DETERMINATION OF ADMITTANCE AND IMPEDANCE BUS MATRICES USING LINEAR ALGEBRA AND MATLAB ™ IN ELECTRIC POWER SYSTEMS

Bruno Osorno

California Sate University Northridge
18111 Nordhoff St
Northridge CA 91330
Email: bruno@ecs.csun.edu
Phone: (818)677-3956

Abstract
In electrical engineering, specifically in Electric Power Systems linear algebra is used widely throughout the curriculum. More specifically in electric power systems, short circuit analysis requires the determination of impedance bus matrices Z_{bus}. Admittance bus matrices, Y_{bus}, are used in load flow analysis amongst other applications. The classic manipulation of these matrices has been done using high-level programs such as FORTAN and C. With the introduction of mathematical packages such as MATLAB and MATHCAD we have reduced the time being consumed in the solution of such matrices and focus more in the techniques and analysis of the results. This paper will attempt to demonstrate the usage of MATLAB in the solution of linear algebra problems in the analysis of electric power systems specifically in the use of
the "building algorithm" and Kron's reduction.

Kron's[1] reduction (Node Elimination)
The size of a real Y_{bus}, admittance matrix, is very large. Computational time can be a problem, therefore, we needed to come up with algorithms to reduce the size of such matrix. The selection of the buses to be eliminated (in order to reduce the size of the matrix) is usually determined by the fact that there is no current being injected and /or the bus is of no importance to the analysis. *As a rule if there is no external load and/or there are not generating sources connected then we can eliminate such bus.*

When we want to eliminate a bus, we use the method of Kron's reduction. We simply identify the buses that are not active, or do not have an effect on the system. In our example it would be buses 5 and 6. The size of our matrix is a 6 by 6 (nxn).
Kron's reduction method is given as:

$$Y_{jk(new)} = Y_{jk} - \frac{Y_{jp} Y_{pk}}{Y_{pp}}$$

[1] Named after Dr. Gabriel Kron. 1901-1968. General Electric

"*p*" is the bus number to be eliminated.
"*j*" row
"*k*" column
j and *k* take values from 1 to n

Therefore Y_{bus} (new) has a new dimension (n-1)(n-1).

Step-by-step analysis (This is the "building algorithm)
The circuit shown below, figure 1 [1], is called a "reactance diagram." We use the building algorithm to obtain the Y_{bus} matrix that represents the diagram. In our class we teach this algorithm step-by-step with an example. We obtain Z_{bus} first then by inverting it we find Y_{bus}. Be aware that the "building algorithm" is a sequence of steps to create Z_{bus}. In this paper we show step by step such procedure.

The diagram below represents a typical electric system [6]. Suffice to say that each inductor, called reactance" represents a model for a specific electric component. For example the reactance j0.20 and the voltage source indicated on the left both represent a synchronous generator.

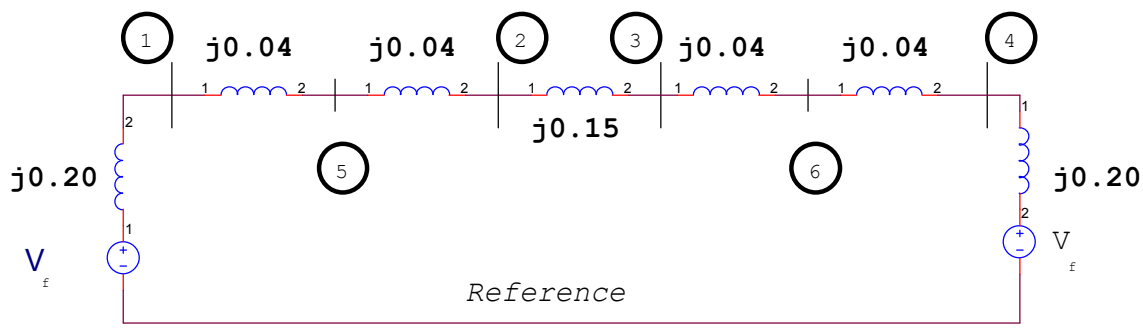

Figure 1. "Reactance diagram." Numbers inside circles are buses.

<u>*Building algorithm. STEP 1*</u>
Add branch 1 to reference node (*One bus has been created then we have a 1 by one size matrix)*

$$[j0.20]$$

Matlab Code [5]:

```
z=[0.20i];
```

<u>*Building algorithm. STEP 2*</u>
Add branch between bus 1 and 5 *(we are creating another bus, therefore the size of our matrix is 2 by 2)* So, every time we create a new bus we increase the size of the matrix by one.

$$\begin{bmatrix} j0.20 & j0.20 \\ j0.20 & j0.24 \end{bmatrix}$$

Matlab Code:

```
size=2;
zb=0.04i;
z(size,size)=z(size-1,size-1)+zb;
for i=1:(size-1);
z(size-i,size)=z(size-i,size-1);
z(size,size-i)=z(size-1,size-i);
end;
```

<u>*Building algorithm. STEP 3*</u>
Add branch between buses 5 and 2 (*a third bus is being added, consequently the size of our matrix is 3 by 3)*

$$\begin{bmatrix} j0.20 & j0.20 & j0.20 \\ j0.20 & j0.24 & j0.24 \\ j0.20 & j0.24 & j0.28 \end{bmatrix}$$

Matlab Code:

```
size=3;
zb=0.04i;
z(size,size)=z(size-1,size-1)+zb;
for i=1:(size-1);
z(size-i,size)=z(size-i,size-1);
z(size,size-i)=z(size-1,size-i);
end;
```

<u>*Building algorithm. STEP 4*</u>
Add branch between buses 2 and 3 (*a fourth bus is added and the size of our matrix is 4 by 4)*

$$\begin{bmatrix} j0.20 & j0.20 & j0.20 & j0.20 \\ j0.20 & j0.24 & j0.24 & j0.24 \\ j0.20 & j0.24 & j0.28 & j0.28 \\ J0.20 & j0.24 & j0.28 & j0.43 \end{bmatrix}$$

Matlab Code:

```
size=4;
zb=0.15i;
z(size,size)=z(size-1,size-1)+zb;
for i=1:(size-1);
z(size-i,size)=z(size-i,size-1);
z(size,size-i)=z(size-1,size-i);
end;
```

Building algorithm. STEP 5

Add branch between buses 3 and 6 *(a new bus is added and now we have a 5 by 5 matrix)*

$$\begin{bmatrix} j0.20 & j0.20 & j0.20 & j0.20 & j0.20 \\ j0.20 & j0.24 & j0.24 & j0.24 & j0.24 \\ j0.20 & j0.24 & j0.28 & j0.28 & j0.28 \\ j0.20 & j0.24 & j0.28 & j0.43 & j0.43 \\ j0.20 & j0.24 & j0.28 & j0.43 & j0.47 \end{bmatrix}$$

Matlab Code:

```
size=5;
zb=0.04i;
z(size,size)=z(size-1,size-1)+zb;
for i=1:(size-1);
z(size-i,size)=z(size-i,size-1);
z(size,size-i)=z(size-1,size-i);
end;
```

Building algorithm. STEP 6

Add branch between buses 6 and 4 *(A new bus is added and we have a 6 by 6 matrix)*

$$\begin{bmatrix} j0.20 & j0.20 & j0.20 & j0.20 & j0.20 & j0.20 \\ j0.20 & j0.24 & j0.24 & j0.24 & j0.24 & j0.24 \\ j0.20 & j0.24 & j0.28 & j0.28 & j0.28 & j0.28 \\ j0.20 & j0.24 & j0.28 & j0.43 & j0.43 & j0.43 \\ j0.20 & j0.24 & j0.28 & j0.43 & j0.47 & j0.47 \\ j0.20 & j0.24 & j0.28 & j0.43 & j0.47 & j0.51 \end{bmatrix}$$

Matlab Code:

```
size=6;
zb=0.04i;
z(size,size)=z(size-1,size-1)+zb;
for i=1:(size-1);
z(size-i,size)=z(size-i,size-1);
z(size,size-i)=z(size-1,size-i)
end;
```

Building algorithm. STEP7

Add branch from existing bus 4 to existing reference node. In this case we need to reduce the matrix to a 6 by 6 size using the method of Kron's reduction. We need to reduce the 7th column and row.

$$\left[\begin{array}{cccccc|c} j0.20 & j0.20 & j0.20 & j0.20 & j0.20 & j0.20 & j0.20 \\ j0.20 & j0.24 & j0.24 & j0.24 & j0.24 & j0.24 & j0.24 \\ j0.20 & j0.24 & j0.28 & j0.28 & j0.28 & j0.28 & j0.28 \\ j0.20 & j0.24 & j0.28 & j0.43 & j0.43 & j0.43 & j0.43 \\ j0.20 & j0.24 & j0.28 & j0.43 & j0.47 & j0.47 & j0.47 \\ j0.20 & j0.24 & j0.28 & j0.43 & j0.47 & j0.51 & j0.51 \\ \hline j0.20 & j0.24 & j0.28 & j0.43 & j0.47 & j0.51 & j0.71 \end{array}\right]$$

Matlab code to obtain the 7-by-7matrix

```
size=7;
zb=0.20i;
z(size,size)=z(size-1,size-1)+zb;
for i=1:(size-1);
z(size-i,size)=z(size-i,size-1);
z(size,size-i)=z(size-1,size-i);
end;
```

Matlab code to reduce a 7-by-7 to a 6-by-6 matrix

```
size=7; % current matrix size
remove=7; % remove= which row/column to remove
for jj=1:(size)
   if jj ~= remove;
      j=jj;
      if jj >= remove;
         j=j j-1;
      end
      for kk=1:(size);
         if kk ~= remove;
            k=kk;
            if kk >= remove;
               k=kk -1;
            end;
  znew(j,k)=z(jj,kk)-z(jj,remove)*z(remove,kk)/z(remove,remove);
         end;
      end;
   end;
end;
```

The first term (a_{11}) is obtained by hand to show how the method of Kron's reduction works:

$$Z_{11} = Z_{11} - \frac{Z_{17}Z_{71}}{Z_{77}} = j0.20 - \frac{(j0.20)(j0.20)}{j0.71} = j0.14366\ \Omega_{pu}$$

Where "pu" [4] is per-unit. (Basically it is a percentage)

Output matrix reduced to a 6 by 6 size.
Columns 1 through 4

Z_{11}

0 + 0.1437i	0 + 0.1324i	0 + 0.1211i	0 + 0.0789i
0 + 0.1324i	0 + 0.1589i	0 + 0.1454i	0 + 0.0946i
0 + 0.1211i	0 + 0.1454i	0 + 0.1696i	0 + 0.1104i
0 + 0.0789i	0 + 0.0946i	0 + 0.1104i	0 + 0.1696i
0 + 0.0676i	0 + 0.0811i	0 + 0.0946i	0 + 0.1454i
0 + 0.0563i	0 + 0.0676i	0 + 0.0789i	0 + 0.1211i

Columns 5 through 6

0 + 0.0676i	0 + 0.0563i
0 + 0.0811i	0 + 0.0676i
0 + 0.0946i	0 + 0.0789i

0 + 0.1454i	0 + 0.1211i
0 + 0.1589i	0 + 0.1324i
0 + 0.1324i	0 + 0.1437i

Building algorithm. STEP 8

Rows and columns 5 and 6 can be eliminated since we assume that there are no loads and/or any current injection.

Matlab program

First we are going to write the program that will obtain the complete matrix without any reduction. It does match the step-by-step sequence shown in the previous section. The outputs match exactly the matrices obtained before.

```
%For Zbus
% Add branch 1 to reference node
z=[0.20i];
za=z
% Add branch 2 between buses 1 to 5
size=2;
zb=0.04i;
z(size,size)=z(size-1,size-1)+zb;
for i=1:(size-1);
z(size-i,size)=z(size-i,size-1);
z(size,size-i)=z(size-1,size-i);
end;
zb=z
% Add branch 3 between buses 5 to 2
size=3;
zb=0.04i;
z(size,size)=z(size-1,size-1)+zb;
for i=1:(size-1);
z(size-i,size)=z(size-i,size-1);
z(size,size-i)=z(size-1,size-i);
end;
zc=z
% Add branch 3 between buses 2 and 3
size=4;
zb=0.15i;
z(size,size)=z(size-1,size-1)+zb;
for i=1:(size-1);
z(size-i,size)=z(size-i,size-1);
z(size,size-i)=z(size-1,size-i);
end;
zd=z
% Add branch 4 between buses 3 and 6
```

```
size=5;
zb=0.04i;
z(size,size)=z(size-1,size-1)+zb;
for i=1:(size-1);
z(size-i,size)=z(size-i,size-1);
z(size,size-i)=z(size-1,size-i);
end;
ze=z
% Add branch 4 between buses 6 and 4
size=6;
zb=0.04i;
z(size,size)=z(size-1,size-1)+zb;
for i=1:(size-1);
z(size-i,size)=z(size-i,size-1);
z(size,size-i)=z(size-1,size-i);
end;
zf=z
% From branch 5 from 4 to reference
size=7;
zb=0.20i;
z(size,size)=z(size-1,size-1)+zb;
for i=1:(size-1);
z(size-i,size)=z(size-i,size-1);
z(size,size-i)=z(size-1,size-i);
end;
zg=z
za =
      0 + 0.2000i
zb =
      0 + 0.2000i      0 + 0.2000i
      0 + 0.2000i      0 + 0.2400i
zc =
      0 + 0.2000i      0 + 0.2000i      0 + 0.2000i
      0 + 0.2000i      0 + 0.2400i      0 + 0.2400i
      0 + 0.2000i      0 + 0.2400i      0 + 0.2800i
zd =
      0 + 0.2000i      0 + 0.2000i      0 + 0.2000i      0 + 0.2000i
      0 + 0.2000i      0 + 0.2400i      0 + 0.2400i      0 + 0.2400i
      0 + 0.2000i      0 + 0.2400i      0 + 0.2800i      0 + 0.2800i
      0 + 0.2000i      0 + 0.2400i      0 + 0.2800i      0 + 0.4300i
ze =
  Columns 1 through 4
      0 + 0.2000i      0 + 0.2000i      0 + 0.2000i      0 + 0.2000i
      0 + 0.2000i      0 + 0.2400i      0 + 0.2400i      0 + 0.2400i
      0 + 0.2000i      0 + 0.2400i      0 + 0.2800i      0 + 0.2800i
      0 + 0.2000i      0 + 0.2400i      0 + 0.2800i      0 + 0.4300i
```

```
    0 + 0.2000i    0 + 0.2400i    0 + 0.2800i    0 + 0.4300i
  Column 5
    0 + 0.2000i
    0 + 0.2400i
    0 + 0.2800i
    0 + 0.4300i
    0 + 0.4700i
zf =
  Columns 1 through 4
    0 + 0.2000i    0 + 0.2000i    0 + 0.2000i    0 + 0.2000i
    0 + 0.2000i    0 + 0.2400i    0 + 0.2400i    0 + 0.2400i
    0 + 0.2000i    0 + 0.2400i    0 + 0.2800i    0 + 0.2800i
    0 + 0.2000i    0 + 0.2400i    0 + 0.2800i    0 + 0.4300i
    0 + 0.2000i    0 + 0.2400i    0 + 0.2800i    0 + 0.4300i
    0 + 0.2000i    0 + 0.2400i    0 + 0.2800i    0 + 0.4300i
  Columns 5 through 6
    0 + 0.2000i    0 + 0.2000i
    0 + 0.2400i    0 + 0.2400i
    0 + 0.2800i    0 + 0.2800i
    0 + 0.4300i    0 + 0.4300i
    0 + 0.4700i    0 + 0.4700i
    0 + 0.4700i    0 + 0.5100i
zg =
  Columns 1 through 4
    0 + 0.2000i    0 + 0.2000i    0 + 0.2000i    0 + 0.2000i
    0 + 0.2000i    0 + 0.2400i    0 + 0.2400i    0 + 0.2400i
    0 + 0.2000i    0 + 0.2400i    0 + 0.2800i    0 + 0.2800i
    0 + 0.2000i    0 + 0.2400i    0 + 0.2800i    0 + 0.4300i
    0 + 0.2000i    0 + 0.2400i    0 + 0.2800i    0 + 0.4300i
    0 + 0.2000i    0 + 0.2400i    0 + 0.2800i    0 + 0.4300i
    0 + 0.2000i    0 + 0.2400i    0 + 0.2800i    0 + 0.4300i
  Columns 5 through 7
    0 + 0.2000i    0 + 0.2000i    0 + 0.2000i
    0 + 0.2400i    0 + 0.2400i    0 + 0.2400i
    0 + 0.2800i    0 + 0.2800i    0 + 0.2800i
    0 + 0.4300i    0 + 0.4300i    0 + 0.4300i
    0 + 0.4700i    0 + 0.4700i    0 + 0.4700i
    0 + 0.4700i    0 + 0.5100i    0 + 0.5100i
    0 + 0.4700i    0 + 0.5100i    0 + 0.7100i
```

Now we are going to perform the reduction of column and row 6 and column and row 5. We will use Kron's reduction.

```
%For Zbus
% Kron's Reduction
 rmv=7; %current matrix size
```

```
remove=rmv; %remove= which row/column to remove
for jj=1:(rmv)
   if jj ~= remove;
      j=jj;
      if jj >= remove;
         j=jj-1;
      end
      for kk=1:(rmv);
         if kk ~= remove;
            k=kk;
            if kk >= remove;
               k=kk -1;
            end;
            znew1(j,k)=z(jj,kk)-z(jj,remove)*z(remove,kk)/z(remove,remove);
         end;
      end;
   end;
end;
z1=znew1
%Kron's reduction again
rmv=6; %current matrix size
remove=rmv; %remove= which row/column to remove
for jj=1:(rmv)
   if jj ~= remove;
      j=jj;
      if jj >= remove;
         j=jj-1;
      end
      for kk=1:(rmv);
         if kk ~= remove;
            k=kk;
            if kk >= remove;
               k=kk -1;
            en d;
            znew2(j,k)=z1(jj,kk)-z1(jj,remove)*z1(remove,kk)/z1(remove,remove);
         end;
      end;
   end;
end;
z2=znew2
%Kron's reduction again
rmv=5; %current matrix size
remove=rmv; %remove= which row/column to remove
for jj=1:(rmv)
   if jj ~= remove;
      j=jj;
```

```
      if jj >= remove;
         j=jj-1;
      end
      for kk=1:(rmv);
         if kk ~= remove;
            k=kk;
            if kk >= remove;
               k=kk -1;
            end;
            znew3(j,k)=z2(jj,kk)-z2(jj,remove)*z2(remove,kk)/z2(remove,remove);
         end;
      end;
   end;
end;
z3=znew3
```

z1 = reduction from a 7 by 7 to a 6 by 6 matrix size. Elimination of bus 4

Columns 1 through 4

0 + 0.1437i	0 + 0.1324i	0 + 0.1211i	0 + 0.0789i
0 + 0.1324i	0 + 0.1589i	0 + 0.1454i	0 + 0.0946i
0 + 0.1211i	0 + 0.1454i	0 + 0.1696i	0 + 0.1104i
0 + 0.0789i	0 + 0.0946i	0 + 0.1104i	0 + 0.1696i
0 + 0.0676i	0 + 0.0811i	0 + 0.0946i	0 + 0.1454i
0 + 0.0563i	0 + 0.0676i	0 + 0.0789i	0 + 0.1211i

Columns 5 through 6

0 + 0.0676i	0 + 0.0563i
0 + 0.0811i	0 + 0.0676i
0 + 0.0946i	0 + 0.0789i
0 + 0.1454i	0 + 0.1211i
0 + 0.1589i	0 + 0.1324i
0 + 0.1324i	0 + 0.1437i

z2 = reduction from a 6 by 6 to a 5 by 5 matrix size. Elimination of bus 6

Columns 1 through 4

0 + 0.1216i	0 + 0.1059i	0 + 0.0902i	0 + 0.0314i
0 + 0.1059i	0 + 0.1271i	0 + 0.1082i	0 + 0.0376i
0 + 0.0902i	0 + 0.1082i	0 + 0.1263i	0 + 0.0439i
0 + 0.0314i	0 + 0.0376i	0 + 0.0439i	0 + 0.0675i
0 + 0.0157i	0 + 0.0188i	0 + 0.0220i	0 + 0.0337i

Column 5

0 + 0.0157i
0 + 0.0188i
0 + 0.0220i
0 + 0.0337i
0 + 0 .0369i

z3 = reduced from a 5 by 5 to a 4 by 4 matrix size. Elimination of bus number 5

0 + 0.1149i	0 + 0.0979i	0 + 0.0809i	0 + 0.0170i
0 + 0.0979i	0 + 0.1174i	0 + 0.0970i	0 + 0.0204i
0 + 0.0809i	0 + 0.0970i	0 + 0.1132i	0 + 0.0238i
0 + 0.0170i	0 + 0.0204i	0 + 0.0238i	0 + 0.0366i

»

For $Y_{bus,}$ we need to take the inverse of Z_{bus}. This is done with a simple command as it can be shown below.

```
Ybus=inv(z3)
Ybus =
    0 -30.0000i     0 +25.0000i     0 + 0.0000i     0 - 0.0000i
    0 +25.0000i     0 -50.0000i     0 +25.0000i     0 + 0.0000i
    0 + 0.0000i     0 +25.0000i     0 -31.6667i     0 + 6.6667i
    0 - 0.0000i     0 + 0.0000i     0 + 6.6667i     0 -31.6667i
```

Conclusions

As it was shown in figure 1, we need to know the electric system that we must analyze. Then we need to convert actual values of reactance from the models of electric equipment to per-unit values and create what we call the reactance diagram. Once this diagram is created it is our task to obtain the Z_{bus} matrix. By inverting the Z_{bus} matrix we can find Y_{bus} matrix. The program written in MATLAB took very little time to write and runs as fast as any higher-level programming language. Our students enjoy this experience because of the fact that changes to the reactance matrix can be done very easily and running the program to find the answer is a matter of seconds. Therefore we focus on the answers, the algorithm, system changes and ultimately the understanding of the material.

Bibliography

1. John J. Grainger, William D. Stevenson, Jr. *"Power Systems Analysis*," McGraw-Hill, First Edition, 1994.
2. Duane Hanselman, Bruce Littlefield, "*Mastering Matlab 5,"* Prentice-Hall, 1998.
3. Stephen J. Chapman, *Matlab "Programming for Engineers,"* Brooks/Cole Thomson Learning, 2000.
4. J. Duncan Glover, Kulukutla Sarma. "*Power System Analysis and Design,"* PWS-KENT, 2nd Edition, 2002.
5. Adrian Biran, Moshe Breiner," *Matlab for Engineers,"* Addison-Wesley, 1st Edition, 1995
6. J. Arrillaga, et.al. "*Computer Modeling of Electric Power Systems,*" 1st Edition, Wiley, 1991.
7. Mohamed E. El-Hawary, "*Electrical Power Systems Design and Analysis,"* 1 st Edit ion 1993. Reston.
8. Bruno Osorno, "*Fourier Analysis of a Single-Phase Full Bridge Rectifier Using Matlab,*" ASEE annual conference proceedings. Montreal, Canada. June 16-20, 2002

BRUNO OSORNO, is a professor of electrical and computer engineering at California State University Northridge. He is the lead faculty member in the Electric Power Systems and Power Electronics program. Professor Osorno has written over 20 technical papers. His current interest in research is Fuzzy Logic applications in Power-Electronics and electric adjustable speed drives. Professor Osorno is a senior member of IEEE and campus representative of ASEE.