Computers & Fluids 88 (2013) 891–896

Contents lists available at ScienceDirect

Computers & Fluids

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

Developments on the Broyden procedure to solve nonlinear problems arising in CFD

L. Berenguer *, D. Tromeur-Dervout [1]

Université de Lyon, CNRS, Université Lyon 1, Institut Camille Jordan UMR 5208, 43 bd du 11 novembre 1918, F-69622 Villeurbanne-Cedex, France

ARTICLE INFO

Article history:
Received 14 March 2013
Received in revised form 27 September 2013
Accepted 1 October 2013
Available online 10 October 2013

Keywords:
Quasi-Newton methods
Time-pipelining
Restricted Additive Schwarz

ABSTRACT

This paper is devoted to the solution of nonlinear time-dependant partial differential equations arising in CFD using Broyden's method in the parallel computing framework. We first use Broyden's method in the context of the domain decomposition: we propose to update the Restricted Additive Schwarz preconditioner from one Newton iteration to another when a Newton–Krylov method is used. We also investigate a time-pipelining method where Broyden's method is used as a solver of the nonlinear problem of each time step.

1. Introduction

We consider the solution of differential equations of the form Eq. (1) for a given initial condition $x(0) = x_0$ and suitable boundary conditions.

$$A\dot{x} = f(x,t) \tag{1}$$

In Eq. (1), $f \in C^1(\Omega, \mathbb{R}^n)$, for Ω an open set in $\mathbb{R}^n \times \mathbb{R}^*$ and $A \in \mathbb{R}^{n\times n}$. This equation is a linear differential–algebraic equation (DAE) if the matrix A is singular. The time discretization of Eq. (1) via backward differentiation formulas leads to solving a system of nonlinear equations $F(x) = 0$ for $F : \mathbb{R}^n \to \mathbb{R}^n$ at each time step.

Domain decomposition methods are often used to parallelize the solution procedure of Eq. (1): the domain decomposition can be applied at the nonlinear level [1], is generally applied to the linearized equation as in this paper. The Newton–Krylov–Schwarz method (NKS) [2] solves the linear system of each Newton iteration by a Krylov subspace method preconditioned by a Schwarz preconditioner such as the Restricted Additive Schwarz [3]. This NKS method and has widely be applied to CFD problems (see for example [4–6]). When Newton–Krylov methods are used, providing preconditioners for the successive linear systems is a critical point. A balance must be found between the ability of the preconditioners to reduce the number of Krylov iterations, and their computational cost (setup and application to vectors). There are usually only slight changes between two consecutive linear systems, that is why the same preconditioning matrix is often used for successive Newton iterations, until it loses its efficiency. To improve its efficiency and longevity, one may want to update the preconditioner using the secant condition [7,8]. We propose to extend this idea to the Restricted Additive Schwarz (RAS) preconditioner.

The strong scaling achieved by such spatial domain decomposition methods declines when the number of domains increases for computational reasons (balance between computation and communication) and numerical reasons (decrease of the convergence rate when the size of the problem grows). In order to improve the parallelism, a complementary approach is to achieve parallelism across time. The main difficulty comes from the sequential nature of the equation: the solution at the previous time steps is required to compute the solution at the current time step. This constraint occurs for all time integrators used in computational fluid dynamics of unsteady problems. During the past years, attempts to develop a numerical integrator that possesses time parallelism, such as the pipelined spectral deferred correction [9,10], were made. These algorithms can be viewed as multiple shooting methods (see Bellen et al. [11,12], Guibert and Tromeur-Dervout [13] and references therein).

A short review of the Broyden's update is given in the first section of this paper. The aim of Section 2 is to extend the idea of a Broyden's update of the preconditioner to domain decomposition preconditioners such as the Restricted Additive Schwarz preconditioner. We discuss the practical issues such as the need of a restarted algorithm, and we provide numerical experiments for the lid-driven cavity problem. In the third section of this paper we study the parallelization of Euler steps when the nonlinear solver is a quasi-Newton method [14]. We particularly discuss the choice

* Corresponding author.
E-mail addresses: laurent.berenguer@univ-lyon1.fr (L. Berenguer), damien.tromeur-dervout@univ-lyon1.fr (D. Tromeur-Dervout).
[1] Principal corresponding author.

0045-7930/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.compfluid.2013.10.002

of the initial guess for the first iteration of each time step. We also propose to propagate the update across the time steps. Numerical results are given for an unsteady version of the Bratu problem, showing that the total number of nonlinear iterations is significantly reduced when few time steps are pipelined.

2. The Broyden's method

This section introduces the Broyden's method to solve the nonlinear problem $F(x) = 0$ arising at each time step. Throughout the paper, the notations $F_k = F(x_k)$ and $\Delta F_k = F_{k+1} - F_k$ are used. Starting from an approximation of the inverse Jacobian matrix G_0, the quasi-Newton update of G_k that satisfies the secant condition:

$$\Delta x_k = G_{k+1}\Delta F_k \tag{2}$$

is given by:

$$G_{k+1} = G_k + (\Delta x_k - G_k\Delta F_k)\frac{v_k^T}{v^T\Delta f_k}\text{for some } v_k \tag{3}$$

Usually, v_k is taken as ΔF_k or $G_k^T\Delta x_k$:

- If $v_k = G_k^T\Delta x_k$, then G_{k+1} minimizes the Frobenius norm $\|G_{k+1}^{-1} - G_k^{-1}\|_F$.
- If $v_k = \Delta F_k$, then G_{k+1} minimizes $\|G_{k+1} - G_k\|_F$.

In both cases, the proof can be derived in straightforward manner from the proof of Theorem 4.1 in [14].

Algorithm 1. Broyden's method to solve $F(x) = 0$

Require x_0 and G_0
1: $F_1 = F(x_1)$
2: **repeat**
3: $\Delta x_k = -G_kF_k$
4: $x_{k+1} = x_k + \Delta x$
5: $F_{k+1} = F(x_{k+1})$
6: $\Delta F_k = F_{k+1} - F_k$
7: $v^T = \Delta x_k^T G_k$ or $v^T = \Delta F_k^T$
8: $G_{k+1} = G_k + (\Delta x_k - G_k\Delta F_k)\frac{v^T}{v^T\Delta F_k}$
9: **until** convergence

Therefore, even if G_0 is a sparse matrix, G_k is not. Consequently, the matrix G_k is never formed, we only compute its application to a vector. Typically, we start with G_0 given by the LU factorization of the Jacobian matrix $J(x_0)$, and G_k is saved as $G_k = G_0 + U_kV_k^T$ where U_k and V_k are $n \times k$ matrices where the k columns correspond to the k rank-one updates. Then, the application of G_k to a vector z involves one forward and backward substitutions to compute G_0z, and two matrix–vector products: $U_k(V_k^Tz)$. It also should be added that a linesearch method is often used to improve the convergence of Newton-like methods: the step 4 of Algorithm 1 is replaced by $x_{k+1} = x_k + \lambda\Delta x$ where λ minimizes the norm of the residual F_{k+1}.

3. Application of the update to preconditioners based upon domain decomposition

In this part, we study the update of the Restricted Additive Schwarz (RAS) preconditioner using the secant condition.

3.1. Rank-one update of the RAS preconditioner

The RAS preconditioner of the linear system $J(x)\Delta x = -F(x)$ decomposed in s overlapping subdomains, is given by:

$$M_{RAS}^{-1} = \sum_{i=1}^{s}(\widetilde{R}^i)^T(J^i(x))^{-1}R^i \tag{4}$$

where R_i is the restriction operator of the *ith* subdomain including the overlap, and $\widetilde{R}_i$ is the restriction operator except that only interior nodes have a corresponding nonzero line. The matrix $J^i(x) = R^iJ(x)(R^i)^T$ is the submatrix of $J(x)$ corresponding to the ith subdomain including the overlap.

We propose to perform Broyden's updates starting from the RAS preconditioner $G_0 = M_{RAS}^{-1}$. The preconditioned linear system of the Newton iterations can be written as:

$$G_kJ(x_k)\Delta x_k = -G_kF(x_k) \tag{5}$$

or

$$J(x_k)G_kG_k^{-1}\Delta x_k = -F(x_k) \tag{6}$$

depending on which side the preconditioner is applied.

The general Algorithm 2 gives an overview of the method within a time-stepper, where the update of G_k and the restarting criterion are to be defined later.

Algorithm 2. Time stepper with update of the RAS preconditioner

Require: restart parameter k_{max}, initial guess x, $k = 0$
1: **for** each time step **do**
2: // *Newton iterations:*
3: **repeat**
4: **if** $k > k_{max}$ **then**
5: $G_0 \leftarrow \sum_i(\widetilde{R}^i)^T(J^i(x_k))^{-1}R^i$ // *Local LU factorizations*
6: $k \leftarrow 0$
7: **end if**
8: solve $J(x)\Delta x = -F(x)$ preconditioned by G_k
9: $x \leftarrow x + \Delta x$
10: get G_{k+1} from G_k
11: $k \leftarrow k + 1$
12: **until** convergence
13: **end for**

- For $v_k = \Delta F_k$ the application of the preconditioner can be rewritten as:

$$G_{k+1}x = G_0x + \sum_{i=0}^{k}u_iv_i^Tx = G_0x + [u_0\cdots u_k][v_0\cdots v_k]^Tx \tag{7}$$

Hence, the additional cost of the application of G_k compared to G_0 is roughly two matrix vector products of $n \times k$ matrices. Furthermore, the computation of u_k involves one application of G_k. One should also notice that the local LU factorizations can also be computed asynchronously, continuing Newton iterations during the computation of the restarted preconditioner.

- For $v_k = G_k^T\Delta x_k$, the explicit computation of v_k should be avoided because it involves G_k^T, so M_{RAS}^{-1} which cannot be easily computed. The matrix–vector product $G_{k+1}x$ is usually rewritten as in Eq. (8).

$$G_{k+1}x = \left(\prod_{i=k}^{0}(I - u_i\Delta x_i^T)\right)G_0x \tag{8}$$

Following an idea of Martínez [15], Bergamaschi et al. proved in [7] that for G_0 and x_0 good enough initial guesses, the norm $\|I - G_kJ(x_k)\|$ can be made arbitrarily small. Since the preconditioner is also reused from one time step to another, it slowly loses its