our method recovers $X$ better than LorSLIM. This may explain the superior performance of our method.

**Parameter Tunning**

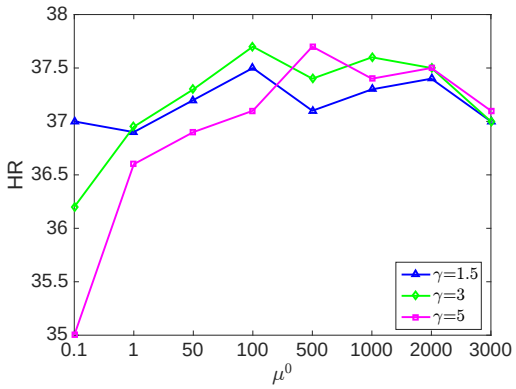


Figure 2: Influence of $\mu^0$ and $\gamma$ on HR for Delicious dataset.

Although our model is parameter-free, we introduce the auxiliary parameter $\mu$ during the optimization. In alternating direction method of multipliers (ADMM) (**?**), $\mu$ is fixed and it is not easy to choose an optimal value to balance the computational cost. Thus, a dynamical $\mu$, increasing at a rate of $\gamma$, is preferred in real applications. $\gamma > 1$ controls the convergence speed. The larger $\gamma$ is, the fewer iterations are to obtain the convergence, but meanwhile we may lose some precision. We show the effects of different initializations $\mu^0$ and $\gamma$ on HR on dataset Delicious in Figure 2. It illustrates that our experimental results are not sensitive to them, which is reasonable since they are auxiliary parameters controlling mainly the convergence speed. In contrast, LorSLIM needs to tune four parameters, which are time consuming and not easy to operate.

**Efficiency Analysis**

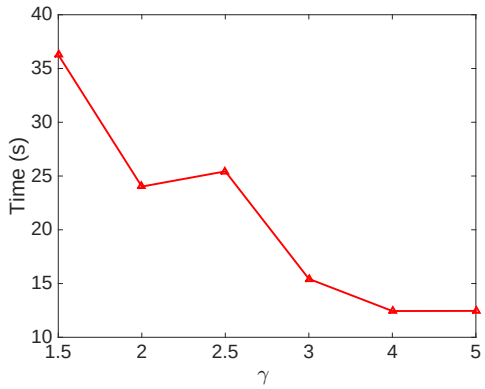


Figure 3: Influence of $\gamma$ on time.

The time complexity of our algorithm is mainly from SVD. Exact SVD of a $m \times n$ matrix has a time complexity of $O(\min\{mn^2, m^2n\})$, in this paper we seek a low-rank matrix and thus only need a few principal singular vectors/values. Packages like PROPACK (**?**) can compute a rank $k$ SVD with a cost of $O(min\{m^2k, n^2k\})$, which can be advantageous when $k \ll m, n$. In fact, our algorithm is much faster than LorSLIM. Among the six datasets, ML100K and lastfm datasets have the smallest and largest sizes, respectively. Our method needs 9s and 5080s, respectively, on these two datasets, while LorSLIM takes 617s and 32974s. The time is measured on the same machine with an Intel Xeon E3-1240 3.40GHz CPU that has 4 cores and 8GB memory, running Ubuntu and Matlab (R2014a). Furthermore, without losing too much accuracy, $\gamma$ can speed up our algorithm considerably. This is verified by Figure 3, which shows the computational time of our method on Delicious with varying $\gamma$.

## Conclusion

In this paper, we present a matrix completion based method for the Top-N recommendation problem. The proposed method recovers the user-item matrix by solving a rank minimization problem. To better approximate the rank, a non-convex function is applied. We conduct a comprehensive set of experiments on multiple datasets and compare its performance against that of other state-of-the-art Top-N recommendation algorithms. It turns out that our algorithm generates high quality recommendations, which improves the performance of the rest of methods considerably. This makes our approach usable in real-world scenarios.

## Acknowledgements

This work is supported by US National Science Foundation Grants IIS 1218712. Q. Cheng is the corresponding author.

Tempora eos provident aperiam earum quibusdam sint perferendis optio non animi sed, nemo quam culpa nulla possimus rem distinctio asperiores quae eveniet