البرمجة الإجرائية

Lecture No. 11

Applications

ميكاترونيك-سنة أولى-فصل أول

Dr. Eng. Essa Alghannam
Ph.D. Degree in Mechatronics Engineering

2024

ind = sub2ind(sz , row , col)

returns the **linear indices** ind corresponding to the **row** and **column** subscripts in **row** and **col** for a **matrix** of size sz .

```
>> ind = sub2ind( [6 7] , 1 , 2 )
ind =
    7

>> [I,J] = ind2sub([6 7],5)
I =
    5
J =
    1
```

[I,J] = ind2sub(siz,IND)

returns the matrices I and J containing the equivalent row and column subscripts corresponding to each linear index in the matrix IND for a matrix of size siz.

- siz is a 2-element vector

```
>> S=[ 1 2 3 0 5 6;7 8 9 10 0 12; 13 14 0 16 17 18; 0 20 21 0 23 24; 0 26 0 28 29 30]


S =

    1    2    3    0    5    6
    7    8    9   10    0   12
   13   14    0   16   17   18
    0   20   21    0   23   24
    0   26    0   28   29   30
```

```
>> [row,col] = ind2sub(size(S),find(S>26))

row =

    5
    5
    5

col =

    4
    5
    6
```

5	4
5	5
5	6

```
>> ind = sub2ind( size(S),find(S>26) )

ind =

   20
   25
   30
```

تطبيق

اكتب تابع يحول عنوان عنصر من مصفوفة من فهرسة خطية إلى فهرسة سطر عمود و اكتب تابع اخر يقوم بالعكس

```
function index=rowcolumn_to_index(A,row,column)
s=size(A);
if row>s(1) || column >s(2) || row<=0 || column <=0
   index=[];
else
   index=s(1)*(column-1)+row;
end
end
```

```
>> S=[ 1 2 3 0 5 6;7 8 9 10 0 12; 13 14 0 16 17 18; 0 20 21 0 23
24; 0 26 0 28 29 30]

S =

    1    2    3    0    5    6
    7    8    9   10    0   12
   13   14    0   16   17   18
    0   20   21    0   23   24
    0   26    0   28   29   30
```

```
>> rowcolumn_to_index(S,0,0)

ans =

    []

>> rowcolumn_to_index(S,-1,1)

ans =

    []
```

```
>> rowcolumn_to_index(S,5,7)

ans =

    []

>> rowcolumn_to_index(S,5,4)

ans =

    20
```

```
function [row,column]=index_to_rowcolumn(A,index)

s=size(A);
if index > s(1)*s(2) || index<=0
   column=[];row=[];
else
   column=ceil(index/s(1));
   row=index-s(1)*(column-1);
end
end
```

```
>> [row,column]=index_to_rowcolumn(S,31)

row =

    []


column =

    []
```

```
>> [row,column]=index_to_rowcolumn(S,-1)

row =

    []


column =

    []
```

```
>> [row,column]=index_to_rowcolumn(S,26)

row =

     1


column =

     6
```

تطبيق: إيجاد عاملي عدد صحيح

```
function y=myfactor(x)
if ~x
   y=1;
elseif x<0
   error('error! negative number')
else
   y=1;
   for i=x:-1:1
      y=y*i;
   end
end
end
```

```
function y=mytranspose(x)

s1=size(x,1);
s2=size(x,2);

for i=1:s1
   for j=1:s2
      y(i,j)=x(j,i);
   end
end
```

إيجاد منقولة مصفوفة مربعة من دون استخدام توابع جاهزة

square matrix

Function for maximum number :

```
function max=mymax(a,b,c)
if a>b
   max=a;
   if c>max
      max=c;
   end
else
   max=b;
   if c>max
      max=c;
   end
end
end
```

Sorting Vector :

```
function vn=mysort(v)
s=length(v);
vn=v;
for i=1:s-1
   for j=i+1:s
      if vn(i)>vn(j)
         x=vn(i);
         vn(i)=vn(j);
         vn(j)=x;
      end
   end
end
disp(vn)
end
```

إيجاد معين مصفوفة مربعة من دون استخدام توابع جاهزة

```
function y=mydelta(x)

s1=size(x,1);
s2=size(x,2);

if s1~=s2
   error('not square matrix')
elseif s1==2 && s2==2
   y=x(1,1)*x(2,2)-x(1,2)*x(2,1);
else
  y=0;
  for c=1:s1
       y=y+(-1)^(1+c)*x(c,1)*mydelta(x([1:c-1,c+1:s1],2:s2));
  end
end
end
```

Recursion

فكرة العودية

اكتب تابع لتحويل عدد عشري الى ثنائي:

```
function y = d2b(x,m)
%% Notes
% x can be columns or rows vector, must be in positive integer
% m length of output binary array

%% Biggest x   x=10
bigX=max(x);   %%10

%% Number of X to be converted
n=length(x);   %%1
c = max(0,floor(log2(bigX))) + 1;
```

```
%% Output Array
if nargin<2 % output array length is not defined by user
   % nargin Number of function input arguments.
   m=c;
elseif m<c % output array is defined by user but smaller than required length
   m=c;
   disp('Determined Binary Size Is Smaller Than Output Binary Size')
end
```

```
y = zeros(n,m); % Initialize output array
%% Compute y
shift = m-c;
for i =1:n
   for j = 1:c
      r = floor(x(i) / 2);
      y(i,shift+c+1-j) = x(i) - 2*r;
      x(i) = r;
   end
end
```

```
>> d2b([4 5 6])

ans =

     1     0     0
     1     0     1
     1     1     0
```

```
>> d2b([4 5 20])

ans =

     0     0     1     0     0
     0     0     1     0     1
     1     0     1     0     0
```

```
>> d2b([4 5 20],3)
Determined Binary Size Is Smaller Than Output Binary Size
ans =
     0     0     1     0     0
     0     0     1     0     1
     1     0     1     0     0

>> d2b([4 5 20],7)
ans =
     0     0     0     0     1     0     0
     0     0     0     0     1     0     1
     0     0     1     0     1     0     0
```

```
function y = b2d(x)
%% Notes
% x in n,m matrix. n is the amount of binary and m is the length of the
binary

%% Better Performance using polyval, suggestion by Matt D
y=zeros(size(x,1),1);
for i=1:size(x,1)
   y(i,1)=polyval(x(i,:),2);
end
```

```
>> y=b2d([1 0 1 0; 0 1 0 1])

y =

    10
     5
```

```
>> polyval([5 4 3],2)
ans =
    31
```

3*2^0 + 4*2 + 5*2^2=3+8+20

collatz

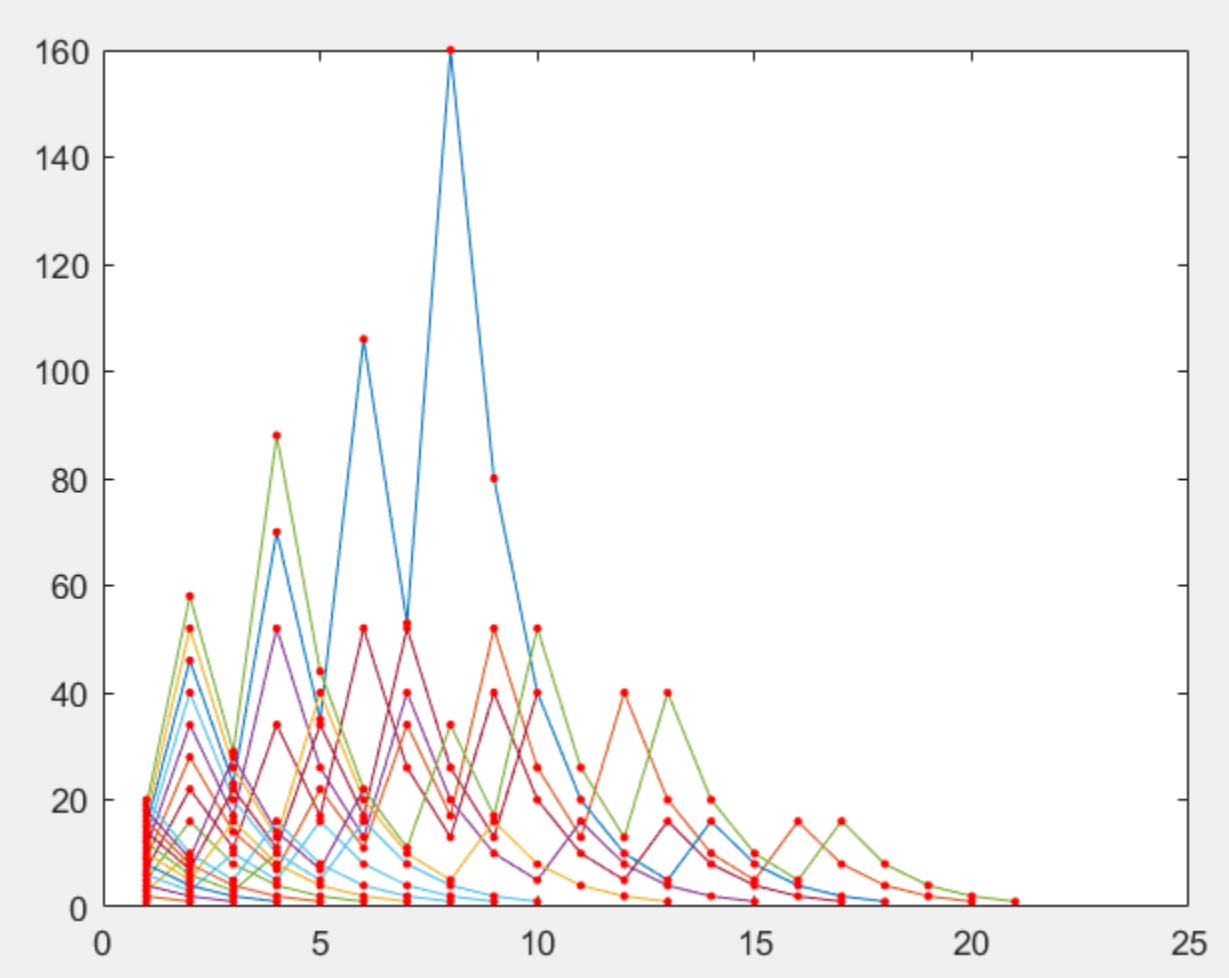
```
function [ns, seq] = collatz(n)
% Your first number in the sequence is n
seq(1) = n;
% You position an index on the next element of the sequence
i = 2;
% Repeat the iteration until you find a 1
while seq(i-1) ~= 1
    % Use a modulus after division to find an even/odd number
    if mod(seq(i-1), 2) == 0
        % Step taken if even
        seq(i) = seq(i-1)/2;
    else
        % Step taken if odd
        seq(i) = 3*seq(i-1) + 1;
    end
    % Increment your index
    i = i+1;
end
% Find the length of the sequence
ns = length(seq);
disp(seq)
% Your first number in the sequence is n

% plot(1:ns,seq)
% hold on
% for i=1:ns
%
plot(i,seq(i),'or','MarkerSize',2,'MarkerFaceColor','r')
% %     axis([1 150 1 2000])
%     pause(.1)
% end
%
% end
```

```
clc; clear
% We're going to start the sequences with integers up to 20
for i = 1 : 20
    % We perform the function and get the number of elements
    % in the sequence and the sequence itself
    [n(i), seq{i}] = collatz(i);
end
```

```
% We find out what's the maximum number of steps found
[max_steps, ix] = max(n)
% We display the appropriate indexed sequence
seq{ix}
```

And we get:

```
max_steps = 21
ix = 18
ans = 18    9   28   14    7   22   11   34   17
      52   26   13   40   20   10    5   16    8
       4    2    1
```


160
140
120
100
80
60
40
20
0
0
5
10
15
20
25

جَامعة
المَنارة
MANARA UNIVERSITY

Thanks .