Algorithm Design and Analysis

David N. JANSEN, Bohua ZHAN
名 姓

算法设计与分析

詹博华，杨大卫

This week's content

这周的内容

- Today Wednesday:
  - Chapter 29: Linear Programming
    - 29.1–29.3
  - Exercises
- Tomorrow Thursday:
  - Exercise solutions
  - Chapter 29: Linear Programming
    - 29.4–

- 今天周三：
  - 第29章：线性规划
  - 练习
- 明天周四：
  - 练习题解答
  - 第29章：线性规划

Algorithm Design and Analysis

Linear Programming

David N. JANSEN
名 姓

算法设计与分析

线性规划

杨大卫

Ch. 29 29章

Optimization Problems

优化问题

- Optimiziation problems ask the question:
  “What is the best solution?”
  - Minimum spanning tree
    (repeatedly add the lightest allowed edge:
    Prim, Kruskal)
  - Shortest path
    (extend breadth-first search: Dijkstra;
    split by largest internal vertex: Floyd–Warshall)
  - Maximum flow (next week)
- This week: Linear Programming, the most general optimization problem in this course (use Simplex method)

- 优化问题问：
  “什么是最好的/最优的解决？”
  - 最小生成树
    （总最轻的允许的边：Prim，Kruskal）
  - 最短路径
    （扩展广度优先搜索：Dijkstra；
    分离在最大的中间结点：Floyd–Warshall）
  - 最大流（下周）
- 这周：线性规划，这个课程的最全面的优化问题（使用单纯形算法）

Example: Production Planning

- A company offers two products: decorated china and white china.
- If it produces only decorated china, it needs 8 employees, and it can produce 2000 pieces per week, which sell at a profit of ¥5/piece (¥1250/ employee).
  However, the company only has six employees.
- If it produces only white china, 4 employees would be enough, it can produce 3500 pieces per week, and the profit is ¥2/piece (¥1750/employee).
- How many should work on which product?

例如：生产计划

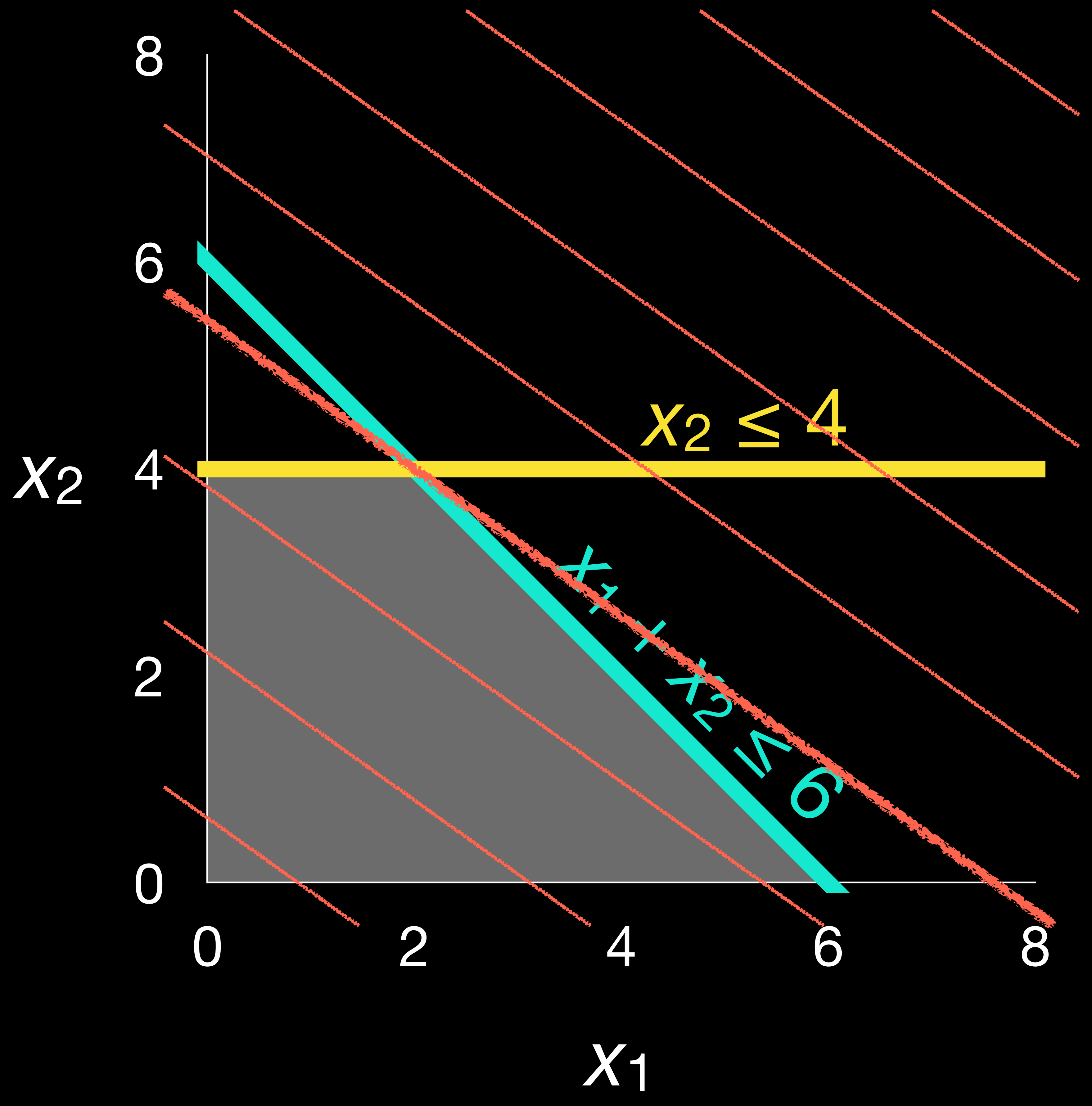
- 一家公司提供两种产品：
  装饰瓷器和白色瓷器。

- 如果它只生产装饰瓷器，它需要8个员工，每周可以生产2000件，销售利润为5元/件（1250元/名员工）。
  然而，该公司只有六名员工。
- 如果它只生产白色瓷器，4个员工就足够了，每周可生产3500件，利润为2元/件（1750元/人）。

- 有多少员工应该在哪种产品上工作?

Example: Production Planning

例如：生产计划



At most 6 employees 最多6个工人员

At most 4 produce white china 最多4个生产白色瓷器

Feasible region contains possible solutions 可行区域包含可能解决方案

等利润线

lines of equal profit:

$P = 1250x_1 + 1750x_2$

x_1 = employees that make decorated products 生产装饰瓷器的人员

x_2 = employees that make white products 生产白色瓷器的人员

Linear Programming Problem

线性规划问题

A linear program in standard form consists of:

- n variables $x_1, x_2, \ldots, x_n$.
  (We want to find optimal values for these variables.)
- real constants $c_1, c_2, \ldots, c_n \in \mathbb{R}$ to form a linear objective function $\Sigma\, c_j x_j$. (The optimal value maximizes the objective function.)
- (linear) constraints, expressed with a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ and a vector $\boldsymbol{b} \in \mathbb{R}^m$: $\Sigma\, a_{ij} x_j \leq b_i$ for $i = 1, 2, \ldots, m$
- nonnegativity constraints $x_j \geq 0$ for j = 1, 2, ..., n
  (The values of $\boldsymbol{x}$ must satisfy all constraints.)

标准型的线性规划包括：

- n 变量 $x_1, x_2, \ldots, x_n$。
  （为这个变量需要找到最优值。）
- 实值常数 $c_1, c_2, \ldots, c_n \in \mathbb{R}$ 组成线性的目标函数 $\Sigma\, c_j x_j$。（最优的值最大化目标函数。）
- （线性）约束，表达使用矩阵 $\mathbf{A} \in \mathbb{R}^{m\times n}$ 和矢量 $\boldsymbol{b} \in \mathbb{R}^m$: $\Sigma\, a_{ij} x_j \leq b_i$ for $i = 1, 2, \ldots, m$
- 非负约束 $x_j \geq 0$ for j = 1, 2, ..., n
  （$\boldsymbol{x}$ 的值需要满足所有的约束。）

Linear Programming Problem

线性规划问题

A linear program in standard form consists of:

标准型的线性规划包括：

- n variables $x_1, x_2, ..., x_n$.
  (We want to find optimal va

  $_n$。
  找到最优值。）

- real constants $c_1, c_2, ..., c_n$
  objective function $\Sigma\, c_j x_j$. (T
  maximizes the objective fu

  , $c_n \in \mathbb{R}$ 组成线性的目标函数
  直最大化目标函数。）

- (linear) constraints, expres
  and a vector $\boldsymbol{b} \in \mathbb{R}^m$: $\Sigma\, a_{ij} x_j$

  达使用矩阵 $\mathbf{A} \in \mathbb{R}^{m \times n}$
  $_{ij} x_j \le b_i$ for $i = 1, 2, ..., m$

- nonnegativity constraints x_j

  or j = 1, 2, ..., n
  (The values of $\boldsymbol{x}$ must satisfy all constraints.)

  ($\boldsymbol{x}$ 的值需要满足所有的约束。）

In short, 短写,

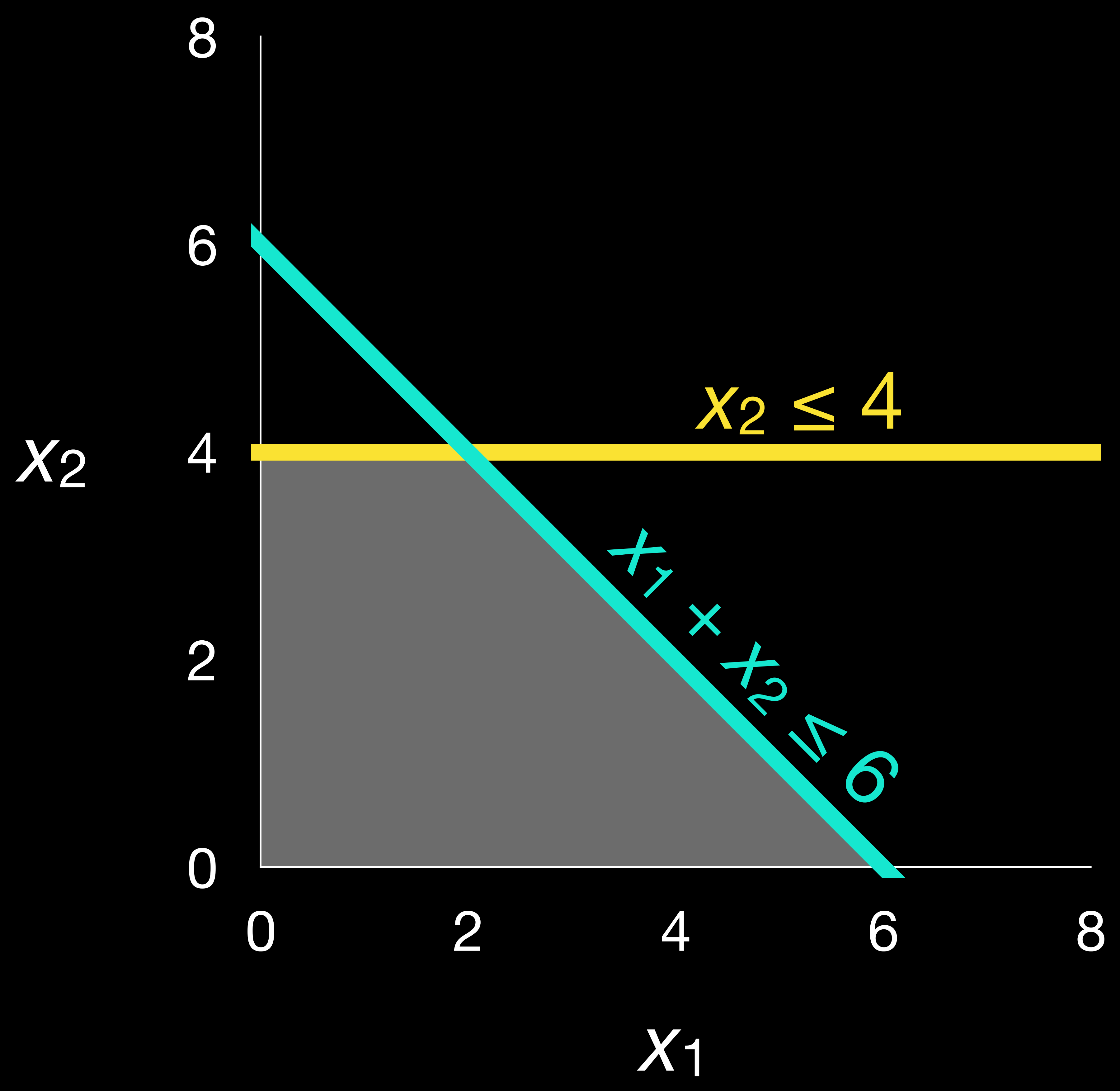
maximize 最大化 $\boldsymbol{c}^\mathsf{T}\boldsymbol{x}$

subject to 满足约束 $\mathbf{A}\boldsymbol{x} \le \boldsymbol{b}$

$\boldsymbol{x} \ge \mathbf{0}$

Example: Production Planning

例如：生产计划



maximize 最大化 $1250x_1 + 1750x_2$

subject to 满足约束

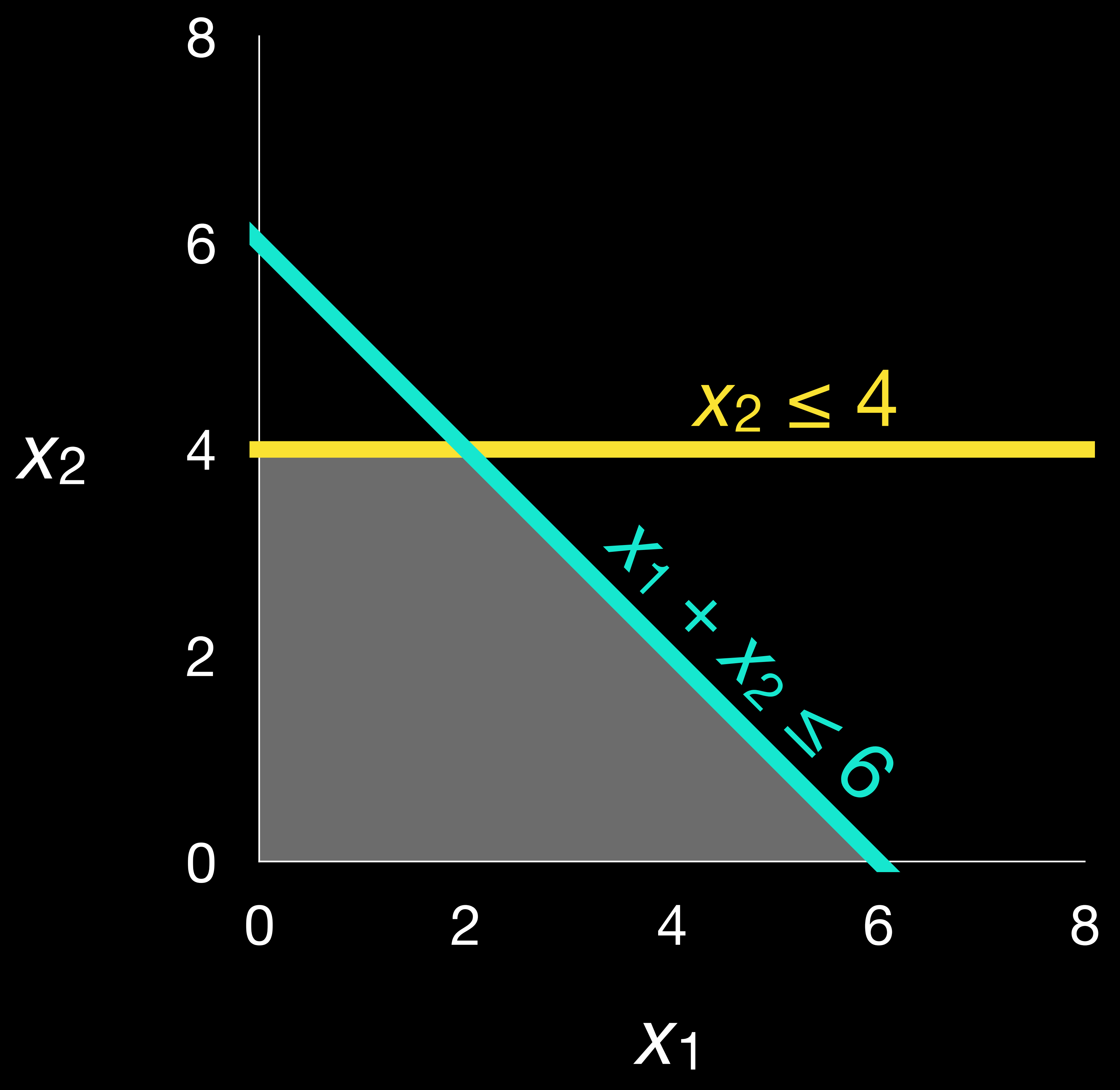
$$x_1 + x_2 \leq 6$$
$$x_2 \leq 4$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$

The Simplex Method　　单纯形算法

- often-used method to solve a linear program
- Underlying idea: One of the corners (vertices) must be optimal.
- Important step of the Simplex Method:
  Try to improve on the current corner by moving to an adjacent corner with (hopefully) higher value.
- simplex = triangle- or tetraeder-like shape in $[0,\infty)^n$ determined by one constraint

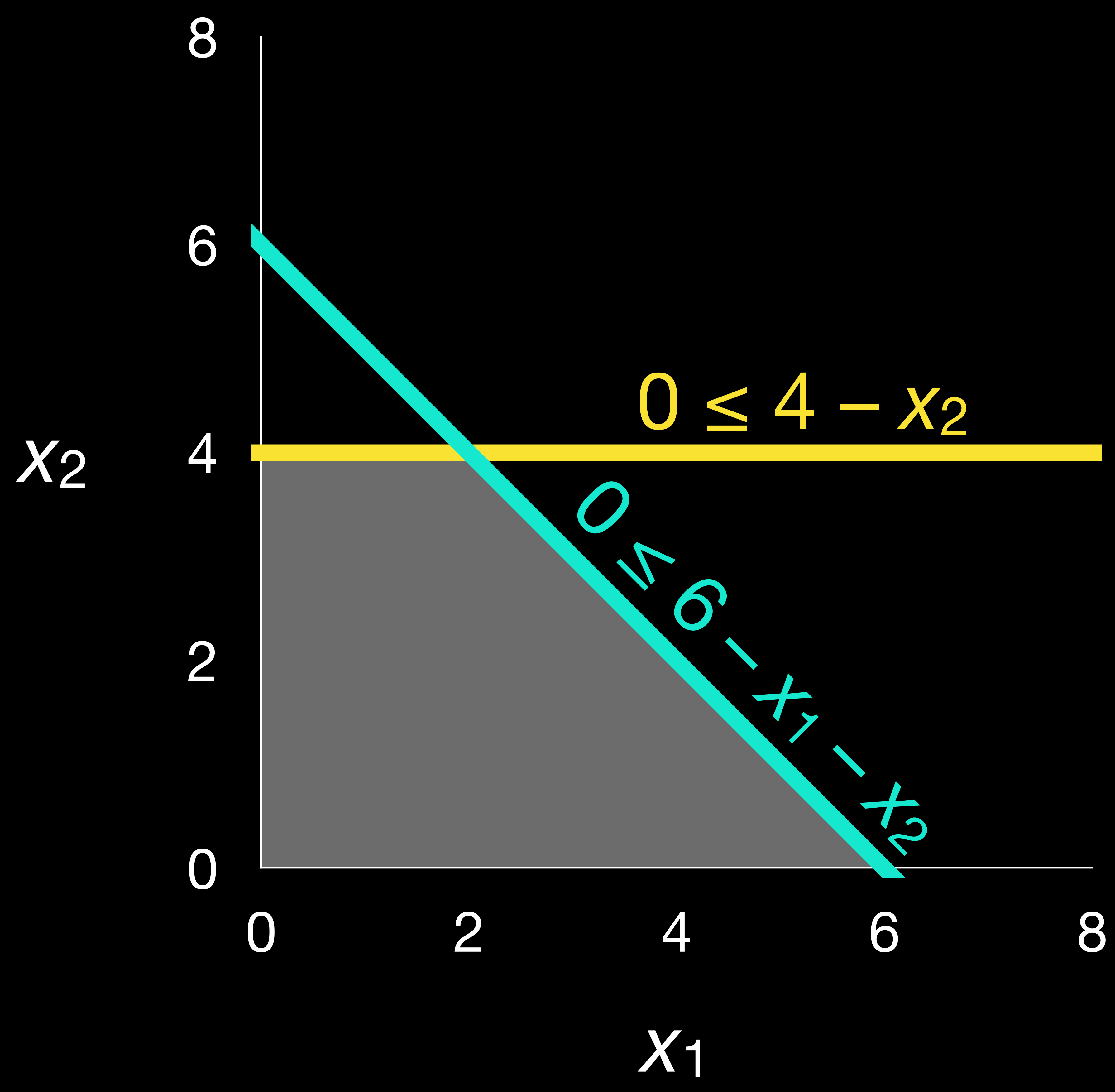
- 常用的算法解决线性规划问题
- 注意：一个角落（顶点）
  必须表达最优值。
- 单纯形法的重要步骤：
  尝试通过移动到具有（希望）更高值的相邻拐角来改进当前拐角。
- 单纯形 = 由一个约束确定的 $[0,\infty)^n$ 中的
  三角形或者四面体形

Example: Slack Form　例如：松弛型



$$
\begin{array}{ll}
\text{maximize} & 1250x_1 + 1750x_2 \\
\text{subject to} & x_1 + x_2 \le 6 \\
 & x_2 \le 4 \\
 & x_1 \ge 0 \\
 & x_2 \ge 0
\end{array}
$$

Example: Slack Form　　例如：松弛型

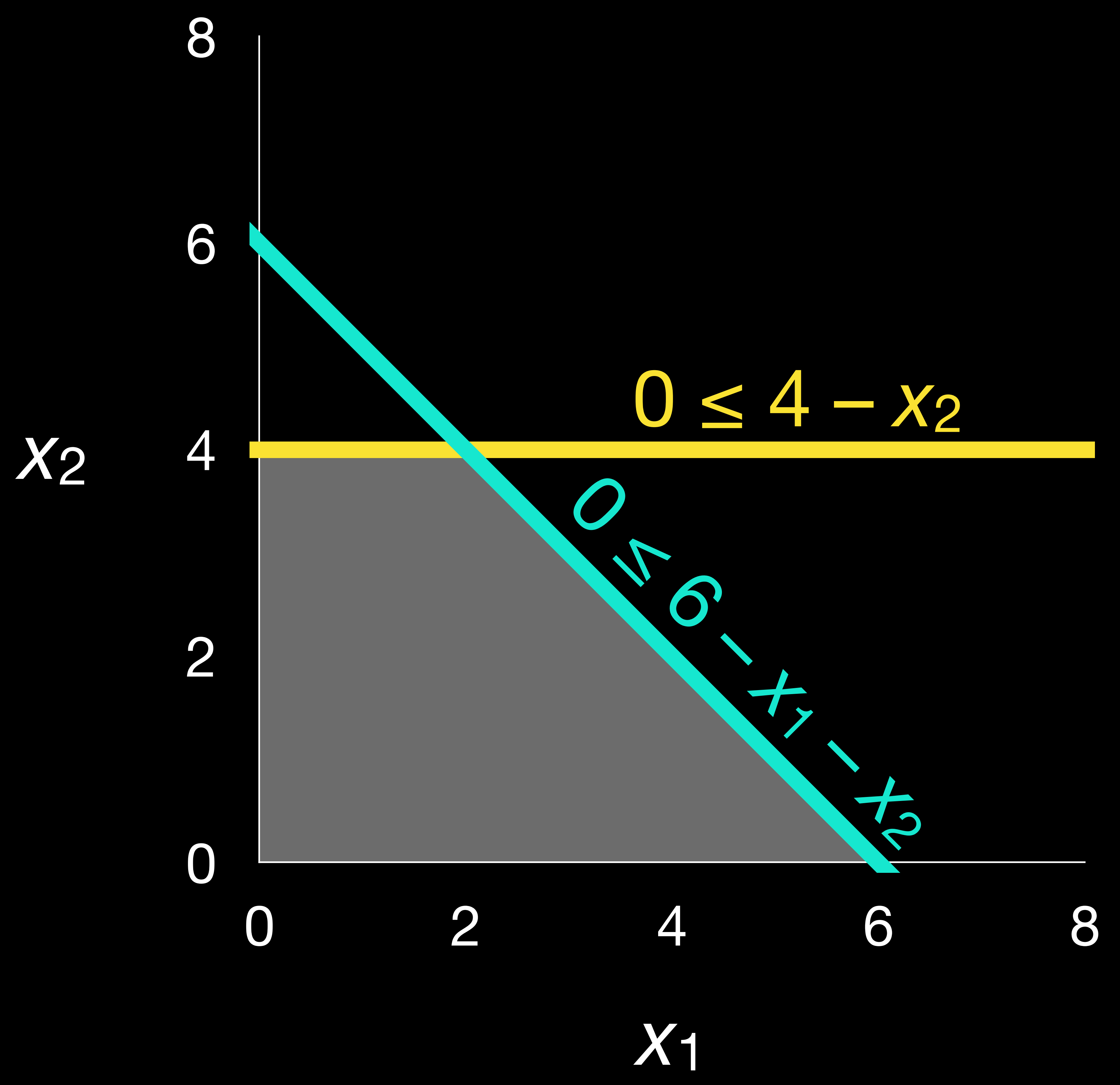


maximize $1250x_1 + 1750x_2$

subject to

$$0 \le 6 - x_1 - x_2$$
$$0 \le 4 - x_2$$
$$x_1 \ge 0$$
$$x_2 \ge 0$$

Example: Slack Form 例如：松弛型



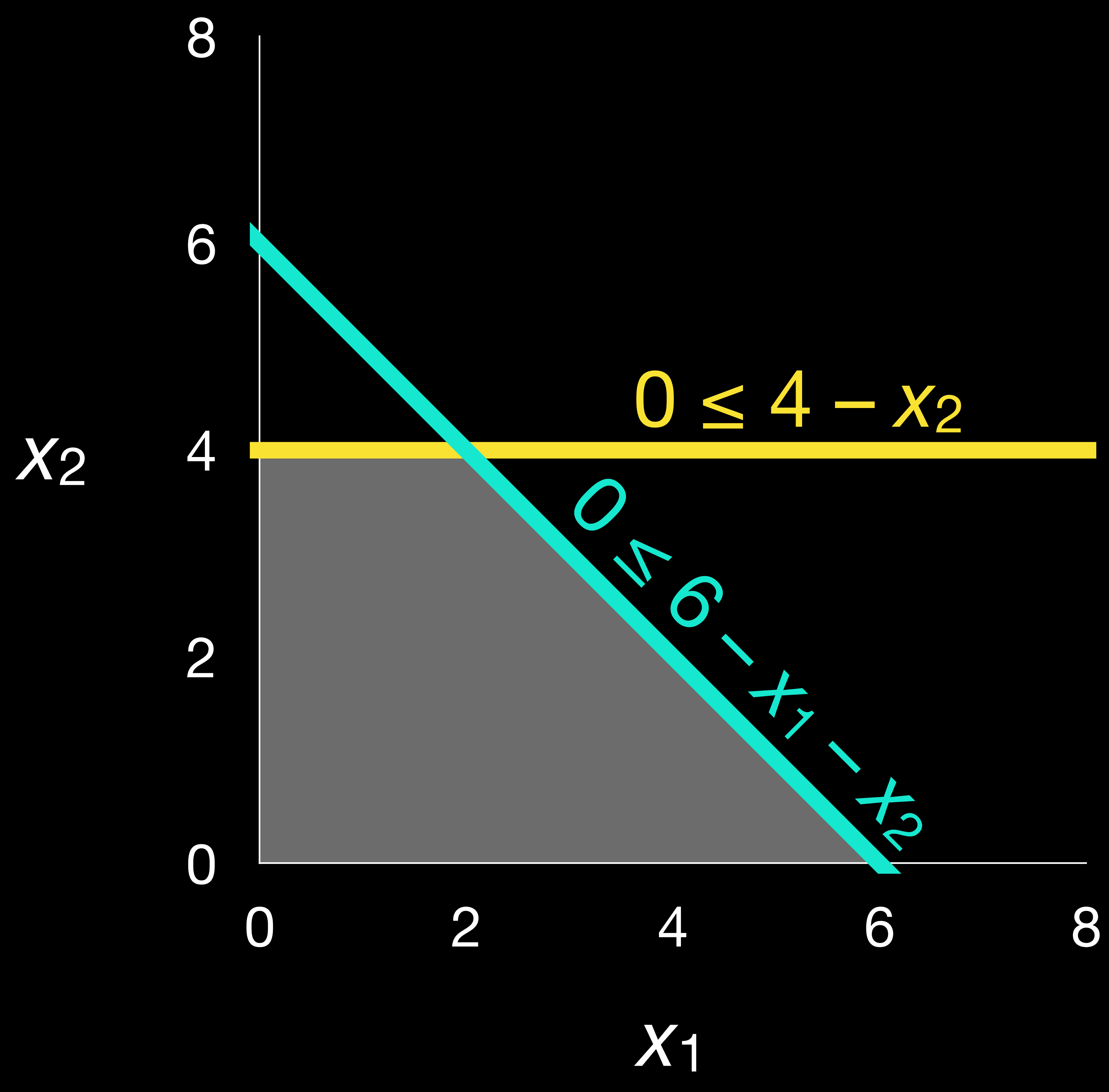
maximize $1250x_1 + 1750x_2$

subject to

$$x_3 = 6 - x_1 - x_2$$
$$x_4 = 4 - x_2$$
$$x_1 \ge 0$$
$$x_2 \ge 0$$
$$x_3 \ge 0$$
$$x_4 \ge 0$$

slack variables
松弛变量

Example: Slack Form 例如：松弛型

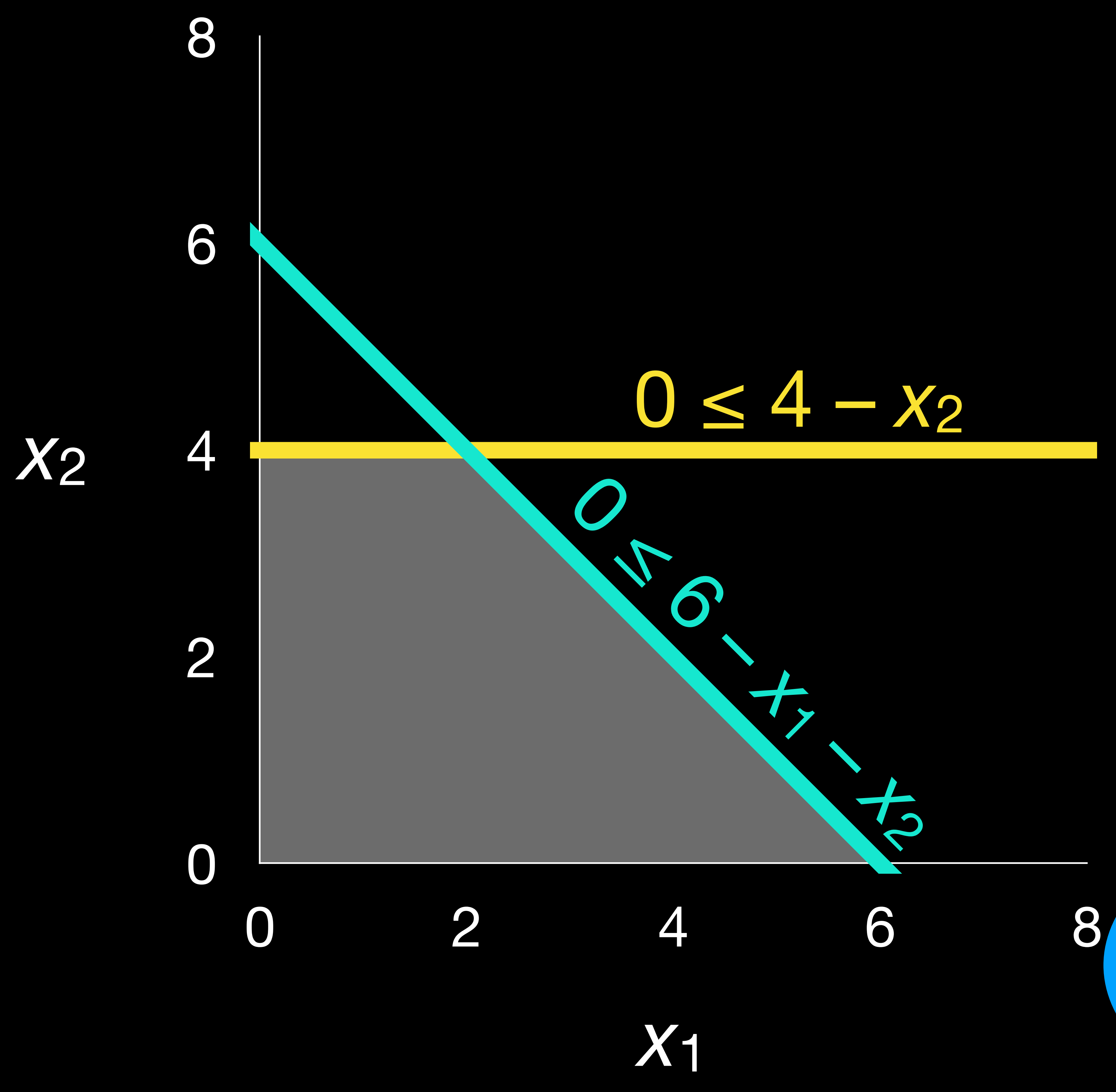


maximize $z = 1250x_1 + 1750x_2$

subject to

$$
\begin{aligned}
x_3 &= 6 - x_1 - x_2 \\
x_4 &= 4 - \phantom{x_1 - {}} x_2 \\
x_1 &\geq 0 \\
x_2 &\geq 0 \\
x_3 &\geq 0 \\
x_4 &\geq 0
\end{aligned}
$$

Example: Slack Form 例如：松弛型



Slack Form 松弛型

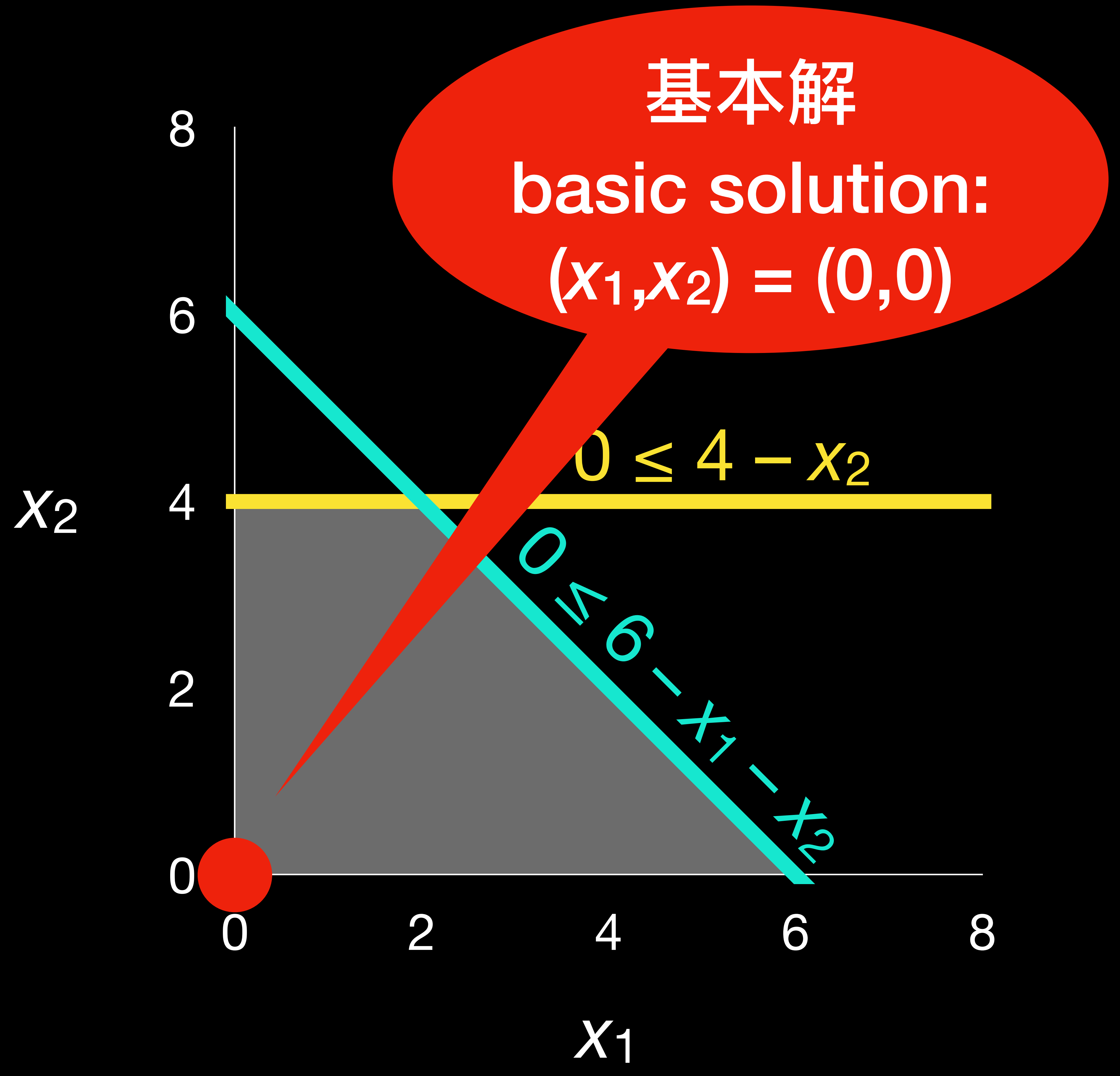
$$z = 1250x_1 + 1750x_2$$

$$\begin{aligned} x_3 &= 6 - x_1 - x_2 \\ x_4 &= 4 - \phantom{x_1 - {}} x_2 \end{aligned}$$

basic variables
基本变量
$B = \{x_3, x_4\}$

nonbasic variables
非基本变量
$N = \{x_1, x_2\}$

Example: Run Simplex 例如：进行单纯想算法



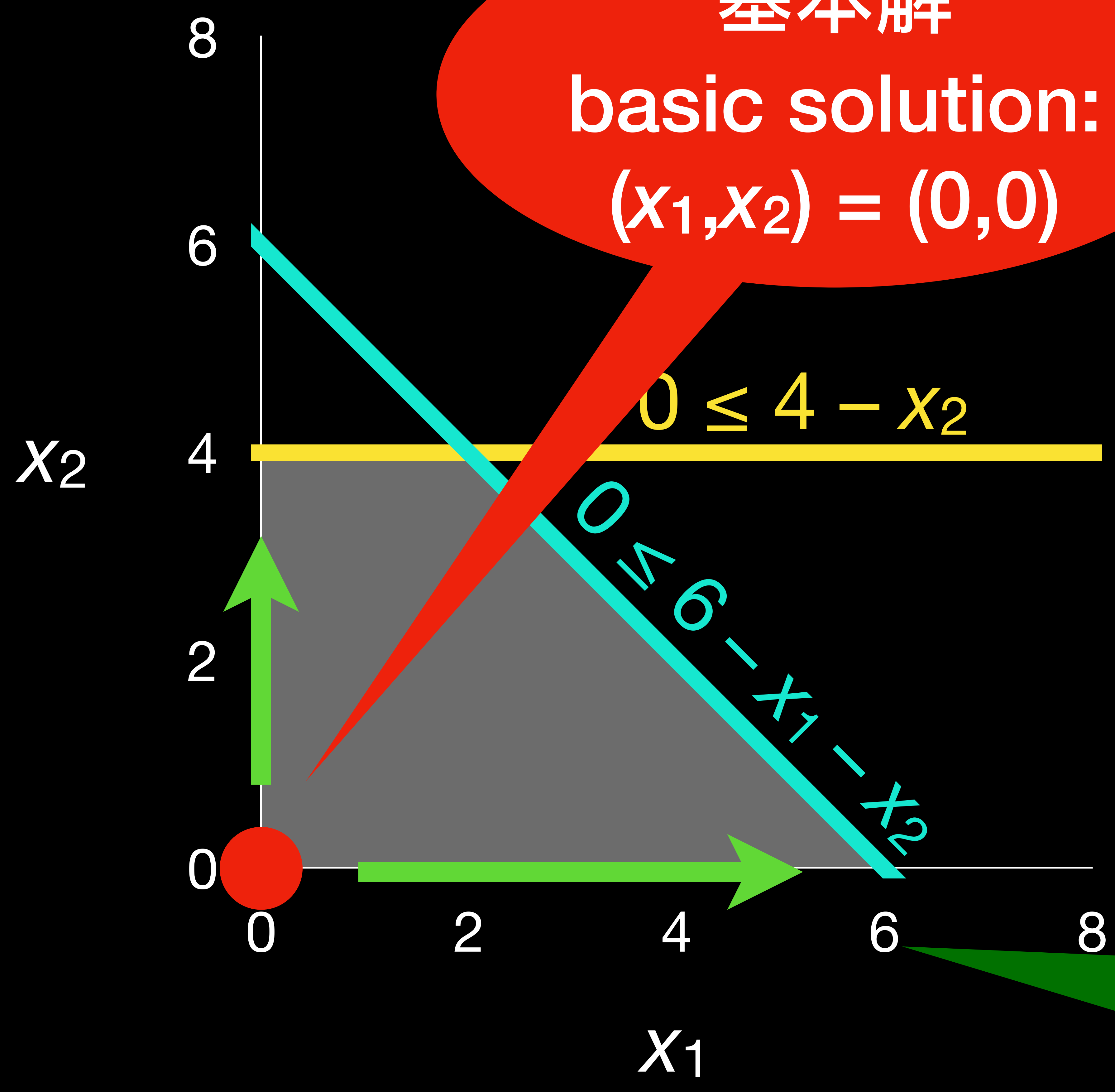
The solution is optimal if all constants are ≤ 0.
如果所有的常数≤ 0，则解决最优。

$$z = 1250x_1 + 1750x_2$$

$$x_3 = 6 - x_1 - x_2$$
$$x_4 = 4 - \phantom{x_1 - {}} x_2$$

Example: Run Simplex 例如：进行单纯性算法

基本解
basic solution:
$(x_1,x_2) = (0,0)$



largest constant = most promising

最大的常数 = 有最大的前途

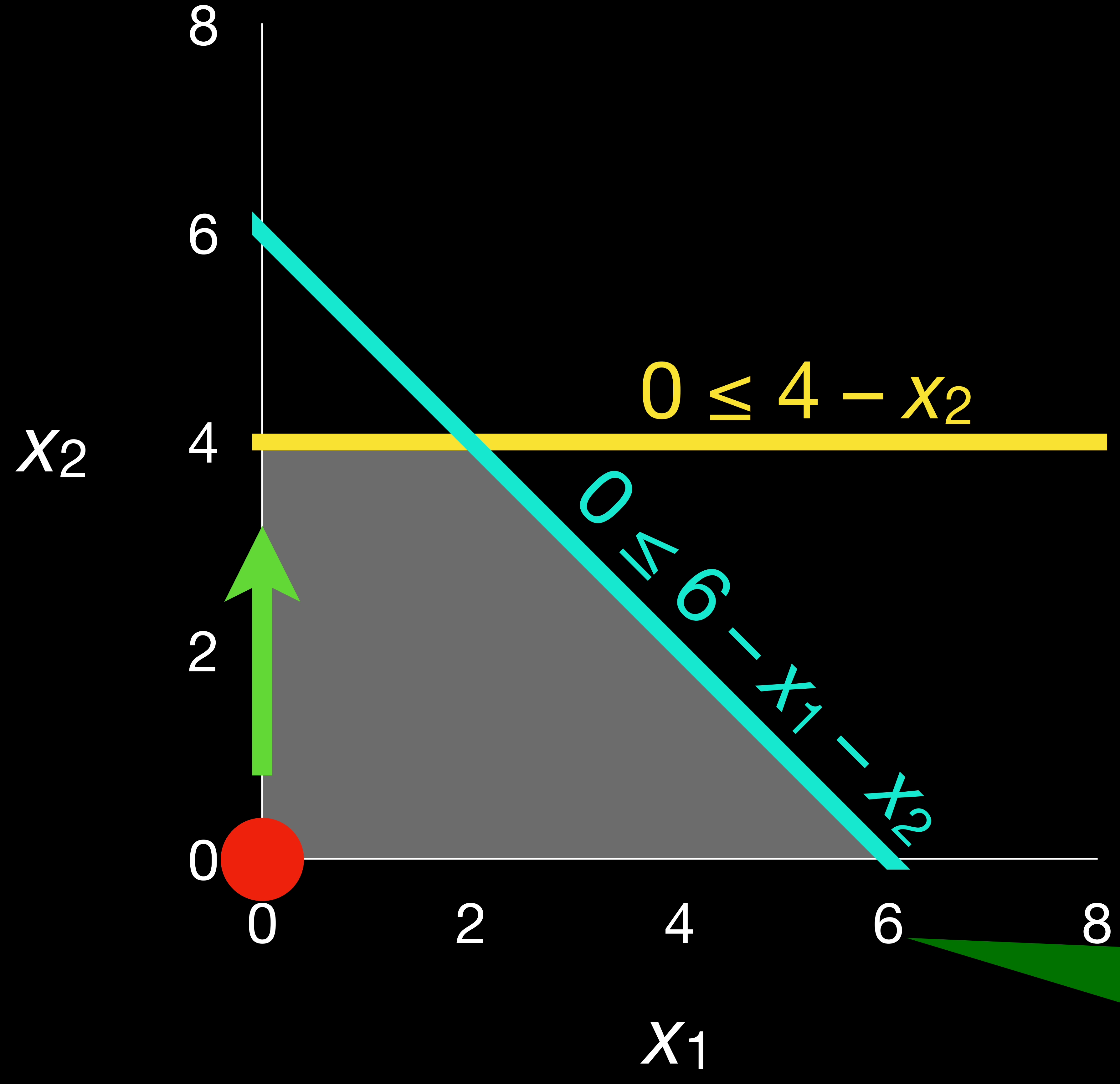
$$z = 1250x_1 + 1750x_2$$

$$x_3 = 6 - x_1 - x_2$$
$$x_4 = 4 - x_2$$

improve profit by pivoting = exchange a basic and a nonbasic variable

优化利润
使用转换操作 = 替换基本变量和非基本变量

Example: Run Simplex 例如：进行单纯性算法



largest constant = most promising

最大的常数 = 有最大的前途

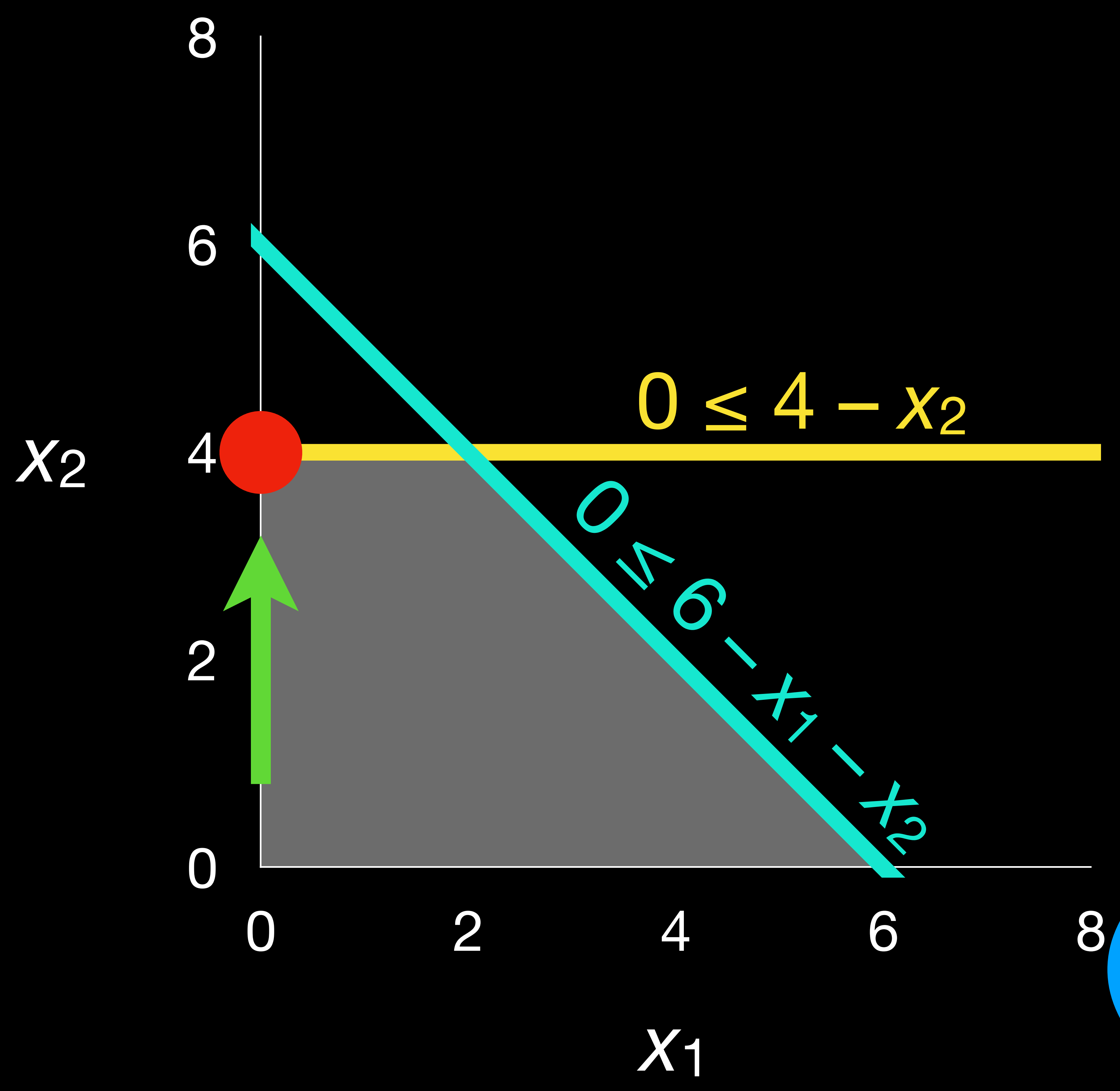
$$z = 1250x_1 + 1750x_2$$

$$x_2 = 6 - x_1 - x_3$$
$$x_2 = 4 - x_4$$

improve profit by pivoting = exchange a basic and a nonbasic variable

优化利润使用转换操作 = 替换基本变量和非基本变量

Example: Run Simplex　例如：进行单纯性算法



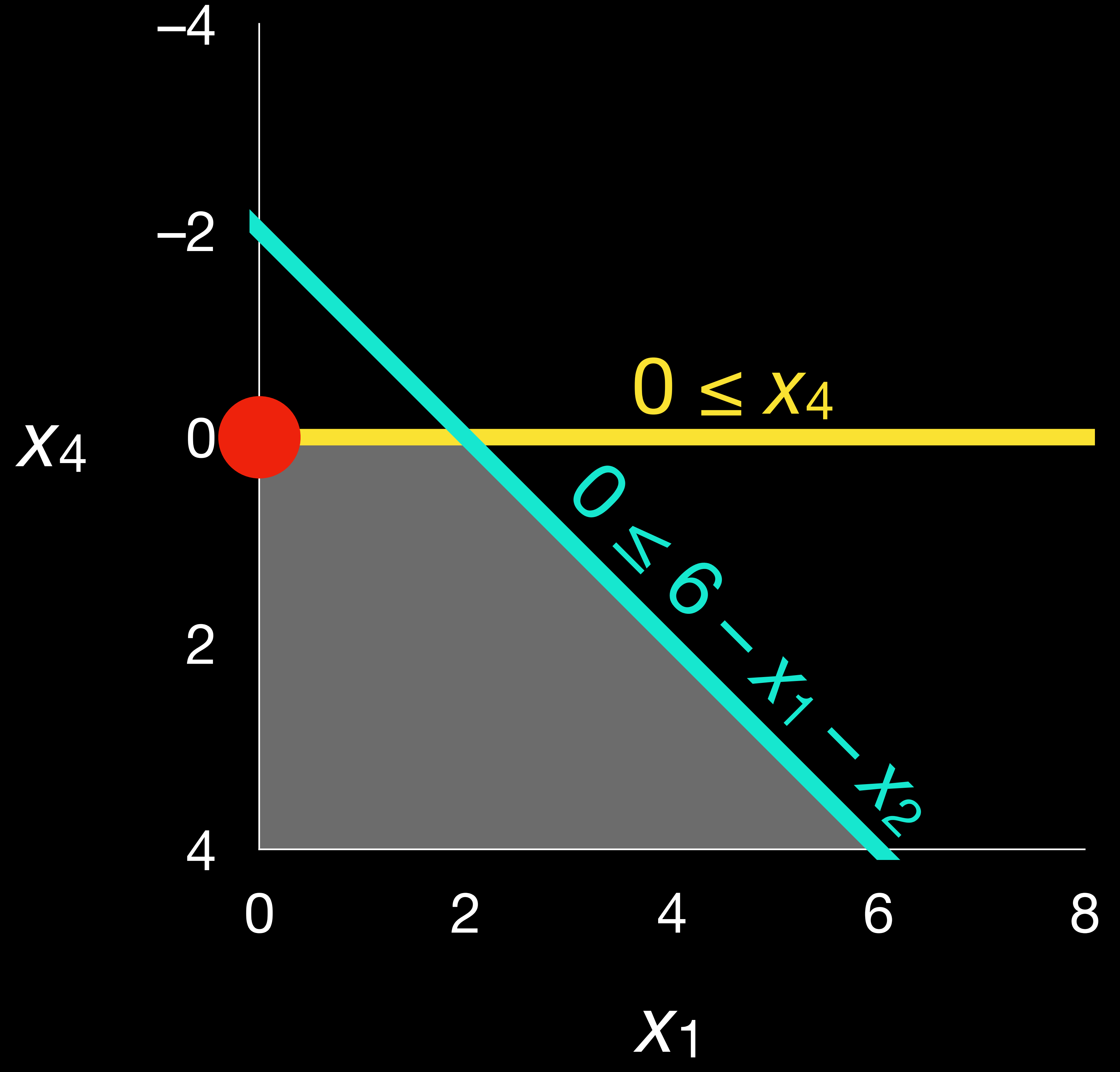
$$z = 1250x_1 + 1750(4 - x_4)$$

$$x_3 = 6 - x_1 - (4 - x_4)$$
$$x_2 = 4 - \qquad x_4$$

basic variables
基本变量
$B = \{x_3, x_2\}$

nonbasic variables
非基本变量
$N = \{x_1, x_4\}$

Example: Run Simplex 例如：进行单纯性算法



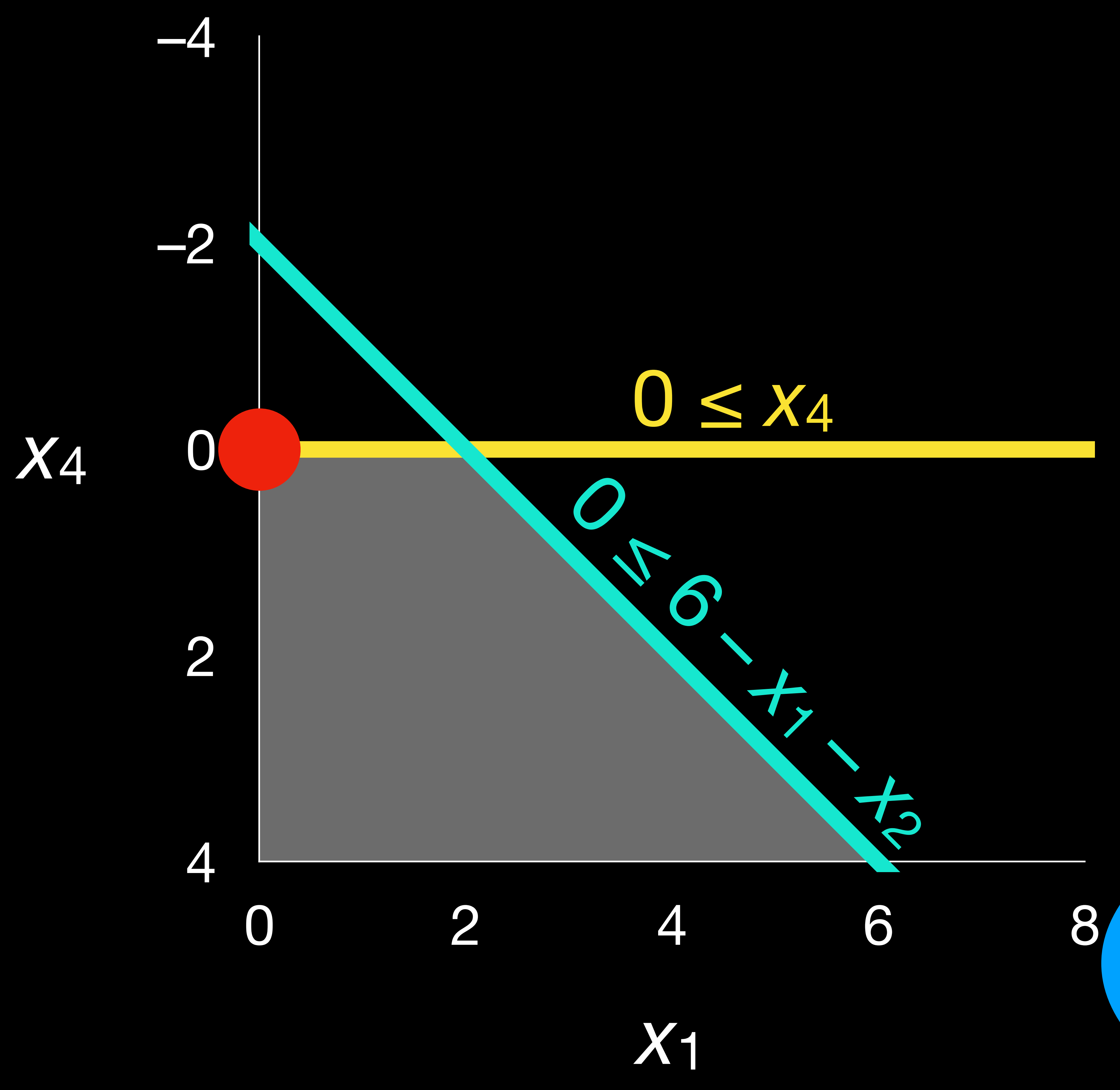
largest constant = most promising

最大的常数 = 有最大的前途

$$z = 7000 + 1250x_1 - 1750x_4$$

$$x_3 = 2 - x_1 + x_4$$
$$x_2 = 4 - \qquad x_4$$

Example: Run Simplex 例如：进行单纯性算法



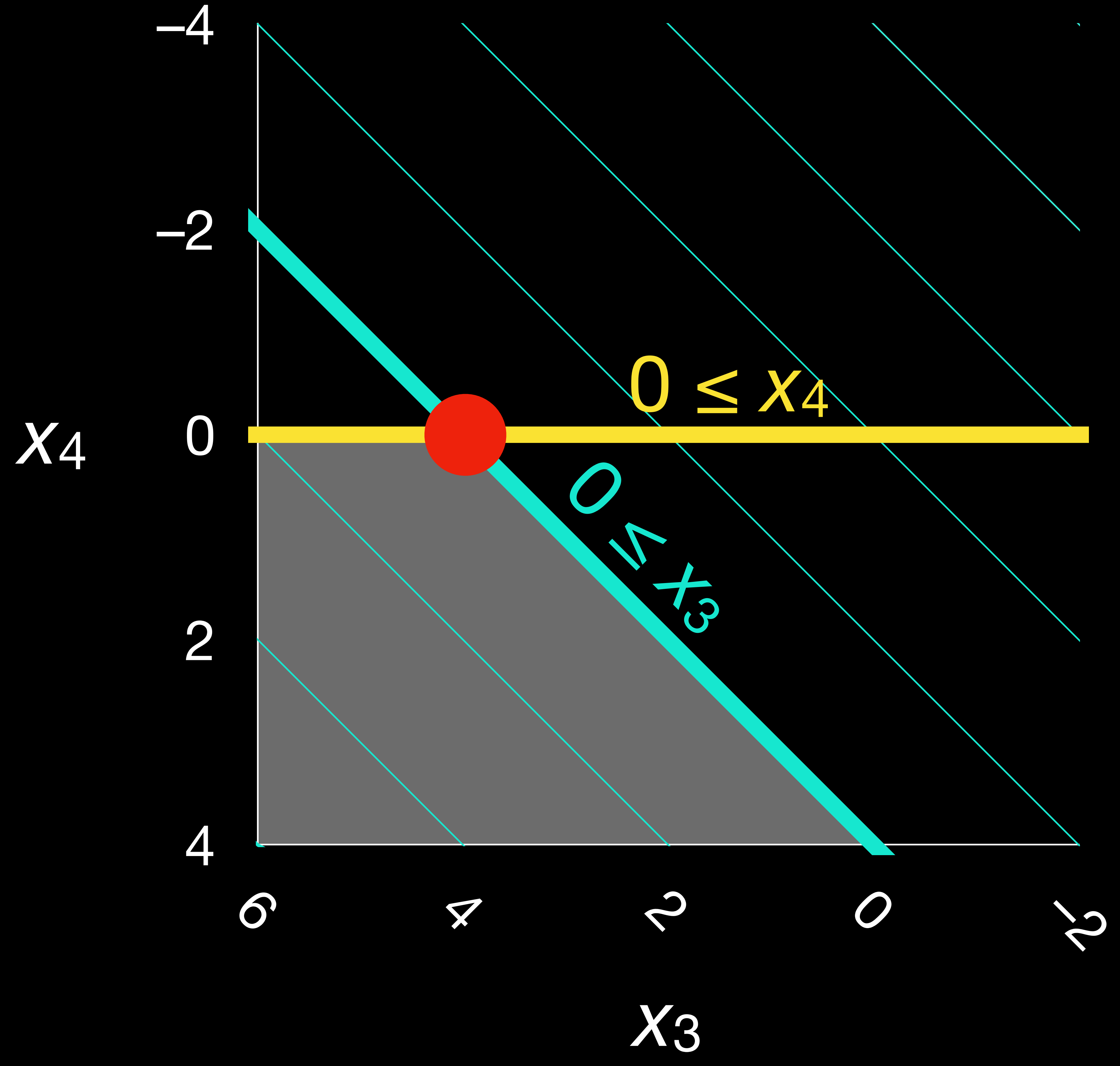
$$z = 9500 - 1250x_3 - 500x_4$$

$$x_1 = 2 - x_3 + x_4$$
$$x_2 = 4 - \phantom{x_3 +{}} x_4$$

basic variables
基本变量
$B = \{x_1, x_2\}$

nonbasic variables
非基本变量
$N = \{x_3, x_4\}$

Example



The solution is optimal if all constants are ≤ 0.
如果所有的常数≤ 0，则解决最优。

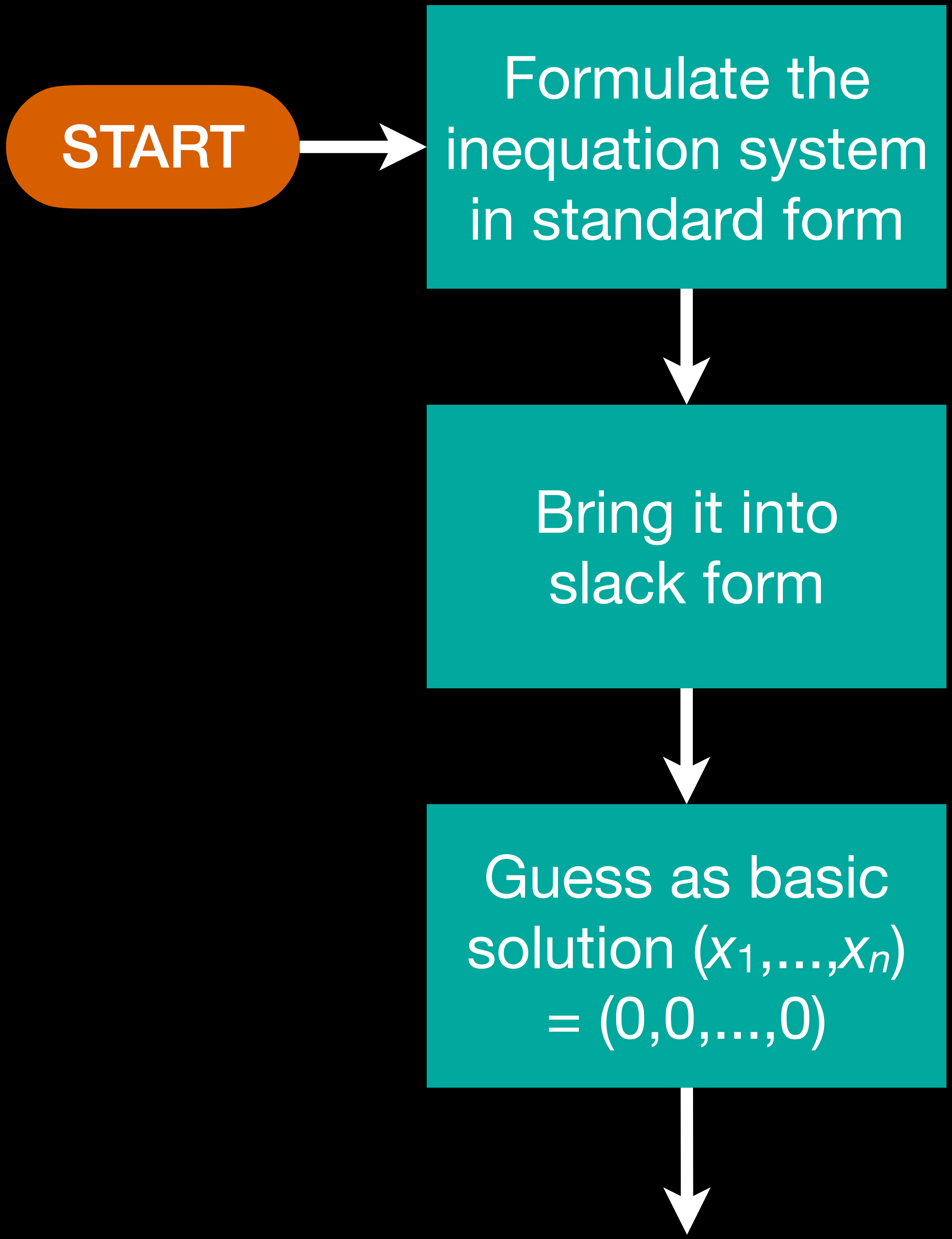
$$z = 9500 - 1250x_3 - 500x_4$$

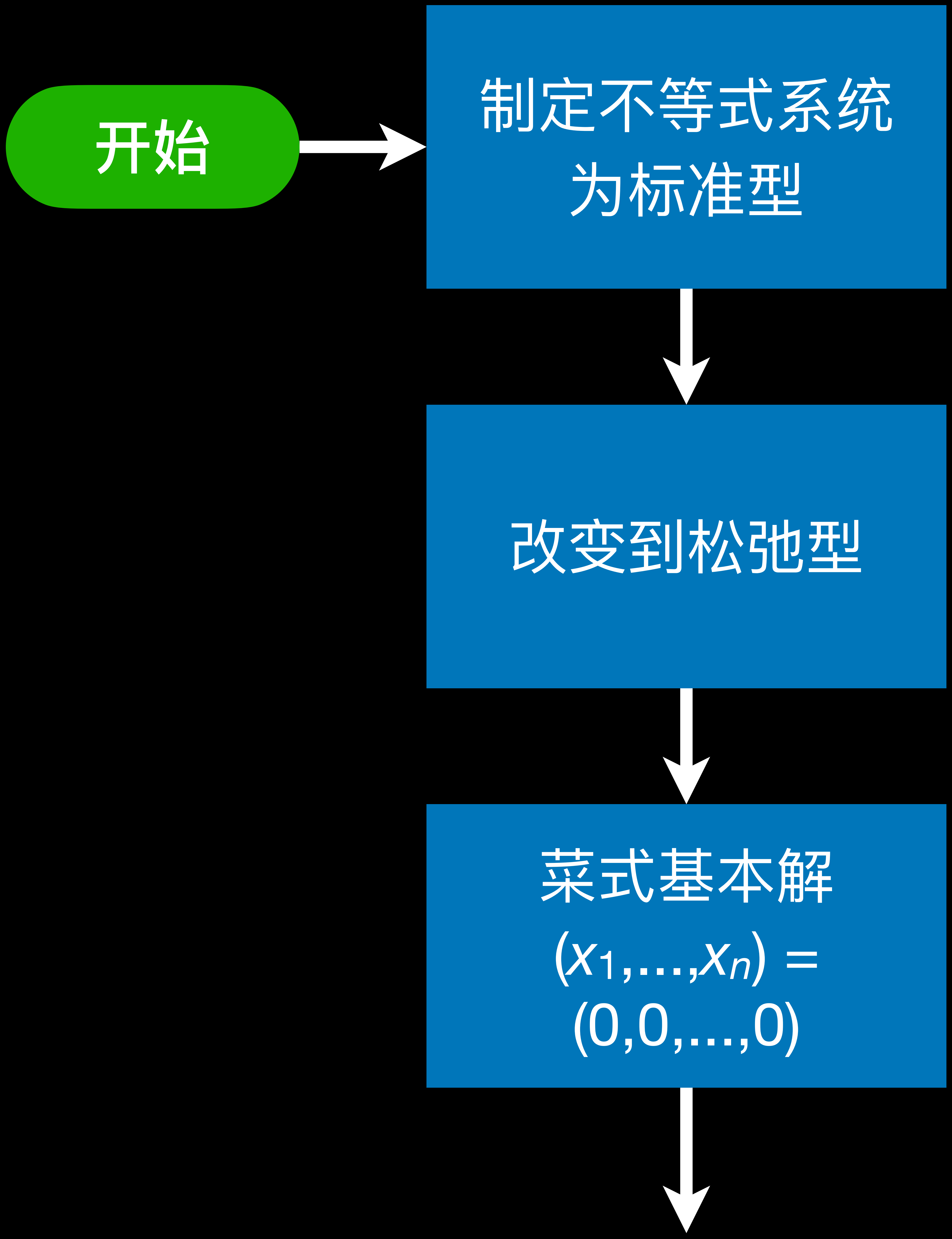
$$x_1 = 2 - x_3 + x_4$$
$$x_2 = 4 - \phantom{x_3 +{}} x_4$$

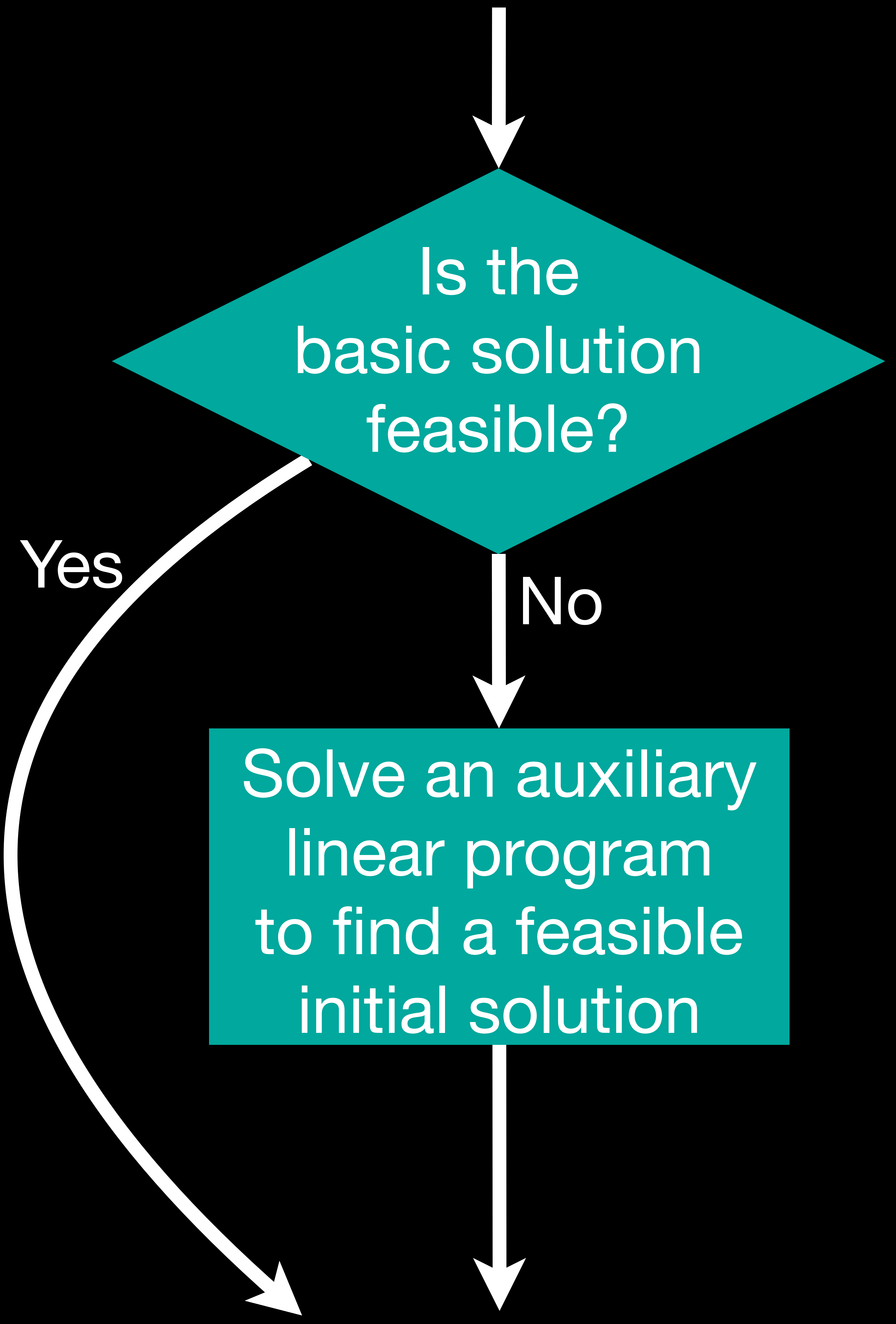
Optimal solution 最优解：

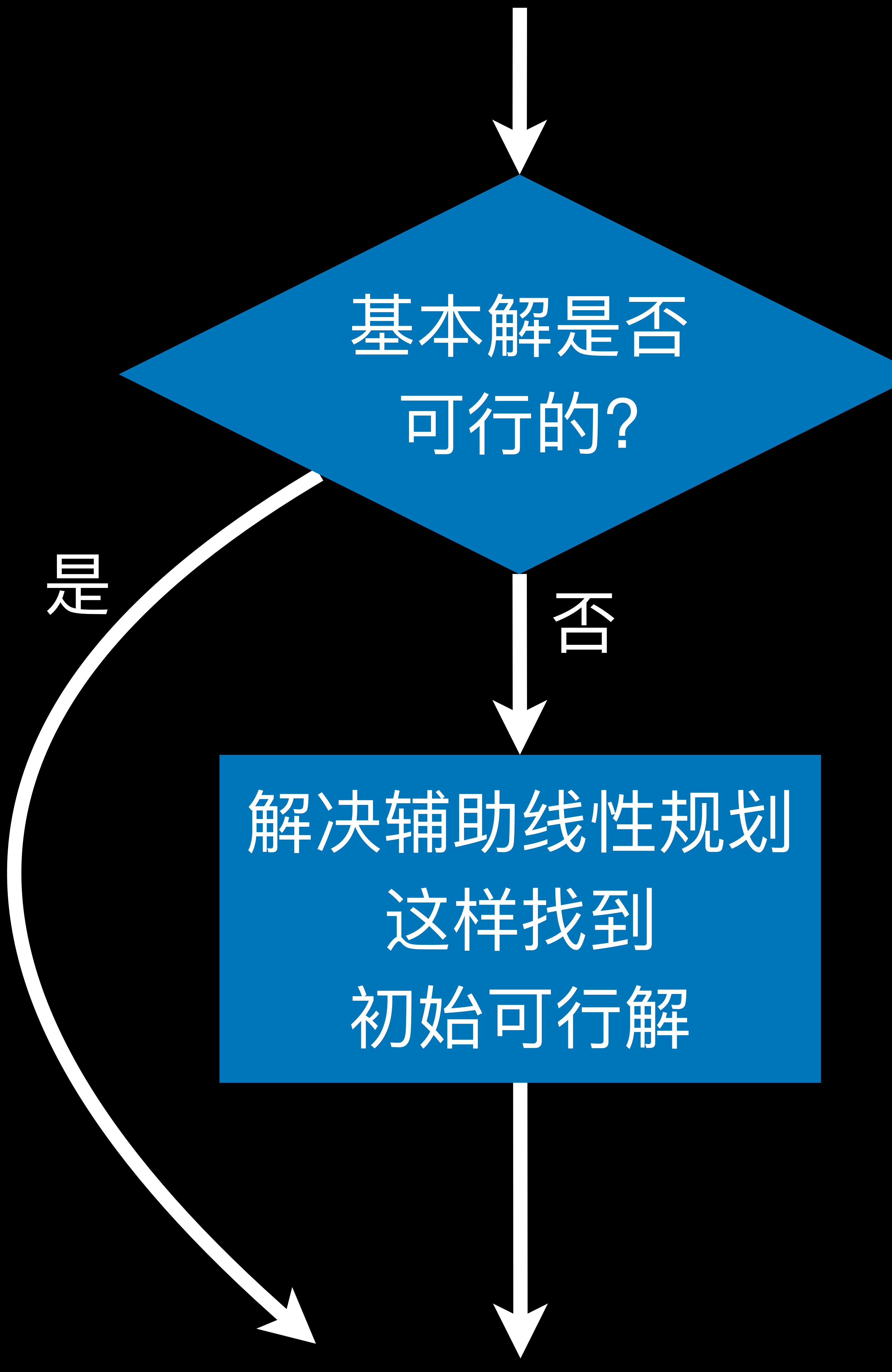
nonbasic variables 非基本变量 $(x_3, x_4) = 0$

➥ $x_1 = 2$, $x_2 = 4$


START
Formulate the inequation system in standard form
Bring it into slack form
Guess as basic solution $(x_1,\ldots,x_n) = (0,0,\ldots,0)$


开始
制定不等式系统为标准型
改变到松弛型
菜式基本解
$(x_1,\ldots,x_n) = (0,0,\ldots,0)$


Is the basic solution feasible?
Yes
No
Solve an auxiliary linear program to find a feasible initial solution


基本解是否可行的?
是
否
解决辅助线性规划这样找到初始可行解

Choose the entering variable x_e
Yes
Is there a nonbasic variable x_j with $c_j > 0$?
Choose a leaving variable x_ℓ that minimizes $b_\ell/a_{\ell e}$
No
Pivot: make x_e basic and make x_ℓ nonbasic
Read off the solution with nonbasic variables = 0
STOP
选择替入变量 x_e
是
存在非基本变量 x_j 这样 $c_j > 0$?
选择替出变量 x_ℓ 这样 $b_\ell/a_{\ell e}$ 最小
否
转换：使 x_e 成为基本的与使 x_ℓ 成为非基本的
读解决所有非基本变量 = 0
停止

Shortest Paths as Linear Program

最短路径也是一种线性规划

Given a weighted graph (G = (V, E), w). Find the shortest paths from a fixed source vertex $s \in V$ to every vertex $t \in V$.

给定一个权重的图(G = (V, E), w)。求从固定源结点 $s \in V$ 到每个结点 $t \in V$ 的最短路径。

d_t = distance from s to t.

d_t = 从 s 到 t 的距离。

maximize 最大化 $\sum_{v \in V} d_v$

subject to 满足约束

$d_v \leq d_u + w(u,v)$ for every $(u,v) \in E$

$d_s \leq 0$

$d_v \geq 0$ for every $v \in V$

What remains to be done?

还要讨论的

- general form of slack form
- pseudocode for Pivot, Simplex
- find an initial feasible solution
  and correct other anomalies of the initial inequation system
- correctness
- efficiency

- 松弛型的通用模型
- Pivot，Simplex的伪代码
- 怎么样找到初始可行解
  与标准化初始不等式系统
- 正确性
- 效率

Slack Form 松弛型

- N = (indices of) nonbasic variables 非基本变量（的指数）$|N| = n$
- B = (indices of) basic variables 基本变量（的指数）$|B| = m$
- objective function 目标函数 $z = v + \sum_{j \in N} c_j x_j$
- constraints 约束 $x_i = b_i - \sum_{j \in N} a_{ij} x_j$ for $i \in B$
- (implicit constraints 含蓄的约束 $x_i \geq 0$ for $i \in B \cup N$)

$B \cup N = \{1, 2, ..., n+m\}$
$B \cap N = \varnothing$

For historical reasons, write – here.

Pivot

- basic calculation step
- transforms slack form into another, equivalent slack form with "better" basic solution
- parameters: (N,B,A,b,c,v) = slack form
  ℓ = index of leaving variable
  (i.e. ℓ leaves B)
  e = index of entering variable
  (i.e. e enters B)
- result: $(\hat{N},\hat{B},\hat{A},\hat{b},\hat{c},\hat{v})$ = new slack form

转换

- 基本的计算步骤
- 转变从一个松弛型到另外一个等价的有更优的基本解的松弛型
- 参数：(N,B,A,b,c,v) = 松弛型
  ℓ = 替出变量的指数
  (ℓ 出 B)
  e = 替入变量的指数
  (e 入 B)
- 回复：$(\hat{N},\hat{B},\hat{A},\hat{b},\hat{c},\hat{v})$ = 新的松弛型

Pivot Pseudocode   Pivot的伪代码

create constraint $x_e = \hat{b}_e - \Sigma\, \hat{a}_{ej} x_j$

eliminate x_e from constraint i

eliminate x_e from objective function

$\text{PIVOT}(N,B,A,b,c,v,\ell,e)$

$\hat{b}_e = b_\ell / a_{\ell e}$

for each $j \in N \setminus \{e\}$ $\hat{a}_{ej} = a_{\ell j} / a_{\ell e}$

$\hat{a}_{e\ell} = 1/a_{\ell e}$

for each $i \in B \setminus \{\ell\}$

  $\hat{b}_i = b_i - a_{ie} \hat{b}_e$

  for each $j \in N \setminus \{e\}$ $\hat{a}_{ij} = a_{ij} - a_{ie} \hat{a}_{ej}$

  $\hat{a}_{i\ell} = -a_{ie} / a_{\ell e}$

$\hat{v} = v + c_e \hat{b}_e$

for each $j \in N \setminus \{e\}$ $\hat{c}_j = c_j - c_e \hat{a}_{ej}$

$\hat{c}_\ell = -c_e / a_{\ell e}$

$\hat{N} = N \setminus \{e\} \cup \{\ell\}$

$\hat{B} = B \setminus \{\ell\} \cup \{e\}$

return $(\hat{N},\hat{B},\hat{A},\hat{b},\hat{c},\hat{v})$

创作约束 $x_e = \hat{b}_e - \Sigma\, \hat{a}_{ej} x_j$

消除 x_e 从约束 i

消除 x_e 从目标函数

Pivot Pseudocode Pivot的伪代码

create constraint $x_e = \hat{b}_e - \Sigma\, \hat{a}_{ej} x_j$

eliminate x_e from constraint i

eliminate x_e from objective function

PIVOT(N,B,A,b,c,v,ℓ,e)

$\hat{b}_e = b_\ell /$

for

$\hat{a}_e$

fo

$e\hat{a}_{ej}$

$\hat{v} =$

for

$\hat{c}_\ell = -c_e$

$\hat{N} = N \setminus \{e\} \cup \{\ell\}$

$\hat{B} = B \setminus \{\ell\} \cup \{e\}$

return ($\hat{N}$,$\hat{B}$,$\hat{A}$,$\hat{b}$,$\hat{c}$,$\hat{v}$)

Transfer information from variables that disappear to variables that appear

将信息从消失的变量转移到出现的变量

创作约束 $x_e = \hat{b}_e - \Sigma\, \hat{a}_{ej} x_j$

消除 x_e 从约束 i

消除 x_e 从目标函数

Pivot

转换

PIVOT transforms a slack form into an equivalent slack form (i.e. a slack form with the same feasible region).

Proof: Let $(\bar{x}_1, ..., \bar{x}_{n+m})$ be a feasible solution of the old slack form. We have to prove:

$$\begin{aligned} x_e &= \hat{b}_e - \sum_{j\in\hat{N}} \hat{a}_{ej}x_j \\ &= b_\ell/a_{\ell e} - \sum_{j\in N\setminus\{e\}} a_{\ell j}/a_{\ell e}\, x_j - 1/a_{\ell e}\, x_\ell \\ &= (b_\ell - \sum_{j\in N\setminus\{e\}} a_{\ell j}\, x_j - x_\ell)/a_{\ell e}. \end{aligned}$$

That is equivalent to $a_{\ell e}\, x_e = b_\ell - \sum_{j\in N\setminus\{e\}} a_{\ell j}\, x_j - x_\ell$.

That is equivalent to the old constraint for x_ℓ.

PIVOT将松弛形转换为等效松弛形（即具有相同可行区域的松弛形）。

证明：设 $(\bar{x}_1, ..., \bar{x}_{n+m})$ 是一个原来的松弛型的可行解。需要证明：

$\hat{N} = N \setminus \{e\} \cup \{\ell\}$

$$\begin{aligned} x_e &= \hat{b}_e - \sum_{j\in\hat{N}} \hat{a}_{ej}x_j \\ &= b_\ell/a_{\ell e} - \sum_{j\in N\setminus\{e\}} a_{\ell j}/a_{\ell e}\, x_j - 1/a_{\ell e}\, x_\ell \\ &= (b_\ell - \sum_{j\in N\setminus\{e\}} a_{\ell j}\, x_j - x_\ell)/a_{\ell e}. \end{aligned}$$

这相当于 $a_{\ell e}\, x_e = b_\ell - \sum_{j\in N\setminus\{e\}} a_{\ell j}\, x_j - x_\ell$.

这相当于原来的 x_ℓ 的约束。

Pivot

转换

PIVOT transforms a slack form into an equivalent slack form (i.e. a slack form with the same feasible region).

Proof: Let $(\bar{x}_1, \ldots, \bar{x}_{n+m})$ be a feasible solution of the new slack form.
We have to prove that it satisfies the constraint of the old slack form.
This can be done by a similar calculation.

PIVOT将松弛形转换为等效松弛形（即具有相同可行区域的松弛形）。

证明：设 $(\bar{x}_1, \ldots, \bar{x}_{n+m})$ 是一个新的松弛型的可行解。
需要证明它满足原来松弛形的约束。

这可以通过类似的计算来完成。

Pivot

转换

PIVOT transforms a slack form into an equivalent slack form (i.e. a slack form with the same feasible region).

PIVOT将松弛形转换为等效松弛形（即具有相同可行区域的松弛形）。

Proof: Also, we have to prove that eliminating x_e does not essentially change other constraints or the objective function.

证明：此外，必须证明消除 x_e 本质上不会改变以外的约束或目标函数。

We only replace x_e by $\hat{b}_e - \sum_{j \in \hat{N}} \hat{a}_{ej} x_j$.

这是因为运算将 x_e 替换为 $\hat{b}_e - \sum_{j \in \hat{N}} \hat{a}_{ej} x_j$。

For example, $z = v + \sum_{j \in N} c_j x_j$

$$= v + \sum_{j \in N \setminus \{e\}} c_j x_j + c_e\Big(\hat{b}_e - \sum_{j \in \hat{N}} \hat{a}_{ej} x_j\Big).$$

例如，$z = v + \sum_{j \in N} c_j x_j$

$$= v + \sum_{j \in N \setminus \{e\}} c_j x_j + c_e\Big(\hat{b}_e - \sum_{j \in \hat{N}} \hat{a}_{ej} x_j\Big).$$

Simplex Pseudocode Simplex的伪代码

find x_e

find x_ℓ

pivot (if possible)

calculate the optimal solution

SIMPLEX(A,b,c)
(N,B,A,b,c,v) = INITIALIZE-SIMPLEX(A,b,c)
while there exists $j \in N$ with $c_j > 0$
  choose an index $e \in N$ with $c_e > 0$
  for each $i \in B$
    if $a_{ie} > 0$ **then** $\Delta_i = b_i / a_{ie}$
    else $\Delta_i = \infty$
  choose an index $\ell \in B$ with $\Delta_\ell = \min_{i \in B} \Delta_i$
  if $\Delta_\ell == \infty$
    return "Optimal solution is unbounded."
  else (N,B,A,b,c,v) = PIVOT(N,B,A,b,c,v,ℓ,e)
for $i = 1$ **to** n
  if $i \in B$ **then** $\bar{x}_i = b_i$
  else $\bar{x}_i = 0$
return $(\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n)$

找到 x_e

找到 x_ℓ

转换（可能的话）

计算最优解

No deterioration

没有恶化

When PIVOT is called as part of SIMPLEX, the value of the basic solution does not decrease.

当PIVOT作为SIMPLEX的一部分被调用时，基本解的值不会减少。

Proof: In the basic solution,
all nonbasic variables are assigned 0.
So the value of the old basic solution is v,
the value of the new basic solution is $\hat{v}$.
It suffices to prove $v \le \hat{v} = v + c_e\hat{b}_e$
$= v + c_e b_\ell / a_{\ell e}$.

证明：在基本解中,
所有非基本变量都被赋值为0。
所以旧的基本解的值是 v,
而新的基本解是$\hat{v}$。
只要证明 $v \le \hat{v} = v + c_e\hat{b}_e$
$= v + c_e b_\ell / a_{\ell e}$。

No deterioration

没有恶化

When PIVOT is called as part of SIMPLEX, the value of the basic solution does not decrease.

Proof: ... It suffices to prove $0 \le c_e b_\ell / a_{\ell e}$.
We have $c_e > 0$, otherwise SIMPLEX would not choose this index for e.
We must have $b_\ell \ge 0$, otherwise the old basic solution is infeasible.
We also must have $a_{\ell e} > 0$, otherwise SIMPLEX would not choose this index for ℓ.

当PIVOT作为SIMPLEX的一部分被调用时，基本解的值不会减少。

证明：只要证明 $0 \le c_e b_\ell / a_{\ell e}$。
有 $c_e > 0$,
否则SIMPLEX不会为 e 选择这个索引。
必须有 $b_\ell \ge 0$,
否则旧的基本解是不可行的。
还必须有 $a_{\ell e} > 0$,
否则SIMPLEX不会为 ℓ 选择这个索引。

Correctness

正确性

Proof in multiple steps:

1. If the algorithm terminates,
   then the solution is feasible.
2. If the algorithm does not loop, then
   it terminates within ... PIVOT steps.
3. If the algorithm returns a solution,
   then it is optimal (uses duality).
4. INITIALIZE-SIMPLEX finds an initial
   feasible solution if one exists
   (uses optimality).

多步骤证明：

1. 如果算法终止，
   那么回复的解决是可行的。
2. 如果算法没有循环，
   那么它在 … 转换步骤内终止。
3. 如果算法回复一个解，
   则它是最优的（使用对偶性）。
4. INITIALIZE-SIMPLEX找到一个
   初始可行的解（如果存在）
   （使用最优性）。

Feasibility 可行性

Lemma 29.2:
Assume that the initial linear program has a feasible basic solution.

If SIMPLEX returns a solution, it is a feasible solution.

If SIMPLEX reports that the linear program is unbounded, then it is unbounded (and its optimal solution is unbounded).

引理29.2：
假设初始的线性规划有一个基本解可行的松弛型。

如果SIMPLEX回复一个解，则这个解是此线性规划的一个可行解。

如果SIMPLEX回复“无界”，则此线性规划是无界的（与它的最优解是无界的）。

Feasibility: Proof 可行性：证明

The proof of Lemma 29.2 uses the loop invariant:

1. The current slack form is equivalent to the slack form returned by INITIALIZE-SIMPLEX.
2. For each $i \in B$, we have $b_i \geq 0$.
3. The basic solution of the current slack form is feasible.

Every time the "**while** there exists $j \in N$ with $c_j > 0$" loop body begins, this invariant holds.

引理29.2的证明使用循环不变式：

1. 此松弛型等价于调用INITIALIZE-SIMPLEX回复的松弛型。
2. 对每个 $i \in B$，我们有 $b_i \geq 0$。
3. 此松弛型相关的基本解是可行的。

每次循环"**while** there exists $j \in N$ with $c_j > 0$"体开始时，这个不变式都成立。

Loop Invariant Proof 循环不变式证明

a) Initialisation: Need to prove: When the loop starts, the loop invariant holds.

b) Maintenance: Need to prove:
If the loop invariant (and the loop condition) hold at the beginning of the loop iteration, then the invariant holds at the end of the loop iteration.

c) Termination: Use the loop invariant (and the negation of the loop condition) to prove a property required after the loop.

a) 初始化：需要证明：
循环开始的时候，不变式成立。

b) 保持：需要证明：
如果循环不变式（和循环条件）
在循环迭代开始时成立，
那么不变式在循环迭代结束时成立。

c) 终止：使用循环不变式
（和循环条件的否定）
来证明任何循环后的要求。

Feasibility 可行性: Proof

a) 1. In the beginning the current slack form is exactly the initial slack form
⤷ equivalence is trivial.

a) 2. We assumed in the lemma
that the initial slack form has a feasible basic solution.

a) 3. Therefore, we must have that every $x_i \geq 0$.
In particular, for $i \in B$, we have $b_i = x_i \geq 0$.

Feasibility 可行性: Proof

b) 1. The transformation of PIVOT
replaces the equation system (= the slack form) by an equivalent one.

b) 2. Only PIVOT changes the values of b_i.
ℓ and e are chosen such that $a_{\ell e} > 0$
and $b_\ell / a_{\ell e} \le b_i / a_{ie}$ for all $i \in B$ with $a_{ie} > 0$.

$\hat{b}_e = b_\ell / a_{\ell e} \ge 0$ because $b_\ell \ge 0$ and $a_{\ell e} > 0$.

$$\hat{b}_i = b_i - a_{ie}\hat{b}_e = b_i - a_{ie}(b_\ell / a_{\ell e}) \ge \begin{cases} b_i - a_{ie}(b_i / a_{ie}) = 0 & \text{if } a_{ie} > 0 \\ b_i \ge 0 & \text{if } a_{ie} \le 0 \end{cases}$$

b) 3. Because the basic solution has $x_i = 0$ or $x_i = \hat{b}_i \ge 0$, it is feasible.

Feasibility 可行性: Proof

c) We need to prove that, after the loop terminates:
If SIMPLEX returns a solution, it is a feasible solution.
If SIMPLEX reports that the linear program is unbounded, its optimal solution is unbounded.

If SIMPLEX returns a solution, it is the current basic solution; this is feasible because of loop invariant 3.

If SIMPLEX reports that the linear program is unbounded, then the solution

$$x_e = \infty$$
$$x_i = b_i - a_{ie}\cdot\infty \qquad \text{if } i \in B \text{ (with } 0\cdot\infty = 0)$$
$$x_i = 0 \qquad \text{otherwise}$$

is optimal. (Note that $a_{ie} \le 0$, so $x_i \ge 0$ for all i.)

Termination

终止性

- Lemma 29.2 did not state
that the algorithm terminates;
it could run in an infinite loop.

- 引理29.2没有说明算法终止；

有可能它无限循环运行。

- Some PIVOT steps do not improve the value of the objective function,
they are degenerate.

- 一些PIVOT步骤没有提高目标函数的值，
它们是退化的。

- An infinite loop is possible
if the algorithm cycles through
degenerate PIVOT steps.

- 无限循环是可能的
如果算法循环通过退化的PIVOT步骤。

Example: Cycling 循环

maximize 最大化 $$2.3x_1 + 2.15x_2 - 13.55x_3 - 0.4x_4$$

subject to 满足约束
$$\begin{aligned} 0.4x_1 + 0.2\ x_2 - 1.4\ x_3 - 0.2x_4 &\le 0 \\ -7.8x_1 - 1.4\ x_2 + 7.8\ x_3 + 0.4x_4 &\le 0 \\ x_1 &\ge 0 \\ x_2 &\ge 0 \\ x_3 &\ge 0 \\ x_4 &\ge 0 \end{aligned}$$

J.A.J. Hall, K.I.M. McKinnon: **The simplest examples where the simplex method cycles and conditions where EXPAND fails to prevent cycling.** https://doi.org/10.1007/s10107-003-0488-1

Example: Cycling 循环

$$\begin{aligned}
\text{maximize 最大化} \quad & z = \boxed{2.3x_1} + 2.15x_2 - 13.55x_3 - 0.4x_4 \\
\text{subject to 满足约束} \quad & x_5 = \boxed{-0.4x_1} - 0.2\ x_2 + 1.4\ x_3 + 0.2x_4 \\
& x_6 = 7.8x_1 + 1.4\ x_2 - 7.8\ x_3 - 0.4x_4
\end{aligned}$$

$$\begin{aligned}
\text{maximize 最大化} \quad & z = x_2 - 5.5x_3 + 0.75x_4 - 5.75x_5 \\
\text{subject to 满足约束} \quad & x_1 = -0.5x_2 + 3.5x_3 + 0.5\ x_4 - 2.5\ x_5 \\
& x_6 = -2.5x_2 + 19.5x_3 + 3.5\ x_4 - 19.5\ x_5
\end{aligned}$$

Example: Cycling 循环

maximize 最大化 $z = \boxed{x_2} - 5.5x_3 + 0.75x_4 - 5.75x_5$

subject to 满足约束

$$\begin{aligned} x_1 &= -0.5x_2 + 3.5x_3 + 0.5\ x_4 - 2.5\ x_5 \\ x_6 &= \boxed{-2.5x_2} + 19.5x_3 + 3.5\ x_4 - 19.5\ x_5 \end{aligned}$$

maximize 最大化 $z = 2.3x_3 + 2.15x_4 - 13.55x_5 - 0.4x_6$

subject to 满足约束

$$\begin{aligned} x_1 &= -0.4x_3 - 0.2\ x_4 + 1.4\ x_5 + 0.2x_6 \\ x_2 &= 7.8x_3 + 1.4\ x_4 - 7.8\ x_5 - 0.4x_6 \end{aligned}$$

Example: Cycling 循环

maximize 最大化 $z = 2.3x_3 + 2.15x_4 - 13.55x_5 - 0.4x_6$

subject to 满足约束

$$\begin{aligned} x_1 &= -0.4x_3 - 0.2\ x_4 + 1.4\ x_5 + 0.2x_6 \\ x_2 &= 7.8x_3 + 1.4\ x_4 - 7.8\ x_5 - 0.4 \end{aligned}$$

The
current slack form
is a permutation of the
original one. It will cycle!

当前松弛形是原始松弛形
的置换。它循环!

maximize 最大化 $z = 2.3x_1 + 2.15x_2 - 13.55x_3 - 0.4x$

subject to 满足约束

$$\begin{aligned} x_5 &= -0.4x_1 - 0.2\ x_2 + 1.4\ x_3 + 0.2x_4 \\ x_6 &= 7.8x_1 + 1.4\ x_2 - 7.8\ x_3 - 0.4x_4 \end{aligned}$$

original
slack form

Termination

终止性

- Lemma 29.2 did not state that the algorithm terminates; it could run in an infinite loop.
- An infinite loop is possible if the algorithm cycles through degenerate PIVOT steps.
- Cycling can be avoided by always choosing the first possible variable. (i.e. if both x_i and x_j can be chosen, choose $x_{\min\{i,j\}}$.)

- 引理29.2没有说明算法终止；

  有可能它无限循环运行。
- 无限循环是可能的
  如果算法循环通过退化的PIVOT步骤。
- 可以避免循环：
  总是选择第一个可能的变量。
  （即，如果 x_i 和 x_j 都可以选择，
  则选择 $x_{\min\{i,j\}}$。）

Termination

终止性

Lemma 29.7: If the initial linear program has a feasible basic solution,
then SIMPLEX either reports that it is unbounded,
or it terminates with a feasible solution within at most $\binom{n+m}{m}$ iterations.

引理29.7：
如果初始的松弛型有可行的基本解，
则SIMPLEX要么报告线性规划是无界的，

要么在至多 $\binom{n+m}{m}$ 此循环内终止，
并回复一个可行解。

Termination: Proof 终止性：证明

- Proof idea: The set of basic variables determines uniquely the slack form.
  If there are $n+m$ variables,
  of which m are basic,
  there are $\binom{n+m}{m}$ possible choices of basic variables.

- 证明思想：基本变量的集合
  唯一地决定松弛形。
  如果存在 $n+m$ 个变量，
  其中 m 是基本的，
  基本变量有 $\binom{n+m}{m}$ 种可能的选择。

- Why does the set of basic variables uniquely determine the slack form?
  See Lemma 29.4.

- 为什么基本变量的集合
  唯一地决定松弛形?
  参见引理29.4。

Termination 终止性: Proof

Lemma 29.4: Let (A,b,c) be a linear program in standard form.
Given a set of basic variables B,
the corresponding slack form is uniquely determined.

- Proof: Given two equivalent slack forms with the same basic variables:

$$\begin{array}{rcl} z &=& v + c_1x_1 + \cdots + c_nx_n \\ x_{n+1} &=& b_{n+1} + a_{n+1,1}x_1 + \cdots + a_{n+1,n}x_n \\ \vdots && \vdots \\ x_{n+m} &=& b_{n+m} + a_{n+m,1}x_1 + \cdots + a_{n+m,n}x_n \end{array} \qquad \begin{array}{rcl} z &=& v' + c'_1x_1 + \cdots + c'_nx_n \\ x_{n+1} &=& b'_{n+1} + a'_{n+1,1}x_1 + \cdots + a'_{n+1,n}x_n \\ \vdots && \vdots \\ x_{n+m} &=& b'_{n+m} + a'_{n+m,1}x_1 + \cdots + a'_{n+m,n}x_n \end{array}$$

One can subtract the right and left equations for x_{n+i} (i = 1, ..., m) and get

$$0 = b_{n+i} - b'_{n+i} + (a_{n+1,1} - a'_{n+i,1})x_1 + \cdots + (a_{n+i,n} - a'_{n+i,n})x_n.$$

This holds for all values of x_1, x_2, ..., x_n.
Therefore $b_{n+i} = b'_{n+i}$ and $a_{n+i,j} = a'_{n+i,j}$ for all i and j.

Correctness

Proof in multiple steps:

1. If the algorithm terminates, then the solution is feasible.
2. ~~If the algorithm does not loop, then~~ it terminates within ... PIVOT steps.
3. If the algorithm returns a solution, then it is optimal (uses duality).
4. INITIALIZE-SIMPLEX finds an initial feasible solution if one exists (uses optimality).

正确性

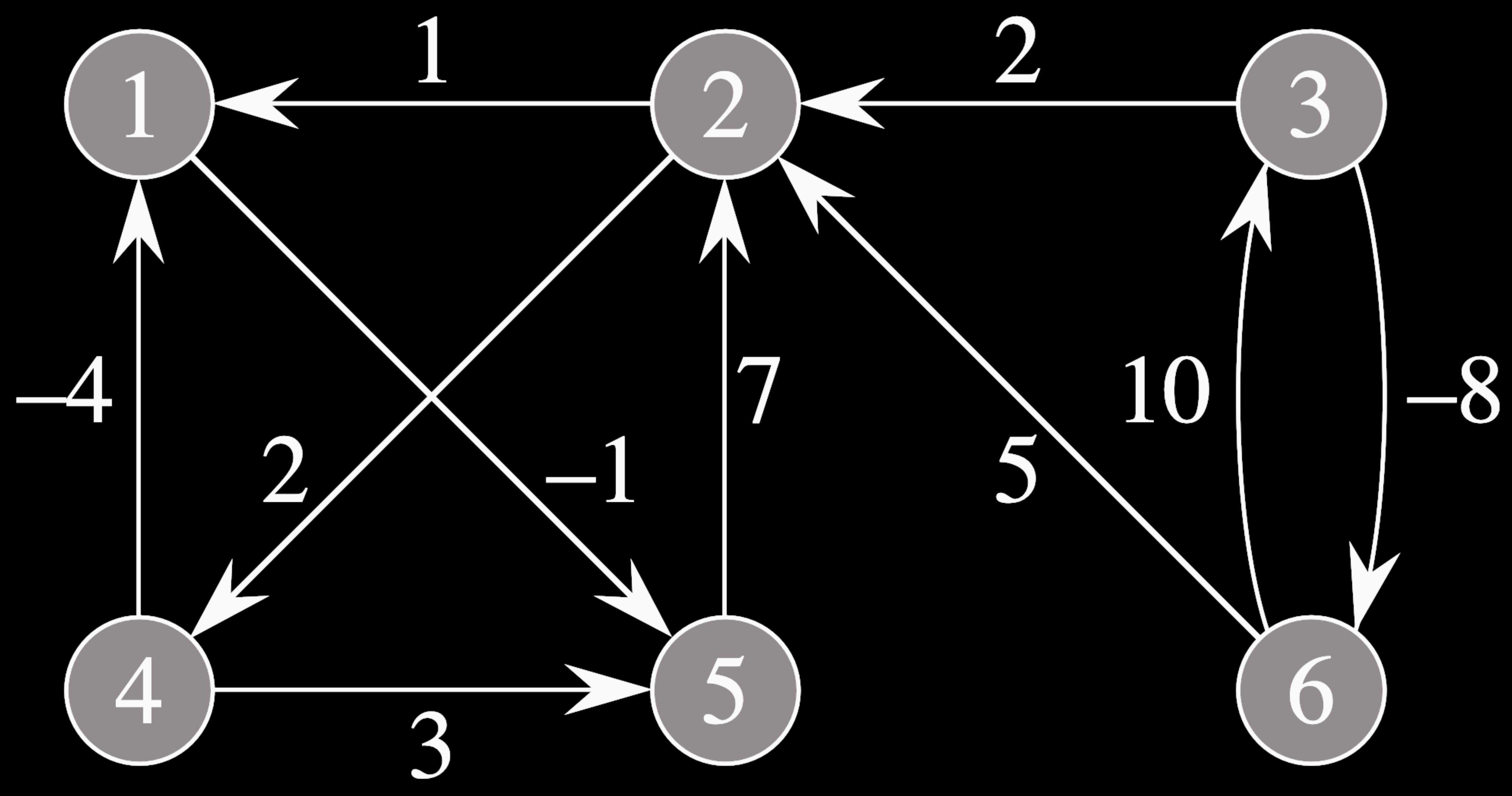
多步骤证明：

1. 如果算法终止，那么回复的解决是可行的。
2. ~~如果算法没有循环，~~那么它在 … 转换步骤内终止。
3. 如果算法回复一个解，则它是最优的（使用对偶性）。
4. INITIALIZE-SIMPLEX找到一个初始可行的解（如果存在）（使用最优性）。

25.2-1

Run the Floyd–Warshall algorithm on the weighted, directed graph of Figure 25.2. Show the matrix $D^{(k)}$ that results for each iteration of the outer loop.

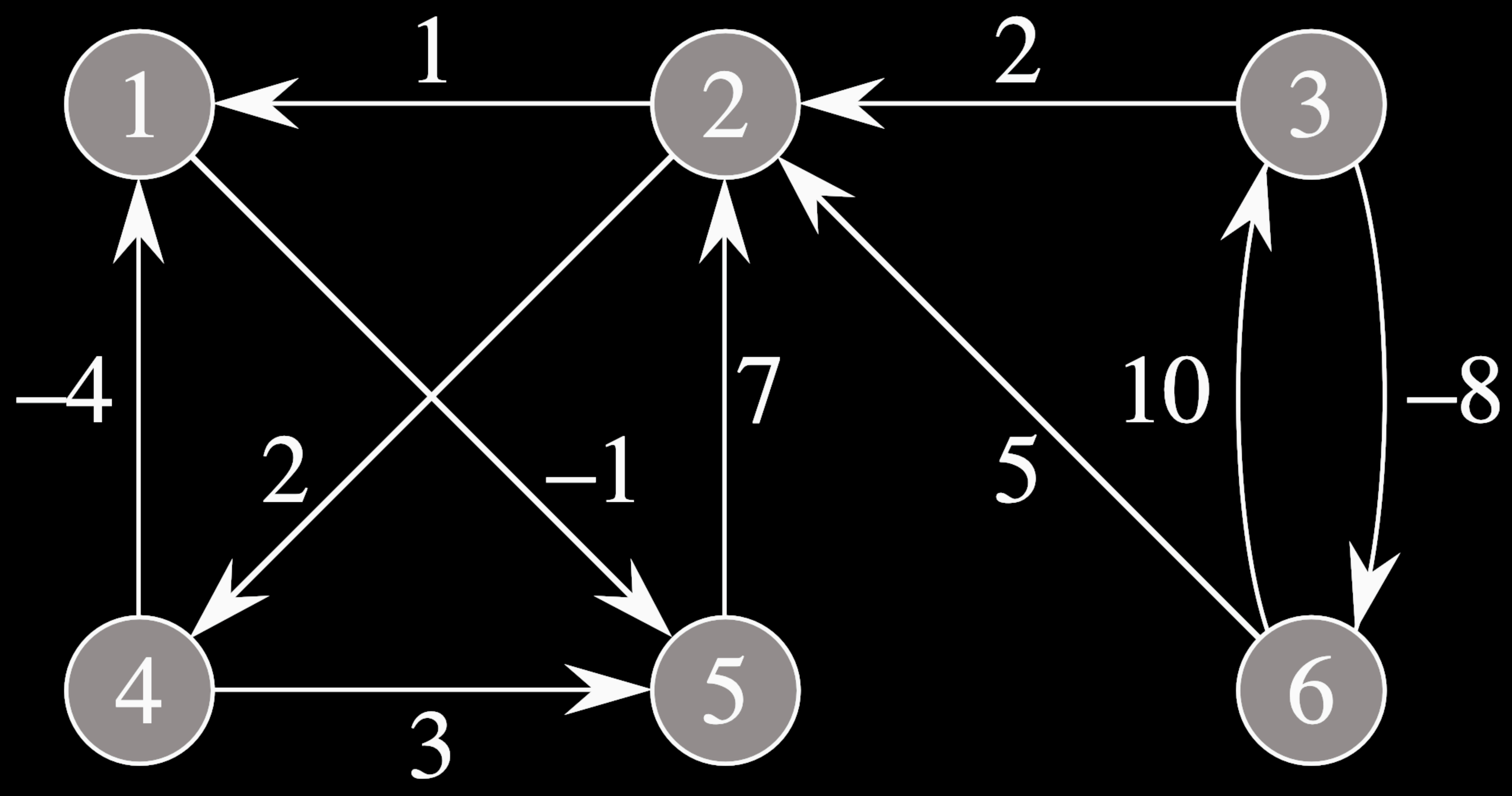
在图25-2所示的带权重的有向图上运行Floyd–Warshall算法，给出外层循环的每一次迭代所生成的矩阵 $D^{(k)}$。



25.3-1

Use Johnson’s algorithm to find the shortest paths between all pairs of vertices in the graph of Figure 25.2. Show the values of h and $\hat{w}$ computed by the algorithm.

请在图25-2上使用Johnson算法来找到所有结点对之间的最短路径。给出算法计算出的 h 和 $\hat{w}$ 值。



29.1-6

Show that the following linear program is infeasible:

说明下面线性规划是不可解的：

$$
\begin{array}{lrl}
\text{maximize 最大化} & 3x_1 - 2x_2 & \\
\text{subject to 满足约束} & x_1 + x_2 & \le 2 \\
 & -2x_1 - 2x_2 & \le -10 \\
 & x_1 & \ge 0 \\
 & x_2 & \ge 0
\end{array}
$$

29.3-6

Solve the following linear program using SIMPLEX:

采用SIMPLEX求解下面的线性规划：

maximize 最大化 $5x_1 - 3x_2$

subject to 满足约束

$$
\begin{aligned}
x_1 - x_2 &\le 1 \\
2x_1 + x_2 &\le 2 \\
x_1 &\ge 0 \\
x_2 &\ge 0
\end{aligned}
$$