Multiresolution Algorithms for Faster Optimization in Machine Learning

Panos Parpas

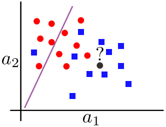
Computational Optimization Group
Department of Computing
Imperial College London

www.doc.ic.ac.uk/~pp500
p.parpas@imperial.ac.uk

Joint work with: Vahan Hovhannisyan & Stefanos Zafeiriou

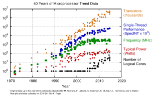
Workshop on the Mathematics of Machine Learning
Isaac Newton Institute,
Cambridge, United Kingdom
May 2018

I. The success of optimization in ML



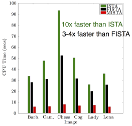
- Learning as an optimization model.
- Stochastic algorithms & large datasets.

II. Challenges for optimization algorithms in ML



- Performance & stability guarantees
- New computer architectures

III. Multiresolution optimization algorithms



- Composite convex optimization
- Theoretical & numerical results

Learning as an optimization problem

- Input: Training data
- Learn a prediction function H
- Learning $\neq$ memorising!

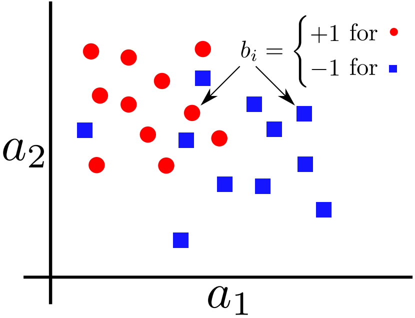
$$H(a') = \begin{cases} b_i & \text{if } a' = a_i \\ \text{random} & \text{otherwise} \end{cases}$$

- Linear prediction function

$$h(x; (a, b)) = a_i^\top x$$

- Minimise #mistakes $x \in \arg\min = |\{i | \operatorname{sign}(a_i^\top x) \neq b_i\}|$
- Even the simplest model is NP-hard!

Data $(a_{1,i}, a_{i,2}, b_i)$



Learning as an optimization problem

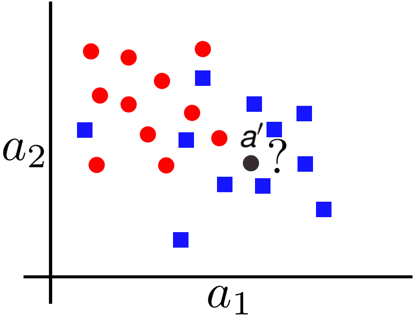
- Input: Training data
- Learn a prediction function H
- Learning $\neq$ memorising!

$$H(a') = \begin{cases} b_i & \text{if } a' = a_i \\ \text{random} & \text{otherwise} \end{cases}$$

- Linear prediction function

$$h(x; (a, b)) = a_i^\top x$$

Classify new observation



- Minimise #mistakes $x \in \arg\min = |\{i | \text{sign}(a_i^\top x) \neq b_i\}|$
- Even the simplest model is NP-hard!

Learning as an optimization problem

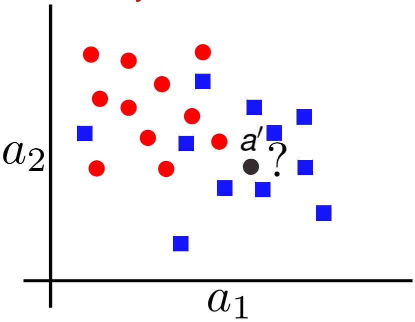
- Input: Training data
- Learn a prediction function H
- Learning $\neq$ memorising!

$$H(a') = \begin{cases} b_i & \text{if } a' = a_i \\ \text{random} & \text{otherwise} \end{cases}$$

- Linear prediction function

$$h(x; (a, b)) = a_i^\top x$$

Classify new observation



- Minimise #mistakes $x \in \arg\min = |\{i | \text{sign}(a_i^\top x) \neq b_i\}|$
- Even the simplest model is NP-hard!

Learning as an optimization problem

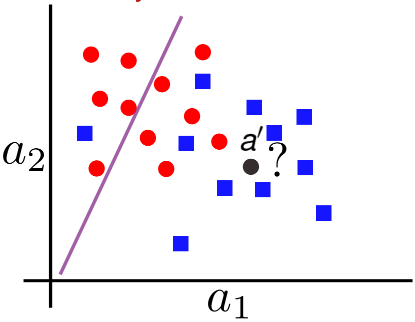
- Input: Training data
- Learn a prediction function H
- Learning $\neq$ memorising!

$$H(a') = \begin{cases} b_i & \text{if } a' = a_i \\ \text{random} & \text{otherwise} \end{cases}$$

- Linear prediction function

$$h(x; (a, b)) = a_i^\top x$$

Classify new observation



- Minimise #mistakes $x \in \arg\min = |\{i | \text{sign}(a_i^\top x) \neq b_i\}|$
- Even the simplest model is NP-hard!

Learning as a tractable optimization problem

- Counting $\to$ non-convex
- Convex model with regulariser

$$F(x) = \sum_{i=1}^{m} L(x; (a_i, b_i)) + G(x)$$

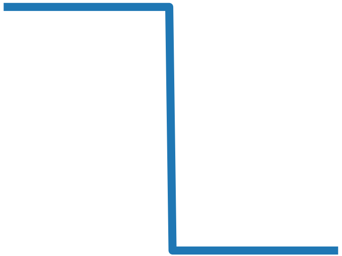
- Example:

$$L(x; (a_i, b_i)) = \ln(1 + \exp(-b_i a_i^\top x$$

$$G(x) = \lambda \|x\|_1 = \lambda \sum_{i=1}^{n} |x_i|$$

Minimise #mistakes

$$= |\{i | \mathrm{sign}(a_i^\top x) \neq b_i\}|$$



Learning as a tractable optimization problem

- Counting $\rightarrow$ non-convex
- Convex model with regulariser

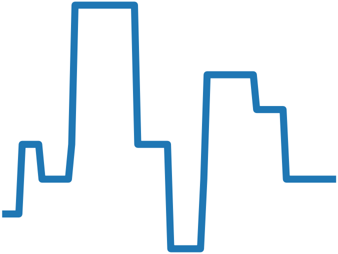
$$F(x) = \sum_{i=1}^{m} L(x; (a_i, b_i)) + G(x)$$

- Example:

$$L(x; (a_i, b_i)) = \ln(1 + \exp(-b_i a_i^\top x$$

$$G(x) = \lambda \|x\|_1 = \lambda \sum_{i=1}^{n} |x_i|$$

Difficult to optimise



Learning as a tractable optimization problem

- Counting $\rightarrow$ non-convex
- Convex model with regulariser

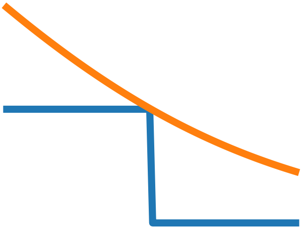
$$F(x) = \sum_{i=1}^{m} L(x; (a_i, b_i)) + G(x)$$

- Example:

$$L(x; (a_i, b_i)) = \ln(1 + \exp(-b_i a_i^\top x))$$

$$G(x) = \lambda \|x\|_1 = \lambda \sum_{i=1}^{n} |x_i|$$

Convex Approximation



Learning as a tractable optimization problem

- Counting $\to$ non-convex
- Convex model with regulariser

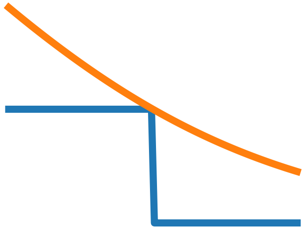
$$F(x) = \sum_{i=1}^{m} L(x; (a_i, b_i)) + G(x)$$

- Example:

$$L(x; (a_i, b_i)) = \ln(1 + \exp(-b_i a_i^\top x))$$

$$G(x) = \lambda \|x\|_1 = \lambda \sum_{i=1}^{n} |x_i|$$

Convex Approximation



Optimisation methods use local approximations

$$\boxed{x^\star \in \arg\min F(x)}$$

- Guess a solution x
- Select d to improve e.g.

$$F(x+d) < F(x)$$
$$\|x+d-x^\star\| < \|x-x^\star\|$$

- Select d to optimise a local approximation:

$$F(x+d) \approx \underbrace{F(x) + \nabla F(x)^\top d}_{\text{linear: } l_x(d)} + \underbrace{\frac{1}{2} d^\top \nabla^2 F(x) d}_{\text{quadratic: } q_x(d)}$$

- Update guess (learning)

$$\boxed{x \leftarrow x+d}$$

Why use a quadratic approximation?

- Greedy/Pragmatic

$$F(x+d) \approx \underbrace{F(x) + \nabla F(x)^\top d}_{\text{linear: } l_x(d)} + \underbrace{\frac{1}{2} d^\top \nabla^2 F(x) d}_{\text{quadratic: } q_x(d)}$$

- Smoothness: $F(x+d) \leq l_x(d) + \frac{L}{2}\|d\|^2$
- Convexity: $F(x+d) \geq l_x(d)$
- Strong convexity: $0 < \frac{1}{2}\mu\|d\|^2 \leq q_x(d)$

$$l_x(d) + \frac{1}{2}\mu\|d\|^2 \leq F(x+d) \leq l_x(d) + \frac{L}{2}\|d\|^2$$

First Order, Gradient Descent: Stochastic, Proximal, Accelerated, Block Coordinate, ...
Second Order: Newton Method, Quasi-Newton, Sketched, Subsampled ...

Why use a quadratic approximation?

- Greedy/Pragmatic

$$F(x+d) \approx \underbrace{F(x) + \nabla F(x)^\top d}_{\text{linear: } l_x(d)} + \underbrace{\frac{1}{2} d^\top \nabla^2 F(x) d}_{\text{quadratic: } q_x(d)}$$

- Smoothness: $F(x+d) \leq l_x(d) + \frac{L}{2}\|d\|^2$
- Convexity: $F(x+d) \geq l_x(d)$
- Strong convexity: $0 < \frac{1}{2}\mu\|d\|^2 \leq q_x(d)$

$$l_x(d) + \frac{1}{2}\mu\|d\|^2 \leq F(x+d) \leq l_x(d) + \frac{L}{2}\|d\|^2$$

First Order, Gradient Descent: Stochastic, Proximal, Accelerated, Block Coordinate, ...
Second Order: Newton Method, Quasi-Newton, Sketched, Subsampled ...

Why use a quadratic approximation?

- Greedy/Pragmatic

$$F(x+d) \approx \underbrace{F(x) + \nabla F(x)^\top d}_{\text{linear: } l_x(d)} + \underbrace{\frac{1}{2} d^\top \nabla^2 F(x) d}_{\text{quadratic: } q_x(d)}$$

- Smoothness: $F(x+d) \leq l_x(d) + \frac{L}{2}\|d\|^2$
- Convexity: $F(x+d) \geq l_x(d)$
- Strong convexity: $0 < \frac{1}{2}\mu\|d\|^2 \leq q_x(d)$

$$l_x(d) + \frac{1}{2}\mu\|d\|^2 \leq F(x+d) \leq l_x(d) + \frac{L}{2}\|d\|^2$$

First Order, Gradient Descent: Stochastic, Proximal, Accelerated, Block Coordinate, ...
Second Order: Newton Method, Quasi-Newton, Sketched, Subsampled ...

Success Story I - Convexity

$$F(x) = \underbrace{\sum_{i=1}^{m} L(x; (a_i, b_i))}_{\text{Fidelity}} + \underbrace{G(x)}_{\text{Sparsity}}$$

Models

- Support Vector Machines
- Basis Pursuit
- Regularised Regression
- Empirical Risk Min.
- Clustering
- Reinforcement Learning
- Bayesian Optimization
- Robust PCA

ML Applications

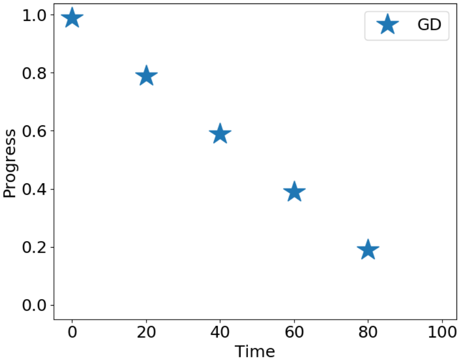
- Sparse signal reconstruction
- Image processing
- Statistical Pattern recognition
- Filtering
- Feature Selection
- Time series analysis

Success Story II - Simple Stochastic Methods

$$F(x) = \sum_{i=1}^{m} L(x; (a_i, b_i)) + G(x)$$

- Large m (observations)
- Large n (model size)
- Fast Algorithm Exist (but need all data)
- Generalization error
- Stochastic Methods (e.g. Stochastic Gradient Descent)

GD with linear rate

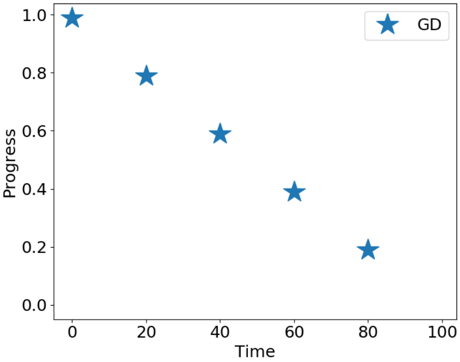


Success Story II - Simple Stochastic Methods

$$F(x) = \sum_{i=1}^{m} L(x; (a_i, b_i)) + G(x)$$

- Large m (observations)
- Large n (model size)
- Fast Algorithm Exist (but need all data)
- Generalization error
- Stochastic Methods (e.g. Stochastic Gradient Descent)

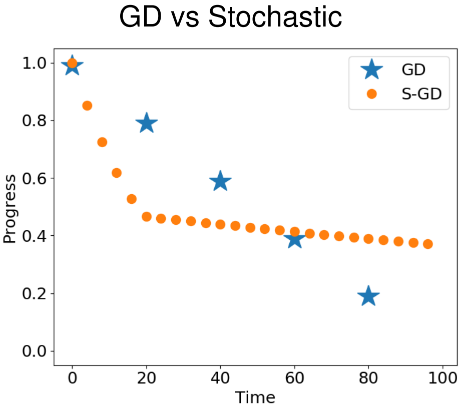
GD with linear rate



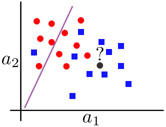
Success Story II - Simple Stochastic Methods

$$F(x) = \sum_{i=1}^{m} L(x; (a_i, b_i)) + G(x)$$

- Large m (observations)
- Large n (model size)
- Fast Algorithm Exist (but need all data)
- Generalization error
- Stochastic Methods (e.g. Stochastic Gradient Descent)

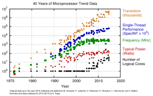


I. The success of optimization in ML



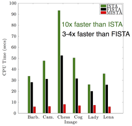
- Learning as an optimization model.
- Stochastic algorithms & large datasets.

II. Challenges for optimization algorithms in ML



- Performance & stability guarantees
- New computer architectures

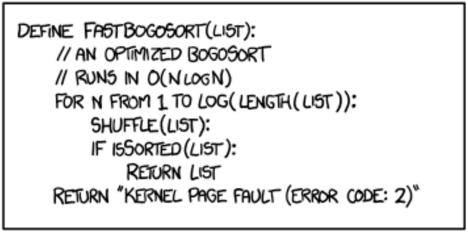
III. Multiresolution optimization algorithms



- Composite convex optimization
- Theoretical & numerical results

Challenge I - Provably Fast and Stable

- Why provably?
  - Affine invariant
  - Guaranteed performance
- Reduce development cost
  - Training
  - Tuning
- Solution accuracy matters
- Models/data keep growing
  - Physical models
  - Engineering models

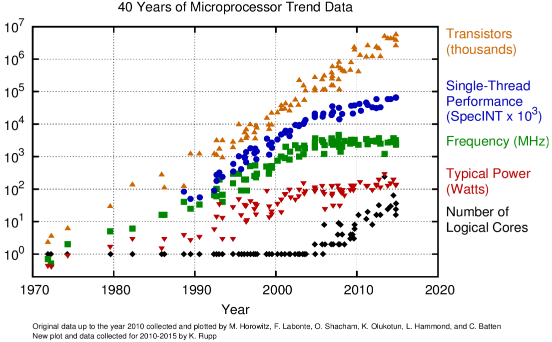


https://xkcd.com/1185/

Challenge II - Evolving Computer Architectures

- Many-core architectures
- Parallelism via:
  - Duality (e.g. ADMM,ALM)
  - Block structures (e.g. BCD, Jacobi, Domain Decomp.)

Breakdown of Dennard scaling

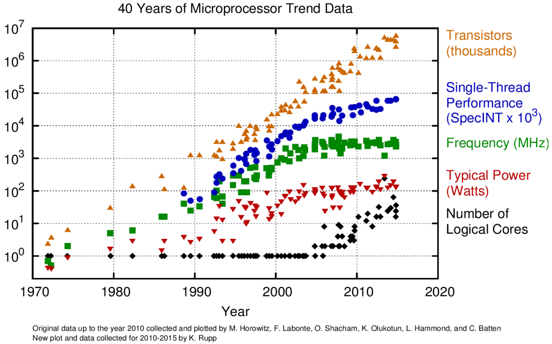


- Simple algorithms (e.g. SGD) are hard to parallelise
- Theory (asynchronous case) in its infancy
  - Pessimistic error bounds
  - Hard to tune parameters
  - Disparity between theory & practice

Challenge II - Evolving Computer Architectures

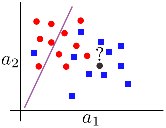
- Many-core architectures
- Parallelism via:
  - Duality (e.g. ADMM,ALM)
  - Block structures (e.g. BCD, Jacobi, Domain Decomp.)

Breakdown of Dennard scaling



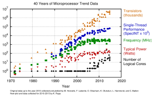
- Simple algorithms (e.g. SGD) are hard to parallelise
- Theory (asynchronous case) in its infancy
  - Pessimistic error bounds
  - Hard to tune parameters
  - Disparity between theory & practice

I. The success of optimization in ML



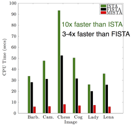
- Learning as an optimization model.
- Stochastic algorithms & large datasets.

II. Challenges for optimization algorithms in ML



- Performance & stability guarantees
- New computer architectures

III. Multiresolution optimization algorithms



- Composite convex optimization
- Theoretical & numerical results

Composite Convex Optimisation

$$\min_{x \in \Omega} \; f(x) + g(x)$$

- $f : \Omega \to \mathbb{R}$ convex & Lipschitz continuous gradient,

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|$$

- $g : \Omega \to \mathbb{R}$ convex, continuous, non-differentiable.
- g is "simple" (e.g. norm).

Composite Convex Optimisation

$$\min_{x \in \Omega_h} \; f_h(x) + g_h(x)$$

- $f_h : \Omega_h \to \mathbb{R}$ convex & Lipschitz continuous gradient,

$$\|\nabla f_h(x) - \nabla f_h(y)\| \leq L_h \|x - y\|$$

- $g_h : \Omega_h \to \mathbb{R}$ convex, continuous, non-differentiable.
- g_h is “simple” (e.g. norm).
- **Multiresolution notation**:
  - h fine (full) model
  - H coarse (approximate) model

Information transfer between levels

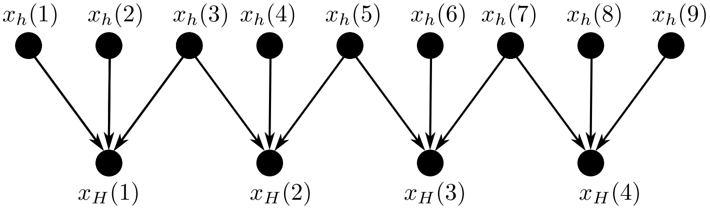
- Coarse model design vector: $x_H \in \mathbb{R}^H$
- Fine model design vector: $x_h \in \mathbb{R}^h$ and $h > H$

Two standard techniques

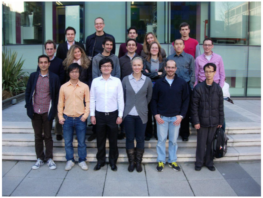
Restriction Operator: $R \in \mathbb{R}^{H \times h}$
Prolongation Operator: $P \in \mathbb{R}^{h \times H}$
Main Assumption:

$$P = cP^\top, \quad c > 0$$

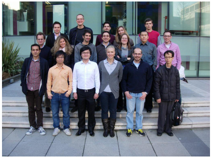
I Geometric



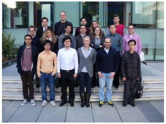
II Algebraic



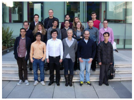
$$1280 \times 1024 = 1310720$$



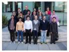
$$1024 \times 768 = 786432$$



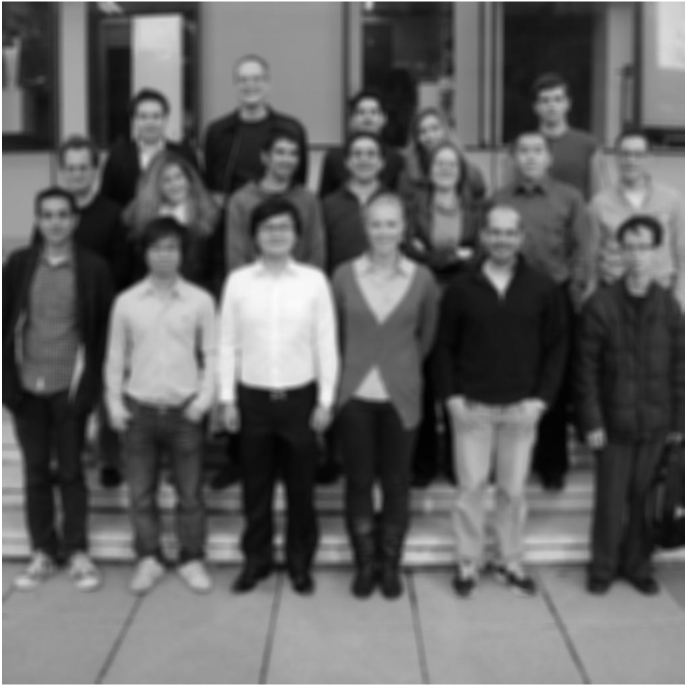
$$800 \times 600 = 480000$$



$$640 \times 480 = 307200$$



$$320 \times 240 = 76800$$



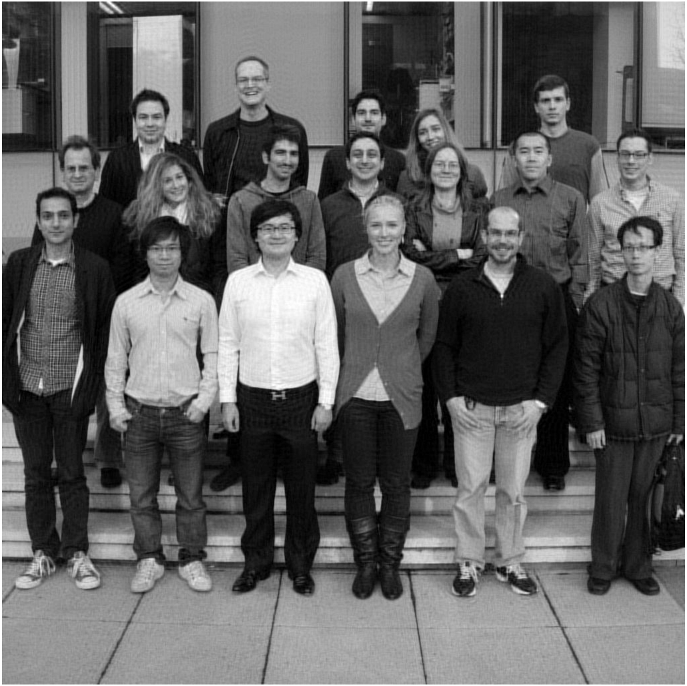
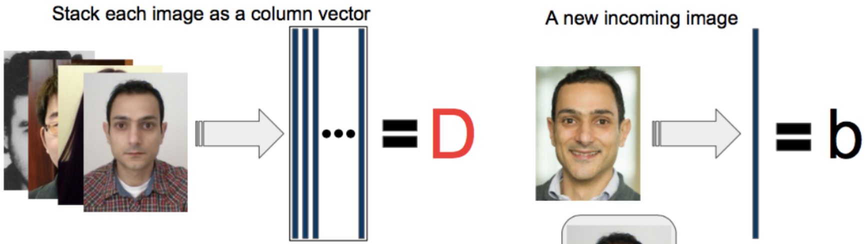


Image Restoration – Problem Formulation

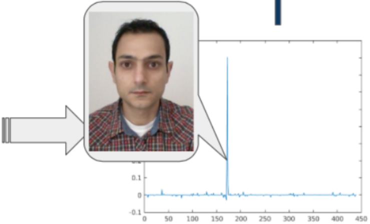
$$\min_{x_h} \|A_h x_h - b_h\|_2^2 + \mu_h \|W(x_h)\|_1$$

- b_h input image
- A_h blurring operator
- $W(\cdot)$ wavelet transform
- $x \in \mathbb{R}^h$ restored image, $h = 1024 \times 1024$



$$\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{D}\mathbf{x} - \mathbf{b}\|_2^2 + \lambda\|\mathbf{x}\|_1$$

LASSO



Algorithms – State of the art

$$\min_x \; f_h(x) + g_h(x)$$

First Order Algorithms

- **Iterative Shrinkage Thresholding Algorithm (ISTA, Proximal Point Algorithm)**[Rockafellar, 1976], [Beck and Teboulle, 2009]
- Accelerated Gradient Methods [Nesterov, 2013]
- Fast Iterative Shrinkage Thresholding Algorithm (FISTA) [Beck and Teboulle, 2009]
- Block Coordinate Descent [Nesterov, 2012]
- Incremental gradient/subgradient [Bertsekas, 2011]
- Mirror Descent [Ben-Tal et al., 2001]
- Smoothing Algorithms [Nesterov, 2005]
- Bundle Methods [Kiwiel, 1990]
- Dual Proximal Augmented Lagrangian Method [Yang and Zhang, 2011]
- Homotopy Methods [Donoho and Tsaig, 2008]

$$\min_{x \in \Omega_h} F_h(x) \triangleq f_h(x) + g_h(x)$$

Iterative Shrinkage Thresholding Algorithm (ISTA)

1. Iteration k: $x_{h,k}$, $f_{h,k}$, $\nabla f_{h,k}$, L_h.
2. **Quadratic Approximation:**

$$Q_L(x_{h,k}, x) = f_{h,k} + \langle \nabla f_{h,k}, x - x_{h,k} \rangle + \frac{L_h}{2} \|x - x_{h,k}\|^2 + g_h(x)$$

3. **Compute Gradient Map**: (minimize Quadratic Approximation)

$$\begin{aligned} D_{h,k} &= x_{h,k} - \mathrm{prox}_h(x_{h,k} - \frac{1}{L_h} \nabla f_{h,k}) \\ &= x_{h,k} - \arg\min_x \left\| x - \left(x_{h,k} - \frac{1}{L_h} \nabla f_{h,k} \right) \right\|^2 + g_h(x) \\ &= x_{h,k} - \arg\min_x Q_L(x_{h,k}, x) \end{aligned}$$

4. **Error Correction Step:**

$$x_{h,k+1} = x_{h,k} - s_{h,k} D_{h,k}.$$

$$\min_{x \in \Omega_h} F_h(x) \triangleq f_h(x) + g_h(x)$$

Iterative Shrinkage Thresholding Algorithm (ISTA)

1. Iteration k: $x_{h,k}$, $f_{h,k}$, $\nabla f_{h,k}$, L_h.
2. **Quadratic Approximation:**

$$Q_L(x_{h,k}, x) = f_{h,k} + \langle \nabla f_{h,k}, x - x_{h,k} \rangle + \frac{L_h}{2} \|x - x_{h,k}\|^2 + g_h(x)$$

3. **Compute Gradient Map:** (minimize Quadratic Approximation)

$$\begin{aligned} D_{h,k} &= x_{h,k} - \text{prox}_h(x_{h,k} - \frac{1}{L_h} \nabla f_{h,k}) \\ &= x_{h,k} - \arg\min_x \left\| x - \left(x_{h,k} - \frac{1}{L_h} \nabla f_{h,k} \right) \right\|^2 + g_h(x) \\ &= x_{h,k} - \arg\min_x Q_L(x_{h,k}, x) \end{aligned}$$

4. **Error Correction Step:**

$$x_{h,k+1} = x_{h,k} - s_{h,k} D_{h,k}.$$

$$\min_{x \in \Omega_h} F_h(x) \triangleq f_h(x) + g_h(x)$$

Iterative Shrinkage Thresholding Algorithm (ISTA)

1. Iteration k: $x_{h,k}$, $f_{h,k}$, $\nabla f_{h,k}$, L_h.
2. **Quadratic Approximation:**

$$Q_L(x_{h,k}, x) = f_{h,k} + \langle \nabla f_{h,k}, x - x_{h,k} \rangle + \frac{L_h}{2} \|x - x_{h,k}\|^2 + g_h(x)$$

3. **Compute Gradient Map**: (minimize Quadratic Approximation)

$$\begin{aligned} D_{h,k} &= x_{h,k} - \text{prox}_h(x_{h,k} - \frac{1}{L_h} \nabla f_{h,k}) \\ &= x_{h,k} - \arg\min_x \left\| x - \left(x_{h,k} - \frac{1}{L_h} \nabla f_{h,k} \right) \right\|^2 + g_h(x) \\ &= x_{h,k} - \arg\min_x Q_L(x_{h,k}, x) \end{aligned}$$

4. **Error Correction Step:**

$$x_{h,k+1} = x_{h,k} - s_{h,k} D_{h,k}.$$

$$\min_{x \in \Omega_h} \; F_h(x) \triangleq f_h(x) + g_h(x)$$

Iterative Shrinkage Thresholding Algorithm (ISTA)

1. Iteration k: $x_{h,k}$, $f_{h,k}$, $\nabla f_{h,k}$, L_h.
2. **Quadratic Approximation:**

$$Q_L(x_{h,k}, x) = f_{h,k} + \langle \nabla f_{h,k}, x - x_{h,k} \rangle + \frac{L_h}{2} \|x - x_{h,k}\|^2 + g_h(x)$$

3. **Compute Gradient Map**: (minimize Quadratic Approximation)

$$\begin{aligned} D_{h,k} &= x_{h,k} - \mathrm{prox}_h(x_{h,k} - \frac{1}{L_h} \nabla f_{h,k}) \\ &= x_{h,k} - \arg\min_x \left\| x - \left(x_{h,k} - \frac{1}{L_h} \nabla f_{h,k} \right) \right\|^2 + g_h(x) \\ &= x_{h,k} - \arg\min_x Q_L(x_{h,k}, x) \end{aligned}$$

4. **Error Correction Step:**

$$x_{h,k+1} = x_{h,k} - s_{h,k} D_{h,k}.$$

$$\min_{x \in \Omega_h} F_h(x) \triangleq f_h(x) + g_h(x)$$

Iterative Shrinkage Thresholding Algorithm (ISTA)

1. Iteration k: $x_{h,k}$, $f_{h,k}$, $\nabla f_{h,k}$, L_h.
2. ~~**Quadratic** Approximation:~~ **Coarse model**
3. **Compute Gradient Map**: (minimize Quadratic Approximation)

$$\begin{aligned} D_{h,k} &= x_{h,k} - \operatorname{prox}_h(x_{h,k} - \frac{1}{L_h}\nabla f_{h,k}) \\ &= x_{h,k} - \arg\min_x \left\| x - \left(x_{h,k} - \frac{1}{L_h}\nabla f_{h,k} \right) \right\|^2 + g_h(x) \\ &= x_{h,k} - \arg\min_x Q_L(x_{h,k}, x) \end{aligned}$$

4. **Error Correction Step:**

$$x_{h,k+1} = x_{h,k} - s_{h,k} D_{h,k}.$$

$$\min_{x \in \Omega_h} F_h(x) \triangleq f_h(x) + g_h(x)$$

Iterative Shrinkage Thresholding Algorithm (ISTA)

1. Iteration k: $x_{h,k}$, $f_{h,k}$, $\nabla f_{h,k}$, L_h.
2. ~~**Quadratic** Approximation:~~ **Coarse model**
3. ~~**Compute** Gradient Map~~ **Solve (approx) coarse model**

$$
\begin{aligned}
\cancel{D_{h,k}} &= \cancel{x_{h,k} - \text{prox}_h\left(x_{h,k} - \frac{1}{L_h}\nabla f_{h,k}\right)} \\
&= \cancel{x_{h,k} - \arg\min_x \left\| x - \left(x_{h,k} - \frac{1}{L_h}\nabla f_{h,k} \right) \right\|^2 + g(x)} \\
&= \cancel{x_{h,k} - \arg\min_x Q_L(x_{h,k}, x)}
\end{aligned}
$$

4. **Error Correction Step:**

$$x_{h,k+1} = x_{h,k} - s_{h,k} D_{h,k}.$$

$$\min_{x \in \Omega_h} F_h(x) \triangleq f_h(x) + g_h(x)$$

Iterative Shrinkage Thresholding Algorithm (ISTA)

1. Iteration k: $x_{h,k}$, $f_{h,k}$, $\nabla f_{h,k}$, L_h.
2. ~~**Quadratic** Approximation:~~ **Coarse model**
3. ~~**Compute** Gradient Map~~ **Solve (approx) coarse model**

$$\begin{aligned} \cancel{D_{h,k}} &= \cancel{x_{h,k} - \text{prox}_h\left(x_{h,k} - \frac{1}{L_h} \nabla f_{h,k}\right)} \\ &= \cancel{x_{h,k} - \arg\min_x \left\| x - \left(x_{h,k} - \frac{1}{L_h} \nabla f_{h,k} \right) \right\|^2 + g(x)} \\ &= \cancel{x_{h,k} - \arg\min_x Q_L(x_{h,k}, x)} \end{aligned}$$

4. ~~**Error Correction Step:**~~ **Compute & Apply Error Correction**

$$\cancel{x_{h,k+1} = x_{h,k} - s_{h,k} D_{h,k}}.$$

Coarse Model Construction – Smooth Case

First Order Coherent Condition

$$\min \; f_h(x_h)$$

$$x_{H,0} = R x_{h,k}, \text{ then } \nabla f_{H,0} = R \nabla f_{h,k}$$

Coarse Model:

$$f_H(x_H) \triangleq \underbrace{\widehat{f}_H(x_H)}_{\text{coarse representation of } f_h} + \underbrace{\langle R\nabla f_{h,k} - \nabla \widehat{f}_{H,0}, x_H \rangle}_{\text{first order coherent}}$$

[Lewis and Nash, 2005, Gratton et al., 2008, Wen and Goldfarb, 2009]

Coarse Model Construction – Smooth Case

First Order Coherent Condition

$$\min \; f_h(x_h)$$

$$x_{H,0} = R x_{h,k}, \text{ then } \nabla f_{H,0} = R \nabla f_{h,k}$$

Coarse Model:

$$f_H(x_H) \triangleq \underbrace{\widehat{f}_H(x_H)}_{\text{coarse representation of } f_h} + \underbrace{\langle R\nabla f_{h,k} - \nabla \widehat{f}_{H,0}, x_H \rangle}_{\text{first order coherent}}$$

[Lewis and Nash, 2005, Gratton et al., 2008, Wen and Goldfarb, 2009]

Non-Smooth Case

$$\min \; f_h(x_h) + g_h(x_h)$$

Optimality Conditions – Gradient Mapping

$$\begin{aligned} D_{h,k} &= x_{h,k} - \operatorname{prox}_h(x_{h,k} - \frac{1}{L}\nabla f_{h,k}) \\ &= x_{h,k} - \arg\min_x \left\| x - \left(x_{h,k} - \frac{1}{L}\nabla f_{h,k} \right) \right\|^2 + g(x) \end{aligned}$$

$D_{h,k} = 0$ if and only if $x_{h,k}$ is stationary.

First Order Coherent Condition:

$$D_{H,0} = R D_{h,k}$$

Non-Smooth Case

$$\min \; f_h(x_h) + g_h(x_h)$$

Optimality Conditions – Gradient Mapping

$$\begin{aligned} D_{h,k} &= x_{h,k} - \mathrm{prox}_h(x_{h,k} - \frac{1}{L}\nabla f_{h,k}) \\ &= x_{h,k} - \arg\min_x \left\| x - \left(x_{h,k} - \frac{1}{L}\nabla f_{h,k} \right) \right\|^2 + g(x) \end{aligned}$$

$D_{h,k} = 0$ if and only if $x_{h,k}$ is stationary.

First Order Coherent Condition:

$$D_{H,0} = RD_{h,k}$$

Non-Smooth Case

$$\min \; f_h(x_h) + g_h(x_h)$$

Optimality Conditions – Gradient Mapping

$$\begin{aligned} D_{h,k} &= x_{h,k} - \mathrm{prox}_h(x_{h,k} - \frac{1}{L}\nabla f_{h,k}) \\ &= x_{h,k} - \arg\min_x \left\| x - \left(x_{h,k} - \frac{1}{L}\nabla f_{h,k} \right) \right\|^2 + g(x) \end{aligned}$$

$D_{h,k} = 0$ if and only if $x_{h,k}$ is stationary.

First Order Coherent Condition:

$$D_{H,0} = R D_{h,k}$$

MISTA

1.0 If condition to use coarse model is satisfied at $x_{h,k}$

1.1. Set $x_{H,0} = Rx_{h,k}$

1.2. m coarse iterations, any monotone algorithm

1.3. Compute feasible coarse correction term,

$$d_{h,k} = P(x_{H,0} - x_{H,m})$$
$$x^{+} = \text{prox}_h(x_{h,k} - \tau d_{h,k})$$

1.4. Update fine model

$$x_{h,k+1} = x_{h,k} - s_{h,k}(x_{h,k} - x_h^{+})$$

1.5. Go to **1.0**

2.0 Otherwise do a fine iteration, any monotone algorithm, go to **1.0.**

MISTA

1.0 If condition to use coarse model is satisfied at $x_{h,k}$

1.1. Set $x_{H,0} = Rx_{h,k}$

1.2. m coarse iterations, any monotone algorithm

1.3. Compute feasible coarse correction term,

$$
\begin{aligned}
d_{h,k} &= P(x_{H,0} - x_{H,m}) \\
x^{+} &= \text{prox}_h(x_{h,k} - \tau d_{h,k})
\end{aligned}
$$

1.4. Update fine model

$$x_{h,k+1} = x_{h,k} - s_{h,k}(x_{h,k} - x_h^{+})$$

1.5. Go to **1.0**

2.0 Otherwise do a fine iteration, any monotone algorithm, go to **1.0.**

MISTA

1.0 If condition to use coarse model is satisfied at $x_{h,k}$

1.1. Set $x_{H,0} = Rx_{h,k}$

1.2. m coarse iterations, any monotone algorithm

1.3. Compute feasible coarse correction term,

$$
\begin{aligned}
d_{h,k} &= P(x_{H,0} - x_{H,m}) \\
x^{+} &= \text{prox}_h(x_{h,k} - \tau d_{h,k})
\end{aligned}
$$

1.4. Update fine model

$$x_{h,k+1} = x_{h,k} - s_{h,k}(x_{h,k} - x_h^{+})$$

1.5. Go to **1.0**

2.0 Otherwise do a fine iteration, any monotone algorithm, go to **1.0.**

Related work in Multiresolution Optimization

- **Nonlinear Optimization**
  - Nash, S. G. A multigrid approach to discretized optimization problems. *Optimization Methods and Software, 2000*
  - Gratton, S., Sartenaer, A., Toint, P. L. . Recursive trust-region methods for multiscale nonlinear optimization. *SIAM Journal on Optimization, 2008*
  - W., Zaiwen, and D. Goldfarb. A line search multigrid method for large-scale nonlinear optimization. *SIAM Journal on Optimization, 2009*

Complexity Results

- First-order-method, rate: $O(L/k)$ (convex)
- Asymptotic convergence for non-convex case

Convergence Rates – Multiresolution Case

- Nonsmooth/constrained problems
- Related to multigrid but beyond PDEs.
  - Convex case[1] (Accelerated rate)

$$F(x_k) - F(x^\star) \leq \mathcal{O}(L_f/k^2)$$

  - Strongly convex case [2](Linear rate)

$$F(x_k) - F(x^\star) \leq \sigma^k (F(x_0) - F(x^\star)) \quad \sigma \in (0,1)$$

  - Non-convex [2] (Sublinear): $F(x_k) - F(x^\star) \leq \mathcal{O}(L_f/k)$

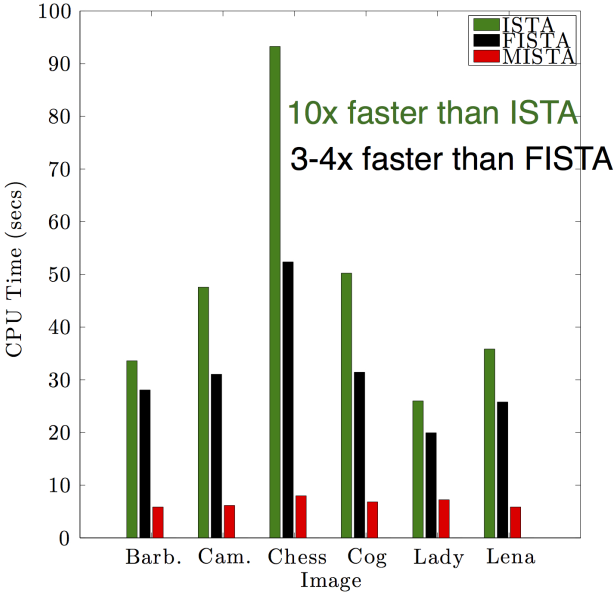
[1] V. Hovhannisyan, P.P, and S. Zafeiriou. *MAGMA: Multi-level accelerated gradient mirror descent algorithm for large-scale convex composite minimization* , SIAM Journal on Imaging Sciences, 9(4), 18291857, 2016.

[2] P.P. *A Multilevel Proximal Gradient Algorithm for Large Scale Optimization* , SIAM Journal on Scientific Computing, Vol. 39, Issue 5, Nov. 2017.

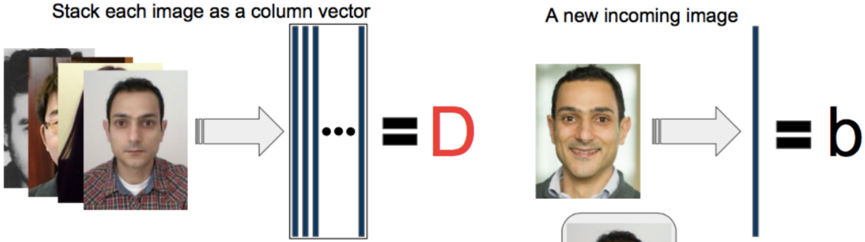
Papers&Code:

http://www.doc.ic.ac.uk/~pp500/publications.html

CPU Time Comparison – Image De-blurring

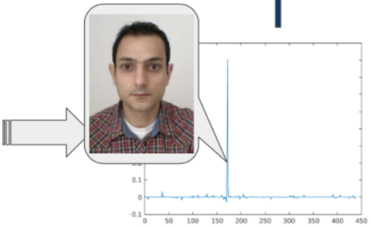


Face Recognition

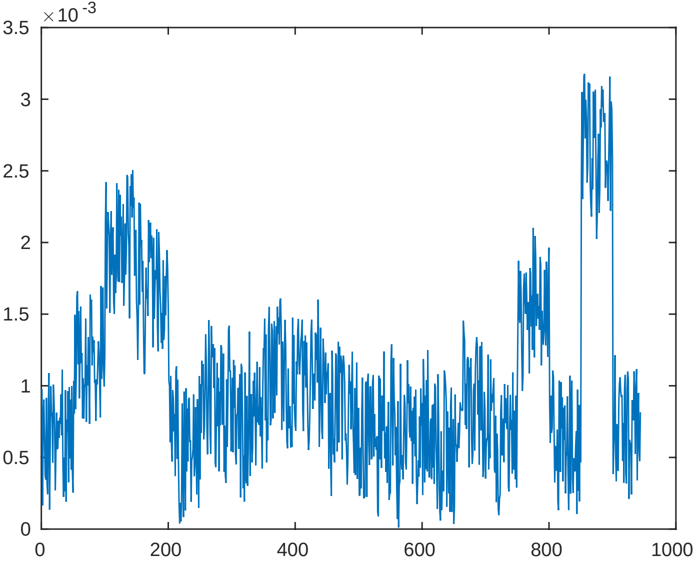


$$\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{D}\mathbf{x}-\mathbf{b}\|_2^2+\lambda\|\mathbf{x}\|_1$$

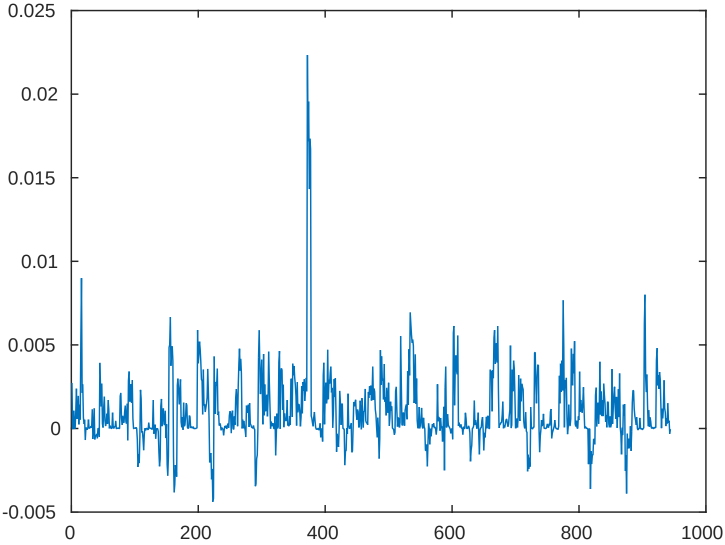
LASSO



Low Accuracy Solution (10e-3)



High Accuracy Solution (10e-7)



Current Research

(A) Structures for multiresolution methods

- Use more structure but have same convergence rate.
- Cannot be expected to work for all problems.

(B) Construction of coarse models

- Known for same problems (e.g. linear PDEs)
- Goals of optimization different than for PDEs

(C) Distributed variants

- Distributed multiresolution optimisation in its infancy

Preliminary results

- (A) Spectral structure of Hessian important
- (A+B) Low rank approximations with randomized linear algebra
- (C) Predict complicating variables (coarse), correct in parallel

Current Research

(A) Structures for multiresolution methods

- Use more structure but have same convergence rate.
- Cannot be expected to work for all problems.

(B) Construction of coarse models

- Known for same problems (e.g. linear PDEs)
- Goals of optimization different than for PDEs

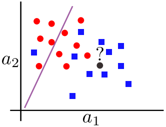
(C) Distributed variants

- Distributed multiresolution optimisation in its infancy

Preliminary results

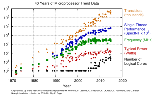
- (A) Spectral structure of Hessian important
- (A+B) Low rank approximations with randomized linear algebra
- (C) Predict complicating variables (coarse), correct in parallel

I. The success of optimization in ML



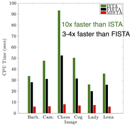
- Learning as an optimization model.
- Stochastic algorithms & large datasets.

II. Challenges for optimization algorithms in ML



- Performance & stability guarantees
- New computer architectures

III. Multiresolution optimization algorithms



- Composite convex optimization
- Theoretical & numerical results

Summary of results

- Nonsmooth/constrained problems
- Beyond PDEs & quadratic approximations
- Improved convergence rates:
  - Convex case[1] (Accelerated rate)

$$F(x_k) - F(x^\star) \leq \mathcal{O}(L_f/k^2)$$

  - Strongly convex case [2](Linear rate)

$$F(x_k) - F(x^\star) \leq \sigma^k (F(x_0) - F(x^\star)) \quad \sigma \in (0, 1)$$

  - Non-convex [2] (Sublinear): $F(x_k) - F(x^\star) \leq \mathcal{O}(L_f/k)$

[1] V. Hovhannisyan, P.P, and S. Zafeiriou. *MAGMA: Multi-level accelerated gradient mirror descent algorithm for large-scale convex composite minimization* , SIAM Journal on Imaging Sciences, 9(4), 18291857, 2016.

[2] P.P. *A Multilevel Proximal Gradient Algorithm for Large Scale Optimization* , SIAM Journal on Scientific Computing, Vol. 39, Issue 5, Nov. 2017.

Papers & Code:

http://www.doc.ic.ac.uk/~pp500/publications.html

References I

Beck, A. and Teboulle, M. (2009).
A fast iterative shrinkage-thresholding algorithm for linear inverse problems.
SIAM Journal on Imaging Sciences, 2(1):183–202.

Ben-Tal, A., Margalit, T., and Nemirovski, A. (2001).
The ordered subsets mirror descent optimization method with applications to tomography.
SIAM Journal on Optimization, 12(1):79–108.

Bertsekas, D. P. (2011).
Incremental gradient, subgradient, and proximal methods for convex optimization: A survey.
Optimization for Machine Learning, 2010:1–38.

Donoho, D. L. and Tsaig, Y. (2008).
Fast solution of-norm minimization problems when the solution may be sparse.
Information Theory, IEEE Transactions on, 54(11):4789–4812.

Gratton, S., Sartenae, A., and Toint, P. (2008).
Recursive trust-region methods for multiscale nonlinear optimization.
SIAM Journal on Optimization, 19(1):414–444.

Kiwiel, K. C. (1990).
Proximity control in bundle methods for convex nondifferentiable minimization.
Mathematical Programming, 46(1-3):105–122.

Lewis, R. and Nash, S. (2005).
Model problems for the multigrid optimization of systems governed by differential equations.
SIAM Journal on Scientific Computing, 26(6):1811–1837.

References II

Nesterov, Y. (2005).
Smooth minimization of non-smooth functions.
Mathematical Programming, 103(1):127–152.

Nesterov, Y. (2012).
Efficiency of coordinate descent methods on huge-scale optimization problems.
SIAM Journal on Optimization, 22(2):341–362.

Nesterov, Y. (2013).
Gradient methods for minimizing composite functions.
Mathematical Programming, 140(1):125–161.

Rockafellar, R. (1976).
Monotone operators and the proximal point algorithm.
SIAM Journal on Control and Optimization, 14(5):877–898.

Wen, Z. and Goldfarb, D. (2009).
A line search multigrid method for large-scale nonlinear optimization.
SIAM Journal on Optimization, 20(3):1478–1503.

Yang, J. and Zhang, Y. (2011).
Alternating direction algorithms for \ell_1-problems in compressive sensing.
SIAM journal on scientific computing, 33(1):250–278.