Applications of Operations Research and Mathematical Programming

Martin Branda

Charles University in Prague
Faculty of Mathematics and Physics
Department of Probability and Mathematical Statistics

Computational Aspects of Optimization

Contents

1. Marketing – Optimization of advertising campaigns
2. Logistic – Vehicle routing problems
3. Scheduling – Reparations of oil platforms
4. Insurance – Pricing in nonlife insurance
5. Power industry – Bidding, power plant operations
6. Environment – Inverse modelling in atmosphere

Operations Research and Mathematical Programming

Goal: improve/stabilize/set of ...

- Problem understanding
- Problem description – probabilistic, statistical and econometric models
- Optimization – mathematical programming
- Verification – backtesting, stresstesting
- Implementation (Decision Support System)
- Decision

Operations Research/Management Science

Operations Research and Mathematical Programming

Goal: improve/stabilize/set of ...

- Problem understanding
- Problem description – probabilistic, statistical and econometric models
- Optimization – mathematical programming
- Verification – backtesting, stresstesting
- Implementation (Decision Support System)
- Decision

Operations Research/Management Science

Operations Research and Mathematical Programming

Goal: improve/stabilize/set of ...

- Problem understanding
- Problem description – probabilistic, statistical and econometric models
- Optimization – mathematical programming
- Verification – backtesting, stresstesting
- Implementation (Decision Support System)
- Decision

Operations Research/Management Science

Optimization of advertising campaigns

- **Goal** – maximization of the effectiveness of a advertising campaign given its costs or vice versa
- **Data** – "peoplemeters", public opinion poll, historical advertising campaigns
- **Target group** – (potential) customers (age, region, education level ...)
- **Effectiveness criteria**
  - GRP (TRP) – rating(s)
  - Effective frequency – relative number of persons in the target group hit k-times by the campaign
- Nonlinear or integer programming

Optimization of advertising campaigns

- **Goal** – maximization of the effectiveness of a advertising campaign given its costs or vice versa
- **Data** – "peoplemeters", public opinion poll, historical advertising campaigns
- **Target group** – (potential) customers (age, region, education level ...)
- **Effectiveness criteria**
  - GRP (TRP) – rating(s)
  - Effective frequency – relative number of persons in the target group hit k-times by the campaign
- Nonlinear or integer programming

Vehicle routing problems

- **Goal** – maximize *filling rate* of the ships (operation planning), fleet composition, i.e. capacity and number of ships (strategic planning)
- **Rich Vehicle Routing Problem**
  - time windows
  - heterogeneous fleet
  - several depots and inter-depot trips
  - several trips during the planning horizon
  - *non-Euclidean distances* (fjords)
- Integer programming :-(, constructive heuristics and tabu search

- **Downstream logistics optimization at EWOS Norway**. Research report, Molde University, submitted. With K. Haugen, J. Novotný, A. Olstad

Vehicle routing problems

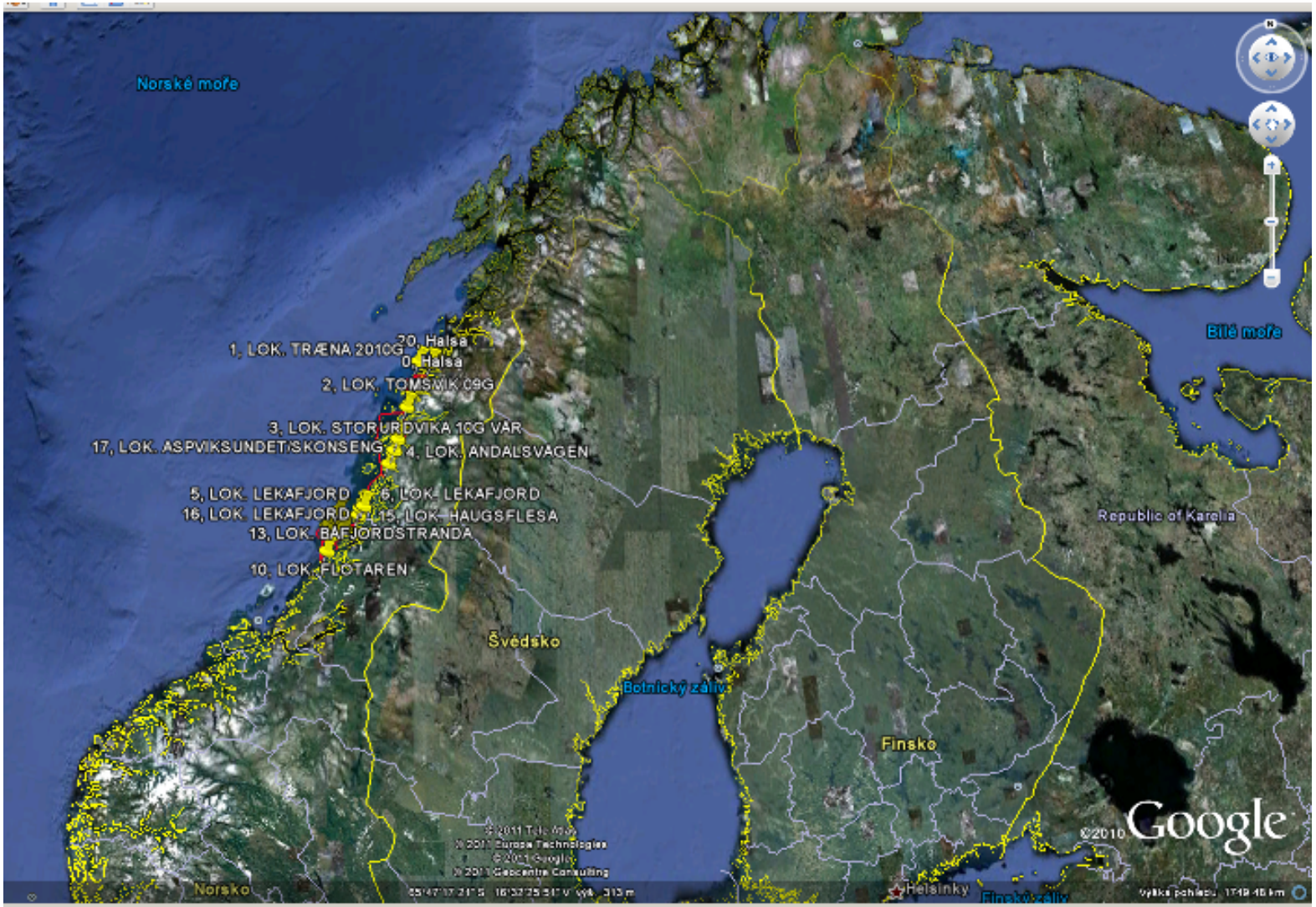
- Traveling Salesman Problem
- Uncapacitated Vehicle Routing Problem (VRP)
- Capacitated VRP
- VRP with Time Windows
- ...

Vehicle routing problems

Our approach:

1. Mathematical formulation
2. Solving using GAMS based on historical data
3. Heuristic(s) implementation
4. Implementation to a Decision Support System

In the future, *Inventory Routing* ...


Norské moře
Bílé moře
1, LOK. TRÆNA 201CG
20, Halsa
0, Halsa
2, LOK. TOMSVIK 09G
3, LOK. STORURDVIKA 10G VÅR
17, LOK. ASPVIKSUNDET/SKONSENG
4, LOK. ANDALSVÅGEN
5, LOK. LEKAFJORD
6, LOK. LEKAFJORD
16, LOK. LEKAFJORD
15, LOK. HAUGSFLESA
13, LOK. BÅFJORDSTRANDA
10, LOK. FLOTAREN
Švédsko
Botnický záliv
Finsko
Republic of Karelia
Norsko
Helsinky
Finský záliv
©2010 Google

Optimization of reparations of oil platforms

- **Goal** – send the right workers to the oil platforms taking into account uncertainty (bad weather – helicopter(s) cannot fly – schedule is delayed)
- **Scheduling** – jobs = reparations, machines = workers (highly costly and educated)
- Integer and stochastic programming

- M.B., J. Novotný, A. Olstad: **Fixed interval scheduling under uncertainty - a tabu search algorithm for an extended robust coloring formulation**. Computers & Industrial Engineering 93, 45–54, 2016.

Pricing in nonlife insurance

- **Goal** – optimization of prices in MTPL/CASCO insurance taking into account riskiness of contracts and competitiveness of prices
- **Risk** – compound distribution of random over 1y (Data-mining & GLM)
- Nonlinear stochastic optimization (probabilistic or expectation constraints)

- M.B. **Optimization approaches to multiplicative tariff of rates estimation in non-life insurance**. Asia-Pacific Journal of Operational Research 31 (5), 1450032, 17 pages, 2014.
- M.B. **Underwriting risk control in non-life insurance via generalized linear models and stochastic programming**. Proceedings of the 30th International Conference on MME 2012, 61–66.

Multiplicative tariff of rates

		GLM	SP model (ind.)	SP model (col.)
TG	up to 1000 ccm	3 805	9 318	5 305
TG	1000–1349 ccm	4 104	9 979	5 563
TG	1350–1849 ccm	4 918	11 704	6 296
TG	1850–2499 ccm	5 748	13 380	7 125
TG	over 2500 ccm	7 792	17 453	9 169
Region	Capital city	1.61	1.41	1.41
Region	Large towns	1.16	1.18	1.19
Region	Small towns	1.00	1.00	1.00
Region	Others	1.00	1.00	1.00
Age	18–30y	1.28	1.26	1.27
Age	31–65y	1.06	1.11	1.11
Age	over 66y	1.00	1.00	1.00
DL less that 5y	YES	1.00	1.00	1.00
DL more that 5y	NO	1.19	1.13	1.12

Optimization in power industry

Energy markets

- **Goal** – profit maximization and risk minimization
- **Day-ahead bidding** from wind (power) farm
- Nonlinear stochastic programming

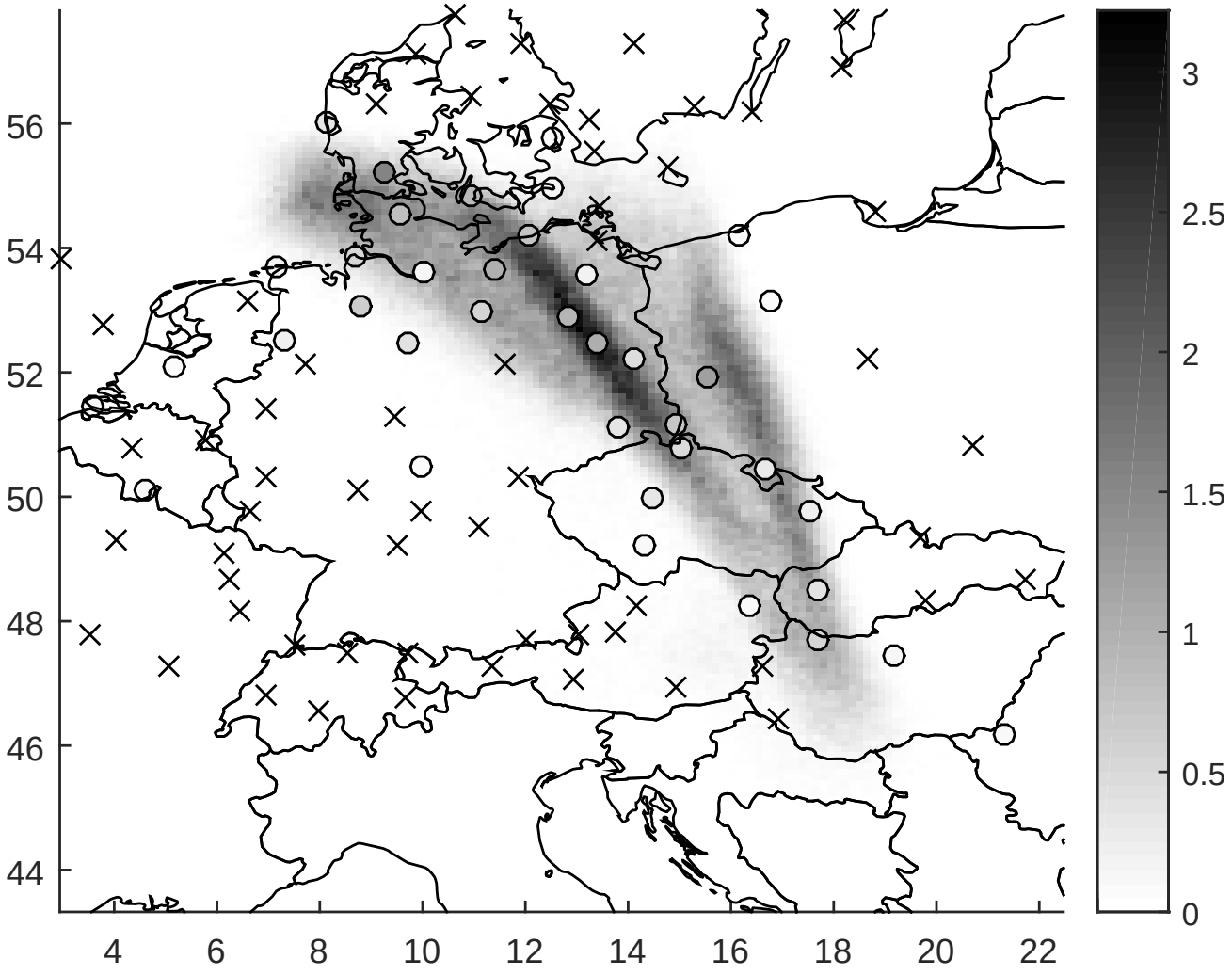
Power plant operations

- **Goal** – profit maximization and risk minimization
- **Coal power plants** – demand seasonality, ...
- Stochastic linear programming (multistage/multiperiod)

Inverse modelling in atmosphere

- **Goal** – identification of the source and of the amount released into the atmosphere
- **Standard approach** – Bayesian models
- **New approach** – Sparse optimization – Nonlinear/quadratic integer programming
- **Application**: (nuclear) power plants accidents, volcano accidents, nuclear tests, ...

- L. Adam, M.B.: **Sparse optimization for inverse problems in atmospheric modelling.** Environmental Modelling & Software 79, 256–266, 2016. (free Matlab codes available)
- **Project:** http://stradi.utia.cas.cz/


56
54
52
50
48
46
44
4
6
8
10
12
14
16
18
20
22
3
2.5
2
1.5
1
0.5
0

Linear programming – simplex algorithm, duality and dual simplex algorithm

Martin Branda

Charles University in Prague
Faculty of Mathematics and Physics
Department of Probability and Mathematical Statistics

Computational Aspects of Optimization

Content

1. Linear programming
2. Primal simplex algorithm
3. Duality in linear programming
4. Dual simplex algorithm
5. Software tools for LP

Linear programming

Standard form LP

$$\begin{aligned} \min\ & c^T x \\ \text{s.t. } & Ax = b, \\ & x \geq 0. \end{aligned}$$

$A \in \mathbb{R}^{m \times n}$, $h(A) = h(A|b) = m$.

$$M = \{x \in \mathbb{R}^n : \ Ax = b, \ x \geq 0\}.$$

Linear programming

Decomposition of M:

- **Convex polyhedron** P – uniquely determined by its vertices (convex hull)
- **Convex polyhedral cone** K – generated by extreme directions (positive hull)

Direct method (evaluate all vertices and extreme directions, compute the values of the objective function ...)

Linear programming trichotomy

One of these cases is valid:

1. $M = \emptyset$
2. $M \neq \emptyset$: the problem is unbounded
3. $M \neq \emptyset$: the problem has an optimal solution (at least one of the solutions is vertex)

Content

1. Linear programming
2. Primal simplex algorithm
3. Duality in linear programming
4. Dual simplex algorithm
5. Software tools for LP

Simplex algorithm – basis

Basis B = regular square submatrix of A, i.e.

$$A = (B|N).$$

We also consider $B = \{i_1, \ldots, i_m\}$.
We split the objective coefficients and the decision vector accordingly:

$$c^T = (c_B^T, c_N^T),$$
$$x^T(B) = (x_B^T(B), x_N^T(B)),$$

where

$$B \cdot x_B(B) = b, \; x_N(B) \equiv 0.$$

- Feasible basis, optimal basis.
- Basic solution(s).
- **ASS.** non-degenerate problem (basic solutions have m positive elements) $\rightarrow$ finiteness of the simplex alg.

Simplex algorithm – simplex table

			x^T
			c^T
c_B	x_B	$B^{-1}b$	$B^{-1}A$
		$c_B^T B^{-1} b$	$c_B^T B^{-1} A - c^T$

Simplex algorithm – simplex table

- **Feasibility condition:**

$$B^{-1}b \geq 0.$$

- **Optimality condition:**

$$c_B^T B^{-1} A - c^T \leq 0.$$

Simplex algorithm – a step

If the optimality condition is not fulfilled:

- Denote the criterion row by

$$\delta^T = c_B^T B^{-1} A - c^T.$$

- Find $\delta_i > 0$ and denote the corresponding column by

$$\rho = B^{-1} A_{\cdot,i}.$$

- Minimize the ratios

$$\hat{u} = \arg\min \left\{ \frac{x_u(B)}{\rho_u} : \ \rho_u > 0, \ u \in B \right\}.$$

- Substitute $x_{\hat{u}}$ by x_i in the basic variables, i.e. $\hat{B} = B \setminus \{\hat{u}\} \cup \{i\}$.

Simplex algorithm – a step

Denote by $\hat{B}$ the new basis. Define a **direction**

$$\begin{aligned}\Delta_u &= -\rho_u,\ u \in B,\\ \Delta_i &= 1,\\ \Delta_j &= 0,\ j \notin B \cup \{i\}.\end{aligned}$$

If $\Delta \leq 0$, then the problem is unbounded ($c^T x(\hat{B}) \to -\infty$). Otherwise, we can **move from the current basic solution to another one**

$$x(\hat{B}) = x(B) + t\Delta,$$

where $0 \leq t \leq \frac{x_{\hat{u}}(B)}{\rho_{\hat{u}}}$. We should prove that the new solution is a feasible basic solution ($\hat{B}$ is regular, $x(\hat{B}) \geq 0$, $\hat{B}x(\hat{B}) = b$) and that the objective value decreases ...

Simplex algorithm – pivot rules

... rules for selecting the entering variable if there are several possibilities:

- **Largest coefficient** in the objective function
- **Largest decrease** of the objective function
- **Steepest edge** – choose an improving variable whose entering into the basis moves the current basic feasible solution in a direction closest to the direction of the vector c

$$\max \frac{c^T(x_{new} - x_{old})}{\|x_{new} - x_{old}\|}.$$

  Computationally the most successful.
- **Blands's rule** – choose the improving variable with the smallest index, and if there are several possibilities for the leaving variable, also take the one with the smallest index (prevents cycling)

Matoušek and Gärtner (2007).

Content

1. Linear programming
2. Primal simplex algorithm
3. Duality in linear programming
4. Dual simplex algorithm
5. Software tools for LP

Transportation problem

- $i = 1, \ldots, n$ – suppliers
- $j = 1, \ldots, m$ – customers
- x_{ij} – decision variable: amount transported from i to j
- c_{ij} – costs for transported unit
- a_i – capacity
- b_j – demand

ASS. $\sum_{i=1}^{n} a_i \geq \sum_{j=1}^{m} b_j$.
(Sometimes $a_i, b_j \in \mathbb{N}$.)

Transportation problem

Primal problem

$$
\begin{aligned}
\min \ & \sum_{i=1}^{n} \sum_{j=1}^{m} c_{ij} x_{ij} \\
\text{s.t.} \ & \sum_{j=1}^{m} x_{ij} \leq a_i, \ \ i = 1, \ldots, n, \\
& \sum_{i=1}^{n} x_{ij} \geq b_j, \ \ j = 1, \ldots, m, \\
& x_{ij} \geq 0.
\end{aligned}
$$

Transportation problem

Dual problem

$$\begin{aligned} \max \ & \sum_{i=1}^{n} a_i u_i + \sum_{j=1}^{m} b_j v_j \\ \text{s.t. } & u_i + v_j \leq c_{ij}, \\ & u_i \leq 0, \\ & v_j \geq 0. \end{aligned}$$

Interpretation: $-u_i$ (shadow) price for buying a unit of goods at i, v_j (shadow) price for selling at j.

Transportation problem

Competition between the transportation company (which minimizes the transportation costs) and an "agent" (who maximizes the earnings):

$$\sum_{i=1}^{n} a_i u_i + \sum_{j=1}^{m} b_j v_j \leq \sum_{i=1}^{n} \sum_{j=1}^{m} c_{ij} x_{ij}$$

Linear programming duality

Primal problem

$$\begin{aligned} \text{(P)} \quad \min\ & c^T x \\ \text{s.t.}\ & Ax \geq b, \\ & x \geq 0. \end{aligned}$$

and corresponding **dual problem**

$$\begin{aligned} \text{(D)} \quad \max\ & b^T y \\ \text{s.t.}\ & A^T y \leq c, \\ & y \geq 0. \end{aligned}$$

Linear programming duality

Denote

$$M = \{x \in \mathbb{R}^n : \ Ax \geq b, \ x \geq 0\},$$
$$N = \{y \in \mathbb{R}^m : \ A^T y \leq c, \ y \geq 0\},$$

Weak duality theorem:

$$b^T y \leq c^T x, \ \forall x \in M, \forall y \in N.$$

Equality holds if and only if (iff) complementarity slackness conditions are fulfilled:

$$\begin{aligned} y^T(Ax - b) &= 0, \\ x^T(A^T y - c) &= 0. \end{aligned}$$

Linear programming duality

Apply KKT optimality conditions to primal LP ...

Linear programming duality

- **Duality theorem:** If $M \neq \emptyset$ and $N \neq \emptyset$, than the problems (P), (D) have optimal solutions.
- **Strong duality theorem:** The problem (P) has an optimal solution if and only if the dual problem (D) has an optimal solution. If one problem has an optimal solution, than the optimal values are equal.

Content

1. Linear programming
2. Primal simplex algorithm
3. Duality in linear programming
4. Dual simplex algorithm
5. Software tools for LP

Linear programming duality

Primal problem (standard form)

$$\begin{aligned} \min\ & c^T x \\ \text{s.t. } & Ax = b, \\ & x \geq 0. \end{aligned}$$

and corresponding **dual problem**

$$\begin{aligned} \max\ & b^T y \\ \text{s.t. } & A^T y \leq c, \\ & y \in \mathbb{R}^m. \end{aligned}$$

Dual simplex algorithm

Dual simplex algorithm works with

- **dual feasible basis** B and
- **basic dual solution** $y(B)$,

where

$$\begin{aligned} B^T y(B) &= c_B^T, \\ N^T y(B) &\leq c_N^T. \end{aligned}$$

Dual simplex algorithm

Primal feasibility $B^{-1}b \geq 0$ **is violated** until reaching the optimal solution.

Primal optimality condition is always fulfilled:

$$c_B^T B^{-1} A - c^T \leq 0.$$

Using $A = (B|N)$, $c^T = (c_B^T, c_N^T)$, we have

$$\begin{aligned} c_B^T B^{-1} B - c_B^T &= 0, \\ c_B^T B^{-1} N - c_N^T &\leq 0, \end{aligned}$$

Setting $\hat{u} = (B^{-1})^T c_B$

$$\begin{aligned} B^T \hat{u} &= c_B^T, \\ N^T \hat{u} &\leq c_N^T. \end{aligned}$$

Thus, $\hat{u}$ is a basic dual solution.

Dual simplex algorithm – a step

... uses the same simplex table.

- Find index $u \in B$ such that $x_u(B) < 0$ and denote the corresponding row by
$$\tau^T = (B^{-1}A)_{u,\cdot\cdot}$$
- Denote the criterion row by
$$\delta^T = c_B^T B^{-1} A - c^T \leq 0.$$
- Minimize the ratios
$$\hat{i} = \arg\min \left\{ \frac{\delta_i}{\tau_i} : \ \tau_i < 0 \right\}.$$
- Substitute x_u by $x_{\hat{i}}$ in the basic variables, i.e. $\hat{B} = B \setminus \{u\} \cup \{\hat{i}\}$. We move to another basic dual solution.

Example – dual simplex algorithm

$$
\begin{aligned}
\min 4x_1 + 5x_2 & \\
x_1 + 4x_2 &\geq 5, \\
3x_1 + 2x_2 &\geq 7, \\
x_1, x_2 &\geq 0.
\end{aligned}
$$

Example – dual simplex algorithm

			4	5	0	0
			x_1	x_2	x_3	x_4
0	x_3	-5	-1	-4	1	0
0	x_4	-7	-3	-2	0	1
		0	-4	-5	0	0
0	x_3	-8/3	0	-10/3	1	-1/3
4	x_1	7/3	1	2/3	0	-1/3
		28/3	0	-7/3	0	-4/3
5	x_2	8/10	0	1	-3/10	1/10
4	x_1	18/10	1	0	2/10	-4/10
		112/10	0	0	-7/10	-11/10

The last solution is primal and dual feasible, thus optimal.

Content

1. Linear programming
2. Primal simplex algorithm
3. Duality in linear programming
4. Dual simplex algorithm
5. Software tools for LP

Software tools for LP

- Matlab
- Mathematica
- GAMS
- MS Excel
- ...

Literature

- Bazaraa, M.S., Sherali, H.D., and Shetty, C.M. (2006). Nonlinear programming: theory and algorithms, Wiley, Singapore, 3rd edition.
- Boyd, S., Vandenberghe, L. (2004). Convex Optimization, Cambridge University Press, Cambridge.
- P. Lachout (2011). Matematické programování. Skripta k (zaniklé) přednášce Optimalizace I (IN CZECH).
- Matoušek and Gärtner (2007). Understanding and using linear programming, Springer.

Questions?

e-mail: branda@karlin.mff.cuni.cz
web: http://artax.karlin.mff.cuni.cz/~ branm1am

Linear programming – duality and shadow prices
Software tools for LP
Quadratic programming – Wolfe algorithm

Martin Branda

Charles University in Prague
Faculty of Mathematics and Physics
Department of Probability and Mathematical Statistics

Computational Aspects of Optimization

Content

1. Duality in linear programming
2. Software tools for LP
3. Quadratic programming problems and Wolfe algorithm

Linear programming duality

Primal problem

$$\begin{aligned} &\min\ c^T x \\ \text{(P)}\ \text{s.t.}\ &Ax \geq b, \\ &x \geq 0. \end{aligned}$$

and corresponding **dual problem**

$$\begin{aligned} &\max\ b^T y \\ \text{(D)}\ \text{s.t.}\ &A^T y \leq c, \\ &y \geq 0. \end{aligned}$$

Linear programming duality

		x_1	x_2	x_3		
		≥ 0	≤ 0	$\in \mathbb{R}$		
		1	2	3	$\leq$	b_1
		4	5	6	$\geq$	b_2
		7	8	9	$=$	b_3
		c_1	c_2	c_3	min	

Linear programming duality

		x_1	x_2	x_3		
		≥ 0	≤ 0	$\in \mathbb{R}$		
y_1	≥ 0	1	2	3	$\geq$	b_1
y_2	≤ 0	4	5	6	$\leq$	b_2
y_3	$\in \mathbb{R}$	7	8	9	$=$	b_3
		$\leq$	$\geq$	$=$		max
		c_1	c_2	c_3	min	

Linear programming duality

Denote

$$M = \{x \in \mathbb{R}^n : \ Ax \geq b, \ x \geq 0\},$$
$$N = \{y \in \mathbb{R}^m : \ A^T y \leq c, \ y \geq 0\}.$$

Weak duality theorem:

$$b^T y \leq c^T x, \ \forall x \in M, \forall y \in N.$$

Equality holds if and only if (iff) complementarity slackness conditions are fulfilled:

$$y^T(Ax - b) \ = 0,$$
$$x^T(A^T y - c) \ = 0.$$

Linear programming duality

Apply KKT optimality conditions to primal LP ...

Linear programming duality

- **Duality theorem:** If $M \neq \emptyset$ and $N \neq \emptyset$, than the problems (P), (D) have optimal solutions.
- **Strong duality theorem:** The problem (P) has an optimal solution if and only if the dual problem (D) has an optimal solution. If one problem has an optimal solution, than the optimal values are equal.

Duality – production planning

Optimize the production of the following products V_1, V_2, V_3 made from materials M_1, M_2.

	V_1	V_2	V_3	Constraints
M_1	1	0	2	54 kg
M_2	2	3	1	30 kg
Gain (\$/kg)	10	15	10	

Duality

Primal problem

$$
(P)\quad
\begin{array}{lrcrcrcl}
\max & 10x_1 & + & 15x_2 & + & 10x_3 & & \\
\text{s.t.} & x_1 & & & + & 2x_3 & \leq & 54, \\
 & 2x_1 & + & 3x_2 & + & x_3 & \leq & 30, \\
 & x_1 & & & & & \geq & 0, \\
 & & & x_2 & & & \geq & 0, \\
 & & & & & x_3 & \geq & 0.
\end{array}
$$

Dual problem

$$
(D)\quad
\begin{array}{lrcrcl}
\min & 54y_1 & + & 30y_2 & & \\
\text{s.t.} & y_1 & + & 2y_2 & \geq & 10, \\
 & & & 3y_2 & \geq & 15, \\
 & 2y_1 & + & y_2 & \geq & 10, \\
 & y_1 & & & \geq & 0, \\
 & & & y_2 & \geq & 0.
\end{array}
$$

Duality

Optimal solution of (D) $\hat{y} = \left(\frac{5}{2}, 5\right)$.
Using the complementarity slackness conditions $\hat{x} = (0, 1, 27)$.
The optimal values (gains) of (P) and (D) are 285.

- Both (P) constraints are fulfilled with equality, thus there in no material left.
- Dual variables are called **shadow prices** and represent the prices of sources (materials).
- **Sensitivity**: If we increase (P) r.h.s. by one, then the objective value increases by the shadow price.
- The first constraint of (D) is fulfilled with strict inequality with the difference 2.5 \$, called **reduced prices**, and the first product is not produced. The producer should increase the gain from V_1 by this amount to become profitable.

Transportation problem

- x_{ij} – decision variable: amount transported from i to j
- c_{ij} – costs for transported unit
- a_i – capacity
- b_j – demand

ASS. $\sum_{i=1}^{n} a_i \geq \sum_{j=1}^{m} b_j$.
(Sometimes $a_i, b_j \in \mathbb{N}$.)

Transportation problem

Primal problem

$$
\begin{aligned}
\min \ & \sum_{i=1}^{n} \sum_{j=1}^{m} c_{ij} x_{ij} \\
\text{s.t.} \ & \sum_{j=1}^{m} x_{ij} \leq a_i, \ \ i = 1, \ldots, n, \\
& \sum_{i=1}^{n} x_{ij} \geq b_j, \ \ j = 1, \ldots, m, \\
& x_{ij} \geq 0.
\end{aligned}
$$

Transportation problem

Dual problem

$$\begin{aligned} \max \ & \sum_{i=1}^{n} a_i u_i + \sum_{j=1}^{m} b_j v_j \\ \text{s.t. } & u_i + v_j \leq c_{ij}, \\ & u_i \leq 0, \\ & v_j \geq 0. \end{aligned}$$

Interpretation: $-u_i$ price for buying a unit of goods at i, v_j price for selling at j.

Transportation problem

Competition between the transportation company (which minimizes the transportation costs) and an "agent" (who maximizes the earnings):

$$\sum_{i=1}^{n} a_i u_i + \sum_{j=1}^{m} b_j v_j \leq \sum_{i=1}^{n} \sum_{j=1}^{m} c_{ij} x_{ij}$$

Content

1 Duality in linear programming

2 Software tools for LP

3 Quadratic programming problems and Wolfe algorithm

Software tools for LP

- **Matlab** – perfect (maybe the best) for problems which can be formulated using vectors and matrices (LP, QP), difficult, but possible, to use for general nonlinear programming problems
- **Mathematica**
- **GAMS** – many supported problems and solvers, free version strongly limited
- MS Excel – problem size is limited (something like 200 decision variables, 100 constraints)
- SAS, R ...

Content

1 Duality in linear programming

2 Software tools for LP

3 Quadratic programming problems and Wolfe algorithm

Quadratic programming problems (QPP)

Basic form suitable for Wolfe algorithm

$$\begin{aligned} \min\ & \frac{1}{2}x^T Cx + p^T x \\ \text{s.t. } & Ax \leq b, \\ & x \geq 0. \end{aligned}$$

i.e. (vector valued constraints)

$$\begin{aligned} f(x) &= \frac{1}{2}x^T Cx + p^T x \\ g_1(x) &= Ax - b, \\ g_2(x) &= -x. \end{aligned}$$

Quadratic programming problems – examples

- Least-squares problems with restrictions (constraints) on coefficients
- Markowitz portfolio problem (variance minimization)
- Second-order approximation of more difficult nonlinear programming problems (Sequential Quadratic Programming)

Quadratic programming problems

Ass. C is positive semidefinite (symmetric)

$$\nabla_x f(x) = Cx + p,$$
$$\nabla^2_{x,x} f(x) = C.$$

KKT optimality conditions

Lagrange function

$$L(x, y, v) = \frac{1}{2}x^T Cx + p^T x + y^T(Ax - b) - v^T x,$$

where $y \geq 0,\ v \geq 0$.

KKT optimality conditions for a feasible point

$$\begin{aligned} \nabla_x L(x, y, v) = \ & Cx + p + A^T y - v = 0, \\ & y^T(Ax - b) = 0,\ \ v^T x = 0, \\ & y \geq 0, v \geq 0. \end{aligned}$$

KKT optimality conditions

Using additional slack variables $w \geq 0$ in $Ax + Iw = b$ we obtain the **linear system**

$$\begin{aligned} Cx + p + A^T y - v &= 0, \\ Ax + Iw &= b \end{aligned}$$

together with **complementarity slackness conditions**

$$v^T x = 0, \;\; w^T y = 0,$$

and **nonnegativity**

$$x, y, v, w \geq 0.$$

We can solve the linear system using the simplex algorithm and take into account the complementarity slackness conditions = **Wolfe algorithm**.

QPP – Example

$$\begin{aligned} \min\ & x_1^2 + x_2^2 + p_1 x_1 + p_2 x_2, \\ \text{s.t. } & x_1 \leq 1, x_2 \leq 1, \\ & x_1 \geq 0, x_2 \geq 0. \end{aligned}$$

$p = (0, 0)$ or $p = (-2, -2)$

Wolfe algorithm

1. Find a **primal basic solution** B of $Ax + Iw = b$.
   - If $b \geq 0$, then $B = I$.
   - Else you can use Gauss–Jordan elimination.

   If such basis does not exist, then END: the problem has no feasible solution.
2. Build a **linear programming problem** using new decision variables z:

$$\begin{aligned} \min \ & \sum_{k=1}^{n} z_k, \\ & Cx + A^T y - v + Dz = -p, \\ & Ax + Iw = b, \\ & x, y, v, w, z \geq 0, \end{aligned}$$

   where D is a diagonal matrix with
   - $d_{kk} = -1$ if $\sum_j c_{kj} x_j(B) > -p_k$,
   - $d_{kk} = +1$ if $\sum_j c_{kj} x_j(B) \leq -p_k$.

Wolfe algorithm

3. Use $L = B \cup \{z_1, \ldots, z_n\}$ as the feasible basis[1] to start the **simplex algorithm**. During the algorithm run ensure that the **complementarity slackness conditions** $v^T x = 0$, $w^T y = 0$ are fulfilled, e.g.
   - If x_k is in basis, then v_k cannot be included into basis.
   - ...
4. Denote by L_{fin} the final basis.
   - If $\sum_{j=1}^{m} y_j(L_{fin}) = 0$, then $x(L_{fin})$ is an optimal solution of QPP.
   - If $\sum_{j=1}^{m} y_j(L_{fin}) > 0$, then the algorithm has not found an optimal solution of QPP.

FINITNESS ensured under C positive semidefinite and $h(C) = h(C|p)$.

[1] We overload the symbol for the basis (sub-matrix or corresponding decision variables).

Literature

- Bazaraa, M.S., Sherali, H.D., and Shetty, C.M. (2006). Nonlinear programming: theory and algorithms, Wiley, Singapore, 3rd edition.
- Boyd, S., Vandenberghe, L. (2004). Convex Optimization, Cambridge University Press, Cambridge.
- Lachout, P. (2011). Matematické programování. Skripta k (zaniklé) přednášce Optimalizace I (IN CZECH).
- Matoušek and Gärtner (2007). Understanding and using linear programming, Springer.
- Wolfe, P. (1959). The Simplex Method for Quadratic Programming. Econometrica 27(3), 382–398.

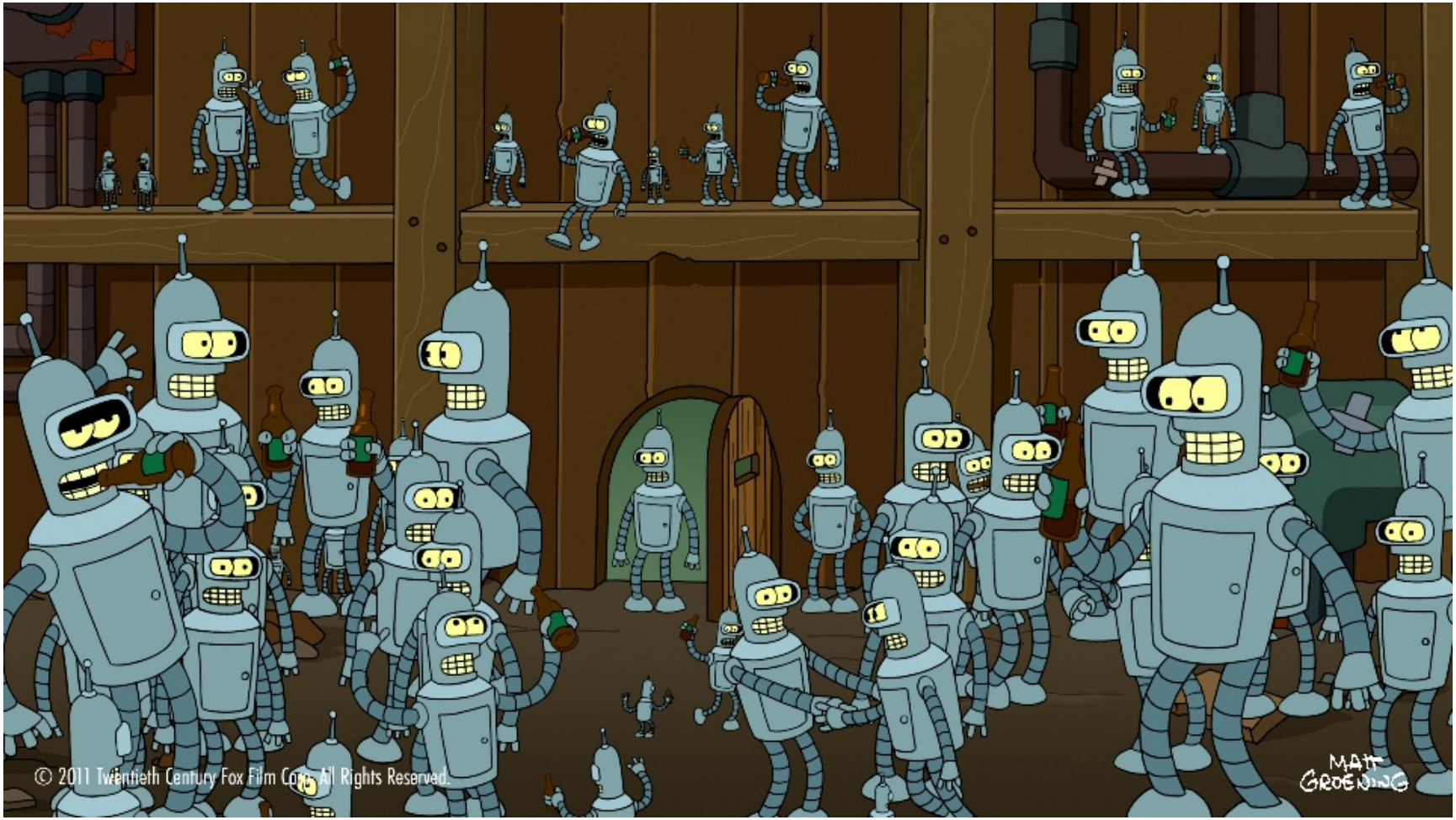
An introduction to Benders decomposition

Martin Branda

Charles University in Prague
Faculty of Mathematics and Physics
Department of Probability and Mathematical Statistics

Computational Aspects of Optimization

Benders decomposition



Benders decomposition

Benders decomposition can be used to solve:

- linear programming
- mixed-integer (non)linear programming
- two-stage stochastic programming (L-shaped algorithm)
- multistage stochastic programming (Nested Benders decomposition)

Benders decomposition for two-stage linear programming problems

$$
\begin{aligned}
\min c^T x &+ q^T y && (1)\\
\text{s.t. } Ax &= b, && (2)\\
Tx + Wy &= h, && (3)\\
x &\geq 0, && (4)\\
y &\geq 0. && (5)
\end{aligned}
$$

ASS. $\mathcal{B}_1 := \{x : \ Ax = b, x \geq 0\}$ is bounded and the problem has an optimal solution.

Benders decomposition

We define the **recourse function** (second-stage function)

$$f(x) \quad = \quad \min\{q^T y : \; Wy = h - Tx, \; y \geq 0\} \tag{6}$$

If for some x is $\{y : \; Wy = h - Tx, \; y \geq 0\} = \emptyset$, then we set $f(x) = \infty$.
The recourse function is piecewise linear, convex, and bounded below ...

Benders decomposition

Proof (outline):

- **bounded below and piecewise linear**: There are finitely many optimal basis B chosen from W such that

$$f(x) = q_B^T B^{-1}(h - Tx),$$

where feasibility $B^{-1}(h - Tx) \geq 0$ is fulfilled for $x \in \mathcal{B}_1$. Optimality condition $q_B^T B^{-1} W - q \leq 0$ does not depend on x.

Benders decomposition

Proof (outline):

- **convex**: let $x_1, x_2 \in \mathcal{B}_1$ and y_1, y_2 be such that $f(x_1) = q^T y_1$ and $f(x_2) = q^T y_2$. For arbitrary $\lambda \in (0,1)$ and $x = \lambda x_1 + (1-\lambda)x_2$ we have
$$\lambda y_1 + (1-\lambda)y_2 \in \{y : \ Wy = h - Tx, \ y \geq 0\},$$
i.e. the convex combination of y's is feasible. Thus we have
$$\begin{aligned} f(x) &= \min\{q^T y : \ Wy = h - Tx, \ y \geq 0\} && (7) \\ &\leq q^T(\lambda y_1 + (1-\lambda)y_2) = \lambda f(x_1) + (1-\lambda) f(x_2). && (8) \end{aligned}$$

A simple example

$$
\begin{aligned}
\min\ & 2x + 2y_1 + 3y_2 \\
\text{s.t.}\ & x + y_1 + 2y_2 = 3, \\
& 3x + 2y_1 - y_2 = 4, \\
& x, y_1, y_2 \geq 0.
\end{aligned}
\tag{9}
$$

Benders decomposition

We have an equivalent NLP problem

$$\begin{aligned} \min c^T x &+ f(x) && (10)\\ \text{s.t. } Ax &= b, && (11)\\ x &\geq 0. && (12) \end{aligned}$$

We solve the master problem (first-stage problem)

$$\begin{aligned} \min c^T x &+ \theta && (13)\\ \text{s.t. } Ax &= b, && (14)\\ f(x) &\leq \theta, && (15)\\ x &\geq 0. && (16) \end{aligned}$$

We would like to approximate $f(x)$ (from below) ...

Algorithm – the feasibility cut

Solve

$$
\begin{aligned}
f(\hat{x}) &= \min\{q^T y : \; Wy = h - T\hat{x}, \; y \geq 0\} && (17)\\
&= \max\{(h - T\hat{x})^T u : \; W^T u \leq q\}. && (18)
\end{aligned}
$$

If the dual problem is unbounded (primal is infeasible), then there exists a growth direction $\tilde{u}$ such that $W^T \tilde{u} \leq 0$ and $(h - T\hat{x})^T \tilde{u} > 0$. For any feasible x there exists some $y \geq 0$ such that $Wy = h - Tx$. If we multiply it by $\tilde{u}$

$$\tilde{u}^T (h - T\hat{x}) = \tilde{u}^T Wy \leq 0,$$

which has to hold for any feasible x, but is violated by $\hat{x}$. Thus by

$$\tilde{u}^T (h - Tx) \leq 0$$

the infeasible $\hat{x}$ is cut off.

Algorithm – the optimality cut

There is an optimal solution $\hat{u}$ of the dual problem such that

$$f(\hat{x}) = (h - T\hat{x})^T \hat{u}.$$

For arbitrary x we have

$$\begin{aligned} f(x) &= \sup_u \{(h - Tx)^T u : \ W^T u \leq q\}, & (19) \\ &\geq (h - Tx)^T \hat{u}. & (20) \end{aligned}$$

From inequality $f(x) \leq \theta$ we have the optimality cut

$$\hat{u}^T (h - Tx) \leq \theta.$$

If this cut is fulfilled for actual $(\hat{x}, \hat{\theta})$, then STOP, $\hat{x}$ is an optimal solution.

Algorithm – master problem

We solve the master problem with cuts

$$
\begin{aligned}
\min c^T x &+ \theta && (21)\\
\text{s.t. } Ax &= b, && (22)\\
\tilde{u}_l^T (h - Tx) &\leq 0, \; l = 1, \ldots, L && (23)\\
\tilde{u}_k^T (h - Tx) &\leq \theta, \; k = 1, \ldots, K, && (24)\\
x &\geq 0. && (25)
\end{aligned}
$$

Algorithm

0. INIC: Set $\theta = -\infty$.
1. Solve the master problem to obtain $\hat{x}$, $\hat{\theta}$.
2. For $\hat{x}$, solve the dual of the second-stage (recourse) problem to obtain a direction (feasibility cut) or an optimal solution (optimality cut).
3. STOP, if the current solution $\hat{x}$ fulfills the optimality cut. Otherwise GO TO Step 1.

(Generalization with lower and upper bounds.)

Convergence of the algorithm

Convergence of the algorithm: see Kall and Mayer (2005), Proposition 2.19.

Example

$$
\begin{aligned}
\min\ & 2x + 2y_1 + 3y_2 \\
\text{s.t.}\ & x + y_1 + 2y_2 = 3, \\
& 3x + 2y_1 - y_2 = 4, \\
& x,\ y_1,\ y_2 \geq 0.
\end{aligned}
\tag{26}
$$

Example

Recourse function

$$
\begin{aligned}
f(x) = \min\ & 2y_1 + 3y_2 \\
\text{s.t.}\ & y_1 + 2y_2 = 3 - x, \\
& 2y_1 - y_2 = 4 - 3x, \\
& y_1,\ y_2 \geq 0.
\end{aligned}
\tag{27}
$$

Iteration 1

Set $\theta = -\infty$ and solve master problem

$$\min_x 2x \text{ s.t. } x \geq 0. \tag{28}$$

Optimal solution $\hat{x} = 0$.

Iteration 1

Solve the dual problem for $\hat{x} = 0$:

$$\begin{aligned} \max_{u} \; & (3 - x)u_1 + (4 - 3x)u_2 \\ \text{s.t.} \; & u_1 + 2u_2 \leq 2, \\ & 2u_1 - u_2 \leq 3. \end{aligned} \tag{29}$$

Optimal solution is $\hat{u} = (8/5, 1/5)$ with optimal value $28/5$, thus no feasibility cut is necessary. We can construct an optimality cut

$$(3 - x)8/5 + (4 - 3x)1/5 = 28/5 - 11/5x \leq \theta.$$

Iteration 2

Add the optimality cut and solve

$$\begin{aligned} \min_{x,\theta} \; & 2x \\ \text{s.t. } & 28/5 - 11/5x \leq \theta, \\ & x \geq 0. \end{aligned} \tag{30}$$

Optimal solution $(\hat{x}, \hat{\theta}) = (2.5455, 0)$ with optimal value 5.0909.

Iteration 2

Solve the dual problem for $\hat{x} = 2.5455$:

$$\begin{aligned} \max_{u} \; & (3-x)u_1 + (4-3x)u_2 \\ \text{s.t.} \; & u_1 + 2u_2 \leq 2, \\ & 2u_1 - u_2 \leq 3. \end{aligned} \tag{31}$$

Optimal solution is $\hat{u} = (1.5, 0)$ with optimal value 0.6818, thus no feasibility cut is necessary. We can construct an optimality cut

$$(3-x)1.5 + (4-3x)0 = 4.5 - 1.5x \leq \theta.$$

Iteration 3

Add the optimality cut and solve

$$\begin{aligned} \min_{x,\theta} \; & 2x \\ \text{s.t.} \; & 28/5 - 11/5x \leq \theta, \\ & 4.5 - 1.5x \leq \theta, \\ & x \geq 0. \end{aligned} \tag{32}$$

...

Two-stage stochastic programming problem

Probabilities $0 < p_s < 1$, $\sum_s p_s = 1$,

$$
\begin{array}{rrrrrrl}
 & & & & & & \min c^T x + \sum_{s=1}^{S} p_s q_s^T y_s \\
 & & & & & & \text{s.t.} \\
 & & & & Ax & = & b, \\
Wy_1 & & & & +T_1 x & = & h_1, \\
 & Wy_2 & & & +T_2 x & = & h_2, \\
 & & \ddots & & \vdots & \vdots & \vdots \\
 & & & Wy_S & +T_S x & = & h_S, \\
 & & & & x \geq 0, & y_s \geq 0. &
\end{array}
$$

One master and S "second-stage" problems.

Minimization of Conditional Value at Risk

If the distribution of R_i is discrete with realizations r_{is} and probabilities $p_s = 1/S$, then we can use **linear programming** formulation

$$\min_{\xi, x_i} \xi + \frac{1}{(1-\alpha)S} \sum_{s=1}^{S} [-\sum_{i=1}^{n} x_i r_{is} - \xi]_+,$$

$$\text{s.t.} \sum_{i=1}^{n} x_i \overline{R}_i \geq r_0,$$

$$\sum_{i=1}^{n} x_i = 1, \; x_i \geq 0,$$

where $\overline{R}_i = 1/S \sum_{s=1}^{S} r_{is}$.

Conditional Value at Risk

Master problem

$$\min_{\xi, x_i} \xi + \frac{1}{(1-\alpha)S} \sum_{s=1}^{S} f_s(x, \xi),$$

$$\text{s.t.} \sum_{i=1}^{n} x_i \overline{R}_i \geq r_0, \ \sum_{i=1}^{n} x_i = 1, \ x_i \geq 0,$$

Second-stage problems

$$f_s(x, \xi) = \min_y y,$$

$$\text{s.t. } y \geq -\sum_{i=1}^{n} x_i r_{is} - \xi,$$

$$y \geq 0.$$

Solve the dual problem quickly ..

Literature

- L. Adam: Nelinearity v úlohách stochastického programování: aplikace na řízení portfolia. Diplomová práce MFF UK, 2011. (IN CZECH)
- P. Kall, J. Mayer: Stochastic Linear Programming: Models, Theory, and Computation. Springer, 2005.

Introduction to integer linear programming

Martin Branda

Charles University in Prague
Faculty of Mathematics and Physics
Department of Probability and Mathematical Statistics

Computational Aspects of Optimization

Knapsack problem

Values $a_1 = 4$, $a_2 = 6$, $a_3 = 9$, costs $c_1 = 4$, $c_2 = 6$, $c_3 = 11$, budget $b = 10$:

$$\begin{aligned} \max \ & \sum_{i=1}^{3} c_i x_i \\ \text{s.t.} \ & \sum_{i=1}^{3} a_i x_i \leq 10, \\ & x_i \in \{0, 1\}. \end{aligned}$$

Consider $=$ instead of $\leq$, or $0 \leq x_i \leq 1$ instead of $x_i \in \{0, 1\}$ ...

Why is integrality so important?

Real (mixed-)integer programming problems (not always linear)

- **Portfolio optimization** – integer number of assets, fixed transaction costs
- **Scheduling** – integer (binary) decision variables to assign a job to a machine
- **Vehicle Routing Problems** (VRP) – binary decision variables which identify a successor of a node on the route
- ...

In general – modelling of **logical relations**, e.g.

- at least two constraints from three are fulfilled,
- if we buy this asset than the fixed transaction costs increase,
- ...

Integer linear programming

$$\min c^T x \tag{1}$$
$$Ax \geq b, \tag{2}$$
$$x \in \mathbb{Z}^n_+. \tag{3}$$

Assumption: all coefficients are integer (rational before multiplying by a proper constant).

Set of feasible solution and its relaxation

$$S = \{x \in \mathbb{Z}^n_+ : Ax \geq b\}, \tag{4}$$
$$P = \{x \in \mathbb{R}^n_+ : Ax \geq b\} \tag{5}$$

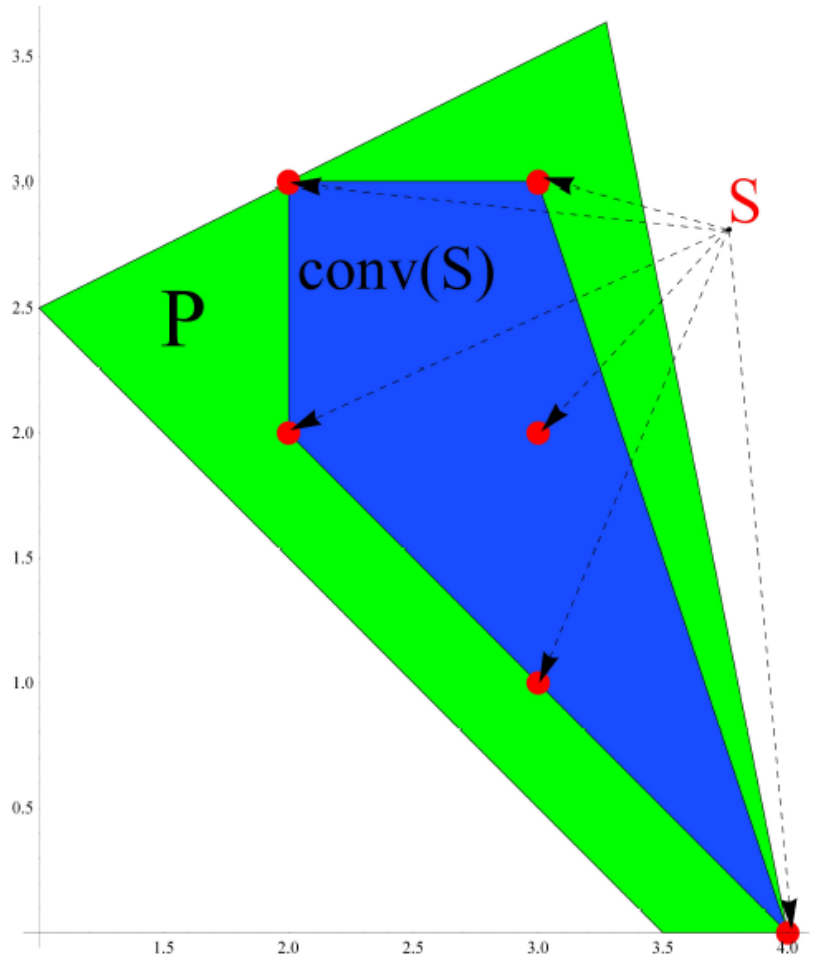
Obviously $S \subseteq P$. Not so trivial that $S \subseteq \mathrm{conv}(S) \subseteq P$.

Example

Consider set S given by

$$
\begin{aligned}
x_1 - 2x_2 &\geq -4, && (6)\\
-5x_1 - x_2 &\geq -20, && (7)\\
2x_1 + 2x_2 &\geq 7, && (8)\\
x_1, x_2 &\in \mathbb{Z}_+. && (9)
\end{aligned}
$$

Set of feasible solutions, its relaxation and convex envelope



Škoda (2010)

Integer linear programming problem

Problem

$$\min c^T x : \; x \in S. \tag{10}$$

is equivalent to

$$\min c^T x : \; x \in \mathrm{conv}(S). \tag{11}$$

$\mathrm{conv}(S)$ is very difficult to construct – many constraints ("strong cuts") are necessary (there are some exceptions).

LP-relaxation:

$$\min c^T x : \; x \in P. \tag{12}$$

Mixed-integer linear programming

Often both integer and continuous decision variable appear:

$$\begin{aligned} \min\ & c^T x + d^T y \\ \text{s.t.}\ & Ax + By \geq b \\ & x \in \mathbb{Z}_+^n,\ y \in \mathbb{R}_+^{n'}. \end{aligned}$$

(WE DO NOT CONSIDER IN INTRODUCTION)

Basic algorithms

We consider:

- **Cutting Plane Method**
- **Branch-and-Bound**

There are methods combining previous alg., e.g. Branch-and-Cut.

Cutting plane method – Gomory cuts

1. Solve LP-relaxation using (primal or dual) SIMPLEX algorithm.
   - If the solution is integral – END, we have found an optimal solution,
   - otherwise continue with the next step.
2. Add a **Gomory cut** (...) and solve the resulting problem using DUAL SIMPLEX alg.

Example

$$\min 4x_1 + 5x_2 \tag{13}$$
$$x_1 + 4x_2 \geq 5, \tag{14}$$
$$3x_1 + 2x_2 \geq 7, \tag{15}$$
$$x_1, x_2 \in \mathbb{Z}_+^n. \tag{16}$$

Dual simplex for LP-relaxation ...

After two iterations of the dual SIMPLEX algorithm ...

			4	5	0	0
			x_1	x_2	x_3	x_4
5	x_2	8/10	0	1	-3/10	1/10
4	x_1	18/10	1	0	2/10	-4/10
		112/10	0	0	-7/10	-11/10

Gomory cuts

There is a row in simplex table, which corresponds to a non-integral solution x_i in the form:

$$x_i + \sum_{j \in N} w_{ij} x_j = d_i, \tag{17}$$

where N denotes the set of non-basic variables; d_i is non-integral. We denote

$$w_{ij} = \lfloor w_{ij} \rfloor + f_{ij}, \tag{18}$$

$$d_i = \lfloor d_i \rfloor + f_i, \tag{19}$$

i.e. $0 \leq f_{ij}, f_i < 1$.

$$\sum_{j \in N} f_{ij} x_j \geq f_i, \tag{20}$$

or rather $-\sum_{j \in N} f_{ij} x_j + s = -f_i$, $s \geq 0$.

Gomory cuts

General properties of cuts (including Gomory ones):

- Property 1: Current (non-integral) solution becomes infeasible (it is cut).
- Property 2: No feasible integral solution becomes infeasible (it is not cut).

Gomory cuts – property 1

We express the constraints in the form

$$x_i + \sum_{j \in N} (\lfloor w_{ij} \rfloor + f_{ij}) x_j = \lfloor d_i \rfloor + f_i, \tag{21}$$

$$x_i + \sum_{j \in N} \lfloor w_{ij} \rfloor x_j - \lfloor d_i \rfloor = f_i - \sum_{j \in N} f_{ij} x_j. \tag{22}$$

Current solution $x_j^* = 0$ pro $j \in N$ a $x_i^* = d_i$ is non-integral, i.e. $0 < x_i^* - \lfloor d_i \rfloor < 1$, thus

$$0 < x_i^* - \lfloor d_i \rfloor = f_i - \sum_{j \in N} f_{ij} x_j^* \tag{23}$$

and

$$\sum_{j \in N} f_{ij} x_j^* < f_i, \tag{24}$$

which is a contradiction with Gomory cut.

Gomory cuts – property 2

Consider an arbitrary integral feasible solution and rewrite the constraint as

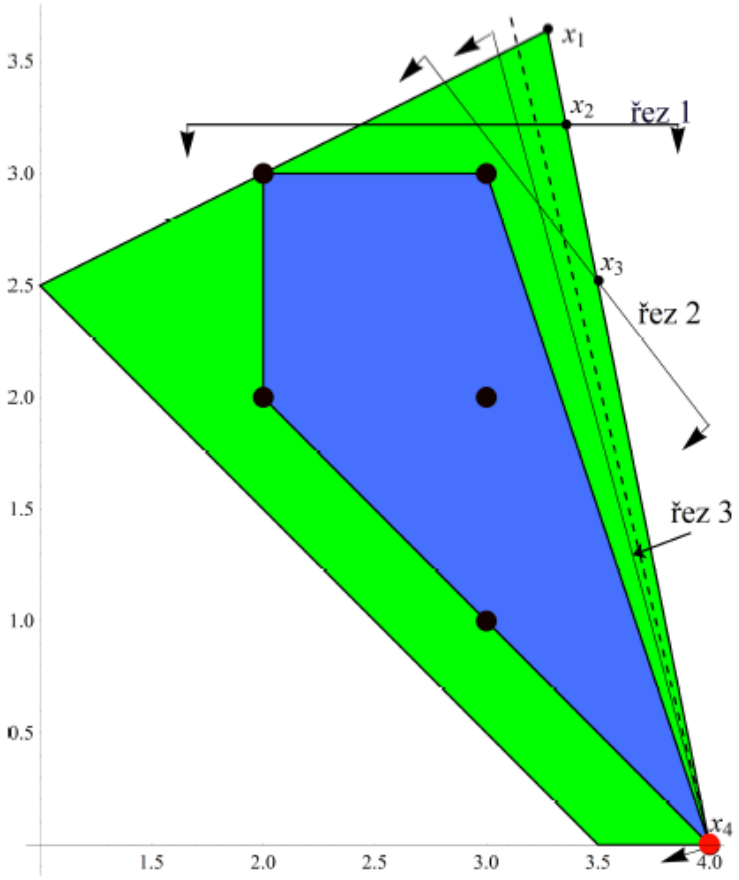
$$x_i + \sum_{j \in N} \lfloor w_{ij} \rfloor x_j - \lfloor d_i \rfloor \quad = \quad f_i - \sum_{j \in N} f_{ij} x_j, \tag{25}$$

Left-hand side (LS) is integral, thus right-hand side (RS) is integral. Moreover, $f_i < 1$ a $\sum_{j \in N} f_{ij} x_j \geq 0$, thus RS is strictly lower than 1 and at the same time it is integral, thus lower or equal to 0, i.e. we obtain Gomory cut

$$f_i - \sum_{j \in N} f_{ij} x_j \leq 0. \tag{26}$$

Thus each integral solution fulfills it.

Cutting plane methods – steps



Škoda (2010)

Dantzig cuts

$$\sum_{j \in N} x_j \geq 1. \tag{27}$$

(Remind that non-basic variables are equal to zero.)

After two iterations of the dual SIMPLEX algorithm ...

			4	5	0	0
			x_1	x_2	x_3	x_4
5	x_2	8/10	0	1	-3/10	1/10
4	x_1	18/10	1	0	2/10	-4/10
		112/10	0	0	-7/10	-11/10

For example, x_1 is not integral:

$$\begin{aligned} x_1 + 2/10x_3 - 4/10x_4 &= 18/10, \\ x_1 + (0 + 2/10)x_3 + (-1 + 6/10)x_4 &= 1 + 8/10. \end{aligned}$$

Gomory cut:

$$2/10x_3 + 6/10x_4 \geq 8/10.$$

New simplex table

			4	5	0	0	0
			x_1	x_2	x_3	x_4	x_5
5	x_2	8/10	0	1	-3/10	1/10	0
4	x_1	18/10	1	0	2/10	-4/10	0
0	x_5	-8/10	0	0	- 2/10	-6/10	1
		112/10	0	0	-7/10	-11/10	0

Dual simplex alg. ...

Branch-and-Bound

General principles:

- Solve LP problem without integrality only.
- Branch using additional constraints on integrality: $x_i \leq \lfloor x_i^* \rfloor$, $x_i \geq \lfloor x_i^* \rfloor + 1$.
- Cut inperspective branches before solving (using bounds on the optimal value).

Branch-and-Bound

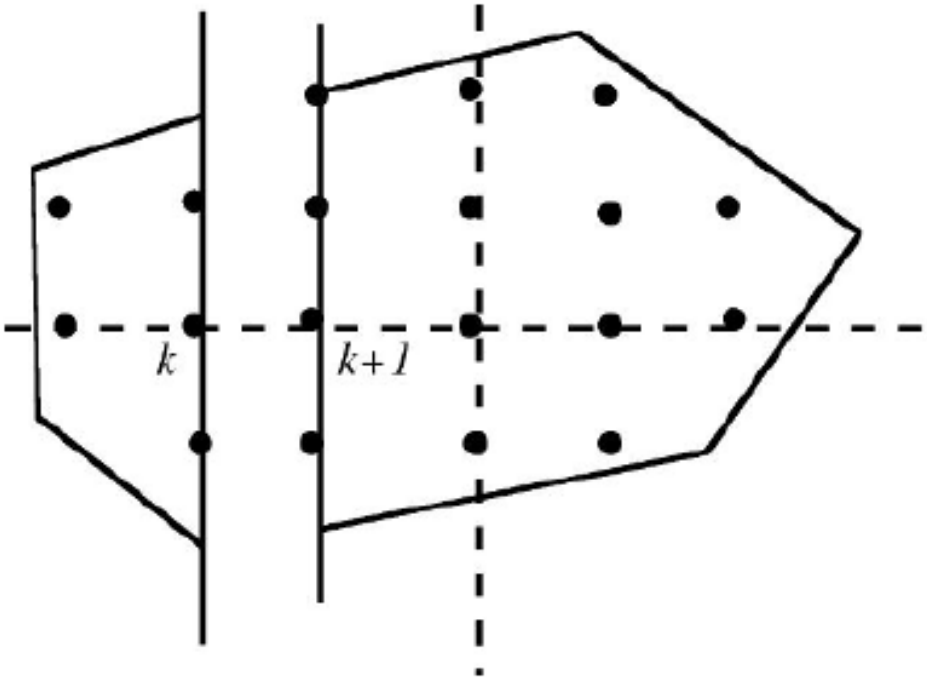
General principles:

- Solve only LP problems with relaxed integrality.
- **Branching**: if an optimal solution is not integral, e.g. $\hat{x}_i$, create and save two new problems with constraints $x_i \leq \lfloor \hat{x}_i \rfloor$, $x_i \geq \lceil \hat{x}_i \rceil$.
- **Bounding** ("different" cutting): save the objective value of the best integral solution and cut all problems in the queue created from the problems with higher optimal values[1].

Exact algorithm ..

[1] Branching cannot improve it.

Branch-and-Bound



P. Pedegral (2004). Introduction to optimization, Springer-Verlag, New York.

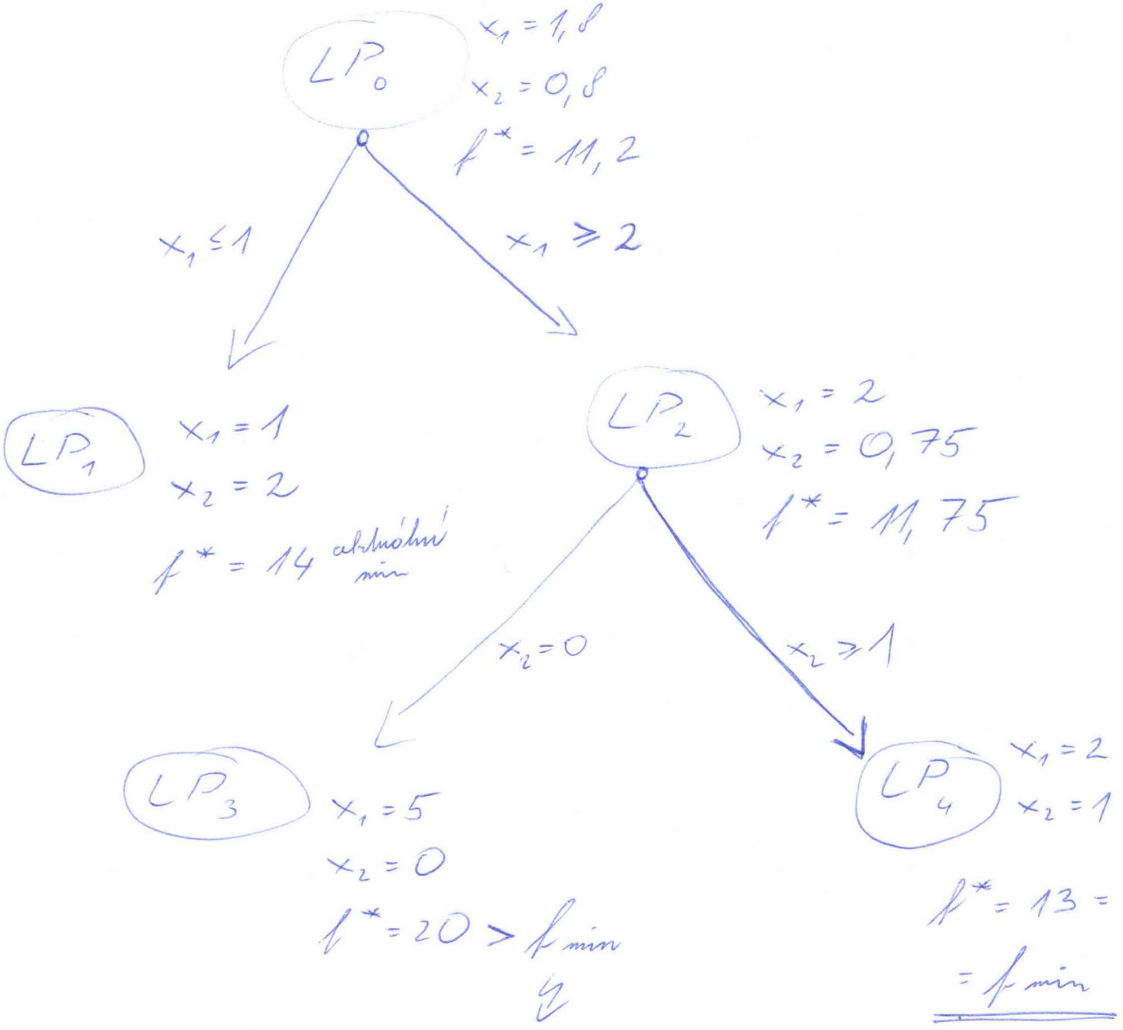
Branch-and-Bound

0. $f_{min} = \infty$, $x_{min} = \cdot$, list of problems $P = \emptyset$
   Solve LP-relaxed problem and obtain f^*, x^*. If the solution is integral, STOP. If the problem is infeasible or unbounded, STOP.
1. **BRANCHING**: There is x_i basic non-integral variable such that $k < x_i < k + 1$ for some $k \in \mathbb{N}$:
   - Add constraint $x_i \leq k$ to previous problem and put it into list P.
   - Add constraint $x_i \geq k + 1$ to previous problem and put it into list P.
2. Take problem from P and solve it: f^*, x^*.
3.
   - If $f^* < f_{min}$ and x^* is non-integral, GO TO 1.
   - **BOUNDING**: If $f^* < f_{min}$ a x^* is integral, set $f_{min} = f^*$ a $x_{min} = x^*$, GO TO 4.
   - **BOUNDING**: If $f^* \geq f_{min}$, GO TO 4.
   - Problem is infeasible, GO TO 4.
4.
   - If $P \neq \emptyset$, GO TO 2.
   - If $P = \emptyset$ a $f_{min} = \infty$, integral solution does not exist.
   - If $P = \emptyset$ a $f_{min} < \infty$, optimal value and solution are f_{min}, x_{min}.

Better ...

2./3. Take problem from list P and solve it: f^*, x^*. If for the optimal value of the current problem holds $f^* \geq f_{min}$, then the branching is not necessary, since by solving the problems with added branching constraints we can only increase the optimal value and obtain the same f_{min}.

Branch-and-Bound



Branch-and-Bound

Algorithmic issues:

- **Problem selection from list** P: FIFO/LIFO/problem with the smallest f^*.
- **Selection of the branching variable** x_i^*: the highest/smallest violation of integrality OR the highest/smallest coefficient in the objective function.

Totally unimodular matrix

Totally unimodular matrix A: for arbitrary INTEGRAL right-hand side vector b we obtain an integral solution, e.g. transportation problem.

Algorithms – a remark

(Relative) difference between a lower and upper bound – construct the upper bound (for minimization) using a feasible solution, lower bound ?

Duality

Set $S(b) = \{x \in \mathbb{Z}_+^n : \ Ax = b\}$ and define the **value function**

$$z(b) = \min_{x \in S(b)} c^T x. \tag{28}$$

A dual function $F : \mathbb{R}^m \to \overline{\mathbb{R}}$

$$F(b) \leq z(b), \ \forall b \in \mathbb{R}^m. \tag{29}$$

A general form of **dual problem**

$$\max_F \{F(b) : \ \text{s.t. } F(b) \leq z(b), \ b \in \mathbb{R}^m, \ F : \mathbb{R}^m \to \mathbb{R}\}. \tag{30}$$

We call F a **weak dual** function if it is feasible, and **strong dual** if moreover $F(b) = z(b)$.

Duality

A function F is **subadditive** over a domain Θ if

$$F(\theta_1 + \theta_2) \leq F(\theta_1) + F(\theta_2)$$

for all $\theta_1 + \theta_2, \theta_1, \theta_2 \in \Theta$.

The value function z is subadditive over $\{b : S(b) \neq \emptyset\}$, since the sum of optimal x's is feasible for the problem with $b_1 + b_2$ r.h.s., i.e. $\hat{x}_1 + \hat{x}_2 \in S(b_1 + b_2)$.

Duality

If F is subadditive, then condition $F(Ax) \leq c^T x$ for $x \in \mathbb{Z}_+^n$ is equivalent to $F(a_{\cdot j}) \leq c_j$, $j = 1, \ldots, m$.

This is true since $F(Ae_j) \leq c^T e_j$ is the same as $F(a_{\cdot j}) \leq c_j$.

On the other hand, if F is subadditive and $F(a_{\cdot j}) \leq c_j$, $j = 1, \ldots, m$ imply

$$F(Ax) \leq \sum_{j=1}^{m} F(a_{\cdot j}) x_j \leq \sum_{j=1}^{m} c_j x_j = c^T x.$$

Duality

If we set

$$\Gamma^m = \{F : \mathbb{R}^m \to \mathbb{R},\ F(0) = 0,\ F \text{ subadditive}\},$$

then we can write a **subadditive dual** independent of x:

$$\max_F \{F(b) :\ \text{s.t. } F(a_{.j}) \leq c_j,\ F \in \Gamma^m\}. \tag{31}$$

Weak and strong duality holds.

An easy feasible solution based on LP duality (= weak dual)

$$F_{LP}(b) = \max_y b^T y \text{ s.t. } A^T y \leq c. \tag{32}$$

Duality

Complementary slackness condition: if $\hat{x}$ is an optimal solution for IP, and $\hat{F}$ is an optimal subadditive dual solution, then

$$(\hat{F}(a_{\dots j}) - c_j)\hat{x}_j = 0, \; j = 1, \dots, m.$$

Software

... even for nonlinear integer problems

- **Interfaces**: GAMS, CPlex Studio, Gurobi, ...
- **Solvers**: CPlex (MILP, MIQP), Gurobi (MILP, MIQP), Baron, Bonmin (MINLP), Dicopt (MINLP), Knitro (MINLP), Lindo, ...

For difficult problems usually **heuristic and meta-heuristic algorithms** (greedy h., genetic alg., tabu search, simulated annealing, ...)

GAMS

Integer variables

- **Integer variables** – nonnegative with predefined upper bound 100 (can be changed using *x.up(i) = 1000;*)!
- **Binary variables**

Command SOLVE using

- MILP
- MIQCP
- MINLP

GAMS – options

TOLERANCE for optimal value of the integer problems:

- **optcr** – relative tolerance (default value 0.1 – usually too high)
- **optca** – absolute tolerance (turned off)
- **reslim** – maximal running time in seconds (default value 1000 – usually too low)
- **nlp** = *conopt* , **lp** = *gurobi* , **mip** = *cplex* – solver selection in code

For example
OPTIONS optcr=0.000001 reslim = 3600;

Literature

- G.L. Nemhauser, L.A. Wolsey (1989). Integer Programming. Chapter VI in Handbooks in OR & MS, Vol. 1, G.L. Nemhauser et al. Eds.
- P. Pedegral (2004). Introduction to optimization, Springer-Verlag, New York.
- Š. Škoda: Řešení lineárních úloh s celočíselnými omezeními v GAMSu. Bc. práce MFF UK, 2010. (In Czech)
- L.A. Wolsey (1998). Integer Programming. Wiley, New York.
- L.A. Wolsey, G.L. Nemhauser (1999). Integer and Combinatorial Optimization. Wiley, New York.

Introduction to computational complexity

Martin Branda

Charles University in Prague
Faculty of Mathematics and Physics
Department of Probability and Mathematical Statistics

Computational Aspects of Optimization

Software

General mathematical and statistical software:

- **Matlab** and (free) OPTI toolbox: http://www.i2c2.aut.ac.nz/Wiki/OPTI/index.php/Main/HomePage
- **SAS**: OR package
- **R**: suitable packages
- **Mathematica**: findminimum, minimize, ...
- ...

Software

Modelling tools for optimization:

- **GAMS**: http://www.gams.com
- **AIMMS**: http://www.aimms.com
- **Gurobi**: http://www.gurobi.com
- **AMPL**: http://ampl.com
- **CPlex Studio**: http://www-03.ibm.com/software/products/cs/ibmilogcpleoptistud
- **MPL**: http://www.maximalsoftware.com/mpl
- ...

Software

Open Source libraries

- **COIN-OR**: http://www.coin-or.org/

Software

Solvers

- Baron (LP, NLP, MIP, MINLP, ...)
- Bonmin (NLP, MIP, MINLP, ...)
- Conopt (LP, NLP, ...)
- CPlex (LP, MIP, MIQCP, ...)
- Dicopt (MIQCP, MINLP, ...)
- Gurobi
- Knitro (LP, MINLP, MIQCP, NLP, ...)
- Lindo (LP, MINLP, MIP, MIQCP, NLP, ...)
- Minos (LP, NLP, ...)
- Mosek (LP, MIP, MIQCP, NLP, ...)
- Xpress (LP, MIP, MIQCP, ...)
- ...

Introduction to complexity theory

Wolsey (1998): Consider **decision problems** having YES–NO answers.
Optimization problem

$$\max_{x \in M} c^T x$$

can be replaced by (for some k integral)

$$\text{Is there an } x \in M \text{ with value } c^T x \geq k?$$

For a problem instance X, the **length of the input** $L(X)$ is the length of the binary representation of a standard representation of the instance.
Instance $X = \{c, M\}$, $X = \{c, M, k\}$

Example: Knapsack decision problem

For an instance

$$X = \left\{ \sum_{i=1}^{n} c_i x_i \geq k,\ \sum_{i=1}^{n} a_i x_i \leq b,\ x \in \{0,1\}^n \right\},$$

the length of the input is

$$L(X) = \sum_{i=1}^{n} \lceil \log c_i \rceil + \sum_{i=1}^{n} \lceil \log a_i \rceil + \lceil \log b \rceil + \lceil \log k \rceil$$

Running time

Definition

- $f_A(X)$ is the **number of elementary calculations** required to run the algorithm A on the instance $X \in P$.
- **Running time of the algorithm** A

$$f_A^*(I) = \sup_X \{f_A(X) : \ L(X) = I\}.$$

- An algorithm A is **polynomial** for a problem P if $f_A^*(I) = O(I^p)$ for some $p \in \mathbb{N}$.

Classes $\mathcal{NP}$ and $\mathcal{P}$

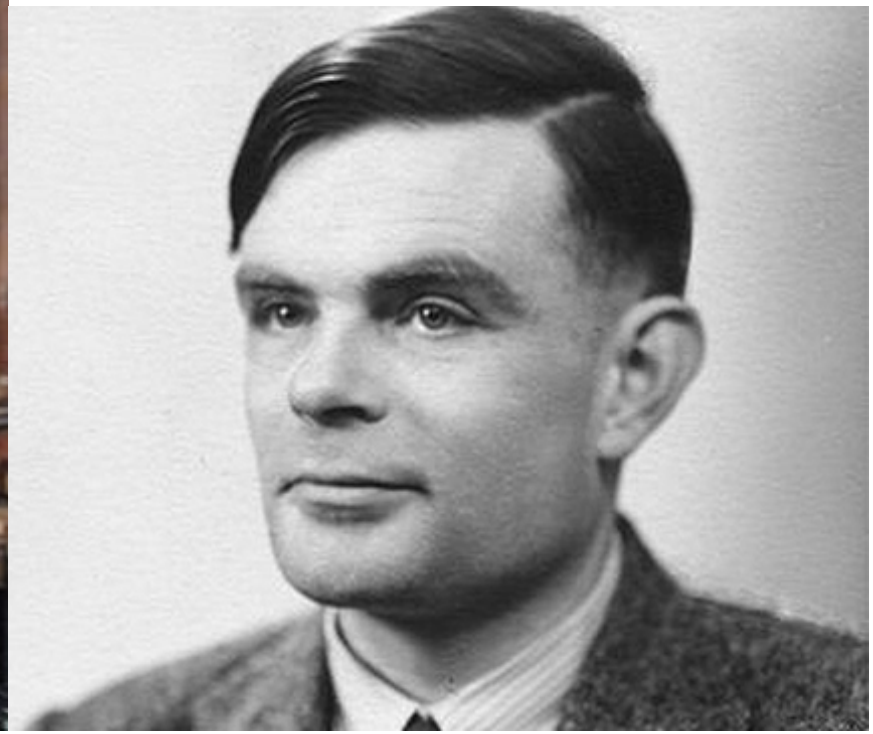
Definition

- $\mathcal{NP}$ (Nondeterministic Polynomial) is the class of decision problems with the property that: for any instance for which the answer is YES, there is a polynomial proof of the YES.
- $\mathcal{P}$ is the class of decision problems in $\mathcal{NP}$ for which there exists a polynomial algorithm.

$\mathcal{NP}$ may be equivalently defined as the set of decision problems that can be solved in polynomial time on a non-deterministic Turing machine[1].

[1] NTM writes symbols one at a time on an endless tape by strictly following a set of rules. It determines what action it should perform next according to its internal state and what symbol it currently sees. It may have a set of rules that prescribes more than one action for a given situation. The machine "branches" into many copies, each of which follows one of the possible transitions leading to a "computation tree".

Alan Turing



The Imitation Game (2014)

Polynomial reduction and the class $\mathcal{NPC}$

Definition

- If problems $P, Q \in \mathcal{NP}$, and if an instance of P can be converted in polynomial time to an instance of Q, then P is **polynomially reducible** to Q.
- $\mathcal{NPC}$, the class of $\mathcal{NP}$-**complete** problems, is the subset of problems $P \in \mathcal{NP}$ such that for all $Q \in \mathcal{NP}$, Q is polynomially reducible to P.

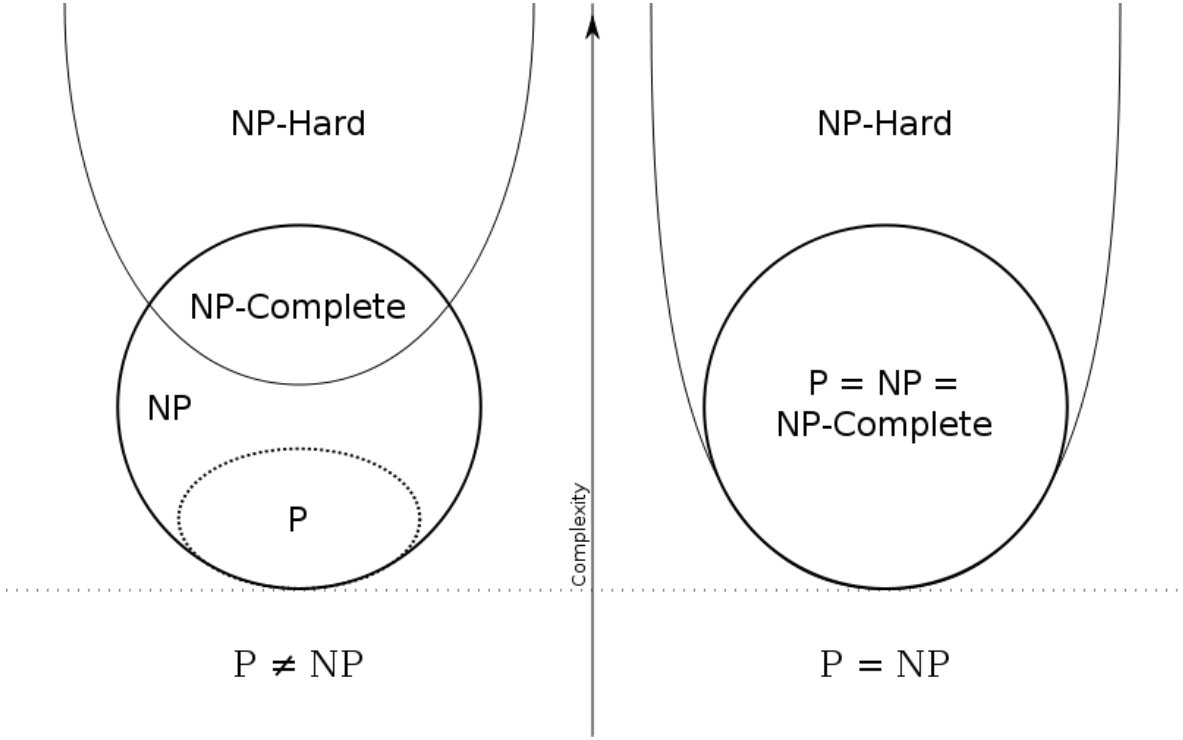
Proposition: Suppose that problems $P, Q \in \mathcal{NP}$.

- If $Q \in \mathcal{P}$ and P is polynomially reducible to Q, then $P \in \mathcal{P}$.
- If $P \in \mathcal{NPC}$ and P is polynomially reducible to Q, then $Q \in \mathcal{NPC}$.

Proposition: If $\mathcal{P} \cap \mathcal{NPC} \neq \emptyset$, then $\mathcal{P} = \mathcal{NPC}$.

Open question & Euler diagram

Is $\mathcal{P} = \mathcal{NP}$?



$\mathcal{NP}$-hard optimization problems

Definition

An optimization problem for which the decision problem lies in $\mathcal{NPC}$ is called $\mathcal{NP}$-hard.

Literature

- G.L. Nemhauser, L.A. Wolsey (1989). Integer Programming. Chapter VI in Handbooks in OR & MS, Vol. 1, G.L. Nemhauser et al. Eds.
- L.A. Wolsey (1998). Integer Programming. Wiley, New York.
- L.A. Wolsey, G.L. Nemhauser (1999). Integer and Combinatorial Optimization. Wiley, New York.

Introduction into Vehicle Routing Problems and other basic mixed-integer problems

Martin Branda

Charles University in Prague
Faculty of Mathematics and Physics
Department of Probability and Mathematical Statistics

Computational Aspects of Optimization

Totally unimodular matrices

Definition

A matrix A is totally unimodular (TU) iff every square submatrix of A has determinant $+1$, -1, or 0.

The linear program has an integral optimal solution for all integer r.h.s. b if and only if A is TU.

Totally unimodular matrices

A set of **sufficient conditions**:

- $a_{ij} \in \{-1, 0, 1\}$ for all i, j
- Each column contains at most two nonzero coefficients, i.e. $\sum_{i=1}^{m} |a_{ij}| \leq 2$,
- There exists a partitioning $M_1 \cap M_2 = \emptyset$ of the rows $1, \ldots, m$ such that each column j containing two nonzero coefficients satisfies

$$\sum_{i \in M_1} a_{ij} = \sum_{i \in M_2} a_{ij}.$$

If A is TU, then A^T and $(A|I)$ are TU.

Minimum cost network flow problem

- $G = (V, A)$ – graph with vertices V and (oriented) arcs A
- h_{ij} – arc capacity
- c_{ij} – flow cost
- b_i – demand, ASS. $\sum_i b_i = 0$
- $V^+(i) = \{k : \ (i,k) \in A\}$ – successors of i
- $V^-(i) = \{k : \ (k,i) \in A\}$ – predecessors of i

$$
\begin{aligned}
\min_{x_{ij}} \quad & \sum_{(i,j)\in A} c_{ij} x_{ij} \\
\text{s.t.} \quad & \sum_{k\in V^+(i)} x_{ik} - \sum_{k\in V^-(i)} x_{ki} = b_i, \ \ i \in V, \\
& 0 \le x_{ij} \le h_{ij}, \ \ (i,j) \in A.
\end{aligned}
$$

Wolsey (1998), Ex. 3.1 ($M_1 = \{1, \ldots, m\}, M_2 = \emptyset$)

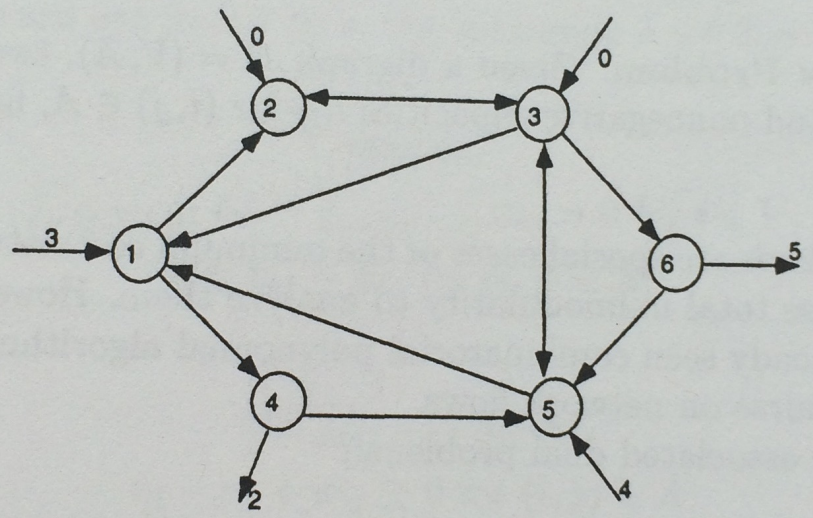


Fig. 3.1 Digraph for minimum cost network flow

equations:

x_{12}	x_{14}	x_{23}	x_{31}	x_{32}	x_{35}	x_{36}	x_{45}	x_{51}	x_{53}	x_{65}		
1	1	0	−1	0	0	0	0	−1	0	0	=	3
−1	0	1	0	−1	0	0	0	0	0	0	=	0
0	0	−1	1	1	1	1	0	0	−1	0	=	0
0	−1	0	0	0	0	0	1	0	0	0	=	−2
0	0	0	0	0	−1	0	−1	1	1	−1	=	4
0	0	0	0	0	0	−1	0	0	0	1	=	−5

Special cases

- The shortest path problem
- The transportation problem

Traveling salesman problem

- n town and in one of them there is a traveling salesman.
- Traveling salesman must visit all towns and return back.
- For each pair of towns he/she knows the traveling costs and he is looking for the cheapest route.

= Finding a Hamilton cycle in a graph with edge prices.

Assignment problem

$$\min \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} \tag{1}$$

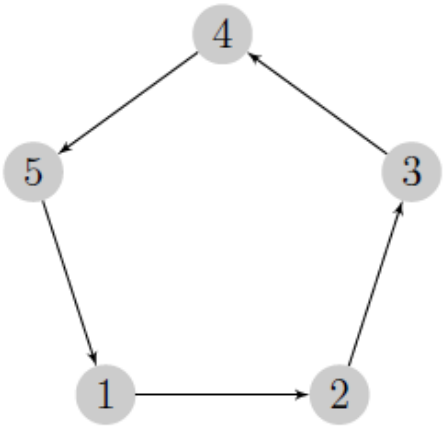
$$\sum_{i=1}^{n} x_{ij} = 1, \; j = 1, \ldots, n, \tag{2}$$

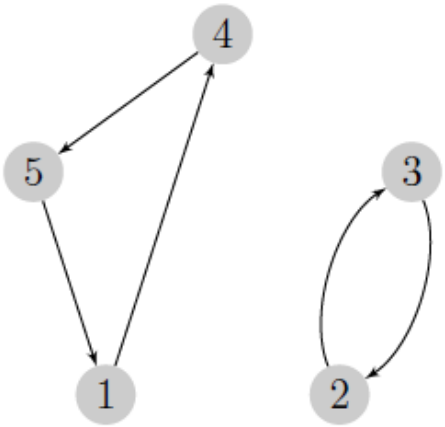
$$\sum_{j=1}^{n} x_{ij} = 1, \; i = 1, \ldots, n, \tag{3}$$

$$x_{ij} \in \{0, 1\}. \tag{4}$$

We minimize the traveling costs, we arrive to j from exactly one i, we leave i to exactly one j.

Example – 5 towns – cycle and subcycles (subroute)





Kafka (2013)

Subroute elimination conditions I

- $x_{ii} = 0,\ c_{ii} = \infty$
- $x_{ij} + x_{ji} \leq 1$
- $x_{ij} + x_{jk} + x_{ki} \leq 2$
- $\ldots$
- $\sum_{i \in S} \sum_{j \in S} x_{ij} \leq |S| - 1,\ S \subseteq \{1, \ldots, n\},\ 2 \leq |S| \leq n - 1$

Approximately 2^n inequalities, it is possible to reduce to $|S| \leq \lceil n/2 \rceil$.

Subroute elimination conditions II

$$u_i - u_j + nx_{ij} \leq n - 1, i, j = 2, \ldots, n$$

Eliminate subroutes: There is at least one route which does not go through vertex 1, denote this route by C and the number of edges by $|E(C)|$. If we sum these inequalities over all edges $\{i, j\}$, which are in C, i.e. the corresponding variables $x_{ij} = 1$, we obtain

$$n|E(C)| \leq (n-1)|E(C)|, \tag{5}$$

which is a contradiction.

Subroute elimination conditions III

$$u_i - u_j + nx_{ij} \leq n - 1, i, j = 2, \ldots, n$$

Hamilton cycle is feasible: let the vertices be ordered as $v_1 = 1$, v_2, ..., v_n. We set $u_i = l$, if $v_l = i$, i.e. u_i represent the order. For each edge of the cycle $\{i, j\}$ it holds $u_i - u_j = -1$, i.e.

$$u_i - u_j + nx_{ij} = -1 + n \leq n - 1. \tag{6}$$

For edges, which are not in the cycle, the inequality holds too: $u_i - u_j \leq n - 1$ a $x_{ij} = 0$.

Traveling Salesman Problem with Time Windows

- t_i – time when customer i is visited
- T_{ij} – time necessary to reach j from i
- l_i, u_i – lower and upper bound (time window) for visiting customer i
- M – large constant

$$\min_{x_{ij}, t_i} \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} \tag{7}$$

$$\sum_{i=1}^{n} x_{ij} = 1, \; j = 1, \ldots, n, \tag{8}$$

$$\sum_{j=1}^{n} x_{ij} = 1, \; i = 1, \ldots, n, \tag{9}$$

$$t_i + T_{ij} - t_j \leq M(1 - x_{ij}) \; i, j = 1, \ldots, n, \tag{10}$$

$$l_i \leq t_i \leq u_i, \; i = 1, \ldots, n, \tag{11}$$

$$x_{ij} \in \{0, 1\}.$$

Capacitated Vehicle Routing Problem

Parameters

- n – number of customers
- 0 – depo (starting and finishing point of each vehicle)
- K – number of vehicles (homogeneous)
- $d_j \geq 0$ – customer demand, for depo $d_0 = 0$
- $Q > 0$ – vehicle capacity ($KQ \geq \sum_{j=1}^{n} d_j$)
- c_{ij} – transportation costs from i to j (usually $c_{ii} = 0$)

Decision variables

- x_{ij} – equal to 1, if j follows after i on the route, 0 otherwise
- u_j – upper bound on transported amount after visiting customer j

Capacitated Vehicle Routing Problem

$$\min_{x_{ij},u_i} \sum_{i=0}^{n} \sum_{j=0}^{n} c_{ij} x_{ij} \tag{12}$$

$$\sum_{i=0}^{n} x_{ij} = 1, \; j = 1, \ldots, n, \tag{13}$$

$$\sum_{j=0}^{n} x_{ij} = 1, \; i = 1, \ldots, n, \tag{14}$$

$$\sum_{i=1}^{n} x_{i0} = K, \tag{15}$$

$$\sum_{j=1}^{n} x_{0j} = K, \tag{16}$$

$$u_i - u_j + d_j \leq Q(1 - x_{ij}) \; i, j = 1, \ldots, n, \tag{17}$$

$$d_i \leq u_i \leq Q, \; i = 1, \ldots, n, \tag{18}$$

$$x_{ij} \in \{0, 1\}.$$

Capacitated Vehicle Routing Problem

(12) minimization of transportation costs

(13) exactly one vehicle arrives to customer j

(14) exactly one vehicle leaves customer i

(15) exactly K vehicles return to depot 0

(16) exactly K vehicles leave depot 0

(17) balance conditions of transported amount (subroute elimination conditions)

(18) bounds on the vehicle capacity

(All vehicles are employed.)

Basic heuristics for VRP

Insertion heuristic:

- Start with empty routes.
- FOR all customers DO: Insert the customer to the place in a route where it causes the lowest increase of the traveled distance.

Clustering:

- Cluster the customers according to their geographic positions ("angles").
- Solve[1] the traveling salesman problem in each cluster.

Possible difficulties: time windows, vehicle capacities, ...

[1] ..exactly, if the clusters are not large.

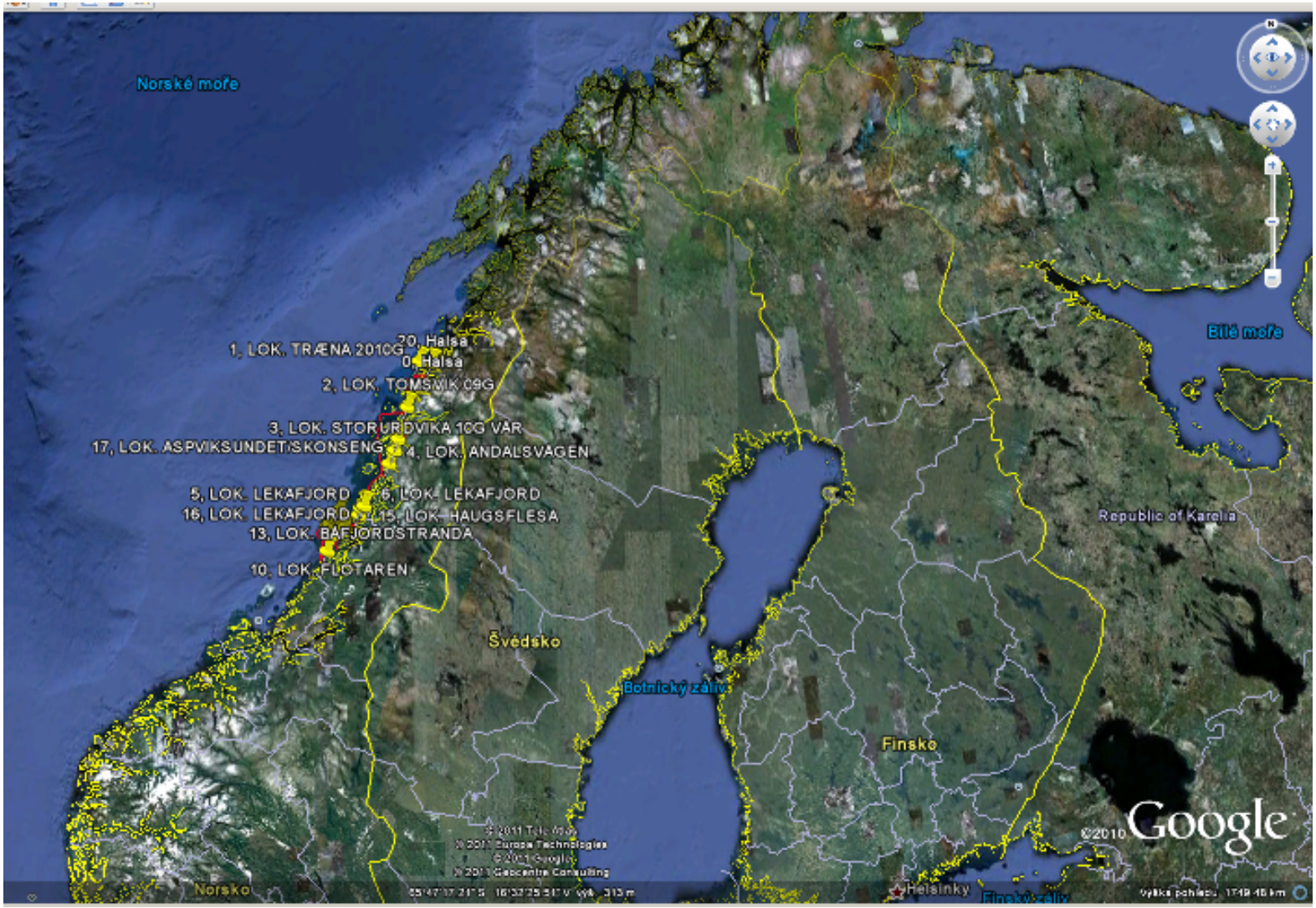
Tabu search for VRP

For a given number of iteration, run the following steps:

- Find the best solution in a *neighborhood* of the current solution. Such solution can be worse than the current one or even infeasible (use penalty function).
- Forbid moving back for a random number of steps by actualizing the **tabu list**.
- Remember the best solution.

The tabu search algorithm enables moving from local solutions (compared with a simple "hill climbing alg.").

Norway ...


Norské moře
1, LOK. TRÆNA 2010G
20, Halsa
0, Halsa
2, LOK. TOMSVIK 09G
3, LOK. STORURDVIKA 10G VÅR
17, LOK. ASPVIKSUNDET/SKONSENG
4, LOK. ANDALSVÅGEN
5, LOK. LEKAFJORD
6, LOK. LEKAFJORD
16, LOK. LEKAFJORD
15, LOK. HAUGSFLESA
13, LOK. BÅFJORDSTRANDA
10, LOK. FLOTAREN
Švédsko
Botnický záliv
Finsko
Bílé moře
Republic of Karelia
Norsko
Helsinky
Finský záliv
©2010 Google

Rich Vehicle Routing Problems

- **Goal** – maximization of the ship *filling rate* (operational planning), optimization of fleet composition, i.e. number and capacity of the ships (strategic planning)
- **Rich Vehicle Routing Problem**
  - time windows
  - heterogeneous fleet (vehicles with different capacities and speed)
  - several depots with inter-depot trips
  - several routes during the planning horizon
  - *non-Euclidean distances* (fjords)
- Mixed-integer programming :-(, constructive heuristics for getting an initial feasible solution and tabu search

- M. Branda, K. Haugen, J. Novotný, A. Olstad, **Downstream logistics optimization at EWOS Norway**. Research report.

Rich Vehicle Routing Problems

Our approach

- Mathematical formulation
- GAMS implementation
- Heuristic (insertion, tabu search) implementation
- Decision Support System (DSS)

Facility Location Problem

- i warehouses (facilities), j customers
- x_{ij} – sent quantity
- y_i – a warehouse is built
- c_{ij} – unit supplying costs
- f_i – fixed costs
- K_i – warehouse capacity
- D_j – demand

$$\min_{x_{ij}, y_i} \sum_{i=1}^{n} \sum_{j=1}^{m} c_{ij} x_{ij} + \sum_{i} f_i y_i$$

$$\text{s.t.} \sum_{j=1}^{m} x_{ij} \leq K_i y_i, \; i = 1, \ldots, n,$$

$$\sum_{i=1}^{n} x_{ij} = D_j, \; j = 1, \ldots, m,$$

$$x_{ij} \geq 0, \; y_i \in \{0, 1\}.$$

Scheduling to Minimize the Makespan

- i machines, j jobs,
- y – machine makespan,
- x_{ij} – assignment variable
- t_{ij} – time necessary to process job j on machine i,

$$\begin{aligned} \min_{x_{ij},y} \; & y \\ \text{s.t.} \; & \sum_{i=1}^{m} x_{ij} = 1, \; j = 1, \ldots, n, \\ & \sum_{j=1}^{n} x_{ij} t_{ij} \leq y, \; i = 1, \ldots, m. \end{aligned} \tag{19}$$

Lot Sizing Problem

Uncapacitated single item LSP

- x_t – production at period t
- y_t – on/off decision at period t
- s_t – inventory at the end of period t ($s_0 \geq 0$ fixed)
- D_t – (predicted) *expected* demand at period t
- p_t – unit production costs at period t
- f_t – setup cost at period t
- h_t – inventory cost at period t
- M – large constant

$$\begin{aligned} \min_{x_t, y_t, s_t} \quad & \sum_{t=1}^{T} (p_t x_t + f_t y_t + h_t s_t) \\ \text{s.t.} \quad & s_{t-1} + x_t - D_t = s_t, \;\; t = 1, \ldots, T, \\ & x_t \leq M y_t, \\ & x_t, s_t \geq 0, \;\; y_t \in \{0, 1\}. \end{aligned} \tag{20}$$

ASS. Wagner-Whitin costs $p_{t+1} \leq p_t + h_t$.

Lot Sizing Problem

Capacitated single item LSP

- x_t – production at period t
- y_t – on/off decision at period t
- s_t – inventory at the end of period t ($s_0 \geq 0$ fixed)
- D_t – (predicted) *expected* demand at period t
- p_t – unit production costs at period t
- f_t – setup cost at period t
- h_t – inventory cost at period t
- C_t – production capacity at period t

$$\begin{aligned} \min_{x_t, y_t, s_t} \quad & \sum_{t=1}^{T} (p_t x_t + f_t y_t + h_t s_t) \\ \text{s.t.} \quad & s_{t-1} + x_t - D_t = s_t, \;\; t = 1, \ldots, T, \\ & x_t \leq C_t y_t, \\ & x_t, s_t \geq 0, \;\; y_t \in \{0, 1\}. \end{aligned} \tag{21}$$

ASS. Wagner-Whitin costs $p_{t+1} \leq p_t + h_t$.

Unit Commitment Problem

- y_{it} – on/off decision for unit i at period t
- x_{it} – production level for unit i at period t
- D_t – (predicted) *expected* demand at period t
- p_i^{min}, p_i^{max} – minimal/maximal production capacity of unit i
- c_{it} – (fixed) start-up costs
- f_{it} – variable production costs

$$
\begin{aligned}
\min_{x_{it}, y_{it}} \quad & \sum_{i=1}^{n} \sum_{t=1}^{T} (c_{it} x_{it} + f_{it} y_{it}) \\
\text{s.t.} \quad & \sum_{i=1}^{n} x_{it} \geq D_t, \;\; t = 1, \ldots, T, \\
& p_i^{min} y_{it} \leq x_{it} \leq p_i^{max} y_{it}, \\
& x_{it} \geq 0, \;\; y_{it} \in \{0, 1\}.
\end{aligned}
\tag{22}
$$

Literature

- O. Kafka: **Optimální plánování rozvozu pomocí dopravních prostředků**, Diploma thesis MFF UK, 2013. (IN CZECH)
- P. Toth, D. Vigo (2002). **The vehicle routing problem**, SIAM, Philadelphia.
- L.A. Wolsey (1998). **Integer Programming**. Wiley, New York.
- L.A. Wolsey, G.L. Nemhauser (1999). **Integer and combinatorial optimization**. Wiley, New York.

Lagrangian duality in nonlinear programming

Martin Branda

Charles University in Prague
Faculty of Mathematics and Physics
Department of Probability and Mathematical Statistics

Computational Aspects of Optimization

Nonlinear Programming Problem (NLP)

Primal problem (P):

$$
\begin{aligned}
(P) = \min_{x \in X} f(x) \text{ s.t. } g_j(x) &\leq 0, \; j = 1, \ldots, m, \\
h_i(x) &= 0, \; i = 1, \ldots, l.
\end{aligned}
$$

Lagrangian function, $u \in \mathbb{R}^m_+$, $v \in \mathbb{R}^l$:

$$
L(x, u, v) = f(x) + \sum_{j=1}^{m} u_j g_j(x) + \sum_{i=1}^{l} v_i h_i(x)
$$

Dual problem

Dual function:

$$\theta(u, v) = \inf_{x \in X} L(x, u, v) \tag{1}$$

Dual problem (D):

$$(D) = \sup_{u \geq 0, v} \theta(u, v) \tag{2}$$

Weak Duality Theorem

Theorem

Let x be feasible for problem (P) and (u, v) be feasible for problem (D). Then

$$\theta(u, v) \leq f(x).$$

Proof.

$$\theta(u, v) = \inf_y L(y, u, v) \leq L(x, u, v) \leq f(x),$$

where the last inequality follows from feasibility of x and (u, v), when $u_j g_j(x) \leq 0$ and $v_i h_i(x) = 0$.

Weak Duality Theorem – Consequences

1. We obtain
$$(P) \geq (D).$$
2. If for some primal feasible $\overline{x}$ and dual feasible $(\overline{u}, \overline{v})$ holds
$$f(\overline{x}) = \theta(\overline{u}, \overline{v}),$$
then $\overline{x}$ is optimal solution of (P) and $(\overline{u}, \overline{v})$ is optimal solution of (D).
3. If $(P) = -\infty$ (unbounded primal problem), then $\theta(u, v) = -\infty$ for all $(u, v) \in \mathbb{R}^m_+ \times \mathbb{R}^l$.
4. If $(D) = \infty$, then (P) is infeasible.

Strong Duality Theorem

Theorem

Let

- X *be a nonempty convex set*
- f, g_j *be* **convex**
- h_i *be* **affine**
- **Slater condition** *be satisfied, i.e. there is* $\hat{x} \in X$ *such that* $g_j(\hat{x}) < 0, \forall j$ *and* $h_i(\hat{x}) = 0, \forall i$, *and* $0 \in \operatorname{int}\{(h_1(x), \ldots, h_l(x)) : \ x \in X\} := h(X)$.

Then $(P) = (D)$.
Moreover, if (P) *is finite, then* sup *in* (D) *is achieved at* $(\overline{u}, \overline{v}) \in \mathbb{R}^m_+ \times \mathbb{R}^l$.
If inf *in* (P) *is achieved at* $\overline{x}$, *then* $\sum_{j=1}^m \overline{u}_j g_j(\overline{x}) = 0$.

A counterexample

Convexity alone is not sufficient. Consider

$$\begin{aligned} p^* = \min_{x,y} \; & e^{-x} \\ \text{s.t. } & x^2/y \leq 0, \\ & y > 0 \;\; (\text{or } y \geq \varepsilon). \end{aligned}$$

The optimal value is $p^* = 1$. The dual function is equal to

$$\theta(u) = \inf_{x,y>0} e^{-x} + ux^2/y = \begin{cases} 0 & u \geq 0, \\ -\infty & u < 0. \end{cases}$$

The dual problem is

$$d^* = \max_{u \geq 0} \theta(u)$$

with optimal value $d^* = 0$. Slater condition is not satisfied since $x = 0$ for any feasible (x, y), i.e. $x^2/y = 0$.

SDT proof

Bazaraa et al. (2006), Lemma 6.2.3:

Lemma

Let $X \subseteq \mathbb{R}^n$ be a convex set, $f, g_j : \mathbb{R}^n \to \mathbb{R}$ be convex, $h_i : \mathbb{R}^n \to \mathbb{R}$ be affine. If System 1 has no solution, then System 2 has a solution (u_0, u, v). The converse holds true if $u_0 > 0$.

System 1: $f(x) < 0$, $g_j(x) \leq 0$, $h_i(x) = 0$ *for some* $x \in X$.

System 2: $u_0 f(x) + \sum_{j=1}^{m} u_j g_j(x) + \sum_{i=1}^{l} v_i h_i(x) \geq 0$ *for all* $x \in X$, $(u_0, u) \geq 0$, $(u_0, u, v) \neq 0$.

SDT proof

Let γ be a (finite) optimal value of (P) and consider the following system:

$$f(x) - \gamma < 0,\ g_j(x) \leq 0, j = 1, \ldots, m,\ h_i(x) = 0, i = 1, \ldots, l,\ x \in X.$$

By the definition of γ the system has no solution. Hence, there exists $(u_0, u, v) \neq 0$ with $(u_0, u) \geq 0$ such that

$$u_0(f(x) - \gamma) + \sum_{j=1}^{m} u_j g_j(x) + \sum_{i=1}^{l} v_i h_i(x) \geq 0,\ \forall x \in X.$$

SDT proof

Suppose that $u_0 = 0$. By assumption there is an $\hat{x} \in X$ such that $g_j(\hat{x}) < 0, \forall j$ and $h_i(\hat{x}) = 0, \forall i$. Substituting into the inequality we obtain $\sum_{j=1}^{m} u_j g_j(\hat{x}) \geq 0$. Since $g_j(\hat{x}) < 0, \forall j$, we have $u_j = 0, \forall j$, and $u_0 = 0$. This implies that $\sum_{i=1}^{l} v_i h_i(x) \geq 0$ for all $x \in X$. Since $0 \in h(X)$, we can pick a $x \in X$ such that $h_i(x) = -\lambda v_i$, where $\lambda > 0$ (small). Therefore

$$\sum_{i=1}^{l} v_i h_i(x) = -\lambda \sum_{i=1}^{l} v_i^2 \geq 0,$$

which implies that $v_i = 0, \forall i$. But this is a contradiction with $(u_0, u, v) \neq 0$. Hence $u_0 > 0 \ldots$

SDT proof

Hence $u_0 > 0$. Thus, if we set $\tilde{u}_j = u_j/u_0$ and $\tilde{v}_i = v_i/u_0$, we get

$$f(x) + \sum_{j=1}^{m} \tilde{u}_j g_j(x) + \sum_{i=1}^{l} \tilde{v}_i h_i(x) \geq \gamma, \ \forall x \in X.$$

This shows that

$$\theta(\tilde{u}, \tilde{v}) = \inf_{x \in X} L(x, \tilde{u}, \tilde{v}) \geq \gamma.$$

Together with the Weak Duality Theorem we obtain that

$$\gamma = \theta(\tilde{u}, \tilde{v}) = \sup_{u \geq 0, v} \theta(u, v).$$

Example: Linear programming duality

$$\begin{aligned} \min\ & c^T x \\ \text{s.t.}\ & Ax = b, \\ & x \geq 0. \end{aligned}$$

Example: Ordinary least squares with equality constraints

$$\begin{aligned} \min \ & \|Ax - b\|_2^2 \\ \text{s.t. } & Fx = g. \end{aligned}$$

Example: The support vector classifier

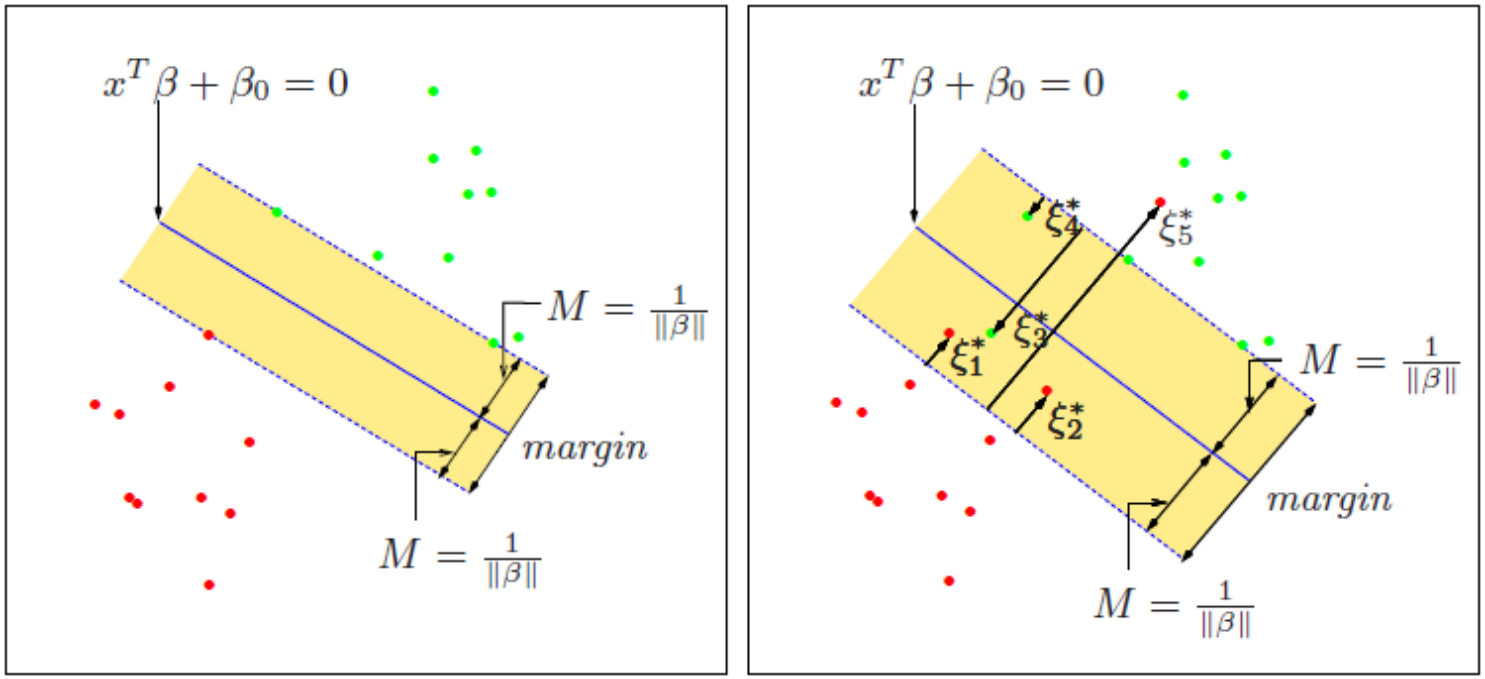
Hastie et al. (2009): Training data: N pairs (x_1, y_1), (x_2, y_2), $\ldots$, (x_N, y_N), $x_i \in \mathbb{R}^p$, $y_i \in \{-1, 1\}$ (classes).
A linear classification rule with $\|\beta\| = 1$

$$G(x) = \mathrm{sign}[x^T \beta + \beta_0].$$

Assume first that the data are separable. We would like to find **the biggest margin** between the training points for class 1 and -1:

$$\begin{aligned} &\max_{\beta_0, \beta} M \\ &\text{s.t. } y_i(x_i^T \beta + \beta_0) \geq M, \ i = 1, \ldots, N, \\ &\qquad \|\beta\| = 1. \end{aligned}$$

Example: The support vector classifier



Hastie et al. (2009)

Example: The support vector classifier

By setting $M = 1/\|\beta\|$:

$$\min_{\beta_0,\beta} \|\beta\|$$
$$\text{s.t. } y_i(x_i^T\beta + \beta_0) \geq 1, \; i = 1, \ldots, N.$$

If the classes overlap:

$$\min_{\beta_0,\beta,\xi} \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{N}\xi_i$$
$$\text{s.t. } y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i, \; i = 1, \ldots, N,$$
$$\xi_i \geq 0,$$

where we penalize the overall overlap.

Example: The support vector classifier

Lagrange function

$$L(\beta_0, \beta, \xi, \alpha, \mu) = \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\mu_i\xi_i$$
$$- \sum_{i=1}^{N}\alpha_i(y_i(x_i^T\beta + \beta_0) - 1 + \xi_i),\ \alpha_i \geq 0, \mu_i \geq 0.$$

The dual function

$$\theta(\alpha, \mu) = \inf_{\beta_0, \beta, \xi} L(\beta_0, \beta, \xi, \alpha, \mu).$$

Example: The support vector classifier

$$L(\beta_0, \beta, \xi, \alpha, \mu) = \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\mu_i\xi_i$$
$$- \sum_{i=1}^{N}\alpha_i(y_i(x_i^T\beta + \beta_0) - 1 + \xi_i),\ \ \alpha_i \geq 0, \mu_i \geq 0$$

Use the derivatives to obtain the dual function:

$$\begin{aligned}
\frac{\partial L}{\partial \beta_0} &= \sum_{i=1}^{N}\alpha_i y_i = 0, \\
\frac{\partial L}{\partial \beta} &= \beta - \sum_{i=1}^{N}\alpha_i y_i x_i = 0, \\
\frac{\partial L}{\partial \xi_i} &= C - \alpha_i - \mu_i = 0.
\end{aligned}$$

Example: The support vector classifier

We can express the dual function

$$
\begin{aligned}
\theta(\alpha,\mu) &= \frac{1}{2}\sum_{i=1}^{N}\sum_{i'=1}^{N}\alpha_i\alpha_{i'}y_iy_{i'}x_i^Tx_{i'} + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\sum_{i'=1}^{N}\alpha_i\alpha_{i'}y_iy_{i'}x_i^Tx_{i'} \\
&\quad -\beta_0\sum_{i=1}^{N}\alpha_iy_i + \sum_{i=1}^{N}\alpha_i - \sum_{i=1}^{N}\alpha_i\xi_i - \sum_{i=1}^{N}\mu_i\xi_i \\
&= -\frac{1}{2}\sum_{i=1}^{N}\sum_{i'=1}^{N}\alpha_i\alpha_{i'}y_iy_{i'}x_i^Tx_{i'} + \sum_{i=1}^{N}\alpha_i,
\end{aligned}
$$

subject to $0 \leq \alpha_i \leq C$, $\sum_{i=1}^{N}\alpha_iy_i = 0$.

Literature

- Bazaraa, M.S., Sherali, H.D., and Shetty, C.M. (2006). **Nonlinear programming: theory and algorithms**, Wiley, Singapore, 3rd edition.
- Boyd, S., Vandenberghe, L. (2004). **Convex Optimization**, Cambridge University Press, Cambridge.
- Hastie, T., Tibshirani, R., Friedman, J. (2009). **The Elements of Statistical Learning: Data Mining, Inference, and Prediction**. Springer Series in Statistics, 2nd edition.

Algorithms for nonlinear programming problems I

Martin Branda

Charles University in Prague
Faculty of Mathematics and Physics
Department of Probability and Mathematical Statistics

Computational Aspects of Optimization

Algorithm classification

- **Order of derivatives**[1]**:** derivative-free, first order (gradient), second-order (Newton)
- **Feasibility of the constructed points**: interior and exterior point methods
- **Deterministic/randomized**
- **Local**/global

[1] If possible, deliver the derivatives.

Unconstrained problems

Let $f : \mathbb{R}^n \to \mathbb{R}$, x^0 be a starting point, $d^k \in \mathbb{R}^n$ be a **descent direction**, and $\lambda \in \mathbb{R}$ be a **step length**.

Find a **descent direction** d^k, solve the **line search problem**

$$\lambda^k = \arg \min_{0 \le \lambda \le \lambda_{max}} f(x^k + \lambda d^k)$$

and set

$$x^{k+1} = x^k + \lambda^k d^k.$$

Iterate until a convergence criterion is not satisfied, e.g. $\left\| d^k \right\| < \varepsilon$ or $|f(x^k) - f(x^{k+1})| < \varepsilon$.

Review of line search methods

Bazaraa et al. (2006):

- Derivative-free: dichotomous search, golden section method, Fibonacci search
- Using derivatives: bisection search, Newton's method

Descent directions – Steepest descent

A vector d is called a descent direction of a function f at x if there exists a $\delta > 0$ such that

$$f(x + \lambda d) < f(x), \ \lambda \in (0, \delta).$$

Steepest descent d with $\|d\| = 1$ minimizes the limit

$$f'(x; d) := \lim_{\lambda \to 0_+} \frac{f(x + \lambda d) - f(x)}{\lambda} < 0.$$

If f is differentiable at x with a nonzero gradient, then

$$d = -\frac{\nabla f(x)}{\|\nabla f(x)\|}$$

leading to the gradient (Cauchy) method.

$$f'(x; d) = \nabla f(x)^T d.$$

Descent directions

If we set

$$h(\lambda) := f(x + \lambda d),$$

then

$$h'(0) = \nabla f(x)^T d.$$

h is decreasing $\Leftrightarrow$ f is decreasing in direction d.

Descent directions

Steepest descent – works well during the early steps, the **zigzagging** phenomenon often appears in later steps, see Bazaraa et al. (2006), Example 8.6.2

Descent directions – Newton direction

Approximation of f by a limited Taylor expansion around x^k

$$g(x) := f(x^k) + \nabla f(x^k)^T (x - x^k) + \frac{1}{2}(x - x^k)^T \nabla^2 f(x^k)(x - x^k)$$

Setting $\nabla_x g(x) = 0$, we obtain the **Newton direction**

$$d = -\left(\nabla^2 f(x^k)\right)^{-1} \nabla f(x^k).$$

If $\nabla^2 f(x^k) > 0$, then d is a descent direction[2].

[2] In general, $d = -A\nabla f(x^k)$ for $A > 0$ is a descent direction $\rightarrow$ **Quasi-Newton methods**.

Descent directions – Newton direction

Convergence of the algorithm: Bazaraa et al. (2006), Theorem 8.6.5 ($f \in C^2$, $\nabla f(\overline{x}) = 0$ and $\nabla^2 f(\overline{x}) > 0$ at a local minimum $\overline{x}$, starting point is sufficiently close.)

Descent directions – Example

$$\min_{x,y} (x-y)^4 + 2x^2 + y^2 - x + 2y$$

Partial derivatives

$$\begin{aligned} \frac{\partial f(x,y)}{\partial x} &= 4(x-y)^3 + 4x - 1 = 0, \\ \frac{\partial f(x,y)}{\partial y} &= -4(x-y)^3 + 2y + 2 = 0. \end{aligned} \tag{1}$$

Second-order partial derivatives

$$\begin{aligned} \frac{\partial^2 f(x,y)}{\partial x^2} &= 12(x-y)^2 + 4, \\ \frac{\partial^2 f(x,y)}{\partial x \partial y} &= -12(x-y)^2, \\ \frac{\partial^2 f(x,y)}{\partial y^2} &= 12(x-y)^2 + 2. \end{aligned} \tag{2}$$

Compare directions $\nabla f(x)$ and $d = -\left(\nabla^2 f(x)\right)^{-1} \nabla f(x)$ ...

Conjugate gradient method

Nocedal and Wright (2006), Chapter 5: Consider (unconstrained) quadratic programming problem

$$\min \frac{1}{2} x^T A x - b^T x.$$

Let $A \in \mathbb{R}^{n \times n}$ be a symmetric positive definite matrix. We say that vectors $p^1, \dots p^n$ are conjugate with respect to A if

$$(p^i)^T A p^j = 0 \text{ for all } i \neq j.$$

If we set $x^{k+1} = x^k + \alpha^k p^k$, where

$$\begin{aligned} r^k &= A x^k - b, \\ \alpha^k &= -\frac{r^{k\,T} p^k}{(p^k)^T A p^k}, \end{aligned} \tag{3}$$

then x^{n+1} is an optimal solution.

Method of Zoutendijk

Bazaraa et al. (2006), Section 10.1: $f : \mathbb{R}^n \to \mathbb{R}$, $g_j : \mathbb{R}^n \to \mathbb{R}$ **differentiable**

$$\min_x f(x) \text{ s.t. } g_j(x) \leq 0, \; j = 1, \ldots, m.$$

(Extension including equality constraints is possible.)

Method based on **improving feasible directions** (remember the "directional" optimality conditions).

Method of Zoutendijk

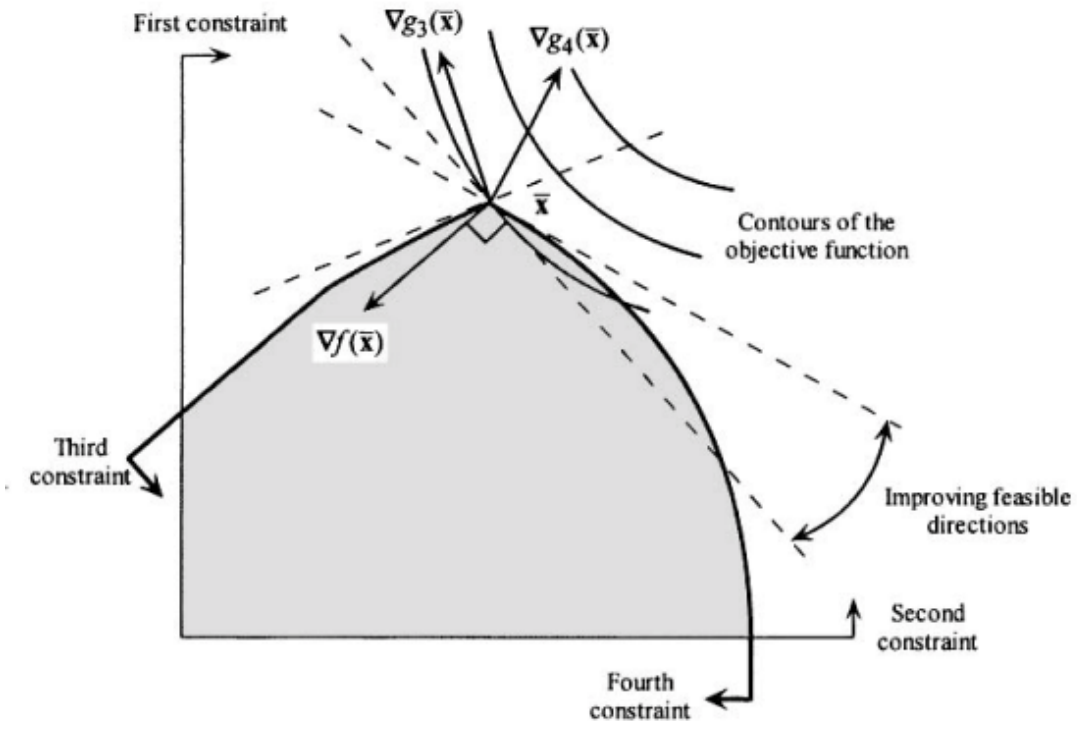
0. Start with a **feasible** x^1. For $k = 1, \dots (, K_{max})$ do
1. Set $J(x^k) = \{j : g_j(x^k) = 0\}$ and solve **linear programming problem for finding a direction**:

$$\begin{aligned} \min_{z,d} \; & z \\ \text{s.t.} \; & \nabla f(x^k)^T d \leq z, \\ & \nabla g_j(x^k)^T d \leq z, \; j \in J(x^k), \\ & -1 \leq d_i \leq 1, \; i = 1, \dots, n. \end{aligned}$$

Denote by $(z^k, d^k) \in \mathbb{R}^{1+n}$ the optimal solution.

- If $z^k = 0$ then STOP (We have found a Fritz-John point).
- Else if $z^k < 0$ then continue with STEP 2.

Method of Zoutendijk



Bazaraa et al. (2006)

Method of Zoutendijk

2. Find **maximal possible step**

$$\lambda_{max} := \sup\{\lambda : \ g_j(x^k + \lambda d^k) \leq 0, j = 1, \ldots, m\},$$

solve the **line search problem**

$$\lambda^k = \arg \min_{0 \leq \lambda \leq \lambda_{max}} f(x^k + \lambda d^k)$$

and set

$$x^{k+1} = x^k + \lambda^k d^k.$$

Continue with STEP 1.

Method of Zoutendijk

Where could be a problem? Direction as well as line search need not to be closed...

Convergence: Bazaraa et al. (2006), part 10.2.

Method of Zoutendijk – example

Bazaraa et al. (2006), Example 10.1.8

$$\begin{aligned} \min\ & 2x_1^2 + 2x_2^2 - 2x_1x_2 - 4x_1 - 6x_2 \\ \text{s.t. } & x_1 + x_2 \leq 5, \\ & 2x_1^2 - x_2 \leq 0, \\ & -x_1 \leq 0, \\ & -x_2 \leq 0. \end{aligned} \tag{4}$$

$$\nabla f(x) = (4x_1 - 2x_2 - 4,\ 4x_2 - 2x_1 - 6)^T \tag{5}$$

Method of Zoutendijk – Example

Starting point $x^0 = (0, 0.75)^T$, $\nabla f(x^0) = (-5.5, -3)^T$, $J(x^0) = \{3\}$. The direction finding problem is then

$$\begin{aligned} \min\ & z \\ \text{s.t.}\ & -5.5d_1 - 3d_2 \leq z, \\ & -d_1 \leq z, \\ & -1 \leq d_1, d_2 \leq 1. \end{aligned} \tag{6}$$

with optimal solution $d^1 = (1, -1)$, $z^1 = -1$.

Method of Zoutendijk – Example

Then

$$x^0 + \lambda d^1 = (\lambda,\ 0.75 - \lambda)$$

and

$$f(x^0 + \lambda d^1) = 6\lambda^2 - 2.5\lambda - 3.375.$$

Maximize it over the set of feasible solutions M to obtain $\lambda_{max} = 0.4114$. Finally

$$\begin{aligned} &\min\ 6\lambda^2 - 2.5\lambda - 3.375 \\ &\text{s.t. } 0 \leq \lambda \leq \lambda_{max}. \end{aligned} \tag{7}$$

$\lambda^1 = 0.2083.$

Cutting plane method

$f : \mathbb{R}^n \to \mathbb{R}$, $g_j : \mathbb{R}^n \to \mathbb{R}$

$$\min_x f(x) \text{ s.t. } g_j(x) \leq 0, \; j = 1, \ldots, m.$$

Denote $M = \{x \in \mathbb{R} : \; g_j(x) \leq 0, \; j = 1, \ldots, m\}$.

ASS. f is affine, g are convex and differentiable, M is compact.

Cutting plane method

0. Start with a polyhedral set M^0 such that $M \subset M^0$, e.g. a box $M^0 = [lb_1, ub_1] \times \cdots \times [lb_m, ub_m]$. For $k = 0, \ldots (, K_{max})$ do
1. Solve the **linear programming problem**

$$\min_x f(x) \text{ s.t. } x \in M^k,$$

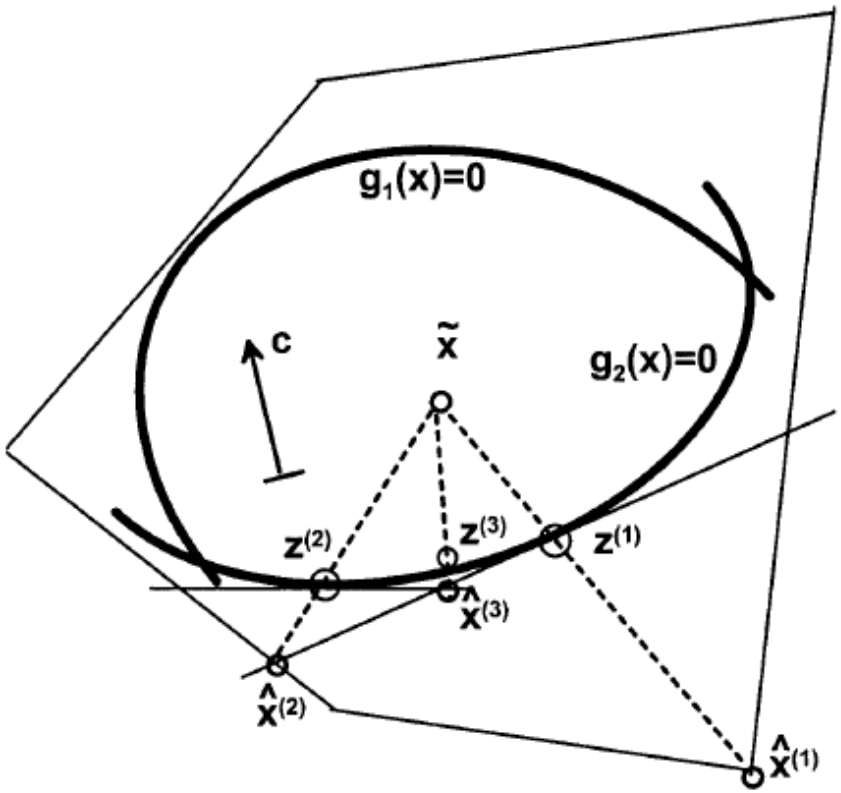
   and obtain $x^k \in M^k$. If $x^k \in M$, then STOP, we have found an optimal solution. Otherwise continue with STEP 2.
2. If $x^k \notin M$, then find $j^k = \arg\max_j g_j(x^k)$, construct a **cutting plane** and set

$$M^{k+1} = M^k \cap \left\{ x \in \mathbb{R} : \; g_{j^k}(x^k) + \nabla g_{j^k}(x^k)^T (x - x^k) \leq 0 \right\}.$$

   Note that x^k violates the cut, and no $x \in M$ is cut off[3] (compare with the integer programming cuts). Return to STEP 1.

[3] From convexity $g_{j^k}(x^k) + \nabla g_{j^k}(x^k)^T (x - x^k) \leq g_{j^k}(x) \leq 0$.

Cutting plane method



Kall and Mayer (2005).

Cutting plane method – Example

$$\begin{aligned} \min_x \quad & -x_1 - x_2 \\ \text{s.t.} \quad & x_1^2 + x_2^2 - 1 \leq 0, \\ & x_1, x_2 \geq 0. \end{aligned}$$

Set $M = \{(x_1, x_2) : \; x_1^2 + x_2^2 - 1 \leq 0, x_1, x_2 \geq 0\}$, $\nabla g(x)^T = (2x_1, 2x_2)$.

Cutting plane method – Example

0. Set $M^0 = [0,1]^2$.
1. Solve $\min_x -x_1 - x_2$ s.t. $x \in M^0$ with optimal solution $x^0 = (1,1)^T$.
2. Since $x^0 \notin M$, construct the cut

$$g(x^0) + \nabla g(x^0)^T (x - x^0) \leq 0,$$

and set

$$M^1 = M^0 \cap \{(x_1, x_2) : \; x_1 + x_2 \leq 3/2\}.$$

Continue with STEP 1.

Penalty method

Perfect penalty (rather theoretical)

$$PP(x) = \begin{cases} 0 & \text{if } g(x) \leq 0, h(x) = 0, \\ \\ \infty & \text{otherwise.} \end{cases}$$

Compare with Lagrangian duality (sup over multipliers).

The following problem is equivalent to the original constrained one.

$$\min_x f(x) + PP(x).$$

Penalty method

$L_{p,q}-$ **penalty function**:

$$PF_N(x) = N \cdot \left(\sum_{j=1}^{m} [g_j(x)]_+^p + \sum_{i=1}^{l} |h_i(x)|^q \right),$$

where N is the penalty parameter, $[\cdot]_+ = \max\{\cdot, 0\}$.

More general penalty using $\Phi(y) = 0$ for $y \leq 0$ and $\Phi(y) > 0$ for $y > 0$ and $\Psi(y) = 0$ for $y = 0$ and $\Psi(y) > 0$ for $y \neq 0$.

Penalty method

Algorithm:

0. Set $\varepsilon > 0$, $N^1 > 0$, $\beta > 1$. For $k = 1, \dots (, K_{max})$ do:
1. Solve
$$\min_x f(x) + PF_{N^k}(x).$$
and obtain x^k
2. IF $PF_{N^k}(x^k) < \varepsilon$, then STOP. ELSE set $N^{k+1} = N^k \cdot \beta$ and continue with STEP 1.

Exterior point method!

Penalty method

Convergence of the method: Bazaraa et al. (2006), Theorem 9.2.2 (continuous f, g_j, h_i, $x_k \in X$ compact).

Penalty functions – Example

Consider

$$\begin{aligned} &\min\ x_1^2 + x_2^2 \\ &\text{s.t. } x_1 + x_2 = 2. \end{aligned}$$

with optimal solution $\hat{x}_1 = \hat{x}_2 = 1$. Penalty function problem

$$\min x_1^2 + x_2^2 + N(x_1 + x_2 - 2)^2.$$

Using optimality conditions

$$\hat{x}_1^N = \hat{x}_2^N = \frac{2N}{2N+1}.$$

Penalty method

Remarks

- **Sequential Unconstrained Minimization** (SUMT): optimal solution x^k is used as a starting point in the next iteration[4] to solve the penalty problem with N_{k+1}.
- **Exact penalty**: Instead of $N \to \infty$ it is sufficient to converge $N \to \overline{N} < \infty$ (numerically more stable).

[4] "warm starting"

Augmented Lagrangian Method

Nocedal and Wright (2006), Section 17.3: $f : \mathbb{R}^n \to \mathbb{R}$, $h_i : \mathbb{R}^n \to \mathbb{R}$ **differentiable**

$$\begin{aligned} &\min_x \ f(x) \\ &\text{s.t. } h_i(x) = 0, \ i = 1, \dots, l. \end{aligned}$$

(Extension including inequality constraints is possible.)

$$L(x, v) = f(x) - \sum_{i=1}^{l} v_i h_i(x).$$

Augmented Lagrangian Method

Augmented Lagrangian function – combination of the **Lagrangian function** with the **quadratic penalty term**

$$L_A(x, \lambda, \mu) = f(x) - \sum_{i=1}^{l} \lambda_i h_i(x) + \frac{\mu}{2} \sum_{i=1}^{l} (h_i(x))^2 .$$

$$\begin{aligned} \nabla_x L_A(x, \lambda, \mu) &= \nabla_x f(x) - \sum_{i=1}^{l} \lambda_i \nabla_x h_i(x) + \mu \sum_{i=1}^{l} h_i(x) \nabla_x h_i(x) \\ &= \nabla_x f(x) - \sum_{i=1}^{l} (\lambda_i - \mu h_i(x)) \nabla_x h_i(x). \end{aligned}$$

We have that $v_i \approx \lambda_i - \mu h_i(x)$.

Augmented Lagrangian Method

0. Set initial $\mu^1 > 0$, $\beta > 1$ and λ^1. Select a tolerance $\varepsilon > 0$. For $k = 1, \ldots (, K_{max})$ do:
1. Solve unconstrained problem

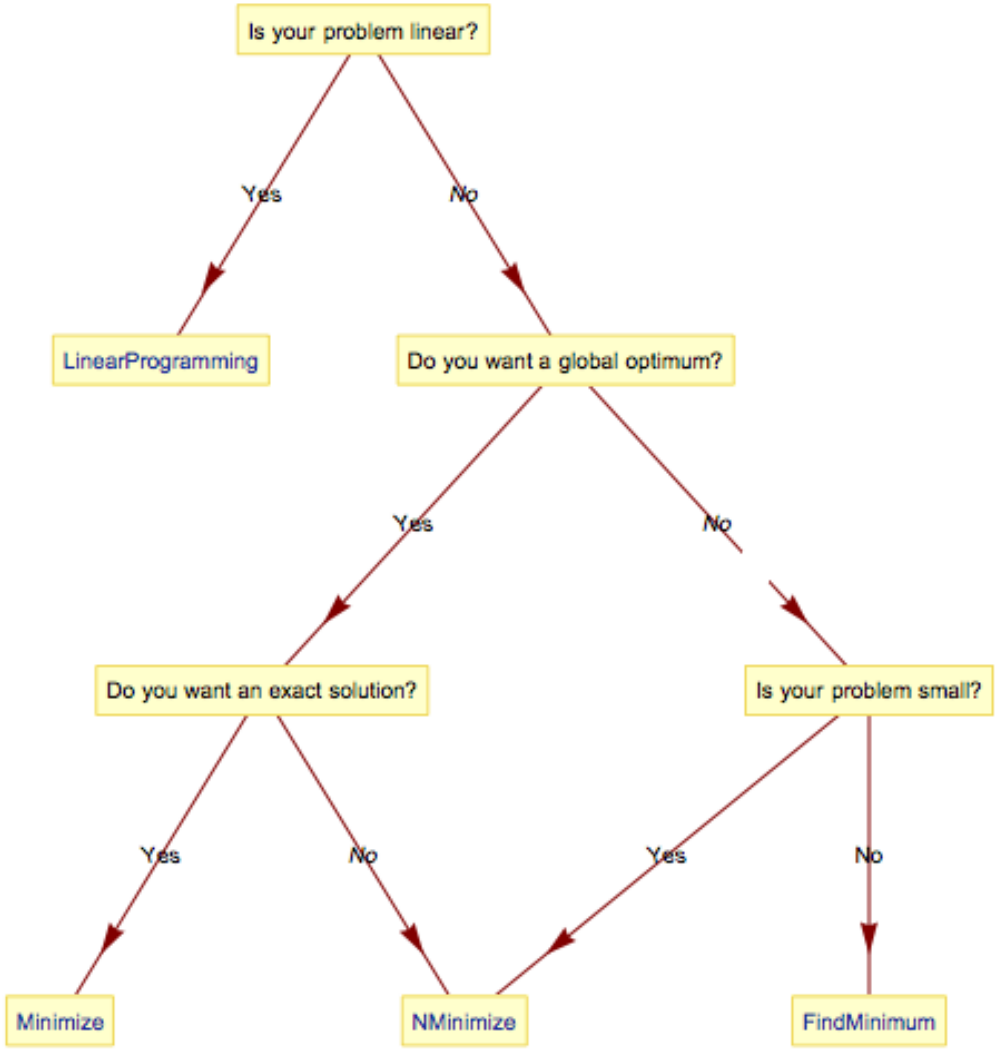
$$\min_x L_A(x, \lambda^k, \mu^k)$$

   and obtain x^k. If $\left\| L_A(x^k, \lambda^k, \mu^k) \right\| \leq \varepsilon$, STOP. Otherwise continue with STEP 2.
2. Update the Lagrange multipliers $\lambda_i^{k+1} = \lambda_i^k - \mu^k h_i(x^k)$ and the penalty parameter $\mu^{k+1} = \beta \mu^k$. Go to STEP 1.

Augmented Lagrangian Method

Convergence of the algorithm: Nocedal and Wright (2006), Theorem 17.5 (LICQ, SOSC).

Mathematica – Solver Decision Tree



Literature

- Bazaraa, M.S., Sherali, H.D., and Shetty, C.M. (2006). **Nonlinear programming: theory and algorithms**, Wiley, Singapore, 3rd edition.
- Boyd, S., Vandenberghe, L. (2004). **Convex Optimization**. Cambridge University Press, Cambridge.
- P. Kall, J. Mayer: **Stochastic Linear Programming: Models, Theory, and Computation**. Springer, 2005.
- Nocedal, J., Wright, J.S. (2006). **Numerical optimization**. Springer, New York, 2nd edition.

Algorithms for nonlinear programming problems II

Martin Branda

Charles University in Prague
Faculty of Mathematics and Physics
Department of Probability and Mathematical Statistics

Computational Aspects of Optimization

Algorithm convergence

Definition

Let $X \subseteq \mathbb{R}^p$, $Y \subseteq \mathbb{R}^q$ be nonempty closed sets. Let $F : X \to Y$ be a set-valued mapping. The map F is said to be **closed** at $x \in X$ if for any sequences $\{x^k\} \subset X$, and $\{y^k\}$ satisfying $x_k \to x$, $y^k \in F(x^k)$, $y^k \to y$ we have that $y \in F(x)$.
The map F is said to be closed on $Z \subseteq X$ if it is closed at each point in Z.

Algorithm convergence – Zangwill's theorem

Bazaraa et al. (2006), Theorem 7.2.3: Let

A1. $X \subseteq \mathbb{R}^p$ be a nonempty **closed** set

A2. $\hat{X} \subseteq X$ be a **nonempty solution set**

A3. $F : X \to X$ be a set-valued mapping **closed** over complement of $\hat{X}$

A4. Given $x^1 \in X$ the sequence $\{x^k\}$ is generated iteratively as follows: If $x^k \in \hat{X}$, then STOP; otherwise, let $x^{k+1} \in F(x^k)$ and repeat.

A5. the sequence $x^1, x^2, \ldots$ be contained in a **compact** subset of X

A6. there exist a **continuous function**[1] α such that $\alpha(y) < \alpha(x)$ if $x \notin \hat{X}$ and $y \in F(x)$

Then either the algorithm stops in a finite number of steps with a point in $\hat{X}$ or it generates an infinite sequence $\{x^k\}$ such that all accumulation points belong to $\hat{X}$ and $\alpha(x^k) \to \alpha(x)$ for some $x \in \hat{X}$.

[1] descent function: $\alpha(x) = f(x)$ or $\alpha(x) = \|\nabla f(x)\|$

Algorithm convergence – Newton method

Let $\overline{x}$ be an optimal solution, set

$$\begin{aligned} F(x) &= x - \left(\nabla^2 f(x)\right)^{-1} \nabla f(x), \\ \alpha(x) &= \|x - \overline{x}\| . \end{aligned}$$

More details: Bazaraa et al. (2006), Theorem 8.6.5

Algorithm classification

- **Order of derivatives**[2]**:** derivative-free, first order (gradient), second-order (Newton)
- **Feasibility of the constructed points**: interior and exterior point methods
- **Deterministic/randomized**
- **Local**/global

[2] If possible, deliver the derivatives.

Barrier method

$f : \mathbb{R}^n \to \mathbb{R}$, $g_j : \mathbb{R}^n \to \mathbb{R}$

$$\min_x f(x) \text{ s.t. } g_j(x) \leq 0, \; j = 1, \ldots, m.$$

(Extension including equality constraints is straightforward.)

Assumption

$$\{x : \; g(x) < 0\} \neq \emptyset.$$

Barrier method

Barrier functions: continuous with the following properties

$$B(y) \geq 0, y < 0, \quad \lim_{y \to 0_-} B(y) = \infty.$$

Examples

$$\frac{-1}{y}, \ -\log \min\{1, -y\}.$$

Maybe the most popular barrier function:

$$B(y) = -\log -y$$

Set

$$\tilde{B}(x) = \sum_{j=1}^{m} B\big(g_j(x)\big),$$

and solve

$$\min f(x) + \mu \tilde{B}(x), \tag{1}$$

where $\mu > 0$ is a parameter.

Polynomial-time interior point methods for LP

$$\begin{aligned} \min\ & c^T x - \mu \sum_{j=1}^{n} \log x_j \\ \text{s.t. } & Ax = b. \end{aligned} \tag{2}$$

KKT-conditions ...

Interior point methods

$f : \mathbb{R}^n \to \mathbb{R}$, $g_j : \mathbb{R}^n \to \mathbb{R}$

$$\min_x f(x) \text{ s.t. } g_j(x) \leq 0, \; j = 1, \ldots, m.$$

(Extension including equality constraints is straightforward.)

New slack decision variables $s \in \mathbb{R}^m$

$$\begin{aligned} \min_{x,s} \; & f(x) \\ \text{s.t. } & g_j(x) + s_j = 0, \; j = 1, \ldots, m, \\ & s_j \geq 0. \end{aligned} \tag{3}$$

Interior point methods

Barrier problem

$$\begin{aligned} \min_{x,s} \; & f(x) - \mu \sum_{j=1}^{m} \log s_j \\ \text{s.t. } & g(x) + s = 0. \end{aligned} \tag{4}$$

The barrier term prevents the components of s from becoming too close to zero.

Lagrangian function

$$L(x, s, z) = f(x) - \mu \sum_{j=1}^{m} \log s_j - \sum_{j=1}^{m} z_j (g_j(x) + s_j).$$

Interior point methods

KKT conditions for barrier problem (matrix notation)

$$\begin{aligned} \nabla f(x) - \nabla g^T(x) z &= 0, \\ -\mu S^{-1} e - z &= 0, \\ g(x) + s &= 0, \end{aligned}$$

$S = \mathrm{diag}\{s_1, \ldots, s_m\}$, $Z = \mathrm{diag}\{z_1, \ldots, z_m\}$, $\nabla g(x)$ is the Jacobian matrix (components of function in rows?)

Multiply the second equality by S

$$\begin{aligned} \nabla f(x) - \nabla g^T(x) z &= 0, \\ -SZe &= \mu e, \\ g(x) + s &= 0, \end{aligned}$$

= Nonlinear system of equalities $\rightarrow$ Newton's method

Newton's method

$$\nabla f(x+v) = \nabla f(x) + \nabla^2 f(x) v = 0$$

with the solution (under $\nabla^2 f(x) \succ 0$)

$$v = -\left(\nabla^2 f(x)\right)^{-1} \nabla f(x)$$

Interior point methods

Use Newton's method to obtain a step $(\Delta x, \Delta s, \Delta z)$

$$\begin{pmatrix} H(x,z) & 0 & -\nabla g^T \\ 0 & -Z & -S \\ \nabla g & I & 0 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta s \\ \Delta z \end{pmatrix} = \begin{pmatrix} -\nabla f(x) + \nabla g^T(x) z \\ SZe + \mu e \\ -g(x) - s \end{pmatrix}$$

$H(x,z) = \nabla^2 f(x) - \sum_{j=1}^m z_j \nabla^2 g_j(x)$, $\nabla^2 f$ denotes the Hessian matrix.

Interior point methods

Stopping criterion:

$$E = \max\left\{ \left\| \nabla f(x) - \nabla g^T(x) z \right\|, \|SZe + \mu e\|, \|g(x) + s\| \right\} \leq \varepsilon,$$

$\varepsilon > 0$ small.

Interior point methods

ALGORITHM:

0. Choose x^0 and $s^0 > 0$, and compute initial values for the multipliers $z^0 > 0$. Select an initial barrier parameter $\mu^0 > 0$ and parameter $\sigma \in (0, 1)$, set $k = 1$.
1. Repeat until a stopping test for the nonlinear program (19.1) is satisfied:
   - Solve the nonlinear system of equalities using Newton's method and obtain (x^k, s^k, z^k).
   - Decrease barrier parameter $\mu^{k+1} = \sigma\mu^k$, set $k = k + 1$.

Interior point methods

Convergence of the method (Nocedal and Wright 2006, Theorem 19.1): continuously differentiable f, g_j, LICQ at any limit point, then the limits are stationary points of the original problem

Interior point method – Example

$$\begin{aligned} &\min_{x} \ (x_1 - 2)^4 + (x_1 - 2x_2)^2 \\ &\text{s.t. } x_1^2 - x_2 \leq 0. \end{aligned} \tag{5}$$

$$\begin{aligned} &\min_{x,s} \ (x_1 - 2)^4 + (x_1 - 2x_2)^2 \\ &\text{s.t. } x_1^2 - x_2 + s = 0, \\ &\qquad s \geq 0. \end{aligned} \tag{6}$$

$$\begin{aligned} &\min_{x,s} \ (x_1 - 2)^4 + (x_1 - 2x_2)^2 - \mu \log s \\ &\text{s.t. } x_1^2 - x_2 + s = 0. \end{aligned} \tag{7}$$

Interior point method – Example

Lagrange function

$$L(x_1, x_2, s, z) = (x_1 - 2)^4 + (x_1 - 2x_2)^2 - \mu \log s - z(x_1^2 - x_2 + s).$$

Optimality conditions together with feasibility

$$\begin{aligned}
\frac{\partial L}{\partial x_1} &= 4(x_1 - 2)^3 + 2(x_1 - 2x_2) - 2zx_1 = 0, \\
\frac{\partial L}{\partial x_2} &= -4(x_1 - 2x_2) + z = 0, \\
\frac{\partial L}{\partial s} &= -\frac{\mu}{s} - z = 0, \\
\frac{\partial L}{\partial z} &= x_1^2 - x_2 + s = 0.
\end{aligned} \tag{8}$$

We have obtained 4 equations with 4 variables ...

Interior point method – Example

Slight modification

$$
\begin{aligned}
4(x_1-2)^3+2(x_1-2x_2)-2zx_1 &= 0, && (9)\\
-4(x_1-2x_2)+z &= 0, && (10)\\
-sz-\mu &= 0, && (11)\\
x_1^2-x_2+s &= 0. && (12)
\end{aligned}
$$

Necessary derivatives

$$
H(x_1,x_2,z) = \begin{pmatrix} 12(x_1-2)^2+2-2z & -4 \\ -4 & 8 \end{pmatrix} \quad (13)
$$

$$
\nabla g(x) = \begin{pmatrix} 2x_1 \\ -1 \end{pmatrix} \quad (14)
$$

Interior point method – Example

System of linear equations for Newton's step

$$\begin{pmatrix} 12(x_1-2)^2+2-2z & -4 & 0 & -2x_1 \\ -4 & 8 & 0 & 1 \\ 0 & 0 & -z & -s \\ 2x_1 & -1 & 1 & 0 \end{pmatrix} \begin{pmatrix} \Delta x_1 \\ \Delta x_2 \\ \Delta s \\ \Delta z \end{pmatrix} =$$

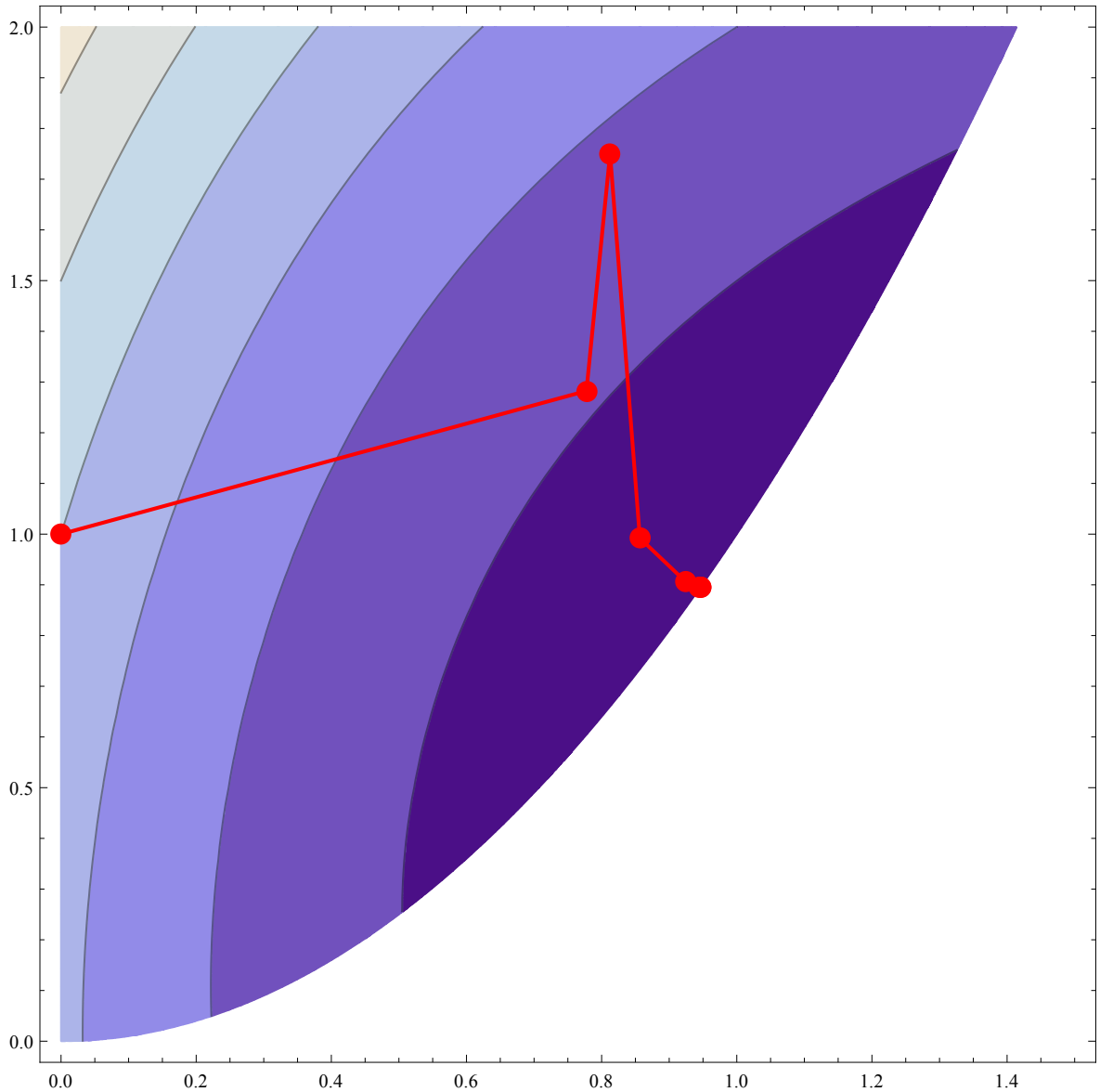
$$= \begin{pmatrix} -4(x_1-2)^3-2(x_1-2x_2)+2zx_1 \\ 4(x_1-2x_2)-z \\ sz+\mu \\ -x_1^2+x_2-s \end{pmatrix}$$

Interior point method – Example

Starting point $x^0 = (0, 1)$, $z^0 = 1$, $s^0 = 1$, $\mu > 0$, then the step ...

$$\begin{pmatrix} 48 & -4 & 0 & 0 \\ -4 & 8 & 0 & 1 \\ 0 & 0 & -1 & -1 \\ 0 & -1 & 1 & 0 \end{pmatrix} \begin{pmatrix} \Delta x_1 \\ \Delta x_2 \\ \Delta s \\ \Delta z \end{pmatrix} = \begin{pmatrix} 36 \\ -9 \\ 1 + \mu \\ 0 \end{pmatrix}$$

Interior point method – Example



Sequential Quadratic Programming

Nocedal and Wright (2006): Let $f, h_i : \mathbb{R}^n \to \mathbb{R}$ be smooth functions,

$$\begin{aligned} &\min_x \ f(x) \\ &\text{s.t. } h_i(x) = 0, \ i = 1, \dots, l. \end{aligned} \tag{15}$$

Lagrange function

$$L(x, v) = f(x) + \sum_{i=1}^{l} v_i h_i(x)$$

and KKT optimality conditions

$$\begin{aligned} \nabla_x L(x, v) &= \nabla_x f(x) + A(x)^T v = 0, \\ h(x) &= 0, \end{aligned} \tag{16}$$

where $h(x)^T = (h_1(x), \dots, h_l(x))$ and
$A(x)^T = [\nabla h_1(x), \nabla h_2(x), \dots, \nabla h_l(x)]$ denotes the Jacobian matrix.

Sequential Quadratic Programming

We have system of $n + l$ equations in the $n + l$ unknowns x and v:

$$\nabla F(x, v) = \begin{bmatrix} \nabla_x f(x) + A(x)^T v \\ h(x) \end{bmatrix} = 0. \tag{17}$$

ASS. (LICQ) Jacobian matrix $A(x)$ has full row rank.
The Jacobian is given by

$$\nabla^2 F(x, v) = \begin{bmatrix} \nabla^2_{xx} L(x, v) & A(x)^T \\ A(x) & 0 \end{bmatrix}. \tag{18}$$

We can use the **Newton algorithm** ..

Sequential Quadratic Programming

Setting $f_k = f(x^k)$, $\nabla^2_{xx} L_k = \nabla^2_{xx} L(x^k, v^k)$, $A_k = A(x^k)$, $h_k = h(x^k)$, we obtain the Newton step by solving the system

$$\begin{bmatrix} \nabla^2_{xx} L_k & A_k^T \\ A_k & 0 \end{bmatrix} \begin{bmatrix} p_x \\ p_v \end{bmatrix} = \begin{bmatrix} -\nabla f_k - A_k^T v_k \\ -h_k \end{bmatrix} \tag{19}$$

Then we set $x^{k+1} = x^k + p_x$ and $v^{k+1} = v^k + p_v$.

Sequential Quadratic Programming

ASS. (SOSC) For all $d \in \{\tilde{d} \neq 0 : \ A(x)\tilde{d} = 0\}$, it holds

$$d^T \nabla^2_{xx} L(x, v) d > 0.$$

Sequential Quadratic Programming

Important alternative way to see the Newton iterations: Consider the **quadratic program**

$$\begin{aligned} &\min_p \; f_k + p^T \nabla f_k + \frac{1}{2} p^T \nabla^2_{xx} L_k p \\ &\text{s.t. } h_k + A_k p = 0. \end{aligned} \tag{20}$$

KKT optimality conditions

$$\begin{aligned} \nabla^2_{xx} L_k p + \nabla f_k + A_k^T \tilde{v} &= 0 \\ h_k + A_k p &= 0, \end{aligned} \tag{21}$$

$$\begin{bmatrix} \nabla^2_{xx} L_k & A_k^T \\ A_k & 0 \end{bmatrix} \begin{bmatrix} p_x \\ p_{\tilde{v}} \end{bmatrix} = \begin{bmatrix} -\nabla f_k \\ -h_k \end{bmatrix} \tag{22}$$

which is the same as the Newton system if we add $A_k^T v_k$ to the first equation. Then we set $x^{k+1} = x^k + p_x$ and $v^{k+1} = p_{\tilde{v}}$.

Sequential Quadratic Programming

Algorithm: Start with an initial solution (x^0, v^0) and iterate until a convergence criterion is met:

1. Evaluate $f_k = f(x^k)$, $h_k = h(x^k)$, $A_k = A(x^k)$, $\nabla^2_{xx} L_k = \nabla^2_{xx} L(x^k, v^k)$.
2. Solve the Newton equations OR the quadratic problem to obtain new (x^{k+1}, v^{k+1}).

If possible, deliver explicit formulas for first and second order derivatives.

Sequential Quadratic Programming

$f, g_j, h_i : \mathbb{R}^n \to \mathbb{R}$ are smooth. We use a **quadratic approximation** of the objective function and linearize the constraints, $p \in \mathbb{R}^p$

$$\begin{aligned} \min_p \; & f(x^k) + p^T \nabla_x f(x^k) + \frac{1}{2} p^T \nabla^2_{xx} L(x^k, u^k, v^k) p \\ \text{s.t.} \; & g_j(x^k) + p^T \nabla_x g_j(x^k) \leq 0, \; j = 1, \ldots, m, \\ & h_i(x^k) + p^T \nabla_x h_i(x^k) = 0, \; i = 1, \ldots, l. \end{aligned} \tag{23}$$

Use an algorithm for quadratic programming to solve the problem and set $x^{k+1} = x^k + p_k$, where u^{k+1}, v^{k+1} are Lagrange multipliers of the quadratic problem which are used to compute new $\nabla^2_{xx} L$.

Convergence: Nocedal and Wright (2006), Theorem 18.1

Literature

- Bazaraa, M.S., Sherali, H.D., and Shetty, C.M. (2006). **Nonlinear programming: theory and algorithms**, Wiley, Singapore, 3rd edition.
- Boyd, S., Vandenberghe, L. (2004). **Convex Optimization**. Cambridge University Press, Cambridge.
- P. Kall, J. Mayer: **Stochastic Linear Programming: Models, Theory, and Computation**. Springer, 2005.
- Nocedal, J., Wright, J.S. (2006). **Numerical optimization**. Springer, New York, 2nd edition.

Zero-sum games of two players

Martin Branda

Charles University in Prague
Faculty of Mathematics and Physics
Department of Probability and Mathematical Statistics

Computational Aspects of Optimization

Zero-sum games of two players

Definition

A triplet $\{X, Y, K\}$ is called a **game of two rational players with zero sum**, if

1. X is a set of strategies of Player 1 (P1),
2. Y is a set of strategies of Player 2 (P2),
3. $K : X \times Y \to \mathbb{R}$ is a payoff function of player 1, i.e. if P1 plays $x \in X$ and P2 plays $y \in Y$, then P1 gets $K(x, y)$ and P2 gets $-K(x, y)$.

Zero-sum games of two players

Definition

For the zero-sum games $\{X, Y, K\}$ we define

- **upper value** of the game $\mathrm{uv}^* = \inf_{y \in Y} \sup_{x \in X} K(x, y)$,
- **lower value** of the game $\mathrm{lv}^* = \sup_{x \in X} \inf_{y \in Y} K(x, y)$,
- **upper price** of the game $\mathrm{up} = \min_{y \in Y} \sup_{x \in X} K(x, y)$,
- **lower price** of the game $\mathrm{lp} = \max_{x \in X} \inf_{y \in Y} K(x, y)$.

If the lower and upper prices exist and it holds $\mathrm{up} = \mathrm{lp}$, then we say that the game has the **price** $p = \mathrm{up} = \mathrm{lp}$.

Upper value can be seen as the lowest payoff of P1, if P1 knows strategy of P2 before his/her move.

Zero-sum games of two players

Definition

We say that

- $\hat{x} \in X$ is an optimal strategy of P1, if $K(\hat{x}, y) \geq \mathrm{lv}^*$ for all $y \in Y$.
- $\hat{y} \in Y$ is an optimal strategy of P2, if $K(x, \hat{y}) \leq \mathrm{uv}^*$ for all $x \in X$.

Zero-sum games of two players

Proposition

For each zero-sum game $\{X, Y, K\}$ the upper and lower value exits and it holds

$$\mathrm{lv}^* \leq \mathrm{uv}^*.$$

For each $\tilde{x} \in X$ and $\tilde{y} \in Y$ it holds

$$\begin{aligned} \inf_{y \in Y} K(\tilde{x}, y) &\leq K(\tilde{x}, \tilde{y}), \\ \sup_{x \in X} \inf_{y \in Y} K(x, y) &\leq \sup_{x \in X} K(x, \tilde{y}), \\ \mathrm{lv}^* = \sup_{x \in X} \inf_{y \in Y} K(x, y) &\leq \inf_{y \in Y} \sup_{x \in X} K(x, y) = \mathrm{uv}^*. \end{aligned} \tag{1}$$

Zero-sum games of two players

Proposition

For each zero-sum game $\{X, Y, K\}$ is holds that

- *There is at least one optimal strategy of P1, if and only if the lower price exists.*
- *There is at least one optimal strategy of P2, if and only if the upper price exists.*

"$\Rightarrow$": Let $\hat{x} \in X$ be an optimal strategy of P1, i.e. $K(\hat{x}, y) \geq \mathrm{lv}^*$ for all $y \in Y$. Then

$$\mathrm{lv}^* \leq \inf_{y \in Y} K(\hat{x}, y) \leq \sup_{x \in X} \inf_{y \in Y} K(x, y) = \mathrm{lv}^*. \tag{2}$$

Thus

$$\mathrm{lv}^* = \inf_{y \in Y} K(\hat{x}, y) = \max_{x \in X} \inf_{y \in Y} K(x, y) = \mathrm{lp}. \tag{3}$$

Zero-sum games of two players

Proposition

Let $\{X, Y, K\}$ be a zero-sum game with X, Y compact and K continuous. Then the upper and lower prices exist.

Zero-sum games of two players

Theorem

A zero-sum game $\{X, Y, K\}$ has a price if and only if the payoff function has a saddle point, i.e. there is a pair[a] $(\hat{x}, \hat{y})$ such that

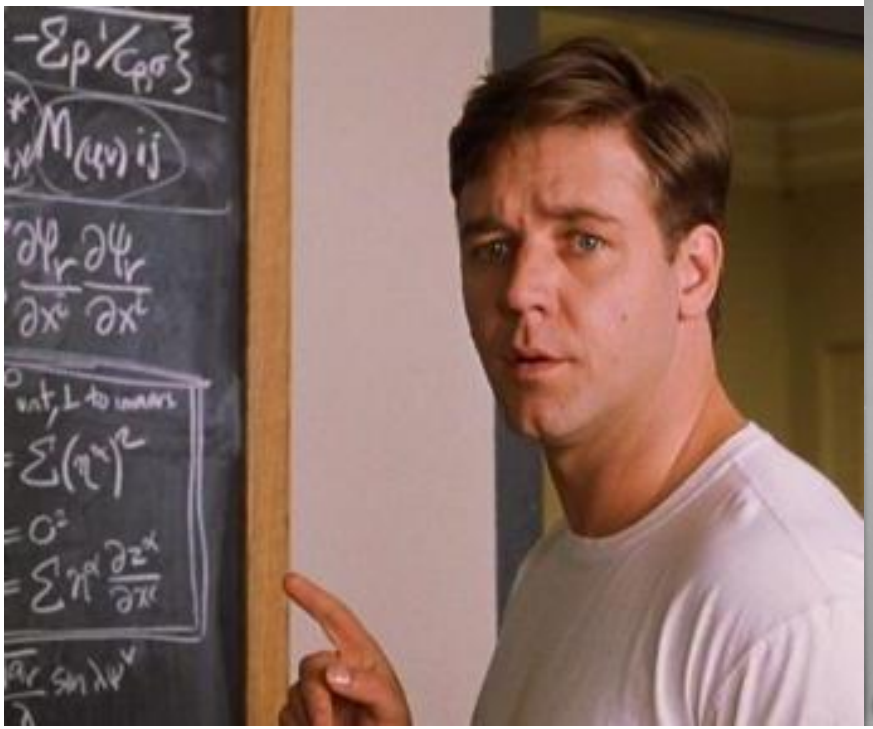
$$K(x, \hat{y}) \leq K(\hat{x}, \hat{y}) \leq K(\hat{x}, y)$$

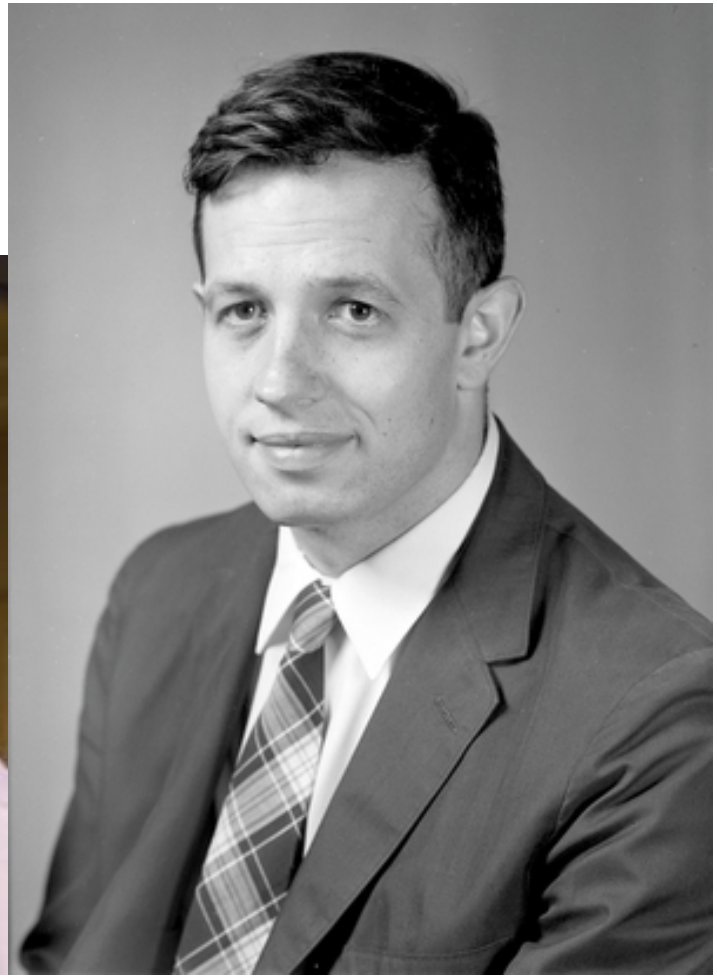
for all $x \in X$ and $y \in Y$. Then $\hat{x}$ is an optimal strategy for P1, $\hat{y}$ is an optimal strategy for P2, and $p = K(\hat{x}, \hat{y})$ is the price of the game.

[a] Such pair can be seen as a Nash equilibrium for two player games.

"$\Rightarrow$": $K(x, \hat{y}) \leq p \leq K(\hat{x}, y)$.

John Forbes Nash (1928–2015)





A Beautiful Mind (2001)

Minimax theorem

Theorem

Let $\{X, Y, K\}$ be a zero-sum game where X, Y are nonempty convex compact sets and $K(x, y)$ is continuous, concave in x and convex in y. Then, there exists the price of the game, i.e.

$$\min_{y \in Y} \max_{x \in X} K(x, y) = \max_{x \in X} \min_{y \in Y} K(x, y).$$

Applicable also out of the game theory, e.g. in robustness.

Generalizations: Rockafellar (1970)

Matrix games

Definition

We say that $\{X, Y, A\}$ is a **matrix game** if it a zero sum game (of two players), $A \in \mathbb{R}^{n \times m}$ is a matrix, and

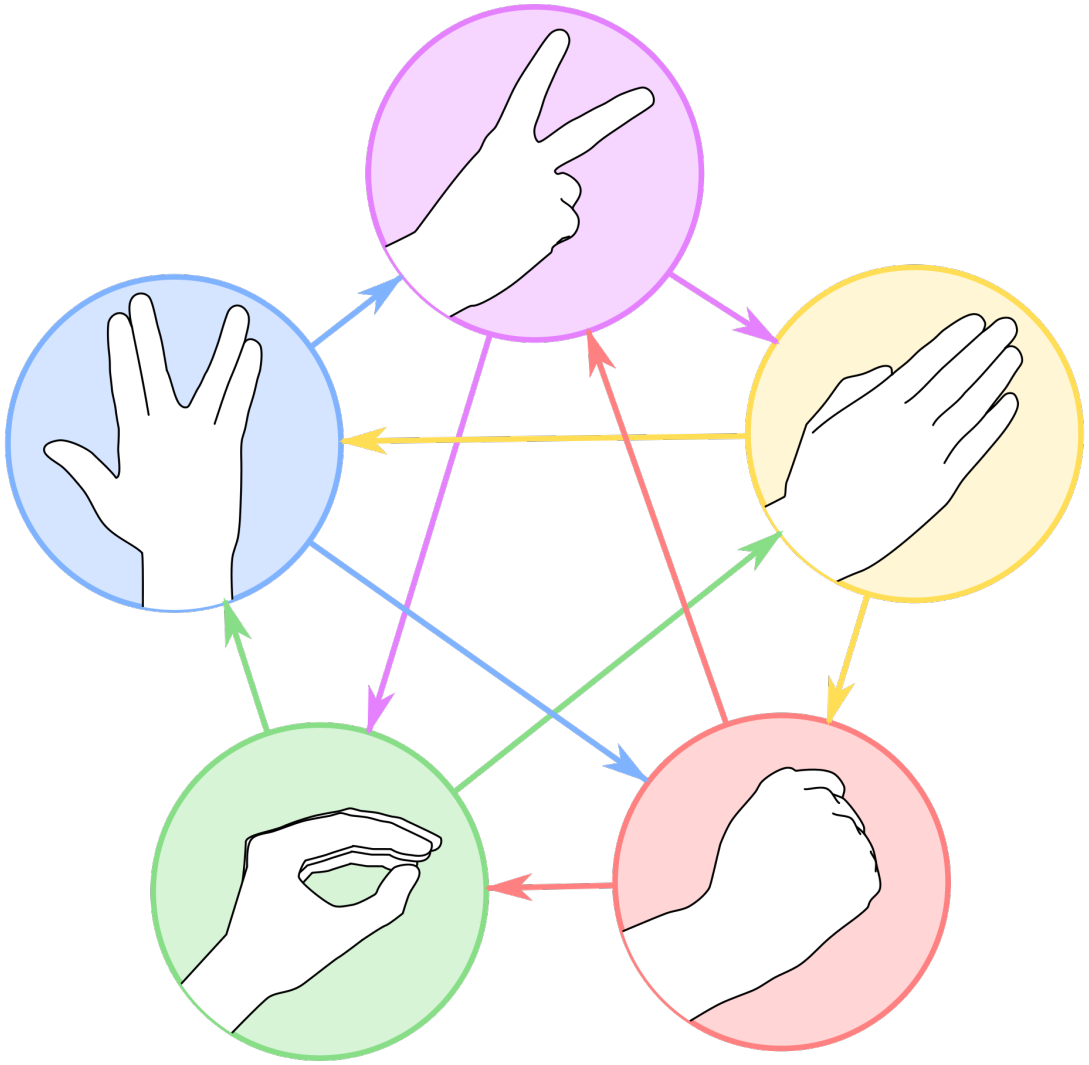
$$
\begin{aligned}
K(x, y) &= x^T A y, \\
X &= \left\{ x \in \mathbb{R}^n : \sum_{i=1}^{n} x_i = 1, \ x_i \geq 0 \right\}, \\
Y &= \left\{ y \in \mathbb{R}^m : \sum_{j=1}^{m} y_j = 1, \ y_j \geq 0 \right\}.
\end{aligned}
\tag{4}
$$

Rock–paper–scissors

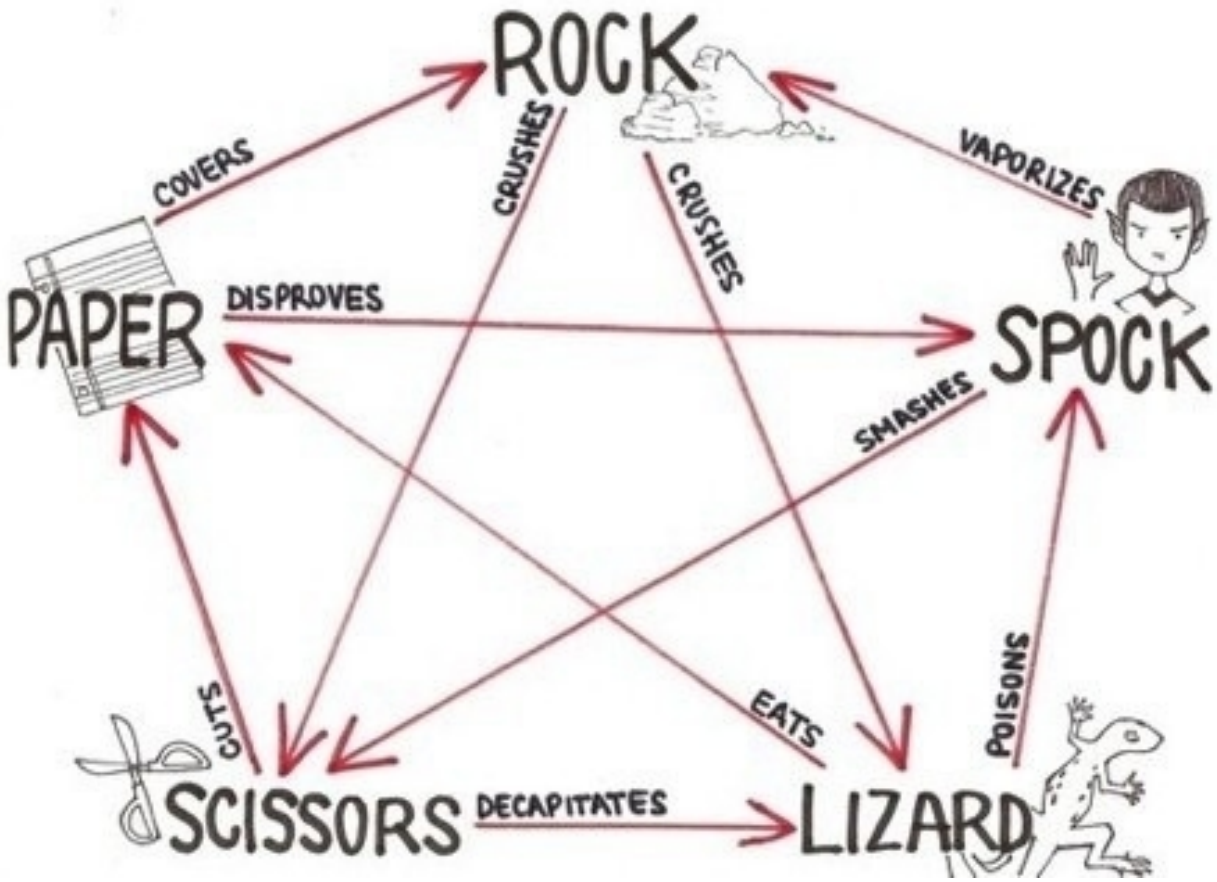
R–P–S

$$A = \begin{pmatrix} 0 & -1 & 1 \\ 1 & 0 & -1 \\ -1 & 1 & 0 \end{pmatrix} \tag{5}$$

Rock–paper–scissors–lizard–Spock



Rock–paper–scissors–lizard–Spock



Matrix games

Definition

For a matrix game $\{X, Y, A\}$, we define a matrix game with **pure strategies** $\{\overline{X}, \overline{Y}, A\}$, where

$$\begin{aligned} \overline{X} &= \left\{ x \in \mathbb{R}^n : \sum_{i=1}^{n} x_i = 1,\ x_i \in \{0,1\} \right\}, \\ \overline{Y} &= \left\{ y \in \mathbb{R}^m : \sum_{j=1}^{m} y_j = 1,\ y_j \in \{0,1\} \right\}. \end{aligned} \tag{6}$$

We say that $\{X, Y, A\}$ has a **price in pure strategies** if both players have optimal pure strategies.

Matrix games

Proposition

Each matrix game has a price and both players have optimal strategies.

Proposition

Matrix game $\{X, Y, A\}$ *has a price in pure strategies if and only if* $\{\overline{X}, \overline{Y}, A\}$ *has a price.*

Matrix games

Proposition

Let $\{X, Y, A\}$ be a matrix game and $\hat{x} \in X$ and $\hat{y} \in Y$ with price p. Then

1. *$\hat{x}$ is an optimal strategy of P1 if and only if $\hat{x}^T A \geq (p, \ldots, p)$,*
2. *$\hat{y}$ is an optimal strategy of P2 if and only if $A\hat{y} \leq (p, \ldots, p)^T$.*

$\hat{x}^T A \geq (p, \ldots, p) \;\Leftrightarrow\; \hat{x}^T A y \geq p, \forall y \in Y.$
(“$\Rightarrow$” $\cdot y$ & $\sum_i y_i = 1$, “$\Leftarrow$” $y = \mathrm{e}_i$)

Matrix games

Proposition

(Complementarity conditions) Let $\{X, Y, A\}$ *be a matrix game with price* p *and let* $\hat{x} \in X$ *and* $\hat{y} \in Y$ *be optimal strategies. Then*

1. *if* $\hat{x}_i > 0$, *then* $(A\hat{y})_i = p$
2. *if* $\hat{y}_j > 0$, *then* $(\hat{x}^T A)_j = p$.

Matrix games – Example

Consider

$$A = \begin{pmatrix} 5 & 1 \\ 0 & 7 \end{pmatrix}$$

$5x_1 \geq p$, $x_1 + 7x_2 \geq p$, $x_1 + x_2 = 1$, $x_1 \geq 0$, $x_2 \geq 0$

$$\max_{x \in X} \min\{5x_1, x_1 + 7x_2\} = p$$

and using $x_1 + x_2 = 1$

$$\max_{x_1 \geq 0} \min\{5x_1, 7 - 6x_1\} = p$$

Maximum is attained at $\hat{x}_1 = 7/11$, $\hat{x}_2 = 4/11$ with the price $p = 35/11$.
Using complementarity conditions, we obtain $\hat{y}_1 = 6/11$, $\hat{y}_2 = 5/11$.

Matrix games

Let $a, b \in \mathbb{R}^n$. We say that a strictly dominates b (b is strictly dominated by a), if $a_i > b_i$ for all $i = 1, \ldots, n$.

Proposition

Let $\{X, Y, A\}$ be a matrix game.

1. *If a row $A_{k,\cdot}$ is strictly dominated by a convex combination of other rows, then each optimal strategy of P1 fulfills $\hat{x}_k = 0$.*
2. *If a column $A_{\cdot,k}$ strictly dominates a convex combination of other columns, then each optimal strategy of P2 fulfills $\hat{y}_k = 0$.*

Matrix games

$$\begin{pmatrix} 3 & 2 & 4 & 0 \\ 3 & 4 & 2 & 3 \\ 6 & 5 & 5 & 1 \\ 1 & 4 & 0 & 7 \end{pmatrix}$$

Show that $(0, 0, 7/11, 4/11)^T$ is optimal strategy for P1, $(0, 0, 6/11, 5/11)^T$ for P2, and the price is $p = 35/11$.

Matrix games

Proposition

Matrix game $\{X, Y, A\}$ has a price p in pure strategies if and only if matrix A has a saddle point , i.e. there is a pair of indices $\{k, l\}$ such that

$$A_{kl} = \min\{A_{kj} : \ j = 1, \dots, m\} = \max\{A_{il} : \ i = 1, \dots, n\}.$$

(minimum in the row, maximum in the column)

e_k,e_l are optimal strategies of P1, P2

$\Leftrightarrow$

$$\begin{aligned} (\mathrm{e}_k^T A)_j &= A_{kj} \geq p, \forall j, \\ (A\mathrm{e}_l)_i &= A_{il} \leq p, \forall i, \end{aligned} \tag{7}$$

$\Leftrightarrow$

$$A_{kl} = \min\{A_{kj} : \ j = 1, \dots, m\} = \max\{A_{il} : \ i = 1, \dots, n\}.$$

Matrix games – Example

Find the saddle point(s) ..

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}, \begin{pmatrix} 2 & 2 & 2 \\ 2 & 1 & 1 \\ 3 & 2 & 2 \end{pmatrix}, \tag{8}$$

Literature

- Lachout, P. (2011). **Matematické programování**. Skripta k (zaniklé) přednášce Optimalizace I (IN CZECH).
- Rockafellar, R.T. (1970). **Convex Analysis**. Princeton University Press, Princeton (N.Y.).
- Webb, J.N. Game Theory (2007). **Decisions, Interactions and Evolution**. Springer-Verlag, London.

Dynamic programming in discrete time – examples

Martin Branda

Charles University in Prague
Faculty of Mathematics and Physics
Department of Probability and Mathematical Statistics

Computational Aspects of Optimization

Dynamic programming

Finite T or infinite ∞ time horizon

$$\max_{A_t, c_t} \sum_{t=1}^{T \vee \infty} \left(\frac{1}{1+i} \right)^{t-1} u(c_t) \tag{1}$$

$$\text{s.t.}$$

$$A_t = (1+r)A_{t-1} + Y_t - c_t.$$

- u – **utility function**
- A_t – **state variables** representing total amount of resources available to the consumer.
- c_t – **control variables** maximizing the consumer's utility. It affects the resources available in the next period.
- Y_t – **exogenous income**
- $1/(1+i)$ – discount factor, r – exogenous interest rate

Dynamic programming

If we assume that there is a finite terminal period T:

$$
\begin{aligned}
V_1(A_0) &= \max_{A_t, c_t} \sum_{t=1}^{T} \left(\frac{1}{1+i}\right)^{t-1} u(c_t) \\
&= \max_{A_t, c_t} u(c_1) + \frac{1}{1+i} u(c_2) + \cdots + \left(\frac{1}{1+i}\right)^{T-1} u(c_T) \\
&= \max_{A_t, c_t} u(c_1) + \frac{1}{1+i} \left[\sum_{t=2}^{T} \left(\frac{1}{1+i}\right)^{t-2} u(c_t)\right] \\
&\quad \text{s.t.} \\
&\quad A_t = (1+r)A_{t-1} + Y_t - c_t.
\end{aligned}
$$

Dynamic programming

We rewrite the maximization problem recursively and obtain the **Bellman equation**

$$V_t(A_{t-1}) = \max_{A_t, c_t} u(c_t) + \frac{1}{1+i} V_{t+1}(A_t),$$

where $A_t = (1+r)A_{t-1} + Y_t - c_t$.

Moreover, since u does not depend on the time period, we can write

$$\begin{aligned} V(A_{t-1}) &= \max_{A_t, c_t} u(c_t) + \frac{1}{1+i} V(A_t), \\ &= \max_{c_t} u(c_t) + \frac{1}{1+i} V\Big((1+r)A_{t-1} + Y_t - c_t\Big), \end{aligned}$$

with $V_{T+1}(A_T) = V(A_T) \equiv 0$.

Dynamic programming

First order optimality conditions

$$\begin{aligned}\frac{\partial V(A_{t-1})}{\partial c_t} &= 0,\\ \frac{\partial V(A_{t-1})}{\partial A_{t-1}} &= 0.\end{aligned}$$

In particular,

$$\begin{aligned}\frac{\partial V(A_{t-1})}{\partial c_t} &= u'(c_t) + \frac{1}{1+i} V'(A_t) \frac{\partial A_t}{\partial c_t},\\ \frac{\partial V(A_{t-1})}{\partial A_{t-1}} &= \frac{1}{1+i} V'(A_t) \frac{\partial A_t}{\partial A_{t-1}},\end{aligned}$$

where using $A_t = (1+r)A_{t-1} + Y_t - c_t$ we have

$$\frac{\partial A_t}{\partial c_t} = -1, \quad \frac{\partial A_t}{\partial A_{t-1}} = 1 + r.$$

Example: Cake eating problem

Cake eating problem:

- $u(c) = 2c^{1/2}$,
- $\Pi_0 = 1$, $\Pi_T = 0$,
- $\Pi_t = \Pi_{t-1} - c_t \ \ldots$

Example: Cake eating problem

Bellman equation

$$V(\Pi_{t-1}) = \max_{c_t} u(c_t) + \frac{1}{1+i} V(\Pi_t),$$
$$\text{s.t. } \Pi_t = \Pi_{t-1} - c_t.$$

Optimality conditions

$$\begin{aligned} \frac{\partial V(\Pi_{t-1})}{\partial c_t} &= u'(c_t) + \frac{1}{1+i} V'(\Pi_t) \frac{\partial \Pi_t}{\partial c_t} = 0, \\ \frac{\partial V(\Pi_{t-1})}{\partial \Pi_{t-1}} &= \frac{1}{1+i} V'(\Pi_t) \frac{\partial \Pi_t}{\partial \Pi_{t-1}} = 0, \end{aligned}$$

From $\Pi_t = \Pi_{t-1} - c_t$

$$\frac{\partial \Pi_t}{\partial c_t} = -1, \ \frac{\partial \Pi_t}{\partial \Pi_{t-1}} = 1. \tag{2}$$

Example: Cake eating problem

Putting them together, we obtain

$$\frac{\partial V(\Pi_{t-1})}{\partial c_t} = u'(c_t) - \frac{1}{1+i} V'(\Pi_t) = 0, \tag{3}$$

$$\frac{\partial V(\Pi_{t-1})}{\partial \Pi_{t-1}} = \frac{1}{1+i} V'(\Pi_t) = 0, \tag{4}$$

Taking (3) for $t-1$

$$u'(c_{t-1}) - \frac{1}{1+i} V'(\Pi_{t-1}) = 0, \tag{5}$$

and plugging it into (4), we have

$$u'(c_{t-1}) = \frac{1}{1+i} u'(c_t),$$

which represents the **optimal path of the cake consumption**.

Example: Cake eating problem

For $u(c) = 2c^{1/2}$, we have

$$\begin{aligned} u'(c_{t-1}) &= \frac{1}{1+i} u'(c_t), \\ (c_{t-1})^{-1/2} &= \frac{1}{1+i} (c_t)^{-1/2}, \\ c_t &= \left(\frac{1}{1+i}\right)^2 c_{t-1}, \end{aligned}$$

with initial and terminal conditions $\Pi_0 = 1$, $\Pi_T = 0$. If we denote $\beta = 1/(1+i)^2$, we obtain

$$c_t = \beta c_{t-1} = \beta^{t-1} c_1.$$

Using

$$c_t = \beta c_{t-1} = \beta^{t-1} c_1.$$

and

$$\Pi_0 - c_1 - c_2 - \ldots - c_T = \Pi_T = 0,$$

we have

$$(1 - \beta - \ldots - \beta^{T-1}) c_1 = \Pi_0,$$

and finally **optimal consumption**

$$\begin{aligned} \hat{c}_1 &= \frac{1-\beta}{1-\beta^T} \Pi_0, \\ \hat{c}_t &= \beta \hat{c}_{t-1} = \beta^{t-1} \hat{c}_1. \end{aligned}$$

Literature

- M. C. Sunny Wong: **Dynamic Optimization: An Introduction**, Lecture Notes – University of Houston, 2013.