Response to Reviewer Comments

Collaborative Imputation for Multivariate Time Series with Convergence Guarantee

We thank the (meta) reviewers for insightful comments and constructive suggestions towards improvement of our submission. Below is the summary of changes in the revised manuscript, referring to the comments from each reviewer. The comments and changes are highlighted in red for Reviewer 1, magenta for Reviewer 3, and blue for Reviewer 4.

To Meta Reviewer

M-C1: Dear authors, the reviewers agree that the paper displays novel ideas that could be interesting to the ICDE audience. Yet, the current manuscript requires a major revision to address major concerns on clarity, assumptions, and experiments. Please go through the opportunities for improvement by each reviewer and in particular address the following points.

1. Highlight technical novelty along with more detailed explanation of the foundations and assumptions, relationship between inter- and intra-temporal models.

2. Experiments: Conduct a scalability discussion paired with limitations and reference to real world scenarios. Addition of an experiment with real world errors and scale would benefit the manuscript. Detailed discussion of hyperparameters is necessary.

3. The authors should clearly discuss the benefits of each individual algorithm and provide a recommendation framework on when to choose which one.

Reply: Thank you for highlighting the key points to improve our work. We have addressed all the reviewers' comments and revision requests. Responses to specific points are as follows.

1. (1) For the concern on "technical novelty", we add new technical contributions about the streaming computation and parameter determinations strategies in the new Sections VI-A and VI-B respectively, as well as clarifying our existing contributions in Section I-C, as requested in R4O1. (2) For the concern on "foundations and assumptions", we add more detailed explanations regarding the assumptions made during the convergence guarantee in Assumption 1 in Section V-D, according to R3O1. (3) For the concern on "relationship between inter- and intra-temporal models", we add more explanations about the intertemporal and intratemporal dependency models in Section II-A, and report all the dependency models of inratemporal and intertemporal patterns used in experiments in Appendix.L in [1], following the comments in R1O2.

2. (1) For the concern on "scalability discussion", we use the additional large-scale Weer [62] dataset with real-world missing values to study the time costs in Table III, the memory consumption in Table V, and the streaming imputation in Table IV, where our methods show the reasonable scalability and practicability, as commented in R1O5, R3O3 and R4O5. (2) For the concern on "limitations and reference to real world scenarios", we clarify that our methods do not necessarily assume a strong correlation between all attributes, with the detailed explanations, in Section II-A, and report all the dependency models in Appendix.L in [1], as requested in R4O2. (3) For the concern on "real world errors and scale", we add two real-world incomplete datasets IMU [49] and Weer [62], where Weer contains more than 1.1 million tuples, and the corresponding imputation RMSE, runtime, memory, and streaming imputation results are reported in Tables II-V, according to R3O3. (4) For the concern on "detailed discussion of hyperparameters", we add determination strategies of our hyperparamters in Section VI-B, with the experiments about the learning rate hyperparamter in Figure 11 and the number of iterations in Figure 12, following R3O2.

3. For the concern on "benefits of each individual algorithm and a recommendation framework on when to choose which one", we add the recommended scenarios along with clarification on the tradeoff between imputation quality and runtime for our four algorithms in Section III, as required in R1O1.

To Reviewer 1

Thank you for your constructive suggestions. Please find our responses and revisions (in red) below.

R1O1: O1. The reader is presented with a total of four algorithms. A clear recommendation on which scenario to employ which algorithm is required, along with clarification on the tradeoff between imputation quality and runtime.

Reply: This is a great point! Yes, as discussed in Section III, we design four algorithms to impute incomplete multivariate time series. CDI imputes missing values using dependency models in the single processor, while PCDI further enhances this process into a parallel version using multiple processors. Moreover, since there may exist many missing cells, making the dependency models trained only over complete imputation contexts not accurate enough to capture dependencies of the whole dataset, CDIH consider dynamic models to impute huge missing values, where both imputation values and models are collaboratively optimized. PCDIH similarly extends CDIH into the parallel version, updating dynamic models and fillings collaboratively in multiple processors.

The experiments in Table II show that PCDIH generally exhibits better performance than PCDI, with the help of more accurate dynamic models, especially for the higher missing rate cases, e.g., AirQuality with 80% missing values. Meanwhile, the additional updates for dynamic models may lead to higher time costs for PCDIH, compared with PCDI, e.g., as shown in the Energy dataset in Table III. Moreover, in Figures 8, 9, 10, 11, 12, as for the imputation accuracy, there is no significant difference between the sequential algorithm CDI and its parallel version PCDI, but PCDI spends a significantly lower time cost than CDI. Similar experimental results are also oberved between CDIH and PCDIH. Therefore, as for the real application scenarios, if there are multiple processors

Table I
Dataset summary

Dataset	$\|I\|$	$\|R\|$	Missing Value	Missing Rate ($\frac{\|M\|}{\|I\|\cdot\|R\|}$)
Energy	19,735	9	artificial	-
Ethanol	917,000	3	artificial	-
AirQuality	9,357	13	real, no truth	13.72%
MIMIC-III	79,700	6	real, no truth	21.84%
GPS	3,155	8	real, labeled truth	42.98%
IMU	13,939	6	real, labeled truth	0.46%
Weer	1,140,600	5	real, labeled truth	0.08%

available, PCDI and PCDIH are more suggested than CDI and CDIH. For instance, supercomputers, servers, and personal computers, which typically have multiple processing units capable of executing multiple tasks or different parts of the same task in parallel, are able to meet the above requirements [4]. Otherwise, if there is only one processor can be used, CDI and CDIH can be applied to impute missing values under such a scenario. For example, low-cost microcontrollers and embedded devices have such a requirement, since they often use low-cost, low-power and single-core microcontrollers [10]. These devices have limited processing power and therefore typically support only the single-process execution.

Moreover, according to the experimental results in Tables II and III, PCDIH (resp. CDIH) is more recommended than PCDI (resp. CDI), when the effectiveness requirement is more favored than the efficiency. For instance, in equipment monitoring and medical health monitoring scenarios [19], there may be large number of missing values, and the requirement for data integrity and imputation accuracy is usually higher than that for the processing speed, due to the sensitivity and complexity of decision making or data analysis.

In a word, if there are few missing values and only one processor is available, CDI will be suggested. As for few missing values and multiple available processors, PCDI is recommended. Moreover, when there exist many missing values and the efficiency requirement is not strict, CDIH will be chosen if we only have one processor, but PCDIH will be more preferable if there are more available processors.

According to the comments, we add the aforesaid explanations about the recommended scenarios along with clarification on the tradeoff between imputation quality and runtime for our four algorithms in Section III.

R1O2: O2. How are the inter- and intra-temporal models (listed in the appendix) produced? By providing the reader with additional information regarding this matter, one can improve the utility of this method.

Reply: We apologize for the inadequate descriptions and explanations of the dependency models in Section II-A and Appendix.L. To produce the dependency models of intratemporal and intertemporal patterns in Appendix.L, for each attribute value, we extract the complete imputation contexts that can model the intertemporal dependency on this attribute over time and the intratemporal dependency between all the attribute values at the same time, following the process shown in Figure 3. We then train the dependency models using the gradient descent method over the complete imputation contexts for each attribute in turn. After training, the weights between different attributes at the same timestamp in the dependency model is associated with the intratemporal dependency and the intertemporal dependency is related to the weights between different timestamps for the same attribute, as defined in the previous works [63], [12]. We add the above explanations about how the intertemporal and intratemporal models are produced in Appendix.L.

For example, as shown in Figure 3, we first find all the complete imputation contexts C_{31}, C_{41}, and C_{51} of the attribute A_1, and then we can train the dependency model g_1 on these three complete imputation contexts. After training, we can get the dependency model g_1: $x_{i1} = 0.21x_{i2} + 0.16x_{i3} + 0.72x_{i-1,1} - 0.04x_{i+1,1} - 0.33$, where 0.21 and 0.16 denote the contributions of attribute A_2 (Humidity) and attribute A_3 (Temperature) to A_1 (NOx), respectively, i.e., humidity and temperature will affect the NOx concentration through the intratemporal dependency. That is, higher temperatures typically increase the reaction rate of NOx with other pollutants such as volatile organic compounds, leading to more ozone production, which affects the concentration of NOx, While higher humidity promotes the removal of NOx from the atmosphere through wet deposition. While 0.72 and -0.04 indicate the influence of previous and subsequent NOx values, respectively, i.e. the previous and following NOx concentration values will contribute to the imputation of the current NOx concentration through the intertemporal dependency, and (-0.33) is the bias term. To improve the utility of our methods, we add the aforesaid discussions in Example 2 in Section II-A.

R1O3: O3. The authors presented briefly in Figure 6 various missing mechanisms and concluded that their method works with all of them. I believe that, in reality, it is not true that all variables follow the same missing mechanism. It is essential to further discuss this direction.

Reply: Thanks for pointing this out.

(1) To better investigate the performance of using our algorithms in reality, in addition to the real-world incomplete datasets AirQuality, MIMIC-III and GPS, we introduce another two new datasets IMU [49] and Weer [62] with real missing values and labeled ground truth. These five real-world incomplete datasets come from different scenarios with different missing mechanisms in reality, e.g., GPS and IMU containing consecutive missing values unlike the others, making our experimental evaluation has more practical significance. Table I summarizes their basic characteristics. IMU [49] consists of 13,939 observations from the inertial measurement units. Since the sensors can be turned off and have precision problems, the observations generated by the low-precision gyroscopes may contain missing values, whose ground truth data are thus labeled by the higher-precision sensors. Weer [62] records weather data for Soesterberg, Nertherland, from 1951 to 2010 with 1,140,600 tuples over 5 attributes. Due to station relocations and sensor failures, there exist real-world missing data in the collected records. Since the fillings of missing values are unavailable, we obtain the ground truth values from De Bilt, the closest neighbor station to Soesterberg. We add the aforesaid descriptions of

Table II

Imputation RMSE of our methods compared to the existing approaches over various datasets with different missing rates

Approach	Energy					Ethanol					AirQuality				MIMIC-III			GPS	IMU	Weer
	10%	20%	30%	40%	80%	10%	20%	30%	40%	80%	20%	30%	40%	80%	30%	40%	80%	42.98%	0.46%	0.08%
BTMF	0.24	0.22	0.20	0.23	0.39	88.2	122.8	166.3	227.1	481.6	397.4	344.4	304.4	258.3	9.40	12.71	29.56	2.26610	74.18	31.51
RecovDB	0.64	0.63	0.64	0.65	0.67	4083.6	4070.7	4071.5	4030.5	14913.0	143.8	142.8	153.5	479.1	0.38	0.41	1.02	0.01478	88.40	81.51
ORBITS	1.82	1.98	2.12	2.44	2.53	5740.6	5242.4	5113.5	4668.2	3616.2	156.6	174.9	181.2	197.1	0.57	0.64	0.38	0.33022	3.80	37.41
BayOTIDE	1.08	1.48	1.78	1.91	4.79	1128.7	1208.5	1167.8	1469.9	1967.0	479.5	480.6	488.8	579.6	0.68	0.94	2.72	0.31051	23.11	83.66
BRITS	1.24	1.20	1.21	1.30	1.61	3321.9	3407.8	3474.7	3694.5	3923.7	149.9	151.0	151.0	512.2	0.32	0.31	0.57	0.56217	4.78	36.10
SAITS	0.99	1.13	1.20	1.17	1.80	890.9	886.2	780.9	1093.2	1293.0	154.9	197.7	181.0	199.0	0.38	0.37	0.40	0.41787	44.98	30.51
GRIN	0.38	0.38	0.40	0.43	1.37	203.4	233.8	202.3	277.5	2616.6	192.8	195.6	196.9	232.9	0.41	0.29	0.57	0.20008	18.59	31.22
DAMR	0.43	0.39	0.40	0.39	0.60	695.5	680.5	689.0	675.5	907.6	177.5	181.6	183.0	283.8	3.53	5.60	7.04	0.24337	1.53	35.04
E2GAN	1.90	2.12	2.58	2.82	2.77	3319.7	6296.0	13582.4	23246.1	24410.6	169.7	172.8	187.2	313.0	0.50	0.36	1.31	0.31900	3.82	80.52
MIWAE	0.64	0.57	0.58	0.56	0.57	3317.4	3232.0	3257.4	3218.9	3596.5	164.0	156.5	139.5	190.5	0.26	0.23	0.50	0.31296	14.23	37.22
CSDI	0.19	0.22	0.23	0.26	0.87	98.8	108.8	130.7	298.0	2393.6	168.0	195.6	237.6	311.0	0.32	0.35	0.43	0.03517	13.81	37.64
PriSTI	0.17	0.18	0.19	0.32	2.49	89.8	180.9	178.5	419.9	6135.5	234.6	306.7	356.5	226.8	0.29	0.29	2.40	0.17946	50.57	40.58
PCDI	**0.12**	0.13	0.13	0.13	0.16	86.4	97.1	100.7	112.7	243.7	130.2	158.1	189.9	243.2	**0.11**	**0.11**	0.36	0.00523	0.77	**28.58**
PCDIH	**0.11**	**0.12**	**0.12**	**0.12**	**0.15**	**84.9**	**92.1**	**96.1**	**108.2**	**221.1**	**124.6**	**124.4**	**125.1**	**167.5**	**0.11**	**0.11**	**0.35**	**0.00519**	**0.76**	**28.58**

*AirQuality and MIMIC-III contain 13.72% and 21.84% real missing values without ground truth respectively, where we can only evaluate the imputation accuracy more than 20% and 30% missing rates for them. GPS, IMU and Weer have real missing values with labeled ground truth.

Table III

Imputation runtime (in seconds) of the approaches over various datasets with different missing rates

Approach	Energy					Ethanol					AirQuality				MIMIC-III			GPS	IMU	Weer
	10%	20%	30%	40%	80%	10%	20%	30%	40%	80%	20%	30%	40%	80%	30%	40%	80%	42.98%	0.46%	0.08%
BTMF	315.5	316.6	316.7	317.2	498.8	8413.3	7536.4	7470.5	7405.8	8535.2	206.5	225.3	220.7	265.0	284.9	291.5	467.1	7.6	136.5	641.6
RecovDB	0.2	0.2	0.2	0.2	0.3	1.4	1.4	1.5	1.5	1.5	0.2	0.1	0.2	0.2	0.7	0.6	0.5	0.1	0.1	1.5
ORBITS	16.3	16.2	16.3	16.0	14.2	174.5	154.4	151.1	134.7	217.8	16.3	16.5	15.4	170.8	43.8	43.6	80.5	3.5	297.5	7.9
BayOTIDE	172.3	175.6	173.0	175.5	180.3	4274.2	4197.5	4292.5	4256.6	3694.9	109.5	108.0	107.6	94.8	498.4	491.3	345.1	23.2	65.7	2363.7
BRITS	184.2	189.9	197.6	239.6	256.6	13625.7	13459.8	13512.0	14566.5	14497.0	117.6	117.8	116.8	121.9	67.4	66.2	97.5	63.0	33.0	1051.0
SAITS	1.9	1.9	1.9	2.1	4.0	19.4	19.6	19.6	19.4	20.8	4.8	4.8	4.8	4.7	4.6	4.4	6.0	1.1	1.6	16.3
GRIN	278.8	276.6	275.5	277.5	285.2	7317.6	7638.3	7077.1	7362.7	7460.1	31.2	31.3	31.4	33.2	216.6	220.1	288.7	46.4	186.7	604.7
DAMR	1302.3	1301.5	1320.3	1328.4	1317.6	31559.3	31376.8	30492.6	32839.2	60106.3	622.5	620.0	633.6	1555.8	5307.5	5220.9	11859.5	211.2	2039.7	16266.1
E2GAN	978.1	985.0	1090.7	1133.4	1227.2	22267.1	22361.5	22857.9	23358.7	22802.5	543.3	542.5	541.6	589.9	813.1	836.2	821.8	289.6	292.4	12864.1
MIWAE	35.6	33.7	33.9	34.3	33.1	863.1	868.0	878.3	871.0	884.2	6.7	1271.3	6.4	10.0	457.4	496.9	485.7	5.9	25.6	884.7
CSDI	309.5	360.8	407.4	454.0	557.7	13714.1	16234.4	18002.5	19069.7	19123.2	255.5	6.7	289.0	301.7	2066.6	1908.2	2018.6	37.6	154.2	1837.4
PriSTI	391.7	402.7	422.5	433.1	454.0	28785.6	28663.4	24106.8	28638.1	28030.8	499.8	271.6	502.1	548.7	6062.5	6044.8	6118.2	72.1	182.5	4712.8
PCDI	66.7	204.8	443.6	788.6	1953.8	1895.7	2535.8	3364.1	4048.6	5335.5	667.1	502.9	1275.1	1800.7	9667.9	7583.9	11083.9	32.0	11.8	138.7
PCDIH	147.6	865.9	1439.7	2389.5	5025.5	2209.9	3041.5	3761.2	4856.2	5421.1	918.4	982.8	1474.3	2023.3	10600.4	8449.9	12455.2	46.1	12.4	141.6

new datasets IMU and Weer with the updated Table I in Section VII-A.

The corresponding experimental results about the comparison between our algorithms and competitive methods are shown in the updated Table II and Table III. We can find that matrix factorization (MF) based BTMF [18] may struggle with imputing various missing data due to the reliance on low-rank approximations. Centroid decomposition (CD) based RecovDB [5] and ORBITS [31] may fall short as centroid decomposition may miss minor features critical for data imputation. ORBITS prioritizes the online imputation, potentially reducing accuracy compared to RecovDB. As for Bayesian-based BayOTIDE [26], its use of Gaussian Processes with smooth and periodic kernels may not suit complex time series data. Deep learning models BRRTS [16], SAITS [24], GRIN [21], E2GAN [39], MIWAE [40], CSDI [61] and PriSTI [36] often require time series segmentation, limiting their effectiveness with long gaps. Since we can not get adjacency matrices for spatiotemporal imputation methods GRIN, DAMR [54] and PriSTI, they do not perform well. Benefiting from the dependency models that capture intertemporal and intratemporal dependencies, with the convergence guarantee to ensure a collaborative imputation, our methods can still show the superiority over other baselines on these two real-world incomplete datasets.

Moreover, as shown in Table III, while DL-based methods consume excessive time due to their large number of parameters, our methods do not have such requirements. Furthermore, we find that PCDI and PCDIH take only 138.7 seconds and 141.6 seconds respectively over the large-scale Weer dataset, which are faster than most of the baselines. Combining with the experimental results of the memory consumption in Table V in Appendix in [1], our methods have the reasonable scalability on large-scale Ethanol and Weer datasets, which demonstrate the contributions of our parallel imputation strategies.

Combining with the imputation results over the GPS dataset in Table II and the application performance in Figure 13 over AirQuality and MIMIC-III datasets, which also contain real-world missing values, our methods demonstrate the practicability in reality.

(2) On the other hand, conducting the experimental evaluation with three missing mechanisms, i.e., MCAR, MAR, MNAR, has been widely used as a setup for exploring different missing scenarios [43], [47], [52]. The experiments with different missing mechanisms in Figure 6 show that our methods can be applied in different missing scenarios with various incomplete values.

We add these new comparison results over additional real-world incomplete datasets IMU and Weer in Tables II and III with the corresponding analyses above into Section VII-B.

R1O4: O4. "PCDI may produce slightly different results with CDI". "we can get the same results for CDI and PCDI by manually specifying the same update steps for them". Is this not limiting when using PCDI?

Reply: Thanks for your insightful comments. It is true that PCDI may produce slightly different results compared to CDI

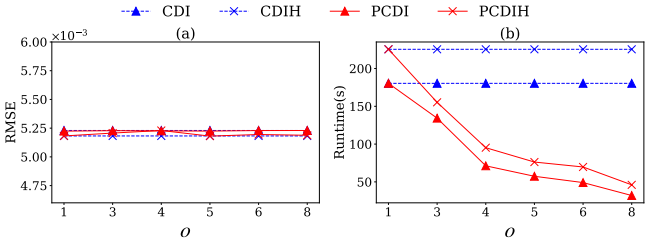


Fig. 8. Varying the number of processors o, over GPS

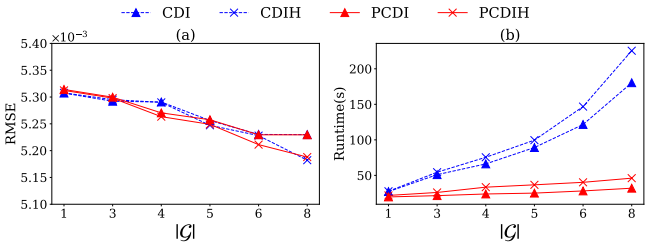


Fig. 9. Varying the number of dependency models $|\mathcal{G}|$, over GPS

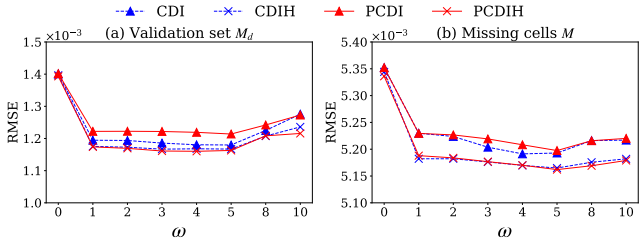


Fig. 10. Varying the temporal window size ω, over the validation set M_d and real-world missing cells M of GPS

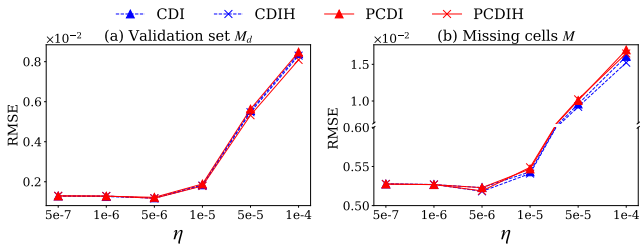


Fig. 11. Varying the learning rate η, over the validation set M_d and real-world missing cells M of GPS

in practice, due to the parallel processing nature of PCDI. Such differences may arise because the imputation contexts assigned to each processor are different, so that the number of iterations on different processors may be slightly different. However, it is important to highlight that although there are minor differences in the iteration paths, these do not significantly impact the overall performance or the final convergence of algorithms. Our theoretical Propositions 1, 3, 4, 6, 7 ensure that, despite these minor variations, the convergence behavior remains robust, leading to consistent and reliable imputation results. Referring to the experimental results presented in Figures 8, 9, 10, 11, we demonstrate that PCDI achieves similar levels of accuracy to CDI. These experiments show that PCDI utilizes multiple processors to speed up the computation without compromising the accuracy of the results, thereby confirming the effectiveness of our parallel processing approaches. Given our theoretical underpinnings and the experimental results, it is unnecessary to manually set the number of update steps for PCDI, to match those of CDI. Our parallel methods come with the convergence guarantees that allow them to autonomously determine the point of convergence and cease the training accordingly. This automatic stopping mechanism negates the need to forcefully align the update steps with those used in non-parallel CDI and CDIH, as doing so does not constitute a condition for convergence. Moreover, both PCDI and PCDIH show reduced runtimes with more processors, and PCDI typically performs faster due to the absence of dynamic model updates. Based on these observations and the performance metrics reported, we advocate for the use of PCDI when multiple processors are available. Therefore, using PCDI is not limiting but rather an enhancement especially in contexts where resource utilization and processing time are critical considerations.

We add the aforesaid explanations for the new Figures 8, 9, 10, 11 over the real incomplete GPS dataset in Section VII-C.

R1O5: O5. The experimental evaluation can be improved substantially.

A- The primary focus of the evaluation of imputation quality was on syntactic noise. The inclusion of additional real-world datasets containing ground truth will enhance the significance of the reported outcomes.

B- It is counterintuitive to employ MCAR in the experiments (e.g., table 3) following the motivation of the paper by emphasizing the dependency between the attributes.

C- Scalability: It is well-known that the joint-set algorithm has a memory overhead. Memory consumption as well as scalability with the size of the dataset are not reported.

D- Parallelization: Reporting on the number of maximum self-connected contexts can provide a better understanding of how parallelizable the problem is.

E- All of the results in Figure 7-11 were discussed using only 10% "artificial" missing values. It would be more informative if real missing data from the GPS were used.

F- On average, how many rounds of imputation are conducted, and on which parameter is this dependent.

G- Why different metrics are being used for the two scenarios in Figure 12. This is making it difficult to compare.

Reply: A- Thank you for the valuable suggestions. To better evalute the imputation quality of using our algorithms in reality, in addition to the real-world incomplete datasets AirQuality, MIMIC-III and GPS, we introduce another two datasets IMU [49] and Weer [62] with real missing values and labeled ground truth. These five datasets come from different real scenarios with different missing mechanisms, enhancing the significance of the reported outcomes. Table I summarizes their basic characteristics. IMU [49] consists of 13,939 observations from the inertial measurement units. Since the sensors can be turned off and have precision problems, the observations generated by the low-precision gyroscopes may contain missing values, whose ground truth data are thus labeled by the higher-precision sensors. Weer [62] records

Table V
Imputation memory consumption (in MiB) of the approaches over various datasets with different missing rates

Approach	Energy					Ethanol					AirQuality				MIMIC-III			GPS	IMU	Weer
	10%	20%	30%	40%	80%	10%	20%	30%	40%	80%	20%	30%	40%	80%	30%	40%	80%	42.98%	0.46%	0.08%
BTMF	217.0	233.8	217.1	220.7	236.8	194.4	194.4	194.4	194.6	194.8	216.1	217.4	210.8	214.8	246.2	249.0	279.5	199.7	228.7	531.1
RecovDB	93.5	93.5	94.9	95.3	94.8	137.4	143.0	148.6	151.7	168.0	86.6	86.7	86.6	88.4	103.4	109.8	109.8	78.9	83.3	143.7
ORBITS	133.1	143.4	153.6	143.4	122.9	163.8	153.6	184.3	174.1	143.4	61.4	92.2	143.4	92.2	61.4	71.7	184.3	51.2	122.9	235.5
BayOTIDE	1130.0	1130.3	1130.4	1130.4	1096.0	1084.9	1084.7	1087.7	1080.8	981.2	715.4	715.9	716.0	703.2	3441.9	3416.0	3097.7	463.0	889.3	4382.7
BRITS	2813.4	2817.0	2818.2	2817.9	2818.9	2855.3	2859.2	2861.7	2864.6	2903.0	2813.5	2813.4	2815.0	2815.5	2830.1	2830.5	2835.3	2810.3	2811.6	3163.8
SAITS	2570.3	2575.0	2573.9	2573.7	2576.8	2570.2	2636.1	2643.6	2648.0	2648.0	2566.0	2567.8	2571.9	2570.2	2588.2	2591.8	2592.5	2562.0	2562.3	2568.5
GRIN	5312.5	5319.1	5321.2	5319.4	5320.4	5399.0	5399.9	5401.4	5400.4	5398.2	5321.3	5320.7	5321.2	5325.5	5334.8	5332.0	5336.7	5306.8	5309.4	5413.4
DAMR	1147.0	1168.4	1145.9	1147.4	1169.7	2295.9	2318.8	2306.7	2310.1	2295.1	874.4	875.3	868.2	868.8	1235.4	1238.5	1239.6	634.0	719.1	38863.1
E2GAN	3274.7	3272.6	3278.0	3278.7	3282.4	3317.7	3322.0	3324.0	3327.8	3363.0	3274.1	3267.8	3270.3	3275.7	3293.3	3296.6	3293.6	3264.2	3264.2	1248.9
MIWAE	2225.7	2227.4	2228.9	2227.4	2230.6	2316.2	2322.6	2326.9	2330.4	2347.0	2219.1	2221.8	2221.4	2224.0	2250.0	2250.3	2252.4	2217.9	2221.3	5413.4
CSDI	2411.2	2416.6	2418.0	2419.0	2420.3	2482.8	2487.1	2495.0	2499.3	2515.5	2411.9	2412.4	2416.3	2415.4	2553.9	2543.9	2536.3	2406.7	2409.5	2611.7
PriSTI	2562.0	2560.6	2561.8	2560.8	2564.2	2601.3	2602.5	2602.2	2599.5	2601.4	2557.2	2558.5	2558.1	2558.4	2568.1	2567.3	2568.7	2548.8	2553.9	2663.7
PCDI	393.9	402.6	413.9	442.6	517.7	317.1	320.4	326.6	329.0	399.1	387.5	523.4	523.3	702.1	857.1	994.2	1088.9	303.2	294.1	436.1
PCDIH	497.4	521.8	583.6	693.3	762.5	319.7	316.2	318.8	323.6	414.3	393.4	525.6	546.1	721.5	948.5	1003.2	1092.3	303.7	295.7	441.8

weather data for Soesterberg, Nertherland, from 1951 to 2010 with 1,140,600 tuples over 5 attributes. Due to station relocations and sensor failures, there exist real-world missing data in the collected records. Since the fillings of missing values are unavailable, we obtain the ground truth values from De Bilt, the closest neighbor station to Soesterberg. We add the aforesaid descriptions of new datasets IMU and Weer with the updated Table I in Section VII-A.

The corresponding experimental results about the comparison between our algorithms and competitive methods are shown in the updated Table II and Table III. Benefiting from the dependency models that capture intertemporal and intratemporal dependencies, with the convergence guarantee to ensure a collaborative imputation, our methods can still show the superiority over other baselines on these two real-world incomplete datasets. Combining with the imputation results over the GPS dataset in Table II and the application performance in Figure 13 over AirQuality and MIMIC-III datasets, which also contain real-world missing values, our methods demonstrate the practicability in reality. We add the aforesaid discussions of new experiments in Section VII-B.

B- Thanks for the comments. MCAR is the most widely used missing mechanism for evaluating imputation methods [68], [7], [74], and its missing values are generated randomly over various attributes [43]. We argue that such a missing mechanism does not contradict the existence of dependencies between attributes. Instead, it proves the robustness and the effectiveness of our algorithms, emphasizing the dependencies between the attributes, to deal with various missing values. For example, for the medical dataset MIMIC-III, the blood pressure reading may be missing due to a recorder's misuse, which is unrelated to any other attribute or the attribute values itself. However, the blood pressure attribute may still have dependencies on attributes such as the heart rate. This missing mechanism does not potentially favor our methods, but our methods can still perform accurate imputation by capturing intratemporal and intertemporal dependencies. In addition, as replied to R1O3, we do not limit the evaluation to the MCAR mechanism. We also explore the imputation effects with three different missing mechanisms in Figure 6, where our algorithms show the consistent superiority in different missing scenarios.

Moreover, as suggested by R1O5.A, in Table I, in addition to GPS, AirQuality and MIMIC-III datasets with real missing values, we include another two real-world incomplete datasets IMU [49] and Weer [62]. As shown in Table II, our methods can perform the superiority against twelve baselines over various datasets under various missing scenarios. Furthermore, we also evaluate the performance of applying imputation methods in downstream applications on the air quality prediction and in-hospital mortality forecasting tasks, with real-world missing values in AirQuality and MIMIC-III datasets. These results demonstrate the ability of our methods to handle missing data in reality, as well as the practicability to serve real applications.

We add the additional explanations for employing MCAR in the experiments and the new results over additional real-world incomplete datasets IMU and Weer in Section VII-B.

C- For the joint-set algorithm, we introduce an array for each missing value to store its ancestor node, resulting in a space complexity $O(|M|)$, where M is the set of missing cells in the time series I. In addition, the time complexity of finding the maximal self-connected contexts set by the joint-set algorithm is $O(|M|\omega m\alpha(|M|))$, where $\alpha(\cdot)$ is the extremely slow-growing inverse Ackermann function [60], $m = |R|$ is the number of attributes, and ω is the window size of dependency models. We add the aforesaid analysis of the joint-set algorithm in Section IV-C.

Further, as commented, we add the experimental results of the memory consumption for different methods over various datasets in Table V into Appendix.I in [1]. To better investigate the scalability of our methods, we introduce an another large-scale dataset Weer [62] with 1,140,600 tuples containing real-world missing values, whose basic characteristics are shown in Table I. According to the memory consumption in Table V and time costs in Table III, our methods have the reasonable scalability on large-scale Ethanol and Weer datasets, which demonstrate the contributions of our parallel imputation strategies. We add the aforesaid explanations in Section VII-B.

Moreover, to serve the real-life applications in real-time scenarios, we also add a new Section VI-A to use our algorithms to conduct the real-time streaming imputation. The additional experiments for processing missing values in the streaming manner over real-world incomplete datasets with labeled truth, i.e., GPS, IMU and Weer, are also reported in Table IV with the following experimental analysis in Section VII-B, where IMU and Weer are the newly introduced datasets with real

Table IV
Streaming imputation RMSE and averaged processing runtime for each missing tuple over real-world incomplete datasets

Approach	GPS		IMU		Weer	
	RMSE	Runtime (ms)	RMSE	Runtime (ms)	RMSE	Runtime (ms)
ORBITS	0.3302	1.63	23.05	59.38	43.18	28.01
BayOTIDE	0.3122	7.35	4.66	47.13	87.40	7.29
CDI	0.0056	1.27	1.37	36.59	30.24	16.58
CDIH	0.0054	1.89	1.37	54.85	30.23	18.73

Table VI
Number of maximal self-connected contexts over various datasets with different missing rates

Dataset	Missing rate	Number of maximal self-connected contexts
Energy	10%	1495.80
	20%	995.60
	30%	431.60
	40%	138.60
	80%	1
Ethanol	10%	90.84
	20%	133.37
	30%	140.51
	40%	122.57
	80%	4.79
AirQuality	20%	26.80
	30%	7.20
	40%	4
	80%	1
MIMIC-III	30%	2552.4
	40%	1164.2
	80%	1
GPS	42.98%	110
IMU	0.46%	2
Weer	0.08%	195

missing values. We consider the first half of tuples in the dataset as already collected historical data, then conduct the online imputation for the other half of the subsequent tuples in real-time scenarios. Among the baselines, only ORBITS [31] and BayOTIDE [26] are specially designed for the streaming imputation of time series data. The other competing methods, such as BTMF [18], RecovDB [5], and various deep learning models BRITS [16], SAITS [24], GRIN [21], DAMR [54], E2GAN [39], MIWAE [40], CSDI [61], PriSTI [36], cannot be applied for the real-time imputation. The reason is that BTMF and RecovDB require the full dataset view to compute their decompositions and cannot dynamically adapt to new data without recalculating the entire model, which are computationally infeasible in real-time scenarios. Deep learning based methods need retraining or significant adjustments when new data batches are introduced, which becomes a bottleneck in real-time scenarios. The experimental results in Table IV show that our algorithms process each imputation quickly, at a millisecond level (comparable with specified real-time approaches ORBITS and BayOTIDE), with the higher imputation accuracy. Such results make our algorithms suitable for the real-time scenarios, which typically require sub-second level responses [26], [58], [31]. We add the new Table IV with the corresponding analysis in Section VII-B.

D- Thank you for the thoughtful comments. Accordingly, we report the number of maximal self-connected contexts over various datasets with different missing rates in Table VI in Appendix.J in [1]. It can be noticed that the vast majority of scenarios have multiple maximal self-connected contexts, demonstrating the rationality of our parallel algorithms. In addition, there are two maximal self-connected contexts in the IMU dataset with real missing values, since the main

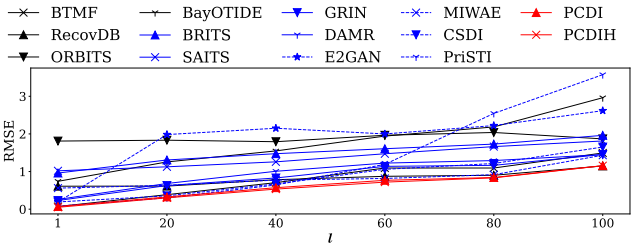


Fig. 7. Varying the maximum missing length l, over Energy with 10% missing values

source of these missing data is that the recording starts with all the inertial measurement unit sensors closed, leading to consecutive incomplete tuples. This condition will increase the difficulty of data imputation while also affecting the efficiency of the parallel imputation. However, our algorithms can still be more efficient than most baselines in Tables III and V, which demonstrates the efficiency of our algorithms, even for the real-world incomplete datasets containing few maximal self-connected contexts.

E- Thanks for the valuable suggestions. We experimentally evaluate the performance of our methods using the GPS dataset with real missing values as suggested and the results are illustrated in Figures 8, 9, 10, 11. As shown in Figure 8, the variation of processors does not significantly affect the imputation accuracy, since models are guaranteed to converge under different settings (Propositions 1, 3, 4, 6, 7). We also observe that PCDI (resp. PCDIH) may produce slightly different results compared to CDI (resp. CDIH), due to the parallel processing nature of PCDI (resp. PCDIH). Such differences may arise because the imputation contexts assigned to each processor are different, so that the number of iterations on different processors may be slightly different. However, it is important to highlight that these minor differences do not significantly impact the overall performance or the final convergence of algorithms. Our theoretical propositions ensure that, despite these minor variations, the convergence behavior remains robust, leading to consistent and reliable imputation results. Our parallel methods come with the convergence guarantees that allow them to autonomously determine the point of convergence and cease the training accordingly. This automatic stopping mechanism negates the need to forcefully align the update steps with those used in non-parallel CDI and CDIH, as doing so does not constitute a condition for convergence. Moreover, both PCDI and PCDIH show reduced runtimes with more processors, and PCDI typically performs faster due to the absence of dynamic model updates.

Then we investigate the imputation results with different numbers of dependency models in Figure 9. Generally, more dependency models improves the imputation accuracy. Given PCDIH's asynchronous dependency model updates and the variations in imputing values across processors, incorporating extensive intratemporal and intertemporal dependencies enhances the accuracy. Therefore, we typically set the number of dependency models $|\mathcal{G}|$ to the attribute number m by default.

Figure 10 presents experimental results for varying temporal window sizes ω. The imputation quality is poorest at $\omega = 0$, where no intertemporal dependency is included. The imputation RMSE first decreases with an increasing ω, since a mall ω may lead to insufficient temporal data for consideration. However, with the further increase of ω, it may include more attribute values which are not very important to model the temporal dependencies, resulting in overfitting and affecting the imputation results. To determine the appropriate window size in reality, we add a new Section VI-B for the parameter determination strategy. In short, we can first impute missing values in the validation set which are manually introduced missing cell. Then the parameter value leading to the higher imputation accuracy can be determined for applications. As shown, the ω value leading to the highest accuracy in the validation set in Figure 10(a) generally returns the best imputation results for the real missing cells in Figure 10(b), which shows the rationale of our determination strategy.

Next, we study the imputation performance with various learning rates η in Figure 11. We can observe that a large η makes approaches difficult to converge, with the lower accuracy. While a small η may result in the local minima of models. To address these extremes, in the new Section VI-B, we also study its determination strategy, to obtain the better performance by determining the η value. We observe that the η value yielding the highest accuracy in the validation set in Figure 11(a) also produces the best imputation results in testing missing values in Figure 11(b), verifying the effectiveness of our parameter determination strategy.

In addition, the experiments of varying the maximum missing length ℓ could not be performed over the real incomplete GPS dataset, since it contains the fixed 42.98% missing values. We thus use the Energy dataset to inject missing values with different missing lengths in Figure 7, which has larger missing lengths than those in the previous Figure 7 in our original submission. As shown in Figure 7, the performance of all methods degrades as the maximum length of continuous missing data increases. This phenomenon underlines the challenge in handling extensive gaps in data collection. Notably, despite the performance degradation across all approaches with larger gaps, the methods presented in our study continue to achieve the superior performance.

We add the new Figures 8, 9, 10, 11 over the real-world incomplete GPS dataset with aforesaid explanations in Section VII-C, and the updated Figure 7 with the corresponding analysis in Section VII-B.

F- Thanks for the detailed comments. For all the experiments, we repeat the procedure five times under different random seeds over various datasets with different missing rates, and we report the average results. Moreover, we add the new Table VII into Appendix.K in [1], to detail the average number of imputation rounds under different settings for our methods. The imputation rounds are mainly influenced by several factors, such as the distribution of imputation contexts in different processors, temporal window size ω, dependency models $\mathcal{G}$ and the learning rate η. For instance, when using parallel algorithms, the rounds required for imputation on different processors may vary due to the differing difficulty levels in filling the incomplete imputation contexts, leading to variations in the number of rounds on different processors. Additionally, when the imputation temporal window size ω or the number of dependent models $\mathcal{G}$ increases, more intertemporal dependencies are considered, which may increase the number of rounds due to the rise in the task complexity. Similarly, when the learning rate η is too small, more imputation rounds are required to the convergence. Following the comments, we add Table VII with the aforesaid explanations into Appendix.K in [1].

G- Figure 13 (i.e., Figure 12 in the previous submission) reports the experiment results of using our algorithms and baselines to serve real applications, where Figure 13(a) shows the air quality prediction performance over the AirQuality dataset with 13.72% real missing values and Figure 13(b) presents the in-hospital mortality forecasting accuracy over the MIMIC-III dataset with the 21.84% real missing rate. Notably, the air quality prediction application is a regression task, and the in-hospital mortality forecasting application is the classification problem, e.g., mortality or survival. Following the same line of existing studies [38], [39], where RMSE is widely used to evaluate the regression performance and AUC is adopted for measuring the classification accuracy, we use these two different metrics in Figure 13 (i.e., Figure 12 in the previous submission) for these two different applications. We apologize for the insufficient explanations of adopting such two different metrics in Figure 13 (i.e., Figure 12 in the previous submission), and add the aforesaid analysis in Section VII-D in this revised version.

To Reviewer 3

Thank you for the detailed comments. Please find below our responses, and changes highlighted in magenta in the revision.

R3O1: O1: The theoretical foundation for the method could benefit from more detailed explanations, particularly regarding the assumptions made during the convergence guarantee.

Reply: Thanks for your constructive comments on the theoretical foundation of our method. We appreciate the opportunity to clarify the assumptions used for the convergence guarantee.

(1) The L-smooth assumption is reasonable and common in previous works, including imputation methods [14], objective functions [13] and parallel optimizations [53], [76]. It ensures that the imputation cost function is well-behaved and does not change too abruptly. Moreover, in Appendix.H in [1], we provide the empirical evidence demonstrating that the imputation cost function is L-smooth. As shown, all the datasets (including the new added IMU [49] and Weer [62] datasets with real-world missing values) hold this assumption, which verify the rationale of our theoretical foundation.

(2) In addition, having upper bounds on the lag step for all the update steps of the dependency models are also reasonable, since we always have a maximum delay value that does not exceed the total number of update steps K. This upper limit ensures that no outdated information excessively

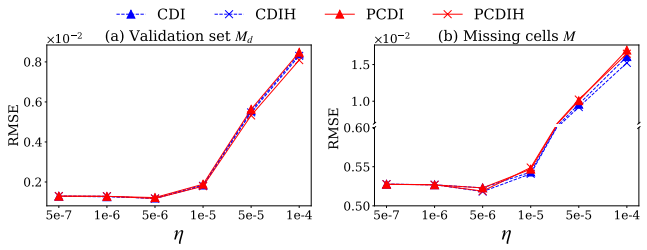


Fig. 11. Varying the learning rate η, over the validation set M_d and real-world missing cells M of GPS

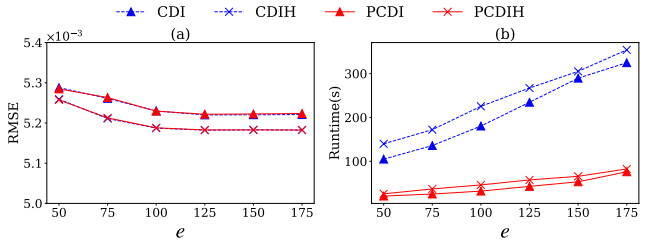


Fig. 12. Varying the number of iterations e, over GPS

influences the updates of dependency models, which is crucial for maintaining the overall efficiency and effectiveness of the learning process. This assumption is not only intuitive but also well-founded within the context of asynchronous updates and distributed data environment [8]. It aligns with typical network and processing constraints that naturally occur in distributed systems. These constraints form a practical basis for assuming bounded delay steps in asynchronous updates.

(3) Moreover, assuming a bound on the derivative of the imputation cost across processors ensures the even and predictable propagation of updates throughout the model training phase. It implies that the gradient's influence does not exceed a certain threshold [76], avoiding scenarios where anomaly updates in certain processors could skew the collective optimization direction. Since the imputation contexts in each processor are assigned based on the size of the maximal self-connected contexts, each processor is assigned those contexts with different timestamps and attributes, making this assumption reasonable.

As suggested, we add these detailed explanations regarding the assumptions made during the convergence guarantee in Assumption 1 in Section V-D, to enhance our theoretical foundation.

R3O2: O2: The experimental results improves by including detailed analysis of key hyperparameters, such as the learning rate, number of iterations, and regularization parameter λ, and their impact on performance.

Reply: Thank you for your suggestion to provide a detailed analysis of key hyperparameters and their impact on performance. (1) Figure 11 presents the imputation performance of our algorithms with the varying learning rate hyperparameter η. We can observe that a large η makes approaches difficult to converge, with the lower accuracy. While a small η may result in the local minima of models.

Therefore, we further devise the determination strategy for the learning rate hyperparameter η in Section VI-B. Practically, for a real incomplete time series I with missing cells M, to determine an appropriate value of the learning rate η, users could run our methods with varying parameter values and choose the one leading to the highest imputation accuracy over the validation set with manually injected missing values. Specifically, in addition to the missing values in M, we can manually introduce more missing cells in I, denoted as M_d, where $M_d \cap M = \emptyset$. Notably, since the ground truth of these missing values in M_d is available and they are unbiased adopted from I, as the same as the missing values in M, we can use M_d as the validation set to determine the parameter values of our algorithms. Then we can consider a candidate set $\mathrm{can}(\eta)$ for the learning rate η. To determine the parameter values of our algorithms, i.e., CDI, PCDI, CDIH, PCDIH, we can impute the missing values in M_d by the algorithm using different candidate parameter values in $\mathrm{can}(\eta)$. Then the parameter value leading to the higher imputation accuracy over the validation set M_d is determined for the algorithm. The reason is that a high imputation accuracy over the validation set with manually injected missing values indicates the good performance of our algorithms in such an application, which is believed to achieve a good imputation result over real missing data as well. Therefore, even though the imputation truth values are unknown in real applications, we can still determine the parameter value in such a reasonable way.

Figure 11 illustrates the imputation RMSE of our algorithms with varying learning rates η over the validation set M_d in Figure 11(a) and the testing set M in Figure 11(b) respectively. We observe that the η value yielding the highest accuracy in the validation set in Figure 11(a) also produces the best imputation results in testing missing values in Figure 11(b), verifying the effectiveness of our parameter determination strategy. We add the aforesaid detailed analysis of the key learning rate hyperparameter η for the new Figure 11 in Section VII-C.

(2) Moreover, as shown in Algorithms 1, 2, 3, 4, the imputation process in our methods is designed to automatically stop when the imputation cost converges. Therefore, there is no need to manually set the number of iterations as a hyperparameter. However, according to the comments, we add the experiments in Figure 12 where we disable the automatic convergence criterion and instead manually set the number of iterations e. We can find that when the number of iteration rounds gradually increases, the imputation cost gradually converges, but the training time will also become

Table I
Dataset summary

Dataset	$\lvert I\rvert$	$\lvert R\rvert$	Missing Value	Missing Rate ($\frac{\lvert M\rvert}{\lvert I\rvert\cdot\lvert R\rvert}$)
Energy	19,735	9	artificial	-
Ethanol	917,000	3	artificial	-
AirQuality	9,357	13	real, no truth	13.72%
MIMIC-III	79,700	6	real, no truth	21.84%
GPS	3,155	8	real, labeled truth	42.98%
IMU	13,939	6	real, labeled truth	0.46%
Weer	1,140,600	5	real, labeled truth	0.08%

Table II

Imputation RMSE of our methods compared to the existing approaches over various datasets with different missing rates

Approach	Energy					Ethanol					AirQuality				MIMIC-III			GPS	IMU	Weer
	10%	20%	30%	40%	80%	10%	20%	30%	40%	80%	20%	30%	40%	80%	30%	40%	80%	42.98%	0.46%	0.08%
BTMF	0.24	0.22	0.20	0.23	0.39	88.2	122.8	166.3	227.1	481.6	397.4	344.4	304.4	258.3	9.40	12.71	29.56	2.26610	74.18	31.51
RecovDB	0.64	0.63	0.64	0.65	0.67	4083.6	4070.7	4071.5	4030.5	14913.0	143.8	142.8	153.5	479.1	0.38	0.41	1.02	0.01478	88.40	81.51
ORBITS	1.82	1.98	2.12	2.44	2.53	5740.6	5242.4	5113.5	4668.2	3616.2	156.6	174.9	181.2	197.1	0.57	0.64	0.38	0.33022	3.80	37.41
BayOTIDE	1.08	1.48	1.78	1.91	4.79	1128.7	1208.5	1167.8	1469.9	1967.0	479.5	480.6	488.8	579.6	0.68	0.94	2.72	0.31051	23.11	83.66
BRITS	1.24	1.20	1.21	1.30	1.61	3321.9	3407.8	3474.7	3694.5	3923.7	149.9	151.0	151.0	512.2	0.32	0.31	0.57	0.56217	4.78	36.10
SAITS	0.99	1.13	1.20	1.17	1.80	890.9	886.2	780.9	1093.2	1293.0	154.9	197.7	181.0	199.0	0.38	0.37	0.40	0.41787	44.98	30.51
GRIN	0.38	0.38	0.40	0.43	1.37	203.4	233.8	202.3	277.5	2616.6	192.8	195.6	196.9	232.9	0.41	0.29	0.57	0.20008	18.59	31.22
DAMR	0.43	0.39	0.40	0.39	0.60	695.5	680.5	689.0	675.5	907.6	177.5	181.6	183.0	283.8	3.53	5.60	7.04	0.24337	1.53	35.04
E2GAN	1.90	2.12	2.58	2.82	2.77	3319.7	6296.0	13582.4	23246.1	24410.6	169.7	172.8	187.2	313.0	0.50	0.36	1.31	0.31900	3.82	80.52
MIWAE	0.64	0.57	0.58	0.56	0.57	3317.4	3232.0	3257.4	3218.9	3596.5	164.0	156.5	139.5	190.5	0.26	0.23	0.50	0.31296	14.23	37.22
CSDI	0.19	0.22	0.23	0.26	0.87	98.8	108.8	130.7	298.0	2393.6	168.0	195.6	237.6	311.0	0.32	0.35	0.43	0.03517	13.81	37.64
PriSTI	0.17	0.18	0.19	0.32	2.49	89.8	180.9	178.5	419.9	6135.5	234.6	306.7	356.5	226.8	0.29	0.29	2.40	0.17946	50.57	40.58
PCDI	**0.12**	0.13	0.13	0.13	0.16	86.4	97.1	100.7	112.7	243.7	130.2	158.1	189.9	243.2	**0.11**	**0.11**	0.36	0.00523	0.77	**28.58**
PCDIH	**0.11**	**0.12**	**0.12**	**0.12**	**0.15**	**84.9**	**92.1**	**96.1**	**108.2**	**221.1**	**124.6**	**124.4**	**125.1**	**167.5**	**0.11**	**0.11**	**0.35**	**0.00519**	**0.76**	**28.58**

*AirQuality and MIMIC-III contain 13.72% and 21.84% real missing values without ground truth respectively, where we can only evaluate the imputation accuracy more than 20% and 30% missing rates for them. GPS, IMU and Weer have real missing values with labeled ground truth.

Table III

Imputation runtime (in seconds) of the approaches over various datasets with different missing rates

Approach	Energy					Ethanol					AirQuality				MIMIC-III			GPS	IMU	Weer
	10%	20%	30%	40%	80%	10%	20%	30%	40%	80%	20%	30%	40%	80%	30%	40%	80%	42.98%	0.46%	0.08%
BTMF	315.5	316.6	316.7	317.2	498.8	8413.3	7536.4	7470.5	7405.8	8535.2	206.5	225.3	220.7	265.0	284.9	291.5	467.1	7.6	136.5	641.6
RecovDB	0.2	0.2	0.2	0.2	0.3	1.4	1.4	1.5	1.5	1.5	0.2	0.1	0.2	0.2	0.7	0.6	0.5	0.1	0.1	1.5
ORBITS	16.3	16.2	16.3	16.0	14.2	174.5	154.4	151.1	134.7	217.8	16.3	16.5	15.4	170.8	43.8	43.6	80.5	3.5	297.5	7.9
BayOTIDE	172.3	175.6	173.0	175.5	180.3	4274.2	4197.5	4292.5	4256.6	3694.9	109.5	108.0	107.6	94.8	498.4	491.3	345.1	23.2	65.7	2363.7
BRITS	184.2	189.9	197.6	239.6	256.6	13625.7	13459.8	13512.0	14566.5	14497.0	117.6	117.8	116.8	121.9	67.4	66.2	97.5	63.0	33.0	1051.0
SAITS	1.9	1.9	1.9	2.1	4.0	19.4	19.6	19.6	19.4	20.8	4.8	4.8	4.8	4.7	4.6	4.4	6.0	1.1	1.6	16.3
GRIN	278.8	276.6	275.5	277.5	285.2	7317.6	7638.3	7077.1	7362.7	7460.1	31.2	31.3	31.4	33.2	216.6	220.1	288.7	46.4	186.7	604.7
DAMR	1302.3	1301.5	1320.3	1328.4	1317.6	31559.3	31376.8	30492.6	32839.2	60106.3	622.5	620.0	633.6	1555.8	5307.5	5220.9	11859.5	211.2	2039.7	16266.1
E2GAN	978.1	985.0	1090.7	1133.4	1227.2	22267.1	22361.5	22857.9	23358.7	22802.5	543.3	542.5	541.6	589.9	813.1	836.2	821.8	289.6	292.4	12864.1
MIWAE	35.6	33.7	33.9	34.3	33.1	863.1	868.0	878.3	871.0	884.2	6.7	1271.3	6.4	10.0	457.4	496.9	485.7	5.9	25.6	884.7
CSDI	309.5	360.8	407.4	454.0	557.7	13714.1	16234.4	18002.5	19069.7	19123.2	255.5	6.7	289.0	301.7	2066.6	1908.2	2018.6	37.6	154.2	1837.4
PriSTI	391.7	402.7	422.5	433.1	454.0	28785.6	28663.4	24106.8	28638.1	28030.8	499.8	271.6	502.1	548.7	6062.5	6044.8	6118.2	72.1	182.5	4712.8
PCDI	66.7	204.8	443.6	788.6	1953.8	1895.7	2535.8	3364.1	4048.6	5335.5	667.1	502.9	1275.1	1800.7	9667.9	7583.9	11083.9	32.0	11.8	138.7
PCDIH	147.6	865.9	1439.7	2389.5	5025.5	2209.9	3041.5	3761.2	4856.2	5421.1	918.4	982.8	1474.3	2023.3	10600.4	8449.9	12455.2	46.1	12.4	141.6

longer. Therefore, in practice, our automatic stop strategy, according to the converged imputation cost, is effective and efficient. We add the aforesaid detailed explanations about the number of iterations in Figure 12 into Section VII-C.

(3) Finally, since our algorithms default to the 0-1 normalization of data, we do not involve the regularization parameter λ as the key hyperparameter. We apologize for the insufficient explanations and clarify the corresponding analysis for this in the last paragraph of Section VII-C.

R3O3: O3: The scalability claims would be more convincing with an in-depth discussion of the method's performance in real-world, large-scale datasets.

Reply: Thank you for the valuable feedback.

(1) To provide a more convincing scalability claims of our method's performance in real-world, large-scale datasets, we conduct additional experiments using two new datasets, i.e., IMU [49] and the large-scale Weer [62], with real missing data. Table I summarizes their basic characteristics. Weer [62] records weather data for Soesterberg, Nertherland, from 1951 to 2010. Due to station relocations and sensor failures, there exist real-world missing data in the collected records, and we obtain the ground truth values from De Bilt, the closest neighbor station to Soesterberg. Notably, Weer is a large-scale dataset, containing over 1.1 million tuples. Moreover, IMU [49] consists of observations from the inertial measurement units. Since the sensors can be turned off and have precision problems, the observations generated by the low-precision gyroscopes may contain missing values, whose ground truth data are thus labeled by the higher-precision sensors. The imputation accuracy and time costs of various methods over these datasets are shown in Tables II and III. It can be found that our methods outperform the baseline methods in these datasets with real missing values again. Furthermore, in Table III, we find that PCDI and PCDIH take only 138.7 seconds and 141.6 seconds, respectively, over the large-scale Weer dataset, which are faster than most of the competitive baselines, demonstrating the scalability of our methods on real-world large-scale incomplete datasets. We add the new experiments over IMU [49] and the large-scale Weer [62] datasets with real-world missing values in Tables II and III, as well as the corresponding analysis, into Section VII-B.

(2) Moreover, to serve the real-life applications in real-time scenarios, we add a new Section VI-A to use our algorithms conduct the online streaming imputation. Specifically, for the real-time scenarios, when we collect a tuple $x_i = \{x_i[A_1], x_i[A_2], , \ldots, x_i[A_m]\}$ with missing vales requiring to be imputed online, we first consider the set of imputation contexts that it involves in. For each cell $x_{ij} \in x_i$, according to Definition 1, we know that it is involved into the imputation context $C_{i-\omega,j} = \{x_{i-\omega,h} \mid A_h \in R \setminus \{A_j\}\} \cup \{x_{i-\omega+l,j} | l = -\omega, \ldots, \omega\}$, with the latest tuples $\{x_{i-2w}, x_{i-2w+1}, \ldots, x_i\}$, where ω is the temporal window size. Therefore, for each tuple x_i, the set of imputation contexts containing it can be $\mathcal{C}_{i-\omega} = \{C_{i-\omega,1}, C_{i-\omega,2}, \ldots C_{i-\omega,m}\}$. Let M_i denote the set

of missing values in x_i. Considering the pre-trained dependency models $\mathcal{G}$ based on the complete imputation contexts in the historical data, for each missing value $x_{ij} \in M_i$, we can compute the imputation result in the streaming manner by

$$x_{ij}^{\prime(k+1)} \leftarrow x_{ij}^{\prime(k)} - \eta \frac{\partial \Theta(\mathcal{C}_{i-\omega}^{\prime(k)}, \mathcal{G})}{\partial x_{ij}^{\prime(k)}},$$

where k is the imputation step, $x_{ij}^{\prime(k)}$ is the temporary imputation value of x_{ij} and $\mathcal{C}_i^{\prime}$ is the temporary imputation context at the k-th step. On the other hand, with the streamingly incoming tuple x_i, we can also consider refining the parameters of dependency models $\mathcal{G}$ online, having

$$\Phi_{\mathcal{G}^{\prime}}^{(k+1)} \leftarrow \Phi_{\mathcal{G}^{\prime}}^{(k)} - \eta \frac{\partial \Theta(\mathcal{C}_{i-\omega}^{\prime(k)}, \mathcal{G}^{\prime(k)})}{\partial \Phi_{\mathcal{G}^{\prime}}^{(k)}},$$

where $\Phi_{\mathcal{G}^{\prime}}$ is the parameters of dependency models.

Furthermore, we also perform additional experiments for processing missing values in the streaming manner over real-world incomplete datasets with labeled truth, i.e., GPS, IMU and Weer, in Table IV in Section VII-B, where IMU and Weer are the newly introduced datasets with real missing values. We consider the first half of tuples in the dataset as already collected historical data, then conduct the online imputation for the other half of the subsequent tuples in real-time scenarios. Among the baselines, only ORBITS [31] and BayOTIDE [26] are specially designed for the streaming imputation of time series data. The other competing methods, such as BTMF [18], RecovDB [5], and various deep learning models BRITS [16], SAITS [24], GRIN [21], DAMR [54], E2GAN [39], MIWAE [40], CSDI [61], PriSTI [36], cannot be applied for the real-time imputation. The reason is that BTMF and RecovDB require the full dataset view to compute their decompositions and cannot dynamically adapt to new data without recalculating the entire model, which are computationally infeasible in real-time scenarios. Deep learning based methods need retraining or significant adjustments when new data batches are introduced, which becomes a bottleneck in real-time scenarios. The experimental results in Table IV show that our algorithms process each imputation quickly, at a millisecond level (comparable with specified real-time approaches ORBITS and BayOTIDE), with the higher imputation accuracy. Such results make our algorithms suitable for the real-time scenarios, which typically require sub-second level responses [26], [58], [31]. We add the new experiments about the streaming imputation in Table IV with the aforesaid discussions into Section VII-B.

To Reviewer 4

Thank you for your comments and insightful suggestions. Please find our responses, and updates in the revision (in blue).

R4O1: - O1: The technical contribution is limited. The core algorithms rely on standard concepts such as gradient descent and correlation between attributes. The authors should enhance the technical contributions.

Reply: Thank you very much for your insightful comments. To further enhance our technical contributions, as claimed in Section I-C, (1) we add a new Section VI-A to compute the imputation for real-time scenarios in the streaming manner with the streaming imputation experiments in Table IV in Section VII-B, and (2) introduce another new Section VI-B for adaptively determining the parameter values of our algorithms, with the determination experiments in Figures 10 and 11 in Section VII-C. (3) Moreover, we clarify the significance and novelty of our existing technical contributions in Section I-C.

(1) Specifically, we add a new Section VI-A that addresses the requirement for real-time streaming imputation. For the real-time scenarios, when we collect a tuple $x_i = \{x_i[A_1], x_i[A_2], , \ldots, x_i[A_m]\}$ with missing vales requiring to be imputed online, we first consider the set of imputation contexts that it involves in. For each cell $x_{ij} \in x_i$, according to Definition 1, we know that it is involved into the imputation context $C_{i-\omega,j} = \{x_{i-\omega,h} \mid A_h \in R \setminus \{A_j\}\} \cup \{x_{i-\omega+l,j} | l = -\omega, \ldots, \omega\}$, with the latest tuples $\{x_{i-2w}, x_{i-2w+1}, \ldots, x_i\}$, where ω is the temporal window size. Therefore, for each tuple x_i, the set of imputation contexts containing it can be $\mathcal{C}_{i-\omega} = \{C_{i-\omega,1}, C_{i-\omega,2}, \ldots C_{i-\omega,m}\}$. Let M_i denote the set of missing values in x_i. Considering the pre-trained dependency models $\mathcal{G}$ based on the complete imputation contexts in the historical data, for each missing value $x_{ij} \in M_i$, we can compute the imputation result in the streaming manner by

$$x_{ij}^{\prime(k+1)} \leftarrow x_{ij}^{\prime(k)} - \eta \frac{\partial \Theta(\mathcal{C}_{i-\omega}^{\prime(k)}, \mathcal{G})}{\partial x_{ij}^{\prime(k)}},$$

where k is the imputation step, $x_{ij}^{\prime(k)}$ is the temporary imputation value of x_{ij} and $\mathcal{C}_i^{\prime}$ is the temporary imputation context at the k-th step. On the other hand, with the streamingly incoming tuple x_i, we can also consider refining the parameters of dependency models $\mathcal{G}$ online, having

$$\Phi_{\mathcal{G}^{\prime}}^{(k+1)} \leftarrow \Phi_{\mathcal{G}^{\prime}}^{(k)} - \eta \frac{\partial \Theta(\mathcal{C}_{i-\omega}^{\prime(k)}, \mathcal{G}^{\prime(k)})}{\partial \Phi_{\mathcal{G}^{\prime}}^{(k)}},$$

where $\Phi_{\mathcal{G}^{\prime}}$ is the parameters of dependency models.

Besides, in Section VII-B, we also perform additional experiments for processing missing values in the streaming manner over real-world incomplete datasets with labeled truth, i.e., GPS, IMU and Weer, in Table IV, where IMU and Weer are the newly introduced datasets with real missing values. We consider the first half of tuples in the dataset as already collected historical data, then conduct the online imputation for the other half of the subsequent tuples in real-time scenarios. Among the baselines, only ORBITS [31] and BayOTIDE [26] are specially designed for the streaming imputation of time series data. The other competing methods, such as BTMF [18], RecovDB [5], and various deep learning models BRITS [16], SAITS [24], GRIN [21], DAMR [54], E2GAN [39], MIWAE [40], CSDI [61], PriSTI [36], cannot be applied for the real-time imputation. The reason is that BTMF and RecovDB require the full dataset view to compute their decompositions and cannot dynamically adapt to new data without recalculating the entire model, which are computationally infeasible in real-time scenarios. Deep learning based methods need

Table IV
Streaming imputation RMSE and averaged processing runtime for each missing tuple over real-world incomplete datasets

Approach	GPS		IMU		Weer	
	RMSE	Runtime (ms)	RMSE	Runtime (ms)	RMSE	Runtime (ms)
ORBITS	0.3302	1.63	23.05	59.38	43.18	28.01
BayOTIDE	0.3122	7.35	4.66	47.13	87.40	7.29
CDI	0.0056	1.27	1.37	36.59	30.24	16.58
CDIH	0.0054	1.89	1.37	54.85	30.23	18.73

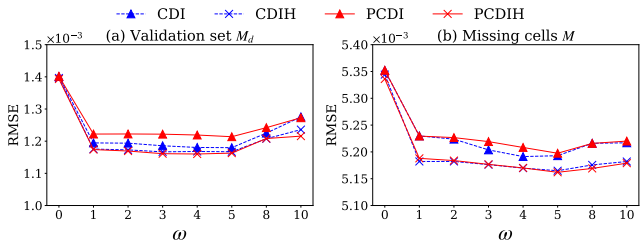


Fig. 10. Varying the temporal window size ω, over the validation set M_d and real-world missing cells M of GPS

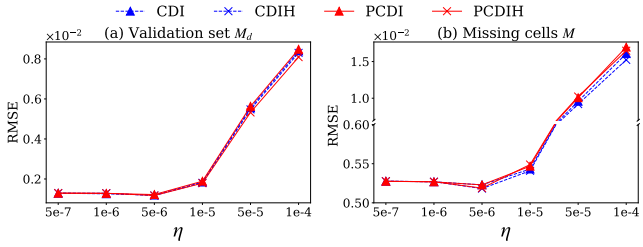


Fig. 11. Varying the learning rate η, over the validation set M_d and real-world missing cells M of GPS

retraining or significant adjustments when new data batches are introduced, which becomes a bottleneck in real-time scenarios. The experimental results in Table IV show that our algorithms process each imputation quickly, at a millisecond level (comparable with specified real-time approaches ORBITS and BayOTIDE), with the higher imputation accuracy. Such results make our algorithms suitable for the real-time scenarios, which typically require sub-second level responses [26], [58], [31].

(2) In addition, we also add a new Section VI-B to automatically determine the parameters of our algorithms. As shown in Figure 8, the variation of processors does not significantly affect the imputation accuracy, but our parallel algorithms PCDI and PCDIH can achieve lower time costs. Therefore, if the hardware conditions permit, more processors are preferred for real applications. Similarly, Figure 9 illustrates that it would be better to consider as many intratemporal and intertemporal dependencies as possible for imputing missing data, and we thus set the number of dependency models $|\mathcal{G}|$ to the attribute number m by default. Therefore, only two parameters, i.e., the window size ω and the learning rate η, need to be adaptively determined for different applications, in the new Section VI-B.

Practically, for a real incomplete time series I with missing cells M, to determine an appropriate value of the algorithmic parameter ω or η, users could run our methods with varying parameter values and choose the one leading to the highest imputation accuracy over the validation set with manually injected missing values. Specifically, in addition to the missing values in M, we can manually introduce more missing cells in I, denoted as M_d, where $M_d \cap M = \emptyset$. Notably, since the ground truth of these missing values in M_d is available and they are unbiasedly adopted from I, as the same as the missing values in M, we can use M_d as the validation set to determine the parameter values of our algorithms. Then we can consider a candidate set for each parameter, say $\mathrm{can}(\omega)$ for the window size ω and $\mathrm{can}(\eta)$ for the learning rate η. To determine the parameter values of our algorithms, i.e., CDI, PCDI, CDIH, PCDIH, we can impute the missing values in M_d by the algorithm using different candidate parameter values in $\mathrm{can}(\omega)$ or $\mathrm{can}(\eta)$. Then the parameter value leading to the higher imputation accuracy over the validation set M_d is determined for the algorithm. The reason is that a high imputation accuracy over the validation set with manually injected missing values indicates the good performance of our algorithms in such an application, which is believed to achieve a good imputation result over real missing data as well. Therefore, even though the imputation truth values are unknown in real applications, we can still determine the parameter value in such a reasonable way.

The corresponding new Figures 10 and 11 illustrate the imputation RMSE of our algorithms with varying window sizes ω and learning rates η over validation set M_d (in subfigure (a)) and testing set M (in subfigure (b)) respectively. As shown, the parameter values leading to the highest accuracy in Figures 10(a) and 11(a) generally return the best imputation results in Figures 10(b) and 11(b), which demonstrates the effectiveness of the determination strategy empirically. Notably, as replied to R4O4, the missing cells M in GPS are continuous due to the sensors being off in some period, thus the randomly generated validation set M_d generally leads to a slightly lower imputation RMSE in Figures 10 and 11. We add these explanations for new Figures 10 and 11 in Section VII-C.

(3) Moreover, as for the contributions of existing techniques in the previous submission, we admit that the gradient descent is a basic technique to solve the optimization objective in the machine learning area and the correlation between attributes is generally used to impute incomplete multivariate time series. However, as we know, the gradient descent is widely used to solve the optimization problem [13], but it would not limit the technical contributions of various methods built upon it. The reason is that the problem statements and how to use the gradient descent to solve problems effectively and efficiently are the core points for serving real applications. Moreover, almost all the time series imputation methods rely on the correlation between attributes to fill missing values, since it is the valuable reference to guide the data imputation process. Notably, it would not decrease the contributions of various imputation methods, since how to fully take usage of it is not trivial, which is actually the main technical contributions of different methods. As discussed in Section VIII, to take full advantage of the correlation between attributes, various

methods as designed according to the way of using it, e.g., capturing the correlation between attributes by regression models [73], [7], matrix factorization [37], [71], [42], [18], centroid decomposition [32], [5], [31], temporal rules [2], Bayesian [26], or deep learning techniques with recurrent neural networks [16], [17], Transformer [24], [6], graph neural networks [21], [54], [25], attention mechanism [69], GAN [38], [39], [29], [44], [75], VAE [40], [27], [48], Diffusion models [61], [36], [66]. Therefore, we argue that using the gradient descent and correlation between attributes are widely accepted basic operations and does not affect the technical contributions of various algorithms with them.

As analyzed in the Introduction, our existing contributions are 1) designing the collaborative imputation algorithms for optimizing both dependent and determinant missing values, as well as the imputation values and models, 2) extending the imputation algorithms into the parallel version, to improve the efficiency, 3) providing the convergence guarantee (Propositions 1, 3, 4, 6, 7) for both sequential and parallel imputation algorithms, to ensure the accurate fillings. Notably, as far as we know, the parallel computation strategy and convergence guarantee are not trivial in existing multivariate time series imputation techniques, which however are very important properties to ensure the efficiency and effectiveness for imputation results. To further enhance the technical contributions, in addition to the existing GPS, AirQuality and MIMIC-III datasets with real-world missing values, we introduce the new real-world incomplete datasets IMU [49] and Weer [62] in experimental comparison. The experiments in Table II show that, with the convergence guarantee, our collaborative imputation algorithms show the superiority over various datasets. Moreover, we observe that PCDI and PCDIH spend 138.7 and 141.6 seconds respectively over the large-scale Weer dataset, which are faster than most of the competing methods and verifies the contribution of our parallel strategies. In this sense, we argue that the standard concepts of gradient descent and correlation between attributes would not decrease our main technical contributions.

Following the comments, in addition to the new experiments over real-world incomplete datasets IMU and Weer in Tables II, III, IV, V and the parameter determination experiments in Figures 10, 11, we add the aforesaid explanations in Section I-C, to further clarify our technical contributions.

R4O2: - O2: The adoption of a dependency model between all attributes assumes a correlation between them. This is a strong assumption that needs to be justified or discussed in the experimental section, perhaps by introducing a column with random (or uncorrelated) values.

Reply: Thanks for your insightful comments regarding the correlation assumptions in our dependency models. We would like to clarify that our methods do not necessarily assume a strong correlation between all attributes. Rather, our methods are designed to adaptively learn the weights of dependencies among attributes. In our dependency models, the correlations are captured through weights that are adaptively learned from the data. Attributes that are not correlated learn weights that are close to zero, effectively adjusting the model to ignore such relationships. For instance, as illustrated in Figure 3 with our dependency model g_2 for the AirQuality data, the attributes A_1 (NOx) and A_3 (Temperature) contribute minimally to A_2 (Humidity) as reflected by their near-zero weights. The reason for the observed low correlation can be attributed to the nature of these attributes. Humidity is primarily influenced by atmospheric conditions rather than by the presence of nitrogen oxides (NOx) or temperature variations directly. Although temperature can affect the capacity of the air to hold moisture, the direct correlation with specific pollutants like NOx is generally weak without additional environmental factors. In addition, our CDIH and PCDIH algorithms can further refine the weights of dependency models in the imputation process. We add the aforesaid clarifications for the correlation between all attributes of dependency models in Section II-A.

In addition, we report all the self-adaptively learned dependency models with specific parameters for various datasets used in experiments, including the new IMU [49] and Weer [62] datasets, in Appendix.L in [1]. As we can see, for each dependency model, there typically exist different weights associated with different attribute values or the same attribute values with different timestamps, and the attribute values with larger weights contribute more to determine the predicted value. Specially, for the IMU dataset, there is no strong intratemporal dependency between the different attributes, as they record data in different axes of the inertial measurement units. Therefore, it can be seen that our methods can self-adaptively learn small weights for other attributes (with parameters for $x_{i1}, \ldots, x_{i6}$) and mainly focus on intratemporal dependencies (with parameters for $x_{i-1,j}, x_{i+1,j}$) of the dependency model g_j. We add these explanations for all the dependency models of various datasets in Appendix.L in [1].

R4O3: - O3: The computation of the dependency model should be discussed in more detail. For example, in Example 2, results are presented without discussion. In Example 3, it is unclear why the imputation cost is $((0.5-2.08)^2)$. Following Example 2, the computed value should be $((0.21*7.4)+(0.16*1.94)+(0.72*0.8)-(0.04*1.9)-0.33=2.0344)$, making the imputation cost $((0.5-2.0344)^2)$. Generally, results in the examples should be discussed in more detail.

Reply: Thank you for the detailed comments. We appreciate your specific observations regarding the inconsistency in the imputation cost computations highlighted in Examples 2 and 3. Sorry for making such an inconsistency, owing to the insufficient explanations. Actually, in Example 2, the exact model parameters are $\phi_{g_1} = (0.2144, 0.1648, 0.7222, -0.0399, -0.3292)^\top$, which is then used to compute the imputation cost in Example 3, having $(0.2144 * 7.4) + (0.1648 * 1.94346189) + (0.7222 * 0.8) - (0.0399 * 1.9) - 0.3292 = 2.07959 \approx 2.08$. However, to make the representation more concise, the model parameters in previous Example 2 are presented by rounding to two decimal places, i.e., $\phi_{g_1} = (0.21, 0.16, 0.72, -0.04, -0.33)^\top$. To avoid the consistency, we revise Examples 2 and 3 as follows.

Example 2: Consider $I = \{x_1, \ldots, x_9\}$ in Figure 1.

Table II
Imputation RMSE of our methods compared to the existing approaches over various datasets with different missing rates

Approach	Energy					Ethanol					AirQuality				MIMIC-III			GPS	IMU	Weer
	10%	20%	30%	40%	80%	10%	20%	30%	40%	80%	20%	30%	40%	80%	30%	40%	80%	42.98%	0.46%	0.08%
BTMF	0.24	0.22	0.20	0.23	0.39	88.2	122.8	166.3	227.1	481.6	397.4	344.4	304.4	258.3	9.40	12.71	29.56	2.26610	74.18	31.51
RecovDB	0.64	0.63	0.64	0.65	0.67	4083.6	4070.7	4071.5	4030.5	14913.0	143.8	142.8	153.5	479.1	0.38	0.41	1.02	0.01478	88.40	81.51
ORBITS	1.82	1.98	2.12	2.44	2.53	5740.6	5242.4	5113.5	4668.2	3616.2	156.6	174.9	181.2	197.1	0.57	0.64	0.38	0.33022	3.80	37.41
BayOTIDE	1.08	1.48	1.78	1.91	4.79	1128.7	1208.5	1167.8	1469.9	1967.0	479.5	480.6	488.8	579.6	0.68	0.94	2.72	0.31051	23.11	83.66
BRITS	1.24	1.20	1.21	1.30	1.61	3321.9	3407.8	3474.7	3694.5	3923.7	149.9	151.0	151.0	512.2	0.32	0.31	0.57	0.56217	4.78	36.10
SAITS	0.99	1.13	1.20	1.17	1.80	890.9	886.2	780.9	1093.2	1293.0	154.9	197.7	181.0	199.0	0.38	0.37	0.40	0.41787	44.98	30.51
GRIN	0.38	0.38	0.40	0.43	1.37	203.4	233.8	202.3	277.5	2616.6	192.8	195.6	196.9	232.9	0.41	0.29	0.57	0.20008	18.59	31.22
DAMR	0.43	0.39	0.40	0.39	0.60	695.5	680.5	689.0	675.5	907.6	177.5	181.6	183.0	283.8	3.53	5.60	7.04	0.24337	1.53	35.04
E2GAN	1.90	2.12	2.58	2.82	2.77	3319.7	6296.0	13582.4	23246.1	24410.6	169.7	172.8	187.2	313.0	0.50	0.36	1.31	0.31900	3.82	80.52
MIWAE	0.64	0.57	0.58	0.56	0.57	3317.4	3232.0	3257.4	3218.9	3596.5	164.0	156.5	139.5	190.5	0.26	0.23	0.50	0.31296	14.23	37.22
CSDI	0.19	0.22	0.23	0.26	0.87	98.8	108.8	130.7	298.0	2393.6	168.0	195.6	237.6	311.0	0.32	0.35	0.43	0.03517	13.81	37.64
PriSTI	0.17	0.18	0.19	0.32	2.49	89.8	180.9	178.5	419.9	6135.5	234.6	306.7	356.5	226.8	0.29	0.29	2.40	0.17946	50.57	40.58
PCDI	**0.12**	0.13	0.13	0.13	0.16	86.4	97.1	100.7	112.7	243.7	130.2	158.1	189.9	243.2	**0.11**	**0.11**	0.36	0.00523	0.77	**28.58**
PCDIH	**0.11**	**0.12**	**0.12**	**0.12**	**0.15**	**84.9**	**92.1**	**96.1**	**108.2**	**221.1**	**124.6**	**124.4**	**125.1**	**167.5**	**0.11**	**0.11**	**0.35**	**0.00519**	**0.76**	**28.58**

*AirQuality and MIMIC-III contain 13.72% and 21.84% real missing values without ground truth respectively, where we can only evaluate the imputation accuracy more than 20% and 30% missing rates for them. GPS, IMU and Weer have real missing values with labeled ground truth.

Table III
Imputation runtime (in seconds) of the approaches over various datasets with different missing rates

Approach	Energy					Ethanol					AirQuality				MIMIC-III			GPS	IMU	Weer
	10%	20%	30%	40%	80%	10%	20%	30%	40%	80%	20%	30%	40%	80%	30%	40%	80%	42.98%	0.46%	0.08%
BTMF	315.5	316.6	316.7	317.2	498.8	8413.3	7536.4	7470.5	7405.8	8535.2	206.5	225.3	220.7	265.0	284.9	291.5	467.1	7.6	136.5	641.6
RecovDB	0.2	0.2	0.2	0.2	0.3	1.4	1.4	1.5	1.5	1.5	0.2	0.1	0.2	0.2	0.7	0.6	0.5	0.1	0.1	1.5
ORBITS	16.3	16.2	16.3	16.0	14.2	174.5	154.4	151.1	134.7	217.8	16.3	16.5	15.4	170.8	43.8	43.6	80.5	3.5	297.5	7.9
BayOTIDE	172.3	175.6	173.0	175.5	180.3	4274.2	4197.5	4292.5	4256.6	3694.9	109.5	108.0	107.6	94.8	498.4	491.3	345.1	23.2	65.7	2363.7
BRITS	184.2	189.9	197.6	239.6	256.6	13625.7	13459.8	13512.0	14566.5	14497.0	117.6	117.8	116.8	121.9	67.4	66.2	97.5	63.0	33.0	1051.0
SAITS	1.9	1.9	1.9	2.1	4.0	19.4	19.6	19.6	19.4	20.8	4.8	4.8	4.8	4.7	4.6	4.4	6.0	1.1	1.6	16.3
GRIN	278.8	276.6	275.5	277.5	285.2	7317.6	7638.3	7077.1	7362.7	7460.1	31.2	31.3	31.4	33.2	216.6	220.1	288.7	46.4	186.7	604.7
DAMR	1302.3	1301.5	1320.3	1328.4	1317.6	31559.3	31376.8	30492.6	32839.2	60106.3	622.5	620.0	633.6	1555.8	5307.5	5220.9	11859.5	211.2	2039.7	16266.1
E2GAN	978.1	985.0	1090.7	1133.4	1227.2	22267.1	22361.5	22857.9	23358.7	22802.5	543.3	542.5	541.6	589.9	813.1	836.2	821.8	289.6	292.4	12864.1
MIWAE	35.6	33.7	33.9	34.3	33.1	863.1	868.0	878.3	871.0	884.2	6.7	1271.3	6.4	10.0	457.4	496.9	485.7	5.9	25.6	884.7
CSDI	309.5	360.8	407.4	454.0	557.7	13714.1	16234.4	18002.5	19069.7	19123.2	255.5	6.7	289.0	301.7	2066.6	1908.2	2018.6	37.6	154.2	1837.4
PriSTI	391.7	402.7	422.5	433.1	454.0	28785.6	28663.4	24106.8	28638.1	28030.8	499.8	271.6	502.1	548.7	6062.5	6044.8	6118.2	72.1	182.5	4712.8
PCDI	66.7	204.8	443.6	788.6	1953.8	1895.7	2535.8	3364.1	4048.6	5335.5	667.1	502.9	1275.1	1800.7	9667.9	7583.9	11083.9	32.0	11.8	138.7
PCDIH	147.6	865.9	1439.7	2389.5	5025.5	2209.9	3041.5	3761.2	4856.2	5421.1	918.4	982.8	1474.3	2023.3	10600.4	8449.9	12455.2	46.1	12.4	141.6

Figure 2 shows the complete imputation context $C_{41} = \{4.4, 6.4, 1.9, 2.9, 3.6\}$ of x_{41}, and incomplete imputation context $C_{72} = \{x_{71}(\perp), 9.3, 2.0, x_{72}(\perp), 0.3\}$ of x_{72} with null cells x_{71}, x_{72}. From Figure 3, g_1 can be trained over complete imputation contexts C_{31}, C_{41}, C_{51}, having $x_{i1} = 0.2144x_{i2} + 0.1648x_{i3} + 0.7222x_{i-1,1} - 0.0399x_{i+1,1} - 0.3292$, i.e., $\phi_{g_1} \approx (0.21, 0.16, 0.72, -0.04, -0.33)^\top$, where 0.21 and 0.16 denote the contributions of attribute A_2 (Humidity) and attribute A_3 (Temperature) to A_1 (NOx), respectively, i.e., humidity and temperature will affect the NOx concentration through the intratemporal dependency. While 0.72 and -0.04 indicate the influence of previous and subsequent NOx values, respectively. We can also get $\phi_{g_2} \approx (-0.03, 0.03, 0.36, 0.70, 0.02)^\top$ and $\phi_{g_3} \approx (-0.44, -0.32, 0.46, 0.86, -0.02)^\top$ by rounding to two decimal places.

Example 3 (Example 2 continued): Consider the incomplete time series I in Figure 1, and a filling instance I' with $x'_{23} = 1.94, x'_{71} = 6.52, x'_{72} = 0.77$, as well as the dependency models $\mathcal{G}$ in Example 2. For the imputation context C_{21} w.r.t. $g_1 \in \mathcal{G}$, the imputation cost is $(0.5 - 2.07959)^2 \approx (0.5 - 2.08)^2 \approx 2.50$. Considering all the fillings in I', we can obtain the imputation cost $\Theta(I', \mathcal{G}) \approx 2.50 + 0.07 + 0.92 + 0.67 + 0.31 + 4.78 + 1.85 + 0.04 + 0.07 + 0.02 + 0.01 \approx 11.22$. Note that we use the original imputation values and model parameters to compute the final imputation cost here, rather than the rounded two decimal place values.

In addition, we also add the explanations in the other examples, e.g., Examples 4, 6, 7, 8, that some example results are approximate by rounding to two decimal places.

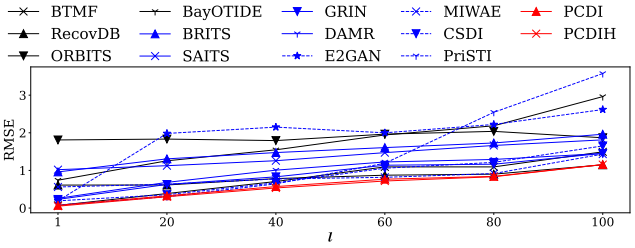


Fig. 7. Varying the maximum missing length l, over Energy with 10% missing values

R4O4: - O4: The adoption of an imputation context improves the model's performance but also makes the approach more fragile. For instance, if a sensor is off for several minutes, causing a cluster of missing values close to each other, this case should be discussed in the experimental section. Intuitively, CDIH should perform better than CDI, but the context window should have a significant impact. Figure 9 shows that the window size has no impact, which is counter-intuitive. Additionally, more details on where the missing values are introduced should be provided.

Reply: Thank you for your insightful comments highlighting the sensitivity of our approaches to clusters of missing values and the impact of the context window size.

(1) In response to this, firstly, to better elucidate the rela-

tionship between the missing value sequence length and the imputation performance, we conduct additional experiments by varying the maximum missing lengths in Figure 7. The experiment incrementally adjusts the maximum missing length from 1 to 100. We observe that the performance of all methods degrades as the maximum length of continuous missing data increases in Figure 7. This phenomenon underlines the challenge in handling extensive gaps in data collection. Notably, despite the performance degradation across all approaches with larger gaps, the methods presented in our study continue to achieve the superior performance.

In addition, the GPS dataset we used before actually contains the consecutive missing values, since the sensors are unavailable for a period of time when collecting the longitude and latitude values. To better investigate the scenario that "sensor is off for several minutes", we also introduce another real-world incomplete dataset, i.e., IMU [49], with consecutive missing values. The consecutive missing data stem from the inertial measurement units not being activated in a period of the recorded time series, which exactly matches the scenario that sensors are off for some time. Table II presents the imputation results of our algorithms and baselines over the two datasets with real-world consecutive missing values. As shown, our algorithms PCDI and PCDIH can obtain the most accurate imputation results. This can be attributed to our methods' capability to encapsulate intertemporal and intratemporal dependencies effectively over imputation contexts, with the convergence guarantee. By modeling these relationships, our methods preserve the ability to perform better even when the sensors are continuously off. We add these new experiments in Figure 7 and Table II with corresponding analysis in Section VII-B.

(2) As for the concern that "Figure 10 (i.e., Figure 9 in previous submission) show that the window size has no impact, which is counter-intuitive", we apologize for the unclear representation of this figure. Lacking the consideration of intertemporal dependencies in the Energy dataset, the imputation RMSE is relatively high when the window size is 0, in previous Figure 9. Therefore, such a higher RMSE than the other cases with larger window sizes, i.e., $\omega = 1, 2, 3, 4, 5$, makes the variation trend unclear, in previous Figure 9. Actually when ω changes, the imputation accuracy also changes in the figure, but the excessive RMSE difference between $\omega = 0$ and $\omega = 1, 2, 3, 4, 5$ makes it hard to distinguish. Therefore, for a clearer representation and requested by the comment in R1O5.E, we add new experiments about the varying temporal window size ω over the GPS dataset with real-world missing values in Figure 10. As shown, with the help of dynamic dependency models, CDIH and PCDIH perform better than CDI and PCDI. Moreover, the imputation RMSE first decreases with an increasing ω, since a mall ω may lead to insufficient temporal data for consideration. However, with the further increase of ω, it may include more attribute values which are not very important to model the temporal dependencies, resulting in overfitting and affecting the imputation results. Therefore, to effectively determine the parameter value in a reasonable way, as replied to R4O1, we add a new Section VI-B to adaptively determine the parameters, including the window size ω, for our algorithms. The window size value ω leading to the best imputation result for the validation set with manually injected missing data will be adopted for imputing missing values in real applications. We add the experiments about the impact of the window size ω in Figure 10 with the aforesaid explanations in Section VII-C.

(3) As for the concern "more details on where the missing values are introduced", we apologize for the insufficient explanations and enhance the corresponding discussions in Section VII-B. Following the same line of the existing study [43], we consider three typical types of missing data injection models, i.e., missing completely at random (MCAR) [11], missing at random (MAR) [68] and missing not at random (MNAR) [64] in Figure 6. As shown, various imputation methods perform similarly under different missing mechanisms and our algorithms consistently achieve the highest accuracy, which verifies the practicability of our work in different missing mechanisms. Therefore, we adopt the MCAR mechanism, which is the most widely used missing data generation strategy [68], to introduce missing values for originally clean datasets, e.g., Energy and Ethanol, by default. As for the real-world incomplete datasets, we directly use the real-world missing values in experiments. We add these explanations for adopting the MCAR mechanism to inject missing data into the originally clean datasets in Section VII-B.

R4O5: - O5: The scalability of the approach is unclear. Table IV reports very different runtime results for PCDI and PCDIH. The authors should discuss these differences. Additionally, the approach is far from being suitable for real-time scenarios (some experiments take 3 hours). In real-life applications, time constraints are very strict.

Reply: Thanks for your valuable feedback regarding the scalability concerns and runtime results of our methods.

(1) In Table III (i.e., Table IV in previous submission), the different runtime results between PCDI and PCDIH primarily stem from the additional complexity introduced by PCDIH. Specifically, PCDIH involves refining dependency model parameters, which inherently requires more computational resources and time. Regarding the extended runtime exceeding 3 hours on the MIMIC-III dataset, it is important to highlight that this is largely due to specific characteristics unique to MIMIC-III. The dataset contains complex medical records with the inherently high missing rate, requiring intricate handling of intratemporal and intertemporal dependencies, posing severe computational challenges. However, this scenario represents an outlier rather than the norm. We add these explanations for the MIMIC-III dataset in Section VII-B.

(2) To further evaluate the scalability and practicality of our algorithms, we conduct additional experiments over the new large-scale Weer dataset (with 1,140,600 tuples) in Table II. This dataset, featuring real-life missing data scenarios, allowing us to benchmark our methods against large-scale data found in everyday applications. As shown, our PCDI and PCDIH take only 138.7 seconds and 141.6 seconds, respec-

tively, over the large-scale Weer dataset, which are faster than most of the baselines. Combining with the experimental results of the memory consumption in Table V in Appendix in [1], our methods have the reasonable scalability on large-scale Ethanol and Weer datasets, which demonstrate the contributions of our parallel imputation strategies. We add the new experiments about the time cost in Table II and the memory consumption in Table V over the additional large-scale Weer dataset, with the aforesaid explanations in Section VII-B.

(3) In addition, we perform real-time imputation experiments over the real-world incomplete datasets with labeled truth, i.e., GPS, IMU and Weer, in Table IV in Section VII-B. We consider the first half of tuples in the dataset as already collected historical data, then conduct the online imputation for the other half of the subsequent tuples in real-time scenarios. Among the baselines, only ORBITS [31] and BayOTIDE [26] are specially designed for the streaming imputation of time series data. The other competing methods, such as BTMF [18], RecovDB [5], and various deep learning models BRITS [16], SAITS [24], GRIN [21], DAMR [54], E2GAN [39], MIWAE [40], CSDI [61], PriSTI [36], cannot be applied for the real-time imputation. The reason is that BTMF and RecovDB require the full dataset view to compute their decompositions and cannot dynamically adapt to new data without recalculating the entire model, which are computationally infeasible in real-time scenarios. Deep learning based methods need retraining or significant adjustments when new data batches are introduced, which becomes a bottleneck in real-time scenarios.

On the contrary, as analyzed in the new Section VI-A, our algorithms can be adapted to the streaming computation, using the imputation contexts and dynamic models. Specifically, for the real-time scenarios, when we collect a tuple $x_i = \{x_i[A_1], x_i[A_2], , \ldots, x_i[A_m]\}$ with missing vales requiring to be imputed online, we first consider the set of imputation contexts that it involves in. For each cell $x_{ij} \in x_i$, according to Definition 1, we know that it is involved into the imputation context $C_{i-\omega,j} = \{x_{i-\omega,h} \mid A_h \in R \setminus \{A_j\}\} \cup \{x_{i-\omega+l,j} | l = -\omega, \ldots, \omega\}$, with the latest tuples $\{x_{i-2w}, x_{i-2w+1}, \ldots, x_i\}$, where ω is the temporal window size. Therefore, for each tuple x_i, the set of imputation contexts containing it can be $\mathcal{C}_{i-\omega} = \{C_{i-\omega,1}, C_{i-\omega,2}, \ldots C_{i-\omega,m}\}$. Let M_i denote the set of missing values in x_i. Considering the pre-trained dependency models $\mathcal{G}$ based on the complete imputation contexts in the historical data, for each missing value $x_{ij} \in M_i$, we can compute the imputation result in the streaming manner by

$$x_{ij}^{\prime(k+1)} \leftarrow x_{ij}^{\prime(k)} - \eta \frac{\partial \Theta(\mathcal{C}_{i-\omega}^{\prime(k)}, \mathcal{G})}{\partial x_{ij}^{\prime(k)}},$$

where k is the imputation step, $x_{ij}^{\prime(k)}$ is the temporary imputation value of x_{ij} and $\mathcal{C}_i^{\prime}$ is the temporary imputation context at the k-th step. On the other hand, with the streamingly incoming tuple x_i, we can also consider refining the parameters of dependency models $\mathcal{G}$ online, having

$$\Phi_{\mathcal{G}^{\prime}}^{(k+1)} \leftarrow \Phi_{\mathcal{G}^{\prime}}^{(k)} - \eta \frac{\partial \Theta(\mathcal{C}_{i-\omega}^{\prime(k)}, \mathcal{G}^{\prime(k)})}{\partial \Phi_{\mathcal{G}^{\prime}}^{(k)}},$$

where $\Phi_{\mathcal{G}^{\prime}}$ is the parameters of dependency models.

As commented, real-time streaming time series imputation is essential in many fields where timely and accurate data are critical for decision making. For transportation and logistics, real-time GPS tracking data of vehicles might suffer from intermittent losses due to signal blockages or device issues. Imputing missing positions and velocities helps in the effective fleet management and route optimization [26]. IoT devices in smart homes or industrial applications generate continuous streams of data regarding the energy usage, machinery status, or environmental conditions [58]. Moreover, real-time imputation also helps in maintaining the integrity of control systems and ensuring operational efficiency even if some data points are missing [31].

As shown in Table IV in Section VII-B, our algorithms process each imputation quickly, at a millisecond level (comparable with specified real-time approaches ORBITS and BayOTIDE), with the higher imputation accuracy. Such results make our algorithms suitable for the aforesaid real-time scenarios, which typically require sub-second level responses [26], [58], [31]. We add these new experiments in Table IV in Section VII-B about the streaming imputation with the analysis into Section VII-B.

R4Other: Other Comments: - Figure 1: It is unclear what the red and blue boxes represent. In Section B (Solution), they are explained in two different ways: as two disjoint dependency models and as the most successful and easy-to-use VAR models.

Reply: Thanks for pointing this out! We apologize for our previous inaccurate illustrations. The red and blue boxes in Figure 1 represent two distinct imputation contexts that can be processed parallelly, where the observations within the red box construct the imputation contexts to impute $x_2[A_3]$, and the observations within the blue box make up the imputation contexts to fill missing values in $x_7[A_1]$ and $x_7[A_2]$. This delineation reflects the efficiency and effectiveness of our parallel algorithms, allowing for simultaneous processing of different data segments within disjoint imputation contexts.

Moreover, to impute missing values in each box, VAR models can be employed to capture the dependencies between temporal data within each designated box, since they are usually used to model the relationships between adjacent data.

We apologize for any confusion caused by the dual explanations before, and revise the corresponding contents in Section I-B, to provide clearer explanations about the roles and representations of these boxes.

R4Availability: Availability: A list of python requirements (in terms of python version and libraries) should be added.

Reply: Thank you for the detailed comments. To improve the availability of our methods, we add the "requirements.txt" file for the list of python requirements (in terms of python version and libraries) in our GitHub repository [1].

Collaborative Imputation for Multivariate Time Series with Convergence Guarantee

Yu Sun †, Xinyu Yang †, Shaoxu Song* §, Ying Zhang †, Xiaojie Yuan †
† *Nankai University*, § *Tsinghua University*
{sunyu@, yangxinyu@dbis., yingzhang@, yuanxj@}nankai.edu.cn, sxsong@tsinghua.edu.cn

***Abstract*—Missing values often occur in multivariate time series, affecting data analysis and applications. Existing studies typically use complete data to train imputation models, which are then used to fill missing values. However, in practice, missing values could appear in various cells. Such varieties unfortunately prevent imputation models performing, even making fillings unavailable without the convergence guarantee, i.e., lacking the ensurance of obtaining the optimal solution when the iteration tends to infinite. The reasons are that (1) the imputed values of multiple cells could affect each other towards the conformance to models, and (2) dependencies obtained from complete data may not be accurate enough to impute many unobserved values, which poses a tougher challenge of the convergence. In this work, we study the collaborative imputation with the convergence guarantee. By "collaborative", we mean (1) all the missing cells can be collaboratively imputed with the guaranteed conformance to models, and (2) the imputation models are collaboratively optimized according to fillings as well. Our major technical highlights include 1) introducing the statistically explainable collaborative imputation via likelihood maximization, 2) designing a collaborative imputation algorithm for multiple missing cells and extending it into a parallel version equivalently, 3) improving the algorithm by both imputation values and models collaboratively optimized with the convergence guarantee in parallel, 4) designing the streaming imputation and adaptive parameter determination strategies. Experiments on real incomplete datasets demonstrate the superiority of our methods against twelve baselines, in both imputation accuracy and downstream applications.**

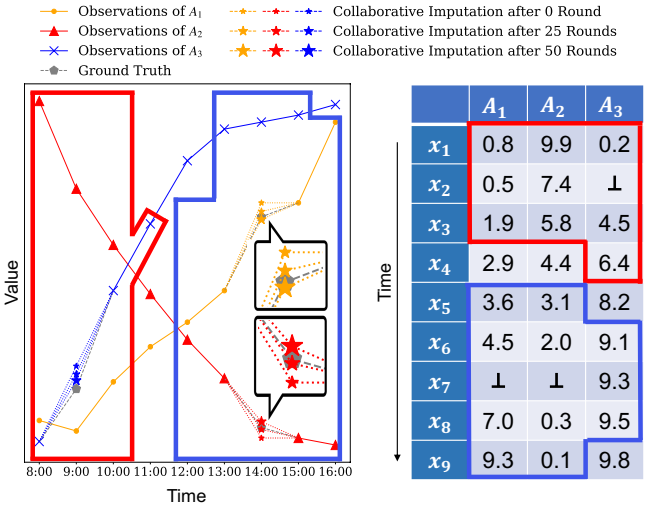
I. Introduction

Multivariate time series involve the values over time consisting of multiple attributes, and are very common in the industrial field [56]. Unfortunately, missing data are often observed due to sensor failures, network outage, etc [35]. Analysis of such missing data may create biased results, misleading downstream applications [55], [28]. Data imputation is thus a necessary process, and it is not surprising that various tasks could benefit from the accurate imputation.

A. Motivation

While existing studies have designed diversified imputation strategies, they still meet intractable challenges of imputing various missing values in multivariate time series.

(1) Missing values could appear in multiple cells of the multivariate time series. For instance, Figure 1 illustrates some example data from the AirQuality dataset on NOx (A_1), Humidity (A_2) and Temperature (A_3) attributes from 8:00 to 16:00 on October 3rd, 2005, with the missing values $x_2[A_3]$, $x_7[A_1]$ and $x_7[A_2]$. During this period, the concentration of NOx increases with the decreasing humidity and the increasing temperature, i.e., the intratemporal dependency. A statistical model [41], [70] or temporal rule [2] may use Humidity (A_2) and Temperature (A_3) values to infer the NOx (A_1) value. However, since both the determinant (a.k.a. left-hand-side) attribute $x_7[A_1]$ and the dependent (a.k.a. right-hand-side) attribute $x_7[A_2]$ are missing, their imputed values could affect each other towards conformance to the dependencies between them. It is thus inaccurate to independently fill them without considering their mutual effects w.r.t. the convergence guarantee, otherwise we cannot get available fillings conforming to the dependencies. For instance, if we assign an inappropriate filling to $x_7[A_2]$, it would also be hard to get an accurate imputation for $x_7[A_1]$, following their dependencies.



	A_1	A_2	A_3
x_1	0.8	9.9	0.2
x_2	0.5	7.4	⊥
x_3	1.9	5.8	4.5
x_4	2.9	4.4	6.4
x_5	3.6	3.1	8.2
x_6	4.5	2.0	9.1
x_7	⊥	⊥	9.3
x_8	7.0	0.3	9.5
x_9	9.3	0.1	9.8

Fig. 1. Example air quality data with missing values denoted by ⊥, and imputed by our work after different rounds

(2) Existing studies (including both traditional methods [7], [2], [18] and deep learning techniques [61], [54], [21]) typically train imputation models over complete data, to capture dependencies and statistics of the entire time series. However, in practice, there may exist many missing cells, making the dependencies obtained only from complete data not accurate enough to impute missing values. Experiments in Table II show that most imputation methods perform poorly on datasets with 80% missing values compared to lower missing rates. Additionally, the convergence guarantee becomes more

*Shaoxu Song (https://sxsong.github.io/) is the corresponding author.

urgent and challenging, with the contradiction of dependencies between complete data and imputed values. For instance, if only those complete values in Figure 1 are involved in training imputation models, they could be insufficient to represent statistics of missing values $x_2[A_3]$, $x_7[A_1]$ and $x_7[A_2]$.

B. Solution

Considering the aforesaid challenges and limitations of existing techniques, we study the collaborative imputation with convergence guarantee to impute missing values in multivariate time series. The benefits are in two aspects.

(1) Both dependent and determinant missing values can be collaboratively imputed with the convergence guarantee, rather than filled independently. For instance, as shown in Figure 1, $x_7[A_1]$ and $x_7[A_2]$ are collaboratively imputed 50 rounds until convergence, where the fillings become more accurate with the increasing rounds. Notably, to improve the imputation efficiency, we further consider the parallel computation strategies for missing cells involved in disjoint dependency models. For instance, the red and blue boxes in Figure 1 represent two distinct imputation contexts that can be processed parallelly, where the observations within the red box construct the imputation contexts to impute $x_2[A_3]$, and the observations within the blue box make up the imputation contexts to fill missing values in $x_7[A_1]$ and $x_7[A_2]$.

(2) Both missing values and imputation models can be collaboratively optimized with the certified convergence, instead of using fixed models. For instance, not only the complete values are utilized for the model training, but also the incomplete cells $x_2[A_3]$, $x_7[A_1]$ and $x_7[A_2]$ are collaboratively involved in the model optimization according to fillings. That is, our work can overcome the assumption of existing studies about the accurate imputation models trained over only complete data. Moreover, the algorithm is also extended into a parallel version, where models are asynchronously updated.

Example 1: Figure 1 show some example data from the AirQuality dataset. Given the temporal window size $\omega = 1$, we establish imputation contexts for missing cells $x_2[A_3]$, $x_7[A_1]$ and $x_7[A_2]$, as marked in red and blue boxes respectively. To impute missing values in each box, the most successful and easy-to-use vector autoregressive (VAR) models [7] can be employed to capture the dependencies between temporal data within each designated box, since they are usually used to model the relationships between adjacent data. The star lines with different sizes show our collaborative imputation results after different rounds, where fillings are initialized by recent attribute values. As for the single missing cell $x_2[A_3]$, both VAR models and our work could obtain a near-optimal filling, according to the dependencies. Unfortunately, when there are multiple missing cells $x_7[A_1]$, $x_7[A_2]$ in both determinant and dependent attributes, directly using VAR is no longer available. In contrast, our collaborative imputation with the convergence guarantee could gradually compute accurate fillings for them until converged.

The example illustrates that, with the convergence guaranteed collaborative imputation, we can optimize both imputation values and models for the multivariate time series.

C. Contributions

Notably, the parallel computation strategy and convergence guarantee are not trivial in existing multivariate time series imputation techniques, which however are very important properties to ensure the efficiency and effectiveness for imputation results. In this sense, we argue that the standard concepts of gradient descent and correlation between attributes would not decrease our main technical contributions as follows.

1) We formalize the likelihood of the imputed multivariate time series w.r.t. dependency models in Section II. The statistically explainable collaborative imputation by the likelihood is then derived, which demonstrates the rationale of our work.

2) We design a sequential collaborative imputation algorithm towards maximizing the likelihood with the convergence guarantee (Proposition 1) in Section IV-B. It is further extended into a parallel version based on whether the imputation contexts are connected in Section IV-C, which ensures to return the same result with the sequential algorithm for fixed updates (Proposition 3).

3) We improve the algorithm with dynamic models to meet the challenge from many missing values in Section V, whose convergence is also ensured (Proposition 4). For efficiency, the algorithm is also improved into a parallel version, with the convergence guarantee (Propositions 6 and 7) w.r.t. both fillings and dynamic models.

4) We consider the optimizations of our algorithms in Section VI, including the streaming imputation for real-time scenarios and the adaptive parameter determination strategy.

Various real incomplete datasets are employed in experiments in Section VII, verifying the superiority of our work.

Appendix with source code and data can be found in [1].

II. Foundations

In this section, we first formalize the imputation contexts and dependency models. The likelihood is then studied for fillings w.r.t. the models. The collaborative imputation for various missing values, which is statistically explainable referring to the maximum likelihood estimation, is formally studied.

A. Imputation Contexts and Dependencies

Consider an incomplete multivariate time series $I = \{x_1, \ldots, x_n\}$ over schema $R = (A_1, \ldots, A_m)$, with a timestamp t_i for each tuple $x_i \in I$. Each $x_i \in I$ contains a collection of cells $\{x_i[A_1], , \ldots, x_i[A_m]\}$, where $x_i[A_j]$, or simply x_{ij}, denotes the value of attribute A_j in the i-th tuple. The *null* cell on attribute A_j at t_i is $x_{ij} = \perp$. A *filling* I' of I is also an instance of R such that existing non-null cells do not change.

To impute missing values in $x_i \in I$, we may refer to the latest ω tuples, e.g., $\{x_{lj} \mid i - \omega \leq l < i + \omega, 1 \leq j \leq m\}$. Because there may exist both intratemporal and intertemporal dependencies in multivariate time series. For instance, in Figure 1, there is the intertemporal dependency

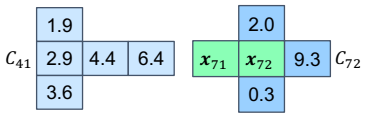


Fig. 2. The complete imputation context C_{41} and the incomplete C_{72}

on temperature data over time, as well as the intratemporal dependency between NOx, humidity and temperature attribute values. Enlightened by the existing study [57] setting a window size to repair time series data, we establish the imputation context for each cell x_{ij}, to abstract such dependencies.

Definition 1 (Imputation Context): The *imputation context* C_{ij} of a cell x_{ij}, $x_i \in I, A_j \in R$, is

$$C_{ij} = \{x_{ih} \mid A_h \in R \setminus \{A_j\}\} \cup \{x_{i+l,j} | l = -\omega, \ldots, \omega\},$$

where ω is the window size of intertemporal dependencies.

From the imputation context C_{ij}, we can exploit the relationship between different attribute values at the same time (by $\{x_{ih} \mid A_h \in R \setminus \{A_j\}\}$), i.e., intratemporal dependencies, as well as the value change between past, present and future events (by $\{x_{i+l,j} | l = -\omega, \ldots, \omega\}$), i.e., intertemporal dependencies, for the cell x_{ij}.

Definition 2 (Dependency Model): For each attribute $A_p \in R$, we consider a *dependency model* g_p. It predicts the value x_{ip} of each tuple $x_i \in I$ on attribute A_p, referring to its imputation context,

$$x_{ip} = g_p(C_{ip} \setminus \{x_{ip}\}) + \varepsilon_p, \tag{1}$$

where ε_p is an error term.

Notably, we can consider not only a linear combination but also a non-linear form, e.g., polynomial regression [9], for the dependency model g_p.

We denote $M = \{x_{ij} | x_{ij} = \perp, x_i \in I, A_j \in R\}$ as the set of null cells in I, $\mathcal{G} = \{g_1, \ldots, g_m\}$ as all dependency models. Following existing studies [22], [72], [55], parameters ϕ_{g_p} of dependency models $g_p \in \mathcal{G}$ can be learned over complete imputation contexts $\{C_{ip} \mid C_{ip} \cap M = \emptyset, i = 1, \ldots, n\}$.

It is notable that our methods do not necessarily assume a strong correlation between all attributes. Rather, our methods are designed to adaptively learn the weights of dependencies among attributes. In our dependency models, the correlations are captured through weights that are adaptively learned from the data. Attributes that are not correlated learn weights that are close to zero, effectively adjusting the model to ignore such relationships. For instance, as illustrated in Figure 3 with our dependency model g_2 for the AirQuality data, the attributes A_1 (NOx) and A_3 (Temperature) contribute minimally to A_2 (Humidity) as reflected by their near-zero weights. The reason for the observed low correlation can be attributed to the nature of these attributes. Humidity is primarily influenced by atmospheric conditions rather than by the presence of nitrogen oxides (NOx) or temperature variations directly. Although temperature can affect the capacity of the air to hold moisture, the direct correlation with specific pollutants like NOx is generally weak without additional environmental factors.

Remark. Unlike the time window based cleaning [20], our dependency models are built on all the complete imputation

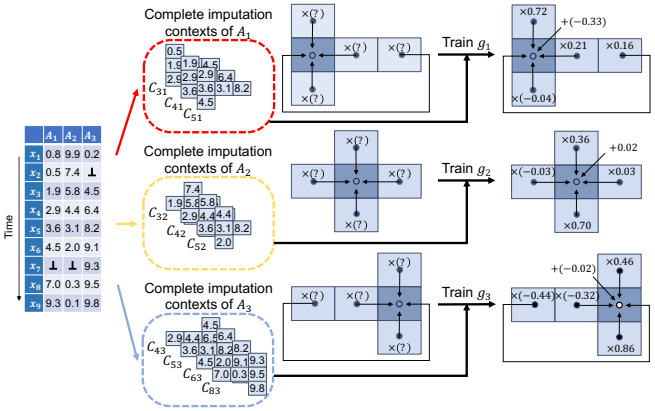


Fig. 3. Intratemporal and intertemporal dependencies over imputation contexts

contexts, rather than just one time window. While each context uses recent data within the window size ω, we argue that long-range dependencies, e.g., the seasonality, are incorporated into our models $\mathcal{G}$. Moreover, as analyzed in Section I-B, although most studies [24], [61], [21] assume the availability of accurate dependency models whose parameters can be learned from the complete data, our dependency models can be collaboratively optimized with the fillings of incomplete data (in Section V), which can thus overcome such an assumption.

Example 2: Consider $I = \{x_1, \ldots, x_9\}$ in Figure 1. Figure 2 shows the complete imputation context $C_{41} = \{4.4, 6.4, 1.9, 2.9, 3.6\}$ of x_{41}, and incomplete imputation context $C_{72} = \{x_{71}(\perp), 9.3, 2.0, x_{72}(\perp), 0.3\}$ of x_{72} with null cells x_{71}, x_{72}. From Figure 3, g_1 can be trained over complete imputation contexts C_{31}, C_{41}, C_{51}, having $x_{i1} = 0.2144x_{i2} + 0.1648x_{i3} + 0.7222x_{i-1,1} - 0.0399x_{i+1,1} - 0.3292$, i.e., $\phi_{g_1} \approx (0.21, 0.16, 0.72, -0.04, -0.33)^\top$, where 0.21 and 0.16 denote the contributions of attribute A_2 (Humidity) and attribute A_3 (Temperature) to A_1 (NOx), respectively, i.e., humidity and temperature will affect the NOx concentration through the intratemporal dependency. While 0.72 and -0.04 indicate the influence of previous and subsequent NOx values, respectively. We can also get $\phi_{g_2} \approx (-0.03, 0.03, 0.36, 0.70, 0.02)^\top$ and $\phi_{g_3} \approx (-0.44, -0.32, 0.46, 0.86, -0.02)^\top$ by rounding to two decimal places.

B. Maximum Likelihood Estimation

Given the observed data and assumed statistical models, the maximum likelihood estimation is a both intuitive and flexible logic. Specifically, in our scenario, we can consider the likelihood estimation of the filling I', referring to the dependency models $\mathcal{G}$.

Let $\mathcal{C}$ and $\mathcal{C}'$ denote the set of all imputation contexts of I and the corresponding filled instance respectively. According to the likelihood theory of the error term [45], the likelihood of a filling I' w.r.t. all the dependency models $\mathcal{G}$ is

$$\mathcal{L}(I' \mid I, \mathcal{G}) = \mathcal{L}(\mathcal{C}' \mid \mathcal{C}, \mathcal{G}) = \sum_{g_p \in \mathcal{G}} \sum_{C'_{ip} \in \mathcal{C}'} \mathcal{L}(C'_{ip} \mid C_{ip}, g_p)$$

$$= \sum_{g_p \in \mathcal{G}} \sum_{C'_{ip} \in \mathcal{C}'} -\frac{\log(2\pi\sigma_p^2)}{2} - \frac{\|x'_{ip} - g_p(C'_{ip} \setminus \{x'_{ip}\})\|^2}{2\sigma_p^2}. \tag{2}$$

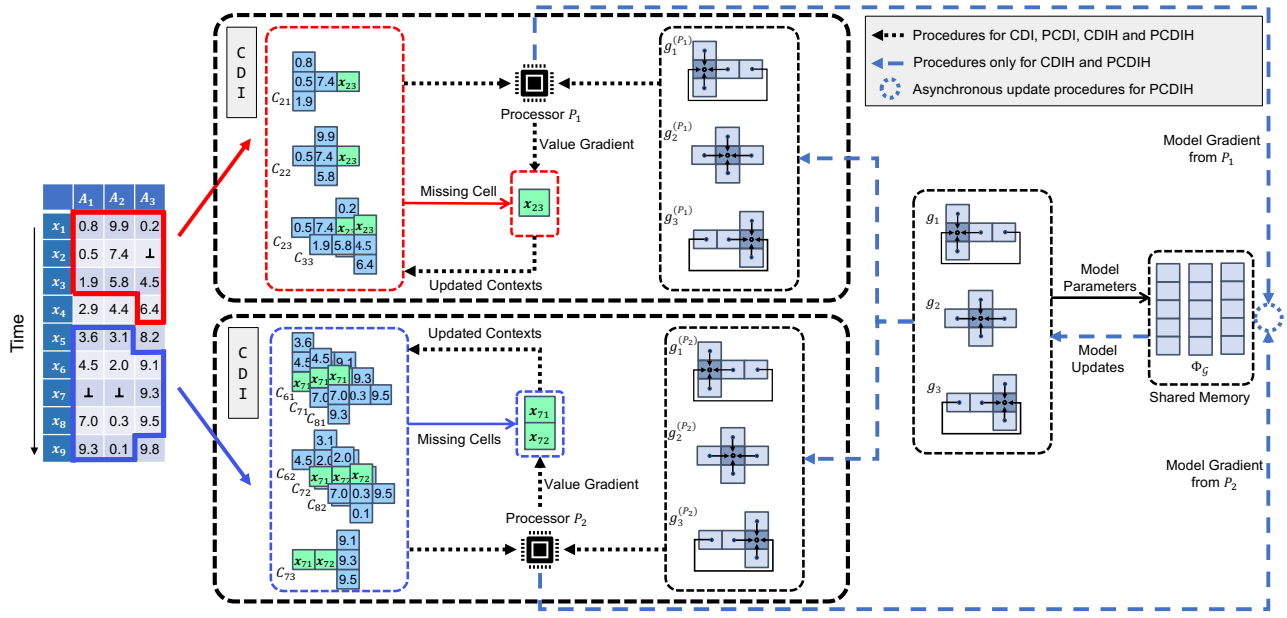


Fig. 4. An overview of our collaborative imputation methods

For an observed time series I, it is desired to find the optimal imputation I' with the maximum likelihood. According to Formula 2, to maximize the likelihood $\mathcal{L}(I' \mid I, \mathcal{G})$, it is equivalent to minimize the conformance loss for I' w.r.t. $\mathcal{G}$,

$$\sum_{g_p \in \mathcal{G}} \sum_{C'_{ip} \in \mathcal{C}'} \|x'_{ip} - g_p(C'_{ip} \setminus \{x'_{ip}\})\|^2. \tag{3}$$

It measures how the filling I' fits dependency models $\mathcal{G}$, by evaluating the least Euclidean deviations (L_2 norm) between the observation x'_{ip} and the prediction $g_p(C'_{ip} \setminus \{x'_{ip}\})$. Thereby, we define the imputation cost of the filling I' accordingly.

Definition 3 (Imputation Cost): The *imputation cost* $\Theta(I', \mathcal{G})$ of a filling I' w.r.t. the dependency models $\mathcal{G}$ is

$$\Theta(I', \mathcal{G}) = \Theta(\mathcal{C}', \mathcal{G}) = \sum_{g_p \in \mathcal{G}} \sum_{C'_{ip} \in \mathcal{C}'} \|x'_{ip} - g_p(C'_{ip} \setminus \{x'_{ip}\})\|^2.$$

Example 3 (Example 2 continued): Consider the incomplete time series I in Figure 1, and a filling instance I' with $x'_{23} = 1.94, x'_{71} = 6.52, x'_{72} = 0.77$, as well as the dependency models $\mathcal{G}$ in Example 2. For the imputation context C_{21} w.r.t. $g_1 \in \mathcal{G}$, the imputation cost is $(0.5 - 2.07959)^2 \approx (0.5 - 2.08)^2 \approx 2.50$. Considering all the fillings in I', we can obtain the imputation cost $\Theta(I', \mathcal{G}) \approx 2.50 + 0.07 + 0.92 + 0.67 + 0.31 + 4.78 + 1.85 + 0.04 + 0.07 + 0.02 + 0.01 \approx 11.22$. Note that we use the original imputation values and model parameters to compute the final imputation cost here, rather than the rounded two decimal place values.

III. Collaborative Imputation Overview

In this section, before explaining our methods in details, we first introduce a high-level overview about our study.

From Figure 4, given the incomplete time series I and dependency models $\mathcal{G}$ learned over complete contexts as input, our methods compute the imputation result I'. Depending on different application scenarios, we provide four different methods, namely Collaborative Data Imputation (CDI), Parallel Collaborative Data Imputation (PCDI), Collaborative Data Imputation for Huge missing values (CDIH), and Parallel Collaborative Data Imputation for Huge missing values (PCDIH).

CDI imputes missing values using dependency models in the single processor, while PCDI further enhances this process into a parallel version using multiple processors. Moreover, since there may exist many missing cells, making the dependency models trained only over complete imputation contexts not accurate enough to capture dependencies of the whole dataset, CDIH consider dynamic models to impute huge missing values, where both imputation values and models are collaboratively optimized. PCDIH similarly extends CDIH into the parallel version, updating dynamic models and fillings collaboratively in multiple processors.

Combining with the experimental results in Section VII, as for the real application scenarios, if there are multiple processors available, PCDI and PCDIH are more suggested than CDI and CDIH. For instance, supercomputers, servers, and personal computers, which typically have multiple processing units, are able to meet the above requirements [4]. Otherwise, if there is only one processor can be used, CDI and CDIH can be applied to impute missing values under such a scenario. For example, low-Cost microcontrollers and embedded devices have such a requirement, since they often use low-cost, low-power and single-core microcontrollers [10]. These devices have limited processing power and therefore typically support only the single-process execution. Moreover, PCDIH (resp. CDIH) is more recommended than PCDI (resp. CDI), when the effectiveness requirement is more favored than the efficiency. For instance, in equipment monitoring and medical health monitoring scenarios [19], there may be large number of missing values, and the requirement for data integrity and imputation accuracy is usually higher than that for the processing speed, due to the sensitivity and complexity

of decision making or data analysis.

In a word, if there are few missing values and only one processor is available, CDI will be suggested. As for few missing values and multiple available processors, PCDI is recommended. Moreover, when there exist many missing values and the efficiency requirement is not strict, CDIH will be chosen if we only have one processor, but PCDIH will be more preferable if there are more available processors.

IV. Collaborative Data Imputation

In this section, we first define the collaborative imputation problem with dependency models. Then a sequential algorithm is designed, with the convergence guarantee (Proposition 1). To enhance the efficiency, we further extend it to a parallel version, which ensures to return the same result with the sequential algorithm for fixed updates (Proposition 3).

A. Problem Statement

For the time series instance with few missing data, imputing them would not significantly conflict with the dependency models $\mathcal{G}$ (in Definition 2) learned from complete imputation contexts (see specific details in Section II-A, with example dependencies in Figures 3). Then the problem is to find the optimal collaborative imputation of the missing data in I, to minimize the imputation cost w.r.t. dependency models $\mathcal{G}$.

Problem 1: Given an incomplete multivariate time series I with dependency models $\mathcal{G}$, the optimal **C**ollaborative **D**ata **I**mputation (**CDI**) problem is to find a filling I' of I, such that the imputation cost $\Theta(I', \mathcal{G})$ is minimized.

As discussed, the optimal filling I^* is statistically explainable. Referring to Formula 2, the filling with the minimum imputation cost $\Theta(I^*, \mathcal{G})$ is the one with the minimum deviation from model predictions $\|x'_{ip} - g_p(C'_{ip} \setminus \{x'_{ip}\})\|^2$, i.e., having the maximum likelihood $\mathcal{L}(I^* \mid I, \mathcal{G})$. That is, a filling most coincide with the underlying intratemporal and intertemporal dependencies. As explained in Section II-A, since dependency models can capture various data characteristics by linear or non-linear forms, we argue that our study can address missing values in complex scenarios and serve real-world applications.

Remark. Even when all values in an imputation context are missing, our methods can still be applied. According to Definition 1, each incomplete cell relates to multiple imputation contexts and has various adjacent reference values. We thus could impute them by minimizing the imputation cost in Definition 3, following both intratemporal and intertemporal dependencies. For instance, if all the values in C_{41} are missing in Figure 3, we can still impute them by considering the adjacent imputation contexts, e.g., C_{31}, C_{51}, C_{42}, C_{43}, etc.

B. Sequential Algorithm

We start with a sequential algorithm for the CDI problem. Existing non-null cells $x_{ij} \neq \perp$ in I remain unchanged, and we focus on imputing the null cells M. As CDI is an unconstrained optimization problem, making gradient descent [50] a natural choice for computing the imputed contexts $\mathcal{C}'$.

Algorithm 1 outlines a sequential imputation strategy that minimizes the imputation cost $\Theta(\mathcal{C}', \mathcal{G})$, as shown in Figure

Algorithm 1: CDI($\mathcal{C}$, $\mathcal{G}$, M)

Input: a set of imputation contexts $\mathcal{C}$ with dependency models $\mathcal{G}$, and the set of null cells M
Output: the imputation result $\mathcal{C}'$ of $\mathcal{C}$

1 $\mathcal{C}' \leftarrow$ initial imputation of $\mathcal{C}$;
2 **while** $\Theta(\mathcal{C}', \mathcal{G})$ *unconverged* **do**
3 **for** *each* $x_{ij} \in \mathcal{C} \cap M$ **do**
4 $x'^{(k+1)}_{ij} \leftarrow x'^{(k)}_{ij} - \eta \frac{\partial \Theta(\mathcal{C}'^{(k)}, \mathcal{G})}{\partial x'^{(k)}_{ij}}$;
5 **return** $\mathcal{C}'$

4. In Line 1, each missing cell $x_{ij} \in \mathcal{C} \cap M$ is initialized with a random value or suggested by existing methods (refer to Section VIII). Let k be the iteration counter, and $x'^{(k)}_{ij}$ is the imputed value of the null cell $x_{ij} \in \mathcal{C} \cap M$ at the k-th iteration, with the learning rate η. Lines 3-4 detail how CDI enumerates all the null cells in $\mathcal{C}$, optimizing their fillings by the gradient descent until the imputation cost $\Theta(\mathcal{C}', \mathcal{G})$ converges or reaches an acceptable level, considering all dependency models $\mathcal{G}$.

By appropriately choosing the learning rate η, we obtain the following convergence guarantee for Algorithm 1.

Proposition 1: Given a learning rate $\eta \leq \epsilon / \left\| \frac{\partial \Theta(\mathcal{C}'^{(k)}, \mathcal{G})}{\partial x'^{(k)}_{ij}} \right\|$, Algorithm 1 returns a filling I' with the imputation cost $\Theta(I', \mathcal{G})$ non-increasing, where ϵ is a small constant.

Please see the complete proof in [1].

This proposition shows that Algorithm 1 can return the optimal solution with the minimum cost $\Theta(I', \mathcal{G})$, when the iteration tends to infinite, i.e., ensuring the convergence.

Example 4: Consider the incomplete time series I in Figure 1, with an initialization $x'^{(0)}_{23} = 2.35, x'^{(0)}_{71} = 7, x'^{(0)}_{72} = 0.3$ in Line 1, as well as the dependency models $\mathcal{G}$ in Example 2. Given $\eta = 0.01$, Lines 3-4 update the filling for each missing cell, having $x'^{(1)}_{23} \approx 2.35 - 0.01 * 1.98 \approx 2.34$, $x'^{(1)}_{71} \approx 7.0 - 0.01 * 4.20 \approx 6.99$, $x'^{(1)}_{72} \approx 0.3 - 0.01 * (-3.28) \approx 0.31$, by rounding to two decimal places. When $k = 50$, the imputation results converge to $x'_{23} \approx 1.89, x'_{71} \approx 6.53, x'_{72} \approx 0.76$.

C. Parallel Algorithm

As analyzed in Section I-A, to enhance the efficiency, we extend the algorithm into a parallel version PCDI, which maintains exact results and convergence guarantees without loss of imputation costs for fixed updates, compared with CDI.

1) Connected Contexts: Since the most costly part of Algorithm 1 is the sequential updates of each null cell $x_{ij} \in \mathcal{C} \cap M$ in Lines 3-4. Therefore, it's intuitive to explore more efficient methods for this process. According to Definition 3, the imputation cost $\Theta(I', \mathcal{G})$ summarizes the costs caused by the contexts $\mathcal{C}'$ of I'. The updated filling of a null cell $x_{ij} \in \mathcal{C} \cap M$ will affect all the costs of those contexts containing it. In this sense, we study which contexts can be imputed in parallel.

Definition 4 (Connected Contexts): Two contexts $C_{ip}, C_{lq} \in \mathcal{C}$ are connected w.r.t the imputation cost if

$$C_{ip} \cap C_{lq} \cap M \neq \emptyset.$$

Two connected contexts, $C_{ip}, C_{lq} \in \mathcal{C}$ share at least one common missing cell in M. Imputing C_{ip} would also affects

the cost caused by C_{lq}, and vice versa. Thus, they cannot be imputed parallelly. It is natural to investigate all the contexts that have to be imputed sequentially, to ensure the exact result of the imputation cost $\Theta(I', \mathcal{G})$.

Definition 5 (Self-Connected Contexts): The self-connected contexts $\mathbf{C}_s \subseteq \mathcal{C}$ consist of those contexts satisfying $\forall C_{ip} \in \mathbf{C}_s$, $\exists C_{lq} \in \mathbf{C}_s \setminus \{C_{ip}\}$, such that $C_{ip} \cap C_{lq} \cap M \neq \emptyset$.

That is, for any context $C_{ip} \in \mathbf{C}_s$, there always exist its connected contexts $C_{lq} \in \mathbf{C}_s$. Therefore, each incomplete context $C_{ip} \in \mathbf{C}_s$ should be imputed in sequential. To impute missing values parallelly, we may divide the dataset into several self-connected contexts, where any two of them share no connected context. It is thus intuitive to construct maximal self-connected contexts as follows, ensuring that each one contains all the contexts having to be conducted in sequential.

Definition 6 (Maximal Self-Connected Contexts): The self-connected contexts $\mathcal{C}_m \subseteq \mathcal{C}$ are *maximal*, if $\forall C_{lq} \in \mathcal{C} \setminus \mathcal{C}_m$, $\nexists C_{ip} \in \mathcal{C}_m$, having $C_{ip} \cap C_{lq} \cap M \neq \emptyset$.

We can identify the maximal self-connected contexts by the Union-Find Algorithm [59] of the Disjoint-set, treating each imputation context with missing values as an independent node. We iterate over each missing value and merge the imputation contexts containing the same missing value into a single set. Each Disjoint-set represents the maximal self-connected contexts. For the joint-set algorithm, we introduce an array for each missing value to store its ancestor node, resulting in a space complexity $O(|M|)$. The time complexity of finding the maximal self-connected contexts set by the joint-set algorithm is $O(|M|\omega m \alpha(|M|))$, where $\alpha(\cdot)$ is the extremely slow-growing inverse Ackermann function [60].

Combining with Definition 3 for the imputation cost, we further study the effect of different orders for imputing two different maximal self-connected contexts.

Proposition 2: For two different maximal self-connected contexts $\mathbf{C}_1, \mathbf{C}_2 \subseteq \mathcal{C}$, they hold

$$\Theta(\mathbf{C}'_1, \mathcal{G} \mid \mathbf{C}'_2) = \Theta(\mathbf{C}'_1, \mathcal{G} \mid \mathbf{C}''_2) = \Theta(\mathbf{C}'_1, \mathcal{G}), \tag{4}$$

where $\mathbf{C}'_1$ is an imputation of $\mathbf{C}_1$, with two different fillings $\mathbf{C}'_2$ and $\mathbf{C}''_2$ of $\mathbf{C}_2$.

That is, the order of filling two maximal self-connected contexts is irrelevant, allowing to impute them in parallel.

Example 5: Consider the incomplete time series I in Figure 1. Since C_{21} and C_{22} share the common missing cell x_{23}, i.e., $C_{21} \cap C_{22} \cap M = x_{23}$, C_{21} and C_{22} are connected contexts. $\mathbf{C}_s = \{C_{21}, C_{22}, C_{33}\}$ is an example of self-connected contexts. From the incomplete time series I, we can get two maximal self-connected contexts $\mathbf{C}_1 = \{C_{21}, C_{22}, C_{23}, C_{33}\}$ and $\mathbf{C}_2 = \{C_{61}, C_{71}, C_{81}, C_{62}, C_{72}, C_{82}, C_{73}\}$.

2) Context Distribution: Given an incomplete I, we could generate several maximal self-connected contexts $\mathcal{C}_m = \{\mathbf{C}_1, \ldots, \mathbf{C}_u\}$. Then they can be distributed to various processors for parallel imputation, according to Proposition 2. However, the challenge is how to distribute the contexts evenly to ensure similar workloads and minimize time costs.

To balance the workload between different processors, a fundamental step is to evaluate the computation cost of different maximal self-connected contexts. Notably, as stated in Section II-A, we only impute null cells, and all the complete values would not be changed. In this sense, for any maximal self-connected contexts $\mathbf{C}_i \in \mathcal{C}_m$, the corresponding computation cost w_i is proportional to the number of null cells in it, and could be defined as $w_i = |\mathbf{C}_i \cap M|$.

Let $\mathbf{S}$ denote the context assignment matrix, where each element $S_{ij} \in \mathbf{S}$ has $S_{ij} = 1$, if the j-th processor P_j is responsible for the maximal self-connected contexts $\mathbf{C}_i$; otherwise $S_{ij} = 0$. The computation cost for each processor $P_j \in \mathbf{P}$ is the total cost of all the maximal self-connected contexts assigned to it, i.e., $\sum_{\mathbf{C}_i \in \mathcal{C}_m} S_{ij} w_i$.

Then we could distribute maximal self-connected contexts between different processors by solving it as the minimum makespan scheduling (MMS) problem [65]. Consider multiple jobs $\mathbf{E}$, where each job $e_i \in \mathbf{E}$ takes time T_i to process, and machines $\mathbf{V}$ on which the jobs are scheduled. For a given scheduling, let $\mathbf{E}_j$ denote the set of jobs assigned to machine $V_j \in \mathbf{V}$, and $\sum_{e_i \in \mathbf{E}_j} T_i$ is the load of the machine V_j. The MMS problem asks for an assignment of jobs to machines that minimizes the makespan, where the makespan is defined as the maximum load over all machines, i.e., $\max_{V_j \in \mathbf{V}} \sum_{e_i \in \mathbf{E}_j} T_i$.

We now present the algorithm DISTRIBUTE($\mathcal{C}_m$, $\mathbf{w}$, $\mathbf{P}$), which distributes the maximal self-connected contexts $\mathcal{C}_m$ with different costs $\mathbf{w}$ to different processors $\mathbf{P}$. Specifically, we interpret the maximal self-connected contexts $\mathcal{C}_m$ as jobs $\mathbf{E}$ and processors $\mathbf{P}$ as machines $\mathbf{V}$ in the MMS problem respectively. For each maximal self-connected contexts $\mathbf{C}_i \in \mathcal{C}_m$, we define its cost $w_i = T_i$, i.e., the time cost in the MMS problem. By using the greedy method [65], it returns a 2-approximation scheduling with $O(uo)$ complexity for the MMS problem. The result forms a context assignment solution $\mathbf{S}$, where $S_{ij} = 1$ for each $e_i \in \mathbf{E}_j$.

3) Putting Techniques Together: We now present the parallel imputation procedure in Algorithm 2, as illustrated in Figure 4. First, we initialize the filling $\mathcal{C}'$ in Line 1, and construct all maximal self-connected contexts $\mathcal{C}_m$ with the corresponding computation costs $\mathbf{w}$, in Line 2. Line 3 calls DISTRIBUTE($\mathcal{C}_m$, $\mathbf{w}$, $\mathbf{P}$) to compute context assignments $\mathbf{S}$. In Lines 5-6, each processor $P_j \in \mathbf{P}$ imputes assigned contexts, by calling CDI($\mathcal{C}_j$, $\mathcal{G}$) Algorithm 1 parallelly. We find that PCDI and CDI algorithms return the same fillings, when different processors converge with the same update step.

Proposition 3: Given the contexts $\mathcal{C}$ and models $\mathcal{G}$, PCDI

Algorithm 2: PCDI($\mathcal{C}$, $\mathcal{G}$, M)

Input: a set of imputation contexts $\mathcal{C}$ with dependency models $\mathcal{G}$, and the set of null cells M
Output: the imputation result $\mathcal{C}'$ of $\mathcal{C}$

1 $\mathcal{C}' \leftarrow$ initial imputation of $\mathcal{C}$;
2 $\mathcal{C}_m$, $\mathbf{w} \leftarrow$ initialization w.r.t. $\mathcal{C}'$ and $\mathcal{G}$;
3 $\mathbf{S} \leftarrow$ DISTRIBUTE($\mathcal{C}_m$, $\mathbf{w}$, $\mathbf{P}$);
4 **for** *each* $P_j \in \mathbf{P}$ **do**
5 $\quad \mathcal{C}_j \leftarrow \{\mathbf{C}_i \mid S_{ij} = 1, S_{ij} \in \mathbf{S}\}$;
6 $\quad \mathcal{C}'_j \leftarrow$ CDI($\mathcal{C}_j$, $\mathcal{G}$);
7 **return** $\mathcal{C}'$

Algorithm 2 returns the same imputation result $\mathcal{C}'$ with CDI Algorithm 1 for the fixed update.

Since it is an exact extension with no loss of imputation costs for fixed updates, i.e., the update step $k_j = k_l$ for any processors $P_j, P_l \in \mathbf{P}$, PCDI retains the convergence guarantee in Proposition 1. It significantly reduces time costs through parallel imputation, with experimental results in Section VII-C showing an average improvement of 76.22% in time costs.

Example 6 (Example 5 continued): Consider the incomplete time series I in Figure 1, with the same initialization as Example 4 and two maximal self-connected contexts $\mathcal{C}_m = \{\mathbf{C}_1, \mathbf{C}_2\}$ in Example 5. Given two processors $\mathbf{P} = \{P_1, P_2\}$, Lines 2-3 in Algorithm 2 compute the context distribution result for maximal self-connected contexts $\mathcal{C}_m = \{\mathbf{C}_1, \mathbf{C}_2\}$. Since the corresponding costs are $w_1 = |\mathbf{C}_1 \cap M| = 1$ and $w_2 = |\mathbf{C}_2 \cap M| = 2$, we can get the assignment matrix $\mathbf{S} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ by solving as the MMS problem, which denotes that $\mathbf{C}_1$ and $\mathbf{C}_2$ are assigned to processors P_1 and P_2 respectively. Given $\eta = 0.01$, Line 6 updates the filling for each missing cell in $\mathbf{C}_1$ (resp. $\mathbf{C}_2$) in processor P_1 (resp. P_2) parallelly, having $x_{23}'^{(1)} \approx 2.35 - 0.01 * 1.98 \approx 2.34$ (resp. $x_{71}'^{(1)} \approx 7.0 - 0.01 * 4.2 \approx 6.99$, $x_{72}'^{(1)} \approx 0.3 - 0.01 * (-3.28) \approx 0.31$). After the fixed update step $k = 50$ with Example 4, we can get the same imputation results $x'_{23} \approx 1.89, x'_{71} \approx 6.53, x'_{72} \approx 0.76$, by rounding to two decimal places.

V. Imputation with Dynamic Models

In practice, there may exist many missing cells, making the dependency models learned over few complete imputation contexts not accurate enough to capture dependencies of the whole dataset. Therefore, in this section, we further study the collaborative imputation with dynamic models, where both fillings and models are collaboratively optimized. We first formally define the problem, and design the sequential algorithm with the guaranteed convergence (Proposition 4). Similarly, to improve the efficiency, we also optimize the algorithm using parallel strategies, where the convergence is also ensured (Propositions 6 and 7).

A. Problem Revisited

Since dependency models trained over complete data may ineligible to describe the dependencies for the whole dataset with many null cells, we collaboratively enhance the models and fillings, to optimize them mutually. Then the problem becomes finding the optimal collaborative imputation of I to minimize the imputation cost w.r.t dynamic models $\mathcal{G}'$, which could be collaboratively optimized with the imputation values.

Problem 2: Given an incomplete multivariate time series I with dependency models $\mathcal{G}$, the optimal **C**ollaborative **D**ata **I**mputation for **H**uge missing values (**CDIH**) problem is to find a filling I' of I, such that the imputation cost $\Theta(I', \mathcal{G}')$ w.r.t dynamic models $\mathcal{G}'$ is minimized.

In addition to updating the fillings of null cells in the CDI Problem 2, CDIH also requires to collaboratively adapt dependency models. The optimal filling I^*, which coincides better with the dynamic models $\mathcal{G}'$ after imputation, is preferred in this case. As shown in Section VII, such an optimization makes our study more applicable for real-world scenarios.

Algorithm 3: CDIH($\mathcal{C}$, $\mathcal{G}$, M)

Input: a set of imputation contexts $\mathcal{C}$ with dependency models $\mathcal{G}$, and the set of null cells M
Output: the imputation result $\mathcal{C}'$ of $\mathcal{C}$

1 $\Phi^{(0)}_{\mathcal{G}'} \leftarrow \Phi_{\mathcal{G}}$;
2 $\mathcal{C}' \leftarrow$ initial imputation of $\mathcal{C}$;
3 **while** $\Theta(\mathcal{C}', \mathcal{G}')$ *unconverged* **do**
4 **for** *each* $x_{ij} \in \mathcal{C} \cap M$ **do**
5 $x_{ij}'^{(k+1)} \leftarrow x_{ij}'^{(k)} - \eta \frac{\partial \Theta(\mathcal{C}'^{(k)}, \mathcal{G}'^{(k)})}{\partial x_{ij}'^{(k)}}$;
6 $\Phi^{(k+1)}_{\mathcal{G}'} \leftarrow \Phi^{(k)}_{\mathcal{G}'} - \eta \frac{\partial \Theta(\mathcal{C}'^{(k)}, \mathcal{G}'^{(k)})}{\partial \Phi^{(k)}_{\mathcal{G}'}}$;
7 **return** $\mathcal{C}'$

B. Sequential Algorithm

To solve the CDIH problem, we also consider the sequential algorithm first. Let $\Phi_{\mathcal{G}}$ (resp. $\Phi_{\mathcal{G}'}$) denote the parameters of models $\mathcal{G}$ (resp. $\mathcal{G}'$). As shown in Algorithm 3, the computation of a filling for each null cell $x_{ij} \in \mathcal{C} \cap M$ is generally similar to Lines 1-4 in Algorithm 1. It is notable that the main difference is the parameter update for dynamic models $\mathcal{G}'$ in Line 6, as shown in Figure 4.

Similar to Proposition 1, for an appropriate learning rate η, the convergence of Algorithm 3 is also guaranteed.

Proposition 4: Given a learning rate

$$\eta \leq min \left\{ \epsilon / \left\| \frac{\partial \Theta(\mathcal{C}'^{(k)}, \mathcal{G}'^{(k)})}{\partial x_{ij}'^{(k)}} \right\|, \epsilon / \left\| \frac{\partial \Theta(\mathcal{C}'^{(k)}, \mathcal{G}'^{(k)})}{\partial \Phi^{(k)}_{\mathcal{G}'}} \right\| \right\},$$

Algorithm 3 returns a filling I' with the imputation cost $\Theta(I', \mathcal{G}')$ non-increasing, where ϵ is a small constant.

Again, such a proposition ensures the convergence of Algorithm 3, which can obtain the optimal filling I' with the minimum imputation cost $\Theta(I', \mathcal{G})$ w.r.t. dynamic models $\mathcal{G}'$, under the scenario that the iteration tends to infinite.

Example 7: Consider the incomplete time series I in Figure 1, dependency models $\mathcal{G}$ in Example 2 and initialization in Example 4. Given $\eta = 0.01$, Line 5 updates the filling for each missing cell, having $x_{23}'^{(1)} \approx 2.35 - 0.01 * 1.98 \approx 2.34$ and $x_{71}'^{(1)} \approx 6.99$, $x_{72}'^{(1)} \approx 0.31$, by rounding to two decimal places. Line 6 also updates parameters $\phi_{g'_p}$ for each model $g'_p \in \mathcal{G}'$, having $\phi^{(1)}_{g'_1} \approx (0.20, 0.17, 0.73, -0.03, -0.32)^\top$, $\phi^{(1)}_{g'_2} \approx (-0.04, 0.02, 0.36, 0.69, 0.01)^\top$, $\phi^{(1)}_{g'_3} \approx (-0.43, -0.31, 0.47, 0.87, -0.01)^\top$. The imputation process converges to the results $x'_{23} \approx 2.16, x'_{71} \approx 6.53, x'_{72} \approx 0.67$, until $k = 50$.

C. Parallel Algorithm

Similar to Section IV-C, we also consider the parallel strategy next. Notably, although the updates for different maximal self-connected contexts would not affect each other and could be conducted in different processors parallelly (Proposition 2), all the dependency models are collaboratively used and

Algorithm 4: PCDIH(C, $\mathcal{G}$, M)

Input: a set of imputation contexts $\mathcal{C}$ with dependency models $\mathcal{G}$, and the set of null cells M
Output: an imputation $\mathcal{C}'$ of $\mathcal{C}$

1 $\Phi_{\mathcal{G}'}^{(0)} \leftarrow \Phi_{\mathcal{G}}$;
2 $\mathcal{C}' \leftarrow$ initial imputation of $\mathcal{C}$;
3 $\mathcal{C}_m$, $\mathbf{w} \leftarrow$ initialization w.r.t. $\mathcal{C}'$ and $\mathcal{G}$;
4 $\mathbf{S} \leftarrow$ DISTRIBUTE($\mathcal{C}_m$, $\mathbf{w}$, $\mathbf{P}$);
5 **for** *each* $P_j \in \mathbf{P}$ **do**
// conduct parallelly
6 $\mathcal{C}_j \leftarrow \{\mathbf{C}_i \mid S_{ij} = 1, S_{ij} \in \mathbf{S}\}$;
7 **while** $\Theta(\mathcal{C}'_j, \mathcal{G}')$ *unconverged* **do**
8 **for** *each* $x_{ij} \in \mathcal{C}_j \cap M$ **do**
9 $x_{ij}'^{(k_j+1)} \leftarrow x_{ij}'^{(k_j)} - \eta \frac{\partial \Theta(\mathcal{C}_j'^{(k_j)}, \mathcal{G}'^{(\kappa-\tau_\kappa)})}{\partial x_{ij}'^{(k_j)}}$;
10 $\Phi_{\mathcal{G}'}^{(\kappa+1)} \leftarrow \Phi_{\mathcal{G}'}^{(\kappa)} - \eta \frac{\partial \Theta(\mathcal{C}_j'^{(k_j)}, \mathcal{G}'^{(\kappa-\tau_\kappa)})}{\partial \Phi_{\mathcal{G}'}^{(\kappa-\tau_\kappa)}}$;
11 **return** $\mathcal{C}'$

updated in all processors. Therefore, the tasks of updating different models $\mathcal{G}'$ cannot be simply distributed to different processors without any communication. However, at the same time, it brings a more serious challenge to the convergence guarantee, which is analyzed in Section V-D.

1) Parallel Dynamic Model Updates: Since the synchronous learning usually causes a communication bottleneck, enlightened by existing studies [53], [76], we adopt an asynchronous learning strategy for dynamic models $\mathcal{G}'$.

Specifically, as illustrated in Figure 5, model parameters $\Phi_{\mathcal{G}'}$ are maintained in the shared memory, which can be accessed and updated by all processors (say P_1 and P_2) without locking. We define τ_κ as the lag between the access and update of the κ-step models $\mathcal{G}'$ (in the shared memory) by the processor P_j. The update rule of $\Phi_{\mathcal{G}'}$ in the shared memory is

$$\Phi_{\mathcal{G}'}^{(\kappa+1)} \leftarrow \Phi_{\mathcal{G}'}^{(\kappa)} - \eta \frac{\partial \Theta(\mathcal{C}_j'^{(k_j)}, \mathcal{G}'^{(\kappa-\tau_\kappa)})}{\partial \Phi_{\mathcal{G}'}^{(\kappa-\tau_\kappa)}}, \tag{5}$$

where η is the learning rate, and contexts $\mathcal{C}_j'^{(k_j)}$ are assigned to the current working processor P_j for its k_j-th updates. For instance, $\tau_\kappa = 0$ from $\Phi_{\mathcal{G}'}^{(\kappa)}$ to $\Phi_{\mathcal{G}'}^{(\kappa+1)}$ by P_1 and $\tau_{\kappa+1} = 1$ from $\Phi_{\mathcal{G}'}^{(\kappa+1)}$ to $\Phi_{\mathcal{G}'}^{(\kappa+2)}$ by P_2. When P_2 updates parameters in the shared memory based on $\Phi_{\mathcal{G}'}^{(\kappa)}$, they have already been updated to $\Phi_{\mathcal{G}'}^{(\kappa+1)}$ by P_1.

2) Parallel Imputation with Dynamic Models: The main procedure of Algorithm 4 is also generally similar to Algorithm 2, despite the asynchronous updates in Line 10 (as illustrated in Figure 4). As explained in Section V-C1, each processor P_j updates model parameters $\Phi_{\mathcal{G}'}^{(\kappa)}$ in the shared memory, based on the gradients w.r.t. its previously available dependencies $\Phi_{\mathcal{G}'}^{(\kappa-\tau_\kappa)}$ and fillings $\mathcal{C}_j'^{(k_j)}$.

Example 8: Consider the incomplete I in Figure 1 and $\mathcal{G}$ in Example 2, with the same initialization in Example 4 and $\mathbf{S}$ in Example 6. Given $\eta = 0.01$, Line 9 updates the fillings with the same results as Example 7 for Line

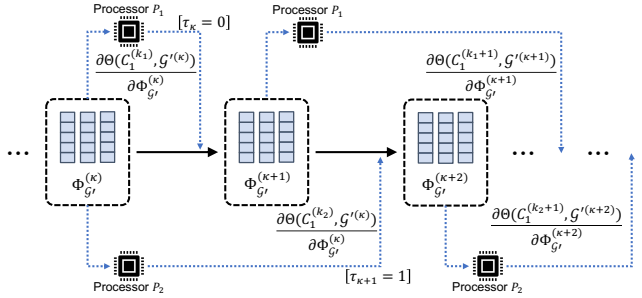


Fig. 5. Asynchronous updates of dynamic models $\Phi_{\mathcal{G}'}$ in the shared memory

5 in Algorithm 3. However, such a procedure is conducted for $\mathbf{C}_1$ in processor P_1 and $\mathbf{C}_2$ in processor P_2 parallelly. In addition, Line 10 will also update the dynamic models. Assuming that $\mathcal{G}$ is first updated by P_1 and then by P_2, we have $\phi_{g_1'}^{(1)} \approx (0.20, 0.15, 0.71, -0.05, -0.34)^\top$, $\phi_{g_2'}^{(1)} \approx (-0.04, 0.02, 0.35, 0.69, 0.01)^\top$, $\phi_{g_3'}^{(1)} \approx (-0.43, -0.31, 0.47, 0.87, -0.01)^\top$, and $\phi_{g_1'}^{(2)} \approx (0.21, 0.16, 0.72, -0.04, -0.33)^\top$, $\phi_{g_2'}^{(2)} \approx (-0.03, 0.03, 0.36, 0.70, 0.01)^\top$, $\phi_{g_3'}^{(2)} \approx (-0.44, -0.32, 0.46, 0.86, -0.02)^\top$. It converges to $x_{23}' \approx 1.94$, $x_{71}' \approx 6.52$, $x_{72}' \approx 0.77$, by rounding to two decimal places.

D. Convergence Analysis

Next we study the convergence guarantee for PCDIH.

Assumption 1: We first make commonly used assumptions:
1. The imputation cost $\Theta(\mathcal{C}'_j, \mathcal{G}')$ for each processor $P_j \in \mathbf{P}$ is L-smooth: There exists a constant L, such that the cost $\Theta(\mathcal{C}'_j, \mathcal{G}'^{(b)}) \leqslant \Theta(\mathcal{C}'_j, \mathcal{G}'^{(a)}) + \frac{\partial\Theta(\mathcal{C}'_j, \mathcal{G}'^{(a)})}{\partial\Phi_{\mathcal{G}'}^{(a)}}(\Phi_{\mathcal{G}'}^{(b)} - \Phi_{\mathcal{G}'}^{(a)}) + \frac{L}{2}\|\Phi_{\mathcal{G}'}^{(b)} - \Phi_{\mathcal{G}'}^{(a)}\|^2$, or $\left\|\frac{\partial\Theta(\mathcal{C}'_j, \mathcal{G}'^{(b)})}{\partial\Phi_{\mathcal{G}'}^{(b)}} - \frac{\partial\Theta(\mathcal{C}'_j, \mathcal{G}'^{(a)})}{\partial\Phi_{\mathcal{G}'}^{(a)}}\right\| \leqslant L\|\Phi_{\mathcal{G}'}^{(b)} - \Phi_{\mathcal{G}'}^{(a)}\|, \forall a, \forall b$. This assumption is reasonable and common in previous works, including imputation methods [14], objective functions [13] and parallel optimizations [53], [76]. Please see the empirical evidence in Appendix.H in [1].
2. There exists a constant T such that $max\{\tau_\kappa \mid \kappa = 0, \ldots, K-1\} \leqslant T$, where K is total number of update steps to $\Phi_{\mathcal{G}}$ for all processors. This assumption is not only intuitive but also well-founded within the context of asynchronous updates and distributed data environment [8]. It aligns with typical network and processing constraints that naturally occur in distributed systems.
3. There exists a constant V such that $\left\|\frac{\partial\Theta(\mathcal{C}'_j, \mathcal{G}')}{\partial\Phi_{\mathcal{G}'}}\right\| \leqslant V$. It implies that the gradient's influence does not exceed a certain threshold [76]. Since the imputation contexts in each processor are assigned based on the size of the maximal self-connected contexts, each processor is assigned those contexts with different timestamps and attributes, making this assumption reasonable.

We show that the gap between $\Phi_{\mathcal{G}'}^{(\kappa)}$ and $\Phi_{\mathcal{G}'}^{(\kappa-\tau_\kappa)}$ is bounded.

Lemma 5: The gap of $\Phi_{\mathcal{G}'}^{(\kappa)}$ and $\Phi_{\mathcal{G}'}^{(\kappa-\tau_\kappa)}$ has an upper bound

$$\mathbb{E}\|\Phi_{\mathcal{G}'}^{(\kappa)} - \Phi_{\mathcal{G}'}^{(\kappa-\tau_\kappa)}\|^2 \leqslant T^2\eta^2V^2.$$

Next, we analyze the convergence of PCDIH algorithm.

Proposition 6: The sum of model gradients is bounded by

$$\sum_{\kappa=0}^{K-1} \mathbb{E} \left\| \frac{\partial \Theta(\mathcal{C}', \mathcal{G}'^{(\kappa)})}{\partial \Phi_{\mathcal{G}'}^{(\kappa)}} \right\|^2 \leqslant \frac{2\Theta\left(\mathcal{C}', \mathcal{G}'^{(0)}\right)}{\eta} + \eta o L K V^2 (\eta L T^2 + 1),$$

where $o = |\mathbf{P}|$ is the number of processors.

Proposition 7: The sum of filling gradients is bounded by

$$\sum_{j=1}^{o} \sum_{k_j=0}^{\kappa_j - 1} \sum_{x_{ij} \in \mathcal{C}_j} \left\| \frac{\partial \Theta(\mathcal{C}_j'^{(k_j)}, \mathcal{G}')}{\partial x_{ij}'^{(k_j)}} \right\|^2 \leq \frac{\Theta\left(\mathcal{C}'^{(0)}, \mathcal{G}'\right)}{\eta},$$

where κ_j is total number of updates by processor P_j.

Propositions 6 and 7 bound the sum of gradients w.r.t. dependency models and fillings respectively. That is, the imputation cost $\Theta(I', \mathcal{G})$ will converge after enough iterations, i.e., reflecting the convergence guarantee for PCDIH.

VI. Optimizations

A. Streaming Imputation

For the real-time scenarios, when we collect a tuple $x_i = \{x_i[A_1], x_i[A_2], , \ldots, x_i[A_m]\}$ with missing vales requiring to be imputed online, we first consider the set of imputation contexts that it involves in. For each cell $x_{ij} \in x_i$, according to Definition 1, we know that it is involved into the imputation context $C_{i-\omega,j} = \{x_{i-\omega,h} \mid A_h \in R \setminus \{A_j\}\} \cup \{x_{i-\omega+l,j} | l = -\omega, \ldots, \omega\}$, with the latest tuples $\{x_{i-2w}, x_{i-2w+1}, \ldots, x_i\}$, where ω is the temporal window size. Therefore, for each tuple x_i, the set of imputation contexts containing it can be $\mathcal{C}_{i-\omega} = \{C_{i-\omega,1}, C_{i-\omega,2}, \ldots C_{i-\omega,m}\}$. Let M_i denote the set of missing values in x_i. Considering the pre-trained dependency models $\mathcal{G}$ based on the complete imputation contexts in the historical data, for each missing value $x_{ij} \in M_i$, we can compute the imputation result in the streaming manner by

$$x_{ij}'^{(k+1)} \leftarrow x_{ij}'^{(k)} - \eta \frac{\partial \Theta(\mathcal{C}_{i-\omega}'^{(k)}, \mathcal{G})}{\partial x_{ij}'^{(k)}},$$

where k is the imputation step, $x_{ij}'^{(k)}$ is the temporary imputation value of x_{ij} and $\mathcal{C}_i'$ is the temporary imputation context at the k-th step. On the other hand, with the streamingly incoming tuple x_i, we can also consider refining the parameters of dependency models $\mathcal{G}$ online, having

$$\Phi_{\mathcal{G}'}^{(k+1)} \leftarrow \Phi_{\mathcal{G}'}^{(k)} - \eta \frac{\partial \Theta(\mathcal{C}_{i-\omega}'^{(k)}, \mathcal{G}'^{(k)})}{\partial \Phi_{\mathcal{G}'}^{(k)}},$$

where $\Phi_{\mathcal{G}'}$ is the parameters of dependency models.

B. Parameter Determination

Practically, for a real incomplete time series I with missing cells M, to determine an appropriate value of the algorithmic parameter ω or η, users could run our methods with varying parameter values and choose the one leading to the highest imputation accuracy over the validation set with manually injected missing values. Specifically, in addition to the missing values in M, we can manually introduce more missing cells in I, denoted as M_d, where $M_d \cap M = \emptyset$. Notably, since the ground truth of these missing values in M_d is available and they are unbiasedly adopted from I, as the same as the missing values in M, we can use M_d as the validation set to determine the parameter values of our algorithms. Then we can consider a candidate set for each parameter, say $\mathrm{can}(\omega)$ for the window size ω and $\mathrm{can}(\eta)$ for the learning rate η. To determine the parameter values of our algorithms, i.e., CDI, PCDI, CDIH, PCDIH, we can impute the missing values in M_d by the algorithm using different candidate parameter values in $\mathrm{can}(\omega)$ or $\mathrm{can}(\eta)$. Then the parameter value leading to the higher imputation accuracy over the validation set M_d is determined for the algorithm. The reason is that a high imputation accuracy over the validation set with manually injected missing values indicates the good performance of our algorithms in such an application, which is believed to achieve a good imputation result over real missing data as well.

TABLE I
Dataset summary

Dataset	$\lvert I\rvert$	$\lvert R\rvert$	Missing Value	Missing Rate ($\frac{\lvert M\rvert}{\lvert I\rvert \cdot \lvert R\rvert}$)
Energy	19,735	9	artificial	-
Ethanol	917,000	3	artificial	-
AirQuality	9,357	13	real, no truth	13.72%
MIMIC-III	79,700	6	real, no truth	21.84%
GPS	3,155	8	real, labeled truth	42.98%
IMU	13,939	6	real, labeled truth	0.46%
Weer	1,140,600	5	real, labeled truth	0.08%

VII. Experiment

In this section, we study the following objectives:
1) We evaluate our algorithms against twelve baselines, over seven datasets with different missing rates, mechanisms, maximum missing lengths, and real-time scenarios.
2) We measure the scalability and comparative performance of our work, for varying # processors, # dependency models, temporal window size, learning rate, and iterations.
3) We validate the effectiveness in downstream applications.

A. Experimental Setup

Experiments run on a machine with 2.5 GHz Intel Core i9 CPU, 16 GB 2933 MHz Memory. Seven datasets are used in experiments. Table I summarizes their main characteristics.

Energy [15] is monitored through a ZigBee wireless sensor network. **Ethanol** [34] is a raw spectral dataset of water and ethanol solutions in whisky bottles. **AirQuality** [23] collects sensor readings from March 2004 to February 2005 in Italy. **MIMIC-III** [30] contains de-identified clinical data collected by Beth Israel Deaconess Medical Center from 2001 to 2012. **GPS** is manually collected by carrying a GPS device when traveling by subway. GPS readings can be temporally unavailable due to weak signals. As we know exactly the trajectory, truth values of real missing data are manually labeled. **IMU** [49] consists of observations from the inertial measurement units. Since the sensors can be turned off and have precision problems, the observations generated by the low-precision gyroscopes may contain missing values, whose ground truth data are thus labeled by the higher-precision sensors. **Weer** [62] records weather data for Soesterberg, Nertherland, from

TABLE II

IMPUTATION RMSE OF OUR METHODS COMPARED TO THE EXISTING APPROACHES OVER VARIOUS DATASETS WITH DIFFERENT MISSING RATES

Approach	Energy					Ethanol					AirQuality				MIMIC-III			GPS	IMU	Weer
	10%	20%	30%	40%	80%	10%	20%	30%	40%	80%	20%	30%	40%	80%	30%	40%	80%	42.98%	0.46%	0.08%
BTMF	0.24	0.22	0.20	0.23	0.39	88.2	122.8	166.3	227.1	481.6	397.4	344.4	304.4	258.3	9.40	12.71	29.56	2.26610	74.18	31.51
RecovDB	0.64	0.63	0.64	0.65	0.67	4083.6	4070.7	4071.5	4030.5	14913.0	143.8	142.8	153.5	479.1	0.38	0.41	1.02	0.01478	88.40	81.51
ORBITS	1.82	1.98	2.12	2.44	2.53	5740.6	5242.4	5113.5	4668.2	3616.2	156.6	174.9	181.2	197.1	0.57	0.64	0.38	0.33022	3.80	37.41
BayOTIDE	1.08	1.48	1.78	1.91	4.79	1128.7	1208.5	1167.8	1469.9	1967.0	479.5	480.6	488.8	579.6	0.68	0.94	2.72	0.31051	23.11	83.66
BRITS	1.24	1.20	1.21	1.30	1.61	3321.9	3407.8	3474.7	3694.5	3923.7	149.9	151.0	151.0	512.2	0.32	0.31	0.57	0.56217	4.78	36.10
SAITS	0.99	1.13	1.20	1.17	1.80	890.9	886.2	780.9	1093.2	1293.0	154.9	197.7	181.0	199.0	0.38	0.37	0.40	0.41787	44.98	30.51
GRIN	0.38	0.38	0.40	0.43	1.37	203.4	233.8	202.3	277.5	2616.6	192.8	195.6	196.9	232.9	0.41	0.29	0.57	0.20008	18.59	31.22
DAMR	0.43	0.39	0.40	0.39	0.60	695.5	680.5	689.0	675.5	907.6	177.5	181.6	183.0	283.8	3.53	5.60	7.04	0.24337	1.53	35.04
E2GAN	1.90	2.12	2.58	2.82	2.77	3319.7	6296.0	13582.4	23246.1	24410.6	169.7	172.8	187.2	313.0	0.50	0.36	1.31	0.31900	3.82	80.52
MIWAE	0.64	0.57	0.58	0.56	0.57	3317.4	3232.0	3257.4	3218.9	3596.5	164.0	156.5	139.5	190.5	0.26	0.23	0.50	0.31296	14.23	37.22
CSDI	0.19	0.22	0.23	0.26	0.87	98.8	108.8	130.7	298.0	2393.6	168.0	195.6	237.6	311.0	0.32	0.35	0.43	0.03517	13.81	37.64
PriSTI	0.17	0.18	0.19	0.32	2.49	89.8	180.9	178.5	419.9	6135.5	234.6	306.7	356.5	226.8	0.29	0.29	2.40	0.17946	50.57	40.58
PCDI	**0.12**	0.13	0.13	0.13	0.16	86.4	97.1	100.7	112.7	243.7	130.2	158.1	189.9	243.2	**0.11**	**0.11**	0.36	0.00523	0.77	**28.58**
PCDIH	**0.11**	**0.12**	**0.12**	**0.12**	**0.15**	**84.9**	**92.1**	**96.1**	**108.2**	**221.1**	**124.6**	**124.4**	**125.1**	**167.5**	**0.11**	**0.11**	**0.35**	**0.00519**	**0.76**	**28.58**

*AirQuality and MIMIC-III contain 13.72% and 21.84% real missing values without ground truth respectively, where we can only evaluate the imputation accuracy more than 20% and 30% missing rates for them. GPS, IMU and Weer have real missing values with labeled ground truth.

TABLE III

IMPUTATION RUNTIME (IN SECONDS) OF THE APPROACHES OVER VARIOUS DATASETS WITH DIFFERENT MISSING RATES

Approach	Energy					Ethanol					AirQuality				MIMIC-III			GPS	IMU	Weer
	10%	20%	30%	40%	80%	10%	20%	30%	40%	80%	20%	30%	40%	80%	30%	40%	80%	42.98%	0.46%	0.08%
BTMF	315.5	316.6	316.7	317.2	498.8	8413.3	7536.4	7470.5	7405.8	8535.2	206.5	225.3	220.7	265.0	284.9	291.5	467.1	7.6	136.5	641.6
RecovDB	0.2	0.2	0.2	0.2	0.3	1.4	1.4	1.5	1.5	1.5	0.2	0.1	0.2	0.2	0.7	0.6	0.5	0.1	0.1	1.5
ORBITS	16.3	16.2	16.3	16.0	14.2	174.5	154.4	151.1	134.7	217.8	16.3	16.5	15.4	170.8	43.8	43.6	80.5	3.5	297.5	7.9
BayOTIDE	172.3	175.6	173.0	175.5	180.3	4274.2	4197.5	4292.5	4256.6	3694.9	109.5	108.0	107.6	94.8	498.4	491.3	345.1	23.2	65.7	2363.7
BRITS	184.2	189.9	197.6	239.6	256.6	13625.7	13459.8	13512.0	14566.5	14497.0	117.6	117.8	116.8	121.9	67.4	66.2	97.5	63.0	33.0	1051.0
SAITS	1.9	1.9	1.9	2.1	4.0	19.4	19.6	19.6	19.4	20.8	4.8	4.8	4.8	4.7	4.6	4.4	6.0	1.1	1.6	16.3
GRIN	278.8	276.6	275.5	277.5	285.2	7317.6	7638.3	7077.1	7362.7	7460.1	31.2	31.3	31.4	33.2	216.6	220.1	288.7	46.4	186.7	604.7
DAMR	1302.3	1301.5	1320.3	1328.4	1317.6	31559.3	31376.8	30492.6	32839.2	60106.3	622.5	620.0	633.6	1555.8	5307.5	5220.9	11859.5	211.2	2039.7	16266.1
E2GAN	978.1	985.0	1090.7	1133.4	1227.2	22267.1	22361.5	22857.9	23358.7	22802.5	543.3	542.5	541.6	589.9	813.1	836.2	821.8	289.6	292.4	12864.1
MIWAE	35.6	33.7	33.9	34.3	33.1	863.1	868.0	878.3	871.0	884.2	6.7	1271.3	6.4	10.0	457.4	496.9	485.7	5.9	25.6	884.7
CSDI	309.5	360.8	407.4	454.0	557.7	13714.1	16234.4	18002.5	19069.7	19123.2	255.5	6.7	289.0	301.7	2066.6	1908.2	2018.6	37.6	154.2	1837.4
PriSTI	391.7	402.7	422.5	433.1	454.0	28785.6	28663.4	24106.8	28638.1	28030.8	499.8	271.6	502.1	548.7	6062.5	6044.8	6118.2	72.1	182.5	4712.8
PCDI	66.7	204.8	443.6	788.6	1953.8	1895.7	2535.8	3364.1	4048.6	5335.5	667.1	502.9	1275.1	1800.7	9667.9	7583.9	11083.9	32.0	11.8	138.7
PCDIH	147.6	865.9	1439.7	2389.5	5025.5	2209.9	3041.5	3761.2	4856.2	5421.1	918.4	982.8	1474.3	2023.3	10600.4	8449.9	12455.2	46.1	12.4	141.6

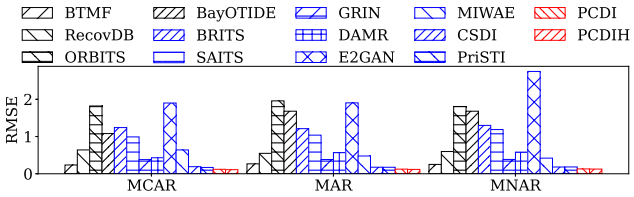


Fig. 6. Varying the missing mechanism, over Energy with 10% missing values

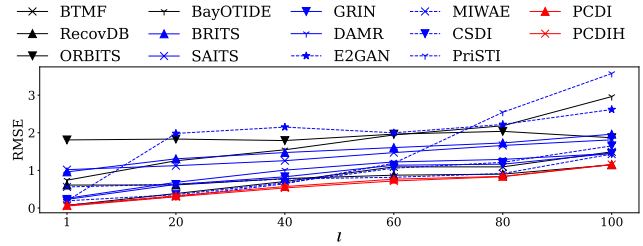


Fig. 7. Varying the maximum missing length l, over Energy with 10% missing values

1951 to 2010. Due to station relocations and sensor failures, there exist real-world missing data in the collected records, and we obtain the ground truth values from De Bilt, the closest neighbor station to Soesterberg.

B. Comparison with Competing Methods

Figure 6 illustrates the experimental evaluation with three missing mechanisms, i.e., missing completely at random (MCAR) [11], missing at random (MAR) [68] and missing not at random (MNAR) [64], which have been widely used as a setup for exploring different missing scenarios [43], [47], [52]. As shown, various imputation methods perform similarly under different missing mechanisms and our algorithms consistently achieve the highest accuracy, which verifies the practicability of our work in different missing mechanisms. Therefore, we adopt the MCAR mechanism, which is the most widely used missing data generation strategy [68], to introduce missing values for originally clean datasets, e.g., Energy and Ethanol, by default. We argue that such a missing mechanism does not contradict the existence of dependencies between attributes. Instead, it proves the robustness and the effectiveness of our algorithms, emphasizing the dependencies between the attributes, to deal with various missing values.

TABLE IV

STREAMING IMPUTATION RMSE AND AVERAGED PROCESSING RUNTIME FOR EACH MISSING TUPLE OVER REAL-WORLD INCOMPLETE DATASETS

Approach	GPS		IMU		Weer	
	RMSE	Runtime (ms)	RMSE	Runtime (ms)	RMSE	Runtime (ms)
ORBITS	0.3302	1.63	23.05	59.38	43.18	28.01
BayOTIDE	0.3122	7.35	4.66	47.13	87.40	7.29
CDI	0.0056	1.27	1.37	36.59	30.24	16.58
CDIH	0.0054	1.89	1.37	54.85	30.23	18.73

Table II explores the performance of different imputation methods with various missing rates. We can find that matrix factorization (MF) based BTMF [18] may struggle with imputing various missing data due to the reliance on low-rank approximations. Centroid decomposition (CD) based RecovDB [5] and ORBITS [31] may fall short as centroid decomposition may miss minor features critical for data imputation. ORBITS prioritizes the online imputation, potentially reducing accuracy compared to RecovDB. As for Bayesian based BayOTIDE [26], its use of Gaussian Processes with smooth and periodic kernels may not suit complex time series data. Deep learning models BRRTS [16], SAITS [24], GRIN [21], E2GAN [39], MIWAE [40], CSDI [61] and PriSTI [36] often require time series segmentation, limiting their effectiveness with long gaps. Since we can not get adjacency matrices for spatiotemporal imputation methods GRIN, DAMR [54] and PriSTI, they do not perform well. Benefiting from the models capturing intertemporal and intratemporal dependencies, with the con-

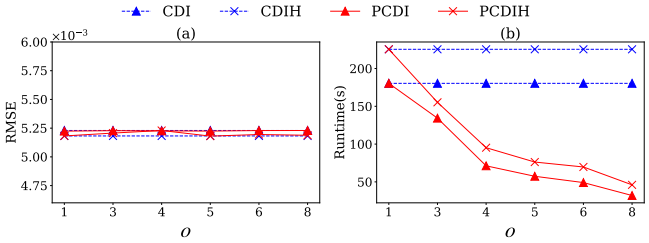


Fig. 8. Varying the number of processors o, over GPS

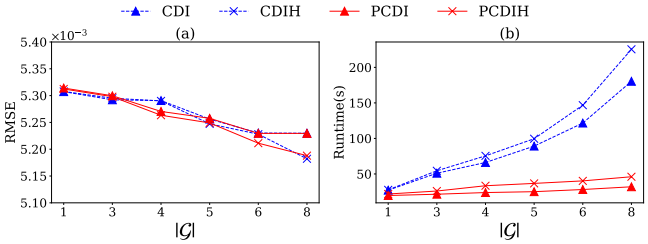


Fig. 9. Varying the number of dependency models $|\mathcal{G}|$, over GPS

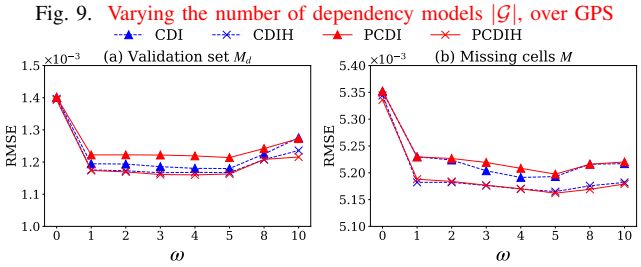


Fig. 10. Varying the temporal window size ω, over the validation set M_d and real-world missing cells M of GPS

vergence guarantee to ensure a collaborative imputation, our methods show the superiority over various datasets. Notably, our algorithms can also obtain the most accurate imputation over GPS and IMU datasets with consecutive missing values, which demonstrate the ability to perform better even when the sensors are continuously off.

Table III reports the runtime results of various imputation methods. While DL-based techniques consume excessive time due to their large number of parameters, our algorithms do not have such requirements. Regarding the extended runtime of our algorithms on the MIMIC-III dataset, it is important to highlight that this is largely due to specific characteristics unique to MIMIC-III. The dataset contains complex medical records with the inherently high missing rate, requiring intricate handling of intratemporal and intertemporal dependencies, posing severe computational challenges. However, this scenario represents an outlier rather than the norm. Furthermore, we find that PCDI and PCDIH take only 138.7 seconds and 141.6 seconds respectively over the large-scale Weer dataset, which are faster than most of the baselines. Combining with the experimental results of the memory consumption in Table V in Appendix in [1], our methods have the reasonable scalability on large-scale Ethanol and Weer datasets.

As shown in Figure 7, the performance of all methods degrades as the maximum length of continuous missing data increases. This phenomenon underlines the challenge in handling extensive gaps in data collection. Notably, despite the performance degradation across all approaches with larger gaps, the methods presented in our study continue to achieve the superior performance.

Table IV shows the experiments for processing missing values in the streaming manner. We consider the first half of tuples in the dataset as already collected historical data, then conduct the online imputation for the other half of the subsequent tuples in real-time scenarios. Among the baselines, only ORBITS and BayOTIDE are specially designed for the streaming imputation of time series data. The other competing methods cannot be applied for the real-time imputation. Table IV show that our algorithms process each imputation quickly, at a millisecond level (comparable with specified real-time approaches ORBITS and BayOTIDE), with the higher imputation accuracy. Such results make our algorithms suitable for the real-time scenarios, which typically require sub-second level responses [26], [58], [31].

C. Performance of Our Methods

As shown in Figure 8, the variation of processors does not significantly affect the imputation accuracy, since models are guaranteed to converge under different settings (Propositions 1, 3, 4, 6, 7). We also observe that PCDI (resp. PCDIH) may produce slightly different results compared to CDI (resp. CDIH), due to the parallel processing nature of PCDI (resp. PCDIH). Such differences may arise because the imputation contexts assigned to each processor are different, so that the number of iterations on different processors may be slightly different. However, it is important to highlight that these minor differences do not significantly impact the overall performance or the final convergence of algorithms. Our theoretical propositions ensure that, despite these minor variations, the convergence behavior remains robust, leading to consistent and reliable imputation results. Our parallel methods come with the convergence guarantees that allow them to autonomously determine the point of convergence and cease the training accordingly. This automatic stopping mechanism negates the need to forcefully align the update steps with those used in non-parallel CDI and CDIH, as doing so does not constitute a condition for convergence. Moreover, both PCDI and PCDIH show reduced runtimes with more processors, and PCDI typically performs faster due to the absence of dynamic model updates.

Then we investigate the imputation results with different numbers of dependency models in Figure 9. Generally, more dependency models improves the imputation accuracy. Given PCDIH's asynchronous dependency model updates and the variations in imputing values across processors, incorporating extensive intratemporal and intertemporal dependencies enhances the accuracy. Therefore, we typically set the number of dependency models $|\mathcal{G}|$ to the attribute number m by default.

Figure 10 presents experimental results for varying temporal window sizes ω. The imputation quality is poorest at $\omega = 0$, where no intertemporal dependency is included. The imputation RMSE first decreases with an increasing ω, since a mall ω may lead to insufficient temporal data for consideration. However, with the further increase of ω, it may include more attribute values which are not very important to model the temporal dependencies, resulting in overfitting and affecting

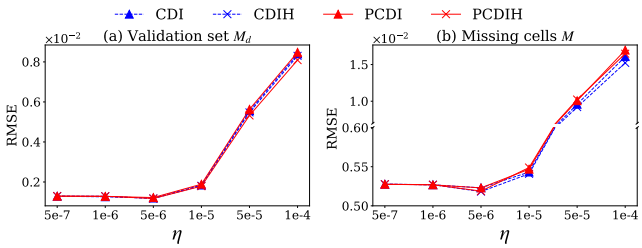


Fig. 11. Varying the learning rate η, over the validation set M_d and real-world missing cells M of GPS

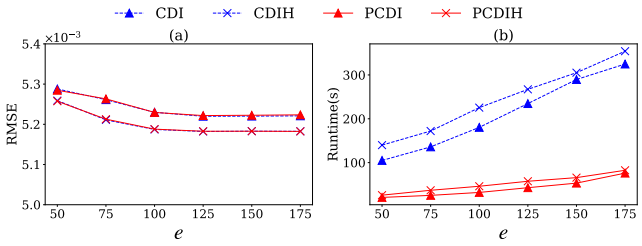


Fig. 12. Varying the number of iterations e, over GPS

the imputation results. We thus can determine the appropriate window size ω in reality, following Section VI-B. As shown, the ω value leading to the highest accuracy in the validation set in Figure 10(a) generally returns the best imputation results for the real missing cells in Figure 10(b), which shows the rationale of our determination strategy.

Next, we study the imputation performance with various learning rates η in Figure 11. We can observe that a large η makes approaches difficult to converge, with the lower accuracy. While a small η may result in the local minima of models. To address these extremes, we also study its determination strategy according to Section VI-B. We observe that the η value yielding the highest accuracy in the validation set in Figure 11(a) also produces the best imputation results in testing missing values in Figure 11(b), verifying the effectiveness of our parameter determination strategy.

Finally, Figure 12 shows the experiments where we disable the automatic convergence criterion and instead manually set the number of iterations e. We can find that when the number of iteration rounds gradually increases, the imputation process of imputation cost gradually converges, but the training time will also gradually become longer. Therefore, in practice, our automatic stop strategy, according to the converged imputation cost, is effective and efficient.

Notably, since our algorithms default to the 0-1 normalization of data, we do not involve the regularization parameter.

D. Application Study

Figure 13 reports the experiment results of serving real applications. Following the same line of existing studies [38], [39], we use RMSE to evaluate the air quality prediction application, i.e., the regression task, in Figure 13(a) and AUC is adopted for measuring the in-hospital mortality forecasting application, i.e., the classification problem, in Figure 13(b).

Following the same line of [38], [39], [61], we could impute 13.72% real missing data in AirQuality, to estimate the downstream air quality prediction accuracy, with the Lasso-normed linear regression implementation [51]. The in-hospital

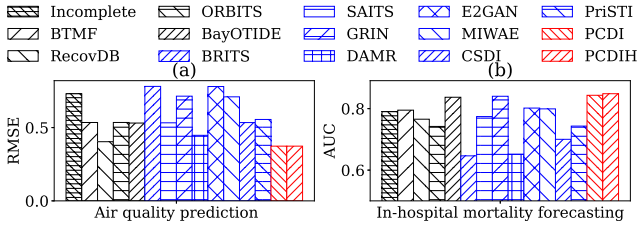


Fig. 13. Air quality prediction and in-hospital mortality forecasting accuracy without/with imputation over AirQuality and MIMIC-III datasets

mortality forecasting task is conducted over MIMIC-III, with 21.84% real missing values, using the random forest classifier [51]. As shown, our approaches achieve the highest application accuracy again, demonstrating the superiority of applying our work in improving real downstream applications.

VIII. Related Work

In this section, we discuss the representative imputation approaches using traditional signals and deep learning models.

Traditional signals, e.g., regression, statistics, matrix factorization (MF), centroid decomposition (CD), are utilized for imputing multivariate time series data [33]. Regression-based methods [73], [7] model data dependencies and suit situations with limited missing cells or complete determinant attributes. Statistical approaches [35], [3], [46], [67] use various statistical measures to address missing data issues. MF and CD techniques [37], [32], [5], [31] explore data correlations but may struggle with heterogeneous datasets. To address heterogeneous data and temporal dependencies, methods using temporal rules are also proposed [2]. Bayesian-based methods [26] can impute missing data by the uncertainty quantification.

DL models have been recently deployed for the imputation task, usually through end-to-end setups. Long-term dependencies are addressed using recurrent neural networks [16], [17] and Transformers [24], [6]. Graph neural networks [21], [54], [25] and attention mechanisms [69] focus on intratemporal dependencies. Generative models such as GAN [38], [39], [29], [44], [75], VAE [40], [27], [48], and Diffusion models [61], [36], [66], address data imputation issues, albeit with specific challenges like mode collapse and KL vanishing.

IX. Conclusion

In this work, we study how to conduct collaborative imputation for various missing values in multivariate time series with the convergence guarantee. To meet the challenges and limitations of existing studies, our major technical highlights include 1) formalizing the statistically explainable collaborative imputation problem, by constructing imputation contexts and dependency models, 2) designing both sequential and parallel imputation algorithms with the convergence guarantee (Propositions 1 and 3), 3) incorporating dynamic models into the collaborative imputation process with the certified convergence (Proposition 4), and further improving it into the parallel version with the convergence guarantee (Propositions 6 and 7), 4) designing streaming computation and parameter determination strategies. Extensive experiments on real-world incomplete datasets show the superiority of our methods.

REFERENCES

[1] https://github.com/CDI25/CDI25.

[2] Z. Abedjan, C. G. Akcora, M. Ouzzani, P. Papotti, and M. Stonebraker. Temporal rules discovery for web data cleaning. In *VLDB*, 2015.

[3] E. Acuna and C. Rodriguez. The treatment of missing values and its effect on classifier accuracy. In *Classification, clustering, and data mining applications*, pages 639–647. 2004.

[4] H. Adeli and P. R. Vishnubhotla. Parallel processing and parallel machines. In *Parallel processing in computational mechanics*, pages 1–20. CRC Press, 2020.

[5] I. Arous, M. Khayati, P. Cudré-Mauroux, Y. Zhang, M. L. Kersten, and S. Stalinlov. Recovdb: Accurate and efficient missing blocks recovery for large time series. In *ICDE*, 2019.

[6] P. Bansal, P. Deshpande, and S. Sarawagi. Missing value imputation on multidimensional time series. In *VLDB*, 2021.

[7] F. Bashir and H.-L. Wei. Handling missing data in multivariate time series using a vector autoregressive model-imputation (var-im) algorithm. *Neurocomputing*, 276:23–30, 2018.

[8] D. Bertsekas and J. Tsitsiklis. *Parallel and distributed computation: numerical methods*. 2015.

[9] C. M. Bishop. *Pattern recognition and machine learning*. 2006.

[10] N. Bleier, M. H. Mubarik, F. Rasheed, J. Aghassi-Hagmann, M. B. Tahoori, and R. Kumar. Printed microprocessors. In *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*, pages 213–226. IEEE, 2020.

[11] P. Bohannon, M. Flaster, W. Fan, and R. Rastogi. A cost-based model and effective heuristic for repairing constraints by value modification. In *SIGMOD*, pages 143–154, 2005.

[12] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung. *Statistical analysis with missing data*. John Wiley & Sons, 2015.

[13] S. Boyd and L. Vandenberghe. *Convex optimization*. 2004.

[14] S. Burns and P. Bond. *Principles of electronic circuits*. West Publishing Co., 1987.

[15] L. M. Candanedo, V. Feldheim, and D. Deramaix. Data driven prediction models of energy use of appliances in a low-energy house. *Energy and buildings*, 140:81–97, 2017.

[16] W. Cao, D. Wang, J. Li, H. Zhou, L. Li, and Y. Li. BRITS: bidirectional recurrent imputation for time series. In *NeurIPS*, 2018.

[17] Z. Che, S. Purushotham, K. Cho, D. Sontag, and Y. Liu. Recurrent neural networks for multivariate time series with missing values. *Scientific reports*, 8(1):1–12, 2018.

[18] X. Chen and L. Sun. Bayesian temporal factorization for multidimensional time series prediction. *TPAMI*, 44(9):4659–4673, 2022.

[19] B. Cho, T. Dayrit, Y. Gao, Z. Wang, T. Hong, A. Sim, and K. Wu. Effective missing value imputation methods for building monitoring data. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 2866–2875. IEEE, 2020.

[20] X. Chu, I. F. Ilyas, and P. Papotti. Holistic data cleaning: Putting violations into context. In *ICDE*, pages 458–469, 2013.

[21] A. Cini, I. Marisca, and C. Alippi. Filling the g_ap_s: Multivariate time series imputation by graph neural networks. In *ICLR*, 2022.

[22] W. S. Cleveland and C. Loader. Smoothing by local regression: Principles and methods. In *Statistical theory and computational aspects of smoothing*. 1996.

[23] S. De Vito, E. Massera, M. Piga, L. Martinotto, and G. Di Francia. On field calibration of an electronic nose for benzene estimation in an urban pollution monitoring scenario. *Sensors and Actuators B: Chemical*, 129(2):750–757, 2008.

[24] W. Du, D. Côté, and Y. Liu. SAITS: self-attention-based imputation for time series. *ESWA*, 219:119619, 2023.

[25] Y. Fan, X. Yu, R. Wieser, D. Meakin, A. Shaton, J. Jaubert, R. Flottemesch, M. Howell, J. Braid, L. S. Bruckman, R. H. French, and Y. Wu. Spatio-temporal denoising graph autoencoders with data augmentation for photovoltaic data imputation. In *SIGMOD*, 2023.

[26] S. Fang, Q. Wen, S. Zhe, and L. Sun. Bayotide: Bayesian online multivariate time series imputation with functional decomposition. In *ICML*, 2024.

[27] V. Fortuin, D. Baranchuk, G. Rätsch, and S. Mandt. GP-VAE: deep probabilistic time series imputation. In *AISTATS*, 2020.

[28] M. K. Hasan, M. A. Alam, S. Roy, A. Dutta, M. T. Jawad, and S. Das. Missing value imputation affects the performance of machine learning: A review and analysis of the literature (2010–2021). *IMU*, 27:100799, 2021.

[29] H. Jiang, C. Wan, K. Yang, Y. Ding, and S. Xue. Continuous missing data imputation with incomplete dataset by generative adversarial networks–based unsupervised learning for long-term bridge health monitoring. *Structural Health Monitoring*, 21(3):1093–1109, 2022.

[30] A. E. Johnson, T. J. Pollard, L. Shen, L.-w. H. Lehman, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. Anthony Celi, and R. G. Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9, 2016.

[31] M. Khayati, I. Arous, Z. Tymchenko, and P. Cudré-Mauroux. ORBITS: online recovery of missing values in multiple time series streams. In *VLDB*, 2020.

[32] M. Khayati, M. H. Böhlen, and J. Gamper. Memory-efficient centroid decomposition for long time series. In I. F. Cruz, E. Ferrari, Y. Tao, E. Bertino, and G. Trajcevski, editors, *ICDE*, 2014.

[33] M. Khayati, A. Lerner, Z. Tymchenko, and P. Cudré-Mauroux. Mind the gap: An experimental evaluation of imputation of missing values techniques in time series. In *VLDB*, 2020.

[34] J. Large, E. K. Kemsley, N. Wellner, I. Goodall, and A. J. Bagnall. Detecting forged alcohol non-invasively through vibrational spectroscopy and machine learning. In *PAKDD*, 2018.

[35] R. J. Little and D. B. Rubin. *Statistical analysis with missing data*, volume 793. John Wiley & Sons, 2019.

[36] M. Liu, H. Huang, H. Feng, L. Sun, B. Du, and Y. Fu. Pristi: A conditional diffusion framework for spatiotemporal imputation. In *ICDE*, 2023.

[37] X. Luo, M. Zhou, H. Leung, Y. Xia, Q. Zhu, Z. You, and S. Li. An incremental-and-static-combined scheme for matrix-factorization-based collaborative filtering. *TASE*, 13(1):333–343, 2016.

[38] Y. Luo, X. Cai, Y. Zhang, J. Xu, and X. Yuan. Multivariate time series imputation with generative adversarial networks. In *NeurIPS*, 2018.

[39] Y. Luo, Y. Zhang, X. Cai, and X. Yuan. E^2gan: End-to-end generative adversarial network for multivariate time series imputation. In *IJCAI*, 2019.

[40] P. Mattei and J. Frellsen. MIWAE: deep generative modelling and imputation of incomplete data sets. In *ICML*, 2019.

[41] C. Mayfield, J. Neville, and S. Prabhakar. ERACER: a database approach for statistical inference and data cleaning. In *SIGMOD*, 2010.

[42] J. Mei, Y. de Castro, Y. Goude, and G. Hébrail. Nonnegative matrix factorization for time series recovery from a few temporal aggregates. In *ICML*, volume 70, pages 2382–2390, 2017.

[43] X. Miao, Y. Wu, L. Chen, Y. Gao, J. Wang, and J. Yin. Efficient and effective data imputation with influence functions. In *VLDB*, 2021.

[44] X. Miao, Y. Wu, J. Wang, Y. Gao, X. Mao, and J. Yin. Generative semi-supervised learning for multivariate time series imputation. In *AAAI*, 2021.

[45] M. Mohri, A. Rostamizadeh, and A. Talwalkar. *Foundations of Machine Learning*. Adaptive computation and machine learning. MIT Press, 2012.

[46] S. Moritz and T. Bartz-Beielstein. imputets: Time series missing value imputation in R. *R J.*, 9(1):207, 2017.

[47] S. Nakagawa. Missing data: mechanisms, methods and messages. *Ecological statistics: Contemporary theory and application*, pages 81–105, 2015.

[48] A. Nazábal, P. M. Olmos, Z. Ghahramani, and I. Valera. Handling incomplete heterogeneous data using vaes. *PR*, 107:107501, 2020.

[49] D. Nemec, J. Andel, V. Simák, and J. Hrbcek. Homogeneous sensor fusion optimization for low-cost inertial sensors. *Sensors*, 23(14):6431, 2023.

[50] P. Pedregal. *Introduction to optimization*, volume 46. Springer Science & Business Media, 2006.

[51] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in python. *JMLR*, 12:2825–2830, 2011.

[52] T. D. Pigott. Handling missing data. *The handbook of research synthesis and meta-analysis*, 2:399–416, 2009.

[53] B. Recht, C. Ré, S. J. Wright, and F. Niu. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In *NeurIPS*, pages 693–701, 2011.

[54] X. Ren, K. Zhao, P. J. Riddle, K. Taskova, Q. Pan, and L. Li. DAMR: dynamic adjacency matrix representation learning for multivariate time series imputation. In *SIGMOD*, 2023.

[55] S. Song and Y. Sun. Imputing various incomplete attributes via distance likelihood maximization. In *KDD*, 2020.

[56] S. Song and A. Zhang. Iot data quality. In *CIKM*, 2020.

[57] S. Song, A. Zhang, J. Wang, and P. S. Yu. SCREEN: stream data cleaning under speed constraints. In *SIGMOD*, pages 827–841, 2015.

[58] M. Soori, B. Arezoo, and R. Dastres. Internet of things for smart factories in industry 4.0, a review. *Internet of Things and Cyber-Physical Systems*, 3:192–204, 2023.

[59] R. E. Tarjan. Efficiency of a good but not linear set union algorithm. *J. ACM*, 22(2):215–225, 1975.

[60] R. E. Tarjan. A class of algorithms which require nonlinear time to maintain disjoint sets. *JCSS*, 18(2):110–127, 1979.

[61] Y. Tashiro, J. Song, Y. Song, and S. Ermon. CSDI: conditional score-based diffusion models for probabilistic time series imputation. In *NeurIPS*, 2021.

[62] K. Trebing and S. Mehrkanoon. Wind speed prediction using multidimensional convolutional neural networks. In *SSCI*, 2020.

[63] R. S. Tsay. *Multivariate time series analysis: with R and financial applications*. 2013.

[64] B. Twala. An empirical comparison of techniques for handling incomplete data using decision trees. *AAI*, 23(5):373–405, 2009.

[65] V. V. Vazirani. Minimum makespan scheduling. In *Approximation Algorithms*, pages 79–83. 2003.

[66] X. Wang, H. Zhang, P. Wang, Y. Zhang, B. Wang, Z. Zhou, and Y. Wang. An observed value consistent diffusion model for imputing missing values in multivariate time series. In *KDD*, 2023.

[67] K. Wellenzohn, M. H. Böhlen, A. Dignös, J. Gamper, and H. Mitterer. Continuous imputation of missing values in streams of pattern-determining time series. In *EDBT*, 2017.

[68] J. Xia, S. Zhang, G. Cai, L. Li, Q. Pan, J. Yan, and G. Ning. Adjusted weight voting algorithm for random forests in handling missing values. *PR*, 69:52–60, 2017.

[69] Q. Xu, S. Ruan, C. Long, L. Yu, and C. Zhang. Traffic speed imputation with spatio-temporal attentions and cycle-perceptual training. In *CIKM*, 2022.

[70] M. Yakout, L. Berti-Équille, and A. K. Elmagarmid. Don't be scared: use scalable automatic repairing with maximal likelihood and bounded changes. In *SIGMOD*, 2013.

[71] H. Yu, N. Rao, and I. S. Dhillon. Temporal regularized matrix factorization for high-dimensional time series prediction. In *NeurIPS*, pages 847–855, 2016.

[72] A. Zhang, S. Song, Y. Sun, and J. Wang. Learning individual models for imputation. In *ICDE*, 2019.

[73] G. P. Zhang. Time series forecasting using a hybrid ARIMA and neural network model. *Neurocomputing*, 50:159–175, 2003.

[74] J. Zhang and P. Yin. Multivariate time series missing data imputation using recurrent denoising autoencoder. In *BIBM 2019*, 2019.

[75] J. Zhao, C. Rong, C. Lin, and X. Dang. Multivariate time series data imputation using attention-based mechanism. *Neurocomputing*, 2023.

[76] S. Zhao, G. Zhang, and W. Li. Lock-free optimization for non-convex problems. In *AAAI*, 2017.

APPENDIX

A. Proof of Proposition 1

According to the updating formula of the null cell $x_{ij} \in \mathcal{C} \cap M$ in Line 4 in Algorithm 1, we have

$$\begin{aligned}
&\Theta\left(\mathcal{C}'^{(k+1)}, \mathcal{G}\right)\\
=&\Theta\left(\mathcal{C}'^{(k+1)}, \mathcal{G} \mid x_{ij}'^{(k+1)} = x_{ij}'^{(k)} - \eta\frac{\partial\Theta(\mathcal{C}'^{(k)}, \mathcal{G})}{\partial x_{ij}'^{(k)}},\ \forall x_{ij} \in M\right).
\end{aligned}$$

Given $\eta \leq \epsilon / \left\|\frac{\partial\Theta(\mathcal{C}'^{(k)}, \mathcal{G})}{\partial x_{ij}'^{(k)}}\right\|$, i.e., $\left\|-\eta\frac{\partial\Theta(\mathcal{C}'^{(k)}, \mathcal{G})}{\partial x_{ij}'^{(k)}}\right\| \leq \epsilon$, referring to the first-order Taylor expansion [45], it follows

$$\begin{aligned}
&\Theta\left(\mathcal{C}'^{(k+1)}, \mathcal{G} \mid x_{ij}'^{(k+1)} = x_{ij}'^{(k)} - \eta\frac{\partial\Theta(\mathcal{C}'^{(k)}, \mathcal{G})}{\partial x_{ij}'^{(k)}},\ \forall x_{ij} \in M\right)\\
=&\Theta\left(\mathcal{C}'^{(k+1)}, \mathcal{G} \mid x_{ij}'^{(k+1)} = x_{ij}'^{(k)}, \forall x_{ij} \in M\right)\\
&+ \sum_{x_{ij} \in M}\left\|-\eta\frac{\partial\Theta(\mathcal{C}'^{(k)}, \mathcal{G})}{\partial x_{ij}'^{(k)}}\right\| \cdot \frac{\partial\Theta(\mathcal{C}'^{(k)}, \mathcal{G})}{\partial x_{ij}'^{(k)}}\\
=&\Theta\left(\mathcal{C}'^{(k+1)}, \mathcal{G} \mid x_{ij}'^{(k+1)} = x_{ij}'^{(k)}, \forall x_{ij} \in M\right)\\
&- \eta \sum_{x_{ij} \in M}\left\|\frac{\partial\Theta(\mathcal{C}'^{(k)}, \mathcal{G})}{\partial x_{ij}'^{(k)}}\right\|^2\\
\leq&\Theta\left(\mathcal{C}'^{(k+1)}, \mathcal{G} \mid x_{ij}'^{(k+1)} = x_{ij}'^{(k)}, \forall x_{ij} \in M\right)\\
=&\Theta\left(\mathcal{C}'^{(k)}, \mathcal{G}\right).
\end{aligned}$$

B. Proof of Proposition 2

We first consider the left term $\Theta(\mathbf{C}_1', \mathcal{G} \mid \mathbf{C}_2')$ of Formula 4, which leads to

$$\begin{aligned}
&\Theta(\mathbf{C}_1', \mathcal{G} \mid \mathbf{C}_2')\\
=&\sum_{g_p \in \mathcal{G}} \sum_{C_{ip}' \in \mathbf{C}_1'} \|x_{ip}' - g_p(C_{ip}' \setminus \{x_{ip}'\})\|^2\\
=&\Theta(\mathbf{C}_1', \mathcal{G}).
\end{aligned}$$

According to Definition 6, we know $\mathbf{C}_1 \cap \mathbf{C}_2 \cap M = \emptyset$, which means that the missing cells in $\mathbf{C}_2$ do not appear in $\mathbf{C}_1$. Therefore, we also have

$$\begin{aligned}
&\Theta(\mathbf{C}_1', \mathcal{G} \mid \mathbf{C}_2'')\\
=&\sum_{g_p \in \mathcal{G}} \sum_{C_{ip}' \in \mathbf{C}_1'} \|x_{ip}' - g_p(C_{ip}' \setminus \{x_{ip}'\})\|^2\\
=&\Theta(\mathbf{C}_1', \mathcal{G}).
\end{aligned}$$

It completes the proof.

C. Proof of Proposition 3

Consider the initialization of $\mathcal{C}_m = \{\mathbf{C}_1, \ldots, \mathbf{C}_u\}$ in Line 2 in Algorithm 2, which follows

$$\mathcal{C}' = \mathbf{C}_1' \cup \mathbf{C}_2' \cup \cdots \cup \mathbf{C}_u' \cup \{\mathcal{C}' \setminus \mathcal{C}_m'\}.$$

According to Definition 3 for the imputation cost, we can obtain

$$\begin{aligned}
\Theta\left(\mathcal{C}', \mathcal{G}\right) =&\Theta\left(\mathbf{C}_1', \mathcal{G}\right) + \Theta\left(\mathbf{C}_2', \mathcal{G}\right) + \ldots\\
&+ \Theta\left(\mathbf{C}_u', \mathcal{G}\right) + \Theta\left(\mathcal{C}' \setminus \mathcal{C}_m', \mathcal{G}\right).
\end{aligned}$$

Combing with Proposition 2, for any $\mathbf{C}_i, \mathbf{C}_j \in \mathcal{C}_m$, they always have

$$\Theta\left(\mathbf{C}_i', \mathcal{G} \mid \mathbf{C}_j'\right) = \Theta\left(\mathbf{C}_i', \mathcal{G} \mid \mathbf{C}_j''\right) = \Theta\left(\mathbf{C}_i', \mathcal{G}\right),$$

where $\mathbf{C}_j'$ and $\mathbf{C}_j''$ are two different fillings of $\mathbf{C}_j$.

Moreover, for any $\mathbf{C}_i, \mathbf{C}_j \in \mathcal{C}_m$ at the k-th round update in Algorithm 2, they always hold

$$\begin{aligned}
&\Theta\left(\mathbf{C}_i'^{(k+1)}, \mathcal{G} \mid \mathbf{C}_j'^{(k+1)}\right)\\
=&\Theta\left(\mathbf{C}_i'^{(k+1)}, \mathcal{G} \mid \mathbf{C}_j'^{(k)}\right)\\
=&\Theta\left(\mathbf{C}_i'^{(k+1)}, \mathcal{G}\right)\\
=&\Theta\left(\mathbf{C}_i'^{(k+1)}, \mathcal{G} \mid x_{lq}'^{(k+1)} = x_{lq}'^{(k)} - \eta\frac{\partial\Theta(\mathbf{C}_i'^{(k)}, \mathcal{G})}{\partial x_{lq}'^{(k)}},\right.\\
&\left.\forall x_{lq} \in \mathbf{C}_i \cap M\right).
\end{aligned}$$

Therefore, for the k-th round update in Algorithm 2, referring to Line 4 in Algorithm 1, it follows

$$\begin{aligned}
&\Theta(\mathcal{C}'^{(k+1)}, \mathcal{G})\\
=&\Theta\left(\mathcal{C}'^{(k+1)}, \mathcal{G} \mid x_{ij}'^{(k+1)} = x_{ij}'^{(k)} - \eta\frac{\partial\Theta(\mathcal{C}'^{(k)}, \mathcal{G})}{\partial x_{ij}'^{(k)}},\right.\\
&\left.\forall x_{ij} \in M\right)\\
=&\Theta\left(\mathbf{C}_1'^{(k+1)}, \mathcal{G} \mid x_{ij}'^{(k+1)} = x_{ij}'^{(k)} - \eta\frac{\partial\Theta(\mathbf{C}_1'^{(k)}, \mathcal{G})}{\partial x_{ij}'^{(k)}},\right.\\
&\left.\forall x_{ij} \in \mathbf{C}_1 \cap M\right)\\
&+ \ldots\\
&+ \Theta\left(\mathbf{C}_u'^{(k+1)}, \mathcal{G} \mid x_{ij}'^{(k+1)} = x_{ij}'^{(k)} - \eta\frac{\partial\Theta(\mathbf{C}_u'^{(k)}, \mathcal{G})}{\partial x_{ij}'^{(k)}},\right.\\
&\left.\forall x_{ij} \in \mathbf{C}_u \cap M\right)\\
&+ \Theta\left(\mathcal{C}' \setminus \mathcal{C}_m', \mathcal{G}\right).
\end{aligned}$$

That is, PCDI Algorithm 2 returns the same result $\mathcal{C}'$ with CDI Algorithm 1 for fixed updates.

D. Proof of Proposition 4

Given

$$\eta \leq min\left\{\epsilon / \left\|\frac{\partial\Theta(\mathcal{C}'^{(k)}, \mathcal{G}'^{(k)})}{\partial x_{ij}'^{(k)}}\right\|, \epsilon / \left\|\frac{\partial\Theta(\mathcal{C}'^{(k)}, \mathcal{G}'^{(k)})}{\partial \Phi_{\mathcal{G}'}^{(k)}}\right\|\right\},$$

which indicates that

$$\left\|-\eta\frac{\partial\Theta(\mathcal{C}'^{(k)}, \mathcal{G}'^{(k)})}{\partial x_{ij}'^{(k)}}\right\| \leq \epsilon$$

and

$$\left\|-\eta\frac{\partial\Theta(\mathcal{C}'^{(k)}, \mathcal{G}'^{(k)})}{\partial \Phi_{\mathcal{G}'}^{(k)}}\right\| \leq \epsilon.$$

According to the proof of Proposition 1, for $\left\|-\eta\frac{\partial\Theta(\mathcal{C}'^{(k)},\mathcal{G}'^{(k)})}{\partial x_{ij}'^{(k)}}\right\| \leq \epsilon$, it leads to

$$\Theta\left(\mathcal{C}'^{(k+1)},\mathcal{G}'^{(k)}\right) \leq \Theta\left(\mathcal{C}'^{(k)},\mathcal{G}'^{(k)}\right).$$

Following Line 6 in Algorithm 3, we have

$$\begin{aligned}
&\Theta\left(\mathcal{C}'^{(k+1)},\mathcal{G}'^{(k+1)}\right)\\
=&\Theta\left(\mathcal{C}'^{(k+1)},\mathcal{G}'^{(k+1)} \mid \Phi_{\mathcal{G}'}^{(k+1)} = \Phi_{\mathcal{G}'}^{(k)} - \eta\frac{\partial\Theta(\mathcal{C}'^{(k)},\mathcal{G}'^{(k)})}{\partial\Phi_{\mathcal{G}'}^{(k)}}\right).
\end{aligned}$$

Moreover, according to the first-order Taylor expansion, given $\left\|-\eta\frac{\partial\Theta(\mathcal{C}'^{(k)},\mathcal{G}'^{(k)})}{\partial\Phi_{\mathcal{G}'}^{(k)}}\right\| \leq \epsilon$, it further has

$$\begin{aligned}
&\Theta\left(\mathcal{C}'^{(k+1)},\mathcal{G}'^{(k+1)} \mid \Phi_{\mathcal{G}'}^{(k+1)} = \Phi_{\mathcal{G}'}^{(k)} - \eta\frac{\partial\Theta(\mathcal{C}'^{(k)},\mathcal{G}'^{(k)})}{\partial\Phi_{\mathcal{G}'}^{(k)}}\right)\\
=&\Theta\left(\mathcal{C}'^{(k+1)},\mathcal{G}'^{(k+1)} \mid \Phi_{\mathcal{G}'}^{(k+1)} = \Phi_{\mathcal{G}'}^{(k)}\right)\\
&+\left\|-\eta\frac{\partial\Theta(\mathcal{C}'^{(k)},\mathcal{G}'^{(k)})}{\partial\Phi_{\mathcal{G}'}^{(k)}}\right\|\cdot\frac{\partial\Theta(\mathcal{C}'^{(k)},\mathcal{G}'^{(k)})}{\partial\Phi_{\mathcal{G}'}^{(k)}}\\
=&\Theta\left(\mathcal{C}'^{(k+1)},\mathcal{G}'^{(k+1)} \mid \Phi_{\mathcal{G}'}^{(k+1)} = \Phi_{\mathcal{G}'}^{(k)}\right)\\
&-\eta\left\|\frac{\partial\Theta(\mathcal{C}'^{(k)},\mathcal{G}'^{(k)})}{\partial\Phi_{\mathcal{G}'}^{(k)}}\right\|^2\\
\leq&\Theta\left(\mathcal{C}'^{(k+1)},\mathcal{G}'^{(k+1)} \mid \Phi_{\mathcal{G}'}^{(k+1)} = \Phi_{\mathcal{G}'}^{(k)}\right)\\
=&\Theta\left(\mathcal{C}'^{(k+1)},\mathcal{G}'^{(k)}\right).
\end{aligned}$$

E. Proof of Lemma 5

According to Formula 5, we know

$$\begin{aligned}
&\mathbb{E}\|\Phi_{\mathcal{G}'}^{(\kappa)} - \Phi_{\mathcal{G}'}^{(\kappa-\tau_\kappa)}\|^2\\
=&\mathbb{E}_{P_j\sim\mathbf{P}}\left\|\Phi_{\mathcal{G}'}^{(\kappa-\tau_\kappa)} - \eta\sum_{t=0}^{\tau_\kappa-1}\frac{\partial\Theta(\mathcal{C}'_j,\mathcal{G}'^{(\kappa-\tau_\kappa+t-\tau_{\kappa-\tau_\kappa+t})})}{\partial\Phi_{\mathcal{G}'}^{(\kappa-\tau_\kappa+t-\tau_{\kappa-\tau_\kappa+t})}}\right.\\
&\left.-\Phi_{\mathcal{G}'}^{(\kappa-\tau_\kappa)}\right\|^2.
\end{aligned}$$

Combining with Assumptions 1.2 and 1.3, it further leads to

$$\begin{aligned}
&\mathbb{E}\|\Phi_{\mathcal{G}'}^{(\kappa)} - \Phi_{\mathcal{G}'}^{(\kappa-\tau_\kappa)}\|^2\\
=&\eta^2\mathbb{E}_{P_j\sim\mathbf{P}}\left\|\sum_{t=0}^{\tau_\kappa-1}\frac{\partial\Theta(\mathcal{C}'_j,\mathcal{G}'^{(\kappa-\tau_\kappa+t-\tau_{\kappa-\tau_\kappa+t})})}{\partial\Phi_{\mathcal{G}'}^{(\kappa-\tau_\kappa+t-\tau_{\kappa-\tau_\kappa+t})}}\right\|^2\\
\overset{Assumption\ 1.3}{\leqslant}&\eta^2\left\|\tau_\kappa V\right\|^2\\
=&\tau_k^2\eta^2V^2\\
\overset{Assumption\ 1.2}{\leqslant}&T^2\eta^2V^2.
\end{aligned}$$

F. Proof of Proposition 6

We start from Formula 5, combining with Line 9 in Algorithm 4 and Proposition 1, it has

$$\begin{aligned}
&\mathbb{E}_{P_j\sim\mathbf{P}}\Theta\left(\mathcal{C}'_j,\mathcal{G}'^{(\kappa+1)}\right)\\
=&\mathbb{E}_{P_j\sim\mathbf{P}}\Theta\left(\mathcal{C}'_j,\mathcal{G}'^{(\kappa+1)} \mid \Phi_{\mathcal{G}'}^{(\kappa+1)} = \Phi_{\mathcal{G}'}^{(\kappa)} - \eta\frac{\partial\Theta(\mathcal{C}'_j,\mathcal{G}'^{(\kappa-\tau_\kappa)})}{\partial\Phi_{\mathcal{G}'}^{(\kappa-\tau_\kappa)}}\right).
\end{aligned}$$

Then, according to Assumption 1.1, we have

$$\begin{aligned}
&\mathbb{E}_{P_j\sim\mathbf{P}}\Theta\left(\mathcal{C}'_j,\mathcal{G}'^{(\kappa+1)}\right)\\
\overset{Assumption\ 1.1}{\leqslant}&\mathbb{E}_{P_j\sim\mathbf{P}}\Theta\left(\mathcal{C}'_j,\mathcal{G}'^{(\kappa)}\right)\\
&-\eta\mathbb{E}_{P_j\sim\mathbf{P}}\left\langle\frac{\partial\Theta(\mathcal{C}'_j,\mathcal{G}'^{(\kappa)})}{\partial\Phi_{\mathcal{G}'}^{(\kappa)}},\frac{\partial\Theta(\mathcal{C}'_j,\mathcal{G}'^{(\kappa-\tau_\kappa)})}{\partial\Phi_{\mathcal{G}'}^{(\kappa-\tau_\kappa)}}\right\rangle\\
&+\frac{L\eta^2}{2}\mathbb{E}_{P_j\sim\mathbf{P}}\left\|\frac{\partial\Theta(\mathcal{C}'_j,\mathcal{G}'^{(\kappa-\tau_\kappa)})}{\partial\Phi_{\mathcal{G}'}^{(\kappa-\tau_\kappa)}}\right\|^2\\
=&\mathbb{E}_{P_j\sim\mathbf{P}}\Theta\left(\mathcal{C}'_j,\mathcal{G}'^{(\kappa)}\right)\\
&+\frac{\eta}{2}\mathbb{E}_{P_j\sim\mathbf{P}}\left\|\frac{\partial\Theta(\mathcal{C}'_j,\mathcal{G}'^{(\kappa)})}{\partial\Phi_{\mathcal{G}'}^{(\kappa)}}-\frac{\partial\Theta(\mathcal{C}'_j,\mathcal{G}'^{(\kappa-\tau_\kappa)})}{\partial\Phi_{\mathcal{G}'}^{(\kappa-\tau_\kappa)}}\right\|^2\\
&-\frac{\eta}{2}\mathbb{E}_{P_j\sim\mathbf{P}}\left\|\frac{\partial\Theta(\mathcal{C}'_j,\mathcal{G}'^{(\kappa)})}{\partial\Phi_{\mathcal{G}'}^{(\kappa)}}\right\|^2\\
&-\frac{\eta}{2}\mathbb{E}_{P_j\sim\mathbf{P}}\left\|\frac{\partial\Theta(\mathcal{C}'_j,\mathcal{G}'^{(\kappa-\tau_\kappa)})}{\partial\Phi_{\mathcal{G}'}^{(\kappa-\tau_\kappa)}}\right\|^2\\
&+\frac{L\eta^2}{2}\mathbb{E}_{P_j\sim\mathbf{P}}\left\|\frac{\partial\Theta(\mathcal{C}'_j,\mathcal{G}'^{(\kappa-\tau_\kappa)})}{\partial\Phi_{\mathcal{G}'}^{(\kappa-\tau_\kappa)}}\right\|^2.
\end{aligned}$$

Combining with Assumption 1.3, we can obtain

$$\begin{aligned}
&\mathbb{E}_{P_j\sim\mathbf{P}}\Theta\left(\mathcal{C}'_j,\mathcal{G}'^{(\kappa+1)}\right)\\
\leq&\mathbb{E}_{P_j\sim\mathbf{P}}\Theta\left(\mathcal{C}'_j,\mathcal{G}'^{(\kappa)}\right)\\
&+\frac{\eta}{2}\mathbb{E}_{P_j\sim\mathbf{P}}\left\|\frac{\partial\Theta(\mathcal{C}'_j,\mathcal{G}'^{(\kappa)})}{\partial\Phi_{\mathcal{G}'}^{(\kappa)}}-\frac{\partial\Theta(\mathcal{C}'_j,\mathcal{G}'^{(\kappa-\tau_\kappa)})}{\partial\Phi_{\mathcal{G}'}^{(\kappa-\tau_\kappa)}}\right\|^2\\
&-\frac{\eta}{2}\mathbb{E}_{P_j\sim\mathbf{P}}\left\|\frac{\partial\Theta(\mathcal{C}'_j,\mathcal{G}'^{(\kappa)})}{\partial\Phi_{\mathcal{G}'}^{(\kappa)}}\right\|^2\\
&+\frac{L\eta^2}{2}\mathbb{E}_{P_j\sim\mathbf{P}}\left\|\frac{\partial\Theta(\mathcal{C}'_j,\mathcal{G}'^{(\kappa-\tau_\kappa)})}{\partial\Phi_{\mathcal{G}'}^{(\kappa-\tau_\kappa)}}\right\|^2\\
\overset{Assumption\ 1.3}{\leqslant}&\mathbb{E}_{P_j\sim\mathbf{P}}\Theta\left(\mathcal{C}'_j,\mathcal{G}'^{(\kappa)}\right)\\
&+\frac{\eta}{2}\left\|\frac{\partial\Theta(\mathcal{C}'_j,\mathcal{G}'^{(\kappa)})}{\partial\Phi_{\mathcal{G}'}^{(\kappa)}}-\frac{\partial\Theta(\mathcal{C}'_j,\mathcal{G}'^{(\kappa-\tau_\kappa)})}{\partial\Phi_{\mathcal{G}'}^{(\kappa-\tau_\kappa)}}\right\|^2\\
&-\frac{\eta}{2}\mathbb{E}_{P_j\sim\mathbf{P}}\left\|\frac{\partial\Theta(\mathcal{C}'_j,\mathcal{G}'^{(\kappa)})}{\partial\Phi_{\mathcal{G}'}^{(\kappa)}}\right\|^2+\frac{L\eta^2V^2}{2}.
\end{aligned}$$

Moreover, according to Assumption 1.1, it has

$$\begin{aligned}
&\mathbb{E}_{P_j\sim\mathbf{P}}\Theta\left(\mathcal{C}'_j,\mathcal{G}'^{(\kappa+1)}\right)\\
\overset{Assumption\ 1.1}{\leqslant}\ &\mathbb{E}_{P_j\sim\mathbf{P}}\Theta\left(\mathcal{C}'_j,\mathcal{G}'^{(\kappa)}\right)\\
&+\frac{\eta L^2}{2}\mathbb{E}\|\Phi^{(\kappa)}_{\mathcal{G}'}-\Phi^{(\kappa-\tau_\kappa)}_{\mathcal{G}'}\|^2\\
&-\frac{\eta}{2}\mathbb{E}_{P_j\sim\mathbf{P}}\left\|\frac{\partial\Theta(\mathcal{C}'_j,\mathcal{G}'^{(\kappa)})}{\partial\Phi^{(\kappa)}_{\mathcal{G}'}}\right\|^2+\frac{L\eta^2V^2}{2}.
\end{aligned}$$

Referring to Lemma 5, it follows

$$\begin{aligned}
&\mathbb{E}_{P_j\sim\mathbf{P}}\Theta\left(\mathcal{C}'_j,\mathcal{G}'^{(\kappa+1)}\right)\\
\overset{Lemma\ 5}{\leqslant}\ &\mathbb{E}_{P_j\sim\mathbf{P}}\Theta\left(\mathcal{C}'_j,\mathcal{G}'^{(\kappa)}\right)\\
&-\frac{\eta}{2}\mathbb{E}_{P_j\sim\mathbf{P}}\left\|\frac{\partial\Theta(\mathcal{C}'_j,\mathcal{G}'^{(\kappa)})}{\partial\Phi^{(\kappa)}_{\mathcal{G}'}}\right\|^2\\
&+\frac{\eta L^2}{2}T^2\eta^2V^2+\frac{L\eta^2V^2}{2}.
\end{aligned}$$

For all processors, according to Definition 3 for the imputation cost, they have

$$\begin{aligned}
&\sum_{j=1}^{o}\mathbb{E}_{P_j\sim\mathbf{P}}\Theta\left(\mathcal{C}'_j,\mathcal{G}'^{(\kappa+1)}\right)\\
=&\mathbb{E}\Theta\left(\mathcal{C}',\mathcal{G}'^{(\kappa+1)}\right)\\
\leqslant&\sum_{j=1}^{o}\mathbb{E}_{P_j\sim\mathbf{P}}\Theta\left(\mathcal{C}'_j,\mathcal{G}'^{(\kappa)}\right)\\
&-\frac{\eta}{2}\sum_{j=1}^{o}\mathbb{E}_{P_j\sim\mathbf{P}}\left\|\frac{\partial\Theta(\mathcal{C}'_j,\mathcal{G}'^{(\kappa)})}{\partial\Phi^{(\kappa)}_{\mathcal{G}'}}\right\|^2\\
&+\frac{\eta L^2}{2}T^2\eta^2V^2o+\frac{L\eta^2V^2}{2}o\\
=&\mathbb{E}\Theta\left(\mathcal{C}',\mathcal{G}'^{(\kappa)}\right)\\
&-\frac{\eta}{2}\mathbb{E}\left\|\frac{\partial\Theta(\mathcal{C}',\mathcal{G}'^{(\kappa)})}{\partial\Phi^{(\kappa)}_{\mathcal{G}'}}\right\|^2\\
&+\frac{\eta L^2}{2}T^2\eta^2V^2o+\frac{L\eta^2V^2}{2}o.
\end{aligned}$$

Summing from $\kappa=0$ to $\kappa=K-1$, we can obtain

$$\begin{aligned}
&\mathbb{E}\Theta\left(\mathcal{C}',\mathcal{G}'^{(K)}\right)\\
\leqslant&\mathbb{E}\Theta\left(\mathcal{C}',\mathcal{G}'^{(0)}\right)-\frac{\eta}{2}\sum_{\kappa=0}^{K-1}\mathbb{E}\left\|\frac{\partial\Theta(\mathcal{C}',\mathcal{G}'^{(\kappa)})}{\partial\Phi^{(\kappa)}_{\mathcal{G}'}}\right\|^2\\
&+\frac{\eta^3L^2T^2V^2Ko}{2}+\frac{L\eta^2V^2Ko}{2}.
\end{aligned}$$

Considering the Definition 3 for the imputation cost, it leads to

$$\mathbb{E}\Theta\left(\mathcal{C}',\mathcal{G}'^{(K)}\right)\geqslant 0.$$

Finally, we can obtain the conclusion

$$\begin{aligned}
\sum_{\kappa=0}^{K-1}\mathbb{E}\left\|\frac{\partial\Theta(\mathcal{C}',\mathcal{G}'^{(\kappa)})}{\partial\Phi^{(\kappa)}_{\mathcal{G}'}}\right\|^2\leqslant&\frac{2\Theta\left(\mathcal{C}',\mathcal{G}'^{(0)}\right)}{\eta}\\
&+\eta oLKV^2(\eta LT^2+1).
\end{aligned}$$

G. Proof of Proposition 7

We start from Line 9 in Algorithm 4, it has

$$\begin{aligned}
&\mathbb{E}_{P_j\sim\mathbf{P}}\Theta\left(\mathcal{C}'^{(k_j+1)}_j,\mathcal{G}'\right)\\
=&\mathbb{E}_{P_j\sim\mathbf{P}}\Theta\left(\mathcal{C}'^{(k_j+1)}_j,\mathcal{G}'\mid\right.\\
&\left.x'^{(k_j+1)}_{ij}=x'^{(k_j)}_{ij}-\eta\frac{\partial\Theta(\mathcal{C}'^{(k_j)}_j,\mathcal{G}')}{\partial x'^{(k_j)}_{ij}},\forall x_{ij}\in\mathcal{C}_j\right).
\end{aligned}$$

Given $\eta\leq\epsilon/\left\|\frac{\partial\Theta(\mathcal{C}'^{(k_j)}_j,\mathcal{G}')}{\partial x'^{(k_j)}_{ij}}\right\|$, referring to the first-order Taylor expansion [45], it follows

$$\begin{aligned}
&\mathbb{E}_{P_j\sim\mathbf{P}}\Theta\left(\mathcal{C}'^{(k_j+1)}_j,\mathcal{G}'\right)\\
&=\mathbb{E}_{P_j\sim\mathbf{P}}\Theta\left(\mathcal{C}'^{(k_j)}_j,\mathcal{G}'\right)-\eta\sum_{x_{ij}\in\mathcal{C}_j}\left\|\frac{\partial\Theta(\mathcal{C}'^{(k_j)}_j,\mathcal{G}')}{\partial x'^{(k_j)}_{ij}}\right\|^2.
\end{aligned}$$

If the processor P_j updates the parameters κ_j times in total, summing from $k_j=0$ to $k_j=\kappa_j-1$, we can obtain

$$\begin{aligned}
&\mathbb{E}_{P_j\sim\mathbf{P}}\Theta\left(\mathcal{C}'^{(\kappa_j)}_j,\mathcal{G}'\right)\\
&=\mathbb{E}_{P_j\sim\mathbf{P}}\Theta\left(\mathcal{C}'^{(0)}_j,\mathcal{G}'\right)-\eta\sum_{k_j=0}^{\kappa_j-1}\sum_{x_{ij}\in\mathcal{C}_j}\left\|\frac{\partial\Theta(\mathcal{C}'^{(k_j)}_j,\mathcal{G}')}{\partial x'^{(k_j)}_{ij}}\right\|^2.
\end{aligned}$$

Considering Definition 3 for the imputation cost, we know

$$\mathbb{E}_{P_j\sim\mathbf{P}}\Theta\left(\mathcal{C}'^{(\kappa_j)}_j,\mathcal{G}'\right)\geqslant 0.$$

It leads to

$$\eta\sum_{k_j=0}^{\kappa_j-1}\sum_{x_{ij}\in\mathcal{C}_j}\left\|\frac{\partial\Theta(\mathcal{C}'^{(k_j)}_j,\mathcal{G}')}{\partial x'^{(k_j)}_{ij}}\right\|^2\leq\mathbb{E}_{P_j\sim\mathbf{P}}\Theta\left(\mathcal{C}'^{(0)}_j,\mathcal{G}'\right).$$

For all processors, we can obtain the following conclusion according to Definition 3:

$$\begin{aligned}
&\sum_{j=1}^{o}\sum_{k_j=0}^{\kappa_j-1}\sum_{x_{ij}\in\mathcal{C}_j}\left\|\frac{\partial\Theta(\mathcal{C}'^{(k_j)}_j,\mathcal{G}')}{\partial x'^{(k_j)}_{ij}}\right\|^2\\
\leq&\frac{1}{\eta}\sum_{j=1}^{o}\mathbb{E}_{P_j\sim\mathbf{P}}\Theta\left(\mathcal{C}'^{(0)}_j,\mathcal{G}'\right)\\
=&\frac{\Theta\left(\mathcal{C}'^{(0)},\mathcal{G}'\right)}{\eta}.
\end{aligned}$$

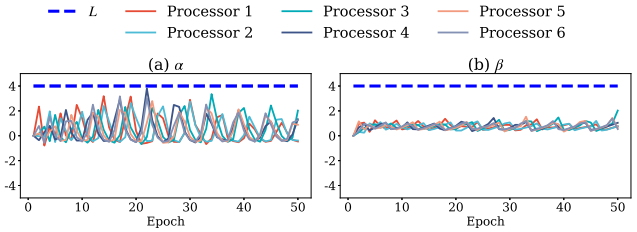


Fig. 14. L-smooth items over Energy dataset with 10% missing values

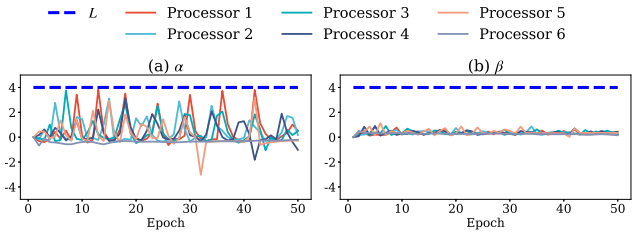


Fig. 15. L-smooth items over AirQuality dataset with 20% missing values

H. Empirical Evidence of L-smooth Assumption

The intuition of Assumption 1.1 is that the gradient of the imputation cost will not be too steep. In this section, we explore the empirical evidence for this assumption over multiple datasets.

Let us consider the two items in Assumption 1.1, the first one is

$$\Theta(\mathcal{C}'_j, \mathcal{G}'^{(b)}) \leqslant \Theta(\mathcal{C}'_j, \mathcal{G}'^{(a)}) + \frac{\partial \Theta(\mathcal{C}'_j, \mathcal{G}'^{(a)})}{\partial \Phi^{(a)}_{\mathcal{G}'}} (\Phi^{(b)}_{\mathcal{G}'} - \Phi^{(a)}_{\mathcal{G}'}) + \frac{L}{2} \|\Phi^{(b)}_{\mathcal{G}'} - \Phi^{(a)}_{\mathcal{G}'}\|^2,$$

and the second one is

$$\left\| \frac{\partial \Theta(\mathcal{C}'_j, \mathcal{G}'^{(b)})}{\partial \Phi^{(b)}_{\mathcal{G}'}} - \frac{\partial \Theta(\mathcal{C}'_j, \mathcal{G}'^{(a)})}{\partial \Phi^{(a)}_{\mathcal{G}'}} \right\| \leqslant L \|\Phi^{(b)}_{\mathcal{G}'} - \Phi^{(a)}_{\mathcal{G}'}\|.$$

We derive the following inequation from the first condition

$$\left[\frac{2\left(\Theta(\mathcal{C}'_j, \mathcal{G}'^{(b)}) - \Theta(\mathcal{C}'_j, \mathcal{G}'^{(a)})\right)}{\|\Phi^{(b)}_{\mathcal{G}'} - \Phi^{(a)}_{\mathcal{G}'}\|^2} - \frac{2\left(\frac{\partial \Theta(\mathcal{C}'_j, \mathcal{G}'^{(a)})}{\partial \Phi^{(a)}_{\mathcal{G}'}} (\Phi^{(b)}_{\mathcal{G}'} - \Phi^{(a)}_{\mathcal{G}'})\right)}{\|\Phi^{(b)}_{\mathcal{G}'} - \Phi^{(a)}_{\mathcal{G}'}\|^2} \right] \leqslant L.$$

Similarly, we can obtain the inequation from the second condition as follows,

$$\frac{\left\| \frac{\partial \Theta(\mathcal{C}'_j, \mathcal{G}'^{(b)})}{\partial \Phi^{(b)}_{\mathcal{G}'}} - \frac{\partial \Theta(\mathcal{C}'_j, \mathcal{G}'^{(a)})}{\partial \Phi^{(a)}_{\mathcal{G}'}} \right\|}{\|\Phi^{(b)}_{\mathcal{G}'} - \Phi^{(a)}_{\mathcal{G}'}\|} \leqslant L.$$

Let $\alpha = \frac{2\left(\Theta(\mathcal{C}'_j, \mathcal{G}'^{(b)}) - \Theta(\mathcal{C}'_j, \mathcal{G}'^{(a)}) - \frac{\partial \Theta(\mathcal{C}'_j, \mathcal{G}'^{(a)})}{\partial \Phi^{(a)}_{\mathcal{G}'}} (\Phi^{(b)}_{\mathcal{G}'} - \Phi^{(a)}_{\mathcal{G}'})\right)}{\|\Phi^{(b)}_{\mathcal{G}'} - \Phi^{(a)}_{\mathcal{G}'}\|^2}$, $\beta = \frac{\left\| \frac{\partial \Theta(\mathcal{C}'_j, \mathcal{G}'^{(b)})}{\partial \Phi^{(b)}_{\mathcal{G}'}} - \frac{\partial \Theta(\mathcal{C}'_j, \mathcal{G}'^{(a)})}{\partial \Phi^{(a)}_{\mathcal{G}'}} \right\|}{\|\Phi^{(b)}_{\mathcal{G}'} - \Phi^{(a)}_{\mathcal{G}'}\|}$. To verify whether the two items of L-smooth are widely applicable to the imputation cost of various datasets, we present α and β of the PCDIH algorithm with six processors trained for 50 epochs.

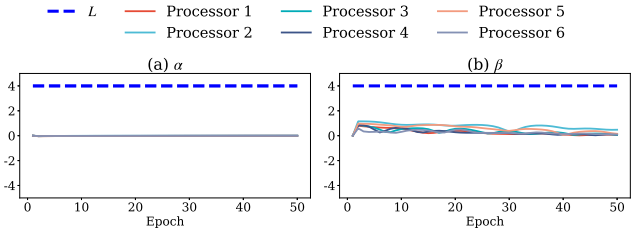


Fig. 16. L-smooth items over Ethanol dataset with 10% missing values

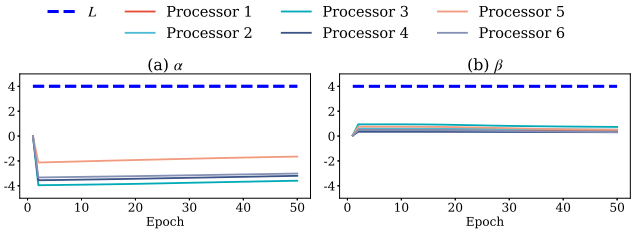


Fig. 17. L-smooth items over MIMIC-III dataset with 21.84% missing values

As shown in Figures 14-20, it is easy to find an L (e.g. 4 for Energy, AirQuality, Ethanol, MIMIC-III, GPS, 20 for IMU, 130 for Weer) satisfying the two items of L-smooth for the training process of the PCDIH model on various datasets. Note that since Ethanol and MIMIC-III datasets are preprocessed with Z-score normalization, as shown in Figure 16 and Figure 17, α and β values of these two datasets are smoother.

I. Memory Consumption

Table V presents the experimental results of the memory consumption for different methods over various datasets with different missing rates. According to the memory consumption in Table V and time costs in Table III, our methods have the reasonable scalability on large-scale Ethanol and Weer datasets, which demonstrate the contributions of our parallel imputation strategies.

J. Imputation Contexts

We report the number of maximal self-connected contexts over various datasets with different missing rates in Table VI. It can be noticed that the vast majority of scenarios have multiple maximal self-connected contexts, demonstrating the rationality of our parallel algorithms. In addition, there are two maximal self-connected contexts in the IMU dataset with real missing values, since the main source of these missing data is that the recording starts with all the inertial measurement unit sensors closed, leading to consecutive incomplete tuples. This condition will increase the difficulty of data imputation while also affecting the efficiency of the parallel imputation. However, our algorithms can still be more efficient than most baselines in Tables III and V, which demonstrates the efficiency of our algorithms, even for the real-world incomplete datasets containing few maximal self-connected contexts.

K. Imputation Rounds

For all the experiments, we repeat the procedure five times under different random seeds over various datasets with

TABLE V

IMPUTATION MEMORY CONSUMPTION (IN MIB) OF THE APPROACHES OVER VARIOUS DATASETS WITH DIFFERENT MISSING RATES

Approach	Energy					Ethanol					AirQuality				MIMIC-III			GPS	IMU	Weer
	10%	20%	30%	40%	80%	10%	20%	30%	40%	80%	20%	30%	40%	80%	30%	40%	80%	42.98%	0.46%	0.08%
BTMF	217.0	233.8	217.1	220.7	236.8	194.4	194.4	194.4	194.6	194.8	216.1	217.4	210.8	214.8	246.2	249.0	279.5	199.7	228.7	531.1
RecovDB	93.5	93.5	94.9	95.3	94.8	137.4	143.0	148.6	151.7	168.0	86.6	86.7	86.6	88.4	103.4	109.8	109.8	78.9	83.3	143.7
ORBITS	133.1	143.4	153.6	143.4	122.9	163.8	153.6	184.3	174.1	143.4	61.4	92.2	143.4	92.2	61.4	71.7	184.3	51.2	122.9	235.5
BayOTIDE	1130.0	1130.3	1130.4	1130.4	1096.0	1084.9	1084.7	1087.7	1080.8	981.2	715.4	715.9	716.0	703.2	3441.9	3416.0	3097.7	463.0	889.3	4382.7
BRITS	2813.4	2817.0	2818.2	2817.9	2818.9	2855.3	2859.2	2861.7	2864.6	2903.0	2813.5	2813.4	2815.0	2815.5	2830.1	2830.5	2835.3	2810.3	2811.6	3163.8
SAITS	2570.3	2575.0	2573.9	2573.7	2576.8	2570.2	2636.1	2643.6	2648.0	2648.0	2566.0	2567.8	2571.9	2570.2	2588.2	2591.8	2592.5	2562.0	2562.3	2568.5
GRIN	5312.5	5319.1	5321.2	5319.4	5320.4	5399.0	5399.9	5401.4	5400.4	5398.2	5321.3	5320.7	5321.2	5325.5	5334.8	5332.0	5336.7	5306.8	5309.4	5413.4
DAMR	1147.0	1168.4	1145.9	1147.4	1169.7	2295.9	2318.8	2306.7	2310.1	2295.1	874.4	875.3	868.2	868.8	1235.4	1238.5	1239.6	634.0	719.1	38863.1
E2GAN	3274.7	3272.6	3278.0	3278.7	3282.4	3317.7	3322.0	3324.0	3327.8	3363.0	3274.1	3267.8	3270.3	3275.7	3293.3	3296.6	3293.6	3264.2	3264.2	1248.9
MIWAE	2225.7	2227.4	2228.9	2227.4	2230.6	2316.2	2322.6	2326.9	2330.4	2347.0	2219.1	2221.8	2221.4	2224.0	2250.0	2250.3	2252.4	2217.9	2221.3	5413.4
CSDI	2411.2	2416.6	2418.0	2419.0	2420.3	2482.8	2487.1	2495.0	2499.3	2515.5	2411.9	2412.4	2416.3	2415.4	2553.9	2543.9	2536.3	2406.7	2409.5	2611.7
PriSTI	2562.0	2560.6	2561.8	2560.8	2564.2	2601.3	2602.5	2602.2	2599.5	2601.4	2557.2	2558.5	2558.1	2558.4	2568.1	2567.3	2568.7	2548.8	2553.9	2663.7
PCDI	393.9	402.6	413.9	442.6	517.7	317.1	320.4	326.6	329.0	399.1	387.5	523.4	523.3	702.1	857.1	994.2	1088.9	303.2	294.1	436.1
PCDIH	497.4	521.8	583.6	693.3	762.5	319.7	316.2	318.8	323.6	414.3	393.4	525.6	546.1	721.5	948.5	1003.2	1092.3	303.7	295.7	441.8

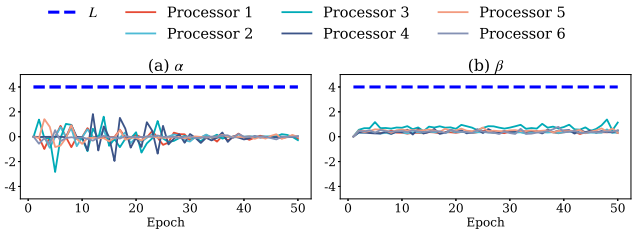


Fig. 18. L-smooth items over GPS dataset with 42.98% missing values

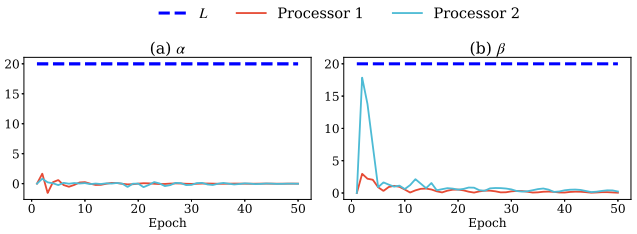


Fig. 19. L-smooth items over IMU dataset with 0.46% missing values

different missing rates, and we report the average results. Moreover, we add the new Table VII to detail the average number of imputation rounds under different settings for our methods. The imputation rounds are mainly influenced by several factors, such as the distribution of imputation contexts in different processors, temporal window size ω, dependency models $\mathcal{G}$ and the learning rate η. For instance, when using parallel algorithms, the rounds required for imputation on different processors may vary due to the differing difficulty levels in filling the incomplete imputation contexts, leading to variations in the number of rounds on different processors. Additionally, when the imputation temporal window size ω or the number of dependent models $\mathcal{G}$ increases, more intertemporal dependencies are considered, which may increase the number of rounds due to the rise in the task complexity. Similarly, when the learning rate η is too small, more imputation rounds are required to the convergence.

L. Intratemporal and Intertemporal Patterns

To illustrate the relationships among attribute values, we show all the intratemporal and intertemporal patterns in each dataset next. Specifically, to produce the dependency models of intratemporal and intertemporal patterns, for each attribute value, we extract the complete imputation contexts that can model the intertemporal dependency on this attribute over time and the intratemporal dependency between all the attribute values at the same time, following the process shown in Figure

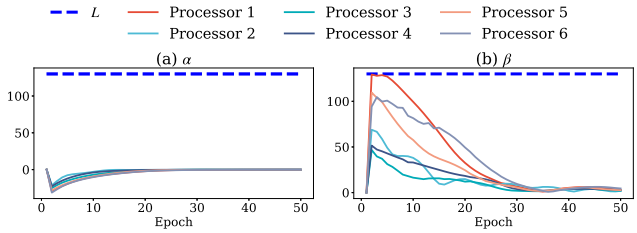


Fig. 20. L-smooth items over Weer dataset with 0.08% missing values

TABLE VI

NUMBER OF MAXIMAL SELF-CONNECTED CONTEXTS OVER VARIOUS DATASETS WITH DIFFERENT MISSING RATES

Dataset	Missing rate	Number of maximal self-connected contexts
Energy	10%	1495.80
	20%	995.60
	30%	431.60
	40%	138.60
	80%	1
Ethanol	10%	90.84
	20%	133.37
	30%	140.51
	40%	122.57
	80%	4.79
AirQuality	20%	26.80
	30%	7.20
	40%	4
	80%	1
MIMIC-III	30%	2552.4
	40%	1164.2
	80%	1
GPS	42.98%	110
IMU	0.46%	2
Weer	0.08%	195

3. We then train the dependency models using the gradient descent method over the complete imputation contexts for each attribute in turn. After training, the weights between different attributes at the same timestamp in the dependency model is associated with the intratemporal dependency and the intertemporal dependency is related to the weights between different timestamps for the same attribute, as defined in the previous works [63], [12].

As we can see below, for each dependency model, there typically exist different weights associated with different attribute values or the same attribute values with different timestamps, and the attribute values with larger weights contribute more to determine the predicted value.

Energy collects the sensor readings with nine attributes, i.e., T1 (A_1), T2 (A_2), T3 (A_3), T4 (A_4), T5 (A_5), T6 (A_6), T7 (A_7), T8 (A_8) and T9 (A_9). Nine corresponding dependency

TABLE VII
AVERAGE IMPUTATION ROUNDS OF OUR METHODS OVER VARIOUS DATASETS WITH DIFFERENT MISSING RATES

Approach	Energy					Ethanol					AirQuality				MIMIC-III			GPS	IMU	Weer
	10%	20%	30%	40%	80%	10%	20%	30%	40%	80%	20%	30%	40%	80%	30%	40%	80%	42.98%	0.46%	0.08%
CDI	16.20	47.80	108.00	191.60	272.80	49.50	66.26	87.64	105.80	139.37	33.20	21.20	59.40	84.00	93.69	92.78	107.79	24.00	11.00	80.00
CDIH	18.00	124.20	350.80	581.60	615.40	50.00	68.30	84.72	109.78	122.24	32.60	33.00	51.00	73.80	95.78	96.66	108.33	30.00	6.00	83.00
PCDI	21.50	52.10	109.50	191.87	272.80	52.06	69.70	90.98	109.12	139.37	27.50	21.93	62.50	84.00	125.27	130.77	107.79	48.80	8.00	74.16
PCDIH	22.00	127.10	348.43	581.47	615.40	52.48	71.67	87.71	112.47	122.24	31.37	36.07	46.23	73.80	133.74	143.24	108.33	70.16	12.50	89.00

models (in the form of Formula 1) are listed below.

$$
\begin{aligned}
x_{i1} =&(4.27e-3)x_{i2}+(1.61e-3)x_{i3}+(5.34e-4)x_{i4}\\
&-(8.10e-4)x_{i5}+(1.47e-3)x_{i6}+(1.13e-3)x_{i7}\\
&+(1.45e-3)x_{i8}-(6.66e-4)x_{i9}+(6.20e-1)x_{i-1,1}\\
&+(3.79e-1)x_{i+1,1}+(4.00e-4)\\
x_{i2} =&-(3.70e-3)x_{i1}+(1.61e-3)x_{i3}+(1.05e-3)x_{i4}\\
&+(1.53e-4)x_{i5}-(2.37e-4)x_{i6}-(2.47e-4)x_{i7}\\
&+(9.65e-4)x_{i8}+(2.99e-3)x_{i9}+(5.57e-1)x_{i-1,2}\\
&+(4.49e-1)x_{i+1,2}+(1.50e-3)\\
x_{i3} =&-(2.62e-3)x_{i1}+(5.83e-4)x_{i2}-(1.00e-3)x_{i4}\\
&-(2.25e-3)x_{i5}-(9.82e-4)x_{i6}+(1.16e-5)x_{i7}\\
&-(8.67e-4)x_{i8}-(1.77e-3)x_{i9}+(5.95e-1)x_{i-1,3}\\
&+(4.05e-1)x_{i+1,3}-(7.00e-4)\\
x_{i4} =&(-2.12e-3)x_{i1}+(2.20e-3)x_{i2}+(1.00e-3)x_{i3}\\
&+(1.37e-3)x_{i5}+(1.36e-3)x_{i6}-(-5.30e-4)x_{i7}\\
&+(1.37e-3)x_{i8}+(5.98e-4)x_{i9}+(5.97e-1)x_{i-1,4}\\
&+(4.08e-1)x_{i+1,4}+(1.00e-3)\\
x_{i5} =&-(1.96e-3)x_{i1}-(7.71e-4)x_{i2}-(1.36e-3)x_{i3}\\
&-(9.06e-4)x_{i4}-(3.24e-4)x_{i6}-(4.67e-4)x_{i7}\\
&-(1.23e-3)x_{i8}-(2.89e-3)x_{i9}+(5.42e-1)x_{i-1,5}\\
&+(4.60e-1)x_{i+1,5}+(8.00e-4)\\
x_{i6} =&-(1.97e-3)x_{i1}+(3.04e-3)x_{i2}+(2.55e-4)x_{i3}\\
&+(1.03e-3)x_{i4}-(2.79e-4)x_{i5}-(1.90e-4)x_{i7}\\
&-(5.95e-4)x_{i8}+(5.48e-3)x_{i9}+(5.79e-1)x_{i-1,6}\\
&+(4.23e-1)x_{i+1,6}+(1.80e-3)\\
x_{i7} =&-(1.65e-3)x_{i1}+(3.19e-4)x_{i2}+(3.19e-4)x_{i3}\\
&-(1.57e-4)x_{i4}-(1.36e-3)x_{i5}+(5.36e-4)x_{i6}\\
&-(3.92e-4)x_{i8}-(1.59e-3)x_{i9}+(6.15e-1)x_{i-1,7}\\
&+(3.84e-1)x_{i+1,7}+(6.00e-4)\\
x_{i8} =&-(1.16e-3)x_{i1}+(1.99e-4)x_{i2}-(9.50e-4)x_{i3}\\
&-(5.73e-4)x_{i4}-(2.00e-3)x_{i5}+(9.76e-4)x_{i6}\\
&-(5.35e-4)x_{i7}-(2.67e-3)x_{i9}+(6.14e-1)x_{i-1,8}\\
&+(3.85e-1)x_{i+1,8}+(1.30e-3)\\
x_{i9} =&(2.12e-3)x_{i1}+(5.83e-4)x_{i2}+(4.64e-4)x_{i3}\\
&+(4.14e-4)x_{i4}-(3.64e-4)x_{i5}+(8.49e-4)x_{i6}\\
&+(5.77e-4)x_{i7}+(1.53e-5)x_{i8}+(6.14e-1)x_{i-1,9}\\
&+(3.90e-1)x_{i+1,9}+(9.00e-4)
\end{aligned}
$$

Ethanol contains first measurement readings (A_1), second measurement readings (A_2) and third measurement readings (A_3), representing the raw spectral time series of water and ethanol solutions in whisky bottles. Please see the full list of dependency models on Ethanol below.

$$
\begin{aligned}
x_{i1} =&0.24x_{i2}+0.17x_{i3}-0.02x_{i-1,1}+0.62x_{i+1,1}-0.01\\
x_{i2} =&0.24x_{i1}-0.05x_{i3}+0.06x_{i-1,2}+0.75x_{i+1,2}+0.01\\
x_{i3} =&0.10x_{i1}-0.10x_{i2}+0.22x_{i-1,3}+0.77x_{i+1,3}-(1.36e-5)
\end{aligned}
$$

AirQuality has a schema over CO (A_1), PT08.S1 (A_2), NMHC (A_3), C6H6 (A_4), PT08.S2 (A_5), NOx (A_6), PT08.S3 (A_7), NO2 (A_8), PT08.S4 (A_9), PT08.S5 (A_{10}), T (A_{11}), RH (A_{12}) and AH (A_{13}). The dependency models are below.

$$
\begin{aligned}
x_{i1} =&0.16x_{i2}+0.15x_{i3}+0.13x_{i4}+0.21x_{i5}+0.23x_{i6}\\
&+0.22x_{i7}+0.15x_{i8}-0.07x_{i9}-0.11x_{i10}-0.21x_{i11}\\
&-0.09x_{i12}+0.19x_{i13}+0.15x_{i-1,1}+0.10x_{i+1,1}-0.08\\
x_{i2} =&0.30x_{i1}-0.22x_{i3}+0.11x_{i4}+0.18x_{i5}+0.02x_{i6}\\
&+0.11x_{i7}+0.11x_{i8}+0.55x_{i9}+0.03x_{i10}-0.14x_{i11}\\
&-0.03x_{i12}+0.06x_{i13}+0.21x_{i-1,2}+0.14x_{i+1,2}-0.08\\
x_{i3} =&1.24x_{i1}-0.39x_{i2}+1.97x_{i4}-1.18x_{i5}-0.80x_{i6}\\
&-0.35x_{i7}-0.13x_{i8}+0.27x_{i9}-0.08x_{i10}-0.29x_{i11}\\
&-0.09x_{i12}+0.08x_{i13}+0.20x_{i-1,3}+0.13x_{i+1,3}+0.42\\
x_{i4} =&0.04x_{i1}-0.03x_{i2}+0.09x_{i3}+0.46x_{i5}+0.05x_{i6}\\
&+0.13x_{i7}-0.03x_{i8}+0.16x_{i9}+0.05x_{i10}+0.04x_{i11}\\
&+0.01x_{i12}-0.08x_{i13}+0.05x_{i-1,4}+0.07x_{i+1,4}-0.17\\
x_{i5} =&0.17x_{i1}+0.09x_{i2}-0.06x_{i3}+0.66x_{i4}-0.11x_{i6}\\
&-0.29x_{i7}-0.04x_{i8}+0.23x_{i9}-0.01x_{i10}+0.16x_{i11}\\
&+0.08x_{i12}-0.26x_{i13}+0.11x_{i-1,5}+0.10x_{i+1,5}+0.15\\
x_{i6} =&0.14x_{i1}+0.01x_{i2}-0.04x_{i3}+0.37x_{i4}-0.28x_{i5}\\
&-0.07x_{i7}-0.03x_{i8}+0.18x_{i9}-0.01x_{i10}+0.04x_{i11}\\
&+0.07x_{i12}-0.18x_{i13}+0.40x_{i-1,6}+0.18x_{i+1,6}+0.03\\
x_{i7} =&0.28x_{i1}-0.10x_{i2}-0.07x_{i3}+0.12x_{i4}-0.48x_{i5}\\
&-0.04x_{i6}-0.06x_{i8}+0.23x_{i9}-0.02x_{i10}+0.19x_{i11}\\
&+0.08x_{i12}-0.27x_{i13}+0.28x_{i-1,7}+0.29x_{i+1,7}+0.18\\
x_{i8} =&0.18x_{i1}+0.18x_{i2}+0.04x_{i3}+0.01x_{i4}+0.04x_{i5}\\
&-0.01x_{i6}+0.10x_{i7}-0.23x_{i9}+0.10x_{i10}+0.01x_{i11}\\
&-0.01x_{i12}+0.01x_{i13}+0.41x_{i-1,8}+0.38x_{i+1,8}-0.01\\
x_{i9} =&-0.05x_{i1}+0.08x_{i2}+0.03x_{i3}+0.44x_{i4}+0.15x_{i5}\\
&+0.13x_{i6}-0.02x_{i7}-0.18x_{i8}+0.02x_{i10}+0.25x_{i11}\\
&+0.1x_{i12}+0.08x_{i13}+0.03x_{i-1,9}+0.08x_{i+1,9}+0.03
\end{aligned}
$$

$$
\begin{aligned}
x_{i10} =& -0.10x_{i1} + 0.19x_{i2} - 0.03x_{i3} + 0.53x_{i4} - 0.39x_{i5} \\
& - 0.06x_{i6} - 0.15x_{i7} + 0.05x_{i8} + 0.44x_{i9} - 0.41x_{i11} \\
& - 0.16x_{i12} + 0.09x_{i13} + 0.44x_{i-1,10} + 0.25x_{i+1,10} + 0.19 \\
x_{i11} =& 0.01x_{i1} - 0.01x_{i2} - 0.03x_{i3} - 0.05x_{i4} - 0.01x_{i5} \\
& - 0.09x_{i6} + 0.05x_{i7} + 0.07x_{i8} + 0.22x_{i9} - 0.05x_{i10} \\
& - 0.25x_{i12} + 0.29x_{i13} + 0.24x_{i-1,11} + 0.32x_{i+1,11} + 0.10 \\
x_{i12} =& -0.11x_{i1} - 0.08x_{i2} - 0.05x_{i3} + 0.03x_{i4} - 0.13x_{i5} \\
& + 0.10x_{i6} - 0.01x_{i7} + 0.05x_{i8} + 0.32x_{i9} - 0.03x_{i10} \\
& - 0.81x_{i11} + 0.56x_{i13} + 0.26x_{i-1,12} + 0.27x_{i+1,12} + 0.28 \\
x_{i13} =& 0.03x_{i1} + 0.07x_{i2} + 0.01x_{i3} + 0.8x_{i4} - 0.25x_{i5} \\
& - 0.02x_{i6} - 0.05x_{i7} - 0.03x_{i8} + 0.13x_{i9} + 0.03x_{i10} \\
& + 0.17x_{i11} + 0.01x_{i12} + 0.48x_{i-1,13} + 0.34x_{i+1,13}
\end{aligned}
$$

MIMIC-III records the daily data of patients, including Hours (A_1), Heart Rate (A_2), Mean blood pressure (A_3), Oxygen saturation (A_4), Respiratory rate (A_5) and Systolic blood pressure (A_6). Below is the list of dependency models.

$$
\begin{aligned}
x_{i1} =& 0.01x_{i2} - 0.02x_{i3} - 0.16x_{i4} + 0.01x_{i5} - 0.01x_{i6} \\
& + 0.47x_{i-1,1} + 0.54x_{i+1,1} + 0.10 \\
x_{i2} =& 0.02x_{i1} + 0.36x_{i3} + 0.05x_{i4} + 0.20x_{i5} - 0.27x_{i6} \\
& + 0.21x_{i-1,2} + 0.28x_{i+1,2} + 0.08 \\
x_{i3} =& -0.01x_{i1} + 0.17x_{i2} - 0.12x_{i4} - 0.04x_{i5} 0.83x_{i6} \\
& - 0.01x_{i-1,3} + 0.05x_{i+1,3} + 0.02 \\
x_{i4} =& 0.01x_{i1} + 0.07x_{i2} + 0.02x_{i3} - 0.04x_{i5} - 0.01x_{i6} \\
& + 0.15x_{i-1,4} + 0.35x_{i+1,4} + 0.26 \\
x_{i5} =& -0.02x_{i1} + 0.73x_{i2} - 0.44x_{i3} - 0.05x_{i4} + 0.31x_{i6} \\
& + 0.18x_{i-1,5} - 0.03x_{i+1,5} + 0.19 \\
x_{i6} =& 0.02x_{i1} - 0.16x_{i2} + 1.18x_{i3} + 0.18x_{i4} + 0.05x_{i5} \\
& + 0.01x_{i-1,6} - 0.04x_{i+1,6} - 0.08
\end{aligned}
$$

GPS consists of trajectory data with Sat-Lon (A_1), Sat-Lat (A_2), Map-Lon (A_3), Map-Lat (A_4), Altitude (A_5), Speed (A_6), hAccuracy (A_7) and vAccuracy (A_8) attributes. Similarly, eight dependency models are also listed below.

$$
\begin{aligned}
x_{i1} =& 0.05x_{i2} - 0.01x_{i3} - 0.05x_{i4} + 0.01x_{i5} - 0.01x_{i6} \\
& - 0.01x_{i7} - 0.01x_{i8} + 0.04x_{i-1,1} + 0.06x_{i+1,1} - 0.01 \\
x_{i2} =& -0.01x_{i1} - 0.01x_{i3} - 0.03x_{i4} - 0.01x_{i5} - 0.01x_{i6} \\
& 0.01x_{i7} + 0.01x_{i8} + 0.39x_{i-1,2} + 0.58x_{i+1,2} + 0.01 \\
x_{i3} =& 0.07x_{i1} + 0.14x_{i2} - 0.14x_{i4} - 0.14x_{i5} + 0.01x_{i6} \\
& - 0.01x_{i7} + 0.01x_{i8} + 0.39x_{i-1,3} + 0.58x_{i+1,3} - 0.04 \\
x_{i4} =& 0.01x_{i1} + 0.22x_{i2} - 0.06x_{i3} + 0.01x_{i5} - 0.01x_{i6} \\
& + 0.01x_{i7} + 0.01x_{i8} + 0.30x_{i-1,4} + 0.49x_{i+1,4} - 0.06 \\
x_{i5} =& 0.03x_{i1} + 0.03x_{i2} - 0.03x_{i3} - 0.02x_{i4} - 0.01x_{i6} \\
& - 0.01x_{i7} - 0.01x_{i8} + 0.53x_{i-1,5} + 0.48x_{i+1,5} - 0.01 \\
x_{i6} =& -0.03x_{i1} + 0.26x_{i2} - 0.06x_{i3} - 0.22x_{i4} - 0.13x_{i5} \\
& - 0.09x_{i7} - 0.03x_{i8} + 0.43x_{i-1,6} + 0.47x_{i+1,6} + 0.11 \\
x_{i7} =& -0.07x_{i1} + 0.05x_{i2} + 0.01x_{i3} - 0.04x_{i4} - 0.01x_{i5} \\
& - 0.02x_{i6} + 0.01x_{i8} + 0.47x_{i-1,7} + 0.46x_{i+1,7} + 0.07 \\
x_{i8} =& 0.02x_{i1} + 0.05x_{i2} - 0.01x_{i3} - 0.05x_{i4} - 0.01x_{i5} \\
& - 0.01x_{i6} + 0.01x_{i7} + 0.50x_{i-1,8} + 0.50x_{i+1,8} - 0.01
\end{aligned}
$$

IMU consists of the observations from an inertial measurement unit with ax_raw (A_1), ay_raw (A_2), az_raw (A_3), wx_raw (A_4), wy_raw (A_5), wz_raw (A_6). Below is a full list of dependency models over IMU dataset. Specially, we can observe that there is no strong intratemporal dependency between the different attributes, as they record data in different axes of the inertial measurement units. Therefore, it can be seen that our methods can self-adaptively learn small weights for other attributes (with parameters for $x_{i1}, \ldots, x_{i6}$) and mainly focus on intratemporal dependencies (with parameters for $x_{i-1,j}$, $x_{i+1,j}$) of the dependency model g_j.

$$
\begin{aligned}
x_{i1} =& -0.001x_{i2} + 0.001x_{i3} - 0.001x_{i4} - 0.003x_{i5} - 0.002x_{i6} \\
& + 0.415x_{i-1,1} + 0.585x_{i+1,1} + 0.003 \\
x_{i2} =& -0.001x_{i1} + 0.001x_{i3} + 0.002x_{i4} + 0.001x_{i5} + 0.001x_{i6} \\
& + 0.417x_{i-1,2} + 0.583x_{i+1,2} - 0.002 \\
x_{i3} =& 0.001x_{i1} + 0.001x_{i2} + 0.001x_{i4} - 0.001x_{i5} - 0.001x_{i6} \\
& + 0.479x_{i-1,3} + 0.521x_{i+1,3} - 0.001 \\
x_{i4} =& 0.001x_{i1} - 0.001x_{i2} - 0.001x_{i3} + 0.01x_{i5} + 0.01x_{i6} \\
& + 0.461x_{i-1,4} + 0.541x_{i+1,4} - 0.001 \\
x_{i5} =& 0.001x_{i1} - 0.001x_{i2} + 0.001x_{i3} - 0.001x_{i4} + 0.001x_{i6} \\
& + 0.417x_{i-1,5} + 0.584x_{i+1,5} - 0.001 \\
x_{i6} =& 0.001x_{i1} - 0.001x_{i2} - 0.001x_{i3} - 0.001x_{i4} - 0.001x_{i5} \\
& + 0.423x_{i-1,6} + 0.578x_{i+1,6} - 0.001
\end{aligned}
$$

Weer records weather data including Wind direction averaged over last 10 minutes of the past hour / Mean wind direction during the 10 minutes period preceding the time of observation (A_1), Hourly mean wind speed (A_2), Wind speed average over the last 10 minutes of the past hour / Mean wind direction during the 10 minutes period preceding the time of observation (A_3), Highest wind gust over the past hour / Maximum wind gust during the hourly division (A_4), Temperature (A_5), Please see the full list of dependency models below.

$$
\begin{aligned}
x_{i1} =& -0.10x_{i2} - 0.16x_{i3} + 0.22x_{i4} + 0.01x_{i5} \\
& + 0.32x_{i-1,1} + 0.32x_{i+1,1} + 0.07 \\
x_{i2} =& -0.01x_{i1} + 0.34x_{i3} + 0.27x_{i4} - 0.01x_{i5} \\
& + 0.25x_{i-1,2} + 0.12x_{i+1,2} - 0.01 \\
x_{i3} =& -0.01x_{i1} + 0.75x_{i2} + 0.13x_{i4} + 0.01x_{i5} \\
& - 0.05x_{i-1,3} + 0.19x_{i+1,3} - 0.01 \\
x_{i4} =& 0.01x_{i1} + 0.48x_{i2} + 0.06x_{i3} + 0.01x_{i5} \\
& + 0.25x_{i-1,4} + 0.22x_{i+1,4} - 0.01 \\
x_{i5} =& 0.01x_{i1} - 0.02x_{i2} + 0.03x_{i3} - 0.01x_{i4} \\
& + 0.5x_{i-1,5} + 0.5x_{i+1,5} - 0.01
\end{aligned}
$$