Solving equilibrium problems and estimating route flow

Cathy Wu\textsuperscript{1}  Jerome Thai\textsuperscript{1}  Steve Yadlowsky\textsuperscript{1}

\textsuperscript{1}Department of Electrical Engineering & Computer Sciences
University of California at Berkeley

September 5, 2014
Outline

Equilibrium problems

Route flow estimation problem

Numerical experiments

Conclusions and extensions
Outline

Equilibrium problems

Route flow estimation problem

Numerical experiments

Conclusions and extensions
Introduction to traffic assignment

Setting of the traffic assignment

- A directed $G(N, A)$ with $N$ the set of nodes and $A$ the set of links
- For each OD pair $k = (s, t) \in N \times N$, we have a demand $d_k$
- Each link $a \in A$ has a flow $v_a$ and a latency function $S_a$

Typical assumptions on $S_a$:

- $S_a$ only depends on $v_a$ the flow on link $a$ (no interactions b/w links)
- $S_a$ is continuous strictly increasing
- Latency function given by the BPR:

\[
S_a(v_a) = t_a(1 + 0.15(v_a/c_a)^4) \tag{1}
\]

with $t_a$ the free flow travel time and $c_a$ the capacity on link $a$
User equilibrium (UE) and system optimum (SO)

**Wardrop’s first principle (UE)**

The journey times in all routes actually used are equal and less than those which would be experienced by a single vehicle on any unused routes. (Each user non-cooperatively chooses its own route to minimize its travel time.)

Note: UE is also called *user-optimum* and is same as *Nash equilibrium*.

**Wardrop’s second principle (SO)**

At SO, the average journey time is minimum. (Each user cooperatively chooses its own route to minimize the average travel time over the whole system.)
Formulation as a mathematical program

The UE link flow $v_{UE} = (v_{a}^{UE})_{a \in A}$ is solution of the following program:

$$\min_{v} \phi_{UE}(v) \quad \text{s.t.} \quad v \in K \quad \text{with} \quad \phi_{UE}(v) = \sum_{a \in A} \int_{0}^{v_{a}} S_{a}(u)du$$  \hspace{1cm} (2)

THE SO link flow $v_{SO} = (v_{a}^{SO})_{a \in A}$ is solution of the following program:

$$\min_{v} \phi_{SO}(v) \quad \text{s.t.} \quad v \in K \quad \text{with} \quad \phi_{SO}(v) = \sum_{a \in A} v_{a}S_{a}(v_{a})$$  \hspace{1cm} (3)

with $K$ the set of feasible link flows (given by the conservation of flows).

Note: $\phi_{UE}$ and $\phi_{SO}$ are both strictly convex from assumptions on $v_{a}$.
Computing UE/SO using route flow formulation

- Enumerate all routes from $s$ to $t$ for all OD pairs $k = (s, t) \in N \times N$.

- Let $P_k := \{\text{all routes between } k \in N^2\}$, and $P := \bigcup_{k \in N^2} P_k$

- Construct the link-route incidence matrix $A \in \{0, 1\}^{|A| \times |P|}$:

$$A_{ap} = \begin{cases} 1 & \text{if } a \in p \\ 0 & \text{o.w.} \end{cases} \quad (4)$$

- Construct the OD-route incidence matrix $U \in \{0, 1\}^{|N|^2 \times |P|}$:

$$U_{kp} = \begin{cases} 1 & \text{if } p \in P^k \\ 0 & \text{o.w.} \end{cases} \quad (5)$$

- $K$ encodes the conservation of route flows:

$$K = \{v \in \mathbb{R}_+^{|A|} | \exists f \in \mathbb{R}_+^{|P|}, Af = v, Uf = d\} \quad (6)$$

- Route flow formulation:

$$\min \phi(Af) \quad \text{s.t.} \quad Uf = d, \ f \succeq 0 \quad (7)$$

Equilibrium problems
Computing UE/SO using link flow formulation

- Construct the node-link incidence matrix $N \in \{-1, 0, 1\}^{\|N\| \times |A|}$

\[
N_{ia} = \begin{cases} 
1 & \text{if link } a \text{ enters node } i \\
-1 & \text{if link } a \text{ leaves node } i \\
0 & \text{o.w.}
\end{cases} \quad \forall \ i \in N, \ \forall \ a \in A
\] (8)

- Construct the source-sink vectors $r^k \in \mathbb{R}^{|N|}$, $\forall k = (s, t) \in N^2$:

\[
r^k_i = \begin{cases} 
-d_k & \text{if node } i \text{ is the origin } s \\
d_k & \text{if node } i \text{ is the destination } t \\
0 & \text{o.w.}
\end{cases} \quad \forall \ i \in N
\] (9)

- $K$ encodes the conservation of flows at each node:

\[
K = \{ v \in \mathbb{R}^{|A|}_+ | \exists v^k \in \mathbb{R}^{|A|}_+, v = \sum_{k \in N^2} v^k, N v^k = r^k, \ \forall \ k \} 
\] (10)

- Link flow formulation:

\[
\min_{v} \left( \sum_{k \in N^2} v^k \right) \quad \text{s.t.} \quad N v^k = r^k, \ v^k \succeq 0, \ \forall \ k \in N^2
\] (11)
Solvers

- Both the link flow and route flow formulations are convex programs
- Any convex optimization packages should perform well
- Implementation on Python using cvxopt.org:
  
  github.com/jeromethai/traffic-estimation-wardrop

- Large-scale implementation projected gradient descent:
  
  github.com/cathywu/traffic-estimation

- More specialized algorithms, such as Frank-Wolfe, see references in: §11.2.3.1, J. de D. Ortuzar and L. G. Willumsen. *Modelling Transport*
Outline

Equilibrium problems

Route flow estimation problem

Numerical experiments

Conclusions and extensions
Problem statement: route flow estimation

**Cellpath flow**
Flow along a sequence of cells

**Route flow estimation problem**

Given
- Road network
- Top routes between origin-destination (OD) pairs
- Cellpath flows, $f$
- OD flows, $d$
- Observed link flows, $b$

Recover
- Flow along routes, $x$
Traffic estimation framework

Network topology, routes, cell towers → Map

Cellular data → Cellpath flow

Traffic cameras, radars, detectors → Link flow

Census, travel surveys, OD models → OD flow

Convex optimization formulation → Solver

Route flow solution
Example problem setup

All flows are in 1000 vehicles/hour.

Pointpath flows:
\[
\begin{align*}
&f_{p1234} = 1 = x_1 \\
&f_{p1654} = 4 = x_2 \\
&f_{p654} = 10 = x_3 + x_4
\end{align*}
\]

OD demands:
\[
\begin{align*}
&d_{AB} = 5 = x_1 + x_2 \\
&d_{CB} = 10 = x_3 + x_4
\end{align*}
\]

Link flow: \( b = 9 = x_2 + x_3 \)

\[
(Ux = f) : \begin{bmatrix} 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 1 \end{bmatrix} x = \begin{bmatrix} f_{p1234} \\
 f_{p1654} \\
 f_{p654} \end{bmatrix}
\]

\[
(Tx = d) : \begin{bmatrix} 1 & 1 & 0 & 0 \\
0 & 0 & 1 & 1 \end{bmatrix} x = \begin{bmatrix} d_{AB} \\
 d_{CB} \end{bmatrix} \quad ; \quad (Ax = b) : [0 \ 1 \ 1 \ 0] x = b
\]

Sol. with pointpaths: \( x^* = [1 \ 4 \ 5 \ 5]^T \); sol. with ODs: \( x = x^* + [1 \ -1 \ 1 \ -1]^T t, \ \forall t \in [-1, 4] \)
Convex optimization formulation

Constrained quadratic program (QP):

\[
\min \quad \frac{1}{2} \| Ax - b \|_2^2 \\
\text{s.t.} \quad Ux = f, \quad x \geq 0
\]
Convex optimization formulation

Constrained quadratic program (QP):

\[
\begin{align*}
\text{min} \quad & \frac{1}{2} \| Ax - b \|_2^2 \\
\text{s.t.} \quad & Ux = f, \quad x \geq 0
\end{align*}
\]

- link-route: \( A_{lr} = \begin{cases} 1 & \text{if } l \in r \\ 0 & \text{else} \end{cases} \); cellpath-route: \( U_{pr} = \begin{cases} 1 & \text{if } r \in \mathcal{R}^p \\ 0 & \text{else} \end{cases} \)

- \( b \in \mathbb{R}^{\mid \mathcal{L} \mid} \) observed link flow vector, \( b = (b_l)_{l \in \mathcal{L}} \)

- \( x \in \mathbb{R}^{\mid \mathcal{R} \mid} \) vector of route flows, \( x = (x_r)_{r \in \mathcal{R}} \)
Convex optimization formulation

Constrained quadratic program (QP):

\[
\begin{align*}
\text{min} & \quad \frac{1}{2} \|Ax - b\|_2^2 \\
\text{s.t.} & \quad Ux = f, \ x \geq 0
\end{align*}
\]

- link-route: \( A_{lr} = \begin{cases} 1 & \text{if } l \in r \\ 0 & \text{else} \end{cases} \)
- cellpath-route: \( U_{pr} = \begin{cases} 1 & \text{if } r \in \mathcal{R}^p \\ 0 & \text{else} \end{cases} \)

- \( b \in \mathbb{R}^{\mid\mathcal{L}\mid} \) observed link flow vector, \( b = (b_l)_{l \in \mathcal{L}} \)
- \( x \in \mathbb{R}^{\mid\mathcal{R}\mid} \) vector of route flows, \( x = (x_r)_{r \in \mathcal{R}} \)
- First constraint: route flows \( x_p = (x_r)_{r \in \mathcal{R}^p} \) for cellpath \( p \in \mathcal{P} \) sums to cellpath flow \( f_p \)
- Second constraint: route flows are nonnegative
Model assumptions

Overall assumptions

- Quasi-static, i.e. traffic demands (flows) remain constant over time
- Noiseless case
- All cellpaths $p \in \mathcal{P}$ are contiguous: each pair of consecutive cells in $p$ shares a boundary
- The set of cellpaths $\mathcal{P}$ is well-posed: there exists a unique cellpath $p \in \mathcal{P}$ for each route $r \in \mathcal{R}$, and we have a cellpath flow measurement $f_p$ for each $p \in \mathcal{P}$
Model assumptions

Overall assumptions

- Quasi-static, i.e. traffic demands (flows) remain constant over time
- Noiseless case
- All cellpaths $p \in \mathcal{P}$ are contiguous: each pair of consecutive cells in $p$ shares a boundary
- The set of cellpaths $\mathcal{P}$ is well-posed: there exists a unique cellpath $p \in \mathcal{P}$ for each route $r \in \mathcal{R}$, and we have a cellpath flow measurement $f_p$ for each $p \in \mathcal{P}$

Simplifications for the large experiment

- Approximated cellpath flow via sets of cells, rather than sequences
- Coupled OD and cellpath flow information, i.e. $U$ is a cellpath&OD-route incidence matrix, $f$ is cellpath&OD flow (Note: we’re removing this simplification, results pending.)
Solvers

- Solve via projected gradient method
- Projection: PAV algorithm, from isotonic regression
  - $O(n)$: improvement over direct simplex projection, $O(n \log n)$, with $n$ the number of routes sharing a cellpath $|R^p|$, $p \in \mathcal{P}$
  - Reduces size of the overall problem by number of cellpaths, $|\mathcal{P}|$
- First-order methods are most suitable for large-scale problems
- We used Barzilai-Borwein (BB): non-monotonic fast first-order descent method with super linear behavior on large scale problems
- Other options: L-BFGS (quasi-newton method with linear behavior), stochastic gradient descent, FISTA, accelerated proximal gradient method, active-set methods, etc.
Projection

Our constrained QP can be put into the following form via:

- Equality constraint elimination
- Selection of a particular solution $x_0$
- Manipulating the nullspace

**Cheap projection**: at each step, projection of $y$ onto the feasible set is in the of $p$ separable block ordinal least squares problems, solved exactly in $O(n)$ with isotonic regression algorithm

$$P_\Omega(y) = (z^*(1), \ldots, z^*(p))$$

where

$$z^*(k) = \arg \min_z \|z^{(k)} - y^{(k)}\|_2^2$$

s.t.

$$0 \leq z_1^{(k)} \leq \cdots \leq z_{n_k-1}^{(k)} \leq 1, \ \forall k$$
**Large scale implementation**

- `scipy.sparse`: sparse matrix computations
- PostGIS database: storage of routes, cell tower Voronoi tessellations, links
- PostGIS spatial queries (via GEOS library): extracting cellpath information for each route
- PAV projection: implemented in C
- Algorithms (BB, LBFGS): implemented in Python 2.7

Implementation available
- Algorithms: github.com/cathywu/traffic-estimation
- System: github.com/syadlowsky/phi-estimation
- Cell generation: github.com/cathywu/synthetic-traffic
Outline

Equilibrium problems

Route flow estimation problem

Numerical experiments

Conclusions and extensions
Experiment flow

- UE
- SO
- agent-based

Model

Solver

route flow estimate $\hat{x}$

Error($\hat{x}$, $x^{true}$)

true route flow $x^{true}$

$A, b, U, f$

data

Numerical experiments
Experiment flow

- **Route flow error**: \( \epsilon_r = \|x^{true} - \hat{x}\|_1 / \|x^{true}\|_1 \), percent error of flow allocation among all routes.
Route flow error: $\epsilon_r = \|x^{true} - \hat{x}\|_1 / \|x^{true}\|_1$, percent error of flow allocation among all routes.

Link flow error, GEH statistic:
1) For observed links: $\epsilon_i^{obs} = \left| GEH_i^{obs} < 5, \forall i \in \hat{b} \right| / |b^{true}|

$b^{true} = Ax^{true}$ true observed flows; $\hat{b} = A\hat{x}$ estimated observed flows

$GEH_i^{obs} = \sqrt{\frac{(b_i^{true} - \hat{b}_i)^2}{0.5(b_i^{true} + \hat{b}_i)}}$ associated GEH measure for each link.
Experiment flow

- Route flow error: $\epsilon_r = \frac{\|x^{true} - \hat{x}\|_1}{\|x^{true}\|_1}$, percent error of flow allocation among all routes.

- Link flow error, GEH statistic:
  1) For observed links: $\epsilon_{i}^{obs} = \left|GEH_{i}^{obs} < 5, \forall i \in \hat{b}\right| / |b^{true}|$
     $b^{true} = Ax^{true}$ true observed flows; $\hat{b} = A\hat{x}$ estimated observed flows
     $GEH_{i}^{obs} = \sqrt{\frac{(b_{i}^{true} - \hat{b}_{i})^2}{0.5(b_{i}^{true} + \hat{b}_{i})}}$ associated GEH measure for each link.
  2) For all links: $\epsilon_{i}^{full} = \left|GEH_{i}^{full} < 5, \forall i \in \hat{v}\right| / |v^{true}|$
     $v^{true} = A^{full}x^{true}$ true full flows, $\hat{v} = A^{full}\hat{x}$ estimated full flows
     $GEH_{i}^{full} = \sqrt{\frac{(v_{i}^{true} - \hat{v}_{i})^2}{0.5(v_{i}^{true} + \hat{v}_{i})}}$ associated GEH measure for each link.
Highway network for I-210

- Coordinates for bounding box: [-118.328, 33.985, -117.681, 34.256]
- Network has $m = 44$ nodes, $n = 122$ links, $N = 4$ sets of 42 OD pairs
OD pairs

Morning rush hour

Add flow in equilibrium to model morning congestion
Experiment

- true delay function
- network geometry
- link capacities
- free flow delays
- OD demands
- observations

compute UE

UE link flows

observer

estimated delay function

structural estimation

add noise
Observed links
Full network (highway + arterials)

- OSM network (10538 nodes, 20476 links)
- PeMS sensors (1033 observed links)
- MATSim data (500k simulated agents)
- 321 TAZ origin/destinations, \( \approx 700 \) TAZ regions utilized

Numerical experiments
Full network (highway + arterials)

- OSM network (10538 nodes, 20476 links)
- PeMS sensors (1033 observed links)
- MATSim data (500k simulated agents)
- 321 TAZ origin/destinations, $\approx 700$ TAZ regions utilized
- Experiments: 200-4000 cells, distributed by employee population and major roads; control experiment with 0 cells (just OD information)
- Experiments: 3-50 top routes per OD (92K-305K routes overall)
MATSim numerical results

**Route flow error from cell + OD data**

- 3 Routes
- 10 Routes
- 20 Routes
- 30 Routes
- 40 Routes
- 50 Routes

**Deg. of freedom from cell + OD data (AM)**

- 3 Routes
- 10 Routes
- 20 Routes
- 30 Routes
- 40 Routes
- 50 Routes

**Model route flow error from cell + OD data**

**MATsim link flow error**

- >2700vph
- 700-2700vph
- <700vph

**Numerical experiments**
Outline

Equilibrium problems

Route flow estimation problem

Numerical experiments

Conclusions and extensions
Future work

Next steps
- Dynamic setting
- Handling noise

Upcoming experiments
- AT&T OD model
- AT&T raw traces

Extensions
- Comparison to other frameworks for route flow estimation/assignment
- Incorporating additional data