

Algebraic multigrid methods for saddle point systems arising from mortar contact formulations

Tobias A. Wiesner¹ | Matthias Mayr^{2,3}  | Alexander Popp²  |
Michael W. Gee⁴  | Wolfgang A. Wall⁵ 

¹Leica Geosystems AG, Heerbrugg, Switzerland

²Institute for Mathematics and Computer-Based Simulation, University of the Bundeswehr Munich, Neubiberg, Germany

³Data Science & Computing Lab, University of the Bundeswehr Munich, Neubiberg, Germany

⁴Mechanics & High Performance Computing Group, Technical University of Munich, Garching, Germany

⁵Institute for Computational Mechanics, Technical University of Munich, Garching, Germany

Correspondence

Matthias Mayr, Institute for Mathematics and Computer-Based Simulation, University of the Bundeswehr Munich, Werner-Heisenberg-Weg 39, D-85577 Neubiberg, Germany.
Email: matthias.mayr@unibw.de

Abstract

In this article, a fully aggregation-based algebraic multigrid strategy is developed for nonlinear contact problems of saddle point type using a mortar finite element approach. While the idea of extending multigrid methods to saddle point systems can already be found, for example, in the context of Stokes and Oseen equations in literature, the main contributions of this work are (i) the development and open-source implementation of an interface aggregation strategy specifically suited for generating Lagrange multiplier aggregates that are required for coupling structural equilibrium equations with contact constraints and (ii) a review of saddle point smoothers in the context of constrained interface problems. The new interface aggregation strategy perfectly fits into an aggregation-based multigrid framework and can easily be combined with segregated transfer operators, which allow to preserve the saddle point structure on the coarse levels. Further analysis provides insight into saddle point smoothers applied to contact problems, while numerical experiments illustrate the robustness of the new method. We have implemented the proposed algorithm within the MUELU package of the open-source TRILINOS project. Numerical examples demonstrate the robustness of the proposed method in complex dynamic contact problems as well as its scalability up to 23.9 million unknowns on 480 MPI ranks.

KEYWORDS

algebraic multigrid methods, contact mechanics, iterative linear solvers, mortar methods, preconditioning

1 | INTRODUCTION

Many engineering applications require the simulation of large-scale contact problems. Therefore, it is not surprising that recent years have seen significant progress in modeling and simulation of contact interaction and its associated phenomena, such as friction,¹⁻³ wear,⁴⁻⁷ adhesion,^{8,9} or multiscale contact phenomena.^{10,11} This is particularly true with regard to robust finite element based discretization techniques for finite deformations and efficient nonlinear

This work was partially performed while these authors were affiliated with the Institute for Computational Mechanics, Technical University of Munich, Boltzmannstr. 15, D-85748 Garching, Germany.

This is an open access article under the terms of the Creative Commons Attribution License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.

© 2021 The Authors. *International Journal for Numerical Methods in Engineering* published by John Wiley & Sons Ltd.

solution algorithms. Above all, mortar finite element methods—originally introduced in the context of domain decomposition^{12,13}—are meanwhile well established as a basis for state-of-the-art contact formulations and widely accepted among researchers as being superior to more classical discretization techniques, such as the node-to-segment (NTS) method, the Gauss-point-to-segment (GPTS) method, and other collocation based approaches.^{14–18}

Nowadays, constraint enforcement in the context of mortar methods is often based on a regularized Lagrange multiplier scheme or an augmented Lagrange method instead of a simple, yet often insufficient penalty approach. Independent from the actual details of the constraint enforcement implementation, the discrete Lagrange multipliers constitute an additional set of degrees of freedom in the mortar finite element contact formulation. When using a dual mortar approach,^{19–25} the discrete Lagrange multiplier basis is chosen based on a biorthogonality condition with the underlying finite element basis. This allows for the localization of the contact constraints and, thus, from a more algebraic point of view, for the trivial condensation of the additional Lagrange multiplier degrees of freedom from the final linearized systems of equations. If such a static condensation is not desired or not feasible (e.g., when choosing a standard basis rather than dual basis functions for the Lagrange multipliers, see, e.g., References 24,26), the linear system remains in its generalized saddle point format arising from the contact constraint equations. Both the standard and the dual mortar approach have become increasingly popular in recent years, with new contributions focusing for example on higher-order finite element interpolation,^{27,28} isogeometric mortar methods,^{29–33} or improved robustness of the solution algorithms,^{34–36} to name only a few particularly active research directions.

It is striking, however, that almost all current research endeavors concerned with mortar finite element methods for contact mechanics focus exclusively on the modeling of various contact phenomena. Yet, for large-scale and industrial applications the appropriate modeling of contact problems is not sufficient. In fact, the demand for efficient solution strategies tailored to the specifics of contact simulations is eminent in order to achieve optimal overall performance. Whereas one could use parallel direct solvers to solve the linear systems, they are not an option for very large problems. Iterative solvers for sparse systems (e.g., References 37,38) combined with good preconditioning strategies are a far better choice with respect to computational resources. In particular, multigrid methods^{39,40} are known to be among the most efficient solution and preconditioning strategies, at least for certain classes of problems.

From the perspective of the linear solvers and multigrid-based preconditioners, the condensation of the Lagrange multipliers seems to be very attractive, since it allows to circumvent the more sophisticated saddle point formulation. For contact problems though, we have experienced that the resulting linear systems after condensation suffer from some challenging matrix properties which cause severe convergence problems for standard preconditioning techniques. In particular, the matrices tend to be nondiagonally dominant due to different (local) coordinate systems that are typically used for the formulation of the structural equilibrium equations and the contact constraints. In our previous work,⁴¹ we have developed multilevel preconditioners that address such issues and are specifically tailored to contact problems using the dual mortar method in a condensed formulation.

On the other hand, multigrid methods already have been successfully applied to saddle point problems as they arise from different applications (e.g., Stokes flow⁴² or incompressible Navier–Stokes problems^{43,44}) and even in the context of mortar finite element methods.^{45,46} The multigrid theory for this particular class of saddle point problems has evolved starting from special multigrid methods for mortar finite element methods (e.g., References 47–50) to mortar finite element methods in saddle point formulation (e.g., References 51,52). Based on these ideas, specific multigrid methods for contact problems in saddle point formulation have been developed in Reference 53. However, most of the literature available on multigrid for mortar finite element methods and contact problems in saddle point formulations is primarily on geometric multigrid methods with abundant work on saddle point smoothers (cf., Reference 54). A first algebraic multigrid preconditioner for mortar-based contact problems has been proposed by Adams,⁵⁵ performing standard aggregation on the graph of an auxiliary matrix imitating the Lagrange multipliers. Alternatively, multigrid methods for contact problems not requiring an outer iteration loop or active set strategy have been developed in References 56,57.

In this article, we address the case of mortar-based contact problems in saddle point formulation and show how to tailor iterative solvers with algebraic multigrid preconditioners to such problems. In contrast to geometric multigrid methods, algebraic multigrid methods (e.g., Reference 58) do not rely on geometric user-provided mesh information, but use only purely algebraic information from the fine-level matrix. Since static condensation of Lagrange multiplier unknowns is not required, our approach is applicable to mortar methods using both standard or dual shape functions. The proposed multigrid method is based on the (smoothed) aggregation algebraic multigrid algorithms (cf., References 59–63) with special extensions for block matrices and some minor contact-specific adaptations. We propose a novel aggregation strategy for

the discrete Lagrange multiplier unknowns along the contact interface, which we consider simpler to implement, computationally less expensive, and more intuitive for contact problems compared with the ideas from Reference 55. Inspired by our prior work on fluid–structure interaction,^{64,65} where we have investigated the beneficial effect of satisfying interface constraints within the preconditioner, we will then use segregated transfer operators suitable for block matrices to transfer and incorporate the contact constraints in all coarse levels. We analyze various Schur complement block smoothers and assess their suitability for satisfying the contact constraints. Finally, we demonstrate and assess the performance of the proposed preconditioner in several three-dimensional examples.

The remainder of this article is organized as follows: Section 2 provides a brief introduction to mortar methods for finite deformation contact problems in saddle point formulation. After the basic notation is introduced, we specifically present the resulting linear system that is arising if the discrete Lagrange multipliers are explicitly included into the set of unknowns to be solved for. After a brief introduction to the general idea of multigrid methods, Section 3 describes our strategy to tailor a multigrid preconditioner to contact problems in saddle point formulation. It comprises the coarsening of the mortar contact constraints as detailed in Section 4 as well as suitable block smoothers as discussed in Section 5. Finally, Section 6 presents numerical examples that showcase the robustness, scalability, and performance of the proposed multigrid preconditioners, before we close with some final remarks.

2 | MORTAR METHODS FOR FINITE DEFORMATION CONTACT

As this article is concerned with preconditioning of the system of linear equations arising from contact problems, just a brief summary to the contact formulation and discretization is given here. For a detailed presentation, the reader is referred to our previous work.²³

2.1 | Problem formulation and governing equations

We consider two solid bodies, which are represented by $\Omega_0^{(1)}, \Omega_0^{(2)} \subset \mathbb{R}^d$ with $d \in \{2, 3\}$ in the reference configuration. Their surfaces $\partial\Omega_0^{(i)}, i \in \{1, 2\}$ are decomposed into three disjoint subsets $\Gamma_D^{(i)}, \Gamma_N^{(i)}$, and $\Gamma_c^{(i)}$ denoting the Dirichlet boundary, the Neumann boundary, and the potential contact interface with unknown contact tractions $\mathbf{t}_c^{(i)}$, respectively. The solid bodies themselves are governed by nonlinear elasticity. Since we are only interested in the algebraic block structure of the final system of equations after discretization and linearization, it is sufficient to discuss a quasi-static contact problem with only two deformable bodies.

In order to describe the contact phenomenon, we state the Hertz–Signorini–Moreau conditions

$$g_n \geq 0 \quad \wedge \quad p_n \leq 0 \quad \wedge \quad g_n p_n = 0. \quad (1)$$

Therein, g_n defines a so-called gap function, which measures the distance of a point on the slave interface $\gamma_c^{(S)}$ to the projected corresponding point on the master side $\gamma_c^{(M)}$ of the contact interface in the current configuration. Furthermore, p_n denotes the normal contact traction. In the mathematical formulation, one introduces the negative slave side contact traction $\mathbf{t}_c^{(1)}$ as Lagrange multiplier, that is, $\lambda = -\mathbf{t}_c^{(1)}$. Using \mathbf{n} to denote the outward unit normal vector, the normal part of the contact stress can be denoted by $\lambda_n := \lambda^\top \mathbf{n}$ and the tangential part by $\lambda_\tau := \lambda - \lambda_n \mathbf{n}$.

We employ the usual function spaces $\mathcal{V}^{(i)}$ and $\mathcal{V}^{(i)}$ for the displacement field \mathbf{u} of the solid body and its weighting function \mathbf{v} , respectively. Furthermore, a suitable function space \mathcal{M}_+ for the Lagrange multiplier field λ and its weighting function $\boldsymbol{\mu}$ is assumed. The weak form of the governing equations then reads: Find $(\mathbf{u}, \lambda) \in \mathcal{U} \times \mathcal{M}_+$ such that

$$-\delta\mathcal{W}_{\text{int,ext}} + \int_{\gamma_c^{(S)}} \lambda (\mathbf{v}^{(1)} - \mathbf{v}^{(2)}) \, dA = 0, \quad \forall \mathbf{v} \in \mathcal{V}, \quad (2a)$$

$$\int_{\gamma_c^{(S)}} (\boldsymbol{\mu}_n - \lambda_n) g_n \, dA \geq 0, \quad \forall \boldsymbol{\mu} \in \mathcal{M}_+. \quad (2b)$$

Herein, the internal and external virtual work contributions $\delta\mathcal{W}_{\text{int,ext}}$ are defined as usual in nonlinear solid mechanics (eg, Reference 66) and, thus, further details are omitted. The second term in (2a) can be identified as contact virtual work $\delta\mathcal{W}_c$ and the expression in (2b) as variational inequality formulation of the contact constraints. An extension to frictional contact based on Coulomb's law is straightforward and can be found in our previous work¹ for example.

2.2 | Finite element discretization

For spatial discretization of the displacement field, either isoparametric finite elements with first-order and second-order Lagrange interpolation or isogeometric analysis (IGA) with NURBS-based shape functions are employed. After discretization, the discrete representation of displacement unknowns is given by the nodal degrees of freedom (DOFs)

$$\mathbf{u} = [\mathbf{u}_{\mathcal{N}_1}, \mathbf{u}_S, \mathbf{u}_M, \mathbf{u}_{\mathcal{N}_2}]^T.$$

Therein, the $\mathbf{u}_{\mathcal{N}_i}$, $i \in \{1, 2\}$, contain all degrees of freedom associated with the mesh nodes of the corresponding solid body without the nodes at the contact interface, where we use the convention that indices 1 and 2 denote the slave and master “body,” respectively. The degrees of freedom associated with the contact interface on the slave and master side are represented by \mathbf{u}_S and \mathbf{u}_M , respectively.

Since (2) represents a mixed variational form, we also have to discretize the Lagrange multiplier field λ . We choose to follow a mortar approach for its mathematical properties and its superiority to other schemes.¹⁴⁻¹⁸ As usual for mortar methods, the Lagrange multiplier field is discretized on the slave side contact interface $\gamma_c^{(S)}$ in the current configuration. We either use standard ansatz functions, that is, Lagrange polynomials with a trace space relation with the underlying volume element, or dual shape functions. The latter satisfy a biorthogonality property and, thus, allow for a computationally cheap condensation of the additional unknown Lagrange multipliers from the final system of equations. For details on dual basis functions in the context of mortar-based contact discretizations, we refer to References 24,27,67. Their interplay with preconditioners for iterative linear solvers has been discussed in our previous work.⁴¹ The vector of discrete Lagrange multipliers is now referred to as λ . A schematic mesh illustrating interior, slave interface, and master interface nodes is sketched in Figure 1.

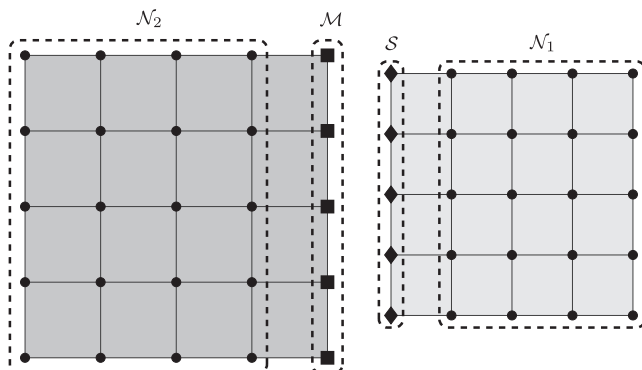
The final spatially discretized formulation of the quasi-static frictionless problem (2) using the nodal vector representation now emerges as

$$\mathbf{f}_{\text{int}}(\mathbf{u}) + \mathbf{f}_{\text{co}}(\mathbf{u}, \lambda) = \mathbf{f}_{\text{ext}}, \tag{3a}$$

$$(\tilde{g}_{n,h})_j \geq 0, \quad (\lambda_n)_j \geq 0, \quad (\tilde{g}_{n,h})_j (\lambda_n)_j = 0, \quad j = 1, \dots, n^{(S)}, \tag{3b}$$

$$(\lambda_\tau)_j = \mathbf{0}, \quad j = 1, \dots, n^{(S)}. \tag{3c}$$

The internal forces $\mathbf{f}_{\text{int}}(\mathbf{u})$ and external forces \mathbf{f}_{ext} are common in nonlinear finite element methods and need no further explanation. The discrete vector of contact forces \mathbf{f}_{co} is computed based on two mortar matrices D and M, arising from the integral over the slave interface $\gamma_c^{(S)}$ in (2a), and the discrete Lagrange multiplier vector λ . For details regarding the computation of D and M, see References 15,23 for example. Using $(\tilde{g}_{n,h})_j$ to denote the discrete weighted gap function at node j , a closer look at the discrete contact constraints reveals that (3b) basically represents a discrete version of the Karush–Kuhn–Tucker (KKT) type conditions in (1) with an additional weighting based on the Lagrange multiplier shape functions ψ_j , while the nodal enforcement of frictionless sliding in (3c) is straightforward anyway.



- Inner nodes with displacement degrees of freedom $\mathbf{u}_{\mathcal{N}_1} \in \Omega_0^{(1)} \setminus \Gamma_c^{(1)}$ and $\mathbf{u}_{\mathcal{N}_2} \in \Omega_0^{(2)} \setminus \Gamma_c^{(2)}$
- ◆ Nodes at slave contact interface $\Gamma_c^{(S)} = \Gamma_c^{(1)}$ with displacement degrees of freedom \mathbf{u}_A and \mathbf{u}_T and Lagrange multipliers λ
- Nodes at master contact interface $\Gamma_c^{(M)} = \Gamma_c^{(2)}$ with displacement degrees of freedom \mathbf{u}_M

FIGURE 1 Schematic mesh illustrating interior, master and slave interface nodes

Since the discrete contact constraints summarized in (3b) are still formulated as inequalities, an active set strategy usually referred to as primal-dual active set strategy (PDASS) is needed in addition to the usual nonlinear solution procedure to identify the currently active and inactive contact regions \mathcal{A} and $\mathcal{I} = S \setminus \mathcal{A}$, respectively. It has been demonstrated in References 68-70 that the PDASS can equivalently be interpreted as a semi-smooth Newton method, thus allowing for an integrated treatment of all nonlinearities (including the search for the active set) within one single-Newton–Raphson-type iteration loop. Meanwhile, many successful applications to small and large deformation contact problems can be found in the literature.^{1,20,21,23}

2.3 | Algebraic formulation of linear systems

For efficient iterative solution strategies based on multigrid methods for nonlinear contact problems, one is primarily interested in the structure of the linear systems arising in each nonlinear iteration step of the underlying Newton–Raphson scheme. For the sake of brevity, details on the linearization process and on the Newton–Raphson procedure are omitted here and the interested reader is instead referred to References 22,23.

Consistent linearization of (3) and a subsequent update of the active set \mathcal{A} and inactive set \mathcal{I} yields the system

$$\begin{pmatrix} K_{\mathcal{N}_1\mathcal{N}_1} & K_{\mathcal{N}_1\mathcal{M}} & 0 & 0 & 0 & 0 & 0 & 0 \\ K_{\mathcal{M}\mathcal{N}_1} & K_{\mathcal{M}\mathcal{M}} & K_{\mathcal{M}\mathcal{I}} & K_{\mathcal{M}\mathcal{A}} & 0 & -\mathbf{M}_I^T & -\mathbf{M}_A^T & 0 \\ 0 & K_{\mathcal{I}\mathcal{M}} & K_{\mathcal{I}\mathcal{I}} & K_{\mathcal{I}\mathcal{A}} & K_{\mathcal{I}\mathcal{N}_2} & \mathbf{D}_{\mathcal{I}\mathcal{I}}^T & \mathbf{D}_{\mathcal{I}\mathcal{A}}^T & 0 \\ 0 & K_{\mathcal{A}\mathcal{M}} & K_{\mathcal{A}\mathcal{I}} & K_{\mathcal{A}\mathcal{A}} & K_{\mathcal{A}\mathcal{N}_2} & \mathbf{D}_{\mathcal{A}\mathcal{I}}^T & \mathbf{D}_{\mathcal{A}\mathcal{A}}^T & 0 \\ 0 & 0 & K_{\mathcal{N}_2\mathcal{I}} & K_{\mathcal{N}_2\mathcal{A}} & K_{\mathcal{N}_2\mathcal{N}_2} & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & \mathbf{I} & 0 & 0 \\ 0 & \mathbf{N}_{\mathcal{M}} & \mathbf{N}_{\mathcal{I}} & \mathbf{N}_{\mathcal{A}} & 0 & 0 & 0 & 0 \\ 0 & 0 & \mathbf{F}_{\mathcal{I}} & \mathbf{F}_{\mathcal{A}} & 0 & 0 & \mathbf{T}_{\mathcal{A}} & 0 \end{pmatrix} \begin{bmatrix} \Delta \mathbf{u}_{\mathcal{N}_1} \\ \Delta \mathbf{u}_{\mathcal{M}} \\ \Delta \mathbf{u}_{\mathcal{I}} \\ \Delta \mathbf{u}_{\mathcal{A}} \\ \Delta \mathbf{u}_{\mathcal{N}_2} \\ \Delta \lambda_{\mathcal{I}} \\ \Delta \lambda_{\mathcal{A}} \end{bmatrix} = - \begin{bmatrix} \mathbf{r}_{\mathcal{N}_1}^\mu \\ \mathbf{r}_{\mathcal{M}}^\mu \\ \mathbf{r}_{\mathcal{I}}^\mu \\ \mathbf{r}_{\mathcal{A}}^\mu \\ \mathbf{r}_{\mathcal{N}_2}^\mu \\ \mathbf{r}_{\mathcal{I}}^\lambda \\ \mathbf{r}_{\mathcal{A}}^{\lambda,n} \\ \mathbf{r}_{\mathcal{A}}^{\lambda,\tau} \end{bmatrix} \quad (4)$$

to be solved in every nonlinear iteration. The 2×2 block matrix indicated by the dashed lines in (4) describes a linear system with a typical generalized saddle point structure. The upper left block consists of the entries of the tangential stiffness matrix (i.e., linearized internal forces) as well as linearizations of contact forces w.r.t. displacement degrees of freedom \mathbf{u} . The upper right block mirrors the discrete contact operator $\mathbf{C}(\mathbf{u})$, that is, basically the two mortar matrices \mathbf{D} and \mathbf{M} , representing the linearizations of the contact forces w.r.t. the Lagrange multiplier unknowns λ . The kinematic constraints are incorporated in the bottom left block. The very simple sixth block row emerges from (3b) and (3c) for inactive nodes, while the last block row imposes frictionless sliding in the directions tangential to the contact interface.

The distinct pattern of zero entries in the upper left block reveals that the two solid bodies (indices \mathcal{N}_1 and \mathcal{N}_2) are indeed only coupled through the slave and master sides of the contact interface (indices \mathcal{S} and \mathcal{M}). Even though formulated for two solid bodies, the generalization to n solid bodies is straightforward and only a matter of notation.

In case of dual shape functions, the matrix \mathbf{D} reduces to a diagonal matrix and, thus, allows for a cheap condensation of the Lagrange multiplier unknowns. Algebraic multigrid preconditioners for this type of condensed system have been proposed in our earlier paper.⁴¹ Furthermore, matrices $\mathbf{N}_{\mathcal{M}}$, $\mathbf{N}_{\mathcal{I}}$, and $\mathbf{N}_{\mathcal{A}}$ denote the linearizations of the weighted gap function of (3b) at all active contact nodes. Finally, linearizations of the frictionless sliding condition (3c) are referred to by matrices $\mathbf{F}_{\mathcal{I}}$, $\mathbf{F}_{\mathcal{A}}$, and $\mathbf{T}_{\mathcal{A}}$, respectively.

Note that the given matrix has eight block rows but only seven block columns in our notation in order to emphasize that the normal and tangential parts of the contact constraints for active nodes are considered separately, that is, these two rows contain consistent linearizations of the active branch of (3b) and of (3c). Again, we point out that this separate notation is possible due to the fact that a local convective coordinate system is employed for evaluating the contact constraints/Lagrange multiplier weights μ , while the standard Cartesian frame is still applied for the discrete Lagrange multiplier values λ as well as the displacement unknowns. Yet, of course, the system matrix remains a square matrix with the total numbers of rows and columns being identical. The discrete vector $\mathbf{g}_{\mathcal{A}}$ contains all weighted gap values $(\tilde{g}_{n,h})_j$ associated with the active nodes at the contact interface.

For ease of notation, the following short block notation is used in the remainder of the manuscript:

$$\begin{pmatrix} \mathbf{K} & \mathbf{C}_1^T \\ \mathbf{C}_2 & -\mathbf{Z} \end{pmatrix} \begin{bmatrix} \Delta \mathbf{u} \\ \Delta \lambda \end{bmatrix} = - \begin{bmatrix} \mathbf{r}^u \\ \mathbf{r}^\lambda \end{bmatrix}. \quad (5)$$

3 | MULTIGRID SCHEME FOR CONTACT PROBLEMS IN SADDLE POINT FORMULATION

Although multigrid methods can be used as standalone solvers for linear systems, they are usually incorporated into an iterative linear solver as a preconditioning method. Throughout this article, we use a preconditioned generalized minimal residual (GMRES) solver⁷¹ with one multigrid V-cycle sweep for preconditioning. A general introduction into the idea of preconditioning is beyond the scope of this article. The reader is referred to the literature, for example, Reference 72.

3.1 | Algebraic multigrid methods in a nutshell

Multigrid methods are based on the finding that many well-known and computationally cheap iterative methods (e.g., relaxation based iterative methods such as Jacobi or Gauss–Seidel methods) to solve linear systems $\mathbf{Ax} = b$ effectively damp the high frequency part of an error vector but are less effective in damping out the low frequency error modes. Multigrid methods heavily make use of this smoothing property by applying such cheap smoothing methods on different coarsened representations of the original fine-level problem.

3.1.1 | Basic multigrid cycle and algorithm

The multigrid algorithm given in Figure 2(A) is briefly described as follows: on each multigrid level ℓ , a level smoothing algorithm \mathcal{S}_ℓ performs ν_1 presmoothing sweeps before the residual vector r is transferred to the next coarser level $\ell + 1$ using the restriction operator \mathbf{R} . After the coarse-level problem has been solved on the coarsest level, the correction c is then prolonged using the prolongation operator \mathbf{P} and the solution vector is smoothed using ν_2 postsmoothing sweeps. Figure 2(B) illustrates this basic multigrid V-cycle, exemplifying a three-level setting. As one can see from Figure 2(B), applying a multigrid method basically means applying level smoothers on coarse representations \mathbf{A}_ℓ of the fine-level problem \mathbf{A}_0 .

3.1.2 | Algebraic multigrid methods

There are different strategies for defining the transfer operators \mathbf{P} and \mathbf{R} which are necessary to generate coarse-level matrices \mathbf{A}_ℓ ($\ell > 0$). For algebraic multigrid (AMG), the fine-level operator \mathbf{A}_0 is sufficient to generate coarse-level

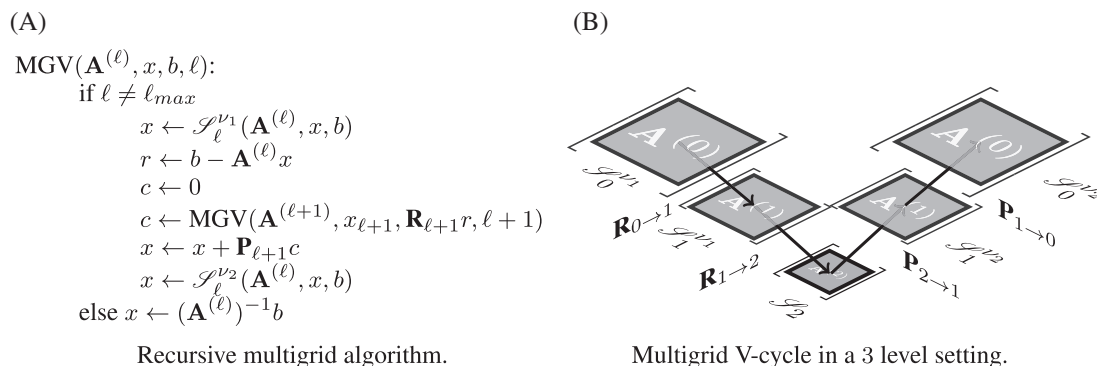


FIGURE 2 Multigrid algorithm and V-cycle

matrices \mathbf{A}_ℓ . An important class of AMG is given with the *smoothed aggregation AMG* which is based on so-called aggregates (see, e.g., References 60,73). The fine-level nodes are agglomerated and put into aggregates, which then represent “supernodes” on the next coarser level. The aggregation information together with the null space information of \mathbf{A}_0 is used to construct the corresponding *tentative transfer operators* $\hat{\mathbf{R}}$ (for restriction) and $\hat{\mathbf{P}}$ (for prolongation). Transfer operators are used to restrict the fine-level residual to the next coarser level and to interpolate the coarse-level correction to the next finer level via prolongation. For an efficient multigrid method, the interaction of fine and coarse levels tackles those error modes, which appear as low-frequency modes on the fine level and cannot effectively be reduced by iterative smoothing methods on the fine level, but resemble high-frequency modes on a coarser level, such that iterative smoothing methods are effective again. On the coarsest level, a direct solver can take care of the remaining error modes. For smoothed aggregation multigrid (cf., References 60,74), the prolongation operator is found by applying one smoothing sweep with a damped Jacobi iteration using

$$\mathbf{P} = \hat{\mathbf{P}} - \omega \mathbf{D}^{-1} \mathbf{A} \hat{\mathbf{P}} \quad (6)$$

with \mathbf{D} being the diagonal part of \mathbf{A} and a damping parameter $\omega > 0$. Depending on the symmetry of the system matrix \mathbf{A} , the restriction operator is either chosen as $\mathbf{R} = \mathbf{P}^T$ or smoothed independently using, for example a Petrov–Galerkin approach (cf., Reference 63).

3.2 | Algebraic multigrid methods for block matrices

Block matrices usually arise if multiple types of equations are coupled together. In the present context of contact problems in saddle point formulation, two types of equations, namely the balances of linear momentum of the solid bodies and the contact constraints, are connected via the off-diagonal blocks in (5). Similarly, multiphysics problems also yield block matrices where the coupling between different physical fields manifests itself in the off-diagonal blocks of the monolithic system matrix.

From a multigrid perspective, the most important question is where to consider the coupling between the different equations within the overall solver layout. In general, there are only two possible strategies to apply multigrid ideas to coupled block systems:

Nested multigrid approach:

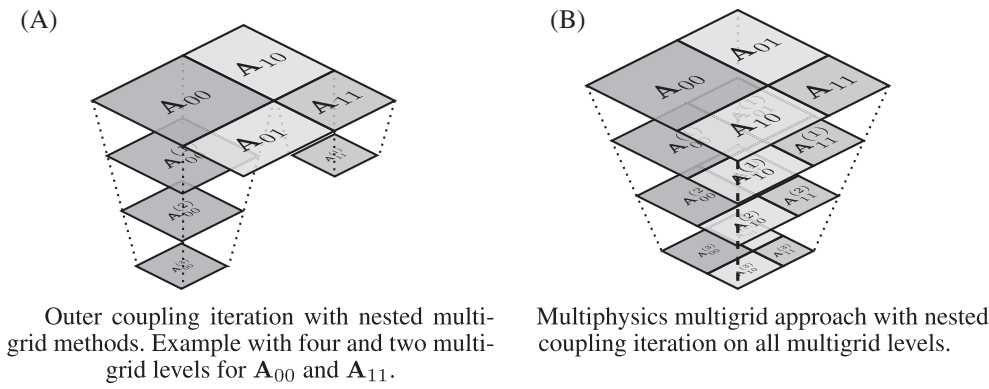
Multigrid methods can serve as local single-field smoothers or solvers within well-known block preconditioners such as the SIMPLE method (cf., Reference 75) and variants for Schur complement based preconditioners or the block Gauss–Seidel (BGS) method. The coupling of the different fields or variables is only considered on the finest level in the outer (SIMPLE or BGS) iteration. This approach is well known in the literature, for example, for the Navier–Stokes equations,^{43,58} fluid–structure interaction,^{64,76} or general n -field problems.⁷⁷ The implementation is very easy and allows to use existing multigrid components in a standalone fashion within the solver. A graphic representation of this approach is shown in Figure 3(A).

Fully coupled multigrid approach:

Truly monolithic algebraic multigrid methods aim at coarsening the fully coupled fine-level problem such that the block structure of the fine-level matrix is preserved and the coupling information is present on all coarser levels, cf., Figure 3(B). This is often achieved by using segregated transfer operators to preserve the characteristics of the sparsity pattern across all levels. Then, each level utilizes block smoothers to address the coupling. In Reference 78, a coupled AMG method is developed and analyzed for a stabilized mixed finite element discretization of the Oseen equations. Fully coupled multigrid methods for multiphysics problems have been described in References 64,77,79.

While the nested multigrid approach is easier to implement, it also allows for a high degree of modularity, since the multigrid hierarchies used to approximate the block inverses of the block smoother on the fine level can easily be swapped by any other method, either another type of multigrid algorithm, or another type of multilevel scheme (e.g.,

FIGURE 3 Multigrid for block matrices



based on domain decomposition), or even any single-level approach if the block is of moderate size and scalability is not deteriorated. This flexibility is particularly useful if the coupled blocks differ a lot in size or if the user wants to apply a highly optimized solver for an individual block. The fully coupled multigrid method does not offer such a degree of flexibility, yet it propagates the coupling conditions throughout the entire preconditioner. Thus, one expects a stronger and more robust preconditioning effect, since the coarse-level corrections are aware of the coupling conditions. This expectation will later be confirmed in the numerical experiments, where the number of iterations for the fully coupled scheme is lower and more independent of the active contact nodes than for the nested multigrid approach.

3.3 | Designing algebraic multigrid methods for contact problems

In the present context, we can interpret the mortar contact problem in saddle point formulation as the coupling of two types of equations: the structural equations and the contact equations which serve as constraint equations. Since the contact constraint equations are only defined along the contact interface, we can further classify the mortar contact problem as an interface-coupled problem (in contrast to volume-coupled problems). This information is important for the choice of coarsening strategy. The contact constraint equations are also responsible for the characteristic saddle point structure, which needs special attention when choosing an appropriate coupling algorithm between the structural equations and the contact constraints. Considering the class of fully coupled AMG schemes, the generalized saddle point problem (5) has to be preserved on all multigrid levels such that the contact constraints are considered on all levels. Due to the constraints, this will require Schur complement based level smoothers on all levels.

Altogether, the key ingredients for designing an algebraic multigrid method for contact problems in saddle point formulation are the coarsening strategy as proposed in Section 4 and the level smoother and the coupling iteration as detailed in Section 5.

4 | A COARSENING STRATEGY FOR MORTAR CONTACT CONSTRAINTS

4.1 | Segregated transfer operators

To keep the characteristic saddle point block structure (5) on all multigrid levels, the common approach is to use *segregated* transfer operators

$$\mathbf{P}_{\ell+1} = \begin{pmatrix} \mathbf{P}^u & 0 \\ 0 & \hat{\mathbf{P}}^\lambda \end{pmatrix}_{\ell+1} \quad \text{and} \quad \mathbf{R}_{\ell+1} = \begin{pmatrix} \mathbf{R}^u & 0 \\ 0 & \hat{\mathbf{R}}^\lambda \end{pmatrix}_{\ell+1}, \quad (7)$$

as, for example, introduced in References 55,80. The segregated block transfer operators (7) are put together from the transfer operator blocks for the different physical and mathematical fields. Here, \mathbf{P}^u and \mathbf{R}^u describe the transfer operator blocks corresponding to the stiffness matrix block \mathbf{K} in (5). The transfer operators $\hat{\mathbf{P}}^\lambda$ and $\hat{\mathbf{R}}^\lambda$ define the level transfer for the Lagrange multipliers.

The block diagonal structure in (7) guarantees that the primary displacement variables and the secondary Lagrange multipliers are not “mixed up” on the coarser levels. That is, the coarse-level matrix still has the same block structure with a clear distinction of momentum and constraint equations as for the fine-level problem since the columns and rows of the transfer operators $\mathbf{P}_{\ell+1}$ and $\mathbf{R}_{\ell+1}$ can be interpreted as some kind of basis functions.

For many volume-coupled problems, for example in thermo-structure-interaction problems,⁸¹ it is straightforward to generate $\hat{\mathbf{P}}^\lambda$ and $\hat{\mathbf{R}}^\lambda$ to be consistent with \mathbf{P}^u and \mathbf{R}^u . That is, in context of smoothed aggregation algebraic multigrid we just use the same aggregates for building \mathbf{P}^u and $\hat{\mathbf{P}}^\lambda$ (and the same for the restrictors, respectively).

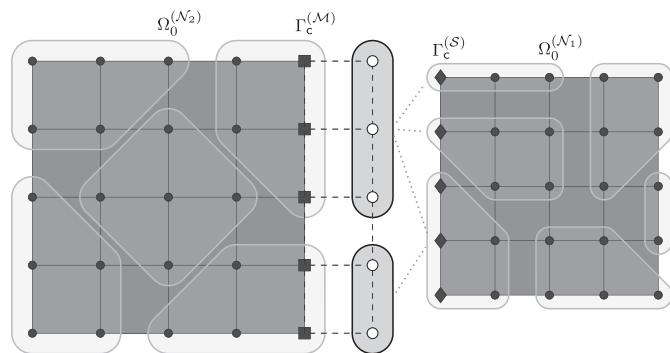
For interface-coupled problems with interface constraints it is more difficult. Due to the saddle point structure of (5), the nonzero pattern of the Z block is insufficient to generate valid aggregates for the Lagrange multipliers. Consequently, we need a special routine for finding aggregates for the Lagrange multipliers λ to be able to build the (nonsmoothed) transfer operators $\hat{\mathbf{P}}^\lambda$ and $\hat{\mathbf{R}}^\lambda$. Nevertheless it seems natural to reflect the aggregation information of the structural equations along the contact interface in the choice of the aggregates for the corresponding Lagrange multipliers λ .

4.2 | Aggregation strategy for displacement variables

In order to preserve the physics of the fine-level contact problem, it is important to keep the two solid bodies separated in the matrix representation on all coarse levels. Therefore, we apply the standard aggregation strategy to a modified K block from (5), where all off-diagonal entries representing connections between the two solid bodies are dropped—in particular the matrix blocks \mathbf{K}_{LM} , \mathbf{K}_{AM} , \mathbf{K}_{ML} , and \mathbf{K}_{MA} —in order to make sure that the resulting displacement aggregates \mathcal{A}_ℓ^u do not cross the contact interface (see Figure 4). Neglecting these blocks during aggregation guarantees that the two solid bodies are not melted together in the coarse matrix representation. We stress that the modified K is never formed explicitly, but rather the off-diagonal entries are dropped on the fly during the aggregation process.

4.3 | Aggregation strategy for Lagrange multipliers

In contrast to geometric multigrid methods, there is not so much literature on aggregation-based AMG methods for contact problems in saddle point formulation. The only publication, the authors are aware of covering all aspects of smoothed aggregation methods for structural contact problems in saddle point formulation, is Reference 55, which also discusses a special aggregation strategy for the Lagrange multipliers. To find aggregates \mathcal{A}_ℓ^λ for the Lagrange multipliers, Adams⁵⁵ proposes to apply the standard aggregation algorithm to the graph of a suitable matrix representing the Lagrange multipliers. However, this approach has some drawbacks: First, the graph used for the aggregation of the Lagrange multipliers λ has to be built explicitly to serve as input for the standard aggregation algorithm. Secondly, one has to run the aggregation algorithm sequentially both for the displacement degrees of freedom and for the Lagrange multipliers.



- Inner nodes with displacement degrees of freedom u_{N_1} and u_{N_2}
- ◆ Nodes at slave contact interface with displacement degrees of freedom u_A and u_T
- Virtual “nodes” for Lagrange multiplier degrees of freedom λ_A and λ_T
- Nodes at master contact interface with displacement degrees of freedom u_M
- Aggregate in \mathcal{A}_ℓ^u for displacement degrees of freedom u_{N_1} and u_{N_2}
- Aggregate in \mathcal{A}_ℓ^λ for Lagrange multipliers λ

FIGURE 4 Aggregation for contact example in saddle point formulation

For the second run of the aggregation method, one might have to use a different set of aggregation parameters to obtain optimal results, which further increases the complexity for the user. Algorithmically, the resulting aggregates \mathcal{A}_ℓ^λ for the Lagrange multipliers are built independently from the displacement aggregates \mathcal{A}_ℓ^u .

In this work, we propose a different approach to build aggregates \mathcal{A}_ℓ^λ for the Lagrange multipliers, which does not suffer from above drawbacks. Instead of explicitly building some helper matrix for the aggregation routine, interface aggregates \mathcal{A}_ℓ^λ for the Lagrange multipliers are directly generated using the aggregation information of the displacement variables (see Figure 4). The resulting interface aggregates for the Lagrange multipliers are by construction aligned with the corresponding displacement aggregates.

Algorithm 1. Aggregation algorithm for Lagrange multipliers

Procedure LagMultAggregation (\mathcal{A}_ℓ^u, D)

Initialize empty set and counter for aggregates \mathcal{A}_ℓ^λ

$\mathcal{A}_\ell^\lambda \leftarrow \emptyset, l \leftarrow 0$

Initialize empty mapping of displacement aggregates to Lagrange multiplier aggregates

$d(k) \leftarrow \emptyset \quad \forall k = 1, \dots, m_{\mathcal{A}_\ell^u}$

Loop over slave displacement DOFs (rows of D)

for $i \in D_S$ **do**

Find displacement node n^u id corresponding to displacement DOF i

$n^u \leftarrow n(i)$

Find aggregate index k that contains displacement node n^u

Find k with $\mathcal{A}_\ell^{(k)} \in \mathcal{A}_\ell^u$ where $n^u \in \mathcal{A}_\ell^{(k)}$

Loop over all Lagrange multipliers j

for $j \in D_\lambda$ **do**

Check whether Lagrange multiplier j is coupled with row i

if $D_{i,j} \neq 0$ **then**

Find pseudo node n^λ for Lagrange multiplier j

$n^\lambda \leftarrow n(j)$

Check whether to build a new Lagrange multiplier aggregate

if $d(k) = \emptyset$ **then**

Increment internal aggregation counter

$l \leftarrow l + 1$

Build a new aggregate and add Lagrange multiplier node n^λ

$\mathcal{A}_\ell^{(l)} \leftarrow \{n^\lambda\}$

Associate displacement aggregate k with Lagrange multiplier aggregate l

$d(k) \leftarrow \{l\}$

Add new aggregate to set of Lagrange multiplier aggregates \mathcal{A}_ℓ^λ

$\mathcal{A}_\ell^\lambda \leftarrow \mathcal{A}_\ell^\lambda \cup \mathcal{A}_\ell^{(l)}$

else

Extend aggregate $0 \leq d(k) \leq l$ with pseudo node

$\mathcal{A}_\ell^{(d(k))} \leftarrow \mathcal{A}_\ell^{(d(k))} \cup \{n^\lambda\}$

end

end

end

end

Return aggregates for Lagrange multipliers

return \mathcal{A}_ℓ^λ

The exact aggregation procedure is described in Algorithm 1. Assuming that the standard aggregates \mathcal{A}_ℓ^u for the displacement degrees of freedom are available, new aggregates \mathcal{A}_ℓ^λ are built by collecting the corresponding Lagrange multiplier degrees of freedom. Beside the displacement aggregates \mathcal{A}_ℓ^u , only the mortar matrix D is needed to algebraically reconstruct the contact interface and find the associated Lagrange multipliers. The new aggregates \mathcal{A}_ℓ^λ for the Lagrange multipliers can be interpreted as the natural extension of the displacement aggregates \mathcal{A}_ℓ^u at the interface. This facilitates to keep the ratio of coarse-level nodes at the slave contact interface and the coarse Lagrange multipliers constant, which also balances the ratio of contact constraints and inner structural displacement degrees of freedom over all multigrid levels.

The coarsening strategy outlined in Algorithm 1 as well as all other AMG components of the presented saddle-point preconditioner for contact problems have been implemented in MUELU,⁸² the next-generation multigrid package within the TRILINOS project.⁸³ For further details on the implementation, we refer to the MUELU user's guide⁸² and the MUELU website.⁸⁴

5 | BLOCK SMOOTHING METHODS FOR MORTAR CONTACT PROBLEMS

Using saddle point preserving aggregation and segregated transfer operators as outlined in Section 4 to generate a fully coupled AMG hierarchy (see Section 3.3), the coupling of structural equilibrium equations and contact constraints on all levels of a fully coupled AMG hierarchy is now addressed by block smoothing methods on each level. Schur complement based coupling iterations present themselves as ideal candidates to deal with the saddle point structure resulting from the constraint-like contact equations.

Nevertheless, not all classical Schur complement based block smoothers for saddle point systems behave the same way when applied to contact problems. Specifically, it is important that the block smoothers account for the contact constraints in (5). While each iteration's intermediate solution itself is not of great interest, its satisfaction of the contact constraints impacts the overall number of required iterations to reach convergence, since only an intermediate solution that satisfies the contact constraints can then also be accepted as the final solution of the iterative process. While an exact satisfaction of the contact constraints by the block smoothers is desirable, practical computations require a compromise between the accuracy of the block smoothers and their computational effort in order to obtain computationally competitive preconditioners.

In Section 5.1, we first revisit some classical smoothing methods for saddle point systems. Therefore, we assume exact inverses for the predictor and corrector step of each smoother to focus on the systematic error resulting from the specific block structure of the smoother. The assumption of exact inverses allows to assess the behavior of each block smoother via an error matrix, such that the impact of the block smoother on the contact constraints can be characterized. Afterwards, we introduce computationally cheaper variants with inexact block inverses in the predictor and corrector step as a compromise between accuracy and performance in Section 5.2, however preventing the discourse on error matrices. While the approximations due to the block structure of a specific smoother originate from the definition of the smoothing method, the quality of the approximate block inverses can fully be controlled by the user. Finally, Section 5.3 discusses their application in the context of saddle point systems for contact formulations.

5.1 | Block smoothers for saddle point problems

Now, we study the systematic errors introduced by specific block smoothers and their impact on the contact constraints. Therefore, we first assume exact mathematical block operations. To carefully distinguish between systematic errors and errors stemming from practical Schur complement approximations, we postpone the discussion of such approximations to Section 5.2.

The general block smoothing scheme can be written as

$$\begin{bmatrix} \Delta \mathbf{u}^{k+1} \\ \Delta \lambda^{k+1} \end{bmatrix} = \begin{bmatrix} \Delta \mathbf{u}^k \\ \Delta \lambda^k \end{bmatrix} + Q^{-1} \left(\begin{bmatrix} \mathbf{r}_u^k \\ \mathbf{r}_\lambda^k \end{bmatrix} - \begin{pmatrix} K & C_1^\top \\ C_2 & -Z \end{pmatrix} \begin{bmatrix} \Delta \mathbf{u}^k \\ \Delta \lambda^k \end{bmatrix} \right), \quad (8)$$

where Q describes the 2×2 block preconditioning matrix approximating the 2×2 block operator in (5). Typical block smoothers consist of an outer coupling iteration with nested subsolvers to build the inverses of the diagonal blocks of Q in an algorithmic predictor–corrector scheme. In the following, a few classical block smoothers from literature (e.g., Reference 85) are introduced, stating that this list is by far not complete. All of them can be interpreted as block extensions of classical iterative smoothing methods following the general block scheme (8), but depending on the definition of Q with a different effect on the contact constraints by introducing certain systematic errors. In general, the better Q approximates the block operator from (5), the lower the number of linear iterations will be when using the block smoother within a multigrid preconditioner.

5.1.1 | Uzawa smoother

For the (inexact) Uzawa smoother, one chooses

$$Q_{\text{UZ}} := \frac{1}{\alpha} \begin{pmatrix} K & 0 \\ C_2 & -\tilde{S} \end{pmatrix}. \quad (9)$$

The parameter $\alpha > 0$ is a damping parameter and \tilde{S} describes a cheap approximation of the Schur complement $S = Z + C_2 K^{-1} C_1^T$. For a theoretical review of Uzawa like smoothers, the reader is referred to References 86–88.

With the off-diagonal coupling block C_2 in (9), the smoother performs a one-way coupling in the sense that the Lagrange multiplier increments now depend on the current increment of the displacement degrees of freedom. Algorithm 2 represents the practical implementation as a predictor–corrector method. In each smoothing iteration, one calculates a prediction for the displacement increments $\delta \mathbf{u}^{k+1}$, which are taken into account when solving for the corresponding Lagrange multiplier increments $\delta \lambda^{k+1}$.

Algorithm 2. Uzawa smoother

Procedure Uzawa (α, k_{\max})

Apply k_{\max} smoothing sweeps with the Uzawa algorithm

for $k \leftarrow 0$ **to** $k_{\max} - 1$ **do**

Prediction step: solve for $\delta \mathbf{u}^{k+1}$

$$K \delta \mathbf{u}^{k+1} = \mathbf{r}_u^k - K \Delta \mathbf{u}^k - C_1^T \Delta \lambda^k$$

Correction step: solve for $\delta \lambda^{k+1}$

$$-\tilde{S} \delta \lambda^{k+1} = \mathbf{r}_\lambda^k - C_2 \Delta \mathbf{u}^k + Z \Delta \lambda^k - C_2 \delta \mathbf{u}^{k+1}$$

Update step: update solution variables

$$\Delta \mathbf{u}^{k+1} \leftarrow \Delta \mathbf{u}^k + \alpha \delta \mathbf{u}^{k+1}$$

$$\Delta \lambda^{k+1} \leftarrow \Delta \lambda^k + \alpha \delta \lambda^{k+1}$$

end

Return smooth solution vector

return $(\Delta \mathbf{u}^{k_{\max}}, \Delta \lambda^{k_{\max}})$

Assuming the smoothing iteration has converged, the error matrix for the Uzawa smoother is given as

$$E_{\text{UZ}} := A - Q_{\text{UZ}} = \begin{pmatrix} (1 - \frac{1}{\alpha})K & C_1^T \\ (1 - \frac{1}{\alpha})C_2 & -Z + \frac{1}{\alpha}\tilde{S} \end{pmatrix}. \quad (10)$$

With $\alpha = 1$, $\tilde{S} = Z + C_2 \tilde{K}^{-1} C_1^T$ being an approximation to the Schur complement, and \tilde{K} denoting an easy-to-invert approximation of K , for example, the diagonal of K as a cheap variant for the approximation \tilde{K} , that is, $\tilde{K} = \text{diag}(K)$, it is

easy to verify that the error matrix of the Uzawa smoother reduces to

$$E_{UZ} = \begin{pmatrix} 0 & C_1^T \\ 0 & C_2 \tilde{K}^{-1} C_1^T \end{pmatrix}. \quad (11)$$

That is, since the second block row in the error matrices (10) or (11) does not vanish, the Uzawa smoother by definition cannot exactly fulfill the contact constraints, but adds a systematic error for the contact constraints. This might have a negative impact on the overall performance of the iterative linear solver.

5.1.2 | Braess–Sarazin smoother

Originally introduced for the Stokes problem in Reference 80, the Braess–Sarazin smoother belongs to the class of block approximate smoothers and is based on the choice

$$Q_{BS} := \begin{pmatrix} \alpha \tilde{K} & C_1^T \\ C_2 & -Z \end{pmatrix} \quad (12)$$

for the block preconditioning matrix Q in (8). Again, the parameter $\alpha > 0$ denotes a scaling parameter and \tilde{K} refers to an easy-to-invert approximation of K .

Algorithm 3. Braess–Sarazin smoother

Procedure BraessSarazin(α, k_{\max})

Apply k_{\max} smoothing sweeps with Braess–Sarazin algorithm

for $k \leftarrow 0$ **to** $k_{\max} - 1$ **do**

Prediction step: determine prediction $\Delta \mathbf{u}^{k+\frac{1}{2}}$ by calculating

$$\Delta \mathbf{u}^{k+\frac{1}{2}} = \Delta \mathbf{u}^k + \frac{1}{\alpha} \tilde{K}^{-1} (\mathbf{r}_u^k - K \Delta \mathbf{u}^k - C_1^T \Delta \lambda^k)$$

Correction step: solve for $\delta \lambda^{k+\frac{1}{2}}$

$$-(Z + \frac{1}{\alpha} C_2 \tilde{K}^{-1} C_1^T) \sim \delta \lambda^{k+\frac{1}{2}} = \mathbf{r}_\lambda^k + Z \Delta \lambda^k - C_2 \Delta \mathbf{u}^{k+\frac{1}{2}}$$

Update step: update solution variables

$$\Delta \lambda^{k+1} \leftarrow \Delta \lambda^k + \delta \lambda^{k+\frac{1}{2}}$$

$$\Delta \mathbf{u}^{k+1} \leftarrow \Delta \mathbf{u}^{k+\frac{1}{2}} - \frac{1}{\alpha} \tilde{K}^{-1} C_1^T \sim \delta \lambda^{k+\frac{1}{2}}$$

end

Return smooth solution vector

return $(\Delta \mathbf{u}^{k_{\max}}, \Delta \lambda^{k_{\max}})$

Assuming convergence of the smoother, the error matrix for the Braess–Sarazin smoother is given as

$$E_{BS} := A - Q_{BS} = \begin{pmatrix} K - \alpha \tilde{K} & 0 \\ 0 & 0 \end{pmatrix}. \quad (13)$$

With the second block row in the blocked operator (5) being retained in (12), the Braess–Sarazin smoother seems to be a reasonable choice for dealing with contact constraints. The error matrix in (13) reveals that the quality of the block smoother only depends on the choice of \tilde{K} .

The implementation as a predictor–corrector method is based on the splitting of (12) into

$$\begin{pmatrix} \alpha\tilde{K} & C_1^\top \\ C_2 & -Z \end{pmatrix} = \begin{pmatrix} \alpha\tilde{K} & 0 \\ C_2 & -Z - \frac{1}{\alpha}C_2\tilde{K}^{-1}C_1^\top \end{pmatrix} \begin{pmatrix} I & \frac{1}{\alpha}\tilde{K}^{-1}C_1^\top \\ 0 & I \end{pmatrix}. \quad (14)$$

As one can easily see from Algorithm 3, the prediction step can be understood as one hard-coded sweep with a (damped) Jacobi iteration. In other words, the quality of the prediction for the displacement degrees of freedom must be considered rather poor. Exactly fulfilling contact constraints with respect to a rather poor prediction of the displacement variables might not be optimal for the overall performance of the preconditioner.

5.1.3 | SIMPLE variants

Originally introduced in References 75,89, the SIMPLE method is based on the approximate block factorization

$$Q_{\text{SIMPLE}} := \begin{pmatrix} K & 0 \\ C_2 & -\tilde{S} \end{pmatrix} \begin{pmatrix} I & \tilde{K}^{-1}C_1^\top \\ 0 & \frac{1}{\alpha}I \end{pmatrix} = \begin{pmatrix} K & K\tilde{K}^{-1}C_1^\top \\ C_2 & (1 - \frac{1}{\alpha})C_2\tilde{K}^{-1}C_1^\top - \frac{1}{\alpha}Z \end{pmatrix} \quad (15)$$

for the iterative method in (8). In (15), \tilde{S} denotes an approximation of the Schur complement $S := Z + C_2K^{-1}C_1^\top$ with a cheap and easy-to-invert approximation \tilde{K} of the block K . Algorithm 4 shows the implementation of the SIMPLE method using the predictor–corrector scheme (cf., Reference 90).

Algorithm 4. SIMPLE smoother

Procedure SIMPLE (α, k_{\max})

Apply k_{\max} smoothing sweeps with SIMPLE algorithm

for $k \leftarrow 0$ **to** $k_{\max} - 1$ **do**

Prediction step: solve for $\Delta\mathbf{u}^{k+\frac{1}{2}}$

$$K\Delta\mathbf{u}^{k+\frac{1}{2}} = \mathbf{r}_u^k - C_1^\top\Delta\lambda^k$$

Correction step: solve for $\delta\lambda^{k+\frac{1}{2}}$

$$-\tilde{S}\delta\lambda^{k+\frac{1}{2}} = \mathbf{r}_\lambda^k + Z\Delta\lambda^k - C_2\Delta\mathbf{u}^{k+\frac{1}{2}}$$

Update step: update solution variables

$$\Delta\lambda^{k+1} \leftarrow \Delta\lambda^k + \alpha \sim \delta\lambda^{k+\frac{1}{2}}$$

$$\Delta\mathbf{u}^{k+1} \leftarrow \Delta\mathbf{u}^{k+\frac{1}{2}} - \alpha\tilde{K}^{-1}C_1^\top \sim \delta\lambda^{k+\frac{1}{2}}$$

end

Return smooth solution vector

return ($\Delta\mathbf{u}^{k_{\max}}, \Delta\lambda^{k_{\max}}$)

For our applications, we found the diagonal matrix containing the row sums of $|K| = (|a_{ij}|)_{i,j=1,\dots,n_K}$ to be a good approximation for block K . This corresponds to the SIMPLEC method as introduced in Reference 91. That is, \tilde{K} is defined as the diagonal lumping of $|K|$ with

$$\tilde{K} = \text{diag} \left(\sum_{j=1}^{n_K} |a_{ij}| \right), \quad i = 1, \dots, n_K. \quad (16)$$

The default choice for \tilde{S} is consequently $\tilde{S} = \alpha Z + \alpha C_2\tilde{K}^{-1}C_1^\top$ with \tilde{K} as defined in (16). A more theoretical discussion on the mathematical consequences of approximations for the Schur complement S can be found in Reference 54.

For the SIMPLE preconditioner, the error matrix is calculated by

$$E_{\text{SIMPLE}} := A - Q_{\text{SIMPLE}} = \begin{pmatrix} 0 & C_1^T - K\tilde{K}^{-1}C_1^T \\ 0 & -Z - C_2\tilde{K}^{-1}C_1^T + \frac{1}{\alpha}\tilde{S} \end{pmatrix}. \quad (17)$$

As one can see from (17), SIMPLE perturbs the Lagrange multipliers, but it does not affect the terms that operate on the primary displacement variables since the first block column in (17) is zero. Choosing $\tilde{S} = \alpha Z + \alpha C_2\tilde{K}^{-1}C_1^T$, the error matrix reduces to

$$E_{\text{SIMPLE}} = \begin{pmatrix} 0 & C_1^T - K\tilde{K}^{-1}C_1^T \\ 0 & 0 \end{pmatrix}. \quad (18)$$

That is, an appropriate approximation \tilde{S} of the Schur complement S allows to exactly satisfy the contact constraints within one smoothing sweep. Depending on the choice for the approximation \tilde{K} , the SIMPLE method admits an error in the coupling between displacements and contact constraints. However, compared with the Braess–Sarazin method from Section 5.1.2, we put more focus on a good prediction for the displacements with a consistent update for fulfilling the contact constraints.

5.2 | Cheap variants of block smoothers

As one can easily see from the Algorithms 2–4, all block smoothing methods internally require inverses of the matrix blocks on the matrix diagonal and of the Schur complement operator \tilde{S} . Specifically, there is one linear system to be solved in the prediction step and one during the correction step. To keep the computational costs low in practical computations, one does not solve for the block inverses exactly, but applies a cheap approximation, for example, by using a fixed number of smoothing sweeps with a relaxation-based smoothing method such as symmetric Gauss–Seidel or an ILU sweep. This allows for more flexibility for finding a good compromise between quality and performance, as the user can decide how much effort should be put on finding a good prediction and fulfilling the Schur complement equation in the corrector step. While the systematic errors introduced by the choice of the block smoother from Section 5.1 are fixed, the practitioner has full control over the quality of the block inverses.

The numerical examples in Section 6 show that such an approximation leads to efficient and computationally reasonable block smoothing methods. As a naming convention, the prefix “cheap” is added to the name of the block smoothing method to indicate the usage of a cheap approximation for finding the inverse of the diagonal blocks in addition to the systematic approximations of building the Schur complement operator as discussed in Section 5.1.

The block smoothing methods and their cheap variants are available in the XPETRA package of the TRILINOS project.⁸³ For further details on the implementation, we refer to the MUELU user’s guide⁸² and the MUELU and XPETRA websites,^{84,92} respectively.

5.3 | Comparison of saddle point smoothing methods for contact problems

Considering the block scheme of (5), there are two main challenges for contact problems. First, we have the two distinct sets of equations as described in Section 2.3: the structural equations formulated in cartesian coordinates and the set of contact constraints formulated in normal-tangential coordinates relative to the contact surface. Second, the coupling of those two distinct sets of structural equations and contact constraint equations.

Algorithmically, the coupling of structural equations at the contact interface via the off-diagonal blocks in (5) is only considered within the block smoothers from Section 5.1. Therefore, constraint smoothers (cf., Reference 93) are the natural choice for contact problems, since the contact problem is implicitly governed by the contact constraint equations. Only solutions that are in alignment with the contact constraints are of interest.

Under certain preconditions as discussed in Section 5.1 and assuming sufficient smoother iterations to reach convergence of the smoother, both the Braess–Sarazin method and the SIMPLE-based methods would exactly fulfill the contact constraints as one can see from Equations (13) or (18). Therefore, with a systematic error in the lower-right block

of (11), the Uzawa method seems to be less promising for contact problems. In the Braess–Sarazin method, the approximation $\tilde{\mathbf{K}} = \text{diag}(\mathbf{K})$ is hard-coded with some scaling parameter $\alpha > 0$ and consistently used within the approximate Schur complement operator, which is defined by $\tilde{\mathbf{S}} = \mathbf{Z} + \frac{1}{\alpha} \mathbf{C}_2 \tilde{\mathbf{K}}^{-1} \mathbf{C}_1^T$.

In contrary to the Braess–Sarazin method, the SIMPLE based methods keep the full \mathbf{K} block whenever possible in the block factorization and use $\tilde{\mathbf{K}}$ only where its inverse is required. Consequently, in “cheap” variants of the SIMPLE method, more elaborate smoothing strategies can be used for the \mathbf{K} block instead of a hard-coded Jacobi sweep. Therefore, one can think of the SIMPLE methods to allow for a more balanced quality of approximations for the displacement degrees of freedom and Lagrange multipliers for the contact constraints, whereas the Braess–Sarazin method exhibits an imbalance in the sense that the computational effort spent for approximating the constraints is much higher than for dealing with the displacement variables.

6 | NUMERICAL EXAMPLES

For the numerical examples, we use our in-house code BACI⁹⁴ that internally uses various capabilities from the TRILINOS project.⁸³ The implementation of the multigrid algorithms is based on Trilinos’ MueLu package.^{82,95} In particular, all block smoothers from Section 5 as well as the contact specific aggregation strategy for the Lagrange multiplier unknowns as described in Section 4 are readily available in MueLu.

6.1 | Two solid bodies example

With the first example, we want to study the effect of the block smoothers from Section 5 for contact problems. Here, we not only compare different block smoothers, but also highlight the effect of varying the quality of the subsmoothing steps within the block smoother versus increasing the number of outer coupling iterations.

Motivated by findings in our previous work,⁴¹ this example briefly revisits a detail that has been problematic in the context of contact problems in condensed contact formulations. While the discrete global unknowns ($\mathbf{u}, \boldsymbol{\lambda}$) are—as usual—formulated with respect to the global Cartesian frame, the discrete Lagrange multiplier weights $\boldsymbol{\mu}$ and therefore the contact constraint equations in (3b) and (3c) are formulated with respect to a local convective coordinate system. This local system is defined at each slave node j by a surface normal vector and two tangent vectors, that is, by a triad of orthonormal basis vectors $(\mathbf{n})_j$, $(\boldsymbol{\tau}_\xi)_j$, and $(\boldsymbol{\tau}_\eta)_j$. Although it represents a quite intuitive and natural choice in contact mechanics, this local constraint formulation may lead to nondiagonally dominant system matrices and therefore poses a serious challenge to the development of iterative linear solvers as has been elaborated in Reference 41. In the following, we will also investigate the susceptibility of the proposed saddle point preconditioners to this phenomenon.

6.1.1 | Geometrical setup

Since we are interested in the solver behavior, by intention we choose a simple 3D contact example as shown in Figure 5. There are two solid bodies with the same material parameters using a Neo-Hookean material (density $\rho_0 = 0.1 \frac{\text{kg}}{\text{m}^3}$, Young’s modulus $E = 10 \text{ GPa}$, Poisson’s ratio $\nu = 0.3$). The initial gap between the two solid bodies is 0.02 m. The upper solid body (size: 0.8 m \times 0.8 m \times 0.5 m) is moving down with constant velocity along the normal to the contact interface toward the lower fixed solid body (size: 1.0 m \times 1.0 m \times 1.0 m).

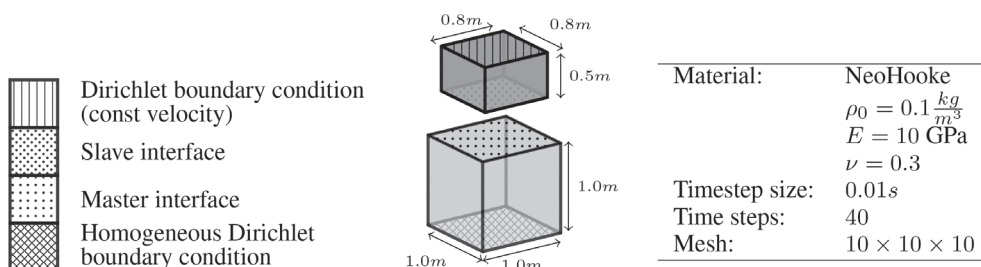


FIGURE 5 Two solid bodies example—Geometric configuration and parameters

6.1.2 | Experimental setup

To investigate a potential impact of the contact formulation in different coordinate systems (cartesian coordinates for the structural degrees of freedom and normal-tangential coordinates for the Lagrange multipliers) on the linear solver, we perform a similar experiment as introduced in Reference 41 and rotate the example setup around α_y and α_z as shown in Figure 6. We expect the number of linear iterations to be independent of the rotation angles α_y and α_z , since the underlying physics do not change. Any dependency of the linear solver on α_y and α_z would be a result of purely numerical effects and would turn out highly problematic for the iterative solution of large and complex contact problems. For reasons of symmetry, it is sufficient to vary α_y and α_z within $0 \leq \alpha_y, \alpha_z \leq \frac{\pi}{2}$.

6.1.3 | Discretization

The spatial discretization is based on a $10 \times 10 \times 10$ mesh for each solid block with altogether 6000 displacement degrees of freedom and 300 Lagrange multipliers modeling the contact coupling constraints for the 10×10 slave nodes at the contact interface (see Figure 5). The simulation runs for 40 time steps with a time step size of 0.01 s on four processors. After six time steps ($t=0.06$ s) both bodies come into contact and are deformed. We assume frictionless contact here. With this example we reduce the contact-specific effects (such as the contact search based on an active set strategy) to a minimum, such that one can focus on the linear solvers. That is, the contact zone is not changing once the two solid bodies are in contact.

6.1.4 | Stopping criteria

The nonlinear iteration inside each time step stops if either $\|\Delta \mathbf{u}\|_e < 10^{-8}$ holds for the Newton increment of the displacement degrees of freedom, or alternatively, if the conditions

$$\|\mathbf{r}_i^u\|_e < 10^{-6} \wedge \|\mathbf{r}_i^\lambda\|_e < 10^{-4} \quad (19)$$

hold for the nonlinear residuals \mathbf{r}_i^u and \mathbf{r}_i^λ in (4) after applying i Newton iterations. Thereby, $\|\bullet\|_e$ denotes the Euclidian vector norm. Those stopping criteria for the nonlinear solver are chosen to result in the same number of nonlinear iterations in each time step for an easier comparison of the linear solver behavior.

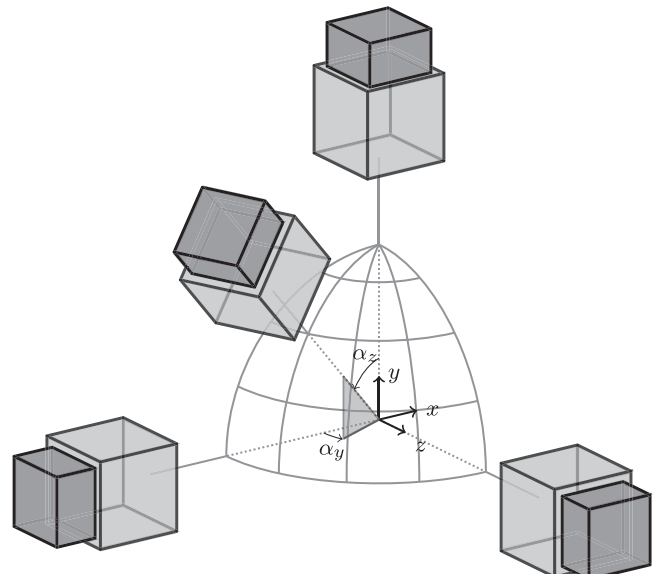


FIGURE 6 Two solid bodies example—Experimental setup to study independence of spatial orientation

Within each Newton iteration, the saddle point system (4) is solved iteratively using a preconditioned GMRES method with a three-level AMG preconditioner as described in Section 3. The iterative process for the linear system is considered to be converged, if it is

$$\left\| \frac{\mathbf{r}^k}{\mathbf{r}^0} \right\|_e < 10^{-8} \quad (20)$$

for the full residual vector $\mathbf{r}^k = \begin{bmatrix} \mathbf{r}^u \\ \mathbf{r}^\lambda \end{bmatrix}$ in the linear iteration step k . Here, the subscript i for the nonlinear Newton iteration is dropped.

In this work, we focus on the behavior of the linear solver. Therefore, a fixed stopping criterion for all tested variants is chosen in (20). This allows the comparison of different preconditioning techniques including their effect on the linear solution strategy. For real world problems, and especially for coupled multiphysics problems, the task of choosing appropriate stopping criteria for both the nonlinear and linear solver turns out to be quite challenging. Usually, one would choose a combination of different (length-scaled) norms for the partial vectors \mathbf{r}^u and \mathbf{r}^λ . In order to reduce the solver time in the inner linear solver, it is recommended to adapt the linear (relative) solver tolerance according to the residual norms of the outer nonlinear solver.

6.1.5 | Results

First, the effect of the different saddle point smoothers on the number of linear iterations is explored. Table 1 summarizes the average number of linear iterations per time step for different combinations of the rotation angles α_y and α_z . The numbers in brackets denote the maximum number of linear iterations needed for solving one linear system during the full simulation, roughly indicating the variation of the number of linear iterations within the simulation. For the CheapUzawa smoother, the number of iterations does not show a dependence on the rotation angles α_y and α_z . Comparing the numbers from Table 1(a) with the results for the CheapBraessSarazin smoother in Table 1(b), the CheapBraessSarazin smoother heavily suffers from the worse approximation of the displacement degrees of freedom using one internal hard-coded Jacobi sweep (cf., Section 5.1.2). The resulting iteration numbers show an obvious dependency on the rotation angles. With a CheapSIMPLEC block smoother, the number of iterations is lower than for the CheapUzawa smoother and independent from α_y and α_z when compared with the CheapBraessSarazin smoother (see Table 1(c)). So, the linear solver has some benefit from the two-way coupling of displacements and Lagrange multipliers within the AMG preconditioner. Compared with the Uzawa smoother, the additional computational costs for the CheapSIMPLEC method are very low with only one additional matrix-vector product by $\tilde{K}^{-1} \mathbf{C}_1^T$ per iteration. Therefore, CheapSIMPLEC is the preferred level smoother for our further experiments with some cheap approximations for the internal single fields using some sweeps with a (symmetric) Gauss-Seidel (SGS) method for the structural block or incomplete LU factorization (ILU) for the Lagrange multipliers.

Table 2 illustrates how the number of CheapSIMPLEC coupling iterations and the quality of the single-field smoothing methods within the CheapSIMPLEC smoother affect the number of linear iterations. Improving the quality of the Schur complement approximations within CheapSIMPLEC (see Table 2(a) vs. (b)) as well as increasing the number of CheapSIMPLEC coupling iterations (see Table 2(c)) unsurprisingly reduces the number of linear solver iterations. Aside from the concrete parameter choices for the level smoother, one can even further reduce the number of linear iterations with a reasonable transfer operator smoothing strategy for the displacement block, for example, as indicated in (6).

By intention, we do not report solver timings, since this example is too small to perform reasonable time measurements, especially when using four processors for altogether only 6300 degrees of freedom.

The intention of this example is to compare typical saddle point smoothers within a fully coupled AMG preconditioner. One can observe the expected behavior that increasing the number of smoothing sweeps reduces the number of linear GMRES iterations. However, in practice, the variant with a smaller number of GMRES iterations may not always be the fastest method. This example shows that the proper choice of block-level smoothing is essential for the overall performance of a saddle point multigrid method. The particular choice of the block smoothing method gives the user full control over the quality of the coupling with field-specific parameters and allows for fine-grained adaptations and problem-specific optimizations.

With the experience from this example one can choose efficient level smoothers which provide results independent from the exact geometric configuration.

TABLE 1 Two solid bodies example—Average (maximum) number of linear GMRES iterations per nonlinear iteration (over all 40 time steps) for different combinations of rotation angles α_y and α_z

(a) Level smoother: three CheapUzawa(0.7) with one SGS (0.7) + ILU(0)					
	α_y				
	0	$\frac{1}{8}\pi$	$\frac{1}{4}\pi$	$\frac{3}{8}\pi$	$\frac{1}{2}\pi$
α_z					
0	26.2 (30)	26.0 (30)	24.6 (29)	24.6 (30)	26.0 (29)
$\frac{1}{8}\pi$	26.8 (30)	25.8 (37)	25.3 (30)	24.7 (30)	27.3 (31)
$\frac{1}{4}\pi$	25.9 (31)	25.1 (32)	25.3 (30)	28.7 (40)	26.0 (31)
$\frac{3}{8}\pi$	25.9 (31)	25.0 (30)	24.9 (30)	25.5 (33)	25.3 (29)
$\frac{1}{2}\pi$	26.1 (30)	25.4 (32)	25.5 (30)	26.8 (31)	26.0 (30)
(b) Level smoother: three CheapBraessSarazin(1.9) with ILU(0)					
	α_y				
	0	$\frac{1}{8}\pi$	$\frac{1}{4}\pi$	$\frac{3}{8}\pi$	$\frac{1}{2}\pi$
α_z					
0	29.9 (37)	29.6 (36)	27.9 (33)	30.7 (38)	29.7 (37)
$\frac{1}{8}\pi$	41.2 (59)	43.1 (68)	42.0 (58)	43.6 (63)	41.4 (59)
$\frac{1}{4}\pi$	56.8 (82)	64.8 (86)	68.6 (95)	64.3 (86)	54.9 (75)
$\frac{3}{8}\pi$	40.9 (60)	55.5 (74)	73.1 (102)	111.1 (141)	41.3 (61)
$\frac{1}{2}\pi$	29.9 (37)	41.0 (55)	56.8 (79)	41.3 (61)	29.7 (36)
(c) Level smoother: three CheapSIMPLEC(0.7) with one SGS (0.7) + ILU(0)					
	α_y				
	0	$\frac{1}{8}\pi$	$\frac{1}{4}\pi$	$\frac{3}{8}\pi$	$\frac{1}{2}\pi$
α_z					
0	20.5 (26)	19.3 (21)	18.8 (23)	19.3 (26)	19.3 (20)
$\frac{1}{8}\pi$	20.1 (22)	19.7 (27)	20.0 (24)	19.8 (21)	21.2 (25)
$\frac{1}{4}\pi$	20.1 (25)	19.9 (23)	20.1 (23)	22.0 (26)	20.8 (30)
$\frac{3}{8}\pi$	19.7 (22)	19.8 (22)	19.8 (23)	20.0 (26)	19.7 (23)
$\frac{1}{2}\pi$	19.3 (20)	19.6 (22)	20.4 (27)	20.7 (25)	20.5 (28)

Note: As preconditioner, a three-level AMG method (PA-AMG + PA-AMG, minimum aggregate size: six nodes) is used with different level smoothers. Within the block smoothers on all multigrid levels, symmetric Gauss–Seidel is used for the structural degrees of freedoms and ILU for the Lagrange multipliers.

6.2 | Weak scaling behavior

To assess the behavior of the proposed multilevel preconditioner applied to large-scale examples, we now report a weak scaling study. To exclude side effects (such as changes in the contact active set) and to fully focus on the behavior of the preconditioner and iterative linear solver, we study a simplified and linear contact problem.

6.2.1 | Setup

We consider a small block (dimensions 0.8 m \times 0.8 m \times 0.4 m) and a slightly bigger block (dimensions 1.0 m \times 1.0 m \times 0.5 m), where contact will occur between the large faces of both blocks. To reduce the complexity of the contact problem and to exclude nonlinearities due to changes in the contact active set, the faces opposite to the contact interface are fixed with Dirichlet boundary conditions, while the blocks initially penetrate each other at the contact

TABLE 2 Two solid bodies example—Average (maximum) number of linear GMRES iterations per nonlinear iteration (over all 40 time steps) for different rotation angles α_y and α_z

(a) Level smoother: one CheapSIMPLEC(0.7) with one SGS (0.7) + ILU(0)					
	α_y				
	0	$\frac{1}{8}\pi$	$\frac{1}{4}\pi$	$\frac{3}{8}\pi$	$\frac{1}{2}\pi$
α_z					
0	37.5 (57)	35.6 (43)	35.2 (46)	36.1 (44)	35.7 (45)
$\frac{1}{8}\pi$	36.4 (45)	37.6 (44)	38.5 (47)	36.3 (43)	38.1 (53)
$\frac{1}{4}\pi$	39.1 (50)	37.8 (47)	37.0 (47)	39.2 (48)	37.0 (44)
$\frac{3}{8}\pi$	36.3 (43)	37.0 (48)	37.8 (46)	37.2 (51)	36.4 (43)
$\frac{1}{2}\pi$	36.2 (47)	38.0 (51)	39.8 (50)	36.9 (50)	36.6 (53)

(b) Level smoother: one CheapSIMPLEC(0.7) with three SGS (0.7) + ILU(0)					
	α_y				
	0	$\frac{1}{8}\pi$	$\frac{1}{4}\pi$	$\frac{3}{8}\pi$	$\frac{1}{2}\pi$
α_z					
0	29.5 (41)	27.5 (34)	28.4 (36)	28.5 (37)	27.1 (36)
$\frac{1}{8}\pi$	29.3 (37)	30.1 (40)	32.7 (40)	28.8 (35)	28.4 (38)
$\frac{1}{4}\pi$	30.0 (35)	29.6 (37)	28.5 (37)	31.1 (38)	29.2 (36)
$\frac{3}{8}\pi$	28.8 (36)	27.8 (34)	28.1 (34)	29.1 (39)	28.1 (34)
$\frac{1}{2}\pi$	28.5 (38)	28.4 (34)	29.7 (40)	27.9 (35)	27.1 (35)

(c) Level smoother: three CheapSIMPLEC(0.7) with three SGS (0.7) + ILU(0)					
	α_y				
	0	$\frac{1}{8}\pi$	$\frac{1}{4}\pi$	$\frac{3}{8}\pi$	$\frac{1}{2}\pi$
α_z					
0	15.5 (16)	17.0 (19)	16.9 (20)	15.3 (16)	15.2 (16)
$\frac{1}{8}\pi$	16.2 (17)	16.1 (17)	16.3 (17)	15.9 (17)	16.1 (17)
$\frac{1}{4}\pi$	16.0 (17)	16.0 (17)	16.0 (17)	15.6 (18)	15.6 (16)
$\frac{3}{8}\pi$	15.7 (17)	15.9 (17)	15.9 (17)	15.5 (19)	15.6 (16)
$\frac{1}{2}\pi$	15.6 (16)	15.8 (16)	15.7 (16)	15.4 (16)	15.1 (16)

Note: As preconditioner, a three-level AMG method (minimum aggregate size: six nodes) is used with different variants of CheapSIMPLEC. Within the CheapSIMPLEC method on all multigrid levels, symmetric Gauss–Seidel sweeps are used for the structural degrees of freedoms and ILU for the Lagrange multipliers.

interface by 0.001. The smaller block acts as the slave side and its entire contact area is initialized as “active.” Application of the contact algorithms will then result in a slight compression of both blocks, such that the initial penetration vanishes. This problem setup allows to distill the performance of the AMG preconditioner under uniform mesh refinement and weak scaling conditions.

Both blocks use a Neo-Hooke material with Young’s modulus $E = 10$ MPa and Poisson’s ratio $\nu = 0.3$. Denoting the mesh refinement factor with κ , both blocks are discretized with 2κ linear hexahedral elements along their longer edges and κ elements along the shorter edges. The Lagrange multiplier field is discretized with standard shape functions, that is, linear Lagrange polynomials.

6.2.2 | Solver and preconditioner settings

As linear solver, a preconditioned GMRES method is used with a fully coupled multigrid approach as described in Section 3.2. The convergence criterion for the linear GMRES solver is set to

$$\left\| \frac{\mathbf{r}^k}{\mathbf{r}^0} \right\|_e < 10^{-8} \quad (21)$$

for the full residual vector $\mathbf{r}^k = \begin{bmatrix} \mathbf{r}^u \\ \mathbf{r}^\lambda \end{bmatrix}$ in the linear iteration step k . For the segregated block transfer operator as introduced in (7), we combine smoothed aggregation (SA-AMG) with a prolongator smoothing factor $\omega = 4/3$ for the displacement aggregates and plain nonsmoothed (PA-AMG) transfer operators for the Lagrange multiplier aggregates. The restriction operators are built as the transposed of the prolongation operators. Coarsening stops when the total number of rows in the saddle-point system drops below 5000.

Following the guidance from Section 5.3 and the findings from Section 6.1.5, we apply three sweeps of CheapSIMPLE(0.8) as a level smoother, where both the predictor and corrector step are approximated by one sweep of SGS each. We use the same level smoother layout on all multigrid levels except of the coarsest level. For the coarsest level, we compare two variants: a direct solver (marked as “LU,” requiring an expensive merging of the block matrix into a regular sparse matrix format and a subsequent LU factorization) versus the CheapSIMPLE level smoother as on all other multigrid levels (marked as “CheapSIMPLE”). We will study the impact of the coarse solver on scaling behavior, iteration counts, as well as AMG setup and V-cycle timings.

6.2.3 | Results

For the weak scaling study, we target the load per rank to be 50k displacement DOFs. A uniform mesh refinement as outlined in Table 3 is performed with the finest mesh consisting of roughly 23.9 million unknowns in the saddle-point system.

Therein, n^{proc} denotes the number of MPI ranks, κ the mesh refinement factor introduced in Section 6.2.1, n_{DOF}^u , n_{DOF}^λ , and $n_{\text{DOF}}^{\text{total}}$ the overall number of displacement, Lagrange multiplier, and total number of unknowns, $n_{\text{DOF}}^u/\text{proc}$ the average load per MPI rank. The mesh refinement factor κ is chosen such that the targeted average number of displacement unknowns per MPI rank is met. The number of levels n^ℓ , the actual size of the coarse-level system $n_{\text{DOF}}^{\text{total}(n^\ell)}$ as well as the operator complexity C_A (defined as the ratio of nonzero entries of all multigrid-level matrices A_ℓ with $\ell = 0, \dots, n^\ell - 1$ and the number of nonzeros on the finest level) are reported in the last three columns of Table 3. The operator complexity C_A slightly increases with larger problem sizes, but does not grow beyond $C_A \approx 1.30$.

TABLE 3 Weak scaling study—Mesh refinement and hierarchy details

n^{proc}	κ	n_{DOF}^u	n_{DOF}^λ	$n_{\text{DOF}}^{\text{total}}$	$n_{\text{DOF}}^u/\text{proc}$	n^ℓ	$n_{\text{DOF}}^{\text{total}(n^\ell)}$	C_A
4	20	211,806	5043	216,849	52,951.5	3	948	1.16
8	25	405,756	7803	413,559	50,719.5	3	1275	1.18
16	32	836,550	12,675	849,225	52,284.4	3	2661	1.23
24	36	1,183,038	15,987	1,199,025	49,293.2	3	3426	1.24
48	46	2,439,018	25,947	2,464,965	50,812.9	4	393	1.26
72	52	3,505,950	33,075	3,539,025	48,693.8	4	570	1.27
96	58	4,845,906	41,067	4,886,973	50,478.2	4	762	1.28
144	66	7,110,978	53,067	7,164,045	49,381.8	4	1071	1.28
192	73	9,594,396	64,827	9,659,223	49,970.8	4	1401	1.29
240	79	12,134,880	75,843	12,210,723	50,562.0	4	1713	1.29
288	84	14,566,110	85,683	14,651,793	50,576.8	4	2103	1.30
336	88	16,729,686	93,987	16,823,673	49,790.7	4	2421	1.30
384	92	19,097,550	102,675	19,200,225	49,733.2	4	2709	1.30
432	96	21,678,918	111,747	21,790,665	50,182.7	4	3090	1.30
480	99	23,760,600	118,803	23,879,403	49,501.2	4	3884	1.30

The scaling study is run on our in-house cluster (20 nodes with 2x Intel Xeon Gold 5118 (Skylake-SP) 12 core CPUs, 480 cores in total, Mellanox Infiniband Interconnect). Solver performance in terms of the number of GMRES iterations as well as wall clock time spent in the AMG preconditioner construction and its application (i.e., sweeping through the V-cycle once per GMRES iteration) are summarized in Figure 7.

With respect to iteration counts, both the direct solver as well as the block smoother yield the GMRES iteration count to be fairly independent of the overall mesh size, especially when taking the slight deviations of the load per core as detailed in Table 3 into account.

Using a direct solver on the coarsest multigrid level perfectly deals with all left-over error modes that are not properly handled by the block smoothers on the finer multigrid levels. Therefore, the “LU” variant results in a lower iteration count and is less sensitive to the actual size of the coarse-level system. In contrast, when using the CheapSIMPLE block smoother as coarse-level solver, we observe a somewhat higher iteration count together with an obvious impact of the number of multigrid levels on the iteration count (e.g., when moving from a three-level hierarchy on 24 MPI ranks to a four-level hierarchy on 48 MPI ranks). Increasing the number of multigrid levels helps to compensate the lack of coupling of the CheapSIMPLE block smoother compared with a direct solve on the coarsest level.

We show the CheapSIMPLE variants for the coarse solve since this variant in practice might be superior with respect to the overall performance: Regarding AMG setup time, the cost for the direct coarse solver becomes more expensive with a growing coarse system size, since (i) more matrix entries need to be moved from a block sparse matrix layout to a regular sparse matrix layout and (ii) a larger matrix needs to be factorized. For the CheapSIMPLE block smoother as coarse-level solver, no additional communication is required on the coarsest level. Hence, the mild slope in the graph for preconditioner setup time for the CheapSIMPLE coarse-level solver originates only from the Galerkin product during hierarchy construction. In terms of preconditioner application, that is, time spent in the AMG V-cycle, both methods are almost identical and scale very well.

While proper weak scalability up to 23.9 million unknowns on 480 MPI ranks has been demonstrated, the actual choice of the coarse solver in practical applications also depends on other factors such as the frequency of rebuilding the AMG hierarchy or the balance of setup time, V-cycle time, and cost of the additional GMRES iterations, which all together impact the overall time to solution. In the next example, we will put attention on effects for the linear solver caused by changes in the active set of contact nodes for larger problems.

6.3 | 1000 deformable rings

Even though only a 2D example of moderate size, the 1000 rings example comes with frequent changes in the contact active set and, thus, tests the robustness of the proposed multigrid methods. Particularly, we are interested in the comparison of the fully coupled multigrid approach and the nested multigrid approach as described in Section 3.2.

6.3.1 | Setup

This example consists of 1000 deformable rings (Neo-Hookean material with $E = 210$ GPa, $\nu = 0.3$, and $\rho_0 = 7.83 \cdot 10^{-6}$ kg/m³) arranged in a rectangle (see Figure 8). A gravitational force is inducing an acceleration toward a rigid wall. The simulated time extends to 2.0 s with a time step size of $\Delta t = 0.0005$ s, yielding 4000 time steps in total. The full mesh consists 110,000 nodes with 110 nodes for each deformable ring.

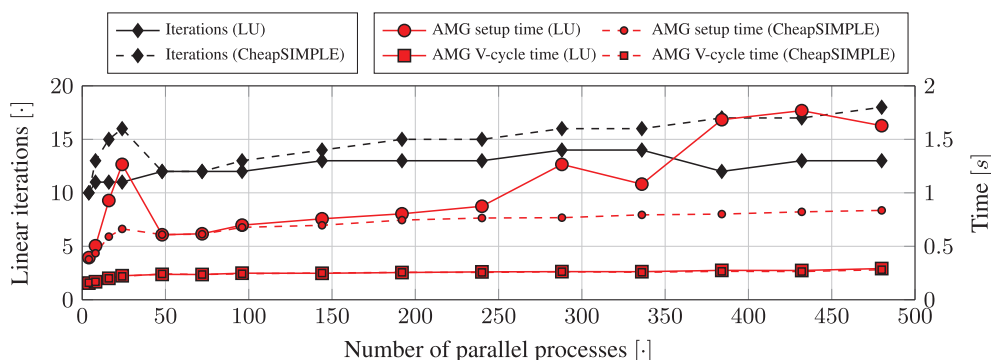


FIGURE 7 Weak scaling study—Linear solver iterations and AMG setup and V-cycle timings demonstrate proper weak scaling behavior. The type of coarse solver is indicated in parentheses

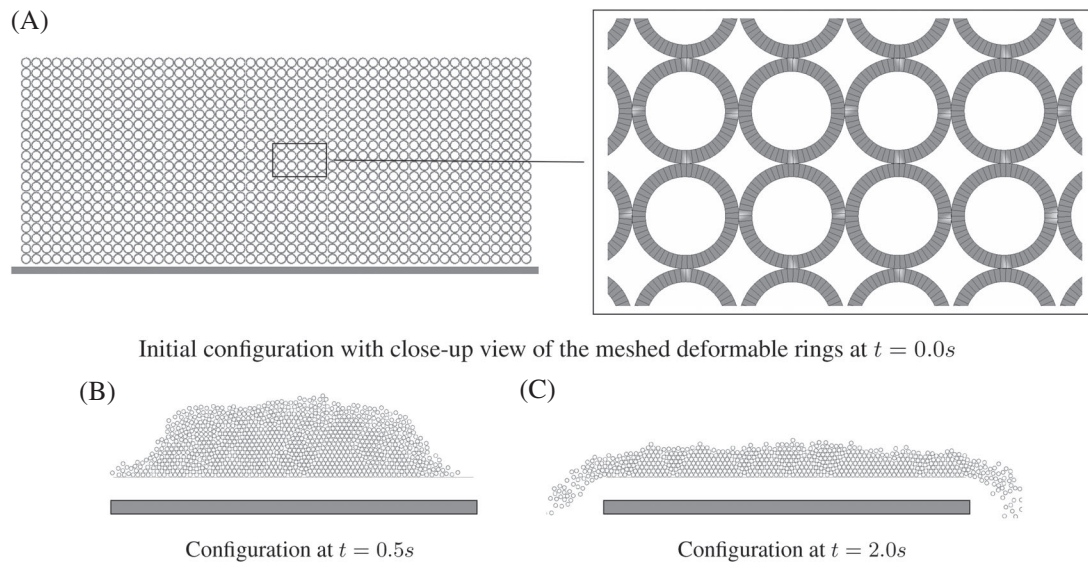


FIGURE 8 1000 deformable rings—Characteristic stages at different times

6.3.2 | Stopping criteria

In each time step, the nonlinear system is handled by a semismooth Newton method. As convergence criteria, we have chosen

$$\|\Delta \mathbf{u}\|_e < 10^{-8} \wedge (\|\mathbf{r}_i^{\mathbf{u}}\|_e < 10^{-8} \wedge \|\mathbf{r}_i^{\lambda}\|_e < 10^{-6}). \quad (22)$$

Here, $\mathbf{r}_i^{\mathbf{u}}$ and \mathbf{r}_i^{λ} denote the nonlinear residual for the displacement and Lagrange multiplier variables after i Newton iterations, respectively. Similarly, $\Delta \mathbf{u}$ denotes the solution increment for the displacement variables in the i th Newton iteration.

For solving the linear systems arising during the simulation a GMRES solver is applied with different variants of AMG preconditioners listed in Table 4. The relative tolerance of convergence for the GMRES solver is set to $\left\| \frac{\mathbf{r}^k}{\mathbf{r}^0} \right\|_e < 10^{-8}$ with $\mathbf{r}^k = \begin{bmatrix} \mathbf{r}^{\mathbf{u}} \\ \mathbf{r}^{\lambda} \end{bmatrix}$ being the full residual vector in the linear iteration step k . Here, the subscript i for the nonlinear Newton step is dropped.

The stopping criteria (22) for the nonlinear solver are carefully chosen in such a way that the simulations with all the tested preconditioner variants shown in Table 4 always result in the same number of nonlinear iterations. This way we can directly compare the number of linear iterations of all tested solver variants which allow to draw some conclusions on the multigrid preconditioners.

6.3.3 | Results

Table 4 provides an overview of the chosen preconditioner parameters for the level smoothers. For both the nested multigrid schemes and the fully coupled multigrid schemes we apply the level smoother on all multigrid levels including the coarsest level. For each class of multigrid preconditioners, only those variants are presented, that give the best overall timings and are able to accomplish the all 4000 time steps of the full simulation.

The multigrid parameters are chosen to be the same for all preconditioner variants: the minimum size of the aggregates is set to six nodes for the two-dimensional problem and the maximum coarse-level size is set to 1000 degrees of freedom, yielding a three-level multigrid method.

For the fully coupled AMG variants, different transfer operator strategies are compared, namely the nonsmoothed (PA-AMG) transfer operators and the energy minimization approach with local damping parameters for transfer operator

TABLE 4 1000 deformable rings—Different AMG variants

Preconditioner type			
Full multigrid based methods		Nested multigrid based methods	
PA-AMG (CheapSIMPLE)		CheapSIMPLE (PA-AMG)	
Transfer operators:	PA-AMG	Block prec.:	One CheapSIMPLEC (0.8)
Level smoother:	One CheapSIMPLEC (0.8)	– Pred. smoother:	AMG
– Pred. smoother:	Three SGS (0.8)	– Transfer op.:	PA-AMG
– Corr. smoother:	ILU (0)	– Level sm.:	One SGS (0.8)
		– Corr. smoother:	KLU
Emin (CheapSIMPLE)		CheapSIMPLE (SA-AMG)	
Transfer operators:	Emin	Block prec.:	One CheapSIMPLEC (0.8)
Level smoother:	One CheapSIMPLEC (0.8)	– Pred. smoother:	AMG
– Pred. smoother:	Three SGS (0.8)	– Transfer op.:	SA-AMG (0.8)
– Corr. smoother:	ILU (0)	– Level sm.:	One SGS (0.8)
		– Corr. smoother:	KLU

smoothing denoted by Emin, cf., Reference 63. For the nested AMG variants, we compare transfer operators based on PA-AMG and SA-AMG. All the simulations have been run on 16 cores (spread over two Intel Xeon E5-2670 Octocore CPUs).

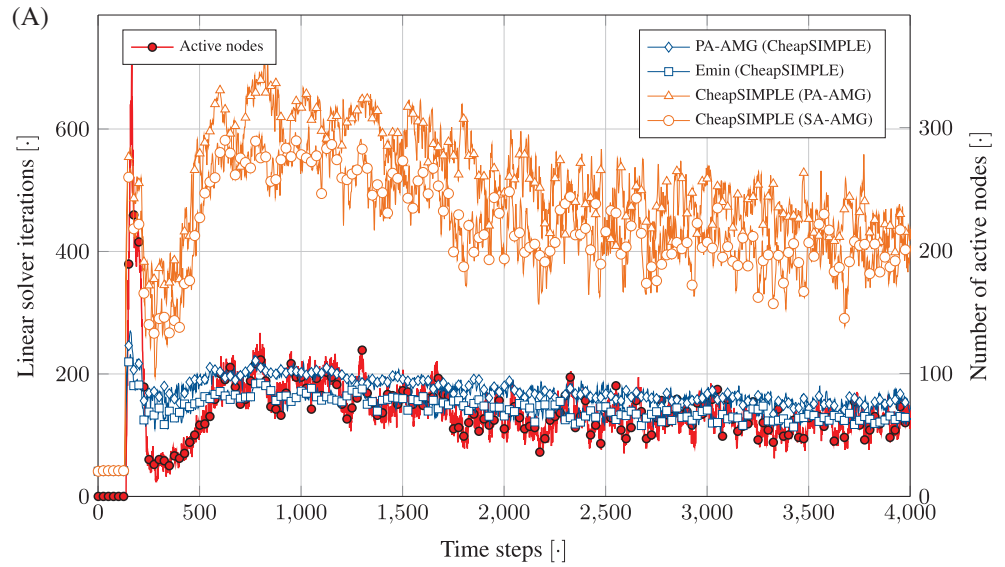
Figure 9(A) shows for each time step the overall number of linear iterations for all nonlinear iterations stacked onto each other. Compared with the CheapSIMPLE based methods, the fully coupled AMG variants need a significantly lower number of linear iterations to solve the problem in each time step. This can be explained by the better approximation of the displacement degrees of freedom using three instead of one damped Gauss–Seidel sweeps. One can also see how the number of linear iterations correlates with the number of active contact nodes. Obviously, the fully coupled AMG variants are less sensitive to the changes in the number of active contact nodes than the CheapSIMPLE based methods. In Figure 9(B), the overall linear solver time is given for each time step. Again, the fully coupled AMG variants perform better, although the difference is less pronounced due to the higher computational effort of fully coupled AMG implementations. The number of nonlinear iterations per time step is 2 for the initial phase without contact and then varies between 4 and 6, when contact is active. It is the same for all preconditioner variants due to the particular choice of stopping criteria of the nonlinear solver (see Section 6.3.2). Moreover, the number of nonlinear iterations is independent of the size of the active set.

Finally, overall timings for the linear solver including preconditioner setup are reported in Table 5. The overall solver time only varies slightly for all given variants. The variants Emin (CheapSIMPLE) and CheapSIMPLE (SA-AMG) need more setup time due to the additional effort of prolongator smoothing. Emin (CheapSIMPLE) can fully amortize the additional setup cost during the solve phase due to coarse grid corrections, that respect the contact constraints, while CheapSIMPLE (SA-AMG) does not have this benefit and, thus, requires the largest overall solver time. To show a fair comparison, the preconditioner has been rebuilt in every nonlinear iteration step. Of course, reuse strategies might increase amortization of setup costs and reduce overall solver timings when solving actual problems.

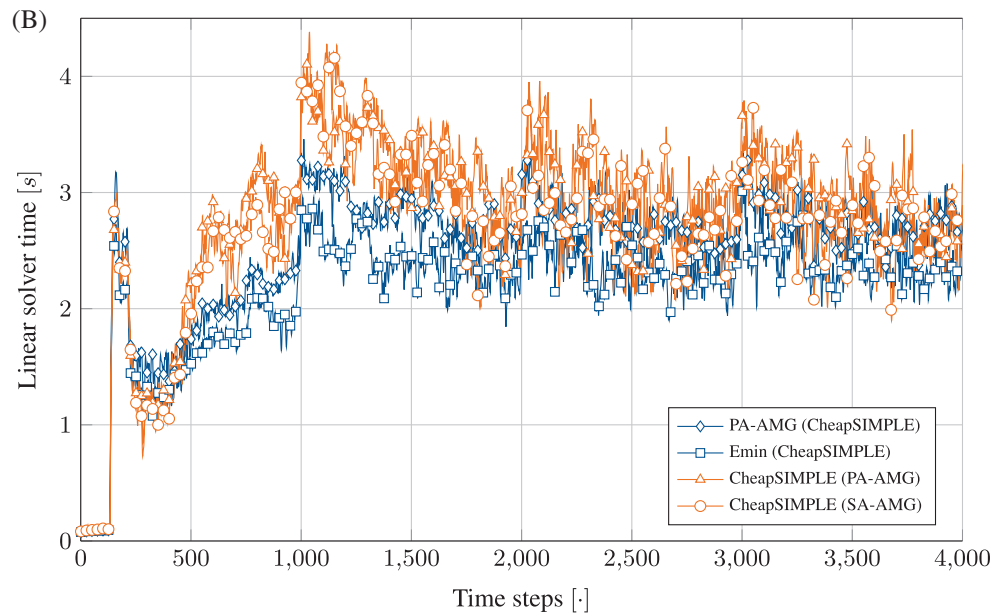
6.4 | Two tori impact example

Inspired by a similar analysis in Reference 96, with the two tori impact example we test the proposed multigrid variants from Section 3.2 on a complex 3D contact example with more than 1 million unknowns. Please refer to Reference 97 for the detailed problem setup of the two tori impact example with geometry and load conditions.

FIGURE 9 1000 collapsing rings example—Results for different AMG preconditioner variants



Number of linear GMRES iterations for all nonlinear iterations over time steps for different preconditioner variants. The red curve shows the number of active nodes for each time step.



Overall linear solver time over time steps.

TABLE 5 1000 deformable rings—Exemplary timings in (s) of the different preconditioning variants from Table 4 for the full simulation (4000 time steps)

Method	Setup costs	Solver time	Overall solver time
PA-AMG (CheapSIMPLE)	11,870	10,013	21,883
Emin (CheapSIMPLE)	12,820	8679	21,499
CheapSIMPLE (PA-AMG)	11,730	11,103	22,833
CheapSIMPLE (SA-AMG)	12,300	10,763	23,063

6.4.1 | Setup

The example consists of two thin-walled tori with a Neo-Hookean material model ($E = 2250$ GPa, $\nu = 0.3$, $\rho_0 = 0.1$ kg/m³) with a major and minor radius of 76 and 24 m and a wall thickness of 4.5 m. The left torus in Figure 10 lies in the xy -plane with resting initial conditions. The right torus has been rotated around the y -axis by 45° and has an initial velocity of 1.0 m/s directed toward the left torus. The simulated time is 10 s divided into 200 time steps with a time step size of 0.05 s using a generalized- α time integration scheme.⁹⁸ The mesh consists of 284,672 first-order hexahedral elements with 350,208 nodes.

With the rather complex geometry and contact configuration, that heavily and frequently changes over time, this example can be considered as a representative test for the robustness and efficiency of the tested numerical methods.

6.4.2 | Stopping criteria

The stopping criteria for the semismooth Newton method are chosen as

$$\|\Delta \mathbf{u}\|_e < 10^{-7} \wedge \left(\left\| \frac{\mathbf{r}_i^u}{\mathbf{r}_0^u} \right\|_e < 10^{-8} \wedge \left\| \mathbf{r}_i^\lambda \right\|_e < 10^{-4} \right). \quad (23)$$

Here, \mathbf{r}_i^u and \mathbf{r}_i^λ denote the (nonlinear) residual for the displacement variables and Lagrange multipliers in the i th Newton iteration, respectively. The quantity $\Delta \mathbf{u}$ describes the solution increment for the displacement variables only. Again, the stopping criteria in (23) for the nonlinear solver are specifically chosen to produce the same number of nonlinear iterations for all tested solver variants. This allows for an easy comparison of the number of linear iterations during the simulation.

As linear solver a preconditioned GMRES method is used with the convergence criterion

$$\left\| \frac{\mathbf{r}^k}{\mathbf{r}^0} \right\|_e < 10^{-8} \quad (24)$$

for the full residual vector $\mathbf{r}^k = \begin{bmatrix} \mathbf{r}^u \\ \mathbf{r}^\lambda \end{bmatrix}$ in the linear iteration step k .

6.4.3 | Results

An overview of the different tested preconditioner variants is given in Table 6. We study variants with the fully coupled multigrid approach, the nested multigrid approach, and a SIMPLE based variant without multigrid at all. For the fully coupled AMG variants, the transfer operators for the displacement blocks are varied. Particularly, nonsmoothed transfer operators (PA-AMG) are compared with smoothed aggregation transfer operators (SA-AMG). For the multigrid schemes there is no special treatment of the coarsest level. For the fully coupled multigrid schemes as well as for the nested multigrid method we apply the same level smoother on all multigrid levels including the coarsest level. All the simulations have been run on 16 cores (spread over two Intel Xeon E5-2670 Octocore CPUs).

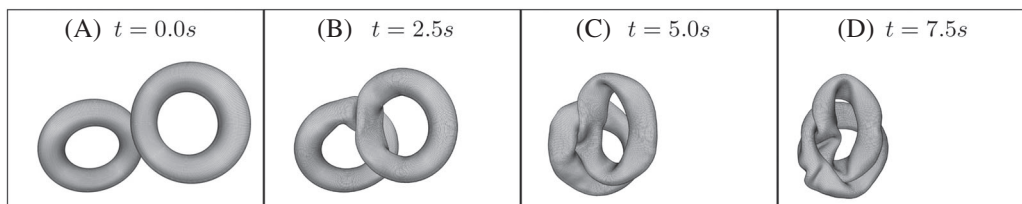
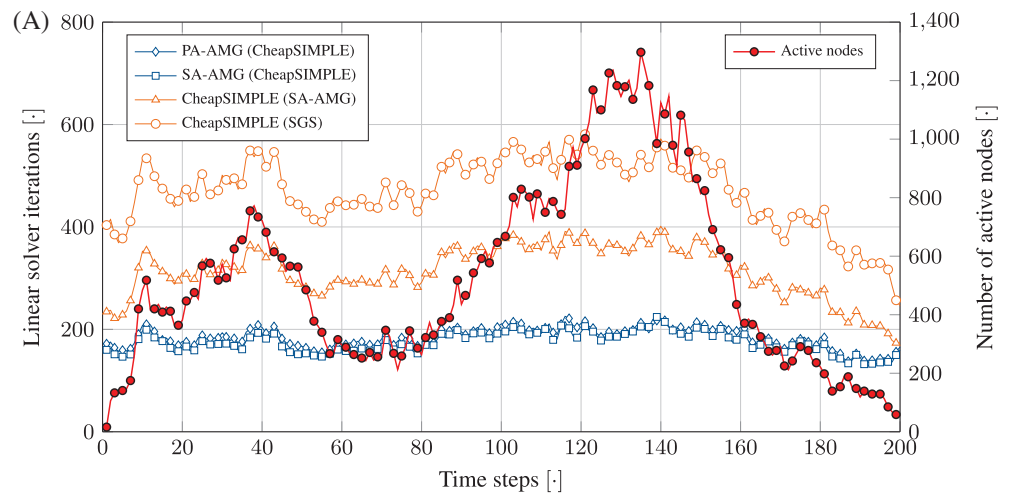


FIGURE 10 Two tori impact example—Characteristic stages of deformation

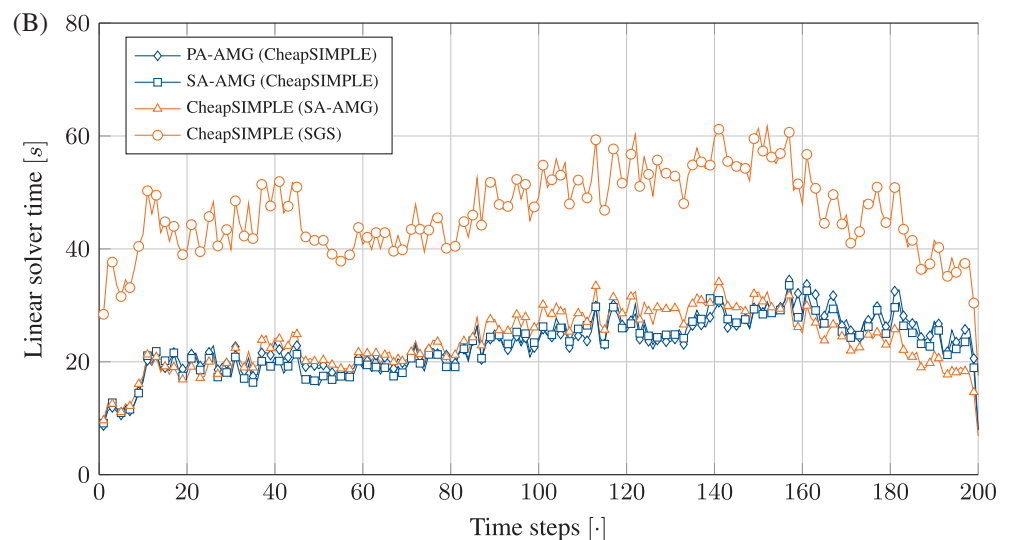
TABLE 6 Two tori impact example—Different AMG variants

Full multigrid based methods		SIMPLE based methods	
PA-AMG (CheapSIMPLE)		CheapSIMPLE (SGS)	
Transfer operators:	PA-AMG	Transfer operators:	—
Level smoother:	One CheapSIMPLEC (0.8)	Block prec.:	Two CheapSIMPLEC (0.8)
– Pred. smoother:	One SGS (0.8)	– Pred. smoother:	Three SGS (0.8)
– Corr. smoother:	ILU (0)	– Corr. smoother:	ILU (0)
SA-AMG (CheapSIMPLE)		CheapSIMPLE (SA-AMG)	
Transfer operators:	SA-AMG (0.4)	Block prec.:	Two CheapSIMPLEC (0.8)
Level smoother:	One CheapSIMPLEC (0.8)	– Pred. smoother:	AMG
– Pred. smoother:	One SGS (0.8)	– Transfer op.:	SA-AMG (0.4)
– Corr. smoother:	ILU (0)	– Level sm.:	Two SGS (0.8)
		– Corr. smoother:	ILU (0)

FIGURE 11 Two tori impact example—Results for different saddle point preconditioner variants



Number of linear GMRES iterations for all nonlinear iterations over time steps for different preconditioner variants. The red curve shows the number of active nodes for each time step.



Overall linear solver time over time steps.

Method	Setup costs	Solver time	Overall solver time
PA-AMG (CheapSIMPLE)	11,630	4658	16,288
SA-AMG (CheapSIMPLE)	12,250	4564	16,814
CheapSIMPLE (SA-AMG)	12,130	4731	16,861
CheapSIMPLE (SGS)	10,270	9320	19,590

TABLE 7 Two tori impact example—Exemplary timings in (s) of the different preconditioning variants from Table 6 for the full simulation over 200 time steps

Figure 11(A) shows the overall number of linear iterations performed to solve all nonlinear iterations for each time step. All preconditioner variants require exactly the same number of nonlinear iterations per time step due to the particular choice of stopping criteria in (23), ranging between six and 10 nonlinear iterations per time step, while also the number of nonlinear iterations is independent of the size of the active set. Obviously, the SIMPLE based methods need more linear iterations than the AMG based methods. In this example, there is nearly no difference between the nonsmoothed transfer operator variant PA-AMG (CheapSIMPLE) and the smoothed transfer operator variant SA-AMG (CheapSIMPLE). Furthermore, there is no clear and obvious correlation between the number of linear iterations and the number of active nodes. This indicates that the fully coupled multigrid method is robust and efficient with regard to the increasing complexity of the contact configuration over time, which is not the case for cheaper methods such as the SIMPLE based methods. Evidently, one can see a significant drop in the linear iterations for the last 20 time steps of the simulation, which may correspond to the small number of nodes in contact.

When looking at the corresponding solver timings over the time steps in Figure 11(B), one finds the CheapSIMPLE (SA-AMG) method to be very close to the AMG based methods PA-AMG (CheapSIMPLE) and SA-AMG (CheapSIMPLE). For the AMG based methods, one sweep with a CheapSIMPLEC method is applied on each level, which internally uses one sweep with a symmetric Gauss–Seidel iteration for the primary variable and one ILU sweep for the constraint equation. That is, quite a lot of time is invested in the coupling on all levels with the comparably expensive ILU method. In contrast to the AMG based method, the CheapSIMPLE (SA-AMG) method uses two sweeps with a CheapSIMPLE preconditioner for the coupling (on the finest level only). Internally, a three-level AMG multigrid is used with two symmetric Gauss–Seidel sweeps for the level smoother and an ILU sweep for the constraint correction equation. These parameters have been found to result in a reasonably low number of linear iterations. For this example the experiment shows that the CheapSIMPLE (SA-AMG) method needs twice as many iterations as the SA-AMG (CheapSIMPLE) method, but the costs per iteration are only half of the costs of the SA-AMG (CheapSIMPLE). Nevertheless, the AMG based methods seem to have a small advantage, when the number of nodes in contact increases.

Last but not least, the overall timings for the linear solver are reported in Table 7. Except for the CheapSIMPLE (SGS) variant, which is far away from the others, there is no clear winner. The setup costs are quite close, since the same transfer operators have to be built for all methods with only a small difference for smoothed versus nonsmoothed transfer operators. To show a fair comparison, the preconditioner has been rebuilt in every nonlinear iteration step. Of course, reuse strategies might increase amortization of setup costs and reduce overall solver timings when solving actual problems.

7 | CONCLUSION

We have presented algebraic multigrid schemes designed for saddle point problems arising from contact problems using mortar finite element methods. The new fully coupled multigrid scheme has the advantage that the contact constraints are considered on all multigrid levels, which significantly reduces the number of iterations. It gives the user full control over the coupling process by appropriately choosing the solver parameters. Additionally, we have proposed a novel aggregation method for the Lagrange multipliers, which reuses existing aggregation information at the contact interface. We have demonstrated the robustness and efficiency of the overall multigrid method for large examples with increasingly complex contact configurations over time as well as its weak scalability up to 23.9 million unknowns on 480 MPI ranks.

DATA AVAILABILITY STATEMENT

The implementation of the AMG preconditioner developed and applied in this study is openly available in Trilinos at <https://github.com/trilinos/Trilinos>.⁸³ All other data are available from the corresponding author upon reasonable request.

ORCID

Matthias Mayr  <https://orcid.org/0000-0002-2780-1233>

Alexander Popp  <https://orcid.org/0000-0002-8820-466X>

Michael W. Gee  <https://orcid.org/0000-0001-9293-7201>

Wolfgang A. Wall  <https://orcid.org/0000-0001-7419-3384>

REFERENCES

- Gitterle M, Popp A, Gee MW, Wall WA. Finite deformation frictional mortar contact using a semi-smooth Newton method with consistent linearization. *Int J Numer Methods Eng.* 2010;84(5):543-571.
- Seitz A, Popp A, Wall WA. A semi-smooth newton method for orthotropic plasticity and frictional contact at finite strains. *Comput Methods Appl Mech Eng.* 2015;285:228-254.
- Popov VL. *Contact Mechanics and Friction: Physical Principles and Applications.* 2nd ed. Berlin, Germany: Springer; 2017.
- Cavalieri FJ, Cardona A. Three-dimensional numerical solution for wear prediction using a mortar contact algorithm. *Int J Numer Methods Eng.* 2013;96(8):467-486.
- Farah P, Wall WA, Popp A. An implicit finite wear contact formulation based on dual mortar methods. *Int J Numer Methods Eng.* 2016;111(4):325-353.
- Lengiewicz J, Stupkiewicz S. Continuum framework for finite element modelling of finite wear. *Comput Methods Appl Mech Eng.* 2012;205-208:178-188.
- Milanesi E, Brink T, Aghababaei R, Molinari J-F. Emergence of self-affine surfaces during adhesive wear. *Nat Commun* 10. 2019;1116(2019). <https://doi.org/10.1038/s41467-019-09127-8>.
- Sauer RA. Enriched contact finite elements for stable peeling computations. *Int J Numer Methods Eng.* 2011;87(6):593-616.
- Li Q, Popov VL. Adhesive contact between a rigid body of arbitrary shape and a thin elastic coating. *Acta Mech.* 2019;230(7):2447-2453.
- Bonari J, Marulli MR, Hagemeyer N, Mayr M, Popp A, Paggi M. A multi-scale FEM-BEM formulation for contact mechanics between rough surfaces. *Comput Mech.* 2020;65:731-749.
- Vakis AI, Yastrebov VA, Scheibert J, et al. Modeling and simulation in tribology across scales: an overview. *Tribol Int.* 2018;125:169-199.
- Belgacem FB. The mortar finite element method with Lagrange multipliers. *Numer Math.* 1999;84(2):173-197.
- Bernardi C, Maday Y, Patera AT. A new nonconforming approach to domain decomposition: the mortar element method. In: Brezis H, Lions J, eds. *Nonlinear Partial Differential Equations and Their Applications.* London, UK; New York, NY: Pitman/Wiley; 1994:13-51.
- De Lorenzis L, Wriggers P, Hughes TJ. Isogeometric contact: a review. *GAMM-Mitteilungen.* 2014;37(1):85-123.
- Puso MA, Laursen TA. A mortar segment-to-segment contact method for large deformation solid mechanics. *Comput Methods Appl Mech Eng.* 2004a;193(6-8):601-629.
- Puso MA, Laursen TA. A mortar segment-to-segment frictional contact method for large deformations. *Comput Methods Appl Mech Eng.* 2004b;193(45-47):4891-4913.
- Wohlmuth BI. Variationally consistent discretization schemes and numerical algorithms for contact problems. *Acta Numerica.* 2011;20:569-734.
- Wriggers P. *Computational Contact Mechanics.* Berlin/Heidelberg, Germany: Springer-Verlag; 2006.
- Hartmann S, Brunssen S, Ramm E, Wohlmuth BI. Unilateral non-linear dynamic contact of thin-walled structures using a primal-dual active set strategy. *Int J Numer Methods Eng.* 2007;70(8):883-912.
- Hüeber S, Stadler G, Wohlmuth BI. A primal-dual active set algorithm for three-dimensional contact problems with Coulomb friction. *SIAM J Sci Comput.* 2008;30(2):572-596.
- Hüeber S, Wohlmuth BI. A primal-dual active set strategy for non-linear multibody contact problems. *Comput Methods Appl Mech Eng.* 2005;194(27-29):3147-3166.
- Popp A, Gee MW, Wall WA. A finite deformation mortar contact formulation using a primal-dual active set strategy. *Int J Numer Methods Eng.* 2009;79(11):1354-1391.
- Popp A, Gitterle M, Gee MW, Wall WA. A dual mortar approach for 3D finite deformation contact with consistent linearization. *Int J Numer Methods Eng.* 2010;83(11):1428-1465.
- Wohlmuth BI. A mortar finite element method using dual spaces for the Lagrange multiplier. *SIAM J Numer Anal.* 2000b;38(3):989-1012.
- Wohlmuth B. *Discretization Techniques and Iterative Solvers Based on Domain Decomposition.* Vol 17. Heidelberg, Germany: Springer; 2001.
- Popp A, Wall WA. Dual mortar methods for computational contact mechanics – overview and recent developments. *GAMM-Mitteilungen.* 2014;37(1):66-84.
- Popp A, Wohlmuth BI, Gee MW, Wall WA. Dual quadratic mortar finite element methods for 3D finite deformation contact. *SIAM J Sci Comput.* 2012;34:B421-B446.
- Wohlmuth BI, Popp A, Gee MW, Wall WA. An abstract framework for a priori estimates for contact problems in 3D with quadratic finite elements. *Comput Mech.* 2012;49:735-747.
- De Lorenzis L, Temizer I, Wriggers P, Zavarise G. A large deformation frictional contact formulation using NURBS-based isogeometric analysis. *Int J Numer Methods Eng.* 2011;87(13):1278-1300.
- Temizer I, Wriggers P, Hughes TJR. Contact treatment in isogeometric analysis with NURBS. *Comput Methods Appl Mech Eng.* 2011;200(9,Är 12):1100-1112.
- Temizer, I., Wriggers, P., and Hughes, T. J. R. (2012). Three-dimensional mortar-based frictional contact treatment in isogeometric analysis with NURBS. *Comput Methods Appl Mech Eng.* 209–212(0), 115–128.

32. Wunderlich L, Seitz A, Alaydn MD, Wohlmuth BI, Popp A. Biorthogonal splines for optimal weak patch-coupling in isogeometric analysis with applications to finite deformation elasticity. *Comput Methods Appl Mech Eng*. 2019;346:197-215.
33. Seitz A, Farah P, Kremheller J, Wohlmuth BI, Wall WA, Popp A. Isogeometric dual mortar methods for computational contact mechanics. *Comput Methods Appl Mech Eng*. 2016;301:259-280.
34. Franke M, Hesch C, Betsch P. An augmentation technique for large deformation frictional contact problems. *Int J Numer Methods Eng*. 2013;94(5):513-534.
35. Hiermeier M, Wall WA, Popp A. A truly variationally consistent and symmetric mortar-based contact formulation for finite deformation solid mechanics. *Comput Methods Appl Mech Eng*. 2018;342:532-560.
36. Popp A, Seitz A, Gee MW, Wall WA. Improved robustness and consistency of 3D contact algorithms based on a dual mortar approach. *Comput Methods Appl Mech Eng*. 2013;264:67-80.
37. Hackbusch W. *Iterative Solution of Large Sparse Systems of Equations, Volume 95 of Applied Mathematical Sciences*. New York, NY: Springer; 1994.
38. Saad Y. *Iterative Methods for Sparse Linear Systems*. 2nd ed. Philadelphia, PA: SIAM; 2003.
39. Hackbusch W. *Multi-Grid Methods and Applications, Volume 4 of Series in Computational Mathematics*. 1st ed. New York, NY: Springer; 1985.
40. Trottenberg U, Oosterlee C, Schüller A. *Multigrid*. Amsterdam, Netherlands: Academic Press; 2001.
41. Wiesner TA, Popp A, Gee MW, Wall WA. Algebraic multigrid methods for dual mortar finite element formulations in contact mechanics. *Int J Numer Methods Eng*. 2018;144(4):399-430.
42. Janka A. Smoothed aggregation multigrid for a Stokes problem. *Comput Vis Sci*. 2008;11(3):169-180.
43. Griebel M, Neunhoffer T, Regler H. Algebraic multigrid methods for the solution of the Navier—Stokes equations in complicated geometries. *Int J Numer Methods Fluids*. 1998;26(3):281-301.
44. Lonsdale R. *An Algebraic Multigrid Scheme for Solving the Navier-Stokes Equations on Unstructured Meshes, Volume 7 of Numerical Methods in Laminar and Turbulent Flow*. Swansea, UK: Pineridge Press; 1991:1432-1442.
45. Wieners C, Wohlmuth B. *A General Framework for Multigrid Methods for Mortar Finite Elements. Technical Report 415*. Augsburg, Germany: Universität Augsburg, Institut für Mathematik; 1999.
46. Wohlmuth B. A multigrid methods for saddlepoint problems arising from mortar finite element discretizations. *ETNA Electron Trans Numer Anal*. 2000a;11:43-54.
47. Braess D, Dryja M, Hackbush W. A multigrid method for nonconforming FE-discretisations with application to non-matching grids. *Computing*. 1999;63(1):1-25.
48. Gopalakrishnan J, Pasciak JE. Multigrid for the mortar finite element method. *SIAM J Numer Anal*. 2000;37(3):1029-1052.
49. Wohlmuth B, Krause R. Multigrid methods based on the unconstrained product space for mortar finite element discretizations. *SIAM J Numer Anal*. 2001;39:192-213.
50. Krause RH, Wohlmuth BI. Multigrid methods for mortar finite elements. *Multigrid Methods, VI (Gent, 1999), Volume 14 of Lecture Notes in Computational Science and Engineering*. Berlin, Germany: Springer; 2000:136-142.
51. Braess D, Dahmen W. Stability estimates of the mortar finite element method for 3-dimensional problems. *East-West J Numer Math*. 1998;6:249-264.
52. Braess D, Dahmen W, Wieners C. A multigrid algorithm for the mortar finite element method. *SIAM J Numer Anal*. 2000;37(1):48-69.
53. Wieners C, Wohlmuth B. Duality estimates and multigrid analysis for saddle point problems arising from mortar discretizations. *SIAM J Sci Comput*. 2003;24:2163-2184.
54. Zulehner W. A class of smoothers for saddle point problems. *Computing*. 2000;65:227-246.
55. Adams MF. Algebraic multigrid methods for constrained linear systems with applications to contact problems in solid mechanics. *Numer Linear Algebra Appl*. 2004;11(2-3):141-153.
56. Kornhuber R, Krause R. Adaptive multigrid methods for Signorini's problem in linear elasticity. *Comput Vis Sci*. 2001;4:9-20.
57. Wohlmuth BI, Krause R. Monotone multigrid methods on nonmatching grids for nonlinear multibody contact problems. *SIAM J Sci Comput*. 2003;25(1):324-347.
58. Stüben K. Appendix A: An Introduction to Algebraic Multigrid. *Multigrid*. Amsterdam, Netherlands: Elsevier Academic Press; 2001:413-533.
59. Vaněk P. Acceleration of convergence of a two-level algorithm by smoothing transfer operators. *Appl Math*. 1992;37:265-274.
60. Vaněk P, Mandel J, Brezina M. Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems. *Computing*. 1996;56:179-196.
61. Vaněk P, Janka A, Guillard H. *Convergence of Algebraic Multigrid Based on Smoothed Aggregation II: Extension to a Petrov-Galerkin Method INRIA Technical Report 3683*. Sophia Antipolis, France: INRIA; 1999.
62. Vaněk P, Brezina M, Mandel J. Convergence of algebraic multigrid based on smoothed aggregation. *Numer Math*. 2001;88:559-579.
63. Sala M, Tuminaro RS. A new Petrov-Galerkin smoothed aggregation preconditioner for nonsymmetric linear systems. *SIAM J Sci Comput*. 2008;31(1):143-166.
64. Gee MW, Küttler U, Wall WA. Truly monolithic algebraic multigrid for fluid-structure interaction. *Int J Numer Methods Eng*. 2011;85(8):987-1016.
65. Mayr M, Noll M, Gee MW. A hybrid interface preconditioner for monolithic fluid-structure interaction solvers. *Adv Model Simul Eng Sci*. 2020;7:15.
66. Zienkiewicz OC, Taylor RL, Fox DD. *The Finite Element Method for Solid and Structural Mechanics*. 7th ed. Oxford, UK: Butterworth-Heinemann; 2014.

67. Lamichhane BP, Stevenson RP, Wohlmuth BI. Higher order mortar finite element methods in 3D with dual Lagrange multiplier bases. *Numer Math*. 2005;102(1):93-121.
68. Christensen PW, Klarbring A, Pang JS, Strömberg N. Formulation and comparison of algorithms for frictional contact problems. *Int J Numer Methods Eng*. 1998;42(1):145-173.
69. Hintermüller M, Ito K, Kunisch K. The primal-dual active set strategy as a semi-smooth Newton method. *SIAM J Optim*. 2002;13(3):865-888.
70. Qi L, Sun J. A nonsmooth version of Newton's method. *Math Program*. 1993;58(1):353-367.
71. Saad Y, Schultz M. GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J Sci Stat Comput*. 1986;7(3):856-869.
72. Benzi M. Preconditioning techniques for large linear systems: a survey. *J Comput Phys*. 2002;182(2):418-477.
73. Vaněk P, Mandel J, Brezina M. *Algebraic Multigrid on Unstructured Meshes. Technical Report*. Denver, CO: University of Colorado at Denver; 1994.
74. Gee MW, Hu JJ, Tuminaro RS. A new smoothed aggregation multigrid method for anisotropic problems. *Numer Linear Algebra Appl*. 2009;16:19-37.
75. Patankar SV, Spalding DB. A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows. *Int J Heat Mass Transf*. 1972;15:1787-1972.
76. Tezduyar TE, Sathe S, Stein K. Solution techniques for the fully discretized equations in computation of fluid-structure interactions with the space-time formulations. *Comput Methods Appl Mech Eng*. 2006;195(41-43):5743-5753.
77. Verdugo F, Wall WA. Unified computational framework for the efficient solution of n -field coupled problems with monolithic schemes. *Comput Methods Appl Mech Eng*. 2016;310:335-366.
78. Wabro M. Coupled algebraic multigrid methods for the Oseen problem. *Comput Vis Sci*. 2004;7:141-151.
79. Langer U, Yang H. Numerical simulation of fluid-structure interaction problems with hyperelastic models: a monolithic approach. *Math Comput Simul*. 2018;145:186-208.
80. Braess D, Sarazin R. An efficient smoother for the stokes problem. *Appl Numer Math*. 1997;23:3-19.
81. Danowski C, Gravemeier V, Yoshihara L, Wall WA. A monolithic computational approach to thermo-structure interaction. *Int J Numer Methods Eng*. 2013;95(13):1053-1078.
82. Berger-Vergiat L, Glusa CA, Hu JJ, et al. *MueLu User's Guide 2.0. Technical Report SAND2019-0537*. Albuquerque, NM: Sandia National Laboratories; 2019.
83. <https://trilinos.github.io>.
84. <https://trilinos.github.io/muelu.html>.
85. Notay Y. A new analysis of block preconditioners for saddle point problems. *SIAM J Matrix Anal Appl*. 2014;35(1):143-173.
86. Bramble JH, Pasciak JE, Vassilev AT. Analysis of the inexact Uzawa algorithm for saddle point problems. *SIAM J Numer Anal*. 1997;34(3):1072-1092.
87. Elman HC, Golub GH. Inexact and preconditioned Uzawa algorithms for saddle point problems. *SIAM J Numer Anal*. 1994;31(6):1645-1661.
88. Zulehner W. Analysis of iterative methods for saddle point problems: a unified approach. *Math Comput*. 2002;71:479-505.
89. Patankar SV. *Numerical Heat Transfer and Fluid Flow*. New York, NY: Hemisphere Publishing Corporation; 1980.
90. Elman H, Howle VE, Shadid J, Shuttleworth R, Tuminaro R. A taxonomy and comparison of parallel block multi-level preconditioners for the incompressible Navier-Stokes equations. *J Comput Phys*. 2008;227(3):1790-1808.
91. Van Doornaal J, Raithby G. Enhancements of the simple method for predicting incompressible fluid flows. *Numer Heat Transf*. 1984;7(2):147-163.
92. <https://trilinos.github.io/xpetra.html>.
93. Keller C, Gould NIM, Wathen AJ. Constrained preconditioning for indefinite linear systems. *SIAM J Matrix Anal Appl*. 2000;21(4):1300-1317.
94. BACI: A comprehensive multi-physics simulation framework. <https://baci.pages.gitlab.lrz.de/website>.
95. Wiesner TA, Gee MW, Prokopenko A, Hu J. *The MueLu Tutorial. Technical Report*. Albuquerque, NM: Sandia National Laboratories; 2014.
96. Yang B, Laursen T. A contact searching algorithm including bounding volume trees applied to finite sliding mortar formulations. *Comput Mech*. 2008;41(2):189-205.
97. Popp A. *Mortar Methods for Computational Contact Mechanics and General Interface Problems* [Ph.D. thesis]. Technische Universität München; 2012.
98. Chung J, Hulbert G. A time integration algorithm for structural dynamics with improved numerical dissipation: the generalized- α method. *J Appl Mech*. 1993;60(2):371-375.

How to cite this article: Wiesner TA, Mayr M, Popp A, Gee MW, Wall WA. Algebraic multigrid methods for saddle point systems arising from mortar contact formulations. *Int J Numer Methods Eng*. 2021;122:3749-3779. <https://doi.org/10.1002/nme.6680>