Stochastic Cubic Regularization for Fast Nonconvex Optimization

Nilesh Tripuraneni, Mitchell Stern, Chi Jin, Jeffrey Regier and Michael I. Jordan

Achin Jain

University of Pennsylvania

STAT991, Spring 2019

Outline

1. Motivation
2. Objectives
3. Algorithm
4. Experiments
5. References

Outline

Motivation

$$\min_{x \in \mathbb{R}^d} f(x) := \mathbb{E}_{\xi \in \mathcal{D}} \left[f(x; \xi) \right]$$

f non-convex and $f(x; \xi)$ stochastic

Variants of stochastic optimization

1. Offline setting: minimize the empirical loss over a fixed amount of data
2. Online setting: minimize the empirical loss when data arrives sequentially

Applications

- Large-scale statistics and machine learning problems
- Example: optimization of deep neural networks

Outline

Survey of (stochastic) gradient descent algorithms for ϵ-second order stationary points

Cubic-regularized gradient descent

Gradient descent

$$x_{t+1} = \arg\min_x \left[f(x_t) + \nabla f(x_t)^T (x - x_t) + \frac{L}{2} ||x - x_t||^2 \right]$$

State of the art convergence:

1. hessian free perturbed GD $\mathcal{O}\left(\epsilon^{-2}\right)$ [Jin et al., 2017]

Cubic-regularized gradient descent

$$x_{t+1} = \arg\min_x \left[f(x_t) + \nabla f(x_t)^T (x - x_t) + \frac{1}{2}(x - x_t)^T \nabla^2 f(x_t)(x - x_t) + \frac{\rho}{6} ||x - x_t||^3 \right]$$

State of the art convergence:

1. full Hessian $\mathcal{O}\left(\epsilon^{-1.5}\right)$ [Nesterov and Polyak, 2006]
2. Hessian-vector product evaluations w/o acceleration $\mathcal{O}\left(\epsilon^{-2}\right)$ [Carmon and Duchi, 2016]
3. Hessian-vector product evaluations w/ acceleration $\mathcal{O}\left(\epsilon^{-1.75}\right)$ [Carmon et al., 2018]

Cubic-regularized stochastic gradient descent

Stochastic gradient descent

$$x_{t+1} = \arg\min_x \left[f(x_t) + g(x_t, \xi_t)^T (x - x_t) + \frac{L}{2} ||x - x_t||^2 \right], \ \mathbb{E} g(x_t, \xi_t) = \nabla f(x_t)$$

State of the art convergence:

1. noisy SGD $\mathcal{O}\left(\epsilon^{-4}\right)$ [Ge et al., 2015]
2. Hessian-vector product evaluations w/ variance reduction $\mathcal{O}\left(\epsilon^{-3.5}\right)$ [Allen-Zhu, 2018]
3. gradient evaluations w/ variance reduction $\mathcal{O}\left(\epsilon^{-3.5}\right)$ [Allen-Zhu and Li, 2018]

Stochastic cubic-regularized gradient descent [this paper]

$$x_{t+1} = \arg\min_x \left[f(x_t) + g_t^T (x - x_t) + \frac{1}{2} (x - x_t)^T B_t (x - x_t) + \frac{\rho}{6} ||x - x_t||^3 \right]$$

State of the art convergence:

1. Hessian-vector product evaluations $\mathcal{O}\left(\epsilon^{?}\right)$ [Tripuraneni et al., 2018]

Problem statement

$$\min_{x \in \mathbb{R}^d} f(x) := \mathbb{E}_{\xi \in \mathcal{D}} \left[f(x; \xi) \right]$$

f non-convex and $f(x; \xi)$ stochastic

1. Can we design a fully stochastic variant of the cubic-regularized Newton method?
2. Is such an algorithm faster than stochastic gradient descent?

What's coming up

$$\min_{x \in \mathbb{R}^d} f(x) := \mathbb{E}_{\xi \in \mathcal{D}} \left[f(x; \xi) \right]$$

f non-convex and $f(x; \xi)$ stochastic

Comparison of different stochastic optimization algorithms to find an ϵ-second-order stationary point:

Method	Run-time	Variance Reduction	Type
SGD [Ge et al., 2015]	$\mathcal{O}(\epsilon^{-4})$	no needed	1^{st} order
Natasha 2 [Allen-Zhu, 2018]	$\mathcal{O}(\epsilon^{-3.5})$	needed	2^{nd} order
Neon 2 [Allen-Zhu and Li, 2018]	$\mathcal{O}(\epsilon^{-3.5})$	needed	2^{nd} order
SCR [this paper]	$\mathcal{O}(\epsilon^{-3.5})$	not needed	2^{nd} order

Outline

Assumptions

Assumption 1. The function$f(x)$ has L-Lipschitz gradients and ρ-Lipschitz Hessians for all x_1 and x_2.

$$||\nabla f(x_1) - \nabla f(x_2)|| \leq L||x_1 - x_2||, \quad \nabla^2 f(x_1) - \nabla^2 f(x_2)|| \leq \rho||x_1 - x_2||$$

Assumption 2. The stochastic gradients and stochastic Hessians, and their variance are bounded.

$$\mathbb{E}\left[||\nabla f(x,\xi) - \nabla f(x)||^2\right] \leq \sigma_1^2, \quad ||\nabla f(x,\xi) - \nabla f(x)|| \leq M_1$$

$$\mathbb{E}\left[||\nabla^2 f(x,\xi) - \nabla^2 f(x)||^2\right] \leq \sigma_2^2, \quad ||\nabla^2 f(x,\xi) - \nabla^2 f(x)|| \leq M_2$$

Cubic-regularized gradient descent

In the **deterministic** setting, we minimize the local upper bound on the function using a third-order Taylor expansion [Nesterov and Polyak, 2006].

$$m_t(x) = \left[f(x_t) + \nabla f(x_t)^T (x - x_t) + \frac{1}{2}(x - x_t)^T \nabla^2 f(x_t)(x - x_t) + \frac{\rho}{6}||x - x_t||^3 \right]$$

$$x_{t+1} = \arg\min_x m_t(x)$$

In the **stochastic** setting,

1. we only have access to stochastic gradients and Hessians, not the true gradient and Hessian,
2. our only means of interaction with the Hessian is through Hessian-vector products, and
3. the cubic submodel $m_t(x)$ cannot be solved exactly in practice, only up to some tolerance.

Stochastic Cubic Regularization (Meta-algorithm)

In the **deterministic** setting, we minimize

$$m_t(x) = \left[f(x_t) + (x - x_t)^T \nabla f(x_t) + \frac{1}{2}(x - x_t)^T \nabla^2 f(x_t)(x - x_t) + \frac{\rho}{6}||x - x_t||^3 \right]$$

$$x_{t+1} = \arg\min_x m_t(x)$$

In the **stochastic** setting, we minimize

1. $\tilde{m}(\Delta) = \Delta^T g_t + \frac{1}{2}\Delta^T \underbrace{B_t[\Delta]}_{\text{Hessian-vector product}} + \frac{\rho}{6}||\Delta||^3, \quad \Delta := x - x_t$ (we need a cubic solver)
2. $\Delta_{t+1} = \arg\min_\Delta \tilde{m}(\Delta), \quad x_{t+1} = x_t + \Delta_{t+1}$
3. $\Delta^\star = \arg\min_\Delta \tilde{m}(\Delta)$ (cubic solver will not solve exactly)
4. $\tilde{m}(\Delta) = m_t(x_t + \Delta) - m_t(x_t)$

Stochastic cubic regularization (meta-algorithm)

Algorithm 1 Stochastic Cubic Regularization (Meta-algorithm)

Input: mini-batch sizes n_1, n_2, initialization $\mathbf{x_0}$, number of iterations T_{out}, and final tolerance ϵ.

for $t = 0, \ldots, T_{\text{out}}$ **do**
  Sample $S_1 \leftarrow \{\xi_i\}_{i=1}^{n_1}$, $S_2 \leftarrow \{\xi_i\}_{i=1}^{n_2}$.
  $\mathbf{g}_t \leftarrow \frac{1}{|S_1|} \sum_{\xi_i \in S_1} \nabla f(\mathbf{x}_t;\, \xi_i)$
  $\mathbf{B}_t[\cdot] \leftarrow \frac{1}{|S_2|} \sum_{\xi_i \in S_2} \nabla^2 f(\mathbf{x}_t, \xi_i)(\cdot)$
  $\boldsymbol{\Delta}, \Delta_m \leftarrow$ Cubic-Subsolver$(\mathbf{g}_t,\, \mathbf{B}_t[\cdot], \epsilon)$
  $\mathbf{x}_{t+1} \leftarrow \mathbf{x}_t + \boldsymbol{\Delta}$
  if $\Delta_m \geq -\frac{1}{100}\sqrt{\frac{\epsilon^3}{\rho}}$ **then**
    $\boldsymbol{\Delta} \leftarrow$ Cubic-Finalsolver$(\mathbf{g}_t,\, \mathbf{B}_t[\cdot], \epsilon)$
    $\mathbf{x}^* \leftarrow \mathbf{x}_t + \boldsymbol{\Delta}$
    break
  end if
end for

Output: $\mathbf{x}^*$ if the early termination condition was reached, otherwise the final iterate $x_{T_{\text{out}}+1}$.

$$\tilde{m}(\Delta) = \Delta^T g_t + \frac{1}{2}\Delta^T B\,[\Delta] + \frac{\rho}{6}||\Delta||^3 = \Delta_m = m_t(x_t + \Delta) - m_t(x_t)$$

$$\Delta = \arg\min_{\Delta} \tilde{m}(\Delta)$$

Gradient descent as a cubic subsolver

Algorithm 2 Cubic-Subsolver via Gradient Descent

Input: $\mathbf{g}$, $\mathbf{B}[\cdot]$, tolerance ϵ.

1: **if** $\|\mathbf{g}\| \geq \frac{\ell^2}{\rho}$ **then**

2: $\quad R_c \leftarrow -\frac{\mathbf{g}^\top \mathbf{B}[\mathbf{g}]}{\rho\|\mathbf{g}\|^2} + \sqrt{\left(\frac{\mathbf{g}^\top \mathbf{B}[\mathbf{g}]}{\rho\|\mathbf{g}\|^2}\right)^2 + \frac{2\|\mathbf{g}\|}{\rho}}$

3: $\quad \mathbf{\Delta} \leftarrow -R_c \frac{\mathbf{g}}{\|\mathbf{g}\|}$

4: **else**

5: $\quad \mathbf{\Delta} \leftarrow 0, \sigma \leftarrow c' \frac{\sqrt{\epsilon\rho}}{\ell}, \eta \leftarrow \frac{1}{20\ell}$

6: $\quad \tilde{\mathbf{g}} \leftarrow \mathbf{g} + \sigma\zeta$ for $\zeta \sim \text{Unif}(\mathbb{S}^{d-1})$

7: $\quad$ **for** $t = 1, \ldots, \mathcal{T}(\epsilon)$ **do**

8: $\quad\quad \mathbf{\Delta} \leftarrow \mathbf{\Delta} - \eta(\tilde{\mathbf{g}} + \mathbf{B}[\mathbf{\Delta}] + \frac{\rho}{2}\|\mathbf{\Delta}\|\mathbf{\Delta})$

9: $\quad$ **end for**

10: **end if**

11: $\Delta_m \leftarrow \mathbf{g}^\top \mathbf{\Delta} + \frac{1}{2}\mathbf{\Delta}^\top \mathbf{B}[\mathbf{\Delta}] + \frac{\rho}{6}\|\mathbf{\Delta}\|^3$

Output: $\mathbf{\Delta}, \Delta_m$

- lines 1–3: when g is large, the submodel $\tilde{m}(\Delta)$ may be ill-conditioned, so instead of doing gradient descent, the iterate only moves one step in the g direction, which already guarantees sufficient descent [Carmon and Duchi, 2016]
- line 6: the algorithm adds a small perturbation to g to avoid a hard case for the cubic submodel

Cubic final solver

Algorithm 3 Cubic-Finalsolver via Gradient Descent

Input: $\mathbf{g}$, $\mathbf{B}[\cdot]$, tolerance ϵ.

1: $\boldsymbol{\Delta} \leftarrow 0$, $\mathbf{g}_m \leftarrow \mathbf{g}$, $\eta \leftarrow \frac{1}{20\ell}$
2: **while** $\|\mathbf{g}_m\| > \frac{\epsilon}{2}$ **do**
3: $\quad \boldsymbol{\Delta} \leftarrow \boldsymbol{\Delta} - \eta \mathbf{g}_m$
4: $\quad \mathbf{g}_m \leftarrow \mathbf{g} + \mathbf{B}[\boldsymbol{\Delta}] + \frac{\rho}{2}\|\boldsymbol{\Delta}\|\boldsymbol{\Delta}$
5: **end while**

Output: $\boldsymbol{\Delta}$

- Algorithm 2 may produce inexact $\boldsymbol{\Delta}$
- line 2, 4: gradient descent but with higher precision

Claims

Condition 1. For a small constant c, Cubic-Subsolver $(g, B[], \epsilon)$ terminates within $\mathcal{T}(\epsilon)$ gradient iterations (which may depend on c), and returns a Δ satisfying at least one of the following

1. the parameter change results in submodel and function decreases that are both sufficiently large
2. if that fails to hold, the second condition ensures that Δ is not too large relative to the true solution $\Delta^\star$, and that the cubic submodel is solved to precision $c\rho||\Delta^\star||^3$ when $\Delta^\star$ is large

Theorem 1. There exists an absolute constant c such that if $f(x)$ satisfies Assumptions 1, 2, CubicSubsolver satisfies Condition 1 with c, then for all $\delta > 0$ and $\Delta_f \geq f(x_0) - f^*$, and $\epsilon \leq \min\{\frac{\sigma_1^2}{cM_1}, \frac{\sigma_2^4}{c^2 M_2^2 \rho}\}$, Algorithm 1 will output an ϵ-second-order stationary point of f with probability at least $1 - \delta$ within

$$\mathcal{O}\left(\frac{\sqrt{\rho}\Delta_f}{\epsilon^{1.5}}\left(\frac{\sigma_1^2}{\epsilon^2} + \frac{\sigma_2^2}{\rho\epsilon}\mathcal{T}(\epsilon)\right)\right)$$

total stochastic gradient and Hessian-vector product evaluations.

Claims

Lemma 1. There exists an absolute constant c, such that under the same assumptions on $f(x)$ and the same choice of parameters n_1, n_2 as in Theorem 1, Algorithm 2 satisfies Condition 1 with atleast $1-\delta$ with

$$\mathcal{T}(\epsilon) \leq \mathcal{O}\left(\frac{L}{\sqrt{\rho\epsilon}}\right)$$

Corollary 1. Under the same settings of Theorem 1, if we instantiate CubicSubsolver with Algorithm 2, and $\epsilon \leq \min\{\frac{\sigma_1^2}{cM_1}, \frac{\sigma_2^4}{c^2M_2^2\rho}\}$, then Algorithm 1 will output an ϵ-second-order stationary point of f with probability at least $1-\delta$ within

$$\mathcal{O}\left(\frac{\sqrt{\rho}\Delta_f}{\epsilon^{1.5}}\left(\frac{\sigma_1^2}{\epsilon^2} + \frac{\sigma_2^2}{\rho\epsilon^{1.5}}\frac{L}{\sqrt{\rho}})\right)\right)$$

total stochastic gradient and Hessian-vector product evaluations.

Proof Sketch

Claim 1.

If x_{t+1} is not an ϵ-second-order stationary point of $f(x)$, the cubic submodel has large descent $m_t(x_{t+1}) - m_t(x_t)$.

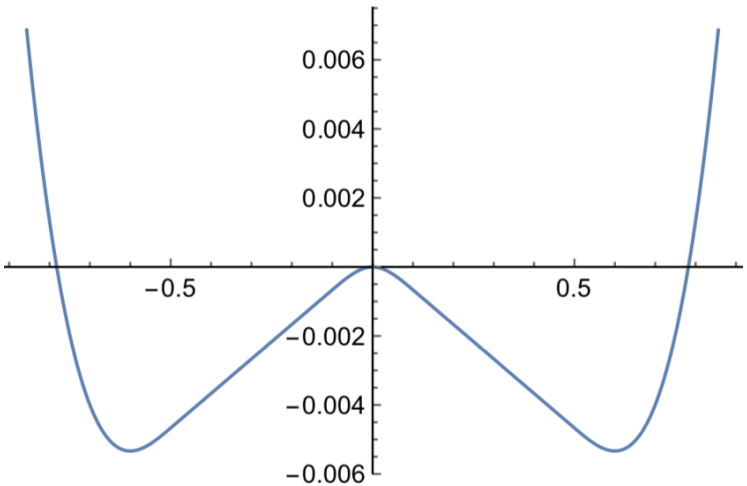
Claim 2.

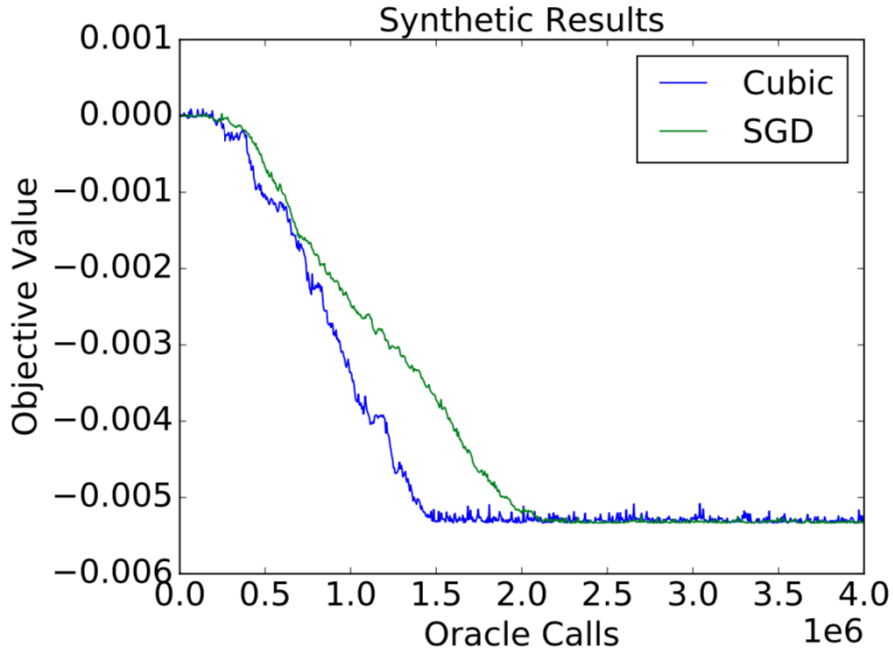
If the cubic submodel has large descent $m_t(x_{t+1}) - m_t(x_t)$, then the true function also has large descent $f(x_{t+1}) - f(x_t)$.

Outline

Synthetic Nonconvex Problem

Piece-wise cubic function $w(x_1)$

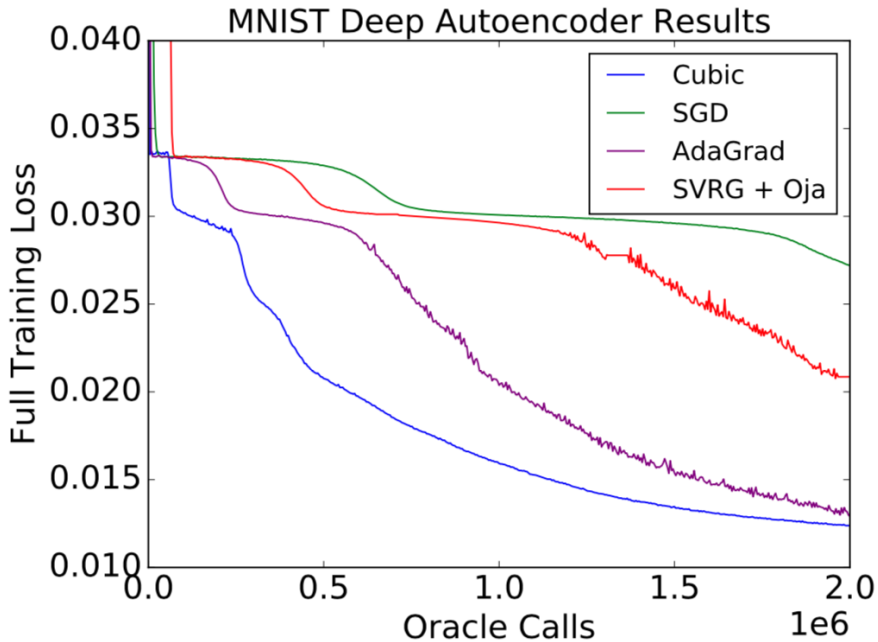




$$\min_{x \in \mathbb{R}^2} \left[w(x_1) + 10x_2^2 \right]$$

Algorithm 1 is able to escape the saddle point at the origin and converge to one of the global minima faster than SGD.

Deep Autoencoder



Encoder: $(28 \times 28) \to 512 \to 256 \to 128 \to 32$

Decoder: $(28 \times 28) \leftarrow 512 \leftarrow 256 \leftarrow 128 \leftarrow 32$

$$\min \sum \text{pixelwise } L_2 \text{ loss}$$

Outline

1. Motivation
2. Objectives
3. Algorithm
4. Experiments
5. References

References I

Allen-Zhu, Z. (2018).
Natasha 2: Faster non-convex optimization than SGD.
In **Advances in Neural Information Processing Systems**, pages 2675–2686.

Allen-Zhu, Z. and Li, Y. (2018).
Neon2: Finding local minima via first-order oracles.
In **Advances in Neural Information Processing Systems**, pages 3716–3726.

Carmon, Y. and Duchi, J. C. (2016).
Gradient descent efficiently finds the cubic-regularized non-convex newton step.
arXiv preprint arXiv:1612.00547.

Carmon, Y., Duchi, J. C., Hinder, O., and Sidford, A. (2018).
Accelerated methods for nonconvex optimization.
SIAM Journal on Optimization, 28(2):1751–1772.

Ge, R., Huang, F., Jin, C., and Yuan, Y. (2015).
Escaping from saddle points – online stochastic gradient for tensor decomposition.
In **Conference on Learning Theory**, pages 797–842.

Jin, C., Ge, R., Netrapalli, P., Kakade, S. M., and Jordan, M. I. (2017).
How to escape saddle points efficiently.
In **Proceedings of the 34th International Conference on Machine Learning-Volume 70**, pages 1724–1732. JMLR. org.

References II

Nesterov, Y. and Polyak, B. T. (2006).
Cubic regularization of newton method and its global performance.
Mathematical Programming, 108(1):177–205.

Tripuraneni, N., Stern, M., Jin, C., Regier, J., and Jordan, M. I. (2018).
Stochastic cubic regularization for fast nonconvex optimization.
In **Advances in Neural Information Processing Systems**, pages 2899–2908.