Inverse Problems
Assignment 3

Georgios Sevastakis

April 2024

Introduction to the problem

Our goal is to determine the location (f), area (A), and width (c) of each peak through non-linear least squares inversion by essentially fitting two models: a Gaussian and a Lorentzian.

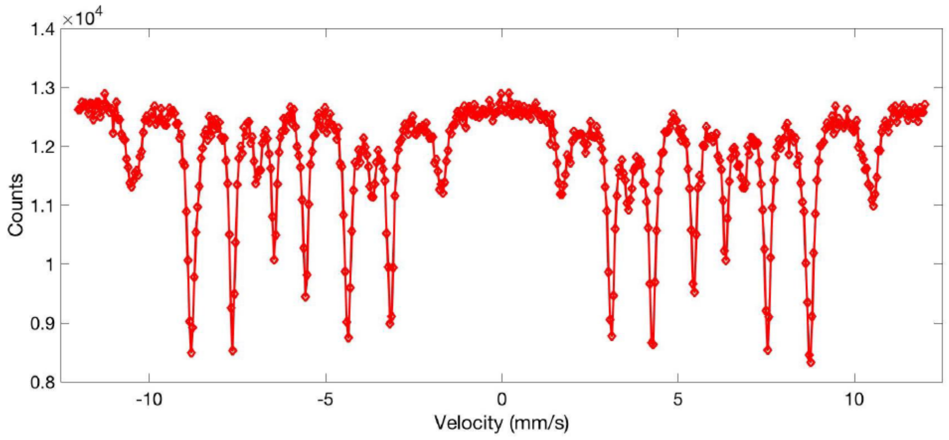

Figure 1: The Mössbauer spectroscopic data to be analyzed in this assignment. For more information about the Mössbauer method, see, e.g., https://en.wikipedia.org/wiki/Mossbauer_spectroscopy. Note that the spectral "peaks" are pointing downwards! Data courtesy of NASA and the University of Mainz.

We are given N data points peaks, each characterized by 3 unknown parameters: its center frequency f_i, area A_i, and width c_i. There are then $M = 3q$ unknown model parameters $m = [A_1, f_1, c_1, ..., A_q, f_q, c_q]^T$. The forward relation is nonlinear and of the form $\mathbf{d} = \mathbf{g}(\mathbf{m})$. The data uncertainties are approximately Gaussian with standard deviation $\sigma = 300$ (counts).

1

In the Gaussian case we have:

$$d_i = g_i(m_j) = \sum_{j=1}^{q} \frac{A_j}{\sqrt{2\pi} c_j} e^{-\frac{\left(z_i - f_j\right)^2}{2c_j^2}} \tag{1}$$

The derivatives are computed analytically and were found to be

$$\frac{\partial g_i}{\partial A_p} = \frac{1}{\sqrt{2\pi}c_p} e^{-\frac{(z_i-f_p)^2}{2c_p^2}} \tag{2}$$

$$\frac{\partial g_i}{\partial f_p} = \frac{A_p}{\sqrt{2\pi}c_p^3}(z_i - f_p) e^{-\frac{(z_i-f_p)^2}{2c_p^2}} \tag{3}$$

$$\frac{\partial g_i}{\partial c_p} = \frac{A_p((z_i - f_p)^2 - c_p^2)}{\sqrt{2\pi}c_p^4} e^{-\frac{(z_i-f_p)^2}{2c_p^2}} \tag{4}$$

2

In the Lorentzian case we have:

$$d_i = g_i(m_j) = \sum_{j=1}^{q} \frac{A_j c_j^2}{(z_i - f_j)^2 + c_j^2} \tag{5}$$

The derivatives were found to be

$$\frac{\partial g_i}{\partial A_p} = \frac{c_p^2}{c_p^2 + (z_i - f_p)^2} \tag{6}$$

$$\frac{\partial g_i}{\partial f_p} = \frac{2A_p c_p^2 (z_i - f_p)}{(c_p^2 + (z_i - f_p)^2)^2} \tag{7}$$

$$\frac{\partial g_i}{\partial c_p} = \frac{2A_p c_p (z_i - f_p)^2}{(c_p^2 + (z_i - f_p)^2)^2} \tag{8}$$

3

To initialize the algorithm an initial guess needs to be made. the closer to the actual data, the higher the chances of convergence. Therefore, by visual inspection of Figure 1. we created the m_{prior} matrix.

Now, the steepest descent algorithm can be written compactly in the following form

$$m_{k+1} = m_k - \varepsilon_k \gamma_k \tag{9}$$

where

$$\gamma_k = G_k^T C_D^{-1}(g_k(m_k) - d) + C_M^{-1}(m_k - m_0) \tag{10}$$

It can be shown in this case that the optimal ε_k is

$$\varepsilon_k = \frac{\gamma_k^T C_M^{-1} \gamma_k}{\gamma_k^T (G_k^T C_D^{-1} G_k + C_M^{-1}) \gamma_k} \tag{11}$$

The C_D matrix constitutes the data covariance matrix, which is a $N \times N$ matrix with the $\sigma^2 = 300^2$ on the diagonals. The C_M matrix, on the other hand, is the model covariance matrix. Since we have no prior information about those, we can stipulate that the uncertainties are independent of each other and assume our own errors (C_m function). Then, we create the C_M matrix by plugging in the uncertainties squared on the diagonal.

```
N_t = 1000
m_k = np.copy(m_prior)

for i in tqdm(range(N_t)):
    Cm = C_m(q)
    G = G_k(N, q, m_k, z_obs.copy(), case)
    g = g_k(m_k, z_obs, case)
    gam = gamma_k(m_k, m_prior, d_obs, G, g, Cm)
    eps = epsilon_k(gam, G, Cm)

    m_k = m_k - eps * gam
```

Listing 1. Code snippet of the steepest descent algorithm namely the gist of the code.

In the code above, the steepest descent algorithm is presented. For a specified number of iterations, the computations are done as follows: First, we compute the model covariance matrix, then the G matrix based on equations the derivatives computed previously. Subsequently, we compute the g_k matrix based on the equations 1 and 5. Then, we plug in everything into γ_k and next into ε_k and we perform the last step of the algorithm.

4

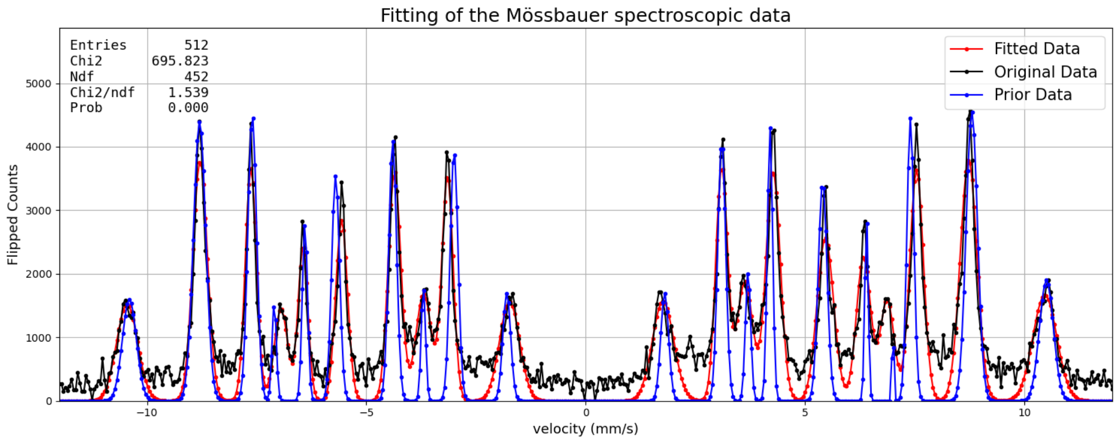

Figure 1: Fitting of the Lorentzian model on the observed data (here the observed data are flipped by subtracting them from the maximum datum value.)

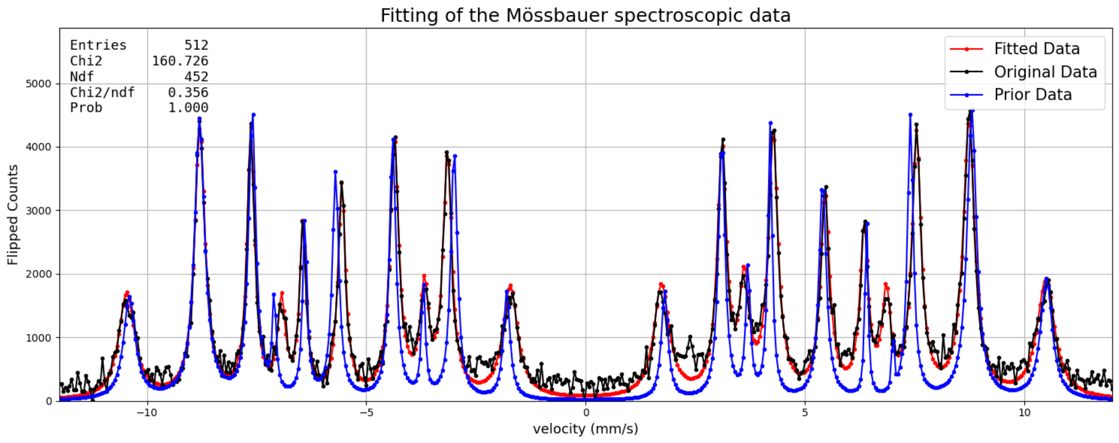

Figure 2: Fitting of the Lorentzian model on the observed data (here the observed data are flipped by subtracting them from the maximum datum value.)

From the Figures presented above, we notice that our implementation yields good results. By taking a closer look we notice that the Lorentzian model is by far the best fit based on the probability value ($p = 1$ in the Lorentzian case, while $p = 0$ for the Gaussian). The Gaussians seem to prefer not to overlap with each other and stay on the 'ground', while the long–tailed Lorentzian are visibly 'raised'. Therefore, we conclude that the Lorentzian fit suits the data best. Note that $p = 1$ does not mean that the fit is perfect, but suggests that we might have overdone with the great error values we put on the parameters $A, fandc$, since there is no such thing as a perfect fit.