Optimisation and Operations Research

Lecture 3: Linear Programming

Matthew Roughan
<matthew.roughan@adelaide.edu.au>
http://www.maths.adelaide.edu.au/matthew.roughan/notes/OORII/

School of Mathematical Sciences,
University of Adelaide

July 30, 2019

Section 1

Linear Programming

Linear equations and inequalities

We will need some tools to understand the shape of a region defined by linear inequalities

- We have n variables x_i, so $\mathbf{x} \in \mathbb{R}^n$
- We have m inequalities defining the shape of the *feasible* region

$$A\mathbf{x} \leq \mathbf{b},$$

  where
  - A is an $m \times n$ matrix
  - $\mathbf{b}$ is a length m column vector
- Also we have n non-negativity constraints $\mathbf{x} \geq 0$

Let's see what we can say/do about this region.

Note that when we use $\geq$ or $\leq$ for vectors in this course, we mean, for each element of the vector.

The Feasible Region

There are four possibilities: the feasible region could be

- equal to $\mathbb{R}^n$
  - only if there are no constraints
- empty
  - if the constraints leave no feasible points
- a subset of $\mathbb{R}^n$
  - bounded
  - unbounded

We'll need to think about this a little more, but there is a problem we do know something about: solving linear *equations*.

Converting to standard form

We will always want to put problems into the standard form above, which has

$$A\mathbf{x} \leq \mathbf{b}$$

If you have any $\geq$, then convert them to $\leq$ by multiplying the inequality by -1

Example

Replace

$$2x - 4y \geq 3$$

with

$$-2x + 4y \leq -3$$

Converting to standard form

We will always want to put problems into the standard form above, which has

$$\mathbf{x} \geq 0$$

If any x_i are free, replace by two new variables $x_i^+ \geq 0$ and $x_i^- \geq 0$ such that $x_i = x_i^+ - x_i^-$

Converting to standard form

Example

$$\begin{array}{lcccccc} \max \ z = & 3x_1 & + & 2x_2 & & \\ \text{subject to} & & & & & \\ & 2x_1 & + & x_2 & \leq & 5 \\ & x_1 & + & 3x_2 & \leq & 7 \\ & x_1 \geq 0, & & x_2 \text{ free} & & \end{array}$$

and replace it with

$$\begin{array}{lcccccccc} \max \ z = & 3x_1 & + & 2x_2^+ & - & 2x_2^- & & \\ \text{subject to} & & & & & & & \\ & 2x_1 & + & x_2^+ & - & x_2^- & \leq & 5 \\ & x_1 & + & 3x_2^+ & - & 3x_2^- & \leq & 7 \\ & x_1 \geq 0, & & x_2^+ \geq 0, & & x_2^- \geq 0 & & \end{array}$$

and when you finally obtain a solution, go back to $x_2 = x_2^+ - x_2^-$

Converting Inequalities to Equations

We want to use 1st year maths – in particular your experience with linear equations so we need to convert the inequalities into equations.

The standard process is to introduce slack variables: *e.g.,*

$$a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \leq b_1$$

becomes

$$a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n + s_1 = b_1$$

where s_1 is a *slack variable*.
By construction $s_1 \geq 0$.

Inequalities and Equations

In matrix form

$$A'\mathbf{x}' \leq \mathbf{b}$$

becomes

$$A\mathbf{x} = \mathbf{b}$$

where

$$A = \big[A' \,\big|\, I \big], \qquad \mathbf{x} = \begin{pmatrix} x'_1 \\ \vdots \\ x'_{n'} \\ s_{n'+1} \\ \vdots \\ s_{n'+m} \end{pmatrix}$$

So now there are m equations and $n = n' + m$ variables.

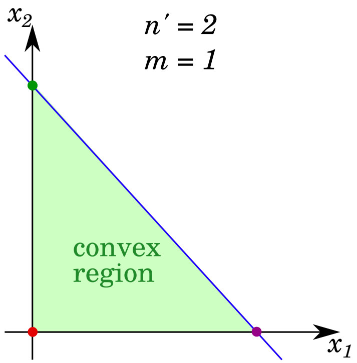
Interpretation

$$A'\mathbf{x}' \leq \mathbf{b} \quad \Leftrightarrow \quad A\mathbf{x} = \mathbf{b}$$

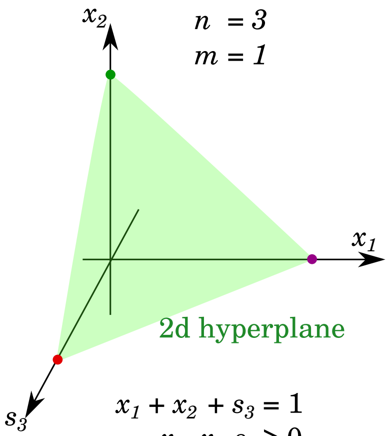
- m constraints (inequalities)
- n' variables
- A' is an $m \times n'$ matrix
- defines a region which is a convex *polytope* of $\mathbb{R}^{n'}$
- vertices are a feasible intersection point of m of the boundary planes
- non-negativity restricts to positive quadrant

- m constraints (equalities)
- $n = n' + m$ variables
- A is an $m \times n$ matrix
- defines an n' dimensional hyperplane in $\mathbb{R}^n$
- "vertices" are points where the hyperplane intersects axes
  - $n' = n - m$ of the variables are 0
- non-negativity restricts to positive quadrant

Inequalities and Equations



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



$x_1 + x_2 + s_3 = 1$
$x_1, x_2, s_3 \geq 0$

Inequalities and Equations

- We need to become expert at converting back and forth between equalities and inequalities, and their interpretation.
  - when you are on a boundary, it means
    - either a slack variable corresponding to an inequality is zero

Example

original inequality	$x + y$	$\leq$	10
slack variable equality form	$x + y + s$	$=$	10
on the boundary $s = 0$	$x + y$	$=$	10

    - or one of the $x_i = 0$
  - vertices are intersections of boundaries, so several variables must be zero, one for each boundary that's intersecting
- We also need to see how to convert a LP back and forth between equality and inequality form.

Converting from equality to inequality

Imagine we want to convert the equality

$$x + y = 10$$

Into an inequality form. We can replace the equality with

$$\begin{aligned} x + y &\leq 10 \\ x + y &\geq 10 \end{aligned}$$

Or, in standard form

$$\begin{aligned} x + y &\leq 10 \\ -x - y &\leq -10 \end{aligned}$$

Standard equality form of a LP

There are several alternatives we could end up with in all of these conversions, so we choose one standard, that we will always aim for.

Definition (Standard equality form)

An LP of the form

$$\begin{aligned} \max \quad & z = \mathbf{c}^T \mathbf{x} \\ \text{subject to} \quad & A\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

is said to be in *standard equality form*.

Standard equality form of a LP

Converting to standard form

1. Convert into standard *inequality* form
   - this isn't strictly necessary, but makes everything more consistent
2. Convert inequalities into equalities
   - introduce slack variables
   - coefficients of **c** corresponding to slack variables are 0
3. Convert to a *max* [1]
   - easy to convert by taking $\min z = \max(-z)$

Example

$$\max z = 2x_1 + 4x_2$$

is equivalent to

$$\min w = -2x_1 - 4x_2$$

[1] MATLAB's `linprog` works with *min* problems, so you need to know how to convert, backwards and forwards.

Converting a LP

Example

Let us consider the LP:

$$\begin{array}{lcccccc} \max & z = & x_1 & + & x_2 & & \\ \text{subject to} & & & & & & \\ & & x_1 & + & 2x_2 & \leq & 6 \\ & & x_1 & - & x_2 & \leq & 3 \\ & & x_1 \geq 0, & & x_2 \geq 0. & & \end{array}$$

We write this as a system of linear equations, by introducing 2 slack variables x_3 and x_4, and leaving the objective unchanged

$$\begin{array}{lccccccccc} \max z & = & x_1 & + & x_2 & & & & & \\ \text{subject to} & & x_1 & + & 2x_2 & + & x_3 & & = & 6 \\ & & x_1 & - & x_2 & & & + \; x_4 & = & 3 \\ & & & & x_1 \geq 0, & x_2 \geq 0, & x_3 \geq 0, & x_4 \geq 0. & & \end{array}$$

Note we often don't use special notation for the slack variables.

Basic feasible solution

Remember from Lecture 2

Definition (Basic solution)

A *basic solution* to $A\mathbf{x} = \mathbf{b}$ is a solution with at least $n - m$ zero variables.

We can add to this:

Definition (Basic feasible solution)

If a basic solution satisfies $\mathbf{x} \geq \mathbf{0}$, then it is called a *basic feasible solution*.

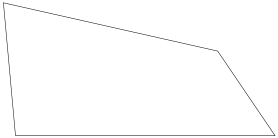
That is, the solution is feasible if it also satisfies the non-negativity requirements (for the original variables, and the slack variables).

Extreme Points

We already know that the feasible set for a LP (in inequality form) is a convex set. We can add to this

Definition (Extreme points)

We say $x \in S$ is an *extreme point* of convex set S if
$$x = \lambda y + (1-\lambda) z$$
for $y, z \in S$ and $0 < \lambda < 1$ implies $x = y = z$.



Intuitively this means that extreme points are not in the interior of any line segment inside the set.

For a convex polytope (such as defined by $A'\mathbf{x}' \le \mathbf{b}$, $\mathbf{x} \ge \underline{0}$) the extreme points are the vertices.

Geometry of linear programming solutions

Theorem

If an LP has a finite optimum, it has an optimum at an extreme point of the feasible set.

For LP problems the feasible set will always have a finite number of extreme points (vertices). This suggests a naïve algorithm:

1. Find all the vertices of the feasible set
2. Choose the optimum.

How do we find vertices?

Theorem

The basic feasible solutions of $A\mathbf{x} = \mathbf{b}$ *are the extreme points of the feasible set* $A'\mathbf{x}' \leq \mathbf{b}$, $\mathbf{x} \geq \underline{0}$.

Unfortunately, we can not typically identify which basic solutions will be feasible *a priori* (*i.e.*, which are vertices)

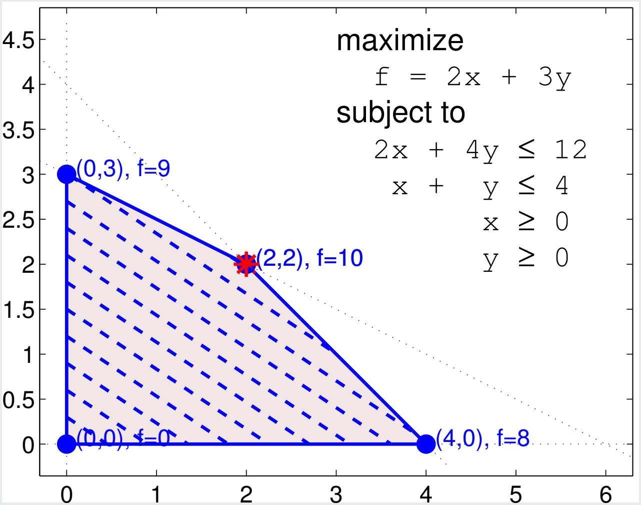
New naïve algorithm

1. Find all the basic solutions
2. Test to see if they are feasible
3. Choose the optimum of the basic feasible solutions

Somehow we also need to check for boundedness at the same time.

Example

Example (cont.)



Section 2

Rank

Linear equations and redundancy

Given $A\mathbf{x} = \mathbf{b}$, where A is of size $m \times n$

- n variables (unknowns)
- m equations (pieces of information)

Naïvely we have m pieces of information, and n variables, so we might expect a unique solution for $m \geq n$. However, some pieces of information (equations) might be redundant.

- examples
  - if two rows are the same, then they don't provide any extra information
  - if one row is a scalar multiple of another, then it doesn't provide any extra information, *e.g.*,

$$\begin{aligned} x_1 + x_2 &= 3 \\ 2x_1 + 2x_2 &= 6 \end{aligned}$$

- how do we assess redundancy in general?

Rank

Definition (Rank)

The rank of a matrix A is the number of linearly-independent rows[a]*.*

A matrix is full-rank if it has the maximum rank for its size (the minimum of n and m for an $m \times n$ matrix).

[a] There are actually several equivalent definitions of rank.

- Note this is defined for any matrix: so we can use it for
  - $rank(A)$ – the rank of the constraint coefficient matrix
  - $rank(M)$ – the rank of the augmented matrix $M = [A \mid \mathbf{b}]$
- they won't necessarily have the same rank, but

$$rank(A) \leq rank(M)$$

because when we add a column the rank either increases by 1, or stays the same.

Rank example

Example

$$A = \begin{bmatrix} 1 & 2 & 1 \\ -2 & -3 & 1 \\ 3 & 5 & 0 \end{bmatrix}$$

The $rank(A) = 2$

- $R_3 = R_1 - R_2$ so the 3 rows are linearly dependent
- any pair of two rows is linearly independent

Now add columns $M = [A \mid \mathbf{b}]$

$$M_1 = \begin{bmatrix} 1 & 2 & 1 & 2 \\ -2 & -3 & 1 & 4 \\ 3 & 5 & 0 & -1 \end{bmatrix} \text{ and } M_2 = \begin{bmatrix} 1 & 2 & 1 & 2 \\ -2 & -3 & 1 & 3 \\ 3 & 5 & 0 & -1 \end{bmatrix}$$

Then $rank(M_1) = 3$ and $rank(M_2) = 2$

Ranks and numbers of solutions

Theorem (Rouché-Capelli)

A system of linear equations $A\mathbf{x} = \mathbf{b}$ *has a solution iff the ranks of its coefficient matrix* A *and its augmented matrix* $M = [A|\mathbf{b}]$ *are equal.*

If there are solutions, they form an affine subspace[a] *of* $\mathbb{R}^n$ *of dimension* $n - rank(A)$ *where there are* n *variables.*

[a] Here this is just a linear subspace translated away from the origin.

Examples:

- $rank(A) = rank([A|\mathbf{b}]) = n$, there will be a unique solution
- $rank(A) = rank([A|\mathbf{b}]) < n$, infinite solutions
- $rank(A) \neq rank([A|\mathbf{b}])$, no solutions

Full-rank

- If the matrix A is less than full rank, and the equations are consistent (there is at least one solution) we can reduce the set of equations down to a new set of full rank
  - some LP pre-processors do just this
- When we go from $A'\mathbf{x}' \leq \mathbf{b}$ to $A\mathbf{x} = \mathbf{b}$ we add m slack variables, and an identity matrix to A, *i.e.*, $A = [A' \mid I_m]$. Even if all of the rows of A' were linearly dependent (*i.e.*, $rank(A') = 1$), adding the identity would lead to $rank(A) = m$, *i.e.*, A is full rank.
- For what follows, we shall usually assume we start with a full rank coefficient matrix A
  - if not, algorithms often still work, but may be inefficient

For us

Take $A\mathbf{x} = \mathbf{b}$, where A is of size $m \times n$

- n variables
- m equations
- $n \leq m$

with full-rank A

- $rank(A) = \min(m, n) = n$
- $rank([A|\mathbf{b}]) = n$ or $n + 1$
  - could be 1 or 0 solutions
- 0 or 1 solutions doesn't leave much room for optimisation, so we will usually consider the case $n > m$
  - this always happens when we start with $A'\mathbf{x} \leq \mathbf{b}$, because we add m slack variables

Solutions

Take $A\mathbf{x} = \mathbf{b}$, where A is of size $m \times n$

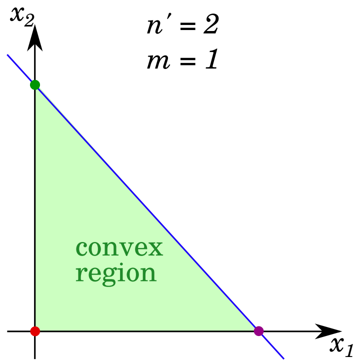
- n variables
- m equations

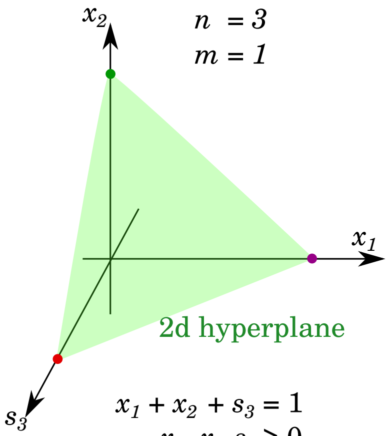
with full rank A, and $n > m$:

- The dimension of the solution subspace is $n - m$
  - the solution is a $n - m$ dimensional hyperplane
  - we can describe the hyperplane by where it intersects the axes (*i.e.*, places where $x_i = 0$ for some set of i)
- Choose $(n - m)$ variables to be 0
  - then there will be a unique solution for the other m variables
    - ★ think of it as adding $n - m$ additional equations
    - ★ or we reduce the number of variables down to m
  - the possible number of choices is

$$\binom{n}{n-m} = \binom{n}{m}$$

Numbers of solutions





$$x_1 + x_2 \le 1 \qquad x_1, x_2 \ge 0$$

$$x_1 + x_2 + s_3 = 1 \qquad x_1, x_2, s_3 \ge 0$$

3 vertices $\Leftrightarrow$ $\binom{n}{m} = 3$ basic solutions

Section 3

Naïve solution

Exhaustive Search

- The above suggests that we could perform optimisation via an exhaustive search
  - look for all the vertices
  - pick the best
- This would be very computationally challenging

Naïve Optimisation

Explore all possible vertices

1. construct all vertices:
    1. each construct requires at least one pivot: $O(nm)$ operations
    2. loose bound on basic solutions $\binom{n}{m}$
        - ★ Stirling' approximation makes it $O(m^n)$
    3. check basic solution is feasible
        - ★ requires comparison of m values
2. for each vertex you have to compute z which is $O(n)$, but this is small compared to performing the pivot.

So the total (worst-case) complexity is $O\big(nmm^n\big)$
This grows VERY VERY VERY quickly, *e.g.*,

- $n = 10$, $m = 10$, this would be around 10^{12}
- $n = 20$, $m = 20$, this would be around 10^{28}

Hirsch conjecture

- We think we can do MUCH better
  Hirsch conjecture says that any two vertices of the polytope must be connected to each other by a path of length at most $m - n$.
- Maybe we could get the optimal value in $m - n$ steps (pivots), if we were really smart?
  - actually the conjecture is wrong, but doesn't rule out $O(m)$ performance

More on computational complexity later

Takeaways

- Mapping between inequalities and equalities
- Solutions at vertices
    1. but naïvely looking for them is VERY inefficient

Further reading I