Model-Method Iteration: Solving Good Problems and Detecting Bad Ones

Todd Munson

Mathematics and Computer Science Division
Argonne National Laboratory

August 4, 2014

Mixed Complementarity Problems

Focus on complementarity problems with equalities

$$\begin{array}{rcl} 0 \leq x & \perp & F(x,y) \geq 0 \\ y & & G(x,y) = 0 \end{array}$$

where $F : \Re^{n+m} \to n$ and $G : \Re^{n+m} \to m$ are continuously differentiable.

Mixed Complementarity Problems

Focus on complementarity problems with equalities

$$\begin{array}{rcl} 0 \le x & \perp & F(x,y) \ge 0 \\ y & & G(x,y) = 0 \end{array}$$

where $F : \Re^{n+m} \to n$ and $G : \Re^{n+m} \to m$ are continuously differentiable.

Good algorithms couple a Newton method to a globalization strategy.

PATH solves mixed linear complementarity problems

$$\begin{array}{rcl} 0 \le x & \perp & Mx + Ny + q \ge 0 \\ y & & Ax + By + c = 0 \end{array}$$

with a line search on a suitable merit function.

Mixed Complementarity Problems

Focus on complementarity problems with equalities

$$\begin{array}{rcl} 0 \leq x & \perp & F(x,y) \geq 0 \\ y & & G(x,y) = 0 \end{array}$$

where $F : \Re^{n+m} \to n$ and $G : \Re^{n+m} \to m$ are continuously differentiable.

Good algorithms couple a Newton method to a globalization strategy.

PATH solves mixed linear complementarity problems

$$\begin{array}{rcl} 0 \leq x & \perp & Mx + Ny + q \geq 0 \\ y & & Ax + By + c = 0 \end{array}$$

with a line search on a suitable merit function.

- In great models, B^{-1} exists and $M - NB^{-1}A$ is positive semidefinite.

Mixed Complementarity Problems

Focus on complementarity problems with equalities

$$\begin{array}{rcl} 0 \leq x & \perp & F(x,y) \geq 0 \\ y & & G(x,y) = 0 \end{array}$$

where $F : \Re^{n+m} \to n$ and $G : \Re^{n+m} \to m$ are continuously differentiable.

Good algorithms couple a Newton method to a globalization strategy.

PATH solves mixed linear complementarity problems

$$\begin{array}{rcl} 0 \leq x & \perp & Mx + Ny + q \geq 0 \\ y & & Ax + By + c = 0 \end{array}$$

with a line search on a suitable merit function.

- In great models, B^{-1} exists and $M - NB^{-1}A$ is positive semidefinite.
- In good models, $(B + \delta I)^{-1}$ exists and

$$M + \delta I - N(B + \delta I)^{-1}A$$

is positive definite for any $\delta > 0$.

Outline

1. Modeling Complementarity Problems
2. Analyzing Complementarity Problems
3. Solving Complementarity Problems

Part I

Modeling Complementarity Problems

Motivation

- What constitutes a good model?
- How are they constructed?
- Why are they good?

Relevant for most of the available algorithms

Section 1

Linear Problems

Bimatrix Games

Problem Specification

- Players select strategies to minimize expected loss
  - $p \in \Re^n$ is the probability player 1 chooses each strategy
  - $q \in \Re^m$ is the probability player 2 chooses each strategy
  - $A \in \Re^{n \times m}$ is the loss matrix for player 1
  - $B \in \Re^{m \times n}$ is the loss matrix for player 2

Bimatrix Games

Problem Specification

- Players select strategies to minimize expected loss
  - $p \in \Re^n$ is the probability player 1 chooses each strategy
  - $q \in \Re^m$ is the probability player 2 chooses each strategy
  - $A \in \Re^{n \times m}$ is the loss matrix for player 1
  - $B \in \Re^{m \times n}$ is the loss matrix for player 2
- Optimization problem for player 1 given q^*

$$\begin{array}{ll} \min\limits_{0 \leq p} & p^T A q^* \\ \text{subject to} & e^T p = 1 \end{array}$$

Bimatrix Games

Problem Specification

- Players select strategies to minimize expected loss
  - $p \in \Re^n$ is the probability player 1 chooses each strategy
  - $q \in \Re^m$ is the probability player 2 chooses each strategy
  - $A \in \Re^{n \times m}$ is the loss matrix for player 1
  - $B \in \Re^{m \times n}$ is the loss matrix for player 2
- Optimization problem for player 1 given q^*

$$\begin{array}{ll} \min\limits_{0 \leq p} & p^T A q^* \\ \text{subject to} & e^T p = 1 \end{array}$$

- Optimization problem for player 2 given p^*

$$\begin{array}{ll} \min\limits_{0 \leq q} & q^T B p^* \\ \text{subject to} & e^T q = 1 \end{array}$$

- Assemble first-order optimality conditions

$$\begin{array}{rcl} 0 \le p & \perp & Aq - \lambda_1 \ge 0 \\ 0 \le q & \perp & Bp - \lambda_2 \ge 0 \\ \lambda_1 & & e^T p = 1 \\ \lambda_2 & & e^T q = 1 \end{array}$$

Bimatrix Games

Mixed Complementarity Formulations

- Assemble first-order optimality conditions

$$\begin{array}{rcl} 0 \le p & \perp & Aq - \lambda_1 \ge 0 \\ 0 \le q & \perp & Bp - \lambda_2 \ge 0 \\ \lambda_1 & & e^T p = 1 \\ \lambda_2 & & e^T q = 1 \end{array}$$

- Equivalent formulation (negative multipliers)

$$\begin{array}{rcl} 0 \le p & \perp & Aq + \lambda_1 \ge 0 \\ 0 \le q & \perp & Bp + \lambda_2 \ge 0 \\ \lambda_1 & & e^T p = 1 \\ \lambda_2 & & e^T q = 1 \end{array}$$

Bimatrix Games

Equation Sign Important

For hard models, solvers will add a diagonal perturbation to regularize the model. They generally employ a single positive perturbation independent of the structure of *your* particular problem.

- First formulation can be a good model

$$
\begin{array}{rcl}
0 \leq x & \perp & (M + \delta I)x - A^T \lambda + q \geq 0 \\
\lambda & & Ax + \delta \lambda + b = 0
\end{array}
$$

  has the reduced formulation

$$
0 \leq x \quad \perp \quad \left(M + \delta I + \tfrac{1}{\delta} A^T A\right) x + q + \tfrac{1}{\delta} A^T b \geq 0
$$

  that inherits the properties of M.

Bimatrix Games

Equation Sign Important

For hard models, solvers will add a diagonal perturbation to regularize the model. They generally employ a single positive perturbation independent of the structure of *your* particular problem.

- Second formulation is not a good model

$$\begin{array}{rcl} 0 \leq x & \perp & (M + \delta I)x + A^T \lambda + q \geq 0 \\ \lambda & & Ax + \delta\lambda + b = 0 \end{array}$$

has reduced formulation

$$0 \leq x \quad \perp \quad \left(M + \delta I - \tfrac{1}{\delta} A^T A\right) x + q - \tfrac{1}{\delta} A^T b \geq 0$$

that does not inherit properties of M.

Bimatrix Games

A Family of Equivalent Problems

- Players select strategies to minimize expected loss
  - $p \in \Re^n$ is the probability player 1 chooses each strategy
  - $q \in \Re^m$ is the probability player 2 chooses each strategy
  - $A \in \Re^{n \times m}$ is the loss matrix for player 1
  - $B \in \Re^{m \times n}$ is the loss matrix for player 2

Bimatrix Games

A Family of Equivalent Problems

- Players select strategies to minimize expected loss
  - $p \in \Re^n$ is the probability player 1 chooses each strategy
  - $q \in \Re^m$ is the probability player 2 chooses each strategy
  - $A \in \Re^{n \times m}$ is the loss matrix for player 1
  - $B \in \Re^{m \times n}$ is the loss matrix for player 2
- Optimization problem for player 1 given q^*

$$\begin{array}{ll} \min\limits_{0 \le p} & p^T(A + \alpha_2 E)q^* \\ \text{subject to} & e^T p = 1 \end{array}$$

Bimatrix Games

A Family of Equivalent Problems

- Players select strategies to minimize expected loss
  - $p \in \Re^n$ is the probability player 1 chooses each strategy
  - $q \in \Re^m$ is the probability player 2 chooses each strategy
  - $A \in \Re^{n \times m}$ is the loss matrix for player 1
  - $B \in \Re^{m \times n}$ is the loss matrix for player 2
- Optimization problem for player 1 given q^*

$$\begin{array}{ll} \min\limits_{0 \le p} & p^T(A + \alpha_2 E)q^* \\ \text{subject to} & e^T p = 1 \end{array}$$

- Optimization problem for player 2 given p^*

$$\begin{array}{ll} \min\limits_{0 \le q} & q^T(B + \alpha_1 E)p^* \\ \text{subject to} & e^T q = 1 \end{array}$$

Bimatrix Games

Knowledge is Not Always Power

If $A + \alpha_2 E < 0$ and $B + \alpha_1 E < 0$, then

$$\begin{array}{rcl} 0 \le p & \perp & (A + \alpha_2 E) q + \lambda_1 \ge 0 \\ 0 \le q & \perp & (B + \alpha_1 E) p + \lambda_2 \ge 0 \\ \lambda_1 & & e^T p = 1 \\ \lambda_2 & & e^T q = 1 \end{array}$$

is equivalent to

$$\begin{array}{rcl} 0 \le p & \perp & (A + \alpha_2 E) q + \lambda_1 \ge 0 \\ 0 \le q & \perp & (B + \alpha_1 E) p + \lambda_2 \ge 0 \\ 0 \le \lambda_1 & \perp & e^T p - 1 \ge 0 \\ 0 \le \lambda_2 & \perp & e^T q - 1 \ge 0 \end{array}$$

Bimatrix Games

Knowledge is Not Always Power

If $A + \alpha_2 E < 0$ and $B + \alpha_1 E < 0$, then

$$\begin{array}{rcl} 0 \leq p & \perp & (A + \alpha_2 E)q + \lambda_1 \geq 0 \\ 0 \leq q & \perp & (B + \alpha_1 E)p + \lambda_2 \geq 0 \\ \lambda_1 & & e^T p = 1 \\ \lambda_2 & & e^T q = 1 \end{array}$$

is equivalent to

$$\begin{array}{rcl} 0 \leq p & \perp & (A + \alpha_2 E)q + \lambda_1 \geq 0 \\ 0 \leq q & \perp & (B + \alpha_1 E)p + \lambda_2 \geq 0 \\ 0 \leq \lambda_1 & \perp & e^T p - 1 \geq 0 \\ 0 \leq \lambda_2 & \perp & e^T q - 1 \geq 0 \end{array}$$

which can be further reduced to finding nonzero p and q with

$$\begin{array}{rcl} 0 \leq p & \perp & (A + \alpha_2 E)q + e \geq 0 \\ 0 \leq q & \perp & (B + \alpha_1 E)p + e \geq 0 \end{array}$$

Bimatrix Games

Knowledge is Not Always Power

If $A + \alpha_2 E < 0$ and $B + \alpha_1 E < 0$, then

$$\begin{array}{rcl} 0 \leq p & \perp & (A + \alpha_2 E)q + \lambda_1 \geq 0 \\ 0 \leq q & \perp & (B + \alpha_1 E)p + \lambda_2 \geq 0 \\ \lambda_1 & & e^T p = 1 \\ \lambda_2 & & e^T q = 1 \end{array}$$

is equivalent to

$$\begin{array}{rcl} 0 \leq p & \perp & (A + \alpha_2 E)q + \lambda_1 \geq 0 \\ 0 \leq q & \perp & (B + \alpha_1 E)p + \lambda_2 \geq 0 \\ 0 \leq \lambda_1 & \perp & e^T p - 1 \geq 0 \\ 0 \leq \lambda_2 & \perp & e^T q - 1 \geq 0 \end{array}$$

which can be further reduced to finding nonzero p and q with

$$\begin{array}{rcl} 0 \leq p & \perp & (A + \alpha_2 E)q + e \geq 0 \\ 0 \leq q & \perp & (B + \alpha_1 E)p + e \geq 0 \end{array}$$

and the matrix class is not pleasant.

Bimatrix Games

Knowledge is Sometimes Power

If $A + \alpha_2 E > 0$ and $B + \alpha_1 E > 0$, then

$$\begin{array}{rcl} 0 \leq p & \perp & (A + \alpha_2 E)q - \lambda_1 \geq 0 \\ 0 \leq q & \perp & (B + \alpha_1 E)p - \lambda_2 \geq 0 \\ \lambda_1 & & e^T p = 1 \\ \lambda_2 & & e^T q = 1 \end{array}$$

is equivalent to

$$\begin{array}{rcl} 0 \leq p & \perp & (A + \alpha_2 E)q - \lambda_1 \geq 0 \\ 0 \leq q & \perp & (B + \alpha_1 E)p - \lambda_2 \geq 0 \\ 0 \leq \lambda_1 & \perp & e^T p - 1 \geq 0 \\ 0 \leq \lambda_2 & \perp & e^T q - 1 \geq 0 \end{array}$$

Bimatrix Games

Knowledge is Sometimes Power

If $A + \alpha_2 E > 0$ and $B + \alpha_1 E > 0$, then

$$\begin{array}{rcl} 0 \le p & \perp & (A + \alpha_2 E)q - \lambda_1 \ge 0 \\ 0 \le q & \perp & (B + \alpha_1 E)p - \lambda_2 \ge 0 \\ \lambda_1 & & e^T p = 1 \\ \lambda_2 & & e^T q = 1 \end{array}$$

is equivalent to

$$\begin{array}{rcl} 0 \le p & \perp & (A + \alpha_2 E)q - \lambda_1 \ge 0 \\ 0 \le q & \perp & (B + \alpha_1 E)p - \lambda_2 \ge 0 \\ 0 \le \lambda_1 & \perp & e^T p - 1 \ge 0 \\ 0 \le \lambda_2 & \perp & e^T q - 1 \ge 0 \end{array}$$

which can be further reduced to

$$\begin{array}{rcl} 0 \le p & \perp & (A + \alpha_2 E)q - e \ge 0 \\ 0 \le q & \perp & (B + \alpha_1 E)p - e \ge 0 \end{array}$$

Bimatrix Games

Knowledge is Sometimes Power

If $A + \alpha_2 E > 0$ and $B + \alpha_1 E > 0$, then

$$\begin{array}{rcl} 0 \leq p & \perp & (A + \alpha_2 E)q - \lambda_1 \geq 0 \\ 0 \leq q & \perp & (B + \alpha_1 E)p - \lambda_2 \geq 0 \\ \lambda_1 & & e^T p = 1 \\ \lambda_2 & & e^T q = 1 \end{array}$$

is equivalent to

$$\begin{array}{rcl} 0 \leq p & \perp & (A + \alpha_2 E)q - \lambda_1 \geq 0 \\ 0 \leq q & \perp & (B + \alpha_1 E)p - \lambda_2 \geq 0 \\ 0 \leq \lambda_1 & \perp & e^T p - 1 \geq 0 \\ 0 \leq \lambda_2 & \perp & e^T q - 1 \geq 0 \end{array}$$

which can be further reduced to

$$\begin{array}{rcl} 0 \leq p & \perp & (A + \alpha_2 E)q - e \geq 0 \\ 0 \leq q & \perp & (B + \alpha_1 E)p - e \geq 0 \end{array}$$

and a special Lemke method can be applied.

Bimatrix Games

Strictly Positive Matrix Formulation

If $A + \alpha_2 E > 0$ and $B + \alpha_1 E > 0$, then

$$\begin{array}{rcl} 0 \leq p & \perp & \alpha_1 Ep + (A + \alpha_2 E)q - \lambda_1 \geq 0 \\ 0 \leq q & \perp & (B + \alpha_1 E)p + \alpha_2 Eq - \lambda_2 \geq 0 \\ \lambda_1 & & e^T p = 1 \\ \lambda_2 & & e^T q = 1 \end{array}$$

is equivalent to

$$\begin{array}{rcl} 0 \leq p & \perp & \alpha_1 Ep + (A + \alpha_2 E)q - \lambda_1 \geq 0 \\ 0 \leq q & \perp & (B + \alpha_1 E)p + \alpha_2 Eq - \lambda_2 \geq 0 \\ 0 \leq \lambda_1 & \perp & e^T p - 1 \geq 0 \\ 0 \leq \lambda_2 & \perp & e^T q - 1 \geq 0 \end{array}$$

Bimatrix Games

Strictly Positive Matrix Formulation

If $A + \alpha_2 E > 0$ and $B + \alpha_1 E > 0$, then

$$\begin{array}{rcl} 0 \le p & \perp & \alpha_1 Ep + (A + \alpha_2 E)q - \lambda_1 \ge 0 \\ 0 \le q & \perp & (B + \alpha_1 E)p + \alpha_2 Eq - \lambda_2 \ge 0 \\ \lambda_1 & & e^T p = 1 \\ \lambda_2 & & e^T q = 1 \end{array}$$

is equivalent to

$$\begin{array}{rcl} 0 \le p & \perp & \alpha_1 Ep + (A + \alpha_2 E)q - \lambda_1 \ge 0 \\ 0 \le q & \perp & (B + \alpha_1 E)p + \alpha_2 Eq - \lambda_2 \ge 0 \\ 0 \le \lambda_1 & \perp & e^T p - 1 \ge 0 \\ 0 \le \lambda_2 & \perp & e^T q - 1 \ge 0 \end{array}$$

which can be further reduced to

$$\begin{array}{rcl} 0 \le p & \perp & \alpha_1 Ep + (A + \alpha_2 E)q - e \ge 0 \\ 0 \le q & \perp & (B + \alpha_1 E)p + \alpha_2 Eq - e \ge 0 \end{array}$$

Bimatrix Games

Strictly Positive Matrix Formulation

If $A + \alpha_2 E > 0$ and $B + \alpha_1 E > 0$, then

$$\begin{array}{rcl} 0 \le p & \perp & \alpha_1 Ep + (A + \alpha_2 E)q - \lambda_1 \ge 0 \\ 0 \le q & \perp & (B + \alpha_1 E)p + \alpha_2 Eq - \lambda_2 \ge 0 \\ \lambda_1 & & e^T p = 1 \\ \lambda_2 & & e^T q = 1 \end{array}$$

is equivalent to

$$\begin{array}{rcl} 0 \le p & \perp & \alpha_1 Ep + (A + \alpha_2 E)q - \lambda_1 \ge 0 \\ 0 \le q & \perp & (B + \alpha_1 E)p + \alpha_2 Eq - \lambda_2 \ge 0 \\ 0 \le \lambda_1 & \perp & e^T p - 1 \ge 0 \\ 0 \le \lambda_2 & \perp & e^T q - 1 \ge 0 \end{array}$$

which can be further reduced to

$$\begin{array}{rcl} 0 \le p & \perp & \alpha_1 Ep + (A + \alpha_2 E)q - e \ge 0 \\ 0 \le q & \perp & (B + \alpha_1 E)p + \alpha_2 Eq - e \ge 0 \end{array}$$

and a Lemke's method (with lexicographic pivoting) can be applied.

Bimatrix Games

Symmetric Diagonal Scaling

Symmetric can be applied to

- Improve condition number of matrices
- Reduce degeneracy with Lemke's method

without changing matrix properties or affecting rank.

Symmetric can be applied to

- Improve condition number of matrices
- Reduce degeneracy with Lemke's method

without changing matrix properties or affecting rank.

For our problem we have

$$\begin{array}{lcl} 0 \le p & \perp & \alpha_1 S_1 E S_1 p + S_1(A + \alpha_2 E) S_2 q - S_1 e \ge 0 \\ 0 \le q & \perp & S_2(B + \alpha_1 E) S_1 p + \alpha_2 S_2 E S_2 q - S_2 e \ge 0 \end{array}$$

for positive diagonal matrices S_1 and S_2.

Complementarity Between Functions

Find an x such that

$$0 \leq Mx + q \quad \perp \quad Nx + r \geq 0$$

Find an x such that

$$0 \leq Mx + q \quad \perp \quad Nx + r \geq 0$$

Add one slack variable to obtain the problem

$$\begin{array}{lcl} x & & Mx - s + q = 0 \\ 0 \leq s & \perp & Nx + r \geq 0 \end{array}$$

with the reduced, perturbed problem

$$0 \leq s \quad \perp \quad (N(M + \delta I)^{-1} + \delta I)s + \tilde{r} \geq 0$$

that may be a good model.

Add two slack variables to obtain

$$\begin{array}{rcl} & & Mx - s + q = 0 \\ & & t - Nx - r = 0 \\ 0 \leq s & \perp & t \geq 0 \end{array}$$

where solver matches variables to equations.

- Good match

$$\begin{array}{rcl} x & & Mx - s + q = 0 \\ t & & t - Nx - r = 0 \\ 0 \leq s & \perp & t \geq 0 \end{array}$$

  may result in a good reduced, perturbed model

$$0 \leq s \quad \perp \quad (N(M + \delta I)^{-1} + \delta(1 + \delta)I)s + \tilde{r} \geq 0$$

Complementarity Between Functions

Alternative Formulations

Add two slack variables to obtain

$$\begin{array}{rcl} & & Mx - s + q = 0 \\ & & t - Nx - r = 0 \\ 0 \leq s & \perp & t \geq 0 \end{array}$$

where solver matches variables to equations.

- Unfortunate match

  $$\begin{array}{rcl} t & & Mx - s + q = 0 \\ x & & t - Nx - r = 0 \\ 0 \leq s & \perp & t \geq 0 \end{array}$$

  may result in a "bad" reduced, perturbed model

  $$0 \leq s \quad \perp \quad ((N - \delta I)(M + \delta N - \delta^2 I)^{-1} + \delta I)s + \tilde{r} \geq 0$$

- Do not rely on the solver to perform a good match!

Summary

- Always match equations to appropriate variables
  - Automatically performing a good match is difficult
  - Modeler knows their problem and should convey the match

Summary

- Always match equations to appropriate variables
  - Automatically performing a good match is difficult
  - Modeler knows their problem and should convey the match
- Always prefer a monotone formulation over other formulations
  - Sign on the equations matters when perturbing
  - Use skew symmetric version of optimality conditions
  - Use problem knowledge if the result is a “better” problem

Summary

- Always match equations to appropriate variables
  - Automatically performing a good match is difficult
  - Modeler knows their problem and should convey the match
- Always prefer a monotone formulation over other formulations
  - Sign on the equations matters when perturbing
  - Use skew symmetric version of optimality conditions
  - Use problem knowledge if the result is a "better" problem
- Use symmetric scaling to
  - Improve condition number of matrices
  - Reduce degeneracy with Lemke's method
  - Beware when using unscaled solution!

Summary

- Always match equations to appropriate variables
  - Automatically performing a good match is difficult
  - Modeler knows their problem and should convey the match
- Always prefer a monotone formulation over other formulations
  - Sign on the equations matters when perturbing
  - Use skew symmetric version of optimality conditions
  - Use problem knowledge if the result is a "better" problem
- Use symmetric scaling to
  - Improve condition number of matrices
  - Reduce degeneracy with Lemke's method
  - Beware when using unscaled solution!
- Rank deficiency can still be an issue
  - Solver can sometimes identify and eliminate dependencies
  - Solver may add a positive diagonal perturbation
    - Improve condition number of matrices
    - Eliminate dependencies for good models

Summary

- Always match equations to appropriate variables
  - Automatically performing a good match is difficult
  - Modeler knows their problem and should convey the match
- Always prefer a monotone formulation over other formulations
  - Sign on the equations matters when perturbing
  - Use skew symmetric version of optimality conditions
  - Use problem knowledge if the result is a “better” problem
- Use symmetric scaling to
  - Improve condition number of matrices
  - Reduce degeneracy with Lemke’s method
  - Beware when using unscaled solution!
- Rank deficiency can still be an issue
  - Solver can sometimes identify and eliminate dependencies
  - Solver may add a positive diagonal perturbation
    - Improve condition number of matrices
    - Eliminate dependencies for good models
- Always check the solution reported!

Section 2

Nonlinear Problems

Nash Games

Problem Formulation

- Non-cooperative game played by n individuals
  - Each player selects a strategy to optimize their objective
  - Strategies for the other players are fixed
- Equilibrium reached when no improvement is possible

Nash Games

Problem Formulation

- Non-cooperative game played by n individuals
  - Each player selects a strategy to optimize their objective
  - Strategies for the other players are fixed
- Equilibrium reached when no improvement is possible
- Characterization of two player equilibrium (x^*, y^*)

$$
\begin{aligned}
x^* &\in \left\{ \begin{array}{ll} \arg\min\limits_{x \geq 0} & f_1(x, y^*) \\ \text{subject to} & c_1(x) \leq 0 \end{array} \right. \\
y^* &\in \left\{ \begin{array}{ll} \arg\min\limits_{y \geq 0} & f_2(x^*, y) \\ \text{subject to} & c_2(y) \leq 0 \end{array} \right.
\end{aligned}
$$

Nash Games

Problem Formulation

- Non-cooperative game played by n individuals
  - Each player selects a strategy to optimize their objective
  - Strategies for the other players are fixed
- Equilibrium reached when no improvement is possible
- Characterization of two player equilibrium (x^*, y^*)

$$
x^* \in \left\{ \begin{array}{ll} \operatorname*{arg\,min}_{x \geq 0} & f_1(x, y^*) \\ \text{subject to} & c_1(x) \leq 0 \end{array} \right.
$$

$$
y^* \in \left\{ \begin{array}{ll} \operatorname*{arg\,min}_{y \geq 0} & f_2(x^*, y) \\ \text{subject to} & c_2(y) \leq 0 \end{array} \right.
$$

- Many applications
  - Walrasian (traffic) equilibrium
  - Arrow-Debreau (economic) models
  - Cournot duopoly models

Nash Games

Complementarity Formulation

- Assume each optimization problem is convex
  - $f_1(\cdot, y)$ is convex for each y
  - $f_2(x, \cdot)$ is convex for each x
  - $c_1(\cdot)$ and $c_2(\cdot)$ convex and satisfy constraint qualification
- Then the first-order conditions are necessary and sufficient

$$\begin{array}{ll} \min\limits_{x \geq 0} & f_1(x, y^*) \\ \text{subject to} & c_1(x) \leq 0 \end{array} \quad \Leftrightarrow \quad \begin{array}{lcl} 0 \leq x & \perp & \nabla_x f_1(x, y^*) + \lambda_1^T \nabla_x c_1(x) \geq 0 \\ 0 \leq \lambda_1 & \perp & -c_1(x) \geq 0 \end{array}$$

Nash Games

Complementarity Formulation

- Assume each optimization problem is convex
  - $f_1(\cdot, y)$ is convex for each y
  - $f_2(x, \cdot)$ is convex for each x
  - $c_1(\cdot)$ and $c_2(\cdot)$ convex and satisfy constraint qualification
- Then the first-order conditions are necessary and sufficient

$$\begin{array}{ll} \min\limits_{y \geq 0} & f_2(x^*, y) \\ \text{subject to} & c_2(y) \leq 0 \end{array} \quad \Leftrightarrow \quad \begin{array}{lcl} 0 \leq y & \perp & \nabla_y f_2(x^*, y) + \lambda_2^T \nabla_y c_2(y) \geq 0 \\ 0 \leq \lambda_2 & \perp & -c_2(y) \geq 0 \end{array}$$

Nash Games

Complementarity Formulation

- Assume each optimization problem is convex
  - $f_1(\cdot, y)$ is convex for each y
  - $f_2(x, \cdot)$ is convex for each x
  - $c_1(\cdot)$ and $c_2(\cdot)$ convex and satisfy constraint qualification
- Then the first-order conditions are necessary and sufficient

$$
\begin{array}{lcl}
0 \leq x & \perp & \nabla_x f_1(x, y) + \lambda_1^T \nabla_x c_1(x) \geq 0 \\
0 \leq y & \perp & \nabla_y f_2(x, y) + \lambda_2^T \nabla_y c_2(y) \geq 0 \\
0 \leq \lambda_1 & \perp & -c_1(x) \geq 0 \\
0 \leq \lambda_2 & \perp & -c_2(y) \geq 0
\end{array}
$$

- Nonlinear complementarity problem
  - Each solution is an equilibrium for the Nash game
  - Formulation is not correct for nonconvex problems
  - Writing the first-order conditions is error prone
  - Recommend using the MOPEC machinery

Nash Games

Constant Elasticity of Substitution

Ensure functions and Jacobians are defined

- Confront and reformulate at the modeling stage
- Report remaining domain violations to solver
  - Function undefined will cause backtrack in line search
  - Function defined and Jacobian undefined will abort solve
- Provide a starting point where functions are defined

Nash Games

Constant Elasticity of Substitution

Ensure functions and Jacobians are defined

- Confront and reformulate at the modeling stage
- Report remaining domain violations to solver
  - Function undefined will cause backtrack in line search
  - Function defined and Jacobian undefined will abort solve
- Provide a starting point where functions are defined

Common in economics for the objective function to use

$$(\alpha_1 x_1^\gamma + \alpha_2 x_2^\gamma)^{\frac{1}{\gamma}}$$

where $x_1 \geq 0$ and $x_2 \geq 0$

Nash Games

Constant Elasticity of Substitution

Ensure functions and Jacobians are defined

- Confront and reformulate at the modeling stage
- Report remaining domain violations to solver
  - Function undefined will cause backtrack in line search
  - Function defined and Jacobian undefined will abort solve
- Provide a starting point where functions are defined

Common in economics for the objective function to use

$$(\alpha_1 x_1^\gamma + \alpha_2 x_2^\gamma)^{\frac{1}{\gamma}}$$

where $x_1 \geq 0$ and $x_2 \geq 0$

- Function not defined when $x_1 = 0$ or $x_2 = 0$
  $$(\alpha_1(x_1 + \epsilon)^\gamma + \alpha_2(x_2 + \epsilon)^\gamma - (\alpha_1 + \alpha_2)\epsilon^\gamma)^{\frac{1}{\gamma}}$$

Nash Games

Constant Elasticity of Substitution

Ensure functions and Jacobians are defined

- Confront and reformulate at the modeling stage
- Report remaining domain violations to solver
  - Function undefined will cause backtrack in line search
  - Function defined and Jacobian undefined will abort solve
- Provide a starting point where functions are defined

Common in economics for the objective function to use

$$(\alpha_1 x_1^\gamma + \alpha_2 x_2^\gamma)^{\frac{1}{\gamma}}$$

where $x_1 \geq 0$ and $x_2 \geq 0$

- Function not defined when $x_1 = 0$ or $x_2 = 0$
$$(\alpha_1(x_1 + \epsilon)^\gamma + \alpha_2(x_2 + \epsilon)^\gamma - (\alpha_1 + \alpha_2)\epsilon^\gamma)^{\frac{1}{\gamma}}$$
- Jacobian not defined when $x_1 = x_2 = 0$
$$(\alpha_1(x_1 + \epsilon)^\gamma + \alpha_2(x_2 + \epsilon)^\gamma)^{\frac{1}{\gamma}} - (\alpha_1 + \alpha_2)^{\frac{1}{\gamma}}\epsilon$$

Nash Games

Constant Elasticity of Substitution

Ensure functions and Jacobians are defined

- Confront and reformulate at the modeling stage
- Report remaining domain violations to solver
  - Function undefined will cause backtrack in line search
  - Function defined and Jacobian undefined will abort solve
- Provide a starting point where functions are defined

Common in economics for the objective function to use

$$(\alpha_1 x_1^\gamma + \alpha_2 x_2^\gamma)^{\frac{1}{\gamma}}$$

where $x_1 \geq 0$ and $x_2 \geq 0$

- Function not defined when $x_1 = 0$ or $x_2 = 0$
  $$(\alpha_1 (x_1 + \epsilon)^\gamma + \alpha_2 (x_2 + \epsilon)^\gamma - (\alpha_1 + \alpha_2)\epsilon^\gamma)^{\frac{1}{\gamma}}$$
- Jacobian not defined when $x_1 = x_2 = 0$
  $$(\alpha_1 (x_1 + \epsilon)^\gamma + \alpha_2 (x_2 + \epsilon)^\gamma)^{\frac{1}{\gamma}} - (\alpha_1 + \alpha_2)^{\frac{1}{\gamma}}\epsilon$$
- Smoothing the functions changes the answer
  - Modeler needs to determine permissible perturbation
  - Modeler may want to determine sensitivity to perturbation

Permissible to add slack variables in the formulation

$$
\begin{array}{lcl}
0 \le x & \perp & \nabla_x f_1(x,y) + \lambda_1^T \nabla_x c_1(x) \ge 0 \\
0 \le y & \perp & \nabla_y f_2(x,y) + \lambda_2^T \nabla_y c_2(y) \ge 0 \\
0 \le s_1 & \perp & \lambda_1 \ge 0 \\
0 \le s_2 & \perp & \lambda_2 \ge 0 \\
\lambda_1 & & -c_1(x) - s_1 = 0 \\
\lambda_2 & & -c_2(y) - s_2 = 0
\end{array}
$$

provided skew symmetric form of optimality conditions used.

Nash Games

Slack Variables and Equations

Changing the sign of the equations

$$
\begin{array}{lcl}
0 \le x & \perp & \nabla_x f_1(x,y) + \lambda_1^T \nabla_x c_1(x) \ge 0 \\
0 \le y & \perp & \nabla_y f_2(x,y) + \lambda_2^T \nabla_y c_2(y) \ge 0 \\
0 \le s_1 & \perp & \lambda_1 \ge 0 \\
0 \le s_2 & \perp & \lambda_2 \ge 0 \\
\lambda_1 & & c_1(x) + s_1 = 0 \\
\lambda_2 & & c_2(y) + s_2 = 0
\end{array}
$$

causes problems when positive diagonal perturbation added.

Nash Games

Slack Variables and Equations

For general mixed nonlinear complementarity problems, use

$$\begin{array}{lcl} 0 \leq x & \perp & F(x) - A^T\lambda \geq 0 \\ \lambda & & Ax = b \end{array}$$

rather than

$$\begin{array}{lcl} 0 \leq x & \perp & F(x) + A^T\lambda \geq 0 \\ \lambda & & Ax = b \end{array}$$

and match the equations to the variables!

Nash Games

Scaling the Problem

- Any scaling can be applied to original optimization problems
  - Ensure consistency in scaled variables across problems
  - Form optimality conditions of the scaled problems

Nash Games

Scaling the Problem

- Any scaling can be applied to original optimization problems
  - Ensure consistency in scaled variables across problems
  - Form optimality conditions of the scaled problems
- Apply only symmetric scaling to the complementarity problem

$$
\begin{array}{rcl}
0 \leq x & \perp & SF(Sx, Ry) \geq 0 \\
y & & RG(Sx, Ry) = 0
\end{array}
$$

Nash Games

Scaling the Problem

- Any scaling can be applied to original optimization problems
  - Ensure consistency in scaled variables across problems
  - Form optimality conditions of the scaled problems
- Apply only symmetric scaling to the complementarity problem

$$
\begin{array}{rcl}
0 \leq x & \perp & SF(Sx, Ry) \geq 0 \\
y & & RG(Sx, Ry) = 0
\end{array}
$$

- For linear constraints, R is the multiplier scaling

$$
\begin{array}{lcl}
0 \leq x & \perp & SF(Sx) - SA^T R\lambda \geq 0 \\
\lambda & & RASx = Rb
\end{array}
$$

Nash Games

Scaling the Problem

- Any scaling can be applied to original optimization problems
  - Ensure consistency in scaled variables across problems
  - Form optimality conditions of the scaled problems
- Apply only symmetric scaling to the complementarity problem

$$\begin{array}{rcl} 0 \leq x & \perp & SF(Sx, Ry) \geq 0 \\ y & & RG(Sx, Ry) = 0 \end{array}$$

- For linear constraints, R is the multiplier scaling

$$\begin{array}{lcl} 0 \leq x & \perp & SF(Sx) - SA^T R\lambda \geq 0 \\ \lambda & & RASx = Rb \end{array}$$

Beware when using unscaled solution!

Summary

- Always ensure functions and Jacobians are defined
  - Confront and reformulate at the modeling stage
  - Report remaining domain violations to solver
    - Function undefined will cause backtrack in line search
    - Function defined and Jacobian undefined will abort solve

Summary

- Always ensure functions and Jacobians are defined
  - Confront and reformulate at the modeling stage
  - Report remaining domain violations to solver
    - Function undefined will cause backtrack in line search
    - Function defined and Jacobian undefined will abort solve
- Recommend using the MOPEC machinery when possible
  - Conveys more information that the solver can use
  - Relieves modeler of writing code for the derivatives
  - Formulates optimality conditions correctly for the solver

Summary

- Always ensure functions and Jacobians are defined
  - Confront and reformulate at the modeling stage
  - Report remaining domain violations to solver
    - Function undefined will cause backtrack in line search
    - Function defined and Jacobian undefined will abort solve
- Recommend using the MOPEC machinery when possible
  - Conveys more information that the solver can use
  - Relieves modeler of writing code for the derivatives
  - Formulates optimality conditions correctly for the solver
- Always match equations to appropriate variables
- Always prefer a monotone formulation over other formulations
- Use scaling when appropriate
  - Any consistent scaling for optimization problems
  - Symmetric diagonal scaling for complementarity problems
  - Beware when using unscaled solution!
- Rank deficiency can still be an issue
  - Solver can sometimes identify and eliminate dependencies
  - Solver may add a positive diagonal perturbation
- Always check the solution reported!

Part II

Analyzing Complementarity Problems

Motivation

Given a complementarity problem:

- What tools can be used to analyze it?
- What do the tools say about the problem?
- What can be done to address any issues uncovered?

Motivation

Given a complementarity problem:

- What tools can be used to analyze it?
- What do the tools say about the problem?
- What can be done to address any issues uncovered?

Based on PATH solver with specified options

Section 3

Variational Inequalities

Variational Inequalities

Definition

- Let $F : \Re^n \rightarrow \Re^n$ be continuously differentiable
- Let $X \subseteq \Re^n$ be a closed convex set
- Variational inequality is to find $x \in X$ such that

$$\langle F(x), y - x \rangle \geq 0 \quad \forall\, y \in X$$

Variational Inequalities

Definition

- Let $F : \Re^n \to \Re^n$ be continuously differentiable
- Let $X \subseteq \Re^n$ be a closed convex set
- Variational inequality is to find $x \in X$ such that

$$\langle F(x), y - x \rangle \geq 0 \quad \forall\, y \in X$$

- Equivalent formulation is the generalized equation

$$0 \in F(x) + N_X(x)$$

where the normal cone to X at $x \in X$ is

$$N_X(x) := \{\bar{x} \mid \langle \bar{x}, y - x \rangle \leq 0 \; \forall\, y \in X\}$$

Variational Inequalities

Definition

- Let $F : \Re^n \to \Re^n$ be continuously differentiable
- Let $X \subseteq \Re^n$ be a closed convex set
- Variational inequality is to find $x \in X$ such that

$$\langle F(x), y - x \rangle \geq 0 \quad \forall\, y \in X$$

- Equivalent formulation is the generalized equation

$$0 \in F(x) + N_X(x)$$

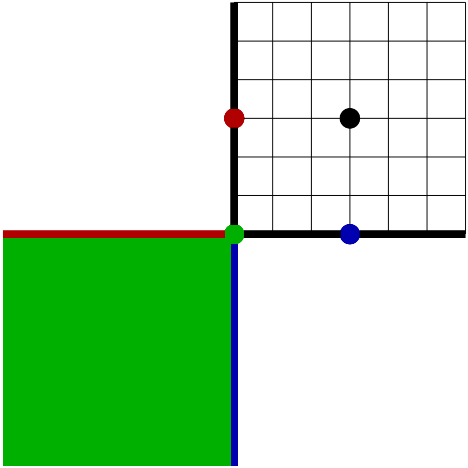
where the normal cone to X at $x \in X$ is

$$N_X(x) := \{\bar{x} \mid \langle \bar{x}, y - x \rangle \leq 0 \; \forall\, y \in X\}$$

- Special cases include
  - Nonlinear equations when $X = \Re^n$
  - Nonlinear complementarity when $X = \Re^n_+$
  - Mixed complementarity when $X = [l, u]^n$

Variational Inequalities

Normal Cone



Variational Inequalities

Simplified Theory (Robinson)

- Let $F : \Re^n \to \Re^n$, $A \in \Re^{m \times n}$ and $b \in \Re^m$.

Variational Inequalities

Simplified Theory (Robinson)

- Let $F : \Re^n \to \Re^n$, $A \in \Re^{m \times n}$ and $b \in \Re^m$.
- If x and λ solve

$$\begin{array}{rcl} 0 \leq x & \perp & F(x) - A^T \lambda \geq 0 \\ \lambda & & Ax + b = 0 \end{array} \tag{1}$$

then

$$0 \in F(x) + N_X(x) \tag{2}$$

where $X = \{x \mid x \geq 0 \text{ and } Ax + b = 0\}$.

Variational Inequalities

Simplified Theory (Robinson)

- Let $F : \Re^n \to \Re^n$, $A \in \Re^{m \times n}$ and $b \in \Re^m$.
- If x and λ solve

$$\begin{array}{rcl} 0 \le x & \perp & F(x) - A^T\lambda \ge 0 \\ \lambda & & Ax + b = 0 \end{array} \qquad (1)$$

  then

$$0 \in F(x) + N_X(x) \qquad (2)$$

  where $X = \{x \mid x \ge 0 \text{ and } Ax + b = 0\}$.
- If x solves (2), then multipliers λ exist such that x and λ solve (1).

Variational Inequalities

Simplified Theory (Robinson)

- Let $F : \Re^n \to \Re^n$, $A \in \Re^{m \times n}$ and $b \in \Re^m$.
- If x and λ solve

$$\begin{array}{rcl} 0 \le x & \perp & F(x) - A^T\lambda \ge 0 \\ \lambda & & Ax + b = 0 \end{array} \qquad (1)$$

  then

$$0 \in F(x) + N_X(x) \qquad (2)$$

  where $X = \{x \mid x \ge 0 \text{ and } Ax + b = 0\}$.
- If x solves (2), then multipliers λ exist such that x and λ solve (1).

Note: X and $N_X(\cdot)$ are geometric objects and we are free to choose among equivalent algebraic representations.

Variational Inequalities

Basic Preprocessing Methodology

- Given an arbitrary complementarity problem

Variational Inequalities

Basic Preprocessing Methodology

- Given an arbitrary complementarity problem
- Discover skew symmetric structure within the problem

Variational Inequalities

Basic Preprocessing Methodology

- Given an arbitrary complementarity problem
- Discover skew symmetric structure within the problem
- Convert the problem into the equivalent variational inequality
- Choose representation of the polyhedral constraint set
  - Reduce model size and complexity
  - Improve algorithm performance
  - Detect unsolvable models

Variational Inequalities

Basic Preprocessing Methodology

- Given an arbitrary complementarity problem
- Discover skew symmetric structure within the problem
- Convert the problem into the equivalent variational inequality
- Choose representation of the polyhedral constraint set
  - Reduce model size and complexity
  - Improve algorithm performance
  - Detect unsolvable models
- Recover reduced complementarity problem

Variational Inequalities

Discovering Generalized Skew Symmetry

- Provided with a list of linear rows and columns for the problem
- For each linear row we perform the following
  - Check that the column is linear
  - Reject rows having a diagonal entry
  - Establish nonzero pattern is identical
  - Ensure elements have the opposite signs

Variational Inequalities

Discovering Generalized Skew Symmetry

- Provided with a list of linear rows and columns for the problem
- For each linear row we perform the following
  - Check that the column is linear
  - Reject rows having a diagonal entry
  - Establish nonzero pattern is identical
  - Ensure elements have the opposite signs
    - Can negate rows corresponding to equalities

$$\begin{array}{rcl} 0 \le x & \perp & x + y + z \ge 0 \\ y & & x + y + \frac{1}{3}z = 4 \\ z & & 2x - y = 5 \end{array}$$

is equivalent to

$$\begin{array}{rcl} 0 \le x & \perp & 2x + 2y + 2z \ge 0 \\ y & & -3x - 3y - z = -12 \\ z & & -2x + y = -5 \end{array}$$

- Provided with a list of linear rows and columns for the problem
- For each linear row we perform the following
  - Check that the column is linear
  - Reject rows having a diagonal entry
  - Establish nonzero pattern is identical
  - Ensure elements have the opposite signs
    - Can negate rows corresponding to equalities

$$\begin{array}{rcl} 0 \le x & \perp & x + y + z \ge 0 \\ y & & x + y + \frac{1}{3} z = 4 \\ z & & 2x - y = 5 \end{array}$$

is equivalent to

$$\begin{array}{rcl} 0 \le x & \perp & 2x + 2y + 2z \ge 0 \\ y & & -3x - 3y - z = -12 \\ z & & -2x + y = -5 \end{array}$$

The `check_skew_symmetry` option will report findings and can be used to check the correctness of your formulation!

Variational Inequalities

Single Constraint Reductions

- Reductions on a single constraint include
  - Singleton rows
  - Doubleton rows with a column singleton
  - Forcing conditions

Variational Inequalities

Single Constraint Reductions

- Reductions on a single constraint include
  - Singleton rows
  - Doubleton rows with a column singleton
  - Forcing conditions
- An example of a singleton row
  1. Complementarity problem

$$
\begin{array}{lcl}
0 \leq x & \perp & x^2 - y + 1 \geq 0 \\
0 \leq y & \perp & x - 1 \geq 0
\end{array}
$$

Variational Inequalities

Single Constraint Reductions

- Reductions on a single constraint include
  - Singleton rows
  - Doubleton rows with a column singleton
  - Forcing conditions
- An example of a singleton row
  1. Complementarity problem

$$\begin{array}{lcl} 0 \le x & \perp & x^2 - y + 1 \ge 0 \\ 0 \le y & \perp & x - 1 \ge 0 \end{array}$$

  2. Form equivalent polyhedral problem

$$0 \in x^2 + 1 + N_X(x)$$

where $X = \{x \mid x \ge 0 \text{ and } x \ge 1\}$

Variational Inequalities

Single Constraint Reductions

- Reductions on a single constraint include
  - Singleton rows
  - Doubleton rows with a column singleton
  - Forcing conditions
- An example of a singleton row
  1. Complementarity problem

$$\begin{array}{lcl} 0 \leq x & \perp & x^2 - y + 1 \geq 0 \\ 0 \leq y & \perp & x - 1 \geq 0 \end{array}$$

  2. Form equivalent polyhedral problem

$$0 \in x^2 + 1 + N_X(x)$$

     where $X = \{x \mid x \geq 0 \text{ and } x \geq 1\}$
  3. Recover reduced complementarity problem

$$1 \leq x \quad \perp \quad x^2 + 1 \geq 0$$

     with the solution $x = 1$

Variational Inequalities

Single Constraint Reductions

- Reductions on a single constraint include
  - Singleton rows
  - Doubleton rows with a column singleton
  - Forcing conditions
- An example of a singleton row
  1. Complementarity problem

$$\begin{array}{lcl} 0 \leq x & \perp & x^2 - y + 1 \geq 0 \\ 0 \leq y & \perp & x - 1 \geq 0 \end{array}$$

  2. Form equivalent polyhedral problem

$$0 \in x^2 + 1 + N_X(x)$$

     where $X = \{x \mid x \geq 0 \text{ and } x \geq 1\}$

  3. Recover reduced complementarity problem

$$1 \leq x \quad \perp \quad x^2 + 1 \geq 0$$

     with the solution $x = 1$

  4. Compute multiplier $y = 2$

Variational Inequalities

Assembling Polyhedral Sets

- Given skew symmetric rows and columns
- Reject those requiring scaling or sign changes (can be relaxed)
- Use a greedy heuristic to assemble maximal polyhedral set
  - Choose a remaining skew symmetric row
  - Add next row sharing some nonzero entries if possible
  - Continue adding rows until no more rows can be added
  - Repeat to identify multiple polyhedral sets

- Given skew symmetric rows and columns
- Reject those requiring scaling or sign changes (can be relaxed)
- Use a greedy heuristic to assemble maximal polyhedral set
  - Choose a remaining skew symmetric row
  - Add next row sharing some nonzero entries if possible
  - Continue adding rows until no more rows can be added
  - Repeat to identify multiple polyhedral sets
- Reductions using polyhedral sets include
  - Duplicate rows
  - Implied variable bounds

Variational Inequalities

Assembling Polyhedral Sets

- Given skew symmetric rows and columns
- Reject those requiring scaling or sign changes (can be relaxed)
- Use a greedy heuristic to assemble maximal polyhedral set
  - Choose a remaining skew symmetric row
  - Add next row sharing some nonzero entries if possible
  - Continue adding rows until no more rows can be added
  - Repeat to identify multiple polyhedral sets
- Reductions using polyhedral sets include
  - Duplicate rows
  - Implied variable bounds
- Structure can be conveyed to capable solvers
  - A primal/dual structure recovered for optimization problems
  - Polyhedral variational inequalities for other problems

Duplicate Rows Example

1. Identify the polyhedral constraints

$$
\begin{array}{rcr}
0 \leq x & \perp & Qx - A^T\lambda + c \geq 0 \\
0 \leq y & \perp & -b^T\lambda + d \geq 0 \\
0 \leq z & \perp & b^T\lambda - d \geq 0 \\
0 \leq \lambda & \perp & Ax + by - bz \geq 0
\end{array}
$$

Variational Inequalities

Duplicate Rows Example

1. Identify the polyhedral constraints

$$\begin{array}{rcr} 0 \le x & \perp & Qx - A^T\lambda + c \ge 0 \\ 0 \le y & \perp & -b^T\lambda + d \ge 0 \\ 0 \le z & \perp & b^T\lambda - d \ge 0 \\ 0 \le \lambda & \perp & Ax + by - bz \ge 0 \end{array}$$

2. Construct the polyhedral problem

$$\begin{array}{l} 0 \in Qx - A^T\lambda + c + N_{\Re^n_+}(x) \\ 0 \in A\lambda + c + N_Y(\lambda) \end{array}$$

where $Y = \left\{ y \mid y \ge 0,\ -b^Ty + d \ge 0, \text{ and } b^Ty - d \ge 0 \right\}$

Variational Inequalities

Duplicate Rows Example

1. Identify the polyhedral constraints

$$
\begin{array}{rcr}
0 \le x & \perp & Qx - A^T\lambda + c \ge 0 \\
0 \le y & \perp & -b^T\lambda + d \ge 0 \\
0 \le z & \perp & b^T\lambda - d \ge 0 \\
0 \le \lambda & \perp & Ax + by - bz \ge 0
\end{array}
$$

2. Construct the polyhedral problem

$$
\begin{array}{l}
0 \in Qx - A^T\lambda + c + N_{\Re^n_+}(x) \\
0 \in A\lambda + c + N_Y(\lambda)
\end{array}
$$

   where $Y = \left\{ y \mid y \ge 0,\ -b^T y + d \ge 0,\ \text{and } b^T y - d \ge 0 \right\}$

3. Replace the polyhedral set

$$
Y = \left\{ y \mid y \ge 0 \text{ and } b^T y - d = 0 \right\}
$$

Variational Inequalities

Duplicate Rows Example

1. Identify the polyhedral constraints

$$\begin{array}{rcr} 0 \le x & \perp & Qx - A^T\lambda + c \ge 0 \\ 0 \le y & \perp & -b^T\lambda + d \ge 0 \\ 0 \le z & \perp & b^T\lambda - d \ge 0 \\ 0 \le \lambda & \perp & Ax + by - bz \ge 0 \end{array}$$

2. Construct the polyhedral problem

$$\begin{array}{l} 0 \in Qx - A^T\lambda + c + N_{\Re^n_+}(x) \\ 0 \in A\lambda + c + N_Y(\lambda) \end{array}$$

   where $Y = \{y \mid y \ge 0,\ -b^Ty + d \ge 0, \text{ and } b^Ty - d \ge 0\}$

3. Replace the polyhedral set

$$Y = \{y \mid y \ge 0 \text{ and } b^Ty - d = 0\}$$

4. Recover reduced mixed complementarity problem

$$\begin{array}{rcr} 0 \le x & \perp & Qx - A^T\lambda + c \ge 0 \\ y & & -b^T\lambda + d = 0 \\ 0 \le \lambda & \perp & Ax + by \ge 0 \end{array}$$

Variational Inequalities

Summary

- Critical for preprocessing to correctly match equations and variables
  - Automatically performing a good match is difficult
  - Modeler knows their problem and should convey the match

Variational Inequalities

Summary

- Critical for preprocessing to correctly match equations and variables
  - Automatically performing a good match is difficult
  - Modeler knows their problem and should convey the match
- Skew symmetric form of optimality conditions is essential
  - Needed to construct polyhedral variational inequalities
  - Check your model to obtain a report on skew symmetry

```
check_skew_symmetry yes
```

Variational Inequalities
Summary

- Critical for preprocessing to correctly match equations and variables
  - Automatically performing a good match is difficult
  - Modeler knows their problem and should convey the match
- Skew symmetric form of optimality conditions is essential
  - Needed to construct polyhedral variational inequalities
  - Check your model to obtain a report on skew symmetry

```
check_skew_symmetry yes
```

- Reductions may determine model is infeasible
  - Infeasible models should be fixed by the modeler
  - List of preprocessing reductions can be obtained

```
output_presolve_level 3
```

Section 4

Block Structure

Block Structure

Example

- Exploitation requires structural identification

$$
\begin{array}{cc|ccc|cc}
x & x & 0 & 0 & 0 & 0 & 0 \\
x & x & 0 & 0 & 0 & 0 & 0 \\
\hline
x & 0 & x & x & x & 0 & 0 \\
x & 0 & x & x & x & 0 & 0 \\
x & 0 & x & x & x & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & x & x \\
x & 0 & x & x & x & x & x
\end{array}
$$

Block Structure

Example

- Exploitation requires structural identification

$$
\begin{array}{cc|ccc|cc}
x & x & 0 & 0 & 0 & 0 & 0 \\
x & x & 0 & 0 & 0 & 0 & 0 \\
\hline
x & 0 & x & x & x & 0 & 0 \\
x & 0 & x & x & x & 0 & 0 \\
x & 0 & x & x & x & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & x & x \\
x & 0 & x & x & x & x & x
\end{array}
$$

- Focus on small block sizes (at most 3×3)
  - Start from a single row
  - Add constraints for variables
  - Stop when no constraints to add or block too big
  - Equations removed via Schur complement when possible

Block Structure

Example

- Exploitation requires structural identification

$$
\begin{array}{cc|ccc|cc}
x & x & 0 & 0 & 0 & 0 & 0 \\
x & x & 0 & 0 & 0 & 0 & 0 \\
\hline
x & 0 & x & x & x & 0 & 0 \\
x & 0 & x & x & x & 0 & 0 \\
x & 0 & x & x & x & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & x & x \\
x & 0 & x & x & x & x & x
\end{array}
$$

- Focus on small block sizes (at most 3×3)
  - Start from a single row
  - Add constraints for variables
  - Stop when no constraints to add or block too big
  - Equations removed via Schur complement when possible
- Apply reductions
  - Preblocks use uniqueness arguments
  - Postblocks use existence arguments

Block Structure

Example

- Exploitation requires structural identification

$$
\begin{array}{cc|ccc|cc}
x & x & 0 & 0 & 0 & 0 & 0 \\
x & x & 0 & 0 & 0 & 0 & 0 \\
\hline
x & 0 & x & x & x & 0 & 0 \\
x & 0 & x & x & x & 0 & 0 \\
x & 0 & x & x & x & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & x & x \\
x & 0 & x & x & x & x & x
\end{array}
$$

- Focus on small block sizes (at most 3×3)
  - Start from a single row
  - Add constraints for variables
  - Stop when no constraints to add or block too big
  - Equations removed via Schur complement when possible
- Apply reductions
  - Preblocks use uniqueness arguments
  - Postblocks use existence arguments
- Matrix classes form the foundation for these methods

Block Structure

Preblocks and Uniqueness

- Small linear complementarity problem

$$0 \leq x \leq u \quad \perp \quad Mx + q$$

- Existence and uniqueness for fixed vector q

Block Structure

Preblocks and Uniqueness

- Small linear complementarity problem

$$0 \leq x \leq u \quad \perp \quad Mx + q$$

- Existence and uniqueness for fixed vector q
- Partial uniqueness for fixed vector q
  - Remaining problem does not rely on non-unique components
  - Remaining block can be reformulated

Block Structure

Preblocks and Uniqueness

- Small linear complementarity problem

$$0 \leq x \leq u \quad \perp \quad Mx + q$$

- Existence and uniqueness for fixed vector q
- Partial uniqueness for fixed vector q
  - Remaining problem does not rely on non-unique components
  - Remaining block can be reformulated
- Computational method is applied for 2×2 blocks
  - Compute all solutions
  - Eliminate if solution is unique
  - Check compact reformulation if not unique

Block Structure

Preblocks and Uniqueness

- Small linear complementarity problem

$$0 \le x \le u \quad \perp \quad Mx + q$$

- Existence and uniqueness for fixed vector q
- Partial uniqueness for fixed vector q
  - Remaining problem does not rely on non-unique components
  - Remaining block can be reformulated
- Computational method is applied for 2×2 blocks
  - Compute all solutions
  - Eliminate if solution is unique
  - Check compact reformulation if not unique
- For larger preblock matrices
  - Test for P matrix
  - Compute unique solution and eliminate

Block Structure

Postblocks and Existence

- Small linear complementarity problem

$$0 \le x \le u \quad \perp \quad Mx + Q$$

- Existence for fixed set Q (lower and upper function bounds)

Block Structure

Postblocks and Existence

- Small linear complementarity problem

$$0 \leq x \leq u \quad \perp \quad Mx + Q$$

- Existence for fixed set Q (lower and upper function bounds)
- Computational method can be applied for 2×2 blocks
  - Evaluate each subproblem
  - Determine sets for which solution exists
  - If Q is a subset of the union, then eliminate
  - Otherwise, compute Q for which no solution exists
  - Use information to (sometimes) restrict remaining variables

Block Structure

Postblocks and Existence

- Small linear complementarity problem

$$0 \leq x \leq u \quad \perp \quad Mx + Q$$

- Existence for fixed set Q (lower and upper function bounds)
- Computational method can be applied for 2×2 blocks
  - Evaluate each subproblem
  - Determine sets for which solution exists
  - If Q is a subset of the union, then eliminate
  - Otherwise, compute Q for which no solution exists
  - Use information to (sometimes) restrict remaining variables
- For larger postblock matrices
  - Existence of trivial solution when $Q \geq 0$
    - Generalize condition for upper bounds
  - Existence for compact sets $0 \leq x \leq u < \infty$
  - Otherwise test for strictly semimonotone matrix

Block Structure

Embedded Blocks and Forcing Conditions

- Determine subproblems with unique solution
  - Find a small index set α such that $M_{\alpha,\alpha}$ is strictly semimonotone
  - Determine possible right-hand sides Q_α
  - If $Q_\alpha \geq 0$, then fix $x_\alpha = 0$ and eliminate

Block Structure

Embedded Blocks and Forcing Conditions

- Determine subproblems with unique solution
  - Find a small index set α such that $M_{\alpha,\alpha}$ is strictly semimonotone
  - Determine possible right-hand sides Q_α
  - If $Q_\alpha \geq 0$, then fix $x_\alpha = 0$ and eliminate
- All strictly semimonotone 2×2 matrices
  - Positive diagonal for singletons
  - Positive diagonals with one positive off diagonal for doubletons
  - Positive diagonals with positive determinant for doubletons
- Identification based the Jacobian structure

Block Structure

Embedded Blocks and Forcing Conditions

- Determine subproblems with unique solution
  - Find a small index set α such that $M_{\alpha,\alpha}$ is strictly semimonotone
  - Determine possible right-hand sides Q_α
  - If $Q_\alpha \geq 0$, then fix $x_\alpha = 0$ and eliminate
- All strictly semimonotone 2×2 matrices
  - Positive diagonal for singletons
  - Positive diagonals with one positive off diagonal for doubletons
  - Positive diagonals with positive determinant for doubletons
- Identification based the Jacobian structure
- Currently implemented only for singleton subproblems

Block Structure

Summary

- Matrix classes are foundational in preprocessing blocks
  - Evaluate small blocks – brute force is possible
  - Otherwise look for identifiable submatrices
    - P matrices
    - Small strictly semimonotone matrices
- Make improvements to the model
  - Only some rules are enabled by default
  - More expensive rules turned on with options

Block Structure

Summary

- Matrix classes are foundational in preprocessing blocks
  - Evaluate small blocks – brute force is possible
  - Otherwise look for identifiable submatrices
    - P matrices
    - Small strictly semimonotone matrices
- Make improvements to the model
  - Only some rules are enabled by default
  - More expensive rules turned on with options
- Reductions may determine model is infeasible
  - Infeasible models should be fixed by the modeler
  - List of preprocessing reductions can be obtained

```
output_presolve_level 3
```

Other Preprocessing Operations

- Duplicate rows
- Duplicate columns
- Inequality based reductions
  - Forcing conditions
    - Complicated by preserving squareness and no side inequalities
    - Iterative procedure that constructs implications
    - Analogous to logic tables in integer programs
  - Implied bounds

Other Preprocessing Operations

- Duplicate rows
- Duplicate columns
- Inequality based reductions
  - Forcing conditions
    - Complicated by preserving squareness and no side inequalities
    - Iterative procedure that constructs implications
    - Analogous to logic tables in integer programs
  - Implied bounds

Analysis can be very complicated!

Section 5

Diagnostics

Diagnostics

Produce Simple Report

```
INITIAL POINT STATISTICS
Maximum of X. . . . . . . . . .  3.0000e+02 var: (x(seattle,chicago))
Maximum of F. . . . . . . . . .  3.8922e+02 eqn: (prdemand(chicago))
Maximum of Grad F . . . . . . .  1.0811e+04 eqn: (prdemand(chicago))
                                            var: (p(chicago))

INITIAL JACOBIAN NORM STATISTICS
Maximum Row Norm. . . . . . . .  1.0813e+04 eqn: (prdemand(chicago))
Minimum Row Norm. . . . . . . .  2.0000e+00 eqn: (profit(seattle,new-york))
Maximum Column Norm . . . . . .  1.0813e+04 var: (p(chicago))
Minimum Column Norm . . . . . .  2.0000e+00 var: (x(seattle,new-york))
```

Summary

- Critical for preprocessing to correctly match equations and variables
  - Automatically performing a good match is difficult
  - Modeler knows their problem and should convey the match

Summary

- Critical for preprocessing to correctly match equations and variables
  - Automatically performing a good match is difficult
  - Modeler knows their problem and should convey the match
- Skew symmetric form of optimality conditions is essential
  - Needed to construct polyhedral variational inequalities
  - Check your model to obtain a report on skew symmetry

```
check_skew_symmetry yes
```

Summary

- Critical for preprocessing to correctly match equations and variables
  - Automatically performing a good match is difficult
  - Modeler knows their problem and should convey the match
- Skew symmetric form of optimality conditions is essential
  - Needed to construct polyhedral variational inequalities
  - Check your model to obtain a report on skew symmetry

```
check_skew_symmetry yes
```

- Matrix classes are foundational in preprocessing blocks
  - Evaluate small blocks – brute force is possible
  - Otherwise look for identifiable submatrices

Summary

- Critical for preprocessing to correctly match equations and variables
  - Automatically performing a good match is difficult
  - Modeler knows their problem and should convey the match
- Skew symmetric form of optimality conditions is essential
  - Needed to construct polyhedral variational inequalities
  - Check your model to obtain a report on skew symmetry

```
check_skew_symmetry yes
```

- Matrix classes are foundational in preprocessing blocks
  - Evaluate small blocks – brute force is possible
  - Otherwise look for identifiable submatrices
- Reductions may determine model is infeasible
  - Infeasible models should be fixed by the modeler
  - List of preprocessing reductions can be obtained

```
output_presolve_level 3
```

  - Analysis can be very complicated!

Summary

- Critical for preprocessing to correctly match equations and variables
  - Automatically performing a good match is difficult
  - Modeler knows their problem and should convey the match
- Skew symmetric form of optimality conditions is essential
  - Needed to construct polyhedral variational inequalities
  - Check your model to obtain a report on skew symmetry

```
check_skew_symmetry yes
```

- Matrix classes are foundational in preprocessing blocks
  - Evaluate small blocks – brute force is possible
  - Otherwise look for identifiable submatrices
- Reductions may determine model is infeasible
  - Infeasible models should be fixed by the modeler
  - List of preprocessing reductions can be obtained

```
output_presolve_level 3
```

  - Analysis can be very complicated!
- Diagnostics can help identify scaling problems

Part III

Solving Complementarity Problems

Motivation

Given an analyzed complementarity problem:

- What numerical methods are available?
- How does one know a solution is computed?
- What can be done to address any issues uncovered?

Motivation

Given an analyzed complementarity problem:

- What numerical methods are available?
- How does one know a solution is computed?
- What can be done to address any issues uncovered?

Based on PATH solver with specified options

Section 6

Numerical Methods

- Sequential linearization methods (PATH)
  1. Solve the linear complementarity problem

$$0 \leq x \quad \perp \quad F(x_k) + \nabla F(x_k)(x - x_k) \geq 0$$

  2. Perform a line search along merit function
  3. Repeat until convergence

Numerical Methods

Overview

- Sequential linearization methods (PATH)
  1. Solve the linear complementarity problem

$$0 \leq x \quad \perp \quad F(x_k) + \nabla F(x_k)(x - x_k) \geq 0$$

  2. Perform a line search along merit function
  3. Repeat until convergence
- Semismooth reformulation methods (SEMI)
  - Solve linear system of equations to obtain direction
  - Globalize with a trust region or line search
- Interior-point methods

Numerical Methods

Semismooth Reformulation

- Define Fischer-Burmeister function

$$\phi(a, b) := a + b - \sqrt{a^2 + b^2}$$

  - $\phi(a, b) = 0$ iff $a \geq 0$, $b \geq 0$, and $ab = 0$
- Define the system

$$[\Phi(x)]_i = \phi(x_i, F_i(x))$$

  - x^* solves complementarity problem iff $\Phi(x^*) = 0$
- Nonsmooth system of equations

1. Calculate $H^k \in \partial_B \Phi(x^k)$ and solve the following system for d^k:

$$H^k d^k = -\Phi(x^k)$$

If this system either has no solution, or

$$\nabla \Psi(x^k)^T d^k \leq -p_1 \|d^k\|^{p_2}$$

is not satisfied, let $d^k = -\nabla \Psi(x^k)$.

Numerical Methods

Semismooth Algorithm

1. Calculate $H^k \in \partial_B \Phi(x^k)$ and solve the following system for d^k:

   $$H^k d^k = -\Phi(x^k)$$

   If this system either has no solution, or

   $$\nabla\Psi(x^k)^T d^k \leq -p_1 \|d^k\|^{p_2}$$

   is not satisfied, let $d^k = -\nabla\Psi(x^k)$.
2. Compute smallest nonnegative integer i^k such that

   $$\Psi(x^k + \beta^{i^k} d^k) \leq \Psi(x^k) + \sigma\beta^{i^k} \nabla\Psi(x^k) d^k$$

3. Set $x^{k+1} = x^k + \beta^{i^k} d^k$, $k = k + 1$, and go to 1.

Section 7

PATH Algorithm

PATH Algorithm

Overview

1. Crash a basis and determine proximal point
2. Solve the linearized complementarity problem
3. Perform a search with semismooth merit function
4. Restart with different options when the method fails

PATH Algorithm

Crashing a Basis

- Compute reduced problem

$$
\begin{aligned}
J_1 &= \{i \mid \ell_i < x_i < u_i\} \\
J_2 &= \{i \mid \ell_i = x_i \text{ and } F_i(x) < 0\} \\
J_3 &= \{i \mid u_i = x_i \text{ and } F_i(x) > 0\} \\
J &= J_1 \cup J_2 \cup J_3
\end{aligned}
$$

- Calculate modified direction

$$
[\nabla F(x) + \mu I]_{J,J}\, d_J = -F_J(x)
$$

- Perturbation $\mu > 0$ chosen to prevent singularity
- Search along direction to minimize merit function
- Update x, μ and repeat

Relevant options: `crash_method` and `crash_perturb`.

PATH Algorithm

Solving Linearization

- Variant of Lemke's method for solving

$$0 \in Mx + q + N_B(x)$$

  - Construct a piecewise linear path
  - Simplex method with complementarity pivoting

PATH Algorithm

Solving Linearization

- Variant of Lemke's method for solving

$$0 \in Mx + q + N_B(x)$$

  - Construct a piecewise linear path
  - Simplex method with complementarity pivoting
- Regular starts
  - Given initial starting point x^k
  - Construct invertible basis
  - Track path from x^k to $\bar{x}^k$
  - Piecewise linear path may cycle

PATH Algorithm

Solving Linearization

- Variant of Lemke's method for solving

$$0 \in Mx + q + N_B(x)$$

  - Construct a piecewise linear path
  - Simplex method with complementarity pivoting
- Regular starts
  - Given initial starting point x^k
  - Construct invertible basis
  - Track path from x^k to $\bar{x}^k$
  - Piecewise linear path may cycle
- Lemke ray starts
  - Start from a ray
  - Path does not cycle for nondegenerate problems
  - Many pivots usually required from all slack basis
  - Advanced basis can be chosen to reduce pivots
  - Used during first major iteration if solve fails

Relevant options: `lemke_start` and `lemke_start_type`.

PATH Algorithm

Linear Method I

1. Check given basis for rank deficiency
   - Eliminate linearly dependent columns
   - Replace with slack or artificial variables
   - If process fails, then use all slack basis
2. Construct covering vector that enters basis
3. Determine leaving variable by ratio test
   - Expand and devex ratio tests implemented
   - Priority assigned to artificial variables
   - If no leaving variable, then stop with ray termination
4. Determine entering variable by complementarity pivoting
   - If x_i leaves basis, then corresponding slack enters
   - If slack leaves basis, then corresponding x_i enters
   - If covering vector leaves basis, then stop at solution

PATH Algorithm

Linear Method II

5. Check for cycling in piecewise linear path
   - Count number of times variable enters basis
   - Reset counters when
     - Artificial variable leaves basis
     - Homotopy parameter larger than previous maximum
   - If count larger than threshold, then stop with cycling
6. If iteration limit reached, then stop
7. Go to Step 3
8. Refine solution
   - Project nonbasic variables onto bounds
   - Refactor the basis
   - Compute values for the basic variables
   - Project basic variables onto bounds

PATH Algorithm

Linear Algebra and Options

- LUSOL used for factorization and updates (Saunders)
  - Markovitz strategy for pivot selection
  - Threshold partial pivoting
  - Sparse rank-1 updates
- Relevant tolerance options
  - `factorization_small_tolerance`
  - `factorization_pivot_tolerance`
  - `factorization_zero_tolerance`
  - `factorization_update_limit`
- Diagnosing rank deficiency
  - `output_warnings`
  - `output_factorization_singularities`
- Cycling and artificial variables
  - `output_minor_iteration_frequency`

PATH Algorithm

Additional Details

- Scale the linear problems

PATH Algorithm
Additional Details

- Scale the linear problems
- Globalize with Fischer-Burmeister merit function
  - Nonmonotone search with watchdog strategy
  - Gradient steps used when direction is not good
  - Projections onto box within search

PATH Algorithm

Additional Details

- Scale the linear problems
- Globalize with Fischer-Burmeister merit function
  - Nonmonotone search with watchdog strategy
  - Gradient steps used when direction is not good
  - Projections onto box within search
- Apply proximal point with resets
  - Used to deal with singularity problems
  - Modifies nonlinear problem with diagonal perturbation
  - Reset when difficulty encountered by linear solver

PATH Algorithm

Additional Details

- Scale the linear problems
- Globalize with Fischer-Burmeister merit function
  - Nonmonotone search with watchdog strategy
  - Gradient steps used when direction is not good
  - Projections onto box within search
- Apply proximal point with resets
  - Used to deal with singularity problems
  - Modifies nonlinear problem with diagonal perturbation
  - Reset when difficulty encountered by linear solver
- Use spacer steps performed after each major iteration

PATH Algorithm

Additional Details

- Scale the linear problems
- Globalize with Fischer-Burmeister merit function
  - Nonmonotone search with watchdog strategy
  - Gradient steps used when direction is not good
  - Projections onto box within search
- Apply proximal point with resets
  - Used to deal with singularity problems
  - Modifies nonlinear problem with diagonal perturbation
  - Reset when difficulty encountered by linear solver
- Use spacer steps performed after each major iteration
- Restart when necessary
  - Detect when algorithm is stalling
  - Restart algorithm with alternate options

Most relevant options: `nms_mstep_frequency` and `nms_memory_size`.

PATH Algorithm

Merit Functions

- Many merit functions can be used to measure progress
  - Complementarity error: $\|(-x_+), (-F(x))_+, x_+ \odot F(x)_+\|$
  - Normal map: $\|F(x_+) + x - x_+\|$
  - Minimum map: $\| \min \{x, F(x)\} \|$
  - Fischer-Burmeister function: $\|\Phi(x, F(x))\|$
  - Gradient of Fischer-Burmeister function: $\nabla \left[\frac{1}{2}\|\Phi(x, F(x))\|_2^2\right]$

PATH Algorithm

Merit Functions

- Many merit functions can be used to measure progress
  - Complementarity error: $\|(-x_+), (-F(x))_+, x_+ \odot F(x)_+\|$
  - Normal map: $\|F(x_+) + x - x_+\|$
  - Minimum map: $\| \min \{x, F(x)\} \|$
  - Fischer-Burmeister function: $\|\Phi(x, F(x))\|$
  - Gradient of Fischer-Burmeister function: $\nabla \left[\frac{1}{2}\|\Phi(x, F(x))\|_2^2\right]$
- Observe vastly different values for some problems
  - Particularly when $x \to \infty$ or $F(x) \to \infty$
  - Can happen when domain violations occur at solution

PATH Algorithm

Merit Functions

- Many merit functions can be used to measure progress
  - Complementarity error: $\|(-x_+), (-F(x))_+, x_+ \odot F(x)_+\|$
  - Normal map: $\|F(x_+) + x - x_+\|$
  - Minimum map: $\| \min \{x, F(x)\} \|$
  - Fischer-Burmeister function: $\|\Phi(x, F(x))\|$
  - Gradient of Fischer-Burmeister function: $\nabla \left[\frac{1}{2}\|\Phi(x, F(x))\|_2^2\right]$
- Observe vastly different values for some problems
  - Particularly when $x \to \infty$ or $F(x) \to \infty$
  - Can happen when domain violations occur at solution
- Want *all* to be near zero to trust the solution

```
Inf-Norm of Complementarity . .  1.0311e-11 eqn: (profit(san-diego,new-york))
Inf-Norm of Normal Map. . . . .  1.1369e-13 eqn: (prdemand(new-york))
Inf-Norm of Minimum Map . . . .  1.1369e-13 eqn: (prdemand(new-york))
Inf-Norm of Fischer Function. .  1.1369e-13 eqn: (prdemand(new-york))
Inf-Norm of Grad Fischer Fcn. .  2.4809e-10 eqn: (prdemand(topeka))
Two-Norm of Grad Fischer Fcn. .  3.4958e-10
```

- Superlinear/quadratic convergence – best outcome

PATH Algorithm

Convergence Behaviors

- Superlinear/quadratic convergence – best outcome
- Linear convergence
  - Far from a solution – merit function is large
  - Jacobian is incorrect – disrupts quadratic convergence
  - Jacobian is rank deficient – gradient of merit function is small
  - Converge to local minimizer – guarantees rank deficiency
  - Limits of finite precision arithmetic
    1. Merit function converges quadratically to small number
    2. Merit function hovers around that number with no progress

PATH Algorithm

Convergence Behaviors

- Superlinear/quadratic convergence – best outcome
- Linear convergence
  - Far from a solution – merit function is large
  - Jacobian is incorrect – disrupts quadratic convergence
  - Jacobian is rank deficient – gradient of merit function is small
  - Converge to local minimizer – guarantees rank deficiency
  - Limits of finite precision arithmetic
    1. Merit function converges quadratically to small number
    2. Merit function hovers around that number with no progress
- Domain violations – excessive backtracking in line search

Summary I

- Always ensure functions and Jacobians are defined
  - Confront and reformulate at the modeling stage
  - Report remaining domain violations to solver
  - Provide a good starting point

Summary I

- Always ensure functions and Jacobians are defined
  - Confront and reformulate at the modeling stage
  - Report remaining domain violations to solver
  - Provide a good starting point
- Always match equations to appropriate variables
  - Automatically performing a good match is difficult
  - Modeler knows their problem and should convey the match
  - Required for effective preprocessing

Summary I

- Always ensure functions and Jacobians are defined
  - Confront and reformulate at the modeling stage
  - Report remaining domain violations to solver
  - Provide a good starting point
- Always match equations to appropriate variables
  - Automatically performing a good match is difficult
  - Modeler knows their problem and should convey the match
  - Required for effective preprocessing
- Always prefer a monotone formulation over other formulations
  - Use skew symmetric version of optimality conditions
  - Required for preprocessing and proximal perturbation
  - Check the skew symmetry report when in doubt

Summary II

- Analysis may determine model is infeasible
  - Infeasible models should be fixed by the modeler
  - List of preprocessing reductions can be obtained

Summary II

- Analysis may determine model is infeasible
  - Infeasible models should be fixed by the modeler
  - List of preprocessing reductions can be obtained
- Use scaling when appropriate
  - Any consistent scaling for optimization problems
  - Symmetric diagonal scaling for complementarity problems
  - Beware when using unscaled solution!

Summary II

- Analysis may determine model is infeasible
  - Infeasible models should be fixed by the modeler
  - List of preprocessing reductions can be obtained
- Use scaling when appropriate
  - Any consistent scaling for optimization problems
  - Symmetric diagonal scaling for complementarity problems
  - Beware when using unscaled solution!
- Rank deficiency can still be an issue
  - Solver can sometimes identify and eliminate dependencies
  - Solver may add a positive diagonal perturbation
  - Use linear algebra options to obtain report

Summary II

- Analysis may determine model is infeasible
  - Infeasible models should be fixed by the modeler
  - List of preprocessing reductions can be obtained
- Use scaling when appropriate
  - Any consistent scaling for optimization problems
  - Symmetric diagonal scaling for complementarity problems
  - Beware when using unscaled solution!
- Rank deficiency can still be an issue
  - Solver can sometimes identify and eliminate dependencies
  - Solver may add a positive diagonal perturbation
  - Use linear algebra options to obtain report
- Always check the solution reported
  - Look for convergence issues in the iteration log
  - Make sure *all* the merit functions are close to zero