MOTIVATING BILEVEL APPROACHES TO FILTER LEARNING: A CASE STUDY

Caroline Crockett and Jeffrey A. Fessler

Department of EECS, University of Michigan, Ann Arbor, Michigan, USA

2021 IEEE International Conference on Image Processing (ICIP) | 978-1-6654-4115-5/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICIP42928.2021.9506489

ABSTRACT

The recent trend in regularization methods for inverse problems is to replace handcrafted sparsifying operators with data-driven approaches. Although using such machine learning techniques often improves image reconstruction methods, the results can depend significantly on the learning methodology. This paper compares two supervised learning methods. First, the paper considers a transform learning approach and, to learn the transform, introduces a variant on the Procrustes method for wide matrices with orthogonal rows. Second, we consider a bilevel convolutional filter learning approach. Numerical experiments show the learned transform performs worse for denoising than both the handcrafted finite difference transform and the learned filters, which perform similarly. Our results motivate the use of bilevel learning.

Index Terms— bilevel, co-sparse, denoising

1 Introduction

Image reconstruction problems are active research areas. New models and algorithms typically attempt to reduce computation time or achieve better results, according to some quality metric, under given application-specific constraints. For medical image reconstruction, the ultimate goal is to provide doctors with information while minimizing patient exposure to radiation, working toward the goal of "as low as reasonably achievable" [1] in X-ray computed tomography (CT) imaging, or reducing scan time in MRI.

To achieve lower radiation doses and/or faster scans, one must decrease the X-ray source intensity or undersample the data. This leads to an under-determined system with fewer knowns than unknowns. Thus, one must make some assumption about the images to reconstruct them. These assumptions come in the form of different models, or priors. Common models are total variation (TV) for approximately piecewise constant (PWC) images and sparsity of a wavelet transform (*e.g.*, [2]). These are "handcrafted" models because an engineer designed them, often using intuitive image features. Handcrafted models are often designed to work for many different types of images and to have computational advantages.

As an alternative to handcrafted models, learned priors can improve image quality [3], with the trade-off of increased training time and possibly decreased generality. Learned priors are becoming more popular in part due to increased computational resources and access to large, labeled training datasets [3]. While (deep) learning approaches have seen much success, they have limited theoretical guarantees and often lack explainability in inverse problems.

To seek further insights into handcrafted versus learned signal models, this paper considers a simplification of the reconstruction problem: denoising 1D signals. Section 2 reviews background material. Section 3 describes two models and introduces a new method of learning a small number of orthogonal filters. Finally, Section 4 experimentally compares the learned filters with a handcrafted, finite difference filter for a simple class of PWC signals. We do not purport to improve on state-of-the-art results or to offer an especially novel denoising method. Instead, this paper investigates *how the structure of a learning problem impacts the learned solutions* by examining a simple class of signals. These insights could apply in more complex image reconstruction tasks. In particular, the results motivate the use of bilevel approaches.

2 Background

Convolution (with circular boundary conditions) is denoted as $\circledast$. We write vectors as column vectors, use bold to denote matrices (uppercase letters) and vectors (lowercase letters), and use Julia style notation for functions, where $f.(\mathbf{x})$ means the function f applied element-wise to $\mathbf{x}$.

Reconstruction aims to find $\hat{\mathbf{x}}$ to match the observed data, $\mathbf{y}$, and satisfy prior assumptions. Written as a cost function:

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}} \frac{1}{2} \|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2 + \beta R(\mathbf{x}). \quad (1)$$

The first term encourages consistency with observations via the system model, $\mathbf{A}$; the second, regularization term encourages $\hat{\mathbf{x}}$ to match the prior assumptions; and the hyperparameter, β, controls the relative importance of the two terms.

This work compares regularizers with co-sparse transforms or filters. While synthesis approaches to sparsity are common, recent co-sparse models show promising results, *e.g.*, [4], [5]. The model in co-sparse transform learning is that a transform matrix, when left-multiplied, sparsifies patches of a signal, *i.e.*, $\mathbf{T}\mathbf{s}_l$ tends to be sparse, where $\mathbf{s}_l$ is one of L training patches. Thus, the transform learning goal is to find

$$\hat{\mathbf{T}} = \underset{\mathbf{T} \in \mathbb{T}}{\operatorname{argmin}} \sum\nolimits_{l=1}^{L} \|\mathbf{T}\mathbf{s}_l\|_0 = \underset{\mathbf{T} \in \mathbb{T}}{\operatorname{argmin}} \|\mathbf{T}\mathbf{S}\|_0 , \quad (2)$$

This work was supported in part by NIH grants R01 EB023618 and NSF grant IIS 1838179. Author contacts: {cecroc,fessler}@umich.edu

where D is the patch size and $\mathbf{S} \in \mathbb{C}^{D\times L}$ is a matrix with one training patch per column. To avoid trivial solutions such as the zero filter or repeated filters, $\mathbb{T} \subseteq \mathbb{C}^{K\times D}$ may be defined, *e.g.*, as the set of matrices with orthonormal rows [6]. The co-sparse filter learning model is equivalent to (2) for corresponding boundary conditions on the convolution and patch extraction. Specifically, the filter perspective views each row of $\mathbf{T}$ as a filter, $\mathbf{h}_k$, where $\mathbf{h}_k \circledast \mathbf{x}$ is assumed sparse.

Section 3 discusses two approaches to learning the elements in $\mathbf{T}$ or $\mathbf{h}$ from training data. The first method learns a transform that (approximately) sparsifies training data, using a relaxed version of (2). The learned transforms can then be used in a regularizer for (1), typically with the assumption that the learned transform separates noise (assumed to not be sparsified by $\mathbf{T}$) from signal (assumed to be sparsified by $\mathbf{T}$).

The second method learns filters to best denoise the training data, according to the following task-based, bilevel set-up

$$\hat{\gamma} = \underset{\gamma}{\operatorname{argmin}} \sum_{j=1}^{J} \frac{1}{2} \|\hat{\mathbf{x}}_j(\gamma) - \mathbf{s}_j\|_2^2 \text{ where} \tag{3}$$

$$\hat{\mathbf{x}}_j(\gamma) = \underset{\mathbf{x}}{\operatorname{argmin}} \frac{1}{2} \|\mathbf{x} - \mathbf{y}_j\|_2^2 + \sum_{k=1}^{K} e^{\beta_k} \mathbf{1}' \psi.(\mathbf{h}_k \circledast \mathbf{x}) \tag{4}$$

where γ contains all learnable parameters ($\mathbf{h}_k$ and β_k), ψ is a sparsity-promoting function, and $\mathbf{y}_j$ is a simulated, noisy version of the jth training sample, $\mathbf{s}_j$. Previous work [7]–[9] also considers the bilevel set-up to learning co-sparse models for reconstruction. This paper is inspired by those works and provides additional motivation for the bilevel formulation by directly comparing the two learning approaches.

3 Learning approaches

3.1 Constrained transform learning

One can relax (2) by splitting the argument [4]:

$$\hat{\mathbf{T}} = \underset{\mathbf{T}\in\mathbb{T}}{\operatorname{argmin}} \min_{\mathbf{z}_l \in \mathbb{C}^K} \sum_{l=1}^{L} \frac{1}{2} \|\mathbf{T}\mathbf{s}_l - \mathbf{z}_l\|_2^2 + \lambda \|\mathbf{z}_l\|_0 . \tag{5}$$

The tuning parameter λ trades-off enforcing sparsity (larger λ) of the sparse codes, $\mathbf{z}$, and forcing $\mathbf{T}\mathbf{s}_l \approx \mathbf{z}_l$ (smaller λ).

Using block coordinate minimization (BCM) to optimize (5), the updates at iteration n are:

$$\begin{aligned} \mathbf{z}_l^{(n)} &= \underset{\mathbf{z}\in\mathbb{C}^K}{\operatorname{argmin}} \sum_{l=1}^{L} \frac{1}{2} \|\mathbf{T}^{(n-1)}\mathbf{s}_l - \mathbf{z}\|_2^2 + \lambda \|\mathbf{z}\|_0 \\ &= \text{prox.}(\mathbf{T}^{(n-1)}\mathbf{s}_l) \end{aligned} \tag{6}$$

$$\mathbf{T}^{(n)} = \underset{\mathbf{T}\in\mathbb{T}}{\operatorname{argmin}} \|\mathbf{T}\mathbf{S} - \mathbf{Z}^{(n)}\|_{\mathrm{F}}^2, \tag{7}$$

where $\mathbf{Z} \in \mathbb{C}^{K\times L}$ contains the sparse codes in its columns. The sparse code update (6) is a proximal problem. The proximal operator for the 0-norm in (5) is hard-thresholding [10] which is cheap to compute despite being non-convex.

When $\mathbb{T}$ describes matrices with orthonormal filters, the transform update (7) is almost a standard Procrustes problem,

$$\hat{\mathbf{Q}} = \underset{\mathbf{Q}:\mathbf{Q}'\mathbf{Q}=\mathbf{Q}\mathbf{Q}'=\boldsymbol{I}}{\operatorname{argmin}} \|\mathbf{B} - \mathbf{Q}\mathbf{A}\|_{\mathrm{F}}^2, \tag{8}$$

Algorithm 1 Learning a wide transform matrix with orthonormal rows. Inputs: an initialization for the transform ($\mathbf{T}^{(0)} \in \mathbb{C}^{K\times D}$ where $K \leq D$), a matrix of training signal patches ($\mathbf{S} \in \mathbb{C}^{D\times L}$), the number of iterations to perform (N), and the proximal operator of ψ (prox).

1: **procedure** PROCRUSTES-WIDE($\mathbf{T}^{(0)}$, $\mathbf{S}$, N, prox)
2: $\quad$ $\mathbf{Q}$, $\mathbf{R}$ = qr($\mathbf{T}^{(0)'}$) ▷ QR decomposition
3: $\quad$ $\tilde{\mathbf{T}}^{(0)} = \mathbf{Q}'$
4: $\quad$ **for** $n = 1$ to N **do** ▷ Perform N iterations
5: $\quad\quad$ $\mathbf{Z} = \tilde{\mathbf{T}}^{(n-1)}\mathbf{S}$
6: $\quad\quad$ $\mathbf{Z}_{1:K,:} =$ prox.$(\mathbf{Z}_{1:K,:})$
7: $\quad\quad$ $\mathbf{U}$, $\mathbf{\Sigma}$, $\mathbf{V}$ = svd($\mathbf{Z}\mathbf{S}'$)
8: $\quad\quad$ $\tilde{\mathbf{T}}^{(n)} = \mathbf{U}\mathbf{V}'$
9: $\quad$ **end for**
10: $\quad$ **return** $\tilde{\mathbf{T}}^{(N)}_{1:K,:}$ ▷ Remove dummy rows
11: **end procedure**

with solution $\hat{\mathbf{Q}} = \mathbf{U}\mathbf{V}'$ where $\mathbf{U}$ and $\mathbf{V}$ are the left and right singular vectors of $\mathbf{B}\mathbf{A}'$. However $\mathbf{T}$ is often rectangular, and thus not unitary. When $\mathbf{T}$ has orthonormal columns, (7) is a generalized Procrustes problem [11]. However, to compare to TV approaches, we want to consider cases where $\mathbf{T}$ is wide.

Alg. 1 solves (7) when $\mathbf{T}$ is wide with orthonormal rows ($K < D$) by learning a unitary, $D \times D$ transform with the last $D-K$ rows containing "dummy" (irrelevant) filters. The $\mathbf{T}$ update in line 8 uses the standard Procrustes problem (8). Mathematically, this approach defines $\tilde{\mathbf{T}} \in \mathbb{C}^{D\times D}$ such that the first K rows contain the filters from $\mathbf{T}$ and the remaining rows contain the dummy filters. In terms of $\tilde{\mathbf{T}}$, (5) is

$$\hat{\tilde{\mathbf{T}}} = \underset{\tilde{\mathbf{T}}\in\tilde{\mathbb{T}}}{\operatorname{argmin}} \min_{\mathbf{z}_l \in \mathbb{C}^D} \sum_{l=1}^{L} \frac{1}{2} \|\tilde{\mathbf{T}}\mathbf{s}_l - \mathbf{z}_l\|_2^2 + \lambda \|\mathbf{W}\mathbf{z}_l\|_0 , \tag{9}$$

where $\mathbf{W} = \begin{bmatrix} \boldsymbol{I} & \mathbf{0} \end{bmatrix} \in \mathbb{R}^{K\times D}$ selects the first K elements of $\mathbf{z}$ and $\tilde{\mathbb{T}}$ is the set of $D \times D$ unitary matrices.

3.2 Bilevel filter learning

To learn hyperparameters, $\hat{\gamma}$, a general bilevel problem is:

$$\hat{\gamma} = \underset{\gamma}{\operatorname{argmin}} \sum_{j=1}^{J} l(\hat{\mathbf{x}}_j(\gamma); \gamma, \mathbf{s}_j), \text{ where} \tag{10}$$

$$\hat{\mathbf{x}}_j(\gamma) = \underset{\mathbf{x}}{\operatorname{argmin}} \Phi(\mathbf{x}; \gamma, \mathbf{y}_j). \tag{11}$$

Here, l is an upper-level loss function (*e.g.*, mean square error between the jth training signal and $\hat{\mathbf{x}}_j$) and Φ is a lower-level objective function for learning $\hat{\mathbf{x}}_j$ that typically matches the end-application of the learned hyperparameters (*e.g.*, a reconstruction problem). We assume the training data is noiseless, then define a new variable, $\mathbf{y}_j = \mathbf{A}\mathbf{s}_j + \mathbf{n}$ that is the forward transform of the training data with noise, to use in the lower-level problem. The bilevel approach does not require a filter diversity constraint, since it should learn filters that are best-suited to the lower-level task.

The strategy in bilevel approaches is to calculate the gradient of l with respect to γ and then use a gradient descent

method to minimize γ. One approach to finding the gradient of the upper-level loss function is to apply implicit differentiation to find an expression for the gradient of Φ with respect to γ [12]. When the lower-level cost function is minimized by a differentiable optimization algorithm, an alternative approach is to "unroll" the lower-level optimization algorithm then calculate the gradient using backpropagation, as is common in convolutional neural networks (CNNs). However, unlike in CNNs, if we unroll a sufficiently large number of iterations to reach convergence, as in [13], the unrolling is still tied to an optimization method with an explainable cost function. The reverse method of calculating the gradient is given in [14].

To apply the co-sparse model in the bilevel setting with unrolling, we can no longer use the 0-norm, as the corresponding algorithm (6) is not differentiable. Instead, following [9], we define the bilevel problem as given in (3)-(4) for some sparsity-penalizing function ψ that is twice differentiable. The vector of hyperparameters to learn, γ, contains all elements in β and $\mathbf{h}$.

Using gradient descent on (4), the differentiable lower-level update is $\mathbf{x}^{(n+1)} = \mathbf{x}^{(n)} - \frac{1}{L}\nabla\Phi(\mathbf{x}^{(n)};\gamma,\mathbf{y})$, where L is the Lipschitz constant of Φ. For a given outer-level step, the filters and tuning parameters are constant and

$$L = 1 + e^{\beta_0} L_\psi \sum\nolimits_k e^{\beta_k} \left\|h_k\right\|^2, \tag{12}$$

where L_ψ is the Lipschitz constant of the sparsity promoting function ψ. We update L after every gradient step on the hyperparameters and we run a sufficient number of iterations to reach convergence for each L.

4 Experiments

Our training data set is noiseless PWC 1D signals (1,024 for transform learning and 128 for bilevel filter learning). Each signal has 32 elements with exactly three "jumps," which are indices where the left difference is non-zero. There is at most one jump in any given length-4 patch. The signal values are uniformly distributed over [-1, 1]. We assume circular boundary conditions. We define $\mathbb{T}$ as the set of single, length-4 filters with unit norm, i.e., $K = 1$ and $D = 4$. We quantify disparities between learned and handcrafted filters using the formula for the angle between vectors: $\cos^{-1}\left(|\langle \mathbf{z}_1, \mathbf{z}_2\rangle| / \left\|\mathbf{z}_1\right\| \left\|\mathbf{z}_2\right\|\right)$.

Our test data includes $\mathbf{s}_1$, a length-1000 input signal with 50 jumps (a slight generalization of our training data), and $\mathbf{s}_2$, a collection of 128 signals created in the same way as the training data but with a different random seed. All test signal values are uniformly distributed over [-1, 1]. The corresponding noisy input signals, $\mathbf{y}_1$ and $\mathbf{y}_2$ are the true signal plus mean zero Gaussian noise with a standard deviation of 0.1. We report the average root mean square error (RMSE), $\sqrt{\frac{1}{N}\left\|\hat{\mathbf{x}} - \mathbf{s}\right\|^2}$ where N is the signal length.

For the PWC signals considered here, we expect that the best sparsifying filter corresponds to the "TV transform:"

$$\mathbf{T}_{\mathrm{TV}} = \frac{1}{\sqrt{2}}\begin{bmatrix}0 & 1 & -1 & 0\end{bmatrix}, \tag{13}$$

which is the minimizer of the transform learning problem (2).

4.1 Transform learning

Using $\mathbf{T}_{\mathrm{TV}}$ as the initialization, we learned transforms for (5) using Alg. 1 for different values of λ. In-line with our empirical observations, a grid search over the free variables in $\mathbf{T}$, showed that, for large training sets,

$$\hat{\mathbf{T}} = \begin{bmatrix} -\sqrt{\frac{1-2d}{2}} & d & -d & \sqrt{\frac{1-2d}{2}}\end{bmatrix}, \tag{14}$$

which is a smoothed version of the $\mathbf{T}_{\mathrm{TV}}$ transform. Without loss of generality, we can assume $\frac{1}{2} \leq d \leq \frac{1}{\sqrt{2}}$ because of the circular shift invariance. Taking (14) as the correct form for the minimizer of (5), finding $\hat{\mathbf{T}}(\lambda)$ is a 1-D problem. Therefore, for a given λ, it is easy to sweep over d, compute the cost function, and find the global minimizer $\hat{d}$.

Fig. 1 shows the value of the cost function (5) and the minimizers, $\hat{d}$, for various λ values. As λ increases, the trend is that $\hat{d}$ decreases (the filter gets smoother). This behavior corresponds to the learned filter moving from $\mathbf{T}_{\mathrm{TV}}$ to 22.5 degrees away from $\mathbf{T}_{\mathrm{TV}}$, as seen in Fig. 2.

For denoising, we use (1) with $\mathbf{A} = \boldsymbol{I}$ and

$$R(\mathbf{x}) = \sum\nolimits_l \min_{\mathbf{z}_l} \left\|\mathbf{T}\mathbf{P}_l\mathbf{x} - \mathbf{z}_l\right\|_2^2 + \alpha \left\|\mathbf{z}_l\right\|_0, \tag{15}$$

where $\mathbf{P}_l$ is the matrix that extracts the lth patch from $\mathbf{x}$. The filter matrix, $\mathbf{T}$, is either the handcrafted filter $\mathbf{T}_{\mathrm{TV}}$ defined in (13) or a learned filter, taking the form given in (14) with $d = 0.67$ (corresponding to $\lambda = 0.23$).

We use BCM to optimize (15). The $\mathbf{x}$ update is the solution to the least squares problem

$$\begin{aligned}\mathbf{x}^{(i+1)} &= (\boldsymbol{I} + \frac{1}{\beta}\sum\nolimits_l \mathbf{P}_l'\mathbf{T}'\mathbf{T}\mathbf{P}_l)^{-1}(\mathbf{y} + \beta\sum\nolimits_l \mathbf{P}_l'\mathbf{T}'\mathbf{z}_l^{(i)}) \\ &= \frac{1}{1+\beta d}(\mathbf{y} + \beta\sum\nolimits_l \mathbf{P}_l'\mathbf{T}'\mathbf{z}_l^{(i)}),\end{aligned}$$

which simplifies because $\mathbf{T}'\mathbf{T} = \boldsymbol{I}$ by the definition of $\mathbb{T}$ and $\sum_l \mathbf{P}_l'\mathbf{P}_l = d\boldsymbol{I}$ since $\mathbf{P}$ creates patches in a circular manner. The $\mathbf{z}_l$ update is simply hard thresholding applied to $\mathbf{T}\mathbf{P}_l\mathbf{x}_l^{(i+1)}$. We alternate updating $\mathbf{x}$ and $\mathbf{z}$ for 10,000 iterations or until $\left\|\mathbf{x}^{(i+1)} - \mathbf{x}^{(i)}\right\| / \left\|\mathbf{x}^{(i+1)}\right\| < 10^{-6}$.

We did a grid search to find the parameters α and β that yield the lowest MSE for $\mathbf{s}_1$. In practice, this would require a validation data set, but we use a test signal to get an optimistic error. For $\mathbf{T}_{\mathrm{TV}}$, the best tuning parameters are $\alpha^* = 0.025$ and $\beta^* = 28$. For $\hat{\mathbf{T}}$, they are $\alpha = 0.006$ and $\beta^* = 310$. Tab. 1 reports the RMSE results.

One could also run a grid search over λ, creating a bilevel transform learning problem. While this would be feasible in our simple experiment, it would be impractical for models and tasks such as those in [6].

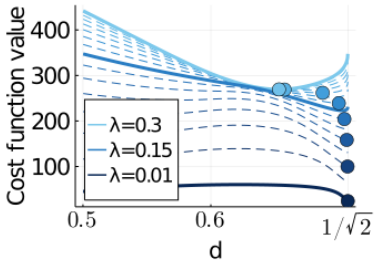

Fig. 1: Plot of the value of the cost function value (5) versus $\mathbf{T}$ as a function of d in (14) for various λs. Smaller values of d mean smoother filters. The points mark $\hat{d}$.

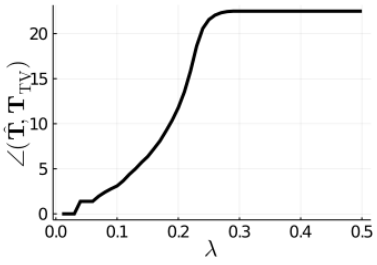

Fig. 2: Plot of the angle (in degrees) between the learned transform and $\mathbf{T}_{\mathrm{TV}}$ versus the tuning parameter in (5).

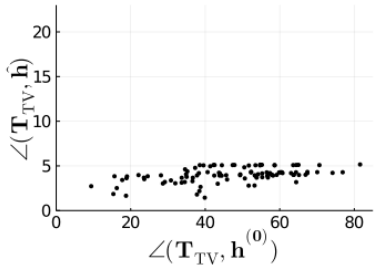

Fig. 3: Scatter plot of the angle between $\mathbf{T}_{\mathrm{TV}}$ and the learned filter, $\hat{\mathbf{h}}$, versus the angle between $\mathbf{T}_{\mathrm{TV}}$ and the randomly initialized filter, $\mathbf{h}^{(0)}$ (in degrees).

4.2 Bilevel filter learning

This section considers minimizing (3) with $\gamma = [\beta_1; \mathbf{h}]$ and $\psi(z) = \sqrt{|z|^2 + 0.1^2}$ ($L_\psi = 10$). We use Adam [15] with the default settings for the γ gradient step since a line search would be expensive (evaluating l requires solving the lower-level function). The returned minimizer is the one with the lowest loss function after 7,000 Adam iterations.

We first consider initializing $\mathbf{h}$ with $\mathbf{h}_{\mathrm{TV}} := \mathbf{T}_{\mathrm{TV}}$ and sweeping values of e^{β_1} between 0.001 and 0.35. With this informed initialization for $\mathbf{h}$, the mean and maximum angle between the learned filters and $\mathbf{T}_{\mathrm{TV}}$ are 1.3 degrees and 1.5 degrees respectively. Further, the minimum upper level loss function shows no obvious trend with the initialization of the tuning parameter, β_1.

The more interesting test of the non-convex bilevel problem is a random initialization. For this, we initialized $\mathbf{h}$ with 100 normalized Gaussian noise realizations and set $e^{\beta_1} = 1$. There is a strong positive correlation (correlation coefficient of 0.80) between the minimum loss function value and the angle between the learned filter and $\mathbf{T}_{\mathrm{TV}}$, suggesting the angle from $\mathbf{T}_{\mathrm{TV}}$ is a reasonable indicator of the denoising performance during training. Fig. 3 shows that, for a wide range of random initial filters, the learned filters are all within 1.44 to 5.16 degrees of $\mathbf{T}_{\mathrm{TV}}$. This result is promising considering the highly non-convex nature of the bilevel problem.

To test denoising performance, we use (4) with the same ψ as in training and the learned parameters. For comparison, we report the denoising performance of $\mathbf{h}_{\mathrm{TV}}$ using (4), with the filter norm and β_1 tuned (19.5 and -4 respectively) using a grid search to minimize RMSE for $\mathbf{s}_1$. Tab. 1 shows the results for the learned filters corresponding to the smallest (best) and largest (worst) training loss functions.

	Transforms		Filters (Bilevel)		
	$\mathbf{T}_{\mathrm{TV}}$	$\hat{\mathbf{T}}$	$\mathbf{h}_{\mathrm{TV}}$	$\hat{\mathbf{h}}_{\mathrm{best}}$	$\hat{\mathbf{h}}_{\mathrm{worst}}$
$\mathbf{s}_1 \in \mathbb{R}^{1000}$	4.0	6.2	4.4	5.1	6.3
$\mathbf{s}_2 \in \mathbb{R}^{32}$	5.2	8.2	5.4	5.5	6.6

Table 1: RMSE $\cdot 100$ for denoised test signals. **Left columns**: denoising using (15) with $\mathbf{T}$ being $\mathbf{T}_{\mathrm{TV}}$ or learned according to (5) for $\lambda = 0.23$. Other values of λ (not shown) also yield higher RMSE values than $\mathbf{T}_{\mathrm{TV}}$. The (smoothed) learned transform performs worse than $\mathbf{T}_{\mathrm{TV}}$. **Right columns**: denoising using the lower-level cost function (4) with $\mathbf{h}_{\mathrm{TV}}$ and the best and worst performing filters learned using the bilevel method with random initializations. The filters learned using bilevel perform better than $\hat{\mathbf{T}}(\lambda = 0.23)$ and similar to $\mathbf{T}_{\mathrm{TV}}$, especially for $\mathbf{s}_2$, which mimics the training data. Note that $\mathbf{T}_{\mathrm{TV}}$ in the left column performs better than $\mathbf{h}_{\mathrm{TV}}$ in the right column due to the zero-norm in (15) (compared to the the corner-rounded 1-norm in (4)).

5 Conclusion

We started this investigation to discover why we did not learn $\mathbf{T}_{\mathrm{TV}}$ using (5) with noiseless PWC training signals. We showed that the smoothness in the learned transform results from splitting the objective function as in (5), and that the smoothness increases as the tuning parameter increases.

By construction, the learned transform achieves a lower training loss (5) than $\mathbf{T}_{\mathrm{TV}}$. Though this might suggest that the learned transform is "better," in fact the handcrafted transform better denoises the test signals. The disparity is due to the structure of the training objective: the transform is learned to make training data approximately match sparse codes, which is best accomplished by a smooth transform, rather than to separate signal and noise for denoising.

This observation naturally leads to the task-based bilevel formulation for learning filters based on denoising performance. Our simple experimental results show the learned filters perform similar to TV, exemplifying the benefit of the bilevel approach, which is more important in problems with no obvious handcrafted solution. Although this paper did not discuss time complexity, optimizing the bilevel problem (3) requires more computation than the transform learning problem (5). However, once optimized, there is no need for a hyperparameter grid search because the learned hyperparameters were trained to work well in the lower-level task.

Bilevel for co-sparse filter learning is a relatively new research area and there are many open questions. For example, future research directions include rigorous comparisons of the two approaches to optimizing the bilevel problem (the implicit function theorem and unrolling), applying bilevel co-sparse filter learning to image reconstruction problems, examining sensitivity to initialization for more complex problems, and using application-specific upper-level loss functions.

References

[1] D. J. Brenner and E. J. Hall, "Computed tomography-An increasing source of radiation exposure," *New England Journal of Medicine*, vol. 357, no. 22, pp. 2277–84, Nov. 2007. DOI: 10.1056/NEJMra072149.

[2] S. G. Mallat, "A theory for multiresolution signal decomposition: The wavelet representation," *IEEE Trans. Patt. Anal. Mach. Int.*, vol. 11, no. 7, pp. 674–89, 1989. DOI: 10.1109/34.192463.

[3] B. Sahiner, A. Pezeshk, L. M. Hadjiiski, *et al.*, "Deep learning in medical imaging and radiation therapy," *Medical Physics*, 2018. DOI: 10.1002/mp.13264.

[4] S. Ravishankar and Y. Bresler, "Learning Sparsifying Transforms," *IEEE Trans. on Signal Process.*, vol. 61, no. 5, pp. 1072–1086, 2013. DOI: 10.1109/TSP.2012.2226449.

[5] I. Y. Chun, D. Hong, B. Adcock, and J. A. Fessler, "Convolutional analysis operator learning: Dependence on training data," *IEEE Signal Process. Lett.*, vol. 26, no. 8, pp. 1137–1141, Aug. 2019. DOI: 10.1109/LSP.2019.2921446.

[6] I. Y. Chun and J. A. Fessler, "Convolutional Analysis Operator Learning: Acceleration and Convergence," *IEEE Transactions on Image Processing*, vol. 29, pp. 2108–2122, 2020. DOI: 10.1109/TIP.2019.2937734.

[7] M. T. McCann and S. Ravishankar, *Supervised Learning of Sparsity-Promoting Regularizers for Denoising*, 2020. arXiv: 2006.05521.

[8] G. Peyré and J. M. Fadili, *Learning Analysis Sparsity Priors*, Singapour, Singapore, 2011. [Online]. Available: https://hal.archives-ouvertes.fr/hal-00542016/document.

[9] Y. Chen, T. Pock, and H. Bischof, "Learning l_1-based analysis and synthesis sparsity priors using bi-level optimization," in *Neural Information Processing Systems Confercence (NIPS)*, 2014. arXiv: 1401.4105 [cs].

[10] S. Ravishankar and Y. Bresler, "Efficient blind compressed sensing using sparsifying transforms with convergence guarantees and application to MRI," *SIAM J. Imaging Sci.*, vol. 8, no. 4, 2519–57, 2015. DOI: 10.1137/141002293.

[11] P. H. Schönemann, "A generalized solution of the orthogonal procrustes problem," *Psychometrika*, vol. 31, no. 1, pp. 1–10, 1966. DOI: 10.1007/BF02289451.

[12] K. G. G. Samuel and M. F. Tappen, "Learning optimized MAP estimates in continuously-valued MRF models," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, Miami, FL: IEEE, Jun. 2009, pp. 477–484, ISBN: 978-1-4244-3992-8. DOI: 10.1109/CVPR.2009.5206774.

[13] H. Antil, Z. Di, and R. Khatri, "Bilevel Optimization, Deep Learning and Fractional Laplacian Regularization with Applications in Tomography," *Inverse Problems*, Mar. 18, 2020, ISSN: 0266-5611, 1361-6420. DOI: 10.1088/1361-6420/ab80d7.

[14] L. Franceschi, M. Donini, P. Frasconi, and M. Pontil, "Forward and Reverse Gradient-Based Hyperparameter Optimization," in *Proceedings of the 34th ICML*, Sydney, Australia, 2017, pp. 1165–1173. [Online]. Available: http://proceedings.mlr.press/v70/franceschi17a.html.

[15] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015*, 2015. arXiv: 1412.6980.