CMSC 828D: Fundamentals of Computer Vision Homework1

Instructors : Larry Davis, Ramani Duraiswami,
Daniel DeMenthon, and Yiannis Aloimonos
Solution based on homework submitted by Haiying Liu

1. Create random vectors ‘a’ and ‘b’ …

`a * b` * represents matrix multiply. It requires that the number of columns in the first matrix must be equal to the number of rows in the second matrix. In the present case, since both `a` and `b` are 10 × 1 vectors, this command will yield error message:

```
??? Error using ==> *
Inner matrix dimensions must agree.
```

`a .* b` .* represents element by element multiplication of matrices, and requires that the dimensions of the two matrixes agree. The result `r = a .* b` will have the same dimension with entries $r_{ij} = a_{ij} \times b_{ij}$.

`a / b` This is right matrix divide. If `r = a / b`, then we have `a = r * b`, where ‘`*`’ is matrix multiplication.

`a ./ b` This performs a term by term division. It requires that the dimension of the two matrixes agree. The result `r = a ./ b` will have the same dimension with entries $r_{ij} = a_{ij} \div b_{ij}$.

`a * b′` This is matrix multiply with the transpose (“`′`” is transpose). Since after the transposition, the number of columns of `a` is equal to the number of rows `b`^{T}, which is 1, the command will yield a 10 × 10 matrix with entry $ans_{ij} = \sum_{k=1}^{N} a_{ik} \cdot b_{kj}$, where the N is the number of columns of `a`, or number of rows of `b`^{T}.

`a \ A` This is left matrix divide, roughly equal to `a`$^+$ `* A`, where is `a`$^+$ the pseudo inverse matrix of `a` — `(a`^{T} `* a)`$^{-1}$ `* a`^{T} — in this example. If `r = a \ A`, then `a`^{T} `* A = a`^{T} `* a * r`, or `A`^{T} `* a = (a * r)`^{T} `* a`. Here the ‘`*`’ is matrix multiply.

An example of these commands is listed in the appendix.

2. Colon notation …

`1:10` This yields an array `[1 2 3 … 10]`.

`a(1:5)` This outputs the first five elements of `a`.

`a(4:9) + b(1:6)` This adds the fourth through ninth elements of `a to` the first through sixth elements of `b and produces a six dimensional` vector.

`0.1:0.1:100` This yields a 1000 element array with entries beginning from 0.1 and ending at 100, i.e. `[0.1, 0.2, 0.3, …, 100]`.

3. Boolean variables …

These commands employ Boolean return those entries in `b` whose values are bigger than the correspondent entries in `a`. They are explained in detail as follow:

`bga = b > a` The `bga` will be a logical array with entries $bga_i = \begin{cases} 1, \text{if } b_i > a_i \\ 0, \text{otherwise} \end{cases}$.

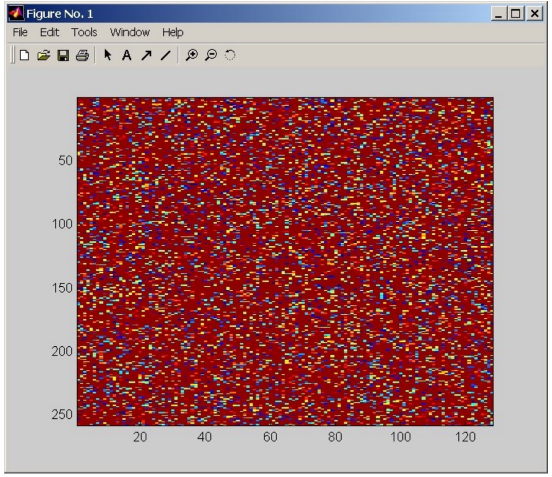
`Ib = find(bga)` The function `find` finds the indices of non-zero elements of `bga`. In this example, array `Ib` contains the indices of those elements of `b` that are larger than the corresponding element of `a`.

`b(Ib)` This command returns those entries of `b` with indices saves in `Ib`. In this example, they are actually the entries in `b` whose values are bigger than the correspondent entries in `a`.

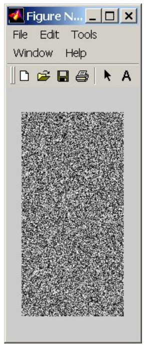
4. Image

The output of the two commands are shown as follow:

`image(I):`



`imshow(I):`



image(I) displays matrix I as an image. Each element of I specifies the color of a rectilinear patch in the image. I can be a matrix of dimension M×N or M×N×3, and can contain double, uint8, or uint16 data. The image will always displayed as a square image. When I is a 2-dimensional M×N matrix, the elements of I are used as indices into the current colormap to determine the color. image(I) places the center of element I(1,1) at (1,1) on the axes, and the center of element (M, N) at (M, N) on the axes, and draws each rectilinear patch as a square with 1 unit in width and height.

imshow(I) displays the intensity image I with N discrete levels of gray. If the N is omitted, imshow uses 256 gray levels on 24-bit displays, or 64 gray levels on other systems. The imshow shows the image to its true size instead of square by default.

* Reference: Matlab help file

5. Function

Please see the appendix for detail. The function [x1, x2] = rootsGQE(coef) returns the roots (real or complex) of a general quadratic equation with one unknown, given the coefficients of the equation. The coefficients are in the form of [a, b, c], corresponding to quadratic equation $ax^2 + bx + c = 0$. The roots are saved in x1 and x2. When a = 0, the equation degrades to a linear equation. When a = b = 0, the function returns –Inf (stands for infinity) as roots. When a = b = c = 0, the function returns NaN (stands for Not a Number) as roots.

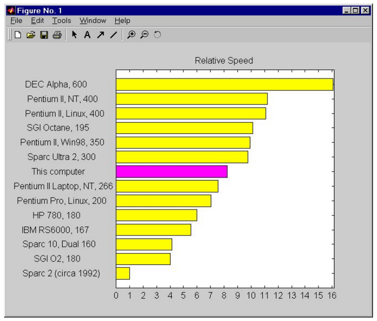
6. Advantage of vectorization

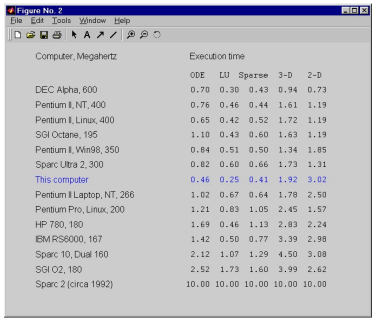
Please see the appendix for detail.

7. Finally a bit of fun

The goal of this problem was to show you that many things affect the speed of algorithm execution, and that different computers do different things better.

In addition, as this result shows, computers can perform unexpectedly badly due to the OS or some other reason





Computer, Megahertz	Execution time				
	ODE	LU	Sparse	3-D	2-D
DEC Alpha, 600	0.70	0.30	0.43	0.94	0.73
Pentium II, NT, 400	0.76	0.46	0.44	1.61	1.19
Pentium II, Linux, 400	0.65	0.42	0.52	1.72	1.19
SGI Octane, 195	1.10	0.43	0.60	1.63	1.19
Pentium II, Win98, 350	0.84	0.51	0.50	1.34	1.85
Sparc Ultra 2, 300	0.82	0.60	0.66	1.73	1.31
This computer	0.46	0.25	0.41	1.92	3.02
Pentium II Laptop, NT, 266	1.02	0.67	0.64	1.78	2.50
Pentium Pro, Linux, 200	1.21	0.83	1.05	2.45	1.57
HP 780, 180	1.69	0.46	1.13	2.83	2.24
IBM RS6000, 167	1.42	0.50	0.77	3.39	2.98
Sparc 10, Dual 160	2.12	1.07	1.29	4.50	3.08
SGI O2, 180	2.52	1.73	1.60	3.99	2.62
Sparc 2 (circa 1992)	10.00	10.00	10.00	10.00	10.00

The machine used is

CPU	: Pentium III 667MHz @ 133MHz bus
Storage	: 128MB @ 133MHz bus memory, 20G HD
OS	: Windows 98 second edition

Appendix: Scripts and examples[1].

1. Scripts and examples for question no. 1

Script:

```
function hw_1
%
% CMSC 828D: Fundamentals of Computer Vision
%                       Homework1
%
% Instructors : Larry Davis, Ramani Duraiswami,
%               Daniel DeMenthon, and Yiannis Aloimonos
% Student     : Haiying Liu
%
% Date        : Aug. 30, 2000
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%

%=============================================================================
=
%= Qustion no.1
%=
%= Excecute thefloowing commands and explain.

a  = rand(10, 1);
b  = rand(10, 1);

A  = rand(10, 10);

%r1 = a * b;
%disp(['a * b = ', num2str(r1)]);

r2 = a .* b;
disp(' ');
disp('a .* b = ');
disp(num2str(r2));

r3 = a / b;
disp(' ');
disp('a / b = ');
disp(num2str(r3));

r4 = a ./ b;
disp(' ');
disp('a ./ b = ');
disp(num2str(r4));

r5 = a * b';
disp(' ');
disp('a * b'' = ');
disp(num2str(r5));
```

[1] All scripts are written as functions to prevent them from interfering each other.

```
r6 = a \ A;
disp(' ');
disp('a \ A = ');
disp(num2str(r6));
```

Examples:

```
hw1_1

a .* b =
0.15265
0.13592
0.28568
0.73495
0.54993
0.67862
0.13004
0.51955
0.17066
0.39732

a / b =
0            0            0            0            0     0.66647            0
0            0            0
0            0            0            0            0     0.23206            0
0            0            0
0            0            0            0            0     0.80834            0
0            0            0
0            0            0            0            0       1.096            0
0            0            0
0            0            0            0            0     0.93441            0
0            0            0
0            0            0            0            0     0.94777            0
0            0            0
0            0            0            0            0     0.78182            0
0            0            0
0            0            0            0            0      1.0717            0
0            0            0
0            0            0            0            0     0.52683            0
0            0            0
0            0            0            0            0      1.1533            0
0            0            0

a ./ b =
 2.0835
0.28369
 1.6377
 1.1704
 1.1368
0.94777
 3.3655
 1.5827
 1.1645
  2.397
```

```
a * b' =
 0.15265     0.39036     0.23554      0.4469     0.39224      0.4772
0.11086     0.32311      0.2159      0.2296
0.053151     0.13592    0.082013     0.15561     0.13657     0.16616
0.038599      0.1125    0.075173    0.079946
 0.18514     0.47345     0.28568     0.54203     0.47573     0.57878
0.13445     0.39189     0.26185     0.27848
 0.25104     0.64196     0.38736     0.73495     0.64505     0.78478
0.18231     0.53137     0.35505     0.37759
 0.21402     0.54729     0.33023     0.62656     0.54993     0.66905
0.15542     0.45301     0.30269     0.32191
 0.21708     0.55512     0.33496     0.63552     0.55779     0.67862
0.15765     0.45949     0.30702     0.32651
 0.17907     0.45792     0.27631     0.52425     0.46013     0.55979
0.13004     0.37903     0.25326     0.26934
 0.24545     0.62767     0.37874     0.71859      0.6307     0.76732
0.17825     0.51955     0.34715     0.36919
 0.12067     0.30857     0.18619     0.35327     0.31006     0.37722
0.087629     0.25541     0.17066      0.1815
 0.26415     0.67549     0.40759     0.77334     0.67875     0.82577
0.19183     0.55913      0.3736     0.39732

a \ A =
0.48763     0.76342     0.51268     0.83627     0.54667     0.52898
0.57729     0.54086     0.78753      0.7749
diary off
```

2. Scripts and examples for question no. 2

Script:

```
function hw_2
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%===========================================================================
%= Qustion no.2
%=
%= Excecute thefloowing commands and explain.

a  = rand(10, 1);
b  = rand(10, 1);

r1 = 1:10;
disp(' ');
disp('1:10 = ');
disp(num2str(r1));

r2 = a(1:5);
disp(' ');
disp('a(1:5) = ');
disp(num2str(r2));

r3 = a(4:9) + b(1:6);
disp(' ');
disp('a(4:9) + b(1:6) = ');
disp(num2str(r3));

r4 = 0.1:0.1:100;
disp(' ');
disp('0.1:0.1:100 = ');
disp(num2str(r4));
```

Examples:

```
hw1_2

1:10 =
1  2  3  4  5  6  7  8  9 10

a(1:5) =
0.27946
0.88505
0.43355
0.37712
0.22666

a(4:9) + b(1:6) =
0.87988
0.49369
0.99302
0.97128
```

```
0.62111
 1.5469

0.1:0.1:100 =
0.1   0.2   0.3   ...  100
diary off
```

3. Scripts and examples for question no. 3

Script:

```
function hw_3
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%= Qustion no.3
%=
%= Excecute theflooing commands and explain.

a  = rand(10, 1)
b  = rand(10, 1)

bga = b > a
Ib  = find(bga)
b(Ib)
```

Example:

```
hw1_3

a =

    0.7463
    0.4629
    0.4730
    0.9297
    0.5494
    0.0268
    0.4832
    0.0426
    0.7744
    0.5816


b =

    0.0300
    0.8845
    0.6577
    0.9891
    0.7825
    0.6678
    0.6423
    0.2053
    0.7586
    0.9711


bga =

     0
     1
     1
     1
     1
     1
     1
     1
     0
     1


Ib =

     2
     3
     4
     5
     6
     7
     8
    10


ans =

    0.8845
    0.6577
    0.9891
    0.7825
    0.6678
    0.6423
    0.2053
    0.9711

diary off
```

4. Scripts and examples for question no. 4

Scripts:

```
function hw_4
%
% CMSC 828D: Fundamentals of Computer Vision
%                       Homework1
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%============================================================================
%= Qustion no.4
%=
%= Create a random gray level image I (0-255) of size 256x128 using the rand
%= command. Convert this image to type uint8. Display it using the commands
%= image(I)
%= imshow(I)
%= What diffeences do you see?

I = rand(258, 128) * 255;
I = uint8(I);

figure;
image(I);

figure;
imshow(I);
```

5. Scripts and examples for question no. 5

Script:

```
function [x1, x2] = rootsGQE(coefficients)
% Syntax [x1, x2] = rootsGQE(coefficients)
%
%        coefficients - coefficients of the general quadratic equation
%        x1, x2       - two roots of the equation
%
% Description: Compute the roots of a general quadratic equation
%              expressed by its coefficients
%
% Date        : Aug. 30, 2000
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%

%============================================================================
=
%= Qustion no.5
%=
%= Write a function that will return the roots of a general quadratic
equation
%= given the coefficients of the equation. Make your program as general as
you
%= can.

a = coefficients(1);
b = coefficients(2);
c = coefficients(3);

if a ~= 0
  temp = sqrt(b .* b - 4 * a * c);
  x1 = (-b + temp) / (a + a);
  x2 = (-b - temp) / (a + a);
else
  x1 = -c / b;
  x2 = x1;
end
```

Examples:

```
[x1, x2] = rootsGQE(rand(3, 1))

x1 =

   -0.1750


x2 =

   -0.8530
```

```
[x1, x2] = rootsGQE([9, 1, 3])

x1 =

  -0.0556 + 0.5747i


x2 =

  -0.0556 - 0.5747i

[x1, x2] = rootsGQE([0, 1, 3])

x1 =

    -3


x2 =

    -3

[x1, x2] = rootsGQE([0, 0, 3])
Warning: Divide by zero.
> In G:\Course\CMSC828D\rootsGQE.m at line 42

x1 =

  -Inf


x2 =

  -Inf

[x1, x2] = rootsGQE([0, 0, 0])
Warning: Divide by zero.
> In G:\Course\CMSC828D\rootsGQE.m at line 42

x1 =

   NaN


x2 =

   NaN

diary off
```

6. Scripts and examples for question no. 6

Scripts:

```
function ans = hw1_6
%
% Date        : Aug. 30, 2000
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%

%============================================================================
=
%= Qustion no.6
%=
%= For problem 3, write a function that explicitly uses for loops to achieve
%= the same result

a       = rand(10, 1)
b       = rand(10, 1)

nRow_a  = size(a);
nCol_b  = size(b);

if nRow_a ~= nCol_b
  error('The degree of a must be equal to b.');
end

idx_Ib  = 0;
for idx = 1:nRow_a
  if b(idx) > a(idx)
    bga(idx)   = 1;
    idx_Ib     = idx_Ib + 1;
    Ib(idx_Ib) = idx;
    bb(idx_Ib) = b(idx);
  end
end

bga = bga'
Ib  = Ib'
ans  = bb';
```

Example:

```
hw1_6

a =

    0.4102
    0.4401
    0.2975
    0.2978
    0.4444
    0.9955
```

```
    0.9377
    0.7363
    0.3665
    0.2783


b =

    0.6961
    0.5942
    0.1874
    0.1653
    0.1057
    0.0114
    0.0398
    0.8772
    0.5112
    0.6458


bga =

     1
     1
     0
     0
     0
     0
     0
     1
     1
     1


Ib =

     1
     2
     8
     9
    10


ans =

    0.6961
    0.5942
    0.8772
    0.5112
    0.6458

diary off
```