Turbulent vertical diffusion

- Physical parameterizations describe subgrid physical processes not accounted for by the fundamental equations of atmospheric motion.
- The dominant parameterization schemes are radiation, microphysics and turbulence parameterizations.
- A diffusive process describes the turbulent exchange of heat and humidity at the surface and the vertical transport in the lower atmosphere.

In the ICON model the quantity

\[
\frac{\partial \rho (x,t)}{\partial t} = \frac{\partial}{\partial z} \left( \frac{\rho (x,z,t) \frac{\partial \rho (x,t)}{\partial z}}{\partial z} \right)
\]

is used for the vertical diffusion, where \( X \) is a placeholder which may be identified with the dry static energy \( e \) or the specific humidity \( q \). \( K \) is the exchange coefficient, \( \Delta \) is the vertical turbulent flux of \( X \), \( \rho \) is the air density, \( t \) is time and \( z \) the vertical coordinate. [2]. Boundary conditions are needed to solve equation (1) over one entire atmospheric column. A zero flux condition is chosen at the top of the planetary boundary and bulk formulas for the surface sensible and latent heat fluxes are introduced for \( \Delta \) at the surface. Using a leapfrog time-stepping scheme (1) is solved numerically with discretization in vertical direction and time. An algorithm proposed by Richtmyer and Morton (1967) solves the system of equations.

Objectives of this work

In a first step, a GPU port of the turbulent diffusion is implemented and compared to a cache-optimized and MPI as well as OpenMP parallelized CPU code by means of runtime measurements. OpenACC compiler directives are used for this purpose. Since the computations are memory bandwidth limited, the execution performance can profit from higher memory bandwidths offered by GPUs. Performance models and benchmarks are conducted to evaluate the attained results. Subsequently, horizontal dimensions are removed from the fortran diffusion subroutines and ICON compiler-generated code performance is compared to the hand-written implementations. Measurements and evaluations are conducted using multiple compilers and resolutions to assess the impact of GPU memory latency hiding on compute performance.

The simulation domain is split into blocks of vertical columns for cache optimization purposes on the CPU. The value \( vdiff \) denotes the size of each block. In the table above, performance measurements of the turbulence upward and downward sweep for a grid consisting of 81920 columns are shown. The block size equals the vector size 81920 for the OpenACC implementation, enabling parallel computations of a full horizontal layer of the domain. Each layer is divided into multiple gangs of vectors containing 128 elements (default value for both compilers).

The simulation domain is split into blocks of vertical columns for cache optimization purposes on the CPU. The value \( vdiff \) denotes the size of each block. In the table above, performance measurements of the turbulence upward and downward sweep for a grid consisting of 81920 columns are shown. The block size equals the vector size 81920 for the OpenACC implementation, enabling parallel computations of a full horizontal layer of the domain. Each layer is divided into multiple gangs of vectors containing 128 elements (default value for both compilers).