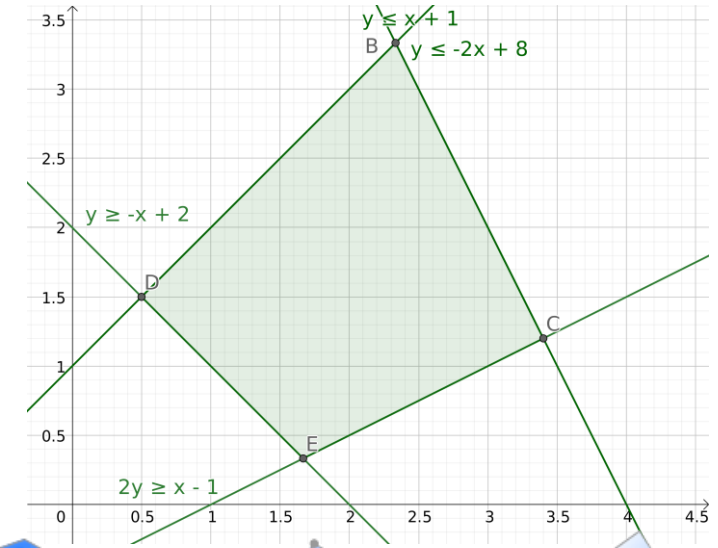
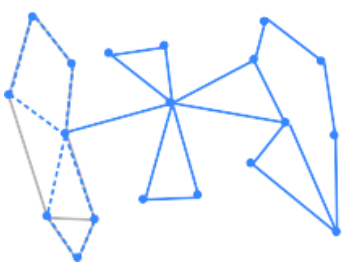
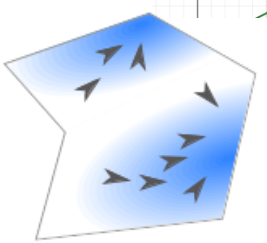
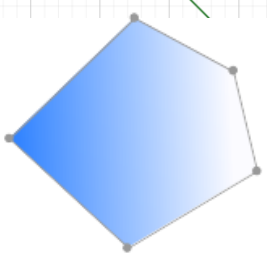
Linear Programming 1

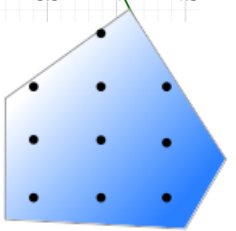
COMP4691 / 8691

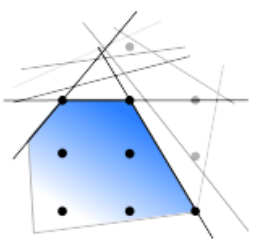


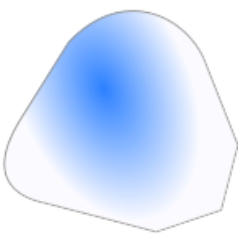










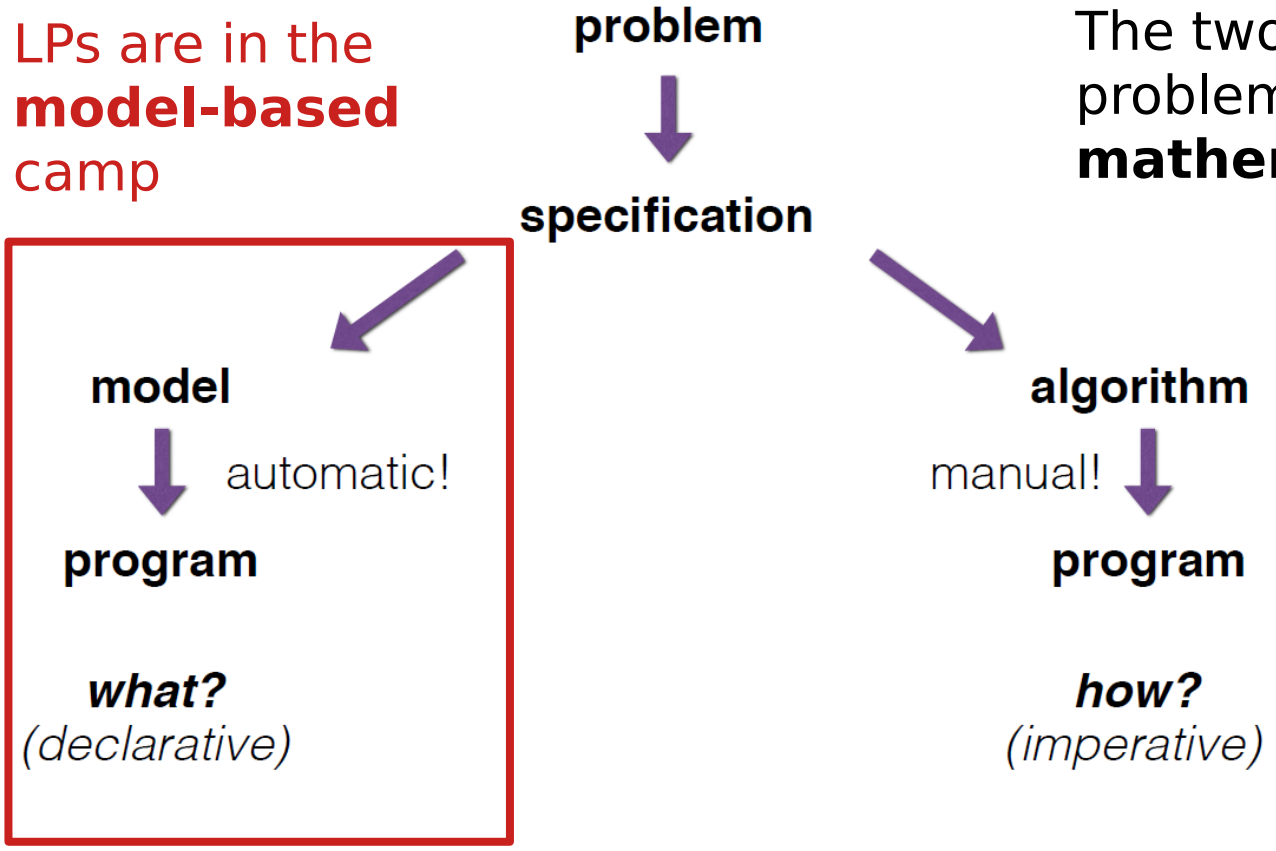


Course Outline

- Linear Programming ← next 4 lectures
- Mixed-Integer Programming
- Decomposition
- Convex Optimisation
- Local Search
- Metaheuristics
- Advanced Topics

Where do these methods fit?

LPs are in the **model-based** camp



The two techniques require problems to have specific **mathematical structure**

Benefits:

- Formulate the problem once
- Automatic solution techniques
- Guarantees of optimal solution in many problems

Downsides:

- It can be slower than algorithm-based methods
- Modelling is not trivial

A Linear Program

A constrained optimisation problem where the objective function and all constraints are **linear in the variables**.

$$\min_{x,y,z \in \mathbb{R}} \; x + 3y$$

← objective function

continuous variables →

$$\begin{array}{l|ll} \text{s.t.} & x \geq 2z & \leftarrow \text{inequality constraint} \\ & z \geq 5 & \\ & y = z + 1 & \leftarrow \text{equality constraint} \end{array}$$

subject to →

conjunction of listed constraints

So what is the answer?

z=5 x=10 y=6
Objective = 28

Linear Programming (LP)

Can be efficiently solved (polynomial time, up to millions of variables and constraints)

Guarantees the global optimum

Widely used and understood technology

So what are the downsides?

Variables are continuous, so in general **cannot represent combinatorial problems**, but its extension (MILP) often can

The **world isn’t linear**, but LPs can often provide approximations

LP Problems

Network flow
Resource allocation
Planning
Logistics
Transport
Management
Production
Assignment
Economics
Games
Control
...

One that affects us every day:

Ensuring the lights remain on

LP Topic Outline

- **LP Introduction**
  - A very brief history
- **Modelling and solving**
  - Overview of PuLP
  - Goods production example
  - General LP forms
- Feasible region and convexity
- Simplex algorithm
- Relaxations and approximations
- The dual of a linear program

A Brief History

1827 Joseph Fourier

1936 Theodore Motzkin

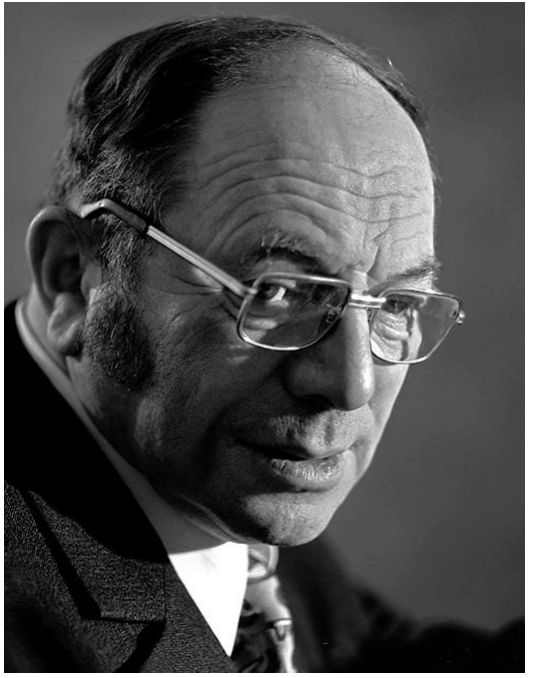
Fourier-Motzkin Elimination

Output solutions to a system of linear inequalities

1939 Leonid Kantorovich

Linear Programming

Soviet government worker optimising production in the plywood industry



A Brief History

Second World War sparked a renewed interest in improving efficiencies. In the West Linear Programming was advanced by Dantzig.

1947 George Dantzig
Simplex Algorithm

Tooling

Many options out there for modelling and solving mathematical optimisation problems (linear, mixed-integer, convex and non-convex).

Proprietary: Gurobi, CPLEX, MOSEK, KNITRO

Open source: Clp, Cbc, GLPK, Ipopt, Bonmin, Scip

Modelling Interfaces

Domain-specific languages:

- minizinc, MathProg, AMPL, GAMS, AIMMS

Libraries for more general-purpose languages:

- pyomo (python), JuMP (Julia), **PuLP (python)**, scip (C / C++ / python, Julia, Java), CasADi (C++ / python), madopt (C++ / python)

Many solvers also supply their solver-specific interface for a variety of languages.

PuLP

A python library for LP and MILP.

Connects to several backed solvers, we'll use Clp + Cbc.

Help: https://www.coin-or.org/PuLP/index.html

Let's try our simple example:

$$\min_{x,y,z \in \mathbb{R}} x + 3y$$

$$\text{s.t.} \quad x \geq 2z$$

$$z \geq 5$$

$$y = z + 1$$

First LP in PuLP

```
# first_lp_pulp.py
import pulp
from pulp import LpVariable as Var

m = pulp.LpProblem()

x = Var('x')
y = Var('y')
z = Var('z')

m += x >= 2 * z
m += z >= 5
m += y == z + 1

m += x + 3 * y

m.solve()
```

Names must be unique!

$$\min_{x,y,z \in \mathbb{R}} \; x + 3y$$

$$\text{s.t.} \begin{cases} x \geq 2z \\ z \geq 5 \\ y = z + 1 \end{cases}$$

First LP in PuLP (explicit)

```python
# first_lp_pulp.py
import pulp
from pulp import LpVariable as Var

m = pulp.LpProblem(sense=pulp.LpMinimize)

x = Var('x', lowBound=None, upBound=None)
y = Var('y', lowBound=None, upBound=None)
z = Var('z', lowBound=None, upBound=None)

m += x >= 2 * z
m += z >= 5
m += y == z + 1

m += x + 3 * y

m.solve(solver=pulp.PULP_CBC_CMD())
```

$$\min_{x,y,z \in \mathbb{R}} \; x + 3y$$

$$\begin{aligned} \text{s.t.} \quad & x \geq 2z \\ & z \geq 5 \\ & y = z + 1 \end{aligned}$$

First LP in PuLP

```
# first_lp_pulp.py
import pulp
from pulp import LpVariable as Var

m = pulp.LpProblem()

x = Var('x')
y = Var('y')
z = Var('z', lowBound=5)

m += x >= 2 * z
m += z >= 5
m += y == z + 1

m += x + 3 * y

m.solve()
```

$$\min_{x,y,z \in \mathbb{R}} \; x + 3y$$

$$\text{s.t.} \quad x \geq 2z$$

$$z \geq 5$$

$$y = z + 1$$

Printing Model / Output

```
# ↑ code from before, plus:

print(m)  # the model
print(m.status)  # status after attempted solve
print(m.objective.value())  # the solution objective value
print((x.value(), y.value(), z.value()))  # the solution values

# To make the Cbc solver more verbose:
# m.solve(solver=pulp.PULP_CBC_CMD(msg=1))
```

Infeasibility

```
# first_lp_pulp.py
import pulp
from pulp import LpVariable as Var

m = pulp.LpProblem()

x = Var('x')
y = Var('y')
z = Var('z')

m += x >= 2 * z
m += z >= 5
m += z <= 1
m += y == z + 1

m += x + 3 * y

m.solve(solver=pulp.PULP_CBC_CMD(msg=1))
```

`m.status` will be equal to -1, which is `pulp.constants.LpStatusInfeasible`

Variables may have values that might be a solution that attempts to minimise the infeasibility. So it might not be that clear that something went wrong...

So make sure you check the status!!!

Optimising Production of Goods

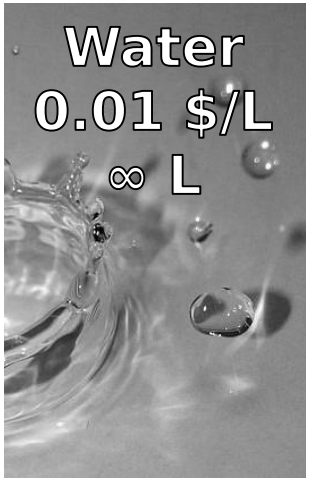
An acquaintance has recently started a **nano-brewery**, and would like help deciding what styles of beer to produce over the coming month.

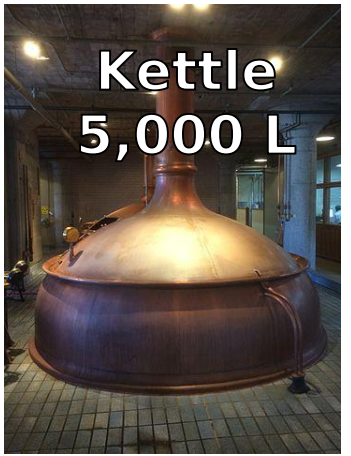
Each style uses different amounts of ingredients, and can be sold for different amounts. The ingredient suppliers have limited available stocks for sale, and the nano-brewery has a finite capacity.

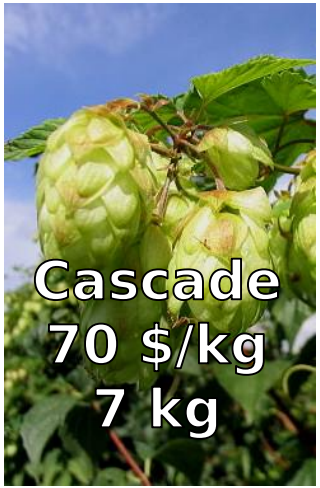
How many litres of each beer style should the nano-brewery produce in order to **maximise profit**?

Brewing

Fixed monthly cost: $17,000







Styles

IPA
6.4 $/L
1L beer requires: 1.5L Water, 25 g Wheat, 250g Barley, 3g Cascade, 0.5g Tettnang

Hefeweizen
6.1 $/L
1L beer requires: 1.3L Water, 200 g Wheat, 25g Barley, 1g Tettnang

Imperial Stout
6.2 $/L
1L beer requires: 1.8 L Water, 25 g Wheat, 350g Barley, 2g Tettnang
Contracted to produce min of 1000 L

Formulation

Indices and sets

$s \in S$ styles

$i \in I$ ingredients

Variables

b_s	beer produced	L
h_i	ingredient purchased	g or L

Parameters

$r_{s,i}$	ingredients of the recipe	g/L or L/L
p_i	price of ingredient	$/g or $/L
q_s	sale price	$/L
l_i	ingredient limit	g or L
k	kettle capacity	L
f	fermenter capacity	L
c	fixed monthly cost	$
$\underline{b}_s$	minimum beer produced	L

Vector notation:

$$\min_{b,h} \; c + \overbrace{\sum_{i \in I} p_i h_i}^{p^\intercal h} - \overbrace{\sum_{s \in S} q_s b_s}^{q^\intercal b}$$

$$\text{s.t.} \quad h_i \le l_i \quad \forall i \in I$$

$$h_i = \sum_{s \in S} r_{s,i} b_s \quad \forall i \in I$$

$$b_s \ge \underline{b}_s \quad \forall s \in S$$

$$\sum_{s \in S} b_s \le f$$

What will be the objective function?

What will be the constraints?

Implementation Remarks

In the mathematical formulation I used concise parameter / variable names, because the maths is easier to work with that way (e.g., by hand).

Best software practices apply when coding model: use self-descriptive parameter / variable names where possible, avoid duplication, etc.

- b might become beer

DEMO: Let's run the brewery problem.

- This demo is in the gitlab (materials/demos/brewery.py)

Active/Binding Constraints

Some of the inequalities may not be influencing the solution. How can we identify this?

Constraint is active is it is satisfied as an equality, e.g.,
$x \geq 1000$ when the value of x is exactly 1000

If the constraint is strictly satisfied, then it is not active.

If we change an active constraint, does it change the solution? How can we test this?

Later on: duality will provide us with an alternative.

DEMO: Let's try this in the code.

Cancel Contract?

We can cancel the contract to deliver 1000 L of Imperial Stout at a penalty of $1000. What should we do? (ignoring the damage this would do to our reputation!)

We can add $1000 to the fixed cost.

We can set the minimum of the Imperial Stout to 0.

DEMO: Let’s try this in the code.

General Form(s)

$$\min_{x \in \mathbb{R}^n} \; c^\intercal x$$

Element-wise inequality

$$\text{s.t. } Ax \leq b$$

How many variables, how many constraints?

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

n variables, m inequality constraints

$$A = \begin{bmatrix} a_{1,1} & \cdots & a_{1,n} \\ \vdots & \ddots & \vdots \\ a_{m,1} & \cdots & a_{m,n} \end{bmatrix}$$

General, but where are the equalities?

$$u = v \iff u \geq v \wedge u \leq v$$

$$b = \begin{bmatrix} b_1 \\ \vdots \\ b_m \end{bmatrix} \quad c = \begin{bmatrix} c_1 \\ \vdots \\ c_n \end{bmatrix}$$

$$\begin{aligned} a_{1,1}x_1 + \ldots + a_{1,n}x_n &\leq b_1 \\ &\vdots \\ a_{m,1}x_1 + \ldots + a_{m,n}x_n &\leq b_m \end{aligned}$$

Columns for variables, rows for constraints

General Form(s)

$$\min_{x \in \mathbb{R}^n} \; c^\intercal x \quad \text{s.t. } Ax \leq b$$

$$\max_{x \in \mathbb{R}^n} \; c^\intercal x \quad \text{s.t. } Ax \leq b$$

$$\min_{x \in \mathbb{R}^n} \; c^\intercal x \quad \text{s.t. } Ax = b, \; x \geq 0$$

etc...

Note: for the same problem, x, A, b and c will take on different values in different forms.

Since they are all general, we should be able to convert between them. How do we convert a min to a max problem?

$$\min_{x \in \mathbb{R}^n} \; c^\intercal x \quad \text{s.t. } Ax \leq b \quad \longrightarrow \quad \max_{x \in \mathbb{R}^n} \; -c^\intercal x \quad \text{s.t. } Ax \leq b$$

Standard Form

You will often see texts refer to the “standard form” of an LP, which is just one of these general forms. Be careful, there is no *standard* standard form, authors will pick their preferred one.

It is worth learning how to **convert** a problem between the different forms, to understand their equivalence, and to learn how to reformulate constraints.

Particular solvers / algorithms are implemented to work with one particular form internally. A good modelling layer will allow you to express the problem freely, and automatically do the conversion for the solver.

Converting Between Forms

Converting an equality to two inequalities:

$$u = v \iff u \geq v \wedge u \leq v$$

Alternatively, the process of elimination can be used to eliminate one variable for each equality constraint:

$$x + y + z = 5 \implies x = 5 - y - z$$

Then eliminate x from all other constraints and objective*

* Careful to account for the domain of x, e.g., if $x \in [0, \infty)$ need $5 - y - z \geq 0$

What about the reverse, eliminating inequalities?

Inequalities are more *general* than equalities, so generally cannot eliminate them, but we can convert a complicated inequality into a complicated equality + simple inequality.

Slack Variables

Inequalities

$$\min_{x \in \mathbb{R}^n} \; c^\intercal x$$
$$\text{s.t. } Ax \leq b$$

to

Equalities +
Nonnegative variables

$$\min_{x \in \mathbb{R}^n} \; c^\intercal x$$
$$\text{s.t. } Ax = b$$
$$x \geq 0$$

$$y + z \leq 5$$

Introduce new non-negative “slack” variable to LHS and set equality:

$$y + z + s = 5, \quad s \geq 0$$

s is not in the objective function, so is free to vary. It automatically takes up the slack on the LHS to emulate an inequality.

$$5 + 0 + s = 5 \qquad s = 0$$
$$0 + 5 + s = 5 \qquad s = 0$$
$$1 + 1 + s = 5 \qquad s = 3$$
$$5 + 5 + s = 5 \qquad s = -5$$

not consistent, s must be non-negative

Non-negative Variables

Inequalities

$$\min_{x \in \mathbb{R}^n} \; c^\intercal x$$
$$\text{s.t. } Ax \leq b$$

to

Equalities +
Nonnegative variables

$$\min_{x \in \mathbb{R}^n} \; c^\intercal x$$
$$\text{s.t. } Ax = b$$
$$x \geq 0$$

$$y + z \leq 5$$

If y and z can take on any real number, we can replace them by new non-negative variables:

$$y^+, y^-, z^+, z^- \geq 0$$
$$y = y^+ - y^-, \; z = z^+ - z^-$$

Now eliminate all occurrences of y and x:

$$y^+ - y^- + z^+ - z^- \leq 5$$

But wait... for $y = y^+ - y^-$

$$5 = 5 - 0$$
$$5 = 1000005 - 1000000$$

Does this matter?

Conversion Example

Let's convert the following problem to an equality + non-negative variable form.

$$\begin{aligned} \min_{x,y,z} \quad & x + 3y \\ \text{s.t.} \quad & x \geq 2z \\ & z \geq 5 \\ & y = z + 1 \end{aligned}$$

$$\begin{aligned} \min_{x,y} \quad & x + 3y \\ \text{s.t.} \quad & x - 2y \geq -2 \\ & y \geq 6 \\ & [z = y - 1] \end{aligned}$$

$$\begin{aligned} \min_{x,y'} \quad & x + 3y' + 18 \\ \text{s.t.} \quad & x - 2y' \geq 10 \\ & y' \geq 0 \\ & [y = y' + 6] \end{aligned}$$

$$\begin{aligned} \min_{x^+,x^-,y',s} \quad & x^+ - x^- + 3y' + 18 \\ \text{s.t.} \quad & x^+ - x^- - 2y' - s = 10 \\ & y', x^+, x^-, s \geq 0 \\ & [x = x^+ - x^-] \end{aligned}$$

Next Week

- LP Introduction
  - A very brief history
- Modelling and solving
  - Overview of PuLP
  - Goods production example
  - General LP forms
- Feasible region and convexity
- Simplex algorithm
- Relaxations and approximations
- The dual of a linear program

Image Attributions

- https://commons.wikimedia.org/wiki/File:Bombeta_de_Llum.JPG#/media/File:Bombeta_de_Llum.JPG
- By Андрей Богданов (Andrei-bogdanoffyandex.ru) - [1], CC BY 3.0, https://commons.wikimedia.org/w/index.php?curid=11494406
- http://www.mathopt.org/?nav=album
- Lucash [CC BY-SA 3.0 (https://creativecommons.org/licenses/by-sa/3.0)], from Wikimedia Commons
- böhringer friedrich [CC BY-SA 2.5 (https://creativecommons.org/licenses/by-sa/2.5)], from Wikimedia Commons
- No machine-readable author provided. Hagen Graebner assumed (based on copyright claims). [CC BY-SA 2.5 (https://creativecommons.org/licenses/by-sa/2.5)], via Wikimedia Commons
- AnRo0002 [CC0], from Wikimedia Commons
- (Henningklevjer) [CC BY-SA 2.5 (https://creativecommons.org/licenses/by-sa/2.5)], via Wikimedia Commons
- Аимаина хикари [CC0], from Wikimedia Commons
- CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=106903
- By Thcipriani - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=70328709
- By Picardin - Own work, CC BY 4.0, https://commons.wikimedia.org/w/index.php?curid=39527449
- By Bernt Rostad from Oslo, Norway - HooDoo Brewing Company, CC BY 2.0, https://commons.wikimedia.org/w/index.php?curid=75850513