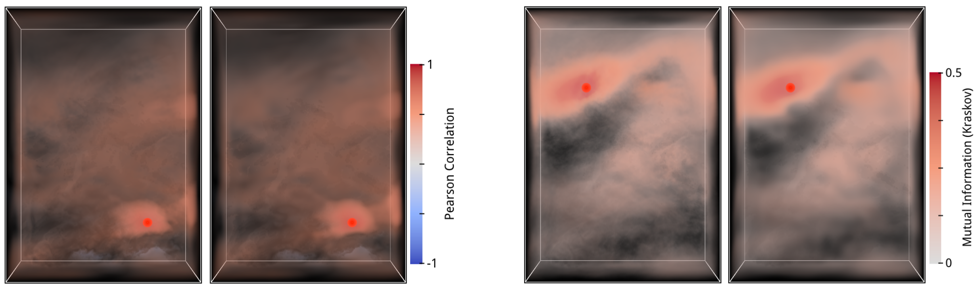
Vision, Modeling, and Visualization (2023)
T. Grosch and M. Guthe (Eds.)

Neural Fields for Interactive Visualization of Statistical Dependencies in 3D Simulation Ensembles

F. Farokhmanesh[1], K. Höhlein[1], C. Neuhauser[1] and R. Westermann[1]

[1]Technical University of Munich, Department of Computer Science, School of Computation, Information and Technology, Germany



(a) *Ground truth* **(b)** *NDF reconstruction* **(c)** *Ground truth* **(d)** *NDF reconstruction*

Figure 1: *Neural dependence fields have learned to infer 1500 billion point-to-point Pearson correlation (left) or mutual information estimates (right) in a 1000-member simulation ensemble. Inference of the dependencies between data values at an arbitrary grid vertex (red dot) to all other vertices in a $250 \times 352 \times 20$ grid takes 9 ms on a high-end GPU. Ground truth volume renderings and network results, respectively, are shown in Figures 1a, 1c and 1b, 1d. The network requires only 1 GB at runtime.*

Abstract

We present neural dependence fields (NDFs) – the first neural network that learns to compactly represent and efficiently reconstruct the statistical dependencies between the values of physical variables at different spatial locations in large 3D simulation ensembles. Going beyond linear dependencies, we consider mutual information as an exemplary measure of non-linear dependence. We demonstrate learning and reconstruction with a large weather forecast ensemble comprising 1000 members, each storing multiple physical variables at a $250 \times 352 \times 20$ simulation grid. By circumventing compute-intensive statistical estimators at runtime, we demonstrate significantly reduced memory and computation requirements for reconstructing the major dependence structures. This enables embedding the estimator into a GPU-accelerated direct volume renderer and interactively visualizing all mutual dependencies for a selected domain point.

CCS Concepts

• Computing methodologies $\rightarrow$ ***Neural networks; Computer graphics; • Applied computing*** $\rightarrow$ ***Earth and atmospheric sciences;***

1. Introduction

Estimating statistical dependencies between physical variables at different spatial locations is crucial for understanding physical systems in various scientific and engineering fields. An important application lies in meteorology, where accurate weather forecasting relies on extensive numerical simulations. Weather forecasts need to account for randomness, and ensembles of simulations with varying initial conditions and model specifications are used to quantify uncertainty. It is essential to analyze statistical relations, such as spatio-temporal auto-correlations within forecast fields or correlations between different forecast variables, to translate volumetric ensemble fields into reliable forecasts. Studying statistical dependence between random variables is well-researched, and measures exist for assessing linear and non-linear relationships. However, determining relations in 3D ensemble fields presents challenges. Computing dependencies on the fly may be computationally costly, and storing all point-to-point correlations leads to an explosion in required memory. For example, in a simulation ensemble with 1000 members on a $250 \times 352 \times 20$ grid, storing

DOI: 10.2312/vmv.20231229

all correlations would need over 3 terabytes of memory, making it infeasible. Additionally, computing correlations between arbitrary point pairs on the fly requires the entire ensemble to fit into the working memory, and more complex measures like mutual information (MI) take roughly 16 minutes on a recent multi-core CPU, hindering interactive analysis of correlation structures.

In this work, we address these challenges by introducing neural dependence fields (NDFs), a novel compact representation of the major correlation structures in large multi-variable ensembles. For this, we propose to interpret fields of two-point correlation measures in 3D ensembles as scalar fields R over the domain of position pairs in 3D space, i.e., $R : \Omega \times \Omega \to \mathbb{R}$, for $\Omega \subset \mathbb{R}^3$. Taking inspiration from recent progress in neural scene representations and multi-dimensional tensor decomposition, we design a neural network architecture that exploits self-similarity in the correlation fields to learn a compact representation thereof. At the same time, the network enables fast sampling out of the neural representation. Thus, we can avoid holding the ensemble in memory and are able to speed up the computation of correlation estimates significantly, especially for complex non-linear dependence measures. For the ensemble considered in this work, it takes roughly 9 ms to reconstruct dependencies between an arbitrary reference point and all other points in the domain. This allows embedding the network into an interactive volume rendering pipeline, which enables instant visualization and comparison of single-variable auto-correlation and inter-variable correlation fields. In summary, our contributions are:

- A compact neural network architecture to learn statistical point-to-point correlations in large ensemble fields.
- The embedding of neural network-based correlation reconstruction into direct volume rendering to enable interactive visual exploration of the dependencies in the 3D domain.
- A demonstration of interactive correlation analysis for a large meteorological 3D ensemble field.

The proposed method is agnostic towards the choice of correlation measure, such that both linear (e.g., Pearson correlation) and compute-intensive non-linear measures (e.g., MI) are supported. The network manages to reconstruct the major correlation structures faithfully, despite showing a tendency to smooth out fine details (cf. Fig. 1). In view of the complexity of the information to be learned, our results demonstrate the potential of network-based correlation learning and open the door for future research in this field, e.g., by looking into more powerful architectures or specialized loss functions. The code for the project is publicly available at [FH23, NS23].

2. Related work

Scene representation networks and neural fields Scene representation networks (SRNs) are neural networks trained to derive compact representations of 3D models and scenes. Originally, they were proposed for 2D or 3D position coordinate mapping. Early examples include encoding surface models as implicit functions or occupation maps using fully-connected neural networks [MON*19, CZ19, PFS*19]. Later, they evolved to encode diverse volumetric scenery information, such as neural radiance fields [TSM*20, MST*21] and were named neural fields [XTS*22]. A neural field is a neural network that learns a parametrization of spatio-temporal multi-dimensional physical fields over spatial coordinates. In inference, coordinates are transformed into latent-space representations and then decoded to obtain the physical quantity.

Recent work on neural fields has shown that domain-oriented input feature encodings can significantly boost reconstruction quality. Chabra et al. [CLI*20] proposed laying out trainable parameters in a grid of latent features to learn spatial variations more directly. Refinements include adaptive data structures [MLL*21], fixed multi-resolution grids [TLY*21], and multi-resolution spatial hashing [MESK22, MRNK21a], enabling multi-scale learning of spatial feature maps. Comprehensive reviews of SRN-related literature focusing on neural scene representations are available [HSB*20, TFT*20].

In scientific data visualization, Lu et al. [LJLB21] introduced SRNs for volumetric data compression, which was sped up and refined by Weiss et al. [WHW22] through the use of trainable feature representations in combination with an efficient GPU implementation. Höhlein et al. [HWW22] employ neural fields for compressing ensemble data by sharing model parameters between different ensemble members. Both works demonstrate the combination of volume rendering and network inference as used in this study.

Correlation visualization Volume rendering was chosen to demonstrate network-based reconstruction for correlation visualization in interactive workflows. However, alternative techniques for correlation visualization have been proposed, including clustering [PW12, LWS18, EHL21], correlation matrices [CWMW11, EHL21], diagram views [STS06, BDSW13, ZMZM14, LS16], and specific feature-based approaches like analyzing dependencies in flow fields with particle trajectories [BMLC19]. Correlation subsampling [GW10, CWMW11] identifies prominent features in correlation fields for analysis. These approaches complement our contribution, as they address significant structures in correlation fields or develop effective visual encodings. Our approach seamlessly integrates with these techniques, reducing memory and computation requirements for accessing correlations in large 3D fields.

3. Statistical dependence in ensemble fields

We quantify statistical dependencies in ensemble datasets through bivariate correlation measures $\rho : \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}$ between vectors of paired random samples. For this, let $\Omega \subset \mathbb{R}^3$ be a simulation domain in 3D space, and let $\mathcal{E} = \{E_i : 0 \leq i < N\}$ be an ensemble of N multi-variable fields $E_i : \Omega \to \mathbb{R}^d$. The index i suggests a fixed but arbitrary enumeration of the members. For all i and $0 \leq \nu < d$, let $E_i^\nu : \Omega \to \mathbb{R}$ denote the scalar field associated with variable ν in member E_i. For a given position $\mathbf{p} \in \Omega$, we refer to the local sample of variable values as $\mathbf{e}^\nu(\mathbf{p}) := (E_i^\nu(\mathbf{p}) : 0 \leq i \leq N) \in \mathbb{R}^N$. Then, for variables μ and ν, we consider the field of μ-ν-correlations, $R_{\mu\nu} : \Omega \times \Omega \to \mathbb{R}$, where for all pairs of positions $(\mathbf{p}_\mu, \mathbf{p}_\nu) \in \Omega^2$ the field value is defined as $R_{\mu\nu}(\mathbf{p}_\mu, \mathbf{p}_\nu) := \rho(\mathbf{e}^\mu(\mathbf{p}_\mu), \mathbf{e}^\nu(\mathbf{p}_\nu))$. Note that the indexing of the position variables is used to imply that position $\mathbf{p}_\mu$ (position $\mathbf{p}_\nu$) alters the value of $R_{\mu\nu}(\cdot,\cdot)$ by changing the reference position in field μ (field ν), respectively. Special attention is payed to the case $\mu = \nu$, which we refer to as the μ-self-correlation field

$S_\mu := R_{\mu\mu}$. Notably, S_μ is symmetric under exchange of position coordinates, i.e., $S_\mu(\mathbf{p}_1, \mathbf{p}_2) = S_\mu(\mathbf{p}_2, \mathbf{p}_1)$ for all $\mathbf{p}_1, \mathbf{p}_2 \in \Omega$.

This work uses Pearson correlation and MI from the wide range of possible dependence measures. Both represent opposite sides of the spectrum of computational cost and indicate different kinds of dependence [BMLC19].

3.1. Pearson product-moment correlation coefficient

The Pearson correlation coefficient, or Pearson's r, measures the linear correlation between random variable pairs. It is commonly used in data visualization to explore relationships between variables. Given a set of paired random samples, $\mathbf{e}_1, \mathbf{e}_2 \in \mathbb{R}^N$, the Pearson correlation coefficient is defined as

$$r = \frac{\mathrm{cov}(\mathbf{e}_1, \mathbf{e}_2)}{\sqrt{\mathrm{var}(\mathbf{e}_1)\,\mathrm{var}(\mathbf{e}_2)}}, \tag{1}$$

wherein var$(\cdot)$ and cov$(\cdot)$ denote sample variance and covariance of the respective random samples. With a range of -1 to 1, Pearson correlation indicates correlation (+1), anti-correlation (-1), or the absence of correlation (0). It is easy to interpret, quantifying the strength and direction of the relationship between variables. However, caution is needed when the relationship is nonlinear or when outliers are present, as they can significantly affect the correlation coefficient.

3.2. Mutual information

MI is widely used in machine learning, statistics, and information theory [CT*91] to measure similarity or correlation between random variable pairs. Unlike linear correlation, MI can detect nonlinear and non-monotonic dependencies that are not evident in covariance. Mathematically, this is expressed as

$$I(X_1;X_2) = H(X_1) - H(X_1|X_2) = H(X_2) - H(X_2|X_1), \tag{2}$$

wherein X_1 and X_2 are random variables, $I(X_1;X_2)$ is the MI of X_1 and X_2, $H(X_1)$ is the entropy of X_1, and $H(X_1|X_2)$ is the conditional entropy of X_1 given X_2. Note that MI is symmetric under the exchange of X_1 and X_2.

Estimating MI from finite samples $\mathbf{e}_1$ and $\mathbf{e}_2$ of random variables X_1 and X_2, i.e. computing $I(\mathbf{e}_1, \mathbf{e}_2)$, is computationally expensive. Existing algorithms struggle to scale with large sample sizes [KSG04, MRL95]. More recent copula-based and neural-network-based variational MI estimators offer better performance in high-dimensional data spaces but still pose computational challenges [ZD11, BBR*18]. In our study, we compute ground truth MI fields using the nearest-neighbor-based estimator of Kraskov et al. [KSG04], implemented in parallel. However, detailed performance analysis in section 6 shows that estimating MI fields for visualization exceeds the time constraints of interactive data analysis. Mutual information neural estimation (MINE) is a recent method that uses neural networks to estimate MI [BBR*18]. It trains a neural network to learn a lower bound on MI, which provides an estimate. MINE has shown better performance than traditional MI estimation methods on benchmark datasets. Nevertheless, in our scenario, the inefficiency arises because MINE operates on a member-wise basis and requires all members to be present in memory.

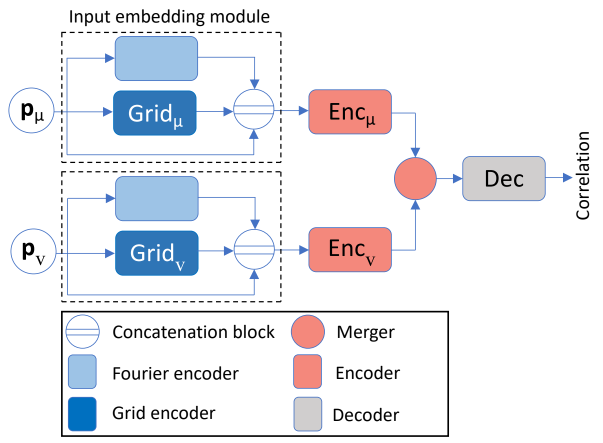


Figure 2: *NDF architecture.* $\mathbf{p}_\mu$ *and* $\mathbf{p}_\nu$ *are respectively the reference and query positions.* Grid$_\mu$ *and* Grid$_\nu$*, respectively, are the hash grids. Variable-specific encoders* Enc$_\mu$ *and* Enc$_\nu$ *are MLPs. Since the feature grids involve trainable parameters, each encoder is equipped with a separate grid. Merger indicates the multiplication of the encoder outputs.*

4. Neural dependence fields

To enable the use of large sets of point-to-point dependence measures, we perform the computationally expensive calculations of these measures in a preprocess and encode the two-point μ-ν-correlation fields $R_{\mu\nu}$ (as defined in section 3) with memory- and compute-efficient neural scene representations, $\Phi_{\mu\nu} : \Omega \times \Omega \to \mathbb{R}$. The network is obtained by solving the optimization problem

$$\Phi_{\mu\nu} := \arg\min_{\Phi} \mathbb{E}_{(\mathbf{p}_1,\mathbf{p}_2)\sim\mathcal{U}(\Omega^2)} \left[d(\Phi(\mathbf{p}_1,\mathbf{p}_2), R_{\mu\nu}(\mathbf{p}_1,\mathbf{p}_2))\right], \tag{3}$$

wherein $\Phi(\cdot,\cdot)$ is the neural network, $d(\cdot,\cdot)$ is a similarity metric, such as L_1 or L_2 loss, and the expectation $\mathbb{E}[\cdot]$ is taken over samples of position pairs from a uniform distribution, $\mathcal{U}(\Omega^2)$, with support Ω^2. The optimization is carried out iteratively using stochastic gradient descent. Using tractably sized batches of position pairs simultaneously avoids storing excessive amounts of correlation samples. After training, $\Phi_{\mu\nu}$ is a compact correlation field encoding, enabling rapid sample reconstruction. Classical compression methods like TThresh and SZ cannot achieve similar compaction due to fixed discretization and computational infeasibility. Additionally, once the correlation network is trained, there is no need to keep the entire ensemble dataset in memory, allowing the approach to scale efficiently to large ensemble sizes.

4.1. Network architecture

NDFs differ from classical neural scene representations as they operate on a bi-spatial domain with six dimensions (6D). Due to the curse of dimensionality, training efficiency is lowered by the additional dimensions since covering the domain adequately with samples becomes exponentially more complex. To overcome this, we construct NDFs to utilize sparse sampling information efficiently, which improves memory efficiency and avoids the computation of excessive amounts of correlation samples.

As shown in Figure 2, we propose a bipartite network architecture, which consists of two variable-specific encoder networks, Enc_μ, and Enc_ν, along with a shared decoder network. All networks are implemented as multi-layer perceptrons (MLPs) with l fully-connected layers and c hidden channels using *SnakeAlt* activation [WHW22]. Each encoder model receives information about one of the two positions between which the correlation should be reconstructed. Positions are translated into a latent feature vector, which is merged via element-wise multiplication and forwarded to the decoder for the final prediction. This architecture allows each encoder to be trained on only the marginal space $\Omega \subset \mathbb{R}^3$, which improves the training efficiency by increasing the effective amount of correlation samples per volume. In combination with the spatial coherence of the ensemble fields, this enables the model to infer correlations even for point pairs that were not seen during training, thus saving computation time and memory requirements.

The decomposition is similar in spirit to the approach used in TensoRF [CXG*22] or K-planes [FKMW*23], where 3D feature tensors are decomposed into linear combinations of tensor products between lower-dimensional feature vectors and matrices for higher parameter efficiency. In the proposed NDF, features over a 6D domain are decomposed into an outer product of fields over a 3D domain. The accuracy of predictions relies heavily on using multiplication for feature merging. Alternative combination methods, such as concatenation, addition or absolute difference, result in a substantial drop in prediction fidelity, which is in line with findings in [FKMW*23]. Deviating from TensoRF and K-planes, we found applying MLPs before and after feature merging beneficial, which we validate in more detail in section 6.

For self-correlation fields S_μ (as defined in section 3) and the corresponding NDFs $\Phi_{\mu\mu}$, we further constrain the architecture to use identical encoders for both positions, i.e., $\mathrm{Enc}_\mu = \mathrm{Enc}_\nu$ (and $\mathrm{Grid}_\mu = \mathrm{Grid}_\nu$, see below for details). This helps to keep the models small and ensures symmetry of the learned fields under exchange of the query positions on an architectural level, i.e., $\Phi_{\mu\mu}(\mathbf{p}_1, \mathbf{p}_2) = \Phi_{\mu\mu}(\mathbf{p}_2, \mathbf{p}_1)$ is fulfilled trivially by design and does not need to be learned in expensive training iterations. To improve the capability of the encoders to learn high-frequency patterns as well as spatially distributed and multi-scale features, we employ Fourier features [MST*21, TSM*20] as well as multi-resolution hash-grids [MESK22] on the position coordinates, which are concatenated to the raw positions before being processed by the encoders. The input embedding modules are marked with dashed rectangles in Figure 2.

4.2. Input embedding

For a given vector of input coordinates, $\mathbf{p} = (p_x, p_y, p_z) \in \mathbb{R}^3$, Fourier features increase the spread between spatially close positions by embedding the position information into a higher-dimensional space using the fixed feature mapping

$$\mathbf{f}_{ij} = (\sin(\omega_i \mathbf{n}_j \cdot \mathbf{p}), \cos(\omega_i \mathbf{n}_j \cdot \mathbf{p})), \tag{4}$$

wherein $\omega_i = 2^i\pi$ for $0 \leq i < L \in \mathbb{N}$, and $\mathbf{n}_j \in \mathbb{R}^3$ are the axis-aligned unit vectors for $j \in \{x, y, z\}$. With Fourier features, the model can better resolve high-frequent patterns while not affecting the number of trainable parameters (and thus memory consumption) due to the fixed functional form of the mapping. In our implementation, we empirically determined $L = 12$ as a good choice for the number of Fourier frequencies.

Multi-resolution hash grids use hash tables filled with trainable feature vectors to populate the domain at various resolutions [MESK22]. By hashing 3D grid indices, vectors are assigned to regular grid positions at multiple scales, enabling retrieval of feature vectors at arbitrary positions through tri-linear interpolation. This allows the creation of virtual feature grids at any resolution with a fixed memory budget.

The feature vectors for different resolution levels are concatenated and trained jointly with subsequent model parts using stochastic gradient descent. Hash collisions equilibrate during training due to the pseudo-random hash mapping and multiple resolutions, ensuring adaptive and local feature capacity distribution. The critical parameter for the expressiveness of the hash grid is the hash table size, 2^T for $T \in \mathbb{N}$, which also determines memory complexity. Other hyper-parameters include the dimension of the feature vectors per resolution level, the number of resolution levels, and the grid resolution on each level. We use 6 resolution levels with virtual grids of size 16^3 on the coarsest level, doubling with each finer level. These parameters ensure that the feature granularity matches the spatial resolution of the original dataset, avoiding higher memory consumption with finer levels while maintaining or improving the reconstruction accuracy.

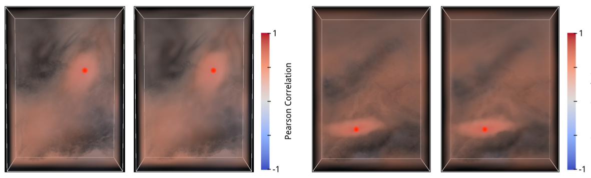
4.3. Training

During training, we use a rectangular simulation domain Ω with data samples on a regular grid, rescaled to fill the symmetric unit cube, i.e., $\Omega = [-1, 1]^3$. We generate 10^6 pairs of uniformly distributed random positions $(\mathbf{p}_\mu, \mathbf{p}_\nu)$ in $[-1, 1]^3$, retrieve samples $\mathbf{e}^\mu(\mathbf{p}_\mu)$ and $\mathbf{e}^\nu(\mathbf{p}_\nu)$ using trilinear interpolation in the original ensemble dataset, and compute correlations $R_{\mu\nu}(\mathbf{p}_\mu, \mathbf{p}_\nu)$. The models are trained to optimize the L_1 loss as a similarity measure, with L_2 loss yielding similar results in our experiments. We use the Adam optimizer [KB14] with an initial learning rate of 3×10^{-4} and 1000 samples per batch. An adaptive learning rate scheduler reduces the learning rate by a factor of 0.1 after 5 passes without improvement in reconstruction accuracy. After every epoch, the training samples are renewed. The total training duration is 200 epochs.

5. Correlation visualization

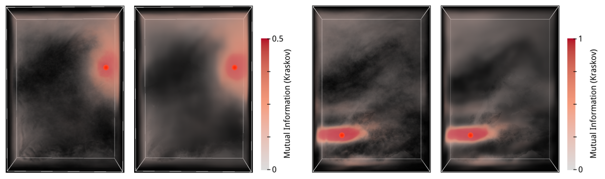
Once trained, the network can estimate dependencies for any position pair. Multiple queries can be batched efficiently for parallel processing on a GPU using the tiny-cuda-nn framework [Mü21], which features a fully-fused MLP implementation [MRNK21b] with fast 16-bit inference using tensor cores on NVIDIA GPUs and provides functionality for the multi-resolution hashed feature grids [MESK22]. The custom activation functions *Snake* and *SnakeAlt* [WHW22] were added to a fork of the library to support the model architecture, as simpler activation functions like ReLU led to inferior reconstruction accuracy.

Network training is performed in PyTorch using the Python bindings of the library. To enable interactive visualizations, the binary



(a) $\mu = \text{tk}$, $PSNR = 87.63$ **(b)** $\mu = \text{u}$, $PSNR = 86.39$

Figure 3: *Point-to-point Pearson self-correlations S_μ for different variables μ (temperature* tk *and longitudinal component* u *of wind speed) and reference positions $\mathbf{p}_{ref}$. Figures show ground truth (left) and NDF reconstruction (right). The reference is shown in red.*



(a) $\mu = \text{tk}$, $PSNR = 86.44$ **(b)** $\mu = \text{u}$, $PSNR = 85.27$

Figure 4: *Point-to-point MI self-correlations S_μ for different variables μ (temperature* tk *and longitudinal component* u *of wind speed) and reference positions $\mathbf{p}_{ref}$. Figures show ground truth (left) and NDF reconstruction (right). The reference is shown in red.*

weights of the MLP encoder and decoder can then be loaded by a tiny-cuda-nn module into whatever GPU correlation visualization is used. In our primary use case, we access the network from a GPU-based volume renderer implemented in Vulkan [The23]. The renderer is tied to a graphical interface, in which the user is able to select reference points $\mathbf{p}_{\text{ref}} \in [-1,1]^3$, for which correlation samples are reconstructed and displayed as a density field. Specifically, we consider the case of displaying volumetric correlation fields, $\phi : [-1,1]^3 \to \mathbb{R}$, where $\phi(\mathbf{p}) := \Phi_{\mu\nu}(\mathbf{p}, \mathbf{p}_{\text{ref}})$ or $\Phi_{\mu\nu}(\mathbf{p}_{\text{ref}}, \mathbf{p})$.

For visualization, the correlation fields are sampled on a grid with resolution $X \times Y \times Z$. A CUDA input buffer is prepared for the reference point $\mathbf{p}_{\text{ref}}$ and passed to the tiny-cuda-nn encoder module to get the encoded reference vector. The grid is divided into $\lceil (X \times Y \times Z)/M \rceil$ query batches of size M. Each batch is encoded, and the reference and query features are multiplied before being passed to the tiny-cuda-nn decoder module to obtain correlation estimates in an output buffer shared between Vulkan and CUDA, preventing race conditions with shared semaphores. Finally, the shared output buffer is copied to a 3D Vulkan image for visualization in the volume renderer.

6. Performance and Quality Analysis

Here, we showcase the NDF model for interactive visual analysis of spatial statistical dependencies in a large weather forecast ensemble. We examine the network's reconstruction speed and memory requirements and compare the results to ground truth dependence fields obtained using Pearson correlation coefficients and MI on GPU and CPU.

6.1. Dataset

We validate our approach using a convective-scale multi-variable ensemble dataset (CSEns) by Necker et al. [NGW*20]. It consists of 1000 numerical simulations of a 3D atmospheric dynamics model over a rectangular region in central Europe, with a grid size of 250×352 nodes and 20 discrete height levels. The simulations span six hours, and we select the last time step with the most interesting features for visualization [HWW22]. The dataset includes 3D data for nine meteorological variables. For validation, we compute ground truth correlation fields for temperature (tk) and longitudinal wind (u) using all 1000 ensemble members. The generalization of our approach to multiple time steps and inter-temporal dependencies will be explored in future work.

6.2. NDF model performance

To shed light on the performance of the NDFs, we conducted a comparative analysis of the runtime of NDFs against reference implementations of Pearson correlation and MI. The implementation of the MI estimator follows Kraskov et al. [KSG04]. All performance measurements are based on the CSEns dataset and were performed on an NVIDIA RTX 3090 GPU and a 6-core Intel Xeon W-2235 CPU, respectively. Notably, a parallel MI estimator is only available on the CPU, whereas we restricted ourselves to a GPU implementation of NDF-based reconstruction. Table 1 shows that our model is about 26x faster than the GPU implementation of the Pearson correlation coefficient. Compared to the CPU implementation of the MI estimator, the factor is 114,106×. Once the NDF model is trained, it takes 9 ms to reconstruct the dependencies between the data values at a selected grid point and all other grid points. The network model requires roughly 1 GB of memory while keeping the entire dataset in memory for one variable would amount to 7 GB. Training of the NDF takes approximately one hour on the aforementioned machine. Even though it is difficult to estimate the performance of a GPU implementation of the MI estimator, it can be assumed that even an optimized GPU-accelerated estimator will be significantly slower than the NDF model. The MI estimator must construct search structures for nearest-neighbour queries for all requested samples, a much more elaborate process than passing data through fully-fused MLP kernels.

6.3. NDF model accuracy

Figures 1, 3, and 4 visually compare the network's ability to reconstruct major dependence structures in the variable fields. While fine

Table 1: *Performance comparison. For a selected grid point, the dependence measures are computed for all other grid points in a $250 \times 352 \times 20$ simulation grid using a single variable. A GPU implementation of the MI estimator is not available.*

	NDF (ours)	**Pearson**	**MI [KSG04]**
CPU	–	4772 ms	1026957 ms
GPU	9 ms	234 ms	–

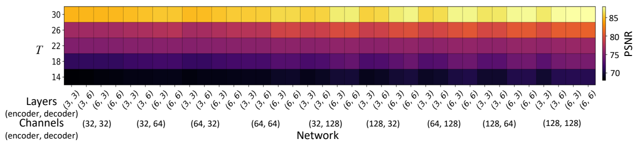
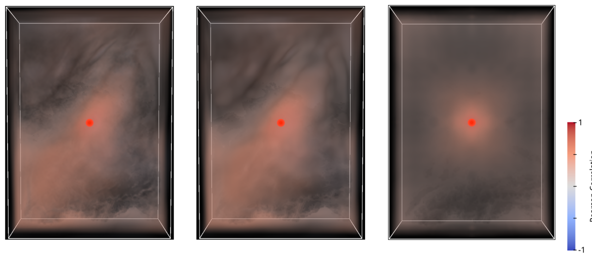


Figure 5: *Impact of hash table size and MLP hyperparameters on NDFs' reconstruction quality. The horizontal axis displays various encoder and decoder configurations, including the number of layers and hidden channels. The vertical axis shows the log-2 hash table size T from section 4.1. The plot considers variable temperature* (tk) *and Pearson correlation, with similar behavior observed for other variables and similarity metrics.*

details may not be equally well reproduced due to the limited number of correlation samples during model training, the proposed bipartite NDF architecture uses sample information efficiently by increasing the effective sampling density for the encoder parts of the model (see section 4.1). Furthermore, experiments with increased sample counts did not significantly improve reconstruction, indicating that sampling density is not the limiting factor for reconstruction accuracy. A trade-off between model capacity (memory size) and reconstruction quality is observed, suggesting that the quality is primarily bounded by the model's ability to store training information rather than sample availability. The figures displayed dependence fields for selected reference points, representing 3D slices in a much larger 6D correlation space where the model was trained. Storing all possible two-point correlations in 32-bit floating point format would require 6 TB of memory space for the CSEns grid of size $250 \times 352 \times 20$. The resulting network size of 1 GB corresponds to an effective compression factor of over 6,000×, showing promising results.

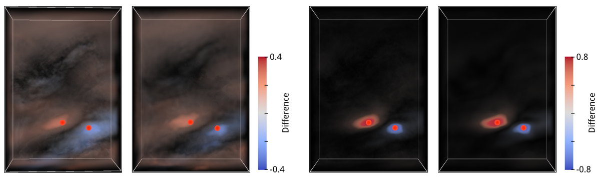
Figure 5 shows the trade-off between model size and reconstruction quality. NDFs are trained with various settings for the hash-table size 2^T and different complexities of encoder and decoder MLPs, using Pearson self-correlation fields of variable *tk*. PSNR values are computed on a set of 10^6 position pairs, sampled uniformly from the grid domain. Models are trained for a shorter duration of 50 epochs for efficiency. The figure indicates a minor advantage for models with more complex encoder and decoder MLPs. However, the most significant factor affecting achievable PSNR is the hash table size, which also strongly affects model memory consumption. Increasing the table size leads to higher PSNR values. A doubling of the table size 2^T roughly doubles the model memory requirements, while the volume of the MLP parameterization is limited to only a few kB, making it negligible. In our implementation, we set the maximum table size to less than 2^{32}. For further experiments, we choose $T = 30$ with 6-layer encoders and decoders, each having 128 channels per layer.

To validate our design choices against TensoRF [CXG*22] and K-planes [FKMW*23], we compare the reconstruction quality of the proposed architecture against a TensoRF-like model, where the MLP in the encoder part is omitted, predicting based solely on the outputs of the input embedding. Figure 6 displays reconstructed Pearson correlation fields for both approaches, highlighting the significant added value of using the MLP before feature merging.



(a) *Ground truth* **(b)** *NDF (ours)* **(c)** *NDF, grid only*

Figure 6: *Accuracy comparison between NDFs with MLP encoder (complete) and pure grid model (without encoder) using Pearson self-correlation fields for variable temperature* (tk)*.*



(a) $\mu = \mathrm{u}$, *PSNR* $= 82.77$ **(b)** $\mu = \mathrm{u}$, *PSNR* $= 83.91$

Figure 7: *Difference field visualization for multiple points in longitudinal component* u *of wind speed. a) Pearson correlation field for two different points, ground truth (left), model reconstruction (right). b) MI for two different points, ground truth (left), and model reconstruction (right).*

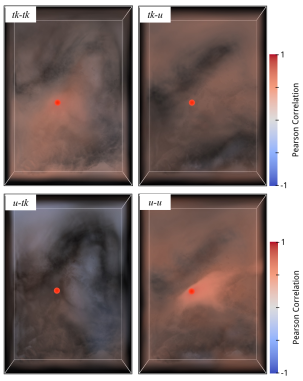
6.4. Additional experiments

The following are quantitative results using NDFs for reconstructing Pearson correlation coefficients and MI values. We begin with experiments on variable self-correlation fields S_μ and their corresponding networks $\Phi_{\mu\mu}$.

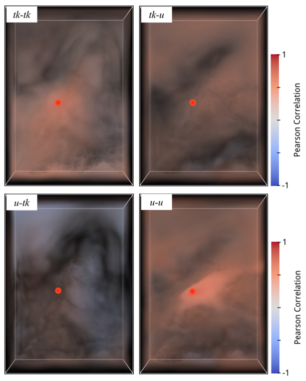
Single-point experiment Firstly, the user selects a point in the 3D domain, and the dependence field $\phi(\mathbf{r})$ is instantly displayed via volume rendering (see supplementary video). Volumetric visualizations, including reconstruction and rendering, can be generated below 10 ms. Moving the reference point interactively highlights different regions with high correlation. This allows the user to identify areas of high internal correlation or observe how correlations change with distance from the reference. The transfer function can be adjusted interactively for better visibility of specific features during the analysis.

Multi-point experiment Secondly, the user selects two points in the domain, and the difference between the reconstructed dependence fields for each point is visualized (Figure 7a for Pearson correlation and Figure 7b for MI). The images show the correlation decay around the points and reveal additional structures in other areas within the fields. This allows users to efficiently analyze the differences in the dependence structures related to different points in the 3D domain.

Multi-variable experiment Our last experiment demonstrates the use of NDFs for analyzing spatial dependencies between different variables. For a set of $d \in \mathbb{N}$ variable fields, i.e., $V = \{v_1, v_2, ..., v_d\}$, this requires training of $d(d+1)/2$ NDFs $\Phi_{v_i v_j}$,



(a) *Ground truth*



(b) *NDF reconstruction*

Figure 8: *Visualizing dependencies between variables temperature (*tk*) and longitudinal component* u *of wind speed using Pearson correlation coefficients. a) Ground truth, b) NDF model reconstruction. Fields show correlations between variables* tk *or* u *at selected points to the same or different variable at other points.*

for each of the combinations $\nu_i, \nu_j \in V$ with $1 \leq i \leq j \leq d$. For a pair of two different physical variables, such as $\nu_1 = tk$ and $\nu_2 = u$, this amounts to training three NDFs, $\Phi_{tk,tk}$, $\Phi_{u,u}$ and $\Phi_{tk,u}$, which emulate the corresponding correlation fields. Note that no separate model is required for $\Phi_{u,tk}$ if the underlying correlation measure ρ (see section 3) is symmetric under exchange of arguments, since due to the symmetric architecture of NDFs $\Phi_{u,tk}(\mathbf{p}_1, \mathbf{p}_2) = \Phi_{tk,u}(\mathbf{p}_2, \mathbf{p}_1)$ for all $\mathbf{p}_1, \mathbf{p}_2 \in \Omega$.

Elaborating on the example of *tk* and *u*, we propose a matrix-like arrangement of linked volumetric correlation visualizations in the spirit of standard correlation matrix visualizations, i.e., correlation volume matrices. For this, a single reference point is selected, and correlation fields with respect to this point are rendered for all NDF configurations, i.e., all combinations of variables. Figure 8 shows an example of this.

NDFs enable easier visualization with multiple variables as they reduce computation time and memory usage compared to on-the-fly computations using raw data. The standard method would require all variables' data in GPU memory, and even high-end GPUs with 24 GB of memory can only load two variable fields simultaneously in practice. NDFs of 1 GB each allow networks for up to four variables to fit into the same memory. Exploring the reuse of variable-specific encoder grids in different networks could prevent the memory requirements of NDFs from growing quadratically with the number of variables.

7. Conclusion and Future Work

We have introduced and evaluated neural dependence fields (NDFs), a novel approach for encoding and visualizing statistical dependencies in large 3D ensemble fields. NDFs infer spatial dependencies within single variables and in pairs of different variables. They offer compact representations of linear and non-linear dependence patterns in large ensembles, facilitating the rapid reconstruction of correlation samples from the compact representation. We demonstrated interactive visual analysis of 3D dependence structures through GPU-accelerated direct volume rendering.

Our evaluations show that NDFs faithfully encode and reconstruct the prominent dependence structures in 3D fields while smoothing out some details due to limited network capacities. In the future, we aim to enhance these capacities by adding network stages in the encoder and decoder, exploring alternative architectures like diffusion networks, and trying different loss functions for preserving fine details better (e.g., gradient regularization [LJLB21]). Additionally, we plan to extend NDFs to infer temporal dependence structures in time-varying ensemble fields by decomposing the data into independent fields with spatial variation. This extension would enable new application scenarios in scientific workflows, such as ensemble sensitivity analysis [KRRW19].

Acknowledgments

The research leading to these results has been done within the subproject "A7" of the Transregional Collaborative Research Center SFB / TRR 165 "Waves to Weather" (www.wavestoweather.de) funded by the German Research Foundation (DFG).

References

[BBR*18] Belghazi M. I., Baratin A., Rajeshwar S., Ozair S., Bengio Y., Courville A., Hjelm D.: Mutual information neural estimation. In *Proceedings of the 35th International Conference on Machine Learning* (10–15 Jul 2018), Dy J., Krause A., (Eds.), vol. 80 of *Proceedings of Machine Learning Research*, PMLR, pp. 531–540. 3

[BDSW13] Biswas A., Dutta S., Shen H.-W., Woodring J.: An information-aware framework for exploring multivariate data sets. *IEEE Transactions on Visualization and Computer Graphics 19*, 12 (2013), 2683–2692. doi:10.1109/TVCG.2013.133. 2

[BMLC19] Berenjkoub M., Monico R. O., Laramee R. S., Chen G.: Visual analysis of spatia-temporal relations of pairwise attributes in unsteady flow. *IEEE Transactions on Visualization and Computer Graphics 25*, 1 (2019), 1246–1256. doi:10.1109/TVCG.2018.2864817. 2, 3

[CLI*20] Chabra R., Lenssen J. E., Ilg E., Schmidt T., Straub J., Lovegrove S., Newcombe R.: Deep local shapes: Learning local sdf priors for detailed 3d reconstruction. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIX 16* (2020), Springer, pp. 608–625. 2

[CT*91] Cover T. M., Thomas J. A., et al.: Entropy, relative entropy and mutual information. *Elements of information theory 2*, 1 (1991), 12–13. 3

[CWMW11] Chen C.-K., Wang C., Ma K.-L., Wittenberg A. T.: Static correlation visualization for large time-varying volume data. In *2011 IEEE Pacific Visualization Symposium* (2011), pp. 27–34. doi:10.1109/PACIFICVIS.2011.5742369. 2

[CXG*22] Chen A., Xu Z., Geiger A., Yu J., Su H.: Tensorf: Tensorial radiance fields. In *Computer Vision–ECCV 2022: 17th European Conference, 2022, Proceedings, Part XXXII* (2022), Springer, pp. 333–350. 4, 6

[CZ19] Chen Z., Zhang H.: Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019), pp. 5939–5948. 2

[EHL21] Evers M., Huesmann K., Linsen L.: Uncertainty-aware Visualization of Regional Time Series Correlation in Spatio-temporal Ensembles. *Computer Graphics Forum* (2021). doi:10.1111/cgf.14326. 2

[FH23] FAROKHMANESH F., HÖHLEIN K.: Neural Fields for Interactive Visualization of Statistical Dependencies in 3D Simulation Ensembles: Code for Experiments, July 2023. doi:10.5281/zenodo.8186686. 2

[FKMW*23] FRIDOVICH-KEIL S., MEANTI G., WARBURG F., RECHT B., KANAZAWA A.: K-planes: Explicit radiance fields in space, time, and appearance. *arXiv preprint arXiv:2301.10241* (2023). 4, 6

[GW10] GU Y., WANG C.: A study of hierarchical correlation clustering for scientific volume data. In *Advances in Visual Computing: 6th International Symposium, ISVC 2010, 1, 2010, Proceedings, Part III 6* (2010), Springer, pp. 437–446. 2

[HSB*20] HOANG D., SUMMA B., BHATIA H., LINDSTROM P., KLACANSKY P., USHER W., BREMER P.-T., PASCUCCI V.: Efficient and flexible hierarchical data layouts for a unified encoding of scalar field precision and resolution. *IEEE Transactions on Visualization and Computer Graphics 27*, 2 (2020), 603–613. 2

[HWW22] HÖHLEIN K., WEISS S., WESTERMANN R.: Evaluation of volume representation networks for meteorological ensemble compression. 2, 5

[KB14] KINGMA D. P., BA J.: Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014). 4

[KRRW19] KUMPF A., RAUTENHAUS M., RIEMER M., WESTERMANN R.: Visual analysis of the temporal evolution of ensemble forecast sensitivities. *IEEE Transactions on Visualization and Computer Graphics 25*, 1 (2019). doi:10.1109/TVCG.2018.2864901. 7

[KSG04] KRASKOV A., STÖGBAUER H., GRASSBERGER P.: Estimating mutual information. *Phys. Rev. E 69* (Jun 2004), 066138. doi:10.1103/PhysRevE.69.066138. 3, 5

[LJLB21] LU Y., JIANG K., LEVINE J. A., BERGER M.: Compressive neural representations of volumetric scalar fields. In *Computer Graphics Forum* (2021), vol. 40, Wiley Online Library, pp. 135–146. 2, 7

[LS16] LIU X., SHEN H.-W.: Association analysis for visual exploration of multivariate scientific data sets. *IEEE Transactions on Visualization and Computer Graphics 22*, 1 (2016), 955–964. doi:10.1109/TVCG.2015.2467431. 2

[LWS18] LIEBMANN T., WEBER G. H., SCHEUERMANN G.: Hierarchical correlation clustering in multiple 2d scalar fields. *Computer Graphics Forum 37*, 3 (2018), 1–12. doi:https://doi.org/10.1111/cgf.13396. 2

[MESK22] MÜLLER T., EVANS A., SCHIED C., KELLER A.: Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph. 41*, 4 (jul 2022). doi:10.1145/3528223.3530127. 2, 4

[MLL*21] MARTEL J. N., LINDELL D. B., LIN C. Z., CHAN E. R., MONTEIRO M., WETZSTEIN G.: Acorn: Adaptive coordinate networks for neural scene representation. *arXiv preprint arXiv:2105.02788* (2021). 2

[MON*19] MESCHEDER L., OECHSLE M., NIEMEYER M., NOWOZIN S., GEIGER A.: Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (2019), pp. 4460–4470. 2

[MRL95] MOON Y.-I., RAJAGOPALAN B., LALL U.: Estimation of mutual information using kernel density estimators. *Physical Review E 52*, 3 (1995), 2318. 3

[MRNK21a] MÜLLER T., ROUSSELLE F., NOVÁK J., KELLER A.: Real-time neural radiance caching for path tracing. *arXiv preprint arXiv:2106.12372* (2021). 2

[MRNK21b] MÜLLER T., ROUSSELLE F., NOVÁK J., KELLER A.: Real-time neural radiance caching for path tracing. *ACM Trans. Graph. 40*, 4 (jul 2021). doi:10.1145/3450626.3459812. 4

[MST*21] MILDENHALL B., SRINIVASAN P. P., TANCIK M., BARRON J. T., RAMAMOORTHI R., NG R.: Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM 65*, 1 (2021), 99–106. 2, 4

[Mü21] MÜLLER T.: tiny-cuda-nn, 4 2021. URL: https://github.com/NVlabs/tiny-cuda-nn. 4

[NGW*20] NECKER T., GEISS S., WEISSMANN M., RUIZ J., MIYOSHI T., LIEN G.-Y.: A convective-scale 1,000-member ensemble simulation and potential applications. *Quarterly Journal of the Royal Meteorological Society 146*, 728 (2020), 1423–1442. doi:https://doi.org/10.1002/qj.3744. 5

[NS23] NEUHAUSER C., STUMPFEGGER J.: chrismile/Correrender: A correlation field renderer using the Vulkan graphics API, v2023-07-29, July 2023. doi:10.5281/zenodo.8195623. 2

[PFS*19] PARK J. J., FLORENCE P., STRAUB J., NEWCOMBE R., LOVEGROVE S.: Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (2019), pp. 165–174. 2

[PW12] PFAFFELMOSER T., WESTERMANN R.: Visualization of global correlation structures in uncertain 2d scalar fields. In *Computer Graphics Forum* (2012), vol. 31, Wiley Online Library, pp. 1025–1034. 2

[STS06] SAUBER N., THEISEL H., SEIDEL H.-P.: Multifield-graphs: An approach to visualizing correlations in multifield scalar data. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (2006), 917–924. doi:10.1109/TVCG.2006.165. 2

[TFT*20] TEWARI A., FRIED O., THIES J., SITZMANN V., LOMBARDI S., SUNKAVALLI K., MARTIN-BRUALLA R., SIMON T., SARAGIH J., NIESSNER M., ET AL.: State of the art on neural rendering. In *Computer Graphics Forum* (2020), vol. 39, Wiley Online Library, pp. 701–727. 2

[The23] THE KHRONOS VULKAN WORKING GROUP: Vulkan 1.3.245 - A Specification. https://registry.khronos.org/vulkan/specs/1.3-extensions/html/vkspec.html, 2023. Accessed: 2023-03-30. 5

[TLY*21] TAKIKAWA T., LITALIEN J., YIN K., KREIS K., LOOP C., NOWROUZEZAHRAI D., JACOBSON A., MCGUIRE M., FIDLER S.: Neural geometric level of detail: Real-time rendering with implicit 3d shapes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2021), pp. 11358–11367. 2

[TSM*20] TANCIK M., SRINIVASAN P., MILDENHALL B., FRIDOVICH-KEIL S., RAGHAVAN N., SINGHAL U., RAMAMOORTHI R., BARRON J., NG R.: Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in Neural Information Processing Systems 33* (2020), 7537–7547. 2, 4

[WHW22] WEISS S., HERMÜLLER P., WESTERMANN R.: Fast neural representations for direct volume rendering. In *Computer Graphics Forum* (2022), vol. 41, Wiley Online Library, pp. 196–211. 2, 4

[XTS*22] XIE Y., TAKIKAWA T., SAITO S., LITANY O., YAN S., KHAN N., TOMBARI F., TOMPKIN J., SITZMANN V., SRIDHAR S.: Neural fields in visual computing and beyond. In *Computer Graphics Forum* (2022), vol. 41, Wiley Online Library, pp. 641–676. 2

[ZD11] ZENG X., DURRANI T.: Estimation of mutual information using copula density function. *Electronics letters 47*, 8 (2011), 493–494. 3

[ZMZM14] ZHANG Z., MCDONNELL K. T., ZADOK E., MUELLER K.: Visual correlation analysis of numerical and categorical data on the correlation map. *IEEE transactions on visualization and computer graphics 21*, 2 (2014), 289–303. 2