18.11.2017

MATLAB_Project

Implementation of an algorithm for Modified Node Analysis(MNA) in MATLAB

Sefa Kayraklık
2015401066

I have neither given nor received any unauthorized aid on this assignment.

Description of the Project:

In the project, it is asked that given a resistive network with independent ideal voltage sources, independent ideal current sources, and ohmic resistors should be analyzed via node voltage method, and the nodes' voltages should be found. In such a problem, there are a lot of algorithm to analyze a circuit that consists of resistors, independent ideal current and voltage sources. To implement such algorithm, MATLAB is required to be used, and the MATLAB-base program reads the text file, which includes the types, the negative polarity nodes, the positive polarity nodes, and values in the unit of volt, ampere, and ohm, respectively in each column, from the user. Then, the program displays the text file name and the voltages of each node.

Explanation of the Modified Node Analysis(MNA) [1]:

For a circuit with *n* nodes and *m* independent voltage sources:[1]

- The **A** matrix:
  - is (*n+m*)x(*n+m*) in size, and consists only of known quantities.
  - the *n*x*n* part of the matrix in the upper left:
    - has only passive elements
    - elements connected to ground appear only on the diagonal
    - elements not connected to ground are both on the diagonal and off-diagonal terms.
  - the rest of the **A** matrix (not included in the *n*x*n* upper left part) contains only 1, -1 and 0 (other values are possible if there are dependent current and voltage sources; I have not considered these cases. Consult Litovski* if interested.)

*http://www.swarthmore.edu/NatSci/echeeve1/Ref/mna/MNARefs.html#Litovski

- The **x** matrix:
    - is an (*n+m*)x1 vector that holds the unknown quantities (node voltages and the currents through the independent voltage sources).
    - the top *n* elements are the *n* node voltages.
    - the bottom *m* elements represent the currents through the *m* independent voltage sources in the circuit.
- The **z** matrix:
    - is an (*n+m*)x1 vector that holds only known quantities
    - the top *n* elements are either zero or the sum and difference of independent current sources in the circuit.
    - the bottom *m* elements represent the *m* independent voltage sources in the circuit.

The circuit is solved by a simple matrix manipulation.

For further information, please check the source in the reference.

Implementation of The MNA in MATLAB

The program has one script named MainScript, and two functions named CalculateVoltages, disp_NV. The implementation of these is the following:

- *MainScript:*

    It consists of 3 lines of codes that take the name of text file from the user and send it to the function CalculateVolatages, and then get the solution and send the solution and the name of the file to the function disp_NV to display the results and the name of the file.

- *CalculateVoltages:*

    The function takes the name of the file and reads the data from it and implements the MNA and returns the nodes' voltages.

    - It begins with the determining the numbers of each type of components (V, I, and R) to be able to define the cell arrays of the matrices' rows number. Each cell array includes 2 matrices which the 1st one is the value of it and the 2nd one is the nodes of it and each row has the corresponding value and nodes of the given component. After determining the numbers, it fills these arrays line by line.

- After filling the cell arrays of each corresponding type, it determines n (# of nodes) and m (# of independent voltage sources) in MNA.
- Constructing the G matrix in the A matrix: The 2 for loops scan each element of G and the diagonal elements take the sum of the conductance of the resistors which are connected to the corresponding node, and the off-diagonal elements take the sum of the negative conductance of the resistors which are connected between the corresponding nodes.
- Constructing the B matrix in the A matrix: Its columns consist of the positive and negative polarity of the independent voltage sources. The positive node of V corresponds 1 at the column's row, the negative node of V corresponds -1 at the column's row, and other elements are 0.
- Constructing the C matrix in the A matrix: It is the transpose of the B matrix.
- Constructing the D matrix in the A matrix: It is m by m zero matrix for the circuit that doesn't have dependent sources.
- Combining the submatrices of the A: It is considered as following: G matrix is nxn, the B matrix is nxm, the C matrix is mxn, and the D matrix is mxm $\mathbf{A} = \begin{bmatrix} \mathbf{G} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix}$
- Constructing the z matrix: It is a column vector, and its first n columns are the sum of independent current sources with the corresponding nodes, and the last m is the values of the independent voltage sources.
- The solution(x) is the form of following: The first n of the x is the node voltages of the circuit. The function returns it.

$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{z} \qquad \mathbf{A}\mathbf{x} = \mathbf{z}$$

- *Disp_NV:*
  - The function displays the text file name without its extension, firstly. Then it displays the node voltages of the corresponding nodes line by line.

REFERENCES:

[1] http://www.swarthmore.edu/NatSci/echeeve1/Ref/mna/MNA3.html

The codes line in MATLAB:

MainScript.m:

```
textName=input('Please enter the of the text file with its expention: ',
's');
soln=CalculateVoltages(textName);
disp_NV(soln, textName);
```

CalculateVoltages.m:

```
function [ solution ] = CalculateVoltages( textName )
%The function takes the name of text file with its extention that consists
%of 4 columns. The 1st column includes the type of components(V, I, or R).
%Remining two ones descripe the nodes between which the corresponding
%component is connected.(1st is negative, 2nd is positive node) And the
%last column is the values of the components in the unit of volt, ampere,
%and ohm. Then it returns the correspondin node voltages, respectively.
fid=fopen(textName, 'r');
if fid==-1
    disp('Something goes wrong! The file cannot be opened.');
end
r=0; v=0; i=0;
while feof(fid)==0%determines # of R, V, and I.
    aline=fgetl(fid);%reads the txt file line by line.
    [txt, num]=strtok(aline);%1st entry is the type.
    if txt(1)=='R'%determines the type
        r=r+1;%increases the # of R.
    end
    if txt(1)=='V'%determines the type
        v=v+1;%increases the # of V.
    end
    if txt(1)=='I'%determines the type
        i=i+1;%increases the # of I.
    end
end
fid=fclose(fid);
if fid==-1
    disp('Something goes wrong! The file cannot be closed.');
end
fid=fopen(textName, 'r');%to be able to use fgetl in while loop, again
opens the text file.
if fid==-1
    disp('Something goes wrong! The file cannot be opened.');
end

resistors={zeros(r,1), zeros(r,2)};%1st field of cell array is the value of
R, and 2nd field is the nodes of R.
vsources={zeros(v,1), zeros(v,2)};%1st field of cell array is the value of
V, and 2nd field is the nodes of V.
isources={zeros(i,1), zeros(i,2)};%1st field of cell array is the value of
I, and 2nd field is the nodes of I.

r=0; v=0; i=0; %resets the indexs of R, V, and I.
while feof(fid)==0
    aline=fgetl(fid);%reads the txt file line by line.
    [txt, num]=strtok(aline);%1st entry is the type, 2nd is the nodes and
the values.
    num=str2num(num);%converts string 'num' matrix to a integer 'num'
matrix.
    if txt(1)=='R'%determines the type
        r=r+1;%increases the # of R.
        resistors{1}(r)=num(3);%value of rth R.
```

```
        resistors{2}(r,:)=[num(1) num(2)];%nodes of rth R.
    end
    if txt(1)=='V'%determines the type
        v=v+1;%increases the # of V.
        vsources{1}(v)=num(3);%value of vth V.
        vsources{2}(v,:)=[num(1) num(2)];%nodes of vth V.
    end
    if txt(1)=='I'%determines the type
        i=i+1;%increases the # of I.
        isources{1}(i)=num(3);%value of ith I.
        isources{2}(i,:)=[num(1) num(2)];%nodes of ith I.
    end
end
fid=fclose(fid);
if fid==-1
    disp('Something goes wrong! The file cannot be closed.');
end
%The procedure of the MNA from
'http://www.swarthmore.edu/NatSci/echeeve1/Ref/mna/MNA3.html'
m=v;%# of independent voltage sources.
max_r=max(max(resistors{2}));
max_v=max(max(vsources{2}));
max_i=max(max(isources{2}));
n=max([max_r max_v max_i]);% determines # of nodes

G=zeros(n);
for a = 1:n
    for b = 1:n
        if a==b%at diagonal of G, the value is sum of the conductance in
the corresponding node.
            for c=1:r
                if any(resistors{2}(c,:)==a)%searchs each R whether it is
connected to the node a, or not.
                G(a,a)=G(a,a)+1/resistors{1}(c);%if true it adds
conductances.
                end
            end
        else%at off-diagonal, the value is the -1 times conductances.
            for c=1:r
                if resistors{2}(c,1)==a && resistors{2}(c,2)==b%Rs that
connected node a to b.G(a,b)
                    G(a,b)=G(a,b)-1/resistors{1}(c);
                end
                if resistors{2}(c,1)==b && resistors{2}(c,2)==a%Rs that
connected node b to a.G(b,a)
                    G(a,b)=G(a,b)-1/resistors{1}(c);
                end
            end
        end
    end
end

B=zeros(n,m);
for a=1:m%columns of B correspond each V, the one has positive polarity is
1, negative polarity is -1, otherwise 0.
     if vsources{2}(a,2)>0%checks nonzero of the indexing.
         B(vsources{2}(a,2),a)=1;%positive polarity corresponds 1.
     end
     if vsources{2}(a,1)>0%checks nonzero of the indexing.
         B(vsources{2}(a,1),a)=-1;%negative polarity corresponds -1.
     end
end
```

```
C=transpose(B);%If there is no dependent source(in the project thereis no),
C equals to B^T.

D=zeros(m,m);%If there is no dependent source(in the project thereis no), D
eqauls m-by-m zero matrix.

A=zeros(n+m,n+m);%construct the A matrix.

for a=1:n
    for b=1:n
        A(a,b)=G(a,b);%first n by n submatrix of A equals to G.
    end
end
for a=1:n
    for b=1+n:n+m
        A(a,b)=B(a,b-n);%the right top of A equals to B.
    end
end
for a=1+n:n+m
    for b=1:n
        A(a,b)=C(a-n,b);%the left buttom of A equals to C.
    end
end

z=zeros(n+m,1);%first n elements reveal the sum of I at the node, last m
elements reveal the value of V.

for a=1:i
    if isources{2}(a,2)>0%checks nonzero of the indexing.
        z(isources{2}(a,2))=z(isources{2}(a,2))+isources{1}(a,1);%positive
polarity corresponds adding the Is' values.
    end
    if isources{2}(a,1)>0%checks nonzero of the indexing.
        z(isources{2}(a,1))=z(isources{2}(a,1))-isources{1}(a,1);%negative
polarity corresponds adding the -Is' values.
    end
end
for a=1+n:n+m
    z(a)=vsources{1}(a-n);
end

x=(A^-1)*z;%computes the unknowns variables.

solution=zeros(n,1);
for i=1:n%first n elements correspond the node voltages.
    solution(i)=x(i);
end
end
```

disp_NV.m:

```
function disp_NV( soln, name )
%It displays the given node voltages as a matrix as a kind of the users can
%understand, and the name of the text file.
for i=1:4
    name(end)=[];
end
fprintf('The name of the file which is executed is %s.\n', name);
N=length(soln);
for i=1:N
    fprintf("The node%d's voltage value is %.4f. \n", i, soln(i));
end
end
```