Mixed-Integer Nonlinear Optimization: Applications, Algorithms, and Computation

Sven Leyffer

Mathematics & Computer Science Division
Argonne National Laboratory

Graduate School in
Systems, Optimization, Control and Networks
Université catholique de Louvain
February 2013

Collaborators





Pietro Belotti, Ashutosh Mahajan, Christian Kirches, Jeff Linderoth, and Jim Luedtke

Outline

Mixed-Integer Nonlinear Optimization

Mixed-Integer Nonlinear Program (MINLP)

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & f(x) \\ \text{subject to} \quad & c(x) \leq 0 \\ & x \in X \\ & x_i \in \mathbb{Z} \text{ for all } i \in I \end{aligned}$$

- $f : \mathbb{R}^n \to \mathbb{R}$, $c : \mathbb{R}^n \to \mathbb{R}^m$ smooth (often convex) functions
- $X \in \mathbb{R}^n$ bounded, polyhedral set, e.g. $X = \{x : l \leq A^T x \leq u\}$
- $I \subset \{1, \ldots, n\}$ subset of integer variables
- $x_i \in \mathbb{Z}$ for all $i \in I$ ... combinatorial problem
- Combines challenges of handling nonlinearities with combinatorial explosion of integer variables
- More general constraints possible, e.g. $l \leq c(x) \leq u$ etc.

Complexity of MINLP

Mixed-Integer Nonlinear Program (MINLP)

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & f(x) \\ \text{subject to} \quad & c(x) \leq 0 \\ & x \in X \\ & x_i \in \mathbb{Z} \text{ for all } i \in I \end{aligned}$$

Complexity of MINLP

- MINLP is NP-hard: includes MILP, which are NP-hard [Kannan and Monma, 1978]
- Worse: MINLP are undecidable [Jeroslow, 1973]: quadratically constrained IP for which no computing device can compute the optimum for all problems in this class
  ... but we're OK if X is compact!

Notation

Some notation used throughout the course ...

- $f^{(k)} = f(x^{(k)})$ evaluated at $x = x^{(k)}$
- $\nabla f^{(k)} = \nabla f(x^{(k)})$ gradient
- Hessian of Lagrangian $\mathcal{L}(x, \lambda) = f(x) - \sum \lambda_i c_i(c)$ is $\nabla^2 \mathcal{L}^{(k)}$
  ... assumes X polyhedral
- Subscripts denote components, e.g. x_i is component i of x
- If $J \subset \{1, \ldots, n\}$ then x_J are components of x corres. to J
- x_I integer and x_C are the continuous variables, $p = |I|$
- Floor and ceiling operators: $\lfloor x_i \rfloor$ and $\lceil x_i \rceil$:
  - $\lfloor x_i \rfloor$ largest integer smaller than or equal to x_i
  - $\lceil x_i \rceil$ smallest integer larger than or equal to x_i

Convexity of Nonlinear Functions

MINLP techniques distinguish convex and nonconvex MINLPs.
For our purposes, we define convexity as ...

Definition

A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex, iff $\forall x^{(0)}, x^{(1)} \in \mathbb{R}^n$ we have:

$$f(x^{(1)}) \geq f(x^{(0)}) + (x^{(1)} - x^{(0)})^T \nabla f^{(0)}$$

In a slight abuse of notation, we say that ...

Definition

MINLP is a *convex* if the problem functions $f(x)$ and $c(x)$ are convex functions. If either $f(x)$ or any $c_i(x)$ is a nonconvex function, then MINLP is *nonconvex*.

Convexity (cont.)

We also define the convex hull of a set S as ...

Definition

For a set S, the *convex hull of* S is $\text{conv}(S)$:

$$\left\{x | x = \lambda x^{(1)} + (1-\lambda)x^{(0)},\ \forall 0 \leq \lambda \leq 1,\ \forall x^{(0)}, x^{(1)} \in S\right\}.$$

- If $X = \{x \in \mathbb{Z}^p : l \leq x \leq u\}$ and $l \in \mathbb{Z}^p$, $u \in \mathbb{Z}^p$, then $\text{conv}(X) = [l, u]^p$
- Finding convex hull is hard, even for polyhedral X.
- Convex hull important for MILP ...

Theorem

MILP can be solved as LP over the convex hull of feasible set.

MILP $\neq$ MINLP

Important difference between MINLP and MILP

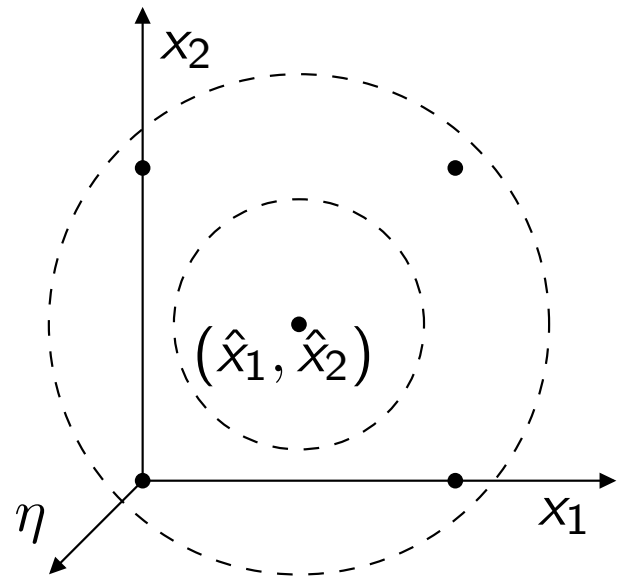
$$\underset{x}{\text{minimize}} \sum_{i=1}^{n} (x_i - \tfrac{1}{2})^2, \quad \text{subject to } x_i \in \{0, 1\}$$

... solution is not extreme point (lies in interior)

Remedy: Introduce objective η and a constraint $\eta \geq f(x)$

$$\begin{cases} \underset{\eta, x}{\text{minimize}} & \eta, \\ \text{subject to} & f(x) \leq \eta, \\ & c(x) \leq 0, \\ & x \in X, \\ & x_i \in \mathbb{Z}, \ \forall i \in I. \end{cases}$$

Assume wlog that MINLP objective is linear



Interlude: SOCN, Social Media, and MINLP

Questions, comments, and suggestions are always welcome!

Who is the audience of this course?

0	Vrije Universiteit Brussel	**4**	FUNDP NAMUR	**0**	UNIVERSITEIT GENT	**7**	KATHOLIEKE UNIVERSITEIT LEUVEN
8	Université de Liège	**1**	POLYTECH MONS	**12**	UCL Université catholique de Louvain	**0**	ULB

You can comment on this course on twitter using hastag `#socn13`
I will post links to course notes using `#socn13` after each lecture

Outline

Relaxation and Constraint Enforcement

Relaxation

- Used to compute a lower bound on the optimum
- Obtained by enlarging feasible set; e.g. ignore constraints
- Typically much easier to solve than MINLP

Constraint Enforcement

- Exclude solutions from relaxations not feasible in MINLP
- Refine or tighten of relaxation; e.g. add valid inequalities

Upper Bounds

- Obtained from any feasible point; e.g. solve NLP for fixed x_I

Relaxations of Integrality

Definition (Relaxation)

Optimization problem $\min\{\breve{f}(x) : x \in \mathcal{R}\}$ is a relaxation of $\min\{f(x) : x \in \mathcal{F}\}$, iff $\mathcal{R} \supset \mathcal{F}$ and $\breve{f}(x) \leq f(x)$ for all $x \in \mathcal{F}$.

Goal: relaxation easy to solve globally, e.g. MILP or NLP

Relaxing Integrality

- Relax Integrality $x_i \in \mathbb{Z}$ to $x_i \in \mathbb{R}$ for all $i \in I$
- Gives *nonlinear relaxation* of MINLP, or NLP:

$$\begin{cases} \underset{x}{\text{minimize}} & f(x), \\ \text{subject to} & c(x) \leq 0, \\ & x \in X, \quad \text{continuous} \end{cases}$$

- Used in branch-and-bound algorithms

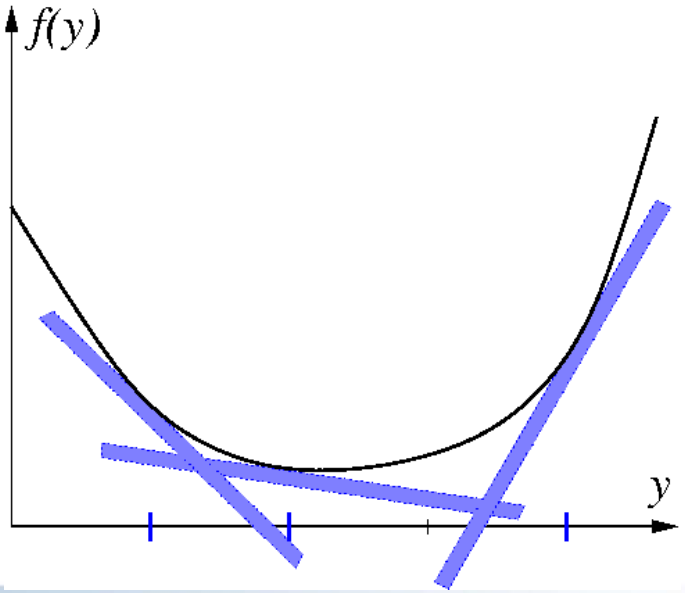
Relaxations of Nonlinear Convex Constraints

Relaxing Convex Constraints

- Convex $0 \geq c(x)$ and $\eta \geq f(x)f$ relaxed by supporting hyperplanes

$$\eta \geq f^{(k)} + \nabla f^{(k)^T}(x - x^{(k)})$$
$$0 \geq c^{(k)} + \nabla c^{(k)^T}(x - x^{(k)})$$

  for a set of points $x^{(k)},\ k = 1, \ldots, K$.
- Obtain polyhedral relaxation of convex constraints.
- Used in the outer approximation methods.



Relaxations of Nonconvex Constraints

Relaxing Nonconvex Constraints

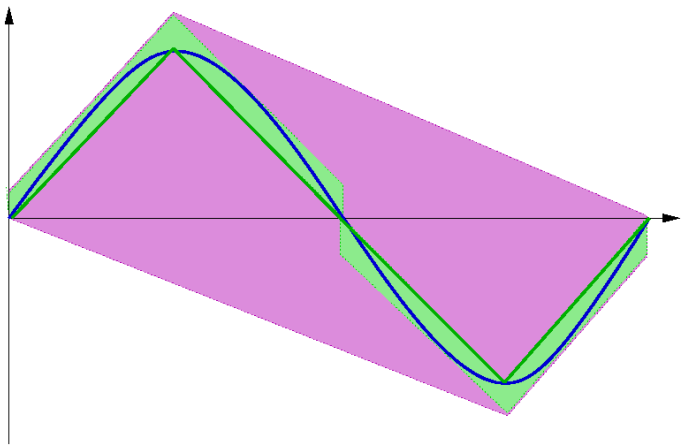
- Construct convex underestimators, $\breve{f}(x)$ and $\breve{c}(x)$ for nonconvex functions $c(x)$ and $f(x)$:

$$\breve{f}(x) \leq f(x) \quad \text{and} \quad \breve{c}(x) \leq c(x), \ \forall x \in \text{conv}(X).$$

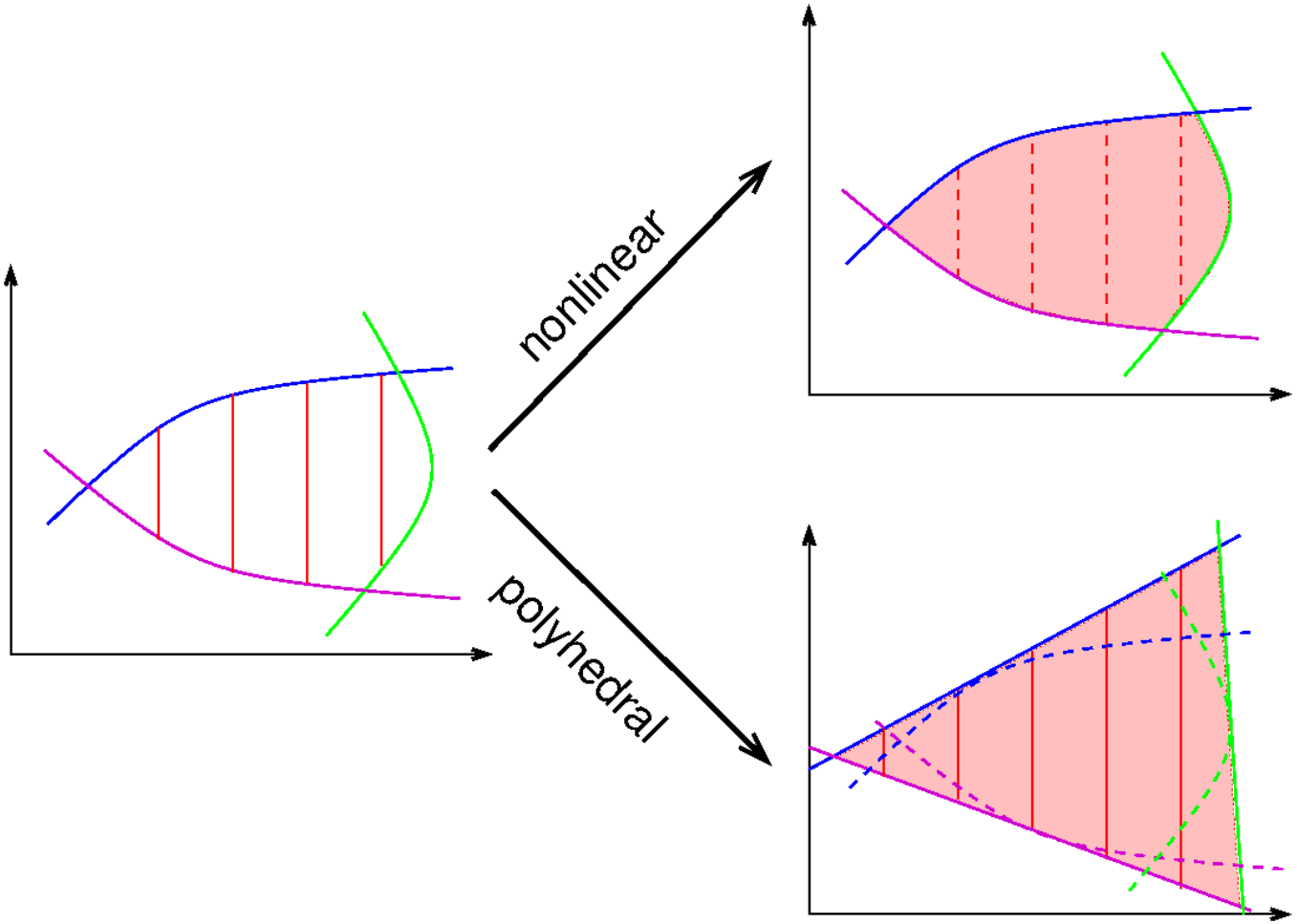
- Relax constraints $z \geq f(x)$ and $0 \geq c(x)$ as

$$z \geq \breve{f}(x) \quad \text{and} \quad 0 \geq \breve{c}(x).$$

- Used in spatial branch-and-bound.



Relaxations Summary



Nonlinear and polyhedral relaxation

Relaxations

Relaxations can be combined to produce better algorithms

- Relax convex underestimators via supporting hyperplanes.
- Relax integrality of polyhedral relaxation to obtain an LP.

Relaxations are useful because we have following result:

Theorem

If the solution of the relaxation of the η-MINLP is feasible in the η-MINLP, then it solves the MINLP.

... but if solution of relaxation is not feasible, then need ...

Constraint Enforcement

Goal: Given solution of relaxation, $\hat{x}$, not feasible in MINLP, exclude it from further consideration to ensure convergence

Three constraint enforcement strategies

1. Relaxation refinement: tighten the relaxation
2. Branching: disjunction to exclude set of non-integer points
3. Spatial branching: divide region into sub-regions

Strategies can be combined ...

Constraint Enforcement: Refinement

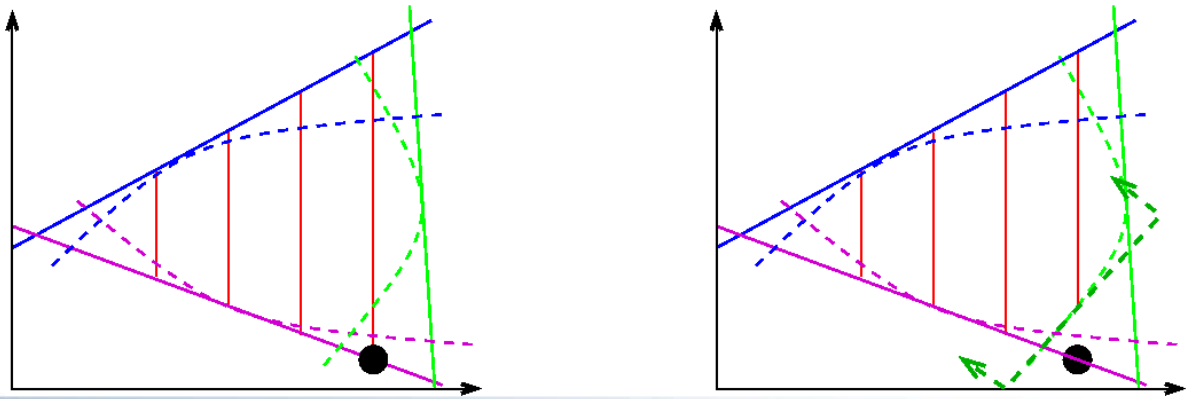
Tighten the relaxation to remove current solution $\hat{x}$ of relaxation

- Add a valid inequality to relaxation, i.e. an inequality that is satisfied by all feasible solutions of MINLP
- Valid inequality is called a cut if it excludes $\hat{x}$
- Example: $c(x) \leq 0$ convex, and $\exists i : c_i(\hat{x}) > 0$, then

$$0 \geq \hat{c}_i + \nabla \hat{c}^T (x - \hat{x})$$

  cuts off $\hat{x}$. Proof: Exercise.

- Used in Benders decomposition and outer approximation.
- MILP: cuts are basis for branch-and-cut techniques.



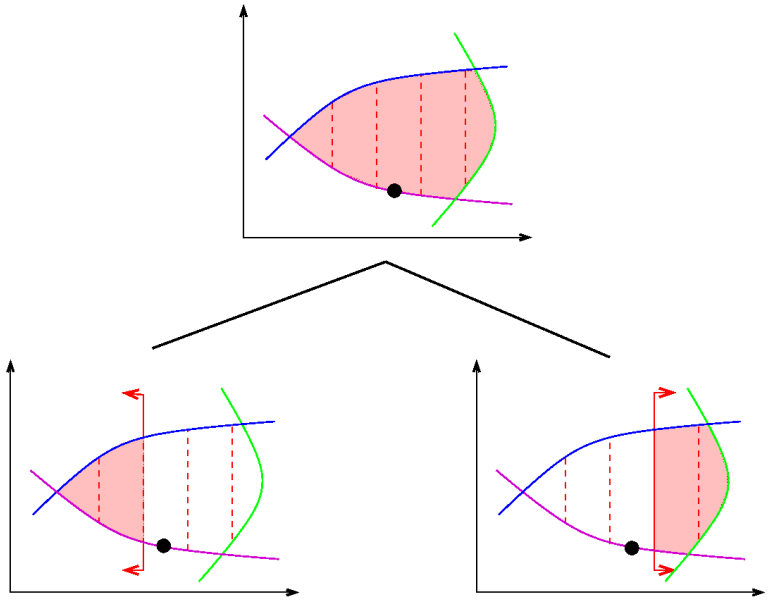
Constraint Enforcement: Branching

Eliminate current $\hat{x}$ solution by branch on integer variables:

1. Select fractional $\hat{x}_i$ for some $i \in I$
2. Create two new relaxations by adding

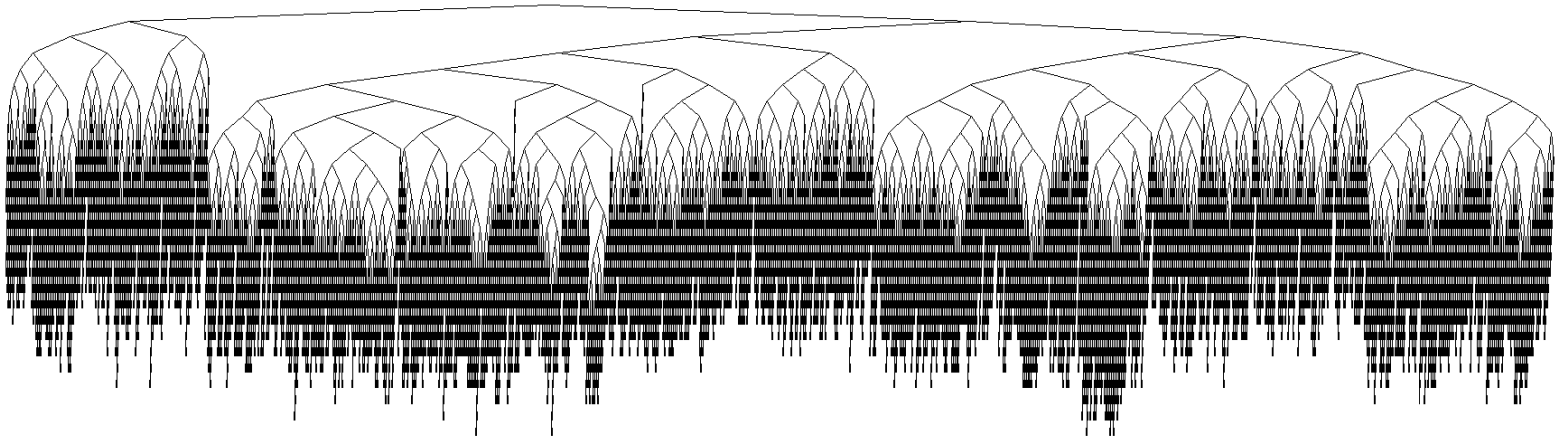
$$x_i \leq \lfloor \hat{x}_i \rfloor \quad \text{and} \quad x_i \geq \lceil \hat{x}_i \rceil \quad \text{respectively}$$

... solution to MINLP lies in one of the new relaxations.



... creates branch-and-bound tree

Branch-and-Bound Trees can be Huge



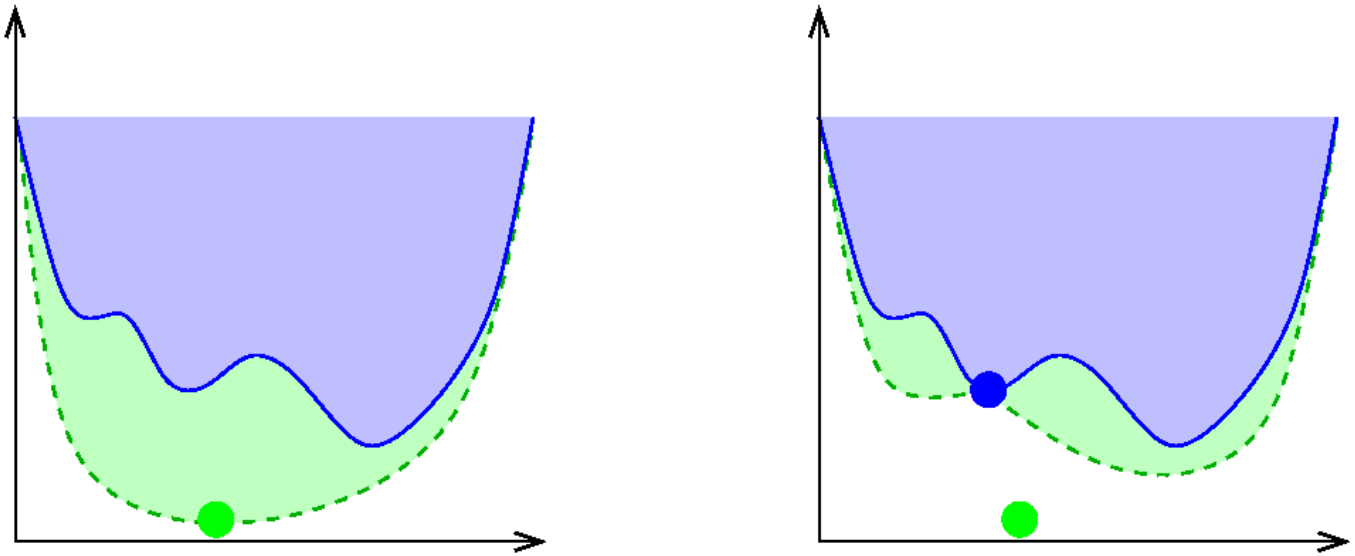
Tree after 360 s CPU time has more than 10,000 nodes

Constraint Enforcement: Spatial Branching

Enforcement for relaxed nonconvex constraints

- Combine branching and relaxation refinement
- Branch on continuous variable and split domain in two parts.
- Create new relaxation over (reduced) sub-domains.
- Generates tree similar to integer branching.
- Mix with interval techniques to eliminate sub-domains.

Nonconvex MINLPs combine all 3 enforcement techniques.



Outline

Nonlinear Optimization Background

Consider nonlinear optimization problem (NLP)

$$(P) \quad \underset{x}{\text{minimize}} \; f(x) \quad \text{subject to } c(x) = 0, \quad x \geq 0$$

- All variables continuous in this section
- $f : \mathbb{R}^n \to \mathbb{R}$ and $c : \mathbb{R}^n \to \mathbb{R}^m$ twice continuously differentiable
- f, c not necessarily convex in this section

Optimality Conditions for NLP

$$(P) \quad \underset{x}{\text{minimize}} \ f(x) \quad \text{subject to } c(x) = 0, \quad x \geq 0$$

Definition (Active Set)

$\mathcal{A}(x) := \{i | x_i = 0\}$ denotes set of active inequality constraints.

Definition (Constraint Qualification)

(P) satisfies the linear independence constraint qualification (LICQ) at x^*, iff $[\nabla c^* : I_{\mathcal{A}(x^*)}]$ has full rank, where $I_{\mathcal{A}(x^*)}$ are unit columns corresponding to active set $\mathcal{A}(x^*)$.

- Other CQs are possible (weakest is MFCQ)
- CQs imply that feasible set "looks locally linear"
- Exclude cusps from consideration
- Ensure we can state optimality conditions & methods

Optimality Conditions for NLP

$$(P) \quad \underset{x}{\text{minimize}} \ f(x) \quad \text{subject to } c(x) = 0, \quad x \geq 0$$

Theorem (Karush-Kuhn-Tucker Conditions)

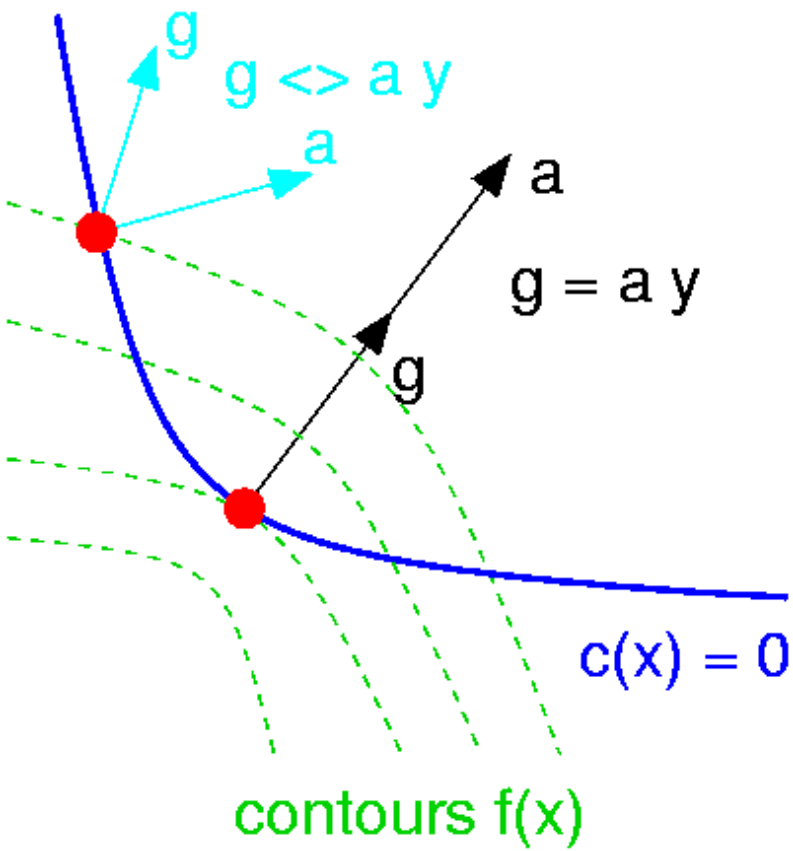
x^* local minimizer & CQ holds $\Rightarrow \exists$ multipliers y^*, z^*:

$$\nabla f^* - \nabla c^{*T} y^* - z^* = 0$$
$$c(x^*) = 0$$
$$x^* \geq 0, \ z^* \geq 0, \quad \text{and} \quad X^* z^* = 0$$

where $X^* = \text{diag}(x^*)$, thus $X^* z^* = 0 \ \Leftrightarrow \ x_i^* z_i^* = 0$

Lagrangian: $\mathcal{L}(x, y, z) := f(x) - y^T c(x) - z^T x$

Optimality Conditions for NLP



Objective gradient is linear combination of constraint gradients

$$g(x) = A(x)y, \qquad \text{where } g(x) := \nabla f(x),\ A(x) := \nabla c(x)^T$$

... do not look at 2nd order conditions.

Alternative Optimality Conditions for NLP

$$(P) \quad \underset{x}{\text{minimize}} \; f(x) \quad \text{subject to } c(x) = 0, \quad x \geq 0$$

Theorem (LP-Based Optimality)

x^* local minimizer & CQ holds $\Rightarrow$ $d = 0$ solves linear approximation:

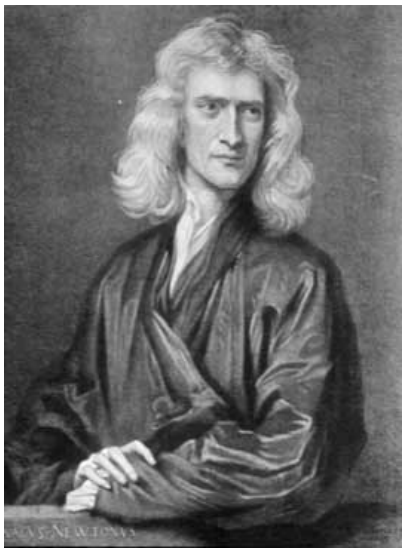
$$\begin{aligned} \underset{d}{\text{minimize}} \;\; & \nabla f^{*^T} d \\ \text{subject to } & c^* + \nabla c^{*^T} d = 0 \\ & x^* + d \geq 0 \end{aligned}$$

There exists no linearized descend direction ... suggests methods!

Used in convergence proof of outer approximation ...

Newton's Method for Nonlinear Equations

KKT conditions are nonlinear set of equations $F(x) = 0$ ...



To solve $F(x) = 0$:
Get approx. $x^{(k+1)}$ of solution of $F(x) = 0$
by solving linear model about $x^{(k)}$:

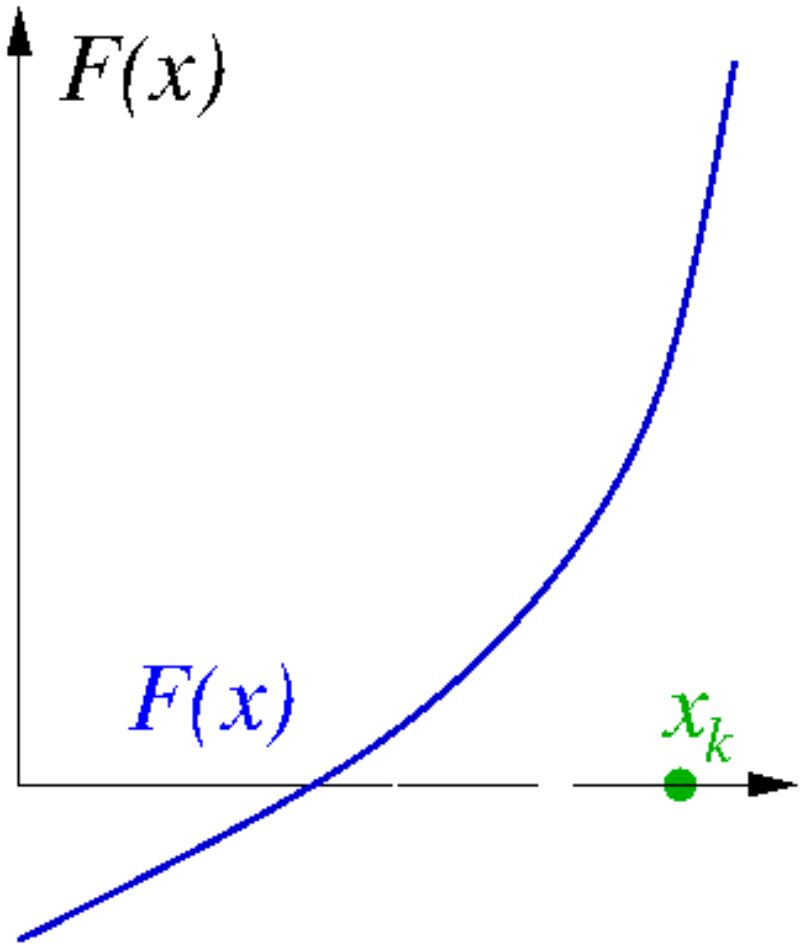
$$F(x^{(k)}) + \nabla F(x^{(k)})^T (x - x^{(k)}) = 0$$

for $k = 0, 1, \ldots$

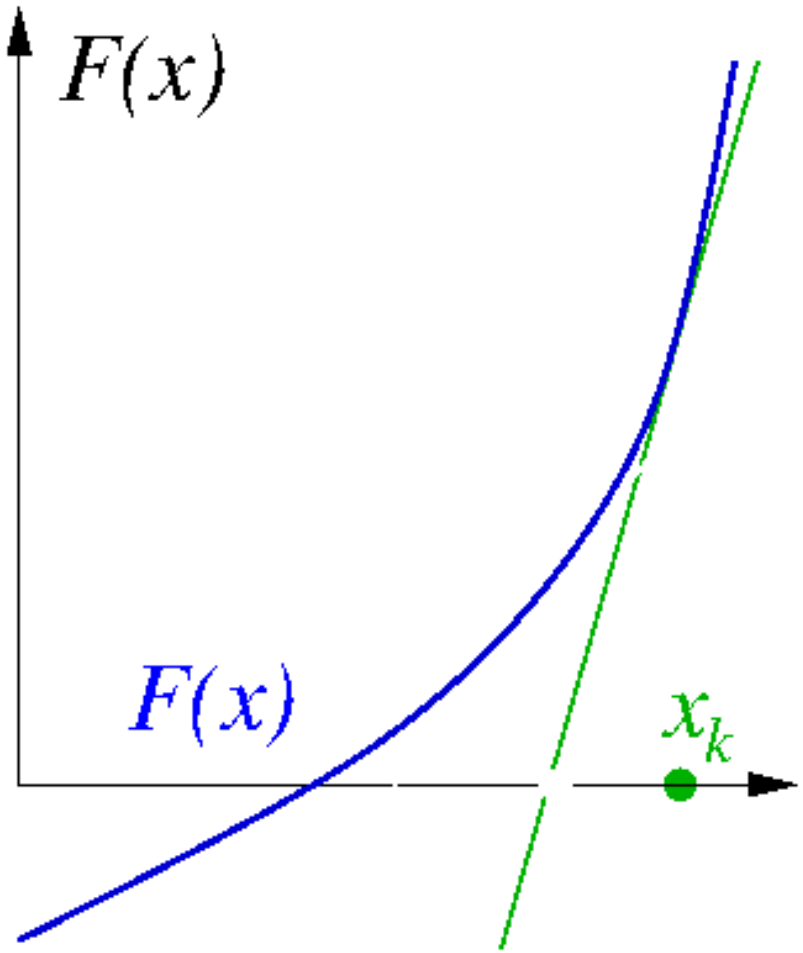
Theorem (Local Convergence of Newton's Method)

If $F \in \mathcal{C}^2$, and $\nabla F(x^*)$ nonsingular, then Newton converges quadratically near x^*.

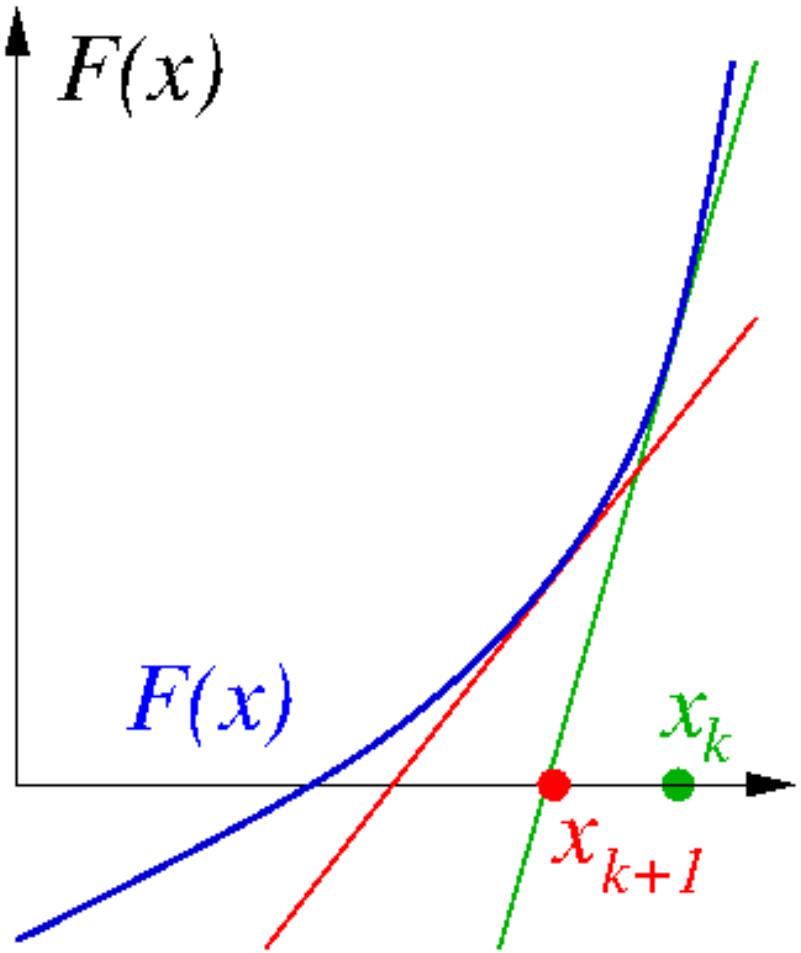
Newton's Method for Nonlinear Equations



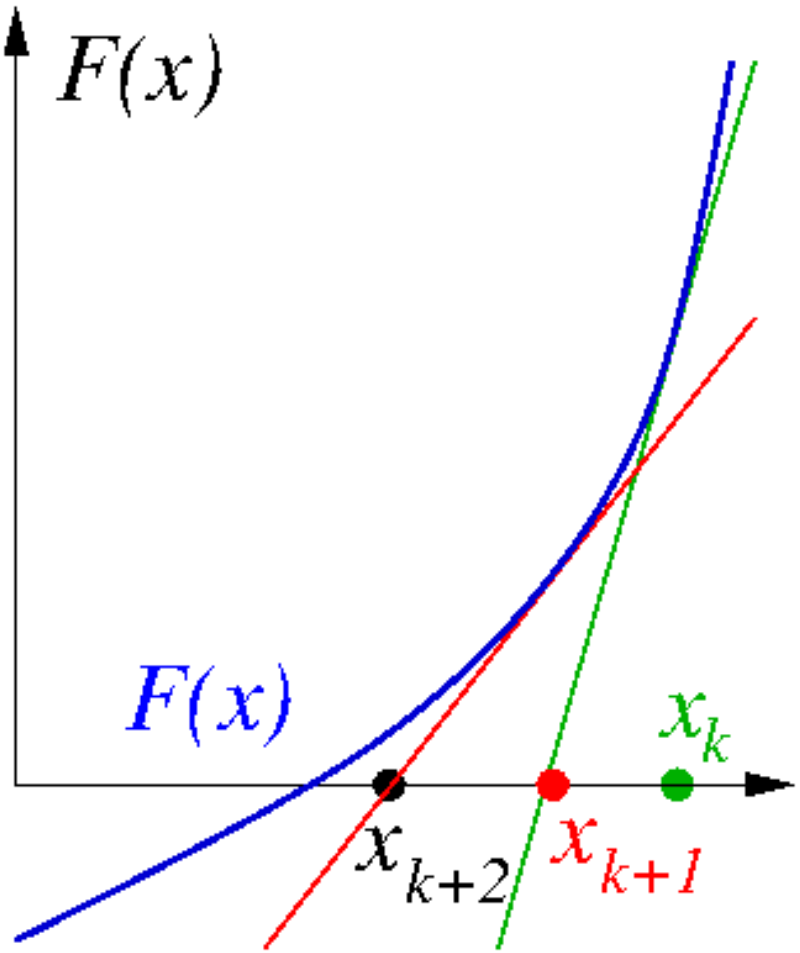
Newton's Method for Nonlinear Equations



Newton's Method for Nonlinear Equations



Newton's Method for Nonlinear Equations



Sequential Quadratic Programming (SQP)

SQP for inequality constrained NLP:

$$\underset{x}{\text{minimize}} \; f(x) \quad \text{subject to } c(x) = 0 \quad \& \quad x \geq 0$$

REPEAT

1. Solve QP for $(s, y^{(k+1)}, z^{(k+1)})$

$$\begin{cases} \underset{s}{\text{minimize}} & \nabla f^{(k)^T} s + \frac{1}{2} s^T H^{(k)} s \\ \text{subject to} & c^{(k)} + A^{(k)^T} s = 0 \\ & x^{(k)} + s \geq 0 \end{cases}$$

   where $H^{(k)} \simeq \nabla^2 \mathcal{L}^{(k)}$ Hessian of Lagrangian

2. Set $x^{(k+1)} = x^{(k)} + s$

... QP solve computationally expensive

Modern Interior-Point Methods (IPM)

General NLP

$$\underset{x}{\text{minimize}} \; f(x) \quad \text{subject to } c(x) = 0 \quad \& \quad x \geq 0$$

Perturbed $\mu > 0$ optimality conditions ($x, z > 0$)

$$F_\mu(x, y, z) = \begin{Bmatrix} \nabla f(x) - \nabla c(x)^T y - z \\ c(x) \\ Xz - \mu e \end{Bmatrix} = 0$$

- Primal-dual formulation, where $X = \text{diag}(x)$
- Central path $\{x(\mu), y(\mu), z(\mu) \; : \; \mu > 0\}$
- Apply Newton's method for sequence $\mu \searrow 0$

Modern Interior-Point Methods (IPM)

Newton's method applied to primal-dual system ...

$$\begin{bmatrix} \nabla^2 \mathcal{L}^{(k)} & -A^{(k)} & -I \\ A^{(k)^T} & 0 & 0 \\ Z^{(k)} & 0 & X^{(k)} \end{bmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta z \end{pmatrix} = -F_\mu(x^{(k)}, y^{(k)}, z^{(k)})$$

where $A^{(k)} = \nabla c(x^{(k)})^T$, $X^{(k)}$ diagonal matrix of $x^{(k)}$.

Polynomial run-time guarantee for convex problems

Solving Nonlinear Optimization Problems

$$(P) \quad \underset{x}{\text{minimize}} \; f(x) \quad \text{subject to } c(x) \geq 0$$

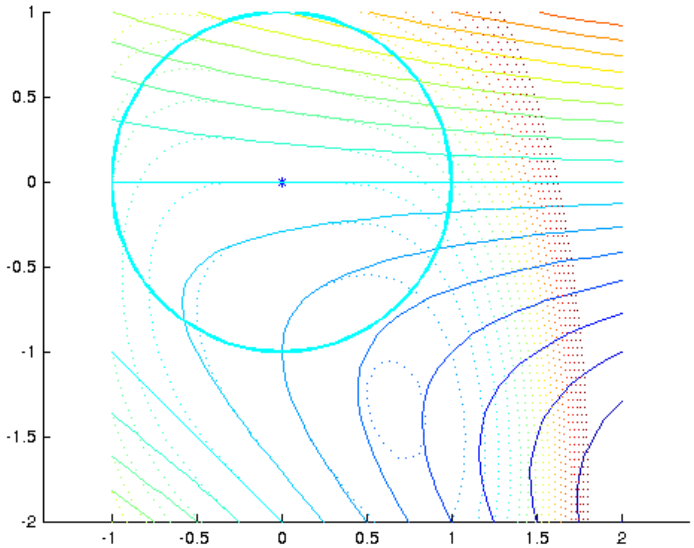
Main ingredients of iterative solution approaches:

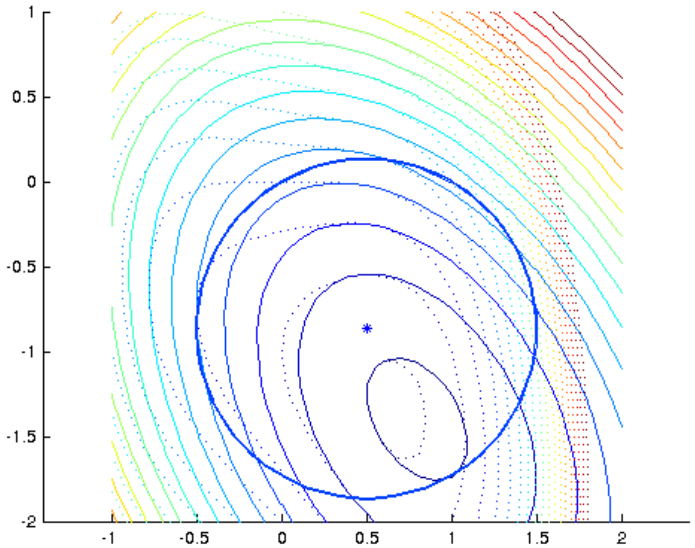
1. Local Method: Given $x^{(k)}$ (solution guess) find a step s.
   - Sequential Quadratic Programming (SQP)
   - Sequential Linear/Quadratic Programming (SLQP)
   - Interior-Point Methods
2. Forcing Strategy: Convergence from remote starting points.
3. Forcing Mechanism: Truncate step s to force progress:
   - Trust-region to restrict s of local problem ... used in this talk.
   - Back-tracking line-search along step s.

Trust-Region Methods

Globalize SQP/IPM using trust region, $\Delta^k > 0$:
Consider unconstrained $f(x)$ minimization by trust-region

$$\min_s \; q^{(k)}(s) := f(x^{(k)}) + \nabla f(x^{(k)})^T s + \frac{1}{2} s^T H(x^{(k)}) s \text{ s.t. } \|s\| \leq \Delta^k$$





Active-Set (SQP) vs Interior-Point for MINLP

What NLP solver should I use in MINLP?

- Warm-start for IPM is active area of research
- Active-set methods have better re-start capabilities
- MINLP $\neq$ MILP:
  - LP basis factors can be re-used in tree-search
  - NLP (KKT) factors are always out-of-date

  ... no quick re-solves ... not even SQP or SLP!!!

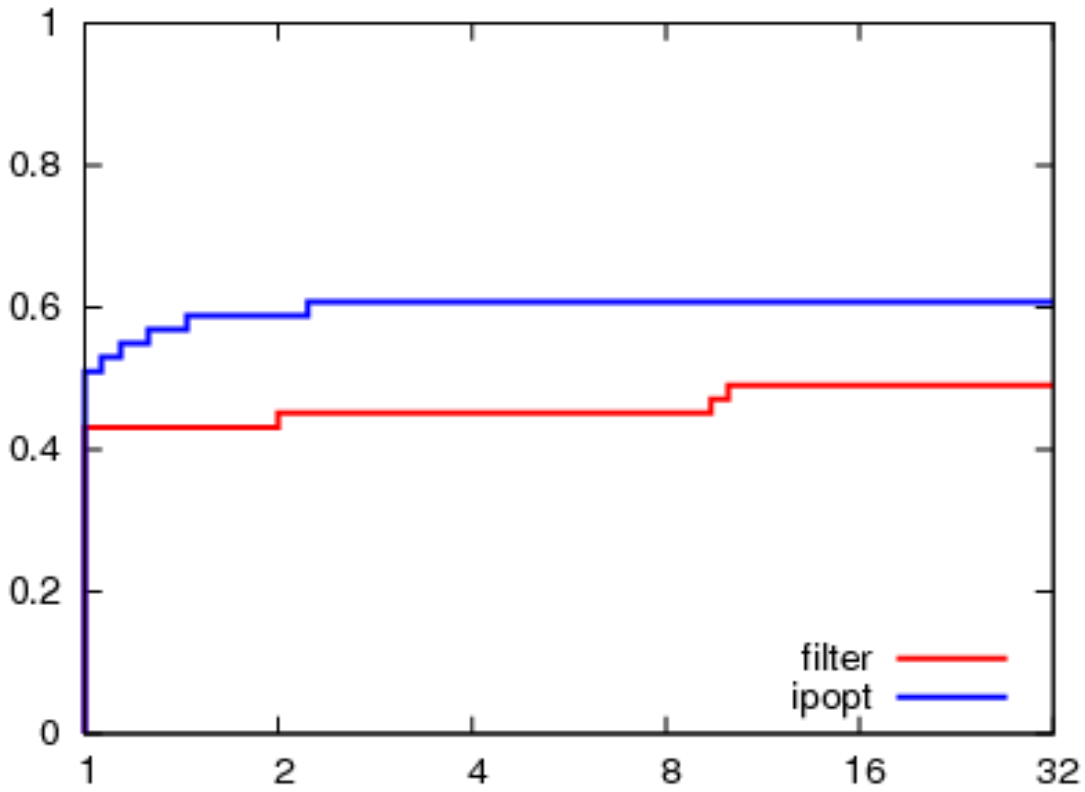
... active-set methods are preferred for MINLPs.

The Three Most Important Things in MINLP

Use LP solvers as much as possible in MINLP

$\Rightarrow$ linearize, linearize, linearize

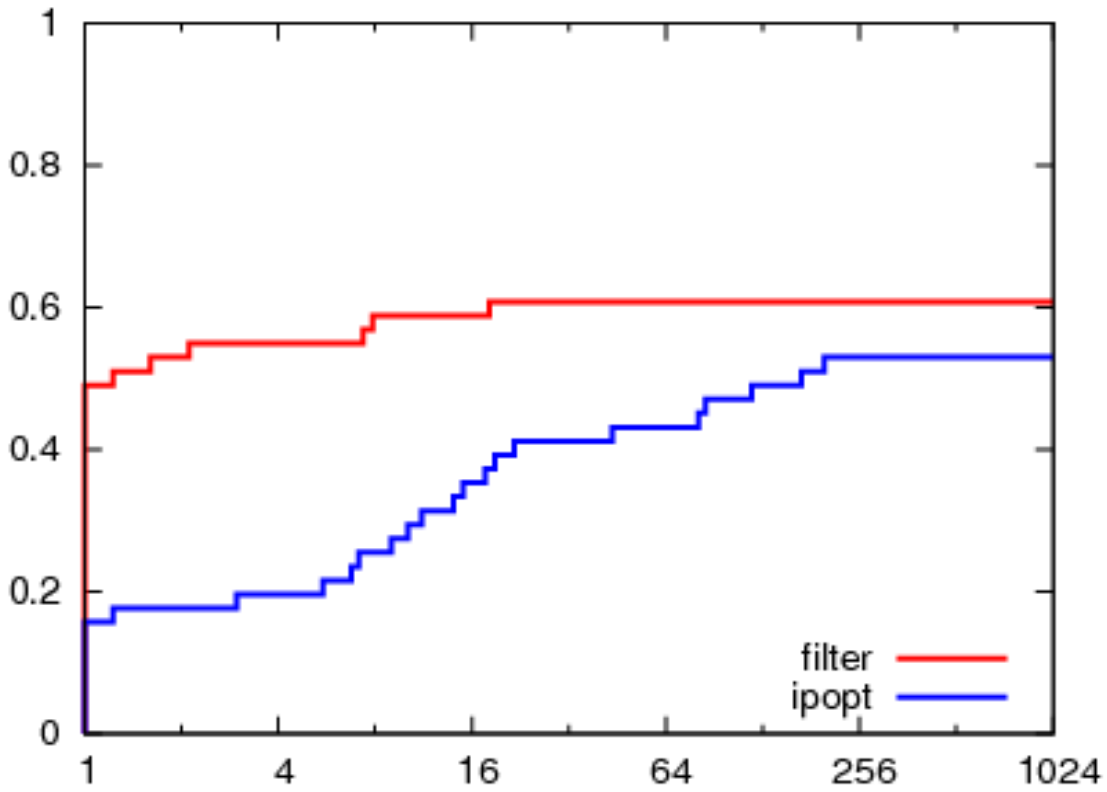
Active-Set vs. Interior-Point Solvers



BONMIN with IPOPT vs. FilterSQP: # nodes

- IPOPT has fewer nodes ... random round-off issue?

Active-Set vs. Interior-Point Solvers



BONMIN with IPOPT vs. FilterSQP: CPU time

- IPOPT has fewer nodes ... random round-off issue?
- FilterSQP warm-starts faster ... not surprising!

Outline

MINLP Modeling Practices

Modeling plays a fundamental role in MILP see [Williams, 1999]
... even more important in MINLP

- MINLP combines integer and nonlinear formulations
- Reformulations of nonlinear relationships can be convex
- Interactions of nonlinear functions and binary variables
- Sometimes we can linearize expressions

MINLP Modeling Preference

We prefer linear over convex over nonconvex formulations.

Convexification of Binary Quadratic Programs

Consider pure binary quadratic function

$$q(x) = x^T Qx + g^T x \quad \text{where } x \in \{0,1\}^p$$

Let λ be smallest eigenvalue of Q

If $\lambda \geq 0$ then $q(x)$ is convex

Convexification of Binary Quadratics

Let $W := Q - \lambda I$ and $c := g + \lambda e$, where $e = (1, \ldots, 1)$, then $q(x) = x^T Wx + c^T x$ is convex.

Modeling of Discrete Variables

We can model discrete variables such as

$$y \in \{Y_1, Y_2, \ldots, Y_k\}$$

where Y_i are discrete parameters (e.g. pipe diameters) with special ordered sets (SOS):

$$y = \sum_{i=1}^{k} z_i Y_i, \quad 1 = \sum_{i=1}^{k} z_i, \quad z_i \in \{0, 1\}$$

see [Beale and Tomlin, 1970, Beale and Forrest, 1976]

- Similarly linearize univariate functions $f(z)$, $z \in \mathbb{Z}$
- Generalizes to higher dimensions
- Solvers detect SOS structure and use special branching rules

Exploiting Low-Rank Hessians

Consider (convex) quadratic function

$$q(x) = x^T W x + g^T x,$$

where x mixture of variables, and W dense with structure:

- $W = Z^T R^{-1} Z$ low rank, e.g. estimation problems
- $R \in \mathbb{R}^{m \times m}$ nonsingular (co-variance matrix)
- $Z \in \mathbb{R}^{m \times n}$, where $m \ll n$ and Z is sparse.

Then introduce variables z, constraints

$$z = Zx, \quad \text{and write} \quad x^T W x = z^T R^{-1} z$$

... QP/NLP solvers can exploit sparsity of Z.

Linearization of Constraints

Assum $x_2 \neq 0$. A simple transformation (a constant parameter):

$$\frac{x_1}{x_2} = a \Leftrightarrow x_1 = ax_2$$

Linearization of bilinear terms $x_1 x_2$ with:

- Binary variable $x_2 \in \{0, 1\}$
- Variable upper bound: $0 \leq x_1 \leq Ux_2$

... introduce new variable x_{12} to replace $x_1 x_2$ and add constraints

$$0 \leq x_{12} \leq x_2 U \quad \text{and} \quad -U(1 - x_2) \leq x_1 - x_{12} \leq U(1 - x_2),$$

Never Multiply a Nonlinear Function by a Binary

Previous example generalizes to nonlinear functions
Often binary variables "switch" constraints on/off

Warning

Never model on/off constraints by multiplying by a binary variable.

Three alternative approaches

- Disjunctive programming, see [Grossmann and Lee, 2003]
- Perspective formulations (not always), see [Günlük and Linderoth, 2012]
- Big-M formulation (weak relaxations)

Avoiding Undefined Nonlinear Expressions

MINLP solvers fail because NLP solver gets IEEE exception, e.g.

$$c(x_1) = -\ln(\sin(x_1)) \leq 0,$$

cannot be evaluated at $\sin(x_1) \leq 0$

Reformulate equivalently as

$$\tilde{c}(x_2) = -\ln(x_2) \leq 0,\ x_2 = \sin(x_1), \quad \text{and} \quad x_2 \geq 0.$$

IPM solvers never evaluate at $x_2 \leq 0$
Active-set method can also safeguard against $x_2 \leq 0$

- $x_2 \geq 0$ is s simple bound which can be enforced exactly
- $x_2 = 0$ get IEEE exception $\Rightarrow$ trap & reduce trust-region
- As $x_2 \to 0$, the constraint violation $c(x_2) \to \infty$

Variable Transformations

Design of multiproduct batch plant includes nonconvex terms

$$\sum_{j \in M} \alpha_j N_j V_j^{\beta_j}; \qquad C_i N_j \geq \tau_{ij}; \qquad \sum_{i \in N} \frac{\psi_i}{B_i} C_i \leq \gamma$$

where variables are upper case, parameters are Greek letters.

Introduce log-transform variables

$$v_j = \ln(V_j), \quad n_j = \ln(N_j), \quad b_i = \ln(B_i), \quad c_i = \ln(C_i).$$

Transformed expressions are convex:

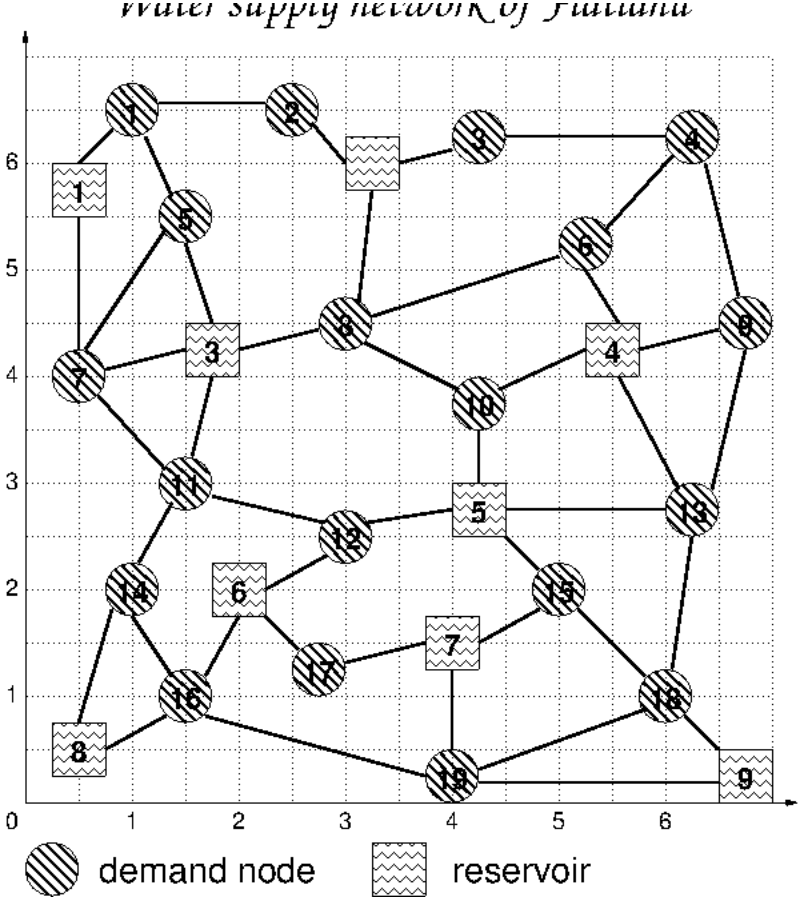
$$\sum_{j \in M} \alpha_j e^{n_j + \beta_j v_j}, \qquad c_i + n_j \geq \ln(\tau_{ij}), \qquad \sum_{i \in N} \psi_i e^{c_i - b_i} \leq \gamma$$

Design of Water Distribution Networks

Model of water, gas, air networks

Goal: design minimum cost network from discrete pipe diameters

- $\mathcal{N}$ nodes in network
- $\mathcal{S}$ source nodes
- $\mathcal{A}$: arcs in the network



Design of Water Distribution Networks

Goal: design minimum cost network from discrete pipe diameters

$\mathcal{N}$ nodes, $\mathcal{S}$ source nodes, $\mathcal{A}$: arcs in the network

Variables:

- q_{ij}: flow pipe $(i,j) \in \mathcal{A}$
- d_{ij}: diameter of pipe $(i,j) \in \mathcal{A}$, where $d_{ij} \in \{P_1, \ldots, P_r\}$
- h_i: hydraulic head at node $i \in \mathcal{N}$
- z_{ij}: binary variables model flow direction $(i,j) \in \mathcal{A}$
- a_{ij}: area of cross section $(i,j) \in \mathcal{A}$
- y_{ijk}: SOS-1 variables to model diameter

NB: $a_{ij} = \pi d_{ij}^2/4$ is redundant ... but useful!

Design of Water Distribution Networks

$\mathcal{N}$ nodes, $\mathcal{S}$ source nodes, $\mathcal{A}$: arcs in the network
Equations for q_{ij} flow pipe $(i,j) \in \mathcal{A}$

- Conservation of flow at every node

$$\sum_{(i,j)\in\mathcal{A}} q_{ij} - \sum_{(j,i)\in\mathcal{A}} q_{ji} = D_i,\ \forall i \in \mathcal{N} - \mathcal{S}.$$

- Flow bounds are linear in d_{ij} ... nonlinear in a_{ij}:

$$-V_{\max} a_{ij} \leq q_{ij} \leq V_{\max} a_{ij},\ \forall (i,j) \in \mathcal{A}.$$

Design of Water Distribution Networks

Modeling Trick: SOS & Nonlinear Expressions

Modeling discrete $d_{ij} \in \{P_1, \ldots, P_r\}$ and nonlinear $a_{ij} = \pi d_{ij}^2/4$:

1. Introduce SOS-1 variables $y_{ijk} \in \{0, 1\}$ for $k = 1, \ldots, r$
2. Model discrete choice as

$$\sum_{k=1}^{r} y_{ijk} = 1, \quad \text{and} \quad \sum_{k=1}^{r} P_k y_{ijk} = d_{ij}. \; \forall (i,j) \in \mathcal{A},$$

3. Model nonlinear relationship as

$$\sum_{k=1}^{r} (\pi P_k/4) y_{ijk} = a_{ij}, \; \forall (i,j) \in \mathcal{A}.$$

$\Rightarrow$ no longer need $a_{ij} = \pi d_{ij}^2/4$!

Design of Water Distribution Networks

Nonsmooth pressure loss model along arc $(i,j) \in \mathcal{A}$

$$h_i - h_j = \frac{\text{sgn}(q_{ij})|q_{ij}|^{c_1} c_2 L_{ij} K_{ij}^{-c_1}}{d_{ij}^{c_3}}$$

... introduce binary variables to model nonsmooth term $|q_{ij}|^{c_1}$

1. Add binary variables $z_{ij} \in \{0,1\}$
2. $$0 \le q_{ij}^{+} \le Q_{\max} z_{ij},\ 0 \le q_{ij}^{-} \le Q_{\max}(1 - z_{ij}),\ q_{ij} = q_{ij}^{+} - q_{ij}^{-}.$$
3. Pressure drop becomes

$$h_i - h_j = \frac{\left[\left(q_{ij}^{+}\right)^{c_1} - \left(q_{ij}^{-}\right)^{c_1}\right] c_2 L_{ij} K_{ij}^{-c_1}}{d_{ij}^{c_3}},\ \forall (i,j) \in \mathcal{A}.$$

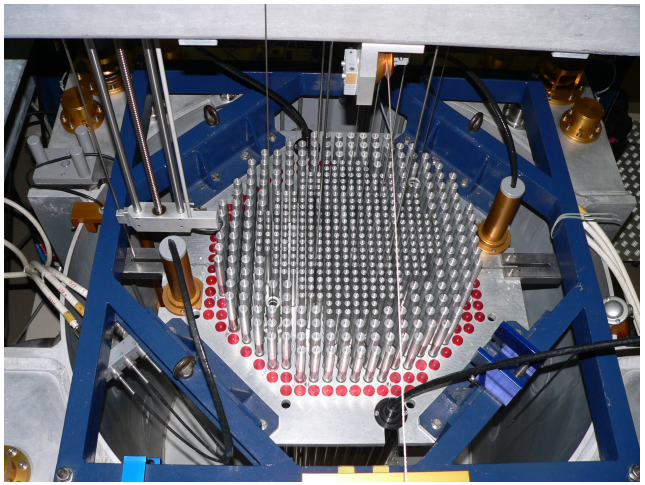
... can again linearize the $d_{ij}^{c_3}$ expression with SOS

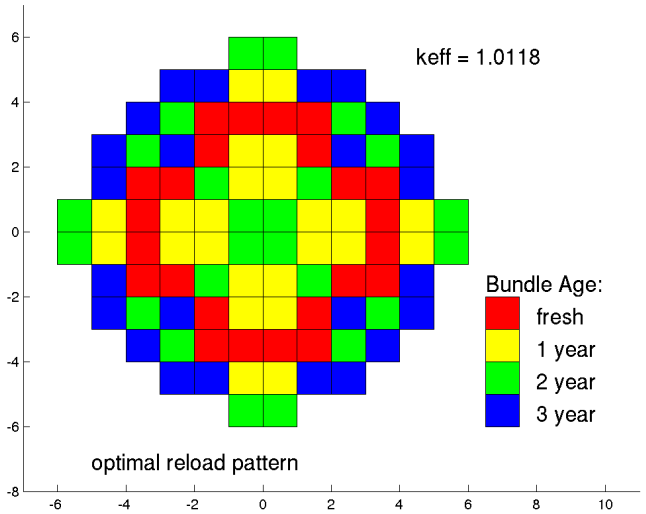
... alternative uses complementarity

Other MINLP Applications

MINLP

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & f(x) \\ \text{subject to} \quad & c(x) \leq 0 \\ & x \in X \\ & x_i \in \mathbb{Z} \text{ for all } i \in I \end{aligned}$$





Applications:

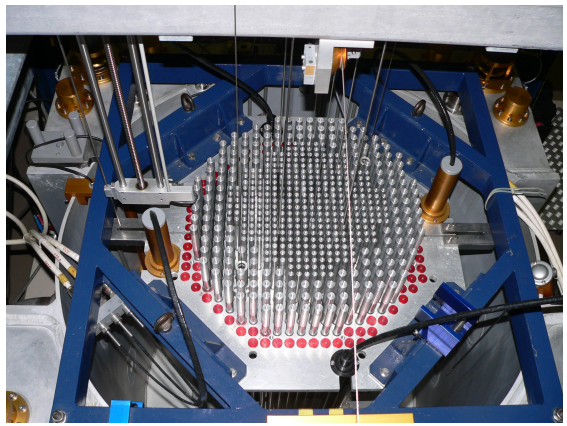
- reactor core reload operation
- power grid operation & design
- buildings co-generation
- optimal oil-spill response
- gas transmission networks

Application: Nuclear Reactor-Core Reloading

Mixed Integer Nonlinear Program (MINLP)

$$\min_s \; f(x) \quad \text{s.t. } c(x) \leq 0,\; x \in X,\; \text{and } x_I \text{ integer}$$

- simplified physics (neutron transport)
- maximize reactor efficiency after reload
- subject to diffusion process & safety
  $\Rightarrow$ integer & nonlinear model

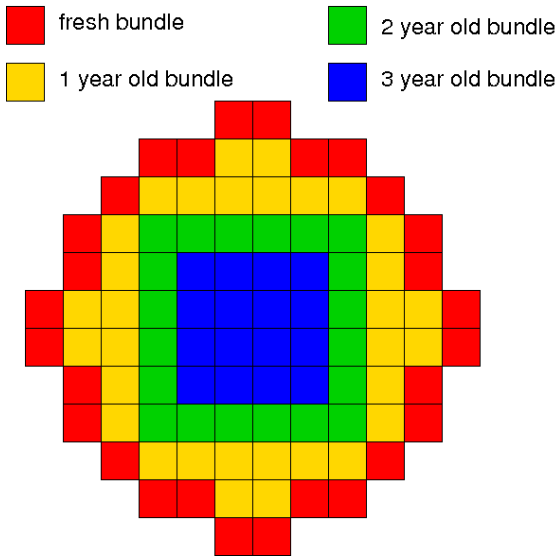


Application: Nuclear Reactor-Core Reloading

Mixed Integer Nonlinear Program (MINLP)

$$\min_{s} \; f(x) \quad \text{s.t. } c(x) \leq 0,\; x \in X, \text{ and } x_I \text{ integer}$$

- simplified physics (neutron transport)
- maximize reactor efficiency after reload
- subject to diffusion process & safety
  $\Rightarrow$ integer & nonlinear model
- avoid reactor becoming sub-critical

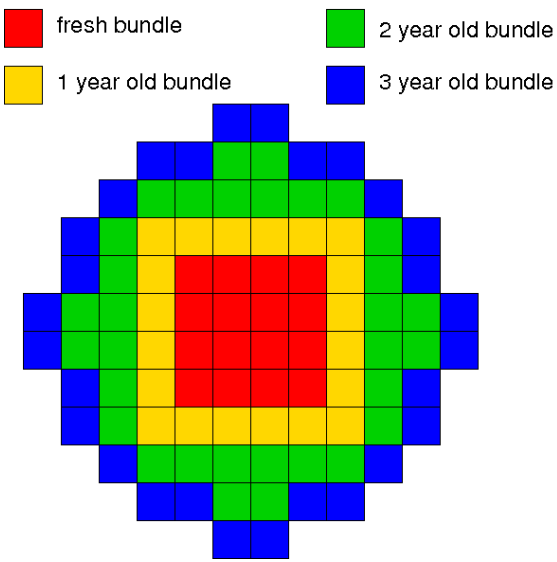


Application: Nuclear Reactor-Core Reloading

Mixed Integer Nonlinear Program (MINLP)

$$\min_s \; f(x) \quad \text{s.t. } c(x) \leq 0,\; x \in X, \text{ and } x_I \text{ integer}$$

- simplified physics (neutron transport)
- maximize reactor efficiency after reload
- subject to diffusion process & safety
  $\Rightarrow$ integer & nonlinear model
- avoid reactor becoming sub-critical
- avoid reactor becoming super-critical

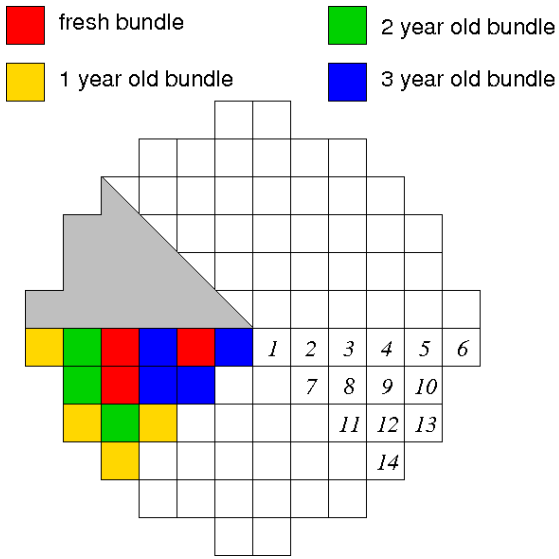


Application: Nuclear Reactor-Core Reloading

Mixed Integer Nonlinear Program (MINLP)

$$\min_s \; f(x) \quad \text{s.t. } c(x) \leq 0,\; x \in X,\; \text{and } x_I \text{ integer}$$

- simplified physics (neutron transport)
- maximize reactor efficiency after reload
- subject to diffusion process & safety
  $\Rightarrow$ integer & nonlinear model
- avoid reactor becoming sub-critical
- avoid reactor becoming super-critical
- look for cycles for moving bundles:
  e.g. $4 \rightarrow 6 \rightarrow 8 \rightarrow 10$
  means bundle moved from 4 to 6 etc

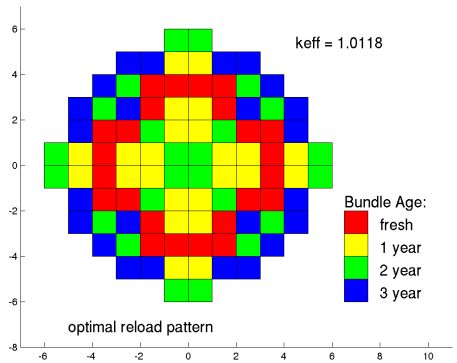


Application: Nuclear Reactor-Core Reloading

Mixed Integer Nonlinear Program (MINLP)

$$\min_s \; f(x) \quad \text{s.t. } c(x) \leq 0,\; x \in X, \text{ and } x_I \text{ integer}$$

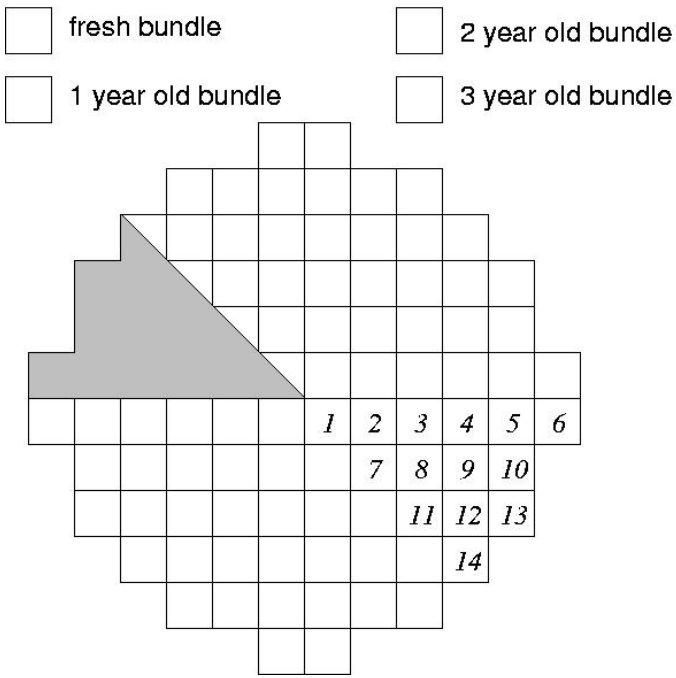
- simplified physics (neutron transport)
- maximize reactor efficiency after reload
- subject to diffusion process & safety
  $\Rightarrow$ integer & nonlinear model
- avoid reactor becoming sub-critical
- avoid reactor becoming super-critical
- look for cycles for moving bundles:
  e.g. $4 \rightarrow 6 \rightarrow 8 \rightarrow 10$
  means bundle moved from 4 to 6 etc



Nuclear Core Reload Optimization

- look for cycles for moving bundles:
  - e.g.
    $4 \rightarrow 6 \rightarrow 8 \rightarrow 10$
  - means bundle moved from 4 to 6 to ...
- model with integer variables $x_{ilm} \in \{0, 1\}$
- $x_{ilm} = 1$: node i has bundle l of cycle m
- exactly one bundle per node:

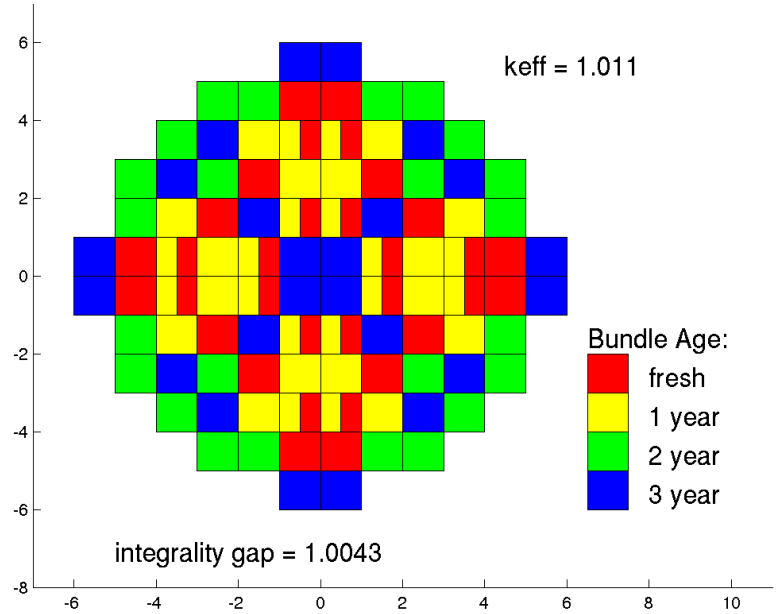
$$\sum_{l=1}^{L} \sum_{m=1}^{M} x_{ilm} = 1 \qquad \forall i \in I$$



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

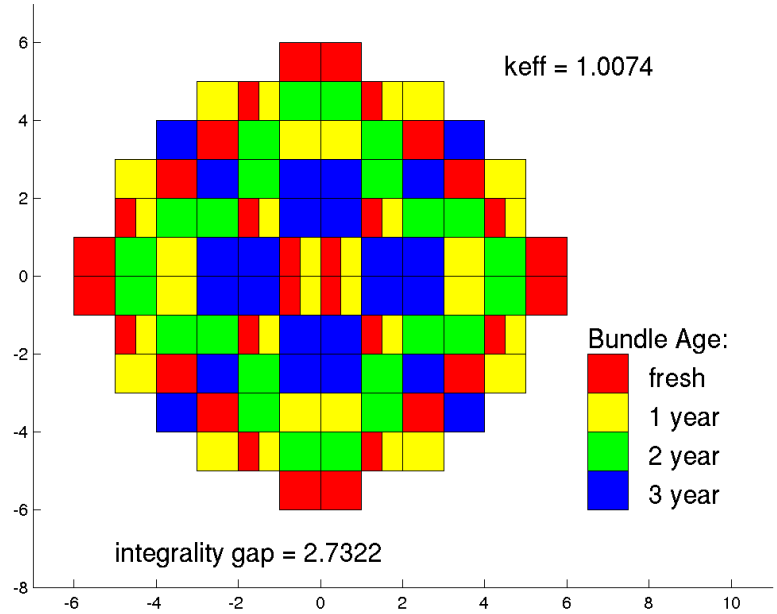
- depth-first, maximum fractional branching



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

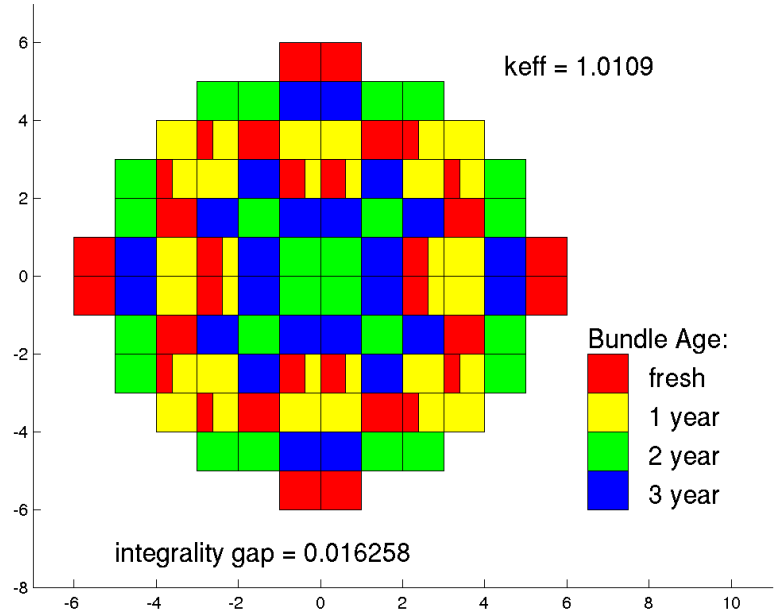
- depth-first, maximum fractional branching



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

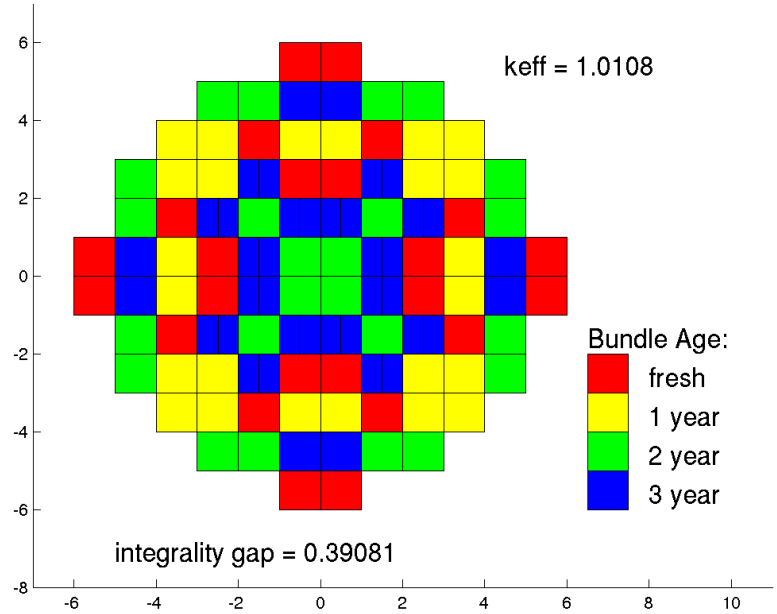
- depth-first, maximum fractional branching



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

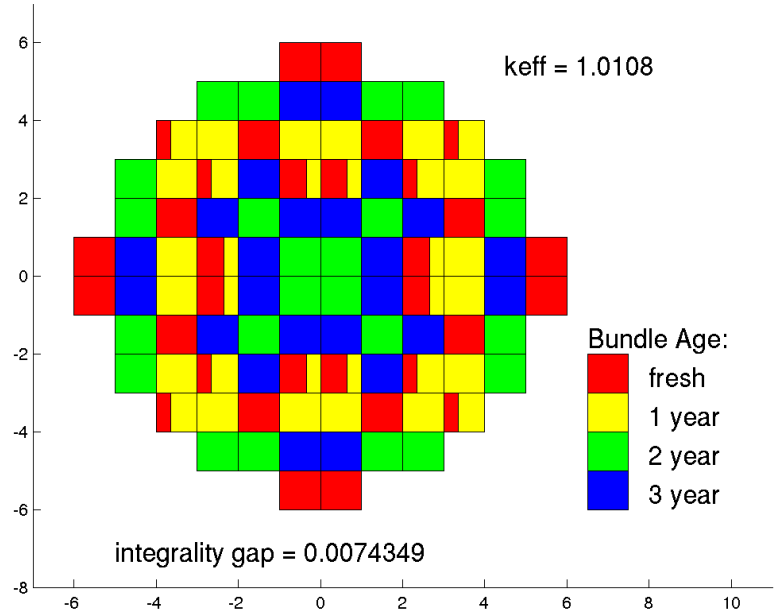
- depth-first, maximum fractional branching



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

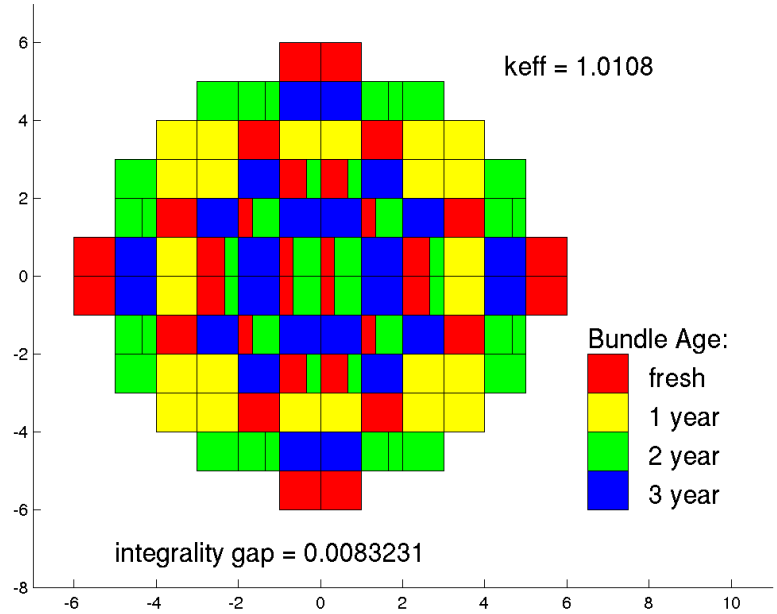
- depth-first, maximum fractional branching



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

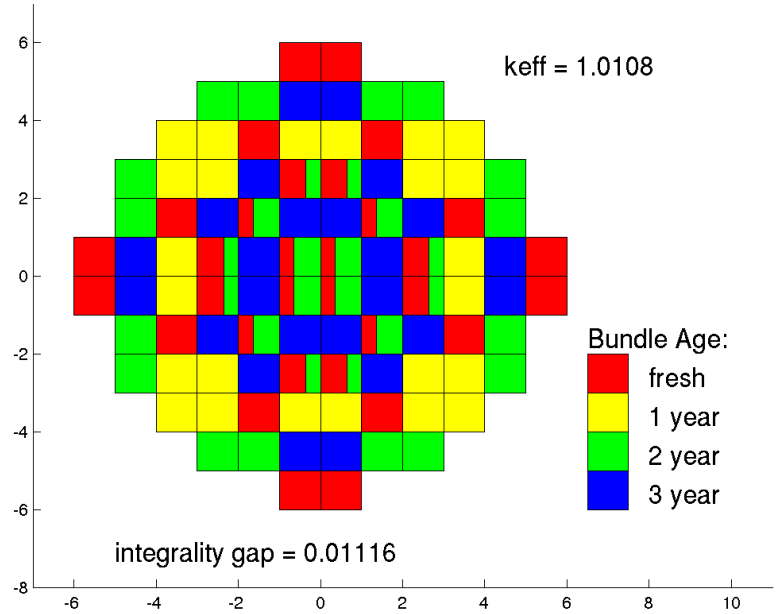
- depth-first, maximum fractional branching



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

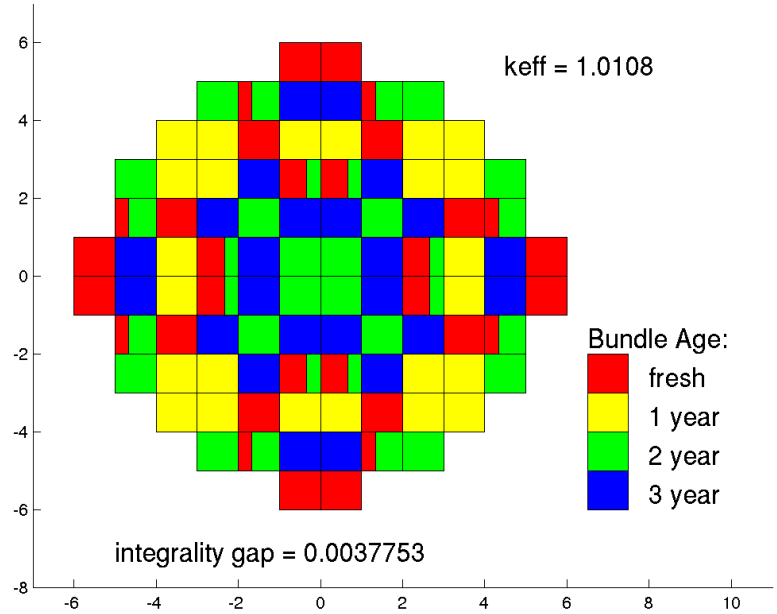
- depth-first, maximum fractional branching



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

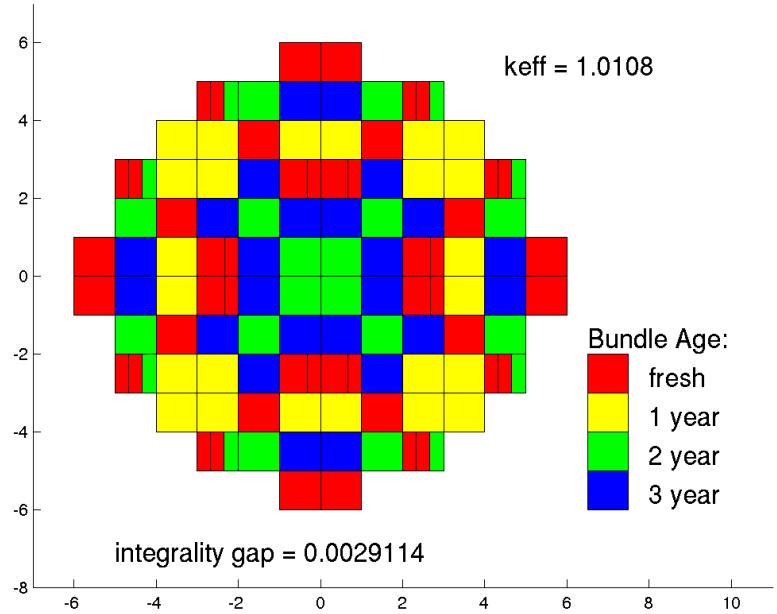
- depth-first, maximum fractional branching



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

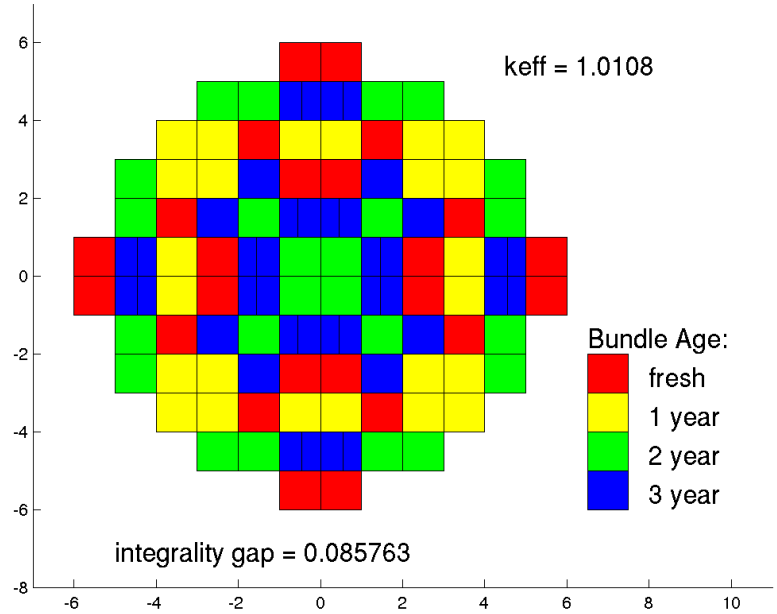
- depth-first, maximum fractional branching



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

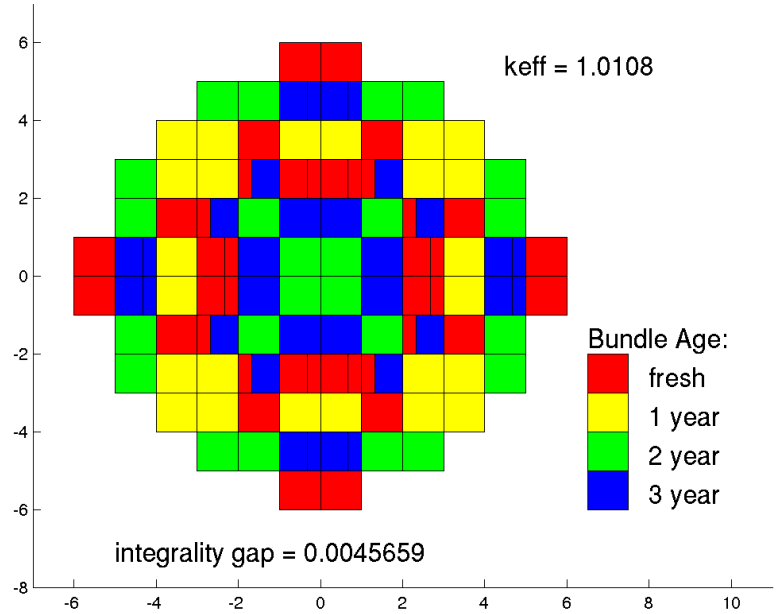
- depth-first, maximum fractional branching



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

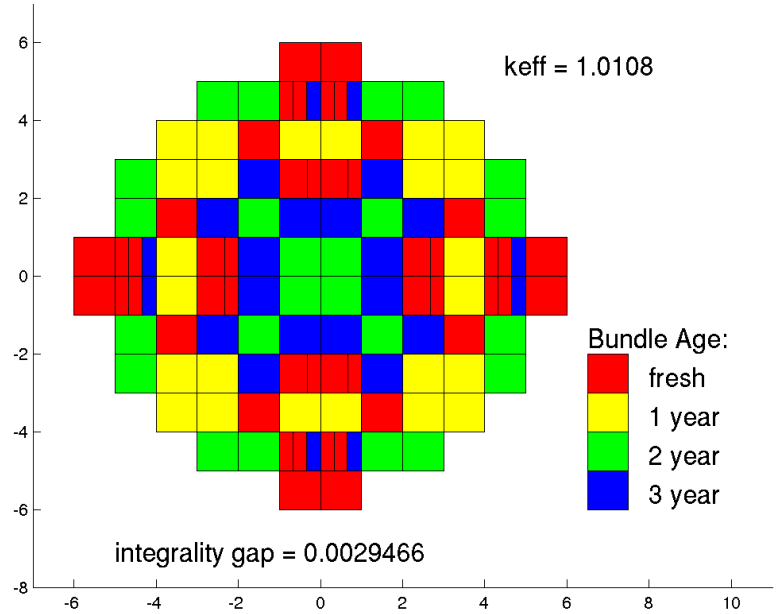
- depth-first, maximum fractional branching



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

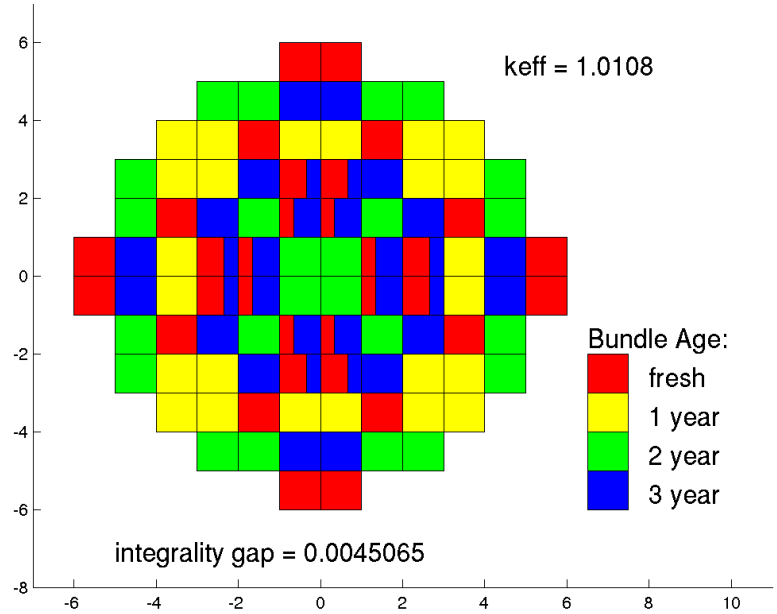
- depth-first, maximum fractional branching



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

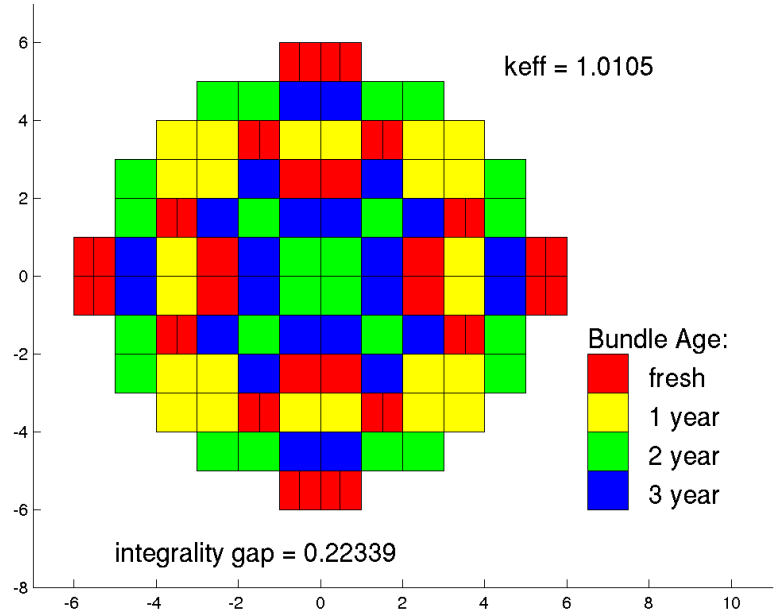
- depth-first, maximum fractional branching



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

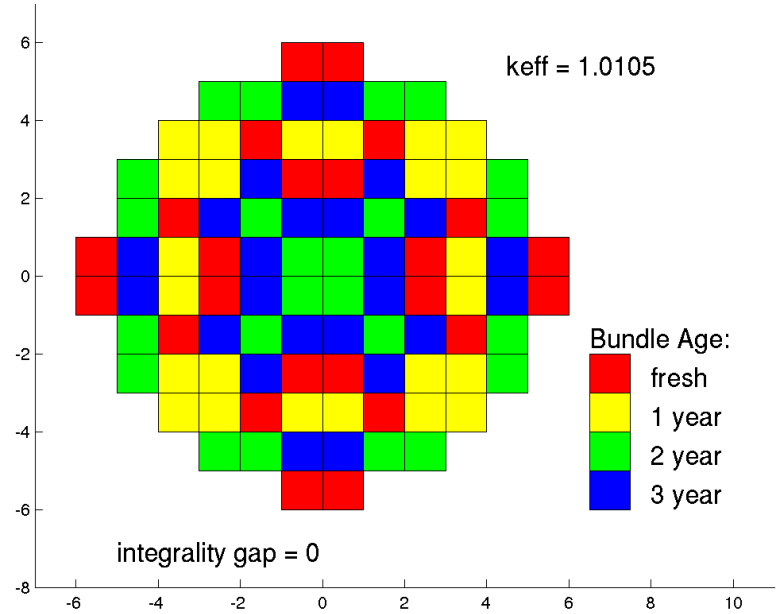
- depth-first, maximum fractional branching



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

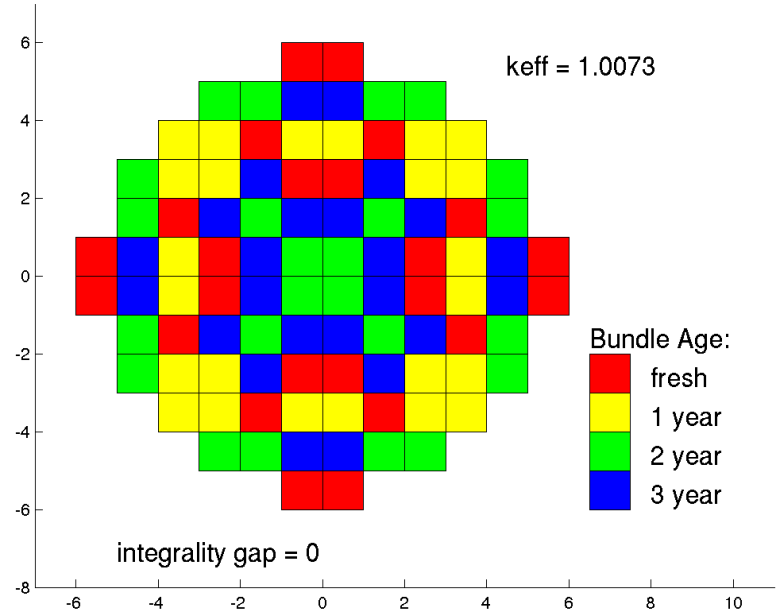
- depth-first, maximum fractional branching
- uncover first integer feasible node & backtrack



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

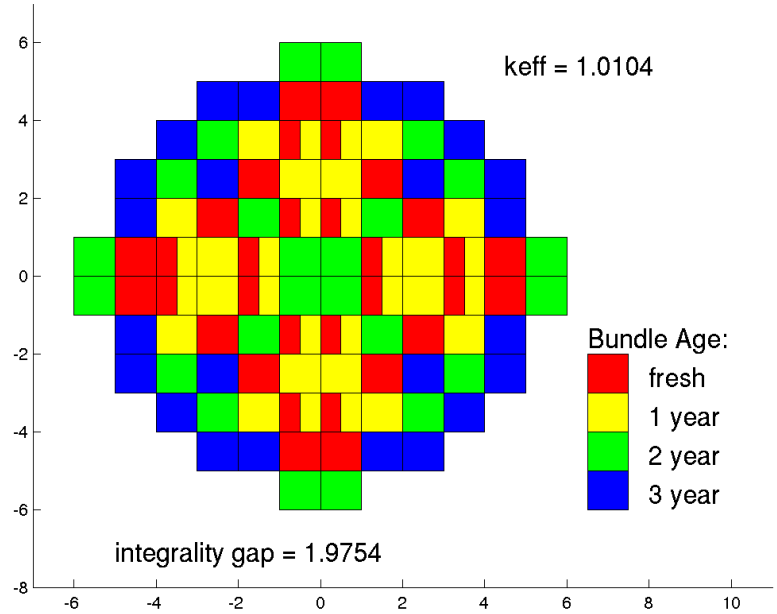
- depth-first, maximum fractional branching
- uncover first integer feasible node & backtrack



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

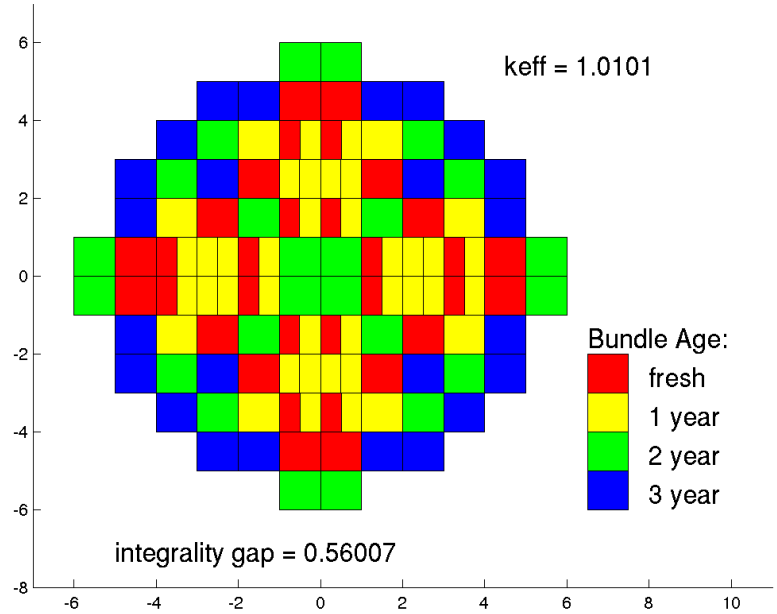
- depth-first, maximum fractional branching
- uncover first integer feasible node & backtrack



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

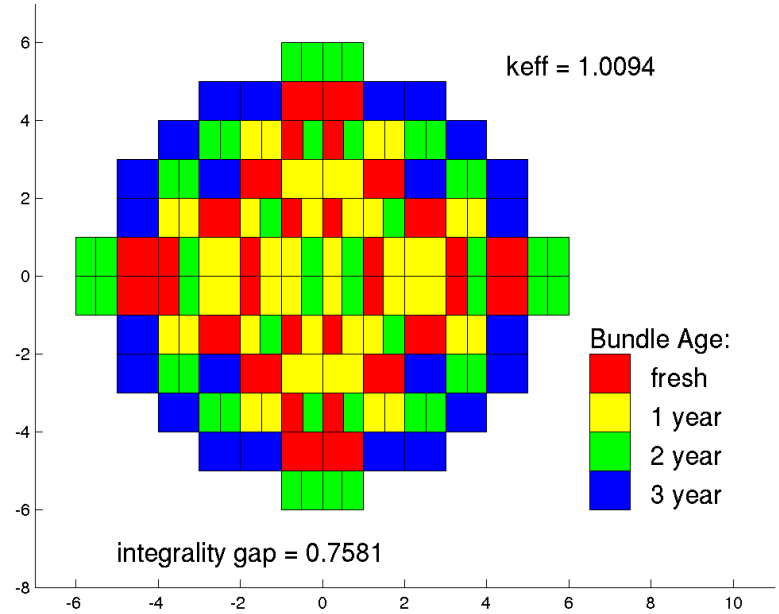
- depth-first, maximum fractional branching
- uncover first integer feasible node & backtrack



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

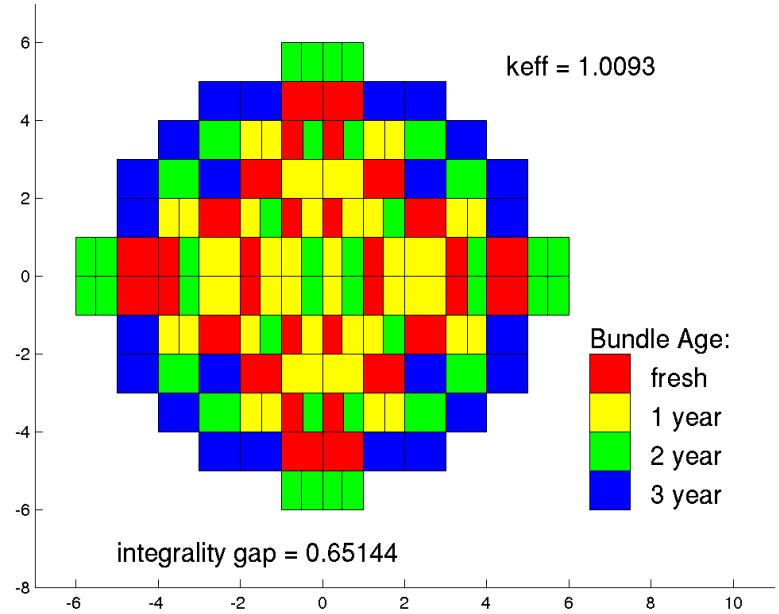
- depth-first, maximum fractional branching
- uncover first integer feasible node & backtrack



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

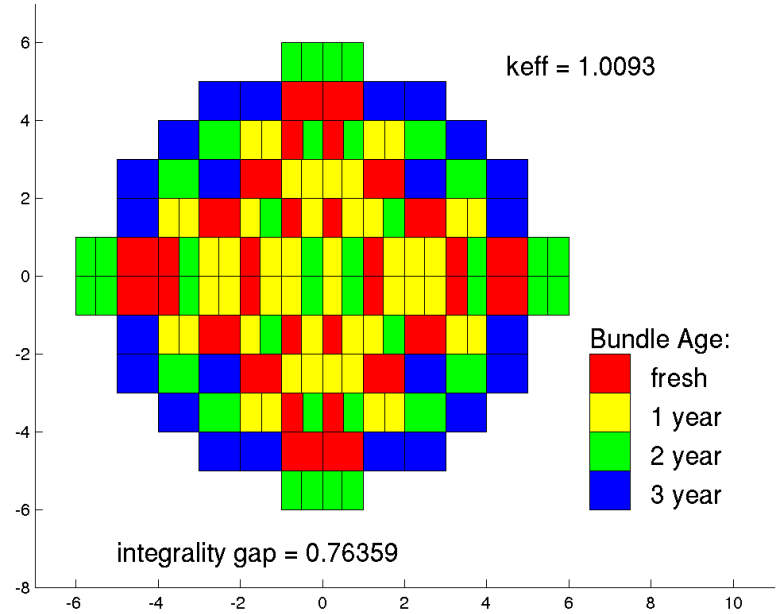
- depth-first, maximum fractional branching
- uncover first integer feasible node & backtrack



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

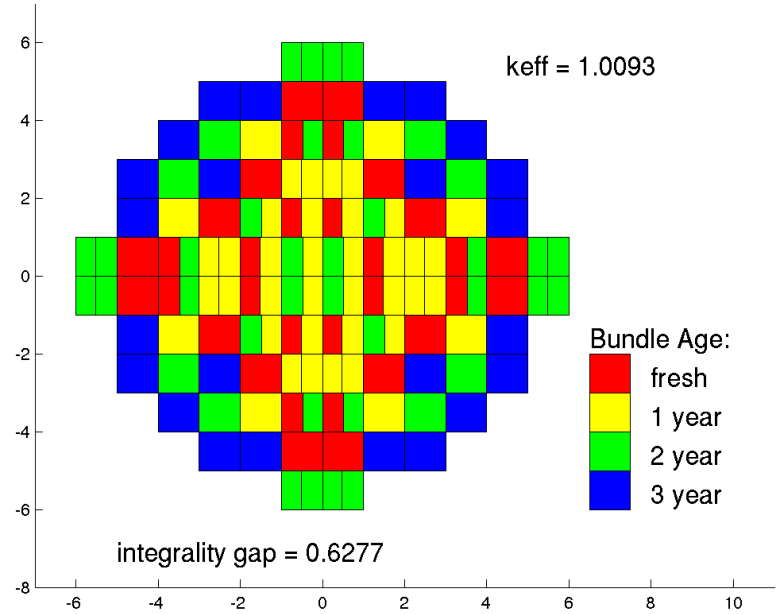
- depth-first, maximum fractional branching
- uncover first integer feasible node & backtrack



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

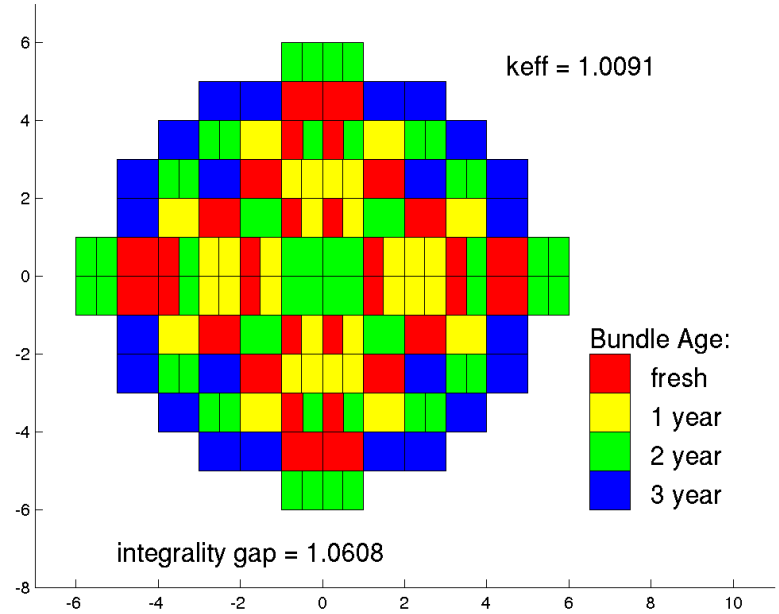
- depth-first, maximum fractional branching
- uncover first integer feasible node & backtrack



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

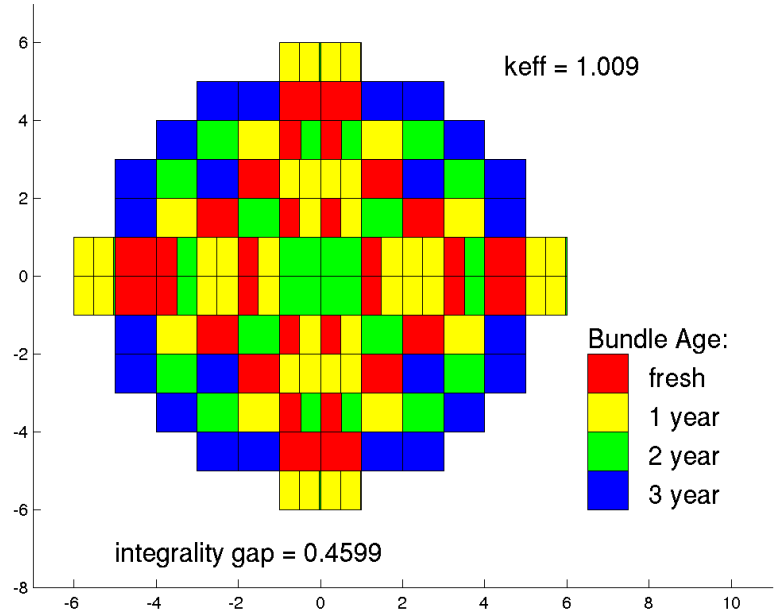
- depth-first, maximum fractional branching
- uncover first integer feasible node & backtrack



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

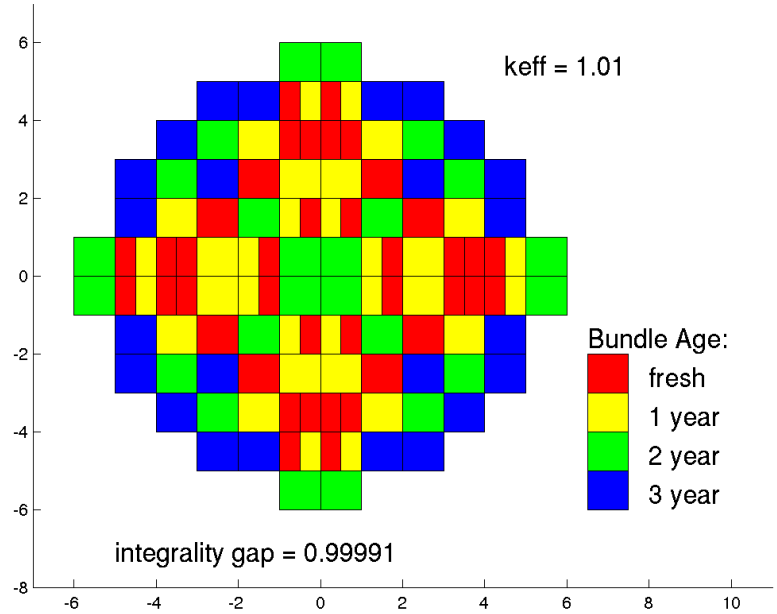
- depth-first, maximum fractional branching
- uncover first integer feasible node & backtrack



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

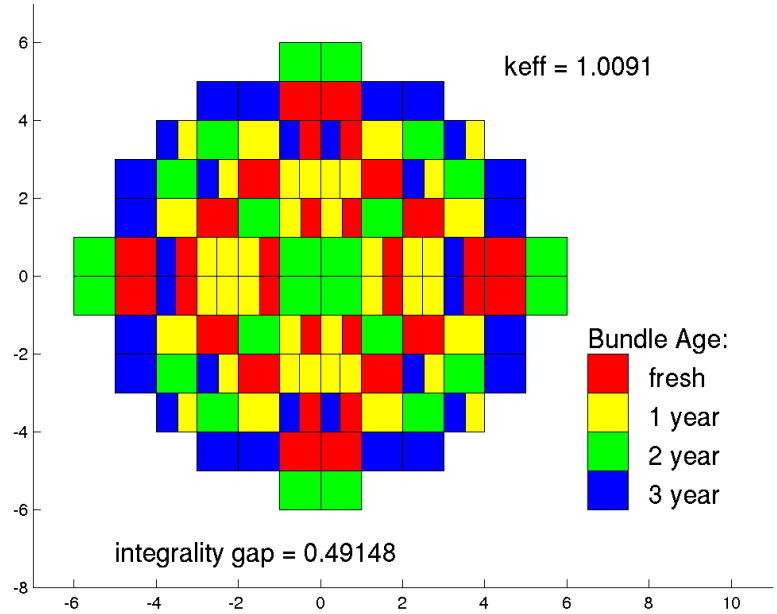
- depth-first, maximum fractional branching
- uncover first integer feasible node & backtrack



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

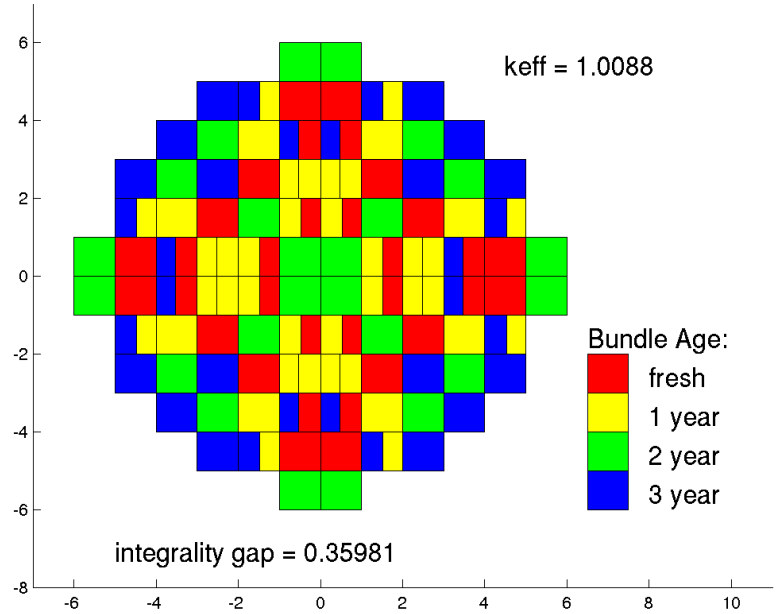
- depth-first, maximum fractional branching
- uncover first integer feasible node & backtrack



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

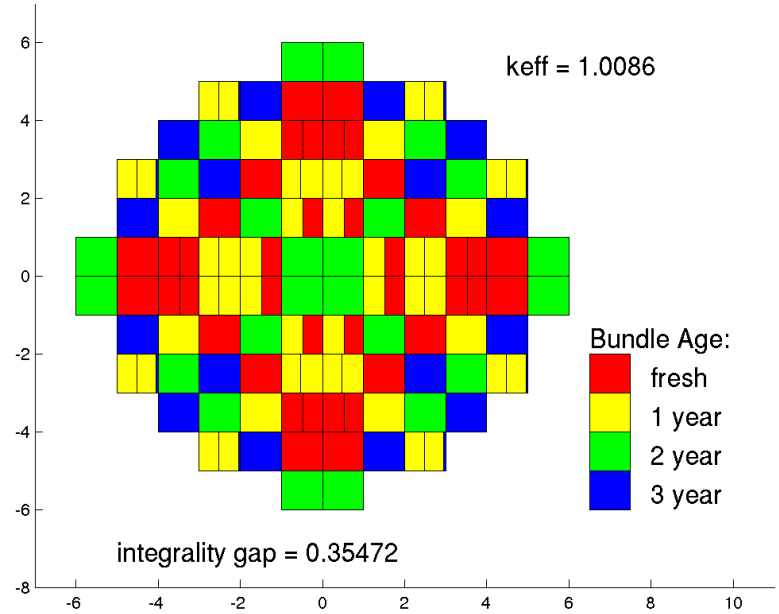
- depth-first, maximum fractional branching
- uncover first integer feasible node & backtrack



Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

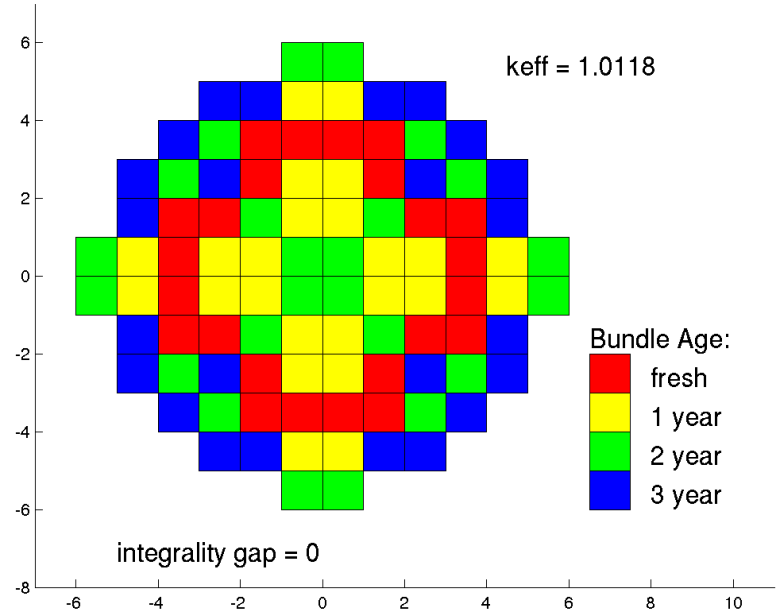
- depth-first, maximum fractional branching
- uncover first integer feasible node & backtrack



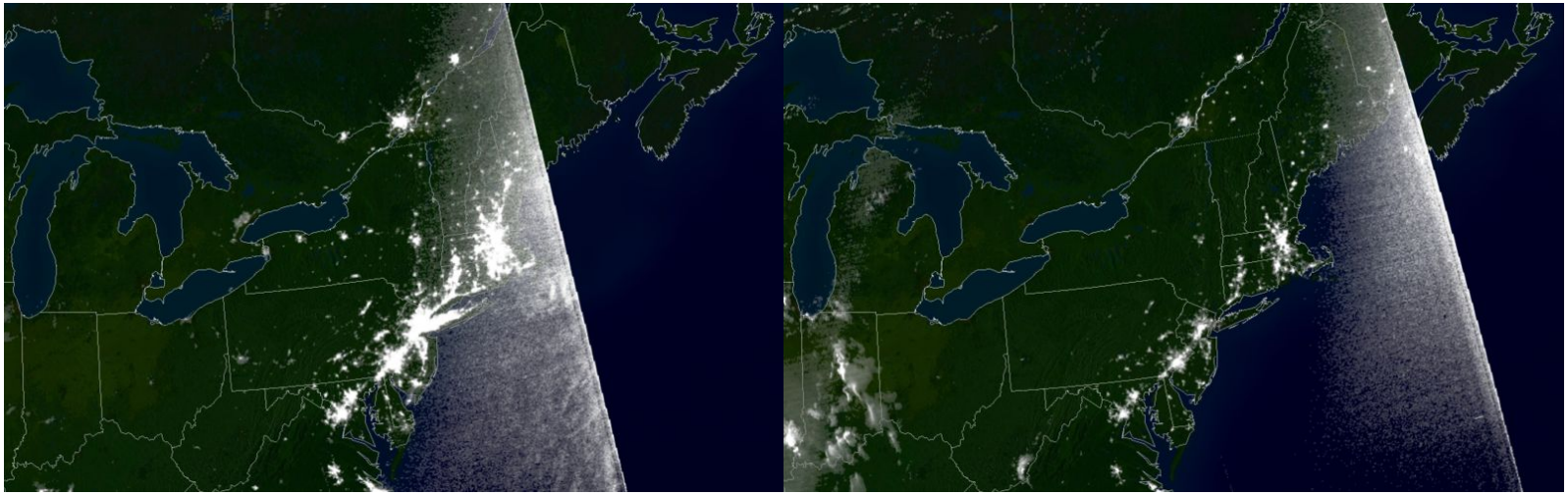
Nuclear Core Reload Optimization

Branch-and-bound: small search-tree ... watch progress:

- depth-first, maximum fractional branching
- uncover first integer feasible node & backtrack
- uncover better feasible point (optimal)
- prune rest of tree



Blackout Prevention in National Power Grid



2003 blackout: before and during

- 2003 blackout cost $4-10 billion and affected 50 million people
- prevent with contingency analysis
  - find least number of transmission lines whose removal results in failure
  - binary variables model removal of lines
  - nonlinearities model power flow
  - results in large integer optimization problem
- current analysis limited to 10s of lines

... similar models arise in many other power-grid applications

Power-Grid Transmission Network Expansion

Problem. Given a power grid network and demand forecast, design an expanded network by adding lines to the existing network that allows secure transmission of increased demand.

Traditional Approach. Simplify nonlinear (AC) power flow model:

$$F(U_k, U_l, \theta_k, \theta_l) := b_{kl} U_k U_l \sin(\theta_k - \theta_l) + g_{kl} U_k^2 - g_{kl} U_k U_l \cos(\theta_k - \theta_l)$$

by setting $\sin(x) \simeq x$ and $\cos(x) \simeq 1$ and $U \simeq 1$

... to obtain the linearized (DC) power flow model.

Nonlinear Optimization Approach. Work directly with nonlinear model $-M(1 - z_{k,l}) \leq f_{k,l} - F(U_k, U_l, \theta_k, \theta_l) \leq M(1 - z_{k,l})$

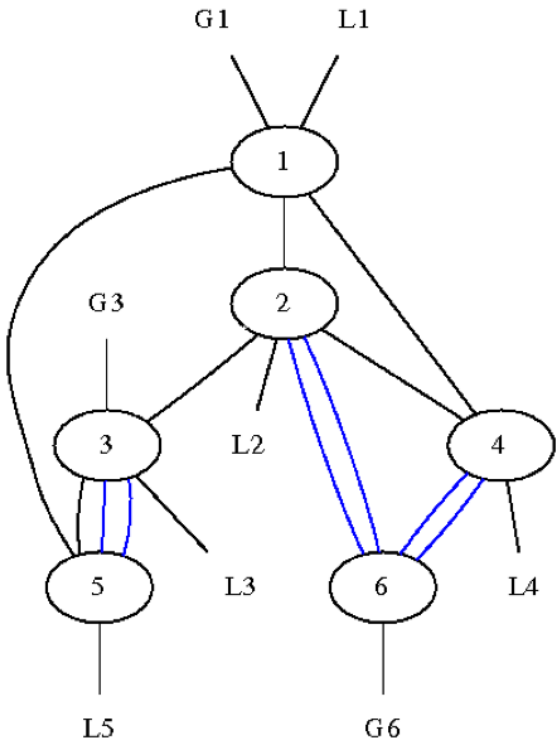
... $M > 0$ constant; $z_{k,l} \in \{0, 1\}$ switch lines on/off ... by setting flow $-Mz_{k,l} \leq f_{k,l} \leq Mz + k, l$

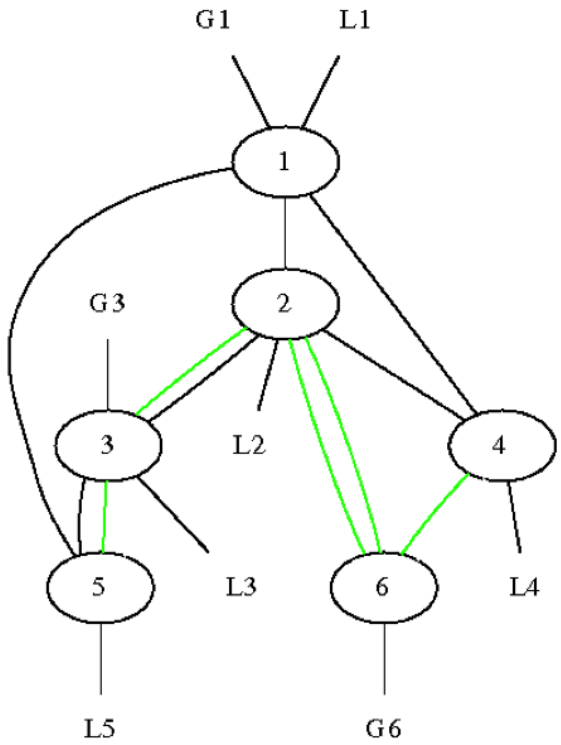
Questions.

Can we solve the nonlinear models? What is a good formulation? Does it matter which flow model we use?

Power-Grid Transmission Network Expansion

Expansion Results for linear vs. nonlinear power flow models





- Solve realistic AC power flow expansion models on desktop
- Significant difference between DC and AC solution
- Linearized DC model not feasible in AC power flow
- Approximation assumptions do not hold when topology changes

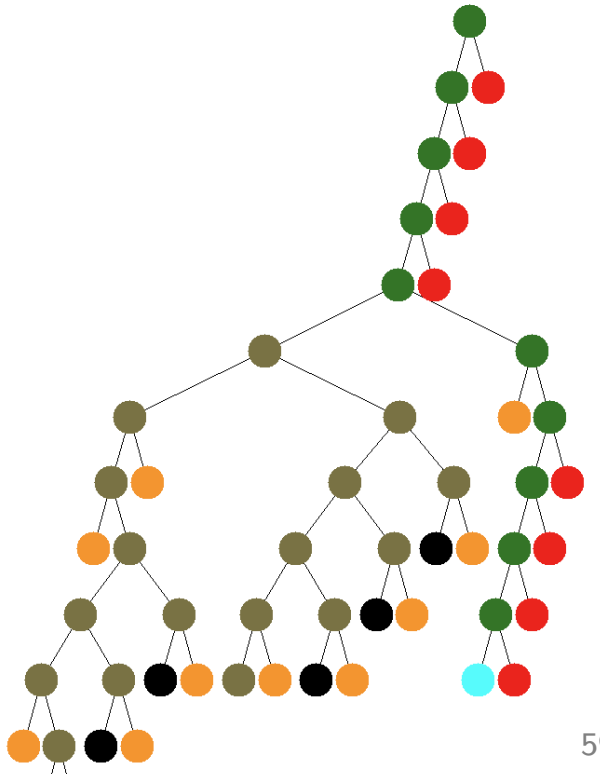
Example: Optimal Transmission Switching

Optimize network by controlling transmission line circuit breakers

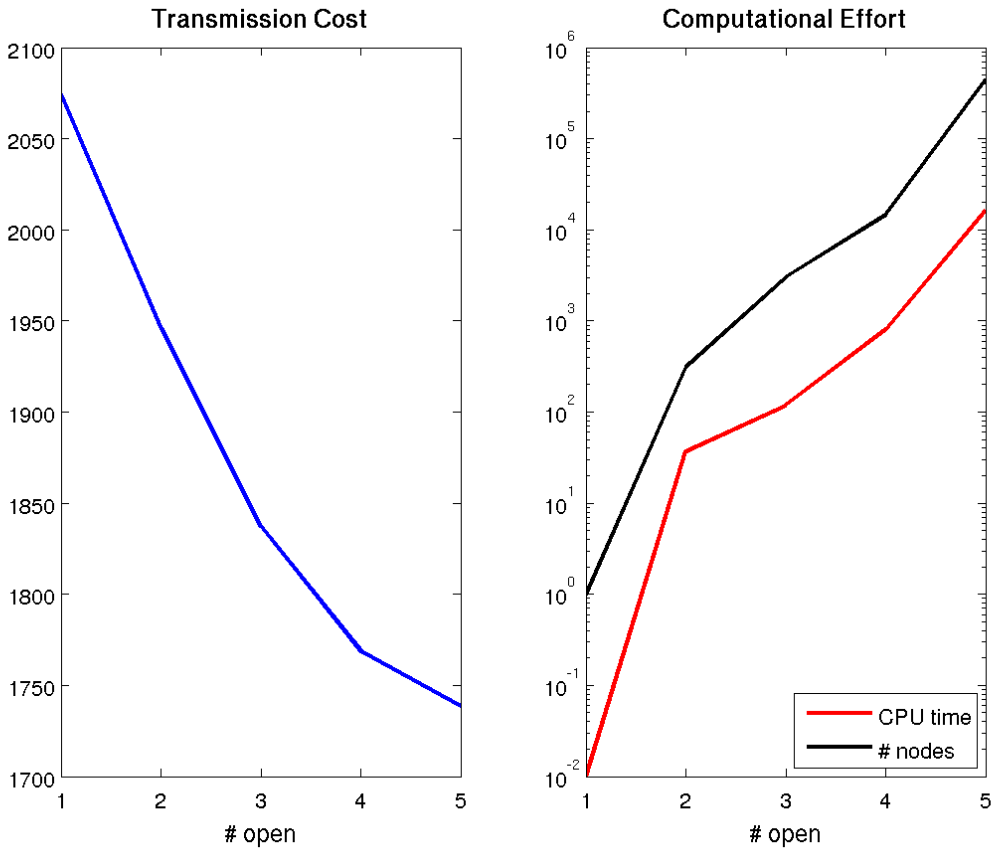
- Improve system dispatch by changing network topology.
- Optimize economic efficiency of power system dispatch: e.g. minimize generator costs.
- Formulate as mixed-integer optimization problem:
  - Binary variables represent state of lines: open/closed.
  - Physical constraints of power system: AC or DC power flow.

Challenging mixed-integer nonlinear optimization problem.

Goal:
Benchmark state-of-the-art solvers:
e.g. Argonne's MINOTAUR.



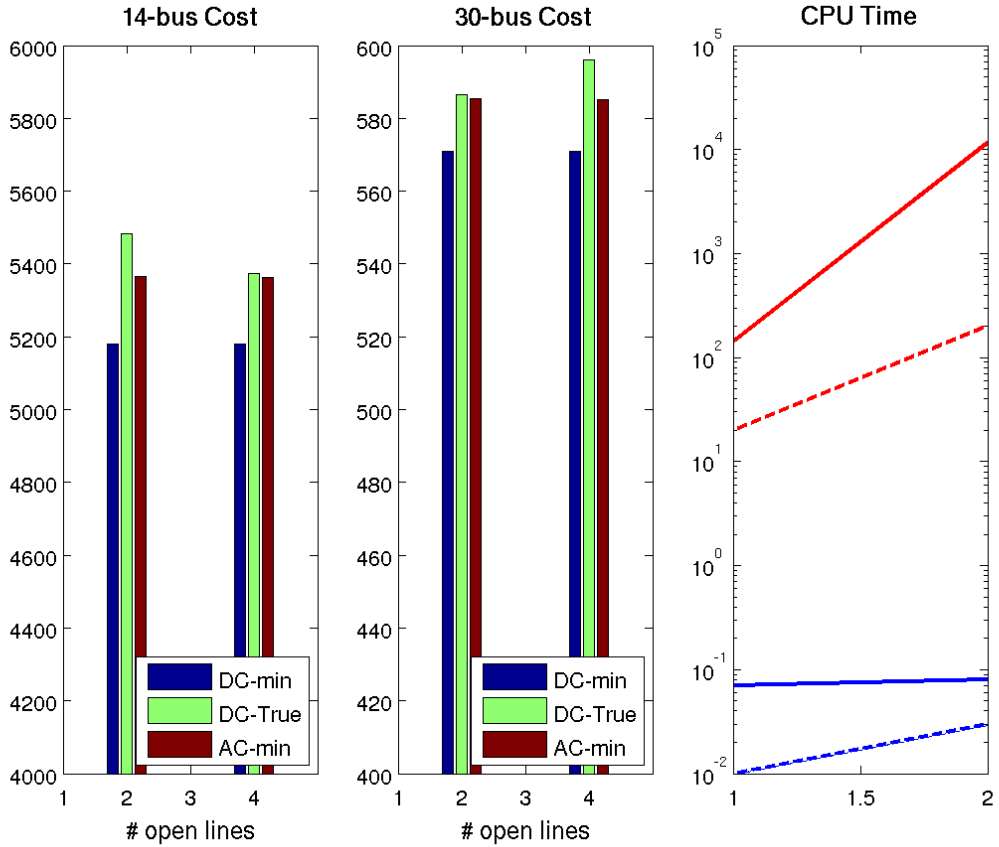
Effect of Increased Transmission Switching



Our DC models agree with observations in literature:

- Increased switching $\Rightarrow$ reduced transmission cost.
- Increased switching $\Rightarrow$ explosion in computational effort.

Comparison of AC and DC Switching



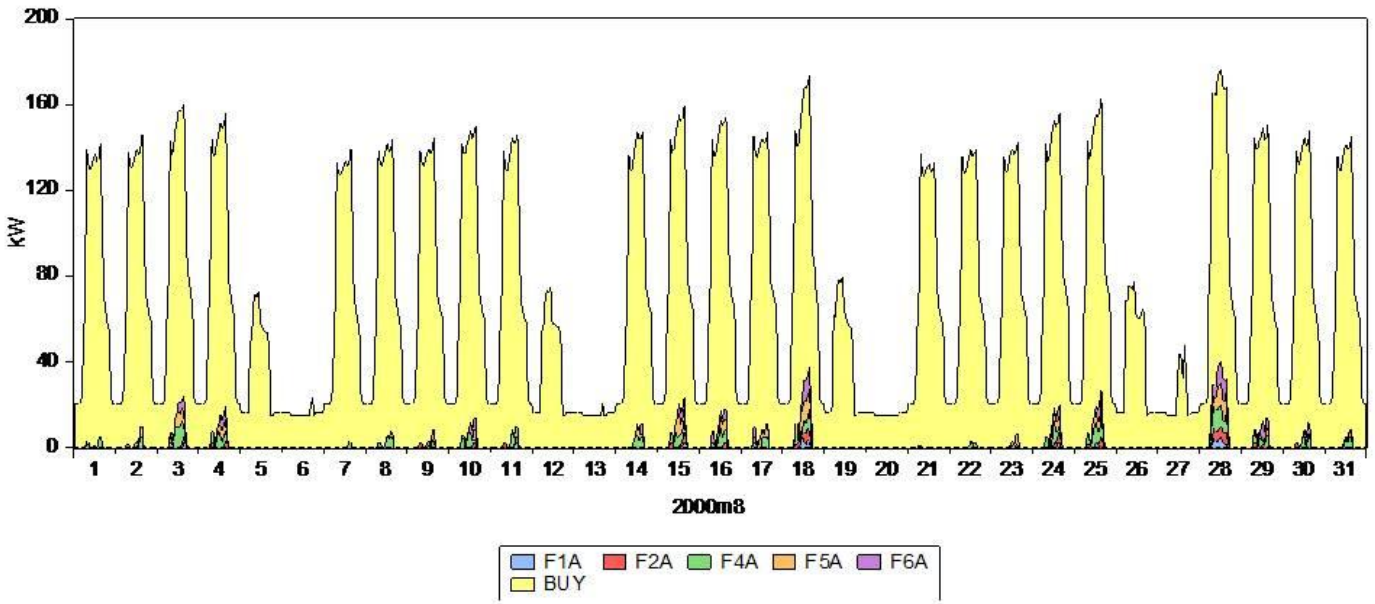
- DC transmission switching:
  DC cost reduced, but "true" AC cost increases!
- AC transmission switching: switch different set of lines.

$\Rightarrow$ nonlinear (AC) models: 2% cost reduction; but harder problem.

Co-Generation for Commercial Buildings

Goal: Net-zero energy buildings by 2020 $\Rightarrow$ 60% reduction of CO_2

- Co-generation units: fuel-cell, solar panel, wind, storage unit.
- Which units to buy to minimize energy and purchase cost? Binary variables model type of equipment & size (discrete).
- Ramping for fuel-cell & storage unit $\Rightarrow$ nonlinearities.
- Optimal hourly operation of units $\Rightarrow$ on/off constraints.



Pruitt, Newman, Braun (Colorado School of Mines & NREL)

Co-Generation for Commercial Buildings

1-Day Data Set

	MINOTAUR					
	BnB	QPD	Bonmin	Baron	Couenne	MINLPBB
Objf	836.30	968.73	836.43	840.64	844.92	836.17
CPU	117.87	2.59	174.496	> 10hrs	> 10hrs	147.98
Nodes	204	5	61	363,358	932,400	129

4-Day Data Set

	MINOTAUR					
	BnB	QPD	Bonmin	Baron	Couenne	MINLPBB
Objf	3344.81	3344.81	3304.69	3304.69	Inf	3266.47
CPU	11.45	23.87	7522.89	> 10hrs	> 10hrs	26293.08
Nodes	1	1	9	17,875	88,454	3,062

7 Day Data Set

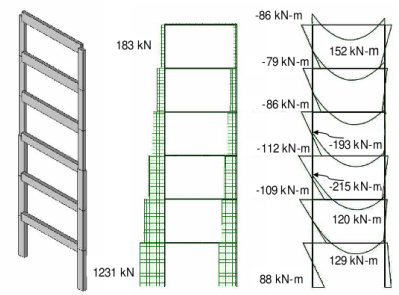
	MINOTAUR					
	BnB	QPD	Bonmin	Baron	Couenne	MINLPBB
Objf	6178.37	6178.37	Inf	5748.18	Inf	5726.0
CPU	168.38	54.55	> 10hrs	> 10hrs	> 10hrs	> 10hrs
Nodes	1	3	350	13,231	38,693	827

... tough problem, and not even the right one!

Optimal Design of Concrete Structures

Optimal design of reinforced concrete structures

- minimize cost: material (concrete & steel), labor, & form-work
- subject to:
  - geometry & sizes of elements
  - stiffness & displacement correlations (FEM)
  - RC element strength & ACI-code safety
    $\Rightarrow$ modeled as complementarity constraints
- discrete variables: reinforced steel dimensions
- integer variables: no. of re-enforcement bars
- binary variables: form-work re-use



Complex nonconvex MINLP with complementarity constraints

Can be "solved" using robust nonlinear branch-and-bound

Optimization of IEEE 802.11 Broadband Networks

Optimize 802.11 broadband networks for resource sharing meshes

- objective: minimizing co-channel and inter-channel interference
- integrality: assign channels to basic nodes within a network
- 13 Direct Sequence Spread Spectrum (DSSS) overlapping channels
- co-channel interference: two access points with same channel
- inter-channel interference: cards with overlapping channels transmit simultaneously

$\Rightarrow$ general nonconvex MINLP

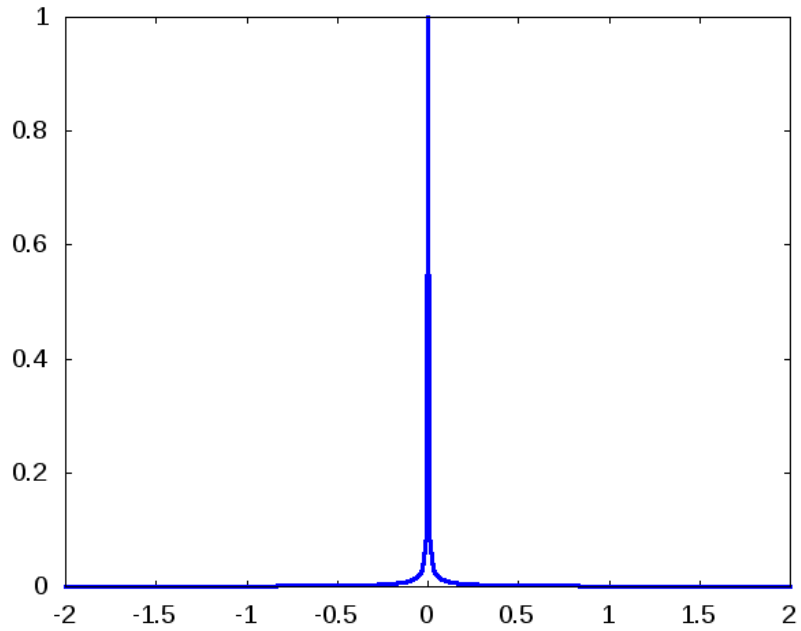
Original model has one horrible constraint ...

Optimization of IEEE 802.11 Broadband Networks

... and the horrible constraint is ...

$$z = \frac{1}{1 + 1000(x - y)^{10}}$$

- highly nonlinear/nonconvex

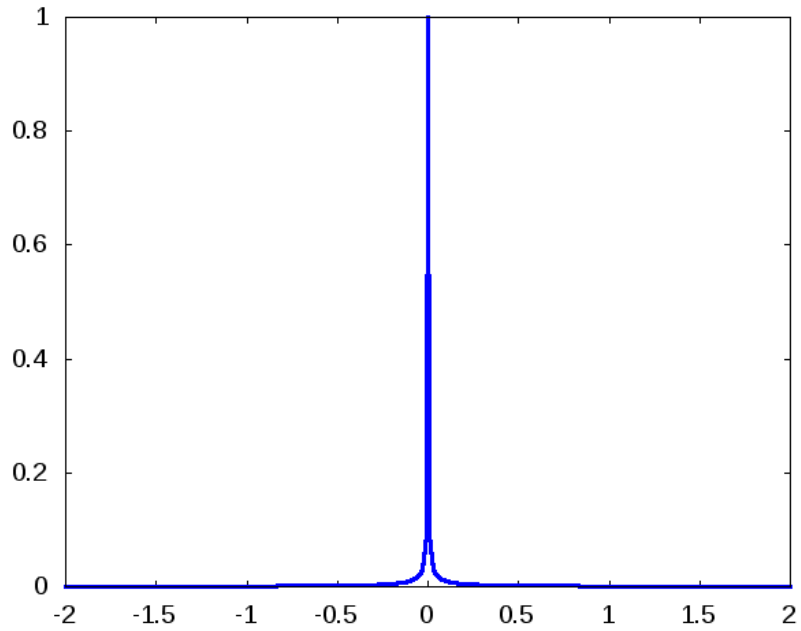


Optimization of IEEE 802.11 Broadband Networks

... and the horrible constraint is ...

$$z = \frac{1}{1 + 1000(x - y)^{10}}$$

- highly nonlinear/nonconvex
- $z = 1$, if $x = y$
- $z = 0$, if $x \neq y$

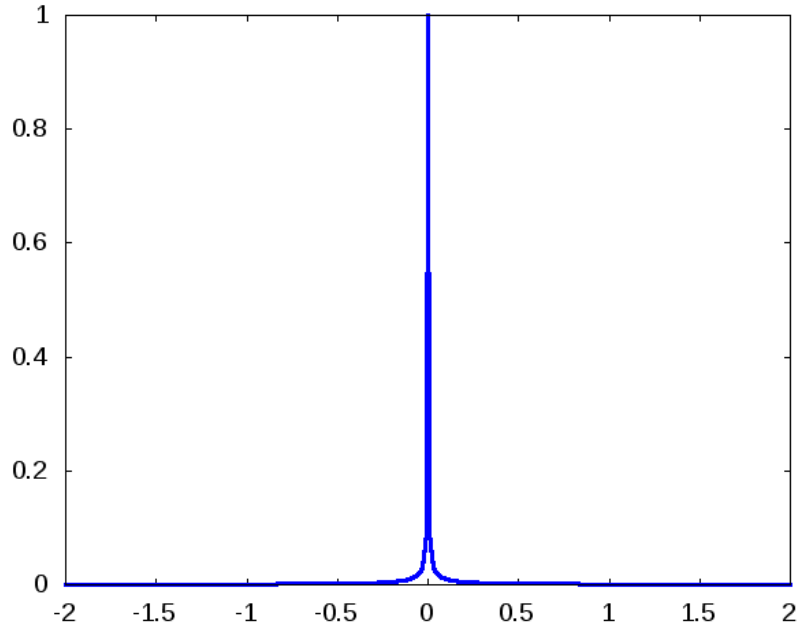


Optimization of IEEE 802.11 Broadband Networks

... and the horrible constraint is ...

$$z = \frac{1}{1 + 1000(x - y)^{10}}$$

- highly nonlinear/nonconvex
- $z = 1$, if $x = y$
- $z = 0$, if $x \neq y$
- x, y integer (channels)

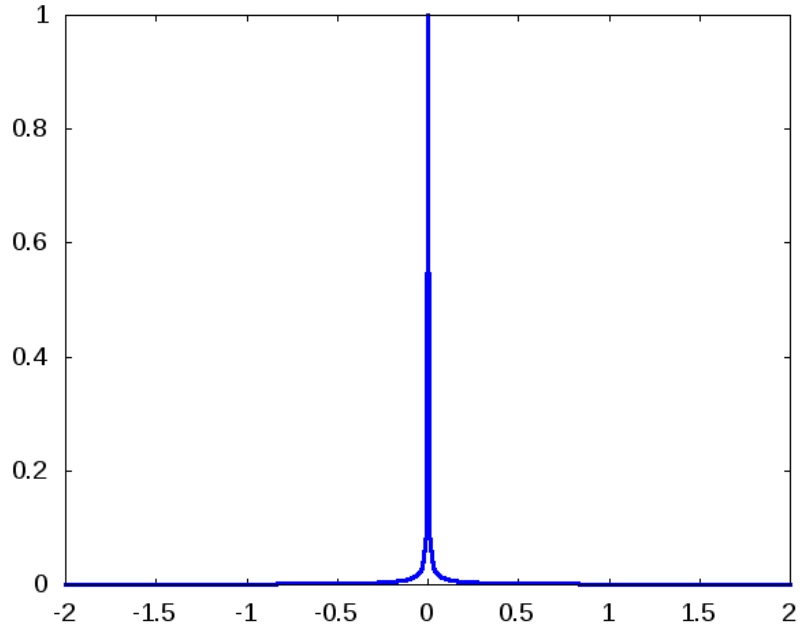


Optimization of IEEE 802.11 Broadband Networks

... and the horrible constraint is ...

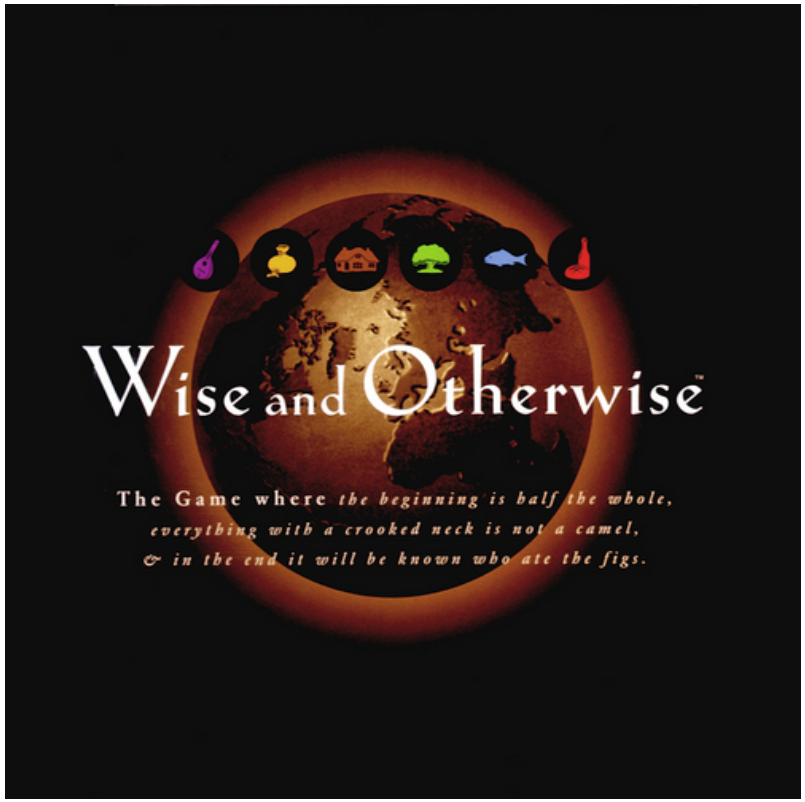
$$z = \frac{1}{1 + 1000(x - y)^{10}}$$

- highly nonlinear/nonconvex
- $z = 1$, if $x = y$
- $z = 0$, if $x \neq y$
- x, y integer (channels)
- model as MIP not NLP



Wise or Otherwise of IEEE 802.11 Broadband Networks

$$z = \frac{1}{1 + 1000(x - y)^{10}}$$

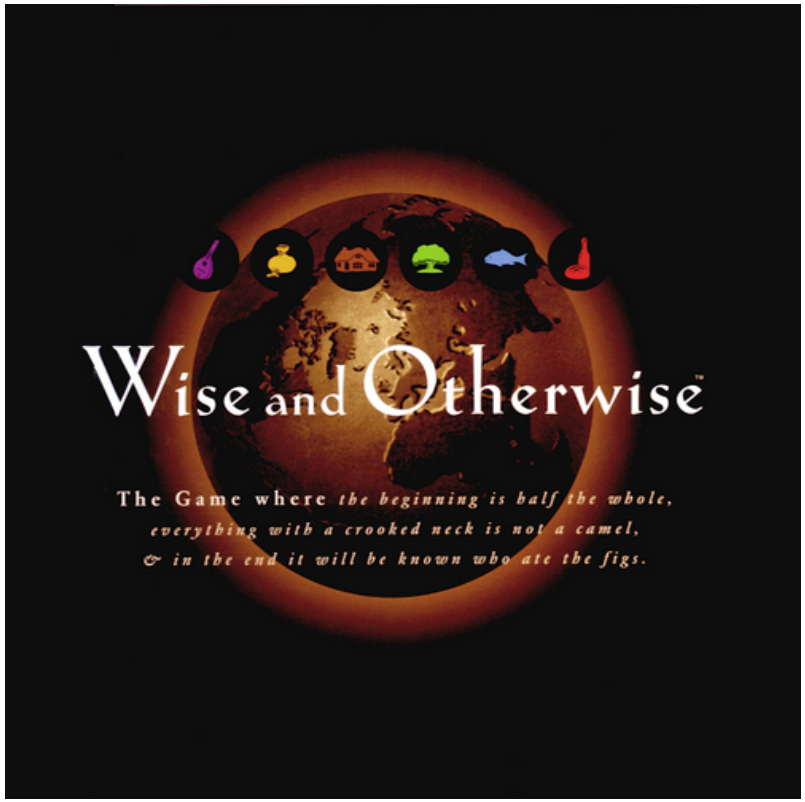


Chinese Proverb

Give a man a fish and he will eat for a day.

Wise or Otherwise of IEEE 802.11 Broadband Networks

$$z = \frac{1}{1 + 1000(x - y)^{10}}$$



Chinese Proverb

Give a man a fish and he will eat for a day.

Teach a man about nonlinear functions and you will lead a troubled life.

Application: Distillation Column Design

Mixed Integer Nonlinear Program (MINLP)

$$\underset{x}{\text{minimize}} \; f(x) \quad \text{subject to } c(x) \leq 0,\; x \in X,\; x_i \in \mathbb{Z}\; \forall\; i \in I$$

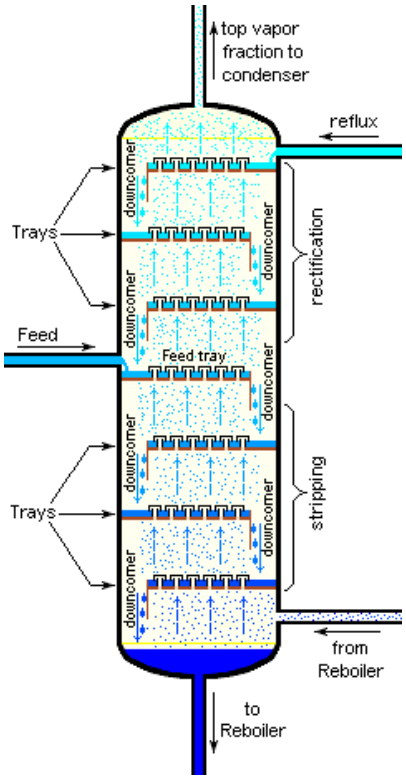
Small process design example:

- synthesis of distillation column

Application: Distillation Column Design

Mixed Integer Nonlinear Program (MINLP)

$$\underset{x}{\text{minimize}} \; f(x) \quad \text{subject to } c(x) \leq 0,\; x \in X,\; x_i \in \mathbb{Z}\; \forall\; i \in I$$



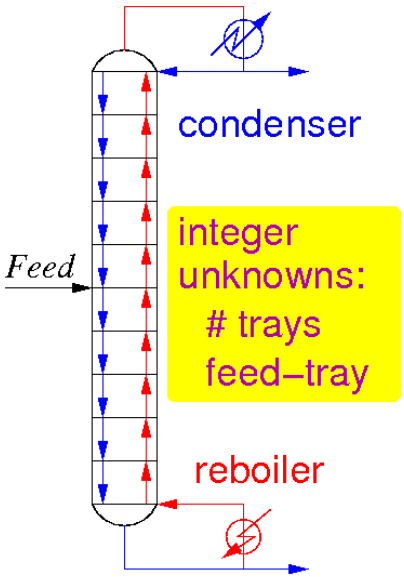
Small process design example:

- synthesis of distillation column
- nonlinear physics: phase equilibrium, component material balance

Application: Distillation Column Design

Mixed Integer Nonlinear Program (MINLP)

$$\underset{x}{\text{minimize}} \; f(x) \quad \text{subject to } c(x) \leq 0,\; x \in X,\; x_i \in \mathbb{Z}\; \forall\; i \in I$$



Small process design example:

- synthesis of distillation column
- nonlinear physics: phase equilibrium, component material balance
- integers model number of trays in columns
- $x_I \in \{0,1\}^p$ models position of feeds

Process network design for fossil power plants ...

Collections of MINLP Test Problems

AMPL Collections of MINLP Test Problems

1. MacMINLP `www.mcs.anl.gov/~leyffer/macminlp/`
2. IBM/CMU collection `egon.cheme.cmu.edu/ibm/page.htm`

GAMS Collections of MINLP Test Problems

1. GAMS MINLP-world `www.gamsworld.org/minlp/`
2. MINLP CyberInfrastructure `www.minlp.org/index.php`

Solve MINLPs online on the NEOS server,
`www.neos-server.org/neos/`

... and there are even a few CUTEr problems in SIF!

Outline

Outline of Remainder of Course

1. Basic Methods for Convex MINLPs
   - outer approximation, Benders decomposition
   - branch-and-bound
   - hybrid methods
2. Advanced Methods for Convex MINLPs
   - cutting planes for MINLP
   - branch-and-cut
   - cutting planes for conic constraints
3. Methods for Nonconvex MINLPs I
   - basic techniques
   - branch-and-bound
   - piecewise linear approximations
4. Methods for Nonconvex MINLPs II
   - special methods for special structure
   - modern solvers for nonconvex problems
5. Heuristics, Software, and Extensions
   - RINS, local branching, guided dives
   - modeling languages, open-source and other software
   - mixed-integer nonlinear optimal control

Outline

Summary and Exercises

Key points

- Modeling is very, very, very important
- Linearize, linearize, linearize as much as possible

Two little exercises

1. Assume $c(x) \leq 0$ convex and $\mathcal{C}^2$, and $\exists i : c_i(\hat{x}) > 0$. Show that $\hat{x}$ violates
$$0 \geq \hat{c}_i + \nabla \hat{c}^T (x - \hat{x}).$$
2. Consider the worst-ever nonlinear function,
$$z = \frac{1}{1 + 1000(x - y)^{10}},$$
which "models" that $z = 1$, if $x = y$, and $z = 0$, if $x \neq y$. Assuming that $0 \leq x, y \leq U$ are integers, derive an equivalent linear model.

... all answers will be revealed on Friday!


Beale, E. and Tomlin, J. (1970).
Special facilities in a general mathematical programming system for non- convex problems using ordered sets of variables.
In Lawrence, J., editor, *Proceedings of the 5th International Conference on Operations Research*, pages 447–454, Venice, Italy.


Beale, E. M. L. and Forrest, J. J. H. (1976).
Global optimization using special ordered sets.
Mathematical Programming, 10:52–69.


Grossmann, I. and Lee, S. (2003).
Generalized convex disjunctive programming: Nonlinear convex hull relaxation.
Computational Optimization and Applications, pages 83–100.


Günlük, O. and Linderoth, J. T. (2012).
Perspective reformulation and applications.
In *IMA Volumes*, volume 154, pages 61–92.


Jeroslow, R. G. (1973).
There cannot be any algorithm for integer programming with quadratic constraints.
Operations Research, 21(1):221–224.


Kannan, R. and Monma, C. (1978).
On the computational complexity of integer programming problems.
In Henn, R., Korte, B., and Oettli, W., editors, *Optimization and Operations Research,*, volume 157 of *Lecture Notes in Economics and Mathematical Systems*, pages 161–172. Springer.


Williams, H. P. (1999).
Model Building in Mathematical Programming.
John Wiley & Sons.