Local Basis Function Representations of Thermodynamic Surfaces

(Submitted)

by J.Michael Brown

This script provides examples of fitting data as a function of 1, 2 or 3 independent variables using the function "spdft" (**sp**line **d**ata **f**itt**e**r). The function allows use of any order tensor b spline with arbitrary distributions of "control points" for the splines and "regularization" to allow "smooth" fitting of data (a specified derivative is required to be "small") that may not adequately span the parameter space.

Usage: **sp=spdft(X,Y,dY,options)**

The input (X and Y) can be gridded or scattered and can be of variable (2, 3, or 4) dimensions. Gridded data are entered as cells and scattered data are entered as matrixes with columns for each independent variable.

X contains independent variables (matrix or cell), **Y** is the dependent variable (a vector), **dY** contains data uncertainties. (1) If **dY** is empty, the stdev(Y) is used as an estimate (2) If dY is a scalar, the same uncertainty is assigned to all data, (3) If dY is a vector (same size as Y) individual uncertainties are assigned to each data point

The following input options are entered within the structure "options"

- **Xc** (vector or cell-if more than one independent variable) are the "control points" used to determine knot locations. If Xc is not provided and X is a cell, Xc is set to X
- **lam** is a vector of smoothing parameters (one for each independent variable). If lam is not provided, values are set to 1
- **RegFac** is a vector of integers for the regularization oversampling (number of regularization points between each control point). (1) if RegFac is not included in options, the default is 1 for all dimensions. (2) if RegFac is included as an empty vector, a least square fit without regularization is attempted (this is likely to fail unless data coverage is adequate)
- **ordr** is a vector of integers defining the spline order for each independent variable. If ordr is not provided the order is set to 4 for all dimensions
- **mdrv** is a vector of integers defining which derivative is to be minimized. If mdrv is not provided, the default is set to 2 (make a smooth fit)
- **mask** is a list of ones or nan to specify which data to exclude from the fit. If mask is not provided, all values of mask are set to 1
- **algorithm** sets the method of solution. The default is to solve a set of "normal equations" . If algorithm is present in the options structure, a "Least Square" solution is undertaken

spdft requires functions from the matlab spline (curve fitting) toolbox. Custom function mkCmn mkgrid area_wt are nested within the function.

Three examples are shown below:

1. fitting with a single indenpendent variable
2. fitting with two independent variables
3. fitting with three independent variables

Fitting a Set of x-y Data (a single independent variable).

The independent variable, x, extends from 0 to 60 but the spline is determined to100 to show the behavior in extrapolation. The data are hard to fit with conventional polynomials since there are large and variable second derivatives for small x and smaller second derivatives for larger x. A range of damping values is used - from overdamped (essentially a linear fit to data) to underdamped (over fitting data) - notice that damping=1 achieves a reduced chi square near 1. Note also that the fit in the extrapolated region has zero second derivatives. Regularization in the absence of data controls the nature of the fit.

```
% Here are trial data to fit:
data2fit=XY2fit_data;
X1=data2fit(:,1);
Y1=data2fit(:,2);
uncert=.05;  % set constant data uncertainty - can be vector or empty
```

Below here, the spline fitting parameters are described. One must:

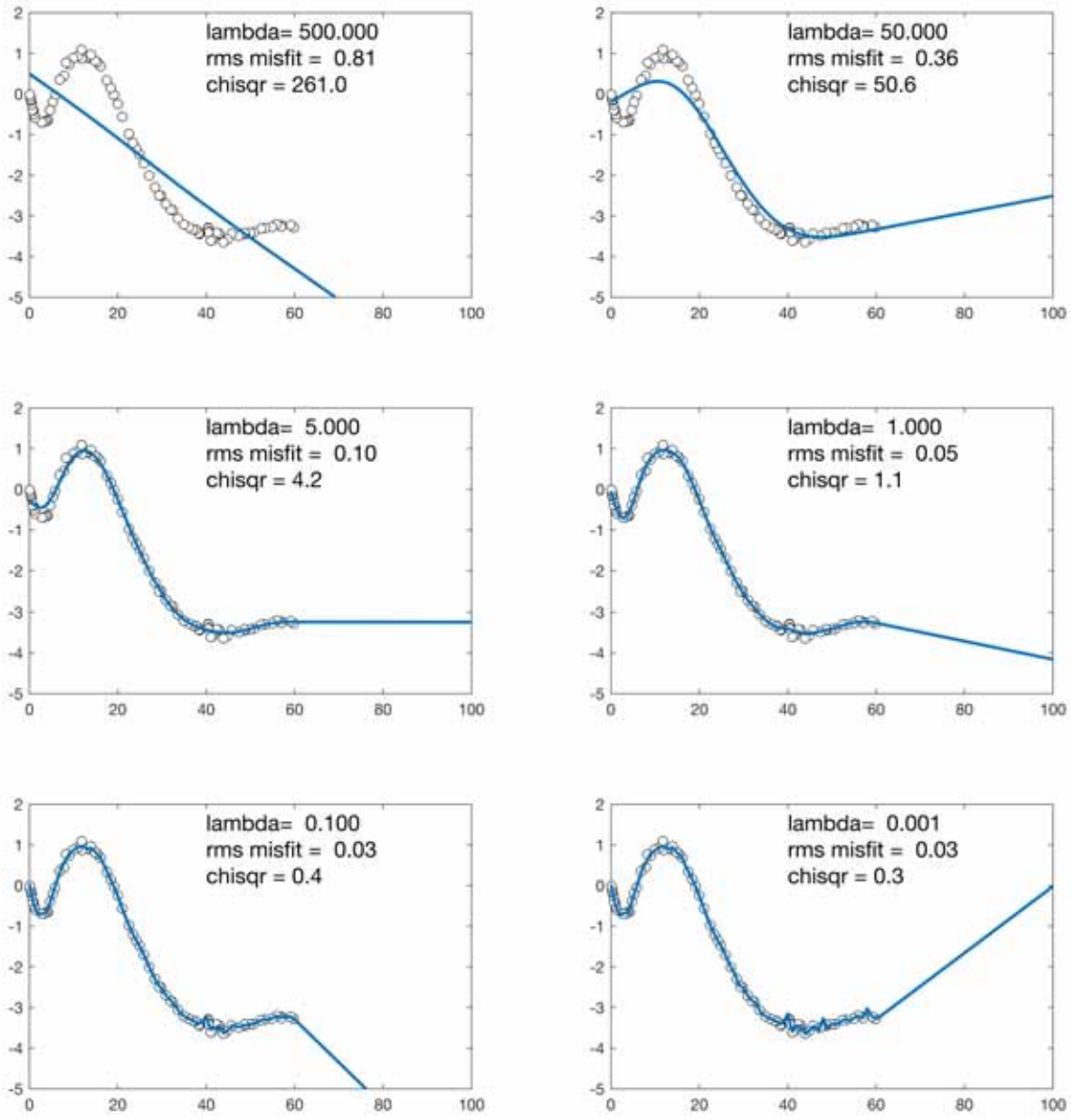
1. define control points,
2. determine how many regularization points per control point (1 is a typical value)
3. specify the order of the spline,
4. specify what derivative to minimize,
5. specify what damping factor to use

```
% for small control point spacings the fit should be insensitive to spacing choice.
% If the spacings are too wide, the spline will not have sufficient flexibility to fit the dat
dx=1;  % try .1 .5 1 2 5 10
clear options % it is a good idea to always clear options to insure that previously set proper
options.Xc=0:dx:100;  % the spline control points are set
```

Other options are allowed to be default values and thus do not need to be set. They are shown below as they would be input if different values were needed

```
% options.RegFac=1;  % chose amount of over-sampling of regularization (empty for no regulariz
% %              1 for 1 regularization point per control point
%
% options.order=4;  % increasing the spline order can compensate for wider control point spaci
% %           (try orders between 4 (minimum for smoothing) and 10 with larger or smaller dx v
%
% % here a smooth fit requires minimizing the second derivative of the fit:
% options.mdrv=2;

% try a range of damping factors - near 1 is always a place to start
lam=[ 500 50 5 1 .1 .001 ];
```

```
scrsz=get(0,'ScreenSize');
figsz=[1 .8*scrsz(4) 2*scrsz(3)/4 .8*scrsz(4)];
h=figure('Name','One Independent Variable','Position',figsz);

for i=1:length(lam)
    options.lam=lam(i);
    subplot(3,2,i)
% here is the call to the spline fitting function:
    sp1=spdft(X1,Y1,uncert,options);
% spdft puts fit statistics into the returned spline structure - pull them out and create a te
txt=sprintf('lambda= %6.3f \nrms misfit = %5.2f \nchisqr = %3.1f',lam(i),sp1.Data.rms,sp1.Data

plot(X1,Y1,'ko','MarkerFaceColor','w')
    hold on
    fnplt(sp1) % fnplt is a MATLAB utility for quickly plotting spline representations
    text(40,.9,txt,'FontSize',14)
    axis([0 100 -5 2 ])
    hold off
end
```




```
subplot(111)
```

Fitting a Set of x-y-z Data (two independent variables).

The Lin and Trusler (2012) sound speeds for pure water are fit as a function of pressure and temperature

```
LT=LT_data;  % this function returns the Lin and Trusler data set.
```

Chose the number of control points for P and T and set the grid of control points

```
npc=15;  % adjust these to chnage the numbers of control points
ntc=15;  % adjust these to chnage the numbers of control points
Xc=linspace(0,500,npc);
Yc=linspace(240,500,ntc);
clear options
options.Xc={Xc,Yc};
```

The values given below were manually adjusted to achieve a reduced chi square of about 1. Change these numbers to see the effect on misfits and smoothness of the resulting surface smaller values for less misfit and less "smoothness", larger values for greater misfit and smoothness

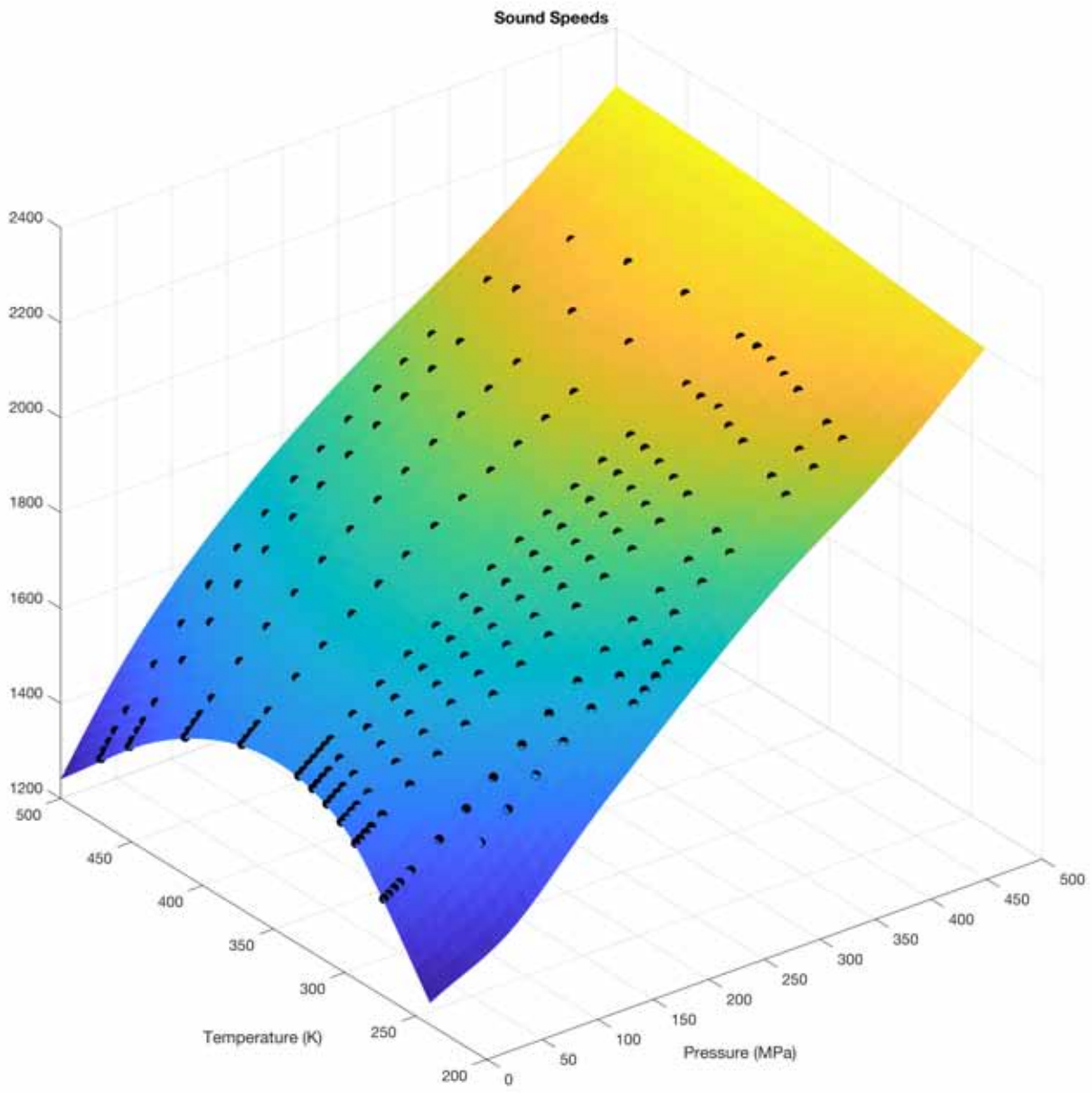
```
options.lam=45*[1 1];
```

All other options are allowed to be the default values. Since data are scattered, input X is a matrix with as many columns as independent variables

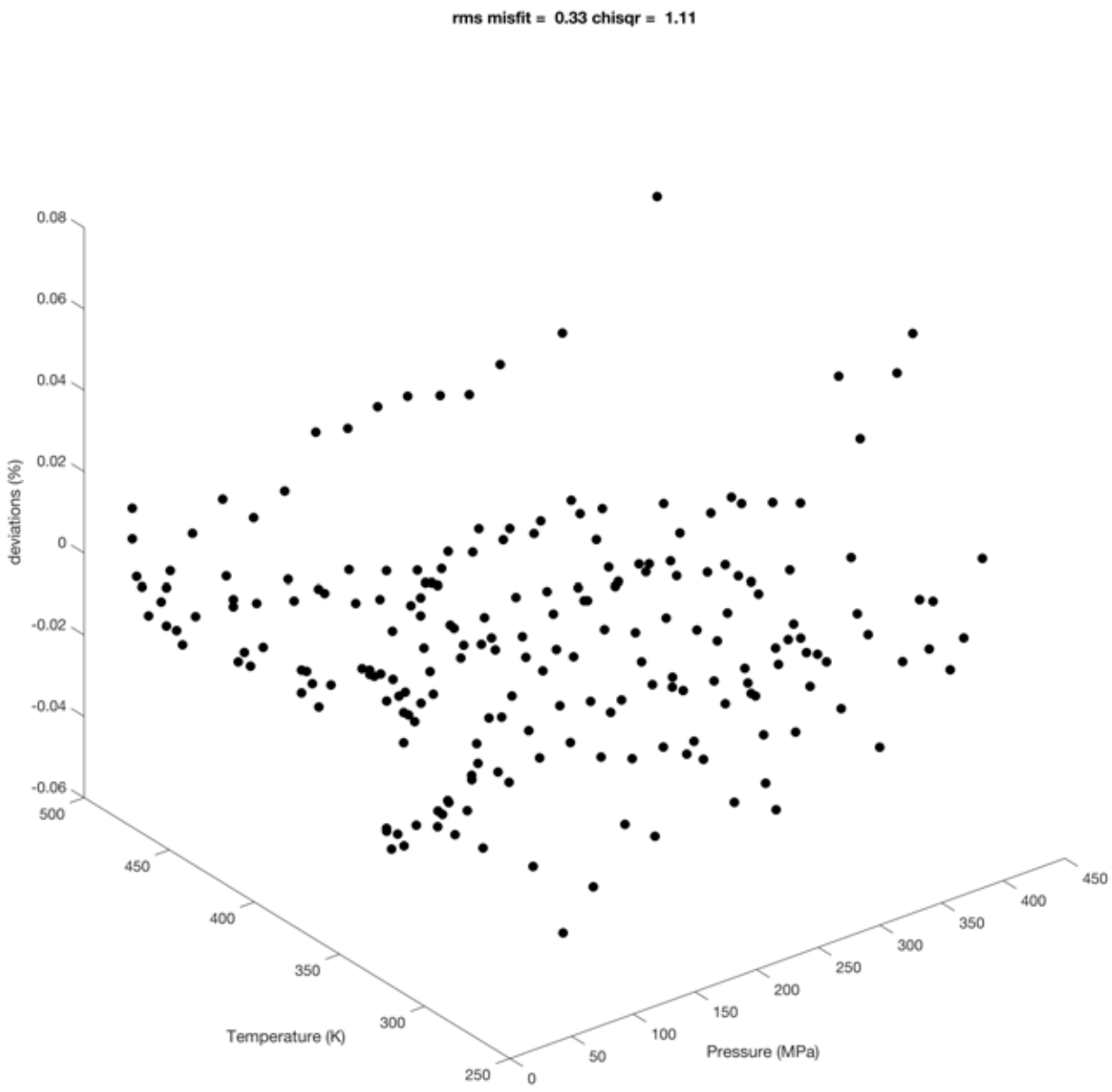
```
sp2=spdft(LT(:,1:2),LT(:,3),LT(:,4),options);
```

examine fit with plots of data and surface and the misfits of the data from the surface

```
txt=sprintf('rms misfit = %5.2f chisqr = %5.2f',sp2.Data.rms,sp2.Data.chisqr);
subplot(111)
clf
colormap gray
fnplt(sp2)
shading 'flat'
hold on
plot3(LT(:,1),LT(:,2),LT(:,3),'ko','MarkerFaceColor','k')
xlabel('Pressure (MPa)')
ylabel('Temperature (K)')
hold off
title('Sound Speeds')
```




```
plot3(LT(:,1),LT(:,2),1e2*(sp2.Data.devs)./LT(:,3),'ko','MarkerFaceColor','k')
xlabel('Pressure (MPa)')
ylabel('Temperature (K)')
zlabel('deviations (%)')
title(txt)
```



Fitting a Set of x-y-z-w Data (three independent variables).

The .mat file MgSO4Data.mat contains "P" "T" "m" "v" "dv" vectors of scattered data including results in Vance and Brown (2013) and other literature reported values including IAPWS for pure water below 400 MPa. Pressures in MPa, T in K, m in mol/kg, and v and dv in km/s

```
load MgSO4Data.mat
dv=.0025*v(:);
```

```
id=find(m(:)==0);
dv(id)=dv(id)/2;
```

Only damping and control points are specified - all else in options are defauklt vaules

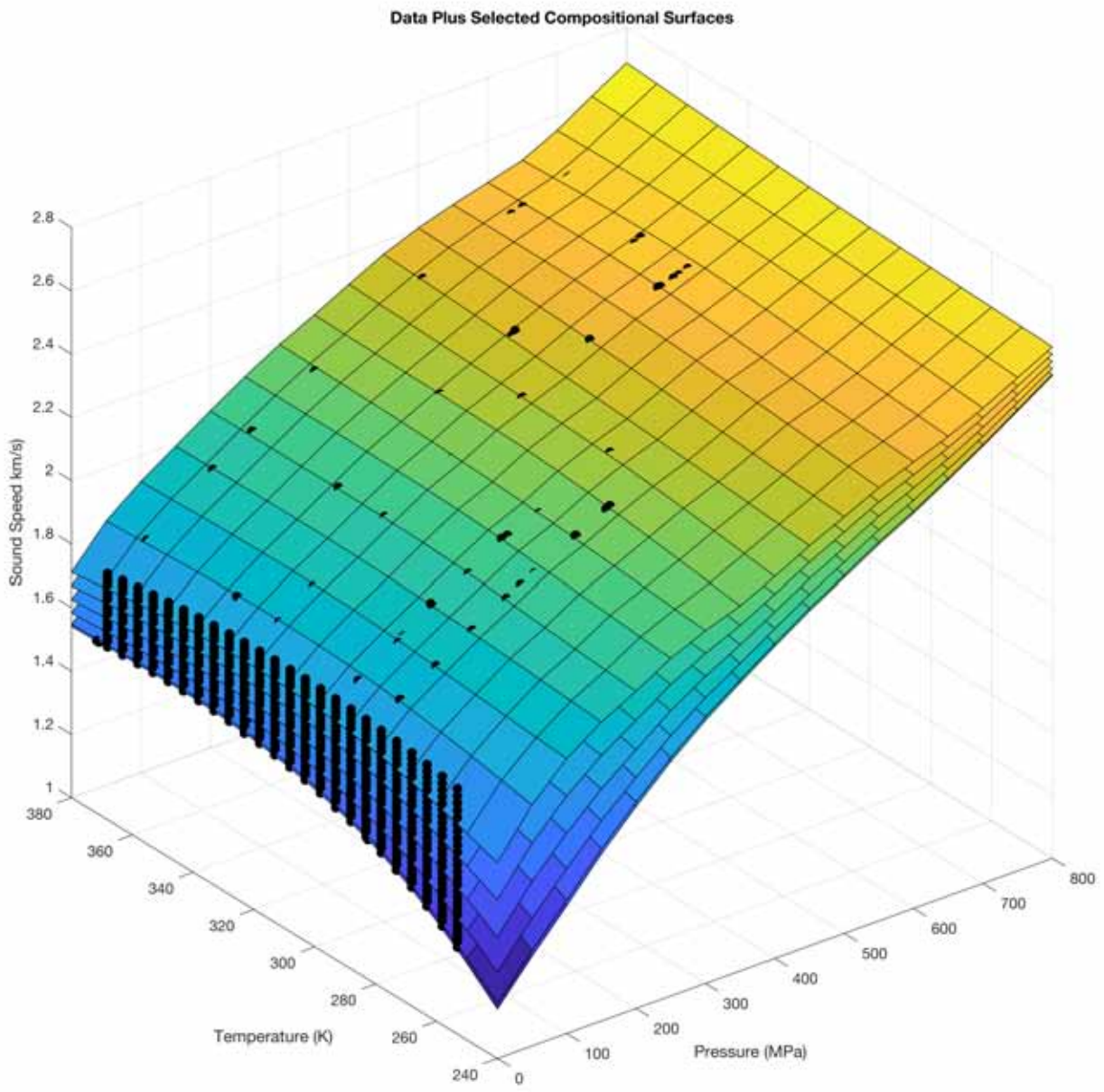
```
clear options
options.lam=50*[1 1 1];
% control points in P T and molality
Tc=240:20:380;
Pc=0:50:800;
mc=[0:.5:3];
options.Xc={Pc,Tc,mc};
```

Do the fit (note independent variables input as a matrix that is 3 wide

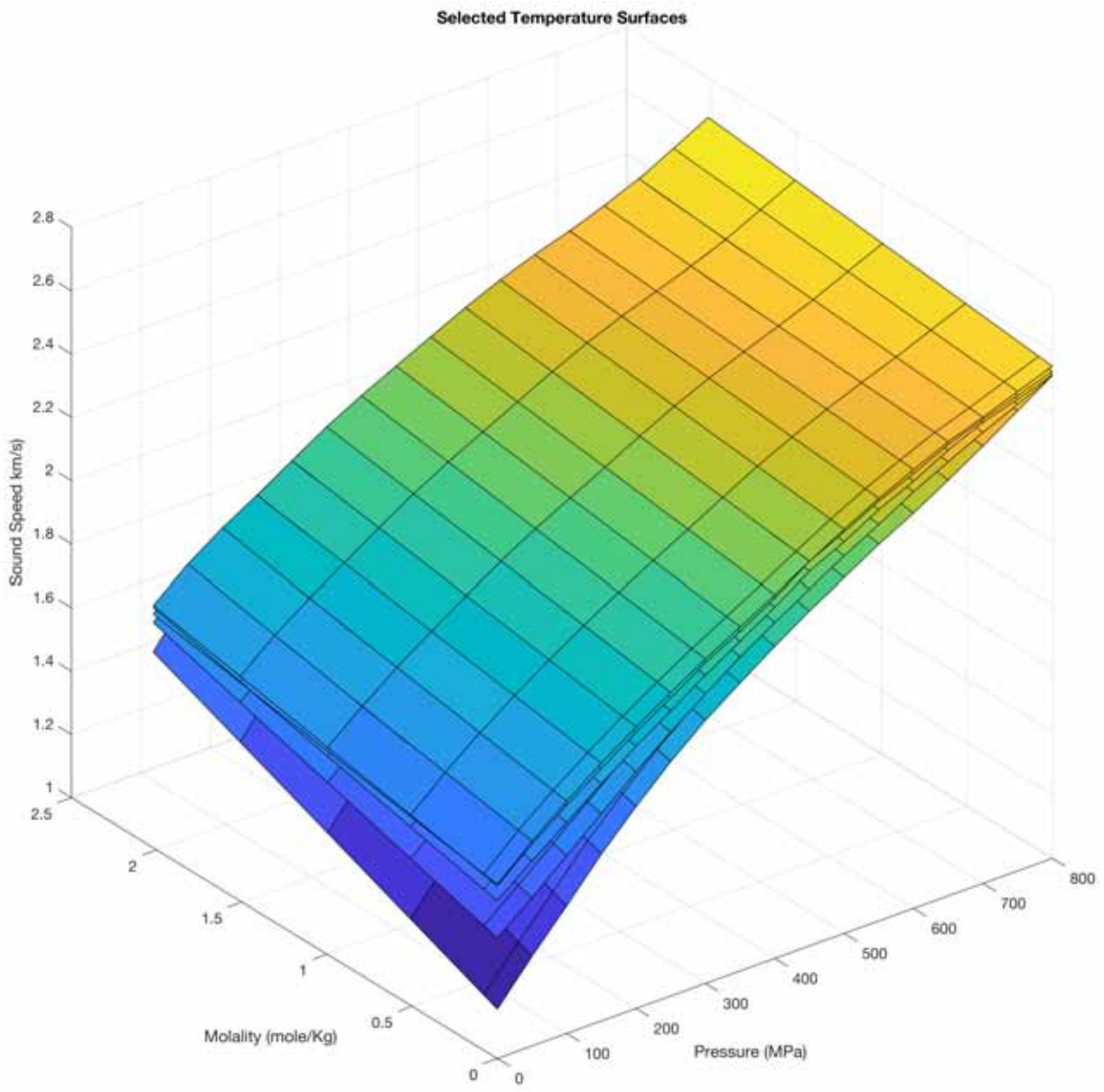
```
sp_ptm=spdft([P(:) T(:) m(:)],v(:),dv(:),options);
```

Calculate deviations of data from the surface

```
devs=[];
Pdevs=[];
ncount=0;

% evaluate fit and deviations at data points
vc=fnval(sp_ptm,[P(:) T(:) m(:)]');
devs=100*(v-vc)./v;

% set up a plotting grid:
mplt=[0 .08 .51 .997 1.51 2.02];
Pplt=0:50:800;
Tplt=240:10:380;
nP=length(Pplt);
nT=length(Tplt);
% calculate the sound speed surface on the plotting grid
vplt=fnval(sp_ptm,{Pplt,Tplt,mplt});
subplot(111)
clf
colormap gray
for i=1:6
    surf(Pplt,Tplt,squeeze(vplt(:,:,i))')
    hold on
end

plot3(P,T,v,'ko','MarkerFaceColor','k')
hold off
xlabel('Pressure (MPa)')
ylabel('Temperature (K)')
zlabel('Sound Speed km/s)')
title('Data Plus Selected Compositional Surfaces')
```

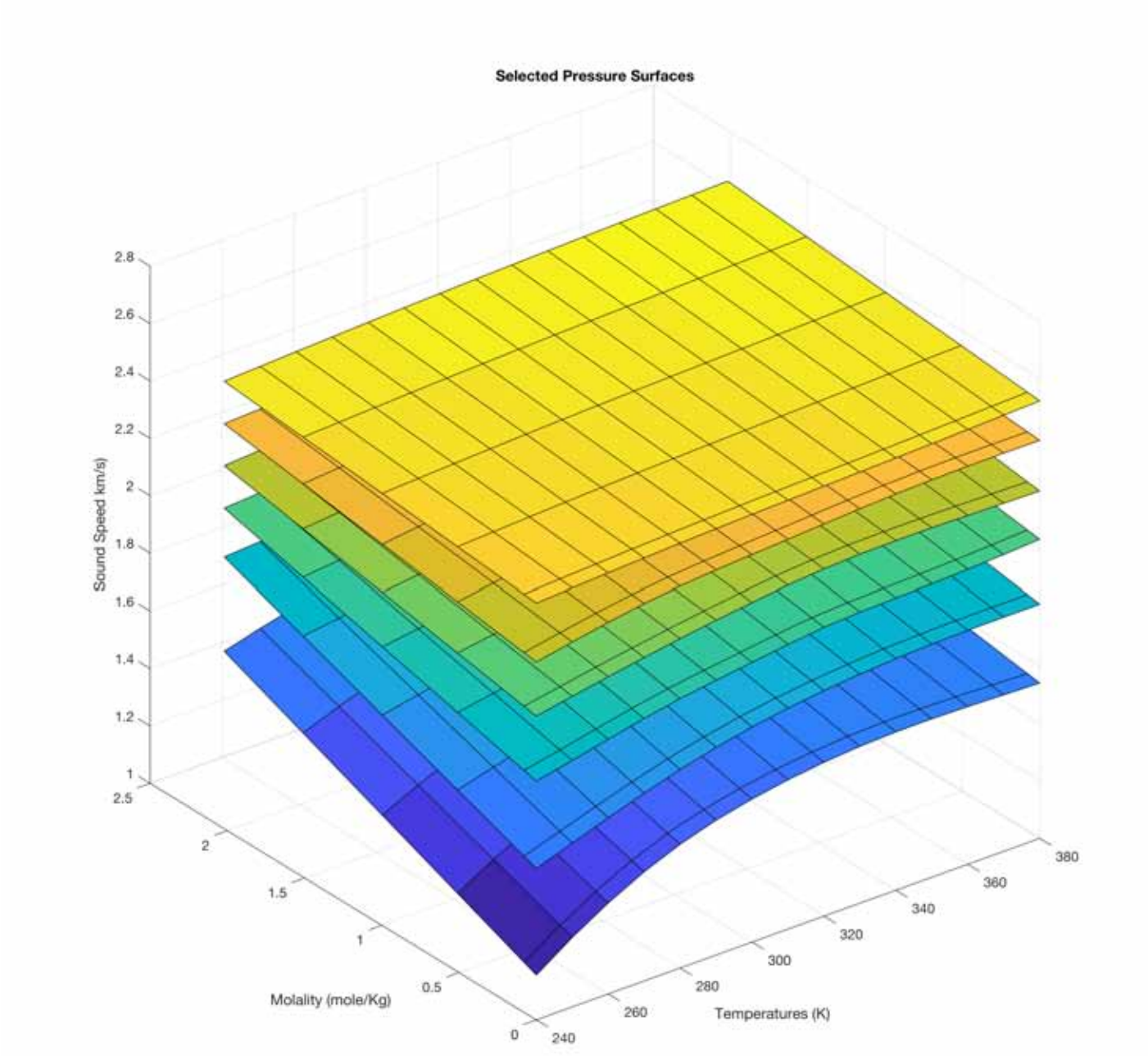



```
nt=length(Tplt);
for i=1:3:nt
    surf(Pplt,mplt,squeeze(vplt(:,i,:))')
    hold on
end
hold off
xlabel('Pressure (MPa)')
ylabel('Molality (mole/Kg)')
zlabel('Sound Speed km/s)')
title('Selected Temperature Surfaces')
```

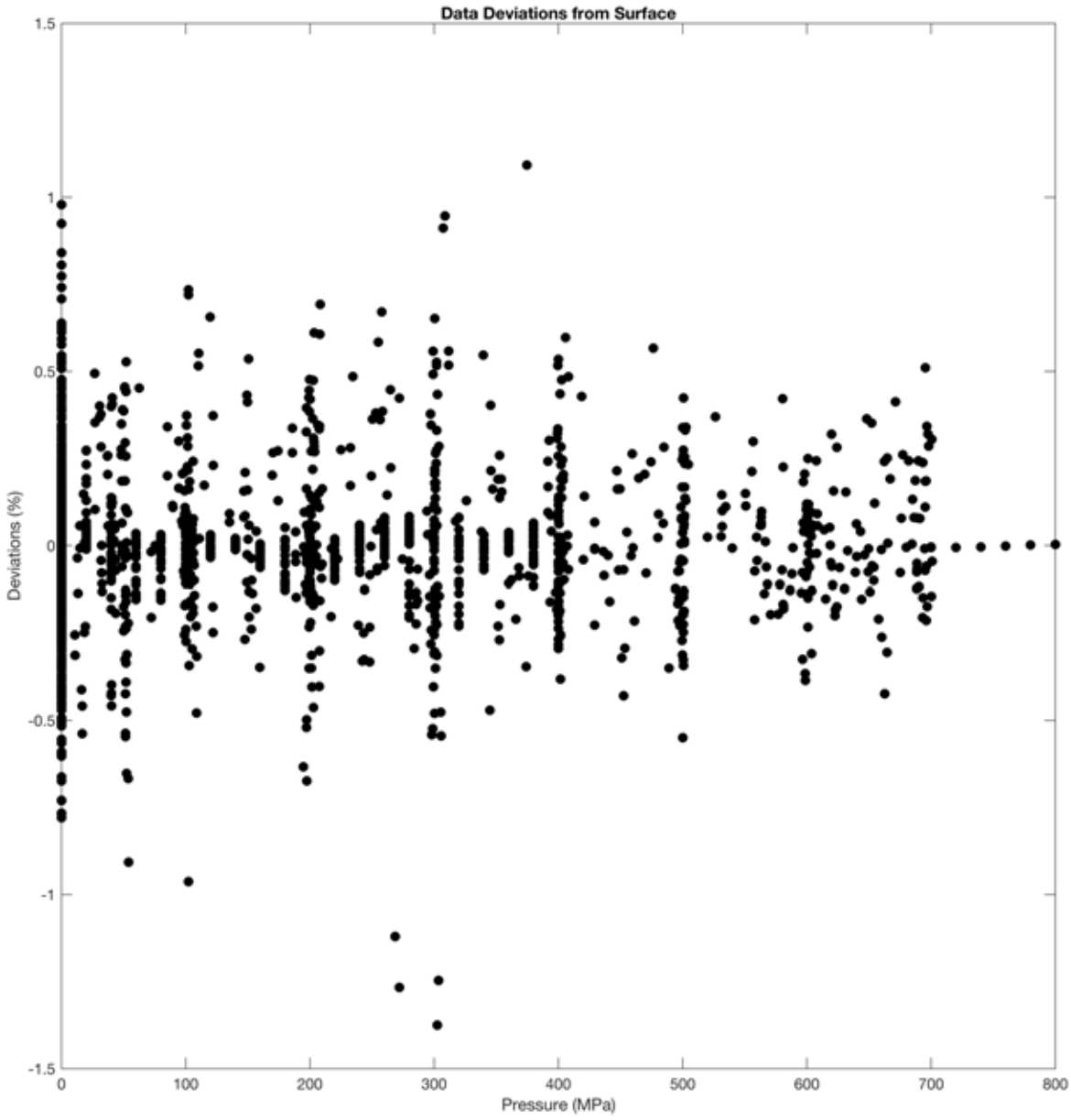



```
np=length(Pplt);
for i=1:3:np
    surf(Tplt,mplt,squeeze(vplt(i,:,:))')
    hold on
end
hold off
xlabel('Temperatures (K)')
ylabel('Molality (mole/Kg)')
zlabel('Sound Speed km/s)')
```

```
title('Selected Pressure Surfaces')
```




```
plot(P,devs,'ko','MarkerFaceColor','k')
xlabel('Pressure (MPa)')
ylabel('Deviations (%)')
title('Data Deviations from Surface')
```


Data Deviations from Surface
Deviations (%)
Pressure (MPa)
1.5
1
0.5
0
-0.5
-1
-1.5
0
100
200
300
400
500
600
700
800