Linear Programming Formulation of the Elevator Trip Origin-destination Matrix Estimation Problem

Juha-Matti Kuusinen, Mirko Ruokokoski, Janne Sorsa and Marja-Liisa Siikonen
KONE Technology, KONE Corporation, POB 7, 02151 Espoo, Finland

Keywords: Elevator traffic, Origin-destination Matrix, Linear Programming, Branch-and-bound.

Abstract: Elevator group control dispatches elevators to passengers' calls in a dynamic environment where new calls constantly emerge. At the moment of making a dispatching decision, it is not known when and at which floors new passengers will register new calls, what is the number of passengers waiting behind these and existing calls, and what are their destinations. Robust dispatching decisions require that future passenger traffic is forecast based on the realized passenger flow in a building. The problem is that this flow cannot be directly measured. It can, however, be estimated by finding the passenger counts for the origins and destinations of every elevator trip occurring in a building. An elevator trip consists of successive stops in one direction of travel with passengers inside the elevator. We formulate the elevator trip origin-destination matrix estimation problem as a minimum cost network flow problem. We also present a branch-and-bound algorithm for finding all solutions to the problem and study its performance based on numerical experiments.

1 INTRODUCTION

Assume that when an elevator stops, the passengers inside the elevator car first alight and then the possibly waiting passengers board the elevator. Then, an elevator trip to up or down direction starts at a stop where passengers board an empty elevator and ends to a stop where the elevator becomes empty again. At each stop, the elevator group control registers the number of boarding and alighting passengers, and the calls given by the boarding passengers.

Our goal is to estimate elevator trip origin-destination (OD) matrices, i.e., the passenger counts for the OD pairs of the elevator trips. The elevator trip OD matrices estimated for a given time interval are combined to form a building OD matrix that describes the traffic flow in the whole building during that interval. The building OD matrices form traffic statistics that are used to forecast future traffic. The required information is: what is the number of passengers waiting behind each existing and new call at the time of serving the call, when and at which floors new calls are likely to occur, and what are the destinations of the passengers. These forecasts are needed in elevator dispatching to make robust call allocation decisions in constantly changing traffic conditions.

An elevator trip is analogous to a single transit route such as a bus line where there is at most one route connecting any OD pair, and usually counts of boarding and alighting passengers are collected from all stops on the route (Nguyen, 1984). The passenger counts for the OD pairs of the route are typically estimated by minimizing the distance to a reference OD matrix with respect to a suitable distance metric while also producing the observed boarding and alighting counts. Popular OD matrix estimation methods are iterative proportional fitting which is known also as the bi-proportional or Bregman's balancing method (Lamond and Stewart, 1981; Bell, 1983; Nguyen, 1984; Ben-Akiva et al., 1985), maximum entropy and minimum information (Zuylen and Willumsen, 1980; Nguyen, 1984; Ben-Akiva et al., 1985), maximum likelihood and generalized least squares (Ben-Akiva et al., 1985; Cascetta and Nguyen, 1988), Bayesian inference (Maher, 1983; Cascetta and Nguyen, 1988; Li, 2009), recursive methods (Tsygalnitsky, 1977; Furth and Navick, 1992; Li and Cassidy, 2007) and bi-level programming formulations (Fisk, 1988; Lundgren and Peterson, 2008).

An elevator trip OD matrix could in principle be estimated with these methods but some basic differences between an elevator trip and a single transit route prevent their use. A single transit route is typically defined in advance and is not often changed. Hence, usually a single OD matrix is estimated for a certain time period using the corresponding reference OD matrix and boarding and alighting counts observed during that period. An elevator trip is request driven, which means that each elevator trip has its own set of OD pairs defined by the calls given by the boarding passengers. Hence, we need to estimate a separate OD matrix for each elevator trip. A natural requirement is then that the estimated OD passenger counts are integer-valued. The above methods do not in general meet this condition. In addition, when elevators are installed in a new building, the reference matrix is not available and the elevator trip OD matrices have to estimated based only on the observed boarding and alighting counts.

We model the elevator trip OD matrix estimation problem as a minimum cost network flow problem. This model can be used to solve the problem when the measured boarding and alighting counts are consistent taking into account also lower bounds on the OD passenger counts. When the counts are consistent, the result of minimizing the distance between the boarding and alighting counts produced by the estimated elevator trip OD matrix and the observed boarding and alighting counts does not depend on the distance metric. The minimum cost network flow formulation is appealing since it is a linear program which can be efficiently solved with the network simplex algorithm (Bertsimas and Tsitsiklis, 1997). The elevator trip OD matrix estimation problem is either over-, exactly- or under-determined, i.e., the number of OD pairs is either less than, equal to or greater than the number of boarding and alighting counts. In the first case, the problem has a unique solution and in the last two cases it has more than one solution.

We present a simple branch-and-bound algorithm to find all solutions to the problem. The algorithm is motivated by the one-three algorithm (Danna et al., 2007). We want to find all solutions and then select one among them randomly because without any additional information it is impossible to know which of the solutions correspond to the true OD passenger counts, i.e., to the realized passenger traffic. In the long term, the random selection results to traffic statistics, i.e., building OD matrices, that are not affected by the algorithm used in solving the problem, and thus, model better the possible realizations of the traffic. This is desirable when the statistics are used to make passenger traffic forecasts for elevator dispatching. An existing method to estimate an elevator trip OD matrix from inconsistent boarding and alighting passenger counts is based on solving a succession of positive inverse problems (Yoneda, 2007). This method, however, cannot be used to find all solutions.

In a real time elevator group control, there is usually only a very short time for solving the elevator trip OD matrix estimation problem. Therefore, we study the execution time of the branch-and-bound algorithm in finding all solutions to four typical example problems.

2 FORMULATION

We define an elevator trip as a directed network of nodes $N = \{1, 2, \ldots, m\}$, and arcs A. The nodes correspond to the floors of a building. Let b_i and a_i denote the measured number of passengers who board and alight the elevator at node i, respectively. Furthermore, let $N^+ \subset \{1, 2, \ldots, m-1\}$ and $N^- \subset \{2, 3, \ldots, m\}$ denote the set of pickup and delivery nodes, respectively. Node $i \in N^+$ if $b_i \geq 1$, and node $j \in N^-$, if $a_j \geq 1$. The nodes $i \in N^+$ and $j \in N^-$, $i+1 \leq j \leq m$, define the OD pair (i, j) if either the passengers who board the elevator at node i register a delivery request to node j, or there is another node k, $k \in \{1, 2, \ldots, i-1\}$, where a delivery request is registered to node j. Let A denote the set of arcs defined by the OD pairs (i, j), $i \in N^+$, $j \in N^-$. Finally, define x_{ij} as the unobservable number of passengers from the origin node i to the destination node j, i.e., the OD passenger count along the arc $(i, j) \in A$.

Figure 1 presents an elevator trip with five nodes and OD pairs. This example corresponds to a situation where the passengers who board the elevator at the first node register delivery requests to nodes 3 and 4, and the passengers who board the elevator at the second node register a new delivery request to node 5. In general, a boarding passenger cannot register a delivery request to a destination where a request already exists but it is of course possible that the passenger travels to this destination. Hence, even if the passengers who board the elevator at the second node cannot register delivery requests to nodes 3 and 4, it is possible that some of them travel to these destinations. In the example, a dashed arc describes this situation, and thus, arcs (2,3) and (2,4) are dashed. It can also be assumed that passengers who board an elevator cannot travel to destinations defined by delivery requests registered at later nodes. This is why there is no arc (1,5) in the example.

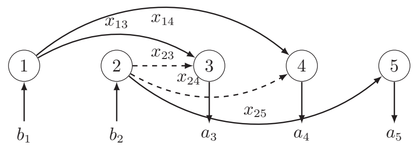

Figure 1: Example elevator trip.

The OD passenger counts are related to the measured boarding and alighting counts through the so called flow conservation constraints, i.e., the total flow out of the node i, $i \in N^+$, equals b_i, and the total flow into the node j, $j \in N^-$, equals a_j:

$$\sum_{j|(i,j)\in A} x_{ij} = b_i \quad \forall i \in N^+, \tag{1}$$

$$\sum_{i|(i,j)\in A} x_{ij} = a_j \quad \forall j \in N^-. \tag{2}$$

We assume that if there is a delivery request from an origin node to a destination node, then at least one passenger must be assigned to that OD pair. Hence, we obtain the following lower bounds for the OD passenger counts:

$$l_{ij} = \begin{cases} 1, & \text{if } i \to j \\ 0, & \text{otherwise} \end{cases}, \tag{3}$$

where $i \to j$ signifies that a delivery request is registered from node i to node j.

The lower bounds could be further tightened. For example, in Figure 1, the lower bound for the OD passenger count x_{25} would be a_5. We assume, however, that the tightening of the bounds is done in the solver before solving the problem. We also assume that the measured boarding and alighting counts are consistent, i.e., $\sum_{i\in N^+} b_i = \sum_{i\in N^-} a_i$, and this holds also when the lower bounds are subtracted from the counts.

2.1 Properties

For every elevator trip with two or more nodes, $m \geq 2$, the number of arcs n satisfies the following bounds:

$$m-1 \leq n \leq m(m-1)/2. \tag{4}$$

The lower bound is the minimum number of edges and the upper bound is the maximum number of edges in any undirected graph. Since the elevator does not change direction during a single trip, the graph describing the elevator trip is acyclic, and thus, the bounds for undirected graphs are valid.

The coefficient matrix corresponding to the constraints (1) and (2) is the node-incidence matrix of the elevator trip. This matrix contains $p = |N^+| + |N^-|$ rows and n columns, and its rank is $p-1$ (Bazaraa et al., 2009). Note that any node on the elevator trip that is both a pickup and a delivery node can be split in two successive nodes so that the resulting delivery node precedes the pickup node. Then, $p = m$ and equation (4) implies that $n \geq p-1$ always. This means that when the node-arc incidence matrix is over-determined, $n < p$, then $n = p-1$ and the matrix has full rank equal to $p-1$. In this case there is a unique OD passenger count vector satisfying the constraints (1)-(3). When the matrix is exactly- or under-determined, $n \geq p$, it is also rank deficient and there are always more than one OD passenger count vector satisfying the constraints. A unique solution can be obtained by formulating the problem of finding the elevator trip OD passenger counts as a minimum cost network flow problem.

2.2 Minimum Cost Network Flow Formulation

To model the elevator trip OD matrix estimation problem as a minimum cost network flow problem, we augment the directed network describing the elevator trip as is done in the simplex method for network flow problems to find an initial basic feasible solution (Bazaraa et al., 2009). We first add to the network an additional node $m+1$, $N = N \cup \{m+1\}$. Then, for every pickup node $i \in N^+$ we define the OD pair $(i, m+1)$ and for every delivery node $j \in N^-$ the OD pair $(m+1, j)$, and add the corresponding new arcs to the set A. The lower bound for each of these arcs is zero. Figure 2 presents the elevator trip of Figure 1 with these additions. A dotted arc corresponds to an OD pair incident to or from the additional node.

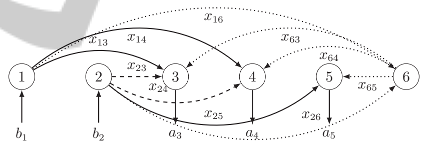

Figure 2: Augmented example elevator trip.

We set the cost of each arc (i, j), c_{ij}, not incident to or from the additional node $m+1$ equal to zero, and the cost of each arc incident to or from the additional node equal to one. The elevator trip OD matrix estimation problem can now be mathematically formulated as a minimum cost network flow problem as follows:

$$\begin{aligned} \text{Minimize} \quad & \sum_{i\in N^+} x_{im+1} + \sum_{j\in N^-} x_{m+1j} \\ \text{subject to} \quad & \sum_{j|(i,j)\in A} x_{ij} = b_i \quad \forall i \in N^+ && (5) \\ & \sum_{i|(i,j)\in A} x_{ij} = a_j \quad \forall j \in N^- && (6) \\ & \sum_{i|(i,m+1)\in A} x_{im+1} + \sum_{j|(m+1,j)\in A} x_{m+1j} = 0 && (7) \\ & x_{ij} \geq l_{ij} \quad \forall (i,j) \in A. \end{aligned}$$

Note that the optimal value of the objective function is zero, i.e., the OD passenger counts of the arcs incident to or from the additional node are zero. Furthermore, because the measured boarding and alighting counts b_i and a_j are always integer-valued and the node-incidence matrix defined by the constraints (5)-(7) is always totally unimodular, all solutions to the problem are always integer-valued (Bazaraa et al., 2009).

3 BRANCH-AND-BOUND ALGORITHM

All solutions to the elevator trip OD matrix estimation problem can be found with a branch-and-bound algorithm presented in Algorithm 1. The algorithm is similar to the second phase of the one-tree algorithm (Danna et al., 2007). One difference between the algorithms is that our algorithm can be used only when the solutions to the linear program (LP) are integer-valued. A second difference is that we accept only optimal solutions, i.e., whose objective value is zero, whereas the one-tree algorithm also accepts solutions whose objective value does not differ more than a predefined percentage of the optimal value. Third, in our algorithm every node always stores the solution of the LP defined by that node. Hence, when we select a new node from the set N, the solution of the LP at that node is readily available, and we proceed to test whether all variables at the node are fixed by the local bounds of the node. If there is a variable that is not fixed by the local bounds, we branch with respect to one such variable, otherwise the node is fathomed.

When branching with respect to variable i at node n, we form the child nodes n_1 and n_2 so that they do not contain the value of the variable i at node n, $x_i(n)$. To find also other optimal solutions where the value of the variable i is equal to $x_i(n)$, we set the local bounds of this variable at node n equal to $x_i(n)$ and add it back to the set of nodes N. When node n is next time selected from the set N, branching at this node is done with respect to some other variable than i, or the node is fathomed. Furthermore, the child nodes and the corresponding solutions are added to the sets N and S only if the solutions are optimal.

It is not always necessary to create two child nodes. For example, when the value of the variable i at node n is equal to its lower bound, $x_i(n) = l_i(n)$, we create only the right children $n_1 = \{x_i \geq x_i(n)+1\}$ and fix the upper bound of variable i at node n equal to its lower bound, i.e., to its value, $u_i(n) = x_i(n)$. In addition, our branching strategy does not produce duplicate solutions. There are also other possibly more efficient algorithms than branch-and-bound to find integer solutions to network flow problems (Baldoni-Silva et al., 2003) but these are not considered in this paper.

Algorithm 1: Branch-and-bound algorithm.

```
Set of nodes: $N \leftarrow \{rootnode\}$
Solve LP at node $rootnode$.
Set of solutions: $S \leftarrow \{x(rootnode)\}$
while $N \neq \emptyset$ do
  Select node $n$ from $N$.
  Set of nodes: $N \leftarrow N \setminus \{n\}$
  if there exists a variable $i$ that is not fixed by local bounds of node $n$: $l_i(n) < u_i(n)$ then
    Create child nodes and fix bounds of node $n$:
    $n_1 = \{x_i \leq x_i(n) - 1\}$, $n_2 = \{x_i \geq x_i(n) + 1\}$
    and $l_i(n) = u_i(n) = x_i(n)$
    Set of nodes: $N \leftarrow N \cup \{n\}$
    for $j = 1 \rightarrow 2$ do
      Solve LP at node $n_j$.
      if objective $z(n_j) = 0$ then
        Set of solutions: $S \leftarrow S \cup \{x(n_j)\}$
        Set of nodes: $N \leftarrow N \cup \{n_j\}$
      end if
    end for
  end if
end while
```

4 NUMERICAL EXPERIMENTS

To study the performance of the branch-and-bound algorithm, we formed four example problems. These problems were selected based on simulations ran with the Building Traffic Simulator (BTS), and they are typical for the elevator trip OD matrix estimation problem (Siikonen et al., 2001; Kuusinen et al., 2012).

The branch-and-bound algorithm was implemented with MATLAB 7.10.0 (R2010a) on a Pentium 2.66 GHz machine with 3.0 GB memory and Windows XP. The LP at each node was solved using the Cplex Class API provided in the CPLEX Optimization Studio V12.2 for MATLAB. CPLEX was chosen since it is somewhat the academic and industry standard for linear optimization problems.

The Cplex class provides methods for the manipulation of the model, which can be used to increase solving speed. One important manipulation is the selection of the algorithm used by the Cplex instance. When the LP is solved for the first time at the root node of the branch-and-bound tree, the algorithm of choice is the network simplex algorithm since it tends to be faster than a general simplex algorithm for network flow problems (Bertsimas and Tsitsiklis, 1997).

Another advantage of the Cplex class is that after solving a LP, the solution and the corresponding basis can be obtained from the Cplex instance. Every node stores this basis which is used as a starting basis at its child nodes together with the dual simplex algorithm. This is a typical approach in branch-and-bound algorithms where the subproblem at each node is a LP and can be solved using a simplex method.

4.1 Example Problems

Problem 1, an over-determined problem, excluding the additional node and arcs, consists of the following nodes, arcs, and boarding and alighting passenger count vectors:

$$\begin{aligned}&N^+ = \{1,2,3\}, N^- = \{3,4\},\\&A = \{(\mathbf{1},\mathbf{3}),(2,3),(\mathbf{2},\mathbf{4}),(3,4)\},\\&b = [10,11,11], a = [11,21].\end{aligned}$$

The lower bound of an arc written in bold equals one, and zero otherwise. Hence, an arc written in bold means that the passengers who board the elevator at the origin of the OD pair register a delivery request to the destination of the OD pair. Otherwise, the delivery request to the destination is registered at an origin before the origin of the OD pair. Furthermore, in this problem there is no arc (1,4) because the delivery request to node 4 is registered by the passengers who board the elevator at node 2, and thus, the passengers who board the elevator at node 1 cannot be traveling to node 4. The boarding and alighting passenger count vectors are formed with respect to the order of the nodes in the sets N^+ and N^-. For example, $b_1 = 10$, $b_2 = 11$, $b_3 = 11$, and $a_3 = 11$ and $a_4 = 21$ for Problem 1.

Problem 2, an exactly-determined problem is defined by:

$$\begin{aligned}&N^+ = \{1,2,5\}, N^- = \{3,4,6\},\\&A = \{(\mathbf{1},\mathbf{3}),(\mathbf{1},\mathbf{4}),(2,3),(2,4),(\mathbf{2},\mathbf{6}),(5,6)\},\\&b = [10,11,10], a = [5,10,16],\end{aligned}$$

and Problem 3, an under-determined problem by:

$$\begin{aligned}&N^+ = \{1,2,3,4\}, N^- = \{5,6,7\},\\&A = \{(\mathbf{1},\mathbf{5}),(\mathbf{1},\mathbf{6}),(\mathbf{1},\mathbf{7}),(2,5),(2,6),(2,7),\\&\quad (3,5),(3,6),(3,7),(4,5),(4,6),(4,7)\},\\&b = [6,5,5,5], a = [7,7,7].\end{aligned}$$

Finally, Problem 4, also an under-determined problem is defined by:

$$\begin{aligned}&N^+ = \{1,2,5,6\}, N^- = \{2,\ldots,16\},\\&A = \{(\mathbf{1},\mathbf{2}),\ldots,(\mathbf{1},\mathbf{6}),(\mathbf{1},\mathbf{8}),\ldots,(\mathbf{1},\mathbf{16}),\\&\quad (2,3),\ldots,(2,6),(\mathbf{2},\mathbf{7}),(2,8),\ldots,(2,16),\\&\quad (5,6),\ldots,(5,16),(6,7),\ldots,(6,16)\},\\&b = [21,1,2,1],\\&a = [5,1,2,2,2,1,1,1,2,1,1,2,2,1,1].\end{aligned}$$

4.2 Numerical Results

The performance of the branch-and-bound algorithm is studied based on the execution time in seconds. To account for the machine dependent variations in the execution time, we solved the first, second and fourth problem 500 times, and the third problem 10 times, and computed the average execution time of these runs. The third problem was solved only 10 times because of a much longer execution time. Table 1 shows the results, namely, the average execution time in seconds and the number of solutions for each problem.

Table 1: Results of the numerical experiments.

Problem	Execution time	No. of solutions
1	0.0075	1
2	0.22	5
3	68.6737	2016
4	1.0513	9

It can be seen that as the number of solutions increases, the execution time increases. The shorter the execution time, the more information about the passenger traffic the elevator group control has at the moment of making a dispatching decision. Furthermore, the group control has to make a decision typically in less than 0.5 seconds. Hence, we conclude that in a real time application, the minimum cost network flow formulation and the branch-and-bound algorithm can be used to find all solutions to problems corresponding to Problems 1 and 2, i.e., over- and exactly-determined problems. Fortunately, most real problems are either over- or exactly-determined (Kusinen et al., 2012).

5 CONCLUSIONS

In this paper, we studied the problem of finding the origin-destination (OD) passenger counts for the OD pairs of an elevator trip. An elevator trip is formally defined as successive stops in one direction of travel with passengers inside the elevator. We formulated the problem as a minimum cost network flow problem. This formulation can be used to estimate the OD passenger counts for elevator trips where the observed number of boarding and alighting passengers are consistent. The elevator trip OD matrices estimated for a given time interval are combined to form a building OD matrix that describes the traffic flow between every pair of floors in the building during that interval. These matrices form traffic statistics that are used to forecast future traffic. The elevator group control uses these forecasts to make robust call allocation decisions in constantly changing traffic conditions.

The coefficient matrix associated with the so called flow conservation constraints is either over-, exactly- or under-determined. In the last two cases the problem has always more than one solution. We presented a simple branch-and-bound algorithm to find all solutions to the problem. When all solutions are available and one is selected randomly, the long-term traffic statistics, i.e., building OD matrices, are not affected by the algorithm used to solve the problem, and thus, model better the possible realizations of the passenger traffic. This is desirable when the statistics are the basis of the passenger traffic forecasts used in elevator dispatching. To assess the performance of the algorithm, we studied its execution time in solving four example problems. Based on the results, the formulation and algorithm can be used in a real time elevator group control application to solve over- and exactly-determined problems. Fortunately, most real problems correspond to these problems.

As the results suggest, the execution time is not acceptable for under-determined elevator trip OD matrix estimation problems. Hence, an ongoing research is to find more efficient ways to solve the problem. We are also studying methods that can be used to estimate the elevator trip OD matrices in the presence of inconsistent boarding and alighting counts. A future challenge is to find out what method to use to make forecasts based on the collected traffic statistics, and detect whether the forecasts made for a given time interval are adequate, and thus, can be used by the elevator group control application.

REFERENCES

Baldoni-Silva, W., De Loera, J., and Vergne, M. (2003). Counting integer flows in networks. Retrieved December 17, 2012, from http://www.math.ucdavis.edu/˜latte/theory/totalresidue.pdf.

Bazaraa, M., Jarvis, J., and Sherali, H. (2009). *Linear Programming and Network Flows*. John Wiley & Sons, Hoboken, New Jersey, 4th edition.

Bell, M. (1983). The estimation of an origin-destination matrix from traffic counts. *Transportation Science*, 17(2):198–217.

Ben-Akiva, M., Macke, P., and Hsu, P. (1985). Alternative methods to estimate route-level trip tables and expand on-board surveys. *Transportation Research Record*, 1037:1–11.

Bertsimas, D. and Tsitsiklis, J. (1997). *Introduction to Linear Optimization*. Athena Scientific/Dynamic Ideas, LLC, Nashua/Charlestown, U.S.A., 4th edition.

Cascetta, E. and Nguyen, S. (1988). A unified framework for estimating or updating origin/destination matrices from traffic counts. *Transportation Research Part B*, 22(6):437–455.

Danna, E., Fenelon, M., Gu, Z., and Wunderling, R. (2007). Generating multiple solutions for mixed integer programming problems. In *IPCO 2007, LNCS 4513*, pages 280–294. Springer-Verlag.

Fisk, C. (1988). On combining maximum entropy trip matrix estimation with user optimal assignment. *Transportation Research Part B*, 22(1):69–79.

Furth, P. and Navick, D. (1992). Bus route o-d matrix generation: Relationship between biproportional and recursive methods. *Tranportation Research Record*, 1338:14–21.

Kuusinen, J.-M., Sorsa, J., and Siikonen, M.-L. (2012). The elevator trip origin-destination matrix estimation problem. Unpublished manuscript submitted to Transportation Science 4.7.2012.

Lamond, B. and Stewart, N. (1981). Bregman's balancing method. *Transportation Research Part B*, 15(4):239–248.

Li, B. (2009). Markov models for bayesian analysis about transit route origin-destination matrices. *Transportation Research Part B*, 43(3):301–310.

Li, Y. and Cassidy, M. (2007). A generalized and efficient algorithm for estimating transit route ods from passenger counts. *Transportation Research Part B*, 41(1):114–125.

Lundgren, J. and Peterson, A. (2008). A heuristic for the bilevel origin-destination matrix estimation problem. *Transportation Research Part B*, 42(4):339–354.

Maher, M. J. (1983). Inferences on trip matrices from observations on link volumes: a bayesian statistical approach. *Transportation Research Part B*, 17(6):435–447.

Nguyen, S. (1984). Estimating origin-destination matrices from observed flows. In Florian, M., editor, *Transportation Planning Models*, pages 363–380. North-Holland, Amsterdam.

Siikonen, M.-L., Susi, T., and Hakonen, H. (2001). Passenger traffic flow simulation in tall buildings. *Elevator World*, August:117–123.

Tsygalnitsky, S. (1977). Simplified methods for transportation planning. Master's thesis, Department of Civil Engineering, Massachusetts Institute of Technology, Cambridge.

Yoneda, K. (2007). Elevator trip distribution for inconsistent passenger input-output data. *Decision Making in Manufacturing and Services*, 1(1-2):175–190.

Zuylen, H. V. and Willumsen, L. (1980). The most likely trip matrix estimated from traffic counts. *Transportation Research Part B*, 14:281–293.