

# Requirements for Automatic Differentiation in Numerical Optimization

Nick Gould (RAL)

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \quad f(x) \quad \text{subject to} \quad c(x) = 0$$

2nd European Workshop on Automatic Differentiation  
Cranfield University at Shrivenham  
18th November 2005



2nd European Workshop on Automatic Differentiation, Shrivenham, 18th Nov 2005 – p. 1/12

## Nonlinear optimization

Smooth optimization problem:

**unconstrained**

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \quad f(x)$$

**constrained**

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \quad f(x) \quad \text{such that} \quad c(x) = 0$$

1st-order optimality:

$$\nabla_x f(x) = 0$$

$$\nabla_x f(x) - \nabla_x c(x) y = 0$$

for Lagrange multipliers  $y$

2nd-order optimality:

$$\nabla_{xx} f(x) \succeq 0$$

$$\nabla_{xx} f(x) - \nabla_{xx} c(x) \cdot y \succeq 0$$

on null-space  $\nabla_{xx} c(x)$

$\implies$  derivatives **important** for verifying optimality



2nd European Workshop on Automatic Differentiation, Shrivenham, 18th Nov 2005 – p. 3/12

## Summary of the talk

- Nonlinear optimization and derivatives
- Partial separability
- Hessian sub-structure
- Lipschitz bounds
- Non-differentiable terms and code branches

## Algorithms — unconstrained optimization

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \quad f(x)$$

**Generic algorithm:** improve estimate  $x$  by computing correction  $s$  to reduce (minimize?) the Taylor model

$$s^T \nabla_x f(x) + \frac{1}{2} s^T H s$$

for some symmetric  $H \approx \nabla_{xx} f(x) \implies$  at least requires  $\nabla_x f(x)$

Choice of  $H$  —

- $\nabla_{xx} f(x)$
- finite-difference approximation  $\implies$  requires  $\nabla_x f(x)$
- secant approximation  $\implies$  requires  $\nabla_x f(x)$

Model minimization methods include

- solve  $Hs = -\nabla_x f(x)$  by factorization  $\implies$  requires  $H$
- iterative method  $\implies$  requires  $Hv$  products + preconditioning

Needs **globalisation**



2nd European Workshop on Automatic Differentiation, Shrivenham, 18th Nov 2005 – p. 4/12

## Algorithms — constrained optimization

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \quad f(x) \quad \text{such that} \quad c(x) = 0$$

**Generic algorithm:** improve estimate  $x$  by computing correction  $s$  to reduce (minimize?) the Taylor model

$$s^T \nabla_x f(x) + \frac{1}{2} s^T H s \quad \text{such that} \quad \nabla_x c(x)^T s + c(x) = 0$$

for some symmetric (indefinite?)  $H \approx \nabla_{xx} f(x) - \nabla_{xx} c(x) \cdot y$   
 $\implies$  at least require  $\nabla_x f(x)$  and  $\nabla_x c(x)$

Model minimization methods include

- solve  $\begin{pmatrix} H & \nabla_x c(x) \\ \nabla_x c(x)^T & 0 \end{pmatrix} \begin{pmatrix} s \\ z \end{pmatrix} = - \begin{pmatrix} \nabla_x f(x) \\ c(x) \end{pmatrix}$   
by factorization  $\implies$  requires  $H$  and  $\nabla_x c(x)$
- iterative method  $\implies$  requires  $Hv$ ,  $\nabla_x c(x)v$  and  $\nabla_x c(x)^T w$  products + preconditioning

Needs **globalisation**



## Generic problem structure — group partial separability

A function  $f$  is **group** partially separable if

(Conn, G. & Toint)

- $f(x) = \sum_{j=1}^g \gamma_j(g_j(x))$
- each **group function**  $\gamma_j : \mathbb{R} \rightarrow \mathbb{R}$
- each **group**  $g_j$  is partially separable

N.B. decomposition is not unique. Common examples —

- least-squares problems —  $g_j(\alpha) = \alpha^2$
- penalty and augmented Lagrangian functions  $f + \rho \|c(x) - v\|_2^2$

Derivatives from chain rule

$$\begin{aligned} \nabla_x f(x) &= \sum_{j=1}^g \gamma_j' \nabla_x g_j \quad \text{and} \\ \nabla_{xx} f(x) &= \sum_{j=1}^g \gamma_j'' \nabla_x g_j \nabla_x g_j^T + \sum_{j=1}^g \gamma_j' \nabla_{xx} g_j \end{aligned}$$

Group functions are often trivial  $\implies$  don't need AD for these!



## Generic problem structure — partial separability

A function  $f$  is **partially separable** if

(Griewank & Toint)

- $f(x) = \sum_{i=1}^e f_i(x) + a^T x + b$
- each **nonlinear element**  $f_i(x)$  has a large **invariant** subspace  $\implies$   
 $f_i(x) \equiv e_i(w_i)$  where  $x = U_i w_i$  and  $U_i$  is  $n$  by  $n_i \ll n$ 
  - e.g. each nonlinear element might only depend on a few variables

N.B. decomposition is not unique

Vital consequence — since differentiation is linear  $\implies$

$$\begin{aligned} \nabla_x f(x) &= \sum_{i=1}^e \nabla_x f_i(x) + a = \sum_{i=1}^e W_i \nabla_{w_i} e_i(w_i) + a \quad \text{and} \\ \nabla_{xx} f(x) &= \sum_{i=1}^e \nabla_{xx} f_i(x) = \sum_{i=1}^e W_i \nabla_{w_i w_i} e_i(w_i) W_i^T \end{aligned}$$

$\implies$  only need derivatives of low-dimensional functions  $e_i$

$\implies$  forward **and** backward AD is feasible



## Sparsity + differentiability $\implies$ partial separability

**Theorem** (Griewank & Toint, 82)

Suppose that  $f \in C^2$  &  $\nabla_{xx} f$  is sparse  $\implies$   $f$  is partially separable  
 [ not all partially separable  $f$  have sparse Hessians, e.g.  $(\sum_{i=1}^n x_i)^2$  ]

$\implies$  many (most?) real-life large-scale problems involve partially separable functions  $\implies$  crucial to exploit this

E.g. LANCELOT B from GALAHAD using group partial separability with exact and automatic 1st & 2nd derivatives — AD02 from HSL for AD

CUTEr problem	dimension	max $n_i$	exact	forward	backward
MINSURF	40401	2	21.83	33.09	28.34
DIXMAANA	90000	2	1.06	10.16	8.40
POWELLSG	5000	1	2.07	3.40	3.05
WOODS	100000	1	14.58	29.29	26.02

**Requirement:** determine (group) partial separable structure automatically



## Hessian sub-structure

Often need access to parts of  $\nabla_{xx}f(x)$  for scaling and/or preconditioning

1. diagonal entries — Jacobi preconditioner — common default
2. band around diagonal — default within LANCELOT
3. multigrid smoothing with Gauss-Seidel: require sequence of solves  
Lower triangle  $(\nabla_{xx}f)w^+ = r$  — strict upper triangle  $(\nabla_{xx}f)w$

- can we form strict upper triangle  $(\nabla_{xx}f)w$ ?
- can we perform substitution with lower triangle  $(\nabla_{xx}f)$ ?

4. given a rectangular matrix  $N$  with few columns, require (dense)  
**reduced Hessian**

$$N^T \nabla_{xx}f(x) N$$

without forming  $\nabla_{xx}f(x)$

**Requirement:** compute subsets of Hessian entries efficiently



## Non-differentiable terms and branches

Bundle (and other) methods for non-smooth optimization require **random** sub-gradients at points of non-differentiability

Can occur

- when evaluating subexpression depending on non-smooth intrinsics such as ABS
- at branches based on arithmetic expressions in the code

**Requirement:** automatically provide random sub-gradients at points of non-differentiability



## Lipschitz bounds

Many algorithms for global or non-smooth optimization are predicated on existence of Lipschitz constants

$$\begin{aligned} |f(x) - f(y)| &\leq \lambda_f \|x - y\| \\ \|\nabla_x f(x) - \nabla_x f(y)\| &\leq \lambda_g \|x - y\| \quad \text{or} \\ \|\nabla_{xx} f(x) - \nabla_{xx} f(y)\| &\leq \lambda_h \|x - y\| \end{aligned}$$

for e.g., all  $x, y$  in a given bounded region

**Requirement:** automatically estimate (tight?) bounds on Lipschitz constants



## Conclusions

- AD has made large (but often transparent) impact on nonlinear optimization
- integral part of modeling languages such as AMPL and GAMS, and available for optimization libraries like GALAHAD
- important outstanding requirements: automatically
  - identify (group) partial separability
  - allow computation of Hessian sub-structures
  - obtain Lipschitz bounds
  - provide (random) sub-gradients of non-differentiable functions

**Thank you all** for providing such wonderful tools!

