EE-556 Mathematics of Data: From Theory to Computation

Homework 1

WEI Jiheng (319872)

October 31, 2021

1 Logistic Regression

(a). Let $g(u) = \log(1 + e^{-u})$, then

$$g'(u) = -\frac{e^{-u}}{1 + e^{-u}} = -\frac{1}{1 + e^{u}}, \tag{1}$$

$$g''(u) = \frac{e^{u}}{(1 + e^{u})^2}. \tag{2}$$

Since $g''(u) > 0$ for all u, the function is strictly convex.

$-b_i \boldsymbol{a}_i^\top \boldsymbol{x}$ is a linear function and thus convex for every $\{(\boldsymbol{a}_i, b_i)\}$, then $\log(1+\exp(-b_i \boldsymbol{a}_i^\top \boldsymbol{x}))$ preserves the convexity. As a result, $f(\boldsymbol{x}) = \sum_{i=1}^n \log(1 + \exp(-b_i \boldsymbol{a}_i^\top \boldsymbol{x}))$ is convex because it is the sum of convex functions. Due to the strict convexity of $f(\boldsymbol{x})$, there is a lower bound for its value. Nevertheless, it is not guaranteed that the function can attain this bound.

(b). The infimum of an arbitrary function f is a value a such that $f(\boldsymbol{x}) \geqslant a$ for all $\boldsymbol{x}$. The minimum is the smallest value of the function in its domain. Therefore, the infimum does not necessarily need to be attained by the function.

The example is e^{-x} whose infimum is 0. However, when $x \to -\infty$, the function approaches but will never arrive at 0.

(c). If $b_i \boldsymbol{a}_i^\top \boldsymbol{x}_0 > 0$, then $\exp(-b_i \boldsymbol{a}_i^\top \boldsymbol{x}_0) \in (0, 1)$. Then, the value of $\log(1 + \exp(-b_i \boldsymbol{a}_i^\top \boldsymbol{x}))$ will always be positive for all i such that $f(\boldsymbol{x}_0) > 0$. Speaking of $f(\alpha \boldsymbol{x}_0)$,

$$\lim_{\alpha \to +\infty} f(\alpha \boldsymbol{x}_0) = \sum_{i=1}^{n} \log(1 + e^{-\infty}) = 0. \tag{3}$$

This means that the infimum of f is attained only with a $\boldsymbol{x}_0$ which is infinitely large. In other words, there is no finite estimator $\boldsymbol{x}^\star_{\mathrm{ML}}$.

(d).

$$\begin{aligned}\nabla f_\mu(\boldsymbol{x}) &= \sum_{i=1}^{n} -\frac{b_i \exp(-b_i \boldsymbol{a}_i^\top \boldsymbol{x})}{1+\exp(-b_i \boldsymbol{a}_i^\top \boldsymbol{x})}\boldsymbol{a}_i + \mu\boldsymbol{x} \\ &= \sum_{i=1}^{n} -\frac{b_i}{1+\exp(b_i \boldsymbol{a}_i^\top \boldsymbol{x})}\boldsymbol{a}_i + \mu\boldsymbol{x} \\ &= \sum_{i=1}^{n} -b_i \sigma(-b_i \boldsymbol{a}_i^\top \boldsymbol{x})\boldsymbol{a}_i + \mu\boldsymbol{x}.\end{aligned} \tag{4}$$

(e). It is obvious that the Hessian of $\mu\boldsymbol{x}$ is $\mu\boldsymbol{I}$. The jth column of the first term's Hessian can be expressed as

$$\begin{aligned}\sum_{i=1}^{n} \frac{b_i^2 \exp(-b_i \boldsymbol{a}_i^\top \boldsymbol{x})}{[1+\exp(-b_i \boldsymbol{a}_i^\top \boldsymbol{x})]^2} a_i^{(j)} \boldsymbol{a}_i^\top &= \sum_{i=1}^{n} b_i^2 \frac{1}{1+\exp(-b_i \boldsymbol{a}_i^\top \boldsymbol{x})} \frac{\exp(-b_i \boldsymbol{a}_i^\top \boldsymbol{x})}{1+\exp(-b_i \boldsymbol{a}_i^\top \boldsymbol{x})} a_i^{(j)} \boldsymbol{a}_i^\top \\ &= \sum_{i=1}^{n} \sigma(-b_i \boldsymbol{a}_i^\top \boldsymbol{x})[1-\sigma(-b_i \boldsymbol{a}_i^\top \boldsymbol{x})] a_i^{(j)} \boldsymbol{a}_i^\top,\end{aligned} \tag{5}$$

where $a_i^{(j)}$ is the jth component of $\boldsymbol{a}_i$ and $b_i \in \{-1, +1\} \implies b_i^2 = 1$. By stacking the columns together, we have

$$\nabla^2 f_\mu(\boldsymbol{x}) = \sum_{i=1}^{n} \sigma(-b_i \boldsymbol{a}_i^\top \boldsymbol{x})[1-\sigma(-b_i \boldsymbol{a}_i^\top \boldsymbol{x})] \boldsymbol{a}_i \boldsymbol{a}_i^\top + \mu\boldsymbol{I}. \tag{6}$$

(f). $\nabla^2 f_\mu$ is a symmetric matrix because

$$\begin{aligned}\nabla^2 f_\mu^\top &= \boldsymbol{A}^\top \operatorname{diag}\left(\sigma(-b_i \boldsymbol{a}_i^\top \boldsymbol{x})[1-\sigma(-b_i \boldsymbol{a}_i^\top \boldsymbol{x})]\right)^\top \boldsymbol{A} + \mu\boldsymbol{I}^\top \\ &= \boldsymbol{A}^\top \operatorname{diag}\left(\sigma(-b_i \boldsymbol{a}_i^\top \boldsymbol{x})[1-\sigma(-b_i \boldsymbol{a}_i^\top \boldsymbol{x})]\right) \boldsymbol{A} + \mu\boldsymbol{I} \\ &= \nabla^2 f_\mu.\end{aligned} \tag{7}$$

Also, given that $\sigma(-b_i \boldsymbol{a}_i^\top \boldsymbol{x}) \in (0,1)$, $1-\sigma(-b_i \boldsymbol{a}_i^\top \boldsymbol{x}) \in (0,1)$ and $\mu > 0$, it is obvious that $\nabla^2 f_\mu \succ 0$. Hence, f_μ is μ-strongly convex.

2 Numerical methods for Logistic Regression

2.1 First-order methods

Table 1: First-order optimisation results

Algorithm	Error (%)	Time (s)
GD	13.87	34
GDstr	9.49	34
AGD	**5.84**	43
AGDstr	**5.84**	43
AGDR	7.30	**4**
AdaGrad	**5.84**	34

All the algorithms achieved an error rate of less than 10% except the ordinary gradient descent method. When the fact that f in this problem is μ-strongly convex is taken into account, the `str` version of the algorithms converges faster. Since the accelerated descent methods are not monotone, we can observe that the more the algorithm approaches the solution, the more it oscillates. This phenomenon is cured by the restart strategy.

`AGDR` spent much less time than other methods but is only the second in terms of correctness. Other methods, regardless of the resulting error rates, require time at the same level.

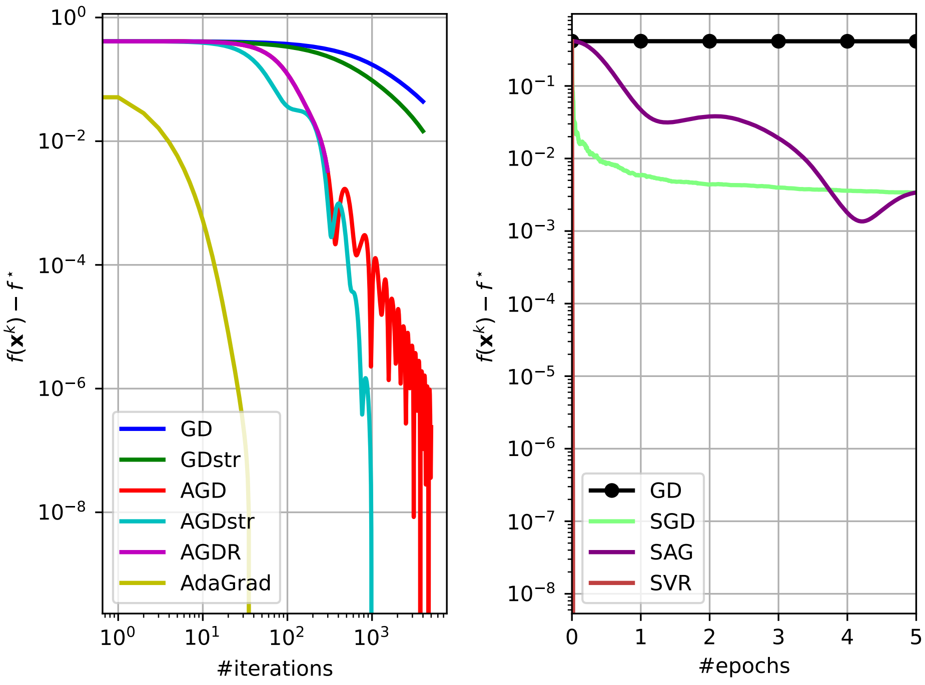


Figure 1: First-order and stochastic gradient methods

2.2 Stochastic gradient methods

(a).

$$\nabla f(\boldsymbol{x}) = \frac{1}{n} \sum_{i=1}^{n} \left\{ -b_i \sigma(-b_i \boldsymbol{a}_i^\top \boldsymbol{x}) \boldsymbol{a}_i + \mu \boldsymbol{x} \right\}. \tag{8}$$

Since all the data points can be picked with equal probability at each iteration (i.e. $\forall i, p(i) = 1/n$),

$$\mathbb{E}[\nabla f_{i_k}(\boldsymbol{x})] = \sum_{i=1}^{n} \frac{1}{n} \left\{ -b_i \sigma(-b_i \boldsymbol{a}_i^\top \boldsymbol{x}) \boldsymbol{a}_i + \mu \boldsymbol{x} \right\} = \nabla f(\boldsymbol{x}). \tag{9}$$

Hence, $\nabla f_{i_k}(\boldsymbol{x})$ is an unbiased estimator of $\nabla f(\boldsymbol{x})$.

Table 2: Stochastic optimisation results

Algorithm	Error (%)	Time (s)
SGD	6.57	**7**
SAG	**5.11**	8
SVR	5.84	32

The stochastic algorithms are not monotone due to random sampling during each iteration. Nevertheless, it is generally more efficient than first-order methods. The performance of SGD is on par with that of SAG, but the latter is slightly better. Though SVR gives a sharp descent, we can see that its iterations are much more computationally expensive.

2.3 Proximal methods

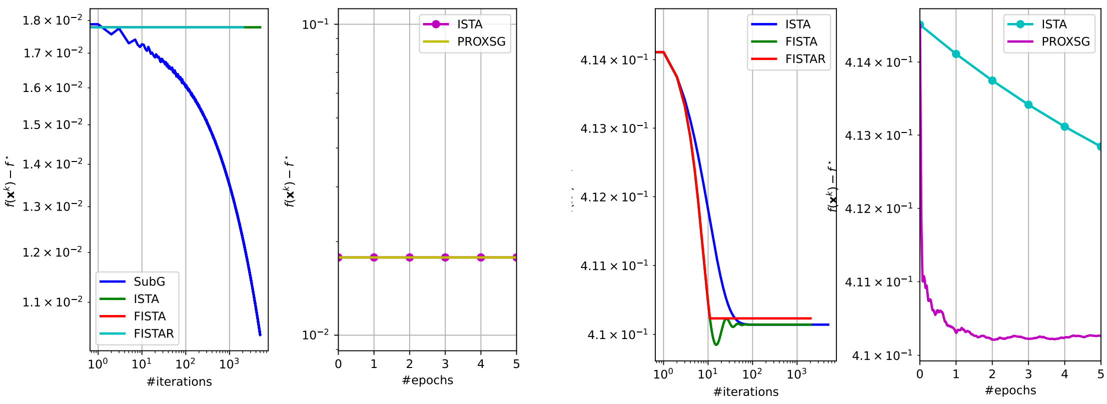
(a).

$$\mathrm{prox}_{\lambda g}(\boldsymbol{z}) = \arg\min_{\boldsymbol{y}} \left\{ \lambda\|\boldsymbol{y}\|_1 + \frac{1}{2}\|\boldsymbol{y}-\boldsymbol{z}\|_2^2 \right\}. \tag{10}$$

$$\nabla_{\boldsymbol{y}} \left(\lambda\|\boldsymbol{y}\|_1 + \frac{1}{2}\|\boldsymbol{y}-\boldsymbol{z}\|_2^2 \right) = \lambda\,\mathrm{sign}(\boldsymbol{y}) + \boldsymbol{y} - \boldsymbol{z}. \tag{11}$$

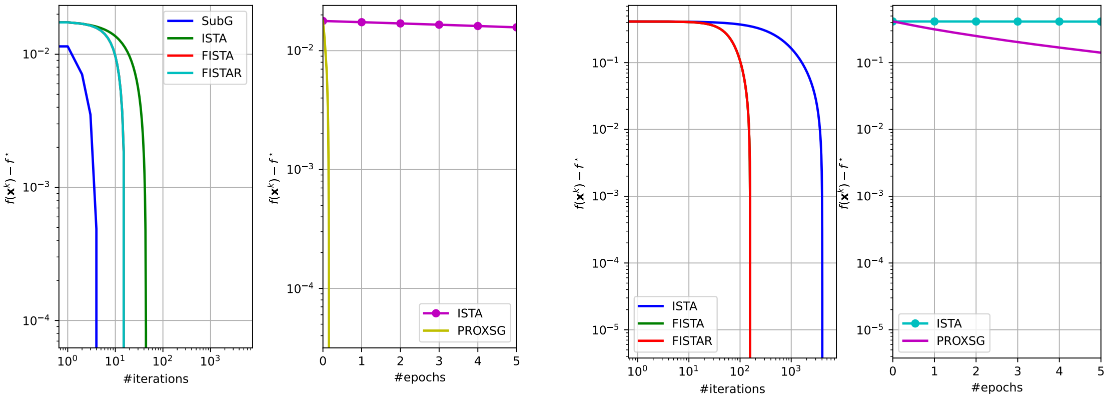
Speaking of the vectors' entries, if $y_i \neq 0$, the above gradient vanishes at $y_i = z_i - \lambda\,\mathrm{sign}(y_i)$. The constraint that $|z_i| > \lambda$ should be satisfied in order to have the same sign on both sides of the equation. If $y_i = 0$, the subgradient of $\lambda|y_i|$ is the interval $[-\lambda, \lambda]$. Then, the gradient vanishes at any $z_i \in [-\lambda, \lambda]$. As a result,

$$\mathrm{prox}_{\lambda g}(\boldsymbol{z}) = \max(|\boldsymbol{z}| - \lambda, 0) \otimes \mathrm{sign}(\boldsymbol{z}). \tag{12}$$



(a) ℓ_1 regulariser (b) ℓ_2 regulariser

Figure 2: Proximal methods, $\lambda_{\ell_1} = 0.3, \lambda_{\ell_2} = 0.1$



(a) ℓ_1 regulariser (b) ℓ_2 regulariser

Figure 3: Proximal methods, $\lambda_{\ell_1} = \lambda_{\ell_2} = 10^{-5}$

Table 3: Proximal optimisation results, $\lambda_{\ell_1} = \lambda_{\ell_2} = 10^{-5}$

	ℓ_1 regulariser		ℓ_2 regulariser	
Algorithm	Error (%)	Time (s)	Error (%)	Time (s)
`SubG`	13.87	41	-	-
`ISTA`	11.68	40	11.68	40
`FISTA`	**4.38**	16	**4.38**	16
`FISTAR`	5.11	16	**4.38**	16
`PROXSG`	13.87	**12**	13.87	**12**

The parameter values originally imposed ($\lambda_{\ell_1} = 0.3, \lambda_{\ell_2} = 0.1$) did not work well. There was no descent at all for `ISTA`, `FISTA(R)` with ℓ_1-norm as regulariser. However, we can already observe that in the ℓ_2 case, `FISTA` oscillates before convergence without restart condition.

As a result, much smaller values ($\lambda_{\ell_1} = \lambda_{\ell_2} = 10^{-5}$) were used. This time, we can observe that the descent is abrupt for all methods. `FISTA` is the best in terms of accuracy-time trade-off. Though `PROXSG` is the fastest, it ranked the last in error rate. Generally, the results are the same regardless of the regulariser used, except for a minor improvement for `FISTAR` with ℓ_2-norm.

2.4 Convergence rates

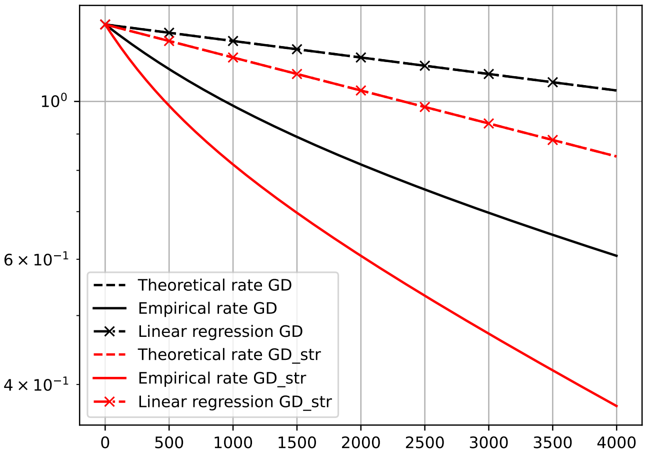


Figure 4: Convergence plot

(a). The theoretical convergence rates are upper bounds of the real ones. We would expect that the performance in reality is at least as good as in theory. Since the observed value of $\|\boldsymbol{x}^k - \boldsymbol{x}^\star\|$ decreases faster than theoretical ones, the convergence rates are consistent.

(b). The observed convergence rates become straight lines after about 2000 interations. Given that the plot is log-linear, the rates are linear.

(c).
- $\alpha = 1/L$:

$$\log(\|\boldsymbol{x}^k - \boldsymbol{x}^\star\|_2) \leqslant \frac{\log\left(\frac{L-\mu}{L+\mu}\right)}{2}k + \log(\|\boldsymbol{x}^0 - \boldsymbol{x}^\star\|_2). \tag{13}$$

$$a = \|\boldsymbol{x}^0 - \boldsymbol{x}^\star\|_2, \quad b = \sqrt{\frac{L-\mu}{L+\mu}}. \tag{14}$$

- $\alpha = 2/(L+\mu)$:

$$\log(\|\boldsymbol{x}^k - \boldsymbol{x}^\star\|_2) \leqslant \log\left(\frac{L-\mu}{L+\mu}\right) k + \log(\|\boldsymbol{x}^0 - \boldsymbol{x}^\star\|_2). \tag{15}$$

$$a = \|\boldsymbol{x}^0 - \boldsymbol{x}^\star\|_2, \quad b = \frac{L-\mu}{L+\mu}. \tag{16}$$

(d). From the plot, we can see that regression estimations overlap the theoretical rates. Hence, the estimation is valid.

Table 4: Convergence rate estimation

	Theoretical		Regression		
Algorithm	a	b	a	b	Rate diffrence
`GD`	1.282909	0.999947	1.282909	0.999947	0
`GDstr`	1.282909	0.999893	1.282909	0.999893	0

3 Image reconstruction

3.1

(a).

$$\nabla_{\boldsymbol{\alpha}} f_{l_1}(\boldsymbol{\alpha}) = -\boldsymbol{W}\boldsymbol{P}_\Omega^\top(\boldsymbol{b} - \boldsymbol{P}_\Omega \boldsymbol{W}^\top \boldsymbol{\alpha}). \tag{17}$$

$$\nabla_{\boldsymbol{x}} f_{\mathrm{TV}}(\boldsymbol{x}) = -\boldsymbol{P}_\Omega^\top(\boldsymbol{b} - \boldsymbol{P}_\Omega \boldsymbol{x}). \tag{18}$$