Kara Gorman
Homework #1
Math 226B: Winter 2018

Problem 1: Let $A \in \mathbb{R}^{n\times n}$ be a row-stochastic matrix, $\alpha \in \mathbb{R}$ a parameter, and

$$A_\alpha := \alpha A + (1-\alpha)\frac{1}{n}ee^T, \text{ where } e := \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \in \mathbb{R}^n.$$

Prove that the matrix A_α is row-stochastic for all parameter values $0 \leq \alpha \leq 1$.

Proof.

Case 1: $\alpha = 0$

$$A_0 = \frac{1}{n}ee^T,$$

where

$$\begin{aligned} B :&= \frac{1}{n}ee^T \\ &= \frac{1}{n}\begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}\begin{bmatrix} 1 & 1 & \dots & 1 \end{bmatrix} \\ &= \frac{1}{n}\begin{bmatrix} 1 & 1 & \dots & 1 \\ 1 & 1 & \dots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \dots & 1 \end{bmatrix} \\ &= \begin{bmatrix} \frac{1}{n} & \frac{1}{n} & \dots & \frac{1}{n} \\ \frac{1}{n} & \frac{1}{n} & \dots & \frac{1}{n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{n} & \frac{1}{n} & \dots & \frac{1}{n} \end{bmatrix} \end{aligned} \tag{1}$$

(1) is an $n \times n$ matrix where $\sum_{j=1}^{n} b_{ij} = 1$, for all $i = 1, \dots, n$, i.e., the sum of each row of matrix B is equal to 1. Thus, $A_0 = B$ is row-stochastic.

Case 2: $\alpha = 1$

$$A_1 = A$$

Since we are given that A is a row-stochastic matrix, then clearly A_1 is row-stochastic.

Case 3: $0 < \alpha < 1$

$$A_\alpha = \alpha A + (1-\alpha)\frac{1}{n}ee^T,$$

where A and $B = \frac{1}{n}ee^T$ are both row-stochastic matrices, as shown in cases 1 and 2. So, we can write A_α as:

$$A_\alpha = \begin{bmatrix} \alpha a_{11} + (1-\alpha)b_{11} & \alpha a_{12} + (1-\alpha)b_{12} & \dots & \alpha a_{1n} + (1-\alpha)b_{1n} \\ \alpha a_{21} + (1-\alpha)b_{21} & \alpha a_{22} + (1-\alpha)b_{22} & \dots & \alpha a_{2n} + (1-\alpha)b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \alpha a_{n1} + (1-\alpha)b_{n1} & \alpha a_{n2} + (1-\alpha)b_{n2} & \dots & \alpha a_{nn} + (1-\alpha)b_{nn} \end{bmatrix}.$$

If we look at the first row, we see that:

$$\begin{aligned} \sum_{j=1}^{n} a_{1j}^{\alpha} &= [\alpha a_{11} + (1-\alpha)b_{11}] + [\alpha a_{12} + (1-\alpha)b_{12}] + \cdots + [\alpha a_{1n} + (1-\alpha)b_{1n}] \\ &= \alpha(a_{11} + a_{12} + \cdots + a_{1n}) + (1-\alpha)(b_{11} + b_{12} + \cdots + b_{1n}) \\ &= \alpha(1) + (1-\alpha)(1) \\ &= \alpha + 1 - \alpha \\ &= 1 \end{aligned}$$

In general, for all $i = 1, \dots, n$,

$$\begin{aligned} \sum_{j=1}^{n} a_{ij}^{\alpha} &= [\alpha a_{i1} + (1-\alpha)b_{i1}] + [\alpha a_{i2} + (1-\alpha)b_{i2}] + \ldots + [\alpha a_{in} + (1-\alpha)b_{in}] \\ &= \alpha(a_{i1} + a_{i2} + \cdots + a_{in}) + (1-\alpha)(b_{i1} + b_{i2} + \cdots + b_{in}) \\ &= \alpha(1) + (1-\alpha)(1) \\ &= \alpha + 1 - \alpha \\ &= 1 \end{aligned}$$

Thus, $A_\alpha = \alpha A + (1-\alpha)\frac{1}{n}ee^T$ is row-stochastic for all α such that $0 \le \alpha \le 1$.

Problem 2: A small company runs an internal network of 10 websites, which have no links to the outside world. The links within the internal network are as follows:

- Website 1 has links to websites 5, 7;
- Website 2 has links to websites 2, 6, 7, 9;
- Website 4 has links to websites 4, 7;
- Website 5 has links to websites 7, 9, 10;
- Website 8 has links to websites 3, 4, 7, 8, 9;
- Website 9 has links to websites 2, 4, 7, 8, 10;
- Website 10 has links to websites 1, 3, 4, 7, 9, 10;

Formulate a linear algebra problem, the solution of which is the PageRank vector x of this internal network, and compute x.

According to your computed PageRank, what is the ranking of the 10 websites from most to least important?

Proof.
First, we need to construct the matrix Q, the matrix associated with the directed graph of the connectivity of the 10 given websites. Note that we do not include self-links in our Q matrix.

$$Q = \begin{bmatrix} 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{3} & \frac{1}{3} & 0 & \frac{1}{3} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{3} & 0 & \frac{1}{3} & \frac{1}{3} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{4} & \frac{1}{4} & 0 & 0 & \frac{1}{4} & 0 & \frac{1}{4} & 0 \\ 0 & \frac{1}{5} & 0 & \frac{1}{5} & 0 & 0 & \frac{1}{5} & \frac{1}{5} & 0 & \frac{1}{5} \\ \frac{1}{5} & 0 & \frac{1}{5} & \frac{1}{5} & 0 & 0 & \frac{1}{5} & 0 & \frac{1}{5} & 0 \end{bmatrix}$$

Now, we can construct the vector $v = [v_j]_{j=1,\dots,n} \in \mathbb{R}^n$, which is defined by

$$v_j = \begin{cases} 1 \text{ if } d_j = 0 \\ 0 \text{ if } d_j > 0, \end{cases}$$

where d_j denotes the out degree of node j of Q. For this problem,

$$v = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Now, we can use the following Matlab function to find the eigenvalue and eigenvectors of A^T, where

$$A = Q + \frac{1}{n} v e^T.$$

Listing 1: PageRank Function

```
function [y] = PageRank(Q,v)

n = length(v);
e = ones(n,1);

A = Q + (1/n)*v*transpose(e);

AT = transpose(A);

[V,D] = eig(AT);

end
```

This Matlab function produces the eigenvalues and eigenvectors of A^T. To determine the page rankings, we identify the leading order eigenvalue, $\lambda = 1$, and then its corresponding eigenvector

$$u = \begin{bmatrix} -1.611685764370067e-01 \\ -1.785251923609917e-01 \\ -2.057998745272545e-01 \\ -2.717395180598953e-01 \\ -1.931698370468546e-01 \\ -1.720939462820155e-01 \\ -7.479616674742428e-01 \\ -1.785251923609919e-01 \\ -3.296982176632033e-01 \\ -2.429151380432770e-01 \end{bmatrix},$$

determines the page rankings of each website. The website with the highest magnitude is the most important. So, the wedsites in order from most to least important are:

$$7, 9, 4, 10, 3, 5, 8, 2, 6, 1.$$

Problem 3: Let $A \in \mathbb{R}^{n \times n}$ be a row-stochastic matrix.

(a) Show that all eigenvalues λ of A satisfy $|\lambda| \leq 1$.

Proof. From the definition of row-stochasticity, we know that A has the eigenvalue $|\lambda_1| = 1$ with the eigenvector $u_1 = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}$.

Now, assume there is an eigenvalue $|\lambda_2| > 1$ of A, with eigenvector u_2. Then, $A^n u_2 = |\lambda_2|^n u_2$ grows exponentially as n increases. However, this implies that for large n, there is an entry of matrix A such that $a_{jk}^n > 1$. But this is a contradiction since the product of two row-stochastic matrices is also a row-stochastic matrix, thus, A^n is a row-stochastic matrix. Therefore, all eigenvalues of A satisfy $|\lambda| \leq 1$.

(b) Show that

$$x^{(i)} \geq 0 \text{ and } \sum_{j=1}^{n} x_j^{(i)} = \sum_{j=1}^{n} x_j^{(0)} \text{ for all } i \geq 0.$$

(ii) $x^{(i)} \geq 0$

Proof. Since A is row-stochastic, then A^T is column-stochastic.

Base Case: Let $i = 0$. Since we are given that $x_n^{(0)} \geq 0$ for all n, and $a_{jk}^T \geq 0$ for all $j, k = 1, \ldots, n$ since it is column-stochastic. Then, it follows that all entries of the product $A^T x^{(0)}$ are positive. Thus, $x^{(1)} = A^T x^{(0)} \geq 0$.

Inductive Step: By induction, we assume $x^{(i-1)} \geq 0$, and can show that $x^{(i)} \geq 0$. Similarly to the base case, since A^T is row-stochastic, then $a_{jk}^T \geq 0$ for all $j, k = 1, \ldots, n$. Then, it follows that all the entries of the product $A^T x^{(i-1)}$ are positive. Thus, $x^{(i)} \geq 0$ for all $i = 0, 1, 2, \ldots$.

(ii) $\sum_{j=1}^n x_j^{(i)} = \sum_{j=1}^n x_j^{(0)}$ for all $i \geq 0$

Proof.

Base Case: Let $i = 0$. Then,

$$\begin{aligned} x^{(1)} &= A^T x^{(0)} \\ &= \sum_{k=1}^{n} a_{jk}^T x_k^{(0)} \end{aligned}$$

Then, we have:

$$\begin{aligned} \sum_{j=1}^{n} x_j^{(1)} &= \sum_{j=1}^{n} \left(\sum_{k=1}^{n} a_{jk}^T x_k^{(0)} \right) \\ &= \sum_{k=1}^{n} \left(\sum_{j=1}^{n} a_{jk}^T \right) x_k^{(0)} \\ &= \sum_{k=1}^{n} x_k^{(0)}. \end{aligned}$$

$\sum_{j=1}^n a_{jk}^T = 1$ since A^T is column-stochastic. Thus, it follows that

$$\sum_{j=1}^{n} x_j^{(1)} = \sum_{j=1}^{n} x_j^{(0)}.$$

Inductive Step: By induction, assume:

$$\sum_{j=1}^{n} x_j^{(i-1)} = \sum_{j=1}^{n} x_j^{(i-2)} = \cdots = \sum_{j=1}^{n} x_j^{(0)}.$$

Then,

$$\begin{aligned} x^{(i)} &= A^T x^{(i-1)} \\ &= \sum_{k=1}^{n} a_{jk}^T x_k^{(i-1)} \end{aligned}$$

Then, we have:

$$\begin{aligned} \sum_{j=1}^{n} x_j^{(i)} &= \sum_{j=1}^{n} \left(\sum_{k=1}^{n} a_{jk}^T x_k^{(i-1)} \right) \\ &= \sum_{k=1}^{n} \left(\sum_{j=1}^{n} a_{jk}^T \right) x_k^{(i-1)} \\ &= \sum_{k=1}^{n} x_k^{(i-1)} \end{aligned}$$

Thus,

$$\sum_{j=1}^{n} x_j^{(i)} = \sum_{j=1}^{n} x_k^{(i-1)}.$$

Therefore, it follows that:

$$\sum_{j=1}^{n} x_j^{(i)} = \sum_{j=1}^{n} x_j^{(0)}.$$

(c) Show that the vectors $x^{(i)}$ generated by the iteration

$$x^{(i+1)} = A^T x^{(i)}, \ i = 0, 1, 2, \ldots$$

converge and that the limit

$$x = \lim_{i \to \infty} x^{(i)}$$

satisfies $A^T x = x$, $x \geq 0$, and $x \neq 0$.

Proof.

$$\begin{aligned}
\lim_{i\to\infty} x^{(i)} &= \lim_{i\to\infty} \left(A^T x^{(i-1)}\right) \\
&= \lim_{i\to\infty} \left(A^T \left(A^T x^{(i-2)}\right)\right) \\
&\vdots \\
&= \lim_{i\to\infty} \left(A^T\right)^i x^{(0)} \\
&= X \left(\lim_{i\to\infty} L^i\right) X^{-1} x^{(0)},
\end{aligned}$$

where

$$L = \begin{bmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \lambda_n \end{bmatrix}.$$

Then,

$$\begin{aligned}
\lim_{i\to\infty} x^{(i)} &= X \left(\lim_{i\to\infty} L^i\right) X^{-1} x^{(0)} \\
&= X \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & 0 \end{bmatrix} X^{-1} x^{(0)},
\end{aligned}$$

where the product

$$X \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & 0 \end{bmatrix} X^{-1} x^{(0)}$$

equals the first column of X, i.e., the eigenvector, u_1, corresponding to the dominant eigenvalue $|\lambda_1| = 1$. So,

$$\begin{aligned}\lim_{i\to\infty} x^{(i)} &= X \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & 0 \end{bmatrix} X^{-1} x^{(0)} \\ &= u_1 x^{(0)} \\ &=: x\end{aligned}$$

Thus, the vectors generated by the iteration converge to x.

Now we need to show that x satisfies the conditions $A^T x = x$, $x \geq 0$, and $x \neq 0$. First, since $x = \lim_{i\to\infty} x^{(i)}$, and we have that $x^{(i+1)} = A^T x^{(i)}$, then for i large, $x^{(i+1)} \approx x^{(i)} = x$. Thus, $A^T x = x$. Second, we are given that $x^{(0)} \geq 0$ and $x^{(0)} \neq 0$, and we know that A is row-stochastic, so it is non-negative. Then, it follows that $x^{(1)} = A^T x^{(0)} \geq 0$. In part (b), we showed that $x^{(i)} \geq 0$ for all $i \geq 0$. So clearly, $x = \lim_{i\to\infty} x^{(i)} \geq 0$.

Problem 4: Let G be a directed graph that describes the connectivity of a set n websites, and let $Q \in \mathbb{R}^{n\times N}$ and $A \in \mathbb{R}^{n\times n}$ be the associated matrices, as defined in class.

(a) Write two Matlab functions, which use n, Q, and J_v as inputs, to compute matrix-vector products of the form

$$y = A^T x, \text{ where } x \in \mathbb{R}^n \text{ is a dense vector in general,}$$

and

$$y = A_\alpha^T x, \text{ where } x \in \mathbb{R}^n \text{ is a dense vector in general,}$$

respectively, as efficiently as possible.

Listing 2: Matrix-Vector Product of Row Stochastic Matrix

```
function [y] = RowStochastic(n,Q,Jv,x0)

e = ones(n,1);
v = zeros(n,1);

for i=1:length(Jv)
    v(Jv(i)) = 1;
end
```

```
v = sparse(v);
vT = transpose(v);
vTx0 = vT*x0;
QT = transpose(Q);

ATx0 = QT*x0 + (1/n)*e*vTx0;

y = ATx0;
end
```

Listing 3: Matrix-Vector Product of Row Stochastic Matrix with α

```
function [yalpha] = RowStochasticAlpha(n,Q,Jv,alpha,x0)

e = ones(n,1);
v = zeros(n,1);

for i=1:length(Jv)
    v(Jv(i)) = 1;
end

v = sparse(v);
vT = transpose(v);
vTx0 = vT*x0;
QT = transpose(Q);
eTx0 = transpose(e)*x0;

ATx0 = QT*x0 + (1/n)*e*vTx0;
ATx0_alpha = alpha*ATx0 + (1 - alpha)*(1/n)*e*eTx0;

yalpha = ATx0_alpha;
end
```

Results:

$A \in \mathbb{R}^{10\times 10}$	$\alpha = 1$	$\alpha = 0.5$	$\alpha = 0.85$
x	e	e	e
y	$\begin{bmatrix} 5.000000000000000e-01 \\ 5.000000000000000e-01 \\ 7.500000000000000e-01 \\ 9.500000000000000e-01 \\ 8.000000000000000e-01 \\ 6.333333333333333e-01 \\ 3.116666666666666e+00 \\ 5.000000000000000e-01 \\ 1.416666666666667e+00 \\ 8.333333333333333e-01 \end{bmatrix}$	$\begin{bmatrix} 7.500000000000001e-01 \\ 7.500000000000000e-01 \\ 8.750000000000000e-01 \\ 9.750000000000001e-01 \\ 9.000000000000000e-01 \\ 8.166666666666667e-01 \\ 2.058333333333334e+00 \\ 7.500000000000000e-01 \\ 1.208333333333333e+00 \\ 9.166666666666667e-01 \end{bmatrix}$	$\begin{bmatrix} 5.750000000000001e-01 \\ 5.750000000000001e-01 \\ 7.875000000000001e-01 \\ 9.575000000000001e-01 \\ 8.300000000000001e-01 \\ 6.883333333333334e-01 \\ 2.799166666666667e+00 \\ 5.750000000000001e-01 \\ 1.354166666666667e+00 \\ 8.583333333333334e-01 \end{bmatrix}$

$A \in \mathbb{R}^{685230\times 685230}$	$\alpha = 1$	$\alpha = 0.85$
x	x_0	x_0
y(2)	-4.833020050815520e-02	-4.108411686447250e-02
y(222222)	1.715159442482511e-01	1.457851061784728e-01
y(300000)	9.711221038679012e-03	8.251091450336577e-03
y(400000)	-7.835267507349471e-02	-6.660322024501107e-02

(b) Write a Matlab program that implements the power method, as stated in the notes provided on the course website.

Listing 4: Power Method Function

```
function [y] = PowerMethod(x, epsilon, kMax, n, Q, Jv, alpha)

lambda = [];
x_abs = abs(x);
[~,b] = max(x_abs);
X = x(b);
lambda(1) = X;

itCount = 0;

%%%
```

```
e = ones(n,1);
v = zeros(n,1);

for i=1:length(Jv)
    v(Jv(i)) = 1;
end

v = sparse(v);
vT = transpose(v);
QT = transpose(Q);
%%%

for k=2:kMax
    vTx = vT*x;
    ATx = QT*x + (1/n)*e*vTx;
    eTx = transpose(e)*x;
    x = alpha*ATx + (1-alpha)*(1/n)*e*eTx;
    x_abs = abs(x);
    %[~,b] = sort(x_abs,'descend');
    [~,b] = max(x_abs);
    X = x(b);
    lambda(k) = X;
    x = x/lambda(k);
    itCount = itCount + 1;

    if (abs(lambda(k) - lambda(k-1)) <= epsilon*(abs(lambda(k-1))))
        break
    end
end

itCount = itCount
evalApprox = lambda(itCount+1)
evecApprox = x;
vTx = vT*x;
ATx = QT*x + (1/n)*e*vTx;
eTx = transpose(e)*x;
ATalphax = alpha*ATx + (1-alpha)*(1/n)*e*eTx;
relEigResidual = norm((ATalphax - evalApprox*x),inf)/norm(x,inf)
v = x;
[~,d] = sort(v, 'descend');
pageRank = d(1:10)
end
```

*note: I know that it would be simpler to call my function RowStochasticAlpha within my PowerMethod function. However, I was having difficulties getting my code to work doing it that way so I essentially just put everything from my RowStochasticAlpha function into my PowerMethod function, and it works that same as if I had called the RowStochasticAlpha function.

(c) Use your program from (b) to compute the dominate eigenvalue λ and a corresponding eigenvector x of the matrix A^T corresponding to the case of $n = 10$ websites stated in Problem 2.

	$x = e$	$x = [1; -1; \ldots; 1; -1]$
α	1	1
Iterations	47	100
λ	1.000000000000000e+00	9.999999999999999e-01
Residual	5.551115123125783e-16	1.665334536937735e-16
x	$\begin{bmatrix} 2.154770537656686e-01 \\ 2.386822749404329e-01 \\ 2.751476225007769e-01 \\ 3.633067440174038e-01 \\ 2.582616803066403e-01 \\ 2.300839117372837e-01 \\ 1.000000000000000e+00 \\ 2.386822749404329e-01 \\ 4.407956075831347e-01 \\ 3.247695017093130e-01 \end{bmatrix}$	$\begin{bmatrix} 2.154770537656687e-01 \\ 2.386822749404331e-01 \\ 2.751476225007770e-01 \\ 3.633067440174039e-01 \\ 2.582616803066405e-01 \\ 2.300839117372837e-01 \\ 1.000000000000000e+00 \\ 2.386822749404331e-01 \\ 4.407956075831347e-01 \\ 3.247695017093132e-01 \end{bmatrix}$
Page Rank	9, 4, 10, 3, 5, 2, 8, 6, 1, 7	9, 4, 10, 3, 5, 2, 8, 6, 1, 7

*Note: Websites 2 and 8 have the same page rank, so their order is interchangeable.

(d) Use your program from (b) to try to compute the dominate eigenvalue λ and a corresponding eigenvector x of the matrix A^T corresponding to the large graph with $n = 685230$ nodes and 7600595 edges given in "www0.mat".

	$x = e$	$x = x_0$	$x = e$	$x = x_0$
α	1	1	0.85	0.85
Iterations	9999	9999	122	163
λ	1.000000895792460e+00	1.124042805557909e+00	9.999999999952399e-01	1.000000000004515e+00
Residual	2.029381492183098e-01	3.622262713234378e-01	1.105315166650966e-10	1.688181712478087e-10
Page Rank	$\begin{bmatrix}403967\\333816\\407349\\158244\\208355\\310294\\115404\\463748\\569443\\525917\end{bmatrix}$	$\begin{bmatrix}407349\\158244\\553109\\505281\\487780\\569443\\525917\\665540\\187367\\382960\end{bmatrix}$	$\begin{bmatrix}629103\\328995\\176090\\5397\\609117\\539881\\63085\\351516\\529968\\363332\end{bmatrix}$	$\begin{bmatrix}629103\\328995\\176090\\5397\\609117\\539881\\63085\\351516\\529968\\363332\end{bmatrix}$

For the last two cases, when $\alpha = 0.85$, the convergence is very fast. Unlike the first two cases, when $\alpha = 1$, that don't converge within the maximum number of iterations. So, the page rankings for the $\alpha = 1$ cases are meaningless. Whereas when $\alpha = 0.85$, the page ranking is the same regardless of what our initial value of x was.

Problem 5:

(a) Show that all n eigenvalues of an $n \times n$ circulant matrix C can be computed via a single call to Matlab's "fft" command.

Proof.
Let H be the Hermitian operator, such that

$$A^H := \overline{A}^T = \overline{A^T}.$$

Note that since $\overline{F}$ is symmetric, then

$$\overline{F}^T = \overline{F}.$$

In the problem statement, the eigenvalues are defined as:

$$\lambda_k = \sum_{j=0}^{n-1} c_j exp\left(\frac{2\pi i}{n} jk\right),$$

which can also be written as:

$$\Lambda = \overline{F}c = \overline{F}^T c,$$

since $\overline{F}$ is symmetric. Then,

$$\begin{aligned}
\overline{F}c &= \overline{F}^T c \\
&= F^H c \\
&= \left(c^H F\right)^H \\
&= \left(c^T F\right)^H \\
&= \left(\left(F^T c\right)^T\right)^H \\
&= \overline{\left((Fc)^T\right)^T} \\
&= \overline{(Fc)}
\end{aligned} \tag{2}$$

So, combining (2) with our result from part (a), we have that

$$\Lambda = \overline{F}c = \overline{\text{fft}(c)}.$$

Thus, the eigenvalues of the circulant matrix C can be computed in Matlab with the command

conj(fft(c)).

(b) Use part (a) and the factorization $C = \frac{1}{n}\overline{F}\Lambda F$ to show that each matrix-vector product $y = Cx$ with a circulant matrix C can be computed via three calls to Matlab's "fft" command.
Write a Matlab function that implements this approach to efficiently compute matrix-vector products with circulant matrices. Use your program to compute $y = Cx$ for

$$C = \begin{bmatrix} 2 & -1 & 0 & -3 \\ -3 & 2 & -1 & 0 \\ 0 & -3 & 2 & -1 \\ -1 & 0 & -3 & 2 \end{bmatrix} \text{ and } x = \begin{bmatrix} -1 \\ 2 \\ 1 \\ 4 \end{bmatrix}.$$

Proof.

$$\begin{aligned} Cx &= \frac{1}{n}\overline{F}\Lambda Fx \\ &= \frac{1}{n}\overline{F}(\text{conj}(\text{fft}(\text{c})))Fx \\ &= \frac{1}{n}\overline{F}(\text{conj}(\text{fft}(\text{c})))(\text{fft}(\text{x})) \\ &= \frac{1}{n}\text{conj}[\text{fft}(\text{conj}(\text{fft}(\text{c}))(\text{fft}(\text{x})))] \end{aligned}$$

Listing 5: Circulant Matrix-Vector Product Function

```
function [y_fast] = fftCirculant(c,x)

n = length(x);

lambdaVec = conj(fft(c));

y_exact = C*x;
y_fast = (1/n)*conj(fft(conj(lambdaVec.*fft(x))));
end
```

Results:

$$y_{\text{exact}} = \begin{bmatrix} -16 \\ 6 \\ -8 \\ 6 \end{bmatrix}$$

$$y_{\text{fast}} = \begin{bmatrix} -16 \\ 6 \\ -8 \\ 6 \end{bmatrix}$$

(c) Show that for any given Toeplitz matrix $T \in \mathbb{R}^{n\times n}$ there is a unique circulant matrix $C \in \mathbb{R}^{(2n-1)\times(2n-1)}$ such that T is the leading principal $n \times n$ submatrix of C.

Proof.
The Toeplitz matrix T is defined as

$$T = [t_{k-j}]_{j,k=1,2,\ldots,n} \in \mathbb{R}^{n\times n},$$

where

$$t := \begin{bmatrix} t_{-(n-1)} & t_{-(n-2)} & \ldots & t_{-1} & t_0 & t_1 & \ldots & t_{n-1} \end{bmatrix}.$$

So, our Toeplitz matrix T is of the form:

$$T = \begin{bmatrix} t_0 & t_1 & t_2 & \ldots & t_{n-1} \\ t_1 & t_0 & t_1 & \ldots & t_{n-2} \\ \vdots & \ddots & \ddots & & \vdots \\ t_{-(n-2)} & t_{-(n-3)} & \ldots & \ldots & \vdots \\ t_{-(n-1)} & t_{-(n-2)} & \ldots & \ldots & t_0 \end{bmatrix}.$$

Since we want to show that for any given Toeplitz matrix $T \in \mathbb{R}^{n\times n}$ there is a unique circulant matrix $C \in \mathbb{R}^{(2n-1)\times(2n-1)}$ such that T is the leading principal $n \times n$ submatrix of C, then we need to redefine t in terms of c, where

$$c := \begin{bmatrix} c_0 & c_1 & c_2 & \ldots & c_{n-1} \end{bmatrix},$$

as follows:

$$t = \begin{bmatrix} c_n & c_{n+1} & \ldots & c_{2n-1} & c_0 & c_1 & c_2 & \ldots & c_{n-1} \end{bmatrix}.$$

Now, we can write our circulant matrix C, with leading submatrix T as:

$$C = \left[\begin{array}{ccccc|ccc} c_0 & c_1 & c_2 & \ldots & c_{n-1} & c_n & \ldots & c_{2n-2} & c_{2n-1} \\ c_{2n-1} & c_0 & c_1 & \ldots & c_{n-2} & c_{n-1} & \ldots & c_{2n-3} & c_{2n-2} \\ c_{2n-2} & c_{2n-1} & c_0 & \ldots & c_{n-3} & c_{n-2} & \ldots & c_{2n-4} & c_{2n-3} \\ \vdots & \ddots & \ddots & & \vdots & \vdots & \ddots & & \vdots \\ c_n & c_{n+1} & c_{n+2} & \ldots & c_0 & \vdots & \ddots & \ddots & \vdots \\ \hline c_{n-1} & c_n & c_{n+1} & \ldots & c_1 & \vdots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & & \vdots & \vdots & \ddots & \ddots & \vdots \\ c_2 & \ldots & \ddots & \ddots & \vdots & \vdots & \ddots & \ddots & \vdots \\ c_1 & c_2 & c_3 & \ldots & c_n & c_{n+1} & \ldots & \ldots & c_0 \end{array}\right].$$

Where the upper left block is the $n \times n$ Toeplitz matrix T, expressed in terms of c. The lower left block is $(n-1) \times n$, the upper right block is $n \times (n-1)$, and the lower right block is $(n-1) \times (n-1)$. So, C is $(2n-1) \times (2n-1)$.

Now, we need to check that the property of Toeplitz matrices,

$$t_{-j} = t_{n-j} \text{ for all } j = 1, 2, \ldots, n-1,$$

holds when t is written in terms of c, as defined above.

First, let's look at the case of $j = 1$. Since T is Toeplits, then

$$t_{-1} = t_{n-1}.$$

So, in terms of c, this equates to

$$c_{2n-1} = c_{n-1}.$$

Which we can see are in the correct spots of the Toeplitz block (upper left block) of our circulant matrix C, so this property holds for $j = 1$.

Next, looking a the case of $j = n-1$, then we have that

$$t_{-(n-1)} = t_1$$

corresponds to

$$c_n = c_1.$$

Which we can see, as before, are in the correct positions of our matrix C.
So, in general, if

$$t_{-j} = t_{n-j},$$

then

$$c_{2n-j} = c_{n-j} \text{ for all } j = 1, 2, \ldots, n-1.$$

Thus, for any given Toeplitz matrix T there is a unique circulant matrix C such that T is the leading principal $n \times n$ submatrix of C.

(d) Use (c) to devise an algorithm that computes matrix-vector products $y = Tx$ with Toeplitz matrices T via your Matlab function from (b).
Write a Matlab function that implements this approach.

Listing 6: Toeplitz Matrix-Vector Product Function

```
function [y] = fftToeplitz(t,x)

n = length(x);

c = zeros(1,2*n-1);

c(1:n) = t(n:2*n-1);
c(n+1:2*n-1) = t(1:n-1);

x_tilde = zeros(2*n-1,1);

for i = 1:n
    x_tilde(i) = x(i);
end

c = c';

y_tilde = fftCirculant(c,x_tilde);

y = y_tilde(1:n);

end
```

(e)

(i) To test your program from (d), first employ it to compute $y = Tx$ for

$$T = \begin{bmatrix} 2 & -1 & 4 & -3 \\ 1 & 2 & -1 & 4 \\ -5 & 1 & 2 & -1 \\ 6 & -5 & 1 & 2 \end{bmatrix} \text{ and } x = \begin{bmatrix} -1 \\ 2 \\ 1 \\ 4 \end{bmatrix}.$$

Results:

$$y_{\text{exact}} = \begin{bmatrix} -12 \\ 18 \\ 5 \\ -7 \end{bmatrix}, \; y_{\text{fast}} = \begin{bmatrix} -1.199999999999999e+01 \\ 1.800000000000000e+01 \\ 5.000000000000000e+00 \\ -7.000000000000001e+00 \end{bmatrix}$$

(ii) Now employ your program to compute $y = Tx$ where T is the $n \times n$ Toeplitz matrix given by the row vector t.

Result	Value
y_1	4.743400116337570e+01 - 1.897571643350681e-14i
y_{100000}	-2.044536337994736e+02 + 3.463356617877540e-14i
y_{200000}	9.854729389973313e+01 - 2.893191580611515e-14i
y_{300000}	-2.273143080393689e+01 + 4.576858382960005e-14i
y_{400000}	4.136625635601638e+02 - 5.480763011145770e-14i
y_{500000}	2.524304751530099e+02 + 9.171369375379089e-14i
$\sum_{j=1}^{n} y_j$	1.030143367304306e+05 + 7.836054370961585e-13i
$\|y\|_2$	1.669299439051479e+05