TerraNeo—Mantle Convection Beyond a Trillion Degrees of Freedom

Simon Bauer, Hans-Peter Bunge, Daniel Drzisga, Siavash Ghelichkhan, Markus Huber, Nils Kohl, Marcus Mohr, Ulrich Rüde, Dominik Thönnies, and Barbara Wohlmuth

Abstract Simulation of mantle convection on planetary scales is considered a grand-challenge application even in the exascale era. The reason being the enormous spatial and temporal scales that must be resolved in the computation as well as the complexities of realistic models and the large parameter uncertainties that need to be handled by advanced numerical methods. This contribution reports on the TerraNeo project which delivered novel matrix-free geometric multigrid solvers for the Stokes system that forms the core of mantle convection models. In TerraNeo the hierarchical hybrid grids paradigm was employed to demonstrate that scalability can be achieved when solving the Stokes system with more than ten trillion ($1.1 \cdot 10^{13}$) degrees of freedom even on present-day peta-scale supercomputers. Novel concepts

---

S. Bauer · H.-P. Bunge
Ludwig-Maximilians-Universität München, Munich, Germany

D. Drzisga · S. Ghelichkhan · M. Huber
Technical University of Munich, München, Germany

N. Kohl
University of Erlangen–Nuremberg, Erlangen, Germany
e-mail: nils.kohl@fau.de

M. Mohr
Ludwig-Maximilians-Universität München, Munich, Germany
e-mail: marcus.mohr@lmu.de

U. Rüde (✉)
University of Erlangen–Nuremberg, Erlangen, Germany

Centre Européen de Recherche et de Formation Avancée en Calcul Scientifique (CERFACS),
Toulouse, France
e-mail: ulrich.ruede@fau.de

D. Thönnies
University of Erlangen–Nuremberg, Erlangen, Germany

B. Wohlmuth
Technical University of Munich, München, Germany
e-mail: barbara.wohlmuth@ma.tum.de

were developed to ensure resilience of algorithms even in case of hard faults and new scheduling algorithms proposed for ensemble runs arising in Multilevel Monte Carlo algorithms for uncertainty quantification. The prototype framework was used to investigate geodynamic questions such as high velocity asthenospheric channels and dynamic topography and to perform adjoint inversions. We also describe the redesign of our software to support more advanced discretizations, adaptivity, and highly asynchronous execution while ensuring sustainability and flexibility for future extensions.

1 Introduction and Motivation

Geodynamics

Mantle convection is a critical component of the Earth system. Although composed of rocks, the Earth's mantle behaves like a highly viscous fluid on geological time-scales. Its motion is driven by internal heating due to radioactive decay and by substantial heating from below. The latter stems from the release of primordial heat stored in the core from the time of the planet's accretion. The mantle convection currents are largely responsible for many of Earth's surface tectonic features, forming mountain chains and oceanic trenches through plate tectonics, and contributing significantly to the accumulation of stresses released in inter-plate earthquakes. Hence, a thorough quantitative understanding of mantle convection is indispensable to gain further insight into these processes. However, besides such sort of fundamental questions, mantle convection does also have direct influence on some societal and commercial issues. Viscous stresses caused by up- and downwellings in the mantle induce dynamic topography, i.e. they lead to elevation or depression of parts of Earth's surface. Reconstructing the latter and the associated sea-levels back in time is crucial for localisation of oil-reservoirs and the determination of future sea-level rises caused by climate change.

Although the basic equations for describing the mantle convection process are not in question, resulting from the force balance between viscous and buoyancy forces and conservation of mass and energy, key system parameters, such as the buoyancies and viscosities, remain poorly known. In particular the rheology of the mantle, which is a fundamental input parameter for geodynamic models, is not well known. Studies based on modeling the geoid e.g. [87], the convective planform e.g. [21], glacial isostatic adjustment e.g. [72, 80], true polar wander e.g. [82, 86, 89] and plate motion changes e.g. [57] consistently point to the need for a significant viscosity increase between the upper and the lower mantle. But the precise form of the viscosity profile remains uncertain. Commonly the viscosity profiles display a peak in the mid lower mantle [81, 85], or involve a rheology contrast located around 1000 km depth [93], or favor an asthenosphere with a strong viscosity reduction to achieve high flow velocities and stress amplification in the sublithospheric mantle [51, 102]. Geodynamic arguments on the uncertainties and trade-offs in the viscosity profile of the upper mantle have been summarized recently by Richards and Lenardic [88].

Fig. 1 Velocity flowlines under the Himalaya mountain range; snapshot from a global convection model simulated with hierarchical hybrid grids

Since the mantle is not directly accessible to observations and laboratory experiments can hardly reproduce the relevant temperatures, pressures, and time-scales, computer simulation is vital for studying mantle convection. An example is given in Fig. 1. One of the challenges is the enormous spatial scales that must be resolved. While the mantle has a thickness of roughly 3000 km, important features, such as subducting slabs might only have a width of several kilometers. Additionally, realistic simulations will require a very large number of time-steps. Furthermore, fully realistic models of the mantle must deal with many complexities such as non-linear viscosity models, phase transitions, or the treatment of the non-constant hydrostatic background density of the mantle, see e.g. [50].

TERRANEO

TERRANEO¹ aims to design and realize a new community software framework for extreme scale Earth Mantle simulations. We have a special constellation where groups from Geophysics, Numerical Mathematics, and High Performance Computing collaborate towards a unique co-design effort. The first 3-year funding period has already been summarized in [7]. Initially, the team had focussed on fundamental mathematical questions together with general scalability and performance issues, as documented in [38, 40–43]. In particular it could be shown that even with peta-scale class machines, computations are possible that resolve the global Earth Mantle with about 1 km resolution, requiring the solution of indefinite sparse linear systems with more than 10^{12} degrees of freedom. These very large computations and scaling experiments were performed with a prototype software that was implemented using the pre-existing hierarchical hybrid grids (HHG) multigrid library [12, 13]. This step was necessary to gain experience with parallel multigrid solvers for

¹<http://terraneo.fau.de>.

indefinite problems and the specific difficulties arising in Earth Mantle models. While this approach led to quick first results, the prototype character of HHG implied several fundamental limitations. Therefore, the second funding period is being used for a fundamental redesign of HHG under the new acronym HYTeG for Hybrid Tetrahedral Grids. The goal is to leverage the lessons learned with HHG for the design of a new, extremely fast, but also flexible, extensible, and sustainable Geophysics software framework.

Additionally, research on several exascale topics was conducted in both funding periods. These include resilience, inverse problems, and uncertainty quantification. Detailed results on these topics, as well as on the new HYTeG software architecture, will be reported in the following sections of this report.

Multigrid Methods

Multigrid methods belong to the set of fastest solvers for sparse linear systems that arise from the discretization of PDEs and are, thus, of central importance for exascale research. Their invention and their popularization e.g. in [17] constitute one of the major breakthroughs in numerical mathematics and computational science. To illustrate their relevance for exascale computing, we summarize here a thought experiment from [90].

Assuming that systems with $N = 10^{12}$, i.e. a trillion degrees of freedom (DoF) would be solved by classical Gaussian elimination, this would require the astronomical number of operations of $2/3N^3 = 2/3 \times 10^{36}$. Better elimination based algorithms can exploit the sparse matrix structure. A typical method of this class, such as *nested dissection* would still require around 10^{24} operations. For such a work load, a modern, fast PC with a performance of 100 GFlops would need more than 100,000 years of compute time. Note that solving one such system is still only worth one time step of many. This line of thought may be seen as just another argument why Earth Mantle Convection is a grand challenge problem and why e.g. the authors of [23] write:

A uniform discretization of the mantle at for instance 1 km resolution would result in meshes with nearly a trillion elements, which is far beyond the capacity of the largest available supercomputers.

It is important to realize that parallel computing alone does not solve the problem. Even the fastest German supercomputer, currently SuperMUC-NG², would still need longer than a year of compute time (if it did not run out of memory before) to execute 10^{24} operations.

In the TERRANE0 project we have demonstrated, and we will report below, that a well designed massively parallel multigrid method is indeed capable of solving such large systems with a trillion unknowns in compute times in the order of seconds. This is fundamentally based on their fast convergence, for many types of problems, that leads to asymptotically optimal complexity solvers, when it is combined with a nested iteration in the form of a so called full multigrid method [17, 41]. Based on this, it comes as little surprise that multigrid methods are employed in several of the

---

²<https://doku.lrz.de/display/PUBLIC/SuperMUC-NG>.

SPPEXA projects, such as e.g. [3, 4, 24, 69, 75] with excellent success for a wide variety of applications.

In the literature on multigrid methods of the past two decades, much attention has been given to so called algebraic multigrid methods (AMG) that do not operate on a mesh structure, but that attempt to construct the necessary hierarchy based on analyzing the sparse system matrix. These methods can exhibit excellent parallel scalability [3] and are often used as components within other parallel solvers, such as domain decomposition methods [62]. AMG methods have several advantages, most importantly that they can be interfaced to an application via classical sparse matrix technology. They can also save the application developer from worrying about the necessary solution process and many numerical analysts who design novel discretizations have taken this perspective. However, AMG methods work only well for certain types of systems, and consequently users of these new discretization techniques now find themselves being trapped with linear systems for which no efficient parallel algorithms are available, yet. Few methods devised in the applied mathematics community can demonstrate even solving systems with 10^9 degrees of freedom, falling three or four orders behind of what we can demonstrate here.

Additionally, the convenience of algebraic multigrid also comes at the price of a loss in performance. Algebraic multigrid methods have only a limited scope of applicability, and additionally they often lose an order of magnitude in performance in their expensive setup phase. Of course they are inherently also not *matrix-free*. As we will discuss below, matrix-free methods are essential to reach maximal performance. Geometric multigrid methods can be realized as matrix-free algorithms, potentially leading to another order of magnitude in performance improvement. For this reason we have invested heavily in new matrix-free methods [8, 9, 11], similar to other SPPEXA projects [5, 67].

The price of using geometric multigrid lies in the more complex algorithms which often have to be tailored carefully to each specific application, and the significantly more complex interfaces that are needed between the other software components and the solver. This in turn creates the need for new software technologies as are being developed in the HYTEG framework.

2 Basic Ideas and Concepts

As mentioned above the mathematical-physical model of mantle convection is based on the force balance between viscous and buoyancy forces and conservation of mass and energy. The resulting general system of equations is e.g. given in [7, 105]. Expressing these in terms of the central quantities of interest, i.e. velocity, pressure and temperature results in a coupled system composed of a stationary Stokes component and a time-dependent equation for the temperature. The former constitutes the most expensive part in these kinds of simulations and we will, thus, concentrate on this aspect mostly in the following. In the case of the Boussinesq

Table 1 Physical quantities and their representing symbols

R_0	Earth's radius	ρ_0	Reference density	g	Gravitational acceleration
η	Dynamic viscosity	α	Thermal expansivity	κ	Heat conductivity
ΔT	Temperature difference between core-mantle-boundary and surface				

approximation the Stokes part is given in non-dimensional form by

$$-\operatorname{div}(\eta(\nabla \mathbf{u} + (\nabla \mathbf{u})^\top)) + \nabla p = -Ra \, T \, \mathbf{e}_r \quad (1)$$

$$\operatorname{div}(\mathbf{u}) = 0 \quad (2)$$

with dimensionless velocity $\mathbf{u}$, pressure p and the unit vector in radial direction $\mathbf{e}_r$. The Rayleigh number Ra describes the vigor of the convection and is given by

$$Ra = \frac{\alpha g \rho_0 \Delta T R_0^3}{\kappa \eta_r}.$$

A description of the physical quantities represented by the symbols is given in Table 1. The finite element discretization that we consider for the stationary equations is backed by the concept of HHG. Initially, the domain is partitioned into an unstructured mesh of tetrahedra. In a second step, each tetrahedron is iteratively, uniformly refined in a way that a block-structured, tetrahedral mesh is created [12, 13, 40]. This refinement strategy results in a grid hierarchy $\mathcal{T} := \{\mathcal{T}_\ell, \ell = 0, 1, \dots, L\}$ that allows for a canonical implementation of geometric multigrid methods.

The computational domain is distributed to the parallel processes by means of the unstructured coarse grid elements. To realize efficient communication procedures, the mesh is enhanced by so-called *interface primitives*. For each shared face, edge, and vertex an additional macro-primitive is allocated in the data structure. Together with the macro-tetrahedra, each primitive is assigned to one parallel process. Starting from a two times refined mesh, each macro-primitive has at least one inner vertex.

We employ conforming finite elements on the domain Ω and define the function spaces of globally continuous, piecewise polynomial functions with polynomial degree d on level ℓ as

$$S_\ell^d(\Omega) := \{v \in C(\overline{\Omega}) : v|_T \in P_d(T), \forall T \in \mathcal{T}_\ell\}.$$

For the majority of our experiments and simulations we implement a stabilized P1-P1 discretization for the Stokes equation, using the velocity and pressure finite element spaces

$$\mathbf{V}_\ell \times Q_\ell := \left[S_\ell^1(\Omega) \cap H_0^1(\Omega) \right]^3 \times S_\ell^1(\Omega) \cap L_0^2(\Omega).$$

Here, $L_0^2(\Omega) := \{q \in L^2(\Omega) : \langle q, 1 \rangle_\Omega = 0\}$ with $\langle \cdot, \cdot \rangle_\Omega$ being the inner product in $L^2(\Omega)$. The P1-P1 element pairing does not fulfill the LBB-condition and we employ a PSPG stabilization [19, 33]. It follows the standard weak formulation for the discretized Stokes problem: find $(\mathbf{u}_\ell, p_\ell) \in \mathbf{V}_\ell \times Q_\ell$ such that

$$\begin{aligned} a(\mathbf{u}_\ell, \mathbf{v}_\ell) + b(\mathbf{v}_\ell, p_\ell) &= f(\mathbf{v}_\ell) & \forall \mathbf{v}_\ell \in \mathbf{V}_\ell, \\ b(\mathbf{u}_\ell, q_\ell) - c_\ell(q_\ell, p_\ell) &= g_\ell(q_\ell) & \forall q_\ell \in Q_\ell, \end{aligned} \quad (3)$$

where

$$\begin{aligned} a(\mathbf{u}, \mathbf{v}) &:= 2\langle \eta D(\mathbf{u}), D(\mathbf{v}) \rangle_\Omega \\ b(\mathbf{u}, q) &:= -\langle \operatorname{div} \mathbf{u}, q \rangle_\Omega \\ f(\mathbf{v}) &:= \langle \mathbf{f}, \mathbf{v} \rangle_\Omega, \end{aligned}$$

for all $\mathbf{u}, \mathbf{v} \in H_0^1(\Omega)^3$ and $q \in L_0^2(\Omega)$. The stabilization terms are defined as

$$\begin{aligned} c_\ell(q_\ell, p_\ell) &:= \sum_{T \in \mathcal{T}_\ell} \frac{1}{12} \left(\int_T dx \right)^{1/3} \langle \nabla p_\ell, \nabla q_\ell \rangle_T \\ g_\ell(q_\ell) &:= - \sum_{T \in \mathcal{T}_\ell} \frac{1}{12} \left(\int_T dx \right)^{1/3} \langle \mathbf{f}, \nabla q_\ell \rangle_T. \end{aligned}$$

The discrete formulation of (3) can then be expressed as the system of linear equations

$$\mathcal{K} \begin{pmatrix} \mathbf{u} \\ \mathbf{p} \end{pmatrix} := \begin{pmatrix} A & B^\top \\ B & -C \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \mathbf{p} \end{pmatrix} = \begin{pmatrix} \mathbf{f} \\ \mathbf{g} \end{pmatrix}. \quad (4)$$

A description of the general form of the temperature equation and our work on its discretization and time-stepping can be found in [7].

3 Summary of Project Results

3.1 Efficiency of Solvers and Software

A rigorous quantitative performance analysis lies at the heart of systematic research in large scale scientific computing. The systematic analysis of performance is central to the research agenda of computational science [91], since it defines how successful computational models are. In this sense, performance analysis here means more than measuring the speedup of a parallel solver or studying its weak and strong scaling

properties. Beyond this, modern scientific computing must attempt to quantify numerical cost in metrics that are independent of the run-time of an arbitrary reference implementation. Most importantly, the numerical cost must be set in relation to what is numerically achieved, i.e. the accuracy that is delivered by a specific computation.

As one step in this direction, we extend Achi Brandt's notion of *textbook multigrid efficiency* (TME) to massively parallel algorithms in [41]. Using the finite element based prototype multigrid implementation of the HHG library, we have employed the TME paradigm for scalar linear equations with constant and varying coefficients as well as to linear systems with saddle-point structure. A core step is then to extend the idea of TME to the parallel setting in a way that is adapted to the parallel architecture under consideration. To this end, we develop a new characterization of a work unit (WU) in an architecture-aware fashion by taking into account modern performance modeling techniques, in particular the standard Roofline model and the more advanced ECM model. The newly introduced parallel TME measure is studied in [41] with large-scale computations and for solving problems with up to 200 billion unknowns.

3.2 Reducing Complexity in Models and Algorithms

When striving for optimal efficiency, it is essential to choose models and discretization schemes whose incurred computational cost is as low as possible. Within TERRANEO we have therefore analyzed and improved the discretization schemes under study. A restricting factor was initially that only simple conforming discretizations are feasible in the prototype HHG library. E.g. discontinuous Galerkin discretizations, as in [6, 69], were not yet feasible in HHG. In particular, buoyancy-driven flow models thus demand a careful treatment of the mass-balance equation to avoid spurious source and sink terms in the non-linear coupling between flow and transport. In the context of finite-elements, it is therefore commonly proposed to employ sufficiently rich pressure spaces, containing piecewise constant shape functions to obtain local or even strong mass-conservation. In three-dimensional computations, this usually requires nonconforming approaches, special meshes or higher order velocities, which make these schemes prohibitively expensive for some applications and complicate the implementation into legacy code. In [39], we propose and analyze a lean and conservatively coupled scheme based on standard stabilized linear equal-order finite elements for the Stokes part and vertex-centered finite volumes for the energy equation. We show that in a weak mass-balance it is possible to recover exact conservation properties by a local flux-correction which can be computed efficiently on the control volume boundaries of the transport mesh. Furthermore, we discuss implementation aspects and demonstrate the effectiveness of the flux-correction by different two- and three-dimensional examples which are motivated by geophysical applications in [101].

In the special case of constant viscosity the Stokes system can be cast into different formulations by exploiting the incompressibility constraint. For instance the strain in the weak formulation can be replaced by the gradient to decouple the velocity components in the different coordinate directions. Thus the discretization of the simplified problem leads to fewer nonzero entries in the stiffness matrix. This is of particular interest in large scale simulations where a reduced memory footprint and accordingly reduced bandwidth requirement can help to significantly accelerate the computations. In the case of a piecewise constant viscosity, as it typically arises in multi-phase flows, or when the boundary conditions involve traction, the situation is more complex, and the cross derivatives in the original Stokes system must be treated with care. A naive application of the standard vectorial Laplacian results in a physically incorrect solution, while formulations based on the strain increase the computational effort everywhere, even when the inconsistencies arise only from an incorrect treatment in a small fraction of the computational domain. In [55], we present a new approach that is consistent with the strain-based formulation and preserves the decoupling advantages of the gradient-based formulation in iso-viscous subdomains. The modification is equivalent to locally changing the discretization stencils, hence the more expensive discretization is restricted to a lower dimensional interface, making the additional computational cost asymptotically negligible. We demonstrate the consistency and convergence properties of the new method and show that in a massively parallel setup, the multigrid solution of the resulting discrete systems is faster than for the classical strain-based formulation.

3.3 Stokes Solvers and Performance

3.3.1 Multigrid Approaches for the Stokes System

In this subsection we briefly review different classes of solvers for Stokes type systems involving a multigrid component. One characteristic feature is the indefinite structure and thus special care in the design of the solver is required. Numerical experiments and performance studies were originally performed with the HHG software and are thus mostly limited to stabilized conforming P1-P1 discretizations. We point here also to the possibility to generate such parallel solvers automatically and achieve similarly good scalability and efficiency results, as demonstrated in the ExaStencils project [70, 75]. A comprehensive summary of the supercomputers used for the simulations conducted in this project is listed in Table 2.

In TERRANEO three classes of solvers we consider are a preconditioned Krylov method for the indefinite system, a preconditioned Krylov method for the positive definite pressure based Schur complement and a monolithic (sometimes also called an *all-at-once*) multigrid solver for the indefinite system. While in the first two approaches the multigrid solver is only applied to the velocity component, the monolithic multigrid scheme works simultaneously on the velocity and the pressure

Table 2 Characteristics of the different supercomputers used for simulations presented in this publication

	SuperMUC Phase 1 (Thin nodes)	SuperMUC Phase 2	Juqueen	Hazel Hen
Operation	2012–2018	2015–present	2012–2018	2015–present
# nodes	9216	3072	28,672	7712
CPU	Intel Sandy Bridge E5-2580 8 Core	Intel Haswell E5-2697v3 14 Core	IBM PowerPC A2 16 Core	Intel Haswell E5-2680v3 12 Core
CPU frequency (GHz)	2.7	2.6	1.6	2.5
# total cores	147,456	80,016	458,752	185,088
Interconnect	Infiniband FDR10	Infiniband FDR14	5D Torus	Aries
Total memory (TByte)	288	194	448	987
Linpac (PFlop/s)	2.897	2.814	5.0	7.42

Table 3 Exploring the limits of the monolithic multigrid solver: weak scaling on JUQUEEN

Nodes	Threads	DoFs	Iter	Time [s]	Time [s] w/o coarse grid solver
5	80	2.7×10^9	10	685.88	678.77
40	640	2.1×10^{10}	10	703.69	686.24
320	5120	1.2×10^{11}	10	741.86	709.88
2560	40,960	1.7×10^{12}	9	720.24	671.63
20,480	327,680	1.1×10^{13}	9	776.09	681.91

component. A systematic quantitative performance study of all three approaches on modern peta-scale systems is given in [43]. More than 750,000 parallel threads are used in the largest simulation runs. Our main finding is that the all-at-once multigrid scheme outperforms the two alternatives. It does not only show the smallest memory footprint, but also results in the shortest compute time. More than 10 trillion³ unknowns ($> 10^{13}$) have been solved for on a Blue Gene/Q system, see Table 3.

For such huge systems a sophisticated matrix-free code design with minimized memory traffic is a must. Although the use of GMRES type Krylov methods for the indefinite system is quite popular due to its robustness, it is not an option for us. The non-optimal memory requirement and compute time per iteration before the next restart slows down the overall performance drastically and restricts the feasible system size. To test the flexibility of the solver, we consider not only a box as geometry but also a thick spherical shell motivated by our application, a pipe filled with spherical obstacles of different diameters and a simplified artery, see Fig. 2. The physical system under consideration then ranges from a thermo-mechanically coupled system to a non-linear viscosity model of Carreau type.

³W.r.t. the short scale naming system.

Fig. 2 Left to right: concentration in a lid driven cavity type setting; streamlines and speed in a benchmark problem; streamlines for a non-linear dynamic viscosity model and stationary flow in a pipe

3.3.2 Smoothers for Indefinite Systems

There are different strategies to construct smoothers for the coupled systems. Among the popular ones are Braess–Sarazin, Vanka, and Uzawa type approaches. While the first two types consider a saddle point problem either in a global or a local form as starting point, the Uzawa type scheme is based on a factorization of the saddle point system in lower and upper triangular matrices, see [16, 77, 98, 99, 104]. The advantage of the Uzawa version is clearly a smaller FLOP count and a reduced communication traffic compared to the two mentioned alternatives. Also in contrast to the Vanka smoother where local patches have to be considered it can be applied node-wise. Let us assume that A is symmetric and positive definite and $\hat{A}$ and $\hat{S}$ satisfy

$$\hat{A} \geq A, \quad \hat{S} \geq S := CBA^{-1}B^\top$$

then the following smoothing property holds

$$\|\mathcal{KM}^\nu\| \leq \sqrt{2} \eta(\nu - 1) \|\mathcal{D}\| ,$$

where ν is the number of smoothing steps, $\|\cdot\|$ a suitable operator norm and

$$\mathcal{M} := \text{Id} - \mathcal{K}^{-1} \begin{pmatrix} \hat{A} & 0 \\ B & \hat{S} \end{pmatrix}, \quad \mathcal{D} := \begin{pmatrix} \hat{A} & 0 \\ 0 & \hat{S} \end{pmatrix}, \quad \eta(\nu) := \frac{1}{2^\nu} \binom{\nu}{\lfloor (\nu - 1)/2 \rfloor}.$$

A mathematically rigorous proof is given in [29]. Following the abstract framework provided by Larin and Reusken [73], the following identity turns out to be of crucial importance

$$\mathcal{KM}^\nu = M_1 \mathcal{KM}_s^{\nu-1} M_2$$

with suitable operators M_1 , M_2 and M_s being the error propagation operator of the standard Uzawa. We note that in comparison to earlier theoretical results [95, 106, 107], we do not require symmetry of the Uzawa type smoother and thus

Table 4 Number of iterations to reduce the residual by factor of 10^{-8} (Intel Xeon)

ν	4		6		8	
DoFs	Iter	Time [s]	Iter	Time [s]	Iter	Time [s]
1.4×10^3	13	0.10	9	0.07	7	0.06
1.4×10^4	12	0.21	9	0.18	7	0.15
1.2×10^5	12	0.61	8	0.51	6	0.44
1.0×10^6	11	2.44	8	2.33	6	2.16
8.2×10^6	11	14.54	8	14.58	6	14.03
6.6×10^7	11	102.66	7	92.09	6	101.90
5.3×10^8	10	700.38	7	693.75	6	769.34

the computational cost is reduced from two applications of the velocity smoother to one in each iteration. The achieved upper bound for the smoothing property is sufficient to guarantee level independent $\mathcal{W}$ -cycle or variable $\mathcal{V}$ -cycle results. However, in large scale applications $\mathcal{V}$ -cycle schemes are much more attractive than $\mathcal{W}$ -cycle based multigrid methods. The number of coarse solves per iteration is one in case of the $\mathcal{V}$ -cycle but grows exponentially with the depth of the multigrid hierarchy in case of a $\mathcal{W}$ -cycle. Unfortunately, there is no $\mathcal{V}$ -cycle theory for this all-at-once multigrid method available, and moreover numerical experiments show that the convergence rate deteriorates for a fixed and level independent number of smoothing steps per level and an increased level count. As a compromise a mildly variable $\mathcal{V}$ -cycle turns out to perform best. The HHG data structure and the matrix-free concept allows us, even on a standard workstation, to test the solver for systems with more than 100 millions of unknowns. The computations reported in Table 4 are performed on an Intel Xeon CPU E3-1226 v3, 3.30 GHz with 32 GB memory. For this first test only a single core is used and thus the timing does not reflect the parallel performance of the solver. For the pressure we use damped Gauss–Seidel applied to the operator C where the damping parameter is determined by a power iteration on a coarse level and not dependent on the mesh-size, but on the mesh-regularity.

While the all-at-once multigrid solver requires a special smoother, the Schur-complement formulation allows for a natural application of multigrid on the velocity component as part of an inner iteration in a preconditioned conjugate gradient method for the pressure. In case of an isoviscous flow the Schur complement is spectrally equivalent to the mass matrix and thus the condition number does not deteriorate with an increasing number of degrees of freedom. Due to its simple structure this iterative solver can be easily implemented by reusing standard software components and is thus suitable for a rigorous performance analysis. We do not report here in further detail, but an innovative performance analysis can be found in [40, 41]. Good scalability on more than half a million parallel threads is demonstrated in [42]. Together with the excellent node-level performance, this is essential for achieving high performance levels.

3.3.3 Multigrid Coarse Grid Solvers

Finally we report on results for agglomeration techniques. In the context of the matrix-free HHG implementation it is natural to use a preconditioned Krylov method as a coarse grid solver. However, in the context of very large systems this is a bottleneck since this coarse solver is non-optimal, and will thus ultimately become a limit to scalability. To improve the parallel efficiency for systems beyond a billion of unknowns, we proceed in two steps. In a first preprocessing step, the coarse grid problem is assembled in a classical fashion storing matrix entries in standard compressed row storage format. The system can then be solved by methods from the PETSc library, [79]. More specifically, a block preconditioned MINRES iteration is applied. For the velocity component we here use an AMG preconditioned conjugate gradient (CG) iteration and for the pressure a lumped mass-matrix approach. Here still the full parallel system is employed so that only small coarse grid subproblems are assigned to each parallel process. In the second step, we therefore propose to use agglomeration to reduce the number of parallel processes on the coarsest level and to reduce the communication overhead in each iteration. This becomes necessary when the ratio between communication time and compute time becomes unbalanced and when the overall coarse solve time is severely dominated by communication. In Table 5, we report on the parallel efficiency for a weak scaling study on JUQUEEN (IBM BlueGene/Q-System, 28 racks, 28,672 nodes, 458,752 cores). The number of unknowns ranges from approximately 80 million to over 100,000 million and the number of parallel threads is increased by a factor of more than 15,000. Over this large range, the parallel efficiency shows only a moderate deterioration and remains above 90% even in the largest run. This results mainly from the fact that for the largest run we reduce the number of processes on the coarsest level by a factor of eight. Using a master-slave agglomeration technique has the advantage of a short collecting and distribution phase.

In Fig. 3 we compare the naive non-optimal coarse solver with the PETSc based agglomeration strategy for the largest run, i.e. $np = 61,440$. We observe that with the naive approach roughly half of the compute time is spent in the coarse solve, although it has only a small fraction of the total number of unknowns. The situation is drastically different for the agglomeration technique in combination with the PETSc solver. Here only roughly 5% of the actual time to solve is spent in the coarse solver routines. A closer look at the ratio between MPI communication and operator computations exhibits that without agglomeration strategy a significant amount of

Table 5 Parallel efficiency for master-slave agglomeration technique and up to more than 50,000 parallel threads on JUQUEEN

np	DoF	Red.	T[s]	Coarse	Par. eff
30	8.3×10^7	1	16.284	0.043	1.00
120	3.3×10^8	1	16.426	0.050	0.99
960	2.6×10^9	1	17.084	0.171	0.95
7680	2.4×10^{10}	1	17.310	0.382	0.94
61,440	1.7×10^{11}	8	17.704	0.877	0.92

Fig. 3 Ratio of relative time to solve spent in the different multigrid components: a non-optimal Krylov based solver without agglomeration (left) and AMG preconditioned Krylov based solver with agglomeration

time is lost by simple communication. The situation is again drastically different for the agglomeration strategy. Here the communication overhead is reduced to less than 5%. We note that for a smaller system size and a moderate number of parallel threads, the difference in the time to solution for both approaches is negligible. For $n_p = 30$ the naive non-optimal coarse solver takes less than 10% of the total run-time and the communication load is less than 5% compared to the FLOP count. Moreover the savings in run-time in case of a PETSc coarse solver are limited due to the set-up phase. In conclusion, and somewhat contrary to conventional wisdom, we find that for up to ten million unknowns, a more sophisticated coarse solver does not pay off.

To show its generality, we also test the strategy on a different peta-scale system (Hazel Hen) and for a problem with a strongly varying discontinuous viscosity profile as it is relevant for geodynamical simulations, see Table 6. Here, we additionally exploit a block low-rank coarse grid solver (BLR) which uses low-rank compressions in an elimination method to devise an approximate solver [1]. The reduced parallel efficiency, compared to the setting before, does not result from the different architecture, nor the use of BLR, but mainly from the fact that one additional multigrid iteration is required.

Table 6 Parallel efficiency in a weak scaling experiment for a geodynamically relevant setting on Hazel Hen

Proc.	DOF	Iter	Time [s]		BLR ϵ	Time [s]		
	Fine		Total	Fine		Coarse Ana. & fac.	Par. eff.	
1920	2.10×10^{10}	15	78.1	77.9	10^{-3}	0.03	2.7	1.00
15,360	4.30×10^{10}	13	88.9	86.8	10^{-3}	0.22	25.0	0.93
43,200	1.70×10^{11}	14	95.5	87.0	10^{-8}	0.59	111.6	0.82

3.4 Multi-Level Monte Carlo

Due to errors in measurements, geophysical data are inevitably stochastic. Therefore, it is desirable to quantify these uncertainties when computing geophysical applications. A typical approach is to use ensemble simulations, i.e. running multiple computations with slight variations of perturbed data, and evaluating computational quantities of interest via Monte Carlo sampling. A naive sampling-based uncertainty quantification for 3D partial differential equations results in an extremely large computational complexity. More sophisticated approaches, such as multilevel Monte Carlo (MLMC), can reduce this complexity significantly. The performance can be further enhanced when the Monte Carlo sampling over several levels of mesh refinement is combined with a fast multigrid solver. In a parallel environment, however, sophisticated scheduling strategies are needed to exploit MLMC based on multigrid solvers. In [28], we explored the concurrent execution across the three layers of the MLMC method. Namely, parallelization across levels, across samples, and across the spatial grid.

An alternative way to classify these different parallel execution models is to consider the time-processor diagram, as illustrated in Fig. 4, where the scheduling of each sample Y_ℓ^i , with index $1 \leq i \leq N_\ell$ on level $0 \leq \ell \leq L$, is represented by a rectangular box with the height expressing the number of processors used. We call a parallel execution model *homogeneous bulk synchronous* if at any time in the processor diagram all tasks execute on the same level ℓ with the same number of processors. Otherwise we call it *heterogeneous bulk synchronous*.

The one-layer homogeneous strategy, as shown in Fig. 4 (left), offers no flexibility. The theoretical run-time is simply given by the sum of the time of all samples. Even if this method guarantees perfect load balancing, it will not lead to an optimal efficiency since on the coarser levels the scalability of the solver is typically significantly worse than on the finer levels. On the coarsest level we may even have less grid points than processors. Thus, we discard this approach from our list of possible options.

We, instead, consider the following two variants: Firstly, *sample synchronous homogeneous* (SaSyHom) where a synchronization step is imposed after each sequential step (see Fig. 5, left). Here statistical quantities can be updated after

Fig. 4 From left to right: illustration of homogeneous (one-layer and two-layer) and of heterogeneous bulk synchronous strategies (two-layer and three-layer)

Fig. 5 Illustration of different homogeneous scheduling strategies. Left: sample synchronous homogeneous (SaSyHom); centre: level synchronous homogeneous (LeSyHom); right: dynamic level synchronous homogeneous (DyLeSyHom)

each synchronization point. Secondly, *level synchronous homogeneous (LeSyHom)*, where each block of processors executes all samples without any synchronization (see Fig. 5, center).

In case of large run-time variations of the samples, we additionally consider dynamic variants where samples can be assigned to processors dynamically. Figure 5 (right) illustrates the DyLeSyHom strategy. Here it is essential that not all processor blocks execute the same number of sequential steps.

We compare static strategies with their dynamic variants in the case of large run-time variations on three MLMC levels, i.e. $L = 2$, with a fine grid resolution of about $1.6 \cdot 10^7$ mesh nodes. We assume a lognormal random coefficient, but in order to increase the run-time variation, we choose a greater variance $\sigma^2 = 4.0$ and $\lambda = 0.5$. As quantity of interest we employ the PDE solution u evaluated at the point $x = (0.5, 0.5, 0.5)$. All samples are computed with a FMG-2V(4,4) cycle with up to a hundred additional V(4,4) cycles until a relative residuum of 10^{-10} . Pre-computed variance estimates lead to an a priori strategy with $(N_\ell)_{\ell=0,1,2} = (27, 151, 10,765, 3792)$ samples per level. In this scenario, 8192 processors were available for the scheduler. In Table 7 we compare the required time of the LeSyHom strategy in the static and dynamic variant and see that the dynamic strategy is 6% faster than its static counterpart.

The largest uncertainty quantification scenario we considered, involved finest grids with almost 70 billion degrees of freedom, and a total number of samples beyond 10,000, most of which are computed on coarser levels. This computation was executed on the JUQUEEN supercomputer using more than 132,000 processor cores in an excellent overall compute time of about 1.5 h.

Table 7 Comparison of static and dynamic scheduling strategies

	LeSyHom	DyLeSyHom	
Level	time [s]	time [s]	Ratio
0	500	460	0.92
1	1512	1347	0.89
2	5885	5596	0.95
Total	7897	7403	0.94

3.5 Inverse Problem and Adjoint Computations

The adjoint method is a powerful technique that allows the computation of sensitivities (Fréchet derivatives) with respect to model parameters. It solves inverse problems where analytical solutions are not available or the cost to solve the associated forward problem many times is prohibitively high. In geodynamics it has been applied to the inverse problem of mantle convection—i.e., to restore past mantle flow e.g. [22], where it finds an optimal initial flow state that evolves into the present-day state of Earth’s mantle. In doing so, the adjoint method has the potential to link together diverse observations and theoretical expectations from seismology, geology, mineral physics, paleomagnetism and fluid dynamics, greatly enhancing our understanding of the solid Earth system.

Adjoint equations for mantle flow restoration have been derived for incompressible [22, 52, 59], compressible [35] and thermo-chemical mantle flow [36], and the uniqueness properties of the inverse problem have been related explicitly to the tangential component of the surface velocity field of a mantle convection model [25]. So knowledge of the latter is essential to assure convergence [100] and to obtain a small null space for the restored flow history [52]. To reduce the computational cost, we use a two-scale time step size predictor-corrector strategy to couple between Stokes and temperature similar to the one discussed in [46].

The framework was used to implement the adjoint technique. Specifically, we minimized the misfit functional

$$\chi(T) := \frac{1}{2} \int_{\Omega} \int_{[t_0, t_1]} \left(T(T_0, \mathbf{x}, t) - T_E(\mathbf{x}) \right)^2 \delta(t - t_1) dt d\mathbf{x} \quad (5)$$

that relates to the squared difference of the geodynamic model temperature and the thermal heterogeneity distribution T_E for the final state at $t = t_1$. The latter can be inferred e.g., from seismology and considerations of mantle mineralogy. We used a first order optimization approach to update the initial temperature T_0 in the descent direction ϕ in a process described by

$$T_0^{i+1}(\mathbf{x}) = T_0^i(\mathbf{x}) - \beta \phi(\mathbf{x}, t_0) \quad \text{for } i = 0, 1, \dots,$$

where β controls the correction in the descent direction and has to be sufficiently small [84]. In order to reduce the computational cost of deriving the step length, we found that fixing $\beta = 0.4$ yields sufficient results in our experiments, in agreement with [22]. The descent direction $\phi(\mathbf{x}, t_0)$ is obtained by solving the adjoint equations, which are given by Bunge et al. [22]. Solving the adjoint equations requires velocity and temperature states from the forward problem. We checkpointed these states in the forward problem such that they could be read for the adjoint iteration. In our exploration of the inverse problem, we used a similar setup as presented in [23]. We considered a $45^\circ \times 45^\circ$ part of a spherical shell domain and discretized it by a tetrahedral mesh. Applying four uniform refinement

steps, the mesh has a resolution of $2.7 \cdot 10^5$ grid points. The initial temperature field given by

$$T(\mathbf{x}) = T_0 + \exp\left(-\frac{1}{\sigma^2} \|\mathbf{x} - \mathbf{x}_0\|^2\right),$$

where $\sigma = 1/20$ determines the extent of the anomaly and $\mathbf{x}_0$ denotes the center. Note that in our setup the latter is situated $2/9 D$ below the core-mantle boundary, where D denotes relative mantle thickness, and thus lower than in [23].

3.5.1 Twin Experiment

To verify our implementation, we considered a so called *twin experiment*. In this setup a reference flow history is generated from the forward model. The final state of this reference flow is used as the terminal state that drives the geodynamic inverse problem. The reconstructed flow history is then compared against the known reference history to assess the quality of the inversion. Thus, while in a realistic geophysical setting only the final state T_E is known, in the twin experiments, the true (unreconstructed) initial T_I and final state T_E are given from the reference twin. This allows us to study the convergence of initial and final states. In addition to the misfit error (5) at the final time, we define the misfit error at the initial time by

$$m_I(T) := \frac{1}{2} \int_{\Omega} \int_{[t_0, t_1]} \left(T(\mathbf{x}, t) - T_I(\mathbf{x})\right)^2 \delta(t - t_0) dt d\mathbf{x}$$

In Fig. 6, we present the initial/final temperature (left/right) from the reference twin. Figure 7 shows isosurfaces of the reconstructed initial (top row) and final (bottom row) temperature after adjoint iterations $i = 0, 5, 10$. For the sake of simplicity, the first guess for the unknown initial temperature is taken from the final state of the reference twin. Therefore, the reconstructed final temperature for $i = 0$ is a plume that has evolved much farther than the final reference state. After solving the inverse problem with 10 iterations, the error at the final state is reduced by more than one order of magnitude, while the error at the initial state is reduced by a

Fig. 6 Left: initial temperature state, right: final temperature state of the reference twin

Fig. 7 From left to right: reconstruction of initial (top row) and final (bottom row) state: $i = 0, 5, 10$ iterations, very right: reference states

factor of 3, and we observe a good agreement between reconstructed and reference temperatures.

3.6 Matrix-Free Algorithms

Matrix-free approaches are an essential component for ultra-high resolution finite-element simulations. The reason for this is twofold. The classical finite-element workflow of assembling the global system matrix from the local element contributions and then solving the resulting linear system of equations with algorithms implemented in matrix-vector fashion leads to enormous data traffic between main memory and computational units (CPU cores). This constitutes a severe performance bottleneck. Another aspect is that the cost for holding the matrix in main memory becomes prohibitive, too. Note that for our simulations with 10^{13} DoFs, see [43] and Sect. 3.3 the solution vector alone requires 80 TByte of memory. Thus, matrix-free methods, which do not assemble the global matrix, but only provide the means to evaluate an associated matrix-vector product receive increasing attention, see e.g. [5, 20, 66–68, 78, 92]. An overview on the history and different approaches can e.g. be found in [9].

We note that the concept of hierarchical hybrid grids, from its inception on, was designed to be matrix-free. Based on its original premise that the macro mesh resolves both, geometry and material parameters, only a single discretization stencil had to be computed and stored for each macro primitive, as opposed to one for every fine mesh node, which corresponds to one row of the global matrix, [13]. For the case of locally varying material parameters, such as the viscosity in mantle convection, this could be extended, using classical local element matrix based approaches, see [9, 14, 37]. However, this approach does not carry over to the case of non-polyhedral domains and/or higher order elements. Thus, we developed alternative and also more efficient approaches.

3.6.1 Matrix-Free Approaches Based on Surrogate Polynomials

In our HHG and HYTEG frameworks the coupling between DoFs is considered in the form of a stencil like in classical finite differences. Due to the local regularity of the refined block-structured mesh the stencil pattern is invariant across an individual macro primitive. For example in the case of P_1 elements and a scalar equation in 3D we obtain a 15-point stencil for all nodes within a volume primitive or on a face primitive. However, in the case of a curved domain and/or locally variable material parameters the stencil weights are non-constant over a macro primitive. These weights can be computed on-the-fly when they are needed to apply the stencil locally, e.g. during a smoothing step, by standard quadrature. Note that at least for P_1 elements this is theoretically still faster than the more sophisticated approach of fusing quadrature and application of the local element-matrix, see [66]. However, it is significantly slower than the original one-stencil-per-primitive scenario. Thus, in [8] we introduce a new two-scale approach that employs surrogate polynomials to approximate the stencil weights. The idea, in a nutshell, is to consider the individual stencil weights as functions on each primitive. These functions can be sampled, in a setup phase, on a certain number of sampling points, typically for all nodes of the primitive on a coarser level of the mesh hierarchy. Then an approximating polynomial is constructed for the weight by determining the polynomial coefficients through a standard least-squares fit. Whenever the stencil weight is needed the associated polynomial is then evaluated. Performance-wise this raises two questions. How much memory is required for storing the polynomial coefficients and what is the cost for evaluating the polynomial, as opposed to that of on-the-fly quadrature. The answer to the first question is that the memory footprint even in 3D is not too large for polynomials of moderate degree. Consider as an example a trivariate polynomial of degree five. It has 56 coefficients. Representing a 15-point stencil by such polynomials for all nodes of a volume primitive, thus, requires only 6720 bytes. Note also that the number of polynomials needed can be reduced by symmetry arguments and the zero row-sum property of consistent weak differential operators, see [8, 30]. Thanks to the logical structuredness of the mesh inside our volume and face primitives evaluation of a polynomial can be performed lexicographically along lines. It, thus, reduces to a 1D problem which can efficiently be executed using incremental updates based on the concept of divided differences, see [8, 30].

In [8] we conducted an extensive performance study of the new approach for the 3D Poisson problem on different curved domains which we connected to a polygonal domain by means of a mapping function. Thus, we are in fact solving a diffusion equation with a non-constant symmetric definite material tensor. As an example Fig. 8 shows results for a weak scaling experiment on SuperMUC. Here we consider a V(3,3) cycle and assign 16 macro elements to one core. On the finest grid there are 3.3×10^5 DoFs per volume primitive. We see that the surrogate operator approach (using quadratic polynomials) clearly outperforms the on-the-fly quadrature and is only about a factor two more expensive than the original one-stencil-per-primitive approach in HHG which cannot capture the domain curvature.

Fig. 8 Weak scaling behavior of a V(3, 3) cycle for different approaches: on-the-fly quadrature (IFEM), surrogate polynomials (LSQP), surrogate polynomials combined with double discretization (LSQP DD) and surrogate polynomials after node-level performance optimization (LSQP OPT); for comparison (CONS) gives times for the one-stencil-per-primitive approach that neglects curvature

Additionally, we study the influence of the approximation on the discretization error and the multigrid convergence rate. The former is influenced by the ratio between the mesh size of the macro mesh and the solution level, the sampling level and also the polynomial degree, while the latter is basically non-existent, see Table 8. In [30] we present a theoretical analysis of the consistency error of the surrogate operator approach and extend our numerical experiments to more challenging differential operators using problems from linear elasticity and non-linear p-Laplacian diffusion.

In mantle convection models viscosity may vary significantly on a local scale, i.e. within a volume primitive. In this case, an approximation of stencil weights by low-order polynomials as in [8] will no longer be sufficient. One might try higher order polynomials as was implemented in HyTEG for the tests presented in [30]. Albeit this is currently only available in 2D. Alternatively, we devised a variant of our approach, see [10, 11], denoted in the following as LSQP_e. Here, instead of directly approximating the stencil weights, we replace the entries of the local element matrices by surrogate polynomials. The stencil weights are then computed

Table 8 Discretization error in the L^2 -norm for IFEM and LSQP (with different polynomial degree q) and ratio of the asymptotic multigrid convergence rates for curved pipe domain (level $\ell = 0$ refers to an already twice refined mesh)

Level	e_{IFEM}	e_{LSQP}	$\rho_{\text{IFEM}}/\rho_{\text{LSQP}}$	e_{LSQP}	$\rho_{\text{IFEM}}/\rho_{\text{LSQP}}$
		$q = 2$	1.00	$q = 3$	1.00
$\ell = 1$	7.50×10^{-5}	7.50×10^{-5}		7.50×10^{-5}	
$\ell = 2$	1.86×10^{-5}	1.86×10^{-5}	1.00	1.86×10^{-5}	1.00
$\ell = 3$	4.64×10^{-6}	4.67×10^{-6}	1.00	4.64×10^{-6}	1.00
$\ell = 4$	1.16×10^{-6}	1.41×10^{-6}	1.00	1.16×10^{-6}	1.00
$\ell = 5$	2.89×10^{-7}	9.87×10^{-7}	1.00	3.10×10^{-7}	1.00

Table 9 Weak scaling of the $LSQP_e$ approach using ca. 2.3×10^7 DoFs per core; shown are: average run-time w/ and w/o coarse grid solver (c.g.) for one UMG cycle and no. of UMG iterations; values in brackets give no. of c.g. iterations (preconditioner/MINRES); parallel efficiency w.r.t. one UMG cycle is given for timings w/ and w/o c.g.; additionally the average time for a single residual application on the finest level is given

Islands	Cores	DoFs	Global resolution	# UMG V-cycles	Time UMG cycle		Parallel efficiency	Time residual
					w/ c.g.	w/o c.g.		
1	5580	1.3×10^{11}	3.4 km	7 (50/150)	192 s	164 s	1.00 / 1.00	11.9 s
2	12,000	2.7×10^{11}	2.8 km	10 (100/150)	213 s	169 s	0.90 / 0.97	12.1 s
4	21,600	4.8×10^{11}	2.3 km	7 (50/250)	210 s	172 s	0.92 / 0.96	12.7 s
8	47,250	1.1×10^{12}	1.7 km	8 (50/350)	230 s	173 s	0.83 / 0.95	12.8 s

from these surrogate element matrices in the standard fashion for the P_1 case. This variant was used to perform the dynamic topography simulations of Sect. 3.10. As an example Table 9 shows weak scaling results on SuperMUC Phase 1 for a mantle convection model with a temperature-dependent viscosity and a global resolution of 1.7 km for the largest scenario. Run-times are for a V-cycle with an Uzawa smoother, stopping criterion is a residual reduction of five orders of magnitude. For complete details we refer to [11]. Note the parallel efficiency of 83% for the largest run, which could be further improved by using a more sophisticated coarse grid solver, see Sect. 3.3, than the one available at the time of the experiment.

3.6.2 A Stencil Scaling Approach for Accelerating Matrix-Free Finite Element Implementations

In [9] we present a novel approach to fast on-the-fly low order finite element assembly for scalar elliptic partial differential equations of Darcy type with variable coefficients optimized for matrix-free implementations. In this approach, we introduce a new operator that is obtained by scaling the reference operator, i.e. the stencil obtained from the constant coefficient case. Assuming sufficient regularity, an a priori analysis showed that solutions obtained by this approach are unique and have asymptotically optimal order convergence in the H^1 -norm and the L^2 -norm on hierarchical hybrid grids. These preliminary considerations motivate our novel approach to reduce the cost by recomputing the surrogate stencil entries for a matrix-free solver in a more efficient way.

To demonstrate the advantages of our novel scaling approach we consider, among other examples, a Poisson problem on a domain defined through a blending function. Particularly, we consider a half cylinder mantle with inner radius $r_1 = 0.8$ and outer radius $r_2 = 1.0$, height $z_1 = 4.0$ and with an angular coordinate between 0 and π as our physical domain Ω_{phy} . The cylinder mantle is additionally warped inwards by

$w(z) = 0.2 \sin(z\pi/z_1)$ in axial direction. The mapping $\Phi : \Omega_{\text{phy}} \rightarrow \Omega$ is given by

$$\Phi(x, y, z) = \begin{pmatrix} \sqrt{x^2 + y^2 + w(z)} \\ \arccos(x/\sqrt{x^2 + y^2}) \\ z \end{pmatrix}$$

with the reference domain $\Omega = (r_1, r_2) \times (0, \pi) \times (0, z_1)$. It follows for the mapping tensor K of the Poisson problem that

$$K = \frac{(D\Phi)(D\Phi)^\top}{|\det D\Phi|} = \sqrt{x^2 + y^2} \begin{pmatrix} w'(z)^2 + 1 & 0 & w'(z) \\ 0 & 1/(x^2 + y^2) & 0 \\ w'(z) & 0 & 1 \end{pmatrix}.$$

Obviously, this tensor is symmetric and positive definite. In addition to the geometry blending, we use a variable material parameter $a(x, y, z) = 1 + z$. This yields the following PDE on the reference domain Ω : $-\text{div}(aK\nabla u) = f$. Additionally, the manufactured solution is chosen as

$$u(x, y, z) = \sin\left(\frac{x - r_1}{r_2 - r_1}\pi\right) \cos(4y) \exp(z/2).$$

For our numerical experiments, we employ a macro mesh composed of 9540 hexahedral blocks, where each block is further split into six tetrahedral elements. The resulting system is solved on SuperMUC Phase 2 using 14,310 compute cores, i.e., four macro elements are assigned per core. The largest system involves solving a linear system with $O(10^{11})$ DoFs. We employ a multigrid solver with a V(3,3) cycle and the iterations are stopped when the residual has been reduced by a factor of 10^{-8} . In Table 10, we report on the resulting discretization error and its estimated order of convergence (eoc), the asymptotic multigrid convergence rate ρ , and the time-to-solution for different refinement levels ℓ . Note that we restricted the scaling approach in this test to volume and face primitives, which contain the vast majority of DoFs.

Table 10 Results for large scale 3D application with errors measured in the discrete L^2 -norm

DoFs	Nodal integration				Scale Vol+Face				Rel.
	Error	eoc	ρ	tts [s]	Error	eoc	ρ	tts [s]	
4.7×10^6	2.43×10^{-4}	—	0.522	2.5	2.38×10^{-4}	—	0.522	2.0	0.80
3.8×10^7	6.00×10^{-5}	2.02	0.536	4.2	5.86×10^{-5}	2.02	0.536	2.6	0.61
3.1×10^8	1.49×10^{-5}	2.01	0.539	12.0	1.46×10^{-5}	2.01	0.539	4.5	0.37
2.5×10^9	3.72×10^{-6}	2.00	0.538	53.9	3.63×10^{-6}	2.00	0.538	15.3	0.28
2.0×10^{10}	9.28×10^{-7}	2.00	0.536	307.2	9.06×10^{-7}	2.00	0.536	88.9	0.29
1.6×10^{11}	2.32×10^{-7}	2.00	0.534	1822.2	2.26×10^{-7}	2.00	0.534	589.6	0.32

These results demonstrate that the new scaling approach reproduces the discretization error, as is expected from our variational crime analysis [8, 9, 30]. Additionally, the multigrid convergence rate is not affected. For larger ℓ the runtime as compared to the nodal integration approach requires only about 30% of the time.

3.6.3 Stencil Scaling for Vector-Valued PDEs with Applications to Generalized Newtonian Fluids

Our target problem is vector-valued, thus, we expanded the scalar stencil scaling idea from Sect. 3.6.2 and developed a similar matrix-free approach for vector-valued PDEs [31]. The construction is again based on the use of hierarchical hybrid grids, the conceptual basis in the HHG and HYTEG [63] frameworks. Vector-valued second-order elliptic PDEs play an important role in mathematical modeling and arise e.g. in problems from elastostatics and fluid dynamics. Numerical experiments indicated that the idea of the scalar stencil scaling (denoted below as unphysical scaling) cannot directly be applied to these equations, because the standard finite-element solution cannot be reproduced, even in the case of linear coefficients. Thus, there is need of a modified stencil scaling method (denoted below as physical scaling) that is also suited for matrix-free finite element implementations on hierarchical hybrid grids. It turns out this vector-valued scaling requires computation of an additional correction term. While this makes it more complicated and expensive, compared to the scalar stencil scaling, it is able to reproduce the standard finite-element solutions, while requiring only a fraction of the time to obtain them. In the best scenario, we could observe a speedup of about 122% compared to standard on-the-fly integration. Our largest example involved solving a Stokes problem with 12,288 compute cores.

One of the examples studied is the scenario of a non-linear incompressible Stokes problem where the fluid is assumed to be of generalized Newtonian type, modeled by a shear-thinning Carreau model

$$\mu(\mathbf{u}) = \eta_\infty + (\eta_0 - \eta_\infty) \left(1 + \kappa |\varepsilon(\mathbf{u})|^2\right)^r.$$

The parameters employed are given in Table 11 in dimensionless form. The values stem from experimental results, cf. [34, Chapter II].

The computational domain Ω is depicted in Fig. 9. The channel has a length of 5 and a depth and height of 1. The kink starts at position $x = 1.5$. The domain is discretized by 14,208 tetrahedra on the coarsest level. The boundary

Table 11 Dimensionless parameters for the Carreau viscosity model

Parameter	η_0	η_∞	κ	r
Value	140.764	1.0	212.2	−0.325

Fig. 9 Velocity streamlines for the non-linear generalized Newtonian Stokes problem

$\partial\Omega$ is composed into Dirichlet and Neumann parts, i.e., $\partial\Omega = \Gamma_D \cup \Gamma_N$ with $\Gamma_D = \{(x, y, z) \in \partial\Omega \mid x < 5\}$ and $\Gamma_N = \partial\Omega \setminus \Gamma_D$. The volume force term $\mathbf{f}$, the external forces $\hat{\mathbf{t}}$, and the Dirichlet boundary term $\mathbf{g}$ are set to

$$\mathbf{f} = (0, 0, 100)^\top, \quad \hat{\mathbf{t}} = (0, 0, 0)^\top, \quad \text{and} \quad \mathbf{g} = 16 \, y \, (1 - y) \, z \, (1 - z) \cdot (1, 0, 1)^\top.$$

We solve this non-linear system by applying an inexact fixed-point iteration where the underlying linear systems are only solved approximately to prevent over-solving.

In order to solve the systems, we employ the inexact Uzawa solver presented from [29] with variable $V(3, 3)$ cycles where 2 smoothing steps are added to each coarser refinement level which enforces convergence of the method.

In Table 12, we present the relative errors of the physical and unphysical scaling approaches along the line $\theta = [0, 5] \times [0.5] \times [0.42]$ in the supremum norm, computed on the 4 times refined grid. It can clearly be seen that the physical scaling yields significantly better results with errors smaller by three orders of magnitude. Further experiments indicate that the unphysical scaling does not converge to the standard finite element solution even after additional mesh refinements. Therefore, we conclude that the unphysical scaling is an inconsistent discretization for vector-valued problems.

Table 13 presents the relative time-to-solutions for the nodal integration and physical scaling measured on SuperMUC Phase 2. On the finest level, after 6 refinements, the relative time-to-solution of the physical scaling is at about 81%. Figure 9 shows the streamlines of the velocity within the computational domain computed with the physical scaling.

Table 12 Relative errors along the line θ in the supremum norm for different scaling approaches

Viscosity		Velocity	
Physical	Unphysical	Physical	Unphysical
8.88×10^{-4}	1.07×10^{-1}	5.65×10^{-5}	3.19×10^{-2}

Table 13 Relative time-to-solution comparison of the nodal integration and physical scaling approach for the non-linear generalized Stokes problem

DoFs	Nodal integration tts [s]	Physical scaling tts [s]	Relative tts
4.69×10^6	309.10	364.97	1.18
3.82×10^7	361.90	412.10	1.14
3.08×10^8	895.18	719.55	0.80
2.47×10^9	3227.45	2626.13	0.81

3.7 Resilience

Extremely concurrent simulations may in the future require a software design that is resilient to faults of individual software and hardware components. The probability of faults in the underlying systems increases with the number of parallel processes. Especially long-running simulations may suffer from lower mean time between failures and restarting applications with run-times of several hours or days consumes vast amounts of resources.

Therefore, fault tolerance techniques have become a research topic as preparation for possibly unreliable future exascale systems. In the TERRANEO project we have so far mainly focused on hard faults. To cope with the failure of a core or node, we mainly distinguish between two categories of methods: one class relies on checkpointing techniques, where snapshots of the simulation are stored in regular intervals so that the state can be loaded upon failure. A second class are algorithm-based fault tolerance (ABFT) techniques, where (partially) lost data can be re-computed on-the-fly. Regarding large-scale simulations, checkpointing techniques often suffer from bad serial and parallel performance if data is written to disk. However, there are approaches that solely rely on distributed, in-memory checkpointing, which could be combined with compression techniques [71] that were also explored in a student project [74]. This can lead to a flexible, fast, and scalable resilience method [64].

In [53, 54] an alternative ABFT technique to provide resilience specifically for multigrid methods is developed. Given a faulty subdomain Ω_F , created when a processor (core or node) crashes, the global solution can be recovered by solving a recovery problem on Ω_F via a local multigrid iteration. After the recovery subproblem is solved, the global iteration continues.

A priori, it is not clear, how many iterations will be required in the faulty subdomain. In [54] a fixed number of V-cycle iterations is employed. This can lead to under- or over-solving in Ω_F . While under-solving leads to a poor recovery, over-solving inhibits the global convergence due to wrong values at the interface. The convergence behavior of both scenarios is illustrated in Fig. 10.

To address this issue, in [56] the recovery algorithm is enhanced by an adaptive control mechanism. Instead of solving the subproblem using a fixed number of iterations, a hierarchically weighted error estimator is introduced to define a stopping criterion for the faulty subdomain. The estimator is designed to be well-suited to extreme-scale parallel multigrid settings as they are employed in

Fig. 10 Convergence behavior for over- and under-solving in the faulty subdomain

Fig. 11 Comparison of the estimated error using the hierarchically weighted error estimator to the error in a fault-free and a recovery-free scenario. $\kappa \sigma^{LRB}$ denotes the local recoupling threshold for details see [56]

applications. It guards the algorithm against over- or under-solving the subproblem and therefore wasting computational resources.

As already suggested in [54], a so-called *superman strategy* is employed, which increases the computing power locally in the faulty subdomain. The recovery computation is accelerated e.g. through a further subdivision of Ω_F by means of additional shared memory parallelism. In Fig. 11 the development of the estimated error of a recovered solution is compared to the error of a fault-free scenario and a no-recovery scenario. Locally, the number of processes is increased by a factor of 4.

We have also conducted studies with *asynchronous* recovery strategies. Here, the local problem on Ω_F and the problem on the healthy domain $\Omega \setminus \overline{\Omega_F}$ are solved

concurrently. This means that during the recovery process, the multigrid iteration is also continued in parallel in the healthy domain. Different strategies have been explored in [54] which boundary conditions are suitable for the healthy domain on the interface. After the stopping criterion in the faulty domain is fulfilled, the local recovery iteration is terminated and a *recoupling* procedure is initiated.

To emphasize the applicability to large-scale scenarios, the adaptive approach is scaled in [56] to a simulation with about 6.9×10^{11} DoFs and 245,766 processes run on the JUQUEEN supercomputer. Additionally, the generality of the approach was demonstrated for a scenario where multiple failures in different regions of the domain have been triggered.

3.8 General Performance Issues

Simulations in geophysics require a high spatial resolution which leads to problems with many degrees of freedom. The success of a geophysics simulation framework will thus depend on its scalability and that it has only minimal memory overhead. In the previous sections we already presented scalability results to highlight the results obtained in the TERRANEO project.

However, message passing scalability is not the only metric that is important in high-performance computing. Even more important is the absolute performance which depends critically also on the performance on a single core or a single node. In the extreme scale computing community the relevance of a systematic node-level performance engineering is increasingly being realized, see e.g. [4, 45, 48, 60, 67]. Note that traditional scalability is easier to achieve when the node-level performance is lower. Publishing only scalability results without absolute performance measures is a frequently observed parallel computing cheat, as exposed in [2]. We emphasize here that the TERRANEO project has in this sense profited from long term research efforts in HPC performance engineering. These stem originally from the Dime project⁴ [65, 97] and have led to the excellent performance features of the HHG library [12, 13]. Thus, node level performance in TERRANEO is not coming as an afterthought imposed on existing codes, but has been an a priori design goal with high priority.

Within TERRANEO, these techniques were continuously employed to analyze the performance and were used to guide the development of new matrix-free methods. This includes rather simple analysis like calculating the update cost for a single stencil update in terms of floating point operations per second which allows comparing the actual performance to the maximally achievable performance on a given hardware. This type of analysis has been performed in [10] and [11].

In [9] this metric was also extended with an analysis of the memory traffic that is required to apply the stencil to a function which, in terms of linear algebra,

---

⁴<http://dime.fau.de>.

Fig. 12 Roofline model for different approaches presented in [9]

corresponds to matrix-vector multiplication. The analysis provided a lower and upper bound for the performance with respect to memory. One for the optimistic assumption that all data required resides in the last level cache, and one for the pessimistic assumption that everything needs to be loaded from main memory. To evaluate this analysis the Intel Advisor⁵ was used as an automatic tool to perform the well established Roofline analysis [58, 103]. The results are given in Fig. 12.

In cooperation with the HPC group of the RRZE in Erlangen, a more sophisticated analysis was performed in [8, 40, 41] where the Execution-Cache-Memory (ECM) model [48] was used to examine the performance.

The Roofline model assumes that only one data location is the bottleneck. This can be the main memory or any of the cache levels represented by the four different limits in Fig. 12. The ECM model however, takes into account memory, all levels of cache, and finally the registers within the CPU. The model helped to design the new computational kernels by identifying performance bottlenecks and guiding subsequent performance optimization steps. The end result of this development procedure are kernels where the analytically predicted optimal duration of a stencil-based update is in good agreement with the measurements in the real program code. Here an error of only $\approx 15\%$ is considered acceptable. In Fig. 13 the occupancy of the execution units within the CPU is shown, as it was derived from the performance models.

⁵<https://software.intel.com/advisor>.

Fig. 13 Duration of the different phases involved during eight stencil-based updates as obtained by the ECM model for the newly developed kernel in [8]

3.9 HyTeG

We successfully demonstrated that the combination of a fully unstructured grid with regular refinement and matrix-free algorithms can be used to solve large geodynamical problems. A substantial amount of work in the TERRANEO project was performed using the HHG prototype. Even though the latter shows excellent performance and scalability, certain issues need to be faced. Due to the fact that the codebase was created over a decade ago, the coding standard cannot live up to current requirements. This makes it hard to maintain the framework and to familiarise new users with it, which is essential for a community code. Furthermore, the prototype was never meant to be used for higher-order discretizations or other than nodal elements.

Therefore a redesign is initiated under a new name: Hybrid Tetrahedral Grids (HYTEG) [63]. In addition to building on the knowledge gathered over the years from developing the HHG framework we could also utilize ideas and infrastructure from the WALBERLA framework [44]. One of the fundamental changes in HYTEG is the strict separation of the data structures that define the macro mesh geometry and the actual simulation data. As in HHG, the tetrahedra of the unstructured macro mesh are split into their geometric primitives, namely volumes, faces, edges and vertices. The lower dimensional primitives (faces, edges and vertices) are used to decouple the volume primitives in terms of communication. Contrary to HHG the parallel partitioning is not solely based on volume primitives, instead all primitives get partitioned between processes. This e.g. allows to take also the computational and memory footprint for face primitives into account for load balancing, for which we can employ sophisticated tools such as ParMetis [61]. Note that in HYTEG the partitioning does not involve global data structures, an essential aspect when entering the exascale era. Like with HHG, it was decided to use C++ as the primary programming language due to its high performance and spread in the HPC

Fig. 14 Basic types can be combined to create various finite element discretizations

community. Up to this point, no compute data are associated with the primitives. In a second step, these can be attached to the primitives. This might be a single value for only some of the primitives or a full hierarchy of grids for every single primitive. This separation also allows for efficient light-weight static and dynamic load balancing techniques similar to those described in [32, 96].

As mentioned above, one major goal when moving from HHG to HYTEG is the realization of different discretizations on hierarchical hybrid grids. In addition to unknowns located at the vertices of each micro-element of the mesh, unknowns on the edges, faces and inside the elements are also supported. These basic types of unknowns are implemented in such a way that different discretizations can be realized by combination which is illustrated in Fig. 14. Using this technique, a Taylor-Hood discretization for a Stokes flow simulation can be realized as well as finite volume discretizations to simulate energy transport [63].

By introducing different types of discretizations, there are also many more compute-intensive kernels that need to be taken care of. The solution chosen in HYTEG to solve this problem is code generation. Inspired by the work in ExaStencils [76, 94], our joint collaboration in this direction [15], and also using experience from pystencils⁶ this can be efficiently realized. One difference to other projects that use whole program generation, however, is that only compute intensive kernels are generated, but not the surrounding data structures. In contrast to HYTEG itself, pystencils is using Python, which allows for much more flexibility and metaprogramming capabilities. The generated kernels, however are in C++, which eliminates the need for a Python environment when running HYTEG on supercomputers.

⁶<https://i10git.cs.fau.de/pycodegen/pystencils>.

Fig. 15 Each combination of one leaf of every color represents one possible configuration

Another important feature that was introduced with the community code in mind is a state-of-the-art continuous integration process. With every change in the codebase, an automated process is started, which ensures the compatibility with different hardware and software configurations. Additionally, extensive testing is also part of the pipeline to guarantee the correctness of all features inside HyTEG. To illustrate the complexity of this process Fig. 15 shows all possible combinations of configurations.

3.10 *Asthenosphere Velocities and Dynamic Topography*

In order to demonstrate the flexibility of the different algorithmic components, we have studied selected application questions. In [102] we investigated the question of flow speeds in the asthenosphere. The latter is a mechanically weak layer in the upper mantle right below the lithosphere, the rigid outermost shell of our planet. It

plays an important role in convection modelling as it distinguishes itself from the lower parts of the mantle by a significantly lower viscosity. The precise details of the contrast are, however, unknown. A variety of geologic observations and geodynamic arguments indicates that velocities in the upper mantle may exceed those of tectonic plates by an order of magnitude [49, 51]. The framework was used to simulate high-resolution whole earth convection models with asthenospheric channels of varying thickness. Reduction of the asthenospheric thickness is balanced by an associated reduction in its viscosity, following the *Cathles* parameter [88]. This resulted for the tested end-member case in a set-up with an asthenosphere channel of only 100 km depth and a significant viscosity contrast of up to 4 orders of magnitude relative to the deeper mantle. We found a velocity increase by a factor of 10 between a reference case with an upper mantle of 1000 km depth and the very thin channel end-member case, translating into speeds of ≈ 20 cm/a within the narrow asthenosphere. Our suggested and numerically verified Poiseuille flow model predicts that the upper mantle velocity scales with the inverse of the asthenospheric thickness.

Note that the prototype implemented in HHG already allows to include real-world geophysical data. The model presented in Fig. 16, e.g., uses present-day plate velocities, see [83], as surface boundary conditions and a buoyancy term based on present day temperature and density fields, converted from a seismic tomography model via techniques described in [27]. Viscosity depends on temperature and includes a jump at the bottom of the asthenosphere, chosen at 660 km.

Another geodynamic quantity we studied is *dynamic topography*. This term, which plays a crucial role in our understanding of long term sea-level variations, refers to deflections of the Earth's surface as a response to viscous stresses in the mantle, and has been known from geodynamic arguments for a long time [47, 87]. The precise magnitude of dynamic topography is still debated, however, due to uncertainties in measuring it [18]. Dynamic topography occurs both at the surface

Fig. 16 Mantle convection model computed in HHG with real-world geophysical data: temperature field from seismic tomography for viscosity (left) and resulting flow speeds (right); quantities are non-dimensional

Fig. 17 Surface dynamic topography for cases: A—viscosity with only radial variations, B—with the addition of viscosity variations due to varying thickness of the lithosphere, C—with additional temperature-dependent viscosity in the lower mantle (left) and pairwise difference between scenarios (right)

and the core-mantle boundary (CMB), see [26] for a review. In [10] we studied the influence of lateral viscosity variations on the amplitude and pattern of dynamic topography. We first considered a standard benchmark with a purely radially varying viscosity profile, see [23, 105], for which a semi-analytic solution exists in terms of the propagator matrix technique [47, 85, 87]. This allowed us to verify suitable correctness of our LSQP_e approach from Sect. 3.6. We then compared models with increasing levels of complexity, see Fig. 17, going up to a profile with lateral viscosity variations due to varying thickness of the lithosphere and temperature-dependent variations in the lower mantle. Full details and a discussion of the outcomes can be found in [10].

4 Conclusions and Future Work

TERRANEO is a project in *Computational Science and Engineering* [91]. We successfully developed innovative scientific computing methods for the exascale era. To this end, new algorithms were devised for simulating Earth Mantle convection. We analyzed the accuracy of these methods and their cost on massively parallel computers, showing that our methods exhibit excellent performance. A highly optimized prototype implementation was designed using the HHG multigrid library. Based on HHG, we demonstrated computations with an unprecedented resolution of the Earth's Mantle based on real-world geophysical input data.

In TERRANEO many new methods for future exascale geodynamic simulations were invented: they include new solver components for monolithic multigrid methods for saddle point problems, specially designed smoothers and new strategies to solve the coarse grid problems. Two new classes of matrix-free methods were introduced and analyzed, one based on surrogate polynomials, the other one based on stencil scaling. New scheduling algorithms for parallel multilevel Monte Carlo methods and uncertainty quantification were studied. Methods for fault tolerance were investigated. This includes methods based on in-memory checkpointing as well as new methods for fast algorithmic reconstruction of lost data when hard faults occur. Inverse problems in mantle convection were studied using adjoint techniques.

Careful parallel performance studies complement our work and assess the suitability of the algorithmic components on future exascale computers. Many of the methods were tested on application oriented problems, such as for example a geophysics study on the relation of the thickness of the asthenosphere and upper mantle velocities and the influence of viscosity variations on dynamic topography.

We found that conventional C++-based implementation techniques lack expressiveness and flexibility for massively parallel and highly optimized parallel codes and that achieving performance portability is a major difficulty. Starting from this insight, we invented new programming techniques based on automatic program generation, learning from neighboring SPPEXA projects such as ExaStencils. These ideas have already been realized in HYTEG and WALBERLA as new and innovative simulation software architectures for multiphysics exascale computing.

Thus, some aspects of our research in TERRANEO are of mathematical nature, others fall into the field of computer science. Our methods have also already been used to perform research in geophysics, the *target discipline* of TERRANEO. However, we point out that the research contribution of TERRANEO falls neither into the intersection of all these fields, nor can our contribution be understood from the viewpoint of mathematics or computer science alone. It is also no project in the geosciences, since its primary goal is not the creation of new geophysical insight. The goal of TERRANEO is the construction and analysis of simulation methods that are the enabling technologies for future exascale research in geodynamics.

This result could not be reached by either of the classical disciplines alone. The synthesis of knowledge and methods from mathematics, computer science, and geophysics becomes more than the sum of its parts. We emphasize additionally

that our research in computational science and engineering is not restricted to the application in geophysics. Many of the innovations developed in TERRANEO can be transferred to other *target disciplines*.

With our new computational methods, we have broken previously existing barriers to computational performance. This is demonstrated for example by our solution of indefinite linear systems that have in excess of 10^{13} degrees of freedom. To our knowledge this constitutes the largest finite element computation published to date, even though the computation was still performed on JUQUEEN, a machine that is now outdated and was already retired.

TERRANEO funding in the last period was reduced from the desired four to only three positions so that the new software design and its development could not be completed as originally proposed. A preliminary version of the TERRANEO software will be made public and will contain essential parts of the promised core functionality, but it still lacks most of the application oriented functionality that would make it fully usable as a Geophysics community code. Efforts will be made to continue the development so that the central research goal of TERRANEO can be reached.

Acknowledgments The authors gratefully acknowledge the Gauss Centre for Supercomputing e.V. for funding this project by providing computing time on the GCS Supercomputers: SuperMUC at Leibniz Supercomputing Centre (LRZ), JUQUEEN at Jülich Supercomputing Centre (JSC) and Hazel Hen at High-Performance Computing Center Stuttgart (HLRS). Financial support by the German Research Foundation through the Priority Programme 1648 Software for Exascale Computing (SPPEXA), WO671/11-1 and BU 1012/15-1 is greatly acknowledged. We also would like to thank our former project collaborators: D. Bartuschat, J. Eitzinger, B. Gmeiner, M. Hofmann, L. John, H. Stengel, Chr. Waluga, J. Weismüller, G. Wellein, and M. Wittmann.

References

1. Amestoy, P., Buttar, A., L'Excellent, J.Y., Mary, T.: On the complexity of the block low-rank multifrontal factorization. *SIAM J. Sci. Comp.* **39**(4), A1710–A1740 (2017)
2. Bailey, D.H.: Misleading performance claims in parallel computations. In: *Proceedings of the 46th Annual Design Automation Conference*, pp. 528–533. ACM, New York (2009)
3. Baker, A., Klawonn, A., Kolev, T., Lanser, M., Rheinbach, O., Yang, U.: Scalability of classical algebraic multigrid for elasticity to half a million parallel tasks. In: *Software for Exascale Computing-SPPEXA 2013–2015*, pp. 113–140. Springer, Berlin (2016)
4. Bastian, P., Engwer, C., Fahlke, J., Geveler, M., Göddeke, D., Iliev, O., Ippisch, O., Milk, R., Mohring, J., Müthing, S., Ohlberger, M., Ribbrock, D., Turek, S.: Hardware-based efficiency advances in the EXA-DUNE project. In: *Software for Exascale Computing - SPPEXA 2013–2015. Lecture Notes in Computational Science and Engineering*. Springer, Berlin (2016)
5. Bastian, P., Müller, E.H., Müthing, S., Piatkowski, M.: Matrix-free multigrid block-preconditioners for higher order discontinuous Galerkin discretisations. *J. Comput. Phys.* **394**, 417–439 (2019). [doi:https://doi.org/10.1016/j.jcp.2019.06.001](https://doi.org/10.1016/j.jcp.2019.06.001). <http://www.sciencedirect.com/science/article/pii/S0021999119303973>
6. Bauer, A., Schaal, K., Springel, V., Chandrashekar, P., Pakmor, R., Klingenberg, C.: Simulating turbulence using the astrophysical discontinuous Galerkin code TENET. In: *Software for Exascale Computing-SPPEXA 2013–2015*, pp. 381–402. Springer, Berlin (2016)

7. Bauer, S., Bunge, H.P., Drzisga, D., Gmeiner, B., Huber, M., John, L., Mohr, M., Rüde, U., Stengel, H., Waluga, C., et al.: Hybrid parallel multigrid methods for geodynamical simulations. In: *Software for Exascale Computing-SPPEXA 2013–2015*, pp. 211–235. Springer, Berlin (2016). https://doi.org/10.1007/978-3-319-40528-5_10
8. Bauer, S., Mohr, M., Rüde, U., Weismüller, J., Wittmann, M., Wohlmuth, B.: A two-scale approach for efficient on-the-fly operator assembly in massively parallel high performance multigrid codes. *Appl. Numer. Math.* **122**, 14–38 (2017). <https://doi.org/10.1016/j.apnum.2017.07.006>
9. Bauer, S., Drzisga, D., Mohr, M., Rüde, U., Waluga, C., Wohlmuth, B.: A stencil scaling approach for accelerating matrix-free finite element implementations. *SIAM J. Sci. Comp.* **40**(6), C748–C778 (2018). <https://doi.org/10.1137/17M1148384>
10. Bauer, S., Huber, M., Mohr, M., Rüde, U., Wohlmuth, B.: A new matrix-free approach for large-scale geodynamic simulations and its performance. In: *International Conference on Computational Science*, pp. 17–30. Springer, Berlin (2018). https://doi.org/10.1007/978-3-319-93701-4_2
11. Bauer, S., Huber, M., Ghelichkhan, S., Mohr, M., Rüde, U., Wohlmuth, B.: Large-scale simulation of mantle convection based on a new matrix-free approach. *J. Comp. Sci.* **31**, 60–76 (2019). <https://doi.org/10.1016/j.jocs.2018.12.006>
12. Bergen, B., Hülsemann, F., Rüde, U.: Is 1.7×10^{10} unknowns the largest finite element system that can be solved today? In: *SC'05: Proceedings of the 2005 ACM/IEEE Conference on Supercomputing*, pp. 5–5. IEEE, Piscataway (2005)
13. Bergen, B., Gratl, T., Hülsemann, F., Rüde, U.: A massively parallel multigrid method for finite elements. *Comp. Sci. Eng.* **8**(6), 56–62 (2006)
14. Bielak, J., Ghattas, O., Kim, E.J.: Parallel octree-based finite element method for large-scale earthquake ground motion simulation. *Comput. Model. Eng. Sci.* **10**(2), 99–112 (2005). <https://doi.org/10.3970/cmescs.2005.010.099>
15. Bolten, M., Franchetti, F., Kelly, P., Lengauer, C., Mohr, M.: Algebraic description and automatic generation of multigrid methods in SPIRAL. *Concurrency Comput. Pract. Exp.* **29**(17) (2017). Special Issue on Advanced Stencil-Code Engineering. <https://doi.org/10.1002/cpe.4105>
16. Braess, D., Sarazin, R.: An efficient smoother for the Stokes problem. *Appl. Numer. Math.* **23**(1), 3–19 (1997). *Multilevel methods* (Oberwolfach, 1995)
17. Brandt, A., Livne, O.E.: *Multigrid Techniques: 1984 Guide with Applications to Fluid Dynamics*, vol. 67. SIAM, Philadelphia (2011)
18. Braun, J.: The many surface expressions of mantle dynamics. *Nat. Geosci.* **3**, 825–833 (2010). <https://doi.org/10.1038/ngeo1020>
19. Brezzi, F., Douglas, J.: Stabilized mixed methods for the Stokes problem. *Numer. Math.* **53**(1–2), 225–235 (1988)
20. Brown, J.: Efficient nonlinear solvers for nodal high-order finite elements in 3D. *J. Sci. Comp.* **45**(1–3), 48–63 (2010). <https://doi.org/10.1007/s10915-010-9396-8>
21. Bunge, H.P., Richards, M.A., Baumgardner, J.R.: Effect of depth-dependent viscosity on the planform of mantle convection. *Nature* **379**(6564), 436–438 (1996). <https://doi.org/10.1038/379436a0>
22. Bunge, H.P., Hagelberg, C.R., Travis, B.J.: Mantle circulation models with variational data assimilation: inferring past mantle flow and structure from plate motion histories and seismic tomography. *Geophys. J. Int.* **152**(2), 280–301 (2003). <https://doi.org/10.1046/j.1365-246X.2003.01823.x>
23. Burstedde, C., Stadler, G., Alisic, L., Wilcox, L.C., Tan, E., Gurnis, M., Ghattas, O.: Large-scale adaptive mantle convection simulation. *Geophys. J. Int.* **192**(3), 889–906 (2013). <https://doi.org/10.1093/gji/ggs070>
24. Clevenger, T.C., Heister, T., Kanschä, G., Kronbichler, M.: A flexible, parallel, adaptive geometric multigrid method for FEM. Technical Report (2019). arXiv:1904.03317

25. Colli, L., Bunge, H.P., Schuberth, B.S.A.: On retrodictions of global mantle flow with assimilated surface velocities. *Geophys. Res. Lett.* **42**(20), 8341i–8348 (2015). <https://doi.org/10.1002/2015gl066001>
26. Colli, L., Ghelichkhan, S., Bunge, H.P.: On the ratio of dynamic topography and gravity anomalies in a dynamic earth. *Geophys. Res. Lett.* **43**(6), 2510–2516 (2016). <https://doi.org/10.1002/2016gl067929>
27. Colli, L., Ghelichkhan, S., Bunge, H.P., Oeser, J.: Retrodictions of Mid Paleogene mantle flow and dynamic topography in the Atlantic region from compressible high resolution adjoint mantle convection models: sensitivity to deep mantle viscosity and tomographic input model. *Gondwana Res.* **53**, 252–272 (2018). <https://doi.org/10.1016/j.gr.2017.04.027>
28. Drzisga, D., Gmeiner, B., Rüde, U., Scheichl, R., Wohlmuth, B.: Scheduling massively parallel multigrid for multilevel Monte Carlo methods. *SIAM J. Sci. Comp.* **39**(5), S873–S897 (2017). <https://doi.org/10.1137/16M1083591>
29. Drzisga, D., John, L., Rüde, U., Wohlmuth, B., Zulehner, W.: On the analysis of block smoothers for saddle point problems. *SIAM J. Mat. Ana. Appl.* **39**(2), 932–960 (2018). <https://doi.org/10.1137/16M1106304>
30. Drzisga, D., Keith, B., Wohlmuth, B.: The surrogate matrix methodology: a priori error estimation (2019). Preprint arXiv:1902.07333
31. Drzisga, D., Rüde, U., Wohlmuth, B.: Stencil scaling for vector-valued PDEs on hybrid grids with applications to generalized Newtonian fluids (2019). Preprint arXiv:1908.08666
32. Eibl, S., Rüde, U.: A systematic comparison of runtime load balancing algorithms for massively parallel rigid particle dynamics. *Comput. Phys. Commun.* **244**, 76–85 (2019). <https://doi.org/10.1016/j.cpc.2019.06.020>
33. Elman, H.C., Silvester, D.J., Wathen, A.J.: *Finite Elements and Fast Iterative Solvers: With Applications in Incompressible Fluid Dynamics*. Oxford University Press, Oxford (2014)
34. Galdi, G.P., Rannacher, R., Robertson, A.M., Turek, S.: *Hemodynamical Flows*. Delhi Book Store, New Delhi (2008)
35. Ghelichkhan, S., Bunge, H.P.: The compressible adjoint equations in geodynamics: derivation and numerical assessment. *GEM Int. J. Geomath.* **7**(1), 1–30 (2016). <https://doi.org/10.1007/s13137-016-0080-5>
36. Ghelichkhan, S., Bunge, H.P.: The adjoint equations for thermochemical compressible mantle convection: derivation and verification by twin experiments. *Proc. R. Soc. A Math. Phys. Eng. Sci.* **474**(2220), 20180329 (2018). <https://doi.org/10.1098/rspa.2018.0329>
37. Gmeiner, B.: Design and analysis of hierarchical hybrid multigrid methods for peta-scale systems and beyond. Ph.D. Thesis, Technische Fakultät der Friedrich-Alexander-Universität Erlangen-Nürnberg (2013)
38. Gmeiner, B., Köstler, H., Stürmer, M., Rüde, U.: Parallel multigrid on hierarchical hybrid grids: a performance study on current high performance computing clusters. *Concurrency Comput. Pract. Exp.* **26**(1), 217–240 (2014)
39. Gmeiner, B., Waluga, C., Wohlmuth, B.: Local mass-corrections for continuous pressure approximations of incompressible flow. *SIAM J. Numer. Ana.* **52**(6), 2931–2956 (2014). <https://doi.org/10.1137/140959675>
40. Gmeiner, B., Rüde, U., Stengel, H., Waluga, C., Wohlmuth, B.: Performance and scalability of hierarchical hybrid multigrid solvers for Stokes systems. *SIAM J. Sci. Comp.* **37**(2), C143–C168 (2015). <https://doi.org/10.1137/130941353>
41. Gmeiner, B., Rüde, U., Stengel, H., Waluga, C., Wohlmuth, B.: Towards textbook efficiency for parallel multigrid. *Numer. Math. Theory, Methods Appl.* **8**(1), 22–46 (2015). <https://doi.org/10.4208/nmtma.2015.w10si>
42. Gmeiner, B., Huber, M., John, L., Rüde, U., Waluga, C., Wohlmuth, B.: Massively parallel large scale Stokes flow simulation. In: *NIC Symposium* (2016)
43. Gmeiner, B., Huber, M., John, L., Rüde, U., Wohlmuth, B.: A quantitative performance study for Stokes solvers at the extreme scale. *J. Comp. Sci.* **17**, 509–521 (2016). <https://doi.org/10.1016/j.jocs.2016.06.006>

44. Godenschwager, C., Schornbaum, F., Bauer, M., Köstler, H., Rüde, U.: A framework for hybrid parallel flow simulations with a trillion cells in complex geometries. In: SC'13: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, pp. 1–12. IEEE, Piscataway (2013)
45. Guillet, T., Pakmor, R., Springel, V., Chandrashekar, P., Klingenberg, C.: High-order magnetohydrodynamics for astrophysics with an adaptive mesh refinement discontinuous Galerkin scheme. *Mon. Not. R. Astron. Soc.* **485**(3), 4209–4246 (2019)
46. Gupta, S., Wohlmuth, B., Helmig, R.: Multi-rate time stepping schemes for hydrogeomechanical model for subsurface methane hydrate reservoirs. *Adv. Water Res.* **91**, 78–87 (2016). <https://doi.org/10.1016/j.advwatres.2016.02.013>
47. Hager, B.H., Clayton, R.W., Richards, M.A., Comer, R.P., Dziewonski, A.M.: Lower mantle heterogeneity, dynamic topography and the geoid. *Nature* **313**(6003), 541–545 (1985). <https://doi.org/10.1038/313541a0>
48. Hager, G., Treibig, J., Habich, J., Wellein, G.: Exploring performance and power properties of modern multicore chips via simple machine models. *Concurrency Comput. Pract. Exp.* **28**, 189–210 (2014)
49. Hartley, R.A., Roberts, G.G., White, N.J., Richardson, C.: Transient convective uplift of an ancient buried landscape. *Nat. Geosci.* **4**(8), 562–565 (2011). <https://doi.org/10.1038/ngeo1191>
50. Heister, T., Dannberg, J., Gassmöller, R., Bangerth, W.: High accuracy mantle convection simulation through modern numerical methods - II: realistic models and problems. *Geophys. J. Int.* **210**(2), 833–851 (2017). <https://doi.org/10.1093/gji/ggx195>
51. Höink, T., Lenardic, A., Richards, M.A.: Depth-dependent viscosity and mantle stress amplification: implications for the role of the asthenosphere in maintaining plate tectonics. *Geophys. J. Int.* **191**(1), 30–41 (2012). <https://doi.org/10.1111/j.1365-246X.2012.05621.x>
52. Horbach, A., Bunge, H.P., Oeser, J.: The adjoint method in geodynamics: derivation from a general operator formulation and application to the initial condition problem in a high resolution mantle circulation model. *GEM Int. J. Geomath.* **5**(2), 163–194 (2014). <https://doi.org/10.1007/s13137-014-0061-5>
53. Huber, M., John, L., Pustejovska, P., Rüde, U., Waluga, C., Wohlmuth, B.: Solution techniques for the Stokes system: A priori and a posteriori modifications, resilient algorithms. In: Proceedings of the ICIAM, Beijing. Higher Education Press, Beijing (2015)
54. Huber, M., Gmeiner, B., Rüde, U., Wohlmuth, B.: Resilience for massively parallel multigrid solvers. *SIAM J. Sci. Comp.* **38**(5), S217–S239 (2016). <https://doi.org/10.1137/15M1026122>
55. Huber, M., Rüde, U., Waluga, C., Wohlmuth, B.: Surface couplings for subdomain-wise isoviscous gradient based Stokes finite element discretizations. *J. Sci. Comp.* **74**(2), 895–919 (2018)
56. Huber, M., Rüde, U., Wohlmuth, B.: Adaptive control in roll-forward recovery for extreme scale multigrid. *Int. J. High Perf. Comp. Appl.* pp. 1–21 (2018). <https://doi.org/10.1177/1094342018817088>
57. Iaffaldano, G., Lambeck, K.: Pacific plate-motion change at the time of the Hawaiian-Emperor bend constrains the viscosity of Earth's asthenosphere. *Geophys. Res. Lett.* **41**(10), 3398–3406 (2014). <https://doi.org/10.1002/2014GL059763>
58. Ilic, A., Pratas, F., Sousa, L.: Cache-aware Roofline model: upgrading the loft. *IEEE Comp. Arch. Lett.* **13**(1), 21–24 (2014). <https://doi.org/10.1109/L-CA.2013.6>
59. Ismail-Zadeh, A., Schubert, G., Tsepelev, I., Korotkii, A.: Inverse problem of thermal convection: numerical approach and application to mantle plume restoration. *Phys. Earth Planet. Int.* **145**(1–4), 99–114 (2004). <https://doi.org/10.1016/j.pepi.2004.03.006>
60. Jumah, N., Kunkel, J., Zängl, G., Yashiro, H., Dubos, T., Meurdesoif, Y.: GGDML: icosahedral models language extensions. *J. Comput. Sci. Technol. Updat.* **4**, 1–10 (2017). <https://doi.org/10.15379/2410-2938.2017.04.01.01>
61. Karypis, G., Kumar, V.: A parallel algorithm for multilevel graph partitioning and sparse matrix ordering. *J. Paral. Distrib. Comput.* **48**(1), 71–95 (1998). <https://doi.org/10.1006/jpdc.1997.1403>

62. Klawonn, A., Lanser, M., Rheinbach, O.: Toward extremely scalable nonlinear domain decomposition methods for elliptic partial differential equations. *SIAM J. Sci. Comput.* **37**(6), C667–C696 (2015). <https://doi.org/10.1137/140997907>
63. Kohl, N., Thönnies, D., Drzisga, D., Bartuschat, D., Rüde, U.: The HyTeG finite-element software framework for scalable multigrid solvers. *Int. J. Parallel Emergent Distrib. Syst.* **34**, 1–20 (2018). <https://doi.org/10.1080/17445760.2018.1506453>
64. Kohl, N., Hötzer, J., Schornbaum, F., Bauer, M., Godenschwager, C., Köstler, H., Nestler, B., Rüde, U.: A scalable and extensible checkpointing scheme for massively parallel simulations. *Int. J. High Perfor. Comput. Appl.* **33**(4), 571–589 (2019)
65. Kowarschik, M., Rüde, U., Weiss, C., Karl, W.: Cache-aware multigrid methods for solving Poisson's equation in two dimensions. *Computing* **64**(4), 381–399 (2000)
66. Kronbichler, M., Kormann, K.: A generic interface for parallel cell-based finite element operator application. *Comput. Fluids* **63**, 135–147 (2012). <https://doi.org/10.1016/j.compfluid.2012.04.012>
67. Kronbichler, M., Kormann, K.: Fast matrix-free evaluation of discontinuous Galerkin finite element operators. *ACM Trans. Math. Softw.* **45**(3), 29:1–29:40 (2019). <https://doi.org/10.1145/3325864>
68. Kronbichler, M., Ljungkvist, K.: Multigrid for matrix-free high-order finite element computations on graphics processors. *ACM Trans. Parall. Comput.* **6**(1) (2019). <https://doi.org/10.1145/3322813>
69. Kronbichler, M., Wall, W.A.: A performance comparison of continuous and discontinuous Galerkin methods with fast multigrid solvers. *SIAM J. Sci. Comp.* **40**(5), A3423–A3448 (2018). <https://doi.org/10.1137/16M110455X>
70. Kuckuk, S., Köstler, H.: Automatic generation of massively parallel codes from ExaSlang. *Computation* **4**(3), 27:1–27:20 (2016). Special Issue on High Performance Computing (HPC) Software Design. <https://doi.org/10.3390/computation4030027>
71. Kunkel, J., Novikova, A., Betke, E.: Towards decoupling the selection of compression algorithms from quality constraints - an investigation of lossy compression efficiency. *Supercomput. Front. Innov.* **4**, 17–33 (2017). <https://doi.org/10.14529/jsfi170402>
72. Lambeck, K., Smither, C., Johnston, P.: Sea-level change, glacial rebound and mantle viscosity for northern Europe. *Geophys. J. Int.* **134**(1), 102–144 (1998). <https://doi.org/10.1046/j.1365-246x.1998.00541.x>
73. Larin, M., Reusken, A.: A comparative study of efficient iterative solvers for generalized Stokes equations. *Numer. Linear Algebra Appl.* **15**(1), 13–34 (2008). <https://doi.org/10.1002/nla.561>
74. Leitenmaier, L.: Data compression for simulation data from Earth mantle convection. *Lehrstuhl für Informatik 10 (Systemsimulation), Friedrich-Alexander-Universität Erlangen-Nürnberg. Bachelor Thesis* (2014)
75. Lengauer, C., Apel, S., Bolten, M., Chiba, S., Rüde, U., Teich, J., Größlinger, A., Hannig, F., Köstler, H., Claus, L., Grebhahn, A., Groth, S., Kronawitter, S., Kuckuk, S., Rittich, H., Schmitt, C., Schmitt, J.: ExaStencils: Advanced Multigrid Solver Generation (In this volume)
76. Lengauer, C., Apel, S., Bolten, M., Größlinger, A., Hannig, F., Köstler, H., Rüde, U., Teich, J., Grebhahn, A., Kronawitter, S., Kuckuk, S., Rittich, H., Schmitt, C.: ExaStencils: Advanced stencil-code engineering. In: *Euro-Par 2014: Parallel Processing Workshops. Lecture Notes in Computer Science*, vol. 8806, pp. 553–564. Springer, Berlin (2014). https://doi.org/10.1007/978-3-n-2_47
77. Maitre, J.F., Musy, F., Nigon, P.: A fast solver for the Stokes equations using multigrid with a Uzawa smoother. In: *Advances in multigrid methods* (Oberwolfach, 1984). *Notes on Numerical Fluid Mechanics*, vol. 11, pp. 77–83. Braunschweig, Vieweg (1985)
78. May, D.A., Brown, J., Pourhiet, L.L.: A scalable, matrix-free multigrid preconditioner for finite element discretizations of heterogeneous Stokes flow. *Comp. Meth. Appl. Mech. Engg.* **290**, 496–523 (2015). <https://doi.org/10.1016/j.cma.2015.03.014>

79. May, D., Sanan, P., Rupp, K., Knepley, M., Smith, B.: Extrem scale multigrid components within PETSc. In: Proceedings of the Platform for Advanced Scientific Computing Conference, PASC (2016)
80. Mitrovica, J.X.: Haskell [1935] revisited. *J. Geophys. Res.* **101**(B1), 555 (1996). <https://doi.org/10.1029/95JB03208>
81. Mitrovica, J.X., Forte, A.M.: A new inference of mantle viscosity based upon joint inversion of convection and glacial isostatic adjustment data. *Earth Planet. Sci. Lett.* **225**(1–2), 177–189 (2004). <https://doi.org/10.1016/j.epsl.2004.06.005>
82. Mitrovica, J.X., Wahr, J.: Ice age earth rotation. *Ann. Rev. Earth Planet. Sci.* **39**, 577–616 (2011). <https://doi.org/10.1146/annurev-earth-040610-133404>
83. Müller, R.D., Sdrolias, M., Gaina, C., Roest, W.R.: Age, spreading rates, and spreading asymmetry of the world's ocean crust. *Geochem. Geophys. Geosyst.* **9**(4), 1525–2027 (2008)
84. Price, M.G., Davies, J.H.: Profiling the robustness, efficiency and limits of the forward-adjoint method for 3-D mantle convection modelling. *Geophys. J. Int.* **212**(2), 1450–1462 (2017). <https://doi.org/10.1093/gji/ggx489>
85. Ricard, Y., Wuming, B.: Inferring the viscosity and the 3-D density structure of the mantle from geoid, topography and plate velocities. *Geophys. J. Int.* **105**(3), 561–571 (1991). <https://doi.org/10.1111/j.1365-246X.1991.tb00796.x>
86. Ricard, Y., Spada, G., Sabadini, R.: Polar wandering of a dynamic earth. *Geophys. J. Int.* **113**(2), 284–298 (1993). <https://doi.org/10.1111/j.1365-246X.1993.tb00888.x>
87. Richards, M.A., Hager, B.H.: Geoid anomalies in a dynamic earth. *J. Geophys. Res.* **89**(B7), 5987–6002 (1984). <https://doi.org/10.1029/JB089iB07p05987>
88. Richards, M.A., Lenardic, A.: The Cathles parameter (Ct): a geodynamic definition of the asthenosphere and implications for the nature of plate tectonics. *Geochem. Geophys. Geosyst.* **19**(12), 4858–4875 (2018). <https://doi.org/10.1029/2018GC007664>
89. Richards, M., Bunge, H., Ricard, Y., Baumgardner, J.: Polar wandering in mantle convection models. *Geophys. Res. Lett.* **26**(12), 1777–1780 (1999). <https://doi.org/10.1029/1999GL900331>
90. Rüde, U.: Mehrgittermethode—Grundlage der computergestützten Wissenschaften. *Informatik Spektrum* **42**(2), 138–143 (2019)
91. Rüde, U., Willcox, K., McInnes, L.C., Sterck, H.D.: Research and education in computational science and engineering. *SIAM Rev.* **60**(3), 707–754 (2018). <https://doi.org/10.1137/16M1096840>
92. Rudi, J., Malossi, A.C.I., Isaac, T., Stadler, G., Gurnis, M., Staar, P.W.J., Ineichen, Y., Bekas, C., Curioni, A., Ghattas, O.: An extreme-scale implicit solver for complex PDEs: Highly heterogeneous flow in Earth's Mantle. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, SC '15, pp. 5:1–5:12. ACM, New York (2015). <https://doi.org/10.1145/2807591.2807675>
93. Rudolph, M.L., Lekić, V., Lithgow-Bertelloni, C.: Viscosity jump in Earth's mid-mantle. *Science* **350**(6266), 1349–1352 (2015). <https://doi.org/10.1126/science.aad1929>
94. Schmitt, C., Kuckuk, S., Hannig, F., Köstler, H., Teich, J.: ExaSlang: A domain-specific language for highly scalable multigrid solvers. In: Proceedings of the Fourth International Workshop on Domain-Specific Languages and High-Level Frameworks for High Performance Computing (WOLFHP), pp. 42–51. IEEE Computer Society, Washington (2014)
95. Schöberl, J., Zulehner, W.: On Schwarz-type smoothers for saddle point problems. *Numer. Math.* **95**(2), 377–399 (2003). <https://doi.org/10.1007/s00211-002-0448-3>
96. Schornbaum, F., Rüde, U.: Extreme-scale block-structured adaptive mesh refinement. *SIAM J. Sci. Comp.* **40**(3), C358–C387 (2018)
97. Stals, L., Rüde, U., Weiß, C., Hellwagner, H.: Data local iterative methods for the efficient solution of partial differential equations. In: John, N., Andrew, G., Michael, T. (eds.) *Computational Techniques And Applications: Ctac 97-Proceedings Of The Eight Biennial Conference*. World Scientific, Singapore (1998)
98. Vanka, S.: Block-implicit multigrid solution of Navier-Stokes equations in primitive variables. *J. Comput. Phys.* **65**, 138–158 (1986). [https://doi.org/10.1016/0021-9991\(86\)90008-2](https://doi.org/10.1016/0021-9991(86)90008-2)

99. Verfürth, R.: A multilevel algorithm for mixed problems. *SIAM J. Numer. Anal.* **21**, 264–271 (1984). <https://doi.org/10.1137/0721019>
100. Vynnytska, L., Bunge, H.: Restoring past mantle convection structure through fluid dynamic inverse theory: regularisation through surface velocity boundary conditions. *GEM - Int. J. Geomath.* **6**(1), 83–100 (2014). <https://doi.org/10.1007/s13137-014-0060-6>
101. Waluga, C., Wohlmuth, B., Rüde, U.: Mass-corrections for the conservative coupling of flow and transport on collocated meshes. *J. Comp. Phys.* **305**, 319–332 (2016). <https://doi.org/10.1016/j.jcp.2015.10.044>
102. Weismüller, J., Gmeiner, B., Ghelichkhan, S., Huber, M., John, L., Wohlmuth, B., Rüde, U., Bunge, H.P.: Fast asthenosphere motion in high-resolution global mantle flow models. *Geophys. Res. Lett.* **42**(18), 7429–7435 (2015). <https://doi.org/10.1002/2015GL063727>
103. Williams, S., Waterman, A., Patterson, D.: Roofline: An insightful visual performance model for multicore architectures. *Commun. ACM* **52**(4), 65–76 (2009). <https://doi.org/10.1145/1498765.1498785>
104. Wobker, H., Turek, S.: Numerical studies of Vanka-type smoothers in computational solid mechanics. *Adv. Appl. Math. Mech.* **1**(1), 29–55 (2009)
105. Zhong, S., McNamara, A., Tan, E., Moresi, L., Gurnis, M.: A benchmark study on mantle convection in a 3-D spherical shell using CitcomS. *Geochem. Geophys. Geosyst.* **9**, Q10017 (2008). <https://doi.org/10.1029/2008GC002048>
106. Zulehner, W.: A class of smoothers for saddle point problems. *Computing* **65**, 227–246 (2000). <https://doi.org/10.1007/s006070070008>
107. Zulehner, W.: Analysis of iterative methods for saddle point problems: a unified approach. *Math. Comp.* **71**(238), 479–505 (2002)

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.