Physics-driven PTV interpolation through learned regularization

J. Floquet[1], Frédéric Champagnat[1*], Samir Beneddine[2]

1: ONERA the French Aerospace Lab, Information Processing and Systems Department, Palaiseau, France
2: ONERA the French Aerospace Lab, Aerodynamics Aeroelasticity Acoustics Department, Meudon, France
*Corresponding author: frederic.champagnat@onera.fr

Keywords: PTV processing, Neural Networks, Unrolled optimization

ABSTRACT

This communication deals with spatial interpolation of instantaneous Particle Tracking Velocimetry (PTV) data. Such a task can be handled using generic signal processing tools like B-splines, radial basis functions (RBF) or Kriging. We deal with non time-resolved PTV as gathered by dual frame particle image velocimetry (PIV) systems followed by dual frame PTV algorithms. In such a dual frame context, Physics-based interpolation techniques typically rely on enforcement of divergence free constraints on the interpolated field. The aim of this communication is to explore the wider field of Physics-based PTV interpolators based on learning a deep neural network. These interpolators are learned on a database of flow examples that may be generated by high resolution PIV or DNS. We propose an instance of such a Physics-based PTV interpolator based on the machine learning concept of *unrolled optimization*. In this communication, this concept is extended in order to deal with random measurement location, characteristic of PTV data. We also show how to use this concept to perform pressure estimation from PTV data, along with velocity interpolation. The concept is illustrated on a 2D laminar flow, DNS data is used to train and benchmark the proposed method against RBF interpolation. The proposed method is shown to be much more robust than RBF interpolation against low density seeding and noise.

1. Introduction

This communication deals with spatial interpolation of instantaneous Particle Tracking Velocimetry (PTV) data. Such a task can be handled using generic signal processing tools like B-splines (Gesemann et al., 2016) or spatial statistic tools such as Kriging and radial basis functions (RBFs) (Azijli & Dwight, 2015), (Vennell & Beatson, 2009). In the context of Data Assimilation, Physics-based interpolation can be handled for time-resolved PTV along an horizon of ten times steps (Schneiders & Scarano, 2016)(Gesemann et al., 2016)(Yang & Heitz, 2021). Here we deal with non time-resolved PTV as gathered by dual frame particle image velocimetry (PIV) systems followed by dual frame PTV algorithms (Fuchs et al., 2016; Cornic et al., 2020; Mikheev & Zubtsov, 2008; Takehara et al., 2000). In this dual frame context, Physics-based interpolation techniques merely

rely on enforcement of divergence free constraints on the interpolated field(Vennell & Beatson, 2009)(Azijli & Dwight, 2015).

The aim of this communication is to explore the wider field of Physics-based PTV interpolators based on learning a deep neural network. These interpolators are based on a database of flow examples that may be generated by high resolution PIV or DNS. We propose an instance of such a Physics-based PTV interpolator based on the machine learning concept of *unrolled optimization*. In this communication, this concept is extended in order to deal with random measurements characteristic of PTV data. The concept is illustrated on a 2D laminar flow, DNS data is used to train and benchmark the proposed method against RBF interpolation. The proposed method is shown to be much more robust than RBF interpolation against low density seeding and noise.

PTV interpolation is by nature an ill-posed problem. Whereas RBF interpolation or Kriging can be interpreted as instances of regularization using hand-crafted penalty functions (e.g. Tikhonov penalties), we propose to learn this penalty function based on flow samples. Therefore, our method intends to provide a Physics-based interpolation without resorting to Data Assimilation. The proposed method is an instance of unrolled optimization methodology (Gilton et al., 2020) and extends the concept to the non trivial case of random location of measurements, which is typical of PTV data.

It is well known that pressure can be inferred from velocity fields for compressible flows (van Oudheusden, 2013). Since our DNS yields the pressure fields in addition to velocities, we also use the pressure field to train our networks in order to show that the present approach can also infer pressure from PTV velocities, without explicitly solving the associated Poisson equation.

2. Proposed method

Unrolled optimization is a general framework to learn a regularizer for solving an inverse problem (Gilton et al., 2020). The following content is specialized to the recovery of the velocity field $\boldsymbol{u}$ from a partial measurement $\boldsymbol{y}$. Let us introduce the measurement operator h such that

$$\boldsymbol{y} = h(\boldsymbol{u}) + \boldsymbol{n}. \tag{1}$$

Depending on the experimental setup, h may be a filtering (PIV) or a sparse sampling (PTV) operator, $\boldsymbol{n}$ is a measurement noise. Operator h and noise statistics can be accurately calibrated at the beginning of the experimental phase, and are therefore considered as known (in a statistical sense) hereafter. The present study is restricted to linear operators h. This assumption is satisfied for classical PIV/PTV setup. A first approach for inversion of a measurement $\boldsymbol{y}$ is to define the best estimate $\widehat{\boldsymbol{u}}$ as the outcome of the following optimization problem

$$\widehat{\boldsymbol{u}} = \arg\min_{\boldsymbol{u}} \frac{1}{2}\|h(\boldsymbol{u}) - \boldsymbol{y}\|^2 + r(\boldsymbol{u}), \tag{2}$$

where $r(\boldsymbol{u})$ is a penalty function required to stabilize the optimization problem, and is typically hand-crafted. The computation of $\widehat{\boldsymbol{u}}$ is classically given by an iterative minimization algorithm. In

the simple case of a steepest gradient descent, it reads

$$\widehat{\boldsymbol{u}}^{(k+1)} = \widehat{\boldsymbol{u}}^{(k)} - \eta \left(h^* \left(h(\widehat{\boldsymbol{u}}^{(k)}) - \boldsymbol{y} \right) \right) - \eta \nabla r(\widehat{\boldsymbol{u}}^{(k)}), \tag{3}$$

where η is a predetermined descending step, $(\cdot)^*$ denotes the adjoint operator (which reduces for a linear operator, in the simplest case, to the transpose). Therefore, solving the optimization problem actually requires to specify ∇r rather than r. The present work proposes to define ∇r through supervised learning techniques.

Unlike gradient descent schemes that rely on a stopping rule based on a threshold on some residual, unrolled techniques specify a fixed number of iterations set to B. The overall Gradient Descent Network (GDN) then consists of B identical and sequential blocs. B is typically of the order of ten. Each block implements one iteration of (3). Last but not least, ∇r is implemented as a convolutional neural network $\nabla r(\cdot) = R(\boldsymbol{\theta}, \cdot)$ whose weights $\boldsymbol{\theta}$ are tuned during a supervised learning phase: the training database is made of pairs $(\boldsymbol{u}, \boldsymbol{y})$ and the weights $\boldsymbol{\theta}$ are calibrated through standard deep learning optimization techniques so that the output of the GDN minimizes a given error with respect to the database. The architecture of the overall GDN is depicted on Fig. 1. These ideas may be generalized to any descent-like optimization algorithm, variations about this template may be found in Adler & Öktem (2018); Aggarwal et al. (2019); Chen & Pock (2016); Gilton et al. (2020).

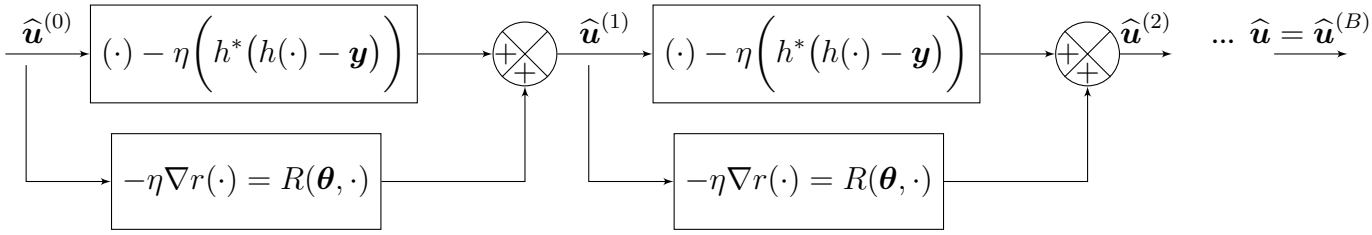

Figure 1. Architecture of a GDN corresponding to direct unrolled optimization. The upper part of each block is specified by the measurement part of the criterion, the only parameter to adjust is η (which is a trainable parameter in the unrolled algorithm). The lower part of each block is made-up of a convolutional neural network that will learn the gradient of the penalty function.

2.1. Including the pressure in the reconstruction

Optical measurement techniques such as PIV/PTV do not yield the pressure fields, which has to be estimated from the velocity field. Small adjustments from the previously-introduced unrolled approach may provide a data-based estimate of the full flow state $\boldsymbol{q} = (\boldsymbol{u}, p)^t$. The pressure $p(x, y)$ may easily be added to the reconstruction procedure by replacing $\boldsymbol{u}$ by the augmented vector $\boldsymbol{s} = (\boldsymbol{u}, p)^t$, and defining an augmented operator H such that $H(\boldsymbol{s}) = (f(\boldsymbol{u}), 0)^t$. Then the new unrolled approach is based on the iterative relation

$$\boldsymbol{s}^{(k+1)} = \boldsymbol{s}^{(k)} - \eta \left(H^* \left(\boldsymbol{y} - H(\boldsymbol{s}^{(k)}) \right) \right) - \eta \nabla r(\boldsymbol{s}^{(k)}). \tag{4}$$

Note that the operator H does not provide any physical or measurement information about the pressure. Therefore, the pressure estimation is entirely left for the term $\nabla r(\cdot)$ to be accounted for: the neural network will have to learn from data how to model the pressure field. However, if pointwise pressure measurement were available, then they could be easily taken into account in H by changing its definition. In the following, we focus on the particular case where no pressure measurements are available and assess the performances of the unrolled approach for flow reconstruction from PTV-only measurements.

2.2. Neumann Network variant for unrolled optimization

Gilton et al. (2019b) proposed a variant of the previous approach that is represented in figure 2. The general structure mainly differs from that of a classical unrolled approach by the presence of skip connections, represented in red. Mathematically, this new architecture is based on a Neumann series expansion of the operators appearing in the gradient descent iterations, details may be found in (Gilton et al., 2019b). Gilton et al. (2019b) showed that for image processing tasks, these so-called Neumann Network (NN) architectures yield better results compared to classical unrolled methods.

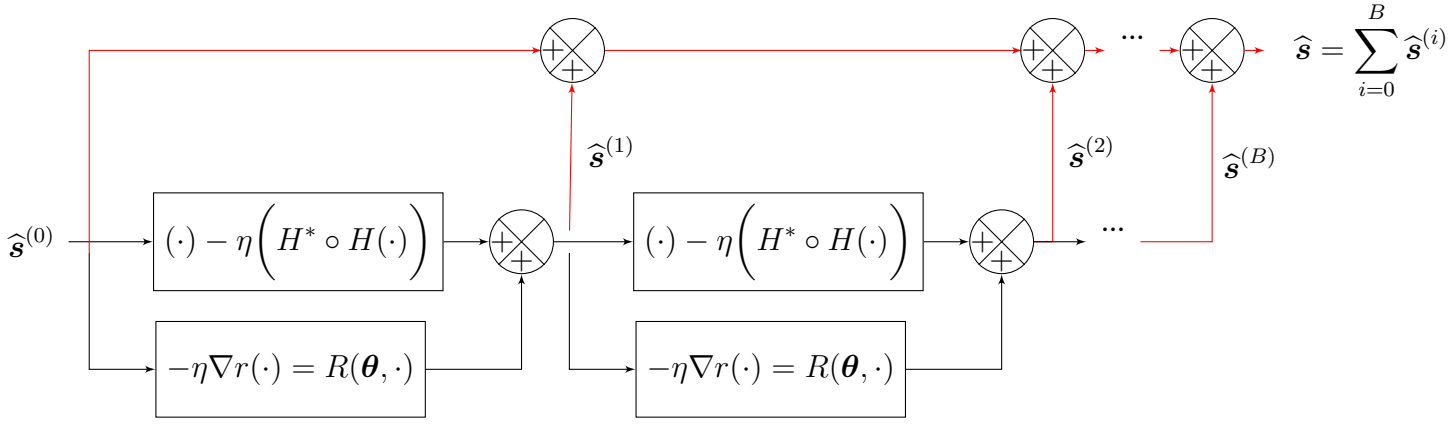

Figure 2. Architecture of a Neumann Network. At the input of the NN $\widehat{\boldsymbol{s}}^{(0)} = \eta H^* \boldsymbol{y}$.

2.3. Modelling PTV measurement operators

A PTV measurement is a sparse and random measurement of the flow, sparse because it is attached to a particle that has been seeded in the flow, random, because there is no way to predict the position of that particle at a given instant.

One possibility for modelling such measurement is through the introduction of a mask. Presently, the flow is represented by a three-channel image. Selecting some location on the flow can be represented as multiplication with a predefined mask, as sketched in the following figure 3.

This is an approximation of the actual situation because there is no reason that all particle positions

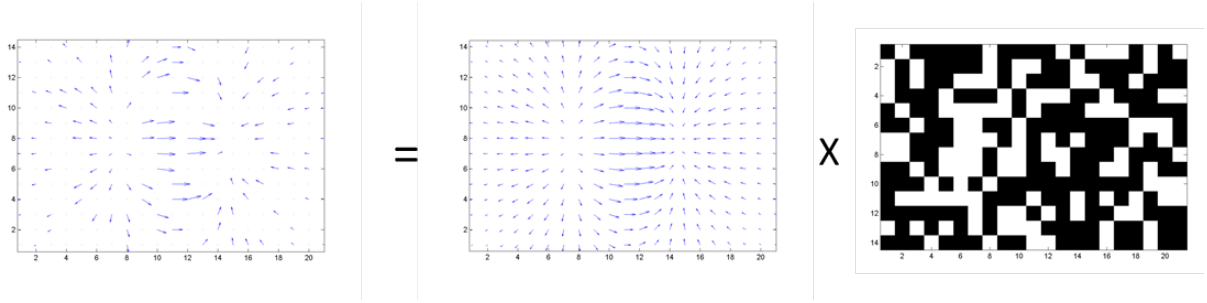

Figure 3. PTV measurement model: the sparse velocity field on the left may be considered as the product of a full velocity field and a zero-one mask.

would actually match pixel positions. A more realistic situation could be modelled as well, but it would require more development without providing more information about the performances of the present approach. The goal of this work is to study the concept of sparse measurement with unrolled approaches, and the proposed model seems sufficient for this goal.

Since the operator H has sparse support due to local measurement, the linear part of the network produces corrections only at pixels of the mask. This produces small spikes in the output U and V channels, these spikes are located at pixels of the mask. This could be corrected in principle by using a large number of blocs B. Instead we relied on median filters that are very efficient in suppressing impulse noise (Huang et al., 1979). In practice we found more efficient to keep a low number of blocks, but to add a final 3×3 median filter stage to the U and V channels output of the network.

Note that locations of flow measurements change from one PTV snapshot to another. Therefore, contrary to the classical unrolled approach from Gilton et al. (2020), the measurement operator H changes for each sample from the training database. This requires an original training strategy developed hereafter.

2.4. Learning strategy

The neural network has two kinds of inputs: the mask and the velocity measurements. It has also two kinds of outputs: the interpolated velocity field and the associated pressure field. The classical unrolled strategy deals with fixed H operator during training and testing. As mentioned earlier, this strategy is useless for PTV applications where measurement locations are random. Thus we consider random masks that change during the learning process in order to span the variety of measurement locations.

The training Data Base (DB) is made-up of T snapshots (U, V, P). Each snapshot is associated to a random mask with a prescribed number of non zero values (this number amounts to the Nppp parameter that characterises the seeding density in PTV). The learning procedure proved to be more efficient when the training quantities have similar range. In order to achieve this we

remove a single constant value to the pressure field before training. Therefore, pressure is presently recovered up to a constant value. Such a constant could be recovered using a side neural network, a task we leave here to future work.

An epoch of the learning process consists in feeding the network with data generated by the sample and the mask then comparing the network output with the original sample considering the three components (U, V, P) of the flow. The loss function to minimize is the Mean Squared Error defined as follows:

$$MSE(X,Y) = \frac{1}{C \cdot M \cdot N} \sum_{c,m,n=1}^{C,M,N} (X[c,m,n] - Y[c,m,n])^2, \tag{5}$$

with C representing the number of output channels (in our case $C = 3$ for the three flow variable (U, V, P)); M, N denote the horizontal and vertical width of the network output.

3. Experimental setting

3.1. Test case

The database used for training, validation and testing has been generated using the finite-volume compressible solver elsA (Cambier et al., 2013). The physical configuration corresponds to a bidimensional laminar wake behind a bluff body. The computational domain as well as the mesh and boundary conditions may be seen in figure 4. The mesh is made of approximately 50000 points, which are distributed in order to properly resolve the regions with strong velocity gradient.

The computation is carried out using non-dimensional quantities: the free-stream velocity and static pressure is set to 1, the Mach number is set to $M = 0.15$ such that compressibility effects (which scale with M^2) are negligible and the flow may be treated as incompressible. The only parameter that is varied is the Reynolds number (28 values considered between 200 and 1050) through the value of the viscosity. The spatial scheme is a Jameson scheme and the temporal integration is performed using the Gear algorithm.

Snapshots are extracted at a given sampling frequency, that is set to have at least 80 snapshots per period of vortex shedding (which depends on the considered Reynolds number). For each Reynolds number considered, the computed snapshots are interpolated on a regular grid spanning the domain $(x, y) \in [0, 8] \times [-2, 2]$ and made of 70×41 points. The interpolation is performed through a 3^{rd} order bi-variate spline using the Python package *sciPy*. An example of resulting fluctuation field may be seen in figure 5. The final database contains a total of 4200 tuples of 3 images associated with (U, V, P).

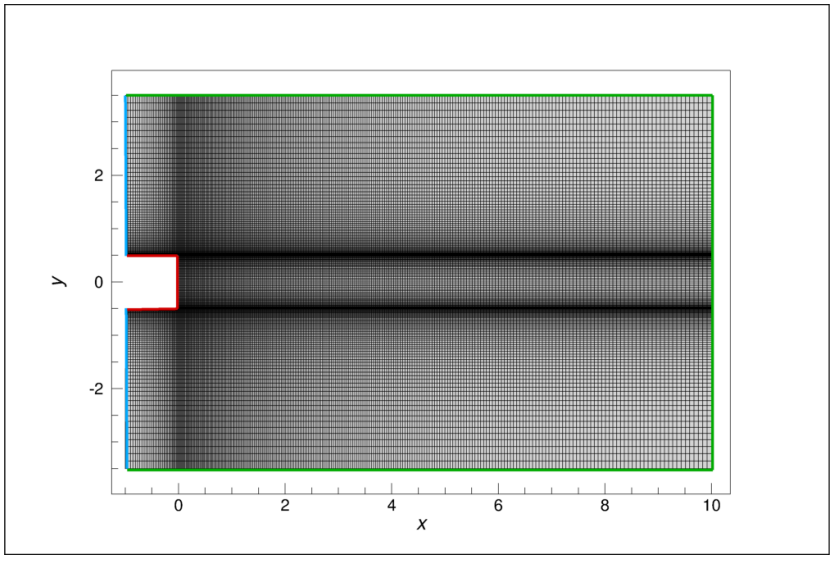

Figure 4. Mesh used for the computations. Boundary conditions are shown in color: uniform inflow conditions are in blue, adiabatic no-slip walls in red, and non-reflective farfield conditions in green.

3.2. Network architecture, learning and hyperperameter tuning

We consider hereafter either the Gradient Descent Network of figure 1 or the Neumann Network of figure 2.

The quantities to be learned are the regularizer $R(\theta, \cdot)$, and the step size η. Following Gilton et al. (2020) $R(\theta, \cdot)$ is a 7-layer convolutional neural network with an auto-encoder structure. the activation functions are ELU, the filters are square with sizes $(5,\ 3,\ 2,\ 3,\ 3,\ 5,\ 3)$, and strides $(1,\ 2,\ 1,\ 2,\ 1,\ 1,\ 1)$. The number of channels in each layer is $(3,\ 64,\ 256,\ 256,\ 256,\ 64,\ 3)$.

All the presented results where performed with 32×32 images on 3 channels (U, V, P) as inputs and outputs, plus 3 mask channels as input. The P input channel is always set to zero, since no pressure data is available.

The database (DB) used for training a model is performed on 32×32 images cropped randomly from the eslA DB, see figure 5.

Several databases of masks (MDB) were also generated with 4200 tuples of three 32×32 binary images coupled with DB. Each tuple have two identical binary matrix generated randomly and a third images with zero for all elements. Each MDB is characterised by an average percent of non-zero values, which amounts to Nppp in a PTV experiment.

The DB and MDB are split in 80% for training and 20% for validation, so 3360 couples of tuples for one and 840 for the second. This step enabled to give a unique mask for each tuple of DB and avoids overfitting for a particular mask.

The network has been trained with an ADAM optimizer with initial step equal to 5.010^{-3} and 0.99

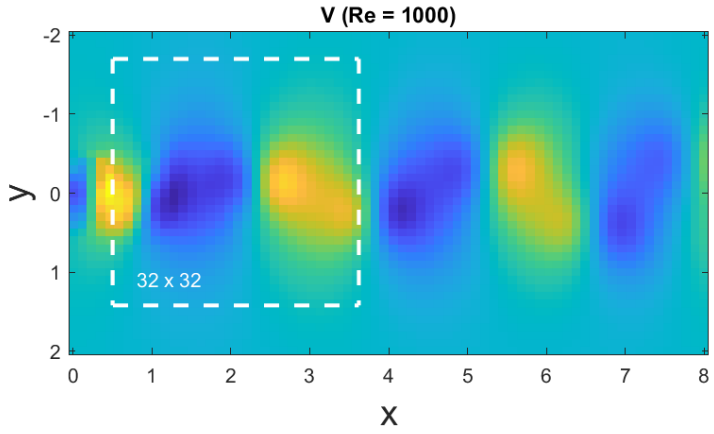

Figure 5. Snapshot of V component of the flow at $Re = 1000$, from the eslA database. An instance of a 32×32 pix. random crop is depicted with a white dashed line square.

as decay factor. Batch size has been set to 32. In order to avoid overfitting, stopping is monitored by checking the decrease of the loss on the validation set. The initial weights are random and $\eta^{(0)} = 1.03$. The Number B of blocks of unrolled algorithm has been tuned to $B = 4$ based on the final MSE for the validation DB. The different numerical results are obtained with Python $3.9.5$, PyTorch $1.10.0$, Torchvision $0.11.1$ with NVIDIA Quadro RTX.

Figure 6 shows a typical learning curve in terms of MSE and Peak SNR (PSNR) defined as

$$PSNR(X, Y) = -10 \log MSE(X, Y).$$

On figure 6, the MSE decreases fast on the first epochs, then this decrease slows down and is better seen on the PSNR logarithmic metric. After about 200 epochs, the validation loss starts to increase, yielding an early stopping (vertical red dashed line in figure 6).

3.3. Reference RBF method

A RBF method has been selected as a competitor to the NN or GDN. Interpolation is then performed independently on U and V channels. The best performing kernel for the DB is the inverse kernel:

$$K_\epsilon(r) = \frac{1}{\sqrt{r^2 + \epsilon^2}}, \tag{6}$$

where ϵ is a regularizing parameter tuned as the average distance between interpolating nodes. The method was implemented using the RBF interpolation package of SciPy.

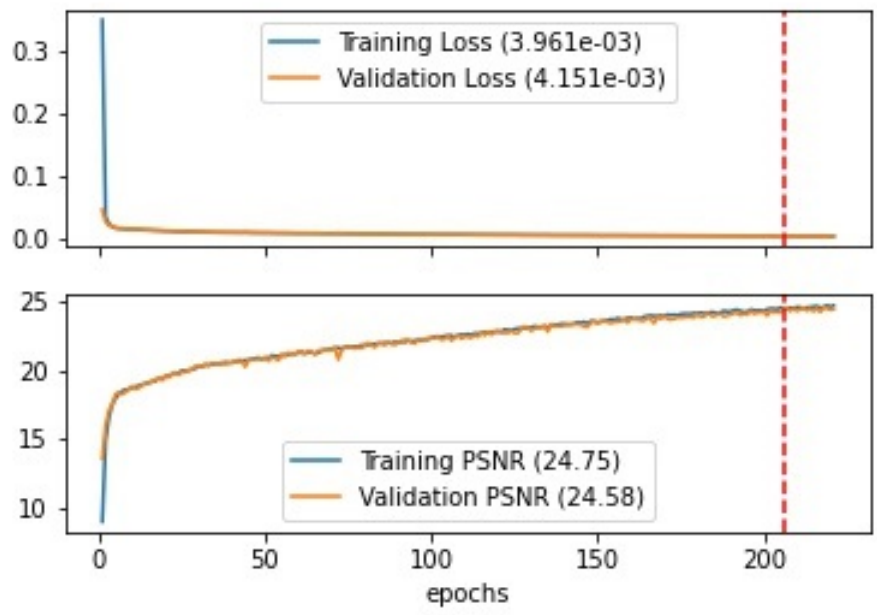

Figure 6. MSE and PSNR of training and validation in function number of epoch until the early stopping (red line). The number of non-zero values per image in the DBM is equal to 2% ($Nppp = 0.02$).

4. Experimental results

We assess hereafter the proposed methods (GDN, NN) and RBF reference on the proposed test case. These comparisons are performed using the error RMS on the velocity magnitude as main performance metric (Sciacchitano et al., 2021). Our first experiment assess GDN vs. NN. GDN is shown to perform best on a variety of seeding density and noise levels. Then we follow on by comparing GDN and RBF both on velocity and pressure estimation. The generalization capability of GDN is explored through decoupling of learning DB and testing DB.

4.1. Comparison of GDN and NN unrolling strategies

We trained the GDN and NN architectures for four $Nppp$ values: $0.02, 0.05, 0.1, 0.2$. During the test, a random 32×32 crop is performed in each of the basic eslA DB, then a random mask is generated in order to produce pseudo PTV measurements. A Gaussian random noise of standard deviation σ is added (since the dynamic of flow is about 1 in these experiments, $\sigma = 0.01$ amounts to 1% noise). Random crop and random masks are performed on-the-flight during the test, therefore generating the same pattern of measurement on the same sample as during training is very unlikely.

Adapting from Sciacchitano et al. (2021), the main performance metric is the error RMS on the velocity magnitude. Indeed, in the present case, this is almost equivalent to random error in Sciacchitano et al. (2021) since the bias is negligible.

The evolution of the RMS metric is shown on figure 7 with respect to seeding density and for two noise configurations ($\sigma = 0.0$ and $\sigma = 0.05$): GDN always performs consistently better than NN. We observed that the Neumann series associated to sparse measurement operators converges very slowly, compared to gradient iteration counterparts. Note also that the RMS increases for NN with no noise for the larger Nppp. The behaviour of NN could be improved using preconditionning as suggested by Gilton et al. (2020). We leave this conjecture to further work and concentrate on GDN in the following.

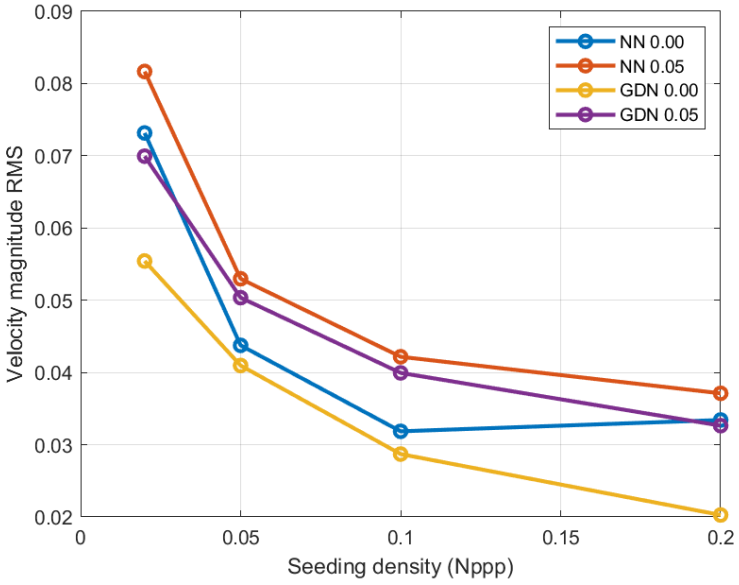

Figure 7. RMS of the velocity magnitude with respect to seeding density. For NN and GDN methods and two noise levels $\sigma = 0.0$ and $\sigma = 0.05$. GDN allways performs better than NN.

4.2. Comparison of GDN with respect to RBF reference

The GDN obtained in section 4.1 for four $Nppp$ values: $0.02, 0.05, 0.1, 0.2$ are tested and compared to the RBF approach, the RMS of the velocity magnitude with respect to seeding density is represented on figure 8.

As expected all quantities decrease as Nppp grows. It can be seen that the GDN approach is much more robust to addition of noise to PTV measurements. Adding 5% noise implies a slight increase in RMS for the GDN method, whereas it significantly degrades the RMS performance of RBF, in particular for the largest density, where the RMS increases by 50%. Moreover, the RMS performance of GDN interpolation is about 70% lower than its RBF competitor at the lowest density, whatever the noise level.

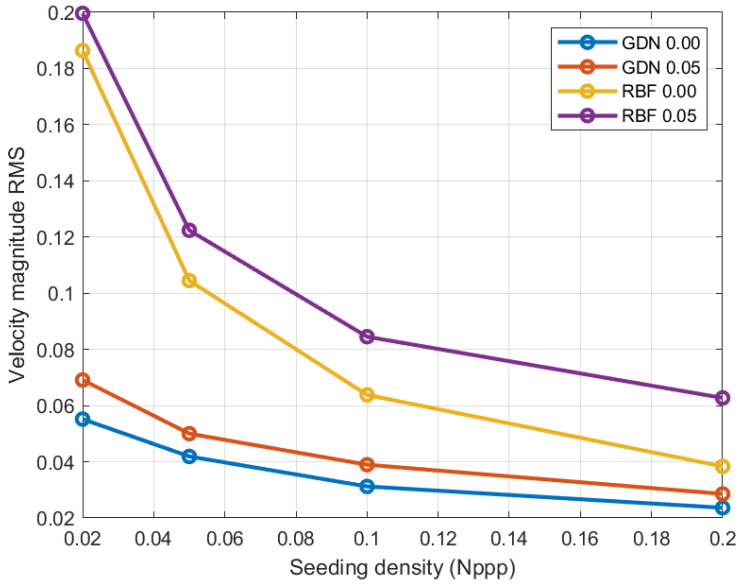

Figure 8. RMS of the velocity magnitude with respect to seeding density. For GDN and RBF methods and two noise levels $\sigma = 0.0$ and $\sigma = 0.05$.

4.3. Qualitative results

This section shows a sample of the output of the GDN approach compared to the RBF method, with $Nppp = 0.02$ and no added noise. The first column of figure 9 provides the ground truth for (U, V, P), and black circles mark the location of velocity data used by the GDN and RBF to recover the full field. Note that there is no circle on the pressure ground truth since there is no pressure measurement. The second row of figure 9 provides the output of the GDN, and the third row provides the RBF result. Since there was no input in terms of pressure, no output can be delivered by the RBF approach.

Figure 9 shows that the overall structure of (U, V, P) estimated by the GDN is much more faithful to the original field than the RBF interpolation. Note that the V field is particularly ill-recovered by the RBF method. One drawback of RBF at low density is clearly apparent here: due to random measurement location there is no measurement of the V field around its lowest values. The driving principle of RBFs is to spread measurements around their location consistently with respect to each other measurement, this principle fails here because there is no V measurement in the central part of the flow where two local minima occur. It is likely that in the present case, the prior learned by the Neumann Network can compensate the lack of data on V by using the correlation between U and V channels. Note that indeed inter-channel correlation enabled the main features of the P field to be recovered despite the lack of pressure data. The performance of pressure reconstruction is assessed in more detail in the next section.

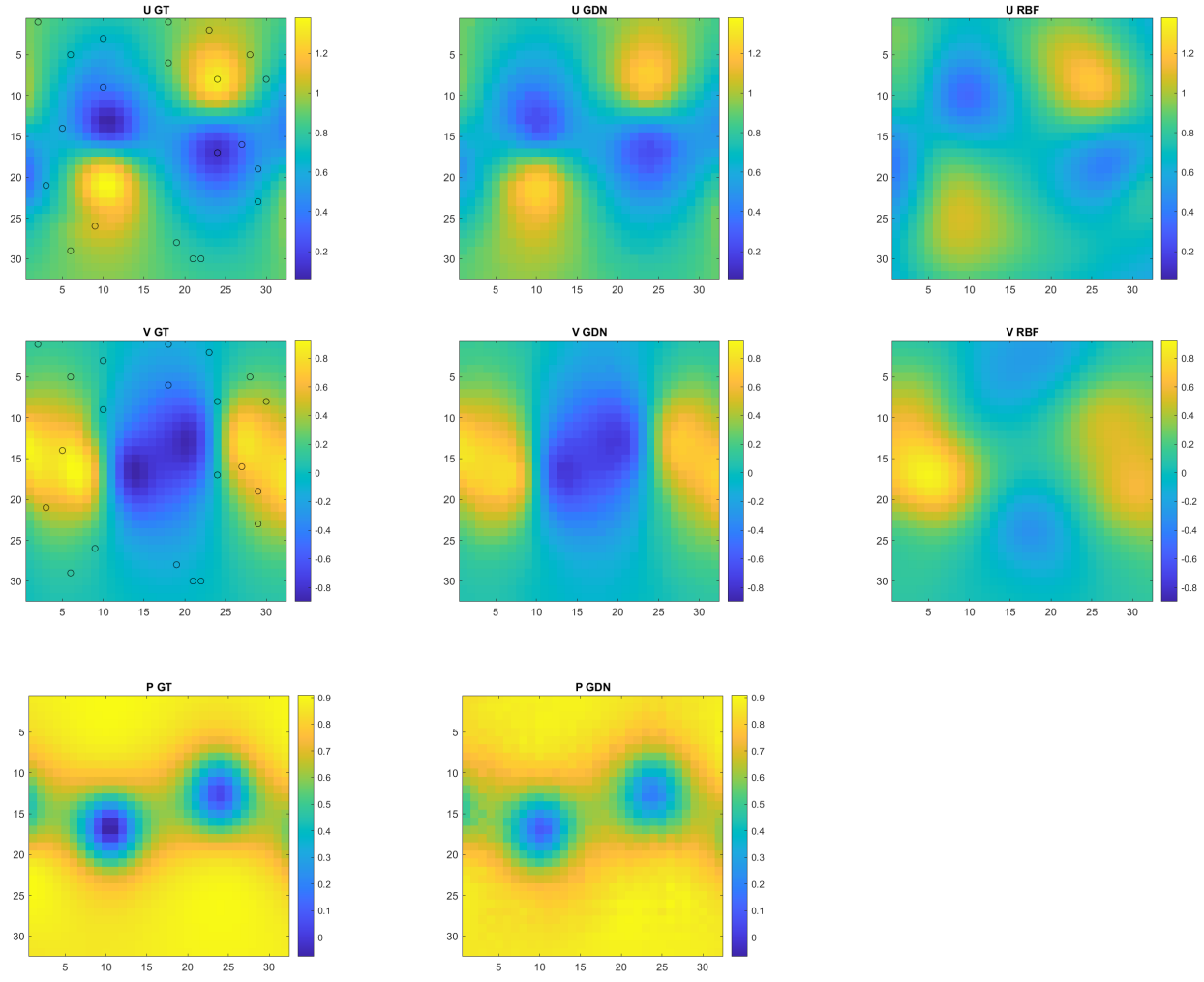

Figure 9. Sample of (U, V, P) fields recovered by Gradient Descent Network and Radial Basis function methods with noise level $\sigma = 0.00$. Black circles mark the location of velocity measurements. The central part of the V component of the flow is erroneous for RBF due to missing measurements in that area that contains to large basins.

4.4. Pressure from PTV measurements

Here we reuse the unrolled architectures learned in section 4.1 and focus on assessment of pressure estimation.

Figure 10 shows the RMS error for the pressure estimation with respect to the seeding density for the NN and GDN methods. The behaviour is consistent with the velocity estimation case of figure 7, indeed the RMS error decreases with Nppp, except for the NN with no noise. In the latter case, RMS increases slightly, consistently with the behaviour observed in figure 7. We believe this behaviour is linked to the slow convergence of the Neumann series associated to the measurement operator. GDN always perform better than NN also for pressure RMS. Noise impact is most noticeable for the lowest Nppp, where the RMS of GDN increases by 25%.

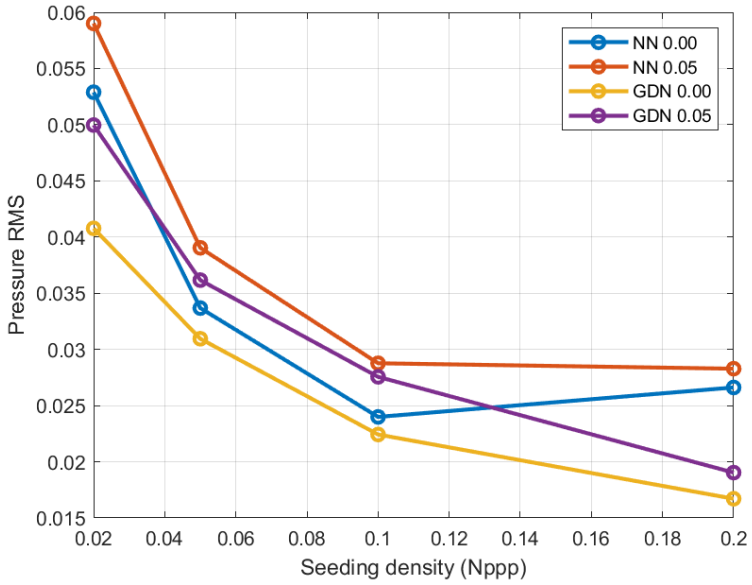

Figure 10. RMS of pressure estimation with respect to seeding density, for NN and GDN for two noise levels $\sigma = 0.00, 0.05$.

4.5. Generalisation capability of unrolled optimization

So far, training and test has been performed on DB covering the DNS flows for all the available Reynolds numbers. We now explore the generalisation capability: training and test DB will be based on non overlapping Reynolds numbers range. The training DB will be made on all simulations with Re number between 200 and 575. During training the seeding density is fixed to Nppp$= 0.02$. All other training parameters are kept as in previous experiments. the testing DB is made of all simulations with Reynolds number between 625 to 1050. This testing DB is split in four groups labelled 650, 750, 850 and 1000 gathering neighbouring Reynolds number simulations. During test, a random crop is performed in the selected flow image and a random mask is generated on the flight. The RMS of the velocity magnitude with respect to Re groups is shown in figure 11.

First, we observe that the RMS of GDN is about twice as large as that of figure 8, but note that the training DB here is about half as small as the one used in figure 8. One can observe a slight increase of RMS with respect to Re for all methods. The RMS of GDN stays about 40% below that of RBF method, whatever the noise level. So GDN extrapolates fairly well in the Reynolds number range between 625 to 1050.

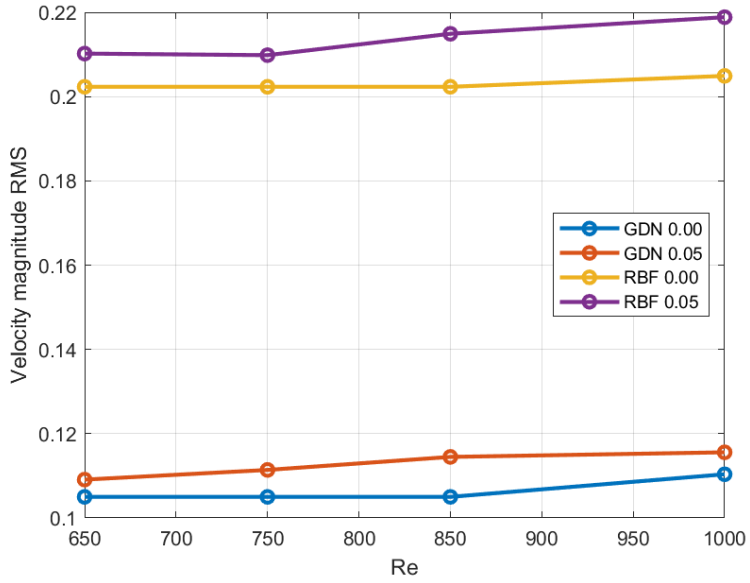

Figure 11. RMS of the velocity magnitude with respect to Re. For GDN and RBF methods and two noise levels $\sigma = 0.0$ and $\sigma = 0.05$. Seeding density is set to Nppp $= 0.02$.

5. Conclusions

We proposed an instance of Physics-based PTV interpolator based on the machine learning concept of *unrolled optimization*. We extended the concept in order to deal with random measurements characteristic of PTV data. We also show that such neural networks can be trained to infer pressure from velocity only measurements. The proposed method is illustrated on a 2D laminar flow, DNS data is used to train and benchmark the proposed method against RBF interpolation. The proposed method is shown to be much more robust than RBF interpolation against low seeding density and noise. Two variants of *unrolled optimization* have been tested, the basic Gradient Descent Network and the Neumann Network variant. The latter proved to be less accurate for recovering velocity magnitude and pressure. The Gradient Descent Network variant has also passed a preliminary test of generalization capability, where testing and learning database are made of simulations with non overlapping Reynolds number range.

This work can be completed and extended in many directions. First, the present approach recovers pressure up to a constant additive factor, that is fixed in the present study. This is because the different (U, V, P) channels of the network should have similar dynamic range. The remaining unknown constant could be estimated separately and added to the actual output of the NN or GDN. Second, the present implementation performs interpolation on 32×32 windows, the size of which can be augmented in relation to the memory available on training hardware. However, this augmentation will not reach Mega pixel images. Following (Gilton et al., 2019a), we propose to split such large images into overlapping patches (of size to be defined) each patch is then processed in parallel by the network and the output patches are then fused in order to yield a full image

result. Third, although the training database fulfils the divergence free constraint, this is not strictly the case for output velocities. The current approach can be reformulated in order to impose the divergence free constraint on output velocities.

References

Adler, J., & Öktem, O. (2018). Learned primal-dual reconstruction. *IEEE Transactions on Medical Imaging*, *37*(6), 1322-1332.

Aggarwal, H. K., Mani, M. P., & Jacob, M. (2019, Feb). MoDL: Model-based deep learning architecture for inverse problems. *IEEE Transactions on Medical Imaging*, *38*(2), 394–405.

Azijli, I., & Dwight, R. (2015). Solenoidal filtering of volumetric velocity measurements using gaussian process regression. *Experiments in Fluids*, *56*(11).

Cambier, L., Heib, S., & Plot, S. (2013). The onera elsa cfd software: input from research and feedback from industry. *Mechanics & Industry*, *14*(3), 159–174.

Chen, Y., & Pock, T. (2016). Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration. *IEEE transactions on pattern analysis and machine intelligence*, *39*(6), 1256–1272.

Cornic, P., Leclaire, B., Champagnat, F., Le Besnerais, G., Cheminet, A., Illoul, C., & Losfeld, G. (2020, Jan). Double-frame tomographic ptv at high seeding densities. *Experiments in Fluids*, *61*(2).

Fuchs, T., Hain, R., & Kähler, C. J. (2016). Double-frame 3D-PTV using a tomographic predictor. *Experiments in Fluids*, 1-5.

Gesemann, S., Huhn, F., Schanz, D., & Schröder, A. (2016). From noisy particle tracks to velocity, acceleration and pressure fields using b-splines and penalties. In *18th international symposium on applications of laser techniques to fluid mechanics* (pp. 1–17).

Gilton, D., Ongie, G., & Willett, R. (2019a). Learned patch-based regularization for inverse problems in imaging. In *2019 ieee 8th international workshop on computational advances in multi-sensor adaptive processing (camsap)* (p. 211-215).

Gilton, D., Ongie, G., & Willett, R. (2019b). Neumann networks for inverse problems in imaging. *arXiv preprint arXiv:1901.03707*.

Gilton, D., Ongie, G., & Willett, R. (2020). Neumann networks for linear inverse problems in imaging. *IEEE Transactions on Computational Imaging*, *6*, 328-343.

Huang, T., Yang, G., & Tang, G. (1979). A fast two-dimensional median filtering algorithm. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, *27*(1), 13-18.

Mikheev, A. V., & Zubtsov, V. M. (2008, jun). Enhanced particle-tracking velocimetry (EPTV) with a combined two-component pair-matching algorithm. *Measurement Science and Technology*, *19*(8), 085401.

Schneiders, J. F. G., & Scarano, F. (2016, 8). Dense velocity reconstruction from tomographic ptv with material derivatives. *Experiments in Fluids*, *57*(9), 1-22.

Sciacchitano, A., Leclaire, B., & Schröder, A. (2021, 08). Main results of the first Lagrangian particle tracking challenge. *14th International Symposium on Particle Image Velocimetry*, *1*.

Takehara, K., Adrian, R. J., Etoh, G. T., & Christensen, K. T. (2000). A kalman tracker for super-resolution piv. *Experiments in Fluids*.

van Oudheusden, B. W. (2013, jan). PIV-based pressure measurement. *Measurement Science and Technology*, *24*(3), 032001.

Vennell, R., & Beatson, R. (2009). A divergence-free spatial interpolator for large sparse velocity data sets. *Journal of Geophysical Research: Oceans*, *114*(C10).

Yang, Y., & Heitz, D. (2021). Kernelized lagrangian particle tracking. *Experiments in Fluids*, *62*.