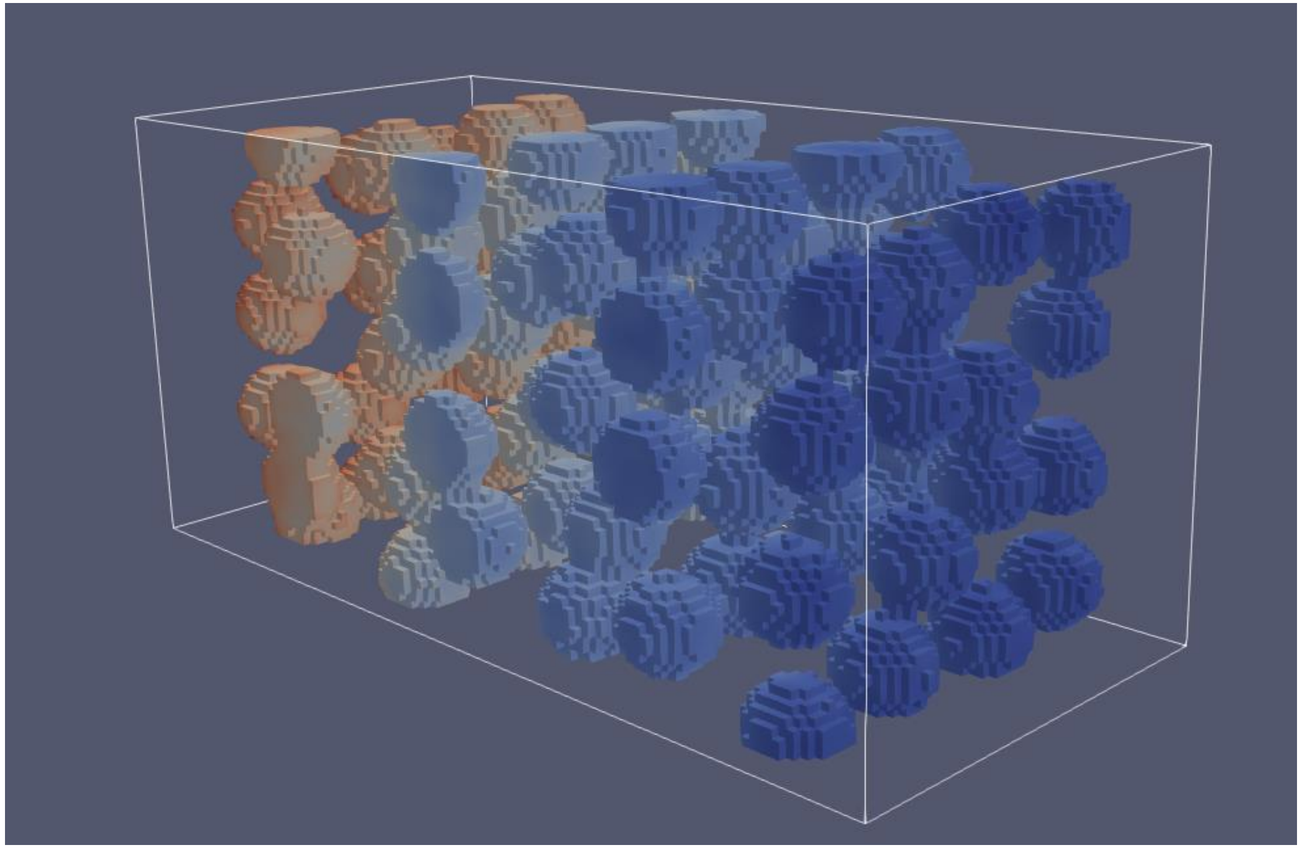
Friedrich-Alexander-Universität
Technische Fakultät

Master Thesis

Estimating Permeability of Porous Media with Fourier Neural Operators

by Lukas Schröder

Agenda

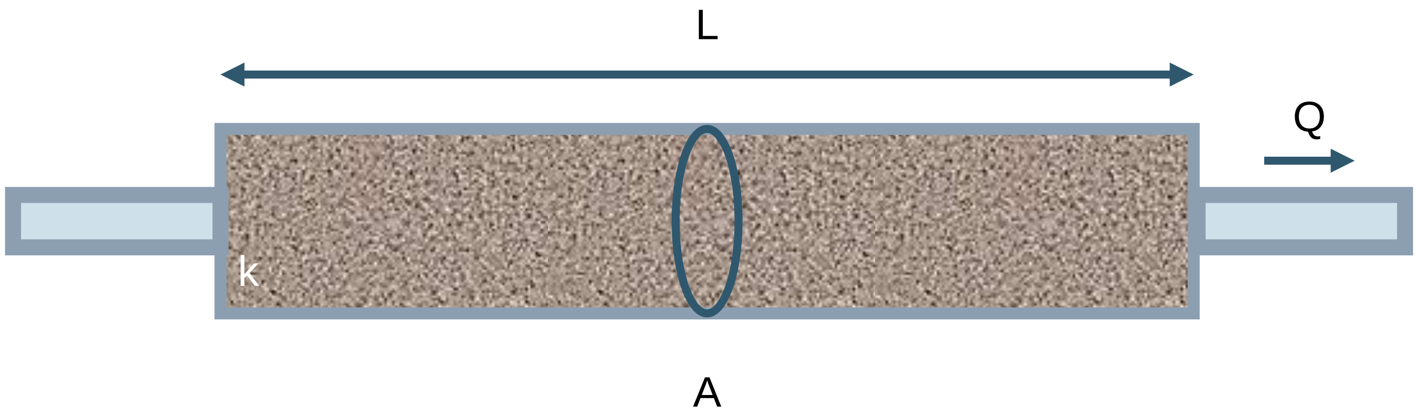


01 Task

01 Task

Goal

- Aim is finding the hydraulic resistance R of provided 3D media
  - Governed by Darcy's law in steady state laminar flows
  - $\mathrm{R} = \frac{\Delta p}{Q}$
- Hydraulic resistance can be calculated from a material property called permeability k together with the geometric scales
  - $R = \frac{\mu L}{kA}$



01 Task

Classical way of permeability estimation

Experiments

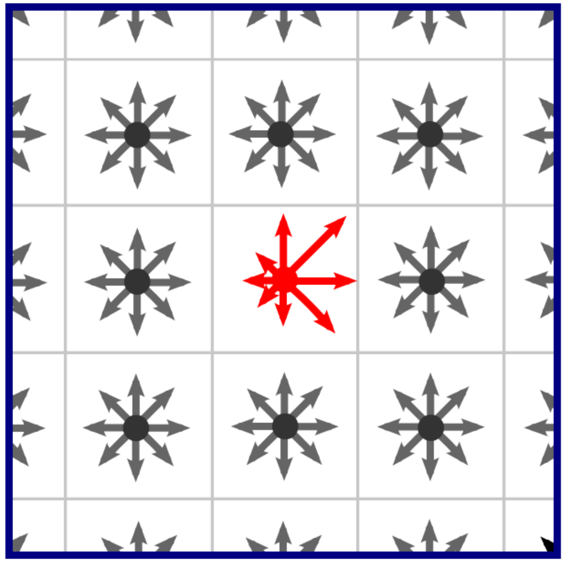
Build a container filled with porous media and measure the pressure difference

Analytical Solutions

For some geometries, like regular sand and certain rock compositions, there are formulas for permeability from porosity and other properties. Basically, finding correlations with easier to measure parameters.

Numerical Simulation

Derive k from the pressure difference of flow simulations like *Lattice Boltzmann methods* (LBM) or finite volumes.



01 Task

Thesis approach

Generate or Provide a Geometry → Calculate Flow with LBM → Train Machine Learning Model on Pressure Field → Extract Pressure Difference with Model

Generate or Provide a Geometry

- 3D grid of spheres from different radii and distances, shifted randomly
- Datasets of porous media

Calculate Flow with LBM

- Calculate velocities and pressure field

Train Machine Learning Model on Pressure Field

- Input: fill flag field
- Output: pressure field

Extract Pressure Difference with Model

- Calculate permeability from estimated pressure field and metrics
- Usage as surrogate model
- Very fast with inference only

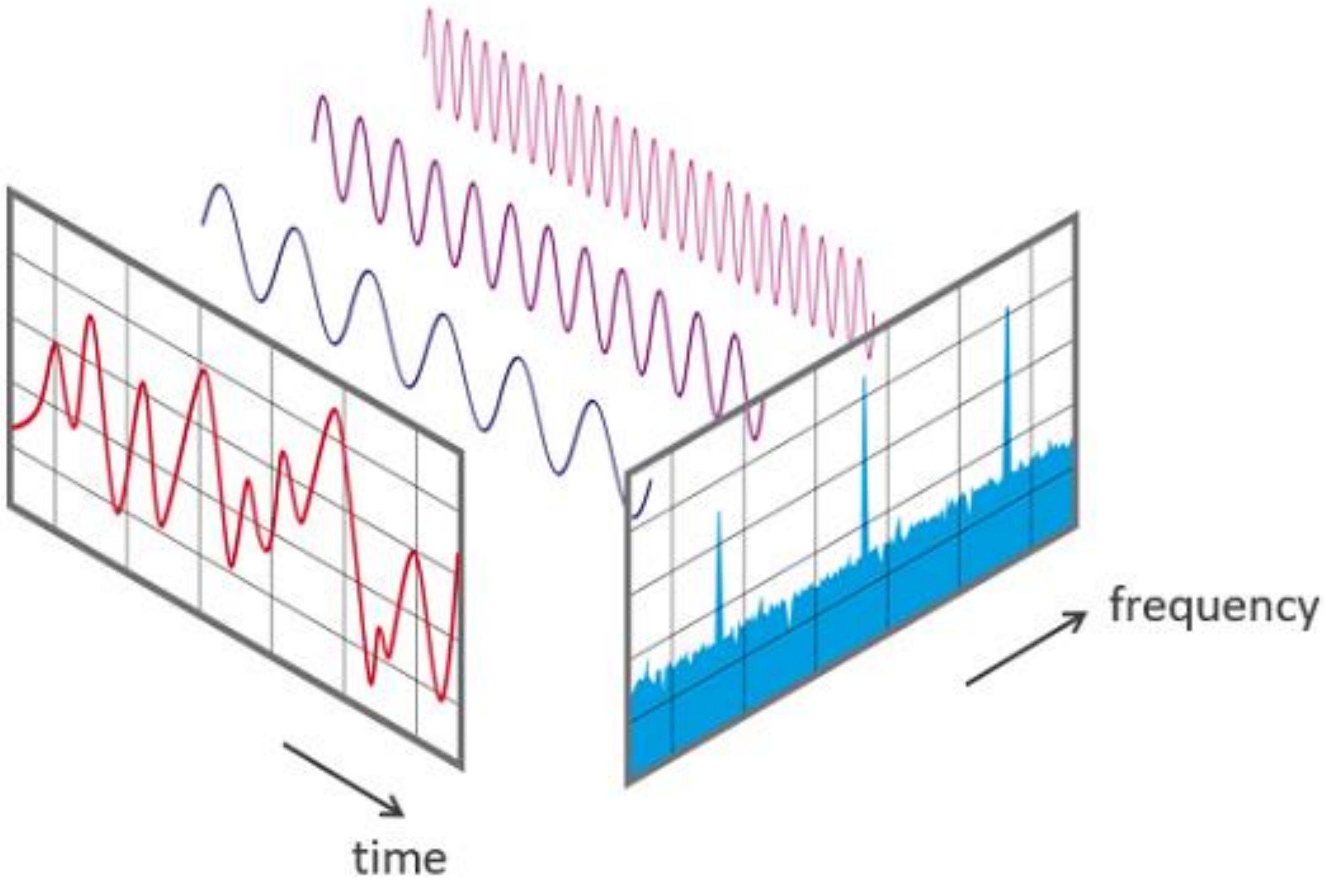
02 Fourier Neural Operators

02 Fourier Neural Operators

Neural Operator

- Idea to learn mapping G between two infinite dimensional function spaces A and U
    - $G: A \times \Theta \rightarrow U$ with finite collection of observed input-output pairs Θ
    - Used for example in learning PDEs
    - Independent of discretization
- Needs a lifting layer Q to transfer discrete input into higher dimensional codomain
    - Fully local operator
- Integral kernel operator K to work on entire domain
- Projection layer P to transport result back into discrete structures
    - *SIREN* or 2 linear layers

Fourier Transformation

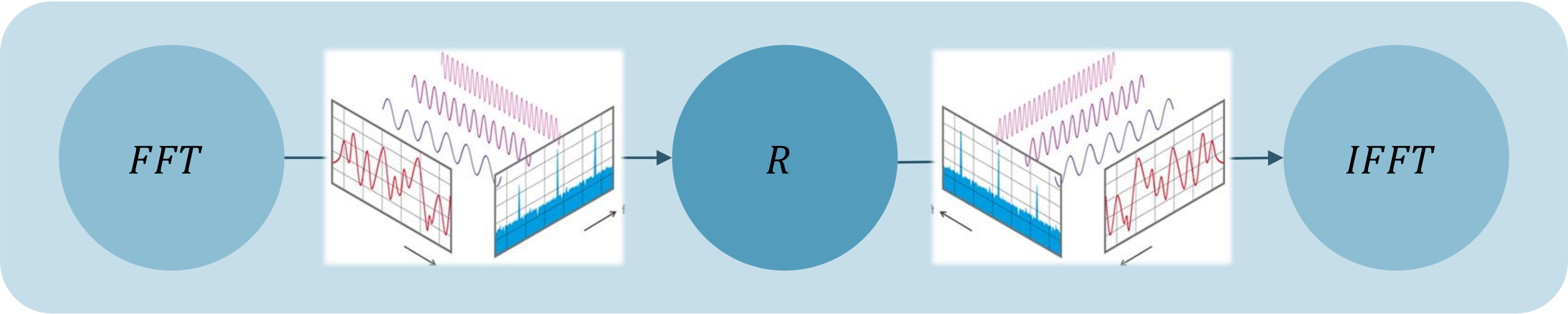


FFT (nti-audio.com)

02 Fourier Neural Operators

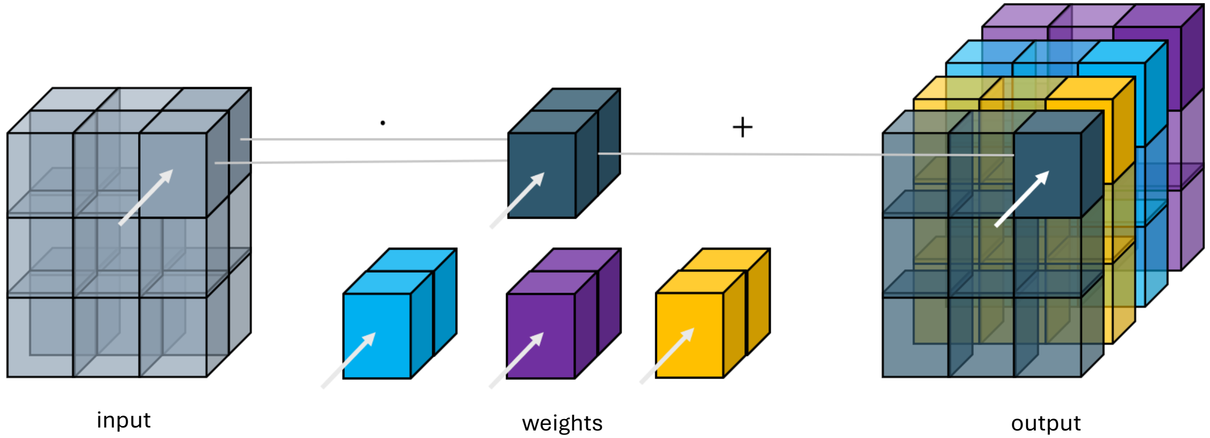
Fourier Neural Operator

- With the kernel integral operator
  - $K(x) = IFFT\big(R \cdot FFT(x)\big)$

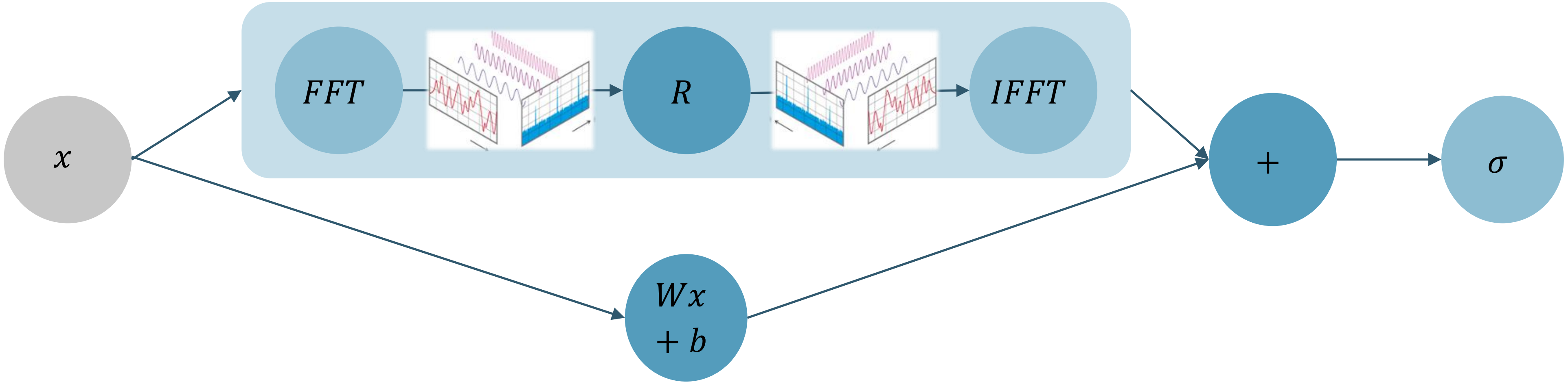


Fourier Neural Operator

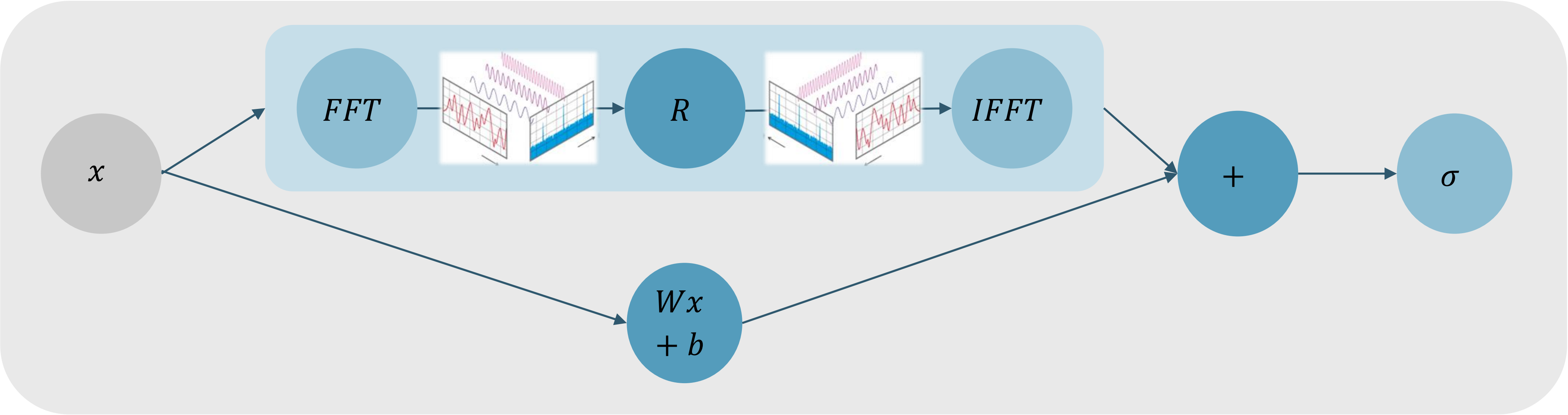
- With the kernel integral operator
  - $K(x) = IFFT\big(R \cdot FFT(x)\big)$
- And with 1x1 convolutions as weight application ($Wx + b$)



- With the kernel integral operator
  - $K(x) = IFFT\big(R \cdot FFT(x)\big)$
- And with 1x1 convolutions as weight application ($Wx + b$)
- Fourier neural operator apply linear transformation in the frequency space in layer L
  - $L(x) = \sigma(Wx + b + K(x))$

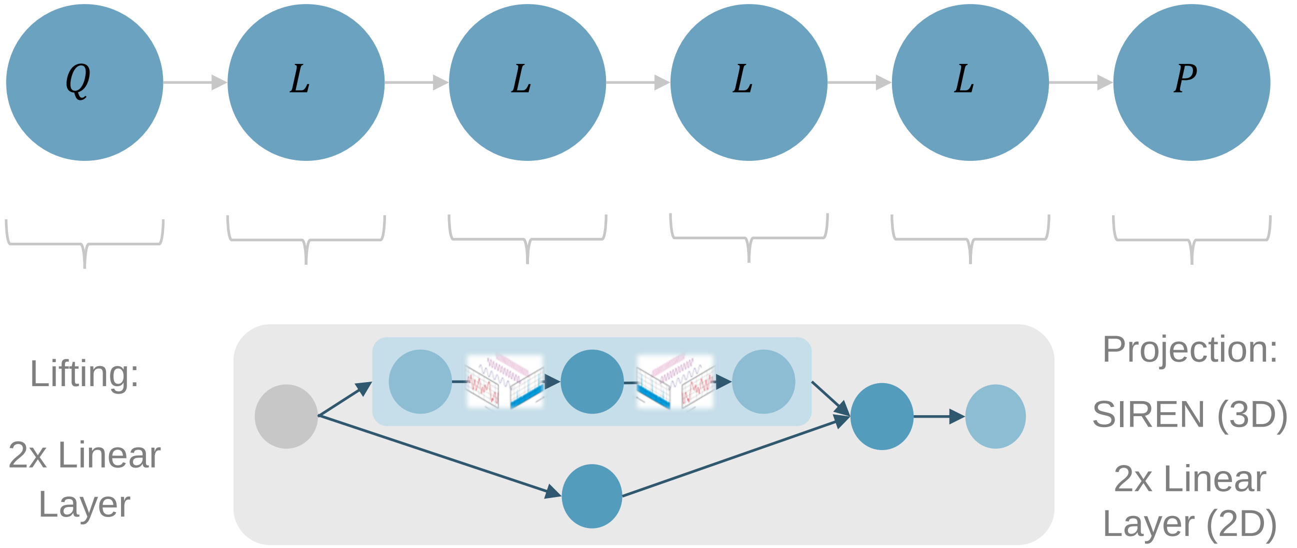


- Fourier neural operator apply linear transformation in the frequency space in layer L
  - $L(x) = \sigma(Wx + b + IFFT(R * FFT(x)))$



Fourier Neural Operator

- Fourier neural operator apply linear transformation in the frequency space in layer L
  - $L(x) = \sigma(Wx + b + IFFT(R * FFT(x))$
- Resulting model: $G: Q \circ \sigma(L^{(1)} \circ L^{(2)} \circ L^{(3)} \circ L^{(4)}) \circ P$



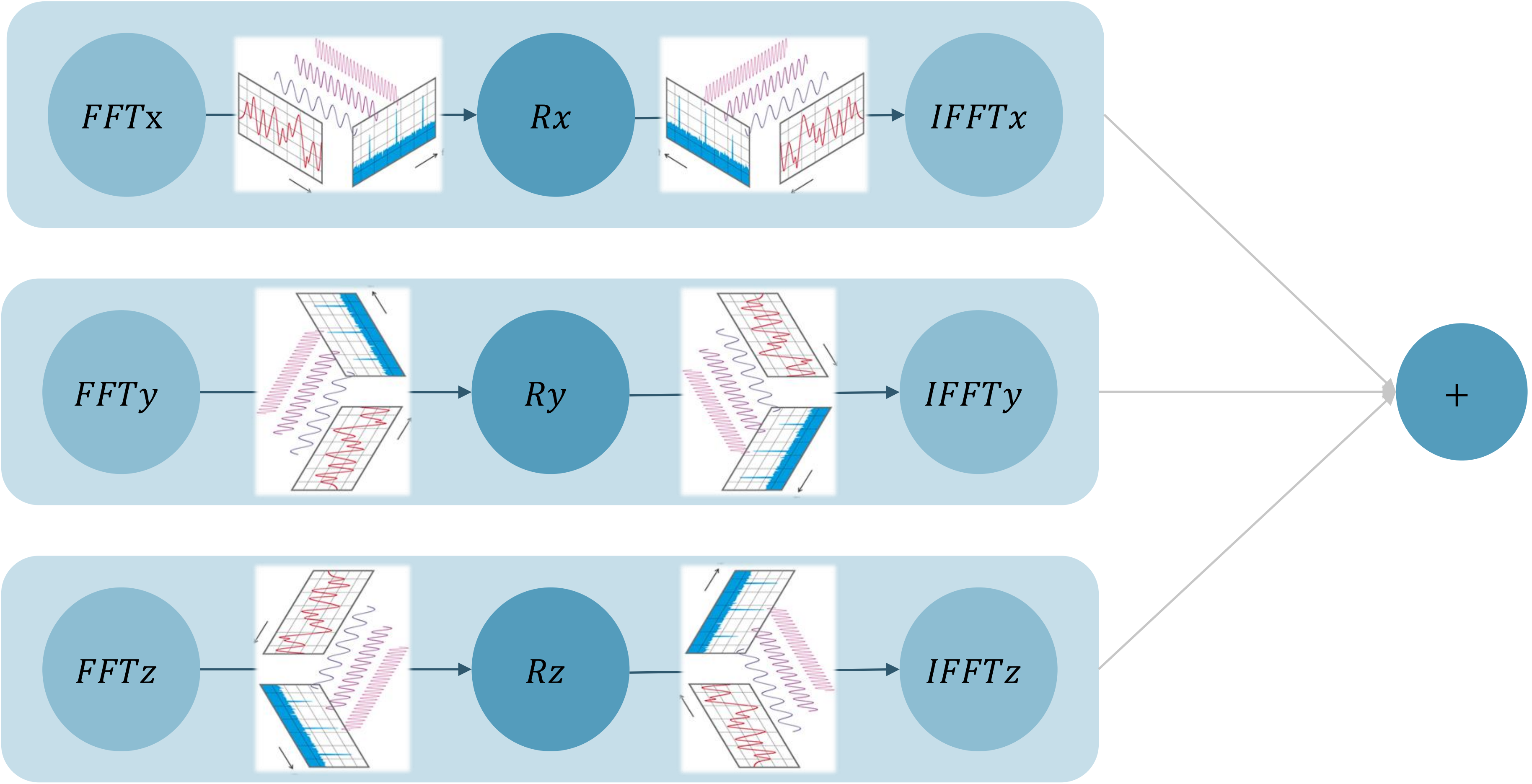
02 Fourier Neural Operators

Factorized Fourier Neural Operator

- *Factorized Fourier Neural Operator* for deeper networks
- Splits the Fourier kernel in the three dimension → Reduction of model complexity

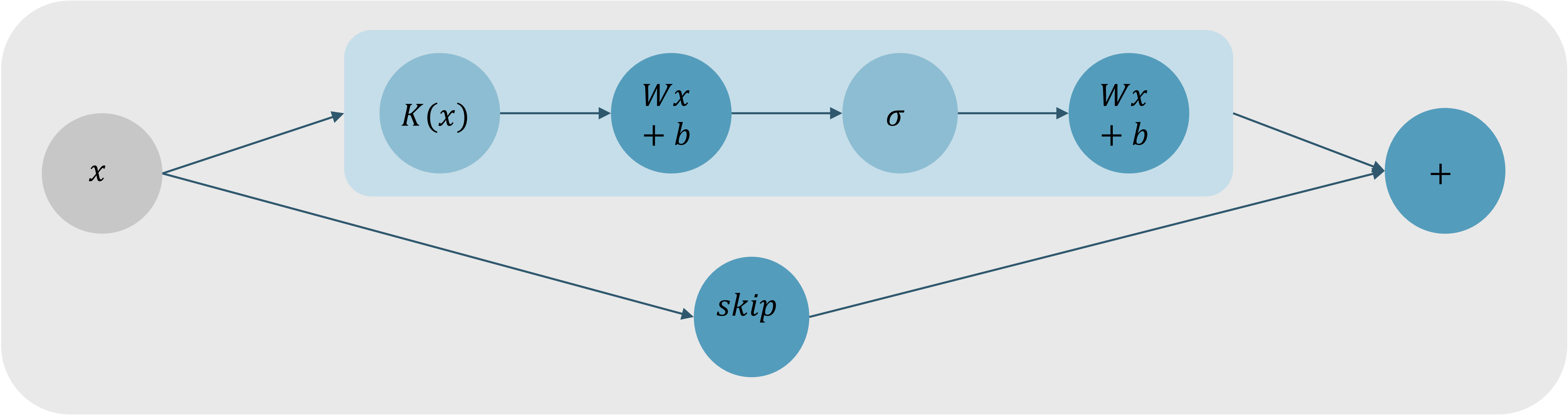
02 Fourier Neural Operators

Factorized Fourier Neural Operator



Factorized Fourier Neural Operator

- *Factorized Fourier Neural Operator* for deeper networks
- Splits the Fourier kernel in the three dimension → Reduction of model complexity
- Borrows concepts from classical deeper networks and transformers
  - Feed forward block with 2 linear layers per Fourier block
  - Residual connections



Factorized Fourier Neural Operator

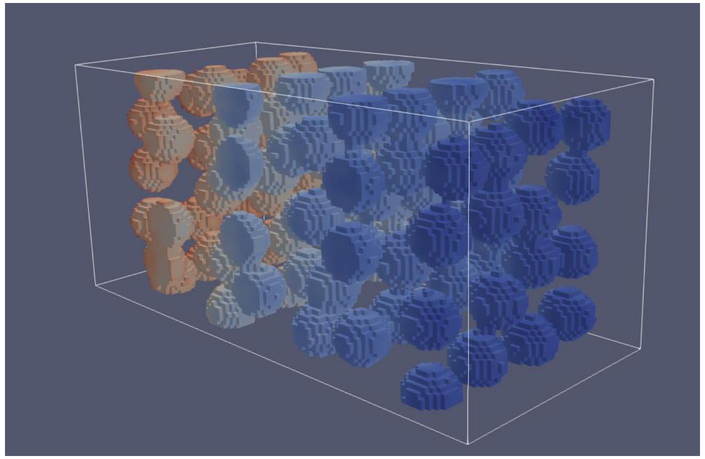
- *Factorized Fourier Neural Operator* for deeper networks
- Splits the Fourier kernel in the three dimension → Reduction of model complexity
- Borrows concepts from classical deeper networks and transformers
  - Feed forward block with 2 linear layers per Fourier block
  - Residual connections
  - Added weight normalization
  - Cosine learning rate scheduling with warmups

03 Experiments & Results

Overview

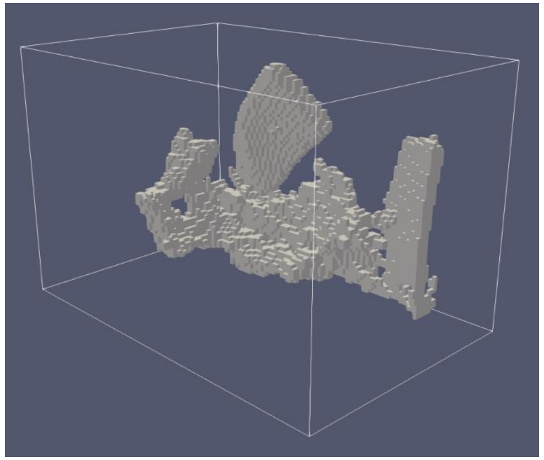
Packed Spheres

- Used for proving realizability
- Created from a waLBerla performance test setup with random shifts



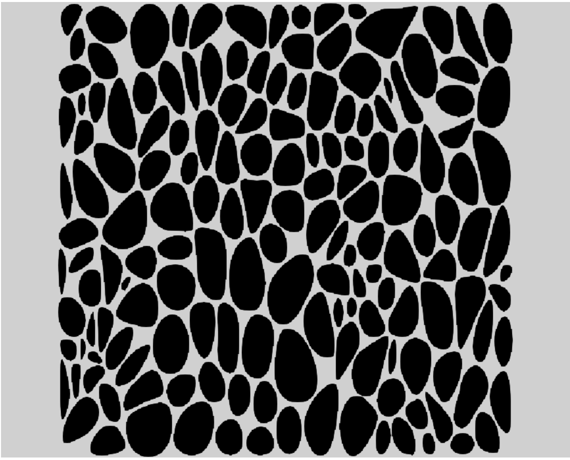
Digital Rock Project Dataset

- Small, complex diverse dataset for limit testing
- Overview set of 133 real and synthetic porous media



2D

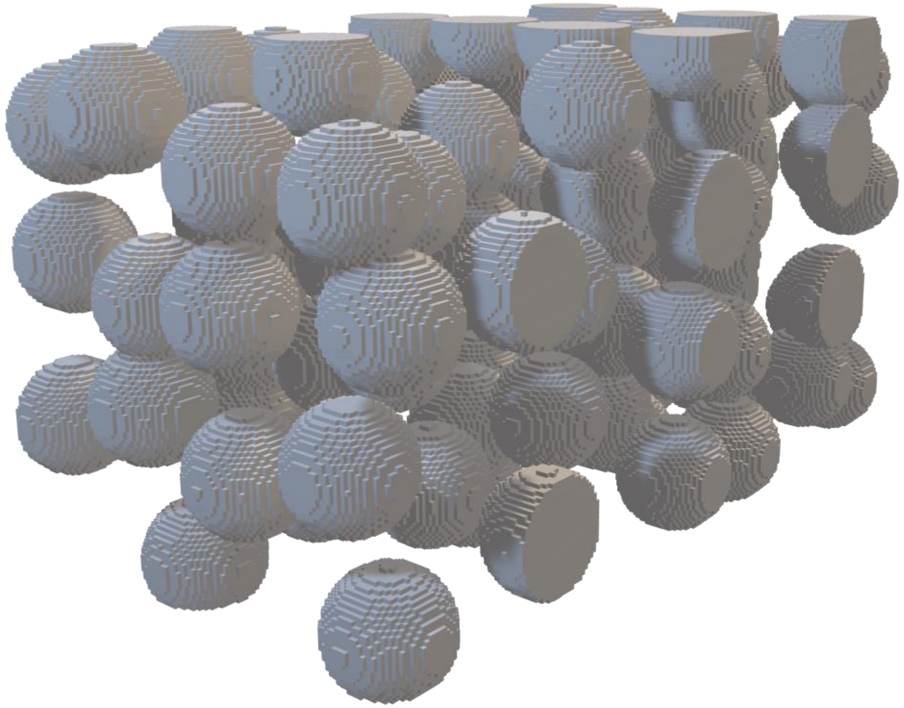
- From original paper used *in Impana Somashekars* thesis
- Used for comparison with Swin transformers performance



Realizability with Packed Spheres

Packed Spheres

- Used for proving realizability
- Created from a waLBerla performance test setup with random shifts
- $256x128x128$ domain filled with spheres of diameter $[15, 30]$ and distances $[\text{diameter} + 1, \text{diameter} + 8]$
- Setup
  - Input as $128x64x64$
  - $[32, 16, 16]$ modes in spectral kernel



Realizability with Packed Spheres

Results

- All reach an R^2 score $> 98\%$
- No performance increase for original model after 4 layers
- Factorized version performs better overall
- Proximity of the results
- Factorized models take longer than their original counterpart

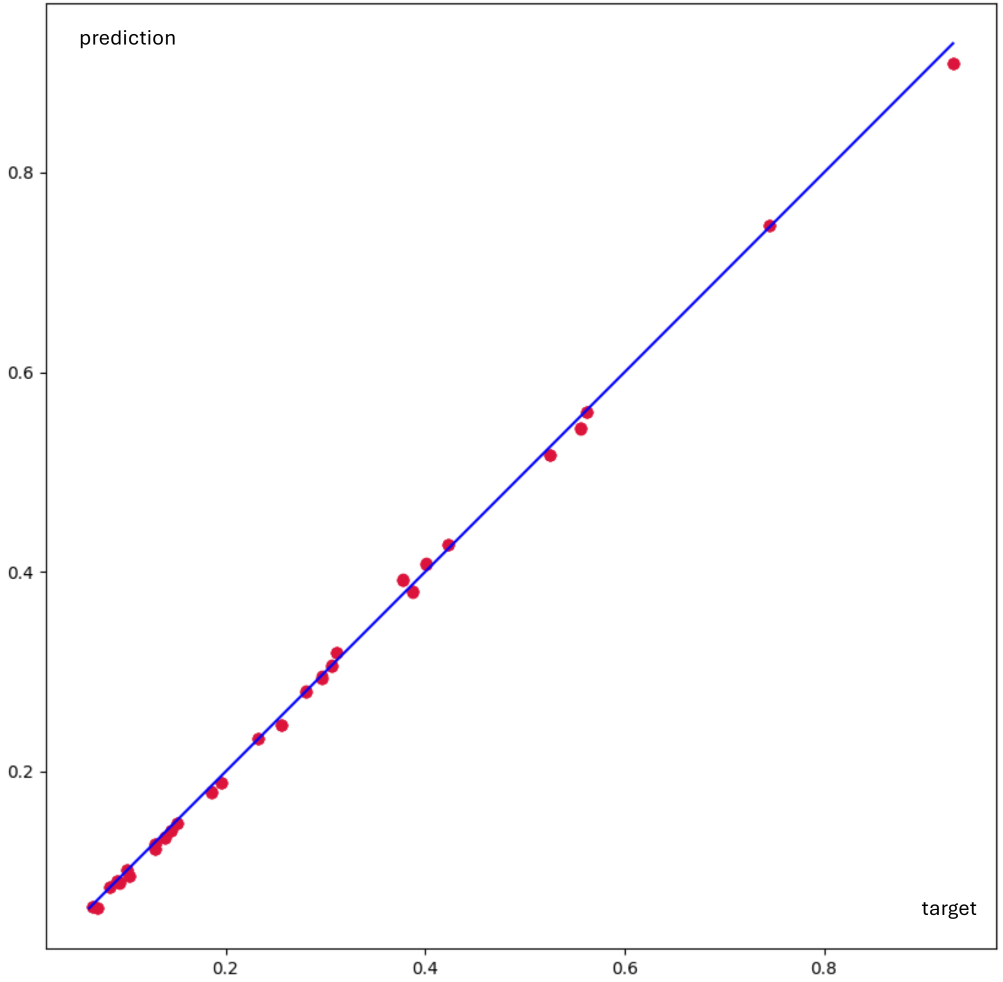
Factorized	Layers	MAE (10^{-2})	% R^2 Score	Max AE (10^{-2})	Time per epoch in s
False	2	1.20	99.18	6.86	**24.28**
False	4	1.11	99.35	5.93	44.66
False	6	1.46	98.84	7.01	54.75
False	8	1.34	99.17	5.56	74.12
True	2	1.25	99.42	4.10	48.23
True	4	**0.51**	**99.91**	**1.44**	102.32
True	6	0.73	99.80	2.29	131.28
True	8	0.56	99.88	1.98	171.56

Realizability with Packed Spheres

Results

- All reach an R^2 score $> 98\%$
- No performance increase for original model after 4 layers
- Factorized version performs better overall
- Proximity of the results
- Factorized models take longer than their original counterpart
- Best configuration: Factorized 4 layer

Factorized	Layers	MSE field (10^{-2})	MAE (10^{-2})	MAPE	% R^2 Score	Max AE (10^{-2})
True	4	**0.006**	**0.51**	**2.02**	**99.91**	**1.44**



Packed Spheres

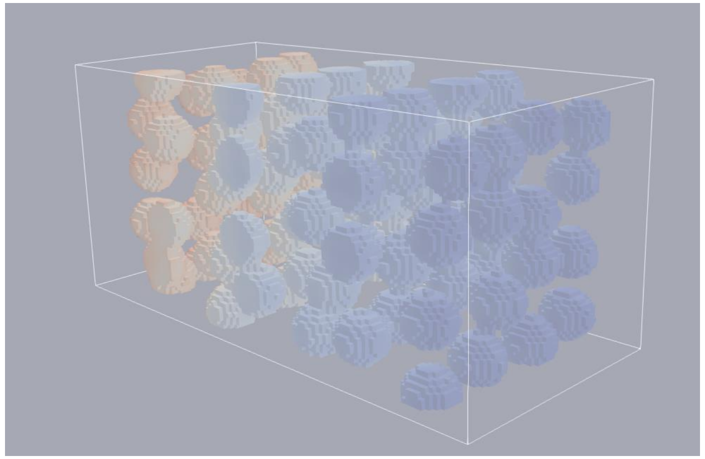
- Used for proving realizability
- Created from a waLBerla performance test setup with random shifts

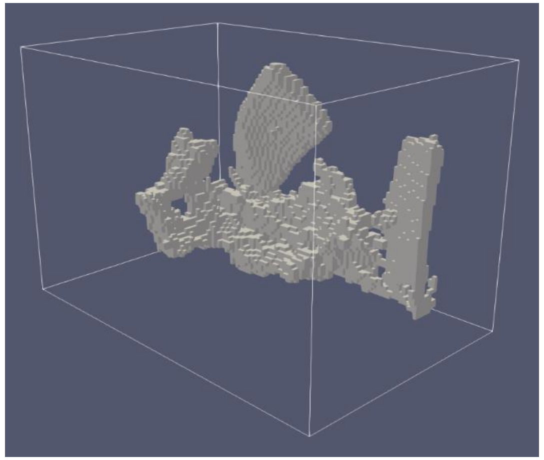
Digital Rock Project Dataset

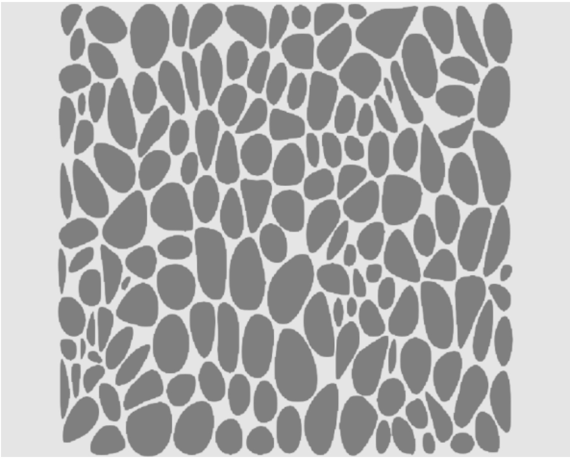
- Small, complex diverse dataset for limit testing
- Overview set of 133 real and synthetic porous media

2D

- From original paper used *in Impana Somashekars* thesis
- Used for comparison with Swin transformers performance

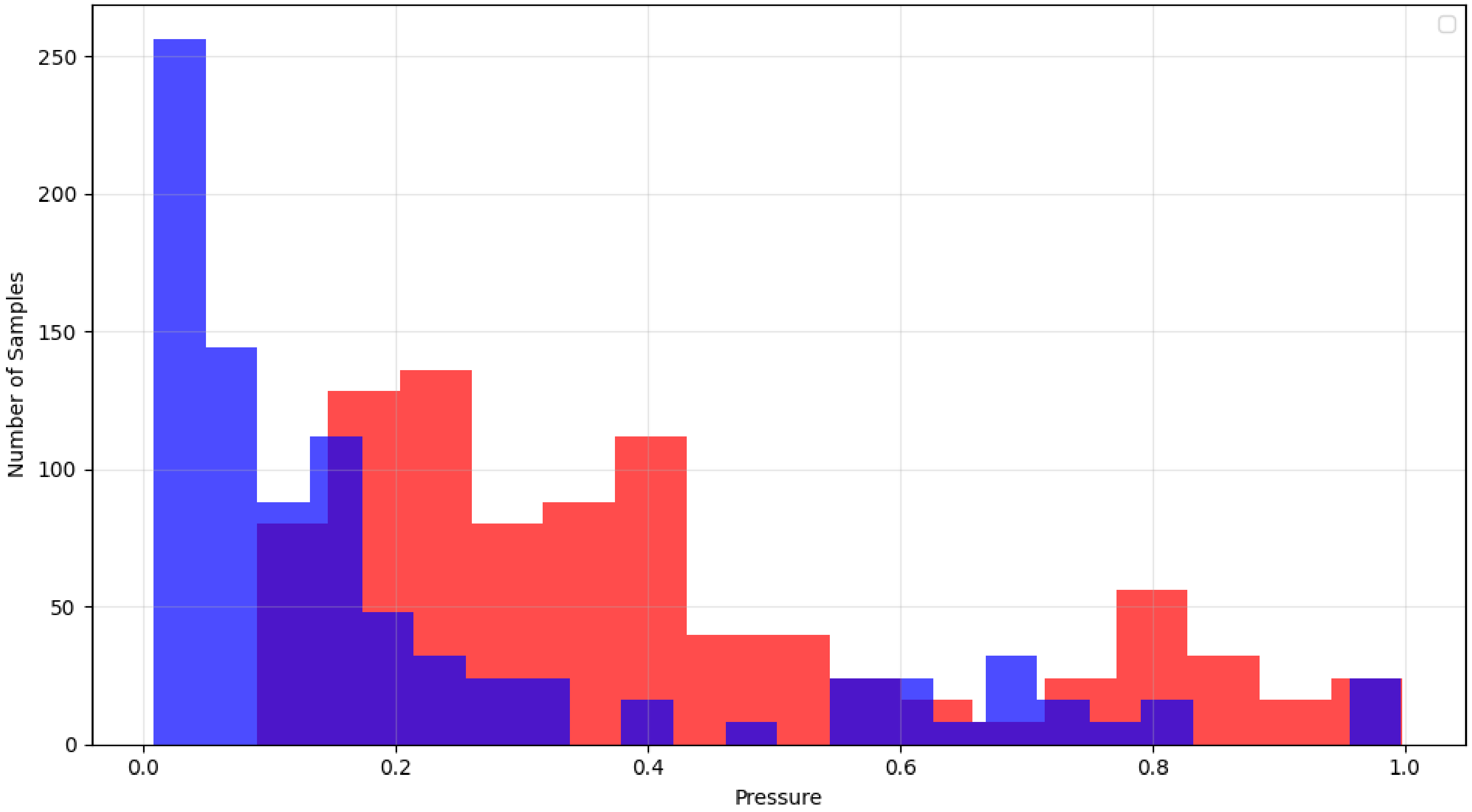






Digital Rock Project Dataset

- Small, complex diverse dataset for limit testing
- Overview set of 133 real and synthetic porous media
- Simulated with lbmpy
- Data transformed with $x_{\text{trans}} = \sqrt{x}$



Challenging DRP Dataset

Results

- Hard to achieve R^2 Score $> 80\%$
- High maximum absolute errors > 0.4
- High intra-model score fluctuations

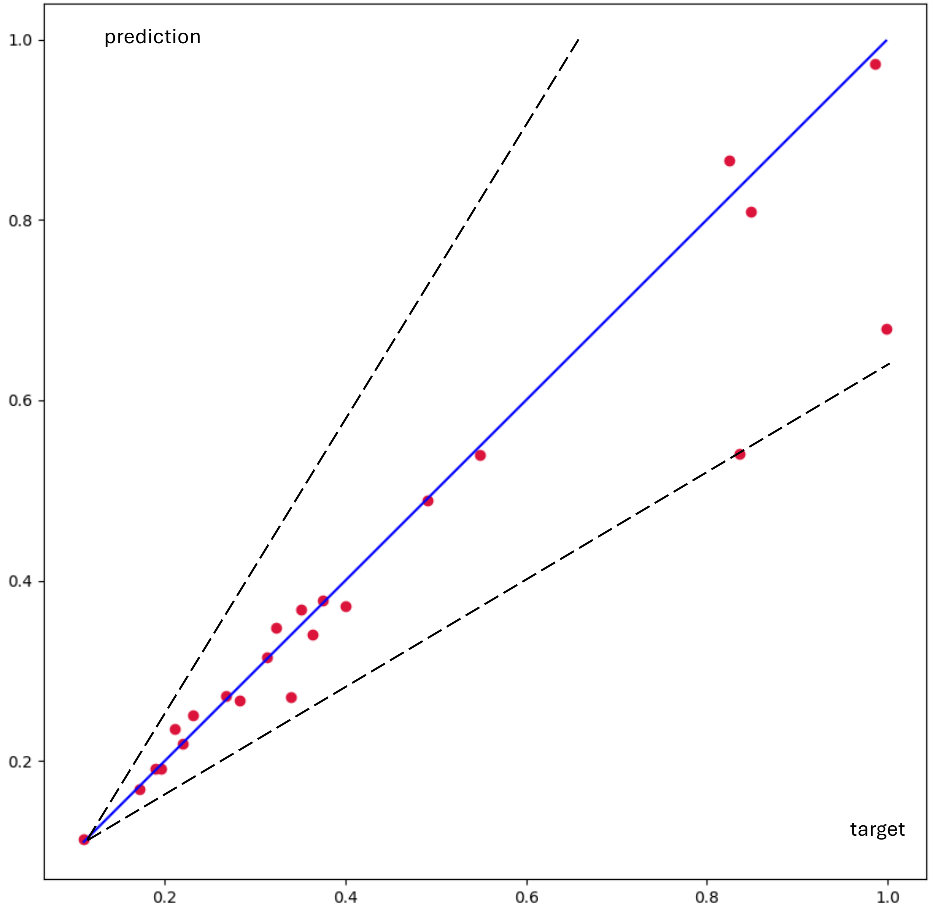
Factorized	Layers	MAE (10^{-2})	% R^2 Score	Max AE (10^{-2})	Time per epoch in s
False	2	6.28	79.19	46.26	**15.15**
False	4	6.56	79.85	52.77	24.97
False	8	5.98	76.43	51.40	45.09
True	2	7.46	73.11	59.94	27.21
True	4	6.00	68.19	70.02	49.50
True	8	**4.42**	**88.54**	**37.48**	102.87
True	12	5.40	77.57	65.52	139.31

Challenging DRP Dataset

Results

- Hard to achieve R^2 Score $> 80\%$
- High maximum absolute errors > 0.4
- High intra-model score fluctuations
- Hard to predict high pressure cases drive errors
  - Idea: Filter highest 10% out

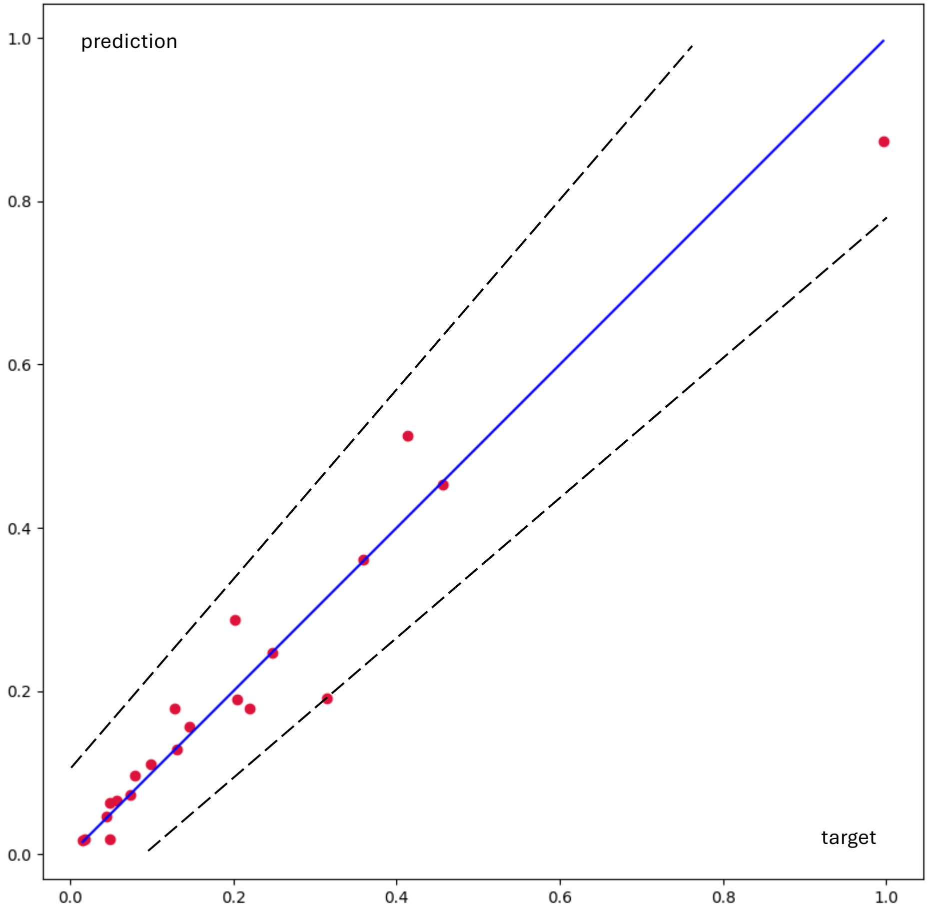
Factorized	Layers	MAE (10^{-2})	% R^2 Score	Max AE (10^{-2})
True	8	4.42	88.54	37.48



Results

- Hard to achieve R^2 Score $> 80\%$
- High maximum absolute errors > 0.4
- High intra-model score fluctuations
- Hard to predict high pressure cases drive errors
    - Idea: Filter highest 10% out

Filtered	MAE (10^{-2})	% R^2 Score	Max AE (10^{-2})
True	**3.06**	**94.62**	**12.34**
False	4.42	88.54	37.48



Packed Spheres

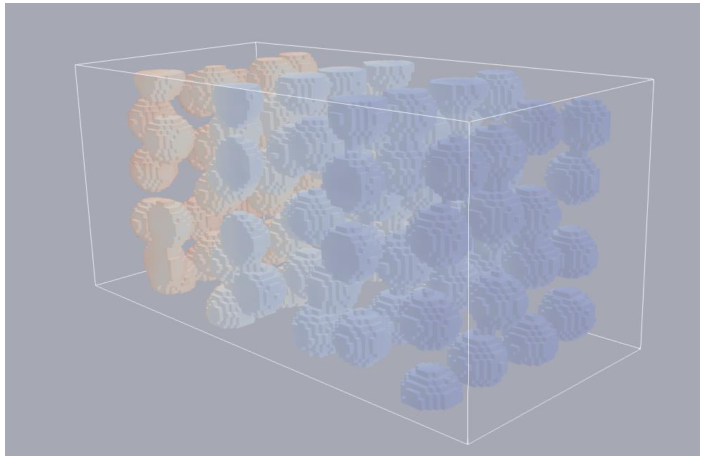
- Used for proving realizability
- Created from a waLBerla performance test setup with random shifts

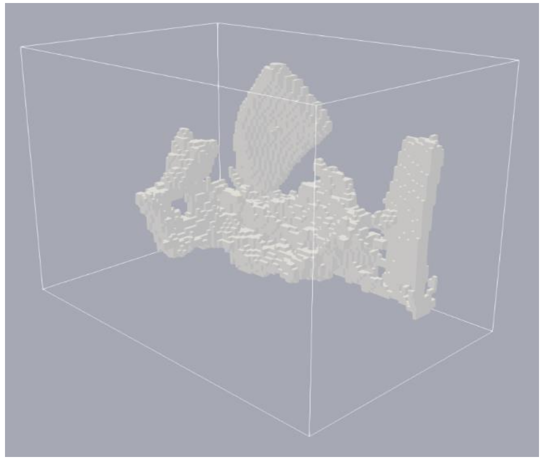
Digital Rock Project Dataset

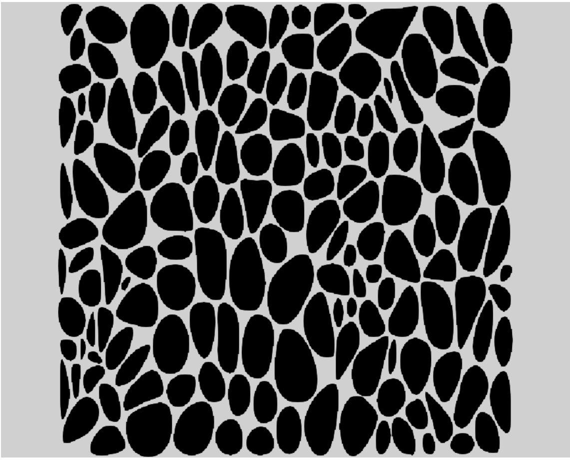
- Small, complex diverse dataset for limit testing
- Overview set of 133 real and synthetic porous media

2D

- From original paper used *in Impana Somashekars* thesis
- Used for comparison with Swin transformers performance



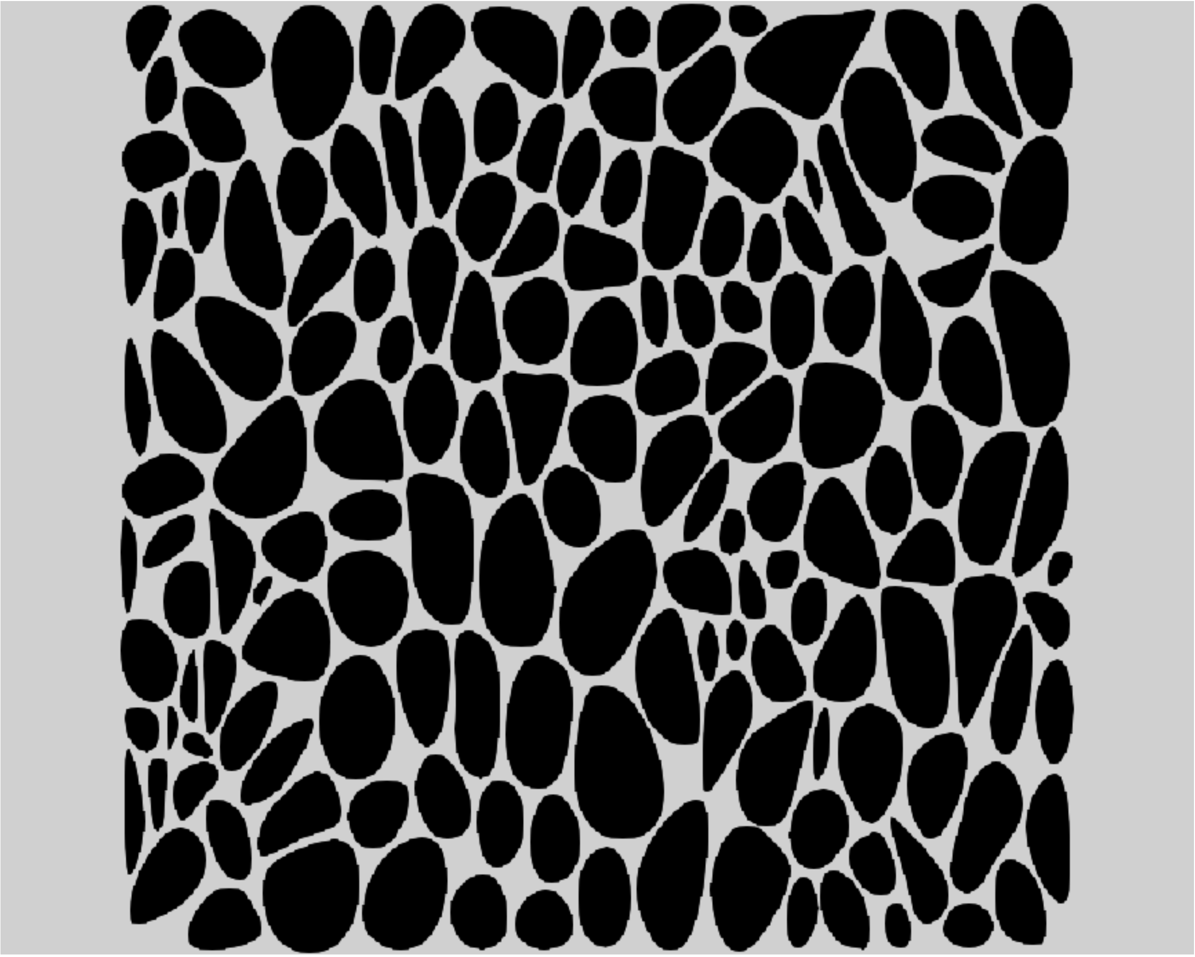




2D Comparison

2D

- From original paper used *in Impana Somashekars* thesis
- Used for comparison with Swin transformers performance
- Simulated with *lbmpy*
- Sets: 1000 generated polynomials, 600 rough sandstone and 200 rocks



2D Comparison

Results

- Good results on standard, full and rock set with R^2 score $> 94\%$
- Noisy set fails with R^2 score $< 50\%$
- Comparison
  - Slightly improved performance in R^2 scores and maximum AE
  - Training takes only 6% of the transformers time!
  - Noisy set still works with the transformer

Model	Dataset	% R^2 Score	Max AE (10^{-2})	Time per epoch in s
FFNO 4-layer	standard	**98.78**	**10.68**	**8.62**
Swin Transformer	standard	98.14	~12	~140
FFNO 8-layer	rocks (on full)	93.29	-	**32.36 (entire set)**
Swin Transformer	rocks	**93.82**	**-**	~80
FFNO 8-layer	noisy (on full)	47.99	-	**32.36 (entire set)**
Swin Transformer	noisy	**92.35**	-	~30

04 Discussion

04 Discussion

Hyperparameter for FNOs

- AdamW, weight normalization and the cosine learning rate scheduling allow for **short training time and stability**

- Original FNO architecture deliver **reliable results in a faster time**
- Original architecture does **not improve with added depth** after 4 layers

- Factorized FNO architecture delivers the **best results** in all tested cases, but not always meaningfully better
- Factorized FNOs **scale better with more layers** until 8 layers (here)

→ 2/4-layer original FNO for efficiency

→ 8-layer factorized FNO for maximum performance

04 Discussion

Use Cases for FNO based Surrogate Models

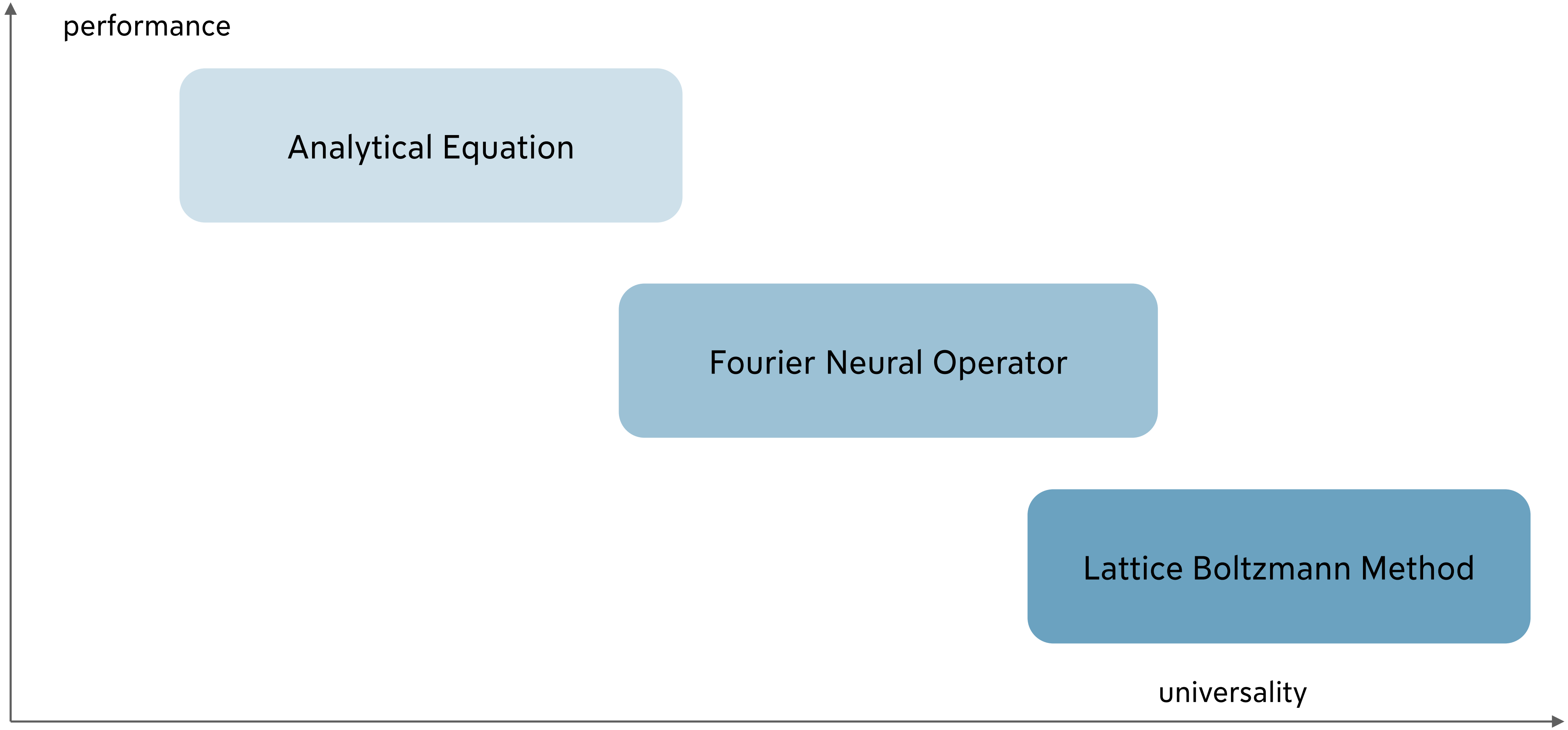
Comparison with Analytical Solution

- Formulaic solution are evaluated instantly
- Limited range of applicable geometries
- Performance Example:
  - Shifted spheres with *Kozeny-Carman* equation
  - R^2 score 95.04% (FFNO 4-layer 99.91%)

Comparison with LBM

- LBM is more accurate and calculations can be parallelized
- Can be simulated for any geometry
- Speed comparison
  - One sample $\sim 207s$ with lbmpy (vs $\sim 0.15s$ inference) on the same GPU
  - Breaking even point with FNO training, simulating training samples and evaluation:
    - ~ 275 evaluation samples
    - when trained on 250 simulated geometries

Use Cases for FNO based Surrogate Models

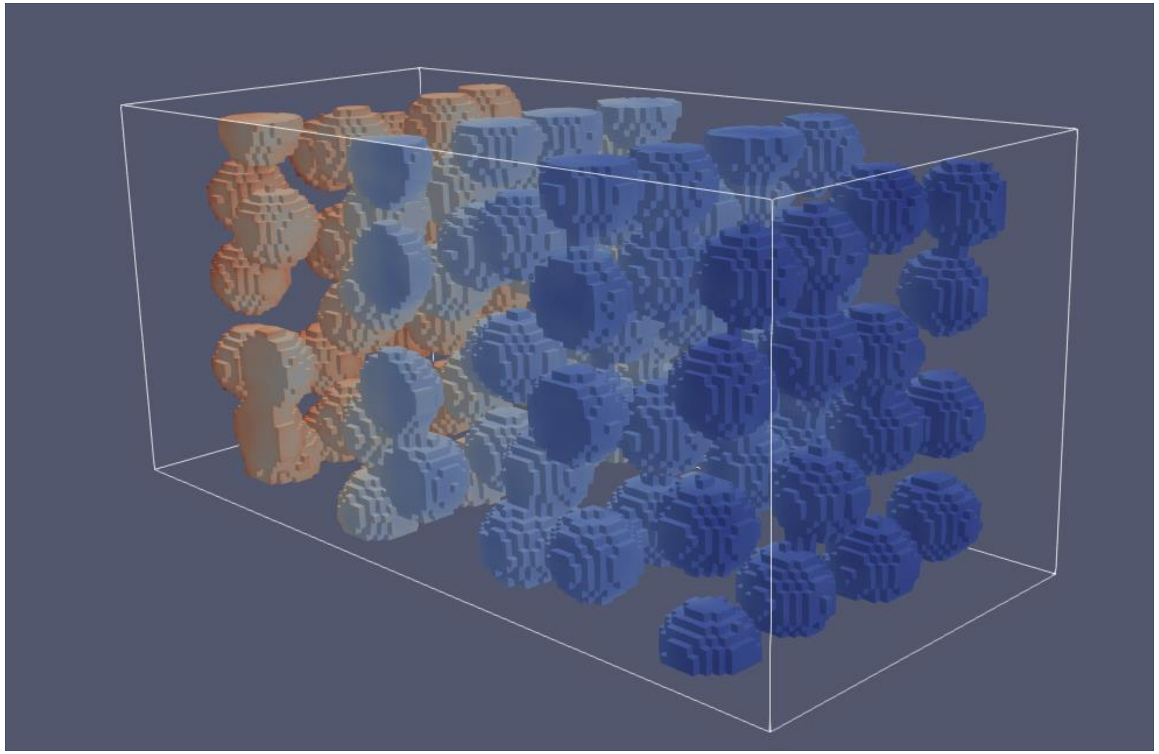


05 Conclusion & Outlook

05 Conclusion & Outlook

5 Learnings

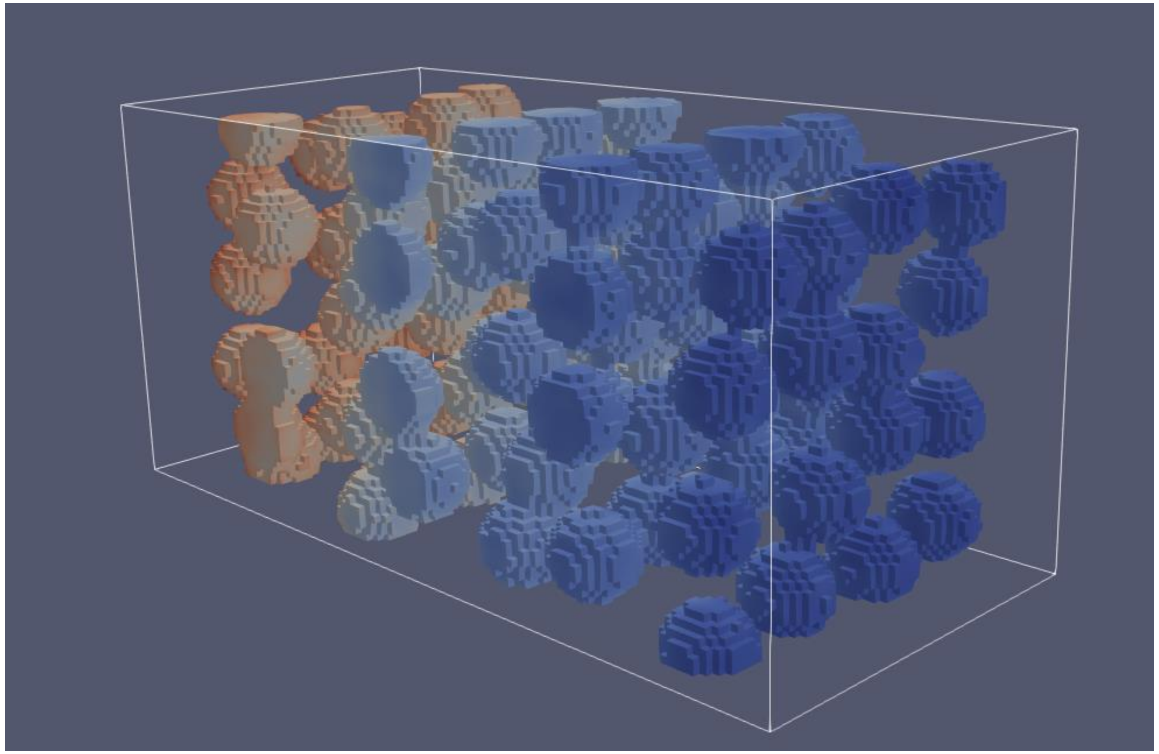
- Fourier neural operators can capture the pressure field to a reliable degree
- When enough geometries are provided of similar cases, wide ranges of cases can be predicted
- Original architecture is best for efficiency, while the factorized performs best
- It can bridge the range of LBM with the speed closer to analytical solutions with a state-of-the-art NN
- Use case is when many similar geometries need to be evaluated quickly



05 Conclusion & Outlook

Outlook

- Test scalability with big and complex databases
- Investigations of physically informed neural networks (PINNs)
- Comparison with more modern and elaborate analytical solution
- Test FNOs on entire flow prediction



Thanks for Your Attention