Sec. 6.1.15

% 6.1.15 Vectorizing nested fors: loan repayment tables

%% forming the table of repayments for a loan of $1000 over 15, 20 and 25 yrs

% rate% 15 yrs 20 yrs 25 yrs

% 10 10.75 9.65 9.09

% 11 11.37 10.32 9.80

% 12 12.00 11.01 10.53

% 13 12.65 11.72 11.28

%% Exercise to write the expression in p.140 by matlab code

% Method 1:

A = 1000; % amount borrowed

n = 12; % number of payments per year

disp ([' rate% 15 yrs 20 yrs 25 yrs']);

pay=zeros(11,3);

for r = 0.1 : 0.01 : 0.2

fprintf( '%4.0f%', 100 \* r );

for k = 15 : 5 : 25

temp = (1 + r/n) ^ (n\*k);

P = r \* A \* temp / (n \* (temp - 1));

pay(j,i)=P;

fprintf( '%10.2f', P );

end;

fprintf( '\n' ); % new line

end;

rate% 15 yrs 20 yrs 25 yrs

10 10.75 9.65 9.09

11 11.37 10.32 9.80

12 12.00 11.01 10.53

13 12.65 11.72 11.28

14 13.32 12.44 12.04

15 14.00 13.17 12.81

16 14.69 13.91 13.59

17 15.39 14.67 14.38

18 16.10 15.43 15.17

19 16.83 16.21 15.98

20 17.56 16.99 16.78

% to store the payment in a matrix:

A = 1000; % amount borrowed

n = 12; % number of payments per year

disp ([' rate% 15 yrs 20 yrs 25 yrs']);

i=1;j=1;

pay=zeros(11,3);

for r = 0.1 : 0.01 : 0.2

fprintf( '%4.0f%', 100 \* r );

i=1;

for k = 15 : 5 : 25

temp = (1 + r/n) ^ (n\*k);

P = r \* A \* temp / (n \* (temp - 1));

pay(j,i)=P;

fprintf( '%10.2f', P );

i=i+1;

end;

j=j+1;

fprintf( '\n' ); % new line

end;

% How to do it by point operation

r=0.1:0.01:0.2

r=r'

rate=repmat(r,[ 1 3])

y=15:5:25

year=repmat(y, [ 11 1])

% Method 3

A = 1000; % amount borrowed

n = 12; % number of payments per year

r = [0.1:0.01:0.2]' % r is a 11\*1 matrix

% Now change this into a table with 3 columns each equal to r:

r = repmat(r, [1 3]) % r is 11\*3 matrix

k = 15:5:25 % k is 3\*1 matrix

k = repmat(k, [11 1]) % k is a 11\*3 matrix

% r (11\*3) matrix =

%

% 0.1000 0.1000 0.1000

% 0.1100 0.1100 0.1100

% 0.1200 0.1200 0.1200

% k (11\*3) matrix =

%

% 15 20 25

% 15 20 25

% 15 20 25

% show the value of r & k

format short

disp ([' rate% 15 yrs 20 yrs 25 yrs']);

temp = (1 + r/n) .^ (n \* k);

P = r \* A .\* temp / n ./ (temp - 1);

disp([ 100\*r(:,1) P])

Sec. 6.4

% Leslie matrix population model

% Iterative processing : repeatly do the same operations.

% Leslie matrices

% prepare L

n=3;

L = zeros(n); % all elements set to zero

L(1,2) = 9;

L(1,3) = 12;

L(2,1) = 1/3;

L(3,2) = 0.5;

% Initial condition

x = [0 0 1]'; % remember x must be a column vector!

format bank

for t = 1:24

x = L \* x;

p(t) = sum(x); % the total population at time t

disp( [t x' sum(x)] ) % x¡¦ is a row

end

figure, plot(1:15, p(1:15)), xlabel('months'), ylabel('rabbits')

hold, plot(1:15, p(1:15),'o')

hold, plot(1:15, p(1:15),'o')

1.00 12.00 0 0 12.00

2.00 0 4.00 0 4.00

3.00 36.00 0 2.00 38.00

4.00 24.00 12.00 0 36.00

5.00 108.00 8.00 6.00 122.00

6.00 144.00 36.00 4.00 184.00

7.00 372.00 48.00 18.00 438.00

8.00 648.00 124.00 24.00 796.00

-----

%% (for) loop for fixed loop : how many times of the loop is fixed .

% (white) loop for the conditional loop: when the condition is satisfied

% the loop is continued.

% Sec. 8.1.2 : An exaple in p. 181 :Update processes

K = 0.05;

F = 10;

a = 0; % start time

b = 100; % end time

time = a; % initialize time

T = 25; % initialize temperature

load train % prepare to blow the whistle

dt=5;

opint=10;

% opint = input( 'output interval (minutes): ');

% if opint/dt ~= fix(opint/dt)

% sound(y, Fs) % blow the whistle!

% disp( 'output interval is not a multiple of dt!');

% break

% end

clc

format bank

disp( ' Time Temperature' );

disp( [time T] ) % display initial values

for time = a+dt : dt : b

T = T - K \* dt \* (T - F);

disp( [time T] )

end

end

% Question : when is the temperture of orange just below 15 ??

K = 0.05;

F = 10;

a = 0; % start time

b = 100; % end time

time = a; % initialize time

T = 25; % initialize temperature

load train % prepare to blow the whistle

dt=5;

opint=10;

% opint = input( 'output interval (minutes): ');

% if opint/dt ~= fix(opint/dt)

% sound(y, Fs) % blow the whistle!

% disp( 'output interval is not a multiple of dt!');

% break

% end

clc

format bank

disp( ' Time Temperature' );

disp( [time T] ) % display initial values

for time = a+dt : dt : b

T = T - K \* dt \* (T - F);

disp( [time T] )

if (T<=15)

break;

end

end

% we can also use the white loop for this problem

K = 0.05;

F = 10;

a = 0; % start time

b = 100; % end time

time = a; % initialize time

T = 25; % initialize temperature

load train % prepare to blow the whistle

dt=5;

opint=10;

% opint = input( 'output interval (minutes): ');

% if opint/dt ~= fix(opint/dt)

% sound(y, Fs) % blow the whistle!

% disp( 'output interval is not a multiple of dt!');

% break

% end

clc

format bank

disp( ' Time Temperature' );

disp( [time T] ) % display initial values

while ( (T>15) && (time<b) ) % the condition

time=time+dt;

T = T - K \* dt \* (T - F);

disp( [time T] );

end

% 8.2.5 Projectile trajectory

% 8.2.5 Projectile trajectory

dt = 0.1;

g = 9.8;

u = 60;

ang = input( 'Launch angle in degrees: ' );

ang = ang \* pi / 180; % convert to radians

x = 0; y = 0; t = 0; % for starters

more(15)

while y >= 0

disp( [t x y] );

t = t + dt;

y = u \* sin(ang) \* t - g \* t^2 / 2;

x = u \* cos(ang) \* t;

end

%% store the data and then plot it as shown in Fig. 8.1 in p.189

clear all;

close all;

dt = 0.1;

g = 9.8;

u = 60;

ang = input( 'Launch angle in degrees: ' );

ang = ang \* pi / 180; % convert to radians

x(1) = 0; y(1) = 0; t(1) = 0; % for starters

more(15)

i=1;

while y (i) >= 0

i=i+1;

t(i)= t(i-1) + dt;

y (i) = u \* sin(ang) \* t(i) - g \* t(i)^2 / 2;

x (i) = u \* cos(ang) \* t(i);

end