An overview of Fourier pseudospectral methods

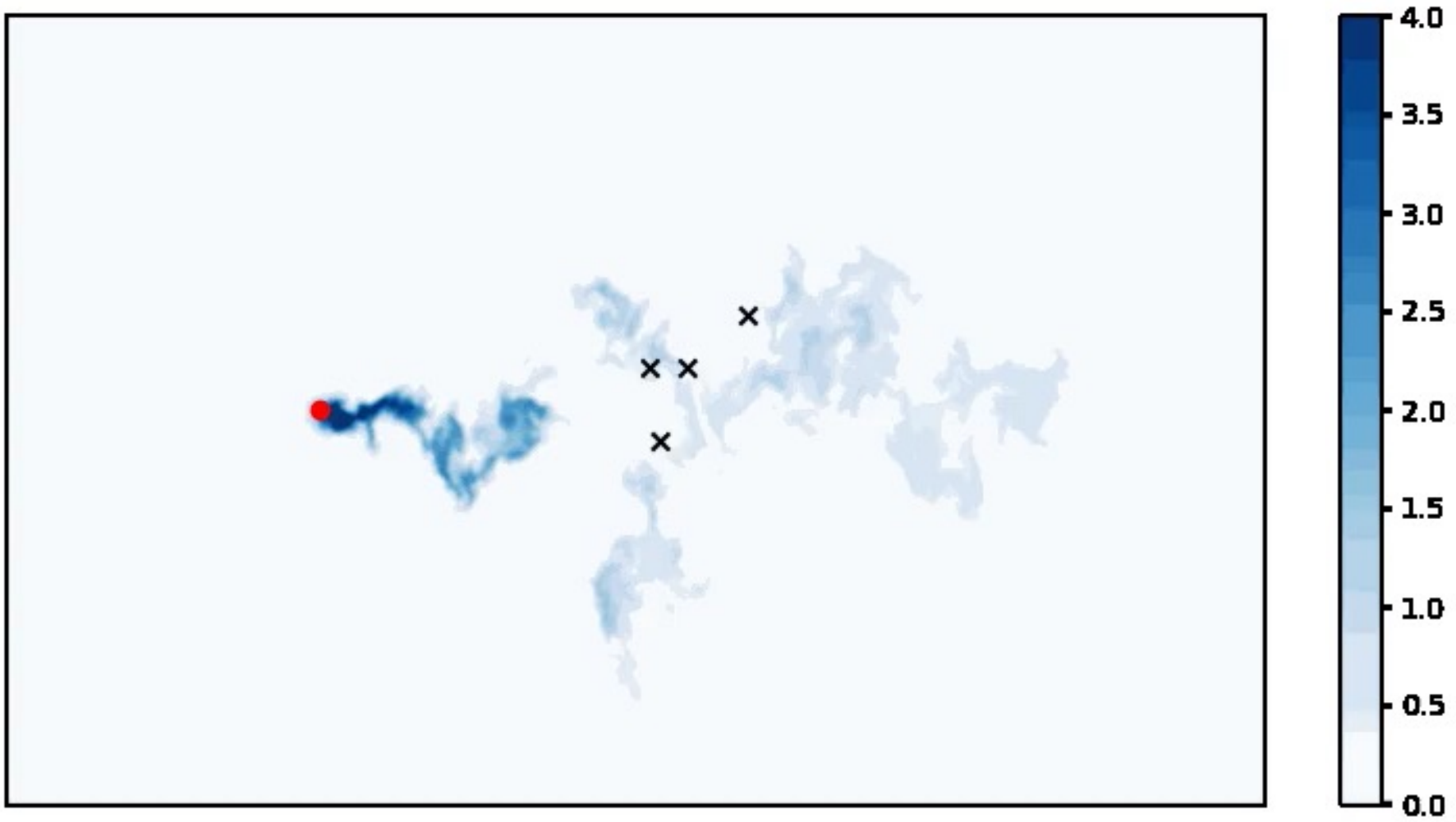
Numerical simulation of passive scalar turbulence

Maurizio Carbone

April 9, 2024
Univ. Tor Vergata, Rome

Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA
Consiglio Nazionale delle Ricerche

A database of turbulent fields for olfactory search

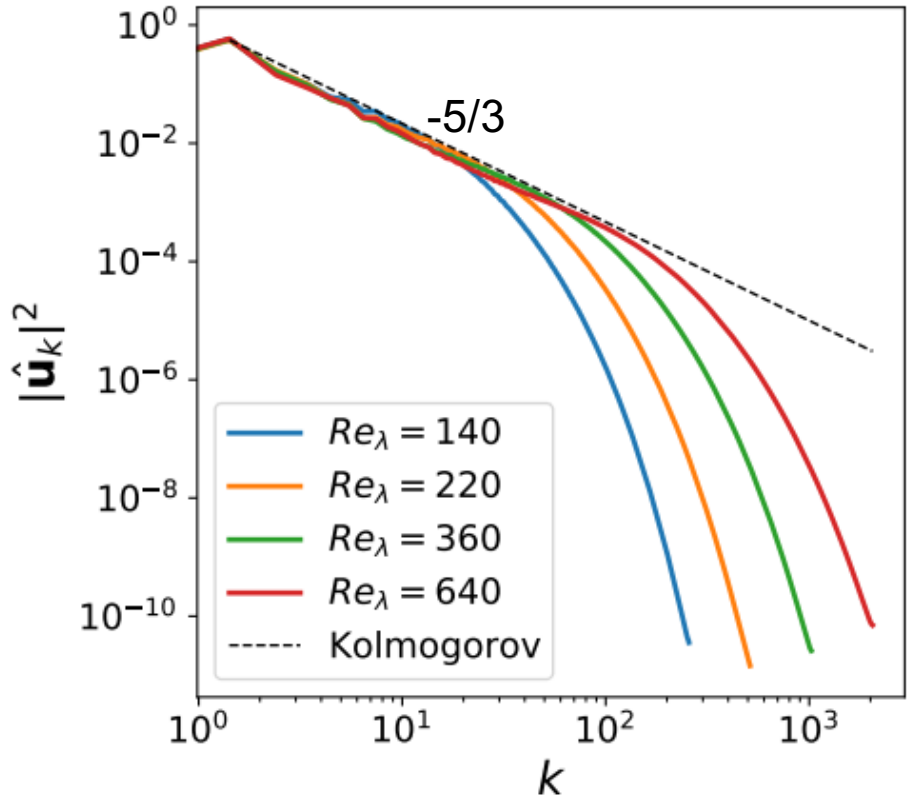


Incompressible Navier-Stokes turbulence

$$\nabla_i u_i = 0$$

$$\partial_t u_i + \nabla_j \left(u_i u_j\right) = -\nabla_i P + \nu \nabla^2 u_i + f_i$$

Velocity ***u(x,t)***
Pressure by density *P(**x**,t)*
*External forcing **f**(**x**,t)*



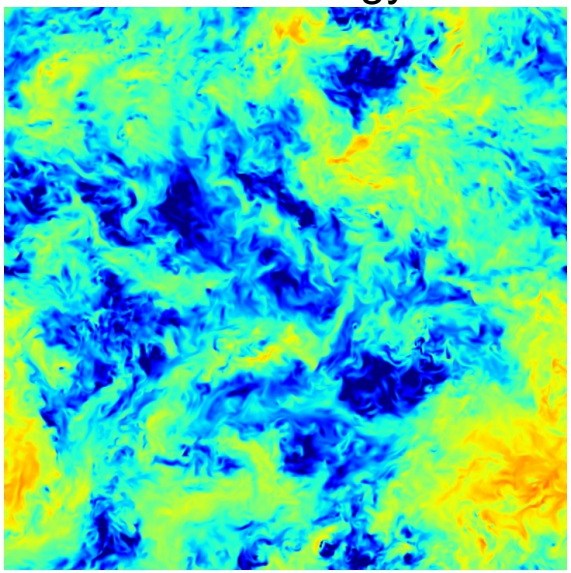
Incompressible Navier-Stokes turbulence

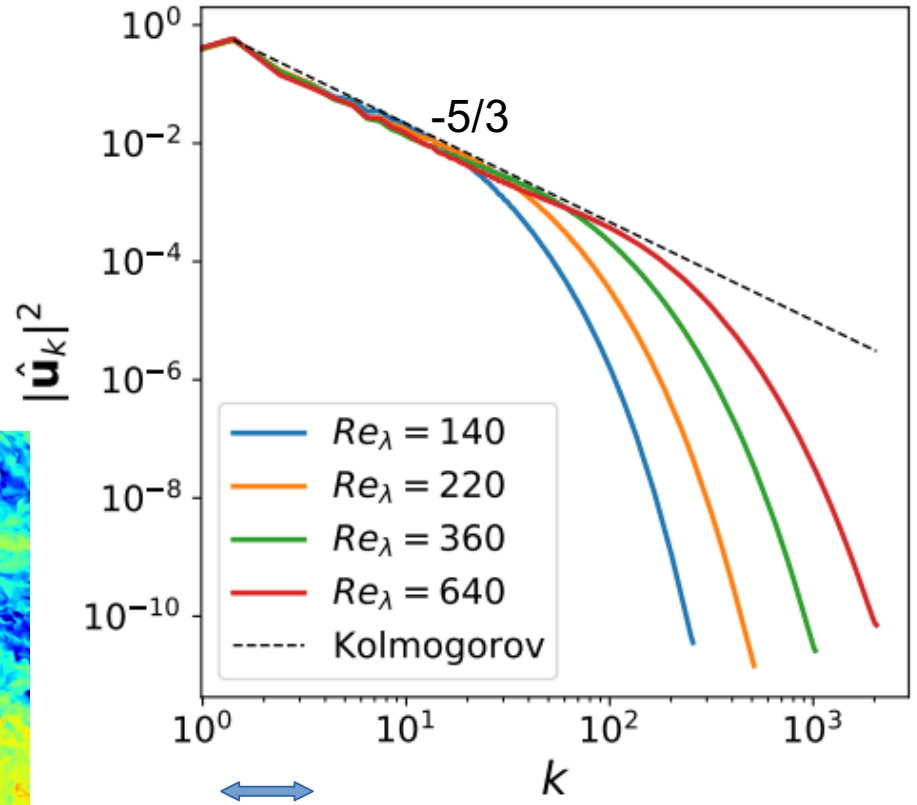
$$\nabla_i u_i = 0$$

$$\partial_t u_i + \nabla_j \left(u_i u_j\right) = -\nabla_i P + \nu \nabla^2 u_i + f_i$$

Velocity ***u(x,t)***
Pressure by density *P(**x**,t)*
*External forcing **f**(**x**,t)*

Kinetic energy





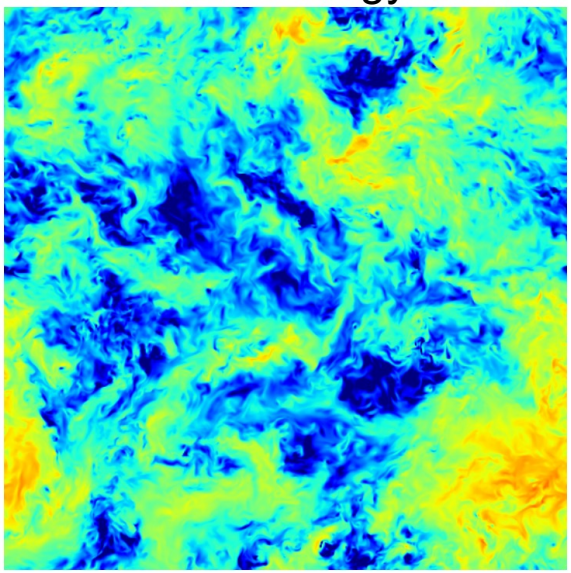
Incompressible Navier-Stokes turbulence

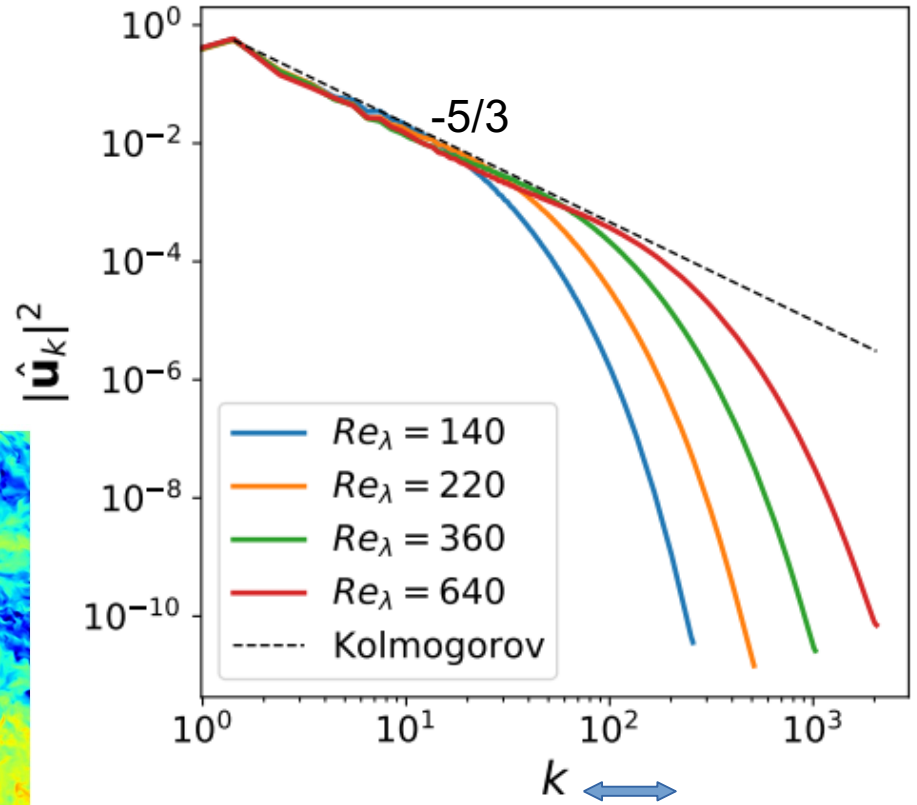
$$\nabla_i u_i = 0$$

$$\partial_t u_i + \nabla_j \left(u_i u_j\right) = -\nabla_i P + \nu \nabla^2 u_i + f_i$$

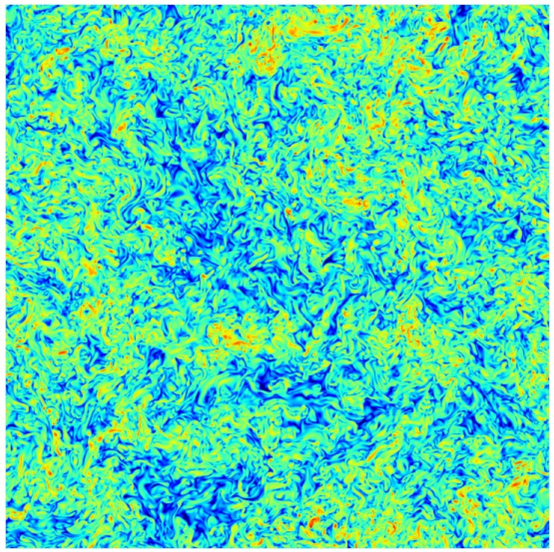
Velocity ***u(x,t)***
Pressure by density *P(**x**,t)*
*External forcing **f**(**x**,t)*

Kinetic energy





Enstrophy



Fourier representation

$$\hat{\boldsymbol{u}}(\boldsymbol{k},t) = \frac{1}{L^3}\int \mathrm{d}\boldsymbol{x}\boldsymbol{u}(\boldsymbol{x},t)\exp(-\mathrm{i}\boldsymbol{k}\cdot\boldsymbol{x})$$

- Employ FFT algorithm
- Efficient, versatile, easy to parallelize
- Discretize: Wavevector **k** = ***k*** *Δk* and position vector ***x*** = ***j*** *Δx*

Fourier representation

- Employ FFT algorithm

$$\hat{\boldsymbol{u}}(\boldsymbol{k},t)=\frac{1}{L^3}\int \mathrm{d}\boldsymbol{x}\boldsymbol{u}(\boldsymbol{x},t)\exp(-\mathrm{i}\boldsymbol{k}\cdot\boldsymbol{x})$$

- Efficient, versatile, easy to parallelize
- Discretize: Wavevector **k** = ***k*** Δk and position vector ***x*** = ***j*** Δx

$$\hat{u}_i(\boldsymbol{k},t)=\frac{1}{N^d}\sum_{\boldsymbol{j}} u_i(\boldsymbol{j}\Delta x,t)\exp(-\mathrm{i}\boldsymbol{j}\cdot\boldsymbol{k}\Delta x\Delta k)$$

$$\Delta x\Delta k=\frac{2\pi}{N}$$

Choice to make FFT work
Domain length *L* discretized using *N* points

Fourier representation

- Employ FFT algorithm

$$\hat{\boldsymbol{u}}(\boldsymbol{k},t) = \frac{1}{L^3}\int \mathrm{d}\boldsymbol{x}\boldsymbol{u}(\boldsymbol{x},t)\exp(-\mathrm{i}\boldsymbol{k}\cdot\boldsymbol{x})$$

- Efficient, versatile, easy to parallelize
- Discretize: Wavevector **k** = ***k*** *Δk* and position vector ***x*** = ***j*** *Δx*

$$\hat{u}_i(\boldsymbol{k},t) = \frac{1}{N^d}\sum_{\boldsymbol{j}} u_i(\boldsymbol{j}\Delta x,t)\exp(-\mathrm{i}\boldsymbol{j}\cdot\boldsymbol{k}\Delta x\Delta k)$$

$$\Delta x\Delta k = \frac{2\pi}{N}$$

Choice to make FFT work
Domain length *L* discretized using *N* points

- Discrete FFT of Navier-Stokes:

$$k_i\hat{u}_i = 0$$

$$\partial_t\hat{u}_i(\boldsymbol{k},t) = Q_{ijl}(\boldsymbol{k})\sum_{\boldsymbol{p}+\boldsymbol{q}=\boldsymbol{k}}\hat{u}_j(\boldsymbol{p},t)\hat{u}_l(\boldsymbol{q},t) - \nu\|\boldsymbol{k}\|^2\hat{u}_i(\boldsymbol{k},t) + \hat{f}_i(\boldsymbol{k},t)$$

Convolution sums and related issues

$$k_i \hat{u}_i = 0$$

$$\partial_t \hat{u}_i(\boldsymbol{k}, t) = Q_{ijl}(\boldsymbol{k}) \sum_{\boldsymbol{p}+\boldsymbol{q}=\boldsymbol{k}} \hat{u}_j(\boldsymbol{p}, t) \hat{u}_l(\boldsymbol{q}, t) - \nu k^2 \hat{u}_i(\boldsymbol{k}, t) + \hat{f}_i(\boldsymbol{k}, t)$$

$$Q_{ijl} = -\mathrm{i} k_l \left(\delta_{ij} - \frac{k_i k_j}{\|\boldsymbol{k}\|^2} \right)$$

- Lack of analytic solutions: Burgers eq. for particular Q_{ijl}

(cf. Lorenz system)

Convolution sums and related issues

$$k_i \hat{u}_i = 0$$

$$\partial_t \hat{u}_i(\boldsymbol{k}, t) = Q_{ijl}(\boldsymbol{k}) \sum_{\boldsymbol{p}+\boldsymbol{q}=\boldsymbol{k}} \hat{u}_j(\boldsymbol{p}, t)\hat{u}_l(\boldsymbol{q}, t) - \nu k^2 \hat{u}_i(\boldsymbol{k}, t) + \hat{f}_i(\boldsymbol{k}, t)$$

$$Q_{ijl} = -\mathrm{i}k_l \left(\delta_{ij} - \frac{k_i k_j}{\|\boldsymbol{k}\|^2} \right)$$

- Lack of analytic solutions: Burgers eq. for particular Q_{ijl}

(cf. Lorenz system)

- Energy and enstrophy transfer (dimensionality-dependent)
- Skewness and intermittency of the velocity gradients

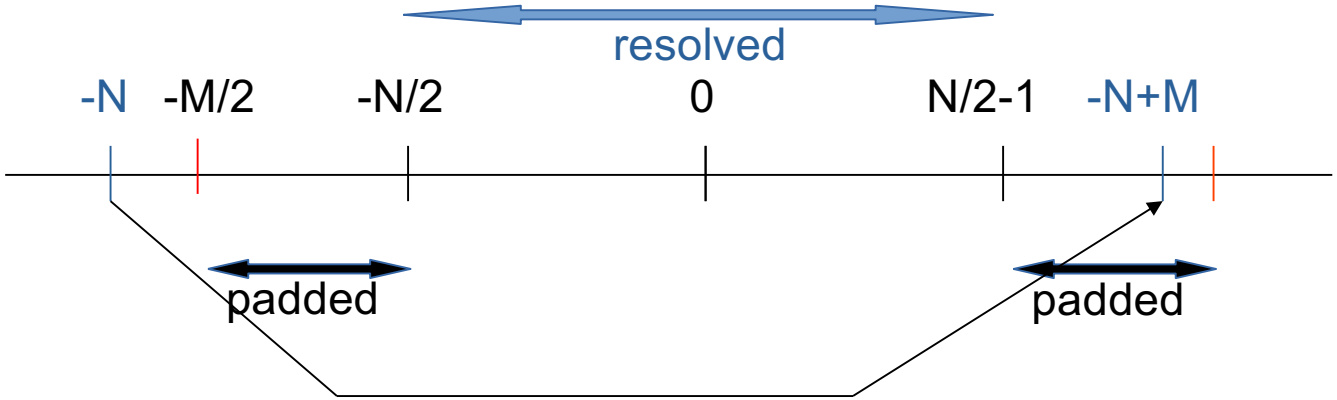
- Products in physical space rather than direct convolution
- Aliasing removal

Building a Fourier pseudospectral code

Aliasing removal by padding

$$\sum_{\boldsymbol{p}+\boldsymbol{q}=\boldsymbol{k}} \hat{u}_j(\boldsymbol{p},t)\hat{u}_l(\boldsymbol{q},t) = \left[\widehat{u_j(\boldsymbol{x},t)u_l(\boldsymbol{x},t)}\right](\boldsymbol{k},t)$$

- $-N/2 \leq k_i < N/2$
- $-M/2 \leq p_i, q_i < M/2$



Aliasing removal by padding

$$\sum_{\boldsymbol{p}+\boldsymbol{q}=\boldsymbol{k}} \hat{u}_j(\boldsymbol{p},t)\hat{u}_l(\boldsymbol{q},t) = \left[\widehat{u_j(\boldsymbol{x},t)u_l(\boldsymbol{x},t)}\right](\boldsymbol{k},t)$$

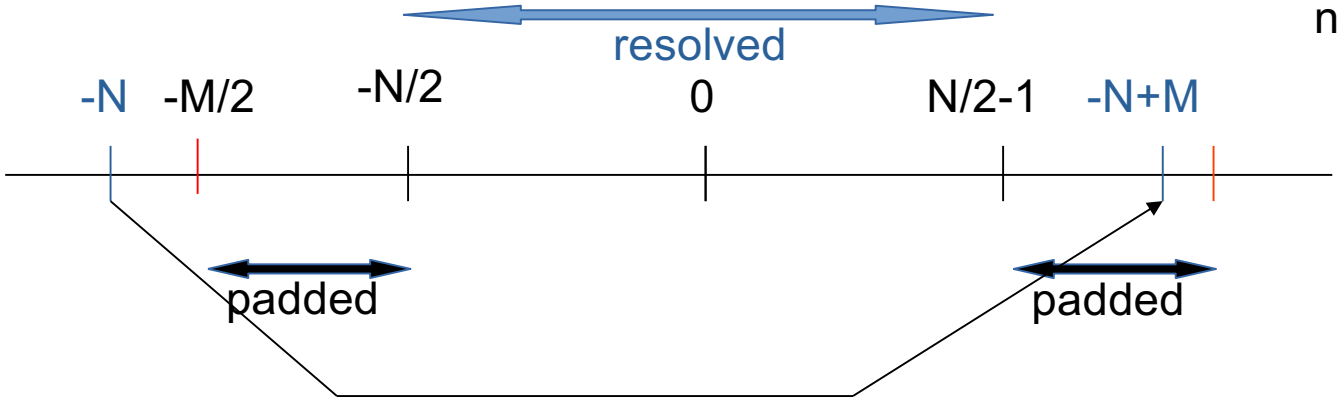
$$-N/2 \le k_i < N/2$$

$$-M/2 \le p_i, q_i < M/2$$

$$-\frac{N}{2} - \frac{N}{2} + M > \frac{N}{2} - 1$$

$$M \ge \frac{\alpha+1}{2}N$$

α: degree of nonlinearity



Aliasing removal by padding

$$\sum_{\boldsymbol{p}+\boldsymbol{q}=\boldsymbol{k}} \hat{u}_j(\boldsymbol{p},t)\hat{u}_l(\boldsymbol{q},t) = \left[\widehat{u_j(\boldsymbol{x},t)u_l(\boldsymbol{x},t)}\right](\boldsymbol{k},t)$$

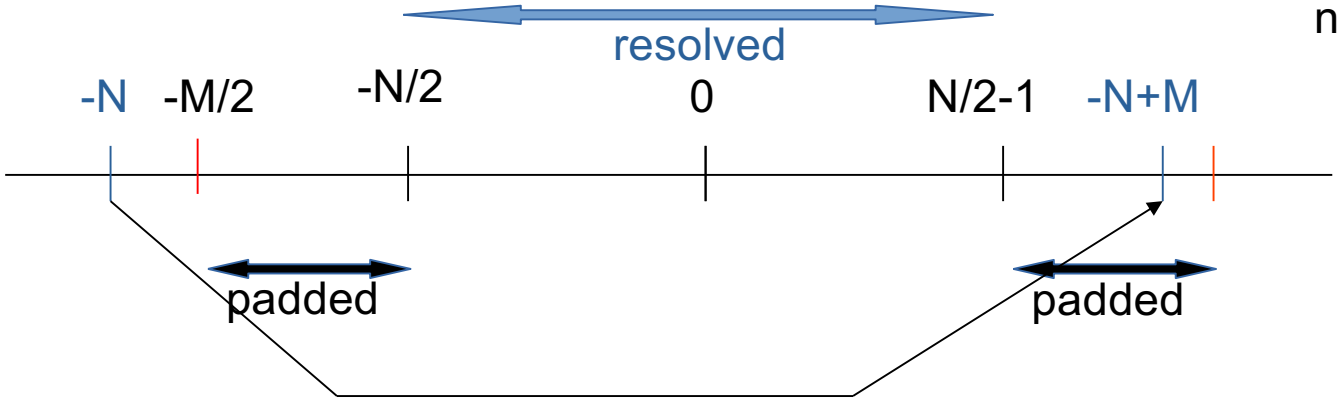
$$-N/2 \leq k_i < N/2$$

$$-M/2 \leq p_i, q_i < M/2$$

$$-\frac{N}{2} - \frac{N}{2} + M > \frac{N}{2} - 1$$

$$M \geq \frac{\alpha+1}{2}N$$

α: degree of nonlinearity

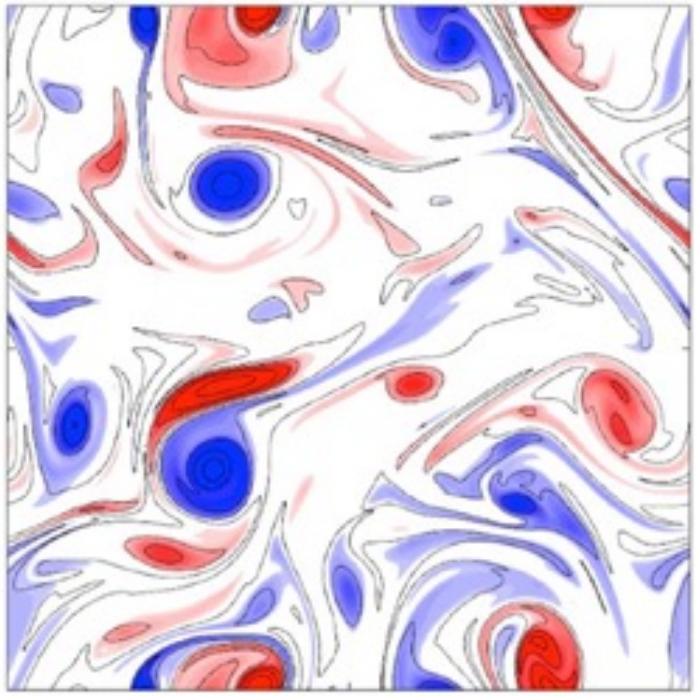


- Removal by truncation [2/3 rule] (padding preferable [3/2 rule])
- Alternatives: phase shift, implicit, ..
- ..or combined with some approximations (Rogallo 1977)

Using incompressibility: 2D

$$
\begin{aligned}
u_1 &= -\nabla_2 \psi \\
u_2 &= \nabla_1 \psi \\
\partial_t \nabla^2 \psi &= -\nabla_1 \nabla^2 \psi \nabla_2 \psi + \nabla_2 \nabla^2 \psi \nabla_1 \psi + \nu \nabla^2 \nabla^2 \psi
\end{aligned}
$$

- All info in one scalar function ***u*** = ***Rot(*** $\psi \mathbf{e}_z$ ***)***

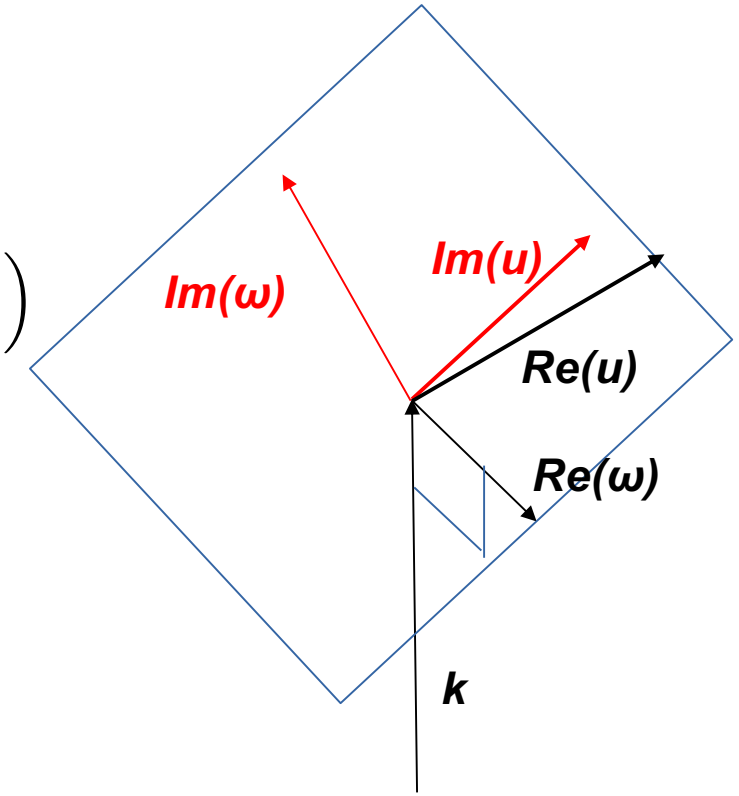


Using incompressibility: 3D

$$\nabla_j \left(u_i u_j + P\delta_{ij}\right) = \nabla_j \left(u_i u_j - \frac{u_k u_k}{3}\delta_{ij} + P'\delta_{ij}\right)$$

$$C_{ij} = u_i u_j - \frac{u_k u_k}{3}\delta_{ij}$$

$$\hat{C}_{33} = -\hat{C}_{11} - \hat{C}_{22}$$



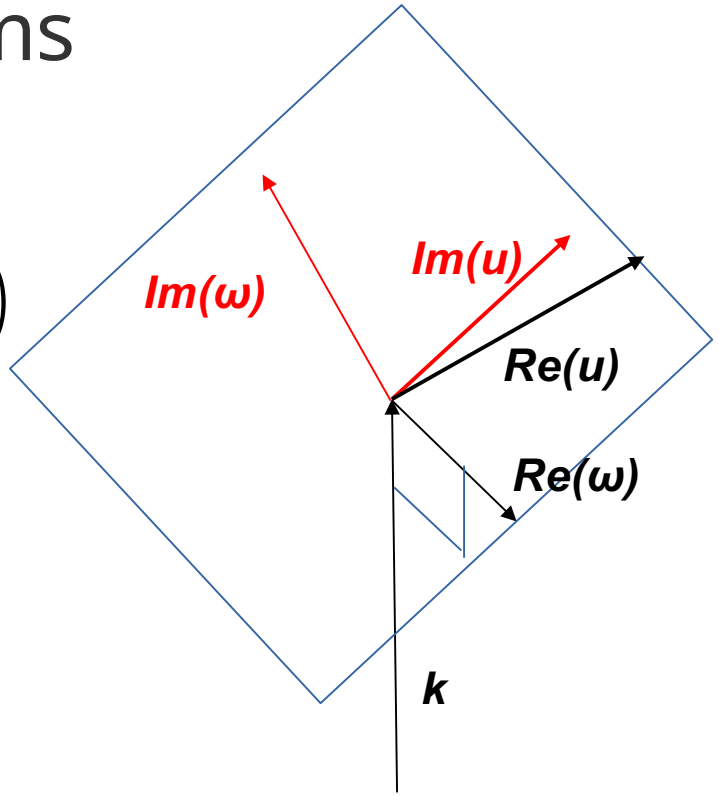
- Craya decomposition: ***u*** actually 2D
- Only traceless part of convective C_{ij} matters
- C_{ij} traceless: We can spare one FFT (Rogallo 1981, Basdevant 1983)
- 3+6-1 = 8 FFT for one time step

Formulation of convective terms

$$\nabla_j \left(u_i u_j + P\delta_{ij}\right) = \nabla_j \left(u_i u_j - \frac{u_k u_k}{3}\delta_{ij} + P'\delta_{ij}\right)$$

$$C_{ij} = u_i u_j - \frac{u_k u_k}{3}\delta_{ij}$$

$$\hat{C}_{33} = -\hat{C}_{11} - \hat{C}_{22}$$



- Unstable (avoid ***u*grad(…)***): $\boldsymbol{\nabla \cdot (uu) = u \cdot \nabla u}$
- Stable, great for helical flows: $\boldsymbol{\nabla \cdot (uu) = \omega \times u + \nabla \frac{u \cdot u}{2}}$

Common types of forcing

- Stochastic: dissipation rate through FDN (Furutsu 1963)

$$\left\langle \hat{f}_i(\boldsymbol{k},t)\hat{f}_j(\boldsymbol{k}',t')\right\rangle = \delta_{ij}\delta(\boldsymbol{k}+\boldsymbol{k}')\hat{R}(\boldsymbol{k})$$

$$\langle u_i f_i\rangle = \nu\left\langle \nabla_j u_i \nabla_j u_i\right\rangle = \frac{1}{2}R_{ii}$$

- Keep constant energy (adjust amplitudes within K_F)

$$\hat{\boldsymbol{u}}_F = \hat{\boldsymbol{u}}'_F\sqrt{1+\frac{E'-\overline{E}}{E'_F}}$$

- Statistically steady input energy input rate

$$\hat{\boldsymbol{f}}(\boldsymbol{k},t) = \epsilon\frac{\hat{\boldsymbol{u}}(\boldsymbol{k},t)}{\sum_{K_F}\|\hat{\boldsymbol{u}}\|^2},\ \boldsymbol{k}\in K_F$$

Time integration: Exponential Runge Kutta

- Integrating factor

$$\partial_t \left(\boldsymbol{u} \exp(\nu k^2 t) \right) = \exp(\nu k^2 t) \boldsymbol{F}[\boldsymbol{u}, t]$$

- Integrate and discretize rhs using Lagrangian polynomials

Time integration: Exponential Runge Kutta

- Integrating factor

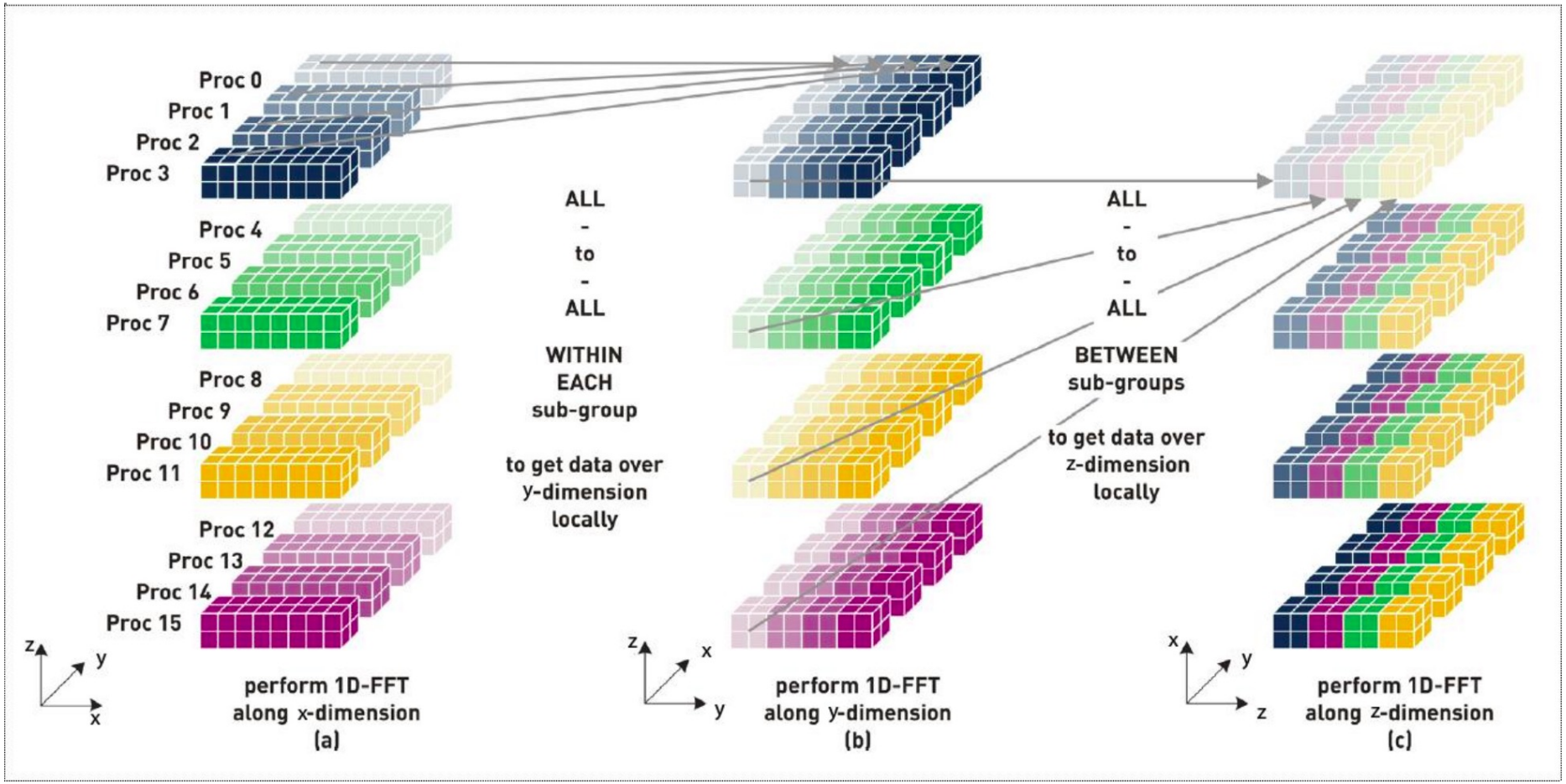
$$\partial_t \left(\boldsymbol{u} \exp(\nu k^2 t) \right) = \exp(\nu k^2 t) \boldsymbol{F}[\boldsymbol{u}, t]$$

- Integrate and discretize rhs using Lagrangian polynomials
- E.g., second-order exponential Runge Kutta (Hochbruck 2010)

$$\boldsymbol{u}_1' = \varphi_0 \boldsymbol{u}_0 + \varphi_1 \boldsymbol{F}_0$$
$$\boldsymbol{u}_1 = \varphi_0 \boldsymbol{u}_0 + (\varphi_1 - \varphi_2) \boldsymbol{F}[\boldsymbol{u}_0, t] + \varphi_2 \boldsymbol{F}[\boldsymbol{u}_1', t + \Delta t]$$
$$z = -\nu k^2 \Delta t \quad \varphi_0 = e^z \quad \varphi_1 = \frac{e^z - 1}{z} \quad \varphi_2 = \frac{e^z - z - 1}{z^2}$$

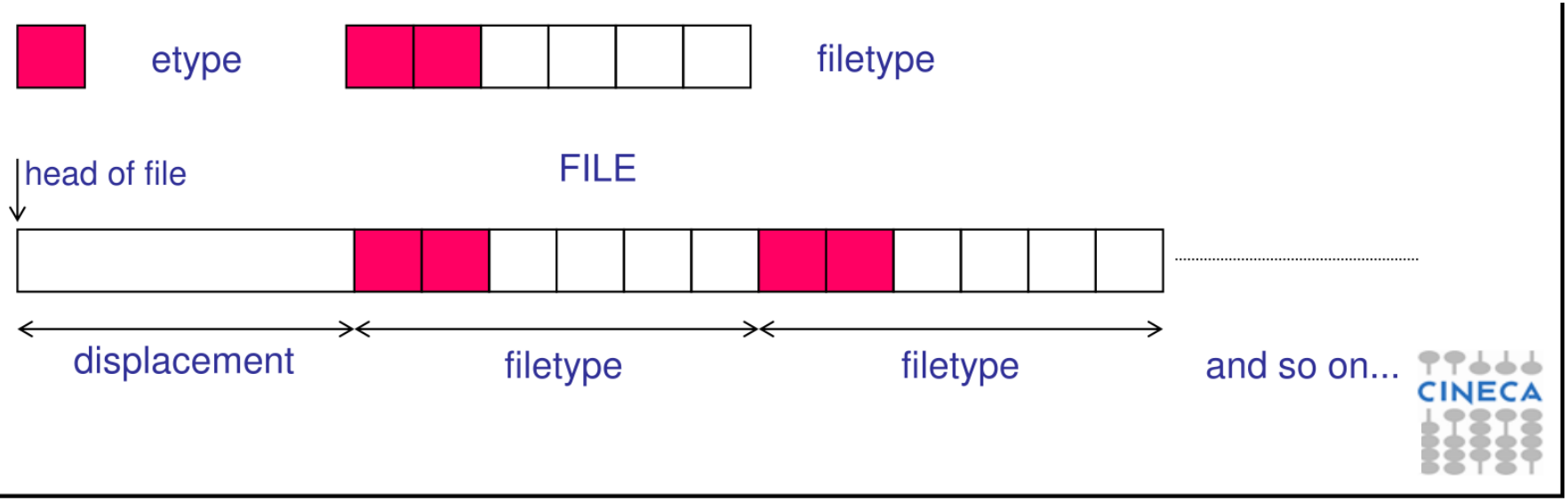
FFT pencil parallelization

- P3DFFT (Pekurovsky 2012)
- Transpositions: MPI Alltoall
- Local FFTW or, recently, cuFFT (now available 3D parallel)



Parallel I/O

- Collective read/write with derived data-types (domain slices)
- avoid write/read_at, non-shared headers (will lock the file)
- MPI_File_write_all(...)



Particle phase: Non-Uniform FFT

- Connect Eulerian fields and point-like particles

$$\boldsymbol{C}(\boldsymbol{x},t) = \sum_p c_p(t)\delta(\boldsymbol{x}-\boldsymbol{x}_p)$$

- Trick: take convolution in physical space..
- ..and then remove it in Fourier

$$\hat{\boldsymbol{C}}(\boldsymbol{x},t) \simeq \frac{\widehat{\boldsymbol{C}(\boldsymbol{x},t) * B(\boldsymbol{x})}}{\hat{B}(\boldsymbol{k})}$$

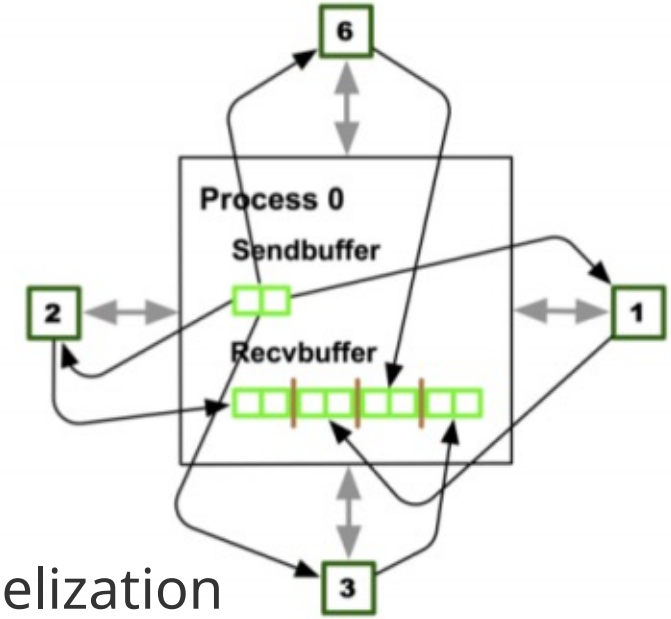
Particle phase: Non-Uniform FFT

- Connect Eulerian fields and point-like particles

$$\boldsymbol{C}(\boldsymbol{x},t) = \sum_p c_p(t)\delta(\boldsymbol{x}-\boldsymbol{x}_p)$$

- Trick: take convolution in physical space..
- ..and then remove it in Fourier

$$\hat{\boldsymbol{C}}(\boldsymbol{x},t) \simeq \frac{\widehat{\boldsymbol{C}(\boldsymbol{x},t) * B(\boldsymbol{x})}}{\hat{B}(\boldsymbol{k})}$$

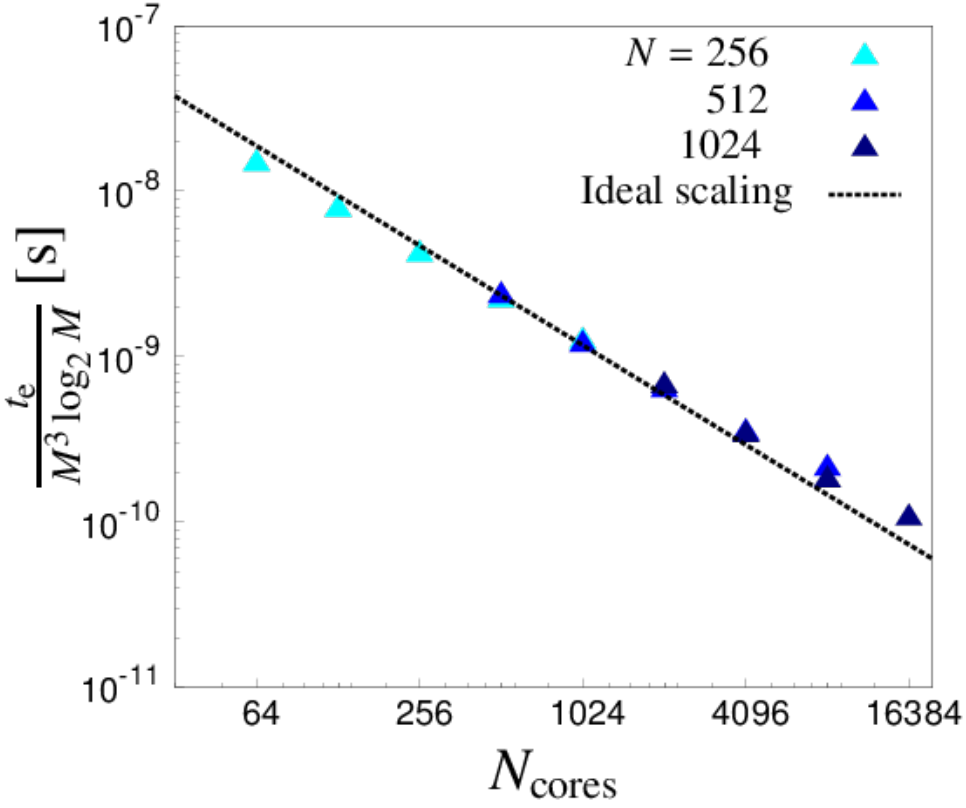


- MPI Neighbor collectives for particle parallelization

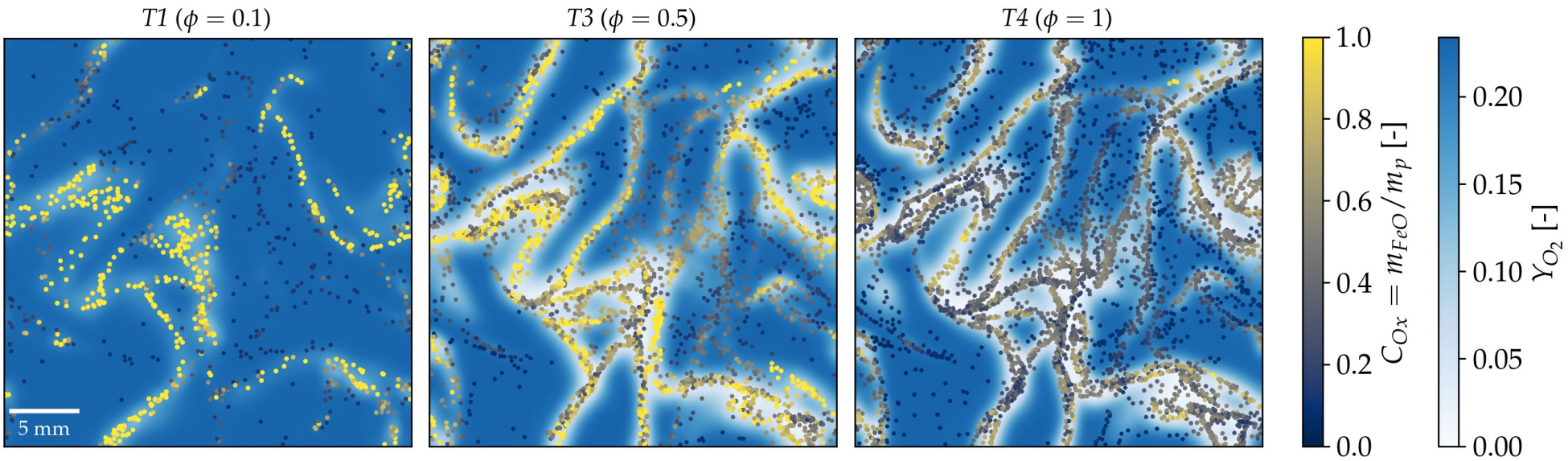
DNS code in a nutshell

- Built-in P3DFFT
- Basdevant trick
- MPI parallel IO
- No external libraries
- NUFFT for field interpolation
- NUFFT for particles feedback
- MPI Neighbor collectives
- Weakly compressible module for particle combustion

Scaling on Marconi, 2017



Extension to weakly compressible



- Immersed boundaries, particles, and much more
- Iron particle combustion
- Generates large heat and mass fluxes
- Multiscale interaction with turbulence

2D passive scalar turbulence model

Hyperviscosity degree 8

$$\partial_t u_i + \nabla_j \left(u_j u_i\right) = -\nabla_i P + \nu(\zeta)\nabla^{2\zeta} u_i - u_i/\mathcal{T}_u + f_i$$

Large-scale damping

Gaussian forcing

$$\partial_t \theta + \nabla_j \left(u_j \theta\right) = \kappa \nabla^2 \theta - \theta/\mathcal{T} + RG\left(\boldsymbol{x} - \boldsymbol{x}_s\right)$$

Large-scale damping

Smoothed point-like source

2D passive scalar turbulence model

Hyperviscosity degree 8

$$\partial_t u_i + \nabla_j \left(u_j u_i\right) = -\nabla_i P + \nu(\zeta)\nabla^{2\zeta} u_i - u_i/\mathcal{T}_u + f_i$$

Large-scale damping

Gaussian forcing

$$\partial_t \theta + \nabla_j \left(u_j \theta\right) = \kappa \nabla^2 \theta - \theta/\mathcal{T} + RG\left(\boldsymbol{x} - \boldsymbol{x}_s\right)$$

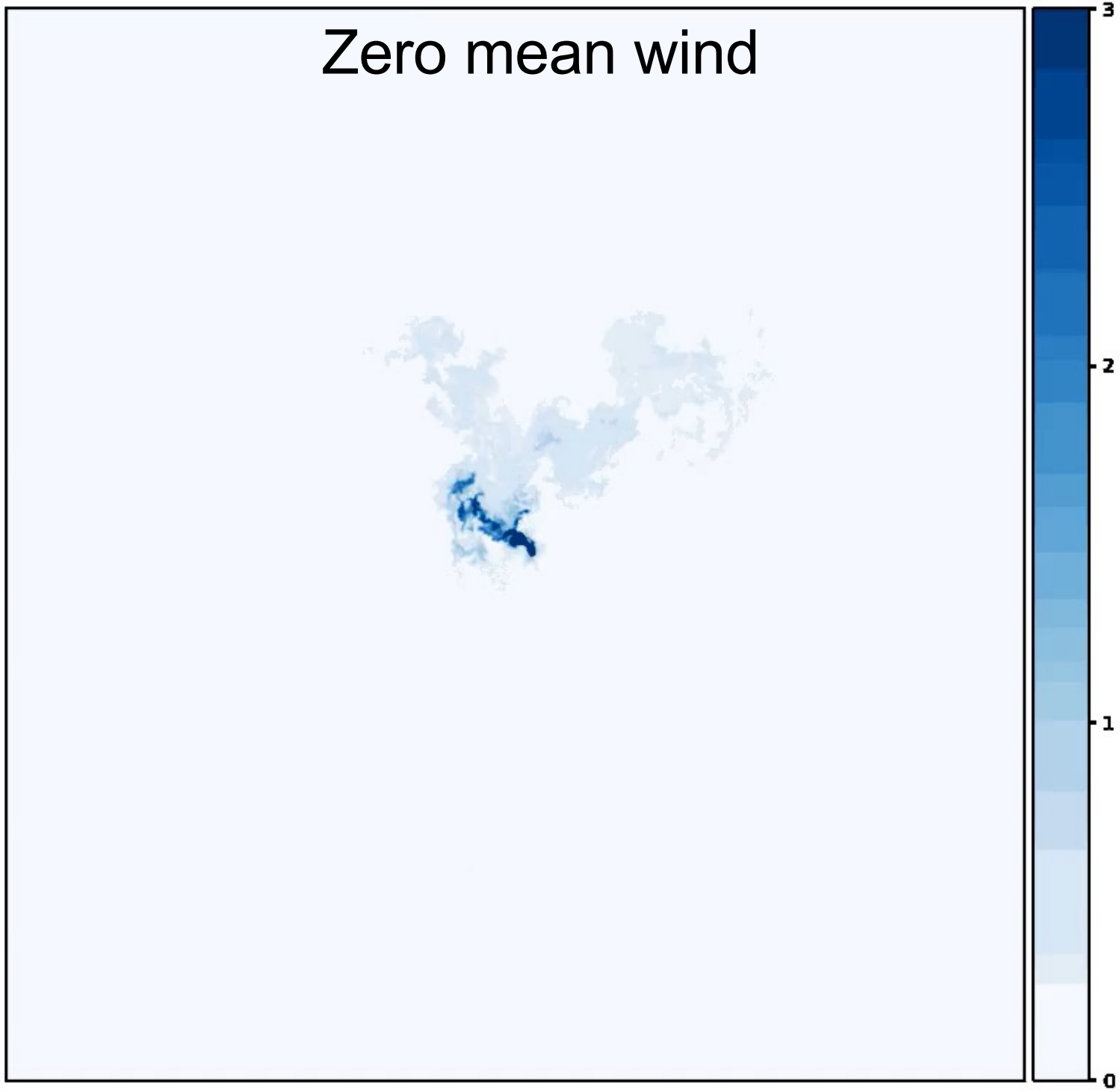
Large-scale damping

Smoothed point-like source

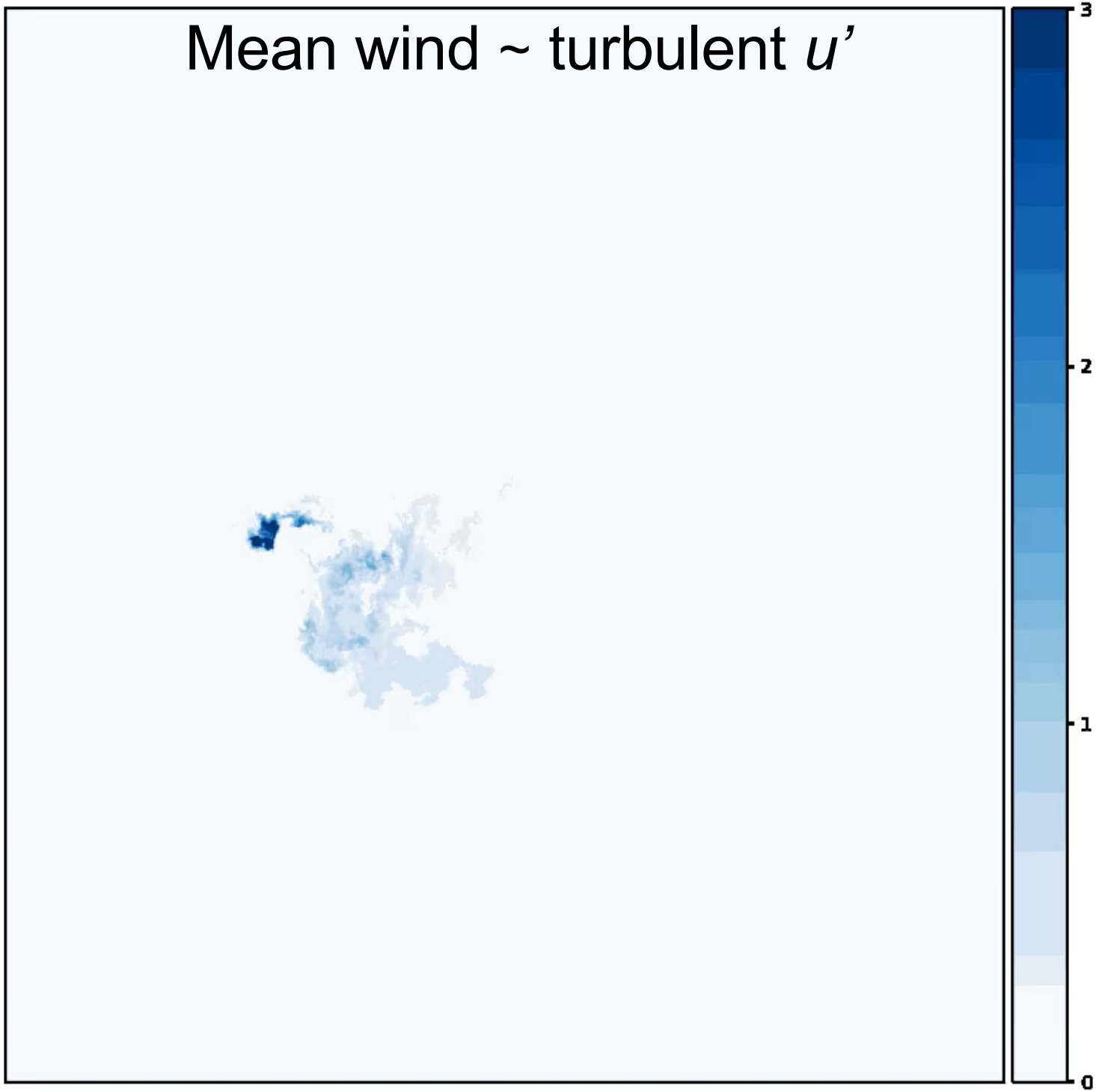
$$G(\boldsymbol{x}) = \frac{1}{2\pi\sigma^2} \exp\left(-\|\boldsymbol{x}\|^2/(2\sigma^2)\right)$$

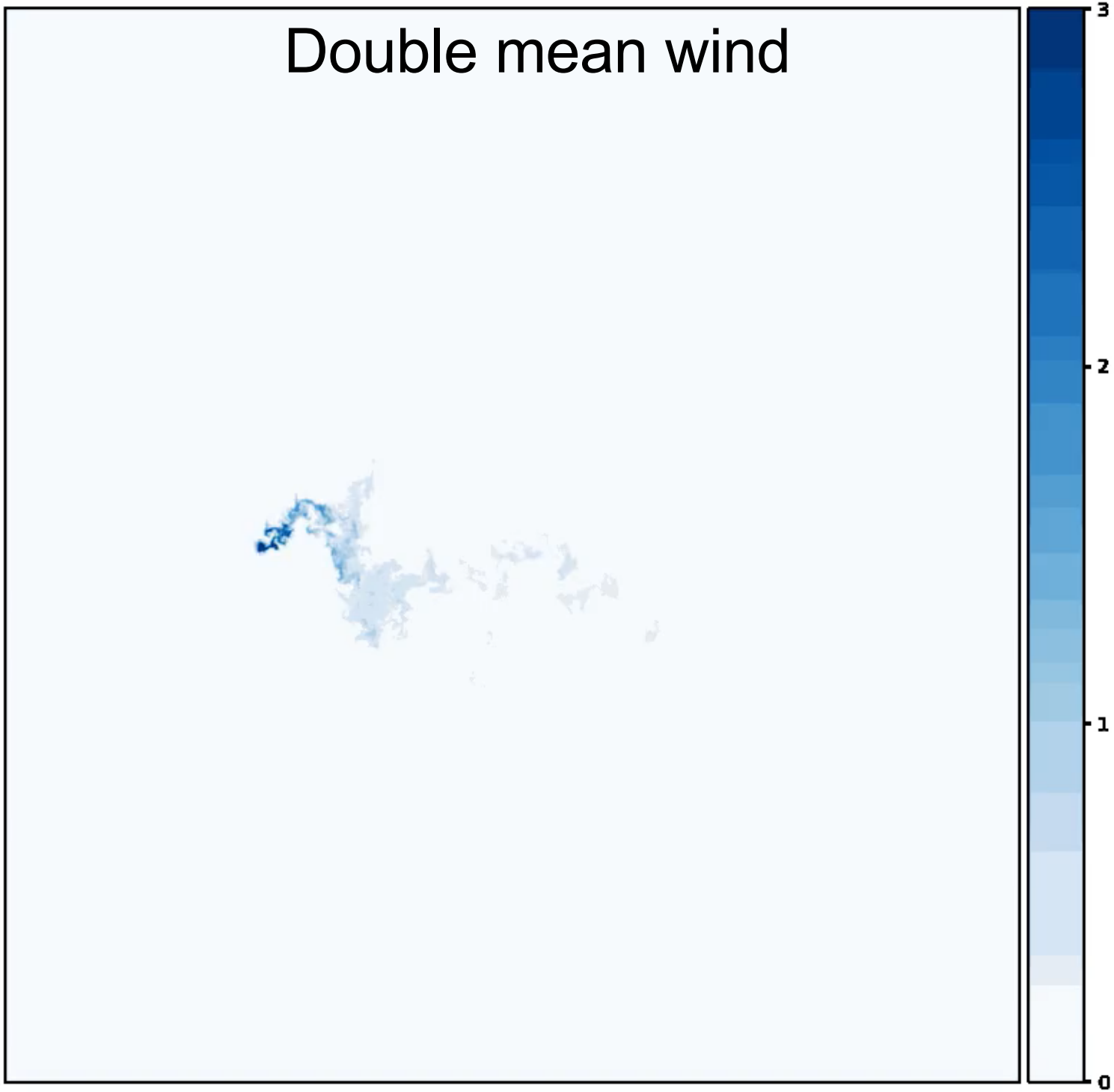
$$\left\langle f_i(\boldsymbol{x}, t) f_j(\boldsymbol{x}', t')\right\rangle = \delta_{ij}\delta\left(t - t'\right)\mathcal{C}(\boldsymbol{x} - \boldsymbol{x}')$$

Gaussian forcing at small scales: **inverse cascade**
Gaussian-like correlation function *C*

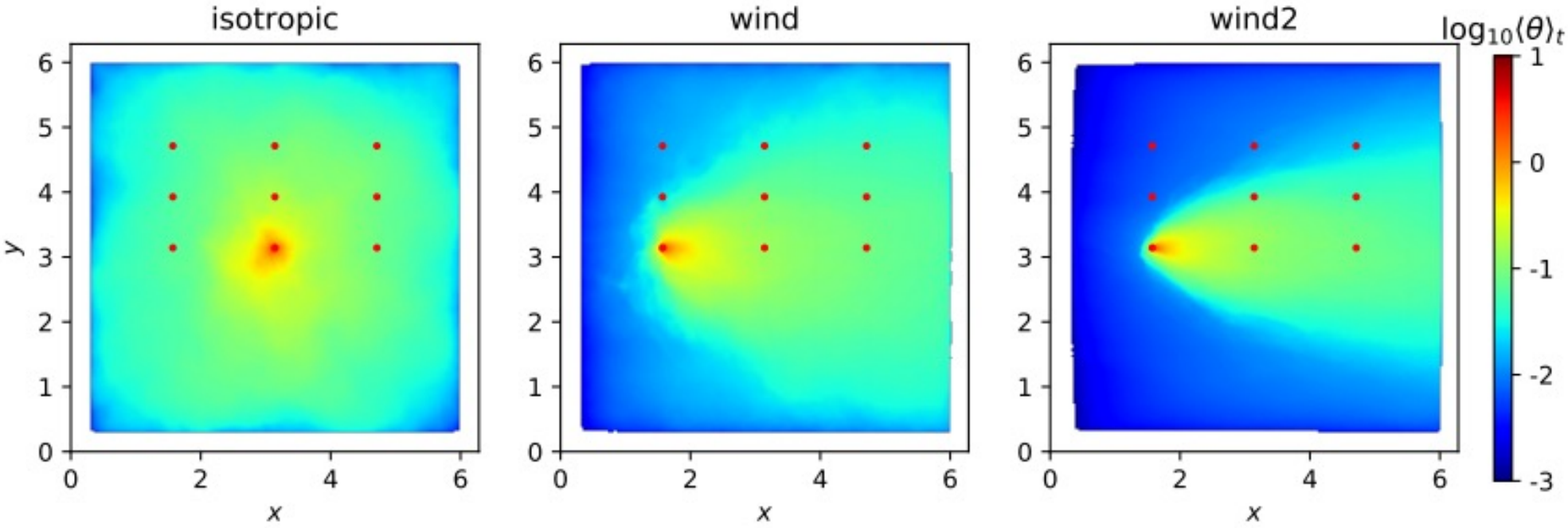

Zero mean wind
3
2
1
0

Mean wind ~ turbulent *u'*




Double mean wind
3
2
1
0

2D passive scalar: average concentration

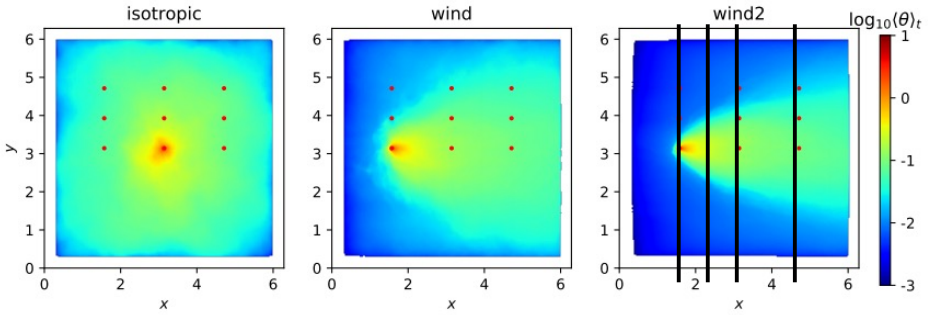


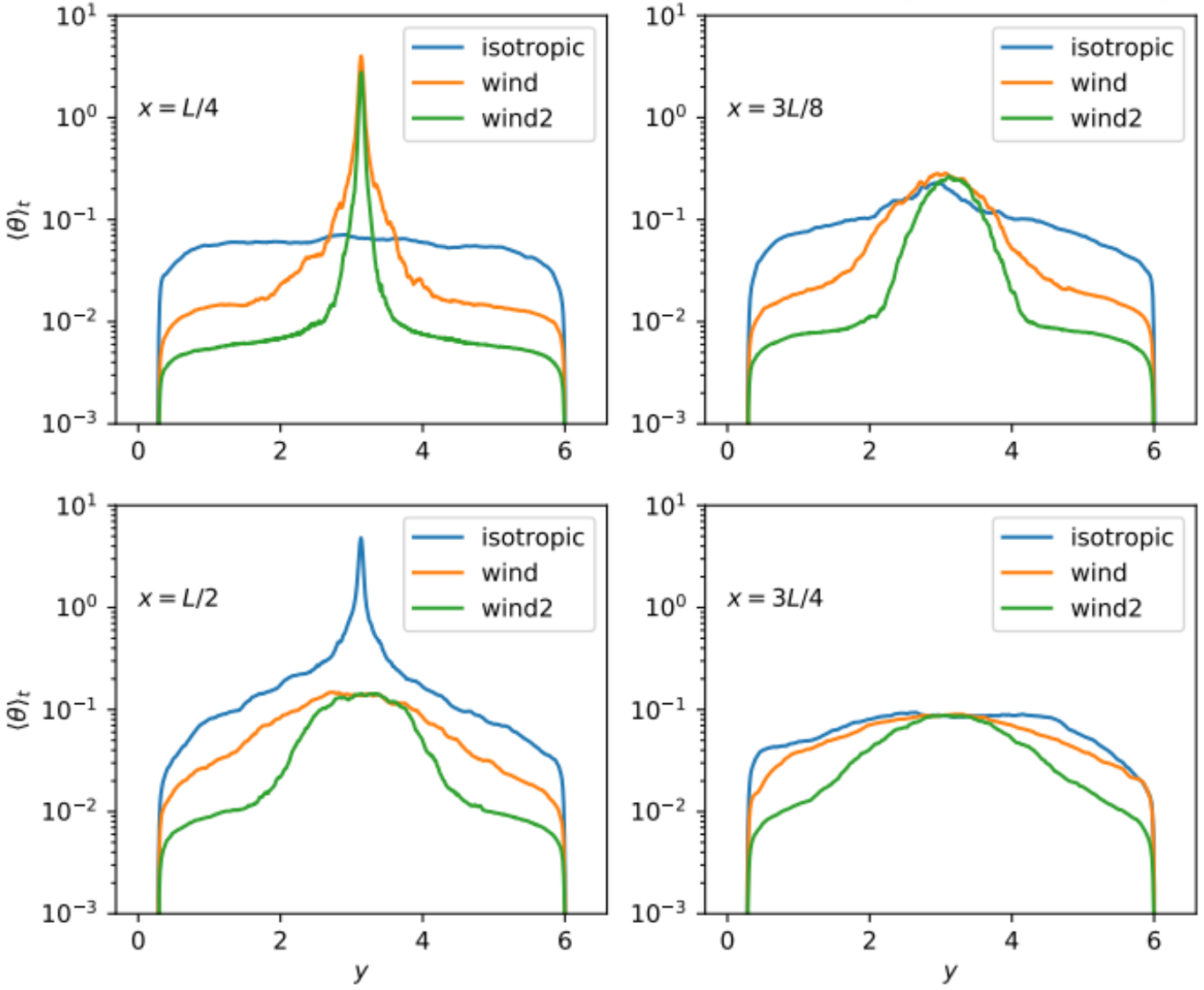
- Mask periodicity effects on ~*10%* domain size

$$\theta(\boldsymbol{x},t) = \prod_i \theta'(\boldsymbol{x},t)\tanh(S\boldsymbol{e}_i \cdot (\boldsymbol{x} - \boldsymbol{x}_{b,i}))$$

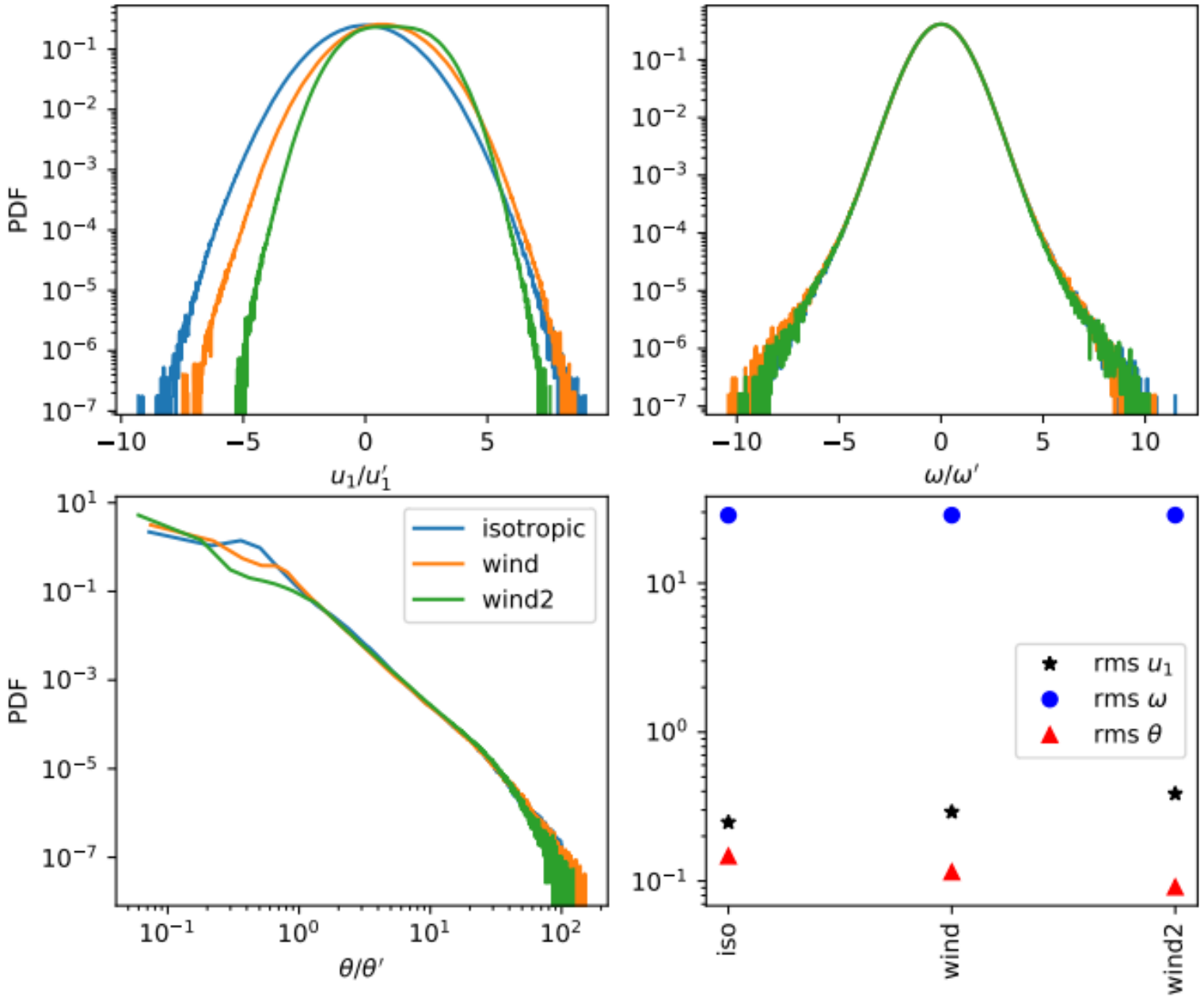
- Average scalar concentration: ~ 5 to <0.1 (decrease damping time)
- Resolved time series at the red points

2D passive scalar: cuts

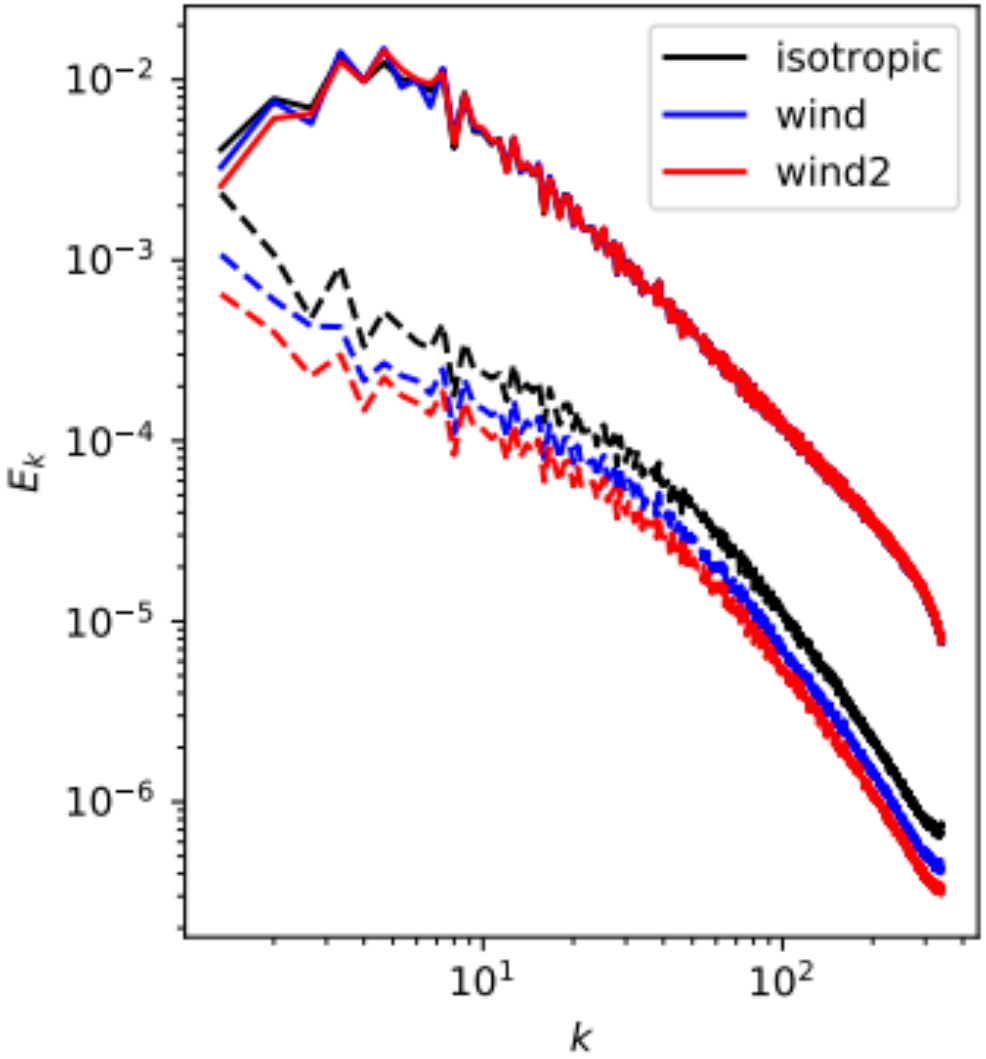


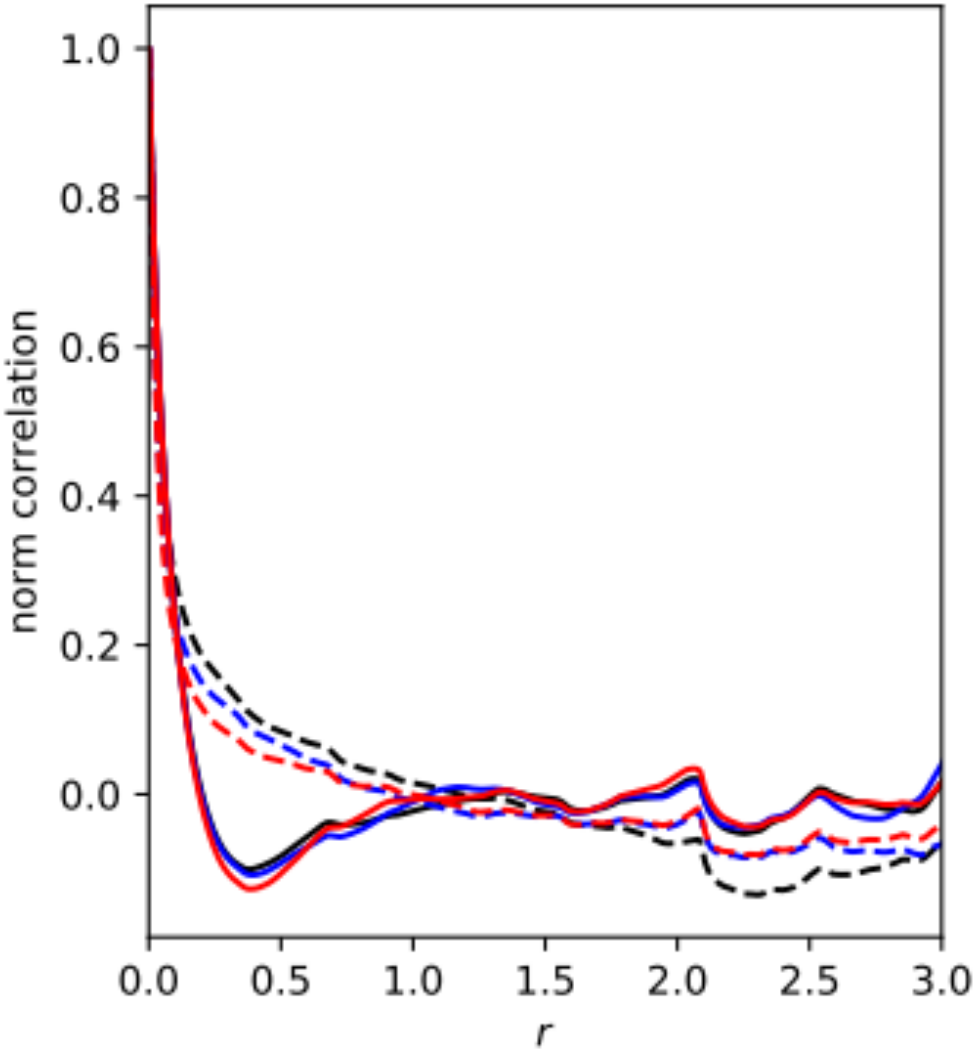


2D passive scalar: single-time statistics



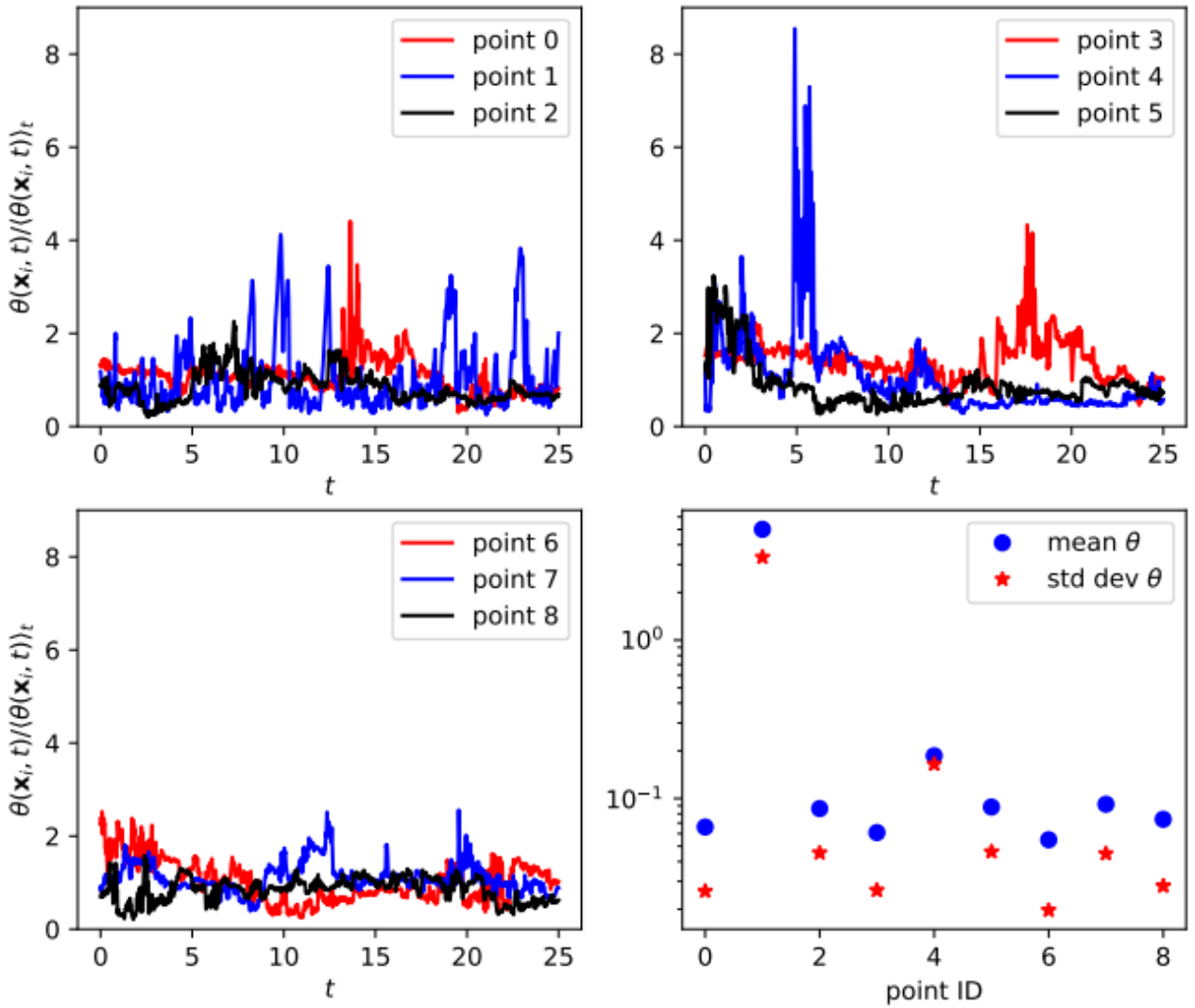
2D passive scalar: correlations

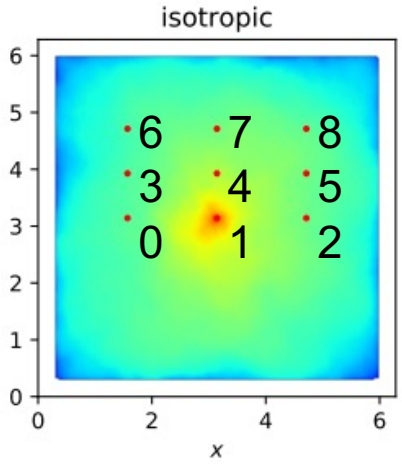




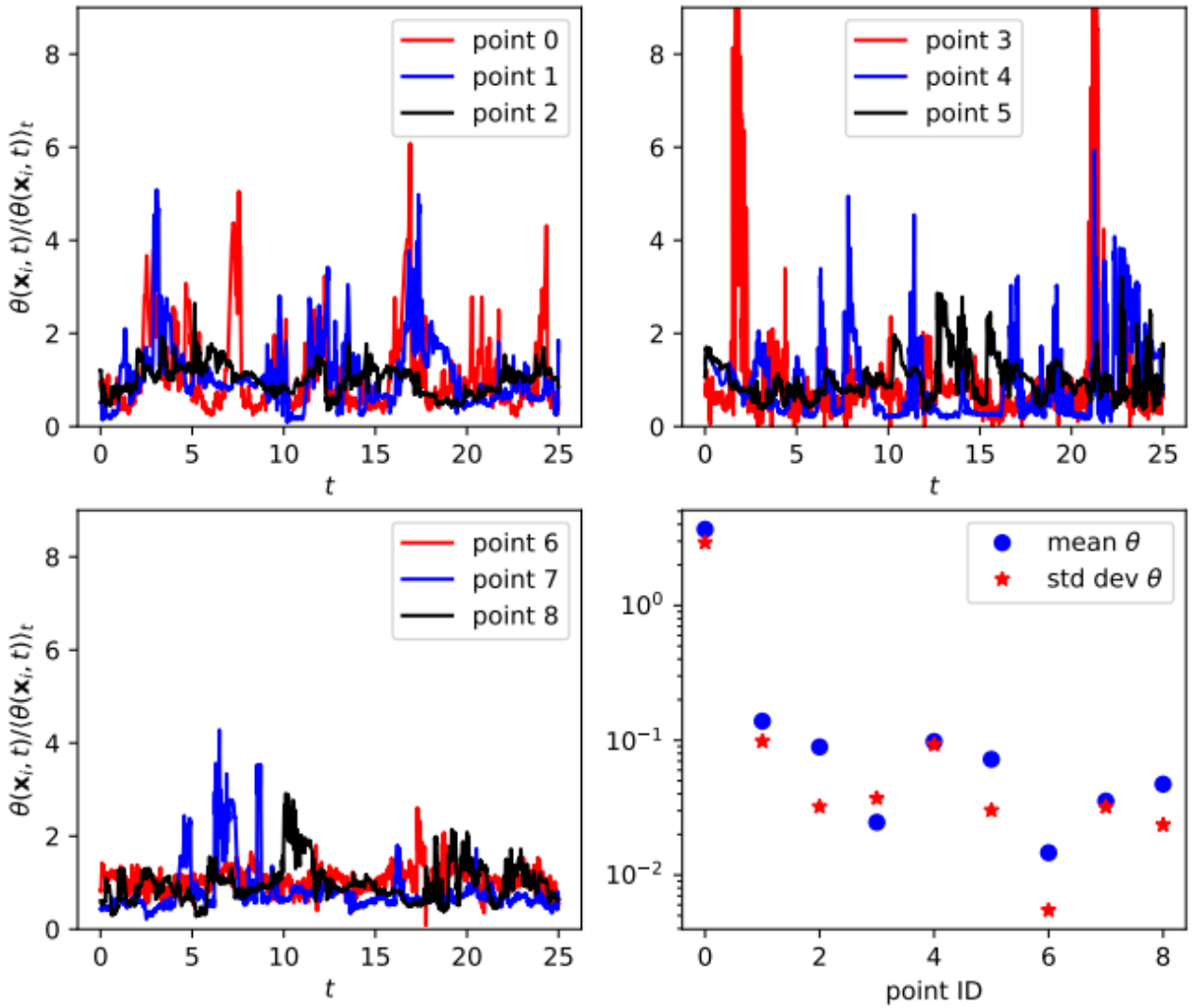
- Hyperviscosity and mask effects

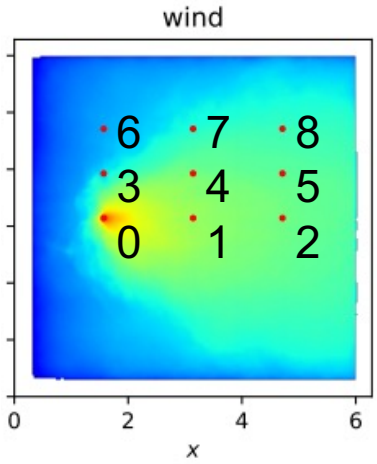
Scalar time series zero mean wind



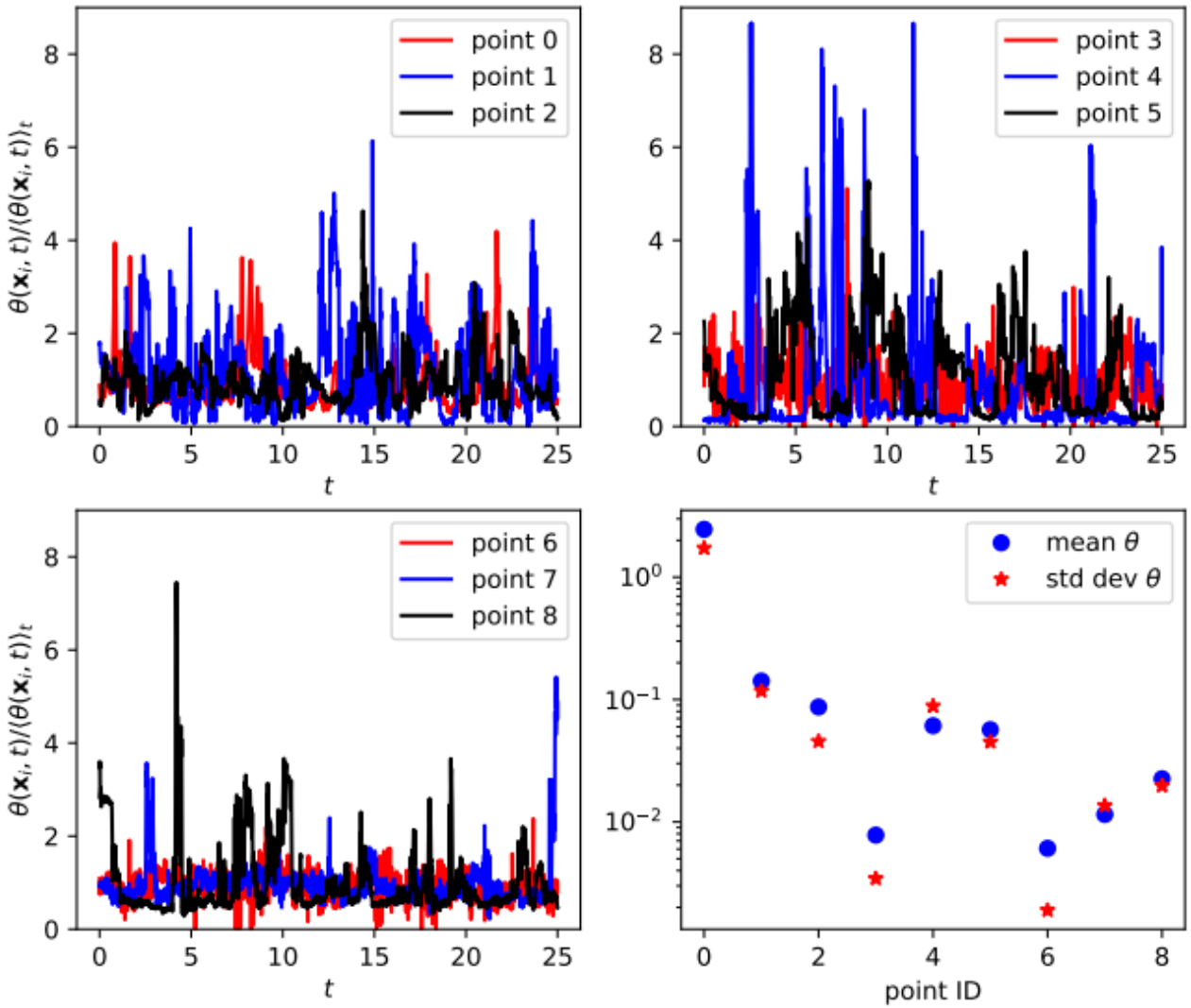


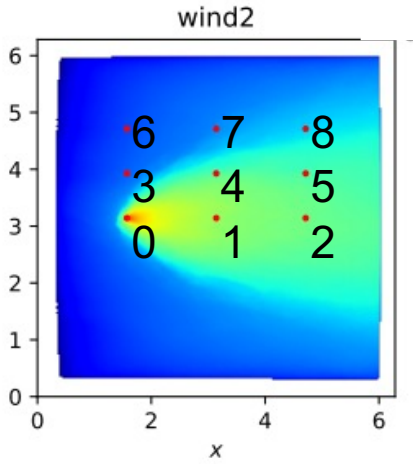
Scalar time series with mean wind





Scalar time series double mean wind





What’s next