Residual Stress Database for Coldworked Holes

Contents

1 Database Interface

The residual stress database rs_database.dll is a .Net assembly which can be COM-registered along with its type library rs_database.tlb. By default the database looks for a subfolder "db" which contains .rs files. Each .rs file represents one database entry with a set of parameters (described by the DBParams struct) and a residual stress point cloud. The file format is described in section 2.

To register the database dll, enter the following in a command prompt with administrator privileges:

```
>cd C:\Windows\Microsoft.NET\Framework\[your .net framework version here]

>regasm "[path to dll]\rs_database.dll" /codebase /tlb
```

1.1 Methods and Properties

string[] AvailableMaterials()

Returns a list of unique material names in the database.

Parameter	Type	Description
Return value	string[]	List of unique material names.

DBParams[] DatabaseEntries()

Returns a list of all entries in the database.

Parameter	Type	Description
Return value	DBParams[]	List of parameters describing all database entries.

DBResult ResidualStress(x, y, param)

Interpolates between entries in the database with the set of parameters described in param. The interpolation method is described in section 3.

Parameter	Type	Description
x	double[]	First coordinate of (x,y) points at which residual stress will be interpolated.
y	double[]	Second coordinate of (x,y) points at which residual stress will be interpolated. Must be the same length at x.
param	DBParam	Set of database entry parameters.
Return value	DBResult	Return data including interpolated residual stress.

string DBFolder

Gets and sets the folder containing database .rs files.

1.2 Objects

Object	Field	Type	Description
DBParams	material	string	Material name
	th	double	Thickness
	dia	double	Diameter
	ed	double	Edge distance (from center of hole to edge)
	pctCX	double	Percent coldwork
DBResult	success	boolean	Successful interpolation? True/False
	RS	double[]	Interpolated residual stress at points (x,y). Same length as input x and y.
	message	string	If success == false, this is an error message describing the failure. If success == true, this identifies which database entries were selected for interpolation.

2 Database File Format

The database entry .rs file format is an ASCII format with the following structure. Any number of (x,y,Sz) points may be specified. Any number of digits after the decimal place is acceptable. Dimensional units must be inches and stress units must be ksi. Comments can be placed on any line after a '%' character. The (x,y) points must define a rectangular grid but can be in any order with arbitrary spacing.

```
materialName (string)
comments (string)
th (double)
dia (double)
ed (double)
pctCX (double)
x1 (double), y1 (double), Sz1 (double)
x2 (double), y2 (double), Sz2 (double)
x3 (double), y3 (double), Sz3 (double)
x4 (double), y4 (double), Sz4 (double)
x5 (double), y5 (double), Sz5 (double)
```

For example:

```
7075-T6          % material
3K1-01-C         % comments
0.100000         % thickness
0.375000         % diameter
0.900000         % edge distance
4.000000         % coldwork percentage
0.000000, 0.000000, -32.100000
```

0.000000, 0.002000, -35.500000
0.000000, 0.004000, -38.700000
0.000000, 0.006000, -40.900000
0.000000, 0.008000, -42.600000
0.000000, 0.011000, -43.900000
0.000000, 0.013000, -44.800000
0.000000, 0.016000, -45.500000
0.000000, 0.018000, -45.900000
0.000000, 0.021000, -46.100000
0.000000, 0.024000, -46.100000

The database interpolator assumes that each entry has a unique set of the (material, th, dia, ed, pctCX) parameters. The (x,y) origin is the edge of the hole at the mandrel entrance face.

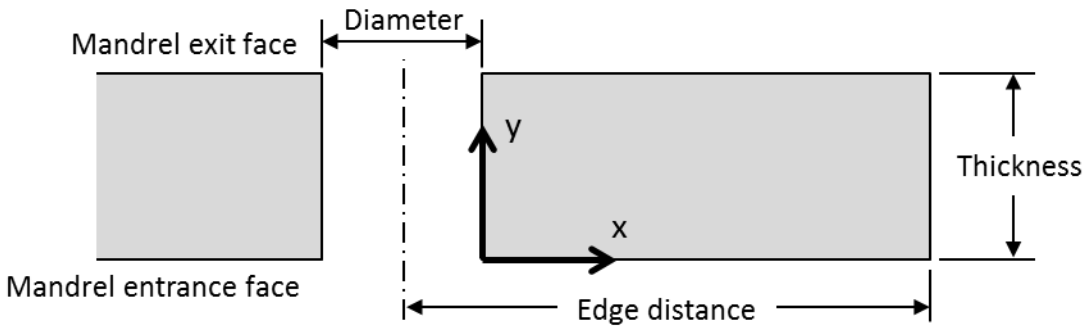


Figure 2.1: Definition of specimen geometric parameters.

3 Extrapolation Limits

The input parameters for the database are limited to the following space:

- 0 <= x <= ed – dia/2
- 0 <= y <= th
- 0 <= dia
- 0 <= th
- ed < th
- 0 < pctCX

4 Interpolation Procedure

The database interpolator uses a 4D Delaunay simplex finder. A simplex is found which encloses the given query/interpolation point described by input parameters DBParam. The vertices of the simplex are database entries, and the query point is interpolated between the vertices.

4.1 Overview

The RS database includes the following spaces (dimensions):

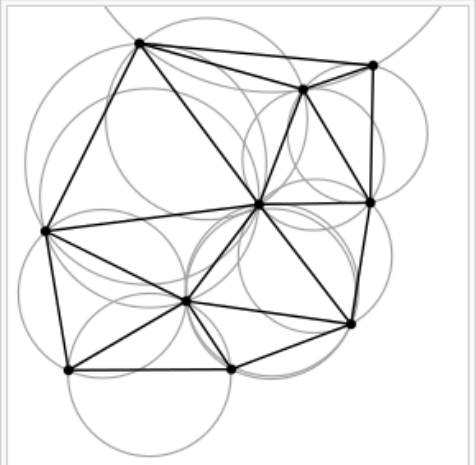
1. Thickness
2. Diameter
3. Edge distance
4. Percent coldwork
5. Material
6. x
7. y

If entries in the database are segregated by material and all (x,y) points are scaled to a standard domain of size 2 x 2, a four dimensional (4D) interpolation space is left. This algorithm finds a D-dimensional "triangle", referred to a D-dimensional simplex, which contains a query point with 4D coordinates that are the desired th, dia, ed, and pctCX and has vertices which are 4D database entry points.

The ideal case would be a D-dimensional Delaunay triangulation of the database entries composed of D-dimensional simplexes. A direct search method is found to be efficient enough and robust for finding the enclosing Delaunay simplex.

A Delaunay triangulation for a two dimensional point cloud produces a triangulation so that no point is inside the circumcircle of any triangle. The vertices of the triangles are the data in the point cloud.

A multidimensional Delaunay triangulation would produce a set of simplexes so that no point lies within the hypercircumsphere of any simplex. The vertices of the simplexes are the data in the point cloud.



A Delaunay triangulation in the plane with circumcircles shown

Figure 4.1. Source: wikipedia.org

4.2 Definitions

Point – A 4D point with coordinates (th, dia, ed, pctCX).

Q – The 4D query point (interpolation point) of the database.

P – A 4D entry in the database.

Simplex – A D-dimensional "triangle". A triangle is a 2D simplex. A tetrahedron is a 3D simplex. A simplex has D+1 vertices.

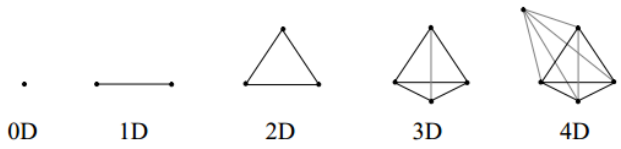


Figure 4.2: Simplexes in 0 through 4 dimensions.

Barycentric coordinates – Coordinates for a simplex that equal 1.0 at one vertex and 0.0 at the remaining vertex. In StressCheck this is analogous to triangular coordinates. Barycentric coordinates also provide weights for linear combinations of database entries to produce the residual stress (RS) distribution at the query point Q.

4.3 Basic Algorithm

1. See if Q exactly matches any P. If so, go to step 9 for just one point P.
2. Filter database data by material and remove duplicate entries.
3. Check points P for constant dimensions and remove that dimension from the search space.
    a. For example, the initial database data all has the same coldwork percentage so the coldwork percentage dimension is removed from the search.
    b. As a consequence, any query point Q must have the exact same constant dimension value (e.g. 4% CW) as the data in the database.
4. Loop through all possible simplexes connecting all database points P.
5. If a simplex contains the query point Q, go on. Otherwise return to step 4.
6. Compute the hypercircumsphere for the simplex containing Q.
7. If the hypercircumsphere does not contain any other point P its simplex is a Delaunay simplex which encloses Q; go on. If it contains another point, return to step 4.
8. Compute the barycentric coordinates L_i for point Q relative to each vertex P_i in the Delaunay simplex.
9. For a point $Q(x, y)$ in RS surface Q, map the coordinates to the standard domain (ξ, η) and linearly interpolate the surface P_i on the standard domain. The RS surface Q is a linear combination of the simplex vertices' RS surfaces.

$$Q(\xi, \eta) = \sum_{i=1}^{N} L_i P_i(\xi, \eta)$$

10. Done.
11. If an enclosing simplex cannot be found then Q must be extrapolated. For an extrapolated Q, just return the closest point P and interpolate on the standard domain as described in step 9. Done.

Matlab code for finding an enclosing Delaunay simplex in 2D is provided in section 4.5. The implemented interpolator is an N-dimensional interpolator for any number of dimensions N.

4.4 Mathematical Details

4.4.1 Barycentric Coordinates

For an enclosing simplex with N vertices in (N-1) dimensional space there are N barycentric coordinates which have a range $0 \leq L_i \leq 1$. If any coordinate is less than zero or greater than 1 then the simplex does not enclose the query/interpolation point Q.

Examples are provided for 1D, 2D, 4D, and then are generalized.

Barycentric Coordinates in 1D

For two simplex vertices P_1 and P_2 (which define a line) the query point $\{Q\}$ is expressed as

$$\{Q\} = L_1\{P_1\} + L_2\{P_2\}$$

where points P_i have coordinates on A-B axes $\begin{Bmatrix} P_i^A \\ P_i^B \end{Bmatrix}$ and the sum of the coordinates is one:

$$L_1 + L_2 = 1$$

Solving this linear system produces the coordinates:

$$\begin{bmatrix} P_1^A & P_2^A \\ 1 & 1 \end{bmatrix} \begin{Bmatrix} L_1 \\ L_2 \end{Bmatrix} = \begin{Bmatrix} Q^A \\ 1 \end{Bmatrix}$$

Barycentric Coordinates in 2D

For three simplex vertices P_1, P_2, P_3 (which define a triangle) the query point $\{Q\}$ is expressed as

$$\{Q\} = L_1\{P_1\} + L_2\{P_2\} + L_3\{P_3\}$$

where points P_i have coordinates on A-B-C axes $\begin{Bmatrix} P_i^A \\ P_i^B \\ P_i^C \end{Bmatrix}$ and the sum of the coordinates is one:

$$L_1 + L_2 + L_3 = 1$$

Solving this linear system produces the coordinates:

$$\begin{bmatrix} P_1^A & P_2^A & P_3^A \\ P_1^B & P_2^B & P_3^B \\ 1 & 1 & 1 \end{bmatrix} \begin{Bmatrix} L_1 \\ L_2 \\ L_3 \end{Bmatrix} = \begin{Bmatrix} Q^A \\ Q^B \\ 1 \end{Bmatrix}$$

Barycentric Coordinates in 4D

For five simplex vertices P_1, P_2, P_3, P_4, P_5 the query point $\{Q\}$ is expressed as

$$\{Q\} = L_1\{P_1\} + L_2\{P_2\} + L_3\{P_3\} + L_4\{P_4\} + L_5\{P_5\} = \sum_{i=1}^{5} L_i\{P_i\}$$

where points P_i have coordinates on A-B-C-D axes and the sum of the coordinates is one:

$$\sum_{i=1}^{5} L_i = 1$$

Solving this linear system produces the barycentric coordinates.

$$\begin{bmatrix} P_1^A & P_2^A & P_3^A & P_4^A & P_5^A \\ P_1^B & P_2^B & P_3^B & P_4^B & P_5^B \\ P_1^C & P_2^C & P_3^C & P_4^C & P_5^C \\ P_1^D & P_2^D & P_3^D & P_4^D & P_5^D \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{Bmatrix} L_1 \\ L_2 \\ L_3 \\ L_4 \\ L_5 \end{Bmatrix} = \begin{Bmatrix} Q^A \\ Q^B \\ Q^C \\ Q^D \\ 1 \end{Bmatrix}$$

For the residual stress database the axes A-B-C-D are:

$$\begin{Bmatrix} A \\ B \\ C \\ D \end{Bmatrix} = \begin{Bmatrix} diameter \\ edge\ distance \\ \%\ coldwork \\ thickness \end{Bmatrix}$$

Generalization

For N simplex vertices in N-1 space with A-B-C-…-(N-1) axes the query point $\{Q\}$ is expressed as

$$\{Q\} = \sum_{i=1}^{N} L_i\{P_i\}$$

so that

$$\sum_{i=1}^{N} L_i = 1$$

Solving this linear system produces the coordinates:

$$\begin{bmatrix} P_1^A & P_2^A & \cdots & P_N^A \\ P_1^B & P_2^B & \cdots & P_N^B \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{bmatrix} \begin{Bmatrix} L_1 \\ L_2 \\ \vdots \\ L_N \end{Bmatrix} = \begin{Bmatrix} Q^A \\ Q^B \\ \vdots \\ 1 \end{Bmatrix}$$

$$[P]\{L\} = \{Q\}$$

4.4.2 Barycentric Coordinates with Coplanar Vertices

If the N simplex vertices are coplanar then barycentric coordinates cannot be computed for an (N-1) dimensional simplex. This case is detected if there are any linearly dependent rows in the matrix $[P]$ described in the previous section:

$$\begin{bmatrix} P_1^A & P_2^A & \cdots & P_N^A \\ P_1^B & P_2^B & \cdots & P_N^B \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{bmatrix} \begin{Bmatrix} L_1 \\ L_2 \\ \vdots \\ L_N \end{Bmatrix} = \begin{Bmatrix} Q^A \\ Q^B \\ \vdots \\ 1 \end{Bmatrix}$$

$$[P]\{L\} = \{Q\}$$

Since $[P]$ is a square matrix, this condition can also be detected if $|P| = 0$.

Simplexes with coplanar vertices are skipped.

4.4.3 Hypercircumspheres

In a 2D Delaunay triangulation each triangle is constructed so that there are no other points within its circumcircle. A 3D Delaunay triangulation is constructed so that there are no other points within each tetrahedron's circumsphere. An (N-1) dimensional triangulation (with N-vertex simplexes) is constructed so that there are no other points within each simplex's hypercircumsphere.

Examples follow for 2D and 4D.

2D Circumcircle

In 2D the equation of a circle with center (A_0, B_0) and radius r is

$$(A - A_0)^2 + (B - B_0)^2 = r^2$$

There are three unknowns in this equation and therefore three points are required to define the circle. The circle equations for points (A_1, B_1) and (A_2, B_2) have equal radii and therefore are equal:

$$(A_1 - A_0)^2 + (B_1 - B_0)^2 = (A_2 - A_0)^2 + (B_2 - B_0)^2$$

This is rearranged so that:

$$2(A_2 - A_1)A_0 + 2(B_2 - B_1)B_0 = A_2^2 + B_2^2 - A_1^2 + B_1^2$$

Writing the same equality for points (A_1, B_1) and (A_3, B_3), a system of equations can be formed:

$$\begin{bmatrix} 2(A_2 - A_1) & 2(B_2 - B_1) \\ 2(A_3 - A_1) & 2(B_3 - B_1) \end{bmatrix} \begin{Bmatrix} A_0 \\ B_0 \end{Bmatrix} = \begin{Bmatrix} A_2^2 + B_2^2 - A_1^2 - B_1^2 \\ A_3^2 + B_3^2 - A_1^2 - B_1^2 \end{Bmatrix}$$

Solving the system provides the center of the circle. The radius is then determined with the circle equation at one point.

4D Circumhypersphere

The center of a 4D sphere (hypersphere) defined by 5 points is obtained by writing a similar set of linear equations.

$$(A - A_0)^2 + (B - B_0)^2 + (C - C_0)^2 + (D - D_0)^2 = r^2$$

$$\begin{bmatrix} 2(A_2 - A_1) & 2(B_2 - B_1) & 2(C_2 - C_1) & 2(D_2 - D_1) \\ 2(A_3 - A_1) & 2(B_3 - B_1) & 2(C_3 - C_1) & 2(D_3 - D_1) \\ 2(A_4 - A_1) & 2(B_4 - B_1) & 2(C_4 - C_1) & 2(D_4 - D_1) \\ 2(A_5 - A_1) & 2(B_5 - B_1) & 2(C_5 - C_1) & 2(D_5 - D_1) \end{bmatrix} \begin{Bmatrix} A_0 \\ B_0 \\ C_0 \\ D_0 \end{Bmatrix} = \begin{Bmatrix} A_2^2 + B_2^2 + C_2^2 + D_2^2 - A_1^2 - B_1^2 - C_1^2 - D_1^2 \\ A_3^2 + B_3^2 + C_3^2 + D_3^2 - A_1^2 - B_1^2 - C_1^2 - D_1^2 \\ A_4^2 + B_4^2 + C_4^2 + D_4^2 - A_1^2 - B_1^2 - C_1^2 - D_1^2 \\ A_5^2 + B_5^2 + C_5^2 + D_5^2 - A_1^2 - B_1^2 - C_1^2 - D_1^2 \end{Bmatrix}$$

The radius is again determined with the hypersphere equation at one point.

4.5 Example

Given the 7075-T6 and 7075-T651 data provided by APES to ESRD for testing the database, the parameters in Figure 4.3 can be used as a sample interpolation to produce the surface shown in Figure 4.4. These parameters define a point that is roughly in the center of the available interpolation space.

Material:	7075-T651	
Percent coldwork:	4	%
Thickness:	0.4	in
Diameter:	0.35	in
Edge distance:	0.75	in

Figure 4.3

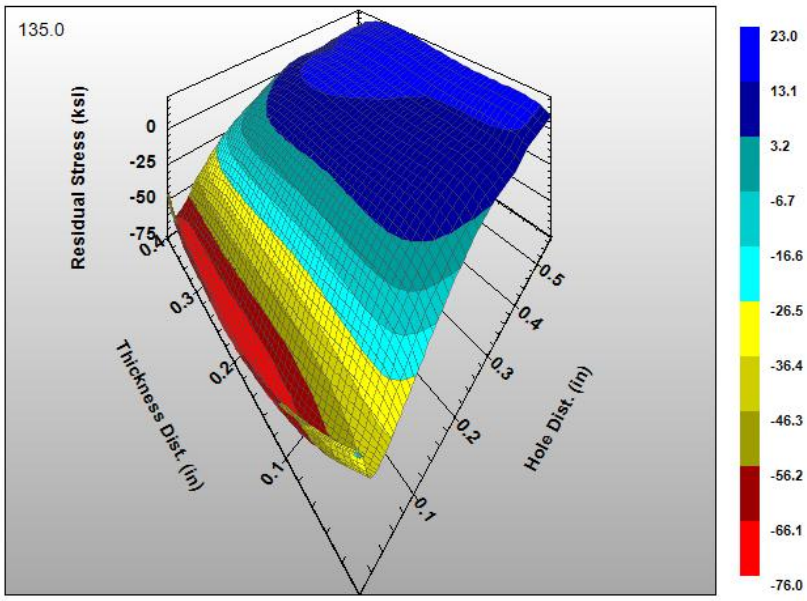
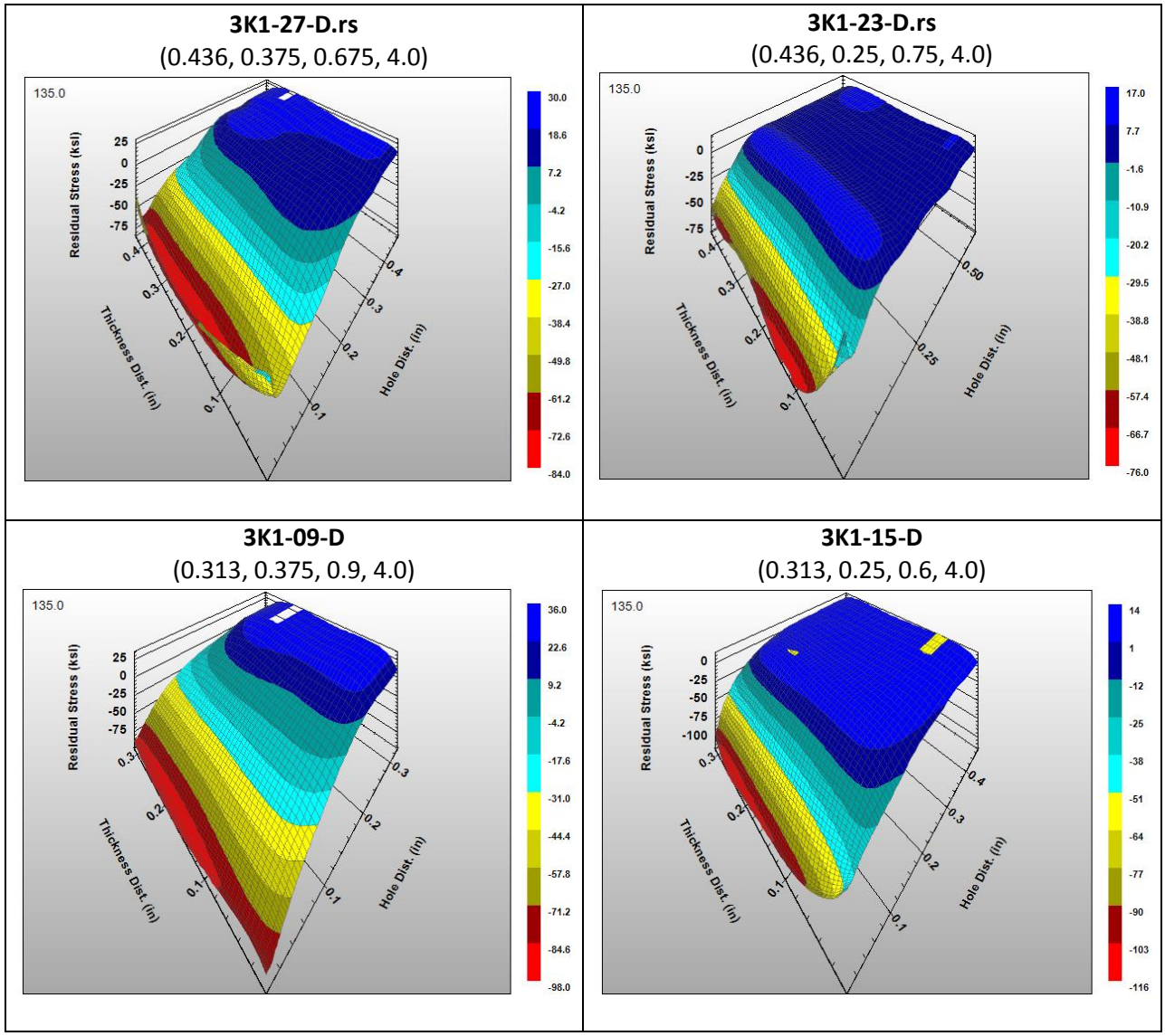


Figure 4.4: Sample interpolated surface

This surface is interpolated from the four database entries in Table 4.1.

Table 4.1: Database entries which are interpolated to produce Figure 4.4.



4.6 Matlab Code for 2D Interpolation

```
clear all;
clf;
close all;

seed = floor(rand*10000);
% rng('default');
```

```
disp(sprintf('Seed: %5d', seed));
rng(seed);

n = 50;
% Random pointcloud
Px = rand(n, 1);
Py = rand(n, 1);

% Random interpolation point
Qx = rand;
Qy = rand;

plot(Px, Py, '.k');
hold on;
plot(Qx, Qy, 'Ob');
axis([0 1 0 1]);

for i=1:length(Px)
    text(Px(i)+.01, Py(i), num2str(i));
end

% Find all enclosing simplexes
found = false;
for i=1:length(Px)
    for j=i+1:length(Px)
        for k=j+1:length(Px)

            % Compute barycentric coordinates
            L = [Px(i) Px(j) Px(k); Py(i) Py(j) Py(k); 1 1 1] \ [Qx Qy 1]';

            % Does the triangle contain the point?
            if all(L >= 0) && all(L <= 1)

                % Compute the circumcircle
                center = [-2*Px(i)+2*Px(j) -2*Py(i)+2*Py(j); ...
                          -2*Px(i)+2*Px(k) -2*Py(i)+2*Py(k)] \ ...
                         [-Px(i)^2-Py(i)^2+Px(j)^2+Py(j)^2; ...
                          -Px(i)^2-Py(i)^2+Px(k)^2+Py(k)^2];
                radius2 = (Px(i) - center(1))^2 + (Py(i) - center(2))^2;

                % See if any points are within the circle
                % If so, this is not a Delaunay triangle
                hasInteriorPoint = false;
                for m=1:length(Px)
                    if m == i
                        continue
                    elseif m == j
                        continue
                    elseif m == k
                        continue
                    end
                    dist2 = (Px(m) - center(1))^2 + (Py(m) - center(2))^2;
                    if dist2 < radius2
```

```
                        hasInteriorPoint = true;
                        break;
                    end
                end

                if ~hasInteriorPoint
                    found = true;
                    break
                end
            end
            if found
                break;
            end
        end
        if found
            break;
        end
    end
    if found
        break;
    end
end

if found
    plot([Px(i) Px(j) Px(k) Px(i)], [Py(i) Py(j) Py(k) Py(i)], '-k');

    radius = sqrt(radius2);
    theta = linspace(0, 2*pi, 50);
    circleX = center(1) + radius * cos(theta);
    circleY = center(2) + radius * sin(theta);
    plot(circleX, circleY, '-b');
else
    disp('Point Q is outside interpolation space');
end
```