Convex Optimization
CMU-10725

3. Linear Programs

Barnabás Póczos & Ryan Tibshirani

MACHINE LEARNING DEPARTMENT
Carnegie Mellon.
School of Computer Science

Administrivia

- **Please ask questions!**
- Slides: http://www.stat.cmu.edu/~ryantibs/convexopt/
- Anonym feedback survey will be on black board today. **Please use it!** Constructive feedback and suggestions are always welcome!

Basic Definitions

- More and more complicated optimization problems
- Definition of LP

Simplest Optimization Problems

Goal:

- Constant function
- 1-dim linear function
- 1-dim linear function with bound constraints

Linear Programs

- n-dim linear function with m linear constraints

Inequality form:

Cost function:

Constraints:

Bounds:

Linear Programs

Inequality form using matrix notation:

Cost function:

Constraints:

Bounds:

Example:

Goal of this (...and next) lecture(s)

- **To be able to solve Linear Programs**

  Simplex Algorithm (Phase I and Phase II)
  (Later we will see other algorithms too)

- **Understand why LP is useful**
  - Motivation
  - Applications in Machine Learning

- **Understand the difficulties**
  - Convergence? Polynomial or Exponential many operations?
  - Will algorithms find the exact solutions, or only approximate ones?

Table of Contents

- **Motivating Examples & Applications:**
  - Pattern classification
- **Linear programs**:
  - standard form
  - canonical form
- **Solutions:**
  - Basic, Feasible, Optimal, Degenerate
- **Simplex algorithm:**
  - Phase I
  - Phase II

Linear Programs

- Motivation
- History
- Sketching LP

History

Dantzig 1947 (Simplex method)

(one of the top 10 algorithms of the twentieth century**)**

Motivated by World War II:

- Job scheduling (Assign 70 men to 70 jobs)
- Blending problem
  (produce a blend (30% Lead, 30% Zinc, 40% Tin) out of 9 different alloys (different mixture, different costs) such that the cost is minimal)
- Network flow optimization (Max flow min cut)

The product mix problem

A furniture company manufactures four models of desks

Number of man hours and profit:

	Desk 1	Desk 2	Desk 3	Desk 4	Available hrs
Carpentry shop hrs	4	9	7	10	6000
Finishing shop hrs	1	1	3	40	4000
Profit	$12	$20	$18	$40	

Why is it called Linear Programing???

Motivation: Why Linear Programing?

- The simplest, nontrivial optimization problem

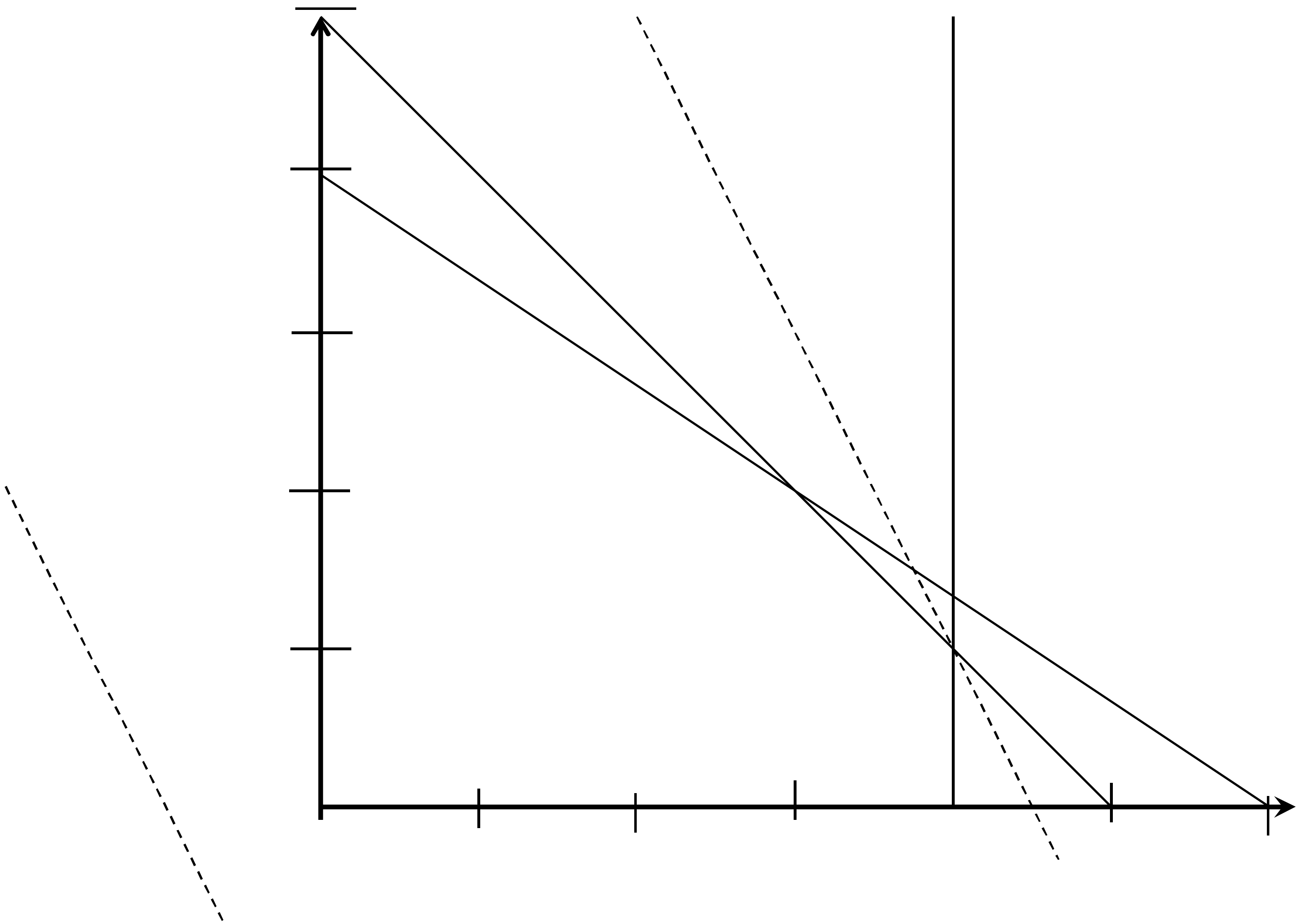
- Many complex system (objective and constraints) can be well approximated with linear equations

- Important applications

- There are efficient toolboxes that can solve LPs

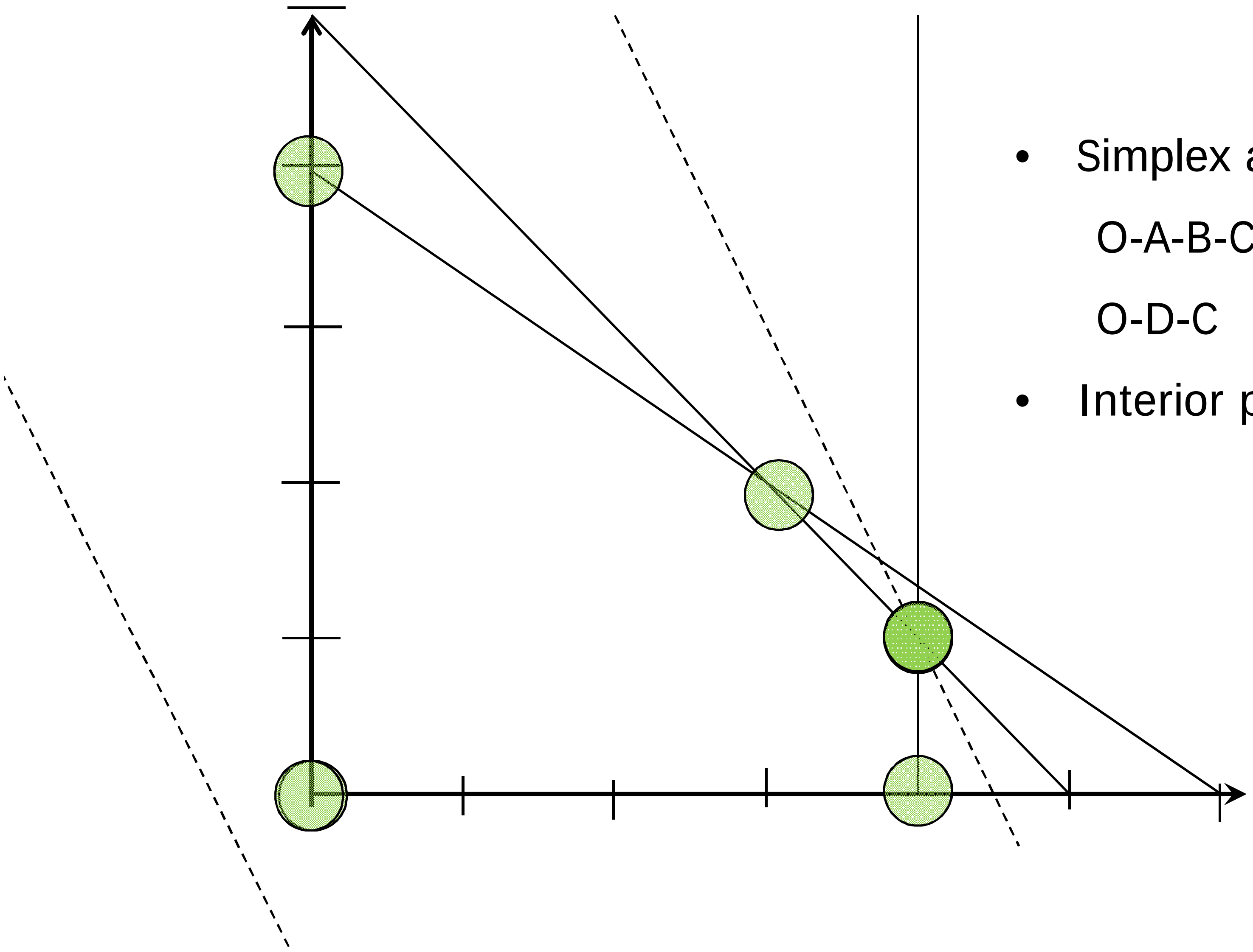
Sketching Linear Programs

Example:



Simplex Algorithm

Example:



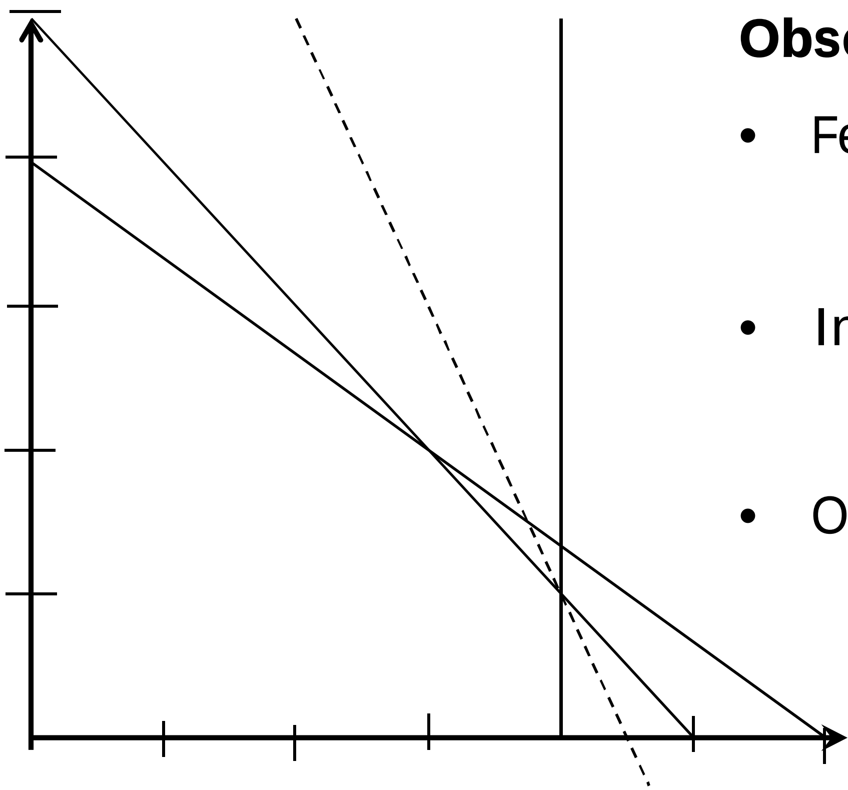
- Simplex algorithm:

  O-A-B-C

  O-D-C

- Interior point methods

Linear Program



Observations, Difficulties:

- Feasible set might not exist, no solution

  (Inconsistency in the constraints)

- Infinite many global optimum

  (Optimum is on an edge)

- Optimum can be – 1 , 1

  (Unbounded optimum)

Linear Program

High dimensional case is similar:

faces, facets instead of edges

cost function = hyperplane

Applications

Pattern Classification via Linear Programming

Application

Pattern Classification via Linear Programming

More info can be found on: cgm.cs.mcgill.ca/~beezer/cs644/main.html

Goal: show how LP can be used for linear classification.

Why LP?

There are many efficient LP solver software packages

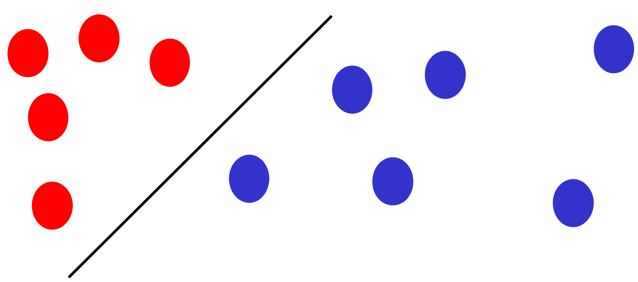
Pattern Classification via LP

Formal goal:

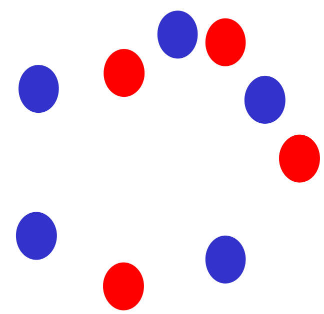
Problem 1: Determine whether H and M are linearly separable

Problem 2: If H and M are linearly separable,
then find a separating hyper plane

Linearly separable sets:

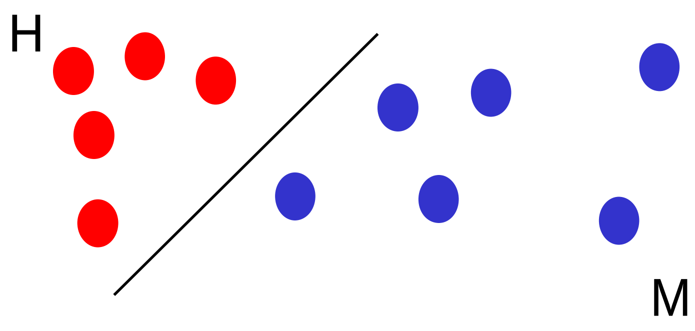


Linearly not separable sets:

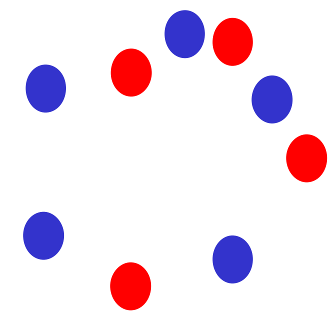


Pattern Classification via LP

Linearly separable sets:



Linearly not separable sets:



Observation:

H and M are linearly separable

Pattern Classification via LP

Lemma 1:

H and M are linearly separable

Proof

Pattern Classification via LP

Lemma 1:

H and M are linearly separable

Proof

Pattern Classification via LP

Proof continued

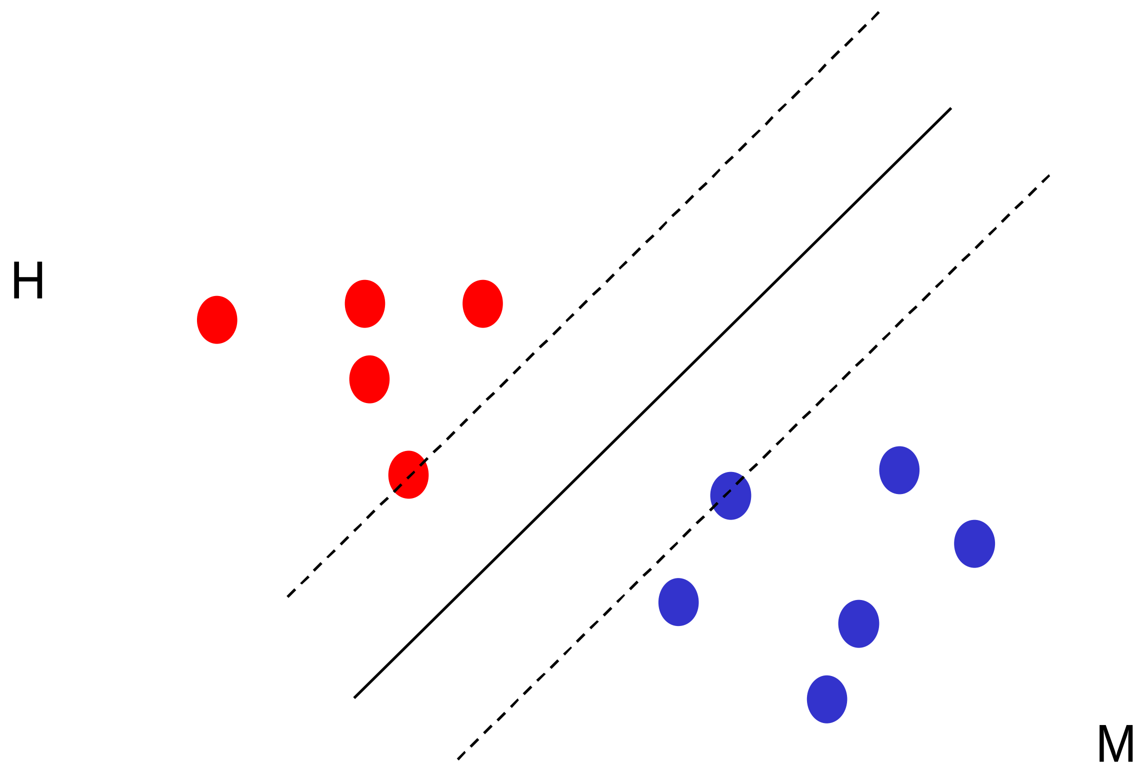
Pattern Classification via LP

Proof continued

Similarly,

Pattern Classification via LP

Proof continued



Pattern Classification via LP

We will see that the following linear problem solves Problem 1 & 2:

[Mansgarian 1995]

Pattern Classification via LP

Theorem 1

H and M are linearly separable iff the optimal value of LP is 0.

Theorem 2

H and M are linearly separable

y^*, z^*, a^*, b^* is an optimal solution of (LP)

$\Rightarrow$ $f(x)=a^{*T}x+b^*$ is a separating hyperplane

Pattern Classification via LP

Proof of Theorems 1 and 2

The optimal value of (LP) is 0

Pattern Classification via LP

Application: Breast Cancer Diagnosis

Used at the University of Wisconsin Hospital

[Mangasarian et al 1995]

1. Fluid sample from breast.
2. Placed on a glass and stained the highlight the nuclei of cells
3. Image is taken
4. 30D features: Area, Radius, perimeter, etc

Goal: Classification between benign lumps and malignant lumps

Results: 97.5% accuracy

Pattern Classification via LP

Example 1: Linearly Separable Case

Pattern Classification via LP

Example 2: Linearly nonseparable case

Linear Programs

- Standard from, Canonical form, Inequality form
- Transforming LPS
  - Pivot transformation

Linear Programs

Inequality form of LPs using matrix notation:

Standard form of LPs:

Theorem: Any LP can be rewritten to an equivalent standard LP

Transforming LPs

Theorem: Any LP can be rewritten to an equivalent standard LP

- Getting rid of inequalities (except variable bounds)

- Getting rid of equalities

Transforming LPs

- Getting rid of negative variables

- Getting rid of bounded variables

- Max to Min

- Negative b_i

From Inequality Form to Standard Form

Inequality form

max 2x+3y s.t.

- x + y · 4
- 2x + 5y · 12
- x + 2y · 5
- x, y ¸ 0

if std fm has n vars, m eqns,

then ineq form has

n–m vars and m+(n–m)=n ineqs

(here m = 3, n = 5)

Standard form

max 2x+3y s.t.

- x + y + u = 4
- 2x + 5y + v = 12
- x + 2y + w = 5
- x, y, u, v, w ¸ 0

Linear Programing 2

Pivot Transformation

Consider the following problem

Definition: [Pivot]

- Choose a nonzero element, e.g. $3X_4$
- Use this to eliminate X_4 from the remaining equations
- = Gauss elimination

Pivot Transformation

Pivot Transformation

After pivot we got an equivalent system: The solution set is the same.

If we pivot again, say in $X_2/3$, then

Let us rewrite this:

Canonical Form

Definition [canonical form**]**

- (*) is in canonical form w.r.t (-Z), X_4, X_2 variables
- X_1, X_3, X_5 = Independent (Nonbasic) variables
- -Z, X_4, X_2 = Dependent (Basic) variables.

  They are expressed with other variables

Canonical Form

- X_1, X_3, X_5 = Independent (Nonbasic) variables
- $-Z$, X_4, X_2 = Dependent (Basic) variables.

If we set the nonbasics to zero, then we get values for the basic variables:

However, if X_1 and X_4 had been chose for pivoting, then

Canonical Form

Goal of pivots: reduce the original LP problem to canonical form

From canonical form it is easy to find a (basic) solution:

(we just need to set the nonbasic variables to zero)

This basic solution might be

- ❑ not feasible (because of the boundary constraint! We have to have $X_i \geq 0$)
- ❑ not optimal (i.e. Z is not minimal)

Pivoting does not alter the solution set.
(After pivots the systems are equivalent)

Canonical Form

Formal definition of canonical form:

A system of m equations and n variables

is in canonical form w.r.t

Example

Canonical Form

Canonical form:

Definition: [Basic solution]

Example

Warming up for the Simplex Algorithm

How to solve LPs if we already have

a canonical form with basic feasible solution?

Simplex Algorithm Phase II

Starting from Canonical Form

Assume that we have a canonical form with feasible basic solution

Using matrix notation:

In this canonical form the basic solution is:

Improving a Nonoptimal Basic Solution

Let us continue the example

Basic feasible solution:

Goal: min Z, s.t. $X_i \geq 0$

- ❑ The relative cost factor of X_3 is (-5)<0
- ❑ **Let us see if we can change X_3 from zero to decrease Z**

Improving a Nonoptimal Basic Solution

- The relative cost factor of X_3 is $(-5)<0$

- **Keep X_3 , and $X_B=(-Z, X_2, X_4)$ as parameters. $(X_1,X_5)=(0,0)$**

X_i ¸ 0, so we can decrease Z by changing

those X_i components which have $C_i<0$

relative cost factors!

We can decrease Z by increasing X_3 from 0,

as long as

Improving a Nonoptimal Basic Solution

X ¸ 0, so we can decrease Z by changing

those X_i components which have C_i<0

relative cost factors

We can decrease Z by increasing X_3 from 0,

as long as

Improving a Nonoptimal Basic Solution

What just happened?

- We brought X_3 into X_B.
- Either X_2 or X_4 can go out into X_N
- We chose X_4 to go out, because that minimizes Z the most
- This is the same as making a pivot on $3X_3$ in (*)

Improving a Nonoptimal Basic Solution

Improving a Nonoptimal Basic Solution

Improving a Nonoptimal Basic Solution

Improving a Nonoptimal Basic Solution

The Simplex Algorithm (Phase II)

Key components of the simplex algorithm

1. Optimality test

2. In each step one variable in, one variable out
   (Traveling on the neighboring corners of the polytope)

3. The adjusted values have to be nonnegative

The Simplex Algorithm (Phase 2)

Assume that we start from a **feasible canonical form**:

The initial feasible solution is:

Steps of the Simplex algorithm

(1) Smallest reduced cost

The Simplex Algorithm

Steps of the Simplex algorithm

(2) Test for optimality

(3) Incoming variable

(4) Test for unbounded Z

The Simplex Algorithm

Steps of the Simplex algorithm

(5) Outgoing variable

- This r will show the outgoing variable
- The basic variable in the r^{th} row of A

Lemma [New basic solution remains feasible**]**

Proof

The Simplex Algorithm

Steps of the Simplex algorithm

(6) Pivot on A_{rs}

- This gives us new basic feasible solution
- We do this pivot regardless if Z changes or not

The Simplex Algorithm

Steps of the Simplex algorithm

- If zero change in the objective Z, then cycling can happen
- Bland rule can avoid cycling

Bland's rule: Whenever the pivot in the simplex method would result in a zero change of the objective Z, do the following:

(i) Incoming column:

(ii) Outgoing column:

The Simplex Algorithm Summary

Theorem:

A basic feasible solution is optimal with total cost Z_0,

if all relative cost factors (C_j, j=1,...,n) are nonnegative.

Proof:

Theorem:

A basic feasible solution is the **unique** optimal solution with total cost Z_0,

if $C_j>0$ for all nonbasic variables.

The Simplex Algorithm Summary

Theorem:

Assuming "non-degeneracy" at each iteration (b_j>0, j=1,...,m),

the simplex algorithm will converge in finite steps.

Proof:

There are only finite many basis, and because of "non-degeneracy",

cycling cannot happen.

Remark:

- If we use infinite-precision arithmetic, then we can find the exact solution. (No approximation used)
- Interior point methods can only converge to an epsilon ball that contains the solution.

The full Simplex Algorithm

So far we have assumed that a basic feasible solution in canonical form is available to start the algorithm...

The Simplex Algorithm Summary

The simplex method can be applied to a Linear Program in standard form:

Phase I:

- Find a starting basic feasible solution in canonical form

  and detect redundancies

- or determine if such solution doesn't exist

  detect inconsistencies

Phase II:

If starting basic solution found, then

- find an optimal solution
- or show that $Z \to -1$ is possible

The Simplex Algorithm Phase I

Example

Goal: We want to find a feasible solution

Phase I:

(i) Forget the cost function $c^T x$.

(ii) Introduce $X_6, X_7 \geq 0$. [One variable for each row]

(iii) Solve

The Simplex Algorithm Phase I

Theorem: (*2) has feasible optimal solution such that $X_6=X_7=0$

iff (*1) has feasible solution

Remarks:

- (*2) is easy to convert to a feasible canonical solution (We will see)
- We can find its optimal solution ($X_6=X_7=0$) with the Phase II algorithm This is a feasible solution of (*1)

Tableaux

Basic variable	Objective (-w)	X_1	X_2	X_3	X_4	X_5	X_6	X_7	RHS
	1	0	0	0	0	0	1	1	0
	0	4	2	13	3	1	1	0	17
	0	1	1	5	1	1	0	1	7

It's easy to convert this to a feasible canonical form:

Basic variable	Objective (-w)	X_1	X_2	X_3	X_4	X_5	X_6	X_7	RHS
-w	1	-5	-3	-18	-4	-2	0	0	-24
X_6	0	4	2	13	3	1	1	0	17
X_7	0	1	1	5	1	1	0	1	7

Tableaux

Now, we can run the Phase 2 algorithm on this table to get a feasible solution of (*1):

B. var	**Obj. (-w)**	X_1	X_2	X_3	X_4	X_5	X_6	X_7	**RHS**
-w	1	-5	-3	-18	-4	-2	0	0	-24
X_6	0	4	2	13	3	1	1	0	17
X_7	0	1	1	5	1	1	0	1	7

Tableaux

Now, we can run the Phase 2 algorithm on this table to get a feasible solution of (*1):

B. var	Obj. (-w)	X_1	X_2	X_3	X_4	X_5	X_6	X_7	RHS
-w	1	-5	-3	-18	-4	-2	0	0	-24
X_6	0	4	2	13	3	1	1	0	17
X_7	0	1	1	5	1	1	0	1	7

B. var	Obj (-w)	X_1	X_2	X_3	X_4	X_5	X_6	X_7	RHS
-w	1	-5+18/13*4	-3+18/13*2	0	-4+18/13*3	-2+18/13	18/13	0	-24+18/13*7
X_3	0	4/13	2/13	1	3/13	1/13	1/13	0	17/13
X_7	0	1-5/13*4	1-5/13*2	0	1-5/13*3	1-5/13	-5/13	1	7-5/13*7

Tableaux

B. var	Obj (-w)	X_1	X_2	X_3	X_4	X_5	X_6	X_7	RHS
-w	1	-5+18/13*4	-3+18/13*2	0	-4+18/13*3	-2+18/13	18/13	0	-24+18/13*7
X_3	0	4/13	2/13	1	3/13	1/13	1/13	0	17/13
X_7	0	1-5/13*4	1-5/13*2	0	1-5/13*3	1-5/13	-5/13	1	7-5/13*7

Let us simplify this Table a little:

B. var	Obj (-w)	X_1	X_2	X_3	X_4	X_5	X_6	X_7	RHS
-w	1	-7/13	-3/13	0	2/13	-8/13	18/13	0	-6/13
X_3	0	4/13	2/13	1	3/13	1/13	1/13	0	17/13
X_7	0	1-7/13	3/13	0	-2/13	8/13	-5/13	1	6/13

Tableaux

B. var	Obj (-w)	X_1	X_2	X_3	X_4	X_5	X_6	X_7	RHS
-w	1	0	0	0	0	0	1	1	0
X_3	0	3/8	1/8	1	1/4	0	-1/8	0	5/4
X_5	0	-7/8	3/8	0	-1/4	1	-5/8	1/8	3/4

All the relative costs are nonnegative) optimal feasible solution.

Phase I is finished.

Tableaux

B. var	Obj (-w)	X_1	X_2	X_3	X_4	X_5	X_6	X_7	RHS
-w	1	0	0	0	0	0	1	1	0
X_3	0	3/8	1/8	1	1/4	0	-1/8	0	5/4
X_5	0	-7/8	3/8	0	-1/4	1	-5/8	1/8	3/4

B. var	Obj (-z)	X_1	X_2	X_3	X_4	X_5	X_6	X_7	RHS
-z	1	2	1	2	1	4	0	0	0
X_3	0	3/8	1/8	1	1/4	0	-1/8	0	5/4
X_5	0	-7/8	3/8	0	-1/4	1	-5/8	1/8	3/4

Tableaux

B. var	Obj (-z)	X_1	X_2	X_3	X_4	X_5	X_6	X_7	RHS
-z	1	2	1	2	1	4	0	0	0
X_3	0	3/8	1/8	1	1/4	0	-1/8	0	5/4
X_5	0	-7/8	3/8	0	-1/4	1	-5/8	1/8	3/4

Make it to canonical form and continue with Phase II...

Do not make pivots in the column of X_6 and X_7

Simplex Algorithm with Matlab

```
f = [-5  -4  -6]';

A =  [ 1 -1  1
       3  2  4
       3  2  0 ];

b = [20  42 30]';

lb = zeros(3,1);

options = optimset('LargeScale','off','Simplex','on');

[x,fval,exitflag,output,lambda] = linprog(f,A,b,[],[],lb,[],[],options);
```

Relevant Books

- Luenberger, David G. *Linear and Nonlinear Programming*. 2nd ed. Reading, MA: Addison Wesley, 1984. ISBN: 0201157942.
- Bertsimas, Dimitris, and John Tsitsiklis. *Introduction to Linear Optimization*. Athena Scientific Press, 1997. ISBN: 1886529191.
- Dantzig, Thapa: Linear Programming

Summary

- Linear programs:
  - standard form,
  - canonical form
- Solutions:
  - Basic, Feasible, Optimal, Degenerate
- Simplex algorithm:
  - Phase I
  - Phase II
- Applications:
  - Pattern classification