Learning the Image Prior by Unrolling an Optimization Method

Silvia Bonettini*, Giorgia Franchini†, Danilo Pezzi‡ and Marco Prato§
Department of Informatics, Physics and Mathematics, University of Modena and Reggio Emilia
Modena, Italy
Email: *silvia.bonettini@unimore.it, †giorgia.franchini@unimore.it, ‡danilo.pezzi@unimore.it, §marco.prato@unimore.it

***Abstract*—Nowadays neural networks are omnipresent thanks to the amazing adaptability they possess, despite their poor interpretability and the difficulties they give when manipulating the parameters. On the other side, we have the classical variational approach, where the restoration is obtained as the solution of a given optimization problem. The bilevel approach is connected to both approaches and consists first in devising a parametric formulation of the variational problem, then in optimizing these parameters with respect to a given dataset of training data. In this work we analyze the classical bilevel approach in combination with unrolling techniques, where the parameters of the variational problem are trained with respect to the results obtained after a fixed number of iterations of an optimization method applied to it. This results in a large scale optimization problem which can be solved by means of stochastic methods; as we observed in our numerical experiments, the stochastic approach can produce medium accuracy results in very few epochs. Moreover, our experiments also show that the unrolling approach leads to results which are comparable with those of the original bilevel method in terms of accuracy.**

***Index Terms*—bilevel optimization, unfolding, image denoising, sparse model, stochastic gradient, machine learning, Green AI.**

I. Introduction

Image denoising is a standard type of linear inverse problem, in which the observed data f is a corrupted version of the original object g by some kind of (usually) additive noise, i.e., $f = g + \eta$. Since, by nature, the noise is the realization of a collection of i.i.d. random variables, one needs to devise a scheme to retrieve only a reconstruction of the unknown image g, starting from f.

One popular way of solving inverse problems is employing a variational model, which requires computing the minimum

$$\operatorname*{argmin}_u E(u) \equiv D(u; f) + \rho R(u),$$

where E represents a suitable energy functional composed by the sum of two terms, which are balanced by the coefficient ρ. The selection of the data-fidelity D term is typically motivated by probabilistic considerations on the nature of the noise: if η follows a Gaussian distribution, then an appropriate distance is the ℓ_2-norm, while if it is obtained by a Poisson distribution, a better choice would be the Kullback-Leibler divergence [1]. The regularization function R, also called image prior, is used to incorporate additional information in the reconstruction. In the regularization field, one of the most famous models is the one proposed by Rudin, Osher and Fatemi in [22] where sparsity is imposed on the gradient of the image in order to preserve its edges.

The selection of the regularization functional is crucial, and in the last years several data-driven approaches have been developed to help its effective design [5]. As described by M. Moeller and D. Cremers in [18], various deep and machine learning techniques are being utilized to find the best parameters of both variational models as well as of the optimization algorithms meant to solve them. In this context, neural networks represent a great tool with a multitude of applications and uses, at the price of interpretability. On the other hand, bilevel optimization strategies (e.g. [4], [20] and [14]), while being more difficult to handle, do not lose sight of the underlying energy functional.

In this work we focus on the latter approach to learn a sparsity inducing image prior. Even if it does not require a massive amount of samples, the training phase can still be significantly time and energy consuming, that is why we use stochastic algorithms to solve the upper level loss minimization.

In the literature, the importance of setting the hyperparameters connected to the stochastic optimiser, in particular steplength and sample size, is well known. By referring to hyperparameters, we are meaning all those parameters that are not tuned during the training phase. This process can be particularly computationally expensive, which is why several approaches have been proposed in the literature for the automatic setting of these hyperparameters in the stochastic field [9], [10], including with reduced variance methods [11]. In all of these approaches, great benefit is derived from the automatic setting of hyperparameters, both from the point of view of performance and resource consumption.

Nowadays, it is necessary to remark the importance of not wasting computational resources, in an optic of Green AI (Artificial Intelligence). In the lower level problem, we unroll, for a small number of iterations, a gradient descent scheme. This choice leads to several advantages, since we circumvent numerical issues, avoid the need of exactly solving a minimization problem, and automatically learn the steplength sequence.

II. The Problem

In this section we discuss the work of Y. Chen, R. Ranftl and T. Pock in [6] and our contributions to improve the overall performance of the model. The focus is on the regularizer of the energy functional

$$E(u) = D(u; f) + \sum_{i=1}^{N_f} \alpha_i \phi(a_i * u), \qquad u \in \mathbb{R}^n$$

where each a_i is a convolution kernel, N_f is the number of filters, α_i is the weight relative to the i-th filter and $\phi \in \mathcal{C}^2(\mathbb{R})$ is a sparsity inducing function, applied pixel-wise to the resulting convolution. The data fitting term D is taken as $\|u - f\|_2^2$ since we work with images corrupted by additive Gaussian noise. The regularization parameter is implicitly included in the filter weights $\{\alpha_i\}_{i=1,\dots,N_f}$.

For the rest of this work, the convolution is going to be written as a matrix-vector dot product, with A_i denoting the matrix constructed from the kernel a_i. Moreover, each A_i is not left completely free, but instead is taken as a linear combination of a basis, denoted with $\{B_j\}_{j=1,\dots,m}$, i.e., $A_i = \sum_{j=1}^{m} \beta_{ij} B_j$. The model parameters $\alpha = (\alpha_1, \dots, \alpha_{N_f})$ and $\beta = (\beta_{11}, \dots, \beta_{N_f m})$ are chosen through a classical supervised learning scheme with a ℓ_2-norm loss function. Let $\mathcal{D} = \{(f_s, g_s)\}$ be the dataset, with S elements, where every element is a couple composed of a noisy observation f_s and its respective ground truth g_s, then the result is a bilevel optimization problem:

$$\begin{cases} \underset{\alpha \geq 0, \beta}{\operatorname{argmin}} \mathcal{L}(\alpha, \beta) \equiv \sum_{s=1}^{S} \|u_s^*(\alpha, \beta) - g_s\|_2^2 \\ \text{s. to } \; u_s^* = \underset{u}{\operatorname{argmin}} \, E(u, \alpha, \beta) \equiv D(u; f_s) + R(u, \alpha, \beta). \end{cases}$$

From now on, the parameters are condensed in one vector variable $\theta = (\alpha_1, \dots, \alpha_{N_f}, \beta_{11}, \dots, \beta_{N_f m}) \in \mathbb{R}^p$ for simplicity. To solve such problem, one needs to find each minimum u_s in order to compute the loss gradient either through implicit differentiation or by applying the first-order optimality conditions. Algorithm 1 sums up the procedure to obtain the solution to the upper level problem.

Algorithm 1:

Data: $\mathcal{D} = \{(f_s, g_s)\}_{s=1}^{S}$, maxIter, $\theta^{(0)}$
for $i = 0, \dots, \text{maxIter}$ **do**
  for $s = 1, \dots, S$ **do**
    Compute $u_s^* = \underset{u}{\operatorname{argmin}} \, E(u, \theta)$.
  end
  Compute $\nabla_\theta \mathcal{L}$ at $\theta^{(i)}$.
  Update the parameters to $\theta^{(i+1)}$ by means of $\nabla_\theta \mathcal{L}(\theta^{(i)})$.
end

This approach is indeed able to find a good set of parameters, but as implemented in [6] it does have some issues. First, in order to obtain $\nabla_\theta \mathcal{L}$ one would need to get the inverse of $\nabla_{uu}^2 E$, i.e. the hessian matrix of the energy functional, but nothing can be assumed on such matrix, in particular non-singularity. Also, since the minimum of the energy functional does not have a closed form, it needs to be approximated, this means that further error is introduced in the loss gradient computation.

Our Contribution

We work on both the upper level and the lower level problems. For the loss minimization, due to its nature, we study the performance of stochastic optimizers, which make the training phase faster without a significant decrease in the performance of the resulting model. As for the lower level problem, we replace it with the unrolling of a gradient descent algorithm, since, in this way, it does admit an explicit solution, and the loss gradient can be computed exactly. Also, we avoid inverting the hessian matrix, which helps in speeding up the training of the model, and can be even more important if one wants to use high dimensional images.

III. Unrolling

In this section we delve deeper in the details of the unrolling of an optimization algorithm. Such algorithm needs to be differentiable, i.e., its general iteration needs to be differentiable, otherwise we would not be able to compute the gradient loss through the chain rule. Specifically, we use a simple gradient descent scheme applied to the energy functional E, which means that the generic new iterate is

$$u^{(k+1)} = u^{(k)} - \lambda_k \nabla_u E(u^{(k)}).$$

where λ_k is the steplength. The loss function is computed at the image that it is obtained after applying K iterations of the gradient descent starting from the noisy observed image. By doing this, the loss gradient computation becomes exact since we are not approximating the solution of the lower level problem, but rather we are trying to make the best use of a predetermined number of iterations. It also gives us the possibility to learn the steplength sequence $\{\lambda_k\}_k$ at the same time. Moreover, an additional advantage is that once the model is trained, if we want to find the reconstruction to a new noisy observation $\bar{f}$, we only need to apply K steps of the gradient descent scheme, with the learned steplengths, to the energy functional relative to $\bar{f}$. Note that we do not impose any constraint other than non-negativity on λ, which has the effect of not giving any guarantee on the decrease of E after each gradient step. Finally, unlike in [20], our filters do not change at each gradient descent iteration, which means that we are still trying to emulate the minimum of the energy functional, but with a smaller number of steps.

In order to simplify the notation, we now assume to have just one training example and we define the variable $\lambda = (\lambda_1, \dots, \lambda_K) \in \mathbb{R}^K$. The bilevel problem that needs to be solved is then:

$$\begin{cases} \underset{\theta, \lambda > 0}{\operatorname{argmin}} \mathcal{L}(\theta, \lambda) \equiv \frac{1}{2} \|u^*(\theta, \lambda) - g\|_2^2 \\ \text{s. to } \; u^*(\theta, \lambda) = u^{(K)} = u^{(K-1)} - \lambda_K \nabla_u E(u^{(K-1)}, \theta) \\ \text{with } u^{(k)} = u^{(k-1)} - \lambda_k \nabla_u E(u^{(k-1)}, \theta) \text{ and } u^{(0)} = f. \end{cases}$$

Algorithm 2 describes the modified training scheme.

This time the loss gradient can be computed with the backpropagation procedure described in [19]. First, we can compute the gradient and the Hessian of the energy functional

$$\nabla_u E(u, \theta) = u - f + \sum_{i=1}^{N_f} \alpha_i A_i^T \phi'(A_i u),$$

$$\nabla_{uu}^2 E(u, \theta) = I + \sum_{i=1}^{N_f} \alpha_i A_i^T D_i A_i,$$

where D_i is the $n \times n$ diagonal matrix with $\phi''(A_i u)$ on the main diagonal.

We can also explicitly compute the partial derivatives of the gradient w.r.t the parameters θ (but not λ, since the dependence is not explicit):

$$\frac{\partial \nabla_u E(u, \theta)}{\partial \alpha_i} = A_i^T \phi'(A_i u) \in \mathbb{R}^n,$$

$$\frac{\partial \nabla_u E(u, \theta)}{\partial \beta_{ij}} = \alpha_i \left(B_j^T \phi'(A_i u) + A_i^T D_j B_j u \right) \in \mathbb{R}^n.$$

At each step of the backpropagation we will need to apply the

chain rule since each iterate depends on the parameters, hence we need the partial derivatives of the generic $u^{(k)}$ w.r.t. the steplength λ_j and the energy parameter θ_j:

$$\frac{\partial u^{(k+1)}}{\partial \lambda_j} = \frac{\partial}{\partial \lambda_j}\left(u^{(k)} - \lambda_k \nabla_u E(u^{(k)}, \theta)\right)$$
$$= \left[\frac{\partial u^{(k)}}{\partial \lambda_j}\right]^T \left(I_n - \lambda_k \nabla^2_{uu} E(u^{(k)}, \theta)\right) - \delta_{kj} \nabla_u E(u^{(k)}, \theta),$$
$$\frac{\partial u^{(k+1)}}{\partial \theta_j} = \frac{\partial}{\partial \theta_j}\left(u^{(k)} - \lambda_k \nabla_u E(u^{(k)}, \theta)\right)$$
$$= \left[\frac{\partial u^{(k)}}{\partial \theta_j}\right]^T \left(I_n - \lambda_k \nabla^2_{uu} E(u^{(k)}, \theta)\right) - \lambda_k \frac{\partial \nabla_u E(u^{(k)}, \theta)}{\partial \theta_j},$$

where δ_{jk} denotes the usual Kronecker delta.

We are now ready to describe a scheme to compute the loss gradient. Starting from the derivative w.r.t. the parameter θ_j we have:

$$\frac{\partial \mathcal{L}(\theta, \lambda)}{\partial \theta_j} = \left[\frac{\partial u^*}{\partial \theta_j}\right]^T (u^* - g)$$
$$= \left[\frac{\partial u^{(K)}}{\partial \theta_j}\right]^T \left(I_n - \lambda_K \nabla^2_{uu} E(u^{(K)}, \theta)\right)^T (u^* - g)$$
$$- \lambda_K \left[\frac{\partial \nabla_u E(u^{(K)}, \theta)}{\partial \theta_j}\right]^T (u^* - g);$$

by defining the following vectors

$$z^{(K+1)} = u^* - g,$$
$$w^{(K+1)} = \mathbf{0},$$
$$\vdots$$
$$z^{(k)} = \left(I_n - \lambda_k \nabla^2_{uu} E(u^{(k)}, \theta)\right)^T z^{(k+1)},$$
$$w^{(k)} = -\lambda_k \left[\frac{\partial \nabla_u E(u^{(k)}, \theta)}{\partial \theta_j}\right]^T z^{(k+1)} + w^{(k+1)},$$

we can then arrive at the result of the chain rule as

$$\frac{\partial \mathcal{L}(\theta)}{\partial \theta_j} = \left[\frac{\partial u^{(K)}}{\partial \theta_j}\right]^T z^{(K)} + w^{(K)}$$
$$\vdots$$
$$= \left[\frac{\partial u^{(1)}}{\partial \theta_j}\right]^T z^{(1)} + w^{(1)}$$
$$= w^{(0)}.$$

One can repeat the same procedure to compute the gradient w.r.t. λ_j, with the only exception of defining the vectors

$$v^{(K+1)} = \mathbf{0},$$
$$v^{(k)} = -\delta_{kj} \nabla_u E(u^{(k)}, \theta)^T z^{(k)} + v^{(k+1)},$$

instead of the $\{w^{(j)}\}_{j=0,\dots K}$. By also noting that for every $j = 1, \dots, K$ all but one of the $v^{(k)}$ are null, one can conclude that

$$\frac{\partial \mathcal{L}(\theta, \lambda)}{\partial \lambda_j} = -\nabla_u E(u^{(j)}, \theta)^T z^{(j+1)}.$$

Algorithm 2:

Data: (f, g), maxIter, K, $\theta^{(0)}$, $\lambda^{(0)}$
for $i = 0, \dots,$ maxIter **do**
  for $k = 1, \dots, K$ **do**
    Compute $u^{(k)} = u^{(k-1)} - \lambda_k^{(i)} \nabla_u E(u^{(k-1)}, \theta^{(i)})$.
  end
  Compute $\nabla_\theta \mathcal{L}(\theta^{(i)}, \lambda^{(i)})$ and $\nabla_\lambda \mathcal{L}(\theta^{(i)}, \lambda^{(i)})$.
  Update the parameters to $\theta^{(i+1)}$ and $\lambda^{(i+1)}$ by means of $\nabla_\theta \mathcal{L}(\theta^{(i)}, \lambda^{(i)})$ and $\nabla_\lambda \mathcal{L}(\theta^{(i)}, \lambda^{(i)})$.
end

IV. Numerical Experiments

For comparison purposes, we utilized the BSDS300 dataset [17] as did the authors of [6], with a training set of 200 images patches of size 64×64 and a test set of 68 elements of dimension 481×321. To generate the noisy counterpart, every image was corrupted with additive white Gaussian noise with $\sigma = 25$.

Regarding the sparsity inducing function and the filter basis, we used, respectively, $\phi(t) = \log(1 + t^2)$ and the DCT basis deprived of the first constant filter. These choices followed the ones made by the authors of [6] in their work, which in turn were motivated with statistical reasoning. In particular, the filter basis forces the $\{a_i\}_{i=1,\dots,N_f}$ to be zero-mean.

To minimize the loss function, we employed various stochastic optimizers (with and without variance reduction) with different fixed learning rates to find the best parameters for the model, both in the case where the lower level problem is still a minimization problem and in the case of the unrolling. We also tested, for reference, a couple of deterministic schemes. When the lower level problem was fully minimized, we computed the loss minimum with both the L-BFGS algorithm [3][1] and the Scaled Gradient Projection method [2] with a BFGS scaling matrix. On the other hand, when we unrolled the gradient descent scheme, we only used the SGP method. The number of outer iterations was always set to a maximum of 500. As stopping criterion we ceased the training when either the relative change in the loss function value between two subsequent iterations was less than 10^{-5}, or when the method exceeded the maximum number of backtracking reductions in the linesearch step.

Full minimization case

Each lower level approximate solution was computed with a L-BFGS algorithm. The algorithm was set to run for a maximum number of 500 iterations (rarely reached) or until $\|\nabla_u E(u)\|_2 \leq 10^{-3}$.

Regarding the stochastic approaches, we used the mini-batch versions of stochastic gradient descent, vanilla [21] and methods with variance reduction: momentum and AdaM [15], processing 20 examples at a time and all with fixed learning rate, for a total of 100 epochs. Where by epoch we mean an entire view of the dataset in groups of 20 elements.

In figure 1 we have plotted the final results of the training. The choice of the learning rate required a bit of tuning: for too high values, the loss would have an oscillating behaviour, while for too low ones, the convergence would be slow.

[1] Code downloaded from https://github.com/stephenbeckr/L-BFGS-B-C

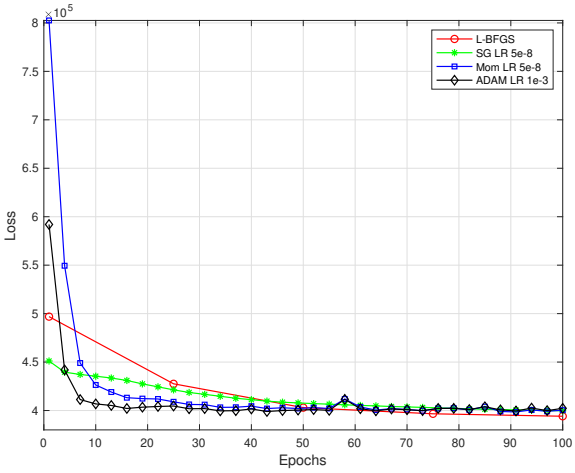

Fig. 1. Plot of the loss decrease by training with the stochastic schemes (only the best learning rates for each scheme is plotted) as well as with the L-BFGS algorithm.

TABLE I
Summary of the training results. The final loss value and the relative average PSNR are computed after 100 epochs for all four schemes. For the stochastic algorithms, only the best learning rates are reported. [6]

Optimizer	Final Loss Value	Average PSNR
L-BFGS	394090	28.61
SG $5e{-}8$	399749	28.48
MOM $5e{-}8$	400388	28.51
AdaM $1e{-}3$	402266	28.50

Moreover, with no surprise, the inertial schemes perform better in the very first epochs, but tend to stall rather quickly. Finally, we also found that all three stochastic schemes performed better than the deterministic approaches in the beginning, which were able to find lower loss values only after more than 50 epochs, and even then, after 100 epochs the gap was not too wide, as the reader can see in table I.

For a matter of completeness, the L-BFGS scheme, after 500 iterations, yields a loss function value of 387930 and a an average PSNR of 28.61 on the test set, a very similar result to what the authors of [6] obtained (i.e., 388053 and 28.66 respectively), as well as all the other algorithms they compared themselves with: BM3D [7], LSSC [16], GMM-EPLL [23], FoE [12], KSVD [8] and GOAL [13].

Unrolling case

We performed the same experiments after replacing the lower level problem with the unrolling of the gradient descent algorithm for $K = 5, 10, 15, 20$ iterates. This time the best performing stochastic scheme was the minibatch SG with fixed learning rate $\lambda = 1e{-}7$, from start to finish. This is due to the fact that the we had to reduce the steplength for the inertial methods to avoid either divergence or early stalling.

As for the number of the inner iterates tested, as one would suppose, the more the better. There is a significant increase in the performance when using 5 inner iterates or 10, as well as going from 10 to 15 (see figure 3). However, the same does not apply with more than 15 iterates, i.e., the increase in the quality of the reconstruction probably does not justify the extra computational effort.

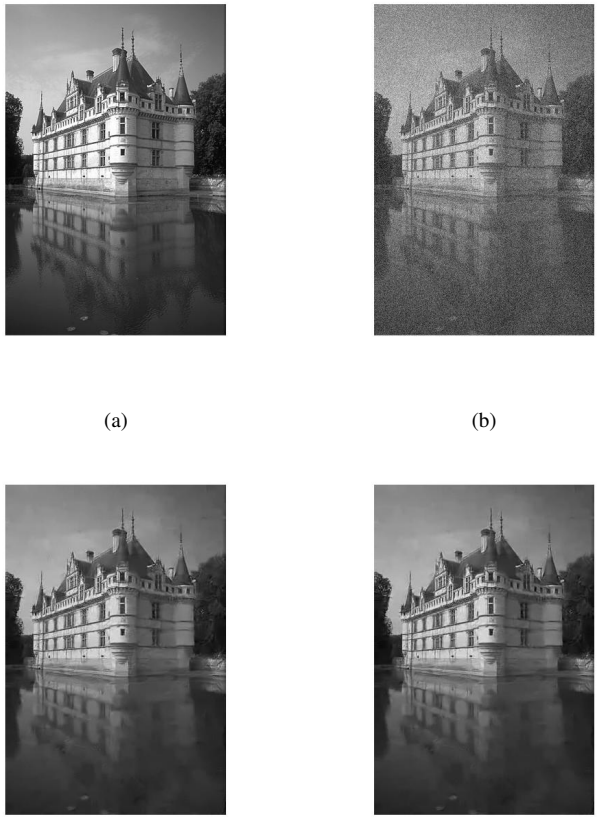

Fig. 2. Example of a reconstruction: (a) ground truth; (b) noisy image; (c) and (d) reconstructions obtained with the unrolling of respectively 10 and 15 steps of gradient descent, where the parameters of the model were learned with SG.

In table II we have reported the final numbers for the unrolling experiments. Besides the stochastic schemes, we also tested the quasi-Newton version of SGP described previously. As in the full minimization case, the final performance for the deterministic scheme is slightly better in the end, but it only comes after a significant number of full iterations.

V. Conclusion

In this work we compared some variants of the classical bilevel approach for learning the parameters of an optimal regularization functional for image restoration. In particular, we replaced the lower level problem with a fixed number of steps of an optimization method, whose parameters are then learned together with the regularization ones. Moreover, we also compared the performances of different stochastic optimization methods for the minimization of the upper level problem.

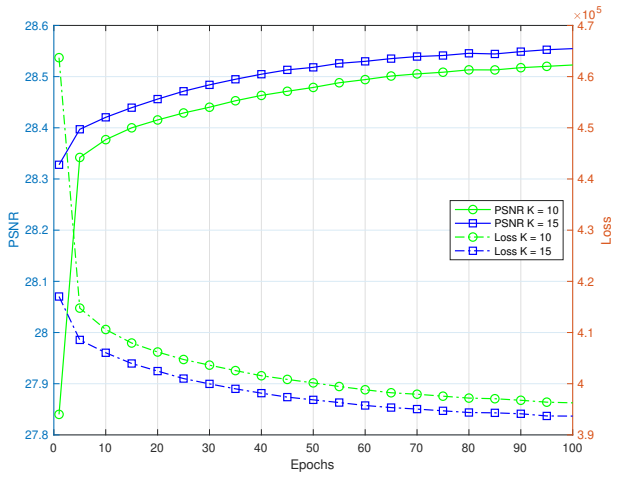

Fig. 3. Loss and average PSNR behaviour for the cases $K = 10$ and $K = 15$. Loss optimizer: SG with $\lambda = 1\text{e}{-7}$

TABLE II
Summary of the training results when the lower level problem is replaced by the unrolling of a gradient descent scheme for $K = 10, 15$ iterations. The loss was optimized with the stochastic schemes (only the best learning rates are reported) and a quasi-Newton deterministic algorithm.

Optimizer	Final Loss Value	Average PSNR
$K = 10$		
Quasi-Newton SGP	387441	28.57
SG 1e−7	396262	28.52
MOM 1e−8	400209	28.46
AdaM 1e−4	401220	28.49
$K = 15$		
Quasi-Newton SGP	386698	28.6
SG 1e−7	393670	28.55
MOM 1e−8	395823	28.52
AdaM 1e−4	398806	28.51

The numerical experiments show that the results of a combination of the unrolling approach with the stochastic optimization for the loss minimization are comparable with those of the original bilevel approach in terms of accuracy. On the other side, the unrolling approach leads to a computable, explicit form of the gradient of the loss function and also to a reduction of the complexity of the overall learning algorithm. This is only the step one, as this formulation allows, in theory, to add many other constraints on the parameters and the reconstructed images, e.g. by applying a differentiable projection at each gradient step to force the images to remain in a certain range. In the future we would also like to investigate the behaviour of this model in other contexts: on one hand, if the lower level problem is solved as a full minimization problem, then technically we are learning the image prior and thus we should be able to use the regularizer, for instance, in inpainting, deblurring and so on; on the other hand, if we unroll an optimization algorithm, it is not as straightforward and the regularizer may be problem-dependent.

References

[1] M. Bertero, P. Boccacci, and V. Ruggiero, *Inverse Imaging with Poisson Data*. IOP Publishing, 2018.

[2] S. Bonettini, R. Zanella, and L. Zanni, "A scaled gradient projection method for constrained image deblurring," *Inverse Problems*, vol. 25, no. 1, p. 015002, 2009.

[3] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, "A limited memory algorithm for bound constrained optimization," *SIAM Journal on Scientific Computing*, vol. 16, no. 5, pp. 1190–1208, 1995.

[4] L. Calatroni, C. Cao, J. C. D. los Reyes, C.-B. Schönlieb, and T. Valkonen, *Bilevel approaches for learning of variational imaging models*. De Gruyter, 2017, pp. 252–290.

[5] P. Cascarano, E. L. Piccolomini, E. Morotti, and A. Sebastiani, "Plug-and-play gradient-based denoisers applied to ct image enhancement," *Applied Mathematics and Computation*, vol. 422, p. 126967, 2022.

[6] Y. Chen, R. Ranftl, and T. Pock, "Insights into analysis operator learning: From patch-based sparse models to higher order mrfs," *IEEE Transactions on Image Processing*, vol. 23, no. 3, pp. 1060–1072, 2014.

[7] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-d transform-domain collaborative filtering," *IEEE Transactions on Image Processing*, vol. 16, no. 8, pp. 2080–2095, 2007.

[8] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Transactions on Image Processing*, vol. 15, no. 12, pp. 3736–3745, 2006.

[9] G. Franchini, V. Ruggiero, and L. Zanni, "On the steplength selection in stochastic gradient methods," in *Numerical Computations: Theory and Algorithms*, Y. D. Sergeyev and D. E. Kvasov, Eds. Springer International Publishing, 2020, pp. 186–197.

[10] G. Franchini, V. Ruggiero, and L. Zanni, "Ritz-like values in steplength selections for stochastic gradient methods," *Soft Computing*, vol. 24, no. 23, pp. 17 573–17 588, 2020.

[11] G. Franchini, V. Ruggiero, and L. Zanni, "Steplength and mini-batch size selection in stochastic gradient methods," in *Machine Learning, Optimization, and Data Science*, G. Nicosia, V. Ojha, E. La Malfa, G. Jansen, V. Sciacca, P. Pardalos, G. Giuffrida, and R. Umeton, Eds. Springer International Publishing, 2020, pp. 259–263.

[12] Q. Gao and S. Roth, "How well do filter-based mrfs model natural images?" in *Pattern Recognition*, A. Pinz, T. Pock, H. Bischof, and F. Leberl, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 62–72.

[13] S. Hawe, M. Kleinsteuber, and K. Diepold, "Analysis operator learning and its application to image reconstruction," *IEEE Transactions on Image Processing*, vol. 22, no. 6, pp. 2138–2150, 2013.

[14] M. Hintermüller and T. Wu, "Bilevel optimization for calibrating point spread functions in blind deconvolution," *Inverse Problems and Imaging*, pp. 1139–1169, 2015.

[15] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[16] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Non-local sparse models for image restoration," in *2009 IEEE 12th International Conference on Computer Vision*, 2009, pp. 2272–2279.

[17] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, vol. 2, 2001, pp. 416–423 vol.2.

[18] M. Moeller and D. Cremers, *Image Denoising—Old and New*. Springer International Publishing, 2018, pp. 63–91.

[19] P. Ochs, R. Ranftl, T. Brox, and T. Pock, "Techniques for gradient-based bilevel optimization with non-smooth lower level problems," *Journal of Mathematical Imaging and Vision*, vol. 56, no. 2, pp. 175–194, 2016.

[20] D. Ren, W. Zuo, D. Zhang, L. Zhang, and M.-H. Yang, "Simultaneous fidelity and regularization learning for image restoration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 1, pp. 284–299, 2021.

[21] H. Robbins and S. Monro, "A stochastic approximation method," *The Annals of Mathematical Statistics*, vol. 22, no. 3, pp. 400–407, 1951.

[22] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: Nonlinear Phenomena*, vol. 60, no. 1, pp. 259–268, 1992.

[23] D. Zoran and Y. Weiss, "From learning models of natural image patches to whole image restoration," in *2011 International Conference on Computer Vision*, 2011, pp. 479–486.