34

Implementational Aspects of Prewavelet Sparse Grid Methods

A. Niedermeier[1], S. Zimmer[2]

Introduction

For sparse grid discretizations of elliptic PDEs [Zen90, Bun98, BD98], prewavelet bases [CW92] have been introduced to construct fast iterative schemes [GO94, GO95]. In this paper, we want to recall that the implementation of the prewavelet preconditioner is simple, since it fits in the usual tensor product scheme underlying the sparse grid approach. The only extra work we have to take care of, are one-dimensional basis transformations and a diagonal scaling; all other calculations, especially the application of the stiffness matrix, are done in the hierarchical basis. Another way to use prewavelets is to implement a complete solver with explicit use of the prewavelets. At first glance, this seems to be more involved than its hierarchical basis counterpart, due to the larger support of the basis functions. But for the Helmholtz equation on the unit cube, it turns out to be a much simpler code than in the usual hierarchical basis, since it decomposes completely into one-dimensional subproblems.

In the following two sections, we describe the discretization with the piecewise d-linear tensor product basis (hierarchical or prewavelet) that, in the case of more than one spatial dimension, can be used for sparse grid discretizations. Due to the tensor product structure of our discretization, we may first restrict the discussion of the corresponding algorithms to the one-dimensional case that we will later apply sequentially in each coordinate direction. We will pay special attention to the exchange of information between different levels, since this is the crucial part for

1 Zentrum Mathematik, Technische Universität München, D-80290 München
2 Institut für Informatik, Technische Universität München, D-80290 München
Eleventh International Conference on Domain Decomposition Methods
Editors Choi-Hong Lai, Petter E. Bjørstad, Mark Cross and Olof B. Widlund

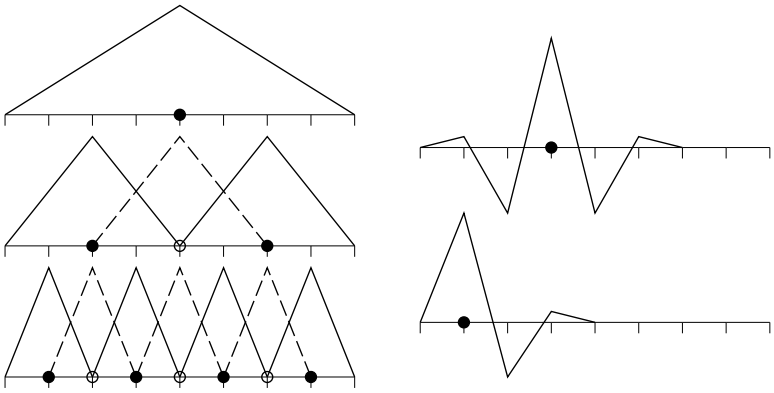

Figure 1 Left: the hierarchical basis B_3^H (solid lines) and the semidefinite system B_3^E (solid and dashed lines). Right: prewavelets for $l = 3$.

the generalization from one to an arbitrary number of dimensions, see [BD98] as an example.

Finally, we will demonstrate the application of the stiffness matrix for the Helmholtz equation in the prewavelet basis and give some numerically computed condition numbers.

The one-dimensional case

For the one-dimensional case, let Ω_l be the regular grid on $[0, 1]$ with mesh size $h_l = 2^{-l}$ and inner grid points $\{x_{l,i}, 1 \leq i \leq 2^l - 1\}$, V_l be the space of functions $u : \Omega \to \mathbf{R}$, piecewise linear with respect to Ω_l, and with $u(0) = u(1) = 0$.

The nodal basis B_l^N and the hierarchical basis B_l^H for V_l consist of the dilated and translated standard hat function $\phi(x) := \max(1 - |x|, 0)$, corresponding to the grid points: $\phi_{l,i}(x) := \phi((x - i \cdot h_l)/h_l)$. Namely, we have $B_l^N := \{\phi_{l,i}, 1 \leq i \leq 2^l - 1\}$ and $B_l^H := \{\phi_{k,i}, 1 \leq k \leq l \text{ and } 1 \leq i \leq 2^k - 1, i \text{ odd}\}$. The hierarchical basis leads to a splitting

$$V_l = \bigoplus_{k=0}^{l} W_k \tag{1}$$

with the hierarchical increments $W_k := \text{span}\{\phi_{k,i}, 1 \leq i \leq 2^k - 1, i \text{ odd}\}$.

The non-orthogonality of (1) with respect to the L_2-Norm leads to difficulties in the construction of efficient solvers for elliptic PDEs. It can be overcome by the introduction of prewavelets [CW92, GO94, GO95], which provide partial L_2-orthogonality for the price of basis functions with a larger support.

The prewavelet basis B_l^P of V_l is constructed by collecting basis functions of the orthogonal complements with respect to the L_2-Norm of V_{k-1} in V_k, $2 \leq k \leq l$. The

natural choice here are those basis functions $\psi_{l,i}$ with minimum support:

$$\psi_{l,i} := \frac{1}{10}\phi_{l,i-2} - \frac{3}{5}\phi_{l,i-1} + \phi_{l,i} - \frac{3}{5}\phi_{l,i+1} + \frac{1}{10}\phi_{l,i+2}$$

for $3 \le i \le 2^l - 3$, i odd, and two special cases on each level $l > 1$ for the two grid points next to the boundary:

$$\psi_{l,1} := \frac{9}{10}\phi_{l,1} - \frac{3}{5}\phi_{l,2} + \frac{1}{10}\phi_{l,3},$$

$\psi_{l,2^l-1}$ symmetrical, and finally $\psi_{1,1} := \phi_{1,1}$. Then, we define $B_l^P := \{\psi_{k,i}, 1 \le k \le l \text{ and } 1 \le i \le 2^k - 1, i \text{ odd}\}$. The corresponding splitting

$$V_l = \bigoplus_{k=0}^{l} \tilde{W}_k$$

with the hierarchical increments $\tilde{W}_k := \text{span}\{\psi_{k,i}, 1 \le i \le 2^k - 1, i \text{ odd}\}$, is orthogonal with respect to the L_2-Norm, but each $\psi_{k,i}$ (sufficiently far away from the boundary) overlaps with four other $\psi_{k,j}$ of the same level l (two on each side) without orthogonality.

We assume that the coefficients u_l^H with respect to the hierarchical basis and coefficients u_l^P with respect to the prewavelets are arranged in a binary tree.

The use of a generating system [Gri94a, Gri94b]

$$B_l^E := \bigcup_{k=0}^{l} B_k^N$$

that contains the nodal bases of all levels will be useful — here not for the construction of multilevel methods, but as an intermediate storage for the transformation between different bases. The storage for the additional coefficients in the extended vector u_l^E is provided by lists of length $l - k$ in the nodes of depth k in the binary tree.

Now, we turn to the transformation of the prewavelet coefficients u_l^P into the hierarchical basis. Besides the use of this operation in the context of a prewavelet preconditioner (where it is basically the only module in the solution process that knows about prewavelets) it can be used as a final step after the direct computation of the solution in prewavelet coefficients, and it can serve as a template for the general structure of all the modules we will need for our solver. On each level, the transformation from the prewavelet basis into the nodal basis is simply the application of a 5-point-stencil with the coefficients $(\frac{1}{10}, -\frac{3}{5}, 1, -\frac{3}{5}, \frac{1}{10})$ — with obvious modifications for the points next to the boundary. Since this nodal basis is part of the generating system, we get by level-wise application a transformation from the prewavelet basis into the B_l^E. To conclude the transformation, we could append a second sweep that runs from the finest towards the coarsest grid and eliminates the coefficients of the basis functions that do not belong to the hierarchical basis. However, it is more economic to avoid the overhead of two tree traversions and combine both sweeps, which is simply a reordering of the operations.

For the extension to d-dimensional problems, we will keep this algorithm on a one-dimensional binary tree as black box. However, it is important to note that the

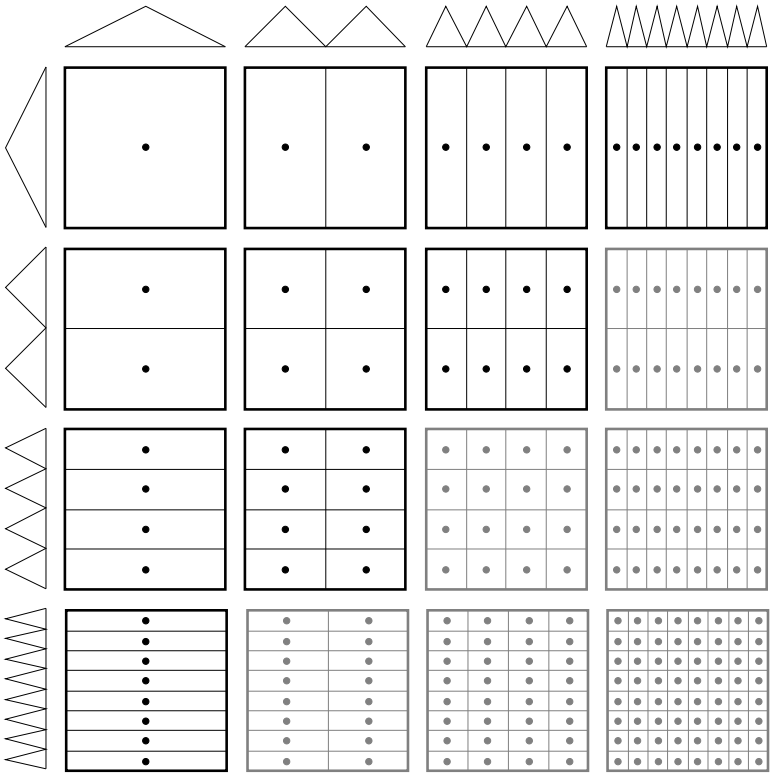

Figure 2 The scheme of grids corresponding to the hierarchical increments $W_{i,j}$ in $\mathbf{K}^{\mathbf{F}}$ (all grids) and $\mathbf{K}^{\mathbf{S}}$ (black grids only) for $d = 2$, $l = 3$.

information transport is only bottom-up: the output coefficients for one specific level k do only depend on the input coefficients of this and the finer grids $k \leq k' \leq l$.

The reverse transformation can be implemented in a similar manner, but it requires the solution of a pentadiagonal linear system of equations on each level. Here, the transport of information is bottom-up, too. The adjoint operators are constructed by the execution of the adjoints of their elementary operations in reverse order. Here, the flow direction changes: they are of top-down type.

The d-dimensional case

The general d-dimensional case is a straightforward tensor product approach. On the unit cube $[0,1]^d$ we define for the multi-index $\mathbf{l} := (l_1, \ldots, l_d)$ the grid $\Omega_{\mathbf{l}}$ with mesh sizes $\mathbf{h_l} := (2^{-l_1}, ..., 2^{-l_d})$, grid points $\mathbf{x_{l,i}} := (x_{l_1,i_1}, \ldots, x_{l_d,i_d})$, piecewise d-linear ansatz functions

$$\phi_{\mathbf{l},\mathbf{i}}(\mathbf{x}) := \prod_{k=1}^{l} \phi_{l_k,i_k}(x_k),$$

$$\psi_{\mathbf{l},\mathbf{i}}(\mathbf{x}) := \prod_{k=1}^{l} \psi_{l_k,i_k}(x_k),$$

and hierarchical increments $W_{\mathbf{k}} := \operatorname{span}\{\phi_{\mathbf{k},\mathbf{i}}, 1 \le i_j \le 2^{k_j} - 1, i_j \text{ odd for all } 1 \le j \le d\}$.

With these $W_{\mathbf{k}}$ we can build our finite element space

$$V_{\mathbf{l}} := \bigoplus_{\mathbf{k} \in \mathbf{K}} W_{\mathbf{k}}$$

for some index set $\mathbf{K}$. The usual full tensor product space is given by $\mathbf{K}^{\mathbf{F}} := \{\mathbf{l}, l_j \le l, 1 \le j \le d\}$ for some fixed l. For reasons of efficiency, thinking about the cost/benefit ratio of the different grids, however, we also allow subspaces hereof, where the only condition on $\mathbf{K}$ is to preserve hierarchy:

$$\mathbf{k} \in \mathbf{K} \Rightarrow \forall \mathbf{k}' \le \mathbf{k} : \mathbf{k}' \in \mathbf{K}$$

with the component-wise relation $\mathbf{k}' \le \mathbf{k} :\Leftrightarrow \forall 1 \le j \le d : k'_j \le k_j$. A typical choice for $\mathbf{K}$, which results in a sparse grid scheme, is

$$\mathbf{K}^{\mathbf{S}} := \{\mathbf{l}, \sum_{j=1}^{d} l_j \le l + d - 1\}, \tag{2}$$

see, for example, [Zen90, Bun98].

From the implementational viewpoint, the major difference between $\mathbf{K}^{\mathbf{F}}$ and schemes like $\mathbf{K}^{\mathbf{S}}$ is the absence of a finest grid $\Omega_{(l,\ldots,l)}$ in the latter case. This makes the usual way of implementation of hierarchical basis algorithms impossible, which uses a transformation onto the finest grid and there the nodal basis. In our case, the operations like application of the stiffness matrix have to be implemented explicitly on the hierarchical basis.

A suitable data structure for the coefficient vectors provides a nested structure of a binary trees, one for each coordinate direction $k = 1, \ldots, l$ with the $k-1$-dimensional structures as node values — except for $k = 1$, where the actual coefficients are stored.

The general principle to extend our one-dimensional algorithms — one may for example think of the transformation from the previous section — consists of one outer loop over the dimensions in which we set up a set of one-dimensional binary trees, namely for those grid points whose coordinates differ only in the k-th component, one for each value of this component. On each of these trees, we perform the one-dimensional algorithm.

With respect to the computational domain, this leads to a decomposition into a possibly large number of subdomains that are treated as one-dimensional objects. In contrast to most domain decomposition methods, the orientation of the subdomains is not fixed, but changes with the loop over the dimensions — there is no explicit mechanism for communication between the subdomains.

For the full tensor product space, one can easily convince oneself about the soundness of this algorithm, if we consider a test vector with all but one coefficient zero, and take into account the linearity of the operators.

For sparse grid spaces, which are no tensor product spaces themselves, but subspaces thereof, things are more difficult, since the flow of information in the full grid scheme in some cases uses finer grids as intermediate storage. They are no longer guaranteed to exist — the assumption on the index set is an assumption about the existence of coarser grids. As an example, consider the scheme from figure 2, and a one-dimensional algorithm that is applied first along horizontal lines and then along vertical lines in the scheme. If both top-down and bottom-up transports take place, information interchange from the lower left grid $\Omega_{(1,4)}$ to the upper right grid $\Omega_{(4,1)}$ would require the finest grid $\Omega_{(4,4)}$.

The direction of the information transport (bottom-up, e.g.) is the important fact — the d-dimensional algorithms work if, and only if, after a top-down transport towards finer grids, no bottom-up transport in any other coordinate direction is necessary.

Therefore, the transformations between u_l^P and u_l^H are no special problem — all transport is done bottom-up. The adjoint transformations can also be handled with this technique, since here all one-dimensional operators work top-down.

Note that we do not need permanent storage for the additional coefficients of the d-dimensional semidefinite system $B_{\mathbf{I}}^E$, since we use them only as intermediate storage within the one-dimensional black boxes. This is important, since the number of extra coefficients for fixed mesh size h grows roughly like 2^d.

Application: The Helmholtz equation

For the solution of the Helmholtz equation $\lambda u - \Delta u = f$ on Ω with Dirichlet boundary conditions, we have to evaluate the integrals

$$\lambda \cdot \int_\Omega u(\mathbf{x}) \cdot v(\mathbf{x})\, d\mathbf{x} + \int_\Omega \nabla u(\mathbf{x}) \cdot \nabla v(\mathbf{x})\, d\mathbf{x}$$

for $u, v \in V_K$ by application of the stiffness matrix. Since for tensor product functions

$$u(\mathbf{x}) := \prod_{j=1}^{d} u_j(x_j), \quad v(\mathbf{x}) := \prod_{j=1}^{d} v_j(x_j),$$

the above integrals can be split up into the one-dimensional integrals

$$\prod_{j=1}^{d} \int_0^1 u_j(x) \cdot v_j(x)\, dx, \tag{3}$$

$$\sum_{i=1}^{d} \int_0^1 u_i'(x) \cdot v_i'(x)\, dx \cdot \prod_{j \neq i} \int_0^1 u_j(x) \cdot v_j(x)\, dx, \tag{4}$$

the one-dimensional building blocks for the application of the stiffness matrix, are the application of the one-dimensional stiffness matrices A and B corresponding to

$$a(u, v) := \int_0^1 u'(x) \cdot v'(x)\, dx \quad \text{and} \quad b(u, v) := \int_0^1 u(x) \cdot v(x)\, dx.$$

Table 1 Condition numbers for the scaled prewavelet stiffness matrix on sparse grids (2), $\lambda = 0$ (top) and different values of λ for fixed $d = 3$ and $l = 7$ (bottom).

l	$d=1$	$d=2$	$d=3$	$d=4$	$d=5$	$d=6$	$d=7$	$d=8$
2	5.2	4.5	4.1	3.8	3.6	3.4	3.2	3.1
3	7.8	8.4	8.0	7.8	7.5	7.3	7.1	6.9
4	10.1	12.5	14.3	13.9	13.4	13.0	12.7	12.3
5	12.0	16.2	21.0	24.9	24.3	23.7	23.1	22.6
6	13.1	19.8	27.2	35.7	43.3	-	-	-
7	14.1	23.1	33.7	46.2	-	-	-	-
8	14.8	25.5	40.0	-	-	-	-	-

λ	0	10^2	10^4	10^6	10^8
κ	33.7	28.8	11.9	8.6	8.6

Due to our tensor product bases — and again the linearity of the operators — they are sufficient for the d-dimensional problem.

The one-dimensional algorithms are of quite different character for the hierarchical basis and for the prewavelets, respectively. The hierarchical basis is a-orthogonal, A^H is therefore diagonal, there is no exchange of information between the levels. The mass matrix B^H, however, is comparatively dense — the basis function of level one, for example, is coupled with all other basis functions. The transport of information takes place top-down and bottom-up as well. In the prewavelet basis, different levels are orthogonal with respect to b. On each level, a pentadiagonal matrix arises, but there are no interactions between the levels. Here, the matrix A^P is quite dense (the most convenient way of implementation is the transformation into the hierarchical basis) and causes top-down and bottom-up transport between the levels.

If we put it together for the integrals (3), (4), we note that for each summand we have at most one integral of type A and d or $d-1$ of type B. This causes difficulties for the hierarchical basis, since there are transport processes up and down in more than one coordinate direction and we can no longer apply the standard loop for d-dimensional problems. The algorithm to overcome this — that splits up the upward and the downward data — is relatively complicated, see, for example, [BD98].

In prewavelets, this problem does not arise, since the levels are decoupled in all but possibly one direction. So, we get with the straightforward d-dimensional extension the complete algorithm for the stiffness matrix of the Helmholtz equation on sparse grids. And — after diagonal scaling — the matrix is well conditioned (which was the original reason to introduce prewavelets for sparse grids).

To illustrate this, in table 1 we give some numerically computed condition numbers for the scaled prewavelet stiffness matrix on sparse grids (2). They were computed by simple power iterations with the algorithms above on a 133MHz Intel Pentium personal computer [Nie98].

Summary

The prewavelet basis for sparse grid discretizations results — besides the well conditioned stiffness matrices — in a particularly simple code for the solution of the Helmholtz equation on the unit cube, since the orthogonality properties allow the direct tensor-product type implementation with the number of spatial dimensions simply as a loop parameter.

For more realistic problems, as, for example, on transformed domains, this advantage is lost. Then, it will probably more reasonable to implement the generalizations in an code for the hierarchical basis and to hide the prewavelets in a preconditioner that simply consists of the transformation between prewavelets and the hierarchical basis, its adjoint operator, and a diagonal scaling, and gives the same condition numbers as the direct implementation in prewavelets.

REFERENCES

[BD98] Bungartz H.-J. and Dornseifer T. (1998) Sparse grids: Recent developments for elliptic partial differential equations. In Hackbusch W. and Wittum G. (eds) *Multigrid Methods V*, volume 3 of *LNCSE*. Springer, Berlin/Heidelberg.

[Bun98] Bungartz H.-J. (1998) *Finite Elements of Higher Order on Sparse Grids.* Shaker Verlag, Aachen.

[CW92] Chui C. K. and Wang J. Z. (1992) On compactly supported spline wavelets. *Trans. Amer. Math. Soc.* 330: 903–915.

[GO94] Griebel M. and Oswald P. (1994) On additive schwarz preconditioners for sparse grid discretizations. *Numer. Math.* 66: 449–463.

[GO95] Griebel M. and Oswald P. (1995) Tensor product type subspace splittings and multilevel iterative methods for anisotropic problems. *Adv. Comput. Math.* 4: 171–206.

[Gri94a] Griebel M. (1994) Multilevel algorithms considered as iterative methods on semidefinite systems. *SIAM J. Sci. Stat. Comput.* 15(3): 547–565.

[Gri94b] Griebel M. (1994) *Multilevelmethoden als Iterationsverfahren über Erzeugendensystemen.* B.G.Teubner , Stuttgart.

[Nie98] Niedermeier A. (1998) Lösung von Differentialgleichungen durch Prewavelets. Diplomarbeit, Fakultät für Mathematik, TU München.

[Zen90] Zenger C. (1990) Sparse grids. In Hackbusch W. (ed) *Parallel Algorithms for Partial Differential Equations Proceeings of the 6th GAMM-Seminar, Kiel. Notes on Numerical Fluid Mechanics Vol. 31.* Vieweg, Braunschweig.