin \theta_A)$ ,  $T_{4,11} = (1/L_2 + 1/L_3 - 2 \sin \theta_A/\rho'_A)/(2 \sin \theta_A)$  and  $T_{4,13} = -1/(2L_3 \sin \theta_A)$ .  $D$  is defined as an  $8 \times 3$  matrix with non-zero elements  $D_{1,1} = -1/L_0$ ,  $D_{1,3} = -\cos \theta_M/L_0$ ,  $D_{1,4} = \sin \theta_M/L_0$ ,  $D_{3,2} = D_{1,1}$ ,  $D_{3,5} = -D_{1,1}$ ,  $D_{2,3} = \cos \theta_M/L_1$ ,  $D_{2,4} = \sin \theta_M/L_1$ ,  $D_{2,6} = \sin \theta_S/L_1$ ,  $D_{2,7} = \cos \theta_S/L_1$ ,  $D_{4,5} = -1/L_1$ ,  $D_{4,8} = -D_{4,5}$ ,  $D_{5,6} = \sin \theta_S/L_2$ ,  $D_{5,7} = -\cos \theta_S/L_2$ ,  $D_{5,9} = -\cos \theta_A/L_2$ ,  $D_{5,10} = \sin \theta_A/L_2$ ,  $D_{7,8} = -1/L_2$ ,  $D_{7,11} = -D_{7,8}$ ,  $D_{6,9} = \cos \theta_A/L_3$ ,  $D_{6,10} = \sin \theta_A/L_3$ ,  $D_{6,12} = 1/L_3$ ,  $D_{8,11} = -D_{6,12}$  and  $D_{8,13} = D_{6,12}$ . Note that [?] contains numerous typos in the definition of matrix  $T$  and  $D$ . The resolution matrix  $M$  in the Popovici approximation is given by:

$$\tilde{M}^{-1} = BA \left[ \{D(S + T^T FT)^{-1} D^T\}^{-1} + G \right]^{-1} A^T B^T \quad (6)$$

### C. Normalization factors

#### 1. Normalization to source flux

The normalization of the measured intensity by the intensity of the neutron source in the Cooper-Nathans approximation is given by:

$$R_0 = \frac{k_f \pi^2 (k_i^3 \cot \theta_M) (k_f^3 \cot \theta_A)}{k_i (16 \sin \theta_M \sin \theta_A)} \sqrt{\frac{\det(F)}{\det(G + C^T FC)}} \times \left[ (2\pi)^2 \sqrt{\det(M)} \right]^{-1}. \quad (7)$$

The factor  $\left[ (2\pi)^2 \sqrt{\det(M)} \right]^{-1}$  is included to comply with the resolution function normalization convention used in Eq. ?? above and [?]. As mentioned above, the  $k_f/k_i$  factor is

included in the normalization coefficient. The expression for  $R_0$  in the Popovici approximation in [?] does not include the effect of Soller collimators. If these are correctly taken into account,  $R_0$  becomes:

$$R_0 = \frac{k_f \pi^2 (k_i^3 \cot \theta_M)(k_f^3 \cot \theta_A)}{k_i (16 \sin \theta_M \sin \theta_A)} \sqrt{\frac{\det(F)}{\det([D(S + T^T F T)^{-1} D^T]^{-1} + G)}} \times \left[ (2\pi)^2 \sqrt{\det(M)} \right]^{-1}. \quad (8)$$

## 2. Normalization to monitor counts

The monitor is typically positioned between the 2nd Soller collimator and the sample. In the Cooper-Nathans approximation the flux on the monitor is straightforward to calculate, since the resolution function is simply a product of transmission functions for each spectrometer component. Including the  $1/k_i$  monitor sensitivity correction, the count rate on the monitor is proportional to:

$$R_{\text{mon}} = \frac{1}{k_i} \frac{\pi (k_i^3 \cot \theta_M)}{(2 \sin \theta_M)} \sqrt{\frac{\det(f)}{\det(g + c^T f c)}}. \quad (9)$$

In this formula the matrices  $g$ ,  $f$  and  $c$  are primary minors of matrices  $G$ ,  $F$  and  $C$ , of sizes  $4 \times 4$ ,  $2 \times 2$  and  $2 \times 4$ , respectively.

In the Popovici approximation the monitor rate will depend on the dimensions and position of the monitor on the second spectrometer arm. To estimate the count rate, one has to define the following matrices. The matrix  $s$  is similar to  $S$  and is defined as  $s^{-1} = \{\langle y_0^2 \rangle, \langle z_0^2 \rangle, S_M^{-1}, \langle y_{\text{mon}}^2 \rangle, \langle z_{\text{mon}}^2 \rangle\}$ . Here  $\langle y_{\text{mon}}^2 \rangle$  and  $\langle z_{\text{mon}}^2 \rangle$  describe the spatial distribution of the monitor perpendicular to the second spectrometer arm, in the horizontal and vertical directions, respectively.  $L_{\text{mon}}$  being the distance from the monochromator to the monitor, the transformation matrices  $t$  and  $d$  are of dimension  $4 \times 7$  and  $2 \times 7$ , respectively, and have the following non-zero elements:  $t_{1,1} = -1/(2L_0)$ ,  $t_{1,3} = \cos(\theta_m)(1/L_{\text{mon}} - 1/L_0)/2$ ,  $t_{1,4} = \sin(\theta_m)(1/L_0 + 1/L_{\text{mon}} - 2/(\rho_M \sin(\theta_m)))/2$ ,  $t_{1,7} = 1/(2L_{\text{mon}})$ ,  $t_{2,2} = -1/(2L_0 \sin(\theta_m))$ ,  $t_{2,5} = (1/L_0 + 1/L_{\text{mon}} - 2 \sin(\theta_m)/\rho'_M)/(2 \sin(\theta_m))$ ,  $d_{1,1} = -1/L_0$ ,  $d_{1,3} = -\cos(\theta_m)/L_0$ ,  $d_{1,4} = \sin(\theta_m)/L_0$ ,  $d_{3,2} = d_{1,1}$ ,  $d_{3,5} = -d_{1,1}$ ,  $d_{2,3} = \cos(\theta_m)/L_{\text{mon}}$ ,  $d_{2,4} = \sin(\theta_m)/L_{\text{mon}}$ ,  $d_{2,6} = 0$ ,  $d_{2,7} = 1/L_{\text{mon}}$ , and  $d_{4,5} = -1/L_{\text{mon}}$ . The monitor rate is then proportional to:

$$R_{\text{mon}} = \frac{1}{k_i} \frac{\pi (k_i^3 \cot \theta_M)}{(2 \sin \theta_M)} \sqrt{\frac{\det(f)}{\det([d(s + t^T f t)^{-1} d^T]^{-1} + g)}}. \quad (10)$$

In either the Popovici or Cooper-Nathans case, to normalize the measured intensity to monitor counts, the factor  $R_0$  in Eq. ?? is to be modified:  $R_0 \rightarrow R_0/R_{\text{mon}}$ .

### 3. Analyzer reflectivity

The prefactor  $R_0$  can include the effect of analyzer reflectivity. The effective reflectivity in the Gaussian model differs from the actual reflectivity of the analyzer crystal. To compensate for this effect,  $R_0$  is corrected so that the integral reflectivity of the effective Gaussian matches the integral reflectivity of an ideal crystal. The reflectivity curve of an ideal crystal can be taken from [? ]. In particular, for a PG(002) analyzer, at  $E_f < 5.5$  meV, where no parasitic ( $hkl$ ) reflections interfere with the reflected (002) peak [? ], this approximation is expected to be valid within a few percent [? ]. At higher final neutron energies this approach can not be considered valid, as the Shapiro-Chesser effects become important [? ]. In *ResLib* the optional reflectivity correction is done following [? ].

### D. Focusing analyzer

*ResLib* provides a means of calculating the Cooper-Nathans resolution function in the case of a multi-blade horizontally focusing analyzer. The central analyzer blade is positioned to reflect at nominal  $\mathbf{k}_f$ . The remaining blades are aligned to reflect at the same  $|k_f|$  as the central blade. Energy resolution is actually improved, because the narrow blades provide additional collimation. Wave vector resolution is of course broadened. The effect can be taken into account through an approximation proposed by C. L. Broholm. Eq. ?? for the resolution matrix is replaced with:

$$\begin{aligned} \tilde{M}^{-1} &= BP^{-1}B^T, & (11) \\ P_{i,j} &= \left[ \left( A(G + C^TFC)^{-1} * A^T \right)^{-1} \right]_{i,j} \text{ for } i, j \neq 4, 5, \\ P_{5,5} &= 12 / (k_f \alpha_f)^2, \\ P_{4,4} &= 12 [\tan \theta_A / (\eta_A k_f)]^2, \\ P_{4,5} &= (P^{-1})_{5,4} = 0, \end{aligned}$$

where  $\alpha_f$  is the angular size of the analyzer as seen from the sample. When calculating the prefactor  $R_0$ , the FWHM horizontal beam divergence in the 3rd Soller collimator  $\alpha_3$  in the

definition of matrix  $G$  is to be replaced by  $\frac{\sqrt{8 \ln 2}}{\sqrt{12}} \alpha_f$ .

### E. Sample mosaic

The effect of sample mosaic can be included in the resolution calculation following [? ]. The matrix  $M$  and the prefactor  $R_0$  are modified as follows:

$$R_0 \rightarrow R_0 \left[ 1 + \frac{1}{8 \ln 2} (|Q_0| \eta_S)^2 M_{4,4} \right]^{-1/2} \left[ 1 + \frac{1}{8 \ln 2} (|Q_0| \eta'_S)^2 M_{2,2} \right]^{-1/2}, \quad (12)$$

$$(M^{-1})_{2,2} \rightarrow (M^{-1})_{2,2} + (|Q_0| \eta_S)^2 / (8 \ln 2), \quad (13)$$

$$(M^{-1})_{4,4} \rightarrow (M^{-1})_{4,4} + (|Q_0| \eta'_S)^2 / (8 \ln 2). \quad (14)$$

Here  $\eta_S$  and  $\eta'_S$  are the FWHM of horizontal and vertical sample mosaic spreads, respectively.

### F. Coordinate systems

All the formulas above describe the resolution matrix in the coordinate system defined by the scattering vector and the scattering plane.  $x$  is chosen along  $\mathbf{Q}_0$ ,  $z$  is vertical and  $y$  completes the orthogonal right-handed basis set. This coordinate system is referred to as the  $Q$ -system in the text below. The *ResLib* function used to calculate resolution parameters in this coordinate system is `ResMat.m`.

*ResLib* also provides a means to calculate the resolution matrix in a coordinate system ( $S$ -system) coupled to the sample. The  $S$ -system is defined by two  $(h, k, l)$  reciprocal-space vectors in the sample. The axis  $x$  is chosen along the 1st orienting vector,  $y$  is normal to  $x$  in the scattering plane and is in the direction of the second vector, and  $z$  completes the orthogonal right-handed basis set. The *ResLib* function `ResMatS.m` performs all the calculations in the  $S$ -system.

### G. Smoothed data

In certain cases it is useful to bin or smooth collected data before proceeding with the data analysis. This is particularly true for very large data sets, such as those collected using area-sensitive detectors. Smoothing consists of folding the measured intensity with a custom smoothing kernel, such as a  $\Pi$ -shaped function of a Gaussian. Of course, the effective

resolution of the measurement is modified by this procedure and is equal to a convolution of the smoothing kernel and the actual spectrometer resolution. *ResLib* can handle the special case of a Gaussian resolution kernel at the level of `ResMatS.m`. The resolution matrix is modified by the  $4 \times 4$  smoothing matrix  $\sigma$  as follows:

$$\Sigma_{1,1} = 8 \ln 2 / \sigma_x^2,$$

$$\Sigma_{2,2} = 8 \ln 2 / \sigma_y^2,$$

$$\Sigma_{3,3} = 8 \ln 2 / \sigma_E^2,$$

$$\Sigma_{4,4} = 8 \ln 2 / \sigma_z^2,$$

$$M^{-1} \rightarrow \Sigma^{-1} + M^{-1}, \tag{15}$$

$$R_0 \rightarrow \frac{\sqrt{\det(\Sigma^{-1} + M^{-1})}}{\sqrt{\det(M^{-1})}} R_0. \tag{16}$$

Here  $\sigma_x, \sigma_y$  and  $\sigma_z$ , are the smoothing Gaussian FWHM along the three components of the scattering vector and  $\sigma_E$  is the smoothing Gaussian FWHM along the energy axis.

### III. *RESLIB* FUNCTIONS FOR RESOLUTION CALCULATION

#### A. `ResMat.m`

This function calculates the resolution matrix  $M$  and the prefactor  $R_0$  in the  $Q$  coordinate system (see definition above).

```
function [R0,M]=ResMat(Q,W,EXP)
```

INPUT ARGUMENTS:

- $Q$  is the wave vector transfer in  $\text{\AA}^{-1}$ .
- $W$  is the energy transfer in meV.
- `EXP` is a structure that contains all the information on the experimental setup and the required method of resolution calculation. See the detailed description below.

OUTPUT ARGUMENTS:

- `R0` is the normalization prefactor as defined above.
- `M` is a  $4 \times 4$  resolution matrix as defined above.

DEPENDENCIES: `ResMat.m` is a stand-alone function that does not rely on any other components in *ResLib*.

`ResMat.m` can be used to simultaneously calculate resolution matrices for several scattering vectors, energies, and experimental setups. The input arguments  $Q$ ,  $W$  and `EXP` can be vectors of the same length  $N$ , in which case `R0` will be returned as a vector of length  $N$  and `M` will be a  $4 \times 4 \times N$  array. If some of the input arguments are scalars, they will be automatically replicated into vectors of appropriate length.

The fields of the structure `EXP` are defined as follows:

- `EXP.method` selects the computation method. If `EXP.method=0` or the field is left undefined, a Cooper-Nathans calculation is performed. For a Popovici calculation set `EXP.method=1`.
- `EXP.moncor` selects the type of normalization used to calculate  $R_0$ . If `EXP.moncor=1` or the field is left undefined,  $R_0$  is calculated in normalization to monitor counts (Section ??).  $1/k_i$  monitor efficiency correction is included automatically. To normalize  $R_0$  to source flux (Section ??), use `EXP.moncor=0`.

- `EXP.mono` is a structure that describes the monochromator:
  - `EXP.mono.tau` is the monochromator reciprocal lattice vector in  $\text{\AA}^{-1}$ . Instead of a numerical input one can use one of the following keyword strings: `'PG(002)'`, `'PG(004)'`, `'Ge(111)'`, `'Ge(220)'`, `'Ge(311)'`, `'Be(002)'` or `'PG(110)'`.
  - `EXP.mono.mosaic` is the monochromator mosaic in minutes of arc.
  - `EXP.mono.vmosaic` is the vertical mosaic of monochromator in minutes of arc. If this field is left unassigned, an isotropic mosaic is assumed.
- `EXP.ana` is a structure that describes the analyzer and contains fields as in `EXP.mono` plus the following optional fields:
  - `EXP.ana.thickness` is the analyzer thickness in cm for ideal-crystal reflectivity corrections (Section ??). If no reflectivity corrections are to be made, this field should remain unassigned or set to a negative value.
  - `EXP.ana.Q` is the kinematic reflectivity coefficient for this correction. It is given by  $Q = \frac{4|F|^2}{V_0} \frac{(2\pi)^3}{\tau^3}$ , where  $V_0$  is the unit cell volume for the analyzer crystal,  $F$  is the structure factor of the analyzer reflection, and  $\tau$  is the analyzer reciprocal lattice vector. For `PG(002)`  $Q = 0.1287$ . Leave this field unassigned or make it negative if you don't want the correction done.
  - `EXP.horifoc` is a flag that is set to 1 if a horizontally focusing analyzer is used (Section ??). In this case `EXP.hcol(3)` (see below) is the angular size of the analyzer, as seen from the sample position. If the field is unassigned or equal to -1, a flat analyzer is assumed. Note that this option is *only available with the Cooper-Nathans method*.
- `EXP.hcol(1:4)` are the horizontal Soller collimations in minutes of arc (FWHM beam divergence) starting from the in-pile collimator. In case of a horizontally-focusing analyzer `EXP.hcol(3)` is the angular size of the analyzer, as seen from the sample position.
- `EXP.vcol(1:4)` are the vertical Soller collimations in minutes of arc (FWHM beam divergence) starting from the in-pile collimator.

- `EXP.dir1` defines the scattering direction in the monochromator. This field is equal to 1 or left unassigned if the scattering direction in the monochromator is opposite to that in the sample. Set this field to -1 if the sample and monochromator scattering directions are the same.
- `EXP.dir2` defines the scattering direction in the analyzer. This field is equal to 1 or left unassigned if the scattering direction in the analyzer is opposite to that in the sample. Set this field to -1 if the sample and analyzer scattering directions are the same.
- `EXP.efixed` is the fixed incident or final neutron energy, in meV.
- `EXP.infin` is a flag set to -1 or left unassigned if the final energy is fixed, or set to +1 in a fixed-incident setup.
- `EXP.sample` is a structure that describes the sample. It contains the following fields:
  - `EXP.sample.mosaic` is the FWHM sample mosaic in the scattering plane in minutes of arc. If this field is left unassigned, no sample mosaic corrections (section ??) are performed.
  - `EXP.sample.vmosaic` is the vertical sample mosaic in minutes of arc. If this field is left unassigned, isotropic mosaic is assumed.

Additional fields in the structure `EXP` are needed for Popovici calculations:

- A series of fields contains information on the linear dimensions of spectrometer components. For each object the “sizes” along a the given coordinate axis  $x$  are defined as  $\sqrt{\langle x^2 \rangle}$ . For a rectangular shape of width  $A$  along the  $x$  axis, for example, the correct input would be  $A/\sqrt{12}$ . For a spherical object of diameter  $D$ , the input would be  $D/4$ .
  - `EXP.beam.width` is the width of the source in length units.
  - `EXP.beam.height` is the height of source in length units.
  - `EXP.detector.width` is the width of detector in length units.
  - `EXP.detector.height` is the height of detector in length units.
  - `EXP.mono.width` is the width of the monochromator crystal.

- `EXP.mono.height` is the height of monochromator crystal.
  - `EXP.mono.depth` is the thickness of monochromator crystal.
  - `EXP.ana.width` is the width of analyzer.
  - `EXP.ana.height` is the height of analyzer.
  - `EXP.ana.depth` is the thickness of analyzer.
  - `EXP.monitor.width` is the width of the monitor. This field is only used in `EXP.moncor=1`.
  - `EXP.monitor.height` is the height of the monitor. This field is only used in `EXP.moncor=1`.
- `EXP.sample.shape` is a matrix describing the sample shape. This is a  $3 \times 3$  matrix similar to the matrix  $S_S$  in section ???. A crucial difference is that `EXP.sample.shape` is defined in the  $Q$  coordinate system used in `ResMat.m`. The two matrices are related by a rotation around the vertical axis, which `ResMat.m` performs automatically in the course of its calculations.
  - `EXP.arms(1:5)` are distances between the source and monochromator, monochromator and sample, sample and analyzer, analyzer and detector, and monochromator and monitor, respectively. The 5th element is only needed if `EXP.moncor=1`.
  - `EXP.mono.rv` is the vertical curvature radius of the monochromator crystal. Use a large value for a flat monochromator.
  - `EXP.mono.rh` is the horizontal curvature radius of the monochromator
  - `EXP.ana.rv` is the vertical curvature radius of the analyzer
  - `EXP.ana.rh` is the horizontal curvature radius of the analyzer

## B. `ResMatS.m`

This function is very similar to `ResMat.m`, and calculates the resolution matrix  $M$  and the prefactor  $R_0$ , but works in the  $S$  coordinate system.

```
function [R0,M]=ResMatS(H,K,W,L,EXP)
```

INPUT ARGUMENTS:

- H, K and L specify the wave vector transfer and are given in reciprocal-lattice units.
- W is the energy transfer in meV.
- EXP is a structure that contains all the information on the experimental setup and the required method of resolution calculation. It requires all the fields as for ResMat.m plus additional fields described below.

OUTPUT ARGUMENTS and vectorization are similar to those for ResMat.m.

DEPENDENCIES: ResMatS.m requires ResMat.m, StandardSystem.m, star.m, scalar.m and modvec.m to function properly.

The function ResMatS.m requires the following additional fields in structure EXP:

- EXP.sample.a is the  $a$  lattice constant of the sample in Å.
- EXP.sample.b is the  $b$  lattice constant of the sample in Å.
- EXP.sample.c is the  $c$  lattice constant of the sample in Å.
- EXP.sample.alpha is the  $\alpha$  lattice angle of the sample in degrees of arc.
- EXP.sample.beta is the  $\beta$  lattice angle of the sample in degrees of arc.
- EXP.sample.gamma is the  $\gamma$  lattice angle of the sample in degrees of arc.
- EXP.orient1(1:3) are Miller indexes of the first reciprocal-space orienting vector for the  $S$  coordinate system, as explained in Section ??.
- EXP.orient2(1:3) are Miller indexes of the second reciprocal-space orienting vector for the  $S$  coordinate system, as explained in Section ??.
- EXP.sample.shape is similar to the corresponding field used with ResMat.m, but defines the sample shape in the  $S$  coordinate system.
- EXP.Smooth defines the smoothing parameters as explained in Section ??. Leave this field unassigned if you don't want this correction done.
  - EXP.Smooth.E is the smoothing FWHM in energy (meV). A small number means “no smoothing along this direction”.

- `EXP.Smooth.X` is the smoothing FWHM along the first orienting vector ( $x_0$  axis) in  $\text{\AA}^{-1}$ .
- `EXP.Smooth.Y` is the smoothing FWHM along the  $y$  axis in  $\text{\AA}^{-1}$ .
- `EXP.Smooth.Z` is the smoothing FWHM along the vertical direction in  $\text{\AA}^{-1}$ .

### C. `MakeExp.m`

This function is an example of creating the `EXP` structure for use with `ResMat.m` and `ResMatS.m`. The user is encouraged to make copies of this m-file and edit them to generate particular experimental setups. The function takes no arguments and returns a `ResMatS`-compatible structure `EXP`.

## IV. CRYSTALLOGRAPHY UTILITY FUNCTIONS

*ResLib* contains several routines for crystallographic calculations “free of charge”. These are `star.m`, `scalar.m`, `modvec.m` and `StandardSystem.m`. They are particularly useful in user-supplied prefactor functions used with `ConvRes.m` and `ConvResSMA.m` (see below).

### A. `star.m`

Given direct-space lattice parameters, this function calculates reciprocal-space lattice parameters (or vice-versa).

```
function [V,Vstar,rattice]=star(lattice)
```

INPUT ARGUMENTS:

- `lattice` is a structure that contains the direct or reciprocal-space lattice parameters (note the difference with the structure `EXP.sample` described above!):
  - `lattice.a` is the  $a$  or  $a^*$  lattice constant in  $\text{\AA}$  or  $\text{\AA}^{-1}$ .
  - `lattice.b` is the  $b$  or  $b^*$  lattice constant in  $\text{\AA}$  or  $\text{\AA}^{-1}$ .
  - `lattice.c` is the  $c$  or  $c^*$  lattice constant in  $\text{\AA}$  or  $\text{\AA}^{-1}$ .
  - `lattice.alpha` is the  $\alpha$  or  $\alpha^*$  lattice constant in *radians*.
  - `lattice.alpha` is the  $\beta$  or  $\beta^*$  lattice constant in *radians*.
  - `lattice.gamma` is the  $\gamma$  or  $\gamma^*$  lattice constant in *radians*.

OUTPUT ARGUMENTS: `V` and `Vstar` are the direct (reciprocal) and reciprocal (direct) unit cell volumes depending on whether direct or reciprocal lattice parameters are put in. `rattice` has the same structure as `lattice`, but contains reciprocated cell parameters.

VECTORIZATION: The function can be used to simultaneously reciprocate several sets of lattice parameters. For  $N$  calculations each field of `lattice` should be a vector of length  $N$  or a scalar (if identical values are assumed for a particular cell constant). `V` and `Vstar` and the fields of `rattice` will then be vectors of length  $N$ .

DEPENDENCIES: `star.m` is a stand-alone function that does not rely on any other components in *ResLib*.

## B. `scalar.m`

Calculate the scalar product of two vectors defined by their fractional cell coordinates. Alternatively, calculates the scalar product of two reciprocal vectors defined by their Miller indexes.

```
function s=scalar(x1,y1,z1,x2,y2,z2,lattice)
```

INPUT ARGUMENTS:

- `lattice` is as in `star.m`. It contains all the lattice parameters for this calculation. Alternatively, it contains reciprocal-lattice parameters.
- `x1`, `y1`, `z1` are fractional cell coordinates of the first vector. Alternatively, they define the Miller indexes of the first reciprocal-lattice vector.
- `x2`, `y2`, `z2` are fractional cell coordinates of the second vector. Alternatively, they define the Miller indexes of the second reciprocal-lattice vector.

OUTPUT ARGUMENTS: The returned value is the scalar product of the two vectors.

VECTORIZATION: The function can be used to simultaneously calculate several scalar products. For  $N$  calculations, `x1–z2` are vectors of length  $N$  or scalars. Correspondingly, each field of `lattice` should be a vector of length  $N$  or a scalar. The returned value will be a vector of length  $N$  as well.

DEPENDENCIES: `scalar.m` is a stand-alone function that does not rely on any other components in *ResLib*.

## C. `modvec.m`

Very similar to `scalar.m`, but takes only one vector for input and returns its modulus.

```
function m=modvec(x,y,z,lattice)
```

DEPENDENCIES: `modvec.m` relies on `scalar.m`.

## D. `StandardSystem.m`

Based on a structure `EXP` of type used by `ResMatS.m` constructs the standard basis set of the  $S$  coordinate system, as defined in Section ??.

`[x,y,z,sample,rsample]=StandardSystem(EXP)`

INPUT ARGUMENTS:

- `EXP` is a structure of type used with `ResMatS.m`. The fields actually used are: `EXP.sample.a`, `EXP.sample.b`, `EXP.sample.c`, `EXP.sample.alpha`, `EXP.sample.beta`, `EXP.sample.gamma`, `EXP.orient1` and `EXP.orient2`.

OUTPUT ARGUMENTS:

- `x(1:3)` are Miller indexes of the first unit length basis vector for the  $S$  system.
- `y(1:3)` are Miller indexes of the second unit length basis vector for the  $S$  system.
- `z(1:3)` are Miller indexes of the third unit length basis vector for the  $S$  system.
- `sample` is a structure containing lattice parameters for the sample in `EXP`. It is of the same form as used by `star.m`, `modvec.m` and `scalar.m`.
- `rsample` is a structure containing reciprocal lattice parameters for the sample in `EXP`. It is of the same form as used by `star.m`, `modvec.m` and `scalar.m`.

VECTORIZATION: If `EXP` is a vector of length  $N$ , then `x`, `y` and `z` are  $3 \times N$  matrices and the fields of `sample` and `rsample` are vectors of length  $N$ .

DEPENDENCIES: `StandardSystem.m` relies on `star.m`, `scalar.m` and `modvec.m`.

## V. CONVOLUTION WITH RESOLUTION FUNCTION: GENERAL 4-D CONVOLUTION

In many cases it is convenient to break down the parameterized model cross section function  $S(\mathbf{Q}, \omega, \mathbf{p})$  into a sum of partial contributions of several excitation “modes”:

$$S(\mathbf{Q}, \omega, \mathbf{p}) = \sum_{m=1}^{N_m} S_m(\mathbf{Q}, \omega, \mathbf{p}). \quad (17)$$

To save numerical computation time, each partial contribution can be further broken down into a product of a “slowly” and “rapidly” varying components:

$$S_m(\mathbf{Q}, \omega, \mathbf{p}) = P_m(\mathbf{Q}, \omega, \mathbf{p}) \tilde{S}_m(\mathbf{Q}, \omega, \mathbf{p}). \quad (18)$$

Here the “slow” prefactor  $P_m(\mathbf{Q}, \omega, \mathbf{p})$  is expected to be almost constant within the volume of the resolution ellipsoid, and therefore does not need to be numerically folded with the resolution function. Typical examples are polarization factors and magnetic form factors. For the “fast” part of the cross section  $\tilde{S}_m(\mathbf{Q}, \omega, \mathbf{p})$  there is no escape and a full convolution has to be performed numerically in four dimensions. Given the above definitions and by a substitution of integration variables Equation ?? can be rewritten as:

$$\begin{aligned} \frac{d\sigma}{d\Omega dE'} &\propto R_0(\mathbf{Q}_0) \frac{1}{\sqrt{\det(M)}} \times \\ &\times \sum_{m=1}^{N_m} \left\{ P_m(\mathbf{Q}_0, \omega_0, \mathbf{p}) \int_{-\pi/2}^{\pi/2} d\phi_1 \int_{-\pi/2}^{\pi/2} d\phi_2 \int_{-\pi/2}^{\pi/2} d\phi_3 \int_{-\pi/2}^{\pi/2} d\phi_4 \tilde{S}_m(\mathbf{Q}, \omega, \mathbf{p}) \times \right. \\ &\times \left. \frac{\exp\left(-\frac{1}{2} \tan^2 \phi_1 - \frac{1}{2} \tan^2 \phi_2 - \frac{1}{2} \tan^2 \phi_3 - \frac{1}{2} \tan^2 \phi_4\right)}{\cos^2 \phi_1 \cos^2 \phi_2 \cos^2 \phi_3 \cos^2 \phi_4} \right\}, \quad (19) \end{aligned}$$

where the new variables are defined as:

$$\begin{aligned} Q_x - Q_{0,x} &= -\frac{1}{\sqrt{\tilde{M}_{11}}} \tan \phi_1, \\ Q_y - Q_{0,y} &= -\frac{1}{\sqrt{\tilde{M}_{22}}} \tan \phi_2 - \frac{\tilde{M}_{12}}{\tilde{M}_{22} \sqrt{\tilde{M}_{11}}} \tan \phi_1, \\ \omega - \omega_0 &= -\frac{1}{M_{33}} \tan \phi_3 - \frac{M_{23}}{M_{33} \sqrt{\tilde{M}_{22}}} \tan \phi_2 - \frac{\left(\frac{M_{13}}{M_{33}} - \frac{M_{23}}{M_{22}} \frac{\tilde{M}_{12}}{M_{22}}\right)}{\sqrt{\tilde{M}_{11}}} \tan \phi_1, \\ Q_z - Q_{0,z} &= -\frac{1}{\sqrt{M_{44}}} \tan \phi_4, \end{aligned}$$

$$\begin{aligned}\tilde{M}_{11} &= M_{11} - \frac{M_{13}^2}{M_{33}} - \frac{\tilde{M}_{12}^2}{\tilde{M}_{22}}, \\ \tilde{M}_{12} &= M_{12} - \frac{M_{13}M_{23}}{M_{33}}, \\ \tilde{M}_{22} &= M_{22} - \frac{M_{23}^2}{M_{33}}.\end{aligned}$$

If this integral is evaluated numerically by sampling of integrand in a set of points uniformly distributed in  $\phi$ -space, the sampling points will be distributed according to a Lorenzian in  $(\mathbf{Q}, \omega)$ -space. The widths of the 4D-Lorenzian will be determined by the resolution ellipsoid. More points will get sampled near the resolution center, and less on the "tails". This allows one to achieve much better results with fewer sampling points.

#### A. `ConvRes.m`

Numerically convolutes a user-supplied cross section function (defined in separate m-files) with the resolution function in a given set of points in  $(\mathbf{Q}, \omega)$ -space. This process requires laborious calculations and one always has to compromise between computation times and accuracy. The sharper the intrinsic features of the cross section, the more difficult it is to get an accurate result. In particular, if the cross section has a singular form (energy  $\delta$ -function), straightforward 4D convolution doesn't work at all. A work-around is to artificially introduce some intrinsic width in the excitations. A much better approach is to use `ConvResSMA.m`, described below. In fact, try to use `ConvResSMA.m` whenever possible, as it is much faster than `ConvRes.m`. `ConvRes.m` implements two algorithms of numerical integration. The "fixed-sampling" method samples the "fast" cross section on a fixed grid in  $\phi$ -space. Alternatively, the Monte Carlo algorithm can be used, eliminating any "structured" artifacts (a common problem with fixed sampling), but introduces "noise". MC is good for simulations, but the irreproducible "noise" will confuse many least-squares fitting routines. For an example of use see `ConvDemo.m`.

```
function conv=ConvRes(sqw,pref,H,K,L,W,EXP,METHOD,ACCURACY,p)
```

INPUT ARGUMENTS:

- 'sqw' is the name of the user-supplied "fast" model cross section. It must be defined in a separate m-file. The requirements for this function are specified below.

- 'pref' is the name of the user-supplied "slow" cross section prefactor. It must be defined in a separate m-file. The requirements for this function are specified below. If this argument is set to [], the prefactors for all modes will be assumed to be unity.
- H, K, L and W specify the wave vector and energy transfers at which the convolution is to be calculated (i.e., define  $\mathcal{Q}_0$ ). H, K and L are given in reciprocal lattice units, and W in meV.
- EXP is a structure that contains all the information on the experimental setup. It has the form used with ResMatS.m, as described above.
- METHOD is a string that specifies which 4D-integration method to use, and ACCURACY determines the number of sampling points in the integration. The following options are available:
  - METHOD='fix': Sample the cross section on a fixed grid of points uniformly distributed in  $\phi$ -space. If this option is chosen,  $2*\text{ACCURACY}(1)+1$  points are sampled along  $\phi_1$ ,  $\phi_2$  and  $\phi_3$ , and  $2*\text{ACCURACY}(2)+1$  along  $\phi_4$  (vertical direction).
  - METHOD='mc': 4D Monte Carlo integration. The cross section is sampled in  $1000*\text{ACCURACY}$  randomly chosen points, uniformly distributed in  $\phi$ -space.
- p is a parameter that is passed on, without change, to sqw and pref.

OUTPUT ARGUMENTS: conv is the calculated value of the cross section, folded with the resolution function, at the give  $\mathcal{Q}_0$  position.

VECTORIZATION: The code is vectorized with respect to H, K, L, W and EXP. It can be used to simultaneously calculate convolutions at several  $\mathcal{Q}_0$ -points or even for several experimental setups. Any vectors among H, K, L, W and EXP must be of the same length. If any of these are scalars, they will be automatically replicated into vectors of appropriate length. For input vectors of length  $N$  the result is, correspondingly, a length- $N$  row-vector of convolution values.

DEPENDENCIES: To properly function ConvRes.m requires the following m-files from ResLib: ResMatS.m, ResMat.m, StandardSystem.m, star.m, scalar.m, modvec.m.

The "fast" cross section functions are defined in a separate user-supplied m-file that must be of the form: function s=sqw(H,K,L,W,p). The arguments H, K and L specify

the scattering vector in rec. lattice units, and  $W$  is the energy transfer in meV.  $p$  is an additional parameter that can be anything: a number, an array, a structure, or a cell array. See `SqwDemo.m` for an example. The function `sqw` must be vectorized, i.e., if  $H$ ,  $K$ ,  $L$  and  $W$  are vectors of length  $N$ , the output should be a vector of values of the same length. Even when some of the input arguments are vectors of length  $N$ , others may be scalars, and in this case should be treated as equivalents on length- $N$  vectors of constant values. `sqw` can return several partial contribution to the cross section. For  $N_m$  modes and  $N$  data points the output argument must be a  $N_m \times N$  matrix.

The “slow” cross section prefactors are defined in a separate user-supplied m-file that must be of the form: `function p=pref(H,K,L,W,p,EXP)`. `pref` is similar to `sqw` function, except that, unlike the latter, it has direct access to experimental conditions and must list `EXP` as its argument. See `PrefDemo.m` for an example. If `sqw` returns several partial contributions to the cross section, `pref` is expected to do the same. For  $N_m$  modes and  $N$  data points the output argument must be a  $N_m \times N$  matrix.

## VI. CONVOLUTION WITH RESOLUTION FUNCTION: SINGLE MODE APPROXIMATION AND VOIGT PROFILES

In many cases the “fast” dynamic structure factor can be broken written as:

$$\tilde{S}_m(\mathbf{Q}, \omega, \mathbf{p}) = s_m(\mathbf{Q}, \mathbf{p})\delta(\omega - \omega_m(\mathbf{Q}, \mathbf{p})). \quad (20)$$

Here  $s_m(\mathbf{Q}, \mathbf{p})$  is the intensity of the  $m$ -th excitation branch, and  $\omega_m(\mathbf{Q}, \mathbf{p})$  is its dispersion relation. `ConvRes.m` becomes unusable in this case, as it can not handle infinitely sharp features of the cross section. The solution is to eliminate the  $\delta$ -functions analytically, also reducing a laborious 4-D numerical convolution to a much faster 3-D convolution.

Equation ?? can be generalized to the case when each mode is characterized by some intrinsic Lorentzian energy-width  $\Gamma_m(\mathbf{Q}, \mathbf{p})$ :

$$\tilde{S}_m(\mathbf{Q}, \omega, \mathbf{p}) = \frac{1}{\pi} s_m(\mathbf{Q}, \mathbf{p}) \frac{\Gamma_m(\mathbf{Q}, \mathbf{p})}{\Gamma_m(\mathbf{Q}, \mathbf{p})^2 + [\omega - \omega_m(\mathbf{Q}, \mathbf{p})]^2} \quad (21)$$

While a convolution of a Gaussian and a Lorentzian (so-called Voigt function) can not be computed analytically, some very efficient numerical approximations are available [? ].

The  $\phi$ -transformation is performed only in 3 dimensions in either case. Exact expressions used in ResLib are as follows: Equation ?? is rewritten as:

$$\begin{aligned} \frac{d\sigma}{d\Omega dE'} &\approx R_0(\mathbf{Q}_0) \frac{1}{\sqrt{\tilde{M}_{11}\tilde{M}_{22} - \tilde{M}_{12}^2}\sqrt{\tilde{M}_{33}}} \times \\ &\times \sum_{m=1}^{N_m} \left\{ P_m(\mathbf{Q}_0, \omega_0, \mathbf{p}) \int_{-\pi/2}^{\pi/2} d\phi_1 \int_{-\pi/2}^{\pi/2} d\phi_2 \int_{-\pi/2}^{\pi/2} d\phi_3 \tilde{s}_m(\mathbf{Q}, \mathbf{p}) V(\tilde{\omega}_m(\mathbf{Q}, \mathbf{p}), \tilde{\Gamma}_m(\mathbf{Q}, \mathbf{p})) \times \right. \\ &\times \left. \frac{\exp\left(-\frac{1}{2}\tan^2\phi_1 - \frac{1}{2}\tan^2\phi_2 - \frac{1}{2}\tan^2\phi_3\right)}{\cos^2\phi_1 \cos^2\phi_2 \cos^2\phi_3} \right\}, \quad (22) \end{aligned}$$

where the new variables are defined as:

$$\begin{aligned} \tilde{\Gamma}_m(\mathbf{Q}, \mathbf{p}) &= \frac{1}{\sqrt{\tilde{M}_{33}}}\Gamma_m(\mathbf{Q}, \mathbf{p}), \\ \tilde{\omega}_m(\mathbf{Q}, \mathbf{p}) &= \omega - \omega_m(\mathbf{Q}, \mathbf{p}) + \frac{M_{13}Q_x + M_{23}Q_y}{\sqrt{\tilde{M}_{33}}}, \\ V(x, y) &\equiv \frac{1}{\pi} \int dt \exp(-t^2) \frac{y}{y^2 + (t - x)^2}, \\ Q_y - Q_{0,y} &= \frac{\sqrt{\tilde{M}_{11}}}{\sqrt{\tilde{M}_{11}\tilde{M}_{22} - \tilde{M}_{12}^2}} \tan\phi_2, \end{aligned}$$

$$\begin{aligned}
Q_x - Q_{0,x} &= \frac{1}{\sqrt{\tilde{M}_{11}}} \tan \phi_1 - \frac{\tilde{M}_{12}}{\sqrt{\tilde{M}_{11}} \sqrt{\tilde{M}_{11} \tilde{M}_{22} - \tilde{M}_{12}^2}} \tan \phi_2, \\
Q_z - Q_{0,z} &= -\frac{1}{\sqrt{M_{44}}} \tan \phi_4, \\
\tilde{M}_{11} &= M_{11} - \frac{M_{13}^2}{M_{33}}, \\
\tilde{M}_{12} &= M_{12} - \frac{M_{13} M_{23}}{M_{33}}, \\
\tilde{M}_{22} &= M_{22} - \frac{M_{23}^2}{M_{33}}.
\end{aligned}$$

#### A. ConvResSMA.m

Handles user-supplied cross section functions written in the single-mode form. Having one dimension less to deal with, it is much faster than `ConvRes.m` and should be used in its stead whenever possible. For an example of use see `ConvDemo.m`.

```
function conv=ConvResSMA(sqw,pref,H,K,L,W,EXP,METHOD,ACCURACY,p)
```

INPUT ARGUMENTS:

- 'sqw' is the name of the user-supplied "fast" model single-mode cross section. It must be defined in a separate m-file. The requirements for this function are listed below.
- 'pref', H, K, L, W, p, and EXP have exactly the same meaning and form as for `ConvRes.m`.
- METHOD is a string that specifies which 3D-integration method to use, and ACCURACY determines
  - METHOD='fix': Sample the cross section on a fixed grid of points uniformly distributed in  $\phi$ -space. If this option is chosen,  $2*\text{ACCURACY}(1)+1$  points are sampled along  $\phi_1$  and  $\phi_2$ , and  $2*\text{ACCURACY}(2)+1$  along  $\phi_3$  (vertical direction).
  - METHOD='mc': 3D Monte Carlo integration. The cross section is sampled in  $1000*\text{ACCURACY}$  randomly chosen points, uniformly distributed in  $\phi$ -space.

OUTPUT ARGUMENTS: See output arguments for `ConvRes.m`.

VECTORIZATION: See vectorization of `ConvRes.m`.

DEPENDENCIES: To properly function `ConvResSMA.m` requires the following m-files from *ResLib*: `ResMatS.m`, `ResMat.m`, `StandardSystem.m`, `star.m`, `scalar.m`, `modvec.m`.

The user-supplied single mode cross section `sqw` is defined in a separate m-file. It must have the form: `function [w0,S,HWHM]=sqw(H,K,L,p)`. The input arguments are as those for the cross section function for `ConvRes.m`. See `SMADemo.m` for an example. For  $N_m$  modes and  $N$  data points the output arguments are  $N_m \times N$  matrices. For mode  $m$  at  $i$ -th reciprocal-space point  $(H(i),K(i),L(i))$ ,  $s(m,i)$  is the mode intensity,  $w0(m,i)$  is its energy in meV, and  $HWHM(m,i)$  is its Lorentzian energy half-width. If  $HWHM(m,i)=0$ , the mode is assumed to have a  $\delta$ -function energy profile at this point.

## VII. LEAST-SQUARES FITTING OF CONVOLUTED CROSS SECTIONS

### A. `FitConv.m`

This function provides Levenberg-Marcquardt least squares fitting of convoluted cross sections to experimental data. The algorithm is exactly as in NR 15.5. Analytic derivatives are not supported in the current version. This function calls `ConvRes.m` for convolution calculations at each iteration step. An example of use is given in `FitDemo.m`. The function call has the following form:

```
function [pa,dpa,chisqN,sim,CN,PQ,nit,kvg,details] =  
FitConv(H, K, L, W, EXP, Iobs, dIobs, sqw, pref, pa, ia, METHOD, ACCURACY,  
nitmax, tol, dtol)
```

INPUT ARGUMENTS:

- `'sqw'`, `'pref'`, `METHOD` and `ACCURACY` are exactly as in `ConvRes.m`.
- `H`, `K`, `L`, `W`, `EXP` are vectors that contain the wave vector components, energy transfers, and experimental conditions for each experimental data point. They all must be of the same length or scalars. See `ConvRes.m` for more details.
- `Iobs`, `dIobs` are vectors of the same length as `H-EXP`, and contain the observed scattering intensities and error bars for each data point, respectively.
- `pa` is the initial guess for parameter values. This should be a 1-dimensional array of the form accepted as the parameter `p` by the `sqw` and `pref` functions.
- `ia` (optional) - a list with zero elements corresponding to fixed parameters and ones for parameters that should be varied. This array should be of the same size as `pa`. The default value is `ia=ones`.
- `nitmax` (optional)- the maximum number of iterations allowed before the program exits. The default value is `nitmax=20`.
- `tol` (optional)- the convergence criterion: convergence declared when  $\chi^2$  decreases by a relative amount less than `tol`. The default value is `tol=0.001`.

- `dtol` (optional)- parameter for relative change in numerical derivatives, default `dtol=1e-5`.

OUTPUT ARGUMENTS:

- `pa, dpa` - refined parameters and error bars.
- `chi2N` -  $\chi^2$  normalized by degrees of freedom.
- `CN` - normalized correlation matrix - includes varying parameters only (in order).
- `PQ` - probability that  $\chi^2$  exceeds that observed.
- `nit` - number of iterations to convergence.
- `kvg` - convergence flag:
  - `kvg= 0` did not converge due to too many iterations `nit`  $\geq$  `nitmax`
  - `kvg= 1` converged normally
  - `kvg= 2` questionable convergence (final  $\lambda > 1 \cdot 10^{-3}$  )
- `sim` - (optional) fitted value of convoluted cross section at input points
- `details` - optional output structure with fields:
  - `chisq` -raw  $\chi^2$
  - `DF` -degrees of freedom
  - `Ndata` -number of data points
  - `Npar` -total number of parameters
  - `Nvar` -number of parameters varied
  - `C` -raw covariance matrix
  - `final_lamda` -final value of  $\lambda$  used in the fitting process
  - `yf` -fitted values of function at input points

DEPENDENCIES: To properly function `FitConv.m` requires the following m-files from *ResLib*: `ConvRes.m`, `ResMatS.m`, `ResMat.m`, `StandardSystem.m`, `star.m`, `modvec.m`, `scalar.m`.

## B. FitConvSMA.m

This function is very similar to `FitConv.m`, but uses single-mode cross sections as in `ConvResSMA.m`. The function call has the following form:

```
function [pa,dpa,chisqN,sim,CN,PQ,nit,kvg,details] =  
FitConvSMA(H, K, L, W, EXP, Iobs, dIobs, sqw, pref, pa, ia, METHOD,  
ACCURACY, nitmax, tol, dtol)
```

INPUT ARGUMENTS:

- 'sqw', 'pref', METHOD and ACCURACY are exactly as in `ConvResSMA.m`
- Other parameters as in `FitConv.m`.

DEPENDENCIES: To properly function `FitConvSMA.m` requires the following m-files from *ResLib*: `ConvResSMA.m`, `ResMatS.m`, `ResMat.m`, `StandardSystem.m`, `star.m`, `modvec.m`, `scalar.m`.

## C. FitConvBoth.m

This function fits a sum of a "standard" and single mode-type cross sections to the data. This is very useful when the scattering contains both a singular and a diffuse part. The function call has the following form:

```
function [pa,dpa,chisqN,sim,CN,PQ,nit,kvg,details]=  
FitConvBoth(H, K, L, W, EXP, Iobs, dIobs, sqw, pref, sma, smapref, pa, ia,  
METHOD, ACCURACY, SMACCURACY, nitmax, tol, dtol)
```

INPUT ARGUMENTS:

- 'sma', 'smapref', METHOD, and SMACCURACY are exactly as in `ConvResSMA.m`.
- 'sma', 'pref', and ACCURACY are as in `ConvRes.m`.
- Other parameters as in `FitConv.m`.

DEPENDENCIES: To properly function `FitConvBoth.m` requires the following m-files from *ResLib*: `ConvResSMA.m`, `ConvRes.m`, `ResMatS.m`, `ResMat.m`, `StandardSystem.m`, `star.m`, `modvec.m`, `scalar.m`.

## VIII. GRAPHIC REPRESENTATIONS OF RESOLUTION ELLIPSOIDS

It is often very helpful to visualize resolution ellipsoids for each point in an inelastic scan, see how they change from one point to the next, and try to understand how they are oriented relative to the dispersion surface in the system under investigation. Also, while the resolution matrix contains all the information on the instrumental resolution at a given point, parameters such as resolution volume, projected energy width, Bragg peak width, etc., are much more useful in planning actual measurements. *ResLib* provides two routines for visualizing resolution ellipsoids and calculating the most relevant resolution parameters.

### A. `ResPlot.m`

This function plots projections and sections of resolution ellipsoids for a user-defined scan, and calculates a bunch of relevant resolution characteristics for the center-point of the scan. As an option, it also plots the dispersion relation calculated from a user-supplied single-mode cross section function, of the type used by `ConvResSMA.m`. The projection and section planes for resolution ellipsoids are defined by the orienting vectors specified in `EXP`. By changing these one can get projections and sections with any desired reciprocal-space planes. An example of use can be found in `PlotDemo.m`.

```
function ResPlot(H,K,L,W,EXP,SMA,SMAp)
```

INPUT ARGUMENTS:

- `H`, `K`, `L`, `W`, `EXP` are vectors that contain the wave vector components, energy transfers, and experimental conditions for each experimental data point. They all must be of the same length or scalars. See `ConvRes.m` for more details. `ResPlot.m` is designed to work with a single scan, measured in a particular experimental setup. Therefore, calculations for all scan points will be performed using experimental conditions for the center-point of the vector `EXP` (center of scan).
- `SMA` (optional) is the name of a single-mode cross section function to be used for plotting the dispersion relation.
- `SMAp` (optional) - parameters to be passed to the cross section function `SMA`.

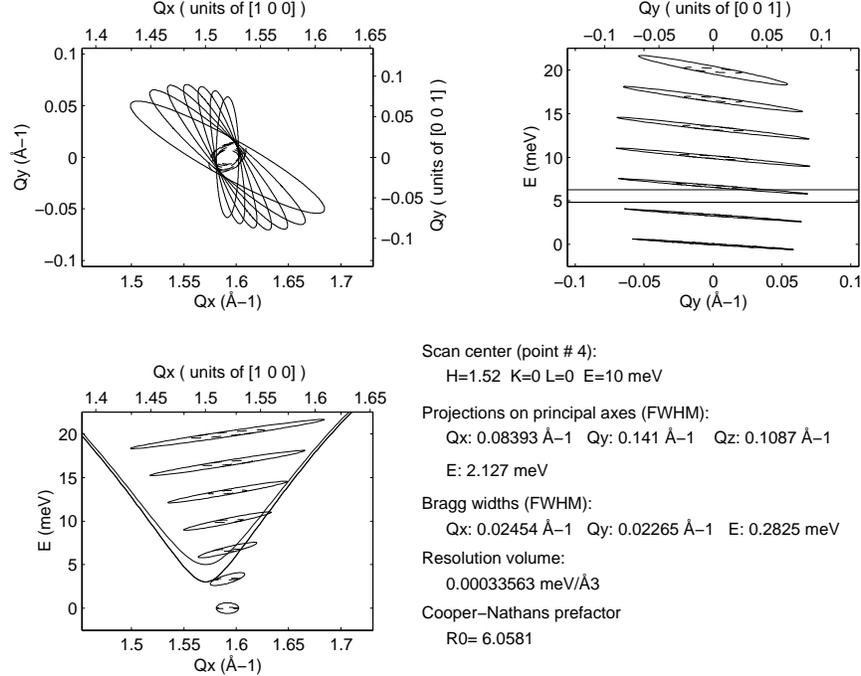


FIG. 1: *Upper left*: projection onto the scattering plane (large ellipses) and constant-energy sections (inner ellipses). *Lower left*:  $Q_x - E$  projections and sections. Additional lines (here: parabola) show the dispersion of the two branches defined in the single mode cross section function. The  $Q_y$  and  $Q_z$  values for this dispersion curve are taken for the mid-point in the scan. *Upper right*:  $Q_y - E$  projections and sections. Additional lines (here: straight lines) show the dispersion of the two branches defined in the single-mode cross section function. The  $Q_x$  and  $Q_z$  values for this dispersion curve are taken for the mid-point in the scan.

OUTPUT: All the information is sent to the current figure. A typical output looks like this:

DEPENDENCIES: To properly function ResPlot.m requires the following m-files from ResLib: ResMatS.m, ResMat.m, StandardSystem.m, star.m, modvec.m, scalar.m.

### B. ResPlot3D.m

This function is similar to ResPlot.m but plots a very neat 3D representation of resolution ellipsoids, projections and dispersion surfaces. See Plot3DDemo.m for an example of use.

function ResPlot3D (H,K,L,W,EXP,RANGE,EllipsoidStyle,XYStyle,XESStyle, YESStyle, SMA, SMap, SXg, SYg)

INPUT ARGUMENTS:

- `H`, `K`, `L`, `W`, `EXP` are as in `ResPlot.m`.
- `RANGE` defines the range of momentum and energy transfer to use in the plot. `RANGE` must be of the form `[Qxmin, Qxmax, Qymin, Qymax, Emin, Emax]`, in  $\text{\AA}^{-1}$  and `meV`, respectively.
- `EllipsoidStyle` (optional) is the color of resolution ellipsoids in the plot. Default is `EllipsoidStyle='red'`.
- `XYStyle` (optional) - line style to use for the projection onto the scattering plane, in the format used with MatLab's `plot` command (something like `'r--'` for a red dashed line). Use the value `'none'` to suppress projections.
- `XESStyle` (optional) - line style to use for the projection onto the  $Q_x - E$  plane, in the format used with MatLab's `plot` command. Use the value `'none'` to suppress projections.
- `YESStyle` (optional) - line style to use for the projection onto the  $Q_y - E$  plane, in the format used with MatLab's `plot` command. Use the value `'none'` to suppress projections.
- `SMA` and `SMAP` (both optional) are as in `ResPlot.m`.
- `SXg` and `SYg` (both optional) define the 2D grid of  $Q_x$  and  $Q_y$  values for plotting the dispersion surfaces. They must be of the same form as used in MatLab's `surf` command, and as generated by `meshgrid`. A good example of use is given in `Plot3Ddemo.m`. If omitted, the dispersion surface is plotted of a  $40 \times 40$  grid within the boundaries specified by the first 4 elements of `RANGE`.

OUTPUT: All the information is sent to the current figure.

DEPENDENCIES: To properly function `ResPlot3D.m` requires the following m-files from *ResLib*: `ResMatS.m`, `ResMat.m`, `StandardSyssem.m`, `star.m`, `modvec.m`, `scalar.m`.

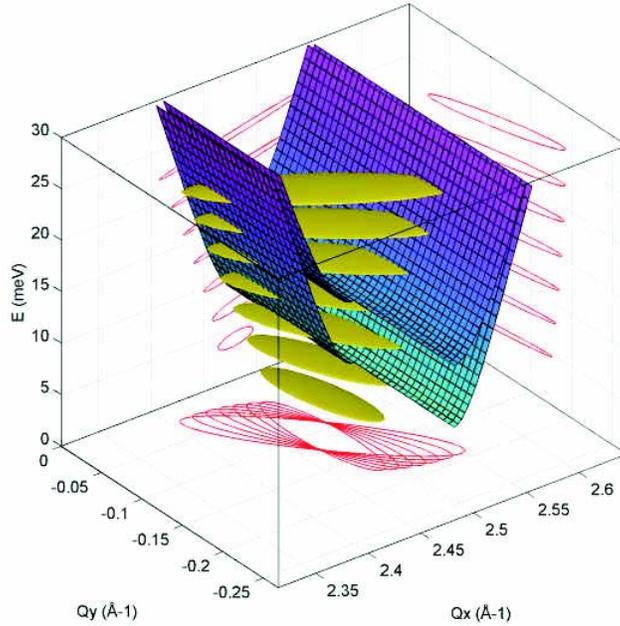


FIG. 2: A typical image generated by `ResPlot3D.m`.

## IX. DOWNLOADS, PACKAGING, INSTALLATION AND FEEDBACK

### A. File download

The most recent version of all *ResLib* files are available here for download as a .zip archive from <http://nsmc1.ssd.ornl.gov/~zhelud/reslib/>.

### B. Packaging

- `ResMat.m` - calculate Cooper-Nathans resolution matrices.
- `ResMatS.m` - same thing, in a coordinate system defined by the scattering vector.
- `ConvRes.m` - convolute a user-defined cross-section function with the resolution.
- `ConvResSMA.m` - convolute a user-defined single-mode cross-section with the resolution.
- `FitConv.m` - fit convoluted cross section to experimental data.
- `FitConvSMA.m` - fit convoluted SMA cross section to experimental data.

- `FitConvBoth.m` - fit convoluted sum of "standard" and SMA cross sections to experimental data.
- `ResPlot.m` - 2-D visualization of resolution ellipsoids and dispersion.
- `ResPlot3D.m` - 3-D visualization of resolution ellipsoids and dispersion.
- `scalar.m` - scalar product of vectors defined by fractional cell coordinates or Miller indexes.
- `modvec.m` - length of vectors defined by fractional cell coordinates or Miller indexes.
- `star.m` - calculate reciprocal-lattice parameters, unit cell volume and reciprocal volume.
- `StandardSystem.m` - calculate components of standard sample-centered orthonormal basis set.
- `MakeExp.m` - Example of setting up a structure that contains details on experimental conditions for use with `ResMat`, `ConvRes` and other functions.
- `SqwDemo.m` - example of a user cross section function for use with `ConvRes`.
- `SMADemo.m` - example of a user single-mode cross section function for use with `ConvResSMA`.
- `PrefDemo.m` - example of a "slowly prefactor" function for use with `ConvRes`, `SqwDemo`, `ConvResSMA` and `SMADemo`.
- `ConvDemo.m` - a demo script of the convolution routines.
- `FitDemo.m` - a demo script of the fitting routines.
- `PlotDemo.m` - a demo script of `ResPlot`
- `Plot3DDemo.m` - a demo script for `ResPlot3D`
- `Demo.dat` - an actual data set used in `FitDemo`
- `ResLib.m` - list of functions supplied in *ResLib* for use with the MatLab `help` command.

- `Manual.pdf` - this file.
- `ResLib.jpg` - *ResLib* logo in JPEG format.

### C. Installation

Uncompress the archive to a separate new directory and add it to your MatLab path by using the `addpath` command. Run the `ConvDemo` script to see that *ResLib* works on your system. Run the `PlotDemo` and `Plot3DDemo` scripts to get some nice graphics. Change to the directory where the files are installed (to have `Demo.dat` in the current directory) and try `FitDemo`.

### D. Feedback

The author welcomes any comments, questions, suggestion and bug reports sent by e@mail to `zhelud@bigfoot.com`.

## X. CREDITS

Many ideas and entire code fragments are borrowed from IDL routines written at Johns Hopkins University by Collin L. Broholm and Igor Zaliznyak (now at BNL). Levenberg-Marquardt algorithm based on C Numerical Recipies, as adapted for MatLab by Stephen E. Nagler (ORNL). Voigt function by A. N. Maurellis. The library was originally developed at Brookhaven National Laboratory, but is now maintained at Oak Ridge National Laboratory.

## XI. DISCLAIMER

There obviously is no guarantee that the above routines work right. 3-axis resolution calculations are a tricky business and have on many occasions led to incorrect interpretation of experimental data, outright wrong publications, and destruction of glorious careers of

young neutron scatterers. Use these m-files freely... at your own risk.

---

- M. J. Cooper and R. Nathans, *Acta Cryst.* **23**, 357, (1967).
- N. J. Chesser and J. D. Axe, *Acta Cryst.* **A29**, 160, (1972).
- M. Popovici, *Acta Cryst* **A31**, 507 (1975).
- G. E. Bacon and R. D. Lowde, *Acta Cryst.* (1948).
- S. M. Shapiro, N. J. Chesser, *Nucl.Instr.& Meth.* (1972).
- T. Riste and K. Otnes, *Nucl. Instr.& Meth.* (1969); B. Dorner and A. Kollmar, *J.Appl. Cryst.* (1974).
- I. Zaliznyak, private communication.
- S. A. Werner and R. Pynn, *J. Appl. Phys.* **42**, 4736, (1971).
- J. Humlicek, *J. Q. Spec. Rad. Transfer* **27**, 437 (1982); F. Schreier, *J. Q. Spec. Rad. Transfer* **48**, 743 (1992).