Urban mobility sensing for traffic using sparse processing

QI LI
Abstract

This thesis was aimed at studying the existing methods for origin-destination (OD) estimation problem and developing a new algorithm which provides higher promise.

The performance was evaluated on a simulated data-set for Stockholm city. Data for this study were obtained with the help of G. Flötteröd from Department of Transport Science in KTH. Information minimizing approach and entropy maximizing approach, which are the state-of-art methods in transport field were modified to implement. Several existing algorithms in signal processing field, such as BP/BPDN, OMP and SP, were implemented and analyzed. A recently proposed algorithm called OMPₚ was described. Then a more effective method SPₚ with better reconstruction performance in sparse signal processing area was proposed in this report.

By numerical experiments, it was concluded that the methods in signal processing field could deal with OD estimation problem well. Hopefully this thesis could make a contribution to opening the door to another field and introducing methods of that universe, as well as developing a new algorithm with robust results and small computation cost.

Keywords: OD estimation, compressive sensing, sparsity, information minimizing, entropy maximizing, convex relaxation, orthogonal matching pursuit, subspace pursuit, OMP₊, SP₊
## Contents

1 Introduction ........................................... 3  
   1.1 Introduction ........................................ 3  
   1.2 Problem statement ................................. 4  
   1.3 Outline ............................................. 4  

2 Background ............................................ 6  
   2.1 Methods ............................................. 6  
   2.2 RIP .................................................. 7  

3 System setup and sparsity ............................. 9  
   3.1 System setup for OD estimation ................. 9  
   3.2 Role of Sparsity .................................. 10  

4 Simulated data for Stockholm .......................... 12  
   4.1 Magnitude of the simulated data ............... 12  
   4.2 Generation of the OD vector $\mathbf{x}$ ........ 12  
   4.3 Generation of the measurements vector $\mathbf{y}$ 13  
   4.4 Acquisition of the system matrix $\mathbf{A}$ .... 13  
   4.5 Simulation of noise $\mathbf{w}$ ................... 13  

5 Algorithms ............................................. 15  
   5.1 Sparse-adapted Information Minimising ......... 15  
   5.2 Sparse-adapted Entropy Maximising ............. 18  
   5.3 BP and BPDN ...................................... 20  
      5.3.1 Algorithms of BP and BPDN ................. 20  
      5.3.2 Norm definitions .......................... 22  
      5.3.3 Choice of $l_1$ norm ....................... 23  
   5.4 OMP .............................................. 24  
   5.5 SP ............................................... 26  
   5.6 OMP$_+$ ........................................... 28  
   5.7 SP$_+$ ............................................. 28
6 Numerical experiments  
6.1 Performance measures ........................................ 34  
6.1.1 SRNR ......................................................... 34  
6.1.2 ASCE ......................................................... 34  
6.1.3 RMSE ......................................................... 35  
6.2 Experiments on artificial data .............................. 35  
6.3 Implementation on a preliminary simulation data ...... 39  
6.4 Implementation on realistic simulated data ............. 41  
6.4.1 Implementation with OMP and OMP+ ................. 44  
6.4.2 Validation of SP and SP+ ................................. 44  
6.4.3 Implementation on SIM ................................... 48  
6.4.4 Implementation on SEM ................................... 55  

7 Discussion and Conclusion  
7.1 Discussion and Conclusion .................................. 60  
7.2 Future Work ................................................. 62  

Bibliography  

Chapter 1

Introduction

In this chapter, the importance of the project is discussed and the problem is briefly introduced. Section 1.3 shows the structure of this dissertation.

1.1 Introduction

Urban sensing is very important for urban modeling. The methods for urban sensing are still under developing and there is no perfect solution so far. The transportation sector plays an important role in global sustainability. In 2011, it accounts for 20% of total energy use all over the world. Understanding the urban individual mobility is important for policy makers to consider the environmental consequences of different development scenarios, and develop regional growth strategies towards a more sustainable future [4].

The traditional method to get an origin-destination (O-D) trip matrix is normally implemented by lots of surveys. It is impossible to get a full scale measurement due to high costs of time and human resources. So people are developing a series of methods and models to obtain the O-D matrices from the traffic. The origin-destination (O-D) matrix estimation problem aims at estimating an OD matrix. This OD matrix contains the information of flows from all the source nodes (origins) to all the sink nodes (destinations) in a transport network. The data available for the OD matrix estimation problem consists of noisy observations of arc flows in this transport network. There are variations of several existing standard statistical methods that have been tried out, which are based on information minimization approach and entropy maximization approach [29], Bayesian estimation [19], generalized least squares [1] [2] [9], and maximum likelihood estimation [25]. A time dimension into this OD matrix estimation problem was introduced in [10].

However, we are still on the way and far from solving the OD matrix estimation problem. The under-determined property results in the key difficulty of this problem. Usually it includes a prior OD matrix which is
collected from, for example, an outdated survey and has a supplementary set of measurements. It is widely acknowledged that this method is arbitrary and inaccurate.

The inherent sparsity of an OD matrix estimation problem is observed from a point of view in signal processing field. It means only a small number of high value elements are contained in the OD estimation problem and the other elements which take a majority are very small or zero. A single individual faces a myriad of possible mobility plans to implement at any point in time, where the mobility plan means a sequence of route/mode/time-annotated trips that connect activity locations [3]. Even plenty of choices exist, only one mobility plan can be chosen actually. The superposition of the individual OD mobility plans constitute the mobility patterns. The sparsity is preserved in these mobility patterns.

Recently, sparse signal processing [17], such as compressed sensing [16] [5], which has achieved key breakthroughs, has drawn significant attention. Existing sparse signal processing algorithms and theoretical results, however, cannot be directly applicable to an OD estimation problem. This is because in a setup of urban mobility sensing, the involved system is large scaled and has a complex structure. This thesis proposes and demonstrates a method for predicting the OD matrix with a high promised performance. The performance of the prediction is reflected in the reconstruction accuracy and the robustness under unexpected disruptions. The new algorithm has an advantage in the scenario of running time as well.

1.2 Problem statement

Various models have been applied for the OD matrix estimation problem. To obtain the mobility prediction, we need a certain number of measurements as well as a system matrix that describes the correlation between the observations and the OD mobility.

In this thesis, two algorithms, sparse-adapted information minimising (SIM) and sparse-adapted entropy maximising (SEM) are firstly implemented. We apply orthogonal matching pursuit (OMP) [26], subspace pursuit (SP) [14] to predict the OD matrix afterward. And then two developed greedy algorithms based on OMP and SP are described. Performance analysis is conducted for algorithms analyzing.

1.3 Outline

This thesis is structured as following: The background about the OD trip matrix study is firstly discussed in Chapter 2. Chapter 3 describes the system setup for the OD estimation problem. Chapter 4 interprets the simulated data for Stockholm. Five typical algorithms which could be used
in this area are described in Chapter 5. Chapter 6 presents the experiments which are based on three different data sets. Chapter 7 gives the discussion and conclusion of the thesis.
Chapter 2
Background

To help the readers have a better understanding of the dissertation, some necessary background information is given in this chapter. Section 2.1 firstly shows a description of the representative models for the OD estimation problem, and then discusses the existing algorithms in the signal processing area. Section 2.2 introduces an important concept in compressive sensing.

2.1 Methods

In 1978, Van Zuylen proposed an information minimising approach to estimate the trip matrix. It chooses the OD matrix that adds the least information by calculating the information contained in the trip matrix. Similarly, Willumsen addressed the OD estimation problem by an entropy maximising approach. It is solved by finding the trip matrix which makes the entropy maximized [29]. Both of these two approaches have a multi-proportional problem that could be modified recurrently to make the modelled flows approximately equal to the observations [18].

However, those methods are designed for dense problem. In realistic case it is quite common to have a large-scale sparse data set to solve, so some practical changes are required to be developed to implement. Besides, the operation time of the old methods is quite annoying, which makes it essential to look for a state-of-art algorithm out of the old methods.

There are a lot of methods existing for a linear under-determined system model-based sparse signal estimation problem. The algorithms could be categorized into three classes, convex relaxation, Bayesian inference and iterative greedy search (IGS), which can be shown in Figure 2.1.

Basis pursuit (BP)/ basis pursuit denoising (BPDN) is a convex relaxation algorithm based on $l_1$ norm minimization [5] [13]. Belief propagation is a Bayesian inference algorithm, which aims at solving inference problems [28]. Thanks to the algorithmic simplicity and lower computation complexity, the iterative greedy search algorithms attracts more and more

Figure 2.1: Methods for CS recovery

interest today. Those generated algorithms could be classified into two categories: serial atom selection and parallel atom selection. In the serial class, orthogonal matching pursuit (OMP) [15] [22] and orthogonal least square (OLS) [12] [23] [20] select one atom in each iteration and increase the support set one by one. While in the parallel class, subspace pursuit (SP) [14] and compressive sampling matching pursuit (CoSaMP) [21] select a predetermined number of atoms in every each iteration.

2.2 RIP

We would like to describe an important concept in CS theory called restricted isometry property (RIP), which is introduced by Emmanuel Candès and Terence Tao in 2005 [6]. It is defined as that for a $m \times n$ matrix $A$, if there exists an isometry constant $\delta_S \in (0, 1)$ for any $S \in [1, n]$ such that

\[
(1 - \delta_S)\|x\|^2 \leq \|Ax\|^2 \leq (1 + \delta_S)\|x\|^2, \tag{2.1}
\]

we could say matrix $A$ satisfies RIP of $S$ order.

**Theorem 1.** Suppose that the $m \times n$ measurement matrix $A$ is such that every set of $2S$ columns of $A$ are linearly independent. Then an $S$-sparse vector $x \in \mathbb{C}^n$ can be reconstructed uniquely from $Ax \in \mathbb{C}^m$. 
A strengthened condition from Theorem 1 was derived in Theorem 2.

**Theorem 2** (Candès-T.2005). *Suppose that the matrix $A$ obeys the restricted isometry property (RIP): every collection of $4S$ columns are almost orthogonal, in that the top $4S$ singular values range between 0.9 and 1.1. Then any given $S$-sparse signal $x$ can be recovered from $Ax$ by basis pursuit.*
Chapter 3

System setup and sparsity

From the previous two chapters, the readers may have a basic understanding of the project. This chapter will show how the system is setup in Section 3.1 and discuss the role of sparsity in Section 3.2.

3.1 System setup for OD estimation

The origin/destination (OD) matrix, as shown in Table 3.1 contains the information of the origins, destinations and the traffic flow between each OD pair.

\begin{table}[h!]
\centering
\begin{tabular}{|c|c|c|c|c|}
\hline
Origin/Destination & 1 & \ldots & i & \ldots & n \\
\hline
1 & $T_{11}$ & \ldots & $T_{1i}$ & \ldots & $T_{1n}$ \\
\hline
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\
\hline
j & $T_{j1}$ & \ldots & $T_{ji}$ & \ldots & $T_{jn}$ \\
\hline
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\
\hline
m & $T_{m1}$ & \ldots & $T_{mi}$ & \ldots & $T_{mn}$ \\
\hline
\end{tabular}
\caption{OD matrix}
\end{table}

An OD matrix estimation problem aims at predicting the traffic flow between each OD pair using the observations and a known system matrix. So the system setup is in the format as

\[ y = Ax + w \in \mathbb{R}^m, \quad (3.1) \]

where \( y \in \mathbb{R}^m \) is a measurement vector representing the arc flows, \( A \in \mathbb{R}^{m \times n} \) is the system matrix (usually called measurement matrix or assignment matrix), \( x \in \mathbb{R}^n \) a vector-formed OD matrix, and \( w \in \mathbb{R}^m \) represents the noise (or called error), which includes both the measurement error and some unexplained deviations between model predictions and observations.
Counting processes, such as the number of pedestrians or vehicles during some time at some place, are comprised in \( y \). \( x \) here is unknown, and it comprises the occurrence of aggregate mobility patterns in the population. As \( x \) is in vector form and the OD matrix in a transport field is usually in matrix form as in Table 3.1, a look-up table between \( x \) and \( T \) can be found according to some rule.

We are interested in estimating the mobility patterns \( x \) from a limited set of observations (i.e. individuals and vehicles) \( y \) and from this predicting the spatio-temporal evolution of the city-wide presence of individuals and vehicles.

### 3.2 Role of Sparsity

In ideal case, the problem (3.1) could be written as

\[
y = Ax \in \mathbb{R}^m,
\]

which is a classical problem in linear algebra. The problem (3.1) is a practical form with noise added and thus the aim becomes getting the prediction with mitigating the impact of the unexpected disruptions.

In this kind of problem, \( x \in \mathbb{R}^n \) is a n-dimensional unknown vector, \( y \in \mathbb{R}^m \) is a vector of the measurements with \( m \) dimensions, and the system matrix \( A \) is known with the dimension of \( m \times n \). According to the relationship between \( m \) and \( n \), it is categorized as a determined (when \( m = n \)), overdetermined (when \( m > n \)) and under-determined problem (when \( m < n \)).

It can be easily worked out when the problem is in a determined case or an overdetermined case. But in the realistic case of OD estimation problem, the number of measurements of arcs are quite limited compared with the number of prediction needed, that is, \( m \) is far less than \( n \) (\( m \ll n \)). So the problem (3.1) is under-determined and has no conclusive (unique) mathematical solution. As we have discussed before that the mobility patterns \( x \) is sparse, we divert our attention to the area of sparse signal process, such as compressed sensing [16] [5], which has recently shown a possibility to get good solution to the problem (3.1).

As a comparison, both the old OD estimation methods in transport field and the existing sparse signal processing approach will be tested in this report by recovering a synthetic OD matrix for the city of Stockholm from simulated link flows. A newly developed algorithm OMP\(_+\) that proposed by S. Chatterjee is implemented after that [11]. Then a greedy algorithm SP\(_+\) in signal processing field is proposed and implemented to show the improvements. The synthetic OD matrix is derived from an activity-based travel demand model, resulting in a realistic (sparsity) structure. We will use
some existing computationally efficient algorithms from the area of sparse signal processing for the experiments reported in the thesis.
Chapter 4

Simulated data for Stockholm

This chapter introduces the simulated data for Stockholm city, including explanation of the variables and the acquisition of the system matrix.

4.1 Magnitude of the simulated data

This simulated data is generated by a detailed simulation model system of a morning rush hour in Stockholm city. The greater Stockholm region is divided into approximately $10^3$ zones. The aggregated of the paths connecting two zones constitute the road network, which has about $10^4$ links. About $10^5$ vehicle trips are simulated.

4.2 Generation of the OD vector $x$

Each path that starts from one zone (origin $i$) and ends to another zone (destination $j$) is called an OD pair and denoted by $x_{ij}$. Thus the OD vector $x$ can be defined as

$$x = \begin{pmatrix} x_1 \\ \vdots \\ x_{ij} \\ \vdots \\ x_n \end{pmatrix}.$$  \hspace{1cm} (4.1)

Based on realistic assumptions about the route choice and traffic flow propagation, the simulated trips are taken to fill the $x$ vector.
4.3 Generation of the measurements vector $y$

The vector of measurements $y$ comprises information of the links and defined as

$$ y = \begin{pmatrix} y_1 \\ \vdots \\ y_k \\ \vdots \\ y_m \end{pmatrix}. \quad (4.2) $$

$y_k$ is used to represent one single link $k$. This $y_k$ records the total traffic flow on link $k$. It obviously follows the rule that

$$ y_k = \sum_{ij} n_{ij}^k \quad (4.3) $$

$$ = \sum_{ij} \frac{n_{ij}^k}{x_{ij}} x_{ij}. \quad (4.4) $$

Here $x_{ij}$ is the number of vehicles traveling from origin $i$ to destination $j$, and $n_{ij}^k$ means the quantity of vehicles that both travels on OD trip $x_{ij}$ and uses link $k$.

4.4 Acquisition of the system matrix $A$

The system matrix $A$ is a concrete matrix and defined by

$$ A = \begin{pmatrix} a_{11}^1 & \cdots & a_{1j}^i & \cdots & a_{1n}^n \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{k1}^1 & \cdots & a_{kj}^i & \cdots & a_{kn}^n \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{m1}^1 & \cdots & a_{mj}^i & \cdots & a_{mn}^n \end{pmatrix}. \quad (4.5) $$

One element $a_{ij}^k$ in $A$ means the probability of an OD trip $x_{ij}$ uses link $k$. Hence the system matrix $A$ could be studied as

$$ a_{ij}^k = \frac{n_{ij}^k}{x_{ij}}. \quad (4.6) $$

4.5 Simulation of noise $w$

There is not any noise existing in an idealized case of this problem. But it is inevitable to deal with a noisy system in practice. The noise term is defined
as

\[ w = \begin{pmatrix} w_1 \\ \vdots \\ w_k \\ \vdots \\ w_m \end{pmatrix}. \]  \hspace{1cm} (4.7)

It is known that all elements of \( y \), \( A \) and \( x \) are non-negative. Besides, the restriction of capacity of the links results in an upper bound of the elements in \( y \). So for an ideal clean case system, the error term \( w \) can be simulated with a mean value at zero and add it to the system. We can truncate the resulting measurement elements \( y = Ax + w \) between zero and the upper bound.
Chapter 5

Algorithms

In this chapter, we will go into the algorithms for the OD estimation problem, including both existing algorithms and a new proposed algorithm. The algorithms are presented with description and explanation of the process. Flow charts are provided to show the methods more clearly. Section 5.1 and Section 5.2 describe two important models in transport planning area. Some modifications are made to apply those two models for our sparse case. A convex relaxation algorithm is introduced in Section 5.3 and two existing sparse signal processing algorithms which are greedy in nature are talked about in Section 5.4 and Section 5.5, respectively. A similar method called OMP$^+$ is shown in Section 5.6 exhibits higher promise than the standard OMP algorithm. In the last part of this chapter, we give the description of a new method called SP$^+$ which is developed based on the standard SP algorithm in Section 5.7.

5.1 Sparse-adapted Information Minimising

There is one representative model in transport analysis field called information minimizing [30], which is designed for a dense problem meaning that the system matrix is full-row-ranked. With modification to that, we conclude this algorithm referred to as the sparse-adapted information minimising approach (SIM) and is adapted to the large-scale sparse case problem.

In a sparse case, most elements of $A$ are zeros. For an all-zero column in the $A$ matrix, it would lead to a never-updated prediction of $x$. Meanwhile, the all-zero column could be also interpreted as the probability that the corresponding vehicle traveling in OD relation can be observed on any link is zero. So we could pick those links out first. The main steps of SIM are shown in Algorithm 1.

In this SIM algorithm, we take $A$, $y$, $\epsilon$ and $n$ as the input. Initially, $k$ is set to be zero as the iteration counter, and $\lambda$ is a $M \times 1$ zero vector. The index of the all-zero columns of $A$ needs to be found out, and we denote
Algorithm 1: Sparse-adapted Information Minimising

Input:
1: \( A, y, \epsilon, n \);

Initialization:
1: Iteration counter \( k \leftarrow 0 \);
2: \( I_0 \leftarrow \) indices of the all-zero columns of \( A \);
3: \( \mathbf{P} \leftarrow A \bar{I}_0 \);
4: \( \lambda \leftarrow 0, \lambda \in \mathbb{R}^M \);
5: \( s \leftarrow \) sum of the rows in \( \mathbf{P} \), \( s \in \mathbb{R}^M \);
6: \( \mathbf{X} \leftarrow y s \cdot e^{-(1+\lambda)} \), \( \mathbf{X} \in \mathbb{R}^M \);
7: \( g \leftarrow \) sum of the columns in \( \mathbf{P} \), \( g \in \mathbb{R}^N \);
8: \( T \leftarrow \prod_{a=1}^{M} X^a \bar{P}^a \), \( T \in \mathbb{R}^N \);

Iterations:
1: \( \text{repeat} \)
2: \( k \leftarrow k + 1 \);
3: \( a \leftarrow 0 \);
4: \( \text{repeat} \)
5: \( a \leftarrow a + 1 \);
6: \( \lambda_a \leftarrow \frac{P^a \cdot T}{y_a} \);
7: \( \alpha \leftarrow \lambda_a - \frac{P^a \cdot g}{s} \);
8: \( T \leftarrow \alpha \cdot T \);
9: \( \text{until} (a == M) \)
10: \( \text{until} \) all \( (|\alpha - 1| < \epsilon) \) or \( (k > n) \)

Output:
1: \( \hat{x} \in \mathbb{R}^N \), satisfying \( \hat{x}_{I_0} = T \) and \( \hat{x}_{I_0} = 0 \).

It is possible to obtain a new matrix \( \mathbf{P} \) by \( A_{I_0} \). A new vector \( s \) with the size of \( M \times 1 \) is obtained by computing the summation of all columns in \( A_{I_0} \), and a \( g \) vector with the size of \( N \times 1 \) is obtained by computing the summation of all rows in \( \mathbf{N} \). We build a new vector \( \mathbf{X} \) and get the value of each element by \( X_i = \frac{y_i}{s_i} \cdot e^{-(1+\lambda_t)} \). A dense format of the OD matrix is recorded in the \( T \) vector, which has the dimension of \( M \times 1 \) and initialized as \( T_i = \prod_{a=1}^{N} \frac{p^a_i}{y_a} \).

Then the first layer of iteration starts as \( k \) has an increment of one and \( a \) is set zero. The second layer of iteration goes through all the \( M \) links. The \( \lambda \) vector is updated by set \( \lambda_a \) equal to \( \frac{P^a \cdot T}{y_a} \) and a \( \alpha \) vector with the size of \( N \times 1 \) is obtained by \( \alpha_j = \lambda_0 \frac{p^a_j}{s_j} \). Thus a new \( T \) vector is acquired by \( T_j = \alpha_j \cdot T_j \). We need to check the amplitudes of \( \alpha \) in all the \( M \) dimensions when the link list is exhausted. If there exists one or more elements of \( \alpha \)
that has the amplitude larger than $1 + \epsilon$ or smaller than $1 - \epsilon$, we go back to step 2. In order to prevent an endless loop, we set $n$ as the upper limit of the loop.

The error, which measures the maximum absolute value between the ratio of the modelled measurements to the actual measurements and one, usually converges monotonously as in Figure 5.1. Here we use the data in section 6.3, and take the fraction of $\alpha = 0.05$ in a clean measurement case. The x-axis shows the iterations and the y-axis shows the absolute value of the error.

![Figure 5.1: Error by SIM of the preliminary data with $\alpha = 0.05$](image)

Sometimes the error converges to the target range $[1 - \epsilon, 1 + \epsilon]$ with an oscillation. Figure 5.2 shows this with the data described in section ??. We take a fraction of $\alpha = 0.05$ of the data and no noise is added. So we set a certain threshold of the error (let us say, 0.05) and only take the monotonous part to cut the running time down with the cost of a larger reconstruction error. It could be noticed that the error is still above 0.05 even after three thousand times of iterations.

When the stopping criteria is satisfied, we can get a $\mathbf{T}$ vector as the OD trip matrix for the dense estimation. And thus the final output of $\hat{\mathbf{x}}$ could be obtained by setting the elements with the index of $\mathcal{I}_0$ equal to $\mathbf{T}$ and all the other elements as zero.

The flow char of this sparse-adapted information minimising algorithm could be seen from Figure 5.3.
Another important model in the transport planning area is entropy maximizing [30]. This method is more effective than the information minimizing model but is also designed for a dense case. So we conclude a similar algorithm from the maximum entropy considerations, which is referred to as the sparse-adapted entropy maximising approach (SEM). It has the same input and output with SIM, and a similar procedure of the main steps which are summarized in Algorithm 2.

With the input of $A$, $y$, $\epsilon$ and $n$ in a SEM algorithm, the index of the columns which are not all zero need to be picked out as $I_0$ as well. The initialization part also includes that setting the iteration counter $k$ as zero, building up a M-size all-zero array $\lambda$ and a N-size all-one array $T$.

In each iteration of the first layer loop, $k$ has an increment of one and the link index $a$ is set zero before the second layer of loop starts. We also do a traversal of all the links and compute the ratio $\lambda$ with updating $\lambda_a = \frac{P^aT}{y_a}$. The coefficient vector $\alpha$ is obtained by $\alpha = \lambda^{-P^w}$ and used to update $T$ by $T = \alpha \cdot T$. If $k$ has come to the upper limit $n$ or the amplitudes of all the $M$ elements of $\alpha$ are very close to 1, that is ($|\alpha - 1| < \epsilon$), the iteration could get stopped and the output $\hat{x}$ is obtained as $\hat{x}_{I_0} = T$ and $\hat{x}_{\bar{I}_0} = 0$. We can get a better overview of the algorithm from Figure 5.6.

We also set an appropriate $n$ as the loop upper limit. Although the error usually decreases monotonously as in Figure 5.4, we still set an additional stopping criterion that the error is larger than the threshold and the iteration
operation only in the monotonous part considering the case of the error oscillation shown in Figure 5.5.
Algorithm 2: Sparse-adapted Entropy Maximising

Input:
1. $A$, $y$, $\epsilon, n$;

Initialization:
1. Iteration counter $k \leftarrow 0$;
2. $I_0 \leftarrow$ indices of the all-zero columns of $A$;
3. $P \leftarrow A_{I_0}$
4. $\lambda \leftarrow 0$, $\lambda \in \mathbb{R}^M$;
5. $T \leftarrow 1$, $T \in \mathbb{R}^N$;

Iterations:
1. repeat
2. $k \leftarrow k + 1$;
3. $a \leftarrow 0$;
4. repeat
5. $a \leftarrow a + 1$;
6. $\lambda_a \leftarrow \frac{P^a y}{\lambda_y}$;
7. $\alpha \leftarrow \lambda_a^{-1} P^a$;
8. $T \leftarrow \alpha \cdot T$;
9. until ($a == M$)
10. until all ($|\alpha - 1| < \epsilon$) or ($k > n$)

Output:
1. $\hat{x} \in \mathbb{R}^N$, satisfying $\hat{x}_{I_0} = T$ and $\hat{x}_{I_0} = 0$.

5.3 BP and BPDN

As a typical convex relaxation algorithm, Basis Pursuit (BP)/Basis Pursuit Denoising (BPDN) aims at finding out a minimized $l_1$ norm of the solution vector regarding convex constraint functions [5]. The algorithm is presented in Section 5.3.1. In Section 5.3.2, the definitions of some commonly used norms are discussed. And Section 5.3.3 will give detailed analysis of the reason why $l_1$ norm is chosen in this convex relaxation algorithm.

5.3.1 Algorithms of BP and BPDN

For the clean measurement case, that is, there is no measurement noise adding, the problem could be solved by

$$\hat{x} = \arg \min \|x\|_1,$$  \hspace{1cm} (5.1)

subject to $y = Ax; \hspace{1cm} (5.2)$
And as to a noisy case, this kind of problem can be solved by

\[ \hat{x} = \arg \min \|x\|_1, \]

subject to \( \|y - Ax\|_2 \leq \epsilon \) \hspace{1cm} (5.3)

using a standard linear programming tool.
5.3.2 Norm definitions

In a mathematical scenario, the $l_1$ norm of an array is actually the summation of the absolute values of all the elements. It is also known as Taxicab
norm or Manhattan norm and written as

$$\|x\|_1 = \sum_{i=1}^{n} |x_i|.$$  \hfill (5.5)

Similarly, there is $l_2$ norm defined as

$$\|x\|_2 = \sqrt{\sum_{i=1}^{n} x_i^2}.$$  \hfill (5.6)

in mathematics. It is also called Euclidean norm and most commonly used. While in signal processing field, a notation refered to zero ”norm” by Donoho is the sparsity of an array as

$$\|x\|_0 = \sum_{i=1}^{n} |x_i|^0 (x_i \neq 0).$$  \hfill (5.7)

### 5.3.3 Choice of $l_1$ norm

$l_0$ norm is the sparse solution for this kind of compressive sensing problem, that is, the sparsest solution to

$$y = Ax.$$  \hfill (5.8)

This is a NP-Complete problem. So although it is precise, but it is quite impractical in our case.

The method of least squares that using $l_2$ norm is much faster, but it might result in a wrong solution in the sparse case. Because the sparse solution is usually different from the least square solution.

Figure 5.7 shows this in a two-dimensional case. As described in (5.6), the method to find a least square solution is increasing the radius gradually until tangent to the line $y = Ax$. The tangent $x^*$ is the least square solution, which is obviously different from the sparse solution $x$ and makes it quite inaccurate. But how does $l_1$ norm work? Similarly with the method solving a least square problem, the basic idea of finding the minimization of $l_1$ norm problem is illustrated in Figure 5.8.

Figure 5.8 shows the $l_1$ norm method to the same problem. It starts from the origin and increases the side length of the rhombus until tangent to the line $y = Ax$. The tangent $x^*$ is the minimization solution of this $l_1$ norm, which is equal to the sparse solution $x$. It was proved by Tao and Candès that the $l_1$ norm problem has a same solution with the $l_0$ norm problem in a $n$-dimensional space with RIP discussed in Section 2.2 obeyed.

In a noisy case of $y = Ax + w$, the norm of the noise $w$ is assumed to have a tolerance $\epsilon$ that $\|Ax - y\| \leq \epsilon$. This $l_1$ norm method in a two dimensional case with noise added is shown in Figure 5.9. The area of $Ax \approx y$ is between the two red lines, and $x^*$ is the basis pursuit solution to the problem $\|Ax - y\| \leq \epsilon$ that is close to the original $x$. 23
Figure 5.7: The least square solution and sparse solution in two-dimensional space

### 5.4 OMP

As an iterative greedy algorithm, orthogonal matching pursuit (OMP) computes the residuals step by step and selects the most correlated atom. It has a more strict restriction than RIP that the correlation of any two columns of $A$ should not be larger than $\frac{1}{2\sqrt{K}}$, which is proved by the Gershgorin circle theorem [27]. First, we summarize the main steps of the OMP in Algorithm 3 (see Algorithm 3 of [26]).

This algorithm approximates the $x$ vector based on the input of $A$, $y$ and $K$. As we have talked before, $A$ is the measurement matrix with the dimension of $M \times N$ and $y$ is the observation with the dimension of $M \times 1$. Here $K$ is the sparsity level of $x$.

Firstly the iteration counter $k$ is set to be zero and the initial residual $r_0$ with a dimension of $M \times 1$ equal to $y$. An empty support vector is set up as $I_0$. Then we go to the iteration step as the iteration counter $k$ is increased one by one each iteration. For the $k^{th}$ step, the highest amplitude of $A^t y$ is picked out with the index $i_k$ recorded. Then the support set $I$ is updated with a new element of $i_k$ added to it. We update the residual by computing $r_k = y - A_{I_k} A_{I_k}^t y$, where $A_{I_k}$ is the $I_k$ columns of $A$ matrix and $A_{I_k}^t$ is the pseudoinverse of $A_{I_k}$. The pseudoinverse $A_{I_k}^t = [A_{I_k}^t A_{I_k}]^{-1} A_{I_k}^t$. The new residual is compared with the previous one. It is stopped when the new
Algorithm 3: OMP for CS Recovery

Input:
1: \( \mathbf{A}, \mathbf{y}, K; \)

Initialization:
1: Iteration counter \( k \leftarrow 0; \)
2: \( \mathbf{r}_0 \leftarrow \mathbf{y}, \mathcal{I}_0 \leftarrow \emptyset; \)

Iterations:
1: repeat
2: \( k \leftarrow k + 1; \)
3: \( i_k \leftarrow \text{index of the highest amplitude of } \mathbf{A}^\dagger \mathbf{r}_{k-1}; \)
4: \( \mathcal{I}_k \leftarrow \mathcal{I}_{k-1} \cup i_k; \) \hspace{1cm} \text{(Note: } |\mathcal{I}_k| = k)\)
5: \( \mathbf{r}_k \leftarrow \mathbf{y} - \mathbf{A}_{\mathcal{I}_k} \mathbf{A}_{\mathcal{I}_k}^\dagger \mathbf{y}; \) \hspace{1cm} \text{(Orthogonal projection)}
6: until \( (||\mathbf{r}_k||_2 > ||\mathbf{r}_{k-1}||_2 \text{ or } (k > K)) \)
7: \( k \leftarrow k - 1; \) \hspace{1cm} \text{(Previous iteration)}

Output:
1: \( \hat{x} \in \mathbb{R}^N, \text{satisfying } \hat{x}_{\mathcal{I}_k} = \mathbf{A}_{\mathcal{I}_k}^\dagger \mathbf{y} \text{ and } \hat{x}_{\mathcal{I}_k^c} = 0. \)

residual is larger than the previous one or \( K \) times of iteration have been executed, or else go back to step 2. When the iteration is done, the output of \( \hat{x} \) could be approximated as the \( \hat{x} \) with index of \( \mathcal{I}_k \) are set to be \( \mathbf{A}_{\mathcal{I}_k}^\dagger \mathbf{y}, \)
and all the other $\hat{x}$ remain to be zero. The flow chart of OMP algorithm is shown in Figure 5.10.

5.5 SP

Subspace pursuit (SP) takes the idea of backtracking, that is, before the signal is reconstructed, a candidate set is initially set up and those unnecessary atoms are given up to form the final candidate set. SP has a lower computation complexity and the main steps are summarized in Algorithm 4 (see Algorithm 1 of [14])

Similarly with orthogonal matching pursuit, the measurement matrix $A$, the observations $y$ and the sparsity $K$ are taken as input of SP algorithm. The iteration counter $k$ is set to be zero firstly, and an initial support set is selected from the $K$ highest amplitudes of $A^t y$. Then a set of residuals could be computed from $y - A_{I_0} A_{I_0}^t y$. After the initialization step, we come to the iteration part. In each step, the counter $k$ has an increment of one. A preliminary support set $I_{(p)}$ is obtained from the $K$ highest amplitudes of $A^t y$. We need to get the union of $I_{(p)}$ and the previous support set $I_{k-1}$ as $I_{(u)}$. If $I_{(p)}$ is exactly the same with $I_{k-1}$, then the cardinality of $I_{(u)}$ should be $K$. And if $I_{(p)}$ is totally different from $I_{k-1}$, then $I_{(u)}$ has a cardinality of $2K$. With a overlapping between $I_{(p)}$ and $I_{k-1}$, the cardinality is in between.
Thus the cardinality of $\mathcal{I}_{(u)}$ has the range $[K, 2K]$. An approximate solution $\hat{x}$ is computed by setting $\hat{x}_{\mathcal{I}_{(u)}}$ equal to $A_{\mathcal{I}_{(u)}}^\dagger y$ and the other elements of $\hat{x}$ zero. The updated support set $\mathcal{I}_k$ is formed by picking out the $K$ highest amplitudes of $\hat{x}$, and the new residual $r_k$ is computed by $y - A_{\mathcal{I}_k} A_{\mathcal{I}_k}^\dagger y$ and compared with the previous one. The stopping criteria is satisfied when $(\|r_k\|_2 > \|r_{k-1}\|_2)$, and an output of $\hat{x}$ is obtained by $\hat{x}_{\mathcal{I}_k} = A_{\mathcal{I}_k}^\dagger y$ and the other elements set as zero. The flow chart of SP algorithm is illustrated in Figure 5.11.
Algorithm 4: SP for CS Recovery

Input:
1: $A$, $y$, $K$;

Initialization:
1: Iteration counter $k \leftarrow 0$;
2: $I_0 \leftarrow \text{indices of the } K \text{ highest amplitudes of } A^t y$;
3: $r_0 \leftarrow y - A_{I_0} A_{I_0}^t y$;

Iterations:
1: repeat
2: $k \leftarrow k + 1$;
3: $I_{(p)} \leftarrow \{\text{indices of } K \text{ highest amplitudes of } A^t r_{k-1}\}$;
4: $I_{(u)} \leftarrow I_{k-1} \cup I_{(p)}$; \hspace{1cm} ($K \leq |I_{(u)}| \leq 2K$)
5: $\hat{x}_{I_{(u)}} \leftarrow A_{I_{(u)}}^t y$; $\hat{x}_{I_{(u)}} \leftarrow 0$; \hspace{1cm} (Orthogonal projection)
6: $I_k \leftarrow \{\text{indices of the } K \text{ highest amplitudes of } \hat{x}\}$;
7: $r_k \leftarrow y - A_{I_k} A_{I_k}^t y$; \hspace{1cm} (Orthogonal projection)
8: until ($\|r_k\|_2 > \|r_{k-1}\|_2$)
9: $k \leftarrow k - 1$; \hspace{1cm} (Previous iteration)

Output:
1: $\hat{x} \in \mathbb{R}^N$, satisfying $\hat{x}_{I_k} = A_{I_k}^t y$ and $\hat{x}_{I_k} = 0$.

5.6 OMP$_+$

The OMP$_+$ algorithm is developed by S. Chatterjee in 2014 that based on the standard OMP algorithm and the main steps are shown in Algorithm 5 [11]. As discussed in Section 4.5, all the elements of $y$, $A$ and $x$ are non-negative, this characteristic could be used to obtain a better performance.

It could be seen that this algorithm is very similar to OMP but with some changes. Firstly in Step 3, the highest element of $A^t r_{k-1}$ is selected, but instead of the highest amplitude. This is because we only want the non-negative projection of the system matrix $A$ and the residual $r_{k-1}$. In Step 5, the estimated $\hat{x}_{I_{(u)}}$ is obtained no longer by a least square method, but using some methodology to get the non-negative least square solution.

The iteration runs until all the elements in the projection of $A$ and $r_{k-1}$ are negative or the iteration times has exceeded the assumed sparsity. The flowchart of this OMP$_+$ algorithm is illustrated in Figure 5.12

5.7 SP$_+$

This SP$_+$ algorithm is proposed based on the standard SP algorithm. It also uses the characteristic that all the elements in $y$, $A$ and $x$ are non-negative with the aim at improving the promise of the reconstruction.

28
Algorithm 6 summarizes the main steps of SP, and the difference between SP and SP+ is easy to tell.

Instead of getting the K highest amplitudes of $A^T y$ as the initial index

Figure 5.11: The flow chart of SP algorithm
Algorithm 5: OMP+ for CS Recovery

Input:
\[A, y, K;\]

Initialization:
1: Iteration counter \( k \leftarrow 0; \)
2: \( r_0 \leftarrow y, I_0 \leftarrow \emptyset; \)

Iterations:
1: \textbf{repeat}
2: \( k \leftarrow k + 1; \)
3: \( i_k \leftarrow \text{index of the highest element of } A^t r_{k-1}; \)
4: \( I_k \leftarrow I_{k-1} \cup i_k; \)
5: \( \hat{x}_{I_k} \leftarrow \arg \min_{\beta} \|y - A_{I_k} \beta\|, \beta \geq 0; \)
6: \( r_k \leftarrow y - A_{I_k} \hat{x}_{I_k}; \)
7: \textbf{until} \((A^t r_{k-1}(i_k) \leq 0) \text{ or } (k > K)) \)
8: \( k \leftarrow k - 1; \)

Output:
1: \( \hat{x} \in \mathbb{R}^N, \text{satisfying } \hat{x}_{I_k} = 0. \)

\(I_0, \text{SP}_+\) first counts the number of non-negative elements of \(A^t y\) in Step 2 and then compare this number \(S_0\) with the assumed sparsity \(K\). The initial index \(I_0\) is generated by picking out the smaller number between \(K\) and \(S_0\) of highest elements of the projection of transposition of the system matrix \(A\) and the measurements vector \(y\). An initial residual \(r_0\) is calculated by the difference between \(y\) and the modeled measurements.

Likewise, Step 4 records a certain number of highest elements of \(A^t r_{k-1}\). This certain number comes from the comparison between \(S_k\) (obtained from Step 3 and \(K\).

Another modification is in Step 7 that we get the \(K\) highest elements of \(\hat{x}\), which is the non-negative least square solution of \(y = A_{I_k} x\). The residual vector \(r_k\) is also acquired from the difference between the observations \(y\) and the modeled observations.

Figure 5.13 gives a more easily understood illustration of the \(\text{SP}_+\) algorithm.
Figure 5.12: The flow chart of OMPplus algorithm
Algorithm 6: SP$^+$ for CS Recovery

Input:
1: $A$, $y$, $K$;

Initialization:
1: Iteration counter $k ← 0$;
2: $S_0 ←$ the number of non-negative elements of $A'y$;
3: $I_0 ←$ indices of the $\min(K, S_0)$ highest elements of $A'y$;
4: $\hat{x}_{I_0} ← \arg\min \|y - A_{I_0}\beta_0\|$, $\beta_0 ≥ 0$;
5: $r_0 ← y - A_{I_0}\hat{x}_{I_0}$;

Iterations:
1: repeat
2: $k ← k + 1$;
3: $S_k ←$ the number of non-negative elements of $A'r_{k-1}$;
4: $I(p) ← \{\text{indices of } \min(K, S_k) \text{ highest elements of } A'r_{k-1}\}$;
5: $I(u) ← I_{k-1} \cup I(p)$;
6: $\hat{x}_{I(u)} ← \arg\min \|y - A_{I(u)}\beta_i\|$, $\beta_i ≥ 0$;
7: $I_k ← \{\text{indices of the } K \text{ highest elements of } \hat{x}\}$;
8: $x_{I(k)} ← \arg\min \|y - A_{I(k)}\beta_i\|$, $\beta_i ≥ 0$;
9: $r_k ← y - A_{I_k}\hat{x}_{I(k)}$;
10: until $(\|r_k\|_2 > \|r_{k-1}\|_2)$ or $(k > K)$
11: $k ← k - 1$;

Output:
1: $\hat{x} ∈ \mathbb{R}^N$, satisfying $x_{I_k} = 0$. 

32
Figure 5.13: The flow chart of SPplus algorithm
Chapter 6

Numerical experiments

In this chapter, we first do experiments on a set of artificial data to study the characteristics of CS recovery, and then implement those CS recovery algorithms as well as the methods in transport field on a very preliminary simulation data. Finally, the implementation on the realistic simulated data of Stockholm is presented. The experiments are carried out on a 64-bit Windows computer. All the algorithms are implemented in MATLAB 2013b. A cvx package is needed to install to implement the BP/BPDN algorithm.

6.1 Performance measures

In this section several common performance measures will be introduced.

6.1.1 SRNR

We first mention about one of the performance measures, which is called signal-to-reconstruction-noise ratio (SRNR) and defined as

$$\text{SRNR} = \frac{\mathcal{E}\{\|x\|^2\}}{\mathcal{E}\{\|x - \hat{x}\|^2\}}$$

(6.1)

Here $\hat{x}$ is the estimated OD trip in a vector form. The estimation is better when SRNR is higher. It is commonly used when the true $x$ is known.

6.1.2 ASCE

We use average support-cardinality error (ASCE) as the second performance measure. ASCE is defined as

$$\text{ASCE} = \mathcal{E}\{d(I, \hat{I})\} = 1 - \frac{1}{K}\mathcal{E}\{|I \cap \hat{I}|\}$$
\( \mathcal{I} \) and \( \hat{\mathcal{I}} \) represent the support set of \( x \) and \( \hat{x} \), respectively. \( K \) is the sparsity of \( x \), which is also the cardinality of \( \mathcal{I} \). The distortion, defined as
\[
d(\mathcal{I}, \hat{\mathcal{I}}) = 1 - \frac{|\mathcal{I} \cap \hat{\mathcal{I}}|}{K},
\]
is used to measure the support set estimation error [24]. When there is a large number of signal vectors, we use ASCE, which is the average of \( d(\mathcal{I}, \hat{\mathcal{I}}) \) to measure the performance. The reconstruction is more accurate as ASCE is smaller. Here ASCE has the range \([0, 1]\).

### 6.1.3 RMSE

Besides the performance measure SRNR in signal processing field, another measure called root-mean-square error (RMSE) is frequently used to evaluate the reconstruction accuracy with use of estimated trips \( \hat{y} \) and observed trips \( y \). RMSE is defined as
\[
\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{m} (\hat{y}_i - y_i)^2}{m}}.
\]
Where \( \hat{y} \) is calculated by
\[
\hat{y} = A\hat{x}.
\]

### 6.2 Experiments on artificial data

The problem to solve is set up as
\[
y = Ax.
\]
Here a set of \( x \) vector is generated with 1000 binary elements (\( N = 1000 \)). We will use BP/BPND and OMP algorithms to study the influence resulting from different sparsity levels (\( K \)) and different measurements numbers (\( M \)). The running time is compared as well. \( A \) is a Gaussian matrix that generated randomly with a dimension of \( M \times N \), and \( y \) is obtained by computation of \( A \times x \).

Figure 6.1 shows the SRNR curves from two different \( x \) vectors with \( K = 50 \) and \( K = 100 \), where OMP algorithm used. And Figure 6.2 shows the output when BP algorithm used. The x-axis represents the number of observations (\( M \)), and the y-axis shows the corresponding SRNR value. The circle line shows the reconstruction performance of \( x \) with \( K = 50 \), and the triangle line shows performance of \( x \) with \( K = 100 \). From both of the figures, we could see that when \( K \) increases, which means we have more non-zero elements in the \( x \) vector, we need more measurements to get a
Figure 6.1: Reconstruction from different sparsity signal with OMP

Figure 6.2: Reconstruction from different sparsity signal with BP

Table 6.1: Computation time of OMP and BP on artificial data

<table>
<thead>
<tr>
<th></th>
<th>OMP</th>
<th>BP</th>
</tr>
</thead>
<tbody>
<tr>
<td>running time</td>
<td>2.6299s</td>
<td>209.6690s</td>
</tr>
</tbody>
</table>
perfect reconstruction. The two algorithms generate similar results, but the running time is quite different that could be seen from Table 6.1.

A similar conclusion could be drawn in Figure 6.3, which shows the ASCE result from the two sparsity $\mathbf{x}$ vector with OMP algorithm implemented.

![Figure 6.3: ASCE of different sparsity signal](image)

Figure 6.4 and Figure 6.5 also show the SRNR curves with OMP and BPDN algorithms implemented. In this case we use the same $\mathbf{x}$ with $K = 50$ as in Figure 6.1. Two set of Gaussian noise with SMNR=10dB and SMNR=20dB are added to the same $\mathbf{x}$ set. The signal-to-measurement-noise-ratio (SMNR) is defined as

$$\text{SMNR} = \frac{\mathcal{E}\{\|\mathbf{x}\|_2^2\}}{\mathcal{E}\{\|\mathbf{w}\|_2^2\}}.$$ (6.5)

The circle line shows the reconstruction performance of the noise added signal with a SMNR level of 10dB, and the star line shows the one with a SMNR level of 20dB. The running time of these two algorithms are illustrated in Table 6.2. And from the similar results of Figure 6.4 and Figure 6.5, we can apparently draw the conclusion that the signal with smaller noise could be reconstructed better.

Figure 6.6 shows the ASCE curve of the reconstruction of $\mathbf{x}$ with different noise level. The triangle line shows the one with 20dB noise added, which decreases more sharply than the circle line of 10dB.
Figure 6.4: Reconstruction from signal with different noise level with OMP

Figure 6.5: Reconstruction from signal with different noise level with BPDN

Table 6.2: Computation time of OMP and BPDN on artificial data

<table>
<thead>
<tr>
<th></th>
<th>OMP</th>
<th>BPDN</th>
</tr>
</thead>
<tbody>
<tr>
<td>running time</td>
<td>1.1142s</td>
<td>686.1376s</td>
</tr>
</tbody>
</table>
We can see that greedy algorithm is computationally simpler than the convex relaxation method, and hence attractive for large systems.

### 6.3 Implementation on a preliminary simulation data

This data is simulated quite preliminarily from the car morning rush hour in Stockholm. The raw data could be divided into three parts: origins, destinations and the measurements, which is in the format as shown in Table 6.3.

<table>
<thead>
<tr>
<th>ORIGIN</th>
<th>DESTINATION</th>
<th>LINK</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>origin₁</td>
<td>link₁, link₂, ⋅⋅⋅, linkᵢ</td>
</tr>
<tr>
<td>2</td>
<td>origin₂</td>
<td>link₂, link₃, ⋅⋅⋅, linkᵢ</td>
</tr>
<tr>
<td>3</td>
<td>originᵢ</td>
<td>linkₗ, link₁₀, ⋅⋅⋅, linkₖ</td>
</tr>
<tr>
<td>⋮</td>
<td>⋮</td>
<td>⋮</td>
</tr>
<tr>
<td>4227</td>
<td>origin₉₇₄</td>
<td>linkᵢ, linkⱼ, ⋅⋅⋅, link₂₀₂₀₁</td>
</tr>
</tbody>
</table>

As the convex algorithm does not work for this problem, we use the greedy algorithms of OMP and SP in signal processing field to evaluate the
OD estimation performance, as well as the discussion of the experimental setups. The algorithms of EM and IM will be tested later.

The measurements here correspond to the quantity of the vehicles that appear in some path and also appear in some link, which are identified with combination of numbers and letters. There are 974 different origins and 824 different destinations in the data. There is some overlap between the origins and destinations. Considering the overlaps, we found that we have 1158 different locations in total. There are 20201 different links. We need to transform the raw data to a standard form of $y = Ax$. We realize this by creating look-up tables for the OD matrix and the links. Thus we could easily get $x$ and $y$ by statistics. The $A$ matrix could be derived with an equation as

$$A(i, j) = \frac{\sum y_{i}^{x_{j}}}{\text{length}(x_{j})}$$

Here $y_{i}^{x_{j}}$ means the frequency of $y_{i}$ that appears in path $x_{j}$. $A(i, j)$ corresponds to the probability that link $k_{i}(y(i))$ is used when traveling between OD pair $j$. For this data, the dimensions of $x$, $y$ and $A$ are 1340964 $\times$ 1, 20201 $\times$ 1 and 20201 $\times$ 1340964, respectively.

We use fraction of sampling (or fraction of measurements) to show the estimation performance under different sampling levels. The fraction of sampling is defined as

$$\alpha = \frac{M'}{M}$$

where $M' \leq M$, and $M = 20201$. Note that $\alpha \in (0, 1]$. Further, Gaussian noise in different levels are added to the true measurement vector to evaluate the estimation performance in noisy environment (with the constraint of positiveness).

The OD estimation performance using OMP and SP algorithms are shown in Figure 6.7, Figure 6.8, Figure 6.9 and Figure 6.7. The $x$-axis shows the fraction of measurements. The $y$-axis shows the SRNR performance. Figure 6.7 shows it in a clean case, which means no noise is added to the system. Figure 6.8 shows the performance of the system has a noisy condition with SMNR = 10 dB. Figure 6.9 is the reconstruction output from a noisy condition with SMNR = 20 dB and Figure 6.10 adds a noise with SMNR = 30 dB to it.

Assuming the sparsity is unknown, we always used $K$ as the half of $M'$. It is noted that performance deteriorates significantly with decrease in link measurements.

We observe that OMP and SP provide similar results. For execution of algorithms for a full sized single data set, the cpu running time of OMP and SP is shown in Table 6.4.
Figure 6.7: OD estimation performance of OMP and SP algorithms for the preliminary simulation data in a clean case

Figure 6.8: OD estimation performance of OMP and SP algorithms for the preliminary simulation data with SMNR=10dB

6.4 Implementation on realistic simulated data

This data is a 2% sample of the full population, and is pretty realistic. But given that there is very much redundancy in the data, we could treat it like
Figure 6.9: OD estimation performance of OMP and SP algorithms for the preliminary simulation data with SMNR=20dB

Figure 6.10: OD estimation performance of OMP and SP algorithms for the preliminary simulation data with SMNR=30dB

a full population, which means that each trip in the data file corresponds to 50 trips in reality. The raw data has a same format with the preliminary one as is shown in Table 6.5. Thus we transform the raw data to the form
Table 6.4: Computation time of OMP and SP on preliminary data

<table>
<thead>
<tr>
<th></th>
<th>OMP</th>
<th>SP</th>
</tr>
</thead>
<tbody>
<tr>
<td>running time</td>
<td>4460s</td>
<td>15s</td>
</tr>
</tbody>
</table>

we need in a same way.

Table 6.5: Raw data from realistic simulation

<table>
<thead>
<tr>
<th>ORIGIN</th>
<th>DESTINATION</th>
<th>LINK</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>origin₁</td>
<td>destination₁</td>
</tr>
<tr>
<td>2</td>
<td>origin₂</td>
<td>destinationᵢ</td>
</tr>
<tr>
<td>3</td>
<td>originᵢ</td>
<td>destinationᵢ</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>14043</td>
<td>origin₁₀⁵₁</td>
<td>destination₈₈₆</td>
</tr>
</tbody>
</table>

There are 951 different origins and 886 different destinations in this new data. We have 25601 links and 1118 different locations in total with the overlapping excluded.

In this case, the dimensions of \( \mathbf{x}, \mathbf{y} \) and \( \mathbf{A} \) are \( 1249924 \times 1 \), \( 25601 \times 1 \) and \( 25601 \times 1249924 \), respectively. \( \mathbf{x} \) is shown in Figure 6.11.

![Figure 6.11: OD matrix \( \mathbf{x} \)](image)

Also we would like to see the reconstruction performance with those algorithms under different sampling levels. Here \( M = 25601 \) and Gaussian
noise with the level of SMNR = 10, 20, 30 dB is added.

### 6.4.1 Implementation with OMP and OMP+

The convex relaxation algorithm (BP/BPDN) still does not work for this problem. We will firstly test the performance with OMP and OMP+.

Figure 6.12, Figure 6.13 and Figure 6.14 show the SRNR performance of OMP and OMP+ with a Gaussian noise SMNR = 10 dB, SMNR = 20 dB and SMNR = 30 dB, respectively. Figure 6.15 shows this in a clean measurement case.

In those figures, the SRNR curve of OMP+ is represented in a blue curve with circles, and the SRNR curve of OMP is represented in a red curve with triangles. The x-axis shows the fraction of measurements. The y-axis shows the SRNR performance. It is clear to see that the SRNR curve of OMP+ is higher than that of OMP, which means the OMP+ works better with a more accurate reconstruction.

### 6.4.2 Validation of SP and SP+

In this part we will validate the performance of SP algorithm and the proposed SP+ algorithm. Similarly, we test this in four cases that SMNR = 10 dB, SMNR = 20 dB, SMNR = 30 dB and clean measurements. The result could be seen from Figure 6.16, Figure 6.17, Figure 6.18 and Figure 6.19.
Figure 6.13: OD estimation performance of OMP and OMP$_+$ on realistic data with SMNR=20dB

Figure 6.14: OD estimation performance of OMP and OMP$_+$ on realistic data with SMNR=30dB

The SRNR curve of SP$_+$ is in a blue curve with circles, and the other red curve with triangles represents the SRNR of SP algorithm. The x-axis shows the fraction of measurements, which is a ratio between the sampled
Figure 6.15: OD estimation performance of OMP and OMP$_+$ on realistic data with clean measurements

Figure 6.16: OD estimation performance of SP and SP$_+$ on realistic data with SMNR=10dB

link measurements and the total link measurements. As we have 25601 link measurements in total, the range of fraction from 0.05 to 1.0 means that the number of sampled link measurements is from 1280 to 25601. The y-axis
Figure 6.17: OD estimation performance of SP and SP$_+$ on realistic data with SMNR=20dB

Figure 6.18: OD estimation performance of SP and SP$_+$ on realistic data with SMNR=30dB

shows the SRNR in dB. From those four figures, we can easily see that SP$_+$ has a higher promise than SP for this problem.
6.4.3 Implementation on SIM

As could be seen from Figure 5.1 and Figure 5.2, the error basically decreases as the iteration times increases. This error is the maximum absolute value between the ratio of the modeled measurements to the actual link measurements and one. It is predictable that there is a relationship between the error and the reconstruction accuracy. Because the ultimate error depends on both the iteration times and the selected threshold, we can say that the SRNR of SIM algorithm is affected by the iteration times and the threshold $\epsilon$.

Figure 6.20, Figure 6.21, Figure 6.22 and Figure 6.23 show the performance of SIM method with different maximum iteration times ($n = 50$, $n = 100$ and $n = 3000$) defined. Those four figures present the results in different noise level cases, as SMNR = 10dB, SMNR = 20dB, SMNR = 30dB and clean measurements. The threshold of error $\epsilon$ is 0.05. It could be seen that in various noise level cases, with a fixed error threshold, the larger the $n$ is the better result SIM could get.

Figure 6.24, Figure 6.25, Figure 6.26 and Figure 6.27 select different error threshold ($\epsilon = 0.05$, $\epsilon = 0.2$, $\epsilon = 0.4$) with the maximum iteration times $n$ is fixed at 100. The three reconstruction curves are very close to each other and seem as if they coincide. That is because the maximum iteration times $n = 100$ is not enough for SIM to converge. After one hundred iterations, the ultimate error is not smaller than 0.4 in any of the four cases.
Figure 6.20: OD estimation performance of SIM under different n conditions with $\epsilon = 0.05$ and 10dB noise added

Figure 6.21: OD estimation performance of SIM under different n conditions with $\epsilon = 0.05$ and 20dB noise added

To verify the influence of $\epsilon$, we need to enlarge the maximum iteration times n (i.e. 300). And because this SIM algorithm does not converge so fast, we can shift the error threshold (i.e. to $\epsilon = 0.5$, to $\epsilon = 0.7$ and to
Figure 6.22: OD estimation performance of SIM under different n conditions with $\epsilon = 0.05$ and 30dB noise added

Figure 6.23: OD estimation performance of SIM under different n conditions with $\epsilon = 0.05$ in a clean case

$\epsilon = 0.8$, respectively) to make a clearer picture of it.

As the noise may result in unexpected influence on the reconstruction work, we still carry out this implementation on different noise level cases,
Figure 6.24: OD estimation performance of SIM under different $\epsilon$ conditions with $n = 100$ and 10 dB noise added

Figure 6.25: OD estimation performance of SIM under different $\epsilon$ conditions with $n = 100$ and 20 dB noise added

that is, SMNR = 10 dB, SMNR = 20 dB, SMNR = 30 dB and clean measurements. The performance could be seen from Figure 6.28, Figure 6.29, Figure 6.30 and Figure 6.31. There is an intersection of the red triangle
Figure 6.26: OD estimation performance of SIM under different $\epsilon$ conditions with $n = 100$ and 30 dB noise added

Figure 6.27: OD estimation performance of SIM under different $\epsilon$ conditions with $n = 100$ in a clean case

curve with threshold $\epsilon = 0.5$ and the blue circle curve with threshold $\epsilon = 0.7$ when fraction $\alpha = 0.7$ in Figure 6.28, which is because of the slow convergency of SIM and the maximum deviation is still above 0.7 after 300
iterations. It could be seen from those four figures that as the chosen error threshold $\epsilon$ decreases, the recovery works better.

Figure 6.28: OD estimation performance of SIM under different $\epsilon$ conditions with $n = 300$ and 10 dB noise added

Figure 6.29: OD estimation performance of SIM under different $\epsilon$ conditions with $n = 300$ and 20 dB noise added
Figure 6.30: OD estimation performance of SIM under different $\epsilon$ conditions with $n = 300$ and 30 dB noise added

Figure 6.31: OD estimation performance of SIM under different $\epsilon$ conditions with $n = 300$ in a clean case

The computation time differs a lot with different conditions, which is shown in Table 6.6 and Table 6.7. Although a better result could be obtained with a larger $n$ and a smaller $\epsilon$, the endless computation is unbearable. So
with a trade-off between the reconstruction accuracy and the computation complexity existing, a suitable balance point should be found.

Table 6.6: Computation time of SIM with different maximum iteration times

<table>
<thead>
<tr>
<th>$\epsilon = 0.05$</th>
<th>$n = 50$</th>
<th>$n = 100$</th>
<th>$n = 3000$</th>
</tr>
</thead>
<tbody>
<tr>
<td>running time</td>
<td>2682s</td>
<td>6329s</td>
<td>150170s</td>
</tr>
</tbody>
</table>

Table 6.7: Computation time of SIM with different error threshold

<table>
<thead>
<tr>
<th>$n = 100$</th>
<th>$\epsilon = 0.05$</th>
<th>$\epsilon = 0.2$</th>
<th>$\epsilon = 0.4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>running time</td>
<td>6329s</td>
<td>4755s</td>
<td>4936s</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$n = 300$</th>
<th>$\epsilon = 0.5$</th>
<th>$\epsilon = 0.7$</th>
<th>$\epsilon = 0.8$</th>
</tr>
</thead>
<tbody>
<tr>
<td>running time</td>
<td>15530s</td>
<td>12011s</td>
<td>10124s</td>
</tr>
</tbody>
</table>

6.4.4 Implementation on SEM

The error that defined in SEM also generally decreases as the iteration times raises, which could be seen from Figure 5.4 and Figure 5.5. In order to find the relationship between the reconstruction accuracy and the maximum iteration times $n$ and the error threshold $\epsilon$, we first validate this under different $n$ conditions and with the same error threshold. And then different error threshold is tested with a fixed $n$.

To test the performance of SEM under different $n$ conditions, we implement this with noise at different level (SMNR = 10 dB, SMNR = 20 dB, SMNR = 30 dB and clean measurements case). Figure 6.32, Figure 6.33, Figure 6.34 and Figure 6.35 show this with different $n$ ($n = 50$, $n = 100$, $n = 3000$) and a same error threshold $\epsilon = 0.05$. It could be seen that as the maximum iteration times $n$ increases, the SRNR curve rises up, which means the reconstruction work is better.

Different level of noise cases are implemented to test the influence of the error threshold $\epsilon$ on the problem as well. For this purpose, we set the maximum iteration times $n = 100$ and see the performance when the error threshold varies ($\epsilon = 0.05$, $\epsilon = 0.2$, $\epsilon = 0.4$). This could be seen from Figure 6.36, Figure 6.37, Figure 6.38 and Figure 6.39.

In the SMNR = 20 dB, SMNR = 30 dB and clean measurement case, the SRNR curves with a smaller $\epsilon$ is always above the curve with a larger $\epsilon$. That means in such an OD estimation problem, a better reconstruction result could be obtained by setting a smaller error threshold with SEM. In Figure 6.36, the three SRNR curves look quite close to each other. This is because the noise level is very high and the maximum iteration times $n = 100$ is not enough for the algorithm to converge to the error threshold we set.
The trade-off between reconstruction accuracy and computation complexity still exists, which is illustrated in Table 6.8 and Table 6.9.
Figure 6.34: OD estimation performance of SEM under different n conditions with $\epsilon = 0.05$ and 30dB noise added

Figure 6.35: OD estimation performance of SEM under different n conditions with $\epsilon = 0.05$ in a clean case

Table 6.8: Computation time of SEM with different maximum iteration times

<table>
<thead>
<tr>
<th>$\epsilon = 0.05$</th>
<th>n = 50</th>
<th>n = 100</th>
<th>n = 3000</th>
</tr>
</thead>
<tbody>
<tr>
<td>running time</td>
<td>3152s</td>
<td>3712s</td>
<td>85565s</td>
</tr>
</tbody>
</table>
Figure 6.36: OD estimation performance of SEM under different $\epsilon$ conditions with $n = 100$ and 10dB noise added

Figure 6.37: OD estimation performance of SEM under different $\epsilon$ conditions with $n = 100$ and 20dB noise added

Table 6.9: Computation time of SEM with different error threshold

<table>
<thead>
<tr>
<th>$n = 100$</th>
<th>$\epsilon = 0.05$</th>
<th>$\epsilon = 0.2$</th>
<th>$\epsilon = 0.4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>computation time</td>
<td>3712s</td>
<td>2080s</td>
<td>1831s</td>
</tr>
</tbody>
</table>

58
Figure 6.38: OD estimation performance of SEM under different $\epsilon$ conditions with $n = 100$ and 30dB noise added

Figure 6.39: OD estimation performance of SEM under different $\epsilon$ conditions with $n = 100$ in a clean case
Chapter 7

Discussion and Conclusion

After studying several standard algorithms (BP/BPDN, OMP and SP) and developed methods (OMP+, SIM and SEM), we propose an greedy algorithm called SP+ for the OD matrix estimation problem. In this chapter, we will discuss the comparison of those methods and talk about some valuable work that could be done in the future.

7.1 Discussion and Conclusion

From the description in previous chapters and our experimental study, we can see that convex relaxation method could not be applied for a large scale problem.

In order to compare the performance of SIM, SEM, OMP, SP, OMP+ and SP+, we need to consider both the reconstruction accuracy and the computation complexity. As SIM and SEM are not fitting the large scale problem so well, we set a relatively larger error threshold $\epsilon$ and add a maximum iteration times $n$ so that it could finish within a reasonable time.

The performance of those algorithms can be seen from Figure 7.1, Figure 7.2, Figure 7.3 and Figure 7.4, which show the reconstruction work in a noise level at SMNR = 10 dB, SMNR = 20 dB, SMNR = 30 dB and clean measurements case, respectively. The computation time is listed in Table 7.1.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>OMP</th>
<th>OMP+</th>
<th>SP</th>
<th>SP+</th>
</tr>
</thead>
<tbody>
<tr>
<td>running time</td>
<td>7701s</td>
<td>23495s</td>
<td>16s</td>
<td>52s</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>SIM</th>
<th>SEM</th>
</tr>
</thead>
<tbody>
<tr>
<td>running time</td>
<td>n=100</td>
<td>6329s</td>
</tr>
<tr>
<td>n=300</td>
<td>14210s</td>
<td>n=300</td>
</tr>
<tr>
<td>n=3000</td>
<td>150173s</td>
<td>n=3000</td>
</tr>
</tbody>
</table>
Figure 7.1: OD estimation performance of different algorithms with 10dB noise added

Basically, we can tell from the figures that the performance decreases apparently as the number of link measurements reducing. The dash lines show the performance of SIM with an error threshold $\epsilon = 0.05$. The gray dash line represents the result of SIM with maximum iteration times $n = 100$, the blue dash line shows that with $n = 300$ and the light blue dash line is the SRNR curve of SIM with $n = 3000$. As $n$ is enlarged, a better performance of SIM is obtained. Although SIM works for this problem, but the performance is not good and unbearably slow from Table 7.1.

The performance of SEM with $\epsilon = 0.05$ is presented by the dotted lines. The gray, blue and light blue dotted lines show the SRNR with $n = 100$, $n = 300$ and $n = 3000$, respectively. We could see that SEM works well, especially when the observations are limited and noise level is large, but the computation speed is still not satisfying according to Table 7.1.

OMP and OMP$_+$ generate good reconstruction results and are faster than SIM and SEM. Comparing OMP and OMP$_+$, we could see that OMP$_+$ shows a higher promise but with a cost of longer computation expense.

SRNR of SP is less than -5 when fraction $\alpha \in [0.4, 0.5)$ and is cut off from the figure. SP shows a similar result with OMP when fraction $\alpha \in [0.5, 1)$,
but is considerably fast. $SP_+$ is the best in our case, even better than $OMP_+$, with a minor longer computation cost than $SP$.

It could be seen from Figure 7.1, Figure 7.2, Figure 7.3 and Figure 7.4 that the new algorithm $SP_+$ shows a better performance when fraction $\alpha \geq 0.6$. This is because for a K-sparse signal $x$, CS reconstruction is under the constraint that the number of measurements $m$ is larger than $2K$ [16] [6] [7] [8].

A similar discussion could be derived from Figure 7.5, Figure 7.6, Figure 7.7 and Figure 7.8, which show the RMSE performance of these algorithms.

Thus we can come to the conclusion that in a setup of urban mobility sensing, sparse processing has advantage in computation complexity and also has a high promise for OD estimation problem when the basic requirements are satisfied.

### 7.2 Future Work

Problem in a highly under-determined system could be studied in the future, which means the number of link measurements is quite limited and the number of OD pairs is very large.
Figure 7.3: OD estimation performance of different algorithms with 30dB noise added.

Another possible plan would be the study of Bayesian estimation.
Figure 7.4: OD estimation performance of different algorithms with clean measurements
Figure 7.5: RMSE of different algorithms with 10dB noise added
Figure 7.6: RMSE of different algorithms with 20dB noise added
Figure 7.7: RMSE of different algorithms with 30dB noise added
Figure 7.8: RMSE of different algorithms with clean measurements
Bibliography

[1] M.G.H. Bell. The estimation of origin-destination matrices by con-


systems. In P. Marcotte and S. Nguyen, editors, Equilibrium and ad-

[4] Francesco Calabrese, Mi Diao, Giusy Di Lorenzo, Joseph Ferreira Jr.,
and Carlo Ratti. Understanding individual mobility patterns from ur-
ban sensing data: A mobile phone trace example.


[6] Emmanuel J. Candès and Terence Tao. Decoding by linear program-
ing. Information Theory, IEEE Transactions on, 51(12):4203–4215,

[7] Emmanuel J. Candès, Justin Romberg, and Terence Tao. Robust uncer-
tainty principles: Exact signal reconstruction from highly incomplete
frequency information. Information Theory, IEEE Transactions on,

Error correction via linear programming. In IEEE Symposium


origin-destination matrices using traffic counts. Transportation Science,
[11] Saikat Chatterjee, David Koslicki, Siyuan Dong, Nicolas Innocenti,
Lu Cheng, Yueheng Lan, Mikko Vehkaperä, Mikael Skoglund, Lars K.
Rasmussen, Erik Aurell, and Jukka Corander. Sek: sparsity exploiting

and their application to non-linear system identification. International


[14] W. Dai and O. Milenkovic. Subspace pursuit for compressive sens-
ing signal reconstruction. Information Theory, IEEE Transactions on,


[17] M. Elad. Sparse and redundant representations: From theory to appli-


[19] M. Maher. Inferences on trip matrices from observations on link vol-


complete and inaccurate samples. Tech. Report, no. 2008-01, California

pursuit: recursive function approximation with applications to wavelet
decomposition. In Signals, Systems and Computers, 1993. 1993 Con-
ference Record of The Twenty-Seventh Asilomar Conference on, pages
40 –44 vol.1, nov 1993.


