Software Pakage User Manual : Chordal Conversion Based Convex Iteration Algorithm for Three-Phase ACOPF

Wei Wang Nanpeng Yu

Department of Electrical & Computer Engineering, University of California, Riverside

November 2017

Contents

1 Preinstalled Packages

1.1 Matlab

The Matlab is a commerical mathematic software package, which can be download here: Matlab Download
The instalation steps are:

- Start the Installer
- Install Using a MathWorks Account
- Review the Software License Agreement
- Log in to Your MathWorks Account
- Select the License You Want to Install
- Specify the Installation Folder
- Specify Products to Install
- Specify Installation Options
- Confirm Your Choices
- Complete the Installation

The supported Matlab version is listed in the Table 1. For More details about Matlab installation, please refer to Matlab Installation

Platform	R2009b-R2011b	R2012a	R2012b	R2013a+
linux32x86	Yes	Yes		
linux64x86	Yes	Yes	Yes	Yes
osx64x86		Yes	Yes	Yes
win32x86	Yes	Yes	Yes	Yes
win64x86	Yes	Yes	Yes	Yes

Table 1: supported Matlab version.

1.2 Yalmip

1. The Yalmip is an open source parser for optimization, which can be downloaded here: Yalmip Download
2. After downloading and extracting the package, add the following directories to Matlab path

/YALMIP-master
/YALMIP-master/extras
/YALMIP-master/solvers
/YALMIP-master/modules
/YALMIP-master/modules/parametric
/YALMIP-master/modules/moment
/YALMIP-master/modules/global
/YALMIP-master/modules/sos
/YALMIP-master/operators

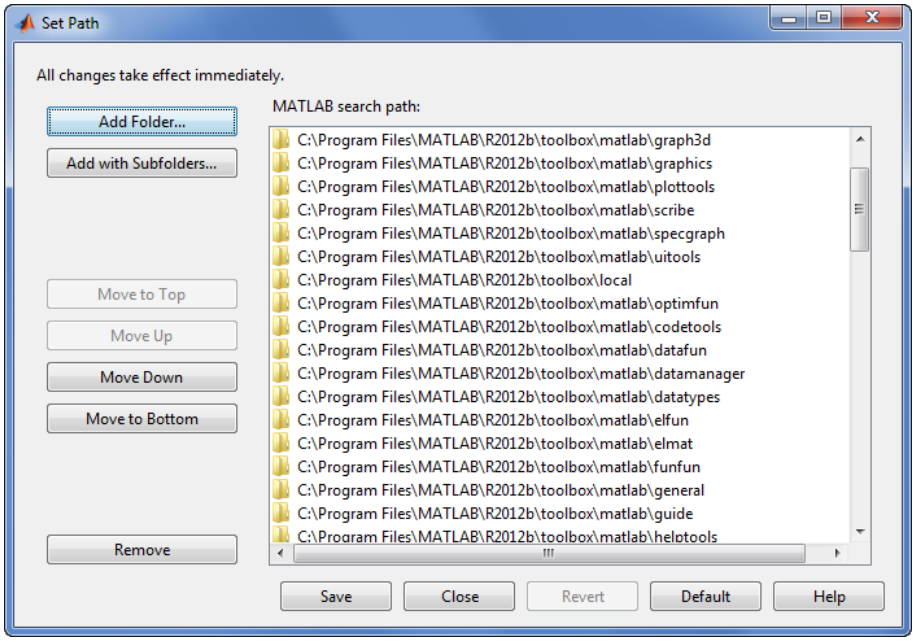

Figure 1: Add path to Matlab

To add directory to Matlab, click **Set Path** in the Environment section On the **Home** tab. Add the above paths in the appearing window shown in Fig. 1.

To verify the installation of Yalmip, type **yalmiptest** in the command window of Matlab

1.3 Mosek

Mosek is a commerical optimization solver which offers free academic licenses and can be downloaded here: Mosek Download

To obtain an academic license, please apply on the company website here:Mosek License

After installing Mosek, add the follwing path to Matlab. For Windows Users:

```
% For versions R2009b to R2011b
addpath c:\Program Files\mosek \7 \toolbox \r2009b
% For versions R2012a and R2012b
addpath c:\Program Files \mosek \7 \toolbox \r2012a
%For R2013a or newer
addpath c: \Program Files \mosek \7 \toolbox \r2013a
```

For Linux/UNIX/MAC OS X Users:

```
% For versions R2009b to R2011b
addpath /home/user/mosek/7/toolbox/r2009b
% For versions R2012a and R2012b
addpath /home/user/mosek/7/toolbox/r2012a
%For R2013a or newer
addpath /home/user/mosek/7/toolbox/r2012a
```

After obtain the license file, place it here for Windows users:

```
%USERPROFILE% \mosek \mosek.lic
```

For the other users and license activation details, please refer here : Mosek Licence Activation

To verify the installation of Mosek, type **mosekdiag** in the command window of Matalb.

2 Optimization Package

The software package include the files for the test cases of IEEE distribution network test feeders including 4-bus, 13-bus, 34-bus, 37-bus, 123-bus, 900-bus networks and the network in the paper.

2.1 File structure

There are three main parts of the file: the main file is the file to run. The data files contain the network parameter data and grid partition data. The libraries files contain the graph partition functions and unit conversion function.

```
% Main files:
bus(N1)_central.m

% Data files
bus(N1)_node_line_data.m
bus(N1)_(N2)Area_data.m

% Libraries
ft2mile.m
graph_partition.m
generate_boundary.m
cut_into_two.m
print_Areas(N1).m
```

where $N1$ is number of bus and $N2$ is the number of portioned areas of the network

2.2 Main file

If the names of network data files or the partitioning of the network are updated in the data files, please update the lines correspondingly:

```
%%%%%%%Bus Data%%%%%%%%%%
bus(N1)_node_line_data
%%%%%%%%%%%%%%%%%%%%%%%
bus(N1)_(N2)Area_data
%%%%%%%%%%%%%%%%%%%%%%%
```

If the three-phase prices need to be changed, please modify the lines correspondingly:

```
c_k1_p1=[1,0.5,0.2];
c_k1_p2=[1,0.5,0.2];
```

where $p1$ is the price vector for the substation, $p2$ is the price vector for distributed energy resources.

If the voltage constraints need to be changed, please modify the lines correspondingly:

```
V_k_p_min=0.95;
V_k_p_max=1.05;
```

If the convergence criteria needed to be changed, please modify the lines correspondingly:

```
terminate_flag2=1;
for i=1:N_Area
terminate_flag2 = terminate_flag2 && (Area(i).Rank_E<0.00001);
end
```

If the solver settings need to be change, please please modify the lines correspondingly:

```
ops=sdpsettings('solver','mosek-sdp');
ops=sdpsettings(ops,
mosek.MSK_DPAR_INTPNT_CO_TOL_MU_RED,10^(-7));
ops=sdpsettings(ops,
mosek.MSK_DPAR_INTPNT_CO_TOL_INFEAS,10^(-7));
ops = sdpsettings(ops,'verbose',0);
ops = sdpsettings(ops,'cachesolvers',1);
```

2.3 Data files

If the voltage base and power base need to be changed, please modify the lines in file $bus(N1)\_node\_line\_data.m$:

```
%12.47KV line to line
V_base=12.47/sqrt(3);
%10MVA
S_base = 10;
```

The node data with phase, generation, and load information is stored in the following data structure:

```
% 1 indicates the existing of phase a or b or c
node(2).phase=[1,1,1];
% number of phases
node(2).np=3;
% 1 indicates it is a node with generation
node(2).g=0; % real power of load on phase a, b, c
node(2).P=[0,0,0]/1000/S_base;
% reactive power of load on phase a, b, c
node(2).Q=[0,0,0]/1000/S_base;
```

The line data with phase, connection, and impedance information is stored in the following data structure:

```
% length in ft
LineSeg(1).Length= 2000;
% from node
LineSeg(1).node1 = 1;
% to node
LineSeg(1).node2 = 2;
% 1 indicates the existing of phase a or b or c
LineSeg(1).phase = [1,1,1];
% number of phases
LineSeg(1).np = 3;
% base voltage
LineSeg(1).Vbase = 12.47/sqrt(3);
% impedance
LineSeg(1).Z_pu = Zy_permile*ft2mile(LineSeg(1).Length)...
*S_base/(LineSeg(1).Vbase^2);
% conductance
LineSeg(1).Y_pu = inv(LineSeg(1).Z_pu);
% line susceptance
LineSeg(1).B_pu = 0;
```

The Area data is stored in the following data structure in file $bus(N1)\_(N2)Area\_data.m$:

```
% the nodes in partitioned area
Area(1).node=[1,2];
% the nodes in extended area (i.e include boundary)
Area(1).extnode=[1,2,3];
```

This file is automatically generated, which is subject to modification.

2.4 Library files

The functionality of each library file are described as following:

```
% convert ft into miles
ft2mile.m
% partition the network modeled in bus(N1)_node_line_data.m file
graph_partition.m
% generate boundary conditions for partitioned areas
generate_boundary.m
% find cut greedily in each step
cut_into_two.m
% generate bus(N1)_(N2)Area_data.m
print_Areas(N1).m
```

If the name of $bus(N1)\_(N2)Area\_data.m$ file need to be updated, please modify the following lines in $print\_Areas(N1).m$

```
fid=fopen('bus4_1Area_data.m','w');
```

```
fclose(fid);
bus4_1Area_data;
```

2.5 Run steps

The steps to run the program are:

- transform the load and network parameter data into file bus(N1)_node_line_data.m with the defined data structures
- run graph_partition.m file to generate partitioned area data
- run print_Areas(N1).m file to generate bus(N1)_(N2)Area_data.m file
- run the bus(N1)_central.m