8. **Function for min. total distance to a set of hyperplanes**: Write a function minDist2hyperplanes(A) that returns a point in a d-dimensional space such that the total distance between this point and all the other n hyperplanes is minimized, where these n hyperlines can be packed into a (d+1)-by-n matrix A, with A(:,i) being the coefficients of the i-th hyperplane: $A(1, i) * x_1 + A(2, i) * x_2 + \cdots + A(d, i) * x_d + A(d+1, 1) = 0$.

   Hint
   - Since there is no analytic solution, you need to use "fminsearch" (with default options) to search for the point.
   - Analytic solution of this problem exists if the "total distance" is replaced by "total squared distance".

9. **Circle fitting via DSS**: A circle in 2D can be described by the following equation

$$(x-a)^2 + (y-b)^2 = r^2,$$

   where (a, b) is the center and r is the radius of the circle. Given a dataset $\{(x_i, y_i) | i = 1, 2, \ldots, n\}$, the sum of distances of these points to the circle can be formulated as follows:

$$f(a, b, r) = \sum_{i=1}^{n} \left| \sqrt{(x_i - a)^2 + (y_i - b)^2} - r \right|$$

   Write a function circleFitByDss.m that can find the best values of $[a, b, r]^T$ such that $f(a, b, r)$ can be minimized. The usage of the function is:

   output = circleFitByDss(data)

   where
   - data: an 2-by-n dataset matrix, with each column being a sample data point.
   - output: a column vector of the derived $[a, b, r]^T$ using Downhill Simplex Search (which is implemented as the function "fminsearch" in MATLAB).

   Note that the initial guess of $[a, b, r]^T$ should be as close as possible to the minimizing point. One good choice is to set the center ($[a, b]^T$) as the mean of all the data points, and set r as the average distance of the center to each data point.

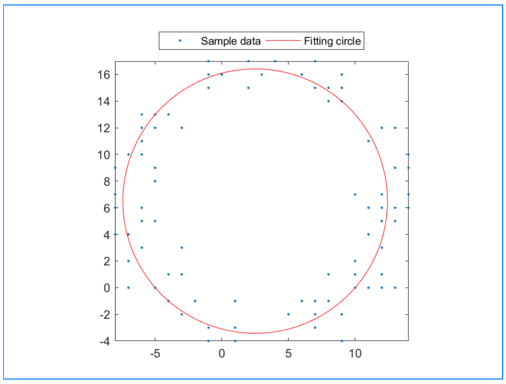
   Here is a test example:

   a. Example 2: 08-一般數學函數的處理與分析/circleFit01.m

```
data=[12 -5 13 -3 -8 3 7 -4 7 -4 -4 -1 -5 2 13 -5 -6 5 -5 -8 -6 -7 2 9 -5 -4 -5 8 4 -1 12 1 7 11 11 -1 9 0 -5 14 -3 -8 -3 12 -1 -1 13 7 -6 -7 12 1
     3 13 6 1 9 16 15 1 17 13 -1 -3 9 17 12 5 3 -2 0 7 6 10 15 -4 8 -1 0 14 17 16 12 -3 -3 6 11 -3 14 16 13 9 -2 6 12 7 -4 15 5 -3 12 4 7 5 4 2 -1 6 1(
theta=circleFit(data);
format long; theta
% Plotting
t=linspace(0, 2*pi);
x1=theta(1)+theta(3)*cos(t);
y1=theta(2)+theta(3)*sin(t);
plot(data(1,:), data(2,:), '.', x1, y1, 'r');
axis image
legend({'Sample data', 'Fitting circle'}, 'location', 'northOutside', 'orientation', 'horizontal');
```

```
theta =

   2.500014916354736
   6.500011175675343
   9.924708551335694
```



10. **Rectangle fitting via DSS**: A rectangle in 2D can be described by 4 parameters $[x, y, \alpha, \beta]$, where $[x, y]$ is the center coordinate of the rectangle, α is the half width, and β is the half height. (Thus the 4 corners of the rectangle can be represented by 4 points at $[x-\alpha, y-\beta]$, $[x-\alpha, y+\beta]$, $[x+\alpha, y+\beta]$, and $[x+\alpha, y-\beta]$.) Given a point in 2D, the distance of this point to the rectangle is defined as the shortest distance of this point to all possible data points on the rectangle. Write a function rectangleFit.m that can find the best values of $[x, y, \alpha, \beta]^T$ such that the total distance of a dataset to the rectangle can be minimized. The usage of the function is: