Continuous Optimization: Recent Developments and Applications

Stephen Wright

Department of Computer Sciences
University of Wisconsin-Madison

SIAM, Portland, July, 2004

Nonlinear Programming

- Background
- Filter Methods
- Interior-Point Methods
- Degenerate NLP
- MPECs: Equibibrium Constraints

Computation, Applications, Software

- Optimization on the Grid
- New Applications
- Software Tools
- Optimization Modeling Languages
- The Internet

Discuss developments within past 10 years (though the origins go further back in some instances).

Nonlinear Programming: Themes

- Nonlinear Programming (NLP) continues to provide fertile ground for research into algorithms and software.
  - New ideas;
  - Continual reevaluation of old ideas.
- NLP is a broad paradigm encompassing many pathological situations.
  - Difficult to design robust algorithms;
  - Relative performance of algorithms/software varies widely between problems.
- Applications are becoming more plentiful and diverse, and are driving developments in algorithms and software.

Background

Focus mainly on the inequality constrained problem.

$$\min f(x) \text{ subject to } c(x) \geq 0,$$

where $f : \mathbb{R}^n \to \mathbb{R}$ and $c : \mathbb{R}^n \to \mathbb{R}^m$ are assumed smooth.

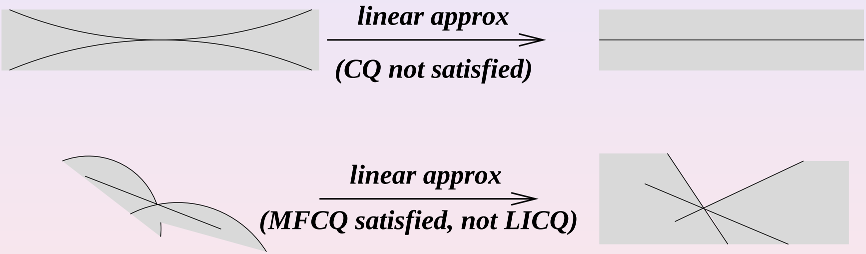
- Algorithms usually find local solutions—not global);
- May converge to less desirable points.

Constraint Qualifications

Since NLP algorithms work with approximations based on the algebraic representation of the feasible set, need conditions to ensure that such approximations capture the essential geometry of the set: **Constraint Qualifications.**

$$\text{Define} \quad \mathcal{A}^* \stackrel{\text{def}}{=} \{i = 1, 2, \dots, m \,|\, c_i(x^*) = 0\}.$$

- Linear Independence (LICQ): The gradients $\nabla c_i(x^*)$, $i \in \mathcal{A}^*$ are linearly independent.
- Mangasarian-Fromovitz (MFCQ): There is a direction d such that $\nabla c_i(x^*)^T d > 0$ for all $i \in \mathcal{A}^*$.


linear approx
(CQ not satisfied)
linear approx
(MFCQ satisfied, not LICQ)

First-Order Necessary Conditions

Karush-Kuhn Tucker (KKT): If x^* is a local minimizer and a CQ is satisfied at x^*, there is $\lambda^* \in \mathbb{R}^m$ such that

$$\begin{aligned} \nabla f(x^*) &= \sum_{i=1}^{m} \lambda_i^* \nabla c_i(x^*), \\ 0 \leq c(x^*) &\perp \lambda^* \geq 0. \end{aligned}$$

Complementarity $\perp$ means that $(\lambda^*)^T c(x^*) = 0$, that is,

$$\lambda_i^* = 0 \text{ or } c_i(x^*) = 0 \text{ for all } i = 1, 2, \ldots, m.$$

"Objective gradient is a conic combination of active constraint gradients."

Optimality conditions can be expressed in terms of the Lagrangian function, which is a linear combination of objectives and constraints, with Lagrange multipliers as coefficients:

$$\mathcal{L}(x, \lambda) = f(x) - \lambda^T c(x).$$

Can write KKT conditions as

$$\begin{aligned} \nabla_x \mathcal{L}(x^*, \lambda^*) &= 0, \\ 0 \le c(x^*) &\perp \lambda^* \ge 0. \end{aligned}$$

Second-Order Conditions

There is a *critical cone* of feasible directions $\mathcal{C}$ for which first derivative information alone cannot determine whether f increases or decreases.

We idenfity "tiebreaking" conditions based on *second* derivatives to ensure increase in f along these directions.

Second-Order Sufficient: If x^* satisfies KKT for some λ^*, and

$$d^T \nabla^2_{xx} \mathcal{L}(x^*, \lambda^*) d > 0, \quad \text{for all } d \in \mathcal{C} \text{ with } d \neq 0,$$

then x^* is a strict local solution.

A Basic Algorithm: SQP for Inequality Constraints

No "outer loop." From current iterate x, obtain step Δx by solving:

$$\min \quad \nabla f(x)^T \Delta x + \tfrac{1}{2}\Delta x^T H \Delta x \quad \text{subject to}$$
$$c(x) + \nabla c(x)^T \Delta x \geq 0,$$

H typically contains curvature information about both objective and active constraints; "ideal" choice is Hessian of the Lagrangian:

$$\nabla^2_{xx}\mathcal{L}(x,\lambda) = \nabla^2 f(x) - \sum_i \lambda_i \nabla^2 c_i(x).$$

- H can also be approximated using a quasi-Newton technique.
- Δx modified via line-search and trust-region variants.
- Merit function often used to determine acceptability of step.

SQP for Equality Constraints

If all constraints are equalities:

$$\min f(x) \text{ subject to } h(x) = 0, \tag{1}$$

Get SQP step by solving

$$\begin{aligned} \min \quad & \nabla f(x)^T \Delta x + \tfrac{1}{2} \Delta x^T H \Delta x \quad \text{subject to} \\ & h(x) + \nabla h(x)^T \Delta x = 0, \end{aligned}$$

When $H = \nabla^2_{xx} \bar{\mathcal{L}}(x, \mu)$ and second-order conditions are satisfied, the same Δx can be obtained by applying Newton's method for algebraic equations to the KKT conditions for (1), which are

$$\begin{aligned} \nabla_x \bar{\mathcal{L}}(x, \mu) = \nabla f(x) - \nabla h(x) \mu &= 0, \\ h(x) &= 0. \end{aligned}$$

1. Filter Methods: Background

Many NLP algorithms use a *merit function* to gauge progress and to decide whether to accept a candidate step. Usually a weighted average of objective value and constraint violation, e.g.

$$\begin{aligned} P(x;\nu) &= f(x) + \nu r(c(x)), \quad \text{where} \\ r(c(x)) &\stackrel{\text{def}}{=} \sum_{i=1}^{m} \max(0, -c_i(x)) = \| \max(0, -c(x)) \|_1. \end{aligned}$$

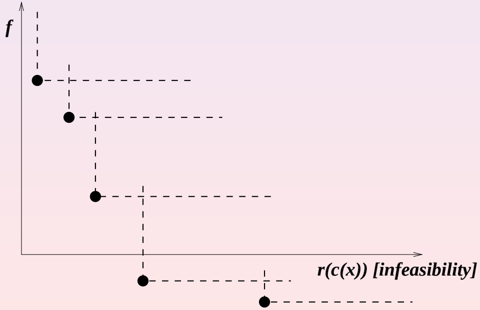
Choosing the penalty parameter ν may be tricky.
Too small: $P(x;\nu)$ unbounded below;
Too large: Slow progress along boundary of Ω.

Seek a "minimalist" approach that doesn't require choice of a parameter, and rejects good steps less often than does P.

(Fletcher & Leyffer, 1997): Filter methods treat NLP as a two-objective optimization problem:

- minimize f;
- minimize $r(c(x)) = \| \max(0, -c(x)) \|_1$ (but we need a *zero* min value).

Construct a filter from $f(x^k)$ and $r(c(x^k))$ values visited, e.g.



Filter-SQP

Subproblem is a trust-region SQP:

$$\min \quad \nabla f(x)^T \Delta x + \tfrac{1}{2}\Delta x^T H \Delta x \quad \text{subject to}$$
$$c(x) + \nabla c(x)^T \Delta x \geq 0, \qquad \|\Delta x\|_\infty \leq \rho.$$

ρ is radius of trust region.

Includes a "restoration" phase to recover a (nearly) feasible point (minimize $r(c(x))$, ignoring $f(x)$) if the QP is infeasible.

Many enhancements added to yield good theoretical behavior, and to deal effectively with some difficult problems.

- Second-order correction: Accounts better for curvature in constraints by solving

$$\min \quad \nabla f(x)^T \Delta x + \tfrac{1}{2}\Delta x^T H \Delta x \quad \text{subject to}$$
$$c(x + \widehat{\Delta x}) + \nabla c(x)^T(\Delta x - \widehat{\Delta x}) \geq 0, \quad \|\Delta x\|_\infty \leq \rho,$$

  where $\widehat{\Delta x}$ is the previous guess of Δx.
- Use Lagrangian in place of f, allows rapid local convergence.
- Sufficient decrease conditions: Accept a candidate only if a significant improvement over current filter.
- Allow for non-global and inexact solution of the subproblem.
- Others...

Following original proposal by Fletcher and Leyffer, contributors to filter-SQP theory have included Gould, M. Ulbrich, S. Ulbrich, Toint, Wächter, others.

The Filter approach is also used in conjunction with interior-point methods (see below).

Recent codes using filters:

- FilterSQP (Fletcher and Leyffer; SQP/filter)
- IPOPT (Wächter and Biegler; interior-point/filter).

2. Interior-Point Methods: Background

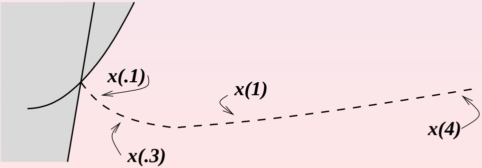
Logarithmic barrier: Choose parameter μ; $x(\mu)$ solves

$$B_x(\mu): \min_x P(x;\mu) \stackrel{\text{def}}{=} f(x) - \mu \sum_{i=1}^{m} \log c_i(x) \qquad (c(x) > 0).$$

Under certain assumptions, have $x(\mu) \to x^*$ as $\mu \to 0$.

Optimality condition:

$$\nabla f(x) - \sum_{i=1}^{m} \frac{\mu}{c_i(x)} \nabla c_i(x) = 0.$$



Barrier-Unconstrained

$k \leftarrow 1$; set $\mu_0 > 0$; set $\bar{x}^0$
repeat
  Solve $B_x(\mu_k)$ for $x(\mu_k)$, starting from $\bar{x}^{k-1}$;
  Choose $\mu_{k+1} \in (0, \mu_k)$;
  Choose next starting point $\bar{x}^k$;
  $k \leftarrow k+1$;
until convergence.

Newton steps obtained from:

$$\nabla^2_{xx} P(x;\mu)\Delta x = -\nabla_x P(x;\mu).$$

Primal barrier was one of the original methods for NLP (Frisch, '55; Fiacco & McCormick, '68). Fell into disuse for various reasons, including severe ill-conditioning of the Hessian $\nabla^2_{xx} P(x;\mu)$.
(Many difficulties later turned out to be fixable.)

Barrier-EQP

Introduce a slack variable w:

$$\min f(x) \text{ subject to } c(x) - w = 0, \ (w > 0).$$

Obtain $x(\mu)$ by solving an equality constrained problem:

$$B_w(\mu) : \min f(x) - \mu \sum_{i=1}^{m} \log w_i, \ \text{ subject to } c(x) - w = 0, \ (w > 0).$$

$B_x(\mu)$ and $B_w(\mu)$ have identical solutions, but obtained differently:

- unconstrained algorithms are applied to $B_x(\mu)$
- equality-constrained algorithms for $B_w(\mu)$ (don't maintain feasibility of $c(x) - w = 0$ at every iteration).

Recent successful codes take their theoretical underpinnings from solving $B_w(\mu_k)$ for a sequence of $\mu_k \downarrow 0$.

Applying SQP to $B_w(\mu)$

KKT conditions for $B_w(\mu)$:

$$\begin{aligned} \nabla f(x) - \nabla c(x)\lambda &= 0, \\ c(x) - w &= 0, \\ -\mu W^{-1} e + \lambda &= 0, \quad (\lambda > 0,\ w > 0), \end{aligned}$$

where $W = \mathrm{diag}(w_1, w_2, \dots, w_m)$, $e = (1, 1, \dots, 1)^T$.

Get pure SQP step by applying Newton's method to KKT conditions:

$$\begin{bmatrix} \nabla^2 \mathcal{L}(x,\lambda) & -\nabla c(x) & 0 \\ \nabla c(x)^T & 0 & -I \\ 0 & I & \mu W^{-2} \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta \lambda \\ \Delta w \end{bmatrix} = - \begin{bmatrix} \nabla_x \mathcal{L}(x,\lambda) \\ c(x) - w \\ -\mu W^{-1} e + \lambda \end{bmatrix}.$$

Eliminate Δw to get a more tractable form:

$$\begin{bmatrix} \nabla^2 \mathcal{L}(x,\lambda) & -\nabla c(x) \\ \nabla c(x)^T & \mu^{-1} W^2 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta \lambda \end{bmatrix} = - \begin{bmatrix} \nabla_x \mathcal{L}(x,\lambda) \\ r(x,w,\lambda,\mu) \end{bmatrix}.$$

(Byrd, Gilbert, Nocedal, '00): Apply a customized SQP-trust region algorithm to $B_w(\mu)$,

(Wächter-Biegler, '02): Apply a filter-SQP method to $B_w(\mu)$, with

- objective $f(x) - \mu \sum \log w_i$
- infeasibility measure $\|c(x) - w\|_2$

However, the theory isn't the whole story! Explicit manipulation of dual variables λ is key to making these algorithms effective in practice.

Primal-Dual Approach

Transform the last KKT condition for $B_w(\mu)$ by multiplying by W; *then* apply Newton-like method for algebraic equations to

$$
\begin{aligned}
\nabla f(x) - \nabla c(x)\lambda &= 0, \\
c(x) - w &= 0, \\
W\lambda - \mu e &= 0, \quad (\lambda > 0,\ w > 0),
\end{aligned}
$$

Choose search directions and steplengths to

- maintain $\lambda > 0$, $w > 0$;
- make progress toward the solution (according to certain measures).

Newton equations for primal-dual step, after elimination of Δw:

$$\begin{bmatrix} \nabla^2 \mathcal{L}(x,\lambda) & -\nabla c(x) \\ \nabla c(x)^T & \Lambda^{-1} W \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta \lambda \end{bmatrix} = - \begin{bmatrix} \nabla_x \mathcal{L}(x,\lambda) \\ r(x,w,\lambda,\mu) \end{bmatrix}.$$

Take steps jointly in (x, w, λ); maintain positivity of w and λ. Choose α_x, α_w, α_λ in $(0,1)$ (not necessarily the same!) and set

$$\begin{aligned} x &\leftarrow x + \alpha_x \Delta x, \\ w &\leftarrow w + \alpha_w \Delta w > 0, \\ \lambda &\leftarrow \lambda + \alpha_\lambda \Delta \lambda > 0. \end{aligned}$$

Primal-Dual Discussion

Primal-dual approaches for linear programming were well known by '87 and were the practical approach of choice by '90.

Extension to NLP obvious in principle, but the devil was in the details! Many enhancements were needed to improve practical robustness and efficiency.

Codes LOQO (Benson, Shanno, Vanderbei), KNITRO (Byrd et al.) and IPOPT (Wächter & Biegler) all generate steps of primal-dual type but differ in important respects.

KNITRO and IPOPT both derive their theoretical justification from the Barrier-EQP approach, but replace $\mu^{-1}W^2$ in the reduced system by $\Lambda^{-1}W$. (IPOPT checks explicitly that $\Lambda^{-1}W$ does not stray too far from $\mu^{-1}W^2$.)

Software

IPOPT (Wächter, Biegler, '04): Line-search filter method based on $B_w(\mu)$, but with primal-dual steps.

Solve step equations using direct solver (MA27) for symmetric-indefinite systems, adding perturbations to the diagonal to "correct" the inertia if necessary.

LOQO uses direct factorization of reduced step equations, with additions to $(1,1)$ block to fix the inertia.

KNITRO Trust-region algorithm for $B_w(\mu)$, with "primal-dual" scaling $(\Lambda^{-1}W)$. Merit function.

Solve step equations using an iterative method: projected conjugate gradient.

KNITRO-Direct: Also uses direct solution of step equations and a line search on some iterations.

3. Degenerate NLP

Reminder of optimality conditions

$$\begin{aligned} \nabla f(x^*) &= \sum_{i=1}^{m} \lambda_i^* \nabla c_i(x^*), \\ 0 \le c(x^*) &\perp \lambda^* \ge 0. \end{aligned}$$

Given the active constraints

$$\mathcal{A}^* \stackrel{\text{def}}{=} \{i = 1, 2, \dots, m \,|\, c_i(x^*) = 0\},$$

we can rewrite optimality conditions as

$$\begin{aligned} \nabla f(x^*) - \sum_{i \in \mathcal{A}^*} \lambda_i^* \nabla c_i(x^*) &= 0, \\ c_i(x^*) &= 0, \quad \text{for all } i \in \mathcal{A}^* \end{aligned}$$

(since by complementarity have $\lambda_i^* = 0$ for $i \notin \mathcal{A}^*$).

Local convergence anaylsis of algorithms usually assumes

- Linear independence of active constraints: $\nabla c_i(x^*)$, $i \in \mathcal{A}^*$ linearly independent (implies that λ^* is unique).
- Strict complementarity: $\lambda_i^* > 0$ for all $i \in \mathcal{A}^*$.
- Second-order sufficient conditions.

Under these assumptions, the system

$$\begin{aligned} \nabla f(x^*) - \sum_{i \in \mathcal{A}^*} \lambda_i^* \nabla c_i(x^*) &= 0, \\ c_i(x^*) &= 0, \quad \text{for all } i \in \mathcal{A}^* \end{aligned}$$

is a "square" nonlinear algebraic system with whose Jacobian

$$\begin{bmatrix} \nabla^2 \mathcal{L}(x^*, \lambda^*) & -\nabla c_{\mathcal{A}^*}(x) \\ \nabla c_{\mathcal{A}^*}(x)^T & 0 \end{bmatrix}$$

is nonsingular at the solution $(x^*, \lambda^*_{\mathcal{A}^*})$.

Hence, for the inequality problem, can get rapid convergence by

- Identifying $\mathcal{A}^*$ correctly;
- Applying Newton's method to the reduced system above.

(Many algorithms do this, implicitly or explicitly.)

The three assumptions are rather strong. *Degenerate* problems are those for which they are not satisfied.

- $\mathcal{A}^*$ hard to identify;
- Jacobian of Newton system is singular in the limit.

We discuss an approach that obtains fast convergence without linear independence and strict complementarity (but still needs second-order sufficient conditions).

Denote $\mathcal{S} = \{(x^*, \lambda^*) \text{ satisfying KKT conditions}\}$.

Under our assumptions, x^* is unique but λ^* may not be. In fact, the set of optimal λ^* may be unbounded.

First Ingredient: Active Set Identification

First issue is to identify $\mathcal{A}^*$ correctly.

Assume that estimates of both x and λ are available.

The following defines a computable *estimate of the distance to solution set* $\mathcal{S}$.

$$\eta(x,\lambda) = \left\| \begin{bmatrix} \nabla_x \mathcal{L}(x,\lambda) \\ \min(\lambda, c(x)) \end{bmatrix} \right\|_1 .$$

Can show that for (x,λ) near $\mathcal{S}$, we have

$$\eta(x,\lambda) \sim \mathrm{dist}((x,\lambda),\mathcal{S}).$$

Active set estimate: Choose $\sigma \in (0,1)$ (say $\sigma = 1/2$) and set

$$\mathcal{A}(x,\lambda) = \{i \,|\, c_i(x) \le \eta(x,\lambda)^{1/2}\}.$$

For (x,λ) sufficiently close to $\mathcal{S}$, we have

$$\mathcal{A}^* = \mathcal{A}(x,\lambda).$$

Proof. Have $\eta(x,\lambda) \to 0$ as $(x,\lambda) \to \mathcal{S}$.

- For $i \notin \mathcal{A}^*$, have $c_i(x) \to c_i(x^*) > 0$, so eventually $c_i(x) > \eta(x,\lambda)^{1/2}$.
- For $i \in \mathcal{A}^*$, have $c_i(x) = c_i(x) - c_i(x^*) = O(\|x - x^*\|) = O(\eta(x,\lambda)) \le \eta(x,\lambda)^{1/2}$.

QED

Second Ingredient: A Stabilized Method for Equalities

Solve the equality-constrained problem

$$\min f(x) \text{ subject to } c_{\mathcal{A}}(x) = 0.$$

Define distance to solution by

$$\bar{\eta}(x, \lambda_{\mathcal{A}}) = \left\| \begin{bmatrix} \nabla_x \bar{\mathcal{L}}(x, \lambda_{\mathcal{A}}) \\ c_{\mathcal{A}}(x) \end{bmatrix} \right\|$$

and solve a stabilized SQP subproblem:

$$\begin{bmatrix} \nabla^2_{xx} \bar{\mathcal{L}}(x, \lambda_{\mathcal{A}}) & -\nabla c_{\mathcal{A}}(x) \\ \nabla c_{\mathcal{A}}(x)^T & \bar{\eta} I \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta \lambda_{\mathcal{A}} \end{bmatrix} = - \begin{bmatrix} \nabla_x \bar{\mathcal{L}}(x, \lambda_{\mathcal{A}}) \\ h(x) \end{bmatrix}.$$

and set $(x, \lambda_{\mathcal{A}}) \leftarrow (x, \lambda_{\mathcal{A}}) + (\Delta x, \Delta \lambda_{\mathcal{A}})$.

A Complete Algorithm

Use the two ingredients above to build a rapidly convergent local phase into any algorithm that generates primal-dual iterates (x^k, λ^k).

- $\eta(x^k, \lambda^k) < \tau$, enter local phase:
- estimate $\mathcal{A}$ and apply the stabilized algorithm to

$$\min\ f(x)\ \text{ subject to } c_{\mathcal{A}}(x) = 0;$$

- check that new iterate is acceptable for the original inequality constrained problem. If not, set $\tau \leftarrow \tau/2$ and return to outer loop;

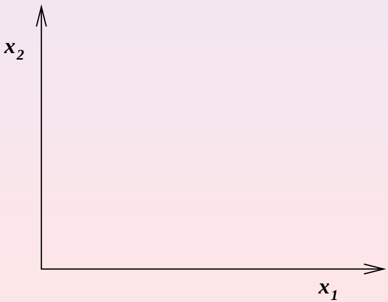
With appropriate checks, get superlinear convergence.

4. Optimization with Equilibrium Constraints (MPEC)

Simplest example: Given x_1 and x_2 in $\mathbb{R}$,

$$\min f(x_1, x_2) \text{ subject to } x_1 \geq 0,\ x_2 \geq 0,\ \ x_1 \perp x_2.$$

Can write complementarity condition as $x_1 x_2 = 0$ or $-x_1 x_2 \geq 0$.



MPEC constraints look simple, but they

- Do not satisfy constraint qualifications at *any* feasible point.
- Are highly nonconvex at $(0,0)$.
- Are a *union* of "nice" feasible regions, rather than an intersection. (Unions are much harder to deal with.)

If $x_1 = 0$, have active constraints $x_1 \geq 0$ and $-x_1 x_2 \geq 0$ with gradients:

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \begin{bmatrix} -x_2 \\ 0 \end{bmatrix},$$

which are linearly dependent.

At $(x_1, x_2) = 0$, all three constraints are active with gradients

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

Application of MPEC: Equilibrium Problems with Design

Systems described by equilibrium conditions (e.g. economic or mechanical), with an optimization or design problem overlaid.

Example: Nash equilibrium in a 2-player game.
Player 1 aims to maximize his some function $f_1(x_1, x_2)$ by choosing "his" variables from the set $x_1 \geq 0$. If $f_1(\cdot, x_2)$ is convex, the optimal choice for a given x_2 satisfies

$$0 \leq x_1 \perp -\nabla_{x_1} f_1(x_1, x_2) \geq 0.$$

Similarly for Player 2 (whose variables are x_2), optimal choice has

$$0 \leq x_2 \perp -\nabla_{x_2} f_2(x_1, x_2) \geq 0.$$

The *Nash equilibrium* is the point (x_1, x_2) at which both sets of conditions are satisfied.

Suppose there's a design variable z that should be chosen to maximize some overall utility, say $g(x_1, x_2, z)$.

Adding z to the utility functions f_1, f_2 and assuming appropriate convexity conditions, we can state the overall MPEC as

$$
\begin{aligned}
&\max_{x_1, x_2, z} \; g(x_1, x_2, z) \text{ subject to} \\
0 \le x_1 \quad &\perp \quad -\nabla_{x_1} f_1(x_1, x_2, z) \ge 0, \\
0 \le x_2 \quad &\perp \quad -\nabla_{x_2} f_2(x_1, x_2, z) \ge 0.
\end{aligned}
$$

(Possibly other constraints on x_1, x_2, z.)

Solving MPECs: 1996-2002

- Lack of CQ $\Rightarrow$ Standard NLP algorithms may not work!
- Surge of interest prompted in part by '96 monograph of Luo, Pang, Ralph.
- Various specialized algorithms proposed ('96-'01) to deal with the special nature of the constraints.

However, Fletcher & Leyffer ('02) showed that ordinary SQP codes performed excellently on MPEC benchmarks (and interior-point codes were also quite good).

Why?

Can robustness of standard NLP approaches be improved further?

MPEC First-Order Optimality Conditions

Consider this formulation (where G and H are functions from $\mathbb{R}^n \to \mathbb{R}^m$):

$$\min f(x) \text{ subject to } 0 \le G(x) \perp H(x) \ge 0.$$

Can include general equality ($h(x) = 0$) and inequality ($c(x) \ge 0$) constraints—they complicate the notation but not the concepts.

Active sets $I_G = \{i \mid G_i(x^*) = 0\}$, $I_H = \{i \mid H_i(x^*) = 0\}$.

Complementarity $\Rightarrow I_G \cup I_H = \{1, 2, \ldots, m\}$.

Stationarity Conditions: Have x^* feasible for the MPEC, and Lagrange multipliers τ_i^*, ν_i^* such that

$$\begin{aligned} \nabla f(x^*) - \sum_{i \in I_G} \tau_i^* \nabla G_i(x^*) - \sum_{i \in I_H} \nu_i^* \nabla H_i(x^*) &= 0, \\ G_i(x^*) &= 0, \quad i \in I_G, \\ H_i(x^*) &= 0, \quad i \in I_H. \end{aligned}$$

For $i \in I_G \backslash I_H$, have $H_i(x) > 0$ for all x near x^*, so complementarity implies $G_i(x) = 0$.

Hence, G_i behaves as an equality constraint and so the multiplier τ_i^* can have either sign.

(Similarly for ν_i^*, $i \in I_H \backslash I_G$.)

However for $i \in I_G \cap I_H$—the **biactive** indices—it is less clear how the sign of τ_i^* and ν_i^* should be restricted. Different flavors of stationarity have been proposed:

- *C-stationary*: τ_i^* and ν_i^* have the same sign;
- *M-stationary*: Does not allow τ_i^* and ν_i^* of different signs, and one of them must be nonnegative.
- *Strong stationarity*: τ_i^* and ν_i^* both nonnegative.

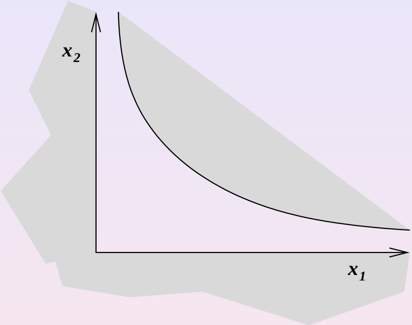
Strong stationarity is the most appealing, as it ensures that there is no first-order direction of descent.

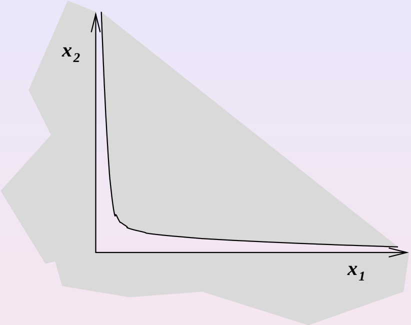
Regularization (Scholtes '01)

For parameter $\sigma > 0$, define

$$\mathrm{Reg}(\sigma): \quad \begin{array}{l} \min\ f(x)\ \text{ subject to} \\ G(x) \geq 0, \quad H(x) \geq 0, \\ G_i(x)H_i(x) \leq \sigma,\ i = 1, 2, \ldots, m, \end{array}$$

- Solve for decreasing sequence of $\sigma = \sigma_k$ with $\sigma_k \downarrow 0$; Can apply a more or less standard NLP algorithm.
- For each $\sigma > 0$, CQ are satisfied (under "reasonable" assumptions).





Relationship between solution $x(\sigma)$ of $\mathrm{Reg}(\sigma)$ and the original MPEC is quite complex. We may have

- $\|x(\sigma) - x^*\| = O(\sigma)$;
- $\|x(\sigma) - x^*\| = O(\sigma^{1/2})$,

depending on exactly what properties hold at a strongly stationary x^* (Scholtes '01, Ralph and SW '04).

Penalization

For parameter $\rho > 0$ define

$$\text{Pen}(\rho): \quad \begin{array}{l} \min\ f(x) + \rho G(x)^T H(x) \ \text{ subject to} \\ G(x) \geq 0, \quad H(x) \geq 0. \end{array}$$

- Force $G(x)^T H(x) = 0$ by penalizing positive values of $G^T H$.
- Solve Pen(ρ_k) for an increasing sequence of ρ_k.
- STOP when $G^T H = 0$ at some ρ_k; the solution of Pen(ρ_k) is then at least stationary for the MPEC.
- CQ are satisfied (under "reasonable" assumptions).
- Good computational success reported for interior-point methods applied to this formulation.

Some Results for Penalization

(Anitescu '01; Ralph and SW '04; Anitescu, Tseng, SW, '04)

If x^* solves the MPEC, then it is also a local solution of Pen(ρ) for all ρ sufficiently large.

If x^* satisfies KKT conditions for Pen(ρ) and $G(x^*)^T H(x^*) = 0$, then x^* is strongly stationary for the MPEC.

(Global Convergence) If second-order necessary conditions are satisfied at each solution of Pen(ρ_k) then either

- for some k, solution of Pen(ρ_k) is a strongly stationary point of the MPEC; or
- any accumulation point is infeasible or does not satisfy the LICQ

SQP for MPEC

- In Fletcher-Leyffer tests ('02), the SQP codes `filterSQP` and `SNOPT` solved almost all test problems.
- Subsequent analysis (Fletcher et al. '02) provided some theoretical support for local convergence of SQP.

MPEC is a prime candidate for the *stabilized SQP approach for degenerate NLP* outlined above.

- No longer need to *assume* feasibility of SQP subproblems;
- Can be embedded in other approaches to produce rapid local convergence;
- However, needs second-order conditions that are stronger than the ideal.

Computation, Applications, Software

- Large optimization problems on computational grids
- Applications
- Software packages; Trends in software design
- Modeling languages
- The Internet

Optimization on the Grid

The Grid isn't all hype! Optimizers have done serious computations on it. Some examples:

- Traveling Salesman Problem: Applegate et al. Pre-"Grid"
- metaNEOS ('97-'01): NSF-funded collaboration between optimizers and grid computing groups: Condor, Globus.
- MW Toolkit for implementing master-worker algorithms on grids
  - Many interesting parallel algorithmic paradigms can be shoehorned effectively into the master-worker framework (task-farming, branch-and-bound, "native" master-worker)
  - Development of MW is continuing.

Successful Implementations with MW

- Quadratic Assignment Problem: Solved the touchstone problem NUG30 (Anstreicher et al., '02). Seven years of CPU time in a week on more than 1000 computers (10 locations, 3 continents).
- Two-stage stochastic linear programming (Linderoth & SW, '03). Solved problems with up to 10^8 constraints and 10^{10} variables on similar platforms. Allowed extensive testing of statistical properties of solutions to such problems and insight into the nature of solutions (Linderoth, Shapiro, SW, '02).

"New" Applications

The NLP user base is smaller than for linear programming or integer (linear) programming, though there has been steady demand for codes such as MINOS, CONOPT, NPSOL for many years.

NLP have arisen (or have attracted renewed attention) in a host of applications in the past 10-15 years.

Optimization with PDE constraints. An area of particular focus. Typically design or control problems overlaid on a system described by PDEs. (see volume of Biegler et al., Springer LNCS, '03)

Cancer Treatment Planning. Find radiotherapy treatments for cancer that deposit a given dose of radiation in the tumor while avoiding nearby normal tissue and critical organs. Involves optimization problems of many types; recent interest in *biological objective functions* give rise to NLPs.

Meteorological Data Assimilation in medium-range weather forecasting. Special case of PDE-constrained optimization. Unknown is the state of atmosphere 48 hours ago; objective function is fit between model and observations. NLP formulations (with specialized algorithms) used at all major forecasting centers.

Model Predictive Control in process engineering. Solve a sequence of nonlinear optimal control problem over finite time horizons; implement closed-loop control using open-loop strategies. Feasible SQP methods have proved to be useful.

Statistics and Machine Learning. Multicategory support vector machines, robust estimation.

Image Reconstruction. Reconstructing images from undersampled MRI data.

Others were presented at this meeting.

Connecting with Users

Optimization is a consumer-oriented discipline with a wide and diverse user base.

- Science
- Engineering
- Operations Research
- Finance and Economics
- "Retail" Users

How is optimization technology transferred to the market? What are the trends?

NLP Software Packages

See NEOS Server `www-neos.mcs.anl.gov`

*=freely available. Red=under active development.

- IPOPT*: interior-point
- KNITRO: interior-point and SLQP variants
- SNOPT: SQP
- LOQO: interior-point
- FILTER: Filter-SQP
- CONOPT: Generalized reduced gradient
- LANCELOT*: Augmented Lagrangian
- MINOS: sequential linearly constrained
- PENNON: Generalized augmented Lagrangian

Other NLP solvers are embedded into larger optimization or numerical computing systems: Matlab, Excel, NAG.

Numerous other commercial packages reviewed in 1998 OR/MS Today survey:
`www.lionhrtpub.com/orms/surveys/nlp/nlp.html`.

NEOS Guide: `www.mcs.anl.gov/otc/Guide/` (somewhat out of date).

Trends in Optimization Software Design

- Once were monolithic Fortran codes. Fortran still popular; C now used in some leading codes; a few experiments in C++.
- Object-oriented design used in some recent codes; allow specialization according to problem structure, modular use of linear algebra:
  - TAO (unconstrained and bound-constrained)
  - OOPS and OOQP (quadratic programming)
  - IOTR (NLP-forthcoming)
- Some interior-point codes are especially able to use off-the-shelf linear algebra packages (e.g. IPOPT and OOQP use Harwell's MA27).
- Multiple user interfaces: native-language calls, batch files, modeling languages, high-level languages (Matlab, Octave), spreadsheets.

Open-Source Software

Some good codes are now open-source.

The COIN-OR initiative `www.coin-or.org` has was founded in 2000 to organize and support community development of optimization codes.

- Peer review process;
- Sessions at optimization meetings;
- OSI: common interface to optimization solvers (currently for LP and MIP);
- Includes several excellent codes, e.g. IPOPT for NLP.

Optimization Modeling Languages

Allows problems to be defined (and the results presented) in a way that is natural to the application. No shoehorning into the matrices/vectors of the underlying software.

AMPL, GAMS, Maximal, AIMMS

They have become an extremely popular way to call optimization software (method of choice on NEOS Server).

Usually incorporate automatic differentiation, relieves user of need to code first derivatives by hand. (Several codes optionally allow second derivatives to be used; these must be provided by the user.)

They make optimzation the "outer loop"—not easy to embed the optimization problem in a larger computation.

The Internet

Using the Internet to reach users and build community.

- Following on from pioneers in the NA community: NAnet, netlib (mid-'80s).
- NEOS ('94-present)
  - Server: online solution of optimization problems; now 1500-3000 submissions/week.
  - Guide: informational resource; not much recent development
- Preprint repositories:
  - IPMO: Interior-Point Methods Online ('94-'01) was vital to the interior-point boom.
  - Optimization Online ('00-present)

THANKS FOR YOUR ATTENTION!