MATLAB code to calculate curvature of an interface
Curvature of a digital curve

Jaya Kumar Alageshan *

Indian Institute of Science, Bangalore, India.
http://jkumarres.github.io

Abstract

This document describes the algorithm used to find the curvature of contours of interfaces in an image, using the accompanying MATLAB code. We use a gradient descent based iterative scheme to estimate the best fit circle for local data points and find the curvature vector. The approach described here leads to a smoother varying curvature field in comparison to some of the methods available in literature.

1 Algorithm

A schematic of the different steps involved are given in Fig. 1.
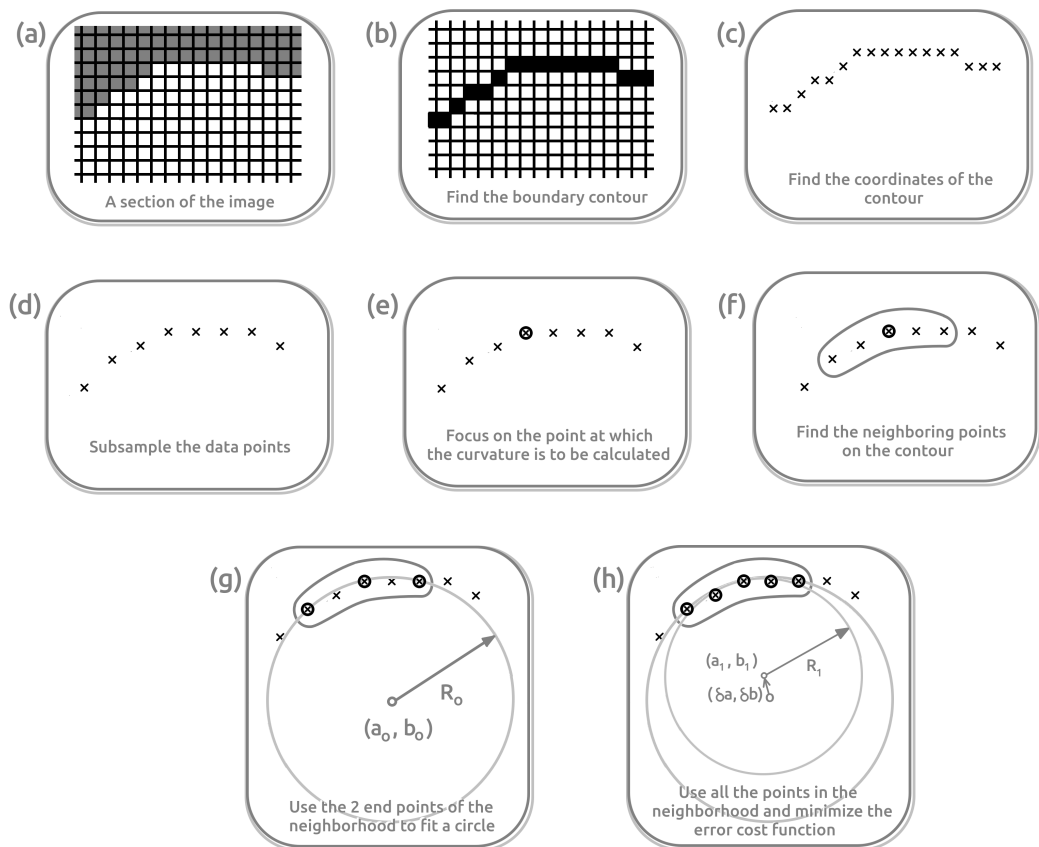
Figure 1. A schematic of sequence of operations performed to find the curvature at a point on the interface.

---

*Email: jkumar.res@gmail.com

2 Function variables

The I/O format of the function is:

function [xy , K] = InterfaceCurvature(I , v , NN , SR)

where the inputs are

$$I \longrightarrow \text{Image data in the form of 2D, 3D or 4D array}$$
$$v \longrightarrow \text{Value of the level-set} \in (0, 1)$$
$$SR \longrightarrow \text{Sub-sample rate to go from Fig. 1(c) to (d) (here } SR = 2)$$
$$NN \longrightarrow \text{To specify the neighborhood in Fig.1(f) (here } NN = 2)$$

and the outputs are

$$xy \longrightarrow \text{X- and Y- coordinates of the sub-sampled contour points in}$$
$$\text{Fig. 1(d) of } N_c \times 1 \text{ cells}$$
$$K \longrightarrow \text{The curvature vector of size } N_c \times 1 \text{ cells}$$

where, N_c is the number of distinct contours.

3 Circle through three points (Fig. 1(g))

Now to perform the operation given in Fig.1(g), we need to fit a circle through three points. The equation of a circle is given by

$$(x - a)^2 + (y - b)^2 = r^2 \tag{1}$$

where, (a, b) is the coordinate of the circle center, and r is the radius.

Now, if (x_1, y_1), (x_2, y_2), and (x_3, y_3) are three given points, then the corresponding circle through them has

$$r = \frac{1}{2} \frac{\sqrt{((x_1 - x_2)^2 + (y_1 - y_2)^2)((x_2 - x_3)^2 + (y_2 - y_3)^2)((x_3 - x_1)^2 + (y_3 - y_1)^2)}}{|x_1(y_2 - y_3) + x_2(y_3 - y_1) + x_3(y_1 - y_2)|} \tag{2}$$

$$a = \frac{1}{2} \frac{x_1^2(y_2 - y_3) + x_2^2(y_3 - y_1) + x_3^2(y_1 - y_2) - (y_1 - y_2)(y_2 - y_3)(y_3 - y_1)}{x_1(y_2 - y_3) + x_2(y_3 - y_1) + x_3(y_1 - y_2)} \tag{3}$$

$$b = \frac{1}{2} \frac{y_1^2(x_2 - x_3) + y_2^2(x_3 - x_1) + y_3^2(x_1 - x_2) - (x_1 - x_2)(x_2 - x_3)(x_3 - x_1)}{y_1(x_2 - x_3) + y_2(x_3 - x_1) + y_3(x_1 - x_2)} \tag{4}$$

4 Circle fit (Fig. 1(h))

We use the gradient descent method to find the best value of $\{a, b, r\}$ that fits a given set of points. A convenient positive-definite cost function, on which to implement gradient descent is

$$\mathcal{E}(a, b, r) := \sum_{i \in \mathcal{N}} \left[(x_i - a)^2 + (y_i - b)^2 - r^2 \right]^2 \tag{5}$$

where $\mathcal{N}$ are the neighboring points.

The gradient descent is an iterative algorithm that updates the variables, as follows

$$r' \longrightarrow r - \eta \frac{\partial \mathcal{E}}{\partial r} = r + 4r \, \eta \sum_{i \in \mathcal{N}} \left[(x_i - a)^2 + (y_i - b)^2 - r^2 \right] \tag{6}$$

$$a' \longrightarrow a - \eta \frac{\partial \mathcal{E}}{\partial a} = a + 4\eta \sum_{i \in \mathcal{N}} (x_i - a) \left[(x_i - a)^2 + (y_i - b)^2 - r^2 \right] \tag{7}$$

$$b' \longrightarrow b - \eta \frac{\partial \mathcal{E}}{\partial b} = r + 4\eta \sum_{i \in \mathcal{N}} (y_i - b) \left[(x_i - a)^2 + (y_i - b)^2 - r^2 \right] \tag{8}$$

where, η is the descent rate, which we fix at 0.01. Starting from the values of $\{a, b, r\}$ estimated using the three extreme points in Sec. 3 (Fig. 1(g)), the above iteration is performed 100 times to

arrive at the final solution.

5 Curvature vector

Once we have calculated (a, b, r) for a point (x_i, y_i), then the curvature vector is defined as

$$\mathbf{K}(x_i, y_i) := \frac{1}{r} \frac{(a - x_i, b - y_i)}{\sqrt{(a - x_i)^2 + (b - y_i)^2}} \tag{9}$$

so that the magnitude of $\mathbf{K}$ is the curvature, and the vector points towards the center of the circle.

6 Visualization

To visualize the curvature field along the contours, we plot the vector field of $-\mathbf{K}(x_i, y_i)$. Note that we plot the negative of the curvature vector, as this ensures that there is no crowding of vectors or formation of cusps.
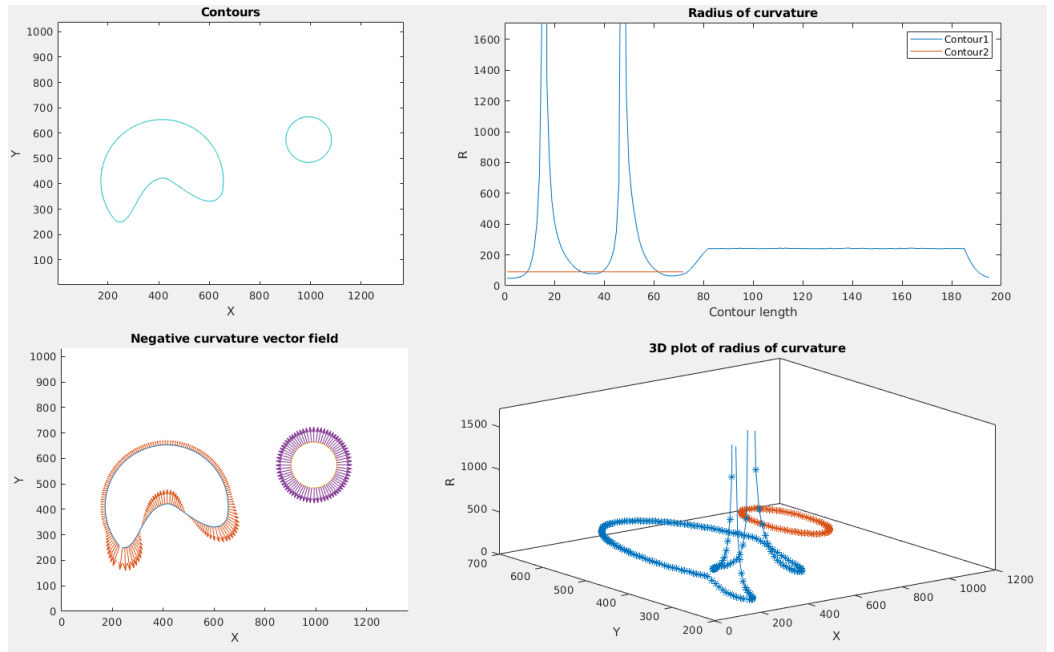
Figure 2. The top-left plot shows the extracted contours; bottom-left plot shows the corresponding curvature vector field; top-right plot shows the variation in radius of curvature along the contour length; bottom-right figure shows the spatial dependence of the radius of curvature.

A Appendix: Radius of curvature

Given a smooth curve in 2D, that is twice differentiable, the curvature is defined as

$$\mathbf{K} := \frac{d\hat{\mathbf{t}}}{ds}, \tag{10}$$

$$\text{where, } \hat{\mathbf{t}} = \frac{d\mathbf{R}(s)}{ds} \tag{11}$$

$\hat{\mathbf{t}}$ is the unit tangent vector, and $\mathbf{R}(s)$ is the position vector of the curve, parameterized by the arc-length parameter s, such that $ds = \sqrt{dx^2 + dy^2}$.

But from a computation point of view, a better representation of curvature is in terms of the inverse radius of the best fit (osculating) circle, as shown in Fig. 3
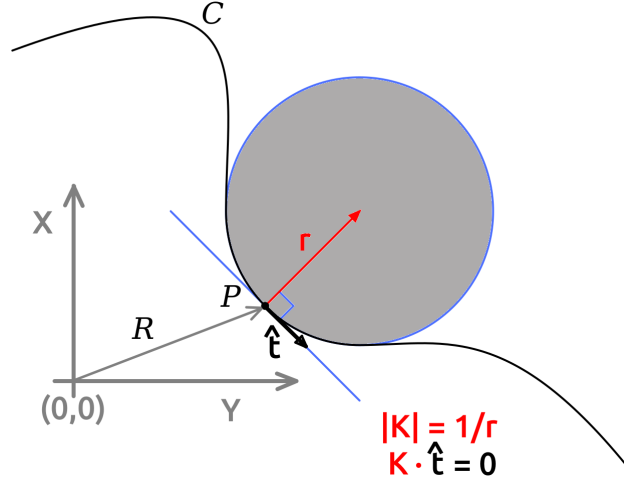

Figure 3. Schematic of curvature calculation

B Appendix: Note on MATLAB *contour()* function format

In MATLAB, to get the coordinates of the contour, corresponding to the level-set value of v, we use

M = contour(I , [v v]);

where the format of M is as described in Fig. 4 (from mathworks documentation)

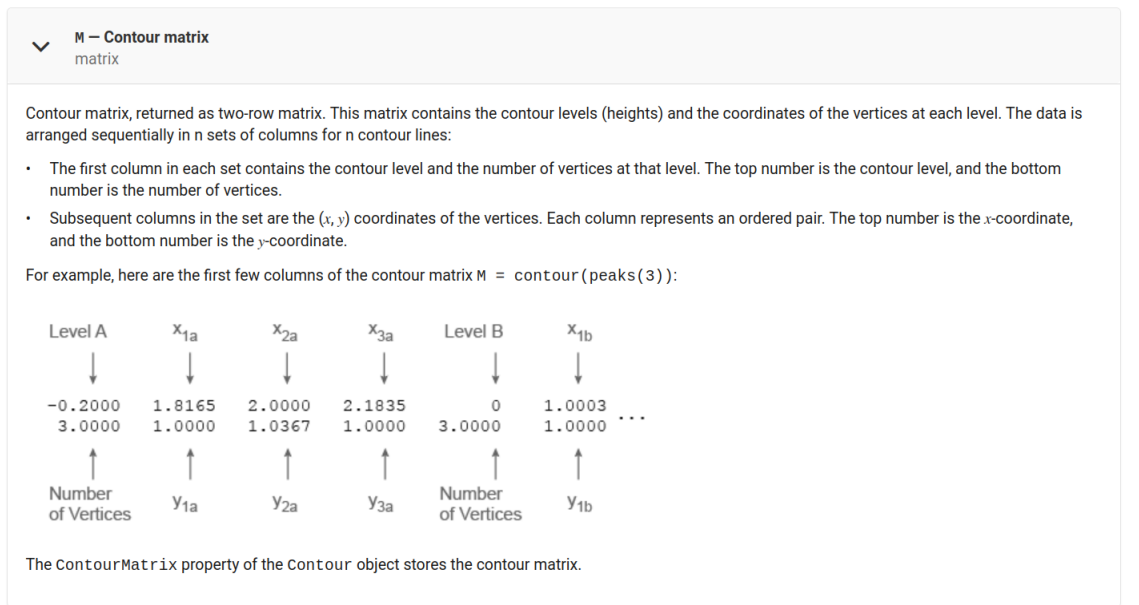**Figure 4.** Caption

C MATLAB code

C.1 main.m

```matlab
1   clear
2   close all
3
4   I = imread('Test1.png');
5   %I = rgb2gray(I);
6
7   [xy,K] = InterfaceCurvature( I , 0.5 , 10 , 10 );
8
9   figure;
10  hold on;
11  [Nc,¬] = size(xy);  % Nc: number of closed contours
12
13  for i=1:Nc
14      plot(xy{i}(:,1),xy{i}(:,2));
15      quiver(xy{i}(:,1),xy{i}(:,2),K{i}(:,1),K{i}(:,2));
16  end
17
18  axis equal;
19  axis tight;
20
21  title('Negative curvature vector field');
22  xlabel('X');
23  ylabel('Y');
```

C.2 InterfaceCurvature.m

```matlab
1   function [ xy , K ] = InterfaceCurvature( I , v , NN , SR )
2
3       Flag = 1;  % Flag = 1 --> Display plots
4                  % Flag = 0 --> Do not display plots
5
6       %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
7       % Description of I/O variables:                                      %
8       %                                                                    %
9       % INPUT:-                                                            %
10      %                                                                    %
11      % I   : image matrix, can be 2D or 3D (RGB)                          %
12      % v   : contour / level-set value, which should be in the range [0,1] %
13      % NN  : number of neighboring points used to fit a circle            %
14      % SR  : sub-sampling rate (should be integer > 0)                    %
15      %                                                                    %
16      % OUTPUT:-                                                           %
17      %                                                                    %
18      % K   : curvature vector (cell of dim equal to number of contours)   %
19      % xy  : coordinates of the points on the contour                     %
20      %                                                                    %
21      %                                                                    %
22      % NOTE: For reliable outputs, the image must be atleast 300x300      %
23      %       and each contour must have atleast 500 points.               %
24      %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
25
26      %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
27      % Author: Jaya Kumar. A
28      % E-mail: jkumar.res@gmail.com
29      %
30      % See the accompanying "doc.pdf" for algorithm details.
31      %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
32
33      %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```matlab
34        %%% Formating the data %%%
35        %%%%%%%%%%%%%%%%%%%%%%%%%%%%
36
37        % Converting a 3D color image to 2D grayscale image
38
39        if length(size(I))==3
40            I = rgb2gray(I);
41        end
42
43        I = double(I);
44        Imin = min(I(:));
45        Imax = max(I(:));
46
47        I = (I-Imin) / (Imax-Imin);  % Normalizing the data [0,1]
48
49        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
50        %%% Padding the data to take care of   %%%
51        %%% interfaces that hit the boundaries %%%
52        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
53
54        [Lx,Ly] = size(I);
55
56        I2 = zeros(Lx+6,Ly+6);
57        I2( 4:Lx+3 , 4:Ly+3 ) = I;
58
59        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%
60        %%% Contour extraction %%%
61        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%
62
63        N = NN;
64        figure;
65        subplot(2,2,1)
66        xy = contour( I2 , [v v] );
67        title('Contours');
68        xlabel('X');
69        ylabel('Y');
70        axis equal;
71        axis tight;
72        %set(gca,'YDir','reverse');
73
74        %%%%%%%%%%%%%%%%%%%%%%%%%%%%
75        %%% Conttour splitting %%%
76        %%%%%%%%%%%%%%%%%%%%%%%%%%%%
77
78        Np = length( xy(1,:) );
79        Lc = xy(2,1);    % Number of points on the 1st contour
80        ll = Lc+1;
81        iLc = 1;
82
83        while( ll < Np )
84            Lc = [Lc ; xy(2,ll+1)];
85            iLc = [iLc ; ll+1];
86            ll = sum( Lc+1 );
87        end
88
89        Nc = length( Lc );    % Number of contours
90
91
92        % Cells to handle multiple contours with different sets of points
93
94        x = cell(Nc,1);    % X-coordinates of the contour
95        y = cell(Nc,1);    % Y-coordinates of the contour
96        r = cell(Nc,1);    % Radius of curvature
97        rr = cell(Nc,1);   %
98        x2 = cell(Nc,1);   % Sampled X-coordinates
99        y2 = cell(Nc,1);   % Sampled Y-coordinates
```

```matlab
100        a = cell(Nc,1);       % X-coordinate of the circle center
101        b = cell(Nc,1);       % Y-coordinate of the circle center
102
103        for i=1:Nc
104            x{i} = xy( 1 , iLc(i)+1 : iLc(i)+Lc(i) )';
105            y{i} = xy( 2 , iLc(i)+1 : iLc(i)+Lc(i) )';
106        end
107
108        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
109        %%% Calculation of curvature %%%
110        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
111
112        for i=1:Nc
113            %%%%%%%%%%%%%%%%%%%%%
114            %%% Sub-sampling %%%
115            %%%%%%%%%%%%%%%%%%%%%
116
117            Np = length(x{i});
118
119            ll = floor( Np/SR );
120            xx = reshape( x{i}(1:SR*ll) , [SR ll] );
121            yy = reshape( y{i}(1:SR*ll) , [SR ll] );
122
123            xx = xx(1,:)';
124            yy = yy(1,:)';
125            Np = ll;
126
127            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
128            %%% Locally fitting circles %%%
129            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
130
131            r{i} = zeros(Np,1);
132            a{i} = zeros(Np,1);
133            b{i} = zeros(Np,1);
134
135            for j=1:Np
136                %%% Maintain contour periodicity
137                j1 = mod( j-NN-1 , Np ) + 1;
138                j2 = mod( j+NN-1 , Np ) + 1;
139
140                if( j1 < j2 )
141                    [a{i}(j),b{i}(j),r{i}(j)] = FitCircle(xx( j1:j2 ),yy( j1:j2 ));
142                else
143                    [a{i}(j),b{i}(j),r{i}(j)] = FitCircle( [xx( j1:end ) ; xx(1:j2)]...
144                                                  ,[yy( j1:end ) ; yy(1:j2)]);
145                end
146            end
147
148            rr{i} = abs(r{i});
149
150            x2{i} = xx;
151            y2{i} = yy;
152            a{i} = real(a{i});
153            b{i} = real(b{i});
154        end
155
156        if( Flag==1 )
157            subplot(2,2,2);
158            plot(rr{1});
159            title('Radius of curvature');
160            hold on;
161            leg = cell(Nc,1);
162            leg{1} = 'Contour1';
163
164            for i=2:Nc
165                plot(abs(rr{i}));
```

```matlab
166             leg{i} = sprintf('Contour%d',i);
167         end
168
169         ylim( [0 , sqrt(Lx^2+Ly^2)] );
170         xlabel('Contour length');
171         ylabel('R');
172         legend( leg );
173
174         subplot(2,2,4);
175         plot3(x2{1},y2{1},rr{1},'-*');
176         title('3D plot of radius of curvature');
177         hold on;
178
179         for i=2:Nc
180             plot3(x2{i},y2{i},rr{i},'-*');
181         end
182
183         box on;
184         zlim( [0 , sqrt(Lx^2+Ly^2)] );
185         xlabel('X');
186         ylabel('Y');
187         zlabel('R');
188     end
189
190     v = cell(Nc,1);
191     K = cell(Nc,1);
192
193     for i=1:Nc
194         v{i} = [(x2{i}-a{i}) (y2{i}-b{i})];
195         vv = sqrt( v{i}(:,1).^2 + v{i}(:,2).^2 );
196         v{i} = v{i} ./ [vv vv];
197
198         K{i} = v{i} ./ [rr{i} rr{i}];
199     end
200
201     if( Flag==1 )
202         subplot(2,2,3);
203         [m,n] = size(I2);
204         I3 = zeros(m,n,3);
205         I3(:,:,1) = I2;
206         I3(:,:,2) = I2;
207         I3(:,:,3) = I2;
208         %imagesc(I3);
209         hold on;
210
211         for i=1:Nc
212             plot(x2{i},y2{i});
213             h = quiver(x2{i},y2{i},K{i}(:,1),K{i}(:,2));
214             set(h,'AutoScale','on', 'AutoScaleFactor', 2)
215         end
216
217         axis equal;
218         axis tight;
219         xlim([0 Ly]);
220         ylim([0 Lx]);
221         title('Negative curvature vector field');
222         xlabel('X');
223         ylabel('Y');
224     end
225
226     xy = cell(Nc,1);
227
228     for i=1:Nc
229         xy{i} = [x2{i} y2{i}];
230     end
231
```

```
232    end
```

C.3 FitCircle.m

```matlab
1    function [a,b,r] = FitCircle(x,y)
2
3    N = length(x);
4    NN = 10;
5    eta = 0.001;
6
7    n = floor(N/2);
8
9    x0 = sum(x)/N;
10   y0 = sum(y)/N;
11
12   x = x - x0;
13   y = y - y0;
14
15   x1 = x(1);
16   y1 = y(1);
17   x2 = x(n);
18   y2 = y(n);
19   x3 = x(N);
20   y3 = y(N);
21
22   x12 = x1-x2;
23   x23 = x2-x3;
24   x31 = x3-x1;
25
26   y12 = y1-y2;
27   y23 = y2-y3;
28   y31 = y3-y1;
29
30   Num = (x12^2 + y12^2) * (x23^2 + y23^2) * (x31^2 + y31^2);
31   Den = (x1 * y23 + x2 * y31 + x3 * y12);
32
33   r = abs( 0.5 * sqrt(Num) / Den );
34
35   Num = x1^2 * y23 + x2^2 * y31 + x3^2 * y12 - y12 * y23 * y31;
36   a = 0.5 * Num / Den;
37
38   Num = y1^2 * x23 + y2^2 * x31 + y3^2 * x12 - x12 * x23 * x31;
39   b = 0.5 * Num / (-Den);
40
41   s = r;
42
43   r = r/s;
44   x = x/s;
45   y = y/s;
46   a = a/s;
47   b = b/s;
48
49   for i=1:NN
50       E = 0;
51       dEda = 0;
52       dEdb = 0;
53       dEdr = 0;
54
55       for j=1:N
56           dE = (x(j)-a)^2 + (y(j)-b)^2 - r^2 ;
57           E = E + dE^2;
58
59           dEda = dEda - 4 * dE * (x(j)-a);
60           dEdb = dEdb - 4 * dE * (y(j)-b);
```

```
61            dEdr = dEdr - 4 * dE * r;
62        end
63
64        r = r - eta * dEdr;
65        a = a - eta * dEda;
66        b = b - eta * dEdb;
67    end
68
69    a = s * a;
70    b = s * b;
71    r = s * r;
72
73    a = a + x0;
74    b = b + y0;
75
76    return;
```