

```

function
[FitResults,GOF,baseline,coeff,residual,xi,yi,BootResults]=peakfit(signal,center>window,NumPeaks,p
eakshape,extra,NumTrials,start,autozero,fixedparameters,plots,bipolar,minwidth,DELTA,clipheight)

%A command-line peak fitting program for time-series signals, written as a
%self-contained Matlab function in a single m-file. Uses a non-linear
%optimization algorithm to decompose a complex, overlapping-peak signal
%into its component parts. The objective is to determine whether your
%signal can be represented as the sum of fundamental underlying peaks
%shapes. Accepts signals of any length, including those with non-integer
%and non-uniform x-values. Fits any number of peaks of any of 33 curve
%shapes. This is a command line version, usable from a remote terminal. It
%is capable of making multiple trial fits with slightly different starting
%values (whose variability is controled by the 14th input argument) and
%taking the one with the lowest mean fit error (example 6). It can
%estimate the standard deviation of peak parameters from a single signal
%using the bootstrap method (example 10.(
%
%Version 7.1: March, 2015, adds peak shapes with three unconstrained
%iterated variables: 30=voigt (variable alpha), 31=ExpGaussian (variable
%time constant), 32=Pearson (variable shape factor), 34=Gaussian/
%Lorentzian blend (variable percent). See Examples 25-28 below.
%
%For more details, see
%http://terpconnect.umd.edu/~toh/spectrum/CurveFittingC.html and
%http://terpconnect.umd.edu/~toh/spectrum/InteractivePeakFitter.htm
%
%peakfit(signal    );(
%Performs an iterative least-squares fit of a single Gaussian peak to the

```

```

%data matrix "signal", which has x values in column 1 and Y values in
%column 2 (e.g. [x y(
%
%peakfit(signal,center>window;(
%Fits a single Gaussian peak to a portion of the matrix "signal". The
%portion is centered on the x-value "center" and has width "window" (in x
%units.(
%
%peakfit(signal,center>window,NumPeaks;(
" %NumPeaks" = number of peaks in the model (default is 1 if not
%specified). No limit to maximum number of peaks in version 3.1
%
%peakfit(signal,center>window,NumPeaks,peakshape ;(
" %peakshape" specifies the peak shape of the model: (1=Gaussian (default,(
=2 %Lorentzian, 3=logistic distribution, 4=Pearson, 5=exponentially
%broadened Gaussian; 6=equal-width Gaussians; 7=Equal-width Lorentzians;
=8 %exponentially broadened equal-width Gaussian, 9=exponential pulse,
=10 %up-sigmoid (logistic function), 11=Fixed-width Gaussian,
=12 %Fixed-width Lorentzian; 13=Gaussian/ Lorentzian blend; 14=Bifurcated
%Gaussian, 15=Breit-Wigner-Fano, 16=Fixed-position Gaussians;
=17 %Fixed-position Lorentzians; 18=exponentially broadened Lorentzian;
=19 %alpha function; 20=Voigt profile; 21=triangular; 22=multiple shapes;
=23 %down-sigmoid; 25=lognormal; 26=slope; 27=Gaussian first derivative ;
=28 %polynomial; 29=piecewise linear; 30=variable-alpha Voigt; 31=variable
%time constant ExpGaussian; 32=variable Pearson; 33=variable
%Gaussian/Lorentzian blend
%
%peakfit(signal,center>window,NumPeaks,peakshape,extra (

```

'%extra' specifies the value of 'extra', used only in the Voigt, Pearson, %exponentially broadened Gaussian, Gaussian/Lorentzian blend, and %bifurcated Gaussian and Lorentzian shapes to fine-tune the peak shape.

%

%peakfit(signal,center>window,NumPeaks,peakshape,extra,NumTrials;(

%Performs "NumTrials" trial fits and selects the best one (with lowest %fitting error). NumTrials can be any positive integer (default is 1.(

%

%peakfit(signal,center>window,NumPeaks,peakshape,extra,NumTrials,start(

%Specifies the first guesses vector "firstguess" for the peak positions %and widths. Must be expressed as a vector , in square brackets, e.g.

%start=[position1 width1 position2 width2[...

%

%peakfit(signal,center>window,NumPeaks,peakshape,extra,NumTrials,start,autozero (

'%autozero' sets the baseline correction mode:

%autozero=0 (default) does not subtract baseline from data segment ;

%autozero=1 interpolates a linear baseline from the edges of the data %segment and subtracts it from the signal (assumes that the %peak returns to the baseline at the edges of the signal ;(

%autozero=2 is like mode 1 except that it computes a quadratic curved baseline ;

%autozero=3 compensates for a flat baseline without reference to the %signal itself (best if the peak does not return to the %baseline at the edges of the signal.(

%

%peakfit(signal,center>window,NumPeaks,peakshape,extra,NumTrials,start...,

%autozero,fixedparameters(

'%fixedparameters' specifies fixed values for widths (shapes 10, 11) or %positions (shapes 16, 17(

```

%
%peakfit(signal,center>window,NumPeaks,peakshape,extra,NumTrials,start...,
%autozero,fixedparameters,plots(
' %plots' controls graphic plotting: 0=no plot; 1=plots draw as usual (default(
%
%peakfit(signal,center>window,NumPeaks,peakshape,extra,NumTrials,start...,
%autozero,fixedparameters,plots,bipolar(
' %bipolar' = 0 constrain peaks heights to be positions; 'bipolar' = 1
%allows positive and negative peak heights.
%
%
%peakfit(signal,center>window,NumPeaks,peakshape,extra,NumTrials,start...,
%autozero,fixedparameters,plots,bipolar,minwidth(
' %minwidth' sets the minimum allowed peak width. The default if not
%specified is equal to the x-axis interval. Must be a vector of minimum
%widths, one value for each peak, if the multiple peak shape is chosen ,
%as in example 17 and 18.
%
%
%peakfit(signal,center>window,NumPeaks,peakshape,extra,NumTrials,start...,
%autozero,fixedparameters,plots,bipolar,minwidth(
' %DELTA' (14th input argument) controls the restart variance when
%NumTrials>1. Default value is 1.0. Larger values give more variance.
%Version 5.8 and later only .
] %FitResults,FitError]=peakfit(signal,center>window...) Returns the
%FitResults vector in the order peak number, peak position, peak height,
%peak width, and peak area), and the FitError (the percent RMS
%difference between the data and the model in the selected segment of that
%data) of the best fit.
%
```

```

] %FitResults,LowestError,BestStart,xi,yi,BootResults]=peakfit(signal(...,
    %Prints out parameter error estimates for each peak (bootstrap method.(
%
%Optional output parameters
.1 %FitResults: a table of model peak parameters, one row for each peak,
    %listing Peak number, Peak position, Height, Width, and Peak area.
.2 %GOF: Goodness of Fit, a vector containing the rms fitting error of the
    %best trial fit and the R-squared (coefficient of determination.(
.3 %Baseline, the polynomial coefficients of the baseline in linear
    %and quadratic baseline modes (1 and 2) or the value of the constant
    %baseline in flat baseline mode.
.3 %coeff: Coefficients for the polynomial fit (shape 28 only; for other
    %shapes, coeff=0(
.5 %residual: the difference between the data and the best fit.
.6 %xi: vector containing 600 interpolated x-values for the model peaks .
.7 %yi: matrix containing the y values of each model peak at each xi .
    %Type plot(xi,yi(1,:)) to plot peak 1 or plot(xi,yi) to plot all peaks
.8 %BootResults: a table of bootstrap precision results for a each peak
    %and peak parameter.
%
%Example 1 :
<< %x=[0:1:10]';y=exp(-(x-5).^2);peakfit([x y([
%Fits exp(-x)^2 with a single Gaussian peak model.
%
    %Peak number Peak position Height Width Peak area
1,7725 1,665 1 5 1 %
%
<< %y=[0 1 2 4 6 7 6 4 2 1 0];x=1:length(y);(

```

```
<< %peakfit([x;y],length(y)/2,length(y),0,0,0,0,0,0(
%Fits small set of manually entered y data to a single Gaussian peak model.
```

```
%
```

```
%Example 2:
```

```
%x=[0:.01:10];y=exp(-(x-5).^2)+randn(size(x));peakfit([x;y](
```

```
%Measurement of very noisy peak with signal-to-noise ratio = 1.
```

```
%ans=
```

```
1.7716    1.7948    .9272    5.279    1    %
```

```
%
```

```
%Example 3:
```

```
%x=[0:.1:10];y=exp(-(x-5).^2)+.5*exp(-(x-3).^2)+.1*randn(size(x);(
```

```
%peakfit([x' y'],0,0,2(
```

```
%Fits a noisy two-peak signal with a double Gaussian model (NumPeaks=2.(
```

```
%ans=
```

```
.8654    1.642    .49489    3.001    1    %
```

```
1.7696    1.6597    1.016    4.9927    2    %
```

```
%
```

```
%Example 4:
```

```
<< %x=1:100;y=ones(size(x))./(1+(x-50).^2);peakfit(y,0,0,1,2(
```

```
%Fit Lorentzian (peakshape=2) located at x=50, height=1, width=2.
```

```
%ans=
```

```
3.1079    1.9971    .99974    5.    1    %
```

```
%
```

```
%Example 5 :
```

```
<< %x=[0:.005:1];y=humps(x);peakfit([x' y'],.3,.7,1,4,3;(
```

```
%Fits a portion of the humps function, 0.7 units wide and centered on
```

```
%x=0.3, with a single (NumPeaks=1) Pearson function (peakshape=4) with
```

```
%extra=3 (controls shape of Pearson function.(
```

```

%
%Example 6 :
<< %x=[0:.005:1];y=(humps(x)+humps(x-.13)).^3;smatrix=[x' y;[
] << %FitResults,FitError]=peakfit(smatrix,.4,.7,2,1,0,10(
%Creates a data matrix 'smatrix', fits a portion to a two-peak Gaussian
%model, takes the best of 10 trials. Returns FitResults and FitError.
%FitResults=
2/.125  ./.31.56  1          %e+006  0.11057  2.3689e+005
2/.24.3  ./.41529  2          %e+006  0.12033  2.8696e+005
%FitError=
1/.1899  %
%
%Example 7:
<< %peakfit([x' y'],.4,.7,2,1,0,10,[.3 .1 .5 .1];([
%As above, but specifies the first-guess position and width of the two
%peaks, in the order [position1 width1 position2 width2[
%
%Example 8: (Version 4 only(
%Demonstration of the four autozero modes, for a single Gaussian on flat
%baseline, with position=10, height=1, and width=1.66. Autozero mode
%is specified by the 9th input argument (0,1,2, or 3.(
<< %x=8:.05:12;y=1+exp(-(x-10).^2);(
] << %FitResults,FitError]=peakfit([x;y],0,0,1,1,0,1,0,0(
%Autozero=0 means to ignore the baseline (default mode if not specified(
%FitResults=
5/.7641  3/.612  1/.8561  1.  1  %
%FitError=
5/.387  %

```

```
] << %FitResults,FitError]=peakfit([x;y],0,0,1,1,0,1,0,1(
```

%Autozero=1 subtracts linear baseline from edge to edge.

%Does not work well because signal does not return to baseline at edges.

```
%FitResults=
```

```
1/5916 1/559 0.96153 9/9984 1 %
```

```
%FitError=
```

```
1/98.1 %
```

```
] << %FitResults,FitError]=peakfit([x;y],0,0,1,1,0,1,0,2(
```

%Autozero=1 subtracts quadratic baseline from edge to edge.

%Does not work well because signal does not return to baseline at edges.

```
%FitResults=
```

```
1/25.3 1/4384 0.81749 9/9996 1 %
```

```
%FitError=
```

```
1/82.4 %
```

%Autozero=3: Flat baseline mode, measures baseline by regression

```
] << %FitResults,Baseline,FitError]=peakfit([x;y],0,0,1,1,0,1,0,3(
```

```
%FitResults=
```

```
1/7645 1/6653 1/0.01 1. 1 %
```

```
%Baseline=
```

```
0.037056 %
```

```
%FitError=
```

```
0.99985 %
```

```
%
```

```
%Example 9:
```

```
%x=[0:1:10];y=exp(-(x-5).^2)+.5*exp(-(x-3).^2)+.1*randn(size(x);((
```

```
] %FitResults,FitError]=peakfit([x' y'],0,0,2,11,0,0,0,0,1.666(
```

%Same as example 3, fit with fixed-width Gaussian (shape 11), width=1.666

```
%
```



```

%Example 10: (Version 3 or later; Prints out parameter error estimates(
%x=0:.05:9;y=exp(-(x-5).^2)+.5*exp(-(x-3).^2)+.01*randn(1,length(x);(
] %FitResults,LowestError,residuals,xi,yi,BootstrapErrors]=peakfit([x;y],0,0,2,6,0,1,0,0,0;(
%
%Example 11: (Version 3.2 or later(
%x=[0:.005:1];y=humps(x);[FitResults,FitError]=peakfit([x' y'],0.54,0.93,2,13,15,10,0,0,0 (
%
%FitResults=
237.64    0.19131    19.041    0.30078    1    %
671999    0.33448    39.552    0.89788    2    %
%FitError =
0.34502    %
%Fits both peaks of the Humps function with a Gaussian/Lorentzian blend
) %shape 13) that is 15% Gaussian (Extra=15.(
%
%Example 12: (Version 3.2 or later(
<< %x=[0:.1:10];y=exp(-(x-4).^2)+.5*exp(-(x-5).^2)+.01*randn(size(x);(
] << %FitResults,FitError]=peakfit([x' y'],0,0,1,14,45,10,0,0,0 (
%FitResults=
276723    47.77    12315    42028    1    %
%FitError=
0.84461    %
%Fit a slightly asymmetrical peak with a bifurcated Gaussian (shape 14(
%
%Example 13: (Version 3.3 or later(
<< %x=[0:.1:10]';y=exp(-(x-5).^2);peakfit([x y],0,0,1,1,0,0,0,0,0(
%Example 1 without plotting (11th input argument = 0, default is 1(
%

```

```

%Example 14: (Version 3.9 or later)
%Exponentially broadened Lorentzian with position=9, height=1.
%x=[0:.1:20 ;[
%L=lorentzian(x,9,1;(
%L1=ExpBroaden(L',-10)+0.02.*randn(size(x;'((
] %FitResults,FitError]=peakfit([x;L1'],0,0,1,18,10(
%
%Example 15: Fitting the humps function with two Voigt profiles, flat
%baseline mode
] %FitResults,FitError]=peakfit(humps(0:.01:2),71,140,2,20,1.7,1,[31 4.7 90 8.8],3(
%FitResults=
255.01    47785    96762    31.047    1    %
1.895    88253    22935    9.09    2    %
%FitError=
.18051    %
%
%Example 16: (Version 4.3 or later) Set +/- mode to 1 (bipolar)
<< %x=[0:.1:10];y=exp(-(x-5).^2)-.5*exp(-(x-3).^2)+.1*randn(size(x;'((
<< %peakfit([x' y'],0,0,2,1,0,1,0,0,0,1,1(
%FitResults=
.9369-    162    .5433-    31636    1    %
19.29    18456    .96859    49487    2    %
%FitError=
82757    %
%
%Example 17: Version 5 or later. Fits humps function to a model consisting
%of one Pearson (shape=4, extra=3) and one Gaussian (shape=1), flat
%baseline mode=3, NumTrials=10.

```

```

%x=[0:.005:1.2];y=humps(x);[FitResults,FitError]=peakfit([x' y'],0,0,2,[2 1],[0 0([
%FitResults=
17/.85      .727892      84/671      .3.154      1      %
2,5399      .7.825      11,545      .88522      2      %
%Baseline=
.9.1      %
%FitError=
1.457      %
%
%Example 18: 5 peaks, 5 different shapes, all heights = 1, widths = 3.
%x=0:.1:60;
%y=modelpeaks2(x,[1 2 3 4 5],[1 1 1 1 1],[10 20 30 40 50....,[
*.).+([2.- 2 . . .],[3 3 3 3 3] %randn(size(x);(
%peakfit([x' y'],0,0,5,[1 2 3 4 5],[0 0 0 2 -20([
%
%Example 19: Minimum width constraint (13th input argument(
%x=1:30;y=gaussian(x,15,8)+.05*randn(size(x);((
%No constraint:
%peakfit([x;y],0,0,5,1,0,10,0,0,0,1,0,0;
%Widths constrained to values above 7:
%peakfit([x;y],0,0,5,1,0,10,0,0,0,1,0,7;
%
%Example 20: Noise test with peak height = RMS noise = 1.
%x=[-5:0.02:5];y=exp(-(x).^2)+randn(size(x));P=peakfit([x;y],0,0,1,1,0,10,0,0,0,1,1;
%
%Example 21: Gaussian peak on strong sloped straight-line baseline, 2-peak
%fit with variable-slope straight line (shape 26, peakfit version 6 only.(
<< %x=8:.05:12;y=x+exp(-(x-10).^2;

```

```
] << %FitResults,FitError]=peakfit([x;y],0,0,2,[1 26],[1 1],1,0(
```

```
%FitResults      =
```

```
1,7642    1,6651    1    1.    1    %  
4,1045    0,105    0,22297    4,485    2    %
```

```
%FitError =0.093
```

```
%
```

```
%Example 22: Segmented linear fit (Shape 29, peakfit version 6 only:(
```

```
%x=[0.9:.005:1.7];y=humps(x;(
```

```
%peakfit([x' y'],0,0,9,29,0,10,0,0,0,1,1(
```

```
%
```

```
%Example 23: Polynomial fit (Shape 27, peakfit version 6 only(
```

```
%x=[0.3:.005:1.7];y=humps(x);y=y+cumsum(y;(
```

```
%peakfit([x' y'],0,0,4,1,6,10,0,0,0,1,1(
```

```
%
```

```
%Example 24: Effect of quantization of independent (x) and dependent (y) variables.
```

```
%x=.5+round([2:.02:7.5]);y=exp(-(x-5).^2)+randn(size(x))/10;peakfit([x;y]([
```

```
%x=[2:.01:8];y=exp(-(x-5).^2)+.1.*randn(size(x));y=.1.*round(10.*y);peakfit([x;y]([
```

```
%
```

```
%Example 25: Variable-alpha Voigt functions, shape 30. (Version 7 and
```

```
%above only). FitResults has an added 6th column for the measured alphas
```

```
%of each peak.
```

```
%x=[0:.005:1.3];y=humps(x);[FitResults,FitError]=peakfit([x' y'],0,0,2,30,15,10(
```

```
%FitResults=
```

```
2,3194    24,563    0,035809    95,896    0,30154    1    %  
0,27815    7,2634    0,89194    19,336    2,3194    2    %
```

```
%FitError=
```

```
0,99886    0,84331    %
```

```
%
```

```

%Example 26: Variable time constant exponentially broadened Gaussian
%functions, shape 31. (Version 7 and above only). FitResults has an added
% %th column for the measured time constant of each peak.
] %FitResults,FitError]=peakfit(DataMatrix3,1860.5,364,2,31,32.9731,5,[1810 60 30 1910 60 30([
%FitResults=
32,781    119,01    60,169    1,8581    180,06    1    %
33,443    30,79    190,04    0,48491    32,781    2    %
%FitError=
0,99999    0,076651    %
%

```

```

%Example 27 Pearson variable shape, PeakShape=32,(Version 7 and above
%only). Requires modelpeaks2 function in path.

```

```

%x=1:1:30;y=modelpeaks2(x,[4 4],[1 1],[10 20],[5 5],[1 10];([
] %FitResults,FitError]=peakfit([x;y],0,0,2,32,10,5(
%

```

```

%Example 28 Gaussian/Lorentzian blend variable shape, PeakShape=33
) %Version 7 and above only). Requires modelpeaks2 function in path.

```

```

%x=1:1:30;y=modelpeaks2(x,[13 13],[1 1],[10 20],[3 3],[20 80];([
] %FitResults,FitError]=peakfit([x;y],0,0,2,33,0,5(
%

```

```

%Example 29: Fixed-position Gaussian (shape 16), positions=[3 5 .[

```

```

%x=0:1:10;y=exp(-(x-5).^2)+.5*exp(-(x-3).^2)+.1*randn(size(x);([
] %FitResults,FitError]=peakfit([x' y'],0,0,2,16,0,0,0,0,[3 5([
%

```

```

%Copyright (c) 2015, Thomas C. O'Haver

```

```

%Permission is hereby granted, free of charge, to any person obtaining a copy
%of this software and associated documentation files (the "Software"), to deal

```

%in the Software without restriction, including without limitation the rights

%to use, copy, modify, merge, publish, distribute, sublicense, and/or sell

%copies of the Software, and to permit persons to whom the Software is

%furnished to do so, subject to the following conditions:

%

%The above copyright notice and this permission notice shall be included in

%all copies or substantial portions of the Software.

%

%THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR

%IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,

%FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE

%AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER

%LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,

%OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN

%THE SOFTWARE.

%

global AA xxx PEAKHEIGHTS FIXEDPARAMETERS AUTOZERO delta BIPOLAR CLIPHEIGHT

format short g

format compact

warning off all

NumArgOut=nargout;

datasize=size(signal;(

if datasize(1)<datasize(2),signal=signal';end

datasize=size(signal;(

if datasize(2)==1, % Must be isignal(Y-vector(

 X=1:length(signal); % Create an independent variable vector

 Y=signal;

else

```

% Must be isignal(DataMatrix(
    X=signal(:,1); % Split matrix argument
    Y=signal(:,2);
end
X=reshape(X,1,length(X)); % Adjust X and Y vector shape to 1 x n (rather than n x 1(
Y=reshape(Y,1,length(Y));(
%If necessary, flip the data vectors so that X increases
if X(1)>X(length(X),((
    disp('X-axis flipped('
    X=flipr(X);(
    Y=flipr(Y);(
end

%Isolate desired segment from data set for curve fitting
if nargin==1 || nargin==2,center=(max(X)-min(X))/2;window=max(X)-min(X);end
%Y=Y-min(Y);(
xoffset=0;
n1=val2ind(X,center-window/2);(
n2=val2ind(X,center+window/2);(
if window==0,n1=1;n2=length(X);end
xx=X(n1:n2)-xoffset;
yy=Y(n1:n2);(
ShapeString='Gaussian;'
coeff=0;
CLIPHEIGHT=max(Y);(
LOGPLOT=0;
%Define values of any missing arguments

```

```
%  
)signal,center>window,NumPeaks,peakshape,extra,NumTrials,start,autozero,fixedparameters,plots,b  
ipolar,minwidth,DELTA(
```

```
switch nargin
```

```
case 1
```

```
    NumPeaks=1;  
  
    peakshape=1;  
  
    extra=0;  
  
    NumTrials=1;  
  
    xx=X;yy=Y;  
  
    start=calcstart(xx,NumPeaks,xoffset;(   
  
    AUTOZERO=0;  
  
    plots=1;  
  
    BIPOLAR=0;  
  
    MINWIDTH=xx(2)-xx(1);(  
  
    delta=1;  
  
    CLIPHEIGHT=max(Y;(
```

```
case 2
```

```
    NumPeaks=1;  
  
    peakshape=1;  
  
    extra=0;  
  
    NumTrials=1;  
  
    xx=signal;yy=center;  
  
    start=calcstart(xx,NumPeaks,xoffset;(   
  
    AUTOZERO=0;  
  
    plots=1;  
  
    BIPOLAR=0;  
  
    MINWIDTH=xx(2)-xx(1);(  

```



```
delta=1;
CLIPHEIGHT=max(Y;(
```

case 3

```
NumPeaks=1;
peakshape=1;
extra=0;
NumTrials=1;
start=calcstart(xx,NumPeaks,xoffset;(
AUTOZERO=0;
FIXEDPARAMETERS=0;
plots=1;
BIPOLAR=0;
MINWIDTH=xx(2)-xx(1);(
delta=1;
CLIPHEIGHT=max(Y;(
```

case 4 % Numpeaks specified in arguments

```
peakshape=1;
extra=0;
NumTrials=1;
start=calcstart(xx,NumPeaks,xoffset;(
AUTOZERO=0;
FIXEDPARAMETERS=0;
plots=1;
BIPOLAR=0;
MINWIDTH=xx(2)-xx(1);(
delta=1;
CLIPHEIGHT=max(Y;(
```

case 5 % Numpeaks, peakshape specified in arguments

```
extra=zeros(1,NumPeaks);  
NumTrials=1;  
start=calcstart(xx,NumPeaks,xoffset);  
AUTOZERO=0;  
FIXEDPARAMETERS=0;  
plots=1;  
BIPOLAR=0;  
MINWIDTH=zeros(size(peakshape))+xx(2)-xx(1);(  
delta=1;  
CLIPHEIGHT=max(Y;(
```

case 6

```
NumTrials=1;  
start=calcstart(xx,NumPeaks,xoffset);  
AUTOZERO=0;  
FIXEDPARAMETERS=0;  
plots=1;  
BIPOLAR=0;  
MINWIDTH=zeros(size(peakshape))+xx(2)-xx(1);(  
delta=1;
```

case 7

```
start=calcstart(xx,NumPeaks,xoffset);  
AUTOZERO=0;  
FIXEDPARAMETERS=0;  
plots=1;  
BIPOLAR=0;  
MINWIDTH=zeros(size(peakshape))+xx(2)-xx(1);(  
delta=1;
```

delta=1;

CLIPHEIGHT=max(Y;(

case 8

AUTOZERO=0;

FIXEDPARAMETERS=0;

plots=1;

BIPOLAR=0;

MINWIDTH=zeros(size(peakshape))+xx(2)-xx(1);((

delta=1;

CLIPHEIGHT=max(Y;(

case 9

AUTOZERO=autozero;

FIXEDPARAMETERS=0;

plots=1;

BIPOLAR=0;

MINWIDTH=zeros(size(peakshape))+xx(2)-xx(1);((

delta=1;

case 10

AUTOZERO=autozero;

FIXEDPARAMETERS=fixedparameters;

plots=1;

BIPOLAR=0;

MINWIDTH=zeros(size(peakshape))+xx(2)-xx(1);((

delta=1;

case 11

AUTOZERO=autozero;

FIXEDPARAMETERS=fixedparameters;

```
BIPOLAR=0;
MINWIDTH=zeros(size(peakshape))+xx(2)-xx(1);(
delta=1;
CLIPHEIGHT=max(Y;(
```

case 12

```
AUTOZERO=autozero;
FIXEDPARAMETERS=fixedparameters;
BIPOLAR=bipolar;
MINWIDTH=zeros(size(peakshape))+xx(2)-xx(1);(
delta=1;
CLIPHEIGHT=max(Y;(
```

case 13

```
AUTOZERO=autozero;
FIXEDPARAMETERS=fixedparameters;
BIPOLAR=bipolar;
MINWIDTH=minwidth;
delta=1;
```

case 14

```
AUTOZERO=autozero;
FIXEDPARAMETERS=fixedparameters;
BIPOLAR=bipolar;
MINWIDTH=minwidth;
delta=DELTA;
CLIPHEIGHT=max(Y;(
```

case 15

```
AUTOZERO=autozero;
FIXEDPARAMETERS=fixedparameters;
```

```

        BIPOLAR=bipolar;

        MINWIDTH=minwidth;

        delta=DELTA;

        CLIPHEIGHT=clipheight;

    otherwise
end % switch nargin

%Saturation Code, skips points greater than set maximum
if CLIPHEIGHT<max(Y,(
    apnt=1;
    for pnt=1:length(xx,(
        if yy(pnt)<CLIPHEIGHT,
            axx(apnt)=xx(pnt);(
            ayy(apnt)=yy(pnt);(
            apnt=apnt+1;
        end
    end
    xx=axx;yy=ayy;
end

%Default values for placeholder zeros1
if NumTrials==0;NumTrials=1;end
if isscalar(peakshape,(
else
    % disp('peakshape is vector;('
    shapvector=peakshape;
    NumPeaks=length(peakshape;(
    peakshape=22;

```

```

end

if peakshape==0;peakshape=1;end

if NumPeaks==0;NumPeaks=1;end

if start==0;start=calcstart(xx,NumPeaks,xoffset);end

if FIXEDPARAMETERS==0, FIXEDPARAMETERS=length(xx)/10;end

if peakshape==16;FIXEDPOSITIONS=fixedparameters;end

if peakshape==17;FIXEDPOSITIONS=fixedparameters;end

if AUTOZERO>3,AUTOZERO=3,end

if AUTOZERO<0,AUTOZERO=0,end

Heights=zeros(1,NumPeaks);

FitResults=zeros(NumPeaks,6);

% %Remove linear baseline from data segment if AUTOZERO==1

baseline=0;

bkgcoef=0;

bkgsz=round(length(xx)/10);

if bkgsz<2,bkgsz=2;end

lxx=length(xx);

if AUTOZERO==1, % linear autozero operation

    XX1=xx(1:round(lxx/bkgsz);(

    XX2=xx((lxx-round(lxx/bkgsz)):lxx);(

    Y1=yy(1:(round(length(xx)/bkgsz);((

    Y2=yy((lxx-round(lxx/bkgsz)):lxx);(

    bkgcoef=polyfit([XX1,XX2],[Y1,Y2],1); % Fit straight line to sub-group of points

    bkg=polyval(bkgcoef,xx);

    yy=yy-bkg;

end % if

```

```

if AUTOZERO==2, % Quadratic autozero operation

    XX1=xx(1:round(lxx/bkgsize);(
    XX2=xx((lxx-round(lxx/bkgsize)):lxx;(
    Y1=yy(1:round(length(xx)/bkgsize);(
    Y2=yy((lxx-round(lxx/bkgsize)):lxx;(
    bkgcoef=polyfit([XX1,XX2],[Y1,Y2],2); % Fit parabola to sub-group of points
    bkg=polyval(bkgcoef,xx;(
    yy=yy-bkg;
end % if autozero

```

```

PEAKHEIGHTS=zeros(1,NumPeaks;(
n=length(xx;(
newstart=start;
%Assign ShapStrings
switch peakshape(1(
    case 1
        ShapeString='Gaussian;'
    case 2
        ShapeString='Lorentzian;'
    case 3
        ShapeString='Logistic;'
    case 4
        ShapeString='Pearson;'
    case 5
        ShapeString='ExpGaussian;'
    case 6
        ShapeString='Equal width Gaussians;'

```

case 7

ShapeString='Equal width Lorentzians;'

case 8

ShapeString='Exp. equal width Gaussians;'

case 9

ShapeString='Exponential Pulse;'

case 10

ShapeString='Up Sigmoid (logistic function;'

case 23

ShapeString='Down Sigmoid (logistic function ;'

case 11

ShapeString='Fixed-width Gaussian;'

case 12

ShapeString='Fixed-width Lorentzian;'

case 13

ShapeString='Gaussian/Lorentzian blend;'

case 14

ShapeString='BiGaussian ;'

case 15

ShapeString='Breit-Wigner-Fano ;'

case 16

ShapeString='Fixed-position Gaussians;'

case 17

ShapeString='Fixed-position Lorentzians;'

case 18

ShapeString='Exp. Lorentzian;'

case 19

ShapeString='Alpha function;'

case 20

ShapeString='Voigt (equal alphas;'

case 21

ShapeString='triangular;'

case 22

ShapeString=num2str(shapesvector;'

case 24

ShapeString='Negative Binomial Distribution;'

case 25

ShapeString='Lognormal Distribution;'

case 26

ShapeString='slope;'

case 27

ShapeString='First derivative;'

case 28

ShapeString='Polynomial;'

case 29

ShapeString='Segmented linear;'

case 30

ShapeString='Voigt (variable alphas;'

case 31

ShapeString='ExpGaussian (var. time constant;'

case 32

ShapeString='Pearson (var. shape constant;'

case 33

ShapeString='Variable Gaussian/Lorentzian;'

```

    otherwise
end % switch peakshape

%Perform peak fitting for selected peak shape using fminsearch function
options = optimset('TolX',.001,'Display','off','MaxFunEvals',1000);(
LowestError=1000; % or any big number greater than largest error expected
FitParameters=zeros(1,NumPeaks.*2 );(
BestStart=zeros(1,NumPeaks.*2 );(
height=zeros(1,NumPeaks );(
bestmodel=zeros(size(yy);((

for k=1:NumTrials ,
%   StartMatrix(k,:)=newstart;

%   disp(['Trial number ' num2str(k) ] ) % optionally prints the current trial number as progress
indicator

    switch peakshape(1(
        case 1
            TrialParameters=fminsearch(@(lambda)(fitgaussian(lambda,xx,yy)),newstart,options);(
            for Peak=1:NumPeaks;
                if TrialParameters(2*Peak)<MINWIDTH,
                    TrialParameters(2*Peak)=MINWIDTH;
                end
            end
        case 2
            TrialParameters=fminsearch(@(lambda)(fitlorentzian(lambda,xx,yy)),newstart,options);(
            for Peak=1:NumPeaks;
                if TrialParameters(2*Peak)<MINWIDTH,
                    TrialParameters(2*Peak)=MINWIDTH;
                end
            end
        end
    end
end

```

```

        end

    end

case 3

    TrialParameters=fminsearch(@(lambda)(fitlogistic(lambda,xx,yy)),newstart,options;(

    for Peak=1:NumPeaks;

        if TrialParameters(2*Peak)<MINWIDTH,

            TrialParameters(2*Peak)=MINWIDTH;

        end

    end

case 4

    TrialParameters=fminsearch(@(lambda)(fitpearson(lambda,xx,yy,extra)),newstart,options;(

    for Peak=1:NumPeaks;

        if TrialParameters(2*Peak)<MINWIDTH,

            TrialParameters(2*Peak)=MINWIDTH;

        end

    end

case 5

    zxx=[zeros(size(xx)) xx zeros(size(xx);[ ((
    zyy=[zeros(size(yy)) yy zeros(size(yy);[ ((

    TrialParameters=fminsearch(@(lambda)(fitexpgaussian(lambda,zxx,zyy,-
extra)),newstart,options;(

    for Peak=1:NumPeaks;

        if TrialParameters(2*Peak)<MINWIDTH,

            TrialParameters(2*Peak)=MINWIDTH;

        end

    end

case 6

    cwnewstart(1)=newstart(1;(

```

```

for pc=2:NumPeaks,
    cwnewstart(pc)=newstart(2.*pc-1;(
end
cwnewstart(NumPeaks+1)=(max(xx)-min(xx))/5;
TrialParameters=fminsearch(@(lambda)(fitewgaussian(lambda,xx,yy)),cwnewstart,options;(
for Peak=1:NumPeaks;
    if TrialParameters(NumPeaks+1)<MINWIDTH,
        TrialParameters(NumPeaks+1)=MINWIDTH;
    end
end
case 7
cwnewstart(1)=newstart(1;(
for pc=2:NumPeaks,
    cwnewstart(pc)=newstart(2.*pc-1;(
end
cwnewstart(NumPeaks+1)=(max(xx)-min(xx))/5;

TrialParameters=fminsearch(@(lambda)(fitewlorentzian(lambda,xx,yy)),cwnewstart,options;(
for Peak=1:NumPeaks;
    if TrialParameters(NumPeaks+1)<MINWIDTH,
        TrialParameters(NumPeaks+1)=MINWIDTH;
    end
end
case 8
cwnewstart(1)=newstart(1;(
for pc=2:NumPeaks,
    cwnewstart(pc)=newstart(2.*pc-1;(
end

```

```

    cwnewstart(NumPeaks+1)=(max(xx)-min(xx))/5;

    TrialParameters=fminsearch(@(lambda)(fitexpwgaussian(lambda,xx,yy,-
extra)),cwnewstart,options;(
    for Peak=1:NumPeaks;
        if TrialParameters(NumPeaks+1)<MINWIDTH,
            TrialParameters(NumPeaks+1)=MINWIDTH;
        end
    end
case 9
    TrialParameters=fminsearch(@(lambda)(fitexpulse(lambda,xx,yy)),newstart,options;(
    for Peak=1:NumPeaks;
        if TrialParameters(2*Peak)<MINWIDTH,
            TrialParameters(2*Peak)=MINWIDTH;
        end
    end
case 10
    TrialParameters=fminsearch(@(lambda)(fitupsigmoid(lambda,xx,yy)),newstart,options;(
    for Peak=1:NumPeaks;
        if TrialParameters(2*Peak)<MINWIDTH,
            TrialParameters(2*Peak)=MINWIDTH;
        end
    end
case 23
    TrialParameters=fminsearch(@(lambda)(fitdownsigmoid(lambda,xx,yy)),newstart,options;(
    for Peak=1:NumPeaks;
        if TrialParameters(2*Peak)<MINWIDTH,
            TrialParameters(2*Peak)=MINWIDTH;
        end
end

```

```

end
case 11
fixedstart;[]=
for pc=1:NumPeaks,
    fixedstart(pc)=min(xx)+pc.*(max(xx)-min(xx))./(NumPeaks+1);
end
TrialParameters=fminsearch(@(lambda)(FitFWGaussian(lambda,xx,yy)),fixedstart,options;(
case 12
fixedstart;[]=
for pc=1:NumPeaks,
    fixedstart(pc)=min(xx)+pc.*(max(xx)-min(xx))./(NumPeaks+1);
end
TrialParameters=fminsearch(@(lambda)(FitFWLorentzian(lambda,xx,yy)),fixedstart,options;(
case 13
TrialParameters=fminsearch(@(lambda)(fitGL(lambda,xx,yy,extra)),newstart,options;(
for Peak=1:NumPeaks;
    if TrialParameters(2*Peak)<MINWIDTH,
        TrialParameters(2*Peak)=MINWIDTH;
    end
end
case 14
TrialParameters=fminsearch(@(lambda)(fitBiGaussian(lambda,xx,yy,extra)),newstart,options;(
for Peak=1:NumPeaks;
    if TrialParameters(2*Peak)<MINWIDTH,
        TrialParameters(2*Peak)=MINWIDTH;
    end
end
end

```

case 15

```
TrialParameters=fminsearch(@(lambda)(fitBWF(lambda,xx,yy,extra)),newstart,options);  
for Peak=1:NumPeaks;  
    if TrialParameters(2*Peak)<MINWIDTH,  
        TrialParameters(2*Peak)=MINWIDTH;  
    end  
end
```

case 16

```
fixedstart;[]=  
for pc=1:NumPeaks,  
    fixedstart(pc)=(max(xx)-min(xx))./(NumPeaks+1);  
    fixedstart(pc)=fixedstart(pc)+.1*(rand-.5).*fixedstart(pc);  
end  
TrialParameters=fminsearch(@(lambda)(FitFPGaussian(lambda,xx,yy)),fixedstart,options);  
for Peak=1:NumPeaks;  
    if TrialParameters(Peak)<MINWIDTH,  
        TrialParameters(Peak)=MINWIDTH;  
    end  
end
```

case 17

```
fixedstart;[]=  
for pc=1:NumPeaks,  
    fixedstart(pc)=(max(xx)-min(xx))./(NumPeaks+1);  
    fixedstart(pc)=fixedstart(pc)+.1*(rand-.5).*fixedstart(pc);  
end  
TrialParameters=fminsearch(@(lambda)(FitFPLorentzian(lambda,xx,yy)),fixedstart,options);  
for Peak=1:NumPeaks;
```

```

        if TrialParameters(Peak)<MINWIDTH,
            TrialParameters(Peak)=MINWIDTH;
        end
    end
case 18
    zxx=[zeros(size(xx)) xx zeros(size(xx);[ ((
    zyy=[ones(size(yy)).*yy(1) yy zeros(size(yy)).*yy(length(yy);[ ((
    TrialParameters=fminsearch(@(lambda)(fitexplorentzian(lambda,zxx,zyy,-
extra)),newstart,options;(
    for Peak=1:NumPeaks;
        if TrialParameters(2*Peak)<MINWIDTH,
            TrialParameters(2*Peak)=MINWIDTH;
        end
    end
case 19
    TrialParameters=fminsearch(@(lambda)(fitalphafunction(lambda,xx,yy)),newstart,options;(
    for Peak=1:NumPeaks;
        if TrialParameters(2*Peak)<MINWIDTH,
            TrialParameters(2*Peak)=MINWIDTH;
        end
    end
case 20
    TrialParameters=fminsearch(@(lambda)(fitvoigt(lambda,xx,yy,extra)),newstart,options;(
    for Peak=1:NumPeaks;
        if TrialParameters(2*Peak)<MINWIDTH,
            TrialParameters(2*Peak)=MINWIDTH;
        end
    end
end

```


case 21

```
TrialParameters=fminsearch(@(lambda)(fittriangular(lambda,xx,yy)),newstart,options;(  
for Peak=1:NumPeaks;  
    if TrialParameters(2*Peak)<MINWIDTH,  
        TrialParameters(2*Peak)=MINWIDTH;  
    end  
end
```

case 22

```
TrialParameters=fminsearch(@(lambda)(fitmultiple(lambda,xx,yy,shapvector,extra)),newstart,options;(  
for Peak=1:NumPeaks;  
    if TrialParameters(2*Peak)<MINWIDTH(Peak,(  
        TrialParameters(2*Peak)=MINWIDTH(Peak;(  
    end  
end
```

case 24

```
TrialParameters=fminsearch(@(lambda)(fitnbinpdf(lambda,xx,yy)),newstart,options;(  
for Peak=1:NumPeaks;  
    if TrialParameters(2*Peak)<MINWIDTH,  
        TrialParameters(2*Peak)=MINWIDTH;  
    end  
end
```

case 25

```
TrialParameters=fminsearch(@(lambda)(fitlognpdf(lambda,xx,yy)),newstart,options;(  
for Peak=1:NumPeaks;  
    if TrialParameters(2*Peak)<MINWIDTH,  
        TrialParameters(2*Peak)=MINWIDTH;
```

```

        end

    end

case 26

    TrialParameters=fminsearch(@(lambda)(fitlinslope(lambda,xx,yy)),polyfit(xx,yy,1),options;(
        coeff=TrialParameters;

case 27

    TrialParameters=fminsearch(@(lambda)(fitd1gauss(lambda,xx,yy)),newstart,options;(

case 28

    coeff=fitpolynomial(xx,yy,extra;(

    TrialParameters=coeff;

case 29

    cnewstart(1)=newstart(1;(

    for pc=2:NumPeaks,

        cnewstart(pc)=newstart(2.*pc-1)+(delta*(rand-.5)/50;(

    end

    TrialParameters=fminsearch(@(lambda)(fitsegmented(lambda,xx,yy)),cnewstart,options;(

case 30

    nn=max(xx)-min(xx;(

    start;[]=

    startpos=[nn/(NumPeaks+1):nn/(NumPeaks+1):nn-(nn/(NumPeaks+1))]+min(xx;(

    for marker=1:NumPeaks,

        markx=startpos(marker)+ xoffset;

        start=[start markx nn/5 extra;[

    end % for marker

    newstart=start;

    for parameter=1:3:3*NumPeaks,

        newstart(parameter)=newstart(parameter)*(1+randn/100;(

```

```

        newstart(parameter+1)=newstart(parameter+1)*(1+randn/20);
        newstart(parameter+2)=newstart(parameter+1)*(1+randn/20);
    end
    TrialParameters=fminsearch(@(lambda)(fitvoigtv(lambda,xx,yy)),newstart;
case 31
    nn=max(xx)-min(xx);
    start;[]=
    startpos=[nn/(NumPeaks+1):nn/(NumPeaks+1):nn-(nn/(NumPeaks+1))]+min(xx);
    for marker=1:NumPeaks,
        markx=startpos(marker)+ xoffset;
        start=[start markx nn/5 extra;[
    end % for marker
    newstart=start;
    for parameter=1:3:3*NumPeaks,
        newstart(parameter)=newstart(parameter)*(1+randn/100);
        newstart(parameter+1)=newstart(parameter+1)*(1+randn/20);
        newstart(parameter+2)=newstart(parameter+1)*(1+randn/20);
    end
%    newstart=newstart
    zxx=[zeros(size(xx)) xx zeros(size(xx);[ ((
    zyy=[ones(size(yy)).*yy(1) yy zeros(size(yy)).*yy(length(yy);[ ((
    TrialParameters=fminsearch(@(lambda)(fitexpgaussianv(lambda,zxx,zyy)),newstart;
case 32
    nn=max(xx)-min(xx);
    start;[]=
    startpos=[nn/(NumPeaks+1):nn/(NumPeaks+1):nn-(nn/(NumPeaks+1))]+min(xx);
    for marker=1:NumPeaks,

```

```

        markx=startpos(marker)+ xoffset;

        start=[start markx nn/5 extra;]
end % for marker

newstart=start;

for parameter=1:3:3*NumPeaks,

    newstart(parameter)=newstart(parameter)*(1+randn/100);(
    newstart(parameter+1)=newstart(parameter+1)*(1+randn/20);(
    newstart(parameter+2)=newstart(parameter+1)*(1+randn/20);(

end

%     newstart=newstart

    TrialParameters=fminsearch(@(lambda)(fitpearsonv(lambda,xx,yy)),newstart;(

case 33

    nn=max(xx)-min(xx);(
start;[]=
startpos=[nn/(NumPeaks+1):nn/(NumPeaks+1):nn-(nn/(NumPeaks+1))]+min(xx);(

for marker=1:NumPeaks,

    markx=startpos(marker)+ xoffset;

    start=[start markx nn/5 extra;]

end % for marker

newstart=start;

for parameter=1:3:3*NumPeaks,

    newstart(parameter)=newstart(parameter)*(1+randn/100);(
    newstart(parameter+1)=newstart(parameter+1)*(1+randn/20);(
    newstart(parameter+2)=newstart(parameter+1)*(1+randn/20);(

end

%     newstart=newstart

    TrialParameters=fminsearch(@(lambda)(fitGLv(lambda,xx,yy)),newstart;(

```

```

        otherwise
    end % switch peakshape

%Construct model from Trial parameters
A=zeros(NumPeaks,n);
for m=1:NumPeaks,
    switch peakshape(1(
        case 1
            A(m,:)=gaussian(xx,TrialParameters(2*m-1),TrialParameters(2*m);((
        case 2
            A(m,:)=lorentzian(xx,TrialParameters(2*m-1),TrialParameters(2*m);((
        case 3
            A(m,:)=logistic(xx,TrialParameters(2*m-1),TrialParameters(2*m);((
        case 4
            A(m,:)=pearson(xx,TrialParameters(2*m-1),TrialParameters(2*m),extra;((
        case 5
            A(m,:)=expgaussian(xx,TrialParameters(2*m-1),TrialParameters(2*m),-extra;('
        case 6
            A(m,:)=gaussian(xx,TrialParameters(m),TrialParameters(NumPeaks+1);((
        case 7
            A(m,:)=lorentzian(xx,TrialParameters(m),TrialParameters(NumPeaks+1);((
        case 8
            A(m,:)=expgaussian(xx,TrialParameters(m),TrialParameters(NumPeaks+1),-extra;('
        case 9
            A(m,:)=exppulse(xx,TrialParameters(2*m-1),TrialParameters(2*m);((
        case 10
            A(m,:)=upsigmoid(xx,TrialParameters(2*m-1),TrialParameters(2*m);((

```

case 11

$A(m,:) = \text{gaussian}(xx, \text{TrialParameters}(m), \text{FIXEDPARAMETERS});$

case 12

$A(m,:) = \text{lorentzian}(xx, \text{TrialParameters}(m), \text{FIXEDPARAMETERS});$

case 13

$A(m,:) = \text{GL}(xx, \text{TrialParameters}(2*m-1), \text{TrialParameters}(2*m), \text{extra});$

case 14

$A(m,:) = \text{BiGaussian}(xx, \text{TrialParameters}(2*m-1), \text{TrialParameters}(2*m), \text{extra});$

case 15

$A(m,:) = \text{BWF}(xx, \text{TrialParameters}(2*m-1), \text{TrialParameters}(2*m), \text{extra});$

case 16

$A(m,:) = \text{gaussian}(xx, \text{FIXEDPOSITIONS}(m), \text{TrialParameters}(m));$

case 17

$A(m,:) = \text{lorentzian}(xx, \text{FIXEDPOSITIONS}(m), \text{TrialParameters}(m));$

case 18

$A(m,:) = \text{explorentzian}(xx, \text{TrialParameters}(2*m-1), \text{TrialParameters}(2*m), -\text{extra});$

case 19

$A(m,:) = \text{alphafunction}(xx, \text{TrialParameters}(2*m-1), \text{TrialParameters}(2*m));$

case 20

$A(m,:) = \text{voigt}(xx, \text{TrialParameters}(2*m-1), \text{TrialParameters}(2*m), \text{extra});$

case 21

$A(m,:) = \text{triangular}(xx, \text{TrialParameters}(2*m-1), \text{TrialParameters}(2*m));$

case 22

$A(m,:) = \text{peakfunction}(\text{shapeselector}(m), xx, \text{TrialParameters}(2*m-1), \text{TrialParameters}(2*m), \text{extra});$

case 23

$A(m,:) = \text{downsigmoid}(xx, \text{TrialParameters}(2*m-1), \text{TrialParameters}(2*m));$

case 24

```

        A(m,:)=nbinpdf(xx,TrialParameters(2*m-1),TrialParameters(2*m);((
case 25
        A(m,:)=lognormal(xx,TrialParameters(2*m-1),TrialParameters(2*m);((
case 26
        A(m,:)=linslope(xx,TrialParameters(2*m-1),TrialParameters(2*m);((
case 27
        A(m,:)=d1gauss(xx,TrialParameters(2*m-1),TrialParameters(2*m);((
case 28
        A(m,:)=polynomial(xx,coeff;(
case 29
        A(m,:)=segmented(xx,yy,PEAKHEIGHTS;(
case 30
        A(m,:)=voigt(xx,TrialParameters(3*m-2),TrialParameters(3*m-1),TrialParameters(3*m) ;((
case 31
        A(m,:)=expgaussian(xx,TrialParameters(3*m-2),TrialParameters(3*m-1),-
TrialParameters(3*m) ;((
case 32
        A(m,:)=pearson(xx,TrialParameters(3*m-2),TrialParameters(3*m-1),TrialParameters(3*m) ;((
case 33
        A(m,:)=GL(xx,TrialParameters(3*m-2),TrialParameters(3*m-1),TrialParameters(3*m) ;((
        otherwise
end % switch
for parameter=1:2:2*NumPeaks,
    newstart(parameter)=newstart(parameter)*(1+delta*(rand-.5)/500;(
    newstart(parameter+1)=newstart(parameter+1)*(1+delta*(rand-.5)/100;(
end
end % for NumPeaks

```

```

%Multiplies each row by the corresponding amplitude and adds them up
if peakshape(1)==29, % Segmented linear
    model=segmented(xx,yy,PEAKHEIGHTS;(
    TrialParameters=PEAKHEIGHTS;
    Heights=ones(size(PEAKHEIGHTS;((
else
    if AUTOZERO==3,
        baseline=PEAKHEIGHTS(1;(
        Heights=PEAKHEIGHTS(2:1+NumPeaks;(
        model=Heights'*A+baseline;
    else
        %size(PEAKHEIGHTS) % error check
        %size(A(
        model=PEAKHEIGHTS'*A;
        Heights=PEAKHEIGHTS;
        baseline=0;
    end
end

if peakshape(1)==28, % polynomial;
    model=polynomial(xx,coeff;(
    TrialParameters=PEAKHEIGHTS;
    Heights=ones(size(PEAKHEIGHTS;((
end

%Compare trial model to data segment and compute the fit error
    MeanFitError=100*norm(yy-model)./(sqrt(n)*max(yy;((

% Take only the single fit that has the lowest MeanFitError

```



```

if MeanFitError<LowestError ,
    if min(Heights)>=-BIPOLAR*10^100, % Consider only fits with positive peak heights
        LowestError=MeanFitError; % Assign LowestError to the lowest MeanFitError
        FitParameters=TrialParameters; % Assign FitParameters to the fit with the lowest
MeanFitError
        BestStart=newstart; % Assign BestStart to the start with the lowest MeanFitError
        height=Heights; % Assign height to the PEAKHEIGHTS with the lowest MeanFitError
        bestmodel=model; % Assign bestmodel to the model with the lowest MeanFitError
    end % if min(PEAKHEIGHTS)>0
end % if MeanFitError<LowestError
%ErrorVector(k)=MeanFitError;
end % for k (NumTrials(
    Rsquared=1-(norm(yy-bestmodel)./norm(yy-mean(yy)));
    SStot=sum((yy-mean(yy)).^2);
    SSres=sum((yy-bestmodel).^2);
    Rsquared=1-(SSres./SStot);
    GOF=[LowestError Rsquared];
%Uncomment following 4 lines to monitor trail fit starts and errors.
%StartMatrix=StartMatrix;
%ErrorVector=ErrorVector;
%matrix=[StartMatrix ErrorVector]'
%std(StartMatrix(
%Construct model from best-fit parameters
AA=zeros(NumPeaks,600);
xxx=linspace(min(xx),max(xx),600);
%xxx=linspace(min(xx)-length(xx),max(xx)+length(xx),200);
for m=1:NumPeaks,
    switch peakshape(1(

```

case 1

```
AA(m,:)=gaussian(xxx,FitParameters(2*m-1),FitParameters(2*m);((
```

case 2

```
AA(m,:)=lorentzian(xxx,FitParameters(2*m-1),FitParameters(2*m);((
```

case 3

```
AA(m,:)=logistic(xxx,FitParameters(2*m-1),FitParameters(2*m);((
```

case 4

```
AA(m,:)=pearson(xxx,FitParameters(2*m-1),FitParameters(2*m),extra;(
```

case 5

```
AA(m,:)=expgaussian(xxx,FitParameters(2*m-1),FitParameters(2*m),-  
extra*length(xxx)./length(xx);'((
```

case 6

```
AA(m,:)=gaussian(xxx,FitParameters(m),FitParameters(NumPeaks+1);((
```

case 7

```
AA(m,:)=lorentzian(xxx,FitParameters(m),FitParameters(NumPeaks+1);((
```

case 8

```
AA(m,:)=expgaussian(xxx,FitParameters(m),FitParameters(NumPeaks+1),-  
extra*length(xxx)./length(xx);'((
```

case 9

```
AA(m,:)=exppulse(xxx,FitParameters(2*m-1),FitParameters(2*m) ;((
```

case 10

```
AA(m,:)=upsigmoid(xxx,FitParameters(2*m-1),FitParameters(2*m) ;((
```

case 11

```
AA(m,:)=gaussian(xxx,FitParameters(m),FIXEDPARAMETERS;(
```

case 12

```
AA(m,:)=lorentzian(xxx,FitParameters(m),FIXEDPARAMETERS;(
```

case 13

```
AA(m,:)=GL(xxx,FitParameters(2*m-1),FitParameters(2*m),extra;(
```

case 14

```
AA(m,:)=BiGaussian(xxx,FitParameters(2*m-1),FitParameters(2*m),extra ;(
```

case 15

```
AA(m,:)=BWF(xxx,FitParameters(2*m-1),FitParameters(2*m),extra ;(
```

case 16

```
AA(m,:)=gaussian(xxx,FIXEDPOSITIONS(m),FitParameters(m;((
```

case 17

```
AA(m,:)=lorentzian(xxx,FIXEDPOSITIONS(m),FitParameters(m;((
```

case 18

```
AA(m,:)=explorentzian(xxx,FitParameters(2*m-1),FitParameters(2*m),-  
extra*length(xxx)./length(xx;'((
```

case 19

```
AA(m,:)=alphafunction(xxx,FitParameters(2*m-1),FitParameters(2*m;((
```

case 20

```
AA(m,:)=voigt(xxx,FitParameters(2*m-1),FitParameters(2*m),extra ;(
```

case 21

```
AA(m,:)=triangular(xxx,FitParameters(2*m-1),FitParameters(2*m;((
```

case 22

```
AA(m,:)=peakfunction(shapesvector(m),xxx,FitParameters(2*m-  
1),FitParameters(2*m),extra(m ;((
```

case 23

```
AA(m,:)=downsigmoid(xxx,FitParameters(2*m-1),FitParameters(2*m ;((
```

case 24

```
AA(m,:)=nbinpdf(xxx,FitParameters(2*m-1),FitParameters(2*m ;((
```

case 25

```
AA(m,:)=lognormal(xxx,FitParameters(2*m-1),FitParameters(2*m ;((
```

case 26

```
AA(m,:)=linslope(xxx,FitParameters(2*m-1),FitParameters(2*m ;((
```

```

case 27
    AA(m,:)=d1gauss(xxx,FitParameters(2*m-1),FitParameters(2*m) ;((
case 28
    AA(m,:)=polynomial(xxx,coeff;(
case 29
case 30
    AA(m,:)=voigt(xxx,FitParameters(3*m-2),FitParameters(3*m-1),FitParameters(3*m) ;((
case 31
    AA(m,:)=expgaussian(xxx,FitParameters(3*m-2),FitParameters(3*m-1),-
FitParameters(3*m)*length(xxx)./length(xx) ;((
case 32
    AA(m,:)=pearson(xxx,FitParameters(3*m-2),FitParameters(3*m-1),FitParameters(3*m) ;((
case 33
    AA(m,:)=GL(xxx,FitParameters(3*m-2),FitParameters(3*m-1),FitParameters(3*m) ;((
    otherwise
end % switch
end % for NumPeaks

%Multiplies each row by the corresponding amplitude and adds them up
if peakshape(1)==29, % Segmented linear
    mmodel=segmented(xx,yy,PEAKHEIGHTS;(
    baseline=0;
else
    heightsize=size(height;('
    AAsize=size(AA;(
    if heightsize(2)==AAsize(1,(
        mmodel=height'*AA+baseline;
    else

```

```

        mmodel=height*AA+baseline;
    end
end
%Top half of the figure shows original signal and the fitted model.
if plots,
    subplot(2,1,1);plot(xx+xoffset,yy,'b. '); % Plot the original signal in blue dots
    hold on
end
if peakshape(1)==28, % Polynomial
    yi=polynomial(xxx,coeff);
else
    for m=1:NumPeaks,
        if plots, plot(xxx+xoffset,height(m)*AA(m,:)+baseline,'g'),end % Plot the individual component
        peaks in green lines
        area(m)=trapz(xxx+xoffset,height(m)*AA(m,:)); % Compute the area of each component peak
        using trapezoidal method
        yi(m,:)=height(m)*AA(m,:); % Place y values of individual model peaks into matrix yi
    end
end
xi=xxx+xoffset; % Place the x-values of the individual model peaks into xi

if plots,
    % Mark starting peak positions with vertical dashed magenta lines
    if peakshape(1)==16 || peakshape(1)==17
    else
        if peakshape(1)==29, % Segmented linear
            subplot(2,1,1);plot(['PEAKHEIGHTS' PEAKHEIGHTS'],[0 max(yy)],'m('-
        else

```

```

    for marker=1:NumPeaks,
        markx=BestStart((2*marker)-1);(
            subplot(2,1,1);plot([markx+xoffset markx+xoffset],[0 max(yy)],'m('--
        end % for
    end
end % if peakshape

% Plot the total model (sum of component peaks) in red lines
if peakshape(1)==29, % Segmented linear
    mmodel=segmented(xx,yy,PEAKHEIGHTS;(
        plot(xx+xoffset,mmodel,'r ');('
else
    plot(xxx+xoffset,mmodel,'r ');('
end
hold off;
lly=min(yy);(
uyy=max(yy)+(max(yy)-min(yy))/10;
if BIPOLAR,
    axis([min(xx) max(xx) lly uyy];([
    ylabel('+ - mode('
else
    axis([min(xx) max(xx) 0 uyy];([
    ylabel('+ mode('
end
switch AUTOZERO,
    case 0
        title(['peakfit.m Version 7 No baseline correction(['

```

case 1

title(['peakfit.m Version 7 Linear baseline subtraction(['

case 2

title(['peakfit.m Version 7 Quadratic subtraction baseline(['

case 3

title(['peakfit.m Version 7 Flat baseline correction(['

end

switch peakshape(1(

case {4,20{

xlabel(['Peaks = ' num2str(NumPeaks) ' Shape = ' ShapeString ' Min. Width = '
num2str(MINWIDTH) ' Shape Constant = ' num2str(extra) ' Error = '
num2str(round(1000*LowestError)/1000) '% R2 = ' num2str(round(100000*Rsquared)/100000([(

case {5,8,18{

xlabel(['Peaks = ' num2str(NumPeaks) ' Shape = ' ShapeString ' Min. Width = '
num2str(MINWIDTH) ' Time Constant = ' num2str(extra) ' Error = '
num2str(round(1000*LowestError)/1000) '% R2 = ' num2str(round(100000*Rsquared)/100000([(

case 13

xlabel(['Peaks = ' num2str(NumPeaks) ' Shape = ' ShapeString ' Min. Width = '
num2str(MINWIDTH) ' % Gaussian = ' num2str(extra) ' Error = '
num2str(round(1000*LowestError)/1000) '% R2 = ' num2str(round(100000*Rsquared)/100000([(

case {14,15,22{

xlabel(['Peaks = ' num2str(NumPeaks) ' Shape = ' ShapeString ' Min. Width = '
num2str(MINWIDTH) ' extra = ' num2str(extra) ' Error = '
num2str(round(1000*LowestError)/1000) '% R2 = ' num2str(round(100000*Rsquared)/100000([(

case 28

xlabel(['Shape = ' ShapeString ' Order = ' num2str(extra) ' Error = '
num2str(round(1000*LowestError)/1000) '% R2 = ' num2str(round(1000*LowestError)/1000([(

otherwise

if peakshape(1)==29, % Segmented linear

```

        xlabel(['Breakpoints = ' num2str(NumPeaks) ' Shape = ' ShapeString ' Error = '
num2str(round(1000*LowestError)/1000) '% R2 = ' num2str(round(100000*Rsquared)/100000( [ (
    else
        xlabel(['Peaks = ' num2str(NumPeaks) ' Shape = ' ShapeString ' Min. Width = '
num2str(MINWIDTH) ' Error = ' num2str(round(1000*LowestError)/1000) '% R2 = '
num2str(round(100000*Rsquared)/100000( [ (
    end % if peakshape(1)==29
end % switch peakshape(1(

% Bottom half of the figure shows the residuals and displays RMS error
% between original signal and model
residual=yy-bestmodel;
subplot(2,1,2);plot(xx+xoffset,residual,'r'.
axis([min(xx)+xoffset max(xx)+xoffset min(residual) max(residual);((
xlabel('Residual Plot('
if NumTrials>1,
    title(['Best of ' num2str(NumTrials) ' fits(['
else
    title(['Single fit(['
end
end % if plots

%Put results into a matrix FitResults, one row for each peak, showing peak index number,
%position, amplitude, and width.
FitResults=zeros(NumPeaks,6;(
switch peakshape(1,(
    case {6,7,8}, % equal-width peak models only
        for m=1:NumPeaks,
            if m==1,

```



```

        FitResults=[round(m) FitParameters(m)+xoffset height(m)
abs(FitParameters(NumPeaks+1)) area(m);[(
    else
        FitResults=[FitResults ; round(m) FitParameters(m)+xoffset height(m)
abs(FitParameters(NumPeaks+1)) area(m);[(
    end
end
case {11,12}, % Fixed-width shapes only
    for m=1:NumPeaks,
        if m==1,
            FitResults=[round(m) FitParameters(m)+xoffset height(m) FIXEDPARAMETERS area(m);[(
        else
            FitResults=[FitResults ; round(m) FitParameters(m)+xoffset height(m) FIXEDPARAMETERS
area(m);[(
        end
    end
case {16,17}, % Fixed-position shapes only
    for m=1:NumPeaks,
        if m==1,
            FitResults=[round(m) FIXEDPOSITIONS(m) height(m) FitParameters(m) area(m);[(
        else
            FitResults=[FitResults ; round(m) FIXEDPOSITIONS(m) height(m) FitParameters(m)
area(m);[(
        end
    end
case 28, % Simple polynomial fit
    FitResults=PEAKHEIGHTS;
case 29, % Segmented linear fit
    FitResults=PEAKHEIGHTS;

```

```

case {30,31,32,33} % Special case of shapes with 3 iterated variables

    for m=1:NumPeaks,
        if m==1,
            FitResults=[round(m) FitParameters(3*m-2) height(m) abs(FitParameters(3*m-1)) area(m)
FitParameters(3*m);[(
                else
                    FitResults=[FitResults ; [round(m) FitParameters(3*m-2) height(m)
abs(FitParameters(3*m-1)) area(m)] FitParameters(3*m);[(
                end
            end

        otherwise % Normal shapes with 2 iterated variables

            for m=1:NumPeaks,
                if m==1,
                    FitResults=[round(m) FitParameters(2*m-1)+xoffset height(m) abs(FitParameters(2*m))
area(m);[(
                else
                    FitResults=[FitResults ; [round(m) FitParameters(2*m-1)+xoffset height(m)
abs(FitParameters(2*m)) area(m);[(
                end % if m==1
            end % for m=1:NumPeaks,
        end % switch peakshape(1(

%Display Fit Results on lower graph

if plots,
    %   Display Fit Results on lower graph
    subplot(2,1,2;(
    startx=min(xx)+(max(xx)-min(xx))./20;
    dxx=(max(xx)-min(xx))./10;
    dyy=((max(residual)-min(residual))./10;(

```

```

starty=max(residual)-dyy;
FigureSize=get(gcf,'Position');
switch peakshape(1)
    case {9,19,11} % Pulse and sigmoid shapes only
        text(startx,starty+dyy/2,['Peak #      tau1      Height      tau2      Area;('
    case 28, % Polynomial
        text(startx,starty+dyy/2,['Polynomial coefficients;('
    case 29 % Segmented linear
        text(startx,starty+dyy/2,['x-axis breakpoints;('
    case {30,31,32,33} % Special case of shapes with 3 iterated variables
        text(startx,starty+dyy/2,['Peak #      Position      Height      Width      Area      Shape
factor      ;('
    otherwise
        text(startx,starty+dyy/2,['Peak #      Position      Height      Width      Area;('
end

% Display FitResults using sprintf
if peakshape(1)==28 || peakshape(1)==29, % Polynomial or segmented linear
    for number=1:length(FitResults,(
        column=1;
        itemstring=sprintf('%0.4g',FitResults(number);((
        xposition=startx+(1.7.*dxx.*(column-1).*(600./FigureSize(3);(((
        yposition=starty-number.*dyy.*(400./FigureSize(4);(((
        text(xposition,yposition,['      ' itemstring;([
    end
else
    for peaknumber=1:NumPeaks,
        for column=1:5,
            itemstring=sprintf('%0.4g',FitResults(peaknumber,column);((

```

```

        xposition=startx+(1.7.*dxx.*(column-1).*(600./FigureSize(3;(((
        yposition=starty-peaknumber.*dyy.*(400./FigureSize(4;(((
        text(xposition,yposition,itemstring;(
    end
end
xposition=startx;
yposition=starty-(peaknumber+1).*dyy.*(400./FigureSize(4;(((
if AUTOZERO==3,
    text(xposition,yposition,['Baseline= ' num2str(baseline;(( (
end % if AUTOZERO
end % if peakshape(1(
if peakshape(1)==30 || peakshape(1)==31 || peakshape(1)==32 || peakshape(1)==33,
    for peaknumber=1:NumPeaks,
        column=6;
        itemstring=sprintf('%0.4g',FitParameters(3*peaknumber;(((
        xposition=startx+(1.7.*dxx.*(column-1).*(600./FigureSize(3;(((
        yposition=starty-peaknumber.*dyy.*(400./FigureSize(4;(((
        text(xposition,yposition,itemstring;(
    end
end
end % if plots

if NumArgOut==8,
    if plots,disp('Computing bootstrap sampling statistics.....'),end
    BootstrapResultsMatrix=zeros(6,100,NumPeaks;(
    BootstrapErrorMatrix=zeros(1,100,NumPeaks;(
    clear bx by

```

```

tic;
for trial=1:100,
    n=1;
    bx=xx;
    by=yy;
    while n<length(xx)-1,
        if rand>.5,
            bx(n)=xx(n+1);(
            by(n)=yy(n+1);(
        end
        n=n+1;
    end
    bx=bx+xoffset;

]FitResults,BootFitError]=fitpeaks(bx,by,NumPeaks,peakshape,extra,NumTrials,start,AUTOZERO,FIXE
DPARAMETERS;(

    for peak=1:NumPeaks,
        switch peakshape{1(
            case {30,31,32,33{
                BootstrapResultsMatrix(1:6,trial,peak)=FitResults(peak,1:6;(
            otherwise
                BootstrapResultsMatrix(1:5,trial,peak)=FitResults(peak,1:5;(
        end
        BootstrapErrorMatrix(:,trial,peak)=BootFitError;
    end
end
if plots,toc;end
for peak=1:NumPeaks,

```

```

if plots,
    disp(' ')
    disp(['Peak #',num2str(peak) '      Position  Height   Width   Area   Shape Factor;(']
end % if plots

BootstrapMean=mean(real(BootstrapResultsMatrix(:,:,peak;('
BootstrapSTD=std(BootstrapResultsMatrix(:,:,peak;('
BootstrapIQR=iqr(BootstrapResultsMatrix(:,:,peak;('
PercentRSD=100.*BootstrapSTD./BootstrapMean;
PercentIQR=100.*BootstrapIQR./BootstrapMean;

BootstrapMean=BootstrapMean(2:6;
BootstrapSTD=BootstrapSTD(2:6;
BootstrapIQR=BootstrapIQR(2:6;
PercentRSD=PercentRSD(2:6;
PercentIQR=PercentIQR(2:6;

if plots,
    disp(['Bootstrap Mean: ', num2str(BootstrapMean(['
    disp(['Bootstrap STD: ', num2str(BootstrapSTD(['
    disp(['Bootstrap IQR: ', num2str(BootstrapIQR(['
    disp(['Percent RSD: ', num2str(PercentRSD(['
    disp(['Percent IQR: ', num2str(PercentIQR(['

end % if plots

BootResults(peak,:)= [BootstrapMean BootstrapSTD PercentRSD BootstrapIQR PercentIQR;

end % peak=1:NumPeaks,

end % if NumArgOut==8,

if AUTOZERO==3;

else

    baseline=bgkcoef;

```

```

end
----- %

function
[FitResults,LowestError]=fitpeaks(xx,yy,NumPeaks,peakshape,extra,NumTrials,start,AUTOZERO,fixed
parameters(

%Based on peakfit Version 3: June, 2012 .

global PEAKHEIGHTS FIXEDPARAMETERS AUTOZERO BIPOLAR MINWIDTH coeff

format short g

format compact

warning off all

FIXEDPARAMETERS=fixedparameters;

xoffset=0;

if start==0;start=calcstart(xx,NumPeaks,xoffset);end

PEAKHEIGHTS=zeros(1,NumPeaks;(

n=length(xx;(

newstart=start;

coeff=0;

LOGPLOT=0;

%Perform peak fitting for selected peak shape using fminsearch function

options = optimset('TolX',.001,'Display','off','MaxFunEvals',1000;(

LowestError=1000; % or any big number greater than largest error expected

FitParameters=zeros(1,NumPeaks.*2 ;(

BestStart=zeros(1,NumPeaks.*2 ;(

height=zeros(1,NumPeaks ;(

bestmodel=zeros(size(yy;((

for k=1:NumTrials,

% StartVector=newstart

```

```

switch peakshape(1(
    case 1
        TrialParameters=fminsearch(@(lambda)(fitgaussian(lambda,xx,yy)),newstart,options);
        for Peak=1:NumPeaks;
            if TrialParameters(2*Peak)<MINWIDTH,
                TrialParameters(2*Peak)=MINWIDTH;
            end
        end
    case 2
        TrialParameters=fminsearch(@(lambda)(fitlorentzian(lambda,xx,yy)),newstart,options);
        for Peak=1:NumPeaks;
            if TrialParameters(2*Peak)<MINWIDTH,
                TrialParameters(2*Peak)=MINWIDTH;
            end
        end
    case 3
        TrialParameters=fminsearch(@(lambda)(fitlogistic(lambda,xx,yy)),newstart,options);
        for Peak=1:NumPeaks;
            if TrialParameters(2*Peak)<MINWIDTH,
                TrialParameters(2*Peak)=MINWIDTH;
            end
        end
    case 4
        TrialParameters=fminsearch(@(lambda)(fitpearson(lambda,xx,yy,extra)),newstart,options);
        for Peak=1:NumPeaks;
            if TrialParameters(2*Peak)<MINWIDTH,
                TrialParameters(2*Peak)=MINWIDTH;
            end
        end
end

```



```

        end

    end

case 5

    zxx=[zeros(size(xx)) xx zeros(size(xx);[ ((
    zyy=[zeros(size(yy)) yy zeros(size(yy);[ ((

    TrialParameters=fminsearch(@(lambda)(fitexpgaussian(lambda,zxx,zyy,-
extra)),newstart,options;(

    for Peak=1:NumPeaks;

        if TrialParameters(2*Peak)<MINWIDTH,

            TrialParameters(2*Peak)=MINWIDTH;

        end

    end

case 6

    cwnewstart(1)=newstart(1;(

    for pc=2:NumPeaks,

        cwnewstart(pc)=newstart(2.*pc-1;(

    end

    cwnewstart(NumPeaks+1)=(max(xx)-min(xx))/5;

    TrialParameters=fminsearch(@(lambda)(fitewgaussian(lambda,xx,yy)),cwnewstart,options;(

    for Peak=1:NumPeaks;

        if TrialParameters(NumPeaks+1)<MINWIDTH,

            TrialParameters(NumPeaks+1)=MINWIDTH;

        end

    end

case 7

    cwnewstart(1)=newstart(1;(

    for pc=2:NumPeaks,

        cwnewstart(pc)=newstart(2.*pc-1;(

```

```

end

cwnewstart(NumPeaks+1)=(max(xx)-min(xx))/5;

TrialParameters=fminsearch(@(lambda)(fitewlorentzian(lambda,xx,yy)),cwnewstart,options;(

for Peak=1:NumPeaks;

    if TrialParameters(NumPeaks+1)<MINWIDTH,

        TrialParameters(NumPeaks+1)=MINWIDTH;

    end

end

case 8

cwnewstart(1)=newstart(1;(

for pc=2:NumPeaks,

    cwnewstart(pc)=newstart(2.*pc-1;(

end

cwnewstart(NumPeaks+1)=(max(xx)-min(xx))/5;

TrialParameters=fminsearch(@(lambda)(fitexpewgaussian(lambda,xx,yy,-
extra)),cwnewstart,options;(

for Peak=1:NumPeaks;

    if TrialParameters(NumPeaks+1)<MINWIDTH,

        TrialParameters(NumPeaks+1)=MINWIDTH;

    end

end

case 9

TrialParameters=fminsearch(@(lambda)(fitexpulse(lambda,xx,yy)),newstart,options;(

for Peak=1:NumPeaks;

    if TrialParameters(2*Peak)<MINWIDTH,

        TrialParameters(2*Peak)=MINWIDTH;

    end

```

```

end
case 10
    TrialParameters=fminsearch(@(lambda)(fitupsigmoid(lambda,xx,yy)),newstar,optionst;(
    for Peak=1:NumPeaks;
        if TrialParameters(2*Peak)<MINWIDTH,
            TrialParameters(2*Peak)=MINWIDTH;
        end
    end
end
case 11
    fixedstart;[]=
    for pc=1:NumPeaks,
        fixedstart(pc)=min(xx)+pc.*(max(xx)-min(xx))./(NumPeaks+1;(
    end
    TrialParameters=fminsearch(@(lambda)(FitFWGaussian(lambda,xx,yy)),fixedstart,options;(
    for Peak=1:NumPeaks;
        if TrialParameters(2*Peak)<MINWIDTH,
            TrialParameters(2*Peak)=MINWIDTH;
        end
    end
end
case 12
    fixedstart;[]=
    for pc=1:NumPeaks,
        fixedstart(pc)=min(xx)+pc.*(max(xx)-min(xx))./(NumPeaks+1;(
    end
    TrialParameters=fminsearch(@(lambda)(FitFWLorentzian(lambda,xx,yy)),fixedstart,options;(
    for Peak=1:NumPeaks;
        if TrialParameters(2*Peak)<MINWIDTH,

```

```

        TrialParameters(2*Peak)=MINWIDTH;
    end
end
case 13
    TrialParameters=fminsearch(@(lambda)(fitGL(lambda,xx,yy,extra)),newstart,options;(
    for Peak=1:NumPeaks;
        if TrialParameters(2*Peak)<MINWIDTH,
            TrialParameters(2*Peak)=MINWIDTH;
        end
    end
end
case 14
    TrialParameters=fminsearch(@(lambda)(fitBiGaussian(lambda,xx,yy,extra)),newstart,options;(
    for Peak=1:NumPeaks;
        if TrialParameters(2*Peak)<MINWIDTH,
            TrialParameters(2*Peak)=MINWIDTH;
        end
    end
end
case 15
    TrialParameters=fminsearch(@(lambda)(fitBWF(lambda,xx,yy,extra)),newstart,options;(
    for Peak=1:NumPeaks;
        if TrialParameters(2*Peak)<MINWIDTH,
            TrialParameters(2*Peak)=MINWIDTH;
        end
    end
end
case 16
    fixedstart;[]=
    for pc=1:NumPeaks,

```

```

        fixedstart(pc)=(max(xx)-min(xx))./(NumPeaks+1);
    end
    TrialParameters=fminsearch(@(lambda)(FitFPGaussian(lambda,xx,yy)),fixedstart,options;(
    for Peak=1:NumPeaks;
        if TrialParameters(Peak)<MINWIDTH,
            TrialParameters(Peak)=MINWIDTH;
        end
    end
end
case 17
    fixedstart;[]=
    for pc=1:NumPeaks,
        fixedstart(pc)=(max(xx)-min(xx))./(NumPeaks+1);
    end
    TrialParameters=fminsearch(@(lambda)(FitFPLorentzian(lambda,xx,yy)),fixedstart,options;(
    for Peak=1:NumPeaks;
        if TrialParameters(Peak)<MINWIDTH,
            TrialParameters(Peak)=MINWIDTH;
        end
    end
end
case 18
    zxx=[zeros(size(xx)) xx zeros(size(xx); ((
    zyy=[zeros(size(yy)) yy zeros(size(yy); ((
    TrialParameters=fminsearch(@(lambda)(fitexplorentzian(lambda,zxx,zyy,-
extra)),newstart,options;(
case 19
    TrialParameters=fminsearch(@(lambda)(alphafunction(lambda,xx,yy)),newstart,options;(
    for Peak=1:NumPeaks;
        if TrialParameters(2*Peak)<MINWIDTH,

```

```

        TrialParameters(2*Peak)=MINWIDTH;
    end
end
case 20
    TrialParameters=fminsearch(@(lambda)(fitvoigt(lambda,xx,yy,extra)),newstart,options;(
    for Peak=1:NumPeaks;
        if TrialParameters(2*Peak)<MINWIDTH,
            TrialParameters(2*Peak)=MINWIDTH;
        end
    end
end

```

```

case 21
    TrialParameters=fminsearch(@(lambda)(fittriangular(lambda,xx,yy)),newstart,options;(
    for Peak=1:NumPeaks;
        if TrialParameters(2*Peak)<MINWIDTH,
            TrialParameters(2*Peak)=MINWIDTH;
        end
    end
end

```

```

case 22

```

```

TrialParameters=fminsearch(@(lambda)(fitmultiple(lambda,xx,yy,shapesvector,extra)),newstart,options;(

```

```

    for Peak=1:NumPeaks;
        if TrialParameters(2*Peak)<MINWIDTH(Peak,(
            TrialParameters(2*Peak)=MINWIDTH(Peak;(
        end
    end
end

```

```

case 23

```

```

    TrialParameters=fminsearch(@(lambda)(fitdownsigmoid(lambda,xx,yy)),newstar,optionst;(

```

```

for Peak=1:NumPeaks;
    if TrialParameters(2*Peak)<MINWIDTH,
        TrialParameters(2*Peak)=MINWIDTH;
    end
end
case 24
    TrialParameters=fminsearch(@(lambda)(fitnbinpdf(lambda,xx,yy)),newstart,options;(
for Peak=1:NumPeaks;
    if TrialParameters(2*Peak)<MINWIDTH,
        TrialParameters(2*Peak)=MINWIDTH;
    end
end
case 25
    TrialParameters=fminsearch(@(lambda)(fitlognpdf(lambda,xx,yy)),newstart,options;(
for Peak=1:NumPeaks;
    if TrialParameters(2*Peak)<MINWIDTH,
        TrialParameters(2*Peak)=MINWIDTH;
    end
end
case 26
    TrialParameters=fminsearch(@(lambda)(fitlinslope(lambda,xx,yy)),polyfit(xx,yy,1),options;(
coeff=TrialParameters;
case 27
    TrialParameters=fminsearch(@(lambda)(fitd1gauss(lambda,xx,yy)),newstart,options;(
for Peak=1:NumPeaks;
    if TrialParameters(2*Peak)<MINWIDTH,
        TrialParameters(2*Peak)=MINWIDTH;

```

```

        end

    end

case 28

    TrialParameters=fitpolynomial(xx,yy,extra;(

case 29

    TrialParameters=fminsearch(@(lambda)(fitsegmented(lambda,xx,yy)),newstart,options;(

case 30

    TrialParameters=fminsearch(@(lambda)(fitvoigtv(lambda,xx,yy)),newstart;(

case 31

    zxx=[zeros(size(xx)) xx zeros(size(xx);[ ((
    zyy=[zeros(size(yy)) yy zeros(size(yy);[ ((
    TrialParameters=fminsearch(@(lambda)(fitexpgaussianv(lambda,zxx,zyy)),newstart;(

case 32

    TrialParameters=fminsearch(@(lambda)(fitpearsonv(lambda,xx,yy)),newstart;(

case 33

    TrialParameters=fminsearch(@(lambda)(fitGLv(lambda,xx,yy)),newstart;(

otherwise

end % switch peakshape

for peaks=1:NumPeaks,
    peakindex=2*peaks-1;
    newstart(peakindex)=start(peakindex)-xoffset;
end

% Construct model from Trial parameters

A=zeros(NumPeaks,n;(

for m=1:NumPeaks,

```


switch peakshape(1(

case 1

A(m,:)=gaussian(xx,TrialParameters(2*m-1),TrialParameters(2*m);((

case 2

A(m,:)=lorentzian(xx,TrialParameters(2*m-1),TrialParameters(2*m);((

case 3

A(m,:)=logistic(xx,TrialParameters(2*m-1),TrialParameters(2*m);((

case 4

A(m,:)=pearson(xx,TrialParameters(2*m-1),TrialParameters(2*m),extra;((

case 5

A(m,:)=expgaussian(xx,TrialParameters(2*m-1),TrialParameters(2*m),-extra;('

case 6

A(m,:)=gaussian(xx,TrialParameters(m),TrialParameters(NumPeaks+1);((

case 7

A(m,:)=lorentzian(xx,TrialParameters(m),TrialParameters(NumPeaks+1);((

case 8

A(m,:)=expgaussian(xx,TrialParameters(m),TrialParameters(NumPeaks+1),-extra;('

case 9

A(m,:)=exppulse(xx,TrialParameters(2*m-1),TrialParameters(2*m);((

case 10

A(m,:)=upsigmoid(xx,TrialParameters(2*m-1),TrialParameters(2*m);((

case 11

A(m,:)=gaussian(xx,TrialParameters(m),FIXEDPARAMETERS;((

case 12

A(m,:)=lorentzian(xx,TrialParameters(m),FIXEDPARAMETERS;((

case 13

A(m,:)=GL(xx,TrialParameters(2*m-1),TrialParameters(2*m),extra;((

case 14

```
A(m,:)=BiGaussian(xx,TrialParameters(2*m-1),TrialParameters(2*m),extra;(
```

case 15

```
A(m,:)=BWF(xx,TrialParameters(2*m-1),TrialParameters(2*m),extra;(
```

case 16

```
A(m,:)=gaussian(xx,FIXEDPOSITIONS(m),TrialParameters(m);((
```

case 17

```
A(m,:)=lorentzian(xx,FIXEDPOSITIONS(m),TrialParameters(m);((
```

case 18

```
A(m,:)=explorentzian(xx,TrialParameters(2*m-1),TrialParameters(2*m),-extra;('
```

case 19

```
A(m,:)=alphafunction(xx,TrialParameters(2*m-1),TrialParameters(2*m);((
```

case 20

```
A(m,:)=voigt(xx,TrialParameters(2*m-1),TrialParameters(2*m),extra;(
```

case 21

```
A(m,:)=triangular(xx,TrialParameters(2*m-1),TrialParameters(2*m);((
```

case 22

```
A(m,:)=peakfunction(shapesvector(m),xx,TrialParameters(2*m-1),TrialParameters(2*m),extra(m);((
```

case 23

```
A(m,:)=downsigmoid(xx,TrialParameters(2*m-1),TrialParameters(2*m) ;((
```

case 24

```
A(m,:)=nbinpdf(xx,TrialParameters(2*m-1),TrialParameters(2*m);((
```

case 25

```
A(m,:)=lognormal(xx,TrialParameters(2*m-1),TrialParameters(2*m);((
```

case 26

```
A(m,:)=linslope(xx,TrialParameters(2*m-1),TrialParameters(2*m);((
```

case 27

```

        A(m,:)=d1gauss(xx,TrialParameters(2*m-1),TrialParameters(2*m) ;((
case 28
        A(m,:)=polynomial(xx,TrialParameters(2*m-1),TrialParameters(2*m) ;((
case 29
        A(m,:)=segmented(xx,yy,PEAKHEIGHTS;(
case 30
        A(m,:)=voigt(xx,TrialParameters(3*m-2),TrialParameters(3*m-1),TrialParameters(3*m) ;((
case 31
        A(m,:)=expgaussian(xx,TrialParameters(3*m-2),TrialParameters(3*m-
1),TrialParameters(3*m) ;((
case 32
        A(m,:)=pearson(xx,TrialParameters(3*m-2),TrialParameters(3*m-
1),TrialParameters(3*m) ;((
case 33
        A(m,:)=GL(xx,TrialParameters(3*m-2),TrialParameters(3*m-1),TrialParameters(3*m);((
end % switch
end % for

% Multiplies each row by the corresponding amplitude and adds them up
if peakshape(1)==29, % Segmented linear
    model=segmented(xx,yy,PEAKHEIGHTS;(
    TrialParameters=coeff;
    Heights=ones(size(coeff);((
else
    if AUTOZERO==3,
        baseline=PEAKHEIGHTS(1;(
        Heights=PEAKHEIGHTS(2:1+NumPeaks;(
        model=Heights'*A+baseline;

```

```

else
    model=PEAKHEIGHTS'*A;
    Heights=PEAKHEIGHTS;
    baseline=0;
end
end

% Compare trial model to data segment and compute the fit error
MeanFitError=100*norm(yy-model)./(sqrt(n)*max(yy;((
% Take only the single fit that has the lowest MeanFitError
if MeanFitError<LowestError,
    if min(Heights)>=-BIPOLAR*10^100, % Consider only fits with positive peak heights
        LowestError=MeanFitError; % Assign LowestError to the lowest MeanFitError
        FitParameters=TrialParameters; % Assign FitParameters to the fit with the lowest
MeanFitError
        height=Heights; % Assign height to the PEAKHEIGHTS with the lowest MeanFitError
    end % if min(PEAKHEIGHTS)>0
end % if MeanFitError<LowestError
end % for k (NumTrials(
    Rsquared=1-(norm(yy-bestmodel).)/norm(yy-mean(yy;(((
    SStot=sum((yy-mean(yy)).^2;(
    SSres=sum((yy-bestmodel).^2;(
    Rsquared=1-(SSres./SStot;(
    GOF=[LowestError Rsquared;[
for m=1:NumPeaks,
    area(m)=trapz(xx+xoffset,height(m)*A(m,:)); % Compute the area of each component peak using
trapezoidal method
end

```

```

%Put results into a matrix FitResults, one row for each peak, showing peak index number,
%position, amplitude, and width.
FitResults=zeros(NumPeaks,6);
switch peakshape(1,(
    case {6,7,8}, % equal-width peak models only
        for m=1:NumPeaks,
            if m==1,
                FitResults=[[round(m) FitParameters(m)+xoffset height(m)
abs(FitParameters(NumPeaks+1)) area(m);[(
            else
                FitResults=[FitResults ; [round(m) FitParameters(m)+xoffset height(m)
abs(FitParameters(NumPeaks+1)) area(m);[(
            end
        end
    case {11,12}, % Fixed-width shapes only
        for m=1:NumPeaks,
            if m==1,
                FitResults=[[round(m) FitParameters(m)+xoffset height(m) FIXEDPARAMETERS area(m);[(
            else
                FitResults=[FitResults ; [round(m) FitParameters(m)+xoffset height(m) FIXEDPARAMETERS
area(m);[(
            end
        end
    case {16,17}, % Fixed-position shapes only
        for m=1:NumPeaks,
            if m==1,
                FitResults=[round(m) FIXEDPOSITIONS(m) height(m) FitParameters(m) area(m);[(
            else

```

```

        FitResults=[FitResults ; [round(m) FIXEDPOSITIONS(m) height(m) FitParameters(m)
area(m);[(
            end
        end
    case 28, % Simple polynomial fit
        FitResults=PEAKHEIGHTS;
    case 29, % Segmented linear fit
        FitResults=PEAKHEIGHTS;
    case {30,31,32,33} % Special case of shapes with 3 iterated variables
        for m=1:NumPeaks,
            if m==1,
                FitResults=[round(m) FitParameters(3*m-2) height(m) abs(FitParameters(3*m-1)) area(m)
FitParameters(3*m);[(
                    else
                        FitResults=[FitResults ; [round(m) FitParameters(3*m-2) height(m)
abs(FitParameters(3*m-1)) area(m) FitParameters(3*m);[(
                            end
                        end
                    otherwise % Normal shapes with 2 iterated variables
                        for m=1:NumPeaks,
                            if m==1,
                                FitResults=[round(m) FitParameters(2*m-1)+xoffset height(m) abs(FitParameters(2*m))
area(m);[(
                                    else
                                        FitResults=[FitResults ; [round(m) FitParameters(2*m-1)+xoffset height(m)
abs(FitParameters(2*m)) area(m);[(
                                            end % if m==1
                            end % for m=1:NumPeaks,
                    end % switch peakshape(1(

```

```

----- %
function start=calcstart(xx,NumPeaks,xoffset(
    n=max(xx)-min(xx);(
    start;[]=
    startpos=[n/(NumPeaks+1):n/(NumPeaks+1):n-(n/(NumPeaks+1))]+min(xx);(
    for marker=1:NumPeaks,
        markx=startpos(marker)+ xoffset;
        start=[start markx n/ (3.*NumPeaks);[(
    end % for marker

```

```

----- %
function [index,closestval]=val2ind(x,val(
    %Returns the index and the value of the element of vector x that is closest to val
    %If more than one element is equally close, returns vectors of indices and values
    %Tom O'Haver (toh@umd.edu) October 2006
    %Examples: If x=[1 2 4 3 5 9 6 4 5 3 1], then val2ind(x,6)=7 and val2ind(x,5.1)=[5 9[
] %indices values]=val2ind(x,3.3) returns indices = [4 10] and values = [3 3[
    dif=abs(x-val);(
    index=find((dif-min(dif))==0);(
    closestval=x(index);(

```

```

----- %
function err = fitgaussian(lambda,t,y(
    %Fitting function for a Gaussian band signal.
    global PEAKHEIGHTS AUTOZERO BIPOLAR LOGPLOT
    numpeaks=round(length(lambda)/2);(
    A = zeros(length(t),numpeaks);(
    for j = 1:numpeaks,
        %if lambda(2*j)<MINWIDTH,lambda(2*j)=MINWIDTH;end
        A(:,j) = gaussian(t,lambda(2*j-1),lambda(2*j);'((

```

```

end

if AUTOZERO==3,A=[ones(size(y))' A];end

if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end

z = A*PEAKHEIGHTS;

if LOGPLOT,

    err = norm(log10(z)-log10(y),'(

else

    err = norm(z-y,'(

end

----- %

function err = fitewgaussian(lambda,t,y(

%Fitting function for a Gaussian band signal with equal peak widths.

global PEAKHEIGHTS AUTOZERO BIPOLAR LOGPLOT

numpeaks=round(length(lambda)-1);(

A = zeros(length(t),numpeaks);(

for j = 1:numpeaks,

    A(:,j) = gaussian(t,lambda(j),lambda(numpeaks+1),'((

end

if AUTOZERO==3,A=[ones(size(y))' A];end

if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end

z = A*PEAKHEIGHTS;

if LOGPLOT,

    err = norm(log10(z)-log10(y),'(

else

    err = norm(z-y,'(

end

----- %

```



```

function err = FitFWGaussian(lambda,t,y(
%      Fitting function for a fixed width Gaussian
global PEAKHEIGHTS AUTOZERO FIXEDPARAMETERS BIPOLAR LOGPLOT
numpeaks=round(length(lambda);((
A = zeros(length(t),numpeaks);(
for j = 1:numpeaks,
    A(:,j) = gaussian(t,lambda(j),FIXEDPARAMETERS;('
end
if AUTOZERO==3,A=[ones(size(y))' A];end
if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end
z = A*PEAKHEIGHTS;
if LOGPLOT,
    err = norm(log10(z)-log10(y);('
else
    err = norm(z-y;('
end

```

----- %

```

function err = FitFPGaussian(lambda,t,y(
%      Fitting function for fixed-position Gaussians
global PEAKHEIGHTS AUTOZERO FIXEDPARAMETERS BIPOLAR LOGPLOT
numpeaks=round(length(lambda);((
A = zeros(length(t),numpeaks);(
for j = 1:numpeaks,
    A(:,j) = gaussian(t,FIXEDPARAMETERS(j), lambda(j);('
end
if AUTOZERO==3,A=[ones(size(y))' A];end
if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end

```

```

z = A*PEAKHEIGHTS;

if LOGPLOT,
    err = norm(log10(z)-log10(y);('
else
    err = norm(z-y;('
end
----- %

function err = FitFPLorentzian(lambda,t,y(
%    Fitting function for fixed-position Lorentzians
global PEAKHEIGHTS AUTOZERO FIXEDPARAMETERS BIPOLAR
numpeaks=round(length(lambda);(
A = zeros(length(t),numpeaks;(
for j = 1:numpeaks,
    A(:,j) = lorentzian(t,FIXEDPARAMETERS(j), lambda(j);('
end
if AUTOZERO==3,A=[ones(size(y))' A];end
if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end
z = A*PEAKHEIGHTS;
err = norm(z-y;('
----- %

function err = FitFWLorentzian(lambda,t,y(
%    Fitting function for fixed width Lorentzian
global PEAKHEIGHTS AUTOZERO FIXEDPARAMETERS BIPOLAR LOGPLOT
numpeaks=round(length(lambda);(
A = zeros(length(t),numpeaks;(
for j = 1:numpeaks,
    A(:,j) = lorentzian(t,lambda(j),FIXEDPARAMETERS;('

```

```

end

if AUTOZERO==3,A=[ones(size(y))' A];end

if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end

z = A*PEAKHEIGHTS;

if LOGPLOT,

    err = norm(log10(z)-log10(y),'(

else

    err = norm(z-y,'(

end

----- %

function err = fitewlorentzian(lambda,t,y(

%Fitting function for a Lorentzian band signal with equal peak widths.

global PEAKHEIGHTS AUTOZERO BIPOLAR LOGPLOT

numpeaks=round(length(lambda)-1);(

A = zeros(length(t),numpeaks);(

for j = 1:numpeaks,

    A(:,j) = lorentzian(t,lambda(j),lambda(numpeaks+1),'((

end

if AUTOZERO==3,A=[ones(size(y))' A];end

if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end

z = A*PEAKHEIGHTS;

if LOGPLOT,

    err = norm(log10(z)-log10(y),'(

else

    err = norm(z-y,'(

end

----- %

```

```

function g = gaussian(x,pos,wid(
%gaussian(X,pos,wid) = gaussian peak centered on pos, half-width=wid
%X may be scalar, vector, or matrix, pos and wid both scalar
%Examples: gaussian([0 1 2],1,2) gives result [0.5000  1.0000  0.5000]
%plot(gaussian([1:100],50,20)) displays gaussian band centered at 50 with width 20.
g = exp(-((x-pos)./(0.6005615.*wid)).^2);

```

```

----- %
function err = fitlorentzian(lambda,t,y(
%      Fitting function for single lorentzian, lambda(1)=position, lambda(2)=width
%      Fitgauss assumes a lorentzian function
global PEAKHEIGHTS AUTOZERO BIPOLAR LOGPLOT
A = zeros(length(t),round(length(lambda)/2);(
for j = 1:length(lambda)/2,
    A(:,j) = lorentzian(t,lambda(2*j-1),lambda(2*j);('
end
if AUTOZERO==3,A=[ones(size(y))' A];end
if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end
z = A*PEAKHEIGHTS;
if LOGPLOT,
    err = norm(log10(z)-log10(y);('
else
    err = norm(z-y;('
end

```

```

----- %
function g = lorentzian(x,position,width(
%lorentzian(x,position,width) Lorentzian function.
%where x may be scalar, vector, or matrix

```

```

%position and width scalar

%T. C. O'Haver, 1988

%Example: lorentzian([1 2 3],2,2) gives result [0.5 1 0.5]
g=ones(size(x))./(1+((x-position)./(0.5.*width)).^2);(
----- %

function err = fitlogistic(lambda,t,y(
%     Fitting function for logistic, lambda(1)=position, lambda(2)=width
%     between the data and the values computed by the current
%     function of lambda. Fitlogistic assumes a logistic function

%T. C. O'Haver, May 2006

global PEAKHEIGHTS AUTOZERO BIPOLAR LOGPLOT
A = zeros(length(t),round(length(lambda)/2);(
for j = 1:length(lambda)/2,
    A(:,j) = logistic(t,lambda(2*j-1),lambda(2*j);('
end
if AUTOZERO==3,A=[ones(size(y))' A];end
if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end
z = A*PEAKHEIGHTS;
if LOGPLOT,
    err = norm(log10(z)-log10(y);('
else
    err = norm(z-y;('
end
----- %

function g = logistic(x,pos,wid(
%logistic function. pos=position; wid=half-width (both scalar(
%logistic(x,pos,wid), where x may be scalar, vector, or matrix

```

```

%pos=position; wid=half-width (both scalar(
%T. C. O'Haver, 1991
n = exp(-(x-pos)/(.477.*wid)) .^2;(
g = (2.*n)/(1+n);(
----- %
function err = fittriangular(lambda,t,y(
%    Fitting function for triangular, lambda(1)=position, lambda(2)=width
%    between the data and the values computed by the current
%    function of lambda. Fittriangular assumes a triangular function
%T. C. O'Haver, May 2006
global PEAKHEIGHTS AUTOZERO BIPOLAR LOGPLOT
A = zeros(length(t),round(length(lambda)/2);(
for j = 1:length(lambda)/2,
    A(:,j) = triangular(t,lambda(2*j-1),lambda(2*j);(
end
if AUTOZERO==3,A=[ones(size(y))' A];end
if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end
z = A*PEAKHEIGHTS;
if LOGPLOT,
    err = norm(log10(z)-log10(y);('
else
    err = norm(z-y;('
end
----- %
function g = triangular(x,pos,wid(
%triangle function. pos=position; wid=half-width (both scalar(
%triangular(x,pos,wid), where x may be scalar or vector,

```

```

%pos=position; wid=half-width (both scalar(
%T. C. O'Haver, 1991
%Example
%x=[0:.1:10];plot(x,trianglar(x,5.5,2.3(' ',(
g=1-(1./wid) .*abs(x-pos);(
for i=1:length(x) ,(
if g(i)<0,g(i)=0;end
end
----- %
function err = fitpearson(lambda,t,y,shapeconstant(
    %Fitting functions for a Pearson 7 band signal.
    %T. C. O'Haver (toh@umd.edu), Version 1.3, October 23, 2006.
global PEAKHEIGHTS AUTOZERO BIPOLAR LOGPLOT
A = zeros(length(t),round(length(lambda)/2);((
for j = 1:length(lambda)/2,
    A(:,j) = pearson(t,lambda(2*j-1),lambda(2*j),shapeconstant;('
end
if AUTOZERO==3,A=[ones(size(y))' A];end
if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end
z = A*PEAKHEIGHTS;
if LOGPLOT,
    err = norm(log10(z)-log10(y);('
else
    err = norm(z-y;('
end
----- %
function err = fitpearsonv(lambda,t,y(

```

```

%Fitting functions for pearson function with independently variable
%percent Gaussian
%T. C. O'Haver (toh@umd.edu), 2015.

global PEAKHEIGHTS AUTOZERO BIPOLAR LOGPLOT

A = zeros(length(t),round(length(lambda)/3));(
for j = 1:length(lambda)/3,
    A(:,j) = pearson(t,lambda(3*j-2),lambda(3*j-1),lambda(3*j);'(
end

if AUTOZERO==3,A=[ones(size(y))' A];end

if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end

z = A*PEAKHEIGHTS;

if LOGPLOT,
    err = norm(log10(z)-log10(y);('
else
    err = norm(z-y;('
end

----- %

function g = pearson(x,pos,wid,m(

%Pearson VII function .

%g = pearson7(x,pos,wid,m) where x may be scalar, vector, or matrix

%pos=position; wid=half-width (both scalar(

%m=some number

%T. C. O'Haver, 1990

g=ones(size(x))./(1+((x-pos)./(0.5.^(2/m)).*wid)).^2).^m;

----- %

function err = fitexpgaussian(lambda,t,y,timeconstant(

%Fitting functions for a exponentially-broadened Gaussian band signal.

```


%T. C. O'Haver, October 23, 2006.

global PEAKHEIGHTS AUTOZERO BIPOLAR LOGPLOT

A = zeros(length(t),round(length(lambda)/2);((

for j = 1:length(lambda)/2,

A(:,j) = expgaussian(t,lambda(2*j-1),lambda(2*j),timeconstant);(

end

if AUTOZERO==3,A=[ones(size(y))' A];end

if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end

z = A*PEAKHEIGHTS;

if LOGPLOT,

err = norm(log10(z)-log10(y);('

else

err = norm(z-y;('

end

----- %

function err = fitexplorentzian(lambda,t,y,timeconstant(

%Fitting functions for a exponentially-broadened lorentzian band signal.

%T. C. O'Haver, 2013.

global PEAKHEIGHTS AUTOZERO BIPOLAR LOGPLOT

A = zeros(length(t),round(length(lambda)/2);((

for j = 1:length(lambda)/2,

A(:,j) = explorentzian(t,lambda(2*j-1),lambda(2*j),timeconstant);(

end

if AUTOZERO==3,A=[ones(size(y))' A];end

if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end

z = A*PEAKHEIGHTS;

if LOGPLOT,

```

    err = norm(log10(z)-log10(y);('
else
    err = norm(z-y;('
end
----- %
function err = fitexpwgaussian(lambda,t,y,timeconstant(
%Fitting function for exponentially-broadened Gaussian bands with equal peak widths.
global PEAKHEIGHTS AUTOZERO BIPOLAR LOGPLOT
numpeaks=round(length(lambda)-1);(
A = zeros(length(t),numpeaks);(
for j = 1:numpeaks,
    A(:,j) = expgaussian(t,lambda(j),lambda(numpeaks+1),timeconstant);(
end
if AUTOZERO==3,A=[ones(size(y))' A];end
if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end
z = A*PEAKHEIGHTS;
if LOGPLOT,
    err = norm(log10(z)-log10(y);('
else
    err = norm(z-y;('
end
----- %
function err = fitexpgaussianv(lambda,t,y(
%Fitting functions for exponentially-broadened Gaussians with
%independently variable time constants
%T. C. O'Haver (toh@umd.edu), 2015.
global PEAKHEIGHTS AUTOZERO BIPOLAR LOGPLOT

```

```

A = zeros(length(t),round(length(lambda)/3));(
for j = 1:length(lambda)/3,
    A(:,j) = expgaussian(t,lambda(3*j-2),lambda(3*j-1),-lambda(3*j);('
end
if AUTOZERO==3,A=[ones(size(y))' A];end
if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end
z = A*PEAKHEIGHTS;
if LOGPLOT,
    err = norm(log10(z)-log10(y);('
else
    err = norm(z-y;('
end

```

```

----- %

```

```

function g = expgaussian(x,pos,wid,timeconstant(
%Exponentially-broadened gaussian(x,pos,wid) = gaussian peak centered on pos, half-width=wid
%x may be scalar, vector, or matrix, pos and wid both scalar
%T. C. O'Haver, 2006
g = exp(-((x-pos)./(0.6005615.*wid)).^2);(
g = ExpBroaden(g',timeconstant;('

```

```

----- %

```

```

function g = explorentzian(x,pos,wid,timeconstant(
%Exponentially-broadened lorentzian(x,pos,wid) = lorentzian peak centered on pos, half-
width=wid
%x may be scalar, vector, or matrix, pos and wid both scalar
%T. C. O'Haver, 2013
g = ones(size(x))./(1+((x-pos)./(0.5.*wid)).^2);(
g = ExpBroaden(g',timeconstant;('

```

```

----- %

```

```

function yb = ExpBroaden(y,t(
%ExpBroaden(y,t) zero pads y and convolutes result by an exponential decay
%of time constant t by multiplying Fourier transforms and inverse
%transforming the result.
hly=round(length(y)./2;(
ey=[y(1).*ones(1,hly)';y;y(length(y)).*ones(1,hly);['(
%figure(2);plot(ey);figure(1;(
fy=fft(ey;(
a=exp(-(1:length(fy))./t;(
fa=fft(a;(
fy1=fy.*fa;'
ybz=real(ifft(fy1))./sum(a;(
yb=ybz(hly+2:length(ybz)-hly+1;(
----- %
function err = fitexppulse(tau,x,y(
%Iterative fit of the sum of exponential pulses
%of the form Height.*exp(-tau1.*x).*(1-exp(-tau2.*x(((
global PEAKHEIGHTS AUTOZERO BIPOLAR LOGPLOT
A = zeros(length(x),round(length(tau)/2);((
for j = 1:length(tau)/2,
    A(:,j) = exppulse(x,tau(2*j-1),tau(2*j);((
end
if AUTOZERO==3,A=[ones(size(y))' A];end
if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end
z = A*PEAKHEIGHTS;
if LOGPLOT,
    err = norm(log10(z)-log10(y);('

```

```

else
    err = norm(z-y;('
end

----- %

function g = exppulse(x,t1,t2(
    %Exponential pulse of the form
    %g = (x-spoin). ./pos. *exp(1-(x-spoin). ./pos; (
    e=(x-t1). /t2;
    p = 4*exp(-e). *(1-exp(-e;((
    p=p .* (p>0; (
    g = p;'
----- %

function err = fitalphafunction(tau,x,y(
    %Iterative fit of the sum of alpha funciton
    %of the form Height. *exp(-tau1. *x). *(1-exp(-tau2. *x(((
global PEAKHEIGHTS AUTOZERO BIPOLAR LOGPLOT
A = zeros(length(x),round(length(tau)/2;((
for j = 1:length(tau)/2,
    A(:,j) = alphafunction(x,tau(2*j-1),tau(2*j;((
end
if AUTOZERO==3,A=[ones(size(y))' A];end
if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end
z = A*PEAKHEIGHTS;
if LOGPLOT,
    err = norm(log10(z)-log10(y;('(
else
    err = norm(z-y;('

```

```

end
----- %
function g = alphafunction(x,pos,spoint(
%alpha function. pos=position; wid=half-width (both scalar(
%alphafunction(x,pos,wid), where x may be scalar, vector, or matrix
%pos=position; wid=half-width (both scalar(
%Taekyung Kwon, July 2013
g = (x-spoint)./pos.*exp(1-(x-spoint)./pos);(
for m=1:length(x);if g(m)<0;g(m)=0;end;end

```

```

----- %
function err = fitdownsigmoid(tau,x,y(
%Fitting function for iterative fit to the sum of
%downward moving sigmioids
global PEAKHEIGHTS AUTOZERO BIPOLAR LOGPLOT
A = zeros(length(x),round(length(tau)/2);((
for j = 1:length(tau)/2,
    A(:,j) = downsigmoid(x,tau(2*j-1),tau(2*j));((
end
if AUTOZERO==3,A=[ones(size(y))' A];end
if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end
z = A*PEAKHEIGHTS;
if LOGPLOT,
    err = norm(log10(z)-log10(y);('
else
    err = norm(z-y;('
end

```

```

----- %
function err = fitupsigmoid(tau,x,y(

```

```

%Fitting function for iterative fit to the sum of
%upwards moving sigmoids
global PEAKHEIGHTS AUTOZERO BIPOLAR LOGPLOT
A = zeros(length(x),round(length(tau)/2);((
for j = 1:length(tau)/2,
    A(:,j) = upsigmoid(x,tau(2*j-1),tau(2*j);((
end
if AUTOZERO==3,A=[ones(size(y))' A];end
if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end
z = A*PEAKHEIGHTS;
if LOGPLOT,
    err = norm(log10(z)-log10(y);('
else
    err = norm(z-y;('
end
----- %
function g=downsigmoid(x,t1,t2(
% down step sigmoid
g=.5-.5*erf(real((x-t1)/sqrt(2*t2);(((
----- %
function g=upsigmoid(x,t1,t2(
%up step sigmoid
g=1/2 + 1/2* erf(real((x-t1)/sqrt(2*t2 ;(((
----- %
function err = fitGL(lambda,t,y,shapeconstant(
%Fitting functions for Gaussian/Lorentzian blend.
%T. C. O'Haver (toh@umd.edu), 2012.
global PEAKHEIGHTS AUTOZERO BIPOLAR LOGPLOT

```

```

A = zeros(length(t),round(length(lambda)/2);((
for j = 1:length(lambda)/2,
    A(:,j) = GL(t,lambda(2*j-1),lambda(2*j),shapeconstant);('
end
if AUTOZERO==3,A=[ones(size(y))' A];end
if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end
z = A*PEAKHEIGHTS;
if LOGPLOT,
    err = norm(log10(z)-log10(y);('
else
    err = norm(z-y;('
end
----- %
function err = fitGLv(lambda,t,y(
%Fitting functions for Gaussian/Lorentzian blend function with
%independently variable percent Gaussian
%T. C. O'Haver (toh@umd.edu), 2015.
global PEAKHEIGHTS AUTOZERO BIPOLAR LOGPLOT
A = zeros(length(t),round(length(lambda)/3);((
for j = 1:length(lambda)/3,
    A(:,j) = GL(t,lambda(3*j-2),lambda(3*j-1),lambda(3*j);('
end
if AUTOZERO==3,A=[ones(size(y))' A];end
if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end
z = A*PEAKHEIGHTS;
if LOGPLOT,
    err = norm(log10(z)-log10(y);('

```



```

else
    err = norm(z-y;('
end

----- %

function g = GL(x,pos,wid,m(
%Gaussian/Lorentzian blend. m = percent Gaussian character
%pos=position; wid=half-width
%m = percent Gaussian character.
%T. C. O'Haver, 2012
%sizeX=size(x(
%sizepos=size(pos(
%sizewid=size(wid(
%sizeM=size(m(
g=2.*((m/100).*gaussian(x,pos,wid)+(1-(m(1)/100)).*lorentzian(x,pos,wid))/2;
----- %

function err = fitvoigt(lambda,t,y,shapeconstant(
%Fitting functions for Voigt profile function
%T. C. O'Haver (toh@umd.edu), 2013.
global PEAKHEIGHTS AUTOZERO BIPOLAR LOGPLOT
A = zeros(length(t),round(length(lambda)/2;((
for j = 1:length(lambda)/2,
    A(:,j) = voigt(t,lambda(2*j-1),lambda(2*j),shapeconstant;('
end
if AUTOZERO==3,A=[ones(size(y))' A];end
if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end
z = A*PEAKHEIGHTS;
if LOGPLOT,

```

```

    err = norm(log10(z)-log10(y);('
else
    err = norm(z-y;('
end
----- %
function err = fitvoigtv(lambda,t,y(
%Fitting functions for Voigt profile function with independently variable
%alphas
%T. C. O'Haver (toh@umd.edu), 2015.
global PEAKHEIGHTS AUTOZERO BIPOLAR LOGPLOT
A = zeros(length(t),round(length(lambda)/3);(
for j = 1:length(lambda)/3,
    A(:,j) = voigt(t,lambda(3*j-2),lambda(3*j-1),lambda(3*j);('
end
if AUTOZERO==3,A=[ones(size(y))' A];end
if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end
z = A*PEAKHEIGHTS;
if LOGPLOT,
    err = norm(log10(z)-log10(y);('
else
    err = norm(z-y;('
end
----- %
function g=voigt(xx,pos,gD,alpha(
%Voigt profile function. xx is the independent variable (energy,
%wavelength, etc), gD is the Doppler (Gaussian) width, and alpha is the
%shape constant (ratio of the Lorentzian width gL to the Doppler width gD.

```

```

%Based on Chong Tao's "Voigt lineshape spectrum simulation ,"
%File ID: #26707
%alpha=alpha
gL=alpha.*gD;
gV = 0.5346*gL + sqrt(0.2166*gL.^2 + gD.^2);(
x = gL/gV;
y = abs(xx-pos)/gV;
g = 1/(2*gV*(1.065 + 0.447*x + 0.058*x^2))*((1-x)*exp(-0.693.*y.^2) + (x./(1+y.^2)) + 0.016*(1-
x)*x*(exp(-0.0841.*y.^2.25)-1./(1 + 0.021.*y.^2.25);(((
g=g./max(g;(
----- %
function err = fitBiGaussian(lambda,t,y,shapeconstant(
    %Fitting functions for BiGaussian.
    %T. C. O'Haver (toh@umd.edu), 2012.
    global PEAKHEIGHTS AUTOZERO BIPOLAR LOGPLOT
    A = zeros(length(t),round(length(lambda)/2);((
    for j = 1:length(lambda)/2,
        A(:,j) = BiGaussian(t,lambda(2*j-1),lambda(2*j),shapeconstant;('
    end
    if AUTOZERO==3,A=[ones(size(y))' A];end
    if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end
    z = A*PEAKHEIGHTS;
    if LOGPLOT,
        err = norm(log10(z)-log10(y);('
    else
        err = norm(z-y;('
    end
    ----- %

```

```

function g = BiGaussian(x,pos,wid,m(
%BiGaussian (different widths on leading edge and trailing edge.(
%pos=position; wid=width
%m determines shape; symmetrical if m=50.
%T. C. O'Haver, 2012
lx=length(x;(
hx=val2ind(x,pos;(
g(1:hx)=gaussian(x(1:hx),pos,wid*(m/100);((
g(hx+1:lx)=gaussian(x(hx+1:lx),pos,wid*(1-m/100);((
----- %
function err = fitBWF(lambda,t,y,shapeconstant(
%Fitting function for Breit-Wigner-Fano.
%T. C. O'Haver (toh@umd.edu), 2014.
global PEAKHEIGHTS AUTOZERO BIPOLAR LOGPLOT
A = zeros(length(t),round(length(lambda)/2);((
for j = 1:length(lambda)/2,
A(:,j) = BWF(t,lambda(2*j-1),lambda(2*j),shapeconstant;('
end
if AUTOZERO==3,A=[ones(size(y))' A];end
if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end
z = A*PEAKHEIGHTS;
if LOGPLOT,
err = norm(log10(z)-log10(y);('
else
err = norm(z-y;('
end
----- %

```

```

function g = BWF(x,pos,wid,m(
% BWF (Breit-Wigner-Fano) http://en.wikipedia.org/wiki/Fano\_resonance
% pos=position; wid=width; m=Fano factor
% T. C. O'Haver, 2014
y=((m*wid/2+x-pos).^2)./(((wid/2).^2)+(x-pos).^2);(
% y=((1+(x-pos./(m.*wid))).^2)./(1+((x-pos)./wid).^2);(
g=y./max(y);(
----- %

function err = fitnbinpdf(tau,x,y(
% Fitting function for iterative fit to the sum of
% Negative Binomial Distributions
) % http://www.mathworks.com/help/stats/negative-binomial-distribution.html(
global PEAKHEIGHTS AUTOZERO BIPOLAR LOGPLOT
A = zeros(length(x),round(length(tau)/2);((
for j = 1:length(tau)/2,
    A(:,j) = nbinpdf(x,tau(2*j-1),tau(2*j);((
end
if AUTOZERO==3,A=[ones(size(y))' A];end
if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end
z = A*PEAKHEIGHTS;
if LOGPLOT,
    err = norm(log10(z)-log10(y);('
else
    err = norm(z-y;('
end
----- %

function err = fitlognpdf(tau,x,y(

```

```

%Fitting function for iterative fit to the sum of
%Lognormal Distributions
) %http://www.mathworks.com/help/stats/lognormal-distribution.html(
global PEAKHEIGHTS AUTOZERO BIPOLAR LOGPLOT
A = zeros(length(x),round(length(tau)/2);((
for j = 1:length(tau)/2,
    A(:,j) = lognormal(x,tau(2*j-1),tau(2*j);((
end
if AUTOZERO==3,A=[ones(size(y))' A];end
if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end
z = A*PEAKHEIGHTS;
if LOGPLOT,
    err = norm(log10(z)-log10(y);('
else
    err = norm(z-y;('
end
----- %
function g = lognormal(x,pos,wid(
%lognormal function. pos=position; wid=half-width (both scalar(
%lognormal(x,pos,wid), where x may be scalar, vector, or matrix
%pos=position; wid=half-width (both scalar(
%T. C. O'Haver, 1991
g = exp(-log(x/pos)/(0.01.*wid)) .^2;(
----- %
function err = fitsine(tau,x,y(
%Fitting function for iterative fit to the sum of
%sine waves (alpha test, NRFPT(

```

```

global PEAKHEIGHTS AUTOZERO BIPOLAR LOGPLOT

A = zeros(length(x),round(length(tau)/2);((
for j = 1:length(tau)/2,
    A(:,j) = sine(x,tau(2*j-1),tau(2*j);((
end
if AUTOZERO==3,A=[ones(size(y))' A];end
if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end
z = A*PEAKHEIGHTS;
if LOGPLOT,
    err = norm(log10(z)-log10(y);('
else
    err = norm(z-y;('
end
----- %
function g=sine(x,f,phase (
    %Sine wave (alpha test(
g=sin(2*pi*f*(x+phase);((
----- %
function err = fitd1gauss(lambda,t,y(
    %Fitting functions for the first derivative of a Gaussian
    %T. C. O'Haver, 2014
global PEAKHEIGHTS AUTOZERO BIPOLAR
A = zeros(length(t),round(length(lambda)/2);((
for j = 1:length(lambda)/2,
    A(:,j) = d1gauss(t,lambda(2*j-1),lambda(2*j);'((
end
if AUTOZERO==3,A=[ones(size(y))' A];end

```

```

if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end

z = A*PEAKHEIGHTS;

err = norm(z-y;('

----- %

function y=d1gauss(x,p,w(

%First derivative of Gaussian (alpha test(

y=-(5.54518.*(x-p).*exp(-(2.77259.*(p-x).^2)./w^2))./w.^2;

y=y./max(y;(

----- %

function coeff = fitpolynomial(t,y,order(

coeff=polyfit(t,y,order;(

%order=order

%coeff=coeff

----- %

function y=polynomial(t,coeff(

y=polyval(coeff,t;(

----- %

function err = fitsegmented(lambda,t,y(

%Fitting functions for articulated segmented linear

%T. C. O'Haver, 2014

global LOGPLOT

breakpoints=[t(1) lambda max(t;[(

z = segmented(t,y,breakpoints;(

%lengthz=length(z;(

if LOGPLOT,

err = norm(log10(z)-log10(y;('(

else

err = norm(z-y;(

```



```

end
----- %
function yi=segmented(x,y,segs(
global PEAKHEIGHTS
clear yy
for n=1:length(segs(
    yind=val2ind(x,segs(n);(
    yy(n)=y(yind(1);(
end
yi=INTERP1(segs,yy,x;(
PEAKHEIGHTS=segs;
----- %
function err = fitlinslope(tau,x,y(
    %Fitting function for iterative fit to linear function
global PEAKHEIGHTS AUTOZERO BIPOLAR LOGPLOT
A = zeros(length(x),round(length(tau)/2);(
for j = 1:length(tau)/2,
    z = (x.*tau(2*j-1)+tau(2*j);(
    A(:,j) = z./max(z;(
end
if AUTOZERO==3,A=[ones(size(y))' A];end
if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end
z = A*PEAKHEIGHTS;
if LOGPLOT,
    err = norm(log10(z)-log10(y);('
else
    err = norm(z-y;('

```

```

end
----- %
function y=linslope(x,slope,intercept(
y=x.*slope+intercept;
%y=y./max(y);(
----- %
function b=iqr(a(
%b = IQR(a) returns the interquartile range of the values in a. For
%vector input, b is the difference between the 75th and 25th percentiles
%of a. For matrix input, b is a row vector containing the interquartile
%range of each column of a.
%T. C. O'Haver, 2012
mina=min(a;(
sizea=size(a;(
NumCols=sizea(2;(
for n=1:NumCols,b(:,n)=a(:,n)-mina(n);end
Sorteda=sort(b;(
lx=length(Sorteda;(
SecondQuartile=round(lx/4;(
FourthQuartile=3*round(lx/4;(
b=abs(Sorteda(FourthQuartile,:)-Sorteda(SecondQuartile;((:,
----- %
function err = fitmultiple(lambda,t,y,shapvector,m(
%Fitting function for a multiple-shape band signal.
%The sequence of peak shapes are defined by the vector "shape."
%The vector "m" determines the shape of variable-shape peaks.
global PEAKHEIGHTS AUTOZERO BIPOLAR LOGPLOT coeff
numpeaks=round(length(lambda)/2;(

```

```

A = zeros(length(t),numpeaks);
for j = 1:numpeaks,
    if shapeselector(j)==28,
        coeff=polyfit(t,y,m(j);(
        A(:,j) = polyval(coeff,t;(
    else
        A(:,j) = peakfunction(shapeselector(j),t,lambda(2*j-1),lambda(2*j),m(j);('
    end
end
if AUTOZERO==3,A=[ones(size(y))' A];end
if BIPOLAR,PEAKHEIGHTS=A\y';else PEAKHEIGHTS=abs(A\y');end
z = A*PEAKHEIGHTS;
if LOGPLOT,
    err = norm(log10(z)-log10(y);('
else
    err = norm(z-y;('
end

```

----- %

```

function p=peakfunction(shape,x,pos,wid,m,coeff(
%function that generates any of 20 peak types specified by number. 'shape'
%specifies the shape type of each peak in the signal: "peakshape" = 1-20.
=1 %Gaussian 2=Lorentzian, 3=logistic, 4=Pearson, 5=exponentially
%broadened Gaussian; 9=exponential pulse, 10=up sigmoid,
=13 %Gaussian/Lorentzian blend; 14=BiGaussian, 15=Breit-Wigner-Fano (BWF, (
=18 %exponentially broadened Lorentzian; 19=alpha function; 20=Voigt
%profile; 21=triangular; 23=down sigmoid; 25=lognormal. "m" is required
%for variable-shape peaks only.

```

switch shape,

case 1

p=gaussian(x,pos,wid;(

case 2

p=lorentzian(x,pos,wid;(

case 3

p=logistic(x,pos,wid;(

case 4

p=pearson(x,pos,wid,m;(

case 5

p=expgaussian(x,pos,wid,m;(

case 6

p=gaussian(x,pos,wid;(

case 7

p=lorentzian(x,pos,wid;(

case 8

p=expgaussian(x,pos,wid,m;'

case 9

p=exppulse(x,pos,wid;(

case 10

p=upsigmoid(x,pos,wid;(

case 11

p=gaussian(x,pos,wid;(

case 12

p=lorentzian(x,pos,wid;(

case 13

p=GL(x,pos,wid,m;(

case 14

p=BiGaussian(x,pos,wid,m;(

case 15

p=BWF(x,pos,wid,m;(

case 16

p=gaussian(x,pos,wid;(

case 17

p=lorentzian(x,pos,wid;(

case 18

p=explorentzian(x,pos,wid,m;'(

case 19

p=alphafunction(x,pos,wid;(

case 20

p=voigt(x,pos,wid,m;(

case 21

p=triangular(x,pos,wid ;(

case 23

p=downsigmoid(x,pos,wid;(

case 25

p=lognormal(x,pos,wid;(

case 26

p=linslope(x,pos,wid;(

case 27

p=d1gauss(x,pos,wid;(

case 28

p=polynomial(x,coeff;(

otherwise

end % switch