OPERATOR-SPLITTING METHODS RESPECTING EIGENVALUE PROBLEMS FOR SHALLOW SHELF EQUATIONS WITH BASAL DRAG.

JÜRGEN GEISER*, REINHARD CALOV†, AND THOMAS RECKNAGEL‡

Abstract.
We discuss different numerical methods for solving the shallow shelf equations with basal drag. The coupled equations are decomposed into operators for membranes stresses, basal shear stress and driving stress. Applying reasonable parameter values, we demonstrate that the operator of the membrane stresses is much stiffer than operator of the basal shear stress. Therefore, we propose a new splitting method, which alternates between the iteration on the membrane-stress operator and the basal-shear operator, with a stronger iteration on the operator of the membrane stress. We show that this splitting improves the computational performance of the numerical method, although the choice of the (standard) method to solve for all operators in one step speeds up the scheme too. (Based on the delicate and coupled equation we propose a new decomposition method to decouple into simpler solvable sub-equations. After a number of approximations we consider the error of the method and proposed a choice of the operators.)

Keywords: partial differential equations, operator-splitting methods, iterative methods, eigenvalue approach.
AMS subject classifications. 80A20, 80M25, 74S10, 76R50, 35J60, 35J65, 65M99, 65Z05, 65N12

1. Introduction.

The shallow shelf approximation with basal drag (SSAB) finds its application in the simulation of ice streams, which are regions of fast ice flow in ice sheets like Greenland or Antarctica, but also former ice bodies like the Laurentide ice sheet or the Fennoscandinavian ice sheet. Due to rather low but still present basal drag, the speed of such streams is one to two orders of magnitude higher than that of ordinary ice flow.

In this paper, we will discus different numerical solutions of SSAB equations. They follow from balance of momentum and the flow law of ice after a number of approximations (scaling, introducing vertically averaged stress, see e.g. [18]). u and v are the x- and y-components of (average) horizontal velocity, respectively.

$$\frac{\partial}{\partial x}\left(2\nu H\left(2\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\right)\right)+\frac{\partial}{\partial y}\left(\nu H\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right)\right)-\rho g H\frac{\partial h}{\partial x}-\tau^{x(b)}=0 \qquad (1.1)$$
$$\frac{\partial}{\partial y}\left(2\nu H\left(2\frac{\partial v}{\partial y}+\frac{\partial u}{\partial x}\right)\right)+\frac{\partial}{\partial x}\left(\nu H\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right)\right)-\rho g H\frac{\partial h}{\partial y}-\tau^{y(b)}=0 \qquad (1.2)$$

where g, ρ, H and h are the Earth's acceleration, density of ice, ice thickness and ice surface elevation, respectively. Physically spoken, above equations balance the membranes stresses (longitudinal stresses and lateral shear stress) with driving stress resulting from the Earth's acceleration and the and the basal shear stress. Glaciologist often subsume the lateral shear stress among longitudinal stress regarding all stress components which are not horizontal plane stress as longitudinal stress ([22]).

*Department of Mathematics, Humboldt Universität zu Berlin, Unter den Linden 6, D-10099 Berlin, Germany, E-mail: geiser@mathematik.hu-berlin.de
†Potsdam Institute for Climate Impact Research, PO Box 60 12 03, D-14412 Potsdam, Germany
‡Potsdam Institute for Climate Impact Research, PO Box 60 12 03, D-14412 Potsdam, Germany and University of Potsdam, Potsdam, Germany

We assume a simple expression for the basal shear stress:

$$\tau^{x(b)} = \frac{c_s u}{(u^2 + v^2)^{p/(2(p+1))}} \tag{1.3}$$

$$\tau^{y(b)} = \frac{c_s v}{(u^2 + v^2)^{p/(2(p+1))}} \tag{1.4}$$

with a parameter p other value possible. In praxis, different values for p apply. Ref. [33] use $p \approx 1.25$. For $p = 0$ equation (1.4) reduces to a linear relation between velocity and basal shear.

The effective viscosity reads

$$\nu = \frac{\bar{B}}{2\left[\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2 + \frac{1}{4}\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)^2 + \frac{\partial u}{\partial x}\frac{\partial v}{\partial y}\right]^{(n-1)/(2n)}}$$

with $n = 3$ and

$$\bar{B} = \int_b^h B(T')\, dz$$

b denotes the elevation of base of the ice stream and T' is the homologous temperature (temperature corrected for melting point).

For our numerical schemes, we assume in our first approach that the average inverse rate factor $\bar{B}$ is constant. In general, $B(T')$ can depend on temperature and water content in the ice and is calculated via separate equations.

The outline of the paper is as follows. The mathematical methods are introduced in the section 2. The solver schemes are discussed in section 3. In Section 4, we discuss the algorithm and the assembling of the splitting schemes. In Section 5 we introduce the application of our methods for some real-life problems. Finally we discuss future works in the area of iterative methods.

2. Mathematical Method. In the following we discuss the iterative splitting methods:

Iterative splitting with respect to one operator

$$\frac{\partial c_i(t)}{\partial t} = Ac_i(t) \;+\; Bc_{i-1}(t), \text{ with } c_i(t^n) = c^n, i = 1, 2, \ldots, m \tag{2.1}$$

Iterative splitting with respect to alternating operators

$$\frac{\partial c_i(t)}{\partial t} = Ac_i(t) \;+\; Bc_{i-1}(t), \text{ with } c_i(t^n) = c^n \tag{2.2}$$
$$i = 1, 2, \ldots, j\ ,$$
$$\frac{\partial c_i(t)}{\partial t} = Ac_{i-1}(t) \;+\; Bc_i(t), \quad \text{ with } c_{i+1}(t^n) = c^n\ , \tag{2.3}$$
$$i = j+1, j+2, \ldots, m\ ,$$

In addition, $c_0(t^n) = c^n$, $c_{-1} = 0$ and c^n is the known split approximation at the time level $t = t^n$. The split approximation at the time-level $t = t^{n+1}$ is defined as $c^{n+1} = c_{2m+1}(t^{n+1})$. (Clearly, the function $c_{i+1}(t)$ depends on the interval $[t^n, t^{n+1}]$, too, but, for the sake of simplicity, in our notation we omit the dependence on n.)

2.1. Iterative operator-splitting method as fixed-point scheme. The iterative operator-splitting method is used as a fixed-point scheme to linearize the nonlinear operators (see [15] and [26]).

We concentrate again on nonlinear differential equations of the form

$$\frac{du}{dt} = A(u(t))u(t) + B(u(t))u(t), \quad \text{with } u(t^n) = u^n, \tag{2.4}$$

where $A(u), B(u)$ are matrices with nonlinear entries and densely defined, where we assume that the entries involve the spatial derivatives of c [38]. In the following we discuss the standard iterative operator-splitting method as a fixed-point iteration method to linearize the operators.

We split our nonlinear differential equation (2.4) by applying

$$\frac{du_i(t)}{dt} = A(u_{i-1}(t))u_i(t) \ + \ B(u_{i-1}(t))u_{i-1}(t), \text{ with } \ u_i(t^n) = u^n, \tag{2.5}$$

$$\frac{du_{i+1}(t)}{dt} = A(u_{i-1}(t))u_i(t) \ + \ B(u_{i-1}(t))u_{i+1}(t), \text{ with } \ u_{i+1}(t^n) = u^n, \tag{2.6}$$

where the time-step is $\tau = t^{n+1} - t^n$. The iterations are $i = 1, 3, \ldots, 2m + 1$. $u_0(t) = u_n$ is the starting solution, where we assume that the solution u^{n+1} is near u^n, or $u_0(t) = 0$. Therefore we have to solve the local fixed-point problem. u^n is the known split approximation at time-level $t = t^n$. The stopping rule is given as

$$||u_i - u_{i-1}|| \leq \epsilon, \tag{2.7}$$

where $\epsilon > 0$ is the error bound.

The split approximation at time-level $t = t^{n+1}$ is defined as $u^{n+1} = u_{2m+2}(t^{n+1})$. We assume that the operators $A(u_{i-1}(t^{n+1})), B(u_{i-1}(t^{n+1}))$ are constant for $i = 1, 3, \ldots, 2m + 1$. Here the linearization is done with respect to the iterations, such that $A(u_{i-1}), B(u_{i-1})$ are at least non-dependent operators in the iterative equations, and we can apply the linear theory. For the linearization we assume at least in the first equation $A(u_{i-1}(t)) \approx A(u_i(t))$, and in the second equation $B(u_{i-1}(t)) \approx B(u_{i+1}(t))$, for small t.
We have

$$||A(u_{i-1}(t^{n+1}))u_i(t^{n+1}) - A(u^{n+1})u(t^{n+1})|| \leq \epsilon,$$

for sufficient iterations $i \in \{1, 3, \ldots, 2m + 1\}$.

Remark 2.1. *The linearization with the fixed-point scheme can be used for smooth or weak nonlinear operators, otherwise we lose the convergence behavior, while we did not converge to the local fixed point [26].*

2.2. Operator-splitting method with embedded Jacobian Newton iterative method. Newton's method is used to solve the nonlinear parts of the iterative operator-splitting method (see the linearization techniques in [26, 28]). We apply the iterative operator-splitting method and obtain:

$$\begin{aligned} F_1(u_i) &= \partial_t u_i - A(u_i)u_i - B(u_{i-1})u_{i-1} = 0, \\ &\text{with} \quad u_i(t^n) = u^n, \\ F_2(u_{i+1}) &= \partial_t u_{i+1} - A(u_i)u_i - B(u_{i+1})u_{i+1} = 0, \\ &\text{with} \quad u_{i+1}(t^n) = u^n, \end{aligned}$$

where the time-step is $\tau = t^{n+1} - t^n$. The iterations are $i = 1, 3, \ldots, 2m+1$. $u_0(t) = 0$ is the starting solution and u^n is the known split approximation at time-level $t = t^n$. The results of the methods are $c(t^{n+1}) = u_{2m+2}(t^{n+1})$. The splitting method with the embedded Newton's method is given by

$$\begin{aligned} & u_i^{(k+1)} = u_i^{(k)} \\ & -D(F_1(u_i^{(k)}))^{-1}(\partial_t u_i^{(k)} - A(u_i^{(k)})u_i^{(k)} - B(u_{i-1}^{(k)})u_{i-1}^{(k)}), && (2.8) \\ & \text{with } D(F_1(u_i^{(k)})) = -(A(u_i^{(k)}) + \frac{\partial A(u_i^{(k)})}{\partial u_i^{(k)}} u_i^{(k)}), && (2.9) \\ & \text{and } k = 0, 1, 2, \ldots, K_1, && (2.10) \\ & \text{with} \quad u_i(t^n) = u^n, && (2.11) \\ & \text{Initialization} : u_i^{(0)} = u^n && (2.12) \\ & \text{Stopping rule} : |u_i^{(k+1)} - u_i^{(k)}| \leq \epsilon_1, && (2.13) \end{aligned}$$

$$\begin{aligned} & u_{i+1}^{(l+1)} = u_{i+1}^{(l)} - D(F_2(u_{i+1}^{(l)}))^{-1}(\partial_t u_{i+1}^{(l)} - A(u_i^{(k)})u_i^{(k)} - B(u_{i+1}^{(k)})u_{i+1}^{(k)}), && (2.14) \\ & \text{with } D(F_2(u_{i+1}^{(l)})) = -(B(u_{i+1}^{(l)}) + \frac{\partial B(u_{i+1}^{(l)})}{\partial u_{i+1}^{(l)}} u_{i+1}^{(l)}), && (2.15) \\ & \text{and } l = 0, 1, 2, \ldots, K_2, && (2.16) \\ & \text{with} \quad u_{i+1}(t^n) = u^n, && (2.17) \\ & \text{Initialization} : u_{i+1}^{(0)} = u^n, && (2.18) \\ & \text{Stopping rule} : |u_{i+1}^{(l+1)} - u_{i+1}^{(l)}| \leq \epsilon_1. && (2.19) \end{aligned}$$

The iterations are $i = 1, 3, \ldots, 2m+1$. $u_0(t) = 0$ is the starting solution and u^n is the known split approximation at the time level $t = t^n$. The stopping rule of the outer iteration is given as:

$$|u_{i+1}^{(\tilde{K}_2)} - u_i^{\tilde{K}_1}| \leq \epsilon_2, \tag{2.20}$$

where $\tilde{K}_1$ and $\tilde{K}_2$ are the number of steps for the inner iterations. The result of the schemes is $c(t^{n+1}) = u_{\tilde{m}}(t^{n+1})$.

REMARK 2.2. *For the iterative operator-splitting method with Newton's method we have two iteration procedures. The first iteration is Newton's method for computing the solution of the nonlinear equations, and the second iteration is the iterative splitting method, which computes the resulting solution of the coupled equation systems. The embedded method is used for strong nonlinearities.*

2.3. Decoupling ideas based on eigenvalue problems. To detect the operators in the differential equation as stiff or non-stiff operators, we can apply the eigenvalues of each operator and use them as reciprocal time scales.

The operator equations are analyzed with the eigenvalue problem:

$$\partial_t c(t) = (A+B)c(t) = (\lambda_A + \lambda_B)c(t), \ t \in (t^n, t^{n+1}), \quad (2.21)$$
$$c(t^n) = g(t), \ c'(t^n) = f(t),$$

where the operators A and B result form the spatial discretization.

Stiffness:

We consider constant coefficients to the abstract Cauchy problem, see [17], where we suppose the stiffness in the following sense:

The matrix A is supposed to be stiff and B non-stiff.

Parabolic Part

Stiffness is seen as τA is huge in norms for the range of step size τ, see [21]. So the step size represents a splitting step size and we assume:

$$||\tau A|| >> 1, ||\tau B|| = O(\tau) \quad (2.22)$$

For the notation of the eigenvalues, we have:

$$Re(\tau\lambda) << -1, |\tau\mu| = O(\tau) \quad (2.23)$$

where λ is a stiff eigenvalue of A and μ a non-stiff eigenvalue of B.

Elliptic Part

Stiffness is seen as A (A is spatial discretised with the spatial step h) is huge in the condition, see [20].

So the maximal and minimal eigenvalues are strong variating, we assume:

$$\lambda_{max} = ||A||_2, \lambda_{min} = 1/||A^{-1}||_2, \quad (2.24)$$
$$\chi_A = \lambda_{max}/\lambda_{min} \quad (2.25)$$

where χ_A is the condition of matrix A.

For stiffness of the matrix, we have:

$$\chi_A >> 1, \chi_B \approx 1. \quad (2.26)$$

where the eigenvalues of A are very strong variating and the eigenvalue of B are very small variating.

In the following we discuss the idea to derive the eigenvalues:

The eigenvalues are detected in the decoupled equations:

$$Ac = \lambda_A c, \quad (2.27)$$

$$Bc = \lambda_B c, \quad (2.28)$$

Based on the eigenvalues $\lambda_{A,min}$, $\lambda_{A,max}$, $\lambda_{B,min}$ and $\lambda_{B,max}$ we can propose the condition number and the stiffness of the systems.

If we deal with a parabolic equation, the time steps $\Delta t_A \approx 1/\lambda_{A,max}$ and $\Delta t_B \approx 1/\lambda_{B,max}$.

If we deal with a elliptic equation, the condition number can be improved by pre-conditioners, e.g. SSOR method, such that $\chi_{A,precond} \approx 1$.

We propose the vector iteration based on the Rayleigh quotient for the computation of the eigenvalues of the operators A and B:

$$c_{i,k+1} = \frac{(A - I\lambda_{A,k})^{-1} c_{i,k}}{||(A - I\mu_i)^{-1} c_{i,k}||} \tag{2.29}$$

$$c_{i,m+1} = \frac{(B - I\lambda_{B,m})^{-1} c_{i,m}}{||(B - I\mu_i)^{-1} c_{i,m}||} \tag{2.30}$$

where $k, m = 0, 1, 2, \ldots$ and the eigenvalues and we assume to have the initialization of a eigenvector $c_{i,0}$ and eigenvalues $\lambda_{A,0}$ and $\lambda_{B,0}$, and set the next approximation of the eigenvalue to the Rayleigh quotient of the current iteration are given as:

$$\lambda_{A,k} = \frac{c_{i,k}^t A c_{i,k}}{c_{i,k}^t c_{i,k}}, \tag{2.31}$$

$$\lambda_{B,m} = \frac{c_{i,m}^t B c_{i,m}}{c_{i,m}^t c_{i,m}}, \tag{2.32}$$

where $k, m = 0, 1, 2, \ldots$ and the eigenvalues are given as

$$\frac{||c_{i+1,k+1}||}{||c_{i+1,k}||} = |\lambda_{A,1}| + \mathcal{O}(p^k), \tag{2.33}$$

$$\frac{||c_{i+1,m+1}||}{||c_{i+1,m}||} = |\lambda_{B,1}| + \mathcal{O}(q^m), \tag{2.34}$$

where $\lambda_{A,1}$ and $\lambda_{B,1}$ are the maximal eigenvalues. The values are given as $p = \frac{\lambda_{A,2}}{\lambda_{A,1}}$ with $\lambda_{A,1} \geq \lambda_{A,2} \ldots \geq \lambda_{A,n}$, $q = \frac{\lambda_{B,2}}{\lambda_{B,1}}$ with $\lambda_{B,1} \geq \lambda_{B,2} \ldots \geq \lambda_{B,n}$.

The following algorithm is used for separating the different scales of the operators A and B, we assume that pre-eigenvalues are first results of eigenvalues with $2-3$ iterative steps:

Algorithm 2.3.

```
1) We have the operators A, B.
2) We compute eigenvalues with a given norm $|| \cdot ||$:
    $||Ac||$ , $||Bc||$ ,
    where c is a possible solution vector of the equations (2.2)-(2.3).
3) We compare the pre-eigenvalues:
    $||Ac|| \leq ||Bc||$: A is stiff,
    or
    $||Ac|| \geq ||Bc||$: B is stiff.
4) We initialize our splitting method (e.g. for the nonlinear case).
4.1.) If A is stiff, we start with the iteration (2.5)
4.2.) If B is stiff, we start with the iteration (2.6)
```

Remark 2.4. *The efficiency of the method is given with the correct decomposition, which means the correct ordering of the underlying operators. If the operators are different based on their spectrum, the error of the scheme, can be reduced more if the iterations are done via the larger eigenvalues to solve the more delicate operator. With respect to the local error, means the higher eigenvalues for example of A, the starting operator A in the first iterative equation dominates the error, if we assume* *large time-steps, or for the stationary problem ($t \rightarrow \infty$). Therefore the pre-processing to obtain the underlying eigenvalues is important and accelerates the solver process. Here we propose the vector iterations to compute the eigenvalues as a method that is embedded to our iterative splitting method. The declaration of the operators to be stiff or non-stiff results in the correct splitting operators.*

At least splitting into two different iterative schemes makes sense, if we have strongly varying eigenvalues, so we decouple into each special iterative problem. If the eigenvalues are nearly the same, the splitting, does not make sense, while we could solve the problem in one operator.

2.4. ADI and LOD methods. We deal with non-stationary formulations means we introduce an artificial time, for which we assume to have the stationary case with $t \rightarrow \infty$.

Further, we have the heat equation, see [15], for which the mathematical equations are given by

$$\partial_t \, u \;=\; D_1(x,y) \, \partial_{xx} \, u \;+\; D_2(x,y) \, \partial_{yy} \, u \;+\; D_3(x,y) \, \partial_{zz} \, u \ , \ \text{in} \ \Omega \times [0,T] \ , \tag{2.35}$$

$$u(x,y,0) = u_0(x,y) \ , \ \text{on} \ \Omega \ , \tag{2.36}$$

The unknown function $u = u(x,t)$ is considered to be in $\Omega \times (0,T) \subset I\!R^d \times I\!R$ where the spatial dimension is given by d . The function $\mathbf{D}(x,y) = (D_1(x,y), D_2(x,y), D_3(x,y))^t \in I\!R^{3,+}$ describes the heat transfer in x, y, z. The functions $u_0(x,y)$ is the initial condition for the heat equation.

The boundary conditions are given as

$$u(x,y,t) = o \ , \ \text{on} \ \partial\Omega \times T \ : \ \text{Dirichlet boundary condition} \ , \tag{2.37}$$

$$\frac{\partial u(x,y,t)}{\partial n} = 0 \ , \ \text{on} \ \partial\Omega \times T \ : \ \text{Neumann boundary condition} \ , \tag{2.38}$$

$$\mathbf{D} \nabla u(x,y,t) = u_{out} \ , \ \text{on} \ \partial\Omega \times T \ : \ \text{outflow boundary condition} \ . \tag{2.39}$$

2.5. Spatial Discretization methods. For the spatial discretization methods we apply higher-order compact methods (HOC), see [27]. We concentrate on the two dimensional case, but the three dimensional case can be done in the same manner.

For equation (2.35) we can derive the following higher order spatial discretization methods

$$L_x L_y \tilde{u} = (L_y A_x + L_x A_y) u + O(h^p), \tag{2.40}$$

where $\tilde{u} = u_t$ (first order time derivative) or $\tilde{u} = u_t$ (second order time derivative) and $p = 2, 4$.
We obtain the following operators :

$$L_x = 1,$$
$$A_x = D_1 \delta_{xx},$$
$$L_y = 1,$$
$$A_y = D_2 \delta_{yy},$$

with fourth order in space, we obtain the following operators :

$$\begin{aligned}
L_x &= 1 + \frac{h_x^2}{12}\delta_{xx}, \\
A_x &= D_1 \delta_{xx}, \\
L_y &= 1 + \frac{h_y^2}{12}\delta_{yy}, \\
A_y &= D_2 \delta_{yy},
\end{aligned}$$

where $h = \max\{h_x, h_y\}$. $\delta_{xx} u_i = \frac{u_{i+2} - u_i + u_{i-2}}{4\Delta x^2}$ and δ_{yy} are the central difference operators for the second derivative.

2.6. Time-Discretization methods. For the time-discretization methods, we apply the standard discretization methods, as Crank-Nicolson scheme for the first order derivative, or the central difference scheme for the second order derivative. We obtain higher order methods by Richardson-Extrapolation or weighted methods.

First order derivative in time

For the first order derivative in time we deal with the second-order Crank-Nicolson method. To obtain higher order we apply Richardson Extrapolation to get fourth and fifth order methods.

Our standard Crank-Nicolson method is given as

$$\begin{aligned}
\tfrac{u^{n+1} - u^n}{\Delta t} &= 1/2(f(u^n) + f(u^{n+1})), \\
u(0) &= u_0,
\end{aligned} \tag{2.41}$$

where u^n is the time-approximated solution at $t^n = n\Delta t$, $n \geq 0$. Δt denotes the time increment.

The scheme is given as

$$\begin{aligned}
S_{CN}(\Delta t, u^n) = u^{n+1,CN} &= u^n + \frac{\Delta t}{2}(f(u^n) + f(u^{n+1})), \\
S_{CN}(\Delta t, u^n) &= B(\frac{\Delta t}{2}, F(\frac{\Delta t}{2}, u^n)),
\end{aligned}$$

where

$$\begin{aligned}
B(\frac{\Delta t}{2}, u^{n+1/2}) = u^{n+1,B} &= u^{n+2} + \frac{\Delta t}{2} f(u^{n+1}), \\
F(\frac{\Delta t}{2}, u^n) = u^{n+1/2,F} &= u^n + \frac{\Delta t}{2} f(u^n).
\end{aligned}$$

where B is the backward and F is the forward discretization scheme.
Based on this second-order method, we can apply an extrapolation method to obtain a higher-order method, which we need for our modified ADI method.

We apply the Richardson extrapolation on the second-order Crank-Nicolson method to obtain higher-order methods.

The idea of the extrapolation method is given as follows.

$$D_4(\Delta t, u^n) = u^{n+1,4th} = \frac{4}{3} S_{CN}(\frac{\Delta t}{2}, S_{CN}(\frac{\Delta t}{2}, u^n)) - \frac{1}{3} S_{CN}(\Delta t, u^n). \tag{2.42}$$

To obtain fifth order, we have to apply a Richardson extrapolation additionally, see [9]:

$$D_5(\Delta t, u^n) = u^{n+1,5th} = \frac{16}{15} D_4(\frac{\Delta t}{2}, D_4(\frac{\Delta t}{2}, u^n)) - \frac{1}{15} D_4(\Delta t, u^n). \qquad (2.43)$$

These methods can be implemented with respect to the basic time-discretization method. Another method, which we can obtain a higher order, is given in the following part.

2.7. Alternating direction implicit (ADI) method).

To obtain a fourth-order ADI method, the underlying time-discretization method has to be at least fourth-order in time.

So we support our new method with the Richard extrapolation that uses the second-order ADI method, based on the Crank-Nicolson time discretization, and we reach at least a fourth-order method.

For the second-order ADI method we apply the second-order time-discretization given as Crank-Nicolson (CN) method.

The CN time discretization is given as

$$(L_x L_y + \frac{\Delta t}{2} L^*) u^{n+1} = (L_x L_y - \frac{\Delta t}{2} L^*) u^n + O(h^4) + O(\Delta t^3), \qquad (2.44)$$

where $L^* = L_y A_x + L_x A_y$ and the discretization is of order 2 in time and order 4 in space, see equation (2.40).

The ADI-method, following [7], is given as follows.

$$(L_x + \frac{\Delta t}{2} A_x) u^* = (L_x - \frac{\Delta t}{2} A_x)(L_y - \frac{\Delta t}{2} A_y) u^n, \qquad (2.45)$$

$$(L_y + \frac{\Delta t}{2} A_y) u^{n+1} = u^*, \qquad (2.46)$$

where we obtain a second order ADI-scheme.

By applying the Richardson extrapolation we obtain a fourth-order method for the CN scheme, see scheme (2.42).

Proposition 2.1.

The ADI-method based on the extrapolation and CN method is given in the next steps.

Step 1 (first $\Delta t/2$ step), $\alpha = 1/2$:

$$(L_x + \alpha \frac{\Delta t}{2} A_x) u^{*,n+1/2} = (L_x - \alpha \frac{\Delta t}{2} A_x)(L_y - \alpha \frac{\Delta t}{2} A_y) u^n, \qquad (2.47)$$

$$(L_y + \alpha \frac{\Delta t}{2} A_y) u^{n+1/2} = u^{*,n+1/2}, \qquad (2.48)$$

Step 2 (Δt step), $\alpha = 1.0$:

$$(L_x + \alpha \frac{\Delta t}{2} A_x) u^{*,n+1} = (L_x - \alpha \frac{\Delta t}{2} A_x)(L_y - \alpha \frac{\Delta t}{2} A_y) u^n, \qquad (2.49)$$

$$(L_y + \alpha \frac{\Delta t}{2} A_y) \tilde{u}^{n+1} = u^{*,n+1}, \qquad (2.50)$$

Step 3 (second $\Delta t/2$ step), $\alpha = 1/2$:

$$(L_x + \alpha \frac{\Delta t}{2} A_x) u^{*,n+1} = (L_x - \alpha \frac{\Delta t}{2} A_x)(L_y - \alpha \frac{\Delta t}{2} A_y) u^{n+1/2}, \tag{2.51}$$

$$(L_y + \alpha \frac{\Delta t}{2} A_y) \tilde{\tilde{u}}^{n+1} = u^{*,n+1}, \tag{2.52}$$

Resulting step:

$$u^{n+1} = 4/3 \tilde{\tilde{u}}^{n+1} - 1/3 \tilde{u}^{n+1}, \tag{2.53}$$

where we obtain a 4th order method, due to the Richardson Extrapolation with 2nd order methods.

2.8. Locally one dimensional (LOD) method). In the following we introduce the LOD method, see [27], as an improved splitting method while using prestepping techniques.

The method was discussed in [27] and is given by:

$$u^{n+1,0} - u^n = dt(A+B)u^n \tag{2.54}$$

$$u^{n+1,1} - u^{n+1,0} = dt\eta A(u^{n+1,1} - u^n) \tag{2.55}$$

$$u^{n+1} - u^{n+1,1} = dt\eta B(u^{n+1} - u^n) \tag{2.56}$$

where $\eta \in (0.0, 0.5)$ and A, B are the spatial discretised operators.

If we eliminate the intermediate values in 2.54- 2.56 we obtain

$$u^{n+1} - u^n = \Delta t(A+B)(\eta u^{n+1} - (1-\eta)u^n)$$

and thus we obtain $O(\Delta t^2)$.

So we obtain a second-order method for $\eta = 0.5$.

By applying the Richardson extrapolation we obtain a fourth-order method for the CN scheme, see scheme (2.42).

PROPOSITION 2.2.

The LOD-method based on the extrapolation and CN method is given in the next steps.

Step 1 (first $\Delta t/2$ step), $\alpha = 1/2$:

$$u^{n+1/2,0} - u^n = \alpha \; dt \; (A+B)u^n \tag{2.57}$$

$$u^{n+1/2,1} - u^{n+1/2,0} = \alpha \; dt \; \eta \; A(u^{n+1/2,1} - u^n) \tag{2.58}$$

$$u^{n+1/2} - u^{n+1,1} = \alpha \; dt \; \eta \; B(u^{n+1/2} - u^n) \tag{2.59}$$

Step 2 (Δt step), $\alpha = 1.0$:

$$u^{n+1,0} - u^n = \alpha \; dt \; (A+B)u^n \tag{2.60}$$

$$u^{n+1,1} - u^{n+1,0} = \alpha \; dt \; \eta \; A(u^{n+1,1} - u^n) \tag{2.61}$$

$$\tilde{u}^{n+1} - u^{n+1,1} = \alpha \; dt \; \eta \; B(\tilde{u}^{n+1} - u^n), \tag{2.62}$$

Step 3 (second $\Delta t/2$ step), $\alpha = 1/2$:

$$u^{n+1,0} - u^{n+1/2} = \alpha \; dt \; (A+B)u^{n+1/2} \tag{2.63}$$

$$u^{n+1,1} - u^{n+1,0} = \alpha \; dt \; \eta \; A(u^{n+1,1} - u^{n+1/2}) \tag{2.64}$$

$$\tilde{\tilde{u}}^{n+1} - u^{n+1,1} = \alpha \; dt \; \eta \; B(\tilde{\tilde{u}}^{n+1} - u^{n+1/2}), \tag{2.65}$$

Resulting step:

$$u^{n+1} = 4/3\tilde{\tilde{u}}^{n+1} - 1/3\tilde{u}^{n+1}, \tag{2.66}$$

where we obtain a 4th order method, due to the Richardson Extrapolation with 2nd order methods.

Remark 2.5.

For $\eta \in (0, 0.5)$ we have unconditionally stable methods and for higher order we use $\eta = \frac{1}{2}$. Then for sufficiently small time steps we get a conditionally stable splitting method.

3. Linear Iterative Methods: Jacobi Methods, Gauss-Seidel Methods and SOR Methods. To decompose a complicate matrix to simpler matrices we discuss in the following iterative schemes. The idea is to obtain a simpler, e.g. diagonal or tridiagonal matrix and a rest matrix. With the simpler matrix, that we assume to compute simpler the inverse, we iterate in the scheme, where the rest matrix (more or less a full matrix) is explicitly in the scheme. Therefore we save computational time in such an iterative scheme.

Decoupling idea

An underlying idea is to decouple a full matrix A into an part D that is simple to invert and a part $E + F$ that are delicate to invert and are expensive to compute. Here we can save computational time and not forget stability reasons while D include the diagonal part of the matrix.

In the following we discuss different splitting schemes.

The SOR method is described together with its properties. The same results hold for the Gauss-Seidel method, since it is a special case of the SOR method.

Consider the decomposition of matrix A:

$$\begin{aligned} & A = D - E - F, && (3.1) \\ & E : \text{strict lower triangular matrix },\\ & F : \text{strict upper triangular matrix }. \end{aligned}$$

The iterative methods result from the choice of approximated inverses and iteration matrices M:

$$\begin{aligned} & N_\omega^{Jacobi} = (\frac{1}{\omega}D)^{-1}, && (3.2) \\ & M_\omega^{Jacobi} = I\!\!1 - (\frac{1}{\omega}D)^{-1}A, && (3.3) \\ & \omega \in]0, 1[_{I\!R}, \end{aligned}$$

describes the weighted Jacobi method. For $\omega = 1$ the method is the same as the Jacobi method, which is also known as method of simultaneous displacements, Richardson iterative method or total-step iterative method [36]).

$$\begin{aligned} & N_\omega^{SOR} = (\frac{1}{\omega}D - E)^{-1}, && (3.4) \\ & M_\omega^{SOR} = I\!\!1 - (\frac{1}{\omega}D - E)^{-1}A, && (3.5) \\ & \omega \in]0, 2[_{I\!R}, \end{aligned}$$

where N^{SOR} denotes the inverse and M^{SOR} the iteration matrix of the SOR method.

Matrix M is applied to vector x^m. The component-wise representation can then be written as a sequence of single corrections x_i^m or x_i^{m+1}, respectively.
There holds:

$$x_i^{m+1} = (1-\omega)x_i^m + \frac{\omega}{D_{ii}} \left(\sum_{j=1}^{i-1} E_{ij} x_i^m + \sum_{j=i+1}^{n} F_{ij} x_i^m \right), \tag{3.6}$$
$$i = 1, \ldots, n,$$

for the weighted Jacobi method and

$$x_i^{m+1} = (1-\omega)x_i^m + \frac{\omega}{D_{ii}} \left(\sum_{j=1}^{i-1} E_{ij} x_i^{m+1} + \sum_{j=i+1}^{n} F_{ij} x_i^m \right), \tag{3.7}$$
$$i = 1, \ldots, n,$$

for the SOR method where n is the number of components.
The SOR method can be differentiated into locally damped methods with $(0 < \omega < 1)$ or locally overrelaxed Gauss-Seidel methods with $(1 < \omega < 2)$.
Choosing $\omega = 1$ the methods coincide.

The convergence of SOR methods can be proved over all positive definite matrices. We have the following theorems:

THEOREM 3.1. *Let A be a symmetric positive definite matrix. Then*

$$\rho(M_\omega^{GS}) \le ||M_\omega^{GS}||_A < 1, \quad \forall \omega \in]0,2[_{I\!R}. \tag{3.8}$$

THEOREM 3.2. *Let $A \in I\!R^{I \times I}$ be an irreducible diagonally dominant matrix, that satisfies the M-matrix conditions. Let I be an sorted index set.*
Then

$$\rho(M_\omega^{SOR}) < 1, \quad \forall \omega \in]0,1]_{I\!R}, \tag{3.9}$$
$$||M_\omega^{SOR}||_\infty < 1, \quad \forall \omega \in]0,1]_{I\!R}. \tag{3.10}$$

The SOR method depends on the sorting. The reversal of the sorting yields a backward oriented SOR method. By successive execution of SOR method and backward oriented SOR method a new method, the so-called *SSOR method* (symmetric SOR method), can be constructed. For $\omega = 1$ the method is called *symmetric Gauss-Seidel method*.

3.1. ILU Method.

3.1. ILU Method. Subsequently we describe the ILU method (incomplete triangular decomposition).

Matrix A can be written as follows using multiplicative splitting:

$$A = LU. \tag{3.11}$$

LU is the denotation for $\underline{\text{L}}$ower$\underline{\text{U}}$pper referring to the decomposition into lower and upper triangular matrices. It is a direct method, where the triangular matrices are usually full. Therefore, these methods aren't suitable due to their efficiency to solve large linear systems.

Hence a modification of the LU method is done yielding the ILU method (IncompleteLowerUpper). The Gauss elimination used for the LU decomposition is executed following a preset allocation pattern and yields using this pattern an iterative method with linear effort. A typical allocation method is achieved by choosing the graph of matrix A.

$$A = LU - R, \tag{3.12}$$
$$L_{\alpha\beta} = 0, (\alpha, \beta) \notin G(A),\ L \text{ lower triangular matrices}, \tag{3.13}$$
$$U_{\alpha\beta} = 0, (\alpha, \beta) \notin G(A),\ U \text{ upper triangular matrices}, \tag{3.14}$$
$$R_{\alpha\beta} = 0, (\alpha, \beta) \in G(A). \tag{3.15}$$

The ILU method is obtained using the following choice of approximated inverse N and iteration matrix M:

$$N^{ILU} := U^{-1}L^{-1}, \tag{3.16}$$
$$M^{ILU} := 1\!\!1 - U^{-1}L^{-1}A, \tag{3.17}$$

where N^{ILU} denotes the inverse and M^{ILU} the iteration matrix of the ILU method.

Consider, that the existence of the decomposition is ensured for special cases. For the M-matrices the following theorems hold:

THEOREM 3.3. *Let A be a M-matrix, and let the pattern for the ILU decomposition be the graph of matrix A, then the ILU decomposition exists.*

THEOREM 3.4. *Let A be a M-matrix. Then the ILU method* (3.16), (3.17), *converges in the maximum norm:*

$$\rho(M^{ILU}) \leq 1, \tag{3.18}$$
$$||M^{ILU}||_{\infty} \leq 1. \tag{3.19}$$

In the next section we apply our theoretical results to a test example with respect to correct or incorrect decompositions.

4. Algorithm and Assembling of the splitting methods with respect to the eigenvalues of the operators.

In the following we discuss the algorithms and their implementation to the software code. Here the underlying idea the calculation of the eigenvalues is embedded.

We discuss in the following operators:

Linearized Part:

$$A_1(\hat{\nu}) = \begin{pmatrix} 4\frac{\partial}{\partial x}\hat{\nu}H\frac{\partial}{\partial x} & 2\frac{\partial}{\partial x}\hat{\nu}H\frac{\partial}{\partial y} \\ 2\frac{\partial}{\partial y}\hat{\nu}H\frac{\partial}{\partial x} & 4\frac{\partial}{\partial y}\hat{\nu}H\frac{\partial}{\partial y} \end{pmatrix} \tag{4.1}$$

where $\hat{\nu} = \nu(u_{old}, v_{old})$

$$A_2(h) = -\begin{pmatrix} \rho g H\frac{\partial}{\partial x}h \\ \rho g H\frac{\partial}{\partial y}h \end{pmatrix}, \tag{4.2}$$

$$B(u,v) = -\mu(u,v) \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{4.3}$$

with

$$\mu(u,v) = \frac{c_s}{(u^2+v^2)^{p/(2(p+1))}} \tag{4.4}$$

and the nonlinear part:

$$B(u,v)(u,v)^t = \begin{pmatrix} \tau^{x(b)} \\ \tau^{y(b)} \end{pmatrix} \tag{4.5}$$

4.1. One side iterative schemes:. That is split into the following schemes, where both operators A_1 and B are solved together in one iterative step:

$$A_1(\nu_{i-1})(u_i,v_i)^t + A_2(h) + B(u_{i-1},v_{i-1})(u_i,v_i)^t = 0 \tag{4.6}$$

or with explicitely expressed operators

$$\frac{\partial}{\partial x}\left(2\nu_{i-1}H\left(2\frac{\partial u_i}{\partial x} + \frac{\partial v_i}{\partial y}\right)\right) + \frac{\partial}{\partial y}\left(\nu_{i-1}H\left(\frac{\partial u_i}{\partial y} + \frac{\partial v_i}{\partial x}\right)\right) - \rho g H\frac{\partial h}{\partial x} - \mu_{i-1}u_i = 0 \tag{4.7}$$

$$\frac{\partial}{\partial y}\left(2\nu_{i-1}H\left(2\frac{\partial v_i}{\partial y} + \frac{\partial u_i}{\partial x}\right)\right) + \frac{\partial}{\partial x}\left(\nu_{i-1}H\left(\frac{\partial u_i}{\partial y} + \frac{\partial v_i}{\partial x}\right)\right) - \rho g H\frac{\partial h}{\partial y} - \mu_{i-1}v_i = 0 \tag{4.8}$$

where ν_{i-1} is given with u_{i-1}, v_{i-1} for the previous solution.

Equations (4.7) and (4.8) are discretised using centred differences in the Arakawa C grid ([3]).

$$
\begin{aligned}
&\frac{1}{\Delta x_{l+1/2,m}}\left[2\nu_{l+1,m}h_{l+1,m}\left(2\frac{u_{l+3/2,m}-u_{l+1/2,m}}{\Delta x_{l+1,m}}\right.\right.\\
&\left.\left.+\ \frac{v_{l+1,m+1/2}-v_{l+1,m-1/2}}{\Delta y_{l+1,m}}\right)\right]-\frac{1}{\Delta x_{l+1/2,m}}\left[2\nu_{l,m}h_{l,m}\right.\\
&\left.\left(2\frac{u_{l+1/2,m}-u_{l-1/2,m}}{\Delta x_{l,m}}+\frac{v_{l,m+1/2}-v_{l,m-1/2}}{\Delta y_{l,m}}\right)\right]\\
&+\frac{1}{\Delta y_{l+1/2,m}}\left[\nu_{l+1/2,m+1/2}h_{l+1/2,m+1/2}\left(\frac{u_{l+1/2,m+1}-u_{l+1/2,m}}{\Delta y_{l+1/2,m+1/2}}\right.\right.\\
&\left.\left.+\ \frac{v_{l+1,m+1/2}-v_{l,m+1/2}}{\Delta x_{l+1/2,m+1/2}}\right)\right]-\frac{1}{\Delta y_{l+1/2,m}}\left[\nu_{l+1/2,m-1/2}h_{l+1/2,m-1/2}\right.\\
&\left.\left(\frac{u_{l+1/2,m}-u_{l+1/2,m-1}}{\Delta y_{l+1/2,m-1/2}}+\frac{v_{l+1,m-1/2}-v_{l,m-1/2}}{\Delta x_{l+1/2,m-1/2}}\right)\right]\\
&-\mu_{l+1/2,m}u_{l+1/2,m}=\rho gH_{l+1/2,m}\left(\frac{h_{l+1,m}-h_{l,m}}{\Delta x_{l+1/2,m}}\right) \qquad (4.9)\\
&\frac{1}{\Delta y_{l,m+1/2}}\left[2\nu_{l,m+1}h_{l,m+1}\left(2\frac{v_{l,m+3/2}-v_{l,m+1/2}}{\Delta y_{l,m+1}}\right.\right.\\
&\left.\left.+\ \frac{u_{l+1/2,m+1}-u_{l-1/2,m+1}}{\Delta x_{l,m+1}}\right)\right]-\frac{1}{\Delta y_{l,m+1/2}}\left[2\nu_{l,m}h_{l,m}\right.\\
&\left.\left(2\frac{v_{l,m+1/2}-v_{l,m-1/2}}{\Delta y_{l,m}}+\frac{u_{l+1/2,m}-u_{l-1/2,m}}{\Delta x_{l,m}}\right)\right]\\
&+\frac{1}{\Delta x_{l,m+1/2}}\left[\nu_{l+1/2,m+1/2}h_{l+1/2,m+1/2}\left(\frac{u_{l+1/2,m+1}-u_{l+1/2,m}}{\Delta y_{l+1/2,m+1/2}}\right.\right.\\
&\left.\left.+\ \frac{v_{l+1,m+1/2}-v_{l,m+1/2}}{\Delta x_{l+1/2,m+1/2}}\right)\right]-\frac{1}{\Delta x_{l,m+1/2}}\left[\nu_{l-1/2,m+1/2}h_{l-1/2,m+1/2}\right.\\
&\left.\left(\frac{u_{l-1/2,m+1}-u_{l-1/2,m}}{\Delta y_{l-1/2,m+1/2}}+\frac{v_{l,m+1/2}-v_{l-1,m+1/2}}{\Delta x_{l-1/2,m+1/2}}\right)\right]\\
&-\mu_{l,m+1/2}v_{l,m+1/2}=\rho gH_{l,m+1/2}\left(\frac{h_{l,m+1}-h_{l,m}}{\Delta y_{l,m+1/2}}\right) \qquad (4.10)
\end{aligned}
$$

where the integers l, m refer to the two horizontal coordinates.

The discretised effective viscosity together with the velocities which appear therein are taken from the previous solution at step $i-1$. It reads

$$
\begin{aligned}
\nu_{l,m}=&\frac{\bar{B}}{2}\left\{\left(\frac{u_{l+1/2,m}-u_{l-1/2,m}}{\Delta x_{l,m}}\right)^2+\left(\frac{v_{l,m+1/2}-v_{l,m-1/2}}{\Delta y_{l,m}}\right)^2\right.\\
&+\frac{1}{4}\left(\frac{u_{l-1/2,m+1/2}-u_{l-1/2,m-1/2}}{\Delta y_{l,m}}+\frac{v_{l+1/2,m-1/2}-v_{l-1/2,m-1/2}}{\Delta x_{l,m}}\right)^2\\
&\left.+\frac{u_{l+1/2,m}-u_{l-1/2,m}}{\Delta x_{l,m}}\ \frac{v_{l,m+1/2}-v_{l,m-1/2}}{\Delta v_{l,m+1/2}}+\epsilon_\nu^2\right\}^{(1-n)/2n} \qquad (4.11)
\end{aligned}
$$

As in [5] we introduce a regularization according $\epsilon_\nu = 3\cdot 10^{-14}\mathrm{s}^{-1}$.

Following the standard procedure, equations (4.9) and (4.10) are rewritten by collecting the coefficients of the discrete velocities:

$$
\begin{aligned}
&\beta^0_{l,m}u_{l+1/2,m} + \beta^1_{l,m}u_{l+3/2,m} + \beta^2_{l,m}u_{l-1/2,m} + \beta^3_{l,m}u_{l+1/2,m+1} \\
&+\beta^4_{l,m}u_{l+1/2,m-1} + \beta'^5_{l,m}v_{l,m+1/2} + \beta'^6_{l,m}v_{l,m-1/2} + \beta'^7_{l,m}v_{l+1,m+1/2} \\
&+\beta'^8_{l,m}v_{l+1,m-1/2} = f_{l,m} \qquad (4.12)\\
&\beta'^0_{l,m}v_{l,m+1/2} + \beta'^1_{l,m}v_{l+1,m+1/2} + \beta'^2_{l,m}v_{l-1,m+1/2} + \beta'^3_{l,m}v_{l,m+3/2} \\
&+\beta'^4_{l,m}v_{l,m-1/2} + \beta^5_{l,m}u_{l-1/2,m+1} + \beta^6_{l,m}u_{l-1/2,m} + \beta^7_{l,m}u_{l+1/2,m+1} \\
&+\beta^8_{l,m}u_{l+1/2,m} = f'_{l,m}. \qquad (4.13)
\end{aligned}
$$

Using these equations as basis, we will construct the matrix corresponding to the operator $A := A_1 + B$ in equation (4.6) of the linear system $A\hat{u} = f$. The right-hand side column vector f is the discretised representation of the two columns vector $-A_2$ in equation (4.6).

We order the velocities and the right-hand side column vector f as

$$\hat{u} = (v_{1,1.5}, u_{1.5,1}, v_{1,2.5}, \cdots, u_{lmax-0.5,mmax})^t, \quad f = (f'_{1,1}, f_{1,1}, f'_{1,2}, \cdots, f_{l,m})^t \qquad (4.14)$$

and find the five diagonal matrix

$$A = \begin{pmatrix} D & R & & & \\ L & D & R & & \\ & L & D & R & \\ & & & \ddots & \\ & & & & L \quad D \end{pmatrix} \qquad (4.15)$$

where the elements L, D and R denote

$$D = \begin{pmatrix}
\beta'^0_{l,1} & \beta^8_{l,1} & \beta'^3_{l,1} & \beta^7_{l,1} & & & & \\
\beta'^5_{l,1} & \beta^0_{l,1} & \cdot & \beta^3_{l,1} & \cdot & & & \\
\beta'^4_{l,2} & \cdot & \beta'^0_{l,2} & \beta^8_{l,2} & \beta'^3_{l,2} & \beta^7_{l,2} & & \\
\beta^6_{l,2} & \beta^4_{l,2} & \beta'^5_{l,2} & \beta^0_{l,2} & \cdot & \beta^3_{l,2} & & \\
\cdot & \cdot & \beta'^4_{l,3} & \cdot & \beta'^0_{l,3} & \beta^8_{l,3} & \beta'^3_{l,3} & \\
 & & \beta^6_{l,3} & \beta^4_{l,3} & \beta'^5_{l,3} & \beta^0_{l,3} & \cdot & \beta^3_{l,3} \\
 & & & & & \ddots & &
\end{pmatrix},$$

$$R = \begin{pmatrix}
\beta'^1_{l,1} & & & & & \\
\beta'^7_{l,1} & \beta^1_{l,1} & & & & \\
 & \cdot & \beta'^1_{l,2} & & & \\
\beta'^8_{l,2} & & \beta'^7_{l,2} & \beta^1_{l,2} & & \\
 & \cdot & & \cdot & \beta'^1_{l,3} & \\
 & & & & & \ddots
\end{pmatrix}, \quad L = \begin{pmatrix}
\beta'^2_{l,1} & \beta^6_{l,1} & \cdot & \beta^5_{l,1} & \\
 & \beta^2_{l,1} & & & \\
 & & \beta'^2_{l,2} & \beta^6_{l,2} & \beta^5_{l,2} \\
 & & & \ddots &
\end{pmatrix}.$$

The coefficients of the velocities are given as

$$\begin{aligned}
\beta^0_{l,m} =& -\frac{4\hat{\nu}_{l+1,m}}{\Delta x_{l+1/2,m}\Delta x_{l+1,m}} - \frac{4\hat{\nu}_{l,m}}{\Delta x_{l+1/2,m}\Delta x_{l,m}} \\
& - \frac{\hat{\nu}_{l+1/2,m+1/2}}{\Delta y_{l+1/2,m}\Delta y_{l+1/2,m+1/2}} - \frac{\hat{\nu}_{l+1/2,m-1/2}}{\Delta y_{l+1/2,m}\Delta y_{l+1/2,m-1/2}} - \mu_{l+1/2,m}, \quad (4.16)\\
\beta^1_{l,m} =& \frac{4\hat{\nu}_{l+1,m}}{\Delta x_{l+1/2,m}\Delta x_{l+1,m}}, \quad (4.17)\\
\beta^2_{l,m} =& \frac{4\hat{\nu}_{l,m}}{\Delta x_{l+1/2,m}\Delta x_{l,m}}, \quad (4.18)\\
\beta^3_{l,m} =& \frac{\hat{\nu}_{l+1/2,m+1/2}}{\Delta y_{l+1/2,m}\Delta y_{l+1/2,m+1/2}}, \quad (4.19)\\
\beta^4_{l,m} =& \frac{\hat{\nu}_{l+1/2,m-1/2}}{\Delta y_{l+1/2,m}\Delta y_{l+1/2,m-1/2}}, \quad (4.20)\\
\beta'^5_{l,m} =& -\frac{2\hat{\nu}_{l,m}}{\Delta x_{l+1/2,m}\Delta y_{l,m}} - \frac{\hat{\nu}_{l+1/2,m+1/2}}{\Delta y_{l+1/2,m}\Delta x_{l+1/2,m+1/2}}, \quad (4.21)\\
\beta'^6_{l,m} =& \frac{2\hat{\nu}_{l,m}}{\Delta x_{l+1/2,m}\Delta y_{l,m}} + \frac{\hat{\nu}_{l+1/2,m-1/2}}{\Delta y_{l+1/2,m}\Delta x_{l+1/2,m-1/2}}, \quad (4.22)\\
\beta'^7_{l,m} =& \frac{2\hat{\nu}_{l+1,m}}{\Delta x_{l+1/2,m}\Delta y_{l+1,m}} + \frac{\hat{\nu}_{l+1/2,m+1/2}}{\Delta y_{l+1/2,m}\Delta x_{l+1/2,m+1/2}}, \quad (4.23)\\
\beta'^8_{l,m} =& -\frac{2\hat{\nu}_{l+1,m}}{\Delta x_{l+1/2,m}\Delta y_{l+1,m}} - \frac{\hat{\nu}_{l+1/2,m-1/2}}{\Delta y_{l+1/2,m}\Delta x_{l+1/2,m-1/2}} \quad (4.24)
\end{aligned}$$

and

$$\begin{aligned}
\beta'^{0}_{l,m} =& - \frac{4\hat{\nu}_{l,m+1}}{\Delta y_{l,m+1/2}\Delta y_{l,m+1}} - \frac{4\hat{\nu}_{l,m}}{\Delta y_{l,m+1/2}\Delta y_{l,m}} \\
& - \frac{\hat{\nu}_{l+1/2,m+1/2}}{\Delta x_{l,m+1/2}\Delta x_{l+1/2,m+1/2}} - \frac{\hat{\nu}_{l-1/2,m+1/2}}{\Delta x_{l,m+1/2}\Delta x_{l-1/2,m+1/2}} - \mu_{l,m+1/2}, && (4.25) \\
\beta'^{1}_{l,m} =& \frac{4\hat{\nu}_{l,m+1}}{\Delta y_{l,m+1/2}\Delta y_{l,m+1}}, && (4.26) \\
\beta'^{2}_{l,m} =& \frac{4\hat{\nu}_{l,m}}{\Delta y_{l,m+1/2}\Delta y_{l,m}}, && (4.27) \\
\beta'^{3}_{l,m} =& \frac{\hat{\nu}_{l+1/2,m+1/2}}{\Delta x_{l,m+1/2}\Delta x_{l+1/2,m+1/2}}, && (4.28) \\
\beta'^{4}_{l,m} =& \frac{\hat{\nu}_{l-1/2,m+1/2}}{\Delta x_{l,m+1/2}\Delta x_{l-1/2,m+1/2}}, && (4.29) \\
\beta^{5}_{l,m} =& - \frac{2\hat{\nu}_{l,m+1}}{\Delta y_{l,m+1/2}\Delta x_{l,m+1}} - \frac{\hat{\nu}_{l-1/2,m+1/2}}{\Delta x_{l,m+1/2}\Delta y_{l-1/2,m+1/2}}, && (4.30) \\
\beta^{6}_{l,m} =& \frac{2\hat{\nu}_{l,m}}{\Delta y_{l,m+1/2}\Delta x_{l,m}} + \frac{\hat{\nu}_{l-1/2,m+1/2}}{\Delta x_{l,m+1/2}\Delta y_{l-1/2,m+1/2}}, && (4.31) \\
\beta^{7}_{l,m} =& \frac{2\hat{\nu}_{l,m+1}}{\Delta y_{l,m+1/2}\Delta x_{l,m+1}} + \frac{\hat{\nu}_{l+1/2,m+1/2}}{\Delta x_{l,m+1/2}\Delta y_{l+1/2,m+1/2}}, && (4.32) \\
\beta^{8}_{l,m} =& - \frac{2\hat{\nu}_{l,m}}{\Delta y_{l,m+1/2}\Delta x_{l,m}} + \frac{\hat{\nu}_{l+1/2,m+1/2}}{\Delta x_{l,m+1/2}\Delta y_{l+1/2,m+1/2}} && (4.33)
\end{aligned}$$

with the abbreviation

$$\hat{\nu}_{\tilde{l}\tilde{m}} = \nu_{\tilde{l}\tilde{m}} H_{\tilde{l}\tilde{m}}, \tilde{l} = l-1/2,\ l,\ l+1/2,\ l+1,\ \tilde{m} = m-1/2,\ m,\ m+1/2,\ m+1.$$

Further, the components of the right-hand side column vector read

$$\begin{aligned}
f_{l,m} =& \rho g H_{l+1/2,m}\left(\frac{h_{l+1,m} - h_{l,m}}{\Delta x_{l+1/2,m}}\right), && (4.34) \\
f'_{l,m} =& \rho g H_{l,m+1/2}\left(\frac{h_{l,m+1} - h_{l,m}}{\Delta y_{l,m+1/2}}\right) && (4.35)
\end{aligned}$$

The first method which we use to solve the system is the Jacobi (total-step iteration) method as described in (3.6). Applied to our problem, we construct the following algorithm:

$$\begin{aligned} v^i_{l,m+1/2} &= (1-\omega)v^{i-1}_{l,m+1/2} + \frac{\omega}{\beta'^0_{l,m}}\Big[f'_{l,m} - \beta'^1_{l,m}v^{i-1}_{l+1,m+1/2} - \beta'^2_{l,m}v^{i-1}_{l-1,m+1/2} \\ &- \beta'^3_{l,m}v^{i-1}_{l,m+3/2} - \beta'^4_{l,m}v^{i-1}_{l,m-1/2} - \beta^5_{l,m}u^{i-1}_{l-1/2,m+1} - \beta^6_{l,m}u^{i-1}_{l-1/2,m} \\ &- \beta^7_{l,m}u^{i-1}_{l+1/2,m+1} - \beta^8_{l,m}u^{i-1}_{l+1/2,m}\Big], \qquad (4.36) \\ u^i_{l+1/2,m} &= (1-\omega)u^{i-1}_{l+1/2,m} + \frac{\omega}{\beta^0_{l,m}}\Big[f_{l,m} - \beta^1_{l,m}u^{i-1}_{l+3/2,m} - \beta^2_{l,m}u^{i-1}_{l-1/2,m} \\ &- \beta^3_{l,m}u^{i-1}_{l+1/2,m+1} - \beta^4_{l,m}u^{i-1}_{l+1/2,m-1} - \beta'^5_{l,m}v^{i-1}_{l,m+1/2} - \beta'^6_{l,m}v^{i-1}_{l,m-1/2} \\ &- \beta'^7_{l,m}v^{i-1}_{l+1,m+1/2} - \beta'^8_{l,m}v^{i-1}_{l+1,m-1/2}\Big] \qquad (4.37) \end{aligned}$$

with $m = 1, ..., M_y - 1$, $l = 1, ..., M_x - 1$.

A appropriate value for weight ω will be found via testing (section **??**).

For the second method, we apply the SOR algorithm to solve the system. According to (3.7) we obtain the component-wise iteration steps for u and v:

$$\begin{aligned} v^i_{l,m+1/2} &= (1-\omega)v^{i-1}_{l,m+1/2} + \frac{\omega}{\beta'^0_{l,m}}\Big[f_{l,m} - \beta'^1_{l,m}v^{i-1}_{l+1,m+1/2} - \beta'^2_{l,m}v^{i}_{l-1,m+1/2} \\ &- \beta'^3_{l,m}v^{i-1}_{l,m+3/2} - \beta'^4_{l,m}v^{i}_{l,m-1/2} - \beta^5_{l,m}u^{i-1}_{l-1/2,m+1} - \beta^6_{l,m}u^{i-1}_{l-1/2,m} \\ &- \beta^7_{l,m}u^{i}_{l+1/2,m+1} - \beta^8_{l,m}u^{i}_{l+1/2,m}\Big], \qquad (4.38) \\ u^i_{l+1/2,m} &= (1-\omega)u^{i-1}_{l+1/2,m} + \frac{\omega}{\beta^0_{l,m}}\Big[f_{l,m} - \beta^1_{l,m}u^{i-1}_{l+3/2,m} - \beta^2_{l,m}u^{i}_{l-1/2,m} \\ &- \beta^3_{l,m}u^{i-1}_{l+1/2,m+1} - \beta^4_{l,m}u^{i}_{l+1/2,m-1} - \beta'^5_{l,m}v^{i-1}_{l,m+1/2} - \beta'^6_{l,m}v^{i-1}_{l,m-1/2} \\ &- \beta'^7_{l,m}v^{i}_{l+1,m+1/2} - \beta'^8_{l,m}v^{i}_{l+1,m-1/2}\Big] \qquad (4.39) \end{aligned}$$

with $m = 1, ..., M_y - 1$, $l = 1, ..., M_x - 1$.

For both of the methods, the stop criterion is given as:

$$\max_{l,m}(|u_{l+1/2,m,i+1} - u_{l+1/2,m,i}|, |v_{l+1/2,m,i+1} - v_{l+1/2,m,i}|) \leq err \qquad (4.40)$$

for all l, m, where err is a given bound, e.g. $err = 10^{-3}$. Later on, we will give a concrete value for err.

We will use both with appropriate modifications in our splitting schemes described in the next section.

4.2. Two side iterative schemes. Two side iterative schemes have the benefit in balancing between the different operators. Such that we obtain a predictor-corrector scheme that is taken into account to improve a previous solution.

Here the eigenvalues are also computed by the Rayleigh coefficient, means we compute the maximal and minimal eigenvalues of A_1 and B. Then we propose to apply a pre-conditioner or the iterative splitting scheme, that iterates over the larger conditioned matrix.

That is split into the following schemes, where the operators A_1 and B are treated separately in each iterative step (so called two side iterative scheme):

We start with $i = 1$ and $u_0 = v_0 = 0$.

The steps are given for $i > 1$:

First step (respecting A_1)

$$A_1(\nu_{i-1})(u_i, v_i)^t + A_2(h) + B(u_{i-1}, v_{i-1})(u_{i-1}, v_{i-1})^t = 0 \tag{4.41}$$

$$\text{if } \max(||u_i - u_{i-1}||, ||v_i - v_{i-1}||) \leq err \text{ then go to step 2,} \tag{4.42}$$

$$\text{else go to step 1 again} \tag{4.43}$$

where ν_{i-1} is given with u_{i-1}, v_{i-1} for the previous solution.

Second step (respecting B)

$$A_1(\nu_i)(u_i, v_i)^t + A_2(h) + B(u_i, v_i)(u_{i+1}, v_{i+1})^t = 0 \tag{4.44}$$

$$\text{if } \max(||u_{i+1} - u_i||, ||v_{i+1} - v_i||) \leq err \text{ then go to step 1,} \tag{4.45}$$

$$\text{else go to step 2 again} \tag{4.46}$$

where ν_i is given with u_i, v_i in the last step (i-th iterative step) and $err \in I\!R^+$ is a predefined error bound.

REMARK 4.1. *The equations (4.41) and (4.44) can be solved alternatively, if we assume the stopping criteria are fulfilled. Otherwise we solve each equation with more iterations to obtain the given error bound.*

Here from the point of computational time, it makes sense, that we have one dominant equation, e.g. (4.41), such that the main part of the iterative steps are taken via this equation.

Alternation between each iterative scheme is expensive and we should reduce it to less changes, e.g. $5-6$ *iterative steps with (4.41), while only one iterative step with (4.44).*

While the spectrum of the operators are often not the same, it makes sense to investigate more iteration steps over the larger spectrum (means stiffer part), see [19].

And we obtain the next solution u_{i+1}, v_{i+1} which is used to iterative the solution of u_{i+2}, v_{i+2}

The iterative scheme ends with the stop criterion:

$$\max(|u_{i+1} - u_i|, |v_{i+1} - v_i|) \leq err \tag{4.47}$$

where err is a given bound, e.g. $err = 10^{-3}$.

Substitution of the operators yields for the first step

$$\frac{\partial}{\partial x}\left(2\nu_{i-1}H\left(2\frac{\partial u_i}{\partial x} + \frac{\partial v_i}{\partial y}\right)\right) + \frac{\partial}{\partial y}\left(\nu_{i-1}H\left(\frac{\partial u_i}{\partial y} + \frac{\partial v_i}{\partial x}\right)\right) - \rho g H\frac{\partial h}{\partial x} - \mu_{i-1}u_{i-1} = 0 \tag{4.48}$$

$$\frac{\partial}{\partial y}\left(2\nu_{i-1}H\left(2\frac{\partial v_i}{\partial y} + \frac{\partial u_i}{\partial x}\right)\right) + \frac{\partial}{\partial x}\left(\nu_{i-1}H\left(\frac{\partial u_i}{\partial y} + \frac{\partial v_i}{\partial x}\right)\right) - \rho g H\frac{\partial h}{\partial y} - \mu_{i-1}v_{i-1} = 0 \tag{4.49}$$

and for the second step

$$\frac{\partial}{\partial x}\left(2\nu_i H\left(2\frac{\partial u_i}{\partial x}+\frac{\partial v_i}{\partial y}\right)\right)+\frac{\partial}{\partial y}\left(\nu_i H\left(\frac{\partial u_i}{\partial y}+\frac{\partial v_i}{\partial x}\right)\right)-\rho g H\frac{\partial h}{\partial x}-\mu_i u_{i+1}=0 \tag{4.50}$$

$$\frac{\partial}{\partial y}\left(2\nu_i H\left(2\frac{\partial v_i}{\partial y}+\frac{\partial u_i}{\partial x}\right)\right)+\frac{\partial}{\partial x}\left(\nu_i H\left(\frac{\partial u_i}{\partial y}+\frac{\partial v_i}{\partial x}\right)\right)-\rho g H\frac{\partial h}{\partial y}-\mu_i v_{i+1}=0 \tag{4.51}$$

Now we will develop the splitting step in A_1. As for the one side scheme (Equations (4.7) and (4.8), one can iterate Equations (4.48), (4.49) using the weighted Jacobi scheme or, alternatively, utilizing the SOR scheme; but the terms $\mu_{i-1}u_{i-1}$ and $\mu_{i-1}v_{i-1}$ remain explicite in this splitting step.

Redefinition of equations (4.16) and (4.25) yields

$$\beta^0_{l,m} = -\,\mu_{l+1/2,m}, \tag{4.52}$$

$$\beta'^0_{l,m} = -\,\mu_{l,m+1/2}. \tag{4.53}$$

Additionally, we introduce the new coefficients

$$\begin{aligned}\beta^9_{l,m} =& -\frac{4\hat{\nu}_{l+1,m}}{\Delta x_{l+1/2,m}\Delta x_{l+1,m}}-\frac{4\hat{\nu}_{l,m}}{\Delta x_{l+1/2,m}\Delta x_{l,m}}\\ &-\frac{\hat{\nu}_{l+1/2,m+1/2}}{\Delta y_{l+1/2,m}\Delta y_{l+1/2,m+1/2}}-\frac{\hat{\nu}_{l+1/2,m-1/2}}{\Delta y_{l+1/2,m}\Delta y_{l+1/2,m-1/2}},\end{aligned} \tag{4.54}$$

$$\begin{aligned}\beta'^9_{l,m} =& -\frac{4\hat{\nu}_{l,m+1}}{\Delta y_{l,m+1/2}\Delta y_{l,m+1}}-\frac{4\hat{\nu}_{l,m}}{\Delta y_{l,m+1/2}\Delta y_{l,m}}\\ &-\frac{\hat{\nu}_{l+1/2,m+1/2}}{\Delta x_{l,m+1/2}\Delta x_{l+1/2,m+1/2}}-\frac{\hat{\nu}_{l-1/2,m+1/2}}{\Delta x_{l,m+1/2}\Delta x_{l-1/2,m+1/2}}.\end{aligned} \tag{4.55}$$

For the splitting step in A_1 we use the SOR algorithm which is proven to be faster than the Jacobi procedure.

$$\begin{aligned}v^i_{l,m+1/2} =& (1-\omega)v^{i-1}_{l,m+1/2}+\frac{\omega}{\beta'^0_{l,m}}\Big[f_{l,m}-\beta'^1_{l,m}v^{i-1}_{l+1,m+1/2}-\beta'^2_{l,m}v^i_{l-1,m+1/2}\\ &-\beta'^3_{l,m}v^{i-1}_{l,m+3/2}-\beta'^4_{l,m}v^i_{l,m-1/2}-\beta^5_{l,m}u^{i-1}_{l-1/2,m+1}-\beta^6_{l,m}u^{i-1}_{l-1/2,m},\\ &-\beta^7_{l,m}u^i_{l+1/2,m+1}-\beta^8_{l,m}u^i_{l+1/2,m}-\beta'^9_{l,m}v^{i-1}_{l,m+1/2}\Big]\end{aligned} \tag{4.56}$$

$$\begin{aligned}u^i_{l+1/2,m} =& (1-\omega)u^{i-1}_{l+1/2,m}+\frac{\omega}{\beta^0_{l,m}}\Big[f_{l,m}-\beta^1_{l,m}u^{i-1}_{l+3/2,m}-\beta^2_{l,m}u^i_{l-1/2,m}\\ &-\beta^3_{l,m}u^{i-1}_{l+1/2,m+1}-\beta^4_{l,m}u^i_{l+1/2,m-1}-\beta'^5_{l,m}v^{i-1}_{l,m+1/2}-\beta'^6_{l,m}v^{i-1}_{l,m-1/2}\\ &-\beta'^7_{l,m}v^i_{l+1,m+1/2}-\beta'^8_{l,m}v^i_{l+1,m-1/2}-\beta^9_{l,m}u^{i-1}_{l+1/2,m}\Big]\end{aligned} \tag{4.57}$$

with $m = 1, ..., M_y - 1$, $l = 1, ..., M_x - 1$.

Together with the coefficients $\beta^9_{l,m}$ and $\beta'^9_{l,m}$, the central velocities $u_{l+1/2,m}$ and $v_{l,m+1/2}$ are additional terms in equations (4.56) and (4.57) compared to equations (4.38) (4.39) and by the definition of the splitting step in A_1 they are taken from the previous iteration step $i-1$.

The splitting step in B can be expressed straightforwardly by solving equations (4.50), (4.51)

$$\begin{aligned} u_{l+1/2,m,i+1} = & \frac{1}{\mu_{l,m+1/2,i}}\left[\frac{\partial}{\partial x}\left(2\nu_i H\left(2\frac{\partial u_i}{\partial x}+\frac{\partial v_i}{\partial y}\right)\right)\right. \\ & \left.+\frac{\partial}{\partial y}\left(\nu_i H\left(\frac{\partial u_i}{\partial y}+\frac{\partial v_i}{\partial x}\right)\right)-\rho g H\frac{\partial h}{\partial x}\right], \end{aligned} \tag{4.58}$$

$$\begin{aligned} v_{l,m+1/2,i+1} = & \frac{1}{\mu_{l,m+1/2,i}}\left[\frac{\partial}{\partial y}\left(2\nu_i H\left(2\frac{\partial v_i}{\partial y}+\frac{\partial u_i}{\partial x}\right)\right)\right. \\ & \left.+\frac{\partial}{\partial x}\left(\nu_i H\left(\frac{\partial u_i}{\partial y}+\frac{\partial v_i}{\partial x}\right)\right)-\rho g H\frac{\partial h}{\partial y}\right]. \end{aligned} \tag{4.59}$$

Equations (4.58) and (4.59) describes an application of the ordinary Jacobi scheme, which is the weighted Jacobi scheme with $\omega = 1$. In principle, weighting of the velocities at the iteration step i and at the iteration step $i+1$ by equations (4.58) and (4.59) in discretised form would lead to a solution with the weighted Jacobi scheme. Here, we will express the weighted Jacobi algorithm applied to the splitting in B in the framework of the coefficients of the velocities, because equations (4.12) and (4.13) are already discretised.

Again we redefine the coefficients $\beta^0_{l,m}$ and $\beta'^0_{l,m}$ in equations (4.16) and (4.25)

$$\begin{aligned} \beta^0_{l,m} = & -\frac{4\hat{\nu}_{l+1,m}}{\Delta x_{l+1/2,m}\Delta x_{l+1,m}}-\frac{4\hat{\nu}_{l,m}}{\Delta x_{l+1/2,m}\Delta x_{l,m}} \\ & -\frac{\hat{\nu}_{l+1/2,m+1/2}}{\Delta y_{l+1/2,m}\Delta y_{l+1/2,m+1/2}}-\frac{\hat{\nu}_{l+1/2,m-1/2}}{\Delta y_{l+1/2,m}\Delta y_{l+1/2,m-1/2}}, \end{aligned} \tag{4.60}$$

$$\begin{aligned} \beta'^0_{l,m} = & -\frac{4\hat{\nu}_{l,m+1}}{\Delta y_{l,m+1/2}\Delta y_{l,m+1}}-\frac{4\hat{\nu}_{l,m}}{\Delta y_{l,m+1/2}\Delta y_{l,m}} \\ & -\frac{\hat{\nu}_{l+1/2,m+1/2}}{\Delta x_{l,m+1/2}\Delta x_{l+1/2,m+1/2}}-\frac{\hat{\nu}_{l-1/2,m+1/2}}{\Delta x_{l,m+1/2}\Delta x_{l-1/2,m+1/2}} \end{aligned} \tag{4.61}$$

and we redefine the coefficients $\beta^9_{l,m}$ and $\beta'^9_{l,m}$ from equations (4.54) and (4.55) and find

$$\beta^9_{l,m} = -\mu_{l+1/2,m}, \tag{4.62}$$

$$\beta'^9_{l,m} = -\mu_{l,m+1/2}. \tag{4.63}$$

The redefinitions expressed by equations (4.60) to (4.63) are complementary to the redefinitions and new-definitions in equations (4.52) to (4.55).

Reformulation of the equations (4.36) and (4.37) leads to a weighted Jacobi algorithm with the additional terms in $\beta^9_{l,m}$ and $\beta'^9_{l,m}$

$$v^i_{l,m+1/2} = (1-\omega)v^{i-1}_{l,m+1/2} + \frac{\omega}{\beta'^0_{l,m}}\Big[f'_{l,m} - \beta'^1_{l,m}v^{i-1}_{l+1,m+1/2} - \beta'^2_{l,m}v^{i-1}_{l-1,m+1/2}$$
$$- \beta'^3_{l,m}v^{i-1}_{l,m+3/2} - \beta'^4_{l,m}v^{i-1}_{l,m-1/2} - \beta^5_{l,m}u^{i-1}_{l-1/2,m+1} - \beta^6_{l,m}u^{i-1}_{l-1/2,m}$$
$$- \beta^7_{l,m}u^{i-1}_{l+1/2,m+1} - \beta^8_{l,m}u^{i-1}_{l+1/2,m} - \beta'^9_{l,m}v^{i-1}_{l,m+1/2}\Big]$$
$$u^i_{l+1/2,m} = (1-\omega)u^{i-1}_{l+1/2,m} + \frac{\omega}{\beta^0_{l,m}}\Big[f_{l,m} - \beta^1_{l,m}u^{i-1}_{l+3/2,m} - \beta^2_{l,m}u^{i-1}_{l-1/2,m}$$
$$- \beta^3_{l,m}u^{i-1}_{l+1/2,m+1} - \beta^4_{l,m}u^{i-1}_{l+1/2,m-1} - \beta'^5_{l,m}v^{i-1}_{l,m+1/2} - \beta'^6_{l,m}v^{i-1}_{l,m-1/2}$$
$$- \beta'^7_{l,m}v^{i-1}_{l+1,m+1/2} - \beta'^8_{l,m}v^{i-1}_{l+1,m-1/2} - \beta^9_{l,m}u^{i-1}_{l+1/2,m}\Big]$$

with $m = 1, ..., M_y - 1$, $l = 1, ..., M_x - 1$.

Separation of A_1 into 2 operators

In the next we separate into $A_1 = A_{11} + A_{12}$, where $A_{11} = diag(A_1)$ or $A_{11} = trigiag(A_1)$ and the rest matrix $A_{12} = A_1 - diag(A_1)$ or $A_{12} = A_1 - trigiag(A_1)$. The idea is to save computational time in the iterative steps with operator A_{11}, where the relaxation step when A_1 is explicitly given is less computative.

We start with $i = 1$ and $u_0 = v_0 = 0$.

The steps are given for $i > 1$:

First step (respecting A_1)

$$A_{11}(\nu_{i-1})(u_i, v_i)^t + A_{12}(\nu_{i-1})(u_{i-1}, v_{i-1})^t$$
$$+A_2(h) + B(u_{i-1}, v_{i-1})(u_{i-1}, v_{i-1})^t = 0 \quad (4.64)$$
$$\text{if } \max(||u_i - u_{i-1}||, ||v_i - v_{i-1}||) \leq err \text{ then go to step 2,} \quad (4.65)$$
$$\text{else go to step 1 again} \quad (4.66)$$

where ν_{i-1} is given with u_{i-1}, v_{i-1} for the previous solution.

Second step (respecting B)

$$A_1(\nu_i)(u_i, v_i)^t + A_2(h) + B(u_i, v_i)(u_{i+1}, v_{i+1})^t = 0 \quad (4.67)$$
$$\text{if } \max(||u_{i+1} - u_i||, ||v_{i+1} - v_i||) \leq err \text{ then go to step 1,} \quad (4.68)$$
$$\text{else go to step 2 again} \quad (4.69)$$

where ν_i is given with u_i, v_i in the last step (i-th iterative step) and $err \in I\!R^+$ is a predefined error bound.

5. Numerical experiments.

In the following we start with our numerical experiments.

5.1. Validation of the schemes with the MacAyeal schematic setup.

In most of the literature about ice streams (e.g. [31], [25]), the SSAB equations are part of inverse modeling of a basal friction parameter from the measured ice surface velocity. Without knowledge of the basal friction, a validation of our model via measured velocities would be difficult. Here, we will not perform a strong model validation in the sense that we assess the modeling errors by giving numbers. We rather make use of the schematic setup for the ice topography and basal friction (Figure 5.1) by [30], a tutorial on control methods. As a byproduct, [30] computes the velocity fields and we use the plots of these velocities for a comparison with our modeled velocities just by eye. At this stage of development, it is not our intention to perform a complete model validation and/or verification as described in [4]. Such an approach would imply the knowledge of an exact solution of the two dimensional SSAB equations, which is not available yet.

Instead of the non-linear relation for the basal shear stress in equation (1.4) (formally $p = 0$ therein) the linear relation

$$\begin{aligned} \tau^{x(b)} &= c_s(x,y)\, u, \\ \tau^{y(b)} &= c_s(x,y)\, v, \\ c_s(x,y) &= \mu(x,y) > 0 \end{aligned} \tag{5.1}$$

is applied by [30].

As boundary conditions Dirichlet conditions apply, with $u = 0$, $v = 0$ at the lateral boundaries of the stream, $u = 0$ at the influx and outflux line of the stream. The velocity v at the influx and outflux line is defined by a linear combination of sinusoidal functions, which higher velocities at outflux compared to that at influx line, see [30] for more details.

For the average inverse rate factor $\bar{B} = 1.8 \times 10^8$ Pa s m^{-1} applies.

Figure 5.2 shows the velocity fields computed with our weighted Jacobi method. As is reality, the ice-stream velocity is dominated by its longitudinal component. Its magnitude increase towards the middle of the ice stream as well as in its downstream direction. The different signs of the traverse velocity component component reflect the undulations in the surface elevation, brought into via the driving stress which contains the gradient of the surface elevation. Our simulated surface velocities compare well with those in former publications. When we used a plot on (sufficient transparent) paper with the same scale and orientation as in [30], we were able to match our isolines with those in the printed publication by that worker. The respective fields yielded with the SOR or the splitting method are not displayed explicitely here, because they are almost identical with those found with the weighted Jacobi method.

In summary, our simulated velocity fields compare excellent with those by [30]. Therefore, we trust our code and will use it for further studies hereafter.

Remark 5.1. *We could find the right ω for Jacobi (as promised above) and SOR in numerical examples. Here the condition of matrix A_1 is important, while this matrix is stiffer than B, see Figure 5.3. Such problems can be solved by investigating a finer spatial resolution. Here e have a strong dependence of the convergence on the spatial resolution and we are motivated to nearer inspect the eigenvalues of the operator A_1 (5.2) which is stiff.*

5.2. Eigenvalues of the SSAB equations.

We can demonstrated that the SOR scheme leads to faster convergence and reduced run times compared to the Jacobi scheme.

Here, we show that equations (4.41) and (4.44) are indeed a reasonable splitting due to the by far larger condition of the matrix B compared to the matrix A_1. We determine the conditions of the matrices via their maximal and minimal eigenvalues (equations 2.24, 2.25) with the free software OCTAVE
(http://www.gnu.org/software/octave/). The condition of the respective matrices show a strong dependence on the iteration steps (Figure 5.3). But the condition of the matrix A_1 is for nearly all iteration steps two orders of magnitude higher that the

TABLE 5.1
Performance of the different numerical methods with applied weights of the operators.

Method	ω^A	ω^{A_1}	ω_u^B	ω_v^B	Run Time (s)
Jacobi	0.7	-	-	-	8.2
SOR	1.4	-	-	-	2.5
A_1-B splitting	-	1.2	0	0.01	1.7

condition of the matrix B; for the very first iteration step the condition of the matrix A_1 is still about a factor of 20 higher compared to that of matrix B. These very clear results motivate a stronger iteration on the matrix A_1 that on the matrix B with fixed iteration steps for the splittings; i.e., for now we assume that the dominance of matrix A_1 in condition will never change.

REMARK 5.2. *The improvement can be done with the A_1-B splitting, while we taken into account more iteration steps to operator A_1 per splitting step. Such more investigation to smooth the resolution of operator A_1 can also be improved by an optimal ω.*

5.3. Performance studies for weighed Jacobi, SOR and the A_1-B splitting. The computations were performed an the IBM iDataPlex Cluster (IPLEX). Table 5.1 shows the run time yielded with the weights which we found before. To avoid interference with different nodes of the machine all nodes which share the same memory except for one are switched off. We checked the accuracy of the run time with the same numerical method for different nodes on the IPLEX and found that the run times compares by far as accurate the two significant digits given in table 5.1.

The Figure 5.4 presents the iteration steps, that are needed to have convergent results. The best results are obtained with the proposed weighted iterative schemes.

6. Conclusions and discussions. We present an iterative operator-splitting method to solve partial differential equations with respect to their underlying time scales. The correct splitting into the underlying operators of the equations is important to reduce the splitting error and contribute an efficient method. Here the eigenvalues of each operators are an indicator for their stiffness in the scheme. We have embed the computation of the eigenvalues and investigate more iterative steps over the stiffer operators. Thus, we balance with a weighting factor in the solver scheme the optimal conditions of the operators. Therefore we present an embedded eigenvalue solver and accelerate with a weighted scheme. First numerical results can validate the correct splitting and the efficiency. Optimal results are obtained with an optimal balance of sufficient iterative steps and the weighting factor in the solver scheme. In future it will be important to have efficient eigenvalue methods, which can be embedded into the splitting methods, to contribute the operator-splitting methods as efficient solver methods for large evolution equations.

Acknowledgments. R. Calov and T. Recknagel are funded by the Deutsche Forschungsgemeinschaft RA 977/6-1.

7. Appendix: Runge-Kutta, BDF and IMEX methods. For the time-discretization of the split equation, the combination of accurate methods, that will fit in the higher-order context of the iterative operator-splitting methods, is important.

Based on the iterative methods the start solution for the first iteration step is important to obtain higher-order results. For the next iteration steps the order has to increase until the proposed order of the time-discretization is achieved.

Therefore we propose the Runge-Kutta and BDF methods as adapted time-discretization methods to reach higher-order results.

For the time-discretization we use the following higher-order discretization methods.

7.1. Runge-Kutta method. We use the implicit trapezoidal rule:

$$\begin{array}{c|cc} 0 & & \\ 1 & \frac{1}{2} & \frac{1}{2} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array} \quad . \tag{7.1}$$

Furthermore we use the following Gauß-Runge-Kutta method:

$$\begin{array}{c|cc} \frac{1}{2}-\frac{\sqrt{3}}{6} & \frac{1}{4} & \frac{1}{4}-\frac{\sqrt{3}}{6} \\ \frac{1}{2}+\frac{\sqrt{3}}{6} & \frac{1}{4}+\frac{\sqrt{3}}{6} & \frac{1}{4} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array} \quad . \tag{7.2}$$

To use these Runge-Kutta methods with our operator-splitting method we have to take into account that we solve in each iteration step equations of the form $\partial_t u_i = Au_i + b$, where $b = Bu_{i-1}$ is a discrete function, as we only have a discrete solution for u_{i-1}.

For the implicit trapezoidal rule this is no problem, because we do not need the values at any sub-points. However, for the Gauß method we need to now the values of b at the sub-points $t_0 + c_1 h$ and $t_0 + c_2 h$ with $c = (\frac{1}{2} - \frac{\sqrt{3}}{6}, \frac{1}{2} + \frac{\sqrt{3}}{6})^T$. Therefore we must interpolate b. On that account we choose the cubic spline functions.

Numerical experiments show that this works properly with non-stiff problems, but not very well with stiff problems.

7.2. BDF method. Because the higher-order Gauß-Runge-Kutta method combined with cubic spline interpolation does not work properly with stiff problems, we use the following BDF method of order three, which does not need any sub-points and therefore no interpolation is needed.

The BDF3 method is defined by

$$\frac{1}{k}(\frac{11}{6}u^{n+2} - 3u^{n+1} + \frac{3}{2}u^n - \frac{1}{3}u^{n-1} = A(u^{n+3}). \tag{7.3}$$

For the pre-stepping, i.e. to obtain u_1, u_2, we use the implicit trapezoidal rule (7.1).

7.3. Implicit-explicit methods. The implicit-explicit (IMEX) schemes have been widely used for time integration of spatial discretised partial differential equations of diffusion-convection type. These methods are applied to decouple the implicit and explicit terms. Treating the convection-diffusion equation for example, one can use the explicit part for the convection and the implicit part for the diffusion term.

In our application we divide between the stiff and non-stiff term, so we apply the implicit part for the stiff operators and the explicit part for the non-stiff operators.

7.3.1. FSRK method. We propose the A-stable fractional-stepping Runge-Kutta (FSRK) scheme, see [6], of first and second order for our applications.
The tableau in the Butcher form is given as

$$
\begin{array}{c|cccc|cccc}
1 & 1 & & & & 0 & & & \\
1 & 1 & 0 & & & 0 & 1 & & \\
\frac{4}{9} & -\frac{88}{45} & 0 & \frac{12}{5} & 0 & 0 & \frac{5}{9} & 0 & \\
\frac{1}{3} & -\frac{407}{75} & 0 & -\frac{144}{25} & 0 & 0 & -\frac{31}{15} & 0 & \frac{12}{5} \\
\hline
\text{order1} & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
\hline
\text{order2} & \frac{1}{10} & 0 & \frac{9}{10} & 0 & 0 & \frac{1}{4} & 0 & \frac{3}{4}
\end{array} . \tag{7.4}
$$

To obtain second-order convergence in numerical examples it is important to split the operator in the right way as we will show later.

7.3.2. SBDF Method. We use the following stiff backward differential formula (SBDF) method, which is a modification of the third-order backward differential formula (BDF3) method.
As pre-stepping method we use again the implicit trapezoidal rule.

$$
\begin{aligned}
&\frac{1}{k}(\frac{11}{6}u^{n+1} - 3u^n + \frac{3}{2}u^{n-1} - \frac{1}{3}u^{n-2}) \\
&= 3A(u^n) - 3A(u^{n-1}) + A(u^{n-2}) + B(u^{n+1}).
\end{aligned} \tag{7.5}
$$

Again it is important to split the operator in the right way.

REFERENCES

[1] Z. S. Alterman and A. Rotenberg *Seismic Waves in a Quarter Plane.* Bulletin of the Seismological Society of America, 59:347–368, 1969.

[2] W. F. Ames *Numerical Methods for Partial Differential Equations* Academic Press, San Diego, 1992

[3] A. Arakawa *Design of the UCLA general circulation model* Technical Report No. 7, Department of Meteorology, University of California, 1972.

[4] E. Bueler, C. S. Lingle, J. A. Kallen-Brown, D. N. Covey, L. N. Bowman. *Exact solutions and verification of numerical models for isothermal ice sheets* J. Glaciol., 51, 173, 291–306, 2005.

[5] E. Bueler and J. Brown *Shallow shelf approximation as a "sliding law" in a thermomechanically coupled ice sheet model* J. Geophys. Res., 114, F03008, DOI:10.1029/2008JF001179, 2009.

[6] J.C. Butcher. Numerical Methods for Ordinary Differential Equations. *John Wiley & Sons Ltd, Chichester*, 2003.

[7] St.M. Day et al. *Test of 3D elastodynamic codes: Final report for lifelines project 1A01.* Technical report, Pacific Earthquake Engineering Center, 2001.

[8] St.M. Day et al. *Test of 3D elastodynamic codes: Final report for lifelines project 1A02.* Technical report, Pacific Earthquake Engineering Center, 2003.

[9] St. Descombes. *Convergence of a splitting method of high order for reaction-diffusion systems.* Mathematics of Computations, 70, 1481–1501, 2001.

[10] G. Fairweather and A.R. Mitchell. *A high accuracy alternating direction method for the wave equations.* J. Industr. Math. Appl., Vol.1, 309–316, 1965.

[11] I. Farago. *Modified iterated operator splitting method.* Applied Mathematical Modelling, Elsevier Science, 2007, (to be reviewed).

[12] I. Farago. *Splitting methods for abstract Cauchy problems.* Lect. Notes Comp.Sci. 3401, Springer Verlag, Berlin, 2005, pp. 35–45

[13] I. Farago, J. Geiser, Iterative Operator-Splitting methods for Linear Problems, Preprint No. 1043 of Weierstrass Institute for Applied Analysis and Stochastics, Berlin, Germany, 2005.

[14] J. Geiser. *Numerical Simulation of a Model for Transport and Reaction of Radionuclides.* Proceedings of the Large Scale Scientific Computations of Engineering and Environmental Problems, Sozopol, Bulgaria, 2001.

[15] J. Geiser. *Decomposition methods for partial differential equations: Theory and Applications in multi physics problems.* Habilitation-thesis, Humboldt Universität zu Berlin, Germany, to be submitted, 2007.

[16] J. Geiser, St. Nilsson. *A Fourth Order Split Scheme for Elastic Wave Propagation.* Preprint 2007-08 of Humboldt Universität zu Berlin, Department of Mathematics, Germany, 2007.

[17] J. Geiser. *Operator-Splitting Methods in Respect of Eigenvalue Problems for Nonlinear Equations and Applications to Burgers Equations.* Journal of Computational and Applied Mathematics, Elsevier, Amsterdam, North Holland, accepted May 2009.

[18] R. Greve, and H. Blatter. *Dynamics of Ice Sheets and Glaciers* Springer, Berlin, Germany etc., 2009.

[19] W. Hackbusch. *Multi-Grid Methods and Applications.* Springer-Verlag, Berlin, Heidelberg, 1985.

[20] W. Hackbusch. *Theorie und Numerik elliptischer Differentialgleichungen.* Teubner-Verlag, Stuttgart, 1986.

[21] E. Hairer and G. Wanner. *Solving Ordinary Differential Equatons II.* SCM, Springer-Verlag Berlin-Heidelberg-New York, No. 14, 1996.

[22] R.C.A. Hindmarsh. *A numerical comparison of approximations to the Stokes equations used in ice sheet and glacier modeling* J. Geophys. Res., 109(F1), F01012, DOI: 10.1029/2003JF000065, 2004.

[23] J.R. Holten. *An introduction to dynamic meteorology* 2nd edition, International Geophysics Series, Vol 23, Academic Press, New York, 1978.

[24] W.H. Hundsdorfer, J. Verwer *Numerical solution of time-dependent advection-diffusion-reaction equations*, Springer, Berlin, (2003).

[25] I. Joughin et al. *Observation and analysis of ice flow in the largest Greenland ice stream* J. Geophys. Res., 106, D24, 34,021–34,034, 2001.

[26] J. Kanney, C. Miller and C. Kelley. *Convergence of iterative split-operator approaches for approximating nonlinear reactive transport problems.* Advances in Water Resources, 26, 247–261, 2003.

[27] S. Karaa. *An accurate LOD scheme for two-dimensional parabolic problems.* Applied mathematics and computation, Vol. 170, 886–894, 2005.

[28] K.H. Karlsen and N. Risebro. *An Operator Splitting method for nonlinear convection-diffusion equation.* Numer. Math., 77, 3, 365–382, 1997.

[29] M. Lees. *Alternating direction methods for hyperbolic differential equations.* J. Soc. Industr. Appl. Math., Vol. 10, No. 4, 610–616, 1962.

[30] D.R. MacAyeal. *A tutorial in the use of control methods in ice-sheet modeling.* J. Glaciol., 39, 131, 91–98, 1993.

[31] D.R. MacAyeal et al. *Basal friction of Ice Stream E, West Antarctica* J. Glaciol., 41, 138, 247–262, 1995.

[32] G.I. Marchuk. *Some applicatons of splitting-up methods to the solution of problems in mathematical physics.* Aplikace Matematiky, 1 (1968) 103–132.

[33] D. Pollard, R.M. DeConto. *A coupled ice-sheet/ice-shelf/sediment model applied to a marine-margin flowline: Forced and unforced variations*, in Glacial Sedimentary Processes and Products (eds. M. Hambrey, P. Christoffersen, N. Glasser, and B. Hubbard) 37-52 (Spec. Publ. 39, International Association of Sedimentologists, Blackwell Publishing, 2007).

[34] G. Strang. *On the construction and comparision of difference schemes.* SIAM J. Numer. Anal., 5:506–517, 1968.

[35] J.W. Thomas. *Numerical Partial Differential Equations: Conservation Laws and Elliptic Equations* Springer-Verlag, New York, Berlin, Heidelberg, 1999.

[36] R.S. Varga. *Matrix Iterative Analysis* Springer-Verlag, New York, Berlin, Heidelberg, 2000.

[37] R.J. LeVeque. *Finite Difference Methods for Ordinary and Partial Differential Equations* SIAM, Philadelphia, 2007.

[38] E. Zeidler. *Nonlinear functional analysis and its applications II/B: Nonlinear Monotone Operators* Springer-Verlag, New York, 1990.

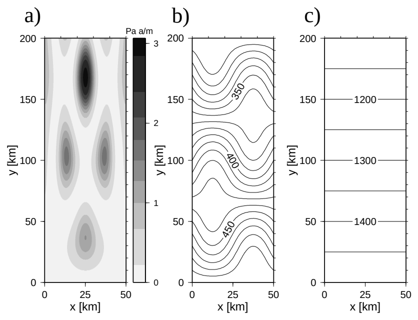

Fig. 5.1. *MacAyeal's schematic model setup. (a) shading plot of basal friction parameter $c_s(x,y)$ in Pa a m^{-1}, isolines of (b) ice surface elevation in m and (c) ice thickness in m.*

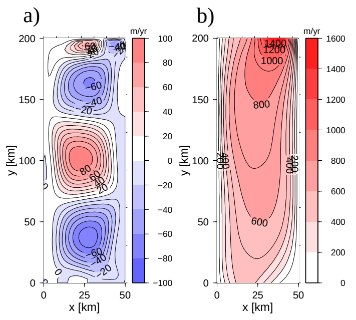

Fig. 5.2. *Simulated velocity. (a) traverse velocity u and (b) longitudinal velocity v in m/a.*

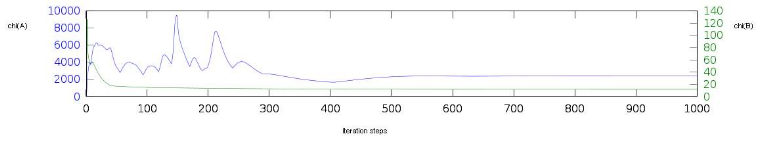

Fig. 5.3. *Conditions of A_1 and B of the SSAB equation.*

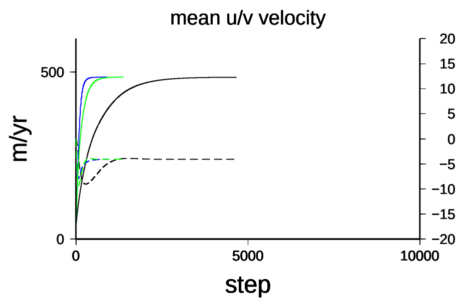

Fig. 5.4. *Convergence of the schemes with respect to the iterative steps. The weighted schemes (green and blue) are faster.*