DM587

Linear Algebra with Applications

Linear Programming

Marco Chiarandini

Department of Mathematics & Computer Science
University of Southern Denmark

Outline

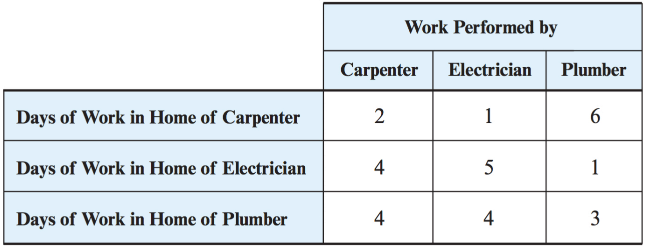
Outline

Closed Model

- Economic system consisting of a finite number of industries, $1, 2, \ldots, k$.
- Over some fixed period of time, each industry produces output of some good or service that is completely utilized in a predetermined manner by the k industries.
- Find suitable prices to be charged for these k outputs so that for each industry, total expenditures equal total income.
- Such a price structure represents an equilibrium position for the economy.

Closed Model

Three homeowners – a carpenter, an electrician, and a plumber – agree to make repairings in their three homes. They agree to work a total of 10 days each according to the following schedule:



	Work Performed by		
	Carpenter	**Electrician**	**Plumber**
Days of Work in Home of Carpenter	2	1	6
Days of Work in Home of Electrician	4	5	1
Days of Work in Home of Plumber	4	4	3

For tax purposes, they must report and pay each other a reasonable daily wage, even for the work each does on his or her own home.

Their normal daily wages are about \$100, but they agree to adjust their respective daily wages so that each homeowner will come out even—that is, so that the total amount paid out by each is the same as the total amount each receives. What should be the prices of their work?

Closed Model

- $$\begin{array}{rcrcrcr} 2p_1 & + & p_2 & + & 6p_3 & = & 10p_1 \\ 4p_1 & + & 5p_2 & + & p_3 & = & 10p_2 \\ 4p_1 & + & 4p_2 & + & 3p_3 & = & 10p_3 \end{array}$$

- $(I - A)\mathrm{p} = \mathrm{p}$. If $\det(I - A) \neq 0$ then non trivial solution. Moreover, it can be shown that for exchange matrices A that are stochastic matrices the solution p is such that its elements are non-negative and if A^m are positive for all m positive integer then all p entries are positive.

Open Model

- Consider a market with n industries producing n different commodities.
- The market is interdependent, meaning that each industry requires input from the other industries and possibly even its own commodity.
- In addition, there is an outside demand for each commodity that has to be satisfied.
- We wish to determine the amount of output of each industry which will satisfy all demands exactly; that is, both the demands of the other industries and the outside demand.

Open Model

- Let $n = 3$ and let a_{ij} indicate the amount of commodity i, $i = 1, 2, 3$ necessary to produce one unit of commodity j, $j = 1, 2, 3$.

- a_{ij} are given in monetary terms:
  a_{ij} cost of the commodity i necessary to produce one unit profit of commodity j.
  $\rightsquigarrow$ Hence, we will assume that $a_{ij} \geq 0$
  Example: to produce an amount of commodity j worth 100 dkk, one needs an amount of commodity i worth 30 dkk.

  For the sake of simplicity we scale all these values such that the profit of each commodity is 1 unity of currency.

  $\rightsquigarrow$ Hence, for each commodity j its production is not profitable unless

$$\sum_{i=1}^{3} a_{ij} < 1.$$

- d_i be the demand of commodity i expressed in units of currency.

For each commodity, the outside demand is covered by the production of the commodity after the subtraction of the amount of commodity that has to go in the other industries and the amount that has to go in the same industry.
Hence:

$$x_i - \sum_{j=1}^{n} a_{ij}x_j = d_i$$

For $n = 3$ we have:

$$\begin{aligned}
x_1 - a_{11}x_1 - a_{12}x_2 - a_{13}x_3 &= d_1\\
x_2 - a_{21}x_1 - a_{22}x_2 - a_{23}x_3 &= d_2\\
x_3 - a_{31}x_1 - a_{32}x_2 - a_{33}x_3 &= d_3
\end{aligned}$$

In matrix terms:

$$I\mathrm{x} - A\mathrm{x} = \mathrm{d} \qquad \text{or} \qquad (I - A)\mathrm{x} = \mathrm{d}$$

which is a system of linear equations. To make sense the solution x must be non-negative. It can be shown that under the conditions expressd above the solution to the system is unique and non-negative.

Open Model

- Unique non-negative solution for x if and only if $(I - A)^{-1}$ exists and $(I - A)^{-1} \geq 0$.
- The matrix A such that $(I - A)^{-1}$ exists and $(I - A)^{-1} \geq 0$ is called productive.
- The matrix A is productive $\iff$ there exists $x \geq 0$ such that $x > Ax$
  $\iff \sum_{j=1}^{n} a_{ij} < 1$ (row sums)
  that is, there is some production plan such that each industry produces (monetarily) more than it consumes.
- A matrix is productive $\iff \sum_{i=1}^{m} a_{ij} < 1$ (column sums)
  that is, the jth industry is profitable if the total value of the outputs of all m industries needed to produce one unit of value of output of the industry j is less than one.

Decision Support Tools

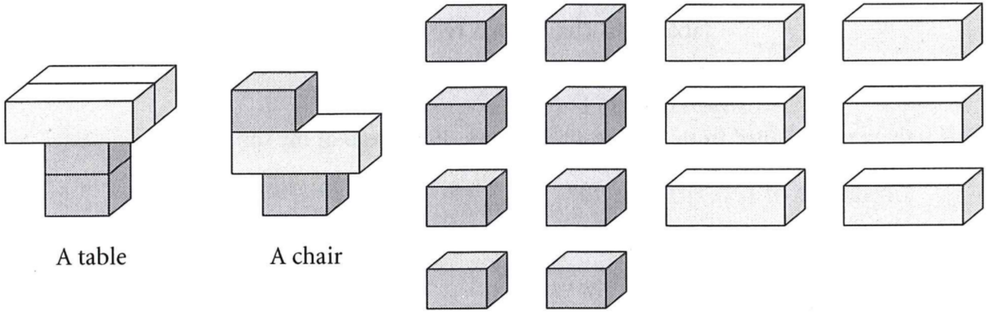
- So far we considered a full economic system (country, region) and the decision making from the point of view of a Government planning
  IO Models and Linear Systems of Equations
- Now, let's consider the Planning of Activities by a single Firm. Eg: Supply chain management, logistics, production scheduling
  Linear Programming

A firm can produce their items in many different ways. The planning problem is characterized by a large number of feasible ways of providing the same output.

Outline

Production Planning

Suppose a company produces only tables and chairs.
A table is made of 2 large Lego pieces and 2 small pieces, while a chair is made of 1 large and 2 small pieces.
The resources available are 8 small and 6 large pieces.



The profit for a table is 1600 dkk and for a chair 1000 dkk. What product mix maximizes the company's profile using the available resources?

Mathematical Model

	Tables	Chairs	Capacity
Small Pieces	2	2	8
Large Pieces	2	1	6
Profit	16	10	

Decision Variables

$x_1 \geq 0$ units of small pieces
$x_2 \geq 0$ units of large pieces

Object Function

$\max\ 16x_1 + 10x_2$ maximize profit

Constraints

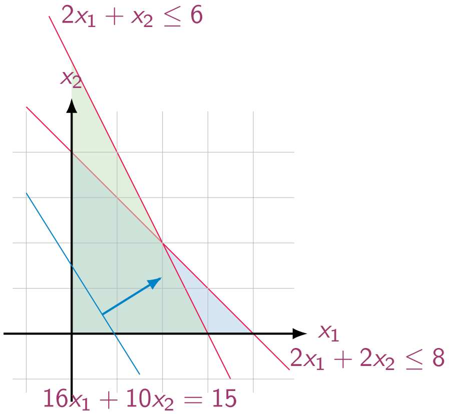
$$\begin{array}{rcrcll} 2x_1 & + & 2x_2 & \leq & 8 & \text{small pieces capacity} \\ 2x_1 & + & x_2 & \leq & 6 & \text{large pieces capacity} \end{array}$$

Mathematical Model

Materials A and B
Products 1 and 2

$$\begin{array}{rrcrcl} \max & 16x_1 & + & 10x_2 & & \\ & 2x_1 & + & 2x_2 & \leq & 8 \\ & 2x_1 & + & x_2 & \leq & 6 \\ & & & x_1 & \geq & 0 \\ & & & x_2 & \geq & 0 \end{array}$$

Graphical Representation:



Resource Allocation - General Model

Managing a production facility

$1, 2, \ldots, n$	products
$1, 2, \ldots, m$	materials
b_i	units of raw material at disposal
a_{ij}	units of raw material i to produce one unit of product j
σ_j	market price of unit of jth product
ρ_i	prevailing market value for material i
$c_j = \sigma_j - \sum_{i=1}^{n} \rho_i a_{ij}$	profit per unit of product j
x_j	amount of product j to produce

$$\begin{array}{rcccccccccccc}
\max & c_1x_1 & + & c_2x_2 & + & c_3x_3 & + & \ldots & + & c_nx_n & = & z \\
\text{subject to} & a_{11}x_1 & + & a_{12}x_2 & + & a_{13}x_3 & + & \ldots & + & a_{1n}x_n & \leq & b_1 \\
 & a_{21}x_1 & + & a_{22}x_2 & + & a_{23}x_3 & + & \ldots & + & a_{2n}x_n & \leq & b_2 \\
 & \ldots & & & & & & & & & & \\
 & a_{m1}x_1 & + & a_{m2}x_2 & + & a_{m3}x_3 & + & \ldots & + & a_{mn}x_n & \leq & b_m \\
 & & & & & & & & & x_1, x_2, \ldots, x_n & \geq & 0
\end{array}$$

Notation

$$\begin{array}{rcccccccccccl}
\text{max} & c_1x_1 & + & c_2x_2 & + & c_3x_3 & + & \dots & + & c_nx_n & = & z \\
\text{s.t.} & a_{11}x_1 & + & a_{12}x_2 & + & a_{13}x_3 & + & \dots & + & a_{1n}x_n & \leq & b_1 \\
 & a_{21}x_1 & + & a_{22}x_2 & + & a_{23}x_3 & + & \dots & + & a_{2n}x_n & \leq & b_2 \\
 & \dots & & & & & & & & & & \\
 & a_{m1}x_1 & + & a_{m2}x_2 & + & a_{m3}x_3 & + & \dots & + & a_{mn}x_n & \leq & b_m \\
 & & & & & & & & & x_1, x_2, \dots, x_n & \geq & 0
\end{array}$$

$$\begin{array}{rl}
\text{max} & \sum_{j=1}^{n} c_j x_j \\
 & \sum_{j=1}^{n} a_{ij} x_j \leq b_i, \quad i = 1, \dots, m \\
 & x_j \geq 0, \quad j = 1, \dots, n
\end{array}$$

In Matrix Form

$$\begin{array}{rrcrcrcccrcl}
\max & c_1x_1 & + & c_2x_2 & + & c_3x_3 & + & \dots & + & c_nx_n & = & z \\
\text{s.t.} & a_{11}x_1 & + & a_{12}x_2 & + & a_{13}x_3 & + & \dots & + & a_{1n}x_n & \leq & b_1 \\
 & a_{21}x_1 & + & a_{22}x_2 & + & a_{23}x_3 & + & \dots & + & a_{2n}x_n & \leq & b_2 \\
 & \dots & & & & & & & & & & \\
 & a_{m1}x_1 & + & a_{m2}x_2 & + & a_{m3}x_3 & + & \dots & + & a_{mn}x_n & \leq & b_m \\
 & & & & & & & & & x_1, x_2, \dots, x_n & \geq & 0
\end{array}$$

$$c^T = \begin{bmatrix} c_1 & c_2 & \dots & c_n \end{bmatrix}$$

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & & & \\ a_{31} & a_{32} & \dots & a_{mn} \end{bmatrix}, x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}$$

$$\begin{array}{rrcl}
\max & z & = & c^Tx \\
 & Ax & \leq & b \\
 & x & \geq & 0
\end{array}$$

Vector and Matrices in Excel

$$\sum_{j=1}^{n} c_j = c_1 + c_2 + \ldots + c_n$$

$$\mathtt{SUM(B5:B14)}$$

Scalar product

$$\begin{aligned} \mathrm{u} \cdot \mathrm{v} &= u_1 v_1 + u_2 v_2 + \ldots + u_n v_n \\ &= \sum_{j=1}^{n} u_j v_j \end{aligned}$$

$$\mathtt{SUMPRODUCT(B5:B14, C5:C:14)}$$

Our Numerical Example

$$\begin{aligned} \max \ & \sum_{j=1}^{n} c_j x_j \\ & \sum_{j=1}^{n} a_{ij} x_j \leq b_i, \quad i = 1, \ldots, m \\ & x_j \geq 0, \quad j = 1, \ldots, n \end{aligned}$$

$$\begin{array}{lrcrcl} \max & 16x_1 & + & 10x_2 & & \\ & 2x_1 & + & 2x_2 & \leq & 8 \\ & 2x_1 & + & x_2 & \leq & 6 \\ & & & x_1, x_2 & \geq & 0 \end{array}$$

$$\begin{aligned} \max \ & c^T x \\ & Ax \leq b \\ & x \geq 0 \end{aligned}$$

$$x \in \mathbb{R}^n, c \in \mathbb{R}^n, A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m$$

$$\max \quad \begin{bmatrix} 16 & 10 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

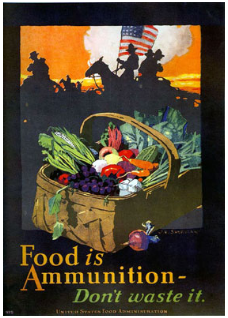
$$\begin{bmatrix} 2 & 2 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} 8 \\ 6 \end{bmatrix}$$

$$x_1, x_2 \geq 0$$

Outline

The Diet Problem (Blending Problems)

- Select a set of foods that will satisfy a set of daily nutritional requirement at minimum cost.
- Motivated in the 1930s and 1940s by US army.
- Formulated as a linear programming problem by George Stigler
- First linear programming problem
- (programming intended as planning not computer code)



min cost/weight
subject to nutrition requirements:
eat enough but not too much of Vitamin A
eat enough but not too much of Sodium
eat enough but not too much of Calories
...

The Diet Problem

Suppose there are:

- 3 foods available, corn, milk, and bread,
- there are restrictions on the number of calories (between 2000 and 2250) and the amount of Vitamin A (between 5000 and 50,000)

Food	Corn	2% Milk	Wheat bread
Vitamin A	107	500	0
Calories	72	121	65
Cost per serving	\$0.18	\$0.23	\$0.05

The Mathematical Model

Parameters (given data)

F = set of foods
N = set of nutrients

a_{ij} = amount of nutrient j in food i, $\forall i \in F$, $\forall j \in N$
c_i = cost per serving of food $i, \forall i \in F$
F_{mini} = minimum number of required servings of food $i, \forall i \in F$
F_{maxi} = maximum allowable number of servings of food $i, \forall i \in F$
N_{minj} = minimum required level of nutrient $j, \forall j \in N$
N_{maxj} = maximum allowable level of nutrient $j, \forall j \in N$

Decision Variables

x_i = number of servings of food i to purchase/consume, $\forall i \in F$

The Mathematical Model

Objective Function: Minimize the total cost of the food

$$\text{Minimize} \sum_{i \in F} c_i x_i$$

Constraint Set 1: For each nutrient $j \in N$, at least meet the minimum required level

$$\sum_{i \in F} a_{ij} x_i \geq N_{minj}, \qquad \forall j \in N$$

Constraint Set 2: For each nutrient $j \in N$, do not exceed the maximum allowable level.

$$\sum_{i \in F} a_{ij} x_i \leq N_{maxj}, \qquad \forall j \in N$$

Constraint Set 3: For each food $i \in F$, select at least the minimum required number of servings

$$x_i \geq F_{mini}, \qquad \forall i \in F$$

Constraint Set 4: For each food $i \in F$, do not exceed the maximum allowable number of servings.

$$x_i \leq F_{maxi}, \qquad \forall i \in F$$

The Mathematical Model

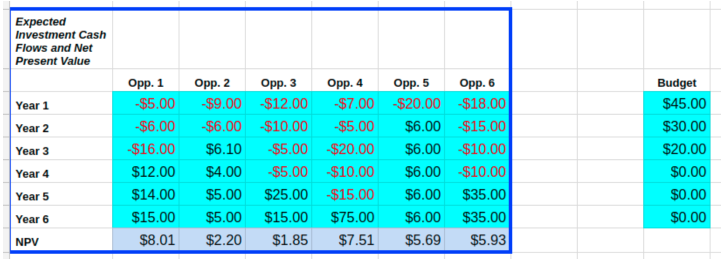
system of equalities and inequalities

$$
\begin{aligned}
\min \quad & \sum_{i \in F} c_i x_i \\
\sum_{i \in F} a_{ij} x_i \geq N_{minj}, \quad & \forall j \in N \\
\sum_{i \in F} a_{ij} x_i \leq N_{maxj}, \quad & \forall j \in N \\
x_i \geq F_{mini}, \quad & \forall i \in F \\
x_i \leq F_{maxi}, \quad & \forall i \in F
\end{aligned}
$$

Outline

Budget Allocation

- A company has six different opportunities to invest money.
- Each opportunity requires a certain investment over a period of 6 years or less.



Expected Investment Cash Flows and Net Present Value							
	Opp. 1	**Opp. 2**	**Opp. 3**	**Opp. 4**	**Opp. 5**	**Opp. 6**	**Budget**
Year 1	-\$5.00	-\$9.00	-\$12.00	-\$7.00	-\$20.00	-\$18.00	\$45.00
Year 2	-\$6.00	-\$6.00	-\$10.00	-\$5.00	\$6.00	-\$15.00	\$30.00
Year 3	-\$16.00	\$6.10	-\$5.00	-\$20.00	\$6.00	-\$10.00	\$20.00
Year 4	\$12.00	\$4.00	-\$5.00	-\$10.00	\$6.00	-\$10.00	\$0.00
Year 5	\$14.00	\$5.00	\$25.00	-\$15.00	\$6.00	\$35.00	\$0.00
Year 6	\$15.00	\$5.00	\$15.00	\$75.00	\$6.00	\$35.00	\$0.00
NPV	\$8.01	\$2.20	\$1.85	\$7.51	\$5.69	\$5.93	

- The company has an investment budget that needs to be met for each year.
- It also has the wish of investing in those opportunities that maximize the combined Net Present Value (NPV) after the 6th year.

Digression: What is the Net Present Value?

- P: value of the original payment presently due
- the debtor wants to delay the payment for t years,
- let r be the market rate of return that the creditor would obtain from a similar investment asset
- the future value of P is $F = P(1+r)^t$

Viceversa, consider the task of finding:

- the present value P of \$100 that will be received in five years, or equivalently,
- which amount of money today will grow to \$100 in five years when subject to a constant discount rate.

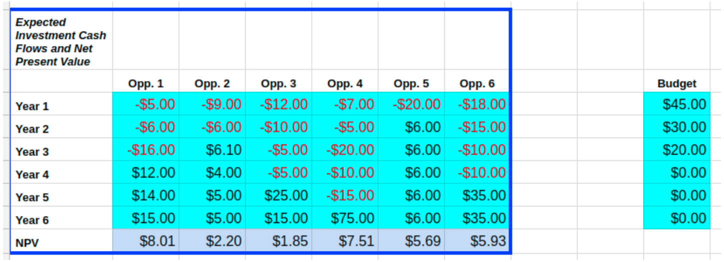
Assuming a 5% per year interest rate, it follows that

$$P = \frac{F}{(1+r)^t} = \frac{\$100}{(1+0.05)^5} = \$78.35.$$

Budget Allocation

Net Present Value calculation:
for each opportunity we calculate the NPV at time zero (the time of decision) as:

$$P_0 = \sum_{t=1}^{5} \frac{F_t}{(1+0.05)^5}$$



Expected Investment Cash Flows and Net Present Value							
	Opp. 1	**Opp. 2**	**Opp. 3**	**Opp. 4**	**Opp. 5**	**Opp. 6**	**Budget**
Year 1	-$5.00	-$9.00	-$12.00	-$7.00	-$20.00	-$18.00	$45.00
Year 2	-$6.00	-$6.00	-$10.00	-$5.00	$6.00	-$15.00	$30.00
Year 3	-$16.00	$6.10	-$5.00	-$20.00	$6.00	-$10.00	$20.00
Year 4	$12.00	$4.00	-$5.00	-$10.00	$6.00	-$10.00	$0.00
Year 5	$14.00	$5.00	$25.00	-$15.00	$6.00	$35.00	$0.00
Year 6	$15.00	$5.00	$15.00	$75.00	$6.00	$35.00	$0.00
NPV	$8.01	$2.20	$1.85	$7.51	$5.69	$5.93	

Budget Allocation - Mathematical Model

- Let B_t be the budget available for investments during the years $t = 1..5$.
- Let a_{tj} be the cash flow for opportunity j and c_j its NPV
- Task: choose a set of opportunities such that the budget is never exceeded and the expected return is maximized. Consider both the case of indivisible and divisible opportunities.

Variables $x_j = 1$ if opportunity j is selected and $x_j = 0$ otherwise, $j = 1..6$

Objective

$$\max \sum_{j=1}^{6} c_j x_j$$

Constraints

$$\sum_{j=1}^{6} a_{tj} x_j + B_t \geq 0 \qquad \forall t = 1..5$$

DM587

Scientific Programming

Affine Scaling Method

Marco Chiarandini

Department of Mathematics & Computer Science
University of Southern Denmark

Outline

Outline

Interior Point Methods

Interior point methods in linear programming are classified as:

- affine scaling methods
- potential reduction methods, and
- central path methods (or central trajectory methods)

and for almost every approach one can consider

- a primal version,
- a dual version,
- a primal–dual version, or
- a self-dual version.

- affine scaling by Dikin
- logarithmic barrier algorithm by Fiacco and McCormick
- ellipsoid algorithm by Khachian
- projective method by Karmarkar $\equiv$ logarithmic barrier
- primal-dual logarithmic barrier
- primal–dual barrier algorithm, combined with Mehrotra's predictor–corrector method

Applied with success also to semidefinite programming and other important classes of optimization problems, such as convex quadratic programming.

Interior point algorithm with affine scaling

- Historically, one of the first interior-point methods to be invented
- No longer considered the method of choice for practical implementations

- Concept 1: Shoot through the interior of the feasible region toward an optimal solution.
- Concept 2: Move in a direction that improves the objective function value at the fastest possible rate.
- Concept 3: Transform the feasible region to place the current trial solution near its center, thereby enabling a large improvement when concept 2 is implemented.

Affine Scaling Method — Phase II

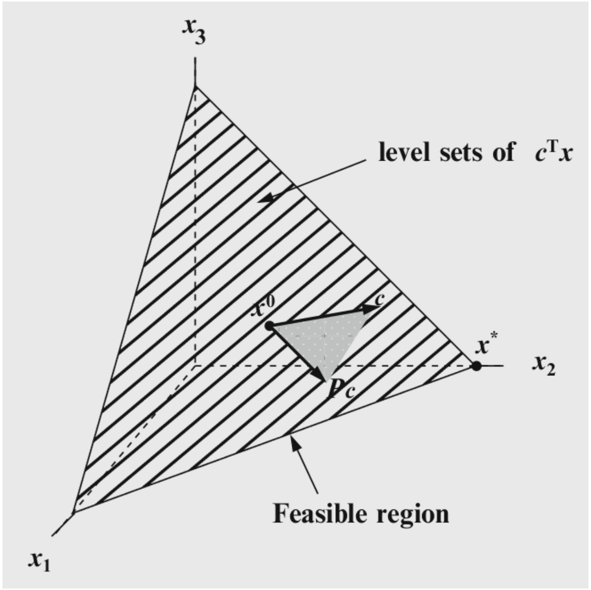
$$\begin{aligned} \text{maximize } & \boldsymbol{c}^T \boldsymbol{x} \\ \text{subject to } & A\boldsymbol{x} = \boldsymbol{b} \\ & \boldsymbol{x} \geq 0 \end{aligned}$$

Two-phase method:

- Phase I uses only the feasibility direction
- Phase II uses only the optimality direction

Phase II
We assume that we have a feasible initial starting point, x_0 that lies in the strict interior of the feasible set. That is:

$$Ax_0 = b \qquad \text{and} \qquad x_0 > 0$$


x_3
level sets of $c^T x$
x^0
c
x^*
x_2
P_c
Feasible region
x_1

Hence, the steepest ascent direction will almost surely cause a move to infeasible points. This is also clear algebraically. Indeed,

$$A(x^0 + \Delta x) = Ax^0 + A\Delta x = b + Ac \neq b$$

(unless Ac = 0 which is not likely).

To see how to find a better direction, let us first review in what sense the gradient is the steepest ascent direction. The *steepest ascent direction* is defined to be the direction that gives the greatest increase in the objective function subject to the constraint that the displacement vector has unit length. That is, the steepest ascent direction is the solution to the following optimization problem:

$$\begin{array}{ll} \text{maximize} & c^T(x^0 + \Delta x) \\ \text{subject to} & \|\Delta x\|^2 = 1. \end{array} \tag{21.2}$$

We can solve this problem using Lagrange multipliers. Indeed, if we let λ denote the Lagrange multiplier for the constraint, the problem becomes

$$\max_{\Delta x, \lambda} c^T(x^0 + \Delta x) - \lambda(\Delta x^T \Delta x - 1).$$

Differentiating with respect to Δx and setting the derivative to zero, we get

$$c - 2\lambda \Delta x = 0,$$

which implies that

$$\Delta x = \frac{1}{2\lambda} c \propto c.$$

Then differentiating the Lagrangian with respect to λ and setting that derivative to zero, we see that

$$\|\Delta x\|^2 - 1 = 0,$$

which implies that

$$\|\Delta x\| = \pm 1.$$

Hence, the steepest ascent direction points in the direction of either c or its negative. Since the negative is easily seen not to be an ascent direction at all, it follows that the steepest ascent direction points in the direction of c.

The problem with the steepest ascent direction is that it fails to preserve feasibility. That is, it fails to preserve the equality constraints $Ax = b$. To remedy this problem, let's add these constraints to (21.2) so that we get the following optimization problem:

$$\begin{array}{ll} \text{maximize} & c^T(x^0 + \Delta x) \\ \text{subject to} & \|\Delta x\|^2 = 1 \\ & A(x^0 + \Delta x) = b. \end{array}$$

Again, the method of Lagrange multipliers is the appropriate tool. As before, let λ denote the Lagrange multiplier for the norm constraint, and now introduce a vector y containing the Lagrange multipliers for the equality constraints. The resulting unconstrained optimization problem is

$$\max_{\Delta x, \lambda, y} \; c^T(x^0 + \Delta x) - \lambda(\Delta x^T \Delta x - 1) - y^T(A(x^0 + \Delta x) - b).$$

$$\max_{\Delta x, \lambda, y} \; c^T(x^0 + \Delta x) - \lambda(\Delta x^T \Delta x - 1) - y^T(A(x^0 + \Delta x) - b).$$

Differentiating this Lagrangian with respect to Δx, λ, and y and setting these derivatives to zero, we get

$$\begin{aligned} c - 2\lambda\Delta x - A^T y &= 0 \\ \|\Delta x\|^2 - 1 &= 0 \\ A(x^0 + \Delta x) - b &= 0. \end{aligned}$$

The second equation tells us that the length of Δx is one. Since we are interested in the direction of Δx and are not concerned about its length, we ignore this second equation. The first equation tells us that Δx is proportional to $c - A^T y$, and again, since we aren't concerned about lengths, we put $\lambda = 1/2$ so that the first equation reduces to

$$\Delta x = c - A^T y. \tag{21.3}$$

Since $Ax^0 = b$, the third equation says that

$$A\Delta x = 0.$$

Substituting (21.3) into this equation, we get

$$Ac - AA^T y = 0,$$

which, assuming that AA^T has full rank (as it should), can be solved for y to get

$$y = (AA^T)^{-1}Ac.$$

Now, substituting this expression into (21.3), we see that

$$\Delta x = c - A^T(AA^T)^{-1}Ac.$$

It is convenient to let P be the matrix defined by

$$P = I - A^T(AA^T)^{-1}A.$$

With this definition, Δx can be expressed succinctly as

$$\Delta x = Pc.$$

We claim that P is the matrix that maps any vector, such as c, to its orthogonal projection onto the null space of A. To justify this claim, we first need to define some of the terms we've used. The *null space* of A is defined as $\{d \in \mathbb{R}^n : Ad = 0\}$. We shall denote the null space of A by $N(A)$. A vector $\tilde{c}$ is the *orthogonal projection* of c onto $N(A)$ if it lies in the null space,

$$\tilde{c} \in N(A),$$

and if the difference between it and c is orthogonal to every other vector in $N(A)$. That is,

$$d^T(c - \tilde{c}) = 0, \qquad \text{for all } d \in N(A).$$

Hence, to show that Pc is the orthogonal projection of c onto the null space of A, we simply check these two conditions. Checking the first, we see that

$$APc = Ac - AA^T(AA^T)^{-1}Ac,$$

which clearly vanishes. To check the second condition, let d be an arbitrary vector in the null space, and compute

$$d^T(c - Pc) = d^T A^T (AA^T)^{-1} Ac,$$

which also vanishes, since $d^T A^T = (Ad)^T = 0$. The orthogonal projection Pc is shown in Figure 21.1.

Scaling

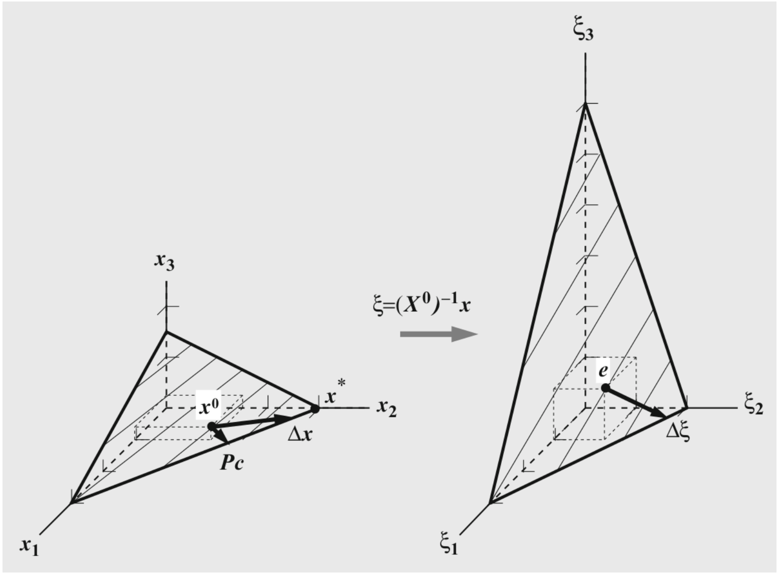
- If it is close to a "wall", the overall increase in one step will be small
- scale the variables in the problem so that the current feasible solution is far from the walls, compute the step direction as the projected gradient in the scaled problem, and then translate this direction back into the original system.
- scale each variable in such a manner that its initial value gets mapped to 1. That is, for each $j = 1, 2, \ldots, n$, we introduce new variables given by

  $$\xi_j = \frac{x_j}{x_j^0} \qquad \Longrightarrow \qquad x_j = x_j^0 \xi_j$$

  In matrix notation:

  $$\boldsymbol{x} = X^0 \boldsymbol{\xi}$$

  (X^0 diagonal matrix of diagonal $\boldsymbol{x}^0$)
  under this change of variables, the initial solution $\boldsymbol{x}_0$ gets mapped to the vector $\boldsymbol{e}$ of all ones, which is at least one unit away from each wall.


x_3
x^0
x^*
x_2
Δx
Pc
x_1
$\xi = (X^0)^{-1} x$
ξ_3
e
ξ_2
$\Delta\xi$
ξ_1

After substitution:

$$\begin{aligned} \text{maximize} \quad & \boldsymbol{c}^T X^0 \xi \\ \text{subject to} \quad & A X^0 \xi = \boldsymbol{b} \\ & \xi \geq 0 \end{aligned}$$

It is a linear programmming problem in standard form with constraint matrix AX^0 and vector of objective function coeffcients $(\boldsymbol{c}^T X^0)^T = X^0 \boldsymbol{c}$ (since X^0 is diagonal matrix).

We can determine the steepest ascent direction applying what seen in the previous slides:

$$\Delta\xi = \left(I - X^0 A^T (A {X^0}^2 A^T)^{-1} A X^0 \right) X^0 \boldsymbol{c}.$$

and

$$\xi^1 = \xi^0 + \Delta\xi = \boldsymbol{e} + \Delta\xi$$

Transforming the new point ξ^1 back into the original unscaled variables, we get a new point $\boldsymbol{x}^1$:

$$\boldsymbol{x}^1 = X^0\xi^1 = X^0(\boldsymbol{e} + \Delta\xi) = \boldsymbol{x}^0 + X^0\Delta\xi$$

The difference between $\boldsymbol{x}^1$ and $\boldsymbol{x}^0$ is the step direction in the original variables. Denoting this difference by $\Delta\boldsymbol{x}$, we see that:

$$\begin{aligned}\Delta\boldsymbol{x} &= \boldsymbol{x}_1 - \boldsymbol{x}_0 = \boldsymbol{x}^0 + X^0\Delta\xi - \boldsymbol{x}^0 = \\ &= X^0\left(I - X^0A^T(A{X^0}^2A^T)^{-1}AX^0\right)X^0\boldsymbol{c} = \\ &= (E - EA^T(AEA^T)^{-1}AE)\boldsymbol{c}\end{aligned}$$

where $E = {X^0}^2$.
This expression for $\Delta\boldsymbol{x}$ is called affine-scaling step direction.

Recall that we worked with $\| \Delta\xi \| = 1$. We can then choose step lengths in such a manner as to ensure “strict” feasibility of each iteration.

Step Length

If the step was chosen so that the new point were to lie exactly on the boundary of the feasible region, then the multiplier for $\Delta\xi$ in the transformed system would be:

$$\boldsymbol{\xi}^1 = \boldsymbol{\xi}^0 + \theta\Delta\boldsymbol{\xi} = \boldsymbol{e} + \theta\Delta\boldsymbol{\xi}$$

$$\theta = \frac{1}{\max_j\{-\Delta\xi_j\}}$$

The definition of θ has been chosen to make the smallest component of $\boldsymbol{\xi}^1$ equal to zero.

In the originl system θ, would be

$$\theta = \frac{1}{\max_j\{-\Delta x_j/x_j^0\}}$$

We then shorten the step by introducing a parameter $0 < \alpha < 1$ and setting the iterations of the affine-scaling algorithm are defined by

$$\boldsymbol{x} \leftarrow \boldsymbol{x} + \alpha\theta\Delta\boldsymbol{x}$$

Convergence

Theorem (Convergence Results)

1. *If the problem and its dual are nondegenerate, then for every* $\alpha < 1$*, the sequence generated by the algorithm converges to the optimal solution.*
2. *For* $\alpha \leq 2/3$*, the sequence generated by the algorithm converges to an optimal solution (regardless of degeneracy).*
3. *There exists an example and an associated* $\alpha < 1$ *for which the algorithm converges to a nonoptimal solution.*

As for the speed of convergence, there is no example but it is believed that the Klee–Minty problem might arise here as well.

Affine-Scaling Algorithm

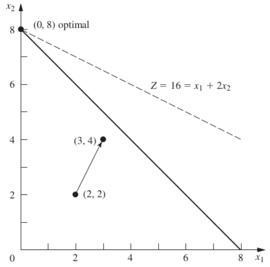
At iteration k:

1. Centering Let $D = X^k = \text{Diag}(\boldsymbol{x}^k)$. Rescale the problem to center the current interior feasible solution by letting $\tilde{A} = AD$, $\tilde{\boldsymbol{c}}^T = \boldsymbol{c}^T D$. Hence, $\boldsymbol{\xi}^k = D^{-1}\boldsymbol{x}^k = \boldsymbol{e}$, the vector consisting of all 1's. Note that $\tilde{A}\boldsymbol{\xi}^k = \boldsymbol{b}$.
2. Search Direction Computation For the rescaled problem, project the steepest ascent direction $\tilde{\boldsymbol{c}}^T$ onto the null space of the constraint matrix $\tilde{A}$, resulting in the search direction $\Delta\boldsymbol{\xi} = (I - \tilde{A}^T(\tilde{A}\tilde{A}^T)^{-1}\tilde{A})\tilde{c}$.
3. Step Length Add a positive multiple θ of the search direction $\Delta\boldsymbol{\xi}$ to the scaled interior feasible point, by computing $\boldsymbol{\xi}^{k+1} = \boldsymbol{e} + \theta\Delta\boldsymbol{\xi}$. If $\Delta\boldsymbol{\xi} \geq 0$, then $\boldsymbol{\xi}^{k+1}$, and hence $\boldsymbol{x}^{k+1}$, can increase without bound; stop the algorithm with an unbounded solution. Otherwise, because $\tilde{A}\Delta\boldsymbol{\xi} = 0$ and $\tilde{A}\boldsymbol{\xi}^{k+1} = \boldsymbol{b}$, then θ must be chosen to ensure that $\boldsymbol{\xi}^{k+1} > 0$, avoiding the border. For any constant α such that $0 < \alpha < 1$, the update $\boldsymbol{\xi}^{k+1} = \boldsymbol{e} + \left(\frac{\alpha}{\max_j\{-\Delta\xi_j\}}\right)\Delta\boldsymbol{\xi}$ suffices.
4. Optimality Test Unscale the problem, setting $x^{k+1} = D\boldsymbol{\xi}^{k+1}$. Test $\boldsymbol{x}^{k+1}$ for optimality by checking whether $\| \boldsymbol{x}^{k+1} - \boldsymbol{x}^k \|$ is small. If $\boldsymbol{x}^{k+1}$ is optimal, stop the algorithm. Otherwise, return to Step 1 with feasible interior point solution $\boldsymbol{x}^{k+1}$.

Outline

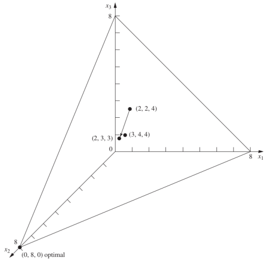
Example

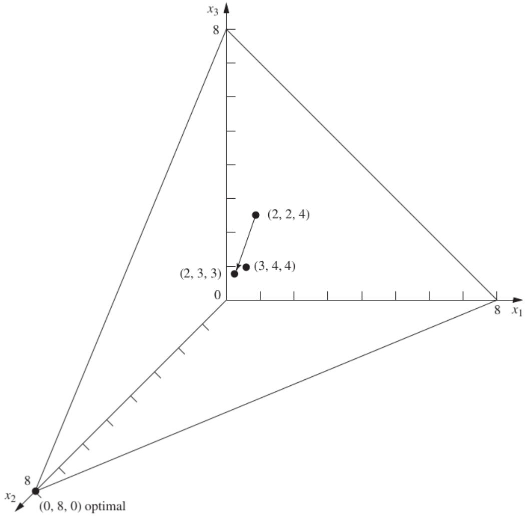
$$\max z = x_1 + 2x_2$$
$$x_1 + x_2 \leq 8$$
$$x_1 \geq 0, x_2 \geq 0$$



In equational standard form:

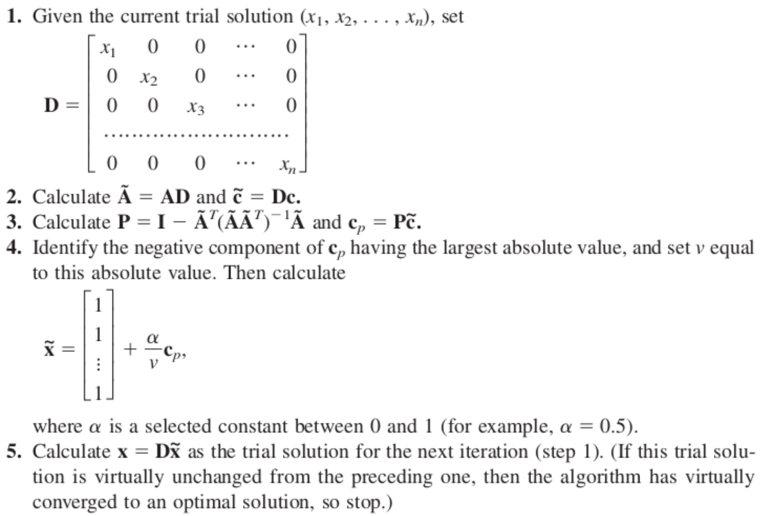
$$\max z = x_1 + 2x_2$$
$$x_1 + x_2 + x_3 = 8$$
$$x_1 \geq 0, x_2 \geq 0, x_3 \geq 0$$





x_3
8
(2, 2, 4)
(2, 3, 3)
(3, 4, 4)
0
8
x_1
8
x_2
(0, 8, 0) optimal

Affine-Scaling Method



1. Given the current trial solution $(x_1, x_2, \ldots, x_n)$, set
$$\mathbf{D} = \begin{bmatrix} x_1 & 0 & 0 & \cdots & 0 \\ 0 & x_2 & 0 & \cdots & 0 \\ 0 & 0 & x_3 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & x_n \end{bmatrix}$$
2. Calculate $\tilde{\mathbf{A}} = \mathbf{AD}$ and $\tilde{\mathbf{c}} = \mathbf{Dc}$.
3. Calculate $\mathbf{P} = \mathbf{I} - \tilde{\mathbf{A}}^T(\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T)^{-1}\tilde{\mathbf{A}}$ and $\mathbf{c}_p = \mathbf{P}\tilde{\mathbf{c}}$.
4. Identify the negative component of $\mathbf{c}_p$ having the largest absolute value, and set v equal to this absolute value. Then calculate
$$\tilde{\mathbf{x}} = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} + \frac{\alpha}{v}\mathbf{c}_p,$$
where α is a selected constant between 0 and 1 (for example, $\alpha = 0.5$).
5. Calculate $\mathbf{x} = \mathbf{D}\tilde{\mathbf{x}}$ as the trial solution for the next iteration (step 1). (If this trial solution is virtually unchanged from the preceding one, then the algorithm has virtually converged to an optimal solution, so stop.)

Watch out the change of notation:

$$\xi \to \tilde{x}$$

$$\Delta\xi \to c_p$$

$$\theta = \frac{1}{v} = \frac{1}{\max_j\{-\Delta\xi_j\}} = \\ = \frac{1}{\min_j\{-\Delta x_j / x_j^k\}}$$

$$\mathbf{D} = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 4 \end{bmatrix}.$$

The rescaled variables then are the components of

$$\tilde{\mathbf{x}} = \mathbf{D}^{-1}\mathbf{x} = \begin{bmatrix} \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{4} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \frac{x_1}{2} \\ \frac{x_2}{2} \\ \frac{x_3}{4} \end{bmatrix}.$$

In these new coordinates, **A** and **c** have become

$$\tilde{\mathbf{A}} = \mathbf{AD} = [1 \quad 1 \quad 1] \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 4 \end{bmatrix} = [2 \quad 2 \quad 4],$$

$$\tilde{\mathbf{c}} = \mathbf{Dc} = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 4 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 \\ 4 \\ 0 \end{bmatrix}.$$

Therefore, the projection matrix is

$$\mathbf{P} = \mathbf{I} - \tilde{\mathbf{A}}^T(\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T)^{-1}\tilde{\mathbf{A}}$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} - \begin{bmatrix} 2 \\ 2 \\ 4 \end{bmatrix} \left([2 \quad 2 \quad 4] \begin{bmatrix} 2 \\ 2 \\ 4 \end{bmatrix} \right)^{-1} [2 \quad 2 \quad 4]$$

$$\mathbf{P} = \mathbf{I} - \tilde{\mathbf{A}}^T(\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T)^{-1}\tilde{\mathbf{A}}$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} - \begin{bmatrix} 2 \\ 2 \\ 4 \end{bmatrix} \left([2 \quad 2 \quad 4] \begin{bmatrix} 2 \\ 2 \\ 4 \end{bmatrix} \right)^{-1} [2 \quad 2 \quad 4]$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} - \frac{1}{24} \begin{bmatrix} 4 & 4 & 8 \\ 4 & 4 & 8 \\ 8 & 8 & 16 \end{bmatrix} = \begin{bmatrix} \frac{5}{6} & -\frac{1}{6} & -\frac{1}{3} \\ -\frac{1}{6} & \frac{5}{6} & -\frac{1}{3} \\ -\frac{1}{3} & -\frac{1}{3} & \frac{1}{3} \end{bmatrix},$$

so that the projected gradient is

$$\mathbf{c}_p = \mathbf{P}\tilde{\mathbf{c}} = \begin{bmatrix} \frac{5}{6} & -\frac{1}{6} & -\frac{1}{3} \\ -\frac{1}{6} & \frac{5}{6} & -\frac{1}{3} \\ -\frac{1}{3} & -\frac{1}{3} & \frac{1}{3} \end{bmatrix} \begin{bmatrix} 2 \\ 4 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 3 \\ -2 \end{bmatrix}.$$

Define v as the *absolute value* of the *negative* component of $\mathbf{c}_p$ having the *largest* absolute value, so that $v = |-2| = 2$ in this case. Consequently, in the current coordinates, the algorithm now moves from the current trial solution $(\tilde{x}_1, \tilde{x}_2, \tilde{x}_3) = (1, 1, 1)$ to the next trial solution

$$\tilde{\mathbf{x}} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} + \frac{\alpha}{v}\mathbf{c}_p = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} + \frac{0.5}{2} \begin{bmatrix} 1 \\ 3 \\ -2 \end{bmatrix} = \begin{bmatrix} \frac{5}{4} \\ \frac{7}{4} \\ \frac{1}{2} \end{bmatrix},$$

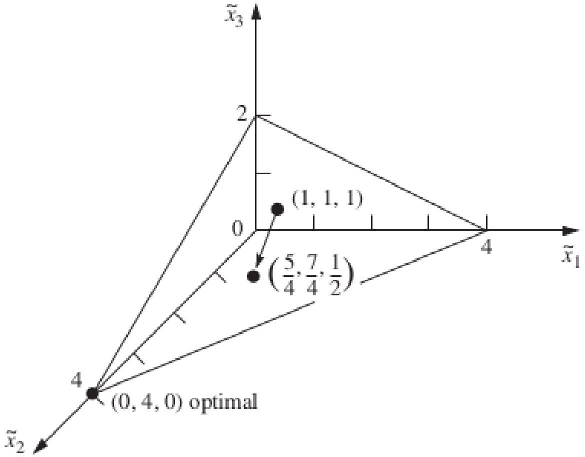
as shown in Fig. 8.5. (The definition of v has been chosen to make the smallest component of $\tilde{\mathbf{x}}$ equal to zero when $\alpha = 1$ in this equation for the next trial solution.) In the original coordinates, this solution is

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \mathbf{D}\tilde{\mathbf{x}} = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 4 \end{bmatrix} \begin{bmatrix} \frac{5}{4} \\ \frac{7}{4} \\ \frac{1}{2} \end{bmatrix} = \begin{bmatrix} \frac{5}{2} \\ \frac{7}{2} \\ 2 \end{bmatrix}.$$

This completes the iteration, and this new solution will be used to start the next iteration.

These steps can be summarized as follows for any iteration.

Example


$\tilde{x}_3$
2
(1, 1, 1)
0
4
$\tilde{x}_1$
$\left(\frac{5}{4}, \frac{7}{4}, \frac{1}{2}\right)$
4
(0, 4, 0) optimal
$\tilde{x}_2$

Iteration 2

Step 1:

Given the current trial solution $(x_1, x_2, x_3) = (\frac{5}{2}, \frac{7}{2}, 2)$, set

$$\mathbf{D} = \begin{bmatrix} \frac{5}{2} & 0 & 0 \\ 0 & \frac{7}{2} & 0 \\ 0 & 0 & 2 \end{bmatrix}.$$

(Note that the rescaled variables are

$$\begin{bmatrix} \tilde{x}_1 \\ \tilde{x}_2 \\ \tilde{x}_3 \end{bmatrix} = \mathbf{D}^{-1}\mathbf{x} = \begin{bmatrix} \frac{2}{5} & 0 & 0 \\ 0 & \frac{2}{7} & 0 \\ 0 & 0 & \frac{1}{2} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \frac{2}{5}x_1 \\ \frac{2}{7}x_2 \\ \frac{1}{2}x_3 \end{bmatrix},$$

so that the BF solutions in these new coordinates are

$$\tilde{\mathbf{x}} = \mathbf{D}^{-1}\begin{bmatrix} 8 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} \frac{16}{5} \\ 0 \\ 0 \end{bmatrix}, \qquad \tilde{\mathbf{x}} = \mathbf{D}^{-1}\begin{bmatrix} 0 \\ 8 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ \frac{16}{7} \\ 0 \end{bmatrix},$$

and

$$\tilde{\mathbf{x}} = \mathbf{D}^{-1}\begin{bmatrix} 0 \\ 0 \\ 8 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 4 \end{bmatrix},$$

as depicted in Fig. 8.6.)

Step 2:

$$\tilde{\mathbf{A}} = \mathbf{AD} = [\tfrac{5}{2}, \tfrac{7}{2}, 2] \qquad \text{and} \qquad \tilde{\mathbf{c}} = \mathbf{Dc} = \begin{bmatrix} \frac{5}{2} \\ 7 \\ 0 \end{bmatrix}.$$

Step 3:

$$\mathbf{P} = \begin{bmatrix} \frac{13}{18} & -\frac{7}{18} & -\frac{2}{9} \\ -\frac{7}{18} & \frac{41}{90} & -\frac{14}{45} \\ -\frac{2}{9} & -\frac{14}{45} & \frac{37}{45} \end{bmatrix} \qquad \text{and} \qquad \mathbf{c}_p = \begin{bmatrix} -\frac{11}{12} \\ \frac{133}{60} \\ -\frac{41}{15} \end{bmatrix}.$$

Step 4:

$\left|-\frac{41}{15}\right| > \left|-\frac{11}{12}\right|$, so $v = \frac{41}{15}$ and

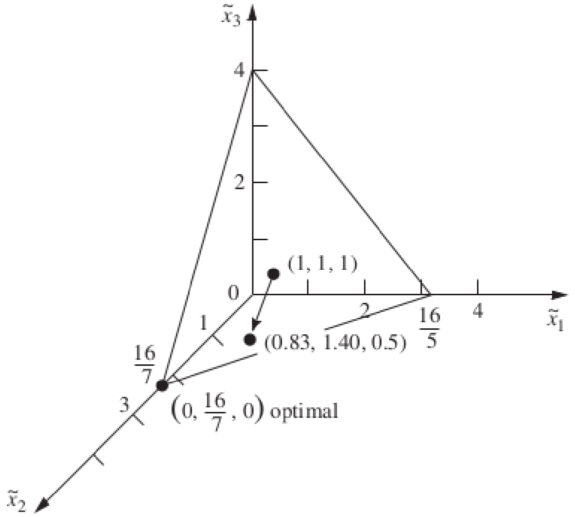
$$\tilde{\mathbf{x}} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} + \frac{0.5}{\frac{41}{15}} \begin{bmatrix} -\frac{11}{12} \\ \frac{133}{60} \\ -\frac{41}{15} \end{bmatrix} = \begin{bmatrix} \frac{273}{328} \\ \frac{461}{328} \\ \frac{1}{2} \end{bmatrix} \approx \begin{bmatrix} 0.83 \\ 1.40 \\ 0.50 \end{bmatrix}.$$

Step 5:

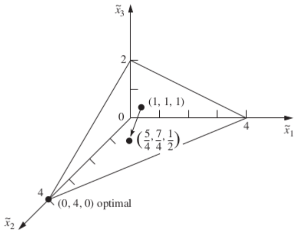
$$\mathbf{x} = \mathbf{D}\tilde{\mathbf{x}} = \begin{bmatrix} \frac{1365}{656} \\ \frac{3227}{656} \\ 1 \end{bmatrix} \approx \begin{bmatrix} 2.08 \\ 4.92 \\ 1.00 \end{bmatrix}$$

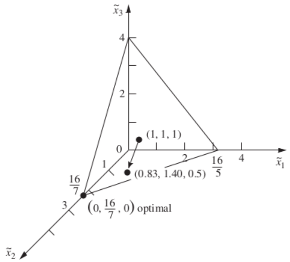
is the trial solution for iteration 3.

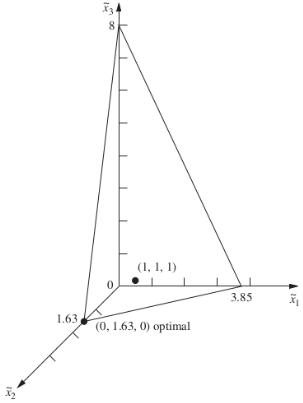
Example


$\tilde{x}_3$
4
2
0
(1, 1, 1)
2
$\frac{16}{5}$
4
$\tilde{x}_1$
1
(0.83, 1.40, 0.5)
$\frac{16}{7}$
3
$\left(0, \frac{16}{7}, 0\right)$ optimal
$\tilde{x}_2$

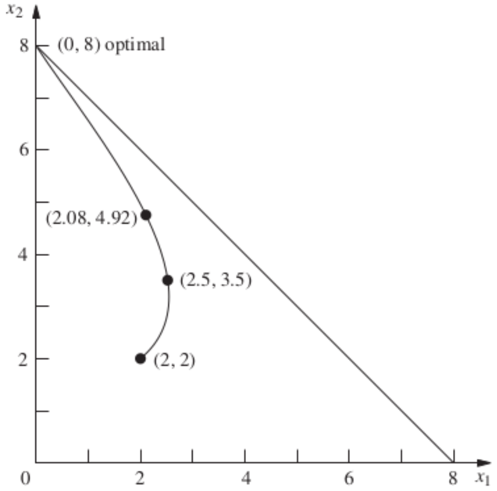
Example


$\tilde{x}_3$
2
0
(1, 1, 1)
4
$\tilde{x}_1$
$\left(\frac{5}{4}, \frac{7}{4}, \frac{1}{2}\right)$
4
(0, 4, 0) optimal
$\tilde{x}_2$


$\tilde{x}_3$
4
2
0
(1, 1, 1)
2
$\frac{16}{5}$
4
$\tilde{x}_1$
1
(0.83, 1.40, 0.5)
$\frac{16}{7}$
3
$\left(0, \frac{16}{7}, 0\right)$ optimal
$\tilde{x}_2$


$\tilde{x}_3$
8
(1, 1, 1)
0
3.85
$\tilde{x}_1$
1.63
(0, 1.63, 0) optimal
$\tilde{x}_2$

Example



Projection Calculations

Projection matrix:

$$P = (I - \tilde{A}^T(\tilde{A}\tilde{A}^T)^{-1}\tilde{A})$$

$$\Delta\boldsymbol{\xi} = P\tilde{\boldsymbol{c}} = \tilde{\boldsymbol{c}} - \tilde{A}^T(\tilde{A}\tilde{A}^T)^{-1}\tilde{A}\tilde{\boldsymbol{c}}$$

Solved by:

- $\tilde{A}\tilde{\boldsymbol{c}} = \boldsymbol{v}$
- $\boldsymbol{w} = (\tilde{A}\tilde{A}^T)^{-1}\boldsymbol{v}$ solved as $(\tilde{A}\tilde{A}^T)\boldsymbol{w} = \boldsymbol{v}$ by Cholesky decomposition
- $\tilde{\boldsymbol{c}} - \tilde{A}^T\boldsymbol{w}$

Phase I: Feasibility Direction

To derive a Phase I procedure for the affine-scaling algorithm, we consider a starting point $\boldsymbol{x}^0$ that has strictly positive components but does not necessarily satisfy the equality constraints $A\boldsymbol{x} = \boldsymbol{b}$. We then let

$$\rho = \boldsymbol{b} - A\boldsymbol{x}^0$$

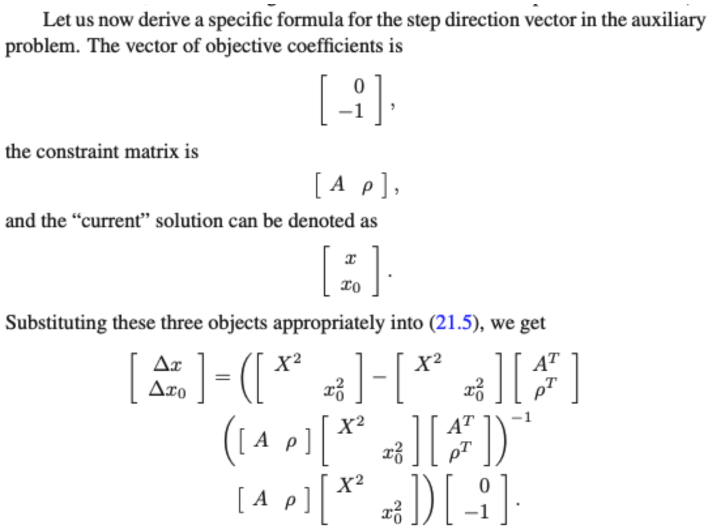
denote the vector of infeasibilities.

Auxiliary problem:

$$\begin{aligned} \max \ & -\boldsymbol{x}_0 \\ & A\boldsymbol{x} + \boldsymbol{x}_0 \rho = \boldsymbol{b} \\ & \boldsymbol{x} \geq 0, \boldsymbol{x}_0 \geq 0 \end{aligned}$$

$\begin{bmatrix} \boldsymbol{x}^0 \\ 1 \end{bmatrix}$ is feasible solution

If $\boldsymbol{x}_0^* > 0$, then the original problem is infeasible.
If $\boldsymbol{x}_0^* = 0$, then the optimal solution to the auxiliary problem provides a feasible starting solution to the original problem (it may not be a strictly interior feasible solution)



Let us now derive a specific formula for the step direction vector in the auxiliary problem. The vector of objective coefficients is

$$\begin{bmatrix} 0 \\ -1 \end{bmatrix},$$

the constraint matrix is

$$\begin{bmatrix} A & \rho \end{bmatrix},$$

and the "current" solution can be denoted as

$$\begin{bmatrix} x \\ x_0 \end{bmatrix}.$$

Substituting these three objects appropriately into (21.5), we get

$$\begin{bmatrix} \Delta x \\ \Delta x_0 \end{bmatrix} = \left(\begin{bmatrix} X^2 & \\ & x_0^2 \end{bmatrix} - \begin{bmatrix} X^2 & \\ & x_0^2 \end{bmatrix} \begin{bmatrix} A^T \\ \rho^T \end{bmatrix} \right. \\ \left(\begin{bmatrix} A & \rho \end{bmatrix} \begin{bmatrix} X^2 & \\ & x_0^2 \end{bmatrix} \begin{bmatrix} A^T \\ \rho^T \end{bmatrix} \right)^{-1} \\ \left. \begin{bmatrix} A & \rho \end{bmatrix} \begin{bmatrix} X^2 & \\ & x_0^2 \end{bmatrix} \right) \begin{bmatrix} 0 \\ -1 \end{bmatrix}.$$

which yeilds $\Delta \boldsymbol{x} = X^2 A^T (A X^2 A^T)^{-1} \rho$.