Supplemental material for:
Differentially Private Associative Co-clustering

Elena Battaglia* Ruggero G. Pensa*

A Fast-τCC algorithm

The iterative procedure to refine partition $\mathcal{R}$ while $\mathcal{C}$ is fixed, is described in Algorithm 1. The procedure uses two $K \times L$ matrices $\mathbf{U}$ and $\mathbf{V}$ to compute all σ values in a $n \times K$ matrix $\mathbf{\Sigma} = (\sigma_{ik})$, where $\sigma_{ik} = \sigma\left(\mathbf{p}_i, \mathbf{q}_k^{(t)}\right)$. More precisely:

$$\mathbf{\Sigma} = \mathbf{PC}(\mathbf{Q}^{(t-1)} \odot \mathbf{U}^{(t-1)} - \mathbf{V}^{(t-1)})^\top \tag{A.1}$$

where $\odot$ indicates the Hadamard matrix product, and

$$\mathbf{U}^{(t)} = \begin{bmatrix} \frac{1}{\sum_l q_{1l}^{(t)}} & \cdots & \frac{1}{\sum_l q_{Kl}^{(t)}} \\ \vdots & \ddots & \vdots \\ \frac{1}{\sum_l q_{1l}^{(t)}} & \cdots & \frac{1}{\sum_l q_{Kl}^{(t)}} \end{bmatrix} \tag{A.2}$$

$$\mathbf{V}^{(t)} = \begin{bmatrix} \frac{1}{\sum_l q_{1l}^{(t)}} & \cdots & \frac{1}{\sum_l q_{1l}^{(t)}} \\ \vdots & \ddots & \vdots \\ \frac{1}{\sum_l q_{Kl}^{(t)}} & \cdots & \frac{1}{\sum_l q_{Kl}^{(t)}} \end{bmatrix} \tag{A.3}$$

At the end, the algorithm also removes all empty clusters. Hence, from one iteration to another, the number of clusters may decrease and $\mathbf{R}^{(0)}$ and $\mathbf{C}^{(0)}$ can be initialized with random partitions using safely high values of K and L. In [1] the authors prove that Algorithm 1 always converges in a finite number of iterations. A similar procedure is defined for updating the column partition $\mathbf{C}^{(t)}$ when $\mathbf{R}$ is fixed. The overall Fast-τCC algorithm is given in Algorithm 2.

Algorithm 1: UpdateRowClusters($\mathbf{P}, \mathbf{C}, \mathbf{R}^{(0)}$)

Input: A $n \times m$ matrix $\mathbf{P}$, the column clustering $\mathbf{C}$, the initial row clustering $\mathbf{R}^{(0)}$
Result: the new row clustering $\mathbf{R}$

$t \leftarrow 1$; *changes* $\leftarrow$ True;
while *changes* **do**
  $\mathbf{Q}^{(t-1)} = \mathbf{R}^{(t-1)\top}\mathbf{PC}$;
  compute $\mathbf{U}^{(t-1)}$ and $\mathbf{V}^{(t-1)}$ as in Eq. A.2 and Eq. A.3;
  $\mathbf{\Sigma} = \mathbf{PC}(\mathbf{Q}^{(t-1)} \odot \mathbf{U}^{(t-1)} - \mathbf{V}^{(t-1)})^\top$;
  for $i = 1, \ldots, n$ **do**
    $k^\star(i) \leftarrow \arg\max_k(\sigma_{ik})$;
  end for
  compute $\mathbf{R}^{(t)}$ using $k^\star$;
  if $\mathbf{R}^{(t)} = \mathbf{R}^{(t-1)}$ **then**
    changes $\leftarrow$ False;
  end if
  remove empty clusters and update $\mathbf{R}^{(t)}$;
  $t \leftarrow t + 1$;
end while

B Theoretical aspects of DP-τCC

B.1 Composability of differentialy-private functions The overall level of privacy guaranteed by the composition of differentially private functions can be computed using the following theorems [3]:

THEOREM 1. (SEQUENTIAL COMPOSITION) *Let $f : \Omega \longrightarrow \mathcal{R}, g : \Omega \longrightarrow \mathcal{S}$ be two stochastic functions and let $h : \mathcal{R} \times \mathcal{S} \longrightarrow \mathcal{T}$ be a function. If f preserves ε_1-differential privacy and g preserves ε_2-differential privacy, then $h(f(D), g(D))$ preserves $(\varepsilon_1+\varepsilon_2)$-differential privacy.*

THEOREM 2. (PARALLEL COMPOSITION) *Let f and g be two ε-differentially private functions and $D_1, D_2 \in \Omega$ two disjoint datasets (i.e. $D_1 \cap D_2 = \emptyset$). Then function $h(D_1 \cup D_2) = (f(D_1), g(D_2))$ also preserves ε-differential privacy.*

THEOREM 3. (POST-PROCESSING) *Let $f : \Omega \longrightarrow \mathcal{R}$ be a randomized function preserving ε-differential privacy and let g be any function with domain $\mathcal{R}$. Then $g \circ f$ preserves ε-differential privacy.*

Theorems 1 and 2 state that the composition of several differentially private functions is differentially private as well and give a way to compute the overall privacy level of the computation: if two functions are applied sequentially, the overall level of privacy ensured is equal to the sum of the levels of privacy guaranteed by each function. Consequently, ε can be interpreted as the total privacy budget, and part of it can be allocated for any

*University of Turin, Italy, ruggero.pensa@unito.it

Algorithm 2: Fast-τCC($\mathbf{A}, t_{max}$)

Input: A $n \times m$ matrix $\mathbf{A}$, the maximum number of iterations t_{max}
Result: $\mathbf{R}$, $\mathbf{C}$

1 Initialize $\mathbf{R}^{(0)}$ and $\mathbf{C}^{(0)}$;
2 $t \leftarrow 1$; *changes* $\leftarrow$ True;
3 compute $\mathbf{P}$ from $\mathbf{A}$;
4 **while** *changes* **and** $t < t_{max}$ **do**
5 $\quad \mathbf{R}^{(t)} \leftarrow$ UpdateRowClusters($\mathbf{P}, \mathbf{C}^{(t-1)}, \mathbf{R}^{(t-1)}$);
6 $\quad \mathbf{C}^{(t)} \leftarrow$ UpdateColumnClusters($\mathbf{P}, \mathbf{R}^{(t)}, \mathbf{C}^{(t-1)}$);
7 $\quad$ **if** $\mathbf{R}^{(t)} = \mathbf{R}^{(t-1)}$ **and** $\mathbf{C}^{(t)} = \mathbf{C}^{(t-1)}$ **then**
8 $\quad\quad$ *changes* $\leftarrow$ False;
9 $\quad$ **end if**
10 $\quad t \leftarrow t + 1$;
11 **end while**

computation required to obtain the final outcome. On the other hand, Theorem 3 says that when an outcome r is computed in a differentially private way, any following transformation of this result is still differentially private and there is no need to allocate part of the privacy budget for it.

B.2 Theorems and proofs

THEOREM 4. *Algorithm 1 in [2] preserves ε-differential privacy.*

Proof. First of all, we will prove that Algorithms 2 and 3 in [2] preserve ε-differential privacy, when applied with privacy budget ε. Algorithm 2 in [2] computes the new contingency table using the Laplace mechanism, with privacy parameter $\frac{\varepsilon_1}{N}$ and sensitivity equal to 1, as derived in Equation 4.11 in [2]. Hence, the contingency table is computed preserving ε_1-differential privacy. Notice that the Laplace mechanism could return negative values, which are not allowed in a contingency matrix: rows 6–10 of the algorithm fix this problem by setting to zero all the negative entries in the matrix. This operation does not access the data, the post-processing theorem ensures that it does not consume the privacy budget (see Theorem 3). Algorithm 3 in [2], instead, assigns each row (or column) to the closest cluster prototype using the exponential mechanism with sensitivity as derived in Equation 4.10 in [2] and privacy budget ε, preserving ε-differential privacy. Hence, thanks to the parallel composition theorem, the overall cluster assignment procedure preserves ε-differential privacy (see Theorem 2).

Let now consider the overall method, described by Algorithm 1 in [2]. We skip initially lines 2 and 3 and discuss from line 4 on. The initialization of $\mathbf{R}^{(0)}$ and $\mathbf{C}^{(0)}$ does not consume the privacy budget, because it does not depend on the data matrix $\mathcal{A}$. Instead, the computation of $\mathbf{C}^{(1)}$ preserves ε_2-differential privacy. When $t = 1$ (the first iteration of lines 8 to 16), Algorithm 2 in [2] is called twice (with privacy budget ε_1, and Algorithm 3 in [2] is called once with privacy budget ε_1. Overall, after the first iteration, the algorithm has consumed $2(\varepsilon_1+\varepsilon_2)$, according to Theorem 1. Each successive iteration consumes exactly the same privacy budget. Consequently, the overall algorithm consumes $2\mathcal{I}(\varepsilon_1 + \varepsilon_2)$ privacy budget. According to line 3 of Algorithm 1 in [2], $\varepsilon_1+\varepsilon_2 = \varepsilon'$, which, in its turn, is equal to $\frac{\varepsilon}{2\mathcal{I}}$ (line 2). In conclusion, the overall privacy level of the algorithm is $2\mathcal{I}\varepsilon' = 2\mathcal{I}\frac{\varepsilon}{2\mathcal{I}} = \varepsilon$. □

C Additional experiments and results

C.1 Additional results on real-world datasets

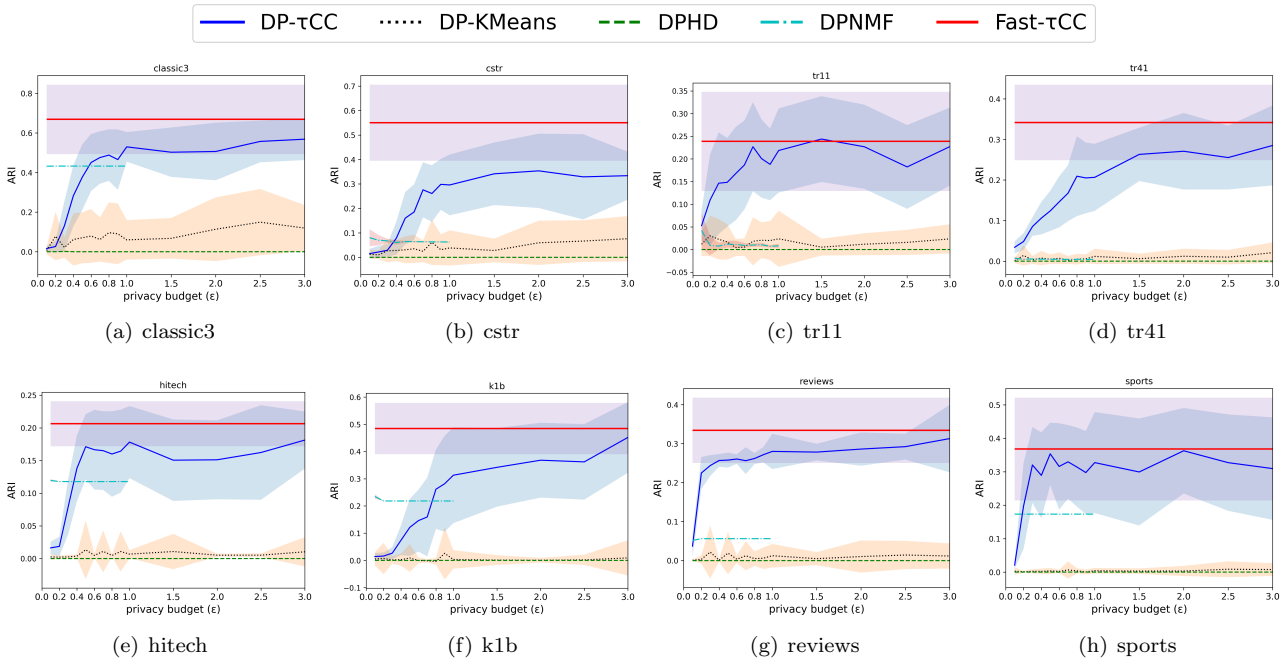
The results in terms of Adjusted Rand Index (ARI) of the experiments discussed in Section 5.1 in [2] are reported in Fig. 1. Fig. 2, instead, reports the results of the experiments described in Section 5.2 in [2] for the other four real-world datasets.

C.2 Analysis of the budget allocation

At the end of Section 4.1 in [2], we argued that the majority of the overall privacy budget ε available during each iteration can be devoted to the computation of the cluster assignment. To motivate this decision experimentally, we measure the objective function value ($\hat{\tau}_{R|C}$) on the privacy-preserving contingency table released by DP-τCC by varying both ε and the fraction of it devoted to the cluster assignment (Algorithm 3 in [2]) and to the computation of the contingency table (Algorithm 2 in [2]). To this purpose, we introduce a parameter $h \in (0, 1)$ quantifying the fraction of ε allocated to the cluster assignment. According to this choice, during each iteration of DP-τCC, the amount of privacy budget for the cluster assignment will be $h \cdot \varepsilon$, while $(1 - h) \cdot \varepsilon$ is the privacy budget devoted to the computation of the contingency table. Fig. 3 reports the results obtained on the eight real-world datasets. According to the plots, it is clear that, although $h = 0.9$ (the value we retain in all other experiments) is not always the optimal choice, the general trend for all eight datasets is that higher values of h lead to higher values of the objective function.

C.3 Experiments on synthetic matrices

We execute all the algorithms on synthetic matrices of increasing size generated by using the `make_biclusters` func-



(a) classic3 (b) cstr (c) tr11 (d) tr41

(e) hitech (f) k1b (g) reviews (h) sports

Figure 1: ARI reported for all competitors on real-world datasets and for increasing values of ε. Results for DPNMF can be computed for $\varepsilon < 1$ only.

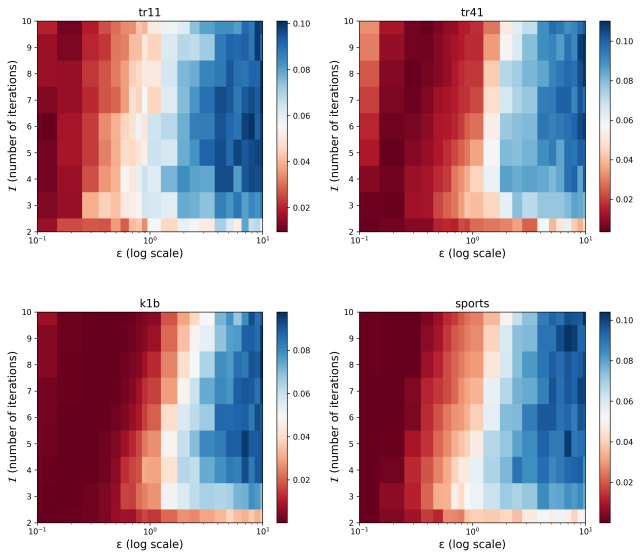


Figure 2: $\hat{\tau}_{R|C}$ for increasing number of iterations (from 2 to 10) and increasing values of ε (from 0.1 to 10, in log scale).

tion of scikit-learn[1]. More specifically, the matrices have all the same number of rows (1 000) and an increasing number of columns (from 10 to 10 000). All matrices have three block diagonal co-clusters with random values ranging from 1 to 10. Gaussian noise with standard deviation $\sigma = 3.0$ is added to the matrix and all rows and columns are eventually shuffled.

All methods are applied with increasing values of the privacy budget from $\varepsilon = 0.1$ to $\varepsilon = 3$. Since DPNMF satisfies (ε, δ)-differential privacy (with $\varepsilon < 1$), only for this algorithm, we set $\delta = 10^{-5}$ and let ε vary between 0.1 and 0.99999.

The results are reported in Fig 4 (for the NMI) and Fig. 5 (for the ARI) along two different perspectives. Fig 4(a)-(d) and Fig 5(a)-(d) show the behavior of the NMI and ARI for increasing values of ε in four different matrices, while Fig. 4(e)-(h) and Fig. 5(e)-(h) report the NMI and ARI for increasing matrix size for four different values of ε. All plots show that the performances of DP-KMeans are only acceptable for low-dimensional data, and DPHD does not perform well for small values of ε with high-dimensional data. DPNFM, instead, achieves rather high values of NMI and ARI for high-dimensional data, but we recall that only one part of this algorithm is actually privacy-preserving. DP-τCC, instead, is more stable, with an average NMI and ARI always close to 0.8 and 0.6, respectively. For the sake of completeness, we also report the running

[1] `https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_biclusters.html`

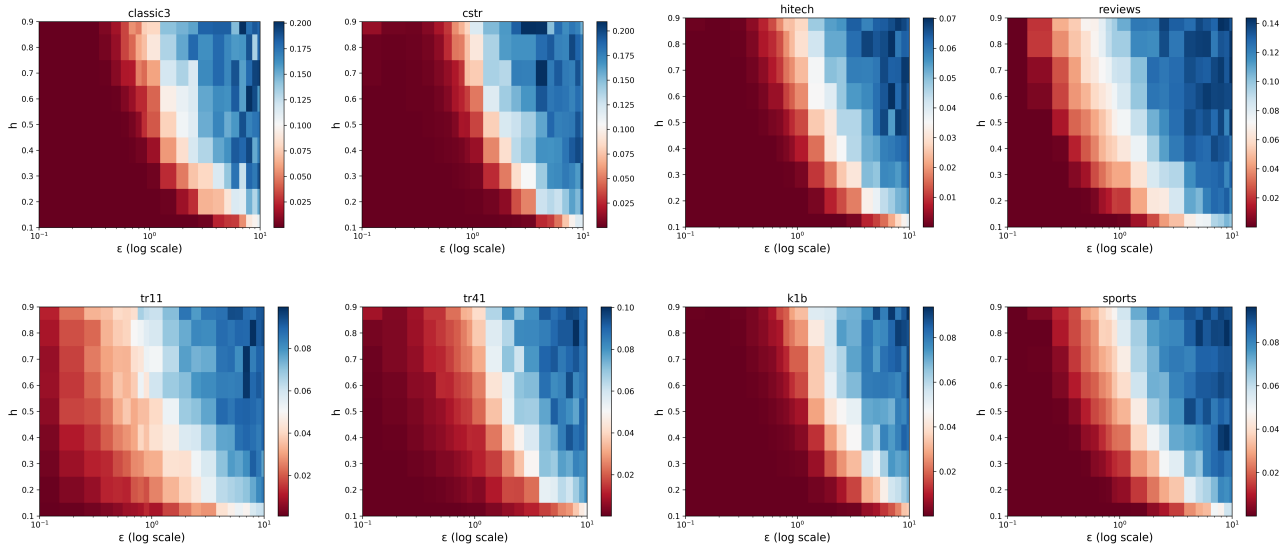
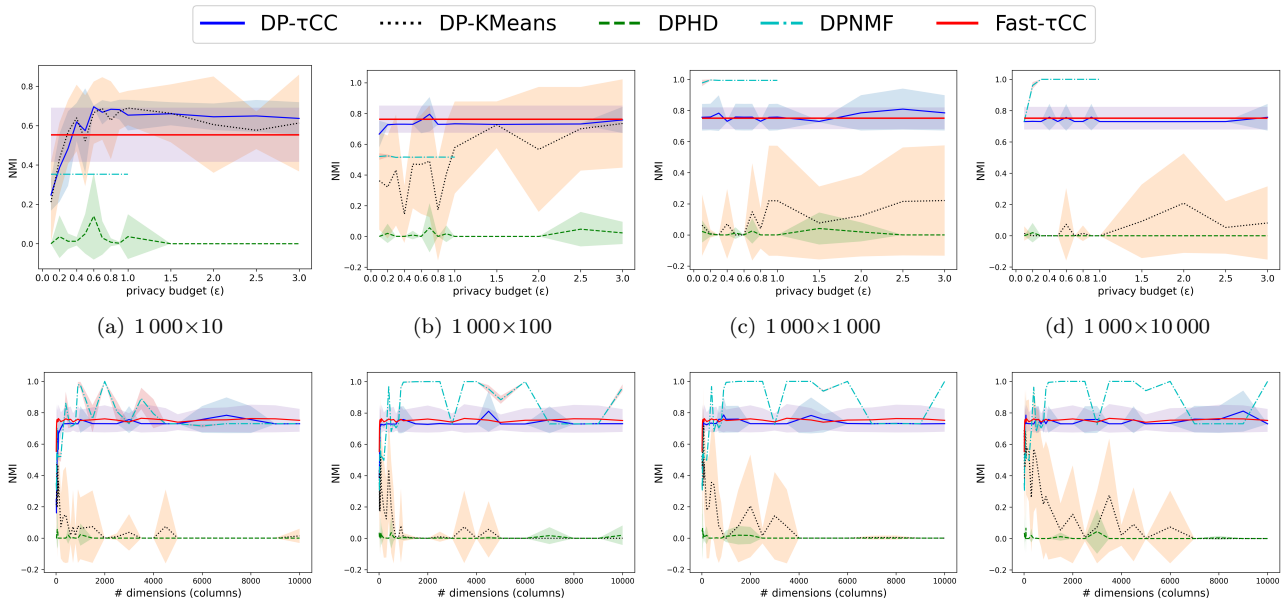


Figure 3: $\hat{\tau}_{R|C}$ for increasing values of h (from 0.1 to 0.9) and increasing values of ε (from 0.1 to 10, in log scale).



(a) 1 000×10 (b) 1 000×100 (c) 1 000×1 000 (d) 1 000×10 000

(e) $\varepsilon = 0.1$ (f) $\varepsilon = 0.2$ (g) $\varepsilon = 0.5$ (h) $\varepsilon = 0.99999$

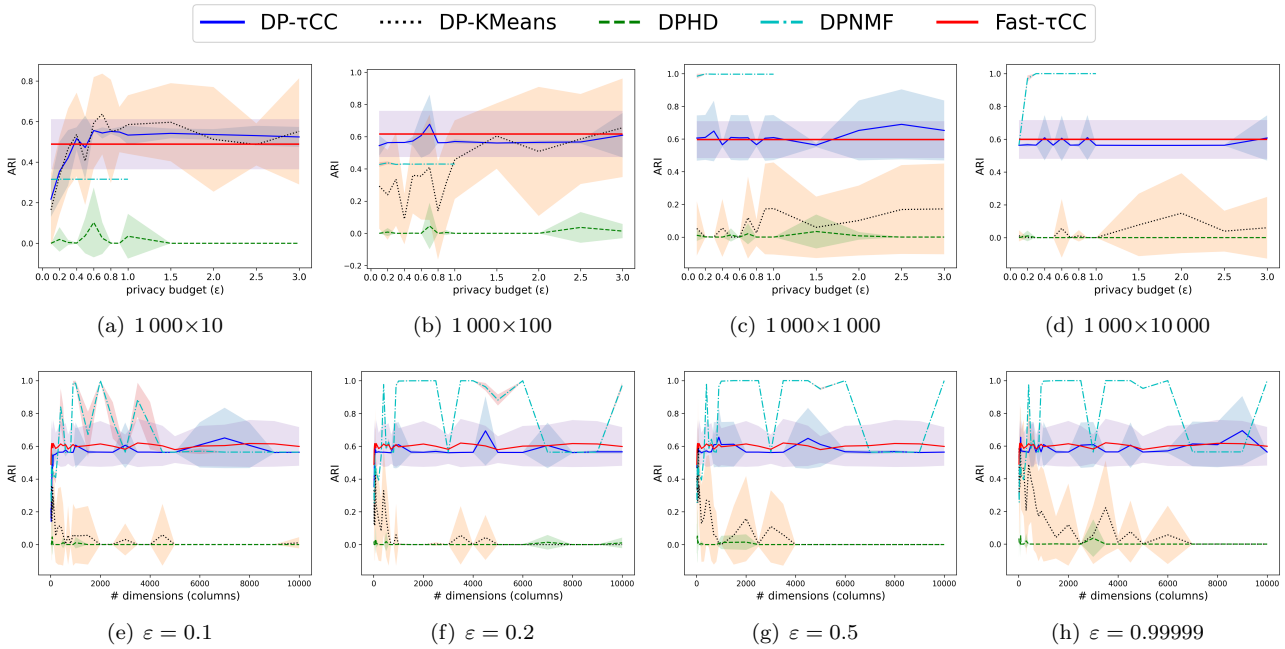
Figure 4: NMI reported for all competitors on synthetic matrices with increasing size (top) and for increasing values of ε (bottom). Results for DPNMF can be computed for $\varepsilon < 1$ only.

time of the algorithms in Fig. 6. While DP-KMeans complexity is linear in the number of dimensions, all other methods have better time performances, with the two co-clustering algorithms resulting way faster than the other competitors.

References

[1] E. Battaglia, F. Peiretti, and R. G. Pensa, *Fast parameterless prototype-based co-clustering*, Mach. Learn., 113 (2024), pp. 2153–2181.

[2] E. Battaglia and R. G. Pensa, *Differentially private associative co-clustering*, in Proceedings of the 2025



(a) 1 000×10 (b) 1 000×100 (c) 1 000×1 000 (d) 1 000×10 000

(e) $\varepsilon = 0.1$ (f) $\varepsilon = 0.2$ (g) $\varepsilon = 0.5$ (h) $\varepsilon = 0.99999$

Figure 5: ARI reported for all competitors on synthetic matrices with increasing size (top) and for increasing values of ε (bottom). Results for DPNMF can be computed for $\varepsilon < 1$ only.

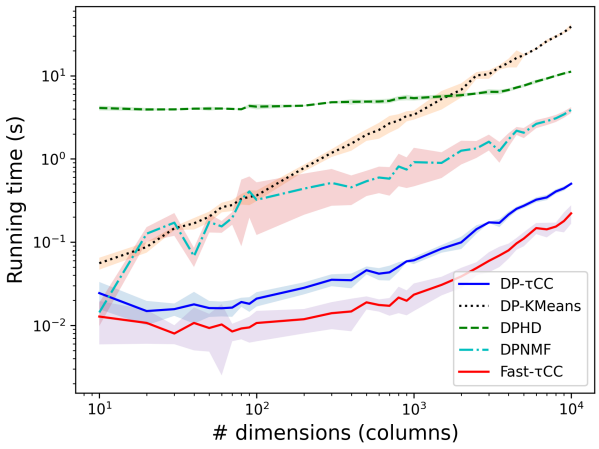


Figure 6: Running time in seconds for all competitors on synthetic matrices of increasing column number ($\varepsilon = 0.99999$ for all differentially private methods).

SIAM International Conference on Data Mining, SDM 2025, Alexandria, VA, USA, May 1-3, 2025, SIAM, 2025, pp. 1–9.

[3] C. Dwork, A. Roth, et al., *The algorithmic foundations of differential privacy.*, Found. Trends Theor. Comput. Sci., 9 (2014), pp. 211–407.