LEARNING THE WEIGHT MATRIX FOR SPARSITY AVERAGING IN COMPRESSIVE IMAGING

Dimitris Perdios*, Adrien Besson*, Philippe Rossinelli*, and Jean-Philippe Thiran*†
*Signal Processing Laboratory (LTS5), École Polytechnique Fédérale de Lausanne (EPFL), Switzerland
†Department of Radiology, University Hospital Center (CHUV) and University of Lausanne (UNIL), Switzerland

Introduction and objectives

Context

- Recover high-quality image $\boldsymbol{x} \in \mathbb{R}^N$ from undersampled measurements $\boldsymbol{y} \in \mathbb{R}^M$, measured with linear operator $\mathrm{A} \in \mathbb{R}^{M\times N}$

$$\boldsymbol{y} = \mathrm{A}\boldsymbol{x} + \boldsymbol{n}$$

- Assume $\boldsymbol{x}$ is sparse in a dictionary $\Psi \in \mathbb{R}^{N\times L}$ $(L > N)$ ➔ CS

Goal: High-quality recovery with few iterations

Strategy: Sparsity Averaging for Reweighted Analysis (SARA) [1]

$$\min_{\boldsymbol{x}\in\mathbb{R}^N} ||\mathrm{W}\Psi^\dagger \boldsymbol{x}||_1 + \frac{1}{2}||\mathrm{A}\boldsymbol{x} - \boldsymbol{y}||_2^2 \quad (1)$$

- $\Psi \in \mathbb{R}^{N\times L}$, $L = qN$ is a concatenation of q bases Ψ_q and $\mathrm{W} \in \mathbb{R}^{L\times L}$
- $\mathrm{W} \in \mathbb{R}^{L\times L}$ is block-diagonal made of q blocks $(N \times N)$ with positive entries
- **Drawback:** Reweighted-ℓ_1 algorithms take **"forever"** due to multiple updates of the weights
- **Proposition:** Learn weight matrix W using DNN so that no update required

Proposed approach

- Unfolding strategy [2]: each iteration of FISTA [3] mapped to a DNN
- Learned Extended FISTA, coined LEFISTA

Require: $\mathrm{G} = \frac{1}{L}\mathrm{A}^T, \mathrm{S} = \left(\mathrm{I} - \frac{1}{L}\mathrm{A}^T\mathrm{A}\right), \mathrm{W}, \Psi, \boldsymbol{y}, L \geq \lambda_{max}\left(\mathrm{A}^T\mathrm{A}\right), T, q$
initialization: $i = 1, t_0 = 1, \boldsymbol{x}_{-1} = \boldsymbol{x}_0 = \boldsymbol{0}$
repeat
$t_i \leftarrow \frac{1+\sqrt{1+4t_{i-1}^2}}{2}, \quad \alpha_i \leftarrow \frac{t_{i-1}-1}{t_i}, \quad \beta_i \leftarrow 1 + \alpha_i$
$\boldsymbol{z}_i \leftarrow \beta_i \mathrm{S}\boldsymbol{x}_{i-1} - \alpha_i \mathrm{S}\boldsymbol{x}_{i-2} + \mathrm{G}\boldsymbol{y}$
for $k = 1$ to q **do**
$\boldsymbol{x}_i \leftarrow \boldsymbol{x}_i + \frac{\Psi_k}{\sqrt{q}}\mathrm{soft}\left(\frac{\Psi_k^\dagger}{\sqrt{q}}\boldsymbol{z}_i; \frac{1}{L}\mathrm{W}_k\right)$
end for
$i \leftarrow i + 1$
until $i = T$
return $(\boldsymbol{x}_i)_{i=1}^T, (\boldsymbol{z}_i)_{i=1}^T, (\alpha_i)_{i=1}^T, (\beta_i)_{i=1}^T$

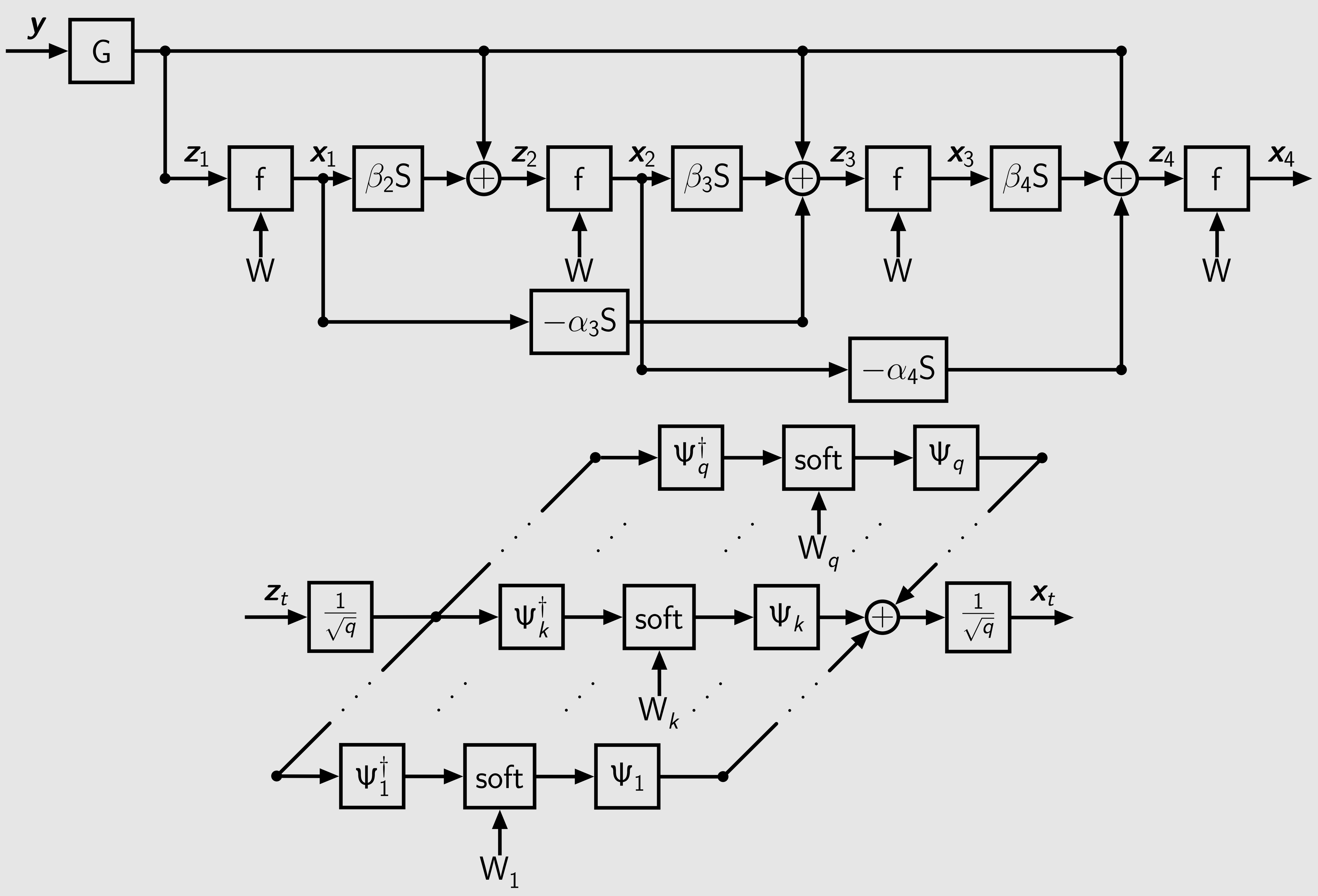

Figure LEFISTA network architecture: (top) 4-first layers (bottom) non-linearity f

Learning and image reconstruction processes

Learning the weight matrix

- Training set made of P pairs $\left(\boldsymbol{y}_p, \boldsymbol{x}_p^\star\right)_{p=1}^P$, $\boldsymbol{x}_p^\star$ the ref., $\boldsymbol{y}_p$ the measurements
- Objective: find W which minimizes the ℓ_2-loss function ➔ BPTT

Image reconstruction

- GPU RAM limitations ➔ patches ➔ block-compressed sensing (BCS)
- $\mathrm{A} \in \mathbb{R}^{M_B\times N_B^2}$ on patches of size $N_B \times N_B$ pixels (64×64)
- Image split into B non-overlapping patches, compressed with A
- Apply LEFISTA forward to reconstruct image with W learned in training phase

Network training

- TensorFlow implementation: `https://github.com/dperdios/lefista`
- Trained on NVIDIA Titan X GPU card
- Different layer number T tested (30, 40, 50), best is 50 (LEFISTA-LN50)
- Mini-batch learning: 43 560 patches from 1200 images ILSVRC 2014 ImageNet
- Optimizer: Adam, learning rate: 10^{-5}, batch size: 32, epoch number: 20

Experimental settings

- Ψ: concatenation of 8 wavelet bases (db 1 to db 8), decomposition level: 2
- A: Gaussian random matrix, with the measurement rate M_B/N_B^2
- Quality evaluated in terms of PSNR and SSIM
- Comparison to a tiled version of SARA and BCS algorithms

Performance evaluation

Figure Test images reconstructed for a measurement rate $M_B/N_B^2 = 0.3$ (first row) with tiled SARA and (second row) with LEFISTA (50 layers)

Table Comparison of LEFISTA (50 layers) against tiled SARA and BCS algorithms

Algorithm	Measurement rate					
	PSNR [dB]			SSIM [–]		
	0.1	0.3	0.5	0.1	0.3	0.5
Barbara						
Tiled SARA	16.15	24.90	**29.70**	0.38	0.79	**0.91**
BCS-SPL-DWT	21.87	24.31	27.06	0.40	0.61	0.75
BCS-SPL-DDWT	22.11	24.74	27.84	0.40	0.62	0.76
MS-BCS-SPL-DWT	22.17	24.86	27.99	0.41	0.63	0.77
LEFISTA-LN50	**22.77**	**25.73**	29.29	**0.61**	**0.79**	0.90
Goldhill						
Tiled SARA	18.56	29.76	33.19	0.45	0.81	0.90
BCS-SPL-DWT	24.57	30.40	33.06	0.42	0.68	0.80
BCS-SPL-DDWT	25.18	30.45	33.11	0.42	0.68	0.80
MS-BCS-SPL-DWT	26.74	30.57	33.19	0.44	0.68	0.80
LEFISTA-LN50	**26.77**	**30.93**	**34.17**	**0.66**	**0.83**	**0.91**
Peppers						
Tiled SARA	18.71	32.81	35.35	0.44	0.84	0.90
BCS-SPL-DWT	27.73	33.38	35.92	0.45	0.65	0.76
BCS-SPL-DDWT	28.09	33.52	36.26	0.45	0.65	0.77
MS-BCS-SPL-DWT	28.22	33.63	36.33	0.45	0.65	0.77
LEFISTA-LN50	**28.45**	**33.72**	**36.34**	**0.77**	**0.87**	**0.91**

Conclusion and perspectives

- FISTA with a sparsity prior in a concatenation of wavelet bases Ψ mapped to a DNN ➔ LEFISTA
- Used to learn the weight matrix W of a weighted ℓ_1-minimization problem
- Once trained, **much faster** than reweighted ℓ_1 with promising results
- Future work:
  - Learn non-linearities (e.g. prox., compression)
  - Address blocking artifacts

References

[1] R. E. Carrillo, J. D. McEwen, D. Van De Ville, J.-P. Thiran, and Y. Wiaux, "Sparsity Averaging for Compressive Imaging", *IEEE Signal Process. Lett.*, vol. 20, no. 6, pp. 591–594, 2013.
[2] K. Gregor and Y. LeCun, "Learning Fast Approximations of Sparse Coding", in *27th Int. Conf. Mach. Learn. ICML 2010*, 2010, pp. 399–406.
[3] A. Beck and M. Teboulle, "A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems", *SIAM J. Imaging Sci.*, vol. 2, no. 1, pp. 183–202, 2009.

Acknowledgments

This work was supported in part by the UltrasoundToGo RTD project (no. 20NA21_145911), evaluated by the Swiss NSF and funded by Nano-Tera.ch with Swiss Confederation financing