COMPUTATION OF THE SIGNED DISTANCE FUNCTION TO A DISCRETE CONTOUR USING MSHDIST

C. DAPOGNY [1,2] AND P. FREY [1]

[1] *UPMC Univ Paris 06, UMR 7598, Laboratoire J.-L. Lions, F-75005 Paris, France.*
[2] *Renault DREAM-DELT'A Guyancourt, France.*

This short note presents the main features of the code `mshdist` for computing the signed distance function to a discrete contour, associated to the journal article [1].

1. **Files structures.** `Mshdist` considers two kinds of data files: `.mesh` files (for meshes), and `.sol` files (for scalar fields defined at the vertices of a mesh).

- A `.mesh`file contains all the required information about the associated mesh; it is the standard meshing formate used by INRIA programs. Such a file is organized as follows:

```
/* Header */
MeshVersionFormatted 1

Dimension
2

/* List of the vertices of the mesh: two floats in 2d (three in 3d) for the
coordinates, and an integer for a possible reference */
Vertices

3030     // Number of vertices

1 1 2
1 0.975 0
0.975 1 2
0.983333333333 0.966666666154 0
1 0.95 0
....

/* List of the elements of the mesh: three integers in 2d (four in 3d) for the
indices of the vertices, and one additional integer for a possible reference */
Triangles    // Tetrahedra in 3d

5898
900 833 899 0
834 828 770 0
769 834 770 0
900 893 834 0
...

/* Ending keyword */
End
```

LISTING 1. Organization of a `.mesh` file

- A `.sol` file is organized as follows:

```
/* Header */
MeshVersionFormatted 1

Dimension
2

/* Number of vertices for supporting solution */
SolAtVertices
3030

/* 1 = 1 field, 1 = scalar field */
1 1

/* List of solutions associated to the previous mesh */
0.92393
0.000270181
0.886448
0.000515695
...

/* Ending keyword */
End
```

LISTING 2. Organization of a `.sol` file

2. **First mode: distancing algorithm.** The first option of `mshdist` generates the signed distance function d_Ω to a domain Ω supplied as a mesh of its boundary $\partial\Omega$ (edges in $2d$, triangles in $3d$), at the vertices of a computational mesh of a bounding box D. The associated line of command is:

```
mshdist box.mesh contour.mesh
```

This operation produces a file `box.sol`, which contains the information about d_Ω at the vertices of the mesh `box.mesh`.

Note that `contour.mesh` could be supplied itself as a volume mesh of the domain Ω (i.e. by means of triangles in $2d$, tetrahedra in $3d$). In this case, `mshdist` will not read the information about the volume part of the mesh, and will only retain information contained in the fields `Edges` (in $2d$) or `Triangles` (in $3d$) in the mesh file `contour.mesh`.

If the supplied contour is not orientable (i.e. it does not define unambiguously an interior and an exterior), the program fails, and an error message is issued.

Unless `mshdist` is explicitly told not to do so, the contour mesh `contour.mesh` is automatically *scaled* so that its bounding box is a given percentage `SIZE` of the bounding box of the mesh `box.mesh` (so as to avoid problems when computational boxes are not expressed in the same units as the models of interest). By default, `SIZE` is set to 95%; this value can be changed in the file `mshdist.h`. This scaling can also be disabled by adding the command `noscale` on the command line.

Eventually, recall that, for attributing a sign to the distance function, `mshdist` starts from an exterior triangle (tetrahedron in $3d$) to Ω (typically an element located at a corner of D). If no scale is applied, the domain Ω may contain such an element, and it should then be provided by the user (for it may depend on the application !); in this case, the user should specify a point exterior to Ω, by changing the coordinates of `p` on the lines

```
  /* identify triangle close to lower corner (boundary) */
  p[0] = 0.05;
  p[1] = 0.05;
```

LISTING 3. specifying an exterior point (element) to `mshdist`

of the function `sgndist_2d`(resp `_3d`) in files `mshdis1_2d.c`or `mshdis1_3d.c`.

3. **Second mode: redistancing algorithm.** The second option of `mshdist` concerns redistancing, an operation of great interest in the context of the level set method. By entering the command line

```
mshdist box.mesh
```

`mshdist` understands that a solution file `box.sol` exists (defined at the vertices of the input mesh of D), which contains the data of a level set function associated to a domain $\Omega \subset D$. Then, `mshdist` regenerates the signed distance function to this domain, and prints it in the file `box.sol` (be careful: the original solution file is overwritten).

4. **Generation of the signed distance function to a subdomain.** This option considers an input mesh `box.mesh`, which encloses a domain Ω as a submesh (i.e. the elements of Ω are also elements of the larger mesh). The elements of Ω are identified by their *reference number*. By default, they are the elements with number 3 (this can be changed in the file `mshdist.h`, by modifying the value of constant `REFINT`).

By using the command line

```
mshdist box.mesh -dom
```

`mshdist` generates a file `box.sol` which contains the signed distance function to Ω.

5. **Additional options.**

- `mshdist` can work in parallel if the command

  `-ncpu` *number*

  is added to the command line.
- The `-noscale` command, which is only useful in the distancing mode, has been described above.
- The number of iterations of the process can be controlled by adding

  `-it` *number*

  to the command line.

Acknowledgement. This work was issued while the first author was partially supported by the Chair "Mathematical modeling and numerical simulation, F-EADS - École Polytechnique - INRIA".

References

[1] C. Dapogny, P. Frey, *Computation of the signed distance function to a discrete contour on adapted triangulation*, Calcolo, Volume 49, Issue 3, pp. 193-219 (2012).