Lecture 13

Feasible direction methods

Emil Gustavsson
Fraunhofer-Chalmers Centre
December 6, 2017

CHALMERS | GÖTEBORGS UNIVERSITET

- Consider the problem to find

$$f^* = \operatorname{infimum} f(x), \tag{1a}$$
$$\text{subject to } x \in X, \tag{1b}$$

X $\subseteq \mathbb{R}^n$ nonempty, **closed & convex**; $f : \mathbb{R}^n \to \mathbb{R}$ is C^1 on X

- Solution idea: generalize unconstrained optimization methods

Step 0. Determine a *starting point* $x_0 \in X$. Set $k := 0$

Step 1. Find a feasible descent *search direction* $p_k \in \mathbb{R}^n$, such that there exists $\bar{\alpha} > 0$ satisfying

- $x_k + \alpha p_k \in X$, $\forall \alpha \in (0, \bar{\alpha}]$
- $f(x_k + \alpha p_k) < f(x_k)$, $\forall \alpha \in (0, \bar{\alpha}]$

Step 2. Determine a *step length* $\alpha_k > 0$ such that $f(x_k + \alpha_k p_k) < f(x_k)$ and $x_k + \alpha_k p_k \in X$

Step 3. Let $x_{k+1} := x_k + \alpha_k p_k$

Step 4. If a *termination criterion* is fulfilled, then stop! Otherwise, let $k := k + 1$ and go to Step 1

Notes

- Just as **local** as methods for unconstrained optimization
- Search direction often of the form $p_k = y_k - x_k$, where $y_k \in X$ solves an (easy) approximate problem
- Line searches analogous to unconstrained case
- Termination criteria and descent based on first-order optimality and/or fixed-point theory ($p_k \approx 0^n$)

- For general X, finding feasible descent direction and step length is difficult (e.g., systems of nonlinear equations)
- X polyhedral $\implies$ search directions and step length easy to find
- X polyhedral $\implies$ local mininma are KKT points
- Methods (to be discussed) will find KKT points

- Frank–Wolfe method based on first-order approximation of f at x_k:

- First-order (necessary) optimality conditions:

$$x^* \text{ local minimum of } f \text{ on } X \implies \nabla f(x^*)^T(x - x^*) \geq 0, \qquad x \in X$$

$$x^* \text{ local minimum of } f \text{ on } X \implies \underset{x \in X}{\text{minimize}} \quad \nabla f(x^*)^T(x - x^*) = 0$$

- Satisfying necessary conditions $\not\Rightarrow$ x^* local minimum

- Violate necessary conditions $\Rightarrow$ can construct feasible descent dir.

- At iterate $x_k \in X$, if

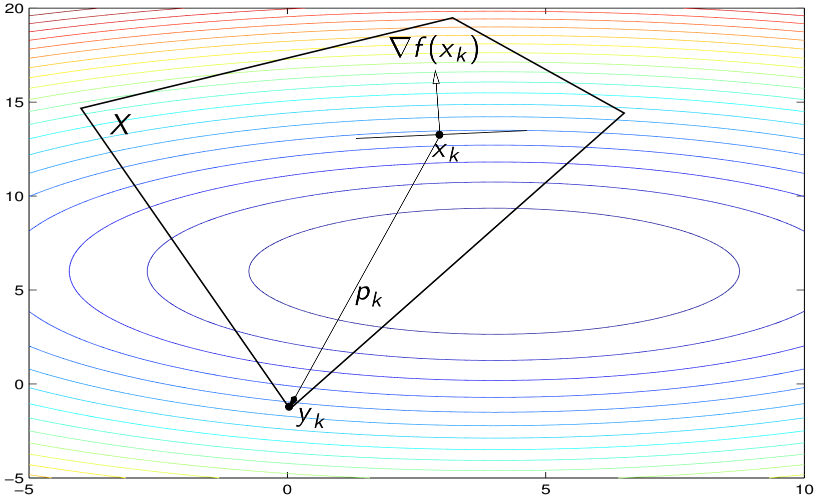
$$\begin{cases} \underset{y \in X}{\text{minimize}} \;\; \nabla f(x_k)^T (y - x_k) < 0, \\ \\ y_k \in \underset{y \in X}{\arg\min} \;\; \nabla f(x_k)^T (y - x_k) \end{cases}$$

  Then,

$$p_k := y_k - x_k \text{ is a feasible descent direction}$$

- Solve **LP** to find y_k (and p_k), since X polyhedral
- Search direction towards an extreme point of X
- This is the basis of the **Frank–Wolfe algorithm**

- If LP has finite optimum $y_k \implies$ search direction $p_k = y_k - x_k$
- If LP obj. val. unbounded, simplex method still finds search dir.
- In this lecture, we assume X bounded for simplicity


20
15
10
5
0
−5
−5
0
5
10
X
∇f(x_k)
x_k
p_k
y_k

Step 0. Find $x_0 \in X$ (e.g. any extreme point in X). Set $k := 0$

Step 1. Find an optimal solution y_k to the problem to

$$\underset{y \in X}{\text{minimize}} \quad z_k(y) := \nabla f(x_k)^T (y - x_k) \tag{2}$$

Let $p_k := y_k - x_k$ be the search direction

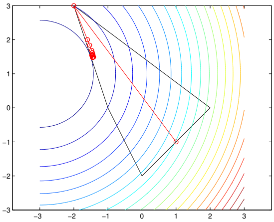
Step 2. Line search: (approximately) minimize $f(x_k + \alpha p_k)$ over $\alpha \in [0, 1]$. Let α_k be the step length

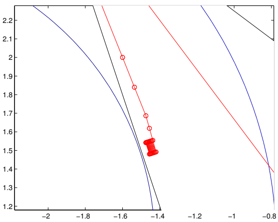
Step 3. Let $x_{k+1} := x_k + \alpha_k p_k$

Step 4. If, for example, $z_k(y_k)$ or α_k is close to zero, then terminate! Otherwise, let $k := k + 1$ and go to Step 1

- *Suppose $X \subset \mathbb{R}^n$ nonempty* **polytope**; *f in C^1 on X*
- *In Step 2 (line search), we either use an* **exact line search** *or the* **Armijo step length rule**
- *Then: the sequence $\{x_k\}$ is bounded and every limit point (at least one exists) is stationary;*
- *If f is convex on X, then every limit point is globally optimal*

Franke-Wolfe convergence





- Suppose f is convex on X. Then for each k, $\forall y \in X$ it holds that

$$\begin{aligned} f(y) &\geq f(x_k) + \nabla f(x_k)^T (y - x_k) && \text{(since } f \text{ convex)} \\ &\geq f(x_k) + \nabla f(x_k)^T (y_k - x_k) && \text{(by definition of } y_k\text{)} \end{aligned}$$

implying that

$$f^* \geq \underbrace{f(x_k) + \nabla f(x_k)^T (y_k - x_k)}_{\text{lower bound of } f^*}$$

- Keep the best lower bound (LBD) up to current iteration. That is,

$$\text{LBD} \leftarrow \max \left\{ \text{LBD}, \;\; f(x_k) + \nabla f(x_k)^T (y_k - x_k) \right\}$$

In step 4, terminate if $f(x_k) - LBD$ is small enough

- Frank–Wolfe uses linear approximations—works **best for almost linear problems**

- For highly nonlinear problems, the approximation is bad—the optimal solution may be far from an extreme point

- In order to find a near-optimum requires many iterations—the algorithm is **slow**

- Extreme points in previous iterations forgotten; can speed up by storing and using previous extreme points

- Representation Theorem (for polytopes):
  - $P = \{\, x \in \mathbb{R}^n \mid Ax = b;\ x \geq 0^n \}$, nonempty and bounded
  - $V = \{v^1, \dots, v^K\}$ be the set of extreme points of P

  Then,

$$x \in P \quad \Longleftrightarrow \quad x = \sum_{i=1}^{K} \alpha_i v^i, \quad \text{for some } \alpha_1, \dots, \alpha_k \geq 0, \ \sum_{i=1}^{K} \alpha_i = 1$$

- **Simplicial decomposition** idea: use some (hopefully few) extreme points to describe optimal solution x^*

$$x^* = \sum_{i \in \mathcal{K}} \alpha_i v^i, \quad |\mathcal{K}| \ll K$$

- Extreme points of feasible set $v^1, \dots, v^K$
- At each iteration k, maintain "working set" $\mathcal{P}_k \subseteq \{v^1, v^2, \dots, v^K\}$
- Check for stationarity of $x_k \in \mathcal{P}_k$ (just like Frank-Wolfe)
  - x_k stationary $\implies$ terminate
  - else, identify (possibly new) extreme pt. y_k; $\mathcal{P}_{k+1} = \mathcal{P}_k \cup \{y_k\}$
- Optimize f over $\text{conv}(\mathcal{P}_{k+1} \cup \{x_k\})$ for x_{k+1}
  – restricted master problem, multi-dimensional line search, etc

Step 0. Find $x_0 \in X$, for example any extreme point in X. Set $k := 0$. Let $\mathcal{P}_0 := \emptyset$

Step 1. Let y_k be an optimal solution to the LP problem

$$\underset{y \in X}{\text{minimize}} \quad z_k(y) := \nabla f(x_k)^T (y - x_k)$$

Let $\mathcal{P}_{k+1} := \mathcal{P}_k \cup \{y_k\}$

Step 2. Min f over $\mathrm{conv}(\{x_k\} \cup \mathcal{P}_{k+1})$. Let $(\mu_{k+1}, \nu_{k+1}) \in \mathbb{R} \times \mathbb{R}^{|\mathcal{P}_{k+1}|}$ minimizes *restricted master problem* (RMP)

$$\begin{aligned}
\underset{(\mu,\nu)}{\text{minimize}} \quad & f\left(\mu x_k + \textstyle\sum_{y_i \in \mathcal{P}_{k+1}} \nu(i) y_i\right) \\
\text{subject to} \quad & \mu + \sum_{i=1}^{|\mathcal{P}_{k+1}|} \nu(i) = 1, \\
& \mu, \nu(i) \geq 0, \qquad i = 1, 2, \ldots, |\mathcal{P}_{k+1}|
\end{aligned}$$

Step 3. Let $x_{k+1} := \mu_{k+1} x_k + \sum_{i=1}^{|\mathcal{P}_{k+1}|} \nu_{k+1}(i) y_i$

Step 4. If $z_k(y_k) \approx 0$ or if $\mathcal{P}_{k+1} = \mathcal{P}_k$ then terminate (why?) Otherwise, let $k := k + 1$ and go to Step 1

- Basic version keeps adding extreme points: $\mathcal{P}_{k+1} \leftarrow \mathcal{P}_k \cup \{y_k\}$
- Alternative: drop members of $\mathcal{P}_k$ with small weights in RMP; or set upper bound on $|\mathcal{P}_k|$
- Special case: $|\mathcal{P}_k| = 1 \implies$ Frank–Wolfe (FW) algorithm!
- Simplicial decomposition (SD) requires fewer iterations than FW
- Unfortunately, solving RMP is more difficult than line search
  - but RMP feasible set structured – unit simplex

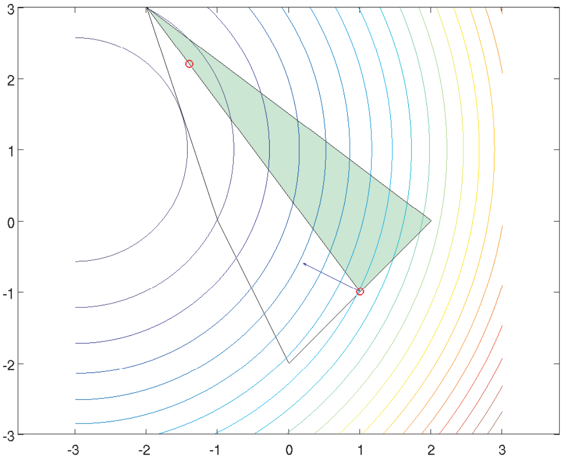


Figure: Example implementation of SD. Starting at $x_0 = (1, -1)^T$, and with $\mathcal{P}_0$ as the extreme point at $(2, 0)^T$, $|\mathcal{P}_k| \leq 2$.

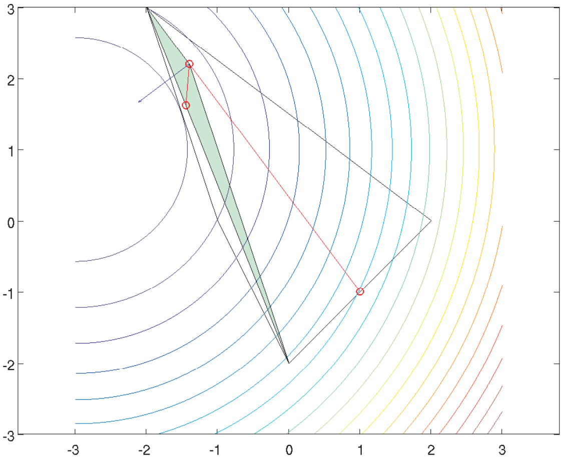


Figure: Example implementation of SD. Starting at $x_0 = (1, -1)^T$, and with $\mathcal{P}_0$ as the extreme point at $(2, 0)^T$, $|\mathcal{P}_k| \leq 2$.

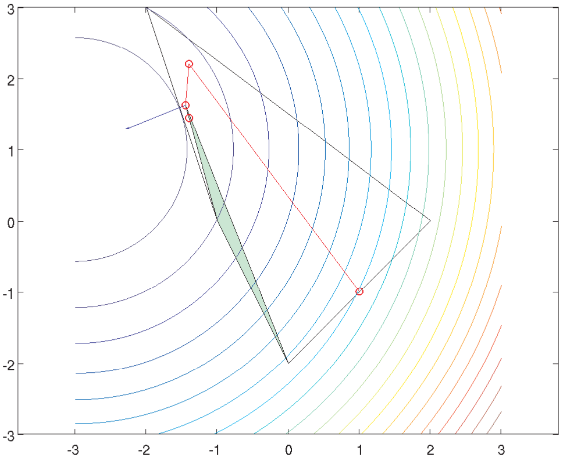


Figure: Example implementation of SD. Starting at $x_0 = (1, -1)^T$, and with $\mathcal{P}_0$ as the extreme point at $(2, 0)^T$, $|\mathcal{P}_k| \leq 2$.

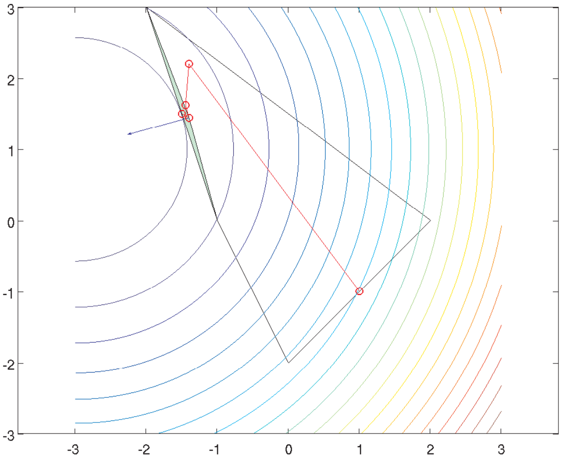


Figure: Example implementation of SD. Starting at $x_0 = (1, -1)^T$, and with $\mathcal{P}_0$ as the extreme point at $(2, 0)^T$, $|\mathcal{P}_k| \leq 2$.

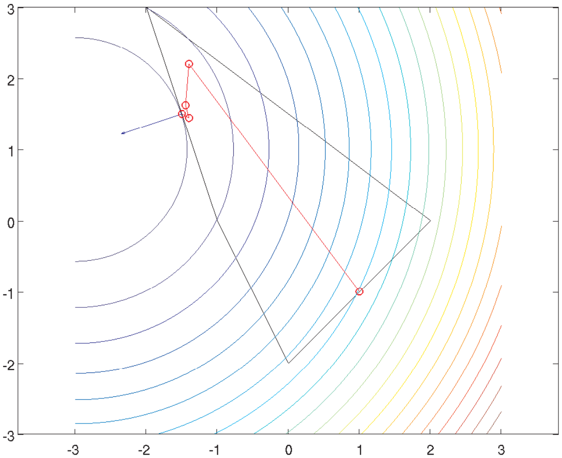
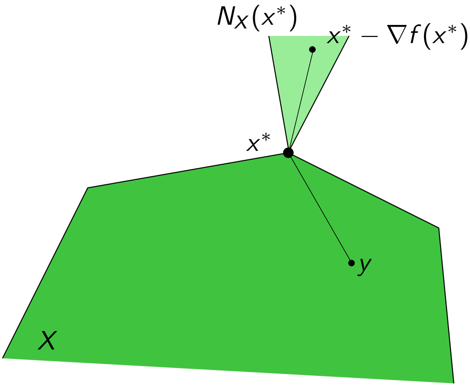


Figure: Example implementation of SD. Starting at $x_0 = (1, -1)^T$, and with $\mathcal{P}_0$ as the extreme point at $(2, 0)^T$, $|\mathcal{P}_k| \leq 2$.

- It does at least as well as the Frank–Wolfe algorithm: line segment $[x_k, y_k]$ feasible in RMP
- SD converges in finite number of iterations if all of following hold
  - x^* unique
  - RMP solved exactly
  - $|\mathcal{P}_k|$ large enough (to represent x^*)
- Much more efficient than the Frank–Wolfe algorithm in practice (consider example solved by FW and SD)
- Can solve the RMPs efficiently, since the constraints are simple

- The gradient projection algorithm based on:

$$x^* \in X \text{ stationary} \iff x^* = \mathrm{Proj}_X[x^* - \alpha \nabla f(x^*)], \quad \forall \alpha > 0$$



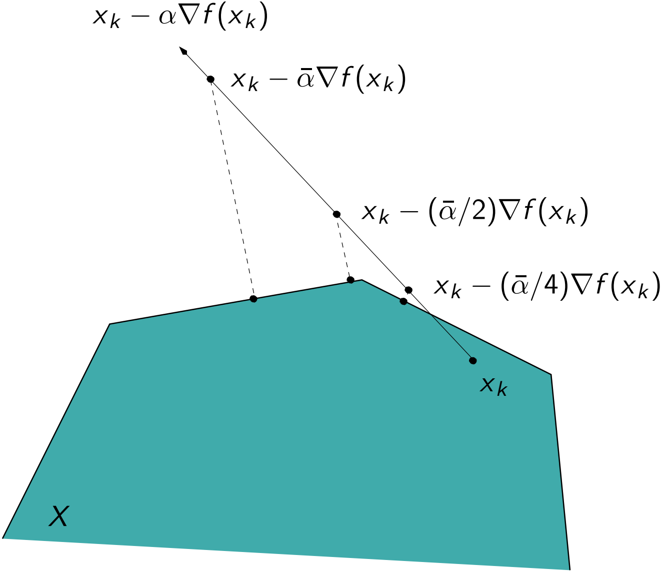
- x not stationary; $p = \mathrm{Proj}_X[x - \alpha \nabla f(x)] - x \neq 0$ for any $\alpha > 0$
  - p feasible descent direction
  - A version of gradient projection method: $x_{k+1} = x_k + \alpha_k p$

- Another version: gradient projection method with **projection arc**:

$$x_{k+1} := \mathrm{Proj}_X[x_k - \alpha_k \nabla f(x_k)]$$

  step size α_k determined using Armijo rule

- $X = \mathbb{R}^n \implies$ gradient projection becomes steepest descent


$x_k - \alpha \nabla f(x_k)$
$x_k - \bar{\alpha} \nabla f(x_k)$
$x_k - (\bar{\alpha}/2) \nabla f(x_k)$
$x_k - (\bar{\alpha}/4) \nabla f(x_k)$
x_k
X

- Bottleneck: how can we compute projections?
- In general, we study the KKT conditions of the system and apply a simplex-like method.
- If we have a specially structured feasible polyhedron, projections may be easier to compute.
  - hypercube $\{x \mid 0 \leq x_i \leq 1, i = 1, \dots, n\}$
  - unit simplex $\{x \mid \sum_{i=1}^{n} x_i = 1,\ x \geq \mathbf{0}\}$ (cf. RMP in simplicial decomposition)

- Example: the feasible set is $S = \{x \in \mathbb{R}^n \mid 0 \leq x_i \leq 1, i = 1, \ldots, n\}$.
- Then $\mathrm{Proj}_S(x) = z$, where

$$z_i = \begin{cases} 0, & x_i < 0, \\ x_i, & 0 \leq x_i \leq 1 \\ 1, & 1 < x_i, \end{cases}$$

  for $i = 1, \ldots, n$.
- Exercise: prove this by applying the variational inequality (or KKT conditions) to the problem

$$\mathrm{min}_{z \in S} \frac{1}{2} \|x - z\|^2$$

- *$X \subseteq \mathbb{R}^n$ nonempty, closed, convex; $f \in C^1$ on X;*
- *for the starting point $x_0 \in X$ it holds that the level set $\mathrm{lev}_f\,(f(x_0))$ intersected with X is bounded*
- *step length α_k is given by the Armijo step length rule along the projection arc*
- *Then: the sequence $\{x_k\}$ is bounded;*
- *every limit point of $\{x_k\}$ is stationary;*
- *$\{f(x_k)\}$ descending, lower bounded, hence convergent*
- Convergence arguments similar to steepest descent one

- *Assume:* $X \subseteq \mathbb{R}^n$ *nonempty, closed, convex;*
- $f \in C^1$ *on* X*;* f *convex;*
- *an optimal solution* x^* *exists*
- *In the algorithm* (4)*, the step length* α_k *is given by the Armijo step length rule along the projection arc*
- *Then: the sequence* $\{x_k\}$ *converges to an optimal solution*
- Note: with $X = \mathbb{R}^n \Longrightarrow$ convergence of steepest descent for convex problems with optimal solutions!

- A large-scale nonlinear network flow problem which is used to estimate traffic flows in cities
- Model over the small city of Sioux Falls in North Dakota, USA; 24 nodes, 76 links, and 528 pairs of origin and destination
- Three algorithms for the RMPs were tested—a Newton method and two gradient projection methods. MATLAB implementation.
- Remarkable difference—The Frank–Wolfe method suffers from very small steps being taken. Why? Many extreme points active = many routes used

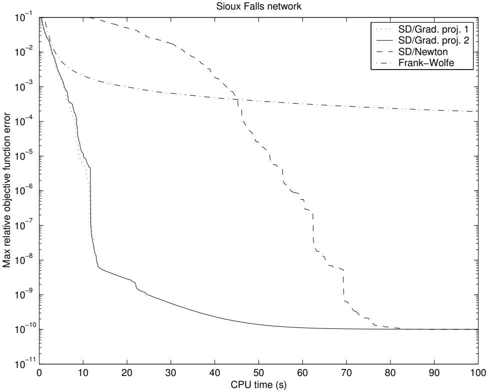


Figure: The performance of SD vs. FW on the Sioux Falls network