Hans Bodlaender
Algorithms for Decision Support 2021/2022

Based on material by
Marjan van den Akker

SOLVING LPS: THE SIMPLEX ALGORITHM

LINEAR PROGRAMS

$\max cx$

Such that $Ax \leq b$ and $x \geq 0$

Here: c, x are (real) vectors, cx is inner product, A is matrix, Ax is matrix-vector multiplication, $x \geq 0$ tells each element of x is nonnegative

Variants with $\min$ (multiply by -1) and $Ax = b$ (use $\leq$ and $\geq$)

SLACK VARIABLES

A constraint $a_{i1}x_1 \ + \ \ldots \ + \ a_{in}x_n \ \leq b_i$ can be turned into an equality with help of a slack variable

Take a new variable z_i which also must fulfill $z_i \geq 0$ and change the constraint to:

$a_{i1}x_1 \ + \ \ldots + a_{in}x_{in} + z_i = b_i$

Note: z_i gets the value $a_{i1}x_1 \ + \ \ldots \ + \ a_{in}x_n - b_\mathrm{i}$

ALGORITHMS TO SOLVE LPS

Simplex method (today)

Ellipsoid method

- Polynomial time algorithm (Khachian, 1979)
- Not practical

Interior points methods

- Polynomial time algorithm (Karmakar, 1984)
- Outperforms Simplex for very large instances

DIFFERENT CASES FOR THE OPTIMAL SOLUTIONS FOR AN LP

No solution

Bounded optimum

Unbounded optimum

- $\max x_1$
  - $-x_1 \leq 0$

SIMPLEX METHOD

Dantzig (1947)

Often fast, but can use exponential number of steps

Always solves LP in finite number of steps

Implementation can give numerical (rounding) problems

Use program like Gurobi, CPLEX, GLPK (or, for small inputs, Excel)

Name comes from: set that fulfills constraints $Ax \leq b$ and $x \geq 0$ is a simplex

BASIC IDEA - NOTIONS

"Greedy" method that moves from vertex to "better" vertex

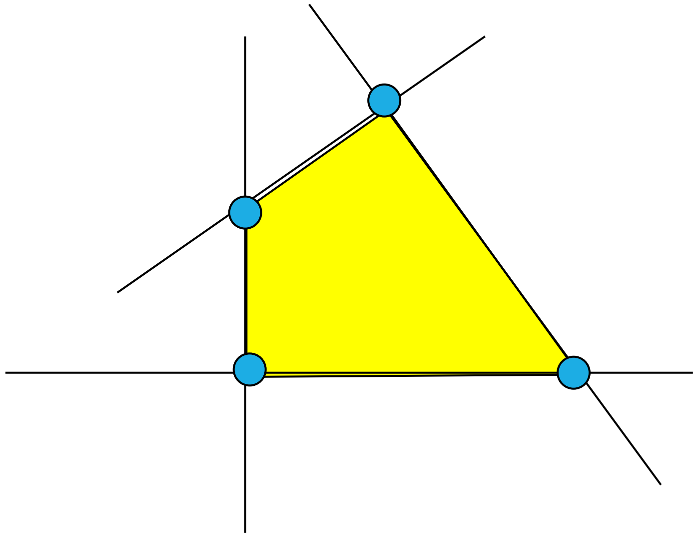
- Called "pivot operation"

Solution or feasible points: fulfills constraints (but not possibly optimum)

Set of feasible points: feasible region

Vertex ~ corner point of set of solutions

Objective function

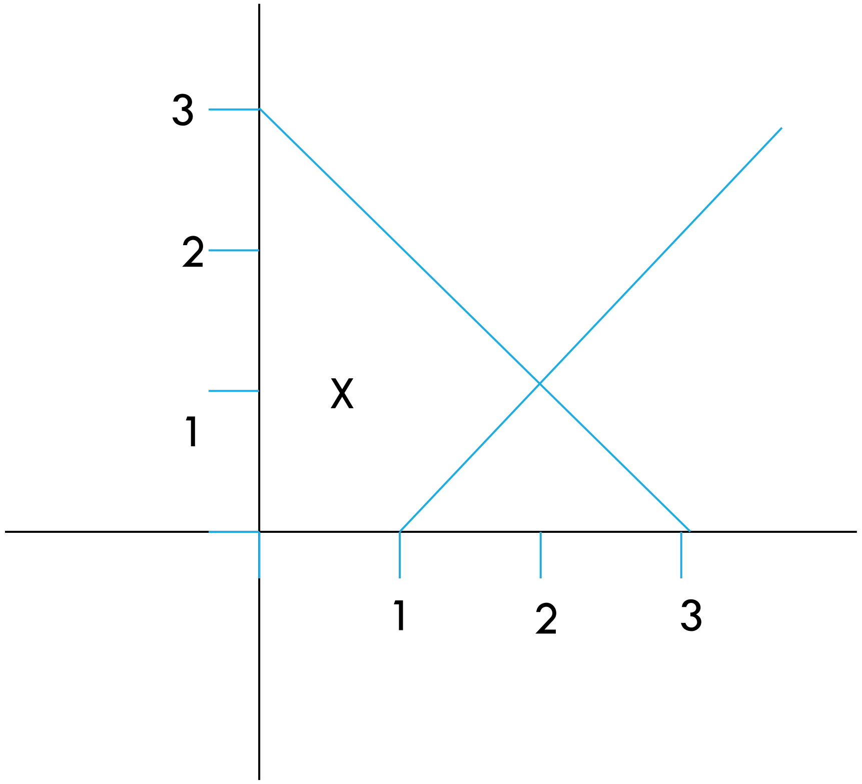


ALGORITHM DESCRIPTION BY EXAMPLE

Consider the LP of the form

$\max 2x_1 + x_2$

- $x_1 - x_2 \leq 1$
- $x_1 + x_2 \leq 3$
- $x_1 \geq 0, x_2 \geq 0$



X: feasible region

INTRODUCE NEW VARIABLES AND FORM EQUATIONS

$\max z$

- $z - 2x_1 - x_2 = 0$
- $x_1 - x_2 + x_3 = 1$
- $x_1 + x_2 + x_4 = 3$
- $x_1 \geq 0, x_2 \geq 0, x_3 \geq 0, x_4 \geq 0$

- Slack variables, and variable for objective function

INTRODUCE NEW VARIABLES AND FORM EQUATIONS

$\max z$

- $z - 2x_1 - x_2 = 0$
- $x_1 - x_2 + x_3 = 1$
- $x_1 + x_2 + x_4 = 3$
- $x_1 \geq 0, x_2 \geq 0, x_3 \geq 0, x_4 \geq 0$

- Start with some solution, e.g., $x_1 = 0, x_2 = 0, x_3 = 1, x_4 = 3$
- Easy when all values in b are non-negative (set the original variables to 0), otherwise additional (similar) tricks

REWRITE: SET VARIABLES THAT ARE 0 AT THE RIGHT HAND SIDE

- $z - 2x_1 - x_2 = 0$
- $x_1 - x_2 + x_3 = 1$
- $x_1 + x_2 + x_4 = 3$

To

- $z = 2x_1 + x_2$
- $x_3 = 1 - x_1 + x_2$
- $x_4 = 3 - x_1 - x_2$

IMPROVING CURRENT SOLUTION

Choose a variable that now is 0.

Can we increase it such that z becomes larger?

- $z = 2x_1 + x_2$
- $x_3 = 1 - x_1 + x_2$
- $x_4 = 3 - x_1 - x_2$

In the example: we can (try to) change x_1 or x_2

All other variables that are 0 (at the right hand side) stay 0

The left hand side must stay non-negative

IMPROVING CURRENT SOLUTION

(0,0,1,3)
Becomes
(1,0,0,2)

- $z = 2x_1 + x_2$
- $x_3 = 1 - x_1 + x_2$
- $x_4 = 3 - x_1 - x_2$

The objective increases by 2 for each unit of x_1 and by 1 for each unit of x_2

Greedy choice: increase x_1

We can increase x_1 as much as possible till a left-hand-side variable becomes 0

So: x_1 becomes 1, and x_3 becomes 0

x_4 is also changed, and becomes 2

PIVOT

We again have two variables that are 0, and rewrite the equations accordingly (our current solution is (1,0,0,2)

- $z \; - 2x_1 - x_2 = 0$
- $x_1 - x_2 + x_3 = 1$
- $x_1 + x_2 + x_4 = 3$

Is rewritten:

- $z = 2x_1 + x_2$ but: x_1 must not be in the right-hand side
- $x_1 = 1 + x_2 - x_3$
- $x_4 = \;\ldots$

We can only use x_2 and x_3 in the right-hand side

PIVOT

Techniques from Linear Algebra (Gauss) to rewrite formulas in desired form

Our current solution is (1,0,0,2)

- $z - 2x_1 - x_2 = 0$
- $x_1 - x_2 + x_3 = 1$
- $x_1 + x_2 + x_4 = 3$

Use that $x_1 = 1 + x_2 - x_3$ to get rid of it at the right-hand side

We get:

- $z = 2 + 3x_2 + 2x_3$
- $x_1 = 1 + x_2 - x_3$
- $x_4 = 2 - 2x_2 + x_3$

NEXT STEP

Again, we are increasing a variable to increase z

- $z = 2 + 3x_2 + 2x_3$
- $x_1 = 1 + x_2 - x_3$
- $x_4 = 2 - 2x_2 + x_3$

Greedy choose to change (increase) x_2 :
we can increase it to 1, making x_4 0, and setting x_1 to 2

From (1,0,0,2) we go to (2,1,0,0)

AGAIN, REFORMULATE

Now, with x_3 and x_4 at the right-hand side

With simple linear algebra:

- $z = 5 - 0.5x_3 - 1.5\,x_4$
- $x_1 = 2 - 0.5x_3 - 0.5x_4$
- $x_2 = 1 + 0.5x_3 - 0.5x_4$

Our solution was (2,1,0,0)

Now: we cannot increase a variable that is 0 and increase z

Theory tells we have the optimum!

SOME REMARKS

In the example, the solution was integer, but this is often not the case

Each step, the objective z increases

Each step, we have a different set of variables that is 0

If we have n variables, then at most 2^n pivot steps

There are examples where we use an exponential number of steps

In practice: fast

Implementation: use 2-dimensional array

Implementation pitfall: cumulating rounding errors, and 0's may be small positive numbers due to rounding

LP VERSUS ILP

Problems can be much harder when we would require that solutions are integers (Integer Linear Program)

So: LP – polynomial time solvable – fast algorithms – fractional or real solutions, and

ILP – NP-hard to solve – sometimes slow – integer solutions

And we also have

Mixed LP: NP-hard to solve – sometimes slow – some variables must be integers, others may be fractional or real