Operations Research (Master's Degree Course)

3. Linear Programming

Silvano Martello
DEI "Guglielmo Marconi", Università di Bologna, Italy

Graphical solution in R^2

- Let's go back to the **Production Planning** example seen in the **Introduction**.

- **Mathematical model:**

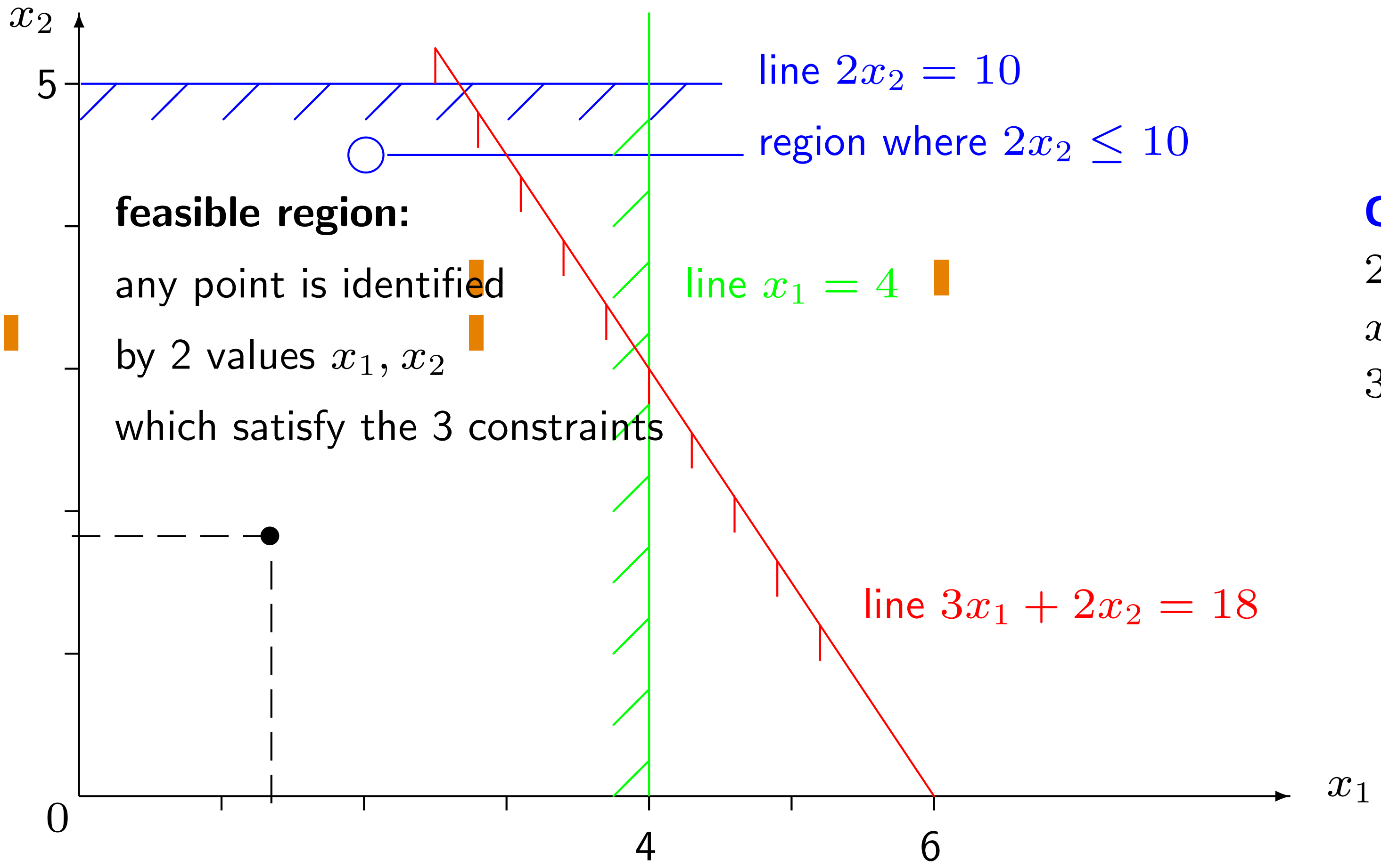
$$
\begin{array}{lrcrcl}
\max\ z = & 30\,x_1 & + & 50\,x_2 & & \\
 & x_1 & & & \le & 4 \\
 & & & 2\,x_2 & \le & 10 \\
 & 3\,x_1 & + & 2\,x_2 & \le & 18 \\
 & x_1, & & x_2 & \ge & 0
\end{array}
$$

- When a linear programming problem involves only two variables, it can be solved through a geometric approach (**graphical solution**).

- The **graphical solution** allows to understand some fundamental aspects of linear programming.

Graphical solution in R^2 (cont'd)

Cartesian coordinate system of variables x_1 and x_2, with $x_1 \geq 0$ and $x_2 \geq 0$:



Constraints:

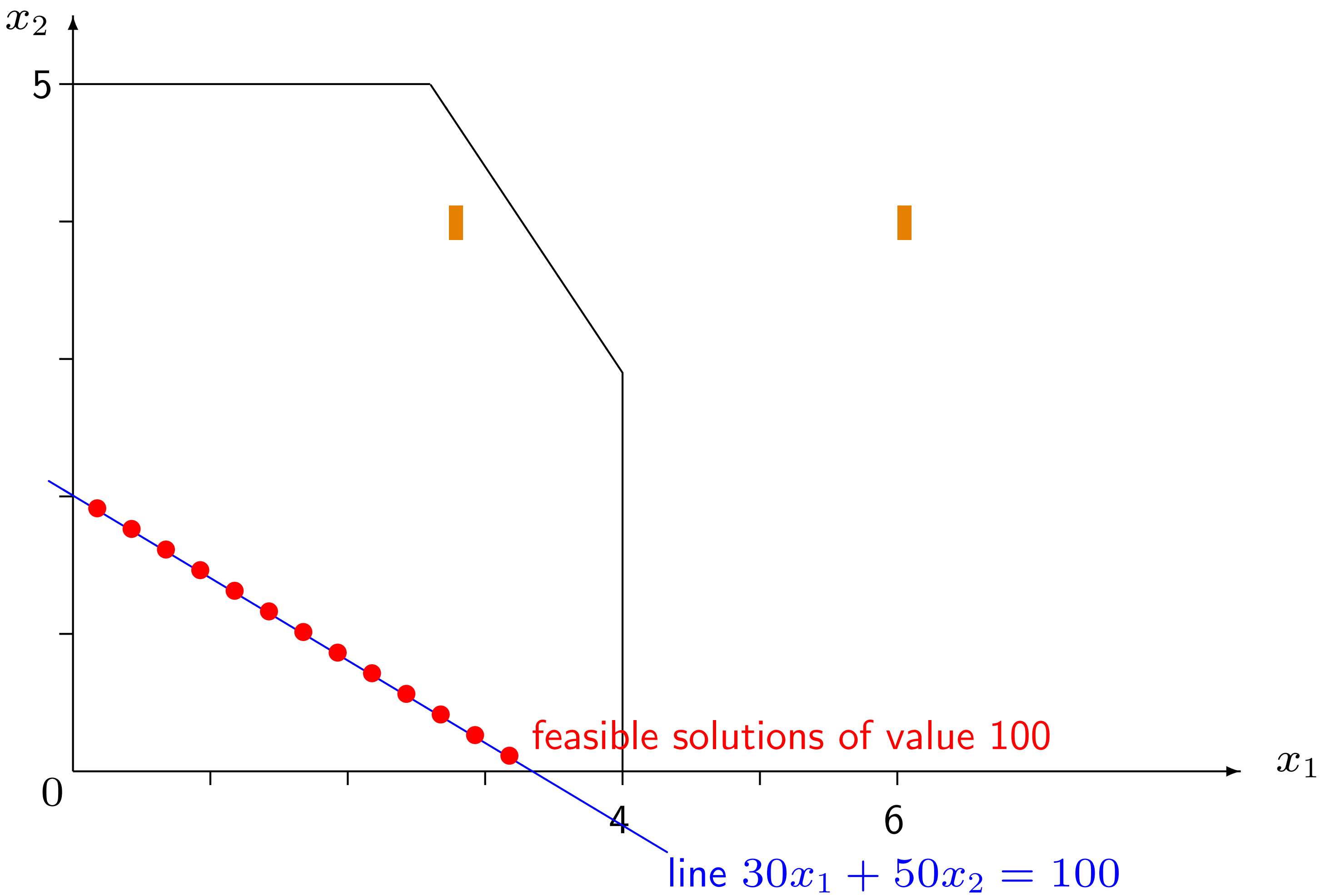
$2x_2 \leq 10$

$x_1 \leq 4$

$3x_1 + 2x_2 \leq 18$

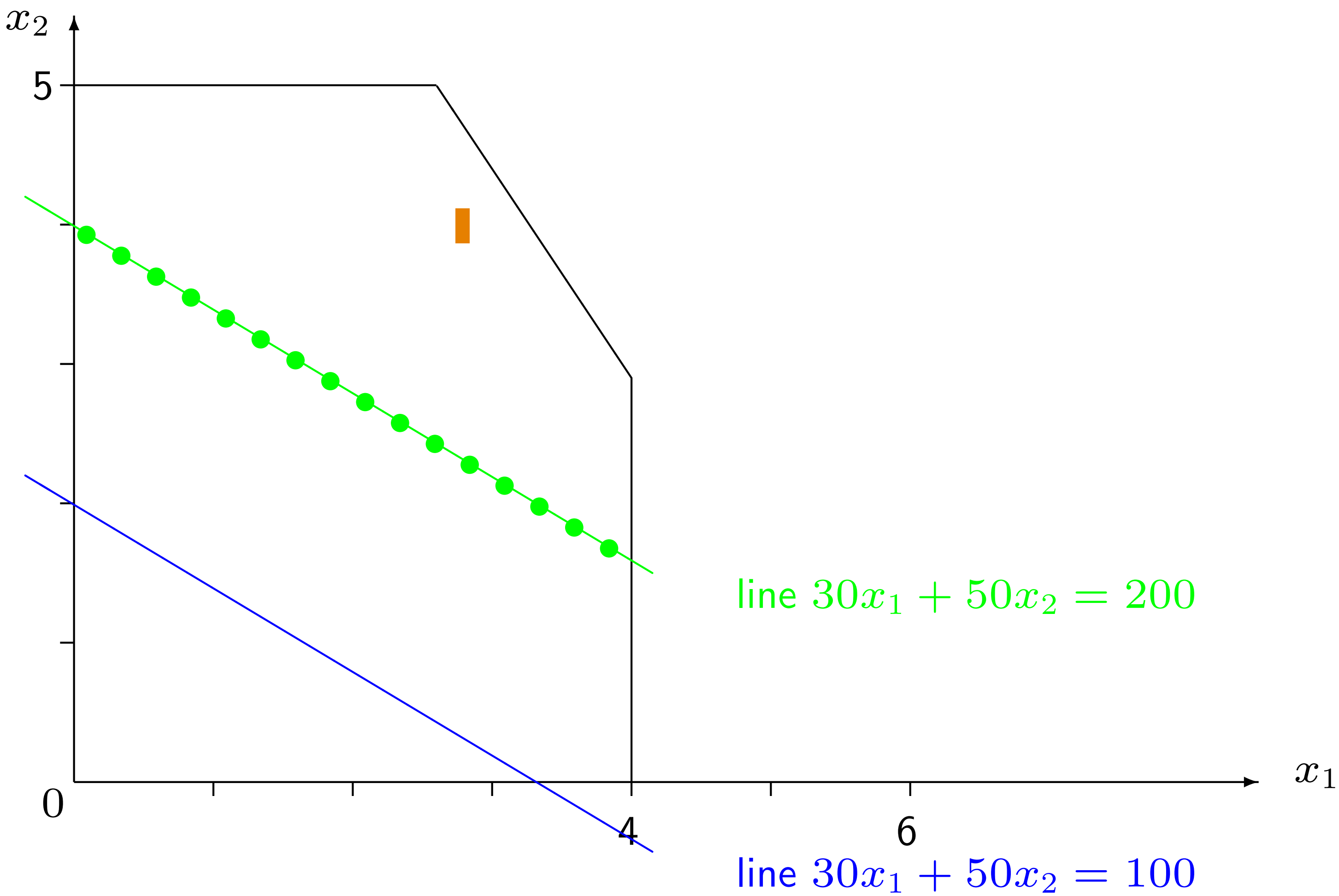
Graphical solution in R^2 (cont'd)

Objective function: $\max z = 30x_1 + 50x_2$, with z **unknown**:



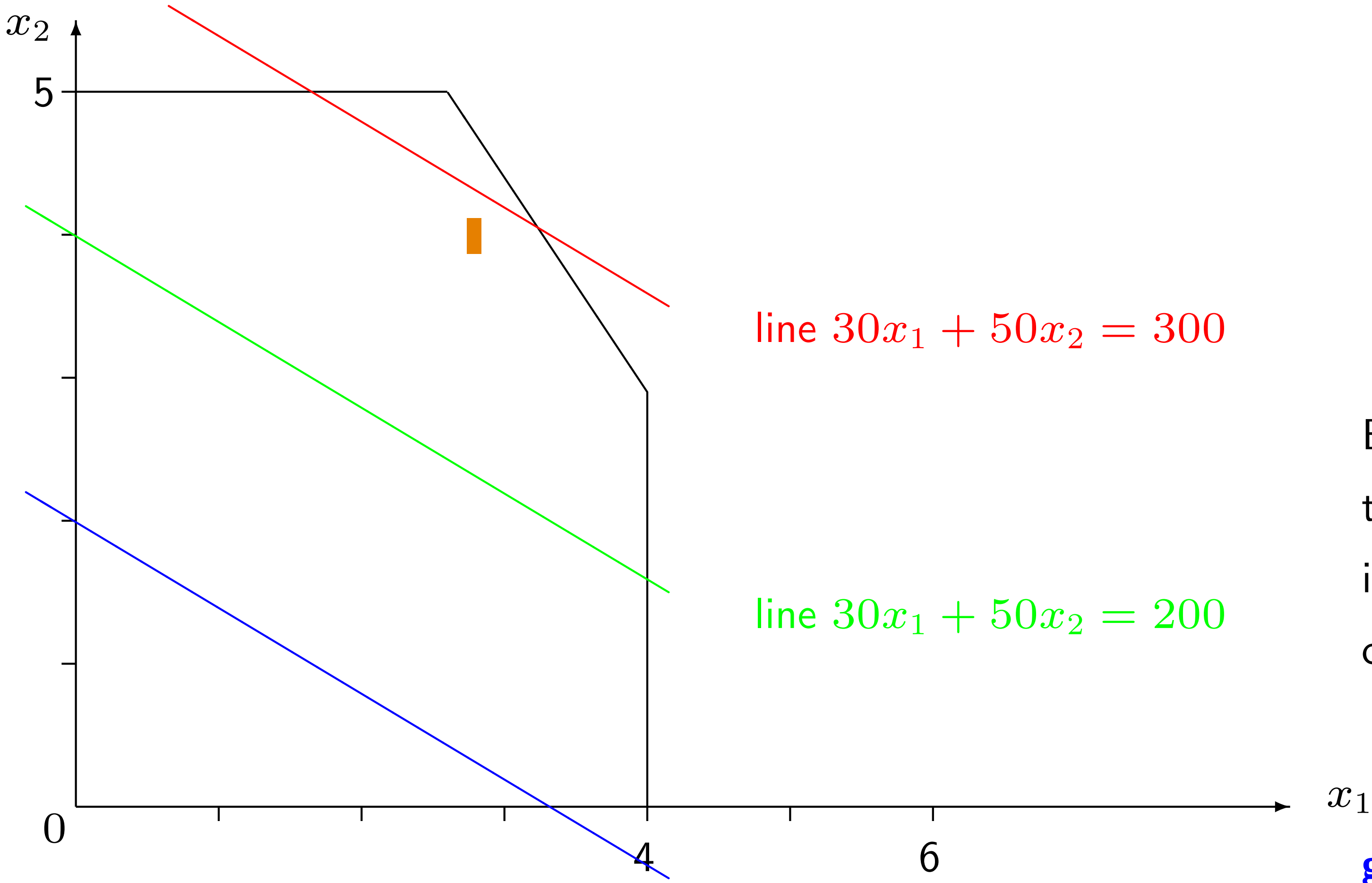
Graphical solution in R^2 (cont'd)

Objective function: $\max z = 30x_1 + 50x_2$, with z **unknown**:



Graphical solution in R^2 (cont'd)

Objective function: $\max z = 30x_1 + 50x_2$, with z **unknown**:

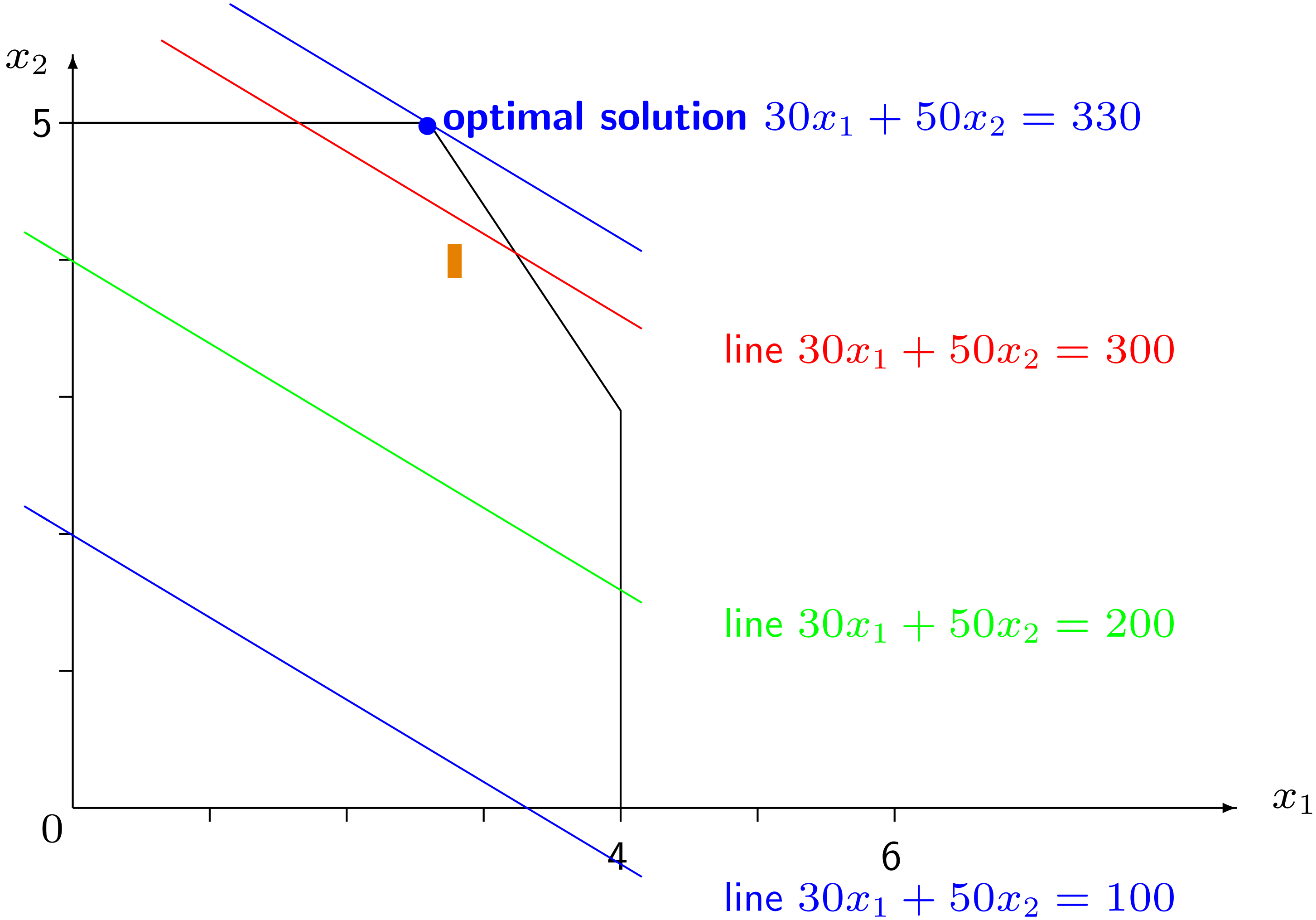


Bundle of parallel lines: the solution value increases in the direction of the **gradient**

gradient $= (\frac{\partial z}{\partial x_1}, \frac{\partial z}{\partial x_2}) = (30, 50)$

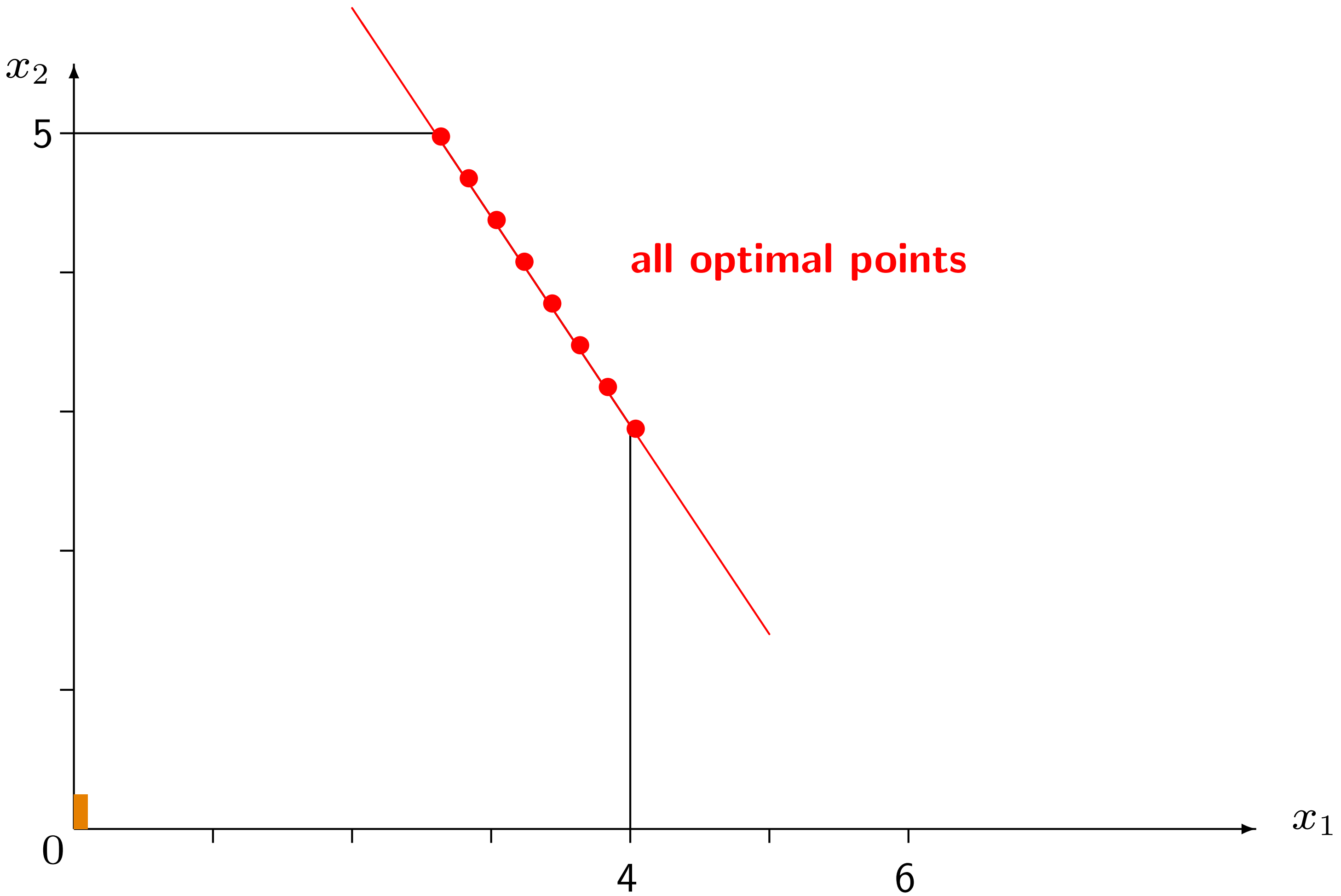
Graphical solution in R^2 (cont'd)

Objective function: $\max z = 30x_1 + 50x_2$, with z **unknown**:



Graphical solution in R^2 (cont'd)

- **Question:** Does this mean that only vertices can provide the optimal solution?
- **Answer:** No! For example, if the objective function is $\max z = 3x_1 + 2x_2$:



- **Conclusion: No**, but it is **enough to consider the vertices** to find an optimal solution!

Forms of Linear Programming

- **General form:**

$A =$ integer $m \times n$ matrix;
$b =$ integer vector of m elements;
$c =$ integer vector of n elements;

$$\min c'x$$
$$\begin{array}{rcll} a_i'x & = & b_i & i \in M \\ a_i'x & \geq & b_i & i \in \overline{M} \\ x_j & \geq & 0 & j \in N \\ x_j & \gtreqless & 0 & j \in \overline{N} \end{array} \qquad (\textbf{Note: } \gtreqless \Leftrightarrow >, < \text{ or } =)$$

- **Example:**

$$\begin{array}{lccccccc} \min & x_1 & & & + & x_3 & & \\ & & & x_2 & - & 2x_3 & = & 4 \\ & x_1 & + & x_2 & & & \geq & 3 \\ & x_1 & , & x_2 & & & \geq & 0 \\ & & & & & x_3 & \gtreqless & 0 \end{array}$$

$m = 2$, $n = 3$; M={1}, $\overline{M}$={2}; N={1,2}, $\overline{N}$={3}.

$$A = \begin{bmatrix} 0 & 1 & -2 \\ 1 & 1 & 0 \end{bmatrix}, \quad b = \begin{bmatrix} 4 \\ 3 \end{bmatrix}, \quad c = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}.$$

A notable application: The diet problem

- A cattle-breeder wants to find the best food mixture to buy in order to conveniently feed cattle.
- **Input data:**
  - n available foods ;
  - m nutrients in each food;
  - $a_{ij} =$ quantity of the ith nutrient in 1 unit of the jth food $(i = 1, \ldots, m;\ j = 1, \ldots, n)$;
  - $r_i =$ requirement (in a week, month, . . .) of the ith nutrient $(i = 1, \ldots, m)$;
  - $c_j =$ cost of 1 unit of the jth food $(j = 1, \ldots, n)$.
- **Objective:**
  - buy quantities of the various foods to guarantee the requirement of each nutrient
  - by minimizing the overall cost.

A notable application: The diet problem (cont'd)

- Numerical example:

Nutrients	Foods (content g/Kg) Meat	Milk	Soy	Requirement(g)
Proteins	500	300	300	800
Fat	300	300	100	400
Carbohydrate	0	100	200	2000
Cost (€/Kg)	5	1.5	0.8	

- Problem:

$$
\begin{array}{llcccccc}
\min z = & 50\,x_1 & + & 15\,x_2 & + & 8\,x_3 & & \\
\text{s.t.} & 5\,x_1 & + & 3\,x_2 & + & 3\,x_3 & \geq & 8 \\
 & 3\,x_1 & + & 3\,x_2 & + & x_3 & \geq & 4 \\
 & & & x_2 & + & 2\,x_3 & \geq & 20 \\
 & x_1, & & x_2, & & x_3 & \geq & 0
\end{array}
$$

(better to use integer values)

- By multiplying or dividing constraints/objective function by a positive constant the problem is unchanged (but remind to congruently divide/multiply the solution).
- **Model:** n variables x_j (= quantity of the jth food to buy) ($j = 1, \ldots, n$)

$$
\begin{array}{ll}
\min & c'x \\
 & Ax \geq r \\
 & x \geq 0
\end{array}
$$

- All '$\geq$' constraints, all non negative variables: LP in **canonical form**.

Forms of Linear Programming (cont'd)

- LP in **canonical form**:

$$\begin{aligned} \min c'x & \\ Ax & \geq b \\ x & \geq 0 \end{aligned}$$

- LP in **standard form**:

$$\begin{aligned} \min c'x & \\ Ax & = b \\ x & \geq 0 \end{aligned}$$

- **The simplex algorithm solves problems in <u>standard form</u> with <u>$m < n$</u>.**
- Hence we need to ensure that there is no loss of generality, i.e., that:
  - the case $m \geq n$ has no interest;
  - the 3 forms are equivalent.
- By assuming that A is of rank m,
  - $m > n$ cannot occur (no solution);
  - if $m = n$ $\exists$ only one solution to $Ax = b$ (i.e., $x = A^{-1}b$);
  - if $m < n$ $\exists$ ∞ solutions to $Ax = b$
    (the system has $n - m$ degrees of freedom);
    (the value of $n - m$ variables can be arbitrarily decided)
  - the simplex algorithm finds the optimal solution among the feasible ones ($\Leftrightarrow$ $x \geq 0$), if any.

The three forms are equivalent

1. general form $\longrightarrow$ canonical form:

$$\alpha)\ \sum_{j=1}^{n} a_{ij}x_j = b_i \longrightarrow \begin{cases} \displaystyle\sum_{j=1}^{n} a_{ij}x_j & \geq & b_i \\ \displaystyle\sum_{j=1}^{n} (-a_{ij})x_j & \geq & -b_i \end{cases}$$

$$\beta)\ \ x_j \gtreqless 0 \qquad \longrightarrow \begin{cases} x_j = x_j^+ - x_j^- \\ x_j^+ \geq 0,\ \ x_j^- \geq 0 \end{cases}$$

2. general form $\longrightarrow$ standard form:

$$\alpha)\ \sum_{j=1}^{n} a_{ij}x_j \geq b_i \longrightarrow \begin{cases} \displaystyle\sum_{j=1}^{n} a_{ij}x_j - s_i = b_i \\ s_i \geq 0 \quad (\textbf{surplus variable}) \end{cases}$$

- $1.\alpha)$ increases m; $1.\beta)$ and $2.\alpha)$ increase n.

3. if the constraint is $\displaystyle\sum_{j=1}^{n} a_{ij}x_j \leq b_i \longrightarrow \begin{cases} \displaystyle\sum_{j=1}^{n} a_{ij}x_j + s_i = b_i \\ s_i \geq 0 \quad (\textbf{slack variable}) \end{cases}$

The three forms are equivalent (cont'd)

- **Example:** general form

$$
\begin{array}{lcccccccc}
\min & x_1 & & & + & x_3 & & \\
 & & & x_2 & - & 2x_3 & = & 4 \\
 & x_1 & + & x_2 & & & \geq & 3 \\
 & x_1 & , & x_2 & & & \geq & 0 \\
 & & & & & x_3 & \lessgtr & 0
\end{array}
$$

- Equivalent canonical form:

$$
\begin{array}{lcccccccccc}
\min & x_1 & & & + & x_3^+ & - & x_3^- & & \\
 & & & x_2 & - & 2x_3^+ & + & 2x_3^- & \geq & 4 \\
 & & - & x_2 & + & 2x_3^+ & - & 2x_3^- & \geq & -4 \\
 & x_1 & + & x_2 & & & & & \geq & 3 \\
 & x_1 & , & x_2 & , & x_3^+ & , & x_3^- & \geq & 0
\end{array}
$$

- Equivalent standard form:

$$
\begin{array}{lcccccccccccc}
\min & x_1 & & & + & x_3^+ & - & x_3^- & & & & \\
 & & & x_2 & - & 2x_3^+ & + & 2x_3^- & & & = & 4 \\
 & x_1 & + & x_2 & & & & & - & s_2 & = & 3 \\
 & x_1 & , & x_2 & , & x_3^+ & , & x_3^- & , & s_2 & \geq & 0
\end{array}
$$

Linear Independence (recall)

- A set of m columns (vectors) may or may not be **Linearly independent**.
- It is **NOT** if a column can be expressed as a linear combination of the others. For example,

$$B = \begin{bmatrix} 1 & 3 & 9 \\ -1 & 0 & -3 \\ 2 & -1 & 4 \end{bmatrix}; \begin{bmatrix} 9 \\ -3 \\ 4 \end{bmatrix} = 3 \begin{bmatrix} 1 \\ -1 \\ 2 \end{bmatrix} + 2 \begin{bmatrix} 3 \\ 0 \\ -1 \end{bmatrix}.$$

  - Hence a linear combination of the columns, with **non-zero coefficients** can produce $\underline{\mathbf{0}}$:

$$\begin{bmatrix} 9 \\ -3 \\ 4 \end{bmatrix} - 3 \begin{bmatrix} 1 \\ -1 \\ 2 \end{bmatrix} - 2 \begin{bmatrix} 3 \\ 0 \\ -1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix};$$

  - $\det(\mathbf{B}) = 0$; the matrix is **not invertible (singular)**.
- If instead the columns **ARE** linearly independent, e.g., $B = \begin{bmatrix} 3 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & 4 \end{bmatrix}$;
  - no column can be expressed as a linear combination of the others;
  - the only linear combination of the columns that can produce $\underline{\mathbf{0}}$ has all null coefficients:

$$r_1 \begin{bmatrix} 3 \\ 0 \\ 0 \end{bmatrix} + r_2 \begin{bmatrix} 0 \\ -2 \\ 0 \end{bmatrix} + r_3 \begin{bmatrix} 0 \\ 0 \\ 4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \iff r_1 = r_2 = r_3 = 0;$$

  - $\det(\mathbf{B}) \neq 0$; the matrix is **invertible (non-singular)**.

Basic solutions

- **Assumption 1: A contains m linearly independent columns A_j** ($\Leftrightarrow$ A is of rank m).

  Important: the algorithm must detect violated assumptions, if any.
- **Basis** of A = collection of m linearly independent columns:

$$\mathcal{B} = \{A_{\beta(1)}, \dots, A_{\beta(m)}\}$$

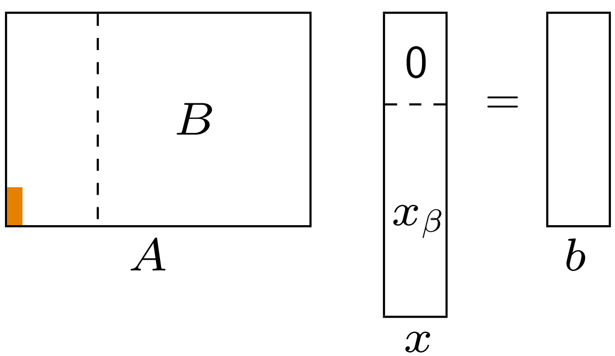
- $\mathcal{B}$ corresponds to an $m \times m$ non singular matrix:

$$B = [A_{\beta(i)}]$$

- **Basic solution** x corresponding to $\mathcal{B}$:

  $x_j =$ 0 for $A_j \notin \mathcal{B}$ (*non basic variables*);

  $x_{\beta(k)} =$ kth component of $B^{-1}b$ $(k = 1, \dots, m)$ (*basic variables*):



$\Rightarrow$ the unique solution $x_\beta = B^{-1}b$ $(x_j = 0 \ \forall A_j \notin \mathcal{B})$
- satisfies $Ax = b$;
- does not necessarily satisfy $x \geq 0$.

Basic solutions (cont'd)

- Example:

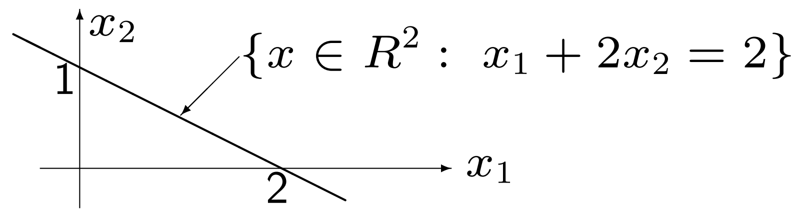
$$\begin{array}{lccccccccccccccc}
\min & & & 2x_2 & & & + & x_4 & & & & & + & 5x_7 & & \\
& x_1 & + & x_2 & + & x_3 & + & x_4 & & & & & & & = & 4 \\
& x_1 & & & & & & & + & x_5 & & & & & = & 2 \\
& & & & & x_3 & & & & & + & x_6 & & & = & 3 \\
& & & 3x_2 & + & x_3 & & & & & & & + & x_7 & = & 6 \\
& x_1 & , & x_2 & , & x_3 & , & x_4 & , & x_5 & , & x_6 & , & x_7 & \geq & 0
\end{array}$$

  - $\mathcal{B} = \{A_4,\ A_5,\ A_6,\ A_7\ \} \ \Rightarrow\ B = I$.
    Basic solution: $x = (0, 0, 0, 4, 2, 3, 6)$ feasible.
  - $\mathcal{B} = \{A_2, A_5, A_6, A_7\} \ \Rightarrow\ B^{-1} = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ -3 & & & 1 \end{bmatrix}$

    Basic solution: $x = (0, 4, 0, 0, 2, 3, -6)$ unfeasible.
- $F = \{x \in R^n \ :\ Ax = b,\ x \geq 0\}$.
- **Basic Feasible Solution (BFS)** $=$ basic solution $\in F$ ($\Leftrightarrow\ x \geq 0$).
- **Assumption 2:** $F \neq \emptyset$.
- **Assumption 3: in F, the objective function $c'x$ is bounded from below** (its value does not tend to $-\infty$), i.e., F **is bounded** in the direction in which $c'x$ decreases.

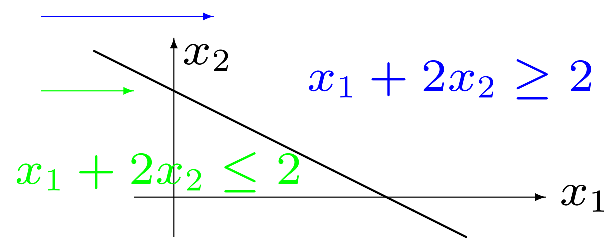
Convex polytopes

- Given a space R^d, a vector $h \neq 0$ and a scalar g: **Hyperplane** $= \{x \in R^d : h'x = g\}$
  - Example: In R^2: $h' = (1,2)$, $g=2$



  - In R^3: a plane.

- A hyperplane defines 2 **Halfspaces**: $\{x \in R^d : h'x \geq g\}$
  $\{x \in R^d : h'x \leq g\}$

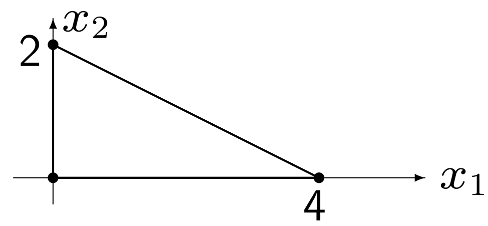


- A halfspace S is a convex set ($\forall$ 2 points $\in S$, the line segment joining the $\in S$).
- $\Rightarrow$ The intersection of halfspaces is convex.
- **Polytope (Convex Polytope)** $=$ intersection of a finite number of halfspaces, if bounded and not empty.
- The constraints of an LP (in canonical form) define an intersection of halfspaces, hence a polytope.

Convex polytopes (cont'd)

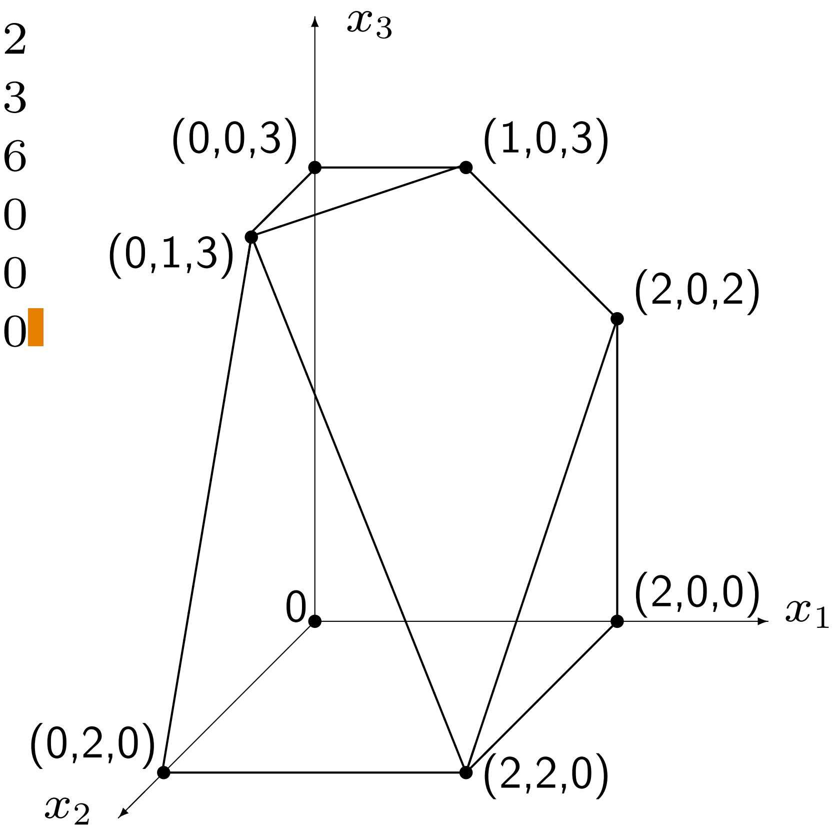
- **Example:** In R^2 :

$$\begin{array}{rcrl} x_1 & + & 2x_2 & \leq 4 \\ x_1 & & & \geq 0 \\ & & x_2 & \geq 0 \end{array}$$



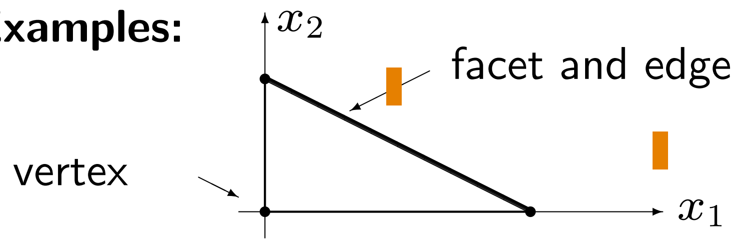
- **Example:** In R^3 :

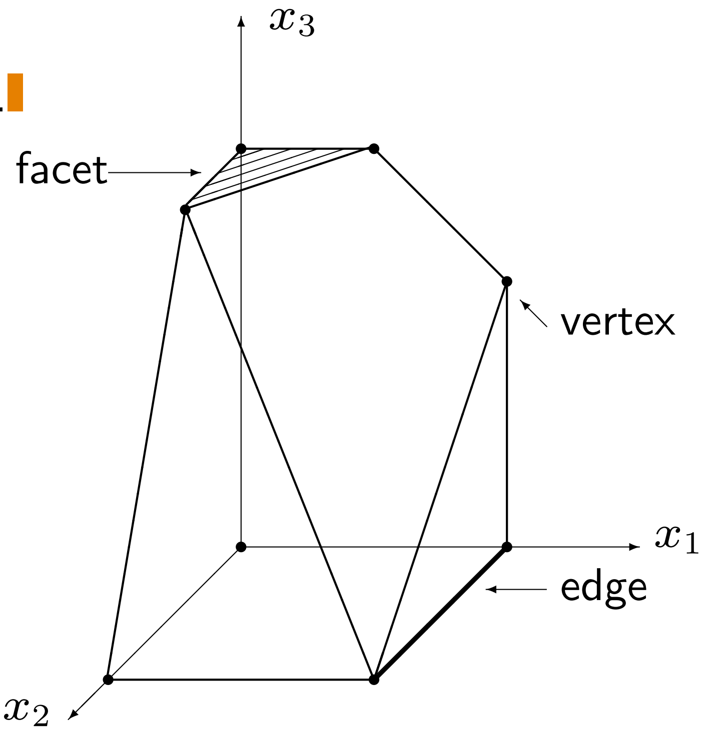
$$\begin{array}{rcrcrl} x_1 & + & x_2 & + & x_3 & \leq 4 \\ x_1 & & & & & \leq 2 \\ & & & & x_3 & \leq 3 \\ & & 3x_2 & + & x_3 & \leq 6 \\ x_1 & & & & & \geq 0 \\ & & x_2 & & & \geq 0 \\ & & & & x_3 & \geq 0 \end{array}$$



Convex polytopes (cont'd)

- P = polytope;
  - H = hyperplane;
  - HS = halfspace defined by H;
  - $f = P \cap HS$;
  - if $\emptyset \neq f \subseteq H$, f is called a **face** of P.
- If d = **dimension of the polytope** (=minimum dimension of a space that contains it):
  - **facet** = face of dimension $d-1$;
  - **vertex** = face of dimension 0 (a point);
  - **edge** = face of dimension 1 (a line segment).
- **Examples:**

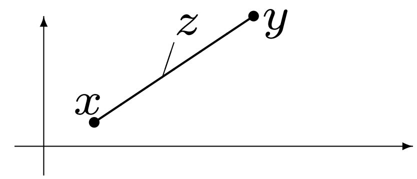




Convex polytopes (cont'd)

- **Convex combination of 2 points** $x, y \in R^n$ = point $z \in R^n$:

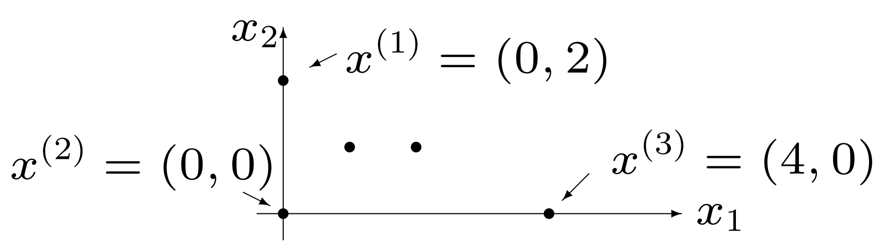
  $z = \lambda x + (1 - \lambda) y$ (with $0 \leq \lambda \leq 1$).



  By varying λ, z describes all points of line segment $[x, y]$.

- **Convex combination of p points** $x^{(1)}, \ldots, x^{(p)} \in R^n$:

  $z = \sum_{i=1}^{p} \alpha_i x^{(i)}$ (with $\sum_{i=1}^{p} \alpha_i = 1$, $\alpha_i \geq 0 \ \forall i$).



  - $\alpha = (\frac{1}{2}, 0, \frac{1}{2})$: $z = \frac{1}{2}(0, 2) + 0(0, 0) + \frac{1}{2}(4, 0) = (2, 1)$;
  - $\alpha = (\frac{1}{2}, \frac{1}{4}, \frac{1}{4})$: $z = \frac{1}{2}(0, 2) + \frac{1}{4}(0, 0) + \frac{1}{4}(4, 0) = (1, 1)$.

Convex polytopes (cont'd)

- **Property** *Every point of a polytope is a convex combination of the vertices and conversely.* (**Proof** (complicated) omitted).

- **Property** *A vertex is not a* **strict** *convex combination* (*i.e., with* $0 < \lambda < 1$) *of two distinct points of the polytope.*

  Proof (sufficiency) Let P be a polytope, $v \in P$ a vertex and suppose there are two other points $y, w \in P$ such that

  $$v = \lambda y + (1 - \lambda) w;$$

  v vertex $\Rightarrow \exists$ halfspace $HS = \{x : h'x \leq g\} : HS \cap P = v$

  $\Rightarrow y, w \notin HS \quad \Rightarrow \quad h'y > g$ and $h'w > g$

  $\Rightarrow h'v = h'(\lambda y + (1 - \lambda) w) > g \quad \Rightarrow \quad v \notin HS,$ absurd. $\square$

  (**Proof (necessity)** omitted).

Polytopes and Linear Programming

- **Property** *The constraints of an* LP *define a polytope.*

  Proof Immediate by considering the canonical form:

  $$\widehat{F} = \{x \in R^q : \widehat{A}x \geq b, x \geq 0\} \qquad \widehat{A}(m \times q)$$

  is an intersection of halfspaces, bounded ($\Leftarrow$ Assumption 3) and $\neq \emptyset$ (Assumption 2). $\square$

- $\widehat{F} \subseteq R^q$ has dimension $d \leq q$.

- By Adding m surplus variables, we get the standard form

  $Ax = b$ with $A = (\widehat{A} \mid -I)$, so A is an $m \times n$ matrix;

  $\Rightarrow$ **the polytope has dimension** $d \leq n - m$**.**

- **Fundamental relationship between vertices and basic solutions:**

  Theorem Given the polytope P defined by the constraints of an LP, a necessary and sufficient condition for a point to be a vertex is that the corresponding vector x be a BFS.

  Proof We will separately proof sufficiency and necessity.

Polytopes and Linear Programming (cont'd)

- **Sufficiency** BFS $x_\beta = (x_{\beta(1)}, \dots, x_{\beta(m)})$ for a base $\mathcal{B} = \{A_{\beta(1)}, \dots, A_{\beta(m)}\} \Rightarrow$
  $$\sum_{A_j \in \mathcal{B}} x_j A_j = b.$$
- We will show that x is a vertex, i.e., it is not a strict convex combination of two other distinct points $y,\ w \in P$.
- Assume it is, i.e., $x = \lambda y + (1-\lambda) w$ with $0 < \lambda < 1$.
- $y,\ w \in P \Longrightarrow y_j,\ w_j \geq 0\ \forall\ j$.
  $\Rightarrow y_j = w_j = 0\ \forall\ A_j \notin \mathcal{B}\ \ (\Leftarrow x_j = 0) \Rightarrow$
- $$\sum_{A_j \in \mathcal{B}} y_j A_j = b;$$
  $$\sum_{A_j \in \mathcal{B}} w_j A_j = b \Rightarrow$$
- $$\sum_{A_j \in \mathcal{B}} (x_j - y_j) A_j = 0;$$
  $$\sum_{A_j \in \mathcal{B}} (x_j - w_j) A_j = 0.$$
- $A_{\beta(1)}, \dots, A_{\beta(m)}$ are linearly independent $\Rightarrow x_j - y_j = x_j - w_j = 0\ \forall\ A_j \in \mathcal{B}\ \ \Rightarrow$
  $x \equiv y \equiv w$. $\square$

Polytopes and Linear Programming (cont'd)

- **Necessity** $x \in F$ vector corresponding to the vertex. $\mathcal{B} = \{A_j : x_j > 0\}$.
- We will show that $A_j \in \mathcal{B}$ are linearly independent.
- Suppose they are not: this implies that $\exists\ d_j$ not all zero s.t.
- $\displaystyle\sum_{A_j \in \mathcal{B}} d_j A_j = 0;$ $\qquad (\alpha)$
- $x \in F \Rightarrow \displaystyle\sum_{A_j \in \mathcal{B}} x_j A_j = b,\ x_j \geq 0\ \forall j;$ $\quad (\beta)$
- now multiply (α) by a scalar ϑ, and add/subtract from (β): $\sum_{A_j \in \mathcal{B}} (x_j \pm \vartheta d_j) A_j = b$
- $x_j > 0\ \forall A_j \in \mathcal{B} \Rightarrow \exists\ \vartheta$ (sufficiently small) s.t. $x_j \pm \vartheta d_j \geq 0\ \forall A_j \in \mathcal{B}$
- $\Leftrightarrow \exists$ two points, defined by:

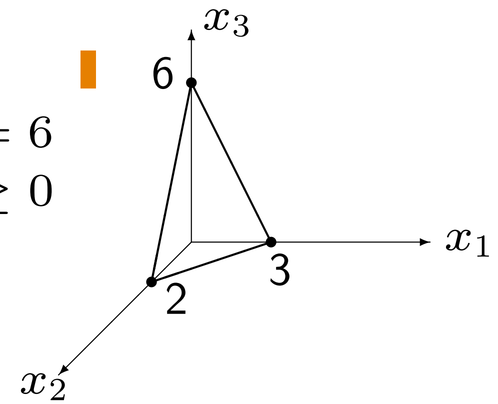
$$\begin{cases} x_j^{(1)} = x_j + \vartheta d_j,\ x_j^{(2)} = x_j - \vartheta d_j & \text{if } A_j \in \mathcal{B}, \\ x_j^{(1)} = x_j^{(2)} = 0, & \text{if } A_j \notin \mathcal{B} \end{cases}$$

  s.t. $x^{(1)}, x^{(2)} \in F$, and $x = \frac{1}{2} x^{(1)} + \frac{1}{2} x^{(2)}$ ($\Leftrightarrow$ the point is not a vertex).
- Hence $A_j \in \mathcal{B}$ are linearly independent $\Rightarrow |\mathcal{B}| \leq m$;
- since A is of rank m, if $|\mathcal{B}| < m$ we can add columns to obtain $\mathcal{B}'$ linearly independent with $|\mathcal{B}'| = m \Rightarrow x$ is a BFS. $\square$

Polytopes and Linear Programming (cont'd)

- **Example**
- **Standard form:** $\min c'x$

$$\begin{array}{ccccccc} 2x_1 & + & 3x_2 & + & x_3 & = 6 \\ x_1 & , & x_2 & , & x_3 & \geq 0 \end{array}$$



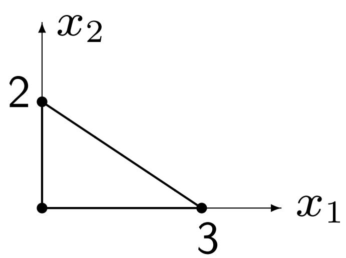
BFSs:

$$\begin{array}{llll} \mathcal{B} = & \{A_1\}: & x_2 = x_3 = 0, & x_1 = 3; \\ \mathcal{B} = & \{A_2\}: & x_1 = x_3 = 0, & x_2 = 2; \\ \mathcal{B} = & \{A_3\}: & x_1 = x_2 = 0, & x_3 = 6. \end{array}$$

- **Canonical form:**

$\min c'x$

$$\left.\begin{array}{ccccc} -2x_1 & - & 3x_2 & \geq & -6 \\ x_1 & , & x_2 & \geq & 0 \end{array}\right\} \Leftrightarrow \left\{\begin{array}{r} 2x_1 + 3x_2 + s_1 = 6 \\ s_1 \geq 0 \end{array}\right.$$



Polytopes and Linear Programming (cont'd)

- **Theorem** *For any* LP *there exists an optimal vertex (i.e., an optimal basis)*

  Proof c = cost vector; $x^{(0)}$ = optimal solution; $x^{(1)}, \ldots, x^{(p)}$ = vertices of P.

  $$x^{(0)} \in P \Rightarrow x^{(0)} = \sum_{i=1}^{p} \alpha_i x^{(i)} \quad \left(\sum_{i=1}^{p} \alpha_i = 1, \quad \alpha_i \geq 0 \ \ \forall\, i \right);$$

  let $x^{(j)}$ be s.t. $c'x^{(j)} = \min_{1 \leq i \leq p}\{c'x^{(i)}\}$;

  $$c'x^{(0)} = c' \sum_{i=1}^{p} \alpha_i x^{(i)} \geq c'x^{(j)} \sum_{i=1}^{p} \alpha_i = c'x^{(j)} \Rightarrow c'x^{(j)} = c'x^{(0)}. \ \square$$

- **Corollary** *Any convex combination of optimal vertices is optimal.*

  Proof $x^{(1)}, \ldots, x^{(q)}$ = optimal vertices;

  $$x = \sum_{i=1}^{q} \alpha_i x^{(i)} \ \Rightarrow \ c'x = \sum_{i=1}^{q} \alpha_i c'x^{(i)} = c'x^{(1)} \sum_{i=1}^{q} \alpha_i = c'x^{(1)}. \ \square$$

- **Hence an LP can be solved in a finite number of steps** by examining
  - all vertices of P, i.e.,
  - all BFSs of $Ax = b$, i.e.,
  - all combinations of m columns of A, and testing feasibility.

- **Simplex algorithm**: method to only explore a small subset of the vertices of P.

Polytopes and Linear Programming (cont'd)

- **Degenerate Bases**
- A base $\mathcal{B}$ uniquely determines a BFS, so $(\text{BFS}' \neq \text{BFS}'') \Rightarrow (\mathcal{B}' \neq \mathcal{B}'')$.
- Instead $(\mathcal{B}' \neq \mathcal{B}'') \not\Rightarrow (\text{BFS}' \neq \text{BFS}'')$. Indeed
- **Example** $A = \begin{bmatrix} 1 & 2 & 1 & 0 & 0 \\ 0 & 3 & 0 & 1 & 0 \\ 2 & 1 & 0 & 0 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 0 \\ 6 \\ 5 \end{bmatrix}.$

$$\mathcal{B}' = \{A_1, A_4, A_5\} : (B')^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -2 & 0 & 1 \end{bmatrix}, \quad x' = (0,0,0,6,5);$$

$$\mathcal{B}'' = \{A_3, A_4, A_5\} : (B'')^{-1} = I, \quad x'' = (\underset{\uparrow}{0}, \underset{\uparrow}{0}, \underset{\uparrow}{0}, 6, 5)$$

more than $n - m$ zeroes.

- A BFS is called **degenerate** if it contains more than $n - m$ zeroes.
- **Theorem** *If two distinct bases $\mathcal{B}'$ and $\mathcal{B}''$ correspond to the same BFS x, then x is degenerate.*

  Proof x has $n - m$ zeroes in those columns that are not in $\mathcal{B}'$ and additional zeroes in the columns of $\mathcal{B}' \setminus \mathcal{B}''(\neq \emptyset)$. $\square$