Stochastic Optimization in Machine Learning

Case Studies in Nonlinear Optimization

F. Bauer S. Chambon R. Halbig S. Heidekrger J. Heuke

Technische Universitt Mnchen

July 11, 2015

We're not running out of data anytime soon. It's maybe the only resource that grows exponentially.

Andreas Weigend

Outline

1. Introduction
2. SQN: A Stochastic Quasi-Newton Method
3. Proximal Method
4. Logistic Regression: An Example
5. Dictionary Learning
6. Conclusion
7. Appendix

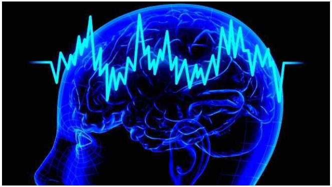
Introduction: What is Machine Learning (ML) ?

Implementation of autonomously learning software for:

- Discovery of patterns and relationships in data
- Prediction of future events

Examples:

Electroence-phalography (EEG)



Section 4

Image Denoising

Section 5

Introduction: ML and Optimization I

Training a Machine Learning model means finding optimal parameters ω:

$$\omega^* = \text{argmin}_\omega F(\omega, X, z)$$

- F: Loss function of chosen ML-model
- X: The training data ($N :=$ #samples $\times$ #features matrix)
- z: Training labels (only in classification models; vector of size N)
- The dimension n of ω is model dependent, often #features$+1$

Introduction: ML and Optimization II

After we have found ω^*, we can do Prediction on new data points:

$$\hat{z}_i := h(\omega^*, x_i)$$

- x_i: new data point with *unknown* label z_i
- h: hypothesis function of the ML model

Introduction: Challenges in Machine Learning

- Massive amounts of training data
- Construction of very large models
- Handling high memory/computational demands

Ansatz: Stochastic Methods

Introduction: Stochastic Framework

$$F(\omega) := \mathbb{E}\left[f(\omega, \xi)\right]$$

Introduction: Stochastic Framework

$$F(\omega) := \mathbb{E}\left[f(\omega, \xi)\right]$$

- ξ: Random variable; takes the form of an input-output-pair (x_i, z_i)

Introduction: Stochastic Framework

$$F(\omega) := \mathbb{E}\left[f(\omega, \xi)\right] = \frac{1}{N}\sum_{i=1}^{N} f(\omega, x_i, z_i)$$

- ξ: Random variable; takes the form of an input-output-pair (x_i, z_i)
- f: Partial loss function corresponding to a single data point.

Introduction: Stochastic Methods

Gradient Method

$$\min F(\omega)$$

Stochastic Gradient Descent

$$\min \mathbb{E}\left[f(\omega, \xi)\right]$$

Introduction: Stochastic Methods

Gradient Method

$$\min F(\omega)$$

$$\omega^{(k+1)} := \omega^k - \alpha_k \nabla F(\omega^k)$$

Stochastic Gradient Descent

$$\min \mathbb{E}\left[f(\omega, \xi)\right]$$

Introduction: Stochastic Methods

Gradient Method

$$\min F(\omega)$$

$$\omega^{(k+1)} := \omega^k - \alpha_k \nabla F(\omega^k)$$

Stochastic Gradient Descent

$$\min \mathbb{E}\left[f(\omega, \xi)\right]$$

$$\omega^{k+1} := \omega^k - \alpha_k \nabla \hat{F}(\omega^k)$$

with

$$\nabla \hat{F}(\omega^k) := \frac{1}{b} \sum_{i \in \mathcal{S}_k} f(\omega, x_i, z_i)$$

where $\mathcal{S}_k \subset [N], \quad b := |\mathcal{S}_k| \ll N$
"Mini Batch"

Stochastic Newton Method

Stochastic Gradient Descent

$$\min \mathbb{E}\left[f(\omega, \xi)\right]$$

Stochastic Newton Method

$$\min \mathbb{E}\left[f(\omega, \xi)\right]$$

Stochastic Newton Method

Stochastic Gradient Descent

$$\min \mathbb{E}\left[f(\omega, \xi)\right]$$

Stochastic Newton Method

$$\min \mathbb{E}\left[f(\omega, \xi)\right]$$

Stochastic Newton Method

Stochastic Gradient Descent

$$\min \mathbb{E}\left[f(\omega, \xi)\right]$$

$$\omega^{k+1} := \omega^k - \alpha_k \nabla \hat{F}(\omega^k)$$

with

$$\nabla \hat{F}(\omega^k) := \frac{1}{b} \sum_{i \in \mathcal{S}_k} f(\omega, x_i, z_i)$$

where $\mathcal{S}_k \subset [N], \quad b := |\mathcal{S}_k| \ll N$

"Mini Batch"

Stochastic Newton Method

$$\min \mathbb{E}\left[f(\omega, \xi)\right]$$

$$\omega^{k+1} := \omega^k - \alpha_k \nabla^2 \hat{F}(\omega^k)^{-1} \nabla \hat{F}(\omega^k)$$

with

$$\nabla^2 \hat{F}(\omega^k) := \frac{1}{b_H} \sum_{i \in \mathcal{S}_{H,k}} f(\omega, x_i, z_i)$$

where

$\mathcal{S}_{H,k} \subset [N], \quad b := |\mathcal{S}_{H,k}| \ll N$

"Mini Batch"

Stochastic Quasi-Newton Method (SQN)

- Stochastically use second-order information
- Approximate $\nabla 2F(\hat{\omega}^k)$ by BFGS matrix H_t
- t running on slower time-scale than k.
- H_t update in $\mathcal{O}(n)$ time and constant memory, using several tricks

Behavior I

Performance on EEG Dataset, Problem size: 69550×600

Armijo-stepsizes, Further SQN-parameters: $L = 10$, $M = 5$

Behavior II

Performance on EEG Dataset, Problem size: 69550×600

Armijo-stepsizes, Further SQN-parameters: $L = 10$, $M = 5$

Results

- Can be faster than SGD on appropriate Datasets
- Requires tedious, manual tuning of hyperparameters to be efficient!

Proximal Method

Problem $$\min_x F(x) := \underbrace{f(x)}_{smooth} + \underbrace{h(x)}_{non-smooth}$$

Proximity Operator $$\mathrm{prox}_f(v) = \underset{x}{\mathrm{argmin}} \left(f(x) + \frac{1}{2}\|x - v\|_2^2 \right)$$

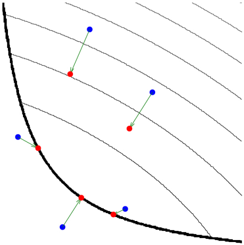


Figure : Evaluating a proximal operator at various points. *N Parikh, S Boyd, Proximal Methods, Foundations and Trends in Optimization 1, 2014*

Proximal Method

Traditional Proximal Gradient Step:

$$x_{k+1} = \text{prox}_{\lambda_k h}(x_k - \lambda_k \nabla f(x_k))$$

Quasi-Newton Proximal Step:

$$x_{k+1} = \text{prox}_h^{B_k}(x_k - B_k^{-1} \nabla f(x_k)),$$

with $B_k = \underbrace{D_k}_{diag} + \underbrace{u_k}_{\in \mathbb{R}^n} u_k^T$.

Proximal Method

$F(x) = \|Ax - b\| + \lambda \|x\|_1$
$A \in \mathbb{R}^{1500 \times 3000}, \; b \in \mathbb{R}^{1500}$
$A_{ij}, \; b_i \; \tilde{} \, \mathcal{N}(0,1), \; \lambda = 0.1$

	0SR1	ProxGrad	L-BFGS-B
Iterations	1,822	135,328	1,989
Run-Time	68 s	1,144 s	56 s

$F(x) = \|Ax - b\| + \lambda \|x\|_1$
$A \in \mathbb{R}^{2197 \times 2197}, \; b \in \mathbb{R}^{2197}$
A: Discretization of 3D Laplacian
$\lambda = 1$

	0SR1	ProxGrad	L-BFGS-B
Iterations	7	18	10
Run-Time	0.037 s	0.004 s	0.022 s

Proximal Method: Stochastic Extension

High-dimensional data: Extension to stochastic framework

Effect of batch size

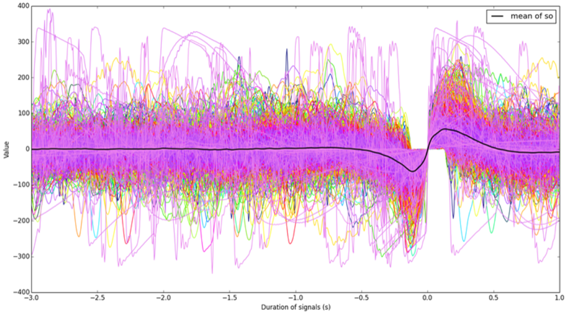
Batch size $= 1$

Batch size $= 50$

Batch size $= 150$

Electroencephalography (EEG)

How deep is your sleep?



Sleeping patient / 20 nights of EEG recordings
Predict next slow wave

EEG: Logistic Regression

Results

Nice table with SQN, SGD (no reg, L2), (Lasso,) Prox (L1) showing Obj. value in found optimum, CPU time, Iterations, F1 score of prediction model

	$F(\omega^*)$	**Model Score**	**Cost**
No regularization			
SGD	0.01	96%	x sec, y AP
SQN	0.5	96%	x sec, y AP
Prox	0.01	96%	x sec, y AP
L1			
LASSO	.71	55%	blablabla
Prox	0.01	96%	x sec, y AP
L2			
SGD	.71	55%	blablabla
SQN	0.01	96%	x sec, y AP

Can we recover the image?

Image is partially destroyed
Reconstruct image

Dictionary Learning

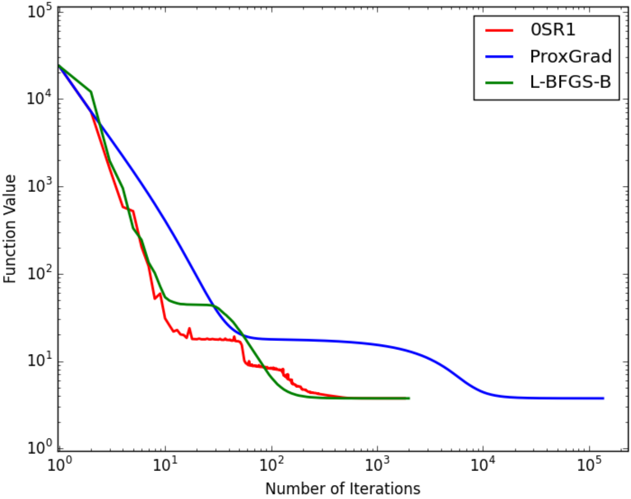
bla

Summary

Questions?

Main References I

Proximal Method



Proximal Method

$F(x) = \|Ax - b\| + \lambda \|x\|_1$
$A \in \mathbb{R}^{1500 \times 3000},\ b \in \mathbb{R}^{1500}$
$A_{ij},\ b_i\ \tilde{}\ \mathcal{N}(0,1),\ \lambda = 0.1$

$F(x) = \|Ax - b\| + \lambda \|x\|_1$
$A \in \mathbb{R}^{2197 \times 2197},\ b \in \mathbb{R}^{2197}$
A: Discretization of 3D Laplacian
$\lambda = 1$

SQN: CPU Time

Proximal Method