Be name khodavande jano kherad

Kazin bartar andishe bar nagzarad

(A poet by Hakim Abolghasem Ferdosi)

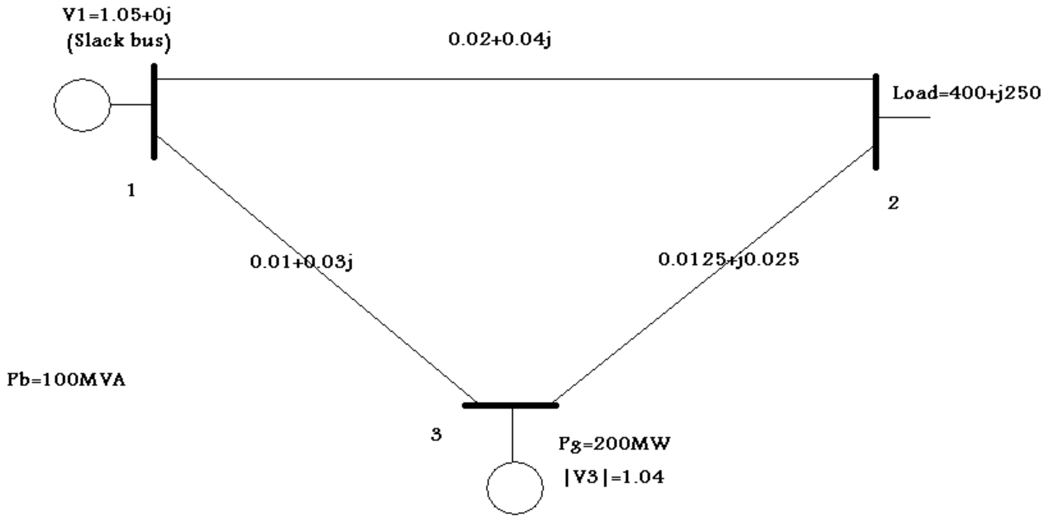
SALAM

What is Lfgauss?

This name stands for "Load Flow using Gauss-Seidel Method"; it's written in a TI-89 Titanium. The formulas used in it are based on "Power System Analysis by Hadi Saadat" (more info.), The program is not written for just giving the final answer, it solves the problem step by step and is specially useful in tests because most of the time you are supposed to solve the problem only for a few steps and write down the procedure, however you can use it for reaching a final answer just by a few changes but when there are programs like POWERWORLD or DIgSILENT I think it won't be needed.
Only those who have solved load flow problem with at least 3 bus and 4 iterations will understand how much useful this program can be however it is still very helpful for smaller networks.
There is also a subprogram which calculates the Ybus matrix. The Ybus calculated after running this program can be used in other programs that I have written like Lfnewton or Lffdcpld. By inversing this matrix you calculate Zbus which is helpful especially when you are going to solve a short circuit problem.
Required slack bus power, transferred power between buses, power loss on lines, current flowing through the lines and some other parameters of system are calculated too.

How to install:
Just send the "shin_adm" and "Lfgauss" files to your TI device. (Any folder but I use PSA for power system analyzing programs and functions.)
The shin_adm is program that generates YBUS matrix. The output is stored in variable "xxx" and this is the only variable that is used by the program, other variables are local.
Another thing to mention is that both programs can be archived after running once, but the variable xxx must be unlocked so shin_adm can store YBUS in it.

OK, let's see **how it works:**
Imagine this system:



Before anything I suggest you to change all impedance values to admittance:

y12=10-20j

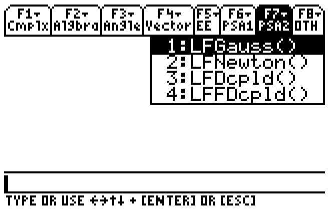
y13=10-30j

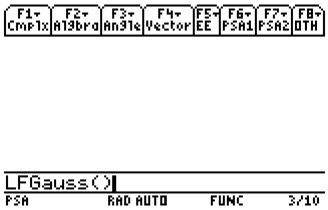
y23=16-32j

If you don't do this here, you will have to enter them in the program like this:

y12= (0.02+0.04j) ^-1

1. Run the program LFGauss() - (no parameters are needed.)





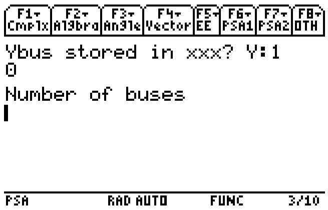
(& Press Enter)

(Most of the times I use the custom menu you see here, the most needed functions and programs for a student of Power Electronics Engineering can be found on it, you can even browse within your custom menus and at the same time the current folder changes on basis of the menu you are using and …)

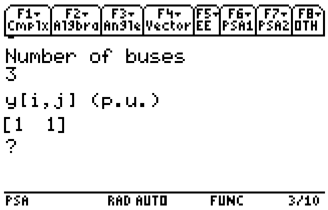
2. The program is asking you about Ybus, if the question has given this matrix to you just store it in this variable and then when you reach hear enter 1, otherwise you

should enter any number except 1 so the program automatically calls the other program shin_adm(n) and after a few steps it stores the Ybus in xxx. (n is number of buses in network.)

We assume that Ybus is not stored in xxx so for example I enter “0”:

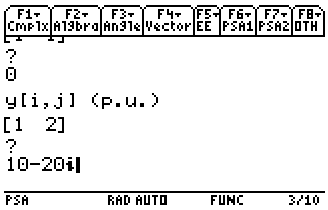


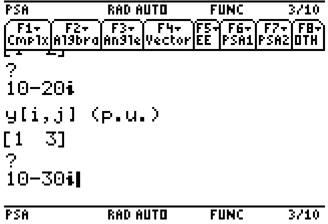
3. Enter number of buses. (3 in the above network)

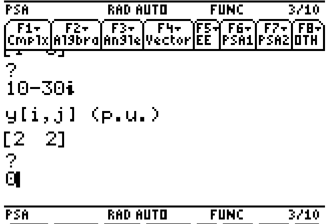


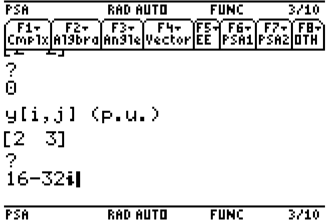
4. Entering the line data.
   - y [i , j] = the admittance between i and j buses. (Note that it’s different from Yij in Ybus matrix.)
     The parallel admittance (if it exists in your line model) is not used here. You should just enter the amount of connected admittance to both i and j buses.
   - y [i , i] = the admittance between bus i and the earth (ground). If the line doesn’t have parallel admittance (that is caused by capacitor between line and the earth) this amount is zero.

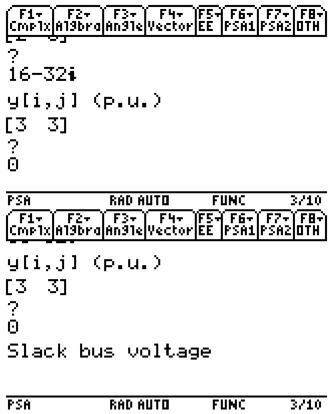
So follow the pictures:









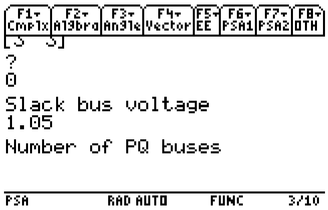


5. Up to now Ybus is created and stored in xxx.
   (You can generate this matrix in the same way by running shin_adm(n) program.)

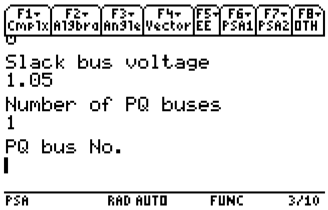
   As we saw the shin_adm(n) is a very usefull program to generate Ybus of a small network but if the network has more than 5 buses then this method of inputting data is not efficient, specially in real world problems. If you want to know why, read the blue text below:

   Another method of inputting line data:
   Another way is to input line data like this:
   [1,2,1+10j;1,3,4-5j;…;3,2;6-7j]
   And after inputting this matrix the program will understand that for example the admittance between bus 1 and bus 2 is 1+10j, and the same for bus 1 and 3, (4-5j), and all the other combinations which are not included are zero.
   If you test both methods you will find out that the one shown in forth step is faster for a small network. (Just try to input that matrix), but when we face a real world problem, it's totally different. Because many of buses are not connected to each other so you have to input zero many times; in this situation inputting line data in the form shown here is much faster. But our program is written to solve a problem in a test, and these kind of problems won't have more than 5 buses!
   Compressing Ybus:
   In real world problems we have lots of buses, for example, imagine a network with 30 buses, but every bus is only connected to a few other buses (usually something between 2 to 4 buses).
   Now most of the element of Ybus matrix become zero, in the above example Ybus is a 30*30 matrix, but each row has only 2 to 4 non zero elements, because the admittance between bus "i" and most of the other buses is zero (there is no line between them). In the past, computers didn't have enough memory to store this BIG matrix, so programmers suggested some ways to compress Ybus (it's not complicated at all, actually all zip programs do the same work, you can even write a program yourself to compress this matrix), but nowadays that computers have lots of memory it not needed to compress this matrix at all. (If you want to learn more about compressing this matrix I suggest you to study "Electric energy systems theory by O.I. Elgerd")

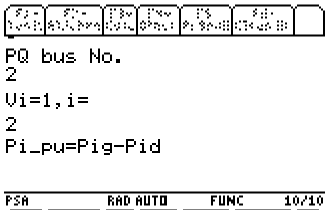
   Now is the time to enter bus data values:

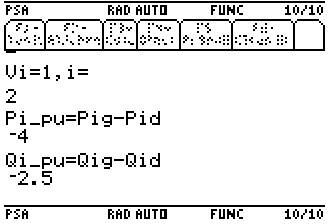


(I'm sure you know what is a PQ or a PV bus but just as a reminder, Slack bus, is a bus which is going to compensate the difference between load and generation. (These kinds of power stations should have special characteristics that I'm not going to explain here.) A PV bus is one which has a generator except the slack bus. PQ buses are the rest, they only have loads. Be careful that a capacitor bank changes a bus into generating bus or PV, because the capacitors generate reactive power.)

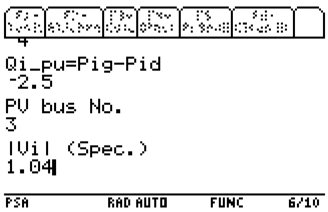


For every bus we assign a number, when using this program use number one for slack bus. (It's not a limit for program because the numbers are optional and in the other hand I can say that this is a standard, even if the question has given a different number for this bus just change the number with the number of a bus that is one.) On basis of the question's picture above, the PQ bus number is 2 and PV is 3.

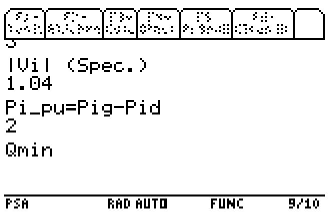




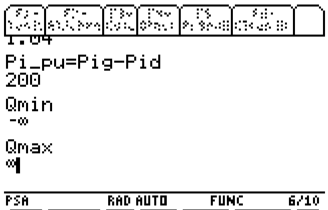
Another important point when using this method or any other method of load flow is that everything should be "per-unite". (The base value of power in this system is 100MVA)

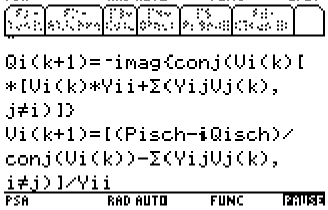


As you see the program automatically finds the number of PV buses, so you just need to enter bus number.

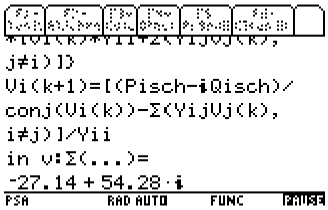


Most of generating buses have a reactive power generation limit, if there is also a load on the PV bus use Qmin=Qm-Qd in which Qm is the minimum generated reactive power by the generator and Qd is the demanded reactive power on this bus. If there is no limit to minimum reactive power for the generator, use -∞.

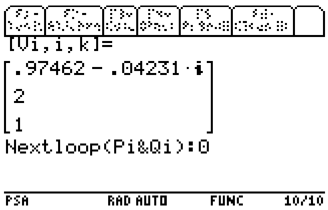




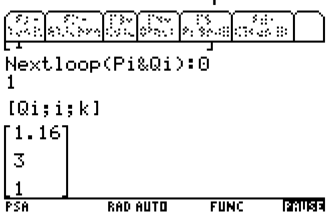
(The program is showing the formulas which are going to be used.)

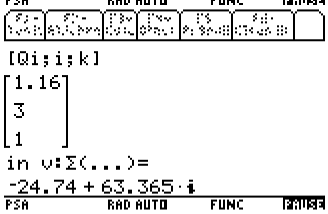


From here on, you will just need to write the answer, first the program calculates the Vi, so this sigma is the answer of sigma in the Vi's formula. (The sigma answer shown in each step is always the sigma answer used in the formula of what will come in the next line.)

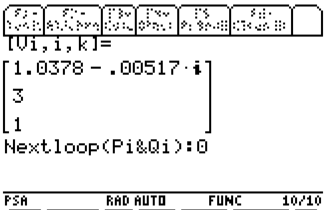


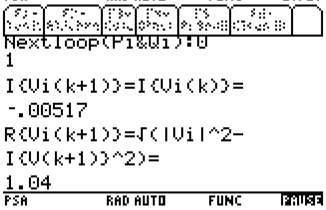
Vi (Voltage of bus i), i and k (Iteration Counter) are shown in a matrix.
If you want to calculate more Vi's with more repeats enter anything except 0.



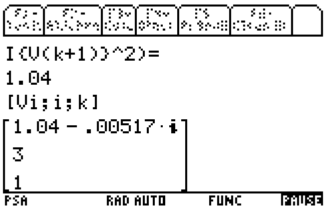


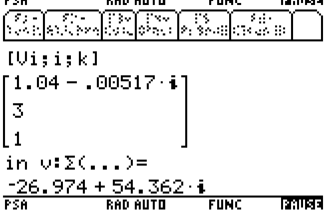
(This amount of Q is going to be used in the next iteration of calculating Vi.)
Again the sigma() =… is the answer of sigma calculation in the next shown parameter. (Vi in the screen below)

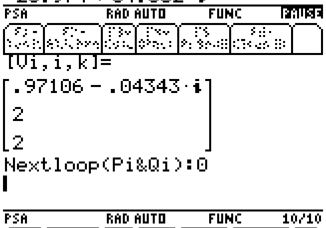


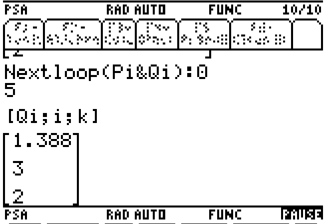


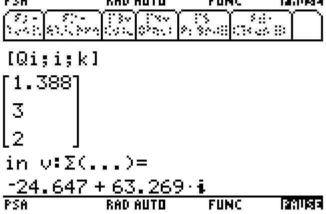
Voltage of bus 3 should be constant, so we keep the imaginary part of it and calculate the real part using the formula shown in the screen. (There are other methods which you don't need to do this step but as I said, I'm explaining everything using Saadat's Book, but don't worry, the final answer for every step will be the same.)

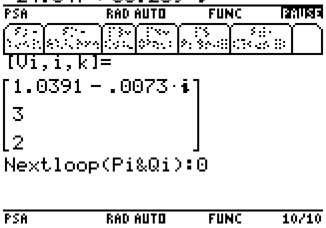






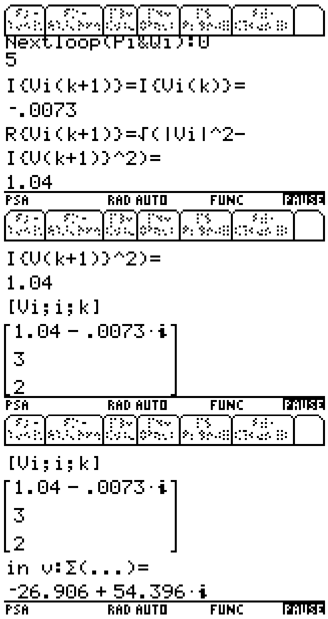




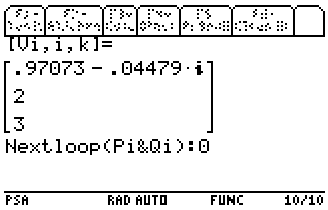


If you are going to calculate the answer only after three steps DO NOT enter zero yet, you must first complete step 3, otherwise you will receive a wrong answer in the end. The question may also ask you to repeat calculations until the error is less than epsilon (for example epsilon=0.00001) in this situation you should write down the answers in every step and check if the difference between all "Vi"s in this step and the "Vi"s in last step are smaller than 0.00001 or:

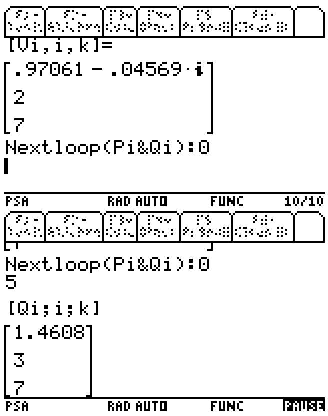
$$V(i, k) - V(i, k-1) < \text{epsilon} \quad i=2,\ldots,n$$

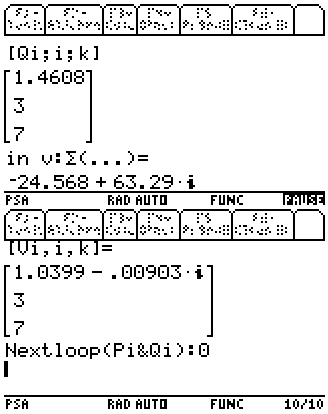


(You may want to break the iteration here after three steps. Or continue :)

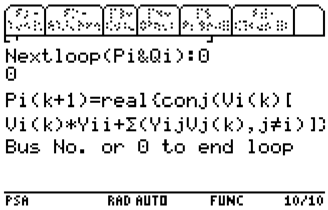


...



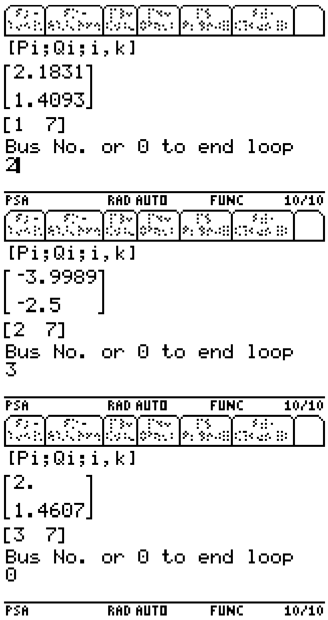


After seven steps the error is less that 5*10^-5.
Enter 0 to end iteration.



We wanted to know how much power should the slack bus produce; the above formula is used to calculate this. (Of course Q is the imaginary part of mentioned formula)

However in some kind of problems the power produced by other PV buses is questioned so I changed the program in Ver.1.1 in a way that you can calculate reactive and real power of any bus, just enter the bus number.



The rest of program just gives a few useful information about the system (Iij, Sij and Sloss,ij) in which Sloss,ij is the Power loss in line i to j.

```
Bus No. or 0 to end loop
0
[Iij;i;j]
[1.7076 - 1.131·i]
[1]
[2]
PSA   RAD AUTO   FUNC   PAUSE
```

```
[1]
[2]
[Iij;i;j]
[.37149 - .21123·i]
[1]
[3]
PSA   RAD AUTO   FUNC   PAUSE
```

```
[1]
[3]
[Iij;i;j]
[-2.2823 + 1.6325·i]
[2]
[3]
PSA   RAD AUTO   FUNC   PAUSE
```

…

```
[1]
[2]
[Sij;i;j]
[.39007 + .2218·i]
[1]
[3]
PSA   RAD AUTO   FUNC   PAUSE
```

…

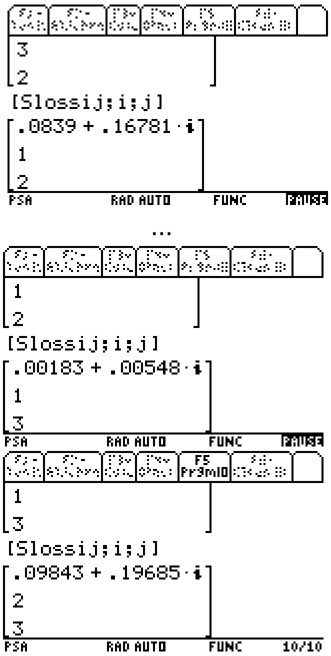
```
[2]
[1]
[Sij;i;j]
[-2.2898 - 1.4802·i]
[2]
[3]
PSA   RAD AUTO   FUNC   PAUSE
```

```
[2]
[1]
[Sij;i;j]
[-2.2898 - 1.4802·i]
[2]
[3]
PSA   RAD AUTO   FUNC   PAUSE
```

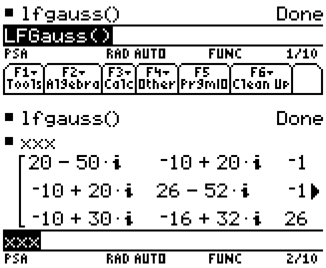
```
[2]
[3]
[Sij;i;j]
[-.38824 - .21632·i]
[3]
[1]
PSA   RAD AUTO   FUNC   PAUSE
```

```
[3]
[1]
[Sij;i;j]
[2.3883 + 1.6771·i]
[3]
[2]
PSA   RAD AUTO   FUNC   PAUSE
```

…



After you finish you can see the Ybus by typing xxx:



That's all.

The only **limit** I found when using this program was that it's not programmed to calculate the effect of tab-changing of transformers. (Of coarse there is a big difference between a real system and what we just explained, but as a program to help you solve your problems I think it's very useful.)

Other related programs:

I have also written 3 other programs which use Newton-Rophson, Decoupled and Fast Decoupled methods to solve load flow problem. (The DC load flow is too simple and I'm not going to program it.)

Here is a list of programs I've written:

- Load flow using Newton-Rophson method.
- Load flow using Decoupled method.
- Load flow using fast decoupled method.
- SCTM (Symmetrical components transformation matrix created by doctor C.L.Fortescue.) & SCTMI (Inverse of SCTM)

- puBasech((a function to calculate per unit values after change of base value.)
- linegc() (calculates line general coefficients for short and medium length lines.)
- llinegc() the same as linegc() but for long lines (uses hyperbolic functions.)
- GMD, GMR, GMRb (Geometric mean distance and radius of lines, these programs are slow I will be very thankful if you can send me any suggestions to improve these programs.)
- OCT() and SCT(). (Open circuit test and short circuit test for a DC machine.)
- IEEE (finds IEEE model of Induction machine.)
- W2D and D2W (delta<->why)
- …

Sorry for my poor English, if there is anything that I should add or any dictation or grammar errors just email me. Thanks. ;D

Any suggestions, any errors, anything, just contact me:
mailto:ali.db65@gmail.com

Ali Dehghan Banadaki
(A university student in Islamic Azad University of Gonabad.)
Thursday, February 07, 2008