§3 Der Algorithmus von Frank-Wolfe für die (konvexe) Optimierung mit linearen Nebenbedingungen (1956)

3.1 Der allgemeine Fall

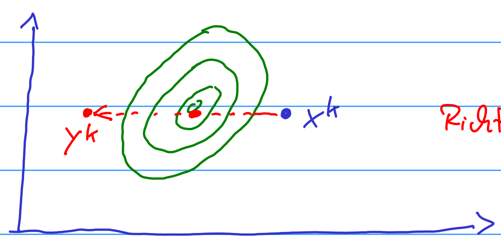
$\min f(x)$ unter $Ax = b \quad x \geq 0$

Algorithmus ist iterativ und erzeugt eine Folge von Punkten $x^0, x^1, \dots$ wobei x^{k+1} wie folgt aus x^k berechnet wird

(1) Bestimme eine Optimallösung y^k des LP

$$\min \nabla_x f(x^k)^T x \quad \leftarrow \text{lineare Fkt}$$
$$\text{unter } Ax = b$$
$$x \geq 0$$

(2) Wähle x^{k+1} als Minimierer von $f(x)$ auf dem Intervall $[x^k, y^k]$ "Line-Search" $\leftarrow$ 1-dim Problem



Richtung $y^k - x^k$

3.1 SATZ Sei f stetig, diffbar, konvex und $X := \{x \in \mathbb{R}^n \mid Ax = b, x \geq 0\}$ beschränkt

Dann enthält die Folge $x^0, x^1, \dots$ eine konvergente Teilfolge und jede solche Teilfolge konvergiert gegen ein globales Minimum von $f(x)$ auf X

Vollständiger Beweis siehe z.B. Skript ADM III vom SS 2006

Buch: Dimitris Bertsekas
Non Linear Programming
Athena Scientific 1999

David G. Luenberger, Yinyu Ye
Linear and NonLinear Programming
Springer, 2008

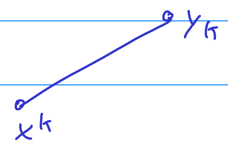
Es bleiben 2 Fragen

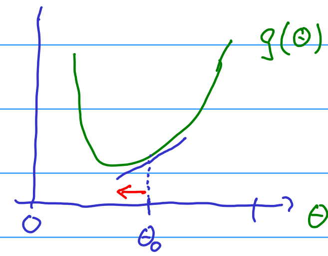
(A) Wie macht man die Line Search ?

(B) Wann bricht man mit der Folge der x^k ab ?

zu (A) Line Search

eine Möglichkeit: binäre Suche auf





$$g(\theta) := f(x^k + \theta(y^k - x^k))$$

binäre Suche $\hat{=}$ Minimierung der 1-dim Fkt $g(\theta)$

in jedem Zwischenpunkt muss dann die Ableitung in θ,

also $\frac{\partial g}{\partial \theta}(\theta)$ ausgewertet werden $\left[\frac{\partial g}{\partial \theta}(\theta) = \nabla_x f(x^k) + \theta(y^k - x^k)^T (y_k - x^k) \right]$

Ist sie > 0 so $\theta_{i+1} := \theta_i - \frac{1}{2^{i+1}} \| y^k - x^k \|$

< 0 ... $+$ ....

Abbruch falls Ableitung 0 ist oder das verbleibende Intervall

$1/2^i \, \| x^k - y^k \|$ klein genug $\Rightarrow$ $i \approx \log\left(\frac{1}{\varepsilon} \| x^k - y^k \| \right)$

Unter Umständen teuer (nicht beim SO)

daher allgemein auch andere Methoden verwendet $\to$ Num. Mathematik

zu (B) Abbruchkriterium

f konvex $\overset{\text{diffbar}}{\iff}$ $f(z) \geq f(x) + \nabla f(x)^T (z - x)$ (*)

x^* optimal $\Rightarrow$ $f(x^*) \geq f(x^k) + \nabla f(x^k)^T (x^* - x^k)$

(*) mit $z = x^*$, $x = x^k$

$$\geq f(x^k) + \underbrace{\nabla f(x^k)^T (y^k - x^k)}$$

y^k minimiert $\nabla f(x^k)^T x$

$$\Rightarrow \underbrace{f(x^k) - f(x^*)}_{\text{verbleibende Optimalitätslücke}} \leq \underbrace{|\nabla f(x^k)^T (y^k - x^k)|}_{\text{im Algo bekannte Größe}}$$

$\Rightarrow$ Optimalitätslücke kann im Verfahren kontrolliert werden

$\Rightarrow$ Abbruch kann entsprechend gestaltet werden

3.2 Spezialisierung auf das Systemoptimum (SO) und das User Equilibrium (UE)

im (SO) haben wir $\min \; C(x(f)) = \sum_{a \in A} x_a(f) \cdot \tau_a(x_a(f))$

$$\text{mit} \quad x_a(f) = \sum_{P \ni a} f_P \qquad \begin{array}{l} f \text{ Pfadfluss} \\ x \text{ Kantenfluss} \end{array}$$

$$\sum_{P \in \mathcal{P}_k} f_P = d_k \qquad \forall k$$

$$f_P \geq 0$$

Aus §2 (bei den Kuhn-Tucker Bed.) haben wir

$$\boxed{\frac{\partial C}{\partial f_P}(x(f)) = \sum_{a \in P} \frac{d}{dx_a} c_a(x_a)} \qquad \text{mit} \quad c_a(x_a) = x_a \cdot \tau_a(x_a)$$

Pfadlänge bzgl. marginaler Kantenkosten

Also ist $\nabla_x C(\bar{x})^T x = \sum_{a \in A} \frac{d}{dx_a} c_a(\bar{x}_a) \cdot x_a = \sum_{a \in A} \frac{d}{dx_a} c_a(\bar{x}_a) \sum_{P \ni a} f_P$

Summation vertauschen

$$= \underbrace{\sum_P \sum_{a \in P} \frac{d}{d x_a} c_a(\bar{x}_a)}_{\boxed{} \; = \; \sum_P \frac{\partial}{\partial f_P} (x(\bar{f}))} \cdot f_P$$

$\nearrow$

$\bar{f}$ Pfadfluss zu $\bar{x}$

$$= \nabla_f C(\bar{f})^T \cdot f$$

also $\nabla_f C(\bar{f})^T \cdot f = \nabla_x C(\bar{x})^T x$

$\uparrow$ in Pfadformulierung $\qquad$ $\uparrow$ in Kantenformulierung

$\Rightarrow$ Das zu lösende LP im Frank-Wolfe Algo kann in Kantenform gelöst werden (brauchen nicht exponentiell viele Wege in Formulierung der Zielfkt des LP), und die Bewertung der Kante ist $\frac{d}{d x_a} c_a(x_a^\ell) = x_a^\ell \cdot \tau_a'(x_a^\ell) + \tau_a(x_a^\ell)$ mit x^ℓ = Kantenfluss in momentaner Iteration ℓ

$\Rightarrow$ LP ist

$$\left.\begin{aligned} &\min \nabla_x C(x^\ell)^T x \\ &\text{unter } \sum_{P \in \mathcal{P}_k} f_P^\ell = d_k \quad \forall k \\ &f_P \geq 0 \end{aligned}\right\} \mathrel{\hat{=}}$$

Min Cost Multi Commodity Fluss Problem mit Kantenkosten $c_a'(x_a^\ell)$

da keine Kantenkapazitäten $\Rightarrow$ Problem zerfällt in #commodities viele Kürzeste Wege Probleme bzgl. Kantenkosten $c_a'(x_a^\ell)$

Lösung ist ein kürzester Weg mit Fluss d_k pro Commodity k

3.2 SATZ: Der erste Teil des Frank-Wolfe Algo reduziert sich in jeder Iteration ℓ auf die Berechnung eines kürzesten Weges für jedes Quelle-Senke Paar bzgl. der Kantenkosten $c_a'(x_a^\ell)$

Line Search

Sei y^ℓ Optimallösung des 1 Teil

$\Rightarrow y^\ell$ besteht aus je einem Weg Q_k für commodity k

mit $y_{Q_k} = d_k \qquad y_P = 0$ sonst

momentane Lösung y^ℓ besteht aus gewissen Wegen $P \in \mathcal{P}_k$ mit $\sum_{P \in \mathcal{P}_k} f_P = d_k$

neue Lösung $f^{\ell+1}$ ergibt sich als $f^\ell + \Theta(y^\ell - f^\ell)$

a) Fließt bereits Fluss auf Q_k bzgl. f^ℓ, d.h. $f^\ell_{Q_k} > 0$

so ergibt sich für $f^{\ell+1}_{Q_k} = f^\ell_{Q_k} + \Theta(d_k - f^\ell_{Q_k}) = \underbrace{(1-\Theta) f^\ell_{Q_k} + \Theta d_k}$

Konvexkomb. von momentaner Lösung und Lösung des LP

b) Fließt kein Fluss auf Q_k bzgl f^ℓ

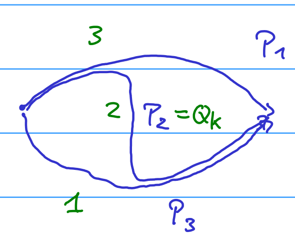
$\Rightarrow f^{\ell+1}_{Q_k} = f^\ell_{Q_k} + \Theta(y^\ell_{Q_k} - f^\ell_{Q_k}) = \Theta d_k$

c) Für alle $P \in \mathcal{P}_k \quad P \neq Q_k$ wird Fluss weggenommen

$f_P^{\ell+1} = f_P^\ell + \Theta(0 - f_P^\ell) = (1-\Theta) f_P^\ell$

Bsp:

$d_k = 6$



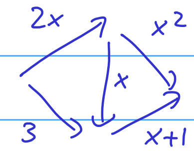
$\Rightarrow$ neuer Fluss für Commodity k ergibt sich zu

$$\begin{pmatrix} 3 \\ 2 \\ 1 \end{pmatrix} (1-\Theta) + \Theta \begin{pmatrix} 0 \\ 6 \\ 0 \end{pmatrix} = \begin{pmatrix} 3-3\Theta \\ 2+4\Theta \\ 1-\Theta \end{pmatrix} \geq 0$$

Optimaler Wert von Θ ergibt sich aus der Line Search

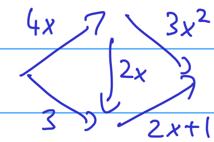
bzgl. $\nabla_x C(x^k + \Theta(y^k - x^k))^T (y^k - x^k)$ (*)

im Beispiel:



$\tau_a(x_a)$

$\longrightarrow$ marginale Kantenkosten



$c'_a(x_a)$

$$\nabla C(x) = \begin{pmatrix} 4x \\ 3 \\ 2x \\ 3x^2 \\ 2x+1 \end{pmatrix}$$

Wert von (*) für Θ ist

$$\begin{pmatrix} 4(5+\Theta) \\ 3(1-\Theta) \\ 2(3+4\Theta) \\ 3(3-3\Theta)^2 \\ 2(3+3\Theta)+1 \end{pmatrix}^T \begin{pmatrix} 6 & -5 \\ 0 & -1 \\ 6 & -2 \\ 0 & -3 \\ 6 & -3 \end{pmatrix} = g(\Theta)$$

$\Downarrow$

3.3 Proposition: Line Search benötigt nur die Fluss führenden Wege plus die neu berechneten Wege aus dem LP (also nicht alle exponentiell vielen Wege) *

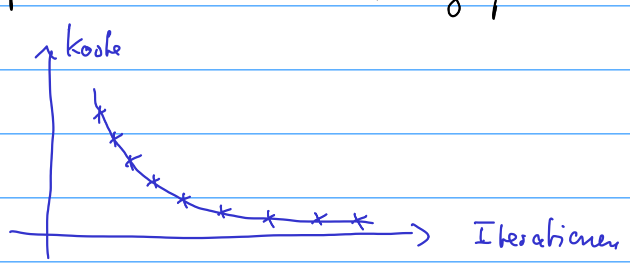
Die Berechnung des optimalen Θ kann im Kantenraum erfolgen

(*): Startet man mit einem Weg pro Commodity so ist die Gesamtzahl der Fluss führenden Wege nach N Iterationen $\leq N \cdot \#$ Commodities

Abbruch entweder über Optimalitätslücke (§ 3.1) oder über die Beobachtung der Weglängen bzgl. $c'_a(x_a^\ell)$.

Abbruch wenn sich neu berechnete Wege kaum von den Kosten der Fluss-führenden unterscheiden, dies nutzt die UE-Interpretation des SO:

Typischerweise ist der Gewinn groß in den ersten Iterationen, später kleiner



Bemerkungen

(1) Das Verfahren von Frank-Wolfe wird in diesem Zusammenhang auch als convex-combinations Algorithmus bezeichnet

(2) Es kann auch für die Berechnung des UE benutzt werden

mit $C(x(f)) = \sum_{a \in A} \underbrace{\int_0^{x_a(f)} \tau_a(t)\,dt}_{\text{Kantenkosten } C_a(x_a)}$

Nebenbedingungen bleiben gleich $\Rightarrow \frac{d}{d x_a} C_a(x_a) = \tau_a(x_a)$

$\Downarrow$

3.4. SATZ: Bei der Berechnung des UE mit Frank Wolfe reduziert sich der erste Teil auf die Berechnung eines kürzesten Weges bzgl $\tau_a(x_a)$ für jedes Quelle-Senke Paar

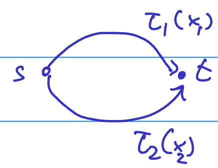
(3) Ein naheliegender "Fixpunkt-Ansatz" funktioniert i. A. nicht

f^{l+1} = opt. Fluss bzgl der Fahrzeiten $\tau_a(x_a^l(f^l))$

aus der vorigen Iteration

da Oszillation eintreten kann

3.1 Aufgabe 3.1 Berechne Folge der Flüsse mit dem Fixpunkt Ansatz und Frank Wolfe für das Bsp



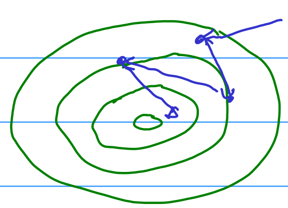
$d = 1$

$\tau_1(x_1) = x_1$

$\tau_2(x_2) = x_2$

Geeignete Wahl des Startflusses $\Rightarrow$ Oszillation

(4) Bei Frank-Wolfe kann "Zick-Zack" Verhalten auftreten



Es gibt verschiedene Verbesserungen, die das vermeiden, z.B. die Partan Methode

(LeBlanc, Helgason, Boyce 85)

intelligentere Line Search durch Ausnutzung der beiden letzten Iterationen, x^ℓ, y^ℓ, $x^{\ell-1}, y^{\ell-1}$

$\Rightarrow$ 30% Verbesserung der Rechenzeit in unseren Rechnungen

3.2 Aufgabe 3.2 Was muss in der Spezialisierung von Frank Wolfe auf SO / UE geändert werden, wenn der Fluss pro Kante durch eine Kapazität beschränkt ist, d.h. $x_a(f) \le u_a$?

3.3 Das eingeschränkte Systemoptimum (CSO)

constrained

Motivation

Vermeide den Preis der Anarchie $\frac{UE}{SO}$

Vermeide Unfairness des SO (einzelne können Routen mit sehr langen Fahrzeiten bekommen)

unfairness einer Routenzuweisung f

$$= \max_k \frac{\max \{ \tau_P(f) \mid P \in \mathcal{P}_k \}}{\text{Fahrzeit } L_k(UE) \text{ im } UE} =: \text{unfairness}(f)$$

Bemerkung: unfairness (UE) = 1

unfairness (SO) = ∞ (§ 1)

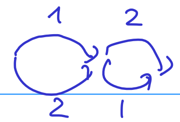
unbeschränkt

Daher Idee (Jahn, M., Schulz, Stier 05)

Lasse nur Wege zu, deren Fahrzeit bzgl. UE-Fahrzeiten nur wenig über $L_k(UE)$ liegen,

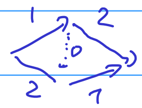
d.h. $\mathcal{P}_k^\varepsilon := \{ P \in \mathcal{P}_k \mid \sum_{a \in P} \tau_a(UE_a) \le (1+\varepsilon) L_k(UE) \}$

implizite Def. der zugelassenen Wege



wollen nur Wege mit Länge ≤ 3

diese Wegemenge nicht als Wege in einem s,t-gewicht. Graphen modellierbar



$\leftarrow$ ein Versuch $\Rightarrow$ nicht auf derselben Kantenmenge realisierbar

$\Downarrow$

Herleitung des Frank-Wolfe Algo wie bisher, nur mit Mengen $\mathcal{P}_k^{\varepsilon}$ statt $\mathcal{P}_k$

hängt nicht vom aktuellen Fluss f ab, sondern nur von UE

d.h. Nebenbedingungen sind

$$\sum_{P \in \mathcal{P}_k^{\varepsilon}} f_P = d_k \quad \forall k$$

$$f_P \geq 0 \quad \forall P \in \bigcup_k \mathcal{P}_k^{\varepsilon}$$

$\Downarrow$

Berechne pro Commodity k einen kürzesten $\overbrace{\text{Weg}}^{P}$ bzgl. $c_a'(x_a^{\ell})$ aus $\mathcal{P}_k^{\varepsilon}$

implizit def

d.h. P muss $\sum_{a \in P} \tau_a(UE_a) \leq (1+\varepsilon) L_k(UE)$

$\Downarrow$

Constrained Shortest Path Problem (CSP)

Geg. Digraph $G = (V, A)$ $\quad s,t \in V$

2 Kantenbewertungen $\quad \tau_a \mathrel{\hat{=}}$ Zeit

$\ell_a \mathrel{\hat{=}}$ Länge $\quad$ Schranke L

Aufgabe: Berechne kürzesten s,t-Weg P bzgl τ_a mit $\ell(P) = \sum_{a \in P} \ell(a) \leq L$

leider Komplexitätssprung: CSP ist (schwach) NP-schwer

genaues Studium in §4