MPTRAC: recent progress on Lagrangian transport simulations on GPUs

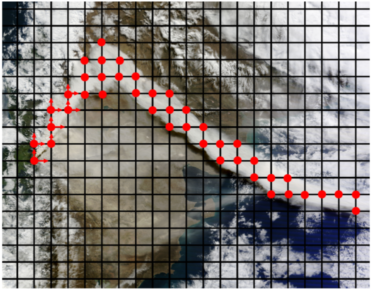
Lars Hoffmann

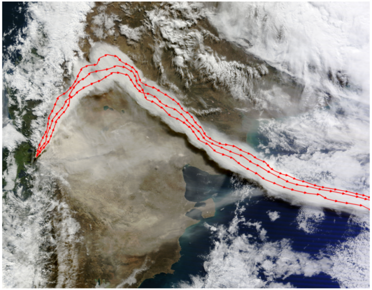
Jülich Supercomputing Centre (JSC)

NVIDIA Application Lab Workshop, Jülich, 21 – 22 June 2022

Overview on MPTRAC

- Massive-Parallel Trajectory Calculations (MPTRAC) is a Lagrangian transport model for the free troposphere and the stratosphere.
- Eulerian versus Lagrangian modeling approach:





- Lagrangian approach has low numerical diffusion and high spatial resolution, but it can be numerically rather costly...

Overview on MPTRAC

- Numerical integration of kinematic equation of motion:

$$\frac{d\mathbf{x}}{dt} = \mathbf{v}(\mathbf{x}, t)$$

- Diffusion and subgrid-scale winds are modeled as a Markov process, for instance:

$$v_i'(t + \Delta t) = r\, v_i'(t) + \sqrt{(1 - r^2)\sigma_{v_i}^2}\xi_i$$

- Additional modules:
  - boundary conditions
  - convection
  - dry deposition
  - exponential decay
  - hydroxyl chemistry
  - isosurface
  - sedimentation
  - wet deposition

Overview on MPTRAC

- Support for different meteo input data:
  - ECMWF data (ERA-Interim, ERA5, forecasts)
  - NASA MERRA and MERRA-2
  - NCEP/NCAR Reanalysis
  - NCEP GFS forecasts
- Variety of output data:
  - particle data
  - grid output
  - ensemble data
  - profile data
  - sample data
  - station data
  - verification data
  - gnuplot interface

Overview on MPTRAC

- Features of the code:
  - about 12,000 lines of C code
  - required libraries: GSL, netCDF
  - git repository: https://github.com/slcs-jsc/mptrac
  - documentation: doxygen, wiki
  - open source (GPL v3)
- Why is MPTRAC called a “massive-parallel” model?
  - MPTRAC features an MPI/OpenMP/OpenACC hybrid parallelization for use on state-of-the-art HPC systems.
  - Largest simulation with 1.446 million compute processes on Tianhe-2 supercomputer (Liu et al., 2020).

Porting to GPUs

Geosci. Model Dev., 15, 2731–2762, 2022
https://doi.org/10.5194/gmd-15-2731-2022

Geoscientific
Model Development
Open Access

Massive-Parallel Trajectory Calculations version 2.2 (MPTRAC-2.2): Lagrangian transport simulations on graphics processing units (GPUs)

Lars Hoffmann[1], Paul F. Baumeister[1], Zhongyin Cai[1,2], Jan Clemens[1,3], Sabine Griessbach[1], Gebhard Günther[3], Yi Heng[4], Mingzhao Liu[1], Kaveh Haghighi Mood[1], Olaf Stein[1], Nicole Thomas[3], Bärbel Vogel[3], Xue Wu[5,6], and Ling Zou[1]

[1]Jülich Supercomputing Centre, Forschungszentrum Jülich, Jülich, Germany
[2]Institute of International Rivers and Eco-security, Yunnan University, Kunming, China
[3]Institut für Energie- und Klimaforschung (IEK-7), Forschungszentrum Jülich, Jülich, Germany
[4]School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, China
[5]Key Laboratory of Middle Atmosphere and Global Environment Observation, Institute of Atmospheric Physics, Chinese Academy of Sciences, Beijing, China
[6]University of Chinese Academy of Sciences, Beijing, China

Correspondence: Lars Hoffmann (l.hoffmann@fz-juelich.de)

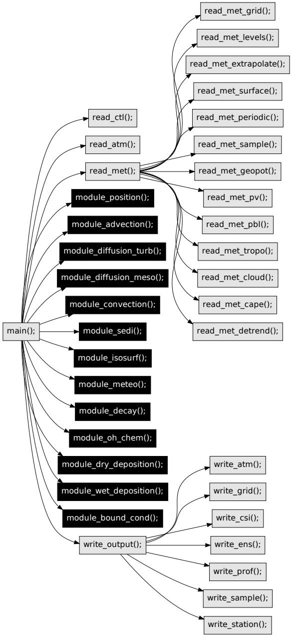
Received: 16 November 2021 – Discussion started: 1 December 2021
Revised: 8 March 2022 – Accepted: 10 March 2022 – Published: 5 April 2022

Abstract. Lagrangian models are fundamental tools to study atmospheric transport processes and for practical applications such as dispersion modeling for anthropogenic and natural emission sources. However, conducting large-scale Lagrangian transport simulations with millions of air parcels or more can become rather numerically costly. In this study, we assessed the potential of exploiting graphics processing units (GPUs) to accelerate Lagrangian transport simu-

providing a peak performance of 71.0 PFlop s^{-1}. As of June 2021, it is the most powerful supercomputer in Europe and listed among the most energy-efficient systems internationally. For large-scale simulations comprising 10^8 particles driven by the European Centre for Medium-Range Weather Forecasts' fifth-generation reanalysis (ERA5), the performance evaluation showed a maximum speed-up of a factor of 16 due to the utilization of GPUs compared to CPU-only runs

Porting to GPUs

- Physics calculations have been off-loaded to GPUs by means of OpenACC.
- File I/O functions need to interact with host CPU and have not been offloaded.
- Simulation mostly “lives” on GPUs. Frequent H2D or D2H data transfers are avoided.



Porting to GPUs

- OpenACC pragmas are used to off-load and distribute loops over air parcels over compute elements of the GPU:

Algorithm 1 Continued.

```
/* Function used to calculate air parcel trajectories... */
void module_advection(met_t * met0, met_t * met1, atm_t * atm, double *dt) {

  /* Loop over air parcels... */
  #ifdef _OPENACC
  #pragma acc data present(met0,met1,atm,dt)
  #pragma acc parallel loop independent gang vector
  #else
  #pragma omp parallel for default(shared)
  #endif
  for (int ip = 0; ip < atm->np; ip++) {
    ...
    /* Interpolate meteorological data... */
    intpol_met_time_3d(met0, met1, p, lon, lat...);
    ...
    /* Get position of the mid point... */
    ...
    }
}

/* Function used to interpolate meteo data... */
#pragma acc routine (intpol_met_time_3d)
void intpol_met_time_3d(met0, met1, p, lon, lat...) {
  ...
```

Porting to GPUs

- OpenACC pragmas for explicit data transfers:

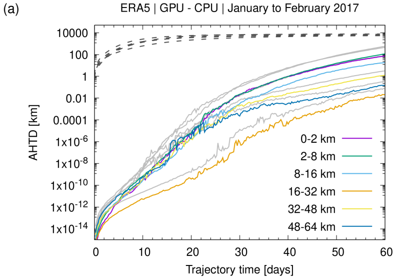
Algorithm 1 GPU porting of MPTRAC by means of OpenACC pragmas.

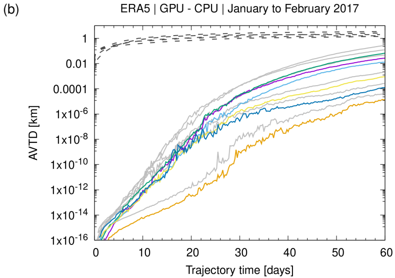
```
/* Main function of the MPTRAC model... */
int main(int argc, char *argv[]) {
  ...
  /* Initialize model run... */
  ...
  /* Create data region on GPUs... */
  #pragma acc enter data create(ctl,atm[:1],met0[:1],met1[:1])
  ...
  /* Read initial air parcel data... */
  read_atm(filename, &ctl, atm);
  #pragma acc update device(atm[:1],ctl)
  ...
  /* Loop over timesteps... */
  for (t = t_start; t < t_stop; t += dt) {
    ...
    /* Read meteorological data... */
    get_met(&ctl, t, &met0, &met1);
    #pragma acc update device(met0[:1],met1[:1],ctl)
    ...
    /* Calculate advection... */
    module_advection(met0, met1, atm, dt);
    ...
    /* Write output... */
    #pragma acc update host(atm[:1])
    write_output(dirname, &ctl, met0, met1, atm, t);
  }
```

Porting to GPUs

- In total, about 60 OpenACC pragmas are used to off-load computation and to implement data transfers.
- Some difficulties and additional changes:
  - cannot use non-GPU library functions (e. g., GSL)
  - implemented cuRAND for random number generation
  - some math functions are not implemented (e. g., fmod)
  - bug in earlier version of the PGI compiler (large structs)
- Advantages of using OpenACC:
  - easy to understand and to implement
  - can keep the same code for CPU and GPU

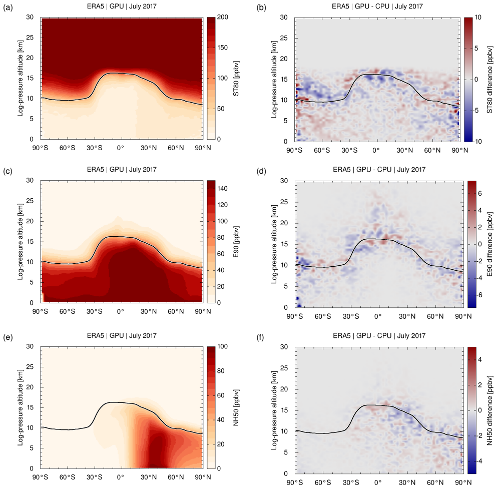
Porting to GPUs





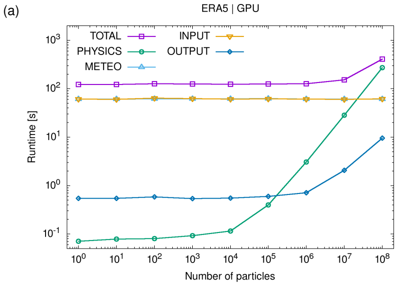
- Verification of transport deviations between CPU and GPU trajectories.
- CPU and GPU results are not bit-identical, but deviations during first 60 days are very small.
- Note: CPU binaries of GNU and PGI compiler are also not bit-identical.
- Differences likely due to different optimization flags at compile time.

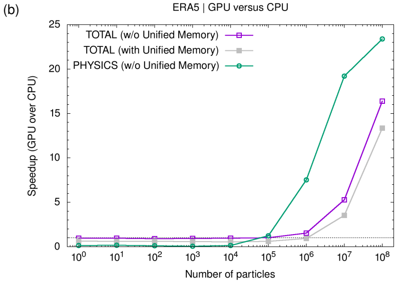
Porting to GPUs



- Verification of transport simulations using artificial tracers.
- CPU and GPU code use different random number generators (GSL vs cuRAND).
- CPU and GPU results agree within statistical noise.

Porting to GPUs

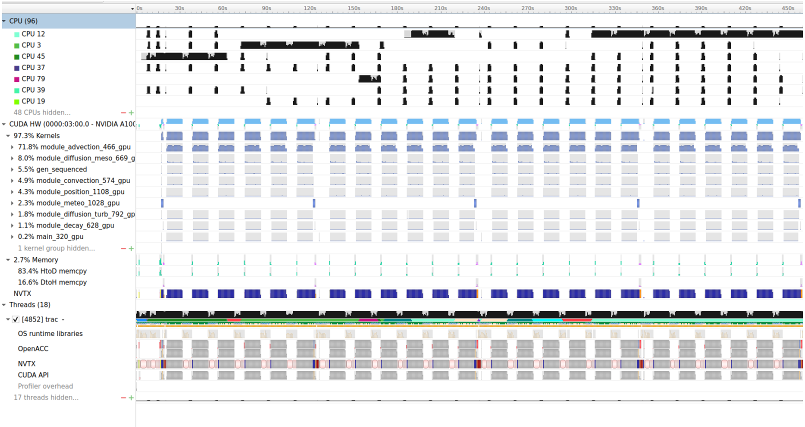




- GPU scaling test with respect to problem size (number of air parcels).
- GPU-over-CPU speed-up difficult to define. Here we compare a single GPU to 12 CPU cores on the JUWELS Booster.
- Speed-up larger 1x for 10^6 particles. Speed-up of 16x for 10^8 particles.

Porting to GPUs

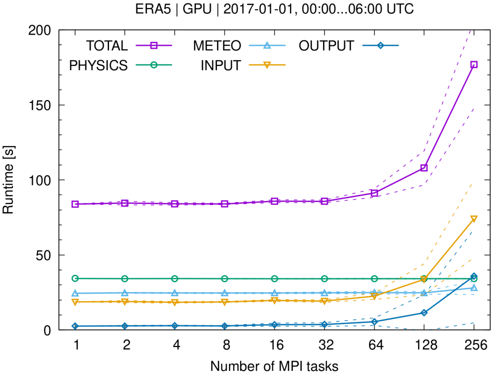
- Timeline analysis with NVIDIA Nsight Systems:



- Performance analysis shows good CPU utilization, but is also shows room for further improvement.

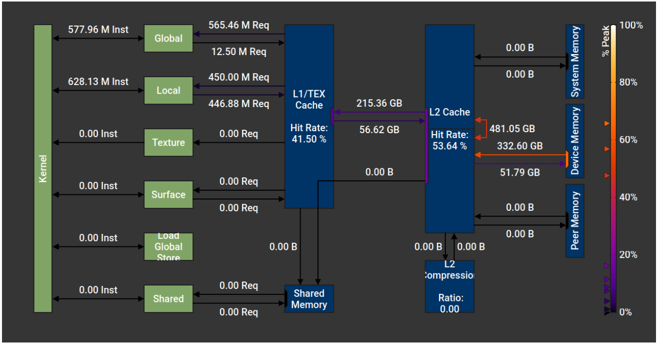
Porting to GPUs

- Multi-GPU usage is enabled via MPI parallelization:
  - Each MPI task runs a separate simulation with independent input/output data, utilizing one GPU device.
  - Weak scaling test shows good scaling up to 128 MPI tasks, but some issues with file-I/O for larger setups:



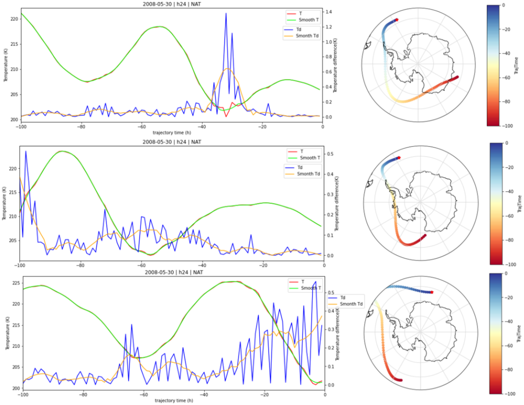
Ongoing and future work

- Recent collaboration with NVIDIA Application Lab team:
  - In-depth analysis to further improve GPU code of MPTRAC
  - Topic 1: implementation of asynchronous file-I/O, meteo data preprocessing, and physics calculations to overlap CPU and GPU workload
  - Topic 2: kernel analysis of advection code (SoA→AoS and particle sorting to optimize memory access)



Ongoing and future work

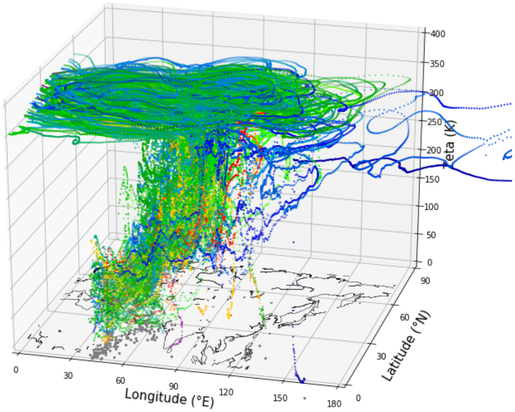
- Postdoc project by Ling Zou: Impact of temperature fluctuations on the occurrence of polar stratospheric clouds



- Technical work: Defining efficient workflows for analyzing large satellite data sets with MPTRAC.

Ongoing and future work

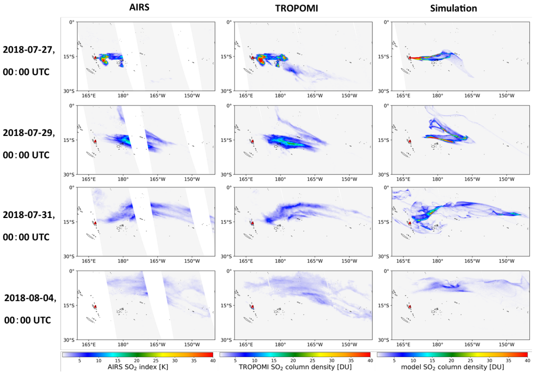
- PhD project by Jan Clemens: Identification of source regions of the Asian Tropopause Aerosol Layer



- Technical work: How to deal best with large meteo input data sets such as ECMWF's ERA5 (data compression)?

Ongoing and future work

- PhD project by Mingzhao Liu: Impact of hydroxyl chemistry, wet deposition, and convection on volcanic sulfur dioxide transport simulations



- Technical work: Implementing new chemistry and physics modules in MPTRAC for more realistic simulations.

Ongoing and future work

- Third-party projects using MPTRAC:
  - joint Sino-German DFG-NSFC project AeroTrac
  - HGF Joint Lab ExaESM activities on Lagrangian modeling
  - BMBF SCALEXA and WarmWorld proposals

Thank you!
Questions?