L. Mazorra, F. Díaz, A. Oliver, E. Rodríguez,
G. Montero, J.M. Escobar, R. Montenegro,
P. Lauret, M. David , J. Polo, G. Morales,
J. Calvo and R. Pérez

Wind Field and Solar Radiation Diagnostic and Forecasting

A Numerical Approach for Complex Terrain

March 15, 2019

Springer

Contents

Chapter 1
Discretization of the Region of Interest (J. M. Cascón, J.M. Escobar, R. Montenegro)

Abstract The meccano method was recently introduced to construct simultaneously tetrahedral meshes and volumetric parameterizations of solids. The method requires the information of the solid geometry that is defined by its surface, a meccano, i.e. an outline of the solid defined by connected polyhedral pieces, and a tolerance that fixes the desired approximation of the solid surface. The method builds an adaptive tetrahedral mesh of the solid (physical domain) as a deformation of an appropriate tetrahedral mesh of the meccano (parametric domain). The main stages of the procedure involve an admissible mapping between the meccano and the solid boundaries, the nested Kossaczký's refinement and our simultaneous untangling and smoothing algorithm. In this chapter, we focus the application of the method to build tetrahedral meshes over complex terrain, that are interesting for simulation of environmental processes. A digital elevation map of the terrain, the height of the domain, and the required orography approximation are given as input data. In addition, the geometry of buildings or stacks can be considered. In these applications we have considered a simple cuboid as meccano.

1.1 Introduction

Mesh generation is one of the most time-consuming process in numerical simulations of engineering problems based on partial differential equations. On one hand, an accurate discretization of the physical domain is fundamental to obtain a realistic simulation. On the other hand, it is well known that the quality of the mesh plays a crucial role in the accuracy and stability of the numerical computation.

Many authors have devoted great effort to solving the automatic mesh generation problem in different ways [3, 12, 13, 24]. Along the past, the main objective has been to achieve high quality adaptive meshes of complex solids with minimal user intervention and low computational cost. At present, it is well known that most mesh generators are based on Delaunay triangulation and advancing front technique, but problems, related to mesh quality or mesh conformity with the solid boundary, can still appear for complex geometries. In addition, an appropriate definition of element sizes is demanded for obtaining good quality elements and mesh adaption. Particularly, local adaptive refinement strategies have been employed to mainly adapt the mesh to singularities of numerical solution. These adaptive methods usually involve remeshing or nested refinement [4, 14, 16, 17, 23].

In this direction, we have introduced the meccano technique in [5, 18, 19, 20, 6] for constructing adaptive tetrahedral meshes of solids. This algorithm requires a coarse computational domain, then builds a surface parameterization and combines refinement and a mesh optimization to produce an adaptive tetrahedral mesh of the input domain. As a result, a piecewise linear volumetric parameterization is obtained. The name of the method is due that the process starts from a coarse approximation of the solid, i.e. a meccano composed by connected polyhedral pieces.

The rest of the chapter is structured as follows. In Section 1.2 we overview the meccano method for a general solid. In Section 1.3 we detail the algorithm to the construction of the tetrahedral meshes over complex terrain. Finally, in Section 1.4, we present several examples that illustrate the capabilities of the method.

1.2 The Meccano method

The meccano method is tetrahedral mesh generator [18, 19, 20, 6]. The method requires a surface triangulation of the solid boundaries and a computational domain that coarsely approximates the solid. This computational domain is called meccano. The procedure builds an adaptive tetrahedral mesh in the meccano and deforms it to match the physical domain. For this purpose, the method combines several procedures: an automatic mapping from the boundary of the meccano to the boundary of the solid, a 3-D local refinement algorithm, and a simultaneous mesh untangling and smoothing. It is important to point out that this method also provides a continuous element-wise linear volumetric parameterization from the computational domain to the solid.

The construction of the meccano (the computational domain) is not automatic for complex solids (genus bigger than zero) and requires user intervention. However, in this chapter, we focus on the generation of tetrahedral meshes over complex terrain, that are interesting for simulation of environmental processes, and whose boundary is genus zero. Therefore, the meccano algorithm is fully automatic procedure.

The main steps of the meccano tetrahedral mesh generation algorithm are summarized in Algorithm 1. The input data is a solid, Ω, defined by its boundary representation (surface triangulation or CAD model), and a given precision to approximate its boundary, ε.

The first step of the procedure, Line 2, is to construct a meccano, $\mathscr{M}$, that approximates the solid, by connecting polyhedral pieces. Then, in Line 3, a discrete mapping, Π, between the boundary of the meccano and the boundary of the solid is computed using a procedure based on the mean value parametrization proposed by Floater in [10, 11]. Note that this parameterization is a continuous and element-

Algorithm 1 Meccano tetrahedral mesh generation.

```
1:  function MeccanoMesher(Solid $\Omega$, Real $\varepsilon$)
2:      Meccano $\mathscr{M} \leftarrow$ getMeccano($\Omega$)
3:      Mapping $\Pi \leftarrow$ getBoundaryMapping($\mathscr{M}$, $\Omega$)
4:      Mesh $\mathscr{T} \leftarrow$ getInitialMesh($\mathscr{M}$, $\Pi$)
5:      while distance($\partial\mathscr{T}$, $\partial\Omega$) $> \varepsilon$ do
6:          TriangleList $\mathscr{R} \leftarrow$ getTrianglesToRefine($\mathscr{T}$, $\Omega$, $\varepsilon$)
7:          TriangleList $\widehat{\mathscr{R}} \leftarrow$refineTriangles($\mathscr{R}$)
8:          projectNewNodesToBoundary($\widehat{\mathscr{R}}$, $\Pi$)
9:      end while
10:     qualityOptimization($\mathscr{T}$)
11: end function
```

wise linear mapping. In Line 4, an initial coarse mesh of the meccano is generated, and the boundary nodes are located on the solid boundary using the mapping Π. In Lines 5–9, we obtain a mesh that approximates the solid boundary with the given tolerance ε. Specifically, in Line 6, we get the list of triangles that do not correctly approximate the boundary of the solid, and then, in Line 7, we refine those triangles by dividing their adjacent tetrahedra using the Kossaczký method [16]. In Line 8, we project the new nodes onto the boundary of the solid using the mapping Π. We iterate this process until there are no triangles to refine. Note that when the nodes are mapped onto the solid boundary, low-quality and inverted elements may appear. Thus, a simultaneous untangling and smoothing procedure [8] is applied in order to obtain a valid and high-quality tetrahedral mesh. As commented before, the meccano method automatically provides a volumetric parameterization from the computational domain to the solid. We show in Fig. 1.1 an example where different steps of the meccano algorithm are sumarized.

1.3 Meccano method for the construction of 3D meshes over complex topography

In this section we describe in detail the meccano method for the discretization of three-dimensional domain that is limited in its lower part by a complex terrain, and in its upper part by a rectangular horizontal plane region placed at a given height. The lateral walls are formed by four vertical planes. The generated tetrahedral mesh is appropriated for finite element simulations of environmental processes, such as weather forecasting, wind field (see Chapter **??**), fire propagation or atmospheric pollution [21, 22].

In this case, the meccano consists on a simple cuboid. A digital elevation map of the terrain and the required orography approximation are given as input data. The mapping between the meccano and domain boundaries is obtained by applying an automatic Floater's parameterization [10, 11].

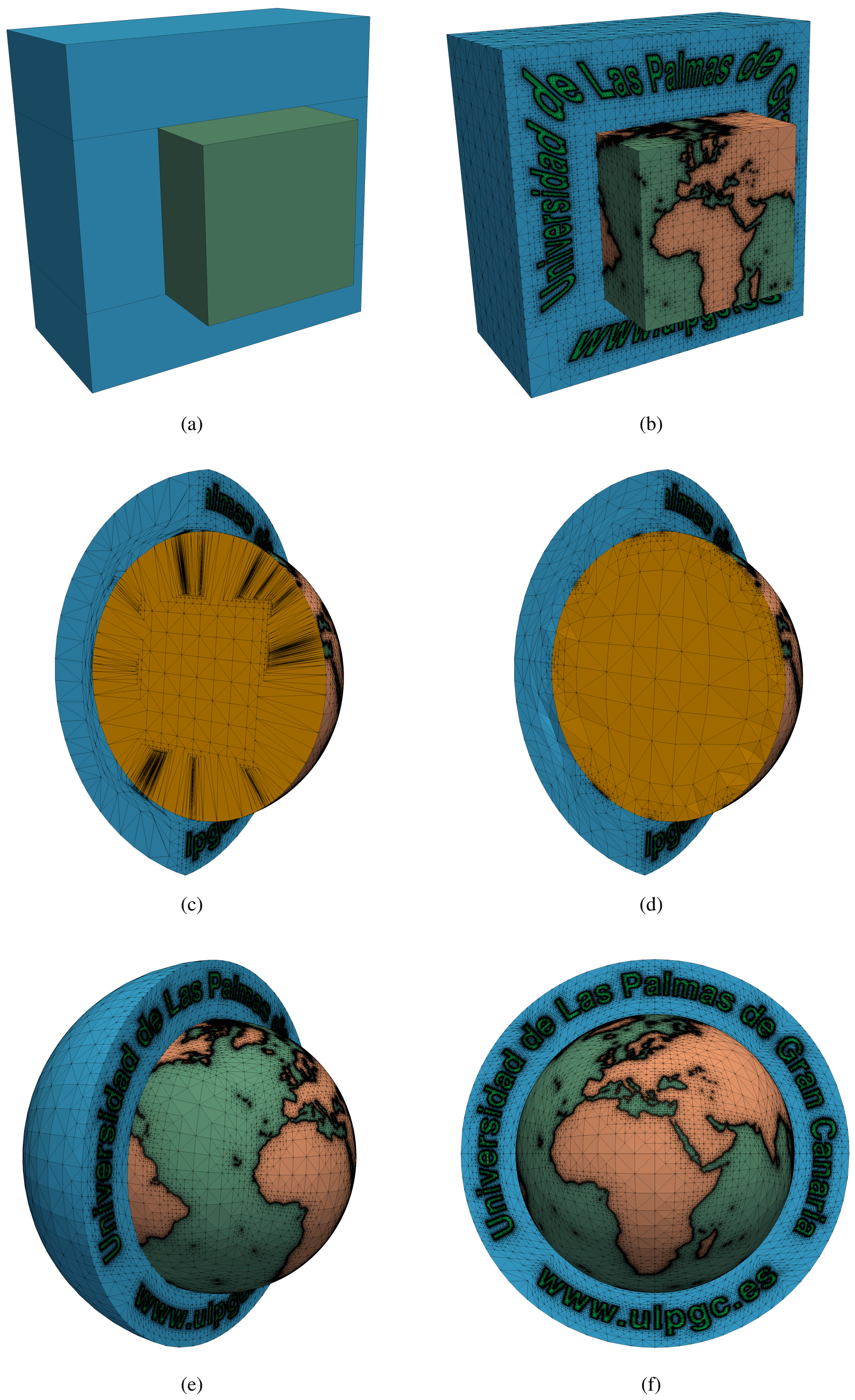


Fig. 1.1 The different meccano steps. (a) Parametric space, (b) adaptive triangulation of the meccano boundary, (c) tangled interior elements after surface mapping, (d) optimized mesh, (e) resulting surface mesh, (f) frontal view

1.3.1 Meccano

As we commented before, a simple cuboid, $\mathscr{M}$, is defined as meccano. Its upper face coincides with the upper boundary of the domain, and its lower face is placed at the minimum terrain height of the rectangular region.

We now divide the surface of the domain in six trivial patches (lower and upper bound and lateral walls), and associate then with counterparts in the meccano boundary. This correspondence must be compatible, in the sense, that if two patches of the domain has non-empty intersection, their corresponding images on meccano boundary also satisfy this property.

1.3.2 Parametrization of the domain boundary

Once the meccano is fixed, we have to determine a mapping between the cuboid faces and the domain boundary. For that purpose, a discrete mapping from each surface patch to the corresponding cuboid face is built using the mean value parameterization proposed in [11]. In order to get an admisible mapping, we note that the discrete mappings have to coincide on the intersection of their associate patches.

We now describe how the parametrization Π_b of the lower bound $\partial\Omega_b$ of the physical space, can be obtained. The other ones (upper bound and lateral walls) are simpler, and could be generated in the same way. In fact, in this particular case, the parametrization of the upper bound is the identity. The method was introduced by Floater [11], and provides a parametrization of a simply connected surface triangulation.

We assume that the digital elevation map, that captures the orography, is given by a triangular mesh embedded in 3D, that we denote $\mathscr{T}_b$. The bottom cuboid face $\partial\mathscr{M}_b$ will be the parametric space. Then, we find a mapping

$$\Pi_b : \partial\mathscr{M}_b \to \partial\Omega_b$$

continuous and piece-wise linear, where $\tau_b = (\Pi_b)^{-1}(\mathscr{T}_b)$ will be the planar triangulation of $\partial\mathscr{M}_b$ associated to $\mathscr{T}_b$. Note that the construction of the mapping is equivalent to find an *admissible* localization of nodes of the planar triangulation τ_b. The Floater solution first fixes the boundary nodes of τ_b and then the position of the inner ones is given by the solution of a linear system based on convex combinations.

Formally, let $\{\mathbf{x}_1,\dots,\mathbf{x}_n\}$ be the inner nodes and $\{\mathbf{x}_{n+1},\dots,\mathbf{x}_N\}$ be the boundary nodes of $\mathscr{T}_b$, respectively, where N denotes the total number of nodes of $\mathscr{T}_b$. Fixed the position of boundary nodes $\{\mathbf{y}_{n+1},\dots,\mathbf{y}_N\}$ of τ_b, the position of its inner nodes $\{\mathbf{y}_1,\dots,\mathbf{y}_n\}$ is given by the solution of the system:

$$\mathbf{y}_k = \sum_{l=1}^{N} \lambda_{kl}\mathbf{y}_l, \qquad k = 1,\dots,n.$$

The values of $\{\lambda_{kl}\}_{k=1,\dots,n}^{l=1,\dots,N}$ are the weights of the convex combinations, such that

$$\begin{aligned}
\lambda_{kl} &= 0, && \text{if } \mathbf{x}_k \text{ and } \mathbf{x}_l \text{ are not connected}\\
\lambda_{kl} &> 0, && \text{if } \mathbf{x}_k \text{ and } \mathbf{x}_l \text{ are connected}\\
\sum_{l=1}^{N} \lambda_{kl} &= 1, && \text{for } k = 1, \dots n.
\end{aligned}$$

The quality parametrization depend on the weights λ_{kl}. In [10] three alternatives are analyzed: uniform parametrization, weighted least squares of edge lengths and shape preserving parametrization. Another choice, called mean value coordinate, is presented in [11]. The goal is to obtain an approximation of a conformal mapping.

Finally, we remark that in order to get an admissible mapping between meccano boundary, $\partial\mathscr{M}$, and domain boundary, $\partial\Omega$, the boundary nodes that the Floater algorithm fixes for the parameterization of each face, must coincide on their common cuboid edges.

In Fig. 1.2 we show a surface triangulation, that approximates the orography of an area of Comunidad de Madrid (Spain), and its corresponding parameterization space obtained by the procedure described in this section.

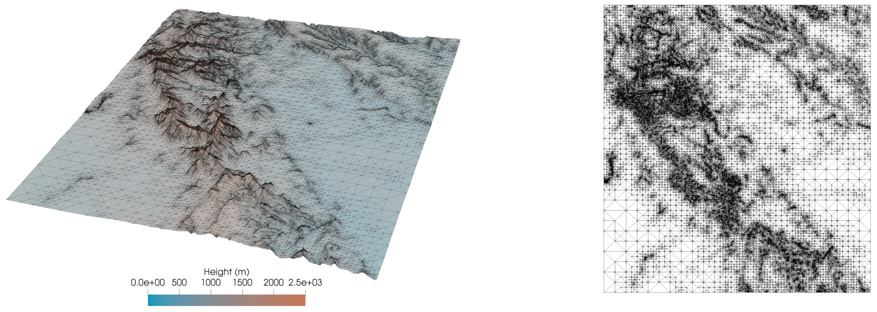


Fig. 1.2 Orography of an area of Comunidad de Madrid (Spain) and its corresponding parameterization space

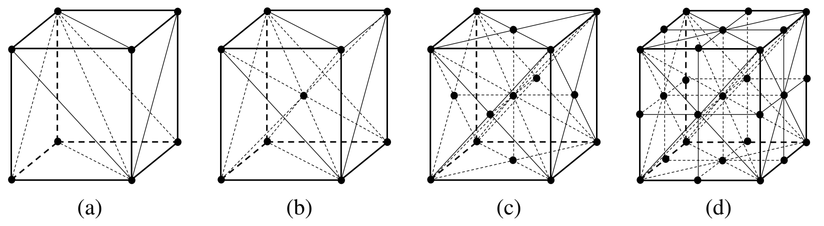


Fig. 1.3 Refinement of a cube by using Kossaczký's algorithm: (a) cube subdivision into six tetrahedra, (b) bisection of all tetrahedra by inserting a new node in the cube main diagonal, (c) new nodes in diagonals of cube faces and (d) global refinement with new nodes in cube edges

1.3.3 Coarse tetrahedral mesh of the meccano

We build a coarse and high quality tetrahedral mesh, $\mathscr{T}_0(\mathscr{M})$ by splitting the cuboid in cubes, and each cube is subdivided into six tetrahedra [16]. For this purpose, it is necessary to define a main diagonal on the cube and corresponding diagonal on its faces, see Figure 1.3(a). The resulting mesh can be recursively and globally bisected [16] for fixing a uniform element size in the whole mesh. Three consecutive global bisections for a cube are presented in Figures 1.3 (b), (c) and (d). The resulting mesh of Figure 1.3(d) contains 8 cubes similar to the one shown in Figure 1.3(a). Therefore, the recursive refinement of the cube mesh produces similar tetrahedra to the initial ones.

1.3.4 Approximation of the orography: refinement and distance evaluation

The next step of the meccano algorithm is to approximate the orography of the terrain with a prescribed tolerance ε. In fact, according to the choice of the meccano, see section 1.3.1, we only need to impose the approximation criteria on the bottom face of the meccano, $\partial\Omega_b$.

In order to obtain this approximation, the initial mesh $\mathscr{T}_0(\mathscr{M})$, is recursively refined. Specifically, we select the set of triangles on the bottom of the meccano, $T \in \partial\mathscr{T}(\mathscr{M}_b)$, that do not correctly approximate the terrain of the domain, that is

$$T \equiv \langle v_0, v_1, v_2 \rangle \in \partial\mathscr{T}(\mathscr{M}_b), \quad d(\langle \Pi(v_0), \Pi(v_1), \Pi(v_2) \rangle, \partial\Omega_b) > \varepsilon \tag{1.1}$$

where, $\langle \Pi(v_0), \Pi(v_1), \Pi(v_2) \rangle$ denotes the triangle that has as vertices $\{\Pi(v_i)\}_{i=0}^{2}$. Then we refine those triangles by dividing their adjacent tetrahedra using the Kossaczký method [16]. This procedure is iterated until the prescribed tolerance is reached.

We use the following strategy for a practical computation of the distance. Given $T \in \partial\mathscr{T}(\mathscr{M}_b)$, let $\{q_i\}_i^k$ be a set of Gauss quadrature points, then $d(\Pi(T), \partial\Omega_b)$ is estimated as

$$d(\langle \Pi(v_0), \Pi(v_1), \Pi(v_2) \rangle, \partial\Omega_b) \approx \max_{i=1,\ldots,k} d(\langle \Pi(v_0), \Pi(v_1), \Pi(v_2) \rangle, \Pi(q_i)). \quad (1.2)$$

Once the final tetrahedral mesh $\mathscr{T}(\mathscr{M})$ is generated in the cuboid, their boundary nodes are mapped by Π onto the boundary of the domain. After this step, low-quality and inverted elements may appear.

1.3.5 Relocation of inner nodes

There would be several strategies for defining a reasonable position for each inner node of the domain: laplacian smoothing, Coons patches, radial basis functions, etc.

Another effective possibility hinges on the volumetric mapping that produces the meccano method (see section 1.3.7). However, this information is not known a priori. In fact, we will reach this piecewise linear volume mapping at the end of the mesh generation.

In practice, a good strategy is: we start meshing the solid by using a high value of ε (a coarse tetrahedral mesh of the solid is obtained) and we continue decreasing it gradually. In the first step of this strategy, no relocation is applied. In this case, the number of nodes of the resulting mesh is low and the mesh optimization algorithm, that we describe below, is fast. In the following steps a relocation of inner nodes is applied by using the mapping (volumetric parameterization) that is defined by the previous iteration.

1.3.6 Simultaneous untangling and smoothing

Since the current mesh could be not valid (it could contain inverted element), it is necessary to optimize it. The process that we describe in this section [8, 9], must be able to smooth and untangle the mesh and is crucial in the proposed mesh generator.

Usual techniques to improve the quality of a *valid* mesh, that is, one that does not have inverted elements, are based upon local smoothing. In short, these techniques consist of finding the new positions that the mesh nodes must hold, in such a way that they optimize an objective function. Such a function is based on a certain measurement of the quality of the *local submesh* $N(q)$, formed by the set of tetrahedra connected to the *free node* q. Usually, objective functions are appropriate to improve the quality of a valid mesh, but they do not work properly when there are inverted elements. This is because they present singularities (barriers) when any tetrahedron of $N(q)$ changes the sign of its Jacobian.

Most of what is stated below is taken from [8], where we developed a procedure for untangling and smoothing tetrahedral meshes simultaneously. For that purpose, we use a suitable modification of the objective function such that it is regular all over $\mathscr{R}^3$. When a feasible region (subset of $\mathscr{R}^3$ where q could be placed, being $N(q)$ a valid submesh) exists, the minima of both the original and the modified objective functions are very close, and when this region does not exist, the minimum of the modified objective function is located in such a way that it tends to untangle $N(q)$. The latter occurs, for example, when the fixed boundary of $N(q)$ is tangled. With this approach, we can use any standard and efficient unconstrained optimization method to find the minimum of the modified objective function, see for example [1].

1.3.6.1 Objective functions

Several tetrahedron shape measures could be used to construct an objective function. Nevertheless, those obtained by algebraic operations [15] are specially indicated for our purpose because they can be computed very efficiently and they allow us to choose the shape of the tetrahedra to optimize. Our objective is to relocate the nodes of $\mathscr{T}$ in positions where not only the mesh gets untangled, but also the distortion introduced by the parameterization is minimized.

Let T be a tetrahedral element of $\mathscr{T}$ whose vertices are $\mathbf{x}_k = (x_k, y_k, z_k)^T \in \mathscr{R}^3$, $k = 0,1,2,3$ and T_R be the reference tetrahedron with vertices $\mathbf{u}_0 = (0,0,0)^T$, $\mathbf{u}_1 = (1,0,0)^T$, $\mathbf{u}_2 = (0,1,0)^T$ and $\mathbf{u}_3 = (0,0,1)^T$. If we choose $\mathbf{x}_0$ as the translation vector, the affine map that takes T_R to T is $\mathbf{x} = A\mathbf{u} + \mathbf{x}_0$, where A is the Jacobian matrix of the affine map referenced to node $\mathbf{x}_0$, and expressed as $A = (\mathbf{x}_1 - \mathbf{x}_0, \mathbf{x}_2 - \mathbf{x}_0, \mathbf{x}_3 - \mathbf{x}_0)$.

Let us consider that T_I is our ideal or target tetrahedron whose vertices are $\mathbf{v}_0$, $\mathbf{v}_1$, $\mathbf{v}_2$ and $\mathbf{v}_3$. If we take $\mathbf{v}_0 = (0,0,0)^T$ the linear map that takes T_R to T_I is $\mathbf{v} = W\mathbf{u}$, where $W = (\mathbf{v}_1 - \mathbf{v}_0, \mathbf{v}_2 - \mathbf{v}_0, \mathbf{v}_3 - \mathbf{v}_0)$ is its Jacobian matrix. As the parametric and real meshes are topologically identical, each tetrahedron of $\mathscr{T}$ has its counterpart in $\mathscr{C}_K$. Thus, in order to reduce the distortion in the volumetric parameterization we will fix the target tetrahedra of $N(q)$ as their counterparts of the local mesh in the parametric space.

The affine map that takes T_I to T is $\mathbf{x} = AW^{-1}\mathbf{v} + \mathbf{x}_0$, and its Jacobian matrix is $S = AW^{-1}$. Note that this weighted matrix S depends on the node chosen as reference, so this node must be the same for T and T_I. We can use matrix norms, determinant or trace of S to construct algebraic quality metrics of T. For example, the *mean ratio*, $Q = \frac{3\sigma^{\frac{2}{3}}}{|S|^2}$, is an easily computable algebraic quality metric of T, where $\sigma = \det(S)$ and $|S|$ is the Frobenius norm of S. The maximum value of Q is the unity, and it is reached when $A = \mu R W$, where μ is a scalar and R is a rotation matrix. In other words, Q is maximum if and only if T and T_I are similar. Besides, any flat tetrahedron has quality measure zero. We can derive an optimization function from this quality metric. Thus, let $\mathbf{x} = (x,y,z)^T$ be the position of the free node, and let S_m be the weighted Jacobian matrix of the m-th tetrahedron of $N(q)$. We define the distortion, η_m, of the m-th tetrahedron as the inverse of its quality:

$$\eta_m = \frac{|S_m|^2}{3\sigma_m^{\frac{2}{3}}} \tag{1.3}$$

Then, the corresponding objective function for $N(q)$ is constructed by using the p-norm of $(\eta_1, \eta_2, \ldots, \eta_M)$ as

$$\left|K_\eta\right|_p(\mathbf{x}) = \left[\sum_{m=1}^{M} \eta_m^p(\mathbf{x})\right]^{\frac{1}{p}} \tag{1.4}$$

where M is the number of tetrahedra in $N(q)$. We obtain the optimal position of the free node by minimizing (1.4).

Although this optimization function is smooth in those points where $N(q)$ is a valid submesh, it becomes discontinuous when the volume of any tetrahedron of $N(q)$ goes to zero. It is due to the fact that η_m approaches infinity when σ_m tends to zero and its numerator is bounded below. In fact, it is possible to prove that $|S_m|$ reaches its minimum, with strictly positive value, when q is placed in the geometric center of the fixed face of the m-th tetrahedron. The positions where q must be located to get $N(q)$ to be valid, i.e., the feasible region, is the interior of the polyhedral set P defined as $P = \bigcap_{m=1}^{M} H_m$,where H_m are the half-spaces defined by $\sigma_m(\mathbf{x}) \geq 0$. This set can occasionally be empty, for example, when the fixed boundary of $N(q)$ is tangled. In this situation, function $\left|K_\eta\right|_p$ stops being useful as an optimization function. Moreover, when the feasible region exists, that is *int* $P \neq \emptyset$, the objective function tends to infinity as q approaches the boundary of P. Due to these singularities, it is formed a barrier which avoids reaching the appropriate minimum when using gradient-based algorithms, and when these start from a free node outside the feasible region. In other words, with these algorithms we can not optimize a tangled mesh $N(q)$ with the above objective function.

1.3.6.2 Modified objective functions

We proposed in [8] a regularization in the previous objective function (1.4), so that the barrier associated with its singularities will be eliminated and the new function will be smooth all over $\mathscr{R}^3$. An essential requirement is that the minima of the original and modified functions are nearly identical when *int* $P \neq \emptyset$. Our modification consists of substituting σ in (1.3) by the positive and increasing function

$$h(\sigma) = \frac{1}{2}(\sigma + \sqrt{\sigma^2 + 4\delta^2}) \tag{1.5}$$

being the parameter $\delta = h(0)$. Thus, the new objective function here proposed is given by

$$\left|K_{\eta}^{*}\right|_{p}(\mathbf{x}) = \left[\sum_{m=1}^{M} (\eta_m^*)^p (\mathbf{x})\right]^{\frac{1}{p}} \tag{1.6}$$

where

$$\eta_m^* = \frac{|S_m|^2}{3h^{\frac{2}{3}}(\sigma_m)} \tag{1.7}$$

is the modified objective function for the m-th tetrahedron. With this modification, we can untangle the mesh and, at the same time, improve its quality. An implementation of the simultaneous untangling and smoothing procedure for an equilateral reference tetrahedron is freely available in [9].

1.3.7 Volumetric parameterization

One of the consequence of the meccano method is that it produces automatically a volumetric parametrization of the domain. This mapping $\boldsymbol{\Pi}$ is an extension of the surface parameterization Π built at Section 1.3.2:

$$\boldsymbol{\Pi} : \mathscr{M} \to \mathscr{T} \approx \Omega \tag{1.8}$$

where a point p included in a tetrahedron of $\mathscr{M}$ is mapped, preserving barycentric coordinates, into a point q belonging to the transformed tetrahedron of $\mathscr{T}$.

1.3.8 Meccano as surface mesher

Numerical simulation of some engineering/environmental problem, such as solar radiation (see Chapter **??**), only requires a surface mesh. The meccano algorithm, that we describe along this section, can be particularized to generate triangular meshes of complex terrains . In this case, the meccano is a rectangle, that plays the role of parametric space of the surface defined by the elevation map of the terrain. Then, Floater's strategy (Section 1.3.2) allows to build a mapping between the meccano and the terrain surface. Finally, an analogous approximation strategy to Section 1.3.4 provides the final surface mesh.

Since the Floater's parameterization is a quasi-conformal mapping, the smoothing step can be avoided in general. Nevertheless, the surface smoothing [7] could be used to improve the mesh quality.

1.4 Numerical examples

The performance of our new mesh generator is shown in the following applications. The first corresponds to a domain placed at La Palma (Canary Islands, Spain), of 10×30 km. In the second example we discretize a bigger domain: Gran Canaria (Canary Island, Spain), 60×70 km, and we analyze several approximations for different values of ε parameter. Finally, in the third example, we include a chimney in a region of Gran Canaria Island and build the associated mesh. In all cases, the topography is given by a digitalization of the area where heights are defined over a uniform grid with a spacing step of 25 m in directions x and y (raster data). The computations were done on an MacBook Pro with two processors, 3.5 GHz, Inter Core and 16 Gb RAM memory.

1.4.1 La Palma

We first consider a rectangular area in *Isla de La Palma* (Canary Islands, Spain) of 10×30 km. The upper boundary of the domain has been placed at 3 km. To define the topography we use a digitalization of the area where heights are defined over a uniform grid with a spacing step of 25 m in directions x and y. Therefore, the associated meccano is a cuboid of dimension $10 \times 30 \times 3$ km. After computing a boundary mapping, Π between the boundary of the meccano and the boundary of the domain (see section 1.3.2 for details), we divided it into $5 \times 15 \times 1$ cuboids. Each cuboid is subdivided into six tetrahedra by using the subdivision proposed in [16], see Figure 1.3. Then, we fix $\varepsilon = 5$ m, and begin the approximation procedure described at Section 1.3.4.

The resulting mesh has 75320 tetrahedra and 17664 nodes and it nears the terrain surface with an error less than $\varepsilon = 5$ m. The adaptive procedure performs 18 refinement steps (bisection). Besides, we relocate the inner nodes using the volumetric parameterization produced by a coarse ε ($\varepsilon = 100$ m.), and reduce the number of inverted tetrahedra from 23612 to 5266. Finally, we apply the optimization process of Section 1.3.6. The node distribution is hardly modified after 12 steps, resulting a valid and high quality mesh (the mesh is untangled after 8 iterations). We remark that we have not relocated those nodes placed on the terrain during this optimization process. In Fig. 1.4 we show several clips of the tetrahedral mesh along the generation procedure and in Fig. 1.5 the surface of the final mesh is presented.

The evolution of the mesh quality during the optimization process is showed in Figure 1.6. These curves are obtained by sorting the elements in increasing order of its quality. This measure tends to stagnate quickly. Note that the quality curves corresponding to the eighth and twelfth optimization steps are very close. The average quality measure increases to $\overline{q}_\kappa = 0.75$. After this optimization process, the worst quality measure of the optimized mesh tetrahedra is 0.25. In addition, we present in Fig. 1.7(a) the 'cumulative frequency polygon' of the generated mesh; for a given value of $x \in (0,1)$ the line represents the percentage of elements that have a quality

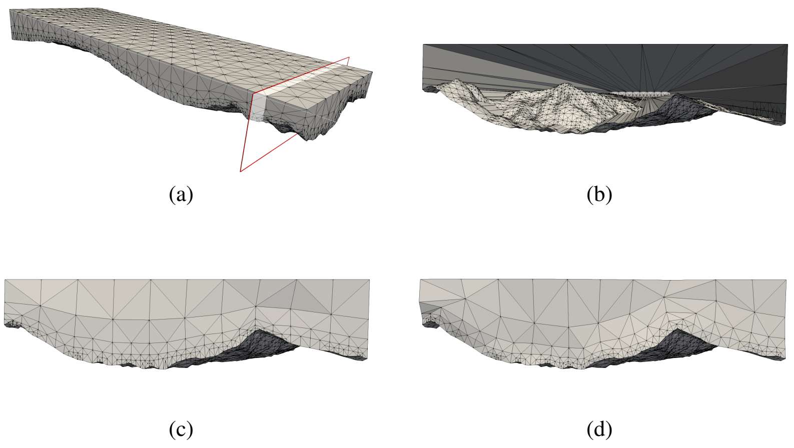


Fig. 1.4 a) Clip of the mesh along the algorithm: b) After node boundary projection, c) After inner node relocation, d) After smoothing (final mesh).

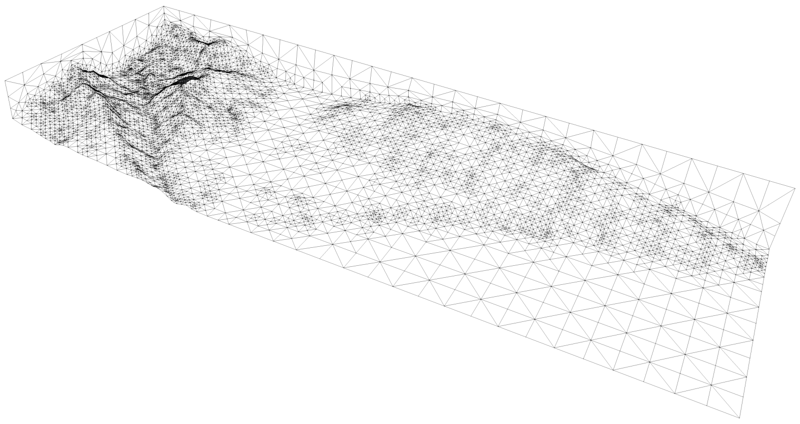
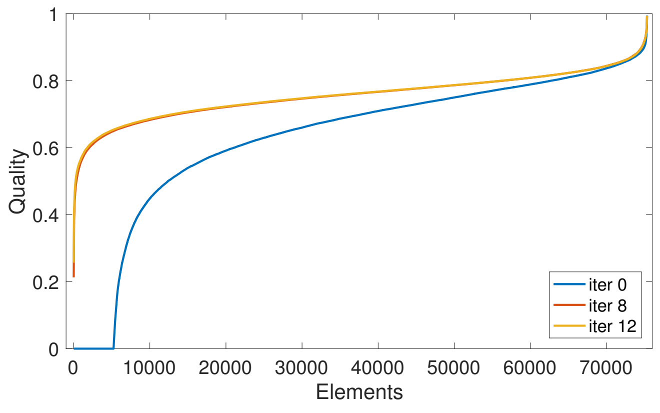


Fig. 1.5 Detail of *Isla de La Palma* (Canary Island): Surface of the final mesh.

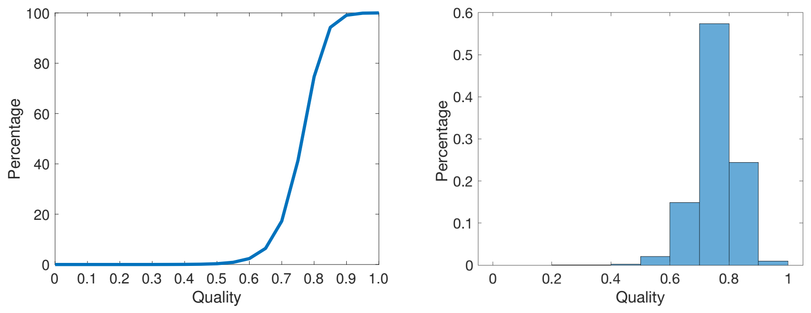
$\leq x$ and in Fig. 1.7(b) the quality histogram of the final mesh: where the height of each bar is the relative number of tetrahedra with quality associate to the bar. We note that the 80% of tetrahedra have a quality bigger than 0.70.

The total CPU time for the mesh generation is less than 20 seconds. In particular, the computational cost of the simultaneous untangling and smoothing procedure is about 8 seconds.



(a)

Fig. 1.6 Quality curves for the initial (blue line) and optimized meshes after 8th (red line) and 12th (orange line) iterations for the domain of La Palma (Canary Island).



(a) (b)

Fig. 1.7 a) Cumulative frequency polygon of the final mesh; for a given quality value $x \in (0,1)$ the line represents the percentage of elements that have a quality $\leq x$. b) Quality histogram of the final mesh. The height of each bar is the relative number of tetrahedra with quality associate to the bar.

1.4.2 Gran Canaria

We now approximate the orography of Gran Canaria (Canary Islands, Spain). We consider a rectangular area of 60×70 km, and fix the upper bound of the domain to 11 km. As in the previous case, the topography is defined by a rasted data, with 25 m as discretization step. In Fig. 1.8 we show the corresponding meccano and its initial triangulation, respectively.

We generate several meshes of Gran Canaria, for several values of the discretization parameter ε. Table 1.1 reports their main features. The volumetric parameterization that induces a tetrahedral mesh is used to relocate the inner nodes of the

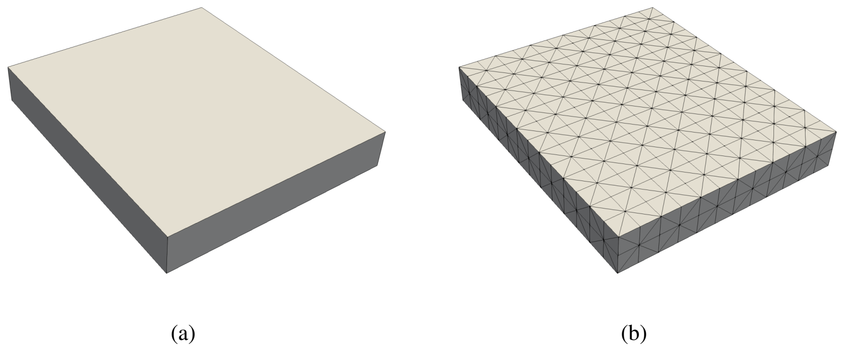


Fig. 1.8 Initial meccano of Gran Canaria Island (a) and its corresponding intial mesh (b).

following finer mesh. Note that neither the number of optimization steps nor the minimum quality depend drastically on ε. The CPU time includes surface parameterization, approximation and smoothing.

ε (m)	Tetrahedra	Nodes	Ref. level	SUS iter. (untang.+smooth.)	$q_{\min}$	$q_{\max}$	CPU Time (s.)
200	8760	2127	12	$3+6$	0.45	0.77	≈ 13
100	39800	9191	17	$3+6$	0.30	0.76	≈ 22
75	70572	16231	18	$2+5$	0.31	0.76	≈ 33
50	153252	35092	21	$2+5$	0.31	0.76	≈ 71
35	278674	63740	24	$3+6$	0.32	0.76	≈ 145
25	478420	109315	24	$3+6$	0.26	0.75	≈ 212

Table 1.1 Main features of meshes of Gran Canaria Island for several values of the discretization parameter ε.

In Fig. 1.9 we show the surface of the final mesh for $\varepsilon = 25$ m and some details of the discretization of Gran Canaria topography.

1.4.3 Industrial chimney

In this last example we show that meccano algorithm allows to include the geometry of building or stacks. We now consider an area located at north east of Gran Canaria, Spain (see Fig. 1.10(a)), where we introduce a chimney of a power plant. The dimension of the domain is $15 \times 25 \times 10$ km, and the chimney has 5 m. of radius and a height of 100 m. In order to get a good approximation of the chimney we define

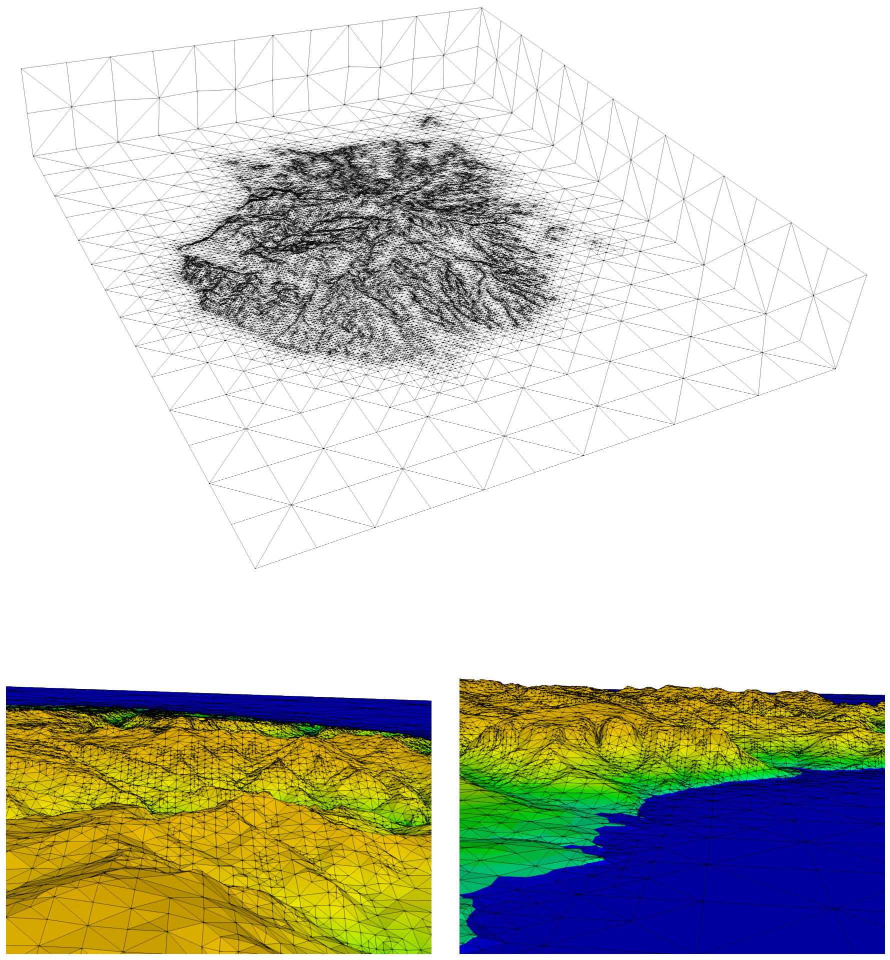


Fig. 1.9 Surface of the final mesh of Gran Canaria Island, for $\varepsilon = 50$m., and several details

the tolerance ε as :

$$\varepsilon(\mathbf{x}) = \begin{cases} 0.5\,m. & \text{if the distance from } \mathbf{x} \text{ to the chimney is less than 100 m.} \\ 5\,m. & \text{otherwise} \end{cases}$$

The adaptive procedure performs 96 refinement step (bisection) to reach the prescribed tolerance. The resulting mesh has 695678 tetrahedra and 159166 nodes and its surface is shown in Fig. 1.11.

In Fig. 1.12 some details of the chimney discretization are presented. Note that the features of the structure are proprertly approximated by our algorithm.

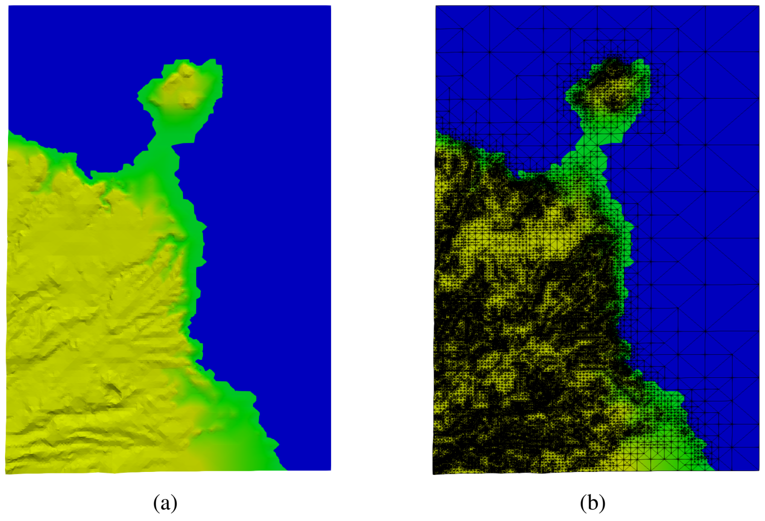


Fig. 1.10 (a) Map of the Gran Canaria's north east, and 2D mesh of the surface.

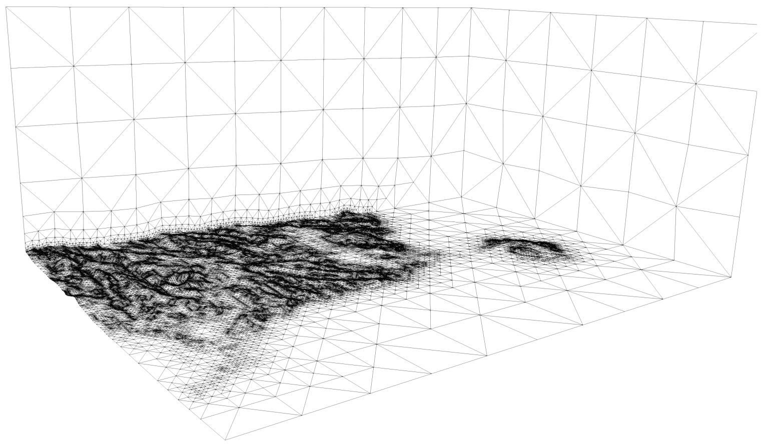


Fig. 1.11 Surface of the final mesh of Gran Canaria's north east.

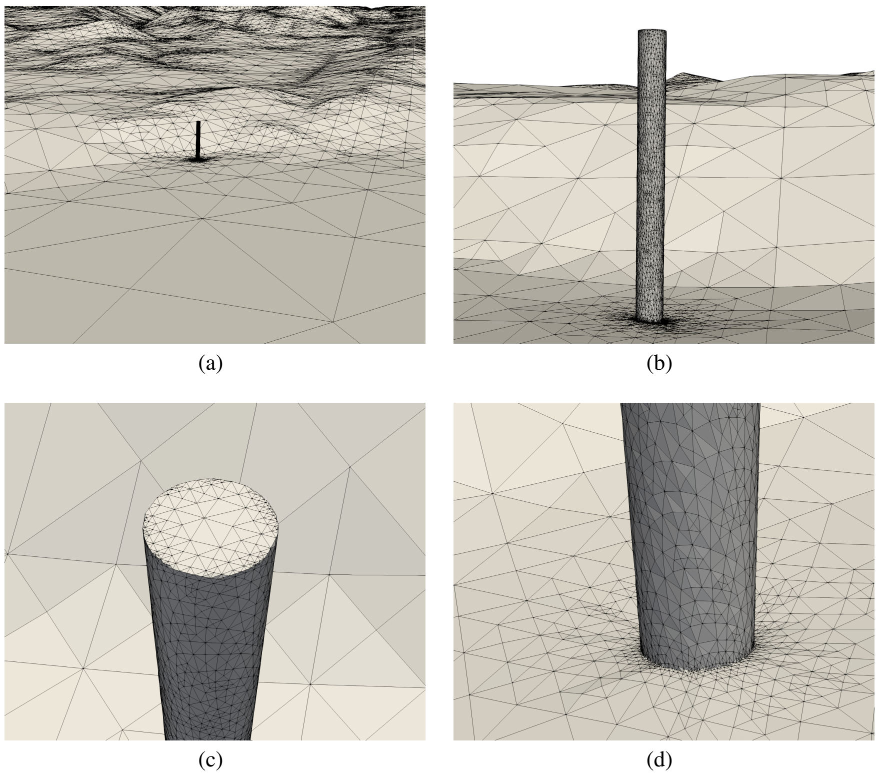


Fig. 1.12 Details of the chimney discretization.

After relocating the inner nodes, using the volumetric parameterization produced by a coarse ε, the optimization process only required 16 steps to untangle and smooth the mesh. The average quality measure of the final mesh is $\overline{q}_{\kappa} = 0.70$. We present in Fig. 1.13(a) the 'cumulative frequency polygon' of the generated mesh; for a given value of $x \in (0,1)$ the line represents the percentage of elements that have a quality $\leq x$ and in Fig. 1.13(b) the quality histogram of the final mesh: where the height of each bar is the relative number of tetrahedra with quality associate to the bar. We note that the 90% of tetrahedra has a quality bigger than 0.7. The CPU time to generate the final mesh is about 600 s. We remark that this time can be drastically reduced by using a parallel algorithm of the simultaneous untangling and smoothing procedure [2].

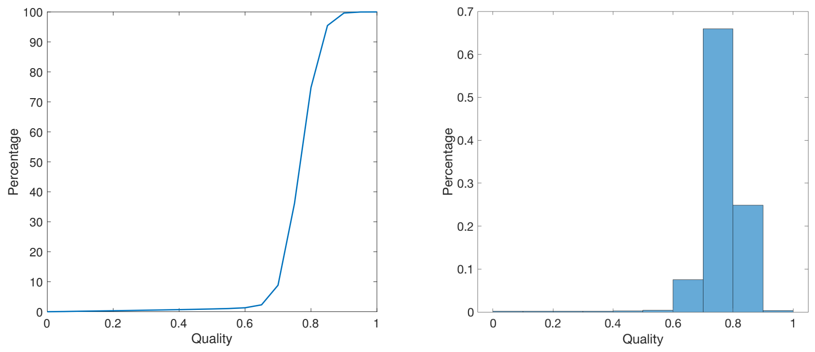


Fig. 1.13 a) Cumulative frequency polygon of the final mesh; for a given quality value $x \in (0,1)$ the line represents the percentage of elements that have a quality $\leq x$. b) Quality histogram of the final mesh. The height of each bar is the relative number of tetrahedra with quality associate to the bar.

Conclusions

We have established the main aspects to generate a tetrahedral mesh able to adapt to the topography of a rectangular area with a minimum intervention of users. Our three-dimensional domain is limited on its lower part by the terrain and on its upper part by a horizontal plane placed at a height where the magnitudes under study may be considered steady. The lateral walls are formed by four vertical planes. The main input data consist on a digital elevation map of the terrain and its desired approximation. The adaptive mesh is efficiently built by using the Meccano Method, that was previously introduced by the authors. The node distribution is obtained automatically in the domain under study, able to get the irregular topographic information of the terrain, and with a decreasing density as altitude increases in relation to the terrain. In addition, the meshes can be generated on a laptop with a low CPU time. Our procedure constructs high quality meshes that are appropriated to simulate environmental problems over complex terrain, in particular those analysed in this book: Wind simulation and solar radiation. In this last case, a surface adaptive triangulation of the terrain is only necessary in the numerical model. The mesh generation procedure can also consider the inclusion of stacks for air pollution simulation.

Acknowledgements

This research was partially supported by FEDER and the Spanish Government, "Ministerio de Economía y Competitividad" under grand contracts CTM2014-55014-C3-1-R, CTM2014-55014-C3-3-R, the Department of Education, Junta of Castilla y León, under grand contract SA020U16 and by CONACYT-SENER

"FondoSectorial CONACYT SENER HIDROCARBUROS" under grand contract 163723.

References

1. Bazaraa, M.S., Sherali, H.D., Shetty, C.M.: Nonlinear Programing: Theory and Algorithms. John Wiley and Sons Inc., New York (1993)
2. Benítez, D., Rodríguez, E., Escobar, J.M., Montenegro, R.: Performance evaluation of a parallel algorithm for simultaneous untangling and smoothing of tetrahedral meshes. In: Proceedings of the 22nd International Meshing Roundtable, pp. 579–598. Springer Science Business Media (2014). DOI 10.1007/978-3-319-02335-9_32
3. Carey, G.F.: Computational Grids: Generation, Adaptation and Solution Strategies. Taylor & Francis, Washington (1997)
4. Carey, G.F.: A perspective on adaptive modeling and meshing (AM&M). Computer Methods in Applied Mechanics and Engineering **195**(4-6), 214–235 (2006). DOI 10.1016/j.cma.2004.11.027. 1st ADMOS Conference 2003, Goteborg, SWEDEN, SEP, 2003
5. Cascón, J.M., Montenegro, R., Escobar, J.M., Rodríguez, E., Montero, G.: A new meccano technique for adaptive 3-d triangulations. In: Proceedings of the 16th International Meshing Roundtable, pp. 103–120. Springer-Verlag, Berlin, Germany (2007). DOI 10.1007/978-3-540-75103-8_6
6. Cascón, J.M., Rodríguez, E., Escobar, J.M., Montenegro, R.: Comparison of the meccano method with standard mesh generation techniques. Engineering with Computers **31**(1), 161–174 (2015). DOI 10.1007/s00366-013-0338-6
7. Escobar, J.M., Montero, G., Montenegro, R., Rodríguez, E.: An algebraic method for smoothing surface triangulations on a local parametric space. International Journal for Numerical Methods in Engineering **66**(4), 740–760 (2006). DOI 10.1002/nme.1584
8. Escobar, J.M., Rodríguez, E., Montenegro, R., Montero, G., González-Yuste, J.M.: Simultaneous untangling and smoothing of tetrahedral meshes. Computer Methods in Applied Mechanics and Engineering **192**(25), 2775–2787 (2003). DOI 10.1016/S0045-7825(03)00299-8
9. Escobar, J.M., Rodríguez, E., Montenegro, R., Montero, G., González-Yuste, J.M.: SUS code: simultaneous mesh untangling and smoothing code. http://www.dca.iusiani.ulpgc.es/SUScode (2010)
10. Floater, M.S.: Parametrization and smooth approximation of surface triangulations. Computer Aided Geometric Design **14**(3), 231 – 250 (1997). DOI 10.1016/S0167-8396(96)00031-3
11. Floater, M.S.: Mean value coordinates. Computer Aided Geometric Design **20**(1), 19–27 (2003). DOI 10.1016/S0167-8396(03)00002-5
12. Frey, P.J., George, P.L.: Mesh Generation. Hermes Science Publishing, Oxford (2000)
13. George, P.L., Borouchaki, H.: Delaunay Triangulation and Meshing: Application to Finite Elements. Editions Hermes, Paris (1998)
14. González-Yuste, J.M., Montenegro, R., Escobar, J.M., Montero, G., Rodríguez, E.: Local refinement of 3-d triangulations using object-oriented methods. Advances in Engineering Software **35**(10), 693 – 702 (2004). DOI 10.1016/j.advengsoft.2003.07.003. Engineering Computational Technology
15. Knupp, P.M.: Algebraic mesh quality metrics. SIAM J. Sci. Comput. **23**(1), 193–218 (2001). DOI 10.1137/s1064827500371499
16. Kossaczký, I.: A recursive approach to local mesh refinement in two and three dimensions. Journal of Computational and Applied Mathematics **55**(3), 275–288 (1994). DOI 10.1016/0377-0427(94)90034-5
17. Löhner, R., Baum, J.D.: Adaptive h-refinement on 3-D unstructured grids for transient problems. International Journal for Numerical Methods in Fluids **14**, 1407–1419 (1992). DOI 10.1002/fld.1650141204

18. Montenegro, R., Cascón, J.M., Escobar, J.M., Rodríguez, E., Montero, G.: An automatic strategy for adaptive tetrahedral mesh generation. Applied Numerical Mathematics **59**(9), 2203–2217 (2009). DOI 10.1016/j.apnum.2008.12.010
19. Montenegro, R., Cascón, J.M., Escobar, J.M., Rodríguez, E., Montero, G.: The meccano method for simultaneous volume parametrization and mesh generation of complex solids. IOP Conference Series: Materials Science and Engineering **10**(1), 012–018 (2010). DOI 10.1088/1757-899X/10/1/012018
20. Montenegro, R., Cascón, J.M., Rodríguez, E., Escobar, J.M., Montero, G.: The meccano method for automatic three-dimensional triangulation and volume parametrization of complex solids. In: B. Topping, J. Adam, F. Pallarés, R. Bru, M. Romero (eds.) Developments and Applications in Engineering Computational Technology, seventh edn., chap. 2, pp. 19–48. Saxe-Coburg Publications, Stirlingshire (2010). DOI 10.4203/csets.26.2
21. Oliver, A., Montero, G., Montenegro, R., Rodríguez, E., Escobar, J.M., Pérez-Foguet, A.: Adaptive finite element simulation of stack pollutant emissions over complex terrains. Energy **49**, 47 – 60 (2013). DOI 10.1016/j.energy.2012.10.051
22. Prieto, D., Asensio, M.I., Ferragut, L., Cascón, J.M., Morillo, A.: A gis-based fire spread simulator integrating a simplified physical wildland fire model and a wind field model. International Journal of Geographical Information Science **31**(11), 2142–2163 (2017). DOI 10.1080/13658816.2017.1334889
23. Rivara, M.: A grid generator bases on 4-triangles conforming mesh-refinement algorithms . International Journal for Numerical Methods in Engineering **24**(7), 1343–1354 (1987). DOI 10.1002/nme.1620240710
24. Thompson, J.F., Soni, B., Weatherill, N.: Handbook of Grid Generation. CRC Press, London (1999)