Efficient Computation of Large-Scale Statistical Solutions to Incompressible Fluid Flows

Tobias Rohner, Siddhartha Mishra

June 04, 2024

Equations

Incompressible Navier-Stokes Equations

$$\partial_t \mathbf{u} + \nabla \cdot (\mathbf{u} \otimes \mathbf{u}) + \nabla p = \frac{1}{\mathrm{Re}} \Delta \mathbf{u}$$

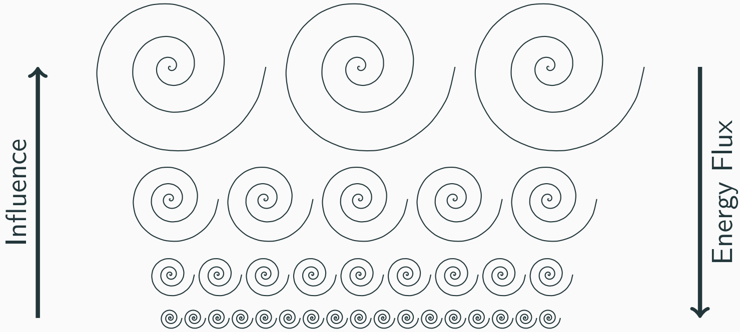
$$\nabla \cdot \mathbf{u} = 0$$

$$\mathbf{u} \,\big|_{t=0} = \mathbf{u}_0$$

- $\mathbf{u}$ flow velocity
- p pressure
- Re Reynold's Number

Vortex Stretching

$$\partial_t \omega + (\mathbf{u} \cdot \nabla)\omega = \underbrace{(\omega \cdot \nabla)\mathbf{u}}_{\text{Vortex Stretching Term}} + \frac{1}{\mathrm{Re}} \Delta \omega$$

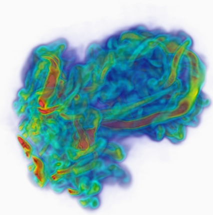


The Problem with Classical Simulations

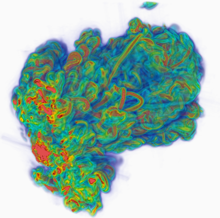
Problem

DNS needs to resolve length scales $\Delta x \ll \mathrm{Re}^{-\frac{3}{4}}$, $N_{\mathsf{dof}} \sim \mathrm{Re}^{\frac{9}{4}}$

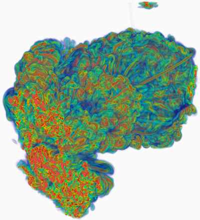
- What happens if we run an underresolved simulation?



$N = 64$

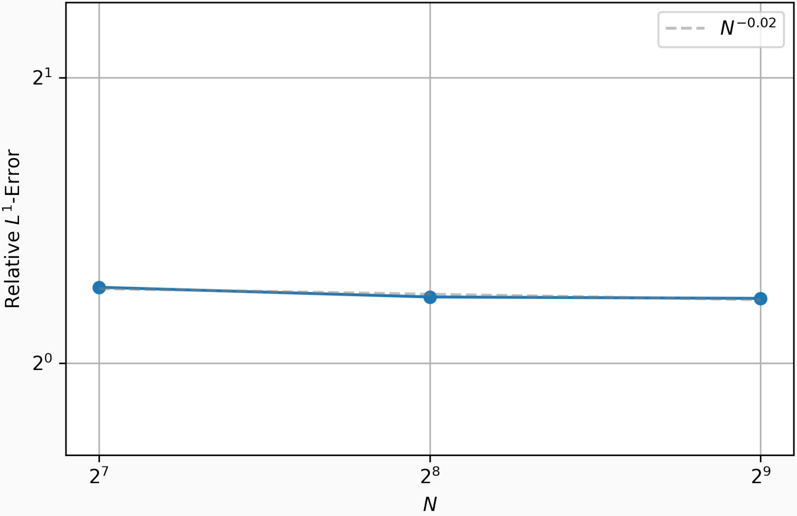


$N = 128$



$N = 256$

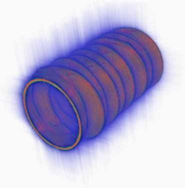
Lack of Convergence



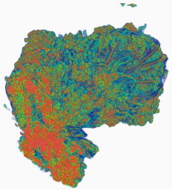
Random Initial Conditions

$$\mathbf{u}_0(x;\sigma) = \mathbf{u}_0(x) + \varepsilon(\sigma)$$

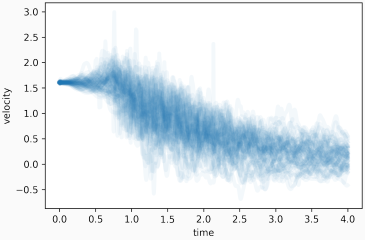
- σ Random Variable
- $\mathbf{u}_0$ classical initial condition
- ε perturbation



$t = 0$

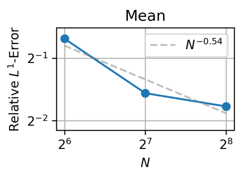


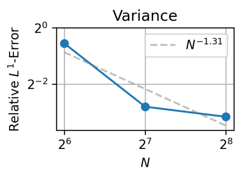
$t = 1$

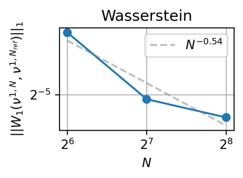


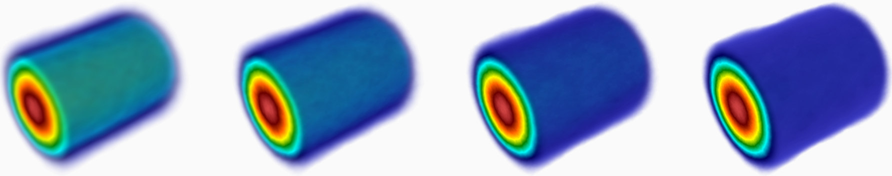
Properties of Statistical Solutions

- Convergence under mesh refinement
- Convergence when reducing perturbation amplitude
- Stability for perturbation types



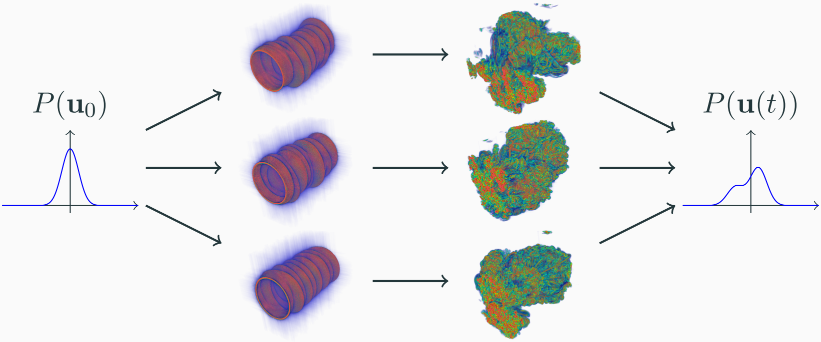






$N = 64$ $\quad$ $N = 128$ $\quad$ $N = 256$ $\quad$ $N = 512$

How to Compute a Statistical Solution

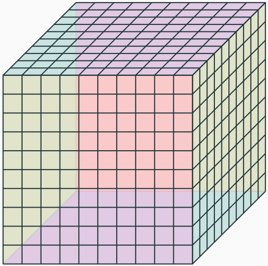


Challenge

Becomes highly computationally expensive!

Some Simplifications

1. Restrict the domain to $[0,1]^d$
2. Enforce *periodic* boundary conditions
3. Compute the solution on an uniform grid
4. Trade resolution for Monte Carlo samples



Spectral Viscosity

$$\begin{cases} \partial_t \hat{\mathbf{u}}_{\mathbf{k}} + \left(\mathbb{1} - \frac{\mathbf{k}\mathbf{k}^\intercal}{|\mathbf{k}|^2}\right) \cdot i\mathbf{k}^\intercal \cdot \hat{\mathbf{B}}_{\mathbf{k}} & = -\varepsilon_N |\mathbf{k}|^2 \hat{\mathbf{u}}_{\mathbf{k}} \\ \hat{\mathbf{u}}_{\mathbf{k}} \,\big|_{t=0} & = \hat{\mathbf{u}}_{0,\mathbf{k}} \end{cases}$$

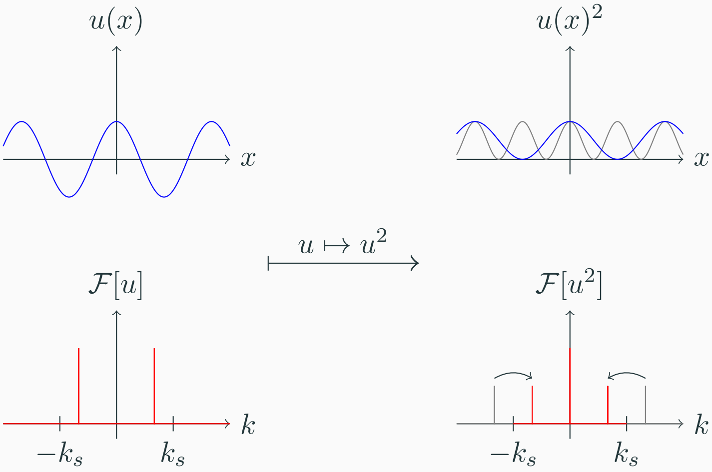
with $\mathbf{B} = \mathbf{u} \otimes \mathbf{u}$.

Computational Cost

- M Samples
- Sample Cost $\mathcal{O}(N^{d+1} \log N)$

Total Cost $\mathcal{O}(MN^{d+1} \log N) = \mathcal{O}(MN^4 \log N)$

Aliasing



Computational Kernels

Want to Solve

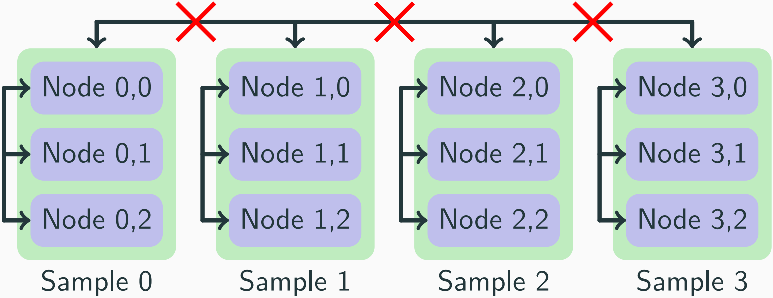
$$\partial_t \hat{\mathbf{u}}_{\mathbf{k}} + \left(\mathbb{1} - \frac{\mathbf{k}\mathbf{k}^\intercal}{|\mathbf{k}|^2}\right) \cdot i\mathbf{k}^\intercal \cdot \hat{\mathbf{B}}_{\mathbf{k}} = -\varepsilon_N |\mathbf{k}|^2 \hat{\mathbf{u}}_{\mathbf{k}}$$

1. Pad $\hat{\mathbf{u}}$ — $\mathcal{O}(N^d)$
2. $\mathbf{u} = \mathcal{F}^{-1}[\hat{\mathbf{u}}]$ — $\mathcal{O}(N^d \log N)$
3. $\mathbf{B} = \mathbf{u} \otimes \mathbf{u}$ — $\mathcal{O}(N^d)$
4. $\hat{\mathbf{B}} = \mathcal{F}[\mathbf{B}]$ — $\mathcal{O}(N^d \log N)$
5. Unpad $\hat{\mathbf{B}}$ — $\mathcal{O}(N^d)$
6. Compute $\partial_t \hat{\mathbf{u}}$ — $\mathcal{O}(N^d)$

Bottleneck

Kernels are bandwidth limited $\implies$ Run everything on the GPU

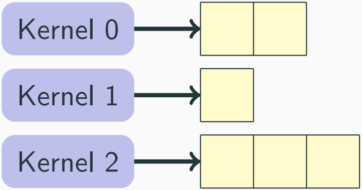
Parallelization Strategy

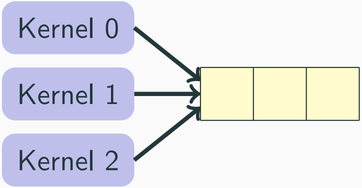


Important

Prefer parallelization over samples rather than splitting up the domain!

Single-Rank Solver

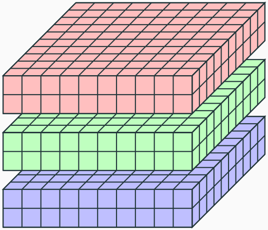




- Naive version
- Each Kernel has its own memory
- Lots of wasted resources

- Optimized version
- All Kernels share a buffer
- 12.5% less memory in single-rank solver
- 50% less memory in distributed solver

Distributed Solver

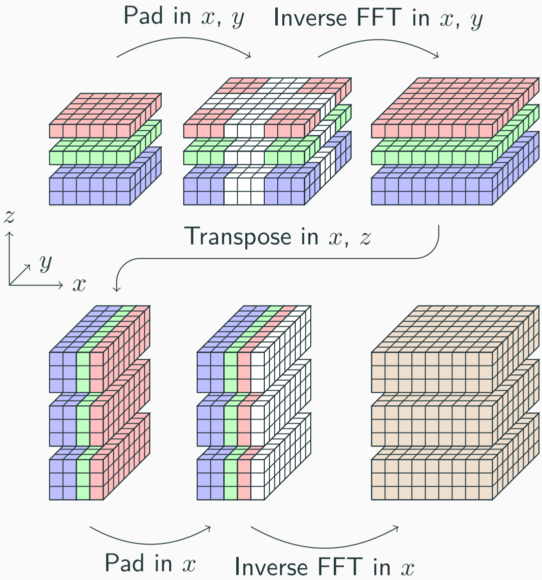


Slab Decomposition ($N = 10$, $p = 3$)

Important Questions

- How to minimize communication?
- How to pad the data?
- Are there some other optimizations?

Padded FFT

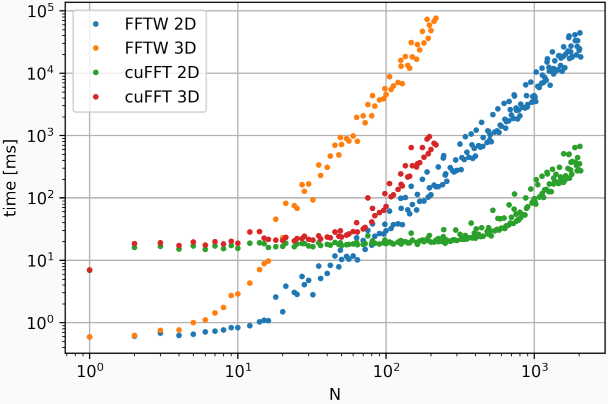


Pad in x, y
Inverse FFT in x, y
Transpose in x, z
z
y
x
Pad in x
Inverse FFT in x

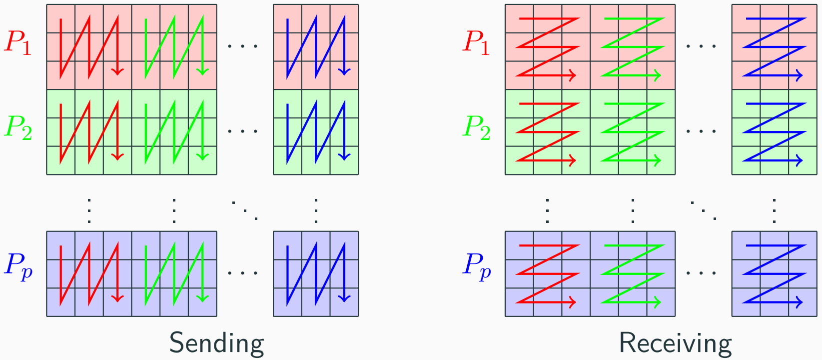
FFT Optimization

FFTW & cuFFT

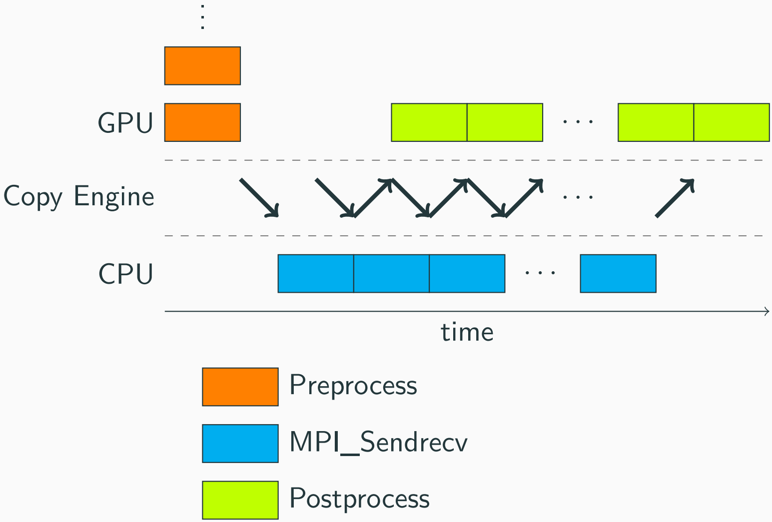
- "Good" sizes: $N = 2^a 3^b 5^c 7^d$
- But sometimes $N' > N$ is faster!



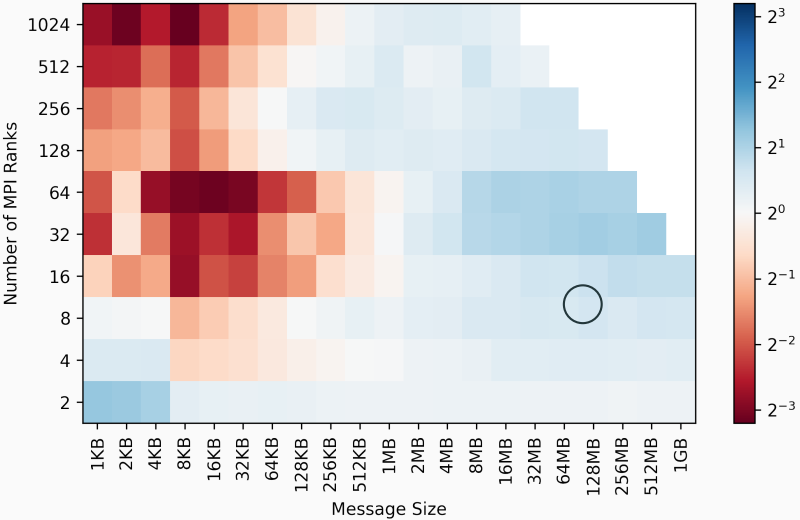
Transpose



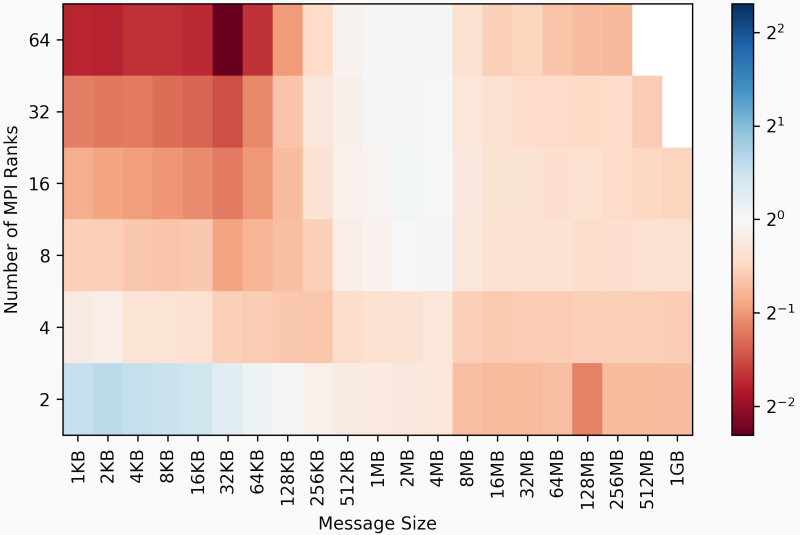
Transpose – Task Scheduling



Alltoall Communication

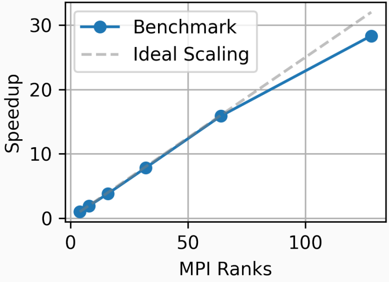


Alltoall Communication

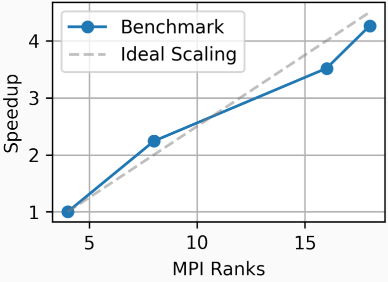


Strong Scaling

- Scaling over MC samples
- Base Case:
  - $N = 512$
  - $M = 128$
  - $p = 4$

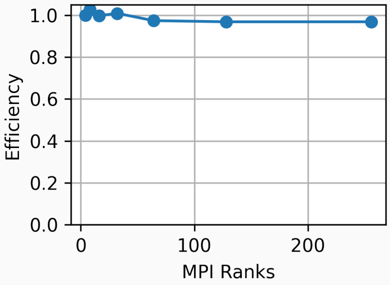


- Scaling over domain
- Base Case:
  - $N = 512$
  - $M = 1$
  - $p = 4$

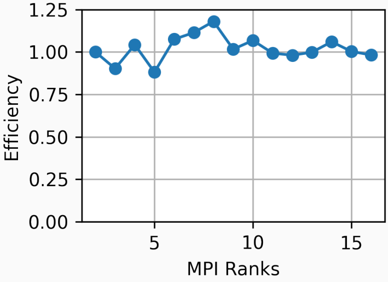


Weak Scaling

- Scaling over MC samples
- Base Case:
  - $N = 512$
  - $M = 128$
  - $p = 4$

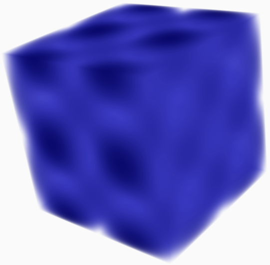


- Scaling over domain
- Base Case:
  - $N = 512$
  - $M = 1$
  - $p = 4$

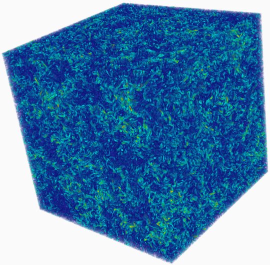


Experiments – Taylor-Green

$$u_0(x, y, z) = \cos(2\pi x)\sin(2\pi y)\sin(2\pi z)$$
$$v_0(x, y, z) = -\sin(2\pi x)\cos(2\pi y)\sin(2\pi z)$$
$$w_0(x, y, z) = 0$$



$t = 0$

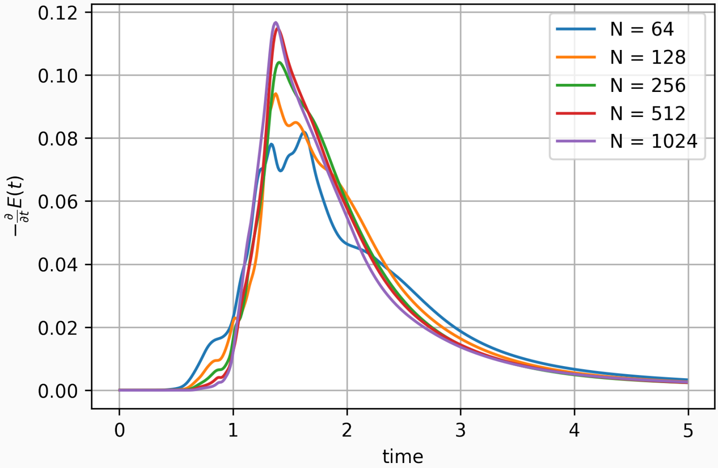


$t = 5$

Experiments – Taylor-Green

K41

$$\lim_{\nu\to 0} \left\langle \nu \left\| \nabla \mathbf{u}^{\nu} \right\|_{L_x^2}^2 \right\rangle = \varepsilon_0 > 0$$

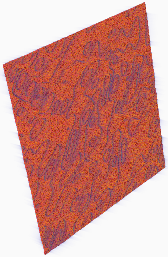


Experiments – Shear Layer

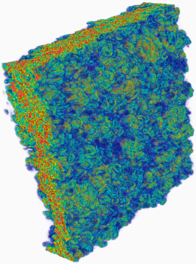
$$u_0(x,y,z) = \begin{cases} 1 & \text{if } z \leq \frac{1}{2} \\ -1 & \text{if } z > \frac{1}{2} \end{cases}$$

$$v_0(x,y,z) = 0$$

$$w_0(x,y,z) = 0$$

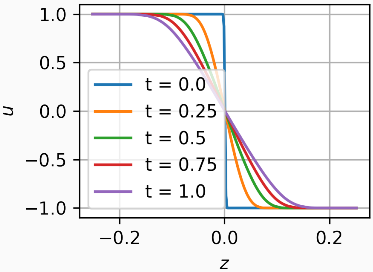


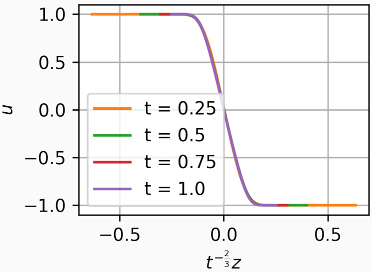
$t = 0$



$t = 1$

Experiments – Shear Layer





Takeaway

Statistical moments seem to behave very well. So although we are not able to really describe turbulence, I am confident that for statistics it is possible to at least at some degree!

Thank you!