Filters: an efficient tool for nonlinear programming

Philippe Toint[1] Nick Gould[2] Sven Leyffer[3] Caroline Sainvitu [1]

[1]Department of Mathematics,University of Namur, Belgium

(philippe.toint@fundp.ac.be)

[2]Rutherford Appleton Laboratory, UK
[3]Argonne National Laboratory, USA

4th Joint OR Days in Lausanne, September 2006

Outline

Outline

Outline

1. Introduction to filter methods
   - The monotonicity issue
   - The filter in constrained optimization
2. Feasibility and least-squares problems
   - The multidimensional filter
   - Numerical experience with FILTRANE
3. Unconstrained optimization
   - A filter for unconstrained optimization
   - Numerical experience
   - Approximate derivatives
4. Bound constrained problems
   - A filter projection method
   - A filter barrier method

Outline

Introduction to filter methods

Constrained optimization

The general nonlinear programming problem:

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & c_{\mathcal{E}}(x) = 0 \\ & c_{\mathcal{I}}(x) \geq 0, \end{array}$$

for $x \in \mathbb{R}^n$, f and c smooth.

Solution algorithms are

- iterative ($\{x_k\}$)
- based on Newton's method (or variant)

$\Rightarrow$ global convergence issues

Monotonicity (1)

Most often, global convergence is theoretically ensured by

- some global measure . . .
  - unconstrained : $f(x_k)$
  - constrained : merit function at x_k
- . . . with strong monotonic behaviour

(Lyapunov function)

Also practically enforced by

- algorithmic safeguards around Newton method (linesearches , trust regions)

Monotonicity (2)

However

the classical safeguards limit algorithmic efficiency!

Question of interest :

design less obstructive safeguards

while

- ensuring better numerical performance
  ($\Rightarrow$ the Newton Liberation Front !)
- continuing to guarantee global convergence properties

Non-monotone methods

Typically:

- abandon strict monotonicity of usual measures
- but insist on average behaviour

linesearch:

- Chamberlain, Powell, Lemarechal, Pedersen (1982)
- Grippo, Lampariello, Lucidi, Facchinei (1986, 1989, 1991, 1992,. . .)
- Panier, Tits, Bonnans, Zhou (1991, 1992), T. (1996), . . .

trust region:

- Deng, Xiao, Zhou (1992, 1993, 1994, 1995)
- T. (1994, 1997), Conn, Gould, T. (2000)
- Ke, Han, Liu (1995, 1996), Burke, Weigmann (1997), Yuan (1999), . . .

Non-monotone methods

Typically:

- abandon strict monotonicity of usual measures
- but insist on average behaviour

linesearch:

- Chamberlain, Powell, Lemarechal, Pedersen (1982)
- Grippo, Lampariello, Lucidi, Facchinei (1986, 1989, 1991, 1992,. . .)
- Panier, Tits, Bonnans, Zhou (1991, 1992), T. (1996), . . .

trust region:

- Deng, Xiao, Zhou (1992, 1993, 1994, 1995)
- T. (1994, 1997), Conn, Gould, T. (2000)
- Ke, Han, Liu (1995, 1996), Burke, Weigmann (1997), Yuan (1999), . . .

Non-monotone trust-regions

The main idea:

$$f(x_{k+1}) < f(x_k) \text{ replaced by } f(x_{k+1}) < f_{r(k)}$$

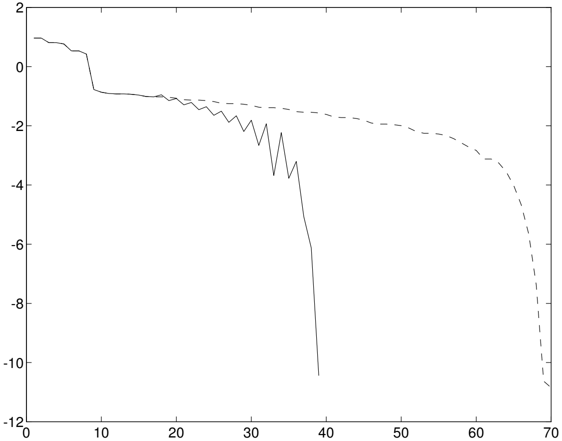
with

$$f_{r(k)} < f_{r(k-1)}$$

Further issues:

- suitably define $r(k)$
- adapt the trust-region algorithm:
  also compare achieved and predicted reductions since reference iteration

An unconstrained example



Monotone and non-monotone TR

A code: LANCELOT B

Introducing the filter

Constrained optimization :

use the Sequential Quadratic Programming step s_k from x_k

Ideally

- reduce the objective function $f(x)$
- reduce the constraint violation $\theta(x)$

☺ two potentially conflicting aims ☹

▶ Filter method

Introducing the filter

Constrained optimization :

use the Sequential Quadratic Programming step s_k from x_k

Ideally

- reduce the objective function $f(x)$
- reduce the constraint violation $\theta(x)$

☺ two potentially conflicting aims ☹

▶ Filter method

Introducing the filter

Constrained optimization :

use the Sequential Quadratic Programming step s_k from x_k

Ideally

- reduce the objective function $f(x)$
- reduce the constraint violation $\theta(x)$

☺ two potentially conflicting aims ☹

▶ Filter method

Accepting a new iterate

Idea of Fletcher and Leyffer

Replace the question

What is a better point ?

by

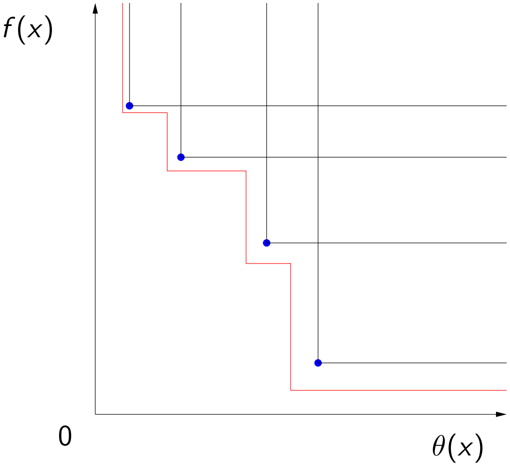
What is a worse point ?

Of course, y is "worse" than x if it is dominated by x, i.e., when

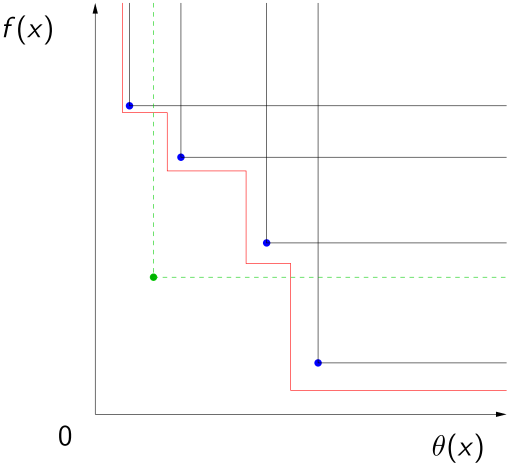
$$f(x) \leq f(y) \quad \text{and} \quad \theta(x) \leq \theta(y)$$

Accept or reject a trial point ?

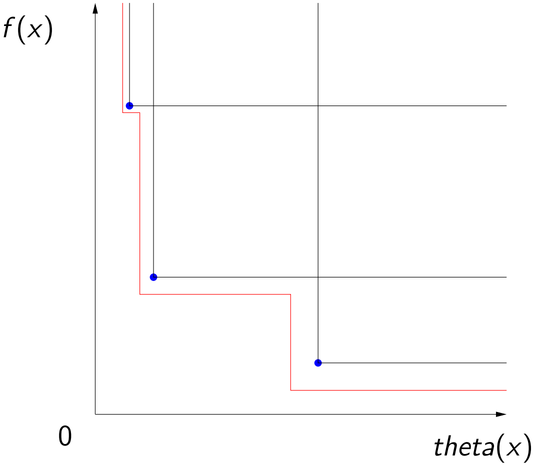
View of a standard filter



View of a standard filter

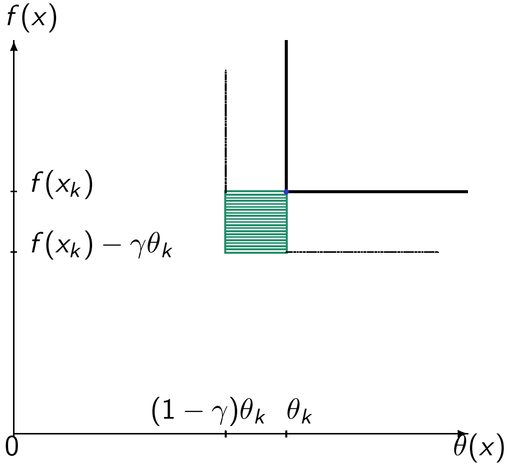


View of a standard filter



Filling up the standard filter

Note: filter area is bounded in the (f, θ) space!



$\Rightarrow$ filter area (non)-monotonically decreasing

Handling many objectives

Gould, Leyffer and T.

Feasibility

Find x such that

$$c_{\mathcal{E}}(x) = 0 \quad \text{and} \quad c_{\mathcal{I}}(x) \geq 0$$

Least-squares

Find x such that

$$\min_{x \in \mathbf{R}^n} \sum_{i \in \mathcal{E} \cup \mathcal{I}} \theta_i(x)^2$$

▶ More dimensions in the filter space . . .

Handling many objectives

Gould, Leyffer and T.

Feasibility

Find x such that

$$c_{\mathcal{E}}(x) = 0 \quad \text{and} \quad c_{\mathcal{I}}(x) \geq 0$$

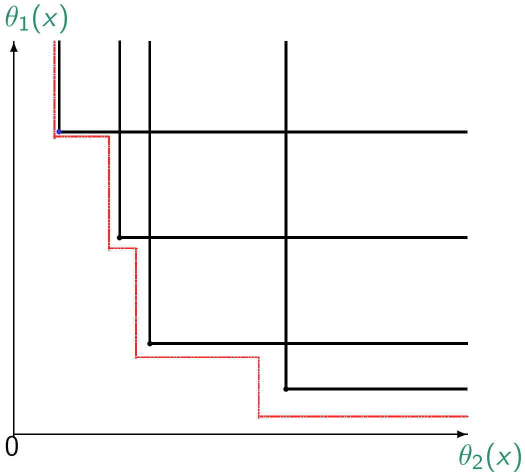
Least-squares

Find x such that

$$\min_{x \in \mathbf{R}^n} \sum_{i \in \mathcal{E} \cup \mathcal{I}} \theta_i(x)^2$$

▶ More dimensions in the filter space . . .

The obvious picture



(full dimension vs. grouping)

Context for the multidimensional filter

In a real algorithm, additionally

- possibly consider unsigned filter entries
- use TR algorithm when
  - trial point unacceptable
  - convergence to non-zero solution

($\Rightarrow$ "internal" restoration)

sound convergence theory

Numerical experience: FILTRANE

The FILTRANE package

- standard Fortran 95
- large scale problems (CUTEr interface)
- includes several variants of the method
  - signed/unsigned filters
  - Gauss-Newton, Newton or adaptive models
  - pure trust-region option
  - uses preconditioned conjugate-gradients + Lanczos for subproblem solution
- part of the GALAHAD library

Main features

From a large body of numerical experiments, two main advantages:

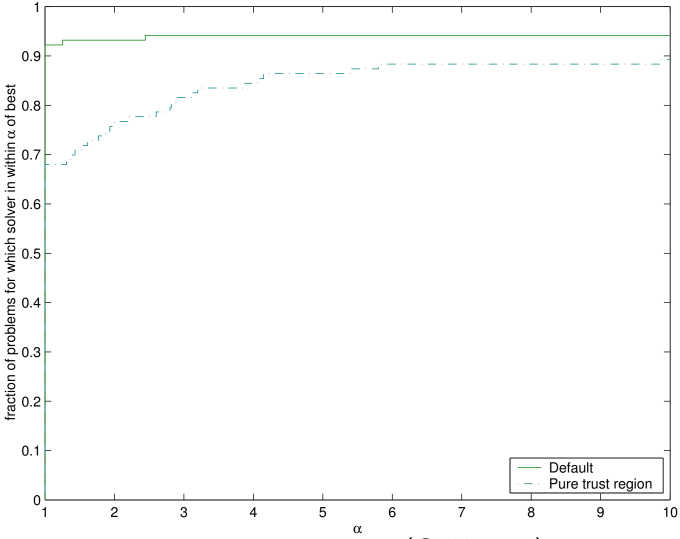
we do not require that $\|s_k\| \leq \Delta_k$ at every iteration

only restrict steps when unrestricted ones are not acceptable

we do not impose monotonicity

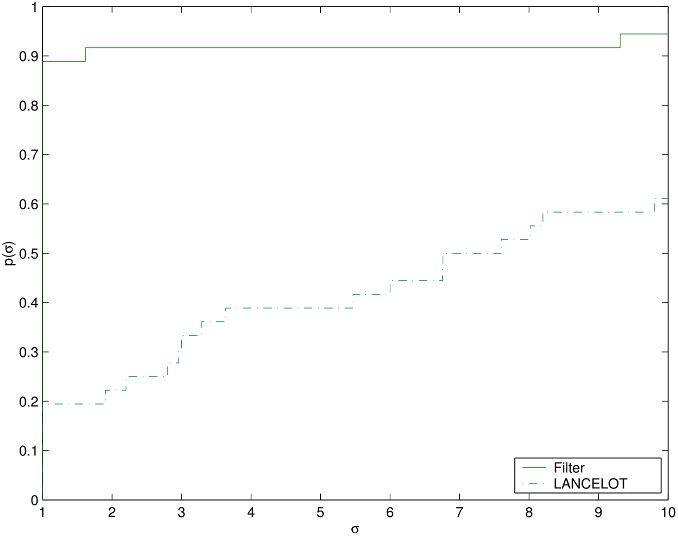
see "around corners"

Numerical experience (1)



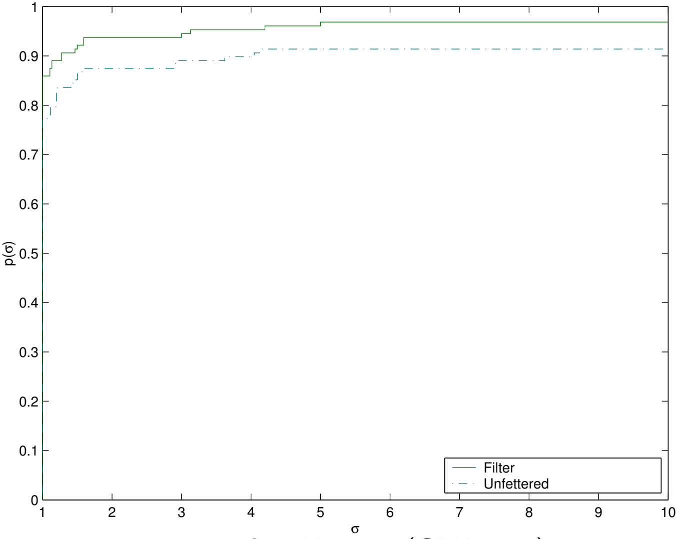
Filter vs. trust-region (CPU time)

Numerical experience (2)



Filter vs. LANCELOT B (CPU time)

Numerical experience (3)



Filter vs. free Newton (CPU time)

The other extreme case

Gould, Sainvitu and T.

Unconstrained optimization

$$\min_{x \in \mathbf{R}^n} \; f(x)$$

Simple-bound constrained optimization

$$\begin{array}{ll} \min & f(x) \\ \text{s.t.} & l \leq x \leq u \end{array}$$

Combined methods:

Trust-region + filter + projected gradient

▶ Multidimensional filter

Unconstrained trust-region methods

$$\min_{x \in \mathbb{R}^n} f(x)$$

- Newton's method $\rightarrow \min_s \quad f(x_k) + \nabla_x f(x_k)^T s + \frac{1}{2} s^T H_k s$
- Trust-region method
- Filter technique

Idea: encourage convergence to first-order critical points by driving every component of the objective's gradient

$$\nabla_x f(x) \stackrel{\text{def}}{=} g(x) = (g_1(x), \ldots, g_n(x))^T$$

to zero.

▶ Multidimensional Filter

Unconstrained trust-region methods

$$\min_{x \in \mathbb{R}^n} f(x)$$

- Newton's method $\rightarrow \min_s \quad f(x_k) + \nabla_x f(x_k)^T s + \frac{1}{2} s^T H_k s$
- Trust-region method
- Filter technique

Idea: encourage convergence to first-order critical points by driving every component of the objective's gradient

$$\nabla_x f(x) \stackrel{\text{def}}{=} g(x) = (g_1(x), \ldots, g_n(x))^T$$

to zero.

▶ Multidimensional Filter

A gradient multidimensional filter

Accept x_k^+ more often

A point x dominates a point y whenever

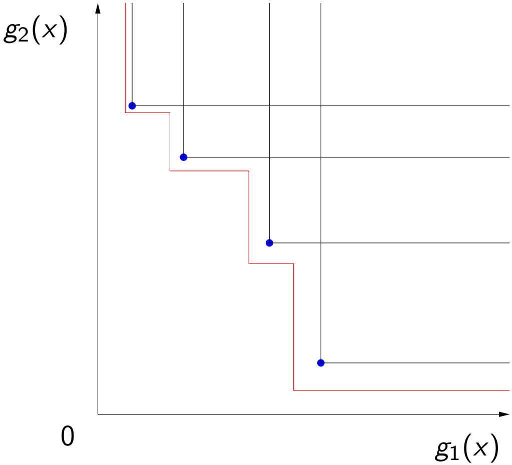
$$|g_i(x)| \leq |g_i(y)| \quad \forall\, i = 1, \ldots, n$$

Remember non-dominated points $\quad \Rightarrow \quad$ FILTER

Accept a new iterate ?

if it is not dominated by any other iterate in the filter

Haven't we seen this before?



A few complications. . .

But . . .

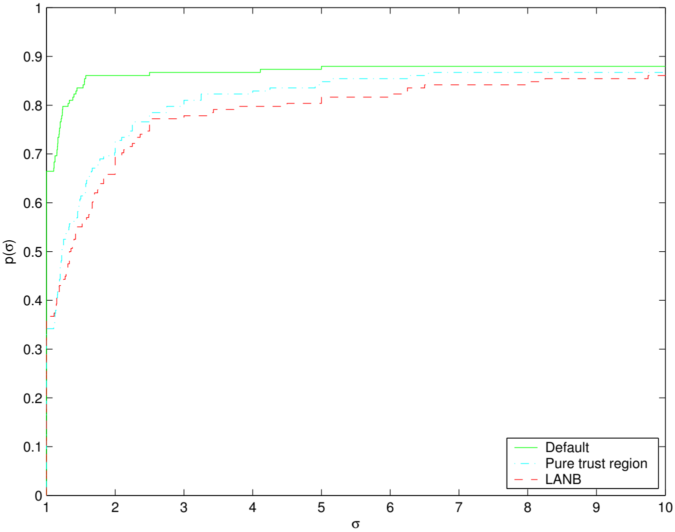
$g(x) = 0$ not sufficient for nonconvex problems!

When negative curvature found:

- reset filter
- set upper bound on acceptable $f(x)$

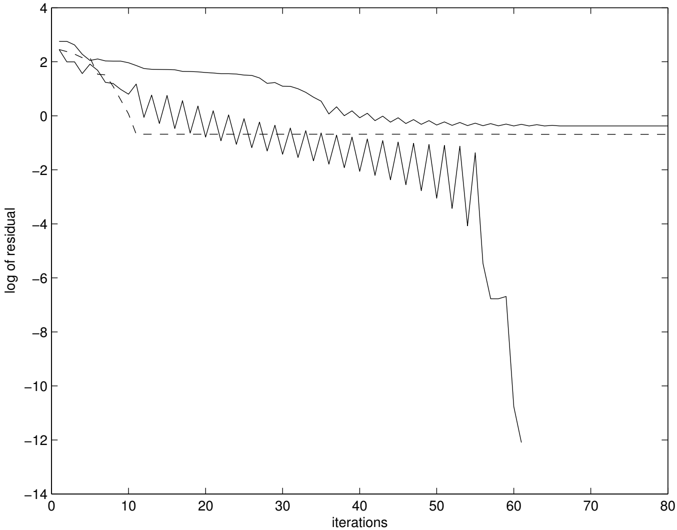
reasonable convergence theory

Numerical experience (1)



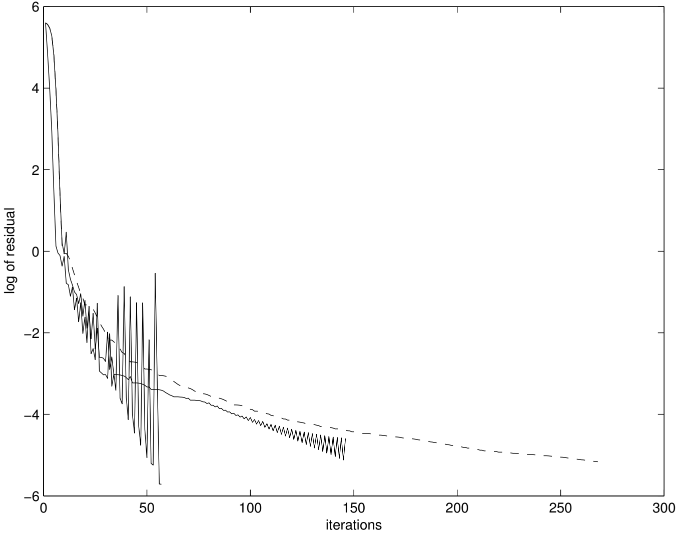
Filter vs. trust-region and LANCELOT B (iterations)

Numerical experience: HEART6



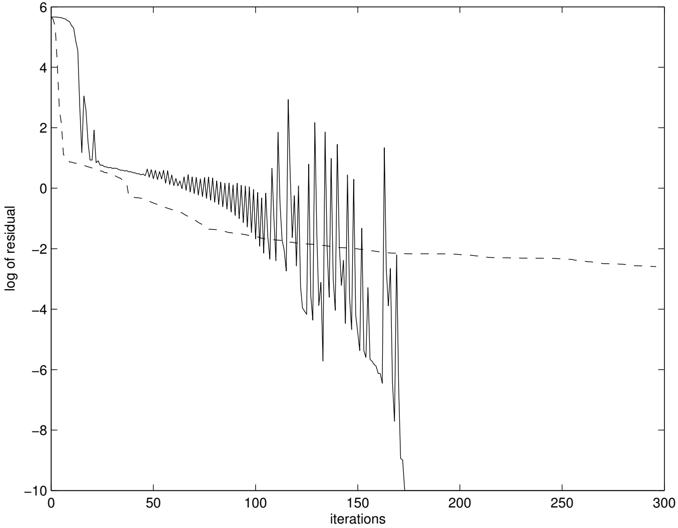
Filter vs. trust-region and LANCELOT B

Numerical experience: EXTROSNB



Filter vs. trust-region and LANCELOT B

Numerical experience: LOBSTERZ



Filter vs. trust-region

Filter and approximate derivatives: a problem?

- filter-trust-region algorithm requires knowledge of first and second derivatives
- derivatives calculated analytically and supplied by the user
- unavailable or computationally expensive

The question

Is the behaviour of the algorithm directly related to the use of exact derivatives ?

Filter and approximate derivatives: a problem?

- filter-trust-region algorithm requires knowledge of first and second derivatives
- derivatives calculated analytically and supplied by the user
- unavailable or computationally expensive

The question

Is the behaviour of the algorithm directly related to the use of exact derivatives ?

The role of derivatives derivatives

Where ?

- definition of the model of the objective-function

$$m_k(s) = f(x_k) + g_k{}^T s + \frac{1}{2} s^T H_k s$$

  - ▶ computation of the next trial point
- Gradient
  - definition of the filter
  - filter test acceptance mechanism
  - stopping criteria
- Hessian
  - decision to use the filter technique or not
  - restricted steps

Which approximate derivatives?

Two different ways to approximate
$\nabla_x f(x)$ and $\nabla_{xx} f(x)$

$\swarrow$ $\searrow$

finite-difference approximations
to the gradient
and/or the Hessian

quasi-Newton approximations
to the Hessian
(BFGS, SR1 updates)

Quasi Newton approximations

- Broyden-Fletcher-Goldfarb-Shanno (BFGS)

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k}$$

- Symmetric rank-one (SR1)

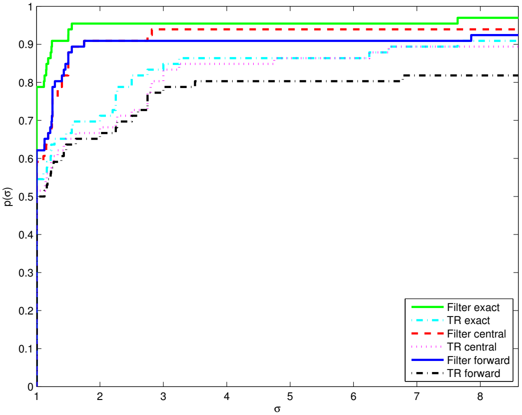
$$B_{k+1} = B_k + \frac{(y_k - B_k s_k)(y_k - B_k s_k)^T}{(y_k - B_k s_k)^T s_k}$$

where $s_k = x_{k+1} - x_k$ and $y_k = \nabla_x f(x_{k+1}) - \nabla_x f(x_k)$

▶ different initial Hessian approximations B_0

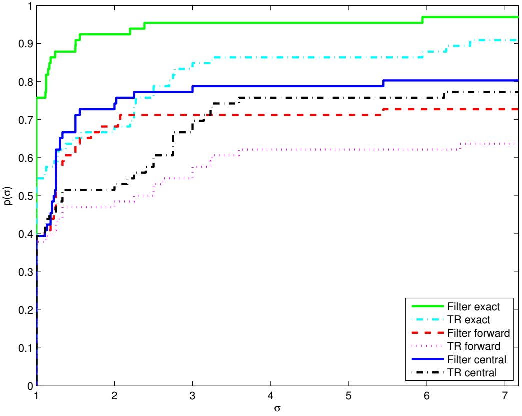
Numerical results : finite-difference H - analytic g

Iterations performance profile



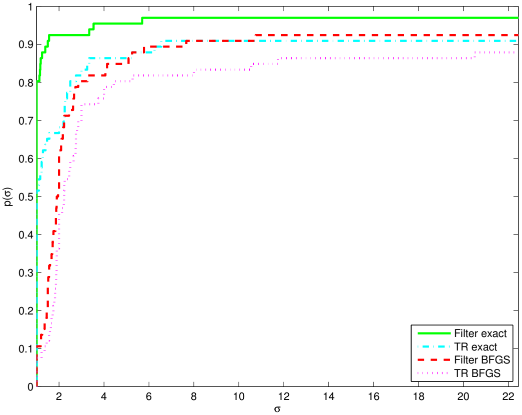
Numerical results : finite-difference H and g

Iterations performance profile



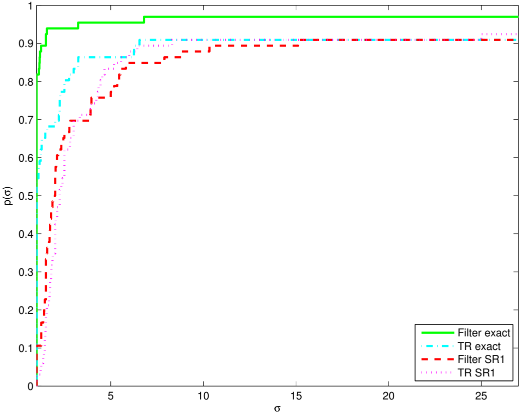
Numerical results : BFGS update

Iterations performance profile



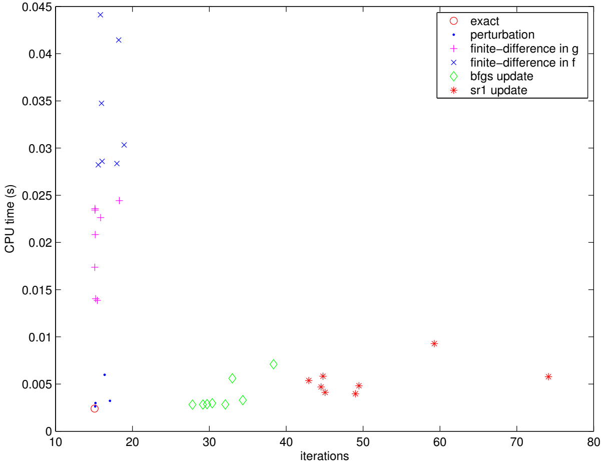
Numerical results : SR1 update

Iterations performance profile

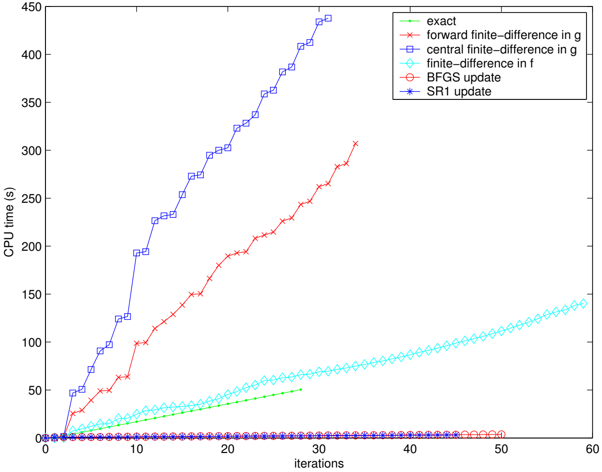


Comparison of quasi-Newton updates

Combined performance



Numerical results : STRATEC (nested logit model)



Bound constraints

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \geq 0, \end{array}$$

(Also applies to convex constraints)

Two approaches:

- projection methods
- interior-point (barrier) methods

A projection method

Consider the bound constrained problem:

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{s.t.} & l \leq x \leq u \end{array}$$

A combined algorithm:

Trust region + filter + projected gradient

Simple idea : Replace the gradient components of the multidimensional filter for unconstrained optimization by the components of

$$\bar{g}(x) \stackrel{\text{def}}{=} x - P[x - \nabla_x f(x), l, u]$$

where P is the projection onto the feasible domain.

▶ drive these components to zero!

A projection method

Consider the bound constrained problem:

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{s.t.} & l \leq x \leq u \end{array}$$

A combined algorithm:

Trust region + filter + projected gradient

Simple idea : Replace the gradient components of the multidimensional filter for unconstrained optimization by the components of

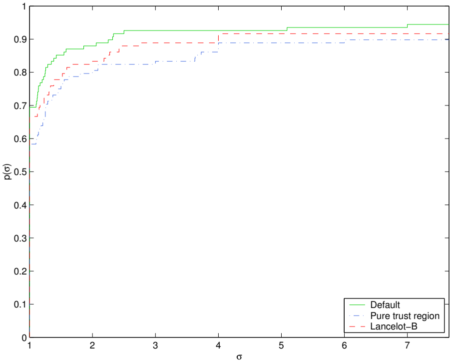
$$\overline{g}(x) \stackrel{\text{def}}{=} x - P[x - \nabla_x f(x), l, u]$$

where P is the projection onto the feasible domain.

- drive these components to zero!

Numerical results

Iterations



Filter - Trust-region - LANCELOT B on 108 CUTEr problems

Another option

$$\text{minimize} \quad f(x) - \mu \log(x)$$

for a sequence of $\mu \searrow 0$.

Question:

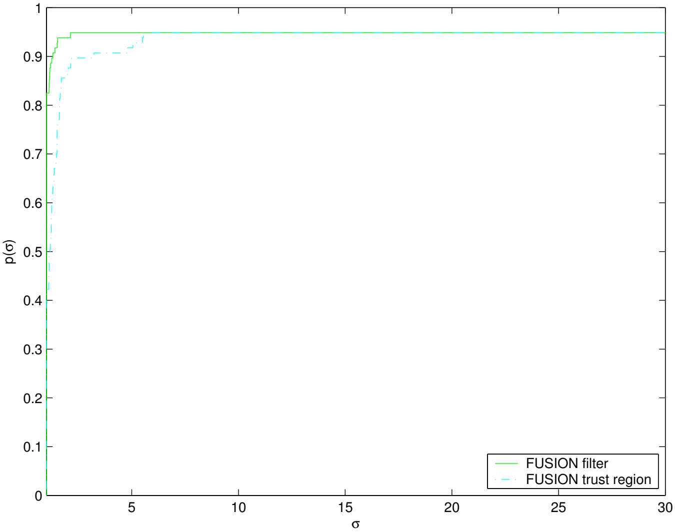
Does filter improve the sequence of unconstrained subproblems?

Issues:

- specific nonlinearity
- (very) approximate solutions

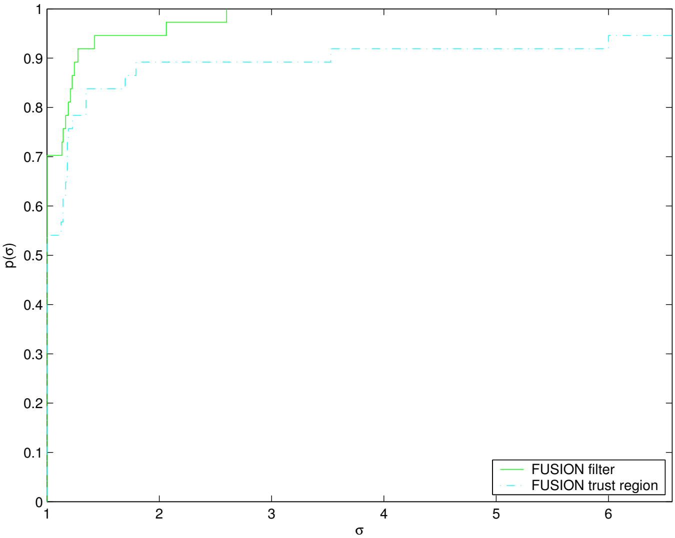
A package (still being developed): FUSION

Preliminary results (1)



Filter vs. trust-region (iterations, 97 CUTEr problems)

Preliminary results (2)



Filter vs. trust-region (CPU time, 37 CUTEr problems)

Conclusion

Not discussed:

- filter and pattern search
- filter and the central path

In the works (as far as I know):

- filter and singular problems
- filter and the complementarity problem
- filter and equality constrained optimization
- filter and negative curvature

Thank you for your attention !

(see `http://perso.fundp.ac.be/~phtoint/publications.html` for references)