Baroreflex threshold calculation

This is a detailed explanation of the algorithm used for the baroreflex threshold calculation. It was written in Matlab (The MathWorks, Natick, MA) code, using functions in Matlab and the Statistics Toolbox.

Input data were text files that contained two columns of data, the diastolic blood pressure (mmHg) and the burst area of the muscle sympathetic nerve activity (MSNA, arbitrary units). Each line represented one heart beat. Burst area 0 meant that no MSNA burst was present in that heart beat.

The result was returned in a structure and was plotted in a figure.

The code is provided as additional information on how the calculations were made in this study. No responsibility for any further use is implied.

```
function result = BaroreflexThreshold(TextFile)
%BaroreflexThreshold    Calculate T50

a = load(TextFile);
DiastBP = a(:,1);
MSNAamp = a(:,2);
BinWidth = 1;                       %1 mmHG
[B,r,p,neff,T50,xw,yw,BinWidth] = BaroRefThreshold
(DiastBP,MSNAamp,BinWidth);

%Probit analysis
T50probit = [];
probitslope = [];
i1 = ((yw > 0) & (yw < 100));
if sum(i1) > 1,
    xp = xw(i1);                    %exclude 0 and 100
    yp = yw(i1) / 100;
    yp = norminv(yp,0,1);           %probit transform
    P = polyfit(xp,yp,1);           %first guess
    P = P(:);

    %Nonlinear estimation of T50probit
    X = [ones(sum(i1),1),xp];
    beta = nlinfit(X,yw(i1),@myf21,P(end:-1:1));
    T50probit = (norminv(0.5) - beta(1))/beta(end);

    %Probit slope
    y499 = 49.9;
    y501 = 50.1;
    T499probit = (norminv(y499/100) - beta(1))/beta(end);
    T501probit = (norminv(y501/100) - beta(1))/beta(end);
    probitslope = (y501 - y499) / (T501probit - T499probit);
end;

%Plot data
plot(xw,yw,'b o');
hold on;

%Plot regression line
plot([min(xw),max(xw)]',[1,1;min(xw),max(xw)]' * B,'r- ');

if sum(i1) > 1,
    %Plot probit curve
    xpp = (xp(1):BinWidth/100:xp(end))';
    ypp = polyval(beta(end:-1:1),xpp);
    ypp = normcdf(ypp) * 100;
    plot(xpp,ypp,'g- ');

    %Plot histogram for DiastBP
    limmin = fix(min(DiastBP)/BinWidth) * BinWidth;
    limmax = ceil(max(DiastBP)/BinWidth) * BinWidth;
    xhc = (limmin + BinWidth/2:BinWidth:limmax - BinWidth/2)';
    Nh = hist(DiastBP,xhc);
    Nh = Nh(:) ./ sum(Nh(:)) * 100;
    plot(xhc,Nh,'g- ');
end;

xlabel('Diastolic BP (mmHg)');
str1 = '% heart beats associated with a burst';
str2 = 'Linear and probit regression';
str3 = 'Diastolic BP histogram';
ylabel({str1,str2,str3});
str1 = sprintf('y = %6.2f * x + %6.2f',B(end:-1:1));
str2 = sprintf('r = %6.3f,   p = %6.4f',r,p);
str3 = sprintf('T50 = %6.2f,   bin width = %6.2f mmHg',T50,BinWidth);
str4 = sprintf('T50(probit) = %6.2f,   slope =
%6.2f',T50probit,probitslope);
title({str1,str2,str3,str4});

result.B = B;
result.r = r;
result.p = p;
result.neff = neff;
result.T50 = T50;
result.BinWidth = BinWidth;
result.T50probit = T50probit;
result.probitslope = probitslope;

return;
%----------------------------------------------------------------------
function yhat = myf21(beta,X)
%yhat                   Model function
yhat = 100 * normcdf(X*beta);
return;
%----------------------------------------------------------------------
function [B,r,p,ne,T50,dbpClass0,NClass0,w] = BaroRefThreshold(dbp,msa,cw)
%BaroThreshold          Kienbaums et al, J Physiol 531: 861-869, 2001
%
% dbp                   diastolic BP
% msa                   muscle sympathetic nerve activity (MSNA) amplitude
% cw                    bin width
%
% B                     linear regression result
% r                     correlation coefficient
% p                     significance
% ne                    adjusted number of points for linear regression
% dbpClass0             weighted independent variable
% NClass0               weighted dependent variable
% w                     real bin width

B = [];
r = [];
```

```
p = [];
ne = 0;
T50 = [];
dbpClass0 = [];
NClass0 = [];
if nargin < 2, DoHelp('BaroThreshold'); return; end;
if (nargin < 3) || isempty(cw), cw = 1; end;
if isempty(dbp) || isempty(msa),
    return;
end;
w = cw;                                         %bin width, e g 1 mmHg


%make classes
limmin = fix(min(dbp)/w) * w;
limmax = ceil(max(dbp)/w) * w;
classes = (limmin + w/2:w:limmax - w/2)';
maxclasses = 5000;
if length(classes) > maxclasses,
    w = (limmax - limmin) / maxclasses;
    disp(['BaroRefThreshold: Bin width increased to ',num2str(w)]);
end;
if limmax <= limmin,
    dd = min([limmin;limmax]) - w;
    limmax = max([limmin;limmax]) + w;
    limmin = dd;
end;
classes = (limmin + w/2:w:limmax - w/2)';

[N0,d,index0] = Hist22(dbp,classes);
if length(classes) == 1,
    N0 = length(dbp);
    index0 = ones(N0,1);
end;
N0 = N0';
dbpClass0 = classes(index0);


%skip zeros
i1 = find(msa(:,1));
if isempty(i1),
    dbpClass0 = [];
    return;
end;
[N1,d,index1] = Hist22(dbp(i1,:),classes);
if length(classes) == 1,
    N1 = length(dbp(i1,:));
    index1 = ones(N1,1);
end;
N1 = N1';

%percent values with multiplicity
NClass0 = N1(index0)./N0(index0) * 100;
NClass1 = N1(index1)./N0(index1) * 100;

%Use a subset of dbpClass0 and NClass0
[dbpClass0,i3] = sort(dbpClass0);
NClass0 = NClass0(i3);
%Exclude 100%, except the first and the last 100%
i1 = find(NClass0 ~= 100);
```

```
if ~isempty(i1) && i1(end) < size(NClass0,1)-1,
    dc = dbpClass0(i1(end)+1);
    i2 = find(dbpClass0 > dc);
    if ~isempty(i2),
        NClass0(i2) = [];
        dbpClass0(i2) = [];
    end;
end;
if ~isempty(i1) && i1(1) > 2,
    dc = dbpClass0(i1(1)-1);
    i2 = find(dbpClass0 < dc);
    if ~isempty(i2),
        NClass0(i2) = [];
        dbpClass0(i2) = [];
    end;
end;
%Exclude 0%, except the first and the last 0%
i1 = find(NClass0 ~= 0);
if ~isempty(i1) && i1(end) < size(NClass0,1)-1,
    dc = dbpClass0(i1(end)+1);
    i2 = find(dbpClass0 > dc);
    if ~isempty(i2),
        NClass0(i2) = [];
        dbpClass0(i2) = [];
    end;
end;
if ~isempty(i1) && i1(1) > 2,
    dc = dbpClass0(i1(1)-1);
    i2 = find(dbpClass0 < dc);
    if ~isempty(i2),
        NClass0(i2) = [];
        dbpClass0(i2) = [];
    end;
end;

%linear regression
ne = length(NClass1);
[B,r,p] = mLinReg(dbpClass0,NClass0,ne);
T50 = (50-B(1))./B(2);

return;
%----------------------------------------------------------------------
function [B,r,p,n] = mLinReg(x,y,ne)
%mLinReg                simple linear regression with adjustable df

if (nargin < 3 || isempty(ne)),
    ne = length(y);
end;

B = [];
r = [];
p = NaN;
n = 0;

if isempty(x) || isempty(y),
    return;
end;

x = x(:);
y = y(:);
```

```
i1 = find(~isnan(x + y));
x = x(i1);
y = y(i1);
n = length(y);

if length(x) > 1,
    X = [ones(length(x),1),x];
    B = (X'*X)\X'*y;
    dfe = ne - 2;
    SSE = (y-X*B)'*(y-X*B);
    MSE = SSE/dfe;
    MSR = B'*X'*(y - mean(y));
    F0 = MSR/MSE;
    p = 1 - fcdf(F0,1,dfe);
end;

r = 1;
if n > 1, r = corrcoef(x,y); end;
if length(r) > 1, r = r(1,2); end;

return;
%----------------------------------------------------------------------
function [no,xo,yoo] = Hist22(y,x)
%Hist2                  Histogram, with index output
%
%   N = HIST2(Y) bins the elements of Y into 10 equally spaced containers
%   and returns the number of elements in each container.  If Y is a
%   matrix, HIST2 works down the columns.
%
%   N = HIST2(Y,M), where M is a scalar, uses M bins.
%
%   N = HIST2(Y,X), where X is a vector, returns the distribution of Y
%   among bins with centers specified by X.
%
%   [N,X,YO] = HIST2(...) also returns the position of the bin centers in X
%   and a matrix of size(Y) that contain the bin numbers into which each
%   element is sorted.
%
%   HIST2(...) without output arguments produces a histogram bar plot of
%   the results.

%   J.N. Little 2-06-86
%   Revised 10-29-87, 12-29-88 LS
%   Revised 8-13-91 by cmt, 2-3-92 by ls.
%   Copyright (c) 1984-98 by The MathWorks, Inc.
%   $Revision: 5.13 $  $Date: 1997/12/02 19:27:08 $

%   Hist2 revised 981015 by tok

yo = [];
yoo = [];

if nargin == 0,
    error('Requires one or two input arguments.');
end;
if nargin == 1
    x = 10;
end;
if min(size(y))==1, y = y(:); end;
if ischar(x) || ischar(y),
    error('Input arguments must be numeric.');
end;
[m,n] = size(y);
if isempty(y),
    if length(x) == 1,
       x = 1:x;
    end;
    nn = zeros(size(x)); % No elements to count
else
    if length(x) == 1,
        miny = min(min(y));
        maxy = max(max(y));
        if miny == maxy,
              miny = miny - floor(x/2) - 0.5;
              maxy = maxy + ceil(x/2) - 0.5;
        end;
        binwidth = (maxy - miny) ./ x;
        xx = miny + binwidth*(0:x);
        xx(length(xx)) = maxy;
        x = xx(1:length(xx)-1) + binwidth/2;
    else
        xx = x(:)';
        miny = min(min(y));
        maxy = max(max(y));
        binwidth = [diff(xx) 0];
        xx = [xx(1)-binwidth(1)/2 xx+binwidth/2];
        xx(1) = miny;
        xx(length(xx)) = maxy;
    end;
    nbin = length(xx);
    nn = zeros(nbin,n);
    yo = zeros(size(y));
    for i=2:nbin,
        nn(i,:) = sum(y <= xx(i));
        for j = 1:n,
            index = find(y(:,j) <= xx(i) & ~yo(:,j));
            yo(index,j) = i - 1;
        end;
    end;
    nn = nn(2:nbin,:) - nn(1:nbin-1,:);
end;

if nargout == 0,
    bar(x,nn,'hist');
else
    yoo = yo;
    if min(size(y))==1, % Return row vectors if possible.
        no = nn';
        xo = x;
    else
        no = nn;
        xo = x';
    end;
end;

return;
```