SCALING UP DEEP LEARNING FOR PDE-BASED MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Across numerous applications, forecasting relies on numerical solvers for partial differential equations (PDEs). Although the use of deep-learning techniques has been proposed, the uses have been restricted by the fact the training data are obtained using PDE solvers. Thereby, the uses were limited to domains, where the PDE solver was applicable, but no further.

We present methods for training on small domains, while applying the trained models on larger domains, with consistency constraints ensuring the solutions are physically meaningful even at the boundary of the small domains. We demonstrate the results on an air-pollution forecasting model for Dublin, Ireland.

1 INTRODUCTION

Solving partial differential equations (PDEs) underlies much of applied mathematics and engineering, ranging from computer graphics and financial pricing, to civil engineering and weather prediction. Conventional approaches to prediction in PDE models rely on numerical solvers and require substantial computing resources in the model-application phase. While in some application domains, such as structural engineering, the longer run-times may be acceptable, in domains with rapid decay of value of the prediction, such as weather forecasting, the run-time of the solver is of paramount importance.

In many such applications, the ability to generate large volumes of data facilitates the use of surrogate or reduced-order models (Benner et al., 2015) obtained using deep artificial neural networks (Goodfellow et al., 2016). Although the observation that artificial neural networks could be applied to physical models is not new (Lagaris et al., 1998; 2000; Lee & Kang, 1990; Ramuhalli et al., 2005; Delpiano & Zegers, 2006; Muro & Ferrari, 2009; Rudd et al., 2014; Lee & Kang, 1990; Rudd, 2013), and indeed, it is seen as one of the key trends (Bellinger et al., 2017; Karpatne et al., 2017; Swischuk et al., 2018) on the interface of applied mathematics, data science, and deep learning, their applications did not reach the level of success observed in the field of the image classification, speech recognition, machine translation, and other problems processing unstructured high-dimensional data, yet. A key issue faced by applications of deep-learning techniques to physical models is their scalability.

Even very recent research on deep-learning for physical models (Tompson et al., 2017; James et al., 2018; Wiewel et al., 2018) uses a solver for PDEs to obtain hundreds of thousands of outputs. The deep learning can then be seen as means of non-linear regression between the inputs and outputs. For example, (James et al., 2018) have recently observed a factor of 12,000 computational speedup compared to that of a leading solver for the PDE, on the largest domain they were able to work with. Considering the PDE solver is used to generate the outputs to train the deep-learning model on, however, the deep-learning model is limited to the domain and application that it is trained on.

We present methods for training Deep Neural Networks (DNNs) on small domains, while applying the trained models on larger domains, with consistency constraints ensuring the solutions are physically meaningful even at the boundaries of the small domains. Our contributions are as follows:

- definition of the consistency constraints, wherein the output for one (tile of a) mesh is used to constrain the output for another (tile of a) mesh.

- methods for applying the consistency constraints within the training of a DNN, which allows for an increase in the extent of the spatial domain by concatenating the outputs of several PDE-based models by considering boundary conditions and state at the boundary.
- a numerical study of the approach on a pollution-forecasting problem, wherein we lose accuracy from 1 to 7 per cent compared to the unconstrained model, but remove boundary artefacts.

We note that the methods can be applied both in terms of "patching" multiple (tiles of a) meshes, and in terms of "zooming" in multi-resolution approaches, where lower-resolution (e.g., city-, country-scale) component constrains higher-resolution components (e.g., district-, city-scale), which in turn impose consistency constraints on the former.

2 Our Approach

Our work is a first attempt to apply techniques based on domain decomposition to deep learning. Conceptually, it promises the ability to concatenate outputs from disparate PDE-based simulation models into a single dataset for training deep learning with constraints to ensure consistency across the boundaries of the disparate simulations, i.e., physical viability across multiple meshes for a physical phenomenon governed by a PDE. The approach under consideration is rather intuitive and consists of training a deep learning model for each available sub-grid, providing a method to ensure consistency across sub-grids, and scaling up to the wider area such that the accuracy of the predictions is increased. Further, by enabling communication between meshes (via constraints), individual domain prediction can be provided with information external to the domain.

Let us consider an index-set $\mathcal{M}$ of meshes M_m, $m \in \mathcal{M}$, with sets of n_m mesh points $P(M_m)$. The output of each PDE-based simulation on such a mesh consists of values in $\mathbb{R}^{d_m}$ at each point of $P(M_m)$. Often, a small sub-set of $n_m^{(r)}$ of such points is of particular interest, which we call receptors $R(M_m)$; the remainder of the points represents hidden points $H(M_m)$. The receptors and hidden points thus partition the mesh points $P(M_m) = H(M_m) \cup R(M_m)$, with $n_m = n_m^{(h)} + n_m^{(r)}$. Further, let us consider the index-set $\mathcal{B} \subseteq \mathcal{M} \times \mathcal{M}$ of boundaries B_{mn} of meshes. Such a possibly infinite boundary $B_{mn} \subseteq P(M_m) \times P(M_n)$ links pairs of points from the two meshes. To each boundary B_{mn} we associate a constant ϵ_{mn} that reflects the importance of this boundary. Further, for each mesh M_m we have an ordered set of simulations indexed with time $t \in \mathbb{Z}$, where each simulation is defined by the inputs $x_t^{(m)} \in X_t^{(m)}$ and a set of outputs $y_t^{(m)} \in (\mathbb{R}^{d_m})^{\times n_m}$. Often, one wishes to consider $y_t^{(m)}$ being part of $x_{t+k}^{(m)}$ for some $k > 0$, in a recurrent fashion.

Our aim is to minimise residuals subject to consistency constraints to ensure physical "sanity" of the results, i.e.,

$$
\begin{aligned}
r^* = \min_f \sum_t \sum_{m \in \mathcal{M}} \left\| \mathrm{proj}_{R(M_m)} \Big(y_t^{(m)} - f^{(m)}\big(x_t^{(m)}\big) \Big) \right\| \qquad (1)\\
\text{s.t. } \forall t \; \forall (m,n) \in \mathcal{B} \; \forall (p_1, p_2) \in B_{mn} : \\
\mathrm{prox}\Big(\mathrm{proj}_{\{p_1\}} f^{(m)}\big(x_t^{(m)}\big) \,,\; \mathrm{proj}_{\{p_2\}} f^{(n)}\big(x_t^{(n)}\big) \Big) \leq \epsilon_{mn} \,,
\end{aligned}
$$

where $\mathrm{proj}_Q : (\mathbb{R}^{d_m})^{\times n_m} \to (\mathbb{R}^{d_m})^{\times |Q|}$ is a projection operator that projects the array of outputs at all points onto the outputs at a subset of points $Q \subset P(M_m)$, prox is a proximity operator, the decision variable defines the mapping $f = \{f^{(m)}\}^{m \in \mathcal{M}}$, whereby $f^{(m)}\big(x_t^{(m)}\big)$ represents the output of a non-linear mapping between inputs and PDE-based simulation outputs at the points of the mesh, $f^{(m)} : X_t^{(m)} \to (\mathbb{R}^{d_m})^{\times n_m}$, on each independent mesh M_m, which can be seen as a non-linear regression, and ϵ_{mn} is a constant specific to $(m,n) \in \mathcal{B}$. We provide examples of $f^{(m)}$ in the following sections. The requirement of physical "sanity" is usually a statement about smoothness of the values of the mapping $f^{(m)}$ across the boundaries of two different meshes. To be able to compare those values, we require that the dimensions are the same, that is $\forall m, n \in \mathcal{M} : \; d_m = d_n \equiv d$. For example, for prox being the norm of a difference of the arguments, "smooth" at a point at the boundary of two meshes means that the values predicted within the two meshes at that point are numerically close to each other. Also adding the norm of the difference of their gradients to that

makes it a statement about the closeness of their first derivatives too. Technically, "smoothness" is a statement about all their higher derivatives as well, however, we will only concern ourselves with their values, or zeroth order of derivatives, for now. Notice though that generically this is an infinitely large problem.

In theory, equation 1 can be solved by Lagrangian relaxation techniques, e.g., by solving:

$$r^* = \inf_{f,\,\lambda} \sum_t \Bigg(\sum_{m\in\mathcal{M}} \Big\| \mathrm{proj}_{R(M_m)} \Big(y_t^{(m)} - f^{(m)}(x_t^{(m)}) \Big) \Big\| + \sum_{(m,n)\in\mathcal{B}} \sum_{(p_1,p_2)\in B_{mn}} \lambda_t^{(m)} \,\mathrm{prox}\Big(\mathrm{proj}_{\{p_1\}} f^{(m)}\big(x_t^{(m)}\big)\,,\ \mathrm{proj}_{\{p_2\}} f^{(n)}\big(x_t^{(n)}\big) \Big) \Bigg), \tag{2}$$

for $\lambda = \{\lambda_t^{(m)}\}_{t\in\mathbb{Z}}^{m\in\mathcal{M}}$. This can be seen as a statement that there exist $\lambda_t^{(m)}$, $t \in \mathbb{Z}$, such that the infimum over $f^{(m)}$ coincides with r^*, for each $m \in \mathcal{M}$. Clearly, if at least some of the boundaries B_{mn} are infinite, then the optimisation problem is infinite-dimensional.

Further, one can borrow techniques from iterative solution schemes in the numerical analysis domain. Notice that the first term in equation 2 is finite-dimensional and separable across the meshes. For each mesh M_m, $m \in \mathcal{M}$, the above can be computed independently. Further, one can subsample the boundaries to obtain a consistent estimator. Subsequently, one could solve the finite-dimensional projections of equation 2, wherein each new solution will increase the dimension of $\lambda_t^{(m)}$. While this is feasible in theory, the inclusion of non-separable terms with $\lambda_t^{(m)}$ would still render the solver less than practical.

Instead, we propose an iterative scheme, which is restricted to separable approximations. Let us imagine that at time t, for a pair of points $(p_1, p_2) \in B_{mn}$ on the boundary indexed with $(m, n) \in \mathcal{B}$, we obtain values from the trained model at those points in the respective mesh, $\mathbb{R}^d \ni f_{p_1,t}^{(m)} = \mathrm{proj}_{\{p_1\}} f^{(m)}\big(x_t^{(m)}\big)$ and $\mathbb{R}^d \ni f_{p_2,t}^{(n)} = \mathrm{proj}_{\{p_2\}} f^{(n)}\big(x_t^{(n)}\big)$. While the two points p_1, p_2 lay in two different meshes, we would like the outputs at those points to coincide. For that we construct vectors $\underline{\chi}_{p_1,p_2}$ and $\overline{\chi}_{p_1,p_2} \in \mathbb{R}^d$ that serve as lower and upper bounds on the values obtained from the next iteration of the training of $f^{(m)}$, that is, we element-wise construct

$$\underline{\chi}_{p_1,p_2\,i} = \min\Big(f_{p_1,t\,i}^{(m)},\, f_{p_2,t_i}^{(n)} \Big) + \epsilon_{mn}\,, \qquad \overline{\chi}_{p_1,p_2\,i} = \max\Big(f_{p_1,t\,i}^{(m)},\, f_{p_2,t_i}^{(n)} \Big) - \epsilon_{mn}\,, \tag{3}$$

from which we can form univariate interval constraints, restricting the corresponding elements of both $f^{(m)}$ at p_1 and $f^{(n)}$ at p_2 of the next iteration to the respective interval $\big(\underline{\chi}_{p_1,p_2\,i}, \overline{\chi}_{p_1,p_2\,i}\big)$. Notice that $\underline{\chi}_{p_1,p_2}$ and $\overline{\chi}_{p_1,p_2}$ become constant in the next iteration. Further, notice also that replacing $\lambda_t^{(m)}$ with a constant λ provides an upper bound on r^*, which is computationally much easier to solve.

In the scheme, we consider a finite-dimensional projection of equation 2. For each $(m, n) \in \mathcal{B}$ we consider a finite sample $\hat{B}_{mn} \subset B_{mn}$ of pairs of points, for which we obtain

$$r^* = \min_{f,\lambda} \sum_t \Bigg(\sum_{m\in\mathcal{M}} \Big\| \mathrm{proj}_{R(M_m)} \Big(y_t^{(m)} - f^{(m)}\big(x_t^{(m)}\big) \Big) \Big\| + \sum_{\substack{(m,n)\in\mathcal{B} \\ (p_1,p_2)\in\hat{B}_{mn}}} \sum_{(l,p)\in\{(m,p_1),(n,p_2)\}} \lambda \Big\| \max\big(0, \underline{\chi}_{p_1,p_2} - f_{p,t}^{(l)}\big) + \max\big(0, f_{p,t}^{(l)} - \overline{\chi}_{p_1,p_2}\big) \Big\|_1 \Bigg), \tag{4}$$

where we consider the function $\max : \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}^d$ to operate element-wise. Further, when we consider λ as a hyperparameter, we obtain an optimisation problem separable in $m \in \mathcal{M}$, which in the limit of $|\hat{B}_{mn}| \to |B_{mn}|$ provides an over-approximation for any λ.

In deep learning, this scheme should be seen as a recurrent neural network (RNN). A fundamental extension of RNN compared to traditional neural network approaches is parameter sharing across different parts of the model. We refer to (Goodfellow et al., 2016, Chapter 10) for an excellent

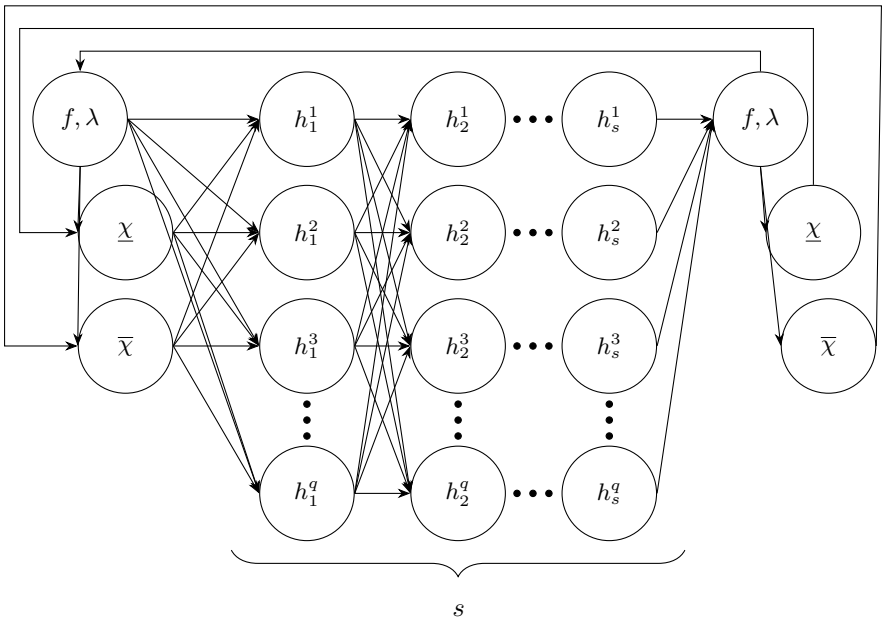

Figure 1: A schematic illustration of our recurrent neural network, where the recursion considers the consistency constraints defined by $\underline{\chi}, \overline{\chi}$.

introduction and Figure 1 for a schematic illustration. Each run provides constants $(\underline{\chi}_{p_1,p_2}, \overline{\chi}_{p_1,p_2})$, which are used in the consistency constraints of the subsequent run.

In terms of training the RNN, it is important to notice that equation 4 allows for very fast convergence rate even in many classes of non-linear maps f. For instance, when $f^{(m)} : X_t^{(m)} \to (\mathbb{R}^{d_m})^{\times n_m}$ is a polynomial of a fixed degree (Gergonne, 1974), then equation 4 is strongly convex, despite the max function making it non-smooth. The subgradient of the max function is well understood (Boyd & Vandenberghe, 2004) and readily implemented in major deep-learning frameworks. Even in cases, when the resulting problem is not convex, one could consider convexifications, following Mevissen et al. (2008).

In numerical analysis, in general, and with respect to the multi-fidelity methods (Peherstorfer et al., 2018), in particular, our approach could be seen as iterative model-order reduction. The original PDEs could be seen as the full-order model (FOM) to reduce, and equation 1 could be seen as a high-fidelity data-fit reduced-order model (ROM), albeit not a very practical one, whereas equation 4 could then be seen as a low-fidelity data-fit ROM, which allows for rapid prediction.

In learning theory, it is well known since the work of Cybenko (1989) that even a feed-forward network with three or more layers of a sufficient number of neurons (e.g., with sigmoidal activation function) allows for a universal approximation of functions on a bounded interval. It is not guaranteed, however, that the approximation has any further desirable properties, such as energy conservation etc. Our consistency constraints allow for such properties.

Fundamentally, the approach can be summarised as learning the non-linear mapping between inputs and predictions on each independent mesh, and iterating to ensure consistency of the solution across meshes. Such an approach draws on a long history of work on setting boundary conditions as consistency constraints in the solution of PDEs Quarteroni & Valli (1999). It can be applied to not only the simple patching of two tiles, but also when changing the resolution of the mesh. We use the term patching for working with neighbouring meshes at a single resolution and zooming when the mesh resolution changes. Both merging and zooming are illustrated in Figure 3.

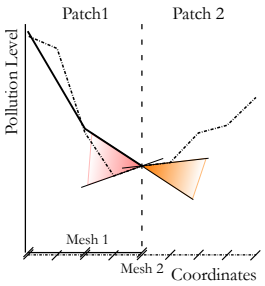

Figure 2: Discrete gradients: sub-differentials between predictions using two different meshes.

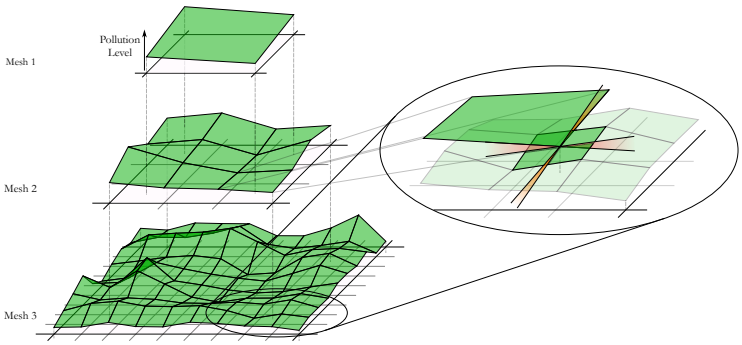

Figure 3: Changing the resolution of meshes imposes consistency constraints on the values and two-dimensional sub-differentials between predictions. Shown is the transition between three meshes of different resolution and the potential difference between the sub-differentials of figure 2.

3 METHODS

To illustrate this framework, we train the Recurrent Neural Network for city-scale pollution monitoring, utilising:

- The 3D structure of the atmosphere from our numerical weather forecasting model comprising the full atmospheric data (i.e., velocities, pressures, humidity, and temperatures in 3D).
- Pollution measurements and traffic data, since traffic is measurable and strongly correlated to (esp. nitrogen oxide, particulate matter) pollution in the cities.
- The given discretisation of a city in multiple meshes, corresponding to multiple geographic areas with their specificities.

Our test case is based in the city of Dublin, Ireland, for which real-time streams of traffic and pollution data (from Dublin City Council), and weather data (from the Weather Company) are available to us, but which did not have any large-scale models of air pollution deployed.

3.1 AIR POLLUTION-BASED FORECASTING

Air pollution is known to have significant health impacts Organization (2018). Typically, in cities, traffic-induced pollution is measured via the levels of nitrogen oxides (NOx) and Particulate Matter (PM). The contribution of traffic to the levels of NOx is known to be around 70% in European cities, whereas the contribution of traffic to the levels of particulate matter pollution is known to be up to 50% in cities of OECD countries, in particular due to the heavy presence of diesel engines.

We aim at estimating and predicting the traffic-induced air pollution levels of NOx, PM2.5 and PM10, for defined receptors across the city. An air pollution dispersion model propagates the pollution levels emitted from the roadway links (line sources). The PDE-based model that we are using is based on the Gaussian Plume model, studied at least since the work of Sutton (1947), and (by now) a standard model in describing the steady-state transport of pollutants. The data inputs are the periodic traffic volumes for a number of roadway links across the city, and periodic updates of atmospheric data. The outputs it provides are the estimates of pollution levels on a periodic basis. For a comprehensive review of line source dispersion models, the interested reader may refer to Nagendra & Khare (2002) or Stockie (2011).

In addition to the traffic and weather data inputs, the Gaussian Plume model takes a lot of parameters as inputs, such as the emission factors associated to the roadway links (depending on the composition of the fleet), the pollution dispersion coefficients which are a proxy for modelling the terrain (density of buildings, etc.), and the background pollution levels (pollution that is not traffic induced).

Such parameters are typically heterogeneous across cities and justify the use of different parameters, resolutions and physical resolution, hence PDE-based models, for the different meshes M_m under consideration.

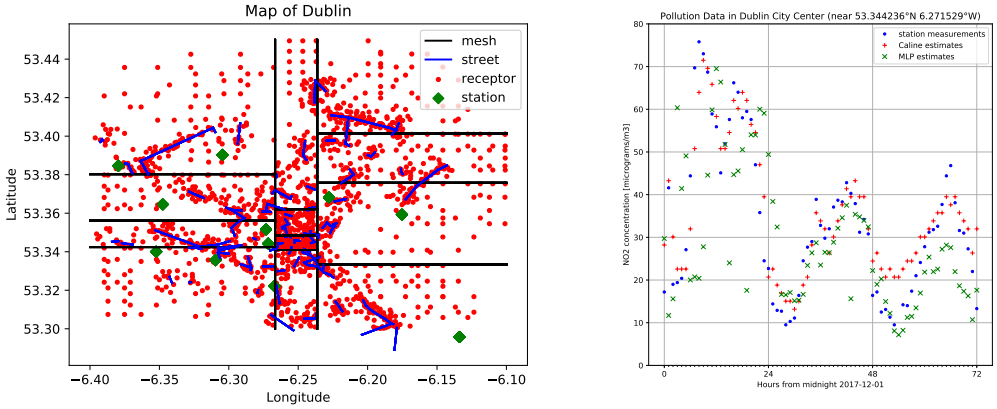

Figure 4: Left: Map of Dublin, partitioned into 12 domains (black lines), displaying the positions of line sources (blue lines), receptors (red dots), and measurement stations (green diamonds). Right: Output at 1 sample receptor collocated with a sensor over 3 days.

3.2 The Implementation

We use Caline 4 (at the California Department of Transportation), the open-source dispersion modelling suite, as a PDE solver to solve the Gaussian Plume model for the hourly inputs for each of the 12 domains described above. We note while Caline is one of the "Preferred and Recommended Air Quality Dispersion Model" of the Environmental Protection Agency in the USA (US-EPA, 2018), it is limited to 20 line sources and 20 receptors per solve, which in turn forces an arbitrary inhomogeneous discretisation of the road network and is another motivation for the use of our deep learning approach.

We have implemented the approach for the use case of Dublin, Ireland. There, the area is partitioned into 12 domains, with 10-20 line sources of pollution each mesh. Time is discretised to hours. For each hour, the inputs to the PDE solver comprise of traffic volume data at each line source, obtained from data aggregation of traffic loop detectors from the SCATS deployment in Dublin, and weather data at a discretisation of the spatial domain, obtained from The Weather Company under a licence: wind speed, wind direction, wind direction standard deviation, temperature, humidity. Available training data comprises almost one year worth of hourly data from July 1st 2017 to April 31st 2018. The outputs include concentrations of NO_2 (which is closely related to the NOx concentration), PM2.5 and PM10 concentrations at predefined receptors per domain, as suggested in Figure 4. The parameters were chosen for each mesh M_m based on the state-of-the-art practices: the emission factors based on the UK National Atmospheric Emissions Inventory database, dispersion coefficients based on the Caline recommendations (values for inner city, outer city areas), and background pollution levels chosen as the minimum time series values across the pollution measurements stations.

The RNN model is implemented in Tensorflow (Abadi et al., 2016) to obtain, in effect, the non-linear regression between the inputs and outputs, with the consistency constraints applied iteratively. That is, with each map from the inputs to the outputs, we also obtain further consistency constraints to use by further runs on the same domain.

Crucially, we use domain knowledge to pick ϵ_{mn} specific to $(m, n) \in \mathcal{B}$ based on the expected accuracy of the PDE-based model therein, as it is clear that a better accuracy can be expected when line sources are situated closer to the boundary. We hence consider ϵ_{mn} to be the maximum of 0.01 and the minimum distance of a line source to the boundary, where 0.01 corresponds to 100 meters. This choice takes effect not only in the threshold in the inequality 1, but via the construction of

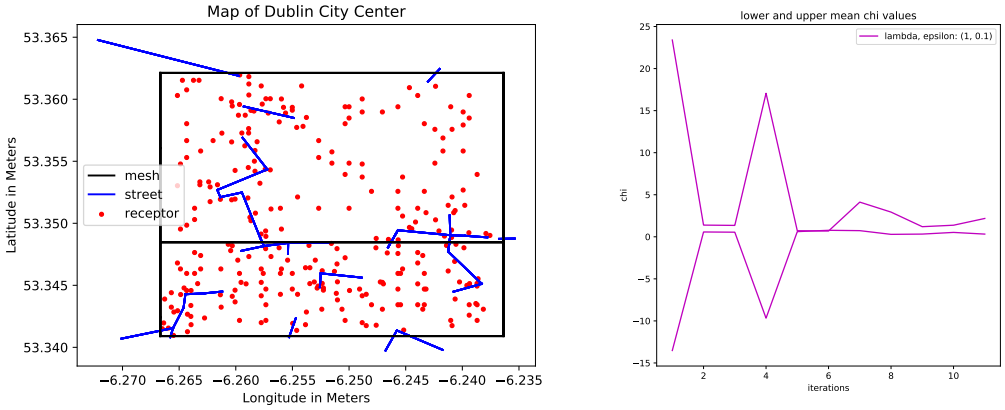

Figure 5: Left: Map of Dublin City Center, a close-up of figure 4. Right: Mean values for $\underline{\chi}$ and $\overline{\chi}$ for the boundary between the tiles left.

$\overline{\chi}_{p_1,p_2\,i}$ and $\underline{\chi}_{p_1,p_2\,i}$, it also affects the "learning rate": the higher the ϵ_{mn}, the faster the consistency constraints adapt to the solution obtained using the model trained on the adjacent mesh.

4 Results

For validation purposes, we have use hourly NOx concentrations measured at 6 sites across the city. (There are also 9 stations providing PM concentrations.) Figure 4 illustrates their positions, as well as the performance of the deep-learning forecaster at one example receptor, collocated with a measurement site used for our validation. The mean absolute percentage error (MAPE) of the deep-learning forecast without the consistency constraints was about 1 per cent, which was reduced to 7 per cent using the consistency constraints. This is because the same number of parameters have to fit an increased number of constraints, however, as can be seen from figure 5, the boundary artefacts disappeared after a few iterations of the training. These values have been taken from a sample training for which we achieved convergence. It will be interesting to optimise the algorithm architecture to observe convergence in more cases.

5 Conclusions

We have presented consistency constraints, which make it possible to train DNN on small domains and apply the trained models to larger domains while allowing incorporation of information external to the domain. The consistency constraints will ensure the solutions are physically meaningful even at the boundary of the small domains in the output of the DNN. We have demonstrated promising results on an air-pollution forecasting model for Dublin, Ireland.

The work is a first that makes possible numerous extensions. First, one could consider further applications of the consistency constraints, e.g., in energy conservation, or in consider merging the outputs of a number of PDE models within multi-physics applications. Second, in some applications, in may be useful to explore other network topologies. Following Wiewel et al. (2018), one could use long short-term memory (LSTM) units. Further, over-fitting control could be based on an improved stacked auto-encoder architecture (Zhou et al., 2017). In interpretation of the trained model, the approach of Cui et al. (2018) may be applicable.

Our work could also be seen as an example of Geometric Deep Learning (Bronstein et al., 2017), especially in conjunction with the use of mesh-free methods (Sirignano & Spiliopoulos, 2017), such as the 3D point clouds (Qi et al., 2017), non-uniform meshing, or non-uniform choice of receptors within the meshes. Especially for applications, where the grids are in 3D or higher dimensions, the need for such techniques is clear. More generally, one could explore links to isogeometric analysis of Cottrell et al. (2009), which integrates solving PDEs with geometric modelling.

Finally, one could generalise our methods in a number of directions of the multi-fidelity (Peherstorfer et al., 2018) modelling, e.g., by combining the reduced-order and full-order models using adaptation, fusion, or filtering. Overall, the scaling up of deep learning for PDE-based models seems to be a particular fruitful area for further research.

Within the domain of our example application, recent surveys (Bellinger et al., 2017) suggest that ours is the first use of deep learning in the forecasting of air pollution levels. Following the copious literature on PDE-based models of air pollution, one could consider further pollutants such as ground-level ozone concentrations (Mallet et al., 2013), and ensemble Mallet (2010) or multi-fidelity methods. One may also consider a joint model, allowing for traffic forecasting, weather forecasting, and air pollution forecasting, within the same network, possibly using LSTM units Cui et al. (2018), at the same time.

REFERENCES

Martín Abadi et al. Tensorflow: A system for large-scale machine learning. In *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation*, OSDI'16, pp. 265–283, Berkeley, CA, USA, 2016. USENIX Association.

Caltrans Team at the California Department of Transportation. Caline4 – a dispersion model for predicting air pollutant concentrations near roadways. *Report No. FHWA/CA/TL-84/15*.

Colin Bellinger, Mohomed Shazan Mohomed Jabbar, Osmar Zaïane, and Alvaro Osornio-Vargas. A systematic review of data mining and machine learning for air pollution epidemiology. *BMC public health*, 17(1):907, 2017.

Peter Benner, Serkan Gugercin, and Karen Willcox. A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM Review*, 57(4):483–531, 2015.

Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst. Geometric deep learning: Going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, July 2017. ISSN 1053-5888. doi: 10.1109/MSP.2017.2693418.

J Austin Cottrell, Thomas JR Hughes, and Yuri Bazilevs. *Isogeometric analysis: toward integration of CAD and FEA*. John Wiley & Sons, 2009.

Zhiyong Cui, Kristian Henrickson, Ruimin Ke, and Yinhai Wang. High-order graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting. *arXiv preprint arXiv:1802.07007*, 2018.

G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, Dec 1989.

J. Delpiano and P. Zegers. Semi-autonomous neural networks differential equation solver. In *The 2006 IEEE International Joint Conference on Neural Network Proceedings*, pp. 1863–1869, July 2006. doi: 10.1109/IJCNN.2006.246907.

JD Gergonne. The application of the method of least squares to the interpolation of sequences. *Historia Mathematica*, 1(4):439–447, 1974.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT Press, Cambridge, MA, 2016.

Scott C. James, Yushan Zhang, and Fearghal O'Donncha. A machine learning framework to forecast wave conditions. *Coastal Engineering*, 137:1–10, 2018. ISSN 0378-3839.

Anuj Karpatne, Gowtham Atluri, James H Faghmous, Michael Steinbach, Arindam Banerjee, Auroop Ganguly, Shashi Shekhar, Nagiza Samatova, and Vipin Kumar. Theory-guided data science: A new paradigm for scientific discovery from data. *IEEE Transactions on Knowledge and Data Engineering*, 29(10):2318–2331, 2017.

I. E. Lagaris, A. Likas, and D. I. Fotiadis. Artificial neural networks for solving ordinary and partial differential equations. *IEEE Transactions on Neural Networks*, 9(5):987–1000, Sept 1998. ISSN 1045-9227. doi: 10.1109/72.712178.

I. E. Lagaris, A. C. Likas, and D. G. Papageorgiou. Neural-network methods for boundary value problems with irregular boundaries. *IEEE Transactions on Neural Networks*, 11(5):1041–1049, Sept 2000. ISSN 1045-9227. doi: 10.1109/72.870037.

Hyuk Lee and In Seok Kang. Neural algorithm for solving differential equations. *Journal of Computational Physics*, 91(1):110–131, 1990.

Vivien Mallet. Ensemble forecast of analyses: Coupling data assimilation and sequential aggregation. *Journal of Geophysical Research: Atmospheres*, 115(D24), 2010.

Vivien Mallet, Alexander Nakonechny, and Sergiy Zhuk. Minimax filtering for sequential aggregation: Application to ensemble forecast of ozone analyses. *Journal of Geophysical Research: Atmospheres*, 118(19), 2013.

Martin Mevissen, Masakazu Kojima, Jiawang Nie, and Nobuki Takayama. Solving partial differential equations via sparse sdp relaxations. *Pacific Journal of Optimization*, 4(2):213–241, 2008.

G. Di Muro and S. Ferrari. A constrained backpropagation approach to solving partial differential equations in non-stationary environments. In *2009 International Joint Conference on Neural Networks*, pp. 685–689, June 2009. doi: 10.1109/IJCNN.2009.5179018.

S.M.S. Nagendra and Mukesh Khare. Line source emission modelling. *Atmospheric Environment*, 36(13):2083 – 2098, 2002. ISSN 1352-2310.

World Health Organization. *Burden of disease from household air pollution for 2016*. WHO, Geneva, Switzerland, 2018. URL http://www.who.int/airpollution/data/HAP_BoD_results_May2018_final.pdf.

Benjamin Peherstorfer, Karen Willcox, and Max Gunzburger. Survey of multifidelity methods in uncertainty propagation, inference, and optimization. *SIAM Review*, 60(3):550–591, 2018.

Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, pp. 5099–5108, 2017.

Alfio Quarteroni and Alberto Valli. Domain decomposition methods for partial differential equations numerical mathematics and scientific computation. *Quarteroni, A. Valli–New York: Oxford University Press.–1999*, 1999.

P. Ramuhalli, L. Udpa, and S. S. Udpa. Finite-element neural networks for solving differential equations. *IEEE Transactions on Neural Networks*, 16(6):1381–1392, Nov 2005. ISSN 1045-9227. doi: 10.1109/TNN.2005.857945.

K. Rudd, G. D. Muro, and S. Ferrari. A constrained backpropagation approach for the adaptive solution of partial differential equations. *IEEE Transactions on Neural Networks and Learning Systems*, 25(3):571–584, March 2014. ISSN 2162-237X. doi: 10.1109/TNNLS.2013.2277601.

Keith Rudd. *Solving Partial Differential Equations Using Artificial Neural Networks*. PhD thesis, 2013.

Justin Sirignano and Konstantinos Spiliopoulos. Dgm: A deep learning algorithm for solving partial differential equations. *arXiv preprint arXiv:1708.07469*, 2017.

John M Stockie. The mathematics of atmospheric dispersion modeling. *SIAM Review*, 53(2):349–372, 2011.

OG Sutton. The problem of diffusion in the lower atmosphere. *Quarterly Journal of the Royal Meteorological Society*, 73(317-318):257–281, 1947.

Renee Swischuk, Laura Mainini, Benjamin Peherstorfer, and Karen Willcox. Projection-based model reduction: Formulations for physics-based machine learning. *Computers and Fluids*, 2018. ISSN 0045-7930.

Jonathan Tompson, Kristofer Schlachter, Pablo Sprechmann, and Ken Perlin. Accelerating Eulerian fluid simulation with convolutional networks. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 3424–3433, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

US-EPA. Air quality dispersion modeling - preferred and recommended models, 2018. URL https://www.epa.gov/scram/air-quality-dispersion-modeling-preferred-and-recommended-models.

Steffen Wiewel, Moritz Becher, and Nils Thuerey. Latent-space physics: Towards learning the temporal evolution of fluid flow. *arXiv preprint arXiv:1802.10123*, 2018.

Teng Zhou, Guoqiang Han, Xuemiao Xu, Zhizhe Lin, Chu Han, Yuchang Huang, and Jing Qin. δ-agree adaboost stacked autoencoder for short-term traffic flow forecasting. *Neurocomputing*, 247: 31–38, 2017.