Self-correcting geometry technique for derivative-free unconstrained optimization.

ANKE TRÖLTZSCH [CERFACS, TOULOUSE]

IN COLLABORATION WITH:

SERGE GRATTON [CNES/CERFACS, OCTOBER INPT/IRIT]
PHILIPPE L. TOINT [FUNDP, NAMUR, BELGIUM]

Journee des Doctorants

ENAC, 8 September 2009

Outline

Outline

Why should we use derivative-free optimization?

Some reasons to apply Derivative-Free Optimization (DFO):

- Growing sophistication of computer hardware and mathematical algorithms and software (opens new possibilities for optimization)
- Derivatives are unavailable - function is non-differentiable
- Accurate approximation of derivatives by finite differences is prohibitive - function evaluations are costly and/or noisy
- Automatic differentiation impossible to apply - source code not available or owned by a company

Applications:

- Tuning of algorithmic parameters
- Medical image registration
- Engineering design optimization

Bibliography on developments in DFO

Numerical optimization using local models:

- Powell, ”A direct search optimization method that models the objective function by quadratic interpolation”, 1994
- Conn, Scheinberg, and Toint, ”On the convergence of derivative-free methods for unconstrained optimization”, 1996
- Powell, ”The NEWUOA software for unconstrained optimization without derivatives”, 2004
- Conn, Scheinberg, and Vicente, ”Introduction in Derivative Free Optimization”, 2008
- Fasano, Nocedal, and Morales, ”On the geometry phase in model-based algorithms for derivative-free optimization”, 2009
- Scheinberg and Toint, ”Self-correcting geometry in model-based algorithms for derivative-free unconstrained optimization”, 2009

Engineering applications use more often SVM, NN, Kriging, ...

Requirements on globally convergent DFO algorithms

A globally convergent DFO algorithm must:

- Guarantee some form of descent away from stationarity.
- Guarantee some control of the geometry of the sample sets where the objective function is evaluated.
  E.g. when interpolation is used, control of accuracy of the interpolant is geometry-based.
- Contain convergence of a step size parameter to zero which indicates global convergence to a stationary point.

By global convergence we mean convergence of the algorithm to a local minimum from arbitrary starting points.

Outline

Interpolation models and trust-region methods

We consider the unconstrained minimization problem

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

where the first derivatives of the objective function are assumed to exist and be Lipschitz continuous, although explicit evaluation of these derivatives is assumed to be impossible.

We consider a model-based trust-region algorithm for computing local solutions of the minimization problem.

The method iteratively uses a local interpolation model of the objective function $f(x)$ to define a descent step, and adaptively adjust the region in which this model is trusted.

Notations in polynomial interpolation (I)

Consider $\mathcal{P}_n^d$, the space of polynomials of degree $\leq d$ in $\mathbb{R}^n$.

A polynomial basis ϕ of $\mathcal{P}_n^d$ is a set of $p+1$ polynomials of degree $\leq d$ that span $\mathcal{P}_n^d$. [Example: $\phi = \{1, x_1, x_2, x_1^2/2, x_2^2/2, x_1 x_2\}$]

For any basis ϕ, any polynomial $m(x) \in \mathcal{P}_n^d$ can be written as

$$m(x) = \sum_{j=0}^{p} \alpha_j \phi_j(x),$$

where α_j are real coefficients.

A polynomial $m(x)$ interpolates the function $f(x)$ at a given point y if $m(y) = f(y)$.

Notations in polynomial interpolation (II)

Given a sample set $Y = \{y^0, y^1, ..., y^p\} \subset \mathbb{R}^n$ and a polynomial $m(x)$ of degree d in $\mathbb{R}^n$ that interpolates $f(x)$ at the points Y, the coefficients $\alpha_0, ..., \alpha_p$ can be determined by solving the linear system:

$$M(\phi, Y)\alpha = f(Y),$$

where

$$M(\phi, Y) = \begin{bmatrix} \phi_0(y^0) & \phi_1(y^0) & \cdots & \phi_p(y^0) \\ \vdots & \vdots & & \vdots \\ \phi_0(y^p) & \phi_1(y^p) & \cdots & \phi_p(y^p) \end{bmatrix} \quad f(Y) = \begin{bmatrix} f(y^0) \\ \vdots \\ f(y^p) \end{bmatrix}.$$

If the coefficient matrix $M(\phi, Y)$ of the system is nonsingular, the set of points Y is called poised for polynomial interpolation in $\mathbb{R}^n$, otherwise the set Y is called non-poised.

Poisedness

Poisedness is a property of the interpolation set which ensures the interpolation process to be well-defined.

The most commonly used measure of well-poisedness in the polynomial interpolation literature is based on Lagrange polynomials.

Let $\{L_y(x), y \in Y\}$ be the set of Lagrange polynomials associated with Y. If Y is poised, Lagrange polynomials exist and are unique.

A poised set Y is said to be Λ-poised in B if one has that

$$\max_{y \in Y} \max_{x \in B} |L_y(x)| \leq \Lambda.$$

The smaller Λ, the better the quality of the geometry of the set of points.

Error bounds on model value and model gradient value

Given a ball $B(x, \Delta)$, a poised interpolation set $Y \in B(x, \Delta)$ and its asscociated basis of Lagrange polynomials $L_j(y), j = 0, ..., p$, there exists constants $\kappa_{ef} > 0$ and $\kappa_{eg} > 0$ such that, for any interpolation polynomial $m(x)$ of degree one or higher and any given point $y \in B(x, \Delta)$,

$$||f(y) - m(y)|| \leq \kappa_{ef} \sum_{j=0}^{p} ||y_j - y||^2 |L_j(y)|$$

and

$$||\nabla_x f(y) - \nabla_x m(y)|| \leq \kappa_{eg} \Lambda \Delta,$$

where $\Lambda = max_{j=0,...,p} max_{x \in B(x,\Delta)} |L_j(x)|$
[Conn, Scheinberg, and Vicente, 2008].

A general DFO trust-region framework (I)

- Compute an initial poised interpolation set Y_0
- Build a quadratic model $m_k(x_k + s)$ of the objective function around an iterate x_k

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

based on well-poised sample sets.

- Calculate a step s_k by solving

$$\min_{s \in B(x_k; \Delta_k)} m_k(x_k + s).$$

in the trust region $B(x_k; \Delta_k)$ where the model is assumed to represent $f(x)$ sufficiently well.

A general DFO trust-region framework (II)

- Evaluate $f(x_{k+1})$ and compute the ratio

$$\rho_k = \frac{f(x_k) - f(x_k + s_k)}{m(x_k) - m(x_k + s_k)} = \frac{\text{achieved reduction}}{\text{predicted reduction}}.$$

- Define $x_{k+1} = x_k + s_k$ and increase Δ_k (at successful iteration)
- Define $x_{k+1} = x_k$ and decrease Δ_k (at unsuccessful iterations)
- Include the new point x_{k+1} in the interpolation set Y_{k+1} depending on its success
- Compute the new interpolation model m_{k+1} around x_{k+1} using interpolation set Y_{k+1} if $Y_{k+1} \neq Y_k$
- Start a new iteration

Outline

Geometry improving steps

- Fasano, Nocedal, and Morales [2009] observed that an algorithm which simply ignores the geometry considerations may in fact perform quite well in practice, but may lose the property of provable global convergence to first-order critical points [Scheinberg and Toint, 2009].
- Failure of current iteration might be due to a too large trust region or a bad quality of the interpolation model (not well-poised)
- Shows that we cannot afford to do without a geometry phase (need to maintain quality of the geometry of the interpolation set)
- Improvement is carried out at special ”geometry improving” steps by computing additional function values at well-chosen points

A DFO trust-region method with geometry restoration

- Compute an initial poised interpolation set Y_0
- Build a quadratic model $m_k(x_k + s)$ of the objective function around an iterate x_k

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

based on well-poised sample sets.

- Calculate a step s_k by solving

$$\min_{s \in B(x_k; \Delta_k)} m_k(x_k + s).$$

in the trust region $B(x_k; \Delta_k)$ where the model is assumed to represent $f(x)$ sufficiently well.

A DFO trust-region method with geometry restoration

- Evaluate $f(x_{k+1})$ and compute the ratio

$$\rho_k = \frac{f(x_k) - f(x_k + s_k)}{m(x_k) - m(x_k + s_k)}.$$

- Define $x_{k+1} = x_k + s_k$ and increase Δ_k (at successful iteration)
- Define $x_{k+1} = x_k$ and decrease Δ_k (at unsuccessful iterations)
- Include the new point x_{k+1} in the interpolation set Y_{k+1} depending on its success
- Improve the position of the interpolation set by a geometry improving step
- Compute the new interpolation model m_{k+1} around x_{k+1} using interpolation set Y_{k+1} if $Y_{k+1} \neq Y_k$
- Start a new iteration

Geometry improving steps

- As those geometry restoration steps are expensive, one may ask if they are necessary.
- Idea is now to reduce the frequency and cost of the necessary tests as much as possible, while maintaining a mechanism for taking geometry into account.
- Design and convergence properties of new algorithm depend on a self-correction mechanism combining trust-region mechanism with polynomial interpolation setting [Scheinberg and Toint, 2009].

The new DFO trust-region algorithm (I)

- Compute an initial poised interpolation set Y_0
- Convergence test and geometry restoration if necessary
- Build a quadratic model $m_k(x_k + s)$ of the objective function around an iterate x_k

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

based on the current interpolation set.

- Calculate a new trial point x_k^+ by solving

$$\min_{s \in B(x_k; \Delta_k)} m_k(x_k + s).$$

in the trust region $B(x_k; \Delta_k)$.

The new DFO trust-region algorithm (II)

- Evaluate $f(x_{k+1})$ and compute the ratio ρ_k
- Define the next iterate
  - **case 1)** Successful iteration: include point in the set Y_{k+1}, adjust Δ and define $x_{k+1} = x_k^+$
  - **case 2)** Replace a far interpolation point: if set

$$F_k = \{y_{k,j} \in Y_k \text{ such that } ||y_{k,j} - x_k|| > \beta\Delta \text{ and } l_{k,j}(x_k^+) \neq 0\}$$

  is non-empty, include point in the set Y_{k+1}, set $\Delta_{k+1} = \Delta_k$
  - **case 3)** Replace a close interpolation point: if set $F_k = \emptyset$ and

$$C_k = \{y_{k,j} \in Y_k \text{ such that } ||y_{k,j} - x_k|| \leq \beta\Delta \text{ and } l_{k,j}(x_k^+) > \Lambda\}$$

  is non-empty, include point in the set Y_{k+1}, set $\Delta_{k+1} = \Delta_k$
  - **case 4)** Reduce trust-region radius

Self-correcting geometry

- Self-correcting property: If iteration k is unsuccessful, $F_k = \emptyset$ and $\Delta \leq \kappa_\Lambda ||g_M||$, then $C_k \neq \emptyset$.
- Provided the trust-region radius is small enough compared to the model gradient and all the significant interpolation points are contained in the trust region, then every unsuccessful iteration must result in an improvement of the interpolation set geometry [Scheinberg and Toint, 2009].

- To avoid large numerical errors when computing and minimizing the local models while not repairing the interpolation set, we considered a shifting and scaling technique.
  - shifts the interpolation set that center of Y is at the origin
  - scales the distances from the center to all other points to be between 0 and 1

Outline

Methodology

CUTEr testing environment

- 54 unconstrained test cases from CUTEr testing environment
- variables vary from 2 to 15 dimensions

Competitor: NEWUOA

- State of the art software developed by M.J.D. Powell [2004]
- Currently one of the best codes for minimization without derivatives.

Stopping criterion

- Stopping criteria are different
- Using optimal objective value computed by KNITRO (using second derivarives) as a reference
- We terminate when 6 correct significant figures in f were attained

Numerical results

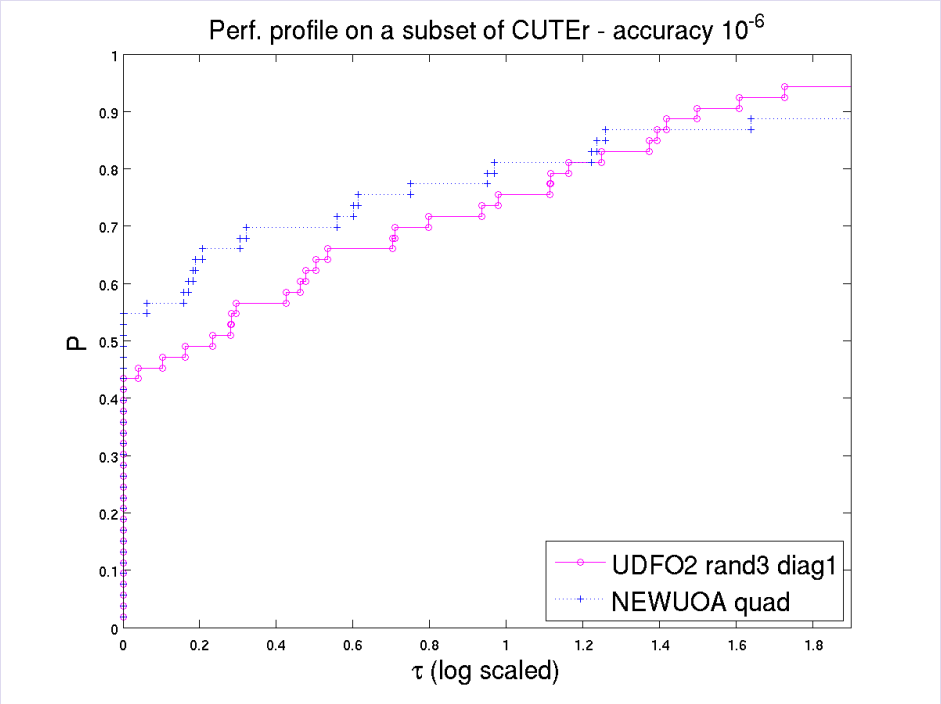


Figure: Performance profile on nbr. of function evaluations

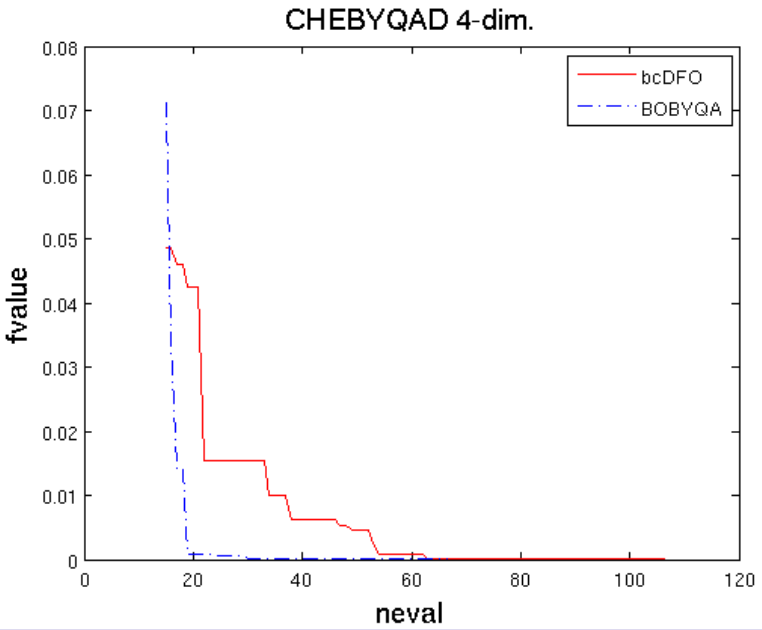
Outline

Extension to bound constraints

Changes in the algorithm:

- Minimizing the quadratic model inside the trust region and the bounds: modified More-Sorensen algorithm where active variables are fixed at its bounds, if one variables becomes active, it is also fixed and More-Sorensen algorithm is started again with the remaining free variables from the current point by updating g and Δ
- Initial interpolation set is computed as before but care must be taken when switching the current iterate to a point with lower function value - current iterate must stay inside bounds!
- Use of projected model gradient for termination
- This is ongoing work - no extensive numerical testing yet

Numerical results



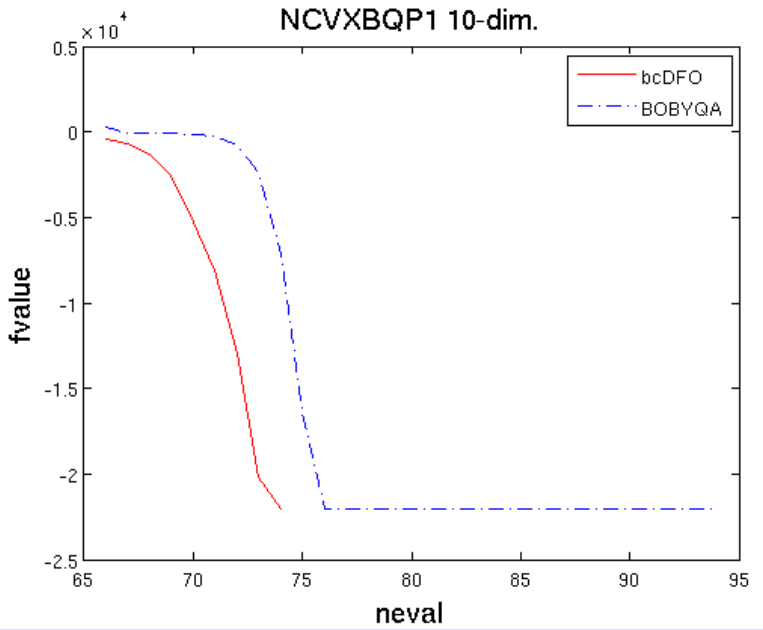


Figure: Comparison by nbr. of function evaluations

Outline

Conclusions and Perspectives

Summary

- Presented a new model-based DFO algorithm
- Implemented a robust version of the unconstrained algorithm UDFO [in collaboration with Ph.L. Toint]
- Compared UDFO to NEWUOA with partly satisfying results
- Algorithm extended to treat bound constraints

Perspectives

- Consider a more sophisticated update of the model Hessian when model degree is less than quadratic
- Numerical testing of bound-constrained DFO algorithm on a large test set of CUTEr
- Comparison of bcDFO to BOBYQA [Powell, 2006]