Adaptive Sparse Grids for Hyperbolic Conservation Laws

Michael Griebel and Gerhard Zumbusch

Abstract. We report on numerical experiments using adaptive sparse grid discretization techniques for the numerical solution of scalar hyperbolic conservation laws. Sparse grids are an efficient approximation method for functions. Compared to regular, uniform grids of a mesh parameter h contain h^{-d} points in d dimensions, sparse grids require only $h^{-1}|logh|^{d-1}$ points due to a truncated, tensor-product multi-scale basis representation.

For the treatment of conservation laws two different approaches are taken: First an explicit time-stepping scheme based on central differences is introduced. Sparse grids provide the representation of the solution at each time step and reduce the number of unknowns. Further reductions can be achieved with adaptive grid refinement and coarsening in space. Second, an upwind type sparse grid discretization in $d+1$ dimensional space-time is constructed. The problem is discretized both in space and in time, storing the solution at all time steps at once, which would be too expensive with regular grids. In order to deal with local features of the solution, adaptivity in space-time is employed. This leads to local grid refinement and local time-steps in a natural way.

1. Sparse Grids

Sparse grids were introduced for the solution of elliptic partial differential equations, see [4] and references in [2]. They provide an efficient approximation method of smooth functions, especially in higher dimensions. So far, Galerkin methods [2] and finite difference schemes [3, 9] for elliptic problems on sparse grids have been investigated. There are also attempts to solve parabolic problems [1] and Navier-Stokes equations [9].

The multi-dimensional approximation scheme of sparse grids can be constructed as a subspace of the tensor-product of one-dimensional spaces represented by a hierarchical multi-resolution scheme [5]. Consider piecewise linear interpolants on a d-dimensional unit hyper-cube. We start with the one-dimensional hierarchical basis [11]. The space of functions on the regular grid of dyadic level l and mesh parameter $h = 2^{-l}$ can be represented by the space of all tensor-products of one-dimensional basis functions of support larger than 2^{-l-1}. The corresponding sparse grid space consists of all products of hierarchical basis functions with support larger than a d-dimensional volume of size 2^{-l-1}, see Figure 1. This is

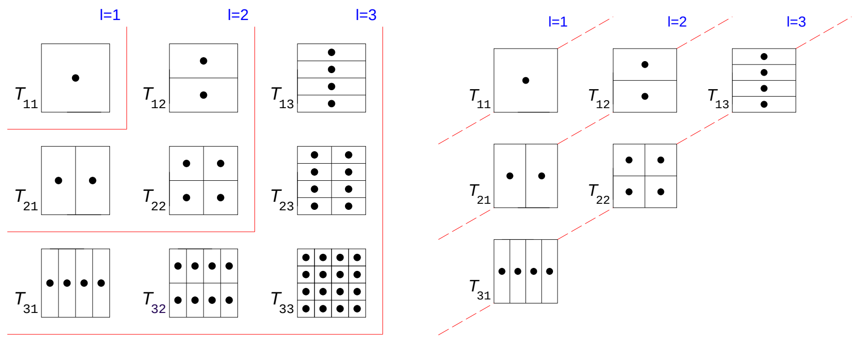


FIGURE 1. Tableau of supports of hierarchical basis functions for a regular (left) and a sparse grid.

a subset of the regular grid space. A regular grid has about $2^{d\cdot l}$ nodes, which is substantially more than the $2^l \cdot l^{d-1}$ nodes of the sparse grid.

Now it is straightforward to apply a Galerkin scheme to the spaces defined [2]. Another way to discretize equations on sparse grids is the combination technique, which is an extrapolation scheme applied to solutions obtained on several regular grids of about 2^l and 2^{l-1} nodes [4]. There exists also a Fourier-collocation method on sparse grids [6].

A different way of a sparse grid discretization is a finite difference scheme [3], which we will employ throughout this paper. We define the hierarchical transformation $\mathbf{H}$ as the hierarchical basis transformation on the regular grid from nodal values to hierarchical values which is restricted to the sparse grid nodes. Based on $\mathbf{H}$, we define the action of a one-dimensional finite difference operator for the discretization of a differential operator: We apply the associated standard difference stencil $\mathbf{D}_i$ along the x_i-axis to values located on the sparse grid nodes in a specific basis representation. To this end the values are given in nodal basis in direction i and in hierarchical basis representation in all other directions $I \setminus \{i\}$. The associated transformation is denoted by $\mathbf{H}_{I\setminus\{i\}}$. The stencil $\mathbf{D}_i$ for each node itself is chosen as the narrowest finite difference stencil available on the sparse grid. It is equivalent to the corresponding stencil on a regular (non-equidistant) grid, e.g. a first order upwind stencil for $\frac{\partial}{\partial x_i}$. In nodal values the finite difference operator reads

$$\frac{\partial}{\partial x_i} u \;\approx\; \mathbf{H}^{-1}_{I\setminus\{i\}} \circ \mathbf{D}_i \circ \mathbf{H}_{I\setminus\{i\}} u$$

A general difference operator is then obtained by dimensional splitting. The linear advection term, for example, can be discretized in nodal basis representation as

$$\nabla \cdot u \;\approx\; \sum_{i=1}^{d} \mathbf{H}^{-1}_{I\setminus\{i\}} \circ \mathbf{D}_i \circ \mathbf{H}_{I\setminus\{i\}} u \qquad (1)$$

	regular grid	sparse grid
# nodes	$\mathcal{O}(h^{-d})$	$\mathcal{O}(h^{-1} \cdot \lvert \log_2 h \rvert^{d-1})$

TABLE 1. Number of nodes.

	d=1	d=2	d=3	d=1	d=2	d=3
order $p=1$	$\mathbf{N^{-1}}$	$N^{-1/2}$	$N^{-1/3}$	$\mathbf{N^{-1}}$	$\mathbf{N^{-1+\gamma}}$	$\mathbf{N^{-1+\gamma}}$
order $p=2$	N^{-2}	$\mathbf{N^{-1}}$	$N^{-2/3}$	N^{-2}	$N^{-2+\gamma}$	$N^{-2+\gamma}$
order $p=3$	N^{-3}	$N^{-3/2}$	$\mathbf{N^{-1}}$	N^{-3}	$N^{-3+\gamma}$	$N^{-3+\gamma}$

TABLE 2. Storage-complexity of a regular (left) and a sparse grid discretization.

where the one dimensional difference operators $\mathbf{D}_i$ may be chosen as a two-point upwind stencil $\frac{1}{x_i - x_{i-1}} \cdot [-1\ 1\ .]$. On adaptively refined grids, the nearest neighbor nodes are chosen, which may lead to unsymmetric stencils.

The major advantage of sparse grids compared to regular grids is their smaller number of nodes for the same level l and resolution 2^{-l}. This is especially true in higher dimensions $d > 1$, see Table 1.

However, the question whether sparse grids have an advantage compared to regular grids does also depend on the accuracy of a solution obtained on a grid. We are interested in a comparison of accuracy versus number of nodes for both types of grids. We define the storage ε-complexity of an approximation method by the accuracy ε which can be achieved with a storage of N nodes. The accuracy depends on the smallest mesh parameter h and an approximation order p like $\varepsilon = \mathcal{O}(h^p)$ for smooth data. For regular grids the number of nodes depends on the space dimension d as $N_{storage} = h^{-d}$ which results in $\varepsilon = \mathcal{O}(N^{-p/d})$. In the case of sparse grids, the dependence on the dimension d is much weaker and we denote $\varepsilon = \mathcal{O}(N^{-p+\gamma})$ for every $\gamma > 0$ and an approximation order p. The sparse grid approximation breaks the curse of dimensionality.

Let us assume that a sparse grid approximation is of first order $p = 1$, which of course depends on the discretization order, the error norm, the smoothness of the solution and the sparse grid approximation itself. Then the sparse grid is competitive to a second order method in two dimensions and to a third order method in three dimensions, which is usually hard to construct, see Table 2.

Furthermore the number-of-operations complexity is of interest, because it is an estimate for the computing time a specific algorithm needs. In some cases the work count is proportional to the number of nodes. This is true for a single time-step of a standard explicit finite difference code. It is also true for the corresponding sparse grid code, because it is true for each operator $\mathbf{H}$ and $\mathbf{D}$. However, the work count usually is higher than the number of nodes for implicit discretizations and for stationary problems involving the solution of (non-) linear equation systems, and for time-dependent problems in general. The number of time-steps for an evolution problem of a fixed time interval is proportional to h due to the CFL condition, which leads to a higher work count complexity. On regular grids we obtain $N_{work} = \mathcal{O}(h^{-d-1})$, which is equivalent to the storage complexity in $d+1$

space dimensions. This means that any reduction in storage $N_{storage}$, e.g. through sparse grids, may reduce the number of operations even further.

We still have to check the assumption on the approximation order $p = 1$ (or some other constant) of finite difference sparse grids discretizations. Up to now such orders had been verified for the Poisson equation under strong regularity assumptions [9]. Furthermore similar estimates are available for the interpolation error under suitable smoothness assumptions [2]. Numerical experiments indicate that adaptive sparse grids can weaken the smoothness requirements.

It is the goal of this article to construct suitable finite difference discretizations for conservation laws and to investigate their numerical order of convergence.

2. Space-Time Schemes

We want to solve the scalar conservation law

$$\begin{array}{rcl} u_t \;+\; \nabla \cdot f(u) & = & 0 \quad \text{for} \;\; u(x,t), \\ x & \in & \Omega \subset \mathbb{R}^d \\ t & \in & [0, t_0] \end{array} \tag{2}$$

written as an initial-boundary value problem. The standard procedure for the numerical solution of (2) is to discretize the space Ω and the initial value $u^0 = u(x, 0)$ on Ω and to step forward in time. The solution u^{t+1} at time step $t+1$ is computed from u^t and the boundary conditions. This *'time-stepping'* scheme is iterated until $t = t_0$ is reached. This will be discussed in chapter 3.

An alternative solution algorithm uses a discretization of (2) in the *'space-time'* domain $\Omega \times [0, t_0]$. The conservation law can be re-written as a boundary value problem

$$\begin{array}{rcl} \nabla \cdot F(u) & = & 0 \quad \text{for} \;\; u(x), \\ x & \in & \Omega \times [0, t_0] \subset \mathbb{R}^{d+1} \end{array} \tag{3}$$

with F given by $F_0(u) = u$ and $F_{1,\ldots,d}(u) = f(u)$. The algorithm requires the numerical solution of a single large (non-) linear equation system and returns an approximation of u at all time steps at once. This approach is related to waveform relaxation [10]. It has been used with finite elements for periodic parabolic equations on sparse grids in [1].

The storage requirements of the space-time formulation are often considered as too high. However, with sparse grid technique the additional dimension in storage is affordable. Furthermore there is the advantage of easy adaptive grid refinement in space-time. In any stage of the computation it is possible to introduce a finer grid, which gives better resolution in space and local time steps. This is often difficult or even impossible for time-stepping algorithms.

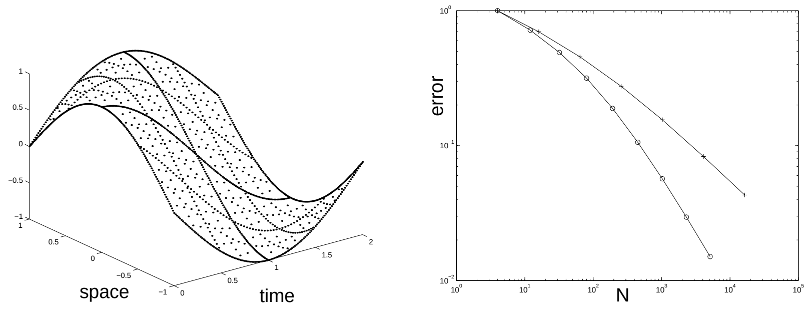


FIGURE 2. Sine wave $u = cos(\pi(x-t))$ (left); convergence history: L_∞ error vs. $N_{storage}$ (space-time), '+' regular grid and 'o' sparse grid.

$2/h$	L_1	L_2	L_∞	L_1	L_2	L_∞
32	1.8973	1.8713	1.7746	1.4347	1.6189	1.6752
64	1.9348	1.9194	1.8696	1.5539	1.7186	1.7834
128	1.9610	1.9534	1.9297	1.6518	1.7985	1.8652
256				1.7247	1.8558	1.9235
512				1.7746	1.8932	1.9606

TABLE 3. Convergence rates on regular (left) and on sparse grids, sine wave.

2.1. Numerical examples

In this section we will consider the linear, oblique advection equation $f(u) := u \cdot \mathbb{1}$ as a simple prototype for a conservation law. In space-time formulation it reads

$$\nabla \cdot u \;=\; 0 \qquad \text{in } \Omega \subset \mathbb{R}^{d+1} \tag{4}$$

We use a first order upwind discretization in space-time, which is equivalent to an implicit Euler discretization in time.

For example on a two dimensional regular grid (one space and one time dimension), this is

$$\frac{1}{h}(u_{i,j} - u_{i-1,j}) \;+\; \frac{1}{h}(u_{i,j} - u_{i,j-1}) \;=\; 0$$

The corresponding finite difference sparse grid discretization is

$$\mathbf{H}_{\{2\}}^{-1} \circ \frac{u_{i,j} - u_{i-1,j}}{h_1} \circ \mathbf{H}_{\{2\}} \;+\; \mathbf{H}_{\{1\}}^{-1} \circ \frac{u_{i,j} - u_{i,j-1}}{h_2} \circ \mathbf{H}_{\{1\}} \;=\; 0 \tag{5}$$

First we test the rate of convergence for smooth data. We choose a sine wave as initial data and choose the Dirichlet data on the inflow boundary such that the solution in space-time is $u = cos(\pi(x-t))$ on the square $[-1, 1] \times [0, 2]$, see Figure 2. The linear equation system is solved with an iterative Krylov method.

Here, the convergence rates of the error in discrete L_1, L_2 and L_∞ norms, that is the ratio of $\varepsilon_{2h}/\varepsilon_h$ converges to the factor two for the regular gird and the

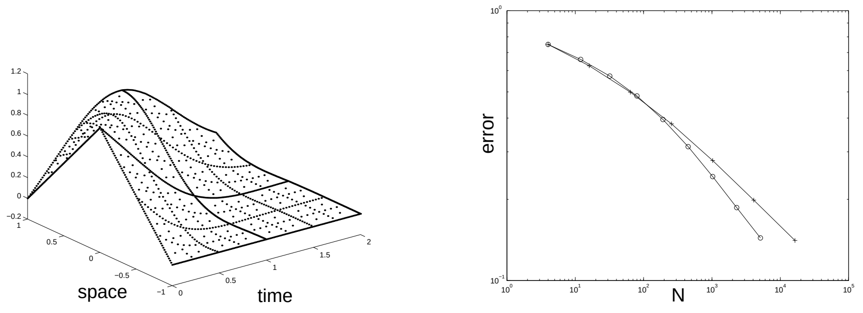


FIGURE 3. Hat function (left); convergence history: L_∞ error vs. $N_{storage}$, '+' regular grid and 'o' sparse grid.

$2/h$	32	64	128	256	512
regular	1.3675	1.4021	1.4112		
sparse	1.2226	1.2606	1.2909	1.3030	1.2960

TABLE 4. L_∞ convergence rates on regular and on sparse grids, hat function.

sparse grid discretization, see Table 3. Hence both methods have the numerical order of convergence $p = 1$, which we expected for a first order scheme. Sparse grid Galerkin schemes often show a slightly slower convergence of $\mathcal{O}(h|logh|)$, which is not the case for finite differences, see also [9].

Convergence histories are given in Figure 2, demonstrating the rapid convergence of the sparse grid solution. In the ε-N diagram the complexity of the sparse grid can be seen. The steeper slope of the sparse grid approach is comparable to the performance of a second order method. Note that $N_{storage}$ accounts for the nodes in space-time.

The performance of sparse grids depends on the smoothness of the function to be approximated. The sine-wave example was analytic. In order to test the sparse grid method for data which does not match the smoothness requirements, we consider a C^0 hat function as initial data $u|_{t=0}$. The inflow data is set to zero. The hat function is advected towards the outflow boundary, see Figure 3. The solution is continuous and contains a jump in the first derivative. The jump is not resolved at each time step on a sparse grid, because there are not enough nodes on each time-slice. The interpolation procedure inherent to the sparse grid discretization introduces artificial viscosity in these cases.

The convergence rates in Table 4 indicate the numerical order of convergence $p = 1/2$ for the regular grid and an even lower order for the sparse grid solution. However, the complexity of the sparse grid algorithm still is better than of the regular grid in Figure 3, due to the lower number of nodes.

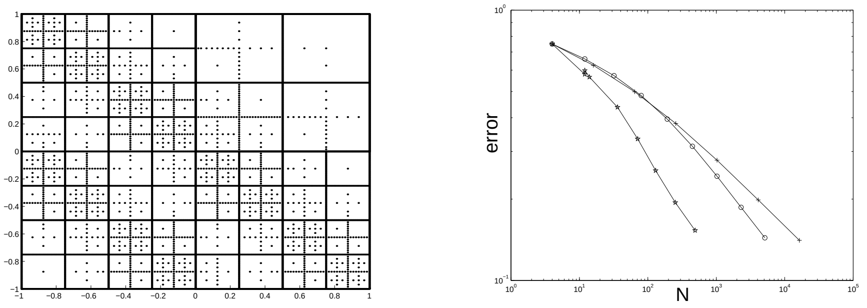


FIGURE 4. Adapted grid (left); convergence history: L_∞ error vs. $N_{storage}$, '+' regular grid, 'o' sparse grid and '$*$' adaptive strategy.

2.2. Adaptive space-time schemes

The performance of the sparse grid for the hat-function example degraded because of the jumps in the first derivative of the solution. Similar effects can be observed for many other discretizations. A common method to fix this is adaptivity.

The sparse grid is refined locally, to resolve such jumps. We employ an error indicator on the space-time mesh which indicates nodes with large errors. It is based on the absolute value of the associated hierarchical basis coefficient, see [3], and is related to residual based error indicators. The grid is refined in the neighborhood of nodes with large errors, inserting the hierarchical sons of a node. Locally a sparse grid of level $l+1$ is obtained. A new solution is computed on the refined grid. The cycle of error indication, grid refinement and solution is iterated until a final error tolerance is matched, for further details see [2, 3].

A solution algorithm based on adaptive grid refinement is applied to the previous non-smooth test example. An adapted grid and the corresponding convergence history are depicted in Figure 4.

The complexity of the adaptive sparse grid in this case is similar to the second order complexity of the standard sparse grid in Figure 2. This means that, with the help of the adaptive grid refinement process, the original sparse grid performance is regained. The adaptive refinement does work as expected. However, the solve-refine cycle adds some overhead to the run time of the solution process.

Adaptive grid refinement added some features to the solution algorithm for time dependent problems: There is only one error indicator operating in the space-time domain, instead of several indicators, which operate separately in space and time and are coupled through CFL type of conditions. Grid refinement now enhances local space-resolution and at the same time introduces local time-stepping. However, due to the discretization implicit in time, there is no CFL condition.

3. Time-Stepping Schemes

Now we briefly look at the construction of time-stepping schemes based on sparse grids in space. We consider central difference schemes which are explicit in time in contrast to Godunov schemes. This means we are able to avoid Riemann solvers and can concentrate on finite difference stencils. We start with the first order prototype of all central scheme, the Lax-Friedrichs scheme [7]. We choose the non-staggered version

$$u_i^{t+1} \;=\; \frac{1}{2}(u_{i-1}^t + u_{i+1}^t) - \lambda\left(f(u_{i+1}^t) - f(u_{i-1}^t)\right) \tag{6}$$

with flux f, which can be extended to systems of equations easily. The construction of a sparse grid version of (6) requires a formulation that is invariant with respect to the hierarchical basis transformations $\mathbf{H}_{I\setminus\{i\}}$, e.g. a basis free formulation. In order to achieve this, we define the field

$$f_i^t \;:=\; f(u_i^t)$$

for a given solution u^t in nodal basis. Equation (6) can be re-written as

$$u_i^{t+1} \;=\; \frac{1}{2}(u_{i-1}^t + u_{i+1}^t) - \lambda_i(f_{i+1}^t - f_{i-1}^t) \tag{7}$$

which is linear in u^t and in f^t. Hence equation (7) holds in any basis, especially in the hierarchical basis. As in the linear case, see equation (1), we apply (first order) dimensional splitting. We obtain the following algorithm for a single time step:

$$\begin{array}{rcl}
\hat{u}_{i,j}^t & = & \mathbf{H}_{\{2\}}\; u_{i,j}^t \\
\hat{f}_{i,j}^t & = & \mathbf{H}_{\{2\}}\; f(u_{i,j}^t) \\
\hat{u}_{i,j}^{t+1/2} & = & \frac{1}{2}(\hat{u}_{i-1,j}^t + \hat{u}_{i+1,j}^t) - \frac{\Delta t}{\Delta 2x_{i,j}}(\hat{f}_{i+1,j}^t - \hat{f}_{i-1,j}^t) \\
u_{i,j}^{t+1/2} & = & \mathbf{H}_{\{2\}}^{-1}\; \hat{u}_{i,j}^{t+1/2} \\
\tilde{u}_{i,j}^{t+1/2} & = & \mathbf{H}_{\{1\}}\; u_{i,j}^{t+1/2} \\
\tilde{g}_{i,j}^{t+1/2} & = & \mathbf{H}_{\{1\}}\; g(u_{i,j}^{t+1/2}) \\
\tilde{u}_{i,j}^{t+1} & = & \frac{1}{2}(\tilde{u}_{i,j-1}^{t+1/2} + \tilde{u}_{i,j+1}^{t+1/2}) - \frac{\Delta t}{\Delta 2y_{i,j}}(\tilde{g}_{i,j+1}^{t+1/2} - \tilde{g}_{i,j-1}^{t+1/2}) \\
u_{i,j}^{t+1} & = & \mathbf{H}_{\{1\}}^{-1}\; \tilde{u}_{i,j}^{t+1}
\end{array} \tag{8}$$

Note that the hierarchical basis transform has to be applied to both the fields u^t and f^t and to $u^{t+1/2}$ and $g^{t+1/2}$.

3.1. Numerical experiments

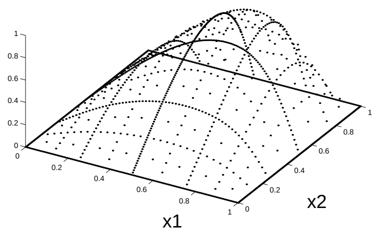
We consider Burgers' equation in two space dimensions.

$$u_t \;+\; \frac{1}{2}\nabla(u^2 \cdot \mathbb{1}) = 0 \tag{9}$$

with smooth initial data $u|_{t=0} = sin\pi x\; sin\pi y$. We compute until $t_0 = .2$ which is before the shock at $t = 1/\pi$, with a CFL number $= 1/2$. This means that the solution remains smooth, see Figure 5 (left). We use the usual sparse grid without adaptive refinement. The numerical convergence rates are depicted in Table 5.

$1/h$	8	16	32	64	128	256	512
regular	1.4559	1.6623	1.8927				
sparse	1.2622	1.2575	1.5461	1.5009	1.4357	1.4422	1.4662

TABLE 5. L_∞ convergence rates on regular and sparse grids for CFL number $= 1/2$, sine example.



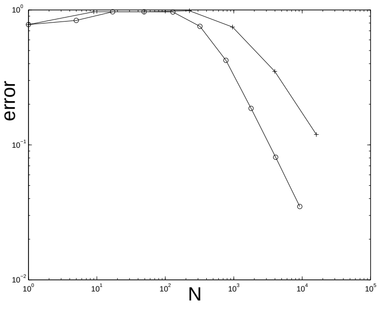


FIGURE 5. Solution for fixed Δt, sine example (left); convergence history: L_∞ error vs. $N_{storage}$ (space), '+' regular grid and 'o' sparse grid for fixed Δt.

We obtain a numerical order of convergence of $p = 1$ for the regular grid case and a rate of about $p = 1/2$ for the sparse grid. This means that both methods seem to have comparable complexity. However, for the same number of nodes N, the sparse grid solution uses a finer h, which implies a smaller time step Δt and a better resolution of the non-linearity.

In order to compare both algorithms, we fix the time step Δt instead of the CFL number in the following. Choosing $\Delta t = 1/1024$, we obtain the convergence history and the solution of Figure 5.

The sparse grid algorithm shows a better complexity and a better complexity order (slope) than the regular grid algorithm. This is due to the smooth solution, which can be resolved on the standard sparse grid. We expect an adaptive algorithm, which refines and coarsens the sparse grid in space and resolves the boundary layer at $x_i = 1$ to perform even better.

We again consider Burgers' equation (9) to test less smooth data, now with non-smooth initial data $u|_{t=0}$ chosen as a jump $t_0 = .5$. The jump is advected with constant velocity towards the corner $(1, 1)$. We fix the time step $\Delta t = 1/256$. The solution and the convergence rates are depicted in Figure 6.

Both regular grid and sparse grid show slow convergence. The sparse grid demonstrates a better convergence than regular grid algorithm. The sparse grid is able to resolve the jump at specific locations such as short binary fractions $x_i = \mathbb{Z}$, $1/2\mathbb{Z}$, $1/4\mathbb{Z}$, ... It cannot resolve the jump at locations in between and uses linear interpolation through the hierarchical basis. Hence the time evolution algorithm

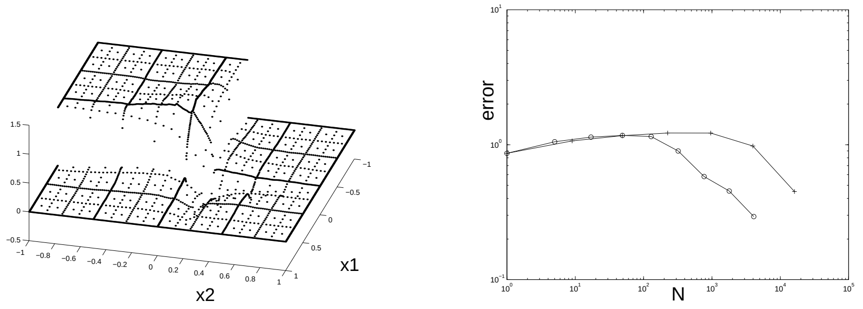


FIGURE 6. Solution for fixed Δt, jump example (left); convergence history: L_∞ error vs. $N_{storage}$, '+' regular grid and 'o' sparse grid for fixed Δt.

creates slight oscillations. In our further experiments, even a piecewise constant basis, as proposed in [5] did not improve the performance significantly.

The initial data is no longer C^0, but discontinuous. However, we still use the linear hierarchical basis and C^0 approximations. We expect an adaptive grid in space, which resolves the jumps, to improve the convergence substantially.

3.2. Higher order central differences

As a brief outlook we propose a second order sparse grid central differencing scheme based on the scheme by Nessyahu-Tadmor [8]. We define the field f and a field ${u'}_i^t$ of reconstructed slopes u', which can be obtained with a slope limiter in nodal basis.

$$f_i^t := f\left(u_i^t - \frac{\lambda_i}{2} f'(u_i^t)\right)$$

The scheme can be written as

$$u_i^{t+1} = \frac{1}{2}(u_{i-1}^t + u_{i+1}^t) - \lambda_i (f_{i+1}^t - f_{i-1}^t) - \frac{1}{8}({u'}_{i+1}^t - {u'}_{i-1}^t) \qquad (10)$$

in any basis, because it is linear in u^t, in ${u'}^t$ and in f^t. A second order sparse grid version can be obtained easily with second order Strang-splitting. In addition adaptive grid refinement will improve the performance of the scheme for non-smooth data.

4. Conclusion

We have constructed several numerical schemes based on sparse grids: An implicit discretization in space-time, an adaptive algorithm in space-time and an explicit central differencing scheme were proposed. Sparse grids have the advantage of a lower complexity than regular grids, especially in higher dimensions. Sparse grids can be interpreted as a global higher order method. Such methods usually require regularity, which either is given by smooth data or can be substituted by suitable adaptive grid refinement. Grid refinement has been demonstrated for the sparse grid space-time discretization, where a single error indicator controlled both refinement in space and in time.

References

[1] R. Balder. *Adaptive Verfahren für elliptische und parabolische Differentialgleichungen auf dünnen Gittern.* PhD thesis, TU München, Inst. für Informatik, 1994.

[2] H.-J. Bungartz, T. Dornseifer, and C. Zenger. Tensor product approximation spaces for the efficient numerical solution of partial differential equations. In *Proc. Int. Workshop on Scientific Computations*, Konya, 1996. Nova Science Publishers, Inc. to appear.

[3] M. Griebel. Adaptive sparse grid multilevel methods for elliptic PDEs based on finite differences. In *Proc. Large Scale Scientific Computations, Varna, Bulgaria.* Vieweg, 1998.

[4] M. Griebel, M. Schneider, and C. Zenger. A combination technique for the solution of sparse grid problems. In P. de Groen and R. Beauwens, editors, *Iterative Methods in Linear Algebra*, pages 263–281. IMACS, Elsevier, 1992.

[5] A. Harten. Multi-resolution representation of data: A general framework. *SIAM J. Numer. Anal.*, 33:1205–1256, 1995.

[6] F. Kupka. *Sparse Grid Spectral Methods for the Numerical Solution of Partial Differential Equations with Periodic Boundary Conditions.* PhD thesis, Universität Wien, Inst. für Math., 1997.

[7] R. J. LeVeque. *Numerical Methods for Conservation Laws.* Birkhäuser, Basel, 1992.

[8] H. Nessyahu and E. Tadmor. Non-oscillatory central differencing for hyperbolic conservation laws. *J. Comput. Phys.*, 87:408–448, 1990.

[9] Th. Schiekofer. *Die Methode der Finiten Differenzen auf Dünnen Gittern zur adaptiven Multilevel-Lösung partieller Differentialgleichungen.* PhD thesis, Universität Bonn, Inst. für Angew. Math., 1998. to appear.

[10] S. Vandewalle. *Parallel Multigrid Waveform Relaxation for Parabolic Problems.* Teubner, Stuttgart, 1992.

[11] H. Yserentant. On the multilevel splitting of finite element spaces. *Numer. Math.*, 49:379–412, 1986.

Institute for Applied Mathematics,
University Bonn,
Wegelerstr. 6
D-53115 Bonn, Germany
E-mail address: `griebel@iam.uni-bonn.de`
E-mail address: `zumbusch@iam.uni-bonn.de`