NTNU | Norwegian University of Science and Technology

LINEAR PROGRAMMING

Jørn Vatn/August-2026

Introduction to programming problems

Programming problems deals with the use or allocation of resources in the best possible manner in terms of minimizing cost or maximizing profit. Resources are materials, labour, machines, transportation capacity, energy, capital and so forth. In general resources are limited. In this course four types of programming are introduced:

- Linear programming (today)
- Dynamic programming (covered later)
- Non-linear programming (covered later)
- Stochastic programming (covered later)

Linear programming

Linear programming deals with problems defined by the following conditions:

1. The decision variables are non-negative
2. The objective function is given as a *linear function of the decision variables*. The linear assumption implies that only the first powers of the decision variables are included, and no cross terms are allowed.
3. Constraints in terms of limitation of resources can be expressed as a *set of linear equations or linear inequalities*.

LP problems are solved by hand (SIMPLEX), Excel or Python.

Motivating example

Consider the shipment of goods from Trondheim to Oslo

Max 100 units

Motivating example

- Consider the shipment of goods from Trondheim to Oslo
- The truck has a capacity to take 100 units
- There are four products types that could be transported. The *profit* per unit depends on the the product type and are given by $c_1 = 10$, $c_2 = 5$, $c_3 = 8$ and $c_4 = 4$ for product type $1, 2, \dots, 4$. Let x_j denote the number of units to take of product type j.
- The objective function is then given by:

$$Z(x_1, x_2, x_3, x_4) = 10x_1 + 5x_2 + 8x_3 + 4x_4 = c_1x_1 + c_2x_2 + c_3x_3 + c_4x_4$$

Motivating example, cont.

There are some constraints in addition to the truck capacity. The customer in Oslo has storage restrictions implying that $2x_1 + 3x_2$ cannot exceed 50. Further there is a maximum of 50 units available in Trondheim of product type 3. The constrains can thus be formulated by:

$$\begin{aligned} x_1 + x_2 + x_3 + x_4 &\leq 100 = b_1 \\ 2x_1 + 3x_2 &\leq 50 = b_2 \\ x_3 &\leq 50 = b_3 \end{aligned}$$

In this problem it is quite obvious that the number of units cannot be negative, hence $x_1 \geq 0, x_2 \geq 0, \ldots, x_4 \geq 0$

▸ Excel solution ▸ Python solution

Linear programming problem on standard form

A linear programming problem generally comprises an *objective function* to maximize or minimize, and a *set of constraints*. The objective function, $Z()$ is always a function of the decision variables $x_j, j = 1, \dots, n$ but usually we do not explicitly state this relation on the left hand side of the equation:

$$\text{Maximize: } Z = c_1 x_1 + c_2 x_2 + \cdots + c_n x_n$$

$$\begin{aligned}
\text{Subject to: } a_{11} x_1 + a_{12} x_2 + \cdots + a_{1n} x_n &= b_1 \\
a_{21} x_1 + a_{22} x_2 + \cdots + a_{2n} x_n &= b_2 \\
&\vdots \\
a_{m1} x_1 + a_{m2} x_2 + \cdots + a_{mn} x_n &= b_m \\
x_1 \geq 0, x_2 \geq 0, \dots, x_n &\geq 0 \\
b_1 \geq 0, b_2 \geq 0, \dots, b_m &\geq 0
\end{aligned}$$

Convert to standard form

Short version

$$\text{Maximize: } Z = \sum_{j=1}^{n} c_j x_j$$

$$\text{Subject to: } \sum_{j=1}^{n} a_{ij} x_j = b_i, i = 1 : m$$

$$x_j \geq 0, j = 1 : n$$

$$b_i \geq 0, i = 1 : m$$

Linear programming problem on matrix form

$$\text{Maximize: } Z = \mathbf{cx}$$

$$\begin{aligned} \text{Subject to: } \mathbf{Ax} &= \mathbf{b} \\ \mathbf{x} &\geq \mathbf{0} \\ \mathbf{b} &\geq \mathbf{0} \end{aligned}$$

Number of constraints vs number of non-negative variables

- If $m = 0$, no solution exists if at least one $c_j > 0$
- If $m = 1$:
  - There will be one $x_j > 0$, say $j = k$, and $x_j = 0$ for $j \neq k$
  - The constraint coefficients need to bound all x_j's where $c_j > 0$
  - The positive x_j will be the one with the largest corresponding c_j
- If $m = 2$, more than one x_j could be > 0, and maximum two x_j's are > 0
- In general we will never have more x_j's > 0 than m, and usually the number of positive x_j's $= m$

Converting to standard form

The motivating example was not exactly on standard form because there were some inequalities in the constraints, i.e., $\cdots \leq b_i$ rather than $\cdots = b_i$. In other cases there are other types of restrictions that do not completely match the standard format of the problem. The following steps may be used to transform a linear programming problem into standard form:

- If the problem is to minimize Z convert the problem to a problem that maximizes $-Z$.
- If there is a less or equal inequality in one or more rows of the constraints, i.e., $\sum_{j=1}^{n} a_{ij}x_j \leq b_i$, convert it into an equality constraint by adding a nonnegative *slack* variable s_i yielding the resulting constraint: $\sum_{j=1}^{n} a_{ij}x_j + s_i = b_i$, where $s_i \geq 0$.

Converting to standard form, cont.

- If there is a greater or equal inequality in one or more rows of the constraints, i.e., $\sum_{j=1}^{n} a_{ij}x_j \geq b_i$, convert it into an equality constraint by subtracting a nonnegative *surplus* variable s_i yielding the resulting constraint: $\sum_{j=1}^{n} a_{ij}x_j - s_i = b_i$, where $s_i \geq 0$.
- If some of the b_i's are negative, multiply that constraint equation by -1.
- If one of the decision variables, say x_j is unrestricted in sign, replace it by $x_j^a - x_j^b$ everywhere in the problem formulation and add to the constraints: $x_j^a \geq 0$ and $x_j^b \geq 0$.

Note that s_i, x_j^a and x_j^b will be renamed to maintain a set of n decision variables, $x_j, j = 1, 2, \ldots, n$.

SIMPLEX

- The SIMPLEX method is used to maximize the objective function with constraints
- The term *system* is used to denote an objective function with related constraints
- Two systems are said to be equivalent if they have the same solution set
- In the SIMPLEX method the original system is in a stepwise manner transferred to equivalent systems until we easily can read the solution from such an equivalent system
  - Such a procedure is similar to the classical Gauss-Jordan elimination procedure for solving a system of linear equations, i.e., we may:
  - Multiply any equation in the system by a constant
  - Add to any equation a constant multiple of any other equation in the system

SIMPLEX, cont

- It is an objective in this course to be able to solve a LP-problem by the SIMPLEX method - i.e., “hand calculation” on the exam
- Note that in some textbooks the SIMPLEX method is formulated slightly differently. If you have learned an alternative formulation, use it!
- When we have learned the SIMPLEX, we stick to Excel or other tools for the tedious work, and the focus is on the modelling part

Motivation 1

If we have a system where the constraint coefficients look like:

$$\text{Subject to: } \begin{aligned} a_{11}x_1 + 0x_2 + 0x_3 + 1x_4 &= b_1 \\ a_{21}x_1 + 0x_2 + 1x_3 + 0x_4 &= b_2 \\ a_{31}x_1 + 1x_2 + 0x_2 + 0x_4 &= b_3 \end{aligned}$$

and if we know that *only* x_2, x_3 and x_4 are > 0, then we easily read the solution

$$\begin{aligned} x_4 &= b_1 \\ x_3 &= b_2 \\ x_2 &= b_3 \end{aligned}$$

Motivation 2

$$
\begin{aligned}
\text{Subject to: } a_{11}x_1 + 0x_2 + 0x_3 + 1x_4 &= b_1 \\
a_{21}x_1 + 0x_2 + 1x_3 + 0x_4 &= b_2 \\
a_{31}x_1 + 1x_2 + 0x_2 + 0x_4 &= b_3
\end{aligned}
$$

- The zeros (0) and ones (1) in this equation is the *magic*, and since we know that maximum $m(=3)$ positive x_j's, we need m x_j's having coefficients 0 or 1 in order to read the solution directly from the constraints equation
- We are allowed to manipulate the constraint equations:
  - We need m variables with coefficients 0 or 1, i.e., basic variable
  - We can switch which set of variables are basic variables, if this increases Z

Definitions

- *Basic variable*. A variable x_j is a basic variable if and only if $a_{ij} = 1$ for exactly one equation, say i, and $a_{kj} = 0$ for all other equations, i.e., $k \neq i$.
- A *pivot operation* for a specific variable is a set of matrix operations where one system is transformed to an equivalent system where that variable becomes a basic variable
- A *canonical system* is a system where there is at least one basic variable in each equation. This means that the number of basic variables equals the number of equations.
- A *basic solution* is obtained from a canonical system by setting the non-basic variables to zero and solving for the basic variables
- A *basic feasible solution* is a basic solution where the basic variables are non-negative.

Manipulation of constraints equations, recall:

$$
\begin{aligned}
\text{Subject to: } & a_{11}x_1 + 0x_2 + 0x_3 + 1x_4 = b_1 \qquad (1) \\
& a_{21}x_1 + 0x_2 + 1x_3 + 0x_4 = b_2 \qquad (2) \\
& a_{31}x_1 + 1x_2 + 0x_2 + 0x_4 = b_3 \qquad (3)
\end{aligned}
$$

- x_2, x_3 and x_4 are basic variables, whereas x_1 is a non-basic variable which always takes the value 0
- We can swap e.g., x_1 and x_3 and test if this gives a better value of Z
- Such an operation is denoted a *pivot operation*
- To do this we can
  - Multiply any equation in the system by a constant
  - Add to any equation a constant multiple of any other equation in the system

Manipulation of constraints equations - Demonstration

$$\text{Subject to: } a_{11}x_1 + 0x_2 + 0x_3 + 1x_4 = b_1 \tag{1}$$

$$1x_1 + (0/a_{21})x_2 + (1/a_{21})x_3 + (0/a_{21})x_4 = b_2/a_{21} \tag{2}$$

$$a_{31}x_1 + 1x_2 + 0x_3 + 0x_4 = b_3 \tag{3}$$

- For (2): Divide by a_{21}
- This gives the new $a'_{21} = 1$ in front of x_1
- For x_2 and x_4 we *maintained* 0 in front of the variables as required for the *remaining* basic variables for this equation

Manipulation of constraints equations - Eq. (2) done

$$
\begin{aligned}
\text{Subject to: } & a_{11}x_1 + 0x_2 + 0x_3 \quad +1x_4 = b_1 \quad (1)\\
& 1x_1 + 0x_2 + a'_{23}x_3 + 0x_4 = b'_2 \quad (2)\\
& a_{31}x_1 + 1x_2 + 0x_3 \quad +0x_4 = b_3 \quad (3)
\end{aligned}
$$

- For (1) and (3), we need to get rid of a_{11} and a_{31}
- For (1): Subtract (2) $\times$ a_{11}, i.e., (1) $\leftarrow$ (1) - (2)$\times a_{11}$

Manipulation of constraints equations - Eq. (1):

$$
\begin{aligned}
\text{Subject to: } (a_{11} - a_{11})x_1 + (0 - a_{11}0)x_2 + (0 - a_{11}a'_{23})x_3 + (1 - a_{11}0)x_4 &= b_1 - a_{11}b_2 \\
1x_1 + 0x_2 + a'_{23}x_3 + 0x_4 &= b'_2 \\
a_{31}x_1 + 1x_2 + 0x_3 + 0x_4 &= b_3
\end{aligned}
$$

rewriting:

$$
\begin{aligned}
\text{Subject to: } 0x_1 + 0x_2 + a'_{13}x_3 + 1x_4 &= b'_1 \\
1x_1 + 0x_2 + a'_{23}x_3 + 0x_4 &= b'_2 \\
a_{31}x_1 + 1x_2 + 0x_3 + 0x_4 &= b_3
\end{aligned}
$$

repeat for (3)..., and we are done

SIMPLEX idea

- Ensure that the system is on standard form,
  Max: $Z = \mathbf{cx}$, s.t.: $\mathbf{Ax} = \mathbf{b} \geq \mathbf{0}, \mathbf{x} \geq \mathbf{0}$
- Manipulate to get m basic variables, i.e., a system on canonical form
- For a system on canonical form we can “read” the solution, i.e., we find a feasible solution
- We try to improve the feasible solution by swapping basic variables
  - Find the non-basic variable that would be best to change to a basic variable
  - We need to “kick out” one basic variable, find the most appropriate one
- Repeat until we can not improve the solution

Demonstration of SIMPLEX by our transport example

- Introduce slack variables x_5, x_6 and x_7 to get on standard form:

$$\text{Maximize: } Z = 10x_1 + 5x_2 + 8x_3 + 4x_4 + 0x_5 + 0x_6 + 0x_7$$

$$\begin{aligned}\text{Subject to: } x_1 + x_2 + x_3 + x_4 + 1x_5 + 0x_6 + 0x_7 &= 100 \\ 2x_1 + 3x_2 + 0x_3 + 0x_4 + 0x_5 + 1x_6 + 0x_7 &= 50 \\ 0x_1 + 0x_2 + x_3 + 0x_4 + 0x_5 + 0x_6 + 1x_7 &= 50\end{aligned}$$

Tableau without slack variables

- The problem has now been formulated on standard form and the manipulation of equations may start
- A so-called *tableau* is used to organize the equations and intermediate results
- The figure shows the initial tableau where row 2 contains the objective function coefficients, and rows 7-9 contain the constraint coefficients
- To keep track of the basic variables, k_i, column B for rows 7-9 are used for this purpose.

	A	B	C	D	E	F	G	H
1	Variables:				x_1	x_2	x_3	x_4
2	Objective function coefficients:				10	5	8	4
3	Variable values:		Z=	0				
4	Relative profit:							
5	Number of variables	7						
6	Number of constraints	3						
7	Constraints coefficients	Basic var.	Upper limit	b_i	a_{i1}	a_{i2}	a_{i3}	a_{i4}
8				100	1	1	1	1
9				50	2	3	0	0
10				50	0	0	1	0

Tableau with slack variables

Variables:				x_1	x_2	x_3	x_4	x_5	x_6	x_7
Objective function coefficients:				10	5	8	4	0	0	0
Variable values:		Z=	0					100	50	50
Relative profit:										
Number of variables	7									
Number of constraints	3									
Constraints coefficients	Basic var.	Upper limit	b_i	a_{i1}	a_{i2}	a_{i3}	a_{i4}	a_{i5}	a_{i6}	a_{i7}
	5		100	1	1	1	1	1	0	0
	6		50	2	3	0	0	0	1	0
	7		50	0	0	1	0	0	0	1

- We have added slack variables x_5, x_6 and x_7, $\Longrightarrow$ "magic" basic variables
- Inserting values for basic variables results in zero profit, i.e., $Z = \mathbf{cx} = \mathbf{0}$
- It is time to calculate *relative profit* for the non-basic variables

Improving the solution

- For each non-basic variable x_j consider to increase it with one unit, i.e., from 0 to 1
- All basic variables must then be reduced to maintain the constraints
- Let x_{k_i} be the basic variable in constraint equation i. Since per definition a_{i,k_i} equals one we need to reduce the value of x_{k_i} by $a_{i,j}$
- Such a reduction will reduce the objective function by $c_{k_i} a_{i,j}$
- The *relative profit* by increasing x_j by one unit is then:

$$\mathrm{RP}_j = c_j - \sum_{i=1}^{m} c_{k_i} a_{i,j}$$

We calculate the relative profit for each non-basic variables, and proceed with the best one

Demonstration for variable x_1

$$\mathrm{RP}_1 = c_1 - c_5 \times a_{1,1} - c_6 \times a_{2,1} - c_7 \times a_{3,1} = 10 - 0 \times 1 - 0 \times 2 - 0 \times 0 = 10$$

- c_1 = increased value of Z by increasing x_1 from 0 to 1
- $c_5 \times a_{1,1}$ = is what we loose to fulfil constraint equation 1
- $c_6 \times a_{2,1}$ = is what we loose to fulfil constraint equation 2
- $c_7 \times a_{3,1}$ = is what we loose to fulfil constraint equation 3

VBA Code, RP

```
Function jRP(A(), b(), c(), n, m, k_i())
For j = 1 To n
    RP = c(j)
    For i = 1 To m
        RP = RP - c(k_i(i)) * A(i, j)
    Next i
    If RP > bestRP Then
        bestRP = RP
        jReturn = j
    End If
Next j
jRP = bestJ
End Function
```

Tableau with relative profits calculated

Variables:				x_1	x_2	x_3	x_4	x_5	x_6	x_7
Objective function coefficients:				10	5	8	4	0	0	0
Variable values:		Z=	0					100	50	50
Relative profit:				10	5	8	4	0	0	0
Number of variables	7									
Number of constraints	3									
Constraints coefficients	Basic var.	Upper limit	b_i	a_{i1}	a_{i2}	a_{i3}	a_{i4}	a_{i5}	a_{i6}	a_{i7}
	5		100	1	1	1	1	1	0	0
	6		50	2	3	0	0	0	1	0
	7		50	0	0	1	0	0	0	1

- Since $c_{k_i} = 0$ for the all initial basic variables in the example, the relative profit equals the corresponding coefficients in the objective function
- Since RP_1 has the *highest* value, x_1 is the non-basic variable to add to the system

Which basic variable to remove?

- To find out which basic variable to remove we calculate an *upper limit* for the increase of x_j ($j = 1$ in the example) for each constraint equation such that the basic variable is still non-negative
- For equation i we only consider the variables x_j and x_{k_i}, where x_{k_i} may be reduced to 0, hence

$$a_{i,j}x_j + a_{k_i,j}x_{k_i} = b_i$$
$$\Rightarrow \mathrm{UL}_i = b_i / a_{i,j}$$

yielding

Variables:				x_1	x_2	x_3	x_4	x_5	x_6	x_7
Objective function coefficients:				10	5	8	4	0	0	0
Variable values:		Z=	0					100	50	50
Relative profit:				10	5	8	4	0	0	0
Number of variables	7									
Number of constraints	3									
Constraints coefficients	Basic var.	Upper limit	b_i	a_{i1}	a_{i2}	a_{i3}	a_{i4}	a_{i5}	a_{i6}	a_{i7}
	5	100	100	1	1	1	1	1	0	0
	6	25	50	2	3	0	0	0	1	0
	7	1E+30	50	0	0	1	0	0	0	1

Which basic variable to remove?

Variables:				x_1	x_2	x_3	x_4	x_5	x_6	x_7
Objective function coefficients:				10	5	8	4	0	0	0
Variable values:		Z=	0					100	50	50
Relative profit:				10	5	8	4	0	0	0
Number of variables	7									
Number of constraints	3									
Constraints coefficients	Basic var.	Upper limit	b_i	a_{i1}	a_{i2}	a_{i3}	a_{i4}	a_{i5}	a_{i6}	a_{i7}
	5	100	100	1	1	1	1	1	0	0
	6	25	50	2	3	0	0	0	1	0
	7	1E+30	50	0	0	1	0	0	0	1

- We see that it is constraint equation number 2 which limits the increase and hence the improvement in the objective function
- Basic variable x_6 in the second row of constraint equations will now be replaced by x_1 which will be the new basic variable
- The second row is denoted the *pivot row* in the system

VBA Code, UL

```
Function iUL(j , A(), b(), n , m)
lowestUL = 1E+30
For i = 1 To m
    If A(i, j) > 0 Then
        UL = b(i) / A(i, j)
        If UL < lowestUL Then
            lowestUL = UL
            iReturn = i
        End If
    End If
Next
iUL = iReturn
End Function
```

Pivot operation

- A pivot operation is applied to make x_j, i.e., x_1 the new basic variable in the pivot row, i.e., $i = 2$
- Step 1: Divide all the numbers in the pivot row by $a_{i,j} = a_{2,1} = 2$
  - This will ensure that the coefficient of the new basic variable in the pivot row equals 1
- Step 2: For each of the other equations, say row k, the coefficient of the new basic variable should be 0
  - This is obtained by subtracting a constant multiple of the pivot row equation to each row
  - The multiplication constant is $a_{k,j}$ for equation k

Tableau with variable x_1 inserted as the new basic variable

Variables:				x_1	x_2	x_3	x_4	x_5	x_6	x_7
Objective function coefficients:				10	5	8	4	0	0	0
Variable values:		Z=	250	25	0	0	0	75	0	50
Relative profit:				10	5	8	4	0	0	0
Number of variables	7									
Number of constraints	3									
Constraints coefficients	Basic var.	Upper limit	b_i	a_{i1}	a_{i2}	a_{i3}	a_{i4}	a_{i5}	a_{i6}	a_{i7}
	5	100	75	0	-1	1	1	1	-1	0
	1	25	25	1	1.5	0	0	0	0.5	0
	7	1E+30	50	0	0	1	0	0	0	1

- When the new values of the basic variables are calculated we see that the objective function has increased from 0 to 250

VBA Code, Pivot

```
Function doPivot(i, j, A(), b(), n, m)
h = A(i, j)
b(i) = b(i) / h
For jj = 1 To n
    A(i, jj) = A(i, jj) / h
Next jj
For k = 1 To m
    If k <> i Then
        h = A(k, j)
        b(k) = b(k) - h * b(i)
        For jj = 1 To n
          A(k, jj) = A(k, jj) - h * A(i, jj)
        Next jj
    End If
Next
End Function
```

Towards the final solution

- In the compendium we show the final steps to complete
- First variable x_3 should replace variable x_7 in row 3
- Then variable x_4 will replace variable x_5 in row 1
- Then there are no non-basic variables that will improve the solution, and we are done

Variables:				x_1	x_2	x_3	x_4	x_5	x_6	x_7
Objective function coefficients:				10	5	8	4	0	0	0
Variable values:			Z= 750	25	0	50	25	0	0	0
Relative profit:				0	-8	0	0	-4	-3	-4
Number of variables	7									
Number of constraints	3									
Constraints coefficients	Basic var.	Upper limit	b_i	a_{i1}	a_{i2}	a_{i3}	a_{i4}	a_{i5}	a_{i6}	a_{i7}
	4	25	25	0	-1	0	1	1	-1	-1
	1	1E+30	25	1	1.5	0	0	0	0.5	0
	3	1E+30	50	0	0	1	0	0	0	1

The final solution is given by $x_1 = 25$, $x_2 = 0$, $x_3 = 50$ and $x_4 = 25$ giving a final objective function equal to $Z = 750$ Excel

How to get an LP problem on standard form

- Add slack variables to translate $\leq$ to $=$
- Add surplus variables to translate $\geq$ to $=$
- Use “Big M” to get enough basic variables

How to get an LP problem on standard form - Example

Maximize: $Z = c_1 x_1 + c_2 x_2 + c_3 x_3$

Subject to: $a_{11} x_1 + a_{12} x_2 + a_{13} x_3 \leq b_1$

$a_{21} x_1 + a_{22} x_2 + a_{23} x_3 \geq b_2$

Add slack variable

Maximize: $Z = c_1 x_1 + c_2 x_2 + c_3 x_3 + 0 x_4$

Subject to: $a_{11} x_1 + a_{12} x_2 + a_{13} x_3 + 1 x_4 = b_1$

$a_{21} x_1 + a_{22} x_2 + a_{23} x_3 + 0 x_4 \geq b_2$

Add surplus variable

$$\text{Maximize: } Z = c_1 x_1 + c_2 x_2 + c_3 x_3 + 0 x_4 + 0 x_5$$

$$\begin{aligned}\text{Subject to: } & a_{11} x_1 + a_{12} x_2 + a_{13} x_3 + \color{blue}{1} x_4 + 0 x_5 = b_1 \\ & a_{21} x_1 + a_{22} x_2 + a_{23} x_3 + \color{blue}{0} x_4 - 1 x_5 = b_2\end{aligned}$$

Note the -1 in from of the surplus variable x_5 prevents it from being a basic variable, we need another basic variable:

Add "Big M" variable

$$\text{Maximize: } Z = c_1 x_1 + c_2 x_2 + c_3 x_3 + 0 x_4 + 0 x_5 - M x_6$$

$$\begin{aligned} \text{Subject to: } a_{11} x_1 + a_{12} x_2 + a_{13} x_3 + 1 x_4 + 0 x_5 + 0 x_6 &= b_1 \\ a_{21} x_1 + a_{22} x_2 + a_{23} x_3 + 0 x_4 - 1 x_5 + 1 x_6 &= b_2 \end{aligned}$$

x_4 and x_6 are now basic variables, and the "Big M" variable x_6 will be removed in the first SIMPLEX iteration if M is large

Shadow prices

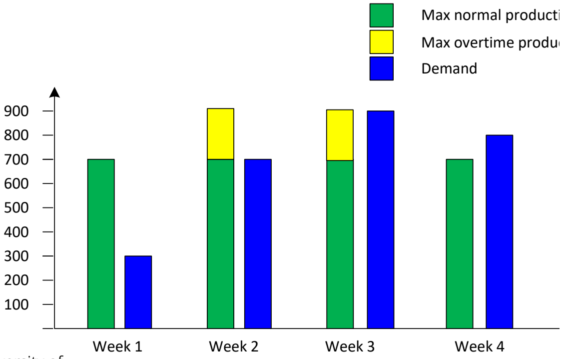
- In constrained optimization like an LP problem, the shadow price is the change, per infinitesimal unit of the constraint, in the optimal value of the objective function obtained by relaxing the constraint
- This typically means that the shadow price associated with a resource tells how much more profit one would get by increasing the amount of that resource by one unit, or put in another way, “How much one would be willing to pay for an additional resource”
- Calculating shadow prices can help us prioritizing where to start improving “framing conditions”, for example increase the capacity of the truck

Step by step - Excel

- The course compendium shows step by step what we need to do in Excel
- Here we demonstrate

Example - Production Scheduling

- We are scheduling the next 4 weeks of production
- The weekly demands are 300, 700, 900 and 800 items for the next 4 weeks
- Normal production cannot exceed 700 units per week, but it is possible to use overtime to produce up to 200 extra units per week in week 2 and 3



Example - Production Scheduling, cont

- We are scheduling the next 4 weeks of production
- The weekly demands are 300, 700, 900 and 800 items for the next 4 weeks
- Production cost is $5 for week 1 and 2, and $10 for week 3 and 4
- Normal production cannot exceed 700 units per week, but it is possible to use overtime to produce up to 200 extra units per week in week 2 and 3 but then at an additional item cost of $2
- Excess production can be stored at a cost of $3 per item (from one month to the other):

Variable/Quantity	Explanation
$x_i, i = 1:4$	Normal production week i
$d_i, i = 1:4$	Demand week i, = {300,700,900,800}
x_5, x_6	Overtime prod. in week 2 and 3 respectively
$x_i, i = 7:10$	Stock level at end of week $i - 6$

LP equations

Minimize: $Z = 5x_1 + 5x_2 + 10x_3 + 10x_4 + 7x_5 + 12x_6 + 3x_7 + 3x_8 + 3x_9 + 3x_{10}$

Subject to:
$$
\begin{aligned}
x_1 &\leq 700 \\
x_2 &\leq 700 \\
x_3 &\leq 700 \\
x_4 &\leq 700 \\
x_5 &\leq 200 \\
x_6 &\leq 200 \\
x_1 &= 300 + x_7 \\
x_2 + x_5 + x_7 &= 700 + x_8 \\
x_3 + x_6 + x_8 &= 900 + x_9 \\
x_4 + x_9 &= 800 + x_{10}
\end{aligned}
$$

LP equations, rearranging to clean the right hand side:

Minimize: $Z = 5x_1 + 5x_2 + 10x_3 + 10x_4 + 7x_5 + 12x_6 + 3x_7 + 3x_8 + 3x_9 + 3x_{10}$

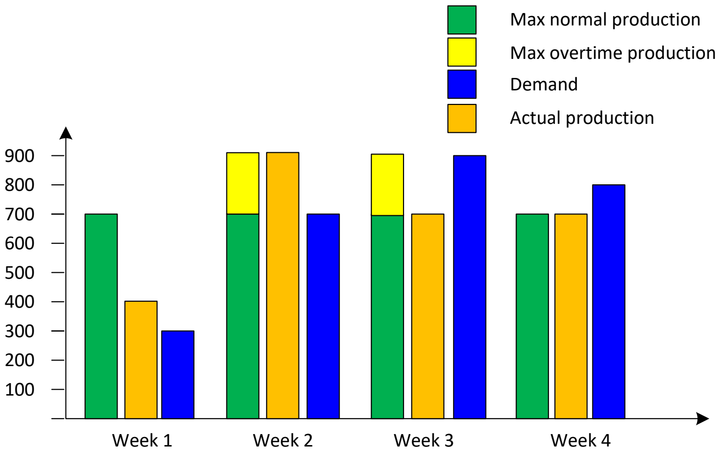
Subject to:
$$
\begin{aligned}
x_1 &\le 700 \\
x_2 &\le 700 \\
x_3 &\le 700 \\
x_4 &\le 700 \\
x_5 &\le 200 \\
x_6 &\le 200 \\
x_1 - x_7 &= 300 \\
x_2 + x_5 + x_7 - x_8 &= 700 \\
x_3 + x_6 + x_8 - x_9 &= 900 \\
x_4 + x_9 - x_{10} &= 800
\end{aligned}
$$

Excel

Variables	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}		
Objective function coefficients (c_j):	5	5	10	10	7	12	3	3	3	3		
x-Values	400	700	700	700	200	0	100	300	100	0		
$Z=$	22 400											
Constraint equations	a_{i1}	a_{i2}	a_{i3}	a_{i4}	a_{i5}	a_{i6}	a_{i7}	a_{i8}	a_{i9}	a_{i10}	b_i	$\Sigma a_{ij}x_j\text{-}b_i$
	1										700	-300
		1									700	0
			1								700	0
				1							700	0
					1						200	0
						1					200	-200
	1						-1				300	0
		1			1		1	-1			700	0
			1			1		1	-1		900	0
				1					1	-1	800	0

Show it in Excel

Solution



Show it in Python

Python considerations

- The standard Python approach for solving LP-problems is to use `linprog` from the `scipy.optimize` package
- In `linprog` one can specify **A** and **b** for both "upper bound" constraints and equality constraints.
- In the ▸ lpLib.py provided by TPK4191 a set of functions are provided where each constraints could be added to the model one by one
- in `lpLib.py` there is also a `balanceConstraint` function where it is possible to specify $\pm x_i, \pm x_j, \ldots, \pm x_n = b$ where $\{i, j, \ldots, n\}$ indicates a subset of the decision variables. Typically "holding start + produced + used = holding end".

Example - Product mix

A company manufactures three products, A, B and C. Below is shown the resources required for each product:

Company	Engineering hours	Direct labour hours	Materials (kg)
A	1	10	3
B	2	4	2
C	1	5	1

Example - Product mix, cont

The company offers discounts for bulk purchases giving a sales dependent profit per unit as shown below:

A Sales (Units)	A Unit Profit ($)	B Sales (Units)	B Unit Profit ($)	C Sales (Units)	C Unit Profit ($)
0-40	10	0-50	6	0-100	5
40-100	9	50-100	4	Over 100	4
100-150	8	Over 100	3		
Over 150	7				

Resource constraints for the scheduling is 100 hours of engineering, 700 hours of labour and 400 kg of materials available

Example - Product mix, cont

- For product A we label the first 39 units A_1, then units from 40 up to 100 with label A_2, units from 100 up to 150 with label A_3 and units from 150 and above with label A_4
- Similarly for product B, the first units up to 50 are labelled B_1 and so on
- The decision variables are $x_i, i = 1:4$ number of units with label A_i, $x_i, i = 5:7$ number of units with label B_{i-4} and $x_i, i = 8,9$ number of units with label C_{i-7}

Objective function

Maximize: $Z = 10x_1 + 9x_2 + 8x_3 + 7x_4 + 6x_5 + 4x_6 + 3x_7 + 5x_8 + 4x_9$

Subject to:
$$
\begin{aligned}
x_1 &\leq 39 \\
x_2 &\leq 60 \\
x_3 &\leq 50 \\
x_5 &\leq 49 \\
x_6 &\leq 50 \\
x_8 &\leq 99 \\
x_1 + x_2 + x_3 + x_4 + 2x_5 + 2x_6 + 2x_7 + x_8 + x_9 &\leq 100 \\
10x_1 + 10x_2 + 10x_3 + 10x_4 + 4x_5 + 4x_6 + 4x_7 + 5x_8 + 5x_9 &\leq 700 \\
3x_1 + 3x_2 + 3x_3 + 3x_4 + 2x_5 + 2x_6 + 2x_7 + x_8 + x_9 &\leq 400
\end{aligned}
$$

Excel

Mixed integer programming

- Many LP problem formulations put additional logical constraints on the decision variables
- Typically some of the decision variables are restricted to integer values or even binary variables
- I.e., we have a *mixed integer programming problem*
- A naive approach to mixed integer programming is to ignore the integer constraint and solve the problem as if it was an ordinary LP problem and then take the nearest integer from the continuous solution for those variables
- The *branch and bound* (B&B) algorithm is maybe the most widely method used for solving mixed integer problems in commercial computer codes
- B&B cannot guarantee that we find an optimal solution in reasonable computational time, but it performs rather good
- The B&B is exemplified in the course compendium

Production scheduling with fixed costs

- We will now revisit the scheduling problem but assume that there is a fixed cost in addition to the variable cost if there is a production in period i
- Numbers are slightly changed from the example and shown on the next slide
- The introduction of the binary variable is a tricks to take fixed costs into account
- We will let Excel help us with the optimization
- But we need to formulate the problem first

Production scheduling with fixed costs

Quantity	Explanation
x_j	Normal production in period j, i.e., number of items produced
$C_{\mathrm{N},j}$	Normal production capacity in period j, = 700 $\forall j$
$c_{\mathrm{I},j}$	Cost per item produced without overtime in period j, = {5,5,10,10}
$c_{\mathrm{F},j}$	Fixed cost in period j if $x_i > 0$, = 3 000 $\forall j$
y_j	indicator variable for production in period j, $y_j = 1$ if $x_j > 0$
d_j	Demand in period j, = {500,300,900,800}
v_j	Items produced by means of overtime in period j
$C_{\mathrm{O},j}$	Overtime production capacity in period j, = 200 $\forall j$
$c_{\mathrm{O},j}$	Cost per item using overtime in period j, = {10,10,15,15}
$c_{\mathrm{H},j}$	Holding cost per item at end of period j, = 3$\forall j$
u_j	Stock level at end of period j

Objective function

$$\text{Minimize: } Z = \sum_j c_{\mathrm{I},j} x_j + \sum_j c_{\mathrm{F},j} y_j + \sum_j c_{\mathrm{O},j} v_j + \sum_j c_{\mathrm{H},j} u_j$$

$$\begin{aligned} \text{Subject to: } x_j &\le C_{\mathrm{N},j} y_j \\ v_j &\le C_{\mathrm{O},j} y_j \\ x_j + v_j + u_{j-1} - u_j &= d_j (u_0 = 0) \\ y_j &= \text{ binary} \end{aligned}$$

The constraints $x_j \le C_{\mathrm{N},j} y_j$ ($\Leftrightarrow x_j - C_{\mathrm{N},j} y_j \le 0$) ensure that there is no production in period j if $y_j = 0$

▸ Excel ▸ Python

Python considerations

- The standard Python LP-solver `linprog` from the `scipy.optimize` package will not handle mixed integer problems
- It is possible to use the `milp` function from the `scipy.optimize` package and there are also other packages handling mixed integer linear programming
- In the lpLib.py provided by TPK4191 there is a function `all_binary_combinations(n)` which generates all vectors combining *n* binary variables
- The `all_binary_combinations(n)` can then be used to handle MILP problems by fixing one combination of the binary variables, then solve the LP-problem for the remaining continuous variables, and the proceed to the next combination to search for better solutions
- The indicated approach will work for small problems, but for larger problems it is required to search for faster algorithms in Python

Thank you for your attention

NTNU | Norwegian University of Science and Technology