Polymer Layer Example for ORSO

Start by making an instance of the projectClass

```
problem = projectClass('Orso polymer example');
problem.setGeometry('Substrate/liquid');
```

We now need to add all the parameters we need. Define them all as a group and then add them to the problem definition class:

```
Parameters = {
    %       Name                min         val       max       fit?
        {'Oxide thick',         5,          20,       60,       true   };
        {'Oxide SLD',           3.41e-6     3.4e-6    3.42e-6   false  };
        {'Oxide Hydration'      0,          0.2,      0.5,      true   };
        {'Polymer thick'        100         200       400       true   };
        {'Polymer SLD'          1e-6        2e-6      6e-6      true   };
        {'Polymer hydration'    0           0.1       0.2       true   };
        {'Polymer Rough'        2           4         8         true   };
        };
```

Add these to the project:

```
problem.addParamGroup(Parameters);
problem.setParameter(1,'min',1,'max',10); % Change the substrate roughness limits
```

Now we need to group the parameters into the layers that we need in the normal Rascal way

```
oxide = {'Oxide layer',...                % Layer name
                'Oxide thick',...         % Thickness
                'Oxide SLD',...           % SLD
                'Substrate Roughness',... % Roughness
                'Oxide Hydration',...     % Hydration
                'bulk out' };             % Hydrating bulk phase

polymer = {'Polymer layer',...            % Layer name
                'Polymer thick',...       % Thickness
                'Polymer SLD',...         % SLD
                'Polymer Rough',...       % Roughness
                'Polymer hydration',...   % Hydration
                'bulk out' };             % Hydrating bulk phase
```

```
% Add the layers to the projectClass
problem.addLayerGroup({oxide ; polymer});
```

Edit the incoming bulk phase to be Silicon instead of air..

```
problem.setBulkIn(1,'name','Silicon','min',2.07e-6,'value',2.073e-6,'max',2.08e-6,'fit',false);
```

And make the D2O fittable:

```
problem.setBulkOut(1,'fit',true,'min',5e-6);
```

..and adjust the scalefactor to be more suited to a solid/liquid experiment...

```
problem.setScalefactor(1,'Value',1,'min',0.5,'max',2,'fit',true);

problem.setBacksPar(1,'fit',true);
```

Now add the datafiles to the project:

```
% Read in the data file
thisData = dlmread('polymerData.dat');

% Add the data to the problem
problem.addData('Polymer data',thisData);
```

Now group everything together into a contrast:

```
problem.addContrast('name','Si/Oxide/Polymer/D2O',...
    'background','Background 1',...
    'resolution','Resolution 1',...
    'scalefactor', 'Scalefactor 1',...
    'nbs', 'SLD D2O',...
    'nba', 'Silicon',...
    'data', 'Polymer data');
```

And set the model for our contrast by adding the two layers:

```
problem.setContrastModel(1,{'oxide layer','polymer layer'});
```

Take a look at our problem definition to see if everything in OK..

```
disp(problem)
```

```
     ModelType: 'Standard Layers'
experimentName: 'Orso polymer example'
      Geometry: 'substrate/liquid'
```

Parameters:

p	Name	Min	Value	Max	Fit?
1	"Substrate Roughness"	1	3	10	true
2	"Oxide thick"	5	20	60	true
3	"Oxide SLD"	3.41e-06	3.4e-06	3.42e-06	false
4	"Oxide Hydration"	0	0.2	0.5	true
5	"Polymer thick"	100	200	400	true
6	"Polymer SLD"	1e-06	2e-06	6e-06	true
7	"Polymer hydration"	0	0.1	0.2	true
8	"Polymer Rough"	2	4	8	true

Layers:

p	Name	Thickness	SLD	Roughness	Hydration	Hydrate with
1	"Oxide layer"	"Oxide thick"	"Oxide SLD"	"Substrate Roughness"	"Oxide Hydration"	"bulk out"
2	"Polymer layer"	"Polymer thick"	"Polymer SLD"	"Polymer Rough"	"Polymer hydration"	"bulk out"

Bulk In:

p	Name	Min	Value	Max	Fit?
1	"Silicon"	2.07e-06	2.073e-06	2.08e-06	false

Bulk Out:

p	Name	Min	Value	Max	Fit?
1	"SLD D2O"	5e-06	6.35e-06	6.35e-06	true

Scalefactors:

p	Name	Min	Value	Max	Fit?
1	"Scalefactor 1"	0.5	1	2	true

Backgrounds:

(a) Background Parameters:

p	Name	Min	Value	Max	Fit?
1	"Backs par 1"	1e-07	1e-06	1e-05	true

(b) Backgrounds:

p	Name	Type	Value 1	Value 2	Value 3	Value 4	Value 5
1	"Background 1"	"constant"	"Backs Par 1"	""	""	""	""

Resolutions:

(a) Resolutions Parameters:

p	Name	Min	Value	Max	Fit?
1	"Resolution par 1"	0.01	0.03	0.05	false

(b) Resolutions:

p	Name	Type	Value 1	Value 2	Value 3	Value 4	Value 5
1	"Resolution 1"	"gaussian"	"Resolution par 1"	""	""	""	""

Data:

Name	Data	Data Range	Simulation Range
"Simulation"	"No Data"	"–"	"[0.0050 , 0.7000]"
"Polymer data"	"Data array: [408 x 3]"	"[0.0081 , 0.4656]"	"[0.0050 , 0.7000]"

Custom Files:

Name	Filename	Language	Path
""	""	""	""

Constrasts:

p	1
"name"	"Si/Oxide/Polymer/D2O"
"Data"	"Polymer data"
"Background"	"Background 1"
"Bulk in"	"Silicon"

```
"Bulk out"       "SLD D2O"
"Scalefactor"    "Scalefactor 1"
"Resolution"     "Resolution 1"
"Model"          "oxide layer"
""               "polymer layer"
```

Pass all this to RAT to check what we have. For this we need a controlsDef class

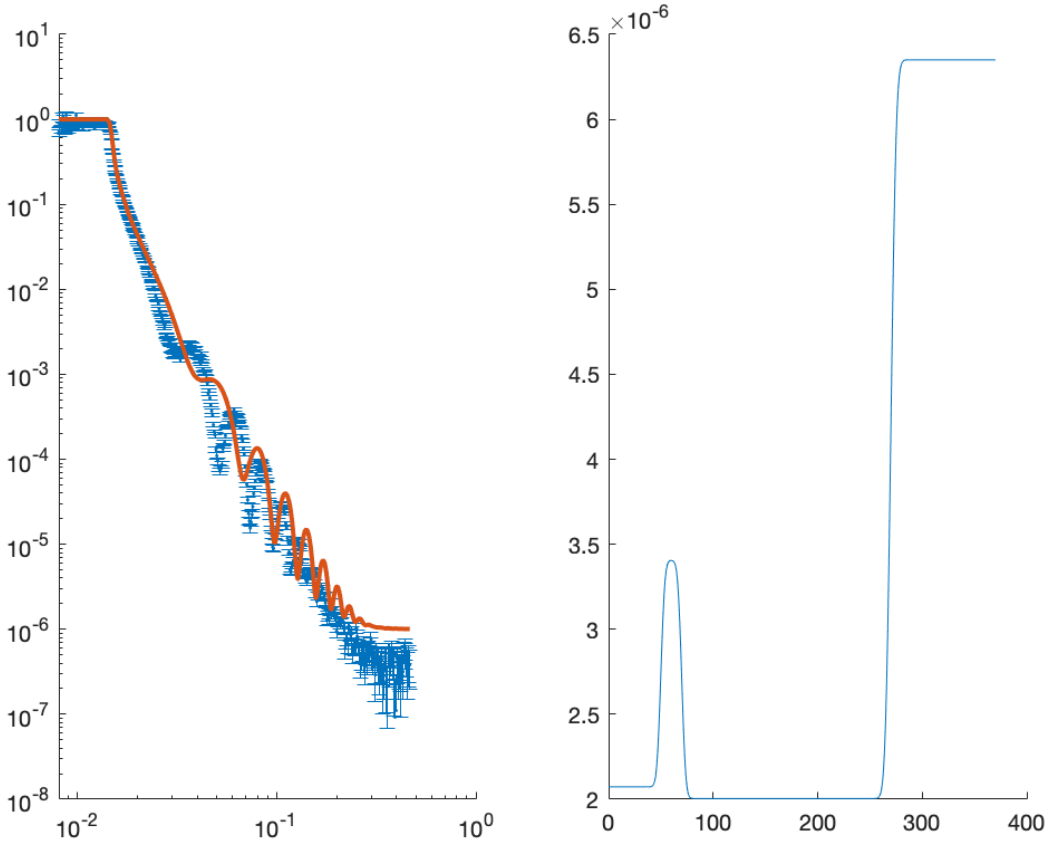
```
controls = controlsDef()
```

```
controls =
  controlsDef with properties:

           parallel: 'single'
          procedure: 'calculate'
    calcSldDuringFit: 'no'
```

```
% Pass these to RAT:
[problem,results] = RAT(problem,controls);
```

```
Starting RAT
_____________________________________________________________________________________________

Elapsed time is 0.003692 seconds.

Finished RAT
_____________________________________________________________________________________________
```

```
% Plot the results to see what we can see
plotRefSLD(problem,results)
```



Need to do a fit - use Differential Evolution:

```
controls.procedure = 'DE';
```

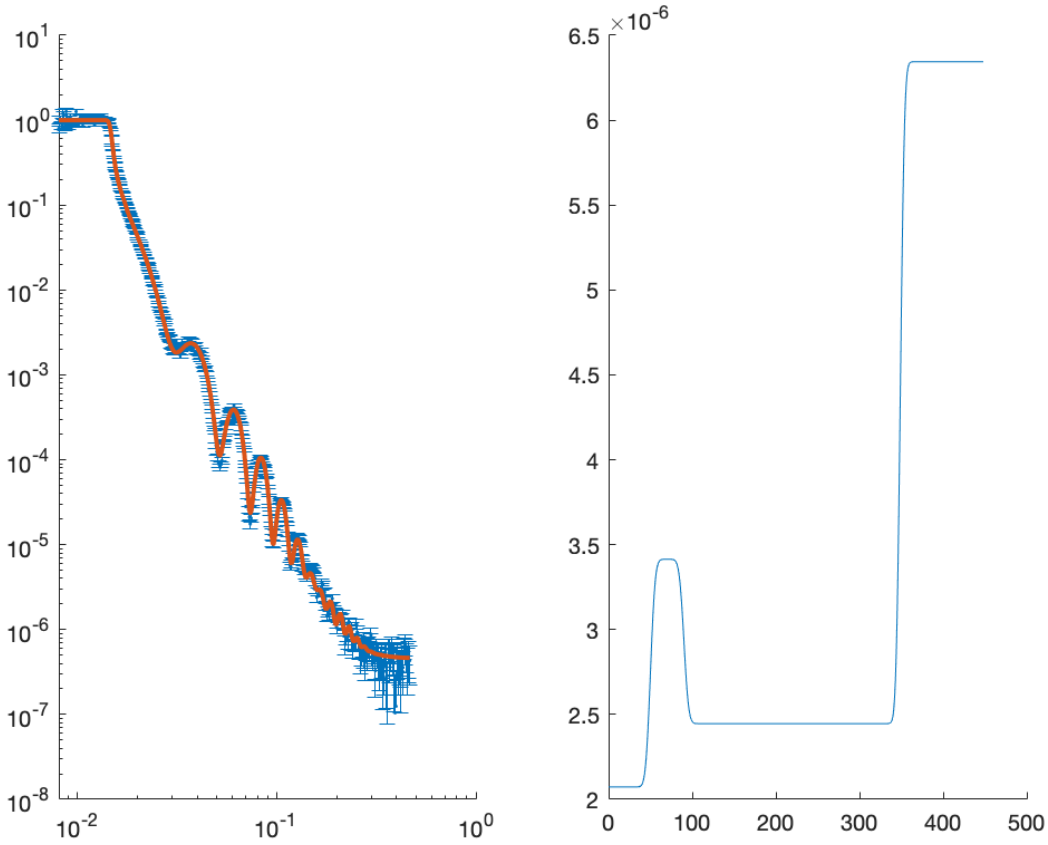
```
Warning: Negative data ignored
```

```
controls.numGenerations = 200;

% Run the fit.....
[problem,results] = RAT(problem,controls);
```

```
Starting RAT
_____________________________________________________________________________________________

***Starting Differential Evolution***

Elapsed time is 3.238821 seconds.

Finished RAT
_____________________________________________________________________________________________
```

Plot this to see the improvement:

```
figure; clf
plotRefSLD(problem,results)
```



Now do a Bayesian analysis to see how well the parameters are defined:

```
% Change the procedure in controls to Bayes
controls.procedure = 'bayes';
controls.repeats = 3;
controls.nsimu = 3000;
```

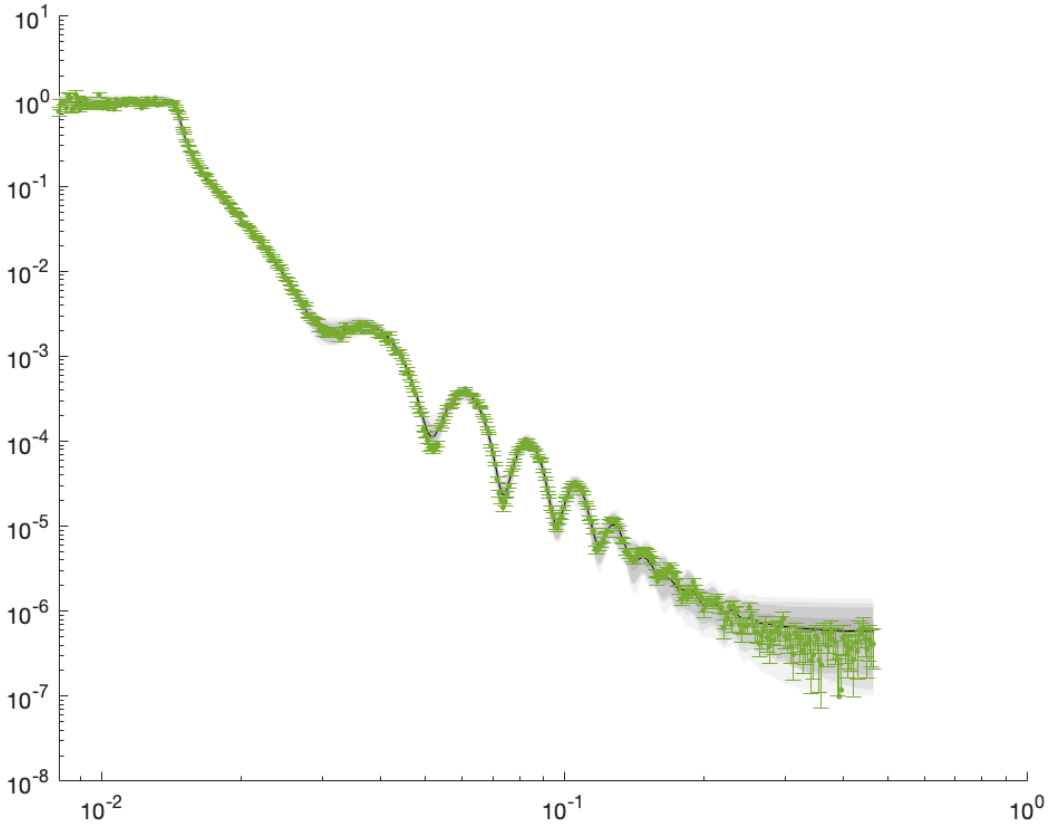
Send this to RAT:

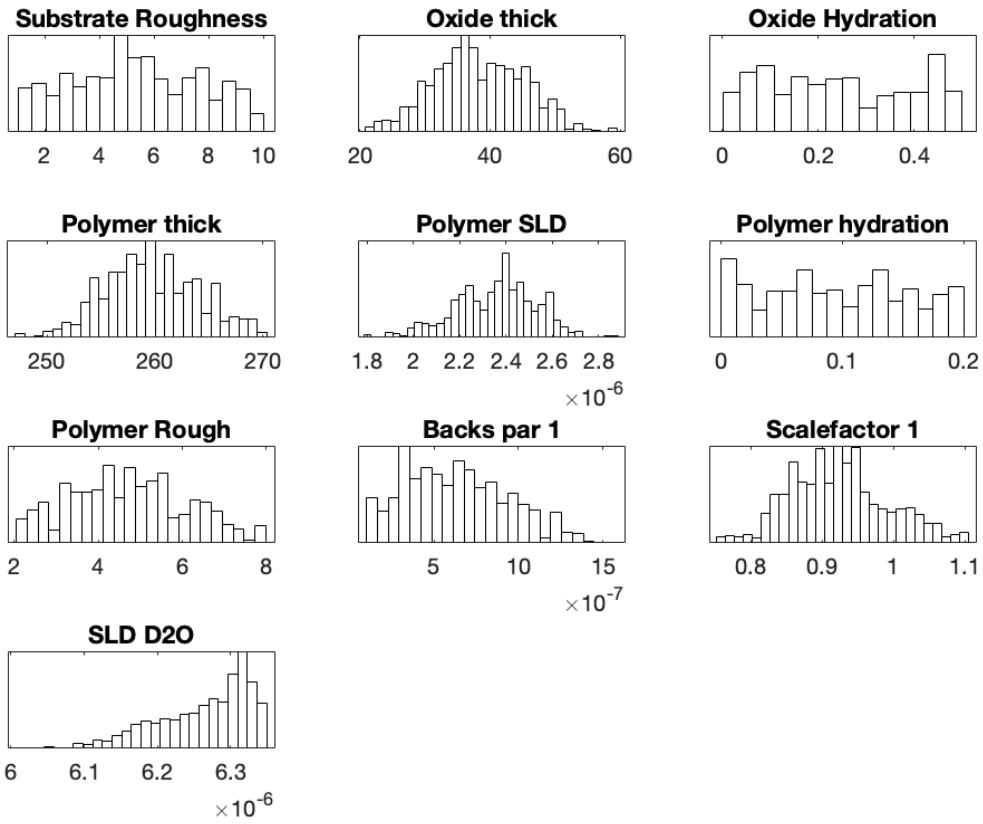
```
[problem,results] = RAT(problem,controls);
```

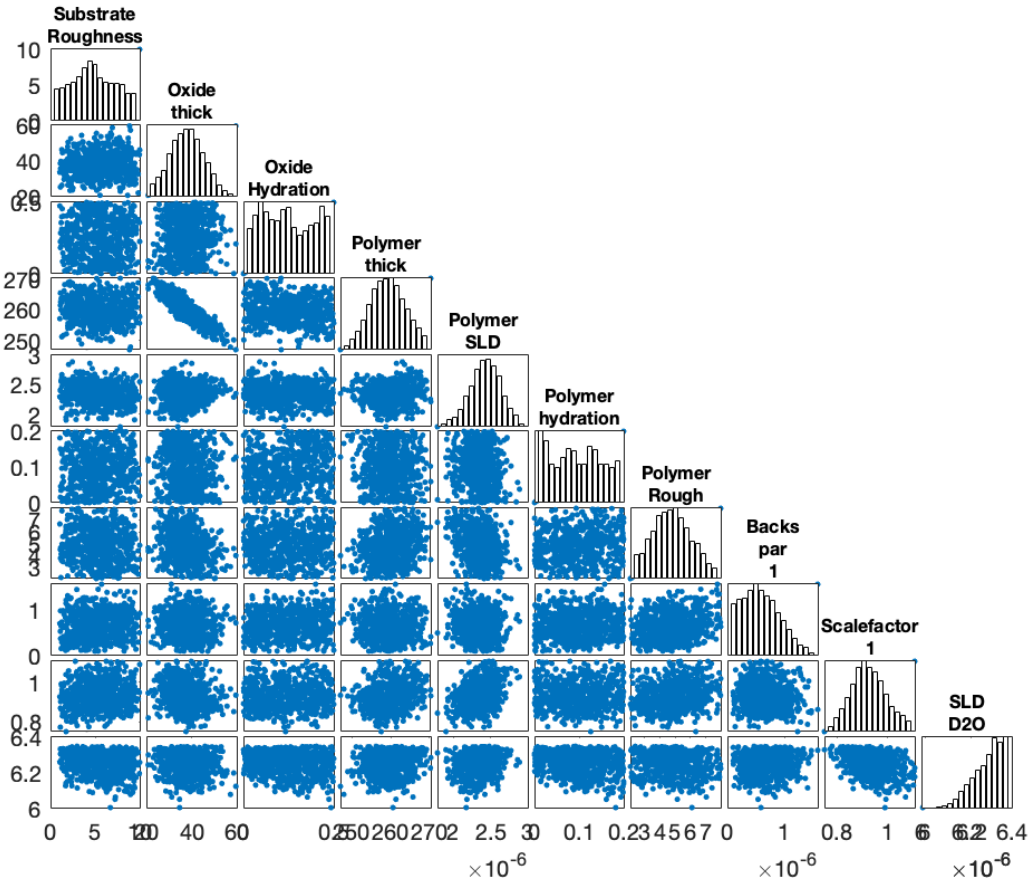
```
Starting RAT
____________________________________________________________________________________________

Running DRAM

Running loop 1 of 3

 Bayes: 100.0% [****************************************]

Running loop 2 of 3 Using values from the previous run

 Bayes: 100.0% [****************************************]

Running loop 3 of 3 Using values from the previous run

 Bayes: 100.0% [****************************************]

Elapsed time is 5.243800 seconds.

Finished RAT
____________________________________________________________________________________________
```

Plot everything out:

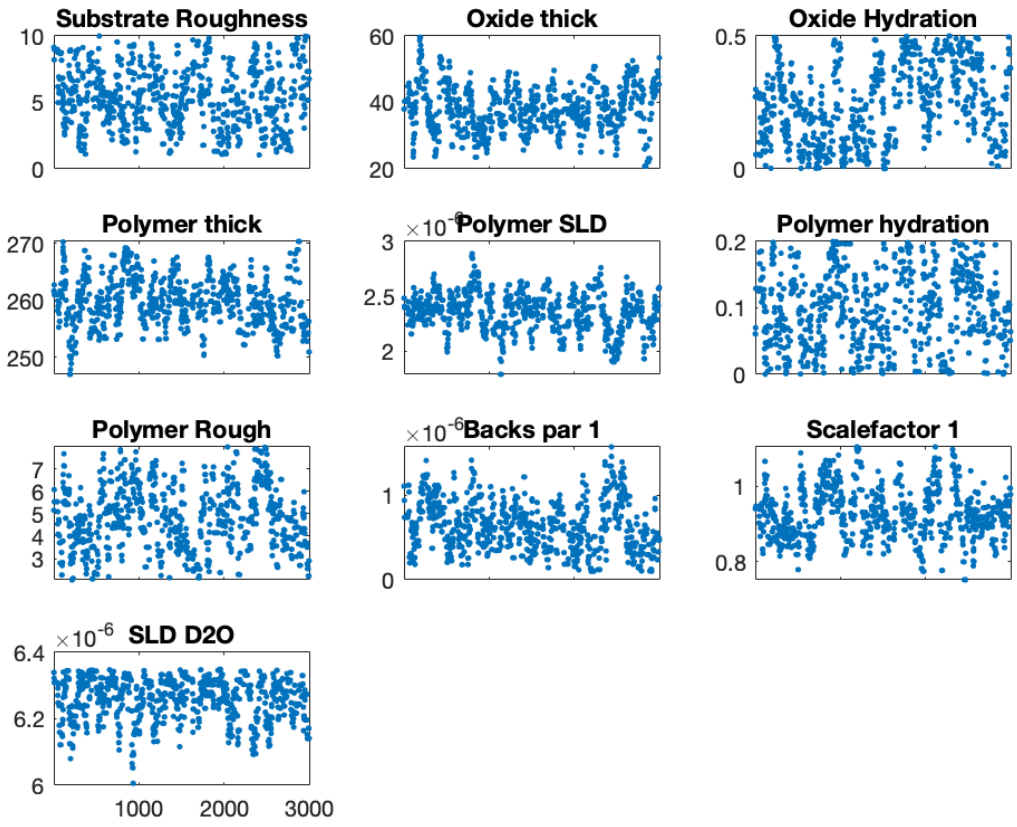
```
switch controls.procedure
    case 'bayes'
        h2 = figure(3); clf
        sf = results.contrastParams.scalefactors;
        bayesShadedPlot(h2,results.predlims,results.shifted_data,sf);

        h3 = figure(4); clf
        mcmcplot(results.chain,[],results.fitNames,'hist');

        h4 = figure(5); clf;
        plotBayesCorrFig(results.chain,results.fitNames,h4)

        h6 = figure(6); clf
        mcmcplot(results.chain,[],results.fitNames,'chainpanel');

    otherwise
        h2 = figure(2); clf
        plotRefSLD(problem,results)
end
```


10^{1}
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
10^{-7}
10^{-8}
10^{-2}
10^{-1}
10^{0}


Substrate Roughness
2 4 6 8 10
Oxide thick
20 40 60
Oxide Hydration
0 0.2 0.4
Polymer thick
250 260 270
Polymer SLD
1.8 2 2.2 2.4 2.6 2.8
×10^-6
Polymer hydration
0 0.1 0.2
Polymer Rough
2 4 6 8
Backs par 1
5 10 15
×10^-7
Scalefactor 1
0.8 0.9 1 1.1
SLD D2O
6 6.1 6.2 6.3
×10^-6



Warning: Negative data ignored



The outputted projectClass ('problem') now has the best-fit values of the parameters:

```
disp(problem)
```

```
     ModelType: 'Standard Layers'
experimentName: 'Orso polymer example'
      Geometry: 'substrate/liquid'

Parameters: ------------------------------------------------------------------------------------------------

p           Name                 Min          Value          Max        Fit?
_    ____________________    ________    ___________    ________    _____

1    "Substrate Roughness"          1         7.2427          10    true
2    "Oxide thick"                  5         38.111          60    true
3    "Oxide SLD"              3.41e-06        3.4e-06    3.42e-06    false
4    "Oxide Hydration"                0        0.25847         0.5    true
5    "Polymer thick"                100         260.68         400    true
6    "Polymer SLD"                1e-06    2.4189e-06        6e-06    true
7    "Polymer hydration"              0      0.077941         0.2    true
8    "Polymer Rough"                  2        4.9342           8    true

Layers: ----------------------------------------------------------------------------------------------------

    p        Name             Thickness          SLD             Roughness              Hydration
Hydrate with
    _    _____________    _______________    ____________    ____________________    ___________________
_____________

    1    "Oxide layer"      "Oxide thick"      "Oxide SLD"      "Substrate Roughness"    "Oxide Hydration"
"bulk out"
    2    "Polymer layer"    "Polymer thick"    "Polymer SLD"    "Polymer Rough"          "Polymer hydration"
"bulk out"
```

Bulk In: ----------

p	Name	Min	Value	Max	Fit?
1	"Silicon"	2.07e-06	2.073e-06	2.08e-06	false

Bulk Out: ----------

p	Name	Min	Value	Max	Fit?
1	"SLD D2O"	5e-06	6.3018e-06	6.35e-06	true

Scalefactors: ----------

p	Name	Min	Value	Max	Fit?
1	"Scalefactor 1"	0.5	0.93218	2	true

Backgrounds: ----------

(a) Background Parameters:

p	Name	Min	Value	Max	Fit?
1	"Backs par 1"	1e-07	8.7115e-07	1e-05	true

(b) Backgrounds:

p	Name	Type	Value 1	Value 2	Value 3	Value 4	Value 5
1	"Background 1"	"constant"	"Backs Par 1"	""	""	""	""

Resolutions: ----------

(a) Resolutions Parameters:

p	Name	Min	Value	Max	Fit?
1	"Resolution par 1"	0.01	0.03	0.05	false

(b) Resolutions:

p	Name	Type	Value 1	Value 2	Value 3	Value 4	Value 5
1	"Resolution 1"	"gaussian"	"Resolution par 1"	""	""	""	""

Data: ----------

Name	Data	Data Range	Simulation Range
"Simulation"	"No Data"	"-"	"[0.0050 , 0.7000]"
"Polymer data"	"Data array: [408 x 3]"	"[0.0081 , 0.4656]"	"[0.0050 , 0.7000]"

Custom Files: ----------

Name	Filename	Language	Path
""	""	""	""

Constrasts: ----------

p	1
"name"	"Si/Oxide/Polymer/D2O"
"Data"	"Polymer data"
"Background"	"Background 1"
"Bulk in"	"Silicon"
"Bulk out"	"SLD D2O"
"Scalefactor"	"Scalefactor 1"
"Resolution"	"Resolution 1"
"Model"	"oxide layer"
""	"polymer layer"

Warning: Negative data ignored

The parameter errors are calculated as the shortest 95% confidence interval of each posterior, but this is still under construction and not outputted yet by RAT. The best fit values are now in ‘problem’, but we can run a separate routine to obtain the errors (95% confidence intervals):

vals = iterShortest(chain,nParams,values,0.95,results.fitNames)

Substrate Roughness 7.6163 (1.2083, 7.843)

Oxide thick 39.988 (28.93, 45.887)

Oxide Hydration 0.3607 (0.0054328, 0.45312)

Polymer thick 257.9 (255.31, 267.45)

Polymer SLD 2.4342e-06 (2.1235e-06, 2.6259e-06)

Polymer hydration 0.0737 (0.046976, 0.19911)

Polymer Rough 3.365 (2.1812, 5.7264)

Backs par 1 4.045e-07 (1.0896e-07, 9.2767e-07)

Scalefactor 1 0.98243 (0.82826, 1.0137)

SLD D2O 6.226e-06 (6.2253e-06, 6.346e-06)