A 2D Finite Volume Non-hydrostatic Atmospheric Model - Implementation of Cut Cells and Further Improvements.

Andreas Dobler

advised by
Dr. Jürg Schmidli
Prof. Christoph Schär, IAC, ETH Zürich

Zürich, 02/27/05

Contents

List of figures

List of tables

Nomenclature

α	Damping and filtering coefficients
δQ	Deviation of complete atmospheric state from the hydrostatic background state
ϵ	Order of magnitude of physical vertical acceleration
γ	Adiabatic coefficient
Φ	Gravitation potential
ψ	Arbitrary right hand side function
ρ	Density
θ	Potential temperature
Θ_{ij}^{h}	Local reconstruction of potential temperature
ζ	Local vertical coordinate, aligned with gravity acceleration vector $g \cdot k$
e	Inner energy
f	Analytic horizontal Euler flux function
$\breve{f}$	Analytic Euler flux function normal to cell boundary
$\breve{F}$	Numerical approximation to the analytic flux function $\breve{f}$
g	Gravitational acceleration
h	Analytic vertical Euler flux function
k	Unit vector aligned with gravitation
N	Brunt-Väisälä frequency
p	Pressure in Pascal
p_{00}	Reference pressure
$P_{ij}^{h}(\zeta)$	Local hydrostatic background pressure function
q	Analytic state vector of Euler equations
$Q_{ij}^{h}(\zeta)$	Local hydrostatic background state vector
R	Gas constant for dry air
$R_{ij}^{h}(\zeta)$	Local hydrostatic background density function
s	Source term in Euler equations
S	Approximation to source term
T	Temperature
v	Horizontal velocity
w	Vertical velocity

Abstract

On small scales terrain-following coordinates lead to strongly deformed grids, computational errors and even numerical instability when following a steep and/or abruptly changing surface. An alternative approach is to use height as a vertical coordinate. This then leads to terrain-intersecting coordinate surfaces and a lower boundary consisting of cut cells. The main aim of the present work is to compare these two approaches. To this end we implement a cut-cell lower boundary into an existing finite volume model. Several additional improvements are made to the model. We add a Rayleigh sponge layer at the top of the domain, implement computational and physical mixing and correct the implemented divergence damping. For an accurate calculation of the vertical acceleration the original well-balanced finite volume model requires the evaluation of a hydrostatic background state at every time step. It was found that a less frequent update of the hydrostatic background state leads to identical simulation results, but with the benefit of an increase in performance of more than 90%. Several tests were undertaken with the improved model including the simulation of internal gravity waves triggered by flow past mountains, the nonlinear development of a negative buoyant thermal bubble and an atmosphere at rest over steep topography. The results obtained with the new model agree well with the reference solutions of these problems.

1 Introduction

1.1 Background

Due to the steady increase in available computer power it is nowadays possible to run numerical weather prediction models on finer and finer computational grids and thus resolve the orography much better. We can resolve surface features that are steep and abruptly changing such as narrow valleys or small mountain peaks. By this the virtually in all existing operational forecasting models used terrain-following vertical coordinates are brought close to their limit. The increasing distortion of the computational mesh over steep terrain leads to poor accuracy and failure of numerical convergence, cf. Klemp et al. [1], Zängl et al. [2] and Bonaventura [3]. Especially in the computation of the horizontal pressure gradient force big truncation errors may arise. But metric terms are contained in all horizontal derivatives and may introduce errors even for not so steep slopes. A possible solution to avoid metric terms is the use of a Cartesian terrain-intersecting grid.

Independent of the used vertical coordinate the grid refinement gives us the opportunity to simulate processes we weren't able to resolve on coarser grids. One of these processes in the case of atmospheric models is convection. For convection the vertical acceleration is not negligible compared to buoyancy forces. But in the hydrostatic approximation vertical acceleration is neglected. Thus, in order to simulate convection we cannot use the hydrostatic approximation. We therefore need a non-hydrostatic model. Non-hydrostatic models have been used for years, at least in research (e.g. MC2, ARPS, MM5, COAMPS). For a review of numerical methods for non-hydrostatic weather prediction models see Steppler et al. [4].

Correctly simulating vertical acceleration yields some new problems. The vertical acceleration in atmospheric flows is a small residue of the sum of two very large nearly balanced forces, the gravitational force and the vertical pressure gradient force. Therefore standard finite volume methods used in computational fluid dynamics are either very inaccurate or prohibitively expensive for atmospheric flows as these methods introduce non-balanced local truncation errors (LTEs) in both forces. These LTEs can generate vertical acceleration orders of magnitude larger than the correct physical acceleration. How big the LTEs are depends on the grid spacing and the order of the numerical scheme. In general a numerical scheme of order r on a grid with spacing h introduces LTEs of order h^r. Keeping the LTEs small demands a very fine grid and/or a high order numerical scheme which both lead to time consuming methods.

The standard approach in numerical weather prediction to control the LTEs are the so called well-balanced methods where the momentum balance is formulated in terms of the deviations of pressure from a hydrostatic background state. In [5] Botta et al. suggested a new method to represent the hydrostatic background state. Instead of the global and time independent background state a local and time dependent hydrostatic background state is used. Because the hydrostatic background state is time dependent the deviation of the complete state from the background state can be kept small even for longtime integrations (cf. Wunderlich in [6] , Müller in [7]). Another advantage of the well balanced methods over the standard approaches is the local representation of the background state required for Godunov type schemes (Botta et al. in [5]). More details on the well balanced methods are given by Botta et al. in [5], Wunderlich in [6] and Müller in [7].

1.2 Motivation

In [6] Wunderlich implemented a 2D well balanced finite volume method based on the ideas of Botta et al. in [5]. He used a terrain-following vertical coordinate in his model. In this work we adapt the model to a cut-cell approach and compare it to the terrain-following approach. To improve performance the hydrostatic background state is no longer calculated in every time step. It is rather kept constant for a number of time steps. Nevertheless the deviation of the complete state from the hydrostatic background state remains small as long as the background state is reconstructed reasonably often. Since the original model is very sensitive to the choice of the divergence damping coefficients, the divergence damping term is reviewed. Then, the following improvements are made to the model:

- The implemented divergence damping is corrected (see section 4.2)
- Computational mixing is added in order to remove small scale noise of computational origin (section 4.3)
- Physical mixing is added to simulate diffusion (section 4.4)
- A Rayleigh sponge layer at the top of the domain is introduced in order to absorb upward propagating waves (section 4.5)

To test the improved model several tests are undertaken. The test cases include

- Linear, non-hydrostatic flow (see section 6.3)
- Linear, hydrostatic flow (section 6.4)
- Flow over a Schär hill (section 6.5)
- Development of a negative buoyant thermal bubble (section 6.6)
- Atmosphere at rest over steep topography (section 6.7)

2 Governing equations and the discrete Archimedes' principle

2.1 The inhomogeneous Euler equations

The inhomogeneous Euler equations constitute a hyperbolic system of conservation laws where the conserved quantities are the mass, the momenta in every space dimension and the total energy. In two space dimensions x and z the inhomogeneous Euler equations are given by

$$\frac{\partial}{\partial t}\begin{bmatrix}\rho\\ \rho u\\ \rho w\\ \rho e+\rho\Phi\end{bmatrix}+\frac{\partial}{\partial x}\begin{bmatrix}\rho u\\ \rho u^2+p\\ \rho uw\\ u(\rho e+\rho\Phi+p)\end{bmatrix}+\frac{\partial}{\partial z}\begin{bmatrix}\rho w\\ \rho uw\\ \rho w^2+p\\ w(\rho e+\rho\Phi+p)\end{bmatrix}=\begin{bmatrix}0\\ 0\\ -\rho g\\ 0\end{bmatrix} \tag{2.1}$$

where ρ is the density, u and w are the horizontal and vertical velocities, p is the pressure, e is the sum of the internal and the kinetic energy, $\Phi = gz$ is the time independent gravitation potential with g the gravitational acceleration and z the elevation above sea level. The pressure p is related to the flux variables ρ, ρu, ρw and ρe over the thermodynamic equation of state

$$p=\phi(\rho,\rho u,\rho w,\rho e):=(\gamma-1)(\rho e-\frac{1}{2}\rho(u^2+w^2)) \tag{2.2}$$

where $\gamma := c_p/c_v = 1/(1-\kappa) = 1.4$ is the adiabatic exponent. As our domain size is about 100 km wide the Coriolis terms are neglected. Defining a state vector q, horizontal and vertical flux vectors f and h and a source vector s as

$$q:=\begin{bmatrix}\rho\\ \rho u\\ \rho w\\ \rho e+\rho\Phi\end{bmatrix}, \tag{2.3}$$

$$f:=\begin{bmatrix}\rho u\\ \rho u^2+p\\ \rho uw\\ u(\rho e+\rho\Phi+p)\end{bmatrix}, \tag{2.4}$$

$$h:=\begin{bmatrix}\rho w\\ \rho uw\\ \rho w^2+p\\ w(\rho e+\rho\Phi+p)\end{bmatrix}, \tag{2.5}$$

$$s:=\begin{bmatrix}0\\ 0\\ -\rho g\\ 0\end{bmatrix} \tag{2.6}$$

the inhomogeneous Euler equations read

$$\frac{\partial q}{\partial t}+\frac{\partial f}{\partial x}+\frac{\partial h}{\partial z}=s. \tag{2.7}$$

2.2 The integral form of the inhomogeneous Euler equations

Alternatively to the differential form (2.7) the inhomogeneous Euler equations can also be written in the integral form

$$\frac{d}{dt}\int_V q\,dV + \oint_{\partial V} f(q)\,n^x\,dS + \oint_{\partial V} h(q)\,n^z\,dS = \int_V s(q)\,dV. \tag{2.8}$$

Here n^x and n^z are the horizontal and vertical part of the outward pointing unity vector $\boldsymbol{n}$ normal to the boundary of volume V. The integral form states that a change of the total amount of q over time in the control volume V can only result from fluxes through its boundary or from a source term inside the control volume. To simplify the notation we define the outward flux normal to the volume boundary as

$$\breve{f}(q) := n^x f(q) + n^z h(q). \tag{2.9}$$

Then (2.8) becomes

$$\frac{d}{dt}\int_V q\,dV + \oint_{\partial V} \breve{f}(q)\,dS = \int_V s(q)\,dV. \tag{2.10}$$

2.3 Finite volume method

Because we want to compare our cut-cell approach with the terrain-following approach we need a method that is easily adaptable to different grids. Finite volume methods feature this property and are therefore well suited to our problem. To this end we divide our whole domain into grid cells c_i and define the cell average of the state vector q over grid cell c_i as

$$q_i(t) := \frac{1}{|c_i|}\int_{c_i} q(x,t)\,dV. \tag{2.11}$$

Then the change of q over time in any grid cell c_i can be calculated with

$$\frac{\partial q_i(t)}{\partial t} = -\frac{1}{|c_i|}\oint_{\partial c_i} \breve{f}(q)\,dS + \frac{1}{|c_i|}\int_{c_i} s(q)\,dV. \tag{2.12}$$

2.4 Discrete Archimedes' principle

In any grid cell c_i let p_i^h and ρ_i^h be exact solutions of the hydrostatic relation

$$\rho_i^h \nabla \Phi = -\nabla p_i^h \tag{2.13}$$

where p_i^h is the local pressure and ρ_i^h the local density. Integrating (2.13) over grid cell c_i and using Gauss' divergence theorem to replace the volume integral of pressure on the right hand side by the surface integral over the cell boundary ∂c_i yields the so called discrete Archimedes' principle of buoyancy

$$\frac{1}{|c_i|}\int_{c_i} \rho_i^h \nabla \Phi \, dV = -\frac{1}{|c_i|}\oint_{\partial c_i} p_i^h \boldsymbol{n} \, dS. \tag{2.14}$$

Now assume that we have approximations P_i and R_i to the exact data p and ρ on cell c_i. Then standard finite volume approximations of the cell averages of the pressure gradient and source term are given by

$$\delta_{c_i}(\nabla p) \approx \frac{1}{|c_i|}\int_{c_i} \nabla P_i dV, \tag{2.15}$$

$$\delta_{c_i}(\rho \nabla \Phi) \approx \frac{1}{|c_i|}\int_{c_i} R_i \nabla \Phi \, dV. \tag{2.16}$$

Using Gauss' divergence theorem to replace the volume integral in (2.15) by a surface integral yields

$$\delta_{c_i}(\nabla p) \approx \frac{1}{|c_i|}\oint_{\partial c_i} P_i \boldsymbol{n} \, dS \tag{2.17}$$

where **n** is the outward pointing vector normal to the boundary ∂c_i. Now let P_i^h and R_i^h be exact solutions of (2.13) interpolating P_i and R_i in the center of cell c_i. Using the discrete Archimedes' principle (2.14) the integral in (2.16) can be replaced, with second order accuracy, by

$$\frac{1}{|c_i|}\int_{c_i} R_i \nabla \Phi \, dV = \frac{1}{|c_i|}\int_{c_i} R_i^h \nabla \Phi \, dV + O(h^2) = -\frac{1}{|c_i|}\oint_{\partial c_i} P_i^h \boldsymbol{n} \, dS + O(h^2). \tag{2.18}$$

Thus we have approximated the source term in the finite volume formulation (2.12)

$$\frac{1}{|c_i|}\int_{c_i} s(q) \, dV = \delta_{c_i}(\rho \nabla \Phi) \tag{2.19}$$

with second order by means of a boundary integral of the local hydrostatic pressure function P_i^h:

$$\frac{1}{|c_i|}\int_{c_i} s(q) \, dV \approx -\frac{1}{|c_i|}\oint_{\partial c_i} P_i^h \boldsymbol{n} \, dS. \tag{2.20}$$

Let us now consider the balance between pressure gradient and source term. In nearly hydrostatic flows the difference between the two terms is

$$(\nabla p + \rho \nabla \Phi) \cdot k = O(\epsilon) \tag{2.21}$$

where ϵ is the order of magnitude of the vertical acceleration and k is the unit vector aligned with the gravity acceleration. In nearly hydrostatic flows the vertical acceleration is much smaller than the two almost equal quantities on the left hand side of (2.21). We give here an example to illustrate the proportions. In [8] Holton shows in a scale analysis of the vertical momentum tendency that pressure gradient and source term both have a magnitude of $10\ \mathrm{m}/s^2$. The simulation of a nearly hydrostatic flow with our model yields a vertical acceleration of the order of $10^{-4}\,\mathrm{m/s^2}$.

The problem of standard numerical models is that on a grid of spacing h any standard numerical method of order r introduces local truncation errors (LTEs) of order h^r to the approximations to $\rho \nabla \Phi$ and ∇p. Therefore spurious accelerations of order h^r are introduced to the balance that may be orders of magnitude larger then the physical vertical acceleration. Thus, keeping the errors small demands a high resolution and/or high order scheme. To circumvent this problem well-balanced methods use the representation of the source term in terms of the local hydrostatic background pressure function P_i^h. With (2.20) the second order approximation to the cell average of $\nabla p + \rho \nabla \Phi$ becomes

$$\delta_{c_i}(\nabla p + \rho \nabla \Phi) \approx \frac{1}{|c_i|} \oint_{\partial c_i} (P_i - P_i^h)\, \boldsymbol{n}\, dS. \tag{2.22}$$

With this the introduced LTEs can be reduced considerably. As long as the deviations from the local hydrostatic background state are of order ϵ, i.e.

$$\nabla (P_i - P_i^h) \cdot k = O(\epsilon), \tag{2.23}$$

$$(R_i - R_i^h) \nabla \Phi \cdot k = O(\epsilon) \tag{2.24}$$

the balanced LTEs we introduce to the cell average of the pressure gradient are only of order ϵh^r instead of h^r.

3 Spatial discretization

In this section we show how the right hand side terms of the finite volume formulation (2.12) are spatially discretized. We then have a semi-discrete formulation

$$\frac{\partial q_i(t)}{\partial t} = \psi_i(q) \tag{3.1}$$

where $\psi_i(q)$ is the spatially discretized right hand side. To solve this differential equation in time we can use a standard time integration scheme asuch s the leapfrog scheme or a Runge-Kutta scheme. Details on time integration are given in section 4.

On a two dimensional, logically rectangular grid the finite volume formulation for any grid cell c_{ij} reads

$$\frac{\partial q_{ij}(t)}{\partial t} = -\frac{1}{|c_{ij}|}\oint_{\partial c_{ij}} \breve{f}(q)\, dS + \frac{1}{|c_{ij}|}\int_{c_{ij}} s(q)\, dV. \tag{3.2}$$

Here $\breve{f}$ is the outward flux normal to the cell boundary defined in (2.9). Using midpoint approximation for the integrals over the boundary interfaces then yields

$$\frac{\partial q_{ij}(t)}{\partial t} = -\frac{1}{|c_{ij}|}\sum_{k=1}^{4}\left[\breve{F}_k(t)\, I_k\right]_{ij} + S_{ij}(t) \tag{3.3}$$

where $\breve{F}_k$ is the numerical outward flux normal to the k-th (east, north, west and south) interface, I_k is the area of the k-th interface and S_{ij} is the approximated source term inside cell c_{ij}. We now have to calculate the numerical flux function at the interfaces and the source term.

3.1 Calculation of the numerical flux

To approximate the flux $\breve{f}(q)$ over the cell interfaces we need a numerical flux function $\breve{F}_k$ at these interfaces and the interface areas I_k. For the eastern interface of cell c_{ij} at time step t the numerical flux must fulfill

$$\breve{F}_{i+,j}(t) \approx \frac{1}{\Delta z}\int_{z_{i+,j-}}^{z_{i+,j+}} \breve{f}(q_{i+,j}(t))\, dz. \tag{3.4}$$

Here the indexes $i+, j\pm$ denote the bottom and top of the eastern interface respectively. The integrals are approximated by the analytic fluxes at the interface midpoints. This yields the numerical flux function

$$\breve{F}_{i+,j}(t^n) = \breve{f}(Q^*_{i+,j}(t^n)) \tag{3.5}$$

where $Q^*_{i+,j}$ is an approximation to the state vector at the eastern interface midpoint of cell c_{ij}. Introducing a new index at the top of the reconstructed state vector Q at the interfaces, denoting the direction of reconstruction, the average state can be written as

$$Q^*_{i+,j} = \frac{1}{2}(Q^{east}_{ij} + Q^{west}_{i+1\,j}). \tag{3.6}$$

In other words $Q^{*}_{i+,j}$ is calculated by averaging the reconstructions from the center of cell c_{ij} to the east and from the center of cell $c_{i+1\,j}$ to the west. Averaging the state at the interfaces assures the uniqueness of the values used for the numerical fluxes.

The numerical flux at the northern, western and southern interface is computed analogously. But since two cells lying next to each other share an interface we calculate only the numerical flux at the western and southern interfaces. The numerical flux at the eastern and northern interfaces is then calculated with

$$\breve{F}_{i+,j}(t) = -\breve{F}_{i+1-,j}(t), \tag{3.7}$$
$$\breve{F}_{i,j+}(t) = -\breve{F}_{i,j+1-}(t). \tag{3.8}$$

With the notation $h_{i-,j}$ and $h_{i,j-}$ for the western and southern interface area of cell c_{ij} the sum of the fluxes on the right hand side of (3.3) can be calculated as

$$\sum_{k=1}^{4} \left[\breve{F}_k(t) I_k \right]_{ij} = -h_{i+1-,j} \cdot \breve{F}_{i+1-,j}(t) - h_{i,j+1-} \cdot \breve{F}_{i,j+1-}(t) + h_{i-,j} \cdot \breve{F}_{i-,j}(t) + h_{i,j-} \cdot \breve{F}_{i,j-}(t). \tag{3.9}$$

3.2 Reconstructing the variables at the cell interfaces

To evaluate the numerical flux function we need to reconstruct the complete state vector at the cell interface midpoints. In this reconstruction well balanced and standard methods differ. Well balanced methods use the local hydrostatic background state to interpolate the complete state vector at the interfaces where standard methods usually simply interpolate the data between two neighboring cells. The evaluation of the local hydrostatic background state is explained in section 3.5

We now show how the complete state is reconstructed at the interface midpoints in our well balanced model. Basically we use a linear Taylor approximation. As we do not necessarily evaluate the local hydrostatic background state at every time step a time derivative appears in addition to the spatial gradient term. Eventually the local approximation Q_{ij} to the complete state vector at the interface midpoints at time Δt after the last evaluation of the background state reads

$$Q_{ij}(x_{i\pm,j}, t_0 + \Delta t) = Q^{h}_{ij}(\zeta_{i\pm,j}, t_0) + (x_{i\pm,j} - x_{ij}) \cdot G_i \delta Q_{ij}(t_0) + \Delta t \cdot G_t Q_{ij}(t_0), \tag{3.10}$$
$$Q_{ij}(x_{i,j\pm}, t_0 + \Delta t) = Q^{h}_{ij}(\zeta_{i,j\pm}, t_0) + (x_{i,j\pm} - x_{ij}) \cdot G_j \delta Q_{ij}(t_0) + \Delta t \cdot G_t Q_{ij}(t_0). \tag{3.11}$$

Here t_0 is the time the local hydrostatic background state Q^{h}_{ij} was evaluated last and $G_i \delta Q_{ij}$ and $G_j \delta Q_{ij}$ are the two dimensional approximations to the gradients of the deviation of the complete state from the local hydrostatic background state along the computational dimensions *i* and *j* respectively. For a more detailed derivation of the spatial gradients see Wunderlich in [6]. How the gradients are calculated is shown in section 3.3. The time derivative vector in (3.10) and (3.11) is given by

$$G_t Q_{ij}(t_0) = \frac{1}{\Delta t} \left(Q_{ij}(x_{ij}, t_0 + \Delta t) - Q_{ij}(x_{ij}, t_0) \right). \tag{3.12}$$

Note that $Q^{h}_{ij}(\zeta, t_0 + \Delta t) \equiv Q^{h}_{ij}(\zeta, t_0)$ where t_0 is the last time the local hydrostatic background state was calculated and Δt is the time that has passed since then.

3.3 Computing the gradients

The evaluation of the complete state at the cell interfaces in (3.10) and (3.11) requires an approximation to the gradients of the deviation from the local hydrostatic background state along the computational dimensions *i* and *j*. We calculate the deviations in the center of all neighbor cells of cell c_{ij}. For the eastern cell this deviation is

$$\delta Q_{i+1,j} = Q_{i+1,j} - Q^h_{ij}(\zeta_{i+1,j}) \tag{3.13}$$

where $\zeta_{i+1,j}$ is the value of the local vertical coordinate ζ evaluated at the center of cell $c_{i+1,j}$. For the other three cells the deviation is calculated analogously. From this four deviations we can compute four gradients, two along each computational dimension. How this is done in detail for curvilinear grids taking metric terms into account is shown in [6]. To avoid spurious oscillations the two gradients along the same computational direction are smoothed using a gradient limiter function. To smoothen the gradients g_1 and g_2 we have used either the minmod limiter

$$\mathcal{L}(g_1, g_2) := \frac{1}{2}(sign(g_1) + sign(g_2)) \cdot min(|g_1|, |g_2|), \tag{3.14}$$

the monotonized central limiter

$$\mathcal{L}(g_1, g_2) := \frac{1}{2}(sign(g_1) + sign(g_2)) \cdot min\left(2 \cdot min(|g_1|, |g_2|), \frac{|g_1 + g_2|}{2}\right) \tag{3.15}$$

or the Van Leer (sometimes also called WENO (weighted essentially non oscillatory)) limiter

$$\mathcal{L}(g_1, g_2) := \frac{w(g_2) \cdot g_1 + w(g_1) \cdot g_2}{w(g_1) + w(g_2)} \tag{3.16}$$

where *w* is the regulated absolute value $w(g_i) = \sqrt{(g_i^2 + \varepsilon)}$, and $\varepsilon = 10^{-6}$.

3.4 Calculation of the source term

As seen in section 2.4 the source term inside cell c_{ij} can be approximated with second order by

$$S_{ij} = -\frac{1}{|c_{ij}|} \oint_{\partial c_{ij}} P^h_{ij} \boldsymbol{n} \, dS. \tag{3.17}$$

To perform a discrete integration of the source term in each cell the hydrostatic background pressure has to be defined at the interfaces. In section 3.5 the hydrostatic background state is defined for each cell as a function of the local vertical coordinate ζ. The discrete integration of the vertical momentum component of the source term then reads

$$S^{\rho w}_{ij}(t) = -\frac{1}{|c_{ij}|} \sum_{k=1}^{4} \left[P^h_{ij}(\zeta_k) \cdot h_k \cdot n^z_k \right]_{ij}. \tag{3.18}$$

All three other components of the source term of the inhomogeneous Euler equations (2.1) equal zero.

3.5 Reconstruction of the local hydrostatic background state

We need the local hydrostatic background state at the cell interfaces to approximate the source term inside each cell c_{ij} and to reconstruct the complete state at the interface midpoints. We therefore define the local hydrostatic background state for each cell as a function of a local vertical coordinate ζ. The local hydrostatic background state has to fulfill the following three requirements in each cell c_{ij} of the computational domain:

1. fulfill the hydrostatic relation

$$\frac{\partial P_{ij}^h(\zeta)}{\partial \zeta} = -g\, R_{ij}^h(\zeta), \tag{3.19}$$

2. interpolate the averaged data in the cell midpoint:

$$Q_{ij}^h(0) = Q_{ij}, \tag{3.20}$$

3. fulfill the thermodynamic equation of state for ideal gases

$$\Theta_{ij}^h(\zeta) = \frac{P_{ij}^h(\zeta)}{R\, R_{ij}^h(\zeta)} \left(\frac{p_{00}}{P_{ij}^h(\zeta)} \right)^{\kappa} \tag{3.21}$$

where p_{00} is the reference pressure defined at sea level and R is the gas constant of dry air. Using (3.21) to replace the expression for $R_{ij}^h(\zeta)$ in (3.19) yields the ordinary differential equation

$$\frac{\partial P_{ij}^h(\zeta)}{\partial \zeta} = -g \frac{P_{ij}^h(\zeta)}{R\, \Theta_{ij}^h(\zeta)} \left(\frac{p_{00}}{P_{ij}^h(\zeta)} \right)^{\kappa} \tag{3.22}$$

for the pressure $P_{ij}^h(\zeta)$ of the local hydrostatic background state. Together with a given potential temperature profile and the interpolation condition (3.20) the ODE (3.22) can be solved. How this is done in detail for different temperature profiles is shown by Müller in [7]. With the solution of equation (3.22) we can reconstruct the complete local hydrostatic background state as a function of ζ in each cell analytically assuming that the horizontal and vertical velocities are constant in each cell.

4 Numerical integration and further developments

In section 3 we spatially discretized the right hand side terms of the finite volume formulation (2.12). We now want to solve the resulting differential equation

$$\frac{\partial q_i(t)}{\partial t} = \psi_i(q) \tag{4.1}$$

where $\psi_i(q)$ is the spatial discretization of the right hand side terms.

4.1 The 2nd order leapfrog scheme

A very common and efficient scheme for time integration is the 2nd order leapfrog scheme. For some given numerical approximations Q^n and Q^{n-1} to the analytic states $q(x, t^n)$ and $q(x, t^{n-1})$ the scheme to calculate the numerical approximation Q^{n+1} at time $t^{n+1} = t^n + \Delta t$ is

$$Q^{n+1} = Q^{n-1} + 2\Delta t \cdot \psi(Q^n). \tag{4.2}$$

Because the 2nd order leapfrog scheme is a three-level scheme we need the states Q^0 and Q^1 to initiate the time integration. Q^0 is given by the initial state of the atmosphere and Q^1 is calculated using a standard two-level scheme. In addition to the leapfrog scheme Wunderlich [6] also implemented a 3rd order Runge-Kutta scheme which we use to calculate Q^1.

The main problem when using the leapfrog scheme is the appearance of computational modes which can lead to numerical instabilities. To prevent these modes from growing we use an Asselin filter where the filtered value $\overline{Q^n}$ is given by

$$\overline{Q^n} = Q^n + \alpha_{ass} \cdot (Q^{n+1} - 2Q^n + \overline{Q^{n-1}}). \tag{4.3}$$

The damping coefficient α_{ass} is a real, positive number typically chosen between 0.05 and 0.2 for atmospheric modeling. Taking the Asselin filter into account the leapfrog scheme becomes

$$Q^{n+1} = \overline{Q^{n-1}} + 2\Delta t \cdot \psi(Q^n). \tag{4.4}$$

In [5] the authors suggested to rewrite the inhomogeneous Euler equations using the homogeneous state vector

$$q = \begin{pmatrix} \rho \\ \rho u \\ \rho w \\ \rho e \end{pmatrix} \tag{4.5}$$

instead of the inhomogeneous state vector defined in (2.3). The state vector we use in our model is as defined here in (4.5). Although this is just an implementation detail we consider here the effects of this change for the sake of completeness. The change yields an additional term on the right hand side of the inhomogeneous Euler equations which then are

$$\frac{\partial q}{\partial t} = -\frac{\partial f}{\partial x} - \frac{\partial h}{\partial z} + s + r \tag{4.6}$$

where

$$r = \begin{pmatrix} 0 \\ 0 \\ 0 \\ -(\rho \Phi)_t \end{pmatrix} \tag{4.7}$$

is the additional term added to the right hand side. Using the homogeneous state vector (4.5) the energy equation of (4.1) becomes

$$\frac{\partial(\rho e)}{\partial t} = \psi(q) - \frac{\partial(\rho \Phi)}{\partial t}. \tag{4.8}$$

In the used leapfrog scheme the energy equation therefore has to be updated after each time step by adding the r term to the right hand side

$$(\rho e)^{n+1} \Leftarrow (\rho e)^{n+1} - \Phi(\rho^{n+1} - \rho^{n-1}). \tag{4.9}$$

4.2 Divergence damping

In [6] Wunderlich showed how to implement a divergence filter. By adding a numerical diffusion term

$$d = \begin{pmatrix} 0 \\ d_{\rho u} \\ d_{\rho w} \\ 0 \end{pmatrix} \tag{4.10}$$

to the right hand side of the semi discrete equation (4.1) acoustic waves can be damped. Surprisingly his model is very sensitive to the values chosen for the divergence filter coefficients. It was found that the damping term had not been implemented correctly. There had been an error in the implementation of the mixed second derivatives. This led to incorrect results and to the observed sensitivities. Replacing the old momentum damping term

$$d_{\rho u, \rho w} = \begin{pmatrix} \tilde{\alpha}_x \left(\dfrac{\partial^2 (\rho u)}{\partial x^2} + \dfrac{\partial^2 (\rho u)}{\partial x \partial z} \right) \\ \tilde{\alpha}_z \left(\dfrac{\partial^2 (\rho w)}{\partial x \partial z} + \dfrac{\partial^2 (\rho w)}{\partial z^2} \right) \end{pmatrix} \tag{4.11}$$

with the correct term

$$d_{\rho u, \rho w} = \begin{pmatrix} \tilde{\alpha}_x \left(\dfrac{\partial^2 (\rho u)}{\partial x^2} + \dfrac{\partial^2 (\rho w)}{\partial x \partial z} \right) \\ \tilde{\alpha}_z \left(\dfrac{\partial^2 (\rho u)}{\partial z \partial x} + \dfrac{\partial^2 (\rho w)}{\partial z^2} \right) \end{pmatrix} \tag{4.12}$$

makes the model less sensitive to the choice of the divergence filter coefficients $\tilde{\alpha}_x$ and $\tilde{\alpha}_z$.

To take the local hydrostatic background state into account we implemented a new damping method. The new damping term replaces the old momentum damping term with

$$d^h_{\rho u, \rho w} = \begin{pmatrix} \tilde{\alpha}_x \left(\dfrac{\partial^2 (\rho^h u)}{\partial x^2} + \dfrac{\partial^2 (\rho^h w)}{\partial x \partial z} \right) \\ \tilde{\alpha}_z \left(\dfrac{\partial^2 (\rho^h u)}{\partial z \partial x} + \dfrac{\partial^2 (\rho^h w)}{\partial z^2} \right) \end{pmatrix} \tag{4.13}$$

where ρ^h denotes the density of the local hydrostatic background state. The default term we use is (4.13) but the old, corrected damping term (4.12) is also available as an option of the leapfrog scheme.

To calculate the damping term in each cell c_{ij} the derivatives with respect to x are approximated using finite differences:

$$\left\langle \frac{\partial^2 \rho^h u}{\partial x^2} \right\rangle_{ij} \approx \frac{\rho^h_{ij}(\zeta_{i+1,j}) u_{i+1,j} \cdot (x_{ij} - x_{i-1,j}) + \rho^h_{ij}(\zeta_{i-1,j}) u_{i-1,j} \cdot (x_{i+1,j} - x_{ij}) - \rho^h_{ij}(0) u_{ij} \cdot (x_{i+1,j} - x_{i-1,j})}{\frac{1}{2}(x_{i+1,j} - x_{i-1,j})(x_{i+1,j} - x_{ij})(x_{ij} - x_{i-1,j})}. \quad (4.14)$$

The derivatives with respect to z are computed analogously. The mixed second derivatives are approximated using centered differences first in x direction and then in z direction or vice versa. In the discrete case the damping coefficients $\tilde{\alpha}_x$ and $\tilde{\alpha}_z$ are

$$\tilde{\alpha}_x = \alpha_x \frac{\Delta x^2}{\Delta t} \text{ and } \tilde{\alpha}_z = \alpha_z \frac{\Delta z^2}{\Delta t} \quad (4.15)$$

where α_x and α_z are non-dimensional coefficients. To ensure stability the divergence damping is applied to the old values of the state vector q. According to the ARPS User Guide [9] the coefficients α_x and α_z have to be smaller than 0.75 to assure stability.

4.3 Computational mixing

We introduce numerical smoothing in the computational space in order to remove small scale noise of computational origin following the ARPS User Guide [9] . Therefore second order computational mixing terms $C_{\rho u, \rho w}$ are added to the right hand side of the conservation equations of momentum. Like divergence damping the mixing is applied to the previous time level to ensure stability. The mixing is applied to the perturbations u' and w' form the local hydrostatic background state. The mixing terms using a second order finite difference operator along the computational variables ξ and ζ in each cell c_{ij} are:

$$\begin{aligned} C_{\rho\phi} = {} & K_x(\rho^h_{ij}(\zeta_{i-1,j})\phi'_{i-1,j} - 2\rho^h_{ij}(\zeta_{ij})\phi'_{ij} + \rho^h_{ij}(\zeta_{i+1,j})\phi'_{i+1,j}) \\ & + K_z(\rho^h_{ij}(\zeta_{i,j-1})\phi'_{i,j-1} - 2\rho^h_{ij}(\zeta_{ij})\phi'_{ij} + \rho^h_{ij}(\zeta_{i,j+1})\phi'_{i,j+1}) \end{aligned} \quad (4.16)$$

where ϕ is either u or w. Note that $\rho^h_{ij}(\zeta_{ij}) = \rho^h_{ij}(0)$ since ζ is the local vertical coordinate. The parameters K_x and K_z equal $\alpha_x / \Delta t$ and $\alpha_z / \Delta t$ respectively. Subject to [9] the non dimensional coefficients α_x and α_z must be smaller then 1/8 to guarantee stability. If not stated otherwise we use a value of 0.0005 for both coefficients in order to keep the computational mixing small.

4.4 Diffusion

An additional diffusion term

$$diff_{\rho u,\rho w} = \rho K \begin{pmatrix} u_{xx} + u_{zz} \\ w_{xx} + w_{zz} \end{pmatrix} \tag{4.17}$$

is added to the right hand side of the momentum equations to simulate diffusion. The second derivatives with respect to x and z are approximated using finite differences analogously to (4.14). K is the diffusion coefficient.

4.5 Rayleigh sponge layer

Also according to the ARPS User Guide [9] a damping layer is introduced at the top boundary in order to absorb upward propagating wave disturbances and to eliminate wave reflection at the top boundary. For this purpose a Rayleigh damping term

$$R_D = -\alpha_R \sin^2\left(\frac{\pi}{2}\frac{z - z_D}{z_T - z_D}\right)\cdot(q - q^0) \tag{4.18}$$

is added to the right hand side of the conservation equations. The Rayleigh damping term operates on the perturbation of the state vector q from the undisturbed initial state q^0. It is applied after updating the energy equation in the leapfrog scheme. z_D is the height of the bottom of the damping layer and z_T is the height of the top boundary. Typically the thickness of the layer corresponds to about 1.5 vertical wavelengths. The inverse damping coefficient α_R^{-1} has a magnitude of 10 to 50 time steps.

4.6 Stability

In two dimensions the CFL criterion for stability including diagonally propagating waves is

$$\left|\frac{max_p \lambda_{i,j}^p \cdot \Delta t}{min_{i,j}(\Delta x_i, \Delta z_j)}\right| \leq \frac{\sqrt{2}}{2} \tag{4.19}$$

where Δx_i and Δz_j are the horizontal and vertical, nonuniform grid spacings and $max_p \lambda_{i,j}^p$ is the speed of the fastest propagating wave in x- or z-direction. However, in our atmospheric simulations the wave speeds are dominated by the sound speed

$$c := \sqrt{\gamma R T} \tag{4.20}$$

where γ is the adiabatic coefficient, R is the gas constant for dry air and T is the temperature in Kelvin. Thus we use c as an approximation to $max_p \lambda_{i,j}^p$. The size of the time step is therefore limited by

$$\Delta t \leq \frac{\sqrt{2}}{2} \cdot \left|\frac{min_{i,j}(\Delta x_i, \Delta z_j)}{c}\right|. \tag{4.21}$$

With a grid spacing of 300 m a value of 0.4 seconds is an appropriate choice for the time step. To prevent the simulations from running into instability we implemented a CFL break condition that stops the simulation as soon as the CFL number gets too big.

5 Boundary conditions and grids

The number of necessary ghost cells outside the physical domain is given by the reconstruction of the state variables at the interfaces. To reconstruct the gradients in the outermost cells of the physical domain we only need one row and column of ghost cells. But in order to calculate the numerical flux at the physical domain boundary we also need the complete state there reconstructed from the ghost cell adjacent to the physical domain boundary, as seen in section 3.1. We therefore need a second row and column of ghost cells to calculate the limited gradients in the inner ghost cells.

As lateral boundary conditions we use periodic conditions. The boundary conditions for the velocity at the top and bottom are free-slip conditions, i.e. the velocity component orthogonal to the domain boundary must be zero. Let v be the velocity vector and n the normal vector pointing outward of the atmosphere. Then the top and bottom boundary conditions for the velocity are

$$v \cdot n = 0. \tag{5.1}$$

Density and energy density at the top and bottom must have a vanishing normal derivative. With $\tilde{q} := (\rho, \rho e)^T$ being the vector of density and energy density the boundary conditions for $\tilde{q}$ are

$$\frac{\partial (\tilde{q} - \tilde{q}^{init})}{\partial n} = 0. \tag{5.2}$$

The boundary conditions (5.2) for the inner and outer ghost cells at the top and bottom then take the form

$$\tilde{q}^{n+1}_{i,inner} = \tilde{q}^{n+1}_{i,1} - \tilde{q}^{init}_{i,1} + \tilde{q}^{init}_{i,inner}, \tag{5.3}$$

$$\tilde{q}^{n+1}_{i,outer} = \tilde{q}^{n+1}_{i,2} - \tilde{q}^{init}_{i,2} + \tilde{q}^{init}_{i,outer} \tag{5.4}$$

where the indexes *1* and *2* refer to the first two rows of cells in the physical domain either from the top or from the bottom. The velocity constraint (5.1) is realized by reflecting the velocity vector v of any inner state q at the boundary. This is done using the reflection operator

$$R(v, n) = v - 2(v \cdot n)n \tag{5.5}$$

to calculate the velocities in both the inner and the outer ghost cells.

5.1 Uniform Cartesian grid using cut cells

To avoid errors coming from metric terms we replace the terrain-following grid with a uniform Cartesian grid. Using such a uniform Cartesian grid requires a special treatment of the boundary cells that are cut by the topography. In [10] Forrer described a new way to treat these cut cells. To avoid numerical instabilities due to small cell sizes he suggested to treat the cut cells as whole cells instead of using only the part of the cut cells lying in the physical domain. A boundary treatment was developed that locally reflects the velocity fields at a straight boundary line. The prognostic equations in the cut cells are then solved as if there was no boundary. All cell values lying outside the physical domain required to update the state variables in the cut cells are locally reflected from the inside of the physical domain. At last the boundary conditions (5.2) for density and energy density are applied.

We now show how the local reflection is done. Inside the cut cells the topography is approximated by a straight line that serves as symmetry line. Suppose this symmetry line goes through the point $\boldsymbol{m}$ and the vector normal to the line pointing into the physical domain is given by

$$\boldsymbol{n} = \begin{pmatrix} -\sin\alpha \\ \cos\alpha \end{pmatrix} \tag{5.6}$$

where α is the lead angle of the line. Any point $\boldsymbol{p} = (x, z)^T$ outside the physical domain can now be reflected at the symmetry line to the point

$$r_{m,\alpha}(\boldsymbol{p}) = \boldsymbol{p} - 2((\boldsymbol{p} - \boldsymbol{m}) \cdot \boldsymbol{n}) \cdot \boldsymbol{n} \tag{5.7}$$

lying inside the physical domain.

Then the state vector q can be extrapolated to the point $\boldsymbol{p}$ with

$$q(\boldsymbol{p}) = R_\alpha \cdot q(r_{m,\alpha}(\boldsymbol{p})) \tag{5.8}$$

where R_α is the reflection matrix

$$R_\alpha = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(2\alpha) & \sin(2\alpha) & 0 \\ 0 & \sin(2\alpha) & -\cos(2\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{5.9}$$

Using (5.9) we can now define the state vector inside any cut cell c_{ij} as

$$q_{ij} = \frac{1}{|c_{ij}|}\left(\int_{c_{ij}^1} q(x,z)\,dx\,dz + \int_{r_{ij}(c_{ij}^2)} R_{\alpha_{ij}} \cdot q(x,z)\,dx\,dz \right). \tag{5.10}$$

Here c_{ij}^1 is the part of the cut cell lying inside the physical domain and $r_{ij}(c_{ij}^2)$ is the reflected part of the cut cell originally lying outside the physical domain, see figure 5.1.

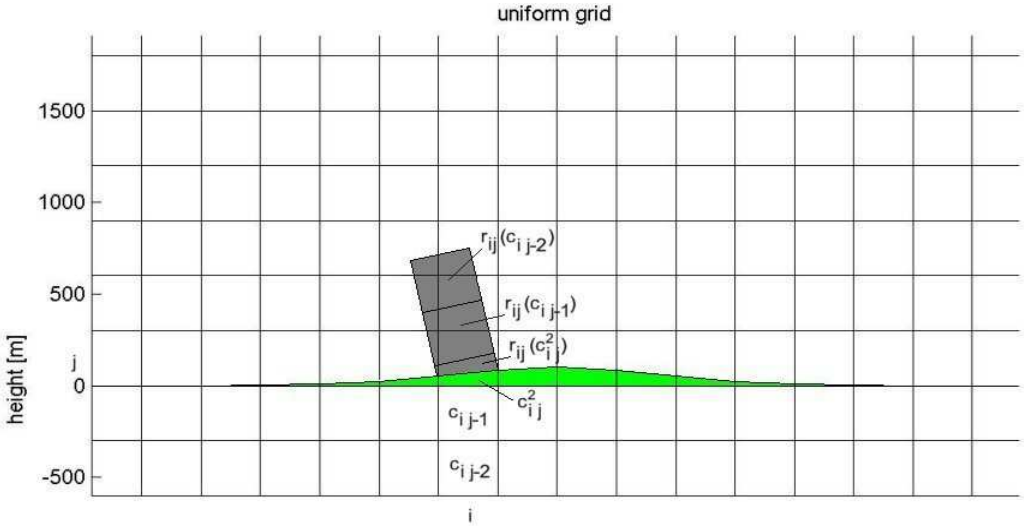

Figure 5.1: Illustration of locally reflected cells in a uniform Cartesian grid with cut cells

In order to update the state vector in the cell c_{ij} we need the complete actual state vectors in the regular cells $c_{i,j+1}, c_{i,j+2}$, in the cut cells $c_{i-2,j}, c_{i-1,j}, c_{i+1,j}, c_{i+2,j}$ and in the empty cells $c_{i,j-1}, c_{i,j-2}$. Note the simpler stencil in comparison to the terrain-following grid. The state vector in the regular cells is given by (2.11) and in the cut cells it is defined by equation (5.10). To calculate the state vector in the empty cells we again use the assumption of local reflection of the velocity field. For the cell $c_{i,j-1}$ this yields

$$q_{i,j-1} = \int_{r_{ij}(c_{i,j-1})} R_{\alpha_{ij}} \cdot q(x,z)\,dx\,dz. \tag{5.11}$$

To evaluate the integral in (5.11) numerically we make a bilinear interpolation of the function $q(x,z)$ at the reflected cell center $r_{m_{i,j-1},\alpha_{ij}}(\boldsymbol{C}_{i,j-1})$ between the centers of the surrounding regular or cut cells, cf. figure 5.1. The coordinates of the reflected cell center can be calculated with

$$r_{m_{i,j-1},\alpha_{ij}}(\boldsymbol{C}_{i,j-1}) = \boldsymbol{C}_{i,j-1} - 2((\boldsymbol{C}_{i,j-1} - \boldsymbol{m}_{ij})\cdot \boldsymbol{n}_{ij})\cdot \boldsymbol{n}_{ij}. \tag{5.12}$$

After reflecting the velocity field we apply the boundary condition (5.2) for density and energy density to ensure that we have a vanishing normal derivative.

So far we have only implemented the case where any empty cell is used to update exactly one cut cell. In this case we can update the cut cells and the empty cells after each time step when we apply the boundary conditions. The case where an empty cell is used by two different cut cells is more complicated to solve and hasn't been implemented in this work. A solution would be to calculate the state vectors on the fly when they are needed to update the cut cells instead of calculating them before the actual updating.

5.2 Simplifications

Due to the use of a uniform Cartesian grid our model can be simplified in some aspects. There is no vertical flux over the eastern and western and no horizontal flux over the northern and southern cell boundaries anymore. This simplifies the numerical flux over a cell boundary in (3.3) to

$$\sum_{k=1}^{4} \left[\breve{F}_k(t) I_k \right]_{ij} = \Delta z (f(Q^*_{i+,j}(t)) + f(Q^*_{i-,j}(t))) + \Delta x (h(Q^*_{i,j+}(t)) + h(Q^*_{i,j-}(t))). \tag{5.13}$$

The normal vector on the eastern and western interface doesn't have any z-component anymore and the discrete integration of the vertical momentum component (3.18) becomes

$$S^{\rho w}_{ij}(t) = -\frac{1}{|c_{ij}|} \sum_{k=1}^{4} \left[p^h_{ij}(\zeta_k)\cdot h_k \cdot n^z_k \right]_{ij} = -\Delta z \frac{1}{|c_{ij}|} \left(p^h_{ij}(\zeta_{i,j+}) - p^h_{ij}(\zeta_{i,j-}) \right). \tag{5.14}$$

As the grid is now uniform the second derivatives with respect to x in (4.14) read

$$\left(\frac{\partial^2 \rho^h u}{\partial x^2} \right)_{ij} \approx \frac{\rho^h_{ij}(\zeta_{i+1,j}) u_{i+1,j} - 2\rho^h_{ij}(0) u_{ij} + \rho^h_{ij}(\zeta_{i-1,j}) u_{i-1,j}}{\Delta x^2} \tag{5.15}$$

and analogously for the z direction. Furthermore the local vertical coordinate ζ doesn't vary along the now horizontal i-coordinate anymore. This causes some additional simplifications in the calculation of the computational mixing and divergence damping terms as the local hydrostatic background density is now constant along the i-coordinate.

5.3 Terrain-following grid

The terrain-following grid we use in this work is given by Wunderlich in [6]. The coordinate surface follows the topography at the bottom and slowly level off to the top as in figure 5.2.

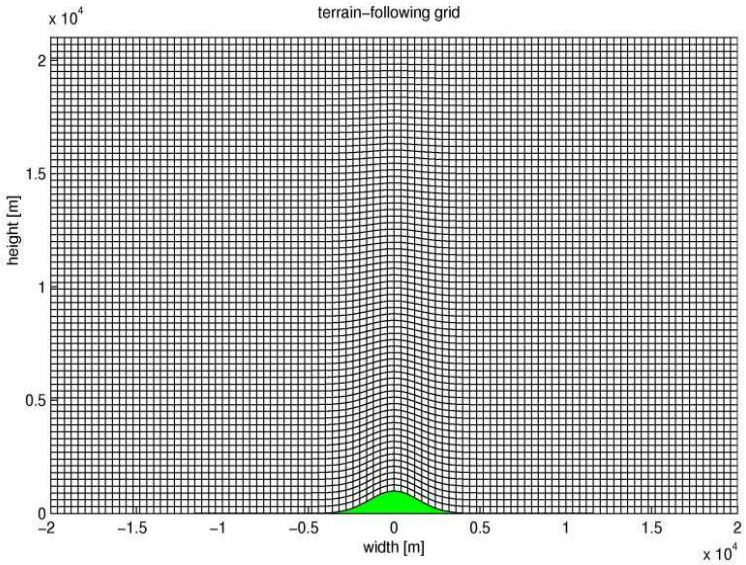

Figure 5.2: Terrain-following grid over a Gaussian hill

5.4 Hybrid coordinates

Instead of using terrain-following coordinates in the whole physical domain we have also the opportunity to use uniform coordinates in the upper part of the domain. This yields hybrid coordinates as in figure 5.3 where we use uniform coordinates above 12000 m and terrain-following below.

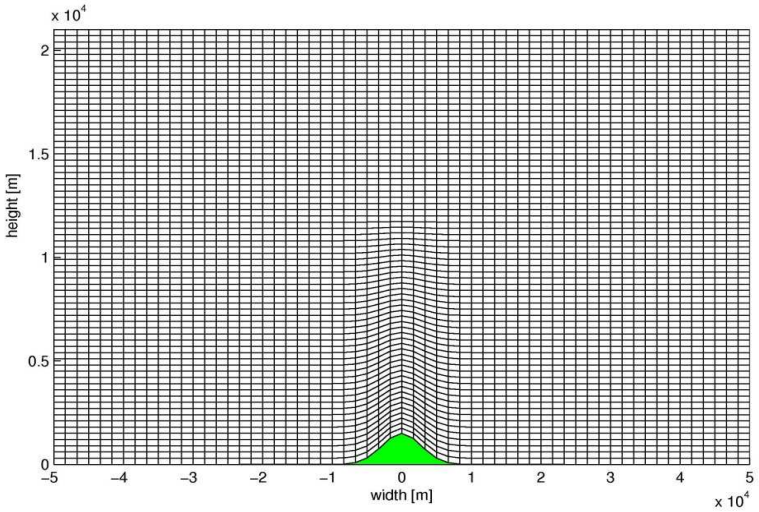

Figure 5.3: Hybrid coordinates over a Gaussian hill using uniform coordinates above and terrain-following coordinates below 12000 m

6 Numerical results

All tests are performed in two dimensions. The initial atmospheric state is either isentropic or constantly stratified (constant Brunt-Väisälä frequency). To simulate the behavior of a thermal bubble (see section 6.6) we have the possibility to insert an elliptic thermal perturbation into the initial state. More details on the typical potential temperature, density and pressure profiles of the two types of atmosphere are given in [6].

The steady state solutions produced by the flow of a stratified fluid over a mountain depends on the different height and width of the mountain. We distinguish between linear and nonlinear as well as hydrostatic and non-hydrostatic flows. To classify the flows we use the non-dimensional mountain width

$$\hat{a} = a\frac{N}{u_0} \tag{6.1}$$

and height

$$\hat{h} = h\frac{N}{u_0}. \tag{6.2}$$

Here a is the mountain half width, h is the height of the mountain and u_0 is the initial horizontal velocity. Hydrostatic flows are flows where the time a parcel of air needs to cross the mountain is much bigger than the inverse Brunt-Väisälä frequency *1/N*. With use of (6.1) a hydrostatic flow is therefore characterized by $\hat{a} \gg 1$. Linear flows are characterized by a low non-dimensional mountain height $\hat{h} \ll 1$ typically around 0.001. To test our model we simulate hydrostatic and non-hydrostatic, linear flows.

6.1 Setup

Three hill types with a maximum height H and a half width a are defined by the topography function $h(x)$. They are

the Gaussian hill

$$h(x) = H\exp\left(-(x/a)^2\right), \tag{6.3}$$

the Bell hill

$$h(x) = \frac{H}{1+(x/a)^2} \tag{6.4}$$

and the Schär hill

$$h(x) = H\exp\left(-(x/a)^2\right)\cos^2\left(\frac{x\pi}{\lambda}\right). \tag{6.5}$$

The parameter λ in (6.5) defines the wavelength of the Schär hill. We simulate the flow over a Gaussian hill as well as the flow over a Schär hill.

6.2 Performance improvement

The main part of increase in performance is due to the change in evaluating the local hydrostatic background state. Instead of doing the time consuming evaluation every time step we merely do it every 10^{th} time step. To illustrate the resulting increase in performance we consider a typical simulation of a linear, non-hydrostatic flow past a small Gaussian hill. The simulation is done using the cut-cell approach. The complete list of used parameters is given in table 6.1. For this simulation the number of calls for the procedure to calculate the local hydrostatic background state has been reduced from 12501 to 1251 calls. This reduces the totally used CPU time from about 6500 seconds to about 3400 seconds which is an increase in performance of more than 90%. However, not calling the procedure every time step does not have a noticeable impact on the solution. Figure 6.1 shows the result of a simulation where the local hydrostatic background state is evaluated every time step, figure 6.2 the same for every 10^{th} time step.

parameter	***value***	***unit***
time integration scheme	leapfrog	-
atmosphere	stratified	-
mountain height	1	m
mountain half width	1000	m
mountain type	Gaussian	-
horizontal domain size	40	km
vertical domain size	21	km
horizontal cell size	400	m
vertical cell size range	300	m
number of horizontal grid cells	100	-
number of vertical grid cells	70	-
initial Brunt-Väisälä frequency	0.01	1/s
bottom potential temperature	288	K
initial vertical velocity	10	m/s
time step	0.4	s
integration time	5000	s
x-divergence damping coefficient	0.05	-
z-divergence damping coefficient	0.05	-
Asselin coefficient	0.05	-
Rayleigh damping coefficient	0.02	-
Rayleigh damping layer thickness	9000	m
x-computational mixing coefficient	0.0005	-
z-computational mixing coefficient	0.0005	-

Table 6.1: Default parameters used for the simulation of a linear, non-hydrostatic flow past a Gaussian hill.

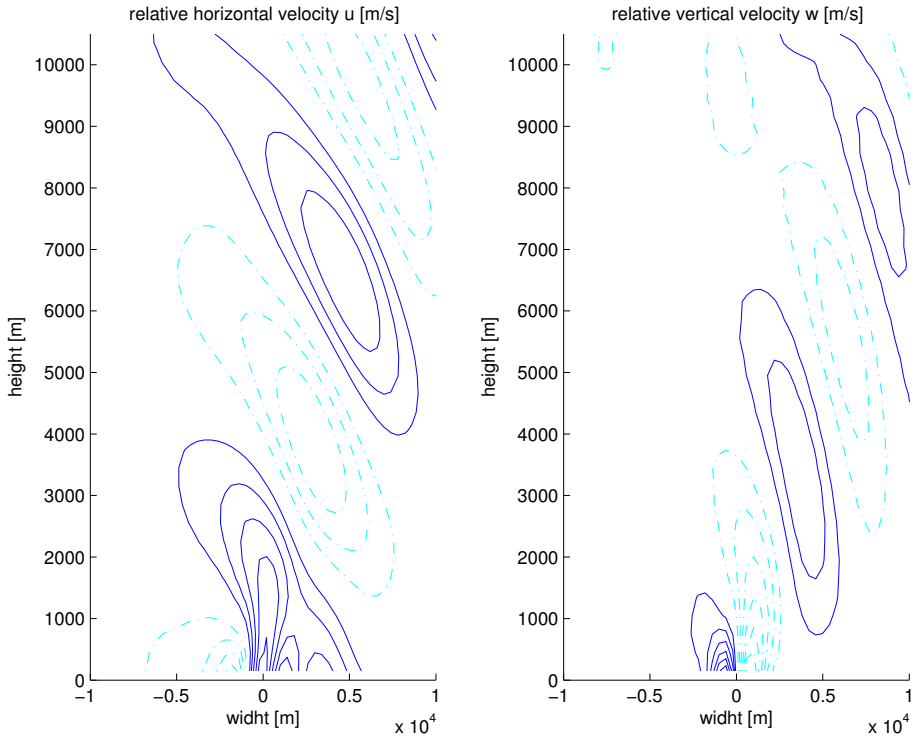

Figure 6.1: Horizontal and vertical velocity perturbations of a linear, non-hydrostatic flow. Calculation of background state at every time step. The contour interval is 0.001 m/s. Positive values are solid blue and negative values are dash-dotted cyan.

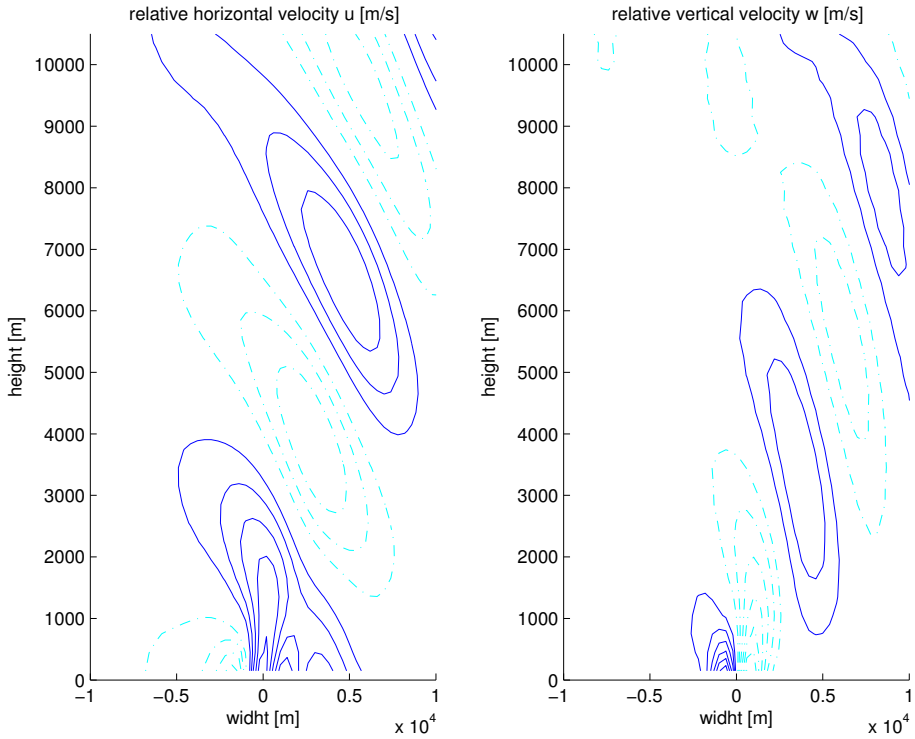

Figure 6.2: As figure 6.1, but background state is calculated every 10^{th} time step.

6.3 Linear, non-hydrostatic flow

With the default parameters from table 6.1 we get a non-dimensional mountain width $\hat{a}=1$ and a non-dimensional mountain height $\hat{h}=0.001$. Thus, we have a linear and non-hydrostatic flow. Figure 6.3 gives the analytic steady state solution in a domain of 20 km in the horizontal and 10500 m in the vertical. We can see a stationary wave with a downwind tilt typical for hydrostatic flows. Comparing the analytic solution to the simulation results shown in figure 6.1 and 6.2 we see, that our model is able to simulate the typical wave pattern in an appropriate way. The simulation results are very smooth and especially in the lower region the amplitude is close to the analytic solution. In the next subsections we want to show how the different improvements made during the course of the present work affect the results and we will compare the new cut-cell approach to the old terrain-following grid.

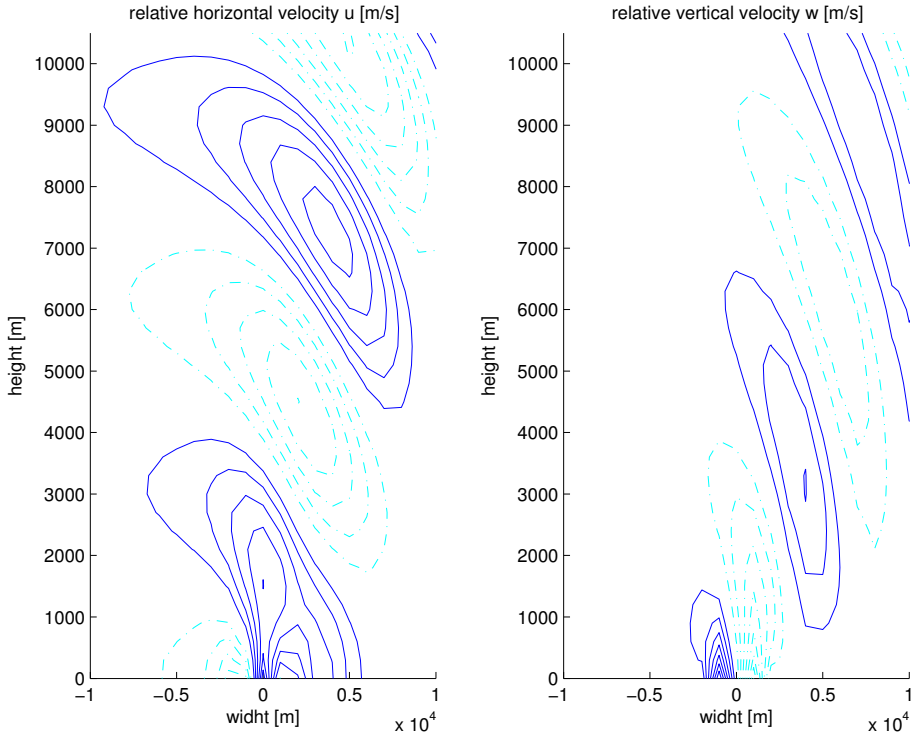

Figure 6.3: Analytic steady state solution of the linear, non-hydrostatic flow past a Gaussian hill. The mountain height and half width are 1 m and 1000 m respectively. Horizontal and vertical velocity perturbations. The contour interval is 0.001 m/s. Positive values are solid blue and negative values are dash-dotted cyan.

6.3.1 Grid comparison

The assumptions we made in the cut-cell approach in section 5.1 are feasible for simple linear test cases. Thus, we can use it for all simulations we consider here with exception of the hydrostatic balance 6.7 where we use the terrain-following grid. The simulation of the flow past a Schär hill, shown in section 6.5, is also done using the cut-cell approach. To prove that our cut-cell approach works we show here the results of a simulation using the same parameters as in table 6.1 but on a terrain-following grid. We decrease the time step from 0.4 to 0.35 seconds to get good, smooth results. When we compare figure 6.4 to figure 6.1 or 6.2 there is almost no visible difference. But where the 5000 seconds of integration time on the terrain-following grid uses 3900 seconds of total CPU time, the cut-cell approach only needs 3400 seconds. This improvement in performance results from some of the simplifications of the cut-cell approach mentioned in section 5.2 and of course from the bigger time step. Using a initial bottom potential temperature of 255 K instead of 288 K decreases the fastest propagating wave speed and allows us

to run the simulation on a terrain-following grid with a time step of 0.4 seconds. The totally used CPU time then reduces to about 3700 seconds which is still 300 seconds more than required for the cut-cell approach.

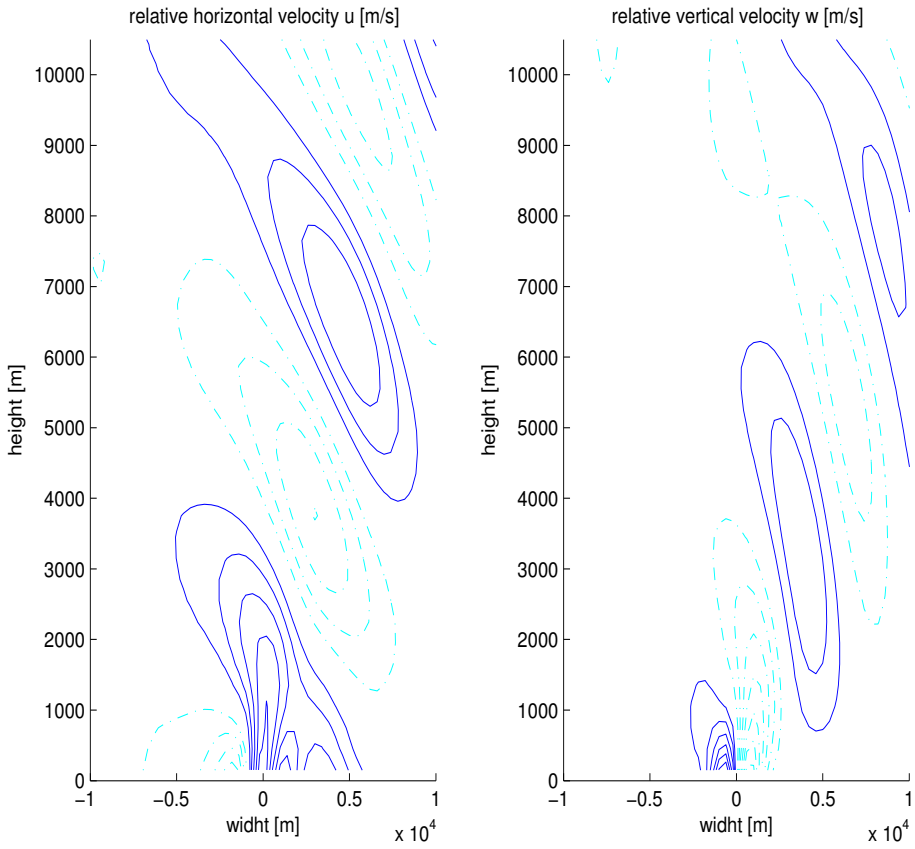

Figure 6.4: As figure 6.2, but for the terrain-following grid.

6.3.2 Impact of the corrected divergence damping

To show how the corrected derivatives in the divergence filter decrease the sensitivity to the choice of the damping parameters we consider a simulation without taking mixed second derivatives into account. The parameters $\alpha_x = 0.001$ and $\alpha_z = 0.0044$ used by Wunderlich in [6] are increased to the default values $\alpha_x = \alpha_z = 0.05$. Therefore, we expect the wind pattern to get smoother. But as can be seen in Figure 6.5 this increase almost cancels out the wave pattern when not using the mixed derivatives. This is also in agreement with the statements of Wunderlich in [6]. Nevertheless the remaining wave pattern is now very smooth.

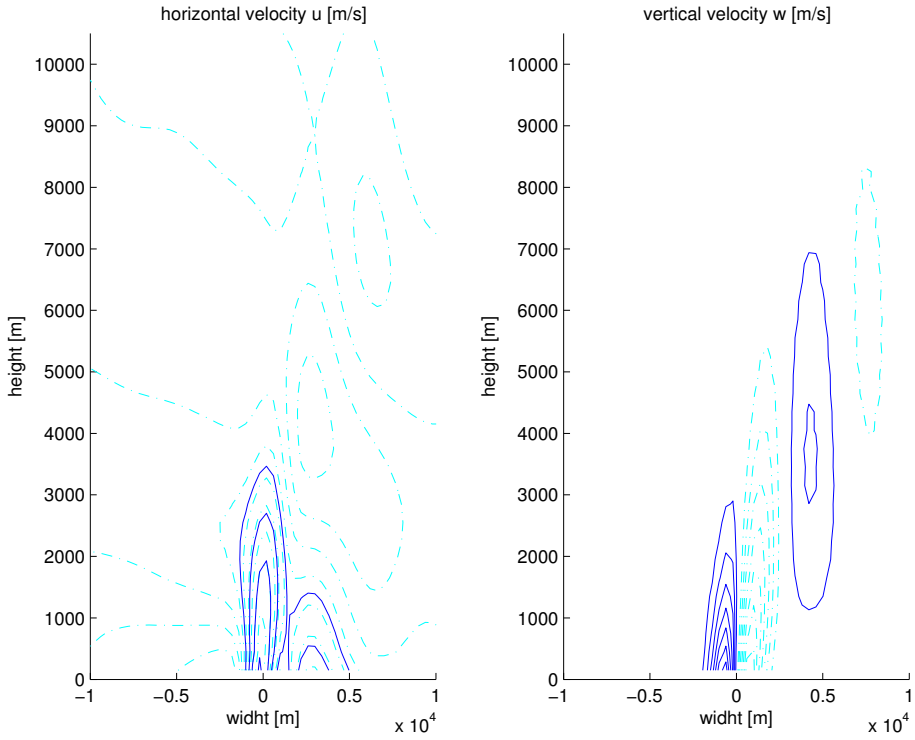

Figure 6.5: Horizontal and vertical velocity of a linear, non-hydrostatic flow. The contour interval is 0.001 m/s centered around zero for the vertical and around 10 m/s for the horizontal velocity. Bigger values are solid blue and smaller values are dash-dotted cyan. Mixed derivatives in the divergence damping term are neglected.

However, when we do not neglect the mixed derivatives, as we usually don't, we get results that are in good agreement with the analytic solution as for example in figure 6.2.

6.3.3 Influence of the Rayleigh sponge layer

The Rayleigh sponge layer at the top is introduced to eliminate wave reflections at the boundary and to absorb upward propagating wave disturbances. To illustrate its effects we made a simulation with the default parameters given in table 6.1 but turned off the sponge layer. The results are shown in figure 6.6. Without sponge layer some waves are reflected at the top. They start to propagate downwards and to disturb the solution. Whereas with the sponge layer the waves are filtered out as can be seen in figure 6.7.

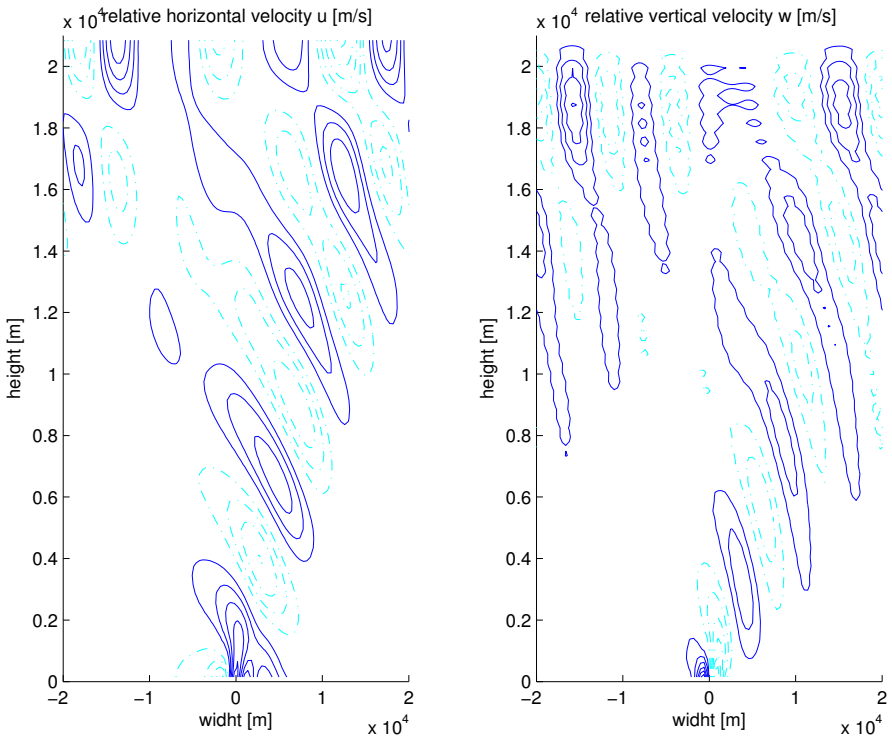

Figure 6.6: Horizontal and vertical velocity perturbations for a linear, non-hydrostatic flow in the whole computational domain. No Rayleigh sponge layer at the top. The contour interval is 0.001 m/s. Positive values are solid blue and negative values are dash-dotted cyan.

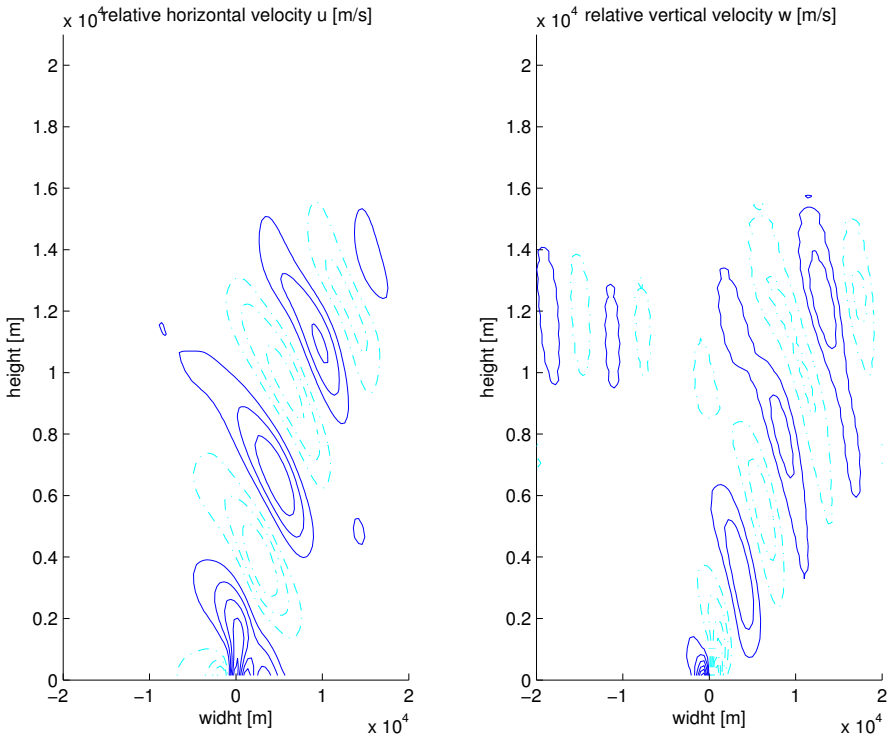

Figure 6.7: As figure 6.6, but with Rayleigh sponge layer at the top.

6.3.4 The effects of computational mixing

Computational mixing removes small scale noise of computational origin. Figure 6.8 shows the result of another simulation using the default parameters but this time without computational mixing. When we compare the results to those shown in figure 6.7 above we see, that the computational mixing prevents small scale disturbances from growing. For example the vertical velocity disturbances on the left side of the domain are an effect of the periodic lateral boundary conditions. They are visibly smaller when using computational mixing. Also the solution at the lower boundary is somewhat smoother. The main drawback of computational mixing is that it also prevents correct wave patterns from growing. Therefore the amplitude of the mountain waves are slightly closer to the analytic solution when we do not use computational mixing.

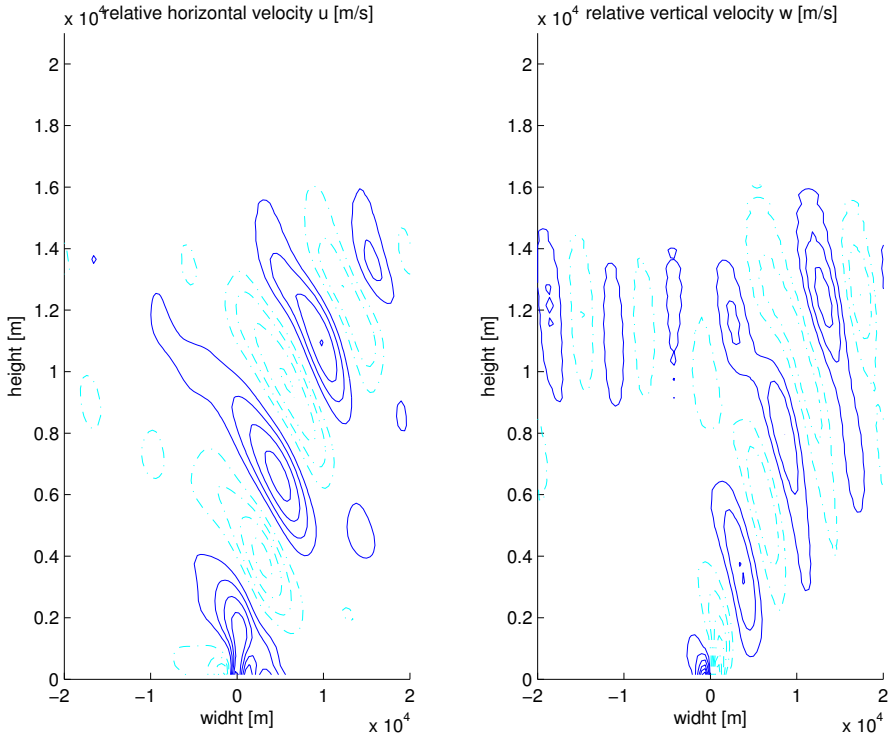

Figure 6.8: As figure 6.7, but without computational mixing.

6.4 Linear hydrostatic flow

Our second test case is the simulation of a linear, hydrostatic flow past a Gaussian hill. The Mountain height H is again 1 m but the mountain half width a is now increased to 5000 m. This yields the non dimensional mountain height and width $\hat{h}=0.001$ and $\hat{a}=5$ which characterize a linear and hydrostatic flow. We use again the cut-cell approach. The complete list of used parameters is given in table 6.2. Figure 6.9 shows the analytic solution in the steady state compared to the simulation results after 10000 seconds. Again, the simulation results are close to the analytic solution.

parameter	*value*	*unit*
time integration scheme	leapfrog	-
atmosphere	stratified	-
mountain height	1	m
mountain half width	5000	m
mountain type	Gaussian	-
horizontal domain size	50	km
vertical domain size	21	km
horizontal cell size	500	m
vertical cell size range	300	m
number of horizontal grid cells	100	-
number of vertical grid cells	70	-
initial Brunt-Väisälä frequency	0.01	1/s
bottom potential temperature	288	K
initial vertical velocity	10	m/s
time step	0.4	s
integration time	10000	s
x-divergence damping coefficient	0.05	-
z-divergence damping coefficient	0.05	-
Asselin coefficient	0.05	-
Rayleigh damping coefficient	0.02	-
Rayleigh damping layer thickness	9000	m
x-computational mixing coefficient	0.00005	-
z-computational mixing coefficient	0.00005	-

Table 6.2: Parameters used for the simulation of a linear, hydrostatic flow past a Gaussian hill.

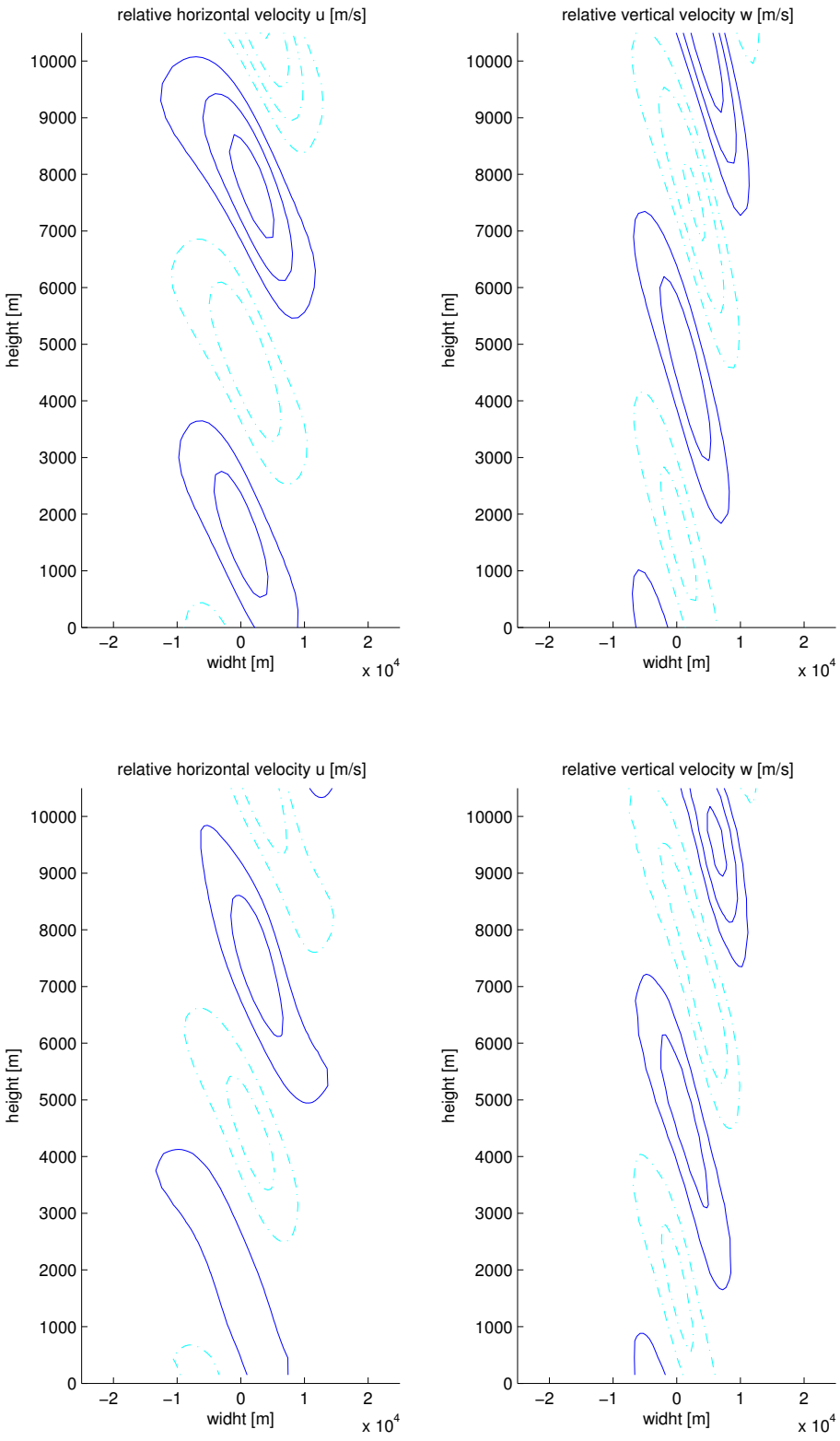

Figure 6.9: Horizontal and vertical velocity perturbations for a linear, hydrostatic flow past a Gaussian hill. Exact steady state solution (upper panel) compared to simulation results (lower panel) after 10000 seconds integration time. The contours intervals are 0.001 m/s for the vertical and 0.004 m/s for the horizontal velocity. Positive values are solid blue and negative values are dash-dotted cyan.

6.5 Flow over a Schär hill

A third test case is the flow over an idealized topography, the so called “Schär-Hill”. The topography is given by

$$h(x) = H \exp\left(-(x/a)^2\right) \cos^2\left(\frac{x\pi}{\lambda}\right) \tag{6.6}$$

with $\lambda = 4$ km, $H = 250$ m and $a = 5$ km. All other parameters we use are listed in table 6.3. Figure 6.10 shows the comparison of our simulation after 10000 seconds integration time, again computed using the cut-cell approach, with the analytic steady state solution. The results we get are comparable to those given in literature using approximatively the same grid spacing and a well balanced method, e.g. Botta et al. in [5].

parameter	*value*	*unit*
time integration scheme	leapfrog	-
atmosphere	stratified	-
mountain height	250	m
mountain half width	5000	m
mountain type	Schär	-
mountain wavelength	4000	m
horizontal domain size	200	km
vertical domain size	19.2	km
horizontal cell size	500	m
vertical cell size range	300	m
number of horizontal grid cells	400	-
number of vertical grid cells	64	-
initial Brunt-Väisälä frequency	0.01	1/s
bottom potential temperature	273.16	K
initial vertical velocity	10	m/s
time step	0.4	s
integration time	10000	s
x-divergence damping coefficient	0.0017	-
z-divergence damping coefficient	0.0017	-
Asselin coefficient	0.05	-
Rayleigh damping coefficient	0.05	-
Rayleigh damping layer thickness	5000	m
x-computational mixing coefficient	0.0015	-
z-computational mixing coefficient	0.0015	-

Table 6.3: Parameters used for the simulation of a linear, non-hydrostatic flow past a Schär hill.

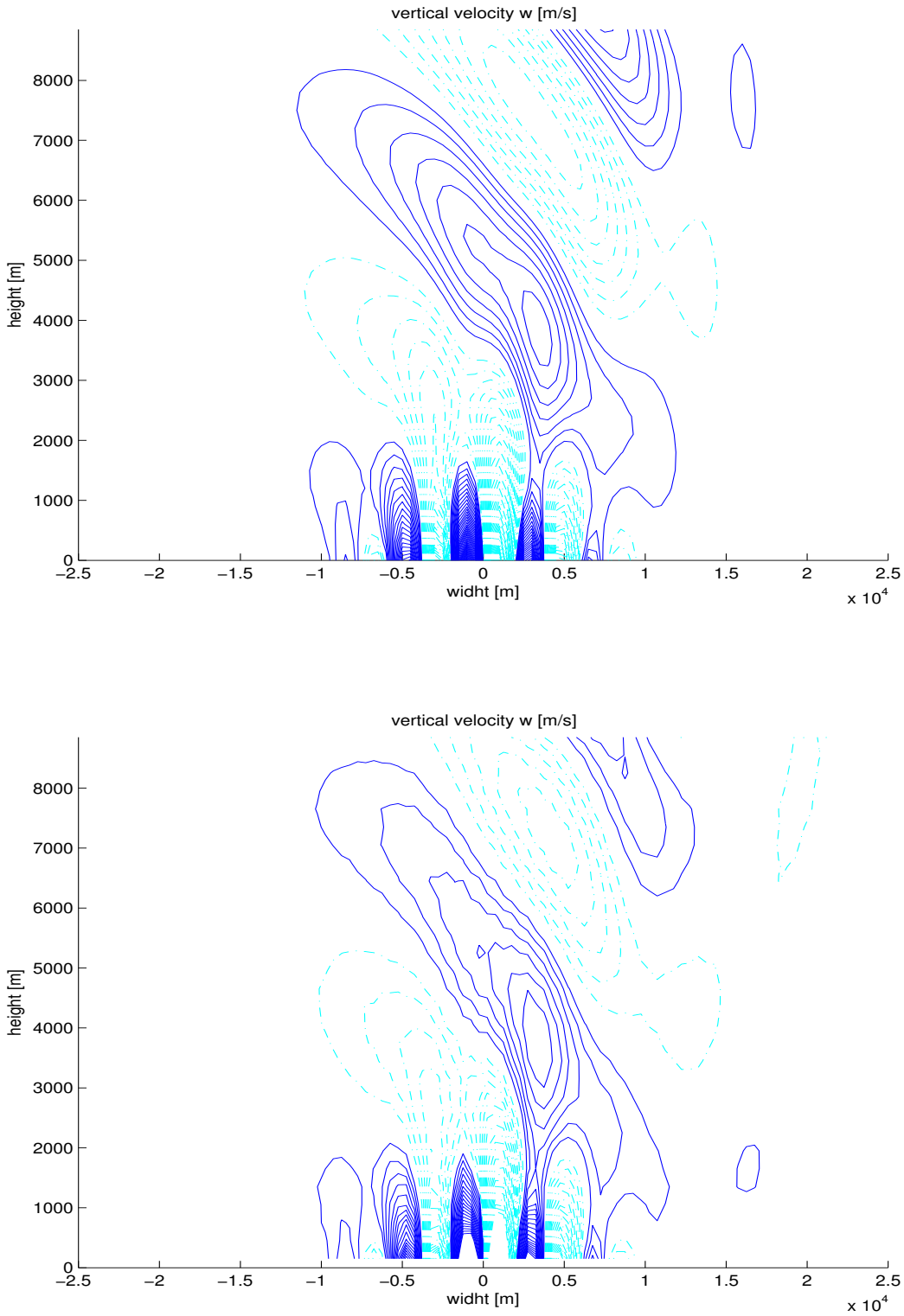

Figure 6.10: Vertical velocity of a linear, non-hydrostatic flow past a Schär hill. Exact solution in steady state (upper panel) compared to simulation after 10000 seconds (lower panel). The contour interval is 0.05 m/s centered around zero. Positive values are solid blue and negative values are dash-dotted cyan.

6.6 Thermal bubble

In [11] Straka et al. present the numerical solution of a nonlinear density current in an otherwise homogeneous and isentropic atmosphere. This density current serves us as another test case. It is initiated as a cold blob of air that subsequently descends to the ground. The elliptic temperature perturbation is specified by

$$\Delta T = \begin{cases} 0.0\,K & \text{if } L > 1.0 \\ -15.0\,K \cdot \frac{1}{2}(\cos(\pi L) + 1.0) & \text{if } L \leqslant 1.0 \end{cases} \tag{6.7}$$

where $x_c = 0.0\,\text{km}$, $x_r = 4.0\,\text{km}$, $z_c = 3.0\,\text{km}$ and $z_r = 2.0\,\text{km}$. In the rest of the domain the potential temperature is 300 K. The perturbation in the potential temperature $\Delta\theta$ resulting from (6.7) can be calculated using the relation $T = \pi\theta$ where π is the exner function $\pi = (pp_0^{-1})^{(\gamma-1)/\gamma}$ and $\gamma = 1.4$ is the adiabatic exponent. Table 6.4 gives the list of the parameters we use for our simulation. Figure 6.11 shows the solution of the thermal bubble simulation at times $t = 0, 300, 600$ and 900 seconds. According to [11] we use a diffusion coefficient of 75 m^2s^{-1}. Our simulation shows the typical development of three so called Kelvin-Helmholtz shear instability rotors during the period from 0 to 900 seconds described in [11].

parameter	*value*	*unit*
time integration scheme	leapfrog	-
atmosphere	isentropic	-
horizontal domain size	30	km
vertical domain size	5	km
horizontal cell size	100	m
vertical cell size	100	m
number of horizontal grid cells	300	-
number of vertical grid cells	50	-
undisturbed potential temperature	300	K
initial vertical velocity	0	m/s
time step	0.125	s
integration time	900	s
x-divergence damping coefficient	0.02	-
z-divergence damping coefficient	0.02	-
Asselin coefficient	0.1	-
diffusion constant	75	m²/s
x-computational mixing coefficient	0.0005	-
z-computational mixing coefficient	0.0005	-
number of steps between background state calculation	5	

Table 6.4: List of used parameters for the thermal bubble simulation.

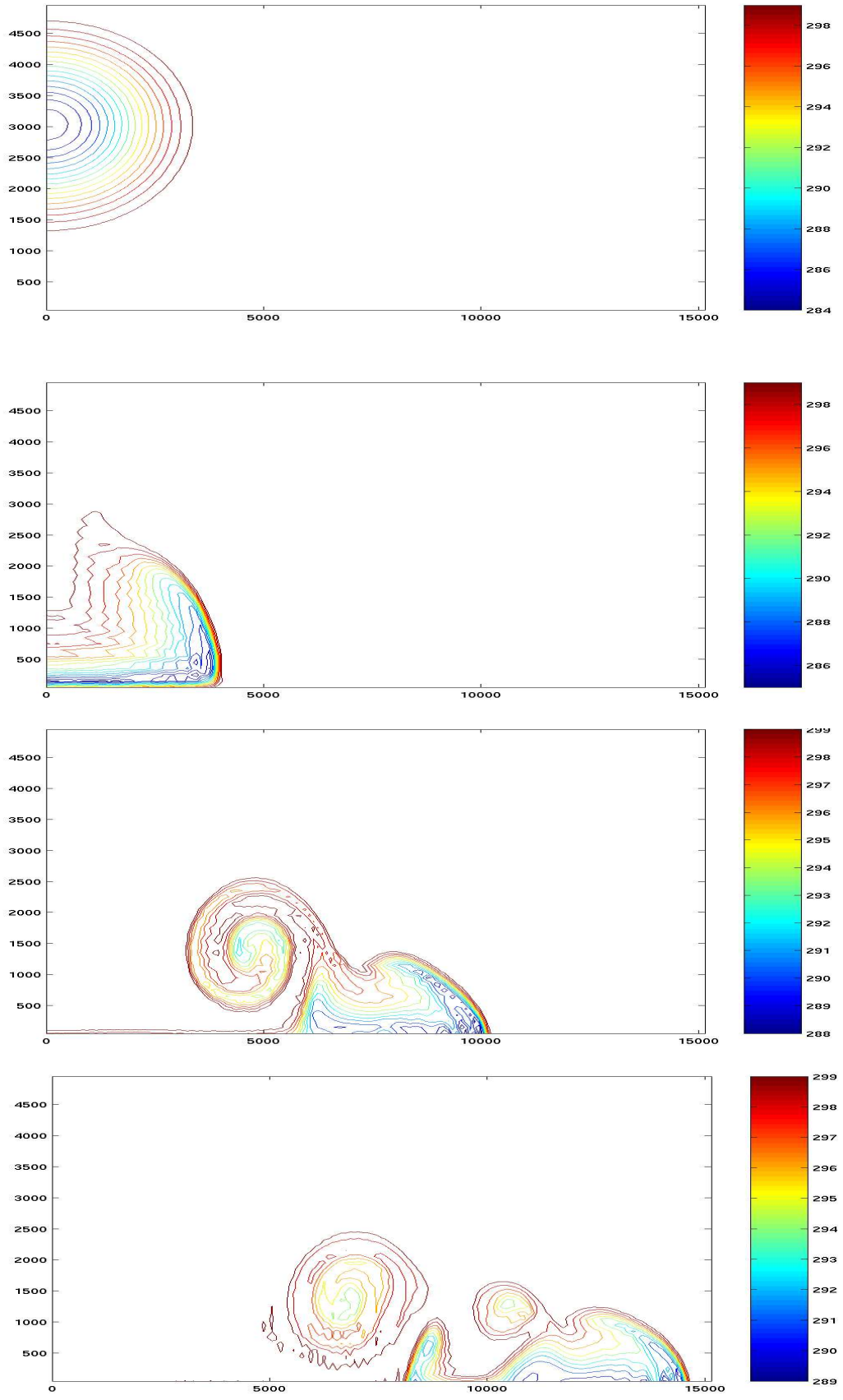

Figure 6.11: Simulation of a negative buoyant blob of cold air in an otherwise isentropic atmosphere. Potential temperature at 0, 300, 600 and 900 seconds (from top to bottom). Contours are downwards from 299K with an interval of 1K. Only the right half of the computational domain is shown.

6.7 Atmosphere at rest

We now want to show that our model is able to simulate an atmosphere at rest over steeper topography. The parameters used are listed in table 6.5. We compare the evolution of the maximum magnitude of the horizontal velocity to the values obtained by Wunderlich in [6]. The grid we use is hybrid with terrain-following coordinates below and uniform coordinates above 12000 m. That's also where our Rayleigh sponge layer starts. We make a time integration of 25000 seconds which is about 7 hours. The numbers we get are slightly smaller than those Wunderlich gets in [6], compare figure 6.12 and 6.13. The main differences of the two models are the improvements we made, the number of time steps between the evaluation of the local hydrostatic background state and the hybrid grid we use.

parameter	*value*	*unit*
time integration scheme	leapfrog	-
atmosphere	stratified	-
mountain height	1500	m
mountain half width	5000	m
mountain type	Gaussian	-
horizontal domain size	40	km
vertical domain size	21	km
horizontal cell size	800	m
vertical cell size range	300	m
number of horizontal grid cells	50	-
number of vertical grid cells	70	-
initial Brunt-Väisälä frequency	0.01	1/s
bottom potential temperature	288	K
initial vertical velocity		m/s
time step	0.4	s
integration time	25000	s
x-divergence damping coefficient	0.05	-
z-divergence damping coefficient	0.05	-
Asselin coefficient	0.05	-
Rayleigh damping coefficient	0.02	-
Rayleigh damping layer thickness	9000	m
x-computational mixing coefficient	0.005	-
z-computational mixing coefficient	0.005	-

Table 6.5: List of parameters used for the simulation of an atmosphere at rest above steep topography.

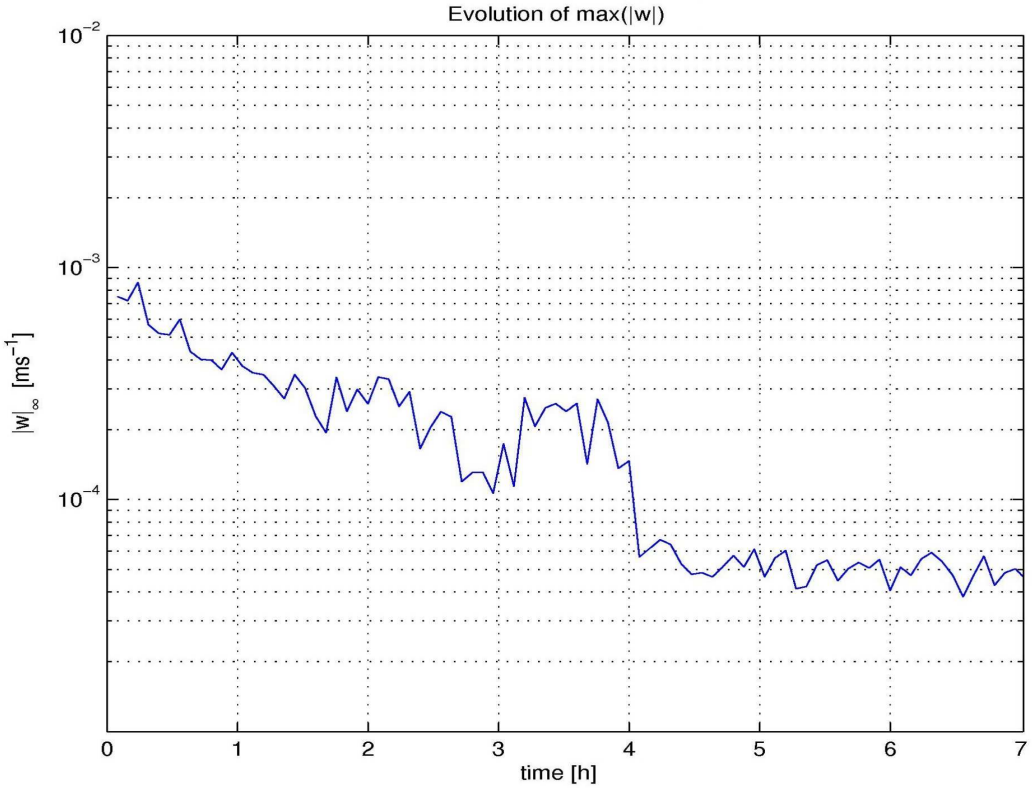

Figure 6.12: Time evolution of maximal vertical velocity above steep topography. Simulation of an atmosphere at rest with unimproved model on terrain-following coordinates.

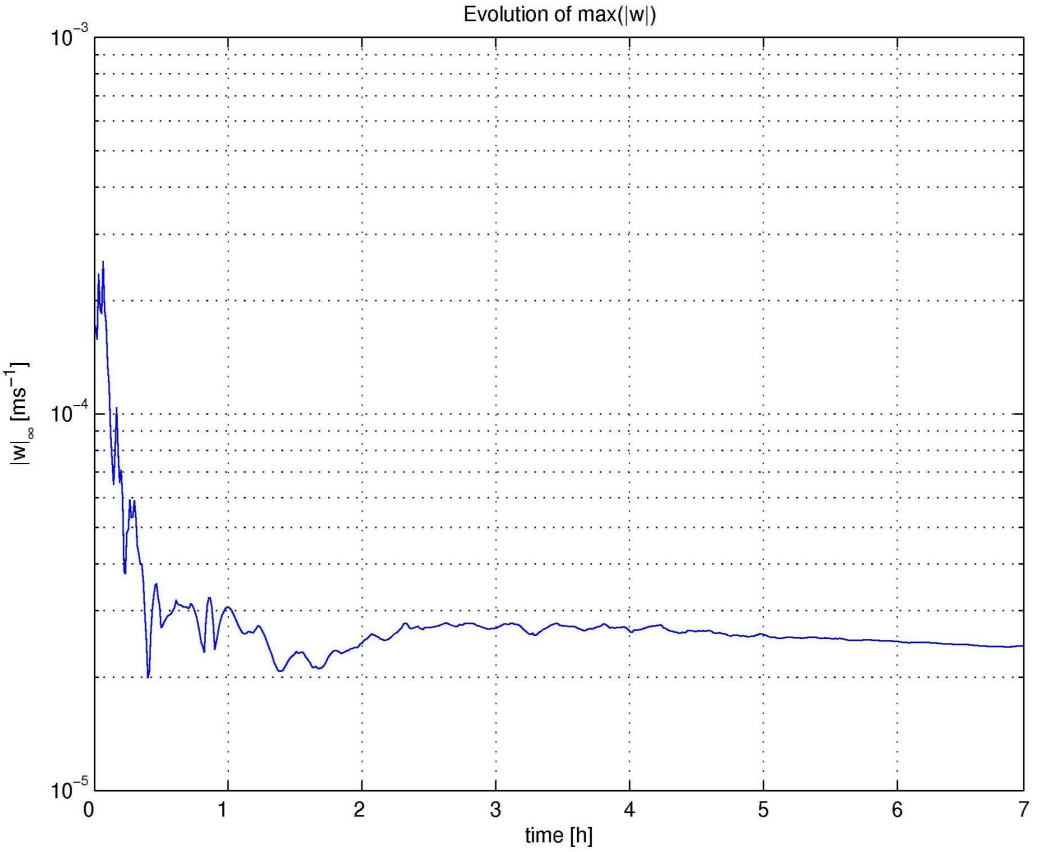

Figure 6.13: As figure 6.12, but using all model improvements and hybrid coordinates consisting of terrain-following coordinates below and uniform above 12000 m.

7 Conclusion and outlook

We have made several improvements to the model developed by Wunderlich in [6], based on the ideas of Botta et al. in [5]. We implemented computational and physical mixing, a Rayleigh sponge layer at the top that prevents wave reflections at the boundary and corrected the divergence damping term so that the model has become less sensitive to the choice of the damping parameters. Performance has been increased by over 90%. This results mostly from the fact that the local hydrostatic background state is now calculated every 10^{th} time step instead of every time step. We looked at results from a cut-cell approach using uniform, terrain intersecting, Cartesian coordinates instead of terrain-following coordinates. To avoid stability problems due to small cells we have used a cut-cell approach that handles cut cells like whole uniform cells. The results we get are close to those from the terrain-following coordinate approach.

Several test cases have shown that our model yields results close to the analytic solutions in the case of hydrostatic and non-hydrostatic linear flows over different topographies. The simulation of a nonlinear density current in form of a descending cold blob of air yields good results too. Furthermore, the simulation of an atmosphere at rest above steep topography produces only very small deviations from absolute rest even after seven hours of integration time.

In summary, the model has been advanced by several improvements, is now more complete and efficient than before and yields good results for several test cases. Furthermore, a cut-cell approach was implemented that makes the use of a uniform Cartesian grid possible.

However, the cut-cell approach we made in this work contains some drawbacks and simplifications that should be further investigated. First, our approach is not suitable for surfaces where there is a big change in steepness between adjacent grid cells. For idealized, smooth terrain this problem can be partly circumvented by a higher grid resolution. But on real surfaces including buildings etc. we will always face abrupt slope changes. Nevertheless the approach looks promising. One could try to replace the calculation of the reflected cells, needed to update the cut cells, by a more sophisticated method than we used. An idea would be to reflect cells locally when they are needed to update a certain cell. Secondly, we do not calculate the fluxes at the border between atmosphere and solid ground but at cell interfaces close to it. Therefore the scheme is not completely conservative and some investigations on mass production inside the domain should be done to show how big the error is.

7.1 Acknowledgments

First of all I'd like to thank Dr. Jürg Schmidli who advised me during this diploma thesis. He was always willing to help me when I faced a new problem and supported me with a lot of background information on the subject. I want to thank Prof. Schär to make this diploma thesis at the IAC ETH possible and his confidence in students of “Rechnergestützte Wissenschaften”. Last but not least I'd like to thank my family and friends for their never ending support during the last years.

References

[1] Klemp, J.B., Skamarock, W.C., Fuhrer, O. 2003: *Numerical consistency of metric terms in terrain-following coordinates,* Mon. Wea. Rev., 131 1229-1239

[2] Zängel, G., Gantner, L., Hartjenstein, G., Noppel, H. 2004: *Numerical errors above steep topography: A model intercomparison*, Meteorlogische Zeitschrift Vol. 13 No. 2 69-76

[3] Bonaventura, L. 2000: *A Semi-implicit Semi-Lagrangian Scheme Using the height Coordinate for a Nonhydrostatic and Fully Elastic Model of Atmospheric Flows*, Jour. of Comp. Phys. 158, 186-213

[4] Steppeler, J., Hess, R., Schättler, U., Bonaventura, L. 2003: *Review of numerical methods for nonhydrostatic prediction models,* Meteorol. Atmos. Phys. 82, 287-301

[5] Botta, N., Klein, R., Langenberger, S., Lützenkirch, S. 2003: *Well balanced finite volume methods for nearly hydrostatic flows,* Jour. of Comp. Phys. 196, 539-565

[6] Wunderlich, S. 2004: *A 2D Well Balanced Finite Volume Non-hydrostatic Atmospheric Model. Hydrostatic Balance and Simulation of Mountain Waves,* Diploma thesis IAC ETH Zürich

[7] Müller, S.,2004: *Hydrostatisches Gleichgewicht und akustische Wellen in nicht-hydrostatischen Atmospären-Modellen*. Diplomarbeit, IAC ETH Zürich

[8] Holton, J.R., 1992: *An Introduction to Dynamic Meteorology,* Academic Press, Third Edition

[9] *ARPS (Advanced Regional Prediction) version 4.0 User Guide 1995,* Center for Analysis and Prediction of Storms, University of Oklahoma

[10] Forrer, H. 1996: *Second Order Accurate Boundary Treatment for Cartesian Grid Methods,* Research Report No 96-13, Seminar für Angewandte Mathematik ETH Zürich

[11] Straka, J.M., Wilhelmson, R.B., Wicker, L.J., Anderson, J.R., Droegemeier, K.K. 1993: *Numerical solutions of a non-linear density current: a benchmark solution and comparisons,* Int. Jour. for numerical methods in fluids, Vol 17, 1-22