A globally convergent primal-dual interior-point filter method for nonlinear programming (`ipfilter`):

New filter optimality measures and computational results

Luis Nunes Vicente

www.mat.uc.pt/~lnv

Dept. Mathematics, Univ. Coimbra

· U C ·

OPTEC, K.U. Leuven, December 2, 2008

Outline

Co-Authors

Joint work with M. Ulbrich (TU München), S. Ulbrich (TU Darmstadt) and R. Silva (Coimbra).

Algorithm development and theoretical support:

- M. ULBRICH, S. ULBRICH, AND L. N. VICENTE, *A globally convergent primal-dual interior-point filter method for nonlinear programming*, Mathematical Programming, 100 (2004) 379-410.
- R. SILVA, M. ULBRICH, S. ULBRICH, AND L. N. VICENTE, *A globally convergent primal-dual interior-point filter method for nonlinear programming: New filter optimality measures and computational results*, submitted for publication.

Co-Authors

Joint work with M. Ulbrich (TU München), S. Ulbrich (TU Darmstadt) and R. Silva (Coimbra).

Algorithm development and theoretical support:

- M. ULBRICH, S. ULBRICH, AND L. N. VICENTE, *A globally convergent primal-dual interior-point filter method for nonlinear programming*, Mathematical Programming, 100 (2004) 379-410.
- R. SILVA, M. ULBRICH, S. ULBRICH, AND L. N. VICENTE, *A globally convergent primal-dual interior-point filter method for nonlinear programming: New filter optimality measures and computational results*, submitted for publication.

The goal is to determine first-order or second-order stationary points.

Newton or quasi-Newton step calculations yield good local behavior (quadratic/superlinear rates of convergence).

NLP algorithms must also converge globally.

NLP Algorithms

The goal is to determine first-order or second-order stationary points.

Newton or quasi-Newton step calculations yield good local behavior (quadratic/superlinear rates of convergence).

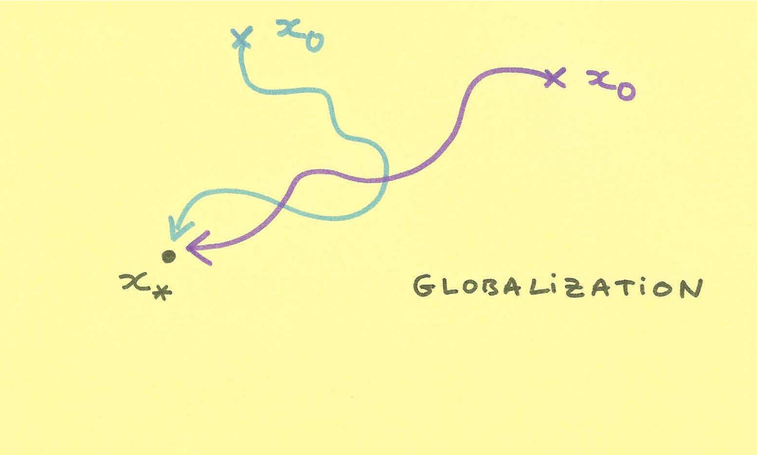
NLP algorithms must also converge globally.

NLP Algorithms

The goal is to determine first-order or second-order stationary points.

Newton or quasi-Newton step calculations yield good local behavior (quadratic/superlinear rates of convergence).

NLP algorithms must also converge globally.



x_0
x_0
x_*
GLOBALIZATION

NLP Globalization Strategies

There are two globalization techniques:

- line searches (search along the step direction).
- trust regions (step is computed by minimizing a quadratic model within a region of prescribed size).

NLP Globalization Strategies

There are two globalization techniques:

- line searches (search along the step direction).
- trust regions (step is computed by minimizing a quadratic model within a region of prescribed size).

NLP Globalization Strategies

There are two globalization techniques:

- line searches (search along the step direction).
- trust regions (step is computed by minimizing a quadratic model within a region of prescribed size).

NLP Globalization Strategies – Constraints

In the presence of constraints, we must use one of the previous together with:

- merit function (combining objective function with a penalization of constraints).
- filter (explained next).

NLP Globalization Strategies – Constraints

In the presence of constraints, we must use one of the previous together with:

- merit function (combining objective function with a penalization of constraints).
- filter (explained next).

NLP Globalization Strategies – Constraints

In the presence of constraints, we must use one of the previous together with:

- merit function (combining objective function with a penalization of constraints).
- filter (explained next).

Filter Scheme

- Globalize without merit functions and penalty parameters.
- Borrow the concept of nondominance from multi-criteria optimization.

$$\min \; f(x) \quad \text{s.t.} \quad h(x) = 0$$

$$\downarrow$$

$$\min \quad f(x) \quad \text{and} \quad \theta(x) = \|h(x)\|$$

(bi-criteria optimization problem)

Filter Scheme

- Globalize without merit functions and penalty parameters.
- Borrow the concept of nondominance from multi-criteria optimization.

$$\min\ f(x) \quad \text{s.t.} \quad h(x) = 0$$

$$\downarrow$$

$$\min \quad f(x) \quad \text{and} \quad \theta(x) = \|h(x)\|$$

(bi-criteria optimization problem)

Filter Scheme

- Globalize without merit functions and penalty parameters.
- Borrow the concept of nondominance from multi-criteria optimization.

$$\min \; f(x) \quad \text{s.t.} \quad h(x) = 0$$

$$\downarrow$$

$$\min \quad f(x) \quad \text{and} \quad \theta(x) = \|h(x)\|$$

(bi-criteria optimization problem)

Filter Scheme – Definitions

A filter is a discrete set of efficient or nondominated points.

A point x or $(f(x), \theta(x))$ is efficient or nondominated if it is not dominated by any other point in the filter.

A point x dominates x' ($x \neq x'$) if

$$\theta(x) \leq \theta(x') \quad \text{and} \quad f(x) \leq f(x').$$

The filter defines the efficient border of the nondominated points.

Filter Scheme – Definitions

A filter is a discrete set of efficient or nondominated points.

A point x or $(f(x), \theta(x))$ is efficient or nondominated if it is not dominated by any other point in the filter.

A point x dominates x' $(x \neq x')$ if

$$\theta(x) \leq \theta(x') \quad \text{and} \quad f(x) \leq f(x').$$

The filter defines the efficient border of the nondominated points.

A filter is a discrete set of efficient or nondominated points.

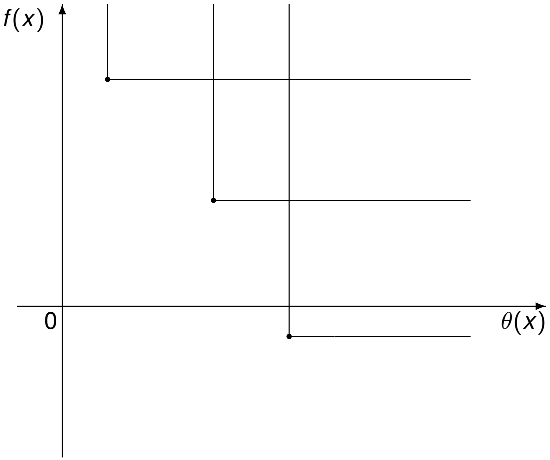
A point x or $(f(x), \theta(x))$ is efficient or nondominated if it is not dominated by any other point in the filter.

A point x dominates x' ($x \neq x'$) if

$$\theta(x) \leq \theta(x') \quad \text{and} \quad f(x) \leq f(x').$$

The filter defines the efficient border of the nondominated points.

Filter Scheme – an Example



Acceptability is defined in a more stringent way (envelope):

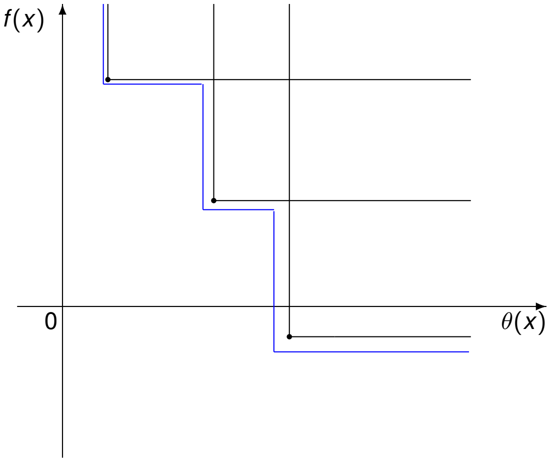
A point x is acceptable to the filter if

$$\theta(x_j) > \theta(x) + \gamma_{\mathcal{F}}\theta(x_j) \quad \text{or} \quad f(x_j) > f(x) + \gamma_{\mathcal{F}}\theta(x_j),$$

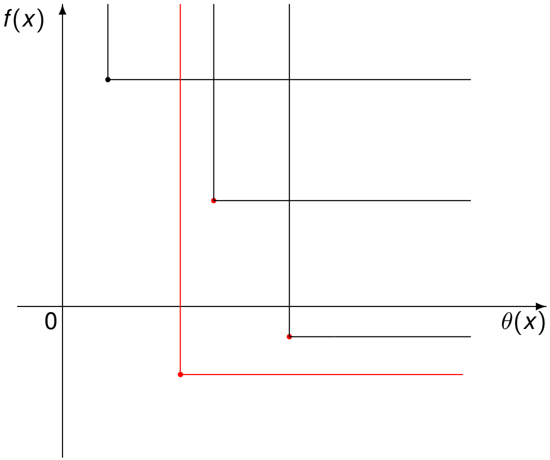
for all filter entries x_j.

$\gamma_{\mathcal{F}} \in (0, \frac{1}{2})$.

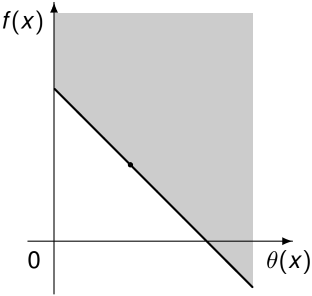
Filter Scheme – an Example

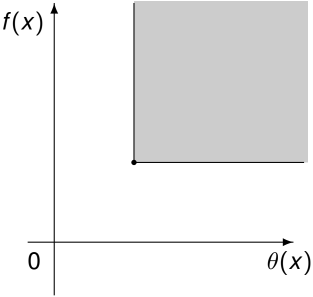


Filter Scheme – an Example



Forbidden Regions – Merit Function vs. Filter





Filter Methods Components

- Definition of filter entries.
- Definition of filter envelope.
- Filter management schemes (including removal of filter entries).
- Restoration phase when feasibility is not sufficiently small.
- Sufficient decrease when feasibility is sufficiently small.

Filter Methods Components

- Definition of filter entries.
- Definition of filter envelope.
- Filter management schemes (including removal of filter entries).
- Restoration phase when feasibility is not sufficiently small.
- Sufficient decrease when feasibility is sufficiently small.

Filter Methods Components

- Definition of filter entries.
- Definition of filter envelope.
- Filter management schemes (including removal of filter entries).
- Restoration phase when feasibility is not sufficiently small.
- Sufficient decrease when feasibility is sufficiently small.

Filter Methods Components

- Definition of filter entries.
- Definition of filter envelope.
- Filter management schemes (including removal of filter entries).
- Restoration phase when feasibility is not sufficiently small.
- Sufficient decrease when feasibility is sufficiently small.

Filter Methods Components

- Definition of filter entries.
- Definition of filter envelope.
- Filter management schemes (including removal of filter entries).
- Restoration phase when feasibility is not sufficiently small.
- Sufficient decrease when feasibility is sufficiently small.

Filter – Literature

- SQP: Fletcher, Leyffer (Math. Prog. 2002)
  Fletcher, Gould, Leyffer, Toint, Wächter (SIOPT 2002)
  Fletcher, Leyffer, Toint (SIOPT 2002)
  S. Ulbrich (Math. Prog. 2004)
- SLP: Fletcher, Leyffer, Toint
  Chin, Fletcher (Math. Prog. 2003)
- IPM: Ulbrich, Ulbrich, Vicente (Math. Prog. 2004)
  Wächter, Biegler (SIOPT 2005)
  Benson, Shanno, Vanderbei (COAP 2002)
- Others:
  Audet, Dennis (SIOPT 2004) – DFO
  Gonzaga, Karas, Vanti (SIOPT 2003)
  Gould, Leyffer, Toint (SIOPT 2004) – systems of nonlinear equations

  nonsmooth optimization, unconstrained optimization...

Lagrange Prize

Roger Fletcher, Sven Leyffer, and Philippe Toint were the recipients in 2006 of the Lagrange Prize for Continuous Optimization, awarded jointly by the Mathematical Programming Society (MPS) and the Society for Industrial and Applied Mathematics (SIAM), for *"outstanding works in the area of continuous optimization"*.

> *"In the development of nonlinear programming over the last decade, an outstanding new idea has been the introduction of the filter. This new approach to balancing feasibility and optimality has been quickly picked up by other researchers, spurring the analysis and development of a number of optimization algorithms in such diverse contexts as constrained and unconstrained nonlinear optimization, solving systems of nonlinear equations, and derivative-free optimization. The generality of the filter idea allows its use, for example, in trust region and line search methods, as well as in active set and interior point frameworks. Currently, some of the most effective nonlinear optimization codes are based on filter methods. The importance of the work cited here will continue to grow as more algorithms and codes are developed."*

- R. Fletcher, S. Leyffer, and Ph. L. Toint, A brief history of filter methods, SIAM SIAG/OPT Views-and-News, 18 (1) (2006) 2–12.

Nonlinear programming (NLP) problems:

$$\min_{x \in \mathbb{R}^n} f(x) \quad \text{s.t.} \quad h(x) = 0 \text{ and } x \geq 0,$$

where $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ and $h : \mathbb{R}^n \longrightarrow \mathbb{R}^m$ are twice continuously differentiable functions on an open set $\Omega \subset \mathbb{R}^n$.

Karush-Kuhn-Tucker (KKT) Conditions

Lagrangian: $\ell(x, y, z) = f(x) + h(x)^\top y - x^\top z$.

First-order necessary conditions:

$$
\begin{aligned}
\nabla_x \ell(x, y, z) &= 0 \iff \nabla f(x) + \nabla h(x) y - z = 0 \\
h(x) &= 0 \\
Xz &= 0 \\
x \geq 0 \ \text{e} \ z &\geq 0,
\end{aligned}
$$

where $y \in \mathbb{R}^m$ and $z \in \mathbb{R}^n$ are the Lagrange multipliers.

Karush-Kuhn-Tucker (KKT) Conditions

Lagrangian: $\ell(x, y, z) = f(x) + h(x)^\top y - x^\top z$.

First-order necessary conditions:

$$\begin{aligned}
\nabla_x \ell(x, y, z) &= 0 \iff \nabla f(x) + \nabla h(x) y - z = 0 \\
h(x) &= 0 \\
Xz &= 0 \\
x \geq 0 \ \text{e} \ z &\geq 0,
\end{aligned}$$

where $y \in \mathbb{R}^m$ and $z \in \mathbb{R}^n$ are the Lagrange multipliers.

Perturbed KKT System

Perturbing the KKT system, we obtain:

$$\begin{aligned} \nabla_x \ell(x,y,z) &= 0 \\ h(x) &= 0 \\ Xz &= \hat{\mu} e. \end{aligned}$$

The step $\Delta w = (\Delta x, \Delta y, \Delta z)$ is the solution of the Newton system:

$$\begin{pmatrix} \nabla^2_{xx}\ell(x,y,z) & \nabla h(x) & -I \\ \nabla h(x)^\top & 0 & 0 \\ Z & 0 & X \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta z \end{pmatrix} = - \begin{pmatrix} \nabla_x \ell(x,y,z) \\ h(x) \\ Xz - \hat{\mu} e \end{pmatrix}.$$

Perturbed KKT System

Perturbing the KKT system, we obtain:

$$\begin{aligned} \nabla_x \ell(x, y, z) &= 0 \\ h(x) &= 0 \\ Xz &= \hat{\mu} e. \end{aligned}$$

The step $\Delta w = (\Delta x, \Delta y, \Delta z)$ is the solution of the Newton system:

$$\begin{pmatrix} \nabla^2_{xx} \ell(x, y, z) & \nabla h(x) & -I \\ \nabla h(x)^\top & 0 & 0 \\ Z & 0 & X \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta z \end{pmatrix} = - \begin{pmatrix} \nabla_x \ell(x, y, z) \\ h(x) \\ Xz - \hat{\mu} e \end{pmatrix}.$$

Linear Programming (LP): $f(x) = c^\top x$ and $h(x) = Ax - b$.

The central path is formed by all (x, y, z) such that $(x, z) > 0$ and

$$\begin{aligned} \nabla_x \ell(x, y, z) &= 0 \\ h(x) &= 0 \\ Xz &= \hat{\mu} e, \end{aligned}$$

for all $\hat{\mu} > 0$.

Central Path – LP

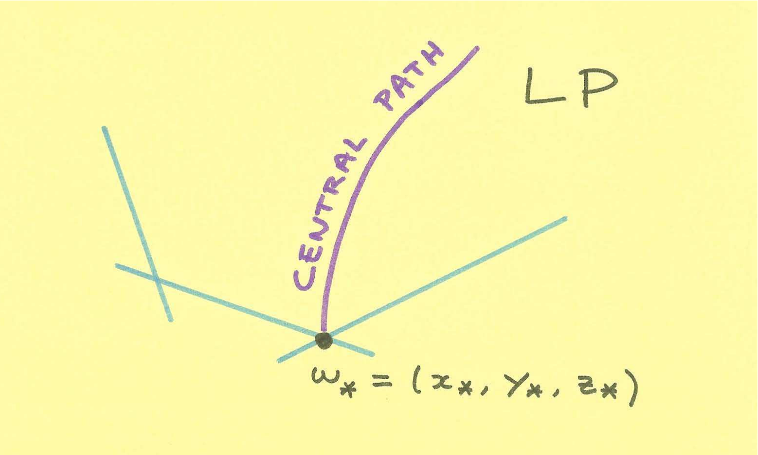
Linear Programming (LP): $f(x) = c^\top x$ and $h(x) = Ax - b$.

The central path is formed by all (x, y, z) such that $(x, z) > 0$ and

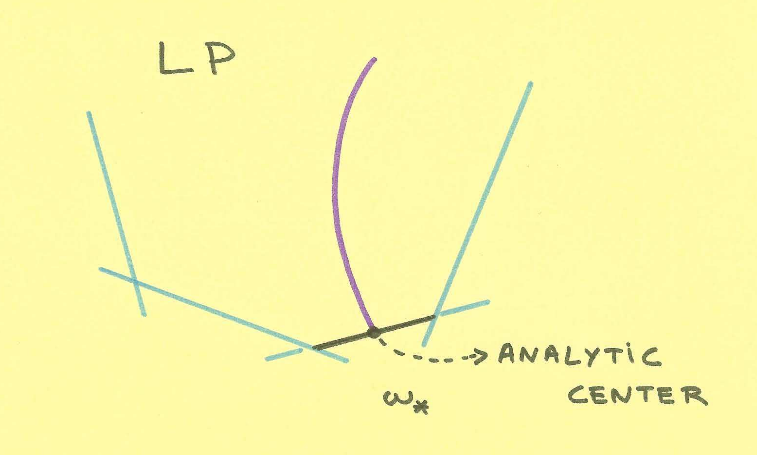
$$\begin{aligned} \nabla_x \ell(x, y, z) &= 0 \\ h(x) &= 0 \\ Xz &= \hat{\mu} e, \end{aligned}$$

for all $\hat{\mu} > 0$.

Central Path – LP



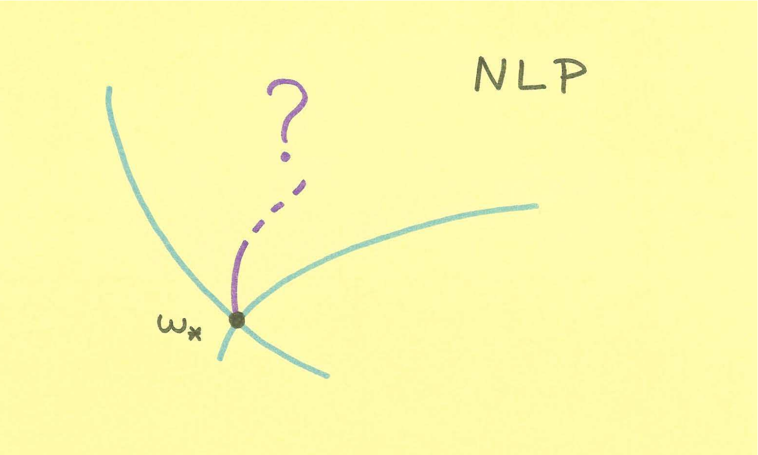
Central Path – LP



Central Path – NLP

The central path only exists locally in NLP!

Central Path – NLP



The quasi-central path is formed by all $(x, z) > 0$ such that

$$\begin{aligned} h(x) &= 0 \\ Xz &= \hat{\mu} e, \end{aligned}$$

for all $\hat{\mu} > 0$.

The quasi-central path has a global existence.

Quasi-Central Path – NLP

The quasi-central path is formed by all $(x, z) > 0$ such that

$$\begin{aligned} h(x) &= 0 \\ Xz &= \hat{\mu} e, \end{aligned}$$

for all $\hat{\mu} > 0$.

The quasi-central path has a global existence.

Perturbed KKT System

Perturbing the KKT system, we obtain:

$$
\begin{aligned}
\nabla_x \ell(x, y, z) &= 0 \\
h(x) &= 0 \\
Xz &= \hat{\mu} e.
\end{aligned}
$$

The step $\Delta w = (\Delta x, \Delta y, \Delta z)$ is the solution of the Newton system:

$$
\begin{pmatrix} \nabla^2_{xx} \ell(x, y, z) & \nabla h(x) & -I \\ \nabla h(x)^\top & 0 & 0 \\ Z & 0 & X \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta z \end{pmatrix} = - \begin{pmatrix} \nabla_x \ell(x, y, z) \\ h(x) \\ Xz - \hat{\mu} e \end{pmatrix}.
$$

We set $\hat{\mu} = \sigma \mu$, with $\mu = \frac{x^\top z}{n}$ a measure of centrality and $\sigma \in (0, 1)$ a centering parameter.

Perturbed KKT System

Perturbing the KKT system, we obtain:

$$\begin{aligned} \nabla_x \ell(x, y, z) &= 0 \\ h(x) &= 0 \\ Xz &= \hat{\mu} e. \end{aligned}$$

The step $\Delta w = (\Delta x, \Delta y, \Delta z)$ is the solution of the Newton system:

$$\begin{pmatrix} \nabla^2_{xx} \ell(x, y, z) & \nabla h(x) & -I \\ \nabla h(x)^\top & 0 & 0 \\ Z & 0 & X \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta z \end{pmatrix} = - \begin{pmatrix} \nabla_x \ell(x, y, z) \\ h(x) \\ Xz - \hat{\mu} e \end{pmatrix}.$$

We set $\hat{\mu} = \sigma \mu$, with $\mu = \frac{x^\top z}{n}$ a measure of centrality and $\sigma \in (0, 1)$ a centering parameter.

Perturbed KKT System

Perturbing the KKT system, we obtain:

$$\begin{aligned} \nabla_x \ell(x, y, z) &= 0 \\ h(x) &= 0 \\ Xz &= \hat{\mu} e. \end{aligned}$$

The step $\Delta w = (\Delta x, \Delta y, \Delta z)$ is the solution of the Newton system:

$$\begin{pmatrix} \nabla^2_{xx} \ell(x, y, z) & \nabla h(x) & -I \\ \nabla h(x)^\top & 0 & 0 \\ Z & 0 & X \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta z \end{pmatrix} = - \begin{pmatrix} \nabla_x \ell(x, y, z) \\ h(x) \\ Xz - \hat{\mu} e \end{pmatrix}.$$

We set $\hat{\mu} = \sigma \mu$, with $\mu = \frac{x^\top z}{n}$ a measure of centrality and $\sigma \in (0, 1)$ a centering parameter.

Motivation – in the NLP Context

- Find optimality and feasibility measures.
- Find a decomposition of the primal-dual step Δw (normal and tangential) yielding decrease in both measures, respectively.

Feasibility and centrality measure:

$$\theta(w) = \|h(x)\| + \|Xz - \mu e\|$$

Optimality measure:

$$\theta_g(w) = \mu + \|\nabla_x \ell(w)\|^2$$

$w = (x, y, z)$, with $(x, z) \geq 0$, satisfies the KKT conditions if and only if

$$\theta(w) = \theta_g(w) = 0.$$

Filter Measures

Feasibility and centrality measure:

$$\theta(w) = \|h(x)\| + \|Xz - \mu e\|$$

Optimality measure:

$$\theta_g(w) = \mu + \|\nabla_x \ell(w)\|^2$$

$w = (x, y, z)$, with $(x, z) \geq 0$, satisfies the KKT conditions if and only if

$$\theta(w) = \theta_g(w) = 0.$$

Filter Measures

Feasibility and centrality measure:

$$\theta(w) = \|h(x)\| + \|Xz - \mu e\|$$

Optimality measure:

$$\theta_g(w) = \mu + \|\nabla_x \ell(w)\|^2$$

$w = (x, y, z)$, with $(x, z) \geq 0$, satisfies the KKT conditions if and only if

$$\theta(w) = \theta_g(w) = 0.$$

Step Decomposition

Normal step:

$$\begin{pmatrix} \nabla^2_{xx}\ell(w) & \nabla h(x) & -I \\ \nabla h(x)^\top & 0 & 0 \\ Z & 0 & X \end{pmatrix} s^n = -\begin{pmatrix} 0 \\ h(x) \\ Xz - \mu e \end{pmatrix},$$

"towards" the quasi-central path.

Tangential step:

$$\begin{pmatrix} \nabla^2_{xx}\ell(w) & \nabla h(x) & -I \\ \nabla h(x)^\top & 0 & 0 \\ Z & 0 & X \end{pmatrix} s^t = -\begin{pmatrix} \nabla_x \ell(w) \\ 0 \\ (1-\sigma)\mu e \end{pmatrix}.$$

Adding both steps: $\Delta w = s^n + s^t$.

Step Decomposition

Normal step:

$$\begin{pmatrix} \nabla^2_{xx}\ell(w) & \nabla h(x) & -I \\ \nabla h(x)^\top & 0 & 0 \\ Z & 0 & X \end{pmatrix} s^n = -\begin{pmatrix} 0 \\ h(x) \\ Xz - \mu e \end{pmatrix},$$

"towards" the quasi-central path.

Tangential step:

$$\begin{pmatrix} \nabla^2_{xx}\ell(w) & \nabla h(x) & -I \\ \nabla h(x)^\top & 0 & 0 \\ Z & 0 & X \end{pmatrix} s^t = -\begin{pmatrix} \nabla_x \ell(w) \\ 0 \\ (1-\sigma)\mu e \end{pmatrix}.$$

Adding both steps: $\Delta w = s^n + s^t$.

Step Decomposition

Normal step:

$$\begin{pmatrix} \nabla^2_{xx}\ell(w) & \nabla h(x) & -I \\ \nabla h(x)^\top & 0 & 0 \\ Z & 0 & X \end{pmatrix} s^n = -\begin{pmatrix} 0 \\ h(x) \\ Xz - \mu e \end{pmatrix},$$

"towards" the quasi-central path.

Tangential step:

$$\begin{pmatrix} \nabla^2_{xx}\ell(w) & \nabla h(x) & -I \\ \nabla h(x)^\top & 0 & 0 \\ Z & 0 & X \end{pmatrix} s^t = -\begin{pmatrix} \nabla_x \ell(w) \\ 0 \\ (1-\sigma)\mu e \end{pmatrix}.$$

Adding both steps: $\Delta w = s^n + s^t$.

Step Sizes

Using a trust-region type parameter:

$$\|\alpha^n(\Delta)s^n\| \leq \Delta, \qquad \|\alpha^t(\Delta)s^t\| \leq \Delta$$

$$\Downarrow$$

$$\|s(\Delta)\| = \alpha^n(\Delta)s^n + \alpha^t(\Delta)s^t \leq 2\Delta$$

$$\alpha^n(\Delta) = \min\left\{1, \frac{\Delta}{\|s^n\|}\right\}, \quad \alpha^t(\Delta) = \min\left\{\alpha^n(\Delta), \frac{\Delta}{\|s^t\|}\right\}$$

$$w(\Delta) = w + s(\Delta) = (x(\Delta), y(\Delta), z(\Delta)).$$

Step Sizes

Using a trust-region type parameter:

$$\|\alpha^n(\Delta)s^n\| \leq \Delta, \qquad \|\alpha^t(\Delta)s^t\| \leq \Delta$$

$$\Downarrow$$

$$\|s(\Delta)\| = \alpha^n(\Delta)s^n + \alpha^t(\Delta)s^t \leq 2\Delta$$

$$\alpha^n(\Delta) = \min\left\{1, \frac{\Delta}{\|s^n\|}\right\}, \quad \alpha^t(\Delta) = \min\left\{\alpha^n(\Delta), \frac{\Delta}{\|s^t\|}\right\}$$

$$w(\Delta) = w + s(\Delta) = (x(\Delta), y(\Delta), z(\Delta)).$$

Step Sizes

Using a trust-region type parameter:

$$\|\alpha^n(\Delta)s^n\| \leq \Delta, \qquad \|\alpha^t(\Delta)s^t\| \leq \Delta$$

$$\Downarrow$$

$$\|s(\Delta)\| = \alpha^n(\Delta)s^n + \alpha^t(\Delta)s^t \leq 2\Delta$$

$$\alpha^n(\Delta) = \min\left\{1, \frac{\Delta}{\|s^n\|}\right\}, \quad \alpha^t(\Delta) = \min\left\{\alpha^n(\Delta), \frac{\Delta}{\|s^t\|}\right\}$$

$$w(\Delta) = w + s(\Delta) = (x(\Delta), y(\Delta), z(\Delta)).$$

Using a trust-region type parameter:

$$\|\alpha^n(\Delta)s^n\| \leq \Delta, \qquad \|\alpha^t(\Delta)s^t\| \leq \Delta$$

$$\Downarrow$$

$$\|s(\Delta)\| = \alpha^n(\Delta)s^n + \alpha^t(\Delta)s^t \leq 2\Delta$$

$$\alpha^n(\Delta) = \min\left\{1, \frac{\Delta}{\|s^n\|}\right\}, \quad \alpha^t(\Delta) = \min\left\{\alpha^n(\Delta), \frac{\Delta}{\|s^t\|}\right\}$$

$$w(\Delta) = w + s(\Delta) = (x(\Delta), y(\Delta), z(\Delta)).$$

The iterates will be kept in:

$$\mathcal{N}(\gamma, M) = \{w : \ (x, z) > 0; \quad Xz \geq \gamma\mu e; \quad \|h(w)\| + \|\nabla_x \ell(w)\| \leq M\mu\}.$$

Lemma

If $\quad \|F'(w)^{-1}\| \leq C, \quad \gamma \in (0,1), \quad M > 0, \quad$ then

$$\boxed{\begin{array}{c} w \in \mathcal{N}(\gamma, M) \\ \Downarrow \\ \exists \Delta_{\min} > 0 \ : \ w(\Delta) \in \mathcal{N}(\gamma, M) \end{array}}$$

for all $\Delta \in]0, \Delta_{\min}]$.

$F'(w)$ is the KKT matrix (the Jacobian of the KKT residual).

Centrality Neighborhood

The iterates will be kept in:

$$\mathcal{N}(\gamma, M) = \{w : \ (x, z) > 0; \quad Xz \geq \gamma\mu e; \quad \|h(w)\| + \|\nabla_x \ell(w)\| \leq M\mu\}.$$

Lemma

If $\quad \|F'(w)^{-1}\| \leq C, \quad \gamma \in (0, 1), \quad M > 0, \quad$ then

$$\boxed{\begin{array}{c} w \in \mathcal{N}(\gamma, M) \\ \Downarrow \\ \exists \Delta_{\text{min}} > 0 \ : \ w(\Delta) \in \mathcal{N}(\gamma, M) \end{array}}$$

for all $\Delta \in]0, \Delta_{\text{min}}]$.

$F'(w)$ is the KKT matrix (the Jacobian of the KKT residual).

Primal-Dual Interior-Point Filter Method

- Decomposition of the primal-dual step (normal and tangential) yielding decrease in a feasibility and an optimality measures (associated with the filter components).
- Different step lengths for the normal and the tangential steps.
- The iterates are kept in a central neighborhood.
- Possible use of a feasibility restoration procedure.
- The new iterates are always acceptable to the filter.
- Imposition of a sufficient decrease if the feasibility measure is small enough.

Primal-Dual Interior-Point Filter Method

- Decomposition of the primal-dual step (normal and tangential) yielding decrease in a feasibility and an optimality measures (associated with the filter components).
- Different step lengths for the normal and the tangential steps.
- The iterates are kept in a central neighborhood.
- Possible use of a feasibility restoration procedure.
- The new iterates are always acceptable to the filter.
- Imposition of a sufficient decrease if the feasibility measure is small enough.

Primal-Dual Interior-Point Filter Method

- Decomposition of the primal-dual step (normal and tangential) yielding decrease in a feasibility and an optimality measures (associated with the filter components).

- Different step lengths for the normal and the tangential steps.

- The iterates are kept in a central neighborhood.

- Possible use of a feasibility restoration procedure.

- The new iterates are always acceptable to the filter.

- Imposition of a sufficient decrease if the feasibility measure is small enough.

Primal-Dual Interior-Point Filter Method

- Decomposition of the primal-dual step (normal and tangential) yielding decrease in a feasibility and an optimality measures (associated with the filter components).
- Different step lengths for the normal and the tangential steps.
- The iterates are kept in a central neighborhood.
- Possible use of a feasibility restoration procedure.
- The new iterates are always acceptable to the filter.
- Imposition of a sufficient decrease if the feasibility measure is small enough.

Primal-Dual Interior-Point Filter Method

- Decomposition of the primal-dual step (normal and tangential) yielding decrease in a feasibility and an optimality measures (associated with the filter components).
- Different step lengths for the normal and the tangential steps.
- The iterates are kept in a central neighborhood.
- Possible use of a feasibility restoration procedure.
- The new iterates are always acceptable to the filter.
- Imposition of a sufficient decrease if the feasibility measure is small enough.

Primal-Dual Interior-Point Filter Method

- Decomposition of the primal-dual step (normal and tangential) yielding decrease in a feasibility and an optimality measures (associated with the filter components).
- Different step lengths for the normal and the tangential steps.
- The iterates are kept in a central neighborhood.
- Possible use of a feasibility restoration procedure.
- The new iterates are always acceptable to the filter.
- Imposition of a sufficient decrease if the feasibility measure is small enough.

0. Choose $(x_0, z_0) > 0$, y_0, $\Delta_0 > 0$, $k = 0$. Choose also γ, M such that $(x_0, y_0, z_0) \in \mathcal{N}(\gamma, M)$.

1. Stop if $\boxed{\theta(w_k) + \theta_g(w_k) < \epsilon_{\text{tol}}}$. Otherwise compute s_k^n and s_k^t.

2/3. Compute Δ_k such that $w_k(\Delta_k) \in \mathcal{N}(\gamma, M)$.

0. Choose $(x_0, z_0) > 0$, y_0, $\Delta_0 > 0$, $k = 0$. Choose also γ, M such that $(x_0, y_0, z_0) \in \mathcal{N}(\gamma, M)$.

1. Stop if $\boxed{\theta(w_k) + \theta_g(w_k) < \epsilon_{\text{tol}}}$. Otherwise compute s_k^n and s_k^t.

2/3. Compute Δ_k such that $w_k(\Delta_k) \in \mathcal{N}(\gamma, M)$.

Primal-Dual Interior-Point Filter Method

0. Choose $(x_0, z_0) > 0$, y_0, $\Delta_0 > 0$, $k = 0$. Choose also γ, M such that $(x_0, y_0, z_0) \in \mathcal{N}(\gamma, M)$.

1. Stop if $\boxed{\theta(w_k) + \theta_g(w_k) < \epsilon_{\text{tol}}}$. Otherwise compute s_k^n and s_k^t.

2/3. Compute Δ_k such that $w_k(\Delta_k) \in \mathcal{N}(\gamma, M)$.

Sufficient reduction criterion: $\boxed{\rho_k \geq \eta}$

$$\eta \in (0,1) \quad \text{and} \quad \rho_k \overset{\text{def}}{=} \frac{\text{actual reduction}}{\text{predicted reduction}} \overset{\text{def}}{=} \frac{\theta_g(w_k) - \theta_g(w_k(\Delta_k))}{m_k(w_k) - m_k(w_k(\Delta_k))}$$

where

$$\boxed{\begin{array}{c} m \overset{\text{def}}{=} \text{linearization of } \frac{x^\top z}{n} \\ + \\ \text{squared norm of the linearization of } \nabla_x \ell(w) \end{array}}$$

is a quadratic model of θ_g.

Sufficient reduction criterion: $\boxed{\rho_k \geq \eta}$

$$\eta \in (0,1) \quad \text{and} \quad \rho_k \stackrel{\text{def}}{=} \frac{\text{actual reduction}}{\text{predicted reduction}} \stackrel{\text{def}}{=} \frac{\theta_g(w_k) - \theta_g(w_k(\Delta_k))}{m_k(w_k) - m_k(w_k(\Delta_k))}$$

where

$$\boxed{\begin{array}{c} m \stackrel{\text{def}}{=} \text{linearization of } \frac{x^\top z}{n} \\ + \\ \text{squared norm of the linearization of } \nabla_x \ell(w) \end{array}}$$

is a quadratic model of θ_g.

Sufficient reduction criterion: $\boxed{\rho_k \geq \eta}$

$$\eta \in (0,1) \quad \text{and} \quad \rho_k \stackrel{\text{def}}{=} \frac{\text{actual reduction}}{\text{predicted reduction}} \stackrel{\text{def}}{=} \frac{\theta_g(w_k) - \theta_g(w_k(\Delta_k))}{m_k(w_k) - m_k(w_k(\Delta_k))}$$

where

$$m \stackrel{\text{def}}{=} \text{linearization of } \frac{x^\top z}{n} + \text{squared norm of the linearization of } \nabla_x \ell(w)$$

is a quadratic model of θ_g.

Sufficient reduction criterion: $\boxed{\rho_k \geq \eta}$

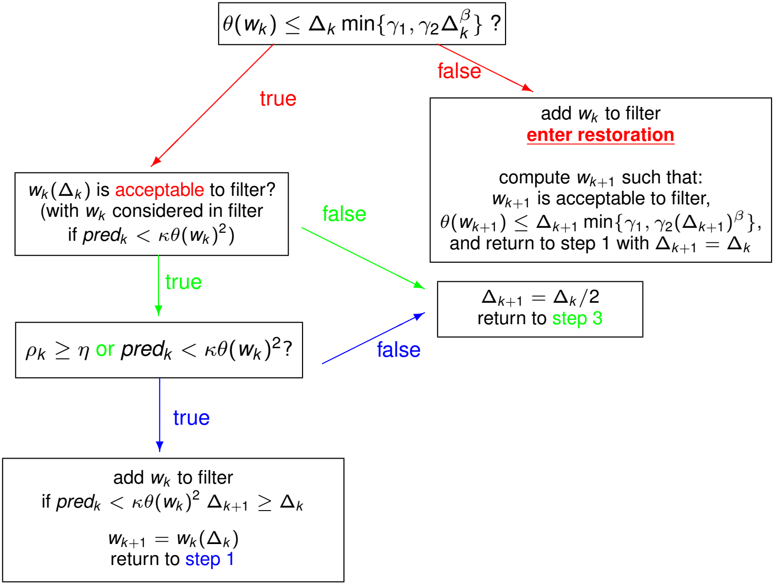
$$\eta \in (0,1) \quad \text{and} \quad \rho_k \stackrel{\text{def}}{=} \frac{\text{actual reduction}}{\text{predicted reduction}} \stackrel{\text{def}}{=} \frac{\theta_g(w_k) - \theta_g(w_k(\Delta_k))}{m_k(w_k) - m_k(w_k(\Delta_k))}$$

where

$$\boxed{\begin{array}{c} m \stackrel{\text{def}}{=} \text{linearization of } \frac{x^\top z}{n} \\ + \\ \text{squared norm of the linearization of } \nabla_x \ell(w) \end{array}}$$

is a quadratic model of θ_g.

Primal-Dual Interior-Point Filter Method



Global Convergence – Assumptions

(A1) $\{(x_k, y_k, z_k\}$ is bounded.

(A2) ∇h and $\nabla^2_{xw}\ell$ are Lipschitz continuous in an open set D such that $w_k \in D$ and $[w_k, w_k + s(\Delta_k)] \in D$.

(A3) $\exists C > 0$ such that $\|F'(w)^{-1}\| \leq C$ for all k.

Global Convergence – Main Result

Theorem

$$\liminf_{k \to +\infty} \theta(w_k) + \theta_g(w_k) = 0$$

(there exists a limit point that is a KKT point).

Previous filter optimality measure:

$$\theta_g(w) = \mu + \|\nabla_x \ell(w)\|^2$$

- Choice made in Ulbrich, Ulbrich, and Vicente [2004].
- Does not distinguish minima from maxima!

New Filter Measures

New possibilities ($c > 0$):

$$\theta_g(w) \;=\; f(x) + c\mu$$

or

$$\theta_g(w) \;=\; f(x) + h(x)^\top y + c\mu \;=\; \ell(x,y,z) + (c+n)\mu$$

- Reflect better the minimization goal.
- f is the driven force (when $h(x)$ and $\mu = x^\top z/n$ are small).

Let $w \in \mathcal{N}(\gamma, M)$ and $c \geq constant(\sigma, \gamma, n)$.

We can derive an estimate of the type:

$$\theta_g(w(\Delta)) - \theta_g(w) \;\leq\; -(K\mu)\Delta + \mathcal{O}(\theta(w)) + \mathcal{O}(\Delta^2).$$

When $\theta(w)$ is small, the primal-dual step is a descent direction for $\theta_g(w)$.

New Filter Measures

Let $w \in \mathcal{N}(\gamma, M)$ and $c \geq constant(\sigma, \gamma, n)$.

We can derive an estimate of the type:

$$\theta_g(w(\Delta)) - \theta_g(w) \ \leq \ -(K\mu)\Delta + \mathcal{O}(\theta(w)) + \mathcal{O}(\Delta^2).$$

When $\theta(w)$ is small, the primal-dual step is a descent direction for $\theta_g(w)$.

New Filter Measures

Let $w \in \mathcal{N}(\gamma, M)$ and $c \geq constant(\sigma, \gamma, n)$.

We can derive an estimate of the type:

$$\theta_g(w(\Delta)) - \theta_g(w) \;\leq\; -(K\mu)\Delta + \mathcal{O}(\theta(w)) + \mathcal{O}(\Delta^2).$$

When $\theta(w)$ is small, the primal-dual step is a descent direction for $\theta_g(w)$.

New Filter Measures

One can prove as before:

Thorem

$$\liminf_{k \to +\infty} \; \|h(x_k)\| + \frac{x_k^\top z_k}{n} + \|\nabla_x \ell(w_k)\| \; = \; 0$$

(there exists a limit point that is a KKT point).

...under the assumption that $\nabla^2_{xx}\ell(w_k) + (1/2)X_k^{-\frac{1}{2}} Z_k$ is positive semi-definite on the null space of $\nabla g(x_k)^\top$ for all k...

...same algorithm (the only modification is in the model m for $\theta_g(w)$)...

Consider a more general scenario that allows the use of different types of approximations $H \neq \nabla^2_{xx}\ell(w)$:

$$\mathcal{N}(\gamma, M, p) = \{w : (x,z) > 0, Xz \geq \gamma\mu e, \|h(x)\| + \|\nabla_x \ell(w)\|^p \leq M\mu\}$$

is the family of centrality neighborhoods parameterized by $p \in [1, 2]$.

- When $p = 1$: $\mathcal{N}(\gamma, M, 1) = \mathcal{N}(\gamma, M)$ (*old centrality neighborhood*).

As before, we can prove that

$w \in \mathcal{N}(\gamma, M, p) \Longrightarrow w(\Delta) \in \mathcal{N}(\gamma, M, p)$, for $\Delta > 0$ sufficiently small.

New Centrality Neighborhoods

Consider a more general scenario that allows the use of different types of approximations $H \neq \nabla^2_{xx}\ell(w)$:

$$\mathcal{N}(\gamma, M, p) = \{w\ : (x, z) > 0, Xz \geq \gamma\mu e, \|h(x)\| + \|\nabla_x \ell(w)\|^p \leq M\mu\}$$

is the family of centrality neighborhoods parameterized by $p \in [1, 2]$.

- When $p = 1$: $\mathcal{N}(\gamma, M, 1) = \mathcal{N}(\gamma, M)$ (*old centrality neighborhood*).

As before, we can prove that

$$\boxed{w \in \mathcal{N}(\gamma, M, p) \Longrightarrow w(\Delta) \in \mathcal{N}(\gamma, M, p), \text{ for } \Delta > 0 \text{ sufficiently small.}}$$

New Centrality Neighborhoods

Consider a more general scenario that allows the use of different types of approximations $H \neq \nabla^2_{xx}\ell(w)$:

$$\mathcal{N}(\gamma, M, p) = \{w \ : (x,z) > 0, Xz \geq \gamma\mu e, \|h(x)\| + \|\nabla_x \ell(w)\|^p \leq M\mu\}$$

is the family of centrality neighborhoods parameterized by $p \in [1,2]$.

- When $p = 1$: $\mathcal{N}(\gamma, M, 1) = \mathcal{N}(\gamma, M)$ (*old centrality neighborhood*).

As before, we can prove that

$w \in \mathcal{N}(\gamma, M, p) \implies w(\Delta) \in \mathcal{N}(\gamma, M, p)$, for $\Delta > 0$ sufficiently small.

New Centrality Neighborhoods

Consider a more general scenario that allows the use of different types of approximations $H \neq \nabla^2_{xx}\ell(w)$:

$$\mathcal{N}(\gamma, M, p) = \{w\ : (x,z) > 0, Xz \geq \gamma\mu e, \|h(x)\| + \|\nabla_x \ell(w)\|^p \leq M\mu\}$$

is the family of centrality neighborhoods parameterized by $p \in [1,2]$.

- When $p = 1$: $\mathcal{N}(\gamma, M, 1) = \mathcal{N}(\gamma, M)$ (*old centrality neighborhood*).

As before, we can prove that

$$\boxed{w \in \mathcal{N}(\gamma, M, p) \Longrightarrow w(\Delta) \in \mathcal{N}(\gamma, M, p),\ \text{for } \Delta > 0 \text{ sufficiently small.}}$$

Use of Second-Order Derivatives

A sufficient decrease condition on $\|\nabla_x \ell(w)\|^p$ must be satisfied:

$$\|\nabla_x \ell(w(\Delta))\|^p \leq (1 - p\,\alpha^t(\Delta))\|\nabla_x \ell(w)\|^p + M_\ell \max\{\Delta^q, \Delta^2\},$$

where $q \in (1, 2]$.

- TRUE for $p = 1$, $q = 2$, and $H = \nabla^2_{xx}\ell(w)$ — UUV 2004
- When $p = 1$, TRUE if

$$\|H - \nabla^2_{xx}\ell(w)\| \leq N\Delta^{q-1}.$$

- When $p = 2$, TRUE if

$$\|[H - \nabla^2_{xx}\ell(w)]\nabla_x \ell(w)\| \leq N\|\nabla_x \ell(w)\|\Delta^{q-1}.$$

Use of Second-Order Derivatives

A sufficient decrease condition on $\|\nabla_x \ell(w)\|^p$ must be satisfied:

$$\|\nabla_x \ell(w(\Delta))\|^p \leq (1 - p\,\alpha^t(\Delta))\|\nabla_x \ell(w)\|^p + M_\ell \max\{\Delta^q, \Delta^2\},$$

where $q \in (1, 2]$.

- TRUE for $p = 1$, $q = 2$, and $H = \nabla^2_{xx}\ell(w)$ — UUV 2004
- When $p = 1$, TRUE if

$$\|H - \nabla^2_{xx}\ell(w)\| \leq N\Delta^{q-1}.$$

- When $p = 2$, TRUE if

$$\|[H - \nabla^2_{xx}\ell(w)]\nabla_x \ell(w)\| \leq N\|\nabla_x \ell(w)\|\Delta^{q-1}.$$

Use of Second-Order Derivatives

A sufficient decrease condition on $\|\nabla_x \ell(w)\|^p$ must be satisfied:

$$\|\nabla_x \ell(w(\Delta))\|^p \leq (1 - p\,\alpha^t(\Delta))\|\nabla_x \ell(w)\|^p + M_\ell \max\{\Delta^q, \Delta^2\},$$

where $q \in (1, 2]$.

- TRUE for $p = 1$, $q = 2$, and $H = \nabla^2_{xx}\ell(w)$ — UUV 2004
- When $p = 1$, TRUE if

$$\|H - \nabla^2_{xx}\ell(w)\| \leq N\Delta^{q-1}.$$

- When $p = 2$, TRUE if

$$\|[H - \nabla^2_{xx}\ell(w)]\nabla_x \ell(w)\| \leq N\|\nabla_x \ell(w)\|\Delta^{q-1}.$$

Use of Second-Order Derivatives

A sufficient decrease condition on $\|\nabla_x \ell(w)\|^p$ must be satisfied:

$$\|\nabla_x \ell(w(\Delta))\|^p \leq (1 - p\,\alpha^t(\Delta))\|\nabla_x \ell(w)\|^p + M_\ell \max\{\Delta^q, \Delta^2\},$$

where $q \in (1, 2]$.

- TRUE for $p = 1$, $q = 2$, and $H = \nabla^2_{xx}\ell(w)$ — UUV 2004
- When $p = 1$, TRUE if

$$\|H - \nabla^2_{xx}\ell(w)\| \leq N\Delta^{q-1}.$$

- When $p = 2$, TRUE if

$$\|[H - \nabla^2_{xx}\ell(w)]\nabla_x \ell(w)\| \leq N\|\nabla_x \ell(w)\|\Delta^{q-1}.$$

Use of Second-Order Derivatives

A sufficient decrease condition on $\|\nabla_x \ell(w)\|^p$ must be satisfied:

$$\|\nabla_x \ell(w(\Delta))\|^p \leq (1 - p\,\alpha^t(\Delta))\|\nabla_x \ell(w)\|^p + M_\ell \max\{\Delta^q, \Delta^2\},$$

where $q \in (1, 2]$.

- TRUE for $p = 1$, $q = 2$, and $H = \nabla^2_{xx}\ell(w)$ — UUV 2004
- When $p = 1$, TRUE if

$$\|H - \nabla^2_{xx}\ell(w)\| \leq N\Delta^{q-1}.$$

- When $p = 2$, TRUE if

$$\|[H - \nabla^2_{xx}\ell(w)]\nabla_x \ell(w)\| \leq N\|\nabla_x \ell(w)\|\Delta^{q-1}.$$

Restoration Phase

Purpose:

Find w_{k+1} such that

- $w_{k+1} \in \mathcal{N}(\gamma, M)$
- w_{k+1} is acceptable to the filter
- $\theta(w_{k+1}) \leq \Delta_k \min\{\gamma_1, \gamma_2 \Delta_k^{\beta}\}$.

The restoration algorithm must terminate in a finite number of iterations.

Restoration Phase

Purpose:

Find w_{k+1} such that

- $w_{k+1} \in \mathcal{N}(\gamma, M)$
- w_{k+1} is acceptable to the filter
- $\theta(w_{k+1}) \leq \Delta_k \min\{\gamma_1, \gamma_2 \Delta_k^{\beta}\}$.

The restoration algorithm must terminate in a finite number of iterations.

Restoration Phase

The current restoration algorithm uses our step framework:

$$s(\Delta) = \alpha^n(\Delta)s^n + \alpha^t(\Delta)s^t, \qquad w(\Delta) = w + s(\Delta),$$

imposing descent on: $\theta_2(w) = \frac{1}{2}\left(\|h(x)\|^2 + \|Xz - \mu e\|^2\right).$

Note: $\nabla\theta_2(w)^\top s^n = -\theta_2(w)$ and $\nabla\theta_2(w)^\top s^t = 0.$

This restoration algorithm is proved to terminate successfully in a finite number of iterations.

Restoration Phase

The current restoration algorithm uses our step framework:

$$s(\Delta) = \alpha^n(\Delta)s^n + \alpha^t(\Delta)s^t, \qquad w(\Delta) = w + s(\Delta),$$

imposing descent on: $\theta_2(w) \; = \; \frac{1}{2}\left(\|h(x)\|^2 + \|Xz - \mu e\|^2\right).$

Note: $\nabla\theta_2(w)^\top s^n = -\theta_2(w)$ and $\nabla\theta_2(w)^\top s^t = 0.$

This restoration algorithm is proved to terminate successfully in a finite number of iterations.

Restoration Phase

The current restoration algorithm uses our step framework:

$$s(\Delta) = \alpha^n(\Delta)s^n + \alpha^t(\Delta)s^t, \qquad w(\Delta) = w + s(\Delta),$$

imposing descent on: $\theta_2(w) = \frac{1}{2}\left(\|h(x)\|^2 + \|Xz - \mu e\|^2\right).$

Note: $\nabla\theta_2(w)^\top s^n = -\theta_2(w)$ and $\nabla\theta_2(w)^\top s^t = 0.$

This restoration algorithm is proved to terminate successfully in a finite number of iterations.

Restoration Phase

The current restoration algorithm uses our step framework:

$$s(\Delta) = \alpha^n(\Delta)s^n + \alpha^t(\Delta)s^t, \qquad w(\Delta) = w + s(\Delta),$$

imposing descent on: $\theta_2(w) \;=\; \frac{1}{2}\left(\|h(x)\|^2 + \|Xz - \mu e\|^2\right)$.

Note: $\nabla\theta_2(w)^\top s^n = -\theta_2(w)$ and $\nabla\theta_2(w)^\top s^t = 0$.

This restoration algorithm is proved to terminate successfully in a finite number of iterations.

Restoration Phase

0. Choose parameters, $w_k^0 = w_k$, $\Delta_k^0 = \Delta_k$, $j = 0$

1. Test the stop criterion

2. Compute $s_{k,j}^n$ and $s_{k,j}^t$

3. Compute Δ_k^j such that, for all $\Delta \in [0, \Delta_k^j]$,

$$X_k^j(\Delta) z_k^j(\Delta) \geq \gamma \mu_k^j(\Delta) \quad \text{and} \quad (x_k^j(\Delta), z_k^j(\Delta)) > 0$$

4.

$\dfrac{\text{ARED}_k^j}{\text{PRED}_k^j} \geq \xi$ and $\|h(w_k^j(\Delta_k^j))\| + \|\nabla_x \ell(w_k^j(\Delta_k^j))\| \leq M\mu_k^j(\Delta_k^j)$?

- False: $\Delta_{k+1}^j = \Delta_k^j / 2$, repeat step 4
- True: $w_k^{j+1} = w_k^j(\Delta_k^j)$, go to step 1

0. Choose parameters, $w_k^0 = w_k$, $\Delta_k^0 = \Delta_k$, $j = 0$

1. Test the stop criterion

2. Compute $s_{k,j}^n$ and $s_{k,j}^t$

3. Compute Δ_k^j such that, for all $\Delta \in [0, \Delta_k^j]$,

$$X_k^j(\Delta) z_k^j(\Delta) \geq \gamma \mu_k^j(\Delta) \quad \text{and} \quad (x_k^j(\Delta), z_k^j(\Delta)) > 0$$

4.

$\dfrac{\text{ARED}_k^j}{\text{PRED}_k^j} \geq \xi$ and
$\|h(w_k^j(\Delta_k^j))\| + \|\nabla_x \ell(w_k^j(\Delta_k^j))\| \leq M\mu_k^j(\Delta_k^j)$?

- False: $\Delta_{k+1}^j = \Delta_k^j / 2$ repeat step 4
- True: $w_k^{j+1} = w_k^j(\Delta_k^j)$ go to step 1

0. Choose parameters, $w_k^0 = w_k$, $\Delta_k^0 = \Delta_k$, $j = 0$

1. Test the stop criterion

2. Compute $s_{k,j}^n$ and $s_{k,j}^t$

3. Compute Δ_k^j such that, for all $\Delta \in [0, \Delta_k^j]$,

$$X_k^j(\Delta)z_k^j(\Delta) \geq \gamma\mu_k^j(\Delta) \quad \text{and} \quad (x_k^j(\Delta), z_k^j(\Delta)) > 0$$

4. $\dfrac{\text{ARED}_k^j}{\text{PRED}_k^j} \geq \xi$ and $\|h(w_k^j(\Delta_k^j))\| + \|\nabla_x \ell(w_k^j(\Delta_k^j))\| \leq M\mu_k^j(\Delta_k^j)$?

   - False: $\Delta_{k+1}^j = \Delta_k^j/2$ repeat step 4
   - True: $w_k^{j+1} = w_k^j(\Delta_k^j)$ go to step 1

Restoration Phase

0. Choose parameters, $w_k^0 = w_k$, $\Delta_k^0 = \Delta_k$, $j = 0$

1. Test the stop criterion

2. Compute $s_{k,j}^n$ and $s_{k,j}^t$

3. Compute Δ_k^j such that, for all $\Delta \in [0, \Delta_k^j]$,

$$X_k^j(\Delta) z_k^j(\Delta) \geq \gamma \mu_k^j(\Delta) \quad \text{and} \quad (x_k^j(\Delta), z_k^j(\Delta)) > 0$$

4.

$\frac{\text{ARED}_k^j}{\text{PRED}_k^j} \geq \xi$ and $\|h(w_k^j(\Delta_k^j))\| + \|\nabla_x \ell(w_k^j(\Delta_k^j))\| \leq M\mu_k^j(\Delta_k^j)$?

- False: $\Delta_{k+1}^j = \Delta_k^j / 2$ repeat step 4
- True: $w_k^{j+1} = w_k^j(\Delta_k^j)$ go to step 1

0. Choose parameters, $w_k^0 = w_k$, $\Delta_k^0 = \Delta_k$, $j = 0$

1. Test the stop criterion

2. Compute $s_{k,j}^n$ and $s_{k,j}^t$

3. Compute Δ_k^j such that, for all $\Delta \in [0, \Delta_k^j]$,

$$X_k^j(\Delta) z_k^j(\Delta) \geq \gamma \mu_k^j(\Delta) \quad \text{and} \quad (x_k^j(\Delta), z_k^j(\Delta)) > 0$$

4.

$\dfrac{\text{ARED}_k^j}{\text{PRED}_k^j} \geq \xi$ and

$\|h(w_k^j(\Delta_k^j))\| + \|\nabla_x \ell(w_k^j(\Delta_k^j))\| \leq M \mu_k^j(\Delta_k^j)$?

- False: $\Delta_{k+1}^j = \Delta_k^j / 2$, repeat step 4
- True: $w_k^{j+1} = w_k^j(\Delta_k^j)$, go to step 1

0. Choose parameters, $w_k^0 = w_k$, $\Delta_k^0 = \Delta_k$, $j = 0$

1. Test the stop criterion

2. Compute $s_{k,j}^n$ and $s_{k,j}^t$

3. Compute Δ_k^j such that, for all $\Delta \in [0, \Delta_k^j]$,

$$X_k^j(\Delta) z_k^j(\Delta) \geq \gamma \mu_k^j(\Delta) \quad \text{and} \quad (x_k^j(\Delta), z_k^j(\Delta)) > 0$$

4.

$$\frac{\text{ARED}_k^j}{\text{PRED}_k^j} \geq \xi \text{ and}$$
$$\|h(w_k^j(\Delta_k^j))\| + \|\nabla_x \ell(w_k^j(\Delta_k^j))\| \leq M \mu_k^j(\Delta_k^j)?$$

- False: $\Delta_{k+1}^j = \Delta_k^j / 2$
  repeat step 4
- True: $w_k^{j+1} = w_k^j(\Delta_k^j)$
  go to step 1

Restoration Phase

0. Choose parameters, $w_k^0 = w_k$, $\Delta_k^0 = \Delta_k$, $j = 0$

1. Test the stop criterion

2. Compute $s_{k,j}^n$ and $s_{k,j}^t$

3. Compute Δ_k^j such that, for all $\Delta \in [0, \Delta_k^j]$,

$$X_k^j(\Delta) z_k^j(\Delta) \geq \gamma \mu_k^j(\Delta) \quad \text{and} \quad (x_k^j(\Delta), z_k^j(\Delta)) > 0$$

4.

$$\frac{\text{ARED}_k^j}{\text{PRED}_k^j} \geq \xi \text{ and}$$
$$\|h(w_k^j(\Delta_k^j))\| + \|\nabla_x \ell(w_k^j(\Delta_k^j))\| \leq M \mu_k^j(\Delta_k^j)?$$

- False: $\Delta_{k+1}^j = \Delta_k^j / 2$, repeat step 4
- True: $w_k^{j+1} = w_k^j(\Delta_k^j)$, go to step 1

Code – `ipfilter 0.2`

- Written in FORTRAN 90/95.
- Handles all kinds of NLP (including unconstrained problems and problems with only simple bounds).
- Includes standard, SIF and AMPL interfaces.
- Symmetrizes the primal-dual systems (avoiding the inversion of X_k):

$$\begin{bmatrix} X_k^{1/2}\nabla^2_{xx}\ell(w_k)X_k^{1/2} + Z_k & X_k^{1/2}\nabla h(w_k) \\ (X_k^{1/2}\nabla h(w_k))^\top & 0 \end{bmatrix}.$$

  Sparse linear algebra handled by HSL routine MA27 (symmetric systems).
- The formulation handles $\ell \leq x \leq u$ explicitly.

Code – `ipfilter 0.2`

- Written in FORTRAN 90/95.
- Handles all kinds of NLP (including unconstrained problems and problems with only simple bounds).
- Includes standard, SIF and AMPL interfaces.
- Symmetrizes the primal-dual systems (avoiding the inversion of X_k):

$$\begin{bmatrix} X_k^{1/2}\nabla^2_{xx}\ell(w_k)X_k^{1/2}+Z_k & X_k^{1/2}\nabla h(w_k) \\ (X_k^{1/2}\nabla h(w_k))^\top & 0 \end{bmatrix}.$$

  Sparse linear algebra handled by HSL routine MA27 (symmetric systems).
- The formulation handles $\ell \le x \le u$ explicitly.

Code – ipfilter 0.2

- Written in FORTRAN 90/95.
- Handles all kinds of NLP (including unconstrained problems and problems with only simple bounds).
- Includes standard, SIF and AMPL interfaces.
- Symmetrizes the primal-dual systems (avoiding the inversion of X_k):

$$\begin{bmatrix} X_k^{1/2}\nabla^2_{xx}\ell(w_k)X_k^{1/2} + Z_k & X_k^{1/2}\nabla h(w_k) \\ (X_k^{1/2}\nabla h(w_k))^\top & 0 \end{bmatrix}.$$

  Sparse linear algebra handled by HSL routine MA27 (symmetric systems).
- The formulation handles $\ell \leq x \leq u$ explicitly.

Code – `ipfilter 0.2`

- Written in FORTRAN 90/95.
- Handles all kinds of NLP (including unconstrained problems and problems with only simple bounds).
- Includes standard, SIF and AMPL interfaces.
- Symmetrizes the primal-dual systems (avoiding the inversion of X_k):

$$\begin{bmatrix} X_k^{1/2}\nabla^2_{xx}\ell(w_k)X_k^{1/2} + Z_k & X_k^{1/2}\nabla h(w_k) \\ (X_k^{1/2}\nabla h(w_k))^\top & 0 \end{bmatrix}.$$

  Sparse linear algebra handled by HSL routine MA27 (symmetric systems).

- The formulation handles $\ell \leq x \leq u$ explicitly.

Code – `ipfilter 0.2`

- Written in FORTRAN 90/95.
- Handles all kinds of NLP (including unconstrained problems and problems with only simple bounds).
- Includes standard, SIF and AMPL interfaces.
- Symmetrizes the primal-dual systems (avoiding the inversion of X_k):

$$\begin{bmatrix} X_k^{1/2}\nabla^2_{xx}\ell(w_k)X_k^{1/2} + Z_k & X_k^{1/2}\nabla h(w_k) \\ (X_k^{1/2}\nabla h(w_k))^\top & 0 \end{bmatrix}.$$

  Sparse linear algebra handled by HSL routine MA27 (symmetric systems).
- The formulation handles $\ell \leq x \leq u$ explicitly.

Code – `ipfilter 0.2`

- Incorporates a restoration phase based on the same step decomposition.

- Performs a warm start by carrying 5 iterations of

$$\min_{x} \; f(x) + \rho\, h(x)^\top h(x) - \mu_0 \sum_{i=1}^{n} \log x_i, \quad \rho > 0,$$

  starting from the initial given point x_0.

- Perturbs the systems using inertia control and regularization

- Scales the system matrix, variables, objective function and constraints.

Code – `ipfilter 0.2`

- Incorporates a restoration phase based on the same step decomposition.

- Performs a warm start by carrying 5 iterations of

$$\min_{x} \; f(x) + \rho\, h(x)^\top h(x) - \mu_0 \sum_{i=1}^{n} \log x_i, \quad \rho > 0,$$

  starting from the initial given point x_0.

- Perturbs the systems using inertia control and regularization

- Scales the system matrix, variables, objective function and constraints.

Code – ipfilter 0.2

- Incorporates a restoration phase based on the same step decomposition.

- Performs a warm start by carrying 5 iterations of

$$\min_{x} \; f(x) + \rho\, h(x)^\top h(x) - \mu_0 \sum_{i=1}^{n} \log x_i, \quad \rho > 0,$$

  starting from the initial given point x_0.

- Perturbs the systems using inertia control and regularization

- Scales the system matrix, variables, objective function and constraints.

- Incorporates a restoration phase based on the same step decomposition.
- Performs a warm start by carrying 5 iterations of

$$\min_{x}\ f(x) + \rho\, h(x)^{\top} h(x) - \mu_0 \sum_{i=1}^{n} \log x_i, \quad \rho > 0,$$

starting from the initial given point x_0.

- Perturbs the systems using inertia control and regularization
- Scales the system matrix, variables, objective function and constraints.

Testing – `ipfilter 0.2`

- We tunned `ipfilter` with an old version of the CUTE collection (469 problems) and we also tested 631 problems from the recent CUTEr collection (Sept. 2008).
- All problems have at least one equality or inequality constraint (different from bounds) and satisfy $n \geq m$.
- The tests were run on a Fujitsu-Siemens Celsius V810 workstation (4G RAM, 2 processors AMD 2.2GHz).
- We made a comparison with `ipopt` (C++, version 3.5.1), a Barrier-Filter code from IBM developed by Andreas Wächter.

  We use the same stopping criterion as in `ipfilter`.
  Stopping tolerance: 10^{-8}; Maximum # of iterations: 1000.

Testing – `ipfilter 0.2`

- We tunned `ipfilter` with an old version of the CUTE collection (469 problems) and we also tested 631 problems from the recent CUTEr collection (Sept. 2008).
- All problems have at least one equality or inequality constraint (different from bounds) and satisfy $n \geq m$.
- The tests were run on a Fujitsu-Siemens Celsius V810 workstation (4G RAM, 2 processors AMD 2.2GHz).
- We made a comparison with `ipopt` (C++, version 3.5.1), a Barrier-Filter code from IBM developed by Andreas Wächter.

  We use the same stopping criterion as in `ipfilter`.
  Stopping tolerance: 10^{-8}; Maximum # of iterations: 1000.

Testing – `ipfilter 0.2`

- We tunned `ipfilter` with an old version of the CUTE collection (469 problems) and we also tested 631 problems from the recent CUTEr collection (Sept. 2008).
- All problems have at least one equality or inequality constraint (different from bounds) and satisfy $n \geq m$.
- The tests were run on a Fujitsu-Siemens Celsius V810 workstation (4G RAM, 2 processors AMD 2.2GHz).
- We made a comparison with `ipopt` (C++, version 3.5.1), a Barrier-Filter code from IBM developed by Andreas Wächter.

  We use the same stopping criterion as in `ipfilter`.
  Stopping tolerance: 10^{-8}; Maximum # of iterations: 1000.

Testing – `ipfilter 0.2`

- We tunned `ipfilter` with an old version of the CUTE collection (469 problems) and we also tested 631 problems from the recent CUTEr collection (Sept. 2008).
- All problems have at least one equality or inequality constraint (different from bounds) and satisfy $n \geq m$.
- The tests were run on a Fujitsu-Siemens Celsius V810 workstation (4G RAM, 2 processors AMD 2.2GHz).
- We made a comparison with `ipopt` (C++, version 3.5.1), a Barrier-Filter code from IBM developed by Andreas Wächter.

  We use the same stopping criterion as in `ipfilter`.
  Stopping tolerance: 10^{-8}; Maximum # of iterations: 1000.

Constrained Tested Problems

	number of problems	
dimensions	an old CUTE set	CUTEr (Sept. 2008)
$n < 1000$	388 (45)	390 (56)
$1000 \leq n < 10000$	76 (1544)	182 (3194)
$n \geq 10000$	5 (7979)	59 (6354)
total	469	631

problem class	an old CUTE set	CUTEr (Sept. 2008)
equality constrained	245	327
inequality constrained	177	226
mixed (equalities and inequalities)	48	78
linearly constrained	171	205
nonlinearly constrained	298	426
quadratic programming	91	103

Numerical Results – All Constrained Problems

	an old CUTE set		CUTEr (Sept. 2008)	
	ipfilter	ipopt	ipfilter	ipopt
# problems solved	449	448	532	549
% robustness	95.74%	95.52%	84.34%	87.00%
# average iterations	27.55	27.19	47.44	38.58
# problems solved ($<$ 500 iter.)	449	447	525	545
% robustness ($<$ 500 iter.)	95.74%	95.31%	83.20%	86.37%
# average iterations ($<$ 500 iter.)	27.55	25.78	37.51	34.14

Numerical Results – QP Problems

	an old CUTE set		CUTEr (Sept. 2008)	
	`ipfilter`	`ipopt`	`ipfilter`	`ipopt`
# problems solved	91	88	97	93
% robustness	100.00%	96.70%	94.17%	90.29%
# average iterations	26.74	36.45	42.35	47.09
# problems solved ($<$ 500 iter.)	91	88	96	92
% robustness ($<$ 500 iter.)	100.00%	96.70%	93.20%	89.32%
# average iterations ($<$ 500 iter.)	26.74	36.45	33.90	40.96

eNLP Solver – ESA Project

- Prime Contractor: Astos Solutions (Germany).
- Consortium partners: University of Birmingham, University of Bremen, *University of Coimbra*, and Skysoft (Portugal).
- Objective: Produce a general purpose European sparse NLP Solver, especially for NLPs associated with trajectory optimization problems.
- Problems: Ascent/reentry of a spacecraft, trajectories planning of satellite missions, launcher test case, low thrust orbit transfer, ...

Numerical Results – COPS Problems

COPS colection: difficult nonlinearly constrained optimization problems arising from *optimal design*, *fluid dynamics*, *mesh smoothing*, *optimal control*,...

- 15 problems from COPS 0.2 are included in the CUTEr collection (Sept. 2008).

	ipfilter 0.2	ipopt 3.5.4
# problems solved	13	13
% robustness	86.67%	86.67%
# average iterations	64.54	98.23
# problems solved ($<$ 500 iter.)	13	12
% robustness ($<$ 500 iter.)	86.67%	80.00%
# average iterations ($<$ 500 iter.)	64.54	53.42

COPS colection: difficult nonlinearly constrained optimization problems arising from *optimal design*, *fluid dynamics*, *mesh smoothing*, *optimal control*,...

- 15 problems from COPS 0.2 are included in the CUTEr collection (Sept. 2008).

	ipfilter 0.2	ipopt 3.5.4
# problems solved	13	13
% robustness	86.67%	86.67%
# average iterations	64.54	98.23
# problems solved ($<$ 500 iter.)	13	12
% robustness ($<$ 500 iter.)	86.67%	80.00%
# average iterations ($<$ 500 iter.)	64.54	53.42

Numerical Results – COPS Problems

COPS colection: difficult nonlinearly constrained optimization problems arising from *optimal design*, *fluid dynamics*, *mesh smoothing*, *optimal control*,...

- 15 problems from COPS 0.2 are included in the CUTEr collection (Sept. 2008).

	`ipfilter 0.2`	`ipopt 3.5.4`
# problems solved	13	13
% robustness	86.67%	86.67%
# average iterations	64.54	98.23
# problems solved ($<$ 500 iter.)	13	12
% robustness ($<$ 500 iter.)	86.67%	80.00%
# average iterations ($<$ 500 iter.)	64.54	53.42

Future Work

- Increase the number of problems solved (robustness).
- Improve the CPU time.
- Use iterative solvers (linear systems) and Hessian secant updates.

Future Work

- Increase the number of problems solved (robustness).
- Improve the CPU time.
- Use iterative solvers (linear systems) and Hessian secant updates.

Future Work

- Increase the number of problems solved (robustness).
- Improve the CPU time.
- Use iterative solvers (linear systems) and Hessian secant updates.

ipfilter Software

- The current ipfilter version (0.2) requires first and second-order derivatives and is only available for problems where $n \geq m$.
- ipfilter is freely available for academic and research purposes.
- ipfilter web page: http://www.mat.uc.pt/ipfilter.