

matlab checklist

a little scavenger hunt

January 21, 2004

1 .m files

Very easy. Open up a new document in matlab (as if it were word or sth.) and save it as YOURNAME.m Done. To run whatever you will write in this file on matlab you can either use the "F5" button or type YOURNAME into the main matlab terminal. If you choose the latter, make sure you save your .m file into the right location (usually matlab will do that for you). Otherwise matlab won't be able to find it. You can also save it where ever it is convenient for you, but in that case you have to define the pathway for matlab in the "file menu" I think. (Pressing "F5" while you are in your .m file is really the easiest solution.)

2 numbers and vectors

Please write everything in your .m file. Makes things much easier to repeat. Before everything else, you should always write a

```
clear all;
```

command at the beginning of EVERY file. It will prevent matlab from confusing old variables from old programs and the variables you feed into it. This may cause you great sorrow. Mark my words :) Another thing: matlab does not care about whether you use one or more spaces or tabs between two variables.

2.1 a single number

Make a variable

```
a = 5;
```

Done. Good. If you wanna check if you were successful, run YOURNAME.m, type "a" into your matlab terminal and press "return" matlab should spit out something like

```
a
=5.
```

2.2 a vector (nothing but a bunch of numbers)

Ok. Still easy. We are going to make a vector b like dat:

```
b = [10 20];
```

Done. Run YOURNAME.m and check the output. Now add a third number to that vector by writing

```
b= [b 30];
```

What's the difference between [b 30] and [30 b]? Now create a second vector by writing

```
c=ones(1,3); (matlab always asks for rows, then columns)
```

then multiply:

```
c= c*10; ( you could also just type c=ones(1,3)*10; it has the same effect.)
```

Now multiply a, b and c. What do you have to pay attention to? Remember class? (.* vs * When do you have to use the . ?) The output in matlab should look like this:

```
ans
    500 1000 1500
```

Make the output into a vector d. Check the dimensions of your vector with the comands `length(d)` or `size(d)` Comment out everything you have written so far. Do so by adding a % sign to the beginning of every line (not in front of the "clear all" command!!) Check what happens when you run your file now? Good trick? This is generally helpful to structure your code and to leave notes. Matlab simply ignores everything behind a "%" sign.

3 FOR loops (and again and again and again)

If you want to create vectors like [1 2 3 4 5] or we want to perform more complex calculations on single elements of a vector it is sometimes necessary to use a "FOR loop". They are pretty standard and easy once you get the concept. Create a variable time:

```
time=20;
```

Now create a new vector with length time:

```
timeline=zeros(1,time);
```

Now to fill this vector, we will use a for loop. Write

```
for i = 2: time
    timeline (1, i) = timeline (1,i-1) +1;
end
```

Why does the for loop start at 2 and not at 1? What would be the problem if you started at 1?

First real calculation

Ok. Imagine you can calculate the velocity of an accelerating car at time t by a certain mathematical rule (like, say: $v(t) = \frac{t}{1+t}$). If its top velocity is known, we can plot its velocity over time very easily. Try this:

```
topspeed=150; speed = zeros(1,time);
for t = 1:time
    speed (1, t) = topspeed*(timeline(1,t)/(1+timeline(1,t)));
end
```

Now checking this will require some plotting action, which we will do a little later on, for now, just try

```
plot (speed)
```

Now imagine that the driver really only drives fast when he feels like it (maybe more than a 5 on a scale from 0 to 10), which is a pretty random variable. If he doesn't feel like it, he will just drive at a constant speed. We will take the

driver's mood into account by using an:

4 if loop

Pretty easy really. But first we have to create a random variable for the drivers mood:

```
Fahrvergnuegen=rand*10;
```

Now:

```
if (Fahrvergnuegen > 5)
    for t = 1:time
        speed (1, t) =topspeed*(timeline(1,t)/(1+timeline(1,t)));
    end
else
    speed (1,:) = 5;
end
plot (speed)
```

You learned six things in this:

1. You can create a random number between 0 and one with the `rand` command. `randn` also works. Find out the difference between them with the `help` function.
2. To create an if-loop, you write the statement between "the if-condition in parentheses behind an `if` " and an `end`
3. You can "nest" a `for`-loop into an `if`-loop. You can do the same with multiple `for`-loops or `if`-loops.
4. You can add a command to what is supposed to happen when the condition behind the `if` is not met with an `else` before you end the `if`-loop.
5. What does `speed (1, :)` do? I guess you can figure it out by looking at the plots.
6. Look at the way you wrote the commands. Notice the tabbing whenever I open a new `for` or `if`. It helps to keep your code tidy and makes your TA very happy when he has to go through your code. Try it out :)

5 plotting

This will be the last thing for today. You have already done some of it earlier. Please comment out all the earlier print commands at this time. You can do that by either putting a `%` sign in front of the line or by highlighting the line and then choosing "comment" in the "text" menu. There is also a shortcut which is `ctrl` + `/` and to uncomment again it is `ctrl` + `t`. Try these out a couple of times, they will come in rather handy in the future.

But back to plotting: To plot the speed of your car plot

```
figure(1)
plot(timeline, speed)
```

Try also:

```
figure(2)
plot(speed, timeline, 'r:' )
```

This brings up windows and plots `timeline` on the horizontal axis and `speed` on the vertical axis. What does the `"r"` and the `":"` do? Check out other options in `help plot` Why could it be important to plot "speed" against "timeline" and not as before, alone?

All figures are numbered in Matlab – you can use any integer you want, and if you just call 'figure' it will make up a number for you. If you want to print the window, you can do that by choosing print from the file menu. I recommend using "page setup" to make sure the window maps onto the paper you're using in some good way.

Use

```
hold on
```

in your .m file to plot multiple plots after each other into the same figure. The `hold on` command tells Matlab to hold what is currently on the plot if new data is plotted; otherwise, the default behavior is to replace what is there (you can go back to this by typing 'hold off'). To change the color of your plot, write

```
plot(timeline, speed, 'r')
```

To really get into all of the various options is complicated (type `help plot` if you want to see), but if you follow the pattern of the examples here you might not need to get into all of the details.

Now add some labels and practice changing the axis limits. They always need to follow the `plot` command.

```
title('Speed of ma caa')
xlabel('time [seconds]')
ylabel('velocity');
legend('second', 'first');
```

Also try out this command:

```
axis([0 time 0 topspeed])
```

What does it do?

That will be it for now. Let me know if you find any errors on this sheet, or if there is anything you think I should change.

6 If you are too fast and bored already...

There are a couple more things you can do that will actually be helpful to you.

Get rid of the `if` loop. Now imagine that our race car accelerates differently depending on the level of octane in the gas. There are 5 different types of gas available. They change the function of acceleration a bit as it now follows:

$v(t) = \text{octanelevel} * \frac{t}{1+t+\frac{1}{\text{octanelevel}}}$. Your exercise consists of the following:

1. Create a vector `octane` with 5 entries, 0.1, 0.3, 0.5, 0.7, and 1.
2. Change the dimensions of your speed vector. It must now be a matrix. (`speed = zeros(5,time)`);
3. Change your calculation line in the for loop and nest it into a second for loop.
4. Plot acceleration curves for all of them, using the `surf` command or also `imagesc`.

If you get frustrated and need to be reminded of why you are doing this, just plug in the command `why` !

Good luck.