HomeRun: Scalable Sparse-Spectrum Reconstruction of Aggregated Historical Data

Faisal M. Almutairi
University of Minnesota
Minneapolis, MN
almut012@umn.edu

Fan Yang
University of Pittsburgh
Pittsburgh, PA
fay28@pitt.edu

Hyun Ah Song
Carnegie Mellon University
Pittsburgh, PA
hyunahs@cs.cmu.edu

Christos Faloutsos
Carnegie Mellon University
Pittsburgh, PA
christos@cs.cmu.edu

Nicholas Sidiropoulos
University of Virginia
Charlottesville, VA
nikos@virginia.edu

Vladimir Zadorozhny
University of Pittsburgh
Pittsburgh, PA
vladimir@sis.pitt.edu

ABSTRACT

Recovering a time sequence of events from multiple aggregated and possibly overlapping reports is a major challenge in historical data fusion. The goal is to reconstruct a higher resolution event sequence from a mixture of lower resolution samples as accurately as possible. For example, we may aim to disaggregate overlapping monthly counts of people infected with measles into weekly counts. In this paper, we propose a novel data disaggregation method, called HOMERUN, that exploits an alternative representation of the sequence and finds the spectrum of the target sequence. More specifically, we formulate the problem as so-called *basis pursuit* using the Discrete Cosine Transform (DCT) as a sparsifying dictionary *and* impose non-negativity and smoothness constraints. HOMERUN utilizes the energy compaction feature of the DCT by finding the sparsest spectral representation of the target sequence that contains the largest (most important) coefficients. We leverage the *Alternating Direction Method of Multipliers* to solve the resulting optimization problem with scalable and memory efficient steps. Experiments using real epidemiological data show that our method considerably outperforms the state-of-the-art techniques, especially when the DCT of the sequence has a high degree of energy compaction.

PVLDB Reference Format:
F.Almutairi, F.Yang, H.Song, C.Faloutsos, N.Sidiropoulos, V.Zadorozhny. HomeRun: Scalable Sparse-Spectrum Reconstruction of Aggregated Historical Data. *PVLDB*, 11 (11): 1496-1508, 2018.
DOI: https://doi.org/10.14778/3236187.3236201

1. INTRODUCTION

Gathering and analyzing information from multiple historical data sources requires reconstructing the time sequences in finer scale. For example, given multiple *monthly* sums of patient counts, how can we recover the *weekly* patient counts? This is so-called data disaggregation problem [26].

Proceedings of the VLDB Endowment, Vol. 11, No. 11

DOI: https://doi.org/10.14778/3236187.3236201

Notable challenges of historical data disaggregation are: 1) each data source may report the aggregated sums on *different scales* (e.g., one data source may report the weekly number of patients while another source reports on monthly scale), 2) the *time periods* covered by different data sources *may or may not overlap* (e.g., one source may report the number of patients for years 1920-1930, and another for 1940-1950, resulting in missing information for years 1930-1940), and 3) the reports may have conflicts (e.g., one data source may report 100 patients while another may report 80 patients for the same time period). Our informal problem definition is given as follows:

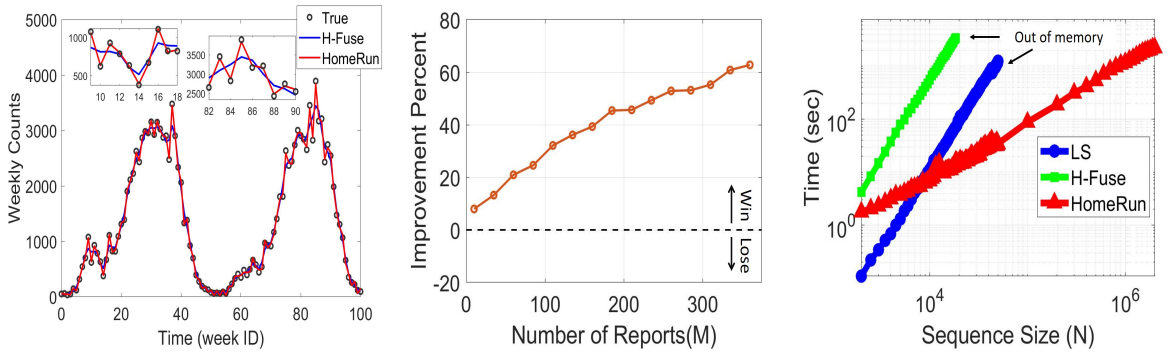
INFORMAL PROBLEM 1 (DISAGGREGATION).

1. *Given: the multiple reports of the aggregated sums of the time sequence (e.g., monthly sums)*
2. *Recover: the time sequence in finer scale (e.g., weekly sums)*

The prevailing approach is to formulate the problem as linear Least Squares (LS), however, as we will explain in more details later, this problem is usually under-determined in practice. In cases where the number of available reports is much smaller than the length of the target sequence, the LS approach becomes inefficient. There have been previous works for solving the disaggregation problem that add different regularizers to the LS, such as smoothness and periodicity in the data. Enforcing smoothness and periodicity in the time domain is a reasonable approach since many of the time sequences that we observe are smooth and *quasi-periodic* in nature. The main issue with smoothness and periodicity regularized LS is the lack of identifiability, especially when the time series is not exactly smooth, nor exactly periodic.

In this work, we propose HOMERUN – an efficient algorithm for solving the disaggregation problem in which we exploit an alternative representation of the target time sequence. More specifically, we search for the coefficients that best represent the sequence in a fixed dictionary of cosine basis, i.e., we solve for the coefficients of the Discrete Cosine Transform (DCT) of the sequence we are seeking. As we will explain in the following section, DCT with few non-zeros represents a sum of few cosines, i.e., few dominant periodicities. Therefore, DCT transformation is a good basis for quasi-periodic historical data. Moreover, expressing the time sequence using the DCT basis functions provides a *sparse* representation as most of the energy is compacted in the coefficients of low frequencies.

We formulate the data disaggregation in the form of the so-called *Basis Pursuit* (BP) where we enforce sparsity in the DCT



(a) More accurate reconstruction (b) HOMERUN wins (c) Linear scalability

Figure 1: HOMERUN is effective and scalable: (a) visible improvement of HOMERUN over the baseline method H-FUSE; (b) performance of HOMERUN versus H-FUSE across different number of reports; (c) HOMERUN is memory efficient and scales linearly with the length of the target sequence.

coefficients of the target sequence. We call the resulting BP formulation HOMERUN-0, which is the basic version of our proposed method. One significant advantage of our approach is that it automatically detects the prominent periodicities in the data, as opposed to the methods in related works, which assume that there is only one or few known periodicities. In addition to the periodicity, other common domain knowledge properties of the time sequences are non-negativity and smoothness over timestamps. We also propose HOMERUN-N method that improves HOMERUN-0 by enforcing non-negativity constraint on the time domain sequence. We further extend our method by imposing smoothness in the time sequence in addition to non-negativity, resulting in the final version of the proposed method: HOMERUN. We derive the steps of the *Alternating Direction Method of Multipliers* (ADMM) algorithm that can solve the optimization problem after adding non-negativity and smoothness constraints. Finally, we derive a scalable and memory efficient implementation of HOMERUN.

We apply HOMERUN to the epidemiological data from the Tycho project [27]. Our dataset contains the number of cases for major epidemic diseases (hepatitis A, measles, mumps, pertussis, polio, rubella, and smallpox) in the US over 100 years. We demonstrate that HOMERUN helps to recover the time sequences much better than the competing baseline methods, H-FUSE [21] and LS.

Figure 1 shows an example of the results of HOMERUN when reconstructing the weekly counts of measles, given multiple aggregated reports. We observe in Fig. 1 (a) that HOMERUN is closer to the true sequence compared to the baseline H-FUSE, HOMERUN estimates the number of patients with Root Mean Square Error (RMSE $=$ 20.30), while the RMSE of H-FUSE is 104.23. In data analysis, e.g., studying the impact of vaccination, not only the average error matters, but also the weekly single error. We can see that for several weeks, H-FUSE underestimates (or overestimates) the counts by the order of hundreds, while HOMERUN is very close to the true value. Fig. 1 (b) shows the percentage of improvement/diminishment in the RMSE between HOMERUN and the baseline H-FUSE with various numbers of given aggregated reports – HOMERUN always improves the RMSE, '70' means HOMERUN reduces the RMSE of H-FUSE by 70%, and so on. Fig. 1 (c) compares the running time of HOMERUN with the baselines. It shows how HOMERUN is memory efficient and scales linearly in time with the sequence length (up to 2 million) – note the log scales. In summary, the contributions of our work are as follows:

- **Formulation and Algorithm**: we propose to formulate the data disaggregation problem in the form of so-called Basis Pursuit (BP), add domain knowledge constraints, and derive the iterative updates of the ADMM algorithm to solve the resulting optimization problem.
- **Effectiveness**: our HOMERUN method recovers the time sequences with up to 94% improvement in the accuracy of the best of baseline methods.
- **Scalability**: we derive an efficient accelerated implementation of HOMERUN that scales linearly with the length of the target sequence.
- **Adaptability**: HOMERUN is parameter-free and it adapts to the input signal and automatically detects the prominent periodicities in the data.

Reproducibility: The Tycho dataset is publicly available [27]; we plan to release the code upon publication of the paper.

The paper structure is as follows. We explain the necessary background and the related work in section 2, and introduce our proposed method in section 3. Then, we explain our experimental setup in section 4 and show the experimental results in section 5. We conclude in section 6.

2. BACKGROUND

In this section, we provide background on both the problem of historical data disaggregation *and* the techniques we employ in the proposed approach to solve this problem. We also review the related work relevant to both the problem and the proposed method.

Notation: bold capital letters (e.g., $\mathbf{A}$) denote matrices; bold small letters (e.g., $\mathbf{x}$) denote column vectors; $\mathbf{A}^\dagger$ is the Moore-Penrose pseudo-inverse of a matrix $\mathbf{A}$; $\mathbf{A}^T$ denotes the transpose of $\mathbf{A}$. x_n is the n^{th} element in vector $\mathbf{x}$. $(\mathbf{a})_+$ is the non-negative projection of a vector $\mathbf{a}$, performed by zeroing out the negative elements in $\mathbf{a}$. Table 1 summarizes the symbols we use frequently.

2.1 Historical Data Disaggregation

Data disaggregation considered in this work is a special case of data fusion as we aim to reconstruct an unknown time sequence from multiple aggregated observations with possible overlaps. For example, consider reconstructing the weekly counts of infection incidents (e.g., by measles) in the United States from reports aggregated over multiple weeks. In general, those aggregated

Table 1: Symbols and Definitions

Symbol	Definition
$\mathbf{y}$	$\in \mathbb{R}^M$ vector contains the known measurements
$\mathbf{x}$	$\in \mathbb{R}^N$ the target time sequence
$\mathbf{s}$	$\in \mathbb{R}^N$ sparse presentation of $\mathbf{x}$ in fixed basis
$\mathbf{O}$	$\in \mathbb{R}^{M \times N}$ observation matrix
$\mathbf{D}$	matrix of DCT basis functions
RD	report duration
$shift$	difference between the starts of adjacent reports
H	$\in \mathbb{R}^{(N-1) \times N}$ smoothness matrix

reports could have overlaps, gaps or conflicts between them. Figure 2 shows an illustrative example of each case, respectively.

Formally, we want to reconstruct a detailed time sequence $\mathbf{x} = \{x_n\}_{n=1}^{N}$, given the aggregated observations $\mathbf{y} = \{y_m\}_{m=1}^{M}$, where y_m corresponds to the sum of multiple elements in $\mathbf{x}$. Thus, we specify a linear system $\mathbf{y} = \mathbf{O}\mathbf{x}$, where each row of an observation matrix $\mathbf{O} \in \mathbb{R}^{M \times N}$ is a binary vector that has ones for the elements of $\mathbf{x}$ that contribute in y_m (see Example in Eq. 1). We refer to the number of timeticks covered by a report as *Report Duration* (RD), i.e., ones in the m^{th} row of $\mathbf{O}$, and the difference between the starting points of two successive reports as $shift$ (marked in blue in Eq. 1). Observed reports may have different RD values, e.g., we may have one report covering a month and another covering two weeks. In any case, we can sort the reports according to their starting points. Below is an illustrative example containing three reports with $RD = 4, 4$, and 2, and $shift = 1$ between the first two reports and $shift = 2$ between the 2^{nd} and 3^{rd} ones. Note that the reports in the example have overlaps (marked in green), however they could have gaps if the $shift$ between two reports is larger than RD of the first one. Moreover, conflict occurs when two rows of $\mathbf{O}$ are identical, but with different y_m values.

$$\underbrace{\begin{bmatrix} 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}}_{\mathbf{O}} \times \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix}}_{\mathbf{x}} = \underbrace{\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}}_{\mathbf{y}} \tag{1}$$

If $\mathbf{O}$ is square (i.e., $N = M$) and full rank, then the solution is trivial and error free. In practical settings, the resulting system of linear equations is under-determined (number of reports $\ll$ number of timeticks in the target sequence). In this case, the linear system has many solutions and Least Square (LS) solution finds $\mathbf{x}$ with minimum norm (min $\|\mathbf{x}\|_2^2$). However, there is no special reason why the best reconstructed sequence would have the minimum norm for this problem, which led researchers to add domain knowledge penalty terms to the linear system [21] to improve the LS solution.

Instead of solving for $\mathbf{x}$ directly, as it is common in the literature for this problem, we exploit an alternative signal representation and propose to solve for the DCT representation of the target sequence (as formulated in Section 3). We define the DCT in the next section before proceeding to the proposed methods.

2.2 Discrete Cosine Transform

Discrete Cosine Transform (DCT) transforms a finite-length discrete-time data sequence from the time (or spatial) domain into the frequency domain. In particular, DCT represents the finite-length sequence in terms of a sum of *basis sequences*. These basis are cosines oscillating at different frequencies [1, 22]. We focus here on one-dimensional DCT since the problem of our interest is the reconstruction of one-dimensional sequence. Formally, the most common DCT definition of a data sequence $\mathbf{x}$ of length N is as follows [19]:

Index	Event	Begin Time	End Time	Count
1	Measles	1st Week	4th Week	300
2	Measles	3th Week	7th Week	400

Index	Event	Begin Time	End Time	Count
1	Measles	1st Week	4th Week	300
2	Measles	7th Week	11th Week	400

Index	Event	Begin Time	End Time	Count
1	Measles	1st Week	4th Week	300
2	Measles	1st Week	4th Week	400

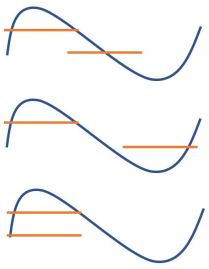


Figure 2: Historical data examples: overlap; gap; conflict (from top to bottom).

$$s_k = \sum_{n=0}^{N-1} x_n \underbrace{\alpha(k) cos\left(\frac{\pi k(2n+1)}{2N}\right)}_{\phi(k,n)} = \sum_{n=0}^{N-1} x_n \phi(k,n) \tag{2}$$

for $0 \leq k \leq N-1$, where $\alpha(k)$ is a coefficient factor defined as follows:

$$\alpha(k) = \begin{cases} \sqrt{\frac{1}{N}}, & k = 0, \\ \sqrt{\frac{2}{N}}, & 1 \leq k \leq N-1. \end{cases} \tag{3}$$

Similarly, the original finite-length sequence can be uniquely recovered from its DCT using the inverse DCT (iDCT) defined as:

$$x_n = \sum_{k=0}^{N-1} s_k \phi(k,n) \tag{4}$$

for $0 \leq n \leq N-1$.

To facilitate concisely formulating the problem, we define a DCT matrix $\mathbf{D} \in \mathbb{R}^{N \times N}$ whose entries are the cosine basis functions:

$$\mathbf{D} = \begin{bmatrix} \phi(0,0) & \dots & \phi(0,N-1) \\ \vdots & \ddots & \\ \phi(N-1,0) & \dots & \phi(N-1,N-1) \end{bmatrix} = \begin{bmatrix} \mathbf{d}_0^T \\ \vdots \\ \mathbf{d}_{N-1}^T \end{bmatrix} \tag{5}$$

where, as it is clear from (2), $\phi(k,n) = \alpha(k) cos(\pi k(2n+1)/2N)$. The inner product of any row $\mathbf{d}_n$ with itself is 1, while the inner product of any two different rows is 0. Thus, $\mathbf{D}$ is orthogonal (and DCT is an orthogonal transform [22]), i.e., $\mathbf{D}^{-1} = \mathbf{D}^T$. Equations (2) and (4) can be written as:

$$\mathbf{s} = \mathbf{D}\mathbf{x} \tag{6}$$

$$\mathbf{x} = \mathbf{D}^T\mathbf{s} \tag{7}$$

Since cosines are *periodic* and *even symmetric*, the DCT transform imposes periodicity in the time domain signal [22]. An important property of DCT is energy compaction, which is the reason why DCT is widely used in many data compression applications, such as image compression [25]. Specifically, the DCT of a signal is usually concentrated in the coefficients of the low

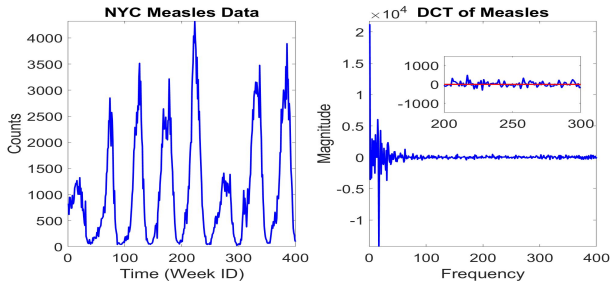


Figure 3: NYC measles data in time domain, $\mathbf{x}$ (left) and its spectrum, $\mathbf{s}$ (right).

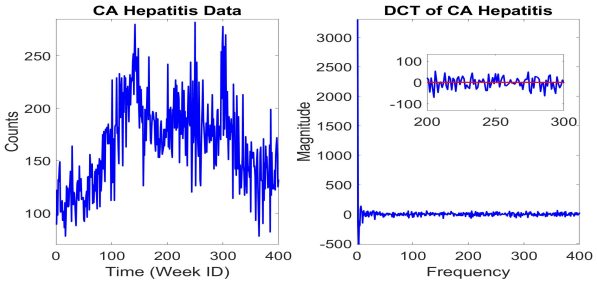


Figure 4: CA hepatitis data in time domain, $\mathbf{x}$ (left) and its spectrum, $\mathbf{s}$ (right).

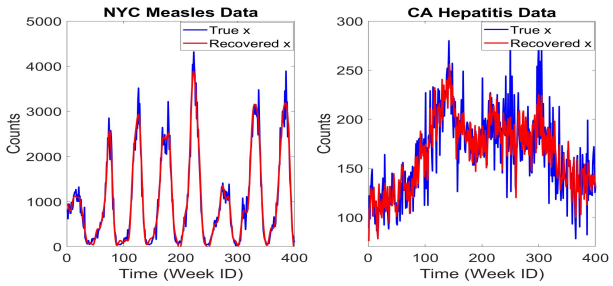


Figure 5: Data recovered from the largest 10% coefficients of their DCT – NYC measles (left) and CA hepatitis (right).

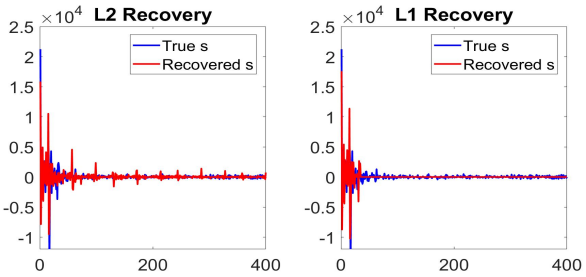


Figure 6: Recovery with $L1$ norm (left) and recovery with $L2$ norm by replacing $L1$ by $L2$ in (8) (right).

frequencies and the remaining coefficients can be discarded without a significant impact [22]. The degree of DCT energy compaction depends on how correlated the original signal is in time (or spatial) domain. For example, in image processing, the DCT energy compaction of an image relies on the correlation degree between its pixels. We demonstrate this phenomenon by showing New York (NYC) measles and California (CA) hepatitis weekly counts and their DCT in Figure 3 and 4, respectively. We can see that CA hepatitis sequence is less correlated (less smooth) in time domain, and therefore its DCT has high frequency components that are larger than in the case of NYC measles (*relative to their maximum values*) as clear in the zoomed parts.

If we discard the small coefficients of DCT, the DCT representation of the signal becomes sparse. In other words, although the reports of those diseases do not have zero values across all timeticks, most of their DCT coefficients are small, and the few large coefficients carry most of the energy and capture most of the information. We show an illustrative example by keeping only the largest 10% of the DCT coefficients of NYC measles and CA hepatitis weekly counts sequences and set the rest to zero, i.e., we pick the largest 10% elements in $\mathbf{s}$ and zero out the rest. In Figure 5, we show the time sequence of both data sets recovered from this 10% (using Equation (7)). It is clear that NYC measles has a better recovery since its DCT is sparser (compact and has less significant components). Finally, we should note that the ability of accurately estimating the DCT coefficients ($\mathbf{s}$) of a sequence enables us to recover this sequence in time-domain ($\mathbf{x}$). In the following section, we explain the basics of sparse reconstruction since it is essential to our methods.

2.3 Sparse Signal Recovery

The goal of sparse reconstruction and compressive sensing [5] is to find a *sparse* approximate solution $\mathbf{s}$ of an under-determined linear system $\mathbf{As} = \mathbf{y}$, where $\mathbf{A} \in \mathbb{R}^{M \times N}, \mathbf{s} \in \mathbb{R}^N, \mathbf{y} \in \mathbb{R}^M$, with $M < N$. Then, $\mathbf{s}$ could be recovered by solving the following convex problem known as BP [8]:

$$\begin{aligned} \min_{\mathbf{s}} \quad & \|\mathbf{s}\|_1 \\ s.t. \quad & \mathbf{As} = \mathbf{y} \end{aligned} \tag{8}$$

where $\|s\|_1 = \sum_{n=1}^{N} |\mathbf{s}_n|$ is the $L1$ norm which promotes sparsity in the solution. In general, $\mathbf{A}$ may be a matrix containing the elements of an over-complete dictionary [13] and we seek to solve for the sparsest coefficient vector $\mathbf{s}$ to represent the observed measurements $\mathbf{y}$.

Now we will consider a more practical scenario: suppose we have a (non-sparse) signal in time-domain $\mathbf{x} \in \mathbb{R}^N$ under-sampled in such a way that we have fewer linear measurements $\mathbf{y} \in \mathbb{R}^M$ about $\mathbf{x}$ in the form $\mathbf{y} = \mathbf{\Phi x}$, where $\mathbf{\Phi} \in \mathbb{R}^{M \times N}$. Thus, we are interested in solving for the unknown signal $\mathbf{x}$ given the observed measurements $\mathbf{y}$. In the case of $M \ll N$ where there are much fewer measurements than the unknowns, solving the linear problem may appear too challenging. However, if $\mathbf{x}$ can be compressed (accurately represented as sparse coefficients on some fixed basis) such that the number of the non-zero coefficients that carry most of the energy is less than N (the size of $\mathbf{x}$), then this changes the problem radically, making the search for solutions feasible [5]. In particular, suppose we have a sparse vector $\mathbf{s}$ that contains the coefficients of a time (spatial)-domain signal $\mathbf{x}$ in an orthonormal basis $\mathbf{\Psi}$, i.e., $\mathbf{x} = \mathbf{\Psi s}$. For example, $\mathbf{x}$ is the vector containing pixels of an image, $\mathbf{s}$ is the coefficient sequence of $\mathbf{x}$ in the wavelet basis, and $\mathbf{\Psi}$ is an $N \times N$ matrix containing the *wavelet basis functions* as its entries [7]. In this case, we would recover the coefficient sequence $\mathbf{s}$ with the minimum $L1$ norm that satisfies $\mathbf{As} = \mathbf{y}$, where $\mathbf{A} = \mathbf{\Phi\Psi}$ in problem (8).

As mentioned above, BP is often used as a heuristic algorithm for finding a sparse solution to an under-determined system of linear equations and $L1$ promotes sparsity in the solution. In Figure 6, we

illustrate the advantage of using $L1$ norm by showing the solution we get from (8) and when replacing the $L1$ norm by $L2$ norm. In this example, we try to recover the DCT representation of NYC measles data (shown in Figure 3) from 29 measurements in the time domain (each measurement has the sum of counts over 21 weeks with overlaps). We can see that the two solutions are different. The $L2$ solution does not give a good approximation as it has spikes where the original signal is almost zero.

Alternating Direction Method of Multipliers: problem (8) can be recast as a linear program. However, we propose to use the ADMM algorithm as it is well suited for large-scale problems. ADMM solves the convex optimization of the following form

$$\begin{aligned} \min_{\mathbf{s},\mathbf{z}} \quad & f(\mathbf{s}) + g(\mathbf{z}) \\ s.t. \quad & \mathbf{A}\mathbf{s} + \mathbf{E}\mathbf{z} = \mathbf{c}. \end{aligned} \tag{9}$$

by iteratively updating the following blocks

$$\mathbf{s} \leftarrow arg\min_{\mathbf{s}} f(\mathbf{s}) + (\rho/2)\|\mathbf{A}\mathbf{s} + \mathbf{E}\mathbf{z} - \mathbf{c} + \mathbf{u}\|_2^2, \tag{10a}$$

$$\mathbf{z} \leftarrow arg\min_{\mathbf{z}} g(\mathbf{z}) + (\rho/2)\|\mathbf{A}\mathbf{s} + \mathbf{E}\mathbf{z} - \mathbf{c} + \mathbf{u}\|_2^2, \tag{10b}$$

$$\mathbf{u} \leftarrow \mathbf{u} + (\mathbf{A}\mathbf{s} + \mathbf{E}\mathbf{z} - \mathbf{c}) \tag{10c}$$

where $\mathbf{u}$ is a scaled version of the dual variable corresponding to the equality constraint in (9), and $\rho > 0$ is the augmented Lagrangian parameter specified by the user.

Problem (8) can be reformulated as follows after introducing the auxiliary variable $\mathbf{z} \in \mathbb{R}^N$:

$$\begin{aligned} \min_{\mathbf{s},\mathbf{z}} \quad & \mathcal{I}_{\{\mathbf{A}\mathbf{s}=\mathbf{y}\}} + \|\mathbf{z}\|_1 \\ s.t. \quad & \mathbf{s} - \mathbf{z} = \mathbf{0} \end{aligned} \tag{11}$$

where $\mathcal{I}_{\{\mathbf{A}\mathbf{s}=\mathbf{y}\}}$ is an indicator function such that:

$$\mathcal{I}_{\{\mathbf{A}\mathbf{s}=\mathbf{y}\}} = \begin{cases} 0 & \text{if } \mathbf{A}\mathbf{s} = \mathbf{y} \\ \infty & \text{otherwise.} \end{cases}$$

We will skip the derivation of the algorithm for brevity – refer to [4] for more comprehensive review of the ADMM algorithm. The solution to (8) is provided by the following iterative updates:

$$\mathbf{s}^{k+1} \leftarrow (\mathbf{I} - \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A})(\mathbf{z}^k - \mathbf{u}^k) + \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{y}$$

$$\mathbf{z}^{k+1} \leftarrow (\mathbf{s}^{k+1} + \mathbf{u}^k - 1/\rho)_+ - (-\mathbf{s}^{k+1} - \mathbf{u}^k - 1/\rho)_+$$

$$\mathbf{u}^{k+1} \leftarrow \mathbf{u}^k + (\mathbf{s}^{k+1} - \mathbf{z}^{k+1})$$

2.4 Related Work

Disaggregation: The aggregation of the data vector can be seen as representing or summarizing the data vector using linear transform. In [10, 9], the idea of sketches has been introduced as means of data aggregation or summarization. With the advance of the data collection technologies, we have been gaining more access to various sources of historical data in aggregated form. This has led to an increasing interest in data integration and fusion including, in particular, disaggregation of the data sources [3, 12, 24, 21, 29, 15]

The disaggregation problem is of interest in various domains. In the image and signal processing communities, for example, there have been works on solving under-determined problems for various applications, such as super-resolution reconstruction of image data [23], or information recovery from noisy or missing data [6].

In recent work [21], the authors proposed an algorithm called H-FUSE that enforces smoothness and periodicity constraints for the reconstruction of the historical data. The authors show that the proposed algorithm improves the reconstruction compared to the minimum-norm linear LS formulation, which is the most common formulation for solving the disaggregation problem. We will use this algorithm as our main baseline.

DCT and Sparse reconstruction: DCT is one of the most commonly used compression techniques in the signal processing community. It has been shown to be an effective transformation for compressing the data in large networks [20]. DCT is widely used especially for image compression [28] due to its energy compaction property, and for image denoising [16, 14]. DCT has been also used in databases community for answering queries in compressed form via DCT transform [17], representing the time series in spectral domain [11], etc.

The principle of finding a sparse signal representation in a basis dictionary has been used in various applications, such as denoising [8] and information recovery from incomplete and/or noisy measurements [6]. Basis Pursuit (BP) formulation is used to obtain such a sparse solution to the ill-posed problem. Expressing a signal in a proper basis (dictionary), where it is sparse for the purpose of recovering this signal from fewer linear measurements has been used in compressive sensing [5]. As an example, wavelet basis has been considered in order to sparsely represent an image to recover it from fewer measurements [7]. In [14], DCT is used as a dictionary for sparse representation of a noisy image for the purpose of removing the noise from the image.

To our knowledge, the application of DCT and sparse representation has not been exploited in historical data fusion domain. Table 2 summarizes our proposed HOMERUN method compared to the related approaches.

Table 2: HOMERUN satisfies all properties listed above.

Property	LS	H-FUSE	HOMERUN
Overlapping reports	✓	✓	✔
Smoothness reconstruction		✓	✔
Periodicity reconstruction		✓	✔
Multiple periods (quasi-periodic)			✔
Automatically detects periodicity			✔
Non-negativity			✔

3. PROPOSED METHOD: HOMERUN

In this section, we explain our proposed method HOMERUN and the algorithmic solutions associated with it. Recollect, that the objective of reconstruction methods is finding the disaggregated sequence $\mathbf{x}$ that minimizes the following problem:

$$\min_{\mathbf{x}} \quad \|\mathbf{y} - \mathbf{O}\mathbf{x}\|_2^2 \tag{12}$$

where $\mathbf{O}, \mathbf{x}, \mathbf{y}$ have the same definition as in Section 2.1. More advanced methods are proposed to infuse domain knowledge, such as smoothness and periodicity, by penalizing (12) [21]. The role of this penalty is to make the under-determined linear system (that has infinite number of solution) an over-determined one, constraining the solution to adhere to some domain knowledge. All these methods solve the problem directly by the closed form of LS using *Moore-Penrose pseudo-inverse*. However, LS solution does not

always give a good approximation, especially when the number of observations is much less than the number of unknown variables.

The main idea behind our proposed method is to deal with the under-determinacy of the linear system by solving for the coefficient vector $\mathbf{s}$ that represents the target sequence $\mathbf{x}$ in the DCT basis as the number of non-zero coefficients is much less than the length of the sequence. The accuracy of this reconstruction hinges on the degree of DCT energy compaction feature explained in Section 2.2. Moreover, DCT involves implicit assumptions of periodicity which makes it a good dictionary for this problem as the time sequence exhibits some degree of periodicity. A significant advantage of the proposed approach is that it automatically detects the prominent periodicities in the data, as opposed to assuming that there is only one or few *known* periodicities by constraining the LS as in [21]. In the rest of this section, we explain our proposed method in the order as it was derived.

- HOMERUN-0: the basic version of our method.
- HOMERUN-N: with added non-negativity constraint.
- **HOMERUN: the final and complete version of the proposed method.**

3.1 HOMERUN-0: The Basic Version

DCT matrix (defined in Equation (5)) offers a convenient way to compute the transform and its inverse as follows: $\mathbf{s} = \mathbf{D}\mathbf{x}$ (Equation (6)) is the DCT of the time sequence $\mathbf{x}$, and $\mathbf{x} = \mathbf{D}^T\mathbf{s}$ (Equation (7)) reconstructs the time sequence from $\mathbf{s}$. The following Insight shows the formulation of our HOMERUN-0 method.

INSIGHT 1. *The historical data disaggregation problem can be formulated in the form of Basis Pursuit as follows:*

$$\begin{aligned} \min_{\mathbf{s}} \quad & \|\mathbf{s}\|_1 \\ s.t. \quad & \mathbf{A}\mathbf{s} = \mathbf{y} \end{aligned} \tag{13}$$

RATIONALE 1. *Given* $\mathbf{O}\mathbf{x} = \mathbf{y}$*, we want to find the* sparse *vector that contains the DCT of the target sequence* $\mathbf{x}$*. Since minimizing the* $L1$ *norm promotes sparsity in the solution, we look for the minimum* $\|\mathbf{s}\|_1$ *that satisfies* $\mathbf{O}\mathbf{x} = \mathbf{y}$*. Replacing* $\mathbf{x}$ *by* $\mathbf{D}^T\mathbf{s}$*, we get the problem in the following form:*

$$\begin{aligned} \min_{\mathbf{s}} \quad & \|\mathbf{s}\|_1 \\ s.t. \quad & \mathbf{O}\mathbf{D}^T\mathbf{s} = \mathbf{y} \end{aligned} \tag{14}$$

Note that (14) *is similar to* (13) *with* $\mathbf{A} = \mathbf{O}\mathbf{D}^T$.

We solve the above problem using the ADMM algorithm with the iterative updates presented in Section 2.3. After we get the solution to $\mathbf{s}$, the approximate solution of the target sequence is obtained as $\mathbf{x}_{\text{HOMERUN-0}} = \mathbf{D}^T\mathbf{s}$.

Conflicting reports: if there is a conflict between the reports (as explained in Fig. 2), then the linear system $\mathbf{O}\mathbf{x} = \mathbf{y}$ is inconsistent. As a result, the constraints in (14) can not be satisfied. We resolve this issue by the following preprocessing step: if $\mathbf{O} \in \mathbb{R}^{M\times N}$ is full row rank (i.e., rank($\mathbf{O}$) $= M$), then there is no conflict and we proceed with the algorithm to solve (14). If the rows of $\mathbf{O}$ have some linear dependency (i.e., rank($\mathbf{O}$) $< M$), then we have one of two cases: a) if $\mathbf{y} \in$ span($\mathbf{O}$) (column space of $\mathbf{O}$), then the system is consistent (there is no conflict) and we proceed with the algorithm; b) if $\mathbf{y} \notin$ span($\mathbf{O}$), then we have an inconsistent linear system. In this case, we replace $\mathbf{y}$ with its projection onto the span($\mathbf{O}$), $\overline{\mathbf{y}}$ as follows

$$\overline{\mathbf{y}} = proj_{\mathbf{O}}(\mathbf{y}) = \mathbf{O}(\mathbf{O}^T\mathbf{O})^{-1}\mathbf{O}^T\mathbf{y} \tag{15}$$

This orthogonal projection results in the nearest vector (set of reports) to $\mathbf{y}$ that is free of conflict. Previous methods for this problem (H-FUSE and LS) provide solutions that minimize the squared error in case of conflicts. Assuming that flawed reports are rare (correct reports are the norm), we would normally want to satisfy as many equations as possible, instead of using an inconsistent solution that minimizes the squared norm of the violations but could violate all equations. This turns out to be NP-hard however [2]. The advantage of our projection approach is that the solution satisfies *all* the equations in the linear system with $\overline{\mathbf{y}}$, which is the closest vector to $\mathbf{y}$ that belongs to the column space of $\mathbf{O}$. Note that this applies to the coming optimization formulations (HOMERUN-N, and HOMERUN).

3.2 HOMERUN-N: The Non-Negative Version

In the applications of our interest, the target sequence $\mathbf{x}$ is always non-negative. Therefore, we exploit this domain knowledge by adding the constraint $\mathbf{x} \geq \mathbf{0} \Leftrightarrow \mathbf{D}^T\mathbf{s} \geq \mathbf{0}$ to (14). The resulting formulation becomes:

$$\begin{aligned} \min_{\mathbf{s}} \quad & \|\mathbf{s}\|_1 \\ s.t. \quad & \mathbf{O}\mathbf{D}^T\mathbf{s} = \mathbf{y}, \quad \mathbf{D}^T\mathbf{s} \geq \mathbf{0} \end{aligned} \tag{16}$$

STATEMENT 1. *The ADMM algorithm can be adapted to solve the optimization formulation of* HOMERUN-N *in Equation* (16)*.*

PROOF. Problem (16) is convex and we reformulate it by introducing the auxiliary variables $\mathbf{r}, \mathbf{z} \in \mathbb{R}^N$ as follows

$$\begin{aligned} \min_{\mathbf{r},\mathbf{s},\mathbf{z}} \quad & \mathcal{I}_{\{\mathbf{r}\geq\mathbf{0}\}} + \|\mathbf{s}\|_1 \\ s.t. \quad & \mathbf{O}\mathbf{D}^T\mathbf{z} = \mathbf{y}, \quad \mathbf{D}^T\mathbf{z} = \mathbf{r}, \quad \mathbf{z} = \mathbf{s} \end{aligned} \tag{17}$$

where, again, $\mathcal{I}$ is an indicator function of $\{\mathbf{r} \in \mathbb{R}^N\colon\ \mathbf{r} \geq \mathbf{0}\}$ defined as:

$$\mathcal{I}_{\{\mathbf{r}\geq\mathbf{0}\}} = \begin{cases} 0 & \text{if } \mathbf{r} \geq \mathbf{0} \\ \infty & \text{otherwise.} \end{cases}$$

To facilitate concise notation and precisely present the algorithm using only matrix algebra, we define the following (components of $\mathbf{u}$ are defined and used later):

$$\mathbf{B} := \begin{bmatrix} \mathbf{O}\mathbf{D}^T \\ \mathbf{D}^T \\ \mathbf{I} \end{bmatrix}; \quad \mathbf{b} := \begin{bmatrix} \mathbf{y} \\ \mathbf{r} \\ \mathbf{s} \end{bmatrix}; \quad \mathbf{u} := \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \mathbf{u}_3 \end{bmatrix}, \tag{18}$$

where $\mathbf{I} \in \mathbb{R}^{N\times N}$ is the identity matrix. By concatenating the constraints as $\mathbf{B}\mathbf{z} - \mathbf{b} = \mathbf{0}$, it is straightforward to see that problem (17) above is in the form of the ADMM form defined in Equation (9) with $g(\mathbf{z}) = 0$, and $f(\mathbf{r}, \mathbf{s}) = \mathcal{I}_{\{\mathbf{r}\geq\mathbf{0}\}} + \|\mathbf{s}\|_1$. Thus, we can derive the ADMM iterative updates, starting by forming the augmented Lagrangian of (17):

$$\begin{aligned} \mathcal{L}_\rho(\mathbf{r}, \mathbf{s}, \mathbf{z}, \mathbf{u}) = \|s\|_1 + \mathcal{I}_{\{\mathbf{r}\geq\mathbf{0}\}} + \frac{\rho}{2}\big(\|\mathbf{O}\mathbf{D}^T\mathbf{z} - \mathbf{y} + \mathbf{u}_1\|_2^2 \\ + \|\mathbf{D}^T\mathbf{z} - \mathbf{r} + \mathbf{u}_2\|_2^2 + \|\mathbf{z} - \mathbf{s} + \mathbf{u}_3\|_2^2\big) \end{aligned} \tag{19}$$

where $\mathbf{u}_1 \in \mathbb{R}^M$, $\mathbf{u}_2, \mathbf{u}_3 \in \mathbb{R}^N$ are scaled versions of the dual variables, and ρ is the augmented Lagrangian parameter. We solve for $\mathbf{z}$, $\mathbf{s}$, $\mathbf{r}$, and $\mathbf{u}$ by minimizing $\mathcal{L}_\rho$ in terms of one variable while fixing the rest in an alternating optimization fashion. We let $\mathbf{z}$ be

the first block update, $(\mathbf{s}, \mathbf{r})$ is the second block update, and $\mathbf{u}$ is the third block update. Note that $\mathbf{s}$ and $\mathbf{r}$ can be updated independently since they do not appear together in one term in $\mathcal{L}_\rho$ (hence the parenthesis below). The solution to each block update is provided by solving the following optimization subproblems

$$\mathbf{z} \leftarrow arg\min_{\mathbf{z}} \mathcal{L}_\rho(\mathbf{r}, \mathbf{s}, \mathbf{z}, \mathbf{u}), \tag{20a}$$

$$\begin{cases} \mathbf{s} \leftarrow arg\min_{\mathbf{s}} \mathcal{L}_\rho(\mathbf{r}, \mathbf{s}, \mathbf{z}, \mathbf{u}), \\ \mathbf{r} \leftarrow arg\min_{\mathbf{r}} \mathcal{L}_\rho(\mathbf{r}, \mathbf{s}, \mathbf{z}, \mathbf{u}), \end{cases} \tag{20b}$$

$$\mathbf{u} \leftarrow \mathbf{u} + (\mathbf{B}\mathbf{z} - \mathbf{c}) \tag{20c}$$

where the solution to $\mathbf{z}$ is the closed form solution of least squares via Moore-Penrose pseudo-inverse. The solutions to $\mathbf{s}$, and $\mathbf{r}$ are cases of the so-called *proximity operator* (see [18] for details), where $\mathbf{s}$ is solved using the *soft-thresholding* and the solution to $\mathbf{r}$ boils down to non-negative projection $(.)_+$ by zeroing out the negative values. Algorithm 1 states all these updates using linear algebraic notations. □

Algorithm 1 : HOMERUN-N (16)

Initialization: set $k = 1$ and $\mathbf{z}^k$, $\mathbf{s}^k$, $\mathbf{r}^k$, and $\mathbf{u}^k$ to all zero vectors; $\mathbf{b}^k$ as defined in (18); compute the pseudo-inverse of $\mathbf{B}$ and save it (i.e., $\mathbf{R} = \mathbf{B}^\dagger$)

Repeat

- $\mathbf{z}^{k+1} = \mathbf{R}(\mathbf{b}^k - \mathbf{u}^k)$
- $\mathbf{s}^{k+1} = (\mathbf{z}^{k+1} + \mathbf{u}_3^k - 1/\rho)_+ - (-\mathbf{z}^{k+1} - \mathbf{u}_3^k - 1/\rho)_+$
- $\mathbf{r}^{k+1} = (\mathbf{D}^T\mathbf{z}^{k+1} + \mathbf{u}_2^k)_+$
- update $\mathbf{b}^{k+1}$ as defined in (18) using $\mathbf{s}^{k+1}$ and $\mathbf{r}^{k+1}$
- $\mathbf{u}^{k+1} = \mathbf{u}^k + (\mathbf{B}\mathbf{z}^{k+1} - \mathbf{b}^{k+1})$
- Set $k := k + 1$.

Until maximum number of iterations K is reached (K = 3000)

Since the same pseudo-inverse ($\mathbf{B}^\dagger$) is used throughout the iterations in Algorithm 1, we compute it once in the initialization step and cache it in a variable we call $\mathbf{R}$ to save computation. After $\mathbf{s}$ is obtained, the approximate solution of the target sequence is $\mathbf{x}_{\text{HOMERUN-N}} = \mathbf{D}^T\mathbf{s}$.

3.3 HOMERUN: The Final version

The main idea here is to exploit the domain knowledge of smoothness as for most cases the solution sequence $\mathbf{x} = \mathbf{D}^T\mathbf{s}$ should be smooth. Thus, we penalize the large differences between adjacent timeticks by adding the smoothness penalty to HOMERUN-N, resulting in the formulation of HOMERUN as follows:

$$\begin{aligned} \min_{\mathbf{s}} \quad & \|\mathbf{s}\|_1 + 1/2\|\mathbf{H}\mathbf{D}^T\mathbf{s}\|_2^2 \\ s.t. \quad & \mathbf{O}\mathbf{D}^T\mathbf{s} = \mathbf{y}, \quad \mathbf{D}^T\mathbf{s} \geq \mathbf{0} \end{aligned} \tag{21}$$

where $\mathbf{H} \in \mathbb{R}^{(N-1)\times N}$ is a smoothness matrix with the n^{th} row has -1 and 1 in the n^{th} and $(n+1)^{th}$ columns, respectively. One could add a regularization (weighting) parameter λ with the smoothness penalty above, however, we observe that $\lambda = 1$ gives optimal or near-optimal performance.

Although both HOMERUN and H-FUSE in [21] have the same regularizer (smoothness constraint), their approach to the problem is very different (i.e., same domain knowledge constraint is added to different optimization cost functions). Again, the approach in [21] penalizes the under-determined LS system and solve for the sequence using the closed form solution. HOMERUN solves for $\mathbf{s}$, the sparse DCT representation of the sequence using the $L1$ norm.

We propose to solve Equation (21) in a similar manner as HOMERUN-N model in the previous section. Thus, we reformulate (21) as follows:

$$\begin{aligned} \min_{\mathbf{r},\mathbf{s},\mathbf{z}} \quad & \mathcal{I}_{\{\mathbf{r}\geq\mathbf{0}\}} + \|\mathbf{s}\|_1 + 1/2\|\mathbf{H}\mathbf{D}^T\mathbf{z}\|_2^2 \\ s.t. \quad & \mathbf{O}\mathbf{D}^T\mathbf{z} = \mathbf{y}, \quad \mathbf{D}^T\mathbf{z} = \mathbf{r}, \quad \mathbf{z} = \mathbf{s} \end{aligned} \tag{22}$$

where $\mathbf{r}, \mathbf{z} \in \mathbb{R}^N$ are auxiliary variables. The augmented Lagrangian of (22) is:

$$\begin{aligned} \mathcal{L}_\rho(\mathbf{r}, \mathbf{s}, \mathbf{z}, \mathbf{u}) = \|s\|_1 + \mathcal{I}_{\{\mathbf{r}\geq\mathbf{0}\}} + 1/2\|\mathbf{H}\mathbf{D}^T\mathbf{z}\|_2^2 \\ + \frac{\rho}{2}\big(\|\mathbf{O}\mathbf{D}^T\mathbf{z} - \mathbf{y} + \mathbf{u}_1\|_2^2 \\ + \|\mathbf{D}^T\mathbf{z} - \mathbf{r} + \mathbf{u}_2\|_2^2 + \|\mathbf{z} - \mathbf{s} + \mathbf{u}_3\|_2^2\big) \end{aligned} \tag{23}$$

For the same reason as λ, we set the augmented Lagrangian parameter to $\rho = 1$ as it gives optimal or near-optimal performance, resulting in a parameter-free model. Recall the variables defined in (18), and similarly we define:

$$\mathbf{Q} := \begin{bmatrix} \mathbf{O}\mathbf{D}^T \\ \mathbf{D}^T \\ \mathbf{I} \\ \mathbf{H}\mathbf{D}^T \end{bmatrix}; \quad \mathbf{q} := \begin{bmatrix} \mathbf{y} \\ \mathbf{r} \\ \mathbf{s} \\ \mathbf{0} \end{bmatrix}; \quad \mathbf{v} := \begin{bmatrix} \mathbf{u} \\ \mathbf{0} \end{bmatrix}, \tag{24}$$

where $\mathbf{0} \in \mathbb{R}^{N-1}$ is a vector of all zeros. Similarly, we solve for $\mathbf{z}$, $\mathbf{s}$, $\mathbf{r}$, and $\mathbf{u}$ by minimizing $\mathcal{L}_\rho$ in (23) in terms of one variable while fixing the rest. We describe the implementation of HOMERUN in what follows.

3.3.1 *Direct Implementation of* HOMERUN

In Algorithm 2, we present the iterative updates that solve the optimization problem of HOMERUN with *direct* implementation. These steps provide a convenient way to understand HOMERUN using simple vector and matrix operations. Similarly, after we obtain $\mathbf{s}$, the approximate solution of the target sequence is $\mathbf{x}_{\text{HOMERUN}} = \mathbf{D}^T\mathbf{s}$.

Algorithm 2 : HOMERUN (21) (direct implementation)

Initialization: set $k = 1$ and $\mathbf{z}^k$, $\mathbf{s}^k$, $\mathbf{r}^k$, and $\mathbf{u}^k$ to all zero vectors; $\mathbf{b}^k$ as defined in (18); $\mathbf{q}^k$, $\mathbf{v}^k$ as defined in (24); compute the pseudo-inverse of $\mathbf{Q}$ and save it (i.e., $\mathbf{W} = \mathbf{Q}^\dagger$)

Repeat

- $\mathbf{z}^{k+1} = \mathbf{W}(\mathbf{q}^k - \mathbf{v}^k)$
- $\mathbf{s}^{k+1} = (\mathbf{z}^{k+1} + \mathbf{u}_3^k - 1)_+ - (-\mathbf{z}^{k+1} - \mathbf{u}_3^k - 1)_+$
- $\mathbf{r}^{k+1} = (\mathbf{D}^T\mathbf{z}^{k+1} + \mathbf{u}_2^k)_+$
- update $\mathbf{b}^{k+1}$ as defined in (18) using $\mathbf{s}^{k+1}$ and $\mathbf{r}^{k+1}$
- update $\mathbf{q}^{k+1}$ as defined in (24) using $\mathbf{s}^{k+1}$ and $\mathbf{r}^{k+1}$
- $\mathbf{u}^{k+1} = \mathbf{u}^k + (\mathbf{B}\mathbf{z}^{k+1} - \mathbf{b}^{k+1})$ ($\mathbf{B}$ as defined in (18))
- update $\mathbf{v}^{k+1}$ as defined in (24) using $\mathbf{u}^{k+1}$
- Set $k := k + 1$.

Until maximum number of iterations K is reached (K=3000)

3.3.2 *Accelerated Implementation of* HOMERUN

In this section, we analyze the complexity of Algorithm 2, and derive a fast and memory efficient implementation of HOMERUN.

The steps in Algorithm 2 provide a convenient way to implement HOMERUN using simple vector and matrix operations. These steps involve a one time Moore-Penrose pseudo-inverse in the initialization step which have a complexity of $\mathcal{O}(N^3)$, and the iterative updates consist of matrix-vector multiplications, vector additions, and element-wise vector updates with complexity dominated by $\mathcal{O}(nnz(\mathbf{O})N)$, where $nnz(\mathbf{O})$ is the number of non-zero in the observation matrix $\mathbf{O}$. Implementing these steps directly may be acceptable with small to moderate-size data since the matrix inversion need to be done only once. However, the direct implementation is not recommended for large data sets. Thus, we propose *accelerated* steps of HOMERUN explained in what follows.

Computing the pseudo-inverse of $\mathbf{Q}$ is the most time consuming step, which is used in the update of $\mathbf{z}$. We can state the update of $\mathbf{z}$ as

$$\begin{aligned}
\mathbf{z} &= \mathbf{Q}^{\dagger}(\mathbf{q}-\mathbf{v}) \\
&= (\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{Q}^T(\mathbf{q}-\mathbf{v}) \\
&= (\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{D}\underbrace{\big(\mathbf{O}^T(\mathbf{y}-\mathbf{u}_1)+(\mathbf{r}-\mathbf{u}_2)+\overbrace{\mathbf{D}^T(\mathbf{s}-\mathbf{u}_3)}^{\mathbf{g}\in\mathbb{R}^N}\big)}_{\mathbf{p}\in\mathbb{R}^N} \\
&= (\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{D}\mathbf{p} \\
&= \big(\mathbf{D}\mathbf{O}^T\mathbf{O}\mathbf{D}^T+\mathbf{D}\mathbf{D}^T+\mathbf{I}+\mathbf{D}\mathbf{H}^T\mathbf{H}\mathbf{D}^T\big)^{-1}\mathbf{D}\mathbf{p} \\
&= \big(\mathbf{D}(\mathbf{O}^T\mathbf{O}+2\mathbf{I}+\mathbf{H}^T\mathbf{H})\mathbf{D}^T\big)^{-1}\mathbf{D}\mathbf{p} \\
&= \mathbf{D}\underbrace{(\mathbf{O}^T\mathbf{O}+2\mathbf{I}+\mathbf{H}^T\mathbf{H})}_{\mathbf{Z}\in\mathbb{R}^{\mathbf{N}\times\mathbf{N}}}{}^{-1}\underbrace{\mathbf{D}^T\mathbf{D}}_{\mathbf{I}}\mathbf{p} \\
&= \mathbf{D}\mathbf{Z}^{-1}\mathbf{p}
\end{aligned} \tag{25}$$

multiplying both side by $\mathbf{D}^T$, we get:

$$\underbrace{\mathbf{D}^T\mathbf{z}}_{\mathbf{t}\in\mathbb{R}^N} = \mathbf{Z}^{-1}\mathbf{p} \tag{26}$$

where the second equality in (25) is by the pseudo-inverse equation for $\mathbf{Q}$ (since it is tall with linearly independent columns due to $\mathbf{I}$; Eq. (24)); the third and fifth equalities follow from the definition of $\mathbf{Q}$, $\mathbf{q}$, and $\mathbf{v}$; the sixth equality is because $\mathbf{D}$ is orthogonal, i.e., $\mathbf{D}\mathbf{D}^T = \mathbf{D}^T\mathbf{D} = \mathbf{I}$; and the seventh equality is because $(\mathbf{ABC})^{-1} = \mathbf{C}^{-1}\mathbf{B}^{-1}\mathbf{A}^{-1}$ and $\mathbf{D}^{-1} = \mathbf{D}^T$.

The goal of the above derivation is to reduce the computational cost of updating $\mathbf{z}$. Note that the left side of (26) (called $\mathbf{t}$) is the inverse DCT of $\mathbf{z}$ – denoted as $iDCT(\mathbf{z})$. Fortunately, $\mathbf{Z}$ has a nice structure, it is a banded symmetric positive-definite matrix with bandwidth = $2RD_{max}-1$, where RD_{max} is the duration of the report with the maximum length. Thus, we exploit its structure to efficiently compute the matrix inversion needed to get $\mathbf{t}$ as follows. We use *Cholesky decomposition*, i.e., $\mathbf{Z} = \mathbf{L}\mathbf{L}^T$, where $\mathbf{L}$ is a lower triangular matrix with the same bandwidth as $\mathbf{Z}$. Then, at every iteration, we perform forward substitution and back substitution steps to get $\mathbf{t}$, i.e., $\mathbf{t} = (\mathbf{L}^T)^{-1}\mathbf{L}^{-1}\mathbf{p}$. Moreover, we reduce the complexity by computing the required DCT and iDCT transforms more efficiently using Fast Fourier Transform (FFT) instead of using the matrix $\mathbf{D}$ (Matlab $fft(.)$ function is more efficient than multiplying by $\mathbf{D}$). The steps of the *accelerated* implementation of HOMERUN are presented in Algorithm 3.

Algorithm 3 : HOMERUN (21) (accelerated implementation)

Initialization: set $k=1$ and $\mathbf{g}^k$, $\mathbf{p}^k$, $\mathbf{t}^k$, $\mathbf{z}^k$, $\mathbf{s}^k$, $\mathbf{r}^k$, $\mathbf{u}_1^k$, $\mathbf{u}_2^k$ and $\mathbf{u}_3^k$ to all zero vectors; compute $\mathbf{L}$ from the *Cholesky decomposition* of $\mathbf{Z}$

Repeat

1. $\mathbf{g}^{k+1} = iDCT(\mathbf{s}^k - \mathbf{u}_3^k)$ %using fft in Matlab%
2. $\mathbf{p}^{k+1} = \mathbf{O}^T(\mathbf{y}-\mathbf{u}_1^k) + (\mathbf{r}^k - \mathbf{u}_2^k) + \mathbf{g}^{k+1}$
3. $\mathbf{t}^{k+1} = (\mathbf{L}^T)^{-1}\mathbf{L}^{-1}\mathbf{p}^{k+1}$ %Matlab: $t = L'\backslash(L\backslash p)$%
4. $\mathbf{z}^{k+1} = DCT(\mathbf{t}^{k+1})$ % using fft Matlab%
5. $\mathbf{s}^{k+1} = (\mathbf{z}^{k+1} + \mathbf{u}_3^k - 1)_+ - (-\mathbf{z}^{k+1} - \mathbf{u}_3^k - 1)_+$
6. $\mathbf{r}^{k+1} = (\mathbf{t}^{k+1} + \mathbf{u}_2^k)_+$
7. $\mathbf{u}_1^{k+1} = \mathbf{u}_1^k + \mathbf{O}\mathbf{t}^{k+1} - \mathbf{y}$
8. $\mathbf{u}_2^{k+1} = \mathbf{u}_2^k + \mathbf{t}^{k+1} - \mathbf{r}^{k+1}$
9. $\mathbf{u}_3^{k+1} = \mathbf{u}_3^k + \mathbf{z}^{k+1} - \mathbf{s}^{k+1}$
10. Set $k := k+1$.

Until maximum number of iterations K is reached (K=3000)

LEMMA 1. *For any report setting in the historical disaggregation problem, if $M \le N$, then the total computational complexity of* HOMERUN *(Algorithm 3) is*

$$\mathcal{O}(b^2N + NlogN) \tag{27}$$

where b is the bandwidth of $\mathbf{O}^T\mathbf{O}$ and equal to $2RD_{max}-1$, and RD_{max} is the duration of the longest report.

PROOF. The cost of computing Cholesky decomposition of a banded matrix in the initialization step is $\mathcal{O}(b^2N)$; performing the $iDCT$ in step 1 in Algorithm 3 and DCT in step 4 cost $\mathcal{O}(NlogN)$ using FFT; the matrix-vector multiplications in steps 2 and 7 have complexity of $\mathcal{O}(nnz(\mathbf{O}))$, where $nnz(\mathbf{O}) \le (RD_{max} \times M)$; the rest are vector additions and element-wise updates in $(.)_+$ which are done with cost $\mathcal{O}(N)$.
Thus, the final complexity is:

$$\mathcal{O}(b^2N + NlogN)$$

The dominant term in the above final cost depends on whatever is larger, $logN$ or b^2, which depends on the maximum report duration RD_{max}. □

In addition to reducing the running time, the resulting algorithmic steps above are very efficient in terms of memory consumption and can handle very large amounts of data as we demonstrate in Section 5.3.

4. EXPERIMENTAL DESIGN

In this section, we explain the set up of the experiments performed to evaluate the proposed method. Data sets are explained in Section 4.1, the baselines and metrics are listed in Section 4.2, and Section 4.3 contains description of the input settings and configuration.

4.1 Data Sets

In order to study the performance of HOMERUN, we apply it to the data from project Tycho [27], which includes real epidemiological time sequences spanning over more than 100 years. We select the data about measles in NYC to be our main data set for analyzing and evaluating the performance of the three proposed methods. Furthermore, since the effectiveness of the proposed

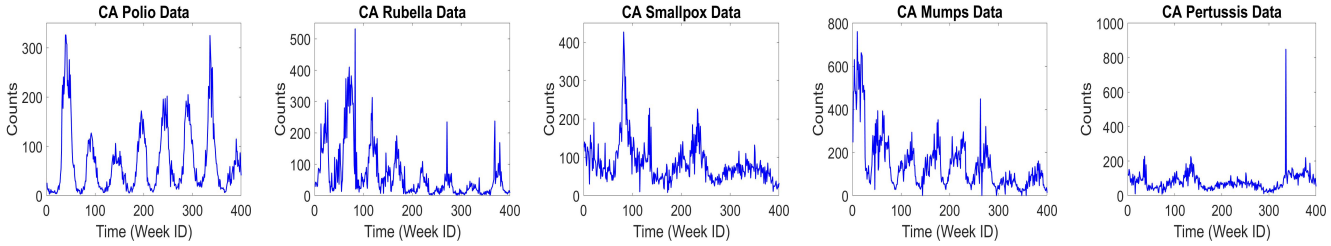


Figure 7: Disease Time Sequences

method hinges upon the degree of energy compaction of the DCT of the data (refer to Section 2.2), we explore the performance using six more data sets with different behavior. We pick the intervals with the *least missing values* – the particular weeks used for testing for each data set are NYC measles (from Week 51 to Week 450), CA polio from (1659 – 2058), CA rubella (2805 – 3204), CA smallpox(501 – 900), CA mumps (2756 – 3155), CA hepatitis (2653 – 3051), CA pertussis (1649 – 2048). The behavior of the counts of each disease across these weeks is shown in Figure 7 (NYC measles and CA hepatitis are shown earlier in Figures 3 and 4). We can see here that each disease has notably different dynamic, and, as we observed, they have different DCT with different degree of sparsity and energy compaction, which provide us with a rich test to evaluate the performance of our method.

4.2 Baselines and Evaluation Metrics

We compare the performance of our method against two baselines, LS in (12) and H-FUSE [21], focusing on H-FUSE since it is a state-of-the-art for this problem and has better performance than LS. H-FUSE infuses domain knowledge to improve the reconstruction accuracy by penalizing large differences between adjacent timeticks to promote smoothness. Using our notation, H-FUSE can be written as follows:

$$\min_{\mathbf{x}} \quad \|\mathbf{y} - \mathbf{O}\mathbf{x}\|_2^2 + \|\mathbf{H}\mathbf{x}\|_2^2 \tag{28}$$

where $\mathbf{H}$ is the smoothness matrix defined in Section 3.3.

We use the Relative Error Difference (RED) defined below to compare the performance between the proposed and baseline methods.

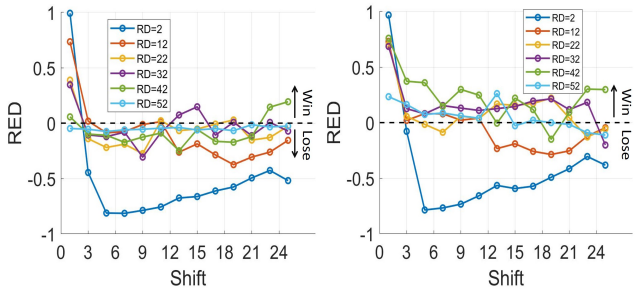
$$RED = \frac{\text{RMSE}(baseline) - \text{RMSE}(proposed)}{\max(\text{RMSE}(baseline), \text{RMSE}(proposed))} \tag{29}$$

We use RED with the result figures in Section 5, ranging between -1 and 1. Clearly, positive RED means that the proposed method improves the baseline and vice versa.

4.3 Input Setting

The task is to reconstruct the weekly reports of each disease sequence. We generate different aggregated reports from the weekly counts. Each report covers multiple successive timeticks. As described earlier in Section 2.1, the number of weeks included in each observation is called *Report Duration* (RD). The difference between the starting week of two successive reports is the *shift*.

For most experiments, we generate reports with the same RD and $shift$ values and show the results on a wide range (RD spans from 2 to 52 weeks (1 year) with increment of 10 and the $shift$ span from 1 to 25 with increment of 2). Specifically, the first report (y_1) includes the weeks from 1 to RD; y_2 includes weeks from ($shift + 1$) to ($shift + 1 + RD$) and so on – refer to the example in Section 2.1. This methodology allows us to study the results for all , e.g., easy cases where RD and $shift$ are both small, more challenging cases where RD or/and $shift$ are large, overlapping reports, and reports with gaps ($RD < shift$). We also show results on experiments where each report covers different number of weeks (different RD) with random starting points, thus they could be overlapped or having gaps.



(a) HOMERUN-0 (b) HOMERUN-N

Figure 8: Comparing HOMERUN-0 and HOMERUN-N versions against the baseline H-FUSE with NYC measles data.

5. RESULTS AND ANALYSIS

In this section, we present and analyze the experimental results in the following order: 1) **effectiveness** of HOMERUN and its early versions (HOMERUN-0 and HOMERUN-N), 2) **discussion** of the observations and findings about the performance of the proposed methods, and 3) **scalability** to demonstrate how HOMERUN scales in terms of running time with data size.

5.1 Effectiveness

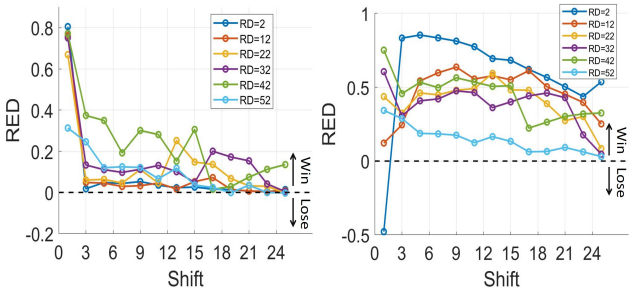
In this section, we show the performance of all three versions of the proposed method in the same order as we explained them in Section 3. Presenting results of the earlier versions of HOMERUN demonstrates the benefit of the added constraints (non-negativity and smoothness).

5.1.1 *Performance of* HOMERUN-0

We compare the reconstruction accuracy of HOMERUN-0 with the baseline H-FUSE using NYC measles data in Figure 8 (a). We can see that HOMERUN-0 improves the baseline significantly for most RD values, but only with $shift$ smaller than 3. For small RD, H-FUSE works better than HOMERUN-0 with a large difference. This is intuitively understandable as if each report covers only few timeticks (i.e, each report is highly localized), then penalizing the large jumps between two adjacent timeticks is good enough to recover the solution sequence. However, we note that HOMERUN-0 loses to H-FUSE with smaller difference with larger RD values (e.g., $RD = 52$). Larger RD means less available reports, hence the problem is more difficult. The performance of

Table 3: RMSE of HOMERUN and the baselines (H-FUSE and LS) using NYC measles data.

	$RD = 2$					$RD = 22$					$RD = 42$				
$shift$	1	7	13	19	25	1	7	13	19	25	1	7	13	19	25
HOMERUN	20.30	**205.96**	**415.79**	**609.03**	**644.38**	**42.49**	**191.10**	**227.03**	**400.07**	894.11	**57.28**	**186.79**	**305.27**	**565.85**	**514.48**
H-FUSE	104.23	215.38	425.56	617.27	653.45	128.38	200.22	303.56	429.02	**891.49**	251.27	231.20	359.86	582.33	595.05
LS	**10.60**	1242.5	1358.4	1402.7	1391.5	75.45	346.49	559.55	654.32	976.30	230.32	371.29	617.42	768.31	762.65



(a) Against H-FUSE (b) Against LS

Figure 9: HOMERUN wins. Performance of HOMERUN versus the baselines H-FUSE and LS using NYC measles data.

this basic version needs to be improved, which led to deriving the more advanced versions.

5.1.2 *Performance of* HOMERUN-N

In this approach, we leverage the knowledge that the patient counts in the target sequence is always non-negative. A comparison between HOMERUN-N and the baseline H-FUSE usin g NYC measles data is shown in Figure 8 (b). Adding non-negativity constraint to HOMERUN-0 improves the accuracy significantly as it is clear from comparing Fig. 8 (a) and (b). HOMERUN-N makes a significant improvement over the baseline for the majority of cases with the following remarks. For $RD = 22, 32, 42, 52$, HOMERUN-N outperforms the baseline except for few outliers. It is therefore clear that more improvement occurs with larger RD values. HOMERUN-N has similar behavior to HOMERUN-0 in the sense that very large improvement happens with small $shift$ for all durations. When $RD = 2$ or 12 and the $shift$ is larger than RD, HOMERUN-N does not improve the baseline, this is consistent with results of HOMERUN-0. Although HOMERUN-N gives encouraging results, we develop the final version of HOMERUN seeking a consistent improvement.

5.1.3 *Performance of* HOMERUN

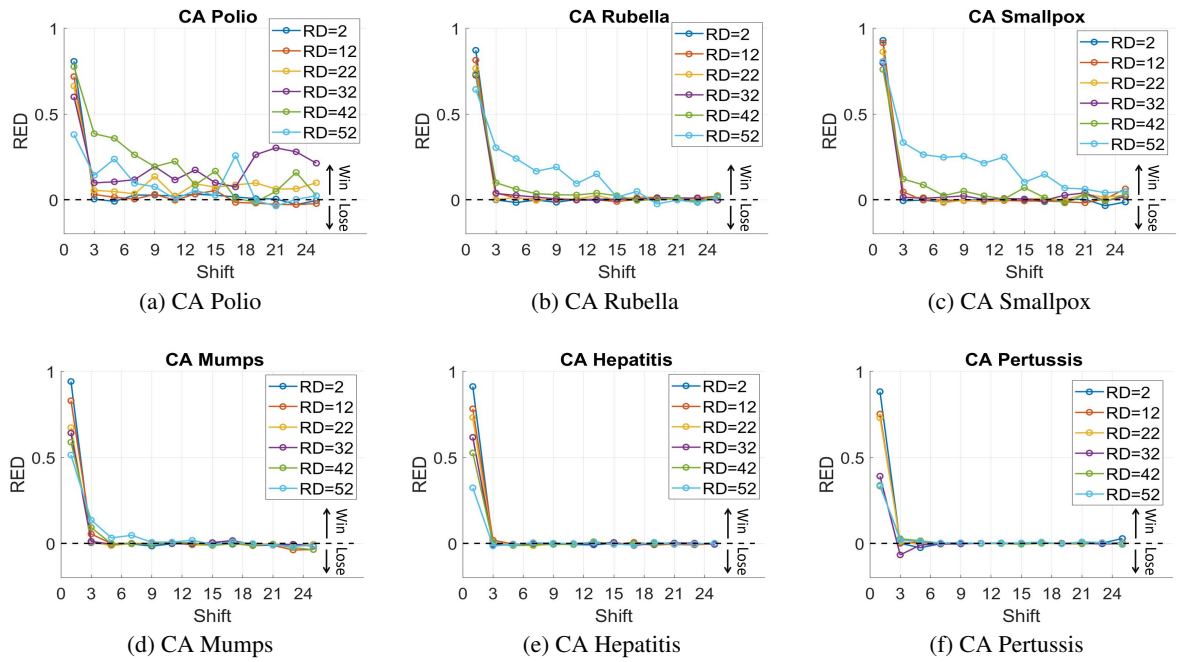
In this section, we show the performance of the final version of the proposed method (HOMERUN) using NYC measles *and* the other six data sets described in Section 4.1. This model reaps the benefits of both the proposed method and the smooth reconstruction, thus it improves the estimation error of its earlier versions and considerably outperforms the competing baseline methods. Figure 9 (a) shows comparison between HOMERUN and the baseline H-FUSE with NYC measles data, HOMERUN is always superior with significant improvement. Note here that generally speaking, greater improvement happens with larger RD, which makes the problem more difficult as we have fewer observations (reports). Figure 9 (b) compares HOMERUN with the baseline LS, HOMERUN vastly outperforms LS at all the input settings (RD and $shift$) except for one. The reason of LS working better in this scenario is because each report covers only two weeks ($RD = 2$) and each variable (week), x_n, is involved in two reports since $shift = 1$ (except for the first and last weeks), resulting in an easy problem for LS solution since $\mathbf{O}$ is "almost" square and full rank. To give the reader an idea about the RED versus the actual RMSE of H-FUSE and the baselines (H-FUSE and LS), we present Table 3, showing the RMSE at various RD and $shift$ values (a subset of the input settings in Fig. 9 (a) and (b), but similar pattern with other RD values) – again, note that the improvement is higher as RD increases.

We compare HOMERUN with H-FUSE (since it is the best baseline) using the other six data sets in Figure 10. Polio data set has similar results to NYC measles because their time series have similar shapes, thus are similar in the DCT domain. HOMERUN significantly improves the baseline when RD is large with rubella and smallpox data sets. It also shows a large improvement with small $shift$ for all durations with rubella and smallpox. For data sets that do not have very smooth time sequence (mumps, hepatitis, and pertussis), HOMERUN improves the baseline significantly with $shift$ smaller than 3 and performs similar to it when the $shift$ is larger. Table 4 highlights the results comparing HOMERUN with the baseline H-FUSE in Fig. 9 (a) and Fig. 10. It shows the maximum and average improvement across the different input settings (RD and $shift$ values) with all the different data sets using RED percentage ($RED(\%)$). As we observe, the improvement is considerable and may be up to 94%.

Table 4: HOMERUN wins consistently. Comparing Performance of HOMERUN and H-FUSE using $RED(\%)$.

	RED (%)	
Data Name	Maximum	Average
NYC Measle	80.52	13.14
CA Polio	80.77	12.56
CA Rubella	87.20	8.19
CA Smallpox	93.04	9.90
CA Mumps	94.14	5.45
CA Hepatitis	91.25	4.73
CA Pertussis	88.24	4.43

So far, each single point in the result plots (or Table 3) corresponds to solving the problem given aggregated reports that have the same duration/resolution (i.e., same RD). In Figure 11, we compare the RMSE of HOMERUN and baselines in recovering the weekly counts of NYC measles data, given M reports with RD values randomly drawn from the range $(2 - 26)$ and the starting week of each report is also random. We test the performance across different M values ranging from 20 to 380 with increment of 10. Since the reports are drawn at random, we repeat each experiment 20 times and take the average RMSE with each data set at a given M. HOMERUN is always better than the baselines with $13\% - 23\%$ improvement depending on the data.



(a) CA Polio (b) CA Rubella (c) CA Smallpox

(d) CA Mumps (e) CA Hepatitis (f) CA Pertussis

Figure 10: HOMERUN consistently wins. Performance of HOMERUN vs. the baseline H-FUSE with different data sets (positive=win and negative=lose).

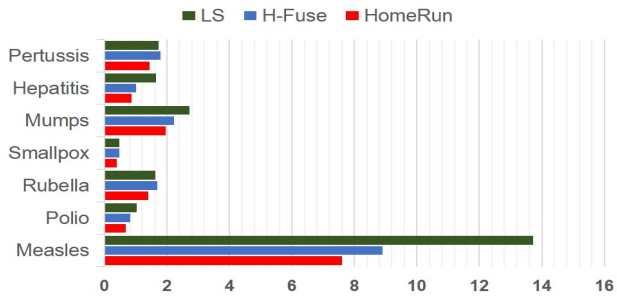


Figure 11: HOMERUN wins. RMSE of HOMERUN compared to H-FUSE and LS with reports having different RD.

5.2 Discussion and Observations

In this section, we provide a higher level discussion about the results, and a detailed analysis of the performance with various input settings (RD and $shift$). We also provide observations about the scope of applicability that can give practitioners insights on when to expect good reconstruction using the proposed method.

Quality of the disaggregation methods in general is affected by the report configurations. Note that as the RD and $shift$ increase, the number of available reports (or equations in the linear system $\mathbf{Ox} = \mathbf{y}$) decreases, resulting in a harder problem. In general, HOMERUN has greater improvement over the baseline H-FUSE as the RD increases as can be observed in Fig. 9 (a), Table 3, and top three data in Fig. 10. Nevertheless, more reports/equations in the system will constraint the solution of $\mathbf{s}$, bringing it closer to the actual DCT coefficients of the target sequence in the proposed method since the optimization problem is constrained by the linear system (Eq. (21)). This is also the case for the baselines (H-FUSE and LS), as the number of equations increases, the LS solution becomes closer to the true sequence. We also have the smoothness penalty in HOMERUN and H-FUSE to help bringing the solution closer to the true sequence *if* the sequence is in fact smooth. In both models, as the number of equations decreases (i.e., large $shift$ and/or RD), the degree of freedom of $\|\mathbf{s}\|_1$ in HOMERUN and $\|\mathbf{y} - \mathbf{Ox}\|_2^2$ in H-FUSE increases, and thus the quality of the solution relies more on the smoothness penalty. This is especially notable when $shift$ is larger than RD, resulting in gaps between reports which leaves some variables x_n out of the constraints. In that case, the solutions produced by HOMERUN and H-FUSE converge, justifying the similar performance with large $shift$ especially in the bottom three data in Fig. 10.

For more analysis, we show the performance of HOMERUN using NYC measles data set with RD spanning from 2 to 52 with increments of 2, and $shift$ ranging from 1 to 25 with increments of 2. In order to explain the comparison more clearly, we map the RED value (Eq. (29)) to the logical value $RED_{logical}$ as follows:

$$RED_{logical} = \begin{cases} 1 & RED > threshold \\ -1 & RED < -threshold \\ 0 & else \end{cases} \quad (30)$$

where we empirically set the threshold to 0.015. In Figure 12, we show the $RED_{logical}$ across the different input settings. The yellow line shows when $x = y$, which separates reports with overlaps (above the line), i.e., *shift* $<$ *RD*, from reports with gaps (below the line), i.e., *shift* $>$ *RD*. Blue color means HOMERUN improves the baseline H-FUSE, light gray means they perform similarly, and red means the baseline works better. Figure 12 shows that HOMERUN never loses to the baseline and the improvement happens in *almost* all the cells in the upper area (overlapped reports), while it is not guaranteed in the lower part. This is because when we have large gaps between the available reports, reconstructing a higher resolution sequence using

smoothness constraint may give the smallest error. One of the advantages of HOMERUN is that it reaps the benefits of its own and the baseline H-FUSE.

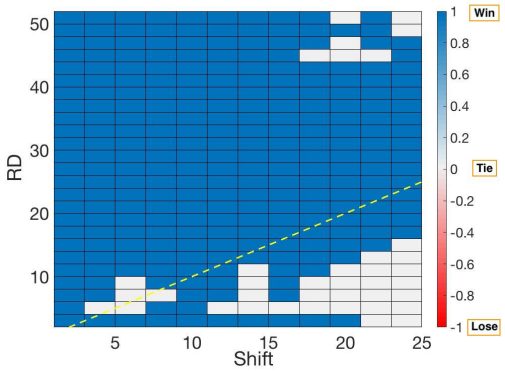


Figure 12: HOMERUN almost always wins and sometimes ties. HOMERUN performance against the baseline H-FUSE using NYC measles data.

5.2.1 *Applicability of* HOMERUN

Quasi-periodicity: if the target sequence is known to have few dominant periodicities, i.e., quasi-periodic (e.g., measles, and polio), then very few cosine functions are needed to approximate it. In other words, this sequence can be accurately represented by few DCT coefficients, i.e., its DCT is very compact and the sequence is very sparse in the DCT domain, thus HOMERUN achieves especially good reconstruction.

The sparser the spectrum of the data, the better the performance of HOMERUN. This is because we optimize for the sparsest spectrum representation of the sequence using $L1$ norm. Empirically, this can be observed by comparing the performance of HOMERUN when applied on data sets with different periodicity behavior in the time domain. When the data set is quasi-periodic, then HOMERUN improves the baseline H-FUSE significantly (e.g., measles in Fig. 9 (a) and polio in Fig. 10 (a)). With the less periodic sequences (e.g., hepatitis in Fig. 10 (e)), the accuracy of HOMERUN drops in comparison to its performance with quasi-periodic sequences, however, it still has a better performance than the baseline H-FUSE (significant improvement with small $shift$).

Smoothness: smoothness penalty has been shown to be effective for time sequences in many applications (e.g., epidemiological data from the Tycho project). Since HOMERUN includes smoothness term, its accuracy increases with data that exhibits higher degree of smoothness. Similarly, this can be seen by comparing the smoothness of measles data (Fig. 3) and hepatitis (Fig. 4) in their time domain, and their performance (Fig. 9 (a) and 10 (e)), respectively.

Non-negativity: moreover, if the data is non-negative in its nature (e.g., Tycho data set used in this work), then adding the non-negativity constraint improves the solution and reduces the error, as is clear from comparing the performance of HOMERUN-0 and HOMERUN-N (Fig. 8 (a) and (b), respectively).

It is important to point out that the smoothness and non-negativity assumptions are flexible in the proposed framework. For instant, if the target sequence in another application is quasi-periodic but not smooth (e.g., climate data), then HOMERUN-0 (or HOMERUN-N if the sequence is non-negative) can be applied and is expected to perform well.

5.3 Scalability

Experiments were performed using Matlab on a Linux server with an Intel Core i7 $-$ 4790 CPU 3.60 GHz processor and 32 GB memory. The accelerated implementation of HOMERUN following Algorithm 3 scales linearly with the length of the time sequence (see Figure 1 (c)). As clear from the comparison in Figure 1 (c), HOMERUN is always faster than H-FUSE. The simple LS method also gets slower than HOMERUN as the sequence size increases, this is due to the pseudo-inverse step in both LS and H-FUSE. We should point out here that the accelerated implementation of HOMERUN dramatically reduces the running time of the direct implementation, while keeping the same accuracy (Algorithm 2 versus Algorithm 3). Moreover, HOMERUN is very efficient in terms of memory and can handle very large sequences.

6. CONCLUSIONS

In this paper, we proposed HOMERUN, a novel algorithm for solving historical data disaggregation problem via DCT-based sparse reconstruction with non-negativity and smoothness constraints. We leverage the ADMM algorithm to solve the resulting optimization problem. We demonstrated that HOMERUN outperforms the baseline methods with real data from the Tycho project. The contributions of this paper are summarized as follows:

1. **Formulation and Algorithm**: we formulate the data disaggregation problem in the form of Basis Pursuit, add domain knowledge constraints (non-negative and smoothness), and derive the steps of the ADMM algorithm that solve HOMERUN optimization problem.
2. **Effectiveness**: HOMERUN improves the competing baselines and recovers the time sequences with up to 94% improvement in the accuracy of the baseline methods.
3. **Scalability**: We derive accelerated steps of HOMERUN, which scale well and have a complexity of $\mathcal{O}((2RD_{max} - 1)^2N + NlogN)$.
4. **Adaptability**: HOMERUN is parameter-free, and it adapts to the input signal, i.e., it automatically detects the prominent periodicities in the data without the need of assuming any known periodicity.

Reproducibility: The Tycho dataset is publicly available [27]; We plan to release the code upon publication of the paper.

7. ACKNOWLEDGE

This work was supported in part by the National Science Foundation under grants IIS-1247489, BCS-1244672, and IIS-1447788, and by the Army Research Laboratory under Cooperative Agreement Number W911NF-09-2-0053.

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation, or other funding parties. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

8. REFERENCES

[1] N. Ahmed, T. Natarajan, and K. R. Rao. Discrete cosine transform. *IEEE transactions on Computers*, 100(1):90–93, 1974.

[2] E. Amaldi and V. Kann. The complexity and approximability of finding maximum feasible subsystems of linear relations. *Theoretical computer science*, 147(1-2):181–210, 1995.

[3] J. Bleiholder and F. Naumann. Data fusion. *ACM Computing Surveys (CSUR)*, 41(1):1–41, Jan. 2009.

[4] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

[5] E. J. Candès. Compressive sampling. In *Proceedings of the international congress of mathematicians*, volume 3, pages 1433–1452. Madrid, Spain, 2006.

[6] E. J. Candes, J. K. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on pure and applied mathematics*, 59(8):1207–1223, 2006.

[7] E. J. Candès and M. B. Wakin. An introduction to compressive sampling. *IEEE signal processing magazine*, 25(2):21–30, 2008.

[8] S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM review*, 43(1):129–159, 2001.

[9] E. Cohen and H. Kaplan. Bottom-k sketches: Better and more efficient estimation of aggregates. In *ACM SIGMETRICS Performance Evaluation Review*, volume 35, pages 353–354. ACM, 2007.

[10] G. Cormode, M. Garofalakis, P. J. Haas, and C. Jermaine. Synopses for massive data: Samples, histograms, wavelets, sketches. *Foundations and Trends in Databases*, 4(1–3):1–294, 2012.

[11] H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. Keogh. Querying and mining of time series data: experimental comparison of representations and distance measures. *PVLDB*, 1(2):1542–1552, 2008.

[12] X. L. Dong and F. Naumann. Data fusion: resolving data conflicts for integration. *PVLDB*, 2(2):1654–1655, 2009.

[13] D. L. Donoho and M. Elad. Optimally sparse representation in general (nonorthogonal) dictionaries via $l1$ minimization. *Proceedings of the National Academy of Sciences*, 100(5):2197–2202, 2003.

[14] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image processing*, 15(12):3736–3745, 2006.

[15] C. Faloutsos, H. V. Jagadish, and N. Sidiropoulos. Recovering information from summary data. *PVLDB*, 1(1):36–45, 1997.

[16] O. G. Guleryuz. Weighted overcomplete denoising. In *Signals, Systems and Computers, 2004. Conference Record of the Thirty-Seventh Asilomar Conference on*, volume 2, pages 1992–1996. IEEE, 2003.

[17] M.-J. Hsieh, W.-G. Teng, M.-S. Chen, and P. S. Yu. Dawn: an efficient framework of dct for data with error estimation. *The VLDB Journal—The International Journal on Very Large Data Bases*, 17(4):683–702, 2008.

[18] K. Huang, N. D. Sidiropoulos, and A. P. Liavas. A flexible and efficient algorithmic framework for constrained matrix and tensor factorization. *IEEE Transactions on Signal Processing*, 64(19):5052–5065, 2016.

[19] S. A. Khayam. The discrete cosine transform (dct): Theory and application. department of electrical and computing engineering, 2003.

[20] J.-H. Lee, D.-H. Kim, and C.-W. Chung. Multi-dimensional selectivity estimation using compressed histogram information. In *ACM SIGMOD Record*, volume 28, pages 205–214. ACM, 1999.

[21] Z. Liu, H. A. Song, V. Zadorozhny, C. Faloutsos, and N. Sidiropoulos. H-fuse: Efficient fusion of aggregated historical data. In *Proceedings of the 2017 SIAM International Conference on Data Mining*, pages 786–794, Houston, Texas, USA, April 2017.

[22] A. V. Oppenheim and R. W. Schafer. *Discrete-Time Signal Processing*. Prentice Hall Press, Upper Saddle River, NJ, USA, 3rd edition, 2009.

[23] A. Panagiotopoulou and V. Anastassopoulos. Super-resolution image reconstruction techniques: Trade-offs between the data-fidelity and regularization terms. *Information Fusion*, 13(3):185–195, 2012.

[24] T. Rekatsinas, M. Joglekar, H. Garcia-Molina, A. Parameswaran, and C. Ré. Slimfast: Guaranteed results for data fusion and source reliability. In *Proceedings of the 2017 ACM International Conference on Management of Data*, pages 1399–1414. ACM, 2017.

[25] S. Saha. Image compression – from dct to wavelets: A review. *Crossroads*, 6(3):12–21, Mar. 2000.

[26] C. Sax and P. Steiner. Temporal disaggregation of time series. *The R Journal*, 5(2):80–87, 2003.

[27] Tycho. Project tycho: Data for health. `https://www.tycho.pitt.edu`, 2013.

[28] A. B. Watson. Image compression using the discrete cosine transform. *Mathematica journal*, 4(1):81, 1994.

[29] V. Zadorozhny and M. Lewis. Information fusion for usar operations based on crowdsourcing. In *Information Fusion (FUSION), 2013 16th International Conference on*, pages 1450–1457. IEEE, 2013.