# Lab 6

*Out: Sunday, Mar 16th*

For this lab you will be working in pairs (two people)!

This is the last lab for which you will use MATLAB. It should serve as a review of some of the concepts you have learned up to this point, while providing you the chance to tackle a more challenging (and hopefully interesting) problem.

The emphasis of this lab is on thinking about a "plan of attack" when working on a problem. Finding a good approach to solving a problem is often half the battle, and even if you don't get to the solution the first time around, it'll be easier to adjust your thinking with a solid plan. Spend some time thinking about the WHOLE lab before beginning; brainstorm some ideas on paper with your partner. For a given problem, think about what you're asked to do, what information you need to do it, and how to logically organize your steps. Write out some pseudocode before you write any real Matlab code. Again, the emphasis is on your plan of attack; the majority of the credit for this lab will be awarded for a well-thought-out approach, even if you don't get the right answer or don't have enough time to finish everything.

### Problem 6.1

## Practice with arrays

- Given nonzero integers $n$ and $k$, build an array that looks like this:

```
1 n+1 ...
2 n+2 ...
3 n+3 ...
4 n+4 ...
...
n 2n ... nk
```

  For example, if n = 3 and k = 4, you'll get:

```
1 4 7 10
2 5 8 11
3 6 9 12
```

Hint: use `1:(n*k)` and `reshape`.

- Now take that array and "pad" it with zeroes, i.e., make an $(n+2) \times (k+2)$ array whose central $n \times k$ block is the matrix above. Continuing our example above, the matrix would look like this:

```
0 0 0 0 0  0
0 1 4 7 10 0
0 2 5 8 11 0
0 3 6 9 12 0
0 0 0 0 0  0
```

- Write a function `res = pad(m)` that does the following transformation in general: If `m` is the matrix

```
1  3  5
2  4  6
```

then `res` is the matrix

```
1 1 3 5 5
1 1 3 5 5
2 2 4 6 6
2 2 4 6 6
```

i.e., surrounds a matrix with a ring of numbers each of which is the same as the original array element that's closest to it.

Hint: When you add the ring of numbers start by adding the new rows. Then add the new columns.

## Diffusion

Suppose we have a drop of ink in a dish of water. Over time, the ink diffuses into the water. We can model this in matlab by representing the dish of ink with an array of "little bits of water". Each one has a "concentration of ink"

in it. So initially, the center element of the array has concentration 1, and all other elements have concentration zero. The way we update the picture is this:

We imagine that some particles in adjacent bits of fluid move into neighboring bits. So particles from `m(i,j)` move into `m(i+1,j)`, `m(i-1,j)`, `m(i,j+1)`, and `m(i,j-1)`. Let's call each "bit of fluid" a "cell." Well, just as particles are moving from cell $(i, j)$ to cell $(i + 1, j)$, other particles are moving from cell $(i + 1, j)$ to cell $(i, j)$. The result of this is that after a short time, the concentration of ink in cell $(i, j)$ changes with this rule:

$$c_{new}(i, j) = (1 - K) * c_{old}(i, j) + K * average(neighbors)$$

where $K$ is some constant depending on the fluid. (Things diffuse more slowly in honey than in in water, for instance). We'll use $K = .5$.

**Task:** Given a matrix $m$ containing concentrations of ink, write a procedure `res = diffusion(m)` that contains the new concentrations, after one diffusion step.

**Tricky issue**: What will you do at the boundary of $m$? For example, how should you update $m(1, 1)$? It doesn't have any neighbors to the left or above it! You should do the following: assume that the values in the neighbors to the left and above are the SAME as the current value of in $m(1, 1)$, and similarly for other edge cells.

**Hint:** The last part of the initial problems will be very helpful here!

**Task:** Write a program `diffusionDemo(n)` that does the following:

- Makes an $2n + 1 \times 2n + 1$ array filled with zeros.
- Puts a "1" in the center element of the array. (Which element is that???)
- 100 times, does the following:
  - Displays the array using `imagesc` or `surf`
  - Pauses for .2 seconds (you may adjust this if you want it faster or slower) (hint: see the doc for `pause`)
  - Updates the array using your `diffusion` procedure

Run your program with $n = 10$ and see what happens.

Try running you program using different display angles on the graph.

**Main Hints:**

- Start by writing out a plan of attack. (either in comments or on some scratch paper).
- Think about how you can update the cells in one line of code. (No need for a loop!!!) This only applies to each small increment of time when the concentrations change. You will still need to loop to make the ink diffuse over time. Please ask a TA if this does not makes sense.

**Good Luck and Have Fun! This is the last lab using MATLAB!**