APPENDIX

A. Gradient Descent Method

To optimize, we exploit gradient descent which is a classical optimization technique.

$$u_{ik} = u_{ik} - \theta_{ik}\frac{\partial \mathcal{O}(\mathbf{U},\mathbf{V})}{\partial U_{ik}}$$
$$= u_{ik} + 2\theta_{ik}(\mathcal{R}_\Omega(\mathbf{X})\mathbf{V}^T - \mathcal{R}_\Omega(\mathbf{UV})\mathbf{V}^T - \lambda\mathbf{LU})_{ik}$$

Similarly, we have

$$v_{kj} = \begin{cases} v_{kj} - \delta_{kj}(-2(\mathbf{U}^T\mathcal{R}_\Omega(\mathbf{X}))_{kj} \\ \quad + 2(\mathbf{U}^T\mathcal{R}_\Omega(\mathbf{UV}))_{kj}), & (k,j) \notin \Phi \\ c_{kj}, & (k,j) \in \Phi \end{cases}$$

where θ_{ik} and δ_{kj} are learning rates for updating u_{ik} and v_{kj} respectively. Gradient descent updates $\mathbf{U}$ and $\mathbf{V}$ iteratively until convergence to obtain a local minimum for the problem.

B. Parameter-Free Updating Rules

As we have mentioned in Section A, θ_{ik} and δ_{kj} are difficult to determine during the optimization process, since the nonnegativity constraint should also be maintained. Then we propose our multiplicative updating rules in Section III-B. In fact, it could be recognized as a parameter-free gradient descent method by the below settings

$$\theta_{ik} \leftarrow \frac{u_{ik}}{2(\mathcal{R}_\Omega(\mathbf{UV})\mathbf{V}^T)_{ik} + 2\lambda(\mathbf{WU})_{ik}}$$
$$\delta_{kj} \leftarrow \frac{v_{kj}}{2(\mathbf{U}^T\mathcal{R}_\Omega(\mathbf{UV}))_{kj}}, (k,j) \notin \Phi$$

We will have

$$u_{ik} = u_{ik} - \theta_{ik}\frac{\partial \mathcal{O}(\mathbf{U},\mathbf{V})}{\partial U_{ik}}$$
$$= u_{ik}\frac{(\mathcal{R}_\Omega(\mathbf{X})\mathbf{V}^T)_{ik} + \lambda(\mathbf{DU})_{ik}}{(\mathcal{R}_\Omega(\mathbf{UV})\mathbf{V}^T)_{ik} + \lambda(\mathbf{WU})_{ik}}$$
$$v_{kj} = v_{kj} - \delta_{kj}\frac{\partial \mathcal{O}(\mathbf{U},\mathbf{V})}{\partial V_{kj}}$$
$$= v_{kj}\frac{(\mathbf{U}^T\mathcal{R}_\Omega(\mathbf{X}))_{kj}}{(\mathbf{U}^T\mathcal{R}_\Omega(\mathbf{UV}))_{kj}}, (k,j) \notin \Phi$$

C. Varying Missing Rates and Error Rates

Figures 10 and 11 present the results by varying the missing rates and error rates over datasets. For most approaches, it is not surprising that imputation performance declines with the growth of the missing rate. The overall trends in both figures follow the results in Tables IV and VI discussed before. Over all the datasets, DLM and IterativeImputer continuously show accurate imputation results. However, as aforementioned, IterativeImputer cannot perform when other columns are also missing, thus showing worse results in high missing rates. SMFL outperforms all the baselines over all datasets, and maintains high performance over different missing rates. The results are analogous in Figure 11 with high error rates. The results also confirm the robustness of our proposed SMFL.

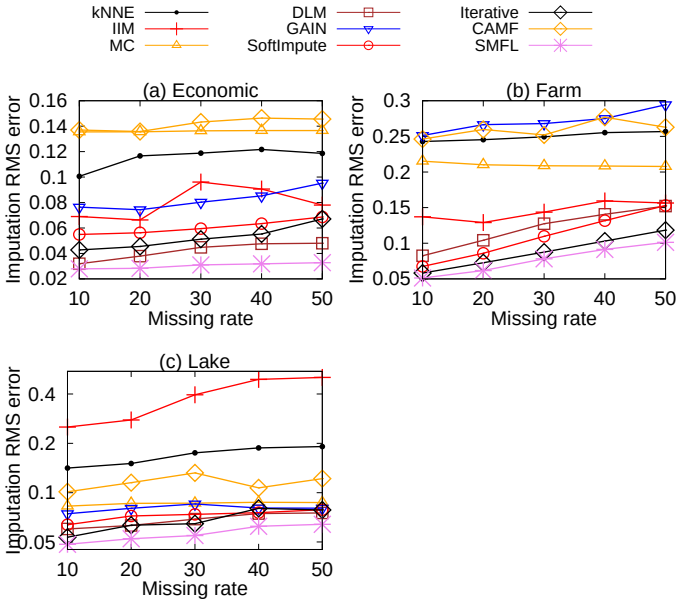

Fig. 10: Imputation RMS error by varying the missing rate (%)

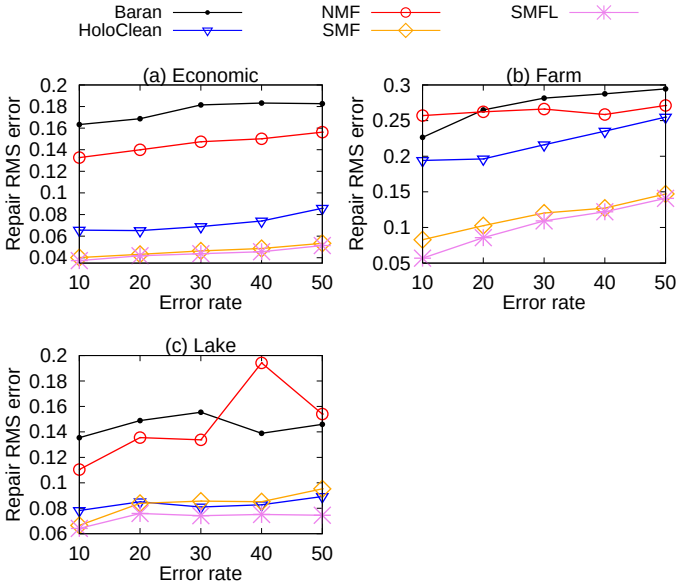

Fig. 11: Repair RMS error by varying the error rate (%)

D. Evaluation of Multiplicative Updating Rules

In Section III-B, we discuss the gradient descent updating rules for SMFL and then propose the multiplicative updating rules. We thereby conduct the comparison of these two updating methods over two datasets. The learning rates of gradient descent are set to 0.001, since we observe that a large learning rate for gradient descent cannot converge. We update $\mathbf{U}$ and $\mathbf{V}$ alternatively over 500 iterations with the same objective function according to Formula 10.

The results are plotted in Figure 12. As expected, in all the experiments, the loss updated by the multiplicative updating

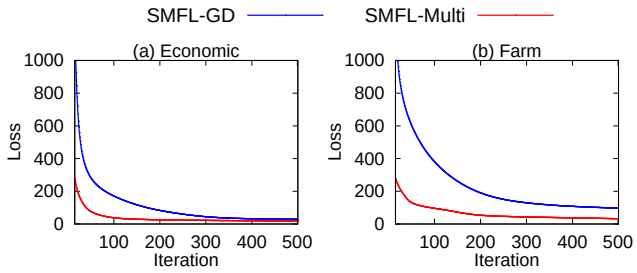

Fig. 12: Convergence curves

rules declines faster than that updated by gradient descent. With the same iteration, the loss updated by the multiplicative updating rules is lower, which indicates the effectiveness of the multiplicative updating rules. Since it can adjust the learning rates adaptively, while gradient descent requires a proper setting of a large number of learning rates, it is more suitable for updating $\mathbf{U}$ and $\mathbf{V}$ to approximate the original matrix $\mathbf{X}$.

Finally, another advantage of the multiplicative updating rules is that they guarantee the non-negativity of these two matrices. A straightforward example is illustrated in Figure 5, where SMF updated with gradient descent learned negative features, which is not allowed. This is also the reason why we prefer the multiplicative updating rules.