Journal of Computational Physics 231 (2012) 7315–7334

Contents lists available at SciVerse ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

An efficient preconditioner for monolithically-coupled large-displacement fluid–structure interaction problems with pseudo-solid mesh updates

Richard L. Muddle [a,b], Milan Mihajlović [b], Matthias Heil [a,*]

[a] School of Mathematics, University of Manchester, Manchester M13 9PL, UK
[b] School of Computer Science, University of Manchester, Manchester M13 9PL, UK

ARTICLE INFO

Article history:
Received 21 December 2010
Received in revised form 19 June 2012
Accepted 1 July 2012
Available online 11 July 2012

Keywords:
Fluid–structure interaction
Pseudo-solid mesh updates
Monolithic discretisation
Preconditioning
Multi-physics
Algebraic multigrid
Krylov methods

ABSTRACT

We present a block preconditioner for the efficient solution of the linear systems that arise when employing Newton's method to solve monolithically-coupled large-displacement fluid–structure interaction problems in which the update of the moving fluid mesh is performed by the equations of large-displacement elasticity. Following a theoretical analysis of the preconditioner, we propose an efficient implementation that yields a solver with near-optimal computational cost, in the sense that the time for the solution of the linear systems scales approximately linearly with the number of unknowns. We evaluate the performance of the preconditioner in selected two- and three-dimensional test problems.

1. Introduction

Large-displacement fluid–structure-interaction (FSI) problems are multi-physics problems in which elastic solids interact with finite-Reynolds-number flows. Applications exist in many areas such as physiological fluid mechanics [1,2], the design of parachutes [3], tent structures [4], and artificial heart valves [5]. The accurate, robust and efficient numerical simulation of such problems poses considerable challenges, not just from a mathematical but also from a software engineering perspective. A key objective in the design of solvers for all multi-physics problems is the ability to re-use existing, often highly-optimised solvers for the constituent single-physics (here fluid and solid mechanics) problems. In so-called partitioned (or segregated) approaches, this re-use is achieved at the level of the nonlinear solvers, typically by coupling the existing single-physics solvers in a fixed-point iteration: starting from an initial guess for the shape of the fluid-loaded solid, compute the fluid flow (keeping the geometry of the domain fixed); determine the traction that the fluid exerts onto the elastic solid; compute the resulting deformation of the solid (keeping the fluid traction constant); and iterate until convergence. Such solvers are relatively easy to implement but they often suffer from a serious lack of robustness and tend to require heavy under-relaxation (resulting in slow convergence rates) to avoid the divergence of the fixed-point iteration, even in cases when good initial guesses for the solution are available. A variety of methods have been developed to accelerate the convergence of the fixed-point iteration (e.g. vector-based [6,7] or component-wise [8] Aitken extrapolation, and steepest decent methods; see e.g. [9] for a recent comparison of some of these methods). In many cases such partitioned approaches work satisfactorily. However, there are also many applications (particularly problems in which the fluid interacts with a

* Corresponding author.
E-mail address: M.Heil@maths.man.ac.uk (M. Heil).

0021-9991/$ - see front matter © 2012 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jcp.2012.07.001

thin-walled solid of comparable or smaller density – a situation that is encountered frequently in physiological FSI problems) where the lack of robustness due to the so-called "added mass effect" [10] presents a significant problem. The convergence problems in the fixed-point iteration may be bypassed by employing so-called loosely coupled schemes [11]. However, these are only applicable to unsteady problems and obtain a solution by solving the constituent single-physics problems sequentially; see [12–14] for a discussion of the accuracy and stability of such methods.

The main alternative to partitioned approaches are so-called monolithic solvers which operate directly on the large system of nonlinear algebraic equations that arises from the fully-coupled implicit discretisation of the fluid and solid equations. Starting from an initial guess for the vector of the discrete unknowns, $\mathbf{x}^{[0]}$, and the corresponding vector of discrete residuals $\mathbf{r}^{[0]} = \mathbf{r}(\mathbf{x}^{[0]})$, Newton's method can be used to solve the problem via the repeated solution of (large) linear systems of the form

$$J^{[m]}\,\Delta\mathbf{x}^{[m]} = -\mathbf{r}^{[m]}, \tag{1}$$

followed by the update $\mathbf{x}^{[m+1]} = \mathbf{x}^{[m]} + \Delta\mathbf{x}^{[m]}$. Here $J^{[m]}$ is the Jacobian matrix formed from the derivatives of the discrete residuals, $\mathbf{r}$, with respect to the discrete unknowns, $\mathbf{x}$. Provided a good initial guess for the solution can be supplied (by continuation methods or timestepping), Newton's method is extremely robust and converges quadratically.

While attractive from a theoretical perspective, the practical implementation of such solvers poses its own challenges. From a software engineering point-of-view, it is difficult to implement monolithic solvers starting from separate fluid and solid mechanics codes, particularly if these are "black-box" closed-source codes for which the residuals and their derivatives are difficult to obtain. Interface Newton–Krylov methods (e.g. [15]) bypass this problem by employing a Newton method that operates only on the variables that define the position of the FSI interface, but even these methods require (or benefit from) additional knowledge about the underlying fluid and solid equations and their specific discretisation. Monolithic solvers are easiest to implement within a software framework that facilitates the formulation of multi-physics interactions via small, hierarchical modifications to any already-existing single-physics capabilities. This requirement lends itself to an object-oriented design in which multi-physics interactions are easily implemented using techniques such as function overloading, templating and multiple inheritance. We refer to [16] for a discussion of how this approach is implemented in `oomph-lib`, the object-oriented multi-physics finite element library (available as open-source software at `http://www.oomph-lib.org`), which was used for the computations presented in this paper.

The key mathematical challenge in the solution of fluid–structure interaction problems by monolithic approaches is how to efficiently solve the large system of nonlinear algebraic equations arising from the fully-coupled implicit discretisation of the fluid and solid equations, a task has frequently been described as being prohibitively expensive (see, e.g., [17,18]). However, more recent studies [19,20] have shown that monolithic approaches are not only competitive with partitioned schemes, but often outperform them, even in problems in which partitioned solvers do not suffer from any convergence problems. Possible approaches to the solution of the nonlinear equations are the use of nonlinear block Gauss–Seidel or block-Newton methods [21,22], methods that operate on the FSI interface problem [23,24], or partitioned Newton methods [25]. In this paper we employ Newton's method in its exact form and focus on the development of an efficient preconditioner for the solution of the linear system (1) by Krylov subspace solvers such as GMRES. We will exploit the block structure of the Jacobian matrix to derive a preconditioner, $\mathcal{P}$, chosen such that GMRES, applied to the preconditioned linear system

$$\mathcal{P}^{-1}J\,\Delta\mathbf{x} = -\mathcal{P}^{-1}\mathbf{r} \tag{2}$$

(where we have dropped the superscript m) converges in a number of iterations that is much smaller than the iteration count for the original linear system (1). The development of the preconditioner is guided by two requirements:

1. $\mathcal{P}$ should be chosen such that the preconditioned matrix $\mathcal{P}^{-1}J$ has a spectrum that contains only a few small clusters of eigenvalues where each cluster is tightly bounded away from the origin of the complex plane. If these bounds on the spectrum are independent of the discretisation, GMRES will converge in a near constant number of iterations regardless of the number of unknowns in the problem [26, p. 54].
2. The setup and application of the preconditioner (i.e. the solution of linear systems of the form $\mathcal{P}\mathbf{u} = \mathbf{v}$) should have optimal computational cost, i.e. be linear in the number of unknowns.

A preconditioning strategy that satisfies these two requirements will result in a solver that itself has optimal computational cost.

We restrict ourselves to ALE-based discretisations of the FSI problem in which the fluid equations are solved on a body-fitted, moving mesh [27]. In earlier work [28,19] we developed an efficient block triangular preconditioner for the linear systems arising from the monolithic discretisation of such problems for the special case in which the update of the fluid mesh in response to the wall deformation is performed by a (user-specified) algebraic node update function. This approach allows fast, sparse node updates but requires significant user input and is restricted to relatively simple geometries and to problems in which the deformation of the fluid-loaded solid can be anticipated *a priori*. Furthermore, the block-triangular preconditioners were obtained by omitting certain coupling blocks in the Jacobian matrix in an *ad hoc* manner. This made it difficult to analyse the preconditioners theoretically, and forced us to evaluate their performance exclusively by means of numerical experiments. In the current paper we present a new preconditioning methodology for the alternative approach in which the