URBAN AIR POLLUTION FORECASTS GENERATED FROM LATENT SPACE REPRESENTATIONS

César Quilodrán Casas & Rossella Arcucci & Yike Guo
Data Science Institute
Imperial College London
London, UK
{caq13,r.arcucci,y.guo}@imperial.ac.uk

ABSTRACT

This paper presents an approach to replicate computational fluid dynamics simulations of air pollution using deep learning. The study area is in London, where a tracer aims to replicate a busy traffic junction. Our method, which integrates Principal Components Analysis (PCA) and autoencoders (AE), is a computationally cheaper way to generate a latent space representation of the original unstructured mesh model. Once the PCA is applied on the original model solution, a Fully-Connected AE is trained on the full-rank Principal Components. This yields a compression of the original data by 10^6. The number of trainable parameters is also reduced using this method. A Long Short Term Memory (LSTM) based approach is used on the latent space to produce faster forecasts of the air pollution tracer.

1 INTRODUCTION

Given the amount of data in Computational Fluid Dynamics (CFD) simulations, data-driven approaches can be seen as attractive solutions to produce low-dimension model surrogates. Traditionally, these methods represent CFD using linear functions. Nonetheless, this is a highly non-linear problem. Previous studies (Wiewel et al., 2019; Kim et al., 2019) have proposed to reduce the dimensionality of CFD simulation using Convolutional Neural Networks (CNNs) and therefore represent the problem in a reduced non-linear space. This can easily be done in structured meshes. However, CNNs applied on unstructured meshes do not guarantee to represent the spatial information between nodes, once the data has been linearised.

We introduce an autoencoder based on full-rank Principal Components of a CFD simulation. This approach is a two-step dimension reduction of the original number of nodes: first with PCA, and then with an autoencoder trained on the full-rank Principal components. The motivation to use this approach is that; PCA is a linearised approach to a non-linear problem and that the truncation of the PCA results in order to reduce dimensionality comes with loss of information. The autoencoder is a non-linear approach which delivers a non-linear representation of the full-rank PCs. Previous works (Reddy et al., 2019) have shown that a similar approach works in 2D, but this work introduces a working 3D case.

The paper is structured as follows: in section 2 we present the methodology , in section 3 we present the study area and data, section 4 shows the results of the application of the methodology on the study area, and finally section 5 presents the summary and future work.

2 METHODS

The methodology combines two dimension reductions of the original data and a temporal neural network. Given that the CFD simulation used in this study contains $\geq 10^5$ dimensions and it is in an unstructured mesh, there are two issues that arise if fast forecasts are needed.

Firstly, a dimension reduction. As a simulation of this size requires the use of a supercomputer in order to produce the next time-step. One way to reduce the dimensionality is using Principal

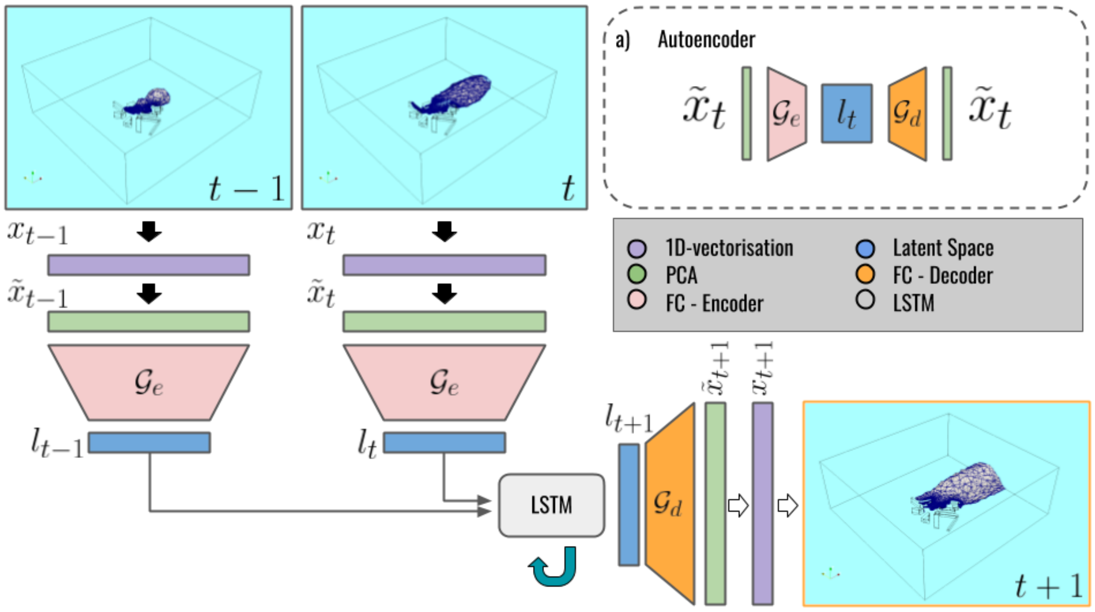

Figure 1: Proposed workflow to produce faster surrogate CFD forecasts. The first step is to reduce the dimensionality in two steps: a PCA, and an autoencoder. This produces a latent space that can be trained further with a Long short-term memory network which yields reduction in runtimes.

Components Analysis (PCA), which decomposes the data into linear basis functions that describe the original problem data. The caveat of using PCA is that the dimension reduction comes from truncating the Principal Components (PC), and by doing so the retained variance decreases. Our methodology proposes the application of PCA on the original data, however, the full-rank PC are kept, i.e. no truncation. This yields a square matrix with dimension $m \times m$, where m is the number of time-steps. An autoencoder is then applied on the full-rank PC and the dimensionality is further reduced. The latent space produced by the encoder is a non-linear representation of the PC and therefore there is no truncation.

Secondly, this is a CFD simulation in an unstructured mesh. State-of-the-art techniques use CNN and autoencoders to reduce the dimensionality of systems (Kim et al., 2019). The problem with unstructured meshes is that the data is not structured and therefore a convolution of the nodes might not necessarily represent the spatial relationship between them within the simulation. Thus, a fully-connected autoencoder (FCAE) is more appropriate in this scenario.

The workflow is presented in Fig 1. The two-step dimension reduction is described in the next two sections.

2.1 Principal Components Analysis

As described by Lever et al. (2017) PCA is an unsupervised learning method that simplifies high-dimensional data by transforming it into fewer dimensions. Let

$$\mathbf{u}^{\mathcal{M}} = \{\mathbf{u}_k^{\mathcal{M}}\}_{k=0,\dots,m} \tag{1}$$

denotes the matrix of the model vectors at each time step. The PCA consists in decomposing this dataset as:

$$\mathbf{u}^{\mathcal{M}} = \mathbf{P}^{\mathcal{M}}\mathbf{\Pi}^{\mathcal{M}} + \bar{\mathbf{u}}^{\mathcal{M}} \tag{2}$$

where $\mathbf{u}^{\mathcal{M}}$ is the dataset of the model; $\mathbf{P}^{\mathcal{M}} \in \Re^{m\times m}$ is the principal components of $\mathbf{u}^{\mathcal{M}}$; $\mathbf{\Pi}^{\mathcal{M}} \in \Re^{m\times n}$ are the Empirical Orthogonal Functions; and $\bar{\mathbf{u}}^{\mathcal{M}}$ is the mean vector of the model. The dimension reduction of the system is then obtained by using $\mathbf{P}^{\mathcal{M}}$ as input and output, respectively of a FCAE.

2.2 Fully connected Auto Encoder

The second dimension reduction comes from using a FCAE (Baldi, 2012). The FCAE is trained to take $\mathbf{P}^{\mathcal{M}}$ as input, and use it as target and output. This way there is no need to traditionally truncate the Principal Components but rather reducing its dimensions even further and preserving them all in a non-linear latent space.

The proposed architecture of the FCAE is shown in 1. The FCAE is divided into two sub architectures: Encoder $\mathcal{G}_e$ and Decoder $\mathcal{G}_d$. Thus, the FCAE function is defined by:

$$\mathcal{G}_{AE} = \mathcal{G}_d \circ \mathcal{G}_e \tag{3}$$

and therefore, the predicted PCs using principal components and autoencoder (PCAE) $\mathbf{u}^{\mathbf{PCAE}}$ is obtained by:

$$\mathbf{P}^{\mathbf{PCAE}} = \mathcal{G}_d(\mathcal{G}_e(\mathbf{P}^{\mathcal{M}})) \tag{4}$$

$$\mathbf{u}^{\mathbf{PCAE}} = \mathbf{P}^{\mathbf{PCAE}}\mathbf{\Pi}^{\mathcal{M}} + \tilde{\mathbf{u}}^{\mathcal{M}} \tag{5}$$

$$\tag{6}$$

The encoder architecture contains the row-wise input from $\mathbf{P}^{\mathcal{M}}$ which is connected to sequential dense layers of 128, 64, 32, 16, and and finally 8 nodes to create the latent space:

$$\mathbf{l_s} = \mathcal{G}_{\mathbf{e}}(\mathbf{P}^{\mathcal{M}}) \tag{7}$$

The FCAE does not necessarily need to have these number of layers, and any other architecture that reduces the dimension of the original PC is welcomed. The decoder takes the latent space $\mathbf{l_s}$ as input and its sequentially brings back the latent space to the size of the row-wise output of $\mathbf{P}^{\mathbf{M}}$. In both encoder and decoder, there is a Leaky Rectified Linear Unit (LeakyReLU) activation function and Batch Normalisation for faster convergence after each dense layer.

The loss function (L_{AE}) of $\mathcal{G}_{AE}$ is then defined by:

$$L_{AE} = \underset{\theta_{\mathcal{G}_e},\theta_{\mathcal{G}_d}}{min} |\mathcal{G}_d(\mathcal{G}_e(\tilde{x}(t)) - \tilde{x}(t)|_2 \tag{8}$$

where θ_E and θ_D are the hyperparameters of the encoder and decoder, respectively.

Once the latent space of the full-rank PCs is obtained, it can be used jointly with a Long short-term memory network (LSTM) (Hochreiter & Schmidhuber, 1997) in order to make a prediction of the next time-step. In this paper, a vanilla LSTM network takes the previous time-steps of $\mathbf{l_s}^{\mathbf{t-N},\ldots,\mathbf{t}}$ as input and predicts $\mathbf{l_s}^{\mathbf{t+1}}$

3 Study area and model data

The computational fluid dynamics (CFD) simulations were carried out using Fluidity (Davies et al., 2011) (`http://fluidityproject.github.io/`). The study area is a 3D realistic representation of a part of South London, UK within the vicinity of the London South Bank University (LSBU). The dispersion of the pollution is described by the classic advection-diffusion equation such that the concentration of the pollution is seen as a passive scalar (eq. (9)).

$$\frac{\partial c}{\partial t} + \nabla.(\mathbf{u}c) = \nabla.\left(\overline{\overline{\kappa}}\nabla c\right) + F \tag{9}$$

where $\overline{\overline{\kappa}}$ is the diffusivity tensor (m^2/s) and F represents the source terms (kg/m^3/s).

The 3D case is composed of an unstructured mesh including $n = 100,040$ nodes and $m = 1000$ time-steps. The wind profile of the atmospheric boundary layer is represented by a log-profile velocity. The top and the sides of the model domain have a perfect slip boundary condition, while the facades of the buildings and the bottom of the model domain have a no-slip boundary. The outflow boundary condition is defined by zero pressure (no-stress) condition. The pollution background is modelled as a sinusoidal function as follows:

$$C(t) = \frac{1}{2}\left(sin\left(\frac{2\pi t}{T}\right) + 1\right) \tag{10}$$

where C is the pollutant concentration, and t and T are time and period, respectively, in seconds. This background pollution mimics waves of pollution in an urban environment. The tracer is a point source located at the centre of the traffic intersection, mimicking pollution in a traffic congested junction.

4 Results

The choice of training $\mathcal{G}_{AE}$ on the principal components of the model background is due to the number of trainable parameters. An equivalent fully connected autoencoder on the full-space correspond to 256.4 M trainable parameters. Our approach only has 98828 trainable parameters, which is cheaper to run and more adequate to the memory capacity of our systems.

The decision of a 8-D latent space comes arbitrarily in order to achieve a 1000-fold dimension reduction. The dimension of the latent space is easily modifiable. The total compression of the dataset is then 10^6 times from 800 Megabytes to $\sim$ 64 kilobytes in the 8-D latent space.

The advantage of having a lower dimension latent space is useful when a LSTM is applied to produce fast forecasts as less trainable parameters are needed. Once the LSTM is trained, the decoder projects the solution back to the full-rank PC space, which is then projected to the physical space. The LSTM is 4×10^4 faster than Fluidity as producing forecasts.

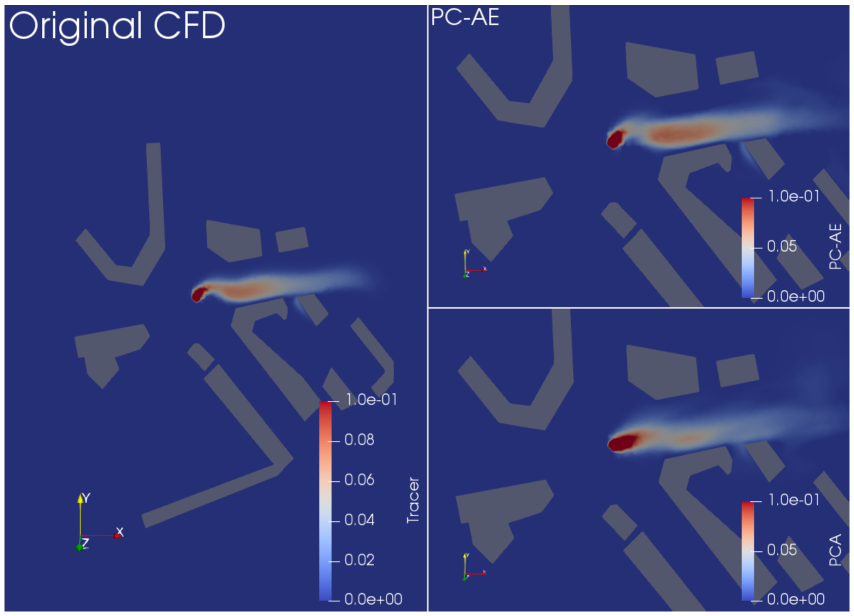

Figure 2: Comparison between CFD (Fluidity, top left), Principal Component Autoencoder (top right), LSTM per time-step (bottom left), and LSTM running free (bottom right).

5 Summary and future work

The framework proposes a two-step dimension reduction based on PCA and autoencoders. The framework was validated using a CFD Fluidity simulation. It is proposed that a two-step dimension

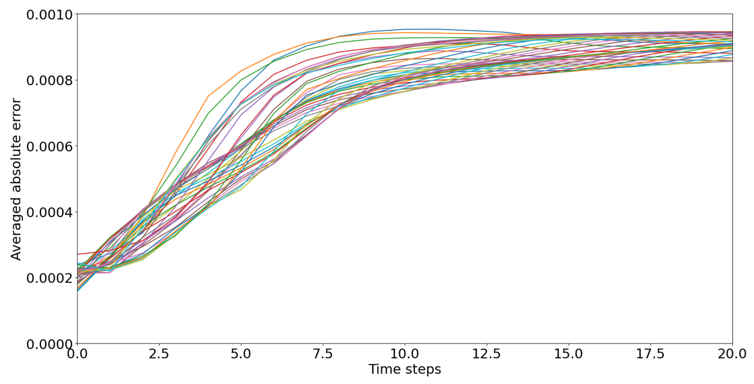

Figure 3: Averaged absolute error between the free running LSTM and the original CFD. From 40 different starting points.

reduction of this type can be used to keep the full-rank Principal Components in a non-linear latent space. Thus, it is not necessary to truncate the PCs.

This framework is encouraging as speed and accuracy are usually mutually exclusive terms. The results shown here are promising and demonstrate how the merger of these techniques could be useful in computationally demanding physical models.

Lastly, this framework is not exclusive to a Fluidity solution and the workflow could be applied to different physical models where sufficient temporal data is available.

Future work includes the scalability towards bigger domains. In order to make these CFD simulations faster, the numerical solution of Fluidity could be replaced by Deep Learning algorithms like Generative Adversarial Networks (GANs) including a temporal sequence in order to create physically realistic flows. The replacement of the CFD solution by these models will speed up the forecast process towards a real-time solution. This future work combined with the framework presented here has the potential to be very rapid. The CFD Fluidity simulation can also produce velocity and pressure vectors and therefore, future work will explore the inclusion of these in the PCAE.

Acknowledgments

This work is supported by the EPSRC Grand Challenge grant ‘Managing Air for Green Inner Cities (MAGIC) EP/N010221/1, the EPSRC Centre for Mathematics of Precision Healthcare EP/N0145291/1 and the EP/T003189/1 Health assessment across biological length scales for personal pollution exposure and its mitigation (INHALE). Thanks to Dr. Laetitia Mottet for the set up of the full model in Fluidity.

References

Pierre Baldi. Autoencoders, unsupervised learning, and deep architectures. In *Proceedings of ICML workshop on unsupervised and transfer learning*, pp. 37–49, 2012.

D Rhodri Davies, Cian R Wilson, and Stephan C Kramer. Fluidity: A fully unstructured anisotropic adaptive mesh computational modeling framework for geodynamics. *Geochemistry, Geophysics, Geosystems*, 12(6), 2011.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Byungsoo Kim, Vinicius C Azevedo, Nils Thuerey, Theodore Kim, Markus Gross, and Barbara Solenthaler. Deep fluids: A generative network for parameterized fluid simulations. In *Computer Graphics Forum*, volume 38, pp. 59–70. Wiley Online Library, 2019.

Jake Lever, Martin Krzywinski, and Naomi Altman. Points of significance: Principal component analysis, 2017.

Sandeep B Reddy, Allan Ross Magee, Rajeev K Jaiman, J Liu, W Xu, A Choudhary, and AA Hussain. Reduced order model for unsteady fluid flows via recurrent neural networks. In *International Conference on Offshore Mechanics and Arctic Engineering*, volume 58776, pp. V002T08A007. American Society of Mechanical Engineers, 2019.

Steffen Wiewel, Moritz Becher, and Nils Thuerey. Latent space physics: Towards learning the temporal evolution of fluid flow. In *Computer Graphics Forum*, volume 38, pp. 71–82. Wiley Online Library, 2019.