Convergent Deep Convex and Non-Convex Regularization

Srijith Nair and Philip Schniter
The Ohio State University
Supported by NSF grant CCF-1955587

John D. and Alice Nelson Kraus Memorial
ECE Graduate Student Poster Competition - 2023

Variational signal/image recovery

Goal: Recover signal/image $\boldsymbol{x}_0$ from noisy measurements $\boldsymbol{y} = \mathcal{F}(\boldsymbol{x}_0, \boldsymbol{w})$

- Applications: denoising, deblurring, superresolution, inpainting, computed tomography, magnetic resonance imaging (MRI), phase-retrieval, de-quantization, photon-limited image recovery, etc.

Variational formulation:

$$\widehat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} J(\boldsymbol{x}) \text{ with } J(\boldsymbol{x}) \triangleq d(\boldsymbol{x};\boldsymbol{y}) + r(\boldsymbol{x};\boldsymbol{\theta})$$

- $d(\cdot;\boldsymbol{y})$ is a data-fidelity term, usually chosen as negative log-likelihood
- $r(\cdot;\boldsymbol{\theta})$ is a regularizer that incorporates prior information
- When r is differentiable, a prototypical algorithm is FBS/PGD:
$$\boldsymbol{x}^{(k+1)} = \operatorname{prox}_{\tau^{(k)}d}\left(\boldsymbol{x}^{(k)} - \tau^{(k)}\nabla r(\boldsymbol{x}^{(k)};\boldsymbol{\theta})\right),$$
where explicit r allows the use of back-tracking line-search to adapt $\tau^{(k)}$
- Key question: How do we choose the regularizer r?

Regularizer design:

- Traditional hand-crafted regularizers use total-variation or wavelet sparsity
- We focus on data-driven regularization leveraging training data $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^n$

Data-driven regularization

Overall goals:

1. Variational optimization should be tractable (e.g., J convex)
2. Resulting $\widehat{\boldsymbol{x}}$ should be close to ground truth $\boldsymbol{x}_0$

Regularizer architecture:

- Many have been proposed, based on, e.g., one-layer networks [1], denoisers [2, 3], autoencoders [4], and deep convolutional networks [5, 6, 7]
- Possible to make $r(\cdot;\boldsymbol{\theta})$ structurally convex [8, 9], but this requires specialized architectures that can limit expressivity

Gradient-step (GS) optimization [6, 10]:

- Train $I - \nabla r$ as a denoiser $\Leftrightarrow$ train ∇r as a noise-estimator:
$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}_{\text{GS}}(\boldsymbol{\theta}) \quad \text{for} \quad \mathcal{L}_{\text{GS}}(\boldsymbol{\theta}) \triangleq \textstyle\sum_{i=1}^n \mathbb{E}\left[L(\nabla r(\boldsymbol{x}_i + \boldsymbol{n};\boldsymbol{\theta}), \boldsymbol{n})\right]$$
with $L(\widehat{\boldsymbol{x}}, \boldsymbol{x})$ a loss like $L_2(\widehat{\boldsymbol{x}}, \boldsymbol{x}) \triangleq \|\widehat{\boldsymbol{x}} - \boldsymbol{x}\|_2^2$,
- Use automatic differentiation to compute ∇r
- Minimize $\widetilde{J}(\boldsymbol{x}) = d(\boldsymbol{x};\boldsymbol{y}) + \lambda\, r(\boldsymbol{x};\boldsymbol{\theta})$ with λ tuned using validation data

Bi-level (BL) optimization [11]:

- Choose $\boldsymbol{\theta}$ to give a accurate $\widehat{\boldsymbol{x}}_i(\boldsymbol{\theta}) \triangleq \arg\min_{\boldsymbol{x}} J(\boldsymbol{x};\boldsymbol{y}_i,\boldsymbol{\theta})$, i.e.,
$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}_{\text{BL}}(\boldsymbol{\theta}) \quad \text{for} \quad \mathcal{L}_{\text{BL}}(\boldsymbol{\theta}) \triangleq \textstyle\sum_{i=1}^n L(\widehat{\boldsymbol{x}}_i(\boldsymbol{\theta}), \boldsymbol{x}_i)$$
- Can show that gradient equals
$$\nabla \mathcal{L}_{\text{BL}}(\boldsymbol{\theta}) = \textstyle\sum_{i=1}^n \nabla_{\boldsymbol{\theta}} \nabla_{\boldsymbol{x}^{\mathsf{T}}} J(\widehat{\boldsymbol{x}}_i;\boldsymbol{y}_i,\boldsymbol{\theta})\boldsymbol{\gamma}_i$$
for the $\boldsymbol{\gamma}_i$ that solves the linear system
$$\left[\nabla_{\boldsymbol{x}} \nabla_{\boldsymbol{x}^{\mathsf{T}}} J(\widehat{\boldsymbol{x}}_i;\boldsymbol{y}_i,\boldsymbol{\theta})\right]\boldsymbol{\gamma}_i = \boldsymbol{x}_i - \widehat{\boldsymbol{x}}_i$$
- Use PGD to compute $\widehat{\boldsymbol{x}}_i$, MINRES to compute $\widehat{\boldsymbol{\gamma}}_i$, then gradient step in $\boldsymbol{\theta}$

Monotonicity

- A function $g: \mathbb{R}^p \to \mathbb{R}$ is convex if and only if its gradient is monotone, i.e.,
$$\frac{[\partial g(\boldsymbol{x}_1) - \partial g(\boldsymbol{x}_2)]^{\mathsf{T}}(\boldsymbol{x}_1 - \boldsymbol{x}_2)}{\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_2^2} \geq 0 \quad \forall \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^p$$
where ∂g is the subdifferential
- Pesquet et al. [12] designed a data-driven denoiser that is the resolvent of a monotone operator, ensuring a convergent PnP algorithm [13]
- Below, we propose two ways to exploit monotonicity with an *explicit differentiable* regularizer r that has more in common with RED [2] than PnP

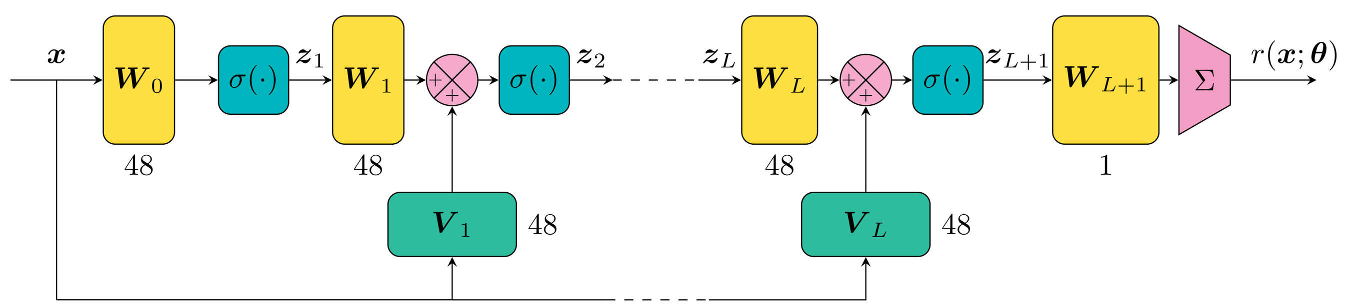
Proposed convex regularizer

- We propose to design r using an adversarial monotonicity penalty on ∇r
$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \left\{\mathcal{L}_*(\boldsymbol{\theta}) + \kappa_{\text{mon}}\mathcal{P}^{\text{mon}}_{\nabla r}(\boldsymbol{\theta}) + \kappa_{\text{lip}}\mathcal{P}^{\text{lip}}_{\nabla r}(\boldsymbol{\theta})\right\}$$
$$\mathcal{P}^{\text{mon}}_{\nabla r}(\boldsymbol{\theta}) \triangleq \textstyle\sum_{i=1}^n \max\{0, -\widehat{\mu}_r(\boldsymbol{x}_i,\boldsymbol{\theta})\}$$
$$\mathcal{P}^{\text{lip}}_{\nabla r}(\boldsymbol{\theta}) \triangleq \textstyle\sum_{i=1}^n \max\{0, \widehat{\beta}_r(\boldsymbol{x}_i,\boldsymbol{\theta}) - \beta_r\}$$
$$\widehat{\mu}_r(\boldsymbol{x}_i,\boldsymbol{\theta}) \triangleq \min_{\boldsymbol{\delta}^1,\boldsymbol{\delta}^2 \in \mathcal{B}(\epsilon)} \frac{[\nabla r(\boldsymbol{x}_i + \boldsymbol{\delta}^1;\boldsymbol{\theta}) - \nabla r(\boldsymbol{x}_i + \boldsymbol{\delta}^2;\boldsymbol{\theta})]^{\mathsf{T}}(\boldsymbol{\delta}^1 - \boldsymbol{\delta}^2)}{\|\boldsymbol{\delta}^1 - \boldsymbol{\delta}^2\|_2^2}$$
$$\widehat{\beta}_r(\boldsymbol{x}_i,\boldsymbol{\theta}) \triangleq \max_{\boldsymbol{\delta}^3,\boldsymbol{\delta}^4 \in \mathcal{B}(\epsilon)} \frac{\|\nabla r(\boldsymbol{x}_i + \boldsymbol{\delta}^3;\boldsymbol{\theta}) - \nabla r(\boldsymbol{x}_i + \boldsymbol{\delta}^4;\boldsymbol{\theta})\|_2}{\|\boldsymbol{\delta}^3 - \boldsymbol{\delta}^4\|_2}$$
where $\mathcal{L}_* \in \{\mathcal{L}_{\text{BL}}, \mathcal{L}_{\text{GS}}\}$ and $\mathcal{B}(\epsilon)$ is an ϵ-ball with suitable ϵ
- The Lipschitz penalty on ∇r helps to stabilize training and improve performance

Proposed non-convex regularizer

- We propose to design r using an adversarial monotonicity penalty on ∇J
$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \left\{\mathcal{L}_{\text{BL}}(\boldsymbol{\theta}) + \kappa_{\text{mon}}\mathcal{P}^{\text{mon}}_{\nabla J}(\boldsymbol{\theta}) + \kappa_{\text{lip}}\mathcal{P}^{\text{lip}}_{\nabla r}(\boldsymbol{\theta})\right\}$$
$$\mathcal{P}^{\text{mon}}_{\nabla J}(\boldsymbol{\theta}) \triangleq \textstyle\sum_{i=1}^n \max\{0, -\widehat{\mu}_J(\boldsymbol{x}_i,\boldsymbol{\theta})\}$$
$$\widehat{\mu}_J(\boldsymbol{x}_i,\boldsymbol{\theta}) \triangleq \min_{\boldsymbol{\delta}^1,\boldsymbol{\delta}^2 \in \mathcal{B}(\epsilon)} \frac{[\nabla J(\boldsymbol{x}_i + \boldsymbol{\delta}^1;\boldsymbol{\theta}) - \nabla J(\boldsymbol{x}_i + \boldsymbol{\delta}^2;\boldsymbol{\theta})]^{\mathsf{T}}(\boldsymbol{\delta}^1 - \boldsymbol{\delta}^2)}{\|\boldsymbol{\delta}^1 - \boldsymbol{\delta}^2\|_2^2}$$
- This encourages J to be convex without forcing r to be convex
- Reminiscent of the "convex non-convex" approach for handcrafted sparse regularizers [14]

Example regularizer architecture



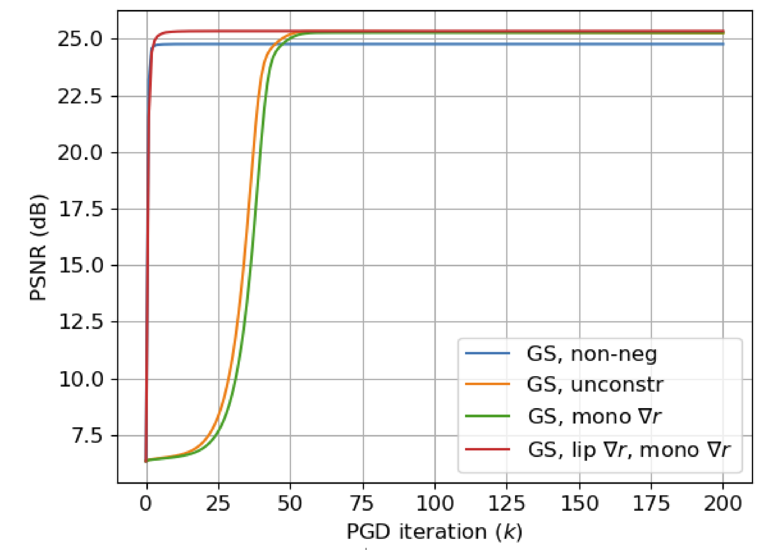
- Structurally convex under non-negativity constraint $[\boldsymbol{W}_l]_{jk} \geq 0$ [8]
  - Facilitates a direct comparison to our monotone approach
- Like Cohen et al. [6], we use it without constraints
- We use SiLU activations [15] (for differentiability) and $L = 7$ layers

Gaussian deblurring experiments (preliminary results)

- 25x25 Gaussian blur kernel with stdv 1.6
- Gaussian noise with stdv 10/255
- BSD400 dataset: 180x180 images, 350 training, 50 validation, 68 test

Gradient-step (GS) training

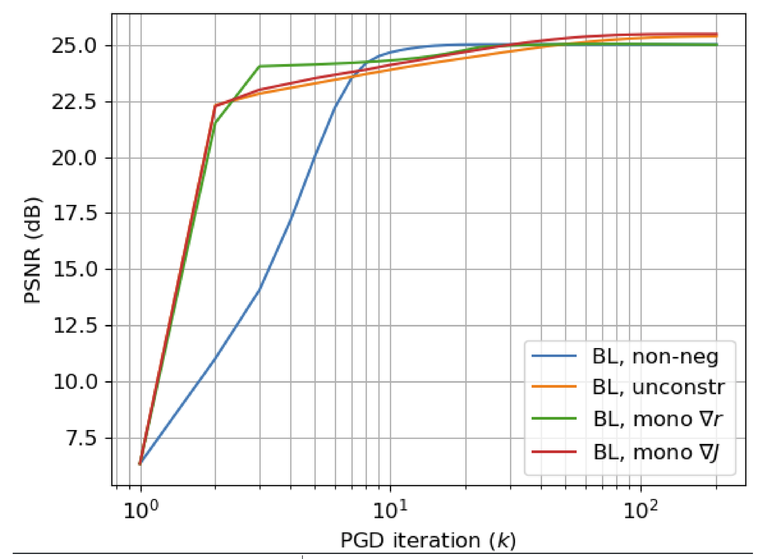
- monotone outperformed structurally convex (more expressive)
- monotone outperformed unconstrained (trapped in local minima?)
- monotone + Lipschitz performed best



Network	PSNR(dB)	SSIM
non-neg	24.77	0.681
unconstrained	25.25	0.698
monotone ∇r	25.28	0.699
mono, 20-Lip ∇r	25.34	0.701

Bi-level (BL) training

- outperforms GS
- monotone-∇r tied structurally convex
- monotone-∇J outperformed unconstrained



Network	PSNR(dB)	SSIM
non-neg	25.01	0.683
unconstrained	25.39	0.723
monotone ∇r	25.01	0.692
monotone ∇J	25.48	0.726

Future Work

- Enforce monotone constraint in the vicinity of ground truth data to increase expressivity.
- Study our methods' performance on other regularizer architectures like [9].

References

[1] S. Roth and M. J. Black, "Field of experts," *Int. J. Comput. Vision*, vol. 82, pp. 205–229, Apr. 2009.
[2] Y. Romano, M. Elad, and P. Milanfar, "The little engine that could: Regularization by denoising (RED)," *SIAM J. Imag. Sci.*, vol. 10, no. 4, pp. 1804–1844, 2017.
[3] T. Salimans and J. Ho, "Should EBMs model the energy or the score?," in *ICLR Workshop on Energy-based Models*, 2021.
[4] H. Li, J. Schwab, S. Antholzer, and M. Haltmeier, "NETT: Solving inverse problems with deep neural networks," *Inverse Problems*, vol. 36, no. 6, p. 065005, 2020.
[5] E. Kobler, A. Effland, K. Kunisch, and T. Pock, "Total deep variation for linear inverse problems," in *Proc. IEEE Conf. Comp. Vision Pattern Recog.*, pp. 7549–7558, 2020.
[6] R. Cohen, Y. Blau, D. Freedman, and E. Rivlin, "It has potential: Gradient-driven denoisers for convergent solutions to inverse problems," in *Proc. Neural Inf. Process. Syst. Conf.*, vol. 34, pp. 18152–18164, 2021.
[7] M. Zach, F. Knoll, and T. Pock, "Stable deep MRI reconstruction using generative priors," *arXiv:2210.13834*, 2022.
[8] S. Mukherjee, S. Dittmer, Z. Shumaylov, S. Lunz, O. Öktem, and C.-B. Schönlieb, "Learned convex regularizers for inverse problems," *arXiv:2008.02839*, 2020.
[9] A. Goujon, S. Neumayer, P. Bohra, S. Ducotterd, and M. Unser, "A neural-network-based convex regularizer for image reconstruction," *arXiv:2211.12461*, p. 4, 2022.
[10] S. Hurault, A. Leclaire, and N. Papadakis, "Proximal denoiser for convergent plug-and-play optimization with nonconvex regularization," in *Proc. Int. Conf. Mach. Learn.*, 2022.
[11] K. G. Samuel and M. F. Tappen, "Learning optimized MAP estimates in continuously-valued MRF models," in *Proc. IEEE Conf. Comp. Vision Pattern Recog.*, pp. 477–484, 2009.
[12] J.-C. Pesquet, A. Repetti, M. Terris, and Y. Wiaux, "Learning maximally monotone operators for image recovery," *SIAM J. Imag. Sci.*, vol. 14, no. 3, pp. 1206–1237, 2021.
[13] S. V. Venkatakrishnan, C. A. Bouman, and B. Wohlberg, "Plug-and-play priors for model based reconstruction," in *Proc. IEEE Global Conf. Signal Inf. Process.*, pp. 945–948, 2013.
[14] I. Selesnick, A. Lanza, S. Morigi, and F. Sgallari, "Non-convex total variation regularization for convex denoising of signals," *J. Math. Imaging Vis.*, vol. 62, no. 6, pp. 825–841, 2020.
[15] S. Elfwing, E. Uchibe, and K. Doya, "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning," *Neural Networks*, vol. 107, 2018.