MIMICA V5, User guide

Julien SAVRE
Meteorological Institute, Ludwig-Maximilians-Universität, Munich
julien.savre@lmu.de

Matthias Brakebusch
Dept. of Environmental Science and Analytical Chemistry, Stockholm University, Stockholm
matthias.brakebusch@aces.su.se

06.03.2019

Contents

1 Introduction: how to compile and run MIMICA

The MIMICA model has been originally designed as an extension of the CRM-MIT model developed by Chien Wang and Julius Chang [**?**]. Since then, the model has been substantially modified and upgraded, and most of the original source code has been rewritten using modern fortran 90 and MPI standards. Despite the efforts put into the standardization and optimization of the model's source code, it should be noted that certain parts of the model still remain from the original CRM (aerosol and chemistry) and may therefore appear outdated.

2 Description of the MIMICA model

The description proposed in this section focuses on a "default" configuration of the MIMICA model. It should be reminded that MIMICA possesses many more capabilities which are not detailed below for the sake of clarity. Some of these "hidden" features are however mentioned when appropriate, and more information can also be found in the description of the *start* and *cm.nml* files given in section 3: "MIMICA inputs".

2.1 Governing equations

MIMICA solves the basic conservation equations for mass (continuity equation), momentum, potential temperature and total water content in 2 or 3 dimensions. Depending of the level of complexity and physical parameterizations requested, a set of additional conservation equations for microphysical quantities (typically the mass and number concentrations of certain hydrometeor classes) as well as passive tracers can also be solved. Two different dynamical cores are available in MIMICA: **an anelastic core**, in which the density is assumed to depend on altitude only so that the continuity equation becomes a diagnostic equation constraining the velocity vector to be divergence free, and **a compressible core**, in which no approximation for the density is made and the continuity equation must be solved explicitly. All the equations presented hereafter are written according to the anelastic approxiation as this is the default dynamical core used in MIMICA (*setenv ANELASTIC TRUE* in *start*). More details about the compressible core are given in section **??**.

In its more general form, the momentum equation can be written as follows:

$$\frac{\partial \rho_0 u}{\partial t} + \nabla \cdot (\rho_0 \mathbf{u} u) = -\rho_0 \frac{\partial}{\partial x}\left(\frac{p'}{\rho_0}\right) + f_0 v + \nabla \cdot (\rho_0 \tau_u) \quad (1)$$

$$\frac{\partial \rho_0 v}{\partial t} + \nabla \cdot (\rho_0 \mathbf{u} v) = -\rho_0 \frac{\partial}{\partial y}\left(\frac{p'}{\rho_0}\right) - f_0 u + \nabla \cdot (\rho_0 \tau_v) \quad (2)$$

$$\frac{\partial \rho_0 w}{\partial t} + \nabla \cdot (\rho_0 \mathbf{u} w) = b - \rho_0 \frac{\partial}{\partial z}\left(\frac{p'}{\rho_0}\right) + \nabla \cdot (\rho_0 \tau_w) \quad (3)$$

$\mathbf{u}$ is the velocity vector (u, v, w), b is the vertical buoyancy flux, τ_u, τ_v, τ_w are the contributions from subgrid scale turbulent diffusion to u, v and w respectively (defined in section 2.5.1), f_0 is the Coriolis parameter (f-plane approximation), ρ_0 is the reference anelastic density (independent of time and varies only with altitude), and $p' = p - p_0$ is the perturbation pressure defined with respect to the base state hydrostatic pressure. The buoyancy b is defined based on the classical definition of the virtual potential temperature:

$$b = \rho_0 g \left[\frac{\theta\,(1 + \epsilon q_v - q_l - q_i)}{\theta_0\,(1 + \epsilon q_{v0})} - 1\right] \quad (4)$$

with θ being the potential temperature, q_v is the water vapor mixing ratio, q_l and q_i are the liquid and ice mixing ratios respectively, g is the gravitational acceleration (constant, set to 9.81 $m.s^{-2}$) and $\epsilon \approx 0.602$. θ_0 and q_{v0} are here defined as the base state potential temperature and vapor mixing ratio (independent of time, functions of altitude only). The different terms appearing in the momentum budget equations can be turned on or off individually using various options in *cm.nml* and *start*.

The potential temperature, total water mixing ratio and passive tracer equations read:

$$\frac{\partial \rho_0 \theta}{\partial t} + \nabla \cdot (\rho_0 \mathbf{u} \theta) = \mathcal{Q}_{rad} + \mathcal{Q}_{micro} + \nabla \cdot (\rho_0 \tau_\theta) \quad (5)$$

$$\frac{\partial \rho_0 q_t}{\partial t} + \nabla \cdot (\rho_0 \mathbf{u} q_t) = \nabla \cdot (\rho_0 \tau_{qt}) + \mathcal{Q}_{prec} \quad (6)$$

$$\frac{\partial \rho_0 \varphi}{\partial t} + \nabla \cdot (\rho_0 \mathbf{u} \varphi) = \nabla \cdot (\rho_0 \tau_\varphi) \quad (7)$$

with $\mathcal{Q}_{rad}$ and $\mathcal{Q}_{micro}$ being source terms representing radiative and latent heating and cooling respectively, $\mathcal{Q}_{prec}$ is a sedimentation term, q_t and φ are the total water mixing ratio and passive tracer, while τ_θ, τ_{qt}, τ_φ are the contributions from subgrid scale turbulent diffusion. Note that the potential temperature is defined as:

$$\theta = T \left(\frac{p}{p_{00}}\right)^{-c_{pa}/R_a}, \quad (8)$$

with T the absolute temperature, $p_{00} = 1000\ hPa$, $cp_a = 1004\ J.K^{-1}.kg^{-1}$ and $R_a = 288\ J.K^{-1}.kg^{-1}$ the dry air specific heat capacity at constant pressure and specific ideal gas constant for dry air respectively. As an alternative

to the potential temperature equation, MIMICA an also operate with the frozen moist static energy (MSE) as the conserved energy quantity (available by setting *setenv ISENTROPIC FALSE* in *start*). The frozen MSE is often used to simulate tropical convection as it is almost exactly conserved upon adibatic processes and phase changes, and is directly related to the enthalpy (and therefore to the first law of thermodynamics).

Finally, the system of equations is closed using the ideal gas law for moist air to relate all thermodynamic properties:

$$p_0 = \rho_0 RT \tag{9}$$

with $R = (1 - q_t)\, R_a + q_v R_v$ the ideal gas constant for moist air ($R_v = 461.5\ J.K^{-1}.kg^{-1}$ being the ideal gas constant for water vapor). By default, the vapor contribution to R is neglected so that $R = R_a = 288\ J.K^{-1}.kg^{-1}$. This can however be changed by setting *cst_cp = 0* in *cm.nml*.

2.2 The anelastic and compressible cores

2.2.1 The anelastic solver

In its default configuration, MIMICA employs the anelastic approximation in which the density is assumed to vary only with altitude and is therefore time independent. Using such a time independent density enables us to filter out acoustic waves to improve the model's stability and overall performances. Solving the momentum equations under the anelastic approximation however requires a trick described below.

First, we need to write the anelastic continuity equation:

$$\frac{\partial \rho_0 u}{\partial x} + \frac{\partial \rho_0 v}{\partial y} + \frac{\partial \rho_0 w}{\partial z} = 0. \tag{10}$$

This diagnostic equation constraining the velocity vector to be divergence free is then used to diagnose the pressure gradient terms appearing in equations 1-3, these terms being the source of the propagation of acoustic waves. The solution procedure essentially follows three steps:

1. The momentum equations 1-3 are first solved by omitting the pressure gradient terms. At this stage, the updated velocity vector $\mathbf{u}^*$ does not satisfy the continuity equation 10.

2. Next, using a simple Euler forward method, we introduce the pressure gradient terms as a correction to $\mathbf{u}^*$ yielding the new updated vector $\mathbf{u}^{n+1}$ at the end of the time step:

$$\rho_0 \mathbf{u}^{n+1} = \rho_0 \mathbf{u}^* - \Delta t \rho_0 \nabla \left(\frac{p'}{\rho_0} \right). \tag{11}$$

   Now taking the divergence of the above equation:

$$0 = \nabla \cdot (\rho_0 \mathbf{u}^*) - \Delta t \nabla \cdot \left(\rho_0 \nabla \frac{p'}{\rho_0} \right), \tag{12}$$

   where we have used the fact that $\nabla \cdot (\rho_0 \mathbf{u}^*)$ must be 0 by continuity. The first term on the right hand side, $\nabla \cdot (\rho_0 \mathbf{u}^*) \neq 0$, can be easily computed, leaving us with an elliptic equation for p'.

3. After solving the equation for p' (see details below), its gradients can be computed, and $\mathbf{u}^{n+1}$ is finally obtained from equation 11.

The whole procedure can be repeated a certain number of times (this is specified by *nsubp* in *cm.nml* which is set to 1 by default) in order to improve the accuracy of the method.

An efficient solver based on Fast Fourier Transforms (FFT) is implemented in MIMICA to solve the elliptic pressure perturbation equation. We first rewrite equation 12 as follows:

$$\nabla^2 \frac{p'}{\rho_0} + \frac{1}{\rho_0} \frac{\partial \rho_0}{\partial z} \frac{\partial}{\partial z} \frac{p'}{\rho_0} = \mathcal{F}, \tag{13}$$

with $\mathcal{F}$ proportional to $\nabla \cdot (\rho_0 \mathbf{u}^*)$, and apply a Fourier transform in the horizontal plane to give:

$$-\left(k^2 + l^2\right) \widehat{\left(\frac{p'}{\rho_0} \right)} + \frac{\partial^2}{\partial z^2} \widehat{\left(\frac{p'}{\rho_0} \right)} + \frac{1}{\rho_0} \frac{\partial \rho_0}{\partial z} \frac{\partial}{\partial z} \widehat{\left(\frac{p'}{\rho_0} \right)} = \widehat{\mathcal{F}}. \tag{14}$$

Using centered finite differences to approximate the first and second order derivatives of the transformed perturbation pressure yields a tridiagonal system for $(\widehat{p'/\rho_0})$ which can be easily and efficiently inverted. Once this is done, the solution $(\widehat{p'/\rho_0})$ can be transformed back into the physical space to find p'/ρ_0 at each grid point. We can then calculate the gradient of this quantity along each direction and introduce it in equation 11 to complete the pressure correction procedure.

The procedure presented above can only estimate the pressure perturbation to within a constant (unknown). This is not a problem when updating the velocity vector as the gradients of the pressure perturbation are not affected by the exact value of the constant. By default, the domain averaged value of p' is subtracted to the pressure perturbation solved by the FFT solver. It is also possible to correct p' more realistically by requiring that the total energy inside the domain is conserved (option *p_mcons* in *cm.nml*).

Although very accurate and efficient, the anelastic solver also comes with its limitations. It must be noted in particular that the present FFT solver assumed by default that the domain's lateral boundaries are periodic (extending the solver to open boundaries is possible but has not been done yet). Furthermore, although the FFT solver is parallelized, allowing for an efficient solution procedure even on large parallel domains, it currently assumes implicitly that the domain possesses an equal number of grid cells along the X and Y directions.

2.2.2 The compressible solver

By setting *setenv ANELASTIC FALSE* in *start*, MIMICA solves for the fully compressible equation of motions, which implies that the density is allowed to vary both in time and space. As mentioned previously, solving the fully compressible system of equations requires much smaller time steps to satisfy the acoustic CFL stability criterion, which often translates into a significant loss of model performances. The compressible solver however has its advantages: there is no *a priori* assumption on the density and it is more flexible in the sense that using periodic or open lateral boundaries does not need any extra coding effort and there is no requirement that NX and NY be equal.

In short, the following continuity equation now needs to be solved:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = \mathcal{Q}_{prec}. \tag{15}$$

The presence of a term related to precipitation ($\mathcal{Q}_{prec}$) can be understood by noting that $q_a + q_v + q_l + q_i = 0$ (q_a being the dry air mixing ratio), so that the continuity equation must be retrieved when summing the governing equations for q_a, q_v, q_l and q_i. The conservation of water mass implies that all microphysical sources and sinks add up to 0 except for the sedimentation terms. All the equations introduced previously still need to be solved, except that ρ_0 must now be replaced by ρ. In addition, note that divergence damping is also applied (an extra term in the momentum equations) in order to improve the solver's stability (the strength of this damping is controlled by *adiv* in *cm.nml* whose value is set by default to 0.2).

To solve the new system of coupled equations, no extra step must be taken. Because the density and potential temperature are solved and known explicitly within each grid cell, the pressure can be found using the ideal gas law. It is then enough to take the gradient of the diagnosed pressure and introduce it directly in the momentum equations. Because this procedure may be unstable at large time steps, this is done using a smaller acoustic time step calculated to satisfy the acoustic CFL criterion and is defined as a fraction of the main MIMICA time step ($\Delta t_{acoustic} = \Delta t_{main}/N_{step}$, where N_{step} is the number of iterations required to solve the compressible system of equations).

2.3 Numerical methods

2.3.1 Numerical grid and boundary conditions

The mesh used by the model is defined in a cartesian coordinate, in 2 or 3 dimensions. The cell size is fixed constant in the horizontal plane (*dx* and *dy* in *cm.nml*) but some refinement and stretching can be applied in the vertical direction by setting *setenv FINE TRUE* in *start*. Predefined vertical grids are available for specific test cases and can be used at will. Alternatively, the vertical grid can be provided as an external input file named *grid.dat* containing a unique column specifying each individual grid level. The number of cells in all 3 directions must be set by the user in *start*. The domain length in X and Y is then obtained by multiplying the number of grid cells in each direction by the cell sizes *dx* and *dy*. The model top height depends on the non-constant grid spacing in the vertical.

Cartesian grids are very convenient to discretize fluid equations using finite differences or finite volume methods. In MIMICA, a staggered grid system is employed, the C-Arakawa grid, defined as follows:

- the pressure, and all the other scalars, are defined at the center of each grid cell

- the three velocity components are shifted by half a cell and are therefore defined at face centers: $u_{ijk} = u\left(x+\frac{\Delta x}{2}, y, z\right)$, $v_{ijk} = v\left(x, y+\frac{\Delta y}{2}, z\right)$, $w_{ijk} = w\left(x, y, z+\frac{\Delta z}{2}\right)$

In addition, the local velocity (respectively pressure) gradients are defined at the point where pressure (respectively velocity) is defined.

In practice, due to grid staggering, two different sets of vertical grid spacings have to be defined when vertical refinement is used: one corresponding to the spacing between two vertical velocity points, the other defined as the spacing between two scalar points. These two quantities are refered to as Δz_w and Δz_p respectively.

Boundary conditions in MIMICA are defined by default as periodic in the horizontal plane, and solid (i.e. the vertical velocity vanishes exactly at the boundary) in the vertical. Because gravity waves are allowed to propagate in the domain and are usually reflected by the top boundary, it is often recommended to apply a sponge layer near the domain top, whose role is to nudge the simulated atmosphere in this region to the initial conditions with a time scale *tdamp* defined in *cm.nml* (the altitude at which the sponge layer is applied is set through *zdamp*).

Open boundary conditions have also been implemented for the lateral boundaries, but this option is not recommended in the current version of MIMICA. It is however possible to apply pseudo-open lateral boundaries by using a large horizonal domain with the default periodic conditions, but by prescribing thick horizontal sponge layers as described in section **??**. Similarly, it is possible to use periodic conditions in the vertical, although this only applies to specific idealized templates. Overall, it is recommended not to modify the default specification of boundary conditions accessible through parameters *bcl(1-6)* in *cm.nml*.

2.3.2 Momentum advection

The momentum advection scheme must provide sufficient accuracy and stability when coupled with the integration scheme. With no additional constraint, this leads to a very limited choice, mainly based on the easiness of implementation. We chose here to discretize momentum advection using high-order central finite differences with hyperviscosity to preserve numerical stability. For clarity sake, the advective fluxes are recast into a conservative form (here taking the advection of the u momentum along the Y direction):

$$\frac{\partial \rho_0 v u}{\partial y} \equiv \frac{F_{i,j+1/2} - F_{i,j-1/2}}{\Delta y}, \tag{16}$$

where $F_{i,j+1/2}$ and $F_{i,j-1/2}$ are the fluxes interpolated respectively at the center of the right and left cell faces on the C-Arakawa grid. For simplicity, we have here considered only a 2D system where the subscript i is used in the x direction and j in the y direction. This formulation is by definition energy and mass conserving. The main issue consists now in evaluating the interpolated fluxes F. If we consider polynomial interpolations at the cell centers, the interpolated fluxes can be expressed as:

$$F_{i,j+1/2} = \sum_{l=0}^{n-1} C_{a,l} f\left(u_{i,j-a+l}\right), \tag{17}$$

where n represents the order of the interpolation, a the first point considered for the interpolation, f the local flux and $C_{a,l}$ the coefficients of the polynom. Selecting $a = 1$ and $n = 4$ yields a 4th order central approximation for the interpolated flux given by:

$$F_{i,j+1/2} = \frac{\rho_0}{2}\left(v_{i+1,j} + v_{i,j}\right)\left[-\frac{1}{12}u_{i,j-1} + \frac{7}{12}u_{i,j} + \frac{7}{12}u_{i,j+1} - \frac{1}{12}u_{i,j+2}\right]. \tag{18}$$

Considering the grid is staggered, an interpolation of the advective velocity v at the cell face is required.

Following Durran [**?**], hyperviscosity is added by rewriting equation 16 as:

$$\frac{\rho_0 v \partial u}{\partial y} \equiv \frac{F_{i,j+1/2} - F_{i,j-1/2}}{\Delta y} + \rho_0 \gamma_n \widehat{u_{i,j}}^{nj}, \tag{19}$$

$\widehat{.}^{nj}$ denotes the discretized nth order spatial derivative along j defined, for $n = 4$, by:

$$\widehat{u_{i,j}}^{4j} = u_{i,j+2} - 4u_{i,j+1} + 6u_{i,j} - 4u_{i,j-1} + u_{i,j-2}. \tag{20}$$

γ_n represents the numerical hyperviscosity. If we fix $\gamma_4 = |u_{i,j}|/12\Delta x_j$ along with the use of 4th order central differences, the resulting advection scheme is exactly equivalent to the 3rd order upwind scheme. In practice, we multiply γ_4 by a constant C_{diff}, with $0 \leq C_{diff} \leq 1$ (*avisc* in *cm.nml*) controling the strength of artificial diffusion.

2.3.3 Scalar advection

Regarding scalar advection, additional constraints must be considered in order to make sure that the advected quantities always remain within prescribed bounds and do not develop spurious numerical instabilities or reach unphysical values. For instance, it is obvious that mass mixing ratios (for vapor or hydrometeors) must always remain positive thus requiring a so-called positive-definite scheme. It must also be realized that low dissipation central difference schemes, as presented above for momentum advection, may develop numerical oscillations in regions where large gradients are encountered. Consequently, specific advective schemes must be employed for scalars to avoid the development of instabilities. In the present model, a wide variety of schemes was implemented giving the user a large choice of methods, each having its particularities.

As for momentum advection, we use a conservative flux formulation for scalar advection:

$$\frac{\partial \rho_0 \Phi}{\partial x} \equiv \frac{F_{i+1/2,j} - F_{i-1/2,j}}{\Delta x}. \tag{21}$$

Several methods are available to estimate the scalar fluxes F, which can all be accessed by the *scal_adv* parameter in *cm.nml*:

tvd enables the finite difference TVD scheme. The default flux limiter there is the MC limiter.

muscl enables the (at most 2nd order) MUSCL (piecewise linear) finite volume scheme. The default flux limiter there is the MC limiter.

quick enables the (at most 3rd order) QUICK finite volume scheme, with its default SMART flux limiter.

ppm enables the (at most 3rd order) PPM (piecewise parabolic) finite volume scheme, with hierarchical flux limiter.

The *muscl* option is the default. In the following, only the default MUSCL scheme is introduced (the other schemes have a similar design, and more details can be found for example in [**?**]).

The MUSCL scheme [**?**] is a central method so that it generates minor numerical dissipation which is a great advantage compared to the usual upwind biased methods. Starting with equation 21, we want to evaluate the flux swept through the cell face (i.e. $F_{i+1/2,j}$ at $x_{i+1/2}$) during a time step Δt. This can be expressed exactly following:

$$F_{i+1/2,j} = \frac{1}{\Delta t} \int_{x_{i+1/2} - u_{i+1/2,j}\Delta t}^{x_{i+1/2}} \widetilde{\Phi}\left(x\right) dx. \tag{22}$$

$\widetilde{\Phi}$ is an approximation of the solution Φ. We can define $\widetilde{\Phi}$ as a piecewise linear function between $x_{i-1/2}$ and $x_{i+1/2}$, giving:

$$\widetilde{\Phi}\left(x\right) = \Phi_i + \frac{\Phi_{i+1} - \Phi_i}{\Delta x}\left(x - x_i\right). \tag{23}$$

We have here voluntarily dropped the j index for clarity. Using the linear approximation, the integral 22 can be evaluated exactly:

$$F_{i+1/2} = \frac{1}{\Delta t} u_{i+1/2} \left[\Phi_i + \frac{1}{2}\left(1 - |\mu|\right)\left(\Phi_{i+1} - \Phi_i\right)\right] \tag{24}$$

where we have introduced the CFL number $\mu = u_{i+1/2}\Delta t/\Delta x$. Note that on the Arakawa C-grid, because u is evaluated at face centers, the advective velocity $u_{i+1/2}$ is directly available and does not require any interpolation.

As mentioned previously, scalar advection often requires the use of so-called limiters to prevent unphysical values to be produced as a result of numerical errors. We thus rewrite equation 24 as:

$$F_{i+1/2} = \frac{1}{\Delta t} u_{i+1/2} \left[\Phi_i + \frac{1}{2} C_{i+1/2}\left(1 - |\mu|\right)\left(\Phi_i - \Phi_{i-1}\right)\right] \tag{25}$$

where we have now introduced the flux limiter $C_{i+1/2}$. $C_{i+1/2}$ si defined as a function of the ratio $\left(\Phi_{i+1} - \Phi_i\right) / \left(\Phi_i - \Phi_{i-1}\right)$. For example, the default MC limiter reads:

$$C_{i+1/2} = \max\left(\min\left(2\frac{\Phi_{i+1} - \Phi_i}{\Phi_i - \Phi_{i-1}}, \frac{1}{2}\left(\frac{\Phi_{i+1} - \Phi_i}{\Phi_i - \Phi_{i-1}} + 1\right), 2\right), 0\right). \tag{26}$$

To generalize, we can rewrite the flux $F_{i+1/2}$ as:

$$F_{i+1/2} = F^{LO}_{i+1/2} + \frac{1}{2} C_{i+1/2}\left(F^{HO}_{i+1/2} - F^{LO}_{i+1/2}\right), \tag{27}$$

where F^{LO} and F^{HO} represent low-order and high-order fluxes respectively. The role of the flux limiter is then to selectively switch between the low-order, upwind biased formulation (very diffusive to prevent the development of numerical oscillations) in the presence of strong gradients, and the 2nd order central scheme (more dispersive and prone to produce wiggles) where gradients are smooth enough. For computational efficiency, the limiter 26 is not applied everywhere in the domain, but only in regions where $|\Phi_{i+1} - \Phi_i| > \lambda |\Phi_i|$, with λ set by default to 10^{-9} (this can be changed through the parameter *limit_tol* in *cm.nml*).

In the situation where $u_{i+1/2} < 0$, the upwinding must be reversed:

$$F^-_{i+1/2} = \frac{1}{\Delta t} u_{i+1/2} \left[\Phi_{i+1} - \frac{1}{2} C^-_{i+1/2} \left(1 - |\mu|\right) \left(\Phi_{i+2} - \Phi_{i+1}\right) \right], \tag{28}$$

and $C^-_{i+1/2}$ is now a function of $\left(\Phi_{i+1} - \Phi_i\right) / \left(\Phi_{i+2} - \Phi_{i+1}\right)$. By denoting $F^+_{i+1/2}$ the flux defined by equation 25, we can finally write, for a velocity of arbitrary sign:

$$F_{i+1/2} = 0.5 \left(1 + sign\left(1, u_{i+1/2}\right)\right) F^+_{i+1/2} + 0.5 \left(1 - sign\left(1, u_{i+1/2}\right)\right) F^+_{i+1/2}. \tag{29}$$

The function $sign\left(1, u_{i+1/2}\right)$ has the sign of $u_{i+1/2}$ and an absolute value equal to 1.

The QUICK and PPM methods are two extensions of the scheme introduced above considering piecewise parabollic approximations to Φ instead of piecewise linear functions.

2.4 Time integration

In the current version of the model, two time integration methods are available: the simple 1st order Euler forward scheme, and a 3rd order predictor-corrector method. The Euler forward method is the default in MIMICA, while if *setenv PREDICTOR_CORRECTOR TRUE* in *start*, the predictor-corrector method is selected.

To simplify, advancing a given scalar Ψ using the forward Euler method reduces to (in semi-discrete form):

$$\Psi^{n+1} = \Psi^n + \Delta t \mathcal{F}^n \left(\Psi\right), \tag{30}$$

where $\mathcal{F}^n$ is the total Ψ tendency (sum of advection and physics contributions) evaluated at time n. The forward Euler method is extremely simple, but it is only 1st order accurate and possesses a very limited domain of stability. Using a simple stability analysis method (for example van Neuman), it can be shown that this scheme is only stable for $CFL < 1$, where CFL is the Courant-Friedrichs-Lewy criterion characterizing the propagation of the dominant waves inside the numerical domain.

In the fully compressible case, where no particular approximation has been made, the acoustic waves propagating at a speed $c = \sqrt{\gamma p / \rho}$ must be resolved, giving $CFL = \left(|u| + c\right) \Delta t / \Delta x$. In the anelastic case however, acoustic waves are filtered out, and numerical stability depends on a purely advective criterion: $CFL = |u| \, \Delta t / \Delta x$. It is then easy to realize that because c is generally an order of magnitude larger than $|u|$ in most atmospheric applications, a compressible solver will typically require time steps that are at least 10 times smaller than with an anelastic solver.

In general in MIMICA, Δt is adjusted dynamically during the course of the simulation to satisfy the condition $CFL_{min} < CFL < CFL_{max}$. Both CFL_{min} and CFL_{max} can be set in *cm.nml* (*cfl_min* and *cfl_max* respectively). Although the theoretical stability limit for the forward Euler scheme is 1, the true limit often observed in practice is much smaller than that, $\sim 0.5 - 0.7$. Note that in three dimensions, the CFL number is defined in MIMICA as the sum of the CFL numbers calculated along each dimension independently.

The second time integration scheme available in MIMICA is a 3rd order predictor-corrector method. As the first predictor step, the scheme employs a two step Adams-Bashforth method:

$$\Phi^* = \Phi^n + \frac{\Delta t}{2} \left(3\mathcal{F}^n - \mathcal{F}^{n-1}\right), \tag{31}$$

where Φ^* represents an intermediate value of Φ, and $\mathcal{F}^{n-1}$ is the total scalar tendency calculated and stored at time $n-1$. The corrector step is then defined as the two-step (implicit) Adams-Moulton method:

$$\Phi^{n+1} = \Phi^n + \frac{\Delta t}{12} \left(5\mathcal{F}^* + 8\mathcal{F}^n - \mathcal{F}^{n-1}\right). \tag{32}$$

This predictor-corrector method has the advantage to be 3rd order accurate, and possesses an extended domain of stability compared to the forward Euler scheme (theoretical CFL_{max} of ~ 1.7 instead of 1). It must be reminded however that all computations must be executed twice to complete a time-step, which doesn't make it faster than the Euler forward method in the end, even if CFL_{max} can be increased. Furthermore, it must be noted that the amount of data that must be stored by this predictor-corrector scheme is about 3 times that required by the Euler forward (data at three time stages, t^{n-1}, t^n, and t^*, are needed for the corrector step).

2.5 Physical parameterizations

2.5.1 Subgrid scale turbulence

Turbulence models are used to evaluate the subgrid scale (SGS) turbulent fluxes of all transported quantities, and are denoted τ_Ψ. Typically, a simple gradient hypothesis is used to model these terms, by analogy with molecular diffusion:

$$\tau_{\Psi,i} = -K_h \frac{\partial \Psi}{\partial x_i} \tag{33}$$

for scalars and

$$\tau_{u,i,j} = -K_m \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \tag{34}$$

for momentum. Two approaches can then be used in MIMICA to evaluate the eddy diffusivities K_h and K_m.

The first approach, selected by setting *setenv TKE TRUE* in *start*, is a 1st order closure based on the solution of a turbulent kinetic energy (TKE) equation [**?**]. A closed form of the SGS TKE equation can be written (in non-conservative form):

$$\frac{\partial e}{\partial t} + u_i \frac{\partial e}{\partial x_i} = 2K_m S_{ij} S_{ij} + 2\frac{\partial}{\partial x_i}\left(K_m \frac{\partial e}{\partial x_i} \right) - K_h \frac{g}{\theta_v}\frac{\partial \theta_v}{\partial z} - C_\epsilon \frac{e^{3/2}}{l}. \tag{35}$$

In the above equation, e is the SGS TKE, $S_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right)$ is the strain tensor, and l a characteristic turbulent length scale. The characteristic length scale can be defined as (in 3D):

$$l_\Delta = \left(\Delta x \Delta y \Delta z \right)^{1/3}, \tag{36}$$

and:

$$l = \min\left(l_\Delta, 0.76\sqrt{\frac{e}{\mathcal{N}^2}}, 0.5 C_l \Delta z\,(1) \right) \tag{37}$$

with z the local altitude, κ the Von Karman constant (≈ 0.41),C_l a constant of the model (given below), and $\Delta z\,(1)$ the size of the first model level. The eddy diffusivity K_m is finally obtained using simple scaling arguments:

$$K_m = C_m l \sqrt{e}. \tag{38}$$

The constants C_l, C_ϵ and C_m are set to 0.845, 0.845 and 0.0856 respectively. It is then usually assumed that $K_h = K_m / Pr$, with Pr the turbulent Prandtl number (set by *pran* in *cm.nml*, with a recommended value of ~ 0.4).

The second closure is based on the 0th order Smagorinsky and Lilly approach [**?**]. If we assume a balance between production of TKE by shear plus buoyancy and dissipation in equation 35, it follows:

$$2K_m S_{ij} S_{ij} - K_h \mathcal{N}^2 - C_\epsilon \frac{e^{3/2}}{l} = 0. \tag{39}$$

Rearranging the terms yields:

$$K_m = \left(C_S l \right)^2 \sqrt{2 S_{ij} S_{ij}} \sqrt{1 - \frac{Ri}{Pr}} \tag{40}$$

with $C_S = 0.18$. We have here introduced the Richardson number $Ri = \frac{\mathcal{N}^2}{|\mathbf{S}|}$ to account for convective stability. In this case, the Brunt-Vaisala frequency $\mathcal{N}^2$ is modeled following [**?**] to include changes in stability related to condensation/evaporation at the SGS level:

$$K_h \mathcal{N}^2 = -\frac{g}{\theta_{00}} \left\langle w' \theta_v' \right\rangle \approx K_h \frac{\partial b}{\partial z}. \tag{41}$$

By considering the buoyancy production generated by the mixing of two separate levels in saturated or subsaturated conditions, Stevens et al. [**?**] propose to model $\mathcal{N}^2$ following:

$$\mathcal{N}^2 = \frac{g}{\theta_{00}} \left[\Delta\theta_l + 0.608 \theta_{00} \Delta q_t + \frac{n}{2} G \left(\Delta q_t - F \Delta \theta_l \right) \right]. \tag{42}$$

F and G are defined by : $F = \Pi dq_s/dT$ and $G = \left(L_v/c_p \Pi - \theta_{00}\left(1 + r_m\right) \right) / \left(1 + L_v/c_p dq_s/dT \right)$. n represents the number of consecutive saturated layers. In the model, we only consider the two neighbouring layers at a given level so that n is always set to 2 (within the cloud layer) or 0 (in subsaturated conditions). The above relation was developed

by considering that the cloud water in saturated conditions is adjusted to saturation. This model does not account for the presence of ice.

The scalar eddy diffusivity is defined as the ratio between K_m and a turbulent Prandtl number (*pran* in *cm.nml*). In MIMICA, it is possible to set *pran* to a negative value, in which case turbulent diffusion is only applied up to the altitude set by *zdec*, and then tends quickly to 0 above. Provided that all scalars are advected using flux-limited schemes which generate numerical diffusion in regions with strong gradients (see section 2.3), it can be argued that numerically induced diffusion is already sufficient and there is no need for additional diffusion through an explicit SGS contribution.

2.5.2 Surface modeling

The surface in MIMICA is treated as a solid wall upon which the vertical velocity vanishes. We therefore first enforce that:

$$w\left(x, y, z = z_1\right) = 0. \tag{43}$$

The modeling of turbulent surface fluxes (for momentum, potential temperature and moisture) is controled through the parameter *isurf* present in *cm.nml*. *isurf* is an integer which can take any value between 0 and 4:

0 Prescribed surface fluxes

1 Prescribed skin surface temperature (SST)

2 Prescribed SST with fixed drag coefficient

3 Similar to 2 but with u^* defined as in [**?**]

4 No surface exchanges (all fluxes set to 0)

Only cases 0, 1 and 2 are described below.

Options 0 and 1 rely on the Monin-Obukhov similarity theory according to which momentum and virtual potential temperature fluxes at the ground are proportional to the friction velocity u^*:

$$u^* u^* = \overline{u'w'} = \overline{v'w'}. \tag{44}$$

$$u^* \theta_v^* = \overline{\theta_v' w'}. \tag{45}$$

In the simple case where surface sensible and latent heat fluxes are prescribed (*isurf* = 0), there is no need to estimate θ_v^* as $\overline{\theta_v' w'}$ is readily given, but u^* still needs to be computed to calculate the surface momentum flux. The turbulent surface fluxes are then rewritten using the usual gradient hypothesis:

$$\overline{u'w'} = -K_m \frac{\partial u}{\partial z}, \tag{46}$$

with K_m approximated by:

$$K_m = \kappa z u^*, \tag{47}$$

where κ is the von Karman constant (≈ 0.41) and z is the altitude of the first velocity grid point. Combining these relations and integrating over z yields the classical logarithmic shape of the velocity profile inside the surface layer:

$$ln\left(\frac{z}{z_0}\right) = \frac{\kappa \left|u\right|}{u^*}, \tag{48}$$

from which u_* can be deduced. Here, z_0 is a surface roughness height depending on the ground nature. z_0 is set by default to $zrough = 1.e-4\ m$ in *cm.nml*, and must be adjusted depending on the simulated surface type. Relation 48 remains valid as long as the surface layer remains neutral. For stable and unstable BL, Monin-Obukhov similarity theory is used to derive relations similar to 48. This first requires the evaluation of a surface buoyancy flux which can take different forms depending on the selected option.

***isurf* = 0: Fixed surface fluxes** The surface buoyancy flux is defined as:

$$\overline{b'w'} = \frac{g}{\theta_{00}} \overline{\theta_v' w'} = \frac{g}{\rho_0 \theta_{00}} \left(\frac{SHF}{cp} + \epsilon \theta_{00} \frac{LHF}{L_v} \right), \tag{49}$$

Where SHF and LHF must be prescribed in *cm.nml* (keywords *shf* and *lhf* respectively). We then define the Monin-Obukhov length scale by:

$$L = -\frac{u^{*3}}{\kappa \overline{b'w'}}, \tag{50}$$

and deduce the friction velocity from a relation similar to 48:

$$ln\left(\frac{z}{z_0}\right) - \Psi_m(\zeta) = \frac{\kappa |u|}{u^*} \tag{51}$$

in which Ψ_m (the stability function) has been introduced to account for stability/instability at the surface. The ratio $\zeta = z/L$ can be viewed as a measure of surface layer stability (stable for $\zeta > 0$ and unstable for $\zeta < 0$). Ψ_m is commonly obtained from observations and depends on the sign of ζ. We use here the forms proposed by Garratt [**?**]:

$$\text{for } \zeta > 0 \qquad \Psi_m(\zeta) = -5.3\zeta \tag{52}$$

$$\text{for } \zeta < 0 \qquad \Psi_m(\zeta) = 2ln\left(\frac{1+x}{2}\right) + ln\left(\frac{1+x^2}{2}\right) - 2tan^{-1}(x) + \frac{\pi}{2} \qquad \text{and} \qquad x = (1-16\zeta)^{1/4} \tag{53}$$

As this was the case for the neutral boundary layer, u^* can be derived from relation 51. However, L depending directly on u^*, an iterative procedure is required to obtain a converged value of u^*. The friction virtual potential temperature can then be deduced from: $\theta_v^* = -\frac{\theta_{00} b}{g u^*}$. Of course, the applied sensible and latent heat fluxes are directly given by the prescribed SHF and LHF, and do not need to be recomputed using θ_v^*.

***isurf* = 1: Fixed surface properties** The surface buoyancy flux is now calculated from the gradient between the virtual potential temperature in the first grid cell above the surface and a skin surface virtual potential temperature:

$$b = \frac{g}{\theta_{00}}(\theta_{v1} - \theta_{v,skin}) = \frac{g}{\theta_{00}}\left[(\theta_1 - \Pi_{00} SST) + 0.608\theta_{00}(q_{t1} - SSM)\right]. \tag{54}$$

SST and SSM are the prescribed skin surface temperature and surface moisture (where it is generally assumed that the near surface air is saturated) which can be prescribed in *cm.nml*. An initial friction potential temperature can readily be obtained under neutral conditions following:

$$Pr_t ln\left(\frac{z}{z_0}\right) = \frac{kb\theta_{00}}{g\theta_{v*}^*}, \tag{55}$$

with Pr_t the turbulent Prandtl number defined in *cm.nml*. We can now define as previously the Monin-Obukhov characteristic length scale, but using the surface buoyancy flux defined above:

$$L = \frac{u^{*2}\theta_{00}}{\kappa g \theta_v^*}. \tag{56}$$

From these relations, the friction virtual potential temperature can be deduced:

$$Pr_t\left[ln\left(\frac{z}{z_0}\right) - \Psi_h(\zeta)\right] = \frac{\theta_{00} b}{g\theta_v^*}. \tag{57}$$

The friction velocity is still estimated following 51. Ψ_m doesn't change compared to the fixed surface fluxes case, and Ψ_h is given by:

$$\text{for } \zeta > 0 \qquad \Psi_h(\zeta) = -5.3\zeta \tag{58}$$

$$\text{for } \zeta < 0 \qquad \Psi_h(\zeta) = 2ln\left(\frac{1+y}{2}\right) \qquad \text{and} \qquad y = (1-16\zeta)^{1/4}. \tag{59}$$

Once again, the dependence of L on u^* and θ_v^* requires subiterations to yield an accurate solution. Once a converged value of θ_v^* is obtained, it can be used to determine the surface latent and sensible heat fluxes.

***isurf* = 2: Fixed surface drag** The option *isurf* = 2 implies a very simple surface flux formulation whereby friction quantities at the surface are obtained using:

$$u^* = C_d |u_1|, \tag{60}$$

and:

$$\theta_v^* = \frac{C_d}{Pr_t}(\theta_{v1} - \theta_{vs}). \tag{61}$$

θ_{vs} is the surface value of the virtual potential temperature calculated using a fixed SST (prescribed in *cm.nml*) and assuming near surface air is saturated ($q_{vs} = q_{sat}(SST)$). The drag coefficient C_d can be set in *cm.nml*. Its default value is 1.2×10^{-3}. Again, sensible and latent heat fluxes can then be recomputed based on θ_v^* and u^*.

2.5.3 Radiation

The radiation solver coupled to MIMICA is the 2005 version of the Fu-Liou model developed by Fu and Liou [**?**]. The solver is a multiband 2-4 stream radiation model and requires certain input files and options to operate properly. Running MIMICA with interactive radiation is done by setting *setenv RADIA TRUE* in *start*. Because the radiative transfer model is the slowest part of all the MIMICA model, radiation is only calculated every *iradx* seconds. *iradx* can be modified in *cm.nml* and its default value is set fo 30 s.

The radiation solver works on 1D columns extending up to the top of the atmosphere. Because MIMICA's domain is limited in altitude, the input data must be completed using standard atmosphere above the numerical domain. Standard soundings are stored in *.lay* files located in the *DATA* directory. The most appropriate standard atmosphere must then be renamed *sounding_rad.dat* to be read by MIMICA. In addition to the standard atmosphere, the radiation solver also requires two additional tables located in *DATA* containing informations used by the CKD (correlated k-distribution) model: *ckd.dat* and *cldwtr.dat*.

In addition to these 1D soundings, surface parameters must also be provided as inputs to the radiation solver. These parameters are: the surface albedo (*alb* in *cm.nml*, set by default to 0.07), the surface skin temperature (*sst* in *cm.nml*, set by default to 295 K), and the surface emissivity (*emi* in *cm.nml*, set by default to 0.984). To finish with, the solar zenith angle is computed at each radiation step. The formula uses the prescribed latitude, julian day, starting time of the simulation (UTC) and the local simulation time. All these parameters can also be set in *cm.nml*.

Note that in some specific cases, as in the DYCOMS and ISDAC templates for example, radiation is computed interactively as a simple function of the liquid water path only, without calling the radiative transfer model introduced above.

2.5.4 Microphysics

2.5.5 Basic definitions

A two-moment bulk microphysics scheme is implemented in the code for all the hydrometeors (including cloud droplets) where assumed size distributions are used, taking the form of regular gamma functions denoted $\Gamma(x)$. They are expressed as:

$$dN = N_0 D^{\alpha} e^{-\lambda D} dD, \tag{62}$$

where N is the number concentration of a given hydrometeor population having diameter D, α is a constant depending on the hydrometeor type, and N_0 and λ are two parameters which can be expressed in terms of N and Q the mass mixing ratio.

$$N_0 = \frac{N\lambda^{\alpha+1}}{\Gamma(\alpha+1)} \qquad \text{and} \qquad \lambda = \left(\frac{A_m \Gamma(\alpha+\beta_m+1)\, N}{\Gamma(\alpha+1)\, Q}\right)^{1/\beta_m}. \tag{63}$$

The parameters A_m and β_m are the constants in the particle's mass-size relation given by:

$$m = A_m D^{\beta_m}. \tag{64}$$

One important property of Γ that will be used later on must be introduced here: $\Gamma(x+1) = x\Gamma(x)$.

N and Q are defined by:

$$N = \int_0^{+\infty} f(D)\, dD \qquad \text{and} \qquad Q = \int_0^{+\infty} m f(D)\, dD, \tag{65}$$

with $f(D) = \frac{dN}{dD}$. Therefore, N is considered to be the 0th moment of distribution 62 while m is the β_mth moment. The mean diameter of a given hydrometeor population, which will be used later, is the first moment of 62: $\overline{D} = \int_0^{+\infty} D f(D)\, dD$. The values of coefficients α, β_m and A_m for all the hydrometeor kinds defined in the model are summarized in table 1. Note that species terminal fall speeds, V_p, are modeled using simple power laws of the diameter as well (the coefficients are gathered in table 1):

$$V_p = A_v D^{\beta_v} \left(\frac{\rho_0}{\rho}\right)^{\gamma}. \tag{66}$$

Cloud droplets are commonly assumed not to precipitate so that $A_{v,c} = 0$.

At the moment, five types of hydrometeors are described by the model: cloud droplets, rain drops, cloud ice, graupel and snow. In the following, quantities refering to a specific type of hydrometeor will be denoted by the initials c, r, i, g or s.

Ice crystals may take different shapes depending on the thermodynamic conditions under which they were nucleated. At the moment, only one type of ice crystal habit can be assumed at a time for each simulation. The proper ice habit has to be selected prior to the beginning of the calculation in *shared_hydro.f90* where a list of ice microphysical parameters is provided (hard coded). All the default ice crystal types are summarized in table 1.

hydrometeor	α	βm	A_m	β_v	A_v
cloud (c)	5	3	523.6	-	-
rain (r)	0	3	523.6	0.8	836.
ice (i, rosette C2a)	2	2.26	0.102	1.225	6059.5
ice (i, columns C1f)	2	1.8	0.0093	0.488	30.14
ice (i, plates P1c)	2	2.79	1.43	0.62	17.9
ice (i, dendrites P1e)	2	2.29	0.0233	0.48	5.02
ice (i, assemblage of den. P7b)	2	2.68	0.35	0.83	62.62
graupel (g)	0	3	65	0.8	199.05
snow (s)	1	2	0.04	0.333	6.962

Table 1: *Coefficients for the microphysics scheme for all five kinds of hydrometeors. Parameters for ice crystals are taken from Propacher and Klett [?] and Khvorostyanov and Curry [?]. Units are SI (kg-m-s).*

The choice of the microphysics level is controled by the integer lmicro in *cm.nml*: lmicro = 0 corresponds to a dry case where no condensed water is present (no microphysics calculated), for lmicro = 1, only warm microphysics is modeled (cloud drops and rain, drizz in *cm.nml* even allows to swith on and off the formation of precipitating rain drops), for lmicro = 2, ice microphysics is included but only for pristine ice crystals (no riming and no aggregation) whereas lmicro = 3 includes the formation of snow flakes and graupel (the 5 hydrometeor species, $nh = 5$).

nh and *nhydro* must not be confused: *nhydro* is the total possible number of hydrometeors which is used to allocate and initialize all the arrays depending on microphysics, whereas *nh* is the number of modeled hydrometeors and is used for the calculations.

2.5.6 Equivalence with mass distributions

It is sometimes convenient to describe hydrometeor populations in terms of mass distributions instead of size. Combining relations 64 and 62 yields a distribution under the form of a generalized gamma distribution:

$$dN = M_0 m^\nu e^{-\Lambda m^\mu} dm, \tag{67}$$

where M_0 and Λ are the new parameters that need to be redefined, and ν and μ two constants dependent on the type of hydrometeor. Comparing the mass and size distributions the following equalities come:

$$M_0 = \frac{N_0}{\beta A_m^{\frac{\alpha+1}{\beta_m}}}, \tag{68}$$

$$\Lambda = \frac{\lambda}{A_m^{1/\beta_m}}, \tag{69}$$

$$\nu = \frac{\alpha+1}{\beta_m} - 1, \tag{70}$$

$$\mu = \frac{1}{\beta_m}. \tag{71}$$

The kth moment of this generalized distribution is given by:

$$M^{(k)} = \int_0^{+\infty} m^k f(m)\, dm \tag{72}$$

$$= M_0 \int_0^{+\infty} m^{k+\nu} e^{-\Lambda m^\mu} dm \tag{73}$$

$$= M_0 \frac{\Gamma\left(\frac{\nu+k+1}{\mu}\right)}{\mu} \Lambda^{-\frac{\nu+k+1}{\mu}}. \tag{74}$$

Note that using relation 74, expressions for M_0 and Λ as functions only of N, Q and the parameters ν and μ can be derived:

$$M_0 = \frac{\mu N}{\Gamma\left(\frac{\nu+1}{\mu}\right)}\Lambda^{\frac{\nu+1}{\mu}} \qquad \text{and} \qquad \Lambda = \left[\frac{\Gamma\left(\frac{\nu+1}{\mu}\right)Q}{\Gamma\left(\frac{\nu+2}{\mu}\right)N}\right]^{-\mu}. \tag{75}$$

Replacing the ν and μ by their corresponding expressions depending on α, A_m and β_m, we find:

$$M_0 = \frac{N}{\beta\Gamma\left(\alpha+1\right)}\Lambda^{\alpha+1} \qquad \text{and} \qquad \Lambda = \left[\frac{\Gamma\left(\alpha+1\right)Q}{\Gamma\left(\alpha+\beta+1\right)N}\right]^{-1/\beta}. \tag{76}$$

2.5.7 Collisions between hydrometeors

The collisional sources and sinks are obtained by integrating a collection kernel $K\left(m, m^*\right)$ over two different hydrometeor populations having masses m and m^*. The time variation of a mass distribution $f\left(m\right)$ due to collision processes can be derived, commonly called the Stochastic Collection Equation:

$$\frac{\partial f\left(m\right)}{\partial t} = \frac{1}{2}\int_0^m f\left(m-m^*\right) f\left(m^*\right) K\left(m-m^*, m^*\right) dm^* - \int_0^{+\infty} f\left(m\right) f\left(m^*\right) K\left(m, m^*\right) dm^*. \tag{77}$$

Introducing the kth moment $M^{(k)}$ of m in the previous relation, an equation for the evolution of $M^{(k)}$ can be found [**?**]. From this general equation (not shown here), and after further simplifications, some simpler expressions can be derived giving the rate of change of the particle's number densities and mass mixing ratios due to collision/coalescence events.

$$\frac{\partial N}{\partial t} = \int_0^{+\infty} f\left(m\right)\left[\int_0^{+\infty} f\left(m^*\right) K\left(m, m^*\right) dm^*\right] dm, \tag{78}$$

and:

$$\frac{\partial Q}{\partial t} = \int_0^{+\infty} mf\left(m\right)\left[\int_0^{+\infty} f\left(m^*\right) K\left(m, m^*\right) dm^*\right] dm, \tag{79}$$

The previous equations represent the evolution of the number and mass of a given hydrometeor species Ψ due to collisions with particles Ψ^* belonging to another species. These relations can be derived without any assumption on the collection kernel or on the mass distribution.

Given that f is usually assumed to take the shape of gamma distribution, the only unknown in the previous equations is the collection kernel K which needs to be presumed. The original model uses the gravitational kernel which reads:

$$K\left(D, D^*\right) = \rho E^* \frac{\pi}{4}\left(D+D^*\right)^2 \left|V_p - V_p^*\right|. \tag{80}$$

The coefficient E^* represents the collision efficiency and usually depends on the considered microphysical species and local conditions. This modelling assumption is still in use in the mixed-phase microphysics part of the LES. Warm microphysics processes are however now modeled using Seifert and Beheng's approach which makes use of a slightly different kernel as shown in the next section.

Provided that the SCE is given in terms of particle masses, it may not be adequate to represent an hydrometeor population by its size distribution. This issue is however commonly overcome by simply assuming that the previous relations for the mass and number microphysical sources can directly be extended to a size distribution. The correspondance between the two relations may not be perfectly exact as mass and diameter are related by a linear power law formula in such a way that additional terms or factors should be included. The assumption seems nevertheless acceptable and will therefore be used in the following.

2.5.8 Warm microphysics: Seifert & Beheng's scheme

Warm microphysics (involving only liquid water) will be treated using Seifert and Beheng's model [**?**, **?**]. The microphysics parameterization uses a double moment approach for both cloud droplets and rain where mass distributions of these two species are represented by generalized gamma functions. The approach chosen by Seifert and Beheng is thus slightly different from the framework introduced previously, but the same assumptions still apply.

The authors use a simple kernel defined by a picewise polynome (Long's kernel):

$$K\left(m, m^*\right) = k_c\left(m^2 + m^{*2}\right) \tag{81}$$

$$K\left(m, m^*\right) = k_r\left(m + m^*\right). \tag{82}$$

The first polynome is used when the mass of the collector m is smaller than a critical mass m_c separating cloud drops and rain drops. This mass is calculated from the separation diameter between cloud droplets and rain prescribed by the parameter *drmin* in *cm.nml*. Cloud drops can only collect smaller cloud drops so that the first polynome will only represent cloud/cloud interactions. The second one is used when the collector's mass is greater than m_c, which corresponds to rain/cloud or rain/rain interactions. The constants k_c and k_r are given the values 9.44*e*9 cm3/g2/s and 5.78*e*3 cm3/g/s respectively.

Integrating the mass distributions following relations 78, and using the above kernels leads to the exact expressions for the rates of autoconversion (c+c → r), accretion (r+c → r) and selfcollection (c+c → c or r+r → r). These rates can be found in [**?**, **?**] and are recalled hereafter:

$$\left.\frac{\partial Q_r}{\partial t}\right|_{auto} = auq = \frac{k_c}{20m_c}\frac{(\nu_c+2)(\nu_c+4)}{(\nu_c+1)(\nu_c+1)}Q_c^2\overline{m}_c\left[1+\frac{\Phi_a(\tau)}{(1-\tau)^2}\right], \tag{83}$$

$$\left.\frac{\partial N_r}{\partial t}\right|_{auto} = aun = \frac{1}{m_c}\left.\frac{\partial Q_r}{\partial t}\right|_{auto} \tag{84}$$

$$\left.\frac{\partial Q_c}{\partial t}\right|_{auto} = -auq \tag{85}$$

$$\left.\frac{\partial N_c}{\partial t}\right|_{auto} = -2\times aun \tag{86}$$

$$\left.\frac{\partial Q_r}{\partial t}\right|_{acc} = clr = k_r Q_c Q_r \Phi_{ac}(\tau) \tag{87}$$

$$\left.\frac{\partial N_r}{\partial t}\right|_{acc} = 0 \tag{88}$$

$$\left.\frac{\partial Q_c}{\partial t}\right|_{acc} = -clr \tag{89}$$

$$\left.\frac{\partial N_c}{\partial t}\right|_{acc} = clrn = -\frac{1}{\overline{m}_c}clr \tag{90}$$

$$\left.\frac{\partial Q_r}{\partial t}\right|_{sc} = 0 \tag{91}$$

$$\left.\frac{\partial N_r}{\partial t}\right|_{sc} = crrn = -k_r N_r Q_r \Phi_b(\tau) \tag{92}$$

$$\left.\frac{\partial Q_c}{\partial t}\right|_{sc} = 0 \tag{93}$$

$$\left.\frac{\partial N_c}{\partial t}\right|_{sc} = cccn = -k_c\frac{(\nu_c+2)}{(\nu_c+1)}Q_c^2 + 2\times aun \tag{94}$$

where ν_c corresponds to the mass exponent in the mass distribution for cloud droplets. Seifert and Beheng suggest that this parameter should be set to 1, which is equivalent to $\alpha = -1/3$ in a size distribution (considering that $\beta_m = 3$). To remain consistent with table 1 where $\alpha = 5$, a value of $\nu_c = 0$ will be chosen (set in *shared_data*).

In the above, $\overline{m} = Q/N$ is a mean mass, $\tau = 1 - \frac{Q_c}{Q_c+Q_r}$ is a dimensionless scaling parameter, and Φ_a and Φ_b are autoconversion, accretion and break-up rates defined by Seifert and Beheng [**?**] using similarity assumptions as:

$$\Phi_a = 400\tau^{0.7}\left(1-\tau^{0.7}\right)^3 \tag{95}$$

$$\Phi_{ac} = \left(\frac{\tau}{\tau+5e-5}\right)^4 \tag{96}$$

$$\Phi_b = k_b\left(\overline{D}_r - D_{eq}\right) \qquad \text{when} \qquad 0.35mm < \overline{D}_r < D_{eq} \tag{97}$$

$$\Phi_b = 2\exp\left[k_b\left(\overline{D}_r - D_{eq}\right)\right] - 1 \qquad \text{when} \qquad \overline{D}_r > D_{eq} \tag{98}$$

with $k_b = 1000$ 1/m and D_{eq} an equilibrium diameter (corresponding to a balance between break-up and selfcollection) set to 0.9 mm. Φ_b is set to 0 when $\overline{D}_r < 0.35$ mm.

2.5.9 mixed phase microphysics

Mixed-phase microphysics is treated using a similar parameterization as in the original CRM model. Three ice microphysical species are described (ice, graupel and snow) and modeled via their size distributions. The gravitational kernel 80 is used to represent their interactions with any other hydrometeor (including liquid drops).

The collision rates between two hydrometeors, 1 collecting 2, for cold microphysics are given by:

$$P_{n12} = \frac{\pi}{4}\rho E_{12}\int_0^{+\infty}\int_0^{+\infty} f(D_1)\, f(D_2)\,(D_1+D_2)^2\,|V_{p1}-V_{p2}|\,dD_1 dD_2, \tag{99}$$

$$P_{q12} = \frac{\pi}{4}\rho E_{12}\int_0^{+\infty}\int_0^{+\infty} D_2 f(D_1)\, f(D_2)\,(D_1+D_2)^2\,|V_{p1}-V_{p2}|\,dD_1 dD_2, \tag{100}$$

After tidious manipulations and simplifications, one can write the general form of the collision rates:

$$\begin{aligned} P_{n12} = \frac{\pi}{4}\rho E_{12} N_1 N_2 \Bigg[& \left\langle D_1^2\right\rangle \left| \frac{(\alpha_1+\beta_{v1}+2)(\alpha_1+\beta_{v1}+1)}{(\alpha_1+2)(\alpha_1+1)}\left\langle V_{p1}\right\rangle - \left\langle V_{p2}\right\rangle \right| \\ & +2\left\langle D_1\right\rangle\left\langle D_2\right\rangle \left| \frac{\alpha_1+\beta_{v1}+1}{\alpha_1+1}\left\langle V_{p1}\right\rangle - \frac{\alpha_2+\beta_{v2}+1}{\alpha_2+1}\left\langle V_{p2}\right\rangle \right| \\ & +\left\langle D_2^2\right\rangle \left| \left\langle V_{p1}\right\rangle - \frac{(\alpha_2+\beta_{v2}+2)(\alpha_2+\beta_{v2}+1)}{(\alpha_2+2)(\alpha_2+1)}\left\langle V_{p2}\right\rangle \right| \Bigg] \end{aligned} \tag{101}$$

$$\begin{aligned} P_{q12} = \frac{\pi}{4}\rho E_{12} Q_1 N_2 \Bigg[& \left\langle D_1^2\right\rangle_m \left| \frac{(\alpha_1+\beta_{v1}+\beta_{m1}+2)(\alpha_1+\beta_{v1}+\beta_{m1}+1)}{(\alpha_1+\beta_{m1}+2)(\alpha_1+\beta_{m1}+1)}\left\langle V_{p1}\right\rangle_m - \left\langle V_{p2}\right\rangle \right| \\ & +2\left\langle D_1\right\rangle_m\left\langle D_2\right\rangle \left| \frac{\alpha_1+\beta_{v1}+\beta_{m1}+1}{\alpha_1+\beta_{m1}+1}\left\langle V_{p1}\right\rangle_m - \frac{\alpha_2+\beta_{v2}+1}{\alpha_2+1}\left\langle V_{p2}\right\rangle \right| \\ & +\left\langle D_2^2\right\rangle \left| \left\langle V_{p1}\right\rangle_m - \frac{(\alpha_2+\beta_{v2}+2)(\alpha_2+\beta_{v2}+1)}{(\alpha_2+2)(\alpha_2+1)}\left\langle V_{p2}\right\rangle \right| \Bigg] \end{aligned} \tag{102}$$

The notations $\langle . \rangle$ and $\langle . \rangle_m$ respectively denote an ensemble mean over the entire hydrometeor population and a mass weighted ensemble mean. Elementary functions were created in MIMICA to calculate the mean diameters and terminal fall speeds as well as any other useful distribution averaged qantity. P_{q12} represents the mass tendency of species 1 when collected by species 2. It is also possible in a similar manner to define the mass tendency of species 2 following the same process.

Two major simplifications can be obtained in the case where two identical hydrometeors interact (selfcollection) or when the collected particle is a cloud droplet (riming). In the latter case, two assumptions can be made: that the cloud droplet terminal fall speed is 0, and that their size or mass is small compared to the collector particle. We can then rewrite P_{n11} as well as P_{n1c} under the following forms (similar simplifications are obtained for P_q):

$$P_{n11} = \frac{\pi}{2}\rho E_{11} N_1^2 \left\langle D_1^2\right\rangle \left\langle V_1\right\rangle \left| \frac{(\alpha_1+\beta_{v1}+2)(\alpha_1+\beta_{v1}+1)}{(\alpha_1+2)(\alpha_1+1)} - 1 \right|, \tag{103}$$

(during selfcollection, the mass of 1 is conserved in such a way that $P_{q11}=0$)

$$P_{n1c} = \frac{\pi}{4}\rho E_{1c} N_1 N_c \frac{(\alpha_1+\beta_{v1}+2)(\alpha_1+\beta_{v1}+1)}{(\alpha_1+2)(\alpha_1+1)}\left\langle D_1^2\right\rangle \left\langle V_{p1}\right\rangle. \tag{104}$$

These general relations can be applied to any type of hydrometeor provided that the coefficients for the mass and terminal fall speed formula are known.

For most of the interactions, the preceding relations are applied as such. All the possible interactions treated in the code are summarized in table 2. A total of 17 processes are modeled in the present version. Some collision processes present particularities. Riming of ice crystals and snow (resulting from the collision between ice/snow and cloud droplets) is for instance assumed to occur only for cloud droplets having a diameter larger than 15 mm (see [**?**]). In these cases, a partial mixing ratio of cloud droplets effectively rimmed is calculated and multiplied to the integral collection rates. This allows to exclude part of the cloud droplet spectrum from the rimming process. Similarly, autoconversion of ice particles to graupel will occur if the ice particle diameter exceeds a threshold $\hat{D}_i = 0.3$ mm. This transition can only result from deposition growth. In that case, only partial integrals must be calculated (see [**?**]).

hydrometeor types	cloud drops	rain	ice crystals	graupel	snow
cloud drops	c + c → c/r auq, aun, cccn 83, 84, 93, 94	c + r → r clr, clrn 90, 89	c + i → g cli, clii, clin 104	c + g → g clg, clgn 104	c + s → s cls, clsn 104
rain		r + r → r crrn (inc. break-up) 92	r + i → g cir, cirn 101, 102	r + g → g crg, crgn 101, 102	r + s → g csr, csrr, csrn 101, 102
ice crystals			i + i → s i → g (autoconv.) ais, aisn 103, aig, aign	i + g → g - efficiency = 0	i + s → s cis, cisn 101, 102
graupel				g + g → g - rebound	g + s → g csg, csgn 101, 102
snow					s + s → s cssn 103

Table 2: *Interactions between hydrometeors treated in the code.*

2.5.10 Precipitation

Precipitation fluxes appearing in equation **??** include a mean terminal fall speed for each hydrometeor kind, defined as a mass weighted average over the entier hydrometeor population. The power law 66 defines the terminal fall speed of an hydrometeor having a given mass m, including a correction term accounting for vertical density variations. The mass weighted mean terminal fall speeds are thus defined as:

$$\langle V_p \rangle = \frac{\int_0^{+\infty} m V_p\left(D\right) f\left(D\right) dD}{\int_0^{+\infty} m f\left(D\right) dD}. \tag{105}$$

After simplifications, it comes:

$$\langle V_p \rangle = A_v \frac{\Gamma\left(\alpha + \beta_v + \beta_m + 1\right)}{\Gamma\left(\alpha + \beta_m + 1\right)} \lambda^{-\beta_v} \left(\frac{\rho_0}{\rho}\right)^{\gamma}. \tag{106}$$

Considering any hydrometeor moment (N or Q) denoted Ψ, the related precipitation flux reads:

$$P^{\Psi} \equiv -\frac{1}{\rho} \frac{\partial \rho \langle V_p \rangle \Psi}{\partial z}, \tag{107}$$

with $\langle V_p \rangle$ the mass weighted averaged terminal fall speed of the chosen hydrometeor kind. This velocity is expressed following relation 66 and exhibits a non-linear dependence on the two hydrometeor moments. The modeled precipitation flux P^{Ψ} is thus a non-linear derivative term which can generate inherent numerical instabilities in the vicinity of discontinuities or sharp gradients (this is a property of non-linear PDEs). In order to avoid the development of such instabilities, precipitation fluxes are discretized using upstream 1st order differences which introduce large numerical dissipation damping any numerical wave. We write (to simplify, only the vertical index k is shown):

$$P_k^{\Psi} = \frac{1}{\rho_k} \frac{\rho_{k+1} \langle V_p \rangle_{k+1} \Psi_{k+1} - \rho_k \langle V_p \rangle_k \Psi_k}{\Delta z_p}. \tag{108}$$

Note that $\langle V_p \rangle$ being an hydrometeor property, it is calculated at the scalar location and is thus cell-centered.

2.5.11 Phase transitions

Here is defined how phase transitions are treated in the model: evaporation/condensation, sublimation/deposition, melting/freezing. Following Pruppacher and Klett [**?**], the mass tendency of a given droplet of diameter D due to evaporation/condensation is given by:

$$\left.\frac{\partial m}{\partial t}\right|_{e/c} = 2\pi D G_{lv}\left(T, p\right) F_v \frac{s}{q_s}, \tag{109}$$

with:

$$G_{lv}(T,p) = \left[\frac{R_w T}{e_s D_v} + \frac{L_{lv}}{K_T T}\left(\frac{L_{lv}}{R_w T} - 1\right)\right]^{-1} \tag{110}$$

$s = q_v - q_s$ is the absolute supersaturation, q_s the saturation mixing ratio over liquid water, e_s the saturation vapor pressure over liquid water, R_w the ideal gas specific constant for water vapor, D_v the diffusion coefficient for water vapor ($\approx 3.e-5$ m2/s), K_T the heat conductivity ($\approx 2.5e-2$ Jm/s/K). The ventilation factor F_v is defined following Chen and Lamb [**?**] as a function of the Best number $X = Sc^{1/3} Re^{1/2}$:

$$F_v = a_v + b_v X^{\gamma}. \tag{111}$$

The Schmidt number Sc is assumed to be constant and equal to 0.7 whereas the Reynolds number Re is defined by:

$$Re = \frac{\rho_0 V_p D}{\mu}, \tag{112}$$

with V_p the drop terminal fall speed and μ the dynamic viscosity of air ($\approx 1.5e-5$ kg/m/s). For $X \leq 1$, Chen and Lamb suggest that $\gamma = 2$, $a_v = 1$ and $b_v = 0.14$ whereas when $X > 1$, $\gamma = 1$, $a_v = 0.86$ and $b_v = 0.28$ for ice crystals and $\gamma = 1$, $a_v = 0.78$ and $b_v = 0.308$ otherwise. Ventilation effects are neglected for cloud droplets so that F_v is set to 1. For the other hydrometeor species, it is possible to turn on the ventilation effects by setting lvent to 1 in *cm.nml* (these effects are not included by default but might have a large impact on ice crystal growth).

In order to obtain condensation/evaporation sources for the entire drop population, one has to integer relation 126 over the assumed size distribution. To simplify integration, we assume that the final total mass tendency will take the form:

$$\left.\frac{\partial Q}{\partial t}\right|_{e/c} = 2\pi G_{lv}(T,p)\frac{s}{q_s}\int_0^{+\infty} D F_v f(m)\, dm \tag{113}$$

$$= 2\pi N G_{lv}(T,p)\langle D\rangle F_v^* \frac{s}{q_s} \tag{114}$$

where we defined a modified ventilation factor depending on an averaged Reynolds number:

$$F_v^* = a_v + b_v Sc^{1/3}\langle Re\rangle^{1/2} \tag{115}$$

$$= a_v + b_v^* Sc^{1/3}\left(\frac{\beta_v + \alpha + 1}{\alpha + 1}\frac{\langle V\rangle\langle D\rangle}{\nu}\right)^{1/2}. \tag{116}$$

The mass weighted terminal fall speeds are defined previously and b_v^* is a modified factor, not detailed here.

Similar relations are found for sublimation/deposition processes applied to ice crystals, graupel and snow flakes:

$$\left.\frac{\partial m}{\partial t}\right|_{e/c} = 4\pi C_i D G_{iv}(T,p) F_v \frac{s_i}{q_{si}}, \tag{117}$$

and:

$$G_{iv}(T,p) = \left[\frac{R_w T}{e_{si} D_v} + \frac{L_{iv}}{K_T T}\left(\frac{L_{iv}}{R_w T} - 1\right)\right]^{-1}, \tag{118}$$

which yields the following total mass tendencies for the solid hydrometeor populations:

$$\left.\frac{\partial Q}{\partial t}\right|_{s/d} = 4\pi N C_i G_{iv}(T,p)\langle D\rangle F_v^* \frac{s_i}{q_{si}}, \tag{119}$$

C_i is the ice crystal capacitance. It takes the value $1/2$ for spherical particles (graupel) and $1/\pi$ for other shapes. The assumption of a constant capacitance is actually very approximate and more advanced methods with a capacitance depending on crystals shape (through their aspect ratio for example) are more accurate.

The integration of condensation/evaporation sources for water droplets may sometimes lead to numerical instability issues if no specific treatment is applied. Most of the time, these problems lead to the apparition of spurious water mixing ratio peaks at the cloud boundaries where in-cloud air parcels are mixed with the ambient dry atmosphere. Direct integration of the condensation/evaporation sources are nonetheless necessary in two particular cases (even though simpler approaches can be used but with a dramatic drop in model accuracy): when three-phase microphysics is used so that water vapor will interact both with liquid drops and ice crystals, and when droplet activation is modeled,

especially with prognostic aerosols. These issues and the solution adopted in MIMICA are discussed in the following sections.

Melting of ice particles is treated in a similar manner as diffusional growth, with the mass tendency of a single particle given by:

$$\left.\frac{\partial m}{\partial t}\right|_{melt} = \frac{2\pi}{L_{il}}\left[K_T\left(T-T_0\right)F_h + \frac{D_v}{L_{lv}}\left(\frac{p_v}{T}-\frac{p_s}{T_0}\right)F_v\right]D. \tag{120}$$

F_h is a heat ventilation coefficient approximated by $F_h = Le_v F_v$ with $Le_v \approx 0.67$ the Lewis number for water vapor, and $T_0 = 273.15$ K is the freezing temperature. Integrating over the particle population gives:

$$\left.\frac{\partial Q}{\partial t}\right|_{melt} = \frac{2\pi}{L_{il}}N\left[K_T\left(T-T_0\right)Le_v + \frac{D_v}{L_{lv}}\left(\frac{p_v}{T}-\frac{p_s}{T_0}\right)\right]F_v^*\left\langle D\right\rangle, \tag{121}$$

with F_v^* defined as previously. All the melted water is considered to be rain.

In the above, positive tendencies mean growth of drops or ice particles via condensation or deposition of water vapor. Negative tendencies mean a loss of ice or liquid water via sublimation or evaporation. Whereas in the first case the number concentration of hydrometeors is left unchanged (the existing particles only grow), evaporation and sublimation lead to a decrease in droplet and ice crystal number concentrations. These variations of number concentration for ice particles and liquid drops due to sublimation and evaporation are expressed following:

$$\left.\frac{\partial N}{\partial t}\right|_{e/s} = \frac{N}{Q}\min\left(\left.\frac{\partial Q}{\partial t}\right|_{e/s}, 0\right). \tag{122}$$

Here, the subscripts e/s mean evaporation or sublimation as we consider both cloud droplets and ice crystals. The mass mixing ratio tendencies on the right hand side are limited by 0 as number concentrations are only allowed to decrease through evaporation/sublimation.

2.5.12 Saturation adjustment

When a detailed knowledge of supersaturation is not required, the saturation adjustment method can be employed, thus overcoming the need to integrate condensation/evaporation sources. The saturation adjustment assumption comes from the observation that in most stratiform clouds, in-cloud water relaxes very quickly to equilibrium. In other words, omitting CCN activation, it can be assumed that at any time, the in-cloud atmosphere is saturated, $q_v = q_s$ or $s = 0$, so that all the water vapor available above saturation condenses instantaneously on the existing drops. Striclty following this assumption, the liquid water mixing ratio can be diagnosed given q_v and q_s. The method is selected by setting the option SAT_ADJ to TRUE in *start*.

Let's note q_l^n and q_v^n the quantities at time t^n, before adjustment, and q_l^* and q_v^* the same quantities after adjustment. Knowing that $q_t = q_v + q_l$, the total water content, is conserved, we can write:

$$q_l^* = q_t^n - q_s\left(P, T^*\right). \tag{123}$$

Here, q_s is expressed following Rogers and Yau [**?**]:

$$q_s = \frac{380.1}{P}\exp\left(\frac{T-273.15}{T-29.65}\right). \tag{124}$$

The liquid water mixing ratio can be introduced in the previous relation by introducing the liquid potential temperature: $T = \Pi\Theta_l + \frac{L_{lv}q_l}{c_p}$. Relation 123 is then solved using Newton-Raphson iterations while updating q_l, q_v, T and Θ_l after each step. The system to be solved can be written:

$$q_l^{m+1} = q_l^m + \frac{q_t - q_l^m - q_s\left(P, T^m\right)}{1 + \frac{\partial q_s(P,T^m)}{\partial q_l^m}}, \tag{125}$$

where m denotes the current Newton step. Iterations are stopped when $\left|q_l^{m+1} - q_l^m\right| < \epsilon$, and we set $q_l^* = q_l^{m+1}$.

When the saturation adjustment scheme is applied, only the mixing ratio of cloud water is updated. It is thus assumed that all the water vapor in excess condenses over cloud droplets only. Rain mixing ratio is thus taken into account to evaluate the total water mixing ratio necessary for the resolution of the above system, but it is not modified after the procedure is completed.

When considering mixed-phase clouds, the method can still be applied with minor biases in liquid dominated clouds but large errors can appear when the ice/liquid equilibrium lies far from saturation over liquid, that is when the ice

proportion within the cloud layer becomes large. In such conditions, an explicit treatment of condensation/deposition sources is required.

Saturation adjustment schemes for mixed-phase environments have been proposed in the litterature, but none of these solutions yield reasonable results. They usually fail to predict the right repartition between liquid and ice phase. As a result, when mixed-phase clouds are to be simulated using MIMICA, the direct integration of all condensation/sublimation sources is employed.

Note that the saturation adjustment scheme is always used prior to the first time step to initiate the cloud layer. The initial cloud cover/liquid water field is indeed usually diagnosed from the input potential temperature and total water mixing ratio soundings using the saturation adjustment method.

2.5.13 Direct integration of condensation/evaporation

When condensation/evaporation sources (respectively deposition/sublimation) are explicitly integrated, the implicit dependence of the diffusional sources on supersaturation over water and ice may be the source of numerical issues. The diffusional growth terms are simplified for the present demonstration:

$$C_k = \frac{s}{\tau_{lk}} \qquad\qquad (D_k = \frac{si}{\tau_{ik}}), \tag{126}$$

with $s = q_v - q_s$ ($si = q_v - q_{si}$) being the supersaturation over liquid water (ice), and τ_{lk} (τ_{ik}) are characteristic time scales of condensation/evaporation (deposition/sublimation) processes that can be deduced from relations given in 2.5.11. A detailed resolution of supersaturation evolution, which is necessary for a proper representation of condensation/evaporation processes, requires time-steps of the order of milliseconds. As this is not conceivable in LES of atmospheric flows, specific alternatives must be found to preserve the system stability as well as the accuracy of the solution.

In the present LES code, the strong implicit relation between supersaturation and condensation/evaporation is modeled using a pseudo-analytic solution of the supersaturation equation over a time step, as proposed for example by Morrison and Grabowski [**?**]. For the purpose of the demonstration, we simplify the entire system to account only for diffusional growth (mixing and advection were neglected):

$$\frac{\partial q_l}{\partial t} = \frac{s}{\tau_l} \tag{127}$$

$$\frac{\partial q_i}{\partial t} = \frac{si}{\tau_i} \tag{128}$$

$$\frac{\partial q_v}{\partial t} = -\frac{s}{\tau_l} - \frac{si}{\tau_i} \tag{129}$$

The system can be completed by the following relations:

$$\frac{\partial q_s}{\partial t} = \frac{\partial q_s}{\partial T}\frac{\partial T}{\partial t} = -A. \tag{130}$$

$$\frac{\partial q_{si}}{\partial t} = \frac{\partial q_{si}}{\partial T}\frac{\partial T}{\partial t} = -B. \tag{131}$$

where we define $\frac{\partial T}{\partial t}$, a temperature tendency term containing radiation, turbulent mixing as well as advection through updrafts/downdrafts. Combining the previous relations, it is possible to derive balance equations for s and si. The equation for s and si can be written as follows:

$$\frac{\partial s}{\partial t} = -\frac{s}{\tau} + \frac{1}{\tau_i}(s - s_i) + A \tag{132}$$

$$\frac{\partial si}{\partial t} = -\frac{si}{\tau} - \frac{1}{\tau_c}(s - s_i) + B, \tag{133}$$

with the characteristic phase relaxation time:

$$\tau = \left(\frac{1}{\tau_l} + \frac{1}{\tau_i}\right)^{-1}, \tag{134}$$

Equation 132 (respectively 133) can be integrated under the assumption that τ_c, τ_i and A remain constant during time t:

$$s(t) = \tau \left[\frac{s_i}{\tau_i} + \frac{s}{\tau_c} - \tau \left(\frac{B}{\tau_i} + \frac{A}{\tau_c} \right) \right] e^{-\frac{t}{\tau}} + \tau^2 \left(\frac{B}{\tau_i} + \frac{A}{\tau_c} \right) + \frac{\tau}{\tau_i} \left[s - s_i + (A - B)\, t \right]. \tag{135}$$

A very similar relation can be found for si. Compared to the work of Morrison et al. [**?**], the dependence of $s - s_i$ on time has been considered. This yields a more complicated but more accurate solution of the supersaturation equation.

Condensation/evaporation (sublimation/deposition) sources are now evaluated using a time averaged supersaturation integrated over a timestep. Integrating relation 135 over Δt gives

$$\overline{s} = \frac{1}{\Delta t} \int_t^{t+\Delta t} s(t^*)\, dt^* \tag{136}$$

$$= \frac{\tau^2}{\Delta t} \left[\frac{s_i}{\tau_i} + \frac{s}{\tau_c} - \tau \left(\frac{B}{\tau_i} + \frac{A}{\tau_c} \right) \right] \left(1 - e^{-\frac{\Delta t}{\tau}} \right) + \tau^2 \left(\frac{B}{\tau_i} + \frac{A}{\tau_c} \right) + \frac{\tau}{\tau_i} \left[s - s_i + \frac{\Delta t}{2} (A - B) \right]. \tag{137}$$

The mean supersaturation obtained above is then used to calculate the condensation/evaporation sources for liquid drops whereas a mean ice supersaturation (taking a similar form) is used to evaluate the deposition/sublimation sources of all ice particles.

Although quite complicated, the method provides a very stable and accurate way of integrating condensation/evaporation sources when a detailed evolution of supersaturation is required. The issue related to supersaturation integration in high resolution models appears to be a non trivial one and finding efficient methods to overcome the issue (not mentioning the saturation adjustment scheme) is not a simple task.

The model is selected by setting lgrowth to 1 in *cm.nml*. If lgrowth is turned to 0, supersaturation for diffusional growth is evluated explicitly at time t.

2.5.14 Droplet activation

When the chemistry/aerosol modules are deactivated, an empirical parameterization of the number of activated CCN is chosen, based on a power law of the supersaturation:

$$N_{CCN} = N_{CCN,0} S^{\kappa}, \tag{138}$$

with N_{CCN} the number of activated CCN and $N_{CCN,0}$ and κ two constants. $N_{CCN,0}$ is usually defined as the background number of aerosols that can be activated, and κ varies between 0.3 to 0.7 depending on the case. The supersaturation is here defined as:

$$S = \frac{q_v - q_s}{q_s} \times 100, \tag{139}$$

where q_v is the vapor mixing ratio and q_s the saturation mixing ratio. Supersaturation is expressed in % and is explicitly limited to a value of 1% in the code (beware if larger values are expected). Note that S, the relative supersaturation, must not be confused with s, the absolute supersaturation (used in the diffusional growth theory above).

The number of droplets formed is explicitly assumed to be equal to the number of activated CCN: $N_c|_{nuc} = N_{CCN}$. All the droplets thus formed have the same mass which is equal to 4.19e15 kg (the mass of a droplet having the minimum diameter possible).

2.5.15 Ice nucleation

At the moment, only a simple ice nucleation scheme is present in MIMICA, as used in the ISDAC intercomparison for instance. Ice nucleation is defined as a simple relaxation of the ice number concentration towards a fixed background IN concentration:

$$\left. \frac{\partial N_i}{\partial t} \right|_{nuc} = \frac{N_i - IN_0}{\Delta t}. \tag{140}$$

To limit the nucleation events, this relation is only applied where liquid water is also present and where supersaturation over ice is sufficient, for example where $Si > 5\%$. All the newly nucleated ice crystals are assumed to have the same mass, defined as the minimum ice crystal mass possible. It results that:

$$\left. \frac{\partial Q_i}{\partial t} \right|_{nuc} = m_{min} \left. \frac{\partial N_i}{\partial t} \right|_{nuc}. \tag{141}$$

At the moment, m_{min} is hard coded but can be changed easily in *micro_ice.f90*.

On a long term basis, a detailed ice nucleation scheme should be implemented in MIMICA. The model will be based on nucleation theory and follow state-of-the-arts parameterization that can be commonly found in the litterature. In such works, all the different nucleation paths are usually included, namely nucleation via deposition, immersion, contact and condensation/freezing.

2.6 Large scale tendencies (nudging)

In some cases, it can be requested that the simulated wind and scalar fields must be nudged to a given reference state. If we consider one particular quantity Φ, nudging is treated by simply adding one extra term N_Φ in the corresponding balance equation. This term allows the relaxation of the Φ-field towards the reference state Φ_0 with a given time scale τ_n. We then write:

$$N_\Phi\left(x,y,z;t\right) = -\frac{\Phi\left(x,y,z;t\right) - \Phi_0\left(x,y,z;t\right)}{\tau_n}. \tag{142}$$

The relaxation time scale τ_n is usually case and variable dependent. It can be a constant, depend on the vertical coordinate or on the simulation time. The reference state Φ_0 is often taken to be one-dimensional only (dependent on the height only) and equal to the initial state.

In a similar fashion, large scale scalar advection can be considered. Large scale advection is simply modeled by a constant source added to the scalar tendencies (only for ice/liquid potential temperature and total water mixing ratio). These sources can possibly be time dependent (to be defined in *timevar.f90*).

3 MIMICA inputs

3.1 *start*

start is a csh script piloting the compilation of MIMICA. Compilation is simply accomplished by typing

```
./start
```

in your local working directory. *start* also contains a series of configuration options, controlling, for example, the selection of various physical modules, the size of the domain or MPI decomposition. Each option can be activated (resp. deactivated) by uncommenting *setenv XXXXX TRUE* (resp. *setenv XXXXX FALSE*), with *XXXXX* being the desired option keyword. Note that MIMICA must be recompiled entirely after changing anything in *start* by first cleaning all the directory (`make clean`). A *compile.log* file is created locally in your working directory each time *start* is executed. This file contains a summary of all the configuration options used to create the last MIMICA executable.

A description of all the options defined in *start* and their meaning is provided below.

3.1.1 *NP* (default: *setenv NP 1*)

NP corresponds to the number of processors on which MIMICA will be run. Any value of *NP* larger than 1 also requires *setenv SPMD TRUE* to allow MPI parallelization.

3.1.2 *CMPLER* (default: commented)

Setting *setenv CMPLER INTEL* enables compilation with the intel ifort compiler. The default compiler (if *CMPLER* is left commented) is the gfortran compiler.

3.1.3 *DEBUG* (default: *setenv DEBUG FALSE*)

Setting *setenv DEBUG TRUE* turns on all recommended debug options. The MIMICA executable thus created is usually larger than an optimized one, and its execution is about 2 to 10 times slower. It is however strongly recommended to use the DEBUG configuration and run MIMICA with debug options when problems possibly related to coding errors is encountered at run time.

3.1.4 *PROF* (default: *setenv PROF FALSE*)

Setting *setenv PROF TRUE* turns on profiling (with gprof) options. The MIMICA executable thus created can be run to estimate the time spent in each routine at run time. This option should be used for optimization purposes.

3.1.5 *SPMD* (default: *setenv SPMD FALSE*)

Setting *setenv SPMD TRUE* allows parallelization with MPI. MIMICA is then compiled using the appropriate MPI wrapper, and must be executed using commands like *mpirun* or *mpiexec*.

3.2 *cm.nml*

cm.nml is a namelist file containing all the most important options and parameters required to run MIMICA. The namelist is organized in 9 "sections":

cm_run contains general options related to time integration and the simulation period

cm_init contains options related model initialization

cm_grid contains options related to the grid and numerical domain

cm_out contains options related to model outputs

cm_num contains options related to the model's numerical schemes

cm_phys contains options related to physical models

cm_bc contains options related to boundary and surface conditions

cm_micro contains options specific to the microphysics schemes

cm_lag contains options specific to lagrangian particle tracking

Default values for all the parameters contained in *cm.nml* are defined in *INCLUDE/default.h*. If an option is not present in your local *cm.nml* file, the default value is applied automatically. MIMICA does not need to be recompiled when an option is modified in *cm.nml*.

3.2.1 *dt0* (default: *dt0 = 1*)

dt0 is the initial/default model time step in seconds. Unless specified otherwise (see *ldtfix*), *dt0* is used to compute the first time step but is adjusted automatically later on based on the CFL stability criterion.

3.2.2 *ldtfix* (default: *ldtfix = 0*)

If *ldtfix* is set to 0 (default), the time step is adjusted automatically during the course of the simulation to satisfy the CFL stability criterion. If *ldtfix* is set to 1, the time step is held constant and equal to *dt0* during the entire simulation.

3.2.3 *limit_ts* (default: *limit_ts = 20*)

limit_ts is the ration between the default time step *dt0* and the minimum time step allowed during the simulation: after adjustment, the time step cannot become smaller than $dt0/limit\_ts$. If it does, the simulation stops with the following error message: "*ERROR: Time-step too small*".

3.2.4 *tstart* (default: *tstart = 0*)

tstart is the time at initialization.

3.2.5 *tstop* (default: *tstop = 1*)

tstop is the simulation end time in seconds. For example, if *tstart = 0* and *tstop = 86400*, the simulated period will be exactly 1 day.

3.3 *out.nml*

Just like *cm.nml*, *out.nml* is a namelist file used to enable/disable output variables. It is organized in 4 "sections":

ou_in contains switches for all output variables included in the full 2D/3D output file and profiles

ou2d_in contains switches for surface or vertically integrated quantities (eg. surface rain rates or LWP)

oud_in contains switches for the tendencies included in the full 2D/3D output file and profiles

pro_in contains options related to the different horizontal profiles to output

Default values for all the parameters contained in *out.nml* are also defined in *INCLUDE/default.h*.

3.3.1 *out_u* (default: *out_u = .true.*)

out_u allows the output of the horizontal wind (X component).

3.3.2 *out_v* (default: *out_v = .true.*)

out_v allows the output of the horizontal wind (Y component).

3.3.3 *out_w* (default: *out_w = .true.*)

out_w allows the output of the vertical wind.

3.4 Initial soundings and idealized profiles

MIMICA can be initialized by either including idealized profiles of potential temperature θ, total water mixing ratio q_t, and wind velocities, or by specifying initial 1D soundings function of altitude. Both initialization methods depend on external files stored in the *INCLUDE* directory: the former have a *.h* extension and contain short pieces of code written in fortran, while the latter are simple text files with a standard column structure. The use of one or the other method depends on two *cm.nml* keywords (in *cm_init*): the name of the simulation/configuration can be specified with *casename* (certain predefined case names make use of specific idealized profiles, see section 5: "Template and examples"), while *file_init* is the name of the input sounding file to be used for model initialization (the specified file must be present in *INCLUDE*). The choice of one method or the other is essentially case dependent: idealized case studies tend to rely on standard initial profiles provided as simple functions of altitude which can easily be implemented in include *.h* files, while more realistic configurations will typically require the specification of a realistic atmospheric sounding.

As mentioned previously, all 1D soundings must have a similar 5 or 6-column structure, with a single line header necessarily starting with # as follows:

```
#           H         T(C)           RH            U            V
```

Each item on the line is in fact a keyword indicating which variables are to be read in the file:

H indicates the altitude of each sounding level, **in meters above the surface**

T(C) indicates the absolute temperature **in C** at each vertical level. The temperature can alternatively be specified **in K** by specifying the keyword `T(K)` instead. An other possibility is to specify the potential temperature at each vertical level using the keyword `PT`

RH indicates the relative humidity **in %** at each vertical level. The water vapor mixing ratio can alternatively be specified **in g/kg** using the keyword `Q` instead

QL (optional) indicates the liquid water mixing ratio **in kg/kg** at each vertical level

U indicates the horizontal U velocity **in m/s** at each vertical level

V indicates the horizontal V velocity **in m/s** at each vertical level.

Note that `QL` is optional, while the specification of temperature and moisture profiles can be done in several ways depending on the selected keyword.

In addition to these two types of initial condition files, there also exists a *sounding_rad.dat* file, located in the *DATA* directory, and required to run MIMICA with interactive radiation (with *setenv RADIA TRUE* in *start*). The role of *sounding_rad.dat* is to describe the thermodynamic state of the atmospheric column extending between the top of the model domain (typically located between 1.5 and 45 km depending on the simulated configuration) and the top of the atmosphere. This is absolutely required by the radiative transfer model to calculate the exact solar insulation at the top of the model domain. Because the exact structure of the full atmosphere is generally not known at the precise location of the simulation, several default soundings are available (*.lay* files in *DATA*), representative of a standard atmosphere at various locations and various seasons. For example, *kmlw.dat* is representative of a mid-latitude winter atmosphere, while *ktrop.lay* is representative of the tropical atmosphere. In order to use one of the available profiles, it is enough to rename it *sounding_rad.dat*. Care must however be taken as the transition between the thermodynamic state at the top of the model domain and that of the standard atmosphere must be smooth enough for the radiation code to work successfully. Modifying the standard *sounding_rad.dat* file may be necessary in some instances to guarantee a smooth transition.

4 MIMICA outputs

4.1 Overview and standards

4.2 Time series

4.3 2D-3D complete outputs

4.4 slices

4.5 Vertical profiles

4.6 Restart files

5 Templates and examples