Computers and Structures 171 (2016) 9–17

Contents lists available at ScienceDirect

Computers and Structures

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

Improving the performance of the partitioned QN-ILS procedure for fluid–structure interaction problems: Filtering

R. Haelterman [a,*], A.E.J. Bogaers [b], K. Scheufele [d], B. Uekermann [c], M. Mehl [d]

[a] Royal Military Academy, Dept. of Mathematics (MWMW), Renaissancelaan 30, B-1000 Brussels, Belgium
[b] Advanced Mathematical Modelling, CSIR, Meiring Naudé Road, Brummeria, Pretoria, South Africa
[c] Institute for Advanced Study, Technische Universität München, Lichtenbergstraße 2a, D-85748 Garching b. München, Germany
[d] Universität Stuttgart, Universitätsstraße 38, D-70569 Stuttgart, Germany

ARTICLE INFO

Article history:
Received 2 October 2015
Accepted 18 April 2016
Available online 17 May 2016

Keywords:
Fluid–structure interaction
Quasi-Newton method
Least squares
Filtering

ABSTRACT

The Quasi-Newton Inverse Least Squares method has become a popular method to solve partitioned interaction problems. Its performance can be enhanced by using information from previous time-steps if care is taken of the possible ill-conditioning that results. To enhance the stability, filtering has been used. In this paper we show that a relatively minor modification to the filtering technique can substantially reduce the required number of iterations.

1. Introduction

Often in nature different systems interact, like fluids and structures, heat and electricity, and populations of species. From the growing number of conferences, publications and software releases it is clear that *in silico* simulations of these kinds of coupled systems are becoming increasingly important in the engineering community. Examples can be found in aeronautics (e.g. [38,40–43,62,77,83]), bio-medical science (e.g. [3,7,16,31,32,48,58,78,86,88]), civil engineering (e.g. [13,33,46,60,61,76,80,81,91,92]), plasma physics (e.g. [55,56]), to name but a few. In this paper we will focus solely on fluid–structure interactions.

For these types of problems powerful solvers often already exist for each physical domain (e.g. structural or fluid). Even so, development of similar tools for multi-physics problems is still ongoing and the paths followed to obtain such a solver can be broadly put in one of the following categories:

- *Monolithic or simultaneous solution*: the whole problem is treated as a monolithic entity and solved simultaneously with a specialized *ad hoc* solver.
- *Partitioned solution*: the physical components are treated as isolated entities that are solved separately. Interaction is modeled as forcing terms and/or boundary conditions.

The relative merits of these methods are very problem dependent. The advantage of the monolithic approach is the enhanced stability [71]. This however comes at the cost of having to develop specialized software for each type of interaction problem, where the resulting solution systems are often very large. Furthermore, it can be inappropriate to use the same basic formulation for both types of problems, which further forces the user to treat non-linearities in the same way for all components. Despite this the monolithic approach has proven to be a very popular method, e.g. [4,7,14,59,79,84].

The partitioned approach allows for the use of available specialized solvers for each physical component, on the condition that the coupling effects can be treated efficiently. The latter is often feasible for problems where the systems only weakly interact. Strongly coupled problems, on the other hand, still pose a real challenge. Many articles can be found on partitioned methods in the literature, e.g. [13,23,44,68–70,75,77,87].

In this paper we will solely focus on the partitioned approach, as the main aim of this work is to improve the performance of the Quasi-Newton Least Squares family of methods, in particular QN-ILS [21], which has found widespread acceptance as a means to accelerate the convergence of partitioned solvers. We are not concerned with the solution process of the constituent physical

* Corresponding author.
E-mail addresses: Robby.Haelterman@mil.be (R. Haelterman), abogaers@csir.co.za (A.E.J. Bogaers), klaudius.scheufele@ipvs.uni-stuttgart.de (K. Scheufele), uekerman@in.tum.de (B. Uekermann), miriam.mehl@ipvs.uni-stuttgart.de (M. Mehl).

http://dx.doi.org/10.1016/j.compstruc.2016.04.001
0045-7949/© 2016 Elsevier Ltd. All rights reserved.

problems as these are assumed to be handled by specialized solvers which we assume to be *black box* operations of which no specific details can be modified or even assessed (e.g. the Jacobian). Furthermore we will assume that the computing time of these black boxes is sufficiently high that the actual computing time of the quasi-Newton step can be neglected.

To accelerate the convergence of a time-dependent problem it can be beneficial to use "histories" from previous time-steps to construct an approximation of the (inverse) Jacobian at the current time-step. This might however lead to numerical breakdown, as the risk of constituent vectors becoming (nearly) linearly dependent increases. To avoid this, QR-filtering has been applied in the past [22]. In this paper we try to improve the convergence performance of QN-ILS by introducing a better form of filtering.

This paper is organized as follows: in Section 2 we give a short overview of QN-ILS; in Section 3 we explain the idea of recovery of data from previous time-steps, while in Section 4 we address the topic of filtering. The currently used method of QR-filtering is explained, together with two new approaches, one based on the ideas of the orthogonalization performed in QR-decomposition, the other on that of Proper Orthogonal Decomposition. In Section 5 the performance of the different filtering techniques is tested and compared using different numerical applications, after which we end with a short conclusion.

2. The Quasi-Newton Inverse Least Squares (QN-ILS) method

2.1. Problem setting

In general we are interested in non-linear surface-coupled problems that can be mathematically stated in the following form:

$$\begin{cases} F(g) = p \\ S(p) = g, \end{cases} \tag{1}$$

where $F : D_F \subset \mathbb{R}^m \to \mathbb{R}^n : g \mapsto F(g)$ and $S : D_S \subset \mathbb{R}^n \to \mathbb{R}^m : p \mapsto S(p)$.

Each equation describes (the discretized equations of) a physical problem that is spatially decomposed. In fluid–structure interaction problems $F(g) = p$ could give the pressure p on the interface between fluid and structure for a given geometry g, while $S(p) = g$ could give the deformed geometry of that same interface under influence of the pressure exerted on it by the fluid.

We limit p and g to values on the interface between the two physical problems. In this way the physically decoupled nature of the problem is exploited. This approach can be regarded as a special case of *heterogeneous domain decomposition methods* [28] and limits the number of variables that the coupling technique will be dealing with, even though the black box solvers that give $F(g)$ and $S(p)$ might use a substantially higher number of internal variables; for instance in the case of a fluid–structure interaction problem where the pressure is passed from the fluid to the structure, the fluid velocity is an internal variable for the flow solver as are all nodal values of the pressure that are not on the interface.

Alternatively, (1) can be written as the fixed point problem

$$F(S(p)) = H(p) = p \tag{2}$$

or the root-finding problem

$$H(p) - p = K(p) = 0. \tag{3}$$

Using (2) or (3) means that we have actually lumped both systems F and S together into one system (either H or K), which in general has a lower number of variables than the sum of the number of variables of both constituent systems F and S.

We assume that H and K satisfy the following hypotheses, which are typical when working with Newton and quasi-Newton type methods [73]:

(1) H (and hence K) are continuously differentiable in an open set D.
(2) $K(p) = 0$ has one solution p^* in D_K.
(3) $(K'(p))^{-1}$ exists and is continuous in an open set containing p^*.

Remark 2.1. We assume the operations $F(g)$ and $S(p)$ (and hence $H(p)$ and $K(p)$) are black box systems, representing the propriety solvers, with a high computational cost and of which nothing is known about the Jacobian; neither do we make assumptions about this Jacobian like sparseness, symmetry, etc. For that reason we count the performance of a method by the number of times $F(g)$ or $S(p)$ are executed. Requirements like actual CPU-time or storage are not taken into consideration.

Remark 2.2. While we write the equations in (1) in explicit form, this is only for convenience; any form is usable as long as for a given value of g (resp. p) a corresponding value of p (resp. g) can be computed that satisfies the equations in (1).

Remark 2.3. We could have used (and sometimes will use)

$$S(F(g)) - g = 0 \tag{4}$$

instead of (2). The choice between both can depend on

- practical implementation issues due to the solvers used;
- the relative sizes of n and m. If $n < m$, resp. $n > m$, the use of (3), resp. (4), will result in a problem that is defined on a space with the lowest dimension.

Remark 2.4. When (1) is derived from a physical problem, it often represents the equations obtained after discretizing the continuous equations in time and space, and thus only represents the evolution over one time-step (see Section 3). This is an example of how we could be presented with a series of related problems.

In this context we can write (1) as

$$\begin{cases} F_{t+1}(g, p_t, g_t) = p \\ S_{t+1}(p, p_t, g_t) = g, \end{cases} \tag{5}$$

where the subscript $t + 1$ $(t = 0, 1, \ldots)$ denotes the time-level at which the problem is solved. The solution of (5) will give the values of p and g at that time-level (p_{t+1}, resp. g_{t+1}); the extra arguments p_t and g_t are added to show that the solution at the next time-level depends on the values at the previous time-level.[1]

In what follows we will almost always simply write $F(g)$ and $S(p)$ and assume it is clear from the context that this either describes an isolated problem or a problem solved over one time-step.

2.2. QN-ILS

The Quasi-Newton Least Squares family of algorithms was developed starting from the Block Quasi-Newton Least Squares algorithm (BQN-LS) [19,20,88] even though the name BQN-LS was not used until later [21]. From BQN-LS, the quasi-Newton Least Squares algorithm (QN-LS) was developed, which was first introduced in [51]. BQN-LS and QN-LS are very closely related to the point that for affine problems both are algebraically identical [54].

[1] It is possible that it depends on more than one of the previous time-levels.