MECH 222 Computer Lab 5—Freighter Parameter Estimation

Learning Objectives

- Matlab: Load and manipulate real data; work with global variables; learn about subplots.
- Mathematics: Find approximate solutions for inconsistent linear systems by least squares.
- Engineering: Experience thrill of doing practical work with Matlab.

PRE-LAB ASSIGNMENT: *You need a computer for this, but you don't need Matlab.*

Some of your prelab results will be needed during the lab session. Make sure you keep a copy for yourself, in addition to producing a copy to hand in when the lab starts.

PL1: Make a rough estimate of the boat's steady-state velocity in each of the six configurations for which you have a physical measurement. Just pick a couple of well-separated times from the latter half of the timing experiment and calculate distance travelled over elapsed time. Two-digit accuracy is good enough.

PL2: To save time in the lab, we are going to analyze just one group of three trials, either the light-cargo trio or the heavy-cargo group. Decide now which group you will process. If possible, choose the group in which the three steady-state velocities calculated in PL1 are the most spread out. Write down which group you will use.

PL3: Trim the three LabVIEW data files you collected in Physical Lab 3 for the group of runs you selected in PL2. Make copies in which the text headers have been discarded, so that each file contains about 100–200 lines of measured numbers. (The text string "NaN" is also allowed: it's "Not a Number", but Matlab can cope.) Delete lines from the beginning and/or the end, as needed, so that the lines you keep correspond precisely to the towing experiment.

You should end up with three files, with meaningful names that identify the experimental setup in each run. Make sure you can access these files somehow from the PACE Lab. (Options: Carry them in on a USB storage device or a laptop, email them to yourself, or post them in some network-accessible location.)

PL4: Make three setup files, one for each experimental run. Get a prototype named `setup0.m` from Vista, and save a fresh copy (with a unique name) to record the physical parameters of your boat and the experimental conditions for each towing run. You will need to know

- $m = M_{\text{drive}}$, the total driving mass,
- $M = M_{\text{boat}} + M_{\text{cargo}}$, the mass of your loaded boat,
- A, the projected frontal area of your boat under water,
- L, the submerged length of your boat, and
- S, the wetted surface area of your boat.

The values of S, L, A, M will be the same in all three light-cargo runs, but m will change. [Loading the boat with more cargo would produce new and larger values of S, L, A, M that apply throughout all three heavy-cargo runs.]

PL5: Reynolds' number is the dimensionless quantity $\text{Re}(v) = L|v|/\nu$ involved in the calculation of skin-friction drag. Use the smallest and largest speeds found in PL1 to calculate an interval of Re-values relevant for the current fluid situation.

ACTIVITY 1: *Load, shift, scale, and save your towing data.*

Repeat steps 1.1–1.4 below three times, once for each of your selected experimental runs. You may find it helpful to write a script to automate the repetitive work required.

1.1 Import the data. If you have a text file named "`labview.txt`" that has been trimmed as outlined in Prelab P1, the single command

```
M = load('labview.txt');
```

will create a matrix `M` in which each line of the file generates one row of the matrix. Notice that the input parameter to `load` is a character-string containing the name of the file: the single-quotes above are essential.

1.2 Extract the measurements. To extract column 2 and column 3 from some given matrix `M` and turn them into independent column vectors, use Matlab's "`:`" selector like this:

```
col2 = M(:,2);    col3 = M(:,3);
```

Decide which column of the data matrix created in step 1.1 contains the experimental times and which column contains wheel-revolution counts. Extract these from the data matrix.

Discard from these column vectors any measurements recorded before the boat was launched, or after it reached the destination. Preserve the correspondence between times and positions by discarding the same number of components from the same positions in both vectors.

Subtract a constant from each entry of the time-vector so that it starts with $t = 0$, and multiply by a constant to get a vector of times in seconds (not milliseconds). Call that `Tlab`.

Apply some elementary operations to the count-vector to turn it into a column vector of increasing x-values, measured in meters along the tank (not degrees), with $x = 0$ when $t = 0$. Call the result `Xlab`. Make sure that after your transformations, the displacement value stored in component `Xlab(i)` still corresponds to the elapsed time stored in component `Tlab(i)`.

1.3 Test your work by plotting displacement (metres) versus time (seconds). This is a one-liner:

```
plot(Tlab,Xlab);
```

Don't hand in this plot: it's just a quick visual check to verify that steps 1.1–1.3 have succeeded. Recall that the tow tank is about 6 metres long, and try to remember about how many seconds your boat required to complete its journey.

1.4 Save your results for future reference. Choose a file name (ending in "`.mat`") that makes it clear which experiment you are dealing with, then say something like

```
save heavy100g.mat Tlab Xlab
```

Your chosen filename is the first input to Matlab's `save` command; it is followed by a *space-separated list* of variables to be saved. In a future Matlab session, the command `load heavy100g.mat` will read the column vectors `Tlab` and `Xlab` back into the current workspace.

Hand-in Checklist:

☐ This activity produces nothing to hand in.

ACTIVITY 2: *Graph position, velocity, and steady speed for each towing experiment.*

The goal here is to produce a two-pane plot like Figure 1. On top is the position versus time, showing one dot for each measurement (not joined by a line). On the bottom is velocity versus

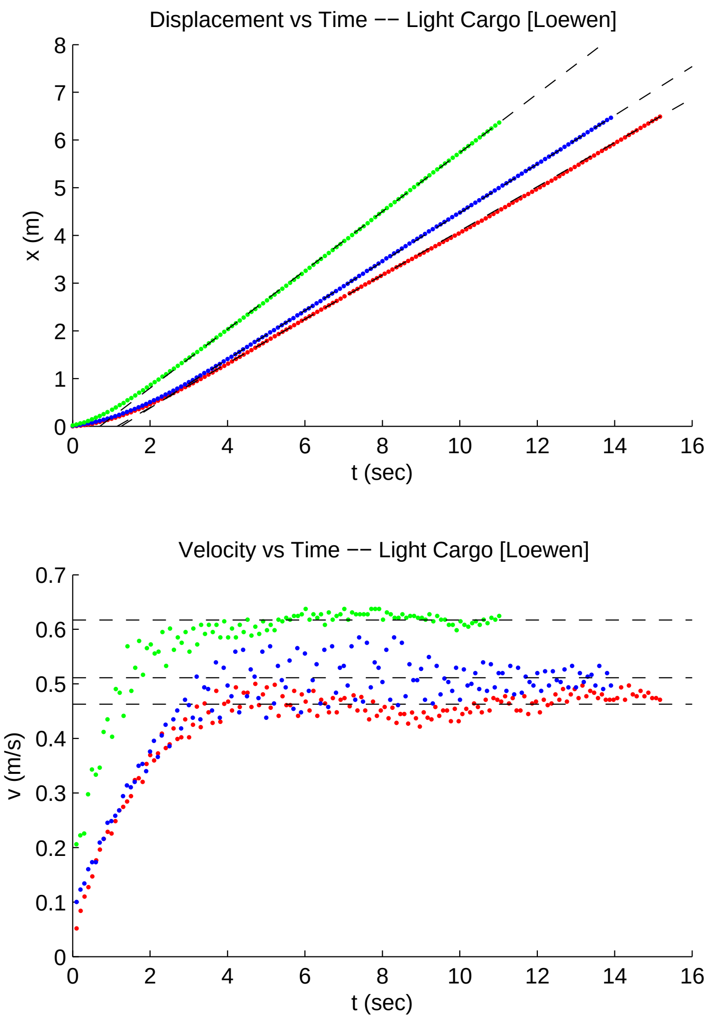

Figure 1: Sample position and velocity plot

time, calculated as outlined below. Dashed lines in contrasting colours show the steady-state speed for each run. *Important*: Both plots use the same scale for the horizontal (time) axis.

2.1 For each of your selected experiments, add a sequence of dots showing the position-vs-time measurements to the top pane in your figure.

2.2 For each of the three data series in 2.1, estimate the steady speed v_s. Don't get fancy, just do the same kind of calculation as in Prelab PL1 based on a pair of (t, x) points for each data series. Use Matlab to overlay the data series of 2.1 with a dashed line through your chosen points and beyond to show the steady-speed motion.

2.3 For each data series, calculate the average speed between adjacent position measurements. Suppose `N = length(Tlab)`. Define `Nmid = N-1`, and make two new vectors with `Nmid` components:

```
Tmid(k) = (Tlab(k+1)+Tlab(k))/2;   % Midpoint of cell k
Vmid(k) = (Xlab(k+1)-Xlab(k))/(Tlab(k+1)-Tlab(k));
```

The first contains times; the second estimates the corresponding boat speeds. Plot boat speed versus time in the second subplot pane. Use dots; don't join them. Overlay each data series for velocity with a dashed horizontal line showing the steady speed calculated in 2.2.

2.4 Every time you calculate the speeds for a given data series, prepare a new archive file. (Give it a new descriptive name.) Assuming the vectors `Tlab` and `Xlab` are still intact, re-save those two and add the new ones created above by entering a command like

```
save h100txv.mat Tlab Xlab Tmid Vmid
```

(Adapt the filename every time, according to some clear system.)

Hand-in Checklist:

- ☐ A two-pane plot showing three observations of position-vs-time in the top pane and velocity-vs-time in the bottom pane. Your name should appear in the plot title, and the axes should be labelled by Matlab.
- ☐ Numerical values for the steady speed v_s in each of your three chosen experiments, with 3 significant digits for each.

ACTIVITY 3: *Choose reasonable values for α and β.*

Refer to Idea 2, "Extracting parameters from terminal velocity," and Idea 3, "Overdetermined Linear Systems," in the Briefing Notes.

3.1 For each of your three selected experiments, find numerical values for

$$c = v_s^p, \qquad b = \frac{mg}{\rho S v_s^2}$$

and have Matlab draw the first-quadrant segment of the line $x + cy = b$. You should have a plot showing three lines. Don't print it yet.

3.2 [*Re-read Idea 5 before you do anything here.*] The theory says that the three lines in 3.1 should all pass through the same point, (α, β). Experimental errors make this unlikely. Using Idea 3 in the Briefing Notes, make a reasonable choice of the point (x^*, y^*) that provides the best approximate solution. Have Matlab add it to the plot from step 3.1 and print the result.

3.3 Using the values of α and β just found, evaluate the function

$$C(v) = \alpha + \beta v^p$$

at each of the three steady speeds in your experiment. Compare these values with the numbers you got for $C_{D,\text{avg}}$ in your Physical Lab writeup.

Hand-in Checklist:

- ☐ A well-labelled plot showing three lines in the first quadrant and a point meant to serve as their approximate intersection point.
- ☐ Numerical values for α and β calculated for your chosen cargo scenario and shown on the plot just mentioned. A brief discussion about how aggressively you had to fudge things to generate a point you could use in later activities.
- ☐ A session transcript showing the command you used to get these results.
- ☐ A brief discussion in response to item 3.3.

ACTIVITY 4: *Simulate your freighter-towing experiment.*

Now that we know α and β, we can use `ode45` to investigate how well the acceleration formula from line $(*)$ in the briefing notes predicts the results of experiment. To save time, we will simulate just one of your three towing experiments. You get to choose which one to simulate.

4.1 Review the requirements of `ode45`, Matlab's built-in differential-equation solver, and write a wrapper function for the given function `accel` (call your wrapper `XdotVdot`) so that the following command will find the boat's position and velocity at every time in the interval $[0, 40]$:

```
x0 =  0.0;
v0 =  0.0;
Tf = 40.0;
[Tlist,xv] = ode45(@XdotVdot,[0,Tf],[x0;v0]);
```

Function `XdotVdot` should be very short, and your name should appear in the comments.

4.2 Choose the physical experiment you want to simulate. Run the corresponding `setup` script that you built in PL2 to store the relevant parameters into the global variable zone. (Make sure that the script includes the updated values of S and C_D that you worked out in Activity 2.) Then run the `ode45` command suggested above to generate a predicted trajectory.

Plot your results, using a two-pane format similar to the one in Task 2 but plotting continuous curves. Align the time-axes as required earlier; shrink them as needed to keep only the part of the motion that would fit into a tow tank of length 6 m.

```
≫ subplot(2,1,1); plot(Tlist,xv(:,1)); title('Position vs Time');
≫ subplot(2,1,2); plot(Tlist,xv(:,2)); title('Velocity vs Time');
```

4.3 Recall the measured boat motion and calculated boat speed from Task 2. Overlay dot-plots of these experimental results on the theoretical predictions graphed in 4.2 above. Put your names in the title and print for submission.

4.4 Come up with a single number that summarizes the mismatch between theory and experiment, as follows. Solving the differential equation with `ode45` gives you a long list of predicted velocity values (let's call these numbers $w_1, w_2, \ldots, w_N$) at each instant where you already have a measured velocity (let's call these $v_1, v_2, \ldots, v_N$). Find the average squared discrepancy,

$$\frac{1}{N}\sum_{j=1}^{N} |v_j - w_j|^2.$$

If the theory matched the experiment exactly, this average would be 0; in general, small values indicate a better agreement than large values.

Hand-in Checklist:

- ☐ One two-pane plot. The top pane shows position versus time for two data sets. One is a continuous curve calculated by `ode45`, the other is a dot-plot of experimental observations. The bottom pane shows velocity versus time for the same two data sets.
- ☐ A numerical value for the average squared error in item 4.4, supported with brief comments comparing the theoretical prediction with the lab observations.

ACTIVITY 5: *Explore Other Exponents.*

Repeat activities 2–4 with a different exponent p. Build a little table relating the choice of p to the average squared error, and identify which p makes the theory fit the data best according to this criterion.

ACTIVITY 6 [bonus]: *Analyze the other three runs.*

If you have time, repeat Activities 1–5 for the towing scenario that you didn't choose the first time around. Compare the results. Does it seem likely that the constant C_D in the definition of form drag is really the same for both low-cargo and high-cargo scenarios?