LCP - Linear Complementarity Problems

A Short Introduction to Definitions and Numerical Methods

Kenny Erleben
Department of Computer Science

The Definition

The Complementarity Problem

If given $x, y \in \mathbb{R}$ where

$$x \geq 0$$
$$y \geq 0$$

and

$$x > 0 \Rightarrow y = 0$$
$$y > 0 \Rightarrow x = 0$$

Then we have a complementarity problem.

The Linear Complementarity Problem (1/2)

If for some $a, b \in \mathbb{R}$

$$y = ax + b$$

and

$$x \geq 0$$
$$y \geq 0$$

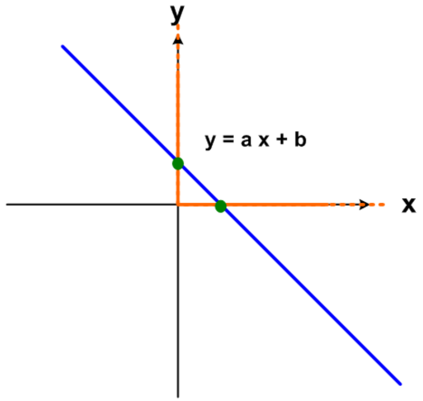
Then we have a Linear Complementarity Problem (LCP).

The Linear Complementarity Problem (2/2)

More compact notation

$$\begin{aligned} x &\geq 0 \\ ax + b &\geq 0 \\ x(ax+b) &= 0 \end{aligned}$$

The Geometry



How many #Solutions of LCP?

Hint: Try to examine signs of a and b

	$b < 0$	$b = 0$	$b > 0$
$a < 0$			
$a = 0$			
$a > 0$			

Answer of # Solutions

Verify this using geometry

	$b < 0$	$b = 0$	$b > 0$
$a < 0$	0	1	2
$a = 0$	0	∞	1
$a > 0$	1	1	1

Going to Higher Dimensions

Let $\mathbf{b}, \mathbf{x} \in \mathbb{R}^n$ and $\mathbf{A} \in \mathbb{R}^{n \times n}$ so $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{b}$,

$$\begin{aligned} \mathbf{x}_i &\geq 0 \quad \forall i \in [1..n] \\ (\mathbf{A}\mathbf{x} + \mathbf{b})_i &\geq 0 \quad \forall i \in [1..n] \\ \mathbf{x}_i(\mathbf{A}\mathbf{x} + \mathbf{b})_i &= 0 \quad \forall i \in [1..n] \end{aligned}$$

In Matrix-Vector Notation

$$\begin{aligned} \mathbf{x} &\geq 0 \\ (\mathbf{A}\mathbf{x} + \mathbf{b}) &\geq 0 \\ \mathbf{x}^T(\mathbf{A}\mathbf{x} + \mathbf{b}) &= 0 \end{aligned}$$

How can we solve this?

What kind of problem is a LCP formulation?

What do you think?

- A constrained minimization problem?
- A nonlinear equation (root search problem) ?
- A fixed-point problem?
- A combinatorial problem?
- Something else?

A NAÏVE Active Set/Pivoting Method

Guessing A Solution

Given the index set $\mathcal{I} = \{1, \ldots, n\}$ define

$$\mathcal{F} = \{i \quad | \quad i \in \mathcal{I} \quad \wedge \quad \mathbf{y}_i > 0\}$$
$$\mathcal{A} = \{i \quad | \quad i \notin \mathcal{F} \quad \wedge \quad \mathbf{y}_i = 0\}$$

Make "lucky" guess of $\mathcal{F}$ and $\mathcal{A}$,

$$\begin{bmatrix} \mathbf{y}_\mathcal{A} \\ \mathbf{y}_\mathcal{F} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_{\mathcal{A}\mathcal{A}} & \mathbf{A}_{\mathcal{A}\mathcal{F}} \\ \mathbf{A}_{\mathcal{F}\mathcal{A}} & \mathbf{A}_{\mathcal{F}\mathcal{F}} \end{bmatrix} \begin{bmatrix} \mathbf{x}_\mathcal{A} \\ \mathbf{x}_\mathcal{F} \end{bmatrix} + \begin{bmatrix} \mathbf{b}_\mathcal{A} \\ \mathbf{b}_\mathcal{F} \end{bmatrix}$$

By assumption $\mathbf{y}_\mathcal{F} > 0 \Rightarrow \mathbf{x}_\mathcal{F} = 0$

$$\begin{bmatrix} 0 \\ \mathbf{y}_\mathcal{F} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_{\mathcal{A}\mathcal{A}} & \mathbf{A}_{\mathcal{A}\mathcal{F}} \\ \mathbf{A}_{\mathcal{F}\mathcal{A}} & \mathbf{A}_{\mathcal{F}\mathcal{F}} \end{bmatrix} \begin{bmatrix} \mathbf{x}_\mathcal{A} \\ 0 \end{bmatrix} + \begin{bmatrix} \mathbf{b}_\mathcal{A} \\ \mathbf{b}_\mathcal{F} \end{bmatrix}$$

Verify if Guess was a Solution

So

$$\begin{bmatrix} 0 \\ \mathbf{y}_{\mathcal{F}} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_{\mathcal{A}\mathcal{A}}\mathbf{x}_{\mathcal{A}} + \mathbf{b}_{\mathcal{A}} \\ \mathbf{A}_{\mathcal{F}\mathcal{A}}\mathbf{x}_{\mathcal{A}} + \mathbf{b}_{\mathcal{F}} \end{bmatrix}$$

Compute

$$\mathbf{x}_{\mathcal{A}} = -\mathbf{A}_{\mathcal{A}\mathcal{A}}^{-1}\mathbf{b}_{\mathcal{A}}$$

Verify

$$\mathbf{x}_{\mathcal{A}} \geq 0$$

Compute

$$\mathbf{y}_{\mathcal{F}} = \mathbf{A}_{\mathcal{F}\mathcal{A}}\mathbf{x}_{\mathcal{A}} + \mathbf{b}_{\mathcal{F}}$$

Verify

$$\mathbf{y}_{\mathcal{F}} > 0$$

How Many Guesses?

We only need

$$\mathbf{A}_{\mathcal{A}\mathcal{A}}^{-1}$$

Hopefully

$$\| \mathcal{A} \| \ll n$$

Cool this will be fast!

How many guesses do we need?

Answer: Non-Polynomial Complexity

Worst case time complexity of guessing

$$\mathcal{O}(n^3 2^n)$$

Not computational very efficient!

Connection to Positive Cones (Linear Programming)

First some algebra on $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{b}$,

$$\mathbf{I}\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{b}$$
$$\begin{bmatrix} \mathbf{I} & -\mathbf{A} \end{bmatrix} \begin{bmatrix} \mathbf{y} \\ \mathbf{x} \end{bmatrix} = \mathbf{b}$$

Make guesses $\mathcal{F} = \{i | \mathbf{y}_i \geq 0\}$ and $\mathcal{A} = \{i | \mathbf{x}_i \geq 0\}$ so $\mathcal{F} \cap \mathcal{A} = \emptyset$ and $\mathcal{F} \cup \mathcal{A} = \{1, \ldots, n\}$

$$\underbrace{\begin{bmatrix} \mathbf{I}_{\cdot\mathcal{F}} & -\mathbf{A}_{\cdot\mathcal{A}} \end{bmatrix}}_{\mathbf{M}} \underbrace{\begin{bmatrix} \mathbf{y}_{\mathcal{F}} \\ \mathbf{x}_{\mathcal{A}} \end{bmatrix}}_{\mathbf{z}} = \mathbf{b}$$

Connection to Positive Cones (Linear Programming)

Verify if LP

$$\mathbf{Mz} = \mathbf{b} \quad \text{subject to} \quad \mathbf{z} \geq 0$$

Has a solution (same as **b** in positive cone of **M**). If true then we have a solution for the LCP too.

The Projected Gauss-Seidel Method

Use a Splitting Method

Use the splitting

$$\mathbf{A} = \mathbf{M} - \mathbf{N}$$

then

$$\begin{aligned} \mathbf{Mx} - \mathbf{Nx} + \mathbf{b} &\geq 0 \\ \mathbf{x} &\geq 0 \\ (\mathbf{x})^T(\mathbf{Mx} - \mathbf{Nx} + \mathbf{b}) &= 0 \end{aligned}$$

Introduce the iteration indices

Assume $\mathbf{x}^k \rightarrow \mathbf{x}^{k+1}$ for $k \rightarrow \infty$ then

$$\begin{aligned} \mathbf{M}\mathbf{x}^{k+1} - \mathbf{N}\mathbf{x}^k + \mathbf{b} &\geq 0 \\ \mathbf{x}^{k+1} &\geq 0 \\ (\mathbf{x}^{k+1})^T(\mathbf{M}\mathbf{x}^{k+1} - \mathbf{N}\mathbf{x}^k + \mathbf{b}) &= 0 \end{aligned}$$

Use Discretization ⇒ Fixed Point Formulation

Thus, we have created a sequence of sub-problems

$$\begin{aligned} \mathbf{M}\mathbf{x}^{k+1} + \mathbf{c}^k &\geq 0 \\ \mathbf{x}^{k+1} &\geq 0 \\ (\mathbf{x}^{k+1})^T(\mathbf{M}\mathbf{x}^{k+1} + \mathbf{c}^k) &= 0 \end{aligned}$$

where

$$\mathbf{c}^k = \mathbf{b} - \mathbf{N}\mathbf{x}^k$$

This is a fixed-point problem.

Use Minimum Map Reformulation

Given subproblem

$$\begin{aligned} \mathbf{M}\mathbf{x}^{k+1} + \mathbf{c}^{k} &\geq 0 \\ \mathbf{x}^{k+1} &\geq 0 \\ (\mathbf{x}^{k+1})^{T}(\mathbf{M}\mathbf{x}^{k+1} + \mathbf{c}^{k}) &= 0 \end{aligned}$$

Same as (Why?)

$$\underbrace{\min(\mathbf{x}^{k+1}, \mathbf{M}\mathbf{x}^{k+1} + \mathbf{c}^{k})}_{\mathbf{H}(\mathbf{x}^{k+1})} = 0$$

A root search problem: $\mathbf{H}(\mathbf{x}^{k+1}) = 0$.

The Minimum Map Formulation

Say $a, b \in \mathbb{R}$ are complimentary

$$a > 0 \Rightarrow b = 0$$
$$b > 0 \Rightarrow a = 0$$

Let us look at the minimum map

$\min(a, b)$	$a > 0$	$a = 0$	$a < 0$
$b > 0$	$+$	0	$-$
$b = 0$	0	0	$-$
$b < 0$	$-$	$-$	$-$

Same solutions as the complementarity problem.

More Clever Manipulation

So

$$\min(\mathbf{x}^{k+1}, \mathbf{M}\mathbf{x}^{k+1} + \mathbf{c}^k) = 0$$

Subtract $\mathbf{x}^{k+1}$

$$\min(0, \mathbf{M}\mathbf{x}^{k+1} + \mathbf{c}^k - \mathbf{x}^{k+1}) = -\mathbf{x}^{k+1}$$

Multiply by minus one

$$\underbrace{\max(0, -\mathbf{M}\mathbf{x}^{k+1} - \mathbf{c}^k + \mathbf{x}^{k+1})}_{\mathbf{F}(\mathbf{x}^{k+1})} = \mathbf{x}^{k+1}$$

A fixed point formulation: $\mathbf{F}(\mathbf{x}^{k+1}) = \mathbf{x}^{k+1}$.

Do A Case-by-Case Analysis

So

$$\max(0, -\mathbf{M}\mathbf{x}^{k+1} - \mathbf{c}^{k} + \mathbf{x}^{k+1}) = \mathbf{x}^{k+1}$$

If

$$(-\mathbf{M}\mathbf{x}^{k+1} - \mathbf{c}^{k} + \mathbf{x}^{k+1})_i \leq 0$$

Then

$$\mathbf{x}_i^{k+1} = 0$$

Else

$$(-\mathbf{M}\mathbf{x}^{k+1} - \mathbf{c}^{k} + \mathbf{x}^{k+1})_i > 0$$

and

$$(\mathbf{x}^{k+1} - \mathbf{M}\mathbf{x}^{k+1} - \mathbf{c}^{k})_i = \mathbf{x}_i^{k+1}$$

That is

$$(\mathbf{M}\mathbf{x}^{k+1}) \qquad \mathbf{c}^{k}$$

Putting it Together

For a suitable choice of $\mathbf{M}$

$$(\mathbf{M}\mathbf{x}^{k+1})_i = -c_i^k \Rightarrow \mathbf{x}_i^{k+1} = (-\mathbf{M}^{-1}\mathbf{c}^k)_i$$

Back-substitution of $\mathbf{c}^k = \mathbf{b} - \mathbf{N}\mathbf{x}^k$ we have

$$\left(\mathbf{M}^{-1}\left(\mathbf{N}\mathbf{x}^k - \mathbf{b}\right)\right)_i = \mathbf{x}_i^{k+1}$$

Insert in fixed point formulation

$$\underbrace{\max\left(0, \left(\mathbf{M}^{-1}\left(\mathbf{N}\mathbf{x}^k - \mathbf{b}\right)\right)\right)}_{\mathbf{G}(\mathbf{x}^k)} = \mathbf{x}^{k+1}$$

Closed form solution for sub problem: $\mathbf{x}^{k+1} \leftarrow \mathbf{G}(\mathbf{x}^k)$.

Final Iterative Scheme – The Projected Gauss-Seidel (PGS) method

Given $\mathbf{x}^1$ set $k = 1$

Step 1 Compute

$$\mathbf{z}^k = \left(\mathbf{M}^{-1}\left(\mathbf{N}\mathbf{x}^k - \mathbf{b}\right)\right)$$

Step 2 Compute

$$\mathbf{x}^{k+1} = \max(0, \mathbf{z}^k)$$

Step 3 If convergence then return $\mathbf{x}^{k+1}$ otherwise goto Step 1

The Projected Gauss–Seidel Method...

AGAIN!!!

Connection to Quadratic Programming (QP) Problems

Consider the minimization problem

$$\mathbf{x}^* = \arg\min_{\mathbf{x}\geq 0} \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x} + \mathbf{b}^T\mathbf{x}$$

where $\mathbf{A}$ is symmetric. First-order optimality (KKT) conditions

$$\begin{aligned} \mathbf{A}\mathbf{x} + \mathbf{b} - \mathbf{I}\mathbf{y} &= 0 \\ \mathbf{x} &\geq 0 \\ \mathbf{y} &\geq 0 \\ \mathbf{y}^T\mathbf{x} &= 0 \end{aligned}$$

Same as the LCP problem.

More on QP relation

From optimization theory, we know

- If $\mathbf{A}$ is positive definite then we have a strict convex QP with a unique solution
- If $\mathbf{A}$ is positive semi-definite then we have a convex QP where a solution exists but it is no longer unique

This is cool if we have this prior knowledge of $\mathbf{A}$.

Observe that the LCP solution needs not to be a global solution of the QP or even a minimizer. It is sufficient that the LCP solution fulfills first-order optimality only.

The QP problem – Again

The LCP problem can be restated as

$$\mathbf{x}^* = \arg\min_{\mathbf{x} \geq \mathbf{0}} f(\mathbf{x})$$

where

$$f(\mathbf{x}) \equiv \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x} + \mathbf{x}^T\mathbf{b}$$

The Idea – Sweep over Coordinates

The i^{th} unit axis vector

$$\mathbf{e}^i_j = \begin{cases} 1 & i = i \\ 0 & i \neq j \end{cases}$$

The i^{th} relaxation step solves the one-dimensional problem

$$\tau^* = \arg \min_{\mathbf{x}+\tau\mathbf{e}^i \geq \mathbf{0}} f(\mathbf{x} + \tau\mathbf{e}^i)$$

and computing $\mathbf{x} \leftarrow \mathbf{x} + \tau\mathbf{e}^i$ same as

$$x_i \leftarrow x_i + \tau$$

One relaxation cycle consists of one sequential sweep over all i’s.

A Closed-Form Solution for the i^{th} Coordinate

The object function of the one-dimensional problem

$$\begin{aligned} f(\mathbf{x}+\tau\mathbf{e}^i) &= \frac{1}{2}(\mathbf{x}+\tau\mathbf{e}^i)^T\mathbf{A}(\mathbf{x}+\tau\mathbf{e}^i) + (\mathbf{x}+\tau\mathbf{e}^i)^T\mathbf{b}, \\ &= \underbrace{\frac{1}{2}\tau^2(\mathbf{A})_{ii} + \tau(\underbrace{\mathbf{Ax}+\mathbf{b}}_{\mathbf{r}})_i}_{g(\tau)} + \underbrace{\frac{1}{2}\mathbf{x}^T\mathbf{Ax}+\mathbf{x}^T\mathbf{b}}_{f(\mathbf{x})\equiv\text{const}}. \end{aligned}$$

So

$$f(\mathbf{x}+\tau\mathbf{e}^i) = g(\tau) + f(\mathbf{x})$$

We just need to minimize $g(\tau)$

Found PGS Again

The unconstrained minimizer of $g(\tau)$

$$\tau^u = -\frac{\mathbf{r}_i}{(\mathbf{A})_{ii}}.$$

Considering the constraint $\mathbf{x}_i + \tau \geq 0$ we have

$$\tau^c = \max\left(-\frac{\mathbf{r}_i}{(\mathbf{A})_{ii}}, -\mathbf{x}_i\right)$$

The final update rule

$$\mathbf{x}_i \leftarrow \max\left(0, \mathbf{x}_i - \frac{\mathbf{r}_i}{\mathbf{A}_{ii}}\right)$$

Algebraic equivalent to the i^{th} step in PGS shown by splitting.

Lessons Learned

- PGS can be derived from a QP reformulation or by a splitting method
- Each derivation assumes different matrix properties of the $\mathbf{A}$-matrix

The Projected Succesive Over Relaxation Method

Relaxing the Steps

Given the polynomial

$$g(\tau) \equiv \frac{1}{2}\tau^2 \mathbf{A}_{ii} + \tau \mathbf{r}_i$$

where $\mathbf{A}_{ii} > 0$ then

- One trival root at $\tau^1 = 0$
- One global minima at $\tau^u = -\frac{\mathbf{r}_i}{\mathbf{A}_{ii}}$ where $g(\tau^0) < 0$
- Second root at $\tau^2 = -2\frac{\mathbf{r}_i}{\mathbf{A}_{ii}}$.

For any τ between τ^1 and τ^2

$$\tau^\lambda = -\lambda \frac{\mathbf{r}_i}{\mathbf{A}_{ii}} \quad \Rightarrow \quad g(\tau^\lambda) < 0, \quad \forall \lambda \in [0..2].$$

The Projected SOR Method

From this it follows that

$$f(\mathbf{x} + \tau^{\lambda}\mathbf{e}^{i}) = g(\tau^{\lambda}) + f(\mathbf{x}) \leq f(\mathbf{x}), \quad \forall \lambda \in [0..2]$$

With equality for $\tau^{\lambda} = 0$. This results in the over-relaxed version

$$\mathbf{x}_i \leftarrow \max\left(0, \mathbf{x}_i - \lambda \frac{\mathbf{r}_i}{\mathbf{A}_{ii}}\right).$$

Algebraic equivalent to the i^{th} step in the projected SOR[1].

The Fischer–Newton Method

The Fischer Function

The Fischer function

$$\phi(a, b) = \sqrt{a^2 + b^2} - (a + b) \quad \text{for some } a, b \in \mathbb{R}.$$

If one has the complementarity problem $a \geq 0 \perp b \geq 0$ then a solution (a^*, b^*) is only a solution if and only if $\phi(a^*, b^*) = 0$

The Fischer Reformulation

Given the LCP

$$\mathbf{x} \geq 0 \quad \perp \quad \mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{b} \geq 0$$

We reformulate

$$\mathbf{F}(\mathbf{x}) = \mathbf{F}(\mathbf{x}, \mathbf{y}) = \begin{bmatrix} \phi(\mathbf{x}_1, \mathbf{y}_1) \\ \vdots \\ \phi(\mathbf{x}_n, \mathbf{y}_n) \end{bmatrix} = \mathbf{0}$$

This is a nonsmooth root search problem

The Fischer–Newton Method

Solved using a generalized Newton–Method. In an iterative fashion solves the generalized Newton subsystem

$$\mathbf{J}\Delta\mathbf{x}^k = -F(\mathbf{x}^k)$$

for the Newton direction $\Delta\mathbf{x}^k$. Here $\mathbf{J} \in \partial F(\mathbf{x}^k)$ is any member from the generalized Jacobian $\partial F(\mathbf{x}^k)$. Then the Newton update yields

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \tau^k \Delta\mathbf{x}^k$$

where τ^k is the step length of the k^{th} iteration.

The Generalized Jacobian

Given $\mathbf{F}$

- let $\mathcal{D} \subset \mathbb{R}^n$ be the set of all $\mathbf{x} \in \mathbb{R}^n$ where $\mathbf{F}$ is continuously differentiable
- Assume $\mathbf{F}$ is Lipschits continuous at $\mathbf{x}$

The B–subdifferential of $\mathbf{F}$ at $\mathbf{x}$ is defined as

$$\partial_B \mathbf{F}(\mathbf{x}) \equiv \left\{ \mathbf{H} \in \mathbb{R}^{n \times n} \quad \middle| \quad \exists (\mathbf{x}_k) \subset \mathcal{D} \quad \text{and} \quad \lim_{\mathbf{x}_k \to \mathbf{x}} \frac{\partial \mathbf{F}(\mathbf{x}_k)}{\partial \mathbf{x}} = \mathbf{H} \right\}$$

Clarke's generalized Jacobian of $\mathbf{F}$ at $\mathbf{x}$ is defined as the convex hull of the B–subdifferential,

$$\partial \mathbf{F}(\mathbf{x}) \equiv \mathbf{co}\left(\partial_B \mathbf{F}(\mathbf{x})\right)$$

Example – The Euclidean Norm

Consider the Euclidean norm $e : \mathbb{R}^2 \mapsto \mathbb{R}$ then for $\mathbf{x} \in \mathbb{R}^2 \setminus \{\mathbf{0}\}$ we have

$$\partial e(\mathbf{x}) = \partial_B e(\mathbf{x}) = \frac{\partial e(\mathbf{x})}{\partial \mathbf{x}} = \frac{\mathbf{x}^T}{\| \mathbf{x} \|} \quad \forall \mathbf{x} \neq \mathbf{0}$$

For $\mathbf{x} = \mathbf{0}$ we have

$$\begin{aligned} \partial_B e(\mathbf{0}) &= \{\mathbf{v}^T \quad | \quad \mathbf{v} \in \mathbb{R}^2 \quad \text{and} \quad \| \mathbf{v} \| = 1\} \\ \partial e(\mathbf{0}) &= \{\mathbf{v}^T \quad | \quad \mathbf{v} \in \mathbb{R}^2 \quad \text{and} \quad \| \mathbf{v} \| \leq 1\} \end{aligned}$$

Illustration of Generalized Jacobian in 1D

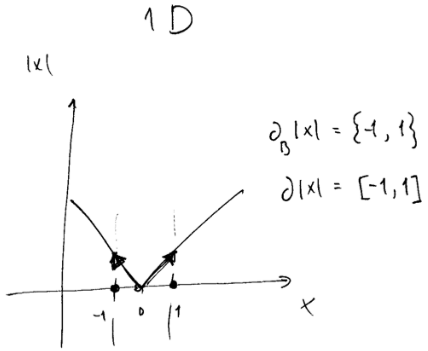
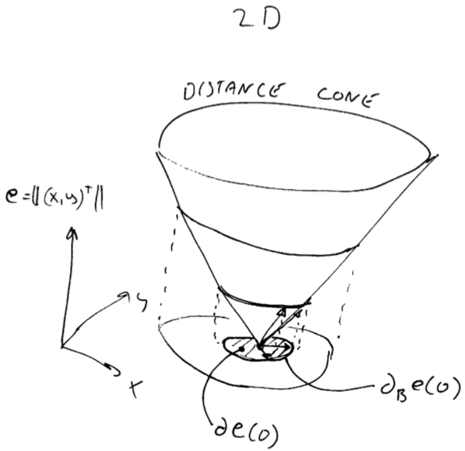


Illustration of Generalized Jacobian in 2D



Example – The Fischer Function

For $\mathbf{x} = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^T \in \mathbb{R}^2$ we may write the Fischer function as

$$\phi(\mathbf{x}) = e(\mathbf{x}) - f(\mathbf{x})$$

where $f(\mathbf{x}) = \left(\begin{bmatrix} 1 & 1 \end{bmatrix}^T \mathbf{x} \right)$. From this we find

$$\partial_B \phi(\mathbf{x}) = \partial_B e(\mathbf{x}) - \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}}$$

$$\partial \phi(\mathbf{x}) = \partial e(\mathbf{x}) - \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}}$$

Hence for $\mathbf{x} \neq 0$,

$$\partial \phi(\mathbf{x}) = \partial_B \phi(\mathbf{x}) = \left\{ \frac{\mathbf{x}^T}{\| \mathbf{x} \|} - \begin{bmatrix} 1 & 1 \end{bmatrix}^T \right\}$$

and

$$\partial_B \phi(\mathbf{0}) = \left\{ \mathbf{v}^T - \begin{bmatrix} 1 & 1 \end{bmatrix}^T \quad \middle| \quad \mathbf{v} \in \mathbb{R}^2 \quad \text{and} \quad \| \mathbf{v} \| = 1 \right\}$$

Generalized Jacobian of The Fischer Reformulation

Written as

$$\partial F(\mathbf{x}) \equiv \mathbf{D}_p(\mathbf{x}) + \mathbf{D}_q(\mathbf{x})\mathbf{A}$$

where $\mathbf{D}_p(\mathbf{x}) = \mathbf{diag}\left(p_1(\mathbf{x}), \dots, p_n(\mathbf{x})\right)$ and $\mathbf{D}_q(\mathbf{x}) = \mathbf{diag}\left(q_1(\mathbf{x}), \dots, q_n(\mathbf{x})\right)$ are diagonal matrices. If $\mathbf{y}_i \neq 0$ or $\mathbf{x}_i \neq 0$ then

$$p_i(\mathbf{x}) = \frac{\mathbf{x}_i}{\sqrt{\mathbf{x}_i^2 + \mathbf{y}_i^2}} - 1,$$
$$q_i(\mathbf{x}) = \frac{\mathbf{y}_i}{\sqrt{\mathbf{x}_i^2 + \mathbf{y}_i^2}} - 1,$$

else if $\mathbf{y}_i = \mathbf{x}_i = 0$ then

$$p_i(\mathbf{x}) = \alpha_i - 1,$$
$$q_i(\mathbf{x}) = \beta_i - 1$$

Proof of $\partial F(\mathbf{x})$

Assume $\mathbf{y}_i \neq 0$ or $\mathbf{x}_i \neq 0$ then the differential is

$$d\mathbf{F}_i(\mathbf{x},\mathbf{y}) = d\left(\left(\mathbf{x}_i^2 + \mathbf{y}_i^2\right)^{\frac{1}{2}}\right) - d\left(\mathbf{x}_i + \mathbf{y}_i\right)$$

By chain rule

$$\begin{aligned} d\mathbf{F}_i(\mathbf{x},\mathbf{y}) &= \frac{1}{2}\left(\mathbf{x}_i^2 + \mathbf{y}_i^2\right)^{-\frac{1}{2}} d\left(\mathbf{x}_i^2 + \mathbf{y}_i^2\right) - d\mathbf{x}_i - d\mathbf{y}_i \\ &= \frac{\mathbf{x}_i d\mathbf{x}_i + \mathbf{y}_i d\mathbf{y}_i}{\sqrt{\mathbf{x}_i^2 + \mathbf{y}_i^2}} - d\mathbf{x}_i - d\mathbf{y}_i \\ &= \left[\underbrace{\left(\frac{\mathbf{x}_i}{\sqrt{\mathbf{x}_i^2 + \mathbf{y}_i^2}} - 1\right)}_{} \quad \underbrace{\left(\frac{\mathbf{y}_i}{\sqrt{\mathbf{x}_i^2 + \mathbf{y}_i^2}} - 1\right)}_{}\right] \begin{bmatrix} d\mathbf{x}_i \\ d\mathbf{y}_i \end{bmatrix} \end{aligned}$$

Proof of $\partial F(\mathbf{x})$ (Contd)

Finally $d\mathbf{y} = \mathbf{A}d\mathbf{x}$, so $d\mathbf{y}_i = \mathbf{A}_{i*}d\mathbf{x}$ by substitution

$$d\mathbf{F}_i(\mathbf{x},\mathbf{y}) = \underbrace{\left(p_i(\mathbf{x})\mathbf{e}_i^T + q_i(\mathbf{x})\mathbf{A}_{i*}\right)}_{\partial F_i(\mathbf{x})} d\mathbf{x}$$

The case $\mathbf{x}_i = \mathbf{y}_i = 0$ follows from the previous examples.

How to solve Generalized Newton System

- whenever $\mathbf{x}_i = \mathbf{y}_i = 0$ one would use $\mathbf{x}'_i = \varepsilon$ in-place $\mathbf{x}_i$ when evaluating the generalized Jacobian where $0 < \varepsilon \ll 1$
- If Newton system is solved with iterative method (GMRES) then we only need to compute $\mathbf{J}\Delta\mathbf{x}$. By definition of directional derivative

$$\mathbf{J}\Delta\mathbf{x} = \lim_{h \to 0^+} \frac{\mathbf{F}(\mathbf{x} + h\Delta\mathbf{x}) - \mathbf{F}(\mathbf{x})}{h}$$

  So we can numerically approximate $\mathbf{J}\Delta\mathbf{x}$ using finite differences
- A projected Armijo back-tracking can be very usefull to globalize the Newton method and to ensure feasibility of all iterates

Projected Armijo Backtracking Line Search

Define natural merit function

$$\theta(\mathbf{x}) = \frac{1}{2}\mathbf{F}(\mathbf{x})^T\mathbf{F}(\mathbf{x})$$

Project Newton Search Direction

$$\Delta x \leftarrow \max\left(\mathbf{0}, \Delta\mathbf{x}\right)$$

Find smallest $k \in \mathbf{Z}_0$ such that

$$\theta(\mathbf{x} + \alpha^k \Delta\mathbf{x}) \leq \theta(\mathbf{x}) + \underbrace{\left(\beta\frac{\partial\theta(\mathbf{x})}{\partial\mathbf{x}}\Delta\mathbf{x}\right)}_{c\equiv\text{const}}\alpha^k$$

for some user defined constants $0 \leq \beta < \alpha < 1$. Now

$$\tau = \alpha^k$$

Main Idea of Interior Point Method

Using the slack variable $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{b}$ we write the LCP as

$$\begin{aligned} (\mathbf{A}\mathbf{x} - \mathbf{y} + \mathbf{b})_i &= 0 \\ \mathbf{x}_i\mathbf{y}_i &= 0 \\ \mathbf{x}_i, \mathbf{y}_i &\geq 0 \end{aligned}$$

The Idea is to solve this iteratively while keeping $\mathbf{x}_i^k, \mathbf{y}_i^k > 0$. To ensure invariant we introduce a postive value $\gamma > 0$ such that

$$\begin{aligned} (\mathbf{A}\mathbf{x} - \mathbf{y} + \mathbf{b})_i &= 0 \\ \mathbf{x}_i\mathbf{y}_i - \gamma &= 0 \\ \mathbf{x}_i, \mathbf{y}_i &> 0 \end{aligned}$$

Here γ is a relaxed complimentarity measure (more on this below).

The Kojima Mapping

Now we introduce the Kojima mapping which we use to reformulate the relaxed LCP formulation above

$$\mathbf{F}(\mathbf{x}, \mathbf{y} : \gamma) = \begin{bmatrix} \mathbf{Ax} - \mathbf{y} + \mathbf{b} \\ \mathbf{MWe} - \gamma \mathbf{e} \end{bmatrix} = \mathbf{0}$$

where $\mathbf{M}$ and $\mathbf{W}$ are diagonal matrices with $\mathbf{M}_{ii} = \mathbf{x}_i$ and $\mathbf{W}_{ii} = \mathbf{y}_i$ and $\mathbf{e}$ is the nD vector of ones. In the k^{th} iteration we pick γ to be

$$\gamma^k = \sigma \frac{\left(\mathbf{x}^k\right)^T \left(\mathbf{y}^k\right)}{n}$$

where n is the dimenstion of the problem and the relaxation (centering) parameter is $0 < \sigma < 1$.

Using an Iterative Method

In the k^{th} iteration of our framework we solve $\mathbf{F}(\mathbf{x}, \mathbf{y} : \gamma^k) = \mathbf{0}$ approximately. The solutions of the Kojima map for all positive values of γ define the central path which is a trajectory that leads to the solution of the LCP as γ tends to zero.
Using Taylor series expansion we define the Newton equation

$$\begin{bmatrix} \mathbf{A} & -\mathbf{I} \\ \mathbf{W}^k & \mathbf{M}^k \end{bmatrix} \begin{bmatrix} \Delta\mathbf{x}^k \\ \Delta\mathbf{y}^k \end{bmatrix} = \begin{bmatrix} -\mathbf{A}\mathbf{x}^k + \mathbf{y}^k - \mathbf{b} \\ -\mathbf{M}^k\mathbf{W}^k\mathbf{e} + \gamma\mathbf{e} \end{bmatrix}$$

Having solved for the Newton direction $(\Delta\mathbf{x}^k, \Delta\mathbf{y}^k)$ we do the Newton update

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \tau^k \Delta\mathbf{x}^k$$
$$\mathbf{y}^{k+1} = \mathbf{y}^k + \tau^k \Delta\mathbf{y}^k$$

where τ^k is the step length.

The Step Length

The step length is chosen as the maximum positive value such that

$$\mathbf{x}_i^{k+1} > 0 \quad \text{and} \quad \mathbf{y}_i^{k+1} > 0 \quad \text{for all } i \quad (**)$$

That is we choose $\tau^k = \min(\tau^x, \tau^y)$ where

$$\tau^x = \max\left\{0 < \tau < 1 \mid \mathbf{x}^k + \tau\Delta\mathbf{x}^k \geq (1-\alpha)\mathbf{x}^k\right\}$$
$$\tau^y = \max\left\{0 < \tau < 1 \mid \mathbf{y}^k + \tau\Delta\mathbf{y}^k \geq (1-\alpha)\mathbf{y}^k\right\}$$

Here $0 < \alpha < 1$ controls how far we back away from the maximum step for which the conditions (**) are satisfied.
To accelerate the convergence we make α approach one in a linear fashion as the iterates approach the solution.

Further Reading

- S. Niebe and K. Erleben: Numerical Methods for Linear Complementarity Problems in Physics-Based Animation. Synthesis Lectures on Computer Graphics and Animation, January 2015, Vol. 7, No. 1 , Pages 1-159.
- K. G. Murty: Linear complementarity, linear and nonlinear programming. Sigma Series in Applied Mathematics. 3. Berlin: Heldermann Verlag, 1988. (Chapter 1 and 9)
- R. W. Cottle, J.-S. Pang, R. E. Stone. The Linear Complementarity Problem. Academic Press. 1992. (Chapter 1 and 5)
- D.E. Stewart and J.C. Trinkle. An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and coulomb friction. International Journal of Numerical Methods in Engineering, 39:2673-2691

More Reading

- D.E. Stewart and J.C. Trinkle. Dynamics, friction, and complementarity problems. In M.C. Ferris and J.S. Pang, editors, Complementarity and Variational Problems, pages 425-439. SIAM, 1997.
- D.E. Stewart and J.C. Trinkle. An implicit time-stepping scheme for rigid body dynamics with coulomb friction. In Proceedings, IEEE International Conference on Robots and Automation, pages 162-169, 2000.
- M. Silcowitz, S. Niebe, and K. Erleben. Nonsmooth newton method for fischer function reformulation of contact force problems for interactive rigid body simulation. In Proceedings of VRIPHYS, 2009.
- M. Silcowitz, S. Niebe, and K. Erleben. A nonsmooth nonlinear conjugate gradient method for interactive contact force problems. The Visual Computer, 2010.

Assignment

Derive an algebraic equation of the PGS method on Page 26 for the i^{th} component only. That is rewrite

$$\mathbf{x}_i^{k+1} = \max(0, \mathbf{z}_i^k)$$

using $\mathbf{r} = \mathbf{A}\mathbf{x}^k + \mathbf{b}$ and letting

$$\begin{aligned} \mathbf{M} &= \mathbf{diag}(\mathbf{diag}(\mathbf{A})) + \mathbf{tril}(\mathbf{A}, -1) \\ \mathbf{N} &= -\mathbf{triu}(\mathbf{A}, 1) \end{aligned}$$

Here we used Matlab-like notation. The final result should only include the terms $\mathbf{r}_i$, $\mathbf{A}_{ii}$ and $\mathbf{x}_i$.

Assignment

- Show that both derivations from splitting method on Page 26 and from QP reformulation on Page 33, have algebraic equivalent update formulas for $\mathbf{x}_i$.
- What properties should the $\mathbf{A}$ have for splitting derivation to work?
- What properties should the $\mathbf{A}$ have for QP derivation to work?
- What properties should the $\mathbf{A}$ have for PGS to converge?
- Speculate what to do if $\mathbf{A}_{ii} \ngtr 0$ (Hint: look at how to minimize $g(\tau)$)?

Assignment

As a challenge if you have time. Derive a Nonsmooth Newton method for the minimum map reformulation of the LCP

$$\min(\mathbf{x}, \mathbf{y}) = \mathbf{0}$$

Hint: Look at

> *Sheldon Andrews, Kenny Erleben, and Zachary Ferguson. 2022. Contact and friction simulation for computer graphics. In ACM SIGGRAPH 2022 Courses (SIGGRAPH '22). Association for Computing Machinery, New York, NY, USA, Article 3, 1–172.* https://doi.org/10.1145/3532720.3535640

Assignment

- Examine the Stewart–Trinkle (ST) LCP formulation of the contact force problem.
- What matrix properties can you identify?
- What do you know about the right hand side vector?
- What kind of reformulations are applicable to the ST LCP formulation?
- What kind of methods can be used to solve a ST LCP formulation problem?

Assignment - Basic Programming

- Obtain Lemke's method from CPNET, `http://www.cs.wisc.edu/cpnet/`.
- Create a routine that can generate random N-dimensional LCP problems.
- Generate a sequence of random LCP test problems with an increasing number of variables, $N = 2^k, k = 2, 3, \ldots, 10$.
- Use Lemke to solve random sequences of LCP test problems (run 10 sequences at least).
- Make plots of computing time as a function of the number of variables and make a histogram showing the fraction of solved problems.
- Discuss your results – what do you think of Lemke's method?

Assignment - Intermediate Programming

- Try to implement a PGS solver, a Fischer–Newton solver, and use a QP reformulation solved by using the Matlab **quadprog** function.
- Create a sequence of random problems of increasing size and use the solvers to find solutions.
  - Compare the accuracy of each iterative solver with the true solution, what is the error
  - Plot how the error of each solver behaves as a function of the number of iterations (Hint: make a log plot and determine convergence rate)
  - Try to measure the computing cost per iteration of each solver as a the function of increasing variables (Hint: compare plot with your complexity analysis)
- Based on your experiments evaluate if your implementations behave as expected, speculate for what purposes you want to use different solvers for.

Assignment – Advanced Programming

Create a simple 2D rigid body simulator using spherical objects of varying size and mass. Ignore friction and use simple first-order time stepping. In each simulation step solve an LCP for the normal penetration constraints.

- Determine the eigenvalue spectrum of the coefficient-matrix
- Determine if the coefficient matrix is symmetric or not
- Try experimenting with using different LCP solvers (take those from the intermediate programming exercise)
- Which solver do you think is best for this simulator and why?
- If you have more time try to add friction to the contact forces and rerun all your tests. Did this change on your conclusion on which solver is the best?