THE PENNSYLVANIA STATE UNIVERSITY
SCHREYER HONORS COLLEGE

HAROLD AND INGE MARCUS DEPARMENT OF INDUSTRIAL AND MANUFACTURING ENGINEERING

MODIFIED STOCHASTIC VARIANCE REDUCTION GRADIENT DESCENT ALGORITHM AND ITS APPLICATION

CAI FEI
FALL 2018

A thesis
submitted in partial fulfillment
of the requirements
for a baccalaureate degree
in Industrial Engineering
with honors in Industrial Engineering

Reviewed and approved* by the following:

Tao Yao
Associate Professor of Industrial and Manufacturing Engineering
Thesis Supervisor

Catherine Harmonosky
Associate Professor of Industrial and Manufacturing Engineering
Honors Adviser

* Signatures are on file in the Schreyer Honors College.

ABSTRACT

While machine learning is becoming an indispensable element in our modern society, various algorithms are developed to help decision makers solve complicated problems. A major theme of this study is to review and analyze popular algorithms with a focus on Stochastic Gradient (SG) based methods in large-scale machine learning problems. While SG has been the fundamental method playing an essential role in optimization problems, the algorithm has been further modified by various researchers for improved performances. Stochastic Gradient Descent with Variance Reduction (SVRG) is a method known for its low computation cost and fast convergence rate in solving convex optimization problems. However, in nonconvex settings, the existence of saddle points negatively influences the performance of the algorithm. While the practical problems in the real-world majorly lie in nonconvex settings, to further improve the performance of SVRG, a new algorithm is designed and discussed in this study. The new algorithm combines traditional SVRG with two additional features introduced by Perturbed Accelerated Gradient Descent (Perturbed AGD) to expedite algorithm from escaping from saddle points, which ultimately leads to convergence in nonconvex optimization. This study focuses on the elaboration of the modified SVRG algorithm and its implementation with a synthetic and an empirical dataset.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to thank Dr. Tao Yao and his PhD student, Xue Wang, for their support and guidance in this thesis study. this study would not be completed without their help and input. I also want to thank Dr. Catherine Harmonosky for her continuous support during my honors academic experience at Penn State. I would also like to thank all the professors and staff in the IME department for their dedication. They have created such an enjoyable and supportive learning environment for all IE students.

I would also like to thank my parents for always believing in me and supporting me in whatever I pursue. This degree would not be possible without their consistent encouragement and support. I am forever grateful.

I would like to send special thanks to my friends: Chenyu, Yuki, Yaqun, as well as all my other friends for helping me go through the most stressful time in college. They have been helping me fight my frustrations and stay motivated in this journey.

Thanks Penn State, the IME department and the Schreyer Honors College for the great four years. I will always be proud of my Penn State roots.

Chapter 1

Motivation of Study

Immersed in the age of internet, information and data, we have witnessed how the Internet of Things is broadening the ways people live, think and make decisions. With artificial intelligence continuously influencing our day to day activities, researchers are taking advantages of the increasing availability of immense data pool to constantly search for revolutionary innovations that could further help us progress in different industries such as healthcare, economics and natural science.

Constructed with the basis of statistics and numerical computations, machine learning algorithms have thrived in recent years and have made great contributions in the society. Our day to day activities are filled with the applications of machine learning. From the "people you may know" suggestions on social media platforms, the automatic spam filtering in our mailboxes, to virtual assistants such as Siri and Alexa, machine learning has greatly influenced the way we live, in one way or the other, and sometimes we do not even realize it. Moreover, machine learning has been a popular topic in healthcare industry in recent years. With more and more patients' data and medical records available, researchers are able to develop algorithms enabling personalized medical treatment for each individual patient. A recent study has developed a personalized approach for type 2 diabetes management that efficiently improves the standard of care (Bertsimas et al, 2017). Overall, machine learning has rooted in all types of different applications, and are continuously bringing more contributions for the society.

All machine learning applications are made possible through the development and implementation of algorithms. Generally, programmed algorithms are designed to use unseen input data to learn and optimize their operations which ultimately enable the prediction of output data. Mathematical optimization, as the bedrock of all machine learning problems, is the process of computing parameters for a problem designed for making decisions based on unseen data. While some of the machine learning algorithms have been around for some time, researchers are constantly searching for faster and cheaper algorithms to solve the underlying mathematical optimization problems.

A main objective of this study is to review the existing machine learning optimization algorithms, addressing its advantages and disadvantages. After a general overview of the common algorithms, this thesis presents a modified algorithm. The modified algorithm is then applied to a synthetic dataset and an empirical dataset.

Chapter 2

Introduction and Literature Review

The goal of an optimization problem is to find an approximate solution of the optimization problem, which is usually addressed by a regularized finite-sum minimization:

$$\min f(\beta) := \frac{1}{n}\sum_{i=1}^{n} f_i(\beta) = \frac{1}{n}\sum_{i=1}^{n} \mathrm{L}(\beta, x_i, y_i) + \lambda \mathrm{R}(\beta)$$

Where $\beta \in R^d$ is the model parameter and n is the number of samples (x_i, y_i). $\mathrm{L}(\beta, x_i, y_i)$ is the loss function, and $\lambda \mathrm{R}(\beta)$ term is the regularizer with a regularization parameter λ ($\lambda \geq 0$).

2.1 Gradient Descent Methods: Batch and Stochastic

The fundamental method of solving this minimization function is through gradient descent, attempting to move to the global minimum of the function. Gradient descent is updated by iteratively moving in the direction of the steepest descent, while the steepest descent is the negative of the gradient of the function at current point.

Gradient descent methods generally fall into two categories: batch and stochastic. While both methods have their trade-offs, stochastic method is usually more popular in large-scale machine learning as it is cheaper in computation.

The most fundamental approach in batch methods is full gradient descent with the following update rule with a step-size of η:

$$\beta_t = \beta_{t-1} - \frac{\eta_t}{n}\sum_{i=1}^{n} \nabla f_i(\beta_{t-1}).$$

This method computes the gradient of cost function to parameter β for the entire training dataset, and finds the ϵ-first order stationary point in $O(1/\epsilon^2)$ iterations (Nesterov, 1998), however this standard method is extremely expensive in computation since it requires the evaluation of n derivatives at each step, in other words, a sum of n calculations of the inner product $\beta^T x_i =$ is needed, which can be extremely costly when we have a large sample.

In order to achieve cheaper computation, Stochastic Gradient Descent is introduced, where at each iteration, we draw a random sample i_t from {1,2,…,n}, and gradient is updated by

$$\beta_t = \beta_{t-1} - \eta_t \nabla f_{i_t}(\beta_{t-1}) = \beta_{t-1} - \eta_t g_t(\beta_{t-1}, \xi_t)$$

Each step relies only on one derivative ∇f_{i_t}, and the calculation cost is independent of n, thus its computational cost is 1/n of the full batch method. However, the underlying trade-off is that the variance generated from random sampling can be very large since we are approximating the full gradient by a noisy estimate with the noisy estimate being unbiased. SGD updates causes the objective function to fluctuate heavily as illustrated in the Figure 1:

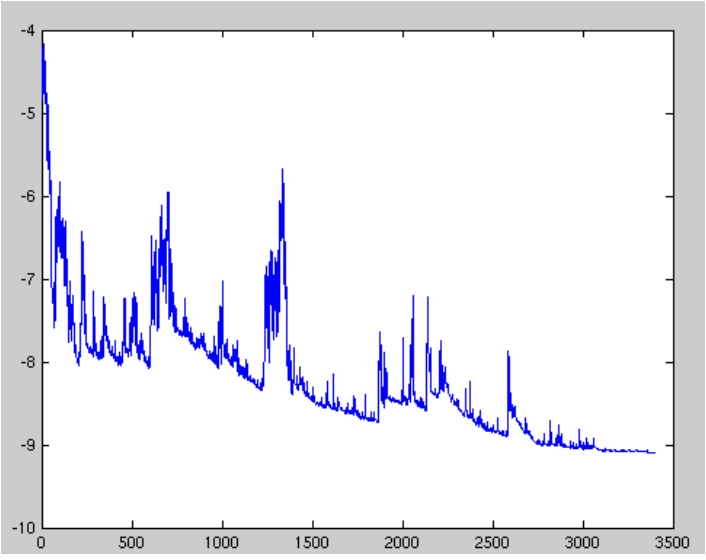

Figure 1: SGD Fluctuations(Wikipedia)

2.2 Variations of GD: SGD with Variance Reduction

To further solve the problem of high variance generated from random sampling, Johnson and Zhang proposed a method based on traditional SGD – Stochastic Variance Reduction Gradient Descent method (SVRG) (Johnson & Zhang, 2012) which leads to superior convergence properties in solving convex problems. The key innovation of this algorithm is through exploiting a full gradient estimation explicitly or implicitly to reduce the variance of the noisy stochastic gradient. In practice, the algorithm includes a snapshot of $\tilde{g}$ taken after every m SGD iterations,

$$\tilde{g} = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\tilde{\beta}).$$

the gradient is then updated in the following rule:

$$\beta_t = \beta_{t-1} - \eta_t(\nabla f_i(\beta_{t-1}) - (\nabla f_{i_t}(\tilde{\beta}) + \tilde{g})$$

It is proved that learning rate η_t for SVRG does not have to decay with this algorithm, making it a superior algorithm in convex problem setting.

Here rises another major problem in modern machine learning. While traditional algorithms perform well in convex settings, most of the complex practical problems nowadays appear to be non-convex in nature. Since first-order stationary points includes local minima, saddle points or local maxima, traditional gradient- based algorithm can have poor performance in solving these problems.

The existence of saddle points is the main challenge in nonconvex optimizations, where traditional gradient descent gets stuck at. While Johnson and Zhang mainly focused on the proof of SVRG's superiority in convex settings, they have not proved its superiority in nonconvex optimization.

2.3 Variations of GD: Momentum Based GD

Besides variance reduced Gradient Descent, another variation of Gradient Descent is developed and commonly known as Accelerated Gradient Descent, an instance of "momentum methods". Nesterov (1983) has showed that the accelerated version of gradient descent (AGD) finds ϵ-suboptimal point in $O(1/\sqrt{\epsilon})$ steps, faster than $O(1/\epsilon)$ steps required by GD

Figure 2: Comparison of SGD without Momentum(Left) And with Momentum(Right) (Willamette.edu)

This momentum-based method performs a traditional step of gradient descent with sliding a little bit further in the direction given by the previous point, as indicated in Table 1.

Table 1:Momentum Based Method Update Rule

$v_0 \leftarrow 0$

$for\ t = 0,1,2 \ldots$

$$\beta_t = \beta_t + (1 - \theta)v_t$$

$$\beta_{t+1} = \beta_t - \eta \nabla f(\beta_t)$$

$$v_{t+1} = \beta_{t+1} - \beta_t$$

While the momentum-based algorithm has been proved to have fast convergence in convex problems, the question of its potential performance in non-convex settings with second-order stationary points attracted various researchers.

2.3 Variations of GD: Perturbed AGD

Some researchers have focused on the convergence to a second-order stationary point to eliminate common types of saddle points, allowing only local minima and higher-order saddle points. It is proved by researchers that for most machining learning problems, neither higher-order saddle points nor spurious local minima exist, thus, all second-order stationary points are approximately global minima (Kawaguchi, 2016).

Based on the proof of non-existence of higher-order saddle points and spurious local minima, as well as Nesterov's AGD algorithm, Jin and Netrapalli (2017) proposed an innovative variation of AGD with 2 main additional features: Perturbation and Negative Curvature Exploitation. These two techniques enable algorithm to escape saddle points faster at non-convex settings.

With the traditional AGD gradient update, the perturbation step is added no more than once in a certain number of steps when the gradient is too small to ensure that gradient does not stuck at saddle points. The perturbation ξt is sampled uniformly from d-dimensional ball with radius r. The algorithm is illustrated in Table 2.

Table 2: Perturbation Algorithm

PERTURBATION

for s = 0,1,…,

if $\| \nabla f(\tilde{\beta}_s) \| <= \epsilon$ and no perturbation done in last τ steps

$\tilde{\beta}_s = \tilde{\beta}_s + \xi_s \qquad \xi_s \in Unif\ (B_o(r))$

Negative Curvature Exploitation ensures the gradient moves through the direction where the function value decreases. Table 3 shows the algotirhm of NCE.

Table 3: Negative Curvature Exploitation Algorithm

Negative Curvature Exploitation

if $f(\tilde{\beta}_{s-1}) <= f(\tilde{\beta}_s) + <\nabla f(\tilde{\beta}_s), \beta_m - \tilde{\beta}_s> - \frac{\gamma}{2} \| \tilde{\beta}_s - \tilde{\beta}_{s-1} \|^2$

$v = \tilde{\beta}_s - \tilde{\beta}_{s-1}$

if $\| v \| <= k$

$\tilde{\beta}_{s+1} = argmin_{\beta \in \{\tilde{\beta}_s + \delta, \tilde{\beta}_s - \delta\}} f(\beta)$, where $\delta = k * v / \| v \|$

With these two techniques, it is proved that the Perturbed AGD finds an ϵ-second order stationary point in $\tilde{O}(1/\epsilon^{\frac{7}{4}})$ iterations, much faster than $\tilde{O}(1/\epsilon^2)$ required by Perturbed GD (Jin et al, 2017).

Chapter 3

Problem Statement and Formation

Now the question is that how we can further improve the optimization algorithm so that it stays cheap in computation, maintains fast convergence but also does not stuck at saddle points while solving nonconvex optimization problems. This question serves as the main theme of this research study.

Based on related work, it is shown that SVRG can satisfy fast convergence as well as cheap computation in convex problems, and Perturbed AGD presents two techniques to ensure that gradients do not stuck at saddle points in non-convex settings. Intuitively, the combination of SVRG with Perturbation, NCE techniques from Perturbed AGD would be a more an efficient algorithm comparing the original SVRG, as well as the PAGD.

This project takes a step further to verify that whether this intuition can be proved to be numerically validated, in other words, this thesis study aims to verify that whether Modified SVRG algorithm with techniques from Perturbed AGD can reach convergence efficiently while solving large scale nonconvex problems.

The following sections of this paper then focuses on elaborating how to modify the original SVRG by adding additional feature and how it is implemented with the following objectives:

- Develop Modified SVRG algorithm based on Original SVRG and Perturbed AGD

- Implement the modified algorithm in a coding environment to solve a synthesized nonconvex optimization problem, and compare the numerical results of Modified SVRG and Perturbed AGD
- Apply the modified algorithm to an empirical dataset and report the numerical result
- Propose how further improvements may be made and what the open problems there are for other researchers

Chapter 4

Modified Stochastic Variance Reduction Gradient Descent Algorithm

To further improve SVRG's performance in nonconvex optimization, modified SVRG algorithm is developed combing the Perturbation and Negative Curvature Exploitation technique, shown in Table 4.

Table 4: Modified Stochastic Variance Reduction Gradient Descent Algorithm

MODIFIED SVRG $(\tilde{\beta}_0, \epsilon, \eta, \gamma, k, \tau, r)$

STEP 1 Add Perturbation

if $\| \nabla f(\tilde{\beta}_s) \| <= \epsilon$ and no perturbation in last τ steps

$\tilde{\beta}_s = \tilde{\beta}_s + \xi_s \qquad \xi_s \in Unif\ (B_o(r))$

STEP 2 SVRG update

Initialize $\tilde{\beta}_0$

Iterate: for s = 1,2,…

$\tilde{\beta} = \tilde{\beta}_{s-1}$

$\tilde{g} = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\tilde{\beta})$

$\beta_0 = \tilde{\beta}$

Iterate: for t = 1,2,…m

Randomly pick i_t from {1,2,…,n} and update

$\beta_t = \beta_{t-1} - \eta(\nabla f_{i_t}(\beta_{t-1}) - (\nabla f_{i_t}(\tilde{\beta}) + \tilde{g})$

end

set $\tilde{\beta}_s = \beta_m$

end

STEP 3 Negative Curvature Exploitation

if $f(\tilde{\beta}_{s-1}) <= f(\tilde{\beta}_s) + <\nabla f(\tilde{\beta}_s), \beta_m - \tilde{\beta}_s> - \frac{\gamma}{2} \| \tilde{\beta}_s - \tilde{\beta}_{s-1} \|^2$

$v = \tilde{\beta}_s - \tilde{\beta}_{s-1}$

if $\| v \| <= k$

$\tilde{\beta}_{s+1 = argmin\ \beta \in \{\tilde{\beta}_s + \delta, \tilde{\beta}_s - \delta\}} f(\beta)$, where $\delta = k * v/\| v \|$

Similar with Perturbed AGD, perturbation step is added first to fluctuate the gradient in order to make the gradient escape from saddle points or near saddle points. When the gradient is smaller than pre-set parameter ϵ, and perturbation has not been done is the past τ steps, a small perturbation is sampled uniformly and added to the gradient. This perturbation step mitigated the possibility that the algorithm is stuck at or near saddle points.

After the perturbation step, Standard SVRG update is then followed.

NCE is then added at the end when the function becomes too nonconvex along the $\tilde{\beta}_{s} - \tilde{\beta}_{s-1}$. We reset the gradient and decide which direction we should exploit at the current step by comparing which direction would leads to the decrease of the function value.

In the following chapter of this study, the modified algorithm is applied to a synthetic dataset and an empirical dataset. Both applications are done through MATLAB.

Chapter 5

Application and Numerical Results

The algorithm is applied in two datasets: a synthetic dataset with arbitrary objective function, and an empirical dataset retrieved from an online open source.

The following applications starts by applying the modified algorithm and the benchmarks in a synthetic dataset. As the main techniques of this modified algorithm is derived from Perturbed AGD, Perturbed AGD will serve as the benchmark algorithms for the synthetic dataset. The modified algorithm will then be applied to an empirical dataset retrieved from an online open source, followed by the conclusions and other open problems.

While various programming languages such as Python, R or Julia, can be used for the application, this specific application is developed through MATLAB. A sample code of modified SVRG algorithm for empirical data can be found in this chapter for reference purpose.

5.1 Synthetic Data

5.1.1 Dataset Formation and Objective Function

A three-dimensional dataset was generated with a preset parameter β, and a randomly generated X, Y matrices with 100 data samples. X = randn(100,3) and Y = X* β + randn(100,1)

to solve a nonconvex optimization function 1/100 $(X\beta-Y)'(X\beta-Y) -10\beta^2 +\beta^4$. The objective function consists a least-square term: $(X\beta-Y)'(X\beta-Y)$ and two additional terms: $-10\beta^2 + \beta^4$. Since the least square is convex by itself, the two additional terms are added for the purpose of creating a non-convex optimization problem.

5.1.2 Computational Efficiency

Perturbed SVRG's code is run in MATLAB for 200 times. The number of iterations needed each time for convergence is used as the criteria of determining the computational efficiency of the algorithm. Figure 3 shows distribution of number of iterations for each run, where the average number of iterations of the total 200 runs is 6500.

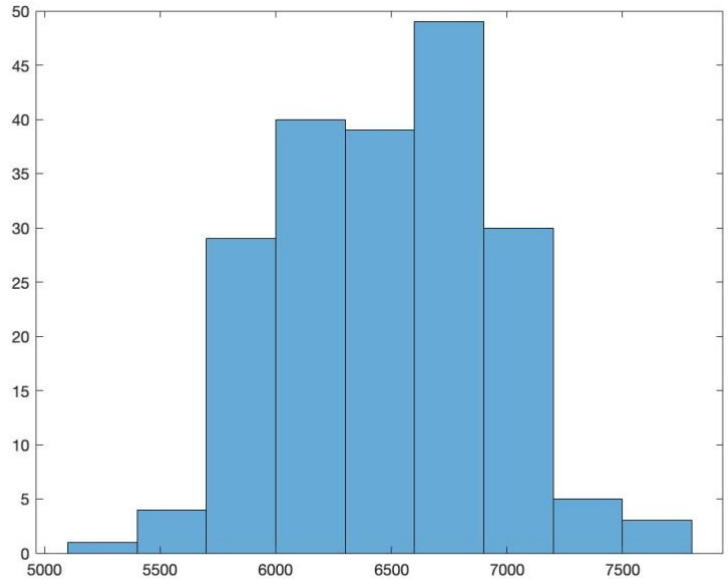

Figure 3: Number of Iterations for 200 Runs

To compare the results with the benchmark algorithm: Perturbed AGD, PAGD's code is also run in MATLAB for 200 times.

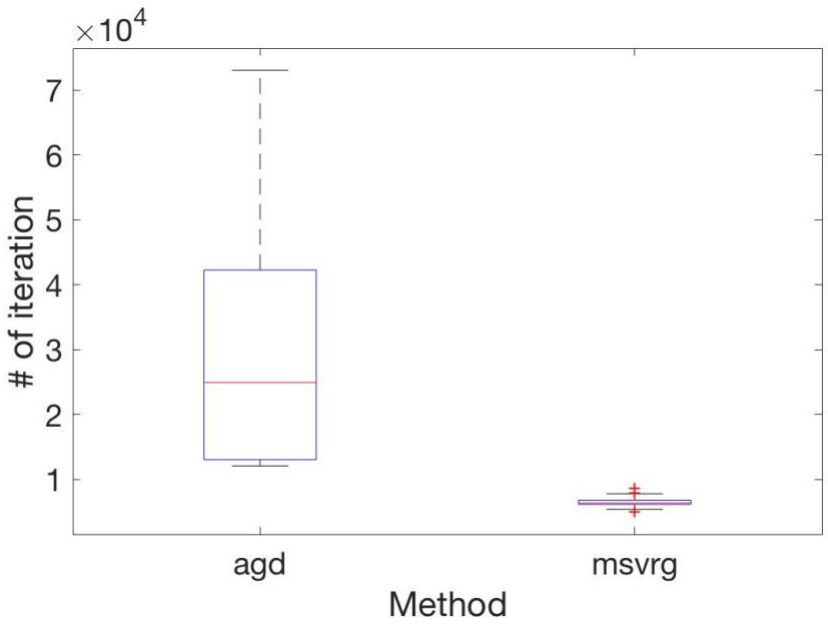

Figure 4: Comparison of Perturbed AGD and Modified SVRG

Figure 4 shows the comparison of number of iterations while using Perturbed AGD and Modified SVRG. While Perturbed AGD requires around in average 23,000 iterations to converge, the Modified SVRG only needs around 6500 iterations to converge. It is showing that Modified SVRG takes significantly smaller average number of iterations to converge comparing to Perturbed AGD, further proves that Modified SVRG is potentially a much more efficient algorithm comparing to Perturbed AGD.

To further investigate the Modified SVRG algorithms, an empirical dataset is used to evaluate their performances in a more complicated non-convex setting.

5.2 Empirical Data

5.2.1 Overview of Dataset

Implementing machine learning algorithm has been a popular approach of providing personalized services in various industries including the banking and financial industry. With the increasing availability of customer data, banking industry has invested in providing more personalized services for customers. With customers' past investment data stored in the database, decision makers can further predict each customers' potential future investment so that they can tailor down customers' specific needs and offer personalized investment package.

The dataset chosen for this study is the Santander Value Prediction Data, extracted from an open source data sharing website: www.kaggle.com (Kaggle). The dataset contains a training set as well as a testing set.

The training set contains 4993 columns, with the first column being the customer ID, the second column being the prediction target, and the rest of columns have either 0 or the value of customers' past recorded investment. Figure 5 provides an overview of the dataset.

ID	target	48df886f9	0deb4b6a8	34b15f335	a8cb14b00	2f0771a37	30347e683	d08d1fbe3	6ee66e115	20aa07010	dc5a8f1d8	11d86fa6a	77c9823f2	8d6c2a0b2	4681de4fd	adf119b9a	cff75dd09	96f83a237
000d6aaf2	38000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
000fbd867	600000	0	0	0	0	0	0	0	0	2200000	0	0	0	0	0	0	0	0
0027d6b71	10000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0028cbf45	2000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
002a68644	14400000	0	0	0	0	0	0	0	0	2000000	0	0	0	0	0	0	0	0
002dbeb22	2800000	0	0	0	0	0	0	0	0	17020000	0	8000	0	0	0	0	0	0
003925ac6	164000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
003eb0261	600000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
004b92275	979000	0	0	0	0	0	0	0	0	58000	0	0	0	0	22000	0	0	0

Figure 5: Overview of Santander Dataset

5.2.2 Objective Function Formulation

Because of the complexity of the dataset, to prevent the problem of over fitting, a regularization term is added in this experiment to control the model complexity by using a penalty term, with the objective function being:

$$\min f(\beta) = L(\beta) + \lambda R(\beta)$$

While the loss function term $L(\beta)$measures the discrepancy between the model and the actual observation, and $R(\beta)$ serves as a penalty to prevent over fitting.

In this experiment, the loss function is the widely-used least squares loss function, a convex property, and the regularization term is the Log Sum Penalty (LSP), a non-convex regularizer.

$$R(\beta) = \log(1 + |\beta_i|/\theta) \quad (\theta > 0)$$

These two terms together present a non-convex optimization problem. The functions are defined in MATLAB as shown in Figure 6.

```
%define loss function
funcl = @(beta_0) 1/1000*(X*beta_0-Y)'*(X*beta_0-Y);
%define LSP regularizer
funcr = @(beta_0) numda * log ((abs(beta_0)/theta)+1);
%define objective function
func = @(beta_0) 1/1000*(X*beta_0-Y)'*(X*beta_0-Y)+ numda * log ((norm(beta_0))/theta+1);
```

Figure 6: Objective Function in Coding Environment

5.2.3 Algorithm in Coding Environment

Since it is known to be challenging to solve non-convex optimization with non-convex regularizer, the General Iterative Shrinkage and Thresholding (GIST) (Gong, 2013) algorithm is used for calculate the gradient outside of the modified SVRG's epoch along with the modified SVRG algorithm for the dataset.

The gradient is updated with the following rule:

$$\beta^{k+1} = argmin\ \frac{1}{2}\| \beta - u^k\|^2 + \frac{1}{t} r(\beta)$$

where $u^k = \beta^k - \ \nabla l(\beta^k)/t$ and t is the stepsize. A findmin function is created in MATLAB to determine the β^{k+1} in each update. The function is coded in MATLAB, shown in Figure 7.

```
function beta = findmin_new(u, numda, theta)

func = @(beta_i, u_i) 0.5*(beta_i-u_i).*(beta_i-u_i) + ...
    numda * log (1 + abs(beta_i)/theta);

n = length(u);
t = 1000;
candidate = NaN(n,5);
beta = zeros(size(u));

I1 = find(abs(u)<numda/t);
candidate(I1,1) = 0;

value1 = sqrt ((theta - u).* (theta - u) - ...
    4 * theta * ((numda/t) - u));
beta_s_1 = 0.5*((u - theta) + value1);
beta_s_2 = 0.5*((u - theta) - value1);
I2 = find(beta_s_1>0);
candidate(I2,2) = beta_s_1(I2);
I3 = find(beta_s_2>0);
candidate(I3,3) = beta_s_2(I3);

value2 = sqrt ((theta + u).* (theta + u) - ...
        4 * theta * ((numda/t)+ u));
beta_s_3 = 0.5*((u + theta) + value2);
beta_s_4 = 0.5*((u + theta) - value2);
I4 = find(beta_s_3<0);
candidate(I4,4) = beta_s_3(I4);
I5 = find(beta_s_4<0);
candidate(I5,5) = beta_s_4(I5);

fmin_all = func(candidate,u.*ones(n,5));
[kappa,index] = min(fmin_all');
    beta = kappa';
end
```

Figure 7: Findmin Function in Coding Environment

To ensure that the algorithm will sufficiently escape the saddle points, the Perturbation and NCE steps are added in MATLAB as shown in Figure 8.

```
if norm(g) <= epsi && flag>=zeta
    ksi = radius*rand (4991,1,1)-radius/2;
    beta_s_bar = beta_s_bar+0.01*ksi;
    flag = 0;
end
```

Figure 8: Perturbation Step in Coding Environment

The code indicated that if the norm of the gradient is smaller than the arbitrary parameter epsi (refer to ϵ in Table 1), and no perturbation is done in the recent zeta (refer to τ in Table 1) steps, then we are drawing a random sample from d-dimensional ball with radius r and adding it to the current β.

The standard SVRG update is then followed, shown in Figure 9.

```
for i = 1:500
    beta_s_bar_0 = beta_s_bar;
    g = 1/500*df_i(i,beta_s_bar);
    if norm(g) <= epsi && flag>=zeta
        ksi = radius*rand (4991,1,1)-radius/2;
        beta_s_bar = beta_s_bar+0.01*ksi;
        flag = 0;
    end
    beta_s = beta_s_bar;
    g_bar = 1/500 * df_i(i,beta_s_bar);
    funcu = @(beta_s_bar) beta_s_bar -  g_bar/t;
    flag = flag + 1;

    u = funcu(beta_s_bar);

    funccn = @(beta_s) (1/2) * (u-beta_s)'*(u-beta_s) + ...
    sum (numda * log(1+abs(beta_s)/theta));
     summation = Zero;
     for k = 1:500
     beta_s_bar = findmin_new(u, numda, theta);
     summation = summation + beta_s_bar;
     end
     snapshot = summation/100;
        for t = 1:1000
            g_i = randperm(500,1);
            g_bar = 1/500*df_i(g_i,beta_s_bar);
            beta_s_1 = beta_s - 0.0001*(g - g_bar + snapshot );
            flag_1 = flag_1 +1;
         end
     beta_s = beta_s_1;
```

Figure 9: SVRG in Coding Environment

At the end of the SVRG update, the NCE step is added as shown in Figure 10.

```
if func(beta_s_bar)<= func(beta_s)+g_bar'*(beta_s_bar-beta_s)...
        -gamma/2*norm(beta_s_bar-beta_s)^2
    vector = beta_s_bar-beta_s;
    if norm (vector) >= 0.000001
        beta_s_bar = beta_s;
    else delta = 0.000001*vector/norm(vector);
        plus = beta_s+delta;
        minus = beata_s-delta;
        if func(plus)>func(minus)
            beta_s_bar = minus;
        else beta_s_bar = plus;
        end
    end
```

Figure 10: NCE Step in Coding Environment

At this step, we are first checking if we should reset the gradient. If so, then we check which direction we should exploit at the current step by comparing which direction would leads to the decrease of the function value. The gradient is then reset to be the one that leads to the decrease of the function value.

5.2.4 Computational Efficiency and Estimator Quality

The algorithm is run for 10 times in MATLAB with step size $\eta = 0.0001$, the average number of iterations for convergence is 284,000.

To measure the quality of the estimator, mean squared errors are calculated with function:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Yi - \widehat{Y}\imath)^2$$

This step was also realized in MATLAB with built in function err = immse(X, Y).

The MSE is $1.2907*10^6$. This value may seem large, but the target values are large in nature with an average value of 7,000,300, and the MSE value is only 18.44% of the average target value. The quality of the estimator is therefore more than acceptable. This application has again illustrated the modified SVRG's ability in solving large-scale, complex non-convex problems.

Chapter 6

Conclusion and Open Problems

In this project, a variant of SVRG is introduced and it is shown to be capable of escaping saddle points in non-convex optimization. The numerical results have shown that the Modified SVRG outperforms the Perturbed AGD significantly in a synthetic dataset application. The modified SVRG algorithm requires in average 6,500 interations to converge to global minima, which is much less than Perturbed AGD that requires in average 23,000 iterations for convergence. It has also shown its ability to solve real-world non-convex optimization problems through the application of the Santander Value prediction data with an average iteration number of 284,000 to find global minima.

Intuitively, adding the extra two steps of perturbation and NCE would improve the algorithm's ability to escape from saddle points, however, the modified SVRG's convergence efficiency would be more convincing with further theoretical proof. Due to the scope of this thesis, theoretical proves are not provided in this project and it remains an open problem for other researchers to further prove that the two add-in techniques would be theoretically improving algorithm's ability in escaping saddle points and converging to global optimal.

Another important note is that, for stochastic optimization algorithms, the performance of the algorithm is strongly influenced not only by the data, but also by the step-size selection. While various literatures have extensively studied how to select step-size for algorithms, how to effectively set the step-size remains an open question for the modified SVRG algorithm.

Meanwhile, due to the nature of the modified SVRG algorithm, there are also other arbitrary parameters to be set while applying the algorithm, such as parameters $\epsilon,\ \tau$ involved in the perturbation step, as well as γ involved in the NCE step. It remains an open question for others to further investigate how to find the values for these arbitrary parameters in order to ensure the optimal performance of the algorithm.

The field of machine learning is evolving rapidly, with new concept being introduced and various new algorithms being developed. It is extremely important for any researcher to keep pace with the innovations in the industry. With the collaborative and continuous effort from the research community, there is no doubt that machine learning will keep improving the ways we live, think, and make decisions.

BIBLIOGRAPHY

Bertsimas, D., Kallus, N., Weinstein, A. M., & Zhuo, Y. D. (2016). Personalized Diabetes Management Using Electronic Medical Records. *Diabetes Care,40*(2), 210-217. doi:10.2337/dc16-0826

Gong, P., Zhang, C., Lu, Z., Huang, J., &Ye, J. (2013). A General Iterative Shrinkage and Thresholding Algorithm for Non-convex Regularized Optimization Problems. Retrieved from https://arxiv.org/abs/1303.4434

Johnson, R., & Zhang, T., (2012). Accelerated Stochastic Gradient Descent using Predictive Variance Reduction. Retrieved from https://papers.nips.cc/paper/4937-accelerating-stochastic-gradient-descent-using-predictive-variance-reduction.pdf

Jin, C., & Jordan, I., M. (2017). Accelerated Gradient Descent Escapes Saddle Points Faster than Gradient Descent. Retrieved from https://arxiv.org/abs/1711.10456

Jin, C., Ge, R., Netrapalli, P., Kakade, S.M., & Jordan, M.I.. (2017) How To Escape Saddle Points Efficiently. *International Conference on Machine Learning (ICML)*. Retrieved from https://arxiv.org/abs/1703.00887

Kaggle, (n.d.). Retrieved from https://www.kaggle.com/c/santander-value-prediction-challenge

Kawaguchi, K., (2016) Deep learning without Poor Local Minima. In Advances in Neural Information Processing Systems, pages 586–594

Momentum and learning Rate Adaption.(n.d.). Retrieved from https://www.willamette.edu/~gorr/classes/cs449/momrate.html

Nesterov, Y., (1983) A method of solving a convex programming problem with convergence rate O (1/k2). Retrieved from http://mpawankumar.info/teaching/cdt-big-data/nesterov83.pdf

Nesterov. Y., (1998) Introductory Lectures on Convex Programming Volume I: Basic course. Springer

Stochastic gradient descent. (2018). Retrieved from https://en.wikipedia.org/wiki/Stochastic_gradient_descent

Academic Vita of Cai Fei (ckf5081@psu.edu)

EDUCATION

The Pennsylvania State University – Honors in Bachelor of Science in Industrial Engineering

RELEVANT EXPERIENCE

Summer Associate, Management Consulting, Customers & Operations **May – July 2018**
KPMG Shanghai, China

- Conducted spend analysis, category opportunities identification, and should cost analysis on a cost reduction project for a global leading commercial HVAC company; leveraged core methodologies to achieve full spend transparency in both Asia direct and indirect procurement, with approximate $300M and $100M spend volume
- Identified more than 50 cost saving initiatives with a team of consultants, resulting in a potential saving of $15M for Asia direct procurement
- Researched diesel benchmark pricing and contracting models; developed a negotiation strategy which led to annual saving of HKD 2.2M for a global waste management company

Manufacturing Engineering Co-op **Jan.– Aug. 2017**
Volvo Construction Equipment Shippensburg, PA

- Imported wheel loader assembly work instructions from PowerPoint to SAP for system transformation, which saved 50% of system maintenance time per product
- Supported the engineering group on continuous improvements tasks; reduced cycle time of an assembly zone by 14% through tooling design

Human Resource Technical Intern **Jun. – Aug. 2016**
Bank of China Head Office Beijing, China

- Programmed macros in Excel to manage working hours for more than 300 employees; analyzed and generated reports on the performance of each department in relationship with its working schedules

LEADERSHIP EXPERIENCE

Vice President, Institute of Industrial and Systems Engineers (IISE), PSU. **Jan. 2015 – May 2018**

- Performed administrative duties to facilitate the effective functioning of the club and its members; planned and organized logistics for chapter's attendance to annual IISE regional conference

Public Relation Coordinator, Chinese Students and Scholars Association. **Jan. 2015 – Mar. 2018**

- Negotiated with company representatives to acquire more than $26,000 of local and international sponsorships for various student events
- Enhanced international students' engagement and represented Chinese student group to communicate with school officials and other student organizations

Teaching Intern, College of Engineering, PSU **Aug. 2017 – Dec. 2017**

- Supported student learning with SolidWorks prototyping and metrology practice in IE305 - Product Design, Specification and Measurement;

HONORS /CERTIFICATES

Honors IISE Dwight D. Gardner Scholarship 2017-2018
Harold and Inge Marcus Undergraduate Scholarships 2017-2018
Julia and Brent Beabout Scholarship 2017
Life Technologies Honors Scholarship in Industrial Engineering 2017

Cert. Six Sigma Green Belt