Available online at www.sciencedirect.com
ScienceDirect
Journal of Computational Physics 219 (2006) 210–227
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

Choice of initial guess in iterative solution of series of systems arising in fluid flow simulations

Damien Tromeur-Dervout [a,b,*], Yuri Vassilevski [c]

[a] CDCSP, Center for the Development of Scientific Parallel Computing, France
[b] UMR5208-CNRS-UCBL, Institut Camille Jordan, Université Lyon 1/ISTIL, 15 Bd Latarjet, 69622 Villeurbanne Cedex, France
[c] Institute of Numerical Mathematics, 8, Gubkina Str., 119991 Moscow, Russia

Received 22 April 2005; received in revised form 16 November 2005; accepted 18 March 2006
Available online 2 May 2006

Abstract

Contemporary time stepping schemes applied to the solution of unsteady nonlinear fluid flow problems are considered. The iterative solution of arising series of linear and nonlinear systems and the choice of the initial guess are addressed. The computation of a better initial guess for two iterative linear system solvers (GCR and GMRES) is based on the history of the evolution problem solving. For implicitly discretized nonlinear evolution problems, a reduced model technique is developed for computing a better initial guess for the inexact Newton method. The computational effect of the chosen initial guess is compared with that of the standard (physically motivated) initial guess.

MSC: 65F10; 90C53; 76D05

Keywords: Krylov subspace methods; Inexact Newton methods; Unsteady Navier–Stokes equations

1. Introduction

It is well known that large systems of equations must be solved iteratively: for nonlinear systems there is no alternative and for linear systems direct methods are either too expensive or too restrictive. Any iterative technique consists of three basic procedures: the choice of the initial guess, the computation of the next iterate, and the stopping criterion. Each procedure influences the efficiency of the iterative solution in its own way: smart start, fast convergence, and prevention of oversolving. The fast convergence is conventionally achieved by a combination of an appropriate Krylov type method and a preconditioning technique [1,25,33]. The stopping criteria have been considered both for nonlinear (inexact Newton methods [16]) and linear solves [25–27].

This paper is devoted to the choice of the initial guess when a series of systems produced by time stepping schemes has to be solved. The case of linear systems with a symmetric positive definite matrix and a series of

* Corresponding author. Tel.: +33 4 72 43 13 56; fax: +33 4 72 43 11 45.
E-mail addresses: dtromeur@cdcsp.univ-lyon1.fr (D. Tromeur-Dervout), vasilevs@dodo.inm.ras.ru (Y. Vassilevski).

0021-9991/$ - see front matter
doi:10.1016/j.jcp.2006.03.014

right hand sides is considered in [5,7]. For linear systems with an unsymmetric matrix, the first method of computation of a better initial guess was suggested in [36]. In this paper, we revise this method and suggest a new choice of initial guess for series of linear systems with different unsymmetric matrices and right hand sides. For Newton type solvers of nonlinear systems generated by fully implicit discretizations, the standard and physically motivated choice of the initial guess is to adopt the solution from the previous time step. Methods of generating a better initial guess using a set of previous solutions have been investigated in [8]. We develop a different methodology based on a model reduction (see [29,30] and references therein). This technology opens a new way of generating initial guesses. The solution of a reduced model provides much better initial guess than that from the previous time step. In spite of an extra work spent on the solution of the reduced model, the total complexity of each time step decreases considerably, due to the super-linear convergence of the inexact Newton solver.

In our experiments, the series of systems are generated by time stepping methods for different formulations of unsteady Navier–Stokes equations. Depending on application, the series may have either common or different matrices (linear systems), or have different nonlinear operators (nonlinear systems). Each series illustrates the efficiency of the appropriate technique for choosing the initial guess.

The plan of this paper is as follows. In Section 2, we consider series of linear systems. The systems with the same matrix generated by the projection method for the unsteady 3D Navier–Stokes equations are examined in Section 2.1. The systems with different matrices generated by the projection method for the unsteady Low Mach number equations are considered in Section 2.2. Also we discuss a negative example showing practical limitations for the presented method. In Section 3, series of nonlinear systems are examined. After a review of a conventional fully implicit discretization and an Inexact Newton solver (Section 3.1), we describe the proper orthogonal decomposition (Section 3.2) and the reduced model (Section 3.3). In Section 3.4 the new method of fully implicit time stepping, the algorithm INB-POD, is introduced and tested numerically.

2. Initial guess for series of linear systems

The series of linear systems considered in this section is produced by projection schemes for a class of unsteady nonlinear fluid flow problems. The projection scheme applied to the Navier–Stokes equations generates a series of linear systems with the same matrix and different right hand sides. The projection scheme applied to the Low Mach number approximation of the full Navier–Stokes system produces a series of linear systems with different matrices and different right hand sides.

2.1. 3D unsteady Navier–Stokes equations

Let $\Omega \in \Re^3$ be a domain with a piece-wise smooth boundary $\partial\Omega$. The domain is occupied by a fluid with a kinematic viscosity ν and a density ρ. We denote by $\mathbf{u}(x,t)$ the velocity with components (u_1,u_2,u_3) and by $p(x,t) = P(x,t)/\rho$ the normalized pressure of the fluid. The flow of incompressible fluid with prescribed values of the velocity on $\partial\Omega$ obeys the Navier–Stokes equations:

$$\frac{\partial \mathbf{u}}{\partial t} - \nu\Delta\mathbf{u} + (\mathbf{u}\cdot\nabla)\mathbf{u} + \nabla p = \mathbf{f} \text{ in } \Omega, \quad \mathbf{u} = \mathbf{g} \quad \text{on } \partial\Omega \quad \operatorname{div}\mathbf{u} = 0 \quad \text{in } \Omega. \tag{1}$$

An important parameter for flow similarity is the Reynolds number $Re = \frac{\overline{U}\overline{D}}{\nu}$, where $\overline{U}$ and $\overline{D}$ are characteristic velocity and length, respectively.

One of the most efficient time stepping schemes for the computation of the approximate solution of (1) $\mathbf{u}^k \simeq u(t_0 + k\,\Delta t)$ is provided by the projection algorithm (pressure correction method) [18,24]:

Step 1: Predict the velocity $\hat{\mathbf{u}}^{k+1}$ by solving

$$\frac{\hat{\mathbf{u}}^{k+1} - \mathbf{u}^k}{\Delta t} - \nu\Delta\frac{\hat{\mathbf{u}}^{k+1} + \mathbf{u}^k}{2} + \left(\left(\frac{3}{2}\mathbf{u}^k - \frac{1}{2}\mathbf{u}^{k-1}\right)\cdot\nabla\right)\frac{\hat{\mathbf{u}}^{k+1} + \mathbf{u}^k}{2} = \mathbf{f}^{k+1/2} - \nabla p^k \quad \text{in } \Omega, \quad \hat{\mathbf{u}}^{k+1} = \mathbf{g} \quad \text{on } \partial\Omega; \tag{2}$$

Step 2: Project the predicted velocity onto the space of divergence free functions by solving the elliptic equation for the pressure correction δp

$$-\operatorname{div}\nabla\delta p = -\frac{1}{\Delta t}\operatorname{div}\hat{\mathbf{u}}^{k+1}, \tag{3}$$

$$\mathbf{u}^{k+1} = \hat{\mathbf{u}}^{k+1} - \Delta t\nabla\delta p, \quad p^{k+1} = p^k + \delta p. \tag{4}$$

The most expensive stage of the projection algorithm is the iterative solution of linear equation (3): the discretized operator is a stiff matrix, and the solution has to be found with a high precision. The matrix of (3) is fixed, therefore, one has to solve a series of linear systems

$$Ax = b^k \tag{5}$$

with different right hand sides b^k. The standard choice of the initial guess is the zero vector since the unknown solution x represents a pressure increment.

In this paper, we do not discuss the discretization schemes in space. We only mention that for rectangular grids with smooth variations of mesh size, one can apply finite difference schemes providing the second order of accuracy at interior grid nodes. For close-to-boundary nodes, the order of accuracy of spatial discretization may be lower, depending on the type of differential operator and boundary configuration [12]. The matrix A is the product of finite difference discretizations of the divergence and gradient operators, and is singular and stiff. Singularity is attributable to the Dirichlet boundary condition for the velocity components: the product of mesh divergence and gradient operators is an approximation of the Laplace operator with Neumann boundary condition and, therefore, is a singular matrix. Symmetry of the matrix depends on whether the divergence and gradient mesh operators are conjugate with respect to Euclidian scalar product; in our discretization they are not conjugate due to particular finite difference stencils in the close-to-boundary nodes. In order to accelerate the convergence of iterative algorithms, we adopt the fictitious domain method [23] with the fast direct solver for discrete separable operators [21,32].

For the iterative scheme we use the preconditioned generalized conjugate residual (GCR) method [33]

1. Compute $r_0 = b - Ax_0$. Set $p_0 = r_0$.
2. For $j = 0, 1, 2, \ldots$, until convergence Do:
3. $\alpha_i = \frac{(r_j, Ap_j)}{(Ap_j, Ap_j)}$
4. $x_{j+1} = x_j + \alpha_j p_j$
5. $r_{j+1} = r_j - \alpha_j\, Ap_j$
6. Compute $\beta_{ij} = -\frac{(Ar_{j+1}, Ap_i)}{(Ap_i, Ap_i)}$, for $i = 0, 1, \ldots, j$
7. $p_{j+1} = r_{j+1} + \sum_{j=0}^{j} \beta_{ij} p_i$
8. EndDo

Although GMRES method requires less vectors to be accumulated, in comparison to the GCR method, and provides similar convergence rate, the additional data available in GCR allow us to construct a very good initial guess. Indeed, besides the Krylov subspace vectors $\mathscr{K} = \{p_j\}_{j=1}^k$ which are $A^{\mathrm{T}}A$-orthogonal, we possess extra data, vectors $\{Ap_j\}_{j=1}^k$. Since the projection of b onto the space $A\mathscr{K}$ is equal to [36]

$$\sum_{j=1}^{k} \frac{(b, Ap_j)}{(Ap_j, Ap_j)} Ap_j,$$

the projection of the solution $x = A^{-1}b$ onto $\mathscr{K}$ is

$$\hat{x} = \sum_{j=1}^{k} \frac{(b, Ap_j)}{(Ap_j, Ap_j)} p_j. \tag{6}$$

This observation implies that the projection of an unknown solution onto the accumulated Krylov subspace may be easily computed (k scalar products). The accumulation of the subspaces $\mathscr{K}$ and $A\mathscr{K}$ may be continued for several right hand sides, if the initial guess for each subsequent solve is computed by (6), and x_0 is set to $\hat{x}$.

It is evident that the larger is the subspace $\mathscr{K}$, the better is the approximation $\hat{x}$ to x. The accumulation of vectors p_j and Ap_j is limited by a practical capacity of the computer memory: as soon as it is exhausted, the data p_j and Ap_j are erased and the new accumulation process begins. Another restriction for the number of accumulated vectors is imposed by the expense of computation of k scalar products. It should be smaller than the cost of the iteration stage.

We illustrate the method on the test case 3D-2Z of quasi-periodic flow described in details in [34]. The obstacle (thin cylinder) is lifted 1 mm above the plane of symmetry of the rectangular channel (Fig. 1). The unsymmetry produces quasi-periodic flows with vortex separations for $Re = 100$. The inflow and outflow Dirichlet boundary conditions simulate the Poiseuille flow, the other part of the boundary represents the no-slip condition. We consider the mesh $80 \times 72 \times 72$ and 800 time steps with $\Delta t = 0.01$. The size of system (5) is 3.5×10^5, the stopping criterion is $\|r_j\| < 10^{-5}$ which implies 10^{11}-fold reduction of the initial residual for $x_0 = 0$. The system (5) is solved by the preconditioned GCR method with the fictitious domain preconditioner [23] on 16 processors of a COMPAQ cluster of alpha ev6 processors (667 MHz). In Table 1 we show the total number of GCR iterations, n_{tot}, and the total GCR time t, for different maximal dimensions of the accumulated Krylov subspace $\max \#\mathscr{K}$. We observe that the minimal computation time is achieved for moderate (50–80) dimensions of Krylov subspaces. The reduction of the total number of iterations for higher values of $\max \#\mathscr{K}$ does not compensate the increase of the projection (6) expense. We remark that for the fully developed flow the projection with even $\max \#\mathscr{K} = 50$ provides 10^5–10^6-fold reduction of the initial residual due to the trivial initial guess. This results in 4–5 GCR iterations per time step, in contrast to 11–12 iterations for $x_0 = 0$. However, the actual speed-up is 1.5 due to the overhead of the projection computation.

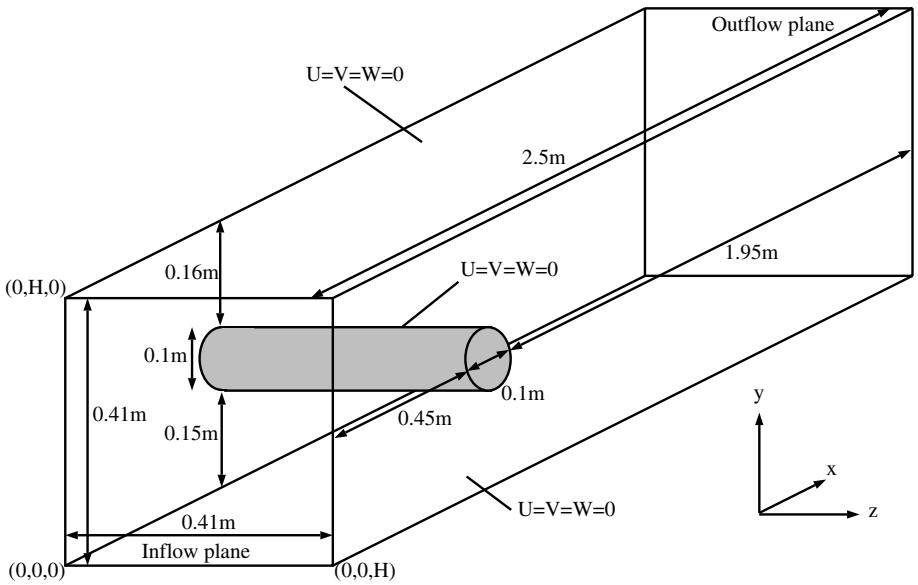

Fig. 1. Computational domain in the case of quasi-periodic flow.

Table 1
Total number of GCR iterations, the iterative solution time and maximal dimension of the accumulated Krylov subspace

$\max \#\mathscr{K}$	0	50	80	125
n_{tot}	10,385	5272	4525	4187
t (s)	1668	1108	1107	1217

2.2. Low-Mach number approximation

The non-reactive flow of compressible fluid with prescribed values of the velocity on $\partial\Omega$ obeys the full system of Navier–Stokes equations [11]. The low-Mach number approximation [17] reduces these equations to

$$\begin{aligned}
&\left(\frac{\partial \mathbf{u}}{\partial t}+(\mathbf{u}\cdot\nabla)\mathbf{u}\right)+\nabla p=\frac{T}{Re}\left(\Delta\mathbf{u}+\frac{1}{3}\nabla\nabla\cdot\mathbf{u}\right)\quad \text{in } \Omega,\quad \mathbf{u}=\mathbf{g}\quad \text{on } \partial\Omega,\\
&\left(\frac{\partial T}{\partial t}+(\mathbf{u}\cdot\nabla)T\right)=\frac{T}{RePr}\Delta T\quad \text{in } \Omega,\quad T=g_T\quad \text{on } \Gamma_D,\quad \frac{\partial T}{\partial n}=0\quad \text{on } \Gamma_N,\\
&\operatorname{div}\mathbf{u}=\frac{1}{RePr}\Delta T\quad \text{in } \Omega,
\end{aligned}\tag{7}$$

where $T(x,t)=T(x,t)^{\circ}/T(x,t)^{\circ}_{\text{inlet}}$ the normalized temperature of the fluid connected with its density by $\rho T=1$, Pr is the Prandt number. The system (7) is the simplest model in reactive flow modeling of short-contact-time chemical reactors. From the CFD perspective the chemical reactor can be modeled as a box-cylinder of very large length and small diameter with an inner surface section covered by a catalyst (platinum or rhodium). The flow should be Poiseuille flow far away from the catalytic surface. We are mainly interested in disturbance of the non-reactive flow due to high temperature gradient between the surface of the catalyst that stays in permanent regime at about 1300 K and preheated flow inlet at fixed temperature in the range 300–800 K. The Reynolds number of the flow is 750 and Prandt number is 0.72.

The pressure correction method [17] may be successfully applied to the solution of (7):

Step 1: Predict the velocity component $\hat{\mathbf{u}}_i^{k+1}$

$$\begin{aligned}
&\frac{\hat{\mathbf{u}}^{k+1}-\mathbf{u}^k}{\Delta t}-\frac{3T^k-T^{k-1}}{2Re}\left(\Delta\frac{\hat{\mathbf{u}}^{k+1}+\mathbf{u}^k}{2}\nabla\nabla\cdot\frac{\hat{\mathbf{u}}^{k+1}+\mathbf{u}^k}{6}\right)+\left(\left(\frac{3}{2}\mathbf{u}^k-\frac{1}{2}\mathbf{u}^{k-1}\right)\cdot\nabla\right)\frac{\hat{\mathbf{u}}^{k+1}+\mathbf{u}^k}{2}\\
&\quad=\mathbf{f}^{k+1/2}-\nabla p^k\quad \text{in } \Omega,\quad \hat{\mathbf{u}}^{k+1}=\mathbf{g}\quad \text{on } \partial\Omega.
\end{aligned}\tag{8}$$

Step 2: Find the pressure correction and project the predicted velocity

$$\begin{aligned}
&-\nabla\cdot(2T^k-T^{k-1})\nabla\delta p=-\frac{1}{\Delta t}\nabla\cdot\hat{\mathbf{u}}^{k+1}+\frac{1}{\Delta t Re\,Pr}\Delta(2T^k-T^{k-1}),\\
&\mathbf{u}^{k+1}=\hat{\mathbf{u}}^{k+1}-\Delta t(2T^k-T^{k-1})\nabla\delta p,\quad p^{k+1}=p^k+\delta p.
\end{aligned}\tag{9}$$

Step 3: Advance the temperature on the computed vector field

$$\begin{aligned}
&\frac{T^{k+1}-T^k}{\Delta t}-\frac{3T^k-T^{k-1}}{2RePr}\Delta\frac{T^{k+1}+T^k}{2}+\left(\frac{\mathbf{u}^{k+1}+\mathbf{u}^k}{2}\cdot\nabla\right)\frac{T^{k+1}+T^k}{2}=0\\
&\quad in\ \Omega,\quad T^{k+1}=g_T\quad \text{on } \Gamma_D,\quad \frac{\partial T^{k+1}}{\partial n}=0\quad \text{on } \Gamma_N.
\end{aligned}$$

Similarly to the projection scheme for solving (1), the most expensive step is the iterative solution of linear equation (9): the discretized operator is a stiff matrix, and the solution has to be found with a high precision. However, the matrix of (9) is not fixed since it depends on $2T^k-T^{k-1}$. This implies the solution of a series of linear systems

$$A^k x=b^k\tag{10}$$

with different matrices A^k and different right hand sides b^k. The matrix A^k is the product of the diagonal matrix $2T^k-T^{k-1}$ and finite difference discretizations of the divergence and gradient operators, and is singular and stiff. By analogy with the pressure correction method for problem (1), singularity of the matrix for (9) is attributable to Dirichlet boundary condition for the velocity components. Symmetry of the matrix depends on whether the divergence and gradient mesh operators are conjugate; in our discretization the are not conjugate due to particular finite difference stencils in the close-to-boundary nodes. The principal feature of the series $\{A^k\}$ is that the matrices (and their eigenvectors and eigenvalues) vary slowly with the growth of k since the temperature T^k changes slowly at each time step. The iterative acceleration is provided by the fictitious

domain method [23] with the fast direct solver for discrete separable operators [21,32]. By analogy with the solution of (1), the standard choice of the initial guess is the zero vector since the unknown solution x represents a pressure increment.

Since the matrices A^k are different, their common Krylov subspace cannot be generated and the advantages of the GCR method may not be used. In this case we suggest to accumulate the sequence of independent Krylov subspaces and associated images of matrices A^k. To this end, we adopt the generalized minimal residual (GMRES) method [33] for the solution of (10). The Krylov subspace $\mathscr{K}$ consists of mutually orthonormal vectors $\{v_j\}_{j=1}^m$ such that

$$AV_m = V_{m+1}H_{m+1,m},$$

where matrix V_m is composed of the vectors v_j and $H_{m+1,m}$ is an upper Hessenberg matrix containing the projection of A^k onto V_m. We correct an initial guess x_0 by the solution of the projected onto $\mathscr{K}$ system for the error e_0^k, $A^k e_0^k = A^k x_0 - b^k$:

$$\hat{x} = x_0 - V_m \hat{H}_{m,m}^{-1} \hat{G}_m V_m^{\mathrm{T}} (A^k x_0 - b^k), \tag{11}$$

where $\hat{G}_m$ is a sequence of Givens rotations such that $\hat{G}_m^{-1}$ reduces $H_{m+1,m}$ to an upper triangular matrix $\hat{H}_{m,m}$. It is Givens rotation that facilitates the solution with the upper Hessenberg matrix. We notice that all the ingredients of the projection (11) are available in the standard realization of the GMRES method [2,33].

In the framework of a series (10), the choice of a better initial guess may be stated as follows. Assume that $k-1$ systems have been solved and for the ith system, $i = k-l,\ldots,k-1$, the following data are accumulated: m_i, V_{m_i}, G_{m_i}, H_{m_i,m_i}. Then the initial guess x_0^k for the kth system is computed by the sequence of projections (11)

$$x_0^{i+1} = x_0^i - V_{m_i} \hat{H}_{m_i,m_i}^{-1} \hat{G}_{m_i} V_{m_i}^{\mathrm{T}} (A^i x_0^i - b^i), \quad i = k-l,\ldots,k-1 \tag{12}$$

with $x_0^{k-l} = 0$.

In contrast to the projection (6), the number of accumulated data l is restricted not only by the practical capacity of the computer memory, but the discrepancy between the eigenvectors and eigenvalues of A^{k-l} and A^{k-1}. In the framework of model (7), the time step $\Delta t = 0.005$ is small in comparison with the characteristic time scale, ($\sim$5), and the time variation of the temperature is small within one time step. Thus, the matrices A^{k-3}, A^{k-2}, A^{k-1} have very close eigenstructures. The chemical reactor is the tube with the rectangular cross section. The X-dimension of the tube is 20, Y dimension is 8, and Z-dimension is 1. The strip of the catalyst of width 1 is located at distance 1 from the inlet and is heated at temperature as much as four times higher than the temperature at the inlet. On the other part of the boundary we pose Neumann boundary conditions for the temperature. The flow at the inlet and outlet is Poiseuille flow with the Reynolds number 750 and Prandt number 0.72. The other part of the boundary has the no-slip condition. We consider a rectangular mesh $80 \times 64 \times 40$. The number of degrees of freedom approaches 10^6 and the number of unknowns in systems (10) is $\sim 2 \times 10^5$. The solution stabilizing to the steady-state ($\mathbf{f} = 0$, $\mathbf{u}^0 = 0$) is simulated within 500 time steps, $\Delta t = 0.005$. The stopping criterion for the GMRES iterations is the reduction of the residual norm to 10^{-3} which implies 10^6-fold relative reduction. The preconditioner is the fictitious domain method [23]. The system (5) is solved on 16 processors of a COMPAQ cluster of alpha ev6 processors (667 MHz). In Table 2 we present the total number of GMRES iterations, n_{tot}, the total number of projections (11), n_{proj}, at time steps 101, …, 500, compared to the case of $x_0^k = 0$. The projection uses three previous Krylov subspaces ($l = 3$). We notice that the projection (11) with $l = 3$ reduces the norm of the initial residual (due to the trivial initial guess) by four orders of magnitude. However, the remaining two orders of magnitude for the residual norm are dumped within 5–7 GMRES iterations, in contrast to the case $x_0^k = 0$, yielding 10^6-fold relative reduction for 11–14 GMRES iterations. The slow down of the convergence rate and the cost of the projection procedure

Table 2
Total number of GMRES iterations and the iterative solution time, with and without projected initial guess, at time steps 101,…,500

	n_{tot}	n_{proj}	t (s)
No projection	5012	0	357
Projection $l = 3$	3208	1044	309

Table 3
Number of GMRES iterations and residual norms for the model 1D diffusion equation

k	1	4	7	10	4	7	10
	$x_0^k = 0$				x_0^k by (11)		
$\|A^k x_0^k - b^k\|$	31	31	31	31	0.5	0.2	0.7
n_{it}	12	15	15	12	12	11	9

(projection onto a Krylov subspace costs one GMRES iteration) reduce the computational gain to 15% of the computational work ($n_{\text{tot}} + n_{\text{proj}}$) and 15% of the solution time.

If the matrices in the series (10) are not "close" to each other, the projection (11) may even increase the initial residual, in comparison with the trivial initial guess. For example, our numerical evidence shows that the projection (11) is useless for the series of systems produced by the Newton method (cf. Table 4). On the other hand, if the matrices are not sufficiently close to each other, then the reduction of the initial residual may not compensate the expenses of the projection (11) requiring the computation of the current residual. Consider, for example, the model 1D diffusion equation on the interval $[0,1]$

$$-(T_k u')' = 1, \quad u(0) = u(1) = 0, \quad T_k(x) = 2 + \sin \pi(x + 0.1k), \quad k = 1, \ldots, 10.$$

We discretize the equation with the finite differences on the uniform mesh, $h = 10^{-3}$. To provide an affordable convergence rate, we choose the preconditioner as the finite differences approximation of the second derivative with Dirichlet boundary conditions. The stopping criterion for the GMRES iterations is the reduction of the residual norm to 10^{-6}. In Table 3, we present the number of GMRES iterations and the effectiveness of the projection (11) for $l = k - 1$: the reduction of iteration count (3–4) does not compensate the cost of the projection ($3, 6, 9$ matrix vector multiplications for $k = 4, 7, 10$).

We conclude this section with the remark that the projection (11) may save certain amount of computations, if the eigendata of matrices of systems (10) are relatively close to each other, and the number of GMRES iterations and the computer memory allow to accumulate the projection data. In this respect, we mention a Ritz's value based strategy [14,31] for the choice of appropriate vectors from a series of Krylov subspaces.

3. Initial guess for series of nonlinear systems

In this section the series of nonlinear systems is generated by fully implicit discretizations of unsteady nonlinear problems.

3.1. Fully implicit solvers

Let $L(u)$ be a nonlinear discrete operator representing a spatial approximation of a parabolic boundary value problem. The simplest robust technique for time approximation of unsteady problems is the backward Euler time stepping:

$$\frac{u^i - u^{i-1}}{\Delta t} + L(u^i) = g^i. \tag{13}$$

The main advantage of the method is its unconditional stability. Being the first order scheme (in time), it may be generalized to higher order approximations (e.g., backward differences formulae). In general, the ith time step of a fully implicit scheme may be represented by the nonlinear system

$$F^i(u^i) = 0, \tag{14}$$

where F^i contains all the problem data and previous solutions. For instance, in the case of scheme (13),

$$F^i = u^i + L(u^i)\Delta t - u^{i-1} - g^i \Delta t.$$

The price to be paid for the robustness of the method is its arithmetical complexity: at each time step, a nonlinear system has to be solved. In last decade, several robust nonlinear solvers have been proposed, analyzed, and implemented [4,15,28]. However, the efficient solution of large nonlinear systems remains a

challenge. One way to obtain efficient nonlinear solvers is to use nonlinear multigrid [6] or Jacobian free Newton–Krylov methodology (see [19] and references therein) where the solution of the linearized Newton step problem is solved by a Krylov method. The main effort is then to provide a good preconditioner for the Krylov method. Our choice of a good initial guess to the Newton method can be complementary to the choice of efficient preconditioner for the Newton–Krylov method. In this paper we address a particular Newton–Krylov solver, the inexact Newton backtracking method, although this is not mandatory.

Consider a nonlinear system

$$F(u) = 0.$$

The inexact Newton backtracking [9,10,28] method offers global convergence properties combined with potentially fast local convergence.

ALGORITHM INB. LET u_0, $\eta_{\max} \in [0,1)$ AND $0 < \theta_{\min} < \theta_{\max} < 1$ BE GIVEN.
FOR $k = 0, 1, \ldots$ (UNTIL CONVERGENCE) DO:
CHOOSE INITIAL $\eta_k \in [0, \eta_{\max}]$ AND s_k SUCH THAT

$$\|F(u_k) + F'(u_k)s_k\| \leqslant \eta_k \|F(u_k)\|, \tag{15}$$

$$\eta_k = \begin{cases} \min\left\{\eta_{\max}, \frac{|\|F(u_k)\| - \|F(u_{k-1}) + F'(u_{k-1})s_{k-1}\|\|}{\|F(u_{k-1})\|}\right\}, & k > 0, \\ 0.5, & k = 0. \end{cases} \tag{16}$$

WHILE $\|F(u_k + s_k)\| > [1 - t(1 - \eta_k)]\|F(u_k)\|$ DO:
CHOOSE $\theta \in [\theta_{\min}, \theta_{\max}]$.
UPDATE $s_k \leftarrow \theta s_k$ AND $\eta_k \leftarrow 1 - \theta(1 - \eta_k)$.
SET

$$u_{k+1} = u_k + s_k. \tag{17}$$

The algorithm has a solid theoretical background [9,10]:

Theorem 1. *Let F be continuously differentiable. If $\{u_k\}$ produced by the algorithm INB has a limit point u such that $F'(u)$ is invertible, then $F(u) = 0$ and $u_k \to u$. Furthermore, if F' is Lipschitz continuous at u, then*

$$\|u_{k+1} - u\| \leqslant \beta \|u_k - u\| \|u_{k-1} - u\|, \quad k = 1, 2, \ldots$$

for a constant β independent of k.

Inequality (15) implies the approximate iterative solution of the system

$$F'(u_k)s_k = -F(u_k) \tag{18}$$

with relative reduction of the residual (for trivial initial guess) η_k. The forcing term η_k (16) is chosen dynamically so that to avoid oversolving the systems (18). Backtracking is used to globalize the convergence, and updated s_k and η_k always satisfy (15). The iterative solution of (18) requires only evaluation of $F'(u_k)$ on a vector. This allows to replace $F'(u_k)v$ by its finite difference approximation, e.g.,

$$F'(u_k)v = \frac{1}{\delta}[F(u_k + \delta v) - F(u_k)]. \tag{19}$$

Hereinafter, the GMRES method restarted after each 30 iterates (GMRES(30)) method is used for the iterative solution of (18).

We recall that the arithmetical complexity of the method is expressed as the total number of function evaluations n_{evF} and the total number of preconditioner evaluations n_{evP} (if any); the remaining overheads are negligible.

The algorithm INB presumes the choice of initial guesses of two types: u_k for nonlinear iterations, and $u_{k,0}$ for linear iterations.

As it was mentioned in Section 2.2, the method (11) of computation of the initial guess for the sequence of systems (18) is ineffective. More precisely, the total numbers of function and preconditioner evaluations, n_{evF}, n_{evP} do not decrease in comparison with the trivial initial guess. To illustrate the assertion, we consider the

Table 4
Complexity of the algorithm INB (NITSOL) for different numbers of Krylov subspaces used in the projection

l in (12)	0	1	2
n_{lit}	306	290	363
n_{evF}	344	350	442
n_{evP}	325	329	420

steady counterpart of the equation (26) with $v \equiv 1$ discretized on a square mesh with $h = 2^{-6}$. The nonlinear system with 3969 unknowns is solved by the algorithm INB implemented in the NITSOL package [28] with GMRES(30) iterations (for other details we refer to Section 3.4). The stopping criterion for the INB algorithm is $\|F(u_k)\| < 10^{-7}\|F(0)\|$. In Table 4 we show the total number of linear iterations n_{lit}, n_{evF} and n_{evP} for different strategies of choosing the initial guess for GMRES iterations: the trivial vector corresponding to $l = 0$ in (12), or $l \neq 0$ in (12) corresponding to l consequent projections on the l newest Krylov subspaces. The data of Table 4 indicate that the complexity of the INB algorithm increases with l growing. The inefficiency of the strategy (12) may be explained by essential differences in spectral properties of subsequent Jacobian matrices. In contrast, an appropriate choice of the initial guess for nonlinear iterations reduces considerably the arithmetical complexity of the method. The next subsections explain this issue in detail.

3.2. Proper orthogonal decomposition

Proper orthogonal decomposition (POD) provides a way to find optimal lower dimensional approximations of the given series of data. More precisely, it produces an orthonormal basis for representing the data series in a certain least squares optimal sense [29,30]. Combined with the Galerkin projection, the POD is a tool for generation of reduced models of lower dimension. The reduced models may give a better initial guess for the Newton solution at the next time step.

The POD provides the definite answer to the question: What m-dimensional subspace $S \subset R^N$ is the most close (in the least squares sense) to the given set of n R^N-vectors $\{u^i\}_{i=1}^n$,

$$S = \arg \min_{S \in R^{N \times m}} \sum_{i=1}^{n} \|u^i - P_S u^i\|^2 ?$$

Here P_S is the orthogonal projection onto S. Define the correlation matrix $R = XX^{\mathrm{T}}$, $X = \{u^1 \ldots u^n\}$, and find m eigenvectors of the problem

$$Rw_j = \lambda_j w_j, \quad \lambda_1 \geqslant \cdots \geqslant \lambda_N \geqslant 0$$

corresponding to m largest eigenvalues $\lambda_1 \geqslant \cdots \geqslant \lambda_m$. Then

$$S = \operatorname{span}\{w_j\}_{j=1}^{m} \tag{20}$$

and

$$\sum_{i=1}^{n} \|u^i - P_S u^i\|^2 = \sum_{j=m+1}^{N} \lambda_j. \tag{21}$$

The computational cost of finding m largest eigenvalues of symmetric matrix R is modest. Indeed, our experience shows that for $m \sim 10$, the application of the Arnoldi process [22] requires a few tens of R-matrix–vector multiplications in order to retrieve the desirable vectors with very high accuracy (cf. Table 9). In spite of large dimension N and density of R, the matrix–vector multiplication is easy to evaluate, due to the factored representation $R = XX^{\mathrm{T}}$. The arithmetical cost of the evaluation Xa (and $X^{\mathrm{T}}b$) is Nn multiplications and not more than Nn additions, therefore, R-matrix–vector multiplication costs at most $4Nn$ flops.

3.3. Generation and solution of the reduced model

Each time step of the scheme (13) generates the equation (14) which we call the original model. A reduced model is generated on the basis of POD for a sequence of solutions $\{u^i\}_{i=i_b}^{i_e}$, $i_e - i_b + 1 = n$ at n time steps. The

eigenvectors $\{w_j\}_{j=1}^m$ may be considered as the basis of m-dimensional subspace with projector $V_m = \{w_1 \ldots w_m\} \in R^{N\times m}$. The reduced model is the Galerkin projection onto this subspace:

$$V_m^{\mathrm{T}} F^i(V_m \hat{u}^i) = 0, \tag{22}$$

or, equivalently,

$$\hat{F}^i(\hat{u}^i) = 0, \tag{23}$$

where the unknown vector $\hat{u}^i \in R^m$ and $\hat{F}^i : R^m \to R^m$.

The reduced model is the nonlinear equation of very low dimension m. For its solution, we adopt the same INB algorithm with the finite difference approximation of the Jacobian-vector multiplication. Being the formal Galerkin projection, each evaluation of function $\hat{F}^i(\hat{u}_k^i)$ in the kth INB iterate is the sequence of the following operations: $u_k^i = V_m \hat{u}_k^i$, $f_k^i = F^i(u_k^i)$, $\hat{f}_k^i = V_m^{\mathrm{T}} f_k^i$. Therefore, the overhead is matrix–vector multiplications for V_m and V_m^{T}, i.e., $4Nm$ flops. We notice that usually $m \sim 10$ and the evaluation of function $F(u)$ is much more expensive than $4Nm$. This implies a negligible contribution of the overhead.

Another important consequence of low dimensionality of (23) is that the INB algorithm may be applied without any preconditioner. If the reduced Jacobian $(\hat{F}^i(\hat{u}_k^i))'$ were known explicitly, the GMRES iterations would converge within at most m GMRES iterations due to m-dimensionality of the reduced model. Since $\delta \ll 1$ in (19), $\frac{1}{\delta}[\hat{F}^i(\hat{u}_k^i + \delta\hat{v}) - \hat{F}^i(\hat{u}_k^i)]$ is an approximation of the matrix–vector multiplication $(\hat{F}^i(\hat{u}_k^i))'v$ with a relative error $O(\delta)$, so that m GMRES iterations provide at most $O(\delta)$ accuracy. The assumption $\delta \ll \epsilon$ (the stopping tolerance for the INB algorithm) implies convergence within m iterations for each system (18).

3.4. Fully implicit solver with POD-reduced model acceleration

Coupling POD and Galerkin projection for the generation of the reduced model gives a powerful tool for acceleration of the fully implicit schemes. Let n, the length of data series, be defined, as well as the desirable accuracy ϵ for F^i: $\|F^i(u^i)\| \leqslant \epsilon$. For any time step $i = 1, \ldots$ perform:

ALGORITHM INB-POD

IF $i \leqslant n$, SOLVE $F^i(u^i) = 0$ BY PRECONDITIONED INB WITH THE INITIAL GUESS $u_0^i = u^{i-1}$ AND ACCURACY ϵ

ELSE

1. IF($mod(i,n) = 1$):
   (I) FORM $X = \{u^{i-n} \ldots u^{i-1}\}$;
   (II) FIND SO MANY LARGEST EIGENVECTORS w_j OF $R = XX^{\mathrm{T}}$ THAT $\sum_{j=m+1}^{N} \lambda_j \leqslant \epsilon$;
   (III) FORM $V_m = \{w_1 \ldots w_m\}$
2. SET $\hat{u}_0^i = V_m^{\mathrm{T}} u^{i-1}$
3. SOLVE $\hat{F}^i(\hat{u}^i) = 0$ BY NON-PRECONDITIONED INB WITH THE INITIAL GUESS $\hat{u}_0^i$ AND ACCURACY $\epsilon/10$
4. SET $u_0^i = V_m \hat{u}^i$
5. SOLVE $F^i(u^i) = 0$ BY PRECONDITIONED INB WITH THE INITIAL GUESS u_0^i AND ACCURACY ϵ

Several remarks are in order. The absence of the preconditioner for the reduced model is dictated by two reasons: (a) it is not clear how to construct a preconditioner for the reduced model, (b) it is hardly needed if m is small. The reduced model is slightly oversolved: this provides better initial guess u_0^i. The number of eigenvectors is chosen adaptively in the above algorithm: it allows to form a reduced model that approximates the original model with the desirable accuracy ϵ. Actually, this condition my be replaced by a more rough condition $N\lambda_{m+1} < \epsilon$ or even by a fixed number m, $m = 10$–40. The solution of the eigenvalue problem may be performed asynchronously with the implicit solution: as soon as V_m is formed, the reduced model becomes the active substep. Moreover, the POD and the implicit INB solution may be performed on different computers: the data to be exchanged are X and V only, and the communication delay does not block the simulation. The underlying client-server architecture of the method makes it very appealing in grid computing applications [35]. Another appealing feature of the method is its modularity: computation of the reduced model solution is separated from the original solver and is based on its simple algebraic modification. It makes the INB-POD algorithm easy to implement in codes with a complex architecture.

We note that our approach differs from the mesh sequencing method [20] where an initial guess for the Newton method is searched on a series of coarser grids. The latter technique is based on explicitly known inter-grid transfer operators and coarser grids. Therefore, it inherits the restrictions typical for the multigrid methods. The POD approach does not depend on the geometry and discretization parameters, and uses only the solution series generated on the fine grid. Hence, it is less restrictive although it requires extra memory for the storage of the basis V_m.

In order to illustrate the basic features of the proposed methodology, we choose the backward Euler approximation of the unsteady 2D Navier–Stokes equations. We consider the classical driven cavity problem in the streamfunction-vorticity formulation [11]:

$$\frac{\partial \omega}{\partial t} - \frac{1}{Re}\Delta\omega + (\psi_y\omega_x - \psi_x\omega_y) = 0 \quad \text{in } \Omega, \tag{24}$$

$$-\Delta\psi = \omega \quad \text{in } \Omega, \tag{25}$$

$$\psi|_{t=0} = 0 \quad \text{in } \Omega,$$

$$\psi = 0 \quad \text{on } \partial\Omega,$$

$$\left.\frac{\partial\psi}{\partial n}\right|_{\partial\Omega} = \begin{cases} v(t) & \text{if } y = 1, \\ 0 & \text{if } 0 \leqslant y < 1. \end{cases}$$

Here, $\Omega = (0,1)^2$, $Re = 1000$, and $v(t)$ is the unsteady boundary condition leading the flow. Eliminating the vorticity, we obtain the streamfunction formulation:

$$\begin{aligned} &\frac{\partial}{\partial t}(\Delta\psi) - \frac{1}{Re}\Delta^2\psi + (\psi_y(\Delta\psi)_x - \psi_x(\Delta\psi)_y) = 0 \quad \text{in } \Omega, \\ &\psi|_{t=0} = 0 \quad \text{in } \Omega, \\ &\psi = 0 \quad \text{on } \partial\Omega, \\ &\left.\frac{\partial\psi}{\partial n}\right|_{\partial\Omega} = \begin{cases} v(t) & \text{if } y = 1, \\ 0 & \text{if } 0 \leqslant y < 1. \end{cases} \end{aligned} \tag{26}$$

Three different types of flows are simulated: converging to the steady solution (stabilizing), quasi-periodic in time, and quasi-periodic in time with variable periods (arrhythmic). These three cases are defined by the unsteady boundary velocity $v(t)$. We set $v(t) = 1 + (t+10)^{-1}$ for the stabilizing flow, $v(t) = 1 + 0.2\sin(t/10)$ for the quasi-periodic flow, and $v(t) = 1 + 0.2\sin([1 + 0.2\sin(t/5)]t/10)$ for the arrhythmic flow, see Fig. 2. We motivate the chosen parameters as follows. In the case of $v(t) = 1$, the unsteady solution is stabilized within $t_s \sim 150$. Therefore, to get a quasi-periodic solution, we need the periodic forcing term with the period $T < t_s$ but comparable with t_s, $T \sim t_s$. Indeed, if $T \ll t_s$, the inertia of the dynamic system will smear out the amplitude of the oscillations; if $T > t_s$, the dynamic system will have enough time to adapt to the periodic forcing term and to demonstrate periodic behavior. The function $\sin(t/10)$ has the period $T = 20\pi$ which fits perfectly the above restrictions for the quasi-periodicity. The arrhythmic flow is a simple modification of the quasi-periodic flow by arrhythmic scaling of the time $t \to [1 + 0.2\sin(t/5)]t$. It is well known that the feasible time step Δt for approximation of periodic solutions satisfies $12\Delta t = T$, i.e., $\Delta\ t \sim 5$. Therefore, in the case of $v(t) = 1$, the stabilization will occur within 30 time steps. This is not enough for the demonstration of the POD-acceleration. Therefore, we artificially extend the stabilization time by choosing the unsteady stabilizing boundary condition $v(t) = 1 + (t+10)^{-1}$.

For the discretization in time, we chose the backward Euler scheme (13). For the discretization in space (developed by P. Brown), we adopt the P_1 finite element spaces applicable to the biharmonic problems as it was shown in [13]. This discretization is also equivalent to that obtained with standard finite differences. We consider two uniform grids with mesh steps $h = 2^{-7}, 2^{-8}$ providing 16,129 and 65,025 unknowns, respectively. Typical quasi-periodic solution isolines are shown in Fig. 3. The nonlinear problems (14) are solved by the NITSOL package [28] with the preconditioned GMRES(30) iterations. The stopping criterion for the INB algorithm is $\|F^i(u_k^i)\| < 10^{-7}\|F^0(0)\|$. For the preconditioner P, we use the discretized version of $\frac{1}{Re}\Delta^2$. For a uniform mesh, systems of the form $Pw = u$ can be solved very efficiently using a fast solver [3]. For small and moderate Reynolds numbers, this preconditioner provides independence of the convergence rate of the mesh

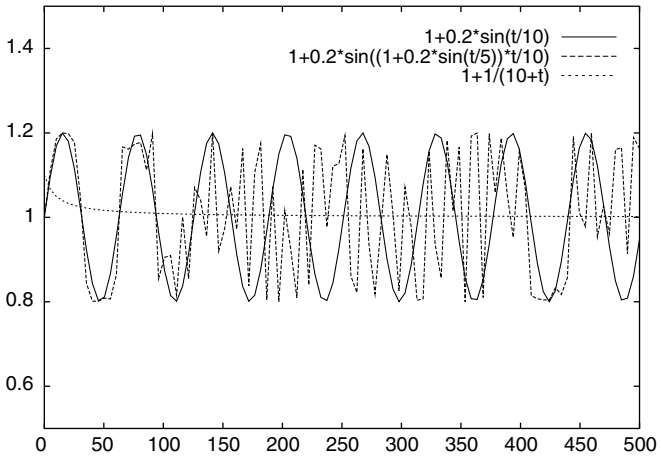

Fig. 2. Different types of boundary velocity $v(t)$.

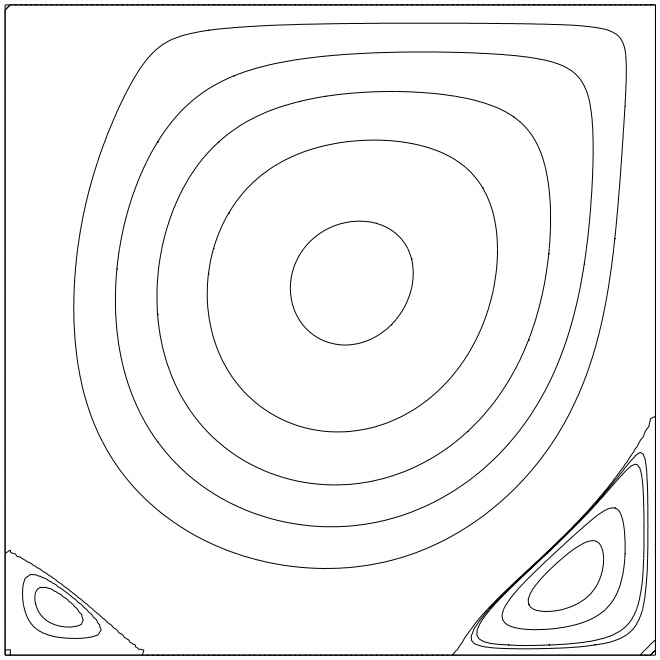

Fig. 3. Streamfunction isolines $\psi = -0.1, -0.08, -0.06, -0.04, -0.02, 0, 0.00005, 0.0001, 0.0005, 0.001, 0.0025$ for quasi-periodic solution at time $t = 500$, $h = 2^{-7}$.

size. Evaluations of the Jacobian are replaced by the finite difference approximations of the first order (19). Therefore, the dominant contribution to the arithmetical work is given by the number of function evaluations n_{evF} and the number of preconditioner evaluations n_{evP} for each time step.

The time measurements presented below have been performed on a COMPAQ alpha ev6 processor running at 667 MHz.

In Table 5, we present the performance of the standard NITSOL solver with the initial guess equal to the solution at the previous time step $u_0^i = u^{i-1}$. We consider the three types of the unsteady solution at several time steps and on two meshes. The numbers of function and preconditioner evaluations do not depend on the mesh size. Consequently, the CPU time (per time step) increases by factor 5 when the number of unknowns is multiplied by factor 4. Slight disproportionality is attributable to the arithmetical complexity of the preconditioner evaluation. However, both n_{evF} and n_{evP} may depend on the time step: for stabilizing solution, they

Table 5
Performance of the algorithm INB (NITSOL) for $u_0^i = u^{i-1}$

Mesh step	$h = 2^{-7}$			$h = 2^{-8}$		
Time step, i	10	20	30	10	20	30
Stabilizing solution						
$\|F^i(u_0^i)\|$	0.005	0.0014	0.0006	0.015	0.004	0.0018
n_{evF}	127	107	96	118	93	83
n_{evP}	122	102	91	113	88	78
CPU time	2	1.73	1.54	9.7	7.3	7
Quasi-periodic solution						
$\|F^i(u_0^i)\|$	0.13	0.28	0.03	0.36	0.79	0.09
n_{evF}	150	191	172	166	186	189
n_{evP}	144	185	167	160	180	183
CPU time	2.4	3.1	2.9	13.4	15.3	16.1
Arrhythmic solution						
$\|F^i(u_0^i)\|$	0.005	0.77	0.46	0.01	2.2	1.3
n_{evF}	122	215	201	115	205	227
n_{evP}	117	208	195	110	198	221
CPU time	2.0	3.5	3.3	9.2	16.7	19.5

decrease monotonically, whereas for the other cases they may oscillate within certain range. The reason is evident: in the first case, the choice $u_0^i = u^{i-1}$ provides better initial guess ($\|F^i(u_0^i)\|$) as $t \to \infty$, while in the other cases the quality of the initial guess $u_0^i = u^{i-1}$ depends on the time moment t, see Fig. 2.

Now we examine the speed-up of computation by POD and its basic features. To this end, we consider the performance of the algorithm INB-POD with the following parameters: the data (solutions) series are $\{u^{20k-10} \dots u^{20k+9}\}$, $k = 1, 2, \dots$, i.e., $n = 20$, and the dimension of the reduced model is fixed to $m = 10$. In Tables 6 and 7 we present the arithmetical complexity of certain time steps, in terms of n_{evF}, n_{evP} and the

Table 6
Performance of the algorithm INB-POD (NITSOL) for the stabilizing solution

Mesh step	$h = 2^{-7}$			$h = 2^{-8}$		
Time step, i	32	42	52	32	42	52
$\|F^i(u_0^i)\|, \times 10^{-7}$	4.4	15	7	14	37	12
n_{evF}	25 + 12	25 + 30	24 + 19	27 + 14	27 + 27	25 + 12
n_{evP}	0 + 10	0 + 27	0 + 16	0 + 12	0 + 24	0 + 10
CPU time	0.1 + 0.16	0.1 + 0.4	0.1 + 0.3	0.7 + 0.9	0.7 + 1.8	0.6 + 0.7

Table 7
Performance of the algorithm INB-POD (NITSOL) for the non-stabilizing solutions

Mesh step	$h = 2^{-7}$			$h = 2^{-8}$		
Time step, i	32	52	72	32	52	72
Quasi-periodic solution						
$\|F^i(u_0^i)\|, \times 10^{-6}$	25	0.9	0.5	22	1	2.6
n_{evF}	54 + 74	45 + 22	44 + 11	44 + 55	45 + 11	44 + 19
n_{evP}	0 + 69	0 + 19	0 + 9	0 + 51	0 + 9	0 + 16
CPU time	0.2 + 1.1	0.2 + 0.3	0.2 + 0.15	1.2 + 4.2	1.1 + 1.1	1.1 + 1.2
Arrhythmic solution						
$\|F^i(u_0^i)\|, \times 10^{-4}$	4	3.6	1.7	5.5	4.3	1.8
n_{evF}	50 + 110	51 + 96	49 + 94	50 + 98	42 + 80	49 + 77
n_{evP}	0 + 105	0 + 91	0 + 89	0 + 93	0 + 75	0 + 72
CPU time	0.2 + 1.7	0.2 + 1.5	0.2 + 1.5	1.3 + 7.7	1.0 + 6.4	1.1 + 5.9

CPU time, as well as the quality of the initial guess $\|F^i(u_0^i)\|$ due to the reduced model, $u_0^i = V_m\hat{u}^i$. The first entry of each sum in the Tables corresponds to the contribution of the reduced model, the second is due to the original model. The first observation is that the acceleration is significant, ~2–5-fold in comparison with the standard algorithm INB. The reason is in much better initial guess for the original model solver (cf. $\|F^i(u_0^i)\|$). Due to the super-linear convergence of the INB algorithm, this results in smaller values of n_{evF}, n_{evP}. The price to be paid for this reduction is the cost of the reduced problem solution. As it was mentioned before, the complexity of function $\hat{F}$ evaluation only slightly exceeds that for F, whereas the number of preconditioner evaluations is zero for the reduced model. Since in the considered application (as well as in the absolute majority of applications), the complexity of the preconditioner evaluation dominates over the complexity of the function evaluation, the speed-up is attributable to the ratio of n_{evP} for the standard algorithm and the accelerated one. As it is seen from the tables, this ratio depends on the type of unsteady solution and on the time moment. For the stabilizing and quasi-periodic solutions, the reduced model is capable to recover the solution at the next time step very accurately ($\|F^i(u_0^i)\| \sim 10^{-6}$) which provides the essential reduction of n_{evP} (and n_{evF}). However, at the time step 32, the quasi-periodic flow is not yet very well stabilized, and the prediction of the reduced model is not as good ($\|F^i(u_0^i)\| \sim 10^{-5}$) yielding only 2-fold acceleration. For the arrhythmic flow, the quality of the prediction is even worse ($\|F^i(u_0^i)\| \sim 10^{-4}$), and the typical acceleration factor is 2-3. At early stages of the arrhythmic flow, the acceleration may be small, as it is observed for the time step 32. Here, the solution from the previous time step occasionally turns out to be almost as good as the initial guess predicted by the reduced model, ~0.01 versus ~0.001. As a result, the reduction of n_{evP} is only 10-20%, and the overall acceleration is insignificant. On the other hand, the complexity of the reduced model solution does not vary considerably: n_{evF} ranges from 25 to 50 for all the cases. The reason is low dimensionality ($m = 10$) of the reduced problem.

In order to present the overall picture of the POD acceleration, we demonstrate in Fig. 4 the elapsed time of the INB and INB-POD computation of the three flows on the two meshes for all the time steps. The POD acceleration starts at the 31th time step. The eigenbasis is computed in parallel on other processors of the COMPAQ cluster every 20th time step starting from the 30th, and the time of the computation is not shown. The elapsed time mean values are computed for the time steps $31, \ldots, 100$. The quasi-constant reduced model computational time (Time POD) illustrates the effect of the low dimensionality of the reduced model. We remark that for the saturated case the POD basis provides very efficient acceleration for the time steps next to V_m update and that the acceleration becomes less efficient for the obsolete basis V_m (time steps 50, 70, 90). However, for the quasi-periodic and arrhythmic cases the INB-POD elapsed time does not systematically increase until the next V_m update.

We remark that the arrhythmic flow is a very tough test for the methodology: the solution at the next time step hardly can be approximated by a composition of several solutions at previous time steps. Consequently, the POD-reduced model is not capable to provide a very good initial guess and many-fold speed-ups. May the extension of data (solution) series, n, and/or enrichment of the reduced model basis, m, increase the speed-up? Controversial tendencies do not allow to answer the question a priori: the larger m is, the better is the prediction of the reduced model, but its quality is limited by the accuracy of the representation of u^{i+1} via the series $\{u^{i_b} \ldots u^{i_e}\}$; on the other hand, the complexity of the reduced model solution increases for large m. Therefore, the answer depends on the particular problem and parameters and must be based on a numerical evidence. For the case of the arrhythmic solution on the mesh $h = 2^{-8}$, we present in Table 8 the complexity of certain time steps for several pairs m, n. The 2-fold increase of m does provide a better initial guess but the increased cost of the reduced model solution is not compensated by the speed-up of the original model solution. On the other hand, the 2-fold increase of n does not provide a better resolution of the reduced model: due to irregularity (in time) of the unsteady solution, the extension of data series may not provide a better approximation. Moreover, the extension of data series may slightly deteriorate the performance of the algorithm INB-POD. The reason is that the more "useless" solutions contribute to the basis of the reduced model, the less adequate is the reduced model (with fixed m). We notice that for the unsteady problems with "predictable" solutions (stabilizing or quasi-periodic) the positive effects of the increase of n may be more pronounced, with the exception that for the quasi-periodic solution the choice $n > T/\Delta t$ seems to be not feasible. The increase of m is always constrained by the compromise between the cost of the solution of the reduced model and the actual speed-up of the original model.

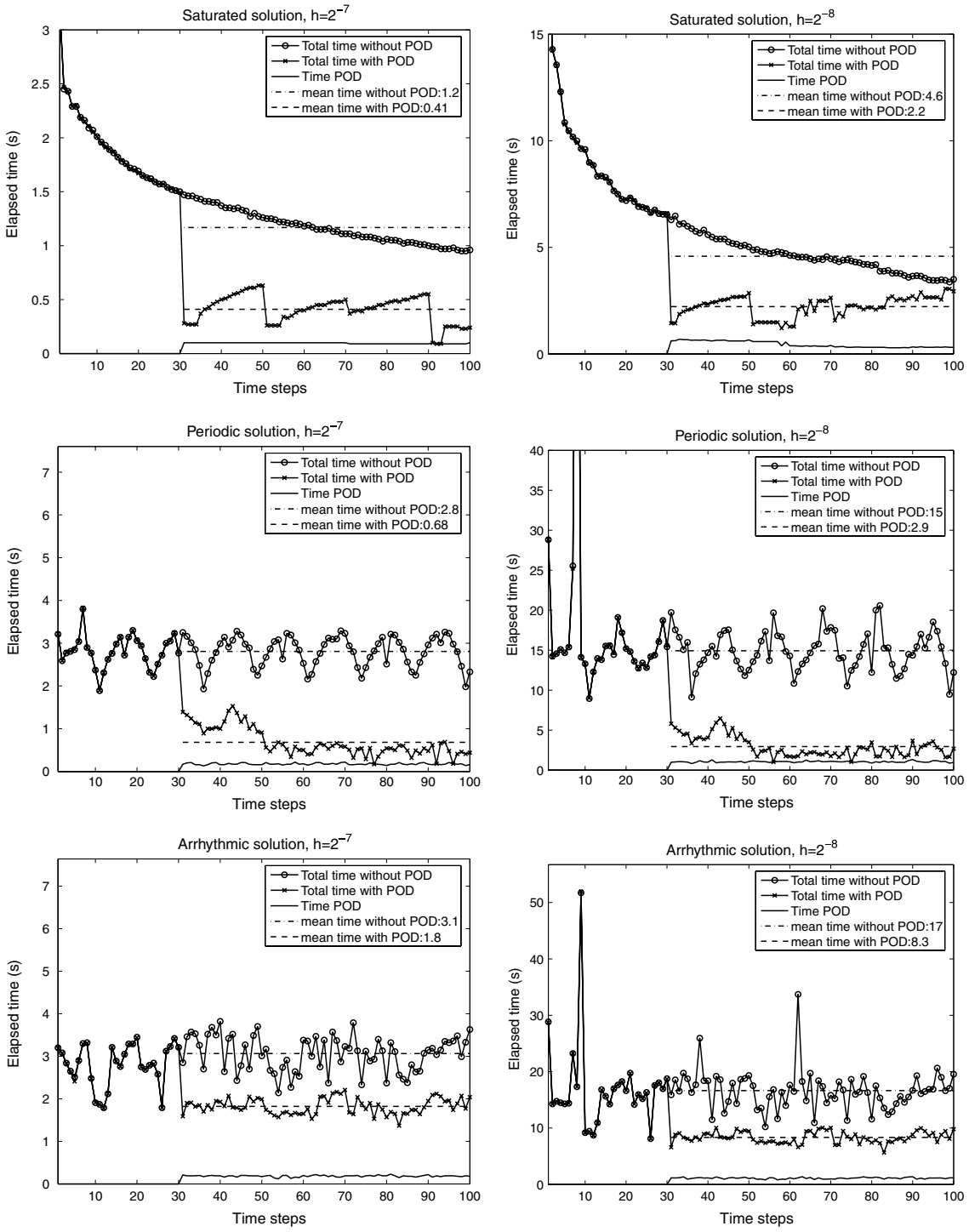

Fig. 4. Elapsed time with and without the POD acceleration and computational time of the reduced model (time POD), on different time steps with mesh steps $h = 2^{-7}$ and $h = 2^{-8}$. The POD acceleration starts at the time step 31; the mean values of elapsed time are computed for the time steps $31, \ldots, 100$.

Finally, we examine the performance of the Arnoldi process with 50 matrix multiplications used in the computation of m largest eigenvalues of matrix R. We set $m = 10$, $n = 20$, and consider the case of quasi-periodic solution generating the sequence of vectors $u^{13}, \ldots, u^{32}$. In Table 9 the computed Ritz values corresponding to

Table 8
Performance of the algorithm INB-POD (NITSOL) for the arrhythmic solution and different m,n

Time step, i	32	52	72	112	152
$m = 10$, $n = 20$, $X = \{u^{20k-10} \dots u^{20k+9}\}$					
$\|F^i(u_0^i)\|$, $\times 10^{-4}$	5.5	4.3	1.8	6	1.4
n_{evF}	50 + 98	42 + 80	49 + 77	48 + 101	51 + 78
n_{evP}	0 + 93	0 + 75	0 + 72	0 + 96	0 + 73
CPU time	1.3 + 7.7	1.0 + 6.4	1.1 + 5.9	1.1 + 8.0	1.2 + 6.0
$m = 20$, $n = 20$, $X = \{u^{20k-10} \dots u^{20k+9}\}$					
$\|F^i(u_0^i)\|$, $\times 10^{-4}$	0.5	0.4	2.0	0.4	0.4
n_{evF}	101 + 67	96 + 61	99 + 80	107 + 69	86 + 58
n_{evP}	0 + 63	0 + 57	0 + 75	0 + 65	0 + 54
CPU time	4.0 + 5.2	4.2 + 4.8	4.2 + 6.3	4.7 + 5.3	3.5 + 4.5
$m = 10$, $n = 40$, $X = \{u^{40k-30} \dots u^{40k+9}\}$					
$\|F^i(u_0^i)\|$, $\times 10^{-4}$		3.9	5.5	6.0	3.6
n_{evF}		42 + 86	52 + 93	58 + 97	45 + 90
n_{evP}		0 + 81	0 + 88	0 + 92	0 + 85
CPU time		1.0 + 6.7	1.2 + 7.2	1.3 + 7.6	1.1 + 7.1
$m = 20$, $n = 40$, $X = \{u^{40k-30} \dots u^{40k+9}\}$					
$\|F^i(u_0^i)\|$, $\times 10^{-4}$		0.5	0.7	0.5	0.3
n_{evF}		86 + 59	100 + 64	109 + 70	87 + 54
n_{evP}		0 + 55	0 + 60	0 + 66	0 + 50
CPU time		4.0 + 4.5	4.5 + 4.9	4.8 + 5.4	3.6 + 4.1

Table 9
Accuracy of the Arnoldi process with 50 multiplications by matrix R

Mesh size	# dof	λ_1	$\|Rw_1 - \lambda_1 w_1\|/\lambda_1$	λ_{10}	$\|Rw_{10} - \lambda_{10} w_{10}\|/\lambda_{10}$
$h = 2^{-6}$	3969	2.6E + 2	4E − 16	1.8E − 9	3E − 6
$h = 2^{-7}$	16,129	1.2E + 3	5E − 16	1.1E − 8	1E − 6
$h = 2^{-8}$	65,025	4.7E + 3	5E − 16	5.6E − 8	8E − 7

the eigenvalues λ_1 and λ_{10} of R are shown together with their relative residuals. We see that even for very small values λ_{10} the relative residual is small and decreases with $h \to 0$. Another important observation is that 10 eigenvectors is more than enough for a good accuracy of the POD approximation: $\lambda_k < 10^{-7}$, $k \geqslant 10$.

4. Conclusions

Three methods for the choice of the initial guess in the iterative solution of the series of linear and nonlinear systems have been considered. The systems have been generated by the contemporary time stepping schemes applied to the unsteady nonlinear fluid flow problems. The computational effect of the chosen initial guess has been compared with that of the standard (physically motivated) initial guess. The influence of parameters as well as possible restrictions has been numerically studied. The study shows that the efficiency of the initial guess based on the Krylov subspace data depends on the difference between the matrices in the series. If the matrices are the same (pressure correction step in the Navier–Stokes solver), the GCR data provide the very good initial guess. This results in the computational speed-up. If the matrices are slightly different (pressure correction step in the low Mach number solver), the GMRES data provide a better initial guess in comparison with trivial one, which may save some computational work. However, the efficiency depends severely on the discrepancy between the matrices: the iterative solution of the Newton method series is not accelerated at all. The need to accelerate the Newton method solver has led us to the new choice of the initial guess. It is computed by the solution of the reduced model generated by the Galerkin projection onto the basis of the proper orthogonal decomposition for the solution series. This choice of the initial guess results in the considerable computational speed-up of the fully implicit solution of unsteady nonlinear problems.

Acknowledgment

This work has been supported by the Région Rhône-Alpes in the framework of the project: ''Développement de méthodologies mathématiques pour le calcul scientifique sur grille''.

References

[1] O. Axelsson, Iterative Solution Methods, Cambridge University Press, New York, 1996.
[2] R. Barrett, M. Berry, T. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, H.V. der Vorst, Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods, SIAM, Philadelphia, PA, 1994.
[3] P. Bjorstad, Fast numerical solution of the biharmonic Dirichlet problem on rectangles, SIAM J. Numer. Anal. 20 (1983) 59–71.
[4] P. Brown, Y. Saad, Hybrid Krylov methods for nonlinear systems of equations, SIAM J. Sci. Statist. Comput. 11 (1990) 450–481.
[5] T.F. Chan, W.L. Wan, Analysis of projection methods for solving linear systems with multiple right-hand sides, SIAM J. Sci. Comput. 1 (6) (1997) 1698–1721.
[6] D. Drikakis, O.P. Iliev, D.P. Vassileva, A nonlinear multigrid method for the three-dimensional incompressible Navier–Stokes equations, J. Comput. Phys. 146 (1998) 301–321.
[7] J. Erhel, F. Guyomarc'h, An augmented conjugate gradient method for solving consecutive symmetric positive definite linear systems, SIAM J. Matrix Anal. Appl. 2 (4) (2000) 1279–1299.
[8] P.F. Fischer, Projection techniques for iterative solution of $A\underline{x} = \underline{b}$ with successive right-hand sides, Comput. Methods Appl. Mech. Eng. 16 (1-4) (1998) 193–204.
[9] S. Eisenstat, H. Walker, Globally convergent inexact Newton methods, SIAM J. Optim. 4 (1994) 393–422.
[10] S. Eisenstat, H. Walker, Choosing the forcing terms in an inexact Newton method, SIAM J. Sci. Comput. 17 (1996) 16–32.
[11] M. Feistauer, Mathematical Methods in Fluid Dynamics, Pitman Monographs and Surveys in Pure and Applied Mathematics Series 67, Longman Scientific & Technical, Harlow, 1993.
[12] M. Garbey, Y. Vassilevski, A parallel solver for unsteady incompressible 3D Navier–Stokes equations, Parallel Computing 27 (2001) 363–389.
[13] R. Glowinski, O. Pironneau, Numerical methods for the first biharmonic equation and for the two-dimensional Stokes problem, SIAM Rev. 21 (1979) 167–212.
[14] P. Gosselet, Ch. Rey, On a selective reuse of Krylov subspaces in Newton–Krylov approaches for nonlinear elasticityDomain Decomposition Methods in Science and Engineering, Natl. Auton. Univ. Mex., Mexico, 2003, pp. 419–426.
[15] W. Gropp, L. Curfman McInnes, B. Smith, Using the scalable nonlinear equations solvers package, Technical Report ANL/MCS-TM-193, Argonne National Laboratory, Argonne, IL, 1995.
[16] C.T. Kelley, Iterative methods for optimization, Frontiers in Applied Mathematics 18, SIAM, Philadelphia, 1999.
[17] I. Keshtiban, F. Belblidia, M. Webster, Compressible flow solvers for Low Mach number flows – a review, Technical Report CSR2, Institute of Non-Newtonian Fluid Mechanics, University of Wales, Swansea, UK, 2004.
[18] J. Kim, P. Moin, Application of a fractional step method to incompressible Navier–Stokes equations, J. Comput. Phys. 59 (1985) 308–323.
[19] D.A. Knoll, D.E. Keyes, Jacobian-free Newton–Krylov methods: a survey of approaches and applications, J. Comput. Phys. 193 (2004) 357–397.
[20] D.A. Knoll, P.R. McHugh, An inexact newton algorithm for solving the Tokamak edge plasma fluid equations on a multiply-connected domain, J. Comput. Phys. 116 (1995) 281–291.
[21] Y. Kuznetsov, Numerical methods in subspaces, Computational Processes and Systems, Nauka, Moscow, 1985, pp. 265–350 (in Russian).
[22] R. Lehoucq, D. Sorensen, C. Yang, ARPACK users' guide: solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods, Software, Environments, and Tools, vol. 6, SIAM, 1998.
[23] G. Marchuk, Y. Kuznetsov, A. Matsokin, Fictitious domain and domain decomposition methods, Sov. J. Numer. Anal. Math. Modelling 1 (1986) 3–35.
[24] M. Marion, R. Temam, Navier–Stokes equations: theory and approximation, Handbook of Numerical Analysis, vol. VI, Elsevier, Amsterdam, 1998, pp. 503–689.
[25] G. Meurant, Computer Solution of Large Linear Systems, North-Holland, Amsterdam, 1999.
[26] G. Meurant, The computation of bounds for the norm of the error in the conjugate gradient algorithm, Numer. Algorithms 16-1 (1998) 77–87.
[27] G. Golub, G. Meurant, Matrices, moments and quadrature II: how to compute the norm of the error in iterative methods, BIT 37-3 (1997) 687–705.
[28] M. Pernice, H. Walker, NITSOL: a Newton iterative solver for nonlinear systems, SIAM J. Sci. Comput. 19 (1998) 302–318.
[29] M. Rathinam, L. Petzold, Dynamic iteration using reduced order models: a method for simulation of large scale modular systems, SIAM J. Numer. Anal. 40 (2002) 1446–1474.
[30] M. Rathinam, L. Petzold, A new look at proper orthogonal decomposition, SIAM J. Numer. Anal. 41 (2003) 1893–1925.
[31] F. Risler, Ch. Rey, Iterative accelerating algorithms with Krylov subspaces for the solution to large-scale nonlinear problems, Numer. Algorithms 2 (1) (2000) 1–30.

[32] T. Rossi, J. Toivanen, A parallel fast direct solver for the discrete solver for block tridiagonal systems with separable matrices of arbitrary dimension, SIAM J. Sci. Comp. 20 (1999) 1778–1793.
[33] Y. Saad, Iterative Methods for Sparse Linear Systems, PWS Publishing Co., Boston, 1996.
[34] M. Schefer, S. Turek, Benchmark computations of laminar flow around a cylinder, in: E.H. Hirschel (Ed.), Flow Simulation with High-Performance Computers II, Notes on Numerical Fluid Mechanics, vol. 52, Vieweg, 1996, pp. 547–566.
[35] D. Tromeur-Dervout, Y. Vassilevski, Fully implicit solution of nonlinear PDEs using POD basis on metacomputing architecture, preprint CDCSP-0500, 2005.
[36] D. Tromeur-Dervout, Résolution des Equations de Navier–Stokes en Formulation Vitesse Tourbillon sur Systèmes multiprocesseurs à Mémoire Distribuée, Thesis, Univ. Paris VI, 1993.