Lagrangean Relaxation

Want to solve:

$$\text{(ILP)}\quad \begin{aligned} \max\ & c^T x \\ \text{s.t. } & Ax \le b \quad \longleftarrow \text{"difficult ineq." (problem much easier without them)} \\ & Dx \le d \\ & x \text{ integer} \end{aligned}$$

Instead, we solve for some $\lambda \ge 0$:

$$(LR_\lambda)\quad \begin{aligned} \max\ & c^T x + \lambda^T(b - Ax) \\ \text{s.t. } & Dx \le d \\ & x \text{ integer} \end{aligned}$$

- Instead of explicitly enforcing constraints, we modify the objective such that infeasible solutions get penalized.
- (LR_λ) is called Lagrangean Relaxation of (ILP). The coefficients λ_i are called Lagrangean multipliers.
  → If $Ax = b$, then λ_i unconstrained; if $Ax \le b$, then $\lambda_i \ge 0$.
- Technique very useful in non-lin. programming, here ILP.

Thm. [relaxation]
Let $\lambda \ge 0$, and z^*, $z(\lambda)$ opt values for (ILP) and (LR_λ), respect.
(i) feasible region of (ILP) $\subseteq$ feasible region of (LR_λ)
(ii) $z(\lambda) \ge z^*$.

Pf. (i) Obvious, since we only remove constraints.
(ii) Let x^* be opt. solution to (ILP).
$\Rightarrow\ Ax^* \le b$.

$$z(\lambda) \underset{(i)}{\ge} c^T x^* + \lambda^T \underbrace{(b - Ax^*)}_{\ge 0} \ge c^T x^* = z^*. \qquad \square$$

$\Rightarrow (LR_\lambda)$ gives upper bound on z^* for any $\lambda \ge 0$.
⇝ We want best upper bound $\hat{=}$ $\min\,(z(\lambda) - z^*)$.

$$z_{LD} = \min \{ z(\lambda) \mid \lambda \ge 0 \}$$

(Lagrangean Dual)

How good can z_{LD} be?

$\rightsquigarrow$ sometimes best possible:

Thm. Let $\lambda \geq 0$, and
(i) x_λ be opt. solution of (LR_λ)
(ii) $Ax_\lambda \leq b$, and
(iii) $\lambda^T(b - Ax_\lambda) = 0$.
Then x_λ is optimal solution for (ILP).

Pf. Let z^* be optimal value to (ILP), then $z^* \geq c^T x_\lambda$.
(x_λ is also feasible for (ILP) by (ii).)

$$z^* \geq c^T x_\lambda \overset{\text{(iii)}}{=} c^T x_\lambda + \lambda^T(b - Ax_\lambda) = z(\lambda) \geq z_{LD} \geq z^*.$$ $\square$

$\rightsquigarrow$ Relation between LP relax. and Lagrange relax.:

Thm. $z_{LD} \leq z_{LP}$
Equality holds if $\{x \mid Dx \leq d\}$ is integral polyhedron, that is, the integrality constraint in (LR_λ) can be removed.

Pf.
$$z_{LD} = \min_{\lambda \geq 0} z(\lambda) = \min_{\lambda \geq 0} \max_{\substack{x \text{ integer} \\ Dx \leq d}} Z(\lambda, x) = \min_{\lambda \geq 0} \max_{\substack{Dx \leq d \\ x}} Z(\lambda, x)$$

$\{x \mid Dx \leq d\}$ integral polyhedron $\Rightarrow$ "="; othw. "$\leq$".

$$= \min_{\lambda \geq 0} \max_{\substack{x \\ Dx \leq d}} \left(c^T x + \lambda^T(b - Ax)\right)$$

$$= \min_{\lambda \geq 0} \Big[\lambda^T b + \underbrace{\max_{\substack{x \\ Dx \leq d}} (c^T - \lambda^T A)x}\Big] = \min_{\lambda \geq 0} \Big[\lambda^T b + \underbrace{\min_{\substack{y \geq 0 \\ y^T D = c - \lambda^T A}} y^T d}\Big]$$

LP duality

$$= \min_{\substack{\lambda \geq 0 \\ y \geq 0 \\ y^T D = c - \lambda^T A}} \left[\lambda^T b + y^T d\right] = \max_{\substack{Ax \leq b \\ Dx \leq d}} c^T x = z_{LP}.$$ $\square$

$$\boxed{z^* \leq z_{LD} \leq z_{LP}}$$

Questions:
- How to compute $z(\lambda)$ for some λ?
- How to compute "best" $z(\lambda) = z_{LD}$?

How to compute $z(\lambda)$? ⟶ Example:

TSP – The Held-Karp-Bound

$$\min \sum_{e\in E} c_e x_e$$
$$\text{s.t.} \quad \boxed{\sum_{e\in\delta(v)} x_e = 2, \quad \forall v\in V}$$
$$\sum_{e\in E(S)} x_e \le |S|-1, \quad \forall S\subseteq V,\ |S|\ge 2,\ |S|\le |V|-1$$
$$x_e \in \{0,1\}, \quad \forall e\in E$$

$$\min \sum_{e\in E} c_e x_e$$
$$\text{s.t.} \quad \sum_{e\in E} x_e = n \qquad (1)$$
$$\sum_{e\in\delta(v)} x_e = 2, \quad v=2,\dots,n \qquad (2)$$
$$\sum_{e\in\delta(1)} x_e = 2 \qquad (3)$$
$$\sum_{e\in E(S)} x_e \le |S|-1, \quad \forall S\subseteq V,\ \underset{|S|\le|V|-1}{|S|\ge 2},\ 1\notin S \qquad (4)$$
$$x_e\in\{0,1\}, \quad \forall e\in E \qquad (5)$$

(1) ← added redundant equality

$1\notin S$ ↑ because of redundancy

For λ arbitrary:

$$(LR_\lambda) \left[\begin{array}{l} \min \sum_{e\in E} c_e x_e + \sum_{v=2,\dots,n} \lambda_v \Big(2 - \sum_{e\in\delta(v)} x_e\Big) \\ \text{s.t. } (1),\ (3),\ (4),\ (5) \end{array}\right.$$

▷ Combinatorial structure of feasible solutions for (LR_λ):

__Lemma__: x is feas. solution for (LR_λ) $\Longleftrightarrow$ x is a __1-tree__.

__Def.__ A __1-tree__ of G is a subgraph T of G s.t.

(i) The degree of vertex 1 in T is 2.

(ii) The subgraph of T induced by $\{2, 3, \dots, n\}$ is a tree.

__Pf__: "$\Rightarrow$" (1), (3), (5) x has degree 2 for vertex 1 and $\leq n-2$ edges on $n-1$ nodes

(4) connected

$\Rightarrow$ tree on $\{2, \dots, n\}$.

"$\Leftarrow$" Any 1-tree satisfies constraints. $\square$

▷ Reformulate cost function $L(\lambda, x)$ for easier combinatorial interpretation:

$$Z(\lambda, x) = \sum_{e \in E} c_e x_e + \sum_{v=2,\dots,n} \lambda_v \underbrace{\left(\sum_{e \in \delta(v)} x_e - 2 \right)}_{= \text{"divergence from node degree 2"}}$$

// replaced λ_i by $-\lambda_i$; no restriction

Let $\lambda_1 := 0$.

$$Z(\lambda, x) = \sum_{e \in E} c_e x_e + \sum_{v \in V} \lambda_v \sum_{e \in \delta(v)} x_e - 2 \sum_{\substack{\cancel{v=2,\dots,n} \\ v \in V}} \lambda_v$$

$$= \sum_{e \in E} c_e x_e + \sum_{e=(v,u)} (\lambda_v + \lambda_u) x_e - 2 \sum_{v \in V} \lambda_v$$

$$= \sum_{(v,u)=e \in E} (c_e + \lambda_v + \lambda_u) x_e - 2 \sum_{v \in V} \lambda_v .$$

▷ Interpretation of Lagrange relaxation = __Held-Karp-Bound__:

- For given λ: (LR_λ) corresponds to finding a 1-tree of min cost w.r.t. edge cost:

$$c'_e = c_e + \lambda_v + \lambda_u \quad , \quad \forall e = (v,u) \in E.$$

- 1-tree problem can be solved optimally in polyn. time (combinatorial alg.)
  - (1) Compute MST on $G|_{\{2,\dots,n\}}$ by Kruskal or Prim.
  - (2) Choose 2 cheapest edges $(1,v)$, $v \in V$.
- $\boxed{z_{LD} = z_{LP}}$ (See Thm. above)

Why not "simply" solve LP-relax. to get z_{LD}?
(using Ellipsoid, interior point method)
⇝ in practice too slow!
often preferred <u>subgradient method</u> (see later)

- variation of lagrangean multipliers λ_i = variation of edge cost
  ⇝ strengthening of upper bound (towards z_{LD})

- <u>Algorithm</u>

Let $\lambda_v := 0$, $v = 1,\dots,n$; step width $\omega := 1$.

<u>Repeat</u>:
- Compute min 1-tree for cost fct. $c'_e = c_e + \lambda_v - \lambda_u$ for $e = (v,u)$.
- If solution x is a tour ⇒ return x.
- Othw. let $\lambda_v := \lambda_v + (d_v - 2)\,\omega$ with d_v = degree of node v.

  [if $d_v > 2$ ⇒ increase λ_v ⇒ $c_v^* \nearrow$
  if $d_v < 2$ ⇒ decrease λ_v ⇒ $c_v^* \searrow$
  $d_1 = 2$ ⇒ $\lambda_1 = 0$]

(• maybe adjust step width ω)

<u>Until</u>: $z(\lambda)$ is "good enough".

Is that opt? Why?
→ "=" in dualized con (see Thm.)

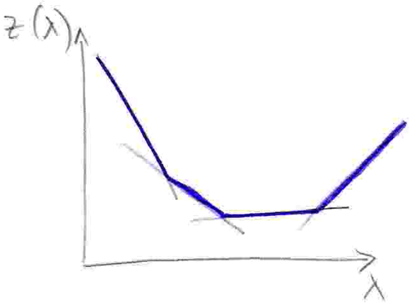
Return best x and $z(\lambda)$ found so far.

How to compute $\min\limits_{\lambda \geq 0} z(\lambda) = z_{LD}$?

Assume feas. region F of LR_λ contains finite set of points $x^1, x^2, \dots, x^k$.

Then $z_{LD} = \min\limits_{\lambda \geq 0} z(\lambda) = \min\limits_{\lambda \geq 0} \{ \max\limits_{x \in F} [c^T x + \lambda^T (b - Ax)] \}$

$$= \min_{\lambda \geq 0} \{ \underbrace{\max_{i=1,\dots,k} [c^T x^i + \lambda^T (b - Ax^i)]} \}$$



= maximum over finitely many affine functions

= piecewise linear convex function

$\rightarrow$ but not differentiable!

$\Rightarrow$ $\boxed{z_{LD} = \text{min. of a piecewise lin. convex non-diff. function}}$

- For <u>differentiable</u> fct.: use gradient and gradient descent method.
- Here: use <u>subgradient</u> (more general definition)

<u>Def.</u> A subgradient at λ of a convex fct. $f: \mathbb{R}^m \to \mathbb{R}$ is a vector $\gamma(\lambda) \in \mathbb{R}^m$ s.t. $f(v) - f(\lambda) \geq \gamma(\lambda)^T (v - \lambda), \forall v \in \mathbb{R}^m$.

$\rightarrow$ For Lagrange relaxation (opt. sol.), we get subgradient <u>for free</u>:

<u>Lemma</u>: Let x^* be opt. sol. to (LR_λ) for $\lambda = \lambda_0$.
Then $b - Ax^*$ is subgradient of $z_{LD} = \min\limits_{\lambda \geq 0} z(\lambda)$.

<u>Subgradient Method</u>

Set $\lambda := \lambda^0$, $i := 0$, step width ω.

<u>While</u> stopping criterion not fulfilled:

- Find opt. solution $x^*(\lambda^i)$ to (LR_λ) for $\lambda = \lambda^i$.
- Set $\lambda^{i+1} = \max\{\lambda^i + \omega(b - Ax^*(\lambda^i)), 0\}$
- $i := i+1$