ecTrans: Performance Portable and Highly Scalable Spectral Transforms

Sam Hatfield, with thanks to Lukas Mosimann (Nvidia) and Paul Mullowney (AMD)

European Centre for Medium-Range Weather Forecasts; samuel.hatfield@ecmwf.int

ecTrans: open-source spectral transforms

- ECMWF's traditional model for **numerical weather prediction** is the **Integrated Forecasting System (IFS)**
- This **global atmospheric model** uses a **spectral transform technique** to solve the governing equations
- In the IFS this role is fulfilled by the **ecTrans library**

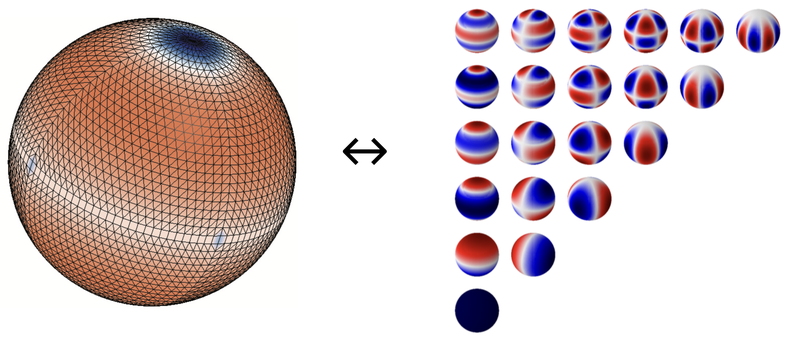

Figure 1: The spectral transform algorithm employed by ecTrans, for transforming between grid point space and spectral space.

- A spectral transform involves heavy **computation** (a Legendre and a Fourier transform) and **communication**
- It is therefore a useful benchmarking system for new architectures, in particular for predicting the performance of the IFS as a whole
- ecTrans was released as an open-source library in 2022, and since then has undergone a substantial **refactoring and optimisation**, thanks to **collaborations with both Nvidia and AMD**
- This collaboration was facilitated by ecTrans' **open-source license**
- Notably, support for GPUs has been added through a combination of **offloading directives** (OpenACC and OpenMP) and **numerical libraries** (hipBLAS / hipFFT and cuBLAS / cuFFT)
- Here we present the state of the art performance of ecTrans, focusing on **portability** and **scalability**

Single-node benchmarks

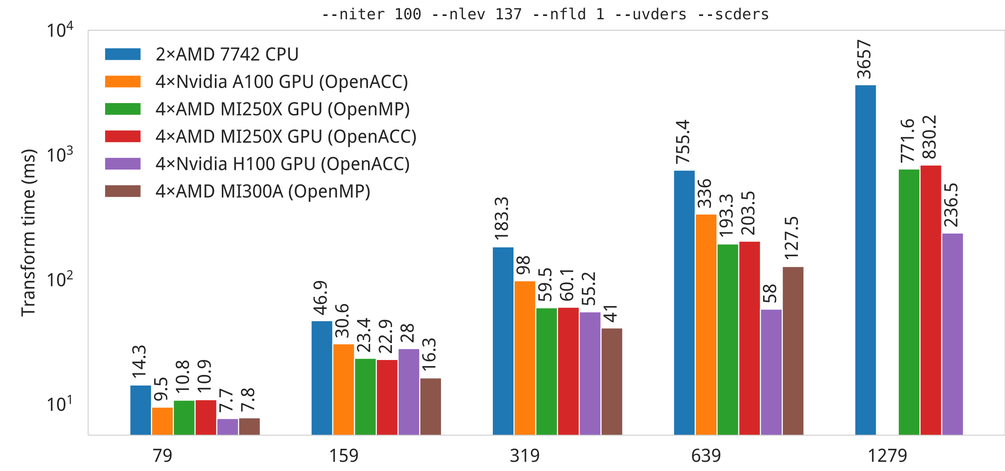

Figure 2: Benchmarks of ecTrans on a single node of several systems with a range of problem sizes, from TCO79 up to TCO1279.

- Figure 2 summarises the **single-node performance of ecTrans** on several platforms, including the current ECMWF HPC system
- This is compared with several GPU platforms, last gen (Nvidia A100 and AMD MI250X) and current gen (Nvidia H100 and AMD MI300A) (all four devices per node)
- Transforming a full complement of meteorological fields at TCO1279 resolution (~8 km) takes about **3.7 s** on one node of ECMWF's HPC but only **0.24 s** on four H100 GPUs
- Thanks to colleagues at AMD, we also have good performance on the MI250 and MI300 series through **OpenMP target directives**, even supporting the **new AMD compiler, ROCm AFAR** (for some cases at TCO1279 there was insufficient device memory)
- These benchmarks only test computational performance, however
- At scale, ecTrans also requires an efficient communication system, as one must perform a **sparse global communication twice on each transform**

Strong scalability on JUPITER

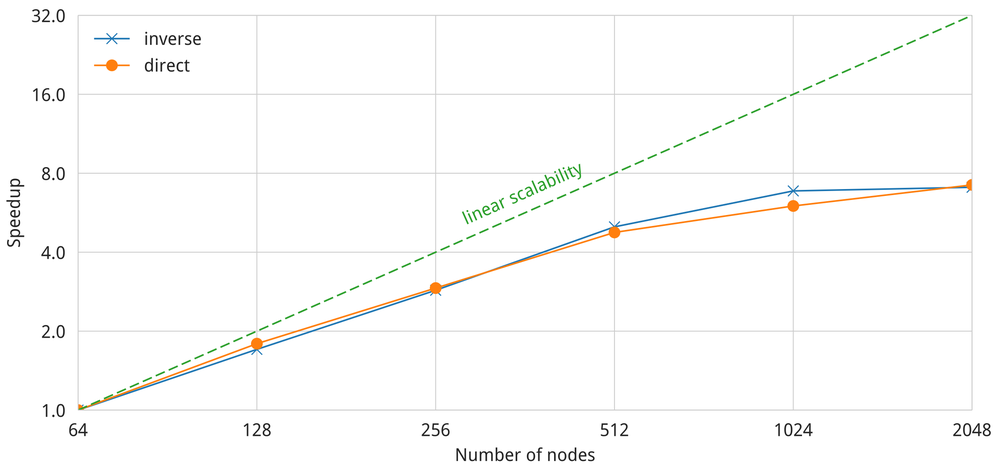

Figure 3: Strong scalability benchmark of ecTrans on JUPITER. The problem size is fixed at TCO7999 and the node count is varied.

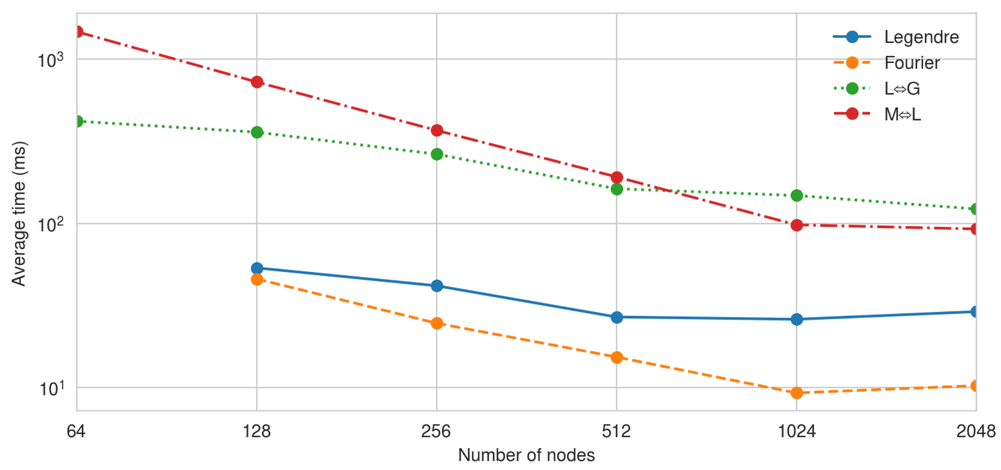

Figure 4: A breakdown of the strong scalability test of Figure 3 by transform steps: **computation** (Legendre and Fourier) and **communication** (L⇔G and M⇔L).

- Figure 3 illustrates strong scaling behaviour of ecTrans
- We performed benchmarks with a truncation of **TCO7999** which is about **1.45 km resolution** - extremely high resolution
- We performed these benchmarks on node counts ranging from **64** to **2048 nodes** on the **JUPITER supercomputer**
- We were granted early access thanks to the JUREAP program
- Strong scalability is reasonable from 64 to 512 nodes
- Above this, **scalability reaches a plateau**
- Figure 4 gives the scalability for individual components - **computation** (Legendre and Fourier transforms) and **communication** (L⇔G and M⇔L)
- Communication routines, especially M⇔L, scale **very well**
- Other routines scale poorly, notably the **Legendre transform**
- Nevertheless, the **communication routines still dominate** the overall wall time, and should be the target of any optimisation efforts on JUPITER going forward

Further ideas

- **Low-precision (16-bit) communication:** the MPI standard defines an MPI_REAL2 type which could be used for communication buffers, effectively doubling bandwidth:
  - Few MPI implementations support MPI_REAL2
  - Additional packing would be required to transfer to/from buffers
  - Communication latency could still be a bottleneck
- **Overlapping communication / computation**:
  - The batched "fields" dimension would have to be split up into smaller chunks, losing the efficiencies of GPU batching
- **Explore NPRTRV / NPRTRW parameter space:**
  - ecTrans uses two different parallel decomposition strategies
  - It may be possible to reduce communication costs by reorganising the subcommunicator structure

Destination Earth Funded by the European Union Implemented by ECMWF esa EUMETSAT ECMWF