# HW 1

Remember that you may (and should!) talk about the problems amongst yourselves, or discuss them with me, providing attribution for any good ideas you might get – but your final write-up should be your own.

**1: A problem of performance.** Implement the band matrix multiply in algorithm 1.2.2 in MATLAB. For a variety of square matrix sizes $n$ and bandwidths $b$, compare the speed of

1. Your matrix multiply
2. Ordinary matrix multiply with a MATLAB dense matrix
3. Matrix multiply with a sparse matrix (use `As = sparse(A)` to make the sparse matrix object).

What do you observe about the relative performance of these three options?

**2: Seeking structure.** Given $n$ points $(x_i, y_i)$, define the *square distance matrix* $D$ by

$$d_{ij} = (x_i - x_j)^2 + (y_i - y_j)^2.$$

Write MATLAB code for an $O(n)$ algorithm to multiply the distance matrix with a random vector. Your program should have the interface

```
function [Dv] = dist2prod(x,y,v)

% Quickly compute Dv = D*v, where
% D(i,j) := (x(i)-x(j))^2 + (y(i)-y(j))^2.
```

**3: Something with singular values.** Suppose $A \in \mathbb{R}^{n\times m}$ is a fixed matrix and $z \in \mathbb{R}^m$ is a random vector with independent standard normal entries, i.e., $z_j \sim N(0,1)$. Find a simple closed-form expression for $E[\|AZ\|^2]$. Write an (efficient) MATLAB script to verify your computation.

*Hint*: Note that for any fixed orthogonal matrix $Q$, the entries of $Qz$ are independent standard normal random variables. Then use the SVD.

**4: Error in a classic recurrence.** The following routine estimates $\pi$ by recursively computing the semiperimeter of a sequence of $2^{k+1}$-gons embedded in the unit circle:

```
N = 4;
L(1) = sqrt(2);
s(1) = N*L(1)/2;
for k = 1:30
  N = N*2;
  L(k+1) = sqrt( 2*(1-sqrt(1-L(k)^2/4)) );
  s(k+1) = N*L(k+1)/2;
end
```

Plot the error $|s_k - \pi|$ against $k$. Explain why the algorithm behaves as it does. For extra credit, suggest a reformulation of the recurrence that does not suffer from this problem.