15-853:Algorithms in the Real World

Linear and Integer Programming I

- Introduction
- Geometric Interpretation
- Simplex Method

Linear and Integer Programming

Linear or Integer programming

minimize $z = c^T x$ **cost** or **objective** function
subject to $Ax \le b$ **equalities**
$x \ge 0$ **inequalities**
$c \in R^n$, $b \in R^m$, $A \in R^{n \times m}$

Linear programming:
$x \in R^n$ (polynomial time)

Integer programming:
$x \in Z^n$ (NP-complete)

Extremely general framework, especially IP

Related Optimization Problems

Unconstrained optimization
$\min\{f(x) : x \in R^n\}$

Constrained optimization
$\min\{f(x) : c_i(x) \le 0, i \in I, c_j(x) = 0, j \in E\}$

Quadratic programming
$\min\{1/2x^TQx + c^Tx : a_i^Tx \le b_i, i \in I, a_i^Tx = b_j, j \in E\}$

Zero-One programming
$\min\{c^Tx : Ax = b, x \in \{0,1\}^n, c \in R^n, b \in R^m\}$

Mixed Integer Programming
$\min\{c^Tx : Ax \le b, x \ge 0, x_i \in Z^n, i \in I, x_r \in R^n, r \in R\}$

How important is optimization?

- 50+ packages available
- 1300+ papers just on interior-point methods
- 100+ books in the library
- 10+ courses at most Universities
- 100s of companies
- All major airlines, delivery companies, trucking companies, manufacturers, …
  make serious use of optimization.

Linear+Integer Programming Outline

Linear Programming

- General formulation and geometric interpretation
- Simplex method
- Ellipsoid method
- Interior point methods

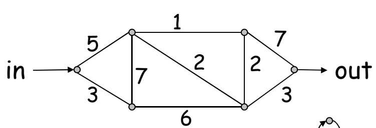
Integer Programming

- Various reductions of NP hard problems
- Linear programming approximations
- Branch-and-bound + cutting-plane techniques
- Case study from Delta Airlines

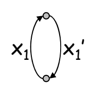
Applications of Linear Programming

1. A substep in most integer and mixed-integer linear programming (MIP) methods
2. Selecting a mix: oil mixtures, portfolio selection
3. Distribution: how much of a commodity should be distributed to different locations.
4. Allocation: how much of a resource should be allocated to different tasks
5. Network Flows

Linear Programming for Max-Flow

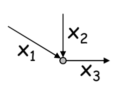


Create two variables per edge:



Create one equality per vertex:

$x_1 + x_2 + x_3' = x_1' + x_2' + x_3$



and two inequalities per edge:

$x_1 \leq 3$, $x_1' \leq 3$

add edge x_0 from **out** to **in**

maximize x_0

In Practice

In the "real world" most problems involve at least some integral constraints.

- Many resources are integral
- Can be used to model yes/no decisions (0-1 variables)

Therefore "1. A subset in integer or MIP programming" is the most common use in practice

Algorithms for Linear Programming

- **Simplex** (Dantzig 1947)
- **Ellipsoid** (Kachian 1979)
  first algorithm known to be **polynomial time**
- **Interior Point**
  first practical polynomial-time algorithms
  - **Projective method** (Karmakar 1984)
  - **Affine Method** (Dikin 1967)
  - **Log-Barrirer Methods** (Frisch 1977, Fiacco 1968, Gill et.al. 1986)

Many of the interior point methods can be applied to nonlinear programs. Not known to be poly. time

State of the art

1 million variables

10 million nonzeros

No clear winner between Simplex and Interior Point

- Depends on the problem
- Interior point methods are subsuming more and more cases
- All major packages supply both

The truth: the sparse matrix routines, make or break both methods.

The best packages are highly sophisticated.

Comparisons, 1994

problem	Simplex (primal)	Simplex (dual)	Barrier + crossover
binpacking	29.5	62.8	560.6
distribution	18,568.0	won't run	too big
forestry	1,354.2	1,911.4	2,348.0
maintenace	57,916.3	89,890.9	3,240.8
crew	7,182.6	16,172.2	1,264.2
airfleet	71,292.5	108,015.0	37,627.3
energy	3,091.1	1,943.8	858.0
4color	45,870.2	won't run	too big

Formulations

There are many ways to formulate linear programs:

- **objective (or cost) function**
  maximize c^Tx, or
  minimize c^Tx, or
  find any feasible solution
- **(in)equalities**
  $Ax \leq b$, or
  $Ax \geq b$, or
  $Ax = b$, or any combination
- **nonnegative variables**
  $x \geq 0$, or not

Fortunately it is pretty easy to convert among forms

Formulations

The two **most common** formulations:

Canonical form		Standard form
minimize $c^T x$ subject to $Ax \geq b$ $x \geq 0$	**slack** variables →	minimize $c^T x$ subject to $Ax = b$ $x \geq 0$

e.g.

$7x_1 + 5x_2 \geq 7$ $x_1, x_2 \geq 0$	y_1 →	$7x_1 + 5x_2 - y_1 = 7$ $x_1, x_2, y_1 \geq 0$

More on slack variables later.

Geometric View of Canonical Form

A **polytope** in n-dimensional space

- Each inequality corresponds to a half-space.
- The "feasible set" is the intersection of the half-spaces
- This corresponds to a polytope
- Polytopes are **convex**: if x,y is in the polytope, so is the line segment joining them.
- The optimal solution is at a vertex (i.e., a corner).

Geometric View of Canonical Form

minimize:

$z = -2x_1 - 3x_2$

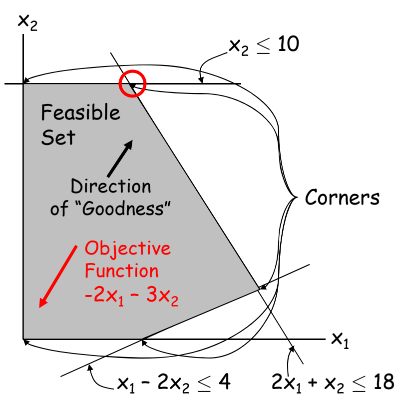
subject to:

$x_1 - 2x_2 \leq 4$

$2x_1 + x_2 \leq 18$

$x_2 \leq 10$

$x_1, x_2 \geq 0$



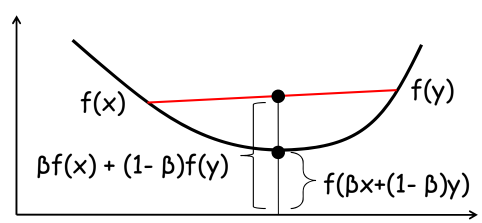
An intersection of 5 halfspaces

Optimum is at a Vertex

Holds even if the objective function is merely convex:

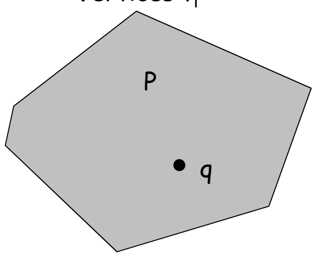
f is convex if for all vectors x, y in S and β in [0,1]

f(βx+(1- β)y) <= βf(x) + (1- β)f(y)



Optimum is at a Vertex

Vertices v_i



If f is linear, then
$f(q) = f(\sum \beta_i v_i) = \sum \beta_i f(v_i)$ so
$\min_i f(v_i) \leq f(q) \leq \max_i f(v_i)$

Every point q in P is a convex combination of vertices of P.
There exist β_i
$q = \sum \beta_i v_i$

Fix convex f, then
$f(q) = f(\sum \beta_i v_i)$
$\leq \sum \beta_i f(v_i)$
$\leq \max_i f(v_i)$

Geometric View of Canonical Form

A **polytope** in n-dimensional space
- Each inequality corresponds to a half-space.
- The "feasible set" is the intersection of the half-spaces.
- This corresponds to a polytope
- The optimal solution is at a corner.

Simplex moves around on the surface of the polytope
Interior-Point methods move within the polytope

Notes about higher dimensions

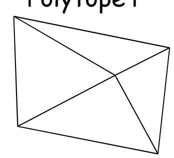
For n dimensions and no degeneracy
(i.e., A has full row rank)
Each corner (extreme point) consists of:
- n intersecting n-1 dimensional **hyperplanes** e.g. n = 3, 2d planes in 3d
- n intersecting **edges**

Each edge corresponds to moving off of one hyperplane (still constrained by n-1 of them)

Simplex will move from corner to corner along the edges

The Simple Essense of Simplex

Polytope P



Input: min $f(x) := cx$
s.t. x in $P := \{x: Ax \leq b, x \geq 0\}$

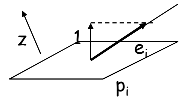
Consider Polytope P from canonical form as a graph G = (V,E) with
V = polytope vertices,
E = polytope edges.

1) Find *any* vertex v of P.
2) While there exists a neighbor u of v in G with $f(u) < f(v)$, update v to u.
3) Output v.

Choice of neighbor if several u have $f(u) < f(v)$?
Termination? Correctness? Running Time?

Optimality and Reduced Cost

The **Reduced cost** for a hyperplane at a corner is the cost of moving one unit away from the plane along its corresponding edge.

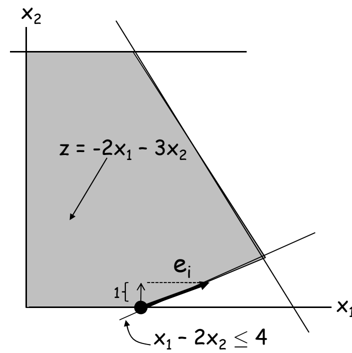


$r_i = z \cdot e_i$

For **minimization**, if all reduced cost are non-negative, then we are at an optimal solution.

Finding the most negative reduced cost is one often used heuristic for choosing an edge to leave on

Reduced cost example



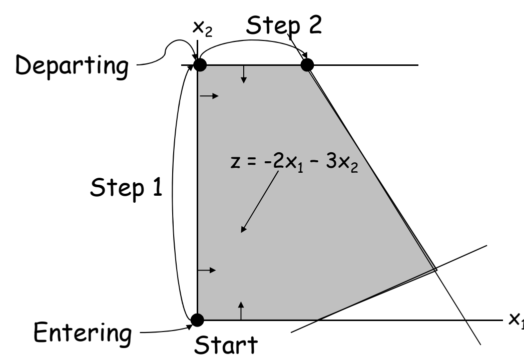
$e_i = (2, 1)$

In the example the reduced cost of leaving the plane x_1 is $(-2,-3) \cdot (2,1) = -7$ since moving one unit off of x_1 will move us (2,1) units along the edge. We take the dot product of this and the cost function.

Simplex Algorithm

1. Find a **corner of the feasible region**
2. **Repeat**
   A. For each of the n hyperplanes intersecting at the corner, calculate its **reduced cost**
   B. If they are all non-negative, then **done**
   C. Else, pick the most negative reduced cost
   This is called the **entering** plane
   D. Move along corresponding edge (i.e. leave that hyperplane) until we reach the next corner (i.e. reach another hyperplane)
   The new plane is called the **departing** plane

Example



Simplifying

Problem:

- The $Ax \leq b$ constraints not symmetric with the $x \geq 0$ constraints.
  We would like more symmetry.

Idea:

- Make all inequalities of the form $x \geq 0$.

Use "slack variables" to do this.

Convert into form:

minimize $c^T x$
subject to $Ax = b$
$x \geq 0$

Standard Form

minimize $c^T x$
subject to $Ax \leq b$
$x \geq 0$

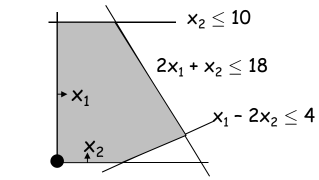
slack variables →

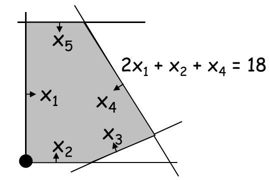
Standard Form

minimize $c^T x'$
subject to $A'x' = b$
$x' \geq 0$

$|A| = m \times n$
i.e. m equations, n variables

$|A'| = m \times (m+n)$
i.e. m equations, m+n variables





Example, again

minimize:

$z = -2x_1 - 3x_2$

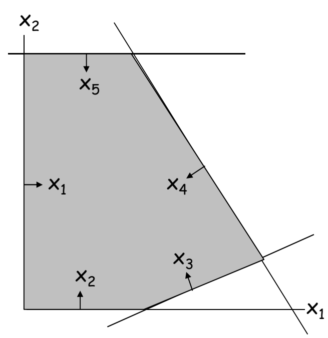
subject to:

$x_1 - 2x_2 + \underline{x_3} = 4$

$2x_1 + x_2 + \underline{x_4} = 18$

$x_2 + \underline{x_5} = 10$

$x_1, x_2, \underline{x_3}, \underline{x_4}, \underline{x_5} \geq 0$



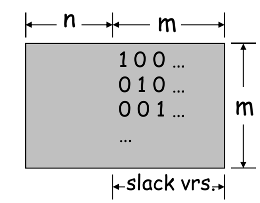
The equality constraints impose a 2d plane embedded in 5d space, looking at the plane gives the figure above

Using Matrices

If before adding the slack variables A has size m x n then after it has size m x (n + m)

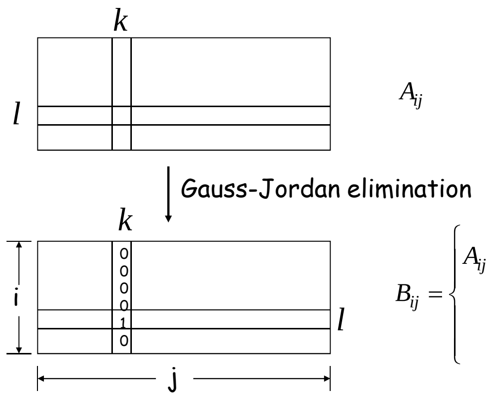
m can be larger or smaller than n

A =



Assuming rows are independent, the solution space of Ax = b is a n dimensional subspace.

Gauss-Jordan Elimination



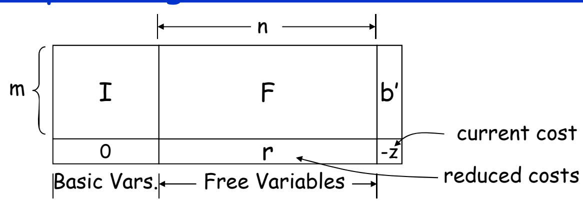
A_{ij}

$$B_{ij} = \begin{cases} A_{ij} - A_{lj}\dfrac{A_{ik}}{A_{lk}} & i \neq l \\ \dfrac{A_{lj}}{A_{lk}} & i = l \end{cases}$$

Simplex Algorithm, again

1. Find a **corner of the feasible region**
2. **Repeat**
   A. For each of the n hyperplanes intersecting at the corner, calculate its **reduced cost**
   B. If they are all non-negative, then **done**
   C. Else, pick the most negative reduced cost
   This is called the **entering** plane
   D. Move along corresponding line (i.e. leave that hyperplane) until we reach the next corner (i.e. reach another hyperplane)
   The new plane is called the **departing** plane

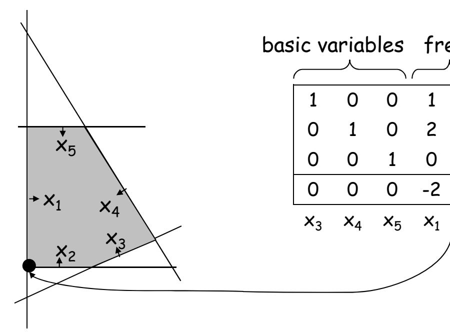
Simplex Algorithm (Tableau Method)



This form is called a **Basic Solution**
- the n "**free**" variables are set to 0
- the m "**basic**" variables are set to b'

A valid solution to Ax = b if reached using Gaussian Elimination
Represents n intersecting hyperplanes
If feasible (i.e. $b' \geq 0$), then the solution is called
a **Basic Feasible Solution** and is a corner of the feasible set

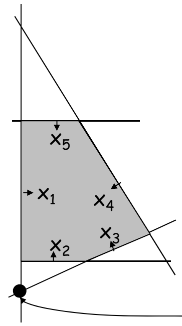
Corner



basic variables | free variables

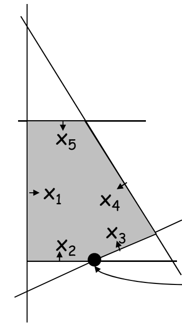
1	0	0	1	-2	4
0	1	0	2	1	18
0	0	1	0	1	10
0	0	0	-2	-3	0
x_3	x_4	x_5	x_1	x_2	

Corner



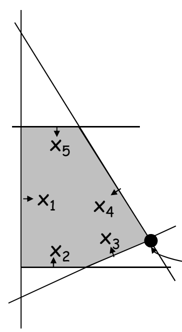
1	0	0	-.5	-1	-2
0	1	0	2.5	1	20
0	0	1	.5	1	12
0	0	0	-3.5	-3	-6
x_2	x_4	x_5	x_1	x_3	

Corner



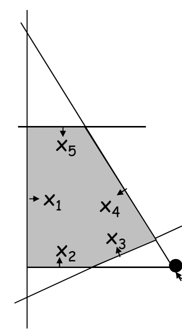
1	0	0	1	-2	4
0	1	0	-2	5	10
0	0	1	0	1	10
0	0	0	2	-7	8
x_1	x_4	x_5	x_3	x_2	

Corner



1	0	0	.2	.4	8
0	1	0	-.4	.2	2
0	0	1	.4	-.2	8
0	0	0	-.8	1.4	22
x_1	x_2	x_5	x_3	x_4	

Corner



1	0	0	.5	-2.5	-5
0	1	0	.5	.5	9
0	0	1	0	1	10
0	0	0	1	-2	18
x_3	x_1	x_5	x_4	x_2	

Note that in general there are n+m choose m corners

Simplex Method Again

Once you have found a basic feasible solution (a corner), we can move from corner to corner by swapping columns and eliminating.

ALGORITHM

1. Find a **basic feasible solution**
2. **Repeat**
   A. If r (reduced cost) ≥ 0 , DONE
   B. Else, pick column with most negative r
   C. Pick row with least positive b'/(selected column)
   D. Swap columns
   E. Use Gaussian elimination to restore form

Tableau Method

A. If r are all non-negative then **done**

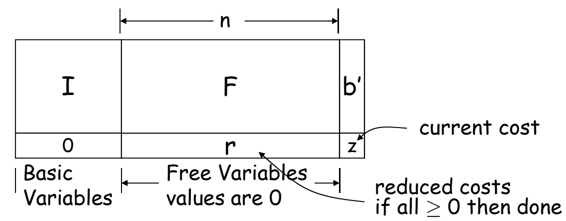


Tableau Method

B. Else, pick the most negative reduced cost
This is called the **entering** plane

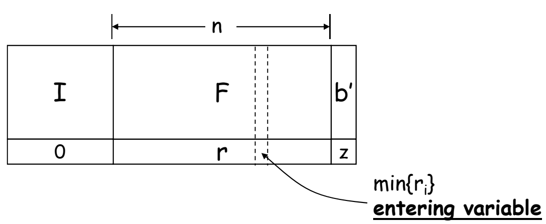


Tableau Method

C. Move along corresponding line (i.e. leave that hyperplane) until we reach the next corner (i.e. reach another hyperplane)
The new plane is called the **departing** plane

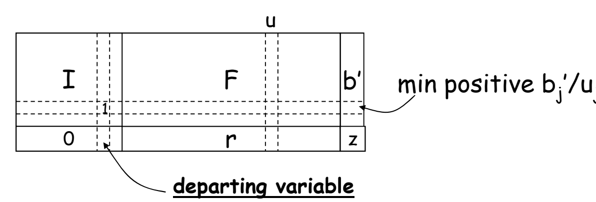
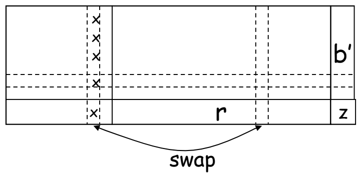


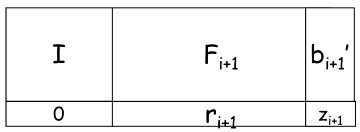
Tableau Method

D. Swap columns



No longer in proper form

E. Gauss-Jordan elimination



Back to proper form

Example

x_1	x_2	x_3	x_4	x_5	
1	-2	1	0	0	4
2	1	0	1	0	18
0	1	0	0	1	10
-2	-3	0	0	0	0

$x_1 - 2x_2 + x_3 = 4$

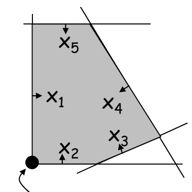
$2x_1 + x_2 + x_4 = 18$

$x_2 + x_5 = 10$

$z = -2x_1 - 3x_2$

Find corner

x_3	x_4	x_5	x_1	x_2	
1	0	0	1	-2	4
0	1	0	2	1	18
0	0	1	0	1	10
0	0	0	-2	-3	0



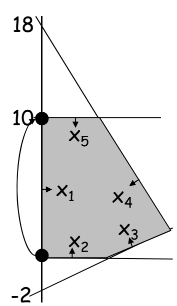
$x_1 = x_2 = 0$ (start)

Example

x_3	x_4	x_5	x_1	x_2	
1	0	0	1	-2	4
0	1	0	2	1	18
0	0	1	0	1	10
0	0	0	-2	-3	0

x_3	x_4	x_5	x_1	x_2		b_j/v_j
1	0	0	1	-2	4	-2
0	1	0	2	1	18	18
0	0	1	0	1	10	10
0	0	0	-2	-3	0	

min positive



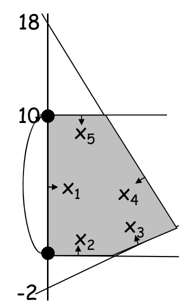
Example

swap

x_3	x_4	x_2	x_1	x_5	
1	0	-2	1	0	4
0	1	1	2	0	18
0	0	1	0	1	10
0	0	-3	-2	0	0

Gauss-Jordan Elimination

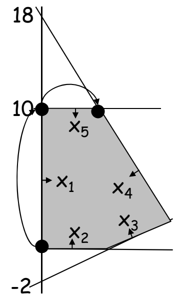
x_3	x_4	x_2	x_1	x_5	
1	0	0	1	2	24
0	1	0	2	-1	8
0	0	1	0	1	10
0	0	0	-2	3	30



Example

1	0	0	1	2	24
0	1	0	2	-1	8
0	0	1	0	1	10
0	0	0	-2	3	30
x_3	x_4	x_2	x_1	x_5	

1	0	0	1	2	24	24
0	1	0	2	-1	8	4
0	0	1	0	1	10	∞
0	0	0	-2	3	30	
x_3	x_4	x_2	x_1	x_5		



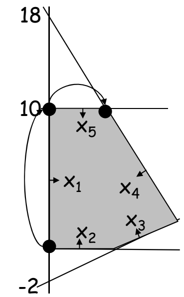
Example

swap

1	1	0	0	2	24
0	2	0	1	-1	8
0	0	1	0	1	10
0	-2	0	0	3	30
x_3	x_1	x_2	x_4	x_5	

1	0	0	-.5	2.5	20
0	1	0	.5	-.5	4
0	0	1	0	1	10
0	0	0	1	2	38
x_3	x_1	x_2	x_4	x_5	

Gauss-Jordan Elimination



Simplex Concluding remarks

For dense matrices, takes O(n(n+m)) time per iteration

Can take an **exponential** number of iterations.

In practice, sparse methods are used for the iterations.

Duality

Primal (P):

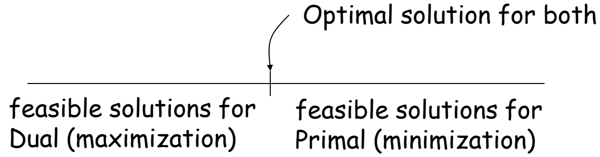
maximize $z = c^T x$
subject to $Ax \le b$
$x \ge 0$ (n equations, m variables)

Dual (D):

minimize $z = y^T b$
subject to $A^T y \ge c$
$y \ge 0$ (m equations, n variables)

Duality Theorem: if x is feasible for **P** and y is feasible for **D**, then $cx \le yb$ and at optimality $cx = yb$.

Duality (cont.)



Quite similar to duality of Maximum Flow and Minimum Cut.

Useful in many situations.

Duality Example

Primal:

maximize:

$z = 2x_1 + 3x_2$

subject to:

$x_1 - 2x_2 \leq 4$

$2x_1 + x_2 \leq 18$

$x_2 \leq 10$

$x_1, x_2 \geq 0$

Dual:

minimize:

$z = 4y_1 + 18y_2 + 10y_3$

subject to:

$y_1 + 2y_2 \geq 2$

$-2y_1 + y_2 + y_3 \geq 3$

$y_1, y_2, y_3 \geq 0$

Solution to both is 38 (x_1=4, x_2=10), (y_1=0, y_2=1, y_3=2).