ASSIGNMENT 11: BAYESIAN FITTING

E. ARNOLD, M-S. LIU, R. PRABHU & C.S. RICHARDS
THE UNIVERSITY OF CAMBRIDGE

Written in XƎLATEX

BAYESIAN FITTING

PREREQUISITES

The chapter *Scattering Functions* from the *Theory Handbook*.

INTRODUCTION

In this tutorial, we will explore how we can use the intermediate scattering functions (ISFs) measured in a spin echo experiment to determine the paramaters of an adsorbate-substrate system.

THEORETICAL BACKGROUND

We will begin the theory section with an example system. Consider a Ru(0001) surface. This is a close-packed plane. On this surface, a substrate undergoes jump diffusion between the two different types of hollow site. The approach that we will use to determine the system parameters is:

1. Find a theoretical model for the ISF (as a function of momenum transfer $\Delta\mathbf{K}$ and spin-echo time t) that accurately describes the system we are measuring.
2. Use an appropriate fitting technique to fit the experimental ISF measurements to the theoretical model, determining which model parameters provide the best fit.

In the following text, we will, in turn, discuss how to perform these two steps.

Analytical Model for Jump Diffusion in Non-Bravais Lattices

How do we progress towards our objective of finding the system parameters? We first require a theoretical model; this model must describe diffusion in the adsorbate-substrate system. When the system displays weak interactions between adsorbates, and high atomic scale friction, analytical models often exist to describe the intermediate scattering function [1]. It should be emphasised that this is very useful. For most systems, the forces present are too complicated to model;

thus it is usually impossible to produce an analytical solution. Therefore we can fit the model used to some data, allowing us to obtain the system parameters.

In this tutorial, we will consider a close-packed plane. On this plane, an adsorbate can undergo jump diffusion between two different types of sites. We will describe the ISF of this system using the model developed in [2]. The key result from this model, which we can apply to any surface, is that for a adsorbate-substrate system where there are m different types of adsorption sites, the ISF can be written as a sum of m exponentials. Here the result of evaluating the model will be summarised for the system considered in this tutorial.

The system that we are interested in is a close-packed plane on iron. For this system, there are two site types, denoted 1 and 2, with total exit jump rates for the two site types being given by $1/\tau_{12}$ and $1/\tau_{21} \equiv 1/\lambda\tau_{12}$ respectively. The adsorbate concentrations on each site type are given by c_1 and $c_2 \equiv \lambda c_1$ respectively. The lattice constant is denoted a. We will only consider the analytic form of the ISF along the high-symmetry $[1\bar{1}0]$ and $[11\bar{2}]$ directions (see Figure 1).

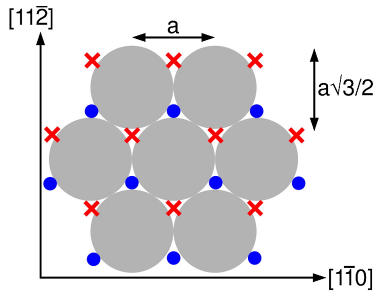

Figure 1 | A schematic of the fcc-(111) surface that we are measuring, with the two different types of hollow site labelled with circles and crosses. Taken from [2]

For the $[1\bar{1}0]$ direction, the ISF is given by:

$$I(\Delta\mathbf{K},t) = \frac{c_1}{n_1}\left|1-\lambda\frac{4\cos\left(\frac{\Delta K a}{2}\right)+2}{3\lambda-3+z}\right|^2 \exp\left(-\frac{1}{6\lambda\tau_{12}}(3\lambda+3+z)t\right);$$
$$+\frac{c_1}{n_2}\left|1-\lambda\frac{4\cos\left(\frac{\Delta K a}{2}\right)+2}{3\lambda-3-z}\right|^2 \exp\left(-\frac{1}{6\lambda\tau_{12}}(3\lambda+3-z)t\right).$$

The constants $n_{1,2}$ and z are given by:

$$n_{1,2} = 1+\lambda\left(\frac{4\cos\left(\frac{\Delta K a}{2}\right)+2}{3\lambda-3\pm z}\right)^2;$$
$$z = \sqrt{9\lambda^2+16\lambda\cos^2\left(\frac{\Delta K a}{2}\right)+16\lambda\cos\left(\frac{\Delta K a}{2}\right)-14\lambda+9}.$$

For the $[11\bar{2}]$ direction, the ISF is given by:

$$I(\Delta\mathbf{K},t) = \frac{c_1}{m_1}\left|1-2\lambda\frac{\exp\left(i\frac{\Delta K a}{\sqrt{3}}\right)+2\exp\left(-i\frac{\Delta K a}{2\sqrt{3}}\right)}{3(\lambda-1+y)}\right|^2 \exp\left(-\frac{1}{2\lambda\tau_{12}}(\lambda+1+y)t\right);$$
$$+\frac{c_1}{m_2}\left|1-2\lambda\frac{\exp\left(i\frac{\Delta K a}{\sqrt{3}}\right)+2\exp\left(-i\frac{\Delta K a}{2\sqrt{3}}\right)}{3(\lambda-1-y)}\right|^2 \exp\left(-\frac{1}{2\lambda\tau_{12}}(\lambda+1+y)t\right).$$

The constants $m_{1,2}$ and y are given by:

$$m_{1,2} = 1+4\lambda\left|\frac{\exp\left(i\frac{\Delta K a}{\sqrt{3}}\right)+2\exp\left(-i\frac{\Delta K a}{2\sqrt{3}}\right)}{3(\lambda-1\pm y)}\right|^2;$$
$$y = \sqrt{\lambda^2+\frac{2\lambda}{9}\left(8\cos\left(\frac{\sqrt{3}\Delta K a}{2}\right)+1\right)+1}.$$

We can see that this provides an analytical model that is paramaterised by two variables: the constants τ_{12} and λ. By fitting experimental ISF measurements to this model ISF, we can estimate the values of these parameters for the system that we are measuring. The constant τ_{12} will be denoted τ for the remainder of the document.

Fitting ISF measurements to the analytical model

In order to find the model that best describes a given set of data, we want to maximise the quantity denoted $P(M|D)$. This represents the probability that the model M is accurate given that the dataset D was obtained. By Bayes' Theorem, this probability can be written as:

$$P(M|D) = \frac{P(D|M)P(M)}{P(D)}$$

where $P(D|M)$ is the probability of obtaining the dataset given that the model is accurate, $P(M)$ is the probability that the model is accurate prior to obtaining the dataset, and $P(D)$ is the probability of obtaining the dataset. We know that choosing different values for the parameters in our model does not change $P(D)$, so this is a constant in our fitting problem. We also make the assumption that there is no *a priori* preference for any particular values for the model paramaters, meaning that $P(M)$ is also a constant. Therefore, the task of maximising $P(M|D)$ is equivalent to the task of maximising $P(D|M)$.

Assuming that each datapoint d_i within the dataset D is measured with a random error chosen from a Gaussian distribution, of mean 0 and standard deviation σ_i, then we can find the probability that the datapoint d_i is obtained from a model point of m_i; this is

$$P(d_i|m_i) = \frac{1}{\sigma_i\sqrt{2\pi}} \exp\left(\frac{-(d_i - m_i)^2{}^2}{2\sigma_i}\right).$$

This means that we can equivalently express the conditional probability $P(D|M)$ as

$$P(D|M) = \prod_i P(d_i|m_i).$$

The model described above gives an analytical form for the ISF; this is due to the motion of dynamic scatterers. In order to fully model a spin echo measurement, we also need to add a term, denoted C, that accounts for the signal due to static scatterers [3]. This means that the model to which we are trying to fit our experimental data can be written in the form:

$$M(t) = Af(t) + C,$$

where $f(t)$ is the model given above, paramaterised by τ and λ, and A and C are further parameters (although we are not interested in their values for this exercise).

This means that $P(D|M)$ can be considered to be a function of A, C, and $f(t)$. If we wish to find the $P(D|M)$ as a function of just one of these variables, we need to 'marginalise' (i.e. integrate) over the other variables to obtain $P(D|M)$ as a function of the desired variable only. Therefore

$$P(D|M) = P(f(t)) = \iint P(A, C, f(t))\,\mathrm{d}A\mathrm{d}C,$$

where the integration must be carried out over all possible values of A and C. It would, in principle, be possible to use this equation to find the most likely values of the parameters τ and λ in $f(t)$. However, given that this would be very difficult to achieve computationally, there is an analytic way of marginalising over A and C such that we can obtain an expression for $P(f(t))$. We can then evaluate this expression at many points in model parameter space and find the maximum of the probability distribution, thus finding the most likely values of λ and τ.

This method discussed above will not be reproduced here - it is beyond the scope of this document. However, the code that carries it out is provided for you. If the reader is interested, they may read about the method in [3], or the supplementary material of [1].

TASK

1. Generate data to simulate a spin-echo measurement on an adsorbate-substrate system. In this system, the substrate is a Ru(0001) surface, and the adsorbate may attach to either of the two types of hollow site. The data should contains ISFs along the $[1\bar{1}0]$ and $[11\bar{2}]$ azimuths, for a reasonable range of values of the parallel momentum transfer ΔK.
2. Use the Bayesian fitting method, and the analytic model outlined above, to find the best estimate for the values of the constants λ and τ that correspond to your experimental data
3. Plot a graph in τ - λ space to illustrate your results

The code that uses the marginalised Bayesian fitting method to give the conditional probability $P(D|M)$ is provided below.

Tip 1

Generate the data by using the analytic model provided to model the system for a particular value of each constant τ and λ. Then add white noise (a Gaussian distribution with a fixed standard deviation, and a mean of 0).

Tip 2

Think carefully about the inputs and outputs of the Bayesian fitting function: as it is written, the function takes a set of data points and a set of model points; it returns $P(D|M)$. Given that we want to evaluate the conditional probability $P(D|M)$ for each possible value of λ and τ, our code should:

1. Evaluate the model prediction for the ISF for each value of λ and τ
2. Input this model prediction, along with the experimental data, into the Bayesian fitting function. This will calculate $P(D|M)$ for these particular values of λ and τ
3. Repeat for different values of λ and τ, making sure you cover a reasonable range of λ - τ space
4. Maximise $P(D|M)$ with respect to both λ and τ to find the most likely values of these parameters.

BAYESIAN FITTING FUNCTION

This function evaluates $P(D|M)$ for a particular dataset D, and a particular model M. It is not necessarily important that you follow every step of the code, but make sure you understand what the inputs and outputs are so that you can use the function properly:

```
function [prob_analytical,logprob,prob_exact,A0,C0,flag] = BProb2_data(n1,n2,D,error,fitval,flag1,flag2)
% The function returns the relative probability that a data set D can be fitted by a model function plus a constant background offset. The analytical method for this can be found in Pepijn's thesis
%
% Input arguments:
%
%  D:       The 1xn array (where n is at least n2) containing the n datapoints
%
%  error:   The 1xn array containing the standard deviation of each data point in D
%
%  fitval:  The 1xn array containing the model function evaluated at the
%           same values of DeltaK and t that each datapoint in D was evaluated at
%
%  flag1:   If this is set to 1 then the function will calculate the
%           numerically integrated marginalised function (prob_exact) as well
%           as the analytical one (prob_analytical). Otherwise, the function simply sets
%           prob_exact=prob_analytical
%
%  flag2:   When set to 1, the best fitting value of A is simply assumed to
%           be 1. Use this if you know your experimental data is of the
%           form D = model + C + noise (with no prefactor on model)
%
%  n1:      The first element in the arrays D, error and fit to be used
%  n2:      The last element in these arrays to be used in this fit.
%
%
% Output arguments:

% prob_analytical: The (relative) probability that function given in the array 'fitval' plus an offset constant will fit the data, formed by integrating the probability over (i.e marginalising) all possible values of A and C where attempts are made to fit the data with:  D(i)=A*fitval(i)+C.
%
%'prob_exact': The same as prob_analytical, except worked out by a numerical integration routine (that requires the function prob2.m) only over non negative values of A and C
%

global K0 K1 K2 K11 K12 K22 evecK sdum A0 C0

% nstand is the number of standard deviations away from peak we will numercially integrate.
% A good value is nstand=4

% try setting it as 4
nstand=4;


if (isrow(fitval) && iscolumn(D)) || (isrow(D) && iscolumn(fitval))
    fitval = fitval';
end

[D,fitval,devider] = prep_isf_for_comparison(D,fitval);


% ftol is the fractional tolerance that we wish to integrate numerically to
% A reasonable value is ftol=1e-4
ftol=1e-4;

% create the sums needed for the least squares fit
K0=0.0;
K1=0.0;
K2=0.0;
K11=0.0;
K12=0.0;
K22=0.0;
sdum=0.0;

variance=error(n1:n2).*error(n1:n2);
K0=sum(D(n1:n2).*D(n1:n2)./variance);
K1=sum(fitval(n1:n2).*D(n1:n2)./variance);
K2=sum(D(n1:n2)./variance);
K11=sum(fitval(n1:n2).*fitval(n1:n2)./variance);
```

```
K12=sum(fitval(n1:n2)./variance);
K22=sum(1./variance);
sdum=sum(log(variance));

sdum=sdum+(1+n2-n1)*log(2*pi);

% sum from n1 to n2, so the number of terms is 1+n2-n1
%  the least squares best fit values of A0 and C0
A0=(K22*K1-K12*K2)/(K11*K22-K12^2);
C0=(K11*K2-K12*K1)/(K11*K22-K12^2);
if flag2==1
    A0=1;
    C0=0;
end


% perform the marginalisation - integrate probability of a fit over all (reasonable!) values of A and C
flag=1;

% this piece of code was a rough attempt at eliminating values where the fits went out of the acceptable range and uses 'flag=0' to show  when this is occurring - i.e. the most likely value of A and C, A0 and C0 have one of the negative if(C0 < 0.0)
if(C0/(A0+C0) < -0.1)
    flag=0;
end
if(A0 < 0.0)
    flag=0;
end

S0=K0-2*A0*K1-2*C0*K2+A0*A0*K11+2*A0*C0*K12+C0*C0*K22+sdum;
tmp = K11*K22-K12^2; if tmp<=0, tmp=eps; end
logprob=-S0/2+log(2*pi)-0.5*log(tmp);

%prob(ialpha)=exp(max(-200,logprob(ialpha)));
prob_analytical=exp(logprob);
if(flag < 1)
   prob_analytical=0.0;
end
if(flag1==1)

% Calculate the numerical integrated value. Start by working out  eigenvectors (give the symmetry directions of the pdf in A and C space) and values of the K matrix
KMat(1,1)=K11;
KMat(2,1)=K12;
KMat(1,2)=K12;
KMat(2,2)=K22;
evalK=eig(KMat);
[evecK,Dum] = eig(KMat);

% prob=exp(-S/2)  where
% S=S0+evalK(1).(evec_coord1)^2+evalK(2).(evec_coord2)^2
% eigenvalues are 1/(stdev)^2 in each direction
evc1range=1/sqrt(evalK(1))*nstand;
evc2range=1/sqrt(evalK(2))*nstand;

% Integrate exp(-0.5*(S0+(A-A0,C-C0)Kmat(A-A0,C-C0)T)) but in eigenvector coordinates over nstand standard deviations in
```

```
    each eigen vector %direction - prob2 works out internally A
    and C, and then sets the integrand to zero if A<0 or if C<0
tol=ftol*exp(logprob);

% use the previous analytical estimate of the probability so as
    to work out an absolute tolerance (tol) as needed by the
    integration routine as opposed to the relative (or
    fractional) tolerance (ftol) that it is more useful to think
     in terms of.
prob_exact = dblquad(@prob2,-evc1range,evc1range,-evc2range,
    evc2range,tol);
else

% if flag1 is not set to 1 we do not numerically integrate, but
    just set the exact probability to be equal to the analytical
     one.
    prob_exact=prob_analytical;
end
end

function z = prob2( ec1,ec2 )
        % ec1 and ec2 are coordinates in eigen vector space
        global K0  K1 K2 K11 K12 K22 evecK sdum A0 C0

        A=evecK(1,1)*ec1+evecK(1,2)*ec2+A0;
        C=evecK(2,1)*ec1+evecK(2,2)*ec2+C0;
        z=(abs(A)+A)./A.*(abs(C)+C)./C/4.*exp(-(K0-2*A*K1-2*C*K2
            +A.*A*K11+2*A.*C*K12+C.*C*K22+sdum)/2);
end

function [isfA,isfB,devider] = prep_isf_for_comparison(isfA,isfB
    )
        max_A = max(isfA);
        max_B = max(isfB);
        if max_A > 1 || max_B > 1
            max_of_all = max(max_A,max_B);
            isfA = isfA/max_of_all;
            isfB = isfB/max_of_all;
        else
            max_of_all = 1;
        end

        devider = max_of_all;
end
```

REFERENCES

1. B. A. J. Lechner, P. R. Kole, H. Hedgeland, A. P. Jardine, W. Allison, B. J. Hinch and J. Ellis, *Phys. Rev. B* 89(12),121405
2. F. E. Tuddenham, H. Hedgeland, A. P. Jardine, B. A. J. Lechner, B. J. Hinch, and W. Allison, *Surf. Sci.* 604, 1459 (2010).
3. Pepjin Kole, *PhD Thesis*.