Practical issues with DTA

CE 392D

PREPARING INPUT DATA

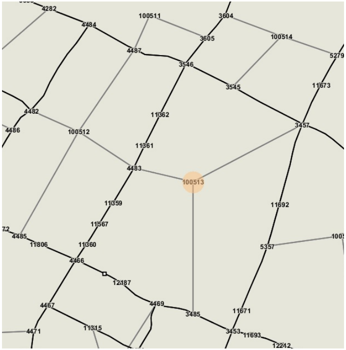
What do you need to run the basic traffic assignment model?

- The network itself
- Parameters for link models (capacities, free-flow speeds, etc.)
- OD matrix

The extensions of TAP require additional information, such as demand functions, destination attractiveness, logit parameters, value of time, etc.

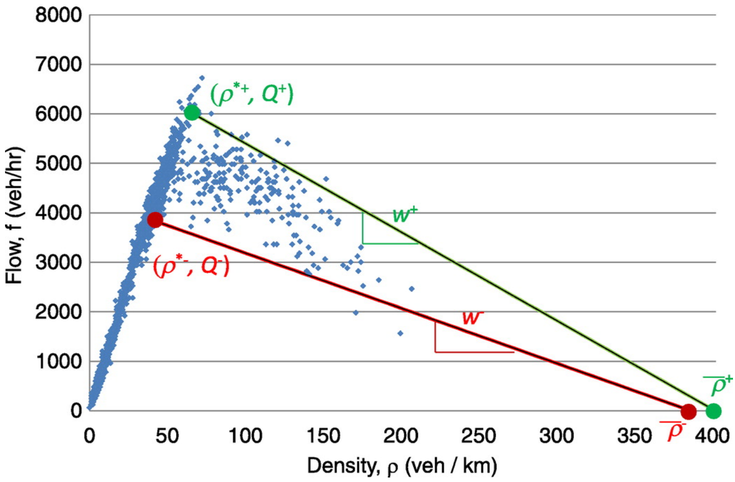
Deciding which streets to include in the network is a balance of accuracy and computation time/data collection requirements.

In practice, regional models typically include minor arterials and larger roads; neighborhood streets are typically abstracted into centroid connectors:

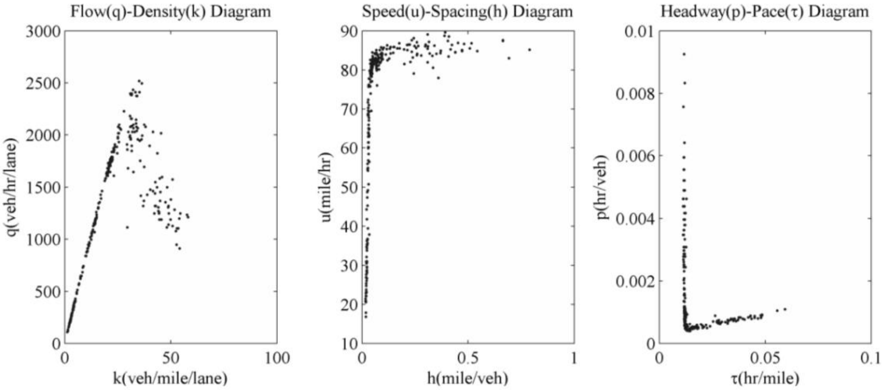


Neighborhood streets are typically uncongested, so there isn't a need to model them in great detail. (Or is there?)

Ideally, fundamental diagrams are obtained through regression of field data. What are the complications?



(Aside: recent research is looking at changes-of-variables which produce a better fit to data.



These use Lagrangian coordinates (n, t) or (n, x), instead of Eulerian coordinates (t, x).

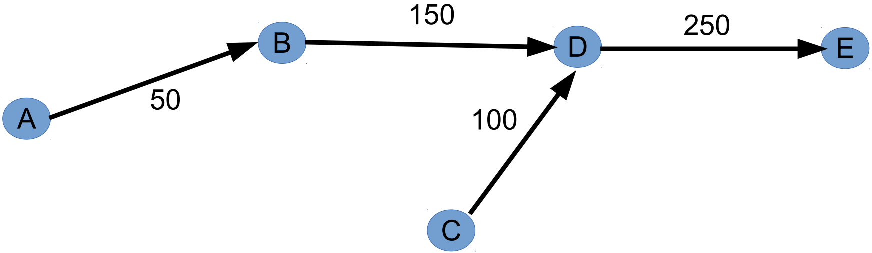
The OD matrix is often the most challenging input data to calibrate, for several reasons:

- There are many more OD matrix entries than links.
- The OD matrix can't be observed directly (unlike link speeds and flows).

Can we use direct observations (say, link flows) to try to estimate the OD matrix?

This is surprisingly difficult!

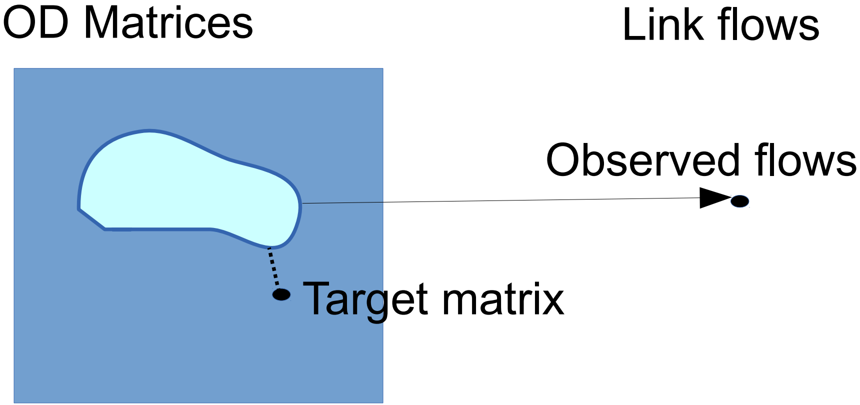
Because there are more OD matrix entries than links, the problem is highly underdetermined; the problem is not finding an OD matrix that matches the



One trivial solution is for all trips to go from one node to a neighboring node.

(As an aside, the distinction between a good regression fit and a good model is absolutely critical here.)

We often have an OD matrix available from other parts of the planning process, say, a gravity model. Can we use this "target" OD matrix as a starting point which can be adjusted to conform to link flows?



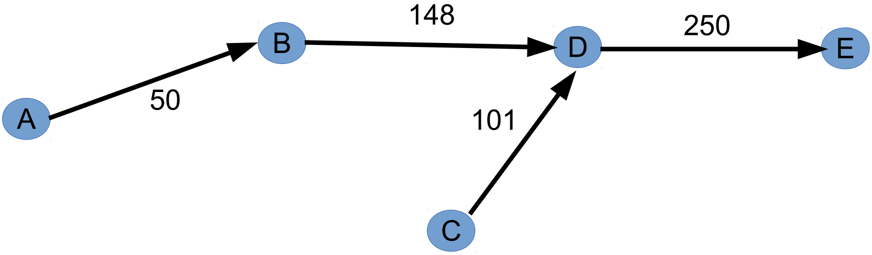
We can try a least-squares approach where we try to match *both* the target OD matrix and the link flows:

$$\min_{\mathbf{d},\mathbf{x}} \lambda \sum_{rs} \left(d^{rs} - \overline{d}^{rs} \right)^2 + (1-\lambda) \sum_{ij} (x_{ij} - \overline{x}_{ij})^2$$

where λ reflects the importance put on matching the OD matrix relative to the link flows; the proper balance is a matter of judgment and depends on the level of trust in the accuracy of $\overline{\mathbf{d}}$ and $\overline{\mathbf{x}}$.

We have two constraints: nonnegativity of OD matrix entries $d^{rs} \geq 0$, and that the link flows $\mathbf{x}$ must be a user equilibrium solution given the OD matrix $\mathbf{d}$.

Because field data contains some noise and error, however, all solutions which satisfy link flows exactly may have short trips:



A **quasi-dynamic** approach can help match the number of decision variables and observations.

Divide the analysis period into "sub-periods"; assume that within each sub-period, the distribution of trips between origins and destinations is fixed.

If we have counts available on n_{lc} links, and there are n_θ time steps in the analysis period, then we have $n_{lc}n_\theta$ observations we can use.

If there are n_{od} OD pairs, then there are $n_{od}n_\theta$ entries in the dynamic OD matrices; generally $n_{od} \gg n_{lc}$, so we have many more unknowns than equations.

With the approach of Cascetta et al., let n_o be the number of origins, and n_τ the number of "sub-periods" of the analysis peirod. Then the number of unkowns is:

- $n_\theta n_o$ values, giving total departures from each origin at each time step.
- $n_\tau(n_{od} - n_o)$ values, giving the proportion of the demand from each origin to each destination during time period τ.

By adjusting the length of the “sub-periods” and timestep, we can bring balance to the number of unknowns and equations.

The ratio is approximately 1 if $n_{lc} \approx n_o + \frac{n_\tau}{n_\theta}(n_{od} - n_o)$

$$\{\mathbf{g}^{*1},\ldots,\mathbf{g}^{*\theta},\ldots,\mathbf{g}^{*n_\theta};\mathbf{p}^{*1},\ldots,\mathbf{p}^{*\tau},\ldots,\mathbf{p}_o^{*n_\tau}\} = \underset{\substack{x_o^\theta \geqslant 0 \quad \forall o, \quad \forall \theta \in T \\ 0 \leqslant \pi_{d|o}^\tau \leqslant 1 \quad \forall \pi_{d|o}^\tau \in \boldsymbol{\pi}_{d|o}^\tau \quad \forall \tau \in T \sum_d \pi_{d|o}^\tau = 1 \quad \forall o, \quad \forall \tau \in T}}{\arg\min}$$

$$\left\{ \sum_{\theta=1}^{n_\theta} \sum_{od=1}^{n_{od}} \frac{\left(x_o^\theta \cdot \pi_{d|o}^{\tau(\theta)} - \hat{d}_{od}^\theta\right)^2}{\sigma_{od}^\theta} + \sum_{\theta=1}^{n_\theta} \sum_{l=1}^{n_{lc}} \frac{\left(\sum_{\theta'=\theta_l}^{\theta} \sum_{od=1}^{n_{od}} m_{odl}^{\theta'\theta} x_o^{\theta'} \cdot \pi_{d|o}^{\tau(\theta')} - \hat{f}_l^\theta\right)^2}{\sigma_l^\theta} \right\}$$

The $m_{odl}^{\theta'\theta}$ parameters are key: they reflect the fraction of flow that departred OD pair *od* at time θ' which is on link *l* at time θ. This can be obtained from network loading; **but if the OD matrix changes substantially from the seed, we need to do another loading**.

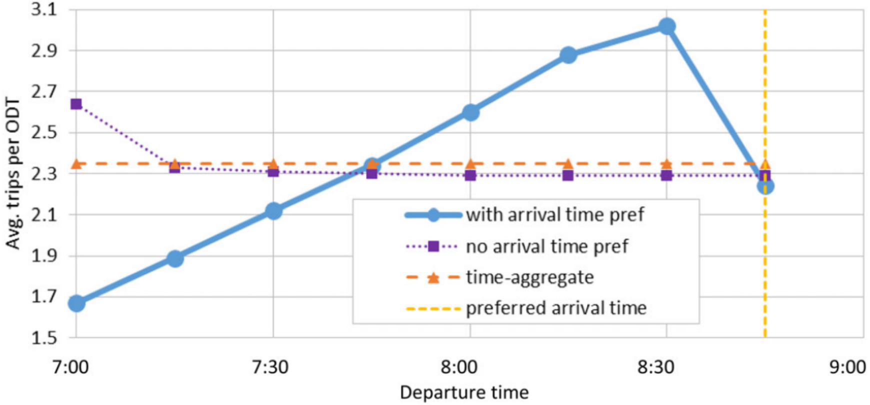
An alternative formulation: use departure time choice to automatically "profile" the demand.

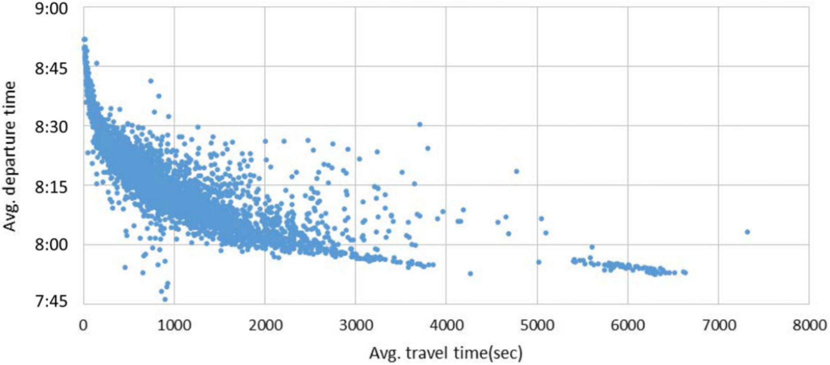
Advantage: Departure time profiles are endogenous; determined *behaviorally* rather than *statistically*

Disadvantage: Parameters in schedule delay equations may vary over the population and with time.

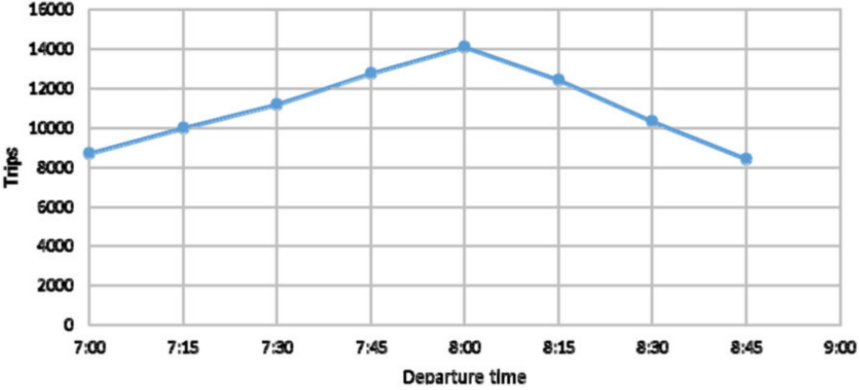
For more details, see:
Levin, M. W., S. D. Boyles, and J. Duthie. (2016) Demand profiling for dynamic traffic assignment by integrating departure time choice and trip distribution. *Computer-Aided Civil and Infrastructure Engineering* 31, 86–99

Uniform preferred arrival time



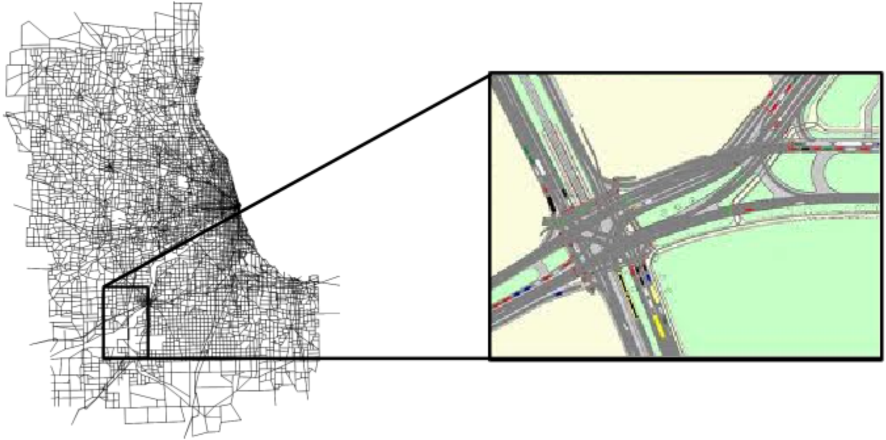

9:00
8:45
8:30
8:15
8:00
7:45
Avg. departure time
0
1000
2000
3000
4000
5000
6000
7000
8000
Avg. travel time(sec)

Normally distributed preferred arrival time



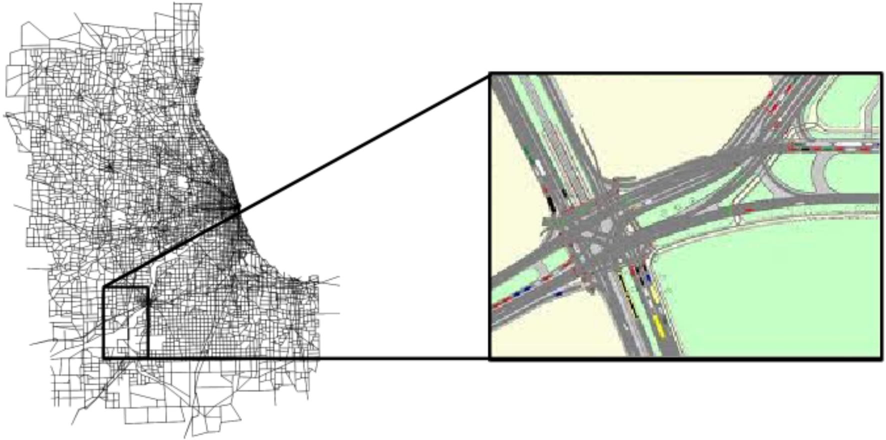
MULTISCALE MODELING

Multiscale modeling aims to get the “best of both world,” so to speak.

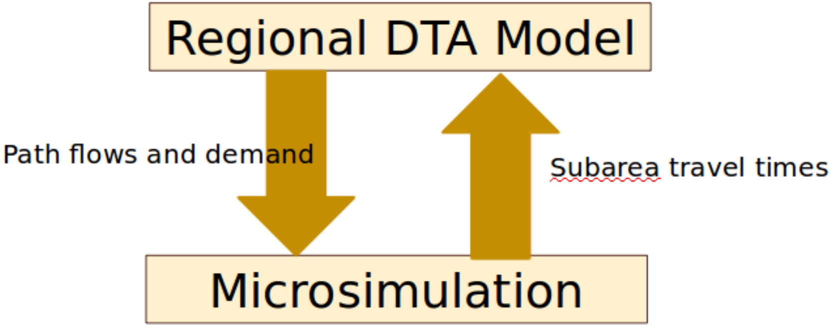


A microsimulator provides detailed results on a small area; a regional model gives more aggregate results on a larger area.

But how do two models communicate with each other?

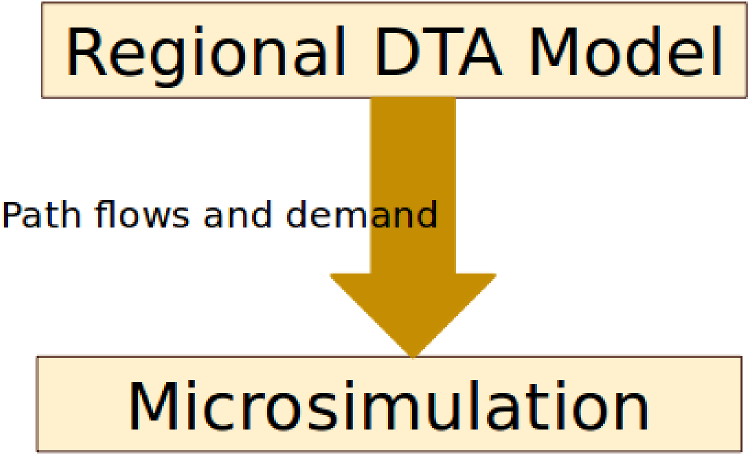


In principle, information can flow from either model to the other.



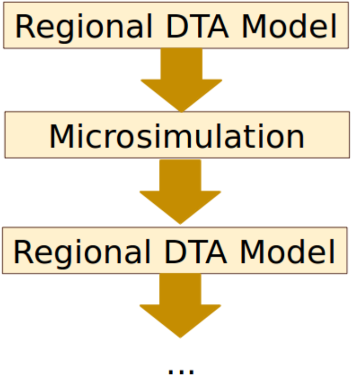
A **consistent** solution to both models respects both directions.

The main focus is usually on the boundary conditions, translating one models outputs to anothers inputs.



Solve for equilibrium on the regional model; see which vehicles enter the subarea, and use those as the path flows for microsimulation.

In practice, this usually involves an *ad hoc* fitting-together and sequential solution.



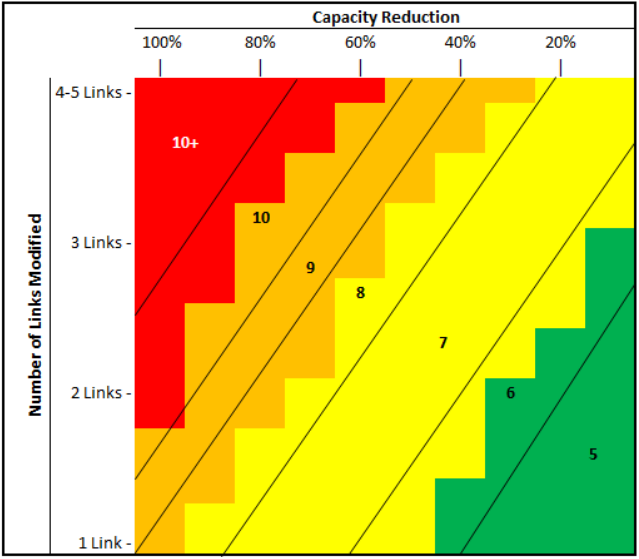
Can we do better?

How large should the subarea be?

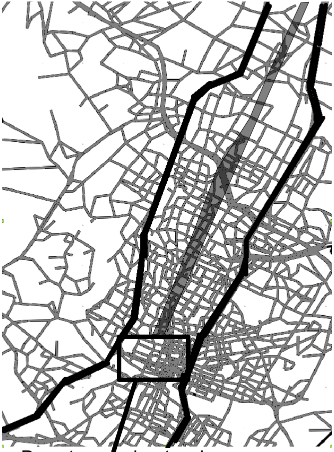
This question was studied for two “nested” DTA models in the Austin area.

(Bringardner, Gemar, Machemehl, Boyles, 2014–15)

These analyses gave recommendations about the subarea "radius" in terms of the number of affected links, and the severity of the capacity reduction.

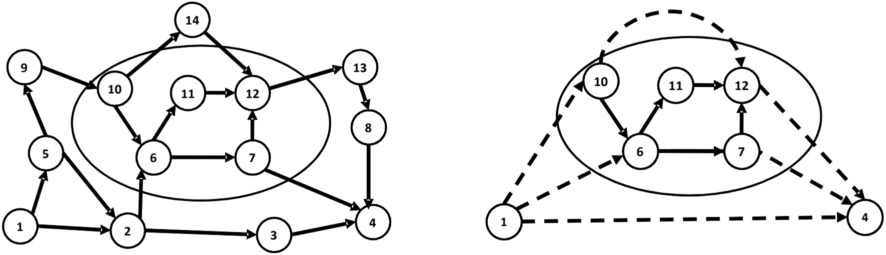


Or is the "how large" question a red herring?



The real question is where the *interactions* lie, and how large they are.

An emerging alternative is the use of "soft" boundaries, which blur the distinction between the models.



The subarea model retains a **simplified** version of the regional network, rather than eliminating it entirely. Route choice and diversion can be modeled naturally.

ACCURACY AND STABILITY

Multiscale models highlight the question of model *stability*: how do model outputs change with inputs. Ideally, a model is not overly sensitive to having exactly the right input parameters.

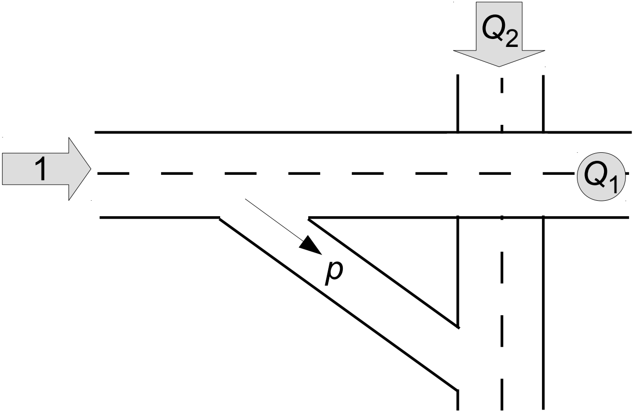
In reality, this has been relevant all along!

I'm starting with a simpler setting, with two equally-tractable network loading models. The objective is to predict the steady-state flow rates on links in one of two ways:

Spillback: If a link's outflow is restricted, its steady-state inflow will be similarly restricted.

No spillback: Restrictions to a link's outflow are not transmitted to its inflow.

Clearly, the spillback model is more realistic, and will be treated as "ground truth." However, spillback can introduce discontinuities into flow models, so small input errors can potentially propagate into much larger output errors.



p and Q_2 are model parameters, the objective is to estimate the flow Q_1.

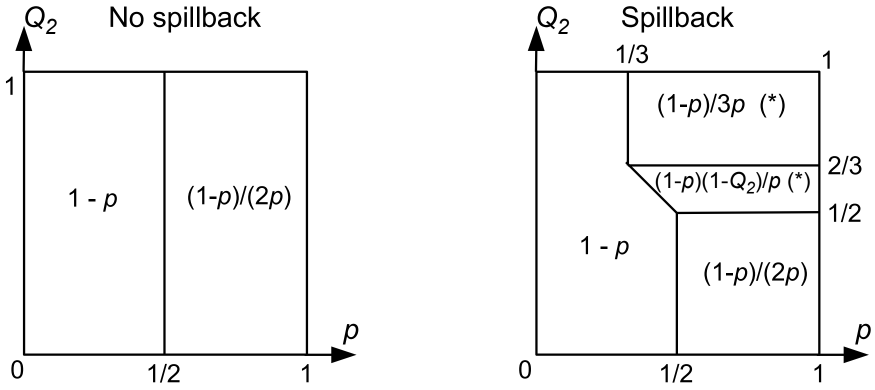
Standard merge and diverge equations apply

At the merge, in highly congested conditions flow is allocated proportionate to capacity, if the sending flow from an approach is less than this the other approach can increase its flow.

The diverge respects the first-in, first-out principle, flows waiting to exit the freweay will obstruct thru traffic.

In a spillback model, the steady-state inflow rate to the onramp cannot exceeds its outflow rate.

This example is small enough that it can be solved exactly.



If we don't know p and Q_2 exactly, which model gives better results?

For a given “true” value of of p and Q_2, perform the following:

- Generate n sampled values of Q_2 and p, using independent normal distributions, with means $\hat{p}$ and $\hat{Q}_2$, and given standard deviation.
- For each sample, ϵ^{NS} and ϵ^{S} are the absolute errors of the no-spillback and spillback models.
- Calculate the additional expected error in the no-spillback model: $\delta = E[\epsilon^{NS} - \epsilon^{S}]$, and its standard deviation s.
- Calculate the t score: $t = \delta/(s/\sqrt{n})$

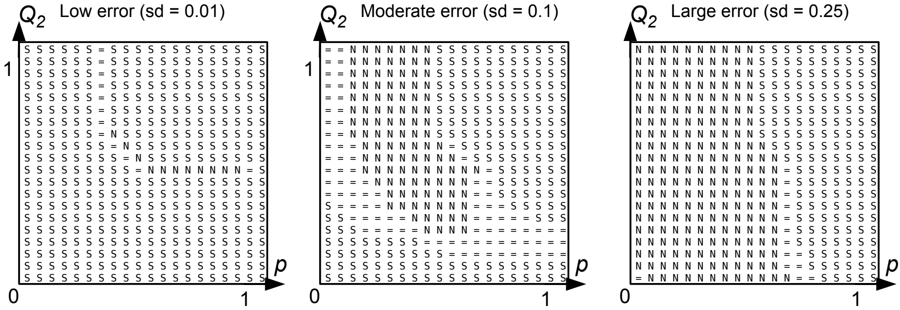
If t is greater than a positive critical value, we can conclude the no-spillback model has higher error.

If t is less than a negative critical value, the spillback model has higher error.

Otherwise that there is no significant difference between the models in terms of error.

400 true values were chosen, uniformly distributed in $[0, 1]^2$.

2500 samples were run for each case.



Average positive t score: $+87.4$, average negative t score: -13.9.