# Week 11: More Vectorization and Generating Linear Data

## Problem 1: Vectorization

Here is a potential cipher you might use: given some lowercase plaintext that has been wrapped into a matrix of dimension n-by-m, we change each letter in the first column to the letter 1 before that, each letter in the second column to the letter 2 before that, …, each letter in the m-th column to the letter m before that, wrapping back around to Z if necessary. For example,

a c z a

->

c d b b

Please implement this encryption function.

Sample:

encrypt( [ 'matlab' ; 'isreal' ; 'lyfun!' ] ) = [ ‘lyqhvv’ ; ‘hqoavf’ ; ‘kwcqii’ ]

(the ‘!’ can map to any character you want)

### Problem 2: Linear Regression Cost Function

Given some 2D data points, calculate the cost function for a potential line of best fit. Remember that the cost function is defined as

The data points will be given as an m-by-2 matrix where each row is a data point. You will also be given two parameters Θ0 and Θ1 which represent the equation

y = Θ0 + Θ1x

and you will have to calculate the y values that the hypothesis function predicts and use the difference between the prediction and actual value to calculate the total cost.

Sample: cost\_function([1 , 2 ; 3 , 4 ; 2 , 3 ; 1.3 , 2.4; 2.4 3.2 ; 3.3 , 3.7] , 0 , 1 ) = 0.4175

## Problem 3: Generating Linear Data with Noise

Now that we know how the cost function works, we should try to generate some data that we can try to approximate. Write a function that takes three parameters: thetas, m, noiseSize. thetas will be the two constants, Θ0 and Θ1 in the linear relation

y = Θ0 + Θ1x

and m will be the number of data points to generate. Your function should pick m random x values, calculate the corresponding y value, and then apply a random noise to the y value. Then, it should return two values, x and y, which are the x coordinates and y coordinates of your random, linear data.

Function header:

function [x, y] = genData( thetas , m , noiseSize )