Transforming bus smart card transaction data into vehicle trajectories

Author(s):
Fourie, Pieter J.

Publication Date:
2014-12

Permanent Link:
https://doi.org/10.3929/ethz-b-000082277

Rights / License:
In Copyright - Non-Commercial Use Permitted
Transforming bus smart card transaction data into vehicle trajectories

Pieter J. Fourie

Working paper
Future Cities Laboratory, Singapore ETH Centre

December 2014
Contents

1 Introduction: ......................................... 1

2 Data description .................................. 1
   2.1 The Singapore Contactless ePurse Application System (CEPAS) ............. 1
   2.2 Other data sources .................................. 2

3 Problem .............................................. 2

4 Method ............................................... 4
   4.1 Identifying bus routes ................................ 5
   4.2 Correcting GPS errors ................................ 5
   4.3 Clustering into dwell operations ................................ 7
   4.4 Grouping dwell operations into circuit runs ................................ 8
   4.5 Dwell event interpolation ................................ 8
   4.6 Constructing vehicle trajectories between dwell operations .................. 9

5 Applications ...................................... 9
   5.1 Average waiting times by time of day ........................................ 9
   5.2 Waiting times related to vehicle trajectories for a given bus service .......... 11
   5.3 A simple multiple regression to predict bus headways and their variation .... 12

6 Conclusion ......................................... 15

7 References ......................................... 16

8 Appendix: derivation of system-wide macroscopic fundamental diagrams from smart card records .................................. 18
   8.1 Transforming CEPAS smartcard records into accumulation and output ...... 18
   8.2 Comparing dynamic vs expected steady-state accumulation ................. 20

List of Figures

1 Problems in identifying bus dwell operations from smart-card transactions .... 3
2 Overview of the trajectory reconstruction process .................................. 4
3 Filtering of GPS errors using the "speed" between consecutive transactions ...... 6
4 Interpolation of missing dwell operations ............................................ 7
5 Average waiting time distributions by time of day .................................. 10
List of Tables

1. Simple regression model of headway: 14
2. Multiple regression model of headway standard deviation: 14
3. Performance summary: 15
Transforming bus smart card transaction data into vehicle trajectories

Pieter J. Fourie
Future Cities Laboratory
Singapore ETH Centre
1 CREATE Way, 138602 Singapore
phone: +65-9116-8440

fourie@ivt.baug.ethz.ch

December 2014

Abstract

A rule-based procedure reconstructs vehicle trajectories from bus smart card transaction data. It takes as input data a series of tap-in and tap-out transactions, geo-located to bus stops, recording the card identifier, bus line and direction, vehicle identifier and transaction time. Furthermore, it requires a network description and transit routes and schedule information compliant with the MATSim input format. The process produces a series of “events” compliant with the MATSim format; these are time-stamped atomic units of information that describe the movements of the buses through the network, their dwell operations at stops, together with passenger boardings and alightings. MATSim-compliant analysis tools are then capable of visualising bus movements and perform analyses of ridership, in-vehicle travel times, transfer times and more. Multinomial regression models are derived to describe headway and its variation. Variables tested in the regression analyses include system-wide congestion-levels, taken from a macroscopic fundamental diagram of passenger accumulation and output. A simple method to derive these quantities from the smartcard data is shown, and accumulation is also compared against expected steady-state values calculated using Little’s Law. Further application of the resulting trajectory data includes a realistic model of travel time between transit stops during the course of the day in a fast, simplified transit simulation, and to drive a travel time and waiting time-aware, dynamic transit router.

Keywords
Trajectory reconstruction, transit smart cards, agent-based simulation, MATSim, Singapore

Preferred citation style
1 Introduction

Trajectory reconstruction from transit smart card data includes constructing passenger trajectories or flows (e.g., Itoh et al., 2013; Yang et al., 2013; Yuan et al., 2013) or train trajectory reconstruction through regression analysis (e.g., Sun et al., 2012). It appears that no work has been done so far to reconstruct the detailed trajectories of bus vehicles from these data.

The reconstruction of these trajectories could facilitate the analysis of transit performance measures on a per-vehicle basis. One could also identify causes of phenomena like bus bunching and interactions between different lines holding each other up at bus stops, to name only a few obvious applications.

In this work, bus vehicle trajectories are reconstructed from a full day of smart card transactions in Singapore, from a mid-April Wednesday, 2013. The trajectories were recorded in a format compliant with the output from the agent-based simulation system, MATSim (www.matsim.org), using the specifications developed by Rieser (2010).

This procedure makes visualisation and analysis possible in compliant software, such as Senozon Via (senozon AG; 2015). A significant application of the resulting trajectory data is for the construction of realistic meta-models of travel time between transit stops during the course of a day in a fast pseudo-simulation (Fourie et al., 2013), as well as to route agents using realistic travel times in a dynamic transit router (Ordóñez Medina and Erath, 2013).

2 Data description

2.1 The Singapore Contactless ePurse Application System (CEPAS)

The CEPAS system operates with a contactless smart card that keeps track of a user’s pre-loaded cash balance. When a user taps the card on a system sensor, a transit transaction is recorded and relevant information calculated and the user’s balance adjusted. The Singapore public transportation system uses distance-based pricing, and operates differently depending on transportation mode (rail or bus). If they need to use the mass rapid transit (MRT) train service, users tap in and out at station entrances. The exact service and vehicle they use, as well as their boarding/alighting time are therefore not directly recorded, but can be derived through regression analysis (e.g., Sun et al., 2012).
In the case of bus services, users have to tap in and out when they enter and leave the bus. For each transaction, the system records an array of information, including the vehicle’s registration number, the user’s card identifier, the service and direction of the bus, the boarding or alighting time, as well as the boarding or alighting stop locations, inferred from the vehicle’s GPS tracker.

2.2 Other data sources

We also require a description of the road network as a directed graph of nodes and links, and the location of transit stops in the road network. Finally we need to know the complete route description for each route in the system, i.e. the sequence of stops visited and the sequence of network links traversed between stops. For the case of Singapore, all of this information has been gathered and converted into the MATSim format in a previous exercise (Erath et al., 2012).

3 Problem

Intuitively, if one wants to reconstruct bus trajectories from smart card records, then one would simply isolate the transactions for a single vehicle, go through the list of transactions at each stop, identify when the first user enters and the last user leaves the bus, and assume that the bus arrived a few seconds before and departed a few seconds after these two respective events. If these conditions are satisfied for each stop in the bus’s route, then you would have the travel time between each and every bus stop along the route. You could then reconstruct the trajectory between the stops using the description from the electronic transit schedule, by assigning a series of reasonable link travel times contingent on maximum speed and link length, for all links between each pair of stops.

Of course the actual problem is much more nuanced and complicated, as can be seen in Fig. 1. This figure shows the transactions for a single bus between 6:40 and 7 o’clock in the morning. The number of passengers on the bus is inferred by keeping track of the number of tap-in and tap-out transactions. Each transaction is coloured by a unique bus stop identifier. A number of typical errors are highlighted.

Firstly, bus stop IDs can be incorrectly identified because of GPS errors. These errors might be due to stops being very close to each other, high buildings in the city centre confusing GPS signals, transient weather conditions, etc.
Bus ridership is inferred from transactions by time of day. Transactions are colored by bus stop ID. The red text highlights possible problems in identifying the start and end of a dwell operation.

We also see a number of transactions that occur before or after larger clusters of transactions at the stop. These transactions appear to occur before and after the bus doors have opened and closed for the dwell operation at the stop. Such late tap–outs are possible because, as the bus approaches the stop, its GPS sensor activates the contactless smartcard sensor and people can start to tap out even before the bus has come to a full stop. Sometimes the bus has to wait in a queue before it can enter the bus bay and safely open its doors, so a substantial amount of time can pass before the first tap-out is registered and passengers can actually start exiting the vehicle.

Late tap-ins, on the other hand, could be due to two possible behaviours: a user might enter the bus, its door closes and it pulls away while the user frantically searches for their smartcard in order to perform the transaction. The other possibility is that, especially in long bus bays that serve a number of lines, the bus takes on passengers at one end of the bus stop and then reopens its doors when it reaches the front of the bus bay in order to take on late passengers.

The final significant problem is the case where buses pass by bus stops that have no passengers
waiting and where no passengers on the bus need to get off. Of course, a bus that doesn’t register any transactions during its circuit run is completely invisible to the reconstruction procedure, and if the bus doesn’t register any transactions for a number of stops at the beginning or end of its run, an extrapolation of its rate of movement will be pure conjecture. However if, as in the case of the bus in Fig. 1, only a number of intermediate stops were passed by without any transactions, one could make a more reasonable interpolation of its rate of movement from stop to stop based on the maximum allowed speed and distances of the links between the stops.

Another subtle problem that does not appear in the figure is the case where a bus route is circular or loops in on itself, i.e. bus routes where one or more stops are visited more than once. These routes can make it difficult to identify when a circuit run has only started or whether it is at its end.

4 Method

An overview of the trajectory reconstruction process appears in Fig. 2: The process starts with a number of preparatory operations, beginning with loading the supporting data structures i.e. the network and electronic transit schedule. A number of look-up data structures are prepared in order to associate a bus’s set of transactions with a particular route, based on the line direction and bus stops it visited.
4.1 Identifying bus routes

An SQL connection is established with the smart card transaction database and transactions are drawn for every combination of bus registration number, transit line and direction. This information is not enough to uniquely identify the route that the bus has operated on for an entire day, because the same bus might be travelling in the same direction on the same transit line a number of times, but in one circuit run it might be operating the full route and in another it might be operating an express route or a shortened contingency service.

Bus lines not appearing in the transit schedule are ignored; this can happen due to simple oversight, or when transactions are recorded for a later date than the date at which the electronic transit schedule was compiled.

For a given combination of bus registration number, line and direction (RLD), a set of possible routes are identified, and routes are ranked according to the number of times transactions at the set of stops in the route have been visited by the bus. The bus route with the highest number of transactions registered at its stops is assumed to be the route traversed by the bus and will form the basis of all further analysis for this particular set of transactions.

4.2 Correcting GPS errors

In Fig. 3, the process for eliminating GPS errors is illustrated. It shows the ridership on a bus inferred from counting the number of boardings and alightings over time. Each transaction in Fig. 3(a) is coloured by a speed value, with slower transactions shifted towards blue and faster transactions shifted towards the red end of the spectrum. These speed values are calculated as the network distance between the bus stops associated with two consecutive transactions divided by the time difference between the transactions.

An iterative process removes transactions with high speed values (arbitrarily taken to be more than 80 km an hour) one at a time, and speeds are recalculated for all transactions after each removal. The number and timing of errant transactions is recorded, in order for the correct stop IDs to be assigned to these transactions in a later procedure that associates them with a dwell operation that overlaps with the errant transaction time.
Figure 3: Filtering of GPS errors using the ‘speed’ between consecutive transactions.

(a) Bus ridership coloured by transaction speed, red denotes high speed

(b) Transactions coloured by stop ID before filtering

(c) Transactions coloured by stop ID after filtering.

High transaction speeds in Fig. 3(a) correspond to sudden changes in color (stop ID) in Fig. 3(b).
Figure 4: Interpolation of missing dwell operations, and removal of erroneously identified dwell operations.

(a) Bus dwell operations coloured by stop ID before interpolation of missing operations

(b) Bus dwell operations coloured by stop ID after interpolation of missing operations, and removal of erroneously identified dwell operations

The two graphs show dwell operations for a bus before and after missing operations have been identified. When an erroneous dwell operation had been identified (usually due to very late tap-ins or GPS errors where a single transaction occurs at a neighboring stop, and that transaction is associated with the preceding stop), comparison against the route description identifies these and they are removed.

4.3 Clustering into dwell operations

For each RLD combination, transactions are then processed by stop ID, and closely timed transactions are clustered to produce a vehicle dwell operation (arrival at and departure from a bus stop). For each dwell operation, the vehicle arrival and departure time is then taken from the largest sub-cluster of closely timed transactions, in order to eliminate false dwell operation timing from early tap-outs and late tap-ins.

If a boarding transaction is found before the assigned arrival time, or an alighting transaction is found after the assigned departure time, the dwell operation timing is adjusted. Departure and arrival is assumed to take place three seconds before and after the first and last transaction used to identify the dwell operation duration. The duration of dwell operations identified in this way is adjusted to a minimum of six seconds for dwell operations where very few transactions occur.

Transactions associated with the dwell operation but that occur outside the dwell operation time
window (due to tapping out early or tapping in late) have their timings adjusted to fit inside the window.

4.4 Grouping dwell operations into circuit runs

In order for the output from the trajectory reconstruction process to be converted into a valid MATSim event stream, each individual circuit run, from first to last stop, needs to be identified for each RLD combination’s set of inferred dwell operations.

To this end, dwell operation order is evaluated against the sequence of stops visited in the associated route description, and each dwell operation is added to the circuit run if its bus stop follows that of the preceding dwell operation. If the dwell operation’s bus stop is not in order, it might indicate the start of the new circuit run or an incorrect bus stop ID associated with the dwell operation. This latter case is possible for bus stops that are relatively close to each other, and dwell operations with a single transaction that has a GPS error but was not removed in the preceding steps. If the dwell operations following this one continue the sequence of the preceding dwell operations, then the errant dwell operation is removed as a mis-classification, and its associated transactions are ignored; otherwise the dwell operation is associated with a new circuit run, and the process repeats.

4.5 Dwell event interpolation

Missing dwell operations (stops driven past with no transactions) are created in order to satisfy the requirements for the MATSim transit specification. The results of this process is illustrated in Fig. 4.

These dwell operations have a zero duration, and they are spaced in time by dividing the interval between the first and last observed dwell operations into a set of intervals that is proportional to the expected travel time between the set of bus stops that were passed by. These expected travel times between stops are produced by a regression model derived from the smart card data, which is described in a forthcoming publication.
4.6 Constructing vehicle trajectories between dwell operations

A vehicle trajectory is written out in a separate file for each RLD combination in the MATSim XML format. The file records each dwell operation and its associated transactions, and the sequence of links the vehicle traverses between consecutive dwell operations, taken from the transit schedule. Given the time between consecutive dwell operations, the time spent on each link is taken as the time interval between stops, multiplied by the free-speed travel time of the link (not exceeding 80 km/h, assumed to be the maximum speed of a bus), divided by the total free-speed travel time of all links between stops. While the procedure does not account for congestion effects on the individual links between two stops, it does capture the total effect of congestion between two stops, in the total travel time.

Finally, the separate vehicle XML files are combined chronologically into a single output file using a merge-sort operation.

5 Applications

In this section a number of possible applications of the output from the procedure is illustrated.

5.1 Average waiting times by time of day

In Fig. 5, we have two views of the average waiting time (AWT) distribution across the entire system in 15 minute intervals. AWT is calculated based on a definition by the London transport authorities (e.g. Schil, 2012) in Eq. (1), using a rolling window operating on the observed headways \( H_i \), for the last \( k \) dwell operations recorded for a particular line and direction at a particular stop.

\[
AWT = \frac{\sum_{i-k}^{i} H_i^2}{2 \sum_{i-k}^{i} H_i}
\]  

(1)

In Fig. 5(a), for every 15 minute interval, the number of bus stops with AWTs within the appropriate interval are counted for each of the 15 second waiting time bins.

In the second plot, Fig. 5(b), the number of passengers boarding during the time of day- and
Figure 5: Average waiting time (AWT) distributions by time of day.

(a) AWT by TOD by dwell operation count

(b) AWT by TOD by passengers boarding.

The colours in the plots indicate the number of dwell operations and number of passengers affected. AWT is calculated using a rolling window operating on the last nine dwell operations recorded for a particular line and direction at a particular stop. Waiting time resolution is 15 seconds, while that of the x-axis is 15 minute time intervals.
waiting time intervals is shown. From this figure can be seen that most passengers experience waiting times in the order of six minutes in the morning peak, with a relatively larger spread than that observed during the evening peak. During intermediate hours waiting times are longer as bus frequencies drop outside peak hours.

5.2 Waiting times related to vehicle trajectories for a given bus service

Fig. 6 summarizes the performance of bus services operating on a relatively long bus line in Singapore. The coloured background shows AWT at each stop, plotted at 15 minute intervals for a rolling window of nine dwell operations within and preceding each interval. Overlaid black lines are the time space plots that show the movement of each individual vehicle along the route. Circular dots of varying sizes indicate the number of passengers boarding at each stop for each vehicle.
The plot clearly illustrates how bus bunching leads to increases in AWT, especially at stops that are located towards the end of the line. Note that, around 4 o’clock in the afternoon at bus stop 51, a short line appears, that does not extend back to the first stop. This line highlights a shortcoming in the current procedure; this particular circuit run probably only had its first transaction recorded at stop 51. Dwell operations and vehicle trajectory before and after the first and last recorded transaction for a circuit run are not extrapolated. However, the plot also reveals that a reasonable remedy to this shortcoming might be to assume that the bus had traversed the preceding stops at a rate intermediate to the buses before and after it.

5.3 A simple multiple regression to predict bus headways and their variation

From Fig. 6, it appears as if there are a number of quantities that correlate with bus headway. In this section a simple regression model is estimated to determine the extent to which the headway of any given service can be predicted during the course of a day, purely from smart-card-derived information. Furthermore, the variability of headway, expressed as a standard deviation, is also estimated as a function of a number of variables.

Firstly, headways are clearly strongly dependent on the scheduled frequency of a bus service, or some quantity strongly correlated with this attribute. Unfortunately, we have found the number of departures in our electronic schedule to be inconsistent with observations in the smart-card dataset; as many as 50% more departures are recorded in the General Transit Feed Service (GTFS)-derived electronic schedule. Therefore, scheduled frequency is calculated as the sum of headways of the last $k$ services at a particular stop for a particular service, divided by $k$.

Another naive expectation is that long headways are correlated with a large number of people waiting at the bus stop, in other words a dwell operation that has a long headway will also have a large number of boarding transactions. However a simple linear regression of headway as a function of the number of boardings shows in fact a negative correlation, with an intercept of 502 seconds, that drops by 6.46 seconds per boarding transaction. While strongly statistically significant, the $R^2$-value comes to only 0.0024.

Headways should vary as a function of the total system load or the degree of congestion. System-wide levels of congestion can be described by a point on the macroscopic fundamental diagram for the bus system (see Appendix), which was estimated on a per-second basis, with accumulation taken as the number of passengers tapped into the bus system, and output taken as the tap-out rate in a moving average window of 15 minutes.
The interaction of buses at the bus bay should affect headway depending on the configuration of the bus bay. In the model estimated here, the degree of interaction is approximated by the number of dwell operations recorded at the stop.

Model coefficients are shown in Table 1. The scale and sign of model coefficients are consistent with their magnitude in the correlation plot shown in Fig. 7. Output (tap-outs per second) was not included as an indication of system-wide service level/congestion because of its strong correlation with accumulation; including the variable breaks the model estimation and inverts the sign of model variables.

The adjusted $R^2$ value for the model comes to 0.327. The scheduled headway approximation (denoted as $sh$) shows the biggest influence, for obvious reasons. The sign of the variable denoting number of boardings remains the same as in the simple regression, but its magnitude is
Table 1: Simple regression model of headway for a given bus service at a given bus stop during the course of the day.

| Variable   | Estimate  | Std. Error | t value     | Pr(>|t|)   |
|------------|-----------|------------|-------------|-----------|
| (Intercept)| 7.4670763 | 1.2369533  | 6.036680    | 0.0000000 |
| boardings  | -7.4293886| 0.0818764  | -90.739083  | 0.0000000 |
| stop_no    | 0.0313146 | 0.0153561  | 2.039227    | 0.0414280 |
| dcas       | 0.0000157 | 0.0004844  | 0.032466    | 0.9741002 |
| acc        | 0.0009131 | 0.0000135  | 67.837803   | 0.0000000 |
| sh         | 0.5361070 | 0.0006599  | 812.459254  | 0.0000000 |

Variable names are the same as in the correlation plot shown in Fig. 7. The units of boardings, stop_no, dcas are seconds per boarding; seconds per bus stop number in the route; and seconds per number of dwell operations, respectively. Accumulation (acc) units is seconds per number of persons tapped-in system-wide, and scheduled headway (sh) is unitless (seconds per second).

Table 2: Multiple regression model of headway standard deviation for a given bus service at a given bus stop during the course of the day. Variable names are the same as for

| Variable   | Estimate  | Std. Error | t value     | Pr(>|t|)   |
|------------|-----------|------------|-------------|-----------|
| (Intercept)| 207.4322800| 0.5020925 | 413.135552  | 0.0000000 |
| boardings  | -3.0222032 | 0.0332449 | -90.907232  | 0.0000000 |
| stop_no    | 3.3581565 | 0.0062342 | 538.665757  | 0.0000000 |
| dcas       | 0.0004835 | 0.0001967 | 2.458322    | 0.0139589 |
| acc        | 0.0001609 | 0.0000055 | 29.455956   | 0.0000000 |
| sh         | 0.0206641 | 0.0002679 | 77.147535   | 0.0000000 |

increased to 8.52 seconds per boarding transaction, with a very small standard error compared to the estimate. Perhaps the counterintuitive sign just reflects the correlation with the higher service frequency and reduced headway during peak hours with many passengers at the stops, as can be seen in Fig. 5(b).

Accumulation also has a strongly significant effect, and as its value can go up to 90,000 passengers during the peak hour, it can contribute up to 82 additional seconds of expected headway during this time. The number of dwell operations at any given stop during the course of an entire day does not seem to have a significant effect; this implies that stops with a high number of services operating on them will be designed to handle the load and not cause too much interference between services.
Table 3: Performance summary for the reconstruction process.

<table>
<thead>
<tr>
<th>Measure</th>
<th>Qty.</th>
</tr>
</thead>
<tbody>
<tr>
<td>number_of_RLDs</td>
<td>6935</td>
</tr>
<tr>
<td>dwell_operation_count</td>
<td>1137133</td>
</tr>
<tr>
<td>interpolated_dwell_ops</td>
<td>449683</td>
</tr>
<tr>
<td>dwell_operations_dropped</td>
<td>15551</td>
</tr>
<tr>
<td>transaction_count</td>
<td>6472872</td>
</tr>
<tr>
<td>dropped_dwell_operation_transactions</td>
<td>89634</td>
</tr>
<tr>
<td>stop_id_not_in_schedule_transactions</td>
<td>20571</td>
</tr>
<tr>
<td>gps_error_transactions</td>
<td>73204</td>
</tr>
</tbody>
</table>

The model of headway standard deviation ($h_{wsd}$) appears to be similarly consistent with the correlation plot in Fig. 7, with variability generally increasing with increasing stop number along the route. The adjusted $R^2$ value for the model comes to 0.180. Variability increases slightly with increasing scheduled headway, which suggests that bus lines with a relatively low frequency can be more adversely affected by bus bunching than higher frequency lines, where there might be a greater emphasis on balancing supply with demand. Accumulation as an indicator of the system-wide level of congestion also has a relatively small but statistically significant effect during peak hours.

6 Conclusion

The procedure, as it stands, is only a first attempt and several improvements can be made, as can be seen from the errors noted in Table 3. But it does establish a methodology that can inform a machine learning approach, which will be the subject of further research.

The procedure is clearly useful for evaluating bus system performance using a variety of measures, and its output can be used in a set of interactive dashboards for daily operations analysis. Furthermore, it is useful to be able to see vehicle movements in compliant software such as senozon via, where dynamic vehicle interaction effects can be directly observed.

Vehicle trajectory data can also be used to create a model of bus speeds between stops throughout the course of a day, as well as expected waiting times at stops, which can be used to power an agent-based simulation of public transportation. In this way, transient conditions in the network
due to congestion can be captured without having to explicitly simulate car traffic.

The process is available in the open source MATSim project as a set of Java classes that can be downloaded with the MATSim nightly build and be used with a compliant smartcard dataset.

7 References


8 Appendix: derivation of system-wide macroscopic fundamental diagrams from smart card records

The verification of the existence of a macroscopic fundamental diagram (MFD) by Geroliminis and Daganzo (2008) is a recent development that has spurred a lot of interest, as it can allow decision-makers to implement demand-side policies and evaluate their effects on a neighbourhood scale. The MFD relates the accumulation of vehicles in a neighbourhood to their average flow, speed or arrival rate.

The concept has been extended to public transportation, in order to investigate the possibility of maximising passenger flow as a function of the accumulation of cars and buses in the transportation network (e.g. Geroliminis et al., 2014; Chiabauti, 2015; Chiabaut et al., 2014). So far in this effort seem to be focused on using simulated data, as the availability of detailed, accurate and synchronised private vehicle, public transport and passenger information is hard to come by.

In this section, a simple method is demonstrated to construct a variant of the MFD from smartcard data, on a passenger level, for the entire public transport system of Singapore. It relates the accumulation of passengers in the system across both modes as a function of output, i.e. the rate at which they leave the system by tapping out at station exits or from buses.

8.1 Transforming CEPAS smartcard records into accumulation and output

The method will be explained using simple structured query language (SQL) pseudocode for the system-wide level; of course it can be applied by transport mode, for a selection of transport lines, or only for trips originating and terminating at the selection of bus stops or stations.

CEPAS data is provided as a series of $n$ trip records, each with a tap-in and tap-out timestamp in seconds, denoted as tapin and tapout respectively. A new table, mfd1, is created from these data (using only complete records), with each record being a vector of the timestamp in seconds and accumulation increment from a tap-in or tap-out transaction:

```
CREATE TABLE mfd1 AS
SELECT tapin AS t, 1 as i
FROM cepas_smartcard_records
WHERE tapin IS NOT NULL
```
AND tapout IS NOT NULL
UNION
SELECT tapout as t, -1 as i
FROM cepas_smartcard_records
WHERE tapin IS NOT NULL
AND tapout IS NOT NULL;

The next step is to aggregate the 10.5 million records to a total value for ridership increment and output in each second. In the pseudocode it is assumed that there are both tap-in and tap-out transactions for every second of the day from the beginning until the end of the service. Further manipulation will be needed for cases with this condition does not apply; such exceptions are easily dealt with by somebody proficient in SQL:

```
-- get the total increment per second:
CREATE TABLE mfd2a AS
SELECT t, SUM(i) AS i_total
    FROM mfd1
    GROUP BY t;

-- output is the total of all tap-outs per second
CREATE TABLE mfd2b AS
SELECT t, - SUM(i) AS output
    FROM mfd1
    WHERE i < 0
    GROUP BY t;
```

At this stage we still do not have the accumulation of the system. In the pseudocode below it is calculated by taking the cumulative sum of the ridership increment from the beginning till the end of the day, after which the tables of output and accumulation are joined together:

```
-- assuming the SQL system supports window functions...
CREATE TABLE mfd3a AS
SELECT t,
    SUM(i_total) OVER (ORDER BY t) AS acc
    FROM mfd2a;

CREATE TABLE mfd3b AS
SELECT * FROM
    mfd2b NATURAL JOIN mfd3a;
```

We essentially now have the data required to plot the MFD; however, such plots will exhibit a large degree of scatter because of the high time resolution. In the plots displayed in this section, a moving average method, specifically the \texttt{rollmean} function in the \texttt{zoo} package for R, with
Figure 8: MFD of bus vs. train, calculated over a 15 minute moving average on a per-second basis.

an adjustable time window was used for both accumulation and output in order to get a better understanding of the average dynamics over time.

8.2 Comparing dynamic vs expected steady-state accumulation

Little’s Law \( L = \lambda W \) (Little, 1961) relates the number of items in a queueing system \( L \) to the arrival rate of items \( \lambda \) and the average waiting/processing time \( W \).

As noted by Helbing (2009), Little’s Law can be used to estimate the average queue length in an urban network as the product of the average time spent in the network and the arrival rate. He remarks that the relationship holds for time-averaged variables, even in the case of non-uniform arrivals, if the system is stable; which means that the queue length is not systematically growing or shrinking.

Of course, queue length, or accumulation, is not stable in the public transport system, which is why we investigate MFD’s in the first place. However, it is interesting to compare the actual
Figure 9: Actual vs. expected (steady-state) accumulation by mode, calculated over a 15 minute moving average on a per-second basis. Expected accumulation is calculated using Little’s law, the product of tap-in rate and average travel time.

Accumulation in the system against an expected value from Little’s law as an indication of how the system deviates from the steady-state and to give an alternative perspective on the larger scale dynamics of the system to that observed in the standard MFD.

It is straightforward to develop moving averages of both the time spent in the system (i.e. moving average of the difference between tap-out and tap-in times for each record), as well as the ‘arrival rate’ (a moving average of tap-outs per second). We can then plot the moving average of actual accumulation against that of the value calculated using Little’s law, as in Fig. 9.

Fig. 10 compares the ratio of actual against expected steady-state accumulation over the course of the day for the bus and rail modes, using rolling average time windows of 1, 15 and 60 minutes respectively. Ignoring the extreme values at the beginning and end of the plots, where a lack of observations causes distortion, we can see that longer time windows in the moving average subsumes small-scale variations and reveals that bus, on average, operates close to steady-state throughout the day, whereas the rail mode experiences longer term fluctuations during the morning and evening peaks.

The difference between the two modes can probably be explained by the way in which transac-
tions are registered. For the bus mode the time spent in the system is only in-vehicle time, as users tap in and tap out when they enter or exit the vehicle. In comparison, we only have the transaction times at the gates of railway stations, therefore the time spent in the system includes access walk time from the station entrance to the platform, access waiting time, in-vehicle time, transfer time and egress walking time from the platform to the station exit. A fairer comparison will therefore be one where the calculation for bus is performed on a journey basis, using the tap-in time of the first bus stage and the tap-out time of the last bus stage, in order to capture transfer time accumulation.

The rail system also has a much larger capacity to accumulate people and provide room for accumulation to rapidly grow or shrink, thereby deviating from steady-state. Performing the journey-level comparison for bus will therefore also partially take account of the increased capacity for accumulation at bus stops.
Figure 10: Ratio of actual over expected (steady-state) accumulation vs. time of day (1 second resolution). Values are moving averages for three window sizes, showing how short-term dynamic effects are subsumed with increasing moving average window size.

(a) 1 minute moving average

(b) 15 minute moving average

(c) 60 minute moving average