Chapter 9

Misc topics

Contents (class version)

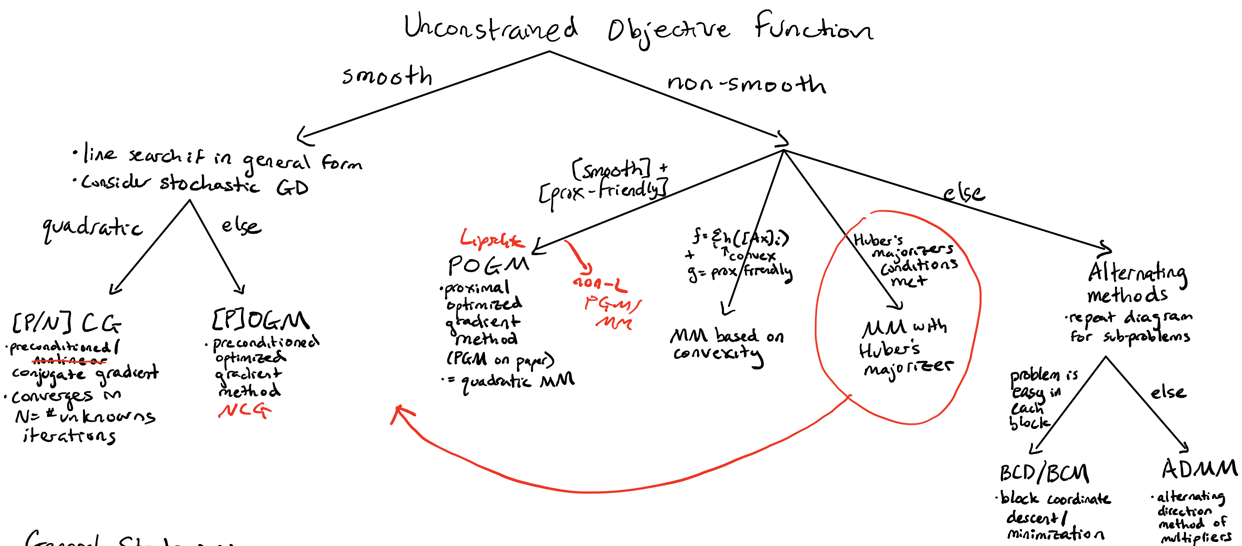
9.0 Review

Final course notes on gdrive.

Two principal kinds of questions.

- **Procedural**. Given $\Psi(\boldsymbol{x})$, given algorithm (by name), then implement/analyze it
  Interesting Exam2 sample questions (by pdf file page)
  - p1: composite with $\|\boldsymbol{x}\|_1$ and box constraint
  - p20: LS + hinge regularizer
  - p24: sigmoid as smooth 0-1 loss
  - p43: prox for leaky ReLU
  - p44: hinge + 0-norm
- **Conceptual**. Given application, then determine $\Psi(\boldsymbol{x})$, choose algorithm, and implement/analyze it
  - p2: $\|\boldsymbol{x}\|_0$ constraint
  - p6: L+S
  - p45: compressed sensing with transform/analysis sparsity

Caroline's review diagram



General Strategies:

- Rewrite as a joint cost function $\tilde{x} = \begin{bmatrix} x \\ z \end{bmatrix}$
- Write constraints using the characteristic function
- Rewrite to make convex, e.g., $\|x\|_1 = u + v$
  or linear: $\|x\|_\infty < t \Rightarrow -t\mathbf{1} \le x \le t\mathbf{1}$

- Things never appearing here:
  - PSD / PGD → use [P] OGM?
  - FGM → use OGM
  - FPGM → use POGM

Binary classifier design: Review

General cost function for binary classifier design for $\mathrm{sign}(\boldsymbol{v}'\boldsymbol{x})$ with $\boldsymbol{x} \in \mathbb{R}^N$, $\boldsymbol{A} \in \mathbb{R}^{M \times N}$, $\beta \geq 0$:

$$\Psi(\boldsymbol{x}) = f(\boldsymbol{x}) + \beta \|\boldsymbol{x}\|_p^p, \quad f(\boldsymbol{x}) = \mathbf{1}_M' h.(\boldsymbol{A}\boldsymbol{x}), \quad p \in \{1, 2\}$$

Quadratic

$$h(t) = \frac{1}{2}(t-1)^2 \Longrightarrow f(\boldsymbol{x}) =$$

Method for $p = 2$?

A: GD B: CG C: OGM D: POGM E: ADMM **??**

Other options?

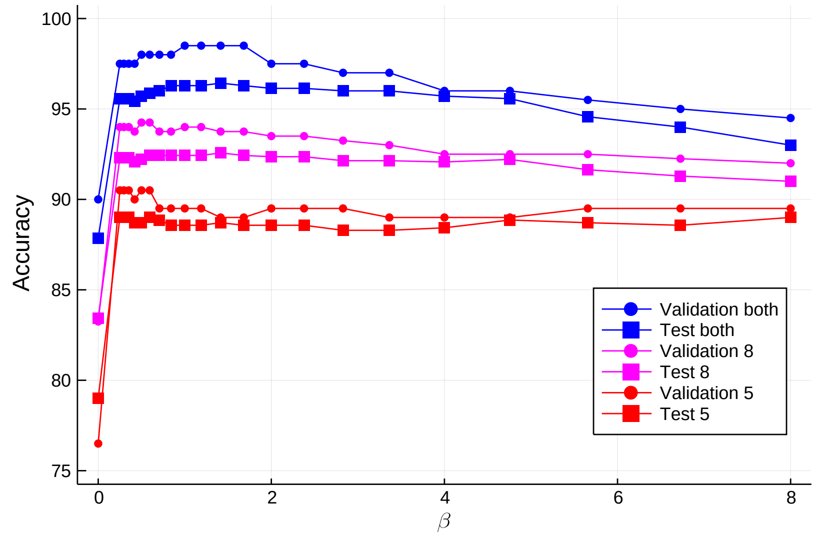
Method for $p = 1$?

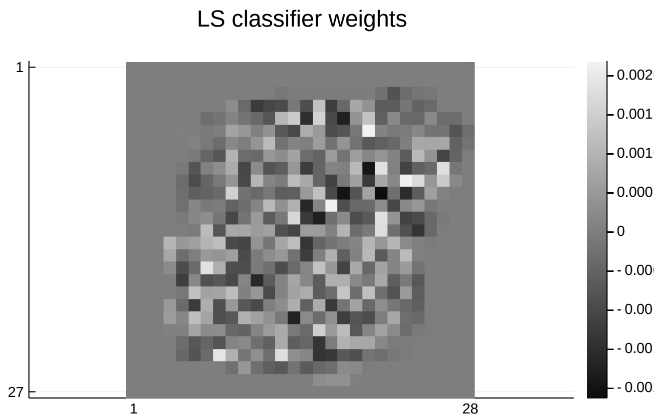
A: GD B: CG C: OGM D: POGM E: ADMM **??**

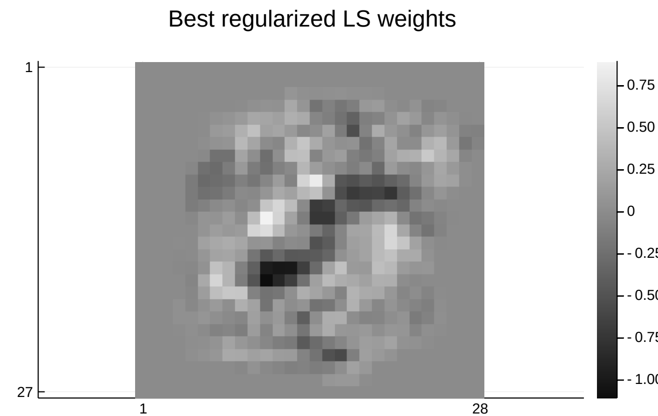
Other options?

Example

Same 5/8 handwritten digit data as in HW




LS classifier weights
1
27
1
28
0.002
0.001
0.001
0.000
0
- 0.00
- 0.00
- 0.00
- 0.00


Best regularized LS weights
1
27
1
28
0.75
0.50
0.25
0
- 0.25
- 0.50
- 0.75
- 1.00

Logistic / Huber hinge

Method for $p = 2$?

A: GD B: CG C: OGM D: POGM E: ADMM $\boxed{\textbf{??}}$

Other options?

Method for $p = 1$?

A: GD B: CG C: OGM D: POGM E: ADMM $\boxed{\textbf{??}}$

Other options?

Hinge

Method for $p = 2$?
A: GD B: CG C: OGM D: POGM E: ADMM $\boxed{??}$

Other options?

Method for $p = 1$?
A: GD B: CG C: OGM D: POGM E: ADMM $\boxed{??}$

Other options?

Sigmoid (smooth version of 0-1) loss

Method for $p = 2$?

A: GD B: CG C: OGM D: POGM E: ADMM $\boxed{\textbf{??}}$

Other options?

Method for $p = 1$?

A: GD B: CG C: OGM D: POGM E: ADMM $\boxed{\textbf{??}}$

Other options?

0-1 loss

(blank)

(blank)

9.1 If time had permitted...

recommender systems

neural networks

parallel computing

semidefinite programming

non-convex regularizers that lead to convex cost functions [9]

relationships of image models and priors for restoration problems [11]

Regularization parameter selection

SURE etc.

CNN training using SURE without ground-truth images [1] [2]

Optimization on manifolds

Minimization subject to constraints like a matrix being unitary (Stiefel manifold) [5–8]

9.2 Towards CNN methods

This section reviews some iterative algorithms based on sparsity models and summarizes how those algorithms provide a foundation for "variational neural networks" when "unrolled."

Review of denoising by soft thresholding

Consider the measurement model $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{\varepsilon}$ and the signal model that assumes $\boldsymbol{T}\boldsymbol{x}$ is sparse for some unitary transform $\boldsymbol{T}$. The natural optimization problem for estimating $\boldsymbol{x}$ is

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \frac{1}{2} \|\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \beta \|\boldsymbol{T}\boldsymbol{x}\|_1 .$$

The non-iterative solution is the following **denoising** operation:

$$\hat{\boldsymbol{x}} =$$

The essential ingredients of a **convolutional neural network** (**CNN**) are present in this simple form:

Review of compressed sensing with transform sparsity

For the compressed sensing measurement model $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{\varepsilon}$, again assuming $\boldsymbol{T}\boldsymbol{x}$ is sparse for some unitary transform $\boldsymbol{T}$, the natural optimization problem for estimating $\boldsymbol{x}$ is

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \frac{1}{2} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \beta \|\boldsymbol{T}\boldsymbol{x}\|_1 .$$

In this case there is no closed-form solution and iterative algorithms are needed. The simplest algorithm is the **proximal gradient method** (**PGM**), aka **ISTA**, which is an MM update based on the majorizer

$$\tilde{\boldsymbol{x}}_k = \boldsymbol{x}_k - \frac{1}{L} \boldsymbol{A}'(\boldsymbol{A}\boldsymbol{x}_k - \boldsymbol{y})$$
$$\phi_k(\boldsymbol{x}) = \frac{L}{2} \|\boldsymbol{x} - \tilde{\boldsymbol{x}}_k\|_2^2 + \beta \|\boldsymbol{T}\boldsymbol{x}\|_1 ,$$

where $L = \|\boldsymbol{A}\|_2^2$, for which the minimization step is a **denoising** operation:

$$\boldsymbol{x}_{k+1} = \arg\min_{\boldsymbol{x}} \phi_k(\boldsymbol{x}) =$$

The "unrolled loop" block diagram for this algorithm is the basis for **learned ISTA** (**LISTA**) [3]:

$$\boldsymbol{y} \to \boldsymbol{x}_0 \to \boxed{\text{data}} \to \tilde{\boldsymbol{x}}_0 \to \boxed{\text{denoise}} \to \boldsymbol{x}_1 \to \boxed{\text{data}} \to \tilde{\boldsymbol{x}}_1 \to \boxed{\text{denoise}} \to \boldsymbol{x}_2 \to \boxed{\text{data}} \to \tilde{\boldsymbol{x}}_2 \cdots$$

Denoise options:

Can learn: Early SURE-LET work: [4].

Review of patch transform sparsity

The natural cost function for a patch transform sparse model is

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \frac{1}{2} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \beta R(\boldsymbol{x}), \quad R(\boldsymbol{x}) = \min_{\boldsymbol{Z}} \sum_{p=1}^{P} \frac{1}{2} \|\boldsymbol{T}\boldsymbol{P}_p\boldsymbol{x} - \boldsymbol{z}_p\|_2^2 + \alpha \|\boldsymbol{z}_p\|_1 .$$

For a BCD approach, the $\boldsymbol{Z}$ update is simple:

$$\boldsymbol{z}_p^{(t+1)} =$$

To better understand the $\boldsymbol{x}$ update, it is helpful to expand the regularizer:

$$\sum_{p=1}^{P} \frac{1}{2} \left\|\boldsymbol{T}\boldsymbol{P}_p\boldsymbol{x} - \boldsymbol{z}_p^{(t+1)}\right\|_2^2 =$$

$$= \frac{1}{2}\boldsymbol{x}'\boldsymbol{H}\boldsymbol{x} - \boldsymbol{x}'\tilde{\boldsymbol{x}}_t + c_1,$$

If $\boldsymbol{T}$ has orthonormal columns (*e.g.*, is unitary), and if the patches have d pixels and are chosen with stride=1 and periodic boundary conditions, then

$$\boldsymbol{H} =$$

Completing the square for the regularizer term yields

$$\ldots = \frac{d}{2}\boldsymbol{x}'\boldsymbol{x} - \boldsymbol{x}'\tilde{\boldsymbol{x}}_t + c_1 = \qquad\qquad \bar{\boldsymbol{x}}_t \triangleq$$

Thus the $\boldsymbol{x}$ update for the **BCD** algorithm is

$$\boldsymbol{x}_{t+1} = \arg\min_{\boldsymbol{x}} \frac{1}{2} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \beta \frac{d}{2} \|\boldsymbol{x} - \bar{\boldsymbol{x}}_t\|_2^2 .$$

Here, $\bar{\boldsymbol{x}}_t$ acts like a prior for the update. For some cases there is a closed-form solution (like single-coil Cartesian MRI). Otherwise, one or more iterations are needed.

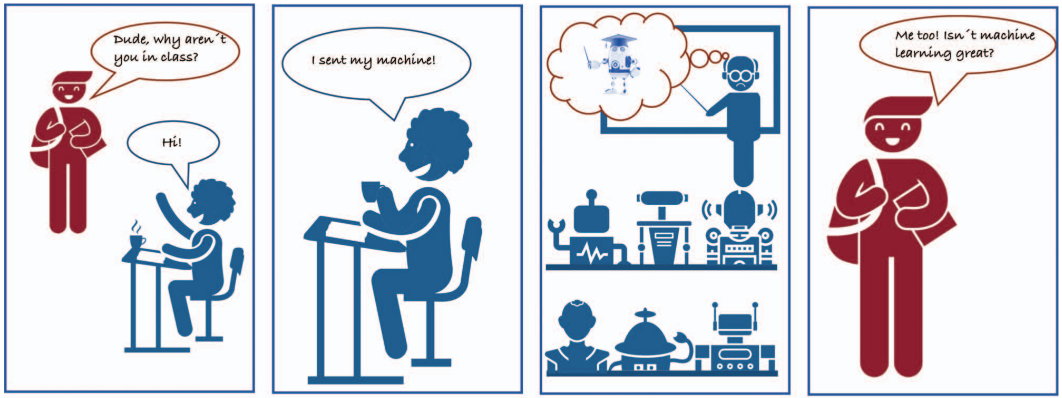
Unrolling the BCD loop in block diagram form:

$$\boldsymbol{y} \to \boldsymbol{x}_0 \to \boxed{\text{denoise}} \to \bar{\boldsymbol{x}}_0 \to \boxed{\text{data}} \to \boldsymbol{x}_1 \to \boxed{\text{denoise}} \to \bar{\boldsymbol{x}}_1 \to \boxed{\text{data}} \to \boldsymbol{x}_2 \to \boxed{\text{denoise}} \to \bar{\boldsymbol{x}}_2 \cdots$$

Denoise options:

Machine-Learning Class

by Robert W. Heath, Jr. and Nuria González-Prelcic



Note: This cartoon was created entirely by real humans, no machine learning was involved with conception or realization to practice.

Digital Object Identifier 10.1109/MSP.2019.2908298
Date of publication: 26 April 2019

SP

Bibliography

[1] C. A. Metzler, A. Mousavi, R. Heckel, and R. G. Baraniuk. *Unsupervised learning with Stein's unbiased risk estimator*. 2018 (cit. on p. 9.13).

[2] S. Soltanayev and S. Y. Chun. "Training deep learning based denoisers without ground truth data". In: *Neural Info. Proc. Sys.* 2018, 3257–67 (cit. on p. 9.13).

[3] K. Gregor and Y. LeCun. "Learning fast approximations of sparse coding". In: *Proc. Intl. Conf. Mach. Learn*. 2010 (cit. on p. 9.15).

[4] T. Blu and F. Luisier. "The SURE-LET approach to image denoising". In: *IEEE Trans. Im. Proc.* 16.11 (Nov. 2007), 2778–86 (cit. on p. 9.15).

[5] A. Edelman, Tomás A Arias, and S. T. Smith. "The geometry of algorithms with orthogonality constraints". In: *SIAM J. Matrix. Anal. Appl.* 20.2 (1998), 303–53 (cit. on p. 9.13).

[6] T. E. Abrudan, J. Eriksson, and V. Koivunen. "Steepest descent algorithms for optimization under unitary matrix constraint". In: *IEEE Trans. Sig. Proc.* 56.3 (Mar. 2008), 1134–47 (cit. on p. 9.13).

[7] B. Gao, X. Liu, X. Chen, and Y. Yuan. "A new first-order algorithmic framework for optimization problems with orthogonality constraints". In: *SIAM J. Optim.* 28.1 (Jan. 2018), 302–332 (cit. on p. 9.13).

[8] S. Chen, S. Ma, A. M-C. So, and T. Zhang. "Proximal gradient method for nonsmooth optimization over the Stiefel manifold". In: *SIAM J. Optim.* 30.1 (Jan. 2020), 210–39 (cit. on p. 9.13).

[9] A. Lanza, S. Morigi, I. W. Selesnick, and F. Sgallari. "Sparsity-inducing nonconvex nonseparable regularization for convex image processing". In: *SIAM J. Imaging Sci.* 12.2 (Jan. 2019), 1099–34 (cit. on p. 9.13).

[10] R. T. Rockafellar. "Lagrange multipliers and optimality". In: *SIAM Review* 35.2 (June 1993), 183–238.

[11] B. Wen, Y. Li, Y. Li, and Y. Bresler. *A set-theoretic study of the relationships of image models and priors for restoration problems*. 2020 (cit. on p. 9.13).

[12] W. Zuo and Z. Lin. "A generalized accelerated proximal gradient approach for total-variation-based image restoration". In: *IEEE Trans. Im. Proc.* 20.10 (Oct. 2011), 2748–59.