NEWUOAs: A Derivative-free Optimization Algorithm Based on Powell's NEWUOA and Subspace Techniques (Is it possible to solve a 20000-dimensional optimization problem without using derivatives on a laptop?)

ZHANG Zaikun

Hong Kong Polytechnic University

06 August 2016, ICNAAO, Beijing

In deep memory of Professor M. J. D. Powell

Why optimize a function without using derivatives?

I started to write computer programs in Fortran at Harwell in 1962. ... after moving to Cambridge in 1976 ... I became a consultant for IMSL. One product they received from me was the `TOLMIN` package for optimization ... which requires first derivatives ... Their customers, however, prefer methods that are without derivatives, so IMSL forced my software to employ difference approximations ... I was not happy ... Thus there was strong motivation to try to construct some better algorithms.

— M. J. D. Powell

A view of algorithms for optimization without derivatives, 2007

Because it is important and cool

- Why work on derivative-free optimization?
- Because the problems are important and cool.

— J. E. Dennis, Jr.
Reasons to study derivative-free algorithms, 2013, Toulouse, France

Derivative-free optimization (DFO)

- Derivative-free optimization (DFO)
  - Minimize a function f using function values but not derivatives.
  - A typical case: f is a black box without an explicit formula.

$$x \longrightarrow \boxed{f} \longrightarrow f(x)$$

Derivative-free optimization (DFO)

- Derivative-free optimization (DFO)
  - Minimize a function f using function values but not derivatives.
  - A typical case: f is a black box without an explicit formula.

$$x \longrightarrow \boxed{f} \longrightarrow f(x)$$

- Various applications
  - S. Gratton, P. Laloyaux, A. Sartenaer, ‘Derivative-free optimization for large-scale nonlinear data assimilation problems’, 2014
  - S. Wild, J. Sarich, N. Schunck, ‘Derivative-free optimization for parameter estimation in computational nuclear physics’, 2015

Existing methods

Many derivative-free methods have been developed:

- Trust region methods: `BC-DFO`, `DFO`, `NEWUOA`, ...
- Direct-search type methods: `BFO`, `GPS`, `NOMAD`, ...

How large is large?

> Perhaps foremost among the limitations of derivative-free methods is that, on a serial machine, it is usually not reasonable to try and optimize problems with more than a few tens of variables, although some of the most recent techniques (`NEWUOA`) can handle unconstrained problems in hundreds of variables.

— A. R. Conn, K. Scheinberg, L. N. Vicente

Introduction to Derivative-Free Optimization, 2007

How large is large?

LINCOA is not suitable for very large numbers of variables because no attention is given to any sparsity. A few calculations with 1000 variables, however, have been run successfully overnight, and the performance of LINCOA is satisfactory usually for small numbers of variables.

— M. J. D. Powell

Comments to LINCOA, December 2013

Subspace techniques in optimization

- N. Gould, A. Sartenaer, Ph. L. Toint. 'On iterated-subspace minimization methods for nonlinear optimization', 1994.
- Y. Yuan, 'Subspace techniques for nonlinear optimization', 2007
- Block coordinate descent
- ...

A framework of subspace algorithms

Algorithm

Step 1. Pick the starting point x_0. $k := 0$.

A framework of subspace algorithms

Algorithm

Step 1. Pick the starting point x_0. $k := 0$.

Step 2. Pick a subspace $\mathcal{S}_k$ of $\mathbb{R}^n$.

A framework of subspace algorithms

Algorithm

Step 1. Pick the starting point x_0. $k := 0$.

Step 2. Pick a subspace $\mathcal{S}_k$ of $\mathbb{R}^n$.

Step 3. Solve the subspace subproblem

$$\min_{d \in \mathcal{S}_k} f(x_k + d)$$

exactly or approximately, obtaining d_k.

A framework of subspace algorithms

Algorithm

Step 1. Pick the starting point x_0. $k := 0$.

Step 2. Pick a subspace $\mathcal{S}_k$ of $\mathbb{R}^n$.

Step 3. Solve the subspace subproblem

$$\min_{d \in \mathcal{S}_k} f(x_k + d)$$

exactly or approximately, obtaining d_k.

Step 4. $x_{k+1} := x_k + d_k$ and $k := k + 1$. Goto Step 2.

Convergence

Theorem

Suppose that

- $\mathrm{dist}(\nabla f(x_k), \mathcal{S}_k)$ is sufficiently small, and
- d_k is sufficiently exact,

then the subspace algorithm will converge (sufficiently fast).

Convergence

Theorem

Suppose that

- $\mathrm{dist}(\nabla f(x_k), \mathcal{S}_k)$ is sufficiently small, and
- d_k is sufficiently exact,

then the subspace algorithm will converge (sufficiently fast).

All we need is

- a good model of f around x_k, and
- a good solver for $\min_{d \in \mathcal{S}_k} f(x_k + d)$.

We do not need derivatives.

A practical derivative-free subspace algorithm: NEWUOAs

- The subspace:

$$\mathcal{S}_k = \text{span}\{-g_k,\, d_{k-1}\},$$

where

$$g_k = \nabla m_k(x_k),$$

m_k being the model at x_k defined by the methodology of NEWUOA.

Ref: Y. Yuan, J. Stoer, 'A Subspace Study on Conjugate Gradient Algorithms', 1995

- The subspace solver: NEWUOA.

$$\texttt{NEWUOAs} = \texttt{NEWUOA} + \text{subspace}$$

The name of the game

From: Mike Powell <M.J.D.Powell@damtp.cam.ac.uk>
Date: 2012-05-22 17:16 GMT+08:00
Subject: Re: Paper on Sobolev Seminorm
To: M.J.D.Powell@damtp.cam.ac.uk, zhangzk@lsec.cc.ac.cn
Cc: yyx@lsec.cc.ac.cn, zaikunzhang@gmail.com

Dear Zaikun,

...Congratulations on finishing your thesis. ...It is often difficult to choose a name for a new algorithm, and NEWUOAs does have some advantages -- there is no need for my permission. ...

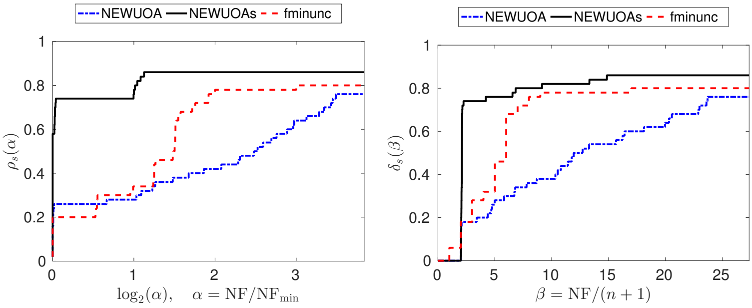
With best wishes,

Grandpa.

‘Perfect’ problems

- Test ptoblems
  - We take 50 unconstrained problems from the `CUTEr` set
  - These problems are smooth and bounded from below
  - The dimensions of these problems are changable
- Performance measures
  - We use the Performance Profile and Data Profile

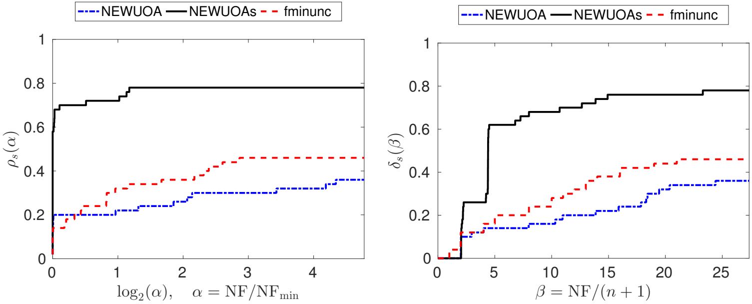
'Perfect' problems



(a) Performance profile (b) Data profile

Figure: `NEWUOA`, `NEWUOAs`, and `fminunc` ($n = 200$, $\tau = 10^{-2}$)

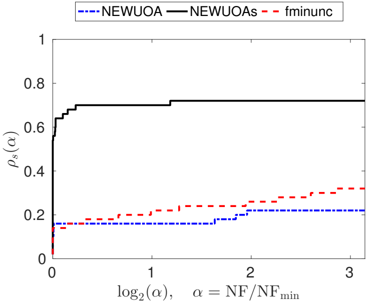
‘Perfect’ problems



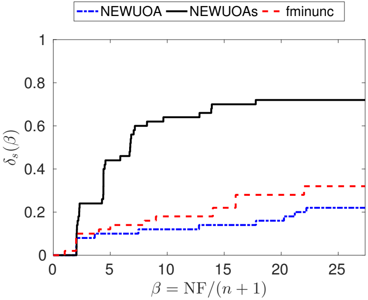
(a) Performance profile (b) Data profile

Figure: `NEWUOA`, `NEWUOAs`, and `fminunc` $(n = 200,\ \tau = 10^{-4})$

'Perfect' problems



(a) Performance profile



(b) Data profile

Figure: `NEWUOA`, `NEWUOAs`, and `fminunc` $(n = 200,\ \tau = 10^{-6})$

Noisy problems

- We still want to solve the previous 50 `CUTEr` problems.
- However, for each objective function f, we have access only to

$$f_e(x) \; = \; f(x)(1+e), \quad \text{with} \quad e \sim N(0, \sigma^2).$$

- In our experiment, $\sigma = 10^{-3}$.

Noisy problems

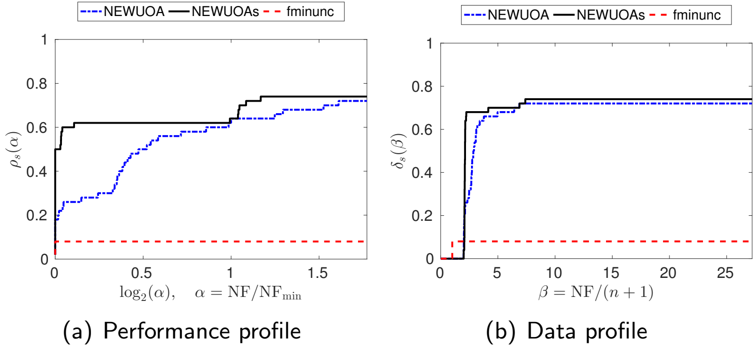
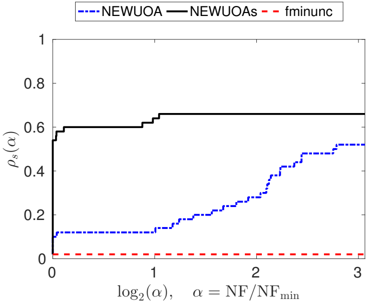
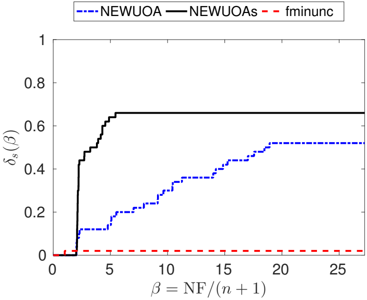


Figure: `NEWUOA`, `NEWUOAs`, and `fminunc` ($n = 100$, $\tau = 10^{-1}$)

Noisy problems



(a) Performance profile



(b) Data profile

Figure: `NEWUOA`, `NEWUOAs`, and `fminunc` $(n = 100,\ \tau = 10^{-2})$

Noisy problems

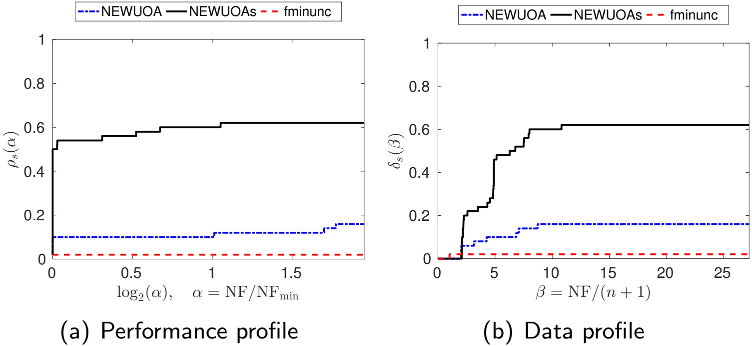
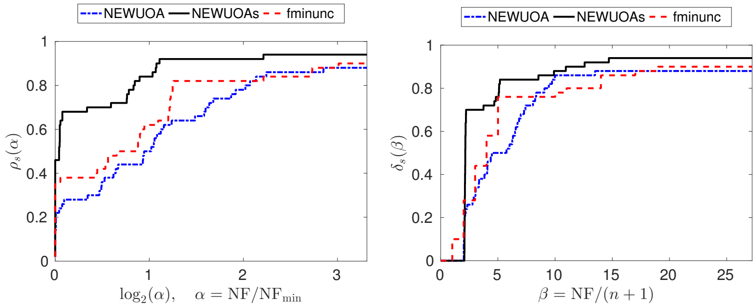


Figure: `NEWUOA`, `NEWUOAs`, and `fminunc` $(n = 100,\ \tau = 10^{-3})$

Nonsmooth problems

- We minimize
$$F(x) \; = \; f(x) + \lambda \|x\|_1$$
for each f of the previous 50 `CUTEr` problems.
- In our experiment, $\lambda = 10$.

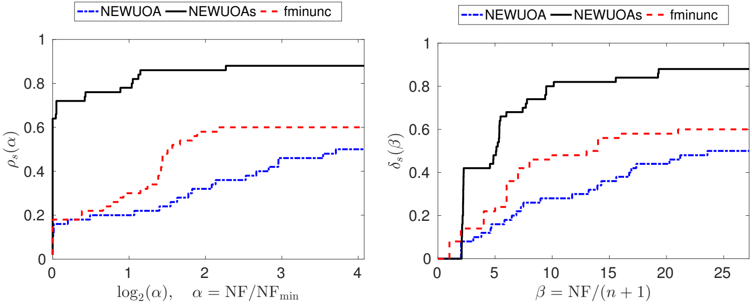
Nonsmooth problems



(a) Performance profile (b) Data profile

Figure: `NEWUOA`, `NEWUOAs`, and `fminunc` $(n = 100,\ \tau = 10^{-1})$

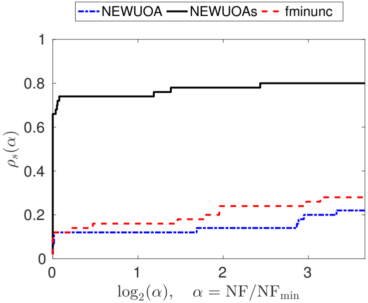
Nonsmooth problems



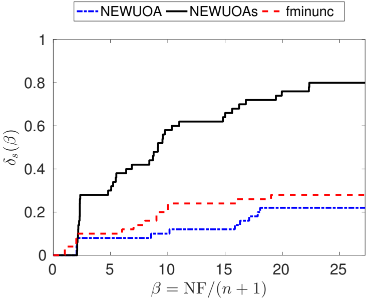
(a) Performance profile (b) Data profile

Figure: `NEWUOA`, `NEWUOAs`, and `fminunc` $(n = 100,\ \tau = 10^{-2})$

Nonsmooth problems



(a) Performance profile



(b) Data profile

Figure: `NEWUOA`, `NEWUOAs`, and `fminunc` $(n = 100,\ \tau = 10^{-3})$

Large problems

Table: The performance of NEWUOAs on some 20000-dimensional problems

	f_{start}	f_{best}	$\#f/n$	CPU (s)
ARWHEAD	5.999700E+04	0.000000E+00	8.0	32.1
CHROSEN	3.999800E+10	1.100760E−10	14.1	78.1
SPARSQUR	5.627812E+07	2.352091E−28	8.0	63.2

Conclusion

Use your information to choose a subspace before doing optimization.