CE 272
Traffic Network Equilibrium

Lecture 9
Frank-Wolfe Algorithm

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...

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.

Previously on Traffic Network 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

Previously on Traffic Network Equilibrium...

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.

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

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.

Lecture Outline

1. Frank-Wolfe Algorithm
2. Examples
3. Conjugate Frank-Wolfe

Lecture Outline

Frank-Wolfe Algorithm

Frank-Wolfe Algorithm

Introduction

In MSA, we showed that $\hat{\mathbf{x}}^k - \mathbf{x}^k$ is a descent direction at every iteration k. But does the objective always decrease?

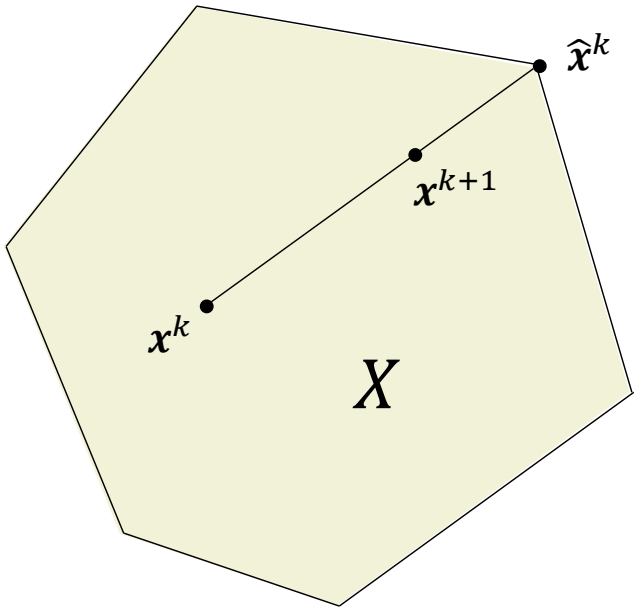
The step size is easy to compute but **can we do better**?

Frank-Wolfe Algorithm

Introduction

Recall that the link flows were updated using

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



For different values of η_k, we get different $\mathbf{x}^{k+1}$ along the above line segment. Can we pick an optimal η_k?

Frank-Wolfe Algorithm

Introduction

Unlike MSA which uses pre-determined step sizes, in Frank-Wolfe (FW) method we select a step size that minimizes the Beckmann function. Let's suppress the index k for brevity.

- **Objective:** $f(\eta) = \sum_{(i,j)\in A} \int_0^{\eta\hat{x}_{ij}+(1-\eta)x_{ij}} t_{ij}(\omega)\, d\omega$
- **Decision Variable:** η
- **Constraint:** $\eta \in [0, 1]$

Note that the current link flows $\mathbf{x}$ and the all-or-nothing link flows $\hat{\mathbf{x}}$ are constant in the above optimization model.

Frank-Wolfe Algorithm

First-order Conditions

The objective is convex in η. (Why?) So the first-order conditions imply that $f'(\eta) = 0$ at an interior η.

$$\frac{d}{d\eta} \sum_{(i,j)\in A} \int_0^{\eta\hat{x}_{ij}+(1-\eta)x_{ij}} t_{ij}(\omega)\, d\omega = 0$$

$$\Rightarrow \sum_{(i,j)\in A} t_{ij}\left(\eta\hat{x}_{ij} + (1-\eta)x_{ij}\right)\left(\hat{x}_{ij} - x_{ij}\right) = 0$$

Solving this equation provides the optimal η (assuming it lies in the interior) which then gives the next iterate.

Within each FW iteration, more computations are needed compared to MSA but the overall number of iterations are reduced.

Frank-Wolfe Algorithm

Optimizing the Step Size

Finding an analytical solution to the earlier equation is difficult unless the travel time functions are linear. Instead, the optimal step size can be calculated using one of the following methods

- Bisection
- Newton's method

Newton's method, as discussed in the last class, requires $f''(\eta)$ which is easy to compute. However, since $\eta \in [0, 1]$, we might need to project it back to the feasible region if it exceeds 1 or falls below 0.

Frank-Wolfe Algorithm

Optimizing the Step Size

BISECTION$(G, \mathbf{x}, \hat{\mathbf{x}})$

$\underline{\eta} = 0, \bar{\eta} = 1$

while $\bar{\eta} - \underline{\eta} > \epsilon$ **do**
  $\eta \leftarrow \frac{1}{2}(\underline{\eta} + \bar{\eta})$
  if $\sum_{(i,j)\in A} t_{ij}\left(\eta\hat{x}_{ij} + (1-\eta)x_{ij}\right)\left(\hat{x}_{ij} - x_{ij}\right) > 0$ **then**
    $\bar{\eta} \leftarrow \eta$
  else
    $\underline{\eta} \leftarrow \eta$
  end if
end while
return η

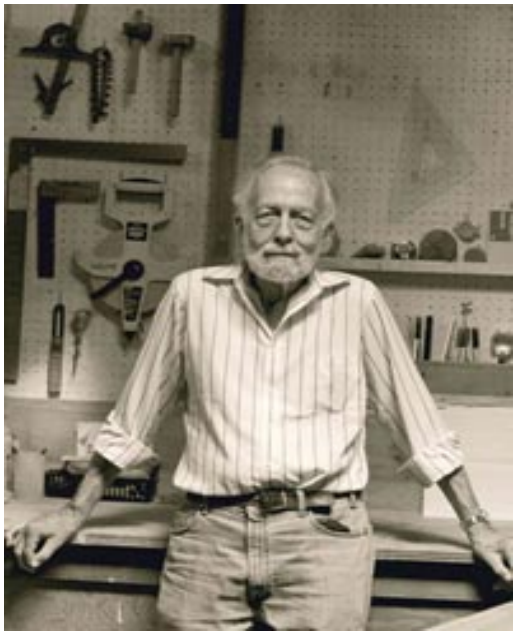
Will the method work if the optimum occurs at the boundary?

Frank-Wolfe Algorithm

History Break

The Frank-Wolfe algorithm was originally proposed in 1956 for quadratic programs and convex programs with linear constraints.

Marguerite Frank



Philip Wolfe

https://www.youtube.com/watch?v=24e08AX9Eww

Optimizing the Step Size

Summary

FRANK-WOLFE(G)

$k \leftarrow 1$
Find a feasible $\hat{\mathbf{x}}$

while Relative Gap $> 10^{-4}$ **do**
  if $k = 1$ **then** $\eta \leftarrow 1$ **else** $\eta \leftarrow$ BISECTION$(G, \mathbf{x}, \hat{\mathbf{x}})$
  $\mathbf{x} \leftarrow \eta\hat{\mathbf{x}} + (1 - \eta)\mathbf{x}$
  Update $\mathbf{t}(\mathbf{x})$
  $\hat{\mathbf{x}} \leftarrow \mathbf{0}$
  for $r \in Z$ **do**
    DIJKSTRA (G, r)
    for $s \in Z, (i, j) \in p^*_{rs}$ **do**
      $\hat{x}_{ij} \leftarrow \hat{x}_{ij} + d_{rs}$
    end for
  end for
  Relative Gap $\leftarrow TSTT/SPTT - 1$
  $k \leftarrow k + 1$
end while

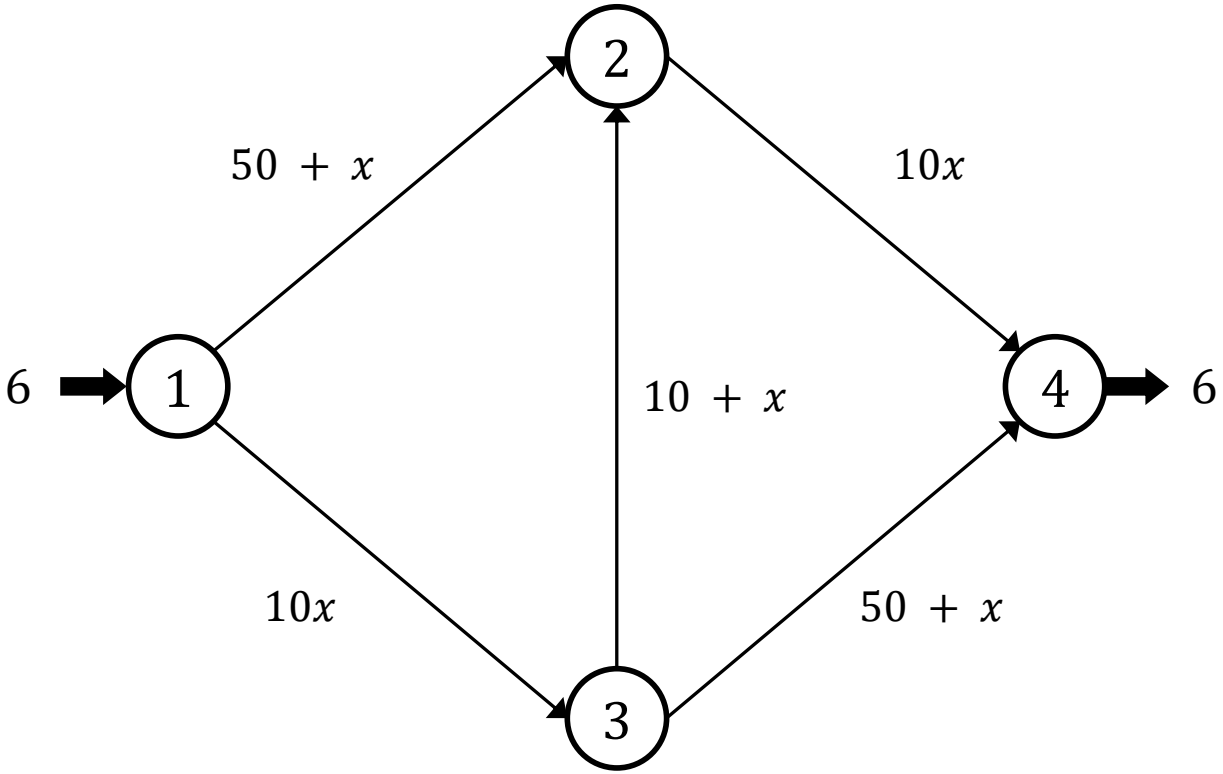
Lecture Outline

Examples

Examples

Example 1

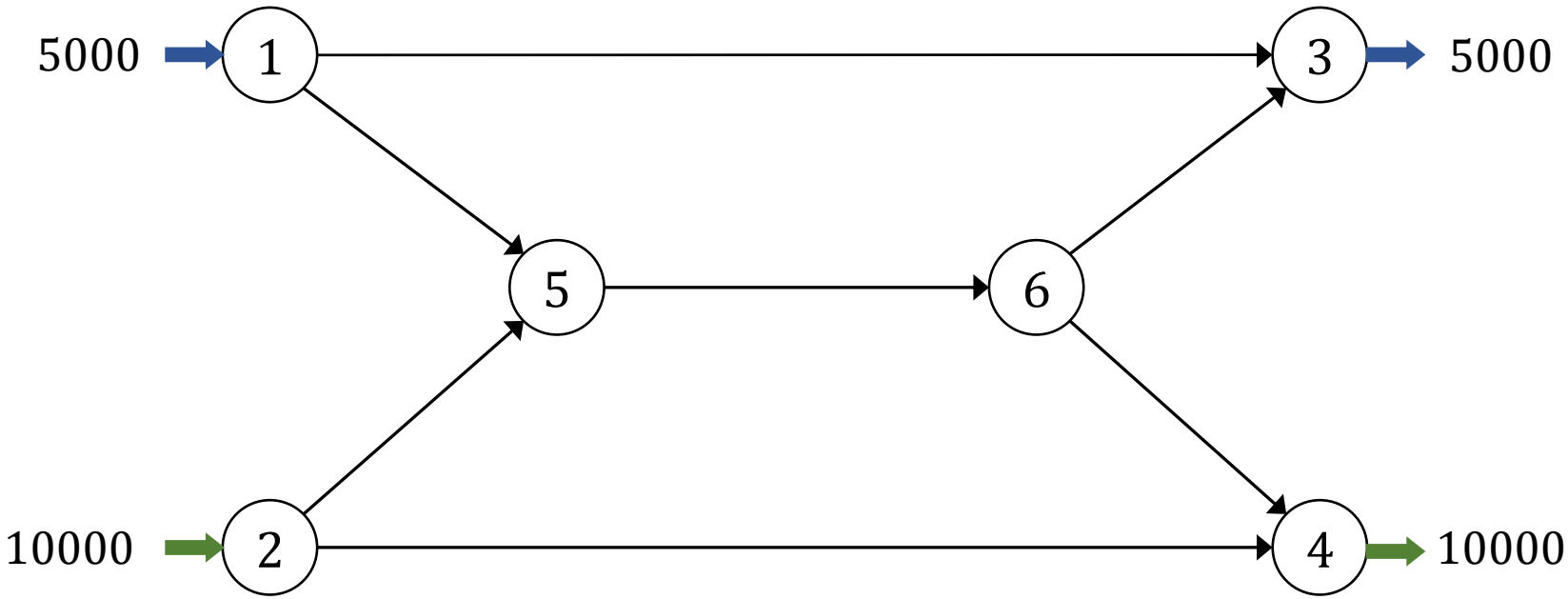
Find the UE flows in the following network using the FW algorithm



Examples

Example 2

Find the UE flows in the following network using the FW algorithm



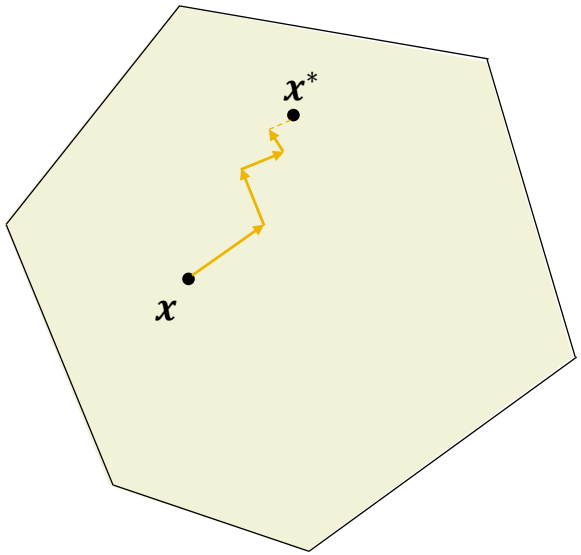
Lecture Outline

Conjugate Frank-Wolfe

Conjugate Frank-Wolfe

Drawbacks of FW

FW and MSA both have a drawback of zig-zagging since they are always constrained to take steps in the directions of corner points.



For this reason, they perform very well during the initial iterations but the rate of convergence decreases over time.

Can we find a search direction which is aligned along (or close to) $\mathbf{x}^* - \mathbf{x}$?

Conjugate Frank-Wolfe

Intuition

Consider a quadratic program of the form $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x} - \mathbf{b}^T\mathbf{x}$

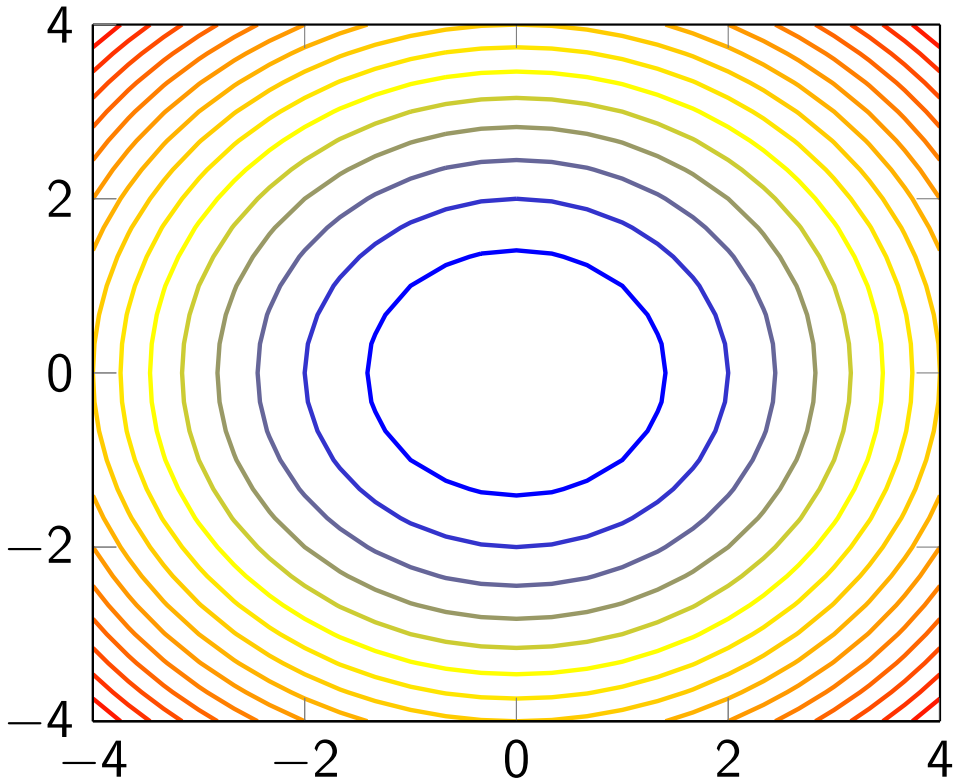
The gradient of f is $\nabla f(\mathbf{x}) = \mathbf{A}\mathbf{x} - \mathbf{b}$. Hence, the optimal solution occurs at $\mathbf{x}^* = \mathbf{A}^{-1}\mathbf{b}$.

Suppose $\mathbf{A}$ is a diagonal matrix, how many operations are needed to compute the optimal solution?

Conjugate Frank-Wolfe

Intuition

One can visualize this by considering the function $f(x_1, x_2) = x_1^2 + x_2^2$. Here, $\mathbf{A} = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$ and $\mathbf{b} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$.

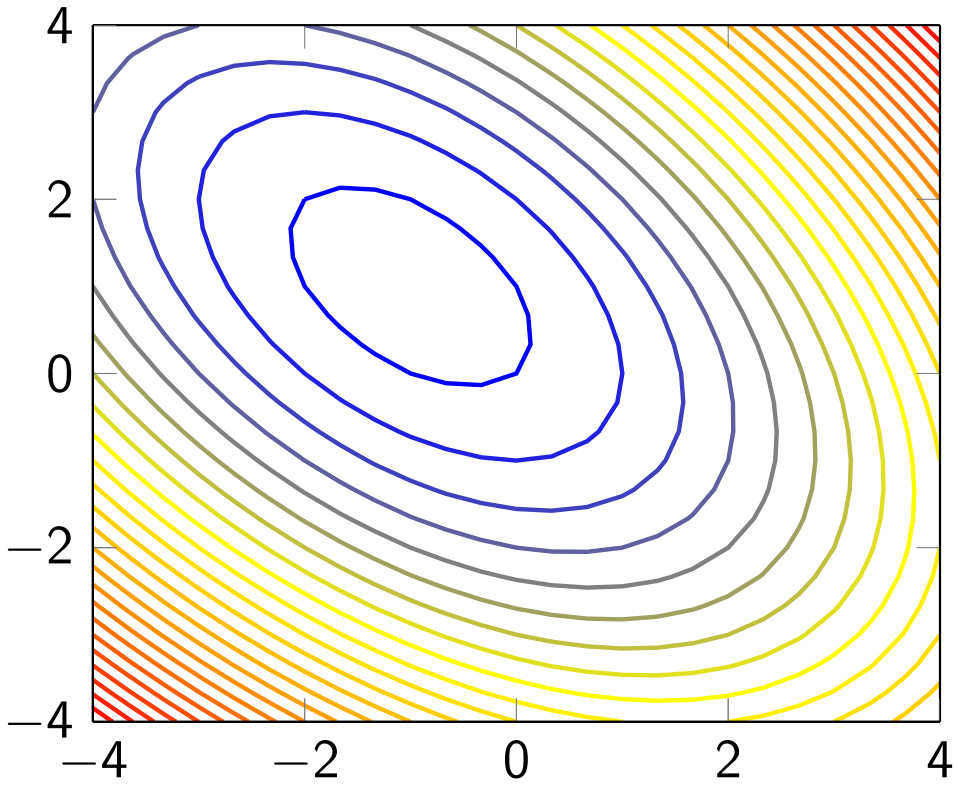


The optimal value of each coordinate can be set sequentially and we can get to the origin in 2 steps instead of moving in the direction of $-\nabla f(\mathbf{x}^k)$.

Conjugate Frank-Wolfe

Intuition

Now consider the function $f(\mathbf{x}) = x_1^2 + x_2^2 + x_1 x_2 + x_1 - x_2$. For this function, $\mathbf{A} = \left[\begin{smallmatrix} 2 & 1 \\ 1 & 2 \end{smallmatrix}\right]$ and $\mathbf{b} = \left[\begin{smallmatrix} -1 \\ 1 \end{smallmatrix}\right]$.



Can we reach the optimum in two steps?

Conjugate Frank-Wolfe

Intuition

Think of a transformation **Px** that maps these ellipses to circles and we move along vectors that are orthogonal in the transformed space. These directions in the original space are called *conjugate vectors*.

The **P** matrix can be defined using the singular value decomposition of **A**.

Conjugate Frank-Wolfe

Conjugacy

Definition

Vectors φ_i and φ_j are conjugate to a symmetric positive definite matrix $\mathbf{A}$ if

$$\varphi_i^T \mathbf{A} \varphi_j = 0 \, \forall \, i \neq j$$

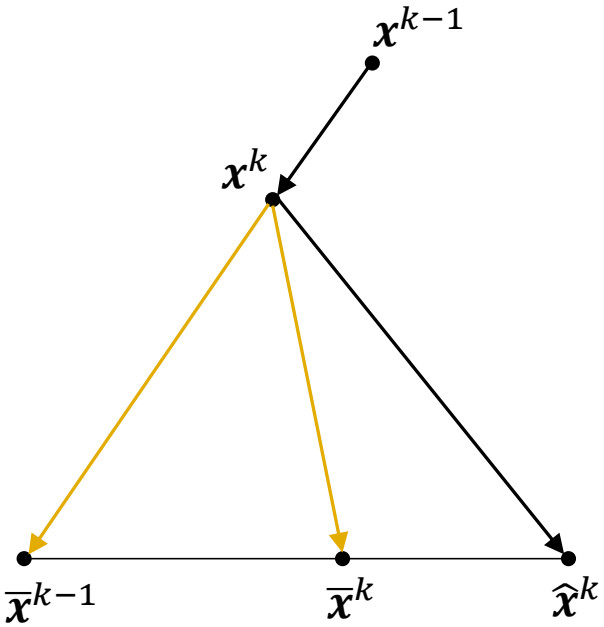
Notice that if $\mathbf{A}$ is the identify matrix, then the vectors are orthogonal to each other.

In Conjugate Frank-Wolfe (CFW) method, we make sure that consecutive search directions are conjugate to the Hessian of the objective (Beckmann function).

Conjugate Frank-Wolfe

Selecting the Direction

We will use the following notation to describe different points:
$\mathbf{x}$ – The current iterate
$\hat{\mathbf{x}}$ – All-or-nothing flows
$\bar{\mathbf{x}}$ – Target direction towards which we want to move in CFW

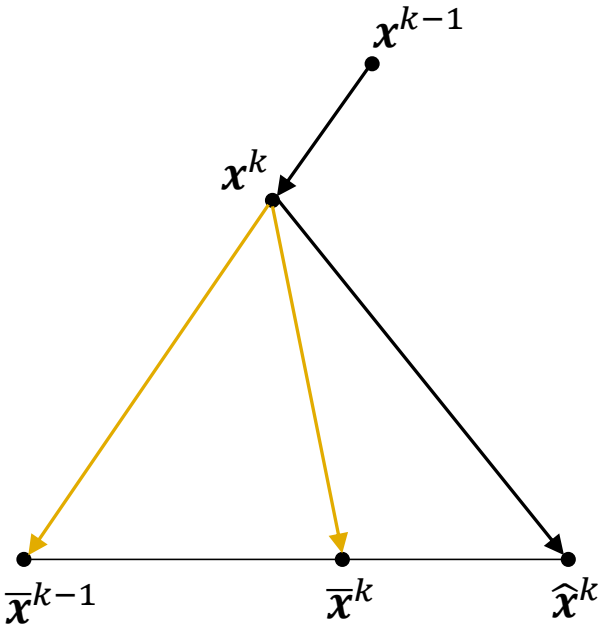


The goal is to ensure that the two yellow vectors (which are directions taken by the algorithm in two consecutive iterations) are conjugate to the Hessian.

What about feasibility? $\bar{\mathbf{x}}^k$ is chosen to lie between $\bar{\mathbf{x}}^{k-1}$ and $\hat{\mathbf{x}}^k$.

Conjugate Frank-Wolfe

Selecting the Direction



From the definition of conjugate vectors,

$$(\bar{\mathbf{x}}^{k-1} - \mathbf{x}^k)^T \nabla^2 f(\mathbf{x}^k)(\bar{\mathbf{x}}^k - \mathbf{x}^k) = 0$$

But $\bar{\mathbf{x}}^k = \theta \bar{\mathbf{x}}^{k-1} + (1-\theta)\hat{\mathbf{x}}^k$.

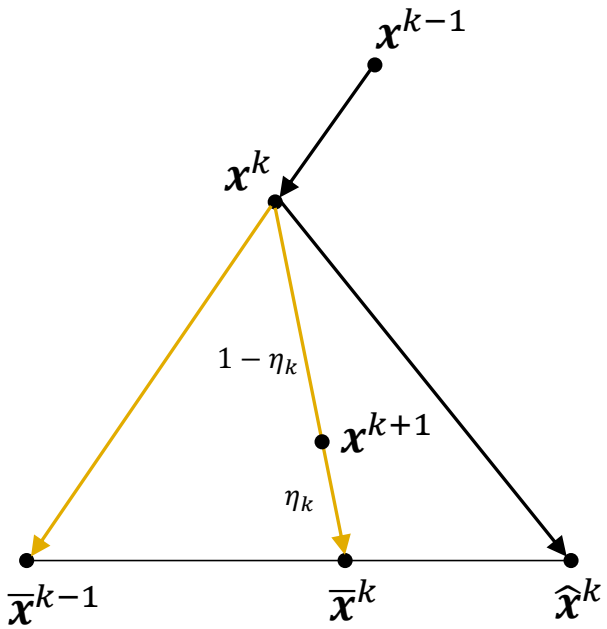
Plugging this in the above equation and solving for θ we get,

$$\theta = -\frac{(\bar{\mathbf{x}}^{k-1} - \mathbf{x}^k)^T \nabla^2 f(\mathbf{x}^k)(\hat{\mathbf{x}}^k - \mathbf{x}^k)}{(\bar{\mathbf{x}}^{k-1} - \mathbf{x}^k)^T \nabla^2 f(\mathbf{x}^k)(\bar{\mathbf{x}}^{k-1} - \hat{\mathbf{x}}^k)}$$

Conjugate Frank-Wolfe

Selecting the Step Size

$\bar{\mathbf{x}}^k$ gives us the direction but not the next iterate. We still need to decide how far to move along this direction.



Just as with FW, the step search η is found by minimizing the Beckmann function along the direction $\bar{\mathbf{x}}^k - \mathbf{x}^k$.

Supplementary Reading

Mitradjieva, M., & Lindberg, P. O. (2013). The stiff is moving-conjugate direction Frank-Wolfe Methods with applications to traffic assignment. Transportation Science, 47(2), 280-293.

Your Moment of Zen

Now that's what you call an accurate VMS!