From Bound Majorization to Stochastic Bound Majorization

Yunian Pan
Department of Electrical Engineering
yp1170@nyu.edu

Abstract

Based on the bound majorization method for optimizing a partition function of log-linear models which uses a quadratic variational upper-bound to approximate the log-linear normalizer, outperforming state-of-the-art first- and second-order optimization methods on various learning tasks, a stochastic setting version of update was proposed in previous paper. The resulting stochastic second-order method outperforms stochastic gradient descent (across variations and various tunings) both in terms of the number of iterations and computation time till convergence while finding a better quality parameter setting. This report provides an empirical evaluation of this algorithm based on a simple multiclass logistic regression framework, along with some additional proof of the convergence.

1 Introduction

Estimation of the probability density function over sets of random variables is central to many learning problems. The estimation often requires optimization of the likelihood of training datasets, which is the goal of logistic regression, CRF(conditional random fields) and many other log-linear models. This report will focus on a general partition function of such sort of log-linear models, discussing the majorization methods to optimize it. Then there will be some discussion about the convergence of both batch and stochastic setting. Finally a simple experiment of multiclass logistic regression based on the given optimization method will be presented.

The main contents of this report will be organized as 3 parts: Section 2 presents the theoretical development of the algorithm and the stochastic version of it; Section 3 will give some analysis on the algorithms' convergence; Section 4 will give some empirical evaluation and comparisons under a multiclass logistic regression example.

2 Theoretical development

2.1 partition function

We begin our discussion from multinomial logistic regression, where we assume that the data are case specific, i.e. each independent variable has a single value for each case. The multinomial logistic model also assumes that the dependent variable cannot be perfectly predicted from the independent variables for any case.

The idea behind it, as in many other statistical classification techniques, is to construct a linear predictor function that constructs a score from a set of weights that are linearly combined with the explanatory variables (features) of a given observation using a dot product:

$$Score(x_i, k) = \theta_k^\top \cdot_i$$

32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada.

where $x_i \in R^{d'}$ is the vector of explanatory variables describing observation i, $\theta_k \in R^{d'}$ is a vector of weights (or regression coefficients) corresponding to category $k \in \Omega$, and $Score(x_i, k)$ is the score associated with assigning observation i to category k. In discrete choice theory, where observations represent people and outcomes represent choices, the score is considered the utility associated with person i choosing outcome k. The final predicted outcome is the one with the highest score.

The difference between this type of setting and many other methods and algorithms such as SVM and perceptron etc. is the procedure for determining (training) the optimal coefficients and the way that the score is interpreted, the score can directly be converted to a probability value, indicating the probability of observation i choosing outcome k given the measured characteristics of the observation.

Then, we model the logarithm of the probability of seeing a given output using the linear predictor as well as an additional normalization factor, the logarithm of the partition function:

$$\begin{aligned}
&\ln \Pr(y_i | y_i = 1, x_i) = \theta_1^\top \cdot x_i - \ln Z \\
&\ln \Pr(y_i | y_i = 2, x_i) = \theta_2^\top \cdot x_i - \ln Z \\
&\cdots\cdots\cdots \\
&\ln \Pr(y_i | y_i = K, x_i) = \theta_K^\top \cdot x_i - \ln Z
\end{aligned}$$

We need an extra term Z that has nothing to do with y_i to ensure that $\sum_k h(y_i = k) \Pr(y_i | y_i = k, X_i) = 1$, (with the non-negative prior $h(y_i = k)$) which is exactly the partition function of a simple conditional random field, this results in a soft-max function $\Pr(y_i = n | x_i) = \frac{h(n) e^{\theta_n^\top x_i}}{\sum_k h(k) e^{\theta_k^\top x_i}}$.

In a case that we don't want a set of different weights, we measure $\theta_k^\top x_i$ as $\theta^\top \mathbf{f}_{x_i}(k)$, where $\mathbf{f} : \Omega \to R^d$ is an arbitary mapping from the space of outcome to a real valued vector, $\theta^\top \in R^d$ is the coefficient that does not depend on the outcome. Thus we get rid of the index of parameter vector and develop a partition function that take in value of the single parameter vector, which is of the following form:

$$Z_{x_i}(\theta) = \sum_{y \in \Omega} h(y) \exp(\theta^\top \mathbf{f}_{x_i}(y)) \tag{1}$$

For simplicity, the mapping we choose here is a filter that place the input x_i on the position that corresponds to the choosing category/outcome, therefore θ can be viewed as a stack of the k categories' parameter vectors, i.e. $\theta \in R^{kd'}$.

Note that the partition function also can arise naturally from the maximum entropy estimation or minimum relative entropy, or, KL divergence, $D(p \| h) = \sum_{y \in \Omega} p(y) \log \frac{p(y)}{h(y)}$, where we estimate the distance from $p(y)$ seeing the prior $h(y)$.

2.2 Upper bound of partition

The partition function $Z(\theta) = \sum_{y \in \Omega} h(y) \exp(\theta^\top \mathbf{f}(y))$ will be the center of this section's analysis. As the supplement of [1] provides, the first thing we do is reorder the outcome using a random mapping $\pi(\cdot) : \Omega \to \{1, \ldots, n\}$, s.t. $h(y) = h(\pi^{-1}(j)) = h_j$, $\mathbf{f}(y) = \mathbf{f}(\pi^{-1}(j)) = \mathbf{f}_j$ and this is for explicity of the notation. Then we introduce a parameter $\lambda = \theta - \tilde{\theta}$, and rewrite $Z(\theta) = \sum_{j=1}^n \alpha_j \exp(\lambda^\top \mathbf{f}_j)$, where $\alpha_j = h(j) \exp(\tilde{\theta}^\top \mathbf{f}_j)$. In order to construct the monotonicity we denote $Z_i(\theta) = \sum_{j=1}^i \alpha_j \exp(\lambda^\top \mathbf{f}_j)$, and a trivial bound holds for $i = 0$:

$$Z_0(\theta) = 0 \leq z_0 \exp(\frac{1}{2} \lambda^\top \mathbf{\Sigma_0} \lambda + \lambda^\top \mu_\mathbf{0}) \tag{2}$$

Here we set the parameters z_0, μ_0, Σ_0 all close or equal to 0, with $z_0 = 0^+$, $\mu_0 = \mathbf{0}$, $\Sigma_0 = z\mathbf{I}$ to make the bound tight. As we add another term $\alpha_1 \exp(\lambda^\top \mathbf{f_1})$, on both side of the above inequality, the bound still holds,

$$Z_1(\theta) \leq z_0 \exp(\frac{1}{2} \lambda^\top \mathbf{\Sigma_0} \lambda + \lambda^\top \mu_\mathbf{0}) + \alpha_\mathbf{1} \exp(\lambda^\top \mathbf{f_1}) \tag{3}$$

Basically the idea is to transform the RHS of the inequality above to quadratic form so that we can update the parameter Σ_i and μ_i to Σ_{i+1} and μ_{i+1} iteratively, ensuring that the final bound can be quadratic which is easy to optimize. For the convenience of observation we write the logarithm of **??**:

$$\begin{aligned}\log Z_1(\theta) &\leq \log z_0 + \log(\exp(\frac{1}{2}\lambda^\top \Sigma_0 \lambda + \lambda^\top \mu_0) + \frac{\alpha_1}{z_0}\exp(\lambda^\top \mathbf{f}_1)) \\ &= \log z_0 + \log(\exp(\frac{1}{2}\lambda^\top \Sigma_0 \lambda + \lambda^\top (\mu_0 - \mathbf{f}_1)) + \frac{\alpha_1}{z_0}) + \lambda^\top \mathbf{f}_1\end{aligned}$$

As we observed that it might be useful to seperate a term $\frac{1}{2}w^\top w = \frac{1}{2}(\mathbf{f}_1 - \mu_0)^\top \Sigma_0^{-1}(\mathbf{f}_1 - \mu_0)$ out of the log-exponential function such that the terms inside the logarithm can form a quadratic form $u^\top u = \frac{1}{2}(\mathbf{f}_1 - \mu_0)^\top \Sigma_0^{-1}(\mathbf{f}_1 - \mu_0) + \frac{1}{2}\lambda^\top \Sigma_0 \lambda + \lambda^\top \mu_0$, (there are still residual terms to dispose):

$$\begin{aligned}RHS &= \log z_0 + \lambda^\top \mathbf{f}_1 - \frac{1}{2}w^\top w + \log \exp \frac{1}{2}w^\top w \cdot (\exp(\frac{1}{2}\lambda^\top \Sigma_0 \lambda + \lambda^\top \mu_0) + \frac{\alpha_1}{z_0}) \\ &= \log z_0 + \lambda^\top \mathbf{f}_1 - \frac{1}{2}w^\top w + \log(\exp(\frac{1}{2}u^\top u) + \gamma)\end{aligned} \tag{4}$$

where $\gamma = \frac{\alpha_1}{z_0}\exp\frac{1}{2}w^\top w$. Thus we are able to apply the lemma in [3] to bound the term $\log\exp(\frac{1}{2}u^\top u + \gamma)$.

Lemma 1 *For all $u \in R^d$ and $v \in R^d$ and any $\gamma \geq 0$, the bound* $\log(\exp(\frac{1}{2}\left\|u\right\|^2) + \gamma) \leq$

$$\log(\exp(\frac{1}{2}\left\|v\right\|^2) + \gamma) + \frac{v^\top (u - v)}{1 + \gamma \exp(-\frac{1}{2}\left\|v\right\|^2)} + \frac{1}{2}(u - v)^\top (I + \Gamma v v^\top)(u - v)$$

holds when the scalar term $\Gamma = \frac{\tanh(\frac{1}{2}\log(\gamma\exp(-\frac{1}{2}\|v\|^2))}{2\log(\gamma\exp(-\frac{1}{2}\|v\|^2)}$, *equality is achieved when* $u = v$.

Here we omit the proof of **??**. Apply the condition it provides that tightens the bound, $u = v$, by choosing $v = \Sigma_0^{-\frac{1}{2}}(\mu_0 - \mathbf{f}_1)$,(after factorization, $u = \Sigma_0^{\frac{1}{2}}\lambda - \Sigma_0^{-\frac{1}{2}}(\mu_0 - \mathbf{f}_1)$, and the inequality **??** is achieved when $\lambda = 0$)

$$\begin{aligned}\log Z_1(\theta) \leq \log z_0 + \lambda^\top \mathbf{f}_1 - \frac{1}{2}(\mathbf{f}_1 - \mu_0)^\top \Sigma_0^{-1}(\mathbf{f}_1 - \mu_0) \\ + \log(\exp(\frac{1}{2}\left\|v\right\|^2) + \gamma) + \frac{v^\top (u - v)}{1 + \gamma \exp(-\frac{1}{2}\left\|v\right\|^2)} + \frac{1}{2}(u - v)^\top (I + \Gamma v v^\top)(u - v)\end{aligned}$$

Recall the previous attemption that transforming the RHS of the inequality to quadratic form:

$$Z_1(\theta) \leq z_1 \exp(\frac{1}{2}\lambda^\top \mathbf{\Sigma_1} \lambda + \lambda^\top \mu_\mathbf{1}) \tag{5}$$

Combined with **??**, using the method of undetermined coefficient, after some ugly algebra work we can solve for the equations and get the update:

$$\begin{aligned}z_1 &= z_0 + \alpha_1 \\ \mu_1 &= \mu_0 + \frac{\alpha_1}{z_0 + \alpha_1}(\mathbf{f}_1 - \mu_0) \\ \Sigma_1 &= \Sigma_0 + \frac{\tanh(\frac{1}{2}\log(\alpha_1/z_0)}{2\log(\alpha_1/z_0)}(\mathbf{f}_1 - \mu_0)(\mathbf{f}_1 - \mu_0)^\top\end{aligned}$$

As we apply the procedure above successively until n, (assuming the cardinality $|\Omega| = n$ is finite and Ω is enumerable), the desired quadratic bound is achieved by mathematical induction. Equipped with the update equation above, Algorithm **??** can be proposed to compute the bound's parameters.

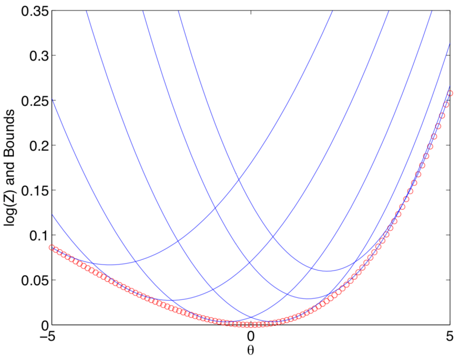


Figure 1: bounds

It's notoriously that the log-expsum function is convex and the lower bound is given by Jensen's inequality, this algorithm contributes an analogous quadratic upper-bound on the partition function, as **??** illustrated.

Algorithm 1: Compute Bound

```
**Input:** Parameters $\tilde{\theta}$, $\mathbf{f}(y)$, $h(y)$
**Initialize:** $z \leftarrow 0^+$, $\mu \leftarrow 0$, $\Sigma \leftarrow zI$;
**for** *each* $y \in \Omega$ **do**
    $\alpha = h(y)\exp(\tilde{\theta}^\top f(y))$
    $\mu = \mu + \frac{\alpha}{z+\alpha}(\mathbf{f}(y) - \mu)$
    $\Sigma = \Sigma + \frac{\tanh(\frac{1}{2}\log(\frac{\alpha}{z}))}{2\log(\frac{\alpha}{z})}(\mathbf{f}(y) - \mu)(\mathbf{f}(y) - \mu)^\top$
    $z = z + \alpha$
**end**
**Output:** $z$, $\mu$, $\Sigma$
```

2.3 bound majorization

Going back to the partition function, given a data-set $(x_1, y_1), \ldots, (x_t, y_t)$ of independent identically-distributed (i.i.d) input-output pairs where y_i is the observed sample in a (discrete) space Ω_i conditioned on the observed input x_i, A CRF, or, let's say multinomial logistic regression's parameter is estimated by maximizing the l_2 regularized loglikelihood of log-posterior: $\log\prod_{i=1}^{n}\Pr(y_i|x_i,\theta) - \frac{t\lambda}{2}\|\theta\|^2$, where $\lambda \in R^+$ is a regularizing paramter to prevent the weights from exploding to infinity, we can write the total loglikelihood as follow:

$$J(\theta) = \sum_{i=1}^{t}[\log\frac{h_{x_i}(y_i)}{Z_{x_i}(\theta)} + \theta^\top \mathbf{f}_{x_i}(y_i) - \frac{\lambda}{2}\|\theta\|^2] \tag{6}$$

If prior knowledge (or constraints) restrict the solution vector to a convex hull Λ, the maximization problem becomes $\arg\max_{\theta\in\Lambda} J(\theta)$, in practice we set $\Lambda \in R^d$. Recall the bound:

$$\begin{aligned} J(\theta) &\geq \sum_{i=1}^{t}[\log h_{x_i}(y_i) - \log z_i - \frac{1}{2}\lambda^\top \mathbf{\Sigma_i}\lambda + \lambda^\top \mu_{\mathbf{i}} + \theta^\top \mathbf{f_{x_i}}(\mathbf{y_i}) - \frac{\boldsymbol{\lambda}}{\mathbf{2}}\|\theta\|^{\mathbf{2}}] \\ &= \sum_{i=1}^{t}[\log h_{x_i}(y_i) - \log z_i - \frac{1}{2}(\theta - \tilde{\theta})^\top \Sigma_i (\theta - \tilde{\theta}) - (\theta - \tilde{\theta})^\top \mu_{\mathbf{i}} + \theta^\top \mathbf{f_{x_i}}(\mathbf{y_i}) - \frac{\lambda}{\mathbf{2}}\|\theta\|^{\mathbf{2}}] \end{aligned} \tag{7}$$

As we drop the terms unrelated to θ, the maximization problem becomes $\arg\min_\theta Q(\theta, \tilde{\theta})$, where:

$$\begin{aligned} Q(\theta,\tilde{\theta}) &= \sum_i \frac{1}{2}(\theta-\tilde{\theta})^\top \Sigma_i (\theta-\tilde{\theta}) + \theta^\top \mu_{\mathbf{i}} - \theta^\top \mathbf{f}_{\mathbf{x_i}}(\mathbf{y_i}) + \frac{\lambda}{\mathbf{2}}\theta^\top\theta + \tilde{\theta}^\top \mu_{\mathbf{i}}] \\ &= \sum_i \frac{1}{2}(\theta-\tilde{\theta})^\top \Sigma_i (\theta-\tilde{\theta}) + \theta^\top(\mu_{\mathbf{i}} - \mathbf{f}_{\mathbf{x_i}}(\mathbf{y_i})) + \frac{\lambda}{\mathbf{2}}(\theta-\tilde{\theta})^\top(\theta-\tilde{\theta}) + \lambda\theta^\top\tilde{\theta} - \frac{\lambda}{\mathbf{2}}\tilde{\theta}^\top\tilde{\theta} \\ &= \frac{1}{2}(\theta-\tilde{\theta})^\top(\sum_i(\Sigma_i+\lambda I))(\theta-\tilde{\theta}) + \sum_i \theta^\top(\mu_{\mathbf{i}} - \mathbf{f}_{\mathbf{x_i}}(\mathbf{y_i}) + \lambda\tilde{\theta}) - \mathbf{constant} \end{aligned} \tag{8}$$

Thus resulting Algorithm 2, which has the similiar structure with EM algorithm, by constructing convex upper bound of the dual objective iteratively, hopefully we can find the optimal solution of the original problem.

```
Algorithm 2: BM
Input: Input $x_i, y_i$ and functions $h_{x_i}, \mathbf{f}_{x_i}$ for $i = 1, 1, \ldots, t$, regularizer $\lambda \in R^+$ and convex hull
       $\Lambda \subseteq R^d$, $\epsilon$
Initialize: $\theta_0$ anywhere inside $\Lambda$ and set $\tilde{\theta} = \theta_0$ ;
while $\theta_{new} - \theta_{old} \geq \epsilon$ do
    for $i = 1, \ldots, t$ do
        Get $\mu_i,\ \Sigma_i$, from $h_{x_i},\ \mathbf{f}_{x_i}, \tilde{\theta}$ via Algorithm 1
    end
    Set $\tilde{\theta} = \arg\min_\theta \frac{1}{2}(\theta-\tilde{\theta})^\top(\sum_i \Sigma_i + \lambda I)(\theta-\tilde{\theta}) + \theta^\top(\sum_i \mu_i - \mathbf{f}_{x_i}(y_i) + \lambda\tilde{\theta})$
    Which means: $\tilde{\theta} = \tilde{\theta} - (\sum_i \Sigma + \lambda I)^{-1}(\sum_i \mu_i - \mathbf{f}_{x_i}(y_i) + \lambda\tilde{\theta})$
end
Output: $\hat{\theta} = \tilde{\theta}$
```

The proof of convergence under certain assumptions is given in [1] p.11, p.12, the basic idea is to construct an upper bound of the matrices Σ_i and compare $J(\theta)$ to $J(\theta^*)$ for every iteration, which will be discussed in the later section.

2.4 Stochastic bound majorization

The batch bound majorization algorithm assumes that we are in offline scenario where all the data are observable at once, [2] proposed a stochastic version of it, which can be applied to online scenario.

```
Algorithm 3: Stochastic Bound Majorization
Input: prior $h(\cdot)$, function $\mathbf{f}(\cdot)$, regularizer $\lambda \in R^+$ and convex hull $\Lambda \subseteq R^d$ $\epsilon$
Initialize: $\theta_0$ anywhere inside $\Lambda$ and set $\tilde{\theta} = \theta_0$ ;
while $\theta_{new} - \theta_{old} \geq \epsilon$ do
    randomly select $p$ mini-batch $x_i, y_i$'s
    for $i = 1, \ldots, p$ do
        Get $\mu_i,\ \Sigma_i$, from $h_{x_i},\ \mathbf{f}_{x_i}, \tilde{\theta}$ via Algorithm 1
    end
    Set $\tilde{\theta} = \arg\min_\theta \frac{1}{2}(\theta-\tilde{\theta})^\top(\sum_i \Sigma_i + \lambda I)(\theta-\tilde{\theta}) + \theta^\top(\sum_i \mu_i - \mathbf{f}_{x_i}(y_i) + \lambda\tilde{\theta})$
    Which means: $\tilde{\theta} = \tilde{\theta} - (\sum_i \Sigma + \lambda I)^{-1}(\sum_i \mu_i - \mathbf{f}_{x_i}(y_i) + \lambda\tilde{\theta})$
end
Output: $\hat{\theta} = \tilde{\theta}$
```

There are also a variation of this algorithm, as we notice that the update of θ can be viewed as a linear system:

$$(\sum_j \Sigma_j(\theta_{n-1}) + \lambda I)(\theta_n - \theta_{n-1}) = \sum_j \mu_j(\theta_{n-1}) \tag{9}$$

Where we can write $(\sum_j \Sigma_j(\theta_{n-1}) + \lambda I)^{-1}$ as M and apply the Sherman-Morrison formula $(\Sigma + (\sqrt{\beta}l)^\top(\sqrt{\beta}l))^{-1} = \Sigma^{-1} - \frac{\Sigma^{-1}(\sqrt{\beta}l)^\top(\sqrt{\beta}l)\Sigma^{-1}}{1+(\sqrt{\beta}l)^\top\Sigma^{-1}(\sqrt{\beta}l)}$, where the term $\beta = \frac{\tanh(\frac{1}{2}\log(\frac{\alpha}{z}))}{2\log(\frac{\alpha}{z})}$, $l = \mathbf{f}(y) - \mu$, thus rewrite the update formula of Σ_i as:

$$M_{n+1} = M_n - \frac{\beta M_n l^\intercal l M_n}{1 + \beta l^\intercal M_n l} \tag{10}$$

?? can be incorporated to the reformulation of θ update, specifically, during period of one-time update over a mini-batch dataset of size t we update the matrix M and μ $T = nt$ times, and we can expand $\phi = M_T \mu_T$ successively T times until $M_0,\ \mu_0$:

$$\begin{aligned}
\theta &= \theta - \eta M_T \mu_T \\
&= \theta - \eta(M_{T-1}\mu_{T-1} + M_T(\kappa_{T-1} l_{T-1} - \mathrm{f}_{T-1} + \frac{\lambda\theta}{t}) - \frac{\beta_{T-1} M_{T-1} l_{T-1}^\intercal l_{T-1} M_{T-1}}{1 + \beta_{T-1} l_{T-1}^\intercal M_{T-1} l_{T-1}} \mu_{T-1}) \\
&\ldots\ldots \\
&= \theta - \eta(M_0 \mu_0 + \sum_{i=0}^{T-1} M_i(\kappa_{i-1} l_{i-1} - \mathrm{f}_{i-1} + \frac{\lambda\theta}{t}) - \frac{\beta_{i-1} M_{i-1} l_{i-1}^\intercal l_{i-1} M_{i-1}}{1 + \beta_{i-1} l_{i-1}^\intercal M_{i-1} l_{i-1}} \mu_{i-1})
\end{aligned}$$

Leveraging the equation above, algorithm **??** was proposed as we can adjust the algorithm by terminating the summation before the θ is allowed to update, i.e. analogous to SGD, the most natural way to convert batch setting algorithm to mini-batch setting is to interleave updates of the parameter θ, the update rule will be given as:

$$\theta_{i+1} = \theta_i - \eta(\sum_{y=1}^{n} M_i(\theta_i)(\kappa_i(\theta_i) l_i(\theta_i) - \mathrm{f}_i(\theta_i) + \frac{\lambda\theta_i}{t}) - \frac{\beta_i M_i(\theta_i) l_i(\theta_i)^\intercal l_i(\theta_i) M_i(\theta_i)}{1 + \beta_i(\theta_i) l_{i-1}(\theta_i)^\intercal M_i(\theta_i) l_i(\theta_i)} \mu_i(\theta_i))$$

Note that the rule above is for one data point, the update rule also admits mini-batch. The algorithm, which performs a full-rank update of Σ, again can be adjusted to low-rank version.

Algorithm 4: SBM

```
Input: prior $h(\cdot)$, function $\mathbf{f}(\cdot)$, regularizer $\lambda \in R^+$ and convex hull $\Lambda \subseteq R^d$ , step size $\eta$, $\epsilon$
Initialize: $\theta_0$ anywhere inside $\Lambda$ and set $\tilde{\theta} = \theta_0$, $\phi = \mathbf{0}$, $M = \frac{1}{\lambda} I$, $\mu = 0$;
while $\theta_{new} - \theta_{old} \geq \epsilon$ do
    randomly select $p$ mini-batch $x_i, y_i$'s
    for $i = 1, \ldots, p$ do
        $z \leftarrow 0^+$; $g = 0$
        for each $y \in \Omega$ do
            $\alpha = h(y)\exp(\tilde{\theta}^\intercal f(y))$    $l = f(y) - g$    $\beta = \frac{\tanh(\frac{1}{2}\log(\frac{\alpha}{z})}{2\log(\frac{\alpha}{z})}$    $z = z + \alpha$    $\kappa = \frac{\alpha}{z}$
            $M = M - \frac{\beta M l^\intercal l M}{1+\beta l^\intercal M l}$
            $\phi = \phi + M(\kappa l - \mathrm{f}_{x_i}(y) + \frac{\lambda\tilde{\theta}}{t}) - \frac{\beta M l^\intercal l M}{1+\beta l^\intercal M l}\mu$
            $\mu = \mu + \kappa l - \mathrm{f}_{x_i}(y) + \frac{\lambda\tilde{\theta}}{t}$
            $g = g + \kappa l$
        end
    end
    $\tilde{\theta} = \tilde{\theta} - \eta\phi$
end
Output: $\hat{\theta} = \tilde{\theta}$
```

3 Discussion of the convergence

3.1 upper bound of batch setting convergence rate

We begin by comparing two functions' supremums, at each iteration there will be a slight change on matrix Σ,there's a lemma which would be useful for quantitatively measuring the difference of each iteration.

Lemma 2 *If $\kappa\Psi \geq \Phi$ for $\Psi \geq \Phi \in R^{d\times d}$, then*

$$L(\theta) = -\frac{1}{2}(\theta - \tilde{\theta})^\top \Phi(\theta - \tilde{\theta}) - (\theta - \tilde{\theta})^\top \mu$$
$$U(\theta) = -\frac{1}{2}(\theta - \tilde{\theta})^\top \Psi(\theta - \tilde{\theta}) - (\theta - \tilde{\theta})^\top \mu$$

satisfy $\sup_{\theta\in\Lambda} U(\theta)$ *for any convex* $\Lambda \subseteq R^d,\ \tilde{\theta} \in \Lambda,\ \mu \in R^d$ *and* $\kappa \in R^+$

The basic idea to prove this lemma is to find the relationship between the dual problems. Leveraging this lemma, first we upper-bound the matrix Σ_i by assuming $\|\mathbf{f}_{x_i}(y)\|^2 \leq r^2 \ \ \forall y \in \Omega$ and since μ_i is a convex combination of $\mathbf{f}_{x_i}(y)$, the outer-product terms in the update for Σ_i satisfy:

$$(\mathbf{f}_{x_i}(y) - \mu_i)(\mathbf{f}_{x_i}(y) - \mu_i)^\top \leq 4r^2 I$$

Thus $\Sigma_i \leq \mathcal{F}(\alpha_1, \ldots, \alpha_n)4r^2 I$ holds, where

$$\mathcal{F}(\alpha_1, \ldots, \alpha_n) = \sum_{i=2}^{n} \frac{\tanh(\frac{1}{2}\log(\frac{\alpha_i}{\sum_{k=1}^{i-1}\alpha_k}))}{2\log(\frac{\alpha_i}{\sum_{k=1}^{i-1}\alpha_k})}$$

$\alpha_1, \ldots, \alpha_n$ are defined in the algorithm 1, the formula for $\mathcal{F}$ starts at $i = 2$ since initialization. Assuming the permutation π is sampled uniformly at random, the expectation of $\mathcal{F}$ will be:

$$E_\pi[\mathcal{F}(\alpha_1, \ldots, \alpha_n)] = \frac{1}{n!}\sum_{\pi}\sum_{i=2}^{n} \frac{\tanh(\frac{1}{2}\log(\frac{\alpha_{\pi(i)}}{\sum_{k=1}^{i-1}\alpha_{\pi(k)}}))}{2\log(\frac{\alpha_{\pi(i)}}{\sum_{k=1}^{i-1}\alpha_{\pi(k)}})}$$

Analyzing this function can be daunting as the derivative is complicated, at the very first we guess that the expectation will be maximized when all α_i's are the same, then we observe that due to expectation, $\frac{\partial E}{\partial \alpha_i} = \frac{\partial E}{\partial \alpha_j} \ \ \forall i, j$, and $\mathcal{F}$ is invariant to any scaling of all α_i's, so the gradient $\nabla_\alpha E_\pi(\mathcal{F})_{|\alpha_i = constant}$ must be all 0, therefore any point $\alpha_i = constant$ will be either an extremum or a saddle point. Then we consider $\frac{\partial^2 \mathcal{F}}{\partial \alpha_i^2}$ and $\frac{\partial^2 \mathcal{F}}{\partial \alpha_i \partial \alpha_j}$,

$$\frac{\partial^2 \mathcal{F}}{\partial \alpha_i^2} = \frac{\partial^2 f(\frac{\alpha_i}{\alpha_{i-1}+\ldots})}{\partial \alpha_i^2} + \frac{\partial^2 f(\frac{\alpha_{i+1}}{\alpha_i+\ldots})}{\partial \alpha_i^2} + \ldots$$
$$\frac{\partial^2 \mathcal{F}}{\partial \alpha_i \partial \alpha_j} = \frac{\partial^2 f(\frac{\alpha_j}{\alpha_{j-1}+\ldots})}{\partial \alpha_i \partial \alpha_j} + \frac{\partial^2 f(\frac{\alpha_{j+1}}{\alpha_j+\ldots})}{\partial \alpha_i \partial \alpha_j} + \ldots (j > i)$$
$$\frac{\partial^2 \mathcal{F}}{\partial \alpha_i \partial \alpha_j} = \frac{\partial^2 f(\frac{\alpha_i}{\alpha_{i-1}+\ldots})}{\partial \alpha_i \partial \alpha_j} + \frac{\partial^2 f(\frac{\alpha_{i+1}}{\alpha_j+\ldots})}{\partial \alpha_i \partial \alpha_j} + \ldots (j < i)$$

where $f(\frac{\alpha_i}{\alpha_{i-1}+\ldots}) = \frac{\tanh(\frac{1}{2}\log(\frac{\alpha_{\pi(i)}}{\sum_{k=1}^{i-1}\alpha_{\pi(k)}}))}{2\log(\frac{\alpha_{\pi(i)}}{\sum_{k=1}^{i-1}\alpha_{\pi(k)}})}$. It's obvious to notice that $\frac{\partial^2 \mathcal{F}}{\partial \alpha_i \partial \alpha_j}$ is either of the same form with $\frac{\partial^2 \mathcal{F}}{\partial \alpha_i^2}$ or be part of it when $\alpha_i = \alpha_j$, therefore once the terms in $\frac{\partial^2 \mathcal{F}}{\partial \alpha_i^2}$ is determined, so is $\frac{\partial^2 \mathcal{F}}{\partial \alpha_i \partial \alpha_j}$. Essentially, $\frac{\partial^2 E}{\partial \alpha_i \partial \alpha_j} = -\frac{1}{n-1}\frac{\partial^2 E}{\partial \alpha_i^2}$ while $\frac{\partial^2 E}{\partial \alpha_i^2}$ is negative thus $\alpha_i = constant$ is locally concave and yeilds the Loewner ordering sense upper bound:

$$\Sigma_i \leq (2r^2 \sum_{i=2}^{n} \frac{\tanh(\frac{1}{2}\log i)}{\log i})I = \omega I \tag{11}$$

Equipped with this bound, it's natural to rewrite the $L(\theta)$ and $U(\theta)$ in **??** as:

$$\begin{aligned} J(\theta) &\geq J(\tilde{\theta}) - \frac{1}{2}(\theta - \tilde{\theta})^\top (\sum_{i}^{t}\Sigma_i + t\lambda I)(\theta - \tilde{\theta}) - (\theta - \tilde{\theta})^\top \mu \\ &= L(\theta) \\ J(\theta) &\leq J(\tilde{\theta}) - \frac{1}{2}(\theta - \tilde{\theta})^\top (t\lambda I)(\theta - \tilde{\theta}) - (\theta - \tilde{\theta})^\top \mu \\ &= U(\theta) \end{aligned} \tag{12}$$

At time τ after some iterations of traverse through the data, suppose we get the loglikelihood $J(\theta_\tau) = J(\tilde{\theta})$, it's clearly that it's bounded by lower bound $L_\tau(\theta)$ and upper bound $U_\tau(\theta)$ and only when $\theta = \theta_\tau$ the equality $L_\tau(\theta) = J_\tau(\theta) = U_\tau(\theta)$ holds, in the next epoch the procedure is to set $\theta_{\tau+1} = \arg\min_{\theta\in\Lambda} J_\tau(\theta)$, but in the context of lower bound where we replace Σ with ωI, it makes less progress when setting $\theta_{\tau+1} = \arg\min_{\theta\in\Lambda} L_\tau(\theta)$, which means $J(\theta_{\tau+1}) \geq \sup_{\theta\in\Lambda} L_\tau(\theta)$, setting $\Psi = t\lambda I$, $\Phi = (tw + t\lambda)I$ and $\kappa = \frac{w+\lambda}{\lambda}$, apply lemma **??**:

$$\sup_{\theta\in\Lambda} L_\tau(\theta) - L_\tau(\theta_\tau) = \frac{1}{\kappa} \sup_{\theta\in\Lambda} U_\tau(\theta) - U_\tau(\theta_\tau)$$

It's trivial that $J(\theta^*) \leq sup_{\theta\in\Lambda} U_\tau(\theta)$, thus:

$$\begin{aligned} J(\theta_{\tau+1}) - J(\theta_\tau) &\geq \frac{1}{\kappa}(J(\theta^*) - J(\theta_\tau)) \\ \Longrightarrow \quad J(\theta_{\tau+1}) - J(\theta^*) &\geq (1 - \frac{1}{\kappa})(J(\theta_\tau) - J(\theta^*)) \end{aligned}$$

successively,

$$J(\theta_\tau) - J(\theta^*) \geq (1 - \frac{1}{\kappa})^\tau (J(\theta_0) - J(\theta^*)) \tag{13}$$

Therefore if we want $J(\theta_\tau) - J(\theta_\tau) \leq \epsilon(J(\theta^*) - J(\theta_\tau))$ we need more than $\tau = \left[\frac{\log(\epsilon)}{\log(\kappa-1)-\log\kappa}\right]$ epochs of training. Note that here no guarantee of convergence is provided, **??** only provided a estimation of how far we have to go in order to obtain the desired optimal.

3.2 Convergence guarantee of batch bound majorization

Define mapping: $G(\theta) \coloneqq \theta - \eta V(\theta)$ where $V(\theta) = \Sigma^{-1}(\theta)\mu(\theta)$, our effort is equivalent to finding the solution to the fixed point equation: $\theta^* = G(\theta^*)$, which simply indicates: $\Sigma^{-1}(\theta^*)\mu(\theta^*) = 0$. (Following convexity we know that θ^* exists.)

Begin with an initialization θ_0 that is close enough to the optimal θ^*, $\|\theta_0 - \theta^*\|_2^2 \leq \frac{d}{2}$. Since $Q(\cdot|\theta)$ is a quadratic function that has very nice properties, one can pick some ϵ and l s.t $Q(\cdot|\theta)$ is both ϵ strongly convex and l smooth. and since the minimization of Q is within some convex hull Λ, or, let's say an Euclidean ball $B(\frac{d}{2}, \theta_0)$, thus all iterates reamain within an d-ball of θ^*.

Following a standard result from the optimization theory, we further propose lemma **??**:

Lemma 3 *Define a mapping $L(\theta) \coloneqq \theta - \eta V(\theta^*)$ which is equivalent to appling gradient operator $T(\theta) \coloneqq \theta - \eta\nabla Q(\theta|\theta^*)$ z_θ times, i.e. $L(\theta) = T^{z_\theta}(\theta)$, where z_θ is a finte integer, and $\nabla Q(\theta|\theta^*)$ is the gradient w.r.t population, under strong convexity condition and smoothness assumption which already hold with stepsize $\eta = \frac{2}{\epsilon+l}$, and because $T(\theta)$ is contractive, we have:*

$$\|L(\theta) - \theta^*\|_2 \leq (\frac{l-\epsilon}{l+\epsilon})^{z_\theta} \|\theta - \theta^*\|_2 \tag{14}$$

Here we leverage several truths as follows to recursively find the inequality:

- The standard result $\|T(\theta) - \theta^*\|_2 \leq (\frac{l-\epsilon}{l+\epsilon}) \|\theta - \theta^*\|_2$
- z_θ is the number of iteration that we perform to optimize a quadratic problem which is theoretically finite.
- $T^{z_\theta}(\theta_{z_\theta}) = TT^{z_\theta-1}(\theta_{z_\theta-1})$

Therefore we are able to focus on the update:

$$\begin{aligned} \|G(\theta) - \theta^*\|_2 &= \|\theta - \eta V(\theta) - \theta^*\|_2 \\ &\leq \|\theta - \eta V(\theta^*) - \theta^*\|_2 + \eta\|V(\theta) - V(\theta^*)\|_2 \\ &= \|L(\theta) - \theta^*\|_2 + \eta\|V(\theta) - V(\theta^*)\|_2 \end{aligned}$$

Notice that we have to make another assumption that the term $\|V(\theta) - V(\theta^*)\|$ is stablized by the distance $\|\theta - \theta^*\|_2$ in order to prove the convergence. Analogous to the gradient stability condition, it's natural to come up with the assumption **??**:

definition 1 *$V(\theta)$ stability*

The functions $\{Q(\cdot|\theta), \theta \in \Omega\}$ statisfy VS(γ) condition, where $\gamma \geq 0$, over Euclidean ball $B_2(d, \theta^*)$, if

$$\left\|\Sigma(\theta)^{-1}\mu(\theta) - \Sigma(\theta^*)^{-1}\mu(\theta^*)\right\|_2 \leq \gamma \|\theta - \theta^*\|_2 \tag{15}$$

for all $\theta \in B_2(d, \theta^*)$

Therefore, going back to the update rule:

$$\|G(\theta) - \theta^*\|_2 \leq ((\frac{l-\epsilon}{l+\epsilon})^{z(\theta)} + \eta\gamma) \|\theta - \theta^*\|_2 \tag{16}$$

Thus result in $\|\theta_n - \theta^*\|_2 \leq ((\frac{l-\epsilon}{l+\epsilon})^{z(\theta)} + \eta\gamma)^n \|\theta_0 - \theta^*\|$. Since $\frac{l-\epsilon}{l+\epsilon} > (\frac{l-\epsilon}{l+\epsilon})^{z_\theta}$, we only need to ponder whether $(\frac{l-\epsilon}{l+\epsilon}) + \eta\gamma < 1$ or not, which obviously holds when we have $\epsilon > \gamma$ (we've already set $\eta = \frac{2}{l+\epsilon}$), thus ensuring the term $(\frac{l-\epsilon}{l+\epsilon})^{z(\theta)} + \eta\gamma < 1$ along with the convergence of the update.

3.3 Convergence guarantee of stochastic bound majorization

Note that this is actually a Robbins Monro problem, where we define stepsize η_t of order $O(\frac{1}{n})$ to ensure that $\sum \eta_t = \infty$ and $\sum \eta_t^2 < \infty$, note that the updating rule in stochastic version becomes $G(\theta) := \theta - \eta\hat{V}(\theta)$, where $\hat{V}(\theta)$ is related to the randomly sampled mini-batch. At the very first beginning let's introduce several machineries. For a wider range of step sizes, lemma **??** holds that

Lemma 4 *Define a mapping $L(\theta) := \theta - \eta V(\theta^*)$ which is equivalent to appling gradient operator $T(\theta) := \theta - \eta\nabla Q(\theta|\theta^*)$ z_θ times, i.e. $L(\theta) = T^{z_\theta}(\theta)$, where z_θ is a finte integer, and $\nabla Q(\theta|\theta^*)$ is the gradient w.r.t population, under strong convexity condition and smoothness assumption which already hold with stepsize $0 \leq \eta \leq \frac{2}{\epsilon+l}$, and because $T(\theta)$ is contractive, we have:*

$$\|L(\theta) - \theta^*\|_2 \leq (1 - \frac{2\eta l\epsilon}{l+\epsilon})^{z_\theta} \|\theta - \theta^*\|_2 \tag{17}$$

Similarly using the exactly the same technique as before we can get:

$$\|G(\theta) - \theta^*\|_2 \leq ((1 - \frac{2\eta l\epsilon}{l+\epsilon})^{z_\theta} + \eta\gamma) \|\theta - \theta^*\|_2 \tag{18}$$

Therefore we are done preparing for all the mathematical machineries to prove the convergence guarantee. Denote $\Delta_{t+1} := \theta_{t+1} - \theta^*$, we have that:

$$\begin{aligned} &\|\Delta_{t+1}\|_2^2 - \|\Delta_t\|_2^2 \leq (\eta_t)^2 \left\|\hat{V}(\theta_t)\right\|_2^2 + 2\eta_t \left\|\hat{V}(\theta_t) \cdot \Delta_t\right\|_2 \\ \Longrightarrow\ &E[\|\Delta_{t+1}\|_2^2] \leq E[\|\Delta_t\|_2^2] + (\eta_t)^2 E[\left\|\hat{V}(\theta_t)\right\|_2^2] + 2\eta_t E[\left\|\hat{V}(\theta_t) \cdot \Delta_t\right\|_2] \end{aligned}$$

Since $\hat{V}(\theta^*) = 0$, we have:

$$E[\|\Delta_{t+1}\|_2^2] \leq E[\|\Delta_t\|_2^2] + (\eta_t)^2 E[\left\|\hat{V}(\theta_t)\right\|_2^2] + 2\eta_t E[\left\|(\hat{V}(\theta_t) - \hat{V}(\theta^*)) \cdot \Delta_t\right\|_2]$$

Then we upper bound the last term using $(\|G(\theta) - \theta^*\|_2 \leq (1 - \frac{2\eta l\epsilon}{l+\epsilon})^{z_\theta} + \eta\gamma) \|\theta - \theta^*\|_2$, which is: $2\eta_t E[\left\|(\hat{V}(\theta_t) - \hat{V}(\theta^*)) \cdot \Delta_t\right\|_2 \leq (1 - \frac{2\eta l\epsilon}{l+\epsilon})^{z_\theta} + \eta\gamma - 1) \|\theta_t - \theta^*\|_2$ and we get:

$$E[\|\Delta_{t+1}\|_2^2] \leq E[\|\Delta_t\|_2^2] + (\eta_t)^2 E[\left\|\hat{V}(\theta_t)\right\|_2^2] - 2((1 - \frac{2\eta_t l\epsilon}{l+\epsilon})^{z_\theta} + \eta_t\gamma - 1)E[\|\Delta_t\|_2^2]$$

For simplicity it's safe to set $z(\theta) = 1$ as the inequality still holds and we get:

$$E[\|\Delta_{t+1}\|_2^2] \leq E[\|\Delta_t\|_2^2] + (\eta_t)^2 E[\left\|\hat{V}(\theta_t)\right\|_2^2] - 2\eta_t\xi E[\|\Delta_t\|_2^2]$$

where $\xi = \frac{2l\epsilon}{l+\epsilon} - \gamma$, combining all the previous results and upper bounding the second term $\sup_{\theta\epsilon\Lambda} E[\left\|\hat{V}(\theta_t)\right\|_2^2] = \sigma_V^2$:

$$\begin{aligned} E[\|\Delta_{t+1}\|_2^2] &\leq (1-2\eta_t\xi)E[\|\Delta_t\|_2^2] + (\eta_t)^2 E[\left\|\hat{V}(\theta_t)\right\|_2^2] \\ &\leq (1-\eta_t\xi)E[\|\Delta_t\|_2^2] + (\eta_t)^2\sigma_V^2 \end{aligned}$$

Setting $\eta_t = \frac{3}{2\xi(t+2)}$ and nwrapping the recursion, after some algebra work including summation, multiplication, contraction and upper bounding,

$$E[\|\Delta_{t+1}\|_2^2] \leq \frac{9\sigma_V^2}{\xi^2}\frac{1}{t+2} + (\frac{2}{t+2})^{3/2}\|\Delta_0\|_2^2 \tag{19}$$

Which summarize the guarantee of convergence.

■

4 Experiment

Several algorithms were performed on a multi-class dataset distributed as **??** describes, $t = 4000$ and $n = 4$, as can be easily observed that different classes have intersection such that they cannot be perfectly classified by any partition function in the 2 dimensional space. We simply choose $h(y) = \frac{\mathbb{1}(y=k)}{\sum_{k=1}^{4}\mathbb{1}(y=k)}$ to be the prior, and $f_x(y) = [\mathbb{1}(y=1)x^\top \mathbb{1}(y=2)x^\top, \mathbb{1}(y=3)x^\top, \mathbb{1}(y=4)x^\top]^\top$ to be the kernel mapping.

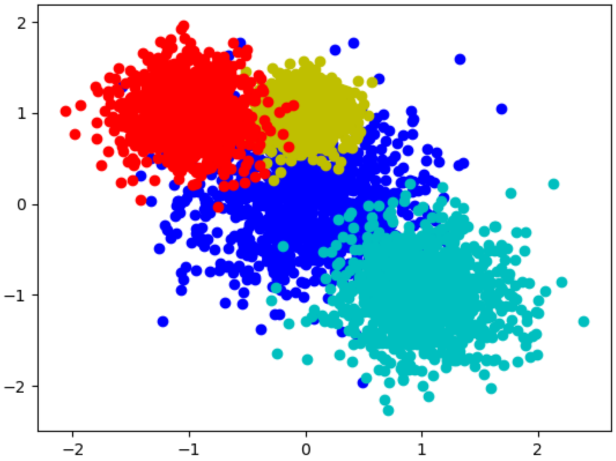


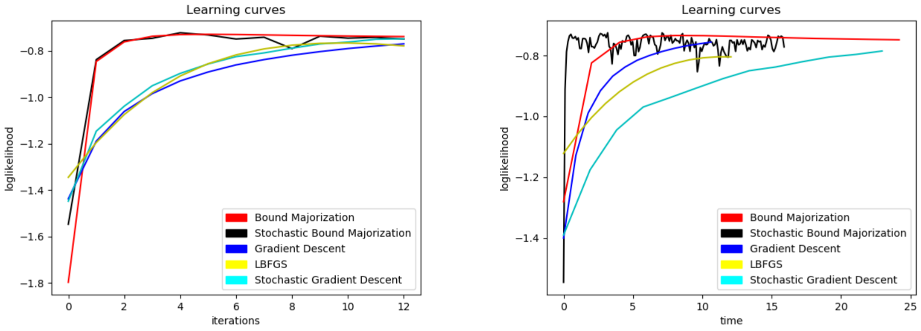
Figure 2: multiclass data points

we explored SBM, BM, LBFGS, GD and SGD whose parameter settings are tuned and shown in **??**

Table 1: parameter setting

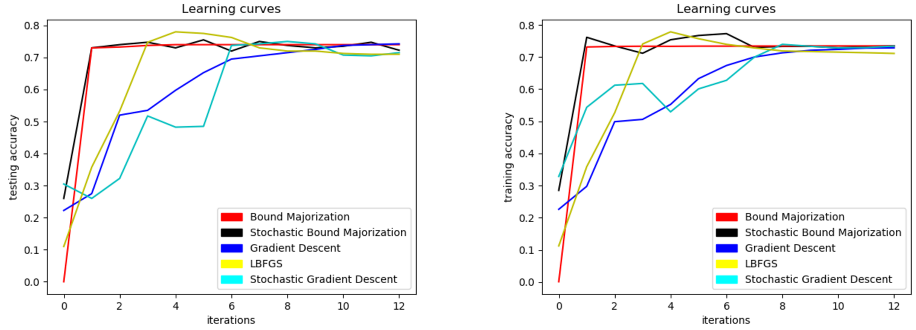
p : batch size		m : number of vectors in LBFGS		
BM	SBM	LBFGS	GD	SGD
$\lambda = 1e-2$ $\epsilon = 1e-6$ $\eta = 1$	$p = 40$ $\epsilon = 1e-6$ $\eta = 1e-2$	η : line search $\epsilon = 1e-5$ $m = 4$	$\lambda = 1e-2$ $\epsilon = 1e-5$ $\eta = 1e-2$	$p = 40$ $\epsilon = 1e-5$ $\eta = 1e-2$

The loglikelihood in the training procedure with respect to time and iteration were plotted, along with the training and testing error with respect to the epoch. Just as what is shown in **??** and **??**.



(a) (b)

Figure 3: loglikelihood change



(a) (b)

Figure 4: training and testing accuracy

As can be seen clearly from the learning curves above, The BM algorithm and SBM algorithm surpassed others in convergence speed with respect to both time and update epoch(where a single iteration corresponds to a single update of the parameter vector), while SBM requires much less time than any other algorithms.

Note that 90% of the data is used for training and the rest for testing after randomly permuting the data, the results are averaged over 10-fold cross-validation, several values of λ were explored and tuned.

5 Conclusion

Starting with interpretation of the log-linear partition model, the theoretical development of the algorithm has been revealed as a majorization procedure similiar to EM algorithm. Applying Sherman Morrison formula to get **??** through which we can update without computing the matrix inverse, thus improving the efficiency. The convergence behavior was described not only in a sense of how many iteration is required under corresponding tolerance, but also in a sense of how close it would get to optimal after certain iterations, the proofs are given analogous to proving the statitiscal guarantee of stochastic EM, several assumptions were proposed without loss of generosity.

the proposed method, which requires no parameter tuning, has been evalueated as much more efficient than most of the first order methods such as LBFGS and SGD. The main drawbacks is that it can only be applied to log-linear model, in the context of specific partition loss function which has the nice property to allow us find a simple quadratic bound.

References

[1] Jebara, Tony, and Anna Choromanska. *"Majorization for CRFs and latent likelihoods." Advances in Neural Information Processing Systems.* 2012.

[2] Choromanska, Anna, and Tony Jebara. *"Stochastic bound majorization." arXiv preprint arXiv:1309.5605* (2013).

[3] T. Jebara. *Multitask sparsity via maximum entropy discrimination. JMLR, 12:75110,* 2011.