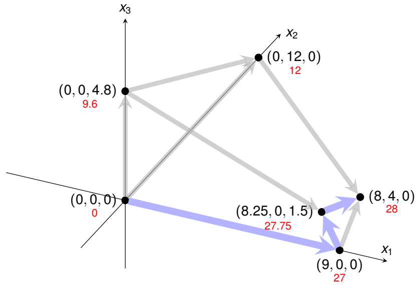


Randomised Algorithms

Lecture 7: Linear Programming: Simplex Algorithm

Thomas Sauerwald (tms41@cam.ac.uk)

Lent 2025

UNIVERSITY OF CAMBRIDGE

Outline

Simplex Algorithm: Introduction

Simplex Algorithm

- classical method for solving linear programs (Dantzig, 1947)
- usually fast in practice although worst-case runtime not polynomial
- iterative procedure somewhat similar to Gaussian elimination

Basic Idea:

- Each iteration corresponds to a "basic solution" of the slack form
- All non-basic variables are 0, and the basic variables are determined from the equality constraints
- Each iteration converts one slack form into an equivalent one while the objective value will not decrease
- Conversion ("pivoting") is achieved by switching the roles of one basic and one non-basic variable

In that sense, it is a greedy algorithm.

Extended Example: Conversion into Slack Form

$$\begin{array}{llllllll} \text{maximise} & 3x_1 & + & x_2 & + & 2x_3 & & \\ \text{subject to} & & & & & & & \\ & x_1 & + & x_2 & + & 3x_3 & \leq & 30 \\ & 2x_1 & + & 2x_2 & + & 5x_3 & \leq & 24 \\ & 4x_1 & + & x_2 & + & 2x_3 & \leq & 36 \\ & & & x_1, x_2, x_3 & & & \geq & 0 \end{array}$$

Conversion into slack form

$$\begin{array}{rcrcrcrcr} z & = & & & 3x_1 & + & x_2 & + & 2x_3 \\ x_4 & = & 30 & - & x_1 & - & x_2 & - & 3x_3 \\ x_5 & = & 24 & - & 2x_1 & - & 2x_2 & - & 5x_3 \\ x_6 & = & 36 & - & 4x_1 & - & x_2 & - & 2x_3 \end{array}$$

Extended Example: Iteration 1

$$\begin{array}{rcrcrcrcr} z & = & & & 3x_1 & + & x_2 & + & 2x_3 \\ x_4 & = & 30 & - & x_1 & - & x_2 & - & 3x_3 \\ x_5 & = & 24 & - & 2x_1 & - & 2x_2 & - & 5x_3 \\ x_6 & = & 36 & - & 4x_1 & - & x_2 & - & 2x_3 \end{array}$$

Basic solution: $(\overline{x_1}, \overline{x_2}, \ldots, \overline{x_6}) = (0, 0, 0, 30, 24, 36)$

This basic solution is **feasible**

Objective value is 0.

Extended Example: Iteration 1

Increasing the value of x_1 would increase the objective value.

$$\begin{array}{rcrcrcrcr} z & = & & & 3x_1 & + & x_2 & + & 2x_3 \\ x_4 & = & 30 & - & x_1 & - & x_2 & - & 3x_3 \\ x_5 & = & 24 & - & 2x_1 & - & 2x_2 & - & 5x_3 \\ x_6 & = & 36 & - & 4x_1 & - & x_2 & - & 2x_3 \end{array}$$

The third constraint is the tightest and limits how much we can increase x_1.

Switch roles of x_1 and x_6:

- Solving for x_1 yields:

$$x_1 = 9 - \frac{x_2}{4} - \frac{x_3}{2} - \frac{x_6}{4}.$$

- Substitute this into x_1 in the other three equations

Extended Example: Iteration 2

Increasing the value of x_3 would increase the objective value.

$$\begin{array}{rcrcrcrcr} z & = & 27 & + & \frac{x_2}{4} & + & \frac{x_3}{2} & - & \frac{3x_6}{4} \\ x_1 & = & 9 & - & \frac{x_2}{4} & - & \frac{x_3}{2} & - & \frac{x_6}{4} \\ x_4 & = & 21 & - & \frac{3x_2}{4} & - & \frac{5x_3}{2} & + & \frac{x_6}{4} \\ x_5 & = & 6 & - & \frac{3x_2}{2} & - & 4x_3 & + & \frac{x_6}{2} \end{array}$$

Basic solution: $(\overline{x_1}, \overline{x_2}, \ldots, \overline{x_6}) = (9, 0, 0, 21, 6, 0)$ with objective value 27

Extended Example: Iteration 2

$$\begin{array}{rcrcrcrcr} z & = & 27 & + & \frac{x_2}{4} & + & \frac{x_3}{2} & - & \frac{3x_6}{4} \\ x_1 & = & 9 & - & \frac{x_2}{4} & - & \frac{x_3}{2} & - & \frac{x_6}{4} \\ x_4 & = & 21 & - & \frac{3x_2}{4} & - & \frac{5x_3}{2} & + & \frac{x_6}{4} \\ x_5 & = & 6 & - & \frac{3x_2}{2} & - & 4x_3 & + & \frac{x_6}{2} \end{array}$$

The third constraint is the tightest and limits how much we can increase x_3.

Switch roles of x_3 and x_5:

- Solving for x_3 yields:

$$x_3 = \frac{3}{2} - \frac{3x_2}{8} - \frac{x_5}{4} - \frac{x_6}{8}.$$

- Substitute this into x_3 in the other three equations

Extended Example: Iteration 3

Increasing the value of x_2 would increase the objective value.

$$\begin{aligned}
z &= \frac{111}{4} + \frac{x_2}{16} - \frac{x_5}{8} - \frac{11x_6}{16}\\
x_1 &= \frac{33}{4} - \frac{x_2}{16} + \frac{x_5}{8} - \frac{5x_6}{16}\\
x_3 &= \frac{3}{2} - \frac{3x_2}{8} - \frac{x_5}{4} + \frac{x_6}{8}\\
x_4 &= \frac{69}{4} + \frac{3x_2}{16} + \frac{5x_5}{8} - \frac{x_6}{16}
\end{aligned}$$

Basic solution: $(\overline{x_1}, \overline{x_2}, \ldots, \overline{x_6}) = (\frac{33}{4}, 0, \frac{3}{2}, \frac{69}{4}, 0, 0)$ with objective value $\frac{111}{4} = 27.75$

Extended Example: Iteration 3

$$\begin{aligned}
z &= \frac{111}{4} + \frac{x_2}{16} - \frac{x_5}{8} - \frac{11x_6}{16}\\
x_1 &= \frac{33}{4} - \frac{x_2}{16} + \frac{x_5}{8} - \frac{5x_6}{16}\\
x_3 &= \frac{3}{2} - \frac{3x_2}{8} - \frac{x_5}{4} + \frac{x_6}{8}\\
x_4 &= \frac{69}{4} + \frac{3x_2}{16} + \frac{5x_5}{8} - \frac{x_6}{16}
\end{aligned}$$

The second constraint is the tightest and limits how much we can increase x_2.

Switch roles of x_2 and x_3:

- Solving for x_2 yields:

$$x_2 = 4 - \frac{8x_3}{3} - \frac{2x_5}{3} + \frac{x_6}{3}.$$

- Substitute this into x_2 in the other three equations

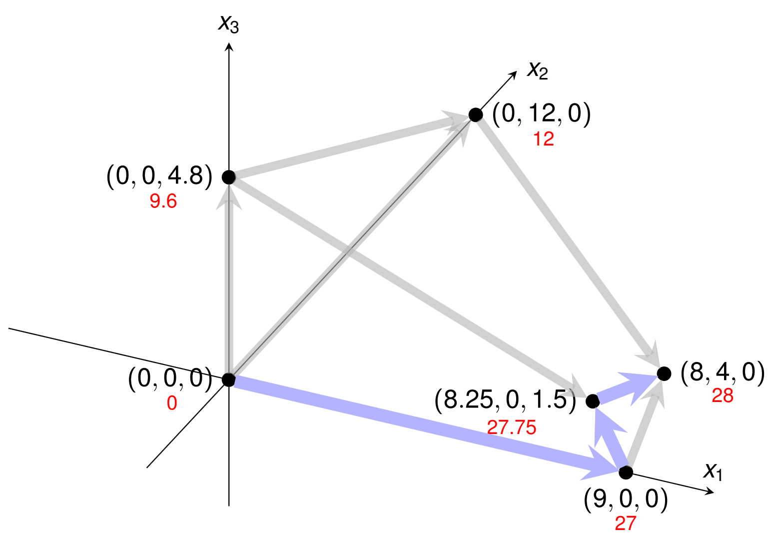
Extended Example: Iteration 4

All coefficients are negative, and hence this basic solution is **optimal**!

$$\begin{aligned}
z &= 28 - \frac{x_3}{6} - \frac{x_5}{6} - \frac{2x_6}{3}\\
x_1 &= 8 + \frac{x_3}{6} + \frac{x_5}{6} - \frac{x_6}{3}\\
x_2 &= 4 - \frac{8x_3}{3} - \frac{2x_5}{3} + \frac{x_6}{3}\\
x_4 &= 18 - \frac{x_3}{2} + \frac{x_5}{2}
\end{aligned}$$

Basic solution: $(\overline{x_1}, \overline{x_2}, \ldots, \overline{x_6}) = (8, 4, 0, 18, 0, 0)$ with objective value 28

Extended Example: Visualization of SIMPLEX



Question: How many basic solutions (including non-feasible ones) are there?

Extended Example: Alternative Runs (1/2)

$$\begin{array}{rcrcrcrcr} z & = & & & 3x_1 & + & x_2 & + & 2x_3 \\ x_4 & = & 30 & - & x_1 & - & x_2 & - & 3x_3 \\ x_5 & = & 24 & - & 2x_1 & - & 2x_2 & - & 5x_3 \\ x_6 & = & 36 & - & 4x_1 & - & x_2 & - & 2x_3 \end{array}$$

Switch roles of x_2 and x_5

$$\begin{array}{rcrcrcrcr} z & = & 12 & + & 2x_1 & - & \frac{x_3}{2} & - & \frac{x_5}{2} \\ x_2 & = & 12 & - & x_1 & - & \frac{5x_3}{2} & - & \frac{x_5}{2} \\ x_4 & = & 18 & - & x_2 & - & \frac{x_3}{2} & + & \frac{x_5}{2} \\ x_6 & = & 24 & - & 3x_1 & + & \frac{x_3}{2} & + & \frac{x_5}{2} \end{array}$$

Switch roles of x_1 and x_6

$$\begin{array}{rcrcrcrcr} z & = & 28 & - & \frac{x_3}{6} & - & \frac{x_5}{6} & - & \frac{2x_6}{3} \\ x_1 & = & 8 & + & \frac{x_3}{6} & + & \frac{x_5}{6} & - & \frac{x_6}{3} \\ x_2 & = & 4 & - & \frac{8x_3}{3} & - & \frac{2x_5}{3} & + & \frac{x_6}{3} \\ x_4 & = & 18 & - & \frac{x_3}{2} & + & \frac{x_5}{2} & & \end{array}$$

Extended Example: Alternative Runs (2/2)

$$\begin{array}{rcrcrcrcr} z & = & & & 3x_1 & + & x_2 & + & 2x_3 \\ x_4 & = & 30 & - & x_1 & - & x_2 & - & 3x_3 \\ x_5 & = & 24 & - & 2x_1 & - & 2x_2 & - & 5x_3 \\ x_6 & = & 36 & - & 4x_1 & - & x_2 & - & 2x_3 \end{array}$$

Switch roles of x_3 and x_5

$$\begin{array}{rcrcrcrcr} z & = & \frac{48}{5} & + & \frac{11x_1}{5} & + & \frac{x_2}{5} & - & \frac{2x_5}{5} \\ x_4 & = & \frac{78}{5} & + & \frac{x_1}{5} & + & \frac{x_2}{5} & + & \frac{3x_5}{5} \\ x_3 & = & \frac{24}{5} & - & \frac{2x_1}{5} & - & \frac{2x_2}{5} & - & \frac{x_5}{5} \\ x_6 & = & \frac{132}{5} & - & \frac{16x_1}{5} & - & \frac{x_2}{5} & + & \frac{2x_3}{5} \end{array}$$

Switch roles of x_1 and x_6

$$\begin{array}{rcrcrcrcr} z & = & \frac{111}{4} & + & \frac{x_2}{16} & - & \frac{x_5}{8} & - & \frac{11x_6}{16} \\ x_1 & = & \frac{33}{4} & - & \frac{x_2}{16} & + & \frac{x_5}{8} & - & \frac{5x_6}{16} \\ x_3 & = & \frac{3}{2} & - & \frac{3x_2}{8} & - & \frac{x_5}{4} & + & \frac{x_6}{8} \\ x_4 & = & \frac{69}{4} & + & \frac{3x_2}{16} & + & \frac{5x_5}{8} & - & \frac{x_6}{16} \end{array}$$

Switch roles of x_2 and x_3

$$\begin{array}{rcrcrcrcr} z & = & 28 & - & \frac{x_3}{6} & - & \frac{x_5}{6} & - & \frac{2x_6}{3} \\ x_1 & = & 8 & + & \frac{x_3}{6} & + & \frac{x_5}{6} & - & \frac{x_6}{3} \\ x_2 & = & 4 & - & \frac{8x_3}{3} & - & \frac{2x_5}{3} & + & \frac{x_6}{3} \\ x_4 & = & 18 & - & \frac{x_3}{2} & + & \frac{x_5}{2} & & \end{array}$$

Outline

The Pivot Step Formally

```
PIVOT(N, B, A, b, c, v, l, e)
// Compute the coefficients of the equation for new basic variable x_e.
let Â be a new m × n matrix
b̂_e = b_l / a_le
for each j ∈ N − {e}
    â_ej = a_lj / a_le
â_el = 1 / a_le
// Compute the coefficients of the remaining constraints.
for each i ∈ B − {l}
    b̂_i = b_i − a_ie b̂_e
    for each j ∈ N − {e}
        â_ij = a_ij − a_ie â_ej
    â_il = −a_ie â_el
// Compute the objective function.
v̂ = v + c_e b̂_e
for each j ∈ N − {e}
    ĉ_j = c_j − c_e â_ej
ĉ_l = −c_e â_el
// Compute new sets of basic and nonbasic variables.
N̂ = N − {e} ∪ {l}
B̂ = B − {l} ∪ {e}
return (N̂, B̂, Â, b̂, ĉ, v̂)
```

Need that $a_{le} \neq 0$!

Rewrite “tight” equation for enterring variable x_e.

Substituting x_e into other equations.

Substituting x_e into objective function.

Update non-basic and basic variables

Formalizing the Simplex Algorithm: Questions

Questions:

- How do we determine whether a linear program is feasible?
- What do we do if the linear program is feasible, but the initial basic solution is not feasible?
- How do we determine whether a linear program is unbounded?
- How do we choose the entering and leaving variables?

Example before was a particularly nice one!

The formal procedure SIMPLEX

```
SIMPLEX(A, b, c)
1  (N, B, A, b, c, v) = INITIALIZE-SIMPLEX(A, b, c)
2  let Δ be a new vector of length m
3  while some index j ∈ N has c_j > 0
4      choose an index e ∈ N for which c_e > 0
5      for each index i ∈ B
6          if a_ie > 0
7              Δ_i = b_i / a_ie
8          else Δ_i = ∞
9      choose an index l ∈ B that minimizes Δ_i
10     if Δ_l == ∞
11         return "unbounded"
12     else (N, B, A, b, c, v) = PIVOT(N, B, A, b, c, v, l, e)
13 for i = 1 to n
14     if i ∈ B
15         x̄_i = b_i
16     else x̄_i = 0
17 return (x̄_1, x̄_2, ..., x̄_n)
```

Returns a slack form with a feasible basic solution (if it exists)

Main Loop:

- terminates if all coefficients in objective function are non-positive
- Line 4 picks enterring variable x_e with positive coefficient
- Lines 6 — 9 pick the tightest constraint, associated with x_l
- Line 11 returns "unbounded" if there are no constraints
- Line 12 calls PIVOT, switching roles of x_l and x_e

Return corresponding solution.

The formal procedure SIMPLEX

```
SIMPLEX(A, b, c)
1  (N, B, A, b, c, v) = INITIALIZE-SIMPLEX(A, b, c)
2  let Δ be a new vector of length m
3  while some index j ∈ N has c_j > 0
4      choose an index e ∈ N for which c_e > 0
5      for each index i ∈ B
6          if a_ie > 0
7              Δ_i = b_i / a_ie
8          else Δ_i = ∞
9      choose an index l ∈ B that minimizes Δ_i
10     if Δ_l == ∞
11         return "unbounded"
```

Proof is based on the following three-part loop invariant:

1. the slack form is always equivalent to the one returned by INITIALIZE-SIMPLEX,
2. for each $i \in B$, we have $b_i \geq 0$,
3. the basic solution associated with the (current) slack form is feasible.

Lemma 29.2

Suppose the call to INITIALIZE-SIMPLEX in line 1 returns a slack form for which the basic solution is feasible. Then if SIMPLEX returns a solution, it is a feasible solution. If SIMPLEX returns "unbounded", the linear program is unbounded.

Outline

Finding an Initial Solution

maximise $2x_1 - x_2$

subject to

$$\begin{aligned} 2x_1 - x_2 &\leq 2 \\ x_1 - 5x_2 &\leq -4 \\ x_1, x_2 &\geq 0 \end{aligned}$$

Conversion into slack form

$$\begin{aligned} z &= & & 2x_1 - x_2 \\ x_3 &= 2 & - & 2x_1 + x_2 \\ x_4 &= -4 & - & x_1 + 5x_2 \end{aligned}$$

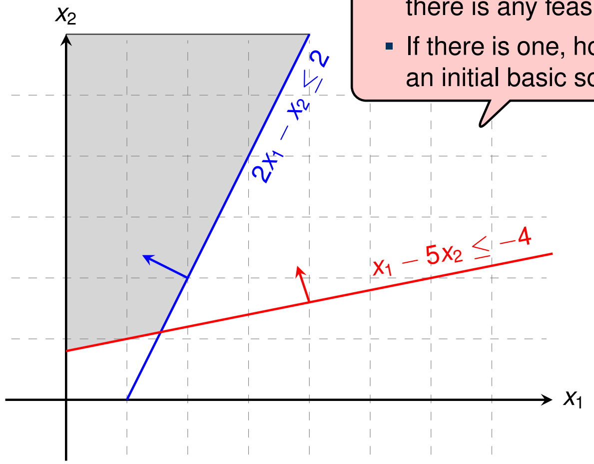
Basic solution $(x_1, x_2, x_3, x_4) = (0, 0, 2, -4)$ is not feasible!

Geometric Illustration

maximise $2x_1 - x_2$

subject to

$$\begin{aligned} 2x_1 - x_2 &\leq 2 \\ x_1 - 5x_2 &\leq -4 \\ x_1, x_2 &\geq 0 \end{aligned}$$



Questions:

- How to determine whether there is any feasible solution?
- If there is one, how to determine an initial basic solution?

Formulating an Auxiliary Linear Program

maximise $\sum_{j=1}^{n} c_j x_j$

subject to

$$\begin{aligned} \sum_{j=1}^{n} a_{ij} x_j &\leq b_i \quad \text{for } i = 1, 2, \ldots, m, \\ x_j &\geq 0 \quad \text{for } j = 1, 2, \ldots, n \end{aligned}$$

Formulating an Auxiliary Linear Program

maximise $-x_0$

subject to

$$\begin{aligned} \sum_{j=1}^{n} a_{ij} x_j - x_0 &\leq b_i \quad \text{for } i = 1, 2, \ldots, m, \\ x_j &\geq 0 \quad \text{for } j = 0, 1, \ldots, n \end{aligned}$$

Lemma 29.11

Let L_{aux} be the auxiliary LP of a linear program L in standard form. Then L is feasible if and only if the optimal objective value of L_{aux} is 0.

Proof. Exercise!

- Let us illustrate the role of x_0 as "distance from feasibility"
- We'll also see that increasing x_0 enlarges the feasible region

Geometric Illustration

$$\begin{array}{ll} \text{maximise} & -x_0 \\ \text{subject to} & \\ & \begin{array}{rcrcrcr} 2x_1 & - & x_2 & - & x_0 & \leq & 2 \\ x_1 & - & 5x_2 & - & x_0 & \leq & -4 \\ & & & & x_0, x_1, x_2 & \geq & 0 \end{array} \end{array}$$

For the animation see the full slides.

- Let us now modify the original linear program so that it is not feasible

⇒ Hence the auxiliary linear program has only a solution for a sufficiently large $x_0 > 0$!

Geometric Illustration

$$\begin{array}{ll} \text{maximise} & -x_0 \\ \text{subject to} & \\ & \begin{array}{rcrcrcr} 2x_1 & - & x_2 & - & x_0 & \leq & -2 \\ -x_1 & + & 5x_2 & - & x_0 & \leq & 4 \\ & & & & x_0, x_1, x_2 & \geq & 0 \end{array} \end{array}$$

For the animation see the full slides.

INITIALIZE-SIMPLEX

```
INITIALIZE-SIMPLEX(A, b, c)
let k be the index of the minimum b_i
if b_k ≥ 0                      // is the initial basic solution feasible?
    return ({1, 2, ..., n}, {n + 1, n + 2, ..., n + m}, A, b, c, 0)
form L_aux by adding −x_0 to the left-hand side of each constraint
    and setting the objective function to −x_0
let (N, B, A, b, c, v) be the resulting slack form for L_aux
l = n + k
// L_aux has n + 1 nonbasic variables and m basic variables.
(N, B, A, b, c, v) = PIVOT(N, B, A, b, c, v, l, 0)
// The basic solution is now feasible for L_aux.
iterate the while loop of lines 3–12 of SIMPLEX until an optimal solution
    to L_aux is found
if the optimal solution to L_aux sets x̄_0 to 0
    if x̄_0 is basic
        perform one (degenerate) pivot to make it nonbasic
    from the final slack form of L_aux, remove x_0 from the constraints and
        restore the original objective function of L, but replace each basic
        variable in this objective function by the right-hand side of its
        associated constraint
    return the modified final slack form
else return "infeasible"
```

> Test solution with $N = \{1, 2, \ldots, n\}$, $B = \{n+1, n+2, \ldots, n+m\}$, $\overline{x}_i = b_i$ for $i \in B$, $\overline{x}_i = 0$ otherwise.

> ℓ will be the leaving variable so that x_ℓ has the most negative value.

> Pivot step with x_ℓ leaving and x_0 entering.

> This pivot step does not change the value of any variable.

Example of INITIALIZE-SIMPLEX (1/3)

$$\begin{array}{ll} \text{maximise} & 2x_1 - x_2 \\ \text{subject to} & \\ & 2x_1 - x_2 \le 2 \\ & x_1 - 5x_2 \le -4 \\ & x_1, x_2 \ge 0 \end{array}$$

Formulating the auxiliary linear program (as basic solution would not be feasible!)

$$\begin{array}{ll} \text{maximise} & -x_0 \\ \text{subject to} & \\ & 2x_1 - x_2 - x_0 \le 2 \\ & x_1 - 5x_2 - x_0 \le -4 \\ & x_1, x_2, x_0 \ge 0 \end{array}$$

Basic solution $(0, 0, 0, 2, -4)$ not feasible!

Converting into slack form

$$\begin{array}{rcl} z &=& -x_0 \\ x_3 &=& 2 - 2x_1 + x_2 + x_0 \\ x_4 &=& -4 - x_1 + 5x_2 + x_0 \end{array}$$

Example of INITIALIZE-SIMPLEX (2/3)

$$\begin{array}{rcl} z &=& -x_0 \\ x_3 &=& 2 - 2x_1 + x_2 + x_0 \\ x_4 &=& -4 - x_1 + 5x_2 + x_0 \end{array}$$

Pivot with x_0 entering and x_4 leaving

$$\begin{array}{rcl} z &=& -4 - x_1 + 5x_2 - x_4 \\ x_0 &=& 4 + x_1 - 5x_2 + x_4 \\ x_3 &=& 6 - x_1 - 4x_2 + x_4 \end{array}$$

Basic solution $(4, 0, 0, 6, 0)$ is feasible!

Pivot with x_2 entering and x_0 leaving

$$\begin{array}{rcl} z &=& -x_0 \\ x_2 &=& \frac{4}{5} - \frac{x_0}{5} + \frac{x_1}{5} + \frac{x_4}{5} \\ x_3 &=& \frac{14}{5} + \frac{4x_0}{5} - \frac{9x_1}{5} + \frac{x_4}{5} \end{array}$$

Optimal solution has $x_0 = 0$, hence the initial problem was feasible!

Example of INITIALIZE-SIMPLEX (3/3)

$$\begin{array}{rcl} z &=& -x_0 \\ x_2 &=& \frac{4}{5} - \frac{x_0}{5} + \frac{x_1}{5} + \frac{x_4}{5} \\ x_3 &=& \frac{14}{5} + \frac{4x_0}{5} - \frac{9x_1}{5} + \frac{x_4}{5} \end{array}$$

Set $x_0 = 0$ and express objective function by non-basic variables

$2x_1 - x_2 = 2x_1 - (\frac{4}{5} - \frac{x_0}{5} + \frac{x_1}{5} + \frac{x_4}{5})$

$$\begin{array}{rcl} z &=& -\frac{4}{5} + \frac{9x_1}{5} - \frac{x_4}{5} \\ x_2 &=& \frac{4}{5} + \frac{x_1}{5} + \frac{x_4}{5} \\ x_3 &=& \frac{14}{5} - \frac{9x_1}{5} + \frac{x_4}{5} \end{array}$$

Basic solution $(0, \frac{4}{5}, \frac{14}{5}, 0)$, which is feasible!

Lemma 29.12

If a linear program L has no feasible solution, then INITIALIZE-SIMPLEX returns "infeasible". Otherwise, it returns a valid slack form for which the basic solution is feasible.

Fundamental Theorem of Linear Programming

Theorem 29.13 (Fundamental Theorem of Linear Programming)

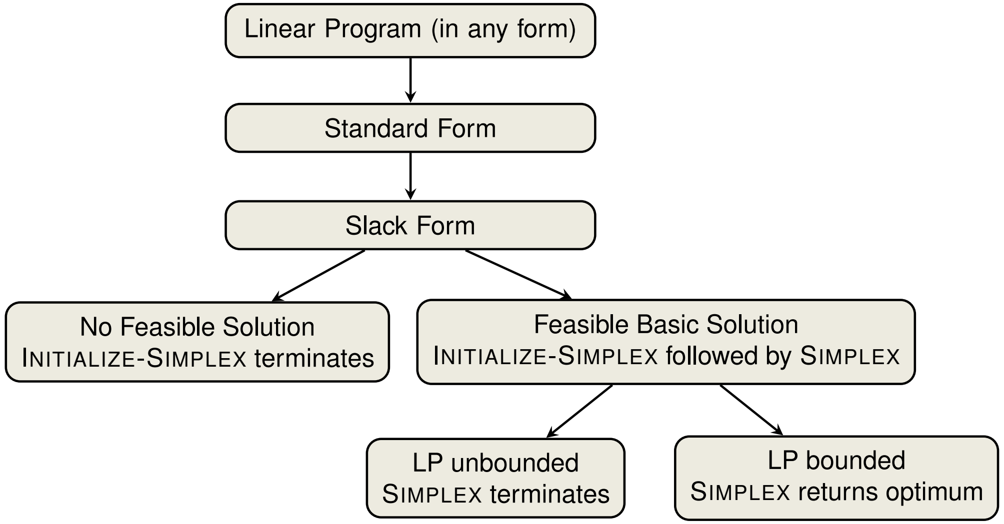
For any linear program L, given in standard form, either:

1. L is infeasible $\Rightarrow$ SIMPLEX returns "infeasible".
2. L is unbounded $\Rightarrow$ SIMPLEX returns "unbounded".
3. L has an optimal solution with a finite objective value $\Rightarrow$ SIMPLEX returns an optimal solution with a finite objective value.

Small Technicality: need to equip SIMPLEX with an "anti-cycling strategy" (see extra slides)

Proof requires the concept of duality, which is not covered in this course (for details see CLRS3, Chapter 29.4)

Workflow for Solving Linear Programs



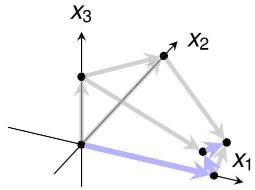
Linear Programming and Simplex: Summary and Outlook

Linear Programming

- extremely versatile tool for modelling problems of all kinds
- basis of Integer Programming, to be discussed in later lectures

Simplex Algorithm

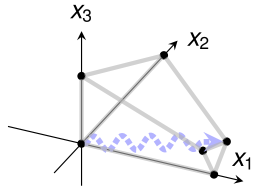
- In practice: usually terminates in polynomial time, i.e., $O(m+n)$
- In theory: even with anti-cycling may need exponential time



Research Problem: Is there a pivoting rule which makes SIMPLEX a polynomial-time algorithm?

Polynomial-Time Algorithms

- Interior-Point Methods: traverses the interior of the feasible set of solutions (not just vertices!)



Outlook: Alternatives to Worst Case Analysis (non-examinable)

1.2 Famous Failures and the Need for Alternatives

For many problems a bit beyond the scope of an undergraduate course, the downside of worst-case analysis rears its ugly head. This section reviews four famous examples in which worst-case analysis gives misleading or useless advice about how to solve a problem. These examples motivate the alternatives to worst-case analysis that are surveyed in Section 1.4 and described in detail in later chapters of the book.

1.2.1 The Simplex Method for Linear Programming

Perhaps the most famous failure of worst-case analysis concerns linear programming, the problem of optimizing a linear function subject to linear constraints (Figure 1.1). Dantzig proposed in the 1940s an algorithm for solving linear programs called the *simplex method*. The simplex method solves linear programs using greedy local

Source: "Beyond the Worst-Case Analysis of Algorithms" by Tim Roughgarden, 2020

Outline

Termination

Degeneracy: One iteration of SIMPLEX leaves the objective value unchanged.

$$\begin{aligned} z &= x_1 + x_2 + x_3 \\ x_4 &= 8 - x_1 - x_2 \\ x_5 &= x_2 - x_3 \end{aligned}$$

Pivot with x_1 entering and x_4 leaving

$$\begin{aligned} z &= 8 + x_3 - x_4 \\ x_1 &= 8 - x_2 - x_4 \\ x_5 &= x_2 - x_3 \end{aligned}$$

Cycling: If additionally slack form at two iterations are identical, SIMPLEX fails to terminate!

Pivot with x_3 entering and x_5 leaving

$$\begin{aligned} z &= 8 + x_2 - x_4 - x_5 \\ x_1 &= 8 - x_2 - x_4 \\ x_3 &= x_2 - x_5 \end{aligned}$$

Exercise: Execute one more step of the Simplex Algorithm on the tableau from the previous slide.

Termination and Running Time

It is theoretically possible, but very rare in practice.

Cycling: SIMPLEX may fail to terminate.

Anti-Cycling Strategies

1. Bland's rule: Choose entering variable with smallest index
2. Random rule: Choose entering variable uniformly at random
3. Perturbation: Perturb the input slightly so that it is impossible to have two solutions with the same objective value

Replace each b_i by $\hat{b}_i = b_i + \epsilon_i$, where $\epsilon_i \gg \epsilon_{i+1}$ are all small.

Lemma 29.7

Assuming INITIALIZE-SIMPLEX returns a slack form for which the basic solution is feasible, SIMPLEX either reports that the program is unbounded or returns a feasible solution in at most $\binom{n+m}{m}$ iterations.

Every set B of basic variables uniquely determines a slack form, and there are at most $\binom{n+m}{m}$ unique slack forms.