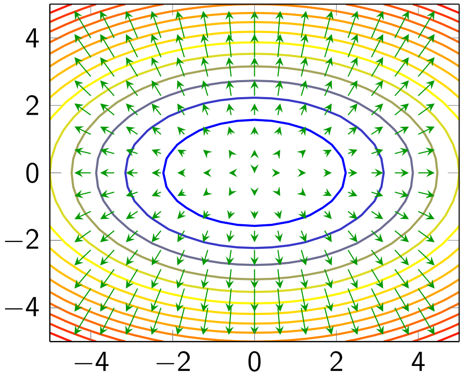
CE 272
Traffic Network Equilibrium

Lecture 8

Method of Successive Averages

Previously on Traffic Network Equilibrium...

The following plot shows the level sets and the gradient $[2x_1 \ 4x_2]^T$.



The gradient vector is 'orthogonal to the level sets'*

Previously on Traffic Network Equilibrium...

Theorem

$\mathbf{x}^*$ *satisfies the VI,* $\mathbf{t}(\mathbf{x}^*)^T(\mathbf{x}-\mathbf{x}^*) \geq 0 \, \forall \, \mathbf{x} \in X \Leftrightarrow$ *it satisfies the Wardrop principle*

Previously on Traffic Network Equilibrium...

The User Equilibrium (UE) formulation in terms of the path flows ys is given by

$$\min \sum_{(i,j)\in A} \int_0^{\sum_{p\in P} \delta_{ij}^p y_p} t_{ij}(\omega)\, d\omega$$

$$\text{s.t.} \sum_{p\in P_{rs}} y_p = d_{rs} \; \forall \; (r,s) \in Z^2$$

$$y_p \geq 0 \; \forall \; p \in P$$

Previously on Traffic Network Equilibrium...

Suppose $\tau_p(\mathbf{y})$ denotes the travel time on path p given a path flow vector $\mathbf{y}$. From the KKT conditions, eliminating λ_p, for all $(r,s) \in Z^2$, $p \in P_{rs}$,

$$\tau_p(\mathbf{y}) \geq \mu_{rs}$$
$$y_p\left(\tau_p(\mathbf{y}) - \mu_{rs}\right) = 0$$

From the above equations, μ_{rs} is the length of the shortest path.

If $y_p > 0$, then path p must be shortest. If $y_p = 0$, the travel time of path p must be at least μ_{rs}. Voila! Wardrop Principle.

Previously on Traffic Network Equilibrium...

Let the feasible region of path flows be represented as

$$Y = \left\{ \mathbf{y} : \sum_{p \in P_{rs}} y_p = d_{rs} \,\forall\, (r,s) \in Z^2, y_p \geq 0 \,\forall\, p \in P \right\}$$

The second-best tolling problem with VI-based equilibrium constraints can be written as

$$\min_{\mathbf{y},\mathbf{c}} \sum_{p \in P} y_p \tau_p(\mathbf{y})$$

$$\text{s.t. } \mathbf{y} \in Y$$

$$c_{ij} = 0 \,\forall\, (i,j) \in A'$$

$$[\boldsymbol{\tau}(\mathbf{y}) + \Delta^T \mathbf{c}]^T (\mathbf{y}' - \mathbf{y}) \geq 0 \,\forall\, \mathbf{y}' \in Y$$

Note that the tolls do not feature in the objective since they are **transfer payments**. We assume that they are returned to the system and hence it does not matter how much toll is collected.

Previously on Traffic Network Equilibrium...

If the link delay functions are separable and non-decreasing, the VIs can be replaced with the KKT conditions of the Beckmann formulation.

$$\min_{\mathbf{y},\mathbf{c},\boldsymbol{\mu}} \sum_{p \in P} y_p \tau_p(\mathbf{y})$$

$$\text{s.t. } \mathbf{y} \in Y$$

$$c_{ij} = 0 \,\forall\, (i,j) \in A'$$

$$\tau_p(\mathbf{y}) + \sum_{(i,j) \in A} \delta_{ij}^p c_{ij} \geq \mu_{rs} \,\forall\, (r,s) \in Z^2, p \in P_{rs}$$

$$y_p\left(\tau_p(\mathbf{y}) + \sum_{(i,j) \in A} \delta_{ij}^p c_{ij} - \mu_{rs}\right) = 0 \,\forall\, (r,s) \in Z^2, p \in P_{rs}$$

Lecture Outline

Lecture Outline

Solving Convex Programs

Solving Convex Programs

Introduction

How can we find the optimal solution of an unconstrained convex program?

Most methods are algorithmic in nature, i.e., we start with a feasible solution and improve it till we reach or are close to the optimal value.

Solving Convex Programs

Descent Directions

Definition

A sequence $\mathbf{x}^1, \mathbf{x}^2, \ldots$ is said to be descending if $f(\mathbf{x}^1) > f(\mathbf{x}^2) > \ldots$

Algorithms used to solve convex program try to generate a descending sequence of feasible values. The general update rule in such algorithms is

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \eta_k \boldsymbol{\varphi}^k$$

where η_k is the step size and $\boldsymbol{\varphi}^k$ is a direction vector in iteration k.

Will we always discover an optimal solution using such an algorithm?

Definition

A direction vector φ^k is said to be a descent direction if $\nabla f(\mathbf{x}^k)^T \varphi^k < 0$ for all k.

$\nabla f(\mathbf{x}^k)^T \varphi^k < 0$ is a measure of how much the objective decreases if we take a step in the direction of φ^k.

Proposition

$-\nabla f(\mathbf{x}^k)$ *is always a descent direction as long as the gradient is non-zero*

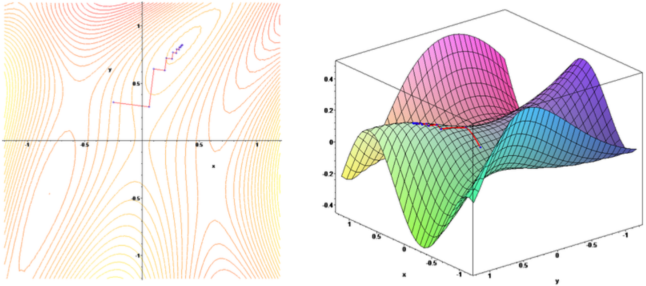
Thus, assuming that we take small steps in the direction of opposite to the gradient, we can guarantee that the algorithm generates a descending sequence.

Solving Convex Programs

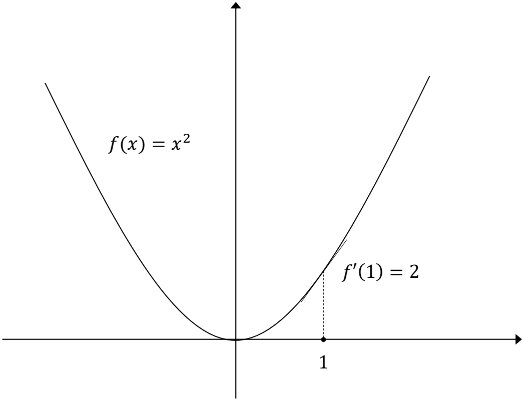
Gradient Descent

These algorithms are called **gradient descent methods** and the update rule can be written as

$$\mathbf{x}^{k+1} = \mathbf{x}^{k} - \eta_k \nabla f(\mathbf{x}^{k})$$



What happens when you take a large step along a descent direction?



Hence, the step size plays a major role in the convergence and rate of convergence of the algorithm.

Solving Convex Programs

Newton's Method

If second derivatives are available and the Hessian is positive definite, one can modify the descent direction to obtain a faster algorithm called the Newton's method.

This method approximates the function with a quadratic at each point and updates the decision variables as

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \eta_k \nabla^2 f(\mathbf{x}^k)^{-1} \nabla f(\mathbf{x}^k)$$

Method of Successive Averages

Method of Successive Averages

General Approach to Compute Equilibrium

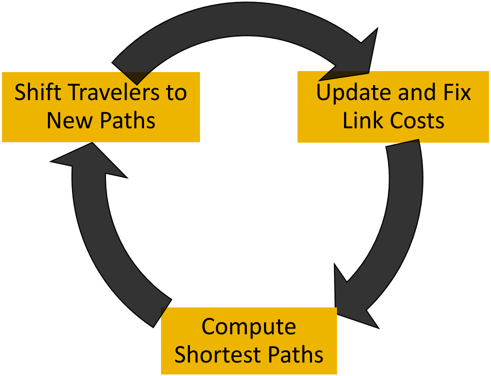
To solve the Beckmann formulation, we cannot simply use any of the earlier off-the-shelf method since the problem has constraints.

Most methods that we discuss are gradient descent-type approaches but we will pick step sizes in such a way that we stay within the feasible region at each iteration.

Method of Successive Averages

General Approach to Compute Equilibrium

All equilibrium algorithms typically involve the following strategy



Method of Successive Averages

General Approach to Compute Equilibrium

1. Initialize the algorithm with a feasible path ($\mathbf{y}$) and link flow ($\mathbf{x}$) solution
2. Compute the link delays using the link flows $\mathbf{x}$
3. Find the shortest paths between all OD pairs
4. For every OD pair, assign d_{rs} to the shortest path between (r, s) This step is called the **all-or-nothing assignment**. Denote the resulting path and link flow vectors by $\hat{\mathbf{y}}$ and $\hat{\mathbf{x}}$
5. If we reach or are close to the optimum, stop. Else, update the link flows $\mathbf{x}^{k+1} = \eta_k \hat{\mathbf{x}}^k + (1 - \eta_k)\mathbf{x}^k$, where $\eta_k \in [0, 1]$ and return to 2

Method of Successive Averages

General Approach to Compute Equilibrium

- Notice that we never violate feasibility. (Why?)
- Step 5 in which the link flow vector is convex combination of the old solution and the all-or-nothing assignment is similar to the gradient descent discussed earlier.

$$\begin{aligned}\mathbf{x}^{k+1} &= \eta_k \hat{\mathbf{x}}^k + (1-\eta_k)\mathbf{x}^k \\ &= \mathbf{x}^k + \eta_k(\hat{\mathbf{x}}^k - \mathbf{x}^k)\end{aligned}$$

Why is $(\hat{\mathbf{x}}^k - \mathbf{x}^k)$ a descent direction? The gradient of Beckmann function is $\mathbf{t}(\mathbf{x}^k)$ and $\mathbf{t}(\mathbf{x}^k)^T(\hat{\mathbf{x}}^k - \mathbf{x}^k) \leq 0$ since $\hat{\mathbf{x}}$ are flows on the shortest paths.

Method of Successive Averages

Step Sizes

Different equilibrium algorithms can be designed just by modifying the way step sizes are chosen in Step 5.

Just as we saw earlier, selecting a very big or small step size can be troublesome.

The method of successive averages is the simplest algorithm in which the step sizes satisfy

$$\sum_k \eta_k = \infty, \sum_k \eta_k^2 \leq \infty$$

Can you think of a sequence that satisfies the above conditions?

$$\eta_k = \frac{1}{k}$$

$\sum_k \frac{1}{k^2} = \frac{\pi^2}{6}$, the Basel problem, was an open problem for 90 years!

Lecture Outline

Measuring Convergence

Measuring Convergence

Introduction

Since the equilibrium flows are a solution to a non-linear program, the solutions can be in the interior of the feasible region.

As computers represent numbers using finite precision, one can only expect flows that are nearly optimal.

When do we terminate our equilibrium algorithms? Can we stop when the differences in the link flows are negligible?

Measuring Convergence

Introduction

Before solving the problem we do not know the optimal flow or the optimal value of the Beckmann function (or a lower bound) to estimate $\|\mathbf{x} - \mathbf{x}^*\|$ or $f(\mathbf{x}) - f(\mathbf{x}^*)$.

Instead, the convergence criteria can be linked to the Wardrop principle. Two most popular gap measures for convergence are

- Relative gap
- Average excess cost

Measuring Convergence

Relative Gap

At the start of any iteration, we can compute the TSTT of the current solution $\sum_{(i,j)\in A} x_{ij} t_{ij}(x_{ij})$.

Define the shortest path travel time $SPTT = \sum_{(i,j)\in A} \hat{x}_{ij} t_{ij}(x_{ij})$ as the total travel time when the travel times are fixed at $t_{ij}(x_{ij})$ and all users are loaded on corresponding shortest paths.

Note that $SPTT < TSTT$ except at equilibrium (Why?)

$$\text{Relative Gap} = \frac{TSTT}{SPTT} - 1$$

The relative gap measure has no units and no intuitive meaning. For most applications, equilibrium algorithms are terminated if the gap is less than 10^{-4}.

Measuring Convergence

Average Excess Cost

The average excess cost (AEC) is defined as

$$AEC = \frac{TSTT - SPTT}{\sum_{(r,s) \in Z^2} d_{rs}}$$

and indicates the average difference between a traveler's path and the shortest path available to him or her.

AEC can be measured in min or seconds. This measure is somewhat related to ϵ-Nash Equilibria.

Measuring Convergence

Summary

MSA(G)

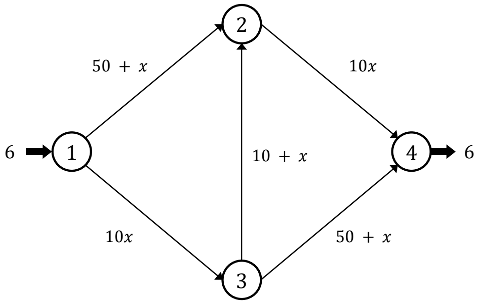
```
k ← 1
Find a feasible x̂

while Relative Gap > 10^{-4} do
    x ← (1/k) x̂ + (1 − 1/k) x
    Update t(x)
    x̂ ← 0
    for r ∈ Z do
        DIJKSTRA (G, r)
        for s ∈ Z, (i, j) ∈ p*_rs do
            x̂_ij ← x̂_ij + d_rs
        end for
    end for
    Relative Gap ← TSTT/SPTT − 1
    k ← k + 1
end while
```

Measuring Convergence

Example 1

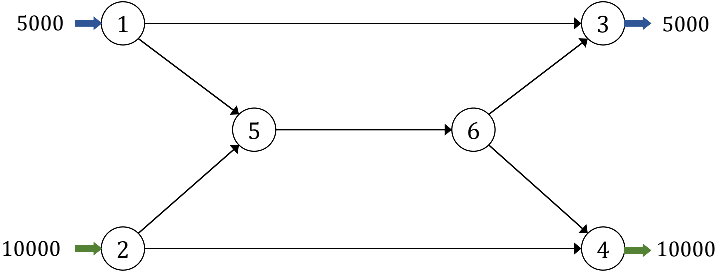
Compute the UE flows in the following network using MSA



Measuring Convergence

Example 2

Find the UE flows using MSA in the following network where the delay function on each link is $10 + x/100$



Lecture Outline

Proof of Convergence

Proof of Convergence

Background

MSA was widely used even before it was formally proved by Powell and Sheffi in 1982. The proof relies on an extended version of the Mean Value Theorem.

Theorem

If f and f' are continuous on $[a, b]$ and differentiable on (a, b) then $\exists\, c \in (a, b)$ such that

$$f(b) = f(a) + f'(a)(b - a) + \frac{1}{2} f''(c)(b - a)^2$$

In higher dimensions, this can be written as

$$f(\mathbf{y}) = f(\mathbf{x}) + \nabla f(\mathbf{x})^T (\mathbf{y} - \mathbf{x}) + \frac{1}{2} (\mathbf{y} - \mathbf{x})^T \nabla^2 f(\mathbf{z})(\mathbf{y} - \mathbf{x})$$

for some $\mathbf{z}$ that lies on the line segment between $\mathbf{x}$ and $\mathbf{y}$.

Proof of Convergence

Bounding the Extent of Descent

At iteration k,

$$f(\mathbf{x}^{k+1}) = f(\mathbf{x}^k) + \nabla f(\mathbf{x}^k)^T(\mathbf{x}^{k+1} - \mathbf{x}^k) + \frac{1}{2}(\mathbf{x}^{k+1} - \mathbf{x}^k)^T \nabla^2 f(\mathbf{z})(\mathbf{x}^{k+1} - \mathbf{x}^k)$$

where $\mathbf{z}$ lies on the line segment between $\mathbf{x}^k$ and $\mathbf{x}^{k+1}$.

$$f(\mathbf{x}^{k+1}) = f(\mathbf{x}^k) + \nabla f(\mathbf{x}^k)^T \eta_k \boldsymbol{\varphi}^k + \frac{1}{2}\eta_k{}^2 \boldsymbol{\varphi}^{k\,T} \nabla^2 f(\mathbf{z}) \boldsymbol{\varphi}^k$$

Adding the above equations for $k = 1, 2, \ldots, K$

$$f(\mathbf{x}^{K+1}) = f(\mathbf{x}^1) + \sum_{k=1}^{K} \eta_k \nabla f(\mathbf{x}^k)^T \boldsymbol{\varphi}^k + \sum_{k=1}^{K} \frac{1}{2}\eta_k{}^2 \boldsymbol{\varphi}^{k\,T} \nabla^2 f(\mathbf{z}) \boldsymbol{\varphi}^k$$

Taking limits as $K \to \infty$, we can show that $-\infty < \nabla f(\mathbf{x}^k)^T \boldsymbol{\varphi}^k \leq 0$. (How?)

Your Moment of Zen