Weighted Low Rank Matrix Approximation

Elena Tuzhilina

1 Background

The Low-rank matrix-completion (LRC) problem has always attracted a lot of attention (see, for example, [1]). One of the most famous application can be found in Netflix Problem, which aims to build a movie recommendation system based on partial user's movie ratings. The general LRC problem statement is: given some incomplete matrix $A \in \mathbf{R}^{n\times m}$ find a low-rank matrix $X \in \mathbf{R}^{n\times m}$ that provides the best approximation to the observed entries of A. There are many ways to define "goodness of approximation" (see, for example, [2], [4]). One of the most convenient ways was suggested by Mazumder et al. in [3] and uses Frobenius distance between projections of A and X onto the set of observed entries of A. Specifically, for a set of observed entries $\Omega \in \{1,\dots,n\}\times\{1,\dots,m\}$ define the projection $P_\Omega(X)$ to be a matrix of the same dimensions as X with entries

$$(P_\Omega(X))_{ij} = \begin{cases} 0, & \text{if } (i,j)\notin\Omega \\ X_{ij}, & \text{if } (i,j)\in\Omega. \end{cases}$$

Then the LRC problem can be stated as

$$\text{minimize} \quad \|P_\Omega(A-X)\|_F^2 \quad \text{w.r.t} \quad X\in\mathbf{R}^{n\times m} \quad \text{subject to} \quad \mathrm{rk}(X)=k \tag{1}$$

where $\|\cdot\|_F$ refers to matrix Frobenius norm.

2 Generalization

Note that $P_\Omega(X)$ can be rewritten as $W * X$, where $W \in \mathbf{R}^{n\times k}$ is a binary matrix with

$$W_{i,j} = \begin{cases} 0, & \text{if } (i,j)\notin\Omega \\ 1, & \text{if } (i,j)\in\Omega \end{cases}$$

and $*$ refers to the Hadamard element-wise product. The natural generalization of LRC problem is, therefore, the Weighted Low Rank Matrix Approximation (WLRA) problem

$$\text{minimize} \quad \|\sqrt{W}*(A-X)\|_F^2 \quad \text{w.r.t} \quad X\in\mathbf{R}^{n\times m} \quad \text{subject to} \quad \mathrm{rk}(X)=k. \tag{2}$$

Here $W\in[0,1]^{n\times k}$ is some matrix of weights (not necessary binary). In [5] Srebro et al. suggest a simple iterative algorithm to solve problem (2) that successively applies the following update

$$X^{(t+1)} = LRA_k(W*A+(1-W)*X^{(t)}), \tag{3}$$

where $LRA_k(X)$ refers to the rank-k approximation of matrix X which can be computed via Singular Value Decomposition of X. Namely, if $X = UDV^T$ and D_k is the matrix with all except the first k singular values shrunk to zero, then $LRA_k(X) = UD_kV^T$.

The update (3) has a natural interpretation as follows: for your current guess $X^{(t)}$ you first "mix" it with the ground truth A (the weights W and $1-W$ correspond to the amount of attention you need to pay to the elements of A and $X^{(t)}$, respectively, while combining them together) and then project the combination back to the rank k matrix space. Another interesting observation is that this procedure is concordant with the famous hard-impute algorithm (see, for example, [3]) that utilizes update $X := LRA_k(P_\Omega(A)+P_{\Omega^\perp}(X))$ for solving LRC problem (1).

3 WLRA Extensions

Although, the authors motivate the WLRA update (3) by EM Procedure, one can show that this update is nothing but Projected Gradient Descent. Specifically, if $f(X) = \frac{1}{2}\|\sqrt{W} * (A - X)\|_F^2$ and $\Pi_C(\cdot)$ refers to the projection onto set C, then the update is

$$X^{(t+1)} = \Pi_C\left(X^{(t)} - \nabla_X f\left(X^{(t)}\right)\right)$$
$$\text{where} \quad \nabla_X f(X) = W * (X - A) \quad \text{and} \quad C = \{X : \mathrm{rk}(X) = k\}. \tag{4}$$

The interpretation of in terms of PGD allows us to derive many natural extensions. For the sake of simplicity, hereafter we denote $G^{(t)} = \nabla_X f(X^{(t)}) = W * (X^{(t)} - A)$.

Add learning rate First, one can easily introduce learning rate to the update scheme:

$$X^{(t+1)} = \Pi_C\left(X^{(t)} - \alpha_t G^{(t)}\right) = LRA_k\left(\alpha_t W * A + (1 - \alpha_t W) * X^{(t)}\right).$$

In our experiments we try different step size rules, such as constant $\alpha_t = \alpha$, square summable $\alpha_t = \frac{1}{t^2}$, nonsummable diminishing $\alpha^{(t)} = \frac{1}{t}$ and Polyak's $\alpha_t = \frac{f(X^{(t)}) - f_{best}^{(t)} + \frac{1}{t}}{\|G^{(t)}\|_F^2}$.

Add acceleration Next, we try and compare different types of Projected Gradient Descent acceleration. There are two main categories of acceleration that we consider: the one that change the update for $X^{(t)}$ (e.g. Heavy Ball method and, in particular, Nesterov acceleration) and the one that update the gradient (e.g. Filtered Subgradient and CFM). These acceleration method have updates as follows

- Heavy Ball: $X^{(t+1)} = LRA_k\left(\alpha_t W * A + (1 - \alpha_t W) * X^{(t)} + \beta_k\left(X^{(k)} - X^{(k-1)}\right)\right)$
- Nesterov: $X^{(t+1)} = LRA_k\left(W * A + (1 - \alpha_t W) * X^{(t)} + \left(1 - \frac{t}{t+3}\right) X^{(k)} - \frac{t}{t+3} X^{(k-1)}\right)$
- Filtered Subgradient: $X^{(t+1)} = LRA_k\left(X^{(t)} - \alpha_k S^{(t)}\right)$ with $S^{(t)} = (1 - \beta)G(t) + \beta S^{(t)}$
- CFM: $X^{(t+1)} = LRA_k\left(X^{(t)} - \alpha_k S^{(t)}\right)$ with $S^{(t)} = G^{(k)} + \beta_k S^{(t)}$ and $\beta_k = \left(-\frac{1.5\langle S^{(t)}, G^{(t)}\rangle_F}{\|S^{(k)}\|_F^2}\right)_+$

Here $\langle\cdot,\cdot\rangle_F$ refers to Frobenius inner product and $(\cdot)_+ = \max(\cdot, 0)$.

ADMM modification Further, one can consider ADMM modification of WLRA algorithm. First we restate the WLRA problem in the form of separable objective. If again $C = \{X : \mathrm{rk}(X) = k\}$ is the set of rank k matrices, the problem (2) is equivalent to

$$\text{minimize} \quad \frac{1}{2}\|\sqrt{W} * (A - X)\|_F^2 + I_C(Z) \quad \text{w.r.t} \quad X, Z \in \mathbf{R}^{n \times m} \quad \text{subject to} \quad X = Z. \tag{5}$$

Then the ADMM update is

$$X^{(t+1)} = \mathrm{argmin}_X\left\{\frac{1}{2}\|\sqrt{W} * (A - X)\|_F^2 + \frac{\rho}{2}\|X - Z^{(t)} + U^{(t)}\|_F^2\right\}$$
$$Z^{(t+1)} = \Pi_C\left(X^{(t+1)} + U^{(t)}\right)$$
$$U^{(t+1)} = U^{(t)} + X^{(t+1)} - Z^{(t+1)}$$

It is possible to find explicit formula for X update which is:

$$X^{(t+1)} = \frac{1}{W + \rho} * (W * A + \rho(Z - U)),$$

here both sum and division in $\frac{1}{W+\rho}$ are considered to be element-wise operations.

Alternating convex optimization Note that problem (2) is equivalent to a weighted matrix factorization problem:

$$\text{minimize} \quad \|\sqrt{W} * (A - UV^T)\|_F^2 \quad \text{w.r.t} \quad U \in \mathbf{R}^{n\times k},\ V \in \mathbf{R}^{m\times k}. \tag{6}$$

Although this problem is not convex w.r.t. pair (U, V), it can be solved by means of alternating convex optimization. To be precise, if we consider U to be fixed then problem (6) becomes convex and boils down to a weighted multivariate and multiresponse regression problem with feature matrix U, response matrix A and weights W. Denoting the solution to weighted least squares problem by

$$WLS(X, Y, W) = \text{argmin}_B \|\sqrt{W} * (Y - XB)\|_F^2$$

we conclude that $V^T = WLS(U, A, W)$. Similarly, if V is fixed then U is the solution to some (convex) regression problem, i.e. $U^T = WLS(V, A^T, W^T)$. Note that to finding $WLS(U, A, W)$ and $WLS(V, A^T, W^T)$ requires to solve m and n separate weighted regression problems, respectively, each one will result in the update of a particular row of V and U. Although the procedure by itself is computationally expensive, it can be done in parallel.

4 Convex relaxation

Note that the set of rank k matrices is a non-convex set leading to the non-convexity of problem (2). In this section we consider the convex relaxation of problem (2) replacing the rank constraint by the the constraint on the nuclear norm:

$$\text{minimize} \quad \|\sqrt{W} * (A - X)\|_F^2 \quad \text{w.r.t} \quad X \in \mathbf{R}^{n\times k} \quad \text{subject to} \quad \|X\|_* \le c. \tag{7}$$

This can be rewritten in the form of regularized weighted matrix approximation problem as

$$\text{minimize} \quad \frac{1}{2}\|\sqrt{W} * (A - X)\|_F^2 + \lambda\|X\|_* \quad \text{w.r.t} \quad X \in \mathbf{R}^{n\times k}. \tag{8}$$

Thus, Projected Gradient Descent update (4) can be replaced by the Proximal Gradient Descent update

$$X^{(t+1)} = \text{prox}_{\lambda\|\cdot\|_*}\left(X^{(t)} - \nabla_X f\left(X^{(t)}\right)\right). \tag{9}$$

Note that since nuclear norm is orthogonal invariant, applying the proximal operator to some matrix X is the same as applying ℓ_1 proximal operator to the singular values of X. This is equivalent to applying the soft-threshold function with λ threshold to the singular values of X, i.e. if $X = UDV^T$ and $d = \text{diag}(D)$ corresponds to the vector of singular values of X, then

$$\text{prox}_{\lambda\|\cdot\|_*}(X) = UD_\lambda V^T \quad \text{where} \quad D_\lambda = \text{diag}((d - \lambda)_+).$$

Hereafter, we use the notation $SLRA_\lambda(X) = \text{prox}_{\lambda\|\cdot\|_*}(X)$ that stands for Soft Low Rank Approximation.

Mirror descent Note that one can easily replace $LRA_k(\cdot)$ operator by $SLRA_\lambda(\cdot)$ in the previous section thereby deriving various learning rate, acceleration and ADMM modifications of update (9). Assume now that $n = m$, W is symmetric and A is PSD, then we can restrict our search space to the set of PSD matrices with bounded trace norm and rewrite (7) as

$$\text{minimize} \quad \|\sqrt{W} * (A - X)\|_F^2 \quad \text{w.r.t} \quad X \in S_n(c), \tag{10}$$

where $S_n(c) = \{X \in S_+^n : \text{tr}(X) \le c\}$ is a spectrahedron of size c. Recall that Bregman divergence $D_h(X, Y)$ computed for von Neumann entropy $h(X) = \sum_{i=1}^n \lambda_i(X) \log \lambda_i(X)$, where $\lambda_i(X)$ is i-th eigenvalue of X, is strongly convex w.r.t. trace norm. Thus, one can use this Bregman divergence as an alternative to Frobenius norm while projecting matrices onto the spectrahedron. Moreover, this projection can be easily calculated via the following formula

$$\Pi^h_{S_n(c)}(X) = \begin{cases} c\frac{X}{\mathrm{tr}(X)} & \text{if } \mathrm{tr}(X) > c \\ X & \text{otherwise} \end{cases}$$

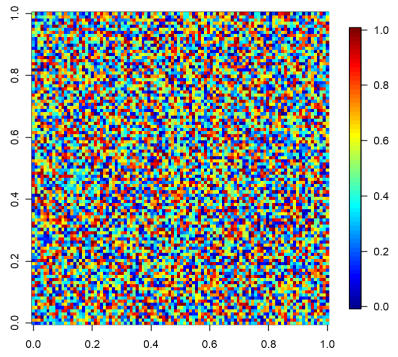
which leads to the Mirror Descent update

$$Y^{(t+1)} = \exp\left(\log\left(X^{(t)}\right) - \alpha_t G^{(t)}\right)$$
$$X^{(t+1)} = \Pi^h_{S_n(c)}(X).$$

Here both $\log(\cdot)$ and $\exp(\cdot)$ are applied to matrix singular values.

5 Experiments

WLRA We implemented all the above methods and compared their convergence on a small simulation example. In our experiments we set $m = n = 100$ and generated the elements of matrix A independently from standard normal distribution. The elements of matrix W are drawn independently from uniform distribution on $[0, 1]$ (see figure).



While seeking for the WLRA solution with rank $k = 5$ we consider the following set of hyperparameters for each method (note that alternating convex optimization method does not have any parameters):

- learning rate $\alpha = \{0.5, 1, 1.5, 2\}$;
- acceleration $\beta = \{0, 0.1, 0.2, 0.3, 0.4\}$;
- ADMM $\rho = \{0.5, 1, 1.5, 2, 2.5, 3\}$.

Next, we do a grid search across all the hyperparameters and pick the ones that correspond to the fastest convergence rate. Finally, we compare the best algorithms in terms of convergence speed which is measured in two ways: the deviation of loss from the optimal value, i.e. $|f(X^{(t)}) - p^*|$ (whrer p^* is pre-computed by running basic PGD for 1000 iterations with some small learning rate), and the absolute change in the optimal solution, i.e. $\|X^{(t+1)} - X^{(t)}\|_F$. To take into account the fact that all of the methods have different computational complexities we check the convergence speed vs. both iteration and time.

WLRA relaxation In this part we test the performance of the algorithms in the case of symmetric matrix W and for PSD matrix A. We utilize matrices A and W from the previous part to create such data. Specifically, we replace A by AA^T and symmetrize W by setting it's lower triangle to be equal to the upper triangle. We set regularization parameter to $\lambda = 50$ and consider the same set of hyperparameters as above adding one extra:

- Mirror descent $\alpha = \{0.001, 0.005, 0.01, 0.02\}$.

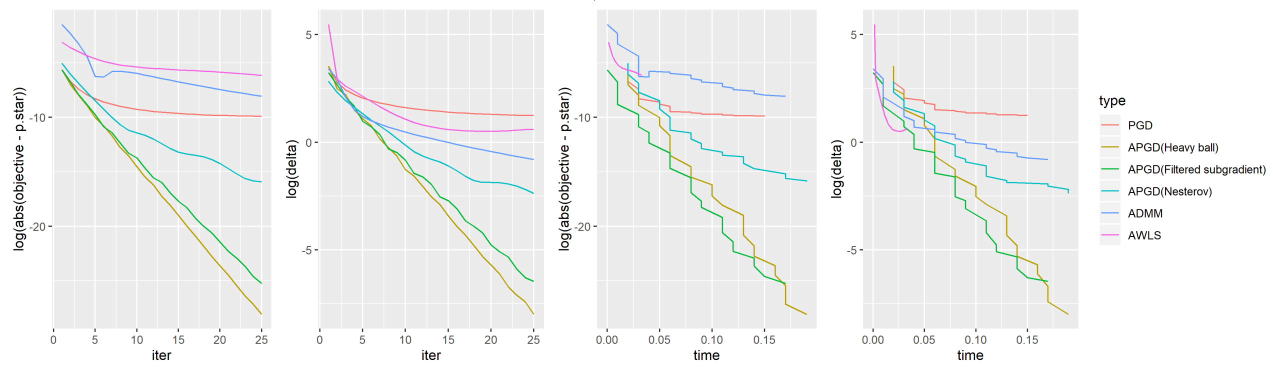
6 Results

WLRA The optimal values for the above approaches are (see Supplementary for the grid search plots)

- Basic PGD: $\alpha = 2$;
- Heavy Ball: $\alpha = 2$ and $\beta = 0.1$;

- Nesterov: $\alpha = 1.5$, method diverges for $\alpha = 2$;
- Filtered Subgradient: $\alpha = 2$ and $\beta = 0.1$;
- CFM: no parameters, this method did not speed up the convergence;
- ADMM: $\rho = 1$;
- Alternating convex optimization: no parameters.

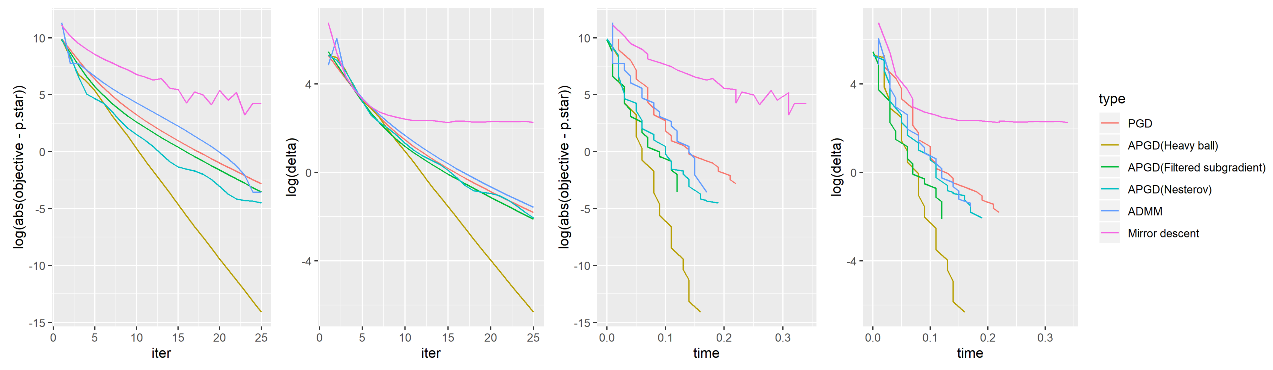
The plot below suggests that all the approaches under consideration converge to the optimal point. For both convergence metrics the best convergence speed measured in terms of iterations is demonstrated by PGD with the Heavy Ball acceleration. Filtered subgradient acceleration slightly outperforms Heavy Ball in terms of time. The worst convergence in terms of iterations is observed in the alternating convex optimization procedure (AWLS) case whereas ADMM performs the worst in terms of time (note that for AWLS the time per iteration is divided by the number of fitted weighted regressions as in the case of parallel computations).



WLRA relaxation The optimal values for the above approaches are (see Supplementary for the grid search plots)

- Basic PGD: $\alpha = 1$;
- Heavy Ball: $\alpha = 1$ and $\beta = 0.4$;
- Nesterov: $\alpha = 1$;
- Filtered Subgradient: $\alpha = 1$ and $\beta = 0.3$;
- CFM: no parameters, this method did not speed up the convergence;
- ADMM: $\rho = 1$;
- Mirror descent: $\alpha = 0.01$.

The following conclusions can be made from the relaxed WLRA plots. Again, the Heavy Ball acceleration significantly improves the convergence speed of Projected Gradient Descent. Other acceleration methods as well as ADMM behave similar to the Basic PGD in terms of speed. Unfortunately, the convergence of Mirror Descent is quite slow in terms of both convergence metrics.



7 Discussion and further directions

In this project we considered Weighted Low Rank Matrix Approximation Problem. We introduced and compared different modifications of basic Projected Gradient Descent. In addition we studied the convex relaxation of the problem and extensions that can be derived from Proximal Gradient Descent.

In practice, matrix A can be extremely large (as in Netflix Problem), so one possible interesting direction in which this project could be developed is to use distributed computation methods to perform the updates in parallel. Another interesting question for the future research is the possibility to utilize the link between WLRA and it's convex relaxation to study theoretical bounds on convergence.

References

[1] Luong Trung Nguyen, Junhan Kim, Byonghyo Shim, *Low-Rank Matrix Completion: A Contemporary Survey* (2019), arXiv, https://arxiv.org/abs/1907.11705

[2] Emmanuel J. Candes, Benjamin Recht, *Exact Matrix Completion via Convex Optimization* (2008), arXiv, https://arxiv.org/abs/0805.4471

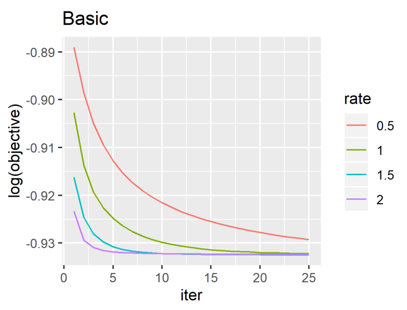
[3] Rahul Mazumder, Trevor Hastie, Robert Tibshirani, *Spectral Regularization Algorithms for Learning Large Incomplete Matrices* (2010), Journal of Machine Learning Research, https://web.stanford.edu/ hastie/Papers/mazumder10a.pdf

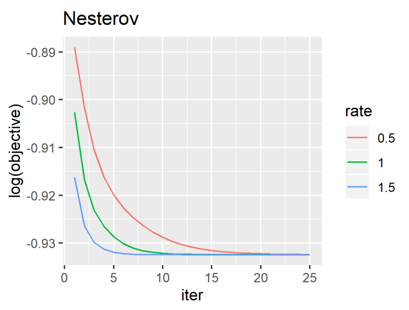
[4] Jian-Feng Cai, Emmanuel J. Candes, Zuowei Shen, *A Singular Value Thresholding Algorithm for Matrix Completion* (2008), arXiv, https://arxiv.org/abs/0810.3286

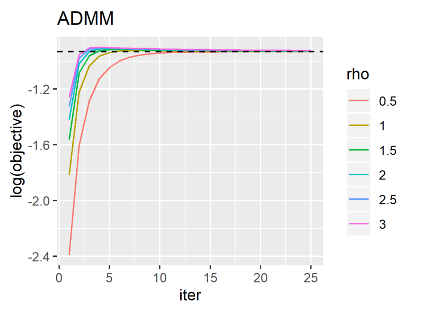
[5] Nathan Srebro, Tommi Jaakkola, *Weighted Low-Rank Approximations* (2003), ICML-2003, https://www.aaai.org/Papers/ICML/2003/ICML03-094.pdf

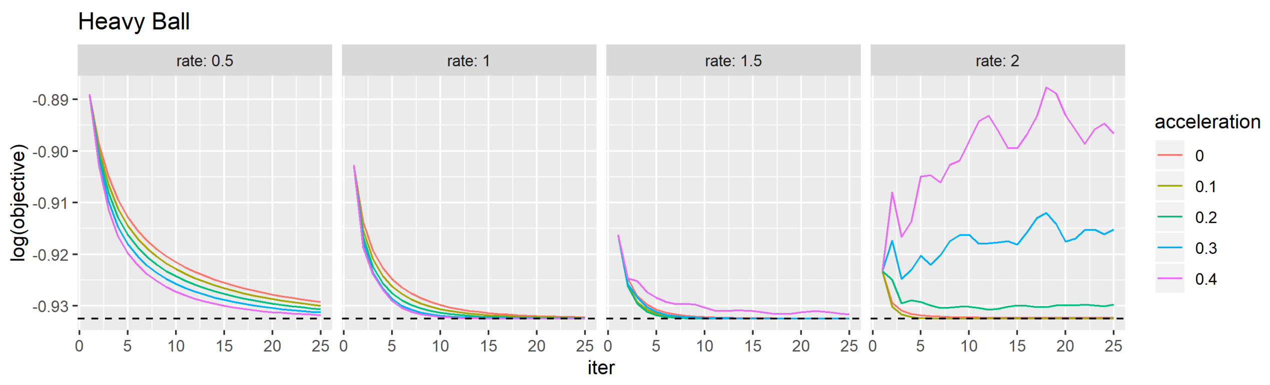
8 Supplementary

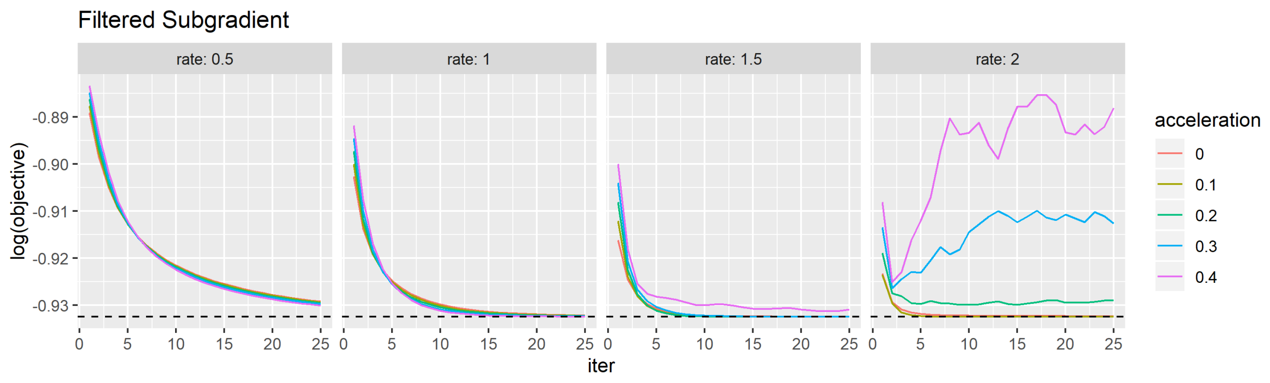
8.1 Plots for WLRA

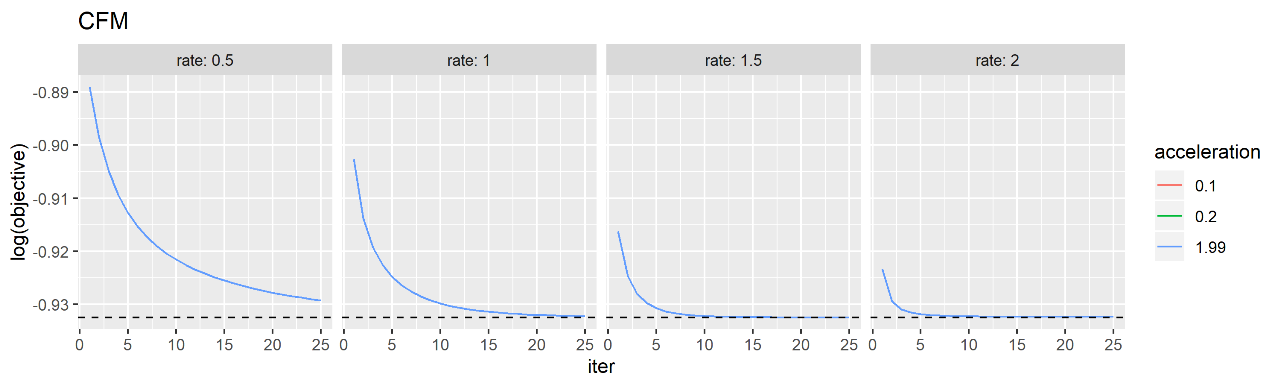




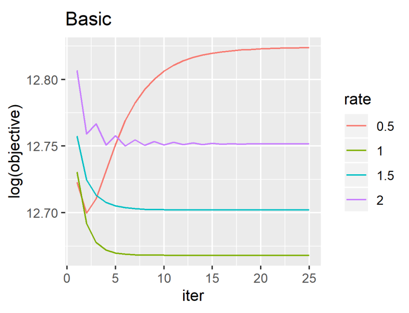


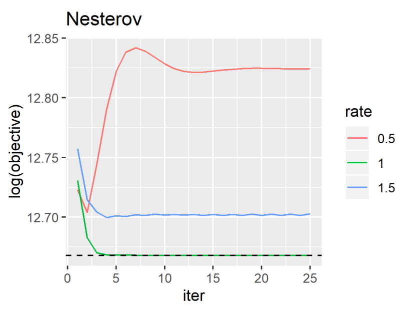


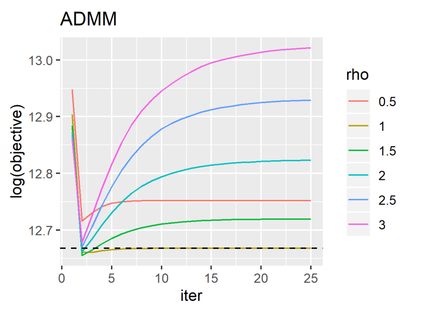


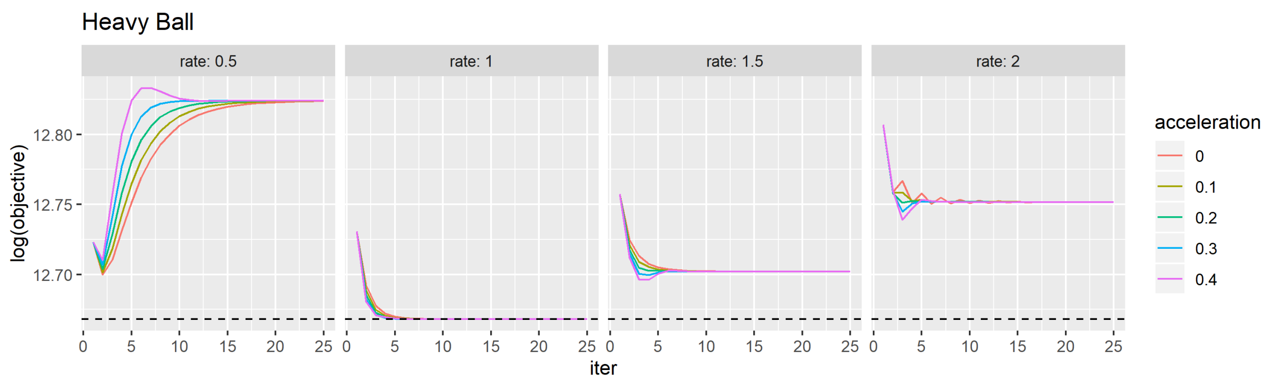


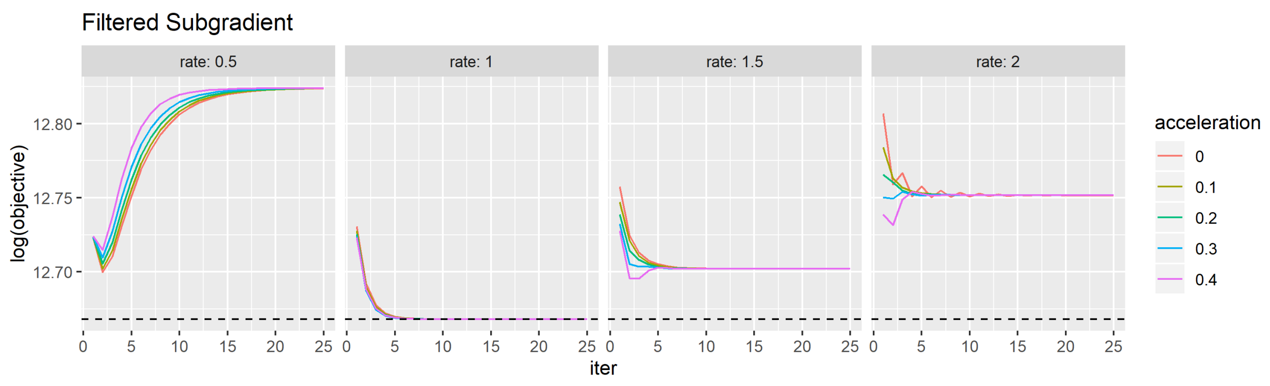
8.2 Plots for WLRA convex relaxation

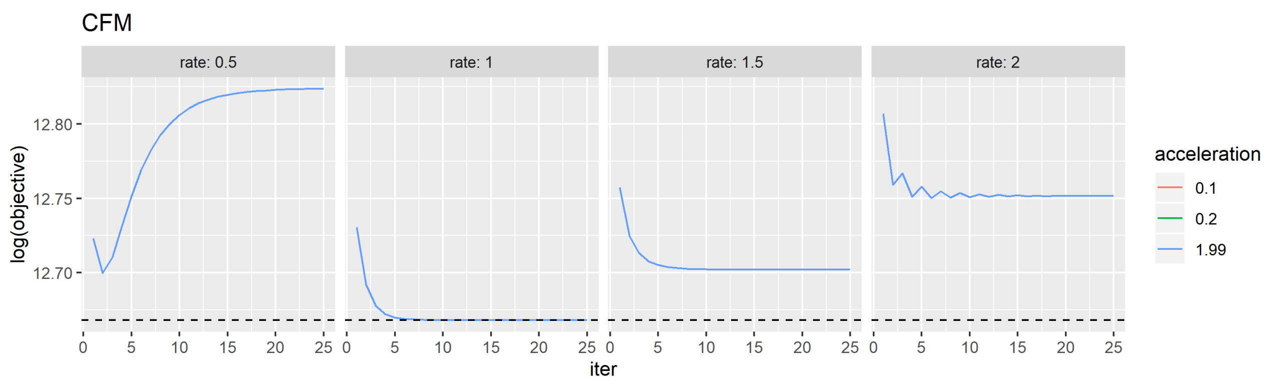

Basic
log(objective)
iter
rate

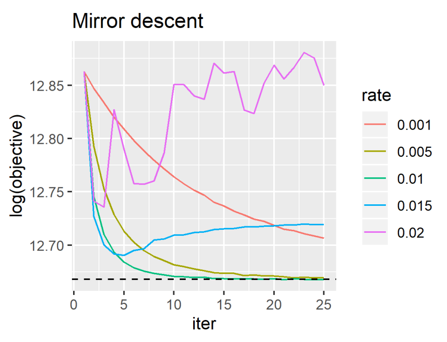

Nesterov
log(objective)
iter
rate


ADMM
log(objective)
iter
rho


Heavy Ball
rate: 0.5
rate: 1
rate: 1.5
rate: 2
log(objective)
iter
acceleration


Filtered Subgradient
rate: 0.5
rate: 1
rate: 1.5
rate: 2
log(objective)
iter
acceleration


CFM
rate: 0.5
rate: 1
rate: 1.5
rate: 2
log(objective)
iter
acceleration


Mirror descent
log(objective)
iter
rate