

## Math 100 — MATLAB Project

---

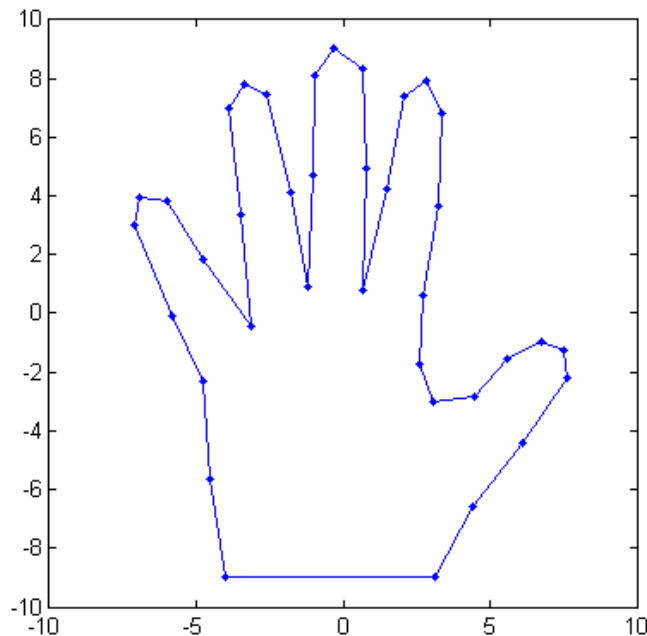
For this project you will need several MATLAB files:

- `hand.m`, `dot_to_dot.m` and `wiggle.m` from *Experiments with MATLAB*.
- `splinetx.m`, `pchiptx.m` and `tridisolve.m` from *Numerical Computing with MATLAB*.

You can find all files at their original MathWorks website or at [http://math.iit.edu/~fass/100\\_handouts.html](http://math.iit.edu/~fass/100_handouts.html). Helpful background reading should be provided by (or some other equivalent source)

- Chapter 5 (Matrices) of *Experiments with MATLAB*:  
<http://www.mathworks.com/moler/exm/chapters/matrices.pdf>,
- Chapter 3 (Interpolation) of *Numerical Computing with MATLAB*:  
<http://www.mathworks.com/moler/interp.pdf>,
- Chapter 11 (Parametric Equations and Polar Coordinates) of your Stewart Calculus book.

1. Get started by reproducing the following plot. Use the functions `hand` and `dot_to_dot` to do so.



2. Make a plot of your own hand. Start with

```
figure('position',get(0,'screensize'))
axes(10*[-1 1 -1 1])
[x,y] = ginput;
```

Place your hand on the computer screen. Use the mouse to select a few dozen points outlining your hand. Terminate the `ginput` with a carriage return. You might find it easier to trace your

hand on a piece of paper and then put the paper on the computer screen. You should be able to see the `ginput` cursor through the paper.

The data you have collected forms two column vectors, `x` and `y`, with entries in the range from -10 to 10. You can arrange the data as two rows in a single matrix with

```
H = [x y]';
```

Then you can work with your hand data by using commands such as `dot_to_dot(H)`.

You can save your data in the file `myhand.mat` with `save myhand H` and retrieve it in a later MATLAB session with `load myhand`.

3. Now think of  $x$  and  $y$  as two functions of an independent variable,  $t$ , that goes from one to the number of points you collected, so together  $x = x(t)$  and  $y = y(t)$  can be viewed as the parametric equations describing the outline of your hand. Earlier you used the function `dot_to_dot` to interpolate the points given by the vectors `x` and `y` to give you a rough approximation of the outline of your hand which in reality is of course much smoother.

You can interpolate both functions on a finer grid and plot the result with so-called *piecewise polynomial* or *spline* interpolants by using something like

```
n = length(x);
s = (1:n)';
t = (1:.05:n)';
u = splinetx(s,x,t);
v = splinetx(s,y,t);
clf reset
plot(x,y,'.',u,v,'-');
```

Do the same thing with `pchiptx` instead of `splinetx`. Which do you prefer?

4. Discuss — at least intuitively (all details of the formulas are not required now, but will be in MATH 350) — what the functions `splinetx` and `pchiptx` presented in Chapter 3 of *Numerical Computing with MATLAB* do. What kind of spline does the `dot_to_dot` function provide?
5. How large is your hand? The `dot_to_dot` function used above creates a polygonal outline of your hand. Denote the  $n$  vertices of this polygon by  $(x_i, y_i)$ ,  $i = 1, \dots, n$ . These vertices should be given by the rows of the matrices `hand` or `H` you worked with earlier.

A classic, but little known, fact is that the area of this polygon is

$$(x_1y_2 - x_2y_1 + x_2y_3 - x_3y_2 + \dots + x_ny_1 - x_1y_n)/2.$$

This formula was supposedly first mentioned by Meister in 1769<sup>1</sup> and later also by Gauss (opportunity to add a little historical dimension to your presentation).

- (a) Prove this formula is true. *Hint*: How would you compute the area inside a polygon using elementary methods?
- (b) If `x` and `y` are column vectors, the area formula above can be evaluated with the MATLAB one-liner

$$(\mathbf{x}' * \mathbf{y}([2:n \ 1]) - \mathbf{x}([2:n \ 1])' * \mathbf{y})/2$$

---

<sup>1</sup>A. L. F. Meister, *Generalia de genesi figurarum planarum et inde pendentibus earum affectionibus*, *Novi Comm. Soc. Reg. Scient. Gotting.* 1 (1769/1770), 144-180 + plates.

- (c) What happens if you trace the outline of your hand in reverse order? Why?
  - (d) Try this method on a few other polygons you create analogously to the outline of your hand and compare their areas.
6. Make `wiggler.m`, your own version of `wiggle.m`, with two sliders that control the speed and amplitude. In the initialization, replace the statements

```
thetamax = 0.1;  
delta = .025;
```

with

```
thetamax = uicontrol('style','slider','max',1.0, ...  
'units','normalized','position',[.25 .01 .25 .04]);  
delta = uicontrol('style','slider','max',.05, ...  
'units','normalized','position',[.60 .01 .25 .04]);
```

The quantities `thetamax` and `delta` are now the handles to the two sliders. In the body of the loop, replace `thetamax` by

```
get(thetamax,'value');
```

and replace `delta` by

```
get(delta,'value');
```

7. You are now ready to end the project (and your final project presentation) by running the command `wiggler(hand)` (or some other polygon you used above).

Feel free to use your creativity to add a special touch to the graphics (or the effects) you're studying in this project (painted fingernails anyone?).