Advanced Neutron Reflectometry Data Analysis for Soft Matter Systems

Introduction

During the following practical you will fit layer models to experimental reflectivity data sets and use these fits to gain quantitative information about the molecular structure of material across a buried interface. To do this you will use the **RasCal 2 software** which calculates slab layer models using Abeles matrix formalism, which is conceptually very similar to Parratt's recursive formalism which is also used in reflectometry data analysis.

In these approaches, each layer is described by its thickness, roughness and scattering length density (SLD, ρ). The thickness of each layer is used to determine the phase of the reflected waves from each "slab" layer interface (top and bottom). Kiessig fringes in the reflectivity data are due to interference between these reflected waves.

More information on the mathematics of Parratt's recursive formalism is given at the end of this document.

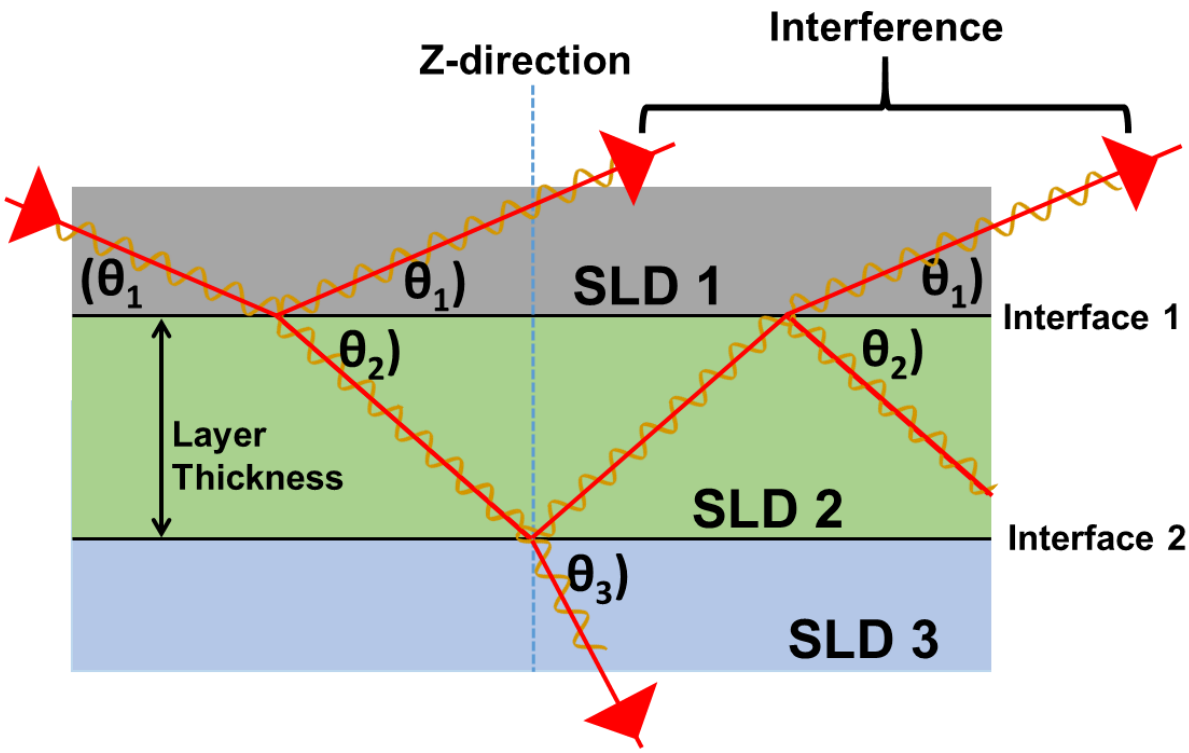


Figure 1, The "slab" layer model is used to calculate model reflectivity data by both the Abele's matrix formalism and Parratt's recursive formalism. The difference in scattering length density (ρ) at an interface and

the angle/wavelength of the neutron beam give the magnitude of reflection while thickness of the layer and its effect on the phase difference between the top and bottom reflections gives rise to Kiessig Fringes.

In the following practical you will start by examining the structure of a silicon-D_2O interface using the Rascals model building GUI followed by fitting the same data using a "custom" i.e. ***scripted modelling capabilities***. Then, the structure of a bilayer of 1,2-dipalmitoylphosphatidylcholine (DMPC) will be examined using multiple solution isotopic contrasts firstly with a simple "volume" fraction model followed by analysis using a more advanced "area per molecule" model. The data you use for this will be the reflectivity data obtained from the ISIS neutron training course solid-liquid flow cells practical on the OFFSPEC reflectometer. Finally, time permitting, we will take a look at a complex layer structure with a composite fringe pattern from a multi-layered membrane complex.

Part 1: *The Silicon-Water Interface*

The silicon water interface is a common surface used in NR studies. For these experiments, the reflection occurs within a silicon substrate.

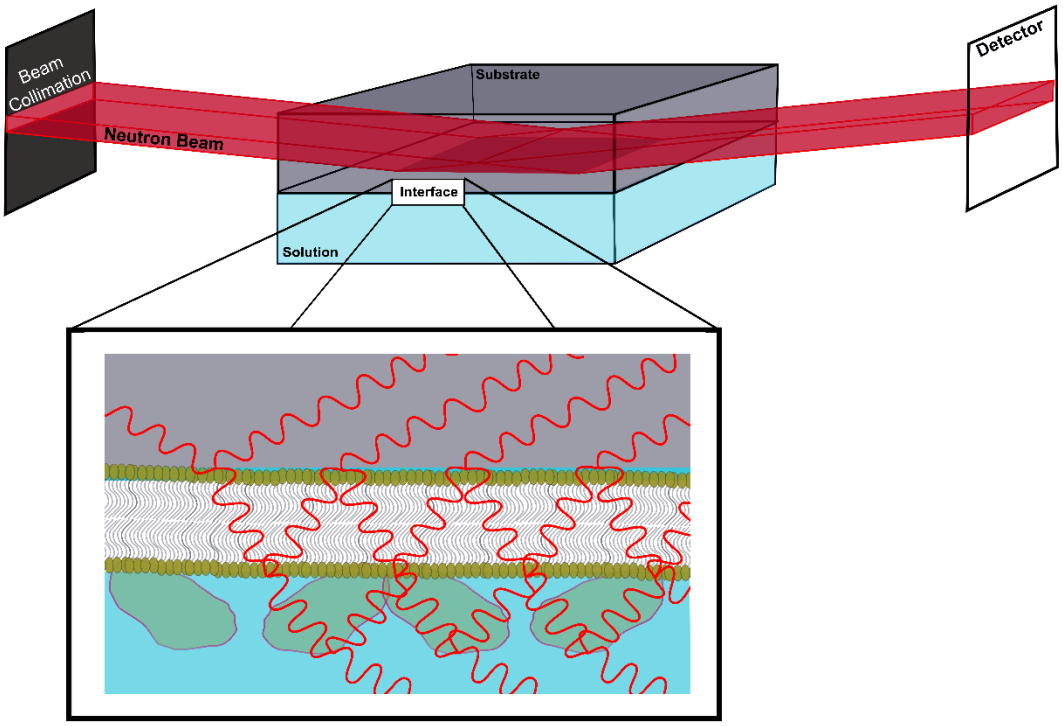


Figure 2, Neutron Reflectometry in a solid-liquid flow cell. Note: the neutron beam is reflected inside of the substrate.

You will begin the practical by fitting a bare SiO_2 coated silicon-D_2O interface using Rascals GUI option.

Moving the Practical to the IDAaaS Desktop

If you haven't already. Move the school folder onto your IDAaaS desktop so you can save your progress in the practical. To do this you must use the following steps:

Applications>System>File manager

In file manager navigate to:

/mnt/ceph/auxiliary/reflectometry/Virtual Reflectometry School 2026/

Copy the **Rascal 2 Practical** folder, and paste it to:

/home/username/Desktop/

HINT: Username is your **individual username**, mine is lc10795

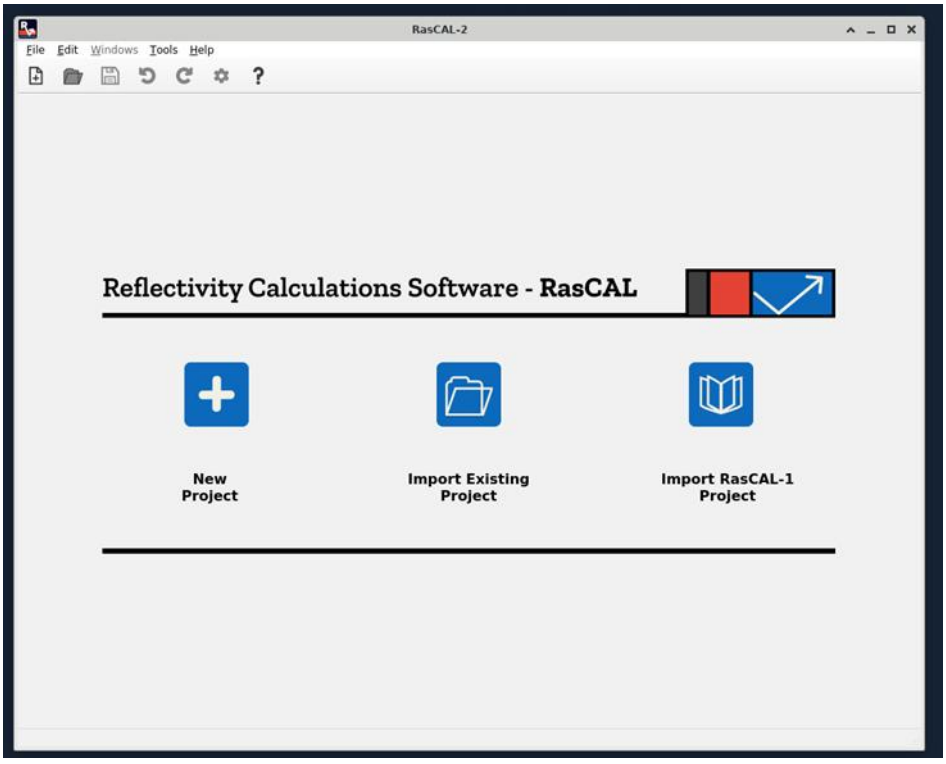
Alternatively hit the **home** button (top right) and navigate to desktop from there.

Starting the Software.

In your IDAaaS session:

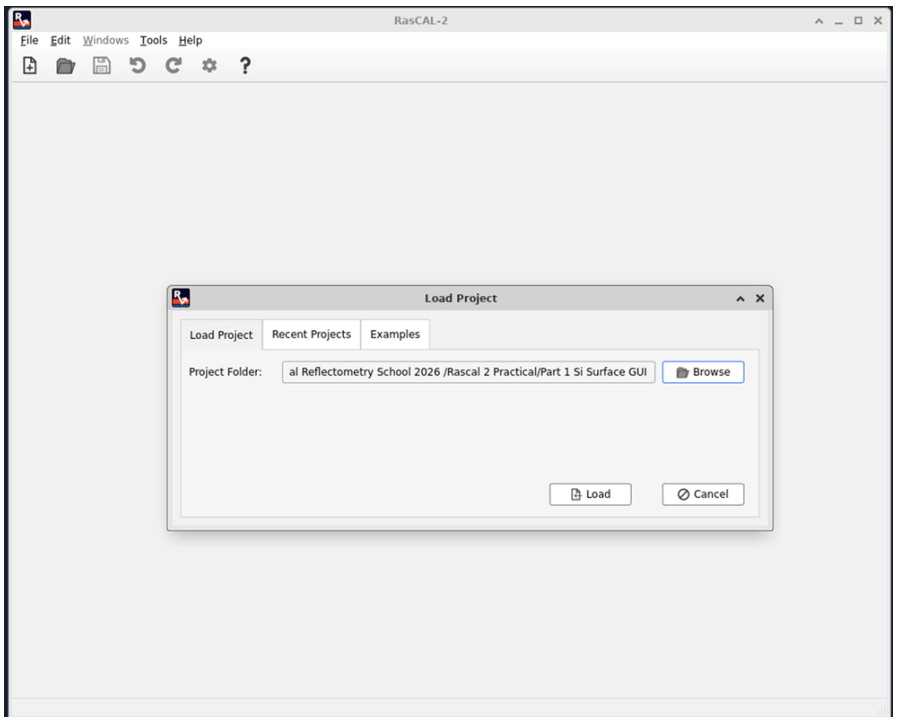
Applications>Software>RasCal 2 nightly

The **RasCal 2** software will load and you will get the following screen:



- Click on **Import Existing Project**

Browse to the folder named **Rascal 2 Practical Student/ Part 1 Si_D2O Interface** and select the folder named **Part 1 Su Surface GUI** and select open. The select **Load** (see below)

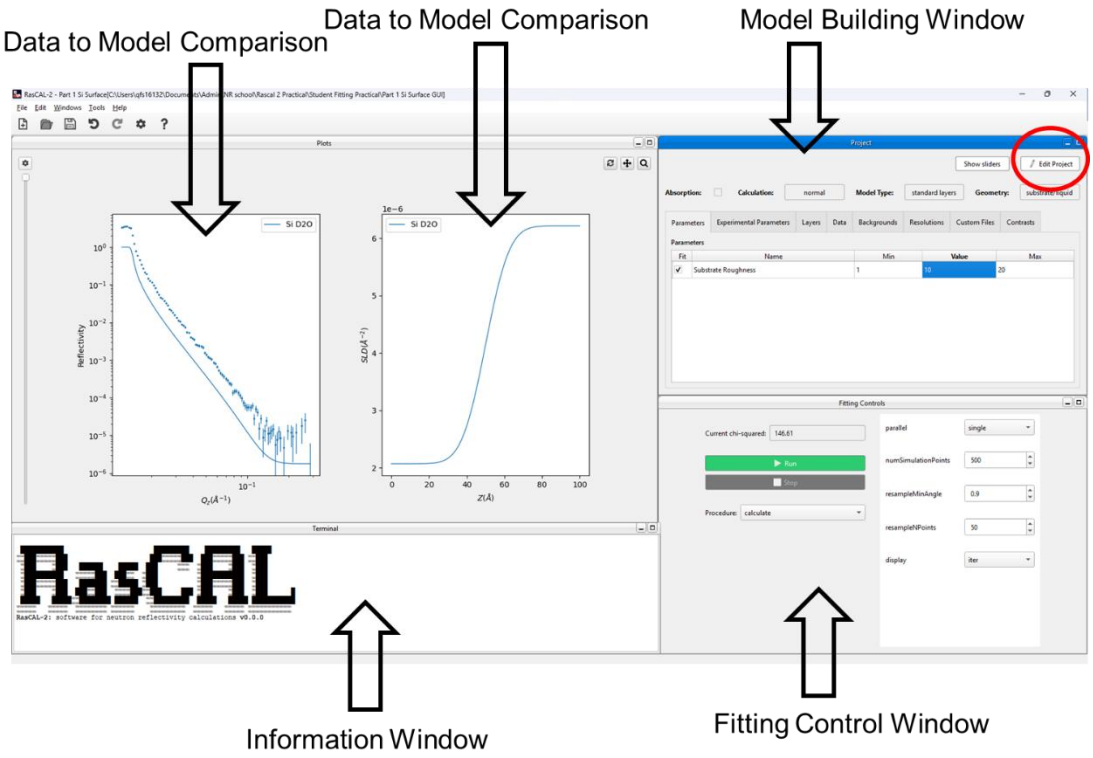


- The **RasCal 2** software will load.

  Rearrange the windows in the software as you see fit (see below)

Setting up the model to fit

Once the Rascal 2 project has loaded you will see the following:



The 1st thing you will notice is that the real and model reflectivity data is not on the same scale, i.e. the scale factor is incorrect. There are similarities and differences between the model data (**line**) and the experimental reflectivity data (**error bars**). Specifically, the critical edge of the reflection is in the same position in both and while the general decay of the model data intensity against momentum transfer (Q_z) and the background do not match the experimental data.

Therefore, we begin fitting by setting the correct experimental parameters.

1. On the model building window click on the tab labelled **experimental parameters**.

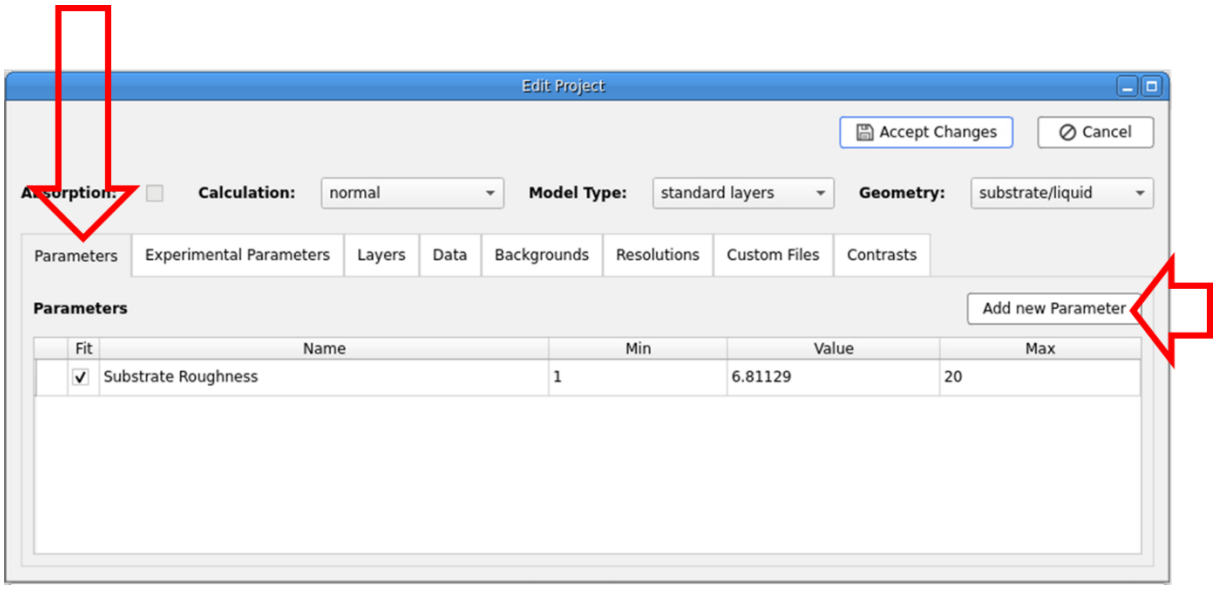
2. Next correctly scale the data using the **scale factor** (do this so the model and data critical edges meet) and then set the **background** for the sample (the flat region in the high Q_z regime). If you want you can fit these parameters by ticking the fit box against each parameter, selecting **simplex** in **Fittings Controls** and clicking on "**Run**".

3. Once these are set fit the substrate roughness (the roughness between the bulk interfaces) to gain an approximate fit of the data. You will notice you get a good fit to the experimental reflectivity data producing a reflectivity profile with a defined step function.

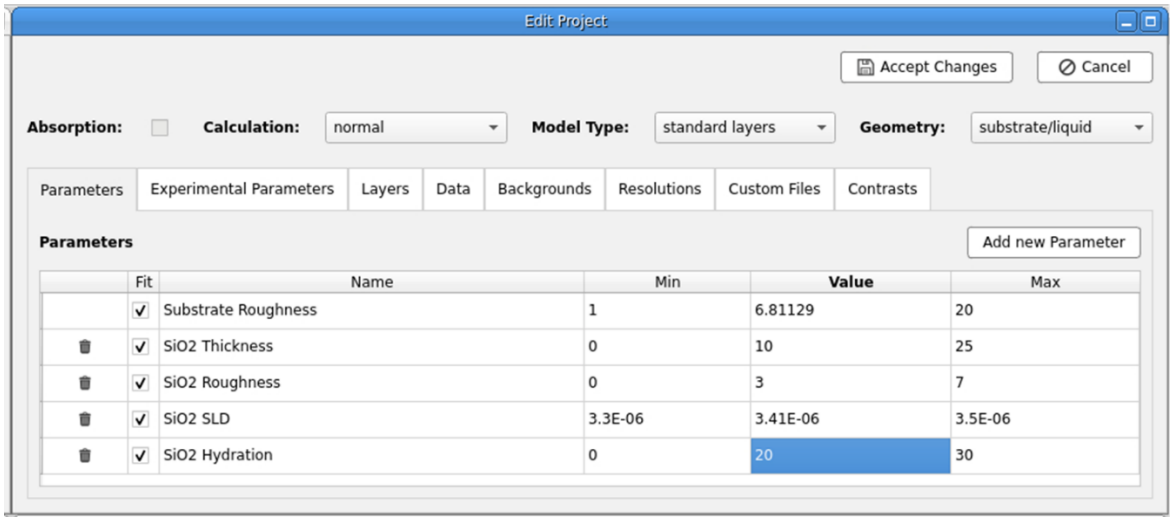
4. However, the roughness will be artificially high as the model does not accurately portray the interfacial structure. The silicon substrate will have a thin (~10 Å) silicon dioxide layer on the surface. We will now add this layer by editing the model.

Adding an interfacial layer.

1. Click **Edit Project** on the **Project window** and you will now be able to edit the model window:



2 Click on the parameters tab and add the following four parameters with the following bounds:



Place the fitted value between the lower and upper bounds for each parameter except the SiO_2 SLD where you should use the value given above.

When you run the fit do not fit the SiO2 SLD value (un-tick the box for this).

Appropriate range for fitting for the SiO2 layer are:

SiO2 Thickness: Lower = 0 Å, Upper = 20 Å

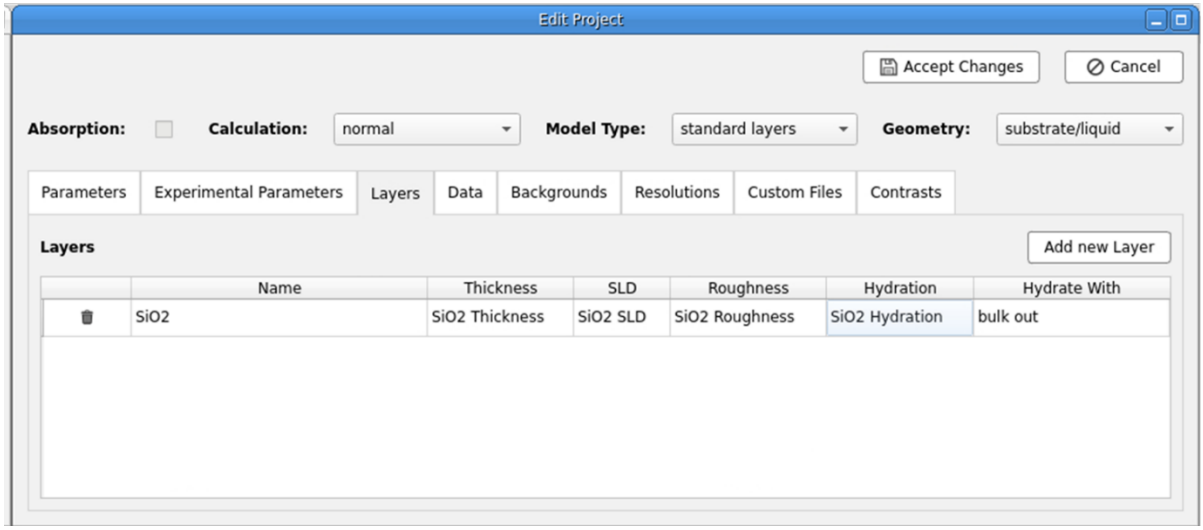
SiO2 Roughness: Lower = 0 Å, Upper = 7

SiO2 SLD: Set to 3.41e-6 $Å^{-2}$ and **do not fit** (untick)

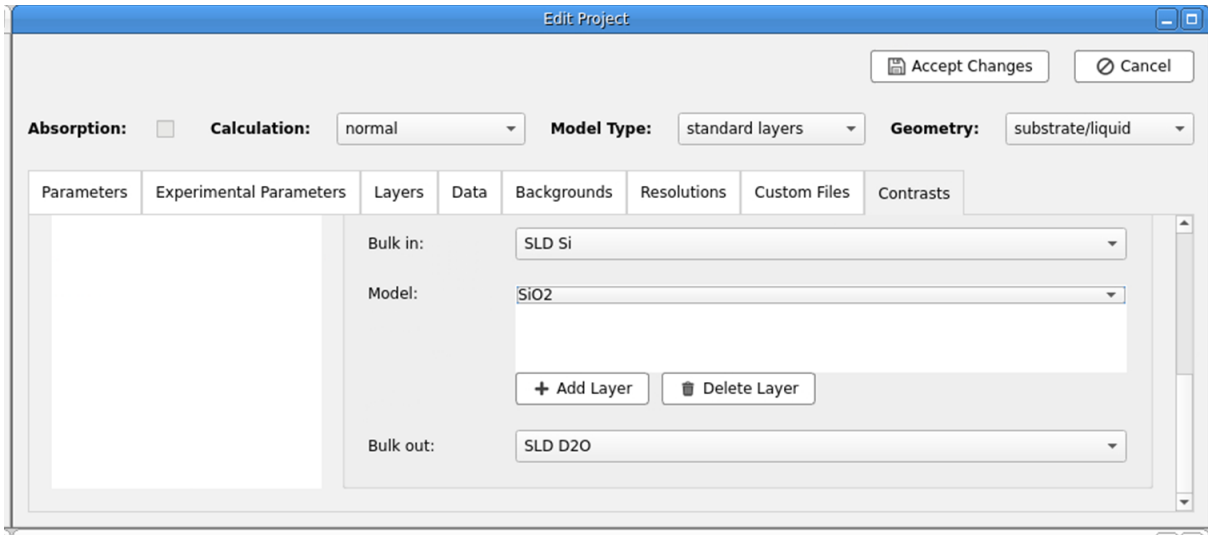
SiO2 Hydration: Lower = 0%, Upper = 30%

HINT!: The value must be set between these bounds before you start the fit.

3. Next on the "**Layers**" tab **Add New Layer** and then populate the layer with each parameter in the correct place and naming the layer appropriately:



4. In the "**Contrasts**" tab select the only contrast (labelled Si D2O) and at the bottom of the tab add your layer to the model section between the **Bulk in** and **Bulk out**:



Now click **Accept Changes** and the model should be updated with your new layer. Now rerun the fit.

5. You should now see that you have a new layer between the bulk phases which is rather ambiguous i.e. poorly described by the experimental data.

Think about why this is ambiguous and what might help to better resolve this layer? Write you answer thoughts in the box below:

6. Place the fit values in the table overleaf:

Substrate roughness/Å	
SiO2 thickness/Å	
SiO2 roughness/Å	
SiO2 Hydration/%	
Background 1	
Scale factor 1	

Now save the project using a name of your choice on the IDAaaS desktop by:

Creating a new project in the Rascal 2 practical folder on your desktop.

File>save to folder> select folder > open > save

Part 2: Custom Model Fitting

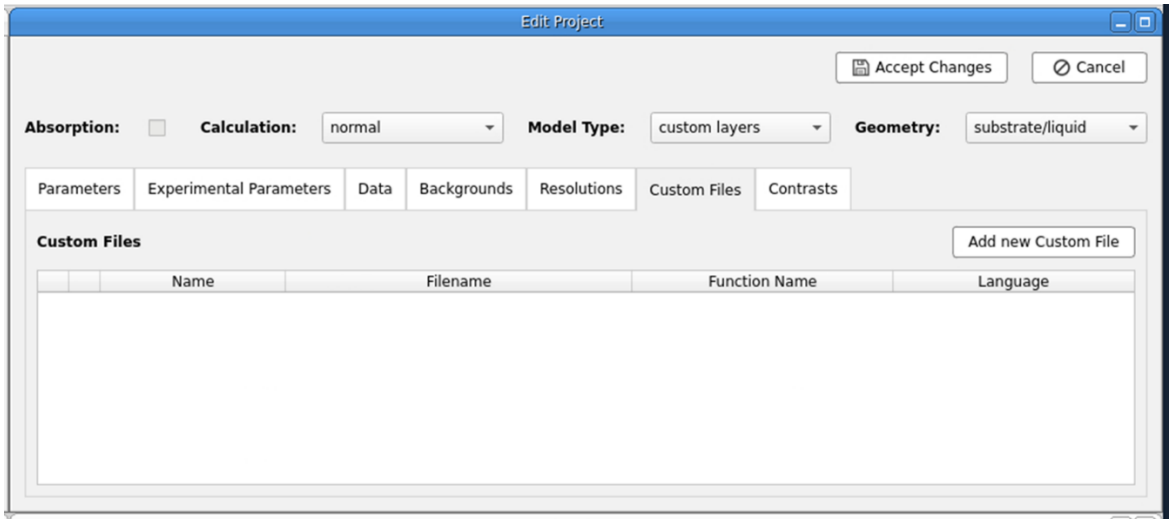
Using RasCals GUI is an easy way to begin fitting NR data but the real flexibility in the software comes from using the custom model approach where the fit is defined using a script (i.e. a short piece of code).

We will start our journey into scripted modelling by fitting a predefined Custom model of the Silicon-D_2O interface we have already fitted.

1. Edit the project and delete the parameter for the **SiO2 SLD** from the parameters table.

2. On the **Model Type:** change from **standard layers** to **custom layers**.

3. You will note a new tab in the model called **Custom files** has appeared. Navigate to this tab.



4. Click on **Add new Custom File**. Inside the current project folder you will find a python script called **custom_model.py**. Select this.

5. Change the name in the Custom Files list to **Custom_model**.

6. Click on **Edit File**.

A script editor window will appear showing you the code which makes up the custom model. The script is a sequential description of individual parameters, how these parameters relate to the interfacial layers and how those layers are structured between the bulk phases.

The simple script for the silicon water interface is given below with an explanation of its structure:

```
import numpy as np

def custom_model(params, bulk_in, bulk_out, contrast):

    sub_rough = params[0]
    oxide_thick = params[1]
    oxide_roughness = params[2]
    oxide_hydration = params[3]

    # We have a constant SLD for the bilayer
    oxide_SLD = 3.41e-6
    DMPC_Tails_SLD = -0.39e-6
    DMPC_Heads_SLD = 1.98e-6


    # Make the layers
    oxide = [oxide_thick, oxide_SLD, oxide_roughness, oxide_hydration, 2]

    # Match the layer stack to the contrast

    match contrast:
        case 1 | 2:
            output = np.array([oxide])
        case 3:
            output = np.array([oxide])
        case 4:
            output = np.array([oxide])
        case 5:
            output = np.array([oxide])

    return output, sub_rough
```

1, Name of function which defines the model.

2, The list of parameters you want to fit (numbered starting from 1). These should match the parameter list in RasCAL.

3, Known parameters to use in the fitting.

4, Layer arrays which contain: Layer_thickness, Layer_nSLD, Roughness, Hydration, Hydrade_with_what (2=bulk)

5, The stack of layers and how that relates to each contrast in the contrasts list (numbered starting from 1)

3. Click **Save** on the script go back into Rascal.

4. On the **Contrasts** Tab select your only contrast (**Si D2O**) and on Model: select **custom model.**

Note : If you can't see the model function listed change **model type** from **custom layers** to **standard layers** and back again.

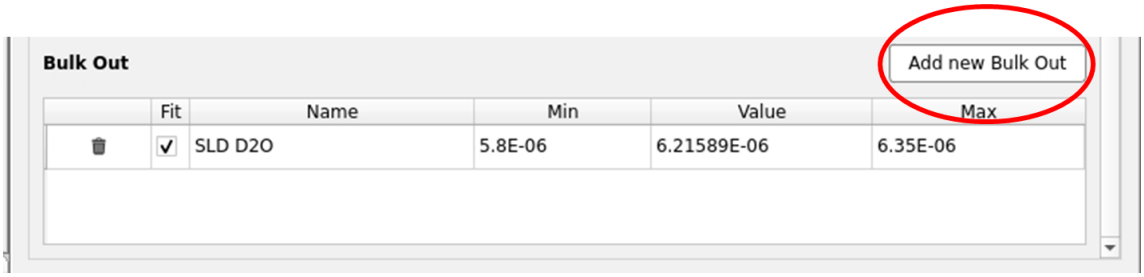
5. In the **Parameters** Tab delete the parameters which do not appear in the model.

6. Accept changes

7. Refit the data. You will note that the fit should be fairly identical to what we have found previously.

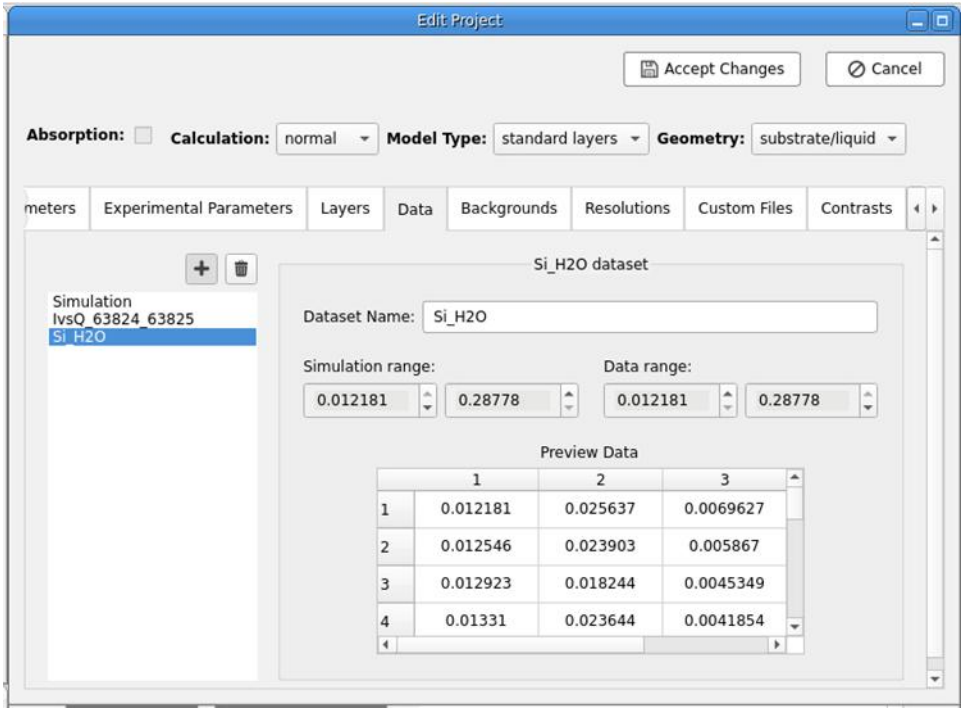
8. We will now add an additional contrast to enable a more precise resolution of the surface structure.

9. **Edit Project**>**Experimental Parameters Tab**>Navigate to the bottom>**Add new Bulk Out**



10. Name this new solvent **SLD H2O** and set a **Min** value to **-0.6e-6**, a Value to **-0.56e-6** and a **Max** to **-0.4e-6**.

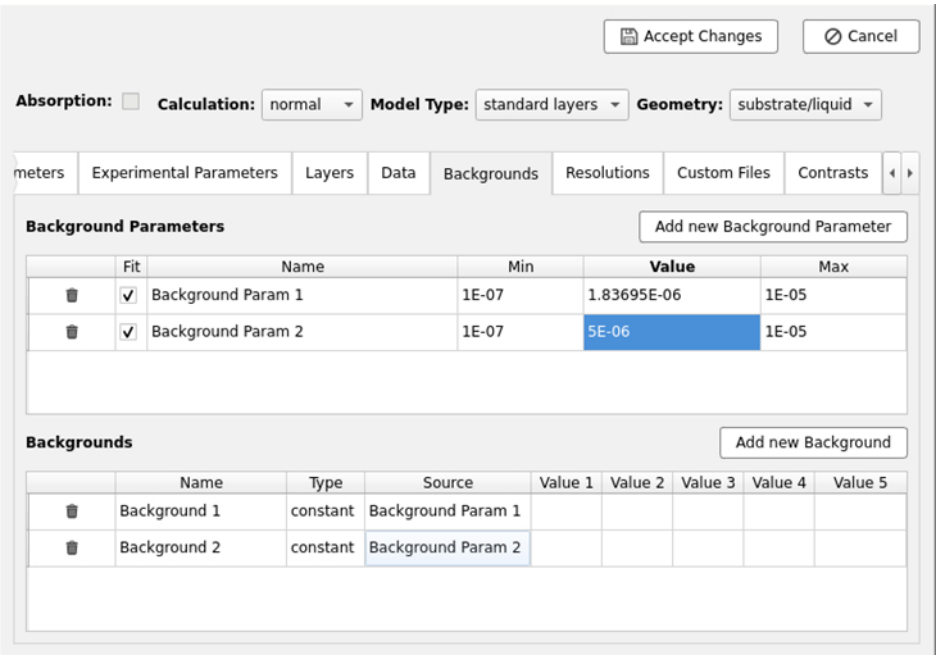
11. Click on the Data tab and add (+) an additional data set. Navigate to the folder called "DMPC data named" and select the data set "Si_H2O.dat" and select open.



12. Now navigate to the **Contrasts** tab and add a new contrast (+).

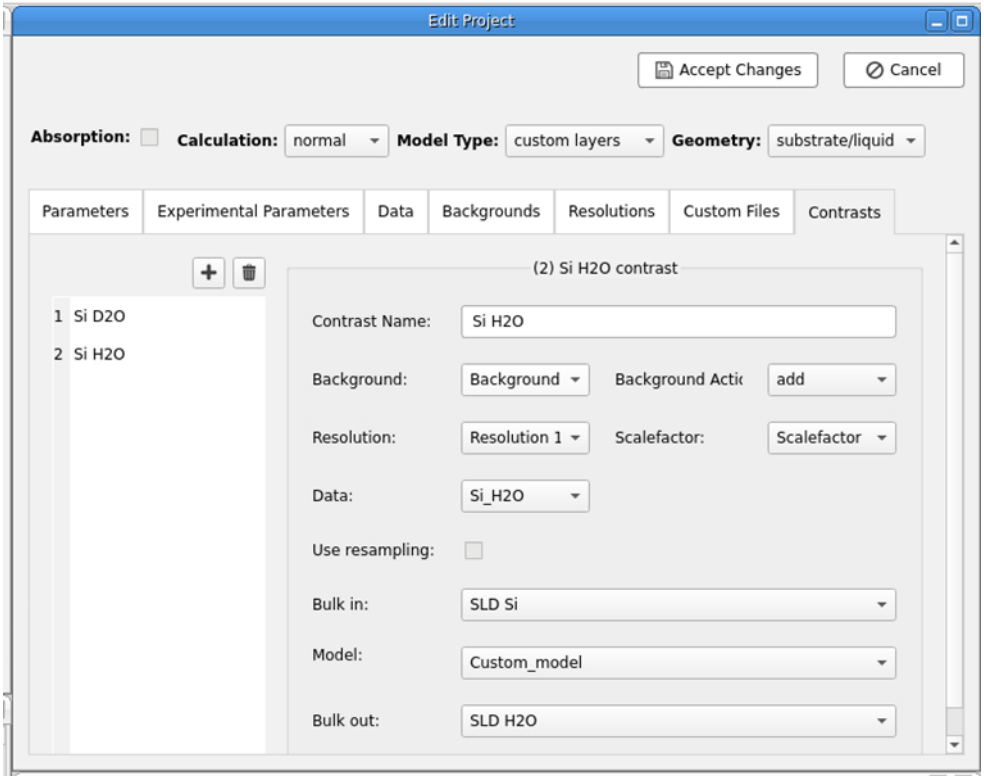
13. Name that contrast **Si H2O**. You will now need to select the experimental parameters for this background. The incoherent scattering of H2O is significant higher than that of D2O due to the high incoherent scattering length of protium compared to deuterium. Therefore we expect the background in H2O to be higher than in D2O and therefore need a dedicated parameter for this for the H2O contrast.

13. Go to the **Backgrounds** tab. **Add** a **new Background Parameter** called **"Background Param 2"** then **Add** a **new Background** called "**Background 2**", select **Background Param 2** as the Source for this.



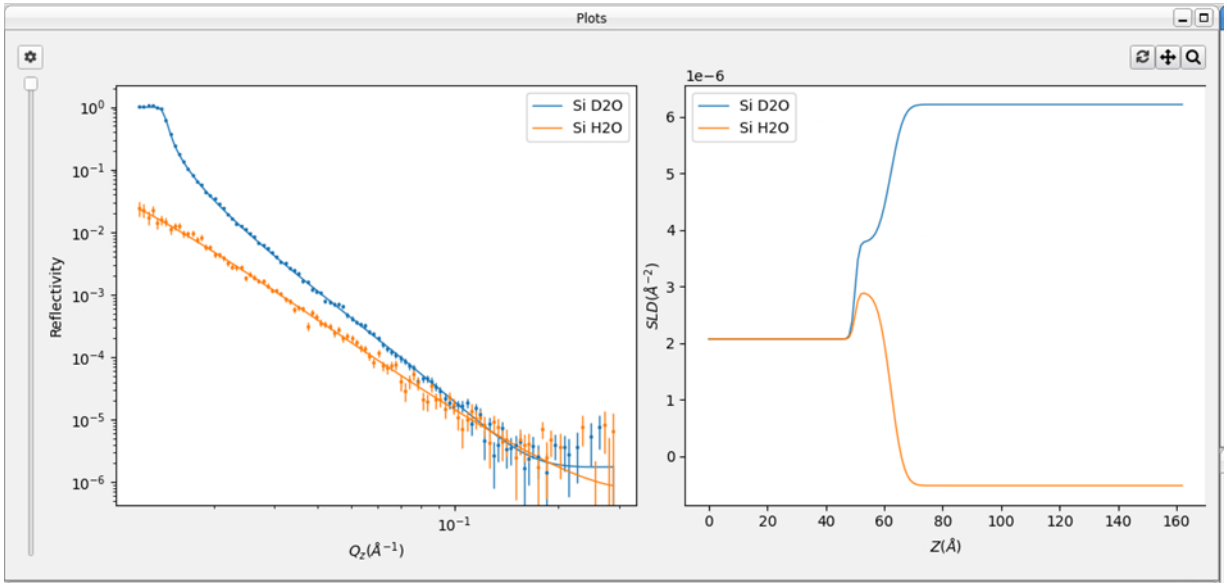
15. Go back to the **contrasts** Tab and select the experimental parameters for the Si H2O contrast.

Hint: These will be the same as for the **"Si D2O contrast"** except the **Data** and the **Background**.



16. - Set the **Bulk in** to be **SLD Si** and the **Bulk out** to be the **SLD** of **H2O.** The model is the custom model as with the Si D2O contrast. Accept changes.

17. You should now see two contrasts in Plots windows, D2O and H2O solution contrasts, constrained to have the same structure across the Si/Solution interface. Run a fit to see if how the constraint of a second contrast changes in the fitted parameters:



You have now set up a custom model and simultaneously fitted multiple reflectivity data sets under differing solution isotopic contrasts to accurately resolve an interfacial structure. Next we will take this further!

18. Save the Project at this point on the IDAaaS desktop twice. Firstly:

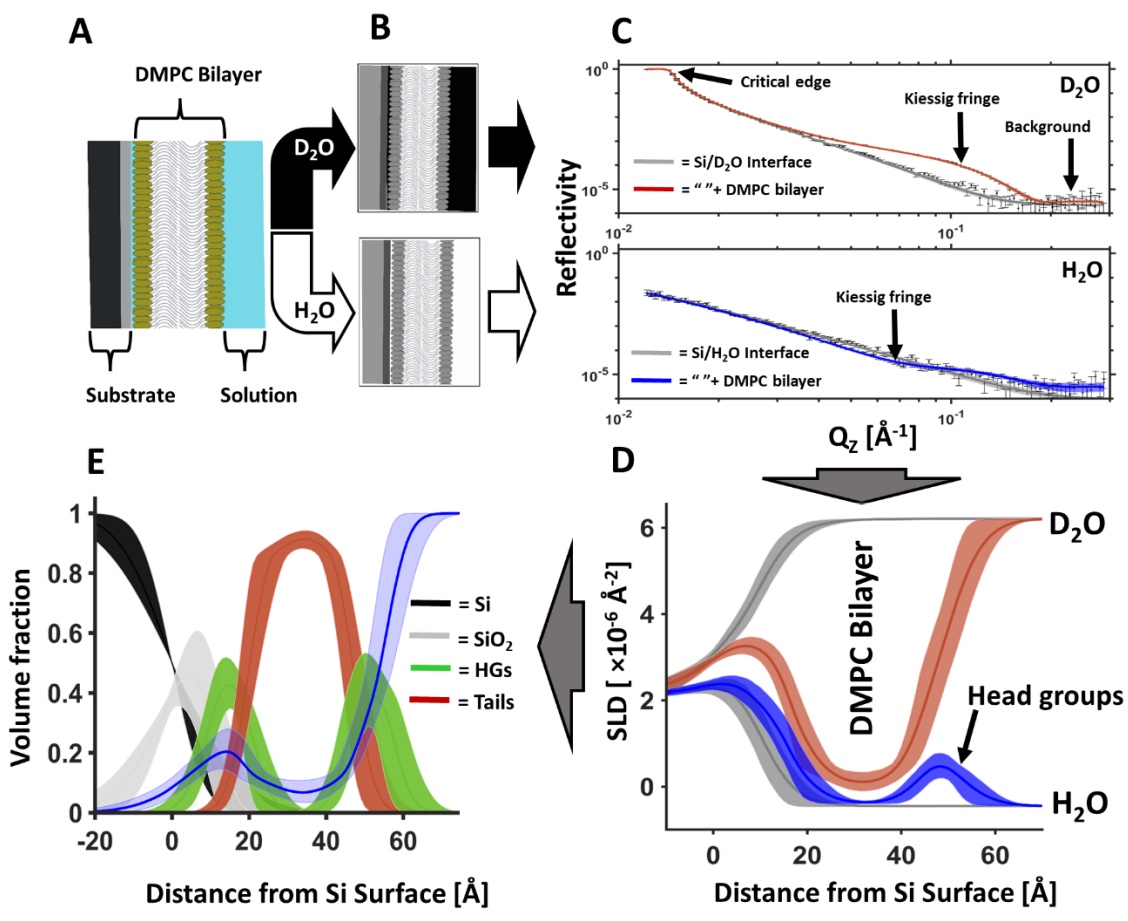
Save To Folder > new folder>“completed part 2”>open>save

Secondly save again in a new folder called something like “Part 3 Custom DMPC Volume Fraction” and keep this open.

Part 3: Fitting a Custom Model of the Bilayer by Volume Fraction

We will now create a custom model of the interfacial structure coated with a 1,2-dimyristoyl-sn-glycero-3-phosphocholine (DMPC) bilayer and fit the experimental reflectometry data. Starting with the project you have created in part two, using the skills you've learned you will create a custom model to resolve the structure of the supported lipid bilayer.

Below is an example of the encoding and decoding of the structure of a DMPC bilayer at the solid-liquid interface by neutron reflectometry. Two solution isotopic contrasts (D_2O; red, and H_2O; blue, **C**) are used to resolve the scattering length density (SLD) profiles (**D**) across the surface. The resolved structural parameters are ultimately used to resolve the component volume fraction vs. distance profile across the surface:



The simultaneous fitting of these same two solution isotopic contrast data sets (H_2O and D_2O) was used to resolve the interfacial lipid bilayer structure here, which will be constrained against the data for the Si Interface only. You will be doing the same thing in RasCal 2 but with more data!

To test your fitting skills

1. Let's start by setting up the DMPC contrasts and fitting parameters.
2. **Edit the project** and in the **Data** tab add for the DMPC supported lipid bilayer. This is found in the "**DMPC data named**" folder as "**Si_DMPC_D2O.dat**","**Si_DMPC_SMW.dat**" and "**Si_DMPC_H2O.dat**".
3. You will note that there are three data sets here under three solution isotopic contrast conditions. D_2O and H_2O are the same as for our previously fitted Si surface contrasts but we have an additional contrast called SMW. This is silicon matched water (38% D_2O and 62% H_2O). This additional contrast provides additional constraint for our model fitting.
4. Add a background for the SMW contrast (like you did previously for the H2O contrast).
5. Add a new bulk out for the SMW contrast with a value of 2.07e-6 $Å^{-2}$. Set a min and a max around this.
6. Next we want to start building a model of the DMPC interfacial structure.

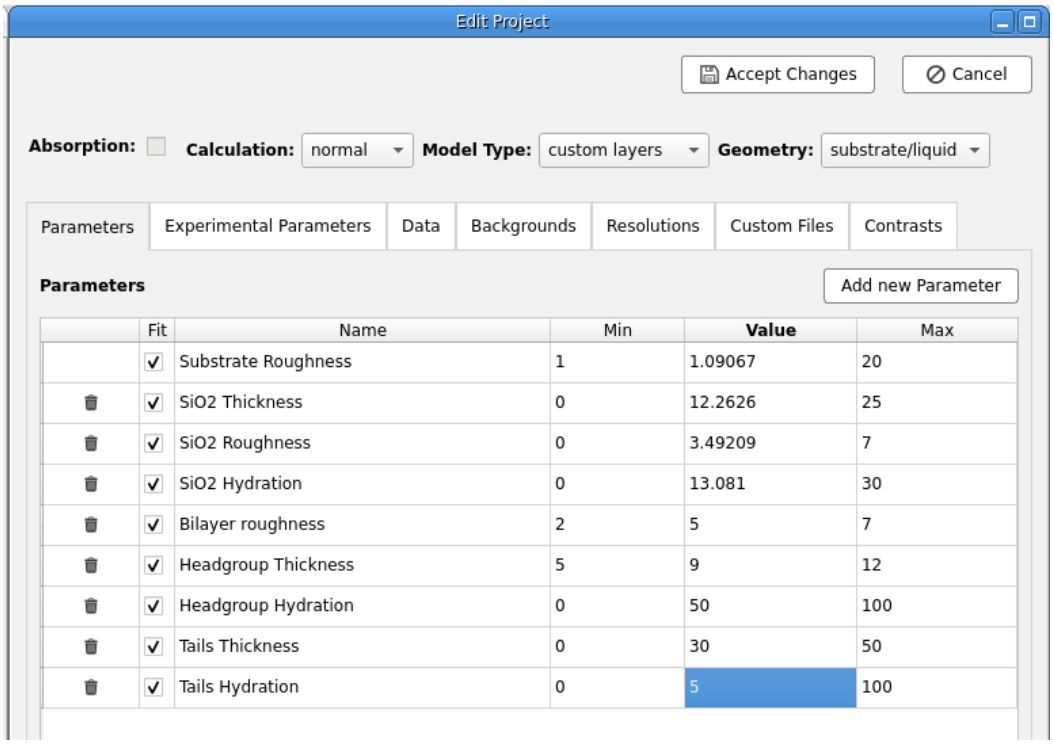
7. In do this we identify our **"knowns"** and **"unknowns"**. We know the scattering length densities of the DMPC headgroups and tails:

   Tails = $\mathbf{-0.39 \times 10^{-6}\ Å^{-2}}$

   Head groups = $\mathbf{1.98 \times 10^{-6}\ Å^{-2}}$

   But this only takes into account a situation where the layer is composed of a single component only. The DMPC bilayer, like the SiO_2 layer is immersed in solution and therefore hydrating water is present in the head groups and potentially in the tails if the bilayer has defects, therefore we should fit two hydration parameters (heads and tails) to experimentally resolve these unknowns.

   Additionally we should fit the **tails thickness**, **headgroup thickness**, and a single **bilayer roughness** parameter (to account for undulations across the bilayer). Add these to the **Parameters** tab using the bounds shown below.



8. Now we need to edit the custom model to add the DMPC bilayer structure. Go to the **General tab** and click **Edit**. Now edit the custom model script to add the new unknown parameters and the known parameters. Define layers for the headgroups and tails of the bilayer and create two new contrast cases both with silicon oxide layers but additionally with a bilayer layers across the interface as well (**Headgroup; Tails; Headgroup**).

You will be given **30 minutes to complete this task**!

HINT!: How to construct your bilayer parameters.

To complete this assignment you need to describe a structure across the interface for three data sets from a DMPC coated silicon surface which is composed of the following layers:

([oxide, Head, Tail, Head])

Below is an example of this done with one <u>Headgroup and SiO2 layers only as a guide:</u>

```
def custom_model(params, bulk_in, bulk_out, contrast):
# Fitted parameters i.e. unknowns
sub_rough = params[0]
oxide_thick = params[1]
oxide_roughness = params[2]
oxide_hydration = params[3]
```

```
Bilayer_roughness = params[4]

Headgroup_thickness = params[5]

Headgroup_hydration = params [6]

Tails_thickness = params [7]

Tails_hydration = params [8]


# knowns

oxide_SLD = 3.41e-6

DMPC_Heads_SLD = 1.98e-6


# Make the layers

oxide = [oxide_thick, oxide_SLD, oxide_roughness, oxide_hydration, 2]

Heads    =    [Headgroup_thickness,    DMPC_Heads_SLD,    Bilayer_roughness,
Headgroup_hydration, 2]


# Match the layer stack to the contrast list in RasCal 2

match contrast:

    case 1 | 2:

       output = np.array([oxide])

    case 3:

        output = np.array([oxide, Heads])

    case 4:

        output = np.array([oxide, Heads])

    case 5:

        output = np.array([oxide, Heads])


return output, sub_rough
```

BEWARE!: Parameter names in the RasCal custom model cannot have spaces between words while in the RasCal GUI parameter list they can.

8. Once you have added the new data sets and updated the custom model compile your new script by saving the script and clicking **Accept changes** in the **project** window. It will not compile if there are errors.

9. You should now see three additional NR data sets appear in the Plots window. Run a simplex fit to resolve the structure of the DMPC coated silicon surface

Now save the project on the desktop on IDAaaS using a name of your choice.

Save your project as a new project on the IDAaaS desktop.

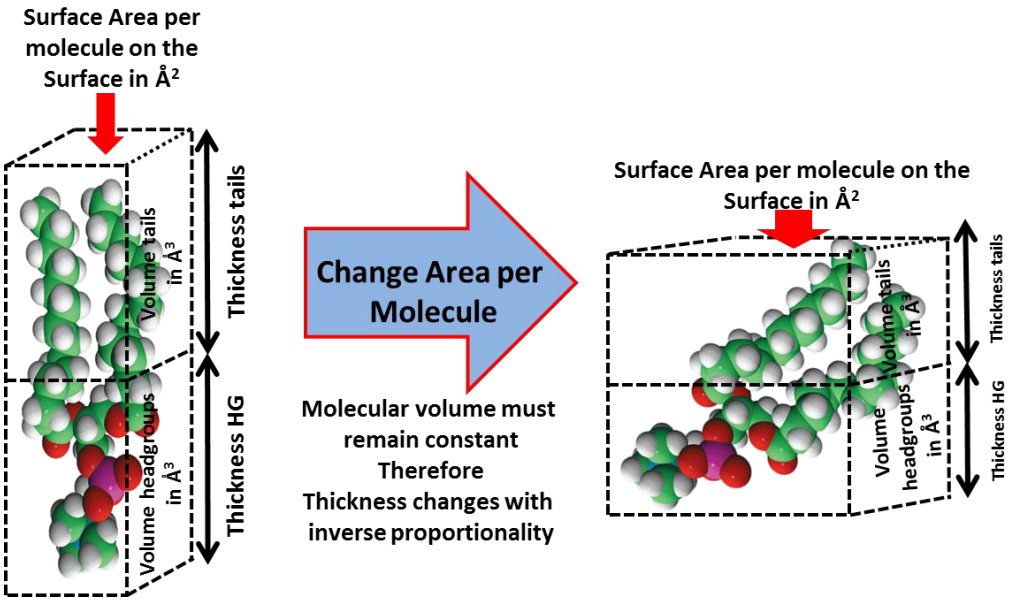
Part 4. Fitting a Custom Model of the Bilayer by Area per Molecule

In the previous section different layers arising from the same molecular species are fitted in an unconstrained manner. This can cause some ambiguity in the analysis as there can be differing total molar amounts of head and tail components in the resolved structural data.

A better approach is to couple these parameters together using a known shared parameter. In most cases this will be **area of the surface occupied by the molecule** (i.e. Area Per Molecule). As each lipid head and tail, being a part of the same molecule, will occupy the same 2-dimentional area of the sample surface. This area will be related to the thickness of the sample by:

$$Occupied\ Surface\ Area\ = \frac{Component\ Volume}{Layer\ Thickness}$$

The volume of the tail and head groups are linked to the sum of the atomic volumes which make up the molecule and therefore will not change (within a given phase). The thickness is inversely correlated to the area per molecule and therefore only a single parameter for both the head and tail groups needs to be fitted, that is the area per molecule at the surface. Below is a diagrammatic representation of this concept:



In the example we will use the molecular volumes of the head and tail components of DMPC to construct a model which fits the bilayer at the silicon water interface by four parameters only. That is, the **area per molecule**, the **total bilayer hydration** (i.e. presence of defects across the surface) and the **headgroup hydration** (i.e. water bound to the hydrophilic headgroups) and **bilayer roughness**.

For this section of the practical you will edit a **custom area per molecule model**, completing parts of the code which are missing.

Starting the project

In Rascal 2 load the project: **Load>Browse>Part 4 Scripted APM bilayer>Select Folder >Load**

You will see five sets of data, Contrasts 1, 2 describe the SiO_2 layer at the interface only, while 3, 4 and 5 describe the same surface coated with a DMPC supported lipid bilayer in D2O, SMW and H2O solution respectively. Currently however, the model does not describe the DMPC bilayer structure. You will need to add this!

The relationship between layer thickness, component volume and area per molecule is:

$$Thickness = \frac{Volume}{Area\ Per\ Moelcule}$$

And component volume, scattering length and scattering length density is:

$$SLD\ (\rho) = \frac{\sum b}{Volume}$$

The volume and the scattering lengths (∑b) of each of the interfacial components are known. By fitting the experimental reflectivity data we can determine the interfacial lipid area per molecule and layer thicknesses.

The SLD of the lipid headgroups and tails can be calculated using priori information, using this, the solution is present in each of the headgroup and tail regions of the bilayer can be determined.

There is a differential in solution content within the headgroups and tails. As the tails will only have water content through defects while the headgroups have hydration waters as well as water present due to defects. The model built during this part of the exercise will discriminate between these hydration types.

Edit the model and then edit the custom model. **Edit project>Custom Files** tab**>Edit File**.

You will find the following incomplete code to describe the interfacial layer structure:

```
import numpy as np

def custom_bilayer_DMPC(params, bulk_in, bulk_out, contrast):

    ## Parameters to Fit
    sub_rough = params[0]
    oxide_thick = params[1]
    oxide_hydration = params[2]
    HG_bound_waters = params[3]
    #DMPC_APM = params[4]
    #Bilayer_roughness = params[5]
    #Bilayer_hydration = params[6]

    ## We have a constant SLD for the bilayer
    oxide_SLD = 3.41e-6

    ## Known Volume in Angstrom cubed
    DMPC_HG_Volume = 320.9
    DMPC_Tails_Volume = 783.3
    Water_Volume = 30.4

    ##Known Scattering Lengths
    DMPC_HG_SL = 6.41e-4
    DMPC_Tails_SL = -3.08e-4
    H2O_SL = -1.4e-5
    D2O_SL = 2e-4

    ## Relate HG Bound Waters to SL and Volume
    HG_Waters_D2O_SL = HG_bound_waters * D2O_SL
    HG_Waters_H2O_SL = HG_bound_waters * H2O_SL

    Headgroup_water_volume = HG_bound_waters * Water_Volume

    ## Add to HG volumes, SLs and Calculate nSLD

    #Volume_HG = ? + ? ##The headgroup volume plus the water volume

    #DMPC_HG_SL_D2O = ? + ? ##Clue it's the same as before but with the
water added
    #DMPC_HG_SL_H2O = ? + ?

    #SLD_HG_D2O = DMPC_HG_SL_D2O/Volume_HG
    #SLD_HG_H2O = DMPC_HG_SL_H2O/Volume_HG
    #SLD_HG_SMW = (0.38 * SLD_HG_D2O) + (0.62 * SLD_HG_H2O)

    #Tails_SLD = DMPC_Tails_SL/DMPC_Tails_Volume

    ## Thicknesses
    ## Calculate the thickness from the HG volume over the lipid Area per
    ## molecule

    #Headgroup_thickness = Volume_HG/DMPC_APM
    #Tails_thickness = DMPC_Tails_Volume/DMPC_APM

    ## Now construct your layers adding in a single roughness parameter and
a
```

```
## parameter for defects (hydration)

oxide = [oxide_thick, oxide_SLD, sub_rough, oxide_hydration, 2]
#Head_D2O = [?, ?, ?, ?, 2]
#Head_H2O = [?, ?, ?, ?, 2]
#Head_SMW = [?, ?, ?, ?, 2]
#Tail = [?, ?, ?, ?, 2]

## Add your layer defined above to the layer stack
match contrast:
    case 1 | 2:
       output = np.array([oxide])
    case 3:
        output = np.array([oxide])
    case 4:
        output = np.array([oxide])
    case 5:
        output = np.array([oxide])



return output, sub_rough
```

In this code you will notice there are single commented lines (**#**) and double commented (##) lines. The **double commented** lines are **notes and hints** while the **single commented** lines are the **lines of code you must complete** by the replacing the questions marks (?) with the correct information.

The parameters you wish to fit are commented out in the parameters section of the code. These are:

```
HG_bound_waters = params[3]
DMPC_APM = params[4]
Bilayer_roughness = params[5]
Bilayer_hydration = params[6]
```

These also need to be added to the parameter list in RasCal.

Suitable parameter ranges for these are:

HG_bound_waters: lower = 0 waters, upper = 10 waters;

DMPC_AreaPerMolecule: lower = 0 Å^2, Upper = 100 Å^2.

Bilayer_roughness: lower = 2 Å, upper = 10 Å.

Bilayer_hydration: lower = 0%, upper = 100%.

EXTRA HINT: Add these parameters to the list in RasCal

The code below the fitted parameters details the relationship between these parameters and the parameters which define the layers i.e. the layer parameters SLD, thickness, roughness and hydration.

Once you have uncommented and placed the correct info in all the lines of code you must add the bilayer structure to cases **3, 4** and **5**.

HINT! Input the following layer structure in contrast (Case) 2 and 3 once you have determined the relationship between area per molecule, thickness and SLD and defined your layers:

Case 3

output = ([Oxide, HEADFGROUP_D2O, TAIL, TAIL, HEADGROUP_D2O])

Case 4

output = ([Oxide, HEADFGROUP_SMW, TAIL, TAIL, HEADGROUP_SMW])

Note: Here we define the tails as two layers rather than a single layer (as was done previously).

You will be given **40 minutes** to complete this practical. After which time we will discuss the resolved structural information as a group.

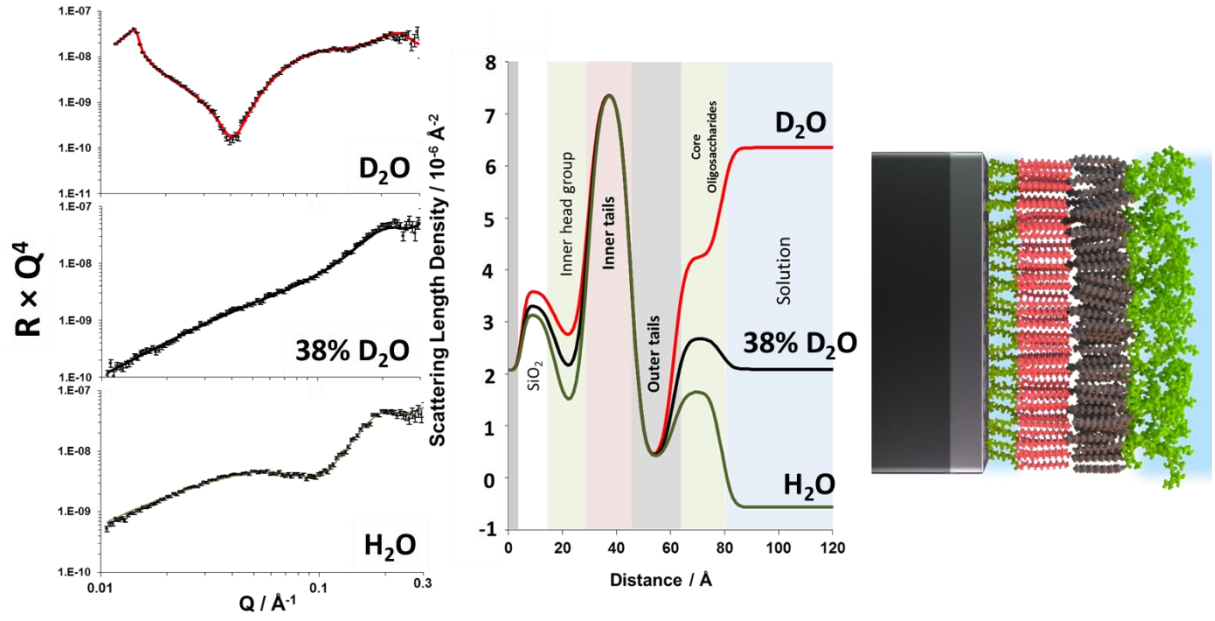
Save your project as a new project on the IDAaaS desktop.

Part 5: Fitting a Complex Protein-Bound Complex Membrane Structure

The final part of this practical is a challenge of you new-found fitting and scripting skills and will test some basic trigonometry.

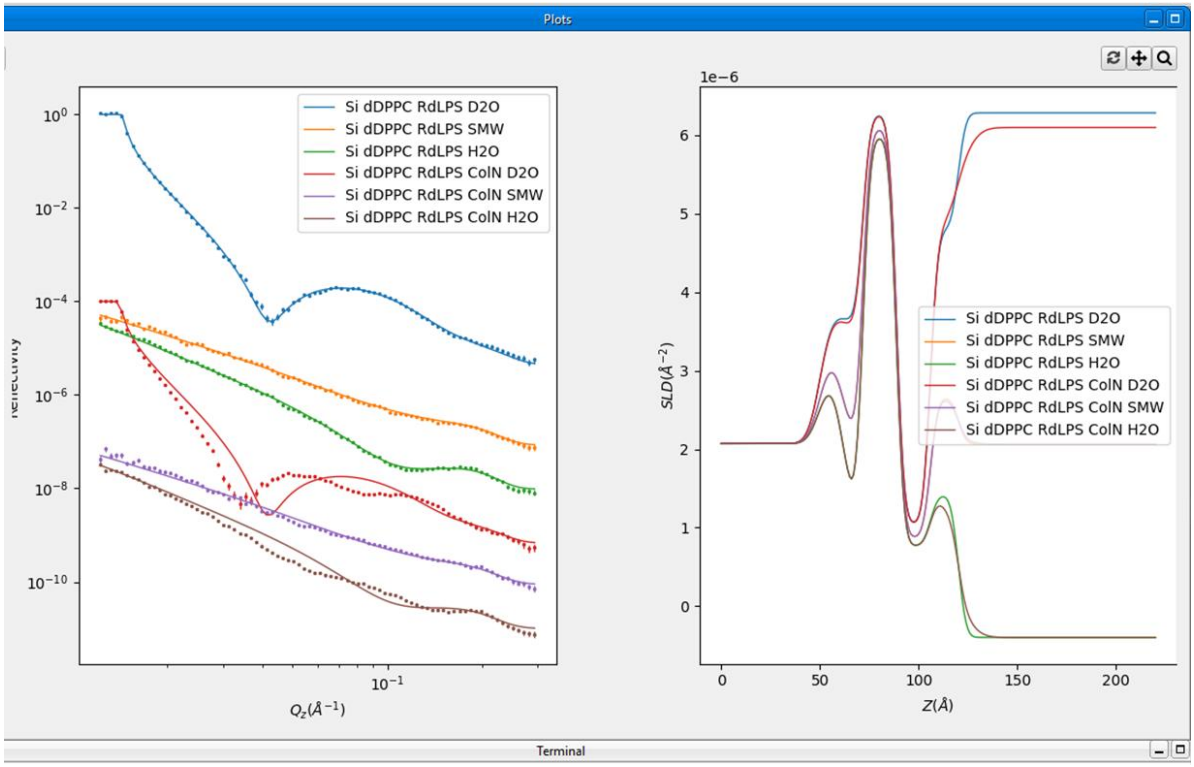
In the folder labelled **"Part 3 Protein-Bound Membrane Structure"** you will find a complete project with data which has previously been published in ***Langmuir 32 (14), 3485-3494***. In this example we have a complex membrane structure at the silicon-water interface which is representative of the Gram negative bacterial outer membrane.

This structure contains an asymmetric distribution of lipids with an inner leaflet composed tails deuterated phospholipid (in this case d_{62}-dipalmitoylphosphatidylcholine) and an outer leaflet of bacterial lipopolysaccharides which are hydrogenous (or 99.98% protium labelled). The figure below gives some details of this structure:



Above we have reflectometry data (shown in $R \times Q^4$ format, great for publications to show quality of fits) from the asymmetric membrane structure examined under three differing solution contrasts (D_2O, Si-MW (38% D_2O) and H_2O). From the analysis of the data we reveal the internal structure of the bilayer at the solid/liquid interface. The deuterated phospholipid component is located in the inner membrane leaflet close to the silicon interface and the hydrogenous lipopolysaccharide is located in the outer leaflet close to the bulk solution. Note that the profile for this consists of lipid tail and sugar head-group (core oligosaccharide) region.

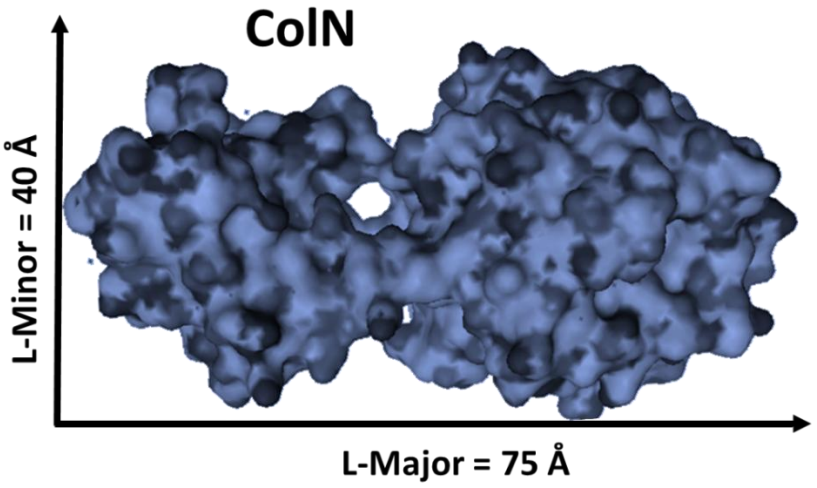
Upon opening the **Part 5 Complex model** folder the following data sets and fits will load in RasCal:



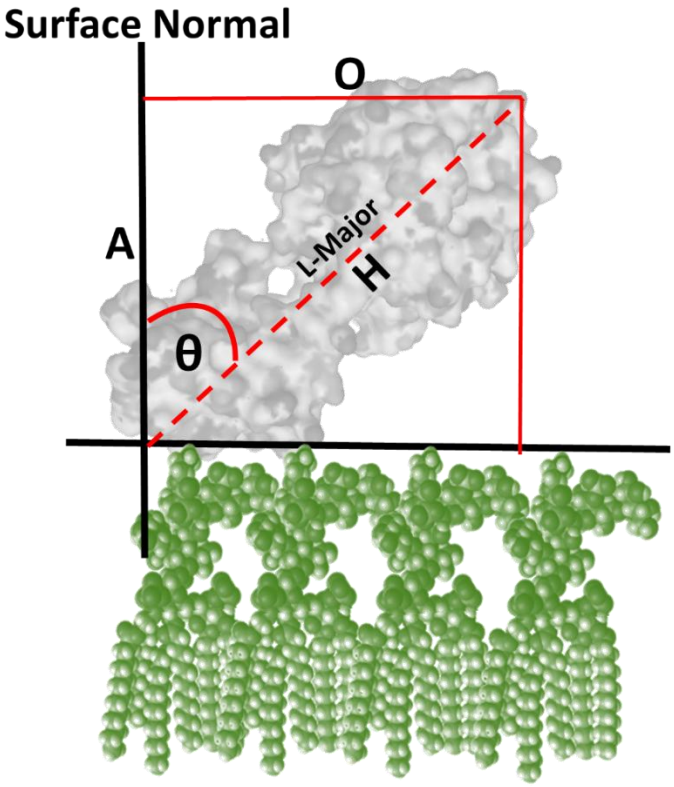
You will see six NR data sets which describe an asymmetric membrane structure across the silicon-water interface.

The top three data sets come from the membrane only and you can see are well fitted to the SLD profile shown on the left. The bottom three data sets are the same membrane after the binding of the antibacterial protein colicin-N (ColN) to the surface and, as can be seen, are not well fitted to a model which does not have the protein on the surface (line is model fit and error bars are data).

ColN is a cigar shaped protein:



This prolate shape means that **a;** with a single layer of protein on the surface of the membrane the thickness of that layer will be between **40** and **75 Å** and **b;** that from that thickness we can determine the angle of the protein relative to the membrane surface:



Using RasCal we will fit the angle of ColN relative to the surface normal and convert this value in the RasCal custom file into a thickness which will be input into the Abeles matrix calculation in the software.

The area of code you need to pay the most attention to is the area where we convert protein surface angle to thickness (lines 26 to 51):

```
    ##ColN angle to thickness conversion code! Remember your trigonometry

    ## Protein axes
    ColN_minor_axis = 40
    ##COlN is a cigar shaped protein and this length is the minor (thinnest
axis)
    ColN_major_axis = 75 ##This is the longest axis.


    ## Convert angle to radians
    ##Here we convert the angle to radians.
    ColN_Layer_angle_rad = ColN_Layer_angle * np.pi / 180.0

    ## Ratio from angle
    ##Here we you need to
    ##specify how we determine the ratio between our two length scales.

    ColN_ratio = 0
```

```
## Thickness interpolation
## Here we then need to relate the ratio to the thickness
## of the proteins minor and major axis

ColN_Layer_Thickness = 0

## From here you don't need to change anything
```

Note the parameters that describe what you want to fit for the protein are already in the parameter list but they have not been added to the structure yet which you need to do once the relationship between angle and protein layer thickness has been defined.

Once you have done this save the custom model and Accept changes. If the model compiles fit the data to resolve the angle of ColN relative to the membrane surface.

HINT: Just select the protein layer parameters to fit

SAVE PROJECT ON THE IDAAAS DESKTOP IF YOU WISH TO LOOK AT IT FURTHER

HINT!: Trigonometry:

SOH-CAH-TOA

Sin = Opposite/Hypotenuse

In python:

y=np.sin(x)

x is angle in radians (angle in degrees × π/180)

Cos = Adjacent/ Hypotenuse

In python:

y=np.cos(x)

Tan = Opposite/Adjacent

In python:

y=np.tan(x)

A Brief Introduction to Parratt's Recursive Algorithm:

INPUT:

Q - single value [$Å^{-1}$] (from an iteration loop)

CONSTANTS:

N - total number of layers (excluding subphase).
ρ_n - scattering length density [$Å^{-2}$] of layer n.
σ_n - roughness [Å] of layer n.
d_n - layer thickness [Å] of layer n N - total number of layers (excluding subphase).

INITIAL CONDITIONS:

Neutron beam incomes via layer n = 0
Layer $N + 1$ is the subphase with $d_{N+1} = \infty$ (e.g. D_2O, Si etc)
$R_{N+1} = 0$, which implies $r_{N+1} = r_{N,N+1}$
For N layers loop over n by decrementing from n = N +1 to n = 1

EQUATIONS

$$K_0 = \sqrt{\frac{Q^2}{4} + 4\pi\rho_0} \tag{1}$$

$$K_n = \sqrt{K_0^2 + 4\pi(\rho_{n-}\rho_0)} \tag{2}$$

For the last film layer when $n = N$:

$$r_{(N,N+1)} = \frac{K_N - K_{N+1}}{K_N + K_{N+1}} e^{-2i[\sigma_{N+1}]^2 K_N K_{N+1}} \tag{3}$$

$$r_N = \frac{r_{(N-1,N)} + r_{(N,N+1)} e^{2id_n k_N}}{1 + r_{(N-1,N)}\, r_{(N,N+1)} e^{2id_n k_N}} \tag{4}$$

For all other interfaces where n < N:

$$r_{(n-1,n)} = \frac{K_{n-1} - K_n}{K_{n-1} + K_n} e^{-2i[\sigma_n]^2 K_{n-1} K_n} \quad (5)$$

$$r_N = \frac{r_{(n-1,n)} + r_{(n+1)} e^{2id_n k_n}}{1 + r_{(n-1,n)}\, r_{(n+1)} e^{2id_n k_n}} \quad (6)$$

Where k_0 is the wavenumber of the incoming neutron beam; k_n is the neutron beam wave-vector in layer n; $r_{N,N+1}$ is the reflectivity from the interface between the last layer N and the subphase $N + 1$ and $r_{N-1,N}$ can be found using the equation (5). Note that for n = N − 1 the r_{n+1} in equation (6) is equal to the value of r_N from equation (4).

Finally once the process has been recursively calculated for each layer interface moving from bottom to top R(Q) is calculated from the reflectivity coefficient of the top interface ,r1, multiplied by its complex conjugate (7):

$$R(Q) = r_1 (r_1)^{\dagger} \quad (7)$$