ADAPTIVE STEP SIZE MOMENTUM METHOD FOR DECONVOLUTION

Trung Vu and Raviv Raich

School of EECS, Oregon State University, Corvallis, OR 97331-5501, USA

ABSTRACT

In this paper, we introduce an adaptive step size schedule that can significantly improve the convergence rate of momentum method for deconvolution applications. We provide analysis to show that the proposed method can asymptotically recover the optimal rate of convergence for first-order gradient methods applied to minimize smooth convex functions. In a convolution setting, we demonstrate that our adaptive schedule can be implemented efficiently without adding computational complexity to traditional gradient schemes.

Index Terms— Deconvolution, Convex Optimization, Momentum, Rate of Convergence

1. INTRODUCTION

Deconvolution is the process of reversing the effects of convolution [1]. It is widely used in the areas of signal processing and image processing [2, 3]. In image processing, this term also refers to recovering the original image by deblurring [4]. Recently there has been an increasing interest in machine learning approaches for deconvolution including nonnegative matrix factorization [5], sparse coding [6, 7], convolutional dictionary learning [8, 9].

Deconvolution is usually performed by representing the convolution in the form of a linear shift-invariant operator and utilize a minimum mean square error as an optimization criterion. From machine learning perspective, the objective function can also be extended to other loss functions like Hinge loss or logistic regression cost function. Deconvolution can be done on either the time domain using circulant matrices or the frequency domain by computing the Fourier Transform [10]. A major challenge of this inverse problem is the ill-posed nature of continuous data that results in ill-conditioned matrices in the optimization [11]. Several techniques have been proposed to accomplish this using regularization theory. One direct way is to compute the closed-form solution of the problem. However, this approach is often inefficient due to the computational complexity of the inverse operator, and more importantly, it only works for apparently simple objective functions [4]. A more common method is to use iterative algorithms, in which various optimization techniques can be exploited to find a close approximation of the solution. With the increasing number of large-scale problems, this approach have been shown to be very well suited to deconvolution. Besides, other deconvolution techniques include recursive filtering [11], wavelets [12], and neural networks [13].

The most widely used among iterative algorithms for deconvolution is gradient descent. Although this method suffers from the slow convergence rate of first-order methods, its low cost and simplicity turn out to be very useful in practice. On the other hand, second-order methods such as Newton-Raphson obtain a rapid convergence rate but require the computation of the Hessian and its inverse, which can be prohibitively expensive for large scale problems [14]. To compromise, momentum method has been proposed to accelerate the convergence of gradient descent while remaining the computational complexity. This slight modification of gradient descent was shown to achieve a fast convergence rate on minimizing a smooth convex function [15]. Nevertheless, while multiple approaches are available for choosing optimal step sizes in gradient descent (e.g., backtracking line search), little is known for step size selection in momentum method when prior knowledge of the function curvature is limited.

To address this issue, we propose an adaptive schedule that uses the gradient information to compute the step size for momentum method at each iteration accordingly. In a convolution setting, the special structure of the objective function allows us to implement the algorithm efficiently without heavy computations of the Hessian. We provide analysis to show that our method asymptotically recovers the optimal convergence rate determined by the Hessian at the solution. Compared to gradient descent methods, the proposed method requires only twice as many as the number of operations per iteration, while dramatically accelerates the convergence in many cases when the objective function is ill-conditioned in general but locally well-conditioned at the solution. Lastly, we present a numerical evaluation that verifies the effectiveness of the proposed approach and suggest potential applications to other domains.

2. PRELIMINARY

Consider the problem of minimizing a twice differentiable, smooth and strongly convex function $f(x) : \mathbb{R}^d \to \mathbb{R}$. In particular, $lI \preceq \nabla^2 f(x) \preceq LI$, $\forall x$. We shall denote by x^*

This work is partially supported by the National Science Foundation grants CCF-1254218 and DBI-1356792.

the unique solution of this optimization problem and $f^* = f(x^*)$. We further assume that λ_1 and λ_d are the largest and smallest eigenvalues of the Hessian at x^*, respectively. Thus, we define the global condition number of f as $r = \frac{L}{l}$, and the local condition number of $\nabla^2 f^*$ as $\kappa = \frac{\lambda_1}{\lambda_d}$. For quadratics, these two number are the same. However, for non-quadratic functions, κ is smaller than r. Exploiting the gap between k and r is often an efficient way to accelerate the convergence of iterative methods.

In gradient descent method, the solution is initialized to $x = x^{(0)}$ and the following step is used to update x:

$$x^{(k+1)} = x^{(k)} - \alpha^{(k)} \nabla f(x^{(k)}). \quad (1)$$

Various algorithms have been proposed to choose the step size $\alpha^{(k)}$ in order to obtain an optimal convergence rate. One notable result from Nesterov regarding fixed step size gradient descent methods is given by Theorem 2.1.

Theorem 2.1. *(Theorem 2.1.15 in [16]). The gradient descent method with fixed step size $\alpha^{(k)} = \frac{2}{L+l}$ has a global linear convergence rate of $R = \frac{r-1}{r+1} = \frac{L-l}{L+l}$, i.e.,*

$$f(x^{(k+1)}) - f^* \leq \frac{L}{2}\left(\frac{L-l}{L+l}\right)^{2k} \left\| x^{(0)} - x^* \right\|^2.$$

Alternatively, adaptive schedules like exact and inexact line search are generally preferred in practice. It has recently been shown to converge at the same rate R for smooth convex functions on the worst-case scenario [17]. However, beyond the worst-case scenario, we should note that the asymptotic convergence rate is generally better for non-quadratic functions, where the objective function is locally well-conditioned, and the asymptotic convergence rate is defined by the local condition number of the Hessian at the solution: $K = \frac{\kappa-1}{\kappa+1}$, which is smaller than R.

Momentum method adds a second term from the previous iterate to the update equation of gradient descent

$$x^{(k+1)} = x^{(k)} - \alpha^{(k)} \nabla f(x^{(k)}) + \beta^{(k)}(x^{(k)} - x^{(k-1)}). \quad (2)$$

In [18], Polyak showed that this method achieves a faster convergence rate of $\frac{\sqrt{r}-1}{\sqrt{r}+1}$ on a quadratic by setting

$$\alpha^{(k)} = \Big(\frac{2}{\sqrt{L}+\sqrt{l}}\Big)^2, \beta^{(k)} = \Big(\frac{\sqrt{L}-\sqrt{l}}{\sqrt{L}+\sqrt{l}}\Big)^2. \quad (3)$$

It is noteworthy that analyses of convergence in this case usually involve performing a change of basis on the domain size $y^{(k)} = U^T(x^{(k)} - x^*)$, where U comes from the eigenvalue decomposition $\nabla^2 f^* = U \Lambda U^T$. The convergence rate is then defined by the slowest decreasing component in $y^{(k)}$, denoted by $y_j^{(k)}$. Similar to gradient descent, fixing the momentum step size does not recover the optimal convergence rate for non-quadratic smooth convex objective function. An in-depth analysis on different momentum regimes which is based on the behavior of second-order dynamic systems is discussed in [19]. The authors also suggested a restart strategy in order to achieve an even faster convergence rate that depends on the condition number of Hessian at solution, $\tau = \frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}$. However, the proposed algorithms are based on Nesterov's Accelerated Gradient method, a variant of momentum method, and hence its motivation is rather the restarting mechanism than choosing the optimal step sizes.

3. PROBLEM FORMULATION

In deconvolution, we are given a training set of $\{\boldsymbol{x}_m, \boldsymbol{y}_m\}_{m=1}^M$ and the objective is to learn a convolution kernel $\boldsymbol{w}$ to minimize a cost function $f(\boldsymbol{w}) = \sum_{m=1}^M \mathcal{C}(\boldsymbol{x}_m * \boldsymbol{w}, \boldsymbol{y}_m) + \Omega(\boldsymbol{w})$, where Ω is a regularization term. Assume all training examples are of the same dimension n and are zero-padded at both ends, we can denote $\boldsymbol{x}_m = [x_m(1), \ldots, x_m(n)]^T$, $\boldsymbol{y}_m = [\boldsymbol{y}_m(1), \ldots, \boldsymbol{y}_m(n)]^T$, and $\boldsymbol{w} = [w(1), \ldots, w(h)]^T$, where h is the window size. Let $\boldsymbol{x}_{mt} = [x_m(t), x_m(t-1), \ldots, x_m(t-h+1)]^T$ be the tth sliding window segment in the mth signal. If $\mathcal{C}$ can be broken down to each sliding window, the objective function is rewritten as

$$f(\boldsymbol{w}) = \sum_{m=1}^M \sum_{t=1}^n c(\boldsymbol{w}^T \boldsymbol{x}_{mt}, \boldsymbol{y}_m(t)) + \Omega(\boldsymbol{w}). \quad (4)$$

We make a further assumption that the cost function $c(\boldsymbol{a}, \boldsymbol{b})$ is smooth convex and the regularizer $\Omega(\boldsymbol{w})$ is smooth and strongly convex, and both are twice differentiable with respect to $\boldsymbol{a}$ and $\boldsymbol{w}$, i.e., $0 \preceq \frac{\partial^2 c}{\partial \boldsymbol{a} \partial \boldsymbol{a}^T} \preceq \mu I$ and $\lambda I \preceq \frac{d^2 \Omega}{d\boldsymbol{w} d\boldsymbol{w}^T} \preceq \gamma I$, for $\mu, \lambda, \gamma > 0$. Note that $c(\boldsymbol{a}, \boldsymbol{b})$ can be a distance metric (e.g., $\|\boldsymbol{a} - \boldsymbol{b}\|^2$), a divergence metric (e.g., $\sum_i \big(b_i \log \frac{b_i}{a_i} + a_i - b_i\big)$ where $a_i, b_i > 0$) or a more general loss (e.g., $\log(\sum_i e^{a_i}) - \sum_i a_i b_i$ where $b_i \in \{0,1\}$). Consider a logistic loss with L2-regularization for example, $c(a,b) = \log(1 + e^a) - ab$, where $b \in \{0,1\}$, and $\Omega(\boldsymbol{w}) = \|\boldsymbol{w}\|^2$, the aforementioned assumption is supported by $0 \leq \frac{\partial^2 c}{\partial a^2} = \frac{\partial c}{\partial a}(1 - \frac{\partial c}{\partial a}) \leq \frac{1}{4}$ and $\frac{d^2 \Omega}{d\boldsymbol{w} d\boldsymbol{w}^T} = \lambda I$. For simplicity, we provide analysis for the case where the cost function parameters $\boldsymbol{a}, \boldsymbol{b}$ are scalars. From (4), we obtain

$$\nabla^2 f(\boldsymbol{w}) = \sum_{m=1}^M \sum_{t=1}^n \frac{\partial^2 c}{\partial (\boldsymbol{w}^T \boldsymbol{x}_{mt})^2} \boldsymbol{x}_{mt} \boldsymbol{x}_{mt}^T + \frac{d^2 \Omega}{d\boldsymbol{w} d\boldsymbol{w}^T}. \quad (5)$$

In this setting, the special structure of the Hessian recalls the autocorrelation $R_{\hat{x}_m} = X_m^T X_m$, where the circulant matrix $X_m = [\boldsymbol{x}_{m1}^T, \ldots, \boldsymbol{x}_{m(n+h-1)}^T]$ is obtained from padding zero to $\boldsymbol{x}_m$ and $\hat{\boldsymbol{x}}_m$ is the time-reversed version of $\boldsymbol{x}_m$. If all signals are normalized to zero mean and unit variance, the Hessian can be bounded by

$$\lambda I \preceq \nabla^2 f(\boldsymbol{w}) \preceq \mu \sum_{m=1}^M R_{\hat{x}_m} + \gamma I \qquad \forall \boldsymbol{w}. \quad (6)$$

Since the power spectrum of $\hat{x}_m$ can be expressed as the Fourier Transform of its autocorrelation function, the maximum eigenvalue of $R_{\hat{x}_m}$ is also the maximum power spectrum $S_{\hat{x}_m}(0)$. Thus, (6) provides us with a decent estimate of the function parameters: $l = \lambda$ and $L = \mu \sum_{m=1}^{M} S_{\hat{x}_m}(0) + \lambda$. In this estimation, the lower bound depends on the choice of the regularization factor, while the upper bound depends on the data itself.

4. ADAPTIVE STEP SIZE SCHEDULE

Motivated by line search approach in gradient descent, this section presents an adaptive schedule to choose the optimal value of step size for each momentum iteration. First, we notice that the optimal convergence rate for momentum, and in general for first-order methods, depends on the condition number of the Hessian at the solution. Indeed, asymptotic analyses of convergence often assume the function can be locally approximated by a quadratic in the region near the optimum, and consider the rate of convergence inside this region. In case of gradient descent, recovering the optimal rate is practically done by backtracking line search. Another inexact line search method stems from second-order Taylor expansion of the objective function $f(\boldsymbol{w}-\alpha\nabla f(\boldsymbol{w})) \approx f(\boldsymbol{w}) - \alpha\nabla f(\boldsymbol{w})^T\nabla f(\boldsymbol{w}) + \frac{1}{2}\alpha^2\nabla f(\boldsymbol{w})^T\nabla^2 f(\boldsymbol{w})\nabla f(\boldsymbol{w})$. The superscripts are omitted for brevity. The step size at each iteration is then determined by minimizing this quadratic function with respect to α, yielding

$$\alpha = \frac{\nabla f(\boldsymbol{w})^T\nabla f(\boldsymbol{w})}{\nabla f(\boldsymbol{w})^T\nabla^2 f(\boldsymbol{w})\nabla f(\boldsymbol{w})}. \tag{7}$$

Since quadratic functions have the same Hessian everywhere, it follows from Section 2 that the resulting iterates obtain the optimal asymptotic convergence rate K inside the quadratic region near the solution.

Naturally, we bring this intuition to the updates in momentum method. Let $\Delta\boldsymbol{w}^{(k)} = \boldsymbol{w}^{(k)} - \boldsymbol{w}^{(k-1)}$, $D_k = [\nabla f(\boldsymbol{w}^{(k)}), -\Delta\boldsymbol{w}^{(k)}]$, and $\boldsymbol{\eta}^{(k)} = [\alpha^{(k)}, \beta^{(k)}]^T$. From (2), we can approximate $f(\boldsymbol{w}^{(k)} - D_k\boldsymbol{\eta}^{(k)}) \approx f(\boldsymbol{w}^k) - \nabla f(\boldsymbol{w}^{(k)})^T D_k\boldsymbol{\eta}^{(k)} + \frac{1}{2}{\boldsymbol{\eta}^{(k)}}^T D_k^T\nabla^2 f(\boldsymbol{w}^{(k)})D_k\boldsymbol{\eta}^{(k)}$. Minimizing this quadratic function with respect to $\boldsymbol{\eta}^{(k)}$ yields

$$\boldsymbol{\eta}^{(k)} = \left(D_k^T\nabla^2 f(\boldsymbol{w}^{(k)})D_k\right)^{-1} D_k^T\nabla f(\boldsymbol{w}^{(k)}). \tag{8}$$

We can further simplify (8) as follows

$$\begin{bmatrix}\alpha\\ \beta\end{bmatrix} = \begin{bmatrix}\nabla f^T\nabla^2 f\nabla f & -\Delta\boldsymbol{w}^T\nabla^2 f\nabla f\\ -\Delta\boldsymbol{w}^T\nabla^2 f\nabla f & \Delta\boldsymbol{w}^T\nabla^2 f\Delta\boldsymbol{w}\end{bmatrix}^{-1}\begin{bmatrix}\nabla f^T\nabla f\\ -\Delta\boldsymbol{w}^T\nabla f\end{bmatrix}$$

Note that the inversion in this equation only involves a 2×2 matrix, and should not be confused with the inversion in Newton-Raphson method. More interestingly, computing this matrix only requires the same complexity as computing the gradient thanks to the decomposition of $\nabla^2 f$ into multiple terms of the form $\boldsymbol{x}\boldsymbol{x}^T$. We propose the adaptive step size momentum method in Algorithm 1, with L2-regularization for simplicity.

Algorithm 1: Adaptive step size schedule for momentum.

```
 1: Given initial guess $\boldsymbol{w}^{(0)}$ and $\boldsymbol{w}^{(1)}$.
 2: repeat for $k = 1, 2, \ldots$
 3:     $\Delta\boldsymbol{w} = \boldsymbol{w}^{(k)} - \boldsymbol{w}^{(k-1)}$                                  ▷ $O(h)$
 4:     $\nabla f = \sum_{m,t}\frac{\partial c}{\partial(\boldsymbol{w}^T\boldsymbol{x}_{mt})}\boldsymbol{x}_{mt} + \lambda\boldsymbol{w}$            ▷ $O(Mnh)$
 5:     for $m = 1, \ldots, M, t = 1, \ldots, n$ do                ▷ $O(Mnh)$
 6:         $p_{mt} = \boldsymbol{x}_{mt}^T\nabla f$, $q_{mt} = \boldsymbol{x}_{mt}^T\Delta\boldsymbol{w}$
 7:         $c_{mt} = \partial^2 c/\partial(\boldsymbol{w}^T\boldsymbol{x}_{mt})^2$
 8:     $u = \nabla f^T\nabla f$, $v = \Delta\boldsymbol{w}^T\nabla f$, $t = \Delta\boldsymbol{w}^T\Delta\boldsymbol{w}$   ▷ $O(h)$
 9:     $a = \sum_{m=1}^{M}\sum_{t=1}^{n} c_{mt}p_{mt}^2 + \lambda u$                  ▷ $O(Mn)$
10:     $b = \sum_{m=1}^{M}\sum_{t=1}^{n} c_{mt}p_{mt}q_{mt} + \lambda v$               ▷ $O(Mn)$
11:     $d = \sum_{m=1}^{M}\sum_{t=1}^{n} c_{mt}q_{mt}^2 + \lambda t$                  ▷ $O(Mn)$
12:     $\alpha^{(k)} = \frac{du-bv}{ad-b^2}$, $\beta^{(k)} = \frac{bu-av}{ad-b^2}$                         ▷ $O(1)$
13:     Update $\boldsymbol{w}^{(k+1)}$ using (2).                          ▷ $O(h)$
14: until convergence
```

Table 1: Computational complexity of fixed step size gradient (GD), adaptive step size gradient (AGD), fixed step size momentum (MO), adaptive step size momentum (AMO), and Newton's method. ϵ is the relative accuracy.

Method	# Ops. / Iter.	Cvg. rate	# Iters. needed
GD	$O(Mnh)$	$\frac{r-1}{r+1}$	$\frac{r+1}{2}\log(1/\epsilon)$
AGD	$O(Mnh)$	$\frac{\kappa-1}{\kappa+1}$	$\frac{\kappa+1}{2}\log(1/\epsilon)$
MO	$O(Mnh)$	$\frac{\sqrt{r}-1}{\sqrt{r}+1}$	$\frac{\sqrt{r}+1}{2}\log(1/\epsilon)$
AMO	$O(Mnh)$	$\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}$	$\frac{\sqrt{\kappa}+1}{2}\log(1/\epsilon)$
Newton	$O(Mnh^3)$	quadratic	$O(1)$

The resulting iterates obtain a provably asymptotic convergence rate $\tau = \frac{\sqrt{\kappa}-1}{\sqrt{\kappa}-1}$. The detailed proof is not given here due to space limitation. The intuition is each iteration Algorithm 1 decreases the objective function more than that of fixed step size momentum chosen by (3). Therefore, inside the optimal region, it converges at least as fast as fixed step size chosen by the local parameters. Although the behavior outside the optimal region is unclear, this adaptive schedule is often helpful in practice.

To simplify the complexity analysis, we assume the calculation of derivatives of c and Ω is $O(1)$. Table 1 shows the computational complexity per iteration of the proposed approach is the same as other methods, but it requires the least number of iterations to reach a certain accuracy to the solution. The computation can even be more efficient if the window size is large enough ($h \approx n$), by using the Fast Fourier Transform to obtain $O(Mn\log n)$ complexity per iteration.

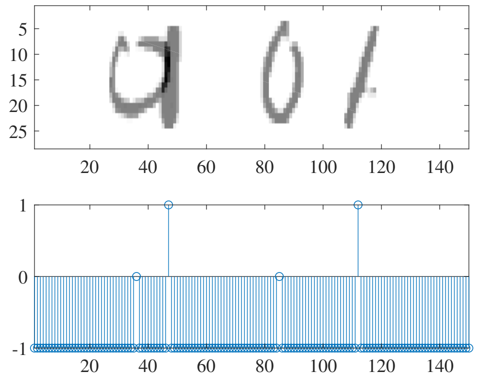

Fig. 1: Top - an image generated by randomly inserting a sequence of $0, 1, 0, 1$. Bottom - the corresponding label series.

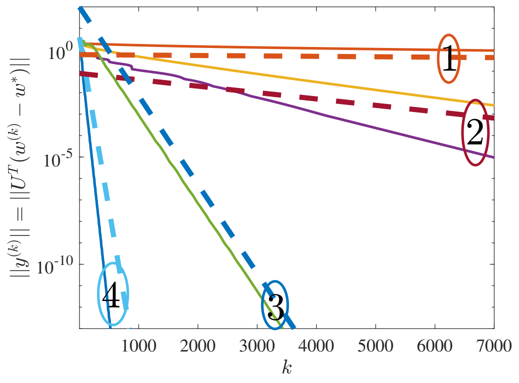

Fig. 2: The log-scale decrease in the distance to the solution on domain value side through iterations.

5. NUMERICAL EXAMPLE

In this experiment, we consider a convolutive logistic regression model for recognizing a sequence of handwritten digits. Our goal is to illustrate the convergence of the proposed algorithm and compare it with the theoretical analysis in the previous section.

Setting. From MNIST database, we generate a dataset of $M = 10$ composite images as follows. Each image of size 28×150 is created by sequentially adding four 28×28 digit images to a zero background such that the bottom left corner of the ith digit is chosen uniformly between positions $28(i-1)+1$ and $28i$ along the width of the composite image (see Fig. 1). We create the feature vector $\boldsymbol{x}_{mt}$ of $h = 784$ elements by vectorizing the 28×28 window centered at tth position along the width of the mth image ($n = 150$). For the purpose of illustration, we only consider images with two digits 0 and 1. Thus, we aim to learn a classifier $\boldsymbol{w}$ for $C = 3$ classes 0, 1 and -1 (for non-digit positions). The parameter $\boldsymbol{w}$ thereupon has 2352 elements ($= 3 \times 28 \times 28$), making the Hessian exceedingly large and infeasible to apply Newton's method. Finally, the label $\boldsymbol{y}_m(t)$ is determined by checking whether the window centered at tth position matches exactly with the digit positions. We represent the label as a vector of class membership $\boldsymbol{y}_m(t) = [y_{mt1}, \ldots, y_{mtC}]^T$.

Recall the multinomial logit-model is given by $p_{mtc} = P(y_{mtc}|\boldsymbol{x}_{mt}, \boldsymbol{w}) = e^{y_{mtc}\boldsymbol{w}_c^T\boldsymbol{x}_{mt}}/(\sum_{j=1}^{C} e^{\boldsymbol{w}_j^T\boldsymbol{x}_{mt}})$ and the cost $c(\boldsymbol{a}, \boldsymbol{b}) = \log(\sum_{c=1}^{C} e^{a_c}) - \sum_{c=1}^{C} a_c b_c$. The Hessian can be extended from (5) as $\nabla^2 f = \frac{1}{Mn}\sum_{m,n}(\Lambda_{\boldsymbol{p}_{mt}} - \boldsymbol{p}_{mt}\boldsymbol{p}_{mt}^T) \otimes \boldsymbol{x}_{mt}\boldsymbol{x}_{mt}^T$, where $\boldsymbol{p}_{mt} = [p_{mt1}, \ldots, p_{mtC}]^T$ (see [20]). Since we have $\Lambda_{\boldsymbol{p}_{mt}} - \boldsymbol{p}_{mt}\boldsymbol{p}_{mt}^T \preceq \frac{1}{2}(I_C - \frac{\mathbf{1}\mathbf{1}^T}{C+1})$, a rough estimate of the Lipschitz constant is $L = \frac{1}{2Mn}\sum_m S_{\hat{x}_m}(0)$. The strong convexity constant l is controlled by the L2-regularization factor $\lambda = 10^{-2}$. Thereupon, we implement four other methods in comparison with our proposed method, namely fixed step size gradient descent, adaptive step size gradient descent, fixed step size momentum, and gradient descent with backtracking line search. For fixed step size schemes, we use the optimal step sizes described in Theorem 2.1 and Equation (3). For backtracking line search, we set the parameters $\alpha = 0.2, \beta = 0.5$. Our adaptive step size schedules require no tuning parameters.

Results and analysis. Figure 2 compares the empirical convergence of the five methods in terms of the distance to the solution. The dash lines are added to the plot in order to depict the theoretical convergence rate corresponding to the global and local condition numbers given in Table 1. Not surprisingly all the methods match their theoretical convergence rates. The convergence of adaptive step size momentum deems to be slightly faster than the optimal rate at the solution (group 4), and clearly outperforms the other four methods. Adaptive schedule applied to gradient descent also results in a competitive convergence to backtracking line search, i.e., purple line versus yellow line (group 2). Fixed step size gradient descent and momentum method converge almost at the rate predicted by the analysis (group 1 and 3). Obviously, those approaches are slower than their adaptive versions because they only depend on the global parameters of the objective function and cannot recover the optimal rate at the solution. For quadratic objectives, this distinction is occluded by the fact that the Hessian is constant everywhere.

6. CONCLUSION

To conclude, we proposed an adaptive schedule for choosing step size in momentum method, under deconvolution settings. It can be readily implemented without adding computational complexity to fixed step size schemes. We showed that our method outperforms other aforementioned iterative methods in terms of convergence rate. It is promising that the proposed approach can be applied to a wide range of problems in the domain of digital signal and image processing.

Acknowledgement We would like to thank anonymous reviewers for their valuable comments.

7. REFERENCES

[1] N. Wiener, *Extrapolation, Interpolation, and Smoothing of Stationary Time Series*, The MIT Press, 1964.

[2] W. E. Blass and G. W. Halsey, *Deconvolution of absorption spectra*, Academic Press, 1981.

[3] M. R. Banham and A. K. Katsaggelos, "Digital image restoration," *IEEE Signal Processing Magazine*, vol. 14, no. 2, pp. 24–41, 1997.

[4] H. C. Andrews and B. R. Hunt, *Digital image restoration*, Englewood Cliffs, N.J. : Prentice-Hall, 1977.

[5] N. Mohammadiha, P. Smaragdis, and A. Leijon, "Supervised and unsupervised speech enhancement using nonnegative matrix factorization," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 10, pp. 2140–2151, 2013.

[6] H. Lee, A. Battle, R. Raina, and Andrew Y. Ng, "Efficient sparse coding algorithms," in *Proceedings of the 19th International Conference on Neural Information Processing Systems*, Cambridge, MA, USA, 2006, pp. 801–808, MIT Press.

[7] M. Šorel and F. Šroubek, "Fast convolutional sparse coding using matrix inversion lemma," *Digit. Signal Process.*, vol. 55, no. C, pp. 44–51, 2016.

[8] R. Chalasani, J. C. Principe, and N. Ramakrishnan, "A fast proximal method for convolutional sparse coding," in *The 2013 International Joint Conference on Neural Networks*, 2013, pp. 1–5.

[9] Z. You, R. Raich, X. Z. Fern, and J. Kim, "Discriminative recurring signal detection and localization," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2017, pp. 2377–2381.

[10] B. Hunt, "A matrix theory proof of the discrete convolution theorem," *IEEE Transactions on Audio and Electroacoustics*, vol. 19, no. 4, pp. 285–288, 1971.

[11] M. Nashed, "Operator-theoretic and computational approaches to ill-posed problems with applications to antenna theory," *IEEE Transactions on Antennas and Propagation*, vol. 29, no. 2, pp. 220–231, 1981.

[12] M. A. T. Figueiredo and R. D. Nowak, "A bound optimization approach to wavelet-based image deconvolution," in *IEEE International Conference on Image Processing 2005*, 2005, vol. 2, pp. II–782–5.

[13] L. Xu, J. Ren, C. Liu, and J. Jia, "Deep convolutional neural network for image deconvolution," in *Advances in Neural Information Processing Systems 27*, pp. 1790–1798. Curran Associates, Inc., 2014.

[14] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, New York, NY, USA, 2004.

[15] Y. Nesterov, "A method of solving a convex programming problem with convergence rate o(1/sqr(k))," *Soviet Mathematics Doklady*, vol. 27, no. 1, pp. 372–376, 1983.

[16] Y. Nesterov, *Introductory lectures on convex optimization: a basic course*, Kluwer Academic Publishers, 2004.

[17] E. Klerk, F. Glineur, and A. Taylor, "On the worst-case complexity of the gradient method with exact line search for smooth strongly convex functions," 2016.

[18] B. Polyak, "Some methods of speeding up the convergence of iteration methods," in *Ussr Computational Mathematics and Mathematical Physics*, 1964, vol. 4, pp. 1–17.

[19] B. O'Donoghue and E. Candès, "Adaptive restart for accelerated gradient schemes," *Found. Comput. Math.*, vol. 15, no. 3, pp. 715–732, 2015.

[20] Dankmar Böhning, "Multinomial logistic regression algorithm," *Annals of the Institute of Statistical Mathematics*, vol. 44, no. 1, pp. 197–200, 1992.