MVE165/MMG631
Linear and Integer Optimization with Applications
Lecture 2
AMPL and CPLEX; Assignment 1

Ann-Brith Strömberg

2018–03–21

AMPL

- Algebraic modelling language for optimization problems

  $\Longrightarrow$ Interface between problems and solvers
  $\Longrightarrow$ Formulate optimization models and examine solutions
  $\Longrightarrow$ Manage communication with an appropriate solver

- Natural syntax
- Separation of model and data
- Support for sets and set operators
- Built-in arithmetic functions
- Looping, if-then-else commands (implement "simple" algorithms)

CPLEX

- Optimization *software package* for solving linear and quadratic optimization problems with continuous and integer variables
- Originally based on the *simplex method*, implemented in C
- The *primal and dual* simplex methods (see Lectures 3–5)
- The *barrier method*
- Techniques for avoiding *degeneracy* (see Lecture 4)
- Generating *cutting planes* (see Lecture 9)
- The *branch&bound* algorithm (see Lecture 8)
- *Heuristic* methods (see Lecture 10)

AMPL licences

- AMPL/CPLEX along with licences are installed on Chalmers Linux system
  - For use of these licences, start by printing the command
    ```
    vcs-select -p ampl-MVE165-20180125
    ```
    in a terminal window
- There are also AMPL packages, including several optimization solvers, for installation on your private computers to download from the PingPong event
  - The licences are time limited and may only be used within the course
  - *The teachers cannot, however, provide any technical support regarding these licences*

Solvers that work with AMPL

- **CPLEX** – solves linear and quadratic optimization problems with continuous and integer variables
- Gurobi – solves linear and quadratic optimization problems with continuous and integer variables
- CONOPT – solves nonlinear optimization problems with continuous variables
- MINOS – solves linear and nonlinear optimization problems with continuous variables
- Baron, IlogCP, Knitro, Snopt, Xpress, etc
- See also within the AMPL packages on PingPong for download

The diet problem—description

- *The diet problem* (G.B. Dantzig, Interfaces 20(4):43–47, 1990) https://resources.mpi-inf.mpg.de/departments/d1/teaching/ws14/Ideen-der-Informatik/Dantzig-Diet.pdf
- Choose foods to meet certain nutritional requirements in the cheapest way
- A more sustainable version:
  - Kinds of food *[beans, egg, milk, potato, tomato]* are available in a limited amount per day and at a given price
  - 100g of each food provide given amounts of certain nutrients *[carbohydrates (CHO), protein, vitamin C, vitamin D]*
  - *Diet: requirements (upper and lower limits) on the daily amounts of each nutrient*

The Diet Problem—data

Food	price [SEK/hg]	available [hg/day]	CHO [g/hg]	protein [g/hg]	vit C [mg/hg]	vit D [μg/hg]
Beans	3.3	7	3.5	1.80	16.0	0
Egg	6.0	6	0.4	12.38	0	1.47
Milk	0.9	8	4.7	3.51	0.6	1.0
Potato	2.6	10	17.5	1.81	17.4	0
Tomato	5.8	5	2.6	0.81	14.8	0
Minimum amount/day			250 g	63 g	75 mg	10 μg
Maximum amount/day			300 g	125 g	1000 mg	1000 μg

* Data from `www.livsmedelsverket.se/livsmedelsdatabasen` and `www.coop.se/Handla-online/`

The Diet Problem—mathematical model

- Sets
  - $\mathcal{J} = \{1, \ldots, 5\}$ — kinds of food
  - $\mathcal{I} = \{1, \ldots, 4\}$ — nutrients
- Variables (something)

The Diet Problem—mathematical model

- Sets
  - $\mathcal{J} = \{1, \ldots, 5\}$ — kinds of food
  - $\mathcal{I} = \{1, \ldots, 4\}$ — nutrients
- Variables (something)
  - x_j, $j \in \mathcal{J}$ — purchased amount of food j per day [hg]
- Parameters

The Diet Problem—mathematical model

- Sets
  - $\mathcal{J} = \{1, \ldots, 5\}$ — kinds of food
  - $\mathcal{I} = \{1, \ldots, 4\}$ — nutrients
- Variables (something)
  - x_j, $j \in \mathcal{J}$ — purchased amount of food j per day [hg]
- Parameters
  - c_j, $j \in \mathcal{J}$ — cost of food j [SEK/hg]
  - a_j, $j \in \mathcal{J}$ — available amount of food j [hg]
  - p_{ij}, $i \in \mathcal{I}$, $j \in \mathcal{J}$ — content of nutrient i in food j [g/hg], [g/hg], [mg/hg], [μg/hg]
  - n_i — lower limit on the amount of nutrient i per day [g], [g], [mg], [μg]
  - N_i — upper limit on the amount of nutrient i per day [g], [g], [mg], [μg]

The Diet Problem—mathematical model

The Diet Problem—mathematical model

$$\text{minimize} \quad \sum_{j=1}^{5} c_j x_j, \qquad \text{(good)}$$

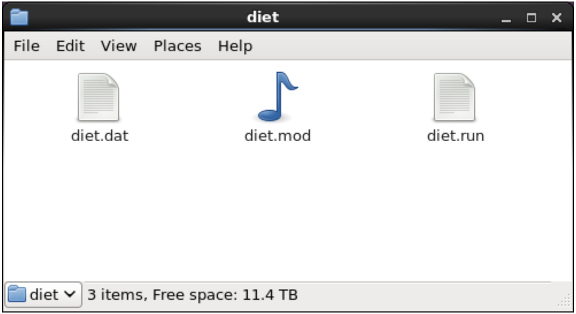
The Diet Problem—mathematical model

$$\text{minimize} \quad \sum_{j=1}^{5} c_j x_j, \qquad \text{(good)}$$

$$\begin{aligned} \text{subject to} \quad n_i \leq & \sum_{j=1}^{5} p_{ij} x_j \leq N_i, & i = 1, \ldots, 4, & \quad \text{(possible)} \\ 0 \leq & \quad x_j \leq a_j, & j = 1, \ldots, 5. & \quad \text{(possible)} \end{aligned}$$

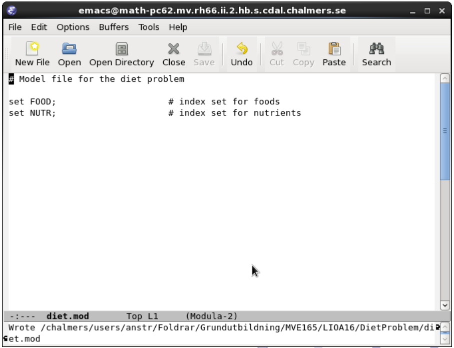
The Diet Problem—AMPL implementation

- Create a folder: *diet*
- Create a model file: *diet.mod*
- Create a data file: *diet.dat*
- Create a run file: *diet.run*



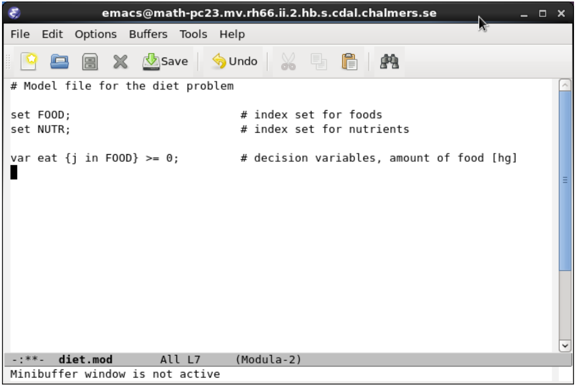
The Diet Problem—AMPL implementation

- Fill the model file using a text editor (Emacs, gedit, ...)
- Introduce index sets: *set*
- Comments start with #, each command ends with ;
- Sets
  - $\mathcal{I} = \{1, 2, 3, 4\}$
  - $\mathcal{J} = \{1, 2, 3, 4, 5\}$



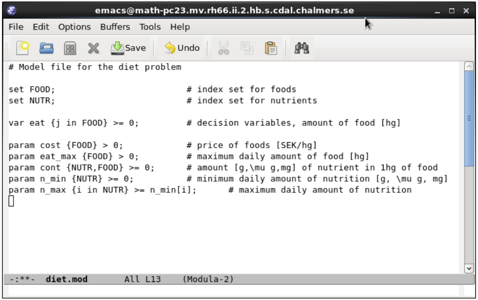
The Diet Problem—AMPL implementation

- Introduce variables: *var*
- Formulate non-negativity requirements
- Variables: x_j
  - $x_j \geq 0,\ j \in \{1, \ldots, 5\}$



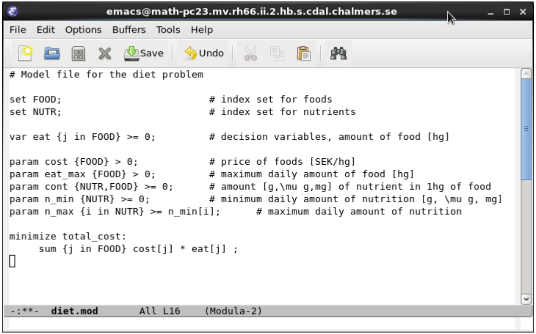
The Diet Problem—AMPL implementation

- Introduce parameters: *param*
- Parameters
    - c_j, $j \in \{1, \ldots, 5\}$
    - a_j, $j \in \{1, \ldots, 5\}$
    - p_{ij}, $i \in \{1, \ldots, 4\}$, $j \in \{1, \ldots, 5\}$
    - n_i, N_i, $i \in \{1, \ldots, 4\}$



The Diet Problem—AMPL implementation

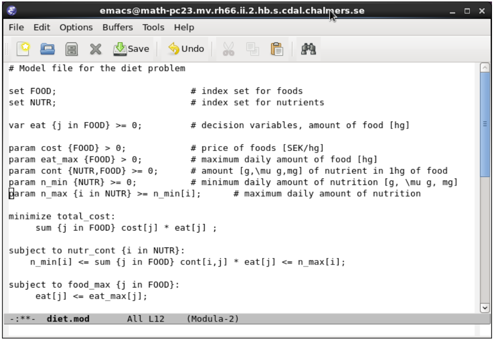
- Formulate an objective function: *minimize, maximize*
- Use built-in arithmetic operations:
  $+, -, *, \wedge, /,$ *sum, prod, abs, log, sin, . . .*
- min $\sum_{j=1}^{5} c_j x_j$




```
# Model file for the diet problem

set FOOD;                          # index set for foods
set NUTR;                          # index set for nutrients

var eat {j in FOOD} >= 0;          # decision variables, amount of food [hg]

param cost {FOOD} > 0;             # price of foods [SEK/hg]
param eat_max {FOOD} > 0;          # maximum daily amount of food [hg]
param cont {NUTR,FOOD} >= 0;       # amount [g,\mu g,mg] of nutrient in 1hg of food
param n_min {NUTR} >= 0;           # minimum daily amount of nutrition [g, \mu g, mg]
param n_max {i in NUTR} >= n_min[i];       # maximum daily amount of nutrition

minimize total_cost:
     sum {j in FOOD} cost[j] * eat[j] ;
```

The Diet Problem—AMPL implementation

- Formulate constraints: *subject to*
- Use arithmetic relations: $>$, $>=$, $<$, $<=$, $==$, $!=$, ...
- $n_i \leq \sum_{j=1}^{5} p_{ij} x_j \leq N_i, \; i = 1, \ldots, 4,$
- $x_i \leq a_i, \; i = 1, \ldots, 3$




```
# Model file for the diet problem

set FOOD;                            # index set for foods
set NUTR;                            # index set for nutrients

var eat {j in FOOD} >= 0;            # decision variables, amount of food [hg]

param cost {FOOD} > 0;               # price of foods [SEK/hg]
param eat_max {FOOD} > 0;            # maximum daily amount of food [hg]
param cont {NUTR,FOOD} >= 0;         # amount [g,\mu g,mg] of nutrient in 1hg of food
param n_min {NUTR} >= 0;             # minimum daily amount of nutrition [g, \mu g, mg]
param n_max {i in NUTR} >= n_min[i];      # maximum daily amount of nutrition

minimize total_cost:
     sum {j in FOOD} cost[j] * eat[j] ;

subject to nutr_cont {i in NUTR}:
    n_min[i] <= sum {j in FOOD} cont[i,j] * eat[j] <= n_max[i];

subject to food_max {j in FOOD}:
     eat[j] <= eat_max[j];
```

The Diet Problem—AMPL implementation

- Fill in the data file using the text editor
- Assign values to the introduced sets and parameters

emacs@math-pc23.mv.rh66.ii.2.hb.s.cdal.chalmers.se

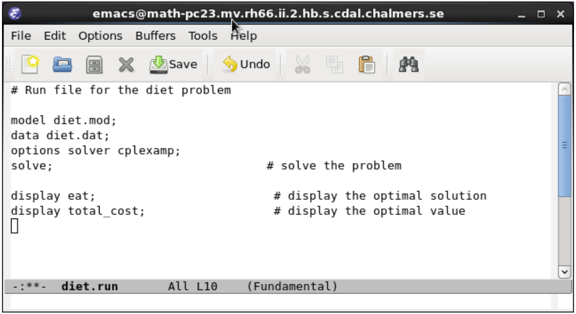
File Edit Options Buffers Tools Help

```
set NUTR := Protein Carbohydrate VitD VitC;
set FOOD := Egg Beans Tomato Milk Potato;

param:   cost eat_max:=    # SEK/hg   hg/day
  Beans   3.3    7
  Egg     6.0    6
  Milk    0.9    8
  Potato  2.6    10
  Tomato  5.8    5;

param:           n_min  n_max :=
  Carbohydrate  250    300        # g/day
  Protein        63    125        # g/day
  VitC           75   1000        # mg/day
  VitD           10   1000 ;      # \mu g/day

param cont (tr):                   # amount per 1hg food
           Carbohydrate Protein  VitC   VitD :=
  Beans          3.5       1.80   16.0    0
  Egg            0.4      12.38    0      1.47
  Milk           4.7       3.51    0.6    1.0
  Potato        17.5       1.81   17.4    0
  Tomato         2.6       0.81   14.8    0  ;

#               g         g        mg    \mu g
```

```
-:---  diet.dat       All L1     (Fundamental)
For information about GNU Emacs and the GNU system, type C-h C-a.
```

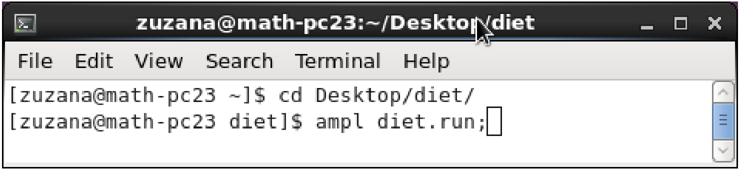
The Diet Problem—AMPL implementation

- Fill the run file using the text editor
- Load the model and the data: *model*, *data*
- Choose solver: *options solver*
- Solve the problem: *solve*
- Display results: *display*



The Diet Problem—AMPL implementation

- Open a Terminal window
- Go to the folder *diet*
- Evaluate the commands in the run file *diet.run* by AMPL

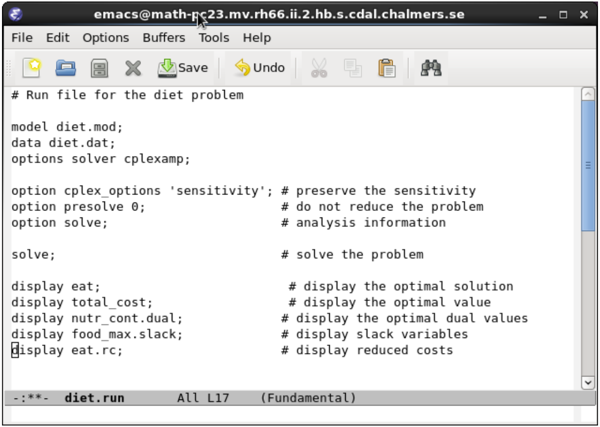


The Diet Problem—AMPL implementation

```
zuzana@math-pc23:~/Desktop/diet
File  Edit  View  Search  Terminal  Help
[zuzana@math-pc23 ~]$ cd Desktop/diet/
[zuzana@math-pc23 diet]$ ampl diet.run;
IBM ILOG License Manager: "university-goteborg" is access
ing AMPL 12.
IBM ILOG AMPL 12.1.0 (5724-Y45) AMPL Version 20090327 (Li
nux 2.6.18-6-amd64)
IBM ILOG License Manager: "university-goteborg" is access
ing CPLEX 12 with option(s): "b e m q use=10 ".
CPLEX 12.1.0: optimal solution; objective 92.02616431
2 dual simplex iterations (0 in phase I)
eat [*] :=
 Beans    7
   Egg    1.36054
  Milk    8
Potato   10
Tomato    4.75222
;

total_cost = 92.0262

[zuzana@math-pc23 diet]$
```

The Diet Problem—AMPL implementation

- Perform sensitivity analysis
- Preserve the sensitivity analysis information
- Use suffices for sensitivity analysis: *.rc*, *.slack*, *.dual*, ...




```
# Run file for the diet problem

model diet.mod;
data diet.dat;
options solver cplexamp;

option cplex_options 'sensitivity'; # preserve the sensitivity
option presolve 0;                  # do not reduce the problem
option solve;                       # analysis information

solve;                              # solve the problem

display eat;                         # display the optimal solution
display total_cost;                  # display the optimal value
display nutr_cont.dual;             # display the optimal dual values
display food_max.slack;             # display slack variables
display eat.rc;                     # display reduced costs
```

The Diet Problem—AMPL implementation

- Change type of variables: *integer*, *binary*, ...

```
# Model file for the diet problem

set FOOD;                          # index set for foods
set NUTR;                          # index set for nutrients

var eat {j in FOOD} integer >= 0;

param cost {FOOD} > 0;             # price of foods [SEK/hg]
param eat_max {FOOD} > 0;          # maximum daily amount of food [hg]
param cont {NUTR,FOOD} >= 0;       # amount [g,\mu g,mg] of nutrient in 1hg of food
param n_min {NUTR} >= 0;           # minimum daily amount of nutrition [g, \mu g, mg]
param n_max {i in NUTR} >= n_min[i];      # maximum daily amount of nutrition

minimize total_cost:
     sum {j in FOOD} cost[j] * eat[j] ;

subject to nutr_cont {i in NUTR}:
    n_min[i] <= sum {j in FOOD} cont[i,j] * eat[j] <= n_max[i];

subject to food_max {j in FOOD}:
     eat[j] <= eat_max[j];
```

- The sensitivity analysis as described is possible only for linear programs in continuous variables (not for integer/binary; this is due to theoretical properties (see the course book, Ch. 5))

The Diet Problem—AMPL implementation

- Solution to the integrality constrained model

```
zuzana@math-pc23:~/Desktop/diet
File  Edit  View  Search  Terminal  Help
[zuzana@math-pc23 ~]$ cd Desktop/diet/
[zuzana@math-pc23 diet]$ ampl diet.run;
IBM ILOG License Manager: "university-goteborg" is accessing AMPL 12.
IBM ILOG AMPL 12.1.0 (5724-Y45) AMPL Version 20090327 (Linux 2.6.18-6-amd64)
IBM ILOG License Manager: "university-goteborg" is accessing CPLEX 12 with optio
n(s): "b e m q use=10 ".
CPLEX 12.1.0: optimal integer solution; objective 97.3
0 MIP simplex iterations
0 branch-and-bound nodes
eat [*] :=
 Beans   7
   Egg   2
  Milk   8
Potato  10
Tomato   5
;

total_cost = 97.3

[zuzana@math-pc23 diet]$ 
```

- Theory for linear optimization problems with integer/discrete constraints: see the course book and Lectures 7–10

The Diet Problem—AMPL implementation

- Print the results on a file

emacs@math-pc23.mv.rh66.ii.2.hb.s.cdal.chalmers.se

File Edit Options Buffers Tools Help

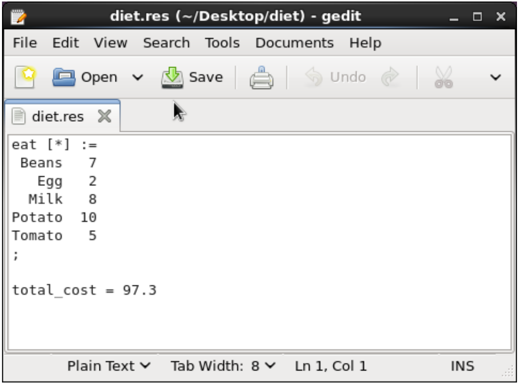
Save Undo

```
# Run file for the diet problem

model diet.mod;
data diet.dat;
options solver cplexamp;

option cplex_options 'sensitivity'; # preserve the sensitivity
option presolve 0;                  # do not reduce the problem
option solve;                       # analysis information

solve;                              # solve the problem

display eat;                         # display the optimal solution
display total_cost;                  # display the optimal value
display nutr_cont.dual;             # display the optimal dual values
display food_max.slack;             # display slack variables
display eat.rc;                     # display reduced costs

# Print interesting results on the file diet.res
display eat > diet.res;
display total_cost > diet.res;
```

-:--- diet.run All L22 (Fundamental)

The Diet Problem—AMPL implementation

- The file *diet.res* is found in the folder *diet*



Other useful AMPL commands

- AMPL options:

  option ...;
- CPLEX options:

  option cplex_options ...;
- Define higher dimensional parameters:

  param a:= [1,,*]: ... := ...*

  [2,,*]: ... := ...;*
- Set parameter value from run file:

  let param[i]:= 0;
- Display information in terminal window:

  print "...";
- *if (...) then {...} else if (...) then {...};*
- *for {i in I} {...};*
- *break;*

Assignment 1: Biofuel supply chain

Chalmers University of Technology
University of Gothenburg
Mathematical Sciences
Optimization
Zuzana Nedělková
Ann-Brith Strömberg
Caroline Granfeldt

MVE165
MMG631
Linear and integer optimization
with applications

Assignment information
March 21, 2018

Assignment 1: Biodiesel supply chain

Below is a description of the biodiesel supply chain problem such that the total profit from supplying the demand of biodiesel is maximized. The assignment tasks are to

- formulate a linear optimization model of the problem described,
- solve the problem using AMPL and CPLEX (or some other solver), and
- analyse the results and answer a number of given questions.

Study the Modeling Language for Mathematical Programming AMPL and the solver CPLEX using the following links or the recommended exercise on linear optimization and software from the course homepage before you start solving the exercises.

- `http://www.ampl.com/BOOK/download.html`
- `http://www.ampl.com/BOOKLETS/amplcplex122userguide.pdf`

To pass the assignment you should (in groups of two persons) write a self-contained report on the project work, in which you give satisfactory answers to the questions below, in the form of a PDF file. You should write the report on a computer, preferably using LaTeX. You shall also estimate the number of hours spent on this assignment and note this in your report. You may discuss the problem with other students. However, each group must hand in their own solution. The report will be checked for plagiarism via `http://www.urkund.com`.

The questions 1, 2, and 3a–3f are mandatory. In addition, students aiming at grade 4, 5, or VG must answer the questions 3g–3h, and the quality of the report must be high.

The file containing your report shall be called **Name1-Name2-Ass1.pdf**, where "Name$k$", $k = 1, 2$, is your respective family name. **Do not forget to write the authors' names also inside the report.** The report should be at most 3–4 pages plus illustrating diagrams and it must be

submitted in PingPong at latest Wednesday 18th of April 2018, 23:55.

In addition, each student must hand in an individually written report describing the distribution of the project work within the group and how the cooperation has worked out. This report must be

submitted in PingPong on 2018-04-19 between 06:00 and 23:55.

1

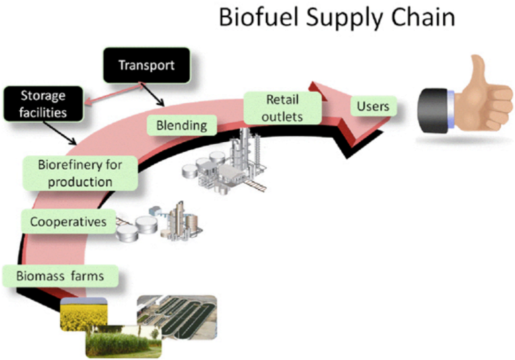
Biofuel supply chain

- Reduce oil dependence
- Reduce greenhouse effect and climate change
- Substitute fuel in transportation sector
- Biofuels can be used in existing cars
- EU quotas to use
  - 10% of energy in transport from renewable sources by 2020
  - 5% of biodiesel in diesel fuel from 2003
- Food versus fuel debate . . .
- Develop a mathematical model of the biofuel supply chain

Biofuel supply chain

The value chain typically includes:

- Feedstock production
- Biofuel production
- Blending
- Distribution
- Consumption



Assignment 1: Biodiesel supply chain

- Biodiesel supply chain problem
  - Maximize the total profit
  - Supply the demand of biodiesel
- Tasks
  - Formulate a linear optimization model
  - Model and solve the problem using AMPL and CPLEX (or other solvers)
  - Perform sensitivity analyses

Crops

- Data
  - Available area for growing crops
  - Crops: Soya, Sunflower, Cotton
  - Each crop yields an expected amount of seeds
  - Each crop has a water demand
  - The available water is limited
- Processes
  - Extraction of vegetable oils from seeds (given yields)
  - Transesterification: vegetable oil + methanol = biodiesel (given proportions)
  - Purchase methanol (given price)

Final Products

- Data
  - Three different products/blends: B5, B30, B100
  - Each product has price
  - Each product is subject to tax (higher amount of biodiesel ⇒ lower tax)
  - Demand of fuels to be delivered
- Processes
  - Blending of biodiesel and petrol diesel
  - Purchase petrol diesel (given price and availability)

Sensitivity analysis

- Analyze results and answer several important questions without changing the model
- How sensitive is the optimal solution and the optimal value to changes in the data? (Course book ch. 4–6 & lectures 4–6)
  - *Reduced costs* of a non-basic variable: the change in the objective value when the value of the corresponding variable is (marginally) increased
  - *Shadow price* of a constraint: the change in the optimal value when the RHS is (marginally) changed; equals the optimal value of the corresponding *dual variable*
  - The optimal value of the *slack variable* of a constraint indicates how much the RHS can be reduced while staying feasible
- Use these concepts to answer the questions

Others

- Cetane number
    - The quality of pure biodiesel is given by the cetane number
    - The cetane number depends on the quality of the crops
    - Requirements for the quality of each product should be incorporated in the model
- Environmental friendly objective function

Literature

- R. Fourer, D.M. Gay, and B.W. Kernighan, *AMPL: A Modeling Language for Mathematical Programming*, Duxbury Press, 2003, http://www.ampl.com/BOOK/download.html

- *IBM ILOG AMPL, Version 12.2, User's Guide, Standard (Command-line) Version Including CPLEX Directives*, IBM, May 2010, http://www.ampl.com/BOOKLETS/amplcplex122userguide.pdf

- Z. Nedělková, A.-B. Strömberg, C. Granfeldt, *Assignment 1: Biodiesel supply chain*, March 21, 2018, http://www.math.chalmers.se/Math/Grundutb/CTH/mve165/1718/

- C. Papapostolou, E. Kondili, J.K. Kaldellis, *Development and implementation of an optimisation model for biofuels supply chain*, Energy, Volume 36, Issue 10, October 2011, Pages 6019–6026

- J. Lundgren, M. Rönnqvist, P. Värbrand, *Optimization*, Studentlitteratur AB, Lund, 2010