SYSG

Generado por Doxygen 1.8.12

Índice general

Capítulo 1

Índice de clases

1.1. Lista de clases

Lista de las clases, estructuras, uniones e interfaces con una breve descripción:

Capítulo 2

Documentación de las clases

2.1. Referencia de la plantilla de la Clase Add< L, R >

Métodos públicos

- **Add** (L const &l, R const &r)
- double **operator[ ]** (int n) const

La documentación para esta clase fue generada a partir del siguiente fichero:

- OperVect.hpp

2.2. Referencia de la Clase BICGSTAB

Métodos públicos

- template<class T >
  void **solve** (T &A, Vector &x, Vector &b)
- template<class T , class U >
  void **solve** (T &A, Vector &x, Vector &b, U &M)
- int **its** ()
- double **error** ()
- void **maxIts** (int it)
- void **tol** (double t)
- void **report** ()

La documentación para esta clase fue generada a partir del siguiente fichero:

- BICGSTAB.hpp

2.3. Referencia de la Clase CG

Métodos públicos

- template<class T >
  void **solve** (T &A, Vector &x, Vector &b)
- template<class T , class U >
  void **solve** (T &A, Vector &x, Vector &b, U &M)
- int **its** ()
- double **error** ()
- void **maxIts** (int it)
- void **tol** (double t)
- void **report** ()

La documentación para esta clase fue generada a partir del siguiente fichero:

- CG.hpp

2.4. Referencia de la Clase COO

Métodos públicos

- **COO** (int m)
- void **zeros** ()
- void **insert** (int i, int j, double val)
- void **change** (int i, int j, double val)
- double **gValue** (int i, int j)
- void **saveData** (std::string filename)
- void **impMatrix** ()
- void **impMatrixE** ()

Atributos públicos

- double * **data**
- int * **row**
- int * **col**
- int **nnz**
- int **nnzMax**
- int **n**

La documentación para esta clase fue generada a partir de los siguientes ficheros:

- COO.hpp
- COO.cpp

2.5. Referencia de la Clase CSR

Métodos públicos

- void **reserveMemory** ()
- void **freeMemory** ()
- **CSR** (int m)
- **CSR** (CSR const &Mtx)
- void **zeros** ()
- void **convert** (COO const &coo)
- int **binarySearch** (int lmin, int lmax, int j)
- double **search** (int i, int j)
- double **dat** (int i) const
- int **co** (int i) const
- int **iro** (int i) const
- void **iterJacobi** (int i, Vector &x)
- void **iterGaussSeidel** (int i, Vector &x, Vector const &b)
- void **operator=** (CSR const &Mtx)

Atributos públicos

- double ∗ **data**
- int ∗ **col**
- int ∗ **irow**
- int ∗ **idiag**
- int **nnzMax**
- int **nnz**
- int **n**

La documentación para esta clase fue generada a partir de los siguientes ficheros:

- CSR.hpp
- CSR.cpp

2.6. Referencia de la Clase CVFEM

Métodos públicos

- void **CI** ()
- void **bondaryCondition** (int index, int type, double value)
- void **typeNode** (Mesh &mesh)
- void **matrixCVFEM** (Mesh &mesh, CSR &A)
- void **matrixCVFEMD** (Mesh &mesh, CSR &A)
- void **matrixCVFEMW** (Mesh &mesh, CSR &A)
- void **sourceC** (Mesh &mesh, Vector &b)
- void **source** (Mesh &mesh, Vector &b)
- void **sourceP** (Mesh &mesh, Vector &f)
- void **sourcePP** (Mesh &mesh, Vector &b)
- void **solT** (Mesh &mesh, Vector &b, int n, double T)
- void **solP** (Mesh &mesh, Vector &b)

- void **semiCircle** (Mesh &mesh, Vector &b)
- void **solC** (Mesh &mesh, Vector &b, Vector &x)
- void **baseFunction** (double a, double b, double c, double x1, double x2, double aa)
- void **permeabilidad** (Mesh &mesh)
- double **permehar** (double f1, double f2)
- double **K11** (int node)
- double **K22** (int node)
- void **gnuplot** (Mesh &mesh, Vector &b, std::string name)
- void **gnuplotE** (Mesh &mesh, Vector &b, std::string name)
- void **showNodeType** ()
- void **showIncognite** ()
- void **showPermeabilidad** ()
- void **showCondition** (Mesh &mesh)
- int **nnodesX** ()
- std::vector< int > **nodeXX** (std::vector< int > inter, std::vector< int > neumman)

La documentación para esta clase fue generada a partir de los siguientes ficheros:

- CVFEM.hpp
- CVFEM.cpp

2.7. Referencia de la Clase GaussSeidel

Métodos públicos

- template<class T >
  void **solve** (T &A, Vector &x, Vector &b)
- int **its** ()
- double **error** ()
- void **maxIts** (int it)
- void **tole** (double t)
- void **report** ()

La documentación para esta clase fue generada a partir del siguiente fichero:

- GaussSeidel.hpp

2.8. Referencia de la plantilla de la Clase ICHOL< P >

Métodos públicos

- std::string **name** ()
- void **calculate** (P const &Mtx)
- void **solve** (Vector &x, Vector const &b)
- void **imchol** ()

La documentación para esta clase fue generada a partir del siguiente fichero:

- ICHOL.hpp

2.9. Referencia de la plantilla de la Clase ILU< P >

Métodos públicos

- std::string **name** ()
- void **calculate** (P const &Mtx)
- void **solve** (Vector &x, Vector const &b)
- void **iSolve** ()

La documentación para esta clase fue generada a partir del siguiente fichero:

- ILU.hpp

2.10. Referencia de la Clase Jacobi

Métodos públicos

- template<class T >
  void **solve** (T &A, Vector &x, Vector &b)
- int **its** ()
- double **error** ()
- void **maxIts** (int it)
- void **tole** (double t)
- void **report** ()

La documentación para esta clase fue generada a partir del siguiente fichero:

- Jacobi.hpp

2.11. Referencia de la plantilla de la Clase JacobiP< P >

Métodos públicos

- std::string **name** ()
- void **calculate** (P &Mtx)
- void **solve** (Vector &z, Vector const &r)
- void **impDiag** ()

La documentación para esta clase fue generada a partir del siguiente fichero:

- JacobiP.hpp

2.12. Referencia de la Clase Mesh

Clase.

```
#include <Mesh.hpp>
```

Métodos públicos

- Mesh ()

  Constructor vacio, inicializa los atributos a valores por defecto.
- Mesh (std::string filename)

  Constructor para cargar un archivo <nombre>.msh y encargado de llamar a initialize(filename), para dar el tratamiento basico a una geometria rectangular obtenida con la herramienta gmsh.
- Mesh (std::string filename, int i)

  Constructor para cargar un archivo <nombre>.msh y encargado de llamar a initialize(filename), para dar el tratamiento basico a una geometria semi-circular obtenida con la herramienta gmsh.
- void **initialize** (std::string filename)
- void **initialize** (std::string filename, int i)
- void **readFileGMSH** (std::string filename)
- void **reorderNodes** ()
- void **reorderElements** ()
- void **borderInteriorNode** ()
- void **nodeBoundarySemiCircle** ()
- void **elementSupportNode** ()
- std::vector< int > **nodeNeighborh** (int node)
- void areaElement ()

  Calcula y almacena en memoria las áreas de cada elemento triangular.
- int nNode ()

  Devuelve el numero de nodos totales leidos producto de la discretización del dominio de interes.
- int nElement ()

  Devuelve el numero de elementos totales leidos producto de la discretización del dominio de interes.
- double coordX (int node)

  Devuelve las coordenadas del sistema cartesiano, para cada nodo de la malla.
- double coordY (int node)

  Devuelve las coordenadas del sistema cartesiano, para cada nodo de la malla.
- std::vector< int > bordeNode ()

  Devuelve los nodos que estan sobre la frontera del dominio de interes.
- int nbordeNode ()

  Devuelve el numero de nodos frontera.
- std::vector< int > **intertiorNode** ()
- int **ninteriorNode** ()
- std::vector< int > **ElementSupoort** (int node)
- int **nElementSupoort** (int node)
- double **areaK** (int element)
- void **displayInfo** ()
- void **showCoord** ()
- void **showCoordOrder** ()
- void **showRelation** ()
- void **showElements** ()
- void **showElementsOrder** ()
- void **showNodeBoundary** ()
- void **showNodeInterior** ()
- void **showNeighbor** ()
- void **showElementSupportNode** ()
- void **showArea** ()
- void **quiteNode** ()

2.12.1. Descripción detallada

Clase.

Versión

1.0

Fecha

6/12/2016

Autor

Mario Arturo Nieto Butron SYSG

2.12.2. Documentación del constructor y destructor

2.12.2.1. Mesh() [1/2]

```
Mesh::Mesh (
            std::string filename )  [inline]
```

Constructor para cargar un archivo <nombre>.msh y encargado de llamar a initialize(filename), para dar el tratamiento basico a una geometria rectangular obtenida con la herramienta gmsh.

Parámetros

filename	Ruta de un archivo <nombre>.msh para leer el dominio de trabajo.

Ver también

initialize

2.12.2.2. Mesh() [2/2]

```
Mesh::Mesh (
            std::string filename,
            int i )  [inline]
```

Constructor para cargar un archivo <nombre>.msh y encargado de llamar a initialize(filename), para dar el tratamiento basico a una geometria semi-circular obtenida con la herramienta gmsh.

Parámetros

filename	Ruta de un archivo <nombre>.msh para leer el dominio de trabajo.
i	Diferencia a los initialize.

Ver también

initialize

2.12.3. Documentación de las funciones miembro

2.12.3.1. areaElement()

```
void Mesh::areaElement ( )
```

Calcula y almacena en memoria las áreas de cada elemento triangular.

Parámetros

Ninguno	

Devuelve

None

2.12.3.2. bordeNode()

```
std::vector<int> Mesh::bordeNode ( )  [inline]
```

Devuelve los nodos que estan sobre la frontera del dominio de interes.

Parámetros

Ninguno.	

Devuelve

Vector stl entero, con el identificador de cada nodo frontera.

2.12.3.3. coordX()

```
double Mesh::coordX (
            int node )  [inline]
```

Devuelve las coordenadas del sistema cartesiano, para cada nodo de la malla.

Parámetros

node	Nodo valido de la malla. x.

Devuelve

Coordenada x, del nodo.

2.12.3.4. coordY()

```
double Mesh::coordY (
            int node )  [inline]
```

Devuelve las coordenadas del sistema cartesiano, para cada nodo de la malla.

Parámetros

node	Nodo valido de la malla. x.

Devuelve

Coordenada y, del nodo.

2.12.3.5. nbordeNode()

```
int Mesh::nbordeNode ( )  [inline]
```

Devuelve el numero de nodos frontera.

Parámetros

Ninguno	

Devuelve

Valor entero con el numero total de nodos de frontera.

2.12.3.6. nElement()

```
int Mesh::nElement ( )  [inline]
```

Devuelve el numero de elementos totales leidos producto de la discretización del dominio de interes.

Parámetros

Ninguno	

Devuelve

Número total de elementos triangulares, de la discretización.

2.12.3.7. nNode()

```
int Mesh::nNode ( )  [inline]
```

Devuelve el numero de nodos totales leidos producto de la discretización del dominio de interes.

Parámetros

Ninguno	

Devuelve

Número total de nodos, de la discretización.

La documentación para esta clase fue generada a partir de los siguientes ficheros:

- Mesh.hpp
- Mesh.cpp

2.13. Referencia de la plantilla de la Clase MILU< P >

Métodos públicos

- std::string **name** ()
- void **calculate** (P const &Mtx)
- void **solve** (Vector &x, Vector const &b)
- void **iSolve** ()

La documentación para esta clase fue generada a partir del siguiente fichero:

- MILU.hpp

2.14. Referencia de la plantilla de la Clase Mult< L >

Métodos públicos

- **Mult** (L const &l, double r)
- double **operator[ ]** (int n) const

La documentación para esta clase fue generada a partir del siguiente fichero:

- OperVect.hpp

2.15. Referencia de la plantilla de la Clase MV< M, L >

Métodos públicos

- **MV** (const M &m, const L &l)
- double **operator[ ]** (int i) const

La documentación para esta clase fue generada a partir del siguiente fichero:

- OperVect.hpp

2.16. Referencia de la Clase SOR

Métodos públicos

- template<class T >
  void **solve** (T &A, Vector &x, Vector &b)
- int **its** ()
- double **error** ()
- void **maxIts** (int it)
- void **tole** (double t)
- void **setOmega** (double o)
- void **report** ()

La documentación para esta clase fue generada a partir del siguiente fichero:

- SOR.hpp

2.17. Referencia de la plantilla de la Clase Sub< L, R >

Métodos públicos

- **Sub** (L const &l, R const &r)
- double **operator[ ]** (int n) const

La documentación para esta clase fue generada a partir del siguiente fichero:

- OperVect.hpp

2.18. Referencia de la Clase Timer

Métodos públicos

- void **tic** ()
- void **toc** ()
- double **etime** ()

La documentación para esta clase fue generada a partir del siguiente fichero:

- Timer.hpp

2.19. Referencia de la Clase Vector

Métodos públicos

- **Vector** (int n)
- **Vector** (Vector const &v)
- int **size** ()
- double **norm** ()
- double **rms** ()
- void **zeros** ()
- void **saveData** (std::string name)
- void **print** ()
- double **operator[ ]** (int i) const
- double & **operator[ ]** (int i)
- template<class Expr >
  Vector const & **operator=** (Expr const &expr)
- void **operator=** (double escalar)
- void **operator=** (Vector const &v)

Atributos públicos

- double * **_data**
- int **_size**

Amigas

- double **operator*** (Vector const &v, Vector const &v1)

La documentación para esta clase fue generada a partir de los siguientes ficheros:

- Vector.hpp
- Vector.cpp

Índice alfabético