# Data-driven Decision Making

## Introduction: linear optimization

João Pedro Pedroso

2024/2025

# Mathematical Optimization

Last class

- installation of AMPL
- very short introduction to linear optimization

# Using AMPL in the internet

- `https://neos-server.org/neos/`
  - includes other languages and many solvers
- *Colab notebook*
  - install dependencies:

```
%pip install -q amplpy matplotlib pandas
```

  - install Python interface:

```
from amplpy import AMPL, ampl_notebook

ampl = ampl_notebook(
    modules=["cbc", "highs"],  # modules to install
    license_uuid="default",  # license to use
)  # instantiate
```

  - for AMPL models, start each cell as:

```
%%ampl_eval
# ampl model comes here
```

- for AMPL models, start each cell as:

```
%%ampl_eval
var x1 >=0;  # define decision variables
var x2 >=0;

maximize  z : 300*x1 + 200*x2 ;  # objective

R1:     x1/40 + x2/60 <= 1;  # constraints
R2:     x1/50 + x2/50 <= 1;
```

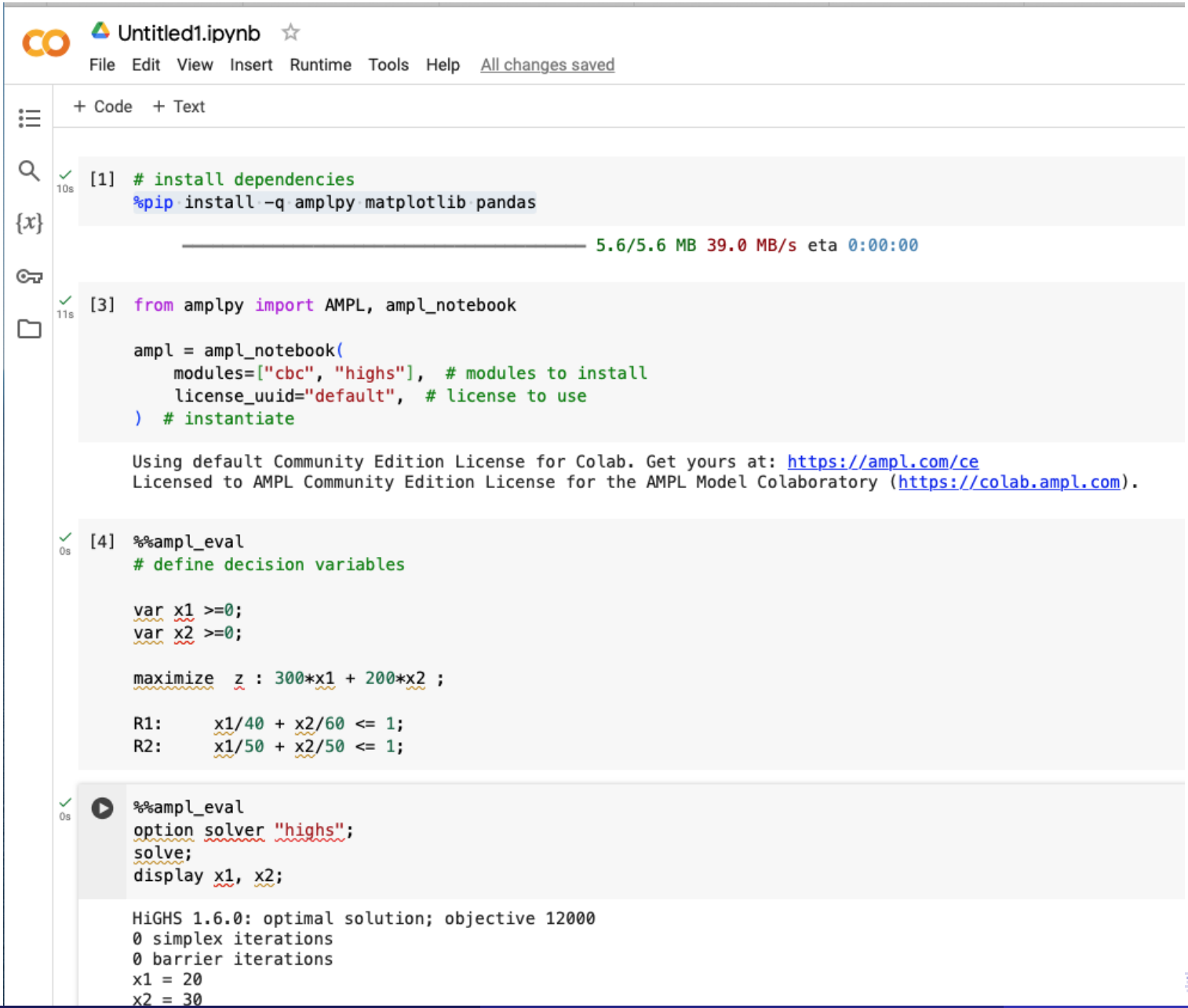
- now we can solve the problem:

```
%%ampl_eval
option solver "highs";
solve;
display x1, x2;
```

- output:

```
HiGHS 1.6.0: optimal solution; objective 12000
0 simplex iterations
0 barrier iterations
x1 = 20
x2 = 30
```

# https://colab.research.google.com



# See also:

- AMPL's colab page:
  `https://colab.ampl.com/`

# Mathematical Optimization

Mathematical optimization: science of finding the "best" solution

- Given an objective function, find a solution which is at least as good as any other possible solution
- Course of action:
  - describe the problem in terms of mathematical expressions
  - use some methodology to obtain an optimal solution from these formulas

# Problem description

- No ambiguity allowed
- Use mathematical expressions
  - objective function (which we want to maximize of minimize);
  - conditions of the problem: constraint 1, constraint 2, ...

# Linear optimization

- How can we obtain a solution to the model?
  $\rightarrow$ *linear programming* (*"optimization in linear structure"*)
  - proposed George Dantzig in 1947
  - based on the work of three Nobel laureate economists:
    - Wassily Leontief
    - Leonid Kantrovich
    - Tjalling Koopmans
- Simplex method
  - developed by Dantzig, has long been the most commong algorithm for linear optimization problems
  - considered one of the 10 most important algorithms of last century
  - in some cases, method requires a very long time $\rightarrow$ *time not bounded by polynomial, in terms of the size of the problem*

# Simplex method

- Idea: like solving a system of simultaneous equations
- Usually: much more variables than equations
- Approach:
  - fix some variables at zero
  - solve the system for the remaining
  - check if current solution is optimum
  - if not, move to an *adjacent* solution
    - swap one zero-variable with a non-zero variable

# Algorithms' efficiency (in the theoretical sense)

- Question: is there an algorithm which solves linear optimization problems in polynomial time?
  - ellipsoid method (Leonid Khachiyan, 1979)
    - only theoretical, in practice simplex method was better
  - interior point (Karmarkar, 1984)
    - theoretically efficient
    - currently, performance similar or higher than the simplex method's
- Currently optimization solvers:
  - usually equipped with simplex method (and its dual version, the dual simplex method) and with interior point methods,
  - users can choose the most appropriate of them

# Linear optimization: simple optimization problem

Example for introducing terminology:

$$
\begin{array}{lrrrr}
\text{maximize} & 15x_1 + & 18x_2 + & 30x_3 & \\
\text{subject to:} & 2x_1 + & x_2 + & x_3 \leq & 60 \\
& x_1 + & 2x_2 + & x_3 \leq & 60 \\
& & & x_3 \leq & 30 \\
& x_1, & x_2, & x_3 \geq & 0
\end{array}
$$

- $x_1, x_2, x_3$ : values that we do not know $\rightarrow$ variables
  - assumed to change continuously (*real/continuous variables*)
- first expression: function to be maximized $\rightarrow$ objective function
- constraints: second and subsequent expressions
  - restrict the value of the variables
  - sign restrictions or non-negativity constraints $\rightarrow$ ensure variables are non-negative

# Linear optimization: simple optimization problem

Example for introducing terminology:

$$
\begin{array}{lrrrr}
\text{maximize} & 15x_1 + & 18x_2 + & 30x_3 & \\
\text{subject to:} & 2x_1 + & x_2 + & x_3 \leq & 60 \\
& x_1 + & 2x_2 + & x_3 \leq & 60 \\
& & & x_3 \leq & 30 \\
& x_1, & x_2, & x_3 \geq & 0
\end{array}
$$

- Objective function and constraints: adding and subtracting $x_1, x_2, x_3$ multiplied by a constant $\rightarrow$ linear expressions
- Maximizing (or minimizing) a linear objective function subject to linear constraints $\rightarrow$ linear optimization problem
- Set of values for variables $x_1, x_2, x_3 \rightarrow$ solution
  - feasible solution $\rightarrow$ if it satisfies all constraints
  - optimal solutions $\rightarrow$ among feasible solutions, those that maximize (or minimize) the objective function
  - optimum $\rightarrow$ maximum/minimum value of objective function
    - there may be multiple solutions with optimum objective value
    - usually, the aim is to find just one of them

# Finding optima

- Finding an optimum solution: explore the search space in some methodical way
  - task of a linear optimization solver
  - given the problem description, any solver reaches the same optimum value
    - though, possibly, a different optimum solution
- Problem description: an appropriate language is AMPL
  - we will separate the *description* of the problem from the rest of the program
  - AMPL: *domain-specific language* for optimization
  - subset for linear optimization: implemented in GNU MathProg

# Problem description

Back to our example:

$$
\begin{array}{lrrrr}
\text{maximize} & 15x_1 + & 18x_2 + & 30x_3 & \\
\text{subject to:} & 2x_1 + & x_2 + & x_3 \leq & 60 \\
& x_1 + & 2x_2 + & x_3 \leq & 60 \\
& & & x_3 \leq & 30 \\
& x_1, & x_2, & x_3 \geq & 0
\end{array}
$$

```
var x1 >=0;
var x2 >=0;
var x3 >=0;

maximize z: 15*x1 + 18*x2 + 30*x3;

subject to
C1: 2*x1 + x2 + x3 <= 60;
C2: x1 + 2*x2 + x3 <= 60;
C3: x3 <= 30;
```

# Steps for writing a model

1. Declare variables; *e.g.*, `var x1 >=0;`
2. Define objective:
   - `maximize` or `minimize` keywords
   - objective name followed by `:` *e.g.*, `z:` *expression*
3. Define constraints:
   - `subject to` (optional)
   - constraint name followed by `:`
   - constraint expression; types: `>=` or `<=` or `=`
4. All statements should end with semicolon `;`

```
var x1 >=0;
var x2 >=0;
var x3 >=0;
maximize z: 15*x1 + 18*x2 + 30*x3;
subject to
C1: 2*x1 + x2 + x3 <= 60;
C2: x1 + 2*x2 + x3 <= 60;
C3: x3 <= 30;
```

# Output

(May be different, depending on the solver used)

```
MINOS 5.51: optimal solution found.
3 iterations, objective 1230

"option abs_boundtol 3.552713678800501e-15;"
or "option rel_boundtol 1.1842378929335003e-16;"
will change deduced dual values.

: _varname _var    :=
1   x1       10
2   x2       10
3   x3       30
;
```

# Transportation Problem

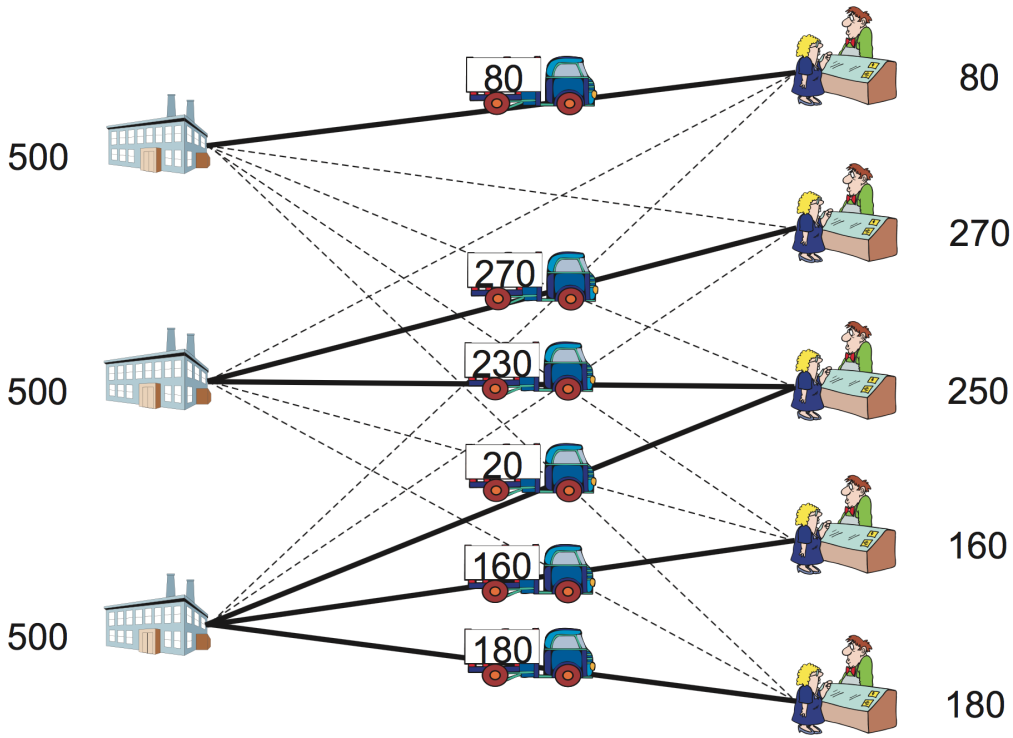
# Transportation Problem

Example:

*You are the owner of a sports equipment sales chain. Your products are manufactured at three factories, and you have to deliver them to five customers (demand points). After elaborating a survey, you found that the production capacity at each factory, the transportation cost to customers, and the demand amount at each customer are as shown in the next table. So, which of the transport routes would you choose to minimize the total cost?*

| transportation cost $c_{ij}$ | | customers $i$ | | | | | |
|---|---|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 | 5 | capacity $M_j$ |
| plant $j$ | 1 | 4 | 5 | 6 | 8 | 10 | 500 |
| | 2 | 6 | 4 | 3 | 5 | 8 | 500 |
| | 3 | 9 | 7 | 4 | 3 | 4 | 500 |
| demand $d_i$ | | 80 | 270 | 250 | 160 | 180 | |

# Transportation Problem




| transportation cost $c_{ij}$ | | customers $i$ 1 | 2 | 3 | 4 | 5 | capacity $M_j$ |
|---|---|---|---|---|---|---|---|
| plant $j$ | 1 | 4 | 5 | 6 | 8 | 10 | 500 |
| | 2 | 6 | 4 | 3 | 5 | 8 | 500 |
| | 3 | 9 | 7 | 4 | 3 | 4 | 500 |
| demand $d_i$ | | 80 | 270 | 250 | 160 | 180 | |

# Formulation: notation

- Number of customers: $n$
  $\rightarrow$ each customer represented by $i = 1, 2, \ldots, n$
  $\rightarrow$ set of customers: $I = \{1, 2, \ldots, n\}$
- Number of factories $m$
  $\rightarrow$ each factory represented by $j = 1, 2, \ldots, m$
  $\rightarrow$ set of factories: $J = \{1, 2, \ldots, m\}$
- Demand amount of customer $i \rightarrow d_i$
- Transportation cost for shipping one unit of demand from plant $j$ to customer $i \rightarrow c_{ij}$
- Production in plant $j$ limited by $M_j$

# Formulation: model

- variables: $x_{ij}$ = amount of goods to be transported from factory $j$ to customer $i$
- model:

$$\begin{aligned}
\text{minimize} \quad & \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij} \\
\text{subject to} \quad & \sum_{j \in J} x_{ij} = d_i \quad \forall i \in I \\
& \sum_{i \in I} x_{ij} \leq M_j \quad \forall j \in J \\
& x_{ij} \geq 0 \quad \forall i \in I, j \in J
\end{aligned}$$

# Solving with AMPL: a structured approach

We will need to define:

- sets (*e.g.*, customers)
- parameters (*e.g.*, demand at each customer)
- variables, whose values the solver is to determine
- objective, to be maximized or minimized
- constraints that the solution must satisfy

# AMPL model

File `transp.mod`

```
set I;
set J;
param c {I, J};
param d {I};
param M {J};

var x {I, J} >=0;

subject to
Demand {i in I}: sum {j in J} x[i,j] = d[i];
Supply {j in J}: sum {i in I} x[i,j] <= M[j];

minimize cost: sum {i in I, j in J} c[i,j] * x[i,j];
```

# AMPL data

File `transp.dat`

```
data;
param: J:  M :=  # defines set "J" and param "M"
       1 500
       2 500
       3 500;
param: I:  d :=  # defines set "I" and param "d"
       1  80
       2 270
       3 250
       4 160
       5 180;
param c (tr) :   # (tr) --> transposed
             1  2  3  4  5 :=
      1      4  5  6  8  10
      2      6  4  3  5  8
      3      9  7  4  3  4 ;
```

# Using AMPL in interactive mode

```
ampl: reset;                      # use when AMPL had a previous model
ampl: model transp.mod            # load model
ampl: data transp.dat             # load data
ampl: option solver "gurobi"; # use solve called "gurobi"
ampl: solve;                      # solve the problem
Gurobi 9.1.1: optimal solution; objective 3370
1 simplex iterations
ampl: option display_1col 0;  # to display variable in tabular form
ampl: display x;
x [*,*]
:   1     2     3     :=
1   80     0     0
2   20   250     0
3    0   250     0
4    0     0   160
5    0     0   180
;
```

# Solving with an AMPL script

Typically, AMPL scripts:

- load AMPL models and data
- choose the solver to use
- solve the models
- display the solution

```
model "transp.mod";
data "transp.dat";
option solver "gurobi";
solve;
display cost;
option display_1col 0; # show as table
display x;
```

output:

```
$ ampl transp.run
Gurobi 9.1.1: optimal solution; objective
1 simplex iterations
cost = 3370

x [*,*]
:   1     2     3     :=
1   80     0     0
2   20   250     0
3    0   250     0
4    0     0   160
5    0     0   180
;

$
```

# Solving from Python

```
from amplpy import AMPL, Environment
ampl = AMPL()  # or AMPL(Environment('full path to the AMPL installation directory'
ampl.option['solver'] = 'gurobi'

# read model, data, and solve it
ampl.read("transp.mod")
ampl.readData("transp.dat")
ampl.solve()

# get values at the optimum
cost = ampl.obj['cost']
x = ampl.var['x']
print("Optimum:", cost.value())
print("x[1,2]:", x[1, 2].value())

# using data frames
df = x.getValues()
print(df)
```

# Output

```
$ python transp.py
Gurobi 9.1.1: optimal solution; objective 3370
1 simplex iterations
Optimum: 3370.0
x[1,2]: 0.0
   index0       index1       |     x.val
     1              1         |       80
     1              2         |       0
     1              3         |       0
     2              1         |       20
     2              2         |      250
     2              3         |       0
     3              1         |       0
     3              2         |      250
     3              3         |       0
     4              1         |       0
     4              2         |       0
     4              3         |      160
     5              1         |       0
     5              2         |       0
     5              3         |      180
```

# Duality

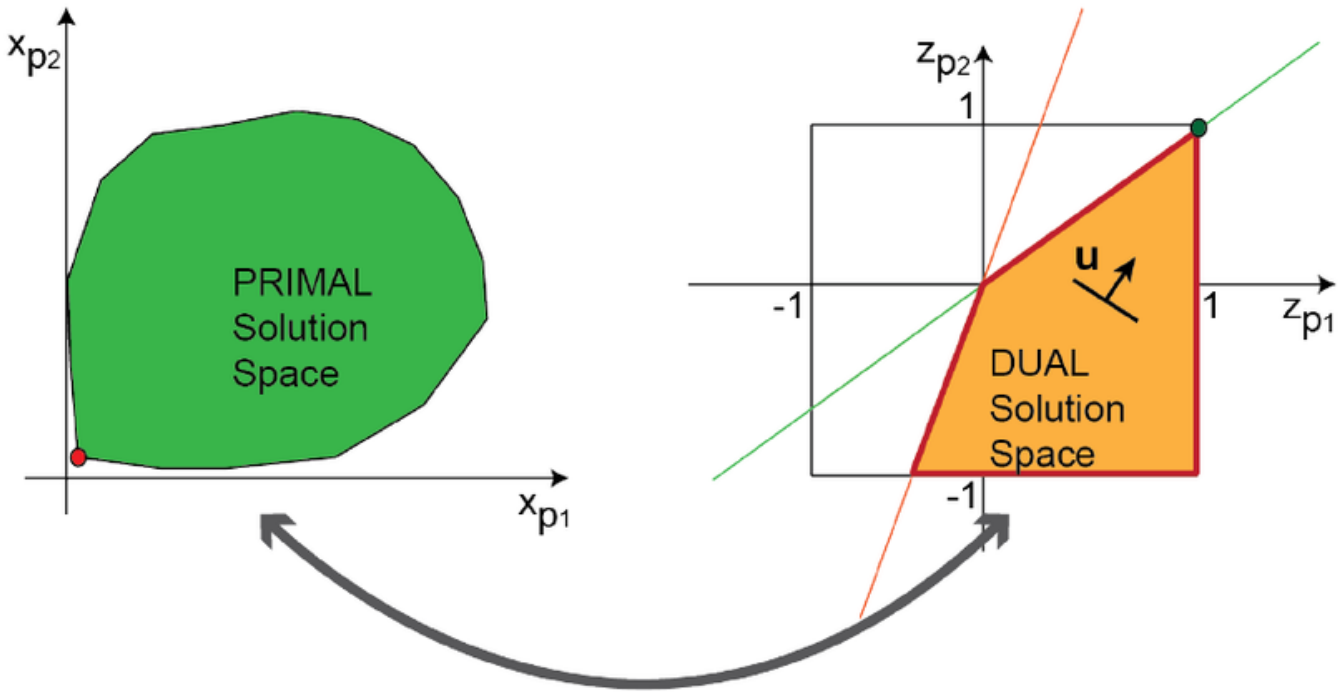
# Duality

Consider the following scenario:

> *You are the owner of the sports equipment sales chain that appeared on the previous example. You feel that factory's capacity has become tight, so you are considering an expansion. What kind of expenses can be expected to be reduced by expanding each of the factories? Also, what is the additional gain that you can you get if you have additional orders from each customer?*

# Duality

- In order to solve this problem smartly, the concept of dual problem is useful
- Here:
  - original problem: primal
  - associated: dual linear optimization problem (details later)



# Duality: investigate whether a factory should be expanded

Focus on the capacity constraint:

$$\sum_{i \in I} x_{ij} \leq M_j \qquad \forall j \in J$$

```
Supply {j in J}: sum {i in I} x[i,j] <= M[j];
```

- slack variable: difference $M_j - \sum_{i \in I} x_{ij}$
  - in AMPL: `.slack` attribute for a constraint
    → *e.g.*, `display Supply.slack;`
- dual variable
  - in AMPL: `.dual` attribute for a constraint
    → *e.g.*, `display Supply.dual;`

# Duality: analysis

```
Supply {j in J}: sum {i in I} x[i,j] <= M[j];
Demand {i in I}: sum {j in J} x[i,j] = d[i];
```

- Optimal value of dual variable
  - `Supply` $\rightarrow$ reduction on costs when increasing the capacity constraint by one unit
  - `Demand` $\rightarrow$ increase in costs as demand increases by one unit

# Duality: analysis

```
Supply {j in J}: sum {i in I} x[i,j] <= M[j];
Demand {i in I}: sum {j in J} x[i,j] = d[i];
```

```
: Supply.slack Supply.dual     :=
1       400             0
2         0            -1
3       160             0
;
```

```
: Demand.slack Demand.dual     :=
1         0             4
2         0             5
3         0             4
4         0             3
5         0             4
;
```

# Duality in linear optimization

- Optimal value of the dual variable associated to a constraint: shadow price
  - impact in the optimum when the right hand side of a constraint is increased by one unit
  - AMPL: `.dual`
- Reduced cost associated to a decision variable:
  - amount by which an objective function coefficient would have to improve for the variable to become positive value in the optimal solution
  - AMPL: `.rc`

# Duality theory

Using the introductory example:

$$
\begin{array}{lrrrr}
\text{maximize} & 15x_1 + & 18x_2 + & 30x_3 & \\
\text{subject to:} & 2x_1 + & x_2 + & x_3 \leq & 60 \\
& x_1 + & 2x_2 + & x_3 \leq & 60 \\
& & & x_3 \leq & 30 \\
& x_1, & x_2, & x_3 \geq & 0
\end{array}
$$

- Original problem: primal problem
  - as oposed to dual problem
- Primal problem: maximization $\rightarrow$ Dual problem: minimization
- First constraint can be written as: $60 - 2x_1 - x_2 - x3 \geq 0$
- We may multiply (60 - 2 $x_1$ - $x_2$ - $x_3$) by $\pi_1 \geq 0$ and add it to the objective; its value will not decrease.
  - the same for constraints 2 and 3, leading to the following problem

- The following problem's optimum is an upper bound to the primal's optimum

$$\begin{aligned}\text{maximize } 15x_1 + 18x_2 + 30x_3 &+(60 - 2x_1 - x_2 - x_3)\pi_1 \\ &+(60 - x_1 - 2x_2 - x_3)\pi_2 \\ &+(30 - x3)\pi_3\end{aligned}$$

subject to: ...

- Reorganizing:

$$\begin{aligned}\text{maximize } &(15 - 2\pi_1 - \pi_2)x_1 + (18 - \pi_1 - 2\pi_2)x_2 + \\ &(30 - \pi_1 - \pi_2 - \pi_3)x_3 + 60\pi_1 + 60\pi_2 + 30\pi_3\end{aligned}$$

subject to: ...

- Let us consider that objective, removing all constraints except non-negativity
  - excluding constraints guarantees that the optimum value of this problem is not smaller than that of the original problem
  - hence, solving this problem gives an upper bound, for arbitrary $\pi_1, \pi_2, \pi_3 \geq 0$

# Reduced costs

$$\begin{aligned} \text{maximize } & (15 - 2\pi_1 - \pi_2)x_1 + (18 - \pi_1 - 2\pi_2)x_2 + \\ & (30 - \pi_1 - \pi_2 - \pi_3)x_3 + 60\pi_1 + 60\pi_2 + 30\pi_3 \\ \text{subject to: } & x_1, x_2, x_3 \geq 0 \end{aligned}$$

- The coefficient of $x_1, x_2, x_3$ are called reduced costs
  - represents *increase in the optimum cost* when a variable $x$ which is 0 is increased by 1
- Let's consider $\pi$ as variable (rather than variable $x$)
- We want to obtain the best upper bound of our optimum

# Dual constraints

$$\begin{aligned} \text{maximize } & (15 - 2\pi_1 - \pi_2)x_1 + (18 - \pi_1 - 2\pi_2)x_2 + \\ & (30 - \pi_1 - \pi_2 - \pi_3)x_3 + 60\pi_1 + 60\pi_2 + 30\pi_3 \\ \text{subject to: } & x_1, x_2, x_3 \geq 0 \end{aligned}$$

- If the coefficient $(15 - 2\pi_1 - \pi_2)$ of $x_1$ is positive, the objective becomes $\infty$
  - hence, for a meaningful upper bound, $15 - 2\pi_1 - \pi_2 \leq 0$
  - similarly, $18 - \pi_1 - 2\pi_2 \leq 0$ and $30 - \pi_1 - \pi_2 - \pi_3 \leq 0$
- In terms of variables $\pi_1, \pi_2, \pi_3$, this is a linear problem

# Dual problem

$$
\begin{array}{lll}
\text{minimize} & 60\pi_1 + 60\pi_2 + 30\pi_3 & \\
\text{subject to:} & 2\pi_1 + \pi_2 & \geq 15 \\
& \pi_1 + 2\pi_2 & \geq 18 \\
& \pi_1 + \pi_2 + \pi_3 & \geq 30 \\
& \pi_1, \pi_2, \pi_3 \geq 0 &
\end{array}
$$

- optimum value of the variables: dual prices or shadow prices

# Duality theory

- Weak duality:
  - the objective value for a feasible solution to the primal (maximization) problem is always less than or equal to the objective value of any feasible solution of the associated dual problem
  - if primal is minimization $\rightarrow$ objective is greater than or equal to the dual objective of any (dual) feasible solution
- Strong duality theorem: if either the primal or dual has an optimal solution, then the other also has an optimal solution, and the optimum is the same
  - If one of the problems is unbounded, the other is infeasible
  - But both problems may be infeasible
- Complementary slackness: at the optimum, for every constraint:
  - if the slack is positive, then the shadow price is zero
  - if the shadow price is positive, then the slack is zero

# Revenue management

# Revenue management

- Important applications in reservations inventory, while selling *flight tickets*
  - determine fare classes for each flight in the flight schedule
  - allocation of seats to each fare class
- Setting:
  - high fixed costs for operating a flight
  - marginal cost of carrying additional passenger very low
  - increasing load factor $\rightarrow$ *significant impact* in revenues
  - goods are perishable
- Difficulty in maximizing revenue in a flight segment:
  - selling less expensive seats $\rightarrow$ *additional revenue*
  - turning away higher fare customers $\rightarrow$ *losing revenue*

# Revenue management

- Key question: is how many seats to sell on discount
- Key consideration: passengers have different valuations
  - *e.g.*, business people value flexibility, people on a vacation may value good deals
  - if we sell too many discounted seats $\rightarrow$ not enough seats for high-paying passengers
  - if we sell too few discounted seats $\rightarrow$ empty seats $\Rightarrow$ lost revenue
  - how allocate seats in order to maximize revenue?

# Example: revenue maximization

Consider the management of trains from Braga to Lisbon (through Porto).

- Users have alternatives: *car*, *bus*
- In an attempt to sell more seats, the train company (CP) may offer discounts
- One possible strategy:
  - selling enough seats to cover fixed operating costs
  - selling remaining seats at higher rates to maximize revenues

# How to allocate seats

- Suppose a train has 200 economy seats
- In each trip Lisbon-Braga, there are two types of economy fares: *discount* and *regular*
  - regular price: 64 euro
  - discounted price: 22 euro
- Demand forecasted for these prices (*e.g.*, using historical data):
  - regular price: 150 tickets
  - discounted price: 200 tickets
  - (forecasts have errors $\rightarrow$ assess how robust a solution is with /sensitivity analysis)
- In this case, can we infer the solution? How does it vary with demand?
- What happens if we have many trains, each doing multiple trips?

# A more complete example

- Problem: one train, making Lisbon-Braga trip
- Two legs: Lisbon-Porto and Porto-Braga
  - travelers may go Lisbon-Braga, Lisbon-Porto or Porto-Braga
  - different prices and demand
  - train capacity unchanged (200 seats)

# Revenue management: problem formulation

- Formulating the problem mathematically allows us to solve it in a systematic way, using linear optimization
- Two legs: Braga-Porto and Porto-Lisbon

| Trip | Class | Price | Dem |
|---|---|---|---|
| L-B | regular | 64 | 120 |
| | discount | 22 | 220 |
| L-P | regular | 51 | 160 |
| | discount | 19 | 240 |
| P-B | regular | 43 | 150 |
| | discount | 19 | 100 |

# Revenue management: problem formulation

- Formulating the problem mathematically allows us to solve it in a systematic way, using linear optimization
- Two legs: Braga-Porto and Porto-Lisbon

| Trip | Class | Price | Dem |
|---|---|---|---|
| L-B | regular | 64 | 120 |
| | discount | 22 | 220 |
| L-P | regular | 51 | 160 |
| | discount | 19 | 240 |
| P-B | regular | 43 | 150 |
| | discount | 19 | 100 |

- Decisions:
  - Regular seats: $R_{LB}, R_{LP}, R_{PB}$
  - Discount seats: $D_{LB}, D_{LP}, D_{PB}$
- Objective: maximize revenue

$$64R_{LB} + 51R_{LP} + 43R_{PB} + 22D_{LB} + 19D_{LP} + 19D_{PB}$$

- Constraints:
  - 200 passengers on the plane

$$R_{LB} + R_{LP} + D_{LB} + D_{LP} \leq 200$$
$$R_{LB} + R_{PB} + D_{LB} + D_{PB} \leq 200$$

  - non-negativity, demand

$$0 \leq R_{LB} \leq 120 \quad 0 \leq R_{LP} \leq 160 \quad 0 \leq R_{PB} \leq 1$$
$$0 \leq D_{LB} \leq 220 \quad 0 \leq D_{LP} \leq 240 \quad 0 \leq D_{PB} \leq 1$$

# Revenue management: problem formulation

$$\begin{aligned}
\text{maximize } & 64R_{LB} + 51R_{LP} + 43R_{PB} + 22D_{LB} + 19D_{LP} + 19D_{PB} \\
\text{subject to } & R_{LB} + R_{LP} + D_{LB} + D_{LP} \leq 200 \\
& R_{LB} + R_{PB} + D_{LB} + D_{PB} \leq 200 \\
& 0 \leq R_{LB} \leq 120 \quad 0 \leq R_{LP} \leq 160 \quad 0 \leq R_{PB} \leq 150 \\
& 0 \leq D_{LB} \leq 220 \quad 0 \leq D_{LP} \leq 240 \quad 0 \leq D_{PB} \leq 100
\end{aligned}$$

```
var rLB >=0; var rLP >=0; var rPB >=0;
var dLB >=0; var dLP >=0; var dPB >=0;

maximize z: 64 * rLB + 51 * rLP + 43 * rPB + 22 * dLB + 19 * dLP + 19 * dPB;

subject to
Leg1: rLB + rLP + dLB + dLP <=200;
Leg2: rLB + rPB + dLB + dPB <=200;

Dem_rLB: rLB <= 120; Dem_rLP: rLP <= 160; Dem_rPB: rPB <= 150;
Dem_dLB: dLB <= 220; Dem_dLP: dLP <= 240; Dem_dPB: dPB <= 100;
```

# Back to revenue management

$$
\begin{aligned}
\text{maximize } & 64R_{LB} + 51R_{LP} + 43R_{PB} + 22D_{LB} + 19D_{LP} + 19D_{PB} \\
\text{subject to } & R_{LB} + R_{LP} + D_{LB} + D_{LP} \leq 200 \\
& R_{LB} + R_{PB} + D_{LB} + D_{PB} \leq 200 \\
& 0 \leq R_{LB} \leq 120 \quad 0 \leq R_{LP} \leq 160 \quad 0 \leq R_{PB} \leq 150 \\
& 0 \leq D_{LB} \leq 220 \quad 0 \leq D_{LP} \leq 240 \quad 0 \leq D_{PB} \leq 100
\end{aligned}
$$

# AMPL model

```
var rLB >=0; var rLP >=0; var rPB >=0;
var dLB >=0; var dLP >=0; var dPB >=0;

maximize z: 64 * rLB + 51 * rLP + 43 * rPB + 22 * dLB + 19 * dLP + 19 * dPB;

subject to
Leg1: rLB + rLP + dLB + dLP <=200;
Leg2: rLB + rPB + dLB + dPB <=200;

Dem_rLB: rLB <= 120; Dem_rLP: rLP <= 160; Dem_rPB: rPB <= 150;
Dem_dLB: dLB <= 220; Dem_dLP: dLP <= 240; Dem_dPB: dPB <= 100;
```

# Solution:

```
z = 17360

rLB = 40
rLP = 160
rPB = 150
dLB = 0
dLP = 0
dPB = 10

Leg1 = 45
Leg2 = 19

Dem_rLB = 0                    rLB.rc = 0
Dem_rLP = 6                    rLP.rc = 0
Dem_rPB = 24                   rPB.rc = 0
Dem_dLB = 0                    dLB.rc = -42
Dem_dLP = 0                    dLP.rc = -26
Dem_dPB = 0                    dPB.rc = 0
```

# Back to revenue management

- CP is trying to figure out whether or not it would be beneficial to invest in marketing its fares.
- The marketing department forecasted that increasing the amount spent on each category by 10 € is likely to attract one more unit of demand, for each kind of ticket.
- Increasing demand → only relevant if we are already meeting the current demand

- Recall current solution

```
rLB = 40
rLP = 160
rPB = 150
dLB = 0
dLP = 0
dPB = 10

z = 17360
```

- we're not even meeting the current demand for discount fares → additional demand is irrelevant
- *e.g.*, demand could be zero for dPB without affecting the solution
  - *shadow price* for $D_{LB} \leq 220$ is 0
- so, CP could decrease budget to market *all discount fares* and rLB

# Back to revenue management

- Shadow prices: dual variables associated to constraints

```
Dem_rLB = 0
Dem_rLP = 6
Dem_rPB = 24
Dem_dLB = 0
Dem_dLP = 0
Dem_dPB = 0
```

- for `rLB`, `rPB`, increasing demand would improve objective
  - 6 € per unit demand of `rLB`
  - 24 € per unit demand of `rLB`
- so, additional investment in marketing would be worthy only for regular tickets, leg Porto-Braga

# More sensitivity analysis

- Would it be beneficial to use a bigger trains?
- Suppose CP could use a train with 250 places
  - current train's cost: 10000 euro per trip
  - larger train's cost: 15000 euro per trip
- Current revenue = 17360
- Which revenue for 250 places?
  - shadow price of constraints: `Leg1 = 45, Leg2 = 19`
  - assuming demand does not change
  - additional revenue $= 50 \times (45 + 19) = 3200 < 5000$
  - so, it would not be worthy to increase train's size

# Next class

Integer optimization