A Derivative-Free Optimization Algorithm with Low-Dimensional Subspace Techniques for Large-Scale Problems

ZHANG, Zaikun
University of Coimbra
www.zhangzk.net

July 29, 2013 — ICCOPT, Lisbon

Outline

1. Derivative-free optimization

1. Derivative-free optimization

- **The problem**

In this talk, to make things simple, we consider unconstrained optimization problem

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

1. Derivative-free optimization

- **The problem**

  In this talk, to make things simple, we consider unconstrained optimization problem

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

  - Most of the current methods depend on derivatives.

1. Derivative-free optimization

- **The problem**

  In this talk, to make things simple, we consider unconstrained optimization problem

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

  - Most of the current methods depend on derivatives.
  - In many real-world problems, the derivatives are unavailable.

1. Derivative-free optimization

- **The problem**

  In this talk, to make things simple, we consider unconstrained optimization problem

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

  - Most of the current methods depend on derivatives.
  - In many real-world problems, the derivatives are unavailable.
  - Suppose that

1. Derivative-free optimization

- **The problem**

  In this talk, to make things simple, we consider unconstrained optimization problem

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

  - Most of the current methods depend on derivatives.
  - In many real-world problems, the derivatives are unavailable.
  - Suppose that
    - f is smooth, but the derivatives are unavailable;

1. Derivative-free optimization

- **The problem**

  In this talk, to make things simple, we consider unconstrained optimization problem

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

  - Most of the current methods depend on derivatives.
  - In many real-world problems, the derivatives are unavailable.
  - Suppose that
    - f is smooth, but the derivatives are unavailable;
    - the function evaluation of f is expensive.

1. Derivative-free optimization

- **Importance and difficulty**

> *We consider optimization without derivatives one of the most important, open, and challenging areas in computational science and engineering, and one with enormous practical potential.*

— A. R. Conn, K. Scheinberg, L. N. Vicente
Introduction to Derivative-Free Optimization

1. Derivative-free optimization

- **Importance and difficulty**

We consider optimization without derivatives one of the most important, open, and challenging areas in computational science and engineering, and one with enormous practical potential.

— A. R. Conn, K. Scheinberg, L. N. Vicente
Introduction to Derivative-Free Optimization

1. Derivative-free optimization

- **The objective**

  Develop optimization algorithms that

1. Derivative-free optimization

- **The objective**

  Develop optimization algorithms that

  - do not use derivatives;

- **The objective**

  Develop optimization algorithms that

  - do not use derivatives;
  - use function evaluations as less as possible.

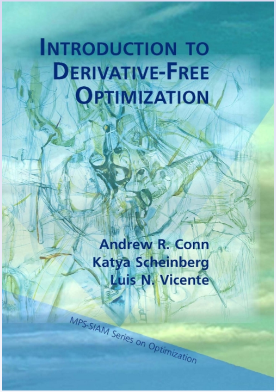
- **Existing methods**
  - Two main classes of rigorous methods in DFO:

- **Existing methods**
  - Two main classes of rigorous methods in DFO:
    - directional methods, like direct search;

1. Derivative-free optimization

- **Existing methods**
  - Two main classes of rigorous methods in DFO:
    - directional methods, like direct search;
    - model-based methods, like trust-region methods.

1. Derivative-free optimization



A. R. Conn, K. Scheinberg, and L. N. Vicente, Introduction to Derivative-Free Optimization, MOS-SIAM Series on Optimization, SIAM, Philadelphia, 2009

1. Derivative-free optimization

- **Interpolation-based trust-region methods**

1. Derivative-free optimization

- **Interpolation-based trust-region methods**
  - Trust region technique:
    Minimize Model Function
    Subject to Trust Region.

1. Derivative-free optimization

- **Interpolation-based trust-region methods**
  - Trust region technique:

    Minimize Model Function

    Subject to Trust Region.
  - Model construction:

    $$\text{Objective Function} \xrightarrow{\text{Interplolation}} \text{Model Function.}$$

1. Derivative-free optimization

- **Underdetermined interpolation**

1. Derivative-free optimization

- **Underdetermined interpolation**

  - Quadratic interpolation:

$$Q(x) = f(x), \quad x \in \mathcal{I}.$$

  - A linear system with degree of freedom $(n+1)(n+2)/2$.

1. Derivative-free optimization

- **Underdetermined interpolation**
  - Quadratic interpolation:

$$Q(x) = f(x), \quad x \in \mathcal{I}.$$

  - A linear system with degree of freedom $(n+1)(n+2)/2$.
  - What if $|\mathcal{I}| < (n+1)(n+2)/2$ (for example, $|\mathcal{I}| = O(n)$)?

- **Underdetermined interpolation**
  - Quadratic interpolation:

$$Q(x) = f(x), \quad x \in \mathcal{I}.$$

  - A linear system with degree of freedom $(n+1)(n+2)/2$.
  - What if $|\mathcal{I}| < (n+1)(n+2)/2$ (for example, $|\mathcal{I}| = O(n)$)?
  - Regularization:

$$\min_{Q \in \mathcal{Q}} \mathscr{F}(Q)$$
$$\text{s.t. } Q(x) = f(x), \quad x \in \mathcal{I}.$$

2. A framework of subspace algorithms

2. A framework of subspace algorithms

- **Why subspace?**

  To solve large-scale problems ($\sim$1000).

2. A framework of subspace algorithms

- **Why subspace?**

  To solve large-scale problems ($\sim$1000).

  Not easy —

2. A framework of subspace algorithms

- **Why subspace?**

  To solve large-scale problems ($\sim$1000).

  Not easy —

  - quadratic-model-based methods:
    - in principle, the degree of freedom of a full quadratic model is $(n+1)(n+2)/2$;
    - in practice, we hope the algorithms finish the job with number of function evaluations of $O(n)$.

2. A framework of subspace algorithms

- **Why subspace?**

  To solve large-scale problems ($\sim$1000).

  Not easy —

  - quadratic-model-based methods:
    - in principle, the degree of freedom of a full quadratic model is $(n+1)(n+2)/2$;
    - in practice, we hope the algorithms finish the job with number of function evaluations of $O(n)$.
  - difficult to exploit the structure.

2. A framework of subspace algorithms

- **Basic idea**

2. A framework of subspace algorithms

- **Basic idea**
  - Solve a difficult problem by solving a sequence of easy problems;

2. A framework of subspace algorithms

- **Basic idea**
  - Solve a difficult problem by solving a sequence of easy problems;
  - solve a large problem by solving a sequence of small problems.

- **Subspace techniques in optimization**
  - Yuan, Ya-xiang. Subspace techniques for nonlinear optimization. Some topics in industrial and applied mathematics 8 (2007): 206-218.
  - Gould, Nick, A. Sartenaer, and Ph L. Toint. On iterated-subspace minimization methods for nonlinear optimization. Rutherford Appleton Laboratory, 1994.

- **Subspace techniques in optimization**
  - Yuan, Ya-xiang. Subspace techniques for nonlinear optimization. Some topics in industrial and applied mathematics 8 (2007): 206-218.
  - Gould, Nick, A. Sartenaer, and Ph L. Toint. On iterated-subspace minimization methods for nonlinear optimization. Rutherford Appleton Laboratory, 1994.
  - Coordinate-search ...

2. A framework of subspace algorithms

- **A general framework**

Algorithm 2.1

2. A framework of subspace algorithms

- **A general framework**

Algorithm 2.1

Step 1. Initialize. Pick the starting point x_1. $k := 1$.

2. A framework of subspace algorithms

- **A general framework**

Algorithm 2.1

Step 1. Initialize. Pick the starting point x_1. $k := 1$.

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

2. A framework of subspace algorithms

- **A general framework**

Algorithm 2.1

Step 1. Initialize. Pick the starting point x_1. $k := 1$.

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

Step 3. Solve the Subproblem. Solve the subspace subproblem

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

exactly or approximately, obtaining d_k.

2. A framework of subspace algorithms

- **A general framework**

Algorithm 2.1

Step 1. Initialize. Pick the starting point x_1. $k := 1$.

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

Step 3. Solve the Subproblem. Solve the subspace subproblem

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

exactly or approximately, obtaining d_k.

Step 4. Update the Iterate. If $f(x_k + d_k) < f(x_k)$, then $x_{k+1} := x_k + d_k$; else $x_{k+1} := x_k$. $k := k + 1$. Goto Step 2.

2. A framework of subspace algorithms

- **A general framework**

Algorithm 2.1

Step 1. Initialize. Pick the starting point x_1. $k := 1$.

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

Step 3. Solve the Subproblem. Solve the subspace subproblem

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

exactly or approximately, obtaining d_k.

Step 4. Update the Iterate. If $f(x_k + d_k) < f(x_k)$, then $x_{k+1} := x_k + d_k$; else $x_{k+1} := x_k$. $k := k + 1$. Goto Step 2.

- **Convergence analysis**

  - Global convergence
  - R-linear convergence rate

2. A framework of subspace algorithms

- **Convergence analysis**

- Global convergence
- R-linear convergence rate

$$\Uparrow$$

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

2. A framework of subspace algorithms

- **Convergence analysis**

- Global convergence
- R-linear convergence rate

$$\Uparrow$$

- $\operatorname{dist}(\nabla f(x_k), \mathcal{S}_k)$ is small enough $\Longleftarrow \mathcal{S}_k \ni \tilde{g}_k \approx \nabla f(x_k)$
- d_k is exact enough

2. A framework of subspace algorithms

- **Convergence analysis**

- Global convergence
- R-linear convergence rate

$$\Uparrow$$

- $\operatorname{dist}(\nabla f(x_k), \mathcal{S}_k)$ is small enough $\Longleftarrow \mathcal{S}_k \ni \tilde{g}_k \approx \nabla f(x_k)$
- d_k is exact enough $\Longleftarrow$ existing derivative-free algorithms

3. A practical subspace algorithm: NEWUOAs

3. NEWUOAs

- NEWUOAs

3. NEWUOAs

- NEWUOAs
  - $\mathcal{S}_k = \mathrm{span}\{\tilde{g}_k, s_{k-1}\}$.

3. NEWUOAs

- NEWUOAs
  - $\mathcal{S}_k = \text{span}\{\tilde{g}_k, s_{k-1}\}$.
    - Interpolation set $\mathcal{I}_k \Longrightarrow$ Quadratic model $Q_k \Longrightarrow$ Approximate gradient $\tilde{g}_k = \nabla Q_k(x_k)$.

3. NEWUOAs

- NEWUOAs
  - $\mathcal{S}_k = \mathrm{span}\{\tilde{g}_k, s_{k-1}\}$.
    - Interpolation set $\mathcal{I}_k \Longrightarrow$ Quadratic model $Q_k \Longrightarrow$ Approximate gradient $\tilde{g}_k = \nabla Q_k(x_k)$.
  - Subproblem solver: NEWUOA.

3. NEWUOAs

- NEWUOAs
  - $\mathcal{S}_k = \mathrm{span}\{\tilde{g}_k, s_{k-1}\}$.
    - Interpolation set $\mathcal{I}_k \Longrightarrow$ Quadratic model $Q_k \Longrightarrow$ Approximate gradient $\tilde{g}_k = \nabla Q_k(x_k)$.
  - Subproblem solver: NEWUOA.
    - The parameter RHOEND controls the precision of d_k.

3. NEWUOAs

- **The convergence of** NEWUOAs

3. NEWUOAs

- **The convergence of** NEWUOAs
  - Global convergence:

- **The convergence of** NEWUOAs
  - Global convergence:
    - $\mathrm{diam}(\mathcal{I}_k) \longrightarrow 0$, RHOEND $\longrightarrow 0$.

- **The convergence of** NEWUOAs
  - Global convergence:
    - $\text{diam}(\mathcal{I}_k) \longrightarrow 0$, RHOEND $\longrightarrow 0$.
  - R-linear convergence rate:

3. NEWUOAs

- **The convergence of** `NEWUOAs`
  - Global convergence:
    - $\mathrm{diam}(\mathcal{I}_k) \longrightarrow 0$, `RHOEND` $\longrightarrow 0$.
  - R-linear convergence rate:
    - $\mathrm{diam}(\mathcal{I}_k) \xrightarrow{\text{R-linear}} 0$, `RHOEND` $\xrightarrow{\text{R-linear}} 0$.

- **Preconditioning techniques of** NEWUOAs

- **Preconditioning techniques of** NEWUOAs
    - Second order information?

- **Preconditioning techniques of** NEWUOAs
  - Second order information?
  - Quadratic model $Q_k \Longrightarrow$ Approximate Hessian $\nabla^2 Q_k \Longrightarrow$ Precondition.

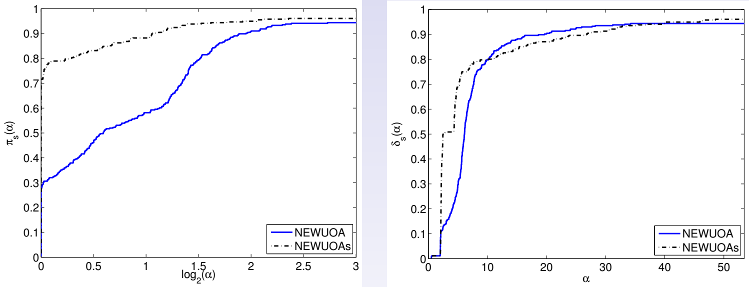
- **Preconditioning techniques of NEWUOAs**
  - Second order information?
  - Quadratic model $Q_k \Longrightarrow$ Approximate Hessian $\nabla^2 Q_k \Longrightarrow$ Precondition.
  - $\mathcal{S}_k = \{A_k\tilde{g}_k,\ \tilde{g}_k,\ s_{k-1}\}$.
    - $\nabla^2 Q_k$ positive definite, $A_k = [\nabla^2 Q_k]^{-1}$;

- **Preconditioning techniques of NEWUOAs**
  - Second order information?
  - Quadratic model $Q_k \Longrightarrow$ Approximate Hessian $\nabla^2 Q_k \Longrightarrow$ Precondition.
  - $\mathcal{S}_k = \{A_k \tilde{g}_k,\ \tilde{g}_k,\ s_{k-1}\}$.
    - $\nabla^2 Q_k$ positive definite, $A_k = [\nabla^2 Q_k]^{-1}$;
    - $\nabla^2 Q_k$ not positive definite, $A_k = ?$

4. Numerical results

4. Numerical results

- **Relatively large problems** $(n=50, 100, 150, 200)$



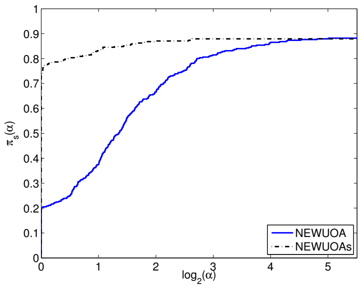
(a) Performance profile

(b) Data profile

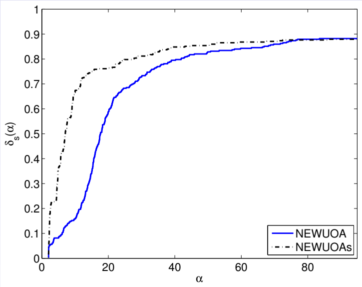
Fig 1 : Numerical comparison between `NEWUOA` and `NEWUOAs` $(\tau = 10^{-2})$

4. Numerical results

- **Relatively large problems** ($n = 50, 100, 150, 200$, cont.)



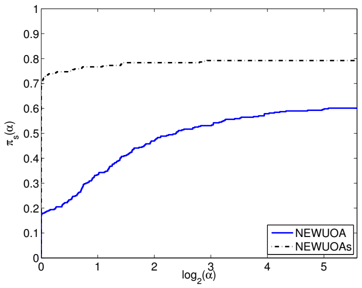
(a) Performance profile



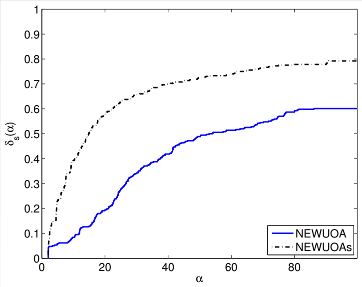
(b) Data profile

Fig 2 : Numerical comparison between `NEWUOA` and `NEWUOAs` ($\tau = 10^{-4}$)

- **Relatively large problems** ($n = 50, 100, 150, 200$, cont.)



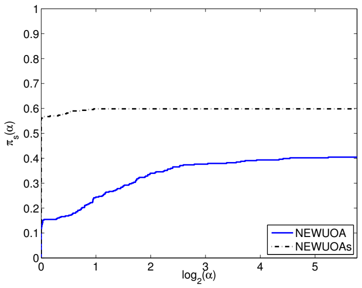
(a) Performance profile



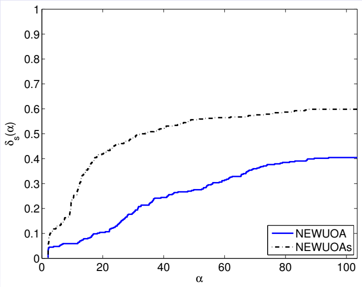
(b) Data profile

Fig 3 : Numerical comparison between `NEWUOA` and `NEWUOAs` ($\tau = 10^{-6}$)

- **Relatively large problems** ($n = 50, 100, 150, 200$, cont.)



(a) Performance profile



(b) Data profile

Fig 4 : Numerical comparison between `NEWUOA` and `NEWUOAs` ($\tau = 10^{-8}$)

- **Large problems**

Tab 1 : The performance of `NEWUOAs` on some 2000-dimensional problems

	f_{start}	f_{best}	$\#f$	CPU (s)
ARWHEAD	5.997000E+03	0.000000E+00	16095	6.42
BRYBND	7.200000E+04	6.486038E−09	50000	26.09
DIXMAANE	1.471453E+04	1.000000E+00	36264	21.12
DIXMAANF	2.734976E+04	1.000000E+00	36384	31.07
DIXMAANG	5.069653E+04	1.000000E+00	36393	22.72
DQRTIC	6.376035E+15	1.214880E−38	40854	14.70
GENHUMPS	5.122260E+07	1.624799E−26	36467	23.54
LIARWHD	1.170000E+06	2.428807E−24	16208	6.73
POWER	2.668667E+09	1.423292E−11	20130	19.19
SPARSQUR	5.627812E+05	6.381755E−30	16209	9.87

5. Concluding remarks

5. Concluding remarks

5. Concluding remarks

- A framework of subspace algorithms

5. Concluding remarks

- A framework of subspace algorithms
- A practical subspace algorithm `NEWUOAs`

5. Concluding remarks

- A framework of subspace algorithms
- A practical subspace algorithm `NEWUOAs`
- Preconditioning techniques

5. Concluding remarks

- A framework of subspace algorithms
- A practical subspace algorithm `NEWUOAs`
- Preconditioning techniques
- New strategy for defining the subspace

5. Concluding remarks

- A framework of subspace algorithms
- A practical subspace algorithm `NEWUOAs`
- Preconditioning techniques

- New strategy for defining the subspace
- Extend to constrained problems

5. Concluding remarks

- A framework of subspace algorithms
- A practical subspace algorithm NEWUOAs
- Preconditioning techniques

- New strategy for defining the subspace
- Extend to constrained problems
- General preconditioning techniques

Obrigado!

Obrigado! 谢谢！

zhang@mat.uc.pt
www.zhangzk.net

Obrigado!

Obrigado! 谢谢！

zhang@mat.uc.pt
www.zhangzk.net

Happy birthday, Grandpa!