**P1.** Solve a linear system *Ax = b* using Gaussian Elimination.

**1.** Gaussian elimination function

function [L,U] = GE(A)

% [L,U] = GE(A)

%

% The LU factorization without pivoting. If A is n-by-n

% and has an

% LU factorization, then L is unit lower triangular and U

% is upper

% triangular so A = LU.

[n,n] = size(A);

for k=1:n-1

A(k+1:n,k) = A(k+1:n,k)/A(k,k);

A(k+1:n,k+1:n) = A(k+1:n,k+1:n) - A(k+1:n,k)\*A(k,k+1:n);

end

L = eye(n,n) + tril(A,-1);

U = triu(A);

**2.** Nonsingular lower triangular system

function x = LTriSol(L,b)

% x = LTriSol(L,b)

%

% Solves the nonsingular lower triangular system Lx = b

% where L is n-by-n, b is n-by-1, and x is n-by-1.

n = length(b);

x = zeros(n,1);

for j=1:n-1

x(j) = b(j)/L(j,j);

b(j+1:n) = b(j+1:n) - L(j+1:n,j)\*x(j);

end

x(n) = b(n)/L(n,n);

**3.** Nonsingular upper triangular system

function x = UTriSol(U,b)

% x = UTriSol(U,b)

%

% Solves the nonsingular upper triangular system Ux = b.

% where U is n-by-n, b is n-by-1, and X is n-by-1.

n = length(b);

x = zeros(n,1);

for j=n:-1:2

x(j) = b(j)/U(j,j);

b(1:j-1) = b(1:j-1) - x(j)\*U(1:j-1,j);

end

x(1) = b(1)/U(1,1);

**4.** Test

n = 5

A = magic(n)

b = [2 3 4 5 6]'

[L, U] = GE(A)

y = LTriSol(L,b)

x = UTriSol(U,y)

+ Result:

n = 5

A =

17 24 1 8 15

23 5 7 14 16

4 6 13 20 22

10 12 19 21 3

11 18 25 2 9

b = 2 3 4 5 6

L =

1.0000 0 0 0 0

1.3529 1.0000 0 0 0

0.2353 -0.0128 1.0000 0 0

0.5882 0.0771 1.4003 1.0000 0

0.6471 -0.0899 1.9366 4.0578 1.0000

U =

17.0000 24.0000 1.0000 8.0000 15.0000

0 -27.4706 5.6471 3.1765 -4.2941

0 0 12.8373 18.1585 18.4154

0 0 0 -9.3786 -31.2802

0 0 0 0 90.1734

y = 2.0000 0.2941 3.5332 -1.1468 2.5434

x = 0.0282 0.0282 0.1949 0.0282 0.0282

**P2.** Matrix Operations

**1.** Discrete Fourier Transform

function y = DFT(x)

% y = DFT(x)

% y is the discrete Fourier transform of a column n-

% vector x.

n = length(x);

y = x(1)\*ones(n,1);

if n > 1

v = exp(-2\*pi\*sqrt(-1)/n).^(0:n-1)';

for k=2:n

z = rem((k-1)\*(0:n-1)',n ) +1;

y = y + v(z)\*x(k);

end

end

**2.** Fast Fourier Transform Recursion

function y = FFTRecur(x)

% y = FFTRecur(x)

% y is the discrete Fourier transform of a column n-

% vector x where

% n is a power of two.

n = length(x);

if n ==1

y = x;

else

m = n/2;

yT = FFTRecur(x(1:2:n));

yB = FFTRecur(x(2:2:n));

d = exp(-2\*pi\*sqrt(-1)/n).^(0:m-1)';

z = d.\*yB;

y = [ yT+z ; yT-z ];

end

**3.** Strassen Multiplication

**(a).** Function Strass.m

function C = Strass(A,B,nmin)

% C = Strass(A,B,nmin)

% This computes the matrix-matrix product C = A\*B (via

% the Strassen Method) where

% A is an n-by-n matrix, B is a n-by-n matrix and n is a

% power of two. Conventional

% matrix multiplication is used if n<nmin where nmin is a

% positive integer.

[n,n] = size(A);

if n < nmin

C = A\*B;

else

m = n/2; u = 1:m; v = m+1:n;

P1 = Strass(A(u,u)+A(v,v),B(u,u)+B(v,v),nmin);

P2 = Strass(A(v,u)+A(v,v),B(u,u),nmin);

P3 = Strass(A(u,u),B(u,v)-B(v,v),nmin);

P4 = Strass(A(v,v),B(v,u)-B(u,u),nmin);

P5 = Strass(A(u,u)+A(u,v),B(v,v),nmin);

P6 = Strass(A(v,u)-A(u,u),B(u,u) + B(u,v),nmin);

P7 = Strass(A(u,v)-A(v,v),B(v,u)+B(v,v),nmin);

C = [ P1+P4-P5+P7 P3+P5; P2+P4 P1+P3-P2+P6];

end

**(b).** Script TestStrass.m

% Script TestStrass.m

V=2:9

A=vander(V')

C=1:2:15

R=[1 2:2:14]

B=toeplitz(C',R')

nmin=8

C=Strass(A,B,nmin)

+ Result:

V = 2 3 4 5 6 7 8 9

A =

128 64 32 16 8 4 2 1

2187 729 243 81 27 9 3 1

16384 4096 1024 256 64 16 4 1

78125 15625 3125 625 125 25 5 1

279936 46656 7776 1296 216 36 6 1

823543 117649 16807 2401 343 49 7 1

2097152 262144 32768 4096 512 64 8 1

4782969 531441 59049 6561 729 81 9 1

C = 1 3 5 7 9 11 13 15

R = 1 2 4 6 8 10 12 14

B =

1 2 4 6 8 10 12 14

3 1 2 4 6 8 10 12

5 3 1 2 4 6 8 10

7 5 3 1 2 4 6 8

9 7 5 3 1 2 4 6

11 9 7 5 3 1 2 4

13 11 9 7 5 3 1 2

15 13 11 9 7 5 3 1

nmin = 8

C =

749 623 817 1171 1605 2079 2573 3077

6552 6553 10928 16761 23080 29561 36096 42649

36403 41865 75999 117301 160395 203937 247591 291273

146480 185543 349606 538669 732732 927795 1123058 1318361

470289 638251 1226117 1883967 2553481 3224939 3896721 4568557

1281064 1830093 3555612 5449201 7366800 9287829 11209348 13130937

3081527 4579493 8961043 13703041 18490095 23282781 28076171 32869649

6726048 10313275 20277794 30950901 41702740 52463327 63224886 73986553