University of Groningen

A method to reduce the spin-up time of ocean models

Bernsen, Erik; Dijkstra, Henk A.; Wubs, Fred W.

Published in:
Ocean modelling

DOI:
[10.1016/j.ocemod.2007.10.008](https://doi.org/10.1016/j.ocemod.2007.10.008)

IMPORTANT NOTE: You are advised to consult the publisher's version (publisher's PDF) if you wish to cite from it. Please check the document version below.

Document Version
Publisher's PDF, also known as Version of record

Publication date:
2008

[Link to publication in University of Groningen/UMCG research database](#)

Citation for published version (APA):

Bernsen, E., Dijkstra, H. A., & Wubs, F. W. (2008). A method to reduce the spin-up time of ocean models. *Ocean modelling*, 20(4), 380-392. <https://doi.org/10.1016/j.ocemod.2007.10.008>

Copyright

Other than for strictly personal use, it is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license (like Creative Commons).

The publication may also be distributed here under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license. More information can be found on the University of Groningen website: <https://www.rug.nl/library/open-access/self-archiving-pure/taverne-amendment>.

Take-down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Downloaded from the University of Groningen/UMCG research database (Pure): <http://www.rug.nl/research/portal>. For technical reasons the number of authors shown on this cover page is limited to 10 maximum.

Short communication

A method to reduce the spin-up time of ocean models

Erik Bernsen^{a,*}, Henk A. Dijkstra^a, Fred W. Wubs^b^a*Institute for Marine and Atmospheric Research Utrecht, Department of Physics and Astronomy, Utrecht University,
3584CC Utrecht, The Netherlands*^b*Department of Mathematics and Computer Science, University of Groningen, Groningen, The Netherlands*

Received 12 July 2007; received in revised form 23 October 2007; accepted 23 October 2007

Available online 22 November 2007

Abstract

The spin-up timescale in large-scale ocean models, i.e., the time it takes to reach an equilibrium state, is determined by the slow processes in the deep ocean and is usually in the order of a few thousand years. As these equilibrium states are taken as initial states for many calculations, much computer time is spent in the spin-up phase of ocean model computations. In this note, we propose a new approach which can lead to a very large reduction in spin-up time for quite a broad class of existing ocean models. Our approach is based on so-called Jacobian–Free Newton–Krylov methods which combine Newton’s method for solving non-linear systems with Krylov subspace methods for solving large systems of linear equations. As there is no need to construct the Jacobian matrices explicitly the method can in principle be applied to existing explicit time-stepping codes. To illustrate the method we apply it to a 3D planetary geostrophic ocean model with prognostic equations only for temperature and salinity. We compare the new method to the ‘ordinary’ spin-up run for several model resolutions and find a considerable reduction of spin-up time.

© 2007 Elsevier Ltd. All rights reserved.

1. Introduction

Once a large-scale ocean model code has been designed, the first long computations usually performed are aimed to determine an equilibrium state of the model under given forcing. The time-scale to reach equilibrium depends on the vertical diffusion of salinity and temperature and is given by D^2/κ_v where D is a characteristic depth of the ocean basin and κ_v is the vertical diffusivity. For typical values of $\kappa_v = 10^{-5} \text{ m}^2 \text{ s}^{-1}$ and $D = 10^3 \text{ m}$, this so-called spin-up time-scale is approximately 3000 years. When the ocean model uses an explicit time-stepping method, then numerical stability conditions (such as the CFL condition) pose limitations on the time step. In many cases, the computation of equilibrium solutions therefore consumes a lot of CPU time.

A classical method for reducing spin-up time is the method of distorted physics (Bryan, 1984). In this method, the model equations are adjusted such that the equilibrium solutions are unchanged but the slow processes in the deep ocean are artificially accelerated. This method has been implemented successfully in

* Corresponding author. Tel.: +31 30 2532978; fax: +31 30 2543163.

E-mail address: E.Bernsen@phys.uu.nl (E. Bernsen).

many existing ocean models but as there remain limitations on the time step in the deep ocean it may still take much CPU time.

Another method of reducing the spin-up time is based on exponential extrapolation (Klinger, 2000). A long period of the spin-up consists of an almost exponential decay of the temperature and salinity fields towards their equilibrium values. Repeatedly extrapolating salinity and temperature based on the assumption of exponential decay resulted in a reduction of spin-up time of approximately a factor two to three.

Recently, Khatiwala et al. (2005) presented a method for determining equilibrium states of passive tracers. In this method a transport matrix, the matrix representation of the linear advection–diffusion equation that govern the evolution of passive tracers, is computed by performing a number of time steps of the model for several tracer distributions. Once this transport matrix is determined, the equilibrium fields for passive tracers are obtained by solving a matrix equation. It was shown in Khatiwala et al. (2005) that a similar method can also be applied successfully to active tracers such as temperature and salinity.

With fully implicit ocean models (e.g. Dijkstra et al., 2001; Weijer et al., 2003; De Niet et al., 2007), one can take relatively large time steps. For example, in Dijkstra et al. (2001) it is shown that in the approach to equilibrium, time steps of 10–100 year can be taken such that the equilibrium state is quickly reached. The fully-implicit methods, however, have the drawback that an explicit Jacobian matrix of the model has to be available. The latter is not easily computed for existing explicit ocean models. In addition, sophisticated preconditioners are needed to solve the giant systems of linear equations which result from the Newton–Raphson method during an implicit time step.

In this paper, we present a method for reducing the spin-up time in explicit ocean models by combining elements from the fully-implicit approach with Jacobian–Free Newton–Krylov (JFNK) methods (Reisner et al., 2000, 2003; Knoll and Keyes, 2004; Knoll et al., 2005). Essentially we apply Newton’s method for obtaining an equilibrium solution to an explicit model but we solve this problem without having to construct the Jacobian matrix explicitly. Instead only matrix–vector products for the Jacobian are needed and these can be obtained from the explicit model.

The general methodology is presented in Section 2. Next we apply it to a relatively simple 3D planetary geostrophic model (Samelson and Vallis, 1997a,b), as described in Section 3.1. Details on the specific implementation of the JFNK method for this model are given in Section 3.2. In Section 4, we compare the spin-up results of the JFNK method with those of the explicit time-stepping method and in Section 5, we draw conclusions and discuss further applications of the JFNK method.

2. The JFNK method

After discretizing the governing equations in space, each ocean model can in general be cast into the following form:

$$\frac{d\vec{x}}{dt} = \vec{F}(\vec{x}, \mu), \quad (1)$$

where $\vec{x}$ is the state vector containing all prognostic variables on all the grid points and $\vec{F}(\vec{x}, \mu)$ is usually referred to as the residual. The parameter μ is a control parameter for the forcing of the model (wind stress, buoyancy flux) with $\mu = 0$ corresponding to no forcing and $\mu = 1$ corresponding to the desired forcing.

Traditionally the equilibrium solution is reached by integrating (1) forward in time until we are close enough to a steady state; so in fact we are solving the nonlinear equation

$$\vec{F}(\vec{x}, \mu) = 0. \quad (2)$$

As an alternative one can use Newton’s method for solving the system of Eq. (2). Here, we start from an initial guess $\vec{x}_0$ and apply the iteration

$$\vec{x}_{k+1} = \vec{x}_k + \delta\vec{x}_{k+1}. \quad (3)$$

In this equation, $\delta\vec{x}_{k+1}$ is satisfying

$$D_{\vec{x}_k} \vec{F} \delta\vec{x}_{k+1} = -\vec{F}(\vec{x}_k, \mu), \quad (4)$$

where $D_{\vec{x}_k} \vec{F}$ is the Jacobian matrix of $\vec{F}$ at $\vec{x}_k$. If we want to apply this method to an existing time-stepping code the residual $\vec{F}$ is available but the problem is that the Jacobian matrix $D_{\vec{x}_k} \vec{F}$ is in general not easily extracted.

In a Jacobian–Free Newton–Krylov (JFNK) method, the system (4) is solved using a Krylov method, for example, the GMRES (Saad, 1996, 2003) method. This is an iterative method in which at the l th iteration the solution of the system (4), more conveniently written as $A_k \delta \vec{x}_{k+1} = \vec{b}_k$ with $A_k = D_{\vec{x}_k} \vec{F}$ and $\vec{b}_k = -F(\vec{x}_k, \mu)$, is approximated with a vector $\delta \vec{x}_{k+1,l}$ from the subspace $\delta \vec{x}_{k,0} + K_l$ with the Krylov subspace given by

$$K_l = \langle \vec{r}_{k+1}, A_k \vec{r}_{k+1}, A_k^2 \vec{r}_{k+1}, \dots, A_k^{l-1} \vec{r}_{k+1} \rangle \quad (5)$$

and $\vec{r}_{k+1} = A_k \delta \vec{x}_{k+1,0} - \vec{b}_k$. The choice of $\delta \vec{x}_{k+1,l}$ is such that the 2-norm of the residual of the matrix equation $\|A_k \delta \vec{x}_{k+1,l} - \vec{b}_k\|_2$ is minimized. As the Krylov subspace is extended with one dimension at each iteration, the norm of the residual of the matrix equation is decreasing with iteration number and in that sense $\delta \vec{x}_{k,l+1}$ will be a better approximate solution than $\delta \vec{x}_{k,l}$. Since the construction of these Krylov subspaces only require matrix–vector products we don’t need the matrix A_k itself, but rather the matrix applied to a vector $\vec{v}$. With $A_k = D_{\vec{x}_k} \vec{F}$, the matrix vector product can be approximated using the finite difference approximation

$$D_{\vec{x}_k} \vec{F} \vec{v} = \frac{\vec{F}(\vec{x}_k + \epsilon \vec{v}) - \vec{F}(\vec{x}_k)}{\epsilon} + \mathcal{O}(\epsilon) \quad (6)$$

with $\epsilon = \epsilon_0(1 + \|\vec{x}_k\|_1/N)/\|\vec{x}_k\| \ll 1$, N the dimension of the vector $\vec{x}_k$, $\|\vec{v}\|_1 = \sum |v_i|$ and for example, $\epsilon_0 = 10^{-6}$.

For a fast convergence rate of the GMRES method a preconditioner may be needed. When using a preconditioner P^{-1} on a linear system $A\vec{x} = \vec{b}$, we solve the equivalent system

$$P^{-1}A\vec{x} = P^{-1}\vec{b}. \quad (7)$$

Requirements of a good preconditioner are that $P^{-1} \approx A^{-1}$ such that $P^{-1}A$ is well conditioned and that $P^{-1}\vec{b}$ is relatively easy to compute. The construction of a preconditioner is usually model dependent and hence we describe it in the following section after the presentation of our demonstration model.

3. Test case

In this section we describe the implementation of the JFNK method to the planetary geostrophic ocean model as developed in Samelson and Vallis (1995, 1997a,b). Details of the model are presented in Section 3.1 below; these are not only for convenience for the reader but they are also important for the description of the preconditioner in Section 3.2.

3.1. Planetary geostrophic model

The model we discuss here is a good prototype model since temperature and salinity are the only prognostic variables and it is the slow transport in the deep ocean of these quantities that causes the extremely long spin-up times. The geometry of the model consists of a rectangular basin of dimension $L_x \times L_y = 6000 \text{ km} \times 6000 \text{ km}$. The bottom is at a constant depth $D = 5000 \text{ m}$. In the interior, $-D < z < 0$, the evolution of temperature and salinity is governed by the advection–diffusion equations

$$\frac{\partial T}{\partial t} + \vec{u}_h \cdot \nabla_h T + w \frac{\partial T}{\partial z} = \frac{\partial}{\partial z} \left(\kappa_v \frac{\partial T}{\partial z} \right) + \kappa_h \nabla_h^2 T - \lambda \nabla_h^4 T, \quad (8a)$$

$$\frac{\partial S}{\partial t} + \vec{u}_h \cdot \nabla_h S + w \frac{\partial S}{\partial z} = \frac{\partial}{\partial z} \left(\kappa_v \frac{\partial S}{\partial z} \right) + \kappa_h \nabla_h^2 S - \lambda \nabla_h^4 S. \quad (8b)$$

In these equations, T is the temperature, S the salinity, $\kappa_h = 10^3 \text{ m}^2 \text{ s}^{-1}$ the horizontal Laplacian diffusivity, $\lambda \approx 0.5 \times 10^{14} \text{ m}^4 \text{ s}^{-1}$ the horizontal biharmonic diffusivity, κ_v the vertical diffusivity, $\vec{u}_h = [u, v]^T$ the horizontal velocity field, w the vertical velocity and $\nabla_h = [\partial/\partial x, \partial/\partial y]^T$ the horizontal gradient operator.

For the horizontal momentum equations the geostrophic balance with a linear friction term is used

$$f\vec{u}_h^\perp = -\frac{\nabla_h p}{\rho_0} - \epsilon_b \vec{u}_h \quad (9)$$

with $f = f_0 + \beta y$ the Coriolis parameter on a β -plane, $f_0 = 8.4 \times 10^{-5} \text{ s}^{-1}$, $\beta = 1.85 \times 10^{-11} \text{ m}^{-1} \text{ s}^{-1}$, p the pressure, $\epsilon_b = 0.42 \times 10^{-5} \text{ s}^{-1}$ the linear friction coefficient, $\rho_0 = 10^3 \text{ kg m}^{-3}$ a reference density and $\vec{u}_h^\perp = [-v, u]^T$. Finally, the system of equations is completed by the hydrostatic balance, the continuity equation and a linear equation of state

$$\frac{\partial p}{\partial z} = -g\rho, \quad (10a)$$

$$\nabla_h \cdot \vec{u}_h + \frac{\partial w}{\partial z} = 0, \quad (10b)$$

$$\rho = \rho_0(1 + \alpha_S(S - S_0) - \alpha_T(T - T_0)). \quad (10c)$$

Here, $g = 9.8 \text{ m s}^{-2}$ is the acceleration of gravity, $\alpha_T = 1.0 \times 10^{-3} \text{ K}^{-1}$ the coefficient of thermal expansion, $\alpha_S = 7.6 \times 10^{-4} \text{ psu}^{-1}$ the coefficient of saline contraction, and T_0 and S_0 are a reference temperature and salinity, respectively.

Lateral boundary conditions are given by no heat and salinity flux

$$\nabla_h(\kappa_h T - \lambda \nabla_h^2 T) \cdot \hat{n} = 0, \quad (11a)$$

$$\nabla_h(\kappa_h S - \lambda \nabla_h^2 S) \cdot \hat{n} = 0, \quad (11b)$$

with $\hat{n} = [n_x, n_y]^T$ the outward normal vector. A second pair of lateral boundary conditions is given by

$$\epsilon_b \nabla T \cdot \hat{n} + f \nabla T \cdot \hat{\tau} = 0, \quad (12a)$$

$$\epsilon_b \nabla S \cdot \hat{n} + f \nabla S \cdot \hat{\tau} = 0, \quad (12b)$$

with $\hat{\tau} = [-n_y, n_x]^T$ tangential to the boundary. Note that the biharmonic diffusion in (8) allows for two lateral boundary conditions for temperature and salinity and that the boundary conditions (12) ensure consistency with the lateral boundary conditions for velocity $\vec{u}_h \cdot \hat{n} = 0$. At the bottom we have the no-normal flow boundary condition ($w = 0$) and no heat and salinity flux.

The model is forced by a wind stress and a buoyancy flux applied to an upper explicit Ekman layer with a thickness $\delta_E = 25 \text{ m}$. The wind stress is given by

$$\vec{\tau} = \mu \left[0, \tau_0 \frac{f}{f_0} \sin(2\pi y/L_y) \right]^T, \quad (13)$$

with the maximum amplitude given by $\tau_0 = 10^{-1} \text{ Pa}$. The resulting horizontal and vertical velocity field in the Ekman layer are then given by $\vec{u}_{E,h} = \vec{\tau}^\perp / f \delta_E$ and $w_E = -\delta_E \nabla_h \cdot \vec{u}_{E,h}$ respectively. The vertical velocity in the Ekman layer acts as a boundary condition for the vertical velocity field in the interior. The horizontal velocity field is used in the prognostic equations for temperature and salinity in the Ekman layer which become

$$\frac{\partial T_E}{\partial t} + \nabla_h \cdot (\vec{u}_{E,h} T_E) = \frac{F_T - F_{T,i}}{\delta_E}, \quad (14a)$$

$$\frac{\partial S_E}{\partial t} + \nabla_h \cdot (\vec{u}_{E,h} S_E) = \frac{F_S - F_{S,i}}{\delta_E}. \quad (14b)$$

The heat and salt fluxes between the base of the Ekman layer and the interior are given by

$$F_{T,i} = \left[w_E T - 2\kappa_v \frac{\partial T}{\partial z} \right]_{z=0}, \quad (15a)$$

$$F_{S,i} = \left[w_E S - 2\kappa_v \frac{\partial S}{\partial z} \right]_{z=0}, \quad (15b)$$

while the restoring conditions are given by

$$F_T = -\tau_T^{-1}(T_E - T_s), \quad (16a)$$

$$F_S = -\tau_S^{-1}(S_E - S_s), \quad (16b)$$

where $\tau_T = \tau_S \approx 20$ days are the relaxation times for temperature and salinity. The surface temperature and salinity are chosen as

$$T_s = T_0 + \frac{\mu \Delta T \cos(\pi y/L_y)}{2}, \quad (17a)$$

$$S_s = S_0 + \frac{\mu \Delta S \cos(\pi y/L_y)}{2}, \quad (17b)$$

where the temperature and salinity difference between the northern and southern boundary given by $\Delta T = 25$ K and $\Delta S = 1$ psu.

Convective adjustment was implemented in Samelson and Vallis (1995) by eliminating all static instabilities instantaneously at the end of each time step. In order to apply the JFNK method it is necessary that the residual $\vec{F}$ can be extracted from the time-stepping code and that this residual is differentiable. Using the convective adjustment scheme as in Samelson and Vallis (1995) this is impossible and therefore we implemented it differently using a variable vertical diffusivity given by

$$\kappa_v = \begin{cases} \kappa_{v,0} & \text{if } N_b^2 \geq 0, \\ \kappa_{v,0} - \frac{\alpha \kappa_{v,C}^2 N_b^2}{1 - \alpha \kappa_{v,C} N_b^2} & \text{if } N_b^2 < 0, \end{cases} \quad (18)$$

with $\kappa_{v,0} = 10^{-4} \text{ m}^2 \text{ s}^{-1}$, $\kappa_{v,C} = 0.99 \times 10^{-2} \text{ m}^2 \text{ s}^{-1}$, the buoyancy frequency N_b given by

$$N_b^2 = -\frac{g}{\rho_0} \frac{\partial \rho}{\partial z},$$

and the scaling parameter $\alpha = 3.89 \times 10^3 \text{ s}^3 \text{ m}^{-2}$. Because of this alternative convective adjustment formulation, the time-stepping code had to be slightly modified. We used the second order Runge–Kutta scheme for both advection and diffusion where the vertical diffusion is treated implicitly to avoid a severe restriction on the time step due to convective adjustment.

To write the planetary geostrophic model in the form of (8) we introduce the dimensional state vector $\vec{x}' = (\vec{x}'_T, \vec{x}'_S)$ with $\vec{x}'_T$ and $\vec{x}'_S$ vectors containing the temperature and salinity at all gridpoints, respectively. Note that the velocity field is not contained in the state vector since it is diagnostic rather than prognostic. The dimensional residual is written as $\vec{F}' = (\vec{F}'_T, \vec{F}'_S)$, where $\vec{F}'_T$ and $\vec{F}'_S$ are obtained from the discretized version of (8a) and (8b), respectively. The JFNK and time-stepping method both use a dimensionless state vector and residual rather than the dimensional ones. Using scales of temperature and salinity variations given by $\hat{T} \approx 8.6 \times 10^{-3} \text{ K}$ and $\hat{S} \approx 1.1 \times 10^{-2} \text{ psu}$, respectively, and a time scale of $\hat{t} \approx 1.6 \cdot 10^2$ years, we obtain the dimensionless state vector $\vec{x} = (\vec{x}_T, \vec{x}_S) = (\vec{x}'_T/\hat{T}, \vec{x}'_S/\hat{S})$ and the dimensionless residual $\vec{F} = (\vec{F}_T, \vec{F}_S) = ((\hat{t}/\hat{T})\vec{F}'_T, (\hat{t}/\hat{S})\vec{F}'_S)$.

3.2. Preconditioner

It turns out that we need to use a preconditioner to speed-up the convergence rate of the GMRES method in the JFNK method. In our approach for the preconditioner, we first construct the Jacobian or an approximation thereof and store this in sparse format. From this we construct a preconditioner using MRILU (Botta and Wubs, 1999). Since the construction of the matrix in sparse format and the corresponding preconditioner is a relative expensive operation compared to applying GMRES iterations we choose to construct the preconditioner once, for the first initial guess for Newton's method and then reuse this preconditioner for subsequent Newton iterations.

For the construction of the Jacobian matrix (or approximation) we use a technique developed by Coleman et al. (1984). This technique is designed to evaluate a sparse matrix A using as few as possible matrix vector products $A\vec{v}$. Assume that the sparsity pattern of the matrix A is known. In (Coleman et al., 1984) a partition $C_1, C_2, \dots, C_p$ of the columns of A is found such that no pair of columns in the same C_q shares a non-zero element on the same row. Then by performing the matrix vector product $A\vec{v}$ with $v_i = 0$ if $i \notin C_q$ and $v_i = 1$ if $i \in C_q$ the columns C_q of the matrix A can be obtained. Since for each row of A which contains a non-zero element in column $k \in C_q$ and because no other column in C_q contains a non-zero element in this row, we have that

$$A_{ik} = \sum_{j \in C_q} A_{ij} v_j = \sum_j A_{ij} v_j = (A\vec{v})_j. \quad (19)$$

For the planetary geostrophic model, the residual for the temperature equation in gridpoint (i, j, k) depends on $T_{i,j,k}$, $S_{i,j,k}$, $T_{i\pm 1,j,k}$, $T_{i,j\pm 1,k}$, $T_{i\pm 2,j,k}$, $T_{i,j,k\pm 1}$ and $S_{i,j,k\pm 1}$ for the horizontal and vertical diffusion and for the advection terms it depends additionally on $u_{i\pm 1/2,j,k}$, $v_{i,j\pm 1/2,k}$ and $w_{i,j,k\pm 1/2}$. Since the velocity field is diagnostic, the velocities themselves depend on temperature and salinity. For instance, it turns out that the velocity $u_{i\pm 1/2,j,k}$ depends on $T_{i+1/2(1\pm 1),j,:}$ and $S_{i+1/2(1\pm 1),j,:}$, thus the velocities depend on whole columns of temperature and salinity. Due to these additional dependencies the number of partitions used for determining the Jacobian matrix increases dramatically and also the construction of the MRILU preconditioner becomes much more expensive. Hence we choose to base our preconditioner not on the true Jacobian, but on an approximation of the Jacobian in which the velocity field is assumed to be independent of T and S .

Formally we can write the residual as

$$\vec{F}(\vec{x}) = C(\vec{x})\vec{x} + R(\vec{x}), \quad (20)$$

with $C(\vec{x})\vec{x}$ the discretization of the advection terms and $R(\vec{x})$ the discretization of all other terms. The matrix $C(\vec{x})$ represents the discretized version of the advection operator $\vec{u}_h \nabla_h + w \partial / \partial z$, where the velocity fields $\vec{u}_h$ and w correspond to the state $\vec{x}$. Now instead of using the true Jacobian for the construction of the preconditioner we use the following approximation $J = C(\vec{x}) + D_{\vec{x}}R$. Once we obtain this approximate Jacobian J we use MRILU (Botta and Wubs, 1999) to construct an incomplete factorization of J such that

$$J \approx LU \quad (21)$$

with L and U lower and upper triangular matrices respectively. This factorization is not exact because during its construction some elements are dropped when they are too small according to the dropping criterion. A dropping tolerance parameter ϵ_d determines which elements are dropped, $\epsilon_d = 0$ resulting in an exact factorization of J . In Section 4 we investigate the dependence of the performance of the method on this parameter.

4. Results

Following the traditional forward time-marching approach, we start with homogeneous salt and temperature as initial condition, specify full forcing ($\mu = 1$) and integrate the model equations in Section 3.1 in time. In Fig. 1 the norm of the residual $\vec{F}$ is plotted as a function of physical time for resolutions $(N_x, N_y, N_z) = (16, 16, 18)$, $(32, 32, 18)$ and $(64, 64, 18)$. To make the results for all resolutions comparable we divide the norm of the residual, $\|\vec{F}(\vec{x})\|_2$, with the dimension, N , of the state vector $\vec{x}$. Further we note that the dimensionless residual is used, as explained at the end of Section 3.1. We see that at some point the residual starts to decrease exponentially, for the lowest resolution after approximately 800 years and for the higher resolutions after 1500 years. After 3500 years a reasonable accurate approximation of the equilibrium solution has been obtained. For instance for the lowest resolution the meridional overturning streamfunction differs at most 0.14 Sv from the solution obtained after 8000 years and the maximum in the meridional overturning streamfunction differs approximately 1.5×10^{-2} Sv.

It is the CPU time that we are interested in and for a spin-up run for 3500 years this is 0.14 h at a resolution of $(16, 16, 18)$, 1.76 h at a resolution of $(32, 32, 18)$ and 45.9 h at a resolution of $(64, 64, 18)$ (see also Table 1). We note that doubling the horizontal resolution increases the amount of CPU time needed per time step by

Fig. 1. The norm of the (non-dimensional) residual $\|\vec{F}\|/N$ as a function of model year for three resolutions, using the new convective adjustment scheme. For a spin-up run for 3500 years it takes 502 seconds at a resolution of (16, 16, 18), 1.76 h at a resolution of (32, 32, 18) and 45.93 h at a resolution of (64, 64, 18).

Table 1

A comparison of the performance of the code for the original and new convective adjustment scheme

Resolution	New time-stepping code		Original time-stepping code	
	CPU time (h)	Δt (h)	CPU time (h)	Δt (h)
(16, 16, 18)	0.14	124.6	0.10	76.1
(32, 32, 18)	1.76	31.8	0.91	24.9
(64, 64, 18)	45.93	4.4	14.43	5.9

For both schemes the total amount of CPU time for a spin-up run of 3500 years and the maximal time step are given.

approximately a factor four. However, since we are using an explicit time-stepping scheme we also have to use a smaller time step.

For the resolutions (16, 16, 18), (32, 32, 18) and (64, 64, 18) we used a time step of $\Delta t \approx 125$ hr, $\Delta t \approx 32$ hr and $\Delta t \approx 4$ hr, respectively and we note that the very small time steps at the highest resolution (64, 64, 18), approximately 8 times smaller than at a resolution (32, 32, 18), are probably caused by the inclusion of biharmonic diffusion leading to a scaling of the time step $\Delta t \propto \Delta x^{-4}$ as $\Delta x \rightarrow 0$, corresponding to a reduction in time step of a factor 16 when doubling the grid resolution.

When comparing the maximum of the meridional overturning streamfunction for different resolutions we see that it increases with 0.27 Sv and 1.67 Sv when going from a resolution of (32, 32, 18) to (64, 64, 18) and (16, 16, 18) to (32, 32, 18) respectively. This suggests that we have at least quadratic convergence and confirms that we used a second order space discretization for the numerical model.

It was noted in Section 3.1 that a new convective adjustment scheme was implemented. To compare the performance of the new and old time-stepping code we consider a spin-up run of 3500 years. In Table 1 the total amount of CPU time for this spin-up run and the maximal time step for which the scheme is stable are shown. At a low resolution of (16, 16, 18) the new code is approximately a factor 1.4 times slower while for the highest resolution of (64, 64, 18) it turns out that the time step restriction for the new code is more severe than for the original code resulting in a slowdown of up to a factor 3.2.

The zonally averaged density field ($\rho - \rho_0$) and the meridional overturning streamfunction of the equilibrium state for the (64, 64, 18) case and the new convective adjustment scheme are plotted in Fig. 2a and b, respectively. From the density plot, it is seen that the new convective adjustment scheme results in stably stratified solutions. The maximum of the meridional overturning streamfunction has a reasonable value of

Fig. 2. Results of a spin-up run, using the new convective adjustment scheme, for a resolution of (64, 64, 18). The zonally averaged density ($\rho - \rho_0$) in (a) and the meridional overturning streamfunction in (b).

$\max \psi_M \approx 14.23$ Sv which is located at about 1000 m depth. The signature of the wind-driven Ekman circulation can be seen at the surface. We note that using the old convective adjustment scheme course results in slightly different solutions, but the spatial pattern of these solutions is similar to those in Fig. 2. For instance the location of the maximum in the streamfunction is at exactly the same gridpoint, but the maximum itself has a value of 13.50 Sv instead of 14.23 Sv. We now apply the JFNK method discussed in Section 2. As an initial guess for the GMRES iterations we use $\delta \tilde{x}_{k+1,0} = 0$ and we stop after a maximum of 50 iterations, with

a restart after 25 iterations. It is well known that Newton's method guarantees local convergence to a solution of the system, but no global convergence. One way to improve the global convergence of Newton's method is by using the minimum reduction method (Eisenstat et al., 1994). Instead of using the update $\vec{x}_{k+1} = \vec{x}_k + \delta\vec{x}_{k+1}$ immediately we check if the norm of the residual is improved enough after applying the Newton update, i.e., we check

$$\|\vec{F}(\vec{x}_k + \delta\vec{x}_{k+1})\|_2 < \eta \|\vec{F}(\vec{x}_k)\|_2, \quad (22)$$

where we took $\eta = 0.7$. If (22) is not satisfied, we take a shorter Newton step by setting $\delta\vec{x}_{k+1} := \theta\delta\vec{x}_{k+1}$ with $\theta = 0.95$. If after 50 steps we still fail to satisfy (22) then we continue with the residual as it is.

In addition to this minimum reduction method we also use a continuation method to improve global convergence. We know that the equilibrium solution for the unforced system

$$\vec{F}(\vec{x}_0, \mu_0) = 0 \quad (23)$$

with $\mu_0 = 0$ is given by $T = T_s$ and $S = S_s$. We now increase μ in steps $\Delta\mu$ each time solving the system

$$\vec{F}(\vec{x}_{k+1}, \mu_{k+1}) = 0 \quad (24)$$

with an initial guess for $\vec{x}_{k+1}$ the solution at the previous step, $\vec{x}_k$. These systems need not be solved very accurately, except for the last step when $\mu = 1$.

In Fig. 3 we plot the norm of the residual $\vec{F}$ as a function of CPU time for the same resolutions $(N_x, N_y, N_z) = (16, 16, 18), (32, 32, 18)$ and $(64, 64, 18)$. Clearly the JFNK method is much faster than the time-stepping method. The residual remains fairly level in the beginning before starting to decrease exponentially. This is due to the fact that at the beginning we need to construct the preconditioner which is the relatively expensive operation within the JFNK method. Then μ is increased slowly towards $\mu = 1$ and only then we see the residual starting to decrease significantly. We note that the solution obtained after 3500 years of time-stepping and the one obtained from the JFNK method differ at most 1.3×10^{-2} K, 2.8×10^{-3} K and 5.5×10^{-3} K in the temperature field for resolutions $(16, 16, 18), (32, 32, 18)$ and $(64, 64, 18)$ respectively, while for the salinity field these values are 5.3×10^{-4} psu, 1.1×10^{-4} psu and 2.2×10^{-4} psu. This indicates that both method are approaching the same equilibrium solution. There are several parameters affecting the performance of the JFNK method. Here, we will only focus on the step size $\Delta\mu$ with which the forcing is increased and the droptol parameter ϵ_d used during the construction of the preconditioner. In Fig. 4a, the value of μ as a function of CPU time is plotted for several step sizes ranging from 2.5×10^{-2} to 2.5×10^{-1} . These runs were

Fig. 3. The norm of the (non-dimensional) residual $\|\vec{F}\|/N$ as a function of CPU time for three resolutions when using the JFNK method.

Fig. 4. In (a) the forcing parameter μ is plotted as a function of CPU time during the JFNK method for several fixed values of $\Delta\mu$ and for a resolution of (16,16,18). In (b) the CPU time as a function of ϵ_d , scaled such that the maximal CPU time corresponds to 1, is plotted for three resolutions.

all done for the lowest resolution (16,16,18) using $\epsilon_d = 10^{-3}$, default values for all other parameters of the MRILU preconditioner and the convergence criterium $\|\vec{F}(\vec{x}_k, \mu_k)\|/N < \epsilon_k$ with N the dimension of the residual vector and with $\epsilon_k = 10^2$ when $\mu_k < 1$ and $\epsilon_k = 10^{-4}$ when $\mu_k = 1$. In all cases the method converges, but for very small stepsizes performance degrades because too many steps have to be taken and for too large stepsizes the computational cost for performing one step increases.

Now we fix the step size to $\Delta\mu = 0.1$ and consider the performance of the method as a function of the drop-tol parameter ϵ_d mentioned in Section 3.2. For the choice of ϵ_d a trade-off has to be made between the CPU time needed for constructing and applying the preconditioner and for the total number of GMRES iterations needed for the algorithm to converge. For instance, when the value of ϵ_d is increased, less time is needed to construct and apply the preconditioner, resulting in cheaper GMRES iterations. But on the other hand, the total number of GMRES iterations needed for the convergence of the algorithm is increased and for too high values of ϵ_d the algorithm may even fail to converge.

Depending on resolution we varied ϵ_d over a range $10^{-5} \leq \epsilon_d \leq 10^{-2}$ for the lowest resolution, $10^{-6} \leq \epsilon_d \leq 10^{-3}$ for a resolution of (32,32,18) and $10^{-7} \leq \epsilon_d \leq 10^{-4}$ for the highest resolution. In Fig. 4b

Fig. 5. Speed-up of the JFNK method compared to the time-stepping method for three resolutions. On the horizontal axis the physical time of the time-stepper is shown, which is a measure for how accurate the steady state is approached. On the vertical axis the speed-up that the JFNK method gives for reaching an equilibrium solution of the same accuracy as the time-stepper is shown.

the CPU times are plotted, scaled such that the maximal CPU time for each resolution corresponds to 1. Optimal values are given by $\epsilon_d \approx 1.3 \times 10^{-3}$, $\epsilon_d \approx 1.3 \times 10^{-4}$ and $\epsilon_d \approx 3 \times 10^{-6}$, respectively.

Finally we consider the speedup achieved by the JFNK method in comparison with the time-stepping method and the result is plotted in Fig. 5. The amount of physical time for which we let the time-stepper run is given on the horizontal axis. This is a measure for the accuracy with which the time stepper has approached the equilibrium solution. On the vertical axes the speedup achieved by the JFNK method to reach the same accuracy as the time-stepping method is given. For a typical spin-up run of 3500 year and for the lowest to highest resolution the JFNK method is approximately a factor 10, 14 and 87 faster.

5. Summary, discussion and conclusion

Using a planetary geostrophic ocean model as test case, we showed that a significant speed-up in spin-up time could be obtained by applying the JFKN method. In particular at higher resolutions the time step restriction of the explicit time-stepping method causes a huge speedup for the JFNK method. In Section 3.1 it was mentioned that necessary changes in the code resulted in a 3.2 times slower performance at the highest resolution, but even then the JFNK method is approximately 27 times faster than the original time-stepping code.

The results here show the potential for applying this method to more realistic ocean models. The basic JFNK framework is very general and only requires the availability of a smooth residual $\vec{F}$ from the time-stepping code. Although we saw that in this relatively simple model there were already some changes required with respect to convective adjustment, in general the residual will be available without major modifications of the time-stepping code.

In Section 3.2 we discussed a preconditioner for the planetary geostrophic model. In principle one could try to apply the preconditioning technique discussed in Section 3.2 to any ocean model. The only requirement is the availability of the sparsity pattern of the Jacobian matrix, which can be obtained by carefully analyzing the time-stepping code. However, for more realistic models at higher resolutions and with irregular domains the Jacobian is much more ill-conditioned and this preconditioner may not be efficient. In this case there is the option to construct the preconditioner not only once before the first Newton iteration, but more often whenever GMRES is converging too slow. Also it might be necessary to develop a preconditioning technique that is more dedicated to the particular model used. The continuation method used for global convergence of the JFNK method worked very well in the planetary geostrophic model, but in more realistic models there is the possibility of encountering bifurcations points which have to be treated separately.

Another problem not dealt with in the JFNK method is the assumption that a true equilibrium state is reached instead of a statistical equilibrium. In these statistical kind of equilibria the condition (2) does not hold at any point in time, but only in a time averaged sense. One approach of this problem would be to require that the time-averaged residual approaches zero, implying that the time-averaged temperature and salinity approach some constant, and hence we solve, instead of (2), the new time-averaged system $\bar{G}(\bar{x}) = 0$ with $\bar{G}(\bar{x}) = (1/\mathcal{T}) \int_0^{\mathcal{T}} \bar{F}(\bar{x}(t)) dt$ with $\mathcal{T}$ an averaging period. However, in this approach the averaging period $\mathcal{T}$ cannot be too long for computational efficiency and a more sophisticated preconditioner has to be used since the Jacobian of the new residual $\bar{G}$ may no longer be sparse. Related to the problem of statistical equilibria are periodic solutions corresponding to a periodic (seasonal) forcing. To find these the approach taken in van Noorden et al. (2003) can be used.

When applying the JFNK method to realistic ocean models another important issue is how efficient the method can be implemented on parallel computers. The GMRES method can be parallelized very well and the same holds for the calculation of the residual $\bar{F}$, provided that an efficient parallel implementation of the original time-stepping code exists. No parallelized version of the MRILU preconditioner exists yet. However, this does not have to be a problem since for more realistic models the preconditioner is likely to be replaced with a more efficient one anyway.

A comparison of the JFNK method with other acceleration methods is not within the scope of this paper. Our next step is to apply this method to a more realistic ocean model such as POP (Smith and Gent, 2002) or MOM (Griffies et al., 2004) and then make a comparison with other acceleration methods, such as the distorted physics method (Bryan, 1984). Some changes in POP or MOM are probably necessary due to the requirement that the residual $\bar{F}$ is smooth. In POP smoothness of the residual with respect to convective adjustment is easily obtained since convection is optionally handled by setting κ_v to high values where static instabilities occur, but the use of partial bottom cells potentially causes problems since it results in a non-differentiable residual. Furthermore the momentum equations in POP are split into a barotropic part, treated implicitly and baroclinic part, treated explicitly. Although this splitting makes it harder to extract the residual from the time-stepping code directly, it also offers an opportunity to possibly use a physics based preconditioner (Nadiga et al., 2006; De Niet et al., 2007).

In conclusion, the method here provides a potentially interesting approach to shorten the spin-up time in large-scale ocean models.

References

- Botta, E., Wubs, F., 1999. Matrix renumbering ILU: an effective algebraic multilevel ILU preconditioner for sparse matrices. *SIAM Journal on Matrix Analysis and Applications* 20 (4), 1007–1026.
- Bryan, K., 1984. Accelerating the convergence to equilibrium of ocean-climate models. *Journal of Physical Oceanography* 14, 666–673.
- Coleman, T.F., Garbow, B.S., Moré, J.J., 1984. Software for estimating sparse Jacobian matrices. *ACM Transactions on Mathematical Software* 10 (3), 329–345.
- De Niet, A., Wubs, F.W., van Scheltinga, A.D.T., Dijkstra, H.A., 2007. A tailored solver for the bifurcation analysis of ocean-climate models. *Journal of Computational Physics* 227, 654–679.
- Dijkstra, H.A., Öksüzöglü, H., Wubs, F.W., Botta, E.F.F., 2001. A fully implicit model of the three-dimensional thermohaline ocean circulation. *Journal of Computational Physics* 173, 685–715.
- Eisenstat, S.C., Walker, H.F., 1994. Globally convergent inexact Newton methods. *SIAM Journal on Optimization* 4 (2), 393–422.
- Frayssé, V., Giraud, L., Gratton, S., Langou, J., 2003. A Set of GMRES Routines for Real and Complex Arithmetics on High Performance Computers. CERFACS, available on-line at <http://www.cerfacs.fr/algor/reports/2003/TR_PA_03_03.pdf>.
- Griffies, S., Harrison, M., Pacanowski, R., Rosati, A., 2004. A Technical Guide to MOM4. NOAA/Geophysical Fluid Dynamics Laboratory, available on-line at <<http://www.gfdl.noaa.gov/fms>>.
- Khatiwala, S., Visbeck, M., Cane, M.A., 2005. Accelerated simulation of passive tracers in ocean circulation models. *Ocean Modelling* 9, 51–69.
- Klinger, B.A., 2000. Acceleration of general circulation model convergence by exponential extrapolation. *Ocean Modelling* 2, 61–72.
- Knoll, D., Keyes, D., 2004. Jacobian-free Newton–Krylov methods: a survey of approaches and applications. *Journal of Computational Physics* 193, 357–397.
- Knoll, D., Mousseau, V., Chacón, L., Reisner, J., 2005. Jacobian-free Newton–Krylov methods for accurate time integration of stiff wave systems. *Journal of Scientific Computing* 25 (Nov.), 213–229.
- Nadiga, B.T., Taylor, M., Lorenz, J., 2006. Ocean modelling for climate studies: eliminating short time scales in long-term, high-resolution studies of ocean circulation. *Mathematical and Computer Modelling* 44 (9–10), 870–886.

- Reisner, J., Mousseau, V., Knoll, D., 2000. Application of the Newton–Krylov method to geophysical flows. *Monthly Weather Review* 129, 2404–2415.
- Reisner, J., Wyszogrodzki, A., Mousseau, V., Knoll, D., 2003. An efficient physics based preconditioner for the fully implicit solution of small-scale thermally driven atmospheric flows. *Journal of Computational Physics* 189, 30–44.
- Saad, Y., 1996. *Iterative Methods for Sparse Linear Systems*. PWS Publishing Company.
- Samelson, R., Vallis, G.K., 1995. Planetary-Geostrophic Ocean Model: A User's Guide. Available on-line at <<http://www-po.coas.oregonstate.edu/homes/rms/ocean/>>.
- Samelson, R., Vallis, G.K., 1997a. Large-scale circulation with small diapycnal diffusion: the two-thermocline limit. *Journal of Marine Research* 55, 223–275.
- Samelson, R., Vallis, G.K., 1997b. A simple friction and diffusion scheme for planetary geostrophic basin models. *Journal of Physical Oceanography* 27 (Jan.), 186–194.
- Smith, R., Gent, P., 2002. Reference Manual for the Parallel Ocean Program (POP). Available on-line at <http://climate.lanl.gov/Models/POP/POP_Reference.ps>.
- van Noorden, T., Lunel, S.V., Blik, A., 2003. The efficient computation of periodic states of cyclically operated chemical precesses. *IMA Journal of Applied Mathematics* 68, 149–166.
- Weijer, W., Dijkstra, H.A., Oksuzoglu, H., Wubs, F.W., De Niet, A.C., 2003. A fully-implicit model of the global ocean circulation. *Journal of Computational Physics* 192, 452–470.