Appendix

A.1 Shallow water equation formulation for the numerical model

A.1.1 Boundary conditions

The boundary conditions used in the shallow water model (equations 2.1) are explicitly given in the following. The kinematic boundary condition states that there is no flow through the boundary, i.e.

$$u(x=0) = u(x=L_x) = v(y=0) = v(y=L_y) = 0. \tag{A.1}$$

Additionally, for numerical convenience, we require no gradient of η across the boundary, hence

$$\partial_x \eta(x=0) = \partial_x \eta(x=L_x) = \partial_y \eta(y=0) = \partial_y \eta(y=L_y) = 0, \tag{A.2}$$

The tangential velocity at the boundary should vanish, which is usually referred to as no-slip boundary conditions

$$v(x=0) = v(x=L_x) = 0, u(y=0) = u(y=L_y) = 0. \tag{A.3}$$

However, sometimes these equations are replaced by

$$\partial_x v(x=0) = \partial_x v(x=L_x) = \partial_y u(y=0) = \partial_y u(y=L_y) = 0, \tag{A.4}$$

which are called *free-slip* boundary conditions, omitting a flux of momentum through the boundary, which are here given only for completeness but not used in this study.

A.1.2 Reformulation with Bernoulli potential and potential vorticity

The prognostic variables in the shallow water equations 2.1 so far are u, v, η, however, as η only appears in gradients, also h can be used as prognostic variable. We can separate from the advective terms in equations 2.1a,b the spatial gradient of kinetic energy. In combination with the pressure gradient term we introduce the Bernoulli potential p as

$$p = \frac{1}{2}(u^2 + v^2) + gh \tag{A.5}$$

Furthermore, with the relative vorticity $\zeta = \partial_x v - \partial_y u$ the potential vorticity can be defined as

$$q = \frac{f + \zeta}{h} \tag{A.6}$$

and equivalent to the shallow water equations we have

$$\begin{aligned}
\partial_t u &= qhv - \partial_x p + F_x + M_x + B_x + \xi_x && \text{(A.7a)}\\
\partial_t v &= -qhu - \partial_y p + F_y + M_y + B_y + \xi_y && \text{(A.7b)}\\
\partial_t \eta &= -\partial_x(uh) - \partial_y(vh) && \text{(A.7c)}
\end{aligned}$$

which are the equations solved by the numerical model as described in Appendix A.2.

A.2 Discretization of the shallow water model

In the following we will describe the discretization of equation A.7 in space and time. This discretization is then implemented into a numerical model code that can be found at `www.github.com/milankl/swm` and forms the numerical shallow water model as used throughout this study.

A.2.0 Notation

Some remarks on the notation used to describe the discretization of the shallow water model.

Surrounding grid points A superscript arrow points in the direction relative to the grid point where an operation is evaluated. More clearly, regarding a variable a at $\mathbf{x} = (x_a, y_a)$ somewhere in the middle of the domain away from the boundaries. Then,

$$b^{\leftarrow} = b\big|_{\mathbf{x}=(x_a-\frac{1}{2}\Delta x, y_a)}, \quad b^{\rightarrow} = b\big|_{\mathbf{x}=(x_a+\frac{1}{2}\Delta x, y_a)} \tag{A.8}$$

where b is a variable that sits on the grid that is shifted by $\frac{1}{2}\Delta x$ in x-direction, and

$$c^{\downarrow} = c\big|_{\mathbf{x}=(x_a, y_a-\frac{1}{2}\Delta y)}, \quad c^{\uparrow} = c\big|_{\mathbf{x}=(x_a, y_a-\frac{1}{2}\Delta y)} \tag{A.9}$$

where c is a variable that sits on the grid that is shifted by $\frac{1}{2}\Delta y$ in y-direction, and

$$d^{\nearrow} = d\big|_{\mathbf{x}=(x_a+\frac{1}{2}\Delta x, y_a+\frac{1}{2}\Delta y)}, \quad d^{\searrow} = d\big|_{\mathbf{x}=(x_a+\frac{1}{2}\Delta x, y_a-\frac{1}{2}\Delta y)} \tag{A.10a}$$

$$d^{\swarrow} = d\big|_{\mathbf{x}=(x_a-\frac{1}{2}\Delta x, y_a-\frac{1}{2}\Delta y)}, \quad d^{\nwarrow} = d\big|_{\mathbf{x}=(x_a-\frac{1}{2}\Delta x, y_a+\frac{1}{2}\Delta y)} \tag{A.10b}$$

where d is a variable that sits on the grid that is shifted by both $\frac{1}{2}\Delta x$ in x-direction and $\frac{1}{2}\Delta y$ in y-direction. The notation is therefore independent of the indexing. For the description of the advective terms (section A.2.4) we may relate grid points that are further away by

$$b^{\Leftarrow} = b|_{\mathbf{x}=(x_a-\Delta x, y_a)}, \quad b^{\Rightarrow} = b|_{\mathbf{x}=(x_a+\Delta x, y_a)} \tag{A.11}$$

and similar for $\Uparrow, \Downarrow$. The single arrow therefore represents a grid point that is either $(\pm n\Delta x, 0)$, $(0, \pm n\Delta y)$ or $(\pm n\Delta x, \pm n\Delta y)$, with $n = \frac{1}{2}$, away. Same for the double arrow, but with $n = 1$. This notation is logically extended for a triple arrow $\Uparrow\uparrow$ (i.e. $n = \frac{3}{2}$), and so on.

The operator stencils usually change close to the boundary due to boundary conditions. In order to treat these cases separately we introduce the following, index-independent notation: Call a grid node northern boundary (NB), western boundary (WB), southern boundary (SB) or eastern boundary (EB), when the evaluation of a stencil for that grid node involves unresolved variables, because they sit either outside the domain $\mathcal{D}$ or on the boundary, where they are given by the boundary conditions.

Note that there is an overlap of two adjacent boundaries that we call accordingly north-east corner (NE), north-west corner (NW), south-west corner (SW) or south-east corner (SE). This notation is unfortunately only

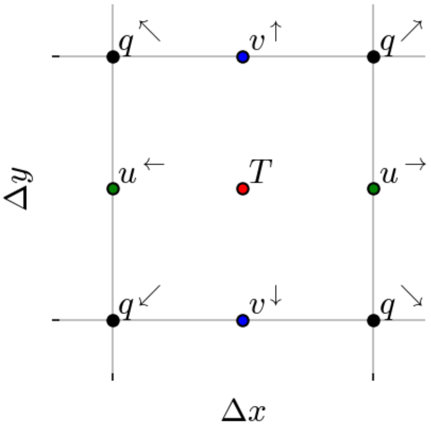

Figure 1: Illustrating the idea of an arrow-based notation for surrounding grid points. Here, the grid points around a grid point on the T-grid.

useful when stencils are small, but then provide a readable alternative.

Non-linear operations As the equations of interest are non-linear we cannot describe all operations in the model in terms of matrix-vector multiplications, additions of vectors or multiplications of a scalar with a vector. In fact, it turns out that all non-linear operations in equation (2.1), once discretized, are element-wise vector-vector multiplications. Let $\mathtt{a}, \mathtt{b}$ be two vectors of same length N, respectively element of a vector space V. Hence, they should sit on the same grid. We then define the element-wise vector-vector multiplication $*$ as

$$* : V \times V \to V, \mathtt{a}*\mathtt{b} \to \begin{pmatrix} a_1 b_1 \\ a_2 b_2 \\ \vdots \\ a_N b_N \end{pmatrix}, \quad \text{with } \mathtt{a} = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_N \end{pmatrix}, \quad \mathtt{b} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_N \end{pmatrix} \tag{A.12}$$

and define the order of computation as inferior to matrix-vector multiplication, i.e.

$$\mathbf{A}\mathtt{b} * \mathtt{c} = (\mathbf{A}\mathtt{b}) * \mathtt{c} \neq \mathbf{A}(\mathtt{b} * \mathtt{c}) \tag{A.13}$$

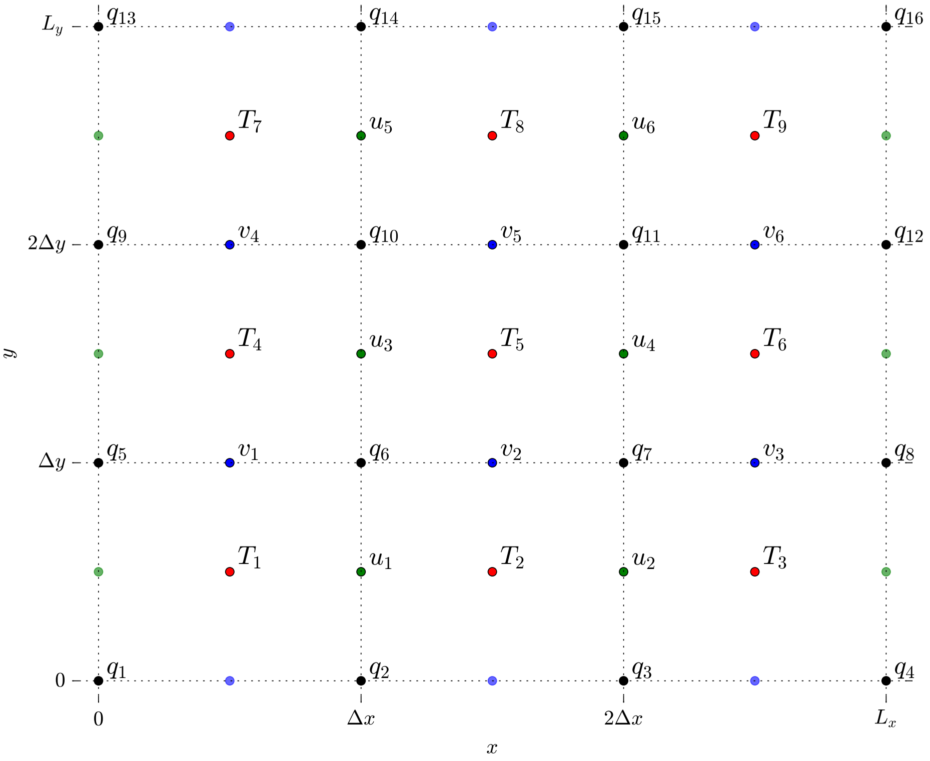
L_y
$2\Delta y$
Δy
0
y
0
Δx
$2\Delta x$
L_x
x
q_1 q_2 q_3 q_4
T_1 u_1 T_2 u_2 T_3
q_5 v_1 q_6 v_2 q_7 v_3 q_8
T_4 u_3 T_5 u_4 T_6
q_9 v_4 q_{10} v_5 q_{11} v_6 q_{12}
T_7 u_5 T_8 u_6 T_9
q_{13} q_{14} q_{15} q_{16}

Figure 2: Row-first indexing on the Arakawa C-grid for an example grid with $n_x = n_y = 3$. The variables on the boundary as denoted by faint green and blue discs are not explicitly resolved but given by the boundary conditions. In contrast, the values of q in the corners (i.e. q_1, q_4, q_{13} and q_{16} here) are numbered for simplicity although always being 0 (also referred to as *ghost points*).

for all matrices $\mathbf{A}$, and vectors $\mathtt{b}, \mathtt{c}$. Furthermore, we write

$$\frac{1}{\mathtt{a}} = \begin{pmatrix} \frac{1}{a_1} \\ \frac{1}{a_2} \\ \vdots \\ \frac{1}{a_N} \end{pmatrix} \tag{A.14}$$

to denote the element-wise multiplicative inverse of $\mathtt{a}$.

A.2.1 Arakawa C-Grid

The domain $\mathcal{D}$ is divided into $n_x \times n_y$ grid cells, evenly spaced, so that each grid cell has the side length $\Delta x = \frac{L_x}{n_x}$ in x-direction and $\Delta y = \frac{L_y}{n_y}$ in y-direction. The total amount of grid cells is $n_x n_y$. Let the variable η_i sit in the middle of the i-th grid cell at position (x_i, y_i), i.e. $\eta_i = \eta(x_i, y_i)$. We use only one index i so that $i \in \{1, ..., n_x n_y\}$ in order to index all grid cells. We choose row-first indexing as in Fig. 2, which is used throughout the numerical model code.

Following the ideas of the Arakawa C-grid [Arakawa & Lamb, 1977, 1981], the discretization of the variables u, v and q is staggered. In general, we might use an independent indexing for u, v and q and therefore introduce $j, k, l \in \mathbb{N}$. We distinguish between 4 different grids: (i) the T-grid, for η, h, (ii) the u-grid, (iii) the v-grid, (iv) the q-grid. Not for all grid cells it is necessary to evaluate u or v, as they might vanish due to the boundary conditions. The grids therefore carry a different amount of grid points. Let N_T, N_u, N_v, N_q be the total number of grid points on the respective grids then

$$\begin{aligned} N_T &= n_x n_y, & N_u &= (n_x - 1) n_y \\ N_q &= (n_x + 1)(n_y + 1), & N_v &= n_x (n_y - 1) \end{aligned} \tag{A.15}$$

The n_x-th column of u-points vanish, as does the n_y-th row of v-points. However, there is no boundary condition for q, which makes it necessary to evaluate the q-grid for all points within the domain $\mathcal{D}$.

Choosing one index for the grid points leads to the advantage that every scalar variable can be represented as a vector with the following vector-representation $\mathtt{u}, \mathtt{v}, \mathtt{h}, \mathtt{q}$ of u, v, h and q

$$\mathtt{u} = \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_{N_u} \end{pmatrix}, \quad \mathtt{v} = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_{N_v} \end{pmatrix}, \quad \mathtt{h} = \begin{pmatrix} h_1 \\ h_2 \\ \vdots \\ h_{N_T} \end{pmatrix}, \quad \mathtt{q} = \begin{pmatrix} q_1 \\ q_2 \\ \vdots \\ q_{N_q} \end{pmatrix}. \tag{A.16}$$

In the special case of $n_x = n_y$ the vectors $\mathtt{u}, \mathtt{v}$ are of same size, but in general $\mathtt{u}, \mathtt{v}, \mathtt{h}, \mathtt{q}$ all differ in their sizes. Unfortunately, the vector-representation of the variables leads to complicated distinguishing between grid nodes that are in vicinity of the boundaries and those in the middle of the domain. In the following, we will evaluate the model variables on different grids and denote that with $\mathbf{x}_i$, $\mathbf{x}^u_j$, $\mathbf{x}^v_k$ and $\mathbf{x}^q_l$, which is short-hand for all grid points of the T-, u-, v- and q-grid, respectively.

A.2.2 Gradients

Matrix-Vector multiplication idea

Representing the model variables in vector-form, as discussed above, enables us to think of a gradient ∂ as a linear map between two vector spaces[1] V_1, V_2. Hence, any gradient ∂ in that sense can be written as a matrix $\mathbf{G}$ which is multiplied with a vector $\mathtt{z}$ representing one of the model's variables:

$$\partial : V_1 \to V_2, \mathtt{z} \to \mathbf{G}\mathtt{z} \tag{A.17}$$

Having 4 different grids, we have to deal with four different vector spaces V_u, V_v, V_T, V_q with dimensions N_u, N_v, N_T, N_q, respectively. In the following we will investigate the entries of $\mathbf{G}$ such that they describe centred finite differences on the four different grids. The same holds for any linear interpolation $\mathcal{I}$ from one grid to another as will be discussed in section A.2.3. We write then

$$\mathcal{I} : V_1 \to V_2, \mathtt{z} \to \mathbf{I}\mathtt{z} \tag{A.18}$$

Gradient operation as matrix multiplication

In the following we will use a notation where the subscript denotes the direction of the derivative, hence x or y, and the superscript u, v, T, q denotes the vector space, where ∂ is mapping from. For the x-derivative of h on the T-grid this is

$$\partial^T_x : V_T \to V_u, \mathtt{h} \to \mathbf{G}^T_x\mathtt{h} \tag{A.19}$$

and the result $\mathbf{G}^T_x\mathtt{h}$ sits then on the u-grid (as defined in equation A.21) or equivalently is element of the vector space V_u. The matrix $\mathbf{G}^T_x$ is therefore

[1] Strictly speaking, these are not vector spaces as the underlying algebraic field are not the real numbers but the set of computer representable numbers given a certain precision, e.g. 64bit. These are mathematically not an algebraic field, but for simplicity here regarded as an approximation to it.

of size $N_u \times N_T$ and so not a square matrix (see eq. A.15). The derivatives $\partial_y h, \partial_x u, \partial_y u, \partial_x v, \partial_y v, \partial_x q$ and $\partial_y q$ can similarly be written as

$$\partial_y^T : V_T \to V_v, \mathtt{h} \to \mathbf{G}_y^T \mathtt{h} \tag{A.20a}$$

$$\partial_x^u : V_u \to V_T, \mathtt{u} \to \mathbf{G}_x^u \mathtt{u} \tag{A.20b}$$

$$\partial_y^u : V_u \to V_q, \mathtt{u} \to \mathbf{G}_y^u \mathtt{u} \tag{A.20c}$$

$$\partial_x^v : V_v \to V_q, \mathtt{v} \to \mathbf{G}_x^v \mathtt{v} \tag{A.20d}$$

$$\partial_y^v : V_v \to V_T, \mathtt{v} \to \mathbf{G}_y^v \mathtt{v} \tag{A.20e}$$

$$\partial_x^q : V_q \to V_v, \mathtt{q} \to \mathbf{G}_x^q \mathtt{q} \tag{A.20f}$$

$$\partial_y^q : V_q \to V_u, \mathtt{q} \to \mathbf{G}_y^q \mathtt{q} \tag{A.20g}$$

All entries of the gradient matrices **G** follow from the equations A.21, A.22, A.23, A.24, A.25, A.29 and A.30 and are then only a matter of indexing.

Centred differences

Writing the scalar variables on a grid as vectors comes with the disadvantage, that the exact mathematical indexing gets rather complicated and lacks readability. We will therefore discuss the gradient and interpolation operations in the following only in terms of their stencil, i.e. the linear combination of the variables at the surrounding grid points that yields the desired result. To be a bit more precise and to include the treatment of the boundaries, we will use the notation described in section A.2.0.

Centred finite difference approximation of the gradient in x-direction of a variable η on the T-grid yields a result that sits on the u-grid and reads

$$\partial_x \eta|_{\mathbf{x}=\mathbf{x}_j^u} \approx \frac{\eta^{\rightarrow} - \eta^{\leftarrow}}{\Delta x} \tag{A.21}$$

This is the well-known $(-1, 1)$-stencil. The y-derivative of a variable of the T-grid sits on the v-grid:

$$\partial_y \eta|_{\mathbf{x}=\mathbf{x}_k^v} \approx \frac{\eta^{\uparrow} - \eta^{\downarrow}}{\Delta y} \tag{A.22}$$

The x-derivative of a variable on the u-grid includes the kinematic boundary conditions, hence a 0 appears for computations involving the boundary nodes. The result sits on the T-grid:

$$\partial_x u|_{\mathbf{x}=\mathbf{x}_i} \approx \frac{1}{\Delta x} \begin{cases} u^{\rightarrow} - 0 & \text{if western boundary} \\ 0 - u^{\leftarrow} & \text{if eastern boundary} \\ u^{\rightarrow} - u^{\leftarrow} & \text{else.} \end{cases} \tag{A.23}$$

Similarly for the y-derivative on the v-grid which sits again on the T-grid:

$$\partial_y v|_{\mathbf{x}=\mathbf{x}_i} \approx \frac{1}{\Delta y} \begin{cases} v^{\uparrow} - 0 & \text{if southern boundary} \\ 0 - v^{\downarrow} & \text{if northern boundary} \\ v^{\uparrow} - v^{\downarrow} & \text{else.} \end{cases} \tag{A.24}$$

The discretization of $\partial_x q$ and $\partial_y q$ are in close relation to equation A.21 and A.22 and do not involve any boundary conditions

$$\partial_x q|_{\mathbf{x}=\mathbf{x}_k^v} \approx \frac{q^{\rightarrow} - q^{\leftarrow}}{\Delta x}, \quad \partial_y q|_{\mathbf{x}=\mathbf{x}_j^u} \approx \frac{q^{\uparrow} - q^{\downarrow}}{\Delta y} \tag{A.25}$$

Note, that some grid nodes of q on the boundary are simply not evaluated in this computation.

Implementing the tangential boundary conditions

For $\partial_y u, \partial_x v$ the tangential boundary conditions as no-slip (equation A.3) or free-slip (equation A.4) come into play. For simplicity we first look at $\partial_y u|_{\mathbf{x}=\mathbf{x}_l^q}$ for $l = 2$, i.e. at $\mathbf{x} = (\Delta x, 0)$, where the derivative is

$$\partial_y u|_{\mathbf{x}=(\Delta x,0)} \approx \frac{u_1 - u_b}{\Delta y} \tag{A.26}$$

with $u_b = u(\Delta x, -\frac{1}{2}\Delta y)$ the velocity just outside the domain if the grid where extend in negative y direction. To match the no-slip boundary condition (equation A.3) we set $u_b = -u_1$, so that at $\mathbf{x} = (\Delta x, 0)$, i.e. right on the boudary we have $u = 0$, when applying linear interpolation. This yields

$$\partial_y u|_{\mathbf{x}=(\Delta x,0)} \approx \frac{2u_1}{\Delta y} \tag{A.27}$$

In contrast, when choosing free-slip boundary conditions (equation A.4) we set $u_b = u_1$ for similar reasoning. Then

$$\partial_y u|_{\mathbf{x}=(\Delta x,0)} \approx \frac{0u_1}{\Delta y} = 0 \tag{A.28}$$

as desired. With introducing the parameter $\alpha \in \{0, 2\}$ Madec [2016] switch between no-slip ($\alpha = 2$) and free-slip boundary conditions ($\alpha = 0$). A choice of $0 < \alpha < 2$ corresponds to partial-slip. Following these ideas we write the y-derivative on the u-grid as

$$\partial_y u|_{\mathbf{x}=\mathbf{x}_l^q} \approx \frac{1}{\Delta y} \begin{cases} 0 & \text{if western or eastern boundary} \\ \alpha u^{\uparrow} & \text{if southern boundary without SW,SE} \\ -\alpha u^{\downarrow} & \text{if northern boundary without NW, NE} \\ u^{\uparrow} - u^{\downarrow} & \text{else.} \end{cases} \tag{A.29}$$

which sits then on the q-grid. Similarly we have the discretization of $\partial_x v$ as

$$\partial_x v|_{\mathbf{x}=\mathbf{x}_l^q} \approx \frac{1}{\Delta x} \begin{cases} 0 & \text{if northern or western boundary} \\ \alpha v^{\rightarrow} & \text{if western boundary without NW,SW} \\ -\alpha v^{\leftarrow} & \text{if eastern boundary without NE, SE} \\ v^{\rightarrow} - v^{\leftarrow} & \text{else.} \end{cases} \tag{A.30}$$

Higher order derivative at the boundary

Shchepetkin & O'Brien [1996] propose for no-slip boundary conditions ($\alpha = 2$) to use instead of equation (A.29), (A.30) the larger stencil $(4, -1, \frac{1}{5})$ at the boundary

$$\partial_y u|_{\mathbf{x}=\mathbf{x}_l^q} \approx \frac{1}{\Delta y} \begin{cases} 0 & \text{if western or eastern boundary} \\ 4u^{\uparrow} - u^{\Uparrow} + \frac{1}{5}u^{\Uparrow\uparrow} & \text{if southern boundary without SW,SE} \\ -4u^{\downarrow} + u^{\Downarrow} - \frac{1}{5}u^{\Downarrow\downarrow} & \text{if northern boundary without NW, NE} \\ u^{\uparrow} - u^{\downarrow} & \text{else.} \end{cases} \tag{A.31}$$

$$\partial_x v|_{\mathbf{x}=\mathbf{x}_l^q} \approx \frac{1}{\Delta x} \begin{cases} 0 & \text{if northern or western boundary} \\ 4v^{\rightarrow} - v^{\Rightarrow} + \frac{1}{5} v^{\substack{\rightarrow \\ \Rightarrow}} & \text{if western boundary without NW,SW} \\ -4v^{\leftarrow} + v^{\Leftarrow} - \frac{1}{5} v^{\substack{\leftarrow \\ \Leftarrow}} & \text{if eastern boundary without NE, SE} \\ v^{\rightarrow} - v^{\leftarrow} & \text{else.} \end{cases} \tag{A.32}$$

These operators will be called ${}^2\mathbf{G}_y^u$ and ${}^2\mathbf{G}_x^v$ (see next section). Note that in the interior of the domain we have ${}^2\mathbf{G}_y^u = \mathbf{G}_y^u$ and ${}^2\mathbf{G}_x^v = \mathbf{G}_x^v$. These operators only appear in the lateral mixing of momentum terms (section A.2.5). For the case of $\alpha \neq 2$ (i.e. free-slip, partial-slip or hyper-slip) we might stick to the notation with 2 as prescript but mean the operator defined by equation (A.29), (A.30) for simplicity.

A.2.3 Interpolation

2-point spatial interpolation

With variables that sit on four different grids it is sometimes necessary to transform one variable from one grid onto another. Regarding the term $\partial_x(uh)$ from equation 2.1 we either need to find a representation of u on the T-grid or of h on the u-grid in order to multiply them (the latter is actually preferred, see eq. A.70). This is done via linear interpolation of the closest grid points. In the following we will investigate the interpolations from any of the four grids to any other.

From T-grid to u- or v-grid and vice versa

Let $\mathcal{I}_T^u(\mathtt{h}) = \mathtt{h}_u$ be the linear interpolation of $\mathtt{h}$ from the T-grid (subscript of the interpolation function $\mathcal{I}$) onto the u-grid (superscript of $\mathcal{I}$), then

$$\mathtt{h}_u = h|_{\mathbf{x}=\mathbf{x}_j^u} \approx \frac{h^{\leftarrow} + h^{\rightarrow}}{2} \tag{A.33}$$

which corresponds to spatial averaging of two neighbouring grid points in the x-direction. Similarly to the gradients, we can write this operation via a matrix multiplication with $\mathbf{I}_T^u$ (sub- and superscript meaning as above)

$$\mathcal{I}_T^u : V_T \rightarrow V_u, \mathtt{h} \rightarrow \mathbf{I}_T^u \mathtt{h} \tag{A.34}$$

Due to the similarity in equation (A.33) and (A.21), $\mathbf{I}^u_T$ is the same as $\mathbf{G}^T_x$ but all non-zero entries replaced by $\frac{1}{2}$. Same holds for the interpolation of $\mathtt{h}$ onto the v-grid, i.e. a spatial averaging in y-direction of two neighbouring grid points

$$\mathtt{h}_v = h|_{\mathbf{x}=\mathbf{x}^v_k} \approx \frac{h^\uparrow + h^\downarrow}{2} \tag{A.35}$$

which can again be written as

$$\mathcal{I}^v_T : V_T \to V_v, \mathtt{h} \to \mathbf{I}^v_T\mathtt{h}. \tag{A.36}$$

with $\mathbf{I}^v_T$ obtained from $\mathbf{G}^T_y$ by setting all non-zero entries to $\frac{1}{2}$. Same relations hold for $\mathbf{I}^v_q$ and $\mathbf{G}^q_x$ (2-point interpolation in x-direction), $\mathbf{I}^u_q$ and $\mathbf{G}^q_y$ (2-point interpolation in y-direction). And also, including the kinematic boundary condition (equation A.1), for $\mathbf{I}^T_u$ and $\mathbf{G}^u_x$ as well as $\mathbf{I}^T_v$ and $\mathbf{G}^v_y$. Interestingly,

$$\mathbf{I}^T_u = \mathbf{I}^u_T{}' \quad , \quad \mathbf{I}^T_v = \mathbf{I}^v_T{}' \tag{A.37}$$

where $'$ denotes the matrix transpose.

From u-grid and v-grid to q-grid

For the interpolation matrices $\mathbf{I}^q_u, \mathbf{I}^q_v$ the lateral boundary conditions are important. In fact, following the ideas around equation (A.29) we obtain the 2-point interpolation from the u-grid onto the q-grid as

$$\mathtt{u}_q = u|_{\mathbf{x}=\mathbf{x}^q_l} \approx \begin{cases} 0 & \text{if western or eastern boundary} \\ (1-\frac{\alpha}{2})u^\uparrow & \text{if southern boundary without SW,SE} \\ (1-\frac{\alpha}{2})u^\downarrow & \text{if northern boundary without NW,NE} \\ \frac{1}{2}(u^\uparrow + u^\downarrow) & \text{else.} \end{cases} \tag{A.38}$$

with α being the tangential boundary condition parameter ($\alpha = 0$ is free-slip, $\alpha = 2$ is no-slip). Again, $\mathbf{I}^q_v$ is then straight forward

$$\mathtt{v}_q = v|_{\mathbf{x}=\mathbf{x}^q_l} \approx \begin{cases} 0 & \text{if northern or western boundary} \\ (1-\frac{\alpha}{2})v^{\rightarrow} & \text{if western boundary without NW,SW} \\ (1-\frac{\alpha}{2})v^{\leftarrow} & \text{if eastern boundary without NE, SE} \\ \frac{1}{2}(v^{\rightarrow}+v^{\leftarrow}) & \text{else.} \end{cases} \tag{A.39}$$

4-point spatial interpolation

From u-grid to v-grid and vice versa

The previous interpolations involve 2-point spatial averaging, however, the interpolations $\mathcal{I}^v_u, \mathcal{I}^u_v, \mathcal{I}^T_q, \mathcal{I}^q_T$ require averaging from the four surrounding grid points and will be described in the following.

The interpolation $\mathcal{I}^v_u$ from the u-grid onto the v-grid is

$$\mathtt{u}_v = u|_{\mathbf{x}=\mathbf{x}^v_k} \approx \frac{1}{4}\begin{cases} (u^{\searrow}+u^{\nearrow}) & \text{if western boundary} \\ (u^{\swarrow}+u^{\nwarrow}) & \text{if eastern boundary} \\ (u^{\searrow}+u^{\nearrow}+u^{\swarrow}+u^{\nwarrow}) & \text{else.} \end{cases} \tag{A.40}$$

and similarly the interpolation $\mathcal{I}^u_v$ reads

$$\mathtt{v}_u = v|_{\mathbf{x}=\mathbf{x}^u_j} \approx \frac{1}{4}\begin{cases} (v^{\searrow}+v^{\swarrow}) & \text{if northern boundary} \\ (v^{\nearrow}+v^{\nwarrow}) & \text{if southern boundary} \\ (v^{\searrow}+v^{\nearrow}+v^{\swarrow}+v^{\nwarrow}) & \text{else.} \end{cases} \tag{A.41}$$

Note, that both $\mathcal{I}^v_u$ and also $\mathcal{I}^u_v$ include the kinematic boundary condition. Once we write this interpolation as a matrix $\mathbf{I}^v_u$, following the same arguments, we can deduce that

$$\mathbf{I}^u_v = \mathbf{I}^{v\prime}_u \tag{A.42}$$

The interpolation from the u-grid to the v-grid is the transpose of the interpolation from v to u.

From q-grid to T-grid and vice versa

The interpolation $\mathcal{I}_q^T$ from the q-grid to the T-grid is

$$\mathtt{q}_T = q|_{\mathbf{x}=\mathbf{x}_i} \approx \frac{1}{4}(q^{\searrow} + q^{\nearrow} + q^{\swarrow} + q^{\nwarrow}) \tag{A.43}$$

And finally the interpolation $\mathcal{I}_T^q$ makes use of the additional boundary condition in equation A.2 for numerical purposes.

$$\mathtt{h}_q = h|_{\mathbf{x}=\mathbf{x}_l^q} \approx \begin{cases} h^{\swarrow} & \text{if north-east corner (NE)} \\ h^{\searrow} & \text{if north-west corner (NW)} \\ h^{\nwarrow} & \text{if south-east corner (SE)} \\ h^{\nearrow} & \text{if south-west corner (SW)} \\ \frac{1}{2}(h^{\searrow} + h^{\swarrow}) & \text{if northern boundary without NW,NE} \\ \frac{1}{2}(h^{\swarrow} + h^{\nwarrow}) & \text{if eastern boundary without NE,SE} \\ \frac{1}{2}(h^{\nearrow} + h^{\searrow}) & \text{if western boundary without NW, SW} \\ \frac{1}{2}(h^{\nearrow} + h^{\nwarrow}) & \text{if southern boundary without SW, SE} \\ \frac{1}{4}(h^{\searrow} + h^{\nearrow} + h^{\swarrow} + h^{\nwarrow}) & \text{else.} \end{cases} \tag{A.44}$$

A.2.4 Advection term

In the following two different schemes are discussed that aim at discretizing the advection terms

$$(qhv, -qhu), \quad \text{with } q = \frac{f + \partial_x v - \partial_y u}{h} \tag{A.45}$$

with the potential vorticity q, that appear in equation (A.7). The Arakawa and Lamb scheme [Arakawa & Lamb, 1981] used in all model simulations of this study and its implementation is presented in the following.

Arakawa and Lamb (1981) energy and enstrophy conserving scheme

The energy and enstrophy conserving scheme developed by Arakawa & Lamb [1981], called AL hereafter, has a wider stencil compared to the scheme from Sadourny [1975] (hereafter SZ), i.e. computationally more costly, but was also found to perform better [Salmon, 2007]. That means AL transports less enstrophy (which is essentially squared vorticity) to higher wavenumbers, which reduces the numerical noise on the grid scale compared to SZ. For further details see also Salmon [2004] where the scheme is provided in a much more readable notation as in the original AL paper.

As in SZ compute the mass fluxes $U = uh$ and $V = vh$ as

$$\mathtt{U} = uh|_{\mathbf{x}=\mathbf{x}^u_j} \approx \mathtt{u} * \mathbf{I}^u_T \mathtt{h} \tag{A.46a}$$

$$\mathtt{V} = vh|_{\mathbf{x}=\mathbf{x}^v_k} \approx \mathtt{v} * \mathbf{I}^v_T \mathtt{h} \tag{A.46b}$$

The advective term in the u-component qhv is then discretized as a summation of linear combinations of the surrounding potential vorticity points q and the mass fluxes U, V. We start with computing the linear combinations of q. Let $\mathbf{A}_{L1}, \mathbf{A}_{L2}, \mathbf{A}_{L3}, \mathbf{A}_{L4}$ (without meaning of the subscripts) be four different interpolations[1] (directly written as matrix) fom the q- to the T-grid defined as

$$\mathbf{A}_{L1}\mathtt{q} = \tfrac{1}{24}(2q^{\nwarrow} + q^{\nearrow} + q^{\swarrow} + 2q^{\searrow}), \quad \mathbf{A}_{L2}\mathtt{q} = \tfrac{1}{24}(q^{\nwarrow} + 2q^{\nearrow} + 2q^{\swarrow} + q^{\searrow}) \tag{A.47a}$$

$$\mathbf{A}_{L3}\mathtt{q} = \tfrac{1}{24}(q^{\nwarrow} + q^{\nearrow} - q^{\swarrow} - q^{\searrow}), \quad \mathbf{A}_{L4}\mathtt{q} = \tfrac{1}{24}(q^{\nwarrow} - q^{\nearrow} + q^{\swarrow} - q^{\searrow}) \tag{A.47b}$$

for visualization the corresponding stencils (denoted with subscript $1, 2, 3, 4$) are

$$\frac{1}{24}\begin{vmatrix} 2 & 1 \\ 1 & 2 \end{vmatrix}_1, \quad \frac{1}{24}\begin{vmatrix} 1 & 2 \\ 2 & 1 \end{vmatrix}_2, \quad \frac{1}{24}\begin{vmatrix} 1 & 1 \\ -1 & -1 \end{vmatrix}_3, \quad \frac{1}{24}\begin{vmatrix} 1 & -1 \\ 1 & -1 \end{vmatrix}_4 \tag{A.48}$$

We might use $\mathbf{A}_L$ and mean then any of matrices in equation (A.47). We define interpolation matrices $\mathbf{R}$ to get the variables $\mathbf{A}_L\mathtt{q}$ from the T- to the u- or v-grid. As the matrices $\mathbf{R}$ contain a maximum of one entry per row, they are rather shift matrices or correspond to nearest-point interpolation. Hence, they could also be written in terms of an index (as actually done in the model code). We first look at $\mathbf{R}^{\uparrow}_v$ which picks for all v-grid points the

[1] $\mathbf{A}_{L3}$ and $\mathbf{A}_{L4}$ should be rather regarded as potential vorticity gradient due to the minus sign in their stencil.

corresponding $\mathbf{A}_L\mathsf{q}$-value that sits $\frac{1}{2\Delta y}$ North of this point on the T-grid.

$$\mathbf{R}_v^{\uparrow}\mathbf{A}_L\mathsf{q} = (\mathbf{A}_L\mathsf{q})^{\uparrow} \tag{A.49}$$

Hence, the interpolation of this operator shifts the T-grid southward by $\frac{1}{2\Delta y}$ to place them on the v-grid. The grid cell row closest to $y = 0$ is therefore left-out. Similar for $\mathbf{R}_v^{\downarrow}$, $\mathbf{R}_u^{\leftarrow}$ and $\mathbf{R}_u^{\rightarrow}$, which are

$$\mathbf{R}_v^{\downarrow}\mathbf{A}_L\mathsf{q} = (\mathbf{A}_L\mathsf{q})^{\downarrow}, \quad \mathbf{R}_u^{\leftarrow}\mathbf{A}_L\mathsf{q} = (\mathbf{A}_L\mathsf{q})^{\leftarrow}, \quad \mathbf{R}_u^{\rightarrow}\mathbf{A}_L\mathsf{q} = (\mathbf{A}_L\mathsf{q})^{\rightarrow}. \tag{A.50}$$

Once the $\mathbf{A}_L$-interpolated potential vorticity sits on the u- and v-grid they are multiplied with the mass fluxes U, V. In order to get a discretized advection term, AL interpolate the surrounding absolute vorticity fluxes qU, qV onto each u- and v-grid point. For this final interpolation we further need another set of shift-matrices $\mathbf{T}$ (they could again be written in terms of an index) that shift a variable v from the v to the u-grid as follows

$$\mathbf{T}_{v\to u}^{\nearrow}\mathsf{v} = \begin{cases} 0 & \text{if northern boundary} \\ v^{\nearrow} & \text{else.} \end{cases} \tag{A.51}$$

and include the kinematic boundary condition (case 1). Similarly, we have

$$\mathbf{T}_{v\to u}^{\searrow}\mathsf{v} = \begin{cases} 0 & \text{if southern boundary} \\ v^{\searrow} & \text{else.} \end{cases} \tag{A.52}$$

$$\mathbf{T}_{v\to u}^{\swarrow}\mathsf{v} = \begin{cases} 0 & \text{if southern boundary} \\ v^{\swarrow} & \text{else.} \end{cases} \tag{A.53}$$

$$\mathbf{T}_{v\to u}^{\nwarrow}\mathsf{v} = \begin{cases} 0 & \text{if northern boundary} \\ v^{\nwarrow} & \text{else.} \end{cases} \tag{A.54}$$

and also $\mathbf{T}_{u\to v}^{\nwarrow}, \mathbf{T}_{u\to v}^{\nearrow}, \mathbf{T}_{u\to v}^{\searrow}, \mathbf{T}_{u\to v}^{\swarrow}$ as well as $\mathbf{T}_u^{\Rightarrow}, \mathbf{T}_u^{\Leftarrow}, \mathbf{T}_v^{\Uparrow}, \mathbf{T}_v^{\Downarrow}$. We are now able to write the u-component of the advection term as

$$\begin{aligned} qhv|_{\mathbf{x}=\mathbf{x}^u_j} \approx\ & \mathbf{T}^{\nearrow}_{v\to u}\left(\mathbf{R}^{\downarrow}_{v}\mathbf{A}_{L1}\mathtt{q} * \mathtt{V}\right) + \mathbf{T}^{\searrow}_{v\to u}\left(\mathbf{R}^{\uparrow}_{v}\mathbf{A}_{L2}\mathtt{q} * \mathtt{V}\right) \\ & \mathbf{T}^{\nwarrow}_{v\to u}\left(\mathbf{R}^{\downarrow}_{v}\mathbf{A}_{L2}\mathtt{q} * \mathtt{V}\right) + \mathbf{T}^{\swarrow}_{v\to u}\left(\mathbf{R}^{\uparrow}_{v}\mathbf{A}_{L1}\mathtt{q} * \mathtt{V}\right) \\ & \mathbf{T}^{\Leftarrow}_{u}\,(\mathbf{R}^{\rightarrow}_{u}\mathbf{A}_{L3}\mathtt{q} * \mathtt{U}) - \mathbf{T}^{\Rightarrow}_{u}\,(\mathbf{R}^{\leftarrow}_{u}\mathbf{A}_{L3}\mathtt{q} * \mathtt{U}) \equiv \mathtt{A}_u \end{aligned} \tag{A.55}$$

and the v-component as

$$\begin{aligned} -qhu|_{\mathbf{x}=\mathbf{x}^v_k} \approx\ & - \mathbf{T}^{\nearrow}_{u\to v}\,(\mathbf{R}^{\rightarrow}_{v}\mathbf{A}_{L1}\mathtt{q} * \mathtt{U}) - \mathbf{T}^{\searrow}_{u\to v}\,(\mathbf{R}^{\rightarrow}_{v}\mathbf{A}_{L2}\mathtt{q} * \mathtt{U}) \\ & - \mathbf{T}^{\nwarrow}_{u\to v}\,(\mathbf{R}^{\leftarrow}_{v}\mathbf{A}_{L2}\mathtt{q} * \mathtt{U}) - \mathbf{T}^{\swarrow}_{u\to v}\,(\mathbf{R}^{\leftarrow}_{v}\mathbf{A}_{L1}\mathtt{q} * \mathtt{U}) \\ & - \mathbf{T}^{\Uparrow}_{v}\left(\mathbf{R}^{\downarrow}_{v}\mathbf{A}_{L4}\mathtt{q} * \mathtt{V}\right) + \mathbf{T}^{\Downarrow}_{v}\left(\mathbf{R}^{\uparrow}_{v}\mathbf{A}_{L4}\mathtt{q} * \mathtt{V}\right) \equiv \mathtt{A}_v \end{aligned} \tag{A.56}$$

These equations correspond to equation B.2 from Salmon [2004] and 3.5 and 3.6 together with 3.34 from Arakawa & Lamb [1981]. However, from this notation it is clear that there are essentially four costly computations that can be precomputed: $\mathbf{A}_{L1}\mathtt{q}, \mathbf{A}_{L2}\mathtt{q}, \mathbf{A}_{L3}\mathtt{q}$ and $\mathbf{A}_{L4}\mathtt{q}$. The remaining shift matrices $\mathbf{R}, \mathbf{T}$ have at maximum one entry per row and can also be implemented as index.

A.2.5 Discrete friction

Discrete bottom friction

We discretize equation (2.6) with the interpolation operators from the previous sections as

$$-\frac{c_D}{h}|\mathbf{u}|u \approx -\frac{c_D}{\mathtt{h}}\mathbf{I}^u_T\left(\sqrt{\mathbf{I}^T_u\mathtt{u}^2 + \mathbf{I}^T_v\mathtt{v}^2}\right) * \mathtt{u} \equiv \mathtt{B}_x \tag{A.57a}$$

$$-\frac{c_D}{h}|\mathbf{u}|v \approx -\frac{c_D}{\mathtt{h}}\mathbf{I}^v_T\left(\sqrt{\mathbf{I}^T_u\mathtt{u}^2 + \mathbf{I}^T_v\mathtt{v}^2}\right) * \mathtt{v} \equiv \mathtt{B}_y \tag{A.57b}$$

In fact, the brackets only have to be computed once, and the term in the square-root also appears in the discrete form of equation (A.5).

Discrete lateral mixing of momentum

The discretization of equation (2.11) is done in the following way:

The stress tensor $\mathbf{S}$ is discretized as

$$\mathbf{S} \approx \begin{pmatrix} \mathbf{G}_x^u\mathtt{u} - \mathbf{G}_y^v\mathtt{v} & {}^2\mathbf{G}_x^v\mathtt{v} + {}^2\mathbf{G}_y^u\mathtt{u} \\ {}^2\mathbf{G}_x^v\mathtt{v} + {}^2\mathbf{G}_y^u\mathtt{u} & -(\mathbf{G}_x^u\mathtt{u} - \mathbf{G}_y^v\mathtt{v}) \end{pmatrix} \equiv \begin{pmatrix} \mathtt{S}_{11} & \mathtt{S}_{12} \\ \mathtt{S}_{12} & -\mathtt{S}_{11} \end{pmatrix} \tag{A.58}$$

Note, as $\mathbf{S}$ is symmetric and has a vanishing trace, only two entries need to be computed explicitly. Then, with $\mathtt{h}_q = \mathbf{I}_T^q\mathtt{h}$

$$\frac{1}{h}\nabla \cdot h\mathbf{S} \approx \begin{pmatrix} \frac{1}{\mathtt{h}_u} * (\mathbf{G}_x^T(\mathtt{h} * \mathtt{S}_{11}) + \mathbf{G}_y^q(\mathtt{h}_q * \mathtt{S}_{12})) \\ \frac{1}{\mathtt{h}_v} * (\mathbf{G}_x^q(\mathtt{h}_q * \mathtt{S}_{12}) - \mathbf{G}_y^T(\mathtt{h} * \mathtt{S}_{11})) \end{pmatrix} \equiv \begin{pmatrix} \mathtt{d}_u \\ \mathtt{d}_v \end{pmatrix} \tag{A.59}$$

which is the harmonic viscosity term without coefficient (which is still assumed to be constant). To obtain a biharmonic viscosity term, we formulate another tensor $\mathtt{R} = (\mathtt{R}_{11}, \mathtt{R}_{12}; \mathtt{R}_{12}, \mathtt{R}_{22})$ as

$$\mathtt{R}_{11} = \mathbf{G}_x^u\mathtt{d}_u - \mathbf{G}_y^v\mathtt{d}_v \tag{A.60a}$$

$$\mathtt{R}_{12} = {}^2\mathbf{G}_x^v\mathtt{d}_v + {}^2\mathbf{G}_y^u\mathtt{d}_u \tag{A.60b}$$

Which is, in principal, evaluating equation (A.58) with $(\mathtt{d}_u, \mathtt{d}_v)$ instead of $(\mathtt{u}, \mathtt{v})$. The divergence of this tensor yields the complete biharmonic lateral mixing of momentum terms

$$\nu_B h^{-1}\nabla \cdot (h\mathbf{S}(h^{-1}\nabla \cdot h\mathbf{S}(u,v))) \approx \nu_B \begin{pmatrix} \frac{1}{\mathtt{h}_u} * (\mathbf{G}_x^T(\mathtt{h} * \mathtt{R}_{11}) + \mathbf{G}_y^q(\mathtt{h}_q * \mathtt{R}_{12})) \\ \frac{1}{\mathtt{h}_v} * (\mathbf{G}_x^q(\mathtt{h}_q * \mathtt{R}_{12}) - \mathbf{G}_y^T(\mathtt{h} * \mathtt{R}_{11})) \end{pmatrix} \equiv \begin{pmatrix} \mathtt{M}_x \\ \mathtt{M}_y \end{pmatrix} \tag{A.61}$$

for a constant biharmonic viscosity coefficient ν_B.

A.2.6 Choosing the viscosity and friction coefficients

The bottom fricton coefficient c_D (equation 2.6) and the biharmonic viscosity ν_B (equation 2.11) that are used in the shallow water model of this study, cannot be chosen from physical principles but their choices should involve considerations of numerical stability [Griffies & Hallberg, 2000]. Arguments for the choice of c_D, ν_B are presented in the following.

Harmonic and biharmonic viscosity

For the given configuration as described in section 2.1.1 we find the choice for the harmonic viscosity

$$\nu_{A,0} = 540\,\mathrm{m}^2\,\mathrm{s}^{-1} \tag{A.62}$$

for a resolution with $\Delta x_0 = \Delta y_0 = 30$ km appropriate once a diffusion operator of the form in equation 2.10 is used. That means it is chosen as small as possible but still removing clearly numerical oscillations that occur at the grid scale. This choice also resolves the Munk boundary layer width [Gill, 1982]

$$W_M = \sqrt[3]{\frac{\nu_{A,0}}{\beta}} \approx \Delta x_0 \tag{A.63}$$

with approximately one grid cell. It was proposed to use this as an argument to set $\nu_A = \beta \Delta x^3$ [Cooper & Zanna, 2015]. Although this might a criterion for stability, for $\Delta x < 30$ km it was not found to prevent numerical oscillations at the grid scale from occuring. Instead, the following scaling argument is proposed: At the grid scale Δx the advective terms are desired to balance with viscosity

$$\mathcal{O}((\mathbf{u} \cdot \nabla)\mathbf{u}) = \frac{U^2}{\Delta x} \sim \nu_A \frac{U}{\Delta x^2} = \mathcal{O}(\nu_A \nabla^2 \mathbf{u}) \tag{A.64}$$

with a velocity scale U. It follows a linear scaling of ν_A with Δx

$$\nu_A = U \Delta x \tag{A.65}$$

under the assumption that the velocity scale does not change considerably with Δx. Based on the empirically found value $\nu_{A,0}$ from equation A.62, it is therefore proposed to use dependent on the resolution

$$\nu_A = \frac{\nu_{A,0}}{\Delta x_0} \Delta x = \frac{540\,\mathrm{m}^2\,\mathrm{s}^{-1}}{30\,\mathrm{km}} \Delta x \tag{A.66}$$

The biharmonic eddy viscosity scaling is derived from the requirement that harmonic and biharmonic viscosity should be on the same order of magnitude

$$1 = \frac{\mathcal{O}(\nu_A \nabla^2 \mathbf{u})}{\mathcal{O}(\nu_B \nabla^4 \mathbf{u})} = \frac{\nu_A}{\nu_B} \Delta x^2 \tag{A.67}$$

Hence, we propose a scaling for ν_B as

$$\nu_B = \frac{\nu_{A,0}}{\Delta x_0}\Delta x^3 = \frac{540\,\mathrm{m}^2\,\mathrm{s}^{-1}}{30\,\mathrm{km}}\Delta x^3. \tag{A.68}$$

Based on this equation, we set the biharmonic viscosity coefficients for all model runs as listed in Table 2.1.

Bottom friction coefficient

Arbic & Scott [2008] propose for general purpose a choice of $c_D = 0.0025$ in equation 2.6 based on a comparison of model simulations with observational data. However, using this value in combination with the physical parameters of section 2.1.1 the model reaches a steady state (all $\partial_t \to 0$) within a month or so. Although this steady state resembles a double gyre, no eddies are permitted to develop. Therefore a smaller c_D is needed to reduce the friction in the model. Some tuning experiments lead to the choice of

$$c_D = 10^{-5} \tag{A.69}$$

which removes energy especially on larger scales but retains vorticity dynamics as discussed in the results of chapter 3. The factor 250 discrepancy between our choice and the one from Arbic & Scott [2008] might be further justified as for a one-layer shallow water model the bottom friction is computed via the vertically averaged velocity, not the bottom velocity which would be smaller.

A.2.7 Summary on spatial discretization

The spatially discretized equations of the reformulated shallow water model (equation A.7) are

$$\partial_t u = \mathtt{A}_u - \mathbf{G}_x^T\mathtt{p} + \mathtt{F}_x + \mathtt{B}_x - \mathtt{M}_x \tag{A.70a}$$

$$\partial_t v = \mathtt{A}_v - \mathbf{G}_y^T\mathtt{p} + \mathtt{B}_y - \mathtt{M}_y \tag{A.70b}$$

$$\partial_t \eta = -\mathbf{G}_x^u(\mathtt{u} * \mathbf{I}_T^u\mathtt{h}) - \mathbf{G}_y^v(\mathtt{v} * \mathbf{I}_T^v\mathtt{h}) \tag{A.70c}$$

with

$$\mathtt{p} = \frac{1}{2}\left(\mathbf{I}_u^T(\mathtt{u}^2) + \mathbf{I}_v^T(\mathtt{v}^2)\right) + g\mathtt{h} \tag{A.71}$$

and $\mathtt{A}_u, \mathtt{A}_v$ from equations (A.55,A.56), and $\mathtt{B}_x, \mathtt{B}_y$ from equation (A.57) as well as $\mathtt{M}_x, \mathtt{M}_y$ from equation (A.61).

A.2.8 Time discretization: Runge-Kutta 4th order

The discrete shallow water model is integrated forward in time with the 4th order Runge-Kutta scheme (RK4, Butcher [2008]). Summarizing the right-hand side of equations A.70 with $\mathrm{rhs}(\mathtt{u},\mathtt{v},\mathtt{h}) = (\mathtt{du},\mathtt{dv},\mathtt{dh})$ the model equations reduce to

$$\partial_t \begin{pmatrix} \mathtt{u} \\ \mathtt{v} \\ \mathtt{h} \end{pmatrix} = \begin{pmatrix} \mathtt{du} \\ \mathtt{dv} \\ \mathtt{dh} \end{pmatrix}. \tag{A.72}$$

Using RK4, discretizing the temporal derivative reads

$$\begin{pmatrix} \mathtt{u}^{n+1} \\ \mathtt{v}^{n+1} \\ \mathtt{h}^{n+1} \end{pmatrix} = \begin{pmatrix} \mathtt{u}^{n} \\ \mathtt{v}^{n} \\ \mathtt{h}^{n} \end{pmatrix} + \frac{\Delta t}{6} \begin{pmatrix} \mathtt{k}_1^u + 2\mathtt{k}_2^u + 2\mathtt{k}_3^u + \mathtt{k}_4^u \\ \mathtt{k}_1^v + 2\mathtt{k}_2^v + 2\mathtt{k}_3^v + \mathtt{k}_4^v \\ \mathtt{k}_1^h + 2\mathtt{k}_2^h + 2\mathtt{k}_3^h + \mathtt{k}_4^h \end{pmatrix} \tag{A.73}$$

with the superscript $n, n+1$ denoting the current and next time step, respectively, that lie time Δt apart. The choice of Δt is discussed in section section A.2.9. $(\mathtt{k}^u, \mathtt{k}^v, \mathtt{k}^h)$ are approximations for $(\partial_t \mathtt{u}, \partial_t \mathtt{v}, \partial_t \mathtt{k})$ and defined as

$$(\mathtt{k}_1^u, \mathtt{k}_1^v, \mathtt{k}_1^h) = \mathrm{rhs}(\mathtt{u}^n, \mathtt{v}^n, \mathtt{h}^n) \tag{A.74a}$$

$$(\mathtt{k}_2^u, \mathtt{k}_2^v, \mathtt{k}_2^h) = \mathrm{rhs}(\mathtt{u}^n + \tfrac{\Delta t}{2}\mathtt{k}_1^u, \mathtt{v}^n + \tfrac{\Delta t}{2}\mathtt{k}_1^v, \mathtt{h}^n + \tfrac{\Delta t}{2}\mathtt{k}_1^h) \tag{A.74b}$$

$$(\mathtt{k}_3^u, \mathtt{k}_3^v, \mathtt{k}_3^h) = \mathrm{rhs}(\mathtt{u}^n + \tfrac{\Delta t}{2}\mathtt{k}_2^u, \mathtt{v}^n + \tfrac{\Delta t}{2}\mathtt{k}_2^v, \mathtt{h}^n + \tfrac{\Delta t}{2}\mathtt{k}_2^h) \tag{A.74c}$$

$$(\mathtt{k}_4^u, \mathtt{k}_4^v, \mathtt{k}_4^h) = \mathrm{rhs}(\mathtt{u}^n + \Delta t\mathtt{k}_3^u, \mathtt{v}^n + \Delta t\mathtt{k}_3^v, \mathtt{h}^n + \Delta t\mathtt{k}_3^h) \tag{A.74d}$$

A.2.9 Choosing the time step Δt

In the shallow water model, the fastest propagating signals are gravity waves. The phase speed c_p of those waves is [Gill, 1982; Vallis, 2006]

$$c_p = \sqrt{gh} \approx \sqrt{gH} \tag{A.75}$$

where the approximation holds in the barotropic case where $|\eta| \ll H$. In contrast, using a reduced gravity $g' \ll g = 10$ m/s^2 usually yields much larger variations in η and the approximation in equation (A.75) may become less justified, but will in many cases still be useful. As a consequence, waves might propagate significantly faster in certain regions, which can affect the numerical stability. The CFL-number ϵ (named after Courant, Friedrichs and Levy; Courant *et al.* [1967]) is then $\epsilon = \frac{c_p \Delta t}{\Delta x}$. Hence, for a desired CFL-number, we obtain the time step Δt as

$$\Delta t = \epsilon \frac{\Delta x}{c_p} \tag{A.76}$$

Using RK4, a choice of $\epsilon \leq 0.9$ was found to be stable in the barotropic set-up of equation 2.2. Multi-step schemes such as Adams-Bashforth [Butcher, 2008] have the advantage, that they only require one evaluation of the right-hand side per time step (in contrast to RK4 which requires 4 evaluations of the right-hand side), which could theoretically decrease the computational time required in order to integrate the model forward. However, in practice, the 3rd order Adams-Bashforth method was found to be stable for $\epsilon \leq 0.2$ (i.e. a decrease of Δt by a factor of 4 to 5), which means that the effective computational performance is on the same order but slightly better with RK4. Therefore all simulations in this study use RK4 with $\epsilon = 0.9$.

A.3 Derivations

A.3.1 Energetics in the shallow water model

The derivation of the energy equation follows the one in Gill [1982] and Vallis [2006] but is here extended to a fully non-linear system. The shallow water equations without forcing or dissipation of momentum and h as the prognostic variable

$$\partial_t u + u\partial_x u + v\partial_y u - fv = -g\partial_x h \tag{A.77a}$$

$$\partial_t v + u\partial_x v + v\partial_y v + fu = -g\partial_y h \tag{A.77b}$$

$$\partial_t h + \partial_x(uh) + \partial_y(vh) = 0 \tag{A.77c}$$