Computational Geometry

d-Dimensional Linear Programming

Michael T. Goodrich

Review: 2D Linear Programming

Find values for some variables

$$x, y$$

Obey linear inequalities, called "constraints"

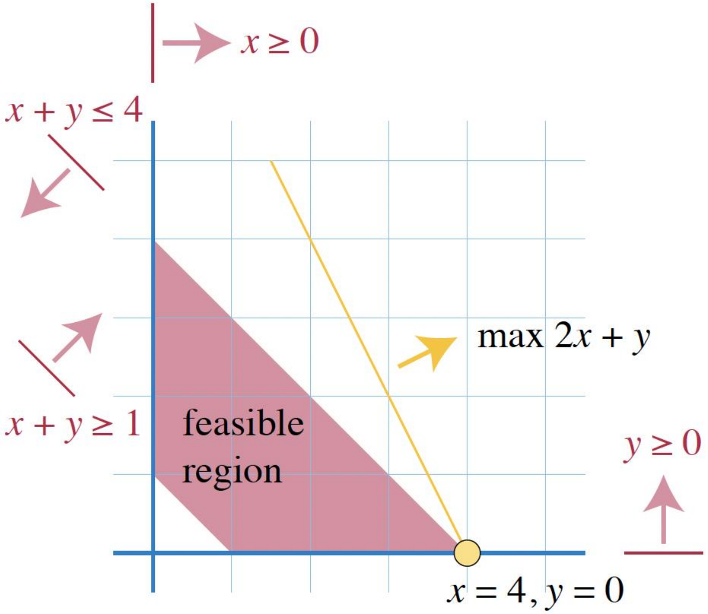
$$x \geq 0$$
$$y \geq 0$$
$$x + y \geq 1$$
$$x + y \leq 4$$

Minimize or maximize a linear "objective function"

$$\max 2x + y$$

Think of variables as coordinates

"Feasible region": convex set, points obeying constraints



Min or max is a vertex

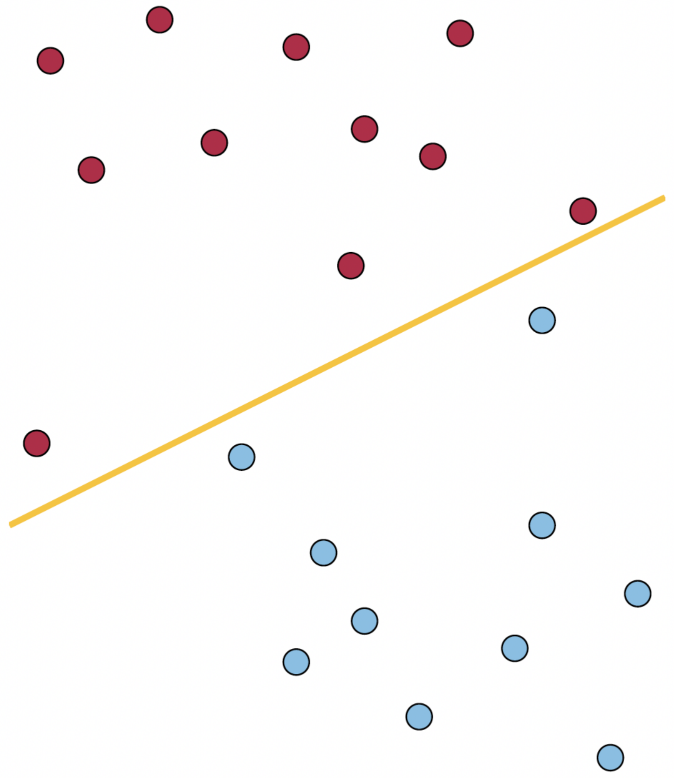
Application – Machine Learning

Given red points and blue points with coordinates (x_i, y_i)

Variables: m, b representing the line $y = mx + b$

Constraints:
$y_i \geq mx_i + b$ (for red points)
$y_i \leq mx_i + b$ (for blue points)

With one more variable, can maximize vertical distance to line $\Rightarrow$ idea behind support vector machine learning



3-Dimensional Linear Programming

Solve this linear programming problem.

$$
\begin{array}{llrcrcrcr}
\textbf{Maximize} & P = & 20x_1 & + & 10x_2 & + & 15x_3 & & \\
\textbf{Subject to:} & & 3x_1 & + & 2x_2 & + & 5x_3 & \leq & 55 \\
 & & 2x_1 & + & x_2 & + & x_3 & \leq & 26 \\
 & & x_1 & + & x_2 & + & 3x_3 & \leq & 30 \\
 & & 5x_1 & + & 2x_2 & + & 4x_3 & \leq & 57 \\
 & & x_1 & , & x_2 & , & x_3 & \geq & 0
\end{array}
$$

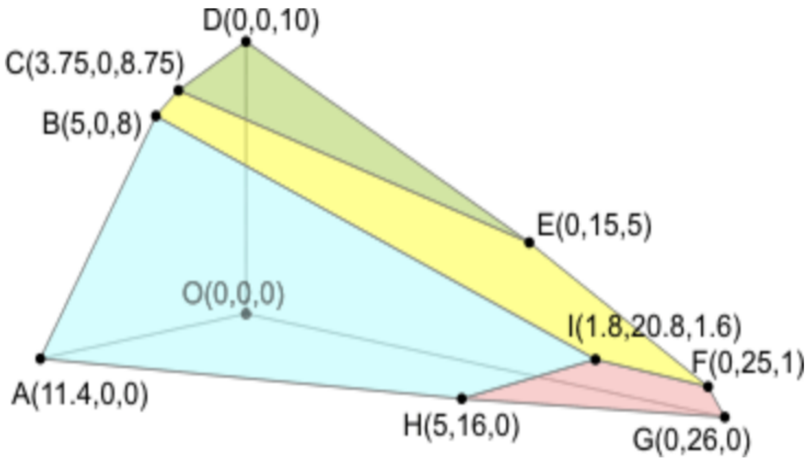
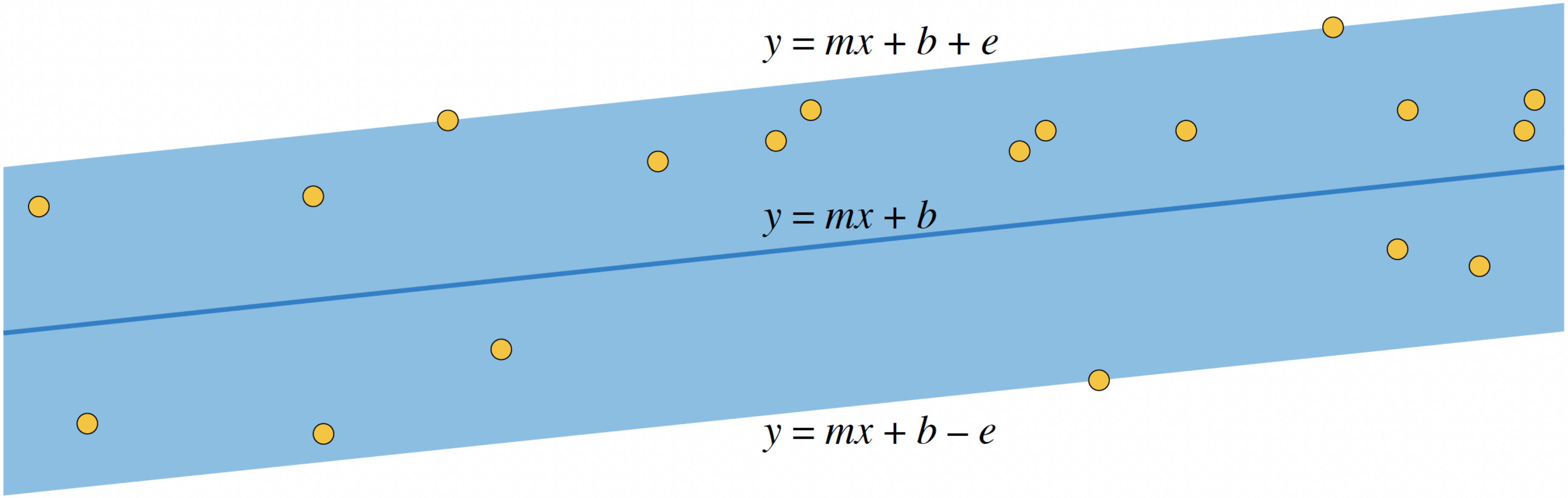


Image from https://people.richland.edu/james/ictcm/2006/3dsimplex.html

Application – Linear Regression

Regression: Fit a line $y = mx + b$ to a set of data points x_i, y_i minimizing some combination of errors $|(mx_i + b) - y_i|$

L_∞: Minimize max error; variables m, b, e,
constraints $-e \le (mx_i + b) - y_i \le e$, objective min e



More useful in metrology (how close to flat is this set of measurements of a surface) than statistics, because L_2 regression (least squares) is easier, less sensitive to outliers

Application – 3D Machine Learning

- Given red points and green points with coordinates (x_i, y_i, z_i)
- Variables: s, t, b representing the plane $z = sx + ty + b$
- Constraints:

$$z_i \geq sx_i + ty_i + b \text{ (for red points)}$$

$$z_i \leq sx_i + ty_i + b \text{ (for green points)}$$

- With one more variable (in 4D), we can maximize vertical distance to plane

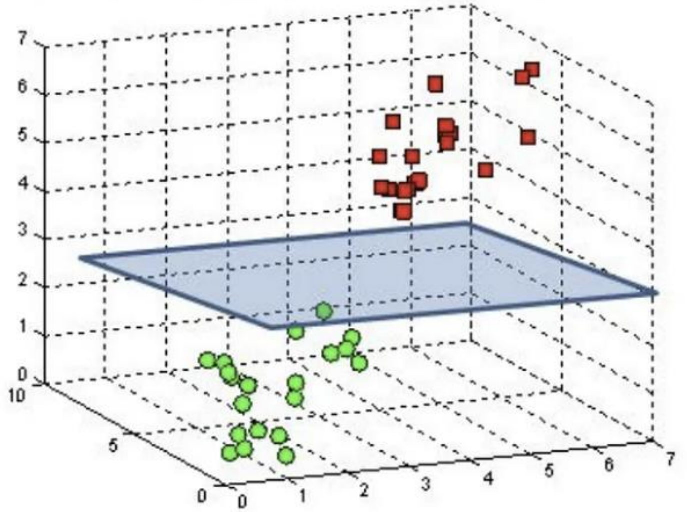


Image from https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47

Seidel's Algorithm for d-dimensional LP

To solve a d-dimensional linear program:

Randomly permute the constraints

Choose coordinates $\pm\infty$ for an optimal solution point
(whichever of $+\infty$ or $-\infty$ is better for objective function)

For each constraint $\sum a_i x_i \leq b$, in a random order:

Check whether solution point obeys the constraint

If not, solve recursively a $d-1$-dimensional LP
and replace solution point by the result

The recursive problem works in the $(d-1)$-dimensional subspace of points $\sum a_i x_i = b$, and uses the constraints that have already been added, restricted to that subspace, in a new random order

Backwards Analysis

After processing the ith constraint, what is the probability that you had to make a recursive call for it?

In any d-dimensional LP, some subset of d constraints is exactly satisfied, and determine the solution

- Solution is solution to d linear equations in d variables
- Fewer constraints $\Rightarrow$ can move solution in a linear subspace and get better in some direction
- More constraints $\Rightarrow$ some of them are redundant and not needed to determine solution

If you just made a recursive call, the last constraint you processed was one of these d constraints

Random permutation $\Rightarrow$ Happens with probability $\leq d/i$

(Can be $< d/i$ if $d > i$ or for multiple sets of d right constraints)

Expected Running Time

Let $T(d,n)$ denote the expected time to solve a d-dimensional LP with n constraints

Expected time for ith constraint: $O(d)$ to check constraint, plus
(probability of making a recursive call) $\times$ (time if we make the call)

Sum this time over all constraints:

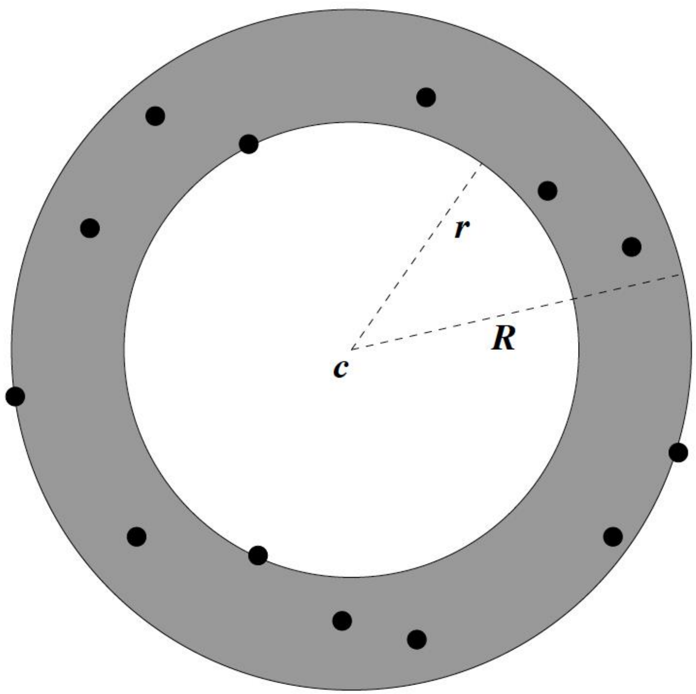
$$T(d,n) \le O(dn) + \sum_{i=1}^{n} \frac{d}{i} T(d-1, i-1)$$

Prove by induction that $T(d,n) = O(d!n)$

Induction hypothesis $\Rightarrow$ sum becomes $\sum d(d-1)!(i-1)/i < d!n$

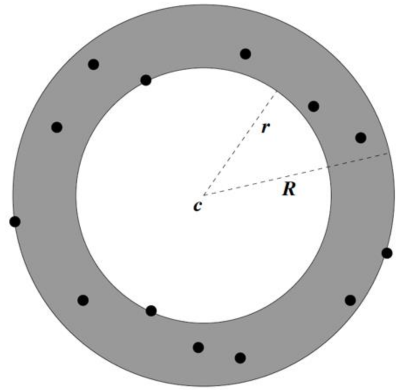
Minimum-area Enclosing Annulus

- Find the minimum-area annulus, which is defined by 2 concentric circles, such that all *n* points are between the two circles.



P = 2D point set

Minimum-area Enclosing Annulus



P = 2D point set

Let us write this as an optimization problem in the variables $c = (c_1, c_2) \in \mathbb{R}^2$ (the center) and $r, R \in \mathbb{R}$ (the small and the large radius).

$$\begin{array}{ll} \text{minimize} & \pi(R^2 - r^2) \\ \text{subject to} & r^2 \leqslant \|p - c\|^2 \leqslant R^2, \quad p \in P. \end{array}$$

(by squaring the distance)

This neither has a linear objective function nor are the constraints linear inequalities. But a variable substitution will take care of this. We define new variables

$$u := r^2 - \|c\|^2, \tag{11.3}$$

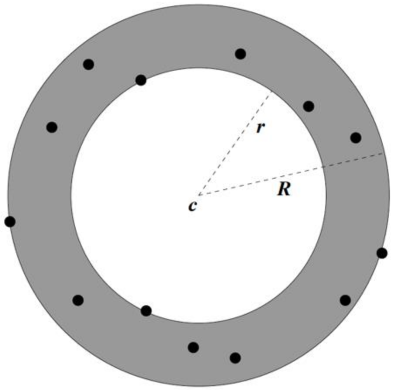
$$v := R^2 - \|c\|^2. \tag{11.4}$$

Omitting the factor π in the objective function does not affect the optimal solution (only its value), hence we can equivalently work with the objective function $v - u = R^2 - r^2$. The constraint $r^2 \leqslant \|p - c\|^2$ is equivalent to $r^2 \leqslant \|p\|^2 - 2p^\top c + \|c\|^2$, or

$$u + 2p^\top c \leqslant \|p\|^2.$$

from https://ti.inf.ethz.ch/ew/lehre/CG12/lecture/Chapter%2011%20and%2012.pdf

Minimum-area Enclosing Annulus



P = 2D point set

In the same way, $\|p - c\| \leqslant R$ turns out to be equivalent to

$$\nu + 2p^{\top}c \geqslant \|p\|^2.$$

This means, we now have a *linear* program in the variables u, ν, c_1, c_2:

$$\begin{array}{ll} \text{maximize} & u - \nu \\ \text{subject to} & u + 2p^{\top}c \leqslant \|p\|^2, \quad p \in P \\ & \nu + 2p^{\top}c \geqslant \|p\|^2, \quad p \in P. \end{array}$$

From optimal values for u, ν and c, we can also reconstruct r^2 and R^2 via (11.3) and (11.4). It cannot happen that r^2 obtained in this way is negative: since we have $r^2 \leqslant \|p - c\|^2$ for all p, we could still increase u (and hence r^2 to at least 0), which is a contradicition to $u - \nu$ being maximal.

from https://ti.inf.ethz.ch/ew/lehre/CG12/lecture/Chapter%2011%20and%2012.pdf

Reference

- Raimund Seidel. Small-dimensional linear programming and convex hulls made easy. Discrete & Computational Geometry, 6(5):423–434, 1991. doi: 10.1007/BF02574699.