Computational optimization in high resolution gravity field modelling by the finite volume method

Macák Marek

joined work with R. Čunderlík, K. Mikula and Z. Minarechová.

Department of Mathematics and Descriptive Geometry

Slovak University of Technology, Faculty of Civil Engineering

Outline

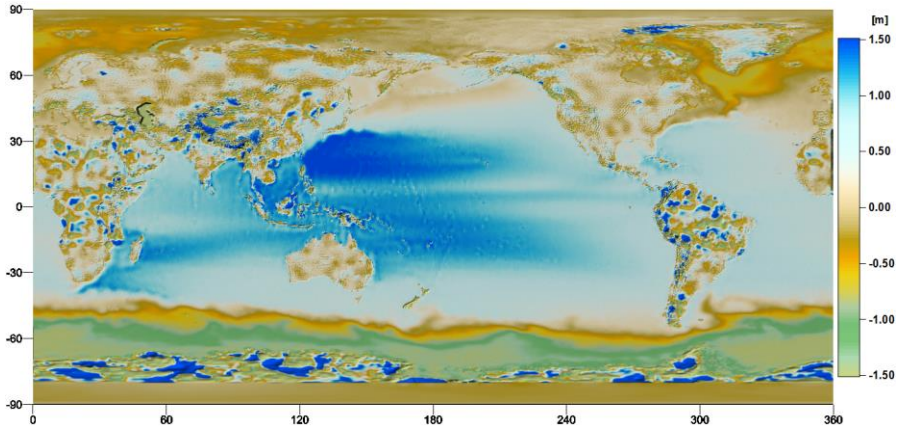
- Motivation
- Solution of the GBVP
  - Optimal linear solver
- Speedup of solution
  - Parallelization
  - Domain decomposition
  - Parallel domain decomposition
- Conclusion

Outline

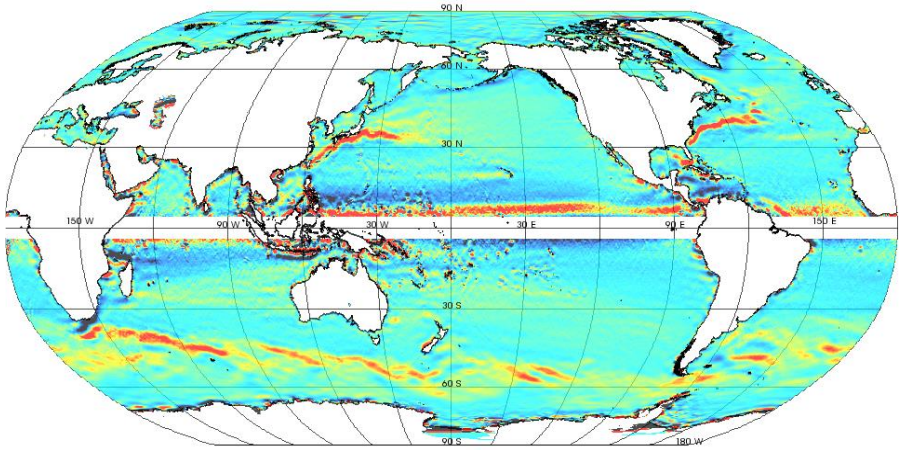
- Motivation
- Solution of the GBVP
  - Optimal linear solver
- Speedup of solution
  - Parallelization
  - Domain decomposition
  - Parallel domain decomposition
- Conclusion

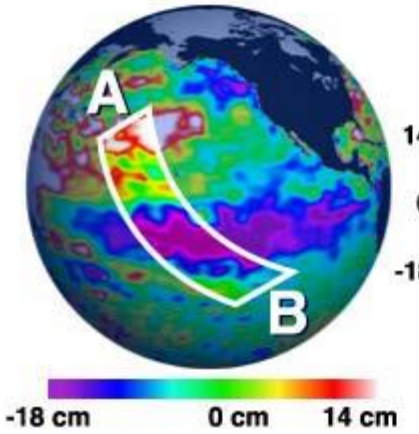
Motivation

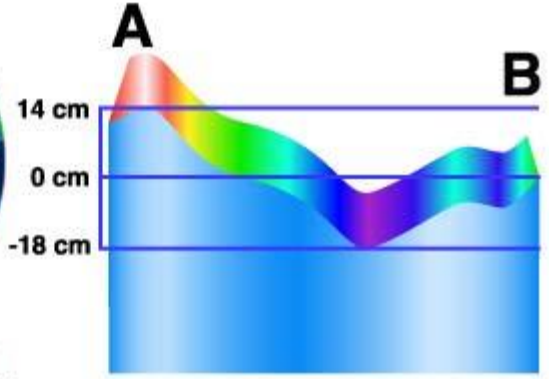
- Oceanography
  - Mean Dynamic Topography



  - Geostrophic Velocity







Outline

- Motivation
- Solution of the GBVP
  - Optimal linear solver
- Speedup of solution
  - Parallelization
  - Domain decomposition
  - Parallel domain decomposition
- Conclusion

Formulation of the boundary-value problem

- We consider the following oblique derivative boundary-value problem (BVP) in the bounded domain Ω :

$$\Delta T = 0 \; in \; \Omega,$$
$$\nabla T . \vec{s} = \delta g \; on \; \Gamma_D,$$
$$T = T_{SAT} \; on \; \Gamma_U,$$

where Ω is the exterior space above the Earth, T is the disturbing potential, Γ_D is the bottom boundary of Ω and Γ_U is the boundary of Ω-Γ_D.

- Using numerical method FEM, FVM or BEM we obtain the system matrix and the right-hand side vector with nonzero entries.

Size of computational domain

- *2D computational domain (BEM)*

Resolution	Element size on equator	Elements on the Earth surface
1° x 1°	~111km	360 x 180
1’ x 1’	~1.85km	21 600 x 10 800

21 600 x 10 800=*233 280 000*

- *3D computational domain (FEM, FVM)*
  - *Satellite altitude ~200km =*

21 600 x 10 800 x 200 = 34 992 000 000

Matrix/Vector storage formats

- Full matrix: (i.e.: BEM) storage locations n^2
- Sparse matrix: (i.e.: FEM, FVM)
  - Compressed storage (Compress Row Storage,Compress Column Storage)
    - create 3 vectors: $Val, col\ ind, row\ ptr$
    - storage locations: $2nnz + n + 1$,
  - Diagonal storage (Compressed Diagonal Storage)
    - create D vectots (in FEM $D = 27$, in FVM with regular mesh $D = 9$)
    - storage locations: Dn

Memory requirements

Method	Number of unknowns	Memory for unknown values	Memory for matrix and right-hand side vector
BEM	233 280 000	~ 1.8 GB	~ 3 484 GB
FEM	34 992 000 000	~ 360 GB	~ 13 063 GB
FVM			~ 3 266 GB

Super computer / Our claster

Where:	HPC STUBA	Our department	Slovak science academy
Cores:	624	224	4 096
Memory:	2 501 GB	1 792 GB	31 232 GB
Processor:	Intel Xeon	AMD	IBM POWER 775

Outline

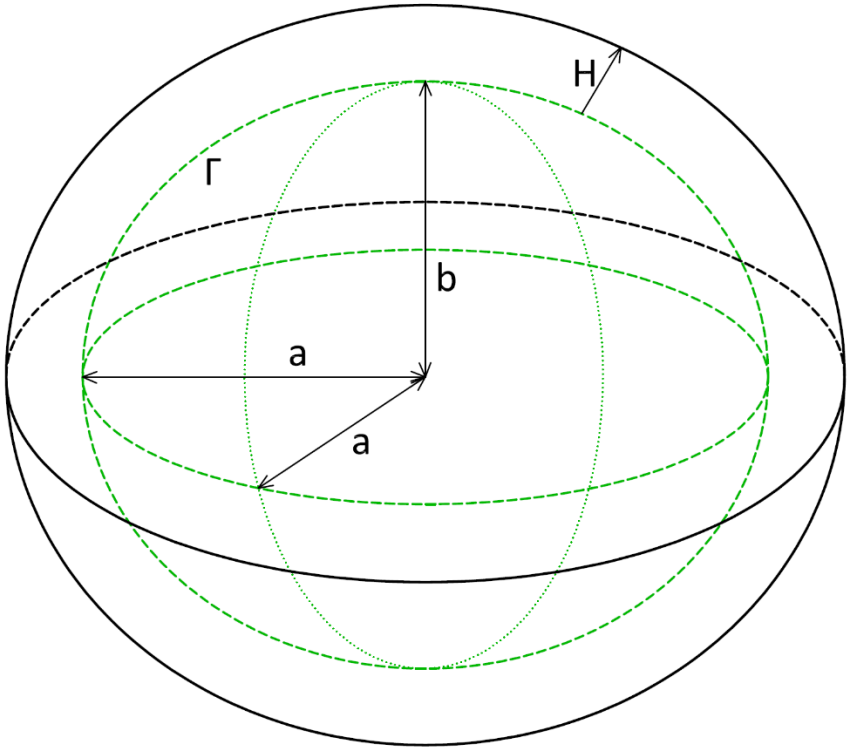
- Motivation
- Solution of the GBVP
  - Optimal linear solver
- Speedup of solution
  - Parallelization
  - Domain decomposition
  - Parallel domain decomposition
- Conclusion

Experiment settings

Resolution	Element size on equator	Elements on the Earth surface
40’ x 40’	~74.1km	540 x 270

540 x 270 x 40 = 4 374 000

- Size of the memory for storage of the matrix and the right hand-side vector for the experiment with unknowns was 52.84 MB.



Solution Ax=b

- **Direct solution:**
  - Cholesky factorization, LU decomposition, QR decomposition
- **Iterative solution:**
  - **Stationary methods**: Jacobi, Gauss–Seidel (GS) and Successive over-relaxation (SOR)
    - slow convergence, no additional vectors
  - **Nonstationary methods**: Conjugate gradient (CG), Generalized Minimal Residual (GMRES), Biconjugate gradient method (BiCG)
    - fast convergence, additional vector are needed

Solution Ax=b

- The memory and operations costs for various linear solvers
- APXY is a number of vector scalar products, DOT is a number of scalar-vector Multiplications, MEM represents a number of additional vectors needed in iterative procedure.

Solver	MV	AXPY	DOT	MEM
Bi-CG	2	7	2	7
Bi-CGSTAB	2	3	2	7
Bi-CGSTAB2	2	6	2	10
Bi-CGSTAB(l)	2	0,75*(l+3)	0,25*(l+7)	2l+5
GMRES(l)	1	0,5*(l+3)	0,5*(l+1)	l+3

Solution Ax=b

- Efficiency comparison for various Bi-CGSTAB linear solvers in the experiment with 4 374 000 unknowns, tested on 1 CPU.

Solver	Number of iterations	CPU time [s]	Additional solver memory [MB]
BiCGstab	1053	403.82	184.37
BiCGstab(2)	585	494.14	258.02
BiCGstab(4)	275	629.01	405.46
BiCGstab(8)	130	860.86	700.34

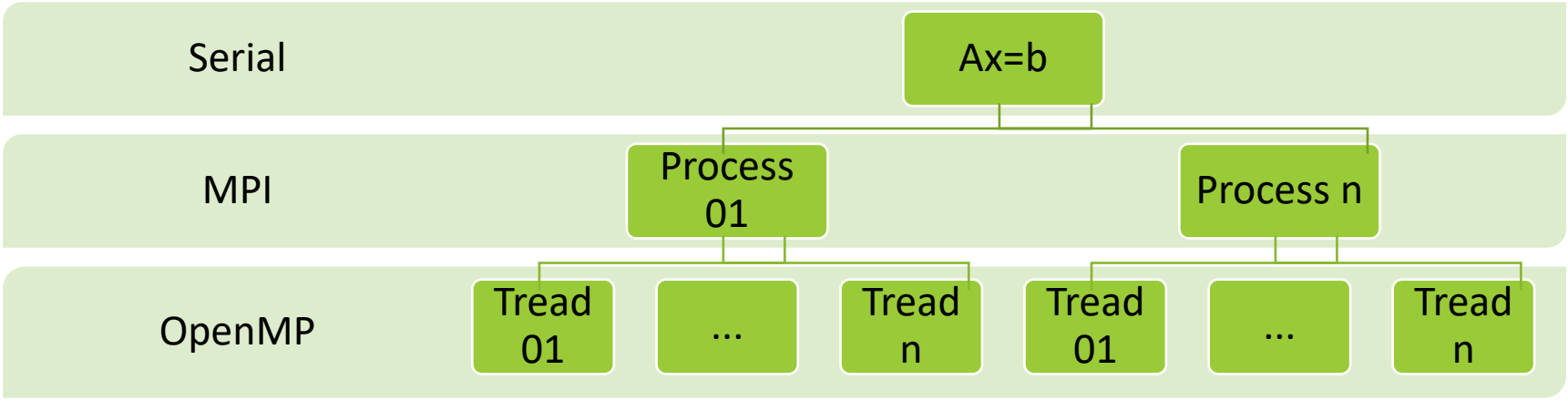
Size of the memory for storage of the matrix and the right hand-side vector for the experiment with unknowns was 52.84 MB.

Outline

- Motivation
- Solution of the GBVP
  - Optimal linear solver
- Speedup of solution
  - Parallelization
  - Domain decomposition
  - Parallel domain decomposition
- Conclusion

Parallelization of the solution

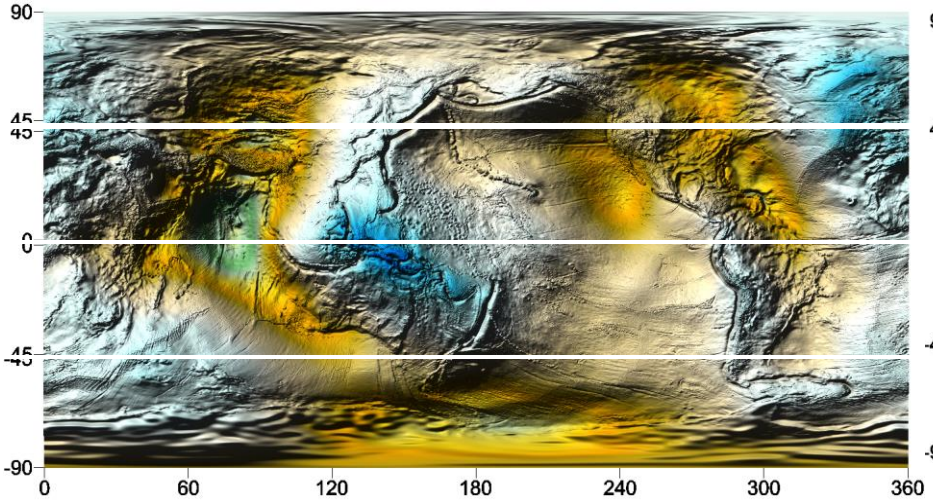
- Nowadays, the speed up of numerical algorithms is performed by distribution of computations into several processes using so-called Massively Parallel Processors architecture together with the Message Passing Interface (MPI) and Open Multi-Processing (OpenMP) programming framework.

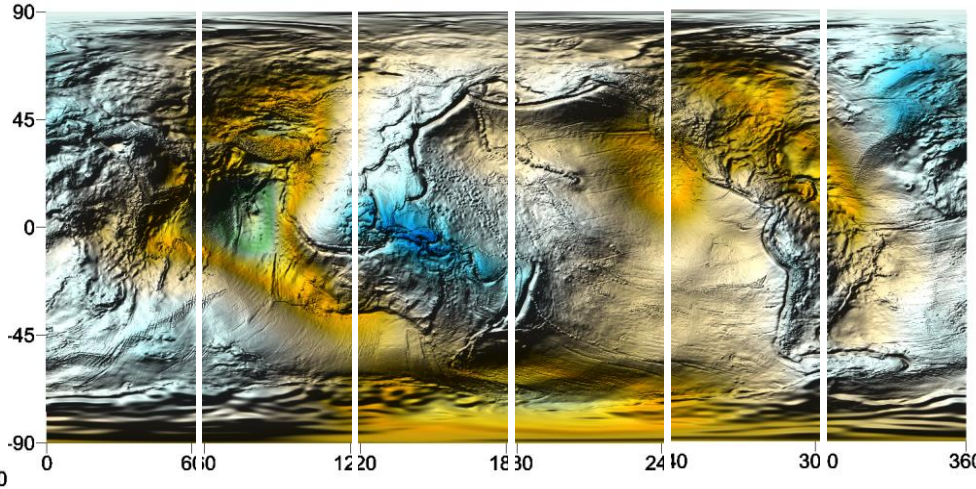


Parallelization of the solution

Resolution	Longitude [MB]	Latitude [MB]	Radial [MB]
40’ x 40’	0.345	0.172	2.332
1’ x 1’	86.400	43.200	3 732.480

- Illustration of data management in MPI parallel implementations.





Parallelization of the solution

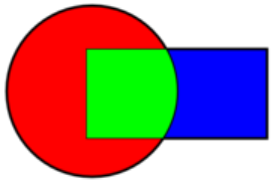
Balancing between numbers of Processors and Treads configuration is based on real configuration of computing resources. In our case we have four quad-core CPU in computing node.

- Comparison of Processor/Tread parallelization in the experiment with 4 374 000 unknowns, computed on four quad-core CPU.

MPI Processors	OpenMP Threads	CPU time [s]	Speedup ratio	RAM [MB]	Memory increase
1	1	403.82	-	237.108	-
	2	232.40	1.73		
	4	191.36	2.11		
	8	87.31	4.63		
	16	57.51	7.02		
2	1	216.84	1.86	245.868	+ 3.7%
	2	126.17	3.20		
	4	98.46	4.10		
	8	85.88	4.70		
4	1	114.01	3.54	266.040	+ 12.2%
	2	79.72	5.06		
	4	55.56	7.26		
8	1	79.34	5.09	308.456	+ 30.0%
	2	70.81	5.7		
16	1	59.51	6.78	390.068	+ 64.5%

Outline

- Motivation
- Solution of the GBVP
  - Optimal linear solver
- Speedup of solution
  - Parallelization
  - Domain decomposition
  - Parallel domain decomposition
- Conclusion

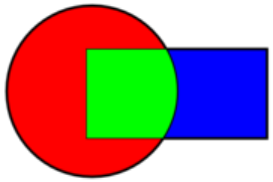


Domain decomposition

- **The Multiplicative Schwarz Method** :
  - Basic DD idea is to decompose the computational domain Ω into M subdomains $\Omega = \bigcup_{i=1}^{M} \Omega_i$
  - Find solution on each Ω_i with proper boundary conditions

$$\Delta T_i^n = 0 \; in \; \Omega_i,$$
$$\nabla T_i^n . \vec{s} = \delta g \; on \; \Gamma_D,$$
$$T_i^n = T_{SAT} \; on \; \Gamma_U,$$
$$T_i^n = T_* \; on \; \hat{\Gamma}_i,$$

where $\hat{\Gamma}$ is a boundary between each pair of neighboring subdomains. The artficial Dirichlet condition T_i is updated by the exchanging some data on $\hat{\Gamma}_i$ from the neighboring subdomains.



Domain decomposition

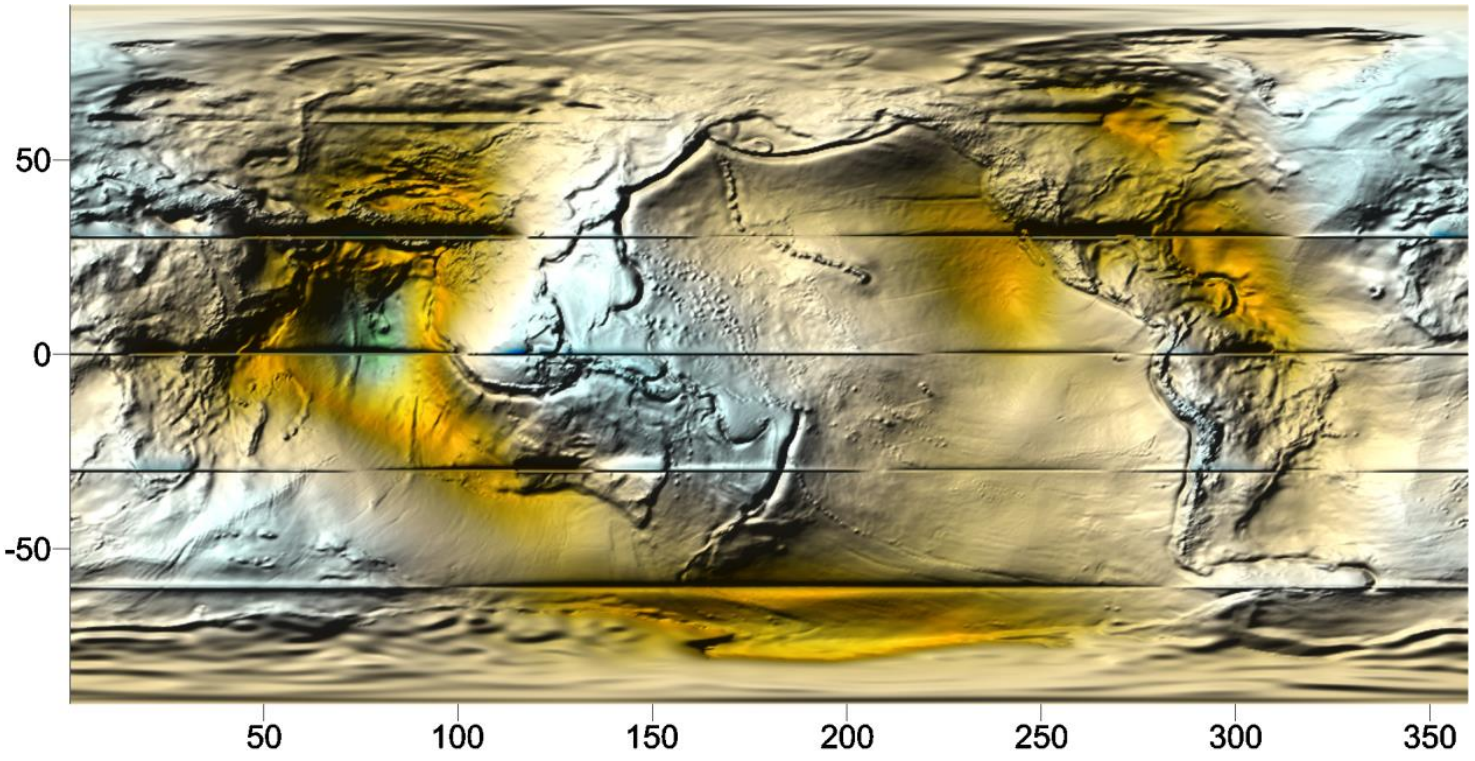
- **The Additive Schwarz Method** :
  - The difference between the multiplicative Schwarz method and the additive Schwarz method is in the way how the artificial Dirichlet condition is updated.
  - Dirichlet condition is updated by using

$$T_i^n = T_*^{n-1} \quad on\ \hat{\Gamma}_i,$$

  - This means that the artificial Dirichlet condition is updated using solutions from all relevant neighboring subdomains from previous iteration.
  - Therefore, the subdomain solution in the additive Schwarz method can be carried out completely independently.

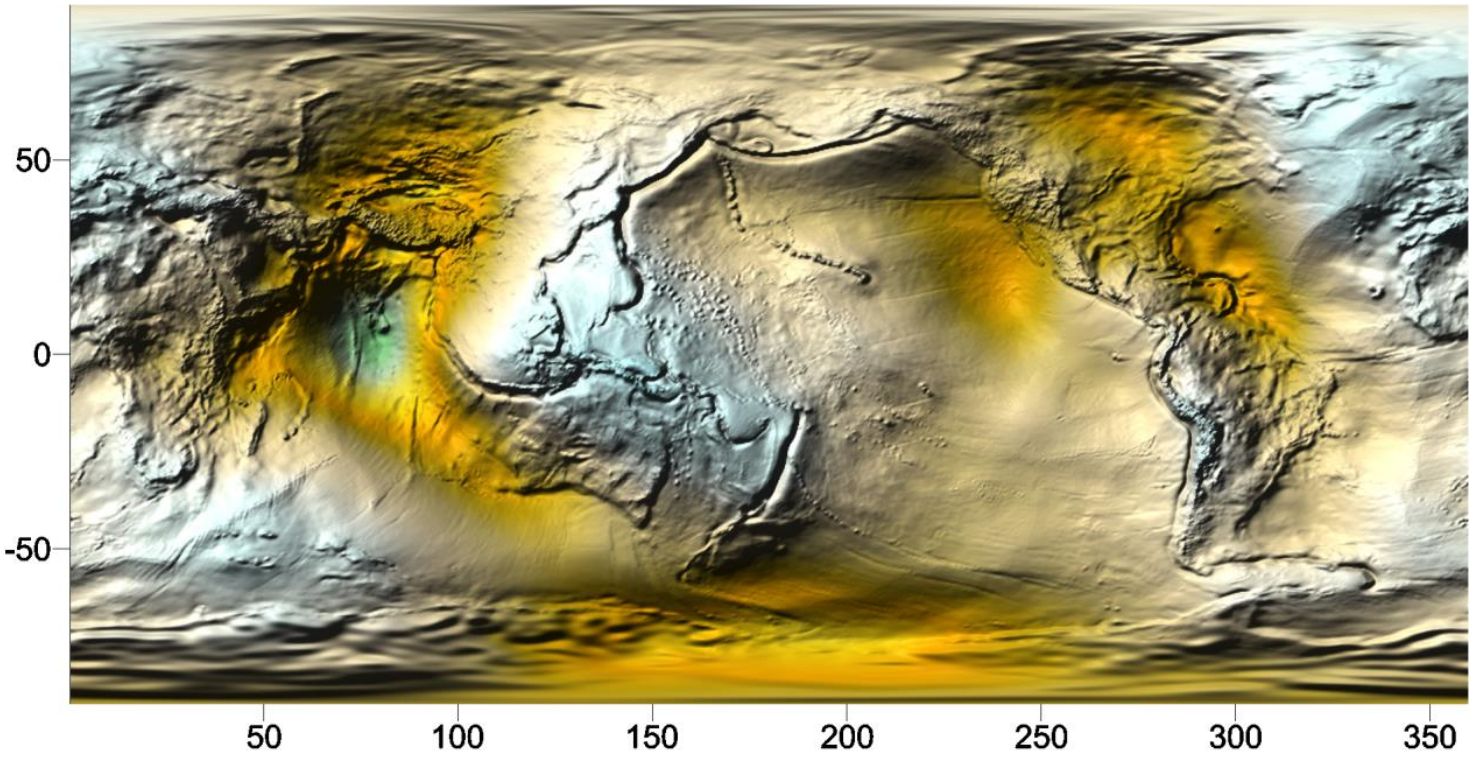
Domain decomposition

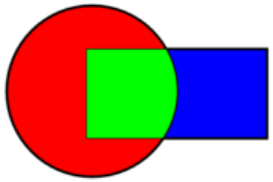
- Illustration of the solution after the first iteration of DD



Domain decomposition

- Illustration of the solution after 10 iterations of DD

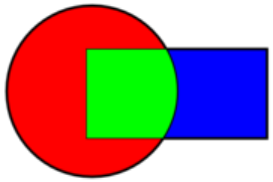




Domain decomposition

- Efficiency comparison for various number of subdomains tested in experiment with 4 374 000 unknowns, computed on 1 CPU.

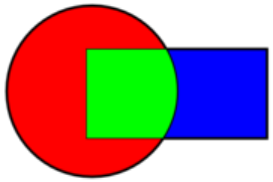
Number of subdomains	CPU time [s]	Speedup ratio	RAM [MB]	Memory saving
1	403.82	-	237.108	-
5	1651.68	0.24	89.868	-62.1%
10	907.99	0.44	71.308	-69.9%
15	856.04	0.46	65.248	72.5%
30	854.24	0.47	57.816	-75.6%



Domain decomposition

- **The additive Schwarz Method** :
  - the artificial Dirichlet condition is updated using solutions from all the relevant neighboring subdomains every μ iteration, where $\mu \in R$.

$$\Delta T_i^n = 0 \; in \; \Omega_i,$$
$$\nabla T_i^n . \vec{s} = \delta g \; on \; \Gamma_D,$$
$$T_i^n = T_{SAT} \; on \; \Gamma_i,$$
$$T_i^n = T_*^{n-\mu} \; on \; \hat{\Gamma}_i.$$



Domain decomposition

- Efficiency comparison for the different number μ in the experiment with 4 374 000 unknowns for case of 30 subdomains, tested on 1 CPU

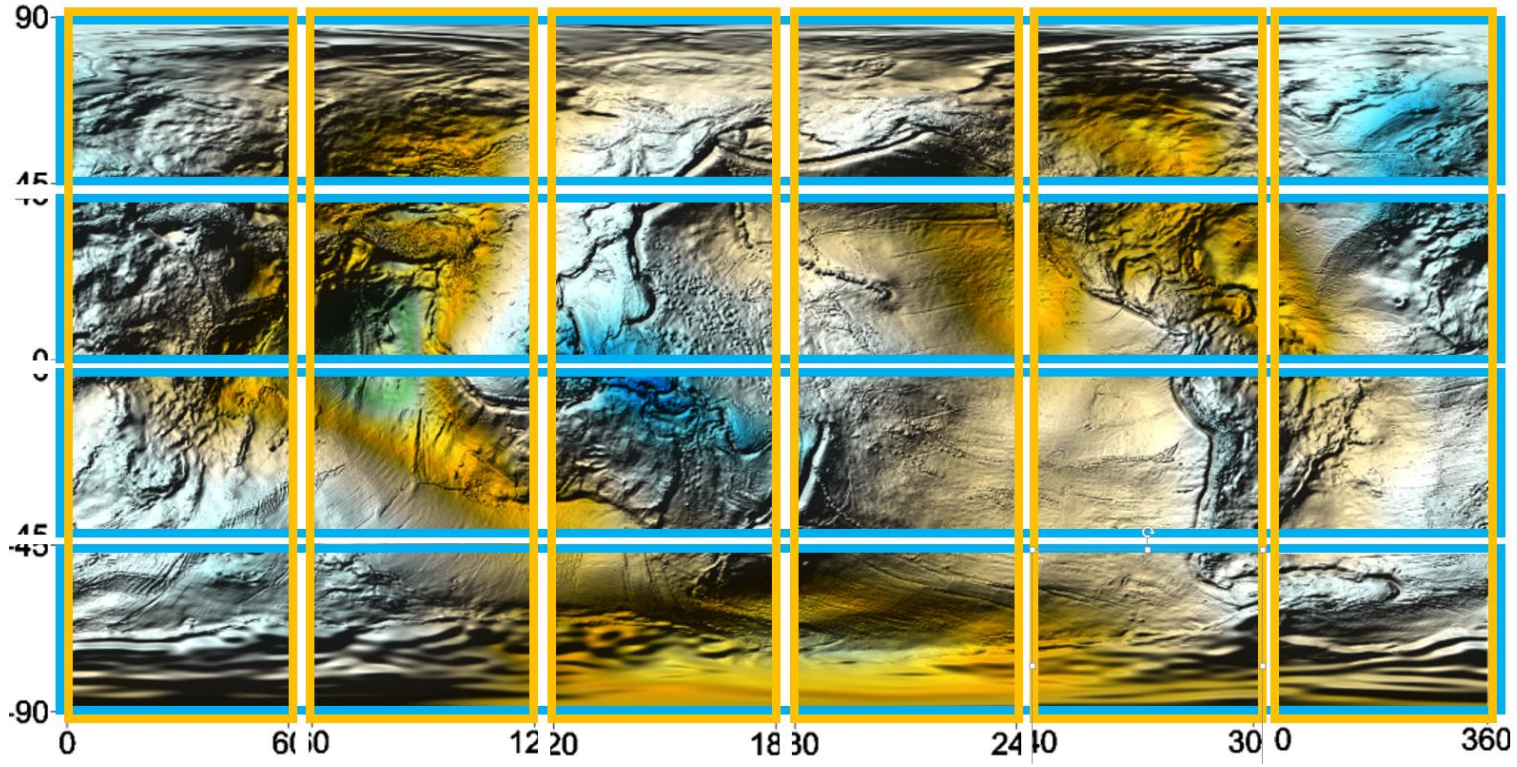
μ	CPU time [s]	Speedup ratio
1	854.24	-
5	308.02	2.77
10	252.33	3.38
15	224.65	3.80
20	236.56	3.61
25	265.73	3.21

Outline

- Motivation
- Solution of the GBVP
  - Optimal linear solver
- Speedup of solution
  - Parallelization
  - Domain decomposition
  - Parallel domain decomposition
- Conclusion

Parallel - Domain decomposition

- Illustration of data management in Parallel DD implementations.



Parallel - Domain decomposition

- Comparison for the different number of subdomains using parallel DD method in the experiment with 4 374 000 unknowns with μ = 15, tested on 4 quad-core CPUs

Number of subdomains	CPU time [s]	Speedup ratio	RAM [MB]	Memory saving
1	55.56	-	266.040	-
5	55.52	1.00	115.508	-56.6%
10	28.47	1.95	97.568	-63.3%
15	17.44	3.18	91.156	-65.7%
30	18.67	2.97	84.128	-68.3%

Efficiency comparison for different computation strategies

- Efficiency comparison for methods, in the experiment with 4 374 000 unknowns, computed on 4 quad-core CPU.

Computational strategies	CPU time [s]	Speedup ratio	RAM [MB]	Memory saving
Serial	403.82	-	237.108	-
Parallel	55.56	7.26	266.040	+10.8%
Domain dec.	224.65	1.79	57.816	-75.6%
Paralel DD	18.67	21.6	84.128	-64.5%

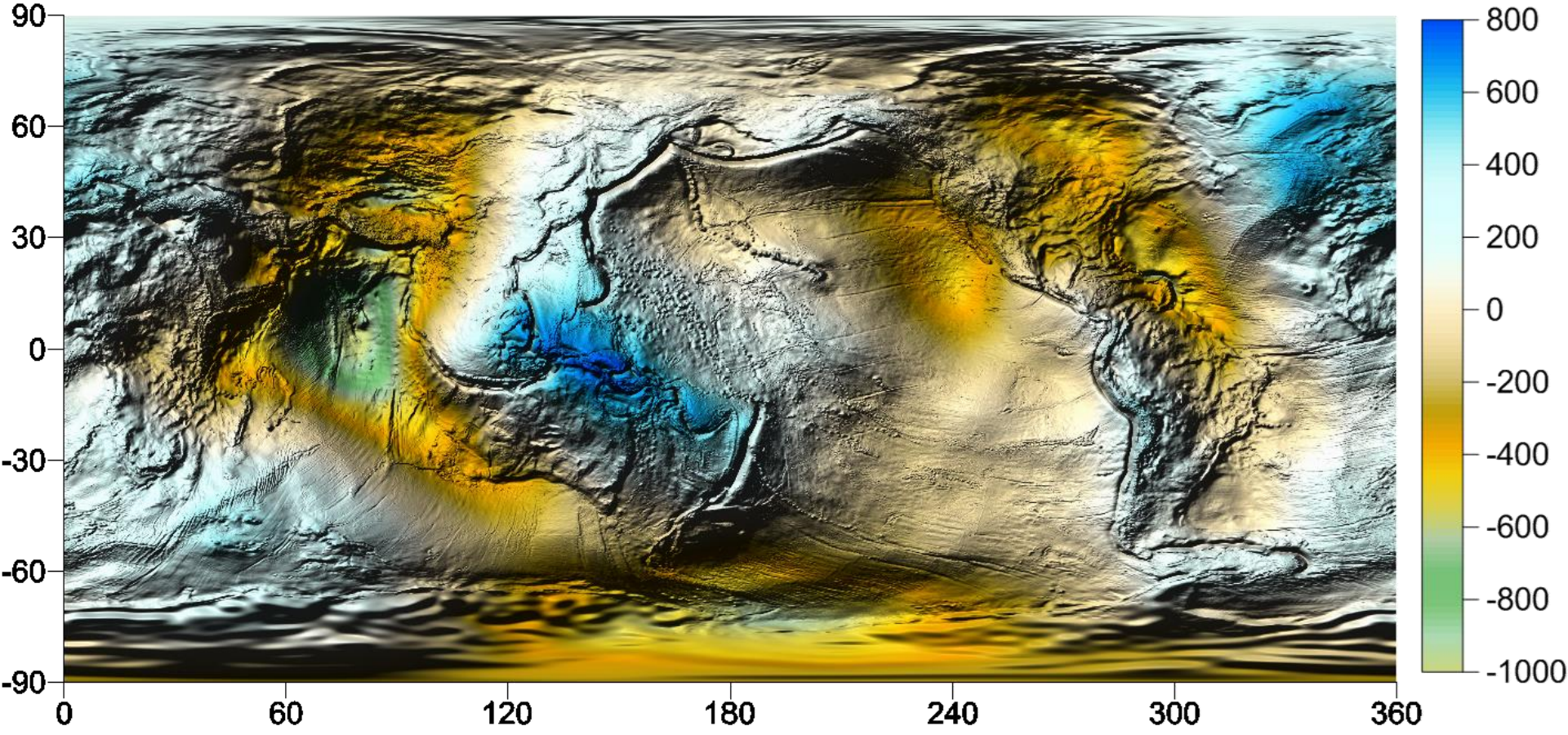
Parallel - Domain decomposition 1’ x 1’

- Comparison for the different number of subdomains using Parallel-Domain decomposition method in the experiment with 34 992 000 000 unknowns, tested on 28 octo-core CPUs.

Number of subdomains	CPU time [s]	Speedup ratio	RAM [GB]	Memory saving
1	706.8	-	1 652	-
5	703.5	1.00	557	66.3%
10	700.9	1.01	420	74.5%
15	710.0	0.99	375	-77.3%
30	718.5	0.98	329	-80.0%

Parallel - Domain decomposition 1’ x 1’

- Global gravity field model with the resolution 1’ x 1’ on the Earth's surface, [m^2s^2].



Outline

- Motivation
- Solution of the GBVP
  - Optimal linear solver
- Speedup of solution
  - Parallelization
  - Domain decomposition
  - Parallel domain decomposition
- Conclusion

Conclusion

- Since solution to the GBVP in a space domain leads to a huge linear system, we have shown that Bi-CGSTAB is an optimal linear solver due to the minimal time consumption.
- In case of speed up of solution, we have presented MPI and OpenMP as one of its possibilities.
- Parallelization together with the DD method have shown an immense contribution to improving algorithms in reduced computation time and memory costs.
- After all optimizations we are able to solve experiment with corresponding 34 992 000 000 unknowns in acceptable time using minimal memory requirements
- All numerical computations have space to improve convergence and data management.

Thank for your attention