Challenges in fluid flow simulations using Exa-scale computing

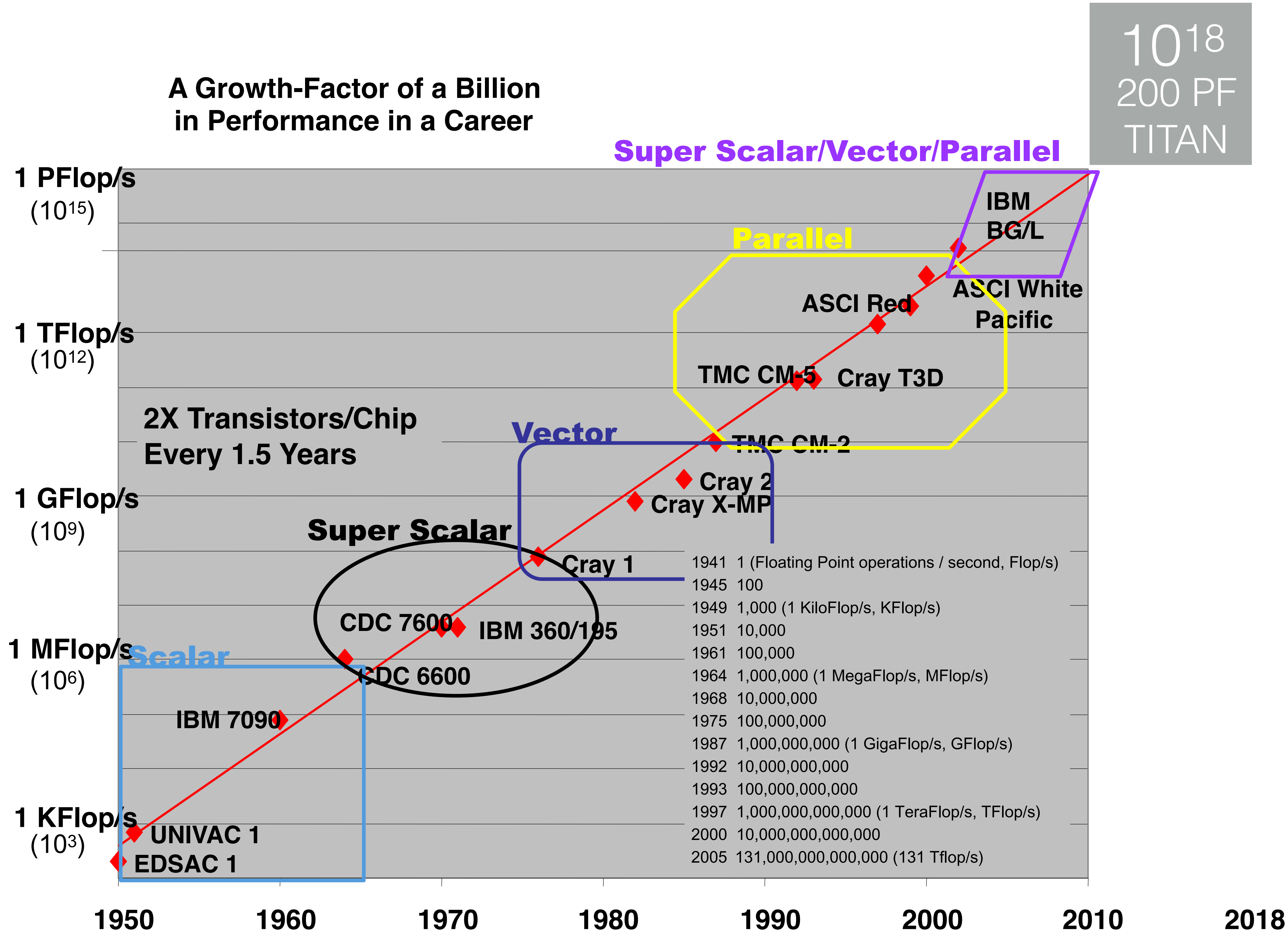
Mahendra Verma
IIT Kanpur

http://turbulencehub.org

mkv@iitk.ac.in

Hardware

From Karniakadis's course slides



AMD EPYC™ 7551

– Specifications

# of CPU Cores: 32	**# of Threads:** 64	**Base Clock:** 2GHz
Max Boost Clock: 3GHz	**All Core Boost Speed:** 2.55GHz	**Total L3 Cache:** 64MB
Socket Count: 1P/2P	**PCI Express Version:** x128	**Default TDP / TDP:** 180W

– System Memory

System Memory Specification: 2666MHz	**Memory Channels:** 8	**Mem BW (2S Theo):** 341 GB/s

https://www.amd.com/en/products/cpu/amd-epyc-7551

NODE: 2 proc/node; Focus on a node

Flop rating for 2 procs: 2*32*24 = 1536 GF

Wants data ~ 8 TB/sec.

Cache, RAM, HD

Data transfer

FLOPS free, data transfer expensive (Saday)

Memory BW = 341 GB/s

SSD: transfer rate = 6 Gbit/s

peak IB Switch speed/port = 200 Gb/s

software challenges

For beginners

- Abundance (MPI, OpenMP, CUDA, ML)
- Leads to confusion and non-start..
- Structured programming
- Pressure to do the science..
- Some times CS tools are too complex to be practical.

For advanced users

- Optimised use of hardware.
- Structured and modular, usable code with documentation.
- Keeping up with upgrades and abundance (MPI3, ML, C++11, Vector processors, GPU, XeonPhi, Rasberry Pi).
- Optimization
- Interactions with users + programers

Now CFD
(Computational fluid dynamics)

Applications

- Weather prediction and climate modelling
- Aeroplane and cars (transport)
- defence / offences
- Turbines, dams, water management
- Astrophysical flows
- Theoretical understanding

Field reversal

with Mani Chandra

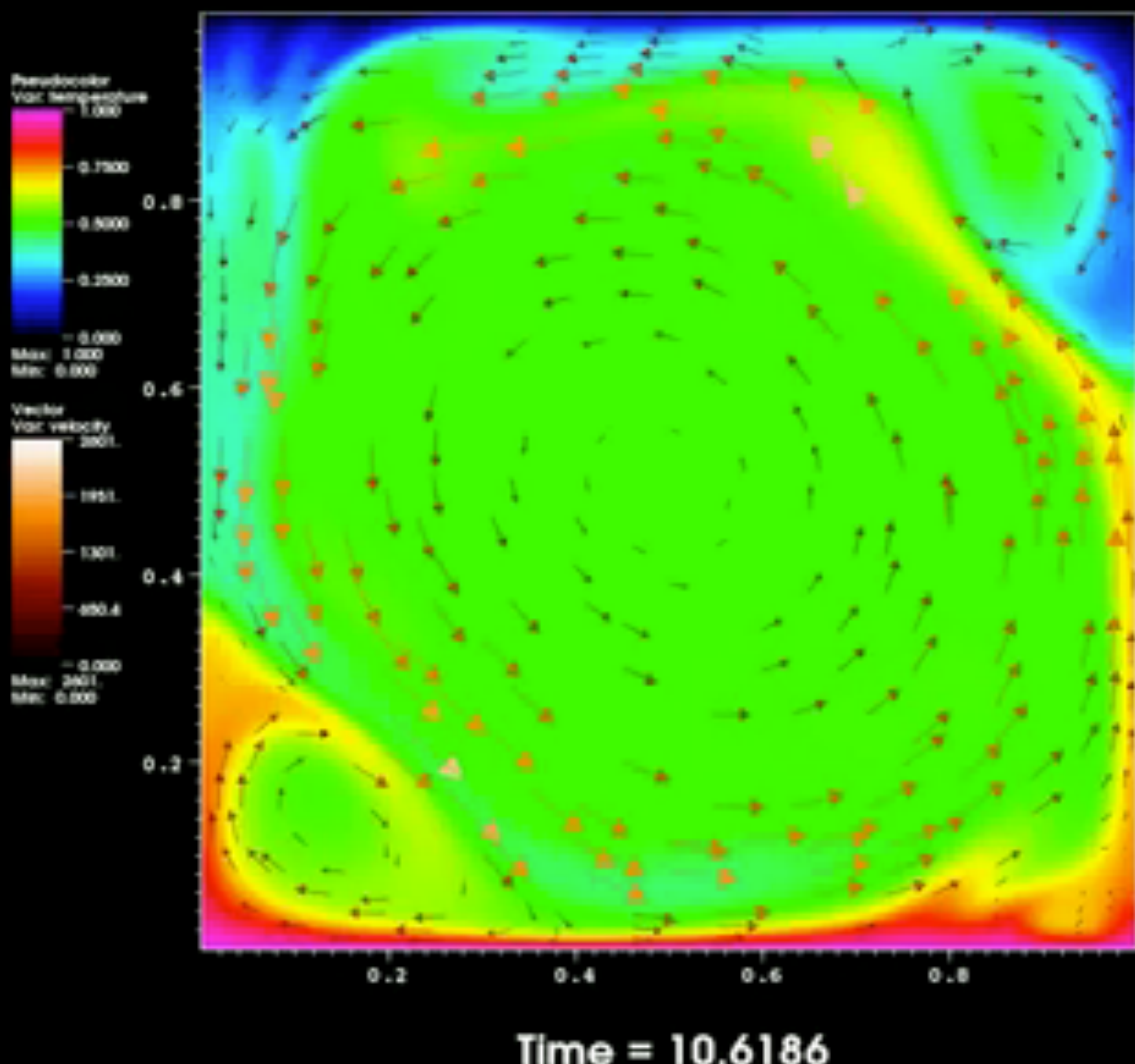
Geomagnetism

Glatzmaier & Roberts
Nature, 1995

Polarity reversals after random time intervals (tens of millions of years to 50K years).

Last reversal took place around 780,000 years ago.

Nek5000 (Spectral-element) simulation



(1,1)➞(2,2) ➞(1,1)

spectral-element code
Nek5000

Chandra & Verma, PRE 2011, PRL 2013

Methods

- Finite difference
- Finite volume
- Finite element
- Spectral
- Spectral element

Spectral method

Example:
Fluid solver

velocity field

Pressure

Ext. Force

$$\partial_t \mathbf{u} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\nabla p + \nu \nabla^2 \mathbf{u} + \mathbf{F}$$

kinematic viscosity

$$\nabla \cdot \mathbf{u} = 0$$

Incompressibility

Reynolds no = $\frac{UL}{\nu}$

Procedure

$$f(x) = \sum_{k_z} \hat{f}(k_x)\exp[i(k_x x)]$$

$$df(x)/dx = \sum_{k_z} [ik_x \hat{f}(k_x)]\exp[i(k_x x)]$$

Set of ODEs

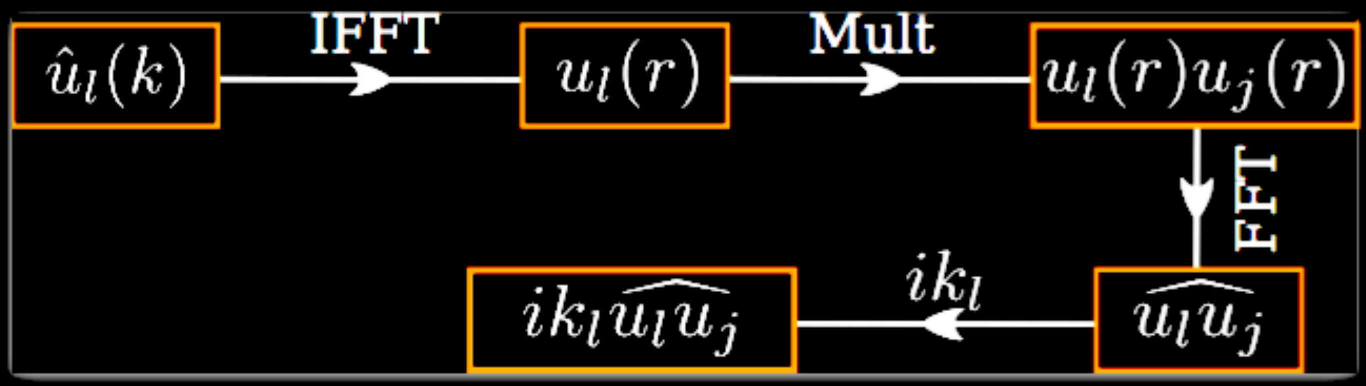
$$\frac{du_i(\mathbf{k})}{dt} = -jk_m \widehat{u_m(\mathbf{r})u_i(\mathbf{r})} - jk_i p(\mathbf{k}) - \nu k^2 u_i(\mathbf{k})$$

Time advance (e.g., Euler's scheme)

$$u_i(\mathbf{k}, t+dt) = u_i(\mathbf{k}) + dt \times \mathrm{RHS}_i(\mathbf{k}, t)$$

Stiff equation for small viscosity ν (use exponential trick)

Nonlinear terms computation:



(pseudo-spectral)

Fourier transforms take around 80% of total time.

Tarang = wave (Sanskrit)

Spectral code (Orszag)

One code to do many
turbulence & instabilities problems

VERY HIGH RESOLUTION (6144^3)

Cores: 196692 of Shaheen II of KAUST

Opensource, download from
http://turbulencehub.org

Chatterjee et al., JPDC 2018

Fluid
MHD, Dynamo
Scalar
Rayleigh-Bénard convection
Stratified flows
Rayleigh-Taylor flow
Liquid metal flows
Rotating flow
Rotating convection

Instabilities
Chaos
Turbulence

No-slip BC
Cylinder
sphere
Toroid
(in progress)

Periodic BC
Free-slip BC

Rich libraries to compute
Spectrum
Fluxes
Shell-to-shell transfer
Structure functions

New things
Fourier modes
Real space probes
Ring-spectrum
Ring-to-ring transfer

Tested up to 6144^3 grids

Object-oriented design

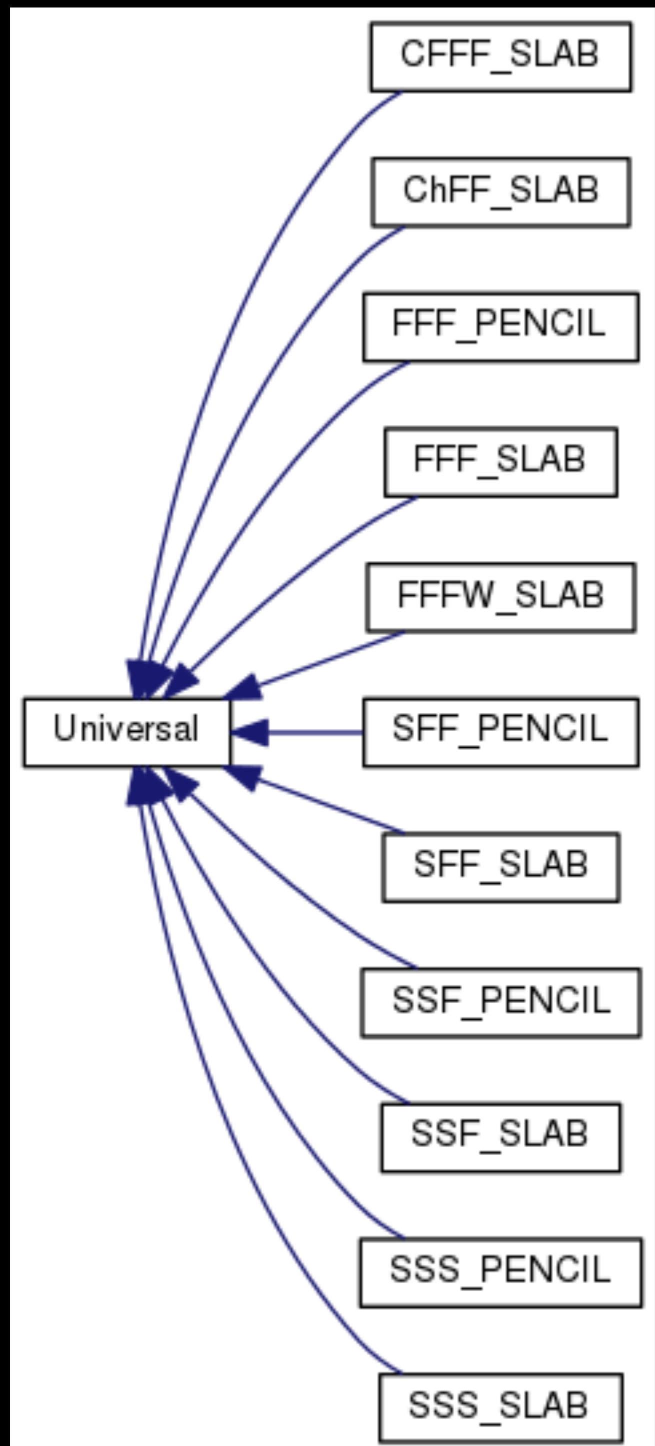
Basis functions (FFF, SFF, SSF, SSS, ChFF)

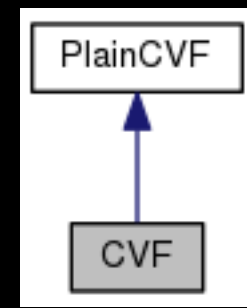
Basis-independent universal function (function overloading)

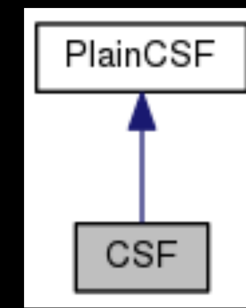
e.g., compute_nlin $(u.\ \nabla)u$,
$(b.\ \nabla)u$, $(b.\ \nabla)b$, $(u.\ \nabla)T$.

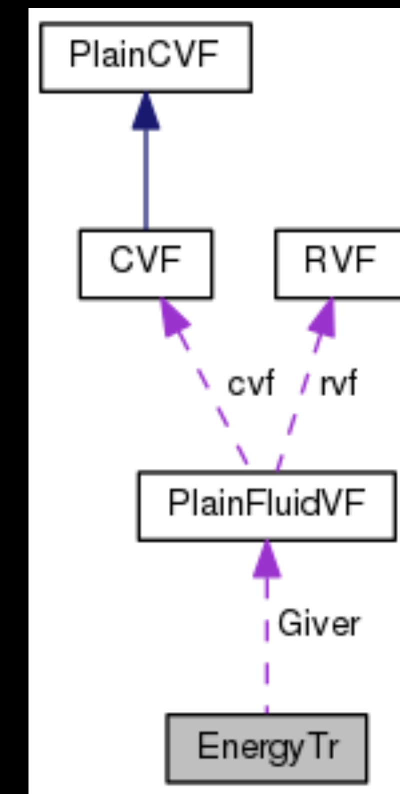
General PDE solver

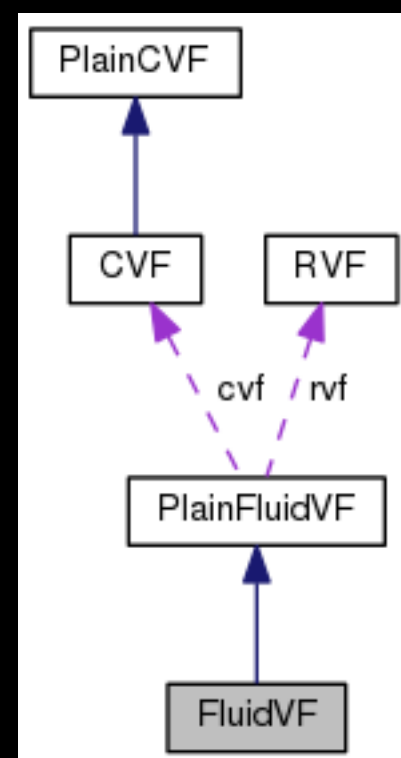
We can use these general functions to simulate
MHD, convection etc.

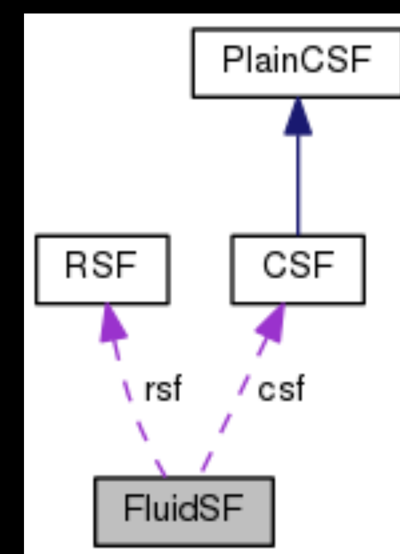


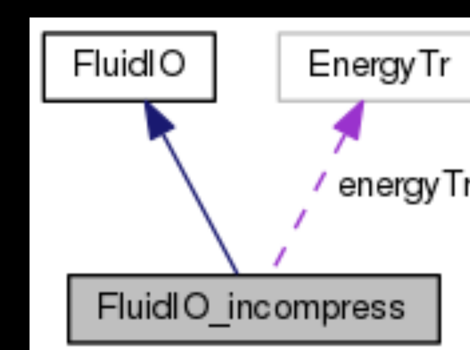












Generated by Doxygen

Parallelization

Spectral Transform
(FFT, SFT, Chebyshev)

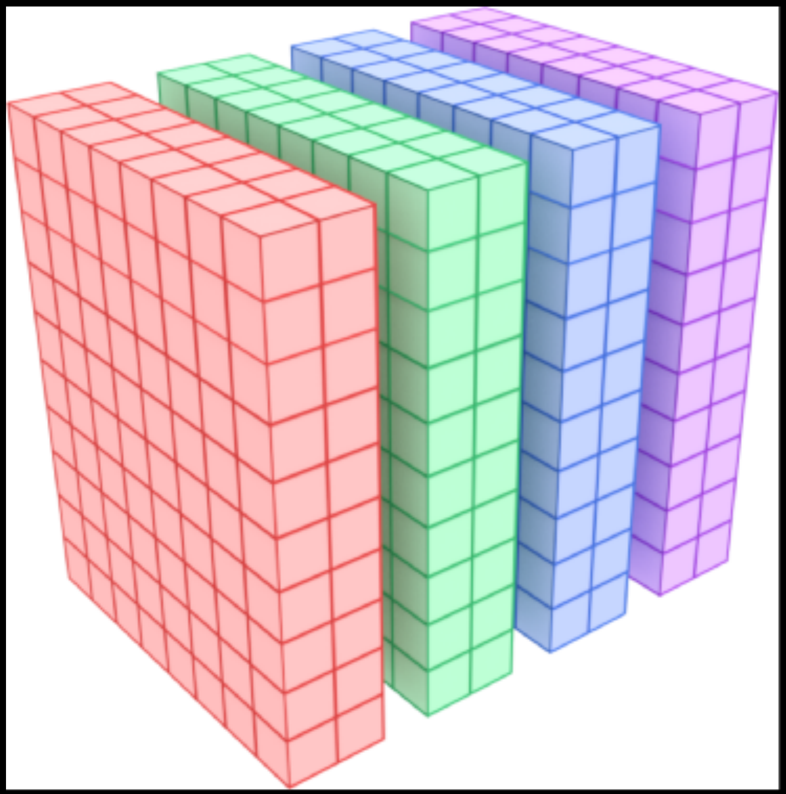
Multiplication in real space

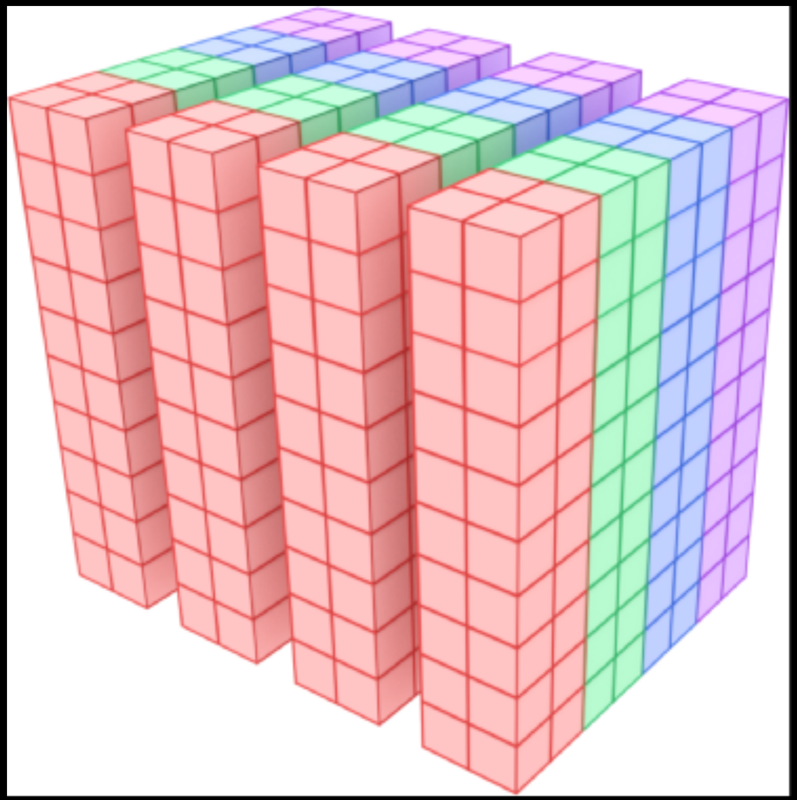
Input/Output
HDF5 lib

FFT Parallelization

$$f(x,y,z) = \sum_{k_x}\sum_{k_y}\sum_{k_z}\hat{f}(k_x,k_y,k_z)\exp[i(k_x x + k_y y + k_z z)]$$

Slab decomposition

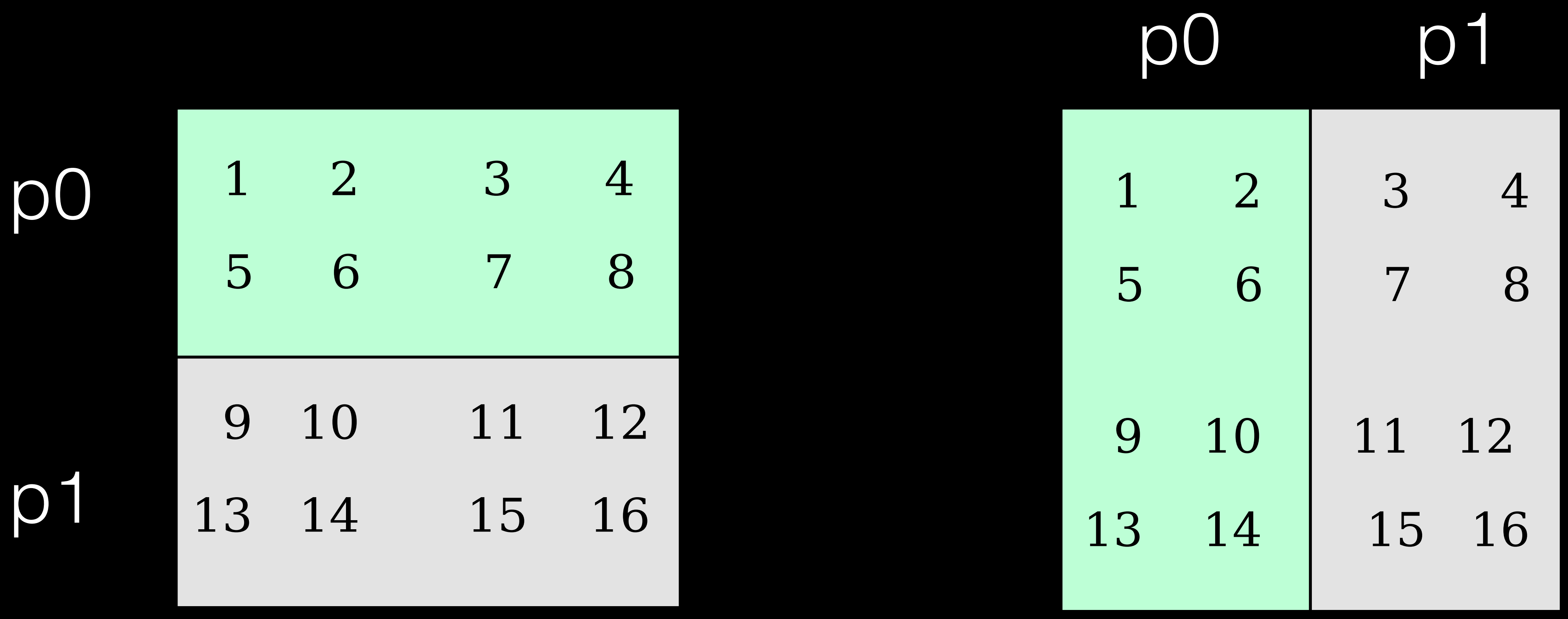




Data divided among 4 procs

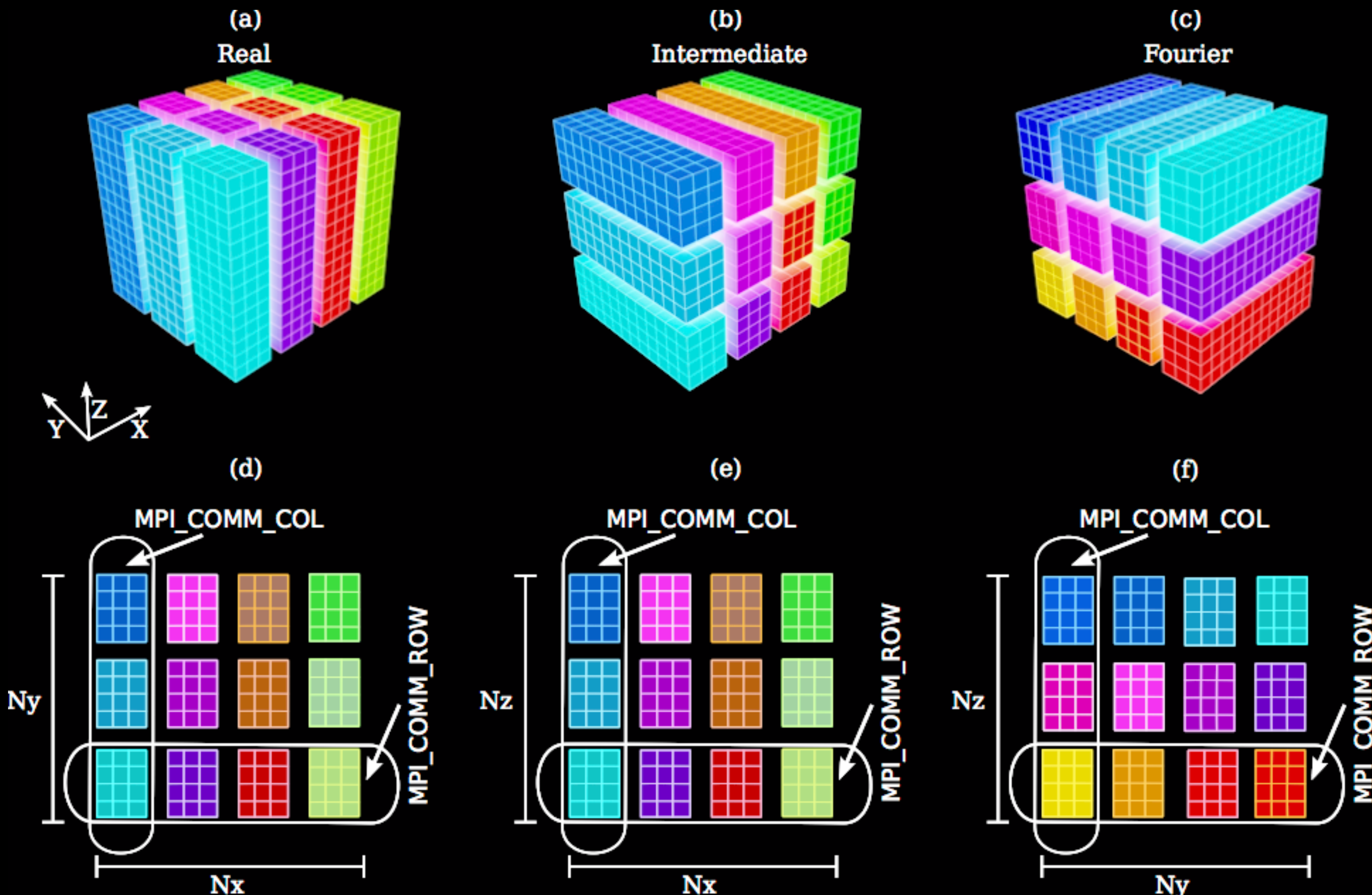
Transpose-free FFT

MPI vector, conconsecutive data transfer



12-15% faster compared to FFTW

Pencil decomposition

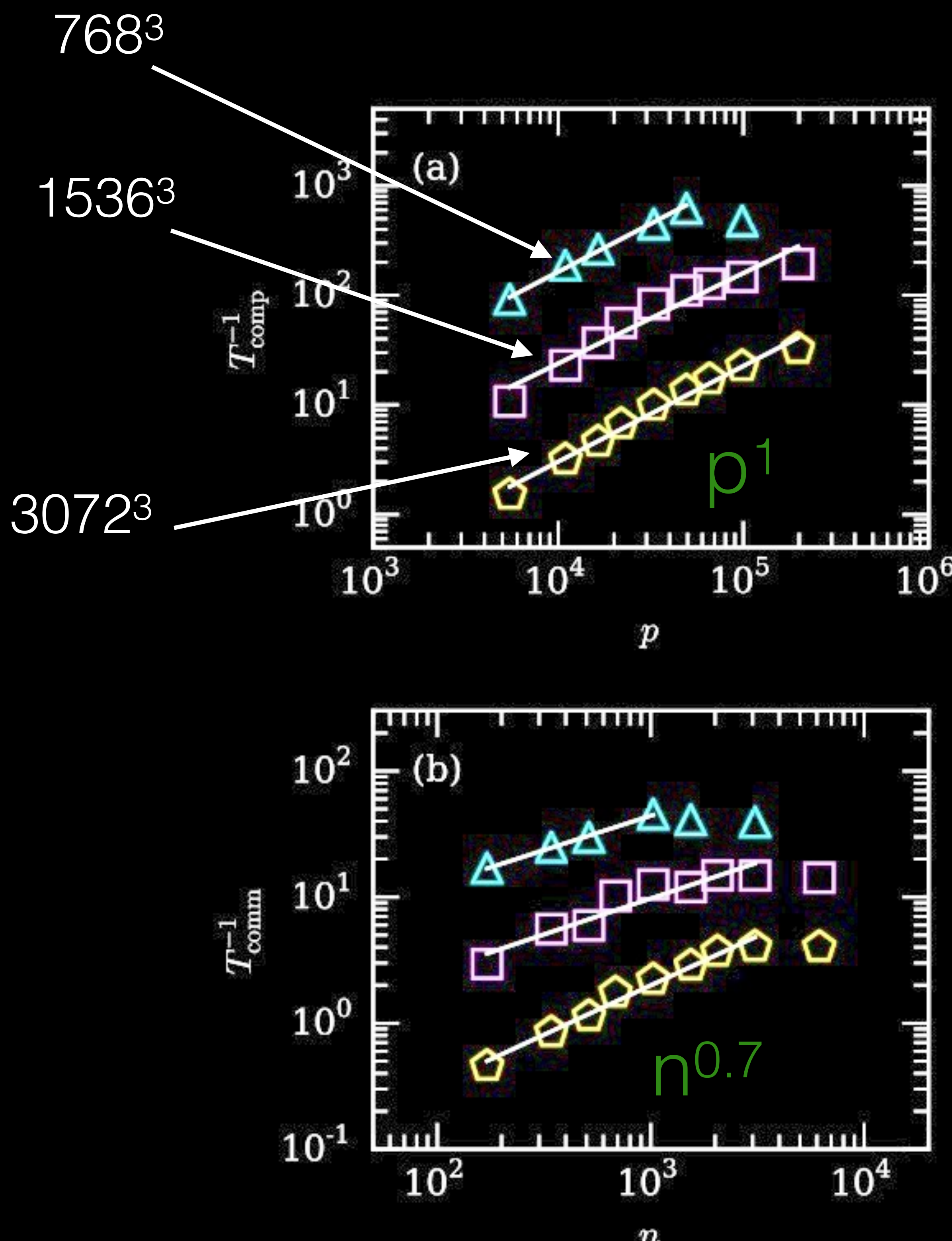


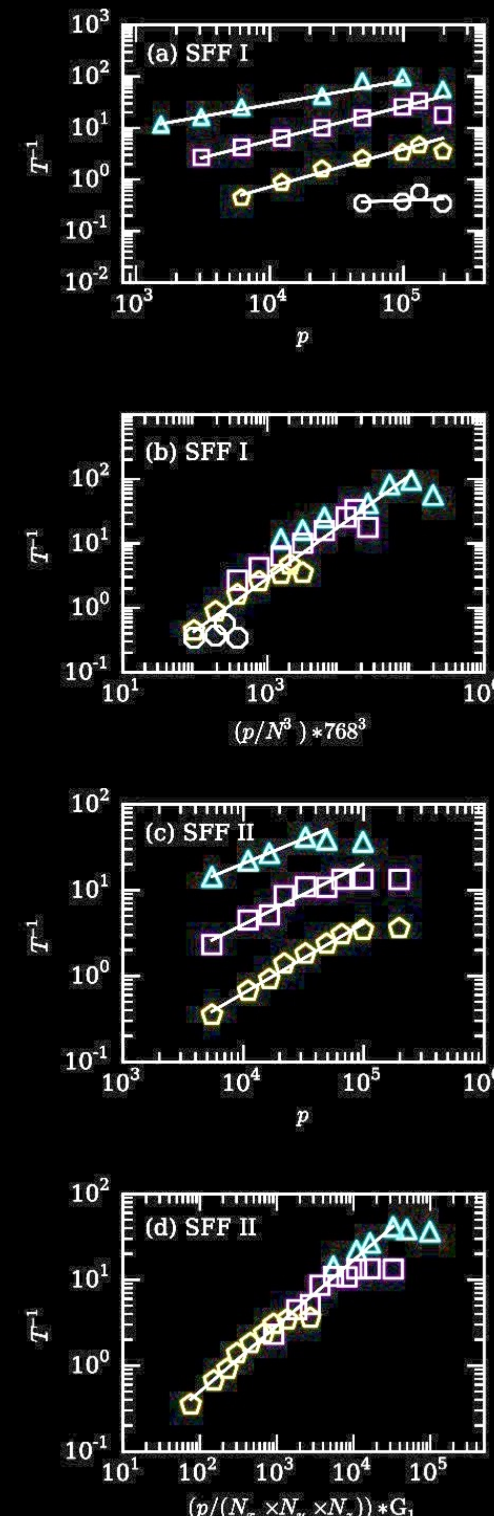
FFT scaling

On Shaheen 2 at KAUST
with Anando Chatterjee, Abhishek Kumar,
Ravi Samtaney, Bilel Hadri, Rooh Khurram

Cray XC40
ranked 9th in top500

Chatterjee et al., JPDC 2018

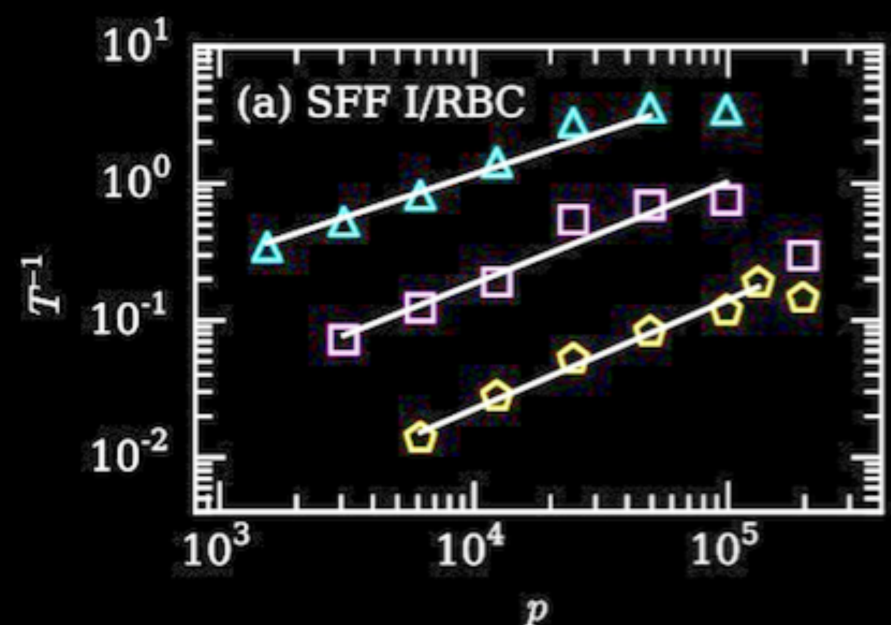

768³
1536³
3072³
(a)
p¹
T_comp^-1
p
(b)
n^0.7
T_comm^-1
n

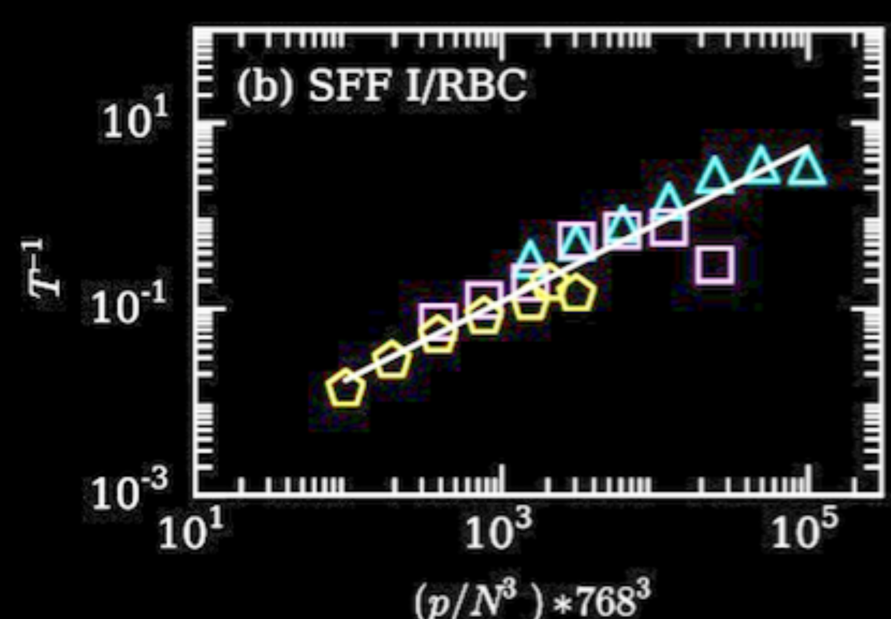

(a) SFF I
T^-1
p
(b) SFF I
(p/N³) *768³
(c) SFF II
p
(d) SFF II
(p/(N_x ×N_y ×N_z)) *G_1

Tarang scaling

On Shaheen at KAUST

- Weak scaling: When we increase the size of the problem, as well as number of procs, then should get the same scaling.





Average flop rating/core (~1.5 %)

Compare with BlueGene/P (~8 %)

Overlap Communication & Computation ??

GPUs ??

Xeon Phi ??

To Petascale & then Exascale

Finite difference code

General code: Easy porting to GPU, MiC

Collaborators:
Roshan Samuel
Fahad Anwer (AMU)
Ravi Samtaney (KAUST)

Summary

- ★ Code development
- ★ Module development
- ★ Optimization
- ★ Porting to large number of processors
- ★ GPU Porting
- ★ Testing

Acknowledgements

Students:

Anando Chatterjee

Abhishek Kumar

Roshan Samuel

Sandeep Reddy

Mani Chandra

Sumit Kumar & Vijay

Faculty:

Ravi Samtaney

Fahad Anwer

Ported to:

PARAM, CDAC

Shaheen, KAUST

HPC system IITK

Funding

Dept of Science and Tech., India

Dept of Atomic Energy, India

KAUST (computer time)

Thank you!