Dynamic Equivalencing: A Tutorial

Joe Chow
May 13, 1992

1. Introduction

The purpose of dynamic equivalencing is to reduce a large size power system model to a smaller one for control system design or rapid security assessment. The process of dynamic equivalencing consists of two distinct steps: 1. the identification of the coherent groups, and 2. the aggregation of the power network and the machine models. In the Power System Toolbox (PST), there are several functions available for each step. The remainder of this tutorial will discuss the use these functions to set up a reduced order model.

2. Data Requirement

The user needs to set up the bus data, line data, and machine data in the arrays bus, line, and mac_con, respectively. The bus data must be a solved loadflow case, like the output of the function loadflow. If a solved loadflow data is used as the input, the reduced model will also be a solved loadflow data set, which is important for maintaining the accuracy of the reduced models.

3. Coherency

The first step in dynamic equivalencing is to find the coherent machines to be aggregated. There are five algorithms available in PST to perform this step:

1. Slow coherency [1] (function L_group)
2. Tight coherency (function ex_group)
3. Clustering algorithm [2] (function zabors)
4. Weak link algorithm [3] (function weaklink)
5. Sign coherency [4] (function sign_coh)

All of these methods are disturbance independent and require as the input the linearized system matrix $A = -M^{-1}K$, where M is the matrix of machine inertias, and K is the matrix of synchronizing coefficients. The matrix A can be generated using the function svm_em. The clustering and weak link algorithms work directly with the matrix A, while the slow, tight and sign coherency methods use the eigenvector matrix of the slow eigenvalues of A.

Instead of using these functions directly, the function coherent allows the user to pick any one of the methods to be used.

To obtain a relatively small number of coherent groups, the slow coherency algorithm in L_group seems to be most desirable. If a larger number of groups is desired, it is suggested that the tight coherency algorithm in ex_group be used, with the number of eigenvalues set to a relatively small number and the tolerance value set to about 0.95.

4. Aggregation

The areas found from the coherency algorithms can be used in the aggregation process. The machines to be aggregated are stored in an array. Machines not contained in that array will not be aggregated, such as the case of the machines in the study area.

There are three aggregation algorithms available in PST:

1. Podmore method [5] (function podmore)
2. Inertia aggregation method [6] (function i_agg)
3. Slow coherency method [6] (function s_coh3)

In the Podmore method, the coherent machines are moved to a common terminal bus and aggregated. This technique tends to produce an impedance stiffening effect, causing the frequencies of the swing modes to increase. The inertia aggregation method is a zeroth order approximation of the slow coherency method based on the singular perturbation theory [1]. In this technique, the coherent machines are moved to a common *internal* bus and aggregated. This reduces the stiffening effect. In practice, a 50% improvement in the approximation of the frequencies of the dominant modes can be obtained. The slow coherency method is a first order approximation to re-introduce additional links between the buses. Theoretically, the slow coherency method can produce another 50% improvement in the approximation of the frequencies of the dominant modes. However, unlike the Podmore and the inertia aggregation process, it is not possible to exactly reconstruct a physical network in slow coherency. Thus the slow coherency aggregate model usually is only slightly better than the inertia

aggregate model. Of the three methods, the inertia aggregation and the slow coherency methods are recommended. If the input bus and line data contain a solved loadflow case, the output aggregated bus and line data will also contain a solved loadflow case. These data can then be used directly for simulation and to build state variable models, without solving for a new loadflow.

The machine model parameters are aggregated such that impedances are combined in parallel and time constants averaged. The aggregation functions automatically call the function eqgen to perform the machine model aggregation.

The unnecessary buses in the aggregated bus and line data can be eliminated by using the function reduce. All three aggregation methods generate phase shifters. In the function reduce, the nodes to be eliminated may be connected by lines with phase shifters. Reconstruction of the line parameters from the non-symmetrical reduced admittance matrix is done properly, such that the solved loadflow will be maintained. For large systems, sometimes it is more advisable not to eliminate the unnecessary buses as the elimination process increases the density of the network connections.

5. Demos

Several demo files are available:

1. coh_np16.m – coherency grouping of the NPCC 16 machine system.
2. coh_np48.m – coherency grouping of the NPCC 48 machine system.
3. agg_np48.m – aggregation of the NPCC 48 machine system.

In coh_np16.m and coh_np48.m, the user can choose any one of the 5 coherency functions to select the coherent areas. In agg_np48.m, the use can choose any one of the 3 aggregation functions. In addition, electromechanical modes are computed for the reduced systems, and some of the load buses are eliminated using reduce.

6. References

1. J. H. Chow, editor, *Time-Scale Modeling of Dynamic Networks with Applications to Power Systems*, Springer-Verlag, Berlin, 1982.
2. J. Zaborszky *et al*, "A Clustered Dynamic Model of a Class of Linear Autonomous Systems Using Simple Enumerative Sorting," *IEEE Trans. Circuits and Systems,* vol. CAS-29, pp. 747-758, 1982.

3. R. Nath, S. S. Lamba and K. S. P. Rao, "Coherency Based System Decomposition into Study and External Areas using Weak Coupling," *IEEE Trans. Power Apparatus and Systems,* vol. PAS-104, pp. 1443-1449, 1985.

4. B. Eliasson, *Damping of Power Oscillations in Large Power Systems,* Ph.D. Thesis, Lund Institute, 1990.

5. R. Podmore, "Development of Dynamic Equivalents for Transient Stability Studies," EPRI Report EL-456, 1977.

6. J. H. Chow, R. A. Date, H. Othman and W. W. Price, "Slow Coherency Aggregation of Large Power Systems," in *Eigenanalysis and Frequency Domain Methods for System Dynamic Performance,* IEEE Publication 90TH0292-3- PWR, 1990.

Function:

add_area,rem_area

Purpose:

Add or remove a machine from an area.

Synopsis:

[areax,nareax] = add_area(area,narea,i,k)
[areax,nareax] = rem_area(area,narea,i)

Description:

[areax,nareax] = add_area(area,narea,i,k) adds machine i to area k. The number of areas is equal to the number of rows of the vector narea. The ith entry of narea is the number of machines in the ith area. The ith row of the matrix area contains the machines in the ith area. The output variables areax and nareax contain the machines in the areas and the number of machines in each area after machine i has been added.

[areax,nareax] = rem_area(area,narea,i) removes machine i from its area. The output variables areax and nareax contain the machines in the areas and the number of machines in each area after machine i has been removed.

Examples:

For the input data

area =

$$\begin{matrix} 1 & 2 & 4 \\ 5 & 0 & 0 \end{matrix}$$

narea =

$$\begin{matrix} 3 \\ 1 \end{matrix}$$

[areax,nareax] = add_area(area,narea,3,2) returns

areax =

$$\begin{matrix} 1 & 2 & 4 \\ 3 & 5 & 0 \end{matrix}$$

nareax =

$$\begin{matrix} 3 \\ 2 \end{matrix}$$

and [areay,nareay] = rem_area(area,narea,2) returns

areay =

$$\begin{matrix} 1 & 4 \\ 5 & 0 \end{matrix}$$

nareay =

$$\begin{matrix} 2 \\ 1 \end{matrix}$$

Algorithm:

The functions are implemented in the M-files add_area and rem_area in the POWER SYSTEM TOOLBOX.

See also: ex_group

Function:

coh_map

Purpose:

Generate the coherency map of a power system based on the mode shapes.

Synopsis:

[g,cmap] = coh_map(V,tol)

Description:

[g,cmap] = coh_map(V,tol) sets up the coherency map g of a power system based on the mode shapes specified in the eigenvector matrix V. The ijth entry of g is the cosine of the angle between the ith and jth row vectors of the matrix V minus tol. The input variable tol < 1 is the coherency threshold. If the ijth entry of g is positive, then machines i and j are coherent. The matrix cmap is obtained by setting the negative entries of g to zero. By displaying g using format +, a visual coherency map is generated. Alternatively, mesh(cmap) shows a 3D display of coherency. A formal coherency grouping of the machines can be obtained by applying the function ex_group to g.

Examples:

Let V =

$$\begin{matrix} 1.0000 & 1.0000 \\ 1.0000 & 0.8000 \\ 1.0000 & -1.0000 \end{matrix}$$

Then [g,cmap] = coh_map(V,0.9) returns

cmap =

$$\begin{matrix} 0.1000 & 0.0938 & 0.0000 \\ 0.0938 & 0.1000 & 0.0000 \\ 0.0000 & 0.0000 & 0.1000 \end{matrix}$$

g =

$$\begin{matrix} 0.1000 & 0.0938 & -0.9638 \\ 0.0938 & 0.1000 & -0.8527 \\ -0.9638 & -0.8527 & 0.1000 \end{matrix}$$

Using format +,
g =

$$\begin{matrix} + & + & - \\ + & + & - \\ - & - & + \end{matrix}$$

Algorithm:

The columns of the matrix V is first normalized such that the 2-norm is unity. Then the ijth entry of g is calculated as

$$(\mathsf{V}_i, \mathsf{V}_j)/(|\mathsf{V}_i||\mathsf{V}_j|) - tol,$$

where V_i is the ith row of V, and (.,.) denotes the dot product of two vectors.

The function is implemented in the M-file coh_map in the POWER SYSTEM TOOLBOX.

See also: ex_group

Function:

coherent

Purpose:

Find the coherent areas of a power system.

Synopsis:

[area,narea,areal,nareal] = coherent(bus,line,meth,ns,tol)

Description:

[area,narea,areal,nareal] = coherent(bus,line,meth,ns,tol) allows the user to find the slow coherent areas in a power system using any one of the algorithms available in the POWER SYSTEM TOOLBOX. The input variables bus and line contain a solved loadflow. The machine data should be contained in mac_con.

If meth is set to 1, then the slow coherency algorithm will be used to find the coherent areas. The user has to supply ns, the number of slow eigenvectors used for the identification.

If meth is set to 2, then the tight coherency algorithm will be used to find the coherent areas. The user has to supply ns, the number of slow eigenvectors used for the identification, and tol, the tolerance used to determine the coherent groups.

If meth is set to 3, then the Zaborszky clustering algorithm will be used to find the coherent areas. The user has to set ns to 0 and supply tol, the tolerance used to determine the coherent groups.

If meth is set to 4, then the weaklink algorithm will be used to find the coherent areas. The user has to set ns to 0 and supply tol, the tolerance used to determine the weakly coherent groups.

If meth is set to 5, then the sign coherent algorithm will be used to find the coherent areas. The user has to supply ns, the number of slow eigenvectors used for the identification.

Algorithm:

See the functions L_group, ex_group, zabors, weaklink, sign_coh for the details of the various grouping algorithms.

The function is implemented in the M-file coherent in the POWER SYSTEM TOOLBOX.

See also: coh_map,L_group,ex_group,sign_coh,zabors,weaklink

Function:

diagb,offdb

Purpose:

Retain the diagonal or off-diagonal blocks of a matrix.

Synopsis:

[B] = diagb(A,area,narea)
[B] = offdb(A,area,narea)

Description:

[B] = diagb(A,area,narea) sets the off-diagonal blocks of A to zero and returns the result in the matrix B. The diagonal block structure is specified by the input variables area and narea. The number of diagonal blocks is equal to the number of rows of the vector narea. The ith entry of narea is the number of indices in the ith diagonal block. The ith row of the matrix area contains the (positive) indices of the ith diagonal block.

[B] = offdb(A,area,narea) sets the diagonal blocks of A to zero and returns the result in the matrix B.

Examples:

For the input data

A =

$$\begin{matrix} 0.05 & -0.01 & 0.03 & 0.02 \\ -0.01 & 0.05 & -0.02 & -0.03 \\ 0.03 & -0.02 & 0.05 & 0.01 \\ 0.02 & -0.03 & 0.01 & 0.05 \end{matrix}$$

area =

$$\begin{matrix} 1 & 3 & 4 \\ 2 & 0 & 0 \end{matrix}$$

narea =

$$\begin{matrix} 3 \\ 1 \end{matrix}$$

B = diagb(A,area,narea) returns

B =

$$\begin{matrix} 0.05 & 0 & 0.03 & 0.02 \\ 0 & 0.05 & 0 & 0 \\ 0.03 & 0 & 0.05 & 0.01 \\ 0.02 & 0 & 0.01 & 0.05 \end{matrix}$$

and C = offdb(A,area,narea) returns

C =

$$\begin{matrix} 0 & -0.01 & 0 & 0. \\ -0.01 & 0 & -0.02 & -0.03 \\ 0 & -0.02 & 0 & 0 \\ 0 & -0.03 & 0 & 0 \end{matrix}$$

Algorithm:

The functions are implemented in the M-files diagb and offdb in the POWER SYSTEM TOOLBOX.

See also: ex_group

Function:

element

Purpose:

Test for membership.

Synopsis:

[k] = element(x,v)

Description:

[k] = element(x,v) returns k = 1 if the scalar x is an entry of the vector v. Otherwise k = 0.

Algorithm:

The function is implemented in the M-file element in the POWER SYSTEM TOOLBOX.

See also: ex_group

Function:

eqgen

Purpose:

Computes the aggregate generator model parameters for a group of coherent generators.

Synopsis:

[nmac] = eqgen(mac,mac_list,basemva,bus_num,mac_num)

Description:

[nmac] = eqgen(mac,mac_list,basemva,bus_num,mac_num) computes the aggregate generator model parameters for a group of coherent generators contained in the vector mac_list. The internal machine numbers are used in mac_list. The array mac contains the machine constants, which are normally contained in mac_con. Note that since mac_con is a global variable, it cannot be passed in a function. The constant basemvc is the system base MVA, which will be used as the base MVA for the aggregate machine. The aggregate machine will be numbered as mac_num and connected to the bus bus_num.

Algorithm:

The aggregate generator impedances and time constants will be computed as the parallel combination of the coherent generator impedances and the averages of the coherent generator time constants, weighted according to the machine inertias, respectively.

The function is implemented in the M-file eqgen in the POWER SYSTEM TOOLBOX.

See also: podmore, i_agg, s_coh3

Function:

ex_group

Purpose:

Find the slow coherent areas of a power system using the tight coherency algorithm.

Synopsis:

[area,narea,areal,nareal] = ex_group(g)

Description:

[area,narea,areal,nareal] = ex_group(g) uses the coherency matrix g to determine the loose-coherent and tight-coherent areas in a power system. The coherency matrix g, the first output variable of the function coh_map, is generated from an eigenvector matrix V.

In a loose coherent area, a machine is coherent to at least one other machine. The number of areas is not limited to be the same as the number of columns in V. The number of loose coherent areas is given by the number of rows in the output matrix areal. The machines in the ith coherent area are contained in the ith row of areal and the number of the machines in the ith area is the ith entry of the vector nareal.

In a tight-coherent area, the coherency of the machines is larger, on average, than the coherency threshold. The number of coherent areas is given by the number of rows in the output matrix area. The machines in the ith coherent area are contained in the ith row of area and the number of the machines in the ith area is the ith entry of the vector narea.

Examples:

Let V be the eigenvector matrix corresponding to the 9 slowest eigenvalues of the NPCC 48 machine system [1] and the coherency threshold tol = 0.9.

```
[g] = coh_map(V,tol);
```

[area,narea] = ex_group(g)

area =

3	4	5	6	7	8	0	0	0
1	2	9	0	0	0	0	0	0
15	16	17	18	19	20	21	22	23
10	0	0	0	0	0	0	0	0
13	14	26	0	0	0	0	0	0
24	25	0	0	0	0	0	0	0
11	12	36	0	0	0	0	0	0
27	28	29	30	0	0	0	0	0
31	0	0	0	0	0	0	0	0
32	37	38	40	42	0	0	0	0
33	0	0	0	0	0	0	0	0
34	35	0	0	0	0	0	0	0
39	0	0	0	0	0	0	0	0
41	0	0	0	0	0	0	0	0
43	44	45	46	0	0	0	0	0
47	0	0	0	0	0	0	0	0
48	0	0	0	0	0	0	0	0

narea =

6	3	9	1	3	2	3	4	1	5	1	2	1	1	4	1	1

Algorithm:

The tight coherency algorithm is based on a collection of expert system rules on coherency. The rules are listed in the following. Let G be the coherency matrix and J_α be the machines in coherent area α.

1. Machines i and j are coherent if $G(i, j) > 0$.
2. If machines i and j are coherent and machines j and k are coherent, then machines i and k are also coherent.
3. A *loose* coherent area is formed by machines that are coherent under Rules 1 and 2.
4. Extract a submatrix G_α from G for all the machines in a coherent area J_α. Under Rule 2, not all $G_\alpha(i, j)$ are positive.
5. If all $G_\alpha(i, j)$ are positive, then J_α is a *packed* area.

6. If the column sums of G_α excluding the diagonal entries are all positive, then J_α is a *tight* coherent area. In a tight coherent area, a machine is coherent, on the average, to every other machines in the same area. From Rule 5, a packed area is also a tight area.

7. If any of the column sums of G_α excluding the diagonal entries is negative, then J_α should be decomposed into tight areas.

8. The least coherent machine in a loose-coherent group is the one corresponding to the columns of G_α with the smallest average sum.

9. The coherency of an area may be improved by removing the least coherent machine from the area and reassigning it to a different area to achieve a tighter coherency.

10. Given two partitions I_1 and I_2 of a loose-coherent area, the partition I_1 is tighter than I_2 if the sum of the off-diagonal entries of G_α corresponding to I_1 is smaller than that of I_2.

Based on the expert system rules on coherency, the following algorithm for tight coherency is used:

1. Find the loose kcoherent areas using Rules 1 - 3.

2. For each coherent area,

   (a) Use Rules 6 to determine tight area, which requires no further decomposition.

   (b) If the area is not tight, decompose the area into tight coherent areas. Start by identifying the least coherent machine in the area using Rule 8 and reassigning it to improve the coherency using Rule 10. Repeat the process until no improvement is possible.

The function is implemented in the M-file ex_group in the POWER SYSTEM TOOLBOX.

See also: coh_map,L_group

Reference:

1. J. H. Chow, editor, *Time-Scale Modeling of Dynamic Networks with Applications to Power Systems*, Springer-Verlag, Berlin, 1982.

Function:

i_agg

Purpose:

Aggregate the coherent machines using the inertia aggregation method.

Synopsis:

[nbus,nline,nmac_con] = i_agg(bus,line,area,narea)

Description:

[nbus,nline,nmac_con] = i_agg(bus,line,area,narea) aggregates the coherent machines contained in each row of the array area onto a new aggregate bus. The bus and line data of the full order system are contained in the arrays bus and line. The bus array must contain a solved loadflow of the system. The vector narea contains the number of machines in each row of area. The function i_agg also requires the machine data in mac_con which is a global variable.

On output, nbus, nline, and nmac_con contain the aggregated bus, line and machine data, respectively. If the input bus data contain a solved loadflow, nbus will also contain a solved loadflow.

Algorithm:

In the inertia aggregation algorithm [1], an aggregate internal node of the coherent generator internal nodes is created. The voltage magnitude and phase of the new aggregate node are the averages of the coherent generator terminal internal nodes. Then the coherent generators are moved to the aggregate internal node, and the original generator internal nodes are connected to the aggregate internal node through ideal transformers and phase shifters to preserve the power flow. The original generator terminal buses are preserved, in contrast to the function podmore which eliminates these buses. The coherent machines are aggregated into a single aggregate machine by paralleling the machine impedances and averaging the time constants according to the machine inertias.

The inertia aggregation method tends to reduce the stiffening effect in the aggregate network by 50% over the Podmore method, because the generators are aggregated at the generator internal nodes. The inertia aggregation model is the zeroth order approximation from the singular perturbation asymptotic expansion method.

The function is implemented in the M-file i_agg in the POWER SYSTEM TOOLBOX.

See also: podmore, s_coh3, eqgen

Reference:

1. J. H. Chow, R. A. Date, H. Othman and W. W. Price, "Slow Coherency Aggregation of Large Power Systems," in *Eigenanalysis and Frequency Domain Methods for System Dynamic Performance,* IEEE Publication 90TH0292-3- PWR, 1990.

Function:

L_group

Purpose:

Find the slow coherent areas of a power system using the L-matrix clustering algorithm.

Synopsis:

[area,narea,L] = L_group(V)

Description:

[area,narea,L] = L_group(V) uses the eigenvector matrix V to find the slow coherent areas in a power system. The matrix V can be obtained from the function V_slow. The number of coherent areas is equal to the number of columns in V. The machines in the ith coherent area are contained in the ith row of the output matrix area and the number of the machines in the ith area is the ith entry of the vector narea.

Examples:

Let V be the eigenvector matrix corresponding to the 9 slowest eigenvalues of the NPCC 48 machine system [1].

[area,narea,L] = L_group(V) returns

area =

5	1	2	3	4	6	7	8	9	0	0	0
39	42	0	0	0	0	0	0	0	0	0	0
44	43	45	46	0	0	0	0	0	0	0	0
34	35	0	0	0	0	0	0	0	0	0	0
48	0	0	0	0	0	0	0	0	0	0	0
41	32	37	38	40	0	0	0	0	0	0	0
17	13	14	15	16	18	19	20	21	22	23	24
29	10	27	28	30	0	0	0	0	0	0	0
36	11	12	33	0	0	0	0	0	0	0	0

0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
25	26	31	47
0	0	0	0
0	0	0	0

narea =

9	2	4	2	1	5	16	5	4

Algorithm:

Gaussian elimination with complete pivoting is applied to the matrix V to find the reference machines. Reorder V such that V_1, the first part of V, contains the rows corresponding to the reference machines, and V_2, the second part of V, contains the rows corresponding to the other machines. Then

$$L = V_2 V_1^{-1}.$$

The largest entry in each row of L is used to identify the coherency of the other machines with the reference machines. A detailed description of the algorithm can be found in [1,chapter 5].

The function is implemented in the M-file L_group in the POWER SYSTEM TOOLBOX.

See also: coh_map,ex_group

Reference:

1. J. H. Chow, editor, *Time-Scale Modeling of Dynamic Networks with Applications to Power Systems*, Springer-Verlag, Berlin, 1982.

Function:

podmore

Purpose:

Aggregate the coherent machines using the Podmore (EPRI/SCI) method.

Synopsis:

[nbus,nline,nmac_con] = podmore(bus,line,area,narea)

Description:

[nbus,nline,nmac_con] = podmore(bus,line,area,narea) aggregates the coherent machines contained in each row of the array area onto a new aggregate bus. The bus and line data of the full order system are contained in the arrays bus and line. The bus array must contain a solved loadflow of the system. The vector narea contains the number of machines in each row of area. The function podmore also requires the machine data in mac_con which is a global variable.

On output, nbus, nline, and nmac_con contain the aggregated bus, line and machine data, respectively. If the input bus data contain a solved loadflow, nbus will also contain a solved loadflow.

Algorithm:

In the Podmore aggregation algorithm [1], an aggregate bus of the coherent generator terminal buses is created. In the function podmore, the voltage magnitude and phase of the new aggregate bus are the averages of the coherent generator terminal buses. Then the coherent generators are moved to the aggregate bus, and the original generator terminal buses are connected to the aggregate bus through ideal transformers and phase shifters to preserve the power flow. The original generator terminal buses are eliminated. The coherent machines are aggregated into a single aggregate machine by paralleling the machine impedances and averaging the time constants according to the machine inertias.

The Podmore method tends to introduce a stiffening effect in the aggregate network, because the generators are aggregated at the generator terminal buses. Viewing the power system as a multiple mass-spring system, the stiffening effect increases the oscillatory (swing) frequencies in the aggregate network. The inertia aggregation and slow coherency methods will have smaller stiffening effects and hence, are more preferable.

The function is implemented in the M-file podmore in the POWER SYSTEM TOOLBOX.

See also: i_agg, s_coh3, eqgen

Reference:

1. R. Podmore, "Development of Dynamic Equivalents for Transient Stability Studies," EPRI Report EL-456, 1977.

Function:

reduce

Purpose:

Eliminates static load buses in a power network.

Synopsis:

[bus_red,line_red] = reduce(bus,line,bus_list,flag,tol)

Description:

[bus_red,line_red] = reduce(bus,line,bus_list,flag,tol) eliminates static load buses in a power network. The full order system bus and line data are contained in the arrays bus and line. The array bus must contain a solved loadflow of the system. If flag = 0, the function eliminates the buses contained in the vector bus_list; otherwise, if flag = 1, the function retains the buses contained in bus_list. The input variable tol is the tolerance. A branch is set to zero if its admittance is less than tol. The default value of tol is set at 10^{-8}.

The output arrays bus_red and line_red contain the bus and line data, respectively, of the reduced network. The bus_red data will also contain a solved loadflow of the reduced system.

Algorithm:

The function retains all the generator buses, even if such a bus is contained in bus_list. Generations and loads on the static load buses to be eliminated will be first converted to constant impedance loads. Then an admittance matrix is constructed and a Kron reduction is performed to eliminate the desired buses. The reduced network is reconstructed from the reduced admittance matrix. Entries in the admittance matrix smaller than the tolerance tol will be set to zero. The reconstruction is straightforward if no phase shifter is involved. For a network with phase shifters, such as that obtained from the aggregation process, the reconstruction of an off-diagonal entry of a non-symmetrical admittance matrix may require a two-branch reconstruction.

The function is implemented in the M-file reduce in the POWER SYSTEM TOOLBOX.

See also: podmore, i_agg, s_coh3

Function:

s_coh3

Purpose:

Aggregate the coherent machines using the slow coherency method.

Synopsis:

[nbus,nline,nmac_con] = s_coh3(bus,line,area,narea)

Description:

[nbus,nline,nmac_con] = s_coh3(bus,line,area,narea) aggregates the coherent machines contained in each row of the array area onto a new aggregate bus. The bus and line data of the full order system are contained in the arrays bus and line. The bus array must contain a solved loadflow of the system. The vector narea contains the number of machines in each row of area. The function s_coh3 also requires the machine data in mac_con which is a global variable.

On output, nbus, nline, and nmac_con contain the aggregated bus, line and machine data, respectively. If the input bus data contain a solved loadflow, nbus will also contain a solved loadflow.

Algorithm:

In the slow coherency algorithm [1], a linearized electromechanical model is constructed and a first order singular perturbation correction is added to the linearized reduced order model. Thus the linearized model further reduces by 50% the stiffening effect compared to the inertia aggregate model. However, the linearized model cannot be reconstructed exactly to a power network. As a result, some of accuracy occurs in the network reconstruction stage. Similar to the function i_agg, the original generator terminal buses are preserved. The coherent machines are aggregated into a single aggregate machine by paralleling the machine impedances and averaging the time constants according to the machine inertias.

The function is implemented in the M-file s_coh3 in the POWER SYSTEM TOOLBOX.

See also: podmore, i_agg, eqgen

Reference:

1. J. H. Chow, R. A. Date, H. Othman and W. W. Price, "Slow Coherency Aggregation of Large Power Systems," in *Eigenanalysis and Frequency Domain Methods for System Dynamic Performance,* IEEE Publication 90TH0292-3- PWR, 1990.

Function:

sign_coh

Purpose:

Find the coherent areas of a power system using the sign coherency algorithm.

Synopsis:

[area,narea] = sign_coh(V)

Description:

[area,narea] = sign_coh(V) uses the eigenvector matrix V to determine the slow coherent areas in a power system. The matrix V can be obtained from the function V. In the sign coherency grouping algorithm, two machines are coherent if and only if their mode shapes in all the modes in V have the same signs. In general, the number of areas found from this algorithm will not be equal to the number of columns of V. The number of coherent areas is given by the number of rows in the output matrix area. The machines in the ith coherent area are contained in the ith row of area and the number of the machines in the ith area is the ith entry of the vector narea.

Examples:

Let V be the eigenvector matrix corresponding to the 9 slowest eigenvalues of the NPCC 48 machine system [1].

[area,narea] = sign_coh(V) returns

area =

1	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0
3	4	5	6	7	8	9	0
10	0	0	0	0	0	0	0
11	12	0	0	0	0	0	0
13	14	24	26	0	0	0	0
15	0	0	0	0	0	0	0
16	17	18	19	20	21	22	23
25	0	0	0	0	0	0	0
27	28	29	30	0	0	0	0
31	0	0	0	0	0	0	0
32	37	38	40	0	0	0	0
33	0	0	0	0	0	0	0
34	35	0	0	0	0	0	0
36	0	0	0	0	0	0	0
39	0	0	0	0	0	0	0
41	0	0	0	0	0	0	0
42	0	0	0	0	0	0	0
43	0	0	0	0	0	0	0
44	46	0	0	0	0	0	0
45	47	0	0	0	0	0	0
48	0	0	0	0	0	0	0

narea' =

1 1 7 1 2 4 1 8 1 4 1 4 1 2 1 1 1 1 1 2 2 1

Algorithm:

The sign coherency algorithm is based on a coherency criterion proposed in [2]. From V a sign eigenvector matrix Vsg matrix is established as follows:

If $\mathsf{V}(i,j) > 0$ then $\mathsf{Vsg}(i,j) = 1$, else if $\mathsf{V}(i,j) < 0$ then $\mathsf{Vsg}(i,j) = -1$.

Then machines i and j are coherent if the inner product of the ith and jth rows of Vsg is equal to n_s, the column dimension of V.

The function is implemented in the M-file sign_coh in the POWER SYSTEM TOOLBOX.

See also: coh_map,L_group,ex_group

Reference:

1. J. H. Chow, editor, *Time-Scale Modeling of Dynamic Networks with Applications to Power Systems*, Springer-Verlag, Berlin, 1982.

2. B. Eliasson, *Damping of Power Oscillations in Large Power Systems,* Ph.D. Thesis, Lund Institute, 1990.

Function:

union

Purpose:

Form the union of two sets of integers.

Synopsis:

[v] = union(v1,v2)

Description:

[v] = element(v1,v2) returns the union of the integers contained in the vectors v1 and v2.

Algorithm:

The function is implemented in the M-file union in the POWER SYSTEM TOOLBOX.

See also: ex_group, element

Function:

V_slow

Purpose:

Form the slow eigenvector matrix.

Synopsis:

[s,V] = V_slow(A,n)

Description:

[s,V] = V_slow(A,n) returns the n smallest in magnitude eigenvalues of the matrix A in the output vector s. The corresponding eigenvectors are contained in the output matrix V.

This function is used in finding the slow coherent machines in a large power system. The input matrix is $A = M^{-1}K$ derived from the electromechanical model of the power system, where M is the diagonal matrix of machine inertias, and K is the matrix of linearized connection constants between the machines. These matrices can be obtained from a linearization of the power system model using the machine models available in the POWER SYSTEM TOOLBOX.

The output eigenvector matrix V can be used to find slow coherent machine groups using the function L_group or to generate a coherency map using the function coh_map.

Algorithm:

The function is implemented in the M-file V_slow in the POWER SYSTEM TOOLBOX.

See also: L_group,coh_map,ex_group

Function:

weaklink

Purpose:

Find the coherent areas of a power system using the weak coupling method of Lamba and Rao.

Synopsis:

[area,narea,list,S] = weaklink(MK)
[area,narea,areal,nareal,list,S] = weaklink(MK,tol)

Description:

[area,narea,list,S] = weaklink(MK) uses the coupling factor S of the linearized electromechanical model MK to find the weakly coupled areas in a power system. The matrix MK is the matrix $-M^{-1}K$ where M is the diagonal matrix of machine inertias and K is the matrix of synchronizing coefficients. MK can be obtained from the state matrix generation function svm_em. The algorithm first reorder the machines in order of their relative coupling. The reordered machines are given in list. The number of coherent areas is given by the number of rows in the output matrix area. The machines in the ith coherent area are contained in the ith row of area and the number of the machines in the ith area is the ith entry of the vector narea.

[area,narea,areal,nareal,list,S] = weaklink(MK,tol) also uses the tolerance tol to find the weakly coherent machines. The number of weakly coherent areas is given by the number of rows in the output matrix areal. The machines in the ith coherent area are contained in the ith row of areal and the number of the machines in the ith area is the ith entry of the vector nareal.

Examples:

Let MK be the linearized matrix corresponding of the NPCC 48 machine system [1].

[area] = weaklink(MK) returns

area =

39	0	0
42	41	40
38	0	0
37	32	0
34	35	33
36	31	0
17	29	0
16	19	0
20	18	22
21	0	0
15	23	0
13	0	0
14	26	0
25	24	0
10	12	0
11	8	0
9	1	0
2	27	0
30	7	0
20	6	0
3	4	0
5	48	0
43	44	47
45	46	0

Algorithm:

The weak coupling method is based on the algorithm proposed in [2]. The method uses the coupling factors S to reorder the machines according to their relative coupling. The change in the slope of S is use to identify the strongly coherent machines and the weakly coherent machines.

The function is implemented in the M-file weaklink in the POWER SYSTEM TOOLBOX.

See also: coh_map,L_group,ex_group,sign_coh,zabors

Reference:

1. J. H. Chow, editor, *Time-Scale Modeling of Dynamic Networks with Applications to Power Systems*, Springer-Verlag, Berlin, 1982.

2. R. Nath, S. S. Lamba and K. S. P. Rao, "Coherency Based System Decomposition into Study and External Areas using Weak Coupling," *IEEE Trans. Power Apparatus and Systems,* vol. PAS-104, pp. 1443-1449, 1985.

Function:

zabors

Purpose:

Find the coherent areas of a power system using the cluster algorithm of John Zaborszky.

Synopsis:

[area,narea] = zabors(MK,tol)

Description:

[area,narea] = zabors(MK,tol) uses the connection strength of the linearized electromechanical model to find the slow coherent areas in a power system. The matrix MK is the matrix $-M^{-1}K$ where M is the diagonal matrix of machine inertias and K is the matrix of synchronizing coefficients. MK can be obtained from the state matrix generation function svm_em. The Zaborszky clustering algorithm finds the boundaries of weak connections in a power system. The degree of weak connection is defined by the input tol, which is usually selected to be greater than 0 but less than 0.5. The function zabors will generate more areas if a higher tol is used. The number of coherent areas is given by the number of rows in the output matrix area. The machines in the ith coherent area are contained in the ith row of area and the number of the machines in the ith area is the ith entry of the vector narea.

Examples:

Let MK be the linearized matrix corresponding of the NPCC 48 machine system [1].

[area,narea] = zabors(MK,0.5) returns

area =

1	0	0	0	0	0
2	0	0	0	0	0
3	4	5	0	0	8
6	0	0	0	0	0
.	.	.	.	.	.
20	0	0	0	0	0
21	22	0	0	0	0
23	0	0	0	0	0
24	25	27	45	47	48
26	0	0	0	0	0
.	.	.	.	.	.
42	0	0	0	0	0
43	44	46	0	0	0

narea' =

1 1 3 1 . 1 2 1 4 1 . 1 3

Algorithm:

The clustering algorithm is based on the α-decomposition proposed in [2]. In the α-decomposition, the MK matrix is first normalized such that the entry largest in magnitude is 1. Then for each row of the normalized MK matrix, the entries smallest in magnitude that add to less than $\alpha =$ tol are st to zero. A reduced incidence matrix C is set up such that $C(i,j) = 1$ if the (i,j)th entry of the resulting MK matrix is nonzero. Furthermore, the diagonal entries of C are set to 1 and if $C(i,j) = 1$, $C(j,i) = 1$. The machines i and j are coherent if $C(i,j) = 1$.

The function is implemented in the M-file zabors in the POWER SYSTEM TOOLBOX.

See also: coh_map,L_group,ex_group,sign_coh

Reference:

1. J. H. Chow, editor, *Time-Scale Modeling of Dynamic Networks with Applications to Power Systems*, Springer-Verlag, Berlin, 1982.

2. J. Zaborszky *et al*, "A Clustered Dynamic Model of a Class of Linear Autonomous Systems Using Simple Enumerative Sorting," *IEEE Trans. Circuits and Systems,* vol. CAS-29, pp. 747-758, 1982.