Multiresolution Representation in Unstructured Meshes: I. Preliminary Report

Remi Abgrall, Ami Harten

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Multiresolution Representation in Unstructured Meshes: I. Preliminary Report

Rémi Abgrall et Ami Harten

N° 2342

Septembre 1994

PROGRAMME 6

Calcul scientifique,

modélisation

et logiciel numérique

ISSN 0249-6399

INRIA
SOPHIA ANTIPOLIS

Multiresolution Representation in Unstructured Meshes: I. Preliminary Report

Rémi Abgrall * et Ami Harten **

Programme 6 — Calcul scientifique, modélisation et logiciel numérique
Projet Sinus

Rapport de recherche n ° 2342 — Septembre 1994 — xlvi pages

Abstract: In this paper we describe techniques to represent data which originate from discretization of functions in unstructured meshes in terms of their *local* scale components. To do so we consider a nested sequence of discretization which corresponds to increasing level of resolution, and define the scales as the "difference in information" between any two successive levels. We obtain data compression by eliminating scale-coefficients which are sufficiently small. This capability for data compression can be used to reduce the cost of numerical schemes by solving for the more compact representation of the numerical solution in terms of its significant scale-coefficients.

Key-words: Multiscale analysis, unstructured meshes

(Résumé : tsvp)

*INRIA Sophia-Antipolis, Department of Mathematics (UCLA), abgrall@sophia.inria.fr
**School of Mathematical Sciences, (Tel-Aviv University) et Department of Mathematics (UCLA)

Analyse multiéchelle sur des maillages non structurés : I rapport préliminaire

Résumé : Dans ce rapport, nous décrivons diverses techniques permettant de représenter, au moyen de coefficients d'échelle *locaux*, des données provenant de la discrétisation de fonctions sur des maillages non structurés. Pour ce faire, nous considérons une suite emboitée de discrétisations correspondant à des discrétisations de plus en plus fines. Puis nous définissons les échelles comme la "différence d'information" entre deux niveaux successifs. La compression des données est obtenue en éliminant tous les coefficients suffisament petits. Cette technique peut être utilisée afin de réduire le coût de schémas numériques en Mécanique des Fluides Numérique .

Mots-clé : Analyse multiéchelle, maillages non structurés

1 Introduction

Fourier analysis, which provides a way to represent square-integrable functions in terms of their sinusoidal scale-components, has contributed greatly to all fields of science. The main drawback of Fourier analysis is in its globality – a single irregularity in the function dominates the behavior of the scale-coefficients and prevents us from getting immediate information about the behavior of the function elsewhere.

The recent development of the theory of wavelets (see [21] and [20]) was a great step towards local scale decomposition, and has already had great impact on several fields of science. In numerical analysis representation by compactly supported wavelets (see [8] and [7]) is used to reduce the cost of many numerical solution algorithms (see [4]). The main drawback of the theory of wavelets is that it decomposes any square integrable function into scale-components which are translates and dilates of a *single* function. Consequently there are conceptual difficulties in extending wavelets to bounded domains and general geometries.

In [10, 11] we introduced the concept of "nested discretization" which enables us to represent data that originates from unstructured grids in bounded domains in terms of its scale decomposition. This framework is a generalization of the theory of wavelets in the sense that under conditions of uniformity its natural result is wavelets.

The main application of this new capability is to the numerical solution of partial differential equations in complex geometries, e.g. the solution of the equations of compressible gas around an aeroplane. As we have demonstrated in a series of articles on multiresolution schemes for the solution of hyperbolic conservation laws in Cartesian grids (see [13, 14] and [5, 6]), there is a lot to be gained by formulating the time-evolution of the problem in terms of the more compact representation of the solution by its significant scale-coefficients. This technique is an attractive alternative to the methodology of adaptive grids and it enables us to dynamically adjust the local level of resolution to the variation of the solution.

Unstructured meshes have been used primarily for two purposes: (1) To have a faithful description of the boundary in order to accurately impose the boundary conditions which determine the solution. (2) To serve as an implementation of adaptive grid ideas. Unfortunately the computation of numerical solutions on unstructured grids is considerably more expensive than that on Cartesian grids. Since we can accomplish the adaptivity part by using multiresolution schemes on uniform grids, what remains is the use of unstructured meshes to describe the geometry of the boundary. In the future we plan to use a relatively thin layer of unstuctured mesh around the aeroplane, and to switch to a more regular grid further away. In a forthcoming paper [3] we shall present preliminary results on multiresolution schemes for hyperbolic conservation laws on unstructured grids.

In the present paper we describe multiresolution representation (MR) schemes for data which is obtained by discretization of functions in unstructured meshes, by either taking point values at the "nodes" or by taking averages over the "cells" of the mesh. We pay special attention to the description of the boundary of the domain in the process of coarsening/refinement which is associated with generating the various levels of resolution.

2 General framework for multiresolution representation

In this section we describe the abstract general framework for multiresolution representation of data. We consider discrete data which is associated with a nested sequence of discretization $\{\mathcal{D}_k\}_{k=0}^{L}$ and show how to design schemes for its multiresolution representation. Later we shall apply this general framework to data which corresponds to discretization of functions in unstructured grids.

Definition. We say that a sequence of linear operators $\{D_k\}_{k=0}^{\infty}$ is a nested sequence of discretization if

(i)

$$\mathcal{D}_k : \mathcal{F} \stackrel{onto}{\longrightarrow} V^k, \quad \dim V^k = J_k, \tag{2.1a}$$

(ii)

$$\mathcal{D}_k f = 0 \Rightarrow \mathcal{D}_{k-1} f = 0 \tag{2.1b}$$

Here $\mathcal{F}$ is a space of mappings and V^k is a linear space of dimension J_k.

In the following we show how to obtain multiresolution representation of any discrete data $v^L = \mathcal{D}_L f$, where the scale-decomposition corresponds to the levels of resolution which are introduced in (2.1). This is a very general framework which allows for discretizations corresponding to unstructured grids in several space dimensions.

First we show that a nested sequence of discretization comes equipped with a decimation operator D_k^{k-1} which is a linear mapping from $V^k = \mathcal{D}_k(\mathcal{F})$ onto $V^{k-1} = \mathcal{D}_{k-1}(\mathcal{F})$

$$D_k^{k-1} : V^k \stackrel{onto}{\longrightarrow} V^{k-1}. \tag{2.2a}$$

This decimation operator is defined as follows: For any v in V^k there is at least one $f \in \mathcal{F}$ such that $\mathcal{D}_k f = v$; the decimation of v is $\mathcal{D}_{k-1} f \in V^{k-1}$, i.e.,

$$v \in V^k, \; v = \mathcal{D}_k f, \quad D_k^{k-1} v = \mathcal{D}_{k-1} f. \tag{2.2b}$$

It follows from (2.1b) that D_k^{k-1} is well-defined by (2.2b), i.e. its definition is independent of the particular f. To see that let us take f_1 and f_2 in $\mathcal{F}$ such that

$$\mathcal{D}_k f_1 = v = \mathcal{D}_k f_2,$$

then by (2.1b)

$$0 = \mathcal{D}_k f_1 - \mathcal{D}_k f_2 = \mathcal{D}_k(f_1 - f_2) \Rightarrow 0 = \mathcal{D}_{k-1}(f_1 - f_2) = \mathcal{D}_{k-1} f_1 - \mathcal{D}_{k-1} f_2$$

which proves our claim.

Given $v^L \in V^L$ we can evaluate $\{v^k\}_{k=0}^{L-1}$ by repeated decimation

$$v^{k-1} = D_k^{k-1} v^k, \quad k = L, \ldots, 1. \tag{2.3}$$

Since (2.2b) implies that

$$D_k^{k-1}(\mathcal{D}_k f) = \mathcal{D}_{k-1} f \quad \text{for any } f \in \mathcal{F} \tag{2.4}$$

we get for any $f \in \mathcal{F}$ for which $v^L = \mathcal{D}_L f$, that $v^k = \mathcal{D}_k f$ for all k in (2.3). We would like to stress the point that this decimation is done without explicit knowledge of f.

Since by (2.1a) $V^k = \mathcal{D}_k(\mathcal{F})$, it follows that $\mathcal{D}_k$ has a right-inverse (at least one) which we denote by $\mathcal{R}_k$:

$$\mathcal{R}_k : V^k \to \mathcal{F}, \quad \mathcal{D}_k \mathcal{R}_k = I_k, \tag{2.5}$$

where I_k denotes the identity operator in V^k. Since $(\mathcal{R}_k v^k) \in \mathcal{F}$ is an approximation to any $f \in \mathcal{F}$ for which $\mathcal{D}_k f = v^k$, we refer to $\mathcal{R}_k$ as a reconstruction of $\mathcal{D}_k$.

Next we show that any sequence of corresponding reconstruction operators $\{\mathcal{R}_k\}_{k=0}^{L}$ defines a MR scheme for discrete data v^L in V^L. Starting from v^{k-1} in (2.3) we can get an approximation to v^k by

$$v^k \approx \mathcal{D}_k(\mathcal{R}_{k-1} v^{k-1}). \tag{2.6a}$$

We denote

$$P_{k-1}^k =: \mathcal{D}_k \mathcal{R}_{k-1}, \quad P_{k-1}^k : V^{k-1} \to V^k \tag{2.6b}$$

and refer to it as prediction operator. It follows immediately from taking $f = \mathcal{R}_{k-1} v^{k-1}$ in (2.4) and using (2.5) that P_{k-1}^k is a right-inverse of the decimation D_k^{k-1}

$$D_k^{k-1} P_{k-1}^k = I_{k-1}. \tag{2.6c}$$

We observe that the prediction error e^k

$$e^k = v^k - P_{k-1}^k v^{k-1} = (I_k - P_{k-1}^k D_k^{k-1}) v^k \tag{2.7a}$$

satisfies the relation

$$D_k^{k-1} e^k = D_k^{k-1} v^k - (D_k^{k-1} P_{k-1}^k) v^{k-1} = v^{k-1} - v^{k-1} = 0$$

and therefore it is in the null space of the decimation operator

$$e^k \in \mathcal{N}(D_k^{k-1}) = \{v | \quad v \in V^k, \ D_k^{k-1} v = 0\}. \tag{2.7b}$$

It follows from (2.2a) that

$$\dim \mathcal{N}(D_k^{k-1}) = J_k - J_{k-1} \tag{2.8a}$$

and therefore the prediction error e^k, which is described in terms of J_k components in V^k, can be represented by $(J_k - J_{k-1})$ scale-coefficients $d^k = \{d_j^k\}_{j=1}^{J_k - J_{k-1}}$. Specifically, let $\{\mu_j^k\}_{j=1}^{J_k - J_{k-1}}$ be any basis of $\mathcal{N}(D_k^{k-1})$,

$$\mathcal{N}(D_k^{k-1}) = \text{span}\{\mu_j^k\}_{j=1}^{J_k - J_{k-1}}, \tag{2.8b}$$

and let d^k denote the coordinates of e^k in this basis

$$e^k = \sum_{j=1}^{J_k - J_{k-1}} d_j^k \mu_j^k =: E_k d^k, \quad d^k =: G_k e^k. \tag{2.9a}$$

Here G_k denotes the operator which assigns to $e^k \in \mathcal{N}(D_k^{k-1})$ its coordinates d^k in the basis $\{\mu_j^k\}_{j=1}^{J_k - J_{k-1}}$; observe that $E_k G_k$ is the identity operator in $\mathcal{N}(D_k^{k-1})$, i.e.

$$E_k G_k e^k = e^k \quad \text{for any } e^k \in \mathcal{N}(D_k^{k-1}). \tag{2.9b}$$

Next we show that there is a one-to-one correspondence between v^k and $\{d^k, v^{k-1}\}$: Given v^k we evaluate

$$\begin{cases} v^{k-1} &= D_k^{k-1} v^k \\ d^k &= G_k (I_k - P_{k-1}^k D_k^{k-1}) v^k \quad ; \end{cases}$$

given v^{k-1} and d^k we recover v^k by

$$\begin{aligned} P_{k-1}^k v^{k-1} + E_k d^k &= P_{k-1}^k D_k^{k-1} v^k + E_k G_k (I_k - P_{k-1}^k D_k^{k-1}) v^k \\ &= P_{k-1}^k D_k^{k-1} v^k + (I_k - P_{k-1}^k D_k^{k-1}) v^k \\ &= v^k. \end{aligned}$$

Applying the above for $k = L, \ldots, 1$ we get that

$$v^L \overset{1:1}{\longleftrightarrow} \{d^L, \ldots, d^1, v^0\} =: \hat{v}_M; \tag{2.10}$$

we refer to $\hat{v}_M$ as the multiresolution representation (MR) of v^L.

The direct MR transform $\hat{v}_M = M \cdot v^L$ is given by the algorithm

$$\begin{cases} DO\ k = L, \ldots, 1 \\ v^{k-1} = D_k^{k-1} v^k \\ d^k = G_k (I_k - P_{k-1}^k D_k^{k-1}) v^k =: G_k^D v^k \end{cases} \tag{2.11}$$

The inverse MR transform $v^L = M^{-1} \cdot \hat{v}_M$ is given by

$$\begin{cases} DO\ k = 1, \ldots, L \\ \\ v^k = P_{k-1}^k v^{k-1} + E_k d^k. \end{cases} \tag{2.12}$$

We remark that in multigrid terminology D_k^{k-1} is "restriction" and P_{k-1}^k is "prolongation." In signal processing D_k^{k-1} plays the role of "low-pass filter" while G_k^D, which is defined in (2.11), plays the role of "high-pass filter."

In order to apply this multiresolution representation to real-life problems for purposes of analysis and data compression, we have to make sure that the direct MR transform and its inverse are stable with respect to perturbations. In [11] we present stability analysis for MR schemes and derive a sufficient condition which seems to be "close" to necessary; this condition also implies existence of a multiresolution basis for functions in $\mathcal{F}$. In appendix A we review some elements of this analysis and relate them to the particular examples of the present paper.

In the following we describe techniques to generate a nested sequence of discretization which corresponds to unstructured meshes in R^m, and present specific algorithms for R^2. The main application of this methodology is to the numerical solution of PDEs in complex geometries. Our basic approach to this application is that "the user" should provide a mesh and an appropriate numerical method, and we assume that he is satisfied with the quality of these numerical results; furthermore, we assume that the solution is overresolved in large parts of the computational domain (this may be due to propagation in time of existing regions of large variation, or due to the natural laziness of a "user"...). Our task is to provide a MR scheme which will enable us to calculate these same results, within a user-supplied tolerance for error but in a much faster way, by performing the computation in the suitable local level of resolution. To do so we apply a coarsening procedure to the given mesh to generate a *nested* sequence of discretization, and find an appropriate sequence of reconstruction operators. Once this is accomplished we use the machinery of this section to obtain multiresolution representation of data in the user-supplied mesh. We remark that the preliminary results of applying this program to hyperbolic conservation laws are encouraging (see [13, 14], [5, 6],[3]).

A more ambitious program is to endow the MR scheme with the capability to increse the level of resolution above that of the user-supplied mesh if the analysis of the scale coefficients indicate the need to do so due to development of large variation on a smaller scale (of course the user has to supply a limit on the smallest scale that he is willing to pay for). At present we are not doing that, and therefore the main interest in this paper is in developing coarsening procedures that result in a sequence of nested discretization; however we shall also consider the question of refinement. We would like to point out again that the main use of unstructured

grids is for the geometry of the problem, and thus in both coarsening and refinement one has to pay special attention to the boundaries, and make sure that they are appropriately resolved.

In this preliminary report we present some numerical experiments of data compression in unstructured meshes, and compare them to similar experiments with MR schemes for uniform tensor-product grids (which are known to be stable). These experiments indicate that, inspite of the strong nonuniformity in our unstructured meshes, the compression ratio and the compression error are of the same order as those of the uniform tensor-product grids.

3 MR schemes for pointvalue discretization

Consider bounded functions $f \in \mathcal{F}$

$$f : \Omega \subset \mathrm{R}^m \longrightarrow \mathrm{R}, \qquad \mathcal{F} = \mathcal{B}(\Omega), \tag{3.1}$$

where Ω is a bounded domain, take any sequence

$$X^k = \{x_i^k\}_{i=0}^{J_k}, \ x_i^k \in \Omega, \tag{3.2a}$$

and define $v^k = \mathcal{D}_k f$ by

$$v_i^k = (\mathcal{D}_k f)_i = f(x_i^k), \quad v^k = \{v_i^k\}_{i=0}^{J_k}. \tag{3.2b}$$

We refer to (3.2) as discretization by pointvalue. Note that here the index starts from $i = 0$ and thus there are $J_k + 1$ elements in X^k.

The sequence of discretization $\{\mathcal{D}_k\}$ in (3.2) is nested if, and only if, for all k

$$X^{k-1} \subset X^k; \tag{3.3}$$

decimation in this case amounts to removing from v^k components $v_i^k = f(x_i^k)$ for $x_i^k \notin X^{k-1}$. Note that the decimation operator D_k^{k-1} is defined directly from the sequence $\{\mathcal{D}_k\}$.

Let $\mathcal{I}_k(x; v^k)$ denote *any* interpolation of $\{v_i^k\}$ at the corresponding nodes $\{x_i^k\}$, i.e.

$$\mathcal{I}_k(x_i^k; v^k) = v_i^k \qquad \text{for all} \ \ x_i^k \in X^k, \tag{3.4a}$$

and observe that

$$\mathcal{D}_k \mathcal{I}_k(\cdot; v^k) = v^k. \tag{3.4b}$$

The above relation shows that reconstruction in this case amounts to a selection of an interpolation technique in (3.4a). Given v^{k-1} we approximate v^k by (2.6a), and get the prediction error e^k (2.7a)

$$e_i^k = v_i^k - \mathcal{I}_{k-1}(x_i^k; v^{k-1}) \qquad \text{for all} \ \ x_i^k \in X^k; \tag{3.5}$$

using multigrid terminology this prediction can be expressed by saying that we use injection of the values corresponding to x_i^k which are in X^{k-1}, and interpolation for those which are not in X^{k-1}. Observe that the prediction error $e_i^k = 0$ for all i such that $x_i^k \in X^{k-1}$. We define the scale-coefficients $d^k = \{d_j^k\}$ as the prediction error in $X^k - X^{k-1}$, i.e.

$$d_j^k = e_{i_j}^k \ \ \text{for all} \ \ x_{i_j}^k \notin X^{k-1}, \tag{3.6a}$$

where i_j for $j = 1, \ldots, (J_k - J_{k-1})$ is some ordering of the points in $X^k - X^{k-1}$

$$\{x^k_{i_j}\}_{j=1}^{J_k - J_{k-1}} = X^k - X^{k-1}. \tag{3.6b}$$

Example 3.1. Nonuniform partition of [0,1]

Let us consider continuous functions f in the interval [0,1]

$$f \in \mathcal{F} = C^0[0,1],$$

and let X^L be an arbitrary partition of $[0,1]$

$$X^L = \{x^L_i\}_{i=0}^{J_L},\ x^L_0 = 0,\ x^L_{J_L} = 1,\ J_L = 2^L J_0 \tag{3.7a}$$

where the sequence above is strictly increasing and J_0 is some integer. We define the grids $X^k = \{x^k_i\}_{i=0}^{J_k},\ k = L-1, \ldots, 1$ by the coarsening

$$x^{k-1}_i = x^k_{2i}, \quad i = 0, \ldots, J_{k-1} =: J_k/2, \tag{3.7b}$$

in which we delete from X^k all the points with odd indices (The only reason that we remove *every* other point is to simplify the notations). Let us consider now the discrete values in (3.2b); clearly v^{k-1} is obtained from v^k by the decimation

$$v^{k-1}_i =: (D^{k-1}_k v^k)_i = v^k_{2i}, \quad i = 0, \ldots, J_{k-1}. \tag{3.7c}$$

In this case the direct multiresolution representation (MR) transform $v^L \mapsto \hat{v}_M$ in (2.11) can be expressed by

$$\begin{cases} DO\ k = L, \ldots, 1 \\ \\ v^{k-1}_i = v^k_{2i}, \quad i = 0, \ldots, J_{k-1} \\ \\ d^k_j = v^k_{2j-1} - \mathcal{I}_{k-1}(x^k_{2j-1}; v^{k-1}), \quad j = 1, \ldots, J_{k-1}. \end{cases} \tag{3.8}$$

Similarly the inverse MR transform $\hat{v}_M \mapsto v^L$ can be expressed by

$$\begin{cases} DO\ k = 1, \ldots, L \\ \\ v^k_{2i} = v^{k-1}_i, \quad i = 0, \ldots, J_{k-1} \\ \\ v^k_{2j-1} = \mathcal{I}_{k-1}(x^k_{2j-1}; v^{k-1}) + d^k_j, \quad j = 1, \ldots, J_{k-1}, \end{cases} \tag{3.9}$$

We refer to d^k as the scale-coefficients of the k-th level of resolution. For the piecewise-linear interpolation

$$\mathcal{I}_k(x; v^k) = v^k_{i-1} + \frac{x - x^k_{i-1}}{x^k_i - x^k_{i-1}}(v^k_i - v^k_{i-1}), \quad \text{for} \quad x^k_{i-1} \le x \le x^k_i \tag{3.10}$$

we get for any f for which $v^k = \mathcal{D}_k f$ that

$$d_j^k(f) = -(h_{2j-1}^k)(h_{2j}^k) \cdot f[x_{2j-2}^k, x_{2j-1}^k, x_{2j}^k], \tag{3.11a}$$

where $f[x_1, x_2, x_3]$ is the second divided difference of f and $h_i^k =: x_i^k - x_{i-1}^k$. Hence if $f(x)$ is twice differentiable in $[x_{j-1}^{k-1}, x_j^{k-1}]$ then

$$d_j^k(f) = -\frac{1}{2}(h_{2j-1}^k)(h_{2j}^k) f''(\xi) \qquad \text{for some } \xi \in (x_{j-1}^{k-1}, x_j^{k-1}). \tag{3.11b}$$

If, on the other hand, f is discontinuous at a point, then the scale-coefficients $d_j^k(f)$ at this location are proportional to the size of the jump and remain large independent of refinement.

We can obtain data compression by setting to zero all scale coefficients which fall below a prescribed tolerance. Let us denote

$$\tilde{d}_j^k = \begin{cases} 0 & \text{if } |d_j^k| \le \varepsilon_k \\ \\ d_j^k & \text{if } |d_j^k| > \varepsilon_k, \end{cases} \tag{3.12a}$$

and

$$\tilde{v}^L = M^{-1} \cdot \{\tilde{d}^L, \ldots, \tilde{d}^1, v^0\}. \tag{3.12b}$$

Since the piecewise-linear interpolation is hierarchic (see appendix A) we get the following bound on the compression error:

$$\max_{0 \le i \le J_L} |v_i^L - \tilde{v}_i^L| \le \sum_{k=1}^{L} \varepsilon_k. \tag{3.13a}$$

Given any $\varepsilon > 0$ we can take

$$\varepsilon_k = 2^{k-L-1}\varepsilon \tag{3.13b}$$

and thus ensure by (3.13a) that

$$\max_{0 \le i \le J_L} |v_i^L - \tilde{v}_i^L| < \varepsilon. \tag{3.13c}$$

We remark that spline interpolation and spectral collocation are generally hierarchic and therefore the corresponding MR schemes are automatically stable (see [12]). However for the application that we have in mind, and for many others, we need a local interpolation method . For this purpose we use the following piecewise-polynomial interpolation: Let $\mathcal{S}_i^k$ denote a stencil of r consecutive points of X^k which includes x_i^k and x_{i+1}^k and let $p_i^k(x; v^k)$ denote the unique polynomial of degree $(r-1)$ which interpolates v^k at the points of this stencil . We define the piecewise-polynomial interpolation $\mathcal{I}^k(x; v^k)$ by

$$\mathcal{I}^k(x; v^k) = p_i^k(x; v^k) \quad \text{for} \quad x_i^k \le x \le x_{i+1}^k. \tag{3.14}$$

Up to now we have not specified the stencil $\mathcal{S}_i^k$ of r consecutive points of X^k that we assign to $[x_i^k, x_{i+1}^k]$. Clearly if we choose $\mathcal{S}_i^k$ independently of the data v^k then the most accurate choice is that of a centered stencil (away from the boundaries), i.e. for $r = 2s$ we take

$$\mathcal{S}_i^k = \{x_{i-s+1}^k, \ldots, x_{i+s}^k\} \quad \text{for} \quad s-1 \leq i \leq J_k - s, \tag{3.15a}$$

and near the boundaries

$$\begin{aligned} \mathcal{S}_i^k &= \{x_1^k, \ldots, x_r^k\} \quad \text{for} \quad 0 \leq i \leq s-1 \\ \mathcal{S}_i^k &= \{x_{J_k-r+1}^k, \ldots, x_{J_k}^k\} \quad \text{for} \quad J_k - s \leq i \leq J_k. \end{aligned} \tag{3.15b}$$

In [9] we presented a data-dependent piecewise-polynomial interpolation technique which avoids the Gibbs-phenomenon by an adaptive selection of stencil $\mathcal{S}_\rangle^{\|}$ in (314) ; we refer to this technique as Essentially Non-Oscillatory (ENO) interpolation. The basic idea of ENO interpolation is to assign to $[x_i^k, x_{i+1}^k]$ which is in the smooth part of the sampled function, a stencil $\mathcal{S}_\rangle^{\|} = \{\S_{\rangle,}^{\|}, \ldots, \S_{\rangle,+\nabla-\infty}^{\|}\}$ with $i_0 = i_0(i)$, which is likewise in the smooth part of the function (provided that this is possible, i.e. that discontinuities are well separated and are far enough from the boundaries). This is done by choosing $\mathcal{S}_\rangle^{\|}$ to be the stencil for which the interpolation polynomial $p_i^k(x; v^k)$ in (3.14) is the "smoothest" among all candidate-stencils, i.e. those of r consecutive points of X^k (starting with $x_{i_0}^k$) which contain both x_i^k and x_{i+1}^k, e.g. by taking $i_0(i)$ to be the index for which

$$\min_{i_0} \left| \frac{d^{r-1}}{dx^{r-1}} p_i^k(x; v^k) \right|$$

is attained among all candidate-stencils.

Example 3.2. Tensor-product grids in $[0,1]^2$

We consider continuous functions f

$$f : [0,1] \times [0,1] \longrightarrow \mathbf{R} \tag{3.16a}$$

which are discretized on the tensor-product grid

$$X^k = \{(x_i^k, x_j^k)\}_{i,j=0}^{N_k} \tag{3.16b}$$

by

$$f^k =: \{f_{i,j}^k\}_{i,j=0}^{N_k}, \quad f_{i,j}^k = f(x_i^k, x_j^k), \tag{3.16c}$$

where $\{x_i^k\}_{i=0}^{N_k}$ are the one-dimensional gridpoints in (3.7); note that here we replaced J_k by N_k, since we reserve J_k for the dimension of the vector space V^k in (2.1), i.e. $J_k = (N_k+1)^2$. Although this case is covered by the general framework in (2.10)-(2.12), it is more

convenient to describe the tensor-product extension of the one-dimensional MR scheme by using matrix notation. For this purpose we represent the two-dimensional array in (3.16c) as the $(N_k+1)\times(N_k+1)$ matrix A^k, and use the general algorithm of appendix B which describes tensor-product extensions of one-dimensional MR schemes for the input A^L.

We remark that the tensor-product algorithm inherits its stability from the one-dimensional MR scheme, and that existence of a MR basis for one-dimensional functions goes over to the corresponding two-dimensional ones. In this paper we shall use the tensor-product MR schemes for a uniform grid with the piecewise-polynomial interpolation (3.14)-(3.15) as our "yard stick" in order to measure the performence of the MR schemes for unstructured meshes in similar circumstances.

We remark that one can easily extend the tensor-product MR scheme in appendix B to the grid

$$X^k = \{(x_i^k, y_j^k)\}_{i,j=0}^{N_k}$$

where $\{y_j^k\}$ is a different partition of $[0,1]$ (see [16]).

Example 3.3. Triangulation in R^2

As a result of the success of Finite Element Method (FEM) for the numerical solution of PDEs, we have many triangulation techniques (some of them are packaged as computer codes) to construct a mesh of desired resolution with a good description of the boundary for complex geometries, and also corresponding interpolation methods for the nodes. Let us denote the triangles in such a mesh by $T_i^k \subseteq \Omega, \quad 1 \le i \le n_t$, and denote by X^k the set of vertices in these triangles; we refer to $\mathcal{T}^k =: \{T_i^k\}_{i=1}^{n_t}$ as the triangulation of Ω. Observe that $\Omega^k =: \cup_{i=1}^{n_t} T_i^k \subseteq \Omega$ is a polygon and that $\partial\Omega^k$ is a piecewise-linear approximation to $\partial\Omega$.

In our particular application we are interested in local approximations. For this purpose we use, as in example 3.1, the following piecewise-polynomial interpolation: Let $\mathcal{S}_i^k$ be a stencil of $s = r(r+1)/2$ points of X^k which is assigned to the triangle $T_i^k \in \mathcal{T}^k$ and includes its vertices . Let $p_i^k(x;\mathcal{D}_k f)$ denote the unique polynomial of degree $(r-1)$ for $r \ge 2$ which interpolates $f(x)$ at the points of the stencil $\mathcal{S}_i^k$, and define

$$\mathcal{I}_k(x;\mathcal{D}_k f) = p_i^k(x;\mathcal{D}_k f) \quad \text{for} \quad x \in T_i^k \tag{3.17}$$

Clearly this technique is exact for data of polynomial functions of degree less or equal $r-1$. Observe however that unlike 1D, $\mathcal{I}_k$ is not necessarily continuous on the boundary of T_i^k; in this case we take $\mathcal{F}$ to be the space $\mathcal{B}(\Omega)$ with the sup norm, rather than $C^0(\Omega)$.

For $r = 2$ the 3 points of $\mathcal{S}_i^k$ are necessarily the vertices of T_i^k, and $p_i^k(x;\mathcal{D}_k f)$ is the piecewise-linear function which interpolates f at these three points. Observe that in this

case $\{\mathcal{I}_k(x;\mathcal{D}_k f)\}$ is a sequence of continuous functions which forms an hierarchic sequence of approximation in $C^0(\Omega)$.

For $r > 2$ we have several reasonable choices of stencils; as in example 3.1 one can use this freedom to adapt the interpolation to the nature of the data by choosing the stencil in which the data is smoothest, and thus avoid the Gibbs' phenomenon (see [1] and [15]).

In subsection 5.1.3 we describe a simple strategy to select a "centered" stencil of 6 vertices for piecewise-quadratic interpolation ($r = 3$).

Our main problem is to design procedures for coarsening and refinement for which the vertices of $\mathcal{T}^{k-1}$ are contained in those of $\mathcal{T}^k$. Observe that in multigrid methods this is not a requirement but a desirable option. The design of such procedures is the topic of section 5.2.

4 MR schemes for cell-average discretization

Consider absolutely integrable functions $f \in \mathcal{F}$

$$f : \Omega \subset \mathbf{R}^m \longrightarrow \mathrm{R}, \qquad \mathcal{F} = L^1(\Omega) \tag{4.1a}$$

where Ω is a compact set, and let $\mathcal{C}^k = \{C_i^k\}_{i=1}^{J_k}$ be a set of cells such that

$$\Omega^k =: \overline{\cup_{i=1}^{J_k} C_i^k} \subseteq \Omega, \qquad C_i^k \cap C_j^k = \emptyset \ \text{ for } \ i \neq j, \tag{4.1b}$$

We define the cell-average discretization of f by

$$(\mathcal{D}_k f)_i = \frac{1}{|C_i^k|}\int_{C_i^k} f(x)dx, \qquad |C_i^k| = \int_{C_i^k} dx. \tag{4.1c}$$

Next let us consider a refinement sequence $\{\mathcal{C}^k\}_{k=0}^L$, in which $\mathcal{C}^k$ is formed from $\mathcal{C}^{k-1}$ by dividing each cell C_i^{k-1} into, say q, disjoint cells $\{C_{i_\ell}^k\}_{\ell=1}^q$,

$$\overline{\cup_{\ell=1}^q C_{i_\ell}^k} = \overline{C_i^{k-1}}. \tag{4.2a}$$

Alternatively we can consider (4.2a) to be a coarsening procedure in which we agglomerate every q cells of $\mathcal{C}^k$ into a larger cell of $\mathcal{C}^{k-1}$; the only reason that we take here a fixed q is to simplify the notations. In any case the sequence of discretization $\{\mathcal{D}_k\}_{k=0}^L$ is nested and it follows from the additivity of the integral that

$$(\mathcal{D}_{k-1} f)_i = \frac{1}{|C_i^{k-1}|}\sum_{\ell=1}^{q} |C_{i_\ell}^k| (\mathcal{D}_k f)_{i_\ell} =: (D_k^{k-1}\mathcal{D}_k f)_i \tag{4.2b}$$

which directly defines the decimation operator in (2.2). Let $\mathcal{R}_k$ denote any reconstruction from cell-averages, linear or not, and let e^k denote the prediction error in (2.7a), then

$$D_k^{k-1} e^k = 0 \Rightarrow \sum_{\ell=1}^{q} |C_{i_\ell}^k| e_{i_\ell}^k = 0. \tag{4.3}$$

This relation shows that we can define the scale coefficients d^k by taking $(q-1)$ properly chosen linear combinations of the q prediction errors $\{e_{i_\ell}^k\}_{\ell=1}^q$ in each cell C_i^{k-1} . These linear combinations should be chosen so that together with (4.3) they constitute an invertible system of q linear equations for the prediction errors $\{e_{i_\ell}^k\}_{\ell=1}^q$ in the cell C_i^{k-1} . (see e.g. [16] for such combinations in representation of matrices).

Example 4.1. Nonuniform partition of [0,1]

Let us consider absolutely integrable functions f in [0,1]

$$f \in \mathcal{F} = L^1[0,1],$$

and let

$$\mathcal{C}^k = \{C_i^k\}_{i=1}^{J_k}, \quad C_i^k = (x_{i-1}^k, x_i^k), \tag{4.4a}$$

where $\{x_i^k\}$ are the gridpoints of X^k in (3.7); observe that

$$\overline{C_i^{k-1}} = \overline{C_{2i-1}^k \cup C_{2i}^k}. \tag{4.4b}$$

Let $v^k = \mathcal{D}_k f$ denote the cell-averages in (4.1c); v^{k-1} is obtained from v^k by the decimation (4.2b), i.e.

$$v_i^{k-1} = \frac{1}{|C_i^{k-1}|}(|C_{2i-1}^k| v_{2i-1}^k + |C_{2i}^k| v_{2i}^k), \ i = 1, \ldots, J_{k-1}. \tag{4.5}$$

It follows from (4.3) that the prediction error e^k satisfies the relation

$$|C_{2i-1}^k| e_{2i-1}^k + |C_{2i}^k| e_{2i}^k = 0 \quad \text{for} \quad i = 1, \ldots, J_{k-1}. \tag{4.6a}$$

Therefore, if we define

$$d_j^k = \frac{1}{2|C_j^{k-1}|}(|C_{2j-1}^k| e_{2j-1}^k - |C_{2j}^k| e_{2j}^k) \quad \text{for} \quad j = 1, \ldots, J_{k-1} \tag{4.6b}$$

we can recover the prediction error e^k by

$$\begin{cases} e_{2i-1}^k = d_i^k \cdot |C_{2i-1}^k| / |C_{2i-1}^k| \\ e_{2i}^k = -d_i^k \cdot |C_{2i-1}^k| / |C_{2i}^k| \end{cases} \tag{4.6c}$$

The direct MR transform $\hat{v}_M = M \cdot v^L$ is given by the algorithm

$$\begin{cases} DO \ k = L, \ldots, 1 \\ v_i^{k-1} = \frac{1}{|C_i^{k-1}|}(|C_{2i-1}^k| v_{2i-1}^k + |C_{2i}^k| v_{2i}^k), \quad i = 1, \ldots, J_{k-1} \\ d_j^k = [|C_{2j-1}^k| v_{2j-1}^k - \int_{C_{2j-1}^k} (\mathcal{R}_{k-1} v^{k-1})(x) dx] / |C_j^{k-1}|, \quad j = 1, \ldots, J_{k-1} \end{cases} \tag{4.7}$$

The inverse MR transform $v^L = M^{-1} \cdot \hat{v}_M$ is given by

$$\begin{cases} DO \ k = 1, \ldots, L \\ DO \ i = 1, \ldots, J_{k-1} \\ v_{2i-1}^k = \frac{1}{|C_{2i-1}^k|}[\int_{C_{2i-1}^k} (\mathcal{R}_{k-1} v^{k-1})(x) dx + d_i^k |C_i^{k-1}|] \\ v_{2i}^k = (|C_i^{k-1}| v_i^{k-1} - |C_{2i-1}^k| v_{2i-1}^k) / |C_{2i}^k| \end{cases} \tag{4.8}$$

Observe that the last statement of (4.8) is obtained from (4.5).

In [17] we showed that any interpolation method in example 3.1 gives rise to a corresponding method for reconstruction from cell-averages by the following "reconstruction via primitive function" technique: Given cell-averages $v^k = \mathcal{D}_k f$ we calculate the point values of the "primitive function"

$$F_i^k = F(x_i^k), \quad F(x) = \int_0^x f(y)dy$$

by

$$F_0^k = 0, \quad F_i^k = \sum_{j=1}^{i} |C_j^k| v_j^k, \quad 1 \le i \le J_k, \tag{4.9a}$$

and define

$$(\mathcal{R}_k v^k)(x) = \frac{d}{dx}\mathcal{I}_k(x; F^k), \tag{4.9b}$$

where $I^k(x; F^k)$ is any interpolation of the values $F^k = \{F_i^k\}_{i=0}^{J_k}$ at the grid points of X^k in (3.7). For example, if we take the piecewise-linear interpolation (3.10), we get that $\mathcal{R}_k v^k$ is the piecewise-constant reconstruction

$$(\mathcal{R}_k v^k)(x) = v_i^k \quad \text{for} \quad x \in C_i^k. \tag{4.10}$$

Example 4.2. Tensor-product meshes in $[0,1]^2$

We consider absolutely integrable functions f

$$f : [0,1] \times [0,1] \longrightarrow \mathrm{R} \tag{4.11a}$$

which are discretized on the tensor-product mesh

$$\mathcal{C}^k = \{C_i^k \times C_j^k\}_{i,j=1}^{N_k} \tag{4.11b}$$

by

$$\bar{f}^k =: \{\bar{f}_{i,j}^k\}_{i,j=1}^{N_k}, \ \bar{f}_{i,j}^k = \frac{1}{|C_i^k| \cdot |C_j^k|} \int_{C_i^k} \int_{C_j^k} f(x_1, x_2) dx_1 dx_2 \tag{4.11c}$$

where $\{C_i^k\}_{i=1}^{N_k}$ are the one-dimensional cells in (4.4a); note that here we replaced J_k by N_k, since we reserve J_k for the dimension of the vector space V^k in (2.1), i.e. $J_k = (N_k)^2$. Although this case is covered by the general framework in (2.10)-(2.12), it is more convenient to describe the tensor-product extension of the one-dimensional MR scheme by using the matrix notation of appendix B. Therefore we represent the two-dimensional array in (4.11c) by the $N_k \times N_k$ matrix A^k, and use the general algorithm of appendix B to describe the tensor-product MR scheme for the input A^L.

We remark that, as in the corresponding example 3.3, the cells in (4.11b) can be defined as a tensor-product of different partitions in the two space variables.

Example 4.3. Unstructured meshes in R^m .

Using agglomeration as a coarsening technique may result in cells which are general polygons. Such a technique is described in section 5.2. In the following we describe a piecewise-polynomial reconstruction technique which is suitable for our purpose (see [15]): Let us denote by $\mathcal{S}_i^k$ a stencil of $s(r)$ *cells* in $\mathcal{C}^k$ which includes C_i^k, i.e.

$$\mathcal{S}_i^k = \{C_{i_m}^k\}_{m=1}^{s(r)}, \qquad C_i^k \in \mathcal{S}_i^k; \tag{4.12a}$$

here $s(r)$ is the number of coefficients in a polynomial of degree $(r-1)$ in R^m. Let $p_i^k(x; \mathcal{D}_k f)$ denote the unique polynomial of degree $(r-1)$ which attains the averages $(\mathcal{D}_k f)_{i_m}$ in $\mathcal{S}_i^k$, i.e. the one which satisfies the following system of $s(r)$ linear equations for its $s(r)$ coefficients:

$$\frac{1}{|C_{i_m}^k|}\int_{C_{i_m}^k} p_i^k(x; \mathcal{D}_k f)dx = (\mathcal{D}_k f)_{i_m}, \qquad m = 1, \ldots, s(r), \tag{4.12b}$$

and define

$$(\mathcal{R}_k \mathcal{D}_k f)(x) = p_i^k(x; \mathcal{D}_k f) \quad \text{for} \quad x \in C_i^k. \tag{4.12c}$$

Clearly (4.12) defines a reconstruction of $\mathcal{D}_k f$ which is exact for polynomial functions of degree less or equal $(r-1)$, and thus is r-th order accurate. Such a polynomial exists if

1. $s(r) = r(r+1)/2$,
2. a Vandermonde type condition on $\mathcal{S}_i^k$ is satisfied.

For a general discussion on this kind of polynomial approximation and its computation, see [1] and [15]. In the following, we always assume that the stencils we construct are admissible. In practice, there is no particular restriction, see [1].

Note that for $r = 1$ in (4.12) we have $s(r) = 1$ and we get the piecewise-constant reconstruction

$$(\mathcal{R}_k \mathcal{D}_k f)(x) = \sum_i (\mathcal{D}_k f)_i \chi_{C_i^k}(x), \tag{4.13}$$

where $\chi_C(x)$ denotes the characteristic function of the set C,

$$\chi_C(x) = \begin{cases} 1 & x \in C \\ 0 & \text{otherwise} \end{cases}. \tag{4.14}$$

In [15] we present an hierarchial algorithm for the selection of a "centered" stencil, which is applicable even to completely unstructured meshes $\mathcal{C}^k$ in R^m. In this context the "centered" stencil is defined as the one which minimizes the reconstruction error for the one-higher degree polynomials (i.e. degree r). This algorithm is of "crystal growth" type: starting with the cell C_i^k we begin to add successively, one cell at a time, to the cluster of cells that we

have at the beginning of each step. The cell which is being added is selected from the set of all side-neighbors of the existing cluster by the requirement that it will minimize the reconstruction error of suitably chosen monomials.

In [15] we also present an adaptive "crystal growth" algorithm which is designed to assign a stencil $\mathcal{S}_i^k$ from the smooth part of $f(x)$, if available, to all cells C_i^k which are themselves in the smooth part of $f(x)$. This way a Gibbs-like phenomenon is avoided, and the resulting approximation is r-th order accurate everywhere, except at cells which contain a discontinuity. This is accomplished by selecting the cell from the set of side-neighbors which minimizes the derivatives of the so-defined reconstruction.

We refer the reader to [1] for details of special ENO reconstruction techniques for triangulated meshes. Following the same principles, we present in section 5.2 a simple technique to compute a "central" stencil, that works on triangulated meshes. This central stencil is then used to define the polynomial $p_i^k(x; \mathcal{D}_k f)$ of (4.12a) in section 6.

We remark that if we generate the nested sequence of $\{\mathcal{C}^k\}_{k=0}^L$ by refinement of a triangulated mesh C^0, then all the meshes may be triangulations and then we can use special strategies for assignment of stencils. As in example 3.3 we find that there are some refinement methods that have a built-in convenient assignment of stencils for the purpose of reconstruction. In the case of cell-average discretization these are typically of the "dual mesh" type: e.g. one can start with a covering by hexagons, each containing 6 triangles, and then divide each triangle into 4 by connecting the midpoints of its sides as before. This results in a new system of smaller hexagons (4 times as many). To each triangle of $\mathcal{T}^k$ we assign the 6 triangles of the hexagon to which it belongs – this is a convenient assignment of stencil for 3-rd order accurate reconstruction from cell-averages.

5 Implementation

In this section we show how to apply the general framework presented above to two kinds of representation of functions on unstructured meshes : point values and average values. In both cases, we describe the discretization operator and the reconstruction operator.

We start with notations: Let Ω be the computational domain, and assume that it is bounded. The set Ω is approximated by a polygon Ω^L which is triangulated ; let $\mathcal{T}^L$ denote the triangulation and let $X^L = \{M_i\}_{1 \le i \le J_L}$ denote its nodes. As before we denote by $k = L$ the finest level of resolution in the sequence and by $k = 0$ is the coarsest. In general we shall refer to the number of nodes by n_s and to the number of triangles by n_t .

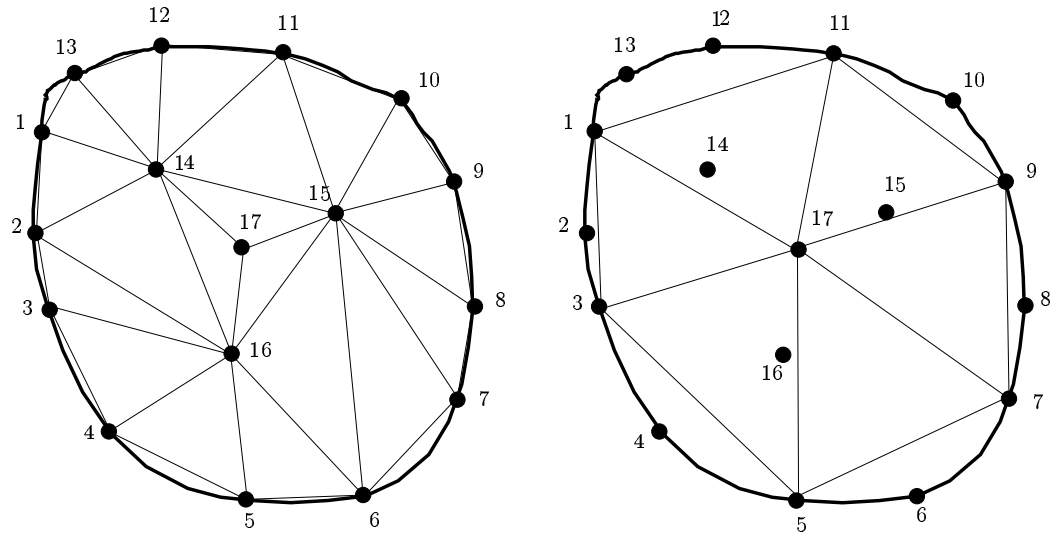
5.1 Discretization by point values

In order to define a nested sequence of discretization $\{\mathcal{D}_k\}_{1 \le k \le L}$ as in section 3, one has to define a sequence of nested sets $\{X^k\}_{1 \le k \le L}$, $X^{k-1} \subset X^k$. In constructing these sets we have to make sure that each of the sets X^k has enough points on the boundary of Ω so that the approximation to $\partial\Omega$ is adequate for the k-th level of resolution. The purpose of this section is to describe : 1- two ways of constructing the sets X^k, 2- the stencil assignement to each of the triangles T_i^k which is used to define the interpolation polynomial in (3.17).

5.1.1 Coarsening

We start from a given fine triangulation, and have to construct coarser and coarser triangulations. To do so we have adopted the coarsening procedure of [18] which we review in the following. We start with the triangulation $\mathcal{T}^L$ and eliminate points of X^L to obtain X^{L-1}; then we produce a triangulation $\mathcal{T}^{L-1}$ for which X^{L-1} are the nodes, and repeat this cycle till we get to the coarsest level. In order to have better control of the shape of the boundary we make the convention that the points along the boundary $\partial\Omega^k$ are listed first in the array X^k in, say, counterclockwise order, and that we also keep an ordered list of the boundary segments of $\partial\Omega^k$. One eliminates points by the following method :

Boundary : one starts with x_1^L which is a boundary point and marks all of its neighbors. This point will also be the first point of X^{L-1}. Then, following the boundary, one looks for the first unmarked point, adds it to X^{L-1} and marks all of its neighbors. Once the list of boundary points in X^L has been exhausted, one goes to the next step in which we eliminate points from the interior. In the example of Figure 1, point 13 is eliminated because it is a neighbor of point 1; observe that 12 is also eliminated because it is a



Initial mesh Coarsened mesh

Figure 1:

neighbor of 11. The points 14, 15 and 16 are eliminated because they are neighbors of a point on the boundary that remains in X^{L-1}, namely $\{1, 3, 5, 7, 9, 11\}$.

Interior points : One adds to X^{L-1} the first not yet marked point of X^L, and mark all of its neighbors. Then one proceeds the same way until the list X^L has been exhausted. In the example of Figure 1, the final set X^{L-1} is then $\{1, 3, 5, 7, 9, 11, 17\}$.

The boundary points are listed first in order to have a better control of the geometry of the "coarsened" approximation of Ω^L. Following [18], one may impose some additional constraints on the process of elimination, for example to keep certain points of the boundary of Ω^L in X^{L-1}, because otherwise the geometry of the problem could be changed.

Next we prepare an ordered list of the boundary segments of $\partial\Omega^{L-1}$ and proceed to generate a triangulation $\mathcal{T}^{L-1}$ of the set X^{L-1}. Here, as in [18], the triangulation $\mathcal{T}^{L-1}$ is obtained from the Delaunay triangulation of a square large enough to contain X^{L-1}. Using our list of segment elements in $\partial\Omega^{L-1}$ we modify this triangulation, if needed, so that these boundary segments are sides of the triangles. Once this is done we can identify the exterior of Ω^{L-1} and remove it. This way we get a triangulation $\mathcal{T}^{L-1}$ which is consistent with $\mathcal{T}^L$ in the following sense: the boundary points of $\mathcal{T}^{L-1}$ are also boundary points of $\mathcal{T}^L$ and no boundary point of $\mathcal{T}^{L-1}$ is an interior point of $\mathcal{T}^L$.

When we want L levels of mesh, one repeats the above algorithm $L-1$ times . It is difficult to know in advance the number of points that each level will have. The elimination procedure depends strongly on the ordering of the nodes of the fine mesh in the array X^L. In general it is desirable to eliminate points away from the boundary by shells. Experimentally,

one can see that if the fine mesh has n_s nodes, then X^{L-1} has between $\frac{n_s}{3}$ and $\frac{n_s}{4}$ nodes, and so on.

5.1.2 Refinement

Another way of computing a sequence of nested subsets of X^L is to proceed in the reverse way : define first a coarse mesh, then add nodes in a regular way. This approach is especially simple if the domain Ω is itself a polygon. A simple way of adding nodes is the following: Consider a triangle T with vertices A, B, C. We define the barycentric coordinates of the point M by the scalars Λ_A, Λ_B, Λ_C which satisfy the following two equations:

- $\Lambda_A + \Lambda_B + \Lambda_C = 1$,
- $\Lambda_A\, A + \Lambda_B\, B + \Lambda_C\, C = M$.

These coordinates are first order polynomials of the coordinates of M. A point M is an interior point if and only if its barycentric coordinates are positive. Let us consider the set of points M_{ij} inside T which are defined in terms of their barycentric coordinates by :

$$\Lambda_A(M_{ij}) = \frac{i}{n+1} \quad , \quad \Lambda_B(M_{ij}) = \frac{j}{n+1}; \tag{5.1}$$

here n is an integer number, i and j satisfy $0 \leq i, j \leq n+1$ and $0 \leq i+j \leq n$. This defines a set of $(n+1)(n+2)/2$ points. The triangulation is obtained from these points as shown in Figure 2 for $n = 2$.

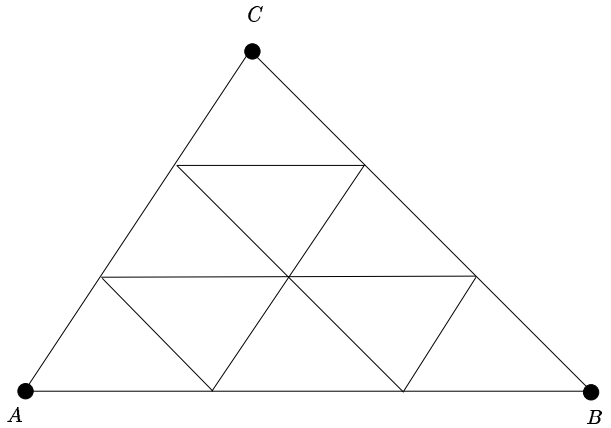


Figure 2:

5.1.3 Interpolation

For the purpose of multiresolution representation one needs a right-inverse to $\mathcal{D}_k$. Here it is the Lagrange interpolation which is uniquely defined once we provide the stencils $\mathcal{S}_i^k$ in (3.17). In order to simplify the presentation, we only describe what can be done for a third order interpolation. At the end of this paragraph, we indicate how this can be extended to more general situations.

In order to define the stencils, we have chosen a rather heuristic approach. We consider a triangulation $\mathcal{T}$ that may be any of the triangulations we have defined in the previous paragraph, and let T be one of its triangles, as in Figure 3.

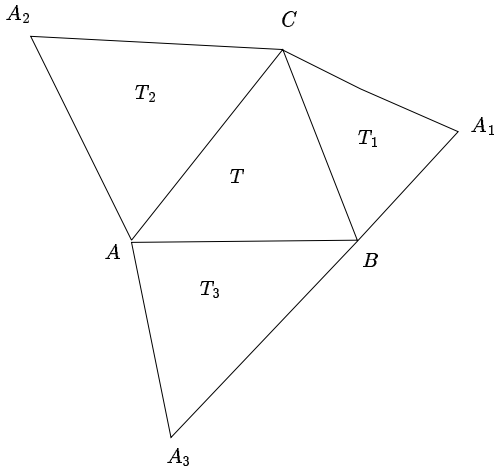


Figure 3:

We first assume that T is an interior triangle, i.e. none of its edges are on the boundary. Then there exists a triangle on the other side of each side of T, namely T_1, T_2, T_3 (see Figure 3), which we refer to as side-neighbors. The set of all the vertices of T, T_1, T_2, T_3 has 6 elements. In general, this set is is not geometrically degenerate, i.e. there is a unique solution to the Lagrange interpolation problem. It would not be the case if these points were on a conic.

Now, if T is on the boundary, we may make a similar construction, in general. If T is a triangle that plays the same role as T_1, T_2 or T_3 above, then we may take the stencil made by these six points as a stencil for T. If no such construction is possible, it means that T is in a sharp angle of the triangulation and other heuristics must be found. For example : T has at least one neighbor which itself has at least one neighbor, etc. Proceeding this way, one can find six vertices that may be candidates for stencils. We remark that this situation is rarely encountered in a fine mesh, and its occurence can serve as a criterion for over-coarsening.

This method of constructing stencils can be generalized to higher degree polynomials, for example see [1].

If one constructs $\{T^k\}_{k=1}^L$ from a given coarse triangulation $\mathcal{T}^0$ by the refinement in (5.1), then there is an obvious and convenient way to assign stencils. At any level, a triangle T is obtained from a "coarser" triangle T' where T' has been subdivided into four smaller triangles, among which is T, see Figure 3. Then, obviously, the stencil made of the node of T' and the three mid-points of its three edges is an admissible stencil. Observe that this assignment of stencil is particularly efficient since the same stencil is used for all the points of T' (and not T only as above) and that there are closed formulas that gives the Lagrange interpolation of any function in T' : if Λ_1, Λ_2, Λ_3 are the barycentric coordinates of T', if we denotes by f_1, f_2, f_3 the values of f at the nodes of T' and by f_{12}, f_{13} and f_{23} those at the mid points (with obvious notations), then the Lagrange interpolation polynomial is well known :

$$P = \sum_{i=1}^{3} f_i \Lambda_i(1 - 2\Lambda_i) + \sum_{midpoints} f_{ik}\, 4\Lambda_i\Lambda_k. \tag{5.2}$$

Obviously, this method can be generalized to higher orders of accuracy: For r-th order approximation we take a subdivision mesh as above with $n = r - 1$. It is well known that close formulas of the type of (5.2) exists, and make the procedure very efficient. Moreover, unlike the first method of interpolation, we get here an hierarchial sequence of approximation which automatically leads to a stable MR scheme (see appendix A).

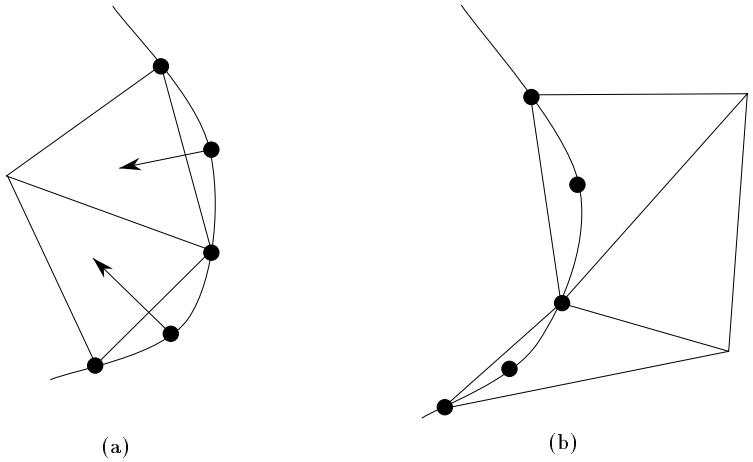


Figure 4: (a) : convex boundary, (b) : concave boundary

In all cases, the stencil one gets is used to have an approximation in T. Depending on the local concavity of the boundary, it is possible for some points of $\mathcal{T}^k$ to be in the exterior of the coarser triangulation $\mathcal{T}^{k-1}$, see Figure 4; note that this does not contradict the requirement

of nestedness. In this case, we predict the corresponding values by extrapolation from the closest triangle in $\mathcal{T}^{k-1}$.

5.2 Discretization by cell average

In this section, we assume that we are given a partition of Ω by disjoint cells $\mathcal{C}^L = \{C_i^L\}_{i=1}^{J_L}$ such that

$$\Omega^L =: \overline{\cup_{i=1}^{J_L} C_i^L} \subseteq \Omega, \qquad C_i^L \cap C_j^L = \emptyset \ \text{ for } \ i \neq j, \tag{5.3}$$

In order to generate this fine partition we can start with a triangulation $\mathcal{T}^L$ of Ω and define cells in the following three ways : (1) The triangles of $\mathcal{T}^L$; (2) the dual mesh (i.e. the Voronoi partition) or (3) the set of cells which is used in Finite Elements methods : For any point vertex M of $\mathcal{T}^L$ consider all the triangles that have M as a vertex ; then for any of these triangles connect the midpoints of the two edges that share M and the centroid of the triangle. The area between these lines and M is collected into a cell which is identified with the point vertex M.

Each of these choices has its advantages and drawbacks. The choice (1) of triangles enables us to have cells with a small and *constant* number of sides. However, for n_s nodes, the mesh has $\simeq 3n_s$ triangles, and this ratio is even worse in 3D; consequently, the resulting memory cost may be high. The choice (3) of "Finite Element" cells enables us to refine the mesh in a reasonable way, if needed, but on the other hand, the number of sides of a cell is not known *a priori*. The choice (2) of cells based on the Voronoi diagram of the mesh offers an attractive compromise.

One may also consider other choices of cells, in particular if we do not use a partition based on a triangulation but a partition with various kinds of elements : squares, triangles, etc.

The decimation and prediction operators are defined in section 4. We provide here a description of : 1- the coarsening/refinement algorithm that we have used and 2- a procedure to assign the stencil $\mathcal{S}_i^k$ to the cell $\mathcal{C}_i^k$; then the reconstruction is defined by (4.12). operator.

5.2.1 Coarsening

In order to define a set of nested discretizations, we follow ideas which were applied in Multigrid techniques on unstructured meshes, in particular by Lallemand et al. [19] and then by others, e.g. [22]. The partitions $\mathcal{C}^k = \{C_i^k\}_{i=1}^{J_k}$ are defined by successive agglomeration of cells. To describe this process we need the following notation: Let $\mathcal{C}$ denote a sequence of cells

which includes C_i ; we denote by $I_i(\mathcal{C})$ the set of indices, including i, of all the side-neighbors of C_i in $\mathcal{C}$. We start with the given fine level $\mathcal{C}^L$, and assume that we have already defined $\mathcal{C}^m$ for $k \leq m \leq L$. We proceed now to define $\mathcal{C}^{k-1}$ as follows.

First cell : We consider $\{C_\ell^k\}_{\ell \in I_1(\mathcal{C}^k)}$; these are the cells in $\mathcal{C}^k$ which share a common side with C_1^k. We define

$$C_1^{k-1} = \bigcup_{\ell \in I_1(\mathcal{C}^k)} C_\ell^k.$$

Then we remove all the cells C_ℓ^k with $\ell \in I_1(\mathcal{C}^k)$ from $\mathcal{C}^k$, and denote the new sequence by $\tilde{\mathcal{C}}^k$.

Other cells : Assume that $\{C_1^{k-1}, \cdots, C_{i-1}^{k-1}\}$ have been defined, take the first cell in $\tilde{\mathcal{C}}^k$ and denote its index in $\mathcal{C}^k$ by j . We define

$$C_i^{k-1} = \bigcup_{\ell \in I_j(\tilde{\mathcal{C}}^k)} C_\ell^k$$

and remove all the cells with indices in $I_j(\tilde{\mathcal{C}}^k)$ from $\tilde{\mathcal{C}}^k$; we rename the new set as $\tilde{\mathcal{C}}^k$.

Once the list of elements of $\tilde{\mathcal{C}}^k$ has been exhausted, the set of coarsened cells so obtained constitutes $\mathcal{C}^{k-1}$. We continue this process until we get to the coarsest level $k = 0$.

It is useful to notice that, as in the previous case, the structure of the coarsened cells depend a lot on the ordering of $\mathcal{C}^L$. In particular, it is possible that a "coarsened" cell C_i^k for some level $k > 1$ may be made of only one "finer" cell C_j^k of the previous level. We do not want this situation, because the analysis on this cell do not provide any information. The same argument shows that the number of "finer" cells that have been agglomerated to make a "coarser" cell is not controlled.

In [22] some heuristics, based on the effective problem one wants to solve, are presented to control the number of cells that are agglomerated to make one coarser cell. Here, we have chosen a more primitive method. If a "coarse" cell is too small, then it is agglomerated to one of its neighbours. We choose the one that is made of the least number of "fine" elements. Similarly, if a cell becomes too large, one stops the agglomeration process.

As in the case of point value discretization, another method to construct a sequence of nested discretization may also be devised. Instead of going from the fine level to the coarse one, we can proceed in the opposite direction. Let n be an integer, and $\mathcal{T}_L$ a triangulation of a domain Ω. This domain is assumed to have a polygonal boundary. For each of its triangles T, we consider the $n(n+1)/2 - 3$ points of barycentric coordinates

$$\Lambda_A(M_{ij}) = \frac{i}{n} \quad , \quad \Lambda_B(M_{ij}) = \frac{j}{n}$$

as in subsection 5.1.2, see Figure 2. We can make $(n-1)^2$ new triangles. By proceeding inductively this way, we construct a sequence of set of triangles ; here the cells are triangles. The corresponding coarsening procedure is simple : for each level, one considers the set of triangles generated by a triangle T of the next coarser level.

5.2.2 Reconstruction

For any level k, we take the reconstruction to be the piecewise polynomial function in (4.12). The only remaining thing to do is to to describe the assignment of the stencil $\mathcal{S}_i^k$ for the cell C_i^k. For the sake of clarity, we give the details of the case $r = 3$ which corresponds to piecewise-quadratic reconstruction, though a more general discussion can be given.

We follow the procedure of subsection 5.1.2. For that, we identify each of the cells C_i^k with a point which we take to be the centroid of C_i^k (even though it may not belong to C_i^k). For this set of points we construct a triangulation $\mathcal{T}'$, here a Delaunay mesh. It is clear that some constraints, coming from the geometry of the problem have to be added. The most obvious constraint is that two cells that are not close (because they are separated by a hole) should not be connected, see Figure 5. We decide that a triangle is inadmissible if its centroid does not belong to the fine triangulation $\mathcal{T}^L$. For example, in Figure 5 the exact domain is inside

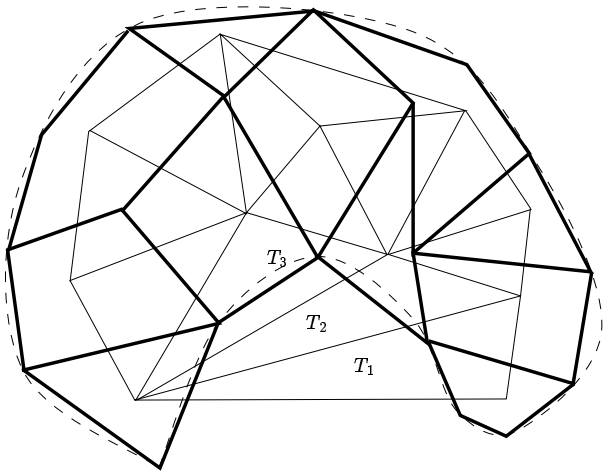


Figure 5:

the dotted line. The partition is made of the cells which are bounded by the thick lines. The corresponding triangulation of these cells is represented by the thin lines. The triangles T_1 and T_2 are not accepted because their centroid is outside of the computational domain, while T_3 is accepted. Another constraint is that the accepted triangles should not be too flat. This

case may be encountered at an almost flat boundary. The boundary triangles are tested as follows : we consider all the boundary points of $\mathcal{T}_l'$. For any of these points we consider the triangles having it as vertex. We compute the average ratio $(\frac{h}{\rho})_{av}$ of its circumcircle and its inner circle. If this ratio for a boundary triangle is too far from the average one,

$$|1 - \frac{h/\rho}{(h/\rho)_{av}}| > \delta$$

we remove this triangle. We have chosen $\delta = 0.1$.

This choice of stencil is so that this reconstruction procedure is not hierarchical in general. On the other hand, the refinement procedure of the previous section enable to construct a hierarchical reconstruction sequence. If we want a reconstruction with piecewise polynomial of degree $(r-1)$, the reconstruction will be hierarchical provided the triangles are divided into m^2 smaller triangles with $m^2 \geq r(r+1)/2$. We give an example where $r = 3$ and $m = 3$, see Figure 6 (note that any triangle may be mapped onto the standard triangle of Figure 6).

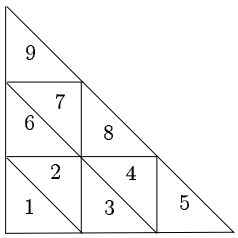


Figure 6:

We have 10 points and 9 triangles denoted $1, \cdots, 9$. By symmetry, it is necessary to describe stencils for triangles 1, 2 and 3 only. Among the many choice, we favor those giving the smallest support, namely :

- for 1 and 2, we take $S_1 = \{1, 2, 3, 4, 6, 8\}$,
- for 3, we take $S_2 = \{2, 3, 4, 6, 7, 8\}$.

It is easy to check that they are admissible by computing the determinants

$$\left| \frac{\int_k x^i y^j}{|k|} \right|, \ k \in S_1 \ \text{ or } \ S_2.$$

One finds -2 for S_1 and $4/9$ for S_2. It is interesting to notice that some combinations are not admissible, for example $\{1, 2, 3, 4, 6, 7\}$.

6 Numerical results

We have performed two kinds of numerical tests : for the point value discretization, we have tested the two methods of sections 3 and 4 ; for the cell average discretization, only the general method has been tested. In this case, we also compare our results with a multiresolution scheme where centered tensor product stencils and a regular grid diadic Cartesian grid is used.

As described in section 2, the method involve three steps : decoding, truncation and encoding. The data v^L on the fine level are represented as :

$$Mv^L = (v^0, d^1, \cdots, d^{L-1})^T$$

where v^0 is the representation of v^L on the coarsest level, and d^l are scale coefficients. In the truncation step , we truncate the scale coefficients according to

$$\tilde{d}_i^l = \begin{cases} 0 & \text{if } |d_i^l| \leq \epsilon_k, \\ d_i^l & \text{else.} \end{cases}$$

In the numerical experiments to follow, we take ϵ_k to be :

- for point values, $\epsilon_k = \epsilon$,
- for cell averages,

  $$\epsilon_k = \alpha_k \epsilon_{k+1}$$

  with α_k defined by :

  $$\alpha_k = \max_{C_i^k \in \mathcal{C}^k} \{ \sqrt{\frac{|C_j^{k+1}|}{|C_i^k|}}, \text{ where } C_i^k = \bigcup_{j \in I_i^k} C_j^{k+1} \},$$

  that is the maximum ratio of the area between a cell of level k and all the cells of level $k+1$ that have been agglomerated in it. For Cartesian grid, we take $\alpha_k = 2$.

We define the compression factor μ to be

$$\mu = \frac{N_L}{N_0 + \Big| \{(i,l) \text{ such that } |d_i^l| > \epsilon_k\} \Big|}.$$

We have performed numerical tests on two functions :

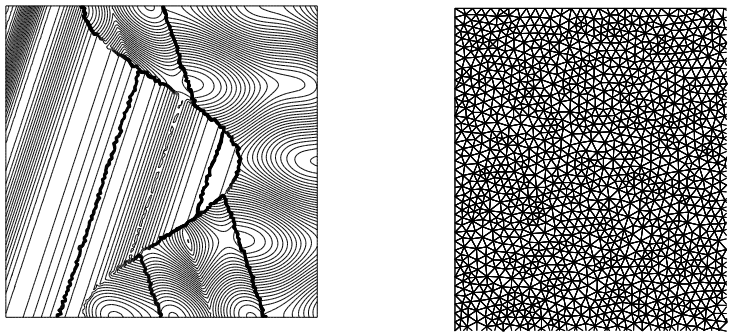
1. $f_1(x,y) = \cos 2\pi(x^2 + y^2)$ in $[0,1]^2$,

2. f_2, shown on Figure 7, is defined in $[-1,1]^2$ by

$$\begin{aligned} &\text{if } \; x \leq \tfrac{1}{2}\cos(\pi y), \quad f_2(x,y) = u_{\sqrt{\pi/2}}(x,y), \\ &\text{if } \; x > \tfrac{1}{2}\cos(\pi y), \quad f_2(x,y) = u_{-\sqrt{\pi/2}}(x,y) + \cos(2\pi y) \end{aligned}$$

where u_ϕ is

$$\left.\begin{array}{l} \text{if } r \leq -\frac{1}{3}, \; u_\phi(x,y) = -r\sin(\frac{\pi}{2}r^2) \\ \text{if } r \geq \frac{1}{3}, \; u_\phi(x,y) = 2r - 1 + \frac{1}{6}\sin(3\pi r) \\ \text{if } |r| < \frac{1}{3}, \; u_\phi(x,y) = |\sin(2\pi r)| \end{array}\right\} \quad \text{where} \quad r = x + \tan(\phi)y.$$



(a) (b)

Figure 7: (a) :Isolines of f_2, (b) : zoom of the left upper corner of the mesh.

6.1 Point value discretization

In a first example, the mesh has been obtained by refining 4 times a coarse mesh according to the procedure of subsection 5.1.2, with $n = 2$. It has 56 633 nodes and 112 384 triangles. The maximum radius of the triangles is $\rho \simeq 9\ 10^{-3} \simeq 1/156$ and $J_L \simeq 240^2$. We use 3 levels. The results are displayed in Table 1. For comparison sake, we show in Table 2 results which are obtained from a Cartesian regular grid. From the approximation error point of view, we should consider the maximal radius. In this respect we have to compare to a grid of 156×156. From the point of view of number of elements, we have to compare to a grid

Function	ϵ	μ	error (L^∞)
f_1	10^{-2}	57.43	$1.28\ 10^{-2}$
	10^{-3}	25.08	$1.59\ 10^{-3}$
	10^{-4}	5.85	$1.84\ 10^{-4}$
f_2	10^{-2}	11.75	$1.54\ 10^{-2}$
	10^{-3}	7.96	$1.63\ 10^{-3}$
	10^{-4}	3.45	$1.71\ 10^{-4}$

Table 1: Results for the refined grid

Function		64×64		128×128		256×256	
	ϵ	μ	error (L^∞)	μ	error (L^∞)	μ	error (L^∞)
f_1	10^{-2}	7.37	$1.85\ 10^{-2}$	29.04	$3.37\ 10^{-2}$	115.3	$4.33\ 10^{-2}$
	10^{-3}	1.94	$2.0\ 10^{-3}$	7.06	$2.64\ 10^{-3}$	28.02	$3.09\ 10^{-3}$
	10^{-4}	1.11	$1.28\ 10^{-4}$	1.83	$1.95\ 10^{-4}$	6.9	$2.12\ 10^{-4}$
f_2	10^{-2}	2.55	$2.34\ 10^{-2}$	5.57	$3.70\ 10^{-2}$	11.81	$4.6\ 10^{-2}$
	10^{-3}	1.26	$1.27\ 10^{-3}$	3.10	$1.61\ 10^{-3}$	7.96	$2.54\ 10^{-3}$
	10^{-4}	1.08	$9.75\ 10^{-5}$	1.28	$1.50\ 10^{-4}$	3.51	$1.80\ 10^{-4}$

Table 2: Results for Cartesian grids, pointvalue discretization

of 240×240. Therefore we should compare these results to the entries for 128×128 and 256×256 in Table 2.

In a second example, we use a random mesh. The mesh generator ensure a certain regularity : the ratio of the circumcircle and the inner circle are controled. Here 3 levels are used, respectively with 12 526, 2 992, 706 nodes and 24 650, 5 782 and 1 310 triangles. The results are displayed in Table 3. The maximum radius of the triangles of the finest level is $\simeq 1.8\ 10^{-2} \simeq 1/55$ and $J_L \simeq 110^2$. From the approximation error point of view, we consider

Function	ϵ	μ	error (L^∞)
f_1	10^{-2}	16.03	$5.03\ 10^{-2}$
	10^{-3}	5.43	$1.76\ 10^{-3}$
	10^{-4}	1.50	$1.77\ 10^{-4}$
f_2	10^{-2}	4.22	$1.23\ 10^{-1}$
	10^{-3}	2.43	$2.29\ 10^{-3}$
	10^{-4}	1.17	$1.36\ 10^{-4}$

Table 3: Results for the coarsening procedure

the maximal radius and therefore we have to compare to a grid of 55×55. From the point of view of number of elements, we have to compare to a grid of 110×110. Therefore we should compare these results to the entries for 64×64 and 128×128 in Table 2.

These two examples, and many other that are not presented here indicate that the multiresolution analysis is stable. Furthermore, the MR scheme for the unstructured grid seems to be caparable in performance to corresponding Cartesian grids.

6.2 Cell-average discretization

In a first set of experiments, we have compared the compression and relative error for the functions f_1 and f_2 on $[-1, 1]^2$ our general agglomeration method to corresponding MR schemes for cell-average discretization on Cartesian grids. We start with a fine mesh which is generated from the triangulation of the previous example, by Finite element cells which are describe as choice (3) in section 5.2.1, therefore the number of cells is the same as the number of nodes in the triangulation, namely $J_L = 12\ 526$. It corresponds to $J_L \simeq 110^2$. The size of the maximal cell is of the order of $1/50$. Using 5 levels, the process of agglomeration generates successively 1 980, 322, 57 and 11 cells. As in the previous example, this should be compared with Cartesian grids between 64×64 to 128×128. The results are displayed in Table 4 (agglomeration procedure) and 5 (regular analysis). As we can see, the compression factors

Function	ϵ	μ	error (L^∞)	error (L^1)
f_1	10^{-2}	16.45	$7.75\ 10^{-2}$	$5.37\ 10^{-3}$
	10^{-3}	4.05	$3.06\ 10^{-3}$	$4.46\ 10^{-4}$
	10^{-4}	1.15	$2.98\ 10^{-4}$	$1.61\ 10^{-6}$
f_2	10^{-2}	2.85	$4.83\ 10^{-2}$	$2.56\ 10^{-3}$
	10^{-3}	1.71	$2.97\ 10^{-3}$	$3.42\ 10^{-4}$
	10^{-4}	1.02	$2.92\ 10^{-4}$	$3.1\ 10^{-6}$

Table 4: Results for the agglomeration procedure

Function		64×64			128×128		
	ϵ	μ	error (L^∞)	error (L^1)	μ	error (L^∞)	error (L^1)
f_1	10^{-2}	7.69	$8.9\ 10^{-3}$	$1.75\ 10^{-3}$	21.79	$6.57\ 10^{-3}$	$1.14\ 10^{-3}$
	10^{-3}	2.32	$1.11\ 10^{-3}$	$2.29\ 10^{-4}$	6.97	$8.99\ 10^{-4}$	$1.75\ 10^{-4}$
	10^{-4}	1.29	$9.20\ 10^{-5}$	$1.66\ 10^{-5}$	2.23	$1.09\ 10^{-4}$	$2.19\ 10^{-5}$
f_2	10^{-2}	2.45	$1.13\ 10^{-2}$	$1.81\ 10^{-3}$	4.74	$1.07\ 10^{-2}$	$1.49\ 10^{-3}$
	10^{-3}	1.38	$7.39\ 10^{-4}$	$0.95\ 10^{-4}$	3.00	$1.16\ 10^{-3}$	$1.92\ 10^{-4}$
	10^{-4}	1.09	$6.27\ 10^{-5}$	$0.92\ 10^{-5}$	1.52	$8.86\ 10^{-5}$	$1.12\ 10^{-5}$

Table 5: Results for cartesian grid, cell average discretization

are very similar for both method. The Figure 8 shows the multiresolution representation of f_2 (on a 512×512 mesh), and $\epsilon = 10^{-3}$. It should be red according to (B.10), Appendix B.

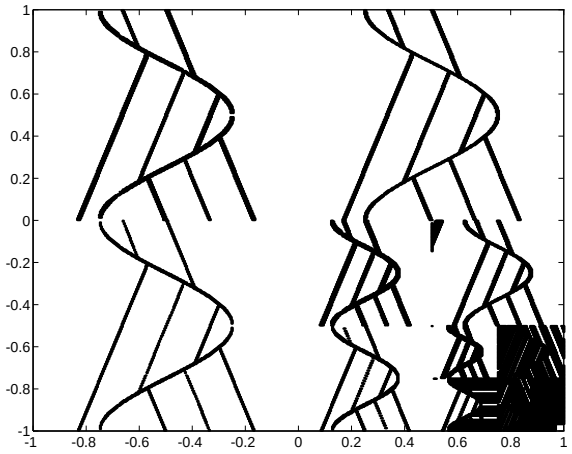


Figure 8: Multiresolution representation of f_2 on a 512×512 mesh

The encoding/decoding procedure, for the agglomeration procedure) is illustrated on Figure 9. We analyse the same function, on the same mesh, with only 3 levels this times. The fist line of figure (noted (a)) represent the effect of the decimation operator. One starts form the representation of f_2 on the fine level, then decimates and gets the figures medium-(a) and coarse-(a). The errors between two consecutive levels are plotted on the second line. For example, the first (b)-picture represents the (truncated) error between the (a)-fine plot and what is reconstructed from the (a)-medium plot. The third line represents what is obtained by reconstruction and correction with the truncated errors from the coarsest representation of f_2, plotted on coarse-(c) and coarse-(a).

In a second set of experiments, we show all the avantages of using an unstructured mesh : the geometrical flexibility. The same functions are analysed on a domain similar to a smiling face. Four levels are used. The mesh has 12 962 cells, the agglomerated meshes have 2 122, 369 and 68 cells. On Figure 10, we show the cells of the coarsest level. Some figures concerning f_2 are given. The Figure 11 show the encoded function on the first line. The coarsest level is plotted on the right, and the finest on the left. On the second line of Figure 11, we show the trunctated errors. Last, on Table 6, we give the compression factor and L^1 error for this particular domain.

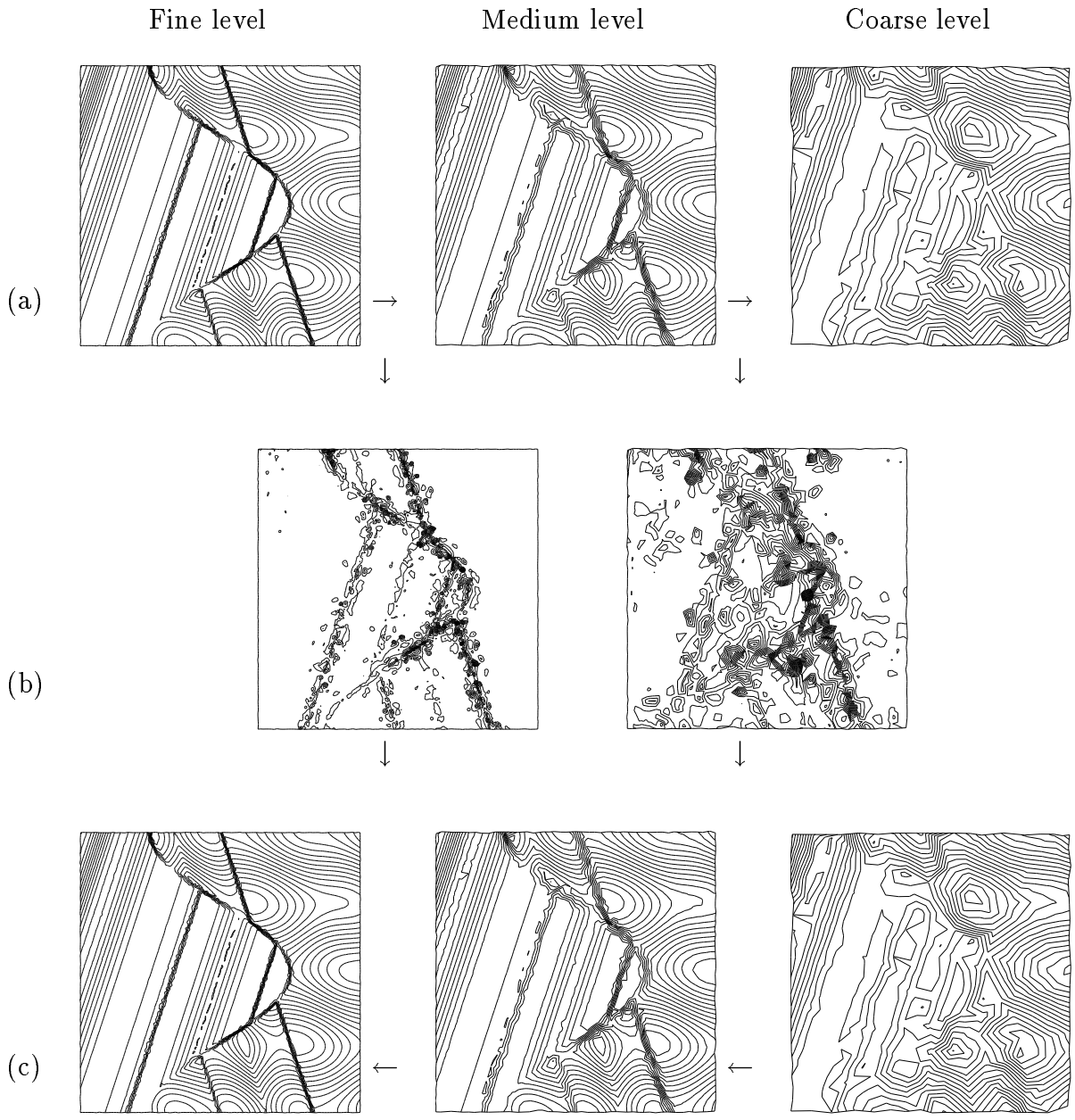


Figure 9: Encoding/Decoding procedure : (a) Decimation, (b) truncated errors, (a)+(b)=Decoding, (c) Encoding