

## Benchmarking PINN architectures for temperature-reconstruction by determining viscosity deviations

<sup>1</sup>Michael Mommert\*; <sup>1,2</sup>Marie-Christine Volk; <sup>1</sup>Christian Bauer

<sup>1</sup>German Aerospace Center (DLR), Institute of Aerodynamics and Flow Technology,  
Department Ground Vehicles, Bunsenstr. 10, 37073 Göttingen, Germany;

<sup>2</sup>Technische Universität Ilmenau, Institute of Thermodynamics and Fluid  
Mechanics, 98693 Ilmenau, Germany

**Keywords:** PINN, temperature reconstruction, thermal convection, trainable viscosity

**Abstract.** In search of efficient PINN architectures to reconstruct temperature fields from velocity vector data, we explore the usage of the dimensionless viscosity  $\nu = \sqrt{\text{Pr}/\text{Ra}}$  as trainable parameter of a PINN model. The results show that it improves the temperature reconstruction results. The estimation of the viscosity through the training can further be used as performance indicator, when an expected value is known.

### 1. Introduction

Since their introduction [1], physics-informed neural networks (PINNs) have been brought to a great variety of applications [2]. The particular application investigated here is the reconstruction of temperature fields [3, 4], prospectively based on velocity measurements by particle image velocimetry [5] or particle tracking velocimetry [6, 7].

While this type of application is currently predominantly investigated for canonical flows, like Rayleigh-Bénard convection, future applications to non-canonical flow problems pose questions regarding the non-dimensionalization of the Navier-Stokes equations. In this context, the dimensionless numbers, and especially the Rayleigh number  $\text{Ra}$ , which sets the convective forcing in relation to diffusive forces, are often not clearly defined.

Hence, we examine whether  $\text{Ra}$  can be unknown to reconstruct temperature and pressure fields from sparse velocity fields. This also raises the question whether  $\text{Ra}$  can be reconstructed from the dimensionless viscosity  $\nu = \sqrt{\text{Pr}/\text{Ra}}$  of the Navier-Stokes equations with adequate accuracy within the same process.

Vice versa, if prior knowledge of  $\text{Ra}$  exists, its reconstruction accuracy can be used as a performance indicator for the field's reconstruction, which is investigated in the second part of the paper for different network architectures.

### 2. Use Case

The use case for the presented investigation is Rayleigh-Bénard convection inside a cubic enclosure. Its boundaries are no-slip walls. While the side walls are considered as adiabatic, the bottom and top faces are set to have constant temperatures  $T_H = 0.5$  and  $T_C = -0.5$ , respectively.

Together with the respective quantities expressed in dimensional units (indicated by  $\hat{\cdot}$ ), namely the gravitational acceleration  $\hat{g}$ , the thermal expansion coefficient  $\hat{\alpha}$ , the thermal diffusivity  $\hat{\kappa}$ , the kinematic viscosity  $\hat{\nu}$ , and the height of the domain  $\hat{H}$ , the system becomes fully described by the Prandtl number  $\text{Pr} = \hat{\nu}/\hat{\kappa}$  and the Rayleigh number  $\text{Ra} = \hat{g}\hat{\alpha}(\hat{T}_H - \hat{T}_C)\hat{H}^3/(\hat{\nu}\hat{\kappa})$ .

By applying the Oberbeck-Boussinesq approximation, the resulting three-dimensional flow is governed by the following set of partial differential equations, in which  $p$  is the pressure field:

$$\partial_t \mathbf{u} + (\mathbf{u} \cdot \nabla) \mathbf{u} = -\nabla p + \sqrt{\text{Pr}/\text{Ra}} \nabla^2 \mathbf{u} + T \mathbf{e}_z \quad (1)$$

$$\partial_t T + (\mathbf{u} \cdot \nabla) T = \sqrt{1/(\text{Ra} \text{Pr})} \nabla^2 T \quad (2)$$

$$\nabla \cdot \mathbf{u} = 0 \quad (3)$$

In order to evaluate the PINN performance against ground truth data, a direct numerical simulation of the flow at  $Pr = 0.71$  and  $Ra = 10^8$  were conducted (for details, see [4] or [5]). Figure 1 gives an impression of the resulting turbulent flow, with hot and cold plumes detaching from the bottom and top plates, respectively, which is visualised through a volume rendering of the fluid temperature.

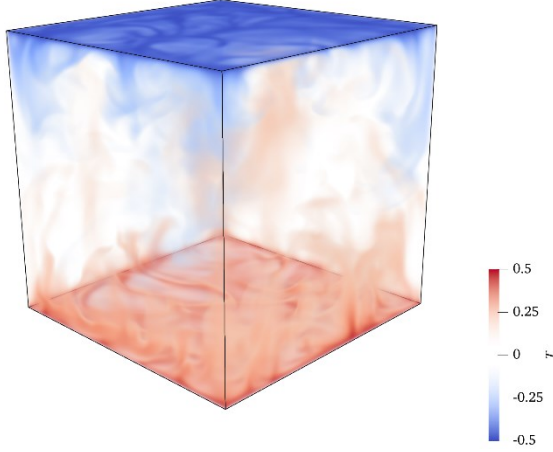


Figure 1: Instantaneous volume rendering of the fluid temperatures in cubic Rayleigh-Bénard convection at  $Ra=10^8$  and  $Pr=0.71$ .

To emulate measured velocity ( $\mathbf{u}$ ) data, the data provided to the PINN is interpolated at randomly distributed points. Spatially, Poisson disk sampling [8] is employed for this task. In contrast to mimicking particle tracking data, these positions remain locked for the progression over the selected time steps, to avoid any leakage of spatial resolution. Related to the dimensionless parameters sample height  $H$  and free-fall time  $t_{ff} = \sqrt{\hat{H}/\hat{g}\hat{\alpha}(\hat{T}_H - \hat{T}_C)}$ , the here examined sampling densities are  $0.04H$  in space and  $0.12t_{ff}$  in time. To account for statistical fluctuations of the PINN method, each case is tested for three different random samplings according to the aforementioned parameters, covering  $2.46t_{ff}$  at between 9060 and 9124 spatial locations, which is well within the limit achievable for a real experimental setup [7].

### 3. PINNs with trainable viscosity

PINNs are neural networks, which are set up to represent the function  $[\mathbf{u}, T, p] = \mathbf{f}([x, y, z, t], \boldsymbol{\theta})$ , where  $\boldsymbol{\theta}$  is the totality of trainable parameters, including the trainable viscosity.

For the following investigation, we consider three different architecture types to pose as the function  $\mathbf{f}$ . The first one is represented by sine-activated multi-layered perceptrons (MLPs) [9, 4]. They are a sequence of  $N_l$  dense layers with  $N_n$  neurons each. Of these layers, the weights of the first one are initialised with a three-times wider distribution than the standard for sine-activated layers [9] to provide higher wavenumbers. This network type also includes a final linear dense layer which combines the  $N_n$  features of the of the penultimate layer to the 5 output fields [5].

The second architecture considered are modified Pirate-Nets (PNs) [10], which are an architecture based on residual blocks with gating operations referring to shallow latent features. The here examined networks differ from the original publication in the following aspects: All layers are sine-activated. Therefore, the first layer, which originally acts as a fixed Fourier embedding, is implemented as the same sine-activated layer, with a wider weight initialization like for the MLP. The outputs of these networks are also combined by an additional linear layer, as in the MLP. This way, we isolate the effect of the introduction of the shallow-latent-feature layers incorporated by gating operations in the first two layers of each of the three-layer-wide blocks, the residual operation for the last layer of these  $N_b$  blocks, and the usage of random weight factorization layers [10] throughout the blocks.

The third and last considered architecture type is the Kolmogorov-Arnold network with Chebychev polynomials (cKANs), which was e.g. used by [6]. For this type of architecture, each layer first performs a non-linear (Chebychev polynomial of 5<sup>th</sup> degree) transformation for each of its inputs, before they are

combined by summation. These networks are also specified by their number of layers  $N_l$  and neurons  $N_n$ .

For the present comparison, these networks were trained by an Adam optimizer with the learning rate  $10^{-4}$ , with the combined loss function

$$\mathcal{L} = \mathcal{L}_{data} + 10^{-1}\mathcal{L}_{NS} + 10^{-2}\mathcal{L}_{EE} + 10^{-3}\mathcal{L}_{continuity} + \mathcal{L}_{b-p} + 10^{-3}\mathcal{L}_{b-s} + 10^{-4}\mathcal{L}_{b-T}; \quad (4)$$

$$\mathcal{L}_{NS} = (\partial_t \mathbf{u} + (\mathbf{u} \cdot \nabla) \mathbf{u} + \nabla p - \nu \nabla^2 \mathbf{u} - T \mathbf{e}_z)^2, \quad (5)$$

$$\mathcal{L}_{EE} = (\partial_t T + (\mathbf{u} \cdot \nabla) T - (\nu / \text{Pr}) \nabla^2 T)^2, \quad (6)$$

$$\mathcal{L}_{continuity} = (\nabla \cdot \mathbf{u})^2. \quad (7)$$

The data, sampled as described in section 2, is divided into batches of size 4096 for each training step. The losses  $\mathcal{L}_{NS}$ ,  $\mathcal{L}_{EE}$ , and  $\mathcal{L}_{continuity}$  are simultaneously evaluated at the same number of collocation points, random-uniformly sampled within the domain for each step. Regarding the trainable parameter  $\nu$ , only gradients originating in  $\mathcal{L}_{NS}$  are considered for its optimization. Also, random-uniformly sampled, 512 boundary point are generated on the faces of the cube for each step. On all faces, a mean-squared error (MSE) loss is computed for the wall-normal ( $\mathcal{L}_{b-p}$ ) and wall-parallel ( $\mathcal{L}_{b-s}$ ) velocity components. On the top and bottom faces, the temperature is prescribed via an MSE loss ( $\mathcal{L}_{b-T}$ ). For comparability, all investigated cases cover  $10^6$  optimization steps.

#### 4. Rayleigh number reconstruction

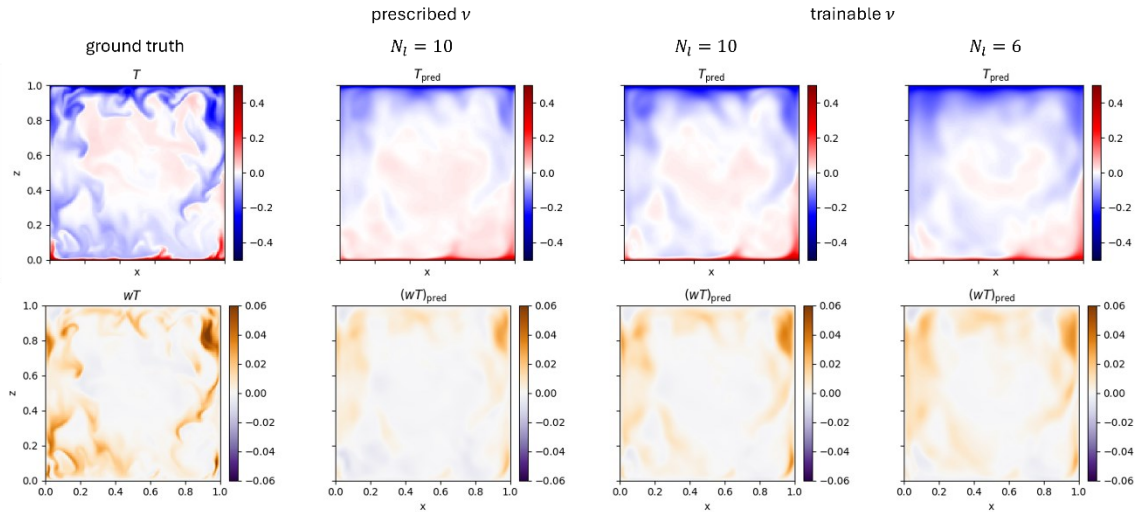


Figure 2: Comparison of the fields of the temperatures  $T$  and convective heat fluxes  $wT$  for Rayleigh-Bénard convection at  $Ra=10^8$ . From left to right: Ground truth, PINN reconstruction with prescribed viscosity (MLP,  $N_n=256$ ,  $N_l=10$ ) and trainable viscosity for networks with 10 and 6 layers ( $N_n=256$ ).

To investigate reconstruction-capabilities of the Rayleigh number, related to the trainable viscosity by  $Ra = \text{Pr}/\nu^2$ , we focus on two exemplary runs of the case described in section 2 with an MLP with  $N_n = 256$  and  $N_l = \{6; 10\}$ . Additionally, we regard a respective case with  $N_l = 10$  without a trainable viscosity.

Figure 2 shows a central section of the respectively reconstructed temperature fields against the ground truth on the left. Overall, the PINNs are not able to reconstruct the features on all scales, due to the limited provided sampling density. However, with regard to the  $N_l = 10$  cases, making  $\nu$  trainable actually improves the results from Pearson correlation coefficient of the temperature field from  $\rho_T = 0.89$  to  $\rho_T = 0.90$ . This improvement becomes especially pronounced in the field of the convective heat flux, for which the case with trainable viscosity produces a much better prediction in the region of the

bottom left corner. With a reduced number of layers, see the  $N_l = 6$  case, this effect prevails, while the deficit in small scale reconstruction increases due the reduced expressivity.

This shows that introducing  $\nu$  as a trainable parameter is beneficial, as it appears to open new training trajectories which lead to an improved reconstruction. Thereby, the predicted Rayleigh numbers of the two cases are  $3.6 \times 10^7$  and  $2.6 \times 10^7$  for  $N_l = 10$  and  $N_l = 6$ , respectively. They are robust predictions, but indicate a systematic overprediction of the viscosity due to the limitations of the sampling resolution and network expressivity.

## 5. Benchmarking

In this section, we explore the performance of the different network types with a trainable  $\nu$  on the sampling case described in section 2. For this, the metric of the Pearson correlation coefficient of the temperature field  $\rho_T$  is used. First, we consider the pure performance of the different models with respect to this metric. Second, we investigate whether the relative deviation of  $\nu$  can be used as a performance indicator when both Ra and Pr are known.

### 5.1. Model complexity

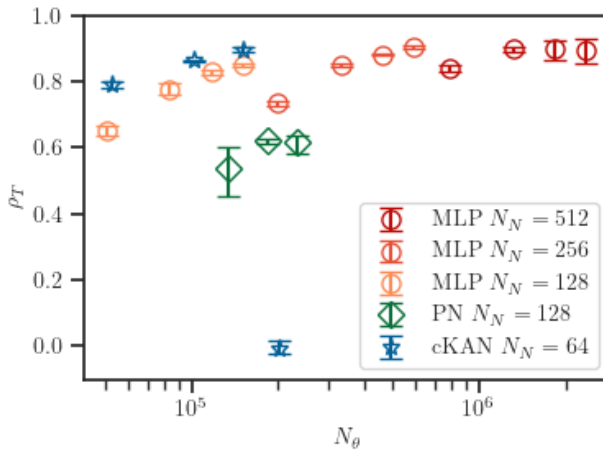


Figure 3: Pearson correlation coefficient of the temperature field for different network architectures MLP, PN, and cKAN with varying numbers of layers  $N_l$  and neurons per layer  $N_N$ , leading to different total numbers of trainable parameters  $N_\theta$ . The error bars indicate the spread between minimum and maximum temperature correlation for 3 runs with different random samplings.

To investigate the effect of model complexity on performance, the architecture of the MLPs was varied in terms of the number of layers,  $N_l = \{4; 6; 8; 10\}$ , and neurons per layer,  $N_n = \{128; 256; 512\}$ . For the PN, the number of blocks,  $N_b = \{2; 3; 4\}$ , was varied for a prescribed layer width of  $N_n = 128$ . For the cKAN, the layer width was set to  $N_n = 64$  with  $N_l = \{4; 6; 8; 10\}$  as depths.

Figure 3 shows the results of these benchmarks plotted over the total number of trainable parameters  $N_\theta$ . It reveals a trend for all network types, that increasing depth improves the results, however with diminishing returns. For the MLPs, figure 3 further shows that, for a similar total trainable parameter count, the option with more but smaller layers appears to be more efficient.

Comparing the different network types with each other renders the cKAN as the most efficient architecture considering total parameter count, equalling the performance of significantly larger MLPs. The here used PNs lack in performance, which contradicts other investigations [10], and poses the question whether this is related to the specific use case or the modifications applied here.

Another overall trend is that very large, and especially deep, networks become overall harder to train. This shows in an increased variation of the results of the largest MLP, a small decline in the mean performance of the largest PN, and a complete loss of trainability for the largest cKAN.

While the number of trainable parameters is relevant for the memory requirements, figure 4 shows the performance indicator  $\rho_T$  of the different networks plotted over the expended wall time of the training on an Nvidia A100 GPU. In this regard, the multiplicative operations within PNs are especially

expensive, which leads to them having the largest wall time requirements for training. The backpropagation for cKANs is also significantly less efficient compared to the one of MLPs, as their training requires wall-times similar to MLPs with widths of 256 or 512, while having the parameter count equivalent to MLPs with a width of  $N_n = 128$ . Hence, within this comparison, MLPs are the most efficient option regarding the required wall time and therefore also energy. Within the MLP category, narrower yet deeper networks generally yield better results for a given wall time.

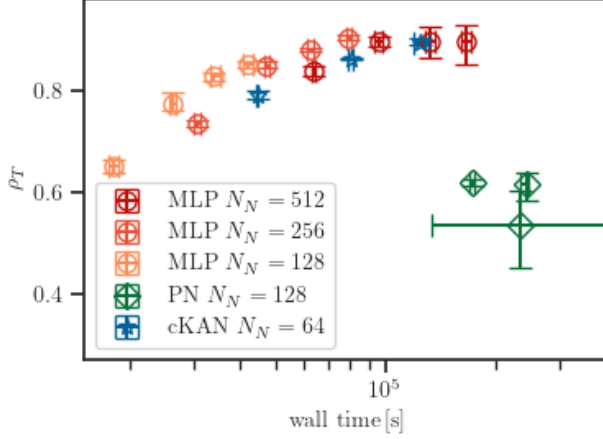


Figure 4: Pearson correlation coefficient of the temperature field for different network architectures MLP, PN, and cKAN with varying numbers of trainable parameters plotted over the wall time expended for  $10^6$  training steps (data point of one color vary in network depth). The error bars indicate the spread between minimum and maximum temperature correlation for 3 runs with different random samplings.

### 5.2. Trainable viscosity as performance indicator

For the cases considered in the previous section, we now also consider their trained viscosity value, in particular its relative deviation from the ground truth  $\Delta\nu/\nu$ .

Figure 5 exhibits its relation to the Pearson correlation metric of the temperature reconstruction. It shows that there is a strong negative correlation between the two metrics for each individual network type. While the point clouds for MLPs and cKANs even overlap, the PN cases form a separate point cloud. This shows that the deviations of a trainable viscosity can be used as performance indicators, as long as no major changes on the network architecture occur. This can be explained by a better prediction of the viscosity requiring the network to render as many scales of the flow as possible correctly, which is also beneficial for the temperature reconstruction performance.

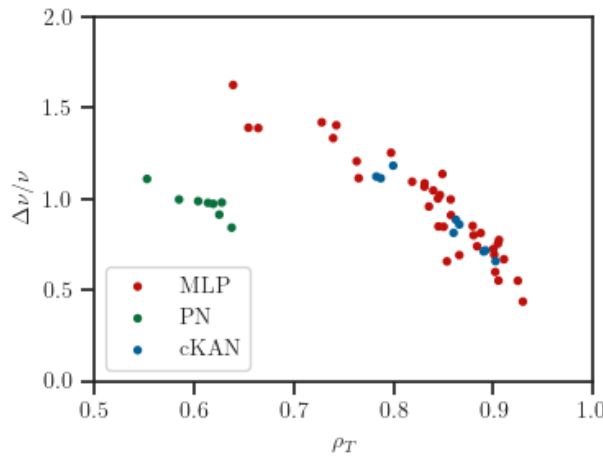


Figure 5: Relative deviation of the trainable viscosity of the cases (single dots for every of the three random samplings) from section 5.1 plotted over their Pearson correlation coefficient for the temperature field. The different colors indicate the network type.

## 6. Conclusion and Outlook

We have shown that allowing PINNs to train the dimensionless viscosity improves temperature reconstruction results and yields robust Rayleigh number estimates. The subsequent benchmarking of sine-activated multi-layered perceptrons, modified Pirate-Nets and Chebychev-Kolmogorov-Arnold networks revealed that, of the three, multi-layered perceptrons and Kolmogorov-Arnold networks are the most efficient for the task at hand. The former is especially efficient regarding wall-time, while the latter is especially efficient when it comes to parameter count and therefore memory usage.

It was also found that the estimation of the viscosity through training can itself function as performance indicator when the Rayleigh and Prandtl numbers are known.

Future research directions include expanding to different temporal and spatial sampling densities, as well as incorporating additional network architectures.

## 7. Acknowledgments

The authors gratefully acknowledge the scientific support and HPC resources provided by the German Aerospace Center (DLR). The HPC system CARA is partially funded by "Saxon State Ministry for Economic Affairs, Labour, Energy and Climate Action" and "Federal Ministry of Research, Technology and Space".

M.M. gratefully acknowledges the financial support of the German Research Foundation (DFG) under grant MO 5240/1-1.

## 8. References

- [1] M. Raissi, P. Perdikaris, G.E. Karniadakis 'Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations', J Comput Phys, <https://doi.org/10.1016/j.jcp.2018.10.045>
- [2] S. Cai, Z. Mao, Z. Wang, M. Yin, G.E. Karniadakis 'Physics-informed neural networks (PINNs) for fluid mechanics: a review', Acta Mech Sin, <https://doi.org/10.1007/s10409-021-01148-1>
- [3] D. Lucor, A. Agrawal, A. Sergent 'Simple computational strategies for more effective physics-informed neural networks modeling of turbulent natural convection', J Comput Phys, <https://doi.org/10.1016/j.jcp.2022.111022>
- [4] M. Mommert, R. Barta, C. Bauer, M.-C. Volk, C. Wagner 'Periodically activated physics-informed neural networks for assimilation tasks for three-dimensional Rayleigh-Bénard convection', Comput & Fluids, <https://doi.org/10.1016/j.compfluid.2024.106419>
- [5] M.-C. Volk, A. Sergent, D. Lucor, M. Mommert, C. Bauer, C. Wagner 'A PINN methodology for temperature field reconstruction in the PIV measurement plane: Case of Rayleigh-Bénard convection', Int Comm Heat Mass Transf, <https://doi.org/10.1016/j.icheatmasstransfer.2025.109284>
- [6] J.D. Toscano, T. Käufer, Z. Wang, M. Maxey, C. Cierpka, G.E. Karniadakis 'AIVT: Inference of turbulent thermal convection from measured 3D velocity data by physics-informed Kolmogorov-Arnold networks', Sci Adv, <https://doi.org/10.1126/sciadv.ads5236>
- [7] R. Barta, M.-C. Volk, C. Bauer, C. Wagner, M. Mommert 'Temperature and pressure reconstruction in turbulent Rayleigh-Bénard convection by Lagrangian velocities using PINN', Meas Sci Technol, <https://doi.org/10.1088/1361-6501/adee38>
- [8] R.L. Cook 'Stochastic sampling in computer graphics', ACM Trans Graph, <https://doi.org/10.1145/7529.8927>
- [9] V. Sitzmann, J. Martel, A. Bergmann, D. Lindell, G. Wetzstein 'Implicit Neural Representations with Periodic Activation Functions', NeurIPS 2020, Online, 6-12 December 2020
- [10] S. Wang, B. Li, Y. Chen, P. Perdikaris 'PirateNets: Physics-informed Deep Learning with Residual Adaptive Networks', J Mach Learn Res, <http://jmlr.org/papers/v25/24-0313.html>