Programming Habits

Instructor: Mark Kramer

Good programming habits

In this course, we'll write programs:

```
Vstar = Iinput*R + Vrest;      %Define Vstar = target voltage.
tau = R*C;                     %Define tau = time constant

for k=1:400-1                                  %For each time step,
    V(k+1) = V(k) + dt*(-(V(k)-Vstar)/tau);    %...evaluate V at the next time step
    if V(k+1) > Vth                            %When we cross threshold
        V(k+1) = Vreset;                       %...set V to reset
        n_spikes = n_spikes+1;                 %...and add 1 to spike counter.
    end
end

taxis = (1:length(V))*dt;          %Return the time axis too.
```

"code" "comments"

Good programming habits

Q. What makes a program “good” ?

A.

- It works
- It’s interpretable
- It’s extendible

Write programs for people, not computers

You should think of (at least) 3 other people programing:

Person 1: Yourself, tomorrow.

Person 2: Yourself, in six months.

Person 3: Another user of your code.

Always code as if the [person] who ends up maintaining your code will be a violent psychopath who knows where you live.

— John Woods comp.lang.c++

Write programs for people, not computers

How? **Comment your code, thoroughly.**

– carefully commenting your code will make you happier.

- your future self
- popular and appreciated in your research group.

```
dt=0.1;                                 %Set the timestep.
V(1)=Vrest;                             %Set the initial condition.

n_spikes = 0;                           %The # of spikes detected.

Vstar = Iinput*R + Vrest;               %Define Vstar = target voltage.
tau = R*C;                              %Define tau = time constant
```

Write programs for people, not computers

How? **Make comments meaningful.**

Ex.

```
i = 10          % i equals 10
```

NO!

```
plot(x,y)       % plot x versus y
```

NO!

Ex.

```
i = 10          % Define the number of iterations

plot(x,y)       % Plot impulse (x) vs rate (y)
```

Write programs for people, not computers

How? **Use comments to break programs into logical blocks**

Ex.

```
4+3
```

NO!

```
4/10^2
```

NO!

Ex.

```
%% Example 1: addition.
4+3

%% Example 2: division and powers.
4/10^2
```

Write programs for people, not computers

How? **Make names consistent, distinctive, and meaningful.**

Ex.

```
a = 10
b = …
c = my_function_18v2(a,b)
```

NO!

Ex.

```
n_trials = 10
lfp_rat  = …
lfp_mean = compute_trial_mean(n_trials,lfp_rat)
```

Write programs for people, not computers

How? **Utilize spacing to make code stylish.**

Ex. **NO!**

```
Iinput=3;
R=10;
C=0.6;
Vrest=-70;
Vthreshold=-60;
Vreset=-80;
```

Ex.

```
Iinput      =    3;
R           =   10;
C           =  0.6;
Vrest       =  -70;
Vthreshold= -60;
Vreset      =  -80;
```

Write programs for people, not computers

How? **Utilize spacing to make code stylish.**

Ex.

```
for k=1:400-1
V(k+1) = V(k) + dt*(-(V(k)-Vstar)/tau);
if V(k+1) > Vth
V(k+1) = Vreset;
n_spikes = n_spikes+1;
end
end
```

NO!

Ex.

```
for k=1:400-1
    V(k+1) = V(k) + dt*(-(V(k)-Vstar)/tau);
    if V(k+1) > Vth
        V(k+1) = Vreset;
        n_spikes = n_spikes+1;
    end
end
```

Write programs for people, not computers

How? **Break up your program into manageable parts.**

Ex. Books organized into chapters, paragraphs.

- Etymology
- Extracts

1. Loomings
2. The Carpet-Bag
3. The Spouter-Inn
4. The Counterpane
5. Breakfast
6. The Street
7. The Chapel
8. The Pulpit

CHAPTER 1

LOOMINGS

Call me Ishmael. Some years ago — never mind how long precisely — having little or no money in my purse, and nothing particular to interest me on shore, I thought I would sail about a little and see the watery part of the world. It is a way I have of driving off the spleen and regulating the circulation. Whenever I find myself growing grim about the mouth; whenever it is a damp, drizzly November in my soul; whenever I find myself involuntarily pausing before coffin warehouses, and bringing up the rear of every funeral I meet; and especially whenever my hypos get such an upper hand of me, that it requires a strong moral principle to prevent me from deliberately stepping into the street, and methodically knocking people's hats off — then, I account it high time to get to sea as soon as I can. This is my substitute for pistol and ball. With a philosophical flourish Cato throws himself upon his sword; I quietly take to the ship. There is nothing surprising in this. If they but knew it, almost all men in their degree, some time or other, cherish very nearly the same feelings towards the ocean with me.

There now is your insular city of the Manhattoes, belted round by wharves as Indian isles by coral reefs — commerce surrounds it with her surf. Right and left, the streets take you waterward. Its extreme downtown is the battery, where that noble mole is washed by waves, and cooled by breezes, which a few hours previous were out of sight of land. Look at the crowds of water-gazers there.

Write programs for people, not computers

How? **Break up your program into manageable parts.**

Ex. Organize code into "pieces" - e.g. each piece a function.

```
d   = preprocessing(data)
pow = power_spectrum(d)
coh = coherence(d)
```

```
function output = preprocessing(d1)
        function P = power_spectrum(d1)
                function C = coherence(d1)
```

Write programs for people, not computers

How? **Break up your program into manageable parts.**

Ex. Organize code into "pieces" - e.g. each piece a function.

```
function output = preprocessing(data)
```

Maximum length of a function:

- ideally no more than one screen's worth of code.

Don’t repeat yourself (or others).

How? **Make the computer repeat tasks**

Ex. Cut & paste over and over again.

Challenge:

Plot `sin(x+k*pi/4)` where k varies from 1 to 5 in steps of 1.

Python …

Don’t repeat yourself (or others).

How? **Make the computer repeat tasks**

Ex. Cut & paste over and over again.

```
a = [[ some voltage trace ]];          … some data
t = (0:1:length(a));                   … define axis for plot
y = a.^2;                              … do something to data
plot(t,y)                              … plot it
title(['Plot of voltage trace A'])     … with a title.


b = [[ another voltage trace ]];       … more data
t = (0:1:length(b));                   … define axis for plot
y = b.^2;                              … do something to data
plot(t,y)                              … plot it
title(['Plot of voltage trace B'])     … with a title.
```

Don’t repeat yourself (or others).

How? **Make the computer repeat tasks**

Ex. Cut & paste over and over again. Define a *function*

```
function square_and_plot(x, plot_title)
  t = (0:1:length(x));
  y = x.^2;
  plot(t,y)
  title(plot_title)
end
```

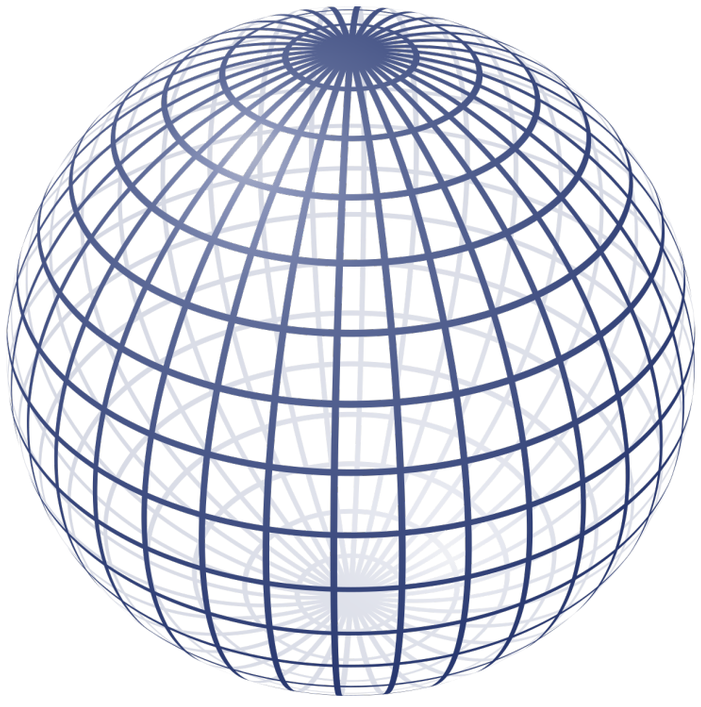
```
a = [[ some voltage trace ]];
b = [[ another voltage trace ]];
square_and_plot(a, ‘Plot of voltage trace A’)
square_and_plot(b, ‘Plot of voltage trace B’)
```

Don’t repeat yourself (or others).

How? **Re-use existing functions (that your trust)**

It’s usually better to find an established library or package that solves a problem than to attempt to write you own routines for well established problems.

Ex. Compute the distance between two points on a sphere.



MATLAB:

```
distance(lat1,lon1,lat2,lon2)
```

Visualize data

How? **When in doubt, plot it out.**

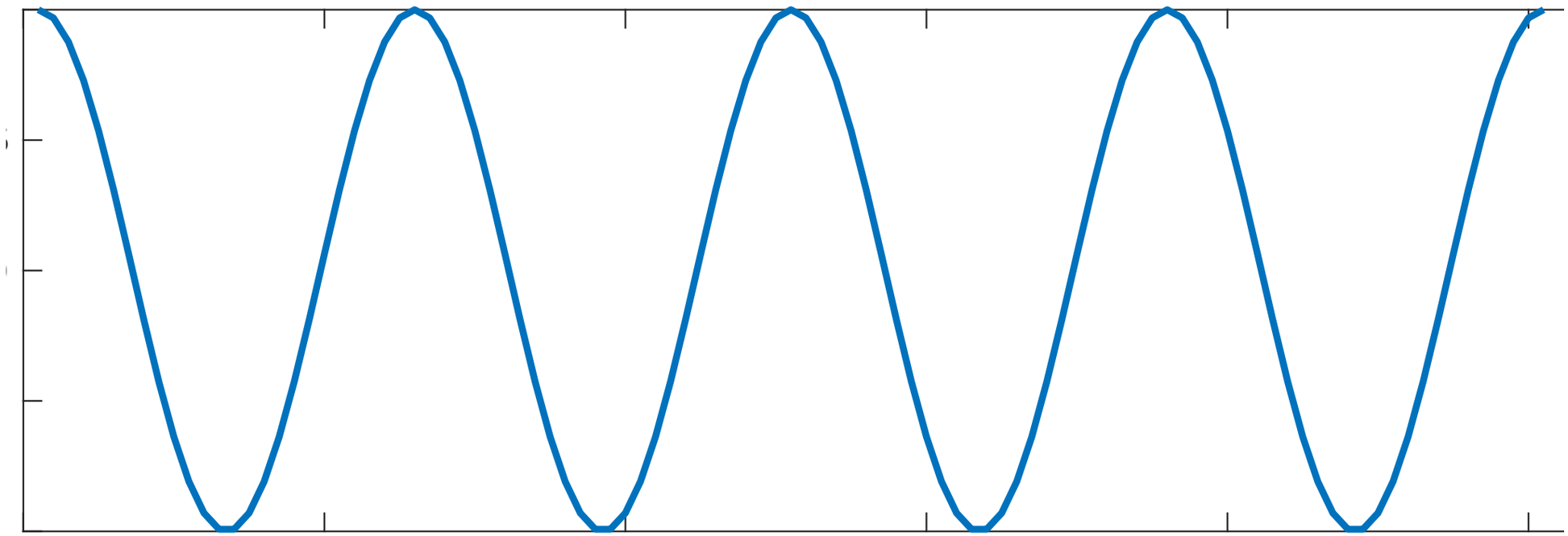
Ex. You're give the data in variable **x**. Let's look at it.

```
x =
  Columns 1 through 6
    1.0000    0.9686    0.8763    0.7290    0.5358
  Columns 7 through 12
    0.0628   -0.1874   -0.4258   -0.6374   -0.8090
Columns 13 through 18
 -0.9921   -0.9921   -0.9298   -0.8090   -0.6374
Columns 19 through 24
 -0.1874    0.0628    0.3090    0.5358    0.7290
Columns 25 through 30
  0.9686    1.0000    0.9686    0.8763    0.7290
Columns 31 through 36
  0.3090    0.0628   -0.1874   -0.4258   -0.6374
Columns 37 through 42
 -0.9298   -0.9921   -0.9921   -0.9298   -0.8090
```

Visualize data

How? **When in doubt, plot it out.**

Ex. You're give the data in variable **x**. Let's plot it.



Visualizing the data: the structure becomes clear.

Do something

Do not let these rules stop you from coding.

- The most important is to write code.
- Think back to these "rules" from time to time …

Let's do it …