

# AMATH 410

## Review:

### Structuring a program in MATLAB

Our aim is to illustrate a basic strategy for how to write a new MATLAB program from scratch. A .m file with the code below is available as `cell_reproduction_review_code_1.m` from our website.

1. define given parameters
2. make a dummy list (or matrix) of values for the variable that are going to evolve over time
3. set the initial value of that dummy list
4. run a for loop to fill in all values of that variable (solution) list
5. process the result, if required
6. plot

Let's say we're given this problem – similar to the cell reproduction model discussed on the first day of class.

Implement the rule

$$n(t) = 4pn(t-1)(1 - n(t-1)/K)$$

starting from an initial population size of  $n(1) = 1000$ , and simulating for 100 timesteps (or generations). Then, compute a list of all times  $t$  where  $n(t) > 1300$ . Use  $p = 0.75$ ,  $K = 2000$ .

#### Step 1

Where do I start? In my first lines of code, I define all of the parameters.

```
p=.75; %fraction of cells that survive each reproduction
generation_end=100; %predict up to this generation in the future
K=2000; %reproduction capacity
```

#### Step 2

OK, now I have to think about what it is that I am going to **evolve over time**. I am going to predefine a “dummy” list or matrix of numbers representing the variable that I am going to simulate. That dummy list is often just filled in with zeros. Then I will go through and fill it in later.

Here, I want a list of values of  $n$  at timesteps  $t = 1, \dots, 100$ . So, let's just define a list of 100 zeros to use as my dummy list. There are a couple of ways to do this:

```
nlist=zeros(1,generation_end);
```

or

```
nlist=0*(1:generation_end) ;
```

### Step 3

Alright. Now I need to set my initial conditions. What's the first value of nlist? That's

```
nlist(1)=1000;
```

### Step 4

Now I need to loop over time. At each timestep, I implement my dynamical rule.

```
for t=2:generation_end
    nlist(t)=4*nlist(t-1)*p*(1-nlist(t-1)/K) ;
end
```

So, I went through and filled in the dummy zero values of nlist with their real values.

### Step 5

Now, I need to make a new list – of generations  $t$  when  $nlist(t) > 30$ . Let's call this new list `list_of_times_over_limit`. Note that I DON'T KNOW IN ADVANCE HOW LONG THIS LIST IS SUPPOSED TO BE. In this way, the situation is similar to **when we are building up a list of dwell times for the markov chain**. We will therefore not make a dummy list in advance, but proceed as follows (all three ways give same answer).

#### Way 1 – keep growing the list via a loop

Here, we have to define an integer-valued number `which_entry_to_update.m` that tells me which entry in the list I'm supposed to be changing. We start with the first entry.

```
which_entry_to_update=1
```

OK, let's use this:

```
for t=1:generation_end
if nlist(t)>1300
list_of_times_over_limit(which_entry_to_update)=t ;
%next I say: update the NEXT entry down the list next time
which_entry_to_update=which_entry_to_update+1;
end
end
```

#### Way 2 – keep growing the list via a loop, version 2

Here, we start with an empty list, and keep appending to it.

```
list_of_times_over_limit=[]
```

```
for t=1:generation_end
if nlist(t)>1300
list_of_times_over_limit=[list_of_times_over_limit t];
end
end
```

### Way 3 – find command

Here I use the find command in matlab.

```
list_of_times_over_limit=find(nlist>1300)
```

That's all we have to do! Here's a review on using the find command:

```
find( mylist==3 ) %returns a list of indices for entries that are equal to 3
      find( mylist<10 ) %returns a list of indices for entries that are less than 10
      find( mylist~=5 ) %returns a list of indices for entries that are not equal to 5
```

The point is: you type

find( CONDITION ) and get out all the entries where CONDITION is true.

For example,

```
find( [1 2 2] == 2 ) returns [2 3]
```

## Step 6

OK, let's plot  $n$  vs  $t$ . I'll make a list of times  $t$  first:

```
tlist=1:generation:end
```

figure

```
plot(tlist,nlist,'.','MarkerSize',24)
```

```
ylabel('n(t)','FontSize',20)
```

```
xlabel('t','FontSize',20)
```

```
set(gca,'FontSize',20)
```