Stochastic Unrolled Proximal Point Algorithm for linear image inverse problems

Brandon Le Bon, Mikaël Le Pendu, Christine Guillemot

Stochastic Unrolled Proximal Point Algorithm for linear image inverse problems

1st Brandon Le Bon
INRIA Rennes - Bretagne Atlantique
Rennes, France
brandon.le-bon@inria.fr

2nd Mikaël Le Pendu
INTERDIGITAL
Cesson-Sévigné, France
mikael.lependu@interdigital.com

3rd Christine Guillemot
INRIA Rennes - Bretagne Atlantique
Rennes, France
christine.guillemot@inria.fr

***Abstract*—Unrolled optimization methods have emerged as a way to combine classical iterative optimization techniques with learned priors to efficiently solve image restoration problems. However, learning the regularization prior along the unrolled iterations requires intensive memory usage due to the deep explicit backpropagation, hence making the number of unrolled iterations usually small in practice. Inspired by deep equilibrium models, unrolling models with implicit backpropagation have been considered for solving this issue. Nevertheless, while these methods yield good restoration quality with reduced memory usage, the theory of implicit backpropagation relies on the knowledge of the fixed point of the function to optimize, usually unknown and estimated by iterating until convergence. Therefore, these methods require intensive computation time to ensure a stable backpropagation. In this paper, we present an unrolled Proximal Point Algorithm method, where the end-to-end optimization problem is re-defined as a per unrolled iteration optimization problem. We prove that the proposed optimization strategy is memory-efficient and applicable for any number of computed unrolled iteration. We empirically show that our method achieves state-of-the-art image restoration quality.**

***Index Terms*—unrolling, stochastic, inverse problems, Proximal Point Algorithm, ADMM**

I. Introduction

Image restoration problems are formalized as the recovery of an image given noisy or incomplete observations. The problem being ill-posed, a common strategy consists in introducing some prior knowledge on the typical images we attempt to restore, which helps restricting the class of admissible solutions. These priors have evolved from handcrafted priors such as sparse priors [1] to learned priors. Learned priors have been introduced as regularization constraints in iterative optimization methods, e.g., the gradient descent method [2] and the Alternating Direction Method of Multipliers (ADMM) [3].

With the rise of deep learning, unrolled optimization algorithms have emerged as a way to combine iterative optimization and deep learning. The network is trained end-to-end within the iterative algorithm, hence in a way which takes into account the degradation operator. Performing a fixed number of unrolled iterations yields optimized results for a given inverse problem. Several optimization algorithms have been unrolled, e.g., the Iterative Shrinkage Thresholding Algorithm (ISTA) [4], the gradient descent [5], the proximal gradient [6], the Half-Quadratic Splitting (HQS) [7] and the Alternating Direction Method of Multipliers (ADMM) [8].

The end-to-end network optimization in unrolled methods requires backpropagating gradients through the whole iterative scheme. Thus, the memory usage of this explicit backpropagation scales linearly with the number of unrolled iterations, hence limiting the number of iterations that can be used in practice. Deep Equilibrium based approaches [9]–[11] introduced implicit backpropagations to reduce the memory usage of the backpropagation. Contrary to an explicit backpropagation, the memory requirements for an implicit backpropagation do not depend on the number of unrolled iterations. However, the theory of the implicit backpropagation relies on the fixed point of the function to optimize, usually estimated by iterating until idempotence, leading to a considerable computation time.

In this paper, we propose a novel memory-efficient unrolled optimization algorithm based on the ADMM. Each unrolled ADMM iteration is re-defined as a proximal mapping, by exploiting the fact that the ADMM is an application of the Proximal Point Algorithm [12], [13]. A stochastic training of the learned weights is performed by considering per-iteration optimization problems. This optimization strategy is thus independent of the number of computed unrolled iterations and is mathematically justified even if the fixed point is not estimated, hence making the training possible for any arbitrary number of unrolled iterations. We assess the method for deblurring and super-resolution in comparison with several recent unrolled optimization methods and task-specific deep methods. We show that the proposed method achieves state-of-the-art restoration performances for every inverse problem.

II. Background

A. Problem statement

Let the image acquisition process be represented by the following linear system:

$$y = Ax + \epsilon, \quad (1)$$

where y and x are respectively the observed and the original images, A is a matrix representing the acquisition process and ϵ is a Gaussian noise with standard deviation σ_0. The problem of restoring the original image x from the measurements y is usually ill-posed due to the ill-conditioned matrix A. We thus

This work was supported by the French ANR research agency in the context of the artificial intelligence project DeepCIM.

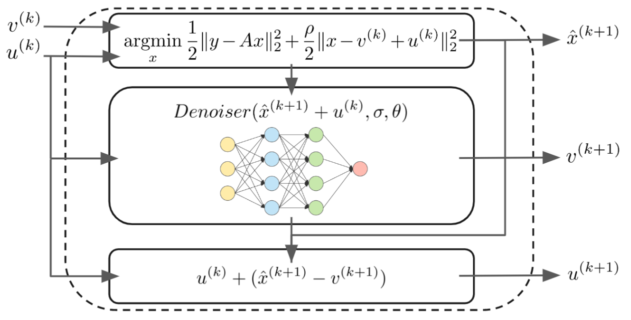
$v^{(k)}$
$u^{(k)}$
$\underset{x}{\operatorname{argmin}} \frac{1}{2}\|y-Ax\|_2^2+\frac{\rho}{2}\|x-v^{(k)}+u^{(k)}\|_2^2$
$\hat{x}^{(k+1)}$
$Denoiser(\hat{x}^{(k+1)}+u^{(k)},\sigma,\theta)$
$v^{(k+1)}$
$u^{(k)}+(\hat{x}^{(k+1)}-v^{(k+1)})$
$u^{(k+1)}$

Fig. 1: Unrolled ADMM optimization with a learned network acting as a Gaussian denoiser.

need to introduce some prior on the signal that we are trying to recover, via a regularization constraint $r(x)$. The problem of recovering an estimated image $\hat{x}$ is therefore re-written as:

$$\hat{x} = \underset{x}{\operatorname{argmin}} \frac{1}{2}\|y - Ax\|_2^2 + \sigma_0^2 r(x), \quad (2)$$

The solution $\hat{x}$ is thus searched by minimizing the objective function in Eq. (2) composed of two terms: a data-fit term $\frac{1}{2}\|y - Ax\|_2^2$ and a regularization term $r(x)$.

B. *Unrolled optimization with learned priors*

First methods for solving inverse problems posed in Eq. (2) have been using iterative optimization techniques, such as the gradient descent [2] and the Alternating Direction Method of Multipliers (ADMM) [3]. Several iterative optimization algorithms have been unrolled, introducing a learned complex prior to regularize the optimization problem. In this paper, we mainly focus on the unrolled ADMM, depicted in Fig. 1. An iteration of the ADMM consists of the following three steps:

$$\hat{x}^{(k+1)} = \underset{x}{\operatorname{argmin}} \frac{1}{2}\|y - Ax\|_2^2 + \frac{\rho}{2}\|x - v^{(k)} + u^{(k)}\|_2^2, \quad (3)$$

$$v^{(k+1)} = \underset{v}{\operatorname{argmin}} \frac{\rho}{2}\|v - \hat{x}^{(k+1)} - u^{(k)}\|_2^2 + \sigma_0^2 r(v), \quad (4)$$

$$u^{(k+1)} = u^{(k)} + (\hat{x}^{(k+1)} - v^{(k+1)}), \quad (5)$$

where $\hat{x}$ and v are the proximal operators of respectively the data-fit term and the regularizer in Eq. (2), and u is the dual variable. In a nutshell, each iteration of the ADMM improves the current estimate by alternatively minimizing the proximal operators of the data-fit term and the regularization term. After several iterations, the algorithm converges to a solution which minimizes both terms. We can notice that the computation of $v^{(k+1)}$ can be re-written as a Gaussian denoising step:

$$v^{(k+1)} = \underset{v}{\operatorname{argmin}} \sigma^2 r(v) + \frac{1}{2}\|v - \hat{x}^{(k+1)} - u^{(k)}\|_2^2, \quad (6)$$

with $\sigma = \frac{\sigma_0}{\sqrt{\rho}}$. Eq. (6) is then a special case of Eq. (2), where A is the identity matrix and the Gaussian noise standard deviation is equal to σ. The proximal operator in Eq. (4) can thus be replaced by a network trained as a Gaussian denoiser for noise standard deviation σ:

$$v^{(k+1)} = Denoiser(\hat{x}^{(k+1)} + u^{(k)}, \sigma, \theta). \quad (7)$$

In this case, the network is pre-trained independently of the inverse problem in hand. Unrolling the ADMM aims to fine-tune the learned denoiser for a specific inverse problem. The denoiser is thus learned end-to-end within the ADMM, such that the unrolled ADMM best reconstructs the original image.

C. *Unrolling with implicit backpropagation*

While classical iterative methods iterate until convergence, unrolled optimization algorithms consider a fixed and small number of iterations, to limit the memory usage of the end-to-end explicit backpropagation at training time. Deep Equilibrium (DEQ) [9], [10] and Jacobian-Free Backpropagation Implicit (JFBI) [11] Networks have been introduced using implicit backpropagations. The differentiation is re-written so that the backpropagation is done only over the last iteration, hence making the memory usage independent of the number of unrolled iterations. However, implicit backpropagation heavily relies on the assumption that the fixed point of the iterative algorithm exists and is reached in practice to compute the gradients accurately. The strategy to compute the fixed point is to iterate until idempotence, estimated with an approximation error value ϵ. While a high value of ϵ may lead to instability due to a wrong estimation of the fixed point, a low value increases the number of iterations, i.e., the computation time.

III. Stochastic Unrolled Proximal Point Algorithm

Both explicit and implicit backpropagations aim to optimize the network parameters θ in order to fit the output of the unrolled iterative algorithm to the groundtruth. Instead, we define the unrolled ADMM as an unrolled Proximal Point Algorithm (PPA) where the backpropagation uses the intermediate unrolled iterations independently of each other.

A. *Unrolled Proximal Point Algorithm*

The Proximal Point Algorithm (PPA) introduced in [14] iteratively computes the proximal point:

$$\hat{x}^{(k+1)} = \operatorname{prox}_g(\hat{x}^{(k)}), \quad (8)$$

$$\text{with} \quad \operatorname{prox}_g(\hat{x}^{(k)}) = \underset{x}{\operatorname{argmin}}\, g(x) + \frac{1}{2}\|x - \hat{x}^{(k)}\|_2^2. \quad (9)$$

When g is convex, the PPA converges to the minimum of g. Eckstein et al. [12] demonstrated that methods applying the Douglas—Rachford splitting algorithm, such as the ADMM [13], are special cases of the PPA. Let us now write the ADMM parameterized with θ as the following series:

$$\hat{x}^{(k+1)} = f(\hat{x}^{(k)}, \theta), \quad (10)$$

where, f represents one iteration of the ADMM algorithm, i.e., f performs Eqs. (3), (4), (5), with Eq. (4) replaced by a pre-trained Gaussian denoiser as in Eq. (7). The ADMM being a special case of the PPA, there exists a scalar-valued function g such that the ADMM iteration $f(\hat{x}^{(k)}, \theta)$ is expressed as:

$$f(\hat{x}^{(k)}, \theta) = \underset{x}{\operatorname{argmin}}\, g(x, \theta) + \frac{1}{2}\|x - \hat{x}^{(k)}\|_2^2 = \operatorname{prox}_{g(.,\theta)}(\hat{x}^{(k)}). \quad (11)$$

Here, each ADMM iteration k aims to recover the solution of the proximal operator of g. To learn the parameters θ of the unrolled ADMM, we thus define an optimization problem for each iteration k in order to fit $f(\hat{x}^{(k)}, \theta)$ to $\text{prox}_{g(.,\theta)}(\hat{x}^{(k)})$, with g defined as a handcrafted convex function. We will show in Section III-C that the end-to-end optimization problem of the unrolled ADMM can then be reduced to a set of per-iteration optimization problems.

B. Definition of g

With g convex, $\text{prox}_{g(.,\theta)}(\hat{x}^{(k)})$ reduces the distance between the estimate $\hat{x}^{(k)}$ and the fixed point of the ADMM algorithm. Since the fixed point depends on θ, we will note it $\hat{x}^*_\theta$. Thus, we define $g(\hat{x}, \theta)$ as the weighted squared ℓ_2-norm between any image $\hat{x}$ and the fixed point $\hat{x}^*_\theta$ of the ADMM:

$$g(\hat{x}, \theta) = \frac{1}{2\lambda}\|\hat{x}^*_\theta - \hat{x}\|_2^2, \tag{12}$$

with $\lambda > 0$. The proximal operator of g has thus a well-known closed-form:

$$\text{prox}_{g(.,\theta)}(\hat{x}) = \frac{\hat{x}^*_\theta + \lambda\hat{x}}{1+\lambda}. \tag{13}$$

From this definition, g is convex with respect to $\hat{x}$ and one can easily verify that iterative applications of $\text{prox}_{g(.,\theta)}$ effectively converges towards the fixed point $\hat{x}^*_\theta$, with a convergence rate controlled by λ.

C. Loss function

Unrolled optimizations aim to optimize θ in order to minimize the difference between the output $\hat{x}^*_\theta$ and the groundtruth image x_{gt}. In addition, we propose to fit $f(\hat{x}, \theta)$ to $\text{prox}_{g(.,\theta)}(\hat{x})$ for any $\hat{x}$. We thus want to learn the parameters θ to approximate:

$$\begin{cases} \hat{x}^*_\theta = x_{gt}, & (14) \\ f(\hat{x}, \theta) = \text{prox}_{g(.,\theta)}(\hat{x}), \quad \forall\hat{x}. & (15) \end{cases}$$

According to Eq. (13), this is equivalent to:

$$\begin{cases} \hat{x}^*_\theta = x_{gt}, & (16) \\ \hat{x}^*_\theta = (1+\lambda)f(\hat{x}, \theta) - \lambda\hat{x}, \quad \forall\hat{x} & (17) \end{cases}$$

$$\begin{cases} \hat{x}^*_\theta = x_{gt}, & (18) \\ x_{gt} = (1+\lambda)f(\hat{x}, \theta) - \lambda\hat{x}, \quad \forall\hat{x} & (19) \end{cases}$$

We can notice that Eq. (18) is equivalent to a particular case of Eq. (19) where $\hat{x} = \hat{x}^*_\theta$, since $f(\hat{x}^*_\theta, \theta) = \hat{x}^*_\theta$ by definition of the fixed point. The system can thus be expressed only with Eq. (19). The end-to-end optimization of the unrolled ADMM is thus reduced to a set of independent optimization problems. However, optimizing θ for any possible images $\hat{x}$ is impractical, since it would require integrating the loss over the space of images. Instead, we consider only the images in the optimization path, i.e., the intermediate estimates of the unrolled algorithm, i.e., $\hat{x}^{(k)}, \forall k \in \{0, ..., K\}$. The full loss function $\mathcal{L}$ is then written as follows:

$$\mathcal{L}(\theta, \lambda, \hat{x}^{(0)}, ..., \hat{x}^{(K)}) = \frac{1}{K}\sum_{k=0}^{K} l(\theta, \lambda, \hat{x}^{(k)}) \tag{20}$$

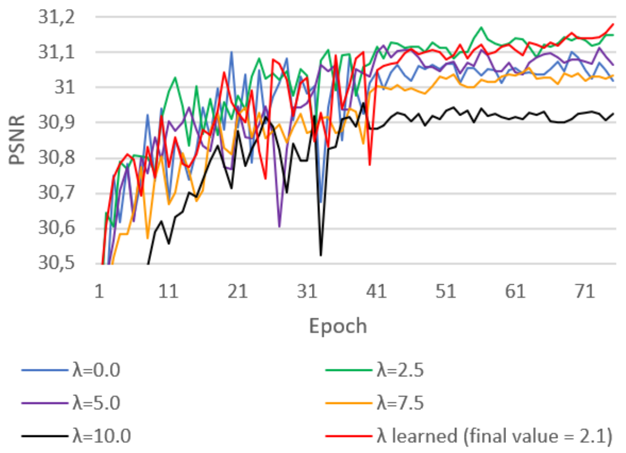

Fig. 2: Average validation PSNR per epoch during training with different values of λ in Eq. (13), for image super-resolution (bicubic x2 without antialiasing).

with:

$$l(\theta, \lambda, \hat{x}^{(k)}) = \|(1+\lambda)f(\hat{x}^{(k)}, \theta) - \lambda\hat{x}^{(k)} - x_{gt}\|_2^2. \tag{21}$$

Each term of the sum in Eq. (20) considers $\hat{x}^{(k)}$ as an input. The computation of $\hat{x}^{(k)}$ is then not taken into account for the backpropagation, which is essential to keep it shallow. This will be further discussed in Section III-D.

The latter loss $\mathcal{L}$ is however dependent on the design of the convergence rate λ in Eq. (13). Its value need to be optimized for both the estimation of the proximal mapping and the image restoration problems, i.e., such that it minimizes the loss l in Eq. (21) for any input $\hat{x}^{(k)}$. We thus propose to learn λ along with the weights θ. to automatically find its best value. To illustrate the importance of correctly setting λ, we trained the SUPPA with different values of λ on the same task. As shown in Fig. 2, the value of λ drastically impacts the image reconstruction quality. Furthermore, learning it offers the best performances.

D. Stochastic unrolled iteration learning

Each term of the loss function $\mathcal{L}$ in Eq. (20) is associated to a specific independent iteration k, with $\hat{x}^{(k)}$ considered as an input. Instead of considering all the K iterations at each training optimization step, we propose a stochastic selection of a small subset of iterations $\mathcal{I} \subset \{1, .., K\}$. The differentiation of the loss $\mathcal{L}$ w.r.t θ can then be written as follows:

$$\frac{\partial\mathcal{L}}{\partial\theta} = \frac{1}{\text{card}(\mathcal{I})}\sum_{i\in\mathcal{I}} \frac{\partial f(\hat{x}^{(i)}, \theta)}{\partial\theta}^T \frac{\partial l}{\partial f(\hat{x}^{(i)}, \theta)}. \tag{22}$$

The differentiation of $\mathcal{L}$ w.r.t. λ can be simplified as:

$$\frac{\partial\mathcal{L}}{\partial\lambda} = \frac{1}{\text{card}(\mathcal{I})}\sum_{i\in\mathcal{I}} \frac{\partial l}{\partial\lambda}. \tag{23}$$

The memory usage of the backpropagation is thus independent of the number of unrolled iteration, and is instead only dependent on the size of the subset of selected iterations $\mathcal{I}$.

In order to favor high restoration quality, we always include the last computed iteration in the set $\mathcal{I}$. Hence, we propose to use 2 iterations ($\text{card}(\mathcal{I}) = 2$) per backpropagation: the last iteration K and a randomly selected iteration $k < K$. We have not observed further benefit in increasing the number of randomly selected iterations. The SUPPA has thus the same memory usage advantage as the implicit unrolled methods.

Algorithm 1 summarizes the different steps of the proposed algorithm. Our PyTorch implementation of the method is available at: https://github.com/BrandonLeBon/SUPPA

Algorithm 1 Proposed Unrolled Proximal Point algorithm

```
1:  initialize $\theta$, $K$, $\lambda$, $\epsilon$
2:  for each epoch do
3:      for each batch do
4:          $\hat{x}^{(0)} \leftarrow$ input batch
5:          $x_{gt} \leftarrow$ groundtruth batch
6:          $\mathcal{I} \leftarrow$ random subset of $\{1, ..., K\}$
7:          $loss \leftarrow 0$
8:          $k \leftarrow 1$
9:          $\hat{x}^{(k)} \leftarrow f(\hat{x}^{(0)}, \theta)$
10:         while $k < K$ and $\|\hat{x}^{(k-1)} - \hat{x}^{(k)}\| > \epsilon$ do
11:             if $k \in \mathcal{I}$ then
12:                 update $loss$ with (20)
13:             $\hat{x}^{(k+1)} \leftarrow f(\hat{x}^{(k)}, \theta)$
14:             $k \leftarrow k + 1$
15:         update $\theta$ and $\lambda$ with (22) and (23)
```

IV. Experimental results

A. Experimental setup

We evaluate the performances of our method on the super-resolution and deblurring inverse problems, against state-of-the-art unrolled and task-specific deep methods. Additional visual results are reported on the project web page (http://clim.inria.fr/DeepCIM/SUPPA/index.html).

Unrolled optimization methods: the selected reference unrolled optimization methods are (i) the explicit unrolled optimization with deep prior [5] (ii) the Deep Equilibrium architectures for inverse problems (DEQ) [10] (iii) an adaptation of the DEQ using the Jacobian-Free differentiation [11], named Jacobian-Free Backpropagation Implicit Unrolled (JFBI). Networks parameters are shared between all unrolled iterations and are pre-trained in order to initialize the proximal operator of the regularizer (4) for Gaussian noise removal as presented in [16] with the DRUnet denoising architecture. The explicit unrolled ADMM uses 6 iterations. As stopping criteria for the DEQ, the JFBI, and the SUPPA, we use a maximum of 50 iterations, with an idempotence estimated with a precision of $\epsilon = 10^{-3}$ as in [10].

State-of-the-art task-specific deep methods: the efficient task-specific deep methods considered are (i) the RCAN method [17] and MoG-DUN method [18] for the super-resolution (ii) the method of Dong et al. [19] for deblurring.

Training parameters: all the above methods have been retrained. We used 75 epochs, a batch size of 16, a learning rate of 10^{-5}. We also used an additional learning rate of 10^{-1} for the convergence rate λ in the proposed method. **Super-resolution:** we consider three scales: x2, x3 and x4. The low resolution images are generated with a bicubic downsampling without anti-aliasing. As initialisation, we perform a bicubic interpolation on the downsampled image. As in [18], we used the DIV2K dataset [20] [21] with random patches of size 48x48 for training, and Set5 [22], Set14 [23] and BSDS100 [15] datasets for testing. We use the closed-form presented in [16] to solve Eq. (3).

Deblurring: we consider a Gaussian blur kernel of parameters $\sigma = 2$, $size = 2 * \sigma$ and a 1% noise level. As in [19], we randomly cropped 256x256 patches from the Waterloo Exploration dataset [24] for training and we used the BSDS500 dataset [15] for testing. The exact solution $\hat{x}$ in Eq. (3) was computed using the same closed-form as for the super-resolution with a scaling factor equal to 1.

B. Reconstruction performances

Reconstruction performances are evaluated with the mean PSNR per dataset on the RGB channels. Note that the PSNR values for RCAN and MoG-DUN are significantly different from the original papers, since these are computed on the Y channel of the YCrCb color space. Super-resolution and deblurring results are respectively in Table I, and II. As shown in both tables, our method performs as well as the best unrolled methods, and outperforms the task-specific deep methods for the two tested inverse problems.

TABLE I: Superresolution results (average PSNR)

	Scale	Set5	Set14	BSDS100
ZHANG et al.	x2	34.30 dB	30.13 dB	28.81 dB
Ning et al.	x2	34.59 dB	30.35 dB	30.02 dB
Unrolled	x2	34.87 dB	30.42 dB	30.04 dB
DEQ	x2	34.75 dB	30.28 dB	29.93 dB
JFBI	x2	34.96 dB	30.61 dB	30.09 dB
SUPPA	x2	34.92 dB	30.54 dB	30.08 dB
Zhang et al.	x3	29.60 dB	25.67 dB	25.91 dB
Ning et al.	x3	29.63 dB	25.65 dB	25.93 dB
Unrolled	x3	30.06 dB	26.03 dB	26.33 dB
DEQ	x3	29.94 dB	25.95 dB	26.27 dB
JFBI	x3	29.97 dB	25.90 dB	26.31 dB
SUPPA	x3	29.95 dB	25.90 dB	26.30 dB
Zhang et al.	x4	27.50 dB	23.80 dB	24.67 dB
Ning et al.	x4	27.77 dB	24.28 dB	24.79 dB
Unrolled	x4	28.27 dB	24.63 dB	25.21 dB
DEQ	x4	27.21 dB	24.17 dB	24.44 dB
JFBI	x4	28.34 dB	24.63 dB	25.16 dB
SUPPA	x4	28.20 dB	24.63 dB	25.12 dB

C. Convergence of the unrolled methods

Fig. 3 illustrates the convergence of the unrolled methods. First, we can notice that the explicit unrolled method does

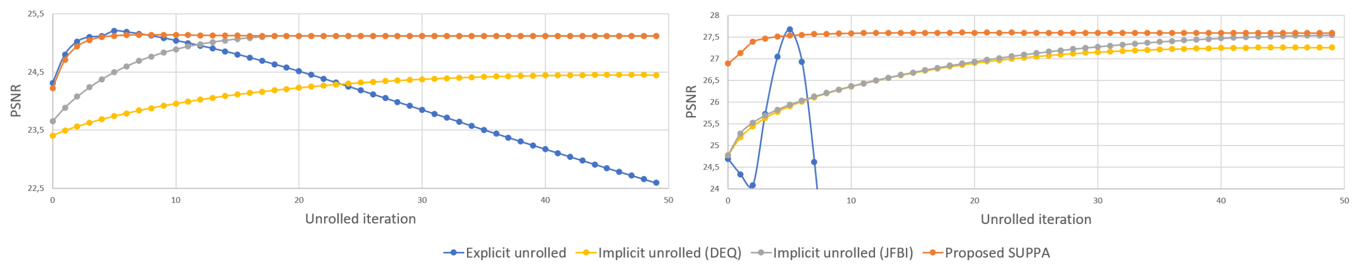

Fig. 3: Average PSNR per unrolled iteration for the different unrolled methods and for (left) super-resolution (x4) on the BSDS100 dataset [15], (right) Gaussian de-blurring on the BSDS500 dataset [15].

TABLE II: Deblurring results (average PSNR)

	BSDS500
Dong et al.	27.28 dB
Unrolled	27.68 dB
DEQ	27.25 dB
JFBI	27.55 dB
SUPPA	27.60 dB

not converge, since it is optimized for a specific number of iterations. Both the implicit backpropagation methods and SUPPA tend to converge, which is an expected behaviour of the ADMM algorithm. Furthermore, SUPPA converges faster than the JFBI and DEQ methods. A possible explanation is that in SUPPA, we explicitly introduce a convergence rate parameter λ allowing a control over the convergence speed, while JFBI and DEQ tend to use the maximum number of iterations allowed in the training (50 in our experiments).

V. Conclusion

In this paper, we proposed a novel unrolled learning-based iterative optimization for solving image restoration problems. The proposed per-iteration optimization strategy permits to significantly reduce the computational burden of training unrolled optimization algorithms, in terms of both memory usage and computation time, while yielding a reconstruction quality on par with the state-of-the-art for the tested applications (i.e., super-resolution and deblurring), considering both other recent unrolled and task-specific deep methods.

References

[1] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM journal on imaging sciences*, vol. 2, no. 1, pp. 183–202, 2009.
[2] C. Lemaréchal, "Cauchy and the gradient method," *Doc Math Extra*, vol. 251, no. 254, p. 10, 2012.
[3] S. Boyd, N. Parikh, and E. Chu, *Distributed optimization and statistical learning via the alternating direction method of multipliers*. Now Publishers Inc, 2011.
[4] K. Gregor and Y. LeCun, "Learning fast approximations of sparse coding," in *International Conf. on international conference on machine learning*, 2010, pp. 399–406.
[5] S. Diamond, V. Sitzmann, F. Heide, and G. Wetzstein, "Unrolled optimization with deep priors," *arXiv preprint arXiv:1705.08041*, 2017.
[6] M. Mardani, Q. Sun, D. Donoho, V. Papyan, H. Monajemi, S. Vasanawala, and J. Pauly, "Neural proximal gradient descent for compressive imaging," *Advances in Neural Information Processing Systems*, vol. 31, 2018.
[7] U. Schmidt and S. Roth, "Shrinkage fields for effective image restoration," in *IEEE Conf. on computer vision and pattern recognition*, 2014, pp. 2774–2781.
[8] Y. Yang, J. Sun, H. Li, and Z. Xu, "Deep admm-net for compressive sensing mri," in *International Conf. on neural information processing systems*, 2016, pp. 10–18.
[9] S. Bai, J. Z. Kolter, and V. Koltun, "Deep equilibrium models," *Advances in Neural Information Processing Systems*, vol. 32, 2019.
[10] D. Gilton, G. Ongie, and R. Willett, "Deep equilibrium architectures for inverse problems in imaging," *IEEE Transactions on Computational Imaging*, vol. 7, pp. 1123–1133, 2021.
[11] S. W. Fung, H. Heaton, Q. Li, D. McKenzie, S. Osher, and W. Yin, "Jfb: Jacobian-free backpropagation for implicit models," *AAAI Conf. on Artificial Intelligence*, 2022.
[12] J. Eckstein and D. P. Bertsekas, "On the douglas—rachford splitting method and the proximal point algorithm for maximal monotone operators," *Mathematical Programming*, vol. 55, no. 1, pp. 293–318, 1992.
[13] D. Gabay, "Chapter ix applications of the method of multipliers to variational inequalities," in *Studies in mathematics and its applications*. Elsevier, 1983, vol. 15, pp. 299–331.
[14] B. Martinet, "Regularisation, d'inéquations variationelles par approximations succesives," *Revue Francaise d'informatique et de Recherche operationelle*, 1970.
[15] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 5, pp. 898–916, May 2011. [Online]. Available: http://dx.doi.org/10.1109/TPAMI.2010.161
[16] K. Zhang, Y. Li, W. Zuo, L. Zhang, L. Van Gool, and R. Timofte, "Plug-and-play image restoration with deep denoiser prior," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2021.
[17] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu, "Image super-resolution using very deep residual channel attention networks," in *European conf. on computer vision (ECCV)*, 2018, pp. 286–301.
[18] Q. Ning, W. Dong, G. Shi, L. Li, and X. Li, "Accurate and lightweight image super-resolution with model-guided deep unfolding network," *IEEE Journal of Selected Topics in Signal Processing*, 2020.
[19] J. Dong, S. Roth, and B. Schiele, "Deep wiener deconvolution: Wiener meets deep learning for image deblurring," *Advances in Neural Information Processing Systems*, vol. 33, 2020.
[20] E. Agustsson and R. Timofte, "Ntire 2017 challenge on single image super-resolution: Dataset and study," in *IEEE Conf. on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 126–135.
[21] A. Ignatov, R. Timofte *et al.*, "Pirm challenge on perceptual image enhancement on smartphones: report," in *European Conf. on Computer Vision (ECCV) Workshops*, January 2019.
[22] M. Bevilacqua, A. Roumy, C. Guillemot, and M. L. Alberi-Morel, "Low-complexity single-image super-resolution based on nonnegative neighbor embedding," 2012.
[23] R. Zeyde, M. Elad, and M. Protter, "On single image scale-up using sparse-representations," in *International Conf. on curves and surfaces*. Springer, 2010, pp. 711–730.
[24] K. Ma, Z. Duanmu, Q. Wu, Z. Wang, H. Yong, H. Li, and L. Zhang, "Waterloo Exploration Database: New challenges for image quality assessment models," *IEEE Trans. on Image Processing*, vol. 26, pp. 1004–1016, Feb. 2017.