```
function [out] = csm_Mirror_closedBoundary_mod(points,maxis)
% Calculate the CSM value for 2D mirror symmetry distance of the sequence
% of points in coor. Assumes points are ordered and first point maps to
itself.
%
% points is a 1x2n list of x,y,x,y...
% maxis the mirror-axis given as 2 points [x1 y1 x2 y2]
%     if missing the function calculates the optimal axis.
%
% Function returns:
%     csmVal - the CSM value,
%     npoints - the new points of the symmetrized structure,
%     theta - the angle between the reflection axis and the y axis.
% Note that the value returned is scaled by 4 compared to that defined in the
% Pami CSM paper.
%
% Note that uses older code with points in array. Input is list so first
% maps points to array, computes and then maps newpoints array back to list.
%

% map points to array coor which is a 2xn structure. Every column is the x,y
coor of a point.
coor = reshape(points,2,length(points)./2);

numPoints = size(coor,2);

% pair the points: i pairs with j then matches(1) = j  and matches(j)=i
% Since points are ordered: first point is on m-axis and maps to itself then
% 2 maps to numpoints and i (>=2) maps to numpoints-i+2
matches = [1, numPoints : -1 : 2];

% Calculate angle theta between mirror axis and y-axis
if nargin>1 % mirror axis is given calc angle theta
   sum1 = (2*maxis(1).*maxis(2) + 2*maxis(3).*maxis(4));
   sum2 = -maxis(1)^2 + maxis(2)^2 -maxis(3)^2 + maxis(4)^2;
   theta=atan2(sum1,sum2);
   theta=theta/2.0;
   center = [(maxis(1)+maxis(3))/2,(maxis(2)+maxis(4))/2];  % m_axis passses
through mid point def by m-axis lines
 else % calc optimal mirror axis angle theta
   sum1 = sum(coor(1,:).*coor(2,matches) +
coor(2,:).*coor(1,matches));
   sum2 = sum(-coor(1,:).*coor(1,matches) +
coor(2,:).*coor(2,matches));
   theta=atan2(sum1,sum2);
   theta=theta/2.0;
   center = [0,0]; % m_axis passses through centroid of shape
end;


% can I delete the following?
% The following is needed when using atan and not atan2
%   if ((cos(theta)*sum2+sin(theta)*sum1)<0.0)
%      theta=theta/2.0;
%   else
%      theta=(theta+pi)/2.0;
%   end
```

```
% reflect points about the reflection axis by doing the following:
% 0) translate points to center
% 1) rotate points counter clockwise by theta (aligning ref axis with y-
axis),
% 2) reflect points accross y-axis
% 3) rotate back clockwise by theta
% 4) move points back from center
Rfold = [cos(-theta) -sin(-theta) ; sin(-theta) cos(-theta)]* ...
         [-1 0 ; 0 1] * [cos(theta) -sin(theta) ; sin(theta) cos(theta)];
% is above same as this? YES!
%Rfold =  [-cos(2*theta)  sin(2*theta); sin(2*theta) cos(2*theta)] * coor;

rcoor =  Rfold * (coor -
center'*ones(1,numPoints))   +  (center'*ones(1,numPoints));


% Calculate CSM val by summing the sqr distance between points and reflected
matching points.
csmVal= mean(sum((coor - rcoor(:,matches)).^2));


% calculate symmetrized points by averaging original points with reflected
matched points
ncoor = (coor + rcoor(:,matches)) ./2;

% map points back from array to list
npoints = ncoor(:)';
% Combine rcoor, ncoor, cval, and theta in a single output vector
out = [points npoints theta csmVal];
```