Applied Numerical Optimization

Prof. Alexander Mitsos, Ph.D.

Quadratic Programs (QP)

Aachener Verfahrenstechnik
RWTH AACHEN UNIVERSITY

Quadratic Programming (QP)

- An optimization problem with quadratic objective function and linear constraints is a **Quadratic Program (QP)**.
- These problems are very important. They appear as sub-problems in solution methods for general NLPs and in applications, e.g., control. We will not cover solution methods for QP in the class.
- The general form of QP is as follows:

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \frac{1}{2}\boldsymbol{x}^T \boldsymbol{G}\boldsymbol{x} + \boldsymbol{d}^T \boldsymbol{x} \\ \text{s.t.} \quad & \boldsymbol{a}_i^T \boldsymbol{x} - b_i = 0, i \in E \\ & \boldsymbol{a}_i^T \boldsymbol{x} - b_i \leq 0, i \in I \end{aligned}$$

$\boldsymbol{a}_i$ is vector, the ith row of a matrix $\boldsymbol{A}$

- $\boldsymbol{G}$ is a symmetric $(n \text{ x } n)$ matrix.
- If $\boldsymbol{G}$ is positive semi-definite, then QP is convex. The QP is (typically) not convex, if $\boldsymbol{G}$ is indefinite.
- Quadratically-Constrained Quadratic Program (QCQP) have both quadratic objective and constraints, and are much harder to solve.

KKT Conditions of Optimality for QPs

- General problem: $\min_{x \in R^n} f(\boldsymbol{x})$

  s.t. $c_i(\boldsymbol{x}) = 0, i \in E$

  $c_i(\boldsymbol{x}) \leq 0, i \in I$

  QP: $\min_{\boldsymbol{x}} \frac{1}{2}\boldsymbol{x}^T \boldsymbol{G}\boldsymbol{x} + \boldsymbol{d}^T \boldsymbol{x}$

  s.t. $\boldsymbol{a}_i^T \boldsymbol{x} - b_i = 0, i \in E$

  $\boldsymbol{a}_i^T \boldsymbol{x} - b_i \leq 0, i \in I$

- Lagrange function:

$$L(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + \sum_{i \in E \cup I} \lambda_i c_i(\boldsymbol{x})$$

$$L(\boldsymbol{x}, \boldsymbol{\lambda}) = \frac{1}{2}\boldsymbol{x}^T \boldsymbol{G}\boldsymbol{x} + \boldsymbol{d}^T \boldsymbol{x} + \sum_{i \in E \cup I} \lambda_i \left(\boldsymbol{a}_i^T \boldsymbol{x} - b_i\right)$$

- KKT conditions:

$$\boldsymbol{\nabla}_{\boldsymbol{x}} L(\boldsymbol{x}^*, \lambda^*) = \boldsymbol{0}$$
$$c_i(\boldsymbol{x}^*) = 0, \forall\, i \in E$$
$$c_i(\boldsymbol{x}^*) \leq 0, \forall\, i \in I$$
$$\lambda_i^* \geq 0, \forall\, i \in I$$
$$\lambda_i^* c_i(\boldsymbol{x}^*) = 0, \forall\, i \in I$$

$\Rightarrow$

$$\boldsymbol{G}\boldsymbol{x}^* + \boldsymbol{d}^T + \boldsymbol{A}^T \boldsymbol{\lambda}^* = \boldsymbol{0}$$
$$\boldsymbol{a}_i^T \boldsymbol{x} - b_i = 0, \forall\, i \in E$$
$$\boldsymbol{a}_i^T \boldsymbol{x} - b_i \leq 0, \forall i \in I$$
$$\lambda_i^* \geq 0, \forall\, i \in I$$
$$\lambda_i^* \left(\boldsymbol{a}_i^T \boldsymbol{x} - b_i\right) = 0, \forall\, i \in I$$

Nonlinear equations!
Bilinear as in LP
All other are linear as in LP

Solution of (convex) QPs

- Convex QPs are very similar to LPs
  - KKT conditions are necessary and sufficient for global optimality
  - Linear stationarity, linear primal feasibility, nonlinear complementarity slackness, linear bounds on variables
  - So overall we have the same choice: active set vs. interior point methods
  - Algorithms are similar

- Nonconvex QPs are hard to solve
  - KKT conditions are only necessary, not sufficient for optimality
  - Algorithms for global optimization of nonconvex NLPs are applicable
  - Special algorithms exist but understanding them requires studying the linear algebra carefully

→We will skip algorithms for QPs

Check Yourself

- What is the standard form of a QP? When is the QP convex?
- Write the optimality conditions for a QP.
- What did we learn about solution of QPs?

Applied Numerical Optimization

Prof. Alexander Mitsos, Ph.D.

Constrained optimization: strategies, elimination, and solver choice

Nonlinear Optimization Problem (Nonlinear Program, NLP)

$$\min_{\boldsymbol{x} \in R^n} f(\boldsymbol{x})$$

$$\text{s.t.} \quad c_i(\boldsymbol{x}) = 0, i \in E$$

$$c_i(\boldsymbol{x}) \leq 0, i \in I$$

$\boldsymbol{x} = [x_1, x_2, \dots, x_n]^T \in D = R^n$ a vector (point in *n*-dimensional space)

D **host set**

$f : D \rightarrow R$ **objective function**

$c_i : D \rightarrow R$ constraint functions $\forall\, i \in E \cup I$

E the index set of **equality constraints**

I the index sets of **inequality constraints**

- Three solution strategies
  - **Elimination of variables** (to convert to unconstrained problem)
  - Approximation as series of unconstrained problems
  - Approximation as series of simpler constrained problems

Elimination of Variables: Idea

$$\min_{x \in R^n} \; f(\boldsymbol{x})$$
$$\text{s.t.} \quad c_i(\boldsymbol{x}) = 0, i \in E$$

- n variables and m equalities $\Rightarrow n - m$ degrees of freedom for the optimization.
- Idea: simplify the problem by eliminating m variables using equalities.

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{z} \end{bmatrix} \begin{array}{l} \longleftarrow \text{dimension } n - m \\ \longleftarrow \text{dimension } m \end{array} \quad \Longrightarrow \quad \min_{\boldsymbol{y}} \; \tilde{f}(\boldsymbol{y})$$

Remarks

- Need to be able to solve the functions $c_i(\boldsymbol{x})$ for $\boldsymbol{z}(\boldsymbol{y})$.
  - Elimination can be symbolic or numeric.
  - Possible for linear and some nonlinear equalities.
  - Sometimes called “reduced-space formulation”.

Elimination of Variables: Example

$$\min_{\boldsymbol{x} \in R^2} \ f = 4x_1 + 5x_2^2$$

$$\text{s.t.} \quad \sqrt{x_1} + x_2 = 3$$

Solve for x_1 and insert into objective function:

$$x_1 = (3 - x_2)^2$$

$$\min_{x_2} \ \tilde{f} = 9x_2^2 - 24x_2 + 36$$

Solve the unconstrained problem:

$$\left.\frac{d\tilde{f}}{dx_2}\right|_{x_2} = 18x_2 - 24 \qquad \Rightarrow \qquad x_2^* = 4/3$$

$$\left.\frac{d^2\tilde{f}}{dx_2^2}\right|_{x_2} = 18 > 0$$

Strictly convex problem: stationary point is global minimum
Original problem appeared nonconvex!

How to Choose a Solver

- Many fundamental choices
  - Direct vs indirect.
  - Interior-point vs. active set.
  - Approximation order, e.g., first order (steepest descent), second order (Newton's method).
  - Sequence of constrained or unconstrained problems.
  - Full-space or reduced space?
  - Feasible or infeasible iterates?
- We care about: robustness, finding good local minimum, low CPU time, acceptable memory
- Many existing solvers, most available under multiple platforms
- Remember arithmetic complexity: CPU time = # iterations * CPU time/iteration
  - Each factor depends on solver and problem structure!

Non-exhaustive List of Local NLP Solvers

Interior point

IPOPT: primal-dual equation, barrier. A. Wächter and L.T. Biegler Math. Prog., 106(1):25–57, 2006. EPL. Open source

KNITRO: primal-dual equation R. H. Byrd, J. Nocedal, and R.A. Waltz, Large-Scale Nonlinear Optimization, pages 35–59. Springer, 2006. Commercial

LOQO: primal-dual with LS Vanderbei, Robert J. Optim. Meth. & Soft. 11.1-4 (1999): 451-484. Commercial

Sequential Quadratic Programs

filterSQP: SQP with trust region and filter method. Fletcher, Roger, and Sven Leyffer. "Nonlinear programming without a penalty function." Math. Prog. 91.2 (2002): 239-269. Commercial

NLPQL: SQP, two merit functions Schittkowski, Klaus. "NLPQL: A FORTRAN subroutine solving constrained nonlinear programming problems." Annals of OR 5.2 (1986): 485-500. Free for academics

SNOPT: BFGS, QP with active set, LS with augmented Lagrangian P. E. Gill, W. Murray and M. A. Saunders., SIAM Review 47 (2005). Commercial

Gradient projection & feasible path

CONOPT: generalized reduced gradient. Drud, A. (1985). Math. Prog., 31(2), 153-191. Commercial

LANCELOT: augmented Lagrangian Conn, A R., N.I.M. Gould, and P. Toint. SIAM J. on Numerical Analysis 28.2 (1991): 545-572. Free for academics

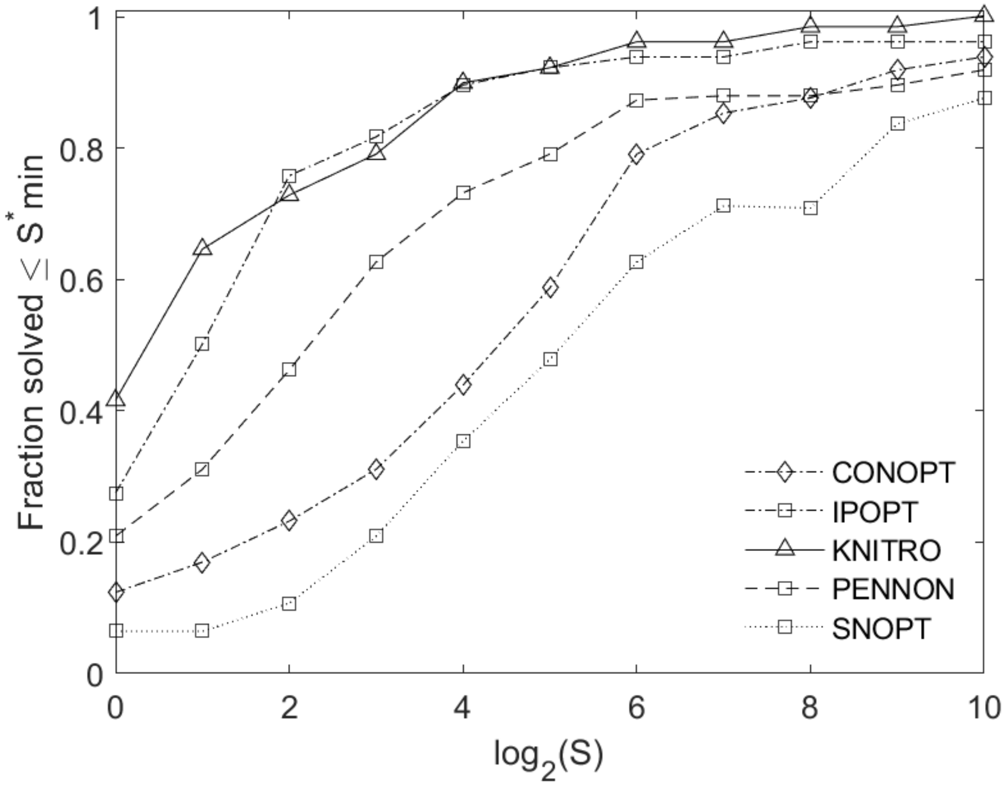
MINOS: Linearly Constrained Augmented Lagrangian. Murtagh, B A., and M. A. Saunders. "Large-scale linearly constrained optimization." Math. Prog. 14.1 (1978): 41-72. Commercial

Quantitative Comparison of Solvers

- Comparison of solvers difficult task:
  - comparison metric?
  - values for tolerances and tuning options?
  - handling of failed instances?

- Metrics used:
  - CPU time
  - scaled CPU time to best solver
  - # function evaluations
  - # iterations

- Dolan-Moré performance profiles: order solvers by #problems solved as function of chosen metric:
  - Cannot safely distinguish between the not-best solvers
  - Chosen problem suite changes ordering
  - Chosen metric changes ordering



Reprinted from Biegler, 2010, p.175 (S = CPU time in min.), data from: *Mittelmann NLP benchmark from July 20, 2009.*

Prof. Mittelmann (http://plato.la.asu.edu/bench.html)

Dolan, Elizabeth D., and Jorge J. Moré. "Benchmarking optimization software with performance profiles." *Mathematical programming* 91.2 (2002): 201-213.

Check Yourself

- Which solution strategies exist for the solution of general NLPs?
- Is the elimination of variables always safe to apply? What are the disadvantages?
- What are performance plots and how can they be used?

Applied Numerical Optimization

Prof. Alexander Mitsos, Ph.D.

Constrained optimization: penalty methods

Aachener Verfahrenstechnik

RWTH AACHEN UNIVERSITY

Penalty and Barrier Methods

- Idea: replace the constrained problem by a sequence of unconstrained optimization problems.

- How to remove constraints?

- **Quadratic Penalty Method (QPM):** replace constraints by adding quadratic penalty to objective.
  - Approximation from infeasible points

- **Augmented Lagrangian Method (ALM):** improve QPM to avoid ill-conditioning by estimating Lagrange parameters

- **Log-Barrier Method (LBM):** use logarithmic barrier to enforce strict satisfaction of inequalities.
  - Approximation from feasible points

Quadratic Penalty Method (QPM) – Equality Constraints

Replace each constraint by a quadratic penalty term in the objective

$$\min_{\boldsymbol{x} \in R^n} \; f(\boldsymbol{x})$$

$$\text{s.t.} \quad c_i(\boldsymbol{x}) \; = 0, \; i \in E$$

Quadratic penalty function: $Q(\boldsymbol{x}; \mu) = f(\boldsymbol{x}) + \frac{1}{2\mu} \sum_{i \in E} [c_i(\boldsymbol{x})]^2$

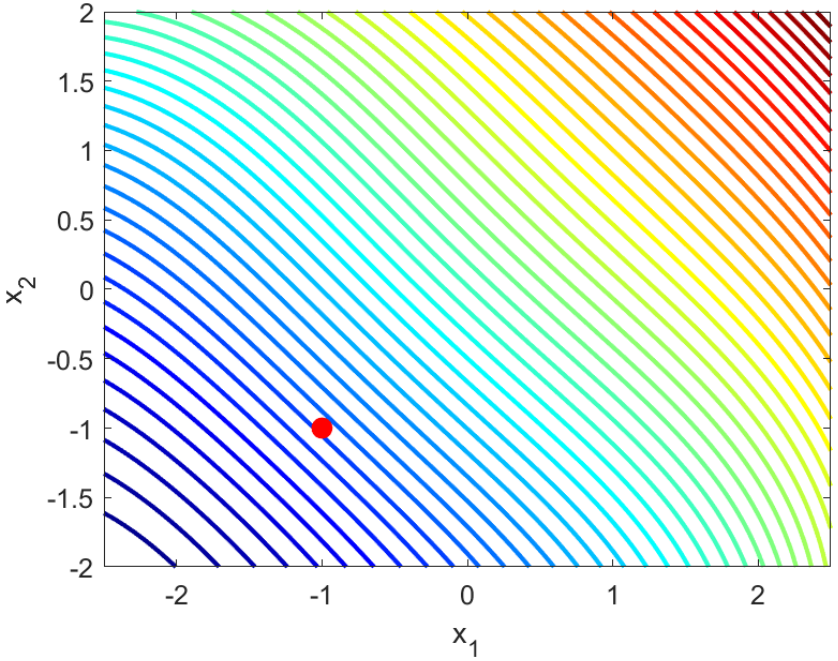
- with penalty parameter $\mu > 0$
- Construct a sequence $\{\mu^{(k)}\}$ with $\mu^{(k)} \to 0$ and minimize $Q\left(\boldsymbol{x}; \mu^{(k)}\right)$.
  - $\boldsymbol{x}^{(k)}$ are infeasible approximate solutions of the original problem.
  - The optimal solution of one step is the initial guess for the next.
- For $\mu \to 0$ the constraint violation is increasingly penalized.
  - The approximation is progressively improved.
  - $\boldsymbol{x}^{(k)}$ converge to a solution, if $Q\left(\boldsymbol{x}; \mu^{(k)}\right)$ are globally minimized.

Example for Quadratic Penalty Method – Contour Plots

$$\min_{\boldsymbol{x}\in R^2} \; x_1 + x_2$$

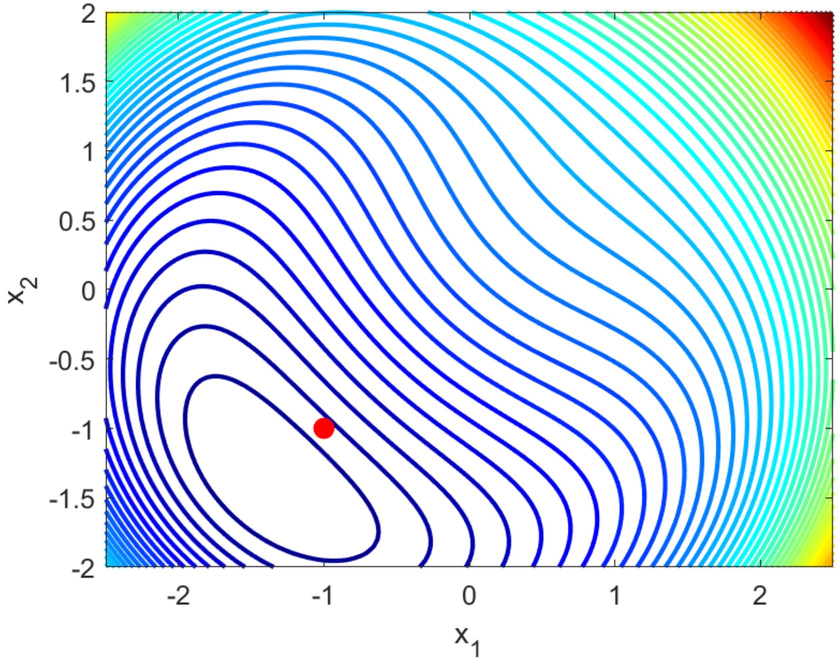
$$\text{s.t.} \;\; x_1^2 + x_2^2 - 2 = 0$$

$\mu = 50$



$$Q(\boldsymbol{x}; 50) = x_1 + x_2 + \frac{1}{100}(x_1^2 + x_2^2 - 2)^2$$

$\mu = 5$



$$Q(\boldsymbol{x}; 5) = x_1 + x_2 + \frac{1}{10}(x_1^2 + x_2^2 - 2)^2$$

Quadratic Penalty Method (QPM) – Include Inequality Constraints

$$
\begin{aligned}
\min_{\boldsymbol{x}\in R^n} \quad & f(\boldsymbol{x}) \\
\text{s.t.} \quad & c_i(\boldsymbol{x}) = 0,\ i \in E \\
& c_i(\boldsymbol{x}) \leq 0,\ i \in I
\end{aligned}
$$

- The sign of inequalities matters:

$$Q(\boldsymbol{x};\mu) = f(\boldsymbol{x}) + \frac{1}{2\mu}\sum_{i\in E}[c_i(\boldsymbol{x})]^2 + \frac{1}{2\mu}\sum_{i\in I}\left[\max(0, c_i(\boldsymbol{x}))\right]^2$$

QPM Algorithm

- Given $\mu^{(1)} > 0$, $\tau^{(1)} > 0$ and an initial point $\boldsymbol{x}^{(0)}$
- for $k = 1,2,\ldots$
  - Use $\boldsymbol{x}^{(k-1)}$ as initial guess. Solve $\boldsymbol{x}^{(k)} \in \operatorname{argmin}_{\boldsymbol{x}} Q\left(\boldsymbol{x};\mu^{(k)}\right)$ approximately: $\left\|\nabla_{\boldsymbol{x}} Q\left(\boldsymbol{x}^{(k)};\mu^{(k)}\right)\right\| < \tau^{(k)}(\mu^{(k)})$
  - IF gradient and constraint violation are sufficiently small, STOP: $\boldsymbol{x}^* = \boldsymbol{x}^{(k)}$
  - ELSE choose $\mu^{(k+1)} \in (0,\mu^{(k)}),\ \tau^{(k+1)}\ (\text{s.t.} \lim_{k\to\infty}\tau^{(k)} = 0)$

Remarks on Quadratic Penalty Method

- The parameter $\mu^{(k)}$ can be chosen adaptively.
  - If $\min_{\boldsymbol{x}} Q(\boldsymbol{x};\mu^{(k)})$ was difficult, decrease μ modestly, e.g., $\mu^{(k+1)} = 0.7\mu^{(k)}$.
  - If $\min_{\boldsymbol{x}} Q(\boldsymbol{x};\mu^{(k)})$ was easy, reduce μ more quickly, e.g., $\mu^{(k+1)} = 0.1\mu^{(k)}$
- For equality constraints, the penalty function is smooth
  - Can use any of the algorithms for unconstrained optimization.
- For inequalities, the penalty function is nonsmooth
  - Continuous first derivative, but discontinuous second derivative.
- As $\mu^{(k)} \to 0$, solving $\min_{\boldsymbol{x}} Q(\boldsymbol{x};\mu^{(k)})$ is increasingly challenging.
  - The Hessian becomes more and more ill-conditioned.
  - The Augmented Lagrangian Method alleviates this issue.

Augmented Lagrangian Method (ALM): Equality Constraints (1)

$$\min_{\boldsymbol{x}\in R^n} \; f(\boldsymbol{x})$$

$$\text{s.t. } c_i(\boldsymbol{x}) = 0, i \in E$$

Lagrangian: $$L(\boldsymbol{x};\boldsymbol{\lambda}) \coloneqq f(\boldsymbol{x}) + \sum_{i\in E} \lambda_i c_i(\boldsymbol{x})$$

Augmented Lagrangian: $$L_A(\boldsymbol{x};\boldsymbol{\lambda};\mu) \coloneqq f(\boldsymbol{x}) + \sum_{i\in E} \lambda_i c_i(\boldsymbol{x}) + \frac{1}{2\mu}\sum_{i\in E}[c_i(\boldsymbol{x})]^2$$

- Advantage of ALM comped to QPM: small constraint violation for relatively large μ
  - Avoid numerical problems of ill-conditioning

- How to iteratively choose parameters μ and $\boldsymbol{\lambda}$?

Augmented Lagrangian Method (ALM): Equality Constraints (2)

- Gradient of the augmented Lagrange function

$$\boldsymbol{\nabla}_{\boldsymbol{x}} L_A(\boldsymbol{x}; \boldsymbol{\lambda}; \mu) = \boldsymbol{\nabla}_{\boldsymbol{x}} f(\boldsymbol{x}) + \sum_{i \in E} \left(\lambda_i + \frac{c_i(\boldsymbol{x})}{\mu} \right) \boldsymbol{\nabla}_{\boldsymbol{x}} c_i(\boldsymbol{x})$$

- $\underset{\boldsymbol{x}}{\operatorname{argmin}} \; L_A\left(\boldsymbol{x}; \boldsymbol{\lambda}^{(k)}; \mu^{(k)}\right)$ should approximate $\underset{\boldsymbol{x} \in \Omega}{\operatorname{argmin}} f(\boldsymbol{x})$

- Stationarity of $L(\boldsymbol{x}; \boldsymbol{\lambda})$ implies $\lambda_i^* \approx \lambda_i^{(k)} + \frac{c_i(\boldsymbol{x}^{(k)})}{\mu^{(k)}}$

- If $\lambda_i^* \approx \lambda_i^{(k)}$, the violation of constraints is small since $c_i\left(\boldsymbol{x}^{(k)}\right) \approx \mu^{(k)} \left(\lambda_i^* - \lambda_i^{(k)}\right)$
  - even for large $\mu^{(k)}$

- As λ_i^* is unknown, we iteratively update: $\lambda_i^{(k+1)} = \lambda_i^{(k)} + \frac{c_i(\boldsymbol{x}^{(k)})}{\mu^{(k)}}$

- Algorithm similar to QPM but converges for larger μ. We expect fewer iterations and better conditioning.

Quadratic Penalty vs. Augmented Lagrangian Method: Example

$$\min_{x_1,x_2} \; [1.5 - x_1(1-x_2)]^2 + [2.25 - x_1(1-x_2^2)]^2 + [2.625 - x_1(1-x_2^3)]^2$$

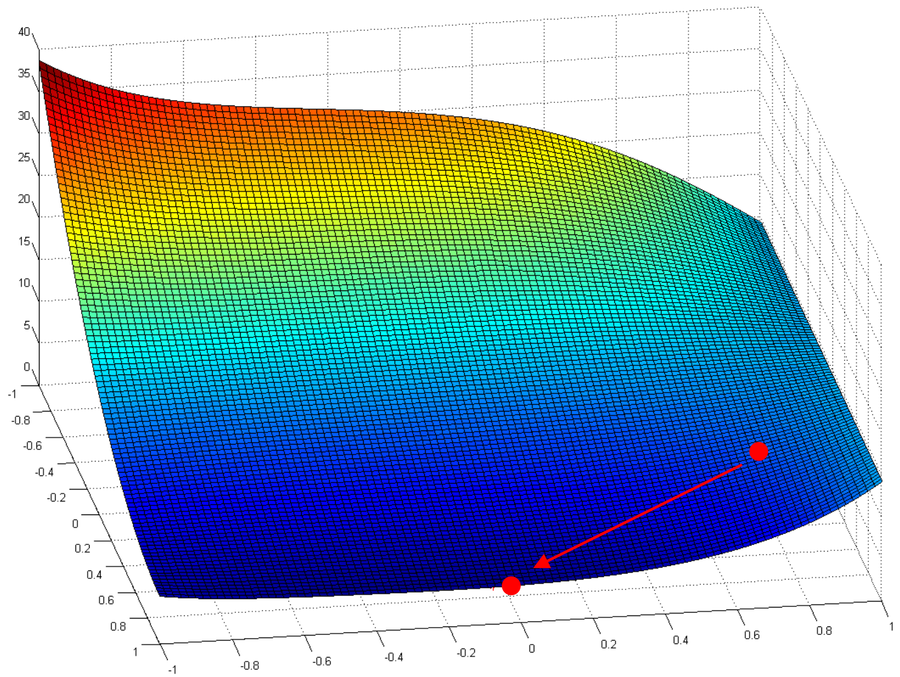
s.t. $\quad x_1^2 + x_2^2 - 1 = 0$

Quadratic Penalty method

- convergence after 39 iterations
- $\mu^{(39)} = 10^{-8}$

Augmented Lagrangian method

- convergence after 28 iterations
- $\mu^{(28)} = 10^{-4}$
- Lagrange multiplier estimates
  $\lambda = 0 \to \cdots \to -2.63 \to -3.33 \to -3.35$



$$\boldsymbol{x}^{(0)} = \frac{\sqrt{2}}{2}\begin{bmatrix}1\\1\end{bmatrix} \qquad \boldsymbol{x}^* = \begin{bmatrix}0.997\\-0.07744\end{bmatrix}$$

$$f(\boldsymbol{x}^*) = 4.42$$

Uri M. Ascher, CPSC 542A Numerical Optimization

Check Yourself

- What is the main idea of the penalty methods?
- Write down the quadratic penalty function
- Describe the quadratic penalty method
- What is the main design idea of the augmented Lagrangian method?

Applied Numerical Optimization

Prof. Alexander Mitsos, Ph.D.

Constrained optimization: barrier method

Aachener Verfahrenstechnik

RWTH AACHEN UNIVERSITY

Penalty and Barrier Methods

- Idea: replace the constrained problem by a sequence of unconstrained optimization problems.
- How to remove constraints?
- **Quadratic Penalty Method (QPM):** replace constraints by adding quadratic penalty to objective.
  - Approximation from infeasible points
- **Augmented Lagrangian Method (ALM):** improve QPM to avoid ill-conditioning by estimating Lagrange parameters
- **Log-Barrier Method (LBM):** use logarithmic barrier to enforce strict satisfaction of inequalities.
  - Approximation from feasible points

Log-Barrier Method (LBM): Inequality Constraints

$$\min_{\boldsymbol{x} \in R^n} \quad f(\boldsymbol{x})$$

$$\text{s.t.} \quad c_i(\boldsymbol{x}) \leq 0, \, i \in I$$

- Replace constraints by a logarithmic barrier term in the objective:

$$P(\boldsymbol{x}; \mu) = f(\boldsymbol{x}) - \mu \sum_{i \in I} \log[-c_i(\boldsymbol{x})]$$

- with the barrier parameter $\mu > 0$.

- The barrier enforces strictly feasible iterates
  - $P(\boldsymbol{x}; \mu) \to \infty$ for $0 > c_i(\boldsymbol{x}) \to 0$. Thus for $\mu > 0$ we enforce $\boldsymbol{c}(\boldsymbol{x}) < \boldsymbol{0}$.

- Similar to QPM, solve sequence of unconstrained problems
  - Solution of one iteration is initial guess of next.
  - As $\mu \to 0$, $\boldsymbol{x}$ the approximations become better.

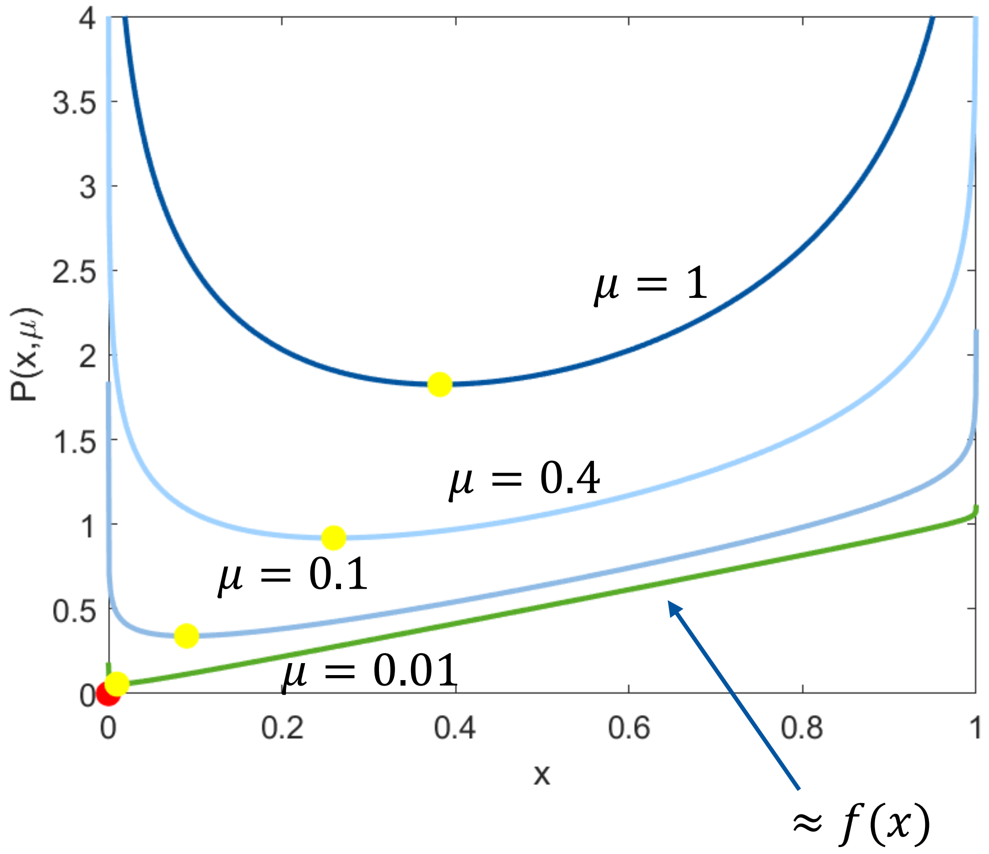
Example Log-Barrier Method: Bound Constrained Univariate Problem

$$\min_{x \in R} \quad x$$

$$\text{s.t. } x \geq 0$$

$$x \leq 1$$

$$P(x;\mu) = x - \mu(\,\log[x] + \log[1-x]\,)$$



Log-Barrier Method (LBM): Equalities and Inequalities

General NLP

$$\min_{x \in R^n} \; f(\boldsymbol{x})$$

$$\text{s.t. } c_i(\boldsymbol{x}) = 0, \; i \in E$$

$$c_i(\boldsymbol{x}) \leq 0, \; i \in I$$

- Replace inequality constraints by a logarithmic barrier
  Replace equality constraint by quadratic penalty

$$B(\boldsymbol{x}; \mu) = f(\boldsymbol{x}) + \frac{1}{2\mu} \sum_{i \in E} [c_i(\boldsymbol{x})]^2 - \mu \sum_{i \in I} \log[-c_i(\boldsymbol{x})]$$

- Similar to inequality-case, solve sequence of unconstrained problems
- Interior-point method w.r.t. inequalities: $c_i\left(\boldsymbol{x}^{(\boldsymbol{k})}\right) < 0, \; i \in I$
- Constraint violation of equalities: $c_i\left(\boldsymbol{x}^{(\boldsymbol{k})}\right) \neq 0, \; i \in E$
  – Equalities have no interior

Check Yourself

- What is the main idea of the barrier methods?
- Write down the log-barrier method.
- What is the main difference between the quadratic penalty method and the logarithmic barrier method?

Applied Numerical Optimization

Prof. Alexander Mitsos, Ph.D.

Constrained optimization: SQP

Aachener Verfahrenstechnik

RWTH AACHEN UNIVERSITY

Nonlinear Optimization Problem (Nonlinear Program, NLP)

$$\min_{\boldsymbol{x} \in R^n} f(\boldsymbol{x})$$

$$\text{s.t.} \quad c_i(\boldsymbol{x}) = 0, i \in E$$

$$c_i(\boldsymbol{x}) \leq 0, i \in I$$

$\boldsymbol{x} = [x_1, x_2, \dots, x_n]^T \in D = R^n$ a vector (point in *n*-dimensional space)

D **host set**

$f : D \rightarrow R$ **objective function**

$c_i : D \rightarrow R$ constraint functions $\forall\, i \in E \cup I$

E the index set of **equality constraints**

I the index sets of **inequality constraints**

- Three solution strategies
  - Elimination of variables (to convert to unconstrained problem)
  - Approximation as series of unconstrained problems
  - **Approximation as series of simpler constrained problems**

Linearly Constrained Lagrangian Method

The linearly constrained Lagrangian (LCL) method is a modification of the augmented Lagrangian method. It is the basis of MINOS.

- In each step, linearize the constraints.

- For the problem

$$\min_{\boldsymbol{x}\in R^n} \; f(\boldsymbol{x})$$
$$\text{s.t.} \;\; c_i(\boldsymbol{x}) = 0, \;\; i \in E$$

in iteration k solve:

$$\min_{\boldsymbol{x}\in R^n} \; F^{(k)}(\boldsymbol{x})$$

$$\text{s.t.} \; \boldsymbol{\nabla} c_i\left(\boldsymbol{x}^{(k)}\right)^T\left(\boldsymbol{x}-\boldsymbol{x}^{(k)}\right) + c_i\left(\boldsymbol{x}^{(k)}\right) = 0, \;\; i \in E$$

- For $F^{(k)}$, often the augmented Lagrangian function is chosen:

$$F^{(k)}(\boldsymbol{x}) = f(\boldsymbol{x}) + \sum_{i\in E} \lambda_i^{(k)} \bar{c}_i^{(k)}(\boldsymbol{x}) + \frac{1}{2\mu}\sum_{i\in E}\left[\bar{c}_i^{(k)}(\boldsymbol{x})\right]^2$$

$$\bar{c}_i^{(k)}(\boldsymbol{x}) = c_i(\boldsymbol{x}) - c_i\left(\boldsymbol{x}^{(k)}\right) - \boldsymbol{\nabla} c_i\left(\boldsymbol{x}^{(k)}\right)^T\left(\boldsymbol{x}-\boldsymbol{x}^{(k)}\right)$$

Sequential Quadratic Programming – SQP

- SQP provides the basis for some good optimization codes.
- We consider

$$\min_{\boldsymbol{x}\in R^n} \ f(\boldsymbol{x})$$

$$\text{s.t.} \ \ c_i(\boldsymbol{x}) = 0, \ \ i \in E$$

- Basic idea: solve sequence $\{k\}$ of QPs, approximating the NLP at iterate $\boldsymbol{x}^{(k)}$ by a QP.
- Simplest choice: Taylor series expansion

$$\min_{\boldsymbol{p}\in R^n} \ \frac{1}{2}\boldsymbol{p}^T\nabla^2 f(\boldsymbol{x}^{(k)})\boldsymbol{p} + \left(\nabla f(\boldsymbol{x}^{(k)})\right)^T\boldsymbol{p}$$

$$\text{s.t.} \ \ \left(\nabla\boldsymbol{c}\left(\boldsymbol{x}^{(k)}\right)\right)^T\boldsymbol{p} + \boldsymbol{c}\left(\boldsymbol{x}^{(k)}\right) = \boldsymbol{0}$$

- Can be interpreted as Newton's method to solve KKT conditions.

Basic SQP Algorithm

- Important points not discussed:
  - approximation of the Hessian matrix (e.g., BFGS-update, in quasi-Newton methods)
  - solution of the Newton-Lagrange equations in each QP step
  - stopping criterion
  - include inequalities
- Basic algorithm
  - Choose $\boldsymbol{x}^{(0)}$
  - for $k = 1,2,\ldots$
    - Calculate $f^{(k)} = f(\boldsymbol{x}^{(k-1)})$, $\boldsymbol{\nabla} f^{(k)} = \boldsymbol{\nabla} f(\boldsymbol{x}^{(k-1)})$, $\boldsymbol{c}^{(k)} = \boldsymbol{c}(\boldsymbol{x}^{(k-1)})$, $\boldsymbol{A}^{(k)} = \boldsymbol{\nabla}\boldsymbol{c}(\boldsymbol{x}^{(k-1)})$, update $\boldsymbol{B}^{(k)}$
    - Solve the QP for $\boldsymbol{p}$
    - Set $\boldsymbol{x}^{(k)} = \boldsymbol{x}^{(k-1)} + \boldsymbol{p}$
    - If the optimality conditions are fulfilled, STOP.

Check Yourself

- Which solution strategies exist for the solution of general NLPs?
- What is the main idea of the SQP method? Which problems have to be solved in each iteration step of the SQP method?

Applied Numerical Optimization

Prof. Alexander Mitsos, Ph.D.

Integer optimization: Introduction and simple example

Aachener Verfahrenstechnik
RWTH AACHEN UNIVERSITY

Mixed-Integer Optimization Problems

Semi-general formulation:

$$\min_{\boldsymbol{x},\boldsymbol{y}} \quad f(\boldsymbol{x}) + \boldsymbol{d}^T\boldsymbol{y}$$
$$\text{s.t.} \quad c_i(\boldsymbol{x}) = 0, \forall\, i \in E$$
$$c_i(\boldsymbol{x}) + \boldsymbol{a}_{y,i}^T\boldsymbol{y} \leq 0, \forall\, i \in I$$

Most general formulation:

$$\min_{\boldsymbol{x},\boldsymbol{y}} \quad f(\boldsymbol{x},\boldsymbol{y})$$
$$\text{s.t.} \quad c_i(\boldsymbol{x},\boldsymbol{y}) = 0, \forall\, i \in E$$
$$c_i(\boldsymbol{x},\boldsymbol{y}) \leq 0, \forall\, i \in I$$

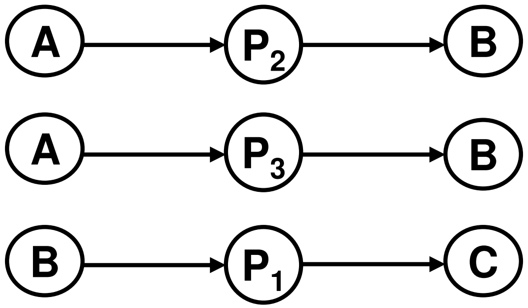
$\boldsymbol{x} \in R^{n_x}$, continuous variables

$\boldsymbol{y} \in Y$, discrete variables (e.g. $\boldsymbol{y} \in \{0,1\}^{n_y}$)

- Classification
  - MIP : Mixed-Integer Programming (typically linear meant)
  - MINLP : Mixed-Integer NLP
  - MILP : Mixed-Integer LP
  - IP : Integer Programming
  - BIP : Binary Integer Programming (also termed “0-1 programming”)

- Formulation matters:
  - Seemingly small differences in problem make huge difference in difficulty
  - “Picking a good formulation is always important. But some clever software does it for you“ – paraphrased from Jeffrey T. Linderoth, MIMOSA, 2015.

Example – Supply Chain Design



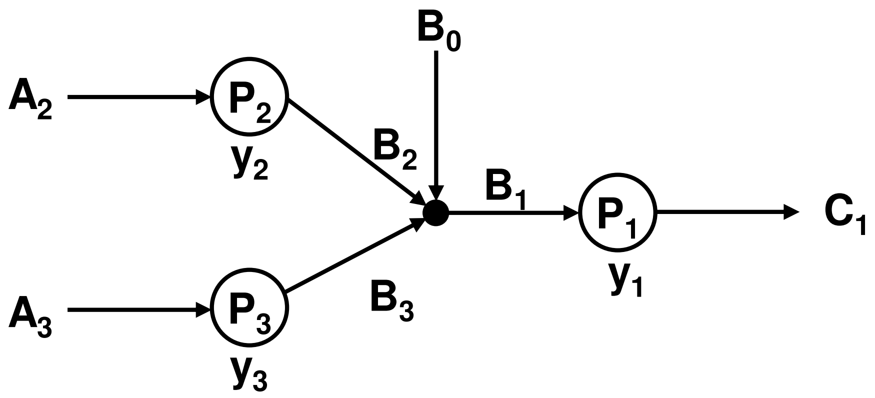
- Product C is manufactured in process P_1 using the intermediate B.
  B can be purchased or manufactured by processes P_2 and/or P_3. Both use A as a raw material.

- P_2 and P_3 have different fixed maximal capacities. The production rates are optimization variables.

- Estimates of fixed and investment costs exist.

Objective: select processes and production rates to maximize profit.
(or minimize cost for fixed production, maximize production for fixed cost, …)

Example – Supply Chain Design: Problem Formulation



- The structure with discrete decision variables $y_j \in \{0,1\}$, where $y_j = 1$ means that the process i exists and $y_j = 0$ the converse.

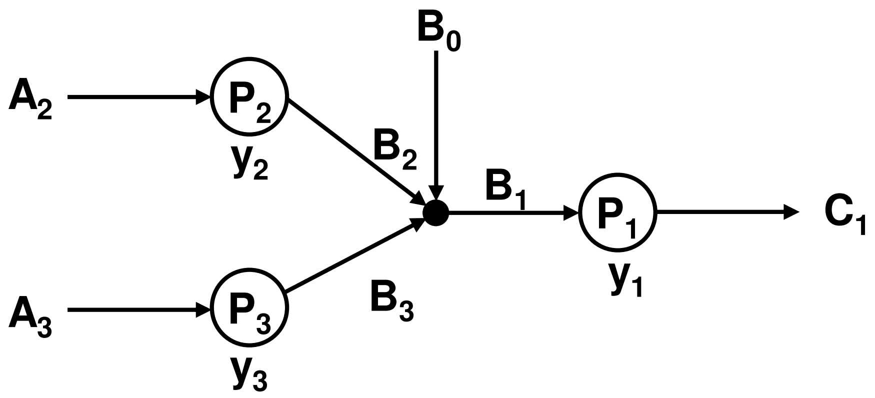
- Constraints
  - process model
    - $C_1 = 0.9\, B_1$
    - $B_2 = \ln(1 + A_2)$
    - $B_3 = 1.2\ln(1 + A_3)$
  - mass balance for B: $B_1 = B_0 + B_2 + B_3$
  - bound on production $C_1 \leq 1$
  - maximum plant capacity
    - $C_1 \leq 2y_1$
    - $B_2 \leq 4y_2$
    - $B_3 \leq 5y_3$
  - nonnegativity conditions
    - $A_i, B_i, C_i \geq 0$

Note: $y_2 = 0 \;\rightarrow B_2 = 0$ by inequality
$\rightarrow A_2 = 0$ by equality

Adding $A_2 \leq My_2$ has advantages

(similar for other variables)

Example – Supply Chain Design: Data



- Objective function
  - material prices
    A2,3: 1.8; B0: 7; C1: 13
  - fixed costs
    P1: 3.5; P2: 1; P3: 1.5
  - operating costs
    P1: 2; P2: 1; P3: 1.2

Profit (objective function):

$$
\begin{aligned}
f(A_i, B_i, C_i, y_i) &= 13C_1 \\
&-(7B_0 + 1.8A_2 + 1.8A_3) \\
&-(3.5y_1 + y_2 + 1.5y_3) \\
&-(2C_1 + B_2 + 1.2B_3)
\end{aligned}
$$

Example – Supply Chain Design: Resulting Mixed-Integer Optimization Problem

Semi-general formulation

$$\min_{\boldsymbol{x},\boldsymbol{y}} f(\boldsymbol{x}) + \boldsymbol{d}^T\boldsymbol{y}$$

$$\text{s.t.} \quad \boldsymbol{c}_i(\boldsymbol{x}) = 0 \quad \forall\, i \in E$$

$$\boldsymbol{c}_i(\boldsymbol{x}) + \boldsymbol{a}_{y,i}^T\boldsymbol{y} \leq 0 \quad \forall\, i \in I$$

$\boldsymbol{x} \in R^{n_x}$ continuous vars.

$\boldsymbol{y} \in Y$ discrete vars.

$(\text{e.g. } \boldsymbol{y} \in \{0,1\}^{n_y})$

Specific problem

$$\max f(A_i, B_i, C_i, y_i) =$$

$$13C_1 - 7B_0 + 1.8A_2 + 1.8A_3 - 3.5y_1 + 2C_1 + y_2 + B_2 + 1.5y_3 + 1.2B_3$$

$$\begin{aligned} \text{s.t.}\quad & C_1 = 0.9\, B_1 \\ & B_2 = \ln(1 + A_2) \\ & B_3 = 1.2 \ln(1 + A_3) \\ & B_1 = B_0 + B_2 + B_3 \\ & A_i, B_i, C_i \geq 0 \\ & C_1 \leq 1 \\ & C_1 \leq 2\, y_1 \\ & B_2 \leq 4\, y_2 \\ & B_3 \leq 5\, y_3 \end{aligned}$$

Check Yourself

- What constitutes a mixed-integer or integer programming problem?
- What are these formulations used for?
- Do you expect mixed-integer programs to be more difficult to solve compared to continuous problems? Why?

Applied Numerical Optimization

Prof. Alexander Mitsos, Ph.D.

MILP: example from systems biology

Aachener Verfahrenstechnik

RWTH AACHEN UNIVERSITY

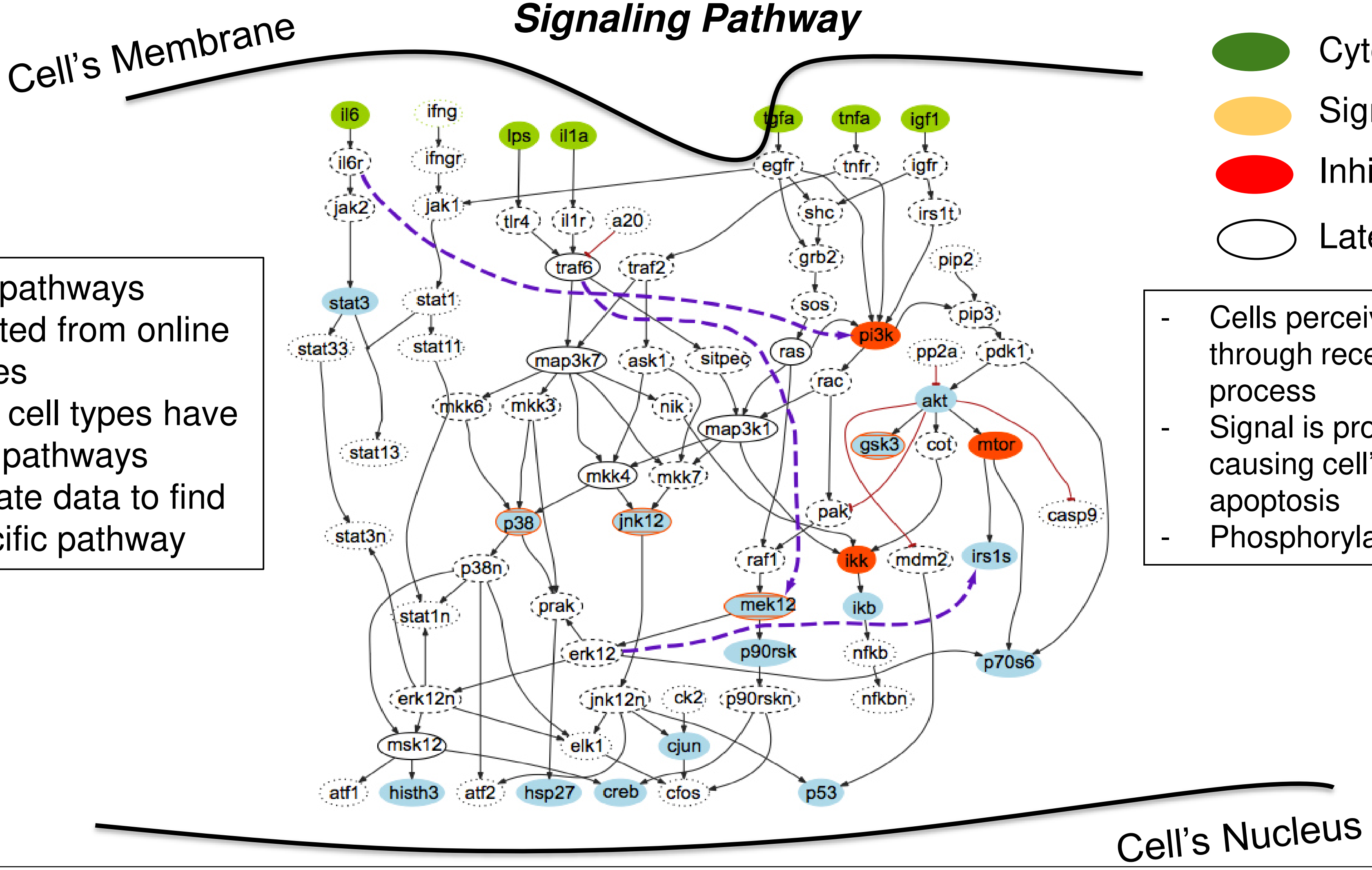
0-1 Mixed-Integer Linear Programs

$$\begin{aligned}
\min_{\boldsymbol{x},\boldsymbol{y}} \; & \boldsymbol{d}_x^T\boldsymbol{x} + \boldsymbol{d}_y^T\boldsymbol{y} \\
\text{s.t.} \; & \boldsymbol{a}_{x,i}^T\boldsymbol{x} + \boldsymbol{a}_{y,i}^T\boldsymbol{y} = b_i, \forall\, i \in \mathrm{E} \\
& \boldsymbol{a}_{x,i}^T\boldsymbol{x} + \boldsymbol{a}_{y,i}^T\boldsymbol{y} = b_i \leq 0, \forall\, i \in I \\
& \boldsymbol{x} \in R^{n_x} \\
& \boldsymbol{y} \in \{0,1\}^{n_y}
\end{aligned}$$

- MILP with finite number of integer realizations can be rewritten in this form

Modeling Signal Transduction

Signaling Pathway



- Cytokines
- Signals
- Inhibited nodes
- Latent nodes

- Generic pathways constructed from online databases
- Different cell types have different pathways
- Incorporate data to find cell-specific pathway

- Cells perceive environment through receptors through the signaling process
- Signal is propagated downstream causing cell's response (proliferation, apoptosis
- Phosphorylation is measured

J. Saez-Rodriguez, *Mol Syst Biol.*, 2009

Aachener Verfahrenstechnik
RWTH AACHEN UNIVERSITY

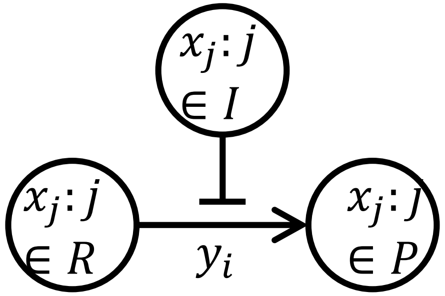
Modeling Signal Transduction: ILP Formulation (1)

Definitions

- Set of reactions: $i = \{1, \ldots, n_r\}$
- Set of species: $j = \{1, \ldots, n_s\}$
- Set of experiments: $k = \{1, \ldots, n_e\}$

Each reaction has three index sets:

- Set of reactants: R_i
- Set of Inhibitors: I_i
- Set of products: P_i



Variables

y_i	$\in \{0,1\},\ i = \{1, \ldots, n_r\}$	is reaction i present?
x_j^k	$\in [0,1],\ j = \{1, \ldots, n_s\};\ k = \{1, \ldots, n_e\}$	is species j formed in experiment k?
$x_j^{k,m}$	$\in [0,1],\ j = \{1, \ldots, n_s\};\ k = \{1, \ldots, n_e\}$	is species j measured in experiment k?
z_i^k	$\in [0,1],\ i = \{1, \ldots, n_r\};\ k = \{1, \ldots, n_e\}$	does reaction i occur in experiment k?

Mitsos, Melas, Siminelakis, Chairakaki, Saez-Rodriguez, Alexopoulos, PLoS Comput Biol 5.12 (2009): e1000591.

Modeling Signal Transduction: ILP Formulation (2)

Objective function: $\min \sum_{j,k} \left| x_j^k - x_j^{k,m} \right| \; \forall\, j = 1, \dots, n_{species}$ and $k = 1, \dots, n_{experiments}$

Secondary objective function: $\min \sum_i y_i \; \forall\, i = 1, \dots, n_{reactions}$

Constraints

$$z_i^k \le y_i, \qquad i = 1, \dots, n_r, k = 1, \dots, n_e$$

$$z_i^k \le x_j^k, \qquad i = 1, \dots, n_r, k = 1, \dots, n_e, j \in R_i$$

$$z_i^k \le 1 - x_j^k, \qquad i = 1, \dots, n_r, k = 1, \dots, n_e, j \in I_i$$

$$z_i^k \ge y_i + \sum_{j \in R_i} \left(x_j^k - 1 \right) - \sum_{j \in I_i} x_j^k, \quad i = 1, \dots, n_r, k = 1, \dots, n_e$$

$$x_j^k \ge z_i^k, \qquad i = 1, \dots, n_r, k = 1, \dots, n_e, j \in P_i$$

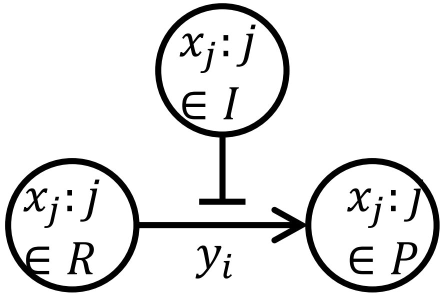
$$x_j^k \le \sum_{i = 1, \dots, n_r, j \in P_i} z_i^k, \qquad j = 1, \dots, n_s, k = 1, \dots, n_e$$

Variables

y_i Reaction i present?

z_i^k Reaction i occurs in experiment k ?

x_j^k Species j formed in experiment k ?

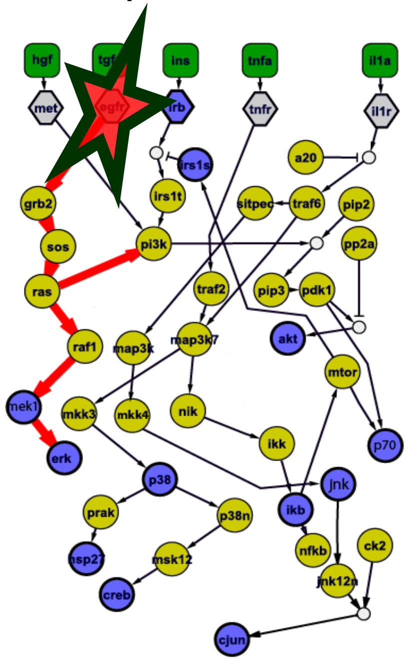


Mitsos, Melas, Siminelakis, Chairakaki, Saez-Rodriguez, Alexopoulos, PLoS Comput Biol 5.12 (2009): e1000591.

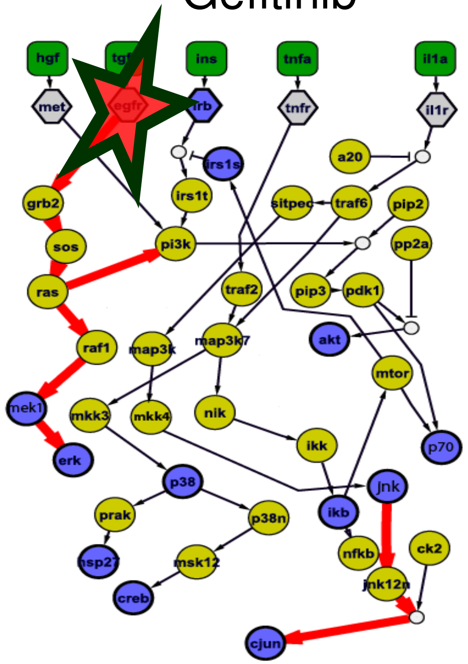
Modeling Signal Transduction: Unbiased Identification of Drug Effects (Liver Cancer)

- Construct a cell type specific pathway
- Train the optimized pathway to data collected under different drugs
- Identify drug induced topology alterations

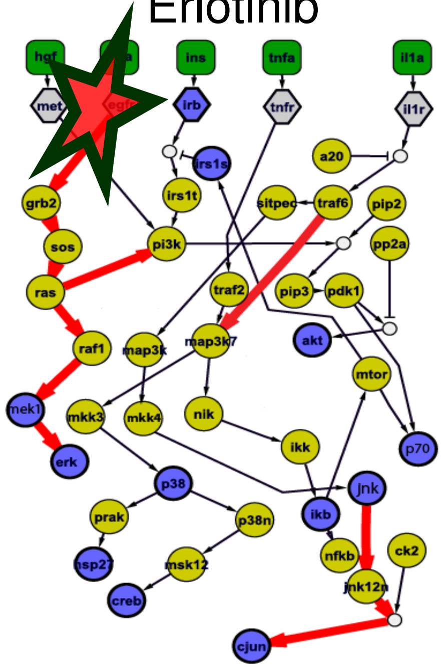
Lapatinib



Gefitinib



Erlotinib



Drug main target: EGFR

Drug effects identified

Alexopoulos, Melas, Chairakaki, Saez-Rodriguez EMBC 2010
bio-IT world 2010: Best practice award 2010

Check Yourself

- Use mixed-integer optimization problems to formulate your problems of interest

Applied Numerical Optimization

Prof. Alexander Mitsos, Ph.D.

Integer optimization: nonconvexity, restrictions and relaxations

Aachener Verfahrenstechnik | RWTH AACHEN UNIVERSITY

Mixed-Integer Optimization Problems

Semi-general formulation:

$$\begin{aligned}\min_{\boldsymbol{x},\boldsymbol{y}} \quad & f(\boldsymbol{x}) + \boldsymbol{d}^T\boldsymbol{y} \\ \text{s.t.} \quad & c_i(\boldsymbol{x}) = 0, \forall\, i \in E \\ & c_i(\boldsymbol{x}) + \boldsymbol{a}_{y,i}^T\boldsymbol{y} \le 0, \forall\, i \in I\end{aligned}$$

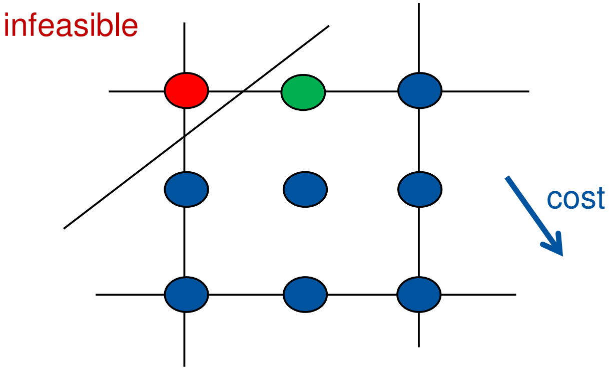
$\boldsymbol{x} \in R^{n_x}$, continuous variables

$\boldsymbol{y} \in Y$, discrete variables (e.g. $\boldsymbol{y} \in \{0,1\}^{n_y}$)

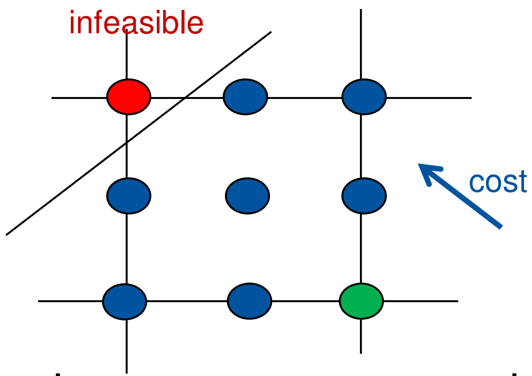
Integer Programs are Nonconvex

- Integrality constraint is nonconvex $\boldsymbol{y} \in \{0,1\}^{n_y}$
  - Recall definition of convex set!
- Optimal solutions of integer program and its continuous relaxation $\boldsymbol{y} \in [0,1]^{n_y}$ may or may not coincide
  - Depends on feasible set and objective function
  - Example: $\boldsymbol{y} \in \{0,1,2\}^2$ with linear constraints and objective

Optimal integer solution **not same** as optimal solution of continuous relaxation



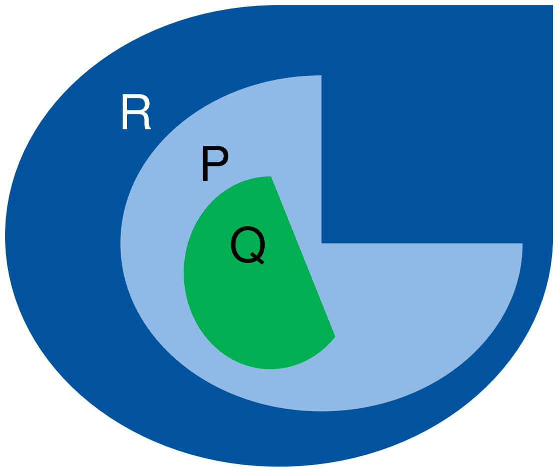
Optimal integer solution **same** as optimal solution of continuous relaxation

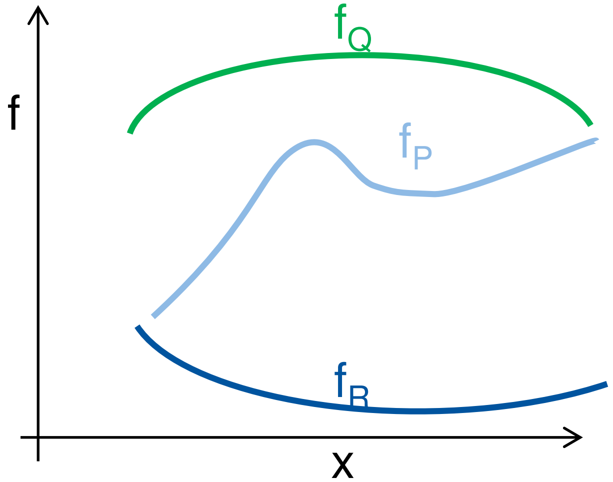


- Distinction in literature: linear functions, convex quadratic functions, nonconvex quadratic, convex nonlinear, nonconvex
  - Often the misnomer “convex mixed-integer program” is used

Restrictions, Relaxations and Approximations

- Consider Problem P, recall Ω the feasible set and f objective function
- (R) is a **relaxation** of (P) if $\Omega_R \supset \Omega_P$ and $\mathrm{f}_R(\boldsymbol{x}) \leq \mathrm{f}_P(\boldsymbol{x}), \forall \boldsymbol{x} \in \Omega_P$.
  - **global** solution of R gives **lower bound** to P
  - e.g. $y_k \in \{0,1\}$ can be relaxed as $y_k \in [0,1]$
  - e.g. linearization of convex function
- (Q) is a **restriction** of (P) if $\Omega_Q \subset \Omega_P$ and $\mathrm{f}_Q(\boldsymbol{x}) \geq \mathrm{f}_P(\boldsymbol{x}), \forall \boldsymbol{x} \in \Omega_Q$.
  - any **feasible** point (e.g. local solution) of Q gives **upper bound** to P
  - e.g., $y_k \in \{0,1\}$ can be restricted to $y_k = 0$
- Approximation can be relaxation, restriction, or neither (e.g. linearization of nonconvex function).





Geoffrion A. M., and Nauss R. Parametric and Postoptimality Analysis in Integer Linear Programming. Management Science, 1977

Check Yourself

- What constitutes a mixed-integer or integer programming problem?
- Do you expect mixed-integer programs to be more difficult to solve compared to continuous problems? Why?
- What are restrictions and relaxations? Give examples for integer variables

Applied Numerical Optimization

Prof. Alexander Mitsos, Ph.D.

Integer optimization: branch-and bound algorithm

Mixed-Integer Optimization

Semi-general MINLP:

$$\begin{aligned}\min_{\boldsymbol{x},\boldsymbol{y}} \quad & f(\boldsymbol{x}) + \boldsymbol{d}^T\boldsymbol{y} \\ \text{s.t.} \quad & c_i(\boldsymbol{x}) = 0, \forall\, i \in E \\ & c_i(\boldsymbol{x}) + \boldsymbol{a}_{y,i}^T\boldsymbol{y} \le 0, \forall\, i \in I\end{aligned}$$

Most general formulation:

$$\begin{aligned}\min_{\boldsymbol{x},\boldsymbol{y}} \quad & f(\boldsymbol{x},\boldsymbol{y}) \\ \text{s.t.} \quad & c_i(\boldsymbol{x},\boldsymbol{y}) = 0, \forall\, i \in E \\ & c_i(\boldsymbol{x},\boldsymbol{y}) \le 0, \forall\, i \in I\end{aligned}$$

$\boldsymbol{x} \in R^{n_x}$, continuous variables

$\boldsymbol{y} \in Y$, discrete variables (e.g. $\boldsymbol{y} \in \{0,1\}^{n_y}$)

0-1 MILP:

$$\begin{aligned}\min_{\boldsymbol{x},\boldsymbol{y}} \ & \boldsymbol{d}_x^T\boldsymbol{x} + \boldsymbol{d}_y^T\boldsymbol{y} \\ \text{s.t.} \ & \boldsymbol{a}_{x,i}^T\boldsymbol{x} + \boldsymbol{a}_{y,i}^T\boldsymbol{y} = b_i, \forall\, i \in \mathrm{E} \\ & \boldsymbol{a}_{x,i}^T\boldsymbol{x} + \boldsymbol{a}_{y,i}^T\boldsymbol{y} = b_i, \forall\, i \in I \\ & \boldsymbol{x} \in R^{n_x} \\ & \boldsymbol{y} \in \{0,1\}^{n_y}\end{aligned}$$

Branch-and-Bound (B&B) Method – Basics

"Branch-and-Bound" (BB)

- is applicable for all functions: linear, convex nonlinear, nonconvex nonlinear

- guarantees an optimal solution in finite # iterations
  - exact for MILP
  - to arbitrary tolerance for MINLP

- is basis for many solvers ("branch-and-do-the-right thing")
  - CPLEX, Gurobi, XPRESS for MILP (and MIQP)
  - ANTIGONE, BARON, Couenne, EAGO, LINDo, MAiNGO, Octeract, SCIP for (MI)NLP
  - Not the only algorithm, but the standard one

- is inherently exponential algorithm → must perform much better than worst-case to be tractable

- relies on multiple heuristics: B&B terminates for any choice, but # iterations varies widely

B&B for MILP – Solution Strategy

Construct a series of restricted MILPs and solve their LP-relaxation

- "Branching"-step: choose $y_j \in \{0,1\}$ and create two MILP sub-problems with $y_j = 0$ and $y_j = 1$
  - Children inherit lower bound from parent
  - Heuristic choice: e.g., pick variable with non-integer optimal value in the LP-relaxation.

- Node selection: select which of the open alternatives to visit next
  - Heuristic: best lower bound, breadth-first, depth-first, ….

- Bounding: relax free binary variables $y_j \in \{0,1\}$ to $y_j \in [0,1]$, solve LP $\rightarrow \boldsymbol{x}^{(k)}, \boldsymbol{y}^{(k)}, f^{(k)}$.
  - If $y_j^{(k)} \in \{0,1\}, \forall j$, then $\boldsymbol{x}^{(k)}, \boldsymbol{y}^{(k)}$ is feasible in the original MILP and $f^{(k)}$ is upper bound. Node can be fathomed.
  - If $\exists j$: $y_j^{(k)} \in (0,1)$, then $\boldsymbol{x}^{(k)}, \boldsymbol{y}^{(k)}$ is infeasible in the original MILP. $f^{(k)}$ is new lower bound for this node.
  - Global lower bound is minimum among the lower bound of all active nodes.
  - Descending the tree, both local and global lower bounds increase.

- Fathoming: eliminate all nodes with lower bound $\geq$ current best upper bound

- End of iteration, if lower bound ≈ upper bound

B&B for MILP – A Simple Example

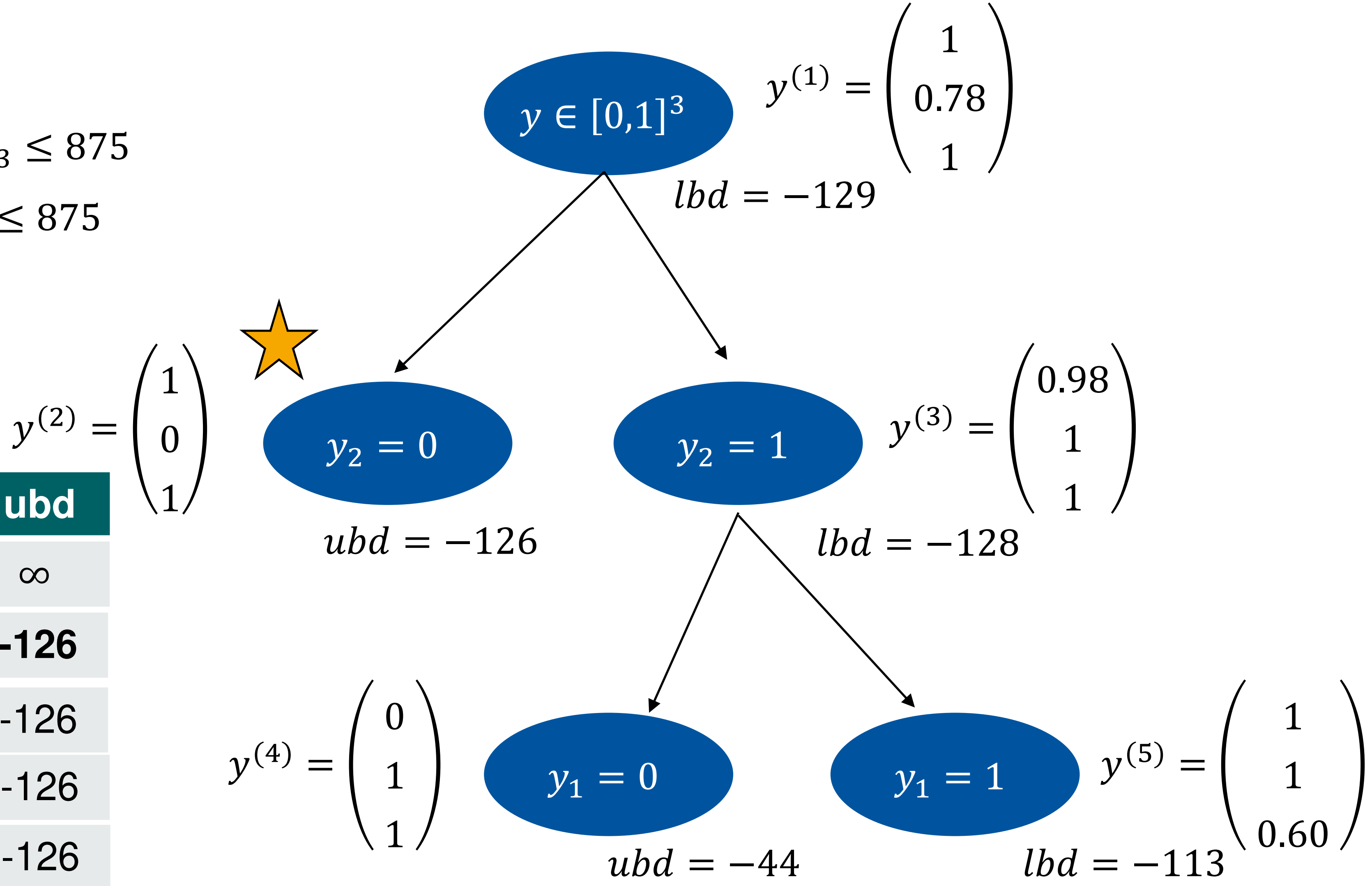
ILP:

$$\min_{y_1,y_2,y_3} \quad -86y_1 - 4y_2 - 40y_3$$

$$\text{s.t.} \quad 774\, y_1 + 76y_2 + 42y_3 \le 875$$

$$67y_1 + 27y_2 + 53y_3 \le 875$$

$$y_{1,2,3} \in \{0,1\}$$



Node	lbd^k	ubd^k	lbd	ubd
1	-129	∞	-129	∞
2	-126	-126	-129	**-126**
3	-128	∞	**-128**	-126
4	-44	-44	-128	-126
5	-113	∞	**-126**	-126

Check Yourself

- What is the main idea of the Branch & Bound algorithm?
- Describe the Branch-and-Bound method. Is this method always efficient from a computational point of view?