Submitted to *INFORMS Journal on Computing*
manuscript JOC-2016-10-OA-211.R3.Supp

Online Supplement to "Transit Pattern Detection Using Tensor Factorization"

Bowen Du
State Key Lab of Software Development Environment, Beihang University, dubowen@buaa.edu.cn

Wenjun Zhou
Department of Business Analytics and Statistics, University of Tennessee, wzhou4@utk.edu

Chuanren Liu
Decision Sciences and MIS Department, Drexel University, chuanren.liu@drexel.edu

Yifeng Cui
State Key Lab of Software Development Environment, Beihang University, naskcyf@buaa.edu.cn

Hui Xiong
Management Science and Information Systems Department, Rutgers University, hxiong@rutgers.edu

This document provides additional details for the main paper.

---

1. Tensor Construction and Decomposition: A Numeric Example

A numeric example of trip summaries, as described in Section 3.3 of the paper, is shown in Figure 8. In this example, we assume that there are $M = 6$ stations. Therefore, the OD matrix Y has a 6×6 dimension, and the OTD tensor has a $6 \times 6 \times 6$ dimension. Each number in the OD matrix Y represents the number of direct trips between stations. For example, there are $Y_{1,3} = 5,858$ trips from s_1 to s_3 during the study period. Each number in the OTD tensor X represents the number of trips between stations with a transfer in between. For example, there are $X_{4,2,5} = 78$ trips from s_4 to s_5 with a connection at s_2 during the study period.

After tensor decomposition, as described in Section 4.2 of the paper, Figure 9 shows example results based on the data in Figure 8, assuming that we decided to find $K_1 = 3$

	s_1	s_2	s_3	s_4	s_5	s_6
s_1	--	400	5858	6528	472	4090
s_2	235	--	5651	6280	561	3953
s_3	177	377	--	4681	352	2437
s_4	1724	1745	1091	--	1863	445
s_5	84	294	4448	4908	--	2555
s_6	1849	1850	686	386	1970	--

(a) The OD Matrix Y

$t = s_1$	s_1	s_2	s_3	s_4	s_5	s_6
s_1	--	303	306	68	1261	324
s_2	141	--	303	65	1247	321
s_3	1504	1336	--	141	211	44
s_4	396	364	75	--	156	41
s_5	1573	1396	76	145	--	44
s_6	438	389	52	76	49	--

$t = s_2$	s_1	s_2	s_3	s_4	s_5	s_6
s_1	--	126	168	42	585	30
s_2	48	--	166	39	579	29
s_3	409	368	--	62	89	9
s_4	59	62	35	--	78	6
s_5	431	387	52	65	--	9
s_6	66	62	23	22	29	--

$t = s_3$	s_1	s_2	s_3	s_4	s_5	s_6
s_1	--	323	321	71	1336	356
s_2	152	--	318	68	1320	353
s_3	1625	1442	--	150	225	48
s_4	433	398	79	--	165	45
s_5	1699	1507	78	154	--	48
s_6	479	425	56	82	51	--

$t = s_4$	s_1	s_2	s_3	s_4	s_5	s_6
s_1	--	308	309	69	1278	331
s_2	144	--	307	66	1263	328
s_3	1531	1359	--	143	214	45
s_4	404	372	76	--	158	42
s_5	1601	1420	76	147	--	45
s_6	447	397	53	77	50	--

$t = s_5$	s_1	s_2	s_3	s_4	s_5	s_6
s_1	--	284	291	65	1187	292
s_2	131	--	288	62	1173	289
s_3	1384	1230	--	132	198	40
s_4	359	331	71	--	148	37
s_5	1448	1285	73	136	--	40
s_6	397	353	49	70	47	--

$t = s_6$	s_1	s_2	s_3	s_4	s_5	s_6
s_1	--	133	173	42	610	41
s_2	52	--	171	40	604	40
s_3	449	404	--	65	93	11
s_4	72	73	37	--	81	7
s_5	472	424	53	68	--	11
s_6	79	74	24	24	30	--

(b) The OTD Tensor X (shown by transfer slices)

Figure 8 **Trip summaries: a numeric example**

	O_1	O_2	O_3
s_1	0.90	0.10	0.00
s_2	0.89	0.06	0.05
s_3	0.10	0.85	0.05
s_4	0.08	0.04	0.88
s_5	0.10	0.90	0.00
s_6	0.00	0.05	0.95

A (K_1=3)

	D_1	D_2	D_3	D_4
s_1	0.00	0.03	0.00	0.97
s_2	0.10	0.00	0.05	0.85
s_3	0.20	0.00	0.79	0.01
s_4	0.00	0.10	0.83	0.07
s_5	0.86	0.10	0.00	0.04
s_6	0.00	0.95	0.05	0.00

B (K_2=4)

	T_1	T_2
s_1	0.87	0.13
s_2	0.05	0.95
s_3	0.96	0.04
s_4	0.89	0.11
s_5	0.78	0.22
s_6	0.08	0.92

T (K_3=2)

(a) The Station-Region Loadings

	D_1	D_2	D_3	D_4
O_1	101.07	4107.88	7612.97	27.08
O_2	15.55	2203.24	5403.81	11.87
O_3	2303.05	8.49	21.92	2002.59

(b) The U Flows

V[T_1]	D_1	D_2	D_3	D_4
O_1	1706.44	431.68	22.30	8.32
O_2	11.72	8.64	33.45	2007.70
O_3	37.20	13.61	58.61	433.73

V[T_2]	D_1	D_2	D_3	D_4
O_1	698.98	11.77	42.78	20.22
O_2	7.45	5.21	40.99	411.59
O_3	31.40	2.06	17.58	25.65

(c) The V Flows

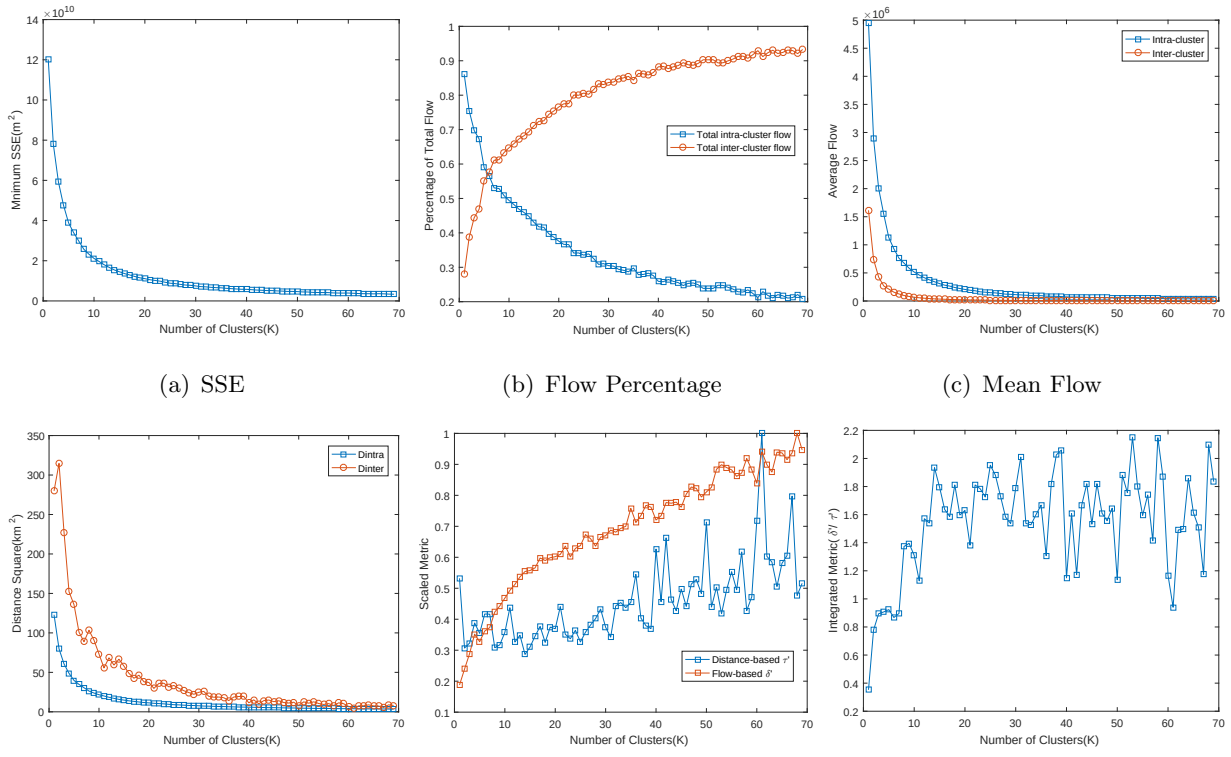
Figure 9 **A running example of decomposition results based on the data in Figure 8**

origin regions, $K_2 = 4$ destination regions, and $K_3 = 2$ transfer regions. The station-region assignments are shown in Figure 9(a). We can see that, given $M = 6$ stations, the A, B, T matrices each has 6 rows. By looking at the magnitude of the numbers in these matrices, we may decide that $O_1 = \{s_1, s_2\}$, $O_2 = \{s_3, s_5\}$, and $O_3 = \{s_4, s_6\}$ make the origin regions. Similarly, we may decide that $D_1 = \{s_5\}$, $D_2 = \{s_6\}$, $D_3 = \{s_3, s_4\}$, and $D_4 = \{s_1, s_2\}$ are destination regions; and that $T_1 = \{s_1, s_3, s_4, s_5\}$ and $T_2 = \{s_2, s_6\}$ are transfer regions. As shown in Figure 9(b), the corresponding direct flows information among the O/D regions are saved in the U matrix. Since there are 3 origin regions and 4 destination regions, the U matrix has a 3×4 dimension. The quantities in this matrix represent the traffic flow among each pair of origin and destination regions. For example, there were $U_{1,2} \approx 4,107.88$ trips from origin region O_1 to destination region D_2. If we use $4,000$ as the cutoff, we would identify $O_1 \to D_2$, $O_1 \to D_3$, and $O_2 \to D_3$ as the most significant OD transit patterns. The corresponding indirect flows information are saved in the V tensor, as shown in Figure 9(c).

Since there are 3 origin regions, 2 transfer regions, and 4 destination regions, the V tensor has a $3 \times 2 \times 4$ dimension. The quantities in this tensor represent the indirect traffic flows among origin, transfer, and destination regions. For example, there were $V_{1,1,2} \approx 431.68$ trips from origin region O_1 to destination region D_2 via transfer region T_1. If we use $1,000$ as the cutoff, we would identify $O_1 \to T_1 \to D_1$ and $O_2 \to T_2 \to D_4$ as the most significant OTD transit patterns.

2. Clustering Results Following Luo et al. (2017)

Working on a similar problem, Luo et al. (2017) proposed a k-means based clustering method to group stations based on both spatial distance and traffic flow. In this section, we provide the corresponding clustering-based results using our data.



(a) SSE (b) Flow Percentage (c) Mean Flow

(d) Distance (e) Metrics (f) Parameter Selection

Figure 10 **Clustering of our data following the methodology proposed in Luo et al. (2017). These results correspond to Figure 3 in Luo et al. (2017). (a) SSE decreases exponentially as the number of clusters increases; (b) variation in both total intra-cluster and total inter-cluster flows; (c) variation in both average intra-cluster and average inter-cluster flows; (d) intra-cluster and inter-cluster flow measures; (e) illustration of two scaled metrics, and (f) integrated metric that reaches the maximum value when the number of the cluster is around $k = 53$.**

According to Luo et al. (2017), the number of clusters is a parameter determined by maximizing the ratio between the flow-based metric and distance-based metric. We found that for our dataset, this ratio reaches its maximum when the number of clusters is set at around 53, as shown Figure 10(f). The corresponding top 15 hotspots (i. e., clusters with high traffic flows) are shown in Figure 5(d) in color.

After clustering with $k = 53$, we ranked the clusters by traffic volume and took the top 15 regions for comparison. Note that Luo et al. (2017) does not separate origins from destinations, and transfers were not considered. Therefore, the traffic volume includes both arrivals (i.e., destination stations) and departures (origin stations). For this comparison, we used the same day's data (April 23, 2015). Since Luo et al. (2017) does not slice data by time periods of the day, for our TTF based method, we did not slice time slots, either. More specifically, we used all data from 5:00 a.m. to 11:00 p.m. on that day (6,556,397 trips) to perform this comparison.

2.1. Quantitative Comparison

To compare the traffic clustering metric, the total, maximum, and minimum traffic volumes among the top 15 hotspot regions by each method are reported in Table 4. Specifically, Luo et al. (2017)'s method does not separate origin, destination, or transfers, so the last row in the table reports the number of arriving and departing travel segments for stations in these hotspot regions. For TTF, we reported the number of departing trips from stations in the origin hotspots, number of transferring trips at stations in the transfer hotspots, and the number of arriving trips at stations in the destination hotspots. To make the comparison fair for TTF, we aggregated these numbers into OD summaries as $OD = (O + D)/2 + T$. We can see that hotspots discovered by the TTF method can cover far more traffic volumes than Luo et al. (2017)'s method.

Table 4 **Total, maximum, and minimum traffic volumes in the reported clusters.**

Method	Hotspot Type	Total	Maximum	Minimum
TTF	O	1,180,280	136,400	38,200
	T	1,740,320	178,200	75,400
	D	1,565,860	174,000	42,600
	OD	**2,583,470**	**283,400**	**109,300**
Luo et al. (2017)	OD	1,472,842	135,678	67,784

To compare the spatial clustering metric, we also calculated distance measures for these hotspot clusters. For computational convenience, we calculated the Euclidean distance based on latitude and longitude value pairs instead of actual travel distance. The withinness are measured as the distances between any two stations in the same cluster, and the betweenness are measured as the distances between centroids of two different clusters. Results are outlined in Table 5. Good clustering results should demonstrate low withinness (i.e., lower within-cluster distances) and high betweenness (i.e., higher within-cluster distances). From Table 5, we can see that the withinness and betweenness metrics are all very close, indicating that clusters found by the TTF method are as spatially clustered as the Luo et al. (2017) method. The best results in each type of distance is shown in bold. We can see that the TTF method reaches the best withinness for transfer hotspots, and the best minimum betweenness for destination hotspots. Overall, the TTF method shows competitive results in terms of spatial clustering.

Table 5 Maximum and minimum distance between two stations in the same cluster.

	Hotspot	Withinness		Betweenness	
Method	Type	Maximum	Minimum	Maximum	Minimum
TTF	O	0.1107	0.0048	0.2752	0.0403
	T	**0.0892**	**0.0035**	0.2474	0.0483
	D	0.1028	0.0037	0.2851	**0.0599**
Luo et al. (2017)	O+D	0.1004	0.0047	**0.2924**	0.0405

2.2. Qualitative Comparison

As shown in Figure 5, TTF finds separate O/T/D regions simultaneously, while Luo et al. (2017) only finds one set of clusters. In Section 5.2 of the manuscript, we discussed examples when O/T/D regions differ and why they are meaningful.

The method by Luo et al. (2017) considers station-level traffic volumes, but does not separate the O/T/D categories of passenger trips. The objective was to maximize the ratio between the flow-based metric and distance-based metric. In contrast, TTF provides flow summaries among the discovered O/T/D regions, and is designed to easily allow for different flow patterns by temporal slices. TTF's flow decomposition objective allows for incorporation of traffic flow patterns among O/T/D regions, making these regions more meaningful. For example, *Xizhimen* is a transportation hub, while *Xidan* is a popular

business zone. They both have high passenger flow most of the time. Therefore, Luo et al. (2017)'s method grouped them into the same cluster. However, when considering passenger transfers, *Xizhimen* is far greater than *Xidan*.

3. External Validation with Feng and Jianfeng (2016)

Feng and Jianfeng (2016) reported spatial distribution of the most active bus stations in Beijing, which will serve as our external benchmark. Specifically, Feng and Jianfeng (2016) identified stations with high boarding volume and high transfer proportions, and visually summarized major regions. Table 6 tabulates a comparison of TTF hotspots with those reported in Feng and Jianfeng (2016).

Table 6 Correspondence among regions identified by TTF and Feng and Jianfeng (2016)

		TTF			**Feng and Jianfeng (2016)**	
	Region	**Origin**	**Transfer**	**Destination**	**High Boarding**	**High Transfer**
1	Zhongguancun	Zhongguancun			Zhongguancun	
2	Dongzhimen		Dongzhimen	Dongzhimen	Dongzhimen	
3	Deshengmen		Deshengmen		Deshengmen	Deshengmen
4	Xizhimen		Xizhimen		Xizhimen	
5	Liuliqiao, Beijing West	Liuliqiao, Beijing West	Liuliqiao, Beijing West	Liuliqiao, Beijing West	Xisanhuan	Liuliqiao, Liuliqiao North
6	Guomao, Beijing Station	Guomao	Guomao, Beijing Station	Guomao, Beijing Station	CBD	
7	Xidan	Xidan		Xidan	Jinrongjie	
8	Gongzhufen		Gongzhufen			Gongzhufen
9	Xinfadiqiao	Xinfadiqiao	Xinfadiqiao	Xinfadiqiao		Xinfadiqiao

We can see that our results are generally consistent with Feng and Jianfeng (2016). In particular, with top 9 origin and transfer regions, we were able to find 7 out of the 8 regions with high boarding volume and all regions with high transfer proportions.

Note that TTF reports O/T/D regions by decomposing traffic flows, while results in Feng and Jianfeng (2016) were reported at the station level. The high-volume region TTF missed was *Nanzhongzhou*, where boarding volumes were high at several stations, but when aggregated into the nearby region, the traffic is no longer high compared to other regions.

4. Implementation Details

Here we provide more details about the model training process summarized in Algorithm 1.

Updating A, B, and T

The updating of A, B, and T is similar to a quadratic programming problem with simplex constraints, which can be efficiently optimized with the projected gradient method (Lin 2007).

To provide the gradients, we define

$$\kappa_o : V \in \mathbb{R}^{K_1 \times K_3 \times K_2} \mapsto \kappa_o(V) \in \mathbb{R}^{K_1 \times K_3 K_2}$$
$$\kappa_t : V \in \mathbb{R}^{K_1 \times K_3 \times K_2} \mapsto \kappa_t(V) \in \mathbb{R}^{K_3 \times K_1 K_2}$$
$$\kappa_d : V \in \mathbb{R}^{K_1 \times K_3 \times K_2} \mapsto \kappa_d(V) \in \mathbb{R}^{K_2 \times K_1 K_3}$$

where the matrix $V_{k_1 **}$ in the tensor $\mathcal{V}$ is vectorized as row $\kappa_o(V)_{k_1 *}$ in matrix $\kappa_o(V)$. Similarly, the matrix $V_{*k_2 *}$ is vectorized as row $\kappa_t(V)_{k_2 *}$, and the matrix V_{**k_3} is vectorized as row $\kappa_d(V)_{k_3 *}$.

It follows that

$$\frac{\partial \mathcal{J}}{\partial A} = \sum_h \left((AU^{(h)}B^\top - Y^{(h)}) \cdot B \cdot (U^{(h)})^\top + \kappa_o(V^{(h)} \times_o A \times_t T \times_d B - X^{(h)}) \cdot \kappa_o(V^{(h)} \times_t T \times_d B)^\top \right)$$
$$\frac{\partial \mathcal{J}}{\partial B} = \sum_h \left((AU^{(h)}B^\top - Y^{(h)}) \cdot AU^{(h)} + \kappa_d(V^{(h)} \times_o A \times_t T \times_d B - X^{(h)}) \cdot \kappa_d(V^{(h)} \times_o A \times_t T)^\top \right)$$
$$\frac{\partial \mathcal{J}}{\partial T} = \sum_h \kappa_t(V^{(h)} \times_o A \times_t T \times_d B - X^{(h)}) \cdot \kappa_t(V^{(h)} \times_o A \times_d B)^\top$$
$$\frac{\partial \Omega}{\partial A} = \left(\frac{1}{2} A^\top (D - G) + \frac{1}{2} A^\top (D - G)^\top \right)^\top$$
$$\frac{\partial \Omega}{\partial B} = \left(\frac{1}{2} B^\top (D - G) + \frac{1}{2} B^\top (D - G)^\top \right)^\top$$
$$\frac{\partial \Omega}{\partial T} = \left(\frac{1}{2} T^\top (D - G) + \frac{1}{2} T^\top (D - G)^\top \right)^\top$$

For example, suppose the current solution is A. We compute the new solution A^{new} using the projected gradient descent:

$$A^{new} \leftarrow \text{proj}_{splx}(A - \beta^r \frac{\partial \mathcal{J}}{\partial A}),$$

where r is the first non-negative integer for which

$$\mathcal{J}(A^{new}) - \mathcal{J}(A) \leq \sigma \frac{\partial \mathcal{J}}{\partial A}(A^{new} - A). \tag{1}$$

The operator $\text{proj}_{splx}(A)$ projects each row of matrix A to the simplex such that $A_{mk} \geq 0$ and $\sum_k A_{mk} = 1$. The parameters β and σ are specified such that $0 < \beta < 1$ and $0 < \sigma < 1$. Note that, Eq. (1) can be reformulated and evaluated efficiently without computing $\mathcal{J}(\cdot)$ explicitly.

Updating $\mathcal{U}$ and $\mathcal{V}$

The following gradients are used to update $\mathcal{U}$ and $\mathcal{V}$.

$$\frac{\partial \mathcal{J}}{\partial U^{(h)}} = A^\top (A U^{(h)} B^\top - Y^{(h)}) B$$
$$\frac{\partial \mathcal{J}}{\partial V^{(h)}} = (V^{(h)} \times_o A \times_t T \times_d B - X^{(h)}) A^\top T^\top B^\top$$

As can be seen, the solution $U^{(h)}$ and $V^{(h)}$ for each time slot can be computed independently as a non-negative least square problem. As an example, consider $U^{(h)}$ for the time slot h. The updating procedure computes the new solution as

$$U^{(h),new} \leftarrow \text{proj}_{nn}(U^{(h)} - \beta^r \frac{\partial \mathcal{J}}{\partial U^{(h)}}),$$

where r is the first non-negative integer for which

$$\mathcal{J}(U^{(h),new}) - \mathcal{J}(U^{(h)}) \leq \sigma \frac{\partial \mathcal{J}}{\partial U^{(h)}} (U^{(h),new} - U^{(h)}). \tag{2}$$

The operator $\text{proj}_{nn}(\cdot)$ projects the entries to be non-negative.

References

Feng C, Jianfeng L (2016) Characteristics of bus passenger flow based on ic card data: A case study in beijing. *Urban Transport of China* 14(1), URL `http://dx.doi.org/10.13813/j.cn11-5141/u.2016.0108`.

Lin Cb (2007) Projected gradient methods for nonnegative matrix factorization. *Neural Computation* 19(10):2756–2779.

Luo D, Cats O, van Lint H (2017) Constructing transit origindestination matrices with spatial clustering. *Transportation Research Record: Journal of the Transportation Research Board* 2652:39–49, URL `http://dx.doi.org/10.3141/2652-05`.