Randomised Algorithms
Lecture 7: Linear Programming: Simplex Algorithm

Thomas Sauerwald (tms41@cam.ac.uk)

Lent 2024

Outline

Simplex Algorithm by Example

Details of the Simplex Algorithm

Finding an Initial Solution

Appendix: Cycling and Termination (non-examinable)

Simplex Algorithm: Introduction

- 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
- In that sense, it is a greedy algorithm
- Conversion (“pivoting”) is achieved by switching the roles of one basic and one non-basic variable

Extended Example: Conversion into Slack Form

maximise $3x_1 + x_2 + 2x_3$
subject to

$x_1 + x_2 + 3x_3 \leq 30$
$2x_1 + 2x_2 + 5x_3 \leq 24$
$4x_1 + x_2 + 2x_3 \leq 36$

$x_1, x_2, x_3 \geq 0$

Conversion into slack form:

$z = 3x_1 + x_2 + 2x_3$
$x_4 = 30 - x_1 - x_2 - 3x_3$
$x_5 = 24 - 2x_1 - 2x_2 - 5x_3$
$x_6 = 36 - 4x_1 - x_2 - 2x_3$
Extended Example: Iteration 1

\[ z = 3x_1 + x_2 + 2x_3 \]

\[ x_4 = 30 - x_1 - x_2 - 3x_3 \]

\[ x_5 = 24 - 2x_1 - 2x_2 - 5x_3 \]

\[ x_6 = 36 - 4x_1 - x_2 - 2x_3 \]

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

This basic solution is feasible

Objective value is 0.

Extended Example: Iteration 1

\[ z = 3x_1 + x_2 + 2x_3 \]

\[ x_4 = 30 - x_1 - x_2 - 3x_3 \]

\[ x_5 = 24 - 2x_1 - 2x_2 - 5x_3 \]

\[ x_6 = 36 - 4x_1 - x_2 - 2x_3 \]

Increasing the value of \(x_1\) would increase the objective value.

\[ z = 27 + \frac{x_2}{4} + \frac{x_3}{2} - \frac{3x_5}{4} \]

\[ x_1 = 9 - \frac{x_2}{4} - \frac{x_3}{2} - \frac{x_5}{4} \]

\[ x_4 = 21 - \frac{3x_2}{4} - \frac{5x_3}{2} + \frac{x_5}{4} \]

\[ x_5 = 6 - \frac{3x_2}{2} - 4x_3 + \frac{x_5}{2} \]

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

Extended Example: Iteration 2

Increasing the value of \(x_3\) would increase the objective value.

\[ z = 27 + \frac{x_2}{4} + \frac{x_3}{2} - \frac{3x_5}{4} \]

\[ x_1 = 9 - \frac{x_2}{4} - \frac{x_3}{2} - \frac{x_5}{4} \]

\[ x_4 = 21 - \frac{3x_2}{4} - \frac{5x_3}{2} + \frac{x_5}{4} \]

\[ x_5 = 6 - \frac{3x_2}{2} - 4x_3 + \frac{x_5}{2} \]

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:
  \[ x_1 = 9 - \frac{x_2}{4} - \frac{x_3}{2} - \frac{x_5}{4} \]
- Substitute this into \(x_1\) in the other three equations

Extended Example: Iteration 2

\[ z = 27 + \frac{x_2}{4} + \frac{x_3}{2} - \frac{3x_5}{4} \]

\[ x_1 = 9 - \frac{x_2}{4} - \frac{x_3}{2} - \frac{x_5}{4} \]

\[ x_4 = 21 - \frac{3x_2}{4} - \frac{5x_3}{2} + \frac{x_5}{4} \]

\[ x_5 = 6 - \frac{3x_2}{2} - 4x_3 + \frac{x_5}{2} \]

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.

$$z = \frac{111}{4} + \frac{x_1}{16} - \frac{x_5}{8} - \frac{11x_6}{16}$$

$$x_1 = \frac{33}{4} - \frac{x_2}{16} + \frac{x_5}{8} - \frac{5x_6}{16}$$

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

$$x_4 = \frac{69}{4} + \frac{3x_2}{16} + \frac{5x_5}{8} - \frac{x_6}{16}$$

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

Extended Example: Iteration 4

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

$$z = 28 - \frac{x_1}{6} - \frac{x_5}{5} - \frac{2x_6}{3}$$

$$x_1 = 8 + \frac{x_3}{6} + \frac{x_5}{5} - \frac{x_6}{3}$$

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

$$x_4 = 18 - \frac{x_3}{2} + \frac{x_5}{2}$$

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

Extended Example: Visualization of SIMPLEX

- 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.

Exercise: [Ex. 6/7.6] How many basic solutions (including non-feasible ones) are there?
Extended Example: Alternative Runs (1/2)

\[ z = 3x_1 + x_2 + 2x_3 \]
\[ x_4 = 30 - x_1 - x_2 - 3x_3 \]
\[ x_5 = 24 - 2x_1 - 2x_2 - 5x_3 \]
\[ x_6 = 36 - 4x_1 - x_2 - 2x_3 \]

Switch roles of \( x_2 \) and \( x_6 \)

\[ z = 12 + 2x_1 - x_3 - x_5 \]
\[ x_2 = 12 - x_1 - \frac{5x_3}{2} - \frac{x_5}{2} \]
\[ x_4 = 18 - x_2 - x_3 - \frac{x_5}{2} \]
\[ x_6 = 24 - 3x_1 + \frac{3x_3}{2} + \frac{x_5}{2} \]

Switch roles of \( x_1 \) and \( x_6 \)

\[ z = 28 - \frac{x_2}{4} - \frac{x_5}{4} - \frac{2x_6}{4} \]
\[ x_1 = 8 + \frac{x_2}{4} + \frac{x_5}{4} - \frac{4x_6}{4} \]
\[ x_2 = 4 - \frac{8x_3}{4} - \frac{2x_5}{4} + \frac{x_6}{4} \]
\[ x_4 = 18 - \frac{x_2}{4} + \frac{x_5}{4} \]

Extended Example: Alternative Runs (2/2)

\[ z = 3x_1 + x_2 + 2x_3 \]
\[ x_4 = 30 - x_1 - x_2 - 3x_3 \]
\[ x_5 = 24 - 2x_1 - 2x_2 - 5x_3 \]
\[ x_6 = 36 - 4x_1 - x_2 - 2x_3 \]

Switch roles of \( x_3 \) and \( x_6 \)

\[ z = \frac{48}{5} + \frac{11x_1}{5} + \frac{x_2}{5} - \frac{2x_5}{5} \]
\[ x_4 = \frac{78}{5} + \frac{x_1}{5} + \frac{x_2}{5} + \frac{3x_5}{5} \]
\[ x_5 = \frac{24}{5} - \frac{2x_1}{5} - \frac{2x_2}{5} - \frac{x_5}{5} \]
\[ x_6 = \frac{132}{5} - \frac{16x_1}{5} - \frac{x_2}{5} + \frac{2x_3}{5} \]

Switch roles of \( x_1 \) and \( x_6 \)

\[ z = \frac{111}{4} + \frac{x_2}{16} - \frac{x_5}{16} - \frac{11x_6}{16} \]
\[ x_1 = \frac{32}{4} - \frac{x_2}{16} + \frac{x_5}{16} - \frac{5x_6}{16} \]
\[ x_2 = 4 - \frac{8x_3}{4} - \frac{2x_5}{4} + \frac{x_6}{4} \]
\[ x_3 = \frac{69}{4} + \frac{3x_1}{16} + \frac{5x_2}{16} - \frac{x_6}{16} \]
\[ x_4 = 18 - \frac{x_1}{4} + \frac{x_2}{4} \]

The Pivot Step Formally

\[ \text{Pivot}(N, B, A, b, c, v, l, e) \]
1. // Compute the coefficients of the equation for new basic variable \( x_e \).
2. let \( \hat{A} \) be a new \( m \times n \) matrix
3. \( \hat{b}_e = b_l/a_{le} \)
4. for each \( j \in N - \{e\} \)
5. \( \hat{a}_{ej} = a_{ij}/a_{le} \)
6. \( \hat{a}_{e} = 1/a_{le} \)
7. // Compute the coefficients of the remaining constraints.
8. for each \( i \in B - \{l\} \)
9. \( \hat{b}_i = b_i - a_{il}\hat{b}_e \)
10. for each \( j \in N - \{e\} \)
11. \( \hat{a}_{ij} = a_{ij} - a_{il}\hat{a}_{e} \)
12. \( \hat{a}_{e} = -a_{il}\hat{a}_{e} \)
13. // Compute the objective function.
14. \( \hat{v} = \psi + c_{e}\hat{b}_e \)
15. for each \( j \in N - \{e\} \)
16. \( \hat{c}_j = c_{j} - c_{e}\hat{a}_{ej} \)
17. \( \hat{c}_{e} = -c_{e}\hat{a}_{e} \)
18. // Compute new sets of basic and nonbasic variables.
19. \( \hat{N} = N - \{e\} \cup \{l\} \)
20. \( \hat{B} = B - \{l\} \cup \{e\} \)
21. return \( (\hat{N}, \hat{B}, \hat{A}, \hat{b}, \hat{c}, \hat{v}) \)
Effect of the Pivot Step (extra material, non-examinable)

Lemma 29.1

Consider a call to \( \text{Pivot}(N, B, A, b, c, v, l, e) \) in which \( a_{le} \neq 0 \). Let the values returned from the call be \( (\hat{N}, \hat{B}, \hat{A}, \hat{b}, \hat{c}, \hat{v}) \), and let \( \hat{x} \) denote the basic solution after the call. Then

1. \( \hat{x}_i = 0 \) for each \( j \in \hat{N} \).
2. \( \hat{x}_e = b_e/a_{le} \).
3. \( \hat{x}_i = b_i - a_{ie}\hat{b}_e \) for each \( i \in \hat{B} \setminus \{e\} \).

Proof:

1. holds since the basic solution always sets all non-basic variables to zero.
2. When we set each non-basic variable to 0 in a constraint
   \[ x_i = \hat{b}_i - \sum_{j \in \hat{N}} \hat{a}_{ij}x_j, \]
   we have \( \hat{x}_i = \hat{b}_i \) for each \( i \in \hat{B} \). Hence \( \hat{x}_e = \hat{b}_e/a_{le} \).
3. After substituting into the other constraints, we have
   \[ \hat{x}_i = \hat{b}_i = b_i - a_{ie}\hat{b}_e. \]

The formal procedure SIMPLEX

\begin{verbatim}
SIMPLEX(A,b,c) = INITIALIZE-SIMPLEX(A,b,c)
1. let A be a new vector of length m
2. while some index j \in N has c_j > 0
3. for each index i \in B
4. choose an index e \in N for which c_e > 0
5. if a_{ie} > 0
6. \Delta_i = b_i/a_{ie}
7. else \Delta_i = \infty
8. choose an index l \in B that minimizes \Delta_i
9. if \Delta_i = \infty
10. return "unbounded"
11. else (N, B, A, b, c, v) = PIVOT(N, B, A, b, c, v, l, e)
12. for i = 1 to n
13. if \hat{x} \in B
14. \hat{x}_i = b_i
15. else \hat{x}_i = 0
16. return (\hat{x}_1, \hat{x}_2, ..., \hat{x}_n)
\end{verbatim}

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 entering 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.

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

\begin{verbatim}
SIMPLEX(A,b,c) = INITIALIZE-SIMPLEX(A,b,c)
1. let A be a new vector of length m
2. while some index j \in N has c_j > 0
3. for each index i \in B
4. choose an index e \in N for which c_e > 0
5. if a_{ie} > 0
6. \Delta_i = b_i/a_{ie}
7. else \Delta_i = \infty
8. choose an index l \in B that minimizes \Delta_i
9. if \Delta_i = \infty
10. return "unbounded"
11. else (N, B, A, b, c, v) = PIVOT(N, B, A, b, c, v, l, e)
12. for i = 1 to n
13. if \hat{x} \in B
14. \hat{x}_i = b_i
15. else \hat{x}_i = 0
16. return (\hat{x}_1, \hat{x}_2, ..., \hat{x}_n)
\end{verbatim}

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.

Lemma 29.2

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.
Finding an Initial Solution

maximise \( 2x_1 - x_2 \)
subject to \( \begin{align*}
2x_1 - x_2 &\leq 2 \\
-x_1 - 5x_2 &\leq -4 \\
x_1, x_2 &\geq 0
\end{align*} \)

Conversion into slack form
\[
\begin{align*}
z &= 2x_1 - x_2 \\
x_3 &= 2 - 2x_1 + x_2 \\
x_4 &= -4 - x_1 + 5x_2
\end{align*}
\]

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

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

maximise \(-x_0\)
subject to \( \begin{align*}
\sum_{j=1}^{n} a_{ij} x_j - x_0 &\leq b_i \quad \text{for } i = 1, 2, \ldots, m, \\
x_j &\geq 0 \quad \text{for } j = 0, 1, \ldots, n
\end{align*} \)

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.

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 $x_0 > 0$!
Example of INITIALIZE-SIMPLEX (2/3)

\[
\begin{align*}
    z &= -2x_1 - 2x_2 - x_0 \\
    x_3 &= 2 - 2x_1 + x_2 + x_0 \\
    x_4 &= -4 - x_1 + 5x_2 + x_0 \\
\end{align*}
\]

Pivot with \( x_0 \) entering and \( x_4 \) leaving

\[
\begin{align*}
    z &= -4 - x_1 + 5x_2 - x_4 \\
    x_0 &= 4 + x_1 - 5x_2 + x_4 \\
    x_3 &= 6 - x_1 - 4x_2 + x_4 \\
\end{align*}
\]

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

Pivot with \( x_2 \) entering and \( x_0 \) leaving

\[
\begin{align*}
    z &= -x_0 \\
    x_2 &= 4 - \frac{3}{5}x_3 + \frac{1}{5}x_1 + \frac{2}{5}x_4 \\
    x_3 &= 14 \frac{5}{5} + 4x_0 - 9x_1 - \frac{3}{5}x_2 + \frac{2}{5}x_4 \\
\end{align*}
\]

Optimal solution has \( x_0 = 0 \), hence the initial problem was feasible!

Example of INITIALIZE-SIMPLEX (3/3)

\[
\begin{align*}
    z &= -x_0 \\
    x_2 &= 4 - \frac{3}{5}x_3 + \frac{1}{5}x_1 + \frac{2}{5}x_4 \\
    x_3 &= 14 \frac{5}{5} + 4x_0 - 9x_1 - \frac{3}{5}x_2 + \frac{2}{5}x_4 \\
\end{align*}
\]

Set \( x_0 = 0 \) and express objective function by non-basic variables

\[
\begin{align*}
    z &= -4 + 9x_1 - \frac{3}{5}x_2 + \frac{1}{5}x_3 + \frac{2}{5}x_4 \\
    x_2 &= 4 + \frac{1}{5}x_3 + \frac{4}{5}x_4 \\
    x_3 &= 14 \frac{5}{5} - 9x_1 + \frac{2}{5}x_4 \\
\end{align*}
\]

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)

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!)

Workflow for Solving Linear Programs

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

Simplex Algorithm by Example

Details of the Simplex Algorithm

Finding an Initial Solution

Appendix: Cycling and Termination (non-examinable)

7. Linear Programming © T. Sauerwald Appendix: Cycling and Termination (non-examinable) 30

Termination

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

\[
\begin{align*}
z &= x_1 + x_2 + x_3 \\
x_4 &= 8 - x_1 - x_2 \\
x_5 &= x_2 - x_3 \\
\end{align*}
\]

\[\text{Pivot with } x_1 \text{ entering and } x_4 \text{ leaving}\]

\[
\begin{align*}
z &= 8 + x_3 - x_4 \\
x_1 &= 8 - x_2 - x_4 \\
x_5 &= x_2 - x_3 \\
\end{align*}
\]

\[\text{Pivot with } x_2 \text{ entering and } x_5 \text{ leaving}\]

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

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

7. Linear Programming © T. Sauerwald Appendix: Cycling and Termination (non-examinable) 31

Termination and Running Time

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.