Advanced Statistical Inference
Loss minimization

1 Aims

- To perform a least squares fit to the Olympic 100m data.

2 Olympic data

Download the olympic matlab file from the ASI collaborative space (you'll need to save it in your workspace). Create a new script and add the following two lines

```
% Least squares Olympic script
clear all; close all; % Clear all variables and close all plots
load <olympicdata>
```

where `<olympicdata>` should be replaced with whatever name you gave the downloaded file.

Run the script and then type the command `whos` in the command window. This lists all of the variables in the current workspace. You should see one (amongst others) called `male100` which holds the Olympic men's 100m data. Type `male100` in the command window so see the contents of this variable. It is a matrix (a 2-dimensional array). The first column holds the Olympic year and the second the winning time (in seconds). Matlab allows us to easily extract subsets of matrices by using the colon notation. For example, the command

```
x = male100(:,1)
```

creates a variable called `x` and copies into it the first column (a colon on its own just means 'all') of `male100`. Add the following two lines to your script:

```
x = male100(:,1);
t = male100(:,2);
```

We'll now plot the data. Back in the command window, run the script and then try the following two commands:

```
plot(x,t);
plot(x,t,'b.');
```

The (optional) third argument specifies the plot format. Possible formats are listed in `help plot` – experiment with them until you find something that looks good. Also experiment with adding the commands `hold on` and `hold off` between two plot commands. Add a suitable `plot` command to your script.

We'll now fit the model $t = w_0 + w_1 x$ to our data. This model fitting step is something we might wist to do more than once, so it makes sense to create a function. Create a new .m file and add the following line

```
function [w0,w1] = lsfit(x,t)
```


Save it as `lsfit.m`. Back in the original script add the following line to call the function:

```
[w0,w1] = lsfit(x,t);
```

We now need to write the least squares function. Recall from the lectures that

$$w_1 = \frac{\overline{tx} - \bar{t}\bar{x}}{\overline{x^2} - \bar{x}^2} \quad (1)$$

$$w_0 = \bar{t} - w_1 \bar{x} \quad (2)$$

where, $\bar{x} = \frac{1}{N}\sum_{n=1}^{N} x_n$, $\overline{x^2} = \frac{1}{N}\sum_{n=1}^{N} x_n^2$ etc. In the `lsfit` function, you could add the following code to compute $\bar{x}$:

```
xmu = 0;
N = length(x);
for i = 1:N
xmu = xmu + x(i);
end
xmu = xmu/N;
```

But....Matlab has an inbuilt `mean` command so we can instead write:

```
xmu = mean(x);
```

This is not only neater but faster. Add something similar to produce a variable for $\bar{t}$.
The two remaining averages, $\overline{x^2}$ and $\overline{xt}$ are a little harder. For $\overline{xt}$ we could use the following code:

```
xtmu = 0;
N = length(X);
for i = 1:N
xtmu = xtmu + x(i)*t(i);
end
xtmu = xtmu/N;
```

Alternatively, we can use element-wise multiplication. In the command window type:

```
xt = x.*t;
```

and examine the result. The '.' tells Matlab to perform the next operation element by element. So, the result is another vector `xt` which has elements `x(1)t(1), x(2)t(2), ....` Using this, the following code will find $\overline{xt}$:

```
xt = x.*t;
xtmu = mean(xt);
```

Add this and something similar for $\overline{x^2}$ to your function.

The final step in the function is using these quantities to compute w_0 and w_1. The following code will compute w_1:

```
w1 = (xtmu - xmu*tmu)/(x2mu-xmu^2);
```

where

```
a^b
```

raises `a` to the power `b`. Add a similar line for w_0, save the function and run the script. Inspect the values help in the variables `w0` and `w1` – do they agree with the values provided in the notes?

Adding the following lines to the script:

```
hold on;
plot(x,w_0 + w_1*x,'r-');
```

and run it again. The plotted line should run through the data as shown in the lectures.

Extend your script in the following ways:

- Add comments so that others (and you) can see what you've done.
- Put labels on the plot axes – `xlabel` and `ylabel` will be useful.
- Compute the average loss – $\mathcal{L} = \frac{1}{N}\sum_{n=1}^{N}(t_n - w_0 - w_1 x_n)^2$. Do this in the `lsfit` function and return is as an additional variable.
- Fit a function to the women's 100m instead. This data is inside the same matlab file – use whos to find the variable name.
- Fit a function to the men's and women's 100m and find the year that the two races are predicted to have the same winning time.