Hamburger Beiträge
zur Angewandten Mathematik

Generalized Multiresolution Analysis on Unstructured Grids

Oliver Friedrich, Friederike Schröder-Pander,
Thomas Sonar

Reihe F
Computational Fluid Dynamics and Data Analysis 5
April 1998

Hamburger Beiträge zur Angewandten Mathematik

Reihe A	Preprints
Reihe B	Berichte
Reihe C	Mathematische Modelle und Simulation
Reihe D	Elektrische Netzwerke und Bauelemente
Reihe E	Scientific Computing
Reihe F	Computational Fluid Dynamics and Data Analysis

Generalized Multiresolution Analysis on Unstructured Grids

Oliver Friedrich, Friederike Schröder-Pander,
Thomas Sonar
Institut für Angewandte Mathematik
Universität Hamburg
Bundesstraße 55
D-20146 Hamburg

April 1, 1998

Abstract

Efficiency of high-order essentially non-oscillatory (ENO) approximations of conservation laws can be drastically improved if ideas of multiresolution analysis are taken into account. These methods of data compression not only reduce the necessary amount of discrete data but can also serve as tools in detecting local low-dimensional features in the numerical solution. We describe the mathematical background of the generalised multiresolution analysis as developed by Abgrall and Harten in [14], [15] and [3]. The functional analytic background is ultimately reduced to matrix-vector operations of linear algebra. We consider the example of interpolation on the line as well as the important case of multiresolution analysis of cell average data which is used in finite volume approximations.

Contents

1 Introduction

One of the most important tasks of Computational Fluid Dynamics (CFD) is the design of highly accurate, robust and efficient numerical methods for the simulation of compressible fluid flow. It is common knowledge in numerical analysis that the three properties: accuracy, robustness, and efficiency, are orthogonal concepts in almost all areas of algorithmic development and that problems due to the complementary nature of these properties can only be circumvented if additional information about the problem to be solved is available. A classical paradigm can be found in the well-known multigrid methods, see [6]. Here, accurate and robust finite element and finite difference methods for elliptic partial differential equations can be made efficient by analyzing the damping properties of linear systems solvers on a sequence of grids with decreasing resolution.
The most sophisticated numerical methods combining accuracy with robustness in the computation of compressible flow fields are finite volume approximations based on essentially non-oscillatory (ENO) recovery procedures, see [2], [8], [23]. While the choice of the type of the recovery function is responsible for high accuracy the choice of the stencil to compute the recovery function adds the required robustness. However, these methods are *a priori* not very efficient since all of the flow field is treated with the same expensive algorithm: stencil selection, ENO recovery, solution of local Riemann problems; but the full ENO algorithm is only required across and close to shocks and contact discontinuities which form lower dimensional manifolds in the solution. Would it be known in advance in which parts of the flow field the discontinuities are located and which parts are dominated by smoothly

varying flow variables, then in most of the flow field cheap non-ENO higher order recoveries could be used in order to increase efficiency.
From another point of view this kind of data analysis could be applied on a sequence of grids with decreasing resolution. It may then turn out that certain expensive operations (recovery) can be done on a coarse grid level and cheaply transfered (by interpolation) to the fine grid. This point of view corresponds to the usual framework of signal analysis. A signal (e.g. a two-dimensional picture) is analysed by decomposing it into different scale components (different frequencies, for example). If some of the scale coefficients fall below a certain tolerance they are neglected, which corresponds to data compression. The compressed (truncated) data on the coarsest grid can then be used to reconstruct the data on the finest grid within a prescribed tolerance.
In the framework of signal analysis the most recent prominent tool in data analysis are wavelets, see [5], [21], [22], [26]. A single function, the analysing wavelet, is transformed through dilation and translation and the data is interpreted in terms of these translated and dilated wavelet. Wavelets and the underlying principle of multiresolution analysis were certainly the inspiring source which initiated the research of Ami Harten and the creation of generalised multiresolution analysis (MRA), see [14], [15], [16]. In generalised MRA any sensible function can serve as analysing generalised wavelet and it is the freedom of this choice where the advantages as well as the disadvantages of Harten's approach can be seen. The main advantage is the possibility of tailoring MRA for practically significant problems. While the use of wavelets is still bound to rectangular domains and periodic boundary conditions the generalised multiresoltion approach is able to work on unstructured grids covering domains with arbitrary boundaries. The main disadvantage is certainly only temporarily important: at this stage of development the rigorous mathematical foundation of wavelets can not yet be carried over to the general case.
It is worth noting that there exists another road to generalized MRA, namely the concept developed by Dahmen et al. (see [4]) in which Haar wavelets are lifted by enlarging their support.
It is furthermore worth mentioning that the application of ideas of signal analysis in CFD is not new. In 1985 E.V. Vorozhtsov and N.N. Yanenko published a monograph in which methods for the localisation of singularities in numerical solutions of gas dynamics were discussed, see [25] for the english translation. We can do no better than to cite from their preface: *As a result of the numerical simulation of multidimensional gas dynamics problems on a computer, the output information is obtained in the form of immense arrays of numerical data. In this connection, there arises the problem of*

extracting the actually needed information from these arrays; in other words, it is necessary to solve the problem of information compression.

In this paper we pick up the ideas of generalised multiresolution analysis especially for the application in ENO approximations of hyperbolic conservation laws following Abgrall and Harten [3]. We reduce the operator-theoretic description of generalised MRA ultimately to the level of linear algebra. This reduction results in two true achievements: The basic ingredients of generalised MRA are most easily understood in terms of kernel and range of matrices and vectors, and, secondly, the construction of actual numerical algorithms is simplified in this context.

The conception of this paper is as follows. The next section contains a brief description of the role which is played by cell averages in the numerical solution of systems of hyperbolic conservation laws. This section gives the justification for concentrating on MRA for cell averages on unstructured grids. Section 3 includes the general framework of generalised MRA. In a first simple example in Section 3.3 we give life to the notions of MRA by applying it to interpolation on the unit interval. Section 4 is then concerned with the main body of this work: the description of MRA for cell averages on unstructured grids in the framework of linear algebra. It is here where reference is made to modern ENO schemes.

This paper is not intended to give a complete overview of generalised wavelets and multiresolution analysis but documents a certain point in the development of a multiresolution ENO code [12].

2 Finite volume methods for conservation laws

We consider hyperbolic systems of conservation laws

$$\partial_t \underline{u} + \sum_{i=1}^{2} \partial_{x_i} \underline{f}_i(\underline{u}) = \underline{0}$$

for functions $\mathbf{R}^+ \times \mathbf{R}^2 \ni (t, \underline{x}) \stackrel{\underline{u}}{\longmapsto} \underline{u}(t, \underline{x}) \in \mathcal{S} \subset \mathbf{R}^m$ in bounded domains $\Omega \subset \mathbf{R}^2$ which we assume to be polygonally bounded. The subset $\mathcal{S}$ is called state space and the non-linear functions $\underline{f}_i$, $i = 1, 2$, are refered to as flux vectors.

As is well known solution of such systems typically develop singularities within finite time, see [20]. Therefore, we introduce the notion of weak solutions. A mapping $\underline{u} \in [BV([0,\tilde{t}]; L^1 \cap L^\infty(\Omega)]^m$ is called weak solution, if

$$\frac{d}{dt}\int_\sigma \underline{u}\, d\underline{x} + \oint_{\partial\sigma} \left\{ \underline{f}_1(\underline{u})n_1 + \underline{f}_2(\underline{u})n_2 \right\} ds = \underline{0}$$

holds for all polygonal bounded subsets $\sigma \subset \mathbf{R}^2$. Here, $\underline{n} = (n_1, n_2)^T$ denotes the outer unit normal vector at $\partial\sigma$. We refer to such subsets σ as control volumes. Introducing the cell average operator

$$\mathfrak{A}(\sigma)\underline{u} := \frac{1}{|\sigma|}\int_\sigma \underline{u}\, d\underline{x}$$

results in the following trivial, but useful,

Remark 2.1 *Cell averages of weak solutions of systems of hyperbolic conservation laws satisfy the evolution equation*

$$\frac{d}{dt}\mathfrak{A}(\sigma)\underline{u}(t) = \frac{1}{|\sigma|}\oint_{\partial\sigma} \left\{ \underline{f}_1(\underline{u})n_1 + \underline{f}_2(\underline{u})n_2 \right\} ds. \tag{1}$$

Since cell averages play such a significant role for conservation laws one could try to design numerical methods to solve (1). The class of discretisations which directly treats (1) is usually called the class of finite volume approximations.

We introduce a conforming triangulation $\mathcal{T}^h$ on Ω and denote a generic triangle by $T \in \mathcal{T}^h$. Each triangle is the convex hull of three nodes of the triangulation. The set

$$K_{h,i} := \{T \in \mathcal{T}^h \mid \text{node } i \text{ is vertex of } T\}$$

contains all triangles sharing node i while

$$E_{h,i} := \{e_{T,k} \mid T \in \mathcal{T}^h, k \in \{1,2,3\}, \text{node } i \text{ is vertex point of } e_{T,k}\}$$

is the set of all edges sharing node i where $e_{T,k}, k = 1,2,3$, denotes the three edges of a triangle T. Connecting the barycentres of each $T \in K_{h,i}$ with the midpoints of the edges $e_{T,k} \subset E_{h,i}$ results in a further tesselation of Ω which consists of cells (boxes) σ_i around nodes i. These cells are the discrete control volumes. The boundary between two such cells σ_i and σ_j is shown in figure 1 together with the two triangles building the edge between points i and j. Using the notation $N(i) := \{j \in \mathbf{N} \mid \sigma_i \cap \sigma_j$ is part of the boundary of $\sigma_i\}$, the system (1) can be re-written on cell σ_i in the form $\frac{d}{dt}\mathfrak{A}(\sigma_i)\underline{u}(t) = \frac{1}{|\sigma_i|}\sum_{j\in N(i)}\sum_{k=1}^2 \int_{l_{ij}^k} \{\underline{f}_1(\underline{u})n_{ij,1}^k + \underline{f}_2(\underline{u})n_{ij,2}^k\}\, ds$. Applying an n_G-point Gaussian quadrature rule on l_{ij}^k results in an error

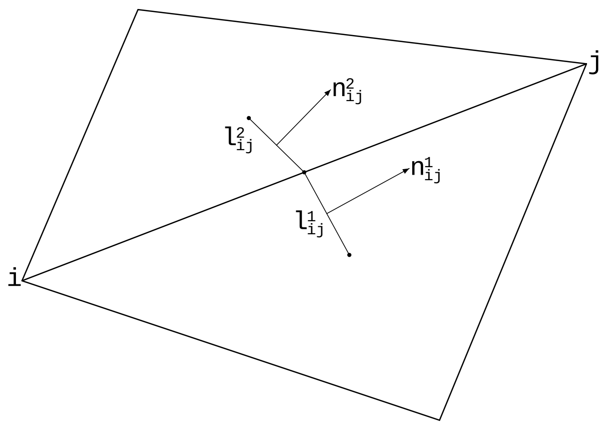


Figure 1: Geometry between cells σ_i and σ_j

of order h^{2n_G}. If we replace the integrand by a consistent numerical flux function $\mathcal{S} \times \mathcal{S} \times \mathbf{R}^2 \ni (\underline{u}, \underline{v}, \underline{n}) \overset{H}{\longmapsto} \underline{H}(\underline{u}, \underline{v}; \underline{n}) \in \mathbf{R}^m$, see [17], we arrive at

$$\frac{d}{dt}\mathfrak{A}(\sigma_i)u(t) =$$
$$-\frac{1}{|\sigma_i|} \sum_{j \in N(i)} \sum_{k=1}^{2} \frac{|l_{ij}^k|}{2} \left\{ \sum_{\nu=1}^{n_G} \omega_\nu^k \underline{H}\left(\underline{u}(\underline{x}_{ij}^k(s_\nu), t), \underline{u}(\underline{x}_{ij}^k(s_\nu), t); \underline{n}_{ij}^k\right) + \mathcal{O}\left(h^{2n_G}\right) \right\}$$

without introducing a further error term.
In order to construct a numerical method we switch completely to cell averages. We call the method defined by the systems of ordinary differential equations

$$\begin{aligned} \frac{d}{dt}\underline{\overline{u}}_i &= -\frac{1}{|\sigma_i|} \sum_{j \in N(i)} \sum_{k=1}^{2} \frac{|l_{ij}^k|}{2} \left\{ \sum_{\nu=1}^{n_G} \omega_\nu^k \underline{H}\left(\underline{\overline{u}}_i(\underline{x}_{ij}^k(s_\nu), t), \underline{\overline{u}}_j(\underline{x}_{ij}^k(s_\nu), t); \underline{n}_{ij}^k\right) \right\} \\ \underline{\overline{u}}_i(0) &= \mathfrak{A}(\sigma_i)\underline{u}(0) \end{aligned}$$

a basic finite volume method. However, due to the weak approximation properties of the cell average operator, namely, for fixed t, $\left(\underline{u} - \mathfrak{A}(\sigma_i)\underline{u}\right)(\underline{x}, t) = \mathcal{O}(h)$, $\quad \underline{x} \in \partial\sigma_i$, the formal order of accuracy of the method is bounded by h. A recovery procedure is needed to increase the spatial accuracy. If we could find a recovery operator $\mathfrak{R}$ which assigns to cell σ_i at fixed time a function $\underline{\pi}_i := \mathfrak{R}\underline{\overline{u}}$ constructed from the cell averages $\underline{\overline{u}}_i$ and additional neighbouring cells satisfying the recovery condition $\mathfrak{A}(\sigma_i)\underline{\pi}_i(t) = \underline{\overline{u}}_i(t)$, as well as the accuracy condition $(\underline{u} - \underline{\pi}_i)(\underline{x}, t) = \mathcal{O}\left(h^r\right), \quad r \geq 1$, then the finite volume method

$$
\begin{aligned}
\frac{d}{dt}\underline{\overline{u}}_i &= -\frac{1}{|\sigma_i|}\sum_{j\in N(i)}\sum_{k=1}^{2}\frac{|l_{ij}^k|}{2}\left\{\sum_{\nu=1}^{n_G}\omega_\nu^k \underline{H}\left(\underline{\pi}_i(\underline{x}_{ij}^k(s_\nu),t),\underline{\pi}_j(\underline{x}_{ij}^k(s_\nu),t);\underline{n}_{ij}^k\right)\right\}\\
\underline{\overline{u}}_i(0) &= \mathfrak{A}(\sigma_i)\underline{u}(0)
\end{aligned}
$$

can be shown to be formally r-th order accurate, see [23].
We do not go into details concerning the actual computation of the recovery functions $\underline{\pi}_i$ but refer the reader to [23]. A very robust and accurate algorithm was developed by Friedrich in [13] in the form of a weighted ENO (WENO) approximation on unstructured grids.

3 The mathematical background of multiresolution analysis

3.1 Restriction, prolongation and recovery operators

We start with a general framework.

Definition 3.1 Let $\{\mathfrak{D}_k\}_{k=0}^{\infty}$ be a sequence of linear operators

$$\mathfrak{D}_k : V \to V^k$$

between two Banach spaces V and V^k. The dimension of the spaces V^k is denoted by

$$J_k := \mathrm{dim} V^k,$$

although this may be infinity. The sequence is called **nested**, if

1. $\mathfrak{D}_k$ is surjective for all k,
2. $\mathfrak{D}_k f = 0 \quad \Longrightarrow \quad \mathfrak{D}_{k-1} f = 0$ for every $f \in V$.

In our context the spaces V^k will be associated with grids of different resolution. It is always desirable to think of V^k being a finer grid than V^{k-1}, i.e.

$$\mathrm{dim} V^{k-1} < \mathrm{dim} V^k.$$

Definition 3.2 A linear, surjective mapping

$$D_k^{k-1} : V^k \to V^{k-1}$$

is called **restriction operator**, if

$$\forall v = \mathfrak{D}_k f \in V^k : \quad D_k^{k-1} v = \mathfrak{D}_{k-1} f.$$

Since an element $f \in V$ is inherent in the previous definition one has to ask wether the definition itself is proper. This is the contents of the following lemma.

Lemma 3.1 *The definition of a restriction operator is independent of the particular $f \in V$, i.e. restriction operators are well defined.*

Proof: Due to the surjectivity of $\mathfrak{D}_k$ there is at least one element $f \in V$ such that $\mathfrak{D}_k f = v$. Let $f_1 \neq f_2$ be elements of V with $\mathfrak{D}_k f_1 = v = \mathfrak{D}_k f_2$. Then, using the linearity of $\mathfrak{D}_k$, it follows

$$0 = \mathfrak{D}_k f_1 - \mathfrak{D}_k f_2 = \mathfrak{D}_k (f_1 - f_2).$$

Exploiting the second property of a nested sequence of operators in definition 3.1 it then follows

$$0 = \mathfrak{D}_{k-1}(f_1 - f_2) = \mathfrak{D}_{k-1} f_1 - \mathfrak{D}_{k-1} f_2,$$

which proves the claim. ■

We now show that the restriction indeed is possible without explicit knowledge of f.

Lemma 3.2 *Let $L \in \mathbf{N}$ and $v^L \in V^L$ be given. Suppose $\{v^k\}_{k=0}^{L-1}$ is computed by repeated restriction*

$$v^{k-1} = D_k^{k-1} v^k, \quad k = L, \cdots, 1,$$

then for any $f \in V$ for which $v^L = \mathfrak{D}_L f$ holds, we have

$$v^k = \mathfrak{D}_k f, \quad k = L, \cdots, 1.$$

Proof: By the definition of the restriction operator it follows that

$$D_k^{k-1}(\mathfrak{D}_k f) = \mathfrak{D}_{k-1} f$$

for any $f \in V$. ■

To each of the operators $\mathfrak{D}_k$ there exists a right inverse operator which will be of use further on.

Lemma 3.3 *For every $\mathfrak{D}_k, k = 0, \cdots, \infty$, there exists a right inverse operator*

$$\mathfrak{R}_k : V^k \to V,$$

i.e. it holds

$$\mathfrak{D}_k \mathfrak{R}_k = \mathfrak{i}_k,$$

where $\mathfrak{i}_k$ denotes the identity in V_k.

Proof: Since $\mathfrak{D}_k$ is surjective from V to V^k it holds

$$\mathfrak{D}_k(\mathfrak{D}_k^{-1}(\tilde{V}^k)) = \tilde{V}^k$$

for all subsets $\tilde{V}^k \subset V^k$, where $\mathfrak{D}_k^{-1}(\tilde{V}^k)$ is the subset of V defined by the property $\mathfrak{D}_k f \in \tilde{V}^k$ for $f \in \mathfrak{D}_k^{-1}(\tilde{V}^k)$. ■

Definition 3.3 The right inverse operator $\mathfrak{R}_k$ of $\mathfrak{D}_k$ is called **recovery operator** in V^k. This is meant in the sense that $f \in V$ can be recovered from data in V^k.

Suppose we are given data $v^L \in V^L$ as well as $v^{k-1} = D_k^{k-1} v^k, k = L, \ldots, 1$, by repeated restriction. Then $\mathfrak{R}_{k-1} v^{k-1}$ is the recovery of some $f \in V$ from data in V^{k-1} and thus, since $v^k = \mathfrak{D}_k f$, we get an approximation to v^k by

$$v^k \approx \mathfrak{D}_k(\mathfrak{R}_{k-1} v^{k-1}).$$

Definition 3.4 Operators

$$P_{k-1}^k : V^{k-1} \to V^k$$

defined by

$$P_{k-1}^k := \mathfrak{D}_k \mathfrak{R}_{k-1}$$

are called **prolongation operators**.

Lemma 3.4 *The prolongation operator P_{k-1}^k is a right-inverse of the restriction operator D_k^{k-1}, i.e. it holds*

$$D_k^{k-1} P_{k-1}^k = \mathfrak{i}_{k-1}.$$

Proof: Since, by definition of the restriction operator,

$$D_k^{k-1}(\mathfrak{D}_k f) = \mathfrak{D}_{k-1} f$$

for every $f \in V$, this is also valid for $f := \mathfrak{R}_{k-1} v^{k-1}$. Thus,

$$D_k^{k-1}(\mathfrak{D}_k \mathfrak{R}_{k-1} v^{k-1}) = \mathfrak{D}_{k-1} \mathfrak{R}_{k-1} v^{k-1}$$

holds. Now, since $P_{k-1}^k v^{k-1} = \mathfrak{D}_k \mathfrak{R}_{k-1} v^{k-1}$, this gives

$$D_k^{k-1} P_{k-1}^k v^{k-1} = \mathfrak{D}_{k-1} \mathfrak{R}_{k-1} v^{k-1} = \mathfrak{i}_{k-1} v^{k-1} = v^{k-1},$$

which proves the lemma. ■

3.2 Prediction error and scale coefficients

Definition 3.5 The function e^k, defined by

$$e^k := v^k - P_{k-1}^k v^{k-1} = (\mathfrak{i}_k - P_{k-1}^k D_k^{k-1}) v^k$$

is called **prediction error**.

The following result is a trivial remark.

Lemma 3.5 *The prediction error is element of the kernel of D_k^{k-1}, i.e.*

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

Lemma 3.6 *It holds*

$$\dim \ker(D_k^{k-1}) = J_k - J_{k-1}.$$

Proof: Since D_k^{k-1} is surjective $\mathrm{ran}(D_k^{k-1}) = V^{k-1}$ is its range and the lemma is a reformulation of the kernel theorem

$$\dim V^k = \dim \mathrm{ran}(D_k^{k-1}) + \dim \ker(D_k^{k-1})$$

for linear mappings. ■

Due to the previous lemma, the prediction error is representable by only $J_k - J_{k-1}$ components.

Definition 3.6 Let $\{\mu_j^k\}_{j=1}^{J_k - J_{k-1}}$ denote any basis of $\ker(D_k^{k-1})$, i.e.

$$\ker(D_k^{k-1}) = \mathrm{span}\{\mu_j^k\}_{j=1}^{J_k - J_{k-1}}.$$

In the representation

$$e^k = \sum_{j=1}^{J_k - J_{k-1}} d_j^k \mu_j^k =: E_k d^k$$

of the prediction error the coefficients $d^k := (d_1^k, \dots, d_{J_k - J_{k-1}}^k)$ are called **scale coefficients**. The operator G_k, defined by

$$d^k =: G_k e^k$$

is called **assignement operator**.

An immediate consequence of the previous definition is

Lemma 3.7 *For any* $e^k \in \ker(D_k^{k-1})$ *it holds*

$$E_k G_k e^k = e^k,$$

i.e.

$$E_k G_k = \mathfrak{i}_{\ker(D_k^{k-1})}.$$

Lemma 3.8 *There is a one-to-one correspondence between* v^k *and* $\{d^k, v^{k-1}\}$.

Proof: Compute v^{k-1} and d^k by

$$\left\{ \begin{array}{lcl} v^{k-1} & = & D_k^{k-1} v^k \\ d^k & = & G_k(\mathfrak{i}_k - P_{k-1}^k D_k^{k-1}) v^k. \end{array} \right.$$

Then recover v^k by

$$\begin{array}{rcl} P_{k-1}^k v^{k-1} + E_k d^k & = & P_{k-1}^k D_k^{k-1} v^k + E_k G_k (\mathfrak{i}_k - P_{k-1}^k D_k^{k-1}) v^k \\ & = & P_{k-1}^k v^{k-1} + (\mathfrak{i}_k - P_{k-1}^k D_k^{k-1}) v^k \\ & = & v^k. \end{array}$$

■

Remark 3.1 Applying the above lemma for $k = L, \ldots, 1$, results in a one-to-one correspondence between v^L and $\{d^L, \ldots, d^1, v^0\}$ which we denote by

$$v^L \leftrightarrow \{d^L, \ldots, d^1, v^0\}.$$

Definition 3.7 The representation

$$\hat{v}_M := \{d^L, \ldots, d^1, v^0\} \leftrightarrow v^L$$

is called **multiresolution representation** of $v^L \in V^L$.

We are now ready to define the generalized multiresolution transform.

Definition 3.8 The **direct multiresolution transform** of $v^L \in V^L$, formally represented by an operator M in the form

$$\hat{v}_M = M v^L,$$

is given by the algorithm: For all $k = L, \dots, 1$ compute

$$\begin{aligned} v^{k-1} &= D_k^{k-1} v^k \\ d^k &= G_k(\mathfrak{i}_k - P_{k-1}^k D_k^{k-1}) v^k. \end{aligned}$$

The **inverse multiresolution transform**, formally written as

$$v^L = M^{-1} \hat{v}_M,$$

is given by: For all $k = 1, \dots, L$ compute

$$v^k = P_{k-1}^k v^{k-1} + E_k d^k.$$

3.3 A first example

To give life to the formal and dry theoretical backgound of multiresolution analysis we consider a simple example concerning interpolation on a grid in one space dimension. On the finest level L define an integer value N and compute the grid spacing as $\Delta x := 2^{-N}$. As a grid on the k-th level we consider the set

$$\mathbb{G}_k := \{x_i^k \in \mathbf{R} \mid x_i^k := i\Delta y, i = 0, \dots, 2^{k-L}2^N, \Delta y = 2^{L-k}\Delta x\}$$

and the next coarser level

$$\mathbb{G}_{k-1} := \{x_i^{k-1} \in \mathbf{R} \mid x_i^{k-1} := x_{2i}^k, i = 0, \dots, 2^{(k-1-L)}2^N\}$$

Given a function $f \in C([0,1]; \mathbf{R}) =: V$ we define the operator $\mathfrak{D}_k$ by means of

$$V \ni f \xmapsto{\mathfrak{D}_k} \mathfrak{D}_k f =: \underline{v}^k := (v_0^k, \dots, v_{2^{(k-L)}2^N}^k) \in \mathbf{R}^{2^{(k-L)}2^N+1} =: V^k,$$

resulting in the values of f at the discrete points of $\mathbb{G}_k$. Note that the discretisation is nested since $\mathfrak{D}_k$ is certainly surjective and $\mathfrak{D}_k f = 0$ implies $\mathfrak{D}_{k-1} f = 0$.
The restriction operator D_k^{k-1} is the simple 'forget-odd-numbered-points' operator defined by

$$\begin{aligned} D_k^{k-1} \underline{v}^k := (v_0^{k-1}, \dots, v_{2^{(k-1-L)}2^N}^{k-1}) &= (v_0^k, v_2^k, v_4^k, \dots v_{2^{(k-L)}2^N-2}^k, v_{2^{(k-L)}2^N}^k) \\ &=: \underline{v}^{k-1} \in \mathbf{R}^{2^{(k-1-L)}2^N+1} =: V^{k-1}. \end{aligned}$$

Thus,

$$J_k = \dim V^k = 2\dim V^{k-1} - 1 = 2J_{k-1} - 1.$$

Note that the restriction operator can be conveniently expressed as a matrix $\underline{\underline{D}}_k^{k-1} \in \mathbf{R}^{(2^{k-1-L}2^N+1)\times(2^{k-L}2^N+1)}$ with $2^{k-1-L}2^N+1$ rows and $2^{k-L}2^N+1$ columns:

$$\begin{aligned}\underline{\underline{D}}_k^{k-1}\underline{v}^k &:= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 1 \end{bmatrix} \begin{bmatrix} v_0^k \\ \vdots \\ v_{2^{(k-L)}2^N}^k \end{bmatrix} \\ &= \begin{bmatrix} v_0^{k-1} \\ \vdots \\ v_{2^{(k-1-L)}2^N}^{k-1} \end{bmatrix} = \underline{v}^{k-1}.\end{aligned}$$

To stay precise we shall talk about the **representation** of the operator D_k^{k-1} if we want to emphasize the matrix structure of $\underline{\underline{D}}_k^{k-1}$.
Let us now discuss a suitable prolongation operator P_{k-1}^k. Although we already know that a right-inverse to the discretisation $\mathfrak{D}_k$ must exist (Lemma 3.3) in almost all cases of interest we shall not be able to compute this operator exactly. In our case $\mathfrak{D}_k$ maps a function space onto a finite dimensional vector space and thus arbitrary many information about the mapped function f is lost. However, we can use this freedom to design prolongation operators. According to Lemma 3.4 the prolongation operator has to be a right-inverse to the restriction. In fact, a matrix representation of P_{k-1}^k is given by

$$\underline{\underline{P}}_{k-1}^k := \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ * & * & * & * & \cdots & * \\ 0 & 1 & 0 & 0 & \cdots & 0 \\ * & * & * & * & \cdots & * \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ * & * & * & * & \cdots & * \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 \end{bmatrix} \in \mathbf{R}^{(2^{(k-L)}2^N+1)\times(2^{k-1-L}2^N+1)},$$

where the $*$'s denote any possible entry. The relation $\underline{\underline{D}}_k^{k-1}\underline{\underline{P}}_{k-1}^k = \mathfrak{i}_{k-1}$ is easily verified.
We choose piecewise linear interpolation to act as recovery operator. Thus,

$$\mathfrak{R}_{k-1}\underline{v}^{k-1} \in \mathcal{S}_1(\mathbb{G}_{k-1})$$

where $\mathcal{S}_1(\mathbb{G}_{k-1})$ denotes the spline space of piecewise linear continuous functions on $\mathbb{G}_k$, i.e.

$$\mathfrak{R}_{k-1}\underline{v}^{k-1}\big|_{[x_i^{k-1},x_{i+1}^{k-1}]} \in \Pi_1(\mathbf{R};\mathbf{R})$$

satisfying the interpolation conditions

$$\mathfrak{R}_{k-1}\underline{v}^{k-1}(x_i^{k-1}) = v_i^{k-1}.$$

For $\lambda \in [0,1]$ this amounts to

$$\mathfrak{R}_{k-1}\underline{v}^{k-1}\big|_{[x_i^{k-1},x_{i+1}^{k-1}]} = v_i^{k-1} + \lambda\left(v_{i+1}^{k-1} - v_i^{k-1}\right) = v_{2i}^k + \lambda\left(v_{2i+2}^k - v_{2i}^k\right),$$

and hence

$$\mathfrak{D}_k\mathfrak{R}_{k-1}\underline{v}^{k-1}\big|_{(x_i^{k-1},x_{i+1}^{k-1})} = \frac{1}{2}\left(v_i^{k-1} + v_{i+1}^{k-1}\right) = \frac{1}{2}\left(v_{2i}^k + v_{2i+2}^k\right)$$

amounts to evaluate the linear interpolant between the nodes of $\mathbb{G}_k$. Therefore, it follows

$$\begin{aligned}
&\mathfrak{D}_k\mathfrak{R}_{k-1}\underline{v}^{k-1} \\
&= \left(v_0^{k-1}, \frac{1}{2}(v_0^{k-1} + v_1^{k-1}), v_1^{k-1}, \frac{1}{2}(v_1^{k-1} + v_2^{k-1}), \ldots, v_{2^{(k-1-L)}2^N}^{k-1}\right) \\
&= \left(v_0^k, \frac{1}{2}(v_0^k + v_2^k), v_2^k, \frac{1}{2}(v_2^k + v_4^k), v_4^k, \ldots, v_{2^{(k-L)}2^N}^k\right).
\end{aligned}$$

Now this choice of recovery operator determines the prolongation and we have

$$\underline{\underline{P}}_{k-1}^k := \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & \cdots & 0 \\
\frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & \cdots & 0 \\
0 & 1 & 0 & 0 & 0 & \cdots & 0 \\
0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & \cdots & 0 \\
0 & 0 & 1 & 0 & 0 & \cdots & 0 \\
0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & 0 & 0 & \cdots & 1
\end{bmatrix} \in \mathbf{R}^{(2^{(k-L)}2^N+1)\times(2^{k-1-L}2^N+1)}.$$

Let us examine the kernel of the restriction operator D_k^{k-1}. We already know that dim ker$D_k^{k-1} = J_k - J_{k-1} = J_{k-1} - 1 = \frac{1}{2}(J_k - 1)$. Due to the representation of D_k^{k-1} as $\underline{\underline{D}}_k^{k-1}$ it is immediately seen that a basis of the kernel is given by the vectors

$$\ker\underline{\underline{D}}_k^{k-1} = \text{span}\{\underline{e}_{2j} =: \underline{\mu}_j^k\}, \quad j = 1, \ldots, 2^{(k-1-L)}2^N,$$

where the unit vectors $\underline{e}_{2j}$ are given by

$$\underline{\mu}_j^k = \underline{e}_{2j} := (\underbrace{0}_{\uparrow\atop 1}, \ldots, 0, \underbrace{1}_{\uparrow\atop 2j}, 0, \ldots, 0) \in \mathbf{R}^{2^{(k-L)}2^N+1}.$$

If we now look at the representation of the error e^k in terms of

$$\underline{e}^k := (\underline{v}^k - \underline{\underline{P}}^k_{k-1}\underline{v}^{k-1}) = (\underline{\underline{E}} - \underline{\underline{P}}^k_{k-1}\underline{\underline{D}}^{k-1}_k)\underline{v}^k$$

where $\underline{\underline{E}}$ denotes the unit matrix of dimension $\dim \underline{v}^k$ then it follows

$$\begin{aligned}\underline{\underline{E}} - \underline{\underline{P}}^k_{k-1}\underline{\underline{D}}^{k-1}_k \quad &= \quad \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ -\frac{1}{2} & 1 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & -\frac{1}{2} & 1 & -\frac{1}{2} & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{2} & 1 & -\frac{1}{2} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \end{bmatrix} \\ &\in \quad \mathbf{R}^{(2^{(k-L)}2^N+1)\times(2^{(k-L)}2^N+1)}\end{aligned}$$

which nicely reflects the structure of the linear interpolation used as recovery operator. However, we already know that the error is representable in terms of the $J_k - J_{k-1} = \frac{1}{2}(\dim V^k - 1)$ elements $\underline{\mu}^k_j$ spanning the kernel of D^{k-1}_k and this fact is also nicely seen in the structure of the above matrix.
The error is then given as the vector $\underline{e}^k = (e^k_i)_{i=1,\dots,2^{(k-L)}2^N+1}$ with components

$$\begin{aligned} e^k_{2j+1} \quad &= \quad 0, \quad j = 0, \dots, 2^{(k-1-L)}2^N, \\ e^k_{2j} \quad &= \quad -\frac{1}{2}v^k_{2j-1} + v^k_{2j} - \frac{1}{2}v^k_{2j+1}, \quad j = 1, \dots, 2^{(k-1-L)}2^N. \end{aligned}$$

Looking at the scale coefficients d^k_j in the representation

$$\underline{e}^k = \sum_{j=1}^{2^{(k-1-L)}2^N} d^k_j \underline{\mu}^k_j$$

shows that

$$d^k_j \quad = \quad e^k_{2j}, \quad j = 1, \dots, 2^{(k-1-L)}2^N.$$

This, in turn, enables us to compute the operators G_k and E_k through

$$d^k = G_k e^k$$

and

$$e^k = E_k d^k.$$

In fact, the representation of E_k is readily seen to be

$$\underline{\underline{E}}_k := \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \end{bmatrix} \in \mathbf{R}^{(2^{(k-L)}2^N+1)\times(2^{(k-1-L)}2^N+1)}.$$

Through the relation

$$E_k G_k = \mathrm{i}_{\ker(D_k^{k-1})}$$

it follows that a possible representation of G_k is given by

$$\underline{\underline{G}}_k := \begin{bmatrix} * & 0 & * & 0 & * & 0 & * & 0 & \cdots & 0 & * \\ * & 1 & * & 0 & * & 0 & * & 0 & \cdots & 0 & * \\ * & 0 & * & 0 & * & 0 & * & 0 & \cdots & 0 & * \\ * & 0 & * & 1 & * & 0 & * & 0 & \cdots & 0 & * \\ * & 0 & * & 0 & * & 0 & * & 0 & \cdots & 0 & * \\ * & 0 & * & 0 & * & 1 & * & 0 & \cdots & 0 & * \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ * & 0 & * & 0 & * & 0 & * & 0 & \cdots & 1 & * \\ * & 0 & * & 0 & * & 0 & * & 0 & \cdots & 0 & * \end{bmatrix} \in \mathbf{R}^{(2^{(k-1-L)}2^N+1)\times(2^{(k-L)}2^N+1)},$$

the $*$'s again denoting arbitrary entries.

In computing the representations of the operators G_k and E_k we not only have completed the description of the mechanisms of multiresolution analysis applied to a simple interpolation problem, but also added a new component to the representation as given by Harten and Abgrall, namely the reduction of multiresolution analysis to linear algebra. We give now a rigorous treatment of multiresolution analysis of cell averages in multiple space dimensions. As in the case of the simple interpolation problem we shall reduce everything to problems of linear algebra.

4 Multiresolution of cell averages on unstructured grids

4.1 Volume agglomeration and multiresolution

We consider a tesselation Σ^k of a bounded domain $\Omega \subset \mathbf{R}^2$ where we assume the elements $\sigma_i^k \in \Sigma$ to be bounded by polygons and furthermore that

$$\Omega = \bigcup_{i=1}^{N_k} \sigma_i^k,$$

i.e. $\partial\Omega$ is also assumed to be polygonal. We call the elements σ_i^k the cells of the tesselation Σ^k.
On each of the cells we consider linear functionals

$$C(\Omega; \mathbf{R}) \ni u \overset{\mathfrak{A}(\sigma_i^k)}{\longmapsto} \mathfrak{A}(\sigma_i^k)u := \frac{1}{|\sigma_i^k|} \int_{\sigma_i^k} u \, d\underline{x} \in \mathbf{R}$$

which are referred to as cell averages. A sequence of tesselations $(\Sigma^k)_{k=L,\dots,0}$ is defined by volume agglomeration following the algorithm of Lallemand [19]. However, while the original Lallemand algorithm leads to skewed agglomerated cells in general, we use a superior modification due to Friedrich [12]. This algorithm can be described in the following form.

1. Start with the finest tesselation Σ^L containing N_L cells by defining $k = L$.

2. For $k = L, \dots, 0$

   (a) As long as agglomeration is possible Do

      i. Select a cell Σ_i^k:

         A. If it is the very first selection at all Then
         consider only cells from the boundary of the domain and choose one of them.

         B. Else consider all the neighbouring cells of all agglomerated cells that have not been agglomerated yet. If there exist cells at the boundary of the domain, then consider any cell from the boundary, otherwise any other considered cell.

      ii. Collect all the neighbouring cells of σ_i^k that have not been agglomerated yet.

iii. Consider all combinations of cell σ_i^k with one, two, and three neighbouring cells, respectively. Thus, we allow agglomeration of two, three, and four cells, respectively. For each of these combinations compute a quality measure for the resulting agglomerated cell and choose the combination with the largest measure. The quality Q of a cell is measured by

$$Q := \frac{\text{area(cell)}}{(\text{circumference(cell)}^2)}$$

(b) <u>If</u> there exists any cell σ_i^k that was not agglomerated with other (fine) cells, <u>Then</u>

i. Collect all agglomerated, i. e. coarse, neighbouring cells.

ii. Consider all combinations with σ_i^k and one coarse neighbouring cell.

iii. Compute the above defined quality measure for all these combinations.

iv. Choose the combination with the largest measure and agglomerate both cells.

We end up with the sequence $(\Sigma^k)_{k=L,\dots,0}$ in which Σ^L is the finest tesselation and Σ^0 the coarsest, i.e. $N_{k-1} < N_k, k = 1, \dots, L$. We now localise our point of view. The multiresolution analysis on the whole tesselation can be entirely described if we look on the local problem of q cells $\sigma_{i_j}^k \in \Sigma^k, j = 1, \dots, q$, being agglomerated into one large cell $\sigma_\ell^{k-1} \subset \Sigma^{k-1}$. Thus, we concentrate on this local problem.

4.2 Restriction and prolongation

The discretisation $\mathfrak{D}_k$ on the q cells $\sigma_{i_j}^k, j = 1, \dots, q$, is described by

$$\mathfrak{D}_k u := (\mathfrak{A}(\sigma_{i_1}^k)u, \dots, \mathfrak{A}(\sigma_{i_q}^k)u) =: \underline{v}^k \in \mathbf{R}^q =: V^k.$$

Lemma 4.1 *The representation $\underline{D}_k^{k-1} \in \mathbf{R}^q$ of the local restriction operator D_k^{k-1} is defined by*

$$\begin{aligned}\underline{D}_k^{k-1}\mathfrak{D}_k u &= \underline{D}_k^{k-1}\underline{v}^k := \frac{1}{|\sigma_\ell^{k-1}|}\sum_{j=1}^{q} |\sigma_{i_j}^k|\mathfrak{A}(\sigma_{i_j}^k)u = \mathfrak{A}(\sigma_\ell^{k-1})u \\ &=: \underline{v}^{k-1} \in \mathbf{R} =: V^{k-1}\end{aligned}$$

and thus representable as

$$\underline{D}_k^{k-1} = \frac{1}{|\sigma_\ell^{k-1}|}\left(|\sigma_{i_1}^k|, |\sigma_{i_2}^k|, |\sigma_{i_3}^k|, \dots, |\sigma_{i_q}^k|\right) \in \mathbf{R}^q.$$

Proof: The restriction operator D_k^{k-1} is derived from the observation

$$\begin{aligned}
\mathfrak{A}(\sigma_\ell^{k-1})u &= \frac{1}{|\sigma_\ell^{k-1}|}\int_{\sigma_\ell^{k-1}} u\, d\underline{x} = \frac{1}{|\sigma_\ell^{k-1}|}\sum_{j=1}^{q}\int_{\sigma_{i_j}^k} u\, d\underline{x} \\
&= \frac{1}{|\sigma_\ell^{k-1}|}\sum_{j=1}^{q}|\sigma_{i_j}^k|\frac{1}{|\sigma_{i_j}^k|}\int_{\sigma_{i_j}^k} u\, d\underline{x} = \frac{1}{|\sigma_\ell^{k-1}|}\sum_{j=1}^{q}|\sigma_{i_j}^k|\mathfrak{A}(\sigma_{i_j}^k)u,
\end{aligned}$$

which concludes the proof. ■

To determine the elements in the kernel of D_k^{k-1} all vectors have to be considered which are orthogonal to $\underline{D}_k^{k-1}$, i.e. for which

$$\underline{D}_k^{k-1}\underline{v}^k \stackrel{!}{=} 0$$

holds. We already know the dimension of the kernel, namely

$$\dim\ker(D_k^{k-1}) = J_k - J_{k-1} = q - 1.$$

To give a complete description of the kernel we introduce the notion of a basis matrix.

Definition 4.1 Let $\{\underline{\mu}_j^k\}_{j=1,\dots,q-1}$ be a basis of $\ker(\underline{D}_k^{k-1})$ and

$$\underline{\sigma}^k := \begin{pmatrix} |\sigma_{i_1}^k| \\ \vdots \\ |\sigma_{i_q}^k| \end{pmatrix}$$

the vector consisting of the areas of the q cells agglomerated into one on level $k-1$. The matrix $\underline{\underline{M}}_j^k \in \mathbf{R}^{q\times q}$ is called the j-th basis matrix, if

$$\underline{\mu}_j^k = \underline{\underline{M}}_j^k\underline{\sigma}^k$$

holds for $j = 1, \dots, q-1$.

The construction of basis matrices turns out to be very simple. The antisymmetric matrix

$$\underline{\underline{M}}_1^k := \begin{bmatrix} 0 & 1 & \cdots & 1 \\ -1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 1 \\ -1 & \cdots & -1 & 0 \end{bmatrix}$$

is easily seen to be a basis matrix which we call generating basis matrix. The the following Lemma is easily obtained.

Lemma 4.2 *The $q-1$ basis matrices are given by $\underline{\underline{M}}_1^k$ and the $q-2$ matrices*

$$
\underline{\underline{M}}_j^k := \begin{bmatrix}
0 & 1 & \cdots & 1 & -1 & 1 & \cdots & 1 \\
-1 & \ddots & \ddots & & & & & \vdots \\
\vdots & \ddots & \ddots & \ddots & & \mathbf{1} & & \vdots \\
-1 & & \ddots & \ddots & \ddots & & & \vdots \\
1 & & & \ddots & \ddots & \ddots & & \vdots \\
-1 & \mathbf{-1} & & & \ddots & \ddots & \ddots & \vdots \\
\vdots & & & & & \ddots & \ddots & 1 \\
-1 & \cdots & \cdots & \cdots & \cdots & \cdots & -1 & 0
\end{bmatrix}, \quad j = 2, \ldots, q-1,
$$

which are generated from $\underline{\underline{M}}_1^k = (m_{ij}^k)_{1 \le i,j \le q}$ through replacement of $m_{1j}^k = 1$ by -1 and, in order to keep the matrices anti-symmetric, by replacing $m_{j1}^k = -1$ by 1.

Since the prolongation is the right-inverse of the restriction we are looking for a mapping $\mathbf{R} \to \mathbf{R}^q$, represented by the column vector

$$
\underline{P}_{k-1}^k := \begin{bmatrix} p_{i_1}^k \\ \vdots \\ p_{i_q}^k \end{bmatrix},
$$

such that

$$
\underline{D}_k^{k-1} \cdot \underline{P}_{k-1}^k = 1
$$

holds true.

Lemma 4.3 *A necessary and sufficient condition for $\underline{P}_{k-1}^k \in \mathbf{R}^q$ to be the representation of a prolongation operator is*

$$
\sum_{j=1}^{q} |\sigma_{i_j}^k| p_{i_j}^k = |\sigma_\ell^{k-1}|.
$$

Proof: Consider the product

$$
\underline{D}_k^{k-1} \cdot \underline{P}_{k-1}^k = \frac{1}{|\sigma_\ell^{k-1}|} \sum_{j=1}^{q} |\sigma_{i_j}^k| p_{i_j}^k = \frac{1}{|\sigma_\ell^{k-1}|} |\sigma_\ell^{k-1}| = 1,
$$

and the Lemma is established. ■

From this observation it is easy to construct a variety of recovery operators $\mathfrak{R}_{k-1}$.

4.3 The trivial recovery

The local recovery operator $\mathfrak{R}_{k-1}$ defined by

$$\mathfrak{R}_{k-1}v^{k-1} = \mathfrak{R}_{k-1}\mathfrak{A}(\sigma_\ell^{k-1})u := \mathfrak{A}(\sigma_\ell^{k-1})u = v^{k-1} \in \mathbf{R}$$

is called the trivial recovery operator.

Clearly, $\underline{P}_{k-1}^{k} = \mathfrak{D}_k\mathfrak{R}_{k-1}v^{k-1} = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} v^{k-1}$, i.e.

$$\underline{P}_{k-1}^{k} = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \in V^k = \mathbf{R}^q,$$

and thus

$$\sum_{j=1}^{q} |\sigma_{i_j}^k| p_{i_j}^k = \sum_{j=1}^{q} |\sigma_{i_j}^k| = |\sigma_\ell^{k-1}|.$$

4.4 Sophisticated recovery procedures

More sophisticated classes of recovery operators can be found in polynomial recovery procedures used in ENO methods, see [7], [8], [9], [10], [2], [23]. Here, a polynomial π_ℓ^{k-1} of certain degree is computed on the coarse grid such that the recovery condition

$$\mathfrak{A}(\sigma_\ell^{k-1})\pi_\ell^{k-1} = \frac{1}{|\sigma_\ell^{k-1}|}\int_{\sigma_\ell^{k-1}} \pi_\ell^{k-1}(\underline{x})\, d\underline{x} \stackrel{!}{=} \mathfrak{A}(\sigma_\ell^{k-1})u$$

holds. Then a recovery operator can be defined by means of

$$\mathfrak{R}_{k-1}v^{k-1} = \pi_\ell^{k-1}.$$

From

$$\underline{P}_{k-1}^{k}v^{k-1} = \mathfrak{D}_k\mathfrak{R}_{k-1}v^{k-1} = \begin{bmatrix} \mathfrak{A}(\sigma_{i_1}^k)\pi_\ell^{k-1} \\ \vdots \\ \mathfrak{A}(\sigma_{i_q}^k)\pi_\ell^{k-1} \end{bmatrix} =: \begin{bmatrix} p_{i_1}^k \\ \vdots \\ p_{i_q}^k \end{bmatrix} v^{k-1}$$

we can compute

$$\begin{aligned}\left(\sum_{j=1}^{q}|\sigma_{i_j}^k|p_{i_j}^k\right)v^{k-1} &= \sum_{j=1}^{q}|\sigma_{i_j}^k|\mathfrak{A}(\sigma_{i_j}^k)\pi_\ell^{k-1} = \sum_{j=1}^{q}|\sigma_{i_j}^k|\frac{1}{|\sigma_{i_j}^k|}\int_{\sigma_{i_j}^k}\pi_\ell^{k-1}(\underline{x})\,d\underline{x}\\ &= \sum_{j=1}^{q}\int_{\sigma_{i_j}^k}\pi_\ell^{k-1}(\underline{x})\,d\underline{x} = \int_{\sigma_\ell^{k-1}}\pi_\ell^{k-1}(\underline{x})\,d\underline{x}\\ &= |\sigma_\ell^{k-1}|\mathfrak{A}(\sigma_\ell^{k-1})u = |\sigma_\ell^{k-1}|v^{k-1},\end{aligned}$$

this showing $\sum_{j=1}^{q}|\sigma_{i_j}^k|p_{i_j}^k = |\sigma_\ell^{k-1}|$.

At no point reference was made to the polynomial structure of π_ℓ^{k-1}. Thus, any recovery function π_ℓ^{k-1} on σ_ℓ^{k-1} respecting the recovery condition $\mathfrak{A}(\sigma_\ell^{k-1})\pi_\ell^{k-1} = \mathfrak{A}(\sigma_\ell^{k-1})u$, not necessarily polynomial, defines a valid prolongation operator. In particular, polynomial recovery algorithms described in [8], [2], [23] and the recovery algorithms based on radial basis functions as described in [23], [24], [18] belong to this category.

However, in this work polynomial recovery is implemented as follows. If $\underline{\alpha} = (\alpha_1, \alpha_2)$ denotes a multiindex we want to compute quadratic polynomials on the coarse grid. Let ν_ℓ denote the number of neighbouring cells $\sigma_{\ell_j}^{k-1}$ of σ_ℓ^{k-1}. We then require

$$\left\{\begin{array}{rcl}\mathfrak{A}(\sigma_\ell^{k-1})\pi_\ell^{k-1} &=& \mathfrak{A}(\sigma_\ell^{k-1})u\\ \mathfrak{A}(\sigma_{\ell_j}^{k-1})\pi_\ell^{k-1} &=& \mathfrak{A}(\sigma_{\ell_j}^{k-1})u, \quad j = 1,\ldots,\nu_\ell\end{array}\right. \tag{2}$$

to hold for a second-degree polynomial $\pi_\ell^{k-1}(x) := \sum_{|\underline{\alpha}|\le 2} a_{\underline{\alpha}}(\underline{x}-\underline{c_\ell})^{\underline{\alpha}}$ on σ_ℓ^{k-1}. Note that $\underline{c_\ell}$ denotes the barycentre of σ_ℓ^{k-1}.

We always try to compute a quadratic polynomial. However, if ν_ℓ is too small (this may happen at a few places at the boundary of a domain) we confine ourselves to a linear polynomial.

In general, system (2) will be overdetermined since ν_ℓ is larger than five almost everywhere. Solving (2) with a least squares approach results in a polynomial with $\mathfrak{A}(\sigma_\ell^{k-1})\pi_\ell^{k-1} \neq \mathfrak{A}(\sigma_\ell^{k-1})u$ in general. Thus, we have to enforce the conservation of the cell average on σ_ℓ^{k-1} by solving instead the system

$$\mathfrak{A}(\sigma_\ell^{k-1})\left(\tilde{\pi}_\ell^{k-1} - u\right) = 0 \tag{3}$$

$$\mathfrak{A}(\sigma_{\ell_j}^{k-1})\left(\tilde{\pi}_\ell^{k-1} - \mathfrak{A}(\sigma_\ell^{k-1})u\right) = \mathfrak{A}(\sigma_{\ell_j}^{k-1})u - \mathfrak{A}(\sigma_\ell^{k-1})u, \quad j = 1,\ldots,\nu_\ell \tag{4}$$

for a polynomial $\tilde{\pi}_\ell^{k-1}$ slightly modified to ensure equality (3). Choosing the ansatz

$$\tilde{\pi}_\ell^{k-1}(x) := \sum_{|\underline{\alpha}|=1}\tilde{a}_{\underline{\alpha}}(\underline{x}-\underline{c_\ell})^{\underline{\alpha}} + \sum_{|\underline{\alpha}|=2}\tilde{a}_{\underline{\alpha}}\left((\underline{x}-\underline{c_\ell})^{\underline{\alpha}} - C_{\underline{\alpha}}\right)$$

with

$$C_{\underline{\alpha}} := \mathfrak{A}(\sigma_\ell^{k-1})\,((\underline{x} - \underline{c}_\ell)^{\underline{\alpha}})$$

enforces (3) in the case of quadratic recovery while

$$\tilde{\pi}_\ell^{k-1}(x) := \sum_{|\underline{\alpha}|=1} \tilde{a}_{\underline{\alpha}}(\underline{x} - \underline{c}_\ell)^{\underline{\alpha}}$$

is the right choice for a linear polynomial. Note that now only $0 < |\underline{\alpha}| \leq 2$ coefficients $\tilde{a}_{\underline{\alpha}}$ have to be computed. The original polynomial coefficients $a_{\underline{\alpha}}$ are finally computed from

$$\begin{aligned} a_{(0,0)} &= \mathfrak{A}(\sigma_\ell^{k-1})u - \sum_{|\underline{\alpha}|=2} \tilde{a}_{\underline{\alpha}} C_{\underline{\alpha}} \\ a_{\underline{\alpha}} &= \tilde{a}_{\underline{\alpha}}, \quad |\underline{\alpha}| \geq 1. \end{aligned}$$

5 Numerical results

We performed tests on two types of domains. The first two tests are concerned with generic functions on the unit square. The third was carried out using the numerical solution of the double Mach reflection on a wedge by the WENO scheme; for a complete description see [12]. The finest grid was generated using the algorithm of Friedrich [11]. The sequence of coarser grids was computed using the agglomeration algorithm described above.
By storing the coefficients of the prediction errors that do not fall below a certain tolerance ε we restricted the discrete cell data from the finest to the coarsest grid. Thus, only the data on the coarsest grid and the remaining coefficients of the prediction errors after restriction had to be stored. We reconstructed the data from the coarsest to the finest grid in consideration of the remaining prediction error coefficients.
Note that not the scale coefficients, but rather the prediction errors were calculated and truncated. The computation of the scale coefficients is only of theoretical interest in the case of data compression:

- For computing the scale coefficients, a system of linear equations has to be solved for each coarse cell with sums of fine cell areas as matrix entries. This leads to a $1/h^2$ scaling that can hardly be handled on unstructured grids. This results in scale coefficients of different order of magnitude than the prediction errors. Thus, it is not recommendable to truncate the scale coefficients, because in this way the prediction errors cannot be sufficiently controlled.

- The computation of the scale coefficients does not reduce the required memory in all cases: Consider a coarse cell and its four fine cells. The calculation leads to a presentation of the errors with three scale coefficients. However, if there are only one or two prediction errors, more memory can be saved without operating with the scale coefficients.

Two different compression rates were calculated. These rates indicate the percentage storage saved in comparison to the storage of all coefficients of the prediction errors and the data on the coarsest grid (1. rate) and the storage of the data on the finest grid (2. rate), respectively:

1. Compression rate:

$$100 - \frac{100 \cdot (|\{\text{all cells of all grids}\}| - |\{\text{pred. errors} < \varepsilon\}|)}{|\{\text{all cells of all grids}\}|}$$

2. Compression rate:

$$100 - \frac{100 \cdot (|\{\text{all cells of all grids}\}| - |\{\text{pred. errors} < \varepsilon\}|)}{|\{\text{all cells of the finest grid}\}|}$$

In addition, the L_1- and L_∞-errors for all test cases were calculated, as given in tables 1–3. Inspection of the tables also reveals that a sequence of five and nine grid levels was used in the test cases, respectively, with ε being $\varepsilon = 10^{-2}, 10^{-3}$ and 10^{-4}.
Suppose we have given levels $L, L-1, \dots, 0$ with the finest level L, the coarsest 0 and a given ε, the level dependent ε_k are defined as follows:
$\varepsilon_L := \varepsilon$

$\forall k = L-1, \dots, 1:$
  $\varepsilon_k := \alpha_k \varepsilon_{k+1}$ with

$$\alpha_k := \max_i \max_j \sqrt{\frac{|C_j^{k+1}|}{|C_i^k|}} \quad \text{with} \quad |C_i^k| = \bigcup_j |C_j^{k+1}|.$$

5.1 Two auxiliary test functions

The finest grid of the unit square is composed of 23833 cells which corresponds to $h = 1/150$, the fifth and ninth are made of 602 and 14 cells, respectively.

The two test functions are defined as:

$$1. \quad f_1(x) = \sin(12(x - \tfrac{1}{2})(y - \tfrac{1}{2}))$$

$$2. \quad f_2(x) = \begin{cases} \sin(x)\cos(y), & x > \frac{1}{2} \\ \sin(y)\cos(x), & \text{otherwise} \end{cases}$$

These functions were chosen to demonstrate the data compression achieved by the multiresolution algorithm.
The first function, f_1, does not display any discontinuous parts. For the cell data and the corresponding prediction errors see figure 2. The first and third column of the figure show the isolines of the cell data reconstructed locally on each cell. The errors shown right of the isolines in figs. 2 and 3, respectively, refer to the transition from the function on the left to the one plotted below. Fig. 2 reveals that almost no errors needed to be stored by the first four restrictions. Starting with the fifth grid an increase of large errors can be detected. Application of the three coarsest grids does not lead to a good representation of the initial function. Excellent compression rates of 98.45% and 97.47% are obtained by setting $\varepsilon = 10^{-2}$, rates of 95.96% and 93.44% with $\varepsilon = 10^{-3}$, using five grids. Extending the number of levels to nine leads to compression rates of 98.85% and 98.12%, respectively ($\varepsilon = 10^{-2}$). Function f_1 is a smooth, but quite an oscillatory function, which results in smaller, but still acceptable compression rates setting $\varepsilon = 10^{-4}$.
In contrast, the function f_2 displays discontinuities along the ray $x = 0.5$. The reconstructed data and the prediction errors are presented in figure 3. Due to the discontinuities more prediction errors needed to be stored in contrast to function f_1. The choise of $\varepsilon = 10^{-2}$ lead to compression rates of 95.44% and 92.58%, respectively. Only the discontinuous parts of f_2 had to be stored on each level for all choices of ε.
The comparison of the compression rates of both functions indicates that application of all nine grids is not recommendable. In all cases the usage of five grids leads to similar compression rates as application of nine grids.

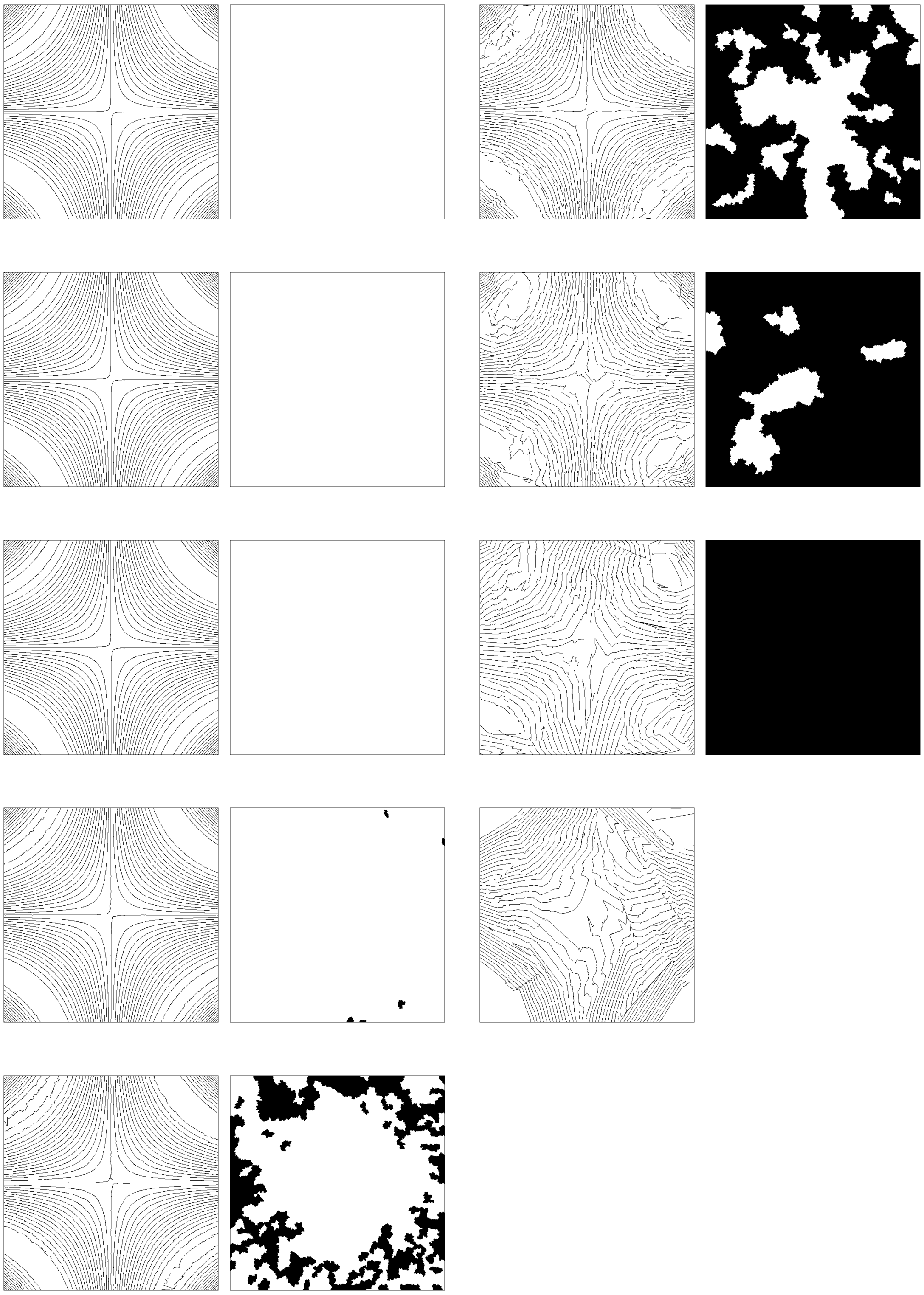


Figure 2: Reconstructed cell data and prediction errors for function f_1 with $\varepsilon = 10^{-3}$

# level	ε-var	L_1-error	L_∞-error	1. cpr.-rate	2. cpr.-rate
	10^{-2}	1.07e-03	1.22e-02	98.45%	97.47%
5	10^{-3}	2.97e-04	2.37e-03	95.96%	93.44%
	10^{-4}	3.73e-05	1.90e-04	81.91%	70.58%
	10^{-2}	2.79e-03	1.99e-02	98.85%	98.12%
9	10^{-3}	3.28e-04	2.11e-03	95.30%	92.28%
	10^{-4}	3.77e-05	1.90e-04	81.15%	69.05%

Table 1: Function f_1: errors and compression rates

# level	ε-var	L_1-error	L_∞-error	1. cpr.-rate	2. cpr.-rate
	10^{-2}	5.50e-04	1.94e-02	95.44%	92.58%
5	10^{-3}	4.02e-05	1.51e-03	92.21%	87.33%
	10^{-4}	1.09e-05	1.62e-04	90.95%	85.29%
	10^{-2}	2.08e-03	2.08e-02	96.27%	93.87%
9	10^{-3}	1.42e-04	1.79e-03	92.71%	88.04%
	10^{-4}	2.13e-05	1.62e-04	90.85%	84.98%

Table 2: Function f_2: errors and compression rates

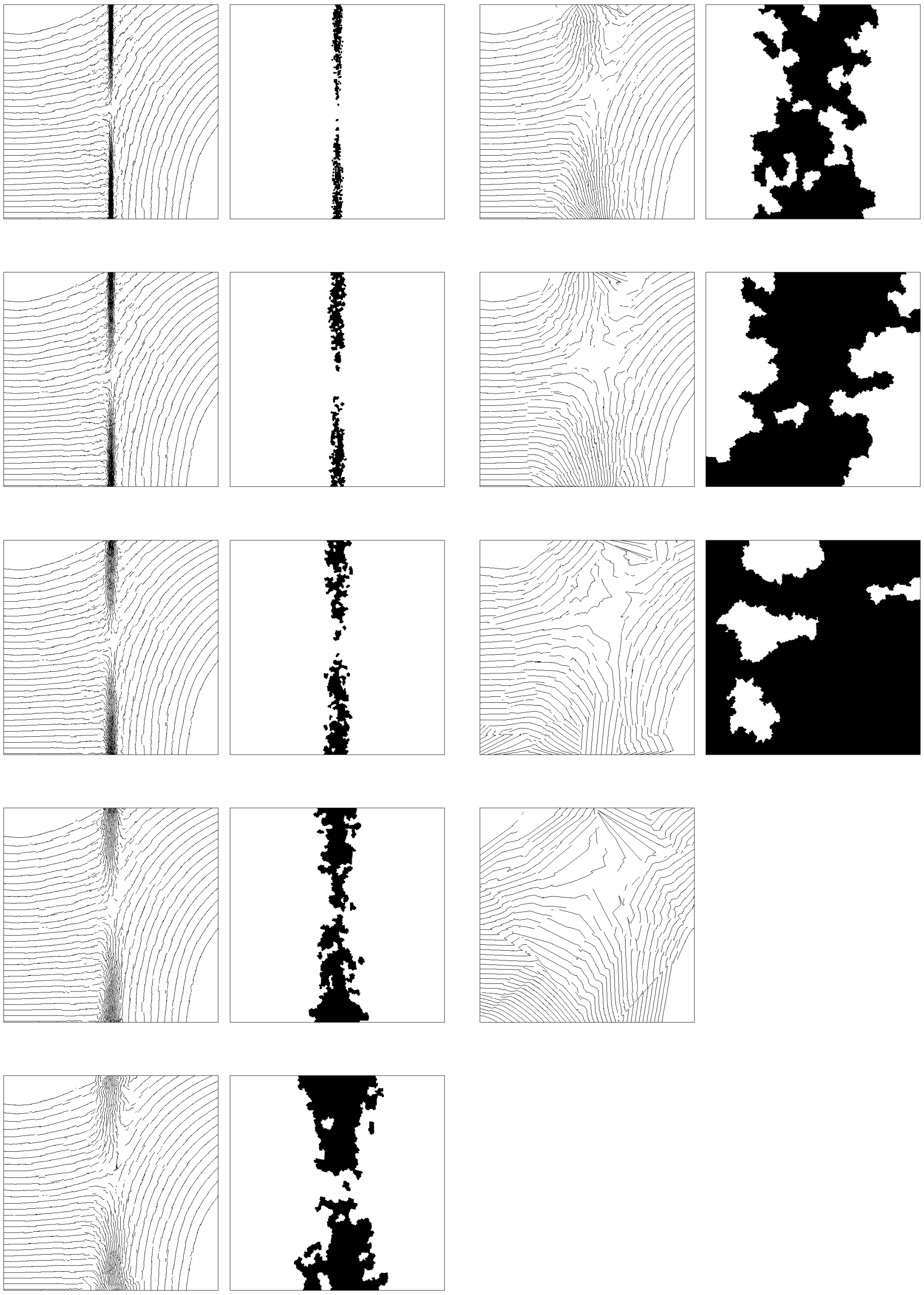


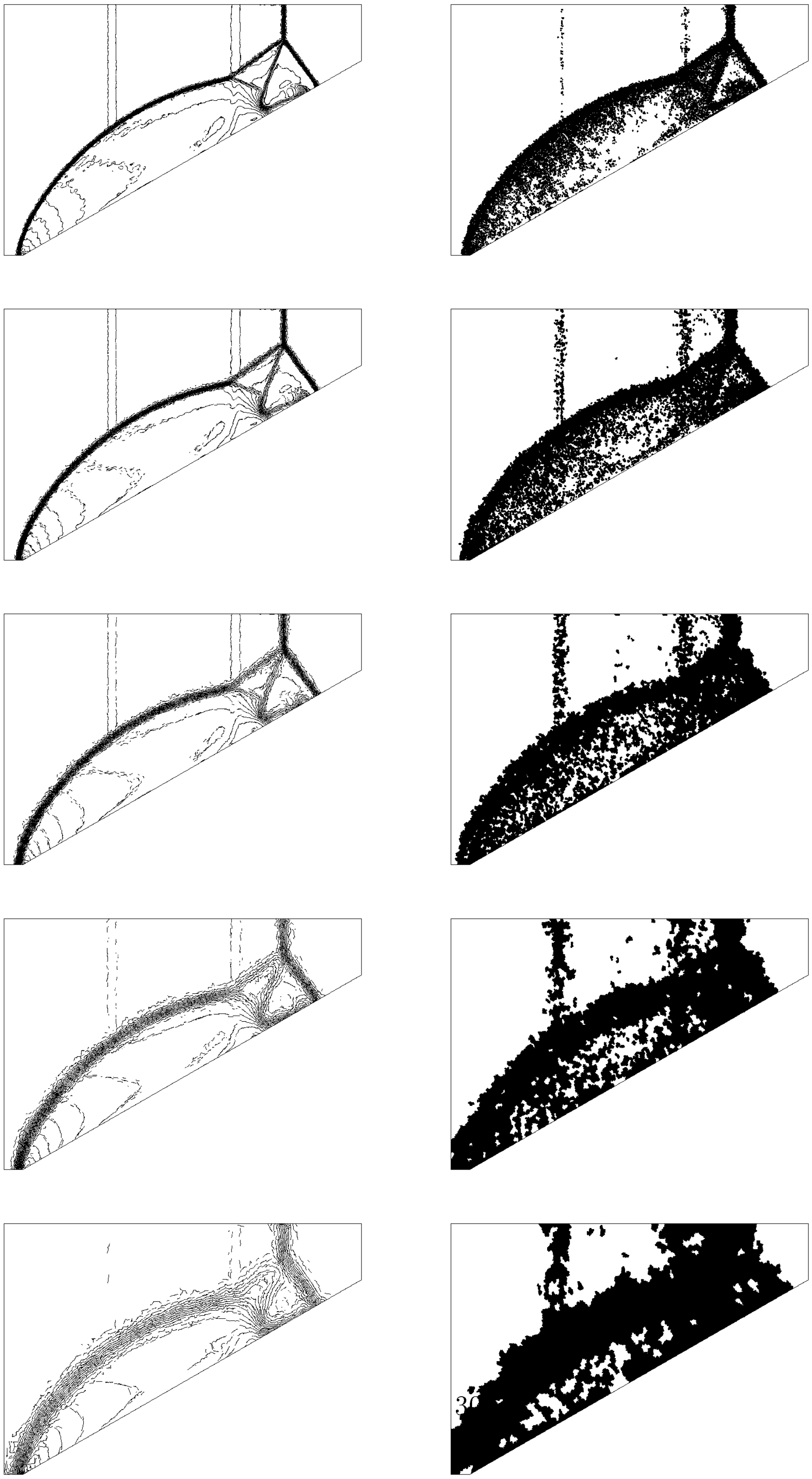
Figure 3: Reconstructed cell data and prediction errors for function f_2 with $\varepsilon = 10^{-2}$

5.2 A solution of a WENO scheme

The finest grid on the wedge contains 63497 cells which corresponds to $h = 1/120$, the fifth 1607, and the ninth 39 cells.
In this test case many prediction errors had to be stored even on the finest grid which is clearly seen in figure 4. This yields compression rates of 75.67% and 60.27%, respectively, with $\varepsilon = 10^{-2}$. With decreasing ε the compression rates decrease dramatically: Setting $\varepsilon = 10^{-4}$ the 2. rate amounts to 0.97%. Thus, in this case the effort of the multiresolution algorithm does not remunerate.

# level	ε-var	L_1-error	L_∞-error	1. cpr.-rate	2. cpr.-rate
	10^{-2}	2.22e-03	2.93e-02	75.67%	60.27%
5	10^{-3}	1.37e-04	2.47e-03	53.17%	23.53%
	10^{-4}	3.96e-06	2.10e-04	39.36%	0.97%
	10^{-2}	2.80e-03	3.00e-02	75.89%	60.25%
9	10^{-3}	1.68e-04	3.04e-03	53.27%	22.96%
	10^{-4}	4.86e-06	2.12e-04	39.49%	0.23%

Table 3: Numerical solution of a WENO scheme: errors and compression rates



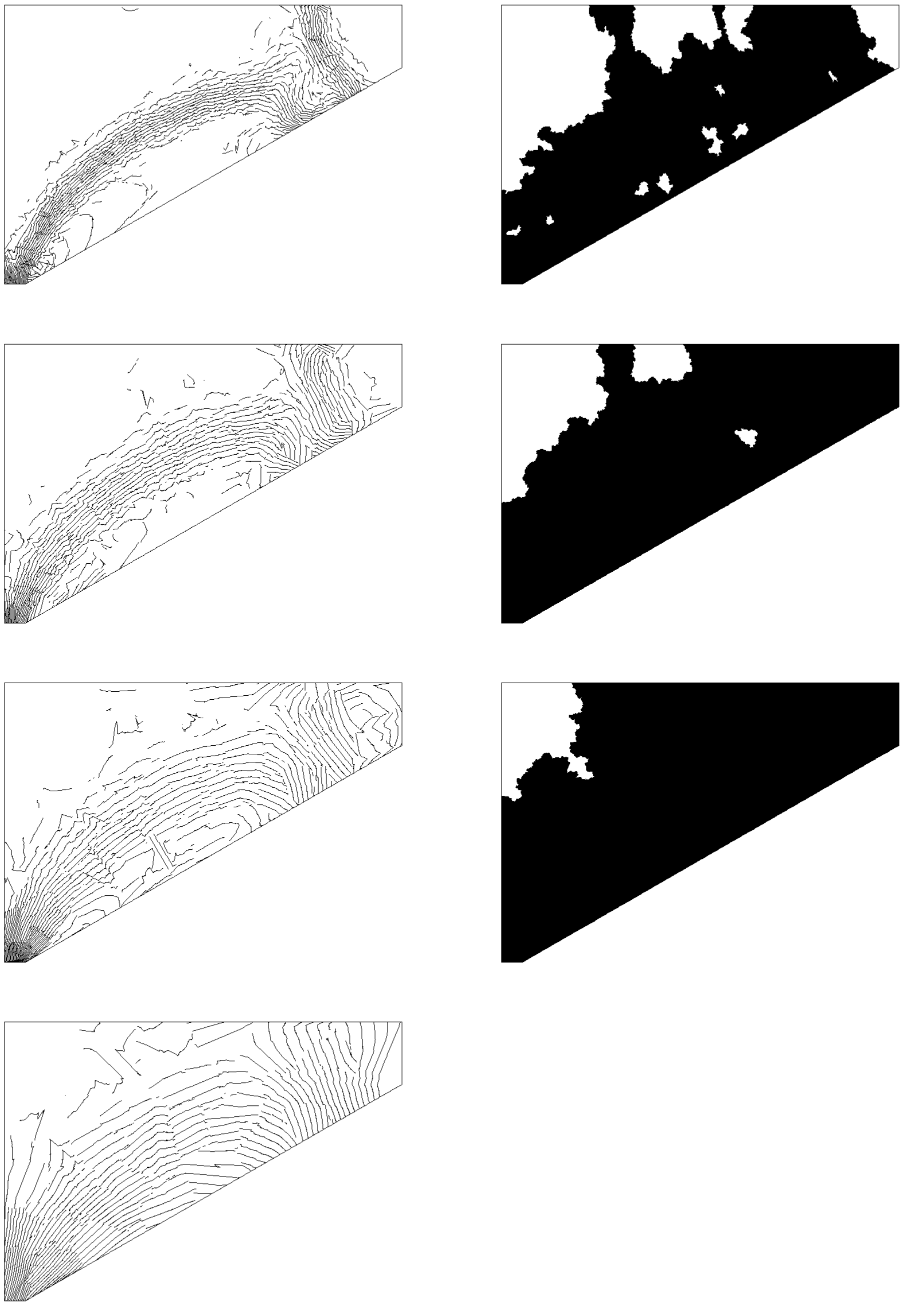


Figure 4: Reconstructed cell data and prediction errors for the solution of the computation with the WENO scheme, $\varepsilon = 10^{-2}$

6 Conclusions

We have presented a fairly general theory of generalized multiresolution analysis based on the work of Harten and Abgrall. Our central interest is in the analysis of discrete data from Computational Fluid Dynamics and hence the multiresolution of cell averages appears to be the main topic. All operations can be broken down to simple matrix-vector arithmetic. In particular, the representation of the scale coefficients is done with the help of basis matrices. A generally applicable agglomeration algorithm for arbitrary tesselations was given and a reliable and robust prolongation procedure on unstructured grids was developed. Numerical experiments document the ability of the algorithms described to accurately detect discontinuities and to allow massive data compression.

Acknowledgements

The work reported here was initiated during a collaborative research project funded by A.P.A.P.E. and DAAD through the PROCOPE program 312/pro-gg some years ago. The authors gratefully acknowledge A.P.A.P.E. and DAAD for the financial support of this exchange program and their french partner, Rémi Abgrall, for introducing them to the topic of generalized MRA.

References

[1] R. Abgrall - Design of an Essentially Nonoscillatory Reconstruction Procedure on Finite-Element-Type Meshes. *Manuscript, (1994)*

[2] R. Abgrall - On Essentially Non-Oscillatory Schemes on Unstructured Meshes: Analysis and Implementation. *J. Comp. Phys.* **114**, *45-58, (1994)*

[3] R. Abgrall, A. Harten - Multiresolution Representation in Unstructured Meshes: I.Preliminary Report. *CAM Report 94-20, University of California, Los Angeles, (1994)*

[4] J.M. Carnicer, W. Dahmen, J.M. Peña - Local Decomposition of Refinable Spaces and Wavelets. *Institut für Geometrie und Praktische Mathematik, RWTH Aachen, Bericht Nr.112, (1994)*

[5] I. Daubechies - Ten Lectures on Wavelets. *Regional Conference Series in Applied Mathematics 61, SIAM, Philadelphia, Pennsylvania, (1992)*

[6] W. Hackbusch - Multi-Grid Methods and Applications. *Springer Series in Computational Mathematics 4, Springer-Verlag, Berlin, Heidelberg, New York, Tokyo, (1985)*

[7] A. Harten, S. Osher - Uniformly High Order Accurate Nonoscillatory Schemes I. *SIAM J. Num. Anal.* **24**, *279-309, (1987)*

[8] A. Harten, S.R. Chakravarthy - Multi-Dimensional ENO Schemes for General Geometries. *ICASE Report No. 91-76, (1991)*

[9] A. Harten, S. Osher, B. Engquist, S.R. Chakravarthy - Some Results on Uniformly High-Order Accurate Essentially Nonoscillatory Schemes. *Appl. Num. Math.* **2**, *347-377, (1986)*

[10] A. Harten, B. Engquist, S. Osher, S.R. Chakravarthy - Uniformly High Order Accurate Essentially Non-Oscillatory Schemes III. *J. Comp. Phys.* **71**, *231-303, (1987)*

[11] O. Friedrich - A New Method for Generating Inner Points of Triangulations in Two Dimensions. *Comput. Methods Appl. Mech. Engrg.* **104**, *77-86, (1993)*

[12] O. Friedrich - Dissertation. *Universität Hamburg, in preparation*

[13] O. Friedrich - Weighted Essentially Non-Oscillatory Schemes for the Interpolation of Mean Values on Unstructured Grids. *J. Comp. Phys., in press*

[14] A. Harten - Multiresolution Representation of Data. *CAM Report 93-13, University of California, Los Angeles, (1993)*

[15] A. Harten - Multiresolution Representation of Data: II.General Framework. *CAM Report 94-10, University of California, Los Angeles, (1994)*

[16] A. Harten - Discrete Multi-Resolution Analysis and Generalized Wavelets. *Applied Numerical Mathematics* **12**, *153-192, (1993)*

[17] C. Hirsch - Numerical Computation of Internal and External Flows, Volume 2: Computational Methods for Inviscid and Viscous Flows. *John Wiley & Sons, (1990)*

[18] A. Iske, Th. Sonar - On the Structure of Function Spaces in Optimal Recovery of Point Data for ENO-Schemes by Radial Basis Functions. *Numerische Mathematik* **74**, *177-201, 1996*

[19] M.-H. Lallemand, H. Steve, A. Dervieux - Unstructured Multigridding by Volume Agglomeration: Current Status. *Computers Fluids* **21**, *397-433, (1992)*

[20] P.D. Lax - Hyperbolic Systems of Conservation Laws and the Mathematical Theory of Shock Waves. *Regional Conference Series in Applied Mathematics 11, SIAM, Philadelphia, (1973)*

[21] Y. Meyer - Wavelets and Operators. *Cambridge Studies in Advanced Mathematics 37, Cambridge University Press, (1992)*

[22] Y. Meyer - Wavelets: Algorithms & Applications. *SIAM, Philadelphia, (1993)*

[23] Th. Sonar - Mehrdimensionale ENO-Verfahren. *B.G. Teubner, Stuttgart, 1997*

[24] Th. Sonar - Optimal Recovery Using Thin Plate Splines in Finite Volume Methods for the Numerical Solution of Hyperbolic Conservation Laws. *IMA J. Num. Anal.* **16**, *549-581, 1996*

[25] E.V. Vorozhtsov, N.N. Yanenko - Methods for the Localization of Singularities in Numerical Solutions of Gas Dynamics Problems. *Springer Series in Computational Physics, Springer-Verlag, New York, Heidelberg, Berlin, London, Paris, Tokyo, Hong Kong, (1990)*

[26] G.G. Walter - Wavelets and Other Orthogonal Systems With Applications. *CRC Press, Boca Raton, Ann Arbor, London, Tokyo, (1994)*