

**An Efficient Newton-Krylov
Implementation of the
Constrained Runs Scheme for
Initializing on a Slow Manifold**

Christophe Vandekerckhove

Ioannis Kevrekidis

Dirk Roose

Report TW 502, October 2007



Katholieke Universiteit Leuven
Department of Computer Science
Celestijnenlaan 200A – B-3001 Heverlee (Belgium)

An Efficient Newton-Krylov Implementation of the Constrained Runs Scheme for Initializing on a Slow Manifold

Christophe Vandekerckhove

Ioannis Kevrekidis

Dirk Roose

Report TW 502, October 2007

Department of Computer Science, K.U.Leuven

Abstract

The long-term dynamics of many dynamical systems evolves on a low-dimensional, attracting, invariant slow manifold, which can be parameterized by only a few variables (“observables”). The explicit derivation of such a slow manifold (and thus, the reduction of the long-term system dynamics) can be extremely difficult, or practically impossible. For this class of problems, the equation-free computational approach has been developed to perform numerical tasks with the unavailable reduced (coarse-grained) model based on short full model simulations. Each full model simulation should be initialized consistent with the values of the observables and close to the slow manifold. For this purpose, a class of constrained runs functional iterations was recently proposed. The schemes in this class only use the full model simulator and converge, under certain conditions, to an approximation of the desired state on the slow manifold. In this paper, we develop a constrained runs implementation that is based on a (preconditioned) Newton-Krylov method rather than on a functional iteration. We implement and compare both the functional iteration and Newton-Krylov method, using, as the full simulator, a lattice-Boltzmann model for a one-dimensional nonlinear reaction diffusion system. Depending on the parameters of the lattice Boltzmann model, the constrained runs functional iteration may converge slowly or even diverge. We show that both issues are largely resolved by using the constrained runs Newton-Krylov method, especially when a coarse grid correction preconditioner is incorporated.

Keywords : initialization, slow manifold, Newton-Krylov method, constrained runs, lattice Boltzmann model, equation-free computing

An Efficient Newton-Krylov Implementation of the Constrained Runs Scheme for Initializing on a Slow Manifold

Christophe Vandekerckhove^(a), Ioannis G. Kevrekidis^(b) and Dirk Roose^(a)

^(a) Department of Computer Science, Katholieke Universiteit Leuven,
3001 Leuven, Belgium

^(b) Department of Chemical Engineering and PACM, Princeton University,
Princeton, NJ 08544, USA

Abstract

The long-term dynamics of many dynamical systems evolves on a low-dimensional, attracting, invariant slow manifold, which can be parameterized by only a few variables (“observables”). The explicit derivation of such a slow manifold (and thus, the reduction of the long-term system dynamics) can be extremely difficult, or practically impossible. For this class of problems, the equation-free computational approach has been developed to perform numerical tasks with the unavailable reduced (coarse-grained) model based on short full model simulations. Each full model simulation should be initialized consistent with the values of the observables and close to the slow manifold. For this purpose, a class of constrained runs functional iterations was recently proposed [10, 6]. The schemes in this class only use the full model simulator and converge, under certain conditions, to an approximation of the desired state on the slow manifold. In this paper, we develop a constrained runs implementation that is based on a (preconditioned) Newton-Krylov method rather than on a functional iteration. We implement and compare both the functional iteration and Newton-Krylov method, using, as the full simulator, a lattice-Boltzmann model for a one-dimensional nonlinear reaction diffusion system. Depending on the parameters of the lattice Boltzmann model, the constrained runs functional iteration may converge slowly or even diverge. We show that both issues are largely resolved by using the constrained runs Newton-Krylov method, especially when a coarse grid correction preconditioner is incorporated.

Keywords: initialization, slow manifold, Newton-Krylov method, constrained runs, lattice Boltzmann model, equation-free computing

1 Introduction

For an important class of multiscale problems, the available model is given on a detailed (microscopic) level, while we would like to analyze the system on a much coarser (macroscopic) level. The detailed full model may for instance describe the complex interactions between a large number of atoms or molecules, while the interest is in certain coarser variables (“observables”) such as the particle density field. Conceptually, we could then start a simulation with the full model and extract the evolving particle density field. Due to the large amount

of particles and the presence of fast time and short space scales in the full model simulator, however, this is often computationally feasible only over a spatio-temporal domain that is much smaller than the domain over which the observables typically evolve.

To bridge this gap between the detailed scale of the full model and the coarse scale of interest, one traditionally attempts to derive a reduced model. For the example above, this could for instance result in a partial differential equation (PDE) for the evolution of the particle density field, which can then be studied using standard numerical tools. In many cases, the derivation of the reduced dynamical model hinges on the existence of a low-dimensional, attracting, invariant slow manifold, which can be parameterized by the observables (e.g., a resolved discretization of the density field). In a full model simulation, all initial conditions are then quickly attracted towards this slow manifold, on which the reduced dynamics subsequently evolves. This implies that the remaining full model variables quickly become slaved to (functionals of) the observables, and that the full model state can be accurately described in terms of the observables only.

Although it should in principle always be possible to derive a reduced dynamical model for a system possessing such a slow manifold, one may fail to do so, for instance because the closures required in its construction are not accurate or not even available in closed form. For this class of problems, Kevrekidis and coworkers proposed the equation-free computational framework [13], which enables the full model simulator to perform coarse-grained numerical tasks directly by using appropriately initialized short full model simulations. (The term “equation-free” emphasizes that the derivation of an explicitly reduced model is circumvented.) When one is for instance interested in computing the long-term transient behavior of the reduced system, coarse projective time integration [13, 7] or one of the related techniques in [5, 8, 16, 32, 31] can be used. These methods allow to take time steps commensurate to the time scales of the reduced dynamics, by extrapolating the observables that are extracted from a short full model simulation. After the extrapolation, a new full model simulation should be initialized so that it is both consistent with the extrapolated observables and close to the slow manifold. In the equation-free context, the initialization is called “lifting”, and its implementation is crucial for both the accuracy and the efficiency of the approach.

In [10, 6], a class of such lifting/initialization schemes was proposed. Given a full model simulator and the values of the observables u_0 , the schemes in this class find the values of the remaining full model variables v so that the full model state (u_0, v) is, up to a certain accuracy, close to the slow manifold. Specifically, the m -th scheme in this class ($m = 0, 1, \dots$) computes the values v so that the $(m + 1)$ -st time derivative $d^{m+1}v/dt^{m+1}(u_0, v)$ is zero (or, more generally, bounded). It was shown that, under certain conditions, the state (u_0, v) is then an m -th-order approximation to the desired point on the slow manifold [6]. It should be noted that the basic idea of setting derivatives to zero (or keeping them bounded) is not new. To the best of our knowledge, it traces back to the work of Kreiss and Lorenz [14, 1, 15, 17]; more recently similar ideas were also used in [3, 11]. In our equation-free setting, we however assume only to have a full model simulator available. Therefore, the analytical time derivatives should be approximated numerically, for instance using forward differences. In many cases, a functional iteration can then be used to find the zero of the resulting $(m + 1)$ -st forward difference equation. In the remainder of this paper, the resulting scheme will be called the *constrained runs functional iteration*, or *functional iteration* in short.

The purpose of this paper is to develop a variant of this constrained runs functional iteration, in which the fixed point of the $(m+1)$ -st forward difference equation is now computed with a Newton-Krylov solver rather than with a functional iteration. The resulting method

will further be called the *constrained runs Newton-Krylov method*, or *Newton-Krylov method* in short. The potential advantage of the Newton-Krylov method over the functional iteration will be illustrated using a lattice Boltzmann model (LBM) for a nonlinear, one-dimensional reaction-diffusion system as the full model. We show that, depending on the parameters of the LBM, the constrained runs functional iteration may converge slowly or even diverge, and that both issues are largely resolved when using the Newton-Krylov approach.

The outline of this paper is as follows. Section 2 introduces the LBM that will be used as an illustrative example throughout the paper. In Section 3, we briefly review the constrained runs functional iteration, and illustrate its performance for the LBM. In Section 4, we develop the constrained runs Newton-Krylov method, and discuss several numerical aspects, such as preconditioning, more extensively. A detailed comparison between the functional iteration and the Newton-Krylov approach is provided in Section 5 for a large range of parameter values of the LBM model problem. A brief discussion of the accuracy of the constrained runs scheme is given in Section 6, after which we conclude in Section 7.

2 The Lattice Boltzmann Model

In this section, we present the lattice Boltzmann model (LBM) for the one-dimensional reaction-diffusion system that will serve as an example throughout the paper.

Let us first note that we have chosen the LBM model problem over one of the singularly perturbed systems from [6] for several reasons. First, in our opinion, the LBM serves as a better caricature of the more challenging multiscale problems that we have in mind, while it is still simple enough to be analyzed in full detail. Second, the LBM example was also used as a model problem in [30, 33], in which several theoretical properties of the constrained runs functional iteration were derived. Some of these results will be of direct use in this paper. Finally, the fully discretized state space of the LBM contains more than just a few scalar variables, providing a good case study for the numerical properties of the iterative linear algebra techniques used later on in the paper.

A LBM [2] describes the evolution of discrete distribution functions of particles $f_i(x_j, t_k)$, which depend on space x_j , time t_k and velocity v_i . For our one-dimensional model problem, only three values are considered for the velocity ($v_i = i\Delta x/\Delta t$, with $i \in \{-1, 0, 1\}$), and each distribution function f_i is discretized in space on the domain $[0, 1]$ using a grid spacing $\Delta x = 1/N$ (N lattice intervals) and in time using a time step Δt . The LBM evolution law for the distributions $f_i(x_j, t_k)$ on the interior of the domain is then

$$\begin{aligned} f_i(x_{j+i}, t_{k+1}) &= f_i(x_j + i\Delta x, t_k + \Delta t) \\ &= f_i(x_j, t_k) - \omega (f_i(x_j, t_k) - f_i^{eq}(x_j, t_k)) + \lambda \frac{\Delta t}{3} \rho(x_j, t_k) (1 - \rho(x_j, t_k)), \end{aligned} \quad (1)$$

with $i \in \{-1, 0, 1\}$. Note that each LBM evolution step naturally decomposes in two subsequent phases: first, the distribution functions $f_i(x_j, t_k)$ are updated, and then they stream to a neighboring lattice site according to their velocity direction. At the boundaries, we impose Dirichlet boundary conditions $\rho(0, t_k) = \rho(1, t_k) = 0$ by assigning the appropriate value to the distribution functions that stream into the domain at $x_0 = 0$ and $x_N = 1$ (here, ρ is the (particle) density field as defined by equation (3) below).

Diffusive collisions are modeled by the Bhatnagar-Gross-Krook (BGK) collision term — the term $-\omega(f_i(x_j, t_k) - f_i^{eq}(x_j, t_k))$ in (1) — as a relaxation to the local diffusive equilibrium

[22]

$$f_i^{eq}(x_j, t_k) = \frac{1}{3}\rho(x_j, t_k). \quad (2)$$

The parameter $\omega \in (0, 2)$ is called the relaxation coefficient and ρ is the (particle) density field, which is defined as the zeroth-order velocity moment of $f_i(x_j, t_k)$

$$\rho(x_j, t_k) = \sum_{i=-1}^1 f_i(x_j, t_k). \quad (3)$$

It should be clear that the BGK diffusive collisions locally conserve density.

The last term in equation (1) models the nonlinear reactions which depend only on the density field [22, 4]. The parameter $\lambda \geq 0$ indicates the strength of the “reaction force”.

Similar to the density ρ , we can also define the momentum ϕ and the energy ξ as (a rescaling of) the first and the second order (or in short: the higher order) moments of $f_i(x_j, t_k)$

$$\phi(x_j, t_k) = \sum_{i=-1}^1 i f_i(x_j, t_k), \quad \xi(x_j, t_k) = \frac{1}{2} \sum_{i=-1}^1 i^2 f_i(x_j, t_k). \quad (4)$$

Since there is a one-to-one relation between the discrete particle distribution functions f_{-1} , f_0 , f_1 and the discrete velocity moments ρ , ϕ , ξ , the LBM state is equally well described by each of these variable sets. In this paper, we will further use the moment description, as this will turn out to be a more natural choice in the context of equation-free computing.

For the LBM above, a multiscale Chapman-Enskog expansion [2, 27] can be used to derive an accurate reduced model for the long-term behavior of the density ρ , which now plays the role of the observable. It turns out that this reduced model is the Fisher equation

$$\frac{\partial \rho(x, t)}{\partial t} = D \frac{\partial^2 \rho(x, t)}{\partial x^2} + \lambda \rho(1 - \rho) = \left(\frac{2 - \omega}{3\omega} \frac{\Delta x^2}{\Delta t} \right) \frac{\partial^2 \rho(x, t)}{\partial x^2} + \lambda \rho(1 - \rho), \quad (5)$$

with Dirichlet boundary conditions $\rho(0, t) = \rho(1, t) = 0$. Note that for any space-time grid, a value of $\omega \in (0, 2)$ can be chosen so that the code effectively implements the prescribed diffusion coefficient $D > 0$. For simplicity, we will choose $D = 1$ in this paper. As the parameters Δx (or N), ω and λ will be varied in order to illustrate different aspects of our computational approach, this implies that we give up the degree of freedom in choosing the time step size Δt .

The fact that the reduced dynamics can be described in terms of the density ρ only, implies that the higher order moments ϕ and ξ , which now play the role of the remaining full model variables, become functionals of (slaved to) ρ on a time scale that is very short compared to the dominant time scales of the reduced model. These functionals, which we will call slaving relations, can be found explicitly as a by-product of the Chapman-Enskog expansion. After dropping the indices j and k , and retaining only terms up to third order, we obtain

$$\phi(x, t) = -\frac{2}{3\omega} \frac{\partial \rho(x, t)}{\partial x} \Delta x + \frac{\omega^2 - 2\omega + 2}{9\omega^3} \frac{\partial^3 \rho(x, t)}{\partial x^3} \Delta x^3 + \mathcal{O}(\Delta x^5), \quad (6)$$

$$\xi(x, t) = \frac{1}{3}\rho(x, t) - \frac{\omega - 2}{18\omega^2} \frac{\partial^2 \rho(x, t)}{\partial x^2} \Delta x^2 + \mathcal{O}(\Delta x^4). \quad (7)$$

These relations define the slow manifold in the LBM phase space. The reduced model (5) is then a description for the dynamics of the density ρ on this manifold. Although a

trajectory starting from an arbitrary LBM initial condition is quickly attracted towards the slow manifold, an off-manifold initialization may introduce a substantial and persistent error, as ρ may also change substantially during the initial, fast transient phase [29, 33]. The latter can/should be avoided by initializing on (or very close to) the slow manifold.

3 The Class of Constrained Runs Functional Iterations

In this section, we give a brief review of the class of constrained runs functional iterations as proposed in [10, 6], and apply it to the LBM model problem.

3.1 Review of the Class of Constrained Runs Functional Iterations

Given a full model simulator and the values of the N_u -dimensional observables u_0 , each scheme in the class of constrained runs functional iterations approximates the values of the N_v -dimensional remaining full model variables v , so that the full model state (u_0, v) is, up to a certain accuracy, close to the slow manifold. Specifically, the m -th scheme in this class ($m = 0, 1, \dots$) computes the remaining full model variables v so that the $(m + 1)$ -st time derivative of v is zero. Hence, v is the solution of the N_v -dimensional nonlinear system

$$\left(\frac{d^{m+1}v}{dt^{m+1}}\right)(u_0, v) = 0, \quad (8)$$

which is referred to as the “ $(m + 1)$ -st derivative condition”. The intuitive reason why this condition (or, more generally, a condition asking for the $m + 1$ -st derivative to be bounded) yields a state close to the slow manifold is that time differentiation amplifies fast components more than slow components, so demanding for a small value of their time derivatives implies that the fast components are small and hence that we are close to the slow manifold. The larger the value of m , the larger the amplification of the fast components, and hence also the closer we get to the slow manifold. In [6], it was shown rigourously that, under certain conditions, the resulting state (u_0, v) is indeed an m -th-order approximation to the desired point on the slow manifold. In our equation-free setting, we assume that only a full model simulator is available, so in practice the analytical time derivative in (8) should be approximated numerically, for instance using finite differences. If we set up a functional iteration to find the fixed point of the resulting N_v -dimensional nonlinear $(m + 1)$ -st order forward difference equation, we obtain the following iterative scheme:

0. Initialize v as well as possible (see further). Then start the iteration 1–3.
1. Run the full model simulator over $m + 1$ time steps, starting from (u_0, v) . This gives the values (u_j, v_j) at times $t = j\Delta t$ ($j = 1, \dots, m + 1$).
2. Compute the forward difference $\delta_v = (-1)^m \Delta^{m+1} v$ from v_j ($j = 1, \dots, m + 1$).
3. If $\|\delta_v\|$ is smaller than a certain tolerance: end the iteration.
Else: set $v = v + \delta_v$ and goto 1.

This scheme is called the constrained runs functional iteration. A close inspection shows that in each iteration step the values of the remaining full model variables v are updated using a backward extrapolation with a polynomial of degree m passing through v_j ($j = 1, \dots, m + 1$),

Table 1: The parameter ranges $(\omega_{\min}, \omega_{\max})$ for which the constrained runs functional iteration is stable when using the LBM with $\lambda = 0$, $N = 100$, and various values of m .

m	0	1	2	3	4
ω_{\min}	0.000	0.690	0.865	0.929	0.959
ω_{\max}	2.000	1.291	1.133	1.072	1.043

while the observables u_0 are reset to their original, prescribed values. Therefore, there is a strong connection with what is called reverse or stationary projective time integration in [9].

The functional iteration will not necessarily converge to the desired solution: the iteration may diverge or we may converge to a nonphysical solution (equation (8) may have multiple roots if we have a nonlinear full model simulator). Using a good initial guess may then be important. Such an initial guess can be based on problem-specific knowledge (for the LBM for instance, one might start from the equilibrium distribution functions (2)). The context in which the procedure is used may also provide a good initial guess. In the case of coarse projective integration for instance, it could be based on previous computations as there is a strong correlation between successive states. We may for instance use just the previously computed remaining full model variables or an appropriate linear combination of the remaining full model variables from several previous lifting steps. Finally, it may also be useful to use the solution obtained with a small value of m as the initial condition to compute the solution for a larger value of m , as often the numerical difficulties gradually arise when m is increased.

3.2 Application to the LBM

In [30], it was shown that for the LBM from Section 2 and $m = 0$, the constrained runs functional iteration is stable for all values of $\lambda \geq 0$ and $\omega \in (0, 2)$ (remember that now $u_0 = \rho$ and $v = (\phi, \xi)$). In this section, we study what happens when $m > 0$.

A first important observation is that the unconditional stability of the functional iteration no longer holds when $m > 0$. This is illustrated in Table 1, which shows the parameter ranges of ω for which the functional iteration is stable when $\lambda = 0$ (the linear LBM case), $N = 100$ and using various values of m . When m grows these ranges clearly become much smaller. Hence, using a higher order derivative imposes a more stringent stability condition. Slightly remarkable here is the fact that the functional iteration is also unstable for values of $\omega \rightarrow 2$, as then $\Delta t \rightarrow 0$ when Δx is kept constant (so driving Δt towards zero is not sufficient for stability). Note that even when the functional iteration is stable, it might still take a large amount of work to converge to the fixed point. In practice, this will happen when ω approaches the boundary of stability.

Concerning the accuracy of the constrained runs functional iteration, we now numerically verify that the solutions are indeed increasingly good approximations of the exact slaved state when m increases; a brief theoretical error analysis for the LBM was presented in [33]. To this end, we do the following experiment. First, we perform an initial simulation of 1000 LBM steps, starting from the arbitrarily chosen initial condition $(\rho, \phi, \xi) = (x(1-x), 0, 0)$ and choosing $\lambda = 0$, $\omega = 0.75$ and $N = 100$. We consider the state at the 1000-th step as the “exact” slaved state (ρ^*, ϕ^*, ξ^*) , as the LBM simulation steps have brought us onto the

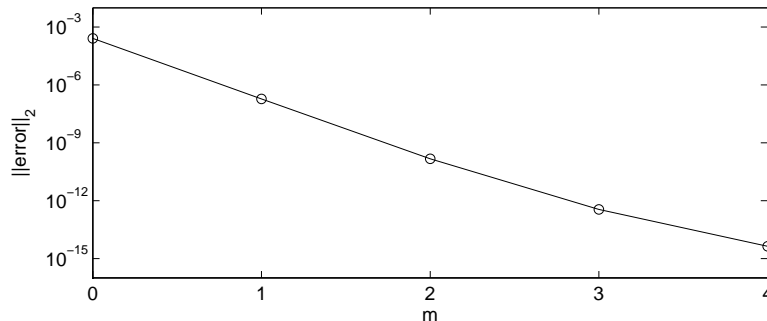


Figure 1: The two-norm of the error in the higher order moments (ϕ, ξ) obtained with the constrained runs Newton-Krylov method for different values of m .

slow manifold corresponding to (6)–(7). (Such a state is also referred to in the literature as a “mature”, “bred” or “healed” state [13, 21].) Then, we reconstruct the higher order moments corresponding to ρ^* (not ρ) using the constrained runs Newton-Krylov method from Section 4 for various values of m . Figure 1 shows the two-norm of the error, i.e., the two-norm of the difference between these approximations of the higher order moments and the “exact” higher order moments (ϕ^*, ξ^*) . We see that we indeed obtain increasingly good approximations to the exact slaved state as m increases (until we reach the level of machine precision). Note that the linear convergence observed is in line with the theoretical results in [6, 33]. From Table 1, it should be clear that the results for $m = 2, 3$ and 4 in Figure 1 could never be computed using the constrained runs functional iteration, as this method is then unstable.

4 The Constrained Runs Newton-Krylov Method

In the previous section, we showed that the constrained runs functional iteration may converge slowly or even diverge, depending on properties of the full model. In this section, we develop a variant in which the fixed points of the $(m + 1)$ -st forward difference equations are now computed with a Newton-Krylov solver rather than with a functional iteration. As we will show further on, this may largely resolve the slow convergence issues and the stability problems.

4.1 The Fixed Point Formulation

The constrained runs functional iteration described in Section 3 can be written symbolically as

$$v_{k+1} = \mathcal{C}(u_0; v_k). \quad (9)$$

Here, \mathcal{C} is a map from $\mathbb{R}^{N_u+N_v}$ to \mathbb{R}^{N_v} , u_0 is a parameter (although u also varies during the full model simulation, we are not solving for u_0) and v_k is the k -th iterate of the functional iteration for the unknown v . Although the functional iteration may encounter difficulties in finding the fixed point (slow convergence or divergence), the fixed point problem

$$g(u_0; v) := v - \mathcal{C}(u_0; v) = 0 \quad (10)$$

is well defined (it has isolated solutions).

In Section 4.2, we will show how this nonlinear system can be solved using Newton's method. It is instructive however to study first what happens when the full model is linear. Equation (10) can then be written as

$$g(u_0; v) = v - [C_u \ C_v] \cdot \begin{bmatrix} u_0 \\ v \end{bmatrix} = 0 \Leftrightarrow (I - C_v) v = C_u u_0, \quad (11)$$

where $I \in \mathbb{R}^{N_v \times N_v}$ is the identity matrix and $C_u = \partial \mathcal{C} / \partial u \in \mathbb{R}^{N_v \times N_u}$ and $C_v = \partial \mathcal{C} / \partial v \in \mathbb{R}^{N_v \times N_v}$ are submatrices of the (constant) Jacobian matrix of the (linear) map \mathcal{C} . Using the backward extrapolation interpretation of the constrained runs functional iteration, we can derive that when L denotes the Jacobian matrix of the full model simulator, C_u and C_v are the appropriate submatrices of $I - (I - L)^{m+1}$. Note that for the LBM and $m = 0$, equation (11) is also valid when $\lambda \neq 0$, as the LBM reaction force only depends on $u_0 = \rho$.

4.2 Solving the Nonlinear System with Newton's Method

The (in general) nonlinear system (10) can be solved using Newton's method

$$v_{k+1} = v_k + \delta v_k, \quad (12)$$

where the correction δv_k made in the k -th Newton iteration step is found by solving the linear system

$$A(u_0; v_k) \cdot \delta v_k = \frac{\partial g}{\partial v}(u_0; v_k) \cdot \delta v_k = \left(I - \frac{\partial \mathcal{C}}{\partial v}(u_0; v_k) \right) \cdot \delta v_k = -g(u_0; v_k), \quad (13)$$

with $A = \partial g / \partial v$ the linearization (Jacobian matrix) of g and $\partial \mathcal{C} / \partial v$ the linearization of \mathcal{C} . We emphasize again that these matrices have dimension $N_v \times N_v$.

In many cases we do not have available the linearization of \mathcal{C} , as it involves the full model simulator. However, we can numerically estimate the action of the Jacobian using a finite difference formula for the directional derivative

$$\left(I - \frac{\partial \mathcal{C}}{\partial v}(u_0; v_k) \right) \cdot \delta v_k \approx \delta v_k - \frac{\mathcal{C}(u_0; v_k + \epsilon \delta v_k) - \mathcal{C}(u_0; v_k)}{\epsilon}, \quad (14)$$

in which ϵ can for instance be chosen as

$$\begin{cases} \epsilon = \sqrt{\bar{\epsilon}} \|v_k\| / \|\delta v_k\| & \text{if } \delta v_k \neq 0, v_k \neq 0 \\ \epsilon = \sqrt{\bar{\epsilon}} / \|\delta v_k\| & \text{if } v_k = 0, \delta v_k \neq 0, \\ \epsilon = 1 & \text{if } \delta v_k = 0 \end{cases}, \quad (15)$$

with $\bar{\epsilon}$ denoting the machine precision. The error of the finite difference approximation of the directional derivative is then roughly minimized and of order $\mathcal{O}(\sqrt{\bar{\epsilon}})$ [12]. If the linear system solver returns a solution of which the norm of the relative residual is also of order $\mathcal{O}(\sqrt{\bar{\epsilon}})$, Newton's method is expected to converge quadratically up to the level of machine precision. It is conceivable that the linear systems in the early Newton steps are then however solved to a precision far beyond what is needed to correct the nonlinear iteration, and it might therefore be better to incorporate a strategy for choosing a sequence of tolerances for the linear solver (see for instance [12]). We then avoid oversolving the linear systems while retaining quadratic convergence under normal circumstances. If the tolerance is however chosen too large, the quadratic convergence will be lost and more Newton iteration steps may be needed.

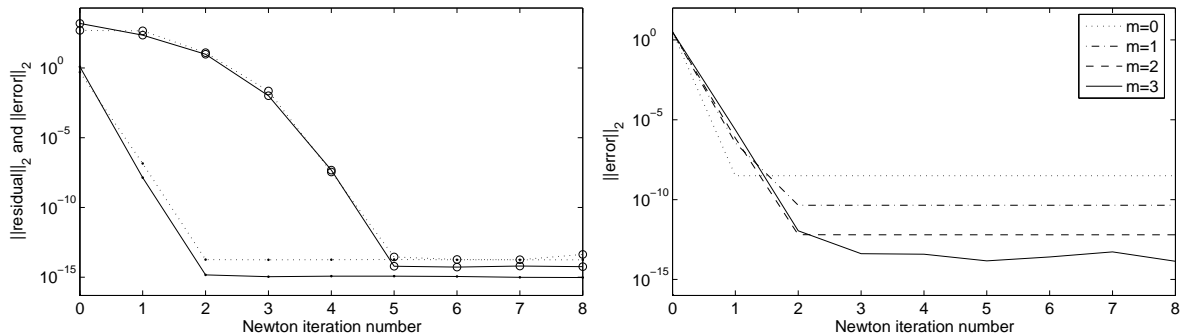


Figure 2: **Left:** the evolution of the two-norm of the nonlinear residual (full lines) and the error (dotted lines) during the Newton process, when $m = 3$ and for $\lambda = 0$ (dot markers) and $\lambda = 1000$ (circle markers). In the nonlinear case, a very irregular initial guess is used. **Right:** the evolution of the error for different values of m , when $\lambda = 1000$. A zero initial guess is used.

The linear system (13) can be solved efficiently using a Krylov subspace method; this will be the topic of the next section. First however, we consider an abstraction of the linear solver (we assume that it is exact) and show how the Newton iteration converges for the LBM model problem. As in the example of Section 4.1, we determine the “exact” slaved state by a preliminary LBM run. Again, we use $N = 100$ but now we choose $\omega = 1.25$. The higher order moments are then reconstructed by solving the nonlinear system (10) as explained above.

Figure 2 (left) shows the evolution of the two-norm of the nonlinear residual and the two-norm of the error (we compare to the “exact” slaved state) during the Newton iteration, when $m = 3$ and for $\lambda = 0$ (linear LBM) and $\lambda = 1000$ (nonlinear LBM). For the linear LBM, two Newton steps are required since (14)–(15) only approximates the matrix-vector product up to about eight digits ($\sqrt{\epsilon} \approx 10^{-8}$). For the nonlinear LBM, we used a very irregular initial guess to demonstrate the quadratic behavior of the Newton process. When a smooth initial guess is used (e.g., a zero initial guess), the convergence is typically so fast that the quadratic behavior cannot clearly be observed.

Figure 2 (right) shows the evolution of the two-norm of the error during the Newton iteration, also for $\lambda = 1000$ but now for various values of m . In this case, we did start from a zero initial guess. Each of the fixed point problems is now solved in just two or three iteration steps. We again observe that the constrained runs approximations are more accurate if m is larger, and that there is linear convergence in m .

4.3 Solving the Linear systems with GMRES

In this section, we focus on solving the linear system (13). A simple but naive solution strategy would be to explicitly construct the matrix $A = \partial g / \partial v$ and the right-hand side $b = -g$ numerically, and to solve the system using Gaussian elimination. For our one-dimensional LBM model problem, a sparse solver of linear CPU/memory complexity could then be used, since reordering the unknowns per spatial grid point results in a sparse banded Jacobian matrix A with a limited bandwidth of $7 + 4m$. However, without making any assumptions or changes to the existing LBM code (an assumption which certainly makes sense for more complex full models), this would require $2N + 3$ evaluations of \mathcal{C} . As evaluating \mathcal{C} involves running the expensive full model simulator, this may in practice lead to an unacceptably large

computational cost. By using a Krylov subspace method, a sufficiently accurate solution can often be found using significantly less evaluations of \mathcal{C} . In this section, we explore the use of GMRES [25]. We first study how unpreconditioned GMRES behaves, and we then show that by incorporating an appropriate preconditioner the efficiency can be increased even further.

4.3.1 Unpreconditioned GMRES

In this section, we solve the system $Ax = b$ with unpreconditioned GMRES [25]. For simplicity, we do not consider special variants of the standard GMRES algorithm such as restarted or truncated GMRES at this point. GMRES approximates the exact solution $x^* = A^{-1}b$ by the vector x_n from the Krylov subspace $\mathcal{K}_n = \langle r_0, Ar_0, \dots, A^{n-1}r_0 \rangle$ generated by A and r_0 , such that the two-norm of the residual $r_n = b - Ax_n$ is minimized. Hence, within the class of Krylov subspace methods, GMRES makes optimal use of the matrix-vector products, which are (at least for the more challenging multiscale problems that we have in mind) very computationally expensive as they involve running the full model simulation code.

A theoretical upper bound for the GMRES convergence rate can be found from its interpretation as a polynomial approximation problem. Loosely speaking, the standard way is to look for a set of polynomials $p_n(z)$ of increasing degree n that satisfy $p_n(0) = 1$ and of which the size on the eigenvalue spectrum Λ of A quickly decreases with n . Specifically, it can be shown that

$$\|r_n\|_2 \leq K \inf_{p_n} \sup_{z \in \Lambda} |p_n(z)|, \quad (16)$$

with K a constant depending on A and r_0 [28]. In the remainder of this section, we now use this theorem to explain the GMRES convergence rate when the full model is the LBM.

The case $m = 0$ for the LBM. In [30], it was shown that for $m = 0$, the eigenvalues of the LBM Jacobian matrix $\partial\mathcal{C}/\partial v$ are spread on a circle with radius $|1 - \omega|$ and center point the origin. Therefore, the eigenvalues of $A = \partial g/\partial v$ lie on a circle with radius $|1 - \omega|$ and center point 1 (cf. Figure 4). An upper bound for the GMRES convergence rate is now easily found by using the polynomials $p_n = (1 - z)^n$; we obtain

$$\|r_n\|_2 \leq K|1 - \omega|^n. \quad (17)$$

Hence, for the LBM model problem, GMRES will converge faster whenever ω is closer to one, and the asymptotic convergence is at least as fast as the convergence of the corresponding constrained runs functional iteration.

The worst-case estimate (17) is confirmed in Figure 3 (left) for various values of ω . Here, we used $m = 0$, $\lambda = 0$, $N = 128$, $\rho = x(1-x)$ and a uniformly distributed random initial guess. By using such an initial guess, the GMRES convergence rate agrees very well with the worst-case estimate (17). In the context of Newton's method, one often uses a zero initial guess; the convergence is then typically much faster. In Figure 3, we iterated until the two-norm of the residual has reached the level of machine precision $\bar{\epsilon}$. In practice, the iteration will of course be ended earlier, near the level of the overall error (that is, including the approximation error of the constrained runs procedure itself). The effect of using a different initial guess or a stopping criterion based on an overall error estimate will be studied in Section 5.

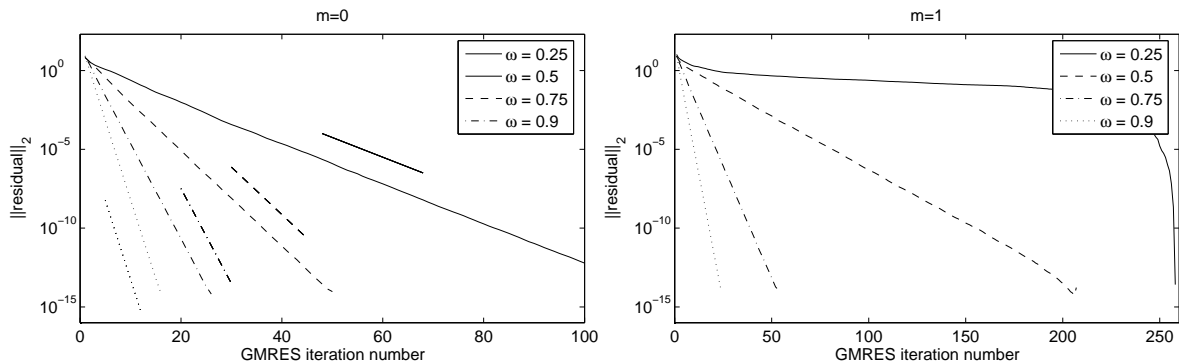


Figure 3: The GMRES convergence history for the LBM with $\lambda = 0$, $N = 128$ and various values of ω , when $m = 0$ (left) or $m = 1$ (right). For the case $m = 0$ the theoretical convergence upper bounds are also depicted (the short lines).

The case $m > 0$ for the LBM. When $m > 0$, the eigenvalues of the Jacobian matrix $\partial\mathcal{C}/\partial v$ no longer lie on circles, and they may be larger than one in magnitude (cf. Figure 6). The latter is the reason that the functional iteration only converges for certain intervals of $\omega \in [0, 2]$ (cf. Table 1). For values of ω outside these intervals, A may no longer be positive definite and the eigenvalues of A may even lie in a region “surrounding the origin”, causing very slow GMRES convergence. This is clearly illustrated in Figure 3 (right), where we used the same setup as before except that now $m = 1$. Note that for $\omega = 0.25$, GMRES only converges faster near the end of the iteration due to the fact that GMRES has to find an exact solution after $2N + 2 = 258$ iteration steps.

4.3.2 Incorporating a Preconditioner

The convergence of the Newton-Krylov method can be accelerated if an adequate matrix-free (we do not have A in equation (13) available!) preconditioner can be incorporated. The idea is that after applying the preconditioner, the spectral properties are more favorable (e.g., the eigenvalues are more clustered and further away from the origin), so that GMRES converges faster and an overall efficiency gain is obtained.

The construction of a good preconditioner is inherently problem-specific. Therefore, we will directly focus on the case where the full model is the LBM from Section 2. Rather than constructing an optimal preconditioner for this model problem however, we will use the more generally applicable two-level coarse grid correction preconditioning approach as presented in [19].

The coarse grid correction preconditioner. We first briefly review the coarse grid correction preconditioner as described in [19]. This preconditioner uses properly chosen projection matrices to correct the low-frequency components in the spectrum. Next to the original fine LBM grid Ω^N with grid spacing $\Delta x = 1/N$, a coarse grid $\Omega^{N/r}$ with grid spacing $r\Delta x$ is introduced. A restriction operator $R_N^{N/r}$ maps the solution from the fine grid Ω^N to the coarse grid $\Omega^{N/r}$, while a prolongation operator $P_{N/r}^N$ implements the reverse mapping from $\Omega^{N/r}$ to Ω^N . In the numerical experiments reported further on, we use injection and linear interpolation for both operations; choosing other operations such as full weighting or higher

order interpolation has only a minor effect on the results. As both the restriction and the prolongation operators are then linear, $R_N^{N/r}$ and $P_{N/r}^N$ can further be considered to be matrices. On the coarse grid, we can solve the coarse system $A_c x = b$, with A_c the coarser discretized counterpart of A ; we further refer to this operation as the “coarse system solve”. As the LBM time step Δt and the spatial grid spacing Δx are coupled through the relaxation coefficient ω (5), it is crucial that we also increase the time step Δt when Δx is increased, in order to keep the value of ω fixed. Using the operations above, the coarse grid correction preconditioner M^{-1} is now defined as

$$M^{-1} = P_{N/r}^N A_c^{-1} R_N^{N/r} + \eta I. \quad (18)$$

Here, η is a parameter with a nonzero value (if $\eta = 0$, M^{-1} is singular!). A detailed analysis of this preconditioner in the case of symmetric positive definite matrices is given in [19].

It can be noted that coarse grid correction preconditioning is closely related to deflation preconditioning, in which a certain deflation subspace is projected out of the residual [18]. While the deflation preconditioner sets the eigenvalues corresponding to the low-frequency components to zero (at least in the ideal case where the deflation subspace is spanned by the corresponding low-frequency eigenmodes), the coarse grid correction preconditioner moves them in the direction of the remaining eigenvalues. In both cases, applying the preconditioner may lead to a substantial improvement of the GMRES convergence rate when the shifted/deflated eigenvalues are the ones causing the slow convergence, for instance because they are lying close to the origin. One of the main advantages of the coarse grid correction preconditioner over the deflation preconditioner is that the coarse system may be solved inexactly, which may lead to a substantial efficiency gain when the computation on the coarse grid is still expensive (as in our case when r is small). The analysis in [19] even shows that in some cases, not solving the coarse system at all ($A_c = I$) may also lead to good results!

A disadvantage of the coarse grid correction preconditioner (18) is that it is, in general, not clear how to choose the optimal value of the parameter η . In this paper, we estimate the optimal value of η in a preprocessing step (which we will further refer to as “ η -estimator”) by solving the linear system for a large number of η -values, and then selecting the value for which the total amount of work (due to both the iterative solver and the preconditioner) is minimized. Although we use this brute force approach to solve what is basically a one-dimensional optimization problem, this extra computational effort will in practice often be negligible compared to the efficiency gain obtained by using the preconditioner. This is especially the case when the same value of η can be reused more than once, for instance during the subsequent lifting steps that are required during coarse projective time integration [13, 7] when we are computing the reduced long-term behavior of the system (cf. Section 5). Along the time integration process, we can also periodically monitor whether the current value of η is still optimal (by continuity, this is not very expensive), and change its value when necessary.

In the following sections we will study how the coarse grid correction preconditioner (18) increases the overall efficiency when the full model is the LBM model problem and $m = 0$ or $m = 1$. We explore the use of different coarse grids and determine the effect of solving the system on the coarse grid up to only a certain accuracy, for instance by using only a small number of “inner”, unpreconditioned GMRES steps on the coarse level. The preconditioner then varies at each “outer” GMRES step (M^{-1} is no longer a constant operator), and therefore we use flexible GMRES [23] at the outer level, as this method allows variations in the preconditioning. Flexible GMRES is based on a modified Arnoldi relation, implying that

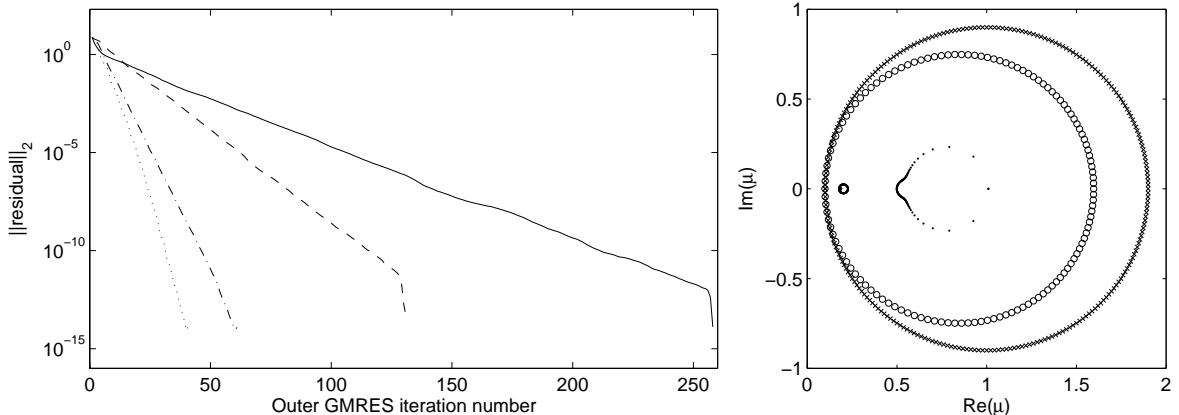


Figure 4: **Left:** the two-norm of the residual as a function of the outer GMRES iteration number, for the preconditioning variants P1: without preconditioner (full), P2: with the preconditioner with exact coarse system solve (dotted), P3: with the preconditioner with partial coarse system solve (dash-dotted), or P4: with the preconditioner without coarse system solve (dashed). **Right:** the eigenvalues of A (crosses), M (circles) and $M^{-1}A$ (dots), when using the preconditioner P2.

the residual is in general no longer minimized over a standard Krylov subspace. Technically, this implies that one additional vector must be saved in each iteration step. The remaining numerical results in this article were computed using flexible GMRES as the outer solver. For simplicity, we will still refer to it as “GMRES”.

The case $m = 0$. The result of our first numerical experiment is shown in Figure 4 (left). Here, the two-norm of the residual is shown as a function of the outer GMRES iteration number, when using four different preconditioning variants: (P1) without preconditioner, (P2) with the preconditioner (18) and solving the coarse system exactly using unpreconditioned GMRES, (P3) with the preconditioner (18) and solving the coarse system only partially using two unpreconditioned GMRES steps, or (P4) with the preconditioner (18) but without solving the coarse system (i.e., $A_c = I$). For the inner GMRES, we use $R_N^{N/2}b$ as the initial guess. Furthermore, we use $N = 128$, $\omega = 0.1$, $\rho = x(1 - x)$, $r = 2$, $\eta = \omega$, and we again use a uniformly distributed random initial guess to illustrate the worst case behavior. Figure 4 (left) shows that the smallest number of outer GMRES iteration steps is required when the coarse system is solved exactly. Solving the coarse system only partially results in just a slight increase of the number of outer iteration steps. Even if the coarse system is not solved at all, the total number of outer iteration steps is still only half the amount that is needed when no preconditioner is used.

Figure 4 (right) shows the eigenvalue spectra of A , M and $M^{-1}A$, when using the preconditioner P2. A large part of the eigenvalues of M are equal to $1/\eta = 10$ and are therefore not visible in the figure. We clearly observe that the left-most eigenvalues μ_i of A are well approximated by some of the eigenvalues of M . A computation reveals that for $\omega < 1$, these left-most eigenvalues of A near $\mu = \omega$ correspond to the low-frequency eigenmodes, while for $\omega > 1$, the right-most eigenvalues of A near $\mu = 2 - \omega$ correspond to the low-frequency eigenmodes. As the coarse grid correction preconditioner now shifts the eigenvalues corresponding to the low-frequency eigenmodes in the direction of the remaining eigenvalues, the spectrum

Table 2: The overall amount of working units (see text) needed to lower the two-norm of the residual towards machine precision, when using the different variants of the coarse grid correction preconditioner.

	P1	P2	P3	P4
$r = 2$	100233	1030254	59094	51471
$r = 8$	100233	202575	49914	34443

will contain less eigenvalues very close to the origin after preconditioning ($\omega = 0.1$). Together with the observation that the spectrum is also “compacted” in one small and one medium-size cluster, this rationalizes the improved GMRES convergence rate (and it also already indicates that the preconditioner may fail when $\omega > 1$).

It is crucial to note that, although solving the coarse system exactly results in the smallest number of outer GMRES iteration steps, it does not lead to the most efficient preconditioner in practice, as it is also the most computationally expensive one. This is illustrated in Table 2. Here, the number of “LBM working units” is shown for the four different preconditioning scenarios and for $r = 2$ and $r = 8$. One such working unit is defined as the amount of work required to update one grid point in the LBM code. So, one LBM call with a grid size N requires $3(N + 1)$ working units. We choose the number of working units rather than the total number of matrix-vector products, as the cost of one matrix-vector product differs on the original and on the coarse grid. In the unpreconditioned case (P1), $(2(N + 1) + 1) \times 3(N + 1) = 100233$ working units are required. Both for $r = 2$ or $r = 8$, the amount of working units increases by applying the preconditioner P2, while it decreases when the preconditioners P3 or P4 are used. The best preconditioner P4 reduces the amount of working units to about half ($r = 2$) or one third ($r = 8$) of its original value in the unpreconditioned case.

Figure 5 shows how the performance of the coarse grid correction preconditioner depends on the value of ω . We use $\rho = x(1 - x)$, set $N = 512$ and plot, as a function of ω , the number of outer GMRES iteration steps that is needed to lower the two-norm of the residual towards machine precision, both in the unpreconditioned (P1) and in a preconditioned (P4, $r = 8$, $\eta = \omega$) case and again starting from a random initial guess. We set $\eta = \omega$ as the η -estimator described earlier on suggests that this is close to optimal. As we saw before, GMRES converges faster when ω is closer to one. For $\omega \in [0.6, 1.4]$, the residual is decreased by at least eight orders of magnitude within 20 iteration steps, even without a preconditioner. Outside this interval, the number of iteration steps increases drastically in the unpreconditioned case. In the range $\omega \in (0, 0.6)$, the coarse grid preconditioner lowers the number of iteration steps substantially, while it has no beneficial effect in the range $\omega \in (1.4, 2)$. As already hinted at before, this is related to the fact that the eigenvalues near zero now correspond to the high-frequency eigenmodes. Finally, we also estimated the number of iteration steps in the unpreconditioned case by combining the theoretical convergence rate (17) with the fact that we let the residual decreased by about 13 orders of magnitude, and taking also into account that GMRES finds an exact solution in at most $2N + 2$ iteration steps. Figure 5 shows that our estimate is very accurate, reconfirming the validity of equation (17).

The case $m = 1$. We now show that the coarse grid correction preconditioner (18) is even more efficient when $m = 1$. To illustrate this, we choose $\omega = 0.25$, well outside the interval

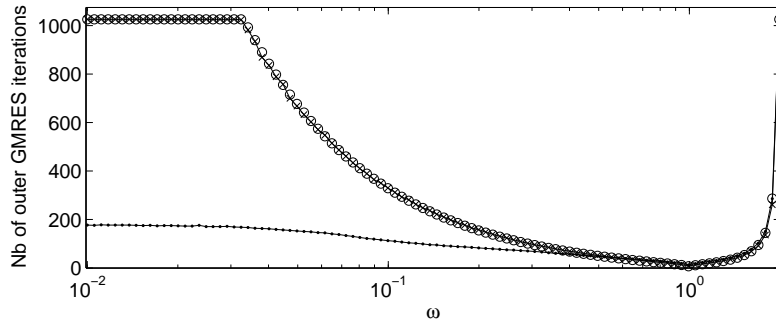


Figure 5: The number of outer GMRES iteration steps as a function of ω , in the unpreconditioned case (cross markers) or when using the preconditioner P4 (dot markers). We use $N = 512$, $r = 8$ and $\eta = \omega$. A theoretical estimate of the number of iteration steps in the unpreconditioned case is also added to the figure (circle markers).

of ω -values for which the constrained runs functional iteration converges (cf. Table 1). We again start from a random initial guess and use $r = 2$, $\rho = x(1 - x)$ and $N = 2048$.

Figure 6 (left) shows the two-norm of the residual as a function of the outer GMRES iteration number, now for the preconditioning variants P1, P2 (with $\eta = 0.1$) and P4 (with $\eta = 0.01$). Remarkably, we observe that the number of outer GMRES iteration steps is now smallest when using the cheapest preconditioner P4 (no additional matrix-vector products are required); in this case the number of iteration steps is reduced by a factor of about 20. The CPU-time is even reduced much more. This indicates that the overall cost is dominated by the quadratically increasing work and storage requirements of the Arnoldi orthogonalization when the Krylov basis grows, rather than by the number of LBM calls through the matrix-vector products. By restarting or truncating the Arnoldi orthogonalization [24], the complexity of the problem can be lowered substantially. Figure 6 (left) shows the convergence histories when using GMRES(20), which restarts after each 20 outer iteration steps (the restarting value has only a minor influence on the results). Although restarting results in a (slight) increase of the number of iteration steps, the CPU-time is decreased substantially, especially in the unpreconditioned case. As a result, the effect of the preconditioner on the CPU-time is now directly proportional to the reduction in the number of outer iteration steps.

Finally, Figure 6 (right) shows the eigenvalue spectra of A , M and $M^{-1}A$, when using the preconditioner P2. To avoid a heavily overloaded figure, only a representative subset of all eigenvalues is actually shown. Again, the left-most eigenvalues μ_i of A are well approximated by some of the eigenvalues of M . These eigenvalues, which are now lying extremely close to the origin, are again shifted to the right by applying the preconditioner. Together with the fact that the spectrum is also clustered more tightly and now fully lies in the right half-plane, this rationalizes the improved GMRES convergence rate.

5 Application: Lifting during Coarse Projective Integration

In this section, we extensively compare the constrained runs functional iteration to the constrained runs Newton-Krylov method. For this purpose, we use both methods for the lifting during coarse projective time integration [13, 7], when the microscopic simulator is the LBM

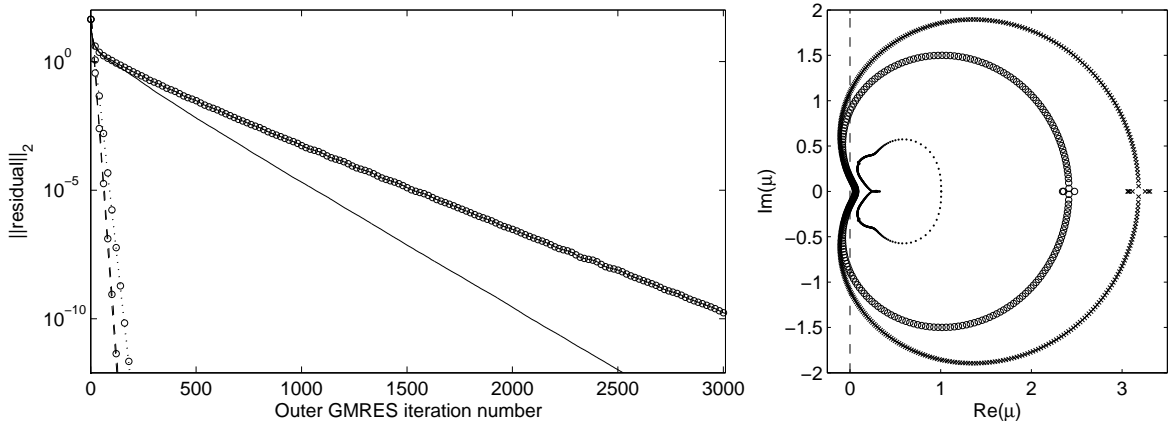


Figure 6: **Left:** the two-norm of the residual as a function of the outer GMRES iteration number, when using full GMRES or GMRES(20) (the circle markers indicate the restart points). In both cases, we use the preconditioning variants P1: without preconditioner (full), P2: with the preconditioner with exact coarse system solve (dotted), or P4: with the preconditioner without coarse system solve (dashed). For P2 and P4, the restarted GMRES convergence histories are nearly indistinguishable from the unrestarted ones. **Right:** a subset of the eigenvalues of A (crosses), M (circles) and $M^{-1}A$ (dots), when using the preconditioner P2.

from Section 2. As we are only interested here in the numerical issues in the lifting step, we use the simplest possible setup. Specifically, we use the LBM with $N = 128$, $\lambda = 100$ and we start at $t = 0$ from the density field $\rho = x(1 - x)$ and zero momentum and energy. First, we run the LBM model until $t = 70\Delta t$. Then, we extract the density field at $t = 60\Delta t$ and $t = 70\Delta t$ and use a linear extrapolation forward in time to predict the density field at $t = 100\Delta t$. After this extrapolation step, the LBM needs to be reinitialized so that it is both consistent with the extrapolated density field and close to the slow manifold; to this end we use the functional iteration in contrast to the Newton-Krylov method.

As in the previous sections, we restrict ourselves to the case $m = 0$ or $m = 1$. Both the functional iteration and the Newton iteration are used until the two-norm of the nonlinear residual has become smaller than the tolerance $\text{tol} = 10^{-8}$. The linear systems are solved with GMRES until the two-norm of the relative residual has become smaller than the same tolerance tol . If we use a less strict tolerance for GMRES, the number of Newton iteration steps is increased so that the overall amount of work (i.e., the total number of LBM steps) remains roughly the same or increases. For GMRES, we use a zero initial guess in all cases. Both the functional iteration and the Newton method itself also start from a zero initial guess, except for the method in the second column in Table 3, where we used a uniformly distributed random initial guess. As in the previous sections, we added these results to illustrate the worst-case convergence scenario. Unlike in the previous sections, we now do not have the “exact” higher order moments available to compare our results to. Therefore, we will approximate the exact slaved state by the solution obtained with the more accurate constrained runs Newton-Krylov method with $m = 4$.

Table 3: Accuracy and efficiency (the latter shown in parentheses) of the constrained runs functional iteration (FI), the constrained runs Newton-Krylov method (NK) starting from both a random and a zero initial guess, and the preconditioned constrained runs Newton-Krylov method (PNK), when $m = 0$ and as a function of ω .

ω	FI	NK (random)	NK	PNK
0.1	4.3e-02 (171)	4.3e-02 (157)	4.3e-02 (124)	4.3e-02 (45)
0.2	6.3e-02 (83)	6.3e-02 (79)	6.3e-02 (63)	6.3e-02 (34)
0.3	2.6e-02 (53)	2.6e-02 (53)	2.6e-02 (35)	2.6e-02 (27)
0.4	1.3e-02 (37)	1.3e-02 (38)	1.3e-02 (23)	1.3e-02 (22)
0.5	7.3e-03 (28)	7.3e-03 (30)	7.3e-03 (16)	7.3e-03 (18)
0.6	4.5e-03 (21)	4.5e-03 (23)	4.5e-03 (12)	4.5e-03 (15)
0.7	3.0e-03 (17)	3.0e-03 (19)	3.0e-03 (9)	3.0e-03 (13)
0.8	2.1e-03 (13)	2.1e-03 (15)	2.1e-03 (7)	2.1e-03 (10)
0.9	1.5e-03 (9)	1.5e-03 (11)	1.5e-03 (7)	1.5e-03 (8)
1.0	1.1e-03 (2)	1.1e-03 (4)	1.1e-03 (4)	1.1e-03 (5)
1.1	7.7e-04 (9)	7.7e-04 (11)	7.7e-04 (7)	7.7e-04 (8)
1.2	5.7e-04 (13)	5.7e-04 (15)	5.7e-04 (7)	5.7e-04 (9)
1.3	4.2e-04 (17)	4.2e-04 (18)	4.2e-04 (7)	4.2e-04 (10)
1.4	3.1e-04 (22)	3.1e-04 (22)	3.1e-04 (8)	3.1e-04 (12)
1.5	2.2e-04 (29)	2.2e-04 (28)	2.2e-04 (8)	2.2e-04 (13)
1.6	1.5e-04 (39)	1.5e-04 (36)	1.5e-04 (9)	1.5e-04 (15)
1.7	1.0e-04 (56)	1.0e-04 (49)	1.0e-04 (9)	1.0e-04 (17)
1.8	6.0e-05 (88)	6.0e-05 (74)	6.0e-05 (10)	6.0e-05 (22)
1.9	2.7e-05 (185)	2.7e-05 (140)	2.7e-05 (23)	2.7e-05 (35)

5.1 The case $m = 0$

The results for the case $m = 0$ are shown in Table 3. We computed, for various values of ω and for the functional iteration, the unpreconditioned Newton-Krylov method and the preconditioned Newton-Krylov method, the accuracy (the two-norm of the difference of the vector (ϕ, ξ) and the “exact” solution) and the efficiency (the total number of LBM function calls — here 1 per GMRES iteration step).

Due to the linearity of the problem ($\lambda = 100$ but $m = 0!$), only one Newton iteration step is required to lower the residual of the fixed point problem (10) by about eight orders of magnitude. We see that the overall error is in all cases significantly larger than `tol` (it includes the approximation error of the constrained runs procedure itself) and hence effectively the same for the different solution methods for each value of ω . This implies that we are oversolving the problem, and that it certainly makes no sense to perform a second Newton step. For our example, the number of LBM calls could, depending on the value of ω , even be lowered to about half to a quarter of its current value by ending the GMRES iteration prematurely when the overall error no longer decreases (while the residual does). Of course, the overall error is not known in practice, and should therefore be estimated. At least for our example, such an error estimator can be based on the relative change of the remaining full model variables compared to the change of the observable variables (during the LBM steps in each of the matrix-vector products).

Another important observation is that the error decreases as ω is increased. Intuitively, this can be explained by the fact that for a constant grid spacing Δx , increasing ω corresponds to a decreasing Δt , which is beneficial for the accuracy of the forward difference approximation of the “ $(m + 1)$ -st time derivative condition”. A more rigorous argument follows from the convergence analysis of the constrained runs functional iteration with $m = 0$ in the context of LBMs, given in [30]. There it was shown that the resulting error in the energy ξ is of second order in Δx , while the error in the momentum ϕ is of third order in Δx . Hence, the overall error is dominated by the error in the energy, and may be approximated by

$$e \approx \left(\frac{1}{2\omega} - \frac{1}{6\omega} \right) \frac{\partial \rho}{\partial t} \Delta t = \frac{1}{3\omega} \frac{\partial \rho}{\partial t} \Delta t = \frac{\Delta x^2 (2 - \omega)}{9\omega^2} \frac{\partial \rho}{\partial t} \sim \frac{2 - \omega}{\omega^2}. \quad (19)$$

This is indeed the dependency on ω seen in Table 3. The minor discrepancy that is still present is due to the fact that $t = 100\Delta t$ (and therefore $\partial \rho / \partial t$ at that time) is slightly different when using different values of ω .

We also observe that the number of LBM calls needed for the functional iteration and for the unpreconditioned Newton-Krylov method are roughly the same when a random initial guess is used, as expected from the worst case estimate (17). However, the unpreconditioned Newton-Krylov solver is clearly more efficient when a zero initial guess is used. Also, the efficiency gain is larger for larger values of ω . The reason for this is that, after the initial evolution phase of 70 LBM steps, the fastest damping high-frequency modes in the density profile have died out. When a zero initial guess is used (but clearly not when a random initial guess is used), the residual then also mainly consists of the low-frequency modes. Therefore, GMRES will converge faster than predicted by the worst-case convergence estimate (17), as the “effective” eigenvalue spectrum is considerably smaller than the “full” spectrum. We typically observe faster convergence in the first few iteration steps, after which the convergence slows down towards the rate (17). Remember that the eigenvalues of the Jacobian matrix A lie on a circle with center point 1 and radius $|1 - \omega|$, and that only for $\omega > 1$ the left-most eigenvalues near $\mu = \omega$ correspond to the high-frequency modes. If $\omega > 1$, the effective spectrum therefore lies considerably further away from the origin, explaining the even faster convergence in that case. In some cases, the coarse projective time integration itself may damp the high-frequency eigenmodes even more and more as it proceeds, yielding increasingly fast GMRES convergence rates in the lifting steps arising along the time integration process. This is for instance clearly seen in the linear LBM case. Important to note also is that the fast damping of the high-frequency modes does not have any influence on the efficiency of the functional iteration. This makes clear that, also for many other applications, an additional efficiency gain may arise because of the fact that the convergence rate of the functional iteration and the Krylov method depend in a different way on the eigenvalue spectrum of the Jacobian matrix A .

As in Section 4.3.2, we observe that for $\omega \rightarrow 0$, the preconditioner (P4 with $A_c = I$, $r = 8$ and η determined using the η -estimator from Section 4.3.2) prevents a too drastic increase of the number of LBM calls. We again observe that the preconditioner should not be used if $\omega > 1$, as the number of LBM calls is then even slightly augmented. Fortunately, the preconditioner is not really required in this case (when a zero initial guess is used).

When comparing the results of the functional iteration to those of the preconditioned Newton-Krylov method in Table 3, we see a clear advantage in using the latter approach.

Table 4: Accuracy and efficiency (the latter shown in parentheses) of the constrained runs functional iteration (FI), the constrained runs Newton-Krylov method (NK) and the preconditioned constrained runs Newton-Krylov method (PNK), when $m = 1$ and as a function of ω . The infinity symbol ∞ is used when the method is unstable.

ω	FI	NK	PNK
0.1	∞ (∞)	1.7e-02 (518)	1.7e-02 (268)
0.2	∞ (∞)	1.3e-03 (492)	1.3e-03 (120)
0.3	∞ (∞)	1.3e-04 (472)	1.3e-04 (64)
0.4	∞ (∞)	5.8e-05 (310)	5.8e-05 (40)
0.5	∞ (∞)	2.6e-05 (134)	2.6e-05 (30)
0.6	∞ (∞)	1.2e-05 (64)	1.2e-05 (24)
0.7	5.8e-06 (56)	5.8e-06 (36)	5.8e-06 (18)
0.8	3.0e-06 (38)	3.0e-06 (20)	3.0e-06 (16)
0.9	1.6e-06 (24)	1.6e-06 (14)	1.6e-06 (14)
1.0	8.9e-07 (4)	8.9e-07 (8)	8.9e-07 (10)
1.1	5.0e-07 (26)	5.0e-07 (14)	5.0e-07 (16)
1.2	2.8e-07 (50)	2.8e-07 (16)	2.8e-07 (20)
1.3	∞ (∞)	1.6e-07 (16)	4.0e-07 (42)
1.4	∞ (∞)	9.0e-08 (24)	1.1e-07 (84)
1.5	∞ (∞)	9.7e-08 (200)	1.3e-07 (218)
1.6	∞ (∞)	7.5e-08 (202)	1.1e-07 (270)
1.7	∞ (∞)	3.6e-08 (242)	1.1e-07 (278)
1.8	∞ (∞)	5.0e-08 (236)	1.5e-07 (280)
1.9	∞ (∞)	2.8e-07 (466)	3.1e-07 (468)

5.2 The case $m = 1$

The results for the case $m = 1$ are shown in Table 4. Again, we computed the accuracy and the efficiency of the same methods as before, now always starting from a zero initial guess. In the preconditioned case, we use $r = 2$. Note that there are now two LBM calls per GMRES iteration step in this case.

When comparing Table 4 to Table 3, we see that the error has decreased at the expense of an increase in computational effort. Also, the functional iteration can no longer be used for a large range of ω -values, as it is now unstable (cf. Table 1). The error is again in most cases larger than `tol`, so a second Newton iteration step is of little use. Only for $\omega \approx 2$ the error could be decreased slightly further by adding an second Newton step (in which the linear system should of course not be solved with the same stringent tolerance as before).

Most of the other observations we made in the case $m = 0$ also hold for the case $m = 1$: the error decreases as ω is increased, the Newton-Krylov method is more efficient than the functional iteration due to the damping of the high-frequency eigenmodes in the residual when using a zero initial guess, and only for values of $\omega < 1$ the preconditioner prevents a too drastic increase of the number of LBM calls. However, the beneficial effect of the preconditioner is now observed for all values of $\omega < 1$, which was clearly not the case when $m = 0$. Moreover, the efficiency gain is now also larger (and if N were larger, it would even be much larger, as was shown in Figure 6).

Even more than in the case $m = 0$, we may conclude that there is a clear advantage in using the Newton-Krylov approach over the functional iteration, as it can be used for a much broader range of ω -values and is faster in all cases where both methods can be applied.

6 On the Accuracy of the Constrained Runs Scheme if $m > 0$

In Figure 1, the error decreased down to machine precision as m was increased, as predicted by the theoretical results in [6, 33]. In this case, we approximated the error by comparing the values ϕ_m and ξ_m (computed with the constrained runs Newton-Krylov method for a certain value of m) to the “exact” slaved values that came from a long LBM simulation of 1000 steps. In Section 5, we approximated the error by comparing the values ϕ_m and ξ_m with ϕ_4 and ξ_4 , which are believed to be more accurate if $m < 4$. The approximate error obtained in this manner is virtually the same as the error shown in Figure 1. Note that this second approach does not require an initial simulation phase over for instance 1000 LBM steps. Without this initial simulation however, the error estimate does not decrease when m is increased.

Extensive numerical experiments led to the following explanation: without the initial LBM simulation, the density field $\rho = x(1 - x)$ contains fast damping high-frequency modes, which evolve at time scales that are comparable with the fast time scales of the slaving process itself. After initializing with ρ and any higher order moments ϕ and ξ , the LBM variables will therefore always evolve in a rather “erratic” (and if $\omega > 1$ oscillatory) manner during the first few LBM steps. As a consequence, the constrained runs solution ϕ_m and ξ_m will also contain fast damping high-frequency modes, especially when larger values of m are used. As these high-frequency components in ϕ_m and ξ_m dominate the overall error, the error estimate does not decrease when m is increased. Note that this is not in contradiction with the theoretical results in [30, 33], as sufficient smoothness is assumed in the derivation of these results, for instance to justify the Taylor expansions. Therefore, the theoretical estimates in [30, 33] only apply to the low-frequency error components of the error. An important consequence of this is that, although the errors are initially of the same magnitude for different values of m , running the LBM afterwards causes a trajectory that started from ρ , ϕ_m and ξ_m to become much more accurate after all when m is large (as the high-frequency error components that initially dominated the error quickly damp as time evolves). Even if we started from a smooth ρ , running the LBM afterwards (for a short time of the order the time scales of the slaving) may be beneficial for the accuracy of the trajectory as we then evolve further towards the slow manifold.

Although the error was estimated as $\phi_m - \phi_4$ in Section 5, the error did decrease substantially by increasing m from 0 to 1. The reason is that the fastest components of the density were already far more damped by the earlier part of the time integration process than they were amplified by the extrapolation forward in time.

The fact that high-frequency components of ρ evolve at time scales comparable with the fast time scales of the slaving process may suggest a scheme where the observable is not the complete spatially discretized density ρ but only its low-frequency components. The remaining full model variables then involve ϕ and ξ as well as the high-frequency components of ρ (which are not computed in a physically relevant manner anyway).

7 Conclusion

In this paper, we developed an efficient Newton-Krylov implementation of the constrained runs scheme [10, 6] for initializing on a slow manifold. Given a full model simulator and the values of the observables u_0 , this scheme approximates the values of the remaining full model variables v so that the full model state (u_0, v) is, up to some accuracy, close to the slow manifold. In [10, 6], the scheme was formulated as a functional iteration. In this paper, we replaced the functional iteration by a (preconditioned) Newton-Krylov solver. Alternatively, the Recursive Projection Method [26] could also be used to stabilize the functional iteration, as was outlined in [34].

The potential advantage of the Newton-Krylov method over the functional iteration was illustrated using a lattice Boltzmann model (LBM) for a nonlinear, one-dimensional reaction-diffusion system as the full model. Depending on the LBM parameters, the constrained runs functional iterations may converge slowly or even diverge. We showed that both issues are largely resolved by using the constrained runs Newton-Krylov method, especially when a coarse grid correction preconditioner is incorporated. This preconditioner was studied extensively for the LBM model problem, but the approach seems sufficiently general to be useful also in the context of other problems.

The work in this paper can be extended in several directions. The current Newton-Krylov implementation can for instance be further optimized by taking into account the suggestions in Section 3.1 for finding a good initial condition. A related idea is to “recycle” subspace information from previous linear systems [20] in the Newton iteration, or even from previous instances in time, when for example used in the context of equation-free coarse projective time integration. Another interesting research direction is to compare our Newton-Krylov approach to the Recursive Projection Method.

Acknowledgements

It is a pleasure to acknowledge Wim Vanroose, Giovanni Samaey and Pieter Van Leemput for several interesting discussions about various aspects of this work. This paper presents research results of the Belgian Network DYSCO (Dynamical Systems, Control, and Optimization), funded by the Interuniversity Attraction Poles Programme, initiated by the Belgian State, Science Policy Office. The research was also funded by the Research Council of the K.U.Leuven (Project OT/03/34). The work of IGK was partially supported by DARPA and by the US DOE (CMPD).

References

- [1] G. Browning and H.-O. Kreiss. Problems with different time scales for nonlinear partial differential equations. *SIAM Journal on Applied Mathematics*, 42(4):704–718, 1982.
- [2] B. Chopard, A. Dupuis, A. Masselot, and P. Luthi. Cellular automata and lattice Boltzmann techniques: An approach to model and simulate complex systems. *Advances in Complex Systems*, 5(2/3):103–246, 2002.
- [3] J. Curry, S. E. Haupt, and M. E. Limber. Low-order models, initializations, and the slow manifold. *Tellus 47A*, pages 145–161, 1995.

- [4] S. P. Dawson, S. Chen, and G. D. Doolen. Lattice Boltzmann computations for reaction-diffusion equations. *Journal of Chemical Physics*, 98(2):1514–1523, 1993.
- [5] W. E. Analysis of the heterogeneous multiscale method for ordinary differential equations. *Communications in Mathematical Sciences*, 1(3):423–436, 2003.
- [6] C. W. Gear, T. J. Kaper, I. G. Kevrekidis, and A. Zagaris. Projecting to a slow manifold: Singularly perturbed systems and legacy codes. *SIAM Journal on Applied Dynamical Systems*, 4(3):711–732, 2005.
- [7] C. W. Gear and I. G. Kevrekidis. Projective methods for stiff differential equations: Problems with gaps in their eigenvalue spectrum. *SIAM Journal on Scientific Computing*, 24(4):1091–1106, 2003.
- [8] C. W. Gear and I. G. Kevrekidis. Telescopic methods for parabolic differential equations. *Journal of Computational Physics*, 187(1):95–109, 2003.
- [9] C. W. Gear and I. G. Kevrekidis. Computing in the past with forward integration. *Physics Letters A*, 321(5-6):335–343, 2004.
- [10] C. W. Gear and I. G. Kevrekidis. Constraint-defined manifolds: A legacy code approach to low-dimensional computation. *Journal of Scientific Computing*, 25(1):17–28, 2005.
- [11] S. S. Girimaji. Reduction of large dynamical systems by minimization of evolution rate. *Phys. Rev. Lett.*, 82:2282–2285, 1999.
- [12] C. T. Kelley. *Iterative Methods for Linear and Nonlinear Equations*, volume 16 of *Frontiers in Applied Mathematics*. Society for industrial and Applied Mathematics, 1995.
- [13] I. G. Kevrekidis, C. W. Gear, J. M. Hyman, P. G. Kevrekidis, O. Runborg, and C. Theodoropoulos. Equation-free, coarse-grained multiscale computation: Enabling microscopic simulators to perform system-level analysis. *Communications in Mathematical Sciences*, 1(4):715–762, 2003.
- [14] H.-O. Kreiss. Problems with different time scales for ordinary differential equations. *SIAM Journal on Numerical Analysis*, 16(6):980–998, 1979.
- [15] H.-O. Kreiss. Problems with different time scales. In J. H. Brackbill and B. I. Cohen, editors, *Multiple Time Scales*, pages 29–57. Academic Press, 1985.
- [16] S. L. Lee and C. W. Gear. Second-order accurate projective integrators for multiscale problems. *Journal of Computational and Applied Mathematics*, 201(1):258–274, 2007.
- [17] E. N. Lorenz. Attractor sets and quasi-geostrophic equilibrium. *J. Atmos. Sci.*, 37:1685–1699, 1980.
- [18] R. A. Nicolaides. Deflation of conjugate gradients with applications to boundary value problems. *SIAM Journal on Numerical Analysis*, 24(2):355–365, 1987.
- [19] A. Padiy, O. Axelsson, and B. Polman. Generalized augmented matrix preconditioning approach and its application to iterative solution of ill-conditioned algebraic systems. *SIAM Journal on Matrix Analysis and Applications*, 22(3):793–818, 2000.

- [20] M. L. Parks, E. de Sturler, G. Mackey, D. Johnson, and S. Maiti. Recycling Krylov subspaces for sequences of linear systems. *SIAM Journal on Scientific Computing*, 28(5):1651–1674, 2006.
- [21] D. J. Patil, B. R. Hunt, E. Kalnay, J. A. Yorke, and E. Ott. Local Low Dimensionality of Atmospheric Dynamics. *Physical Review Letters*, 86(26):5878–5881, 2001.
- [22] Y. H. Qian and S. A. Orszag. Scalings in diffusion-driven reaction $A + B \rightarrow C$: Numerical simulations by lattice BGK models. *Journal of Statistical Physics*, 81(1/2):237–253, 1995.
- [23] Y. Saad. A flexible inner-outer preconditioned GMRES algorithm. *SIAM Journal on Scientific and Statistical Computing*, 14:461–469, 1993.
- [24] Y. Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, second edition, 2003.
- [25] Y. Saad and M. H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal of Scientific and Statistical Computing*, 7(3):856–869, 1986.
- [26] G. M. Shroff and H. B. Keller. Stabilization of unstable procedures: The recursive projection method. *SIAM Journal on Numerical Analysis*, 30(4):1099–1120, 1993.
- [27] S. Succi. *The Lattice Boltzmann Equation for Fluid Dynamics and Beyond*, volume 252 of *Numerical mathematics and scientific computation*. Springer Series in Comput. Math. Oxford University Press, Great Clarendon Street, Oxford OX2 6DP, first edition, 2001.
- [28] L. N. Trefethen and D. Bau. *Numerical Linear Algebra*. Society for industrial and Applied Mathematics, 1997.
- [29] P. Van Leemput, K. Lust, and I. G. Kevrekidis. Coarse-grained numerical bifurcation analysis of lattice Boltzmann models. *Physica D: Nonlinear Phenomena*, 210(1–2):58–76, 2005.
- [30] P. Van Leemput, W. Vanroose, and D. Roose. Mesoscale analysis of the equation-free constrained runs initialization scheme. *Multiscale Modeling and Simulation*, 2007. Accepted.
- [31] C. Vandekerckhove and D. Roose. Accuracy analysis of acceleration schemes for stiff multiscale problems. *Journal of Computational and Applied Mathematics*, In press, 2007.
- [32] C. Vandekerckhove, D. Roose, and K. Lust. Numerical stability analysis of an acceleration scheme for step size constrained time integrators. *Journal of Computational and Applied Mathematics*, 200(2):761–777, 2007.
- [33] C. Vandekerckhove, P. Van Leemput, and D. Roose. Accuracy and stability of the coarse time-stepper for a lattice Boltzmann model. *Journal of Algorithms & Computational Technology*. Accepted., 2007.
- [34] A. Zagaris, C. W. Gear, T. J. Kaper, and I. G. Kevrekidis. Analysis of the accuracy and convergence of equation-free projection to a slow manifold. Submitted, May 2007.