You may freely collaborate on section problems. They are not collected. Solutions are discussed in section and posted on the website. For maximum benefit, it is best to actually write MATLAB solutions and run them. Section problems and their variations can appear on course examinations. Problems are generally taken from *Introduction to Scientific Computing-A Matrix-Vector Approach Using Matlab* by C. Van Loan.

1. What is the output from the script \( v = [10;20;30]; w = [7;-2;3]; A = vw' \)? More generally, what is the \((i,j)\) entry of the matrix \( A = vw' \) where \( v, w \in \mathbb{R}^n \)? Complete the following function without any loops:

   ```matlab
   function S = SineM(n)
   % S is the n-by-n matrix whose (i,j) entry is sin(pi*i*j/(n+1))
   ```

2. Consider the following function

   ```matlab
   function H = HaarMatrix(t)
   % H is the n-by-n Haar matrix where n = 2^t.
   % Avoid values of t larger than 10!
   H = 1;
   for k=1:t
       H = [H H; H -H];
   end
   ```

   Write a script that computes just the last column of the \( n \)-by-\( n \) Haar matrix where \( n = 2^t \). Let \( \alpha_k \) be the number of entries in the \( 2^k \)-by-\( 2^k \) Haar matrix that have value 1. Write a Matlab script that prints \( \alpha_k/4^k \) for \( k = 1:20 \).

3. Assume that a function call and a flop each require \( \tau \) seconds to execute and consider the following function:

   ```matlab
   function y = Haar(x)
   % x a column n-vector where n is a power of 2.
   % y = H*x where H is the n-th order Haar matrix
   n = length(x);
   if n==1
       y = x;
   else
       m = n/2;
       z1 = Haar(x(1:m));
       z2 = Haar(x(m+1:n));
       y = [z1+z2;z1-z2];
   end
   ```

   How big must \( n \) be (approximately) for the function call overhead to be about one percent of the time spent on arithmetic? Repeat the problem for the case when a function call and a vector operation (of any length) requires \( \tau \) seconds.
4. Consider the non-recursive implementation of Haar:

```matlab
function y = HaarNonRecur(x)
    n = length(x);
    t = log2(n);
    for k=1:t
        L = 2^(k-1);
        m = n/L;
        A = reshape(x,L,m);
        % We have 2^(t-k) "Haar" combinations producing vectors of length 2^k
        Z1 = A(:,1:2:m)+A(:,2:2:m); % The sums
        Z2 = A(:,1:2:m)-A(:,2:2:m); % The differences
        A(:,1:2:m) = Z1;
        A(:,2:2:m) = Z2;
    end
    y = x;
```

For the $n = 8$ case, here is what the array $x$ contains as the loop progresses:

<table>
<thead>
<tr>
<th>Original</th>
<th>$k = 1$</th>
<th>$k = 2$</th>
<th>$k = 3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>$H_2x(1:2)$</td>
<td>$H_4x(1:4)$</td>
<td>$H_8x(1:8)$</td>
</tr>
<tr>
<td>$x_2$</td>
<td>$H_2x(3:4)$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_3$</td>
<td>$H_2x(5:6)$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_4$</td>
<td>$H_2x(7:8)$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_5$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_6$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_7$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_8$</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(a) "De-vectorize" this implementation by not using the reshape "device". (b) Implement the following function

```matlab
function Y = HaarNonRecur(X)
    n = length(X);
    t = log2(n);
    for k=1:t
        L = 2^(k-1);
        m = n/L;
        A = reshape(X,L,m);
        % We have 2^(t-k) "Haar" combinations producing vectors of length 2^k
        Z1 = A(:,1:2:m)+A(:,2:2:m); % The sums
        Z2 = A(:,1:2:m)-A(:,2:2:m); % The differences
        A(:,1:2:m) = Z1;
        A(:,2:2:m) = Z2;
    end
    Y = X;
```

Assume $n << m$. 

2