Introduction to Linear Programming

Leo Liberti

LIX, École Polytechnique

liberti@lix.polytechnique.fr

Course material:

http://www.lix.polytechnique.fr/˜liberti/teaching/xct/

Contents

- LP formulations and examples
- The simplex method
- Optimality conditions
- Duality

Definitions

- Mathematical programming formulation:

$$\left.\begin{array}{rl} \min_x & f(x) \\ \text{s.t.} & g(x) \leq 0, \end{array}\right\} [P] \qquad (1)$$

- A point x^* is *feasible* in P if $g(x^*) \leq 0$;
$F(P) =$ set of feasible points of P
- A feasible x^* is a *local minimum* if $\exists B(x^*, \varepsilon)$ s.t.
$\forall x \in F(P) \cap B(x^*, \varepsilon)$ we have $f(x^*) \leq f(x)$
- A feasible x^* is a *global minimum* if $\forall x \in F(P)$ we have
$f(x^*) \leq f(x)$
- Thm.: if f and $F(P)$ convex, any local min. is also global
- If $g_i(x^*) = 0$ for some i, g_i is *active at* x^*

Canonical form

- P is a *linear programming problem* (LP) if $f : \mathbb{R}^n \to \mathbb{R}$, $g : \mathbb{R}^n \to \mathbb{R}^m$ are **linear forms**
- LP in *canonical form*:

$$\left.\begin{array}{rl} \min_x & c^{\mathsf{T}} x \\ \text{s.t.} & Ax \le b \\ & x \ge 0 \end{array}\right\} [C] \qquad (2)$$

- Can reformulate inequalities to equations by adding a *slack variable* x_{n+1}:

$$\sum_{j=1}^{n} a_j x_j \le b \;\Rightarrow\; \sum_{j=1}^{n} a_j x_j + x_{n+1} = b \;\wedge\; x_{n+1} \ge 0$$

Standard form

- LP in *standard form*: all inequalities transformed to equations

$$\left.\begin{array}{rl} \min_x & (c')^{\mathsf{T}} x \\ \text{s.t.} & A'x = b \\ & x \geq 0 \end{array}\right\} [S] \tag{3}$$

- where $x = (x_1, \ldots, x_n, x_{n+1}, \ldots, x_{n+m})$, $A' = (A, I_m)$, $c' = (c, \underbrace{0, \ldots, 0}_{m})$
- Standard form useful because linear systems of equations are computationally easier to deal with than systems of inequalities
- Used in simplex algorithm

Diet problem I

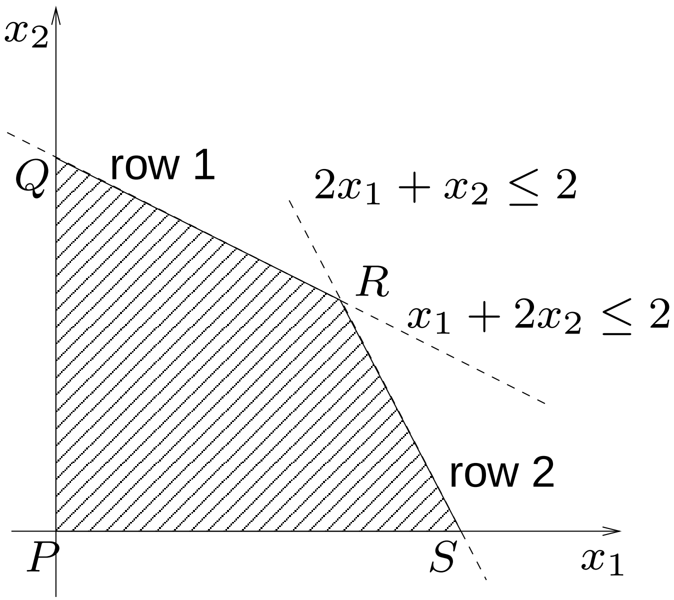
- Consider set M of m nutrients (e.g. sugars, fats, carbohydrates, proteins, vitamins, . . .)
- Consider set N of n types of food (e.g. pasta, steak, potatoes, salad, ham, fruit, . . .)
- A diet is healthy if it has at least b_i units of nutrient $i \in M$
- Food $j \in N$ contains a_{ij} units of nutrient $i \in M$
- A unit of food $j \in N$ costs c_j
- Find a healthy diet of minimum cost

Diet problem II

- Parameters: $m \times n$ matrix $A = (a_{ij})$, $b = (b_1, \ldots, b_m)$, $c = (c_1, \ldots, c_n)$
- Decision variables: $x_j =$ quantity of food j in the diet
- Objective function: $\min\limits_{x} \sum\limits_{j=1}^{n} c_j x_j$
- Constraints: $\forall i \in M \ \sum\limits_{j=1}^{n} a_{ij} x_j \geq b_i$
- Limits on variables: $\forall j \in N \ x_j \geq 0$
- Canonical form: $\min\{c^\mathsf{T} x \mid -Ax \leq -b\}$
- Standard form: add slack variables $y_i =$ surplus quantity of i-th nutrient, get $\min\{c^\mathsf{T} x \mid -Ax + I_m y = -b\}$

Geometry of LP

- A *polyhedron* is the intersection of a finite number of closed halfspaces. A bounded, non-empty polyhedron is a *polytope*



Canonical feas. polyhedron: $F(C) = \{x \in \mathbb{R}^n \mid Ax \leq b \ \wedge \ x \geq 0\}$

$A = \begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix}$, $b^\mathsf{T} = (2, 2)$

Standard feas. polyhedron: $F(S) = \{(x, y) \in \mathbb{R}^{n+m} \mid Ax + I_m y = b \ \wedge \ (x, y) \geq 0\}$

- $P = (0, 0, 2, 2), Q = (0, 1, 0, 1), R = (\frac{2}{3}, \frac{2}{3}, 0, 0), S = (1, 0, 1, 0)$
- Each vertex corresponds to an intersection of at least n hyperplanes $\Rightarrow \geq n$ coordinates are zero

Basic feasible solutions

- Consider polyhedron in "equation form" $K = \{x \in \mathbb{R}^n \mid Ax = b \wedge x \geq 0\}$. A is $n \times m$ of rank m **(N.B. n here is like $n + m$ in last slide!)**
- A subset of m linearly independent columns of A is a *basis* of A
- If β is the set of column indices of a basis of A, variables x_i are *basic* for $i \in \beta$ and *nonbasic* for $i \notin \beta$
- Partition A in a square $m \times m$ nonsingular matrix B (columns indexed by β) and an $(n-m) \times m$ matrix N
- Write $A = (B|N)$, $x_B \in \mathbb{R}^m$ basics, $x_N \in \mathbb{R}^{n-m}$ nonbasics
- Given a basis $(B|N)$ of A, the vector $x = (x_B, x_N)$ is a *basic feasible solution* (bfs) of K with respect to the given basis if $x_B \geq 0$ and $x_N = 0$

Fundamental Theorem of LP

- Given a polyhedron K in "equation form", any bfs of K is a vertex of K and vice versa
- For any $c \in \mathbb{R}^n$, there is always one bfs that solves the LP $\min\{c^\mathsf{T} x \,|x \in K\}$
- Important correspondence between bfs's and vertices suggests geometric solution method based on exploring vertices of K
- Proofs not difficult but long

Simplex Algorithm: Summary

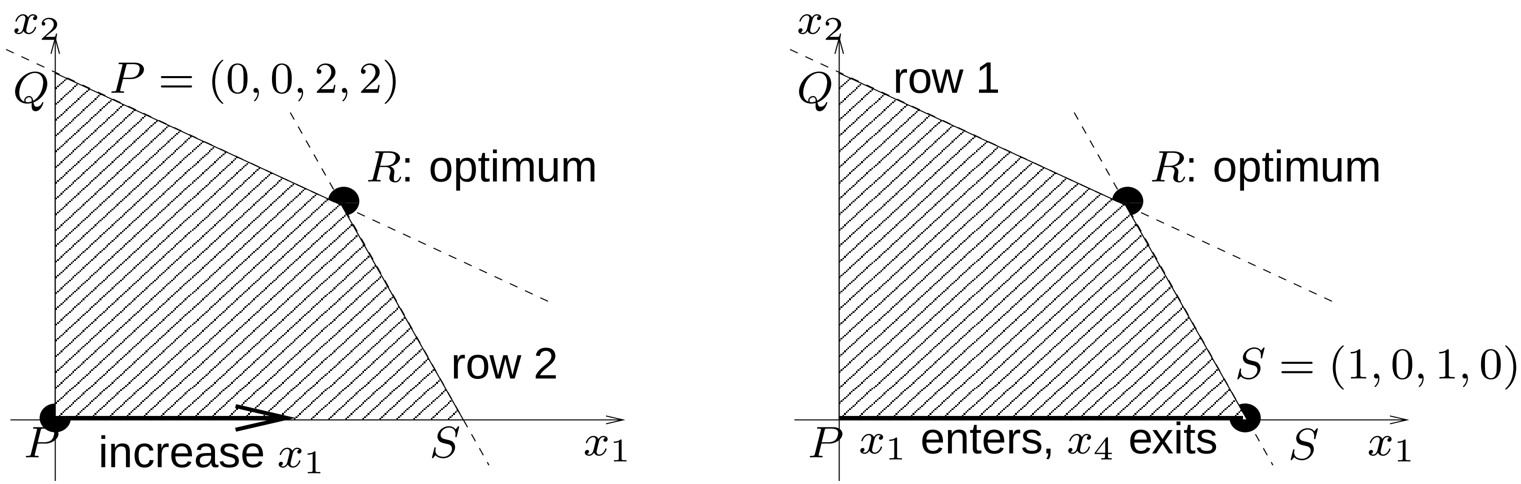
- Solves LPs in form $\min_{x \in K} c^{\mathsf{T}} x$ where $K = \{Ax = b \wedge x \geq 0\}$
- Starts from any vertex x
- Moves to an adjacent improving vertex x'
(i.e. x' is s.t. $\exists$ edge $\{x, x'\}$ in K and $c^{\mathsf{T}} x' \leq c^{\mathsf{T}} x$)
- Two bfs's with basics indexed by β, β' correspond to adjacent vertices if $|\beta \cap \beta'| = m - 1$
- Stops when no such x' exists
- Detects unboundedness and prevents cycling $\Rightarrow$ convergence
- K convex $\Rightarrow$ global optimality follows from local optimality at termination

Simplex Algorithm I

- Let $x = (x_1, \ldots, x_n)$ be the current bfs, write $Ax = b$ as $Bx_B + Nx_N = b$
- Express basics in terms of nonbasics:
  $x_B = B^{-1}b - B^{-1}Nx_N$ (this system is a *dictionary*)
- Express objective function in terms of nonbasics:
  $c^\top x = c_B^\top x_B + c_N^\top x_N = c_B^\top(B^{-1}b - B^{-1}Nx_N) + c_N^\top x_N \Rightarrow$
  $\Rightarrow c^\top x = c_B^\top B^{-1}b + \bar{c}_N^\top x_N$
  ($\bar{c}_N^\top = c_N^\top - c_B^\top B^{-1}N$ are the *reduced costs*)
- Select an improving direction: pick a nonbasic variable x_h with negative reduced cost; increasing its value will decrease the objective function value
- If no such h exists, no improving direction, local minimum $\Rightarrow$ global minimum $\Rightarrow$ termination

Simplex Algorithm II

- Iteration start: x_h is out of basis $\Rightarrow$ its value is zero
- We want to increase its value to strictly positive to decrease objective function value
- ...corresponds to "moving along an edge"
- We stop when we reach another (improving) vertex
- ...corresponds to setting a basic variable x_l to zero



- x_h enters the basis, x_l exits the basis

Simplex Algorithm III

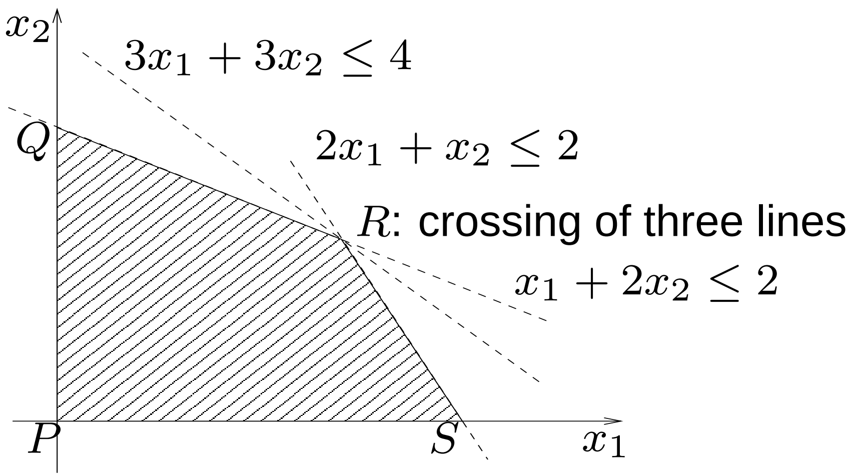
- How do we determine l and new positive value for x_h?
- Recall dictionary $x_B = B^{-1}b - B^{-1}Nx_N$, write $\bar{b} = B^{-1}b$ and $\bar{A} = (\bar{a}_{ij}) = B^{-1}N$
- For $i \in \beta$ (basics), $x_i = \bar{b}_i - \sum_{j \notin \beta} \bar{a}_{ij} x_j$
- Consider nonbasic index h of variable entering basis (all the other nonbasics stay at 0), get $x_i = \bar{b}_i - \bar{a}_{ih} x_h, \forall i \in \beta$
- Increasing x_h may make $x_i < 0$ (infeasible), to prevent this enforce $\forall i \in \beta \ (\bar{b}_i - \bar{a}_{ih} x_h \geq 0)$
- Require $x_h \leq \frac{\bar{b}_i}{\bar{a}_{ih}}$ for $i \in \beta$ and $\bar{a}_{ih} > 0$:

$$l = \text{argmin}\{\frac{\bar{b}_i}{\bar{a}_{ih}} \mid i \in \beta \wedge \bar{a}_{ih} > 0\}, \qquad x_h = \frac{\bar{b}_l}{\bar{a}_{lh}}$$

- If all $\bar{a}_{ih} \leq 0$, x_h can increase without limits: problem unbounded

Simplex Algorithm IV

- Suppose $> n$ hyperplanes cross at vtx R (*degenerate*)
- May get improving direction s.t. adjacent vertex is still R
- Objective function value does not change
- Seq. of improving dirs. may fail to move away from R
- $\Rightarrow$ simplex algorithm cycles indefinitely
- Use Bland's rule: among candidate entering / exiting variables, choose that with *least index*



Example: Formulation

- Consider problem:

$$\left.\begin{array}{rl} \max\limits_{x_1,x_2} & x_1 + x_2 \\ \text{s.t.} & x_1 + 2x_2 \le 2 \\ & 2x_1 + x_2 \le 2 \\ & x \ge 0 \end{array}\right\}$$

- Standard form:

$$\left.\begin{array}{rl} -\min_x & -x_1 - x_2 \\ \text{s.t.} & x_1 + 2x_2 + x_3 = 2 \\ & 2x_1 + x_2 + x_4 = 2 \\ & x \ge 0 \end{array}\right\}$$

- Obj. fun.: $\max f = -\min -f$, simply solve for $\min f$

Example, itn 1: start

- Objective function vector $c^{\mathsf{T}} = (-1, -1, 0, 0)$
- Constraints in matrix form:

$$\begin{pmatrix} 1 & 2 & 1 & 0 \\ 2 & 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 2 \\ 2 \end{pmatrix}$$

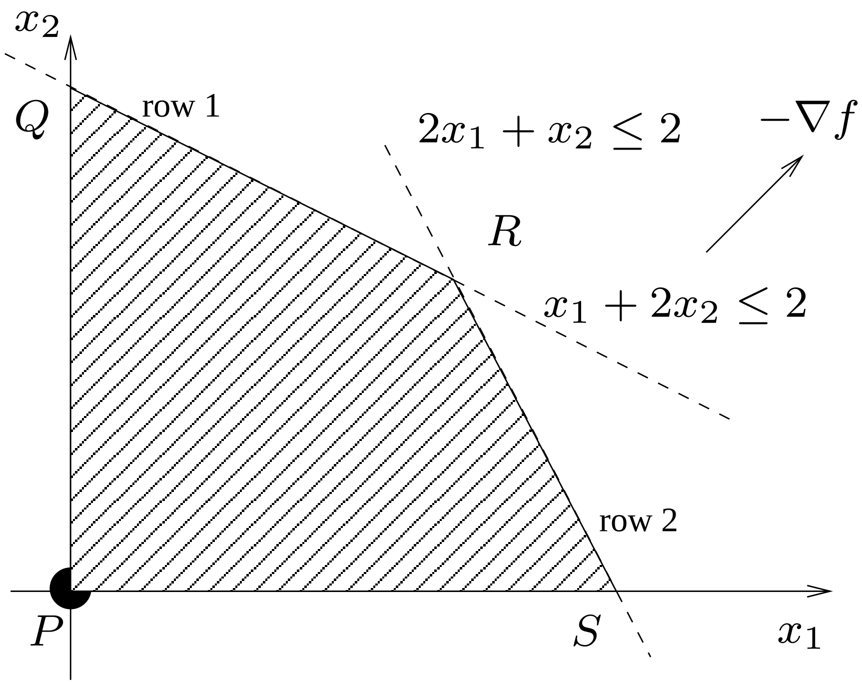
- Choose obvious starting basis with

$$B = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, N = \begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix}, \beta = \{3, 4\}$$

- Corresponds to point $P = (0, 0, 2, 2)$

Example, itn 1: dictionary

- Start the simplex algorithm with basis in P



- Compute dictionary $x_B = B^{-1}b - B^{-1}Nx_N = \bar{b} - \bar{A}x_N$, where

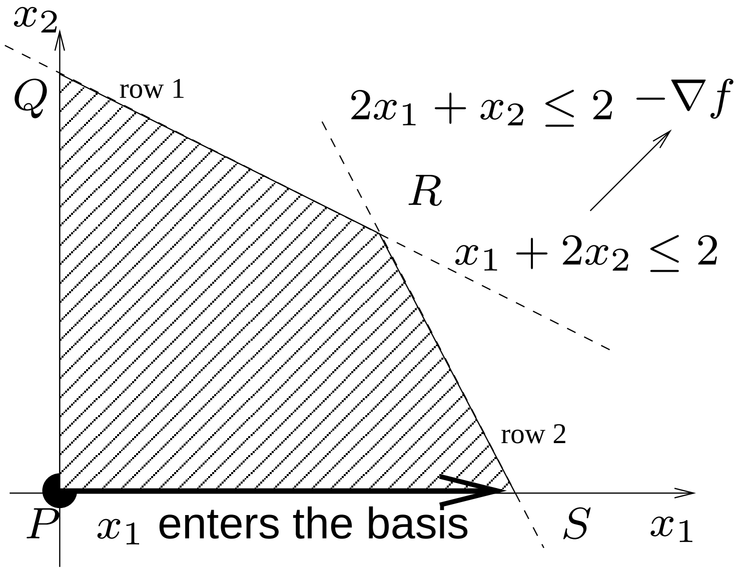
$$\bar{b} = \begin{pmatrix} 2 \\ 2 \end{pmatrix} \quad ; \quad \bar{A} = \begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix}$$

Example, itn 1: entering var

- Compute reduced costs $\bar{c}_N = c_N^\mathsf{T} - c_B^\mathsf{T}\bar{A}$:

$$(\bar{c}_1, \bar{c}_2) = (-1, -1) - (0, 0)\bar{A} = (-1, -1)$$

- All nonbasic variables $\{x_1, x_2\}$ have negative reduced cost, can choose whichever to enter the basis
- Bland's rule: choose entering nonbasic with least index in $\{x_1, x_2\}$, i.e. pick $h = 1$ (move along edge $\overline{PS}$)



Example, itn 1: exiting var

- Select exiting basic index l

$$
\begin{aligned}
l &= \text{argmin}\{\frac{\bar{b}_i}{\bar{a}_{ih}} \mid i \in \beta \wedge \bar{a}_{ih} > 0\} = \text{argmin}\{\frac{\bar{b}_1}{\bar{a}_{11}}, \frac{\bar{b}_2}{\bar{a}_{21}}\} \\
&= \text{argmin}\{\frac{2}{1}, \frac{2}{2}\} = \text{argmin}\{2, 1\} = 2
\end{aligned}
$$

- Means: “select second basic variable to exit the basis”, i.e. x_4
- Select new value $\frac{\bar{b}_l}{\bar{a}_{lh}}$ for x_h (recall $h = 1$ corrresponds to x_1):

$$
\frac{\bar{b}_l}{\bar{a}_{lh}} = \frac{\bar{b}_2}{\bar{a}_{21}} = \frac{2}{2} = 1
$$

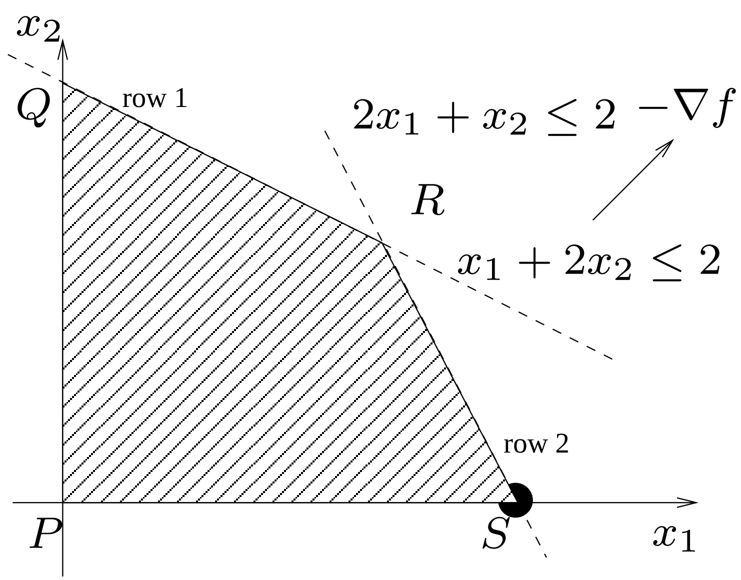
- x_1 enters, x_4 exits (apply swap $(1, 4)$ to β)

Example, itn 2: start

- Start of new iteration: basis is $\beta = \{1, 3\}$

$$B = \begin{pmatrix} 1 & 1 \\ 2 & 0 \end{pmatrix} \quad ; \quad B^{-1} = \begin{pmatrix} 0 & \frac{1}{2} \\ 1 & -\frac{1}{2} \end{pmatrix}$$

- $x_B = (x_1, x_3) = B^{-1}b = (1, 1)$, thus current bfs is $(1, 0, 1, 0) = S$

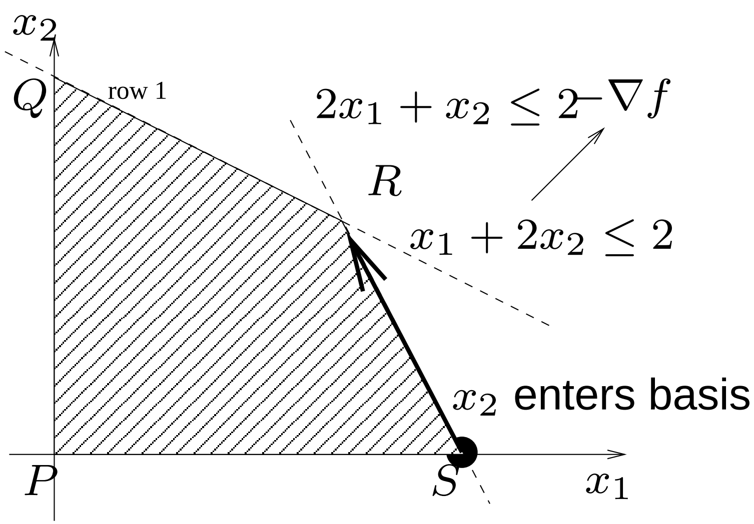


Example, itn 2: entering var

- Compute dictionary: $\bar{b} = B^{-1}b = (1,1)^{\mathsf{T}}$,

$$\bar{A} = B^{-1}N = \begin{pmatrix} 0 & \frac{1}{2} \\ 1 & -\frac{1}{2} \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{3}{2} & -\frac{1}{2} \end{pmatrix}$$

- Compute reduced costs:
  $(\bar{c}_2, \bar{c}_4) = (-1,0) - (-1,0)\bar{A} = (-1/2, 1/2)$
- Pick $h = 1$ (corresponds to x_2 entering the basis)



Example, itn 2: exiting var

- Compute l and new value for x_2:

$$\begin{aligned} l &= \text{argmin}\{\frac{\bar{b}_1}{\bar{a}_{11}}, \frac{\bar{b}_2}{\bar{a}_{21}}\} = \text{argmin}\{\frac{1}{1/2}, \frac{1}{3/2}\} = \\ &= \text{argmin}\{2, 2/3\} = 2 \end{aligned}$$

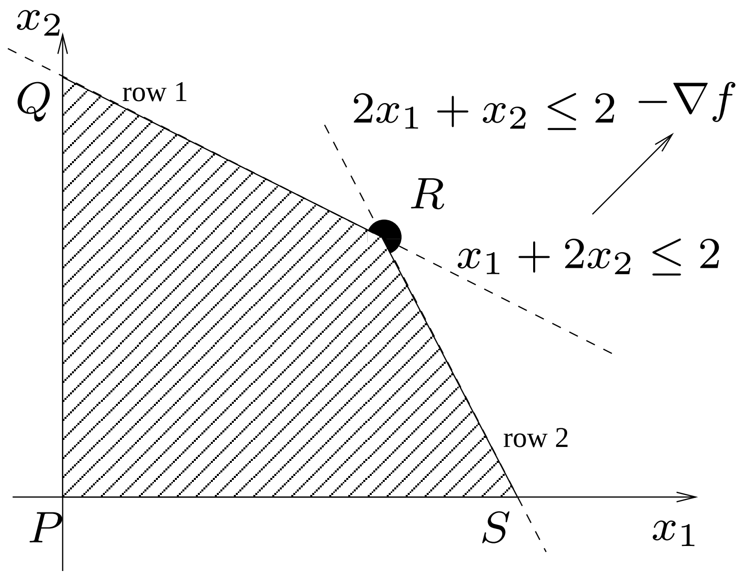
- $l = 2$ corresponds to second basic variable x_3
- New value for x_2 entering basis: $\frac{2}{3}$
- x_2 enters, x_3 exits (apply swap $(2, 3)$ to β)

Example, itn 3: start

- Start of new iteration: basis is $\beta = \{1, 2\}$

$$B = \begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix} \quad ; \quad B^{-1} = \begin{pmatrix} -\frac{1}{3} & \frac{2}{3} \\ \frac{2}{3} & -\frac{1}{3} \end{pmatrix}$$

- $x_B = (x_1, x_2) = B^{-1}b = (\frac{2}{3}, \frac{2}{3})$, thus current bfs is $(\frac{2}{3}, \frac{2}{3}, 0, 0) = R$



Example, itn 3: termination

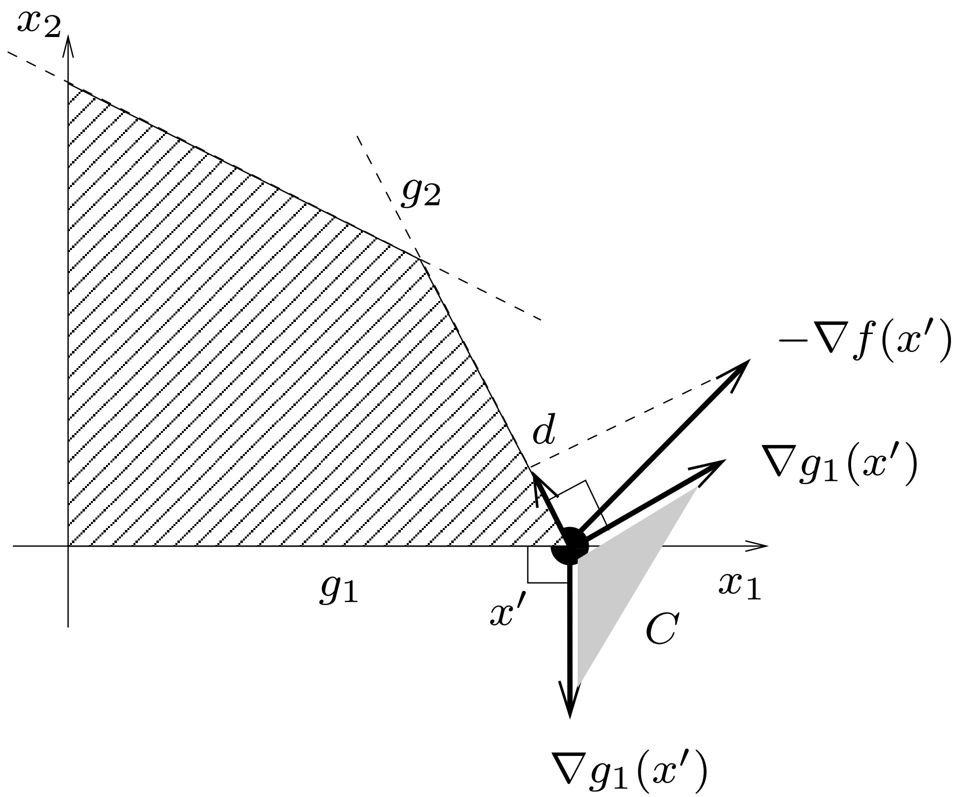
- Compute dictionary: $\bar{b} = B^{-1}b = (2/3, 2/3)^\mathsf{T}$,

$$\bar{A} = B^{-1}N = \begin{pmatrix} -\frac{1}{3} & \frac{2}{3} \\ \frac{2}{3} & -\frac{1}{3} \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} -\frac{1}{3} & \frac{2}{3} \\ \frac{2}{3} & -\frac{1}{3} \end{pmatrix}$$

- Compute reduced costs:
  $(\bar{c}_3, \bar{c}_4) = (0, 0) - (-1, -1)\bar{A} = (1/3, 1/3)$
- No negative reduced cost: algorithm terminates
- Optimal basis: $\{1, 2\}$
- Optimal solution: $R = (\frac{2}{3}, \frac{2}{3})$
- Optimal objective function value $f(R) = -\frac{4}{3}$
- Permutation to apply to initial basis $\{3, 4\}$: $(1, 4)(2, 3)$

Optimality Conditions I

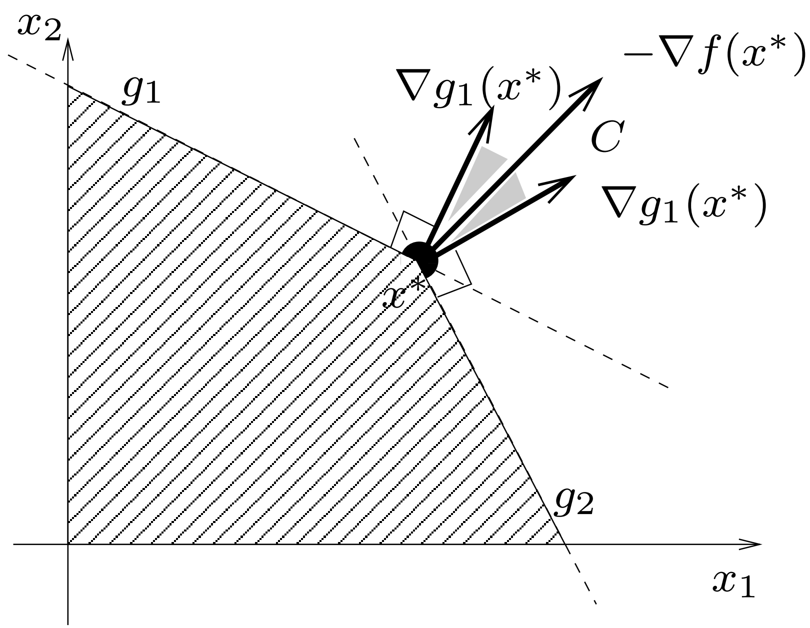
- If we can project improving direction $-\nabla f(x')$ on an active constraint g_2 and obtain a feasible direction d, point x' is not optimal



- Implies $-\nabla f(x') \notin C$ (*cone generated by active constraint gradients*)

Optimality Conditions II

- Geometric intuition: situation as above does not happen when $-\nabla f(x^*) \in C$, x^* optimum



- Projection of $-\nabla f(x^*)$ on active constraints is never a feasible direction

Optimality Conditions III

- If:
  1. x^* is a local minimum of problem $P \equiv \min\{f(x) \mid g(x) \leq 0\}$,
  2. I is the index set of the active constraints at x^*, i.e. $\forall i \in I \; (g_i(x^*) = 0)$
  3. $\nabla g_I(x^*) = \{\nabla g_i(x^*) \mid i \in I\}$ is a linearly independent set of vectors
- then $-\nabla f(x^*)$ is a conic combination of $\nabla g_I(x^*)$, i.e. $\exists \lambda \in \mathbb{R}^{|I|}$ such that

$$\begin{aligned} \nabla f(x^*) + \sum_{i \in I} \lambda_i \nabla g_i(x^*) &= 0 \\ \forall i \in I \; \lambda_i &\geq 0 \end{aligned}$$

Karush-Kuhn-Tucker Conditions

- Define

$$L(x,\lambda) = f(x^*) + \sum_{i=1}^{m} \lambda_i g_i(x^*)$$

as the *Lagrangian* of problem P

- KKT: If x^* is a local minimum of problem P and $\nabla g(x^*)$ is a linearly independent set of vectors, $\exists \lambda \in \mathbb{R}^m$ s.t.

$$\begin{array}{rcl} \nabla_{x^*} L(x,\lambda) & = & 0 \\ \forall i \le m \quad (\lambda_i g_i(x^*) & = & 0) \\ \forall i \le m \quad (\lambda_i & \ge & 0) \end{array}$$

Weak duality

- Let $\bar{L}(\lambda) = \min_{x \in F(P)} L(x, \lambda)$
- Let x^* be the global optimum of P
- Theorem: $\forall \lambda \geq 0 \quad \bar{L}(\lambda) \leq f(x^*)$
- Proof: since $\lambda \geq 0$, if $x \in F(P)$ then $\lambda_i g_i(x) \leq 0$, hence $L(x, \lambda) \leq f(x)$; result follows as we are taking the minimum over all $x \in F(P)$
- Important point: $\bar{L}(\lambda)$ is a lower bound for P for all $\lambda \geq 0$
- The problem of finding the tightest Lagrangian lower bound

$$\max_{\lambda \geq 0} \min_{x \in F(P)} L(x, \lambda)$$

is the *Lagrangian dual* of problem P

Dual of an LP I

- Consider LP P in form: $\min\{c^\mathsf{T}x \mid Ax \geq b \wedge x \geq 0\}$
- $L(x,s,y) = c^\mathsf{T}x - s^\mathsf{T}x + y^\mathsf{T}(b - Ax)$ where $s \in \mathbb{R}^n$, $y \in \mathbb{R}^m$
- Lagrangian dual:

$$\max_{s,y\geq 0} \min_{x\in F(P)} (yb + (c^\mathsf{T} - s - yA)x)$$

- KKT: for a point x to be optimal,

$$\begin{aligned} c^\mathsf{T} - s - yA &= 0 \text{ (KKT1)} \\ \forall j \leq n\ (s_j x_j = 0),\ \forall i \leq m\ (y_i(b_i - A_i x) &= 0) \text{ (KKT2)} \\ s, y &\geq 0 \text{ (KKT3)} \end{aligned}$$

- Consider Lagrangian dual s.t. (KKT1), (KKT3):

Dual of an LP II

- Obtain:

$$\left.\begin{array}{rrcl} \max\limits_{s,y} & yb & & \\ \text{s.t.} & yA+s & = & c^{\mathsf{T}} \\ & s,y & \geq & 0 \end{array}\right\}$$

- Interpret s as slack variables, get *dual of LP*:

$$\left.\begin{array}{rrcl} \max\limits_{y} & yb & & \\ \text{s.t.} & yA & \leq & c^{\mathsf{T}} \\ & y & \geq & 0 \end{array}\right\} [D]$$

Strong Duality

- Assume x optimum, KKT conditions hold
- Recall (KKT2) $\forall j \le n (s_i x_i = 0)$, $\forall i \le m \; (y_i(b_i - A_i x) = 0)$
- Get $y(b - Ax) = sx \Rightarrow yb = (yA + s)x$
- By (KKT1) $yA + s = c^{\mathsf{T}}$
- Obtain $yb = c^{\mathsf{T}} x$
- Theorem: if x is optimum of a convex problem, primal and dual objective functions attain the same values at x

The dual of the Diet Problem

- Recall diet problem: select minimum-cost diet of n foods providing m nutrients
- Suppose firm wishes to set the prices $y \geq 0$ for m nutrient pills
- To be competitive with normal foods, the equivalent in pills of a food $j \leq n$ must cost less than the cost of the food c_j
- Objective: $\max \sum\limits_{i \leq m} b_i y_i$
- Constraints: $\forall j \leq n \ \ \sum\limits_{i \leq m} a_{ij} y_i \leq c_j$
- Economic interpretation:
  at optimum, cost of pills = cost of diet

Example: Dual formulation

- Primal problem P and canonical form:

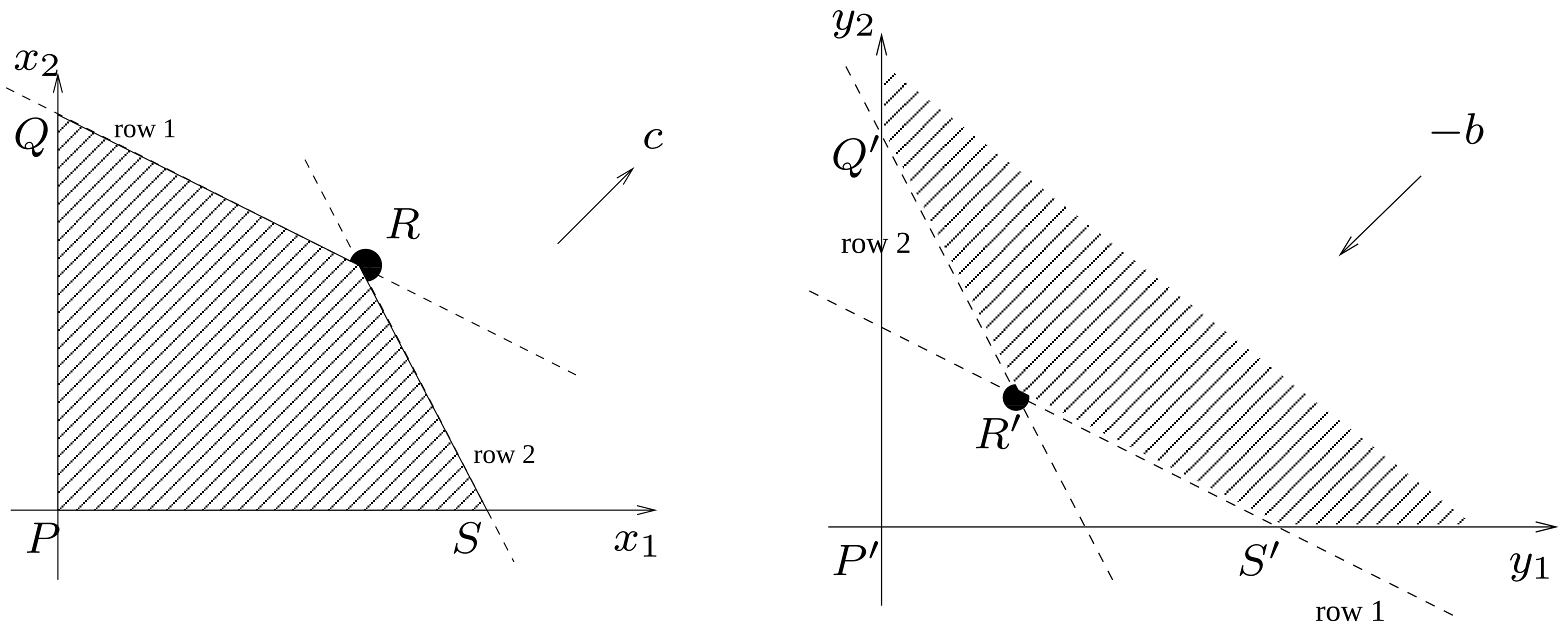
$$\left.\begin{array}{rl} \max\limits_{x_1,x_2} & x_1 + x_2 \\ \text{s.t.} & x_1 + 2x_2 \le 2 \\ & 2x_1 + x_2 \le 2 \\ & x \ge 0 \end{array}\right\} \Rightarrow \left.\begin{array}{rl} -\min\limits_{x_1,x_2} & -x_1 - x_2 \\ \text{s.t.} & -x_1 - 2x_2 \ge -2 \\ & -2x_1 - x_2 \ge -2 \\ & x \ge 0 \end{array}\right\}$$

- Dual problem D and reformulation:

$$\left.\begin{array}{rl} -\max\limits_{y_1,y_2} & -2y_1 - 2y_2 \\ \text{s.t.} & -y_1 - 2y_2 \le -1 \\ & -2y_1 - y_2 \le -1 \\ & y \ge 0 \end{array}\right\} \Rightarrow \left.\begin{array}{rl} \min\limits_{y_1,y_2} & 2y_1 + 2y_2 \\ \text{s.t.} & y_1 + 2y_2 \ge 1 \\ & 2y_1 + y_2 \ge 1 \\ & y \ge 0 \end{array}\right\}$$

Primal and Dual

- Graphical representation

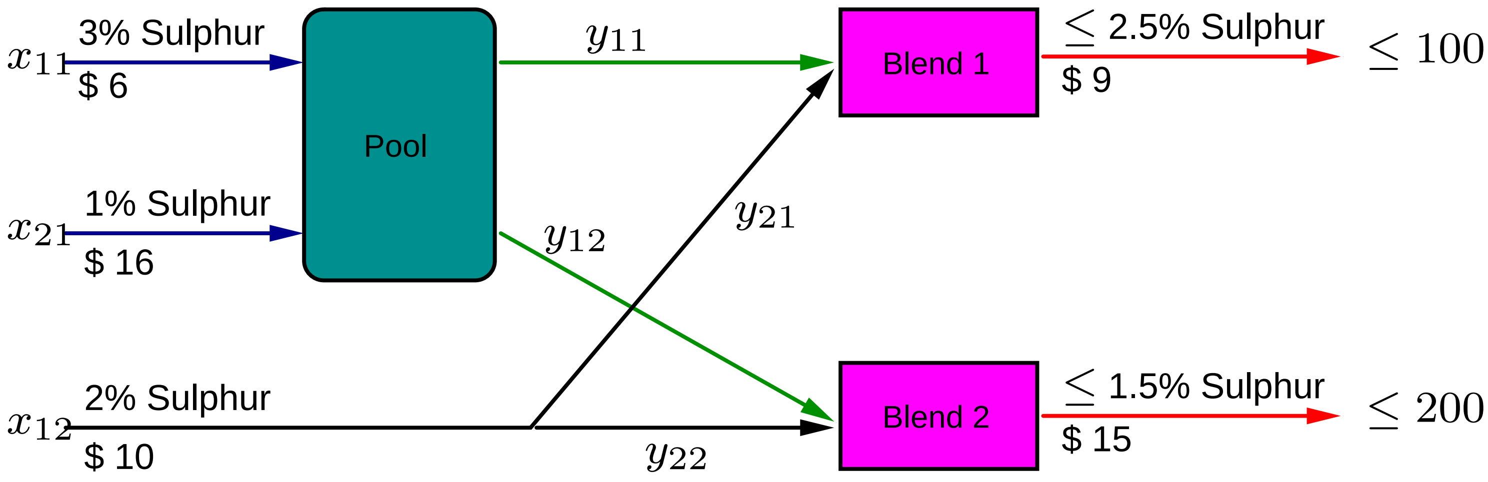


- $R^\mathsf{T} = (\frac{2}{3}, \frac{2}{3})$, $(R')^\mathsf{T} = (\frac{1}{3}, \frac{1}{3})$
- By strong duality, $c^\mathsf{T} R = (R')^\mathsf{T} b$
- $c^\mathsf{T} R = \frac{2}{3} + \frac{2}{3} = \frac{4}{3} = \frac{1}{3}2 + \frac{1}{3}2 = (R')^\mathsf{T} b$

Application: SLP

- SLP: Successive Linear Programming
- Heuristic for solving bilinear programming problems
- Formulation includes bilinear terms $x_i y_j$ where $i \in I, j \in J$
- Fact: fix $x_i, i \in I$, get LP$_1$; fix $y_j, j \in J$, get LP$_2$
- Algorithm: solve LP$_1$, get values for y, update and solve LP$_2$, get values for x, update and solve LP$_1$, and so on
- Iterate until no more improvement

Haverly's pooling problem



$$
\left\{
\begin{array}{ll}
\min\limits_{x,y,p} & 6x_{11} + 16x_{21} + 10x_{12} - \\
& \quad -9(y_{11} + y_{21}) - 15(y_{12} + y_{22}) \\
\text{s.t.} & x_{11} + x_{21} - y_{11} - y_{12} = 0 \text{ linear} \\
& x_{12} - y_{21} - y_{22} = 0 \text{ linear} \\
& y_{11} + y_{21} \leq 100 \text{ linear} \\
& y_{12} + y_{22} \leq 200 \text{ linear} \\
& 3x_{11} + x_{21} - p(y_{11} + y_{12}) = 0 \\
& py_{11} + 2y_{21} \leq 2.5(y_{11} + y_{21}) \\
& py_{12} + 2y_{22} \leq 1.5(y_{12} + y_{22})
\end{array}
\right.
$$