Model-based methods

MTH8418

S. Le Digabel, Polytechnique Montréal

Winter 2020

(v3)

Plan

Quadratic model of f

- Natural basis of the space of polynomials of degree ≤ 2 in $\mathbb{R}^n$
- It has $q+1 = (n+1)(n+2)/2$ elements
- $\phi(x) = \left(\phi_0(x), \phi_1(x), \ldots, \phi_q(x)\right)^\top = \left[1 \ x_1 \ x_2 \ \ldots \ x_n \ \frac{x_1^2}{2} \ \frac{x_2^2}{2} \ \ldots \ \frac{x_n^2}{2} \ x_1x_2 \ x_1x_3 \ \ldots \ x_{n-1}x_n\right]$
- Model of f: m_f defined by $\alpha \in \mathbb{R}^{q+1}$
- $m_f(x) = \alpha^\top \phi(x)$

Interpolation set

- Points at which f is known and which are used to construct the model m_f
- $p+1$ elements of $\mathbb{R}^n$: $Y = \{y^0, y^1, \ldots, y^p\}$. These points are also called data points
- $f(Y) = \big(f(y^0), f(y^1), \ldots, f(y^p)\big)^\top \in \mathbb{R}^{p+1}$
- The geometry of Y is important and will be studied later
- How to select the data points from the cache points? One solution: Take the points around the current iterate

Construction of the model

- Find $\alpha \in \mathbb{R}^{q+1}$ such that $\sum_{y \in Y} \left(f(y) - m_f(y)\right)^2$ is minimal

- Idea: Solve $M(\phi, Y)\alpha = f(Y)$ with

$$M(\phi, Y) = \begin{bmatrix} \phi_0(y^0) & \phi_1(y^0) & \dots & \phi_q(y^0) \\ \phi_0(y^1) & \phi_1(y^1) & \dots & \phi_q(y^1) \\ \vdots & \vdots & \vdots & \vdots \\ \phi_0(y^p) & \phi_1(y^p) & \dots & \phi_q(y^p) \end{bmatrix} \in \mathbb{R}^{(p+1)\times(q+1)}$$

- Cost in $\mathcal{O}(p^3)$

- 3 cases:
  - $p = q$: Determined
  - $p > q$: Overdetermined
  - $p < q$: Underdetermined

Number of necessary interpolation points

n	$q+1=\frac{(n+1)(n+2)}{2}$
2	6
3	10
4	15
5	21
10	66
20	231
50	1326

Typically in the DFO context, $n \leq 20$, but:

- Very limited number of evaluations
- Selection of the data points near the current iterate

$\Rightarrow$ The underdetermined case $p < q$ is the most common

Overdetermined & determined cases: 1/2

- More data points than necessary: $p > q$
- Use regression to solve the system in the least square sense, i.e. solve:
$$\min_{\alpha \in \mathbb{R}^{q+1}} \|M(\phi, Y)\alpha - f(Y)\|^2$$
- If $M(\phi, Y)$ has full column rank, analytic and unique solution given by
$$\alpha = M(\phi, Y)^{+} f(Y)$$
with $M(\phi, Y)^{+} = \left[M(\phi, Y)^{\top} M(\phi, Y)\right]^{-1} M(\phi, Y)^{\top}$ the pseudoinverse of $M(\phi, Y)$
- Works for the determined case $p = q$ (exact interpolation)

Overdetermined & determined cases: 2/2

- $M(\phi, Y)$ can be decomposed using the Singular Value Decomposition (SVD) $M(\phi, Y) = U\Sigma V^\top$ with:
  - $U \in \mathbb{R}^{(p+1)\times(p+1)}$, $U^\top U = I_{p+1}$
  - $\Sigma \in \mathbb{R}^{(p+1)\times(q+1)}$, diagonal: Singular values ≥ 0 (sv)
  - $V \in \mathbb{R}^{(q+1)\times(q+1)}$, $VV^\top = I_{q+1}$
- $M(\phi, Y)$ has not full rank if the smallest sv is 0
- Condition number of $M(\phi, Y)$: Largest sv / smallest sv
- $M(\phi, Y)^+ = V\Sigma^+U^\top$ where Σ^+ is the pseudoinverse of Σ obtained by replacing every non-zero sv by its reciprocal and transposing the resulting matrix
- Cost of the SVD: $\mathcal{O}((p+1)(q+1)^2)$

Underdetermined case: Infinite number of solutions

- Minimum Frobenius Norm (MFN) interpolation: Choose a solution that minimizes the Frobenius norm of the Hessian of the model (the curvature)
- $\alpha = \begin{bmatrix} \alpha_L \\ \alpha_Q \end{bmatrix}$ with $\alpha_L \in \mathbb{R}^{n+1}$, $\alpha_Q \in \mathbb{R}^{n_Q}$, $n_Q = n(n+1)/2$
- $F(\phi, Y) = \begin{bmatrix} M(\phi_Q, Y)M(\phi_Q, Y)^\top & M(\phi_L, Y) \\ M(\phi_L, Y)^\top & 0 \end{bmatrix} \in \mathbb{R}^{(p+n+2)\times(p+n+2)}$
- $F(\phi, Y) \begin{bmatrix} \mu \\ \alpha_L \end{bmatrix} = \begin{bmatrix} f(Y) \\ 0 \end{bmatrix} \to \alpha_L \in \mathbb{R}^{n+1}$ and $\mu \in \mathbb{R}^{p+1}$
- Use decomposition to solve the system, and then

$$\alpha_Q = M(\phi_Q, Y)^\top \mu \in \mathbb{R}^{n_Q}$$

Lagrange polynomials

- Basis of Lagrange polynomials: $p+1$ polynomials ℓ_j for $j = 0, 1, \ldots, p$, with

$$\ell_j(y^i) = \begin{cases} 1 \text{ if } i = j \\ 0 \text{ if } i \neq j \end{cases}$$

- Model: $m_f(x) = \sum_{i=0}^{p} f(y^i)\ell_i(x)$
- Cost of constructing a model is in $\mathcal{O}(p^3)$, but cost of updating the model by one point is in $\mathcal{O}(p^2)$

Lagrange polynomials: Example

- $f(x,y) = x + y + 2x^2 + 3y^3$
- $Y = \left\{ \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 2 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 2 \end{bmatrix} \right\}$
- $\begin{cases} \ell_0(x,y) = 1 - \frac{3}{2}x - \frac{3}{2}y + \frac{1}{2}x^2 + \frac{1}{2}y^2 + xy \\ \ell_1(x,y) = 2x - x^2 - xy \\ \ell_2(x,y) = 2y - y^2 - xy \\ \ell_3(x,y) = -\frac{1}{2}x + \frac{1}{2}x^2 \\ \ell_4(x,y) = xy \\ \ell_5(x,y) = -\frac{1}{2}y + \frac{1}{2}y^2 \end{cases}$
- $m_f(x,y) = 0 \times \ell_0(x,y) + 3\ell_1(x,y) + 4\ell_2(x,y) + 10\ell_3(x,y) + 7\ell_4(x,y) + 26\ell_5(x,y) = 2x^2 + 9y^2 + x - 5y$

Fully linear models

- $f \in \mathcal{C}^1$ and ∇f Lipschitz continuous

- A model m_f is called Fully Linear (FL) on $\mathcal{B}(y;\Delta)$ if

$$\begin{cases} |f(x) - m_f(x)| \leq \kappa_f \Delta^2 \\ \|\nabla f(x) - \nabla m_f(x)\| \leq \kappa_g \Delta \end{cases}$$

  for all $x \in \mathcal{B}(y;\Delta)$ and some constants κ_f and κ_g

- Δ is the accuracy parameter. It controls the model error (in terms of f and ∇f). It should converge to zero within an optimization algorithm

Fully quadratic models

- $f \in \mathcal{C}^2$ and ∇f^2 Lipschitz continuous
- A model m_f is called Fully Quadratic (FQ) on $\mathcal{B}(y;\Delta)$ if

$$\begin{cases} |f(x) - m_f(x)| \leq \kappa_f \Delta^3 \\ \|\nabla f(x) - \nabla m_f(x)\| \leq \kappa_g \Delta^2 \\ \|\nabla^2 f(x) - \nabla^2 m_f(x)\| \leq \kappa_h \Delta \end{cases}$$

for all $x \in \mathcal{B}(y;\Delta)$ and some constants κ_f, κ_g, κ_h

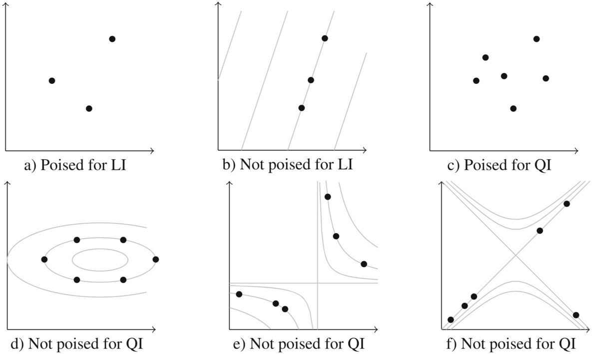
FL and FQ classes of models

A set of models $\mathcal{M} = \{m : \mathbb{R}^n \to \mathbb{R}, m \in \mathcal{C}^2\}$ is called a FL (FQ) class of models if

- there exists a FL (FQ) model in $\mathcal{M}$
- there exists a model-improvement algorithm ($\texttt{MIA}$) that, in a finite number of steps, can
  - determine if a given model is FL (FQ) on $\mathcal{B}(x; \Delta)$
  - **or** find a model that is FL (FQ) on $\mathcal{B}(x; \Delta)$

Poisedness

- A set Y is said poised for polynomial interpolation or regression if $M(\phi, Y)$ is nonsingular ($p = q$), or if $M(\phi, Y)$ has full rank
- Example from [Audet and Hare, 2017]:



a) Poised for LI

b) Not poised for LI

c) Poised for QI

d) Not poised for QI

e) Not poised for QI

f) Not poised for QI

Well-poisedness

- Good geometry of Y = well-poised set Y
- Condition number of $M(\phi, Y)$ may be a good indicator only with some bases ϕ and if some specific scaling is performed
- Lagrange polynomials can be good indicators
- Quantify the well-poisedness with the Λ-poisedness

Λ-poisedness

Let $\Lambda > 0$ and $\mathcal{B} \subseteq \mathbb{R}^n$. Y is Λ-poised in $\mathcal{B}$ if

- $\Lambda \geq \Lambda_\ell = \max\limits_{0 \leq i \leq p} \max\limits_{x \in \mathcal{B}(Y)} |\ell_i(x)|$ where $\mathcal{B}(Y)$ is the smallest ball containing Y
- **or** for all $x \in \mathcal{B}$, there exists $\lambda \in \mathbb{R}^{p+1}$ such that $\phi(x) = \sum\limits_{i=0}^{p} \lambda_i \phi(y^i)$ and $\max\limits_{0 \leq i \leq p} |\lambda_i| \leq \Lambda$
- **or** replacing any point in Y by any $x \in \mathcal{B}$ can increase the volume of the set $\phi(Y)$ at most by a factor Λ, with $\phi(Y) = \{\phi(y^0), \phi(y^1), \dots, \phi(y^p)\}$ and its volume defined by $\frac{|\det(M(\phi, Y))|}{(p+1)!}$

Conclusion on the quality of models

- From [Audet and Hare, 2017]: *"if models are of sufficient quality, then it is possible to adapt algorithms from smooth optimization to derivative-free optimization."*

Introduction

- We consider the unconstrained problem $\min_{x \in \mathbb{R}^n} f(x)$ with $f \in \mathcal{C}^1$. Bounds and linear constraints can be "easily" treated
- Need more elaborate strategies to handle general constraints. See Lesson #9 on the constraints
- Generalization of the gradient method
- Idea: Use the model to find a descent direction on which a line-search is then performed
- Quality of model: "Only" a good representation of ∇f is required:

$$\|\nabla f(x) - \nabla m_f(x)\| \leq \kappa_g \Delta$$

for all $x \in \mathcal{B}(y; \Delta)$ and some constant κ_g

Notations for the MBD framework

- x_0: Starting point; x_k: Current iterate
- m_f model of f such that $\|\nabla f(x) - \nabla m_f(x)\| \leq \kappa_g \Delta$ for all $x \in \mathcal{B}(y; \Delta)$ and some constant κ_g
- $\Delta_k > 0$: Model accuracy parameter
- $\mu_0 > 0$: Target accuracy parameter
- $\eta \in]0; 1[$: Armijo parameter
- $\varepsilon_d \in]0; 1[$: Minimum decrease angle parameter
- $\epsilon \geq 0$: Stopping tolerance

Model-Based Descent (MBD): Algorithm (1/2)

Step 0 **[Initialization]** Choose
x_0, Δ_0, μ_0, η, ε_d, ϵ. Initialize model m_f^0, $k \leftarrow 0$

Step 1 **[Model]** Use Δ_k and a finite number of points to create m_f^k

Step 2 **[Model accuracy checks]**:

- if ($\Delta_k < \epsilon$ and $\|\nabla m_f^k(x_k)\| < \epsilon$): Stop (success)
- if ($\Delta_k > \mu_k \|\nabla m_f^k(x_k)\|$)
  - declare the model inaccurate
  - $\Delta_{k+1} \leftarrow \Delta_k/2$, $\mu_{k+1} \leftarrow \mu_k$, $x_{k+1} \leftarrow x_k$
  - $k \leftarrow k+1$, goto **[Step 1]**
- if ($\Delta_k \leq \mu_k \|\nabla m_f^k(x_k)\|$):
  - declare the model accurate
  - goto **[Step 3]**

Model-Based Descent (MBD): Algorithm (2/2)

Step 3 [Line-search]:

- choose d_k such that $\left(\frac{d_k}{\|d_k\|}\right)^\top \left(\frac{\nabla m_f^k(x_k)}{\|\nabla m_f^k(x_k)}\|\right) < -\varepsilon_d$
- search t_k such that $f(x_k + t_k d_k) < f(x_k) + \eta t_k d_k^\top \nabla m_f^k(x_k)$

Step 4 [Updates]:

- if t_k is found (line-search success):
  - $x_{k+1} \leftarrow y$ with $f(y) \leq f(x_k + t_k d_k)$
  - $\mu_{k+1} \leftarrow \mu_k$
- otherwise (line-search failure):
  - $x_{k+1} \leftarrow x_k$
  - $\mu_{k+1} \leftarrow \mu_k/2$
- $\Delta_{k+1} \leftarrow \Delta_k$
- $k \leftarrow k+1$, goto **[Step 1]**

Convergence [Audet and Hare, 2017] (1/3)

- **Descent direction [Step 3]**:
  - Selection of d_k is left vague
  - $d_k = -\nabla m_f^k(x_k)$ will always satisfy the descent condition
  - Other descent directions can be considered. For example, a quasi-Newton descent direction
  - If μ_k is sufficiently small, then Step 3 implies d_k is a descent direction for f at x_k
- Choice of x_{k+1} in **[Step 4]** ($\simeq$ *Search Step*):
  - $x_{k+1} = x_k + t_k d_k$ is valid
  - Alternative: Use of another (heuristic) method

Convergence [Audet and Hare, 2017] (2/3)

- If the model is not sufficiently accurate, then MBD requests greater accuracy at x_k by decreasing Δ_k
- As $\Delta_k \to 0$, $\nabla m_f^k(x_k) \to \nabla f(x_k)$, and:
  - $\Delta_k \to 0$ and $\|\nabla m_f^k(x_k)\| \to 0$: The stopping test in **[Step 2]** should eventually declare success

    or
  - $\Delta_k \to 0$ and $\|\nabla m_f^k(x_k)\| \to \gamma > 0$: The model should eventually be declared accurate
- As line-search failures occur, $\mu_k \to 0$, demanding higher and higher model accuracy as the algorithm progresses
- MBD can only declare the model inaccurate a finite number of times in a row, i.e., MBD cannot get stuck in a loop between **[Step 1]** and **[Step 2]**

Convergence [Audet and Hare, 2017] (3/3)

- Stopping condition:
  - Based on the approximated gradient becoming sufficiently small
  - One wants the approximate gradient to be small because the true gradient is small, not because of inaccuracy in the approximations
  - The stopping condition in **[Step 2]** includes a condition that the radius Δ_k must also be small
  - Assuming κ_g is reasonably small, by making ϵ sufficiently small, approximate first order optimality is achieved

- If MBD is run with $\epsilon = 0$, then

$$\lim_{k \to \infty} \|\nabla f(x_k)\| = 0$$

Introduction

- Unconstrained problem $\min_{x \in \mathbb{R}^n} f(x)$
- We present a first order algorithm that ensures global convergence to first order critical points using a FL class of models
- This is the general DFTR framework from [Conn et al., 2009]
- We suppose $f \in \mathcal{C}^1$ and ∇f Lipchitz continuous. But derivatives are not available

Notations for the DFTR framework

- x_k: Current iterate
- Model of f:

$$m_f(x) = f(x_k) + g_k^\top (x - x_k) + \frac{1}{2}(x - x_k)^\top H_k(x - xk)$$

with g_k, H_k: Gradient and Hessian of the model at iteration k

- Δ_k: Trust-region radius
- For candidate t: $r_k(t) = \frac{f(x_k)-f(t)}{m_f(x_k)-m_f(t)}$
- $m_f \leftarrow m_f \cup \{t\}$ means: "update the model with t"
- $\varepsilon_c > 0$
- $0 \leq \eta_0 \leq \eta_1 < 1$, $\eta_1 \neq 0$
- $0 < \gamma_{dec} < 1 < \gamma_{inc}$
- $\mu > 0$

First order algorithm: 1/3

Step 0 **[Initialization]** Choose

- FL class of models
- Model-improvement algorithm (`MIA`)
- x_0, Δ_{max}, $\Delta_0 \in]0; \Delta_{max}]$, initialize model m_f, $k \leftarrow 0$

Step 1 **[Criticality test]**: If $\|g_k\| \leq \varepsilon_c$

- Call `MIA` to certify m_f FL on $\mathcal{B}(x_k; \Delta_k)$
- If ((m_f not FL on $\mathcal{B}(x_k; \Delta_k)$) or ($\Delta_k > \mu \|g_k\|$))
  [*model not good enough or trust-region too large*]:
  Construct new model
- Check stopping criterion; Stop or goto **[Step 1]**

First order algorithm: 2/3

Step 2 **[Subproblem Optimization]**

- Find $t \in \underset{x \in \mathcal{B}(x_k;\Delta_k)}{\operatorname{argmin}} \; m_f(x)$
- Evaluate candidate t in $\mathcal{B}(x_k;\Delta_k)$
- Compute $r_k(t) = \frac{f(x_k)-f(t)}{m_f(x_k)-m_f(t)}$

Step 3 **[Acceptance of candidate]**

- If $r_k(t) > \eta_1$ or $(r_k(t) > \eta_0$ and m_f is FL on $\mathcal{B}(x_k;\Delta_k)$ $)$:
  $x_{k+1} \leftarrow t$, $m_f \leftarrow m_f \cup \{t\}$
- Otherwise:
  $x_{k+1} \leftarrow x_k$

Step 4 **[Model Improvement]** If $r_k(t) < \eta_1$ and m_f not FL

- Call `MIA` to certify m_f FL on $\mathcal{B}(x_k;\Delta_k)$

First order algorithm: 3/3

Step 5 **[Trust-region radius update]**

$$\Delta_{k+1} \in \begin{cases} [\Delta_k; \min\{\gamma_{inc}\Delta_k, \Delta_{max}\}] & \text{if } r_k(t) \geq \eta_1 \\ \{\gamma_{dec}\Delta_k\} & \text{if } r_k(t) < \eta_1 \text{ and } m_f \text{ is FL} \\ \{\Delta_k\} & \text{if } r_k(t) < \eta_1 \text{ and } m_f \text{ is not FL} \end{cases}$$

- $k \leftarrow k+1$, goto **[Step 1]**

First order algorithm: Comments

- Successful iteration if $r_k(t) \geq \eta_1$. Then $\Delta_{k+1} \geq \Delta_k$
- Acceptable iteration if $\eta_1 > r_k(t) \geq \eta_0$ and m_f is FL. Then $\Delta_{k+1} < \Delta_k$
- Model-improving iteration if $r_k(t) < \eta_1$ and m_f not FL. Then model must be improved and x_k, Δ_k are not updated
- Unsuccessful iteration if $r_k(t) < \eta_0$ and m_f is FL. Then $\Delta_{k+1} < \Delta_k$ and x_k is not updated
- Do not reduce the trust-region radius when the model is not good

Second order algorithm

- Global convergence to second order critical points using a FQ class of models
- $f \in \mathcal{C}^2$ and ∇f Lipschitz continuous
- Second order stationarity of the model:
  $\sigma_k^m = \max\{\|g_k\|, -\lambda_{min}(H_k)\}$ where $\lambda_{min}(H_k)$ denotes the smallest eigenvalue of H_k
- Criticality test based on σ_k^m instead of $\|g_k\|$

Definition of the subproblem

- Trust-region subproblem
- We want to solve $\min\limits_{x \in \mathcal{B}(x_k;\Delta_k)} m_f(x)$ in order to obtain a candidate t
- The trust-region constraint can be expressed with different norms
- We do not need an exact resolution

Optimization of the subproblem

Some methods to solve the subproblem:

- Gradient projection
- Moré–Sorensen
- Generalized Lanczos trust-region
- Sequential Subspace
- Gould–Robinson–Thorne
- Rendl–Wolkowicz

Quadratic models

Model Quality

Model-based descent

Derivative-Free Trust-Region Framework

References

DFTR solvers

- BOBYQA
- COBYLA
- CONDOR
- DFO
- LINCOA
- NEWUOA
- ORBIT
- SNOBFIT
- Wedge

References I

Audet, C. and Hare, W. (2017).
Derivative-Free and Blackbox Optimization.
Springer Series in Operations Research and Financial Engineering. Springer International Publishing, Berlin.

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

Golub, G. and Van Loan, C. (1996).
Matrix Computations, chapter 2.5.3 The Singular Value Decomposition, pages 70–71.
The John Hopkins University Press, Baltimore and London, third edition. (SVD).

References II

Gould, N., Lucidi, S., and Toint, P. (1999).
Solving the trust-region subproblem using the Lanczos method.
SIAM Journal on Optimization, 9(2):504–525.

Gould, N., Robinson, D., and Thorne, H. (2010).
On solving trust-region and other regularised subproblems in optimization.
Mathematical Programming Computation, 2(1):21–57.

Moré, J. and Sorensen, D. (1983).
Computing a trust region step.
SIAM Journal on Scientific Computing, 4(3):553–572.

References III

Nocedal, J. and Wright, S. (2006).
Numerical Optimization.
Springer Series in Operations Research and Financial Engineering. Springer, Berlin, second edition.
(Gradient projection).

Rendl, F. and Wolkowicz, H. (1997).
A semidefinite framework for trust region subproblems with applications to large scale minimization.
Mathematical Programming, 77(1):273–299.