Spherical harmonic transforms up to degree 100,000 and beyond using distributed-memory systems

Blažej Bucha[1]

[1]Department of Theoretical Geodesy and Geoinformatics, Slovak University of Technology in Bratislava

Motivation

On the algorithmic level, spherical harmonic transforms can nowadays be conducted up to almost any degree, say, tens of thousands and beyond. Hardware-related challenges emerge, however, when the truncation degree exceeds, say, 100,000. For instance, to perform a degree-100,000 spherical harmonic transform, be it forward

$$\bar{f}_{nm} = \frac{1}{4\pi}\frac{R}{\mu}\left(\frac{r}{R}\right)^n \iint_\sigma f(r,\varphi,\lambda)\,\bar{Y}_{nm}(\varphi,\lambda)\,\mathrm{d}\sigma \qquad (1)$$

or backward

$$f(r,\varphi,\lambda) = \frac{\mu}{R}\sum_{n=0}^{100{,}000}\left(\frac{R}{r}\right)^{n+1}\sum_{m=-n}^{n}\bar{f}_{nm}\,\bar{Y}_{nm}(\varphi,\lambda), \qquad (2)$$

it is necessary to store in memory about 75 GBs of spherical harmonic coefficients $\bar{f}_{nm}$ and about 149 GBs of the signal $f(r,\varphi_i,\lambda_j)$ to be analyzed/synthesized at the Gauss–Legendre grid.

To conduct such high-degree transforms, we employ the concept of distributed-memory computing using **MPI**. Our implementation is available through **CHarm**, a C/Python library for high-degree spherical harmonic transforms (`https://www.charmlib.org`).

Besides MPI, CHarm is parallelized using **OpenMP** (parallelization within shared-memory systems) and **SIMD** (vector CPU instructions executed by a single CPU core). The three techniques (MPI, OpenMP and SIMD) can (and should!) be combined simultaneously in CHarm for best performance (Fig. 1).

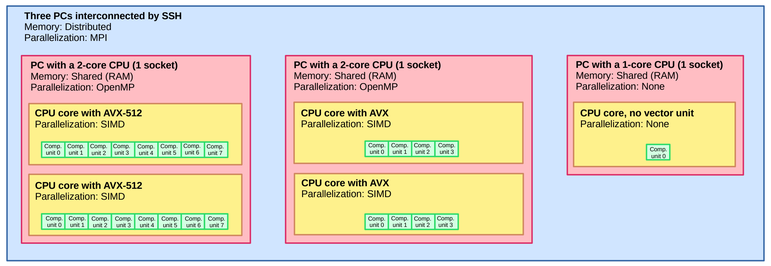

Figure 1. Three parallelization schemes used in CHarm: MPI (ideally one MPI process per CPU socket), OpenMP (ideally all cores within a CPU socket) and SIMD (vector instructions within a single core of a CPU socket). Shown is an example of three PCs interconnected by the SSH protocol. The number of cores (2, 2, 1 from left to right) and the SIMD vector length (8, 4, 1) vary to demonstrate that heterogeneous hardware can be combined. The three PCs perform, respectively, 16, 8 and 1 transforms simultaneously (25 in total). Not employing SIMD parallelization implies 5 simultaneous transforms only (what an excellent way of wasting money: buying an expensive CPU and then using only a fraction of its computing power...).

MPI

CHarm can distribute (i) spherical harmonic coefficients $\bar{f}_{nm}$, (ii) signal $f(r,\varphi_i,\lambda_j)$ and (iii) computing work among MPI processes, which may or may not have access to the same memory space. Using one MPI process per CPU socket (the red blocks in Fig. 1) and OpenMP threads within the socket (the yellow blocks) is usually the best strategy with CHarm.

In spherical harmonic transforms, **all signal data $f(r,\varphi_i,\lambda_j)$ are needed to compute one coefficient $\bar{f}_{nm}$ and vice versa**. When evaluating (1) and (2) up to high degrees, the amount of data transferred is therefore immense. Considering that computing hardware may be heterogeneous (e.g., varying RAM capacity, varying number of CPU cores, varying SIMD vector length), CHarm allows user-defined distribution of $\bar{f}_{nm}$ in any chunks of harmonic orders m. Similarly, the signal data $f(r,\varphi_i,\lambda_j)$ can be split in latitudinal chunks i.

All functions in CHarm that support MPI parallelization are collective communication functions. They must therefore always be called by all participating MPI processes.

The MPI standard version 3.0 (September 2012) or newer is required by CHarm.

OpenMP

OpenMP distributes computing work among threads that have access to the same shared memory space. This parallelization technique can be used, for instance, on a desktop PC or a computing node of a HPC cluster, where each thread (e.g., CPU core) has access to the same memory in RAM.

SIMD

SIMD parallelization employs vector CPU instructions. These are available, for instance, on the popular x86_64 and AArch64 architectures. The most intensive number crunching parts of CHarm were handwritten using SIMD. Supported SIMD extensions are SSE4.1, AVX, AVX-2, AVX-512 (all x86_64) and NEON (AArch64). In grid-wise computations, CHarm can thus assign one grid latitude parallel to one lane of the SIMD vector (the green *Computing Unit* in Fig. 1). **With AVX-512, a single CPU core processes 8 latitude parallels simultaneously**, boosting the performance by a factor of $\sim$4 (the peak theoretical factor is 8).

Experiments

Data

- Random spherical harmonic coefficients from the interval $[-1.0, 1.0]$

HPC cluster (distributed-memory)

- CPU: 6-core Intel Xeon X5670 2.93 GHz
  - GFLOPS (SGEMM): 112.82
  - SIMD: SSE4.1
  - 2 sockets per node
- RAM: 2 x 24 GB per node
- Nodes: 8
- Network: 2 x 10Gb/s Ethernet with throughput 2 x 640Gb/s

Consumer PC (shared-memory)

- CPU: 32-core AMD Ryzen Threadripper 3970X
  - GFLOPS (SGEMM): 1822.0 (as powerful as all 16 CPUs at the HPC cluster combined)
  - SIMD: AVX (has also AVX2, but it was not used)
- RAM: 256 GB

Scan for code example combining CHarm and MPI

Performance

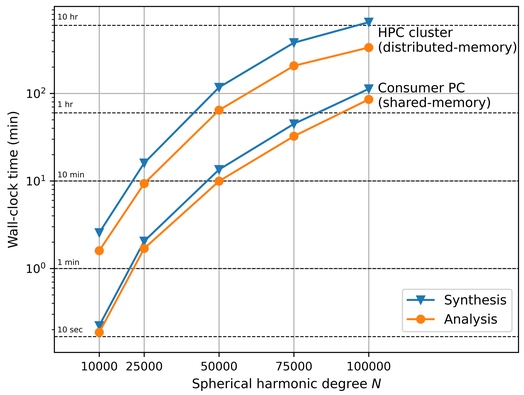

Figure 2. Performance of spherical harmonic synthesis and analysis as a function of series truncation degree N. Gauss–Legendre point grids of dimensions $(N+1)\times(2N+2)$ were used in the experiments.

In terms of performance, the data transfer is clearly the bottleneck with distributed-memory computations. The negative effects due to the data transfer can partially be relaxed by properly setting the `charm_glob_sha_block_lat_multiplier` and `charm_glob_shs_block_lat_multiplier` global variables.

Accuracy

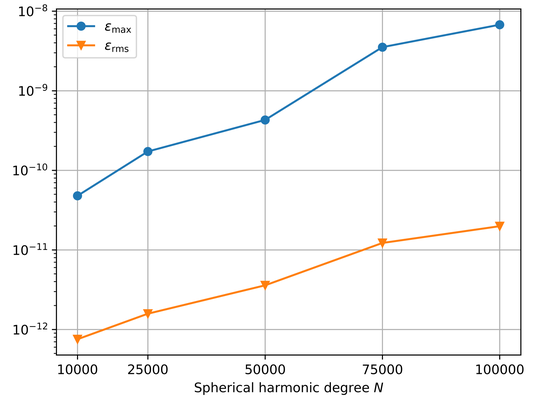

Figure 3. Maximum errors $\varepsilon_{\max} = \max_{n,m}|\bar{f}_{nm} - \bar{f}_{nm}^{(\mathrm{ref})}|$ and root mean square errors $\varepsilon_{\mathrm{rms}} = \sqrt{\frac{2}{(N+1)(N+2)}\sum_{n,m}\left(\bar{f}_{nm} - \bar{f}_{nm}^{(\mathrm{ref})}\right)^2}$ of the recovered coefficients $\bar{f}_{nm}$ in closed-loop experiments.

Applications

- **Topography modelling.** The Earth's and Moon's topographies expanded up to degree 100,000 offer 200-m and 55-m spatial resolutions, respectively. For a comparison, lunar topography is presently known at $\sim$30-m resolution thanks to LOLA.
- **Spectral gravity forward modelling.** Topography truncated at finite non-zero harmonic degree generates full-spectrum gravitational field. Thus, to accurately describe the gravitational field of, say, a degree-10,800 Earth's topography, the truncation degree of the implied potential should be as high as reasonably possible, say, 108,000 or so.

Summary

- Preferred should be shared-memory computations whenever possible; if this is not an option, distributed-memory transforms are viable alternative
- Communication between MPI processes should be reduced as much as possible by properly setting some global CHarm variables and by launching applications using one MPI process per CPU socket (trade-off between cache misses and data transfer)
- Code available at `https://www.charmlib.org`

Acknowledgements

The computations were performed at the HPC centre at the Slovak University of Technology in Bratislava, which is a part of the Slovak Infrastructure of High Performance Computing (SIVVP project, ITMS code 26230120002, funded by the European region development funds, ERDF).

The poster template is due to `https://github.com/anishathalye/gemini`.

Funding

Funded by the EU NextGenerationEU through the Recovery and Resilience Plan for Slovakia under the project No. 09I03-03-V04-00273.