Mathematica Module for Graph Laplacians

MMGL User's Manual
Version 1.2
30 June 2013

by Yoshihiro MIZOGUCHI, Kyushu University.

`Ver. 0.1` 20 December 2010

`Ver. 1.0` 8 June 2012

`Ver. 1.1` 30 June 2013

Table of Contents

1 Introduction

This is a Mathematica Module for analysing graphs, especially using Laplacian Matrices of graphs. This module depends on the Mathematica modules "Combinatorica`" and "ComputationalGeometry`".

To use this package "GraphLaplacian`", users should set a directory where the modules is stored.

```
[Example]

SetDirectory[FileNameJoin[$HomeDirectory, "--- Some Folder ---"]];
<< GraphLaplacian`;
```

This module was used and introduced in the followings:

[1] K.K.K.R. Perera, Y. Mizoguchi, Bipartition of graphs based on the normalized cut and spectral methods, Part I: Minimum normalized cut, Journal of Math-for-industory, Vol.5(2013A-8),pp.59-72.

[2] Y. Mizoguchi, Mathematical Aspects of Interpolation Technique for Computer Graphics, Forum "Math-for-Industry" 2012, Information Recovery and Discovery, 22 October 2022. http://fmi2012.imi.kyushu-u.ac.jp/

[3] Mathematica Module for Graph Laplacians
https://github.com/ymizoguchi/MathematicaGraphLaplacian.git

2 Graph Partitioning

2.1 Fundamental Functions

2.1.1 `Distance`

`Distance[`*x1*`,`*x2*`]`
:: Distance between vertices *x1* and *x2*.

x1, *x2* vertices (2D vectors)

return length

```
[Example]
Distance[{3,0},{0,4}]
5
```

2.1.2 `DistanceVector`

`DistanceVector[`*x1*`,`*x2*`]`
:: Distance between vertices *x1* and *x2*.

x1, *x2* vertices (vectors)

return length

```
[Example]
Distance[{3,0,0},{0,0,4}]
5
```

2.1.3 `LabeledFindClusters`

`LabeledFindClusters[`*set*`,`*n*`]`
:: Divide *set* into *n* clusters using indices.

set set (list)

return clusterd index set (list of list)

```
[Example 1]
LabeledFindClusters[{1, 2, 3, 8, 9, 10}, 2]
{{1, 2, 3}, {4, 5, 6}}

[Example 2]
LabeledFindClusters[{1, 8, 2, 9, 3, 10}, 2]
{{1, 3, 5}, {2, 4, 6}}
```

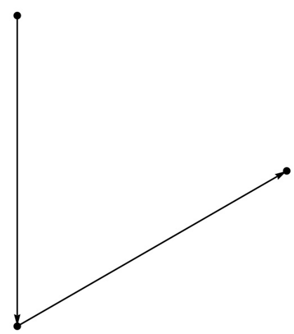
2.1.4 `DirectedtoUndirected`

`DirectedToUndirected[`*graph*`]`
:: Translate a directed graph (*graph*) into an undireced graph.

graph directed graph (Graph)

return undirected graph

```
[Example]
ToOrderedPairs[
 DirectedToUndirected[FromOrderedPairs[{{1, 2}, {2, 3}}]]]
{{2, 1}, {3, 2}, {1, 2}, {2, 3}}
```



2.1.5 DelaunayEdges

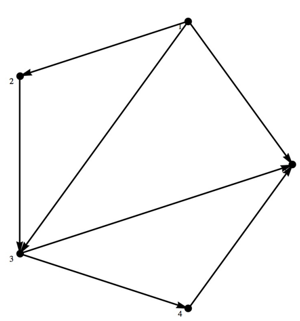
`DelaunayEdges[`*pl*`]`

:: list of edges which construct a Delaunay triangulation of given points *pl*

pl list of points

return list of edges which construct a Delaunay triangulation of given points *pl*

```
[Example]
DelaunayEdges[Vertices[Cycle[5]]]
{{1, 2}, {1, 3}, {1, 5}, {2, 3}, {3, 4}, {3, 5}, {4, 5}}
```



2.1.6 DelaunayGraph

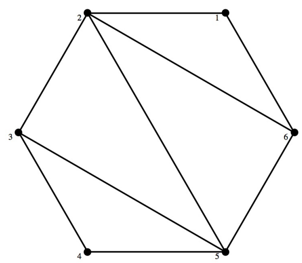
`DelaunayGraph[`*pl*`]`

:: construct a graph using Delaunay triangulation of given points *pl*

pl list of points

return graph using Delaunay triangulation of given points *pl*

```
[Example]
ShowLabeledGraph[DelaunayGraph[Vertices[Cycle[6]]]]
```



2.1.7 `CreateGraph`

`CreateGraph[`*vl*`,`*el*`]`

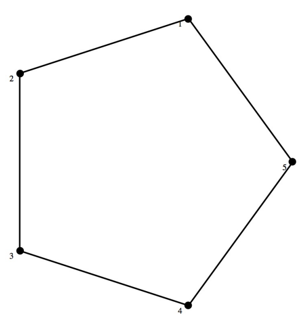
:: constract a graph using a list of coordinate of vertices *vl* and edges *el*

vl list of coordinate of vertices

el list of connected edges (pairs of vertices)

return graph

```
[Example]
ShowLabeledGraph[CreateGraph[Vertices[Cycle[5]], Edges[Cycle[5]]]]
```



2.1.8 `NVertices`

`NVertices[`*n*`]`

:: list of n random coordinates

n number of coordinates

return list of n random coordinates

```
[Example]
NVertices[5]
{{0.702154, 0.314688}, {0.214506, -0.316029}, {0.121768, -0.0316586},
{-0.0175943, -0.198242}, {0.144589, 0.577006}}
```

2.1.9 `SetVertices`

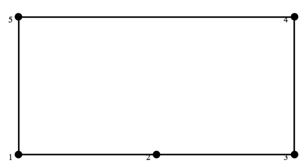
`SetVertices[`*g*`,`*v*`]`
:: Force to be coordinates of vertices *v* in a graph *g*

g graph

v list of coordinates of vertices

return graph

```
[Example]
ShowLabeledGraph[SetVertices[Cycle[5],
                 {{0, 0}, {1, 0}, {2, 0}, {2, 1}, {0, 1}}]]
```



2.1.10 `NRandomGraph`

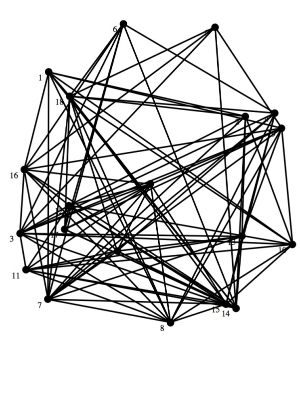
`NRandomGraph[`*n*`,`*p*`]`
:: Construct a $varn vertex graph adding random edges with a proberbility *p*.

n number of vertices

p existing proberbility of edges

return $varn vertex graph adding random edges with a proberbility *p*.

```
[Example]
ShowLabeledGraph[NRandomGraph[20, 0.5]]
```



2.1.11 `CycleVertices`

`CycleVertices[`*n*`,`*s*`]`
:: list of coordinates of *n* vertices rotated *s* radian.

n number of coordinates

s rotation (radian)

return list of coordinates of n vertices rotated s radian.

$$(\cos\left(\frac{2k\pi}{n}+s\right),\sin\left(\frac{2k\pi}{n}+s\right))\quad(k=1,\ldots,n)$$

```
[Example]
CycleVertices[4, Pi/2]
{{-1, 0}, {0, -1}, {1, 0}, {0, 1}}
```

2.1.12 CycledGraph

`CycledGraph[n,s]`

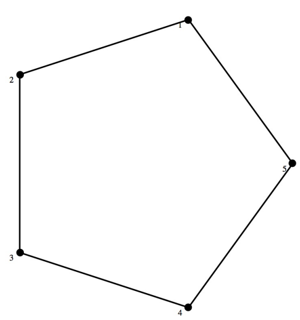
:: Cycle graph which vertices are roted s radian

n number of vertices

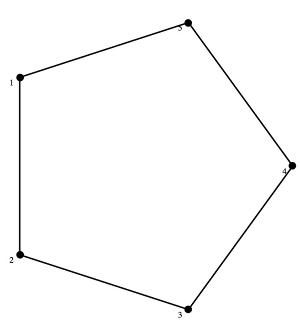
s rotation (radian)

return graph

```
[Example]
ShowLabeledGraph[Cycle[5]]
```




```
ShowLabeledGraph[CycledGraph[5, 2 Pi/5]]
```



2.2 Normalized Cut

2.2.1 DS

`DS[`*s*`,`*g*`]` :: Set of edges between a vertex set *s* and its complement.

s a subset of the vertex set of a graph *g*

g a graph

return set of edges between *s* and the complement of *s*

```
[Example]
DS[{1,2},Cycle[4]]
{{2,3},{1,4}}
```

2.2.2 FDS

`FDS[`*s*`,`*g*`]` :: Transition probability between a vertex set *s* and its complement.

s a subset of the vertex set of a graph *g*

g a graph

return Transition probability between *s* and the complement of *s*

```
[Example 1]
FDS[{1,2},Cycle[4]]
1/4

[Example 2]
FDS[{1,2},CompleteGraph[4]]
1/3
```

2.2.3 WnCut

`WnCut[`*s*`,`*g*`]`
:: Normalized cut value of a vertex set *s* of a graph *g*.

s a subset of the vertex set of a graph *g*

g a graph

return Normalized cut value of a vertex set *s* of a graph *g*.

```
[Example 1]
WnCut[{1,2},Cycle[4]]
1

[Example 2]
WnCut[{1,2},CompleteGraph[4]]
4/3
```

2.2.4 FindMinimumWnCut

`FindMinimumWnCut[`*g*`]`
:: Find a vertex set which minimize normalized cut values.

g a graph

return list of normalized cut values for all subsets of the vertex set of *g*\\ the list is sorted by the order of normalized cut values

`FindMinimumWnCut[g,l]`
:: Find a vertex set in *l* which minimize normalized cut values

g a graph

l list of vertex sets

return list of normalized cut values for subsets in *l*\\ the list is sorted by the order of normalized cut values

```
[Example 1]
FindMinimumWnCut[Cycle[4]]
{{1., {1, 2}}, {1., {1, 4}},{1., {2, 3}}, {1., {3,4}},
{1.33333, {1}}, {1.33333, {2}},{1.33333, {3}}, {1.33333, {4}},
{1.33333, {1, 2, 3}},{1.33333, {1, 2, 4}},{1.33333, {1, 3, 4}},
{1.33333, {2, 3, 4}},{2., {1, 3}}, {2., {2, 4}}}

[Example 2]
FindMinimumWnCut[Cycle[4], {{1}, {1, 2}, {1, 2, 3}, {1, 2, 3, 4}}]
{{1., {1, 2}}, {1.33333, {1}}, {1.33333, {1, 2, 3}}}
```

2.2.5 GDegree

`GDegree[g,x]`
:: degree of a vertex *x* of a graph *g*

g graph

x vertex

return degree of a vertex *x* of a graph *g*

```
[Example]
Degree[Cycle[4], 1]
2
```

2.2.6 GVol

`GVol[g,s]`
:: volume of a vertex subset *s* of a graph *g*

g graph

s subset of the vertex set of a graph *g*

return volume of a vertex subset *s* of a graph *g*

`GVol[g]` :: volume of a vertex subset *s* of a graph *g*

g graph

return volume of the all vertex subset of a graph *g*

```
[Example 1]
GVol[Cycle[4], 1, 2]
4

[Example 2]
GVol[Cycle[4]]
8
```

2.2.7 HG

`HG[s,g]` :: HG cut value of a vertex set *s* of a graph *g*.

s subset of the vertex set of a graph *g*

g graph

return HG cut value of a vertex set *s* of a graph *g*.

```
[Example 1]
HG[{1,2},Cycle[4]]
1/2

[Example 2]
HG[{1,2},CompleteGraph[4]]
2/3
```

2.2.8 FindMinimumHG

`FindMinimumHG[g]`
:: Find a vertex set which minimize HG cut values.

g graph

return list of HG cut values for all subsets of the vertex set of *g*\\ the list is sorted by the order of normalized cut values

```
[Example]
FindMinimumHG[Cycle[4]]
{{0.5, {1, 2}}, {0.5, {1, 4}}, {0.5, {2, 3}},
{0.5, {3, 4}}, {1., {1}}, {1., {2}}, {1., {3}},
{1., {4}}, {1., {1, 3}}, {1., {2, 4}}, {1., {1, 2, 3}},
{1., {1, 2, 4}}, {1., {1, 3, 4}}, {1., {2, 3, 4}}}
```

2.2.9 Ncut

`Ncut[s,g]`
:: Normalized cut value of a vertex set *s* of a graph *g*.

s subset of the vertex set of a graph *g*

g graph

return Normalized cut value of a vertex set *s* of a graph *g*.

```
[Example]
Ncut[{1},Cycle[4]]
4/3
```

2.2.10 FindMinimumNcut

FindMinimumNcut[*g*]
:: Find a vertex set which minimize HG cut values.

g graph

return list of normalized cut values for all subsets of the vertex set of g\\ the list is sorted by the order of normalized cut values

```
[Example]
FindMinimumNcut[Cycle[4]]
{{1., {1, 2}}, {1., {1, 4}}, {1., {2, 3}}, {1., {3, 4}},
{1.33333, {1}}, {1.33333, {2}}, {1.33333, {3}}, {1.33333, {4}},
{1.33333, {1, 2, 3}}, {1.33333, {1, 2, 4}}, {1.33333, {1, 3, 4}},
{1.33333, {2, 3, 4}}, {2., {1, 3}}, {2., {2, 4}}}
```

2.3 Matrix Operations

2.3.1 TruncateMatrix

TruncateMatrix[*m*,*n*]
:: Force to be zero row vectors except the row n in a matrix m

m matrix

n row number

return Force to be zero row vectors except the row n in a matrix m

```
[Example]
TruncateMatrix[{{1, 2, 3}, {4, 5, 6}, {7, 8, 9}}, 2]
{{0, 0, 0}, {4, 5, 6}, {0, 0, 0}}
```

2.3.2 TruncateUptoMatrix

TruncateUptoMatrix
:: Force to be zero row vectors except the row up to n in a matrix m

m matrix

n row number

return Force to be zero row vectors except the row up to n in a matrix m

```
[Example]
TruncateUptoMatrix[{{1, 2, 3}, {4, 5, 6}, {7, 8, 9}}, 2]
{{1, 2, 3}, {4, 5, 6}, {0, 0, 0}}
```

2.3.3 NMatrixPower

NMatrixPower[*A*,*t*]
:: t-th power of a matrix A

A matrix

t real number

return t-th power of a matrix A

It is computed using Typer expansion.

```
[Example]
NMatrixPower[{{1, -1}, {1, 1}}, 3]
{{-2., -2.}, {2., -2.}}
```

2.3.4 MatrixT

`MatrixTA,t]`
:: t-th power of a matrix A

A matrix

t real number

return t-th power of a matrix A

It is computed using the diagonalization of a matrix.

```
[Example]
MatrixT[{{1, -1}, {1, 1}}, 3]
{{-2., -2.}, {2., -2.}}
```

2.3.5 Reordering

`Reordering[S,T]`
:: Arrange the row vectors of T to maximize inner products to the corresponding row vectors of S

S, *T* matrices

return arranged matrix

2.3.6 TransposeReordering

`TransposeReordering[S,T]`
:: Arrange the row vectors of T to maximize inner products to the corresponding row vectors of S

S, *T* matrices

return arranged matrix

2.4 Show Graphs

2.4.1 ColoringVertex

`ColoringVertex[l]`
:: Create an option formula for coloring vertices

l clustered list

return an option formula for coloring vertices

```
[Example]
ColoringVertex[{{1, 2, 3}, {4, 5}}]
{{1, 2, 3, VertexColor -> RGBColor[1, 0, 0]},
{4, 5, VertexColor -> RGBColor[0, 0, 1]}}
```

2.4.2 Coloring

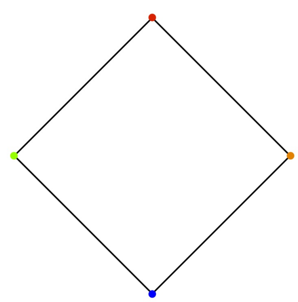
`Coloring[`*g*`]`

:: Coloring vertices of a graph *g*

g graph

return colored graph

```
[Example]
ShowGraph[Coloring[Cycle[4]]]
```



2.4.3 ClusterNumber

`ClusterNumber[`*n*`,`*cl*`]`

:: rerutn the position number of the cluster which includes \varn

n element

cl cluster list

return the position number of the cluster which includes \varn

```
[Example]
ClusterNumber[3, 1, 2, 3, 4, 5, 6]
2
```

2.4.4 ShowColoredGraphs

`ShowColoredGraphs[`*gl*`,`*cl*`]`

:: Show colored graphs in *gl* using a list *cl*

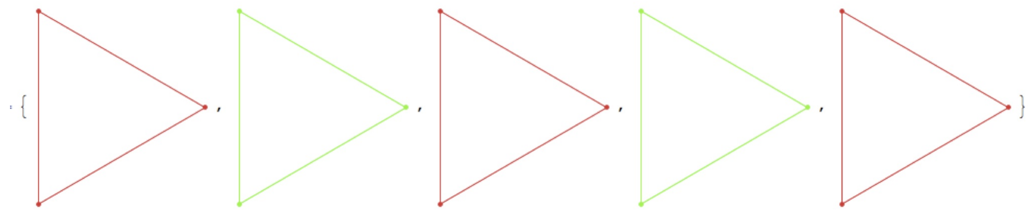
gl list of graphs

cl clustered numbers

return Show colored graphs in *gl* using a list *cl*

If a graph is in n-th cluster then it is colored by n-th color. The color list is (Red, Green, Orange, Cyan, Purple, Black).

```
[Example]
ShowColoredGraphs[Table[Cycle[3], {5}], {{1, 3, 5}, {2, 4}}]
```



2.4.5 ColoringSubset

`ColoringSubset[g,a]`

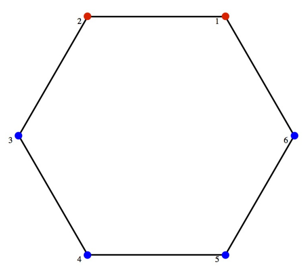
:: Coloring vertices in a subset a of the vertex set of a graph g.

g graph

a subset of the vertex set of a graph g

return Colored graph

```
[Example]
ShowLabeledGraph[ColoringSubset[Cycle[6], {1, 2}]]
```



2.5 Random Walks

2.5.1 NaturalRandomWalkMatrix

`NaturalRandomWalkMatrix[g]`

:: Natural random walk matrix of a graph g

graph

return natural random walk matrix of a graph g

```
[Example]
NaturalRandomWalkMatrix[Cycle[5]]
{{0, 1/2, 0, 0, 1/2}, {1/2, 0, 1/2, 0, 0}, {0, 1/2, 0, 1/2, 0},
{0, 0, 1/2, 0, 1/2}, {1/2, 0, 0, 1/2, 0}}
```

2.5.2 MyStationaryDistribution

`MyStationaryDistribution[g]`
:: Stationary distribution of the natural random walk of a graph *g*

g graph

return stationary distribution of the natural random walk of a graph *g*

```
[Example]
MyStationaryDistribution[Path[5]]
{1/8, 1/4, 1/4, 1/4, 1/8}
```

2.5.3 FS

`FS[s,g]` :: Sum of stationary ditribution probabilities for a subset *g*

s subset of the vertex set of a graph *g*

g argtype

return sum of stationary ditribution probabilities for a subset *g*

```
[Example]
FS[1, 2, Path[5]]
3/8
```

2.6 Spectral Clustering

2.6.1 FirstEigenVector

`FirstEigenVector[M]`
:: The first eigen vector of a matrix *m*.

M matrix

return The first eigen vector of a matrix *m*.

2.6.2 SecondSmallEigenVector

`SecondSmallEigenVector[M]`
:: The second smallest eigen vector of a matrix *m*.

M matrix

return The second smallest eigen vector of a matrix *m*.

2.6.3 ThirdSmallEigenVector

`ThirdSmallEigenVector[M]`
:: The third smallest eigen vector of a matrix *m*.

M matrix

return The third smallest eigen vector of a matrix *m*.

2.6.4 `UndirectedLaplacian`

`UndirectedLaplacian[`*g*`]`
:: Laplacian matrix of the adjacency matrix of a graph *g*

g graph

return Laplacian matrix of the adjacency matrix of a graph *g*

```
[Example]
UndirectedLaplacian[Path[3]]
{{1, -(1/Sqrt[2]), 0}, {-(1/Sqrt[2]), 1, -(1/Sqrt[2])}, {0, -(1/Sqrt[2]), 1}}
```

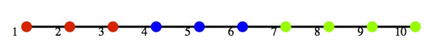
2.6.5 `NormalClustering`

`NormalClustering[`*g*`,`*n*`]`
:: Clustering using a build in Mahtematica function.

g graph

return Clustered colored graph using a build in Mahtematica function.

```
[Example]
ShowLabeledGraph[NormalClustering[Path[10], 3]]
```



2.6.6 `UndirectedSpectralVector`

`UndirectedSpectralVector[`*g*`]`
:: A second eigenvector of the Laplacian matrix of a graph *g*.

g graph

return A second eigenvector of the Laplacian matrix of a graph *g*.

2.6.7 `UndirectedSpectralClustering`

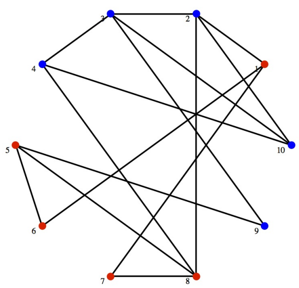
`UndirectedSpectralClustering[`*g*`,`*n*`]`
:: Spectoral clustering of a graph *g*

g graph

n number of clusters

return colored graph clustered by spectral clustering method.

```
[Example]
ShowLabeledGraph[UndirectedSpectralClustering[ExactRandomGraph[10, 15], 2]]
```



2.6.8 UndirectedSpectralVector2

`UndirectedSpectralVector2[`*g*`]`

:: A pair of a second eigen vector and a third eigen vector of the Laplacian matrix of a graph *g*.

g graph

return pair of a second eigen vector and a third eigen vector of the Laplacian matrix of a graph *g*

2.6.9 UndirectedSpectralClustering2

`UndirectedSpectralClustering2`

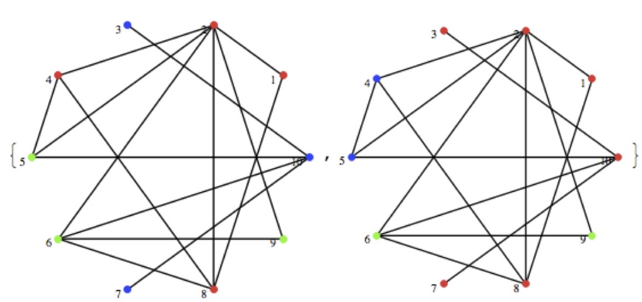
:: Spectoral clustering of a graph *g*

g graph

return spectoral clustering of a graph *g*

This method use a second and third eigen vectors of the Lapalacian matrix of a graph *g*.

```
g = ExactRandomGraph[10, 15];
{ShowLabeledGraph[UndirectedSpectralClustering[g, 3]],
 ShowLabeledGraph[UndirectedSpectralClustering2[g, 3]]}
```



2.6.10 UndirectedSpectralClusteringPlus

UndirectedSpectralClusteringPlus[*g*]
:: Spectoral clustering of a graph *g*

g graph

return return value

This method use a sorted second eigen vector of the Lapalacian matrix of a graph *g*.

2.6.11 UndirectedSpectralClusteringSign

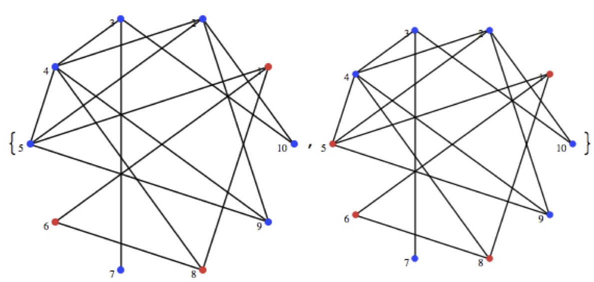
UndirectedSpectralClusteringSign[*g*]
:: Spectoral clustering of a graph *g*

g graph

return return value

This method use a sign of a second eigen vector of the Lapalacian matrix of a graph *g*.

```
[Example]
g = ExactRandomGraph[10, 15];
{ShowLabeledGraph[UndirectedSpectralClusteringPlus[g]],
 ShowLabeledGraph[UndirectedSpectralClusteringSign[g]]}
```



2.6.12 PCA3Clustering

PCA3Clustering[*m*,*n*]
:: Clustering data *m* to *n* clusters using PCA methods

m data vectors

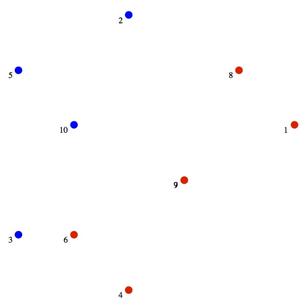
n number of clusters

return Clustering data *m* to *n* clusters using PCA methods

```
[Example]
gv = Table[RandomInteger[5], RandomInteger[5], 10]
{{5, 3}, {2, 5}, {0, 1}, {2, 0}, {0, 4},
 {1, 1}, {3, 2}, {4, 4}, {3, 2}, {1, 3}}

ShowLabeledGraph[SetGraphOptions[CreateGraph[gv, {}],
```

```
ColoringVertex[PCA3Clustering[gv, 2]]]]
```



3 Special Graphs

3.1 Roach Graph and Weighted Path

3.1.1 `RoachGraph`

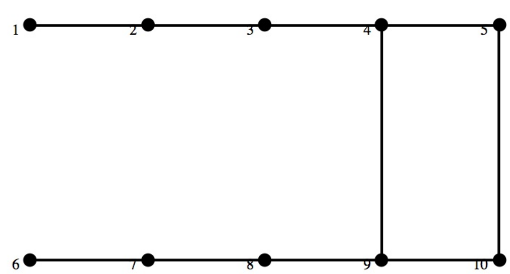
`RoachGraph[`*n*`,`*k*`]`

:: Roach type Graph with size *n* and *k*

n, *k* size

return Roach type graph

```
[Example]
ShowLabeledGraph[RoachGraph[3, 2]]
```



3.1.2 `WeightedNormalizedLaplacian`

`WeightedNormalizedLaplacian[`*M*`]`

:: Weighted Laplacian matrix of a weighted adjacency matrix.

M weighted adjacency matrix of a graph

return Weighted Laplacian matrix of a weighted adjacency matrix.

3.1.3 `WeightedPath`

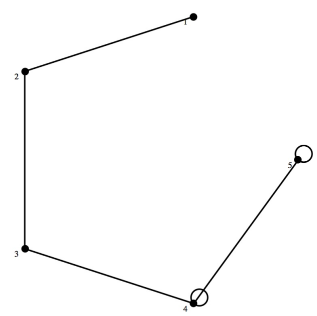
`WeightedPath[`*n*`,`*k*`]`

:: Weighted adjacency matrix of a path which have $n+k$ vertices and *n* vertices have weight 2 and *k* vertices have weight 1.

n, *k* size

return weighted adjacency matrix of a path

```
ShowLabeledGraph[FromAdjacencyMatrix[WeightedPath[3, 2]]]
```



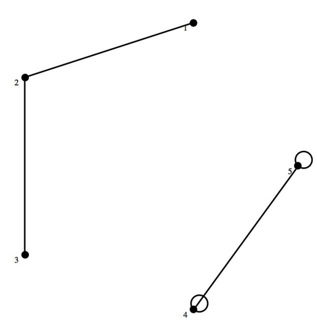
3.1.4 WeightedPathUnion

`WeightedPathUnion[n,k]`
:: Disjoint union graph of Path[*n*] and Path[*k*]

n, *k* size

return disjoint union graph of Path[*n*] and Path[*k*]

```
[Example]
ShowLabeledGraph[FromAdjacencyMatrix[WeightedPathUnion[3, 2]]]
```



3.2 Lollipop, Tree, Tree cross Path

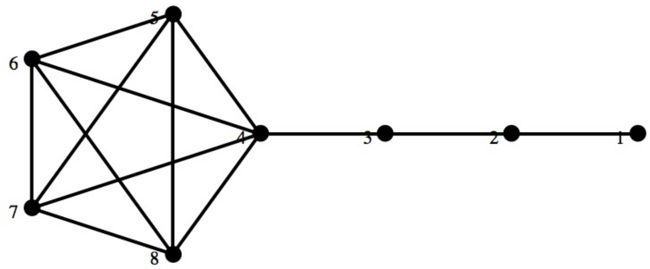
3.2.1 LPG

`LPG[n,m]` :: Lollipop graph with size *n* and *m*.

n, *m* size

return Lollipop graph with size *n* and *m*.

```
[Example]
ShowLabeledGraph[LPG[5, 3]]
```



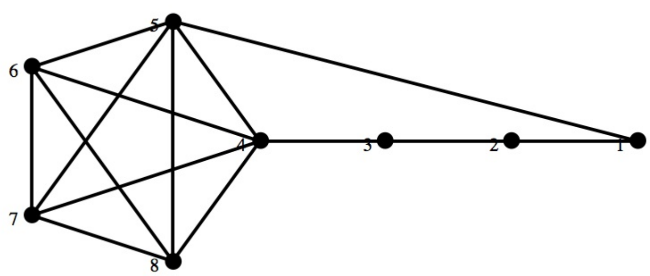
3.2.2 LPG2

`LPG2[n,m]`
:: Modified Lollipop graph with size n and m.

n, m size

return Modified Lollipop graph with size n and m.

```
[Example]
ShowLabeledGraph[LPG2[5, 3]]
```



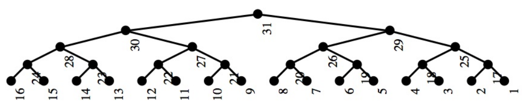
3.2.3 DTG

`DTG[n]` :: Tree graph with depth n

n depth

return Tree graph with depth n

```
[Example]
DTG[5]
```



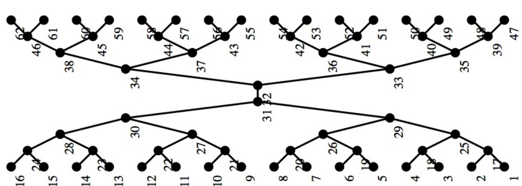
3.2.4 DTG2

`DTG2[n]` :: Double tree graph with depth *n*

n depth

return Double tree graph with depth *n*

```
[Example]
DTG2[5]
```



3.2.5 DTCPG

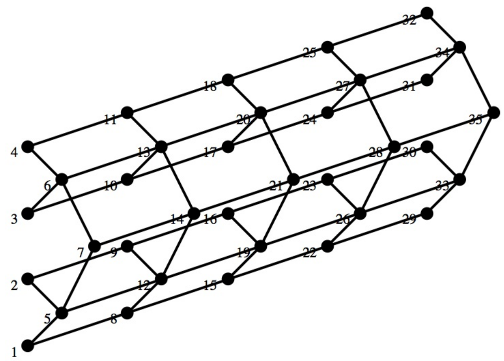
`DTCPG[n,k]`
:: Tree (depth *n*) cross Path (length *k*) Graph

n depth of tree

k length of path

return Tree (depth *n*) cross Path (length *k*) Graph

```
[Example]
DTCPG[3,5]
```



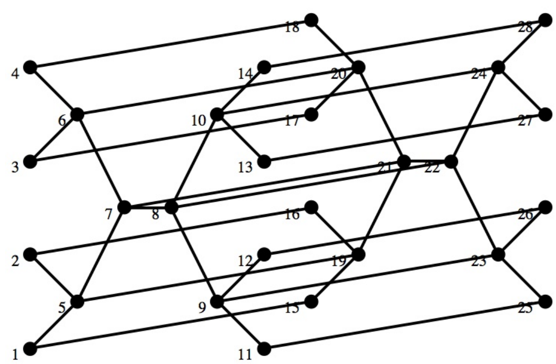
3.2.6 DTCPG2

`DTCPG2[n,k]`

:: Doubl tree (depth *n*) cross Path (length *k*) Graph

n	depth of double tree
k	length of path
return	Double tree (depth *n*) cross Path (length *k*) Graph

```
[Example]
DTCPG2[3,2]
```



Index