Direct Numerical Simulation of Turbulent Flows

Michael Wilczek

Institute for Theoretical Physics, University of Münster

22.07.09

Is turbulence a solved problem?

Is turbulence a solved problem?

Engineering: conceptually yes

- (depends on the complexity of your problem)
- relevant scenarios are computationally accessible

Is turbulence a solved problem?

Engineering: conceptually yes

- (depends on the complexity of your problem)
- relevant scenarios are computationally accessible

Physics: no

- statistical description from first principles is missing
- $\longrightarrow$ non-equilibrium thermodynamics?

Is turbulence a solved problem?

Engineering: conceptually yes

- (depends on the complexity of your problem)
- relevant scenarios are computationally accessible

Physics: no

- statistical description from first principles is missing
- $\longrightarrow$ non-equilibrium thermodynamics?

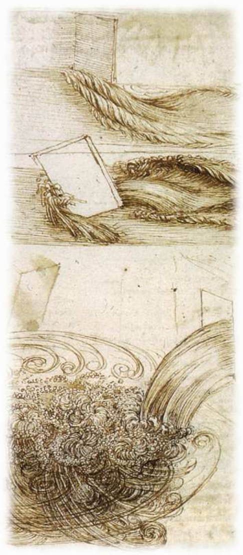
Mathematics: no

- boundedness of solutions with smooth inital conditions?

Introduction

Problem: turbulence ...

- is described by nonlinear equations
- exhibits spatio-temporal chaos
- involves large space- and time-scales



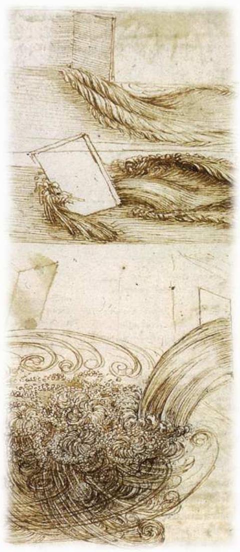
Introduction

Problem: turbulence ...

- is described by nonlinear equations
- exhibits spatio-temporal chaos
- involves large space- and time-scales

Possible solutions:

- understanding of structures
- formulating a statistical theory



Introduction

Problem: turbulence . . .

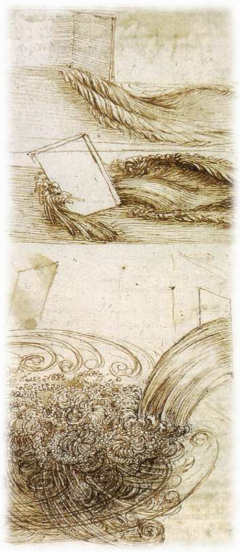
- is described by nonlinear equations
- exhibits spatio-temporal chaos
- involves large space- and time-scales

Possible solutions:

- understanding of structures
- formulating a statistical theory

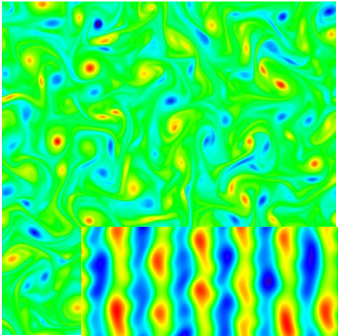
Tools:

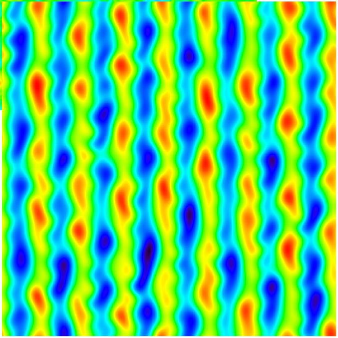
- any kind of mathematics, that will do
- computer simulations

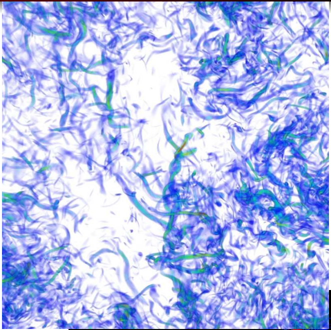


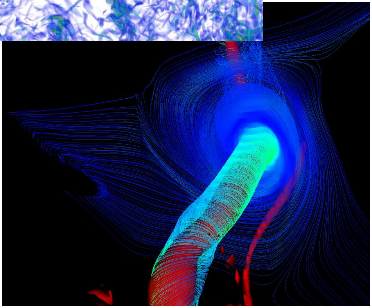
DNS

DNS: Visualization









DNS: equations

Navier-Stokes equations:

$$\frac{\partial \boldsymbol{u}}{\partial t}(\boldsymbol{x},t) + \boldsymbol{u}(\boldsymbol{x},t)\cdot\nabla\boldsymbol{u}(\boldsymbol{x},t) = -\nabla p(\boldsymbol{x},t) + \nu\Delta\boldsymbol{u}(\boldsymbol{x},t) + \hat{\boldsymbol{f}}(\boldsymbol{x},\boldsymbol{t})$$

$$\nabla\cdot\boldsymbol{u}(\boldsymbol{x},t) = 0$$

DNS: equations

Navier-Stokes equations:

$$\frac{\partial \boldsymbol{u}}{\partial t}(\boldsymbol{x},t) + \boldsymbol{u}(\boldsymbol{x},t)\cdot\nabla\boldsymbol{u}(\boldsymbol{x},t) = -\nabla p(\boldsymbol{x},t) + \nu\Delta\boldsymbol{u}(\boldsymbol{x},t) + \hat{\boldsymbol{f}}(\boldsymbol{x},\boldsymbol{t})$$

$$\nabla\cdot\boldsymbol{u}(\boldsymbol{x},t) = 0$$

Vorticity: $\boldsymbol{\omega}(\boldsymbol{x},t) = \nabla\times\boldsymbol{u}(\boldsymbol{x},t)$
Vorticity equation:

$$\frac{\partial \boldsymbol{\omega}}{\partial t}(\boldsymbol{x},t) = \nabla\times\big(\boldsymbol{u}(\boldsymbol{x},t)\times\boldsymbol{\omega}(\boldsymbol{x},t)\big) + \nu\Delta\boldsymbol{\omega}(\boldsymbol{x},t) + \boldsymbol{f}(\boldsymbol{x},t)$$

DNS: numerics I

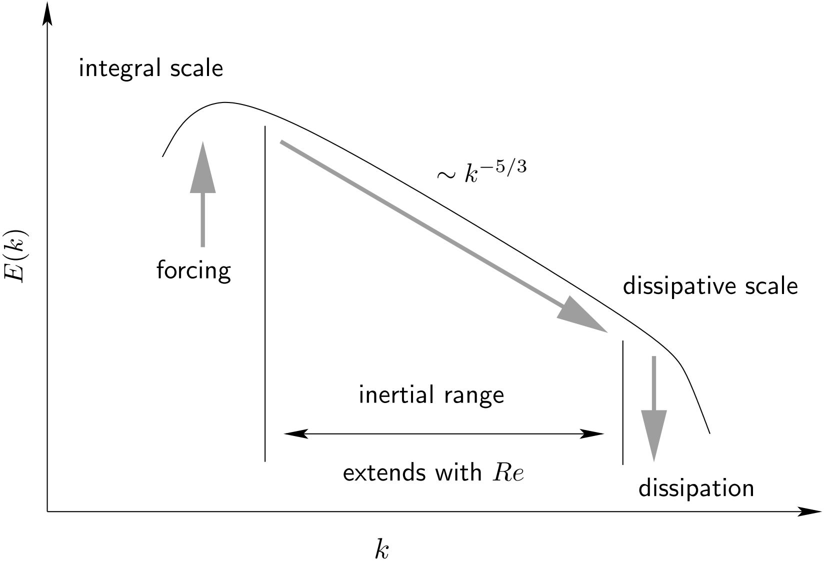
- aim: forced (stationary) homogeneous, isotropic turbulence
- temporal discretization: RK3 TVD
- spatial discretization: box-length 2π, $\dim$ grid points, periodic boundary conditions
- pseudospectral code

DNS: numerics II

$$\frac{\partial \tilde{\boldsymbol{\omega}}}{\partial t}(\boldsymbol{k},t) + \nu\, k^2\, \tilde{\boldsymbol{\omega}}(\boldsymbol{k},t) = i\boldsymbol{k} \times \mathcal{F}\{\boldsymbol{u}(\boldsymbol{x},t) \times \boldsymbol{\omega}(\boldsymbol{x},t)\} + \tilde{\boldsymbol{f}}(\boldsymbol{k},t)$$

- adaptive time-stepping (Courant-Friedrichs-Levy criterion)
- *pseudo*spectral: forward/backward FFT is computationally cheaper than convolution ($N \log N$ vs. N^2)
- aliasing: smooth Fourier filter
- viscosity is treated exactly (integrating factor)
- forcing: freezing of low modes
- code is MPI parallel

DNS: computational costs I



DNS: computational costs II

- forcing scale and dissipative scale should be well seperated
- inertial range extends with increasing Re
- size of smallest structures decreases with Re
- smallest structures should be well-resolved by the grid
- turbulent field should be accurately advanced in time

DNS: computational costs III

to be more precisely ...

$$\eta = \left(\frac{uL}{\nu}\right)^{-3/4} L = Re^{-3/4} L$$

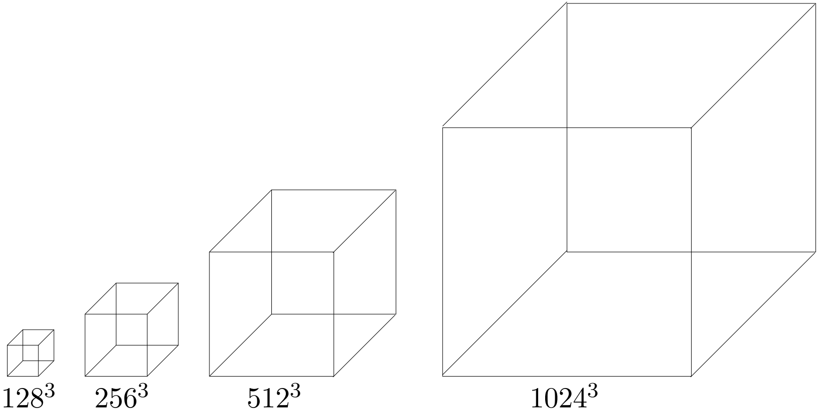
$$\Delta x \sim \eta$$

$$N_x \sim \left(\frac{2\pi}{\Delta x}\right)^3 \sim \left(\frac{2\pi}{L}\right)^3 Re^{9/4} \quad \longrightarrow Re \sim \left(\frac{L}{2\pi}\right)^{4/3} N_x^{4/9}$$

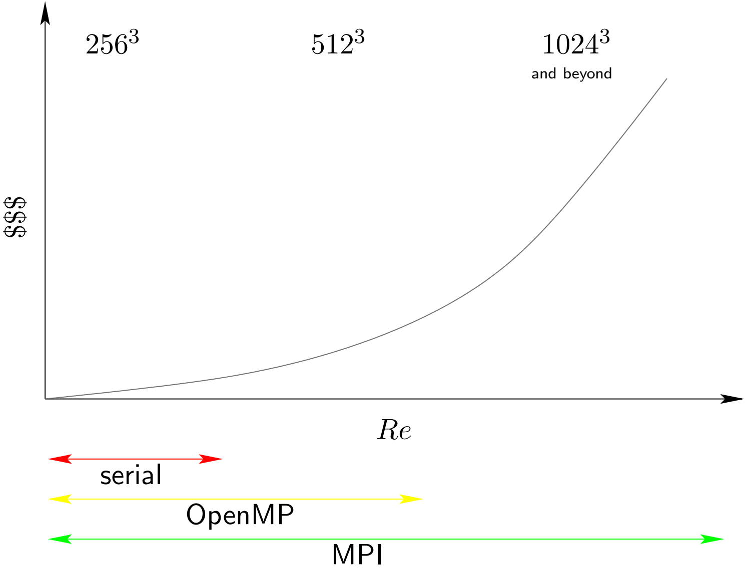
$$N_t \sim \frac{T}{\Delta t} \sim \frac{T}{\Delta x/u} \sim \frac{T}{l/u} Re^{3/4}$$

$$\$\$\$ \sim N_x N_t \sim \left(\frac{T}{l/u}\right)\left(\frac{2\pi}{l}\right)^3 Re^3$$

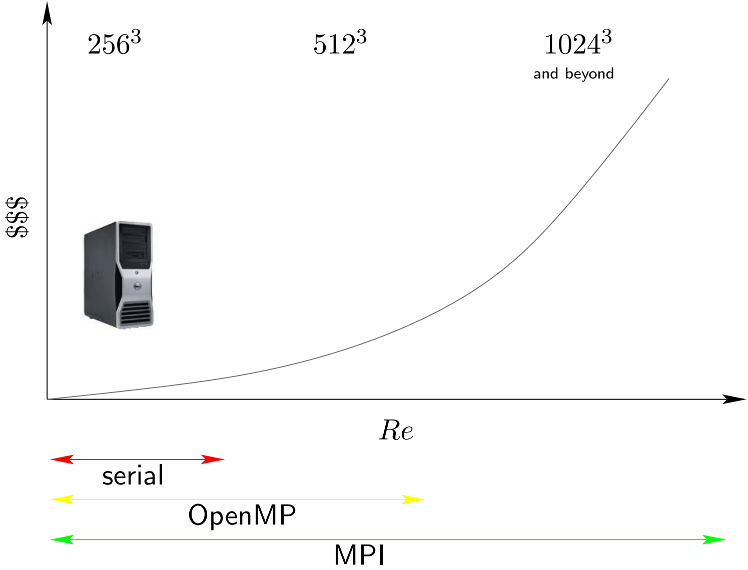
DNS: computational costs IV



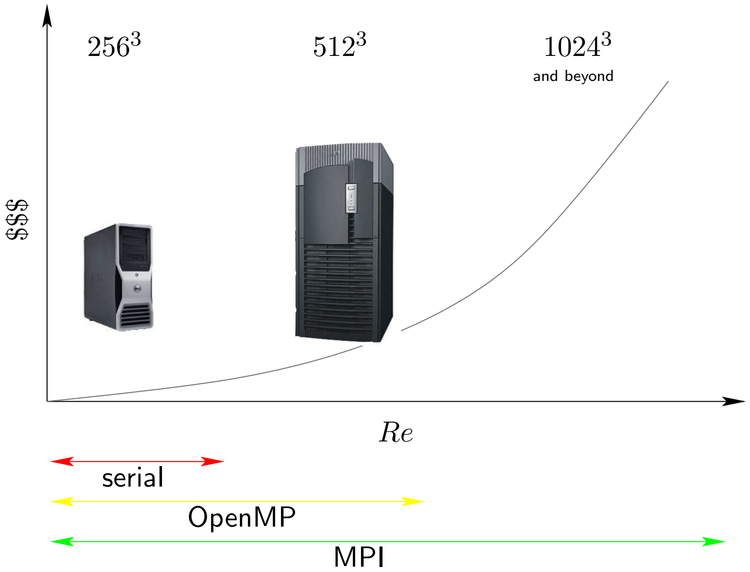
DNS: computational costs V



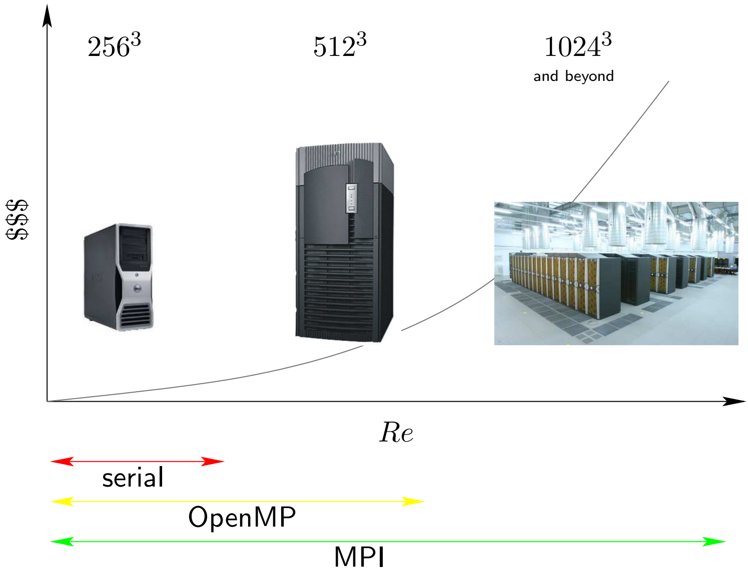
DNS: computational costs V



DNS: computational costs V



DNS: computational costs V



DNS: OpenMP vs. MPI

OpenMP

- “fork and join” principle
- easy
- incremental parallelization
- medium number of cores
- limited degree of scalability

MPI

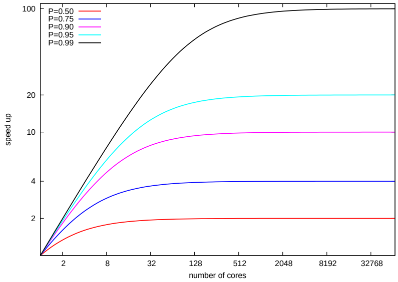
- decomposition of computational domain
- “communicate when necessary” principle
- broad range of application
- high scalability

Amdahl's Law

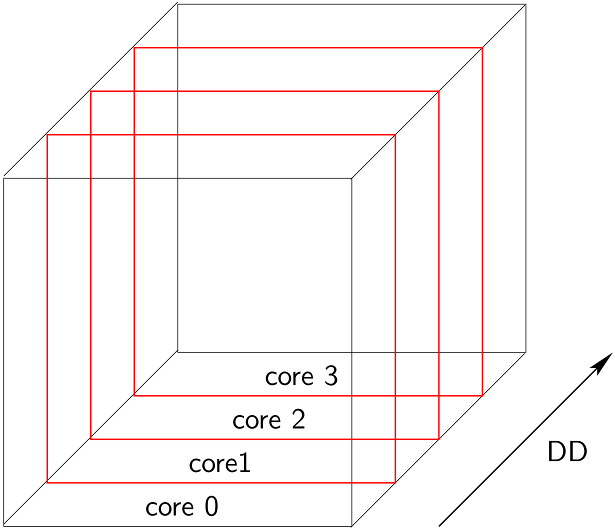
- consider serial code with runtime 1
- P denotes the parallelizable fraction of code
- runtime on n cores: $(1-P)+\frac{P}{N}$
- maximum expectable speed up $S(N)=\frac{1}{(1-P)+\frac{P}{N}}$

Amdahl's Law

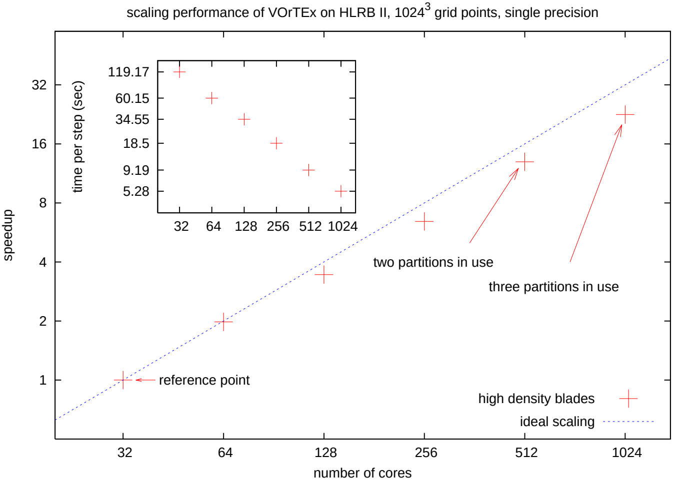
- consider serial code with runtime 1
- P denotes the parallelizable fraction of code
- runtime on n cores: $(1-P)+\frac{P}{N}$
- maximum expectable speed up $S(N)=\frac{1}{(1-P)+\frac{P}{N}}$



DNS: parallelization via MPI



DNS: scaling



scaling performance of VOrTEx on HLRB II, 1024^3 grid points, single precision
time per step (sec)
119.17
60.15
34.55
18.5
9.19
5.28
32
64
128
256
512
1024
speedup
32
16
8
4
2
1
reference point
two partitions in use
three partitions in use
high density blades
ideal scaling
32
64
128
256
512
1024
number of cores

Demos

- Hello OpenMP/ MPI world
- HLRBII @ LRZ
- remote visualization