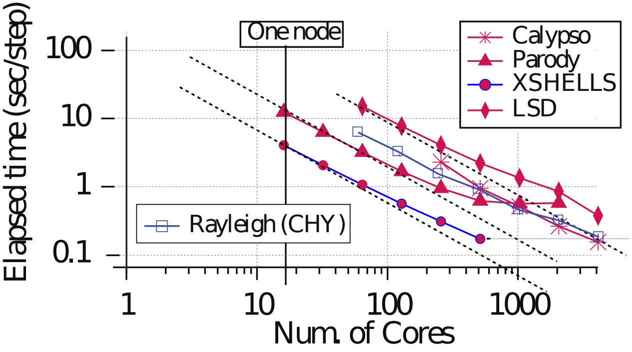
Geodynamo Simulations with XSHELLS + SHTns

Nathanaël Schaeffer

ISTerre / CNRS / Université Grenoble Alpes, Grenoble

Geodynamo Benchmarking Workshop, Boulder, 5 January 2015

Performance benchmark (AGU 2014)



For this test case ($NR = 512$, $L_{max} = 255$):

- At given number of cores, **XSHELLS is at least 3 times faster**.
- For shortest elapsed time, **XSHELLS needs 8 times less ressources**.

Outline

Introduction

The goal was to use spherical harmonics to time-step Navier-Stokes and related equations in spherical geometry.

Why Spherical Harmonics ?

- Advantage of spectral methods
- Don't need to solve for magnetic field in the insulator.
- Strongly reduces the number of variables to solve for.

Introduction

The goal was to use spherical harmonics to time-step Navier-Stokes and related equations in spherical geometry.

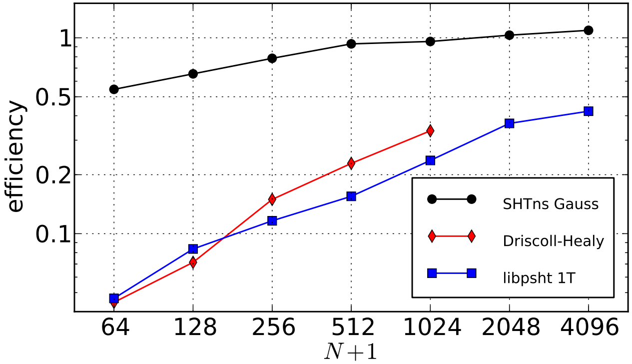
Why Spherical Harmonics ?

- Advantage of spectral methods
- Don't need to solve for magnetic field in the insulator.
- Strongly reduces the number of variables to solve for.

Spherical Harmonics Transform (SHT) is the bottleneck

- Is it possible to use fast algorithm ?
- Is it possible to otherwise improve the SHT ?

Comparison of Spherical Harmonic transform libraries



- Fast transforms are still much slower than carefully optimized Gauss-Legendre algorithm.
- They even stop working at some point because of memory requirements.

Outline

SHTns: outstanding features

Ok, we know it's **blazingly fast**, but why ?

- **Matrix-free algorithm**: computing recurrence on-the-fly is faster than reading from memory [unless matrix-matrix product is used], and you save a lot of memory too.
- **Hand vectorized**, yet easily portable (currently supports Intel SSE2, AVX, AVX2+FMA, MIC ; IBM Blue Gene/Q)
- SHTns perf scales with microarchitecture: x2 w/SSE2, x4 w/AVX, x8 w/AVX2+FMA... x16 w/AVX512 ?

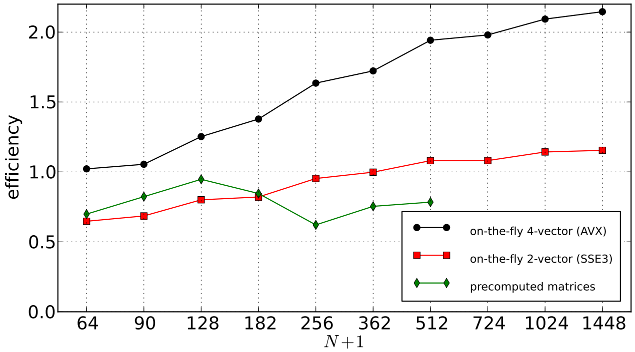
  It also means the gap between classic implementations and SHTns will increase.
- Efficient OpenMP parallelization (not yet used by XSHELLS)
- Reach 80% to 90% of peak performance on Intel SandyBridge.

N. Schaeffer, Efficient Spherical Harmonic Transforms aimed at pseudo-spectral numerical simulations, Gcubed, 2013.

SHTns: other interesting features

- both **scalar and vector** transforms.
- **accurate** up to spherical harmonic degree $\ell = 16383$ (at least).
- **SHT at fixed m** (without fft, aka Legendre transform).
- spatial data can be stored in latitude-major or longitude-major arrays.
- various normalization conventions.
- can be used from **Fortran, c/c++, and Python** programs.
- **free software** : `https://bitbucket.org/nschaeff/shtns`

On-the-fly vs Precomputed matrix



- Better performance for big sizes or large vector instructions (like AVX).
- Low memory requirement: the same as the memory needed to store a spherical harmonic decomposition.

SHTns: short summary

- Matrix-Vector product is too slow.
- Instead of reading a big matrix from memory, recomputing the elements when needed can be a lot faster.
- SHTns squeezes every bit of computing power of nowadays computers.
- With larger vector units, the advantage fo SHTns will increase.
- I wonder if transforming many shells together using matrix-matrix product could be competitive?
- Note that on Xeon Phi, the FFT is now the bottleneck, not the Legendre transform!

https://bitbucket.org/nschaeff/shtns

N. Schaeffer, Efficient Spherical Harmonic Transforms aimed at pseudo-spectral numerical simulations, Gcubed, 2013.

Outline

MHD forced by Boussinesq convection

XSHELLS time-steps the following equations:

$$\partial_t \mathbf{u} + (2\mathbf{\Omega_0} + \nabla \times \mathbf{u}) \times \mathbf{u} = -\nabla p^* + \nu \Delta \mathbf{u} + (\nabla \times \mathbf{b}) \times \mathbf{b} + c \nabla \Phi_0$$

$$\partial_t \mathbf{b} = \nabla \times (\mathbf{u} \times \mathbf{b}) + \eta \Delta \mathbf{b}$$

$$\partial_t c + \mathbf{u}.\nabla (c + C_0) = \kappa \Delta c$$

$$\nabla \mathbf{u} = 0 \qquad\qquad \nabla.\mathbf{b} = 0$$

- **Spherical** shell geometry
- Fields expanded using $\mathbf{u} = \nabla \times (T\mathbf{r}) + \nabla \times \nabla \times (P\mathbf{r})$
- Spherical Harmonic expansion, with **transforms done by SHTns**
- Finite differences in radius
- Semi-implicit scheme: diffusive terms are time-stepped using Cranck-Nicholson while other terms are treated explicitly with Adams-Bashforth scheme.

The XSHELLS code

hybrid MPI/OpenMP parallelization

- **Increase Data Locality**: work shell by shell for spatial terms (do not compute the whole spatial fields at once: lower memory required and faster).
- Distribute shells to MPI processes (at least 2 shells/process).
- **No transposition** required (no `MPI_Alltoall()`), communication with nearest neighbor only.
- Use OpenMP within these processes (currently up to 1 thread/shell) (but OpenMP in the angular directions is planned for near future.)
- max resolution so far: 2688 x 1344 x 1024 @ 1024 cores (5.2 sec/step)

The XSHELLS code

hybrid MPI/OpenMP parallelization

- **Increase Data Locality**: work shell by shell for spatial terms (do not compute the whole spatial fields at once: lower memory required and faster).
- Distribute shells to MPI processes (at least 2 shells/process).
- **No transposition** required (no `MPI_Alltoall()`), communication with nearest neighbor only.
- Use OpenMP within these processes (currently up to 1 thread/shell) (but OpenMP in the angular directions is planned for near future.)
- max resolution so far: 2688 x 1344 x 1024 @ 1024 cores (5.2 sec/step)

For reference:
1995: Glatzmaier and Roberts, 64 x 32 x 49 (the pioneers, with hyperviscosity)
2008: Kageyama et. al., 2048 x 1024 x 511 (Yin-Yang grid, E=1e-6, Re=700, Pm=1)
2009: Sakuraba and Roberts, 768 x 384 x 160 (Chebychev E=2e-6, Re=650, Pm=0.2)
2014: Hotta et. al., 4096 x 2048 x 512 (solar dynamo, Yin-Yang grid)

Why is XSHELLS so fast? – non-linear terms

Design choice: **optimize the computation of non-linear terms first**.

Trick 1: Use SHTns

J. Aubert plugged SHTns into Parody and observed a performance increase by a factor of about 2 for the whole code.
$\Rightarrow$ lower memory required, faster, ready for future architectures.

Trick 2: Increase Data Locality

Work shell by shell when computing spatial terms (do not compute the whole spatial fields at once).
$\Rightarrow$ lower memory required and significantly faster.

Why is XSHELLS so fast? – linear solver

The linear solver is computationally very cheap. It should deal with the data as it is arranged for the non-linear terms to be most efficient.

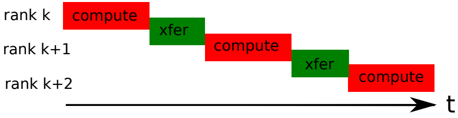
Trick 3: Avoid transpositions and large copies

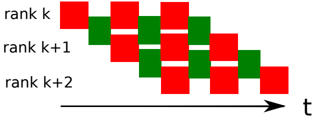
- Transposition is always slow (with and without MPI communications)
- Copying large amounts of data can also be quite slow.

Problem: the forward (backward) substitutions depend on the data computed by previous (next) processes.

Why is XSHELLS so fast? – blocked linear solver

Solution to data dependency in the linear solver: split shells into independent blocks.





Trick 4: Use self-tuning

- The optimal number of blocks is system dependent: trade off between wait time and transfer overhead.
- Find the best number of blocks at startup.
- 30 folds observed performance increase compared to naive solver.

Outline

The model

- thermochemical convection (codensity, 75% chemical driving, Aubert *et al* 2009);
- including some secular cooling effect;
- no-slip, and fixed flux homogeneous boundary conditions
- **high rotation rate, low viscosity**
- **strong forcing** (more than 4000 times critical)

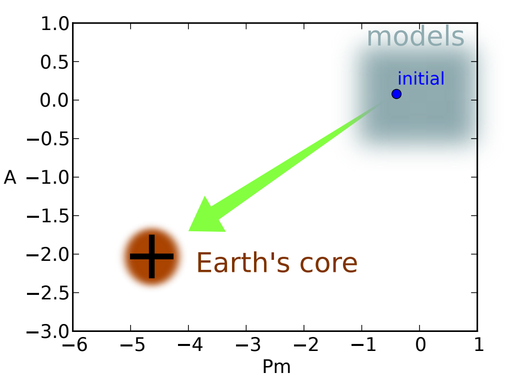
with:

1. Ekman number $E = \nu / D^2 \Omega$
2. Rayleigh number $Ra = \Delta T \alpha g D^3 / \kappa \nu$
3. Magnetic prandtl number $Pm = \nu \mu_0 \sigma$
4. (Thermal) Prandtl number $Pr = 1$.

The simulations

The idea

- Keep super-criticality and Rm fixed.
- go to more Earth-like A and Pm.

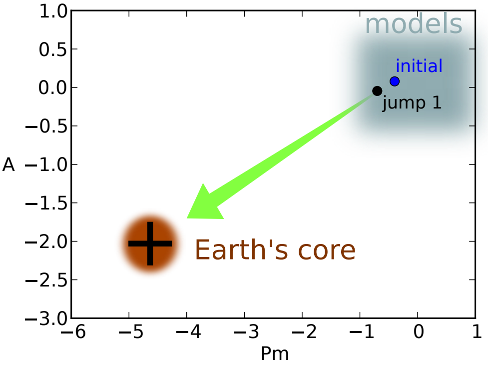


- **initial**: $E = 10^{-5}$, $Pm = 0.4$, $Ra = 6\,10^{10}$ $\Rightarrow A = 1.5 \quad F_\nu = 47\%$

The simulations

The idea

- Keep super-criticality and Rm fixed.
- go to more Earth-like A and Pm.

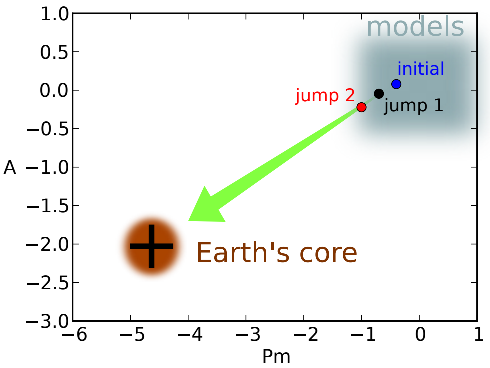


- **initial**: $E = 10^{-5}$, $Pm = 0.4$, $Ra = 6\,10^{10}$ $\Rightarrow A = 1.5 \quad F_\nu = 47\%$
- **jump 1**: $E = 10^{-6}$, $Pm = 0.2$, $Ra = 1.2\,10^{12}$ $\Rightarrow A = 0.61 \quad F_\nu = 24\%$

The simulations

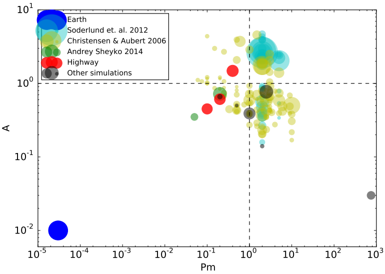
The idea

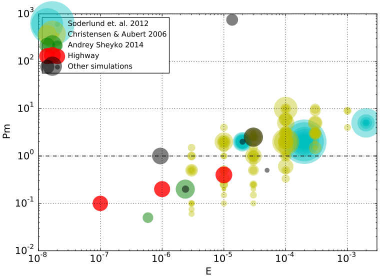
- Keep super-criticality and Rm fixed.
- go to more Earth-like A and Pm.



- **initial**: $E = 10^{-5}$, $Pm = 0.4$, $Ra = 6\,10^{10}$ $\Rightarrow A = 1.5 \quad F_\nu = 47\%$
- **jump 1**: $E = 10^{-6}$, $Pm = 0.2$, $Ra = 1.2\,10^{12}$ $\Rightarrow A = 0.61 \quad F_\nu = 24\%$
- **jump 2**: $E = 10^{-7}$, $Pm = 0.1$, $Ra = 2.4\,10^{13}$ $\Rightarrow A = 0.45 \quad F_\nu = 17\%$

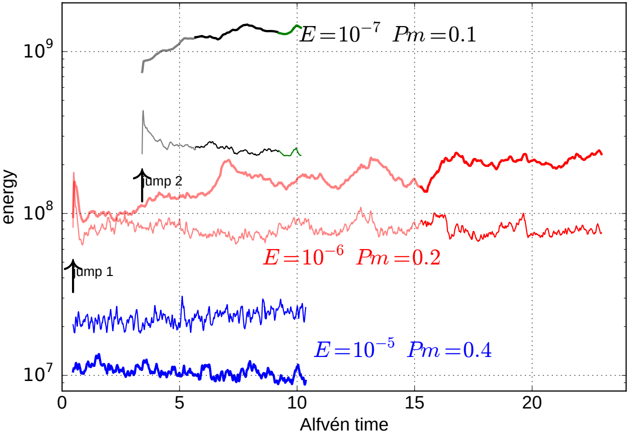
A selection of geodynamo simulations





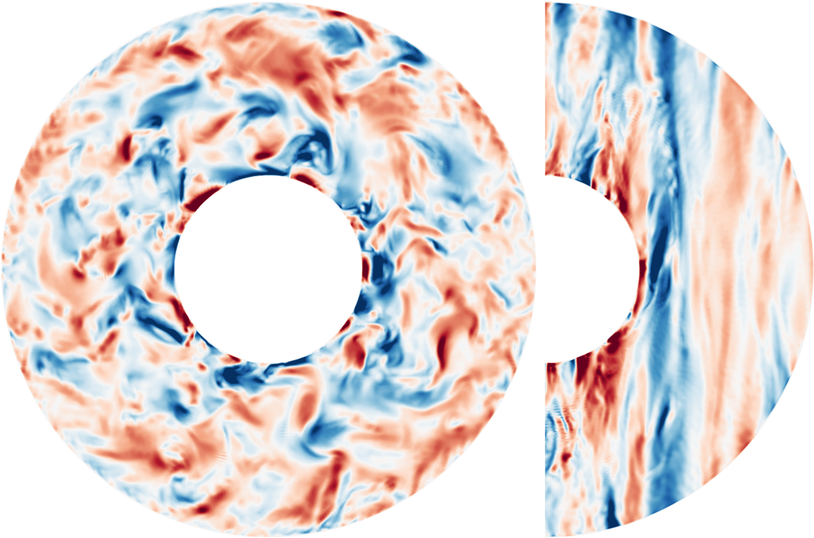
Surface of the discs is proportional to the magnetic Reynolds number (i.e. how strong the magnetic field generation is)

Energy vs Time



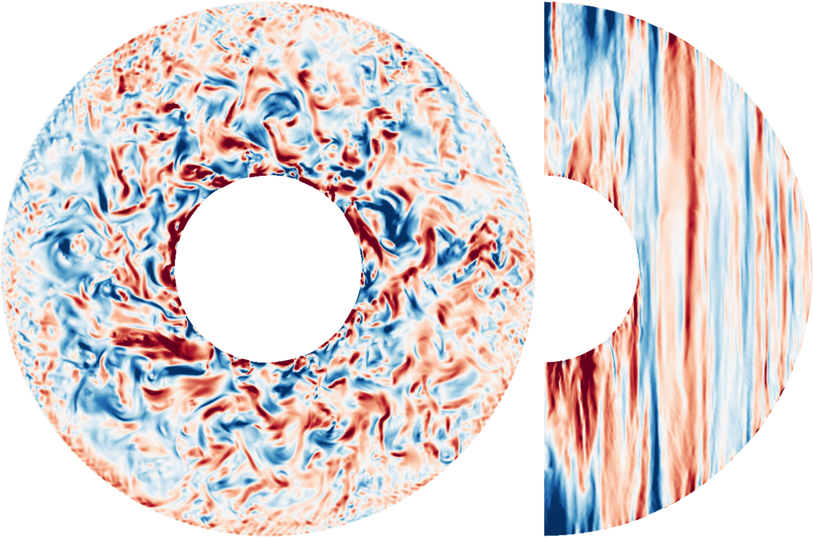
Jump 2 ran for 1.5% of a magnetic diffusion time, and it took about 7 months to compute on 512 cores, spending 2.5 million core hours.

Snapshot: initial U_ϕ ($E = 10^{-5}$, $Pm = 0.4$, $A = 1.5$)



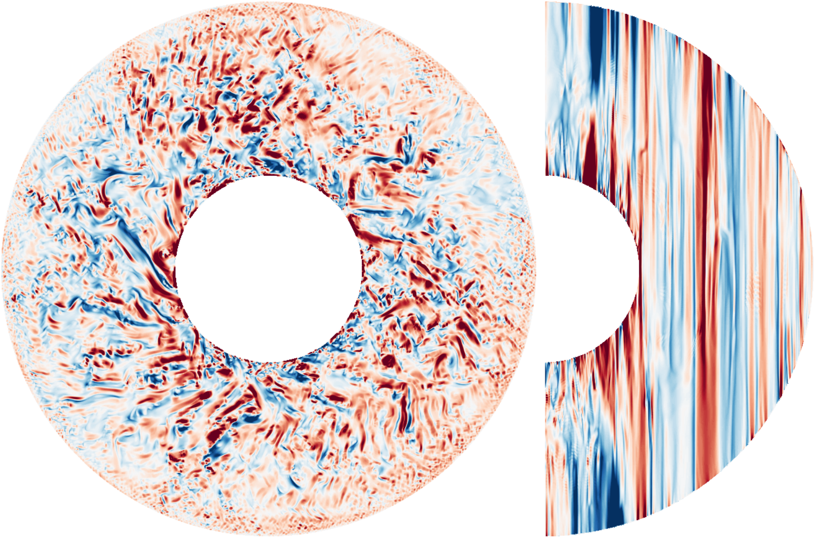
$NR = 224$, $L_{max} = 191$

Snapshot: jump 1 U_ϕ ($E = 10^{-6}$, $Pm = 0.2$, $A = 0.6$)



$NR = 512$, $L_{max} = 479$

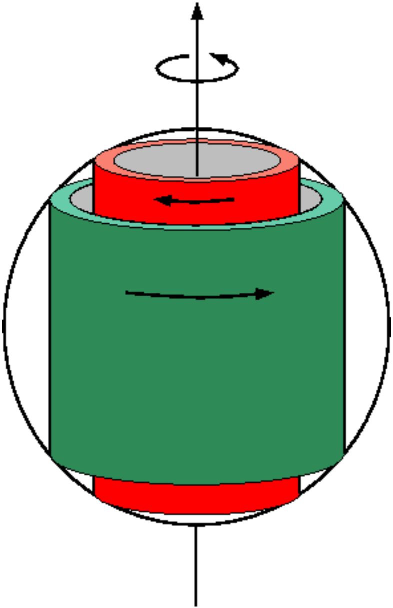
Snapshot: jump 2 U_ϕ ($E = 10^{-7}$, $Pm = 0.1$, $A = 0.45$)



$NR = 1024$, $L_{max} = 893$

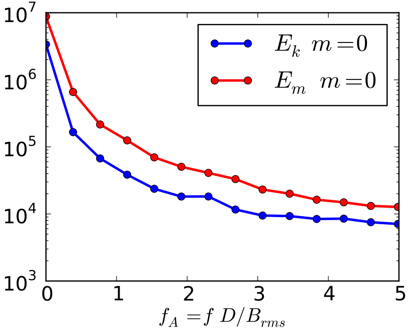
Torsional Afvén Waves in the core

- Alfvén waves constrained by rotation can only propagate as geostrophic cylinders.
- Their speed is related to the integral over z and ϕ of B_s^2.
- Measuring their speed gives information about the magnetic field inside the core.



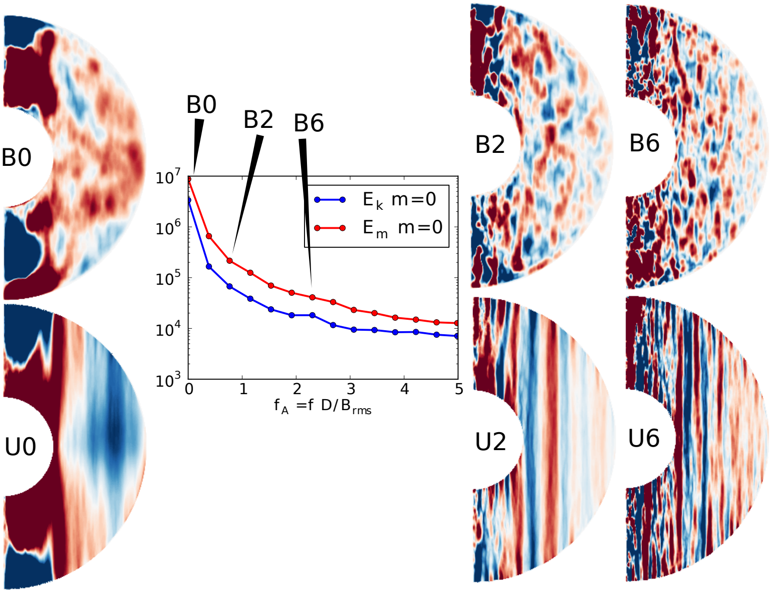
Jump 1: Fourier Analysis of axisymmetric fields

- We perform an FFT of the axisymmetric component over about 3 Alfvén times (defined with B_{rms}).
- Modal analysis similar to Figueroa *et al (2013)*



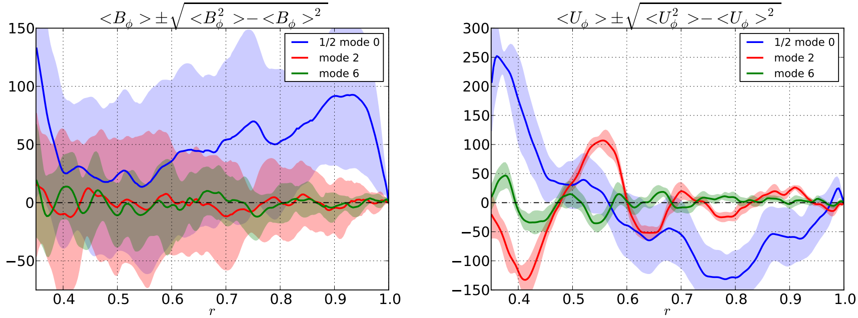
A. Figueroa *et al, Modes and instabilities in magnetized spherical Couette flow, JFM, 2013.*

Jump 1: Fourier modes of axisymmetric U_ϕ and B_ϕ



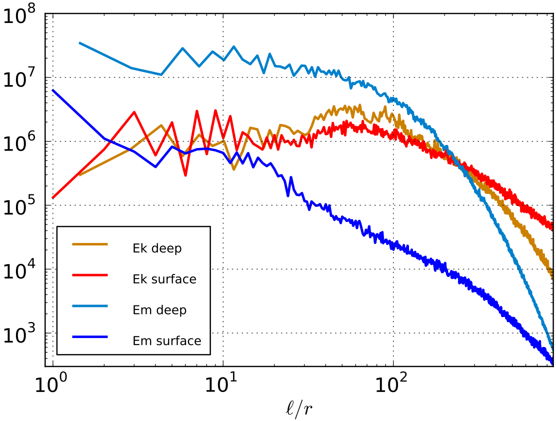
Jump 1: Fourier modes: z-averages and standard deviation

We quantify the z-invariance of the magnetic and velocity fields for mode 0 ($f_A = 0$), mode 2 ($f_A = 0.77$), and mode 6 ($f_A = 2.3$).



Evidence for **geostrophy of intermediate frequency modes** ($0.5 < T < 10$ years, axisymmetric)

Jump 2: spectra

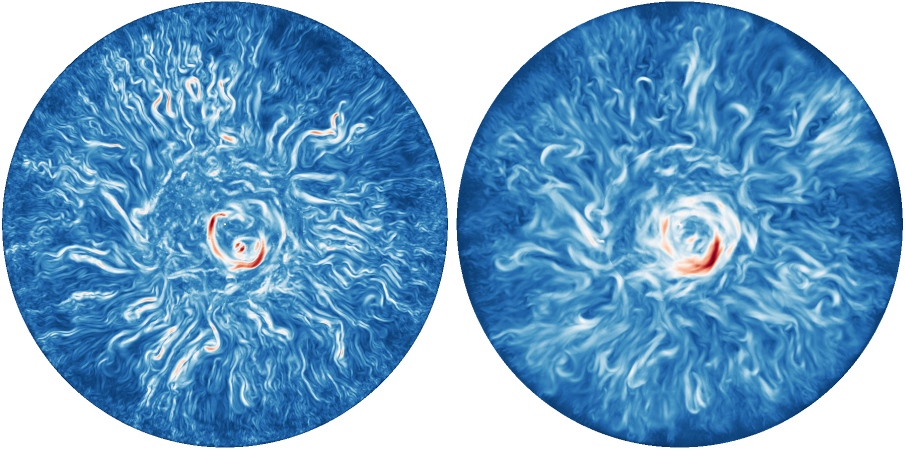


$E = 10^{-7}$
$Pm = 0.1$
$Ra = 2.4\,10^{13}$
$Rm = 600$
$A = 0.45$
$\Lambda = 1.2$
$F_\nu = 17\%$

$NR = 1024$
$L_{max} = 893$

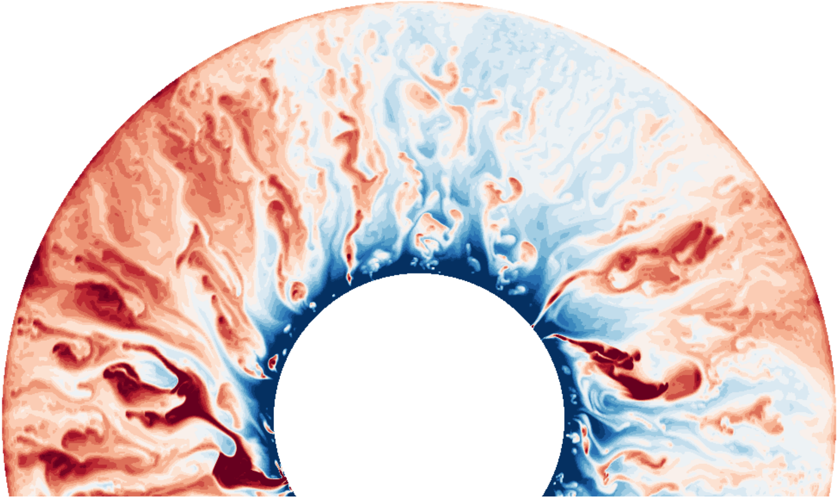
- Magnetic field dominates deep in the core but not near the surface.
- Velocity spectrum nearly flat at the surface but increasing deep down.

Jump 2: z-averaged energy densities



z-averaged equatorial energy densities, left: $< U_{eq}^2 >$, right: $< B_{eq}^2 >$.
$E = 10^{-7}$, $Pm = 0.1$, $Rm = 600$, $A \sim 0.45$.

Jump 2: Temperature field



Mean temperature of each shell has been removed.

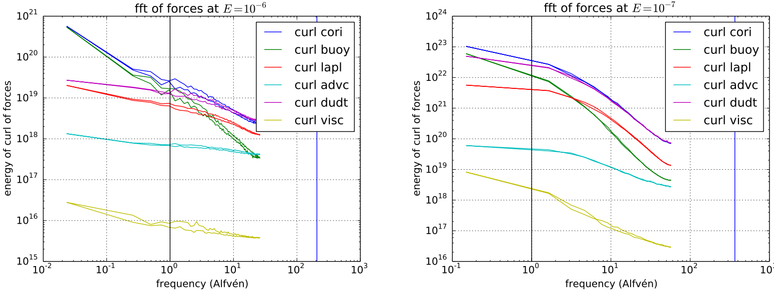
Force balance vs frequency

Method

- Fields up to lmax=30 stored periodically for significant time-spans.
- Post-processing of these provide force balances at different time-scales.
- Boundary layers removed.

Problem: large storage requirement for full fields (only up to lmax=30 here)

Force balance vs frequency (large scales only)



- Geostrophic balance removed by curl.
- For low frequency: Coriolis-Buoyancy balance.
- For high frequency: Coriolis-inertia balance (intertial waves).
- Laplace force overtakes Buoyancy at high frequency.
- contribution of small scales ignored.

High resolution simulation summary

Turbulence in the Earth's core (dynamo magnetic field + strong rotation) is not well understood.

As we go toward more turbulent simulations

- Velocity field peaks at smaller and smaller scales !
- Torsional waves are excited.
- Improved z-invariance.

Open questions:

- Can we realy forget about $u\nabla u$?
- How does the magnetic field affect the flow ?
- The data is available if someone wants to look.

Concluding remarks

XSHELLS:

- Very high raw performance.
- Scaling may be more limited than other approaches, but there is room for improvement.
- Expect to reach good scaling up to 8 to 16 threads/shell for large cases by June 2015.
- Free software: `https://bitbucket.org/nschaeff/xshells`

General conclusions:

- SHTns should be tried in other codes.
- Alternatively, there is libsharp (spin-weighted spherical harmonics).
- Don't rely too much on the compiler for code vectorization.
- Don't underestimate the cost of reading/writing to memory.

`https://bitbucket.org/nschaeff/shtns`

`http://sourceforge.net/projects/libsharp/`

Some numbers

	definition	initial	jump 1	jump 2	Earth's core
N_r		224	512	1024	
L_{max}		191	479	893	
Ek	$\nu/D^2\Omega$	10^{-5}	10^{-6}	10^{-7}	$3\,10^{-15}$
Ra	$\Delta T \alpha g D^3/\kappa\nu$	$6\,10^{10}$	$1.2\,10^{12}$	$2.4\,10^{13}$	10^{30} ?
Pm	ν/η	**0.4**	**0.2**	**0.1**	$3\,10^{-5}$
Pr	ν/κ	1	1	1	0.1 - 10
Rm	UD/η	**700**	**650**	**600**	2000 ?
A	$\sqrt{\mu\rho}U/B$	**1.5**	**0.6**	**0.45**	0.01
Re	UD/ν	1770	3240	5960	$2\,10^{8}$
Ro	$U/D\Omega$	0.018	$3.2\,10^{-3}$	$6\,10^{-4}$	$3\,10^{-6}$
Le	$B/\sqrt{\mu\rho}D\Omega$	0.012	$5\,10^{-3}$	$1.3\,10^{-3}$	10^{-4}
Λ	$B^2/\eta\Omega$	5.8	5.7	1.7	1 - 10
F_ν	$D_\nu/(D_\eta + D_\nu)$	47%	24%	**17%**	?
F_η	$D_\eta/(D_\eta + D_\nu)$	53%	76%	**83%**	?

Table 1: Various input and output parameters of our simulations, where D is the shell thickness, U the rms velocity and B the rms magnetic field.

SHTns Scalar Synthesis time comparison

size (ℓ_{max})	cpu 16c	mic	mic offload	tesla m2090	tesla m2090 (2q)
511	1.4	1.5		4.25	2.46
1023	8.9	7.2	16.3	19.3	11.0
2047	62	74.2	117	104.3	68.5
4095	446	296	885	709	547
8191		2050	6850		

Table 2: Time in milliseconds to complete a spherical harmonic synthesis on various devices and for various sizes. **cpu 16c** is a 16 core 2.7GHz SandyBridge platform. **tesla m2090** with OpenCL (synthesis only) includes the memory transfer (30% to 40%). **2q**: using transfer compute overlap (2 opencl queues). Note that for 511 and 1023, the best times were obtained with "transposed" fft on the mic.

- MIC offload is the slowest (so far, could probably be better)
- MIC native is often fastest (strongly depends on data layout and fft performance !)