Asymptotic analysis and truncated backpropagation for the unrolled primal-dual algorithm

Christoph Brauer*†, Dirk A. Lorenz*
*Institute of Analysis and Algebra, Technical University of Braunschweig, Braunschweig, Germany
†Institute of Lightweight Systems, German Aerospace Center, Stade, Germany
Email: {ch.brauer, d.lorenz}@tu-braunschweig.de

***Abstract*—Algorithm unrolling combines the advantages of model based optimization with the flexibility of data-based methods by adapting a parameterized objective to a distribution of problem instances from a finite sample from that distribution. At inference time, a fixed number of iterations of a suitable optimization algorithm is used to make predictions on unseen data. To compute gradients for learning, the last iterate is differentiated with respect to the parameters by backpropagation schemes that get expensive when the number of unrolled iterations gets large. Therefore, only few unrolled iterations are used which compromises the claimed interpretability in terms of the underlying optimization objective. In this work, we consider convex objective functions, derive an explicit limit of the parameter gradients for a large number of unrolled iterations, derive a training procedure that is computationally tractable and retains interpretability, and show the effectiveness of the method using the example of speech dequantization.**

***Index Terms*—unrolling, learning to optimize, variational problems, convex optimization, speech dequantization**

I. Introduction

We consider the task to recover an unknown ground truth from observations $x \in \mathbb{R}^n$ by means of the variational optimization [1] problem

$$\hat{y} \in \operatorname*{argmin}_{y\in\mathbb{R}^n} F(Ky) + G(y - x) \tag{1}$$

where $F : \mathbb{R}^k \to \mathbb{R}$ and $G : \mathbb{R}^n \to \mathbb{R}$ are proper, convex and lower semi-continuous functions and $K \in \mathbb{R}^{k\times n}$ is a matrix. The latter is assumed to be unknown and shall be learned from data, whereas the aforementioned functions are fixed. This induces a bilevel optimization [2] problem

$$\min_{K\in\mathbb{R}^{k\times n}} \sum_{i=1}^{m} \ell(y_i, \hat{y}_i) \quad \text{s.t. } \forall i : \hat{y}_i \in S(K, x_i) \tag{2}$$

with training data (x_i, y_i) and a loss function ℓ, and $S(K, x)$ denotes the set of solutions of (1). Computing a solution of (2) can be hard for various reasons. First, there is often no well defined solution operator S and hence, it is infeasible to rewrite (2) in terms of an unconstrained objective. Second, in order to perform gradient descent, one needs to differentiate S with respect to K. If the objective in (1) is smooth and attains a unique minimizer, this can theoretically be accomplished by means of implicit differentiation which can, however, still be numerically expensive.

A. Algorithm unrolling

The approach of unrolling [3]–[6] manages without differentiation through the lower-level problem. Instead, iterates of a suitable optimization algorithm for (1) are used to approximate and replace the solution operator in (2). As a result, the bilevel problem can be recast as an unconstrained problem

$$\min_{K\in\mathbb{R}^{k\times n}} \sum_{i=1}^{m} \ell(y_i, A^L(K, x_i)) \tag{3}$$

where $A^L(K, x_i)$ is the L-th iterate of the chosen algorithm applied to (1) with input data x_i and linear operator K. This term directly replaces the solution operator in (2), but with the important difference that the former is single-valued and can thus be shifted from the constraint into the loss. A well-known and customized algorithm for (1) is the primal-dual algorithm proposed by Chambolle and Pock [7] which essentially performs alternating updates of primal and dual variables y^l and ψ^l, respectively, as well as an extrapolation of the primal iterate. In our subsequent analysis of (3) we choose $A^L(K, x_i) = y_i^L$ to be the L-th iterate of the primal-dual algorithm applied to (1) with fixed linear operator K and data x_i. As long as the context is clear, we omit the sample index i for ease of notation. The primal-dual algorithm is outlined in Algorithm 1. The convergence proof in [7] requires $\sigma\tau\|K\|^2 < 1$ and $\theta = 1$. In the dual update, F^* denotes the convex conjugate of F and finally, $\operatorname{prox}_{\sigma F^*}$ and $\operatorname{prox}_{\tau G}$ denote respective proximal operators.

B. Previous work and contribution

The authors of the seminal work [8] proposed LISTA, an unrolled version of the iterative shrinkage-thresholding algorithm (ISTA) [9] with learned dictionaries for sparse coding. In particular, they showed that the number of iterations required for a given accuracy of the predicted codes can be reduced significantly by learning parts of the algorithm. This motivated a line of work in which theoretical aspects, enhancements and variants of LISTA were addressed. Other than ISTA, the unrolling of different optimization algorithms like Chambolle-Pock [10], proximal interior point methods [11] and plain gradient descent [12] have been investigated, just to name a few. Applications include communication systems [13], speech [14] and image [15] processing, as well as compressive sensing

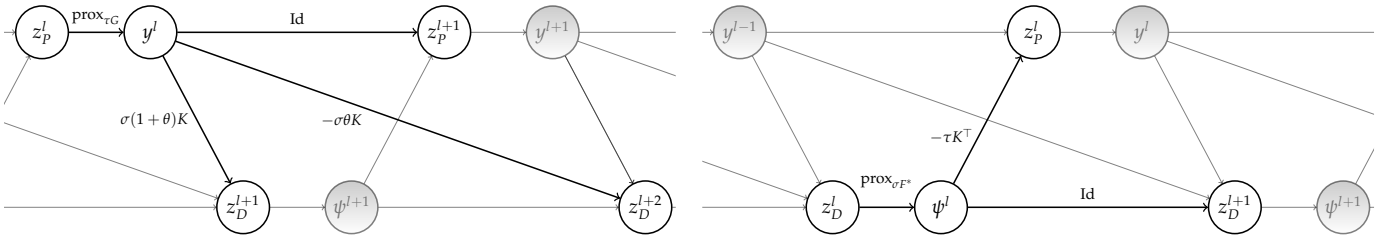

Fig. 1. Computation of primal (left) and dual (right) iterates in Algorithm 1

Algorithm 1 Primal-dual algorithm [7]

Choose $\sigma, \tau > 0$ and $\theta \in [0, 1]$
Initialize $y^0 = \overline{y}^0 = x$ and $\psi^0 = 0$
for $l = 0, \dots, L-1$ **do**

$$z_D^{l+1} = \psi^l + \sigma K \overline{y}^l \qquad \text{(Dual Update)}$$
$$\psi^{l+1} = \text{prox}_{\sigma F^*}(z_D^{l+1})$$
$$z_P^{l+1} = y^l - \tau K^\top \psi^{l+1} \qquad \text{(Prim. Update)}$$
$$y^{l+1} = \text{prox}_{\tau G}(z_P^{l+1} - x) + x$$
$$\overline{y}^{l+1} = y^{l+1} + \theta(y^{l+1} - y^l) \qquad \text{(Extrapolation)}$$

end for

Algorithm 2 Backpropagated gradients

Choose σ, τ and θ as in Algorithm 1
Adopt $y^L, z_P^1, \dots, z_P^L, z_D^1, \dots, z_D^L$ from Algorithm 1
Initialize $\delta_P^{L+1} = \nabla_{\hat{y}}\, \ell(y, y^L)$
Initialize $\delta_D^{L+1} = \overline{\delta}_D^{L+1} = 0$
for $l = L, \dots, 1$ **do**

$$\delta_P^l = \mathcal{J}_{\text{prox}_{\tau G}}(z_P^l - x)^\top (\delta_P^{l+1} + \sigma K^\top \overline{\delta}_D^{l+1}) \tag{4}$$
$$\delta_D^l = \mathcal{J}_{\text{prox}_{\sigma F^*}}(z_D^l)^\top (\delta_D^{l+1} - \tau K \delta_P^l) \tag{5}$$
$$\overline{\delta}_D^l = \delta_D^l + \theta(\delta_D^l - \delta_D^{l+1}) \tag{6}$$

end for

[16]. We refer to [3]–[6] for a comprehensive list of references, including further algorithms and application domains.

On the one hand, for several reasons, the number of unrolled iterations is usually chosen to be relatively small in practice. First, fewer iterations induce faster inference. Second, memory requirements during training scale linearly in the number of unrolled iterations. And third, state-of-the-art results are yielded with few iterations. On the other hand, due to their close connection with underlying optimization problems, unrolling methods are often attributed a high degree of interpretability [4], [17], although the connection between problem and unrolled algorithm may be vague when the number of unrolled iterations is insufficient for convergence. This aspect can be mitigated by deep equilibrium architectures [18], [19] which use fixed-point equations to virtually address the unrolling of infinitely many iterations. In this work, we take up a related point of view. In Section II, we provide an asymptotic analysis of unrolled Chambolle-Pock and show that, in the limiting case $L \to \infty$, parameter gradients are independent of primal-dual iterates (y^l, ψ^l) and depend only on the respective limits (y^*, ψ^*). Further, we show that under mild assumptions, gradients computed during backpropagation constitute a vanishing sequence. From the latter, we derive a truncated backpropagation [20]–[22] scheme that can be utilized to make the unrolling of large numbers of iterations computationally tractable. Our investigation is similar to that in [20], except that we focus specifically on variational lower-level problems. Section III illustrates numerical experiments that demonstrate efficiency and effectiveness of the proposed method, and highlight that the unrolling of many iterations is favorable in terms of interpretability.

II. Theoretical analysis

Lemma 1. *The gradients*

$$\delta_P^l = \nabla_{z_P^l}\, \ell(y, y^L) \quad \textit{and} \tag{7}$$
$$\delta_D^l = \nabla_{z_D^l}\, \ell(y, y^L) \tag{8}$$

can be computed recursively as outlined in Algorithm 2 and it holds that

$$\nabla_K\, \ell(y, y^L) = \sum_{l=1}^{L} \sigma \delta_D^l (\overline{y}^{l-1})^\top - \tau \psi^l (\delta_P^l)^\top \,. \tag{9}$$

Proof. In order to compute (7) and (8) recursively, we first observe that z_P^l affects z_P^{l+1}, z_D^{l+1} and z_D^{l+2} all through y^ℓ (see Figure 1 left). Analogously, z_D^l goes into both z_D^{l+1} and z_P^l via ψ^l (see Figure 1 right). As a consequence, we can apply the chain rule analogous to the well-known backpropagation procedure for neural networks [23], rearrange the terms slightly, and obtain (4)–(6). Finally, as K affects the objective exactly through $z_P^1, \dots, z_P^L$ and $z_D^1, \dots, z_D^L$, one more application of the chain rule gives us (9). □

Up to this point, the number of iterations L has been fixed. Next, we consider the limiting case $L \to \infty$. Therefore, we have to take into account that backpropagated gradients δ_P^l and δ_D^l do not exclusively depend on the iteration index l but also on the total number of iterations. To see that, recall that the initialization δ_P^{L+1} in Algorithm 2 depends on L and hence also the recursive sequence to compute backpropagated gradients. We adapt our notation accordingly and write $\delta_P^{l,L}$ and $\delta_D^{l,L}$ in the following.

Theorem 2. *Suppose that there exist constants $c \geq 0$ and $0 \leq \kappa < 1$ such that*

$$\|\delta_D^{l,L}\| \leq c\kappa^{L-l} \quad \textit{and} \quad \|\delta_P^{l,L}\| \leq c\kappa^{L-l} \tag{10}$$

hold for arbitrary L and $l \in \{1, \dots, L\}$. Then, the limits

$$\Delta_P \coloneqq \lim_{L\to\infty} \sum_{l=1}^{L} \delta_P^{l,L} \quad \textit{and} \quad \Delta_D \coloneqq \lim_{L\to\infty} \sum_{l=1}^{L} \delta_D^{l,L} \tag{11}$$

exist and are finite. Moreover, it holds that

$$\lim_{L\to\infty} \nabla_K \, \ell(y, y^L) = \sigma \Delta_D (y^*)^\top - \tau \psi^*(\Delta_P)^\top . \tag{12}$$

Proof. First, we use (10) and deduce

$$\begin{aligned} \lim_{L\to\infty} \sum_{l=1}^{L} \|\delta_P^{l,L}\| &\leq \lim_{L\to\infty} \sum_{l=1}^{L} c\kappa^{L-l} \\ &= \lim_{L\to\infty} c\frac{1-\kappa^L}{1-\kappa} = \frac{c}{1-\kappa} < \infty \, . \end{aligned}$$

Hence, the limits (11) exist and are finite. To prove (12), we start with (9) from Lemma 1 and obtain

$$\begin{aligned} \nabla_K \, \ell(y, y^L) &= \sum_{l=1}^{L} \sigma \delta_D^{l,L} (\overline{y}^{l-1})^\top - \tau \psi^l (\delta_P^{l,L})^\top \\ &= \underbrace{\left[\sum_{l=1}^{L} \sigma \delta_D^{l,L} (y^*)^\top - \tau \psi^* (\delta_P^{l,L})^\top \right]}_{\xrightarrow{L\to\infty} (12)} - \sigma r_1^L + \tau r_2^L \, . \end{aligned}$$

Thus, it remains to show that both terms

$$r_1^L = \sum_{l=1}^{L} \delta_D^{l,L} (y^* - \overline{y}^{l-1})^\top \text{ and } r_2^L = \sum_{l=1}^{L} (\psi^* - \psi^l)(\delta_P^{l,L})^\top$$

vanish with $L \to \infty$. To see that, let $\varepsilon > 0$. As we have $\overline{y}^l \to y^*$ for $l \to \infty$, there exists a $\lambda \in \mathbb{N}$ such that

$$\forall l \geq \lambda : \|y^* - \overline{y}^{l-1}\| < \frac{\varepsilon}{\lim_{L\to\infty} \sum_{l=1}^{L} \|\delta_D^{l,L}\|} \, . \tag{13}$$

Using (10) and (13), we further obtain

$$\begin{aligned} \|r_1^L\| &\leq \sum_{l=1}^{L} \|\delta_D^{l,L}\| \|y^* - \overline{y}^{l-1}\| \\ &= \sum_{l=1}^{\lambda-1} \|\delta_D^{l,L}\| \|y^* - \overline{y}^{l-1}\| + \sum_{l=\lambda}^{L} \|\delta_D^{l,L}\| \|y^* - \overline{y}^{l-1}\| \\ &\leq \underbrace{c \max_{l<\lambda} \{\|y^* - \overline{y}^{l-1}\|\}}_{\text{const.}} \underbrace{\sum_{l=1}^{\lambda-1} \kappa^{L-l}}_{\xrightarrow{L\to\infty} 0} + \underbrace{\frac{\varepsilon \sum_{l=\lambda}^{L} \|\delta_D^{l,L}\|}{\lim_{L\to\infty} \sum_{l=1}^{L} \|\delta_D^{l,L}\|}}_{\xrightarrow{L\to\infty} \text{ const.} \, \leq \varepsilon} \end{aligned}$$

and thus, $\lim_{L\to\infty} r_1^L \leq \varepsilon$. By analogous arguments, it follows that also $\lim_{L\to\infty} r_2^L \leq \varepsilon$. Finally, letting $\varepsilon \to 0$ completes the proof. □

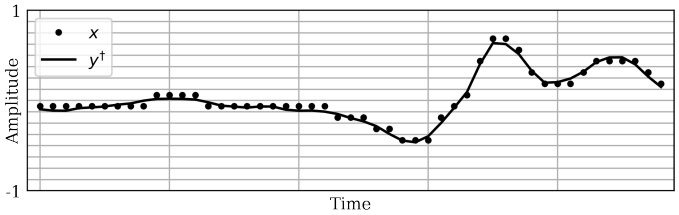

Fig. 2. Example of a ground truth signal $y^\dagger$ (solid) and its quantized counterpart x (dotted). Quantization at the bitrate $\beta = 4$ corresponds to a partition of the signal range $(-1, 1)$ into 16 quantization intervals. The entries of x are located in the centers of these intervals and the goal of dequantization is to recover $y^\dagger$ from x.

III. Numerical experiments

Our experiments are based on the variational problem

$$\hat{y} \in \operatorname*{argmin}_{y\in\mathbb{R}^n} \|Ky\|_1 \quad \text{s.t. } \|y - x\|_\infty \leq \tfrac{\eta}{2} \tag{14}$$

that has previously been considered in [14], [24] for speech dequantization. Therein, x is an observation that is known to originate from an unknown ground truth signal $y^\dagger$ through quantization at a certain bitrate (see Figure 2). The constraint in (14) reflects the prior knowledge that each value x_t is centered in a quantization interval of length η, and that the corresponding value $y_t^\dagger$ must have been in the same interval, i.e., in an $\eta/2$-environment of x_t. Moreover, $K \in \mathbb{R}^{k\times n}$ is an analysis operator and the ℓ_1-norm incorporates the assumption that the analyzed signal Ky is sparse. Regarding assumption (10) in Theorem 2 it can be shown in the special case (14) that

$$\lim_{L\to\infty} \delta_P^{l,L} \in \ker(K) \quad \text{and} \quad \lim_{L\to\infty} \delta_D^{l,L} \in \ker(K^\top) \, .$$

This can be seen by deducing that Algorithm 2 is itself an instance of Chambolle-Pock and then reverse-engineering the underlying optimization problem. As a consequence, as long as K is square and has full rank, it holds that $\delta_P^{l,L}, \delta_D^{l,L} \to 0$ for fixed l which is necessary for (10).

The dequantization objective (14) can be traced back to the more general variational problem formulation (1) by means of

$$F = \|\cdot\|_1 \quad \text{and} \quad G = I_{\|\cdot\|_\infty \leq \frac{\eta}{2}}$$

where the latter is an indicator function encoding the ℓ_∞-norm constraint. To implement Algorithms 1 and 2 we further need the convex conjugate $F^* = I_{\|\cdot\|_\infty \leq 1}$ as well as the proximal operators of F^* and G which are component-wise projections $\min(\alpha, \max(-\alpha, z))$ with $\alpha = 1$ and $\alpha = \frac{\eta}{2}$, respectively. The Jacobians required in steps (4) and (5) of Algorithm 2 are diagonal matrices. Diagonal entries with associated entry $|z_t| < \alpha$ are one and, vice versa, diagonal entries with corresponding $|z_t| > \alpha$ are zero. In the case of $|z_t| = \alpha$ both Jacobians are only subdifferentiable. With regard to this, we observed that using $\alpha = 1$ in both (4) and (5) makes the training procedure more stable and leads to better performance when η is small.

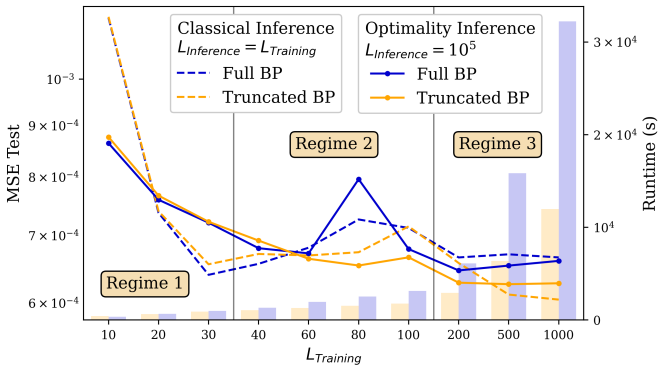

Fig. 3. Comparison of full and truncated backpropagation (with $b = 30$) in terms of the number of unrolled iterations during training L_{Training} and the resulting mean squared error on 1024 test examples. $L_{\text{Inference}}$ refers to the number of iterations performed at inference time. Barplots illustrate running times required for 20 epochs of training with full (blue) and truncated (orange) backpropagation.

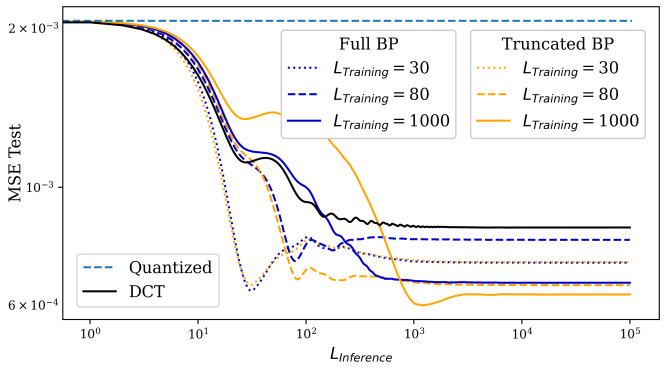

Fig. 4. Comparison of full and truncated backpropagation (with $b = 30$) in terms of the number of iterations at inference and the resulting mean squared error on 1024 test examples. In addition, the performance of $K = \text{DCT}$ (cf. [24]) and the average error of unreconstructed signals are displayed. $L_{\text{Training}} = 30$ corresponds to Regime 1 in Figure 3, $L_{\text{Training}} = 80$ represents Regime 2, and $L_{\text{Training}} = 1000$ is an example for Regime 3.

In accordance with [14], we use a training dataset of 504 sentences from the IEEE corpus [25], frame length $n = 320$ (resulting in a total of 66628 frames in the training data), choose $k = n$, and the discrete cosine transform matrix $K = \text{DCT}$ as initialization (motivated by the model-based approach in [24] where the same was used throughout). In contrast to [14], we do not learn step sizes σ, τ along with K. Instead, we fix $\sigma \lesssim 10/\|K\|$ and $\tau \lesssim 10^{-1}/\|K\|$ before each gradient update so that the above-mentioned condition $\sigma\tau\|K\|^2 < 1$ for convergence of the Chambolle-Pock algorithm is always satisfied, and we fix $\theta = 1$. For validation (early stopping) and testing we reserve separate datasets with 1024 frames each. Full length signals are normalized to the interval $(-1, 1)$ before framing and afterwards quantized at the bitrate $\beta = 4$. Thus, there are $2^\beta = 16$ quantization intervals with length $\eta = \text{length}[(-1, 1)]/2^\beta = 0.125$. All experiments were conducted using TensorFlow [26] on an Intel® Core™ i7-8550U CPU machine. We used the Adam optimizer [27] with linear learning rate decay from initially 10^{-5} to eventually 10^{-8} after 20 epochs, and batch size 32. In order to maintain full control over the differentiation procedure, we did not use the automatic differentiation capabilities of TensorFlow but implemented the Chambolle-Pock algorithm and all gradient computations manually. Our results are reproducible and all code is available under `github.com/chrbraue/primal_dual_networks`.

In Figures 3 and 4 we compare two different methods to approximate gradients of (2). Full backpropagation (**Full BP**) refers to gradient computation according to Lemma 1, i.e., to using exact gradients of the unrolled objective (3). Truncated backpropagation (**Truncated BP**) refers to an alternative approach derived from Theorem 2: We use iterates (y^L, ψ^L) to approximate (y^*, ψ^*) in (12), and a finite number of b backpropagated gradients to approximate the series Δ_P and Δ_B. As a consequence, no iterates y^l and ψ^l with $l < L$ are needed for gradient computation. By using z_P^L and z_D^L throughout in (4) and (5), the backward pass becomes, except for the initialization of δ_P^{L+1}, completely independent of the forward pass. Thus, compared to Full BP where the required working memory to store all y^l, ψ^l, z_P^l and z_D^l vectors grows linearly in L, Truncated BP requires only constant memory and makes the unrolling of many iterations feasible and efficient. Regarding inference, we assess two variants: First, we apply Algorithm 1 with the same number of iterations as during training which corresponds to the **classical inference** procedure in unrolling. Second, we apply Algorithm 1 with 10^5 iterations which means that (14) is solved close to optimal. Therefore, we refer to the latter as **optimality inference**.

With that in mind, Figure 3 can roughly be partitioned as follows: *Regime 1* is the area of few iterations where Full BP and Truncated BP perform similarly well. At thirty iterations, the error is minimal for the combination of Full BP and classical inference, while the performance decreases maximally towards optimality inference. In *Regime 2*, there is no significant increase in performance, and Full BP exhibits a slightly instable behavior. In transition to *Regime 3*, which can be considered the area of many unrolled iterations during training, errors decrease especially in the case of Truncated BP and overall best results are attained for classical and optimality inference. Figure 4 highlights the aforementioned regimes from a different angle. The gap between classical and optimality inference in Regime 1 becomes visible as a bump in the dotted lines between 10^1 and 10^2 iterations for inference. This and similar minima at $L_{\text{Inference}} \in \{30, 80, 1000\}$ indicate that classical unrolling is prone to overfitting to the number of iterations, especially when L_{Training} is too small to ensure convergence of Algorithm 1. In other words, the interpretability of the resulting predictive models in terms of the underlying optimization objective is lower in these cases. Regarding Regime 3 it becomes apparent that, although the respective models are best in terms of the yielded error, they also feature a comparatively low convergence speed here.

The running times illustrated in Figure 3 highlight two aspects. *First*, forward and backward pass are closely related instances of the primal-dual algorithm. As a consequence, the overall complexity of a foward pass followed by a full backward pass is about $2L$ times the complexity of a single iteration. In contrast, the overall complexity of a forward pass followed by a truncated backward pass is about $L + b$ times the complexity of a single iteration, where b is the number of backward steps in Truncated BP. For large L and small b (here we have fixed $b = 30$) the required running time for Full BP is hence at least twice as high as the running time of Truncated BP. *Second*, using only y^L, ψ^L, z_P^L and z_D^L instead of all respective intermediate iterates induces an additional saving of computation time in the case of Truncated BP. In view of Regime 3, it can be seen that the computational cost can be more than halved by using Truncated BP, while at the same time the test error is reduced.

While we did not explicitly track memory footprints of full and truncated backpropagation in our numerical experiments, a rough estimate can be made analogous to above. In the case of full backpropagation, all iterates y^l, ψ^l, z_P^l and z_D^l from the forward pass need to stored for the backward pass which corresponds to a total of $2L(k + n)$ variables. For the proposed truncated backpropagation scheme, as stated above, only the respective values from the very last iteration need to be provided, corresponding to a total of only $2(k+n)$ variables. Thus, in terms of the memory footprint, the savings from using truncated backpropagation scale linearly in the number of unrolled iterations.

IV. Conclusion

We provided a rigorous asymptotic analysis of gradients of the unrolled Chambolle-Pock algorithm. Our results show that, in the limiting case, gradients do not depend on intermediate iterates but only on optimal solutions. Based on that, we derived a tractable method for truncated gradient computation which was shown to outperform the usage of full gradients in terms of running time, memory footprint and reconstruction accuracy. Our empirical results obtained on speech data moreover indicate that the interpretability in terms of the optimization problem tends to get lost when the number of unrolled iterations is small. The extension of our work to a broader class of optimization algorithms and the derivation of methods that combine high interpretability and faster convergence will be subject of future work.

References

[1] O. Scherzer, M. Grasmair, H. Grossauer, M. Haltmeier, and F. Lenzen, *Variational Methods in Imaging*. Springer New York, 2009.

[2] J. Bracken and J. T. McGill, "Mathematical programs with optimization problems in the constraints," *Operations Research*, vol. 21, no. 1, pp. 37–44, 1973.

[3] S. Arridge, P. Maass, O. Öktem, and C.-B. Schönlieb, "Solving inverse problems using data-driven models," *Acta Numerica*, vol. 28, pp. 1–174, 2019.

[4] V. Monga, Y. Li, and Y. C. Eldar, "Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing," *IEEE Signal Processing Magazine*, vol. 38, no. 2, pp. 18–44, 2021.

[5] T. Chen, X. Chen, W. Chen, H. Heaton, J. Liu, Z. Wang, and W. Yin, "Learning to optimize: A primer and a benchmark," *arXiv preprint arXiv:2103.12828*, 2021.

[6] W. Yin, "Learning to optimize: Algorithm unrolling," *CVPR Tutorial*, 2022.

[7] A. Chambolle and T. Pock, "A first-order primal-dual algorithm for convex problems with applications to imaging," *Journal of mathematical imaging and vision*, vol. 40, no. 1, pp. 120–145, 2011.

[8] K. Gregor and Y. LeCun, "Learning fast approximations of sparse coding," in *Proceedings of the 27th international conference on international conference on machine learning*, 2010, pp. 399–406.

[9] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm with application to wavelet-based image deblurring," in *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2009, pp. 693–696.

[10] J. Adler and O. Öktem, "Learned primal-dual reconstruction," *IEEE transactions on medical imaging*, vol. 37, no. 6, pp. 1322–1332, 2018.

[11] C. Bertocchi, E. Chouzenoux, M.-C. Corbineau, J.-C. Pesquet, and M. Prato, "Deep unfolding of a proximal interior point method for image restoration," *Inverse Problems*, vol. 36, no. 3, p. 034005, 2020.

[12] C. Brauer, N. Breustedt, T. de Wolff, and L. D. A., "Learning variational models with unrolling and bilevel optimization," *arXiv preprint arXiv:2209.12651*, 2022.

[13] A. Balatsoukas-Stimming and C. Studer, "Deep unfolding for communications systems: A survey and some new directions," in *2019 IEEE International Workshop on Signal Processing Systems (SiPS)*. IEEE, 2019, pp. 266–271.

[14] C. Brauer, Z. Zhao, D. Lorenz, and T. Fingscheidt, "Learning to dequantize speech signals by primal-dual networks: an approach for acoustic sensor networks," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 7000–7004.

[15] Y. Li, M. Tofighi, V. Monga, and Y. C. Eldar, "An algorithm unrolling approach to deep image deblurring," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 7675–7679.

[16] Y. Yang, J. Sun, H. Li, and Z. Xu, "Deep admm-net for compressive sensing mri," in *Proceedings of the 30th international conference on neural information processing systems*, 2016, pp. 10–18.

[17] J. Zhang and B. Ghanem, "Ista-net: Interpretable optimization-inspired deep network for image compressive sensing," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 1828–1837.

[18] S. Bai, J. Z. Kolter, and V. Koltun, "Deep equilibrium models," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[19] 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.

[20] A. Shaban, C.-A. Cheng, N. Hatch, and B. Boots, "Truncated backpropagation for bilevel optimization," in *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR, 2019, pp. 1723–1732.

[21] P. Vicol, L. Metz, and J. Sohl-Dickstein, "Unbiased gradient estimation in unrolled computation graphs with persistent evolution strategies," in *International Conference on Machine Learning*. PMLR, 2021, pp. 10 553–10 563.

[22] C. Tallec and Y. Ollivier, "Unbiasing truncated backpropagation through time," *arXiv preprint arXiv:1705.08209*, 2017.

[23] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[24] C. Brauer, T. Gerkmann, and D. Lorenz, "Sparse reconstruction of quantized speech signals," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 5940–5944.

[25] P. C. Loizou, *Speech enhancement: theory and practice*. CRC press, 2007.

[26] M. A. et al., "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

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