

An introduction to mesh generation

Part IV : elliptic meshing

Jean-François Remacle

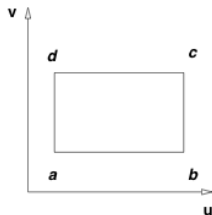
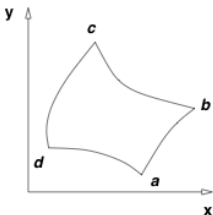
Department of Civil Engineering,
Université catholique de Louvain, Belgium

Curvilinear Meshes

Basic concept

- A curvilinear mesh is build using a parametric space
- Sponge analogy
- Let us consider the real coordonates (x,y) and a parametric space (u,v) .

The transformation $x = x(u,v)$ and $y = y(u,v)$ has to be build.



Curvilinear Meshes

The transformation transforms a cartesian mesh in the parametric space into a structured mesh in the real space with the following properties:

- Two non intersecting lines in the parametric space do not cross in the real space,
- The mesh adapts to the boundary of on the real space,
- The mesh has not to be orthogonal in the real space.

Curvilinear Meshes

The transformation transforms a cartesian mesh in the parametric space into a structured mesh in the real space with the following properties:

- Two non intersecting lines in the parametric space do not cross in the real space,
- The mesh adapts to the boundary of on the real space,
- The mesh has not to be orthogonal in the real space.

Curvilinear Meshes

The transformation transforms a cartesian mesh in the parametric space into a structured mesh in the real space with the following properties:

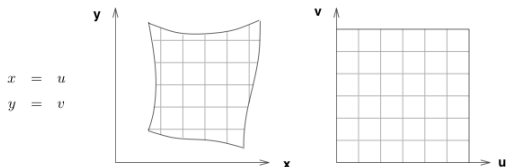
- Two non intersecting lines in the parametric space do not cross in the real space,
- The mesh adapts to the boundary of on the real space,
- The mesh has not to be orthogonal in the real space.

Curvilinear Meshes

The transformation transforms a cartesian mesh in the parametric space into a structured mesh in the real space with the following properties:

- Two non intersecting lines in the parametric space do not cross in the real space,
- The mesh adapts to the boundary of on the real space,
- The mesh has not to be orthogonal in the real space.

The advantage of curvilinear meshes are numerous. They are usually high quality/very regular meshes. Yet, the sponge analogy cannot be applied easily to complex domains.



Transfinite Interpolation

- Let us consider a simply connected 2D manifold that is bounded by 4 curves $\vec{c}_i(t), i = 1 \dots, 4$.
- We consider the four corners \vec{P}_{ij} that are the intersections of curves \vec{c}_i and \vec{c}_j .
- We look for a transformation that maps the unit square in the parametric space into the domain of interest.
- We suppose that curves \vec{c}_1 and \vec{c}_3 will be the mapping of the vertical segments $v = 0$ and $v = 1$.
- We suppose that curves \vec{c}_2 and \vec{c}_4 will be the mapping of the horizontal segments $u = 0$ and $u = 1$.

Transfinite Interpolation

- Let us try the following “intuitive” formula

$$\vec{S}(u, v) = (1 - v)\vec{c}_1(u) + v\vec{c}_3(u) + (1 - u)\vec{c}_4(v) + u\vec{c}_2(v)$$

- Let us see if we get back the boundaries of the domain

$$\vec{S}(u, 0) = \vec{c}_1(u) + (1 - u)\vec{c}_4(0) + u\vec{c}_2(0)$$

$$\vec{S}(u, 1) = \vec{c}_3(u) + (1 - u)\vec{c}_4(1) + u\vec{c}_2(1)$$

$$\vec{S}(0, v) = (1 - v)\vec{c}_1(0) + v\vec{c}_3(0) + \vec{c}_4(v)$$

$$\vec{S}(1, v) = (1 - v)\vec{c}_1(1) + v\vec{c}_3(1) + \vec{c}_2(v)$$

- Let us identify the four corners

$$\vec{c}_1(0) = \vec{c}_4(0) = \vec{P}_{1,4} \quad \vec{c}_1(1) = \vec{c}_2(0) = \vec{P}_{1,2}$$

$$\vec{c}_2(1) = \vec{c}_3(1) = \vec{P}_{2,3} \quad \vec{c}_3(0) = \vec{c}_4(1) = \vec{P}_{3,4}$$

Transfinite Interpolation

- Taking into account the corners

$$\vec{S}(u, 0) = \vec{c}_1(u) + (1 - u)\vec{P}_{1,4} + u\vec{P}_{1,2} \neq \vec{c}_1(u)$$

$$\vec{S}(u, 1) = \vec{c}_3(u) + (1 - u)\vec{P}_{3,4} + u\vec{P}_{2,3} \neq \vec{c}_3(u)$$

$$\vec{S}(0, v) = (1 - v)\vec{P}_{1,4} + v\vec{P}_{3,4} + \vec{c}_4(v) \neq \vec{c}_4(v)$$

$$\vec{S}(1, v) = (1 - v)\vec{P}_{1,2} + v\vec{P}_{2,3} + \vec{c}_2(v) \neq \vec{c}_2(v)$$

- Let us try an alternative formula

$$\begin{aligned} \vec{S}(u, v) = & (1 - v)\vec{c}_1(u) + v\vec{c}_3(u) + (1 - u)\vec{c}_2(v) + u\vec{c}_4(v) - \\ & \left[(1 - u)(1 - v)\vec{P}_{1,4} + uv\vec{P}_{2,3} + u(1 - v)\vec{P}_{1,2} + (1 - u)v\vec{P}_{3,4} \right]. \end{aligned}$$

Transfinite Interpolation

$$\vec{S}(u, v) = (1-v)\vec{c}_1(u) + v\vec{c}_3(u) + (1-u)\vec{c}_2(v) + u\vec{c}_4(v) - \left[(1-u)(1-v)\vec{P}_{1,4} + uv\vec{P}_{2,3} + u(1-v)\vec{P}_{1,2} + (1-u)v\vec{P}_{3,4} \right].$$

Taking into account corners

$$\vec{S}(u, 0) = \vec{c}_1(u) + (1-u)\vec{P}_{1,4} + u\vec{P}_{1,2} - (1-u)\vec{P}_{1,4} - u\vec{P}_{1,2} = \vec{c}_1(u)$$

$$\vec{S}(u, 1) = \vec{c}_3(u) + (1-u)\vec{P}_{3,4} + u\vec{P}_{2,3} - (1-u)\vec{P}_{3,4} - u\vec{P}_{2,3} = \vec{c}_3(u)$$

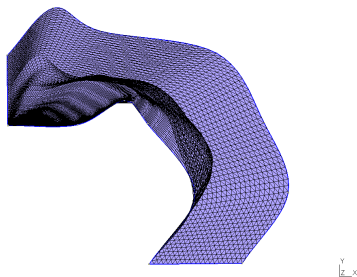
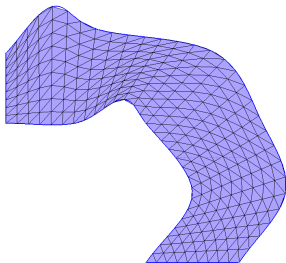
$$\vec{S}(0, v) = (1-v)\vec{P}_{1,4} + v\vec{P}_{3,4} + \vec{c}_4(v) - (1-v)\vec{P}_{1,4} - v\vec{P}_{3,4} = \vec{c}_4(v)$$

$$\vec{S}(1, v) = (1-v)\vec{P}_{1,2} + v\vec{P}_{2,3} + \vec{c}_2(v) - (1-v)\vec{P}_{1,2} - v\vec{P}_{2,3} = \vec{c}_2(v)$$

This formula is correct, it is called the transfinite interpolation formula.

Transfinite Interpolation

Using a transfinite interpolation does not guarantee that the mapping is positive, i.e. the relation $\det|\partial_i S_j| > 0$ is not guaranteed, see some examples with **gmsh**.

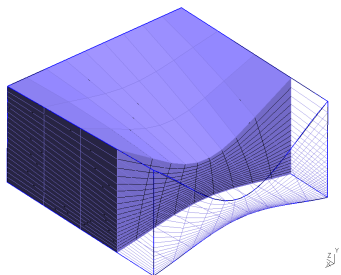
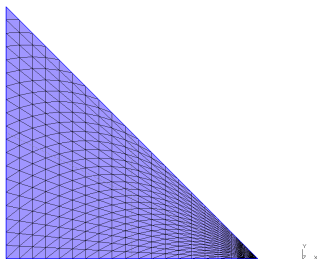


Transfinite Interpolation

Transfinite interpolation exists for 3-sides shapes,

$$\vec{S}(u, v) = u\vec{c}_2(v) + (1-v)\vec{c}_1(u) + v\vec{c}_3(v) - \left[u(1-v)\vec{P}_{1,2} + uv\vec{P}_{2,3} \right].$$

Transfinite interpolation exists for 3D shapes.

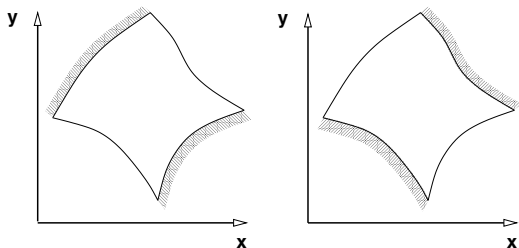


Elliptic meshes

- It is not always simple to devise an explicit formula for the mesh mapping,
- With the transfinite interpolation, mesh lines may cross each others, leading to a wrong mesh
- PDE based mesh have been developped in order to avoid those difficulties
- Two kind of approaches are possible, using either an hyperbolic PDE, or using an **elliptic** PDE. This last approach is now studied.

Elliptic meshes : Thermal Analogy

- Let us consider the same 4-sided domain as the one we used for transfinite meshing,
- We build mesh lines using a thermal analogy : mesh lines are superimposed with the isolines of temperature.
- Steady state heat equation has to be solved.
- Two sides of the domain are adiabatical and a constant temperature is imposed on the two other sides.



Elliptic meshes : Thermal Analogy

We have to build up a PDE system that will lead to two families of (orthogonal) mesh lines. For that purpose, two “thermal problems” will be defined:

- 1 The first family is constructed using the isolines of temperatures ξ_1 of a thermal problem in which two opposite sides are adiabatical, i.e. $\partial_n \xi_1 = 0$. The temperature is set to be constant on each of the two other sides: $\xi_1 = 1$ on the first side and $\xi_1 = 0$ on the other side. The temperature field ξ_1 has to satisfy the Laplace equation inside the domain :

$$\nabla^2 \xi_1 = 0$$

- 2 The second family is constructed using the isolines of temperatures ξ_2 of the adjoint thermal problem in which adiabatic sides are replaced by constant temperature sides and where constant temperature sides are replaced by adiabatic sides. The co-temperature field ξ_2 satisfies the Laplace equation as well :

$$\nabla^2 \xi_2 = 0$$

Elliptic meshes : Thermal Analogy

The two families of mesh lines will have the right properties, thanks to the properties of the solutions of the Laplace equation:

- The maximum principle : any solution of the Laplace equation has its maximum on the boundaries (no local maxima is allowed),
- The solution is unique,
- The solution belongs to H^1 , it is smooth,
- Iso-contours cannot intersect,
- Iso-contours stay inside the domain (obvious for a PDE, not for a general mapping),

Elliptic meshes : Thermal Analogy

The two families of mesh lines will have the right properties, thanks to the properties of the solutions of the Laplace equation:

- The maximum principle : any solution of the Laplace equation has its maximum on the boundaries (no local maxima is allowed),
- The solution is unique,
- The solution belongs to H^1 , it is smooth,
- Iso-contours cannot intersect,
- Iso-contours stay inside the domain (obvious for a PDE, not for a general mapping),

Elliptic meshes : Formulation

- The resulting “mesh equations” are two boundary value problems. We have to find ξ_1 and ξ_2 solution of

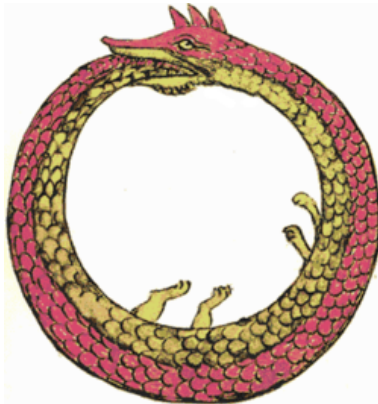
$$\nabla^2 \xi_i = 0 \quad , i = 1, 2.$$

- boundary conditions are

Boundary	ξ_1	ξ_2
\vec{c}_1	$\xi_1(x, y) = 0$	$\partial_n \xi_2(x, y) = 0$
\vec{c}_2	$\partial_n \xi_1(x, y) = 0$	$\xi_2(x, y) = 0$
\vec{c}_3	$\xi_1(x, y) = 1$	$\partial_n \xi_2(x, y) = 0$
\vec{c}_4	$\partial_n \xi_1(x, y) = 0$	$\xi_2(x, y) = 1$

Elliptic meshes : Formulation

- Problem : the equations have to be solved in the real space.
Disappointingly, solving such PDE requires... **a mesh** !



Elliptic meshes : Formulation

- This issue is addressed by the inversion of variables x and y and ξ_1 and ξ_2 ,
- The problem is solved in the space of temperatures ξ_1 and ξ_2 and the unknowns are the positions of the mesh in the real space x and y ,
- For a pair of temperatures (ξ_1, ξ_2) , we look the point (x, y) in the physical space where this temperature occurs.
- Because of the properties of the Laplace operator, there exist such a point (x, y) for any pair (ξ_1, ξ_2) for which $0 \leq \xi_1 \leq 1$ and for which $0 \leq \xi_2 \leq 1$
- This is equivalent to the following change of variables

$$x = x_1 = x_1(\xi_1, \xi_2) \quad \text{and} \quad y = x_2 = x_2(\xi_1, \xi_2)$$

Elliptic meshes : Formulation

- This issue is addressed by the inversion of variables x and y and ξ_1 and ξ_2 ,
- The problem is solved in the space of temperatures ξ_1 and ξ_2 and the unknowns are the positions of the mesh in the real space x and y ,
- For a pair of temperatures (ξ_1, ξ_2) , we look the point (x, y) in the physical space where this temperature occurs.
- Because of the properties of the Laplace operator, there exist such a point (x, y) for any pair (ξ_1, ξ_2) for which $0 \leq \xi_1 \leq 1$ and for which $0 \leq \xi_2 \leq 1$
- This is equivalent to the following change of variables

$$x = x_1 = x_1(\xi_1, \xi_2) \quad \text{and} \quad y = x_2 = x_2(\xi_1, \xi_2)$$

Elliptic meshes : Formulation

- This issue is addressed by the inversion of variables x and y and ξ_1 and ξ_2 ,
- The problem is solved in the space of temperatures ξ_1 and ξ_2 and the unknowns are the positions of the mesh in the real space x and y ,
- For a pair of temperatures (ξ_1, ξ_2) , we look the point (x, y) in the physical space where this temperature occurs.
- Because of the properties of the Laplace operator, there exist such a point (x, y) for any pair (ξ_1, ξ_2) for which $0 \leq \xi_1 \leq 1$ and for which $0 \leq \xi_2 \leq 1$
- This is equivalent to the following change of variables

$$x = x_1 = x_1(\xi_1, \xi_2) \quad \text{and} \quad y = x_2 = x_2(\xi_1, \xi_2)$$

Elliptic meshes : Formulation

- This issue is addressed by the inversion of variables x and y and ξ_1 and ξ_2 ,
- The problem is solved in the space of temperatures ξ_1 and ξ_2 and the unknowns are the positions of the mesh in the real space x and y ,
- For a pair of temperatures (ξ_1, ξ_2) , we look the point (x, y) in the physical space where this temperature occurs.
- Because of the properties of the Laplace operator, there exist such a point (x, y) for any pair (ξ_1, ξ_2) for which $0 \leq \xi_1 \leq 1$ and for which $0 \leq \xi_2 \leq 1$
- This is equivalent to the following change of variables

$$x = x_1 = x_1(\xi_1, \xi_2) \quad \text{and} \quad y = x_2 = x_2(\xi_1, \xi_2)$$

Elliptic meshes : Formulation

- This issue is addressed by the inversion of variables x and y and ξ_1 and ξ_2 ,
- The problem is solved in the space of temperatures ξ_1 and ξ_2 and the unknowns are the positions of the mesh in the real space x and y ,
- For a pair of temperatures (ξ_1, ξ_2) , we look the point (x, y) in the physical space where this temperature occurs.
- Because of the properties of the Laplace operator, there exist such a point (x, y) for any pair (ξ_1, ξ_2) for which $0 \leq \xi_1 \leq 1$ and for which $0 \leq \xi_2 \leq 1$
- This is equivalent to the following change of variables

$$x = x_1 = x_1(\xi_1, \xi_2) \quad \text{and} \quad y = x_2 = x_2(\xi_1, \xi_2)$$

Elliptic meshes : Formulation

We use the chain derivative rule to compute

$$\frac{\partial}{\partial x_j} = \sum_{j=1}^2 \frac{\partial}{\partial \xi_j} \frac{\partial \xi_j}{\partial x_i} \quad \text{and} \quad \frac{\partial^2}{\partial x_i \partial x_k} = \sum_{j=1}^2 \sum_{l=1}^2 \frac{\partial^2}{\partial \xi_j \partial \xi_l} \frac{\partial \xi_l}{\partial x_k} \frac{\partial \xi_j}{\partial x_i}$$

In expanded form

$$\frac{\partial^2}{\partial x_1^2} = \frac{\partial^2}{\partial \xi_1^2} \left(\frac{\partial \xi_1}{\partial x_1} \right)^2 + 2 \frac{\partial^2}{\partial \xi_1 \partial \xi_2} \left(\frac{\partial \xi_1}{\partial x_1} \frac{\partial \xi_2}{\partial x_1} \right) + \frac{\partial^2}{\partial \xi_2^2} \left(\frac{\partial \xi_2}{\partial x_1} \right)^2.$$

$$\frac{\partial^2}{\partial x_2^2} = \frac{\partial^2}{\partial \xi_1^2} \left(\frac{\partial \xi_1}{\partial x_2} \right)^2 + 2 \frac{\partial^2}{\partial \xi_1 \partial \xi_2} \left(\frac{\partial \xi_1}{\partial x_2} \frac{\partial \xi_2}{\partial x_2} \right) + \frac{\partial^2}{\partial \xi_2^2} \left(\frac{\partial \xi_2}{\partial x_2} \right)^2.$$

Elliptic meshes : Formulation

The Laplace operator is written as

$$\nabla^2 = \frac{\partial^2}{\partial \xi_1^2} \left(\left(\frac{\partial \xi_1}{\partial x_1} \right)^2 + \left(\frac{\partial \xi_1}{\partial x_2} \right)^2 \right) + 2 \frac{\partial^2}{\partial \xi_1 \partial \xi_2} \left(\frac{\partial \xi_1}{\partial x_1} \frac{\partial \xi_2}{\partial x_1} + \frac{\partial \xi_1}{\partial x_2} \frac{\partial \xi_2}{\partial x_2} \right) + \frac{\partial^2}{\partial \xi_2^2} \left(\left(\frac{\partial \xi_2}{\partial x_1} \right)^2 + \left(\frac{\partial \xi_2}{\partial x_2} \right)^2 \right).$$

Therefore, in 2D, the PDE's that have to be solved can be written as

$$\alpha \begin{bmatrix} \frac{\partial^2 x_1}{\partial \xi_1^2} \\ \frac{\partial^2 x_2}{\partial \xi_1^2} \end{bmatrix} + 2\beta \begin{bmatrix} \frac{\partial^2 x_1}{\partial \xi_1 \partial \xi_2} \\ \frac{\partial^2 x_2}{\partial \xi_1 \partial \xi_2} \end{bmatrix} + \gamma \begin{bmatrix} \frac{\partial^2 x_1}{\partial \xi_2^2} \\ \frac{\partial^2 x_2}{\partial \xi_2^2} \end{bmatrix} = 0.$$

with

$$\alpha = \left(\frac{\partial \xi_1}{\partial x_1} \right)^2 + \left(\frac{\partial \xi_1}{\partial x_2} \right)^2, \quad \beta = \frac{\partial \xi_1}{\partial x_1} \frac{\partial \xi_2}{\partial x_1} + \frac{\partial \xi_1}{\partial x_2} \frac{\partial \xi_2}{\partial x_2}, \quad \gamma = \left(\frac{\partial \xi_2}{\partial x_1} \right)^2 + \left(\frac{\partial \xi_2}{\partial x_2} \right)^2$$

Elliptic meshes : Formulation

Note that

$$\alpha = J^2 \left[\left(\frac{\partial x_1}{\partial \xi_1} \right)^2 + \left(\frac{\partial x_1}{\partial \xi_2} \right)^2 \right] = \left(\frac{\partial \xi_1}{\partial x_1} \right)^2 + \left(\frac{\partial \xi_1}{\partial x_2} \right)^2,$$

$$\beta = -J^2 \left[\frac{\partial x_1}{\partial \xi_1} \frac{\partial x_2}{\partial \xi_1} + \frac{\partial x_1}{\partial \xi_2} \frac{\partial x_2}{\partial \xi_2} \right] = \frac{\partial \xi_1}{\partial x_1} \frac{\partial \xi_2}{\partial x_1} + \frac{\partial \xi_1}{\partial x_2} \frac{\partial \xi_2}{\partial x_2},$$

$$\gamma = J^2 \left[\left(\frac{\partial x_2}{\partial \xi_1} \right)^2 + \left(\frac{\partial x_2}{\partial \xi_2} \right)^2 \right] = \left(\frac{\partial \xi_2}{\partial x_1} \right)^2 + \left(\frac{\partial \xi_2}{\partial x_2} \right)^2,$$

with

$$J = \det[\partial_{\xi_j} x_i]$$

Elliptic meshes : Formulation

The equations that have to be solved for the mesh problem are coupled and non-linear! They can only be solved using numerical methods. There exists several techniques to solve such systems. We are not going to detail all the methods that are available (this is could be the subject of another course). We are going to illustrate some basic principles. Two items will be detailed here

- The numerical discretization of the mesh equations in the parametric space (ξ_1, ξ_2) ,
- Their resolution.

Elliptic meshes : Formulation

We assume that the “mesh” in the parametric space is a $m \times n$ finite difference mesh. The mesh is uniform in the sense that the spacings $\Delta\tau$ and $\Delta\eta$ in both directions are constant.

$$\begin{aligned} \frac{\partial f}{\partial \tau} &\approx \frac{f_{i+1,j} - f_{i-1,j}}{2\Delta\tau} \\ \frac{\partial f}{\partial \eta} &\approx \frac{f_{i,j+1} - f_{i,j-1}}{2\Delta\eta} \\ \frac{\partial^2 f}{\partial \tau^2} &\approx \frac{f_{i+1,j} - 2f_{i,j} + f_{i-1,j}}{\Delta\tau^2} \\ \frac{\partial^2 f}{\partial \eta^2} &\approx \frac{f_{i,j+1} - 2f_{i,j} + f_{i,j-1}}{\Delta\eta^2} \\ \frac{\partial^2 f}{\partial \tau \partial \eta} &\approx \frac{f_{i+1,j+1} - f_{i+1,j-1} - f_{i-1,j+1} + f_{i-1,j-1}}{4\Delta\tau\Delta\eta} \end{aligned}$$

Elliptic meshes : Formulation

We recall the mesh equations

$$\alpha \begin{bmatrix} \frac{\partial^2 x_1}{\partial \xi_1^2} \\ \frac{\partial^2 x_2}{\partial \xi_1^2} \end{bmatrix} + 2\beta \begin{bmatrix} \frac{\partial^2 x_1}{\partial \xi_1 \partial \xi_2} \\ \frac{\partial^2 x_2}{\partial \xi_1 \partial \xi_2} \end{bmatrix} + \gamma \begin{bmatrix} \frac{\partial^2 x_1}{\partial \xi_2^2} \\ \frac{\partial^2 x_2}{\partial \xi_2^2} \end{bmatrix} = 0.$$

Their discrete version can be written, for any vertex (i, j) of the mesh in the parametric space, as

$$\begin{aligned} \alpha' [x_{i+1,j} - 2x_{i,j} + x_{i-1,j}] + \gamma' [x_{i,j+1} - 2x_{i,j} + x_{i,j-1}] \\ - 2\beta' [x_{i+1,j+1} - x_{i-1,j+1} - x_{i+1,j-1} + x_{i-1,j-1}] = 0 \end{aligned}$$

$$\begin{aligned} \alpha' [y_{i+1,j} - 2y_{i,j} + y_{i-1,j}] + \gamma' [y_{i,j+1} - 2y_{i,j} + y_{i,j-1}] \\ - 2\beta' [y_{i+1,j+1} - y_{i-1,j+1} - y_{i+1,j-1} + y_{i-1,j-1}] = 0 \end{aligned}$$

Elliptic meshes : Resolution

With the numerical evaluation of coefficients α , β and γ

$$\alpha' = \frac{(x_{i,j+1} - x_{i,j-1})^2 + (y_{i,j+1} - y_{i,j-1})^2}{(2\Delta\tau\Delta\eta)^2}$$

$$\gamma' = \frac{(x_{i+1,j} - x_{i-1,j})^2 + (y_{i+1,j} - y_{i-1,j})^2}{(2\Delta\tau\Delta\eta)^2}$$

$$\beta' = \frac{(x_{i+1,j} - x_{i-1,j})(x_{i,j+1} - x_{i,j-1})}{(4\Delta\tau\Delta\eta)^2} + \frac{(y_{i+1,j} - y_{i-1,j})(y_{i,j+1} - y_{i,j-1})}{(4\Delta\tau\Delta\eta)^2}$$

Elliptic meshes : Resolution

Applying the formulation to the $m \times n$ nodes of the finite difference mesh, we obtain a coupled system of non linear equations. In fact, α' , β' and γ' are all functions of the unknowns, i.e. the position of the nodes in the real space. Several methods are available for solving such a system. They can be classified in two categories:

- 1 direct methods,
- 2 iterative methods.

The choice of the method is essentially based on the following criterion

- The CPU time cost and the memory cost,
- The ability of the method to converge to the solution,
- The relative ease of programming !

Elliptic meshes : Resolution

Direct methods are not the choice that is usually made for building elliptic meshes. Direct methods have the advantage of ensuring the convergence in a finite number of step (yet this not true for non-linear systems). Their principal drawback is their memory signature and their (relative) complexity.

Iterative methods have the following advantages:

- 1 They are usually easier to code,
- 2 They are well adapted to structured grids,

The following “smoothers” are often used for building elliptic meshes

- Point Jacobi iteration,
- Block Jacobi iteration (the block may be a line or a column of nodes),
- ADI's, i.e. alternated direction implicit !

Elliptic meshes : Point Jacobi Iteration

We look for the values of the variables x and y , noted x^- and y^- at node (i,j) verifying

$$\alpha' [x_{i+1,j} - 2x_{i,j}^- + x_{i-1,j}^+] + \gamma' [x_{i,j+1} - 2x_{i,j}^- + x_{i,j-1}^+] - 2\beta' [x_{i+1,j+1} - x_{i-1,j+1} - x_{i+1,j-1}^+ + x_{i-1,j-1}^+] = 0$$

$$\alpha' [y_{i+1,j} - 2y_{i,j}^- + y_{i-1,j}^+] + \gamma' [y_{i,j+1} - 2y_{i,j}^- + y_{i,j-1}^+] - 2\beta' [y_{i+1,j+1} - y_{i-1,j+1} - y_{i+1,j-1}^+ + y_{i-1,j-1}^+] = 0$$

Here, x^- and y^- are for the values of x and y at the previous sweep and x^+ and y^+ are the values at the present sweep. The linearization of the coefficients consist in using previous values of x and y in α' , β' and γ' .

Elliptic meshes : Over relaxation

We know that x and y are updated by doing

$$x = x + (x^- - x) \quad \text{and} \quad y = y + (y^- - y).$$

Over relaxing the system consist in doing

$$x^+ = x + \omega(x^- - x) \quad \text{and} \quad y^+ = y + \omega(y^- - y).$$

with $\omega \geq 1$. Current values are computed as

$$x_{i,j}^- = x_{i,j} + C_{x,i,j}/\omega \quad \text{and} \quad y_{i,j}^- = y_{i,j} + C_{y,i,j}/\omega.$$

with the corrections

$$C_{x,i,j} = x_{i,j}^+ - x_{i,j} \quad \text{and} \quad C_{y,i,j} = y_{i,j}^+ - y_{i,j}.$$

Elliptic meshes : Over relaxation

Replacing in the discrete form of the equations, we obtain

$$\begin{aligned} \frac{2(\alpha' + \gamma')}{\omega} C_{x^i, j} &= R_{x^i, j} + \alpha' C_{x^{i-1}, j} \\ &\quad - \beta' (C_{x^{i-1}, j-1} - C_{x^{i+1}, j-1}) + \gamma' C_{x^i, j-1} \\ \frac{2(\alpha' + \gamma')}{\omega} C_{y^i, j} &= R_{y^i, j} + \alpha' C_{y^{i-1}, j} \\ &\quad - \beta' (C_{y^{i-1}, j-1} - C_{y^{i+1}, j-1}) + \gamma' C_{y^i, j-1} \end{aligned}$$

with the residuals

$$\begin{aligned} R_{x^i, j} &= \alpha' [x_{i+1, j} - 2x_{i, j} + x_{i-1, j}] + \gamma' [x_{i, j+1} - 2x_{i, j} + x_{i, j-1}] \\ &\quad - 2\beta' [x_{i+1, j+1} - x_{i-1, j+1} - x_{i+1, j-1} + x_{i-1, j-1}] \\ R_{y^i, j} &= \alpha' [y_{i+1, j} - 2y_{i, j} + y_{i-1, j}] + \gamma' [y_{i, j+1} - 2y_{i, j} + y_{i, j-1}] \\ &\quad - 2\beta' [y_{i+1, j+1} - y_{i-1, j+1} - y_{i+1, j-1} + y_{i-1, j-1}] \end{aligned}$$

Elliptic meshes : Boundary conditions

We do not want to move the boundary nodes. The way to implement it is very simple : do not apply any corrections to those nodes

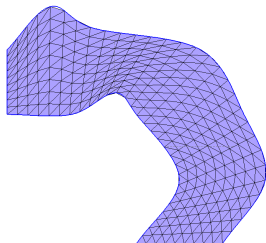
$$C_x 1, j = C_x m, j = 0$$

$$C_y 1, j = C_y m, j = 0$$

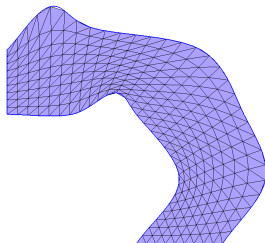
Those are Dirichlet boundary conditions.

Elliptic meshes : Example

Here is an example of a transfinite mesh and an elliptic mesh, both based on the same 1D discretization.



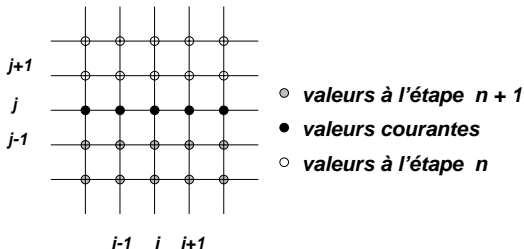
Y
L_x



Y
L_x

Elliptic meshes : Line Jacobi Iteration

Iterative process can be accelerated using a more implicit method. We could compute one line at a time



The resulting system is tridiagonal and ad hoc linear solvers can be used for accelerating the process.

Elliptic meshes : Line Jacobi Iteration

The system of equations can be assembled solved using the following finite difference formula.

$$\alpha' [x_{i+1,j}^- - 2x_{i,j}^- + x_{i-1,j}^-] + \gamma' [x_{i,j+1} - 2x_{i,j}^- + x_{i,j-1}^+] - 2\beta' [x_{i+1,j+1} - x_{i-1,j+1} - x_{i+1,j-1}^+ + x_{i-1,j-1}^+] = 0$$

$$\alpha' [y_{i+1,j}^- - 2y_{i,j}^- + y_{i-1,j}^-] + \gamma' [y_{i,j+1} - 2y_{i,j}^- + y_{i,j-1}^+] - 2\beta' [y_{i+1,j+1} - y_{i-1,j+1} - y_{i+1,j-1}^+ + y_{i-1,j-1}^+] = 0$$