Available online at www.sciencedirect.com

SCIENCE DIRECT®

Transportation Research Part B 39 (2005) 497–518

TRANSPORTATION RESEARCH PART B

www.elsevier.com/locate/trb

Inferring origin–destination trip matrices with a decoupled GLS path flow estimator

Yu Nie *, H.M. Zhang [a], W.W. Recker [b]

[a] Department of Civil and Environmental Engineering, University of California, Davis, CA 95616, USA
[b] Department of Civil and Environmental Engineering, University of California, Irvine, CA 92697, USA

Received 1 May 2003; received in revised form 11 June 2004; accepted 7 July 2004

Abstract

Recently, path flow estimators (PFE) have been used for the estimation of origin–destination (O–D) matrices. This paper develops a formulation that incorporates a decoupled path flow estimator in a generalized least squares (GLS) framework. The approach seeks to solve a GLS problem that minimizes the sum of errors in traffic counts and O–D matrices based on an equilibrium assignment mapping derived exogenously from a K-shortest path ranking procedure. Solving the GLS–PFE inevitably involves non-invertible linear systems and non-negative constraints. A solution algorithm is designed to iteratively identify active constraints and solve linear systems by computing the pseudoinverse. A simplified version of this algorithm is further developed to improve its computational efficiency. The solution properties and computational efficiency of the two methods are tested and compared for small to mid-size networks. It is concluded that the simplified algorithm is efficient in solving the decoupled GLS–PFE problem for realistic size networks.
© 2004 Published by Elsevier Ltd.

Keywords: Generalized least squares; Path flow estimator; Origin–destination trip matrix

* Corresponding author. Tel.: +1 530 754 9203; fax: +1 530 752 7872.
E-mail address: hmzhang@ucdavis.edu (Y. Nie).

0191-2615/$ - see front matter © 2004 Published by Elsevier Ltd.
doi:10.1016/j.trb.2004.07.002

1. Introduction

An essential element of transportation planning and/or operations is to "allocate" trips onto a transportation network according to either a static or dynamic assignment mapping. A fundamental input to traffic assignment is the origin–destination (O–D) trip matrix that depicts traffic demands between each origin and destination pair. Because the "true" O–D matrices are rarely, if ever, directly obtainable in practice, their estimation from limited observations of traffic conditions on the network has been the subject of many research investigations.

Traditionally, O–D matrices are derived from household or roadside surveys. However, survey-based approaches have two main shortcomings: they are costly and labor intensive, and typically they fail to capture temporal changes in O–D demand patterns beyond peak vs off-peak conditions. The latter shortcoming is particularly limiting in devising effective real-time strategies for relieving traffic congestion through application of ITS technologies. This has spurred a stream of research efforts focused on inferring O–D matrices from traffic counts on road segments, which if successful, not only is faster, cheaper and more convenient than household or roadside surveys, but also offers the real potential of providing contemporaneous O–D information for real-time management of traffic.

In terms of link volumes (or traffic counts) during any specified period, the fundamental framework for estimating an O–D matrix for that period having o O–D pairs with m observed link traffic counts, can be expressed by the linear system:

$$\sum_r \sum_s p_{rs}^a q_{rs} = \bar{x}_a \quad \forall r \in R,\ s \in S,\ a \in A \tag{1}$$

where R, S and A denotes a set of origins, destinations and links, respectively. q_{rs} denotes the traffic demand between O–D pair rs to be estimated, p_{rs}^a the proportion of trips between O–D pair rs using link a, and $\bar{x}_a$ represents the average measured traffic volume on link a.

Using matrix notation, system (1) can be rewritten as

$$\mathbf{Pq} = \bar{\mathbf{x}} \tag{2}$$

where $\mathbf{P}$ is an $m \times o$ matrix, $\mathbf{q}$ is a $o \times 1$ vector and $\bar{\mathbf{x}}$ is an $m \times 1$ vector.

As stated, the problem of O–D matrix estimation is to solve the linear system (2) for $\mathbf{q}$ according to a given rule for determining the coefficient matrix $\mathbf{P}$. Generally, (2) is underdetermined and thus not able to yield a unique solution. A common technique to force a "unique" solution is to resort to an optimization program in which either maximum entropy or minimum "distance" (to a specified "target" O–D matrix) is the objective.

Depending on the method by which the coefficient matrix $\mathbf{P}$ is determined, O–D estimation can be categorized into two major groups: proportional-assignment methods and equilibrium-assignment methods. If the effect of traffic congestion in a network is of minor importance, i.e., travel time is unaffected by changes in traffic flow on a given link, $\mathbf{P}$ can be assumed to be exogenously determined and obtained by means of some proportional assignment procedures, e.g., a simple all-or-nothing (AON) assignment or a more advanced stochastic assignment. Alternatively, when congestion becomes significant, Wardrop's first principle (Wardrop, 1952), which gives the well-known user-equilibrium (UE) condition in a congested transportation network, should then be applied. In this case, $\mathbf{P}$ is no longer a simple proportion matrix that depicts the proportion of each

O–D trips using each link. Instead, **P** becomes an equilibrium assignment mapping, which relates the O–D demand matrix to a flow pattern satisfying the UE condition. In a certain sense, the equilibrium-based O–D matrix estimation can be thought of as an inverse of the UE traffic assignment.

Early research efforts using the proportional assignment approach for O–D estimation were credited to: Willumsen (1981), for introducing the maximum entropy form; Van Zuylen and Willumsen (1980), for using the concept of minimizing information; and Cascetta (1984), for casting the problem in a generalized least squares framework. A number of researchers, including Bell (1991), Brenninger-Göthe et al. (1989), and Lo et al. (1996), either conducted tests or proposed improvements along the lines of proportion-based estimation.

Equilibrium-based O–D matrix estimation was pioneered by Nguyen (1977), and Fisk and Boyce (1983). Yang et al. (1992) showed that a bi-level optimization formulation can be used to model the problem by integrating an upper level in which the distance function is minimized, with a lower level in which the user-equilibrium flow pattern, and thereby the coefficient matrix **P**, is sought through a standard traffic assignment routine.

Instead of estimating the O–D matrix directly, Sherali et al. (1994) proposed to infer optimal path flows by formulating a linear programming problem, which is referred to here as the linear path flow estimator (LPFE). The optimal path flows in Sherali's approach are estimated so as to approach the user-equilibrium flow pattern, regenerate link traffic counts and concur with a target O–D matrix as closely as possible. Bell et al. (1997,) extended Shearli's path flow estimator to the log-linear path flow estimator by using stochastic user-equilibrium assignment. In order to avoid enumerating paths or applying a column generation procedure (which is required by LPFE but computationally inefficient), Nie and Lee (2002) suggested decoupling Sherali's linear model, seeking equilibrium paths independently by a K-shortest path ranking procedure.

In either its coupled or decoupled form, the LPFE model has difficulty in driving its estimates toward the real O–D matrix (not the target matrix) when the target matrix is subject to considerable error; this being the case even if the equilibrium flow pattern can be regenerated precisely. In view of the limitations of the linear programming structure, the research reported in this paper incorporates the decoupled path flow estimator (Nie and Lee, 2002) into the generalized least squares (GLS) framework. Although GLS-based O–D estimation has been studied extensively, our decoupled GLS–PFE formulation sheds light on two significant problems attendant to such estimation procedures: (1) accommodating the user equilibrium condition within a one-stage optimization formulation, and (2) dealing with a non-invertible matrix while satisfying non-negativity constraints.

We seek to decouple the estimation and assignment problems because the popular bi-level approach has two well recognized drawbacks: no guarantee to a convergent solution (not to mention global optimum) and potential computational difficulty (particularly when dynamic traffic management is concerned). Mathematical programming with equilibrium constraints, or MPEC, seems to offer a promising alternative in terms of its mathematical rigor (e.g., Luo et al., 1996; Lim, 2002). Yet MPEC is very hard to solve because of its non-convexity. The decoupled approach bypasses these difficulties by applying the following fact: once a sufficiently good set of observations is available, one can always find a set of paths being used at the deterministic user-equilibrium state. Those paths in turn can be used to construct a convex optimization program from which an O–D matrix can be inferred.

Because the path-link matrix **P** typically is not of full rank, the GLS-based PFE formulation involves inversion of a matrix whose strict inverse does not exist. In this research, a pseudoinverse of a non-invertible matrix, computed by a procedure based on singular value decomposition, is adopted to find a least squares solution (or minimum distance solution) for the corresponding linear system. Second, when paths are used as estimated variables, non-negativity constraints for path flows cannot be ignored, because many of these constraints are activated during estimation. Based on Lawson and Hanson's method (1974) for a standard non-negative least squares (NNLS) problem, we present an algorithm for the constrained PFE (ACPFE), which iterates consecutively between identifying the set of active constraints and solving the least squares problem. The ACPFE is further simplified in order to improve the computational efficiency while maintaining sufficient estimation accuracy.

The remaining of this paper is organized as follows. The following section reviews the ordinary least squares (OLS) model for proportion-based O–D matrix estimation. Section 3 develops the GLS path flow estimator that integrates the equilibrium assignment mapping and covariance matrices of measurement errors of link traffic counts and the target O–D matrix. Discussion of solution techniques for non-invertible linear systems and dealing with non-negative constraints are also covered in Section 3. Solution algorithms are presented in Section 4, and Section 5 reports numerical results. In the last section, we draw some conclusions and give directions for further research.

2. The OLS proportion-assignment O–D estimation formulation

We first assume that **P** is given as the link-use proportion matrix. The O–D estimation problem can be formulated as a classic ordinary least squares problem:

$$\min \frac{1}{2}(\bar{\mathbf{x}} - \mathbf{P}\mathbf{q})^*(\bar{\mathbf{x}} - \mathbf{P}\mathbf{q}); \quad \mathbf{P} \in R^{m\times o},\ \bar{\mathbf{x}} \in R^{m\times 1} \tag{3}$$

where * denotes the transpose here and in the remainder of this paper.

As shown by Trefethen and Bau (1997), a classic approach for solving problem (3) is to solve the following normal equation:

$$\mathbf{P}^*\mathbf{P}\mathbf{q} = \mathbf{P}^*\bar{\mathbf{x}} \tag{4}$$

System (4) can be solved by constructing a Cholesky factorization $\mathbf{P}^*\mathbf{P} = \mathbf{R}^*\mathbf{R}$, where **R** is upper-triangular.

Note that Problem (3) has two apparent shortcomings. First, as mentioned in Section 1, (3) may have multiple solutions. Second, the estimated solution can contain negative traffic demands, which is physically impossible. To address these two issues, we introduce explicit non-negativity constraints and add the "distance" between the estimated and target O–D matrices into the objective function:

$$\min \quad z(\mathbf{q}) = \frac{1}{2}(\bar{\mathbf{x}} - \mathbf{P}\mathbf{q})^*(\bar{\mathbf{x}} - \mathbf{P}\mathbf{q}) + \frac{1}{2}(\mathbf{q} - \mathbf{q_0})^*(\mathbf{q} - \mathbf{q_0}) \tag{5}$$

$$\text{s.t.}: \quad \mathbf{q} \geqslant 0 \tag{6}$$

where $\mathbf{q_0}$ denotes the target O–D matrix. Let $\boldsymbol{\mu}$ represent the Lagrange multipliers associated with constraints (6). We can formulate the Lagrangean function of the problem as follows:

$$L(\mathbf{q}, \mu) = z(\mathbf{q}) - \boldsymbol{\mu}^* \mathbf{q} \tag{7}$$

Since the original problem (5), (6) is a convex optimization program with linear constraints, it has a unique optimal solution. The necessary and sufficient conditions for its optimality can be stated explicitly according to Karush–Kuhn–Tucker (KKT) conditions:

$$\frac{\partial L}{\partial \mathbf{q}} = 0 \tag{8}$$

$$\frac{\partial L}{\partial \mu} \mu = 0, \quad \mu \geqslant 0 \tag{9}$$

From (8), it is easy to obtain the following relation:

$$(\mathbf{q} - \mathbf{q_0}) - (\bar{\mathbf{x}} - \mathbf{Pq})^* \mathbf{P} - \boldsymbol{\mu}^* = 0 \tag{10}$$

After taking transformation on both sides and collecting terms, we reach

$$(\mathbf{P}^*\mathbf{P} + \mathbf{I})\mathbf{q} = \mathbf{P}^*\mathbf{x} + \mathbf{q_0} + \boldsymbol{\mu} \tag{11}$$

Note that Eq. (11) is structurally similar to the normal equation (4). Following Bell (1991), we can derive an explicit expression for $\mathbf{q}$ using the Matrix Inversion Lemma:

$$(\mathbf{P}^*\mathbf{P} + \mathbf{I})^{-1} = \mathbf{I} - \mathbf{P}^*(\mathbf{PP}^* + \mathbf{I})^{-1}\mathbf{P} \tag{12}$$

Applying relation (12) in (11) and letting $\mathbf{PP}^* + \mathbf{I} = \mathbf{D}$, we get

$$\mathbf{q} = \mathbf{q_0} - \mathbf{P}^*\mathbf{D}^{-1}\mathbf{Pq_0} + (\mathbf{I} - \mathbf{P}^*\mathbf{D}^{-1}\mathbf{P})\mathbf{P}^*\bar{\mathbf{x}} + (\mathbf{I} - \mathbf{P}^*\mathbf{D}^{-1}\mathbf{P})\boldsymbol{\mu} \tag{13}$$

Furthermore, note that the following relation holds:

$$(\mathbf{I} - \mathbf{P}^*\mathbf{D}^{-1}\mathbf{P})\mathbf{P}^*\bar{\mathbf{x}} = \mathbf{P}^*\mathbf{D}^{-1}\bar{\mathbf{x}} \tag{14}$$

Now, putting (14) into Eq. (13), we obtain

$$\mathbf{q} = \mathbf{q_0} + \mathbf{P}^*\mathbf{D}^{-1}(\bar{\mathbf{x}} - \mathbf{Pq_0}) + (\mathbf{I} - \mathbf{P}^*\mathbf{D}^{-1}\mathbf{P})\boldsymbol{\mu} \tag{15}$$

The second and third terms on the right-hand side of Eq. (15) can be interpreted as two adjustments to the prior estimation (or survey results): the first is related to the difference between the observed traffic counts and those inferred on the basis of the prior estimates, and the second is adjusted according to the Lagrange multipliers due to the non-negativity requirement on O–D demands.

3. A decoupled GLS path flow estimator

The OLS model introduced in Section 2 is based on two assumptions: the measurement errors of observed traffic counts and the target O–D matrix are uncorrelated and have identical values, and the traffic assignment mapping matrix **P** is known a priori. These assumptions are relaxed in this section.

3.1. Introducing equilibrium assignment mapping

Use of a proportional assignment mapping for O–D matrix estimation under conditions of peak-time congestion in an urban transportation network will produce inconsistency in system (2). A conventional way of introducing the more appropriate equilibrium assignment mapping into the O–D estimation is to combine the user-equilibrium (UE) traffic assignment procedure and the estimation process as a bi-level problem. Because assignment and estimation are performed simultaneously in such a formulation, the computational overhead is substantially high even for medium-sized networks, and convergence is usually difficult to reach. A more desirable approach would be to maintain the simple convex structure of the original problem while taking the equilibrium assignment into consideration.

One promising approach along such lines is proposed by Nie and Lee (2002), who present an algorithm to solve the LPFE formulation of Sherali et al. (1994) for O–D estimation, where the equilibrium path flow pattern is determined by an exogenous K-shortest path ranking procedure. The K-shortest path ranking problem is a generalization of the shortest path tree problem, in which not one but a set of shortest paths is produced according to user-specified criteria. In general, if the UE condition holds exactly in a given network, for each O–D pair there exist more than one path having the smallest travel cost. Intuitively, performing a sequence of K-shortest path searches could pick up all these UE paths. Once available, these paths serve to form the coefficient matrix **P** whose columns correspond to the paths in the UE flow pattern. Since **P** has been obtained in accordance with the UE condition, the inconsistency from proportional assignment is eliminated while the computation is kept relatively simple. Consequently, this strategy "decouples" the equilibrium-based O–D estimation problem by determining the equilibrium assignment mapping **P** from the observed link traffic counts independently.

In this paper we apply this "decoupling" idea in a least squares framework instead of solving a linear programming problem as was done in Nie and Lee (2002). Note that an important difference in a PFE is that we do not estimate the O–D demand **q** directly. Instead, path flows are inferred and each O–D demand is derived by summing up flows along its associated equilibrium paths.

Let f_k^{rs} denote the flow on the kth equilibrium path between origin r and destination s, and $\delta_{a,k}^{rs}$ denotes the path-link incidence matrix, where $\delta_{a,k}^{rs}$ equals to 1 if link a belongs to path k, and 0 otherwise. We have the following expression:

$$\sum_r \sum_s \sum_k f_k^{rs}\delta_{a,k}^{rs} = \bar{x}_a \quad \forall a \in A,\ k \in K_{rs},\ r \in R,\ s \in S$$

or

$$\mathbf{\Delta f} = \bar{\mathbf{x}} \tag{16}$$

where $\mathbf{\Delta}$ plays a similar role as **P** in (2). However, $\mathbf{\Delta}$ is regarded to represent an equilibrium assignment mapping since its columns are generated so as to satisfy the UE condition. How to obtain $\mathbf{\Delta}$ from a K-shortest path ranking procedure will be discussed in the following section. At this point, let us assume $\mathbf{\Delta}$ is known. The ordinary least squares path flow estimator for O–D matrix estimation can then be formulated as follows:

Model [OLS–PFE]

$$\min \quad z(\mathbf{f}) = \frac{1}{2}(\bar{\mathbf{x}} - \mathbf{\Delta f})^*(\bar{\mathbf{x}} - \mathbf{\Delta f}) + \frac{1}{2}(\mathbf{Mf} - \mathbf{q_0})^*(\mathbf{Mf} - \mathbf{q_0}) \tag{17}$$

$$\text{s.t.}: \quad \mathbf{f} \geqslant 0 \tag{18}$$

where $\mathbf{M}$ is an $o \times n$ (n is the total number of paths) matrix which converts equilibrium path flows to O–D demands. Note constraint (18) guarantees

$$\mathbf{q} = \mathbf{Mf} \geqslant 0$$

Arguably, restricting $\mathbf{q}$ to be non-negative could be preferable to using (18) since negative path flows might not necessarily generate negative O–D demands. However, using $\mathbf{Mf} \geqslant 0$ as the constraints makes the problem much harder to solve, as we will show later. Because it has to conform to a UE state when optimality is reached, the estimated path flow pattern itself is an important output of a PFE model. Basing the final O–D estimation on a path flow pattern comprising negative path flows is not acceptable.

According to the first-order optimality condition, we get the following normal-type equation:

$$(\mathbf{\Delta}^*\mathbf{\Delta} + \mathbf{M}^*\mathbf{M})\mathbf{f} = \mathbf{\Delta}^*\bar{\mathbf{x}} + \mathbf{M}^*\mathbf{q_0} + \boldsymbol{\lambda} \tag{19}$$

where $\boldsymbol{\lambda}$ is the Lagrange multiplier vector associated with the constraint (18).

3.2. Considering correlated and Homoskedastic errors

We now deal with correlated and Homosekdastic random errors in the observed link counts and the target matrix by extending the OLS–PFE model to a generalized least squares (GLS) PFE model. In the following, observation errors of traffic counts are assumed to be small enough to guarantee that the K-shortest path ranking algorithm can produce a true UE assignment mapping.

A GLS–PFE model can be obtained by introducing the variance–covariance matrices of $\bar{\mathbf{x}}$ and $\mathbf{q_0}$. Let $\mathbf{S}$ and $\mathbf{T}$ represent the variance–covariance matrices for target matrix and traffic counts, respectively. [OLS–PFE] is reformulated as

Model [GLS–PFE]

$$\begin{aligned} &\min \quad z(\mathbf{f}) = \frac{1}{2}(\bar{\mathbf{x}} - \mathbf{\Delta f})^*\mathbf{T}^{-1}(\bar{\mathbf{x}} - \Delta f) + \frac{1}{2}(\mathbf{Mf} - \mathbf{q_0})^*\mathbf{S}^{-1}(\mathbf{Mf} - \mathbf{q_0}) \\ &\text{s.t.}: \quad \mathbf{f} \geqslant 0 \end{aligned} \tag{20}$$

where both $\mathbf{S}$ and $\mathbf{T}$ are positive definite symmetric matrices. The normal equation of model (20) can be derived from its first-order optimality condition as

$$(\mathbf{\Delta}^*\mathbf{T}^{-1}\mathbf{\Delta} + \mathbf{M}^*\mathbf{S}^{-1}\mathbf{M})\mathbf{f} = \mathbf{\Delta}^*\mathbf{T}^{-1}\bar{\mathbf{x}} + \mathbf{M}^*\mathbf{S}^{-1}\mathbf{q_0} + \boldsymbol{\lambda} \tag{21}$$

We refer to Eq. (21) as the fundamental equation for solving the decoupled GLS–PFE model. For the case in which the matrix $\mathbf{V} = \mathbf{\Delta}^*\mathbf{T}^{-1}\mathbf{\Delta} + \mathbf{M}^*\mathbf{S}^{-1}\mathbf{M}$ is invertible and all constraints are inactive, the solution to the fundamental equation is

$$\mathbf{f} = \mathbf{V}^{-1}\mathbf{b} \tag{22}$$

where $\mathbf{b} = \boldsymbol{\Delta}^{*}\mathbf{T}^{-1}\bar{\mathbf{x}} + \mathbf{M}^{*}\mathbf{S}^{-1}\mathbf{q}_{0}$.

However, both the path-link incidence matrix $\boldsymbol{\Delta}$ and the path-OD matrix $\mathbf{M}$ are generally neither full rank nor over-determined. To see this, first note that $\mathbf{M}$ becomes non-full rank and underdetermined whenever an O–D pair has more than one path, a condition common in a congested transportation network. Moreover, the number of paths available is often larger than that of observed link counts. Even if this is not the case, different paths still possibly correlate with one another so as to produce a non-full rank $\boldsymbol{\Delta}$. For such cases, $\mathbf{V}$ is non-invertible and Eq. (22) cannot be used to solve the fundamental equation (21).

3.3. Solving the fundamental equation with non-invertible V

Penrose (see Lawson and Hanson, 1974) showed that a "pseudoinverse" of a matrix $\mathbf{V}$, denoted by $\mathbf{V}^{+}$, which always exists, provides a unique minimum distance solution $\mathbf{V}^{+}\mathbf{b}$ (i.e., a solution for a least squares problem) for the linear system $\mathbf{V}\mathbf{f} = \mathbf{b}$. He proved that the matrix $\mathbf{X} = \mathbf{V}^{+}$ if and only if the following conditions hold:

$$\begin{aligned} \mathbf{VXV} &= \mathbf{V} \\ \mathbf{XVX} &= \mathbf{X} \\ (\mathbf{VX})^{*} &= \mathbf{VX} \\ (\mathbf{XV})^{*} &= \mathbf{XV} \end{aligned} \tag{23}$$

Therefore, in case of non-invertible $\mathbf{V}$, one has to resort to the pseudoinverse of $\mathbf{V}$ to obtain an approximate solution (minimum distance solution) for the fundamental equation. A classical approach to obtaining a pseudoinverse of $\mathbf{V}$ is to decompose $\mathbf{V}$ through singular value decomposition (SVD) as follows:

$$\mathbf{V} = \mathbf{W}\boldsymbol{\Sigma}\mathbf{U}^{*} \tag{24}$$

where both $\mathbf{W}$ and $\mathbf{U}$ are $n \times n$ unitary matrices, i.e., $\mathbf{W}^{*} = \mathbf{W}^{-1}$ and $\mathbf{U}^{*} = \mathbf{U}^{-1}$, and $\boldsymbol{\Sigma}$ is $n \times n$ and diagonal with non-negative entries. When the rank of $\mathbf{V}$ is $l < n$, we can rewrite $\boldsymbol{\Sigma}$ as

$$\boldsymbol{\Sigma} = \begin{bmatrix} \mathbf{S} & 0 \\ 0 & 0 \end{bmatrix}$$

where $\mathbf{S}$ is a $l \times l$ diagonal matrix whose diagonal entries are all positive. It is easy to prove, using Penrose's conditions, that the pseudoinverse of $\boldsymbol{\Sigma}$ is:

$$\boldsymbol{\Sigma}^{+} = \begin{bmatrix} \mathbf{S}^{-1} & 0 \\ 0 & 0 \end{bmatrix}$$

Consequently, the pseudoinverse of $\mathbf{V}$ is given by

$$\mathbf{V}^{+} = \mathbf{U}\boldsymbol{\Sigma}^{+}\mathbf{W}^{*} \tag{25}$$

To prove that this is true, we show that Eq. (25) satisfies all of the Penrose conditions:

$$\mathbf{V}\mathbf{V}^{+}\mathbf{V} = \mathbf{W}\boldsymbol{\Sigma}\mathbf{U}^{*}(\mathbf{U}\boldsymbol{\Sigma}^{+}\mathbf{W}^{*})\mathbf{W}\boldsymbol{\Sigma}\mathbf{U}^{*} = \mathbf{W}\boldsymbol{\Sigma}\boldsymbol{\Sigma}^{+}\boldsymbol{\Sigma}\mathbf{U}^{*} = \mathbf{W}\boldsymbol{\Sigma}\mathbf{U}^{*} = \mathbf{V}$$
$$\mathbf{V}^{+}\mathbf{V}\mathbf{V}^{+} = \mathbf{U}\boldsymbol{\Sigma}^{+}\mathbf{W}^{*}(\mathbf{W}\boldsymbol{\Sigma}\mathbf{U}^{*})\mathbf{U}\boldsymbol{\Sigma}^{+}\mathbf{W}^{*} = \mathbf{U}\boldsymbol{\Sigma}^{+}\boldsymbol{\Sigma}\boldsymbol{\Sigma}^{+}\mathbf{U}^{*} = \mathbf{U}\boldsymbol{\Sigma}^{+}\mathbf{W}^{*} = \mathbf{V}^{+}$$
$$(\mathbf{V}\mathbf{V}^{+})^{*} = (\mathbf{W}\boldsymbol{\Sigma}\mathbf{U}^{*}\mathbf{U}\boldsymbol{\Sigma}^{+}\mathbf{W}^{*})^{*} = (\mathbf{W}\boldsymbol{\Sigma}\boldsymbol{\Sigma}^{+}\mathbf{W}^{*})^{*} = \mathbf{W}\boldsymbol{\Sigma}\boldsymbol{\Sigma}^{+}\mathbf{W}^{*} = \mathbf{V}\mathbf{V}^{+}$$
$$(\mathbf{V}^{+}\mathbf{V})^{*} = (\mathbf{U}\boldsymbol{\Sigma}^{+}\mathbf{W}^{*}\mathbf{W}\boldsymbol{\Sigma}\mathbf{U}^{*})^{*} = (\mathbf{U}\boldsymbol{\Sigma}^{+}\boldsymbol{\Sigma}\mathbf{U}^{*})^{*} = \mathbf{U}\boldsymbol{\Sigma}^{+}\boldsymbol{\Sigma}\mathbf{U}^{*} = \mathbf{V}^{+}\mathbf{V}$$

Thus, a solution of Eq. (21) is given in form of the pseudoinverse as

$$\mathbf{f} = \mathbf{V}^{+}\mathbf{b} = \mathbf{U}\boldsymbol{\Sigma}^{+}\mathbf{W}^{*}\mathbf{b} \tag{26}$$

Codes for Computing the SVD of **V**, which is closely related to the eigenvalue decomposition of $\mathbf{V}^{*}\mathbf{V}$, are available in such software packages as MATLAB as a standard subroutine. We will not discuss the details of SVD computation in this paper; interested readers can refer to Trefethen and Bau (1997, Chapter 31, pp. 234–240).

3.4. Handling non-negative constraint

In applying GLS-based O–D estimation, it is often assumed that the non-negativity constraints can be ignored because they are rarely active (e.g., Yang, 1995). While this assumption might be appropriate if a compact link-use proportion (OD-link) matrix is adopted, employing path flow estimators significantly increases the possibility of activating non-negativity constraints since they operate directly upon path-link and path-OD matrices. Moreover, as the problem size increases so will the likely activation of more constraints, making it important to consider these non-negativity constraints in problems of realistic size.

Bell (1991) presented a simple algorithm for solving this constrained generalized least squares problem. His algorithm iterates between solving the fundamental equation and updating Lagrange multipliers by scaling negative estimated O–D entries with principal diagonal entries of $\mathbf{V}^{-1}$. Yet, apparently, there has been no reported implementation and/or computational experiments on Bell's algorithm in the literature. Furthermore, this algorithm appears not to work well for the GLS–PFE model since it does not consider non-invertible **V**.

Lawson and Hanson (1974), in presenting an algorithm for a standard non-negative least squares problem (NNLS), proved the following important theorem that describes the property of a NNLS solution:

Theorem 1. (Lawson and Hanson's Theorem) *The optimal solution* $\mathbf{f}^0$ *for the problem NNLS*

$$\begin{aligned} \min \quad & \frac{1}{2}(\mathbf{Pf}-\mathbf{x})^{*}(\mathbf{Pf}-\mathbf{x}) \\ \text{s.t.}: \quad & \mathbf{f} \geqslant 0 \end{aligned}$$

satisfies the following conditions simultaneously, given Φ and Ω as the index sets of active and inactive constraints respectively:

1. $f_i^0 := \begin{cases} > 0 & \text{if } i \in \Omega \\ = 0 & \text{if } i \in \Phi \end{cases}$
2. $\lambda_i := \begin{cases} = 0 & \text{if } i \in \Omega \\ > 0 & \text{if } i \in \Phi \end{cases}$ *where* $\boldsymbol{\lambda} = \mathbf{P}^{*}\mathbf{P}\mathbf{f}^{0} - \mathbf{P}^{*}\mathbf{x}$

3. $\mathbf{f}^0$ *is a solution vector for the following least squares problem*:

$$\min \frac{1}{2}(\mathbf{P}_\Omega \mathbf{f} - \mathbf{x})^*(\mathbf{P}_\Omega \mathbf{f} - \mathbf{x}) \quad \text{where } \mathbf{P}_\Omega := \begin{cases} \text{column } i \text{ of } \mathbf{P} & \text{if } i \in \Omega \\ 0 & \text{if } i \in \Phi \end{cases}$$

Note $\boldsymbol{\lambda}$ in the theorem denotes the Lagrange multipliers, and Conditions 1 *and* 2 *correspond to KKT conditions* (8) *and* (9)*. We shall show that this theorem can be easily extended to our decoupled PFE model.*

Theorem 2. *The optimal solution $\mathbf{f}^0$ for Model* [*OLS–PFE*] *satisfies the following conditions simultaneously, given Φ and Ω as the index sets of active and inactive constraints respectively*:

1. $f_i^0 := \begin{cases} > 0 & \text{if } i \in \Omega \\ = 0 & \text{if } i \in \Phi \end{cases}$
2. $\lambda_i := \begin{cases} = 0 & \text{if } i \in \Omega \\ > 0 & \text{if } i \in \Phi \end{cases}$ *where*

$$\boldsymbol{\lambda} = \mathbf{V}\mathbf{f}^0 - \mathbf{b}, \quad \mathbf{V} = \boldsymbol{\Delta}^*\boldsymbol{\Delta} + \mathbf{M}^*\mathbf{M}, \quad \mathbf{b} = \boldsymbol{\Delta}^*\bar{\mathbf{x}} + \mathbf{M}^*\mathbf{q_0} \tag{27}$$

3. $\mathbf{f}^0$ *is a solution vector for the following least squares problem*:

$$\min \frac{1}{2}(\boldsymbol{\Delta}_\Omega \mathbf{f} - \mathbf{x})^*(\boldsymbol{\Delta}_\Omega \mathbf{f} - \mathbf{x}) + \frac{1}{2}(\mathbf{M}_\Omega \mathbf{f} - \mathbf{q}_0)^*(\mathbf{M}_\Omega \mathbf{f} - \mathbf{q}_0)$$

where

$$\boldsymbol{\Delta}_\Omega := \begin{cases} \text{column } i \text{ of } \boldsymbol{\Delta} & \text{if } i \in \Omega \\ 0 & \text{if } i \in \Phi \end{cases} \quad \text{and} \quad \mathbf{M}_\Omega := \begin{cases} \text{column } i \text{ of } \mathbf{M} & \text{if } \quad i \in \Omega \\ 0 & \text{if } \quad i \in \Phi \end{cases} \tag{28}$$

Proof. We first show that [OLS–PFE] is equivalent to the following standard NLLS problem:

$$\begin{aligned} \min \quad & \frac{1}{2}(H\mathbf{f} - \pi)^*(H\mathbf{f} - \pi) \\ \text{s.t.}: \quad & \mathbf{f} \geqslant 0, \end{aligned} \tag{29}$$

where $H = \begin{bmatrix} \boldsymbol{\Delta} \\ \mathbf{M} \end{bmatrix}$ is an $(m + o) \times n$ matrix and $\pi = \begin{bmatrix} \bar{\mathbf{x}} \\ \mathbf{q_0} \end{bmatrix}$ is an $(m + o) \times 1$ matrix. To see this, note

$$\begin{aligned} (H\mathbf{f} - \pi)^*(H\mathbf{f} - \pi) &= \left(\begin{bmatrix} \boldsymbol{\Delta} \\ \mathbf{M} \end{bmatrix} \mathbf{f} - \begin{bmatrix} \bar{\mathbf{x}} \\ \mathbf{q_0} \end{bmatrix} \right)^* \left(\begin{bmatrix} \boldsymbol{\Delta} \\ \mathbf{M} \end{bmatrix} \mathbf{f} - \begin{bmatrix} \bar{\mathbf{x}} \\ \mathbf{q_0} \end{bmatrix} \right) \\ &= \mathbf{f}^*(\boldsymbol{\Delta}^*\boldsymbol{\Delta} + \mathbf{M}^*\mathbf{M})\mathbf{f} - \mathbf{f}^*(\boldsymbol{\Delta}^*\bar{\mathbf{x}} + \boldsymbol{\Delta}^*\mathbf{q_0}) - (\bar{\mathbf{x}}^*\boldsymbol{\Delta} + \mathbf{q_0^*}\mathbf{M})\mathbf{f} + (\bar{\mathbf{x}}^*\bar{\mathbf{x}} + \mathbf{q_0^*}\mathbf{q_0}) \\ &= (\mathbf{f}^*\boldsymbol{\Delta}^*\boldsymbol{\Delta}\mathbf{f} - \mathbf{f}^*\boldsymbol{\Delta}^*\bar{\mathbf{x}} - \bar{\mathbf{x}}^*\boldsymbol{\Delta}\mathbf{f} + \bar{\mathbf{x}}^*\bar{\mathbf{x}}) + (\mathbf{f}^*\mathbf{M}^*\mathbf{M}\mathbf{f} - \mathbf{f}^*\boldsymbol{\Delta}^*\mathbf{q_0} - \mathbf{q_0^*}\mathbf{M}\mathbf{f} + \mathbf{q_0^*}\mathbf{q_0}) \\ &= (\bar{\mathbf{x}} - \boldsymbol{\Delta}\mathbf{f})^*(\bar{\mathbf{x}} - \boldsymbol{\Delta}\mathbf{f}) + (\mathbf{M}\mathbf{f} - \mathbf{q_0})^*(\mathbf{M}\mathbf{f} - \mathbf{q_0}) \end{aligned}$$

Conditions 1 and 3 can be established directly from Theorem 1 and the equivalence between [OLS–PFE] and (29).

To prove that Condition 2 holds, note that Theorem 1 shows that the Lagrange multipliers of (29) can be denoted by

$$\boldsymbol{\lambda} = H^*H\mathbf{f}^0 - H^*\pi = [\boldsymbol{\Delta}^* \quad \mathbf{M}^*]\begin{bmatrix}\boldsymbol{\Delta}\\ \mathbf{M}\end{bmatrix}\mathbf{f}^0 - [\boldsymbol{\Delta}^* \quad \mathbf{M}^*]\begin{bmatrix}\bar{\mathbf{x}}\\ \mathbf{q_0}\end{bmatrix} = (\boldsymbol{\Delta}^*\boldsymbol{\Delta} + \mathbf{M}^*\mathbf{M})\mathbf{f}^0 - (\boldsymbol{\Delta}^*\bar{\mathbf{x}} + \mathbf{M}^*\mathbf{q_0})$$

This completes our proof. □

It is easy to see that Theorem 2 can be applied directly to Model [GLS–PFE]. Note that Theorem 1 cannot be extended easily to Theorem 2 if we take $\mathbf{Mf} \geqslant 0$ as constraints. This explains why constraints $\mathbf{f} \geqslant 0$ are used in our PFE model.

Theorem 2 implies that a constrained PFE model can be transformed into an unconstrained PFE model, which is much easier to solve, once the set of the active constraints is identified a priori. This sheds light on designing efficient algorithms for Model [GLS–PFE], which we address in detail in the following section.

4. Solution algorithms

In this section we present solution algorithms for the decoupled GLS path flow estimator. These include a K-shortest path ranking algorithm to determine matrices $\boldsymbol{\Delta}$ and $\mathbf{M}$ independently, and an iterative procedure to derive non-negative equilibrium path flows based on the least squares method.

4.1. An algorithm to generate the equilibrium mapping

As mentioned in the previous section, the critical idea behind the decoupling method is to use an exogenous K-shortest path ranking procedure to generate the UE assignment mapping. The K-shortest paths ranking is a classic network programming problem of determining not only the shortest path, but also the second, the third, …, to the kth (for a given integer $k > 1$) shortest path. It was used in Nie and Lee (2002) to recognize paths that reproduce the user-equilibrium flow pattern, so as to determine matrices $\boldsymbol{\Delta}$ and $\mathbf{M}$ in (20). The K-shortest path ranking algorithm presented in Nie and Lee (2002) is based on Eppstein's approach (1999), which picks shortest paths from a graph representing all possible deviations from a standard shortest path tree. However, Eppstein's method was originally designed for unconstrained shortest path ranking and allows cycles of the repeated nodes in the generated paths. Nie and Lee (2002) incorporated a heuristics procedure into Eppstein's algorithm that guarantees the generated paths to be cyclic-free and cost-identical. In this paper we focus on procedures of generating matrices $\boldsymbol{\Delta}$ and $\mathbf{M}$. Readers may refer to Eppstein (1999) for details of the K-shortest path ranking algorithm.

The complete algorithmic steps to obtain $\mathbf{\Delta}$ and $\mathbf{M}$ are summarized as follows:

Step 0: Initialization. Set the current O–D pair $k = 0$ and path number $l = 0$.
Step 1: Let $k = k + 1$, if $k > o$, stop, the total number of paths $m = l$. r and s denotes the origin and destination corresponding to O–D pair k.
Step 2: Calculate the shortest path tree T_r^* rooted at r.
Step 3: Build the pseudo path tree that contains all cost-identical (in terms of the tolerant error specified by users) shortest paths.
Step 4: Get paths and update matrices $\mathbf{\Delta}$ and $\mathbf{M}$.
  Step 4.1: Use a backward tracking procedure to obtain a cyclic-free path p^{rs}. If no path can be found, go to Step 1; otherwise Set $l = l + 1$.
  Step 4.2: Set $\mathbf{\Delta}(:, l) = p^{rs}$ and $\mathbf{M}(k, l) = 1$. Go to Step 4.1.

4.2. Algorithms for estimating non-negative equilibrium path flows

Once equilibrium assignment maps, matrices $\mathbf{\Delta}$ and $\mathbf{M}$, are available, we can then estimate the corresponding path flows, as well as O–D traffic demands, by solving the constrained PFE model. Theorem 2 shows that the optimal solution to this constrained GLS problem depends heavily on the set of active constraints. Since specifying these constraints a priori is impossible, we apply iterative methods in which the solution to the unconstrained GLS problem and identification of the active constraints are carried out iteratively.

In the following, we first present an algorithm for solving the constrained PFE model based on Lawson and Hanson's iterative method for a standard NNLS problem:

Algorithm for constrained PFE (ACPFE):

Step 0: Initialization. Set $\Phi = \phi$, $\Omega = \{1, 2, \ldots, n\}$ and $\mathbf{f} = 0$.
Step 1: Compute the Lagrange multipliers $\boldsymbol{\lambda}$ according to (27). If $\Omega = \phi$ or $\lambda_i \geqslant 0$ for all $i \in \Omega$, stop.
Step 2: Move the index $t = \arg\min_k\{\lambda_k, k \in \Omega\}$ from Ω to Φ.
Step 3: Set $\mathbf{\Delta}$ and $\mathbf{M}$ according to (28).
Step 4: Compute $\mathbf{f}^t = \mathbf{V}^+\mathbf{b}$ using SVD decomposition (25), then set $f_i^t = 0$ if $i \in \Phi$, where $\mathbf{V} = \mathbf{\Delta}^*\mathbf{T}^{-1}\mathbf{\Delta} + \mathbf{M}^*\mathbf{S}^{-1}\mathbf{M}$ and $\mathbf{b} = \mathbf{\Delta}^*\mathbf{T}^{-1}\bar{\mathbf{x}} + \mathbf{M}^*\mathbf{S}^{-1}\mathbf{q_0}$.
Step 5: If $f_i^t > 0$ for all $i \in \Omega$, set $\mathbf{f} = \mathbf{f}^t$ and go to Step 1; otherwise, go to Step 6.
Step 6: Set $\mathbf{f} = \mathbf{f} + \alpha(\mathbf{f}^t - \mathbf{f})$, where $\alpha = \min\left\{\frac{f_i}{f_i - f_i^t} : f_i^t \leqslant 0, i \in \Phi\right\}$
Step 7: Move all indices for which $f_i = 0$ from Ω to Φ, go to Step 3.

The convergence of ACPFE has been proven in Lawson and Hanson (1974). However, the algorithm is computationally cumbersome even for medium-size problems since the number of major iterations, in which an unconstrained GLS problem has to be solved, is bounded from below by the number of non-active constraints. Experience has shown that the number of non-active constraints can be substantially large in a real-size estimation problem, which makes the computational overhead of the algorithm prohibitively high. In order to overcome this computational difficulty, a heuristic algorithm is designed to decrease the number of iterations in ACPFE while ensuring that the solution satisfies the conditions given in Theorem 2. We describe this simplified ACPFE (SACPFE) as follows:

Step 0: Initialization. Set $\Phi = \phi$, $\Omega = \{1, 2, \ldots, n\}$ and $\mathbf{f} = \mathbf{0}$.
Step 1: Set $\mathbf{\Delta}$ and $\mathbf{M}$ according to (28).
Step 2: Compute $\mathbf{f} = \mathbf{V}^{+}\mathbf{b}$ using SVD decomposition (25).
Step 3: If $f_i \geqslant 0$ for all $i = 1, 2, \ldots, n$, stop; otherwise, move all indices for which $f_i < 0$ from Ω to Φ, go to Step 1.

Note that this algorithm will terminate after a limited number of iterations since all f_i associated with active constraints will be set to zero compulsorily when $\mathbf{\Delta}$ and $\mathbf{M}$ are changed according to (28); i.e., if a constraint is activated in some iteration, it will never become inactive later. The simplified ACPFE can dramatically decrease the number of iterations required by the original method, as our numerical experiments will show later. However, it is possible that the algorithm may not correctly estimate the set of active constraints, and thereby may jeopardize the solution accuracy. Intuitively, activating some constraints can inadvertently cause other active constraints to become inactive, which cannot be detected precisely by SACPFE. This notwithstanding, the simplified algorithm can produce an approximate optimal path solution that is comparable to ACPFE in reproducing UE link flow pattern and driving estimated O–D traffic matrix to approach a real one. From a practical point of view, the considerable gain in computational efficiency is judged to outweigh the modest degradation in solution accuracy.

We end this section by noting that the algorithms presented here require that all link counts be available and of sufficient accuracy to produce a UE path set. Such a requirement is not often realizable in real systems. This requirement could be relaxed either by applying the flow conservation principle and prior information (as suggested by Sherali et al., 1994), or by introducing an outer loop that consecutively updates link flows as well as link travel times according to estimated path flow patterns. Options incorporating these improvements will be investigated subsequently in a separate paper.

5. Numerical results

In this section, we give numerical results to demonstrate how the algorithms presented in the previous section work using small- to medium-size networks. The cyclic-free Eppstein's K-shortest path ranking algorithm was implemented on a PC using VC++ 6.0, while the GLS path flow estimator algorithm was implemented on a PC using MATLAB 6.0.

In all simulation experiments, "observed" traffic counts are generated from a standard traffic assignment procedure that allocates real O–D traffic demands onto a given network such that a UE flow pattern is produced. For simplicity, we assume that the traffic counts generated from TAP subject to error whose covariance matrix $\mathbf{T} = \mathbf{I}$, where $\mathbf{I}$ is an identity matrix. Link travel time, from which the K-shortest path ranking algorithm recasts the equilibrium paths, is assumed to be dependent only on the traffic flow on that link; i.e., link interactions are ignored. In this paper, link travel time is computed from the standard BPR (Bureau of Public Road) function, which takes the following form:

$$t = t_0\left(1 + 0.15\left(\frac{x}{C}\right)^4\right) \tag{30}$$

where t_0 denotes the free flow travel time, C is the link capacity and x denotes the traffic volume/counts.

In order to obtain a target O–D demand matrix, an artificial deviation is applied to the real O–D matrix that is used for producing "observed" link counts. There are three types of target matrices being tested in each example: an error-free (EF) target matrix that is exactly equivalent to the real matrix, a weak-prior-information (WPI) matrix whose O–D entries take the same values of demands crossing the same origin (total demands from the origin divided by its associated number of O–D pairs), and a strong-prior-information (SPI) matrix whose O–D entries are all changed in small proportions (<30%).

To specify the variance–covariance matrix of $\boldsymbol{q}$, following Brenninger-Göthe et al. (1989), we introduce a weighting parameter γ that depicts the relative belief in the traffic counts compared to the target matrix.. The covariance matrix $\mathbf{S}$ is set as a diagonal matrix with principal diagonal elements equivalent to the inverse of γ. γ can take values between 0 and 1, which means that the target matrix is at most regarded as accurate as the observed link counts, when the covariance matrix of target demands $\boldsymbol{S} = \boldsymbol{I}$. This is a reasonable assumption because the model will make no improvement upon the target matrix by using link counts that are more inaccurate than the given target itself. In our experiments, we use the following heuristic to calculate γ:

$$\gamma = \min\left(1, \frac{o}{\|\bar{\mathbf{q}} - \mathbf{q}_0\|^2}\right) \tag{31}$$

where $\mathbf{q}_0$ denotes the real O–D matrix.

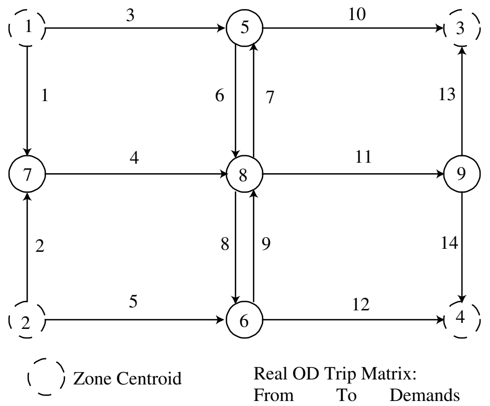
Note that Eq. (31) cannot be applied in practical situations since it requires information pertaining to the "real" O–D, which is regarded as unknown and whose value is precisely the goal of the estimation.

5.1. Yang's 9-node network

The first test example, which contains 9 nodes, 14 links and 4 O–D pairs, is taken from Yang (1995). Fig. 1 depicts the network topology and the real O–D matrix. The "observed" link traffic counts together with the travel times, which are produced by a UE traffic assignment subroutine, are reported in Table 1.

Eight equilibrium paths (see Table 2) are picked up from the K-shortest path ranking algorithm, with a tolerance error of 0.001% (all cyclic-free paths whose costs is less than or equal to 1.00001 times of the shortest path cost will be regarded as identical to the shortest one).

Table 3 provides: the estimated O–D matrix; root means square error (RMSE) between the estimated and the real matrix (RMSE$_q$); and RMSE between the estimated link traffic flows and the



Real OD Trip Matrix:

From	To	Demands
1	3	200
1	4	150
2	3	140
2	4	185

Fig. 1. Topology and real O–D trip matrix of Yang's network.

Table 1
Input data for Yang's network

Index	From	To	Capacity	"Observed" travel time	"Observed" traffic counts
1	1	5	250	13.18	225.03
2	1	7	150	4.29	124.97
3	2	6	250	11.49	185.00
4	2	7	150	4.46	140.00
5	5	3	250	13.24	227.67
6	5	8	150	3.00	25.03
7	6	4	250	12.16	228.85
8	6	8	150	5.00	0.00
9	7	8	250	11.89	264.97
10	8	5	150	4.00	27.67
11	8	6	150	4.00	43.85
12	8	9	250	13.05	218.48
13	9	3	150	4.19	112.33
14	9	4	150	3.11	106.15

observed link counts (RMSE$_x$), based on algorithms ACPFE and SACPFE, for the three different target matrices described in the preceding section. The equilibrium path solution, together with Lagrange multipliers, is reported in Table 4.

When an error-free (EF) target matrix is applied ($\gamma = 1$), both RMSE$_q$ and RMSE$_x$ are observed to be nearly zero, which implies that in this case the real O–D matrix obtained from our decoupled path flow estimator and the user-equilibrium link flow pattern is reproduced perfectly. As shown in Tables 3 and 4, the algorithm ACPFE took seven iterations to identify the second constraint, whose associated Lagrange multipliers are zero, as being active. Alternatively, SACPFE took only one iteration to terminate, but with a different path flow

Table 2
Equilibrium path flow pattern of Yang's network

O–D index	From	To	Path index	Path cost	Path track
1	1	3	1	26.42	$1 \to 5 \to 3$
2	1	4	2	32.34	$1 \to 5 \to 8 \to 9 \to 4$
			3	32.34	$1 \to 7 \to 8 \to 9 \to 4$
			4	32.24	$1 \to 5 \to 8 \to 6 \to 4$
			5	32.24	$1 \to 7 \to 8 \to 6 \to 4$
3	2	3	6	33.59	$2 \to 7 \to 8 \to 9 \to 3$
			7	33.59	$2 \to 7 \to 8 \to 5 \to 3$
4	2	4	8	23.65	$2 \to 6 \to 4$

Table 3
Estimated results for Yang's network

Algorithms	Estimated O–D matrices				RMSE_q	RMSE_x	Iteration	CPU time
	q_{13}	q_{14}	q_{23}	q_{24}				
EF target	200	150	140	185	0	–		
ACPFE	200.00	150.00	140.00	185.00	0.00	0.00	7	0.15(0.09)[a]
SACPFE	200.00	150.00	140.00	185.00	0.00	0.00	1	0.21(0.13)
WPI target	175	175	162.5	162.5	23.78	–		
ACPFE	199.69	150.23	140.11	184.81	0.22	0.22	7	0.15(0.09)
SACPFE	199.69	150.23	140.11	184.81	0.22	0.22	1	0.14(0.09)
SPI target	180	135	125	160	19.20	–		
ACPFE	199.88	150.00	139.98	184.86	0.09	0.12	7	0.22(0.13)
SACPFE	199.88	150.00	139.98	184.86	0.09	0.12	1	0.15(0.10)

[a] The number in parentheses is the CPU time consumed by K-shortest path searching.

Table 4
Equilibrium path solutions and Lagrange multipliers for Yang's network

Path index	EF target				WPI target				SPI target			
	ACPFE		SACPFE		ACPFE		SACPFE		ACPFE		SACPFE	
	Flow	λ	Flow	λ	Flow	λ	Flow	λ	Flow	λ	Flow	λ
1	200.00	0.00	200.00	0.00	199.69	0.00	199.69	0.00	199.88	0.00	199.88	0.00
2	0.00	0.00	25.03	0.00	0.00	0.00	25.25	0.00	0.00	0.00	25.06	0.00
3	106.15	0.00	81.12	0.00	106.23	0.00	80.98	0.00	106.13	0.00	81.07	0.00
4	25.03	0.00	0.00	0.00	25.25	0.00	0.00	0.00	25.06	0.00	0.00	0.00
5	18.82	0.00	43.85	0.00	18.75	0.00	44.00	0.00	18.82	0.00	43.87	0.00
6	112.33	0.00	112.33	0.00	112.29	0.00	112.29	0.00	112.30	0.00	112.30	0.00
7	27.67	0.00	27.67	0.00	27.82	0.00	27.82	0.00	27.68	0.00	27.68	0.00
8	185.00	0.00	185.00	0.00	184.81	0.00	184.81	0.00	184.86	0.00	184.86	0.00

pattern in which the fourth constraint is active, with the associated Lagrange multiplier being zero.

For the weak-prior-information (WPI) target matrix, whose RMSE from the real matrix is about 23.78, the relative belief parameter γ is calculated according to (30) as $1/23.78^2 \approx 0.002$. Both ACPFE and its simplified version generated estimates that are much closer to the real O–D matrix than the given target ($\text{RMSE}_q = 0.22$), indicating that the target matrix can be improved substantially by applying traffic count information. At the same time, the observed link counts are regenerated accurately from estimation, with RMSE_x of 0.22.

A similar observation is made for the case in which the strong-prior-information (SPI) target matrix was adopted: RMSE_q of the O–D matrix estimated (0.09) is much less than that of SPI target (19.20), while the user-equilibrium flow pattern is reproduced with relatively good accuracy ($\text{RMSE}_x = 0.12$).

We note that, even though ACPFE and SACPFE came up with different path flow patterns, the two algorithms generated nearly the same link flow pattern and estimated O–D matrix for the three target matrices,. This is a positive indication that SACPFE can offer satisfying solutions for the proposed decoupled PFE model. As expected from theoretical considerations, in this example, ACPFE needed many more iterations to converge than did the proposed simplified algorithm. CPU times consumed by the two algorithms are not significantly different in this example (Table 4), principally because of the small size of the network. Moreover, more than half of the CPU time has been spent on finding equilibrium paths and matrices $\mathbf{\Delta}$ and $\mathbf{M}$ in both algorithms, suggesting that the K-shortest path ranking is a computationally expensive module. However, these performance observations are due to the small size of this particular example. As we will see in the following section, the proportion of CPU time taken by K-shortest path ranking drops steeply as network size increases, and the computational merit of the simplified algorithm accelerates when the network size becomes reasonably large.

5.2. A 100-node hypothetical network

This example presents computational experiments using a larger network, obtained from a random grid-network generator. A 10 by 10 hypothetical network was generated in such a manner

Table 5
Estimated results for the hypothetical network

Algorithms	RMSE_q	RMSE_x	Iteration	CPU time	Active constraints
EF Target	0	–			
ACPFE	0.00	0.00	157	215.73 (1.74)	158
SACPFE	0.00	0.00	3	10.20 (1.85)	90
WPI target	200.68	–			
ACPFE	67.03	0.65	173	253.73 (1.75)	156
SACPFE	67.03	0.65	4	10.45 (1.85)	86
SPI target	128.88	–			
ACPFE	34.69	0.30	177	260.17 (1.74)	153
SACPFE	34.69	0.30	4	10.36 (1.83)	83

that its nodes are arranged in a square planar grid with a grid arc (whose link performance function is also randomly determined) connecting each pair of adjacent grid nodes in each direction. A node at a corner has two successors while a node at an edge has three. All interior nodes have four

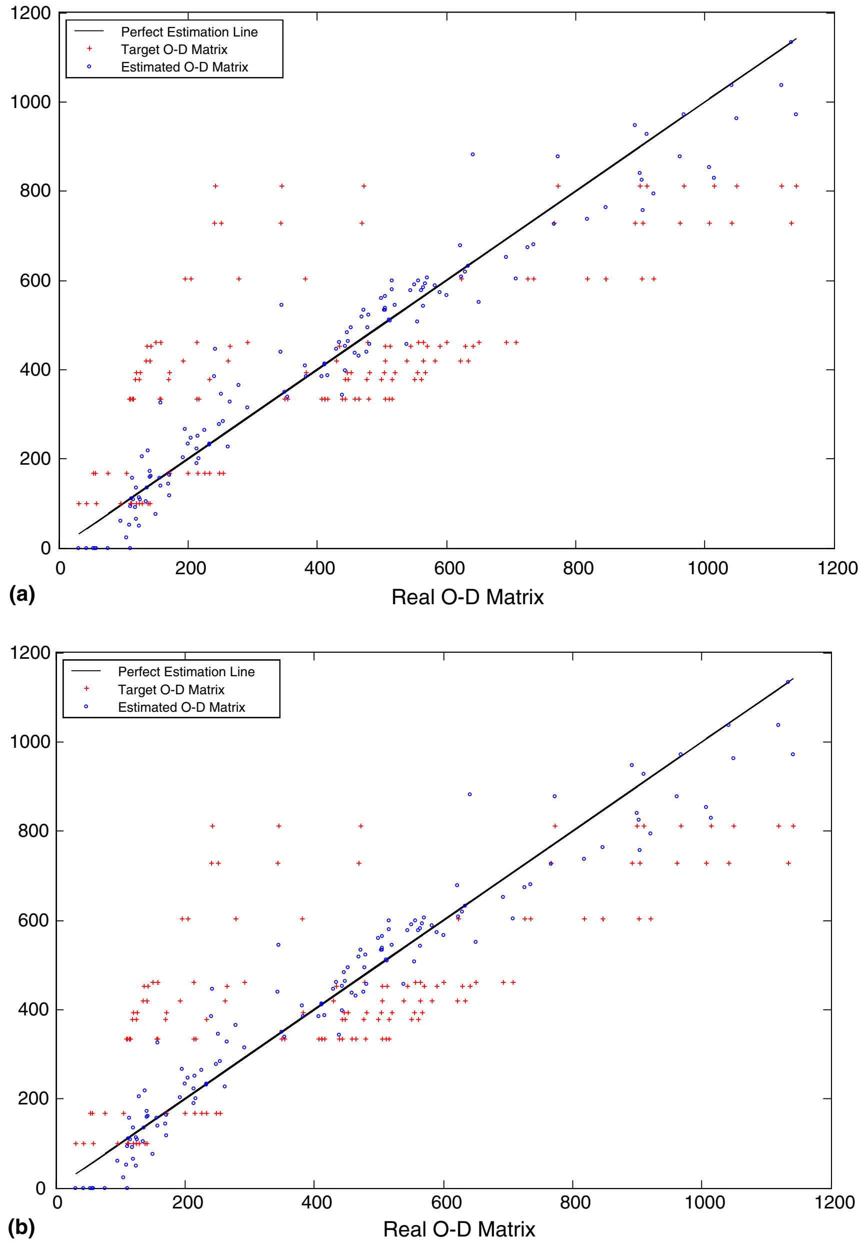


Fig. 2. WIP target vs. estimated O–D matrix for hypothetical network: (a) comparison based on Algorithm ACPFE; (b) comparison based on Algorithm SACPFE.

successors. Twelve nodes were chosen as zone centroids; that is, a 12 by 11 matrix of random elements was produced as the real O–D trip matrix.

The K-shortest path ranking procedure found 312 paths contained in the UE flow pattern, given a tolerance error of 0.01%. Estimated results for the three target matrices are given in Table 5.

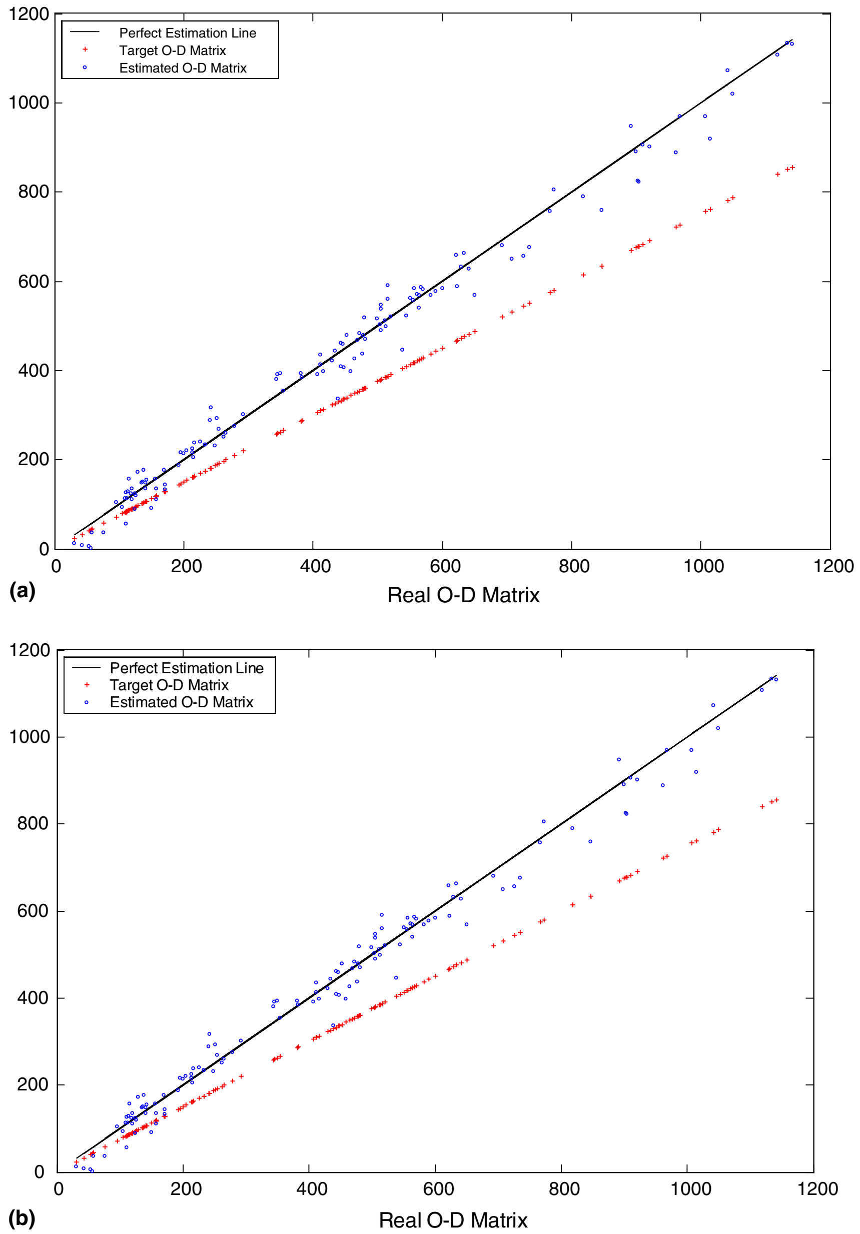


Fig. 3. SPI target vs. estimated O–D matrix for hypothetical network: (a) comparison based on Algorithm ACPFE; (b) comparison based on Algorithm SACPFE.

When an error-free O–D matrix is used as the target, the algorithms regenerate the observed link counts precisely and obtain an estimation that is exactly equivalent to both target and real O–D matrices ($\text{RMSE}_q = \text{RMSE}_x = 0$, although the optimal path solutions from the two algorithms are quite different. ACPFE recognized 158 active constraints when optimality is reached while the simplified algorithm identified only 90. However, the two distinctive path flow patterns produced quite similar link flows as well as O–D estimates, once again demonstrating that SACPFE maintains the accuracy of ACPFE while dramatically improving the computational performance of ACPFE.

Fig. 2(a) and (b) visualize the WPI target and estimated O–D matrices for ACPFE and SACPFE, respectively. The x axis of the plot represents the real O–D traffic demands while the y axis represents the target demands (denoted by cross) or estimated demands (denoted by circle). A perfect estimation, that is, one in which estimated demands equal real demands, should follow a straight line with slope 1, which we refer to as the perfect estimation line hereafter. An immediate observation from these two figures is that ACPFE and SACPFE have produced nearly identical O–D matrix estimates. Furthermore, the estimated demands from both algorithms have an apparent tendency to approach the perfect estimation line compared to the target demands pattern, also reflected in the RMSE_q of the estimated matrix (67.03), which is much smaller than that of the WPI target (200.68).

We obtained the SPI target matrix in this example by uniformly scaling down all real O–D demands by 25%. This produced a SPI target with RMSE_q of 128.88. The estimated O–D matrices obtained from ACPFE and SACPFE significantly reduced the RMSE_q of SPI target to about 34.69. The improvement in estimated O–D matrices upon the SPI target is shown in Fig. 3(a) and (b), which clearly demonstrate that the proposed algorithms drive the SPI target toward the perfect estimation line with link counts information.

In this example, the simplified method outperforms ACPFE significantly in terms of computational overhead for three tested scenarios. Taking the WPI target case as an example, SACPFE spent only 4 iterations and 10.45 seconds CPU time to converge while ACPFE took about 173 iterations and 253.73 seconds CPU time. Note that SACPFE cannot obtain as accurate an optimal path solution as does ACPFE since it likely fails to recognize all active constraints. Nonetheless, SACPFE estimates of O–D matrices and link counts are judged to be reasonably precise.

CPU time consumed by K-shortest path ranking in this example becomes relatively insignificant compared to time spent in the least squares component: it takes around 18% of total CPU time for SACPFE, and only 0.8% for ACPFE, to find all equilibrium paths. We conclude that the K-shortest path ranking is not a computational bottleneck in solving the overall decoupled GLS-based PFE model.

6. Conclusions

This paper combines a decoupled path flow estimator with generalized least squares techniques for equilibrium-based O–D trip matrix estimation. Through the GLS structure, the formulation incorporates measurement errors of link counts and target matrix naturally, and maintains the major advantage of the decoupled PFE model, that is, simplifying the work of determining the equilibrium assignment mapping by exogenously identifying the optimal paths conforming to a user-equilibrium state.

The solution to our GLS–PFE model differs from previous GLS-based models in two aspects: removal of the required solution of a non-invertible linear system and the imposition of non-negativity constraints. The algorithm ACPFE, based on Lawson and Hanson's method, was developed to solve the GLS–PFE model. The algorithm continuously solves a restricted GLS problem through singular value decomposition and determines the set of active constraints. To overcome the relatively slow convergence of the ACPFE solution algorithm due to its one-at-a-time update strategy for the active constraint set, an all-at-a-time update strategy (SACPFE), which simply marks all constraints that are violated in an estimation to be active, was developed.

Our limited numerical experiments lead to the following findings:

1. When solved by either ACPFE or its simplified version, the GLS–PFE model can substantially improve the given target O–D matrix (i.e., producing an estimate that is much closer to the real O–D matrix than the target) after using link traffic counts, while producing a path flow pattern which concurs with a UE state.
2. Although SACPFE may generate optimal path solutions and active constraint sets that are quite different from those of ACPFE, these different path solutions nonetheless produce quite similar estimates for the O–D matrix and link traffic counts. Given its computational advantage, it is recommended that SACPFE be considered in solving large-scale O–D matrix estimation problems in congested networks.
3. The computational cost of finding equilibrium paths is fairly modest when problem size is reasonably large.

Further work need to be done to relax the assumption that all link traffic counts are available. It would also be interesting to investigate the statistical characteristics of the GLS-based path flow estimator when many constraints are active, as well as to study the effects of the relative belief parameter γ on estimation results.

Acknowledgements

This research was partially funded by the California PATH program under the project ATMIS-Task-Order-4135. We also thank two anonymous referees and Professor Fred Mannering for their suggestions to improve the paper. The views expressed in this paper are the authors alone.

References

Bell, M., 1991. The estimation of origin–destination matrices by constrained generalized least squares. Transportation Research B 25, 13–22.

Bell, M.G.H., Iida, Y., 1997. Transportation Network Analysis. John Wiley, New York.

Bell, M.G.H., Shield, C.M., Busch, F., Kruse, K., 1997. A stochastic user equilibrium path flow estimator. Transportation Research C 5, 197–210.

Brenninger-Göthe, M., Jörnsten, K.O., Lundgren, J.T., 1989. Estimation of origin–destination matrices from traffic counts using multi-objective programming formulations. Transportation Research B 23, 257–269.

Cascetta, E., 1984. Estimation of trip matrices from traffic counts and survey data: a generalized least squares estimator. Transportation Research B 18, 289–299.

Eppstein, D., 1999. Finding the K shortest paths. SIAM Journal of Computing 28 (2), 652–673.

Fisk, C.S., Boyce, D.E., 1983. A note on trip matrix estimation from link traffic count data. Transportation Research B 17, 245–250.

Lawson, C., Hanson, R., 1974. Solving Least Squares Problems. Prentice-Hall, Englewood Cliffs, NJ.

Lim, A.C., 2002. Transportation network design problems: an MPEC approach, Ph.D. Thesis, Johns Hopkins University.

Lo, H.P., Zhang, N., Lam, W.H.K., 1996. Estimation of an origin–destination matrix with random link choice proportions: a statistical approach. Transportation Research B 30, 309–324.

Luo, Z.-Q., Pang, J.-S., Ralph, D., 1996. Mathematical Programs with Equilibrium Constraints. Cambridge University Press, Cambridge, UK.

Nguyen, S., 1977. Estimating an OD matrix from network data: a network equilibrium approach. University of Montreal Publication, No. 60.

Nie, Y., Lee, D.-H., 2002. An uncoupled method for the equilibrium-based linear path flow estimator for origin–destination trip matrices. Transportation Research Record 1783, 72–79.

Sherali, H.D., Sivanandan, R., Hobeika, A.G., 1994. A linear programming approach for synthesizing origin–destination trip tables from link traffic volumes. Transportation Research B 28, 213–233.

Trefethen, L., Bau, D., 1997. Numerical Linear Algebra. SIAM, Philadelphia, PA.

Van Zuylen, J.H., Willumsen, L.G., 1980. The most likely trip matrix estimated from traffic counts. Transportation Research B 14, 281–293.

Wardrop, J.G., 1952. Some theoretical aspects of road traffic research. Proceedings of the Institute of Civil Engineers, Part II 1, 325–378.

Willumsen, L.G., 1981. Simplified transport models based traffic counts. Transportation 10, 257–278.

Yang, H., Sasaki, T., Iida, Y., Asakura, Y., 1992. Estimation of origin–destination matrices from link traffic counts on congested networks. Transportation Research B 26, 417–434.

Yang, H., 1995. Heuristic algorithms for the bi-level origin–destination matrix estimation problem. Transportation Research B 29, 231–242.