Science and Technology Facilities Council

Stochastic modelling of particles in turbulent flows with neural networks

Josh Williams[1,2]

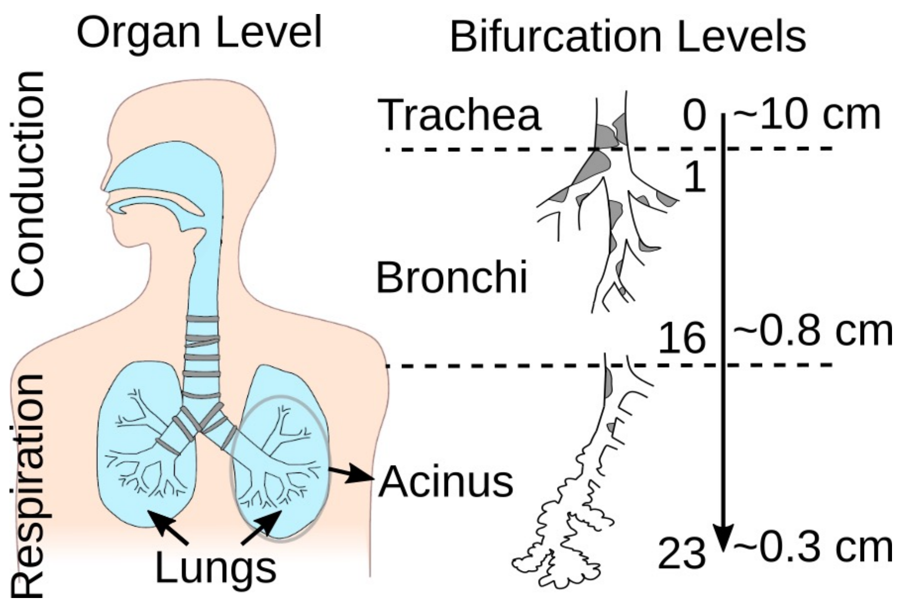
[1] Hartree Centre, STFC UKRI, UK
[2] School of Engineering and Physical Sciences, Heriot-Watt University, UK

Overview

- Motivation and context
- Fluid and particle modelling
- Data generation, neural network training, implementing in OpenFOAM.
- Results
- Conclusions and lessons learned

Motivation and context

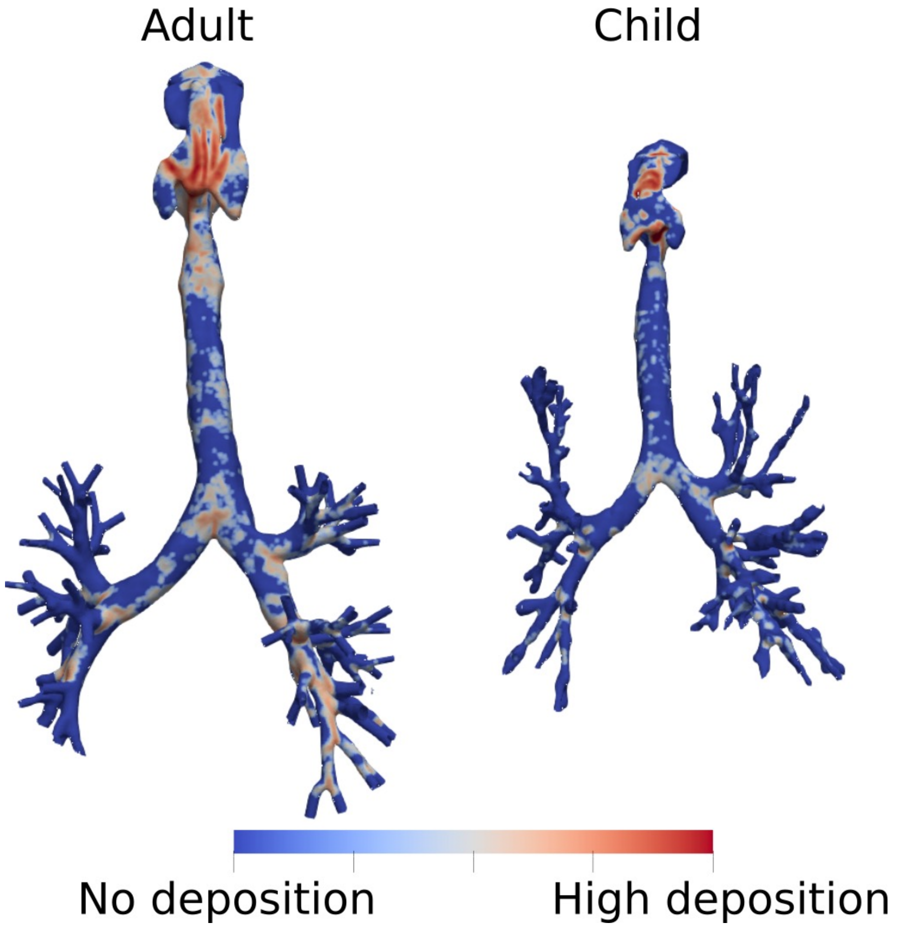
- 1 billion patients worldwide suffer from chronic respiratory disease [1].



[1] The Global Asthma Network. *The Global Asthma Report*, 2018. [2] Williams, J. et al. 2022. *International Journal of Pharmaceutics*.

Motivation and context

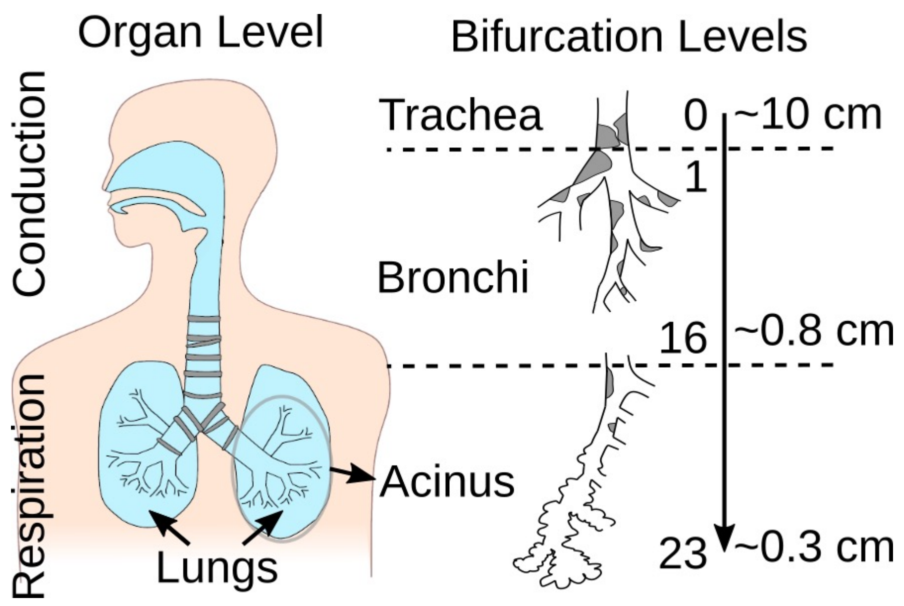
- 1 billion patients worldwide suffer from chronic respiratory disease [1].
- Treatment efficacy varies across patients [2].
  - Lung and airway shape.
  - Disease in different part of the lungs.
  - Patients breathe differently.

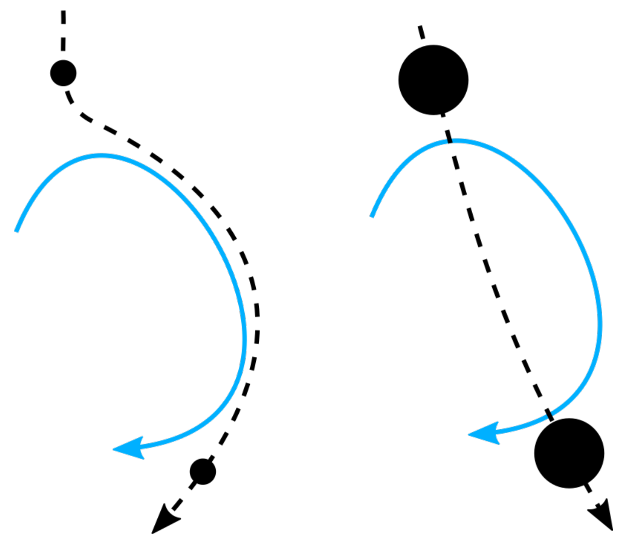


[1] The Global Asthma Network. *The Global Asthma Report*, 2018. [2] Williams, J. et al. 2022. *International Journal of Pharmaceutics*.

Motivation and context

- 1 billion patients worldwide suffer from chronic respiratory disease [1].
- Treatment efficacy varies across patients [2].
  - Lung and airway shape.
  - Disease in different part of the lungs.
  - Patients breathe differently.
- Deposition depends on particle size.

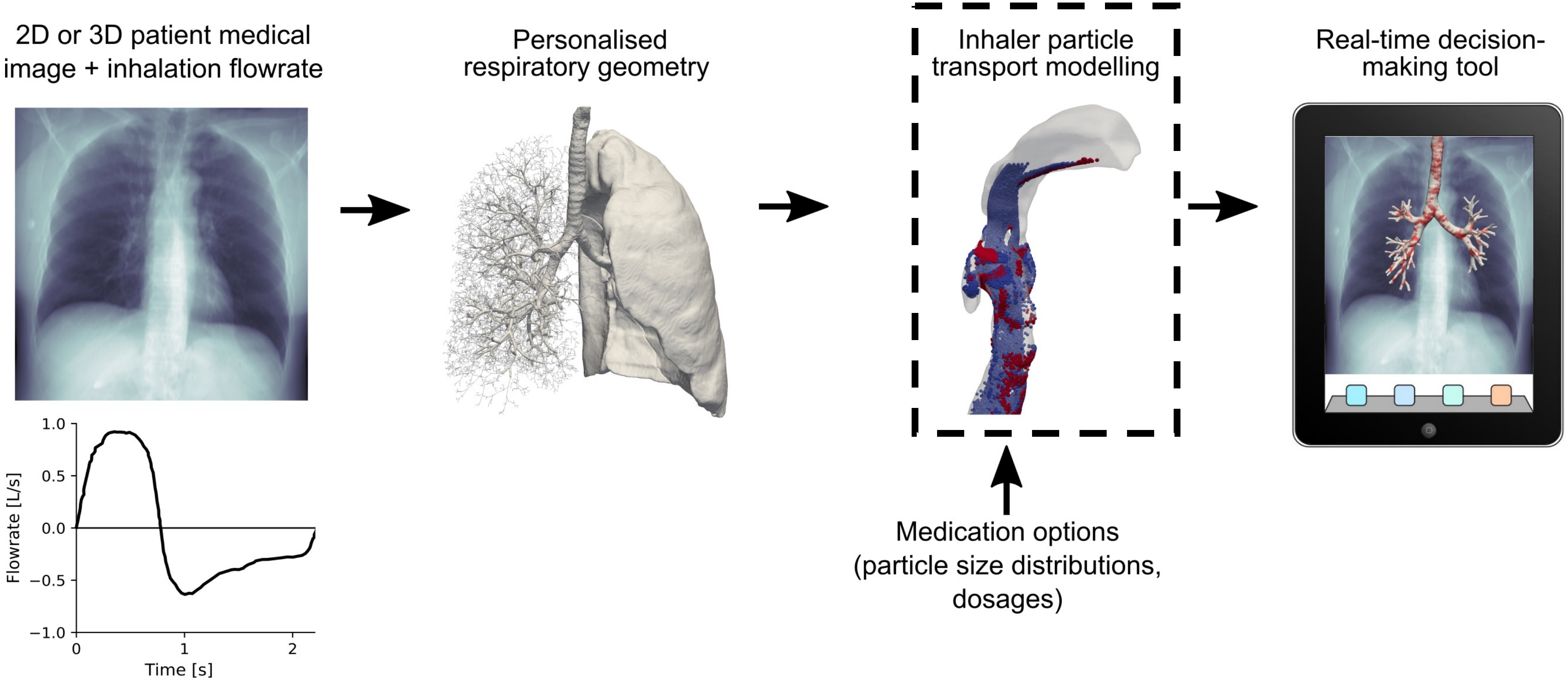




[1] The Global Asthma Network. *The Global Asthma Report*, 2018. [2] Williams, J. et al. 2022. *International Journal of Pharmaceutics*.

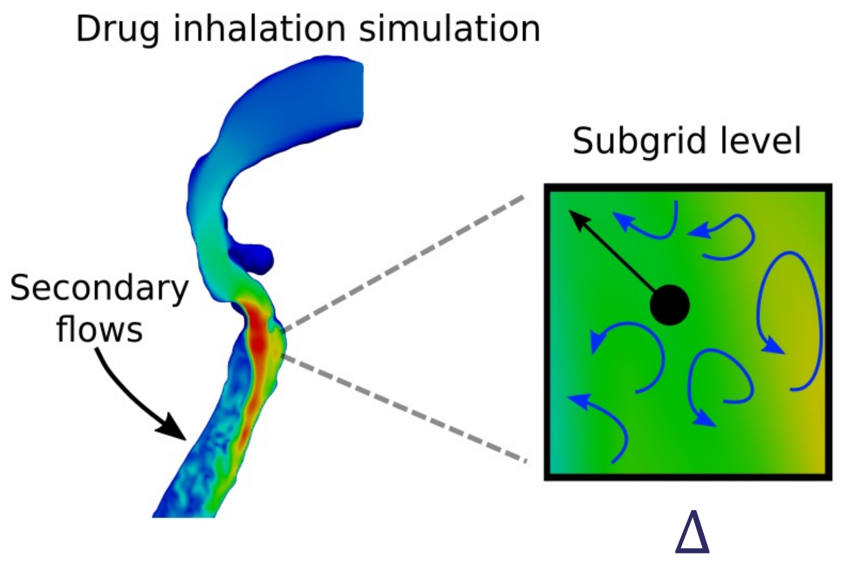
Image-based modelling as a solution

- **Main motivation:** use modelling to predict and optimise drug deposition.
- Large range of length and time-scales (**turbulence**, particle interactions etc).



Fluid modelling framework

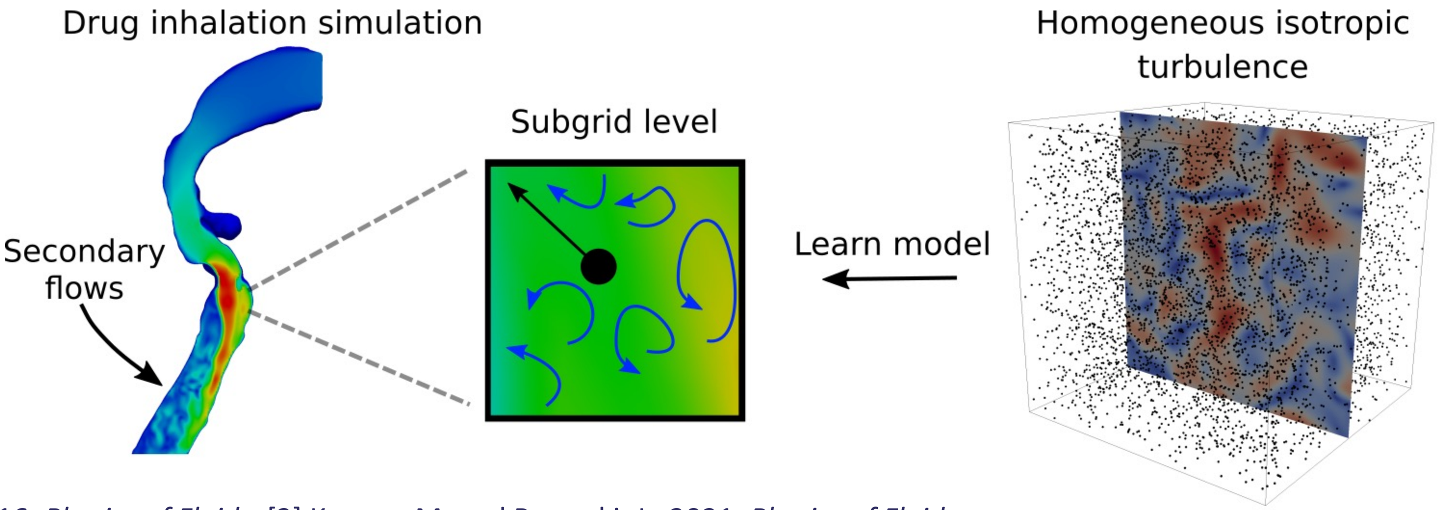
- Cannot resolve all length scales. $\boldsymbol{u}_f = \tilde{\boldsymbol{u}}_f + \boldsymbol{u}_{sgs}$.
- *Filtered* incompressible Navier-Stokes (large eddy simulation, LES).
- Eddy viscosity model for τ_{sgs}. [1]
- k_{sgs} and ε_{sgs} based on mesh size and local velocity gradients.
- Use MPPICFoam with 1-way coupling. [2]



[1] Germano, M. et al., 1991. *Physics of Fluids.*
[2] Kleinstreuer, C. and Zhang, Z., 2010. *Annual Review of Fluid Mechanics.*

Modelling turbulent fluctuations

- Turbulent fluctuations ($\boldsymbol{u}_{sgs}$) may influence deposition [1].
- Existing stochastic models [1] not consistent with near-wall or decaying flow [2].
- **Our aim: learn stochastic turbulence model from highly resolved simulations.**
- Current model limited to homogeneous flows (*proof of concept*).



[1] Innocenti, A. et al., 2016. *Physics of Fluids*. [2] Knorps, M. and Pozorski, J., 2021. *Physics of Fluids*.

Stochastic particle modelling

- Newton's equation of motion:

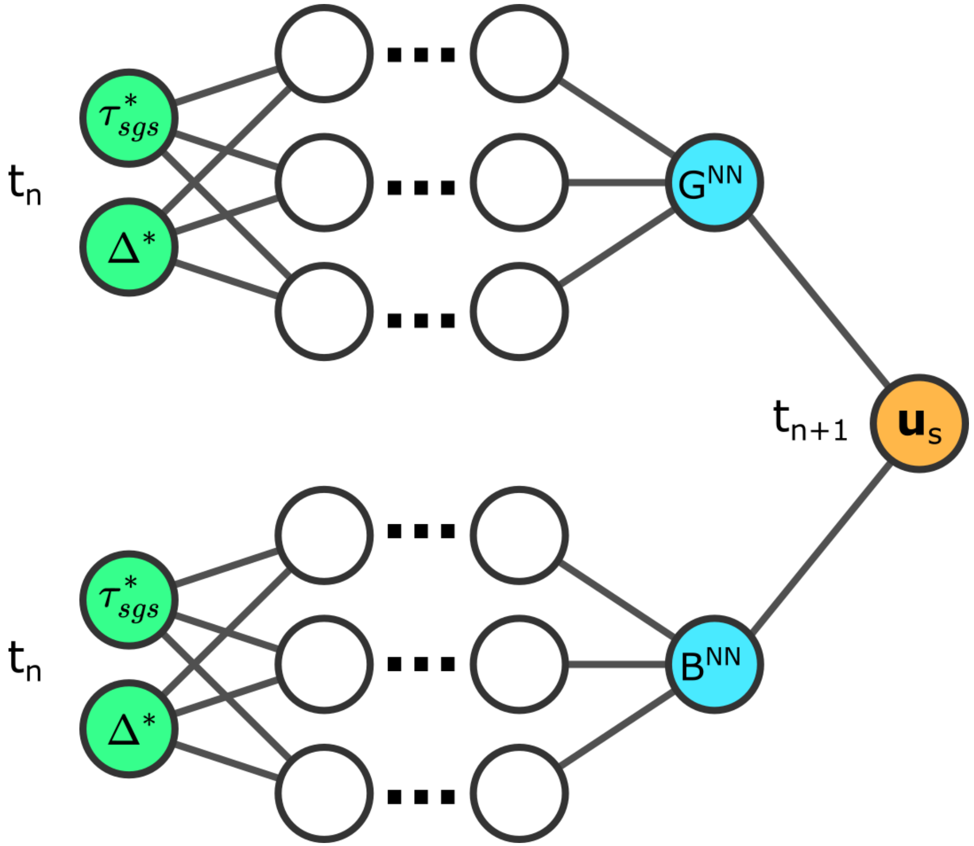
$$d\boldsymbol{u}_p = \frac{\boldsymbol{u}_s - \boldsymbol{u}_p}{\tau_p} dt \qquad \text{and} \qquad \tau_p = \frac{\rho_p}{\rho_f} \frac{4\, d_p}{3 C_D \left| \boldsymbol{u}_s - \boldsymbol{u}_p \right|}$$

- We can use a Langevin-type model for $\boldsymbol{u}_s$[1]:

$$d\boldsymbol{u}_s = -G^{NN}\left(\boldsymbol{u}_s - \widetilde{\boldsymbol{u}}_f\right) dt + B^{NN} d\boldsymbol{w}$$

$$G^{NN} = 1/T^{NN}$$

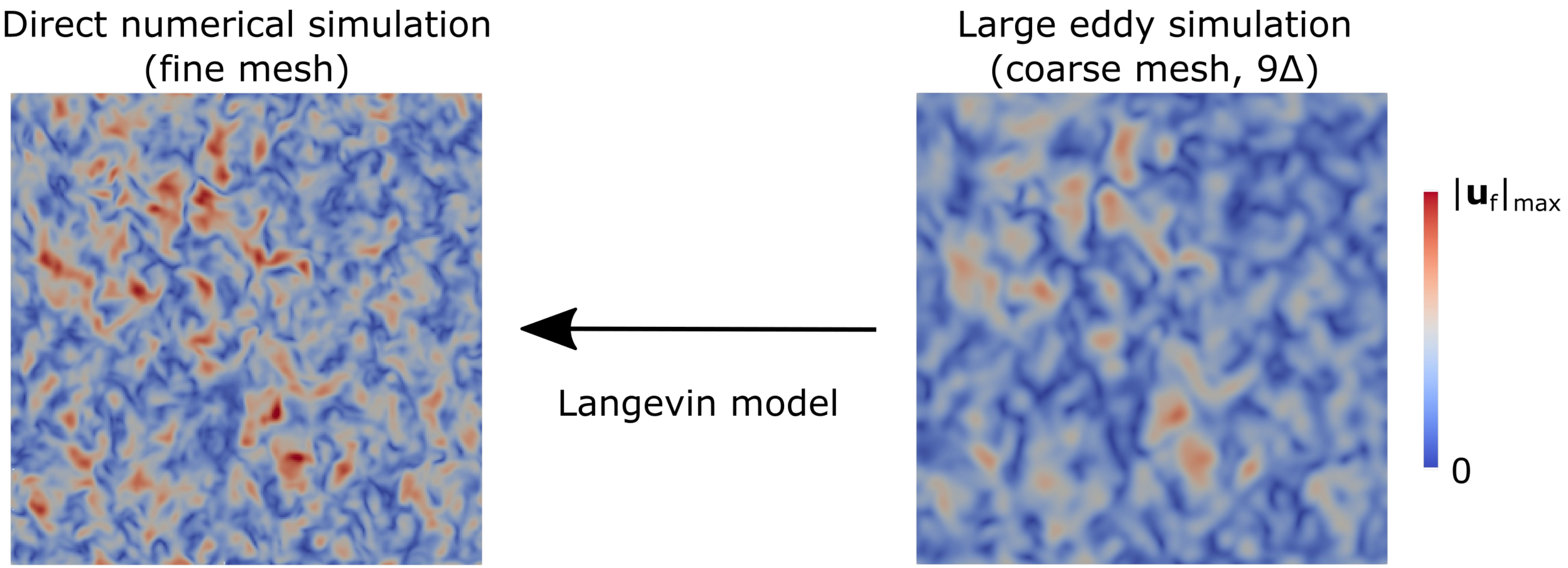
All fluid properties interpolated to particle.



[1] Pozorski, J. and Apte, S., 2009. *International Journal of Multiphase Flow.* [2] Williams, J. et al., 2022. *Physics of Fluids*.

Simulation configuration

- Decaying homogeneous isotropic turbulence at $Re_\lambda^0 = \{9, 33, \mathbf{105}\}$.
- Ground truth has 256³ cells. $St^0 = 0.1$, $N_p = 10^4$.
- Top-hat filter width $\{3\Delta, 5\Delta, 7\Delta, 9\Delta\}$ [1]. Extract subgrid variables $k_{sgs}, \varepsilon_{sgs}$.



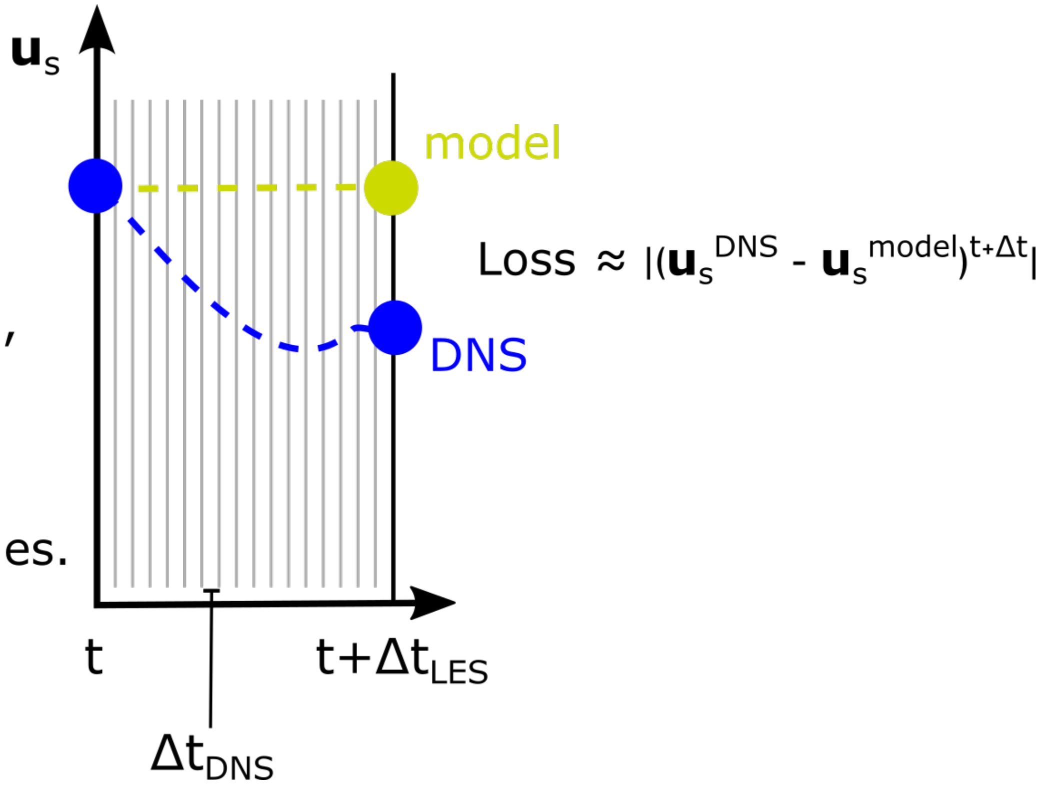
[1] Ozel, A. et al., 2016. *Chemical Engineering Science*.

Neural network training

- Feed-forward neural network (4 hidden layers, 120 neurons per layer).

Train on 2000 particle
trajectories per step,
40 steps ($t \rightarrow t+\Delta t_{LES}$),
2 Reynolds numbers,
4 filter widths

≈ 300k training samples.



Using in OpenFOAM

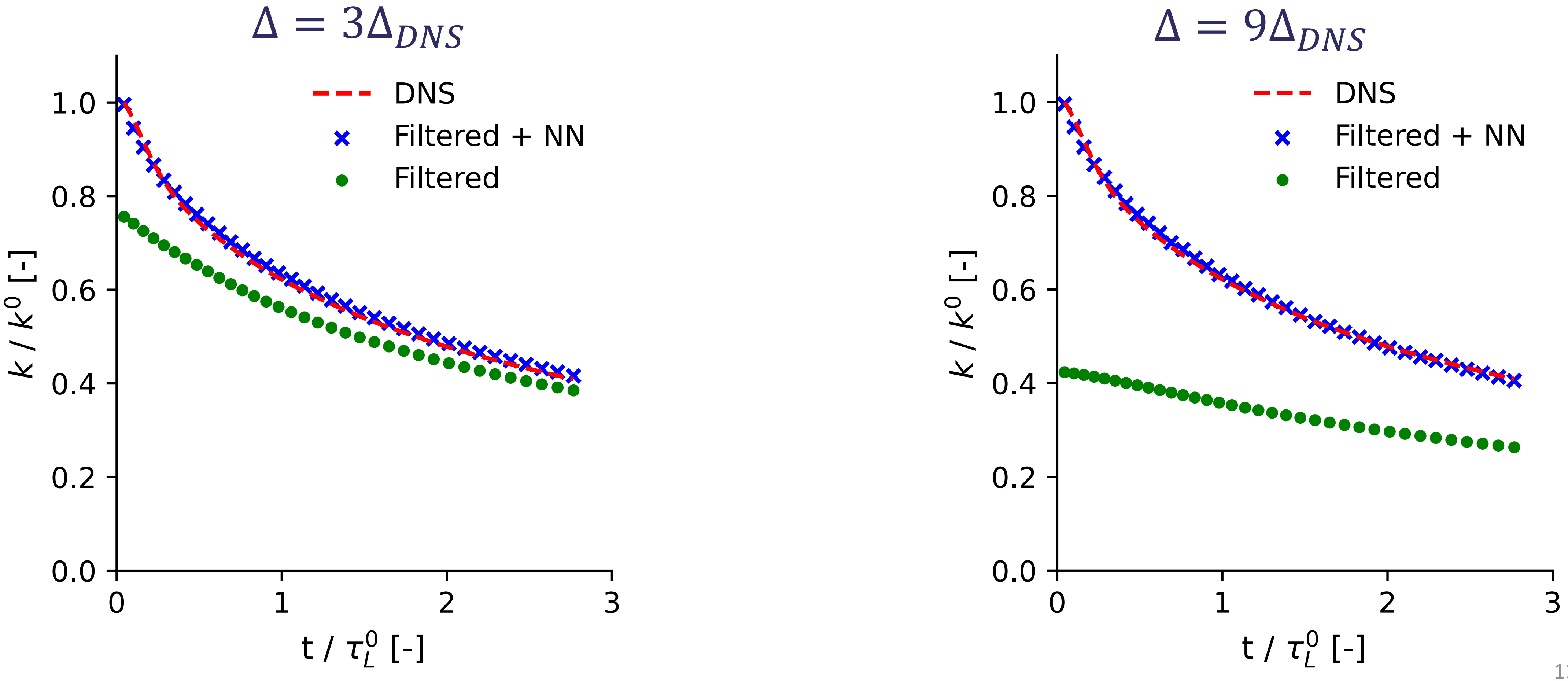
- Save Tensorflow architecture as JSON, parameters as HDF5. Write to readable ASCII format.
- Keras2cpp respository, read weights and perform forward pass.
  - Limited features, not been updated in around 6 years…
- Tried other repositories, but not really lightweight or user-friendly.

```
layers 12
layer 0 Convolution2D
4 1 3 3 same
[ 0.13240439 -0.39469701 -0.05886053]
[-0.16273086  0.27228925  0.70811206]
[-0.0382412   0.91429299  0.72160912]
[ 0.41376248 -0.13718128 -0.55548537]
[ 0.06595665 -0.43903521  0.04035483]
[ 0.34481722  0.63220054  0.07781094]
[ 0.41419399 -0.13306008  0.72000468]
[ 0.28247169  0.10319189  0.50113165]
[-0.14225948  0.33037826 -0.4037863 ]
[ 0.21228614 -0.69273335  0.45505175]
[ 0.63414061 -0.75804955 -0.62078679]
[-0.5811435  -0.20503305  0.65040439]
[ 0.04864343 -0.02648391 -0.00558628  0.06567492]
layer 1 Activation
relu
```

[1] https://github.com/pplonski/keras2cpp

Tracer velocity statistics (*a priori*)

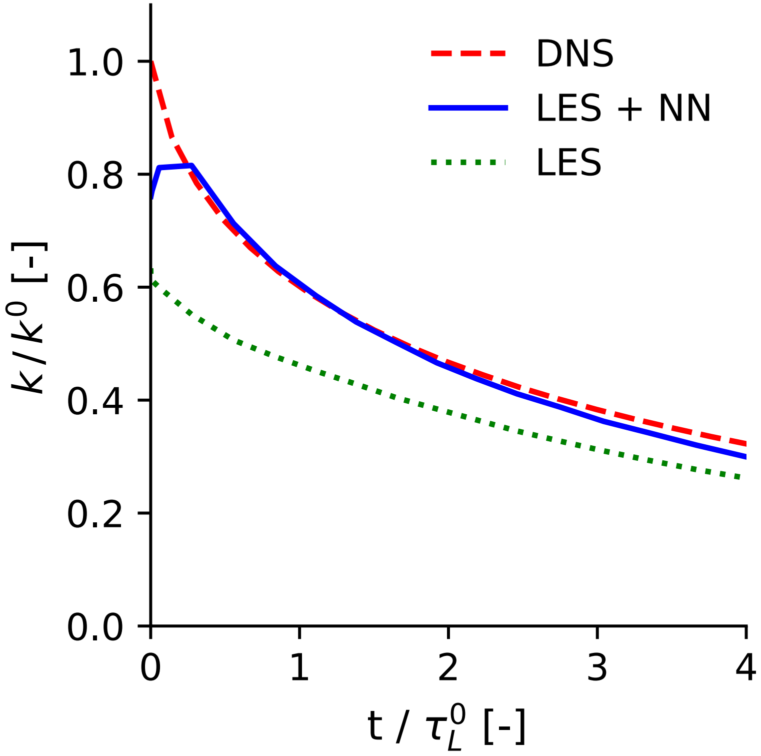
$k \ = tr(\boldsymbol{u}_s \otimes \boldsymbol{u}_s)/2,\ Re_\lambda^0 = 33$



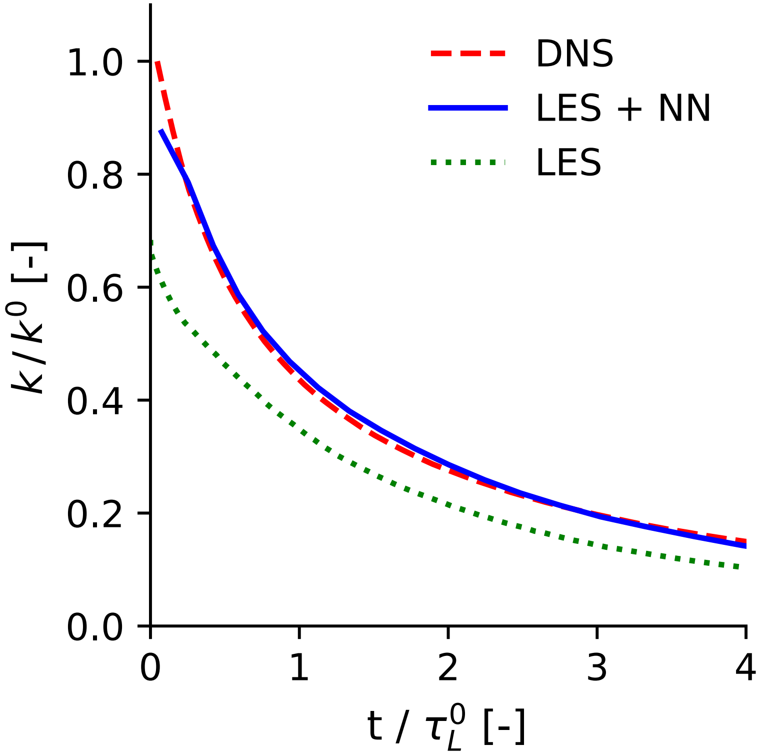
Tracer velocity statistics (*a posteriori*)

Initial condition $\boldsymbol{u}_p^0 = \boldsymbol{u}_f^0$, but in LES we only have filtered fields. So, $\widetilde{\boldsymbol{u}}_f^0 + \sqrt{2k_{sgs}/3}\xi$

$Re_\lambda^0 = 33, \Delta = 7\Delta_{DNS}$

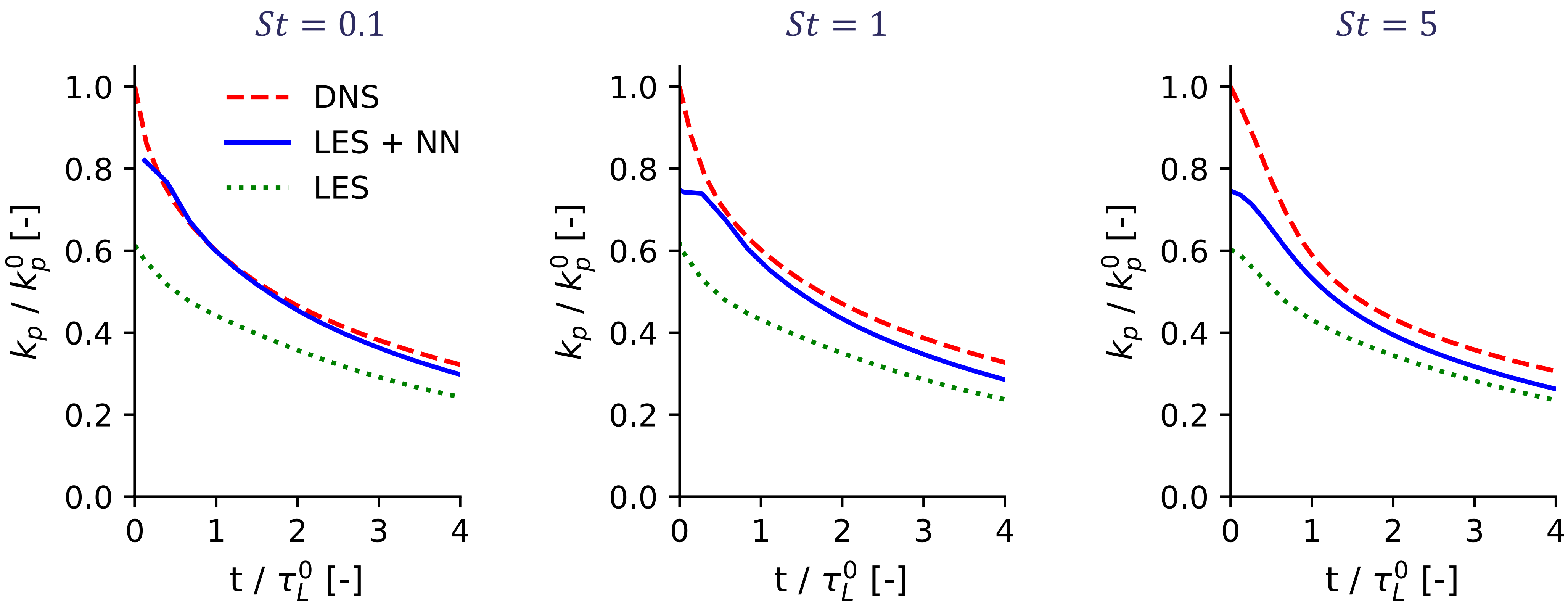


$\boldsymbol{Re_\lambda^0 = 105}, \Delta = 7\Delta_{DNS}$



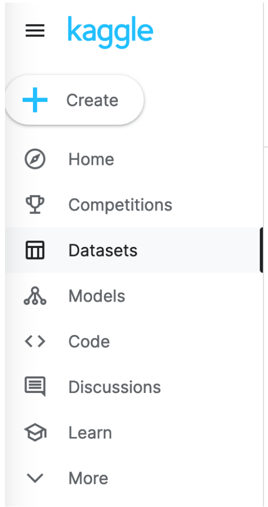
Inertial particle kinetic energy

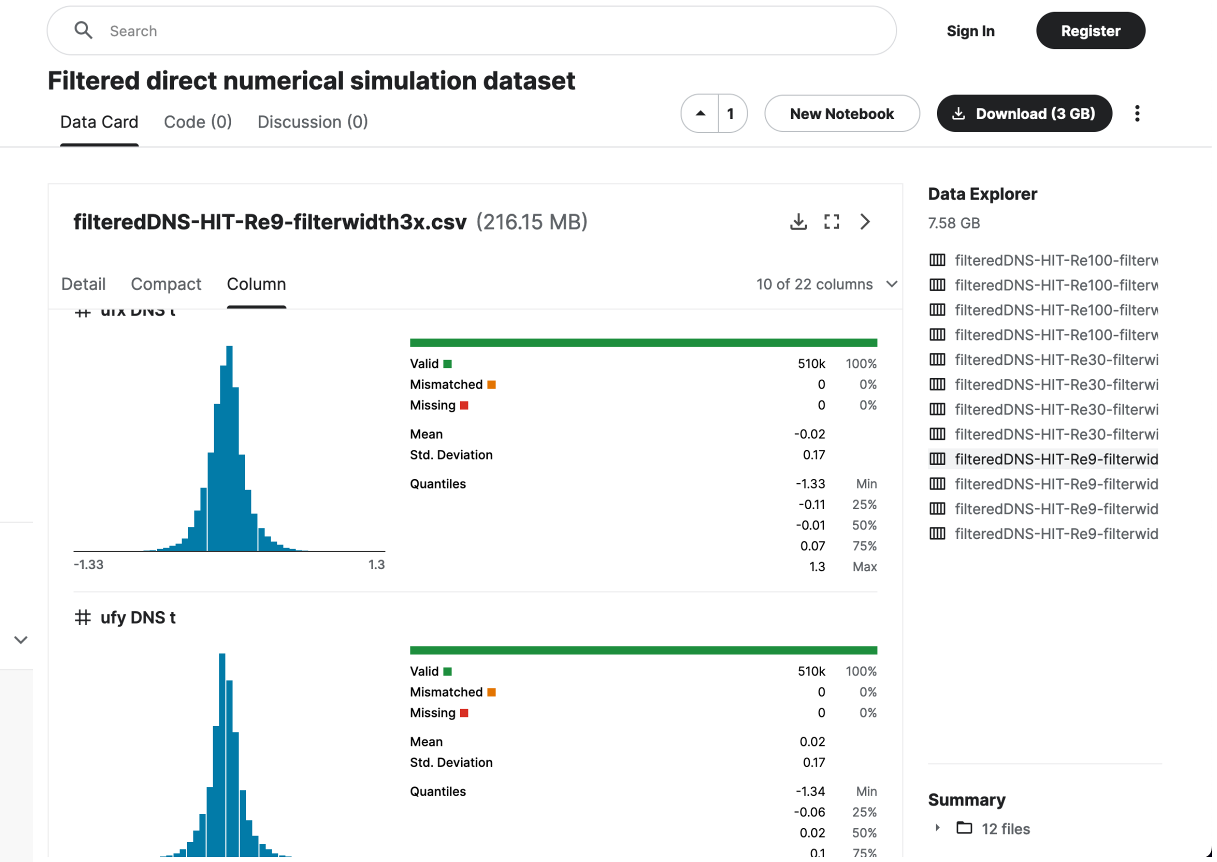
Parameters: $k_p = \frac{1}{2} tr(\boldsymbol{u}_p \otimes \boldsymbol{u}_p)^2$, $Re_\lambda^0 = 33$, $N_{cell} = 36^3$, $St = \tau_p/\tau_L^0$.

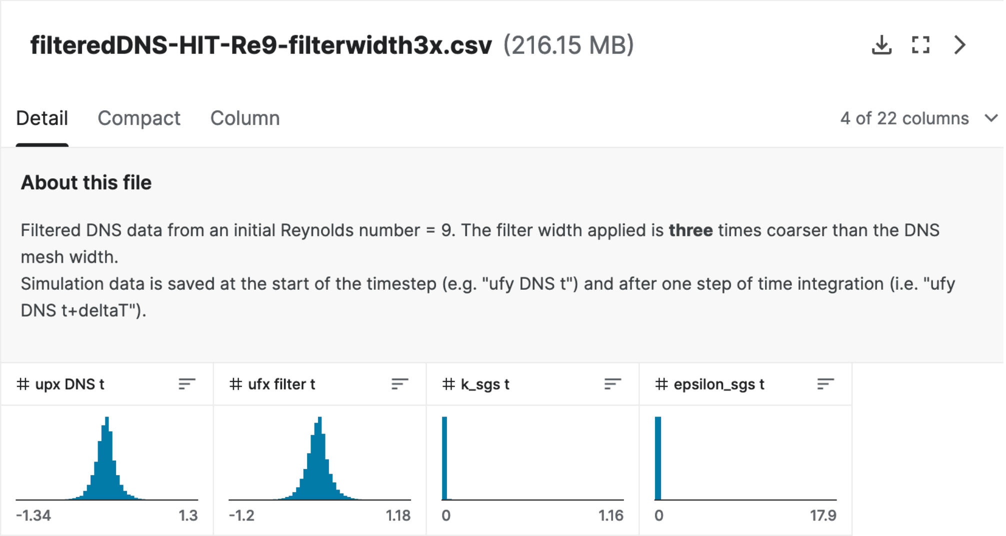


Open science

- Data shared on Kaggle.
- Python + OpenFOAM code on Github.
  - Integrated platform for code + data?







Conclusions

- Developed approach for data-driven modelling based on highly-resolved simulation data.
- NN model improves predictions of particle velocity in unbounded flows.
- Include complex physics (data-driven models for collisions, cohesion, etc).
- Code + pre-trained model shared on Github, data shared on Kaggle (seems rare).
- How to share best-practices? CFD has e.g. OpenFOAM workshop, not much in ML for fluid dynamics.
- **Take home message**: Data-driven modelling can provide more efficient and faster simulations (coarser mesh).

Acknowledgements

- Dr Uwe Wolfram and Dr Ali Ozel (HWU).
- Carnegie-Trust for the Universities of Scotland (PhD scholarship).
- Open Clouds for Research Environments (cloud computing credits).

Science and Technology Facilities Council

Thank you

Physics of Fluids ARTICLE scitation.org/journal/phf

Neural stochastic differential equations for particle dispersion in large-eddy simulations of homogeneous isotropic turbulence

Cite as: Phys. Fluids **34**, 113315 (2022); doi: 10.1063/5.0121344
Submitted: 17 August 2022 · Accepted: 13 October 2022 ·
Published Online: 4 November 2022

View Online Export Citation CrossMark

J. Williams, U. Wolfram,[a] and A. Ozel[a]

Email: josh.williams@stfc.ac.uk **Twitter:** @joshwill_96 **Website:** jvwilliams23.github.io

All code available!

Forward pass code, keras2cpp

```
//define model objects and compute prediction for drift_flux
keras::KerasModel DFnnModel_("Model/DFkerasParameters.nnet", false);
keras::DataChunkFlat dataChunk( num_features, 0.0);
vector<float>& features = dataChunk.get_1d_rw();
float scaled_dimless_drift_flux = 1.0e-6;
float drift_flux = 1.0e-6;
//collect features and normalize the features using mean and std input
unsigned int index = 0;
features.clear();
features.push_back((Reynolds_number - means_[index])/stds_[index]);
++index;
features.push_back((dimless_filter_size - means_[index])/stds_[index]);
++index;
features.push_back((scaled_solid_volume_fraction - means_[index])/stds_[index]);
++index;
features.push_back((dimless_grad_P - means_[index])/stds_[index]);
++index;
features.push_back((dimless_slip_velocity - means_[index])/stds_[index]);

//compute prediction
vector<float> prediction = DFnnModel_.compute_output( &dataChunk );
```