POLITECNICO DI MILANO

Fitting algorithms

Computer Aided Manufacturing

Stefano Petrò

stefano.petro@polimi.it

Distance Functions

Create matlab functions for evaluating the distance

- 2D point to point (**d2Dpp**)
  - The inputs are a n x 2 matrix, each row contains the [x y] coordinates of a point, and a [x0 y0] vector, where x0, y0 are the coordinates of the reference point. The output is a n x 1 vector with the distances.
- 2D point to straight line (**d2Dsl**)
  - The inputs are a n x 2 matrix, each row contains the [x y] coordinates of a point, and a [a b c] vector, where a, b, c are the parameters of the straight line. The output is a n x 1 vector with the distances.
- 3D point to point (**d3Dpp**)
  - The inputs are a n x 3 matrix, each row contains the [x y z] coordinates of a point, and a [x0 y0 z0] vector, where x0, y0, z0 are the coordinates of the reference point. The output is a n x 1 vector with the distances.
- 3D point to plane (**d3Dp**)
  - The inputs are a n x 3 matrix, each row contains the [x y z] coordinates of a point, and a [a b c d] vector, where a, b, c, d are the parameters of the plane. The output is a n x 1 vector with the distances.
- 3D point to straight line (**d3Dsl**)
  - The inputs are a n x 3 matrix, each row contains the [x y z] coordinates of a point, a [a b c] vector, where a, b, c define the direction of the straight line, and a [x0 y0 z0] vector, where x0, y0, z0 are the coordinates of a point belonging to the straight line. The output is a n x 1 vector with the distances.

Distance Functions - reference

2D point to straight line

$$d = \frac{ax + by + c}{\sqrt{a^2 + b^2}}$$

3D point to plane

$$d = \frac{ax + by + cz + d}{\sqrt{a^2 + b^2 + c^2}}$$

2D point to point

$$d = \sqrt{(x - x_0)^2 + (y - y_0)^2}$$

3D point to straight line

$$d = \frac{\left\| \begin{bmatrix} a \\ b \\ c \end{bmatrix} \times \begin{bmatrix} x - x_0 \\ y - y_0 \\ z - z_0 \end{bmatrix} \right\|}{\sqrt{a^2 + b^2 + c^2}}$$

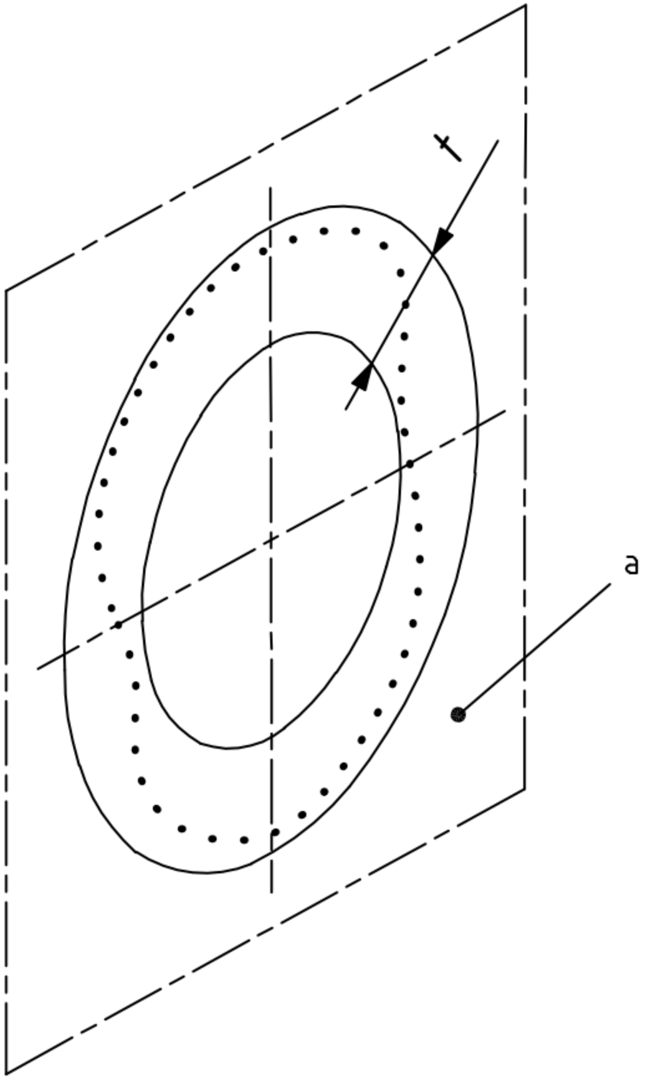
3D point to point

$$d = \sqrt{(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2}$$

Es. 1: least squares roundness

Develop an algorithm for the calculation of the least squares circle and based on this evaluate the roundness deviation. Plot the cloud of points and the LS circle.

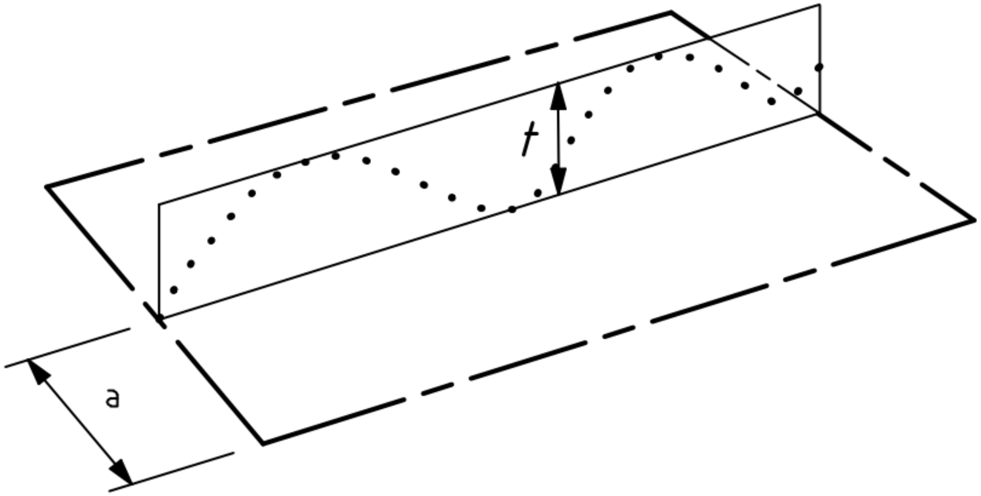
Hint: **function_handle** (see MATLAB help), anonymous function call to d2Dpp will be required. Use **lsqnonlin**.



Es. 2: Minimum zone straightness

Develop an algorithm for the calculation of the Minimum zone straightness and based on this evaluate the straightness deviation. Plot the cloud of points and the tolerance zone.

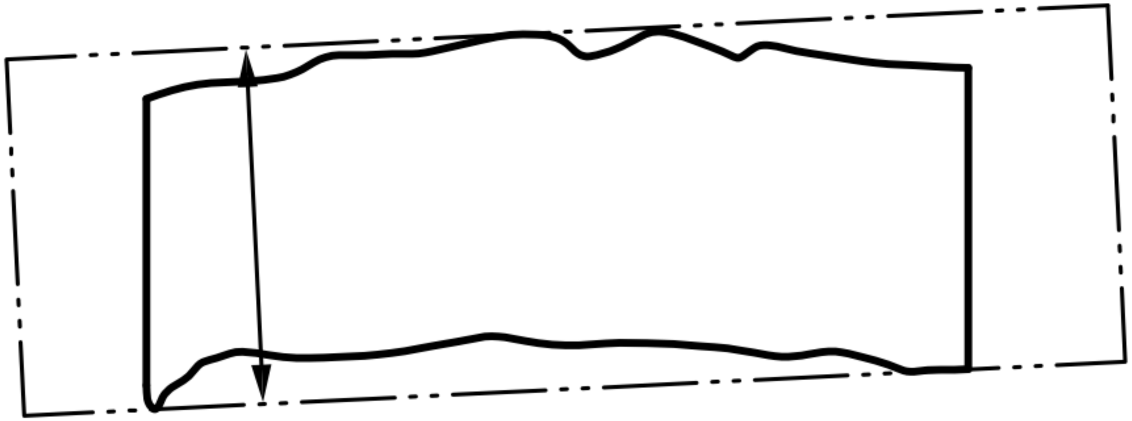
Hint: **function_handle** (see MATLAB help), anonymous function call to d2Dsl will be required. Use **fminimax**.



Es. 3: Minimum circumscribed cylinder

Develop an algorithm for the calculation of the Minimum circumscribed cylinder

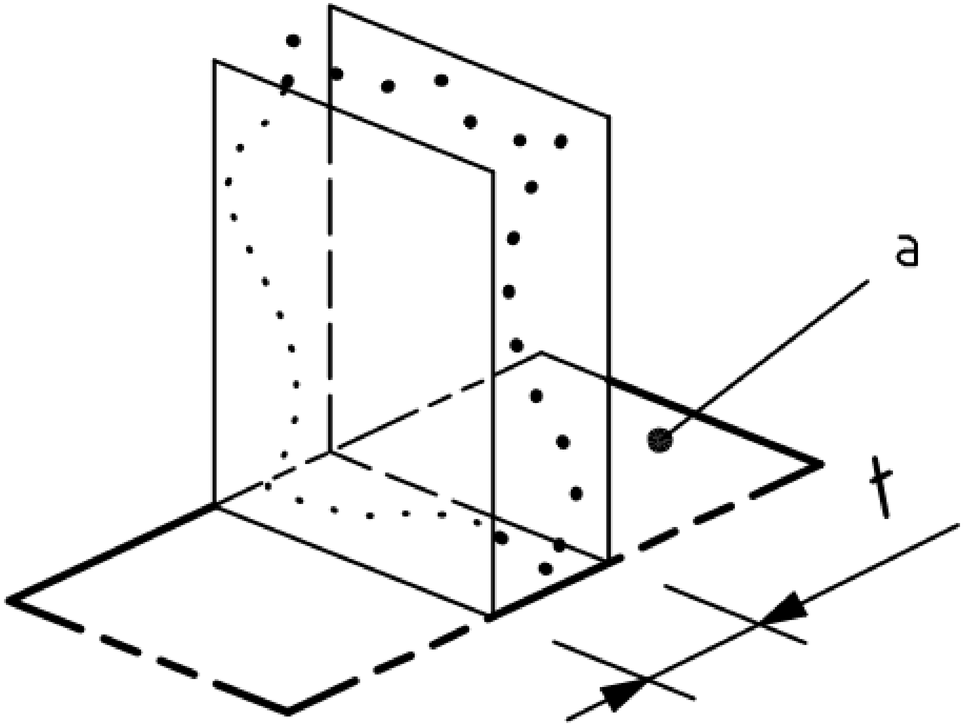
Hint: **function_handle** (see MATLAB help), anonymous function call to d3Dsl will be required. Use **fminimax**.



Es. 4: Planes perpendicularity

Develop an algorithm for the calculation of the perpendicularity of a couple of planes

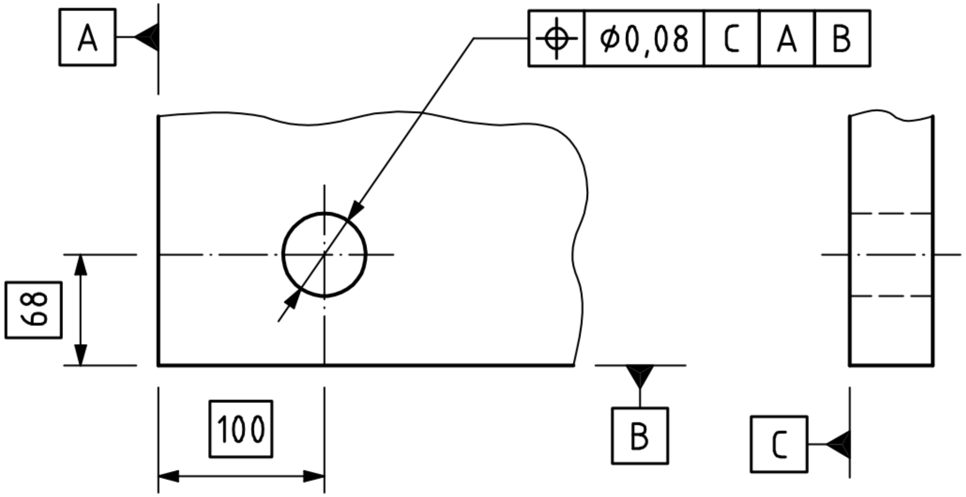
Hint: **function_handle** (see MATLAB help), anonymous function call to d3Dp will be required. Use **fminimax** and **lsqnonlin**.



Es. 5: Location

Develop an algorithm for the calculation of the location of a hole, as in figure

Hint: **function_handle** (see MATLAB help), anonymous function call to d3Dp will be required. Use **lsqnonlin**.


A
⌖ ⌀0,08 C A B
68
100
B
C