# Cartesian Grid Adaption in CFD

Marie-Hélène Lallemand

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

# *Cartesian grid adaption in CFD*

Marie-Hélène Lallemand

**No 3406**

Avril 98

THÈME 4


ISSN 0249-6399

INRIA
ROCQUENCOURT

# Cartesian grid adaption in CFD

Marie-Hélène Lallemand*

Thème 4 — Simulation et optimisation
de systèmes complexes
Projet M3N

Rapport de recherche n° 3406 — Avril 98 — 41 pages

**Abstract:** We present a single block adaption strategy for 3-D cartesian grids. The adaption process proceeds as an interface with the PHOENICS package. It allows continuous runs or unsteady calculations by reconstructing the interpolated solution on the new mesh at each adaption step. To perform the adaption, we define a new metric based on and proportional to the Hessian of the unknown variables. The adapted grid is defined by equidistributing the nodes. The interpolated solution is computed by using its barycenter coordinates together with its first order partial derivatives.

**Key-words:** Adaptive method, 3-D cartesian meshes, Finite Difference and Finite Element Methods, PHOENICS.

*(Résumé : tsvp)*

Ce travail a été réalisé dans le cadre du projet ESPRIT MICA 20966 et nous remercions la Commission Européenne de nous avoir permis de l'effectuer. Remerciements aussi à P. Le Tallec, F. Hecht et C. Martin qui ont chacun contribué en partie à la réalisation de ce projet, sans oublier J. Heritage de CHAM qui nous a fourni le software Phoenics ainsi que tous les modules d'interface nécessaires.

* Marie-Helene.Lallemand@inria.fr

Unité de recherche INRIA Rocquencourt
Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
Téléphone : 01 39 63 55 11 - International : +33 1 39 63 55 11
Télécopie : (33) 01 39 63 53 30 - International : +33 1 39 63 53 30

# Adaptation de maillages cartésiens en CFD

**Résumé :** On présente une stratégie d'adaptation monobloc pour des grilles cartésiennes 3-D. Elle est utilisée via une interface avec le code PHOENICS et permet d'effectuer des calculs évolutifs par la reconstruction d'une solution interpolée sur le nouveau maillage à chaque étape d'adaptation. L'adapation est effectuée en calculant une métrique définie comme étant proportionnelle au Hessien des inconnues du système. La grille adaptée est ensuite déterminée en equirépartissant les nœuds du maillage. L'interpolation des variables sur le nouveau maillage est calculée à l'aide des coordonnées barycentriques de l'ancien maillage et utilise ses dérivées partielles premières.

**Mots-clé :** Méthode adaptative, Maillages cartésiens 3-D, Méthodes d'Éléments Finis et Différences Finies, PHOENICS.

# 1 Introduction

The purpose of this report is to present a single block adaption strategy for 3-D cartesian grids to be used by the software PHOENICS. PHOENICS is a CFD software having a wide panel of physical models and devoted to the use of numerical simulations such as furnace modelling, environment problems, internal or external flows around many structures (buildings, marine structures etc...), multi-phase flow problems, turbulent flows etc ... That software has been chosen in the framework of the ESPRIT contract MICA (Model for Industrial CFD Applications N$^o$ 20 966) to create a CFD expertise network using the Internet for european small and medium enterprises. A virtual reality interface allows non-CFD engineers to pose their design and analysis problem solely in physical and geometrical terms. Our role is to create numerical tools to improve the efficiency of that software. Adaption is one of those tools that we describe in the present report.

# 2 Mesh adaption and PHOENICS

## 2.1 General strategy

The coupling between PHOENICS and the adaptation tool is represented on Fig. 1. The general strategy consists in :

1. reading the interface data file ("gradpt.old") provided by PHOENICS (using the PHOENICS input data file "EARDAT"), in which meshdata and the present value of the computed CFD solution is stored,
2. calculating an error estimate,
3. adapting the mesh by equidistributing the new nodes, region by region, in function of this error estimate,
4. writing a new interface data file ("gradpt.new").

The interaction with the PHOENICS code is done by some interface code "INTERFACE" written in Fortran that will change accordingly the old input data file "EARDAT" into a new one that will be read again by PHOENICS for the next run.

# 3 The adaption strategy

The main point of the adaption strategy is based on the definition of a new metric associated with a given mesh solution.

Indeed, during automatic mesh generation it is necessary to have as much information as possible on the nature and the local behaviour of the solution. Those indications governs

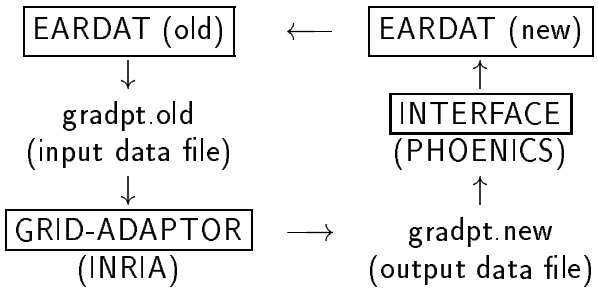


Figure 1: Mesh adaptation and PHOENICS.

the grid generation. As we are interested in the cell size, we need to convert somehow what we know about the solution to something having the dimension of a length and containing directional information. This might be done by giving at each mesh point, say $x$, six parameters in $\mathbb{R}^3$, describing a local metric tensor. Thus, the adaptation metric measures the ideal length (in terms of cell number) of the mesh in each direction at the considered point.

## 3.1 Definition of the metric

In a first stage, the metric can be expressed as a function of the absolute value of the hessian of the solution variables and it can be viewed as a measure of the local error (see [3], [4]). Let $u$ be a variable solution that we suppose known at each cell centers and let $H(u)$ be its hessian computed at each cell face. The metric $M(u)$ associated to $u$ can be simply defined as

$$M(u) = c_{met} \, |H(u)| \, ,$$

where $c_{met}$ is a user-defined constant depending on the mesh size (typically, $c_{met}$ is approximately proportional to the inverse of the mesh size) and $|H(u)|$ is the absolute value of the hessian of the scalar variable $u$. When more than one scalar variable is involved the different individual metrics are intersected following F. Hecht rule (see § 3.3 and [2]).

The main point is therefore to define the second order partial derivative of the solution variables. Since we are restricted to cartesian grids, we adopt a mixed approximation combining both Q1 Finite Element and Finite Difference approximations. Also, scalar variables are given by the PHOENICS package at each cell center and a typical mesh size chosen by the adaption is defined by the intervals joining two cell face centers in each direction. Those facts motivate us to choose Finite Difference approximation to define the second order partial derivatives at each face center of the mesh cells. This definition involves both cell

center and vertex values of the scalar variables but also cell center and face center values of the first order partial derivatives of those scalar variables.

### 3.1.1 Vertex values of the scalar variables

Let $u$ be any scalar variable given at each cell center of a given mesh with dimensions NX, NY and NZ. Let $\mathcal{C}_M$ be a given fluid cell of the mesh and denote $u_M$ or $u_C$ the cell center value of $u$. Then the value of $u$ at the vertex $s$ of cell $\mathcal{C}_M$ is defined by :

$$u(s) \simeq \sum_{M\ ,\ s \text{vertex of } \mathcal{C}_M} \left(u_M + \overline{\nabla} u_M \cdot \overrightarrow{c_M s}\right) / nw(s) \ ,$$

where $nw(s)$ is the number of fluid cells having $s$ as a common vertex, $c_M$ is the cell center of cell $\mathcal{C}_M$. The cell center value of the gradient of $u$ is defined by the following formula based on Finite Difference approximation :

$$\begin{aligned}
\frac{\partial u}{\partial x}(c_M) &\simeq \frac{1}{nc(c_M,1)} \left( \frac{u_{c_E} - u_{c_M}}{x_{c_E} - x_{c_M}} + \frac{u_{c_W} - u_{c_M}}{x_{c_W} - x_{c_M}} \right) \ , \\
\frac{\partial u}{\partial y}(c_M) &\simeq \frac{1}{nc(c_M,2)} \left( \frac{u_{c_N} - u_{c_M}}{y_{c_N} - y_{c_M}} + \frac{u_{c_S} - u_{c_M}}{y_{c_S} - y_{c_M}} \right) \ , \\
\frac{\partial u}{\partial z}(c_M) &\simeq \frac{1}{nc(c_M,3)} \left( \frac{u_{c_H} - u_{c_M}}{z_{c_H} - z_{c_M}} + \frac{u_{c_L} - u_{c_M}}{z_{c_L} - z_{c_M}} \right) \ ,
\end{aligned}$$

where $nc(M,l)$ is the total number of neighboring fluid cells in direction $l$ , $l = 1,2,3$. Notations $c_E$, $c_W$ (for the $x$-direction), $c_N$, $c_S$ (for the $y$-direction), $c_H$ and $c_L$ (for the $z$-direction) stand for the East, West, North, South, High and Low cell centers of the neighboring cells of $\mathcal{C}_M$ in the PHOENICS convention. If $(i,j,k)$ is the set of indexes representing cell $\mathcal{C}_M$, then similarly cells $\mathcal{C}_E$, $\mathcal{C}_W$, $\mathcal{C}_N$, $\mathcal{C}_S$, $\mathcal{C}_H$ and $\mathcal{C}_L$ may be respectively represented by the following index sets: $(i+1,j,k)$, $(i-1,j,k)$, $(i,j+1,k)$, $(i,j-1,k)$, $(i,j,k+1)$ and $(i,j,k-1)$ whenever those sets make sense.

By means of Q1 Finite Element interpolation, we may also define $u$ on any $(x,y,z)$ location of the mesh by :

$$u(x,y,z) \simeq \sum_{s\ \in \mathcal{S}} u(s)\, \psi_s(x,y,z) \ ,$$

where $\mathcal{S}$ is the set of all vertices of the mesh, $\psi_s$ is the Q1 basis function attached to vertex $s$. From this, we can derive the following cell center values of the first order partial derivatives of $u$, that we denote by $D_l(u)(c_M)$ for $l = 1,2,3$ :

$$D_l(u)(c_M) \stackrel{\text{def}}{=} \sum_{s\ \in \mathcal{S}} u(s)\, \psi_{s,l}(c_M) \ .$$

Since second order partial derivatives at cell face centers will be defined using also cell face center values of first order partial derivatives of $u$, we must define those quantities before. We

will do this by using linear interpolation. Let us denote by $\overline{D}_{,l}u(F)$ the first order partial derivatives of $u$ at a given cell face center $F$ in the direction $l$, $l = 1, 2, 3$. Keeping the PHOENICS convention, we will denote by $E$, $W$, $N$, $S$, $H$ and $L$ the six cell face centers of cell $\mathcal{C}_M$. Cell centers will be denoted by $c$ for $\mathcal{C}_M$, $c_E$ and $c_W$ for the East and West neighboring cells of $\mathcal{C}_M$, $c_N$ and $c_S$ for the North and South neighboring cells and finally $c_H$ and $c_L$ for the High and Low neighboring cells. Recall also that the index $M$ can be represented by the index set $(i, j, k)$.

1. East face :

   - If cell $(i, j, k)$ is such that $i <$ NX then set

     $$\alpha \stackrel{\text{def}}{=} (x_{c_E} - x_E)/(x_{c_E} - x_c)$$

     and define

     $$\overline{D}_{,l}u(E) \stackrel{\text{def}}{=} \alpha\, D_l(u)(c) + (1 - \alpha) D_l(u)(c_E) \ ;$$

   - If cell $(i, j, k)$ is such that $i =$ NX then

     $$\overline{D}_{,l}u(E) \stackrel{\text{def}}{=} D_l(u)(c) \ .$$

2. West face :

   - If cell $(i, j, k)$ is such that $i > 1$ then

     $$\overline{D}_{,l}u(W) \stackrel{\text{def}}{=} \overline{D}_{,l}u(E_{i-1,j,k}) \ ,$$

     where $E_{i-1,j,k}$ is the East face center of cell having $(i - 1, j, k)$ as its index set;
   - If cell $(i, j, k)$ is such that $i = 1$ then

     $$\overline{D}_{,l}u(W) \stackrel{\text{def}}{=} D_l(u)(c) \ .$$

3. North face :

   - If cell $(i, j, k)$ is such that $j <$ NY then set

     $$\alpha \stackrel{\text{def}}{=} (y_{c_N} - y_N)/(y_{c_N} - y_c)$$

     and define

     $$\overline{D}_{,l}u(N) \stackrel{\text{def}}{=} \alpha\, D_l(u)(c) + (1 - \alpha) D_l(u)(c_N) \ ;$$

   - If cell $(i, j, k)$ is such that $j =$ NY then

     $$\overline{D}_{,l}u(N) \stackrel{\text{def}}{=} D_l(u)(c) \ .$$

4. South face :

   - If cell $(i,j,k)$ is such that $j > 1$ then

   $$\overline{D}_{,l}u(S) \stackrel{\text{def}}{=} \overline{D}_{,l}u(N_{i,j-1,k}) \ ,$$

   where $N_{i,j-1,k}$ is the North face center of cell having $(i,j-1,k)$ as its index set;
   - If cell $(i,j,k)$ is such that $j = 1$ then

   $$\overline{D}_{,l}u(S) \stackrel{\text{def}}{=} D_l(u)(c) \ .$$

5. High face :

   - If cell $(i,j,k)$ is such that $k <$ NZ then set

   $$\alpha \stackrel{\text{def}}{=} (z_{c_H} - z_H)/(z_{c_H} - z_c)$$

   and define

   $$\overline{D}_{,l}u(H) \stackrel{\text{def}}{=} \alpha\, D_l(u)(c) + (1-\alpha) D_l(u)(c_H) \ ;$$

   - If cell $(i,j,k)$ is such that $k =$ NZ then

   $$\overline{D}_{,l}u(H) \stackrel{\text{def}}{=} D_l(u)(c) \ .$$

6. Low face :

   - If cell $(i,j,k)$ is such that $k > 1$ then

   $$\overline{D}_{,l}u(L) \stackrel{\text{def}}{=} \overline{D}_{,l}u(H_{i,j,k-1}) \ ,$$

   where $H_{i,j,k-1}$ is the High face center of cell having $(i,j,k-1)$ as its index set;
   - If cell $(i,j,k)$ is such that $k = 1$ then

   $$\overline{D}_{,l}u(L) \stackrel{\text{def}}{=} D_l(u)(c) \ .$$

Note that with those definitions given above, if one of the dimensions NX, NY or NZ is equal to one then the first order derivative with respect to that direction at the cell face centers are both equal to the cell center value of the first order derivative in the same direction.

### 3.1.2 Second order partial derivatives

Now that all the previous definitions have been set, we may define the second order partial derivatives of $u$ at each cell face centers by means of centered Finite Difference approximation, using the available values of the first order derivatives. We give below the complete definition of $D^2_{x,l}(u)(F)$, $D^2_{y,l}(u)(F)$ and $D^2_{z,l}(u)(F)$ $(l = x, y, z)$ denoting the second order partial derivatives of $u$ at a the face centers $F = E$ and $F = W$ of cell $\mathcal{C}_M$ as an example. The other definitions can be easily derived from those formulae.

1. East face :

   (a) Computation of $D^2_{x,l}u(E)$ :

   - If cell $(i, j, k)$ is such that $i <$ NX then first set

     $$\alpha = (x_{c_E} - x_E)/(x_{c_E} - x_c)$$

     and define

     $$D^2_{x,l}u(E) \stackrel{\text{def}}{=} \alpha \frac{\overline{D}_{,l}u(E) - D_l(u)(c)}{x_E - x_c} + (1-\alpha)\frac{D_l(u)(c_E) - \overline{D}_{,l}u(E)}{x_{c_E} - x_E} \ ;^{1}$$

   - If cell $(i, j, k)$ is such that $i =$ NX then

     $$D^2_{x,l}u(E) \stackrel{\text{def}}{=} \frac{\overline{D}_{,l}u(E) - \overline{D}_{,l}u(W)}{x_E - x_W} \ ;$$

   (b) Computation of $D^2_{y,l}u(E)$ :

   - If cell $(i, j, k)$ is such that $1 < j <$ NY then first set

     $$\alpha = (y_{E_N} - y_E)/(y_{E_N} - y_{E_S}) \ ,$$

     where $E_N$ is the East face center of cell having $(i, j+1, k)$ as its index set, $E_S$ is the East face center of cell having $(i, j-1, k)$ as its index set and define

     $$D^2_{y,l}u(E) \stackrel{\text{def}}{=} \alpha \frac{\overline{D}_{,l}u(E) - \overline{D}_{,l}u(E_S)}{y_E - y_{E_S}} + (1-\alpha)\frac{\overline{D}_{,l}u(E_N) - \overline{D}_{,l}u(E)}{y_{E_N} - y_E} \ ;$$

   - If cell $(i, j, k)$ is such that $j = 1$ and NY $> 1$ then

     $$D^2_{y,l}u(E) \stackrel{\text{def}}{=} \frac{\overline{D}_{,l}u(E_N) - \overline{D}_{,l}u(E)}{y_{E_N} - y_E} \ ;$$

[1] Observe that this formula, as the following ones, reduces to a standard 2 point centered difference scheme for an equidistributed mesh.

- If cell $(i,j,k)$ is such that $j = \mathrm{NY}$ and $\mathrm{NY} > 1$ then

$$D^2_{y,l}u(E) \stackrel{\mathrm{def}}{=} \frac{\overline{D}_{,l}u(E) - \overline{D}_{,l}u(E_S)}{y_E - y_{E_S}} \ ;$$

- If cell $(i,j,k)$ is such that $j = 1 = \mathrm{NY}$ then

$$D^2_{y,l}u(E) \stackrel{\mathrm{def}}{=} \frac{\overline{D}_{,l}u(N) - \overline{D}_{,l}u(S)}{y_N - y_S} = 0 \ ;$$

(c) Computation of $D^2_{z,l}u(E)$ :

- If cell $(i,j,k)$ is such that $1 < k < \mathrm{NZ}$ then first set

$$\alpha = (z_{E_H} - z_E)/(z_{E_H} - z_{E_L}) \ ,$$

where $E_H$ is the East face center of cell having $(i,j,k+1)$ as its index set, $E_L$ is the East face center of cell having $(i,j,k-1)$ as its index set and define

$$D^2_{z,l}u(E) \stackrel{\mathrm{def}}{=} \alpha \frac{\overline{D}_{,l}u(E) - \overline{D}_{,l}u(E_L)}{z_E - z_{E_L}} + (1-\alpha)\frac{\overline{D}_{,l}u(E_H) - \overline{D}_{,l}u(E)}{z_{E_H} - z_E} \ ;$$

- If cell $(i,j,k)$ is such that $k = 1$ and $\mathrm{NZ} > 1$ then

$$D^2_{z,l}u(E) \stackrel{\mathrm{def}}{=} \frac{\overline{D}_{,l}u(E_H) - \overline{D}_{,l}u(E)}{z_{E_H} - z_E} \ ;$$

- If cell $(i,j,k)$ is such that $k = \mathrm{NZ}$ and $\mathrm{NZ} > 1$ then

$$D^2_{z,l}u(E) \stackrel{\mathrm{def}}{=} \frac{\overline{D}_{,l}u(E) - \overline{D}_{,l}u(E_L)}{z_E - z_{E_L}} \ ;$$

- If cell $(i,j,k)$ is such that $k = 1 = \mathrm{NZ}$ then

$$D^2_{z,l}u(E) \stackrel{\mathrm{def}}{=} \frac{\overline{D}_{,l}u(H) - \overline{D}_{,l}u(L)}{z_H - z_L} = 0 \ ;$$

2. West face :

(a) If cell $(i,j,k)$ is such that $i > 1$ :

$$D^2_{p,l}u(W) \stackrel{\mathrm{def}}{=} D^2_{p,l}u(E_{i-1,j,k}) \ ,$$

where $E_{i-1,j,k}$ is the East face center of cell having $(i-1,j,k)$ as its index set;

(b) If cell $(i,j,k)$ is such that $i = 1$ :

- Computation of $D^2_{x,l}u(W)$ :

$$D^2_{x,l}u(W) \stackrel{\text{def}}{=} \frac{\overline{D}_{,l}u(E) - \overline{D}_{,l}u(W)}{x_E - x_W} \ ;$$

- Computation of $D^2_{y,l}u(W)$ :
  i. If cell $(i,j,k)$ is such that $1 < j < \text{NY}$ then first set

$$\alpha = (y_{W_N} - y_N)/(y_{W_N} - y_{W_S})$$

  where $W_N$ is the West face center of cell having $(i,j+1,k)$ as its index set, $W_S$ is the West face center of cell having $(i,j-1,k)$ as its index set and define

$$D^2_{y,l}u(W) \stackrel{\text{def}}{=} \alpha \frac{\overline{D}_{,l}u(W) - \overline{D}_{,l}u(W_S)}{y_W - y_{W_S}} + (1-\alpha)\frac{\overline{D}_{,l}u(W_N) - \overline{D}_{,l}u(W)}{y_{W_N} - y_W} \ ;$$

  ii. If cell $(i,j,k)$ is such that $j = 1$ and $\text{NY} > 1$ then

$$D^2_{y,l}u(W) \stackrel{\text{def}}{=} \frac{\overline{D}_{,l}u(W_N) - \overline{D}_{,l}u(W)}{y_{W_N} - y_W} \ ;$$

  iii. If cell $(i,j,k)$ is such that $j = \text{NY}$ and $\text{NY} > 1$ then

$$D^2_{y,l}u(W) \stackrel{\text{def}}{=} \frac{\overline{D}_{,l}u(W) - \overline{D}_{,l}u(W_S)}{y_W - y_{W_S}} \ ;$$

  iv. If cell $(i,j,k)$ is such that $j = 1 = \text{NY}$ then

$$D^2_{y,l}u(W) \stackrel{\text{def}}{=} \frac{\overline{D}_{,l}u(N) - \overline{D}_{,l}u(S)}{y_N - y_S} = 0 \ .$$

- Computation of $D^2_{z,l}u(W)$ :
  i. If cell $(i,j,k)$ is such that $1 < k < \text{NZ}$ then first set

$$\alpha = (z_{W_H} - z_W)/(z_{W_H} - z_{W_L}) \ ,$$

  where $W_H$ is the West face center of cell having $(i,j,k+1)$ as its index set and $W_L$ is the West face center of cell having $(i,j,k-1)$ as its index set and define

$$D^2_{z,l}u(W) \stackrel{\text{def}}{=} \alpha \frac{\overline{D}_{,l}u(W) - \overline{D}_{,l}u(W_L)}{z_W - z_{W_L}} + (1-\alpha)\frac{\overline{D}_{,l}u(W_H) - \overline{D}_{,l}u(W)}{z_{W_H} - z_W} \ ;$$

ii. If cell $(i,j,k)$ is such that $k = 1$ and NZ $> 1$ then

$$D^2_{y,l}u(W) \stackrel{\text{def}}{=} \frac{\overline{D}_{,l}u(W_H) - \overline{D}_{,l}u(W)}{z_{W_H} - z_W} \ ;$$

iii. If cell $(i,j,k)$ is such that $k =$ NZ and NZ $> 1$ then

$$D^2_{y,l}u(W) \stackrel{\text{def}}{=} \frac{\overline{D}_{,l}u(W) - \overline{D}_{,l}u(W_L)}{z_W - z_{W_L}} \ ;$$

iv. If cell $(i,j,k)$ is such that $k = 1 =$ NZ then

$$D^2_{y,l}u(W) \stackrel{\text{def}}{=} \frac{\overline{D}_{,l}u(H) - \overline{D}_{,l}u(L)}{z_H - z_L} = 0 \ .$$

## 3.2 Definition of the weighted metric

Once all second order derivatives have been computed, we define a weighted (adimensional) hessian at each cell face center point. Let $H(u)^F_M$ be the hessian associated with $u$ defined on the face center $F$ of cell $\mathcal{C}_M$ and denote by $u_{ref}$ a reference value of $u$ (that will be defined later), then the weighted hessian denoted by $\overline{H}(u)^F_M$ is defined as:

$$\overline{H}(u)^F_M = \frac{H(u)^F_M}{u_M + \varepsilon\, u_{ref}} \ ,$$

where $u_M$ is the cell center value of $u$ and $\varepsilon$ is a positive real number which is user defined (a typical value would be $10^{-5}$). The reference value $u_{ref}$ is defined as a function of the reference enthalpy $h_{ref}$ which can either be defined by its average value over the computational domain or by

$$h_{ref} = E_{ref} + p_{ref}/\rho_{ref} \ ,$$

if the reference total energy $E_{ref}$, the reference pressure $p_{ref}$ and the reference density $\rho_{ref}$ are available. We define the reference values of the different variables by:

- Pressure variable:

$$p_{ref} = \rho_{ref}\, h_{ref} \ ;$$

- Velocity component:

$$v_{ref} = (h_{ref})^{\frac{1}{2}} \ ,$$

- Kinetic energy of turbulence (if dealing with turbulent flow problems):

$$k_{ref} = h_{ref} \ ,$$

- Rate of dissipation of turbulence kinetic energy (if dealing with turbulent flow problems) :

$$Ep_{ref} = (h_{ref})^{\frac{3}{2}}/L_{ref} \ ,$$

where $L_{ref}$ is a reference length typical to the turbulence phenomenom.

If we are dealing with multi-phase flow problem then a reference enthalpy and a reference density should be provided for each phase.

Once the weighted hessian is defined, we first symmetrize and diagonalize it. The diagonalized form at every point $x \in \Omega \subset \mathbb{R}^3$ is given by :

$$\mathcal{M}(x) = \mathcal{R}(x) \begin{pmatrix} \lambda_1(x) & 0 & 0 \\ 0 & \lambda_2(x) & 0 \\ 0 & 0 & \lambda_3(x) \end{pmatrix} \mathcal{R}(x)^{-1}, \tag{1}$$

where $\lambda_i(x) \geq 0$ , $i = 1, \ldots, 3$ are the $\mathcal{M}(x)$ eigen-values and $\mathcal{R}(x)$ is the rotation matrix of angle $\alpha(x)$ that maps the $\mathbb{R}^3$ canonical basis over the $\mathcal{M}(x)$ unit eigen-vectors.

Then we compute the absolute value of the hessian by replacing the eigenvalues by their absolute values in (1).

**Remark 3.1** *In the metric definition, we have to introduce the maximum and minimum edge lengths we tolerate in the mesh to remove the cases of unrealistic metrics that sometimes we might get. This is not really a restriction as we usually have a good idea of what these quantities should be.*

## 3.3 Intersection of metrics

The final local metric is defined by intersecting all variable-wise local metric (see also [1]). We can associate an ellipsoid $\mathcal{E}_{\mathcal{M}}$ to the metric tensor $\mathcal{M}$ having as axes $d_1$, $d_2$ and $d_3$ (eigen-vectors of $\mathcal{M}$) and with respective lengths $1/\sqrt{\lambda_i}$, $i = 1, \ldots, 3$. This ellipsoid $\mathcal{E}_{\mathcal{M}}$ is the unit sphere in this geometry.

Suppose now, that several variables $\eta_1$, $\eta_2, \ldots, \eta_r$ are given, the problem becomes: *find the metric so that the maximum interpolation error is minimized for all the variables.* It is clear, from the geometrical identification ellipsoid–metric that the solution to the previous minimization problem is to find the biggest ellipsoid contained in the intersection of all the ellipsoids $\mathcal{E}_1$, $\mathcal{E}_2, \ldots, \mathcal{E}_r$ corresponding to metrics $\mathcal{M}_1$, $\mathcal{M}_2, \ldots, \mathcal{M}_r$ computed from the variables $\eta_1$, $\eta_2, \ldots, \eta_r$. It is not easy in general to find the optimal solution of this problem. However, the following algorithm seems to be suitable enough (see fig. 2 depicted in the 2-D case). Suppose that initially only two variables ($\eta_1$ and $\eta_2$) are provided. We find an approximation of the optimal intersection ellipsoid by the following procedure :

- if the two ellipsoid curves $\mathcal{E}_1$, $\mathcal{E}_2$ have not any intersection point (this happens when one ellipsoid is contained into another), the one with smallest area is taken as intersection and the suitable metric is the associated one;

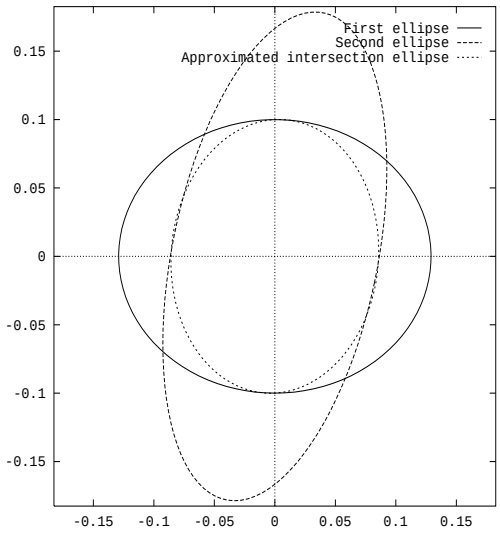


Figure 2: Approximated optimal ellipse of two-ellipse intersection.

- otherwise, let $\lambda_i^j$ and $v_i^j$, $i = 1, \ldots, 3$, $j = 1, 2$ the eigen-values and eigen-vectors (resp.) of $\mathcal{M}_j$, $j = 1, 2$. The intersection metric, $\hat{\mathcal{M}}$, is defined by

$$\hat{\mathcal{M}} = \frac{\hat{\mathcal{M}}_1 + \hat{\mathcal{M}}_2}{2}. \tag{2}$$

  where $\hat{\mathcal{M}}_1$ (resp. $\hat{\mathcal{M}}_2$) has the same eigen-vectors as $\mathcal{M}_1$, $(v_1^1,\ v_2^1,\ v_3^1)$ and has for eigen-values

$$\tilde{\lambda_i^1} = \max(\lambda_i^1, {v_i^1}^T \mathcal{M}_2 v_i^1), \quad i = 1, \ldots, 3\ . \tag{3}$$

  In other words, we compare the $i$-th eigenvalue of $\mathcal{M}_1$ with the Rayleigh quotient of its corresponding eigenvector in $\mathcal{M}_2$.

Now, if $n$ variables are given, the final intersection metric is computed recursively as follows:

- $\hat{\mathcal{M}} = \text{intersection}(\mathcal{M}_1, \mathcal{M}_2)$.
- for $i = 3, \ldots, n$
  $\hat{\mathcal{M}} = \text{intersection}(\hat{\mathcal{M}}, \mathcal{M}_i)$.

## 3.4 Definition of the mesh size in the new metric

Now that the metric $\mathcal{M}$ is defined at each cell face center, we must compute the lengths of the mesh size to start the adaption process. Let us first recall the definition of the measure of a parametric curve in the metric $\mathcal{M}$.

Let $A$ and $B$ be two given points of a bounded domain $\Omega$ in $\mathbb{R}^3$ and $O$ the origin. The measure of the parametric curve $\gamma(t) = (1-t)\,\overrightarrow{OA} + t\,\overrightarrow{OB}$ with $t \in [0,1]$ in the local metric $\mathcal{M}$ is written by :

$$L(\gamma) \stackrel{\text{def}}{=} \int_0^1 \sqrt{\gamma'(t)^T\,\mathcal{M}(\gamma(t))\,\gamma'(t)}\,dt \ ,$$

where $\mathcal{M}(\gamma(t))$ is the previously computed local adapted metric. Just looking at this segment $[A,B]$ locally in one dimension and since the ideal length we want to reach in the adapted segment should be equal to one, the nearest integer value of that measure will give us the number of nodes that should be added (if that measure is greater than one) or removed (if it is less than one and in the sense that this gives us the factor from which this segment should be enlarged) in the interval defined by $A$ and $B$. Since we are dealing with cartesian mesh, the points $A$ and $B$ will define the initial cell length in each direction ($x$, $y$ and $z$) in which refinement or coarsening should be performed. The final process will consist in defining first a quasi-local directional mesh length (for each direction): for e.g the $x$-direction, we will define a $x$-length that will only depend on the $i$-location, so taking the maximum over $j$ and $k$ of the local $x$-lengths answers this requirement. To know the total number of adapted cells in the $x$-direction, we just take the sum over $i$ to get the total length in the new metric of the segment $[0, \mathrm{NX}]$. The nearest integer approaching this length will give us the new number of $x$-cells in the new adapted mesh. We apply the same process for the two other directions. Returning back to our local segment $[A,B]$, the points $A$ and $B$ represent the face centers of each cell $\mathcal{C}_{i,j,k}$ and chosen for each direction by :

- $x$-direction :

$$A \stackrel{\text{def}}{=} W \ , \ B \stackrel{\text{def}}{=} E \ ,$$

- $y$-direction :

$$A \stackrel{\text{def}}{=} S \ , \ B \stackrel{\text{def}}{=} N \ ,$$

- $z$-direction :

$$A \stackrel{\text{def}}{=} L \ , \ B \stackrel{\text{def}}{=} H \ .$$

We need for the following to set some convention notations first (see Fig. 3) :

$$\begin{array}{llll} 1 \equiv \begin{pmatrix} i-1 \\ j-1 \\ k-1 \end{pmatrix} & 2 \equiv \begin{pmatrix} i \\ j-1 \\ k-1 \end{pmatrix} & 3 \equiv \begin{pmatrix} i \\ j \\ k-1 \end{pmatrix} & 4 \equiv \begin{pmatrix} i-1 \\ j \\ k-1 \end{pmatrix} \\ 5 \equiv \begin{pmatrix} i-1 \\ j-1 \\ k \end{pmatrix} & 6 \equiv \begin{pmatrix} i \\ j-1 \\ k \end{pmatrix} & 7 \equiv \begin{pmatrix} i \\ j \\ k \end{pmatrix} & 8 \equiv \begin{pmatrix} i-1 \\ j \\ k \end{pmatrix} \end{array} \qquad (4)$$

and each face is denoted as

| | |
|---|---|
| $x$-direction | face $(2,3,7,6) \equiv$ E (east)<br>face $(1,4,8,5) \equiv$ W (west) |
| $y$-direction | face $(4,8,7,3) \equiv$ N (north)<br>face $(1,5,6,2) \equiv$ S (south) |
| $z$-direction | face $(1,5,6,2) \equiv$ H (high)<br>face $(1,2,3,4) \equiv$ L (low) |

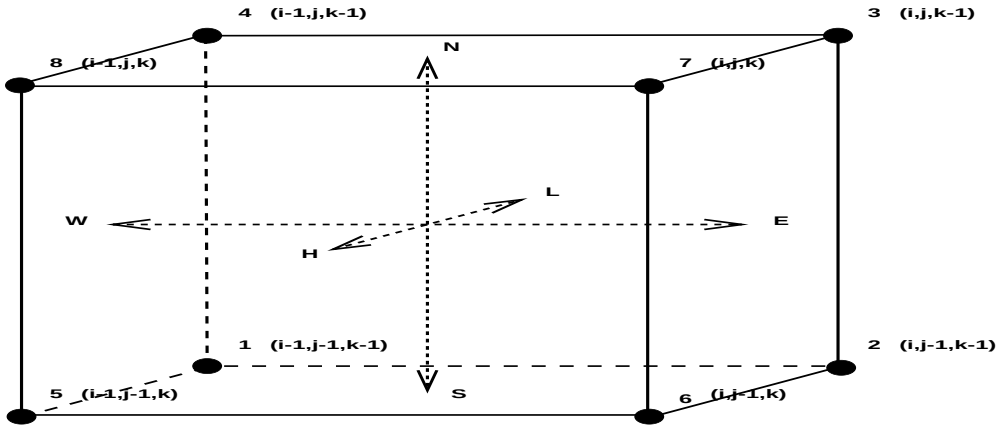


Figure 3: Cell $\mathcal{C}_{i,j,k}$

Since $\mathcal{M}$ is not known everywhere but only on the cell face center locations, we approximate $L(\gamma)$ by assuming the local metric to be linear on each of the segments $[A,B]$ (see [5]). Let us denote by

$$M_A \stackrel{\text{def}}{=} \mathcal{M}(\gamma(0)) \ , \ M_B \stackrel{\text{def}}{=} \mathcal{M}(\gamma(1)) \ ,$$

and set

$$\left\{ \begin{array}{lll} l_A^2 & = \overrightarrow{AB}^T \, \mathcal{M}(\gamma(0)) \, \overrightarrow{AB} & = \overrightarrow{AB}^T \, M_A \, \overrightarrow{AB} \ , \\ l_B^2 & = \overrightarrow{AB}^T \, \mathcal{M}(\gamma(1)) \, \overrightarrow{AB} & = \overrightarrow{AB}^T \, M_B \, \overrightarrow{AB} \ . \end{array} \right.$$

For example, $l_W^2$ would be equal to $\overrightarrow{EW}^T \mathcal{M}(\gamma(W)\,\overrightarrow{EW}$. Since $\mathcal{M}(\gamma(t)) \simeq M_A + t\,(M_B - M_A)$, we have

$$\begin{aligned} L(\gamma) &= \int_0^1 \left(l_A^2 + t\,(l_B^2 - l_A^2)\right)^{\frac{1}{2}}\,dt \\ &= \frac{2}{3}\,\frac{1}{l_B^2 - l_A^2}\left[\left(l_A^2 + t\,(l_B^2 - l_A^2)\right)^{\frac{3}{2}}\right]_0^1 \quad \text{if } l_A \neq l_B \\ &= \frac{2}{3}\,\frac{l_B^3 - l_A^3}{l_B^2 - l_A^2} \\ &= \frac{2}{3}\,\frac{l_A^2 + l_A\,l_B + l_B^2}{l_A + l_B} \end{aligned}$$

We finally get the following approximation :

$$L(\gamma) \simeq \frac{2}{3}\,\frac{l_A^2 + l_A\,l_B + l_B^2}{l_A + l_B}\ ,$$

from which we can deduce, for a given cell $\mathcal{C}_{i,j,k}$ the cell lengths in each direction by :

- In the $x$-direction : $(W \to E)$

$$L_X(\gamma_{i,j,k}^{W\to E}) = \frac{2}{3}\,\frac{l_W^2 + l_W\,l_E + l_E^2}{l_W + l_E}\ .$$

- In the $y$-direction : $(S \to N)$

$$L_Y(\gamma_{i,j,k}^{S\to N}) = \frac{2}{3}\,\frac{l_S^2 + l_S\,l_N + l_N^2}{l_S + l_N}\ .$$

- In the $z$-direction : $(L \to H)$

$$L_Z(\gamma_{i,j,k}^{L\to H}) = \frac{2}{3}\,\frac{l_L^2 + l_L\,l_H + l_H^2}{l_L + l_H}\ .$$

Since we want to keep on dealing with cartesian grids, those definitions above should not depend on the location of the cells but only on the coordinate component associated with the corresponding direction i.e. the new lengths that will be used for the adaption process will be finally defined as the most demanding length among all possible edges in that direction :

- $x$-direction: $i = 1, \ldots, \mathrm{NX}$

$$L_X(i) \stackrel{\text{def}}{=} \max_{j,k} L_X(\gamma_{i,j,k}^{W\to E})\ ,$$

- $y$-direction: $j = 1, \ldots, \mathrm{NY}$

$$L_Y(j) \stackrel{\text{def}}{=} \max_{i,k} L_Y(\gamma^{S\to N}_{i,j,k}) \ ,$$

- $z$-direction: $k = 1, \ldots, \mathrm{NZ}$

$$L_Z(k) \stackrel{\text{def}}{=} \max_{i,j} L_Z(\gamma^{L\to H}_{i,j,k}) \ .$$

The local lengths are stored in the file named "metloc.gen.out" to give an idea of the global quality of the mesh to the user, for each direction $x$, $y$ and $z$ with the following format (that can be used by a graphic software such as "gnuplot":

$i = 1, \ldots, \mathrm{NX}$ , $j = 1, \ldots, \mathrm{NY}$ , $k = 1, \ldots, \mathrm{NZ}$ ,

$$XC(i,j,k)\,,\ YC(i,j,k)\,,\ ZC(i,j,k)\,,\ L_Y(\gamma^{S\to N}_{i,j,k}) \ ,$$

where $XC, YC, YC$ are the cell face center coordinates. The new lengths in the $x$ (resp. $y$ and $z$) direction are stored in the files named "metxloc.out" (resp. "metyloc.out" and "metzloc.out") for a given plane (e.g. $x-y$ plane defined by a given $k = k_0$, $0 \le k_0 \le \mathrm{NZ}$) with the following format (here also given for the use of the graphic software "gnuplot") :

$i = 1, \ldots, \mathrm{NX}$ and $j = 1, \ldots, \mathrm{NY}$

$$XC(i,j,k_0)\,,\ YC(i,j,k_0)\,,\ L_X(i)(\text{resp. } L_Y(j))$$

# 4 The adaption process

## 4.1 Computation of the new coordinates

Once all mesh lengths are computed, the adaption consists in equidistributing the new nodes with respect to these new local lengths. For each direction, e.g $x$-direction, we first compute the maximal number of nodes denoted by $N_x^{new}$ as it follows :

$$N_x^{new} = \mathrm{INT}\,(LG_x(N_x)) \ ,$$

where $\mathrm{INT}\,(\beta)$ is the nearest integer for the given positive real number $\beta$ and $LG_x(i)$, $i = 0, \ldots, N_x$ is defined as the total adapted length of the path joining the origin of the domain to the present node in the $x$ direction, i.e. it is defined as the following cumulated length :

$$LG_x(i) = \sum_{k_x=0}^{k_x=i} L_X(k_x) \ , \tag{5}$$

with the convention $LG_x(0) = 0$. The same is done for the other directions leading to the new dimensions $N_x^{new}$, $N_y^{new}$ and $N_z^{new}$ for the adapted mesh. The coordinates of those new nodes are defined region by region. The nodes delimitating each region remain fixed during the adaption process. We just recall that in softwares such as Phoenics, the regions are first defined locally direction by direction. For the $x$-direction, a region of index $irx$, $irx = 1, \ldots,$ NXREG, is defined by its cell limit indexes $ir1, ir2$ where $ir1$ is the $x$-index of the cell starting the region of index $irx$ and $ir2$ is the $x$-index of the cell ending that region: those information are provided by a table of length $2 *$ NXREG, denoted by IREGN, such that

$$\text{IREGN}(2 * irx) = i_{beg} \ , \ \text{IREGN}(2 * irx + 1) = i_{end} \ , \ irx = 1, \ldots, \text{NXREG} \ ,$$

where $i_{beg}$ (resp. $i_{end}$) is the first (resp. last) cell index starting (resp. ending) the $x$-region $irx$. Same information are provided for the $y$ and $z$ directions and corresponding tables are denoted by JREGN (of dimension $2 *$ NYREG) and KREGN (of dimension $2 *$ NZREG) respectively. For ease of programming, we create an additional table denoted by IRGLIM$(irg, idir, ilim, im)$ , $irg = 1, \ldots,$ NREG, $idir = 1, \ldots, 3$, $ilim = 1, 2$, $im = 1, 2$ that gives us the cell index limits (first index corresponding to $ilim = 1$ and last index to $ilim = 2$) of a global region $irg$ in the direction $idir$ for the old mesh ($im = 1$) and new adapted mesh ($im = 2$) and where NREG = NXREG $*$ NYREG $*$ NZREG. Given a triplet of indices $(irx, iry, irz)$, we can define the associated global region of index $irg$ by setting

$$irg = \text{NZREG} * ( \text{NYREG} * (irx - 1) + (iry - 1) ) + irz \ .$$

The above formula is simply derived by renumbering triplet index entities to single index entities in cartesian (and more generally structured) meshes: e.g the global region $irg = 1$ corresponds indeed to the region having $(irx = 1, iry = 1, irz = 1)$ as a triplet of indices (replacing the values of $irx$, $iry$ and $irz$ in the above formula leads indeed to $irg = 1$). From that given global region we can define the index intervals of the vertices $(ia, ja, ka)$ belonging to the new adapted mesh in that region :

$$\begin{aligned} ia &= ia_{beg}, \ldots, ia_{end} \ , \\ ja &= ja_{beg}, \ldots, ja_{end} \ , \\ ka &= ka_{beg}, \ldots, ka_{end} \ , \end{aligned}$$

which are provided by taking the information from the table IRGLIM, with $im = 2$. As a matter of fact, we know the lengths in the new metric of each interval (in each direction) defining a directional region. We may therefore deduce the number of adapted cells inside each of those regions and therefore deduce the cell indices (in the new numerotation) delimiting that same region. E.g. for the $x$-direction, we first redefine the table IREGN (since information for the old initial mesh are saved in table IRGLIM$(irg, 1, 1 : 2, 1)$) by the following process :

```
NXAT = 0
IXF  = 0
DO irx = 1 , NXREG
   NXA(irx) = 0
   IBEG     = IREGN(2*irx - 1)
   IEND     = IREGN(2*irx    )
   NXA(irx) = max( min(1,NX) , INT( LGX(IEND) - LGX(IBEG) + 0.5) )
   NXAT     = NXAT + NXA(irx)
   IREGN(2*irx - 1) = IXF + 1
   IXF              = NXAT
   IREGN(2*irx    ) = IXF
ENDDO
```

where LGX stands for $LG_x$ defined in formula (5) and NXA($irx$), $irx = 1, \ldots,$ NXREG stands for the total number of new vertices contained in the $x$-region of index $irx$. Similarly, we will denote by NYA($iry$), $iry = 1, \ldots,$ NYREG (resp. NZA($irz$), $irz = 1, \ldots,$ NZREG) the total number of new vertices per $y$ (resp. $z$)-region. Those information are used to define the "global" table IRGLIM($irg, idir, 1:2, 2$) for $irg = 1, \ldots,$ NREG, $idir = 1, \ldots, 3$ for the new adapted mesh. We also have the index intervals of the vertices $(i, j, k)$ belonging to the old initial mesh in the same region (provided from table IRGLIM($irg, idir, 1:2, 1$)) :

$$\begin{aligned} i &= i_{beg}, \ldots, i_{end} \ , \\ j &= j_{beg}, \ldots, j_{end} \ , \\ k &= k_{beg}, \ldots, k_{end} \ . \end{aligned}$$

For both meshes, the index starting the region $irg$ coincides with the cell index given by table IRGLIM, except for $irg = 1$ and if one of the starting indexes is equal to 1, since in that case the corresponding vertex index in the respective direction has to be set to 0. So, for a given index triplet $(ia, ja, ka)$ of a vertex related to the new adapted mesh in the global region $irg$, its node coordinates $xa(ia, ja, ka, id)$, $id = 1, 2, 3$ are defined as the barycenter coordinates of $x(ii-1)$ and $x(ii)$ (resp. $y(jj-1)$ and $y(jj)$, $z(kk-1)$ and $z(kk)$) for the $x$ ($id = 1$) (resp. $y$ ($id = 1$), $z$ ($id = 3$) ) direction :

$$\begin{aligned} xa(ia, ja, ka, 1) &= \alpha_x \, x(ii-1) + (1-\alpha_x) \, x(ii) \ , \\ xa(ia, ja, ka, 2) &= \alpha_y \, y(jj-1) + (1-\alpha_y) \, y(jj) \ , \\ xa(ia, ja, ka, 3) &= \alpha_z \, z(kk-1) + (1-\alpha_z) \, z(kk) \ , \end{aligned}$$

where $x(ii)$ (resp. $y(jj)$, $z(kk)$) stands for $x(ii, jj, kk, 1)$ (resp. $x(ii, jj, kk, 2)$, $x(ii, jj, kk, 3)$) and

$$\begin{aligned} \alpha_x &= (LG_x(ii) - l_x) \quad / \quad (LG_x(ii) - LG_x(ii-1)) \ , \\ \alpha_y &= (LG_y(jj) - l_y) \quad / \quad (LG_y(jj) - LG_y(jj-1)) \ , \\ \alpha_z &= (LG_z(kk) - l_z) \quad / \quad (LG_z(kk) - LG_z(kk-1)) \ , \end{aligned}$$

with the $x$ (resp. $y$, $z$)-length $l_x$ (resp. $l_y$, $l_z$) respresents the accumulated length in the new metric up to the $x$ (resp. $y$, $z$) position of the new node of index $(ia, ja, ka)$ :

$$\begin{array}{lllll} l_x & = LG_x(i_{beg}-1) & + (ia - ia_{beg}+1) & (LG_x(i_{end}) - LG_x(i_{beg}-1)) & / \ \mathrm{NXA}(irx) \ , \\ l_y & = LG_y(j_{beg}-1) & + (ja - ja_{beg}+1) & (LG_y(j_{end}) - LG_y(j_{beg}-1)) & / \ \mathrm{NYA}(iry) \ , \\ l_z & = LG_z(k_{beg}-1) & + (ka - ka_{beg}+1) & (LG_z(k_{end}) - LG_z(k_{beg}-1)) & / \ \mathrm{NZA}(irz) \ , \end{array}$$

(recall that we know the total number of new nodes within each region and that they are equidistributed inside each of those regions); the indexes $(ii, jj, kk)$ are found in $[i_{beg}, i_{end}] \times [j_{beg}, j_{end}] \times [k_{beg}, k_{end}]$ under the following requirements:

$$\begin{array}{lll} LG_x(ii-1) & \leq \ l_x & \leq LG_x(ii) \ , \\ LG_y(jj-1) & \leq \ l_y & \leq LG_y(jj) \ , \\ LG_z(kk-1) & \leq \ l_z & \leq LG_z(kk) \ . \end{array} \tag{6}$$

### 4.2 User-defined scaling

In the above strategy, the control of the mesh size $N_x^{new}$, $N_y^{new}$ and $N_z^{new}$ is left to the user by specifying the value of the constant $c_{met}$ in the construction of the local metric. This user dependent data is not very convenient.

A more flexible and sensible data is to let the user choose the number of cells that he wishes to use *ncm* and we then compute $\overline{c_{met}}$ in order to match this requirement by mean of the formula :

$$\overline{c_{met}} = \left(\frac{ncm}{\mathrm{NX}^a \, \mathrm{NY}^a \, \mathrm{NZ}^a}\right)^{\frac{2}{3}} \ ,$$

where $\mathrm{NX}^a$,$\mathrm{NY}^a$ and $\mathrm{NZ}^a$ are the predicted numbers of cells in the adapted mesh for the given $c_{met}$. By example, we present below (Fig. 4) the number of cells of the refined mesh as a function of *ncm* in the 2-D case of the backward facing step with the initial dimensions NX = 12 and NY = 12 (this test case will be described in the validation section).

## 5 Interpolation of the unknown scalar variables

Now that the vertex coordinates of the new adapted mesh are defined, we must interpolate all the scalar variables at each cell center of the new mesh. This is done by some finite difference approximation applied in each direction as described below: let $(ia, ja, ka)$ be any triplet of indexes in the new mesh and suppose we can define the triplet $(i, j, k)$ of indexes in the old mesh such that :

$$\begin{array}{lll} x(i-1,j,k,1) & \leq xa(ia,ja,ka,1) & \leq x(i,j,k,1) \ , \\ x(i,j-1,k,2) & \leq xa(ia,ja,ka,2) & \leq x(i,j,k,2) \ , \\ x(i,j,k-1,3) & \leq xa(ia,ja,ka,3) & \leq x(i,j,k,3) \ . \end{array}$$

Those indexes $i$, $j$ and $k$ (corresponding to indexes $ii$, $jj$, $kk$ in (6)) may be found by creating an additional table say $\mathrm{INDEX}(ia, ja, ka, idir)$ while defining the new coordinates $xa(ia, ja, ka, idir)$, $ia = 1, \ldots, \mathrm{NXA}$, $ja = 1, \ldots, \mathrm{NYA}$, $ka = 1, \ldots, \mathrm{NZA}$ and $idir = 1, \ldots, 3$.

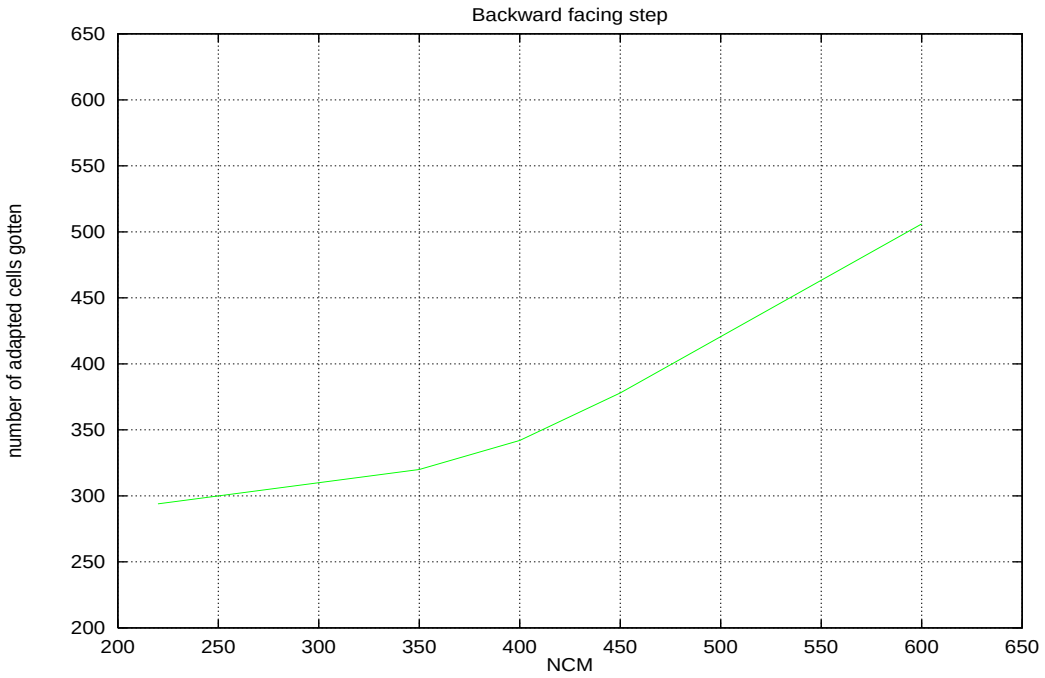


Figure 4: Number of cells in the refined mesh as a function of $ncm$.

Let us denote by $xc(i,j,k,l), l = 1,2,3$ (resp. $xca(i,j,k,l), l = 1,2,3$) the cell center coordinates of the old (resp. new adapted) mesh and set

$$\begin{array}{lll} xc1 = xc(i,j,k,1)\ , & yc1 = xc(i,j,k,2)\ , & zc1 = xc(i,j,k,3)\ , \\ xcn = xca(ia,ja,ka,1)\ , & ycn = xca(ia,ja,ka,2)\ , & zcn = xca(ia,ja,ka,3)\ , \end{array}$$

then we proceed by linear interpolation, direction by direction, using barycenter coordinate weights as it follows :

1. **$x$-direction interpolation of $u$ :**

   (a) If $xcn \leq xc1$ then

   - If $i = 1$ then set

   $$xc2 = x(i-1,j,k,1)\ ,\ \alpha = (xc1 - xcn)/(xc1 - xc2)\ ,$$

   and define

   $$\begin{array}{ll} u_m & = u(i,j,k) + (xc2 - xc1)\, D_1(u)(i,j,k)\ , \\ u_1 & = \alpha\, u_m + (1-\alpha)\, u(i,j,k)\ , \end{array}$$

- If $i > 1$ then set

$$xc2 = xc(i-1,j,k,1)\ ,\ \alpha = (xc1 - xcn)/(xc1 - xc2)\ ,$$

and define

$$u_1 = \alpha\, u(i-1,j,k) + (1-\alpha)\, u(i,j,k)\ ,$$

(b) If $xcn > 1$ then

- If $i = N_x$ then set

$$xc2 = x(i,j,k,1)\ ,\ \alpha = (xc2 - xcn)/(xc2 - xc1)\ ,$$

and define

$$\begin{aligned} u_p &= u(i,j,k) + (xc2 - xc1)\, D_1(u)(i,j,k)\ , \\ u_1 &= \alpha\, u(i,j,k) + (1-\alpha)\, f_p\ , \end{aligned}$$

- If $i < N_x$ then set

$$xc2 = xc(i+1,j,k,1)\ ,\ \alpha = (xc2 - xcn)/(xc2 - xc1)\ ,$$

and define

$$u_1 = \alpha\, u(i,j,k) + (1-\alpha)\, u(i+1,j,k)\ ,$$

2. **$y$-direction interpolation of $u$:**

(a) If $ycn \le yc1$ then

- If $j = 1$ then set

$$yc2 = x(i,j-1,k,2)\ ,\ \alpha = (yc1 - ycn)/(yc1 - yc2)\ ,$$

and define

$$\begin{aligned} u_m &= u(i,j,k) + (yc2 - yc1)\, D_2(u)(i,j,k)\ , \\ u_2 &= \alpha\, u_m + (1-\alpha)\, u(i,j,k)\ , \end{aligned}$$

- If $j > 1$ then set

$$yc2 = xc(i,j-1,k,2)\ ,\ \alpha = (yc1 - ycn)/(yc1 - yc2)\ ,$$

and define

$$u_2 = \alpha\, u(i,j-1,k) + (1-\alpha)\, u(i,j,k)\ ,$$

(b) If $ycn > 1$ then

- If $j = N_y$ then set

$$yc2 = x(i,j,k,2)\ ,\ \alpha = (yc2 - ycn)/(yc2 - yc1)\ ,$$

and define

$$\begin{aligned} u_p &= u(i,j,k) + (yc2 - yc1)\, D_2(u)(i,j,k)\ , \\ u_2 &= \alpha\, u(i,j,k) + (1-\alpha)\, f_p\ , \end{aligned}$$

- If $j < N_y$ then set

$$yc2 = xc(i, j+1, k, 2) \ , \ \alpha = (yc2 - ycn)/(yc2 - yc1) \ ,$$

and define

$$u_2 = \alpha\, u(i,j,k) + (1-\alpha)\, u(i,j+1,k) \ ,$$

3. **$z$-direction interpolation of $u$ :**

(a) If $zcn \leq zc1$ then

- If $k = 1$ then set

$$zc2 = x(i, j, k-1, 3) \ , \ \alpha = (zc1 - zcn)/(zc1 - zc2) \ ,$$

and define

$$\begin{aligned} u_m &= u(i,j,k) + (zc2 - zc1)\, D_3(u)(i,j,k) \ , \\ u_3 &= \alpha\, u_m + (1-\alpha)\, u(i,j,k) \ , \end{aligned}$$

- If $k > 1$ then set

$$zc2 = xc(i, j, k-1, 3) \ , \ \alpha = (zc1 - zcn)/(zc1 - zc2) \ ,$$

and define

$$u_3 = \alpha\, u(i,j,k-1) + (1-\alpha)\, u(i,j,k) \ ,$$

(b) If $zcn > 1$ then

- If $k = N_z$ then set

$$zc2 = x(i, j, k, 3) \ , \ \alpha = (zc2 - zcn)/(zc2 - zc1) \ ,$$

and define

$$\begin{aligned} u_p &= u(i,j,k) + (zc2 - zc1)\, D_3(u)(i,j,k) \ , \\ u_3 &= \alpha\, u(i,j,k) + (1-\alpha)\, f_p \ , \end{aligned}$$

- If $k < N_z$ then set

$$zc2 = xc(i, j, k+1, 3) \ , \ \alpha = (zc2 - zcn)/(zc2 - zc1) \ ,$$

and define

$$u_3 = \alpha\, u(i,j,k) + (1-\alpha)\, u(i,j,k+1) \ ,$$

Finally define the interpolated value at the center of cell $(ia, ja, ka)$ of the new mesh by :

$$u(ia, ja, ka) \stackrel{\text{def}}{=} (u_1 + u_2 + u_3)/3 \ .$$

# 6 Numerical test

## 6.1 User-defined parameters

The adaption process needs some user-defined parameters set in the input data file of the adaptation package :

- $c_{met}$ : constant used to initiate the prediction of the refined mesh size,
- $ncm$ : integer constant used for scaling the number of adapted cells,
- $\varepsilon$ : positive real parameter used for the weighted metric.

Moreover, the frequency of adaptation is set in a PHOENICS input file and frozen during the restart processes. In the case of steady calculations, a PHOENICS parameter is set which limit the number of sweeps that are performed before another adaptation.

## 6.2 Validation in 2D : the backward facing step

We present a numerical test that validates our adaption strategy in two dimensions. It is a steady turbulent flow over a backward facing step of height $h$ in a 2-D channel of width $3\,h$. The Reynolds number is set to 45 000 and $h = 0.0381$ m. The first computation is done on a relative coarse mesh with two regions in the $x$ and $y$ directions (NZ = 1) :

$$\mathrm{NXS1} = 6 \ , \ \mathrm{NXS2} = 6 \ , \ \mathrm{NX} = \mathrm{NXS1} + \mathrm{NXS2} \ ,$$

and

$$\mathrm{NYS1} = 6 \ , \ \mathrm{NYS2} = 6 \ , \ \mathrm{NY} = \mathrm{NYS1} + \mathrm{NYS2} \ .$$

The metric constants are set as below :

$$c_{met} = 1 \ , \ ncm = 580 \ , \ \varepsilon = 10^{-5} \ .$$

The coarse mesh is depicted in Fig. 6 and the resulting solution (pressure and velocities) are given in Fig. 7–9.

The steady state is reached after 96 sweeps and the corresponding metric is calculated. In Fig. 15 and Fig. 16, we give the ideal lengths in the $x$ and $y$ directions obtained in the coarse mesh.

The corresponding adapted mesh is depicted in Fig. 10. Its dimension is 31 cells in the $x$ direction and 15 in the $y$ direction. The interpolated solution (see the pressure Fig. 14) allows PHOENICS to restart with a satisfying solution. The new adapted solution are represented in Fig. 11, 12 and 13. The boundary layer as well as the downstream part of the flow (in the $x$-direction) are well capted by the adaptor. Besides the scarcity of the refinement of the initial coarse mesh just after the step, the recirculation zone is also caught by our adaptor.

## 6.3 Validation in 3D : the 3D oven simulation

We are now interested in the validation of the adaptor in three dimensions. The chosen test case describes the steady three dimensional turbulent flow with conjugate heat and mass transfer in an industrial oven, we cannot give values of the different quantities we refer in describing the test case below. Inside the oven, the air is heated indirectly by a gas-fired heat exchanger as shown schematically in figure 5. The hot exhaust gases flow through a multi-pass heat exchanger located in an outer compartment on the left-hand side of the oven. Heat is then transferred to the oven gas primarly by convection and surface-to-surface radiation. The flow across the heat exchanger is driven by a fan located in a duct at the top of the oven. The hot gas is driven through a porous screen so as to heat the ware within the oven compartment. The circuit is completed by air leaving the compartment via a porous screen situated above the oven and below the fan compartment. Mass transfer is not considered and the turbulence is represented by using a constant eddy viscosity based on the fan speed and the fan-duct height. Buyoncy forces are ignored and the thermal analysis does not account for radiative heat transfer. For simplicity, the fan is modelled by a fixed velocity, and the heat exchanger as a volumic heat source. The porous screens are modelled by a frictional resistance force in the momemtum equations and a area porosity factor. The flow area of each screen is also prescribed.

For air, both the molecular kinematic viscosity and the thermal conductivity are prescribed, and the density is computed from the ideal gas law. The heated ware are represented by steel with the following given properties: $\rho_s$ (in $kg/m^3$), $k_s$ (in $W/mK$) and $C_{p,s}$ (in $J/kgK$). Heat can escape from the system to ambient air via the top, bottom and left-hand external walls; the wall thickness is given together with the thermal conductivity $k_b$ (in $W/mK$). The right-hand wall is a symmetry plane and so zero-flux boundary conditions are applied. The walls separating the outer compartment from the oven are taken to be perfectly insulating.

The initial grid dimensions are 20 cells in the $x$ direction, 36 cells in the $y$ direction and 44 cells in the $z$ direction. The 34 965 points are distributed in a uniform cartesian grid. The furnace length, width and heigh are also precribed (in meters). Those dimensions used for the simulation represent half the oven length and only a single section of its width, exploiting symmetry in each direction. The calculations are performed for a given volumic heat input (in $kW/m^3$) from the heat exchanger. The corresponding level of heat input is selected so as to maintain an average oven temperature of $T_{av}$ (given in Celsius) during steady-state operation. The volumic flow rate of the fan $Q_{fan}$ (given in $m^3/s$) is specified, and the resistance force through the porous plates is specified as $A\,\delta p$, where $A$ is the free-flow area and $\delta p = -1/2\,(\kappa\,\rho)\,V^2$, where $(\kappa\,\rho)$ is given and $V$ is based on the actual velocity within the screen. The oven operates at a prescribed pressure. After 200 sweeps, the error estimates are calculated and the mesh quality is described Fig. 17. A first adaptation is performed with

$$c_{met} = 1\,ncm = 60\,000\ ,\ \varepsilon = 10^{-5}\ .$$

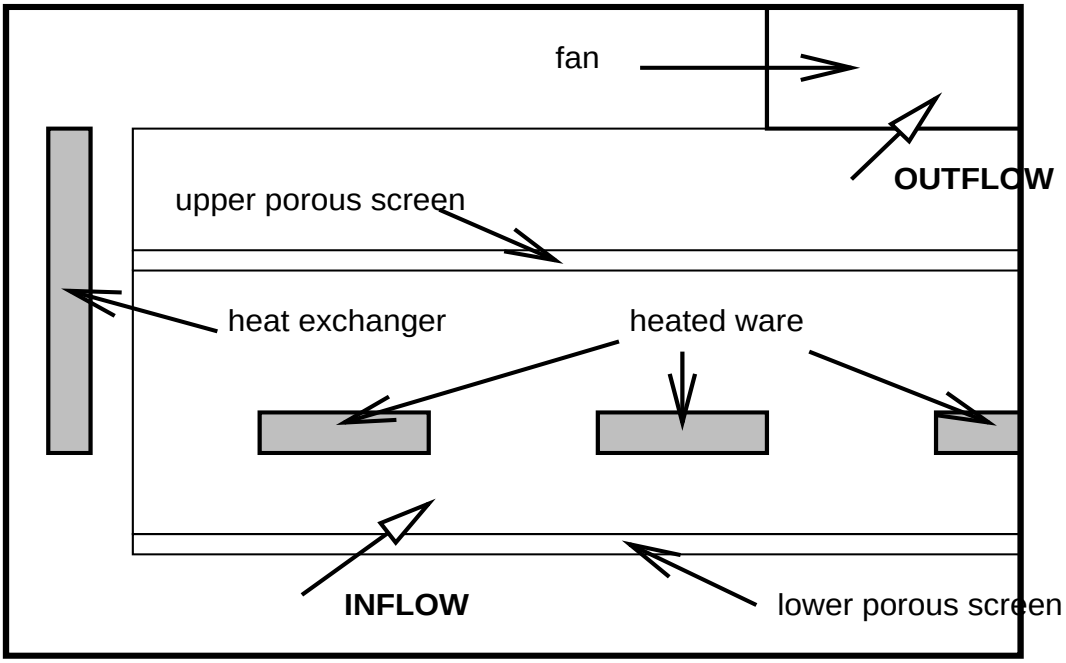


Figure 5: Industrial oven: description of the geometry in the cup plane $x = 0.5m$.

The corresponding new mesh size is 36 cells in the $x$ direction, 40 cells in the $y$ direction and 60 cells in the $z$ direction (i.e. 92 537 points). Then we make run the interface to change the "EARDAT" file accordingly to the new adapted mesh and solution. PHOENICS restarts with this new mesh and the interpolated solution. After again 200 sweeps, a second adaptation is performed with

$$c_{met} = 1\, ncm = 100\,000\ ,\ \varepsilon = 10^{-5}\ .$$

This second adaptation generates a new mesh the dimensions of which are 35 cells in the $x$ direction, 54 cells in the $y$ direction and 66 cells in the $z$ direction (i.e. 132 660 points). PHOENICS restarts again for 200 sweeps. The solution calculated on the second adapted mesh after 200 sweeps is depicted on Fig. 18 (Velocity) and Fig. 19 (temperature). We can observe that the mesh is refined where the solution is the steepest corresponding to a more important error. Moreover, the solution is not polluted by the interpolation on the adapted mesh.

# 7 Conclusion

The validation of the adaptation process is achieved :

- The metric calculation provides a good error estimate of the CFD solution;
- The mesh refinement matches the error estimate behaviour;
- The nodes equidistribution is efficient to generate an adapted cartesian grid;
- The region definition is respected.

The PHOENICS interface is efficient and allows a good restart of the "earth" program. We did not observe any pollution of the solution when the computation restarts with the interpolated values on the new mesh. The next step would be to create a multiblock adaption tool that would avoid the generation of unwanted refined regions. As a matter of fact, the present strategy is global and extend refinement in every direction even if the error has been detected high in a very small region. The way the multiblock adaption would be another alternative is to have a good sensor that would determine those particular regions in which the error is high. To achieve that goal, particular attention has to be paid in defining error troncature and then to find a good algorithm that would colour then define patches in which refinement has to be perform. With a scale strategy defining the error truncature, we may define multi-level patches, i.e. define embedded blocks (which have to be cartesian) with the additional constraint that there should be a fixed cell ratio between two embedded blocks (i.e. refinement has to be uniform from one level to the next inside a same block). Comparing to the present global adaption strategy, only the error estimate has to be kept since we will not use equidistribution anymore. That work is in progress and will be subject to another report.

## References

[1] H. Borouchaki, M. Castro, F. Hecht, P.L. George, and B. Mohammadi. Anisotropic adaptive mesh generation in two dimensions cfd. Technical report, Computational Fluid Dynamics 96, J.A. Désidéri, C. Hirsch, P. Le Tallec, M. Pandolfi, J. Périaux (réd.), J. Wiley & Sons, p. 181–186, September 1996.

[2] M.J. Castro-Díaz and F. Hecht. Error interpolation minimization and anisotropic mesh generation. Technical report, Proceedings of the Conference on Finite Elements in Fluids: New trends and applications, october 1995, Venice (Italy).

[3] E. F. d'Azevedo and R. B. Simpson. On optimal interpolation triangle incidences. *SIAM'S Journal on Scientific and Statistical Computing*, (6):1063–1075, 1989.

[4] E. F. d'Azevedo and R. B. Simpson. On optimal triangular meshes for minimizing the gradient error. *Numerische Mathematik*, 59(4):321–348, 1991.

[5] M.-G. Vallet. Génération de maillages éléments finis anisotropes et adaptatifs. Master's thesis, Université de Paris VI, 30 septembre 1992.

# FIGURES

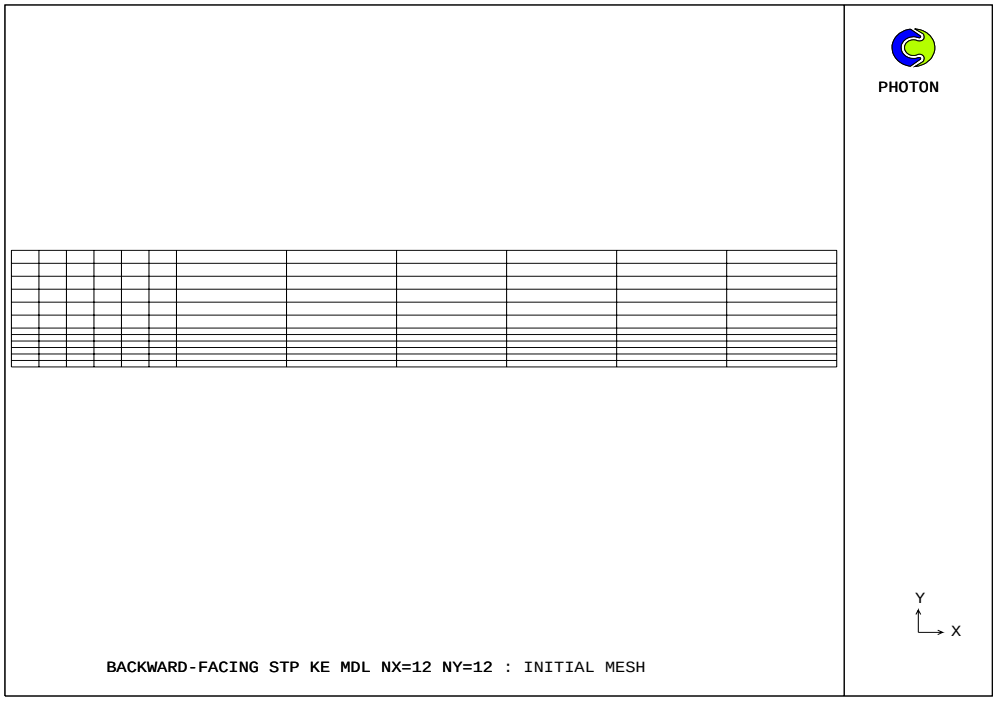


Figure 6: Coarse mesh NX = 12 , NY = 12

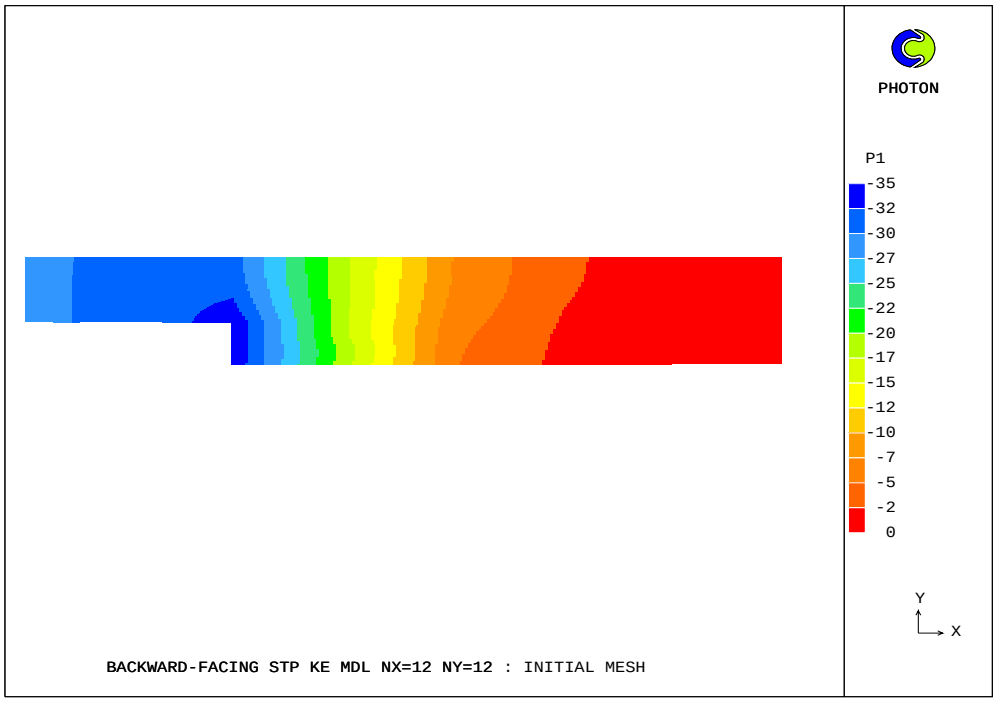


Figure 7: Pressure contours NX = 12 , NY = 12

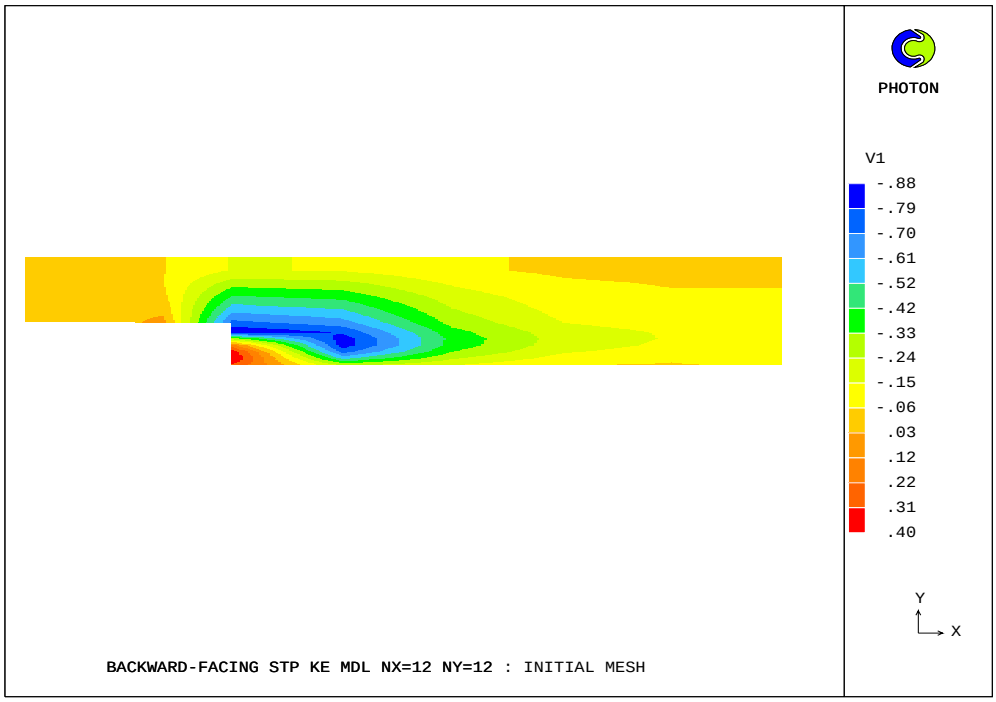


Figure 8: Velocity in $y$-direction NX = 12 , NY = 12

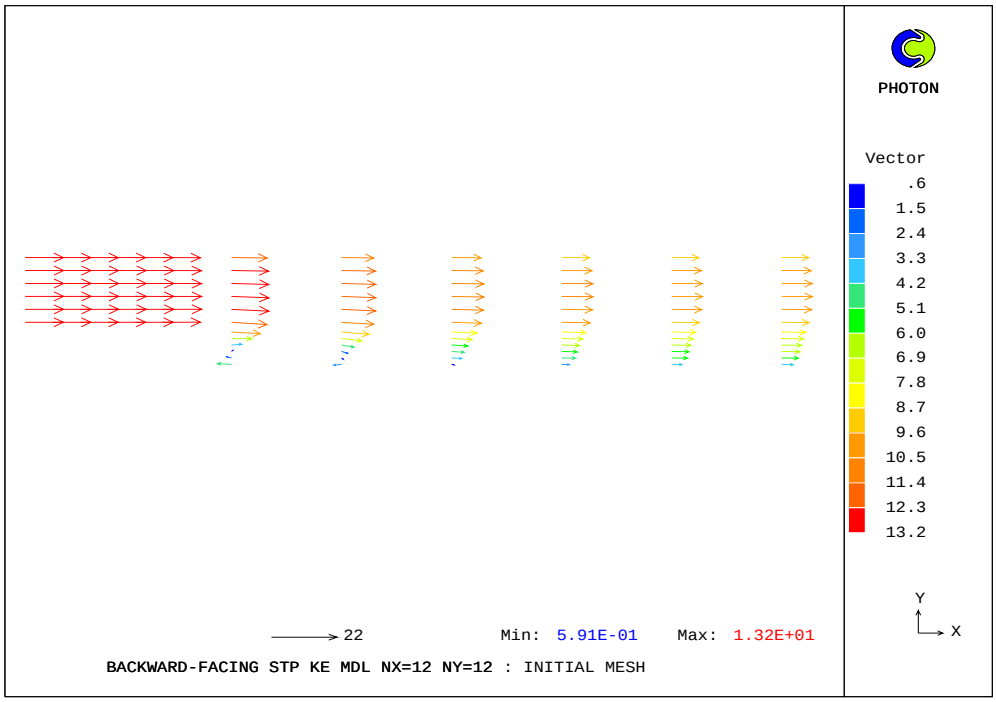


Figure 9: Velocity vectors NX = 12 , NY = 12

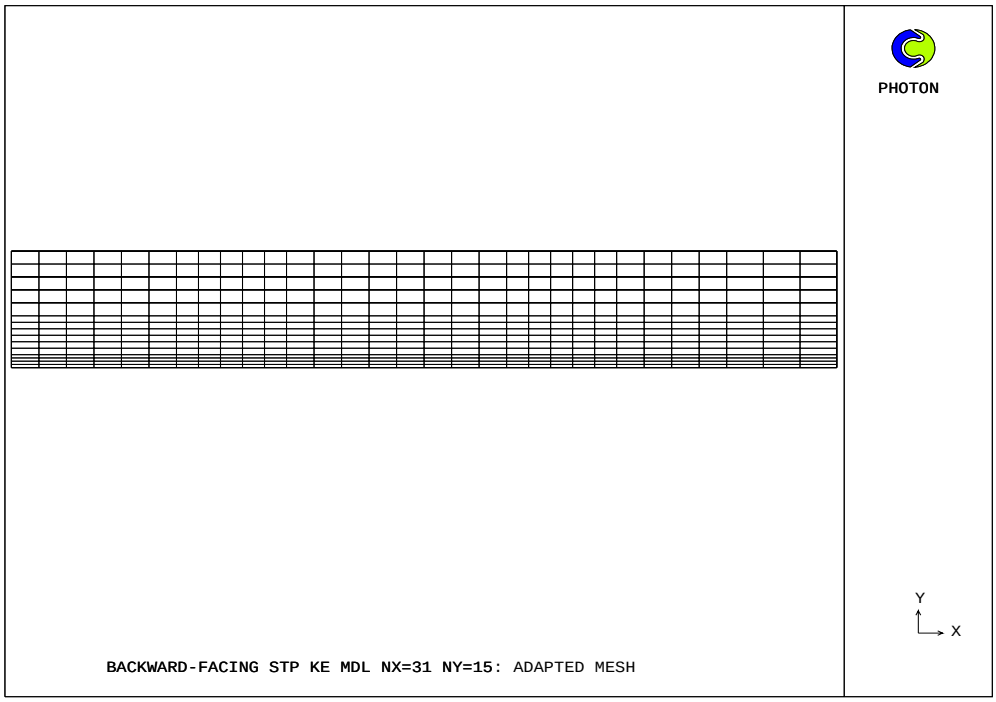


Figure 10: Adapted mesh NX = 31 , NY = 15

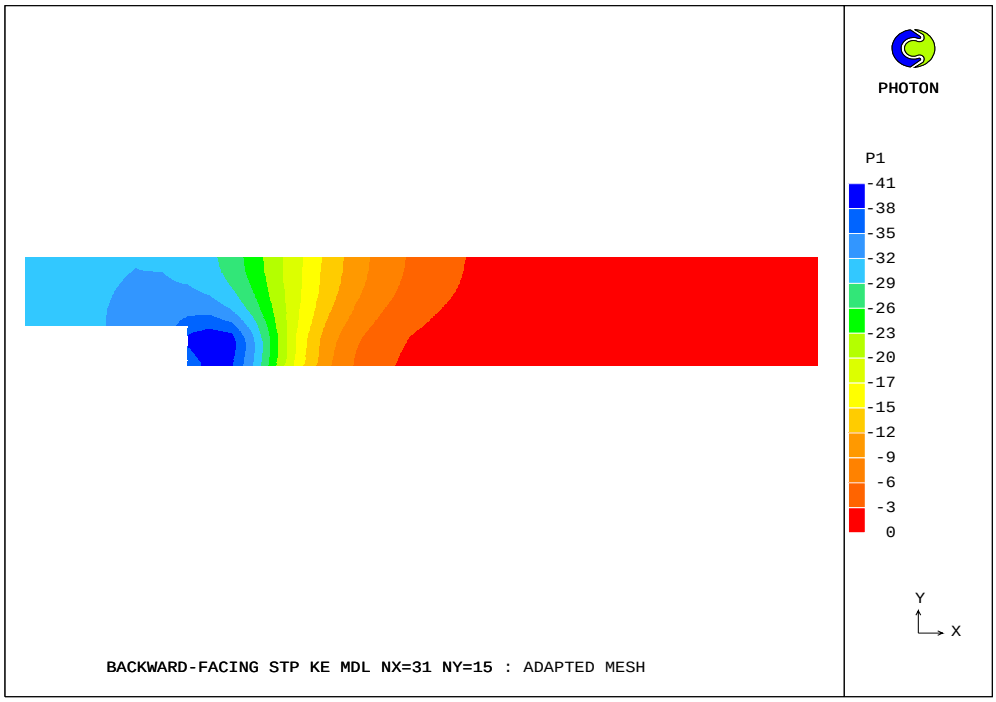


Figure 11: Pressure contours after adaptation NX = 31 , NY = 15

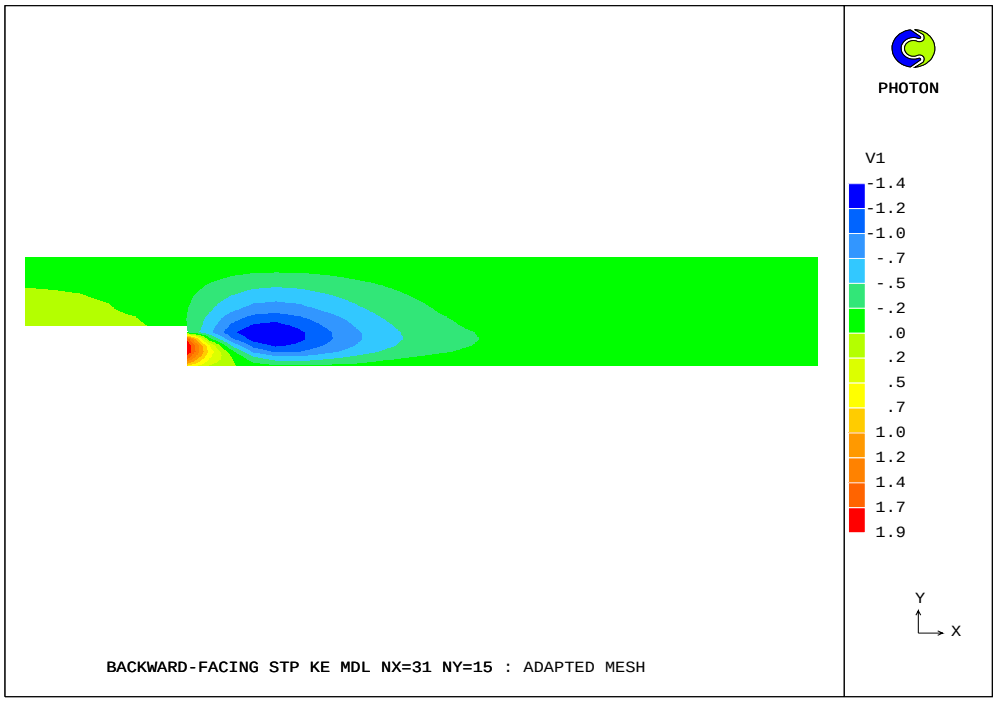


Figure 12: Velocity in $y$-direction after adaptation NX = 31 , NY = 15

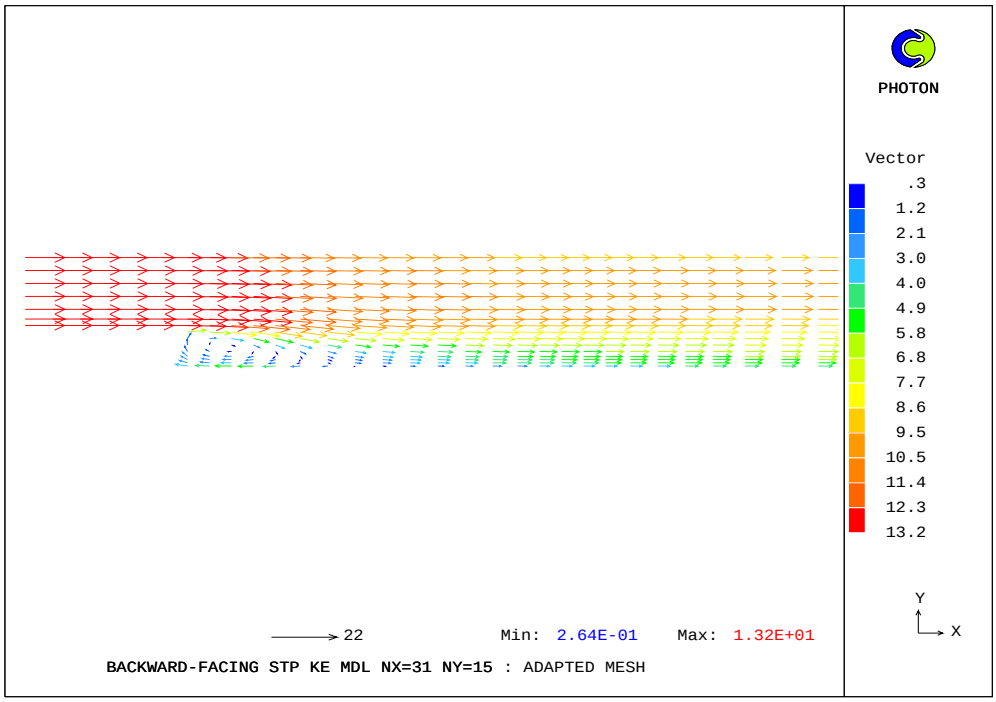


Figure 13: Velocity vectors after adaptation NX = 31 , NY = 15

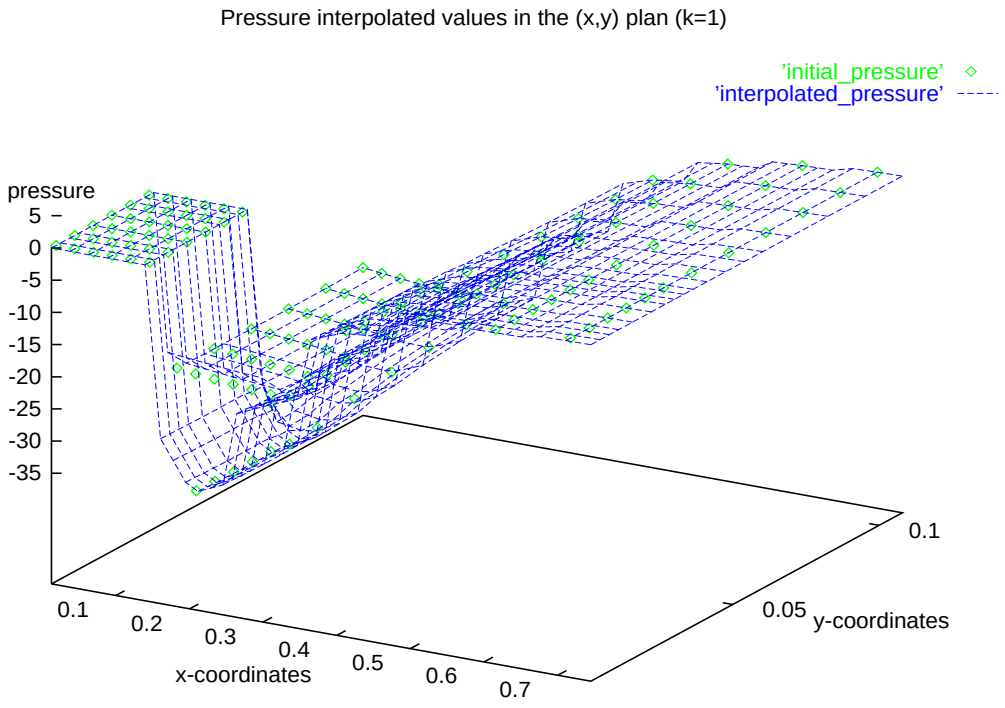


Figure 14: Interpolation of the pressure to restart PHOENICS

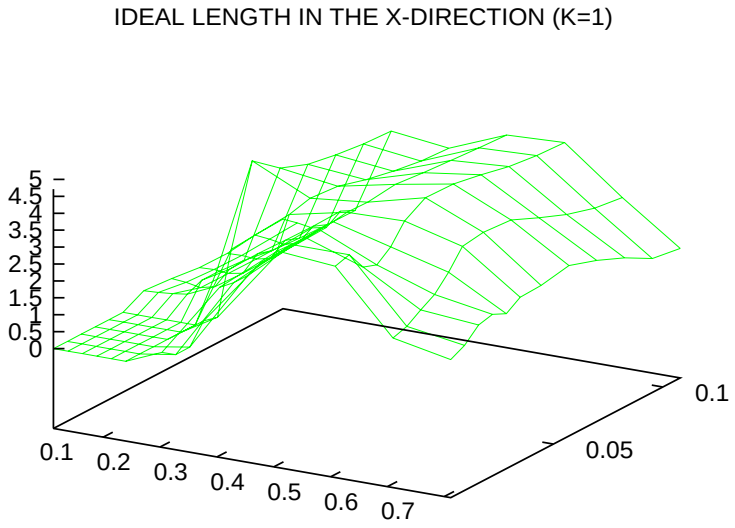


Figure 15: Ideal lengths in the $x$-direction on the coarse mesh NX = 12 , NY = 12

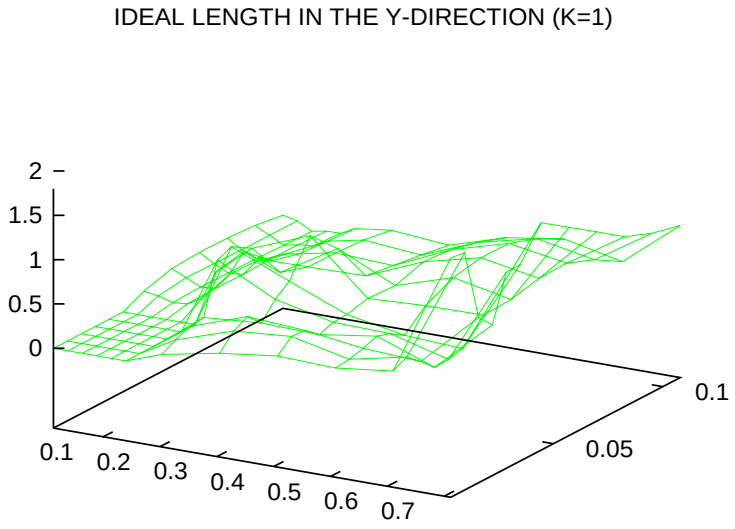


Figure 16: Ideal lengths in the $y$-direction on the coarse mesh NX = 12 , NY = 12

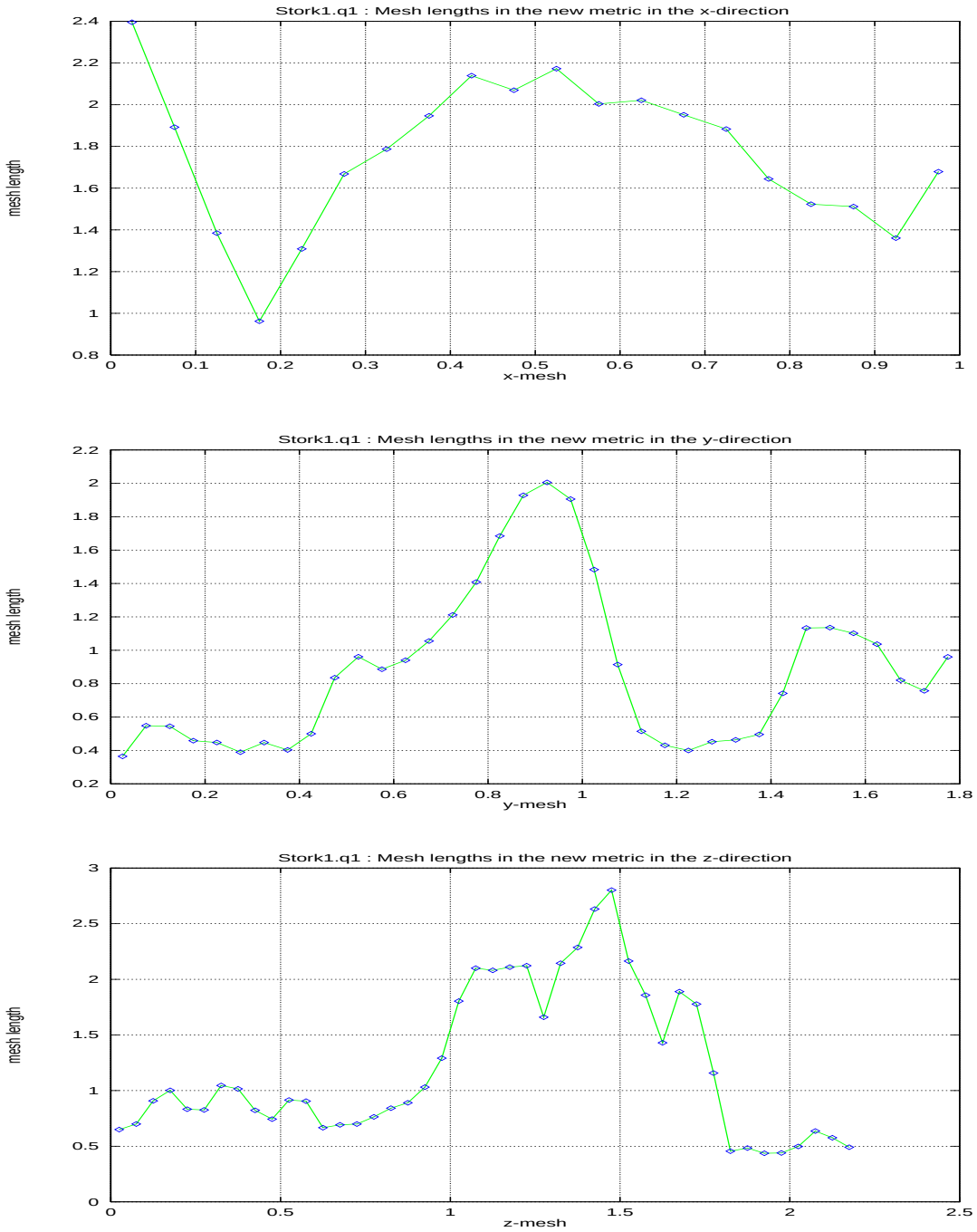


Figure 17: 3D Oven simulation : Ideal lengths in the $x$, $y$, $z$ directions on the coarse mesh

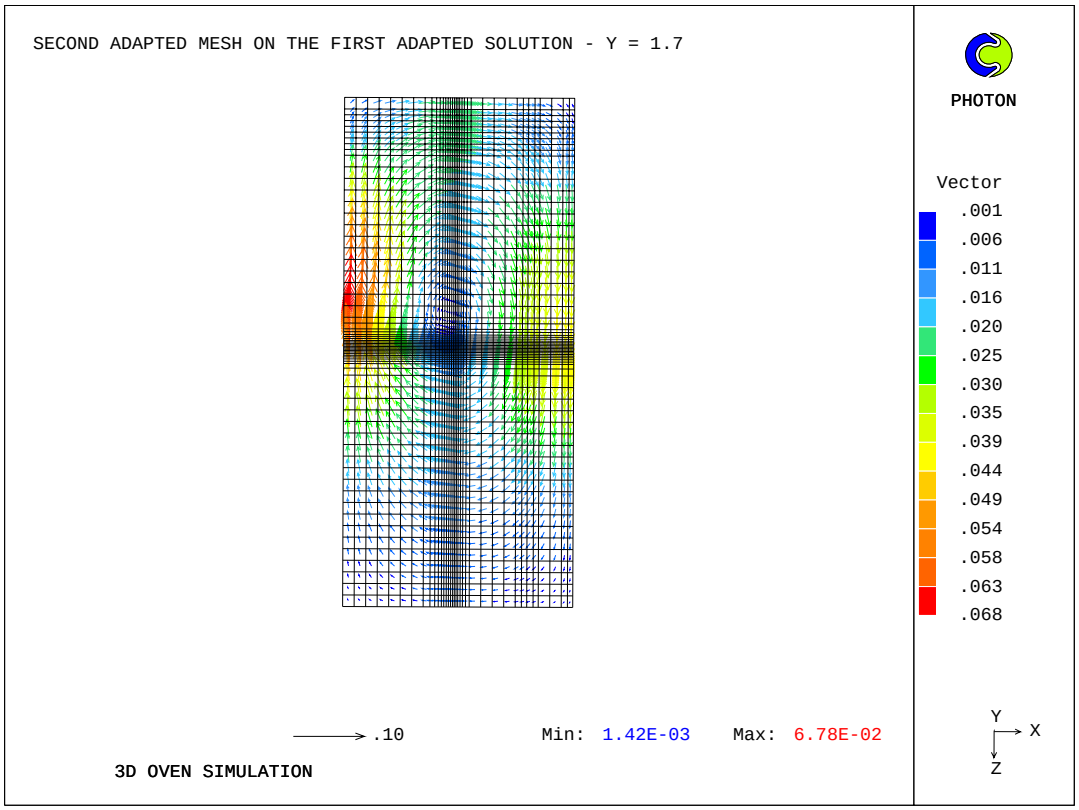


Figure 18: 3D Oven simulation : Second adaptation - Velocity and adapted mesh in the plane NY = 53

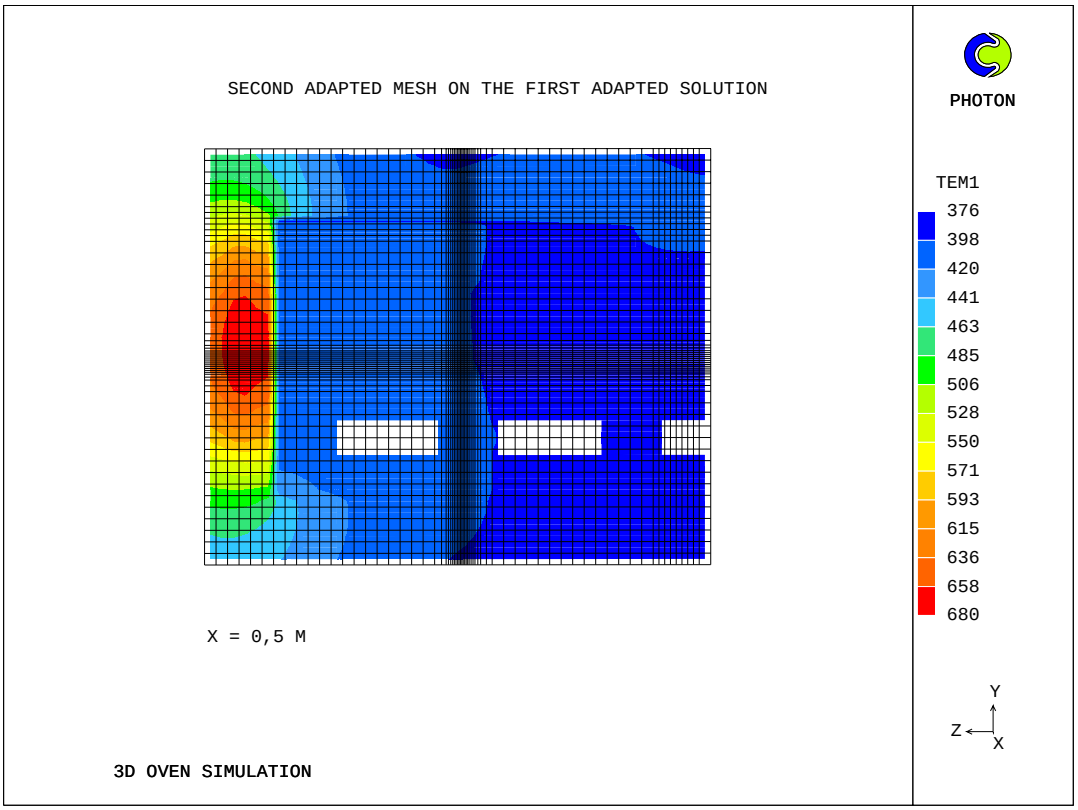


Figure 19: 3D Oven simulation : Second adaptation - Temperature and adapted mesh at $x$=0.5 m

Unit´e de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unit´e de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unit´e de recherche INRIA Rhône-Alpes, 655, avenue de l'Europe, 38330 MONTBONNOT ST MARTIN
Unit´e de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unit´e de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
http://www.inria.fr
ISSN 0249-6399