# Approximating Hessians in multilevel unconstrained optimization

Vincent Malmedy[1,2] Philippe Toint[1]

[1]Department of Mathematics, FUNDP – University of Namur
[2]Research Fellow, F.R.S.-FNRS
(vincent.malmedy@fundp.ac.be)

Sparse Days
CERFACS, Toulouse, France June 23–24, 2008

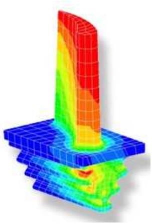
# Outline

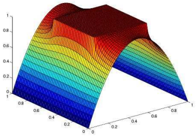
1. Motivation
2. Structure preserving Hessian approximation schemes
   - Finite-difference methods
   - Sparse PSB
   - Partially separable PSB
   - Partitioned BFGS
3. Numerical experience
4. Conclusion and perspectives

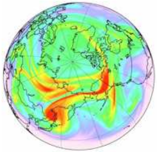
# Multilevel problems

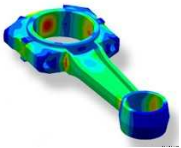
Lots of practical problems defined in an infinite-dimensional space:

- Parameter estimation in ODE or PDE
- Optimal control problems
- Variational problems (minimum surface problem)
- Surface design (shape optimization)
- Data assimilation in weather forecast









# Considered problem

Discretization used to approximate the real solution, but

- several levels of accuracy possible $\rightarrow$ multilevel problems
- fine mesh for good accuracy $\rightarrow$ large-scale problem at the finest level

Consider at finest level, the unconstrained optimization problem:

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

with

- $f : \mathbf{R}^n \rightarrow \mathbf{R}$ twice-continuously differentiable and bounded below
- no convexity assumption
- unavailable (or too expensive) Hessian

## Finite-difference methods


1. Motivation

2. Structure preserving Hessian approximation schemes
   - Finite-difference methods
   - Sparse PSB
   - Partially separable PSB
   - Partitioned BFGS

3. Numerical experience

4. Conclusion and perspectives

# Finite-difference methods

## Classical framework — drawback

- Each Hessian matrix column classically given by a small variation of $\nabla_x f$ in the corresponding canonical direction $e_j$:

$$H_{:,j}(x) := \frac{\nabla_x f(x + he_j) - \nabla_x f(x)}{h}$$

$\rightarrow$ #gradient evaluation = problem size

- However, sparsity typically encountered in such problems
$\rightarrow$ Try to not compute known zeros

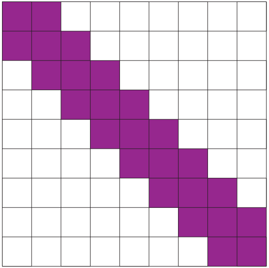
- Powell-Toint method (symmetric adaptation of Curtis-Powell-Reid for Jacobians, 1974)

# Finite-difference methods

## Powell & Toint, 1979

### Powell-Toint

1. Apply CPR algorithm to lower triangular sparsity pattern of $H$
2. Estimate corresponding gradient differences
3. Reconstruct entries of the estimated $H$ by solving a triangular system

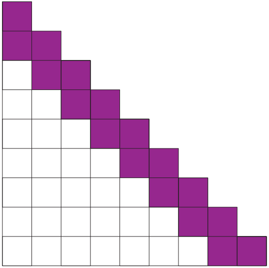


# Finite-difference methods

Powell & Toint, 1979

## Powell-Toint

1. Apply CPR algorithm to lower triangular sparsity pattern of $H$
2. Estimate corresponding gradient differences
3. Reconstruct entries of the estimated $H$ by solving a triangular system

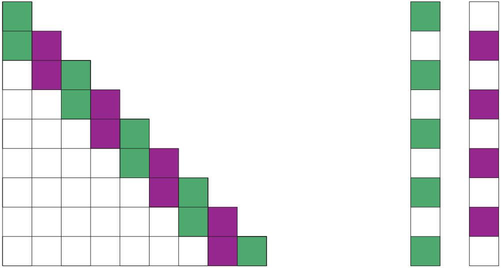


# Finite-difference methods

Powell & Toint, 1979

## Powell-Toint

1. Apply CPR algorithm to lower triangular sparsity pattern of $H$
2. Estimate corresponding gradient differences
3. Reconstruct entries of the estimated $H$ by solving a triangular system

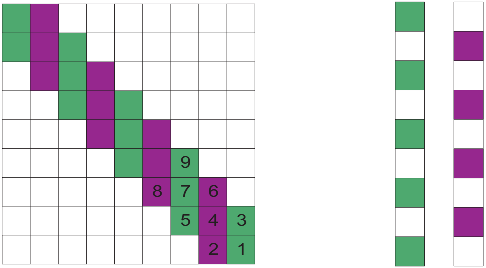


# Finite-difference methods

Powell & Toint, 1979

## Powell-Toint

1. Apply CPR algorithm to lower triangular sparsity pattern of $H$
2. Estimate corresponding gradient differences
3. Reconstruct entries of the estimated $H$ by solving a triangular system



# Sparse PSB


1. Motivation

2. Structure preserving Hessian approximation schemes
   - Finite-difference methods
   - Sparse PSB
   - Partially separable PSB
   - Partitioned BFGS

3. Numerical experience

4. Conclusion and perspectives

# Secant updates

## Classical secant updating schemes — drawback

### Secant equation

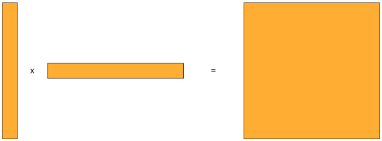
$$H^+ s = y$$

with $s = x^+ - x$ and $y = g^+ - g$

BFGS: $H^+ = H - \frac{(Hs)(Hs)^T}{\langle s, Hs \rangle} + \frac{yy^T}{\langle s, y \rangle}$

SR1: $H^+ = H + \frac{(y - Hs)(y - Hs)^T}{\langle s, y - Hs \rangle}$

- Do not take account of the usually existent structure of large-scale problems
  $\rightarrow$ inefficiency
- Fills the Hessian
  $\rightarrow$ memory storage problems



# Sparse Powell-symmetric-Broyden (PSB)

Impose: Hessian symmetry and sparsity, secant equation

## Sparse PSB

1. Define $S = \mathcal{P}(ss^T) + \mathrm{diag}(s \bullet s)$.
2. Solve $S\lambda = y - Hs$ for $\lambda$.
3. Compute $H^+ = H + \mathcal{P}(s\lambda^T + \lambda s^T)$.

$\mathcal{P}$: operator zeroing entries outside sparsity structure

Global and superlinear local convergence
when combined with trust-region technique

## Partially separable PSB

1. Motivation
2. Structure preserving Hessian approximation schemes
   - Finite-difference methods
   - Sparse PSB
   - Partially separable PSB
   - Partitioned BFGS
3. Numerical experience
4. Conclusion and perspectives

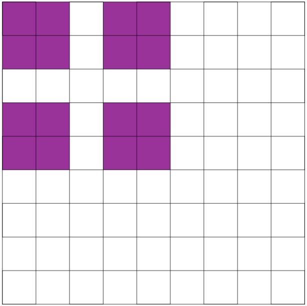
# Partial separability
## Griewank & Toint, 1982

- Structured problem $\rightarrow$ partial separability:

$$f = \sum f_i$$

  with each element function $f_i$ depending on a few components
- Known Hessian shape:

$$\nabla_{xx} f_1(x) =$$



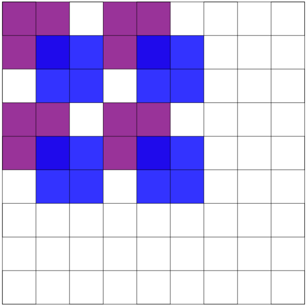
# Partial separability
## Griewank & Toint, 1982

- Structured problem $\rightarrow$ partial separability:

$$f = \sum f_i$$

  with each element function $f_i$ depending on a few components
- Known Hessian shape:

$$\nabla_{xx}(f_1 + f_2)(x) =$$



# Partial separability
## Griewank & Toint, 1982

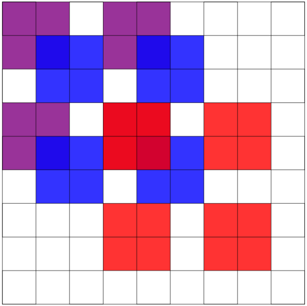
- Structured problem → partial separability:

$$f = \sum f_i$$

with each element function $f_i$ depending on a few components

- Known Hessian shape:

$$\nabla_{xx}(f_1 + f_2 + f_3)(x) =$$



# Partial separability
## Griewank & Toint, 1982

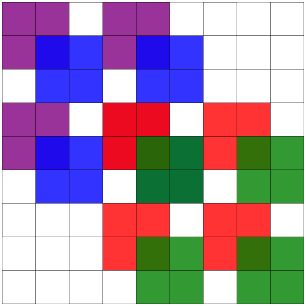
- Structured problem $\rightarrow$ partial separability:

$$f = \sum f_i$$

with each element function $f_i$ depending on a few components

- Known Hessian shape:

$$\nabla_{xx}(f_1 + f_2 + f_3 + f_4)(x) =$$

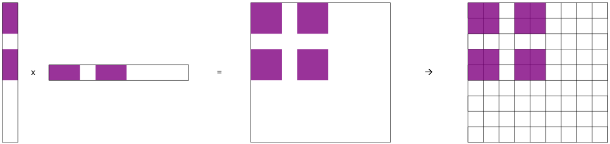


# Partitioned Hessian update

Griewank & Toint, 1982

## Main idea

Update each $H_i$ rather than $H$



Require the knowledge of the gradient differences decomposition

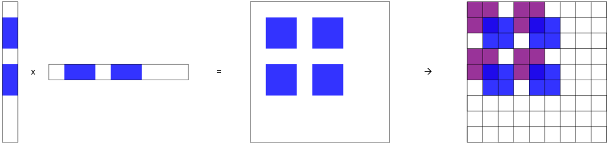
- but not always possible
- or often expensive

# Partitioned Hessian update

Griewank & Toint, 1982

## Main idea

Update each $H_i$ rather than $H$




Require the knowledge of the gradient differences decomposition

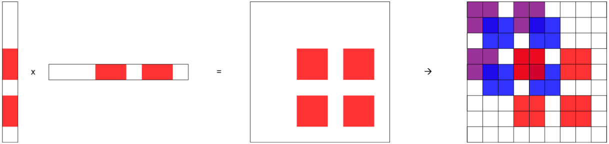
- but not always possible
- or often expensive

# Partitioned Hessian update

## Griewank & Toint, 1982

### Main idea

Update each $H_i$ rather than $H$




Require the knowledge of the gradient differences decomposition

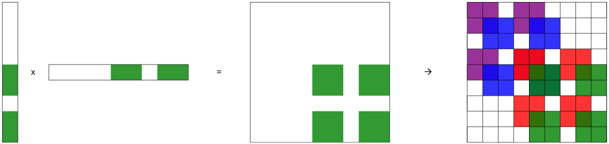
- but not always possible
- or often expensive

# Partitioned Hessian update

## Griewank & Toint, 1982

### Main idea

Update each $H_i$ rather than $H$




Require the knowledge of the gradient differences decomposition

- but not always possible
- or often expensive

# Partitioned Hessian update

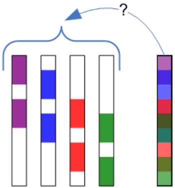
Griewank & Toint, 1982

## Main idea

Update each $H_i$ rather than $H$

Require the knowledge of the gradient differences decomposition

- but not always possible
- or often expensive



# Partially separable PSB

## Variational approach

$$\min_{H_i^+, y_i} \frac{1}{2} \sum \|H_i^+ - H_i\|_F^2$$

$$\text{s.t.} \begin{cases} H_i^+ s_i = y_i & \text{[secant equations]} \\ J_i \bullet H_i^+ = H_i^+ & \text{[sparsity of } H_i^+\text{]} \\ E_i^T = E_i := H_i^+ - H_i & \text{[symmetric correction]} \\ \sum y_i = y & \text{[gradient decomposition]} \\ I_i y_i = y_i & \text{[sparsity of } y_i\text{]} \end{cases}$$

with $J_i = e_i e_i^T$ and $I_i = diag(e_i)$, where

$$[e_i]_j = \begin{cases} 1 & \text{if } f_i \text{ depends on the } j\text{-th component} \\ 0 & \text{otherwise} \end{cases}$$

# Partially separable PSB

## Algorithm

### Partially separable PSB

1. Set $S = \sum(\|s_i\|^2 I_i + s_i s_i^T)$
2. Solve the positive definite system $S\lambda = y - Hs$ for $\lambda$
3. Update $H^+ = H + \sum(\lambda_i s_i^T + s_i \lambda_i^T)$

- do not include in the summation $i$ for which $\|s_i\| \approx 0$
- implicit computation of $H_i^+$ and $y_i$
- equivalence between some weighted sparse PSB and partially separable PSB
- no guarantee of positive definiteness

# Partitioned BFGS

1. Motivation

2. Structure preserving Hessian approximation schemes
   - Finite-difference methods
   - Sparse PSB
   - Partially separable PSB
   - Partitioned BFGS

3. Numerical experience

4. Conclusion and perspectives

# Partitioned BFGS

Other possibilities to split $y$ into $y_i$ to update $H$?

## Straightforward solution

1. Split uniformly $y$ into $y_i$
2. Apply BFGS on each element Hessian $H_i$ using these $y_i$

Still no guarantee of positive definiteness: $\mu_i := \langle s_i, y_i \rangle$ may be negative

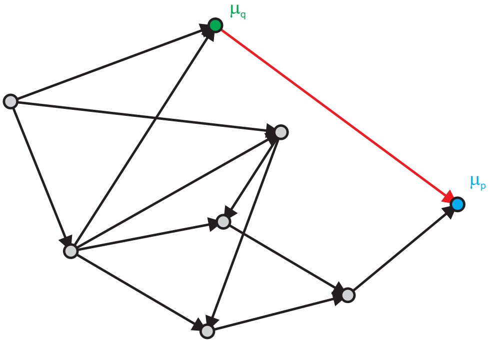
## Poised solution

1. Split uniformly $y$ into $y_i$
2. Perform some poising process on the $\mu_i$
3. Apply BFGS on each element Hessian $H_i$ using these $y_i$

# Partitioned BFGS

## Network Flow

Represent Hessian blocks on a network:

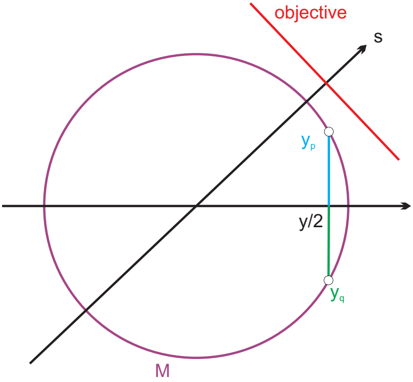



- nodes: each block, with value $\mu_i$
- arcs: binding blocks sharing at least a component, from larger $\mu_q$ to smaller $\mu_p$

# Partitioned BFGS

## Transferring curvature

Now perform transfers along arcs to poise the $\mu_i$



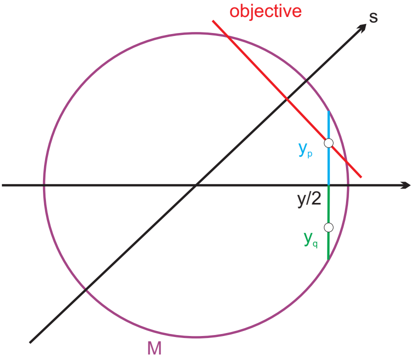

aim: increase $\mu_p$ to obtain some prescribed objective

constraint: summation ($y_p + y_q = y$), max. norm on $y_p, y_q$

# Partitioned BFGS

## Transferring curvature

Now perform transfers along arcs to poise the $\mu_i$



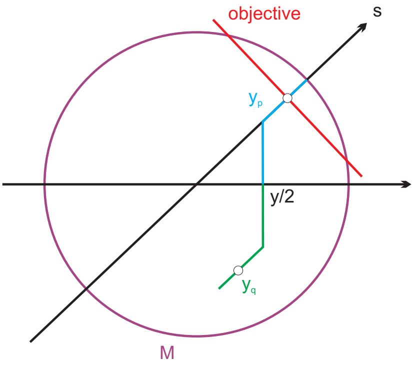

aim: increase $\mu_p$ to obtain some prescribed objective

constraint: summation ($y_p + y_q = y$), max. norm on $y_p, y_q$

# Partitioned BFGS

## Transferring curvature

Now perform transfers along arcs to poise the $\mu_i$




aim: increase $\mu_p$ to obtain some prescribed objective

constraint: summation ($y_p + y_q = y$), max. norm on $y_p, y_q$

# Partitioned BFGS
## Push-relabel algorithm (Goldberg & Tarjan, 1986)

Consider the problem as a maximal flow problem
from sources (with large $\mu_i$) to sinks (with small $\mu_i$)

### Push-relabel

Given:

- distance label for each node,
- a processing order for nodes,

perform at each active node ($\mu_i$ not poised):

push: transfer some amount of curvature between adjacent nodes with consecutive distance label

relabel: update current node label to make a push available

next: go to next node if node becomes inactive

# Numerical experience

1 Motivation

2 Structure preserving Hessian approximation schemes
- Finite-difference methods
- Sparse PSB
- Partially separable PSB
- Partitioned BFGS

3 Numerical experience

4 Conclusion and perspectives

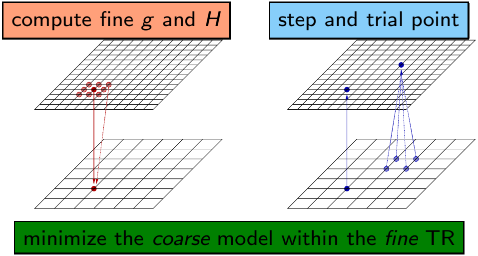
# Recursive Multilevel Trust-Region (RMTR)

Gratton, Sartenaer & Toint, 2005

## Trust-region framework

At each iteration, choose between:

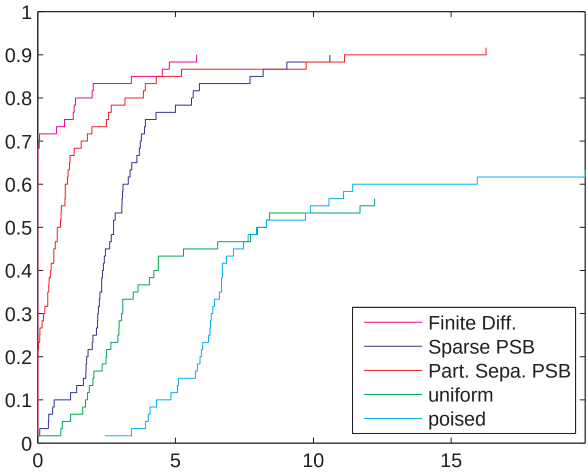
- a local Taylor model
- a model for a coarser description



# Preliminary numerical experience
## Implementation

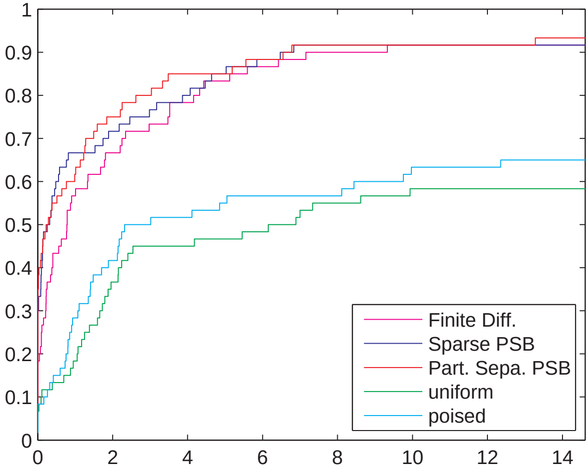
- Experience inside a multilevel algorithm: RMTR (Fortran 90/95)
- Galerkin model: $H_{down} = RH_{up}P$
- Use test problems from the RMTR paper and MINPACK-2 collection (size between 225 and 261121)
- $H_0 = \sum I_i$
- Preconditioned CG used in both PSB updates
- Lowest label ordering based on distance label to the sink, used in poised partitioned BFGS with some heuristics to accept or refuse the poised $y_i$ (based on improvement of $\mu_i$ and non-degradation of element secant equation)

# Performance profile
## CPU time



# Performance profile

## Function and gradient evaluations ($\#f + 5\#g$)



# Conclusion and perspectives


1. Motivation

2. Structure preserving Hessian approximation schemes
   - Finite-difference methods
   - Sparse PSB
   - Partially separable PSB
   - Partitioned BFGS

3. Numerical experience

4. Conclusion and perspectives

# Conclusion – Perspectives

- Structured finite differences efficient
- Partially separable PSB competitive, and especially efficient with costly gradient and larger problems
- Exploiting partial separability appears to be a bit more efficient than only sparsity

- more test problems needed
- hardest and more expensive problems
- investigation of limited memory multisecant Hessian update for a better integration to multilevel algorithm

# Thanks for your attention

# Questions?