International Conference on Adaptive Modeling and Simulation
ADMOS 2023
F.Larsson and P.Díez

A PIGGYBACK-STYLE ALGORITHM FOR LEARNING IMPROVED SHEARLETS AND TGV DISCRETIZATIONS

Lea Bogensperger*, Antonin Chambolle† and Thomas Pock*

* Institute of Computer Graphics and Vision, Graz University of Technology, 8010 Graz, Austria
lea.bogensperger@icg.tugraz.at, pock@icg.tugraz.at

† CEREMADE, CNRS & Universit Paris-Dauphine PSL, 75016 Paris, France
chambolle@ceremade.dauphine.fr

Key words: Learning, Piggyback Algorithm, Primal-Dual Algorithm, Shearlets, Total Generalized Variation, Bilevel Optimization

Summary. This work demonstrates how to use a piggyback-style algorithm to compute derivatives of loss functions that depend on solutions of convex-concave saddle-point problems. Two application scenarios are presented, where the piggyback primal-dual algorithm is used to learn an enhanced shearlet transform and an improved discretization of the second-order total generalized variation.

1 INTRODUCTION

For inverse problems in imaging, there is a wide range of regularizers that can be used, classic choices include the total variation (TV), total generalized variation (TGV), wavelets and shearlets, to name but a few. The typical variational structure is governed by minimizing an energy $h_\theta(x)$ given by

$$\min_{x\in\mathcal{X}} \; h_\theta(x) := g(x,z) + f(K_\theta x), \tag{1}$$

for input data z, where K_θ is a linear operator related to the chosen regularizer. This offers a plug-and-play framework for the choice of the regularizer $f(K_\theta x)$ that is used, whereas the data term $g(x,z)$ is usually determined by the underlying task. As f is often a non-smooth function, problem (1) can also be cast as a saddle-point problem reading as

$$\min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} \; \langle K_\theta x, y\rangle + g(x,z) - f^*(y), \tag{2}$$

where f^* denotes the convex conjugate of f. Depending on the convex imaging problem and the domain of the underlying data, it can be beneficial to learn θ such that we have an optimal K_θ to improve some of the common choices within a convex imaging application. This can be tackled with supervised learning, which allows to learn K_θ in a bilevel setting given a training data set. For such a training data set with N pairs $\{(z_n, t_n)\}_{n=1}^N$ denoting

the corrupted observations and corresponding ground truth samples, respectively, the bilevel optimization problem has the following structure:

$$\begin{aligned} \min_{\theta} \mathcal{L}(K_\theta) + \mathcal{R}(K_\theta) &:= \tfrac{1}{N} \sum_{n=1}^{N} \ell(\hat{x}_n(K_\theta), t_n) + \mathcal{R}(K_\theta), \\ \text{s.t. } \hat{x}_n &\in \arg\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} \ \langle K_\theta x, y \rangle + g(x, z) - f^*(y). \end{aligned} \tag{3}$$

Note that $\ell(\hat{x}_n(K_\theta), t_n)$ is a convex and differentiable loss function, such as the squared ℓ_2 norm, i.e. $\ell(\hat{x}_n(K_\theta), t_n) = \frac{1}{2}\|\hat{x}_n(K_\theta) - t_n\|_2^2$, whereas $\mathcal{R}(K_\theta)$ can additionally impose constraints on the learned θ depending on the chosen regularizer.

The key question is now how derivatives of the loss function with respect to parameters θ (or the linear operator K_θ) can be computed efficiently. After briefly recalling two classical and well-known methods and discussing their advantages but also potential disadvantages, an alternative how to compute the gradients based on a standard sensitivity analysis is reviewed, namely a piggyback-style algorithm [6], which is suitable for convex-concave saddle-point problems [1].

2 METHODS

2.1 Bilevel Optimization

Bilevel optimization using implicit differentiation [12] attempts to solve nested optimization problems of the general form

$$\min_{\theta} \tfrac{1}{N} \sum_{n=1}^{N} \ell(\hat{x}_n(K_\theta), t_n), \ \text{s.t. } \hat{x}_n \in \arg\min_{x \in \mathcal{X}} h_\theta(x).$$

Thus, for a data sample n, the sought gradient of the higher level problem with respect to the parameters θ, i.e. $\nabla_\theta \ell(\hat{x}_n(K_\theta), t_n)$, is computed as

$$\nabla_\theta \ell(\hat{x}_n(K_\theta), t_n) = -\big(\nabla_x \ell(\hat{x}_n(K_\theta), t_n)\big)^T [\nabla_x^2 h_\theta(\hat{x}_n(K_\theta))]^{-1} \nabla_\theta \nabla_x h_\theta(\hat{x}_n(K_\theta)), \tag{4}$$

with $\hat{x}_n$ such that $\nabla_x h_\theta(x_n(K_\theta))|_{x=\hat{x}_n} = 0$. An advantage of this approach is that it does not actually require the saddle-point structure as in (3), since the lower level can represent any energy that minimizes x depending on parameters θ. However, due to the involved Hessian, it can be computationally very expensive. Moreover, it requires a certain regularity of the lower level problem, i.e. the energy function that is optimized for has to be twice continuously differentiable in x.

A second popular strategy to deal with bilevel optimization problems are unrolling techniques [9]. Hereby, the bilevel learning is approximated by unraveling a certain number of iterations $\bar{k}$ of the lower level problem by some gradient-based scheme. Then, $\hat{x}_n$ is replaced by $x_n^{\bar{k}}$ which is the $\bar{k}$-th iterate of the gradient-based scheme, i.e.

$$\min_{\theta} \tfrac{1}{N} \sum_{n=1}^{N} \ell(x_n^{\bar{k}}(K_\theta), t_n), \ \text{s.t. } x_n^{k+1} = x_n^k - \tau^k \nabla_x h_\theta(x_n^k) \quad \text{for } k = 0, 1, \ldots, \bar{k} - 1.$$

Further, the chain rule (or automatic differentiation) is used to obtain the final derivatives. This is a very straightforward approach, however, it only approximates the bilevel optimization problem if $\bar{k} \to \infty$. While there are approaches to interprete this as early-stopping in a discretized continous-time gradient flow setting on the energy model [10], convergence in theory is not guaranteed. Moreover, the number of iterations that can be unraveled is limited by memory constraints.

Thus, unrolling is a very convenient approach but it comes without theoretical convergence; while implicit differentiation on the other hand – which is computing the exact derivative – is computationally heavy and requires more regularity. The questions arises whether something else could be considered for less smooth energies, that offers convergence guarantees while being computationally feasible. Hereby, a piggyback algorithm is reviewed that fulfills these requirements in case of underlying convex-concave saddle-point problems in the form of (2) for the lower level problem.

2.2 Derivation of the Piggyback Algorithm

This section motivates the use of the piggyback algorithm for the derivatives of saddle-point problems as given in (2) and is taken from [4]. For ease of notation we assume that the entire linear operator K is learned and we drop the notation of the data sample n such that $\mathcal{L}(K) = \ell(\hat{x}(K), t)$. The result can subsequently be adapted to only learn a subset K_θ that composes the linear operator given a specific sample n. The saddle-point problem in (2) has the primal problem shown in (1) and the corresponding dual problem

$$\max_{y \in \mathcal{Y}} -f^*(y) - g^*(-K^*y). \tag{5}$$

The analysis is further based on the assumption of a unique saddle-point $(\hat{x}, \hat{y})$ with optimality conditions

$$\begin{cases} K\hat{x} - \partial f^*(\hat{y}) = 0, \\ K^*\hat{y} + \partial g(\hat{x}) = 0. \end{cases}$$

The starting point is to perturb the saddle-point problem by a small variation on the linear operator $K + sL$ with $|s| \ll 1$, which results in a perturbed solution of the saddle-point problem in the directions ξ_s, η_s, i.e. we obtain $\hat{x}_s = \hat{x} + s\xi_s$ and $\hat{y}_s = \hat{y} + s\eta_s$. Substituting the perturbed solution $(\hat{x}_s, \hat{y}_s)$ into the optimality conditions and using the fundamental theorem of calculus, one obtains

$$\begin{cases} K\hat{x} + s(K\xi_s + L\hat{x}_s) - [\partial f^*(\hat{y}) + (\int_0^s D^2 f^*(\hat{y} + t\eta_s)\,\mathrm{d}t)\eta_s] = 0, \\ K^*\hat{y} + s(K^*\eta_s + L^*\hat{y}_s) + [\partial g(\hat{x}) + (\int_0^s D^2 g(\hat{x} + t\xi_s)\,\mathrm{d}t)\xi_s] = 0. \end{cases}$$

Using the optimality conditions and dividing by s yields

$$\begin{cases} K\xi_s + L\hat{x}_s - (\frac{1}{s}\int_0^s D^2 f^*(\hat{y} + t\eta_s)\,\mathrm{d}t)\eta_s = 0, \\ K^*\eta_s + L^*\hat{y}_s + (\frac{1}{s}\int_0^s D^2 g(\hat{x} + t\xi_s)\,\mathrm{d}t)\xi_s = 0, \end{cases}$$

and as $s \to 0$ one can see that ξ_s, η_s go to ξ, η satisfying

$$\begin{cases} K\xi + L\hat{x} - D^2 f^*(\hat{y})\eta = 0, \\ K^*\eta + L^*\hat{y} + D^2 g(\hat{x})\xi = 0. \end{cases}$$

If the order of the equations are changed and the second line is multiplied by -1, one can see that the unique solution is given by

$$\begin{pmatrix} \xi \\ \eta \end{pmatrix} = \begin{pmatrix} D^2 g(\hat{x}) & K^* \\ -K & D^2 f^*(\hat{y}) \end{pmatrix}^{-1} \begin{pmatrix} -L^*\hat{y} \\ L\hat{x} \end{pmatrix}.$$

We can now compute the directional derivative $\mathcal{L}'(K; L) = \langle \nabla \mathcal{L}(K), L \rangle$ as follows:

$$\mathcal{L}'(K; L) = \nabla \ell(\hat{x}, \hat{y})^T \begin{pmatrix} \xi \\ \eta \end{pmatrix} = \nabla \ell(\hat{x}, \hat{y})^T \begin{pmatrix} D^2 g(\hat{x}) & K^* \\ -K & D^2 f^*(\hat{y}) \end{pmatrix}^{-1} \begin{pmatrix} -L^*\hat{y} \\ L\hat{x} \end{pmatrix}.$$

Next, we introduce adjoint variables X, Y which do not require knowledge of the perturbation direction L but only of the derivative of the loss function, such that

$$\begin{aligned} (-X^T, Y^T) = \nabla \ell(\hat{x}, \hat{y})^T \begin{pmatrix} D^2 g(\hat{x}) & K^* \\ -K & D^2 f^*(\hat{y}) \end{pmatrix}^{-1} \\ \Leftrightarrow \quad \nabla \ell(\hat{x}, \hat{y}) = \begin{pmatrix} D^2 g(\hat{x}) & -K^* \\ K & D^2 f^*(\hat{y}) \end{pmatrix} \begin{pmatrix} -X \\ Y \end{pmatrix}, \end{aligned}$$

which can also be written as

$$\begin{cases} D^2 g(\hat{x})X + K^* Y + \nabla_x \ell(\hat{x}, \hat{y}) = 0, \\ -KX + D^2 f^*(\hat{y})Y - \nabla_y \ell(\hat{x}, \hat{y}) = 0. \end{cases} \tag{6}$$

Equation (6) contains the optimality conditions of the quadratic adjoint saddle-point problem:

$$\min_{X \in \mathcal{X}} \max_{Y \in \mathcal{Y}} \ \langle KX, Y \rangle + \frac{1}{2}\langle D^2 g(\hat{x})X, X \rangle - \frac{1}{2}\langle D^2 f^*(\hat{y})Y, Y \rangle + \Big\langle \nabla \ell(\hat{x}, \hat{y}), \begin{pmatrix} X \\ Y \end{pmatrix} \Big\rangle.$$

Denoting by $(\hat{X}, \hat{Y})$ the unique solution of this adjoint saddle-point problem, the directional derivative (6) is given by

$$\mathcal{L}'(K; L) = \langle \hat{X}, L^*\hat{y} \rangle + \langle \hat{Y}, L\hat{x} \rangle, \tag{7}$$

for any L. Using $\mathcal{L}'(K; L) = \langle \nabla \mathcal{L}(K), L \rangle$, the gradient is then given by (which relates to the form in (4))

$$\nabla \mathcal{L}(K) = \hat{Y} \otimes \hat{x} + \hat{y} \otimes \hat{X}.$$

2.3 Piggyback Algorithm

The piggyback algorithm involves both the computation of solutions of saddle-points of the primary (lower level) problem and the adjoint (secondary) problem, which in turn depends on the primary solution. Instead of doing this sequentially, however, the idea is to run both primal-dual algorithms in parallel, see Algorithm 1 for the general iterations (note that $\widetilde{L}$ is the Lipschitz constant of the linear operator K_θ). Naturally, as the secondary primal-dual algorithm depends on the solution of the primary primal-dual algorithm, this is also reflected in its convergence rate (see Section 2.5).

Algorithm 1: Piggyback primal-dual algorithm for solving (2) and its adjoint (6).

- Initialization: $x^0, X^0 \in \mathcal{X}$, $y^0, Y^0 \in \mathcal{Y}$.
- Step sizes: Choose the step sizes τ, σ such that $\sigma\tau\widetilde{L}^2 \leq 1$.
- Iterations: For each $k = 0, \dots, \bar{k} - 1$ let

$$\begin{cases} \widetilde{x}^{k+1} &= x^k - \tau K_\theta^* y^k, \quad \widetilde{X}^{k+1} = X^k - \tau(K_\theta^* y^k + \nabla_x \ell(x^k, t)) \\ x^{k+1} &= \operatorname{prox}_{\tau g}(\widetilde{x}^{k+1}), \quad X^{k+1} = \nabla \operatorname{prox}_{\tau g}(\widetilde{x}^{k+1}) \cdot \widetilde{X}^{k+1} \\ \bar{x}^{k+1} &= 2x^{k+1} - x^k, \quad \bar{X}^{k+1} = 2X^{k+1} - X^k \\ \widetilde{y}^{k+1} &= y^k + \sigma K_\theta \bar{x}^{k+1}, \; \widetilde{Y}^{k+1} = Y^k + \sigma K_\theta \bar{X}^{k+1} \\ y^{k+1} &= \operatorname{prox}_{\sigma f^*}(\widetilde{y}^{k+1}), \quad y^{k+1} = \nabla \operatorname{prox}_{\sigma f^*}(\widetilde{y}^{k+1}) \cdot \widetilde{Y}^{k+1} \end{cases}$$

- Output: saddle-point $(x^{\bar{k}}, y^{\bar{k}}) \approx (\hat{x}, \hat{y})$ and adjoint state $(X^{\bar{k}}, Y^{\bar{k}}) \approx (\hat{X}, \hat{Y})$.

2.4 Computing Derivatives

In practice, the learnable parameters θ are often only a subset of the operator K_θ – although one can of course also learn the entire operator K, e.g. by learning a set of zero-mean filters for denoising. Thus, to compute the derivatives it is convenient to evaluate (7) and use automatic differentiation to extract the gradient with respect to the relevant parameters. Once the gradients are computed, any gradient-based algorithm e.g. with acceleration terms or moving gradient averages is suitable for learning.

Constraints on the learnable parameters as indicated by $\mathcal{R}(K_\theta)$ can be incorporated by means of projections after an update step on the learnable parameters θ was performed.

2.5 Convergence of the Piggyback Algorithm

The piggyback primal-dual algorithm is shown to converge linearly for f^*, g strongly convex and f, g^* locally $\mathcal{C}^{2,\alpha}$ for some $\alpha \in (0, 1]$ [1]. This linear convergence of the primary primal-dual algorithm is consequently reflected in a slower linear convergence of the adjoint problem. This is due to the nature of the piggyback algorithm where solving the adjoint problem follows the primary problem.

In practice, is was shown that the algorithm can be used on less smooth energies as well while remaining remarkably robust with the computed gradients [4, 1].

3 NUMERICAL RESULTS

To demonstrate the applicability of the piggyback primal-dual approach, it is applied to two numerical examples. In both cases, a classical regularizer is enhanced by optimizing it towards a specific inverse imaging problem learned on a chosen training set.

3.1 Shearlet Transform

First, an optimal shearlet transform is learned in an image denoising setting [1] on natural images and cartoon-like data. Shearlets are based on an extension of wavelets with the benefit of offering more isotropy whilst being faithfully discretizable [11]. The digital shearlet transform applied to an image $x \in \mathcal{X}$ at scale $j > 0$ and shear level $|k| \leq \lceil 2^{j/2} \rceil$ reads as

$$DST_{j,k}(x) = \lambda_{j,k} \overline{\psi^d_{j,k}} * x. \tag{8}$$

More specifically $\overline{\psi^d_{j,k}}$ is composed of a few basic building blocks that originate from a low-pass filter h_1 and a 2D directional filter P. A multiresolution analysis allows to derive all subsequent required filters from these two components, see [1] for more details. Moreover, the regularization parameters $\lambda_{j,k} > 0$ can weight the invidual contribution of each shearlet. Thus the learnable parameters are comprised of $\theta = \{\lambda_{j,k}, h_1, P\}$.

This is incorporated into a denoising problem regularized by the shearlet transform in the form of (1) and reads as

$$\min_x g(x, z) + f(K_\theta x),$$

where K_θ is a linear operator that computes the shearlet transform as given in (8). Note that to better fit the framework where the convergence of the piggyback was shown, a smooth approximation for $f_\varepsilon(v) = \sum_i \phi_\varepsilon(v_i)$ is used (i denotes the pixel index of a shearlet coefficient). More specifically, this is a $\mathcal{C}^{2,\alpha}$ approximation ϕ_ε with $\alpha = 1$ of the $|\cdot|$ function and it reads as

$$\phi_\varepsilon(t) = \begin{cases} -\frac{|t|^3}{3\varepsilon^2} + \frac{t^2}{\varepsilon} + \frac{\varepsilon}{3}, & \text{if } |t| \leq \varepsilon, \\ |t| & \text{else.} \end{cases}$$

This function also allows to obtain a closed form solution for the conjugate function ϕ^*_ε as well as for its proximal map. Casting it as a saddle-point problem allows to use a piggyback algorithm to learn the parameters $\theta = \{\lambda_{j,k}, h_1, P\}$.

On two data sets of natural images and cartoon-like images (for which the shearlet transform is optimized), we were able to show the qualitative and quantitative improvements obtained in denoising the corresponding test data set after learning θ. Experimentally we show that with decreasing smoothness parameter ε of the regularizing function f (which then approximates the $|\cdot|$ function) the piggyback algorithm remains robust and in practice works well even on less regular functions. Finally, we examine the primal-dual

gaps of the primary and the adjoint problem for different ε showing the slower linear convergence of the adjoint saddle-point problem, see [1].

3.2 TGV Discretization

The second application scenario is learning an improved discretization of the second-order TGV regularizer [2]. As it builds upon TV, it can also suffer from discretization artefacts related to isotropy and rotational invariance [5, 4], which is due to staggered pixel grids arising from discretized finite differences. The authors in [7] propose an improved handcrafted TGV discretization, where the dual variables are interpolated to denser grids following the style of [5]. This is extended to a more general (consistent) setting, where an improved discretization can be learned using a piggyback algorithm.

The TGV regularizer involves the first-order and second-order finite differences operators D and E (see [2] for definitions) and the resulting symmetrized second-order finite differences operator $D^2 = ED$. Using interpolation operators K and L that map the dual variables p and div p to n_K and n_L times denser grids, respectively, the more general setting of the newly discretized TGV regularizer for an image $x \in \mathcal{X}$ reads as

$$\min_{vK,vL} \alpha_1 \|v_L\|_Z + \alpha_0 \|v_K\|_Z, \text{ s.t. } D^2 x = EL^* v_L + K^* v_K.$$

Here, the norm $\|v\|_Z = \|v\|_{1,1,2}$ for some $v \in \mathbb{R}^{(M\times N)\times J\times I}$ corresponds to the absolute sum of I components of the 2-norm of its J components. Coupling this regularizer with a data term $g(x, z)$ and incorporating the constraint, the saddle-point problem is obtained:

$$\min_{x,v_K,v_L} \max_{y} \; g(x, z) + \alpha_0 \|v_K\|_Z + \alpha_1 \|v_L\|_Z + \langle D^2 x - EL^* v_L - K^* v_K, y\rangle.$$

Using an ℓ_2 norm for the outer bilevel loss function and constraints on the interpolation filter coefficients (to ensure their boundedness), the piggyback algorithm can readily be applied to obtain saddle-points for the primary and the adjoint problems. Once the gradients of the loss function with respect to the filter coefficients are computed, they are updated using a (block-)Adam optimizer [8].

Interestingly, the learned interpolation filters tend to capture different orientations for the operator L while exhibiting more of a smoothing effect for the filter K that operates on the second-order finite differences. Numerical results for denoising a synthetic piecewise affine data set and a natural image data set demonstrate the effectiveness of the learned interpolation filters in terms of visual quality but also quantitative metrics, see [2].

4 CONCLUSION

This work motivates and demonstrates the use of a piggyback-style algorithm for bilevel problems when the lower level problem is a convex-concave saddle-point problem. The approach was applied to two classical regularizers in a convex imaging application. Possible extensions to other data terms or to other regularizers involving linear operators are straightforward. Future work could tackle the extensions to more complex problems such as in 3D and minimal surface problems, as well as an error analysis of the outer bilevel problem depending on the (uniqueness of the) saddle-point of the lower level problem.

REFERENCES

[1] Bogensperger, L., Chambolle, A., & Pock, T. (2022). Convergence of a Piggyback-style method for the differentiation of solutions of standard saddle-point problems. SIAM Journal on Mathematics of Data Science, 4(3), 1003-1030.

[2] Bogensperger, L., Chambolle, A., Effland, A., & Pock, T. (2023). Learned Discretization Schemes for the Second-Order Total Generalized Variation. arXiv preprint arXiv:2303.09349.

[3] Bredies, K., Kunisch, K., & Pock, T. (2010). Total generalized variation. SIAM Journal on Imaging Sciences, 3(3), 492-526.

[4] Chambolle, A., & Pock, T. (2021). Learning consistent discretizations of the total variation. SIAM Journal on Imaging Sciences, 14(2), 778-813.

[5] Condat, L. (2017). Discrete total variation: New definition and minimization. SIAM Journal on Imaging Sciences, 10(3), 1258-1290.

[6] Griewank, A., & Faure, C. (2003). Piggyback differentiation and optimization. In Large-scale PDE-constrained optimization (pp. 148-164). Springer, Berlin, Heidelberg.

[7] Hosseini, A., & Bredies, K. (2022). A Second-Order TGV Discretization with Some Invariance Properties. arXiv preprint arXiv:2209.11450.

[8] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

[9] Kobler, E., Klatzer, T., Hammernik, K., & Pock, T. (2017). Variational networks: connecting variational methods and deep learning. In Pattern Recognition: 39th German Conference, GCPR 2017, Basel, Switzerland, September 1215, 2017, Proceedings 39 (pp. 281-293). Springer International Publishing.

[10] Kobler, E., Effland, A., Kunisch, K., & Pock, T. (2020). Total deep variation for linear inverse problems. In Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition (pp. 7549-7558).

[11] Kutyniok, G., Lim, W. Q., & Reisenhofer, R. (2016). Shearlab 3D: Faithful digital shearlet transforms based on compactly supported shearlets. ACM Transactions on Mathematical Software (TOMS), 42(1), 1-42.

[12] Samuel, K. G., Tappen, M. F. (2009, June). Learning optimized MAP estimates in continuously-valued MRF models. In 2009 IEEE Conference on Computer Vision and Pattern Recognition (pp. 477-484). IEEE.