The Hessian Module

- **Current status and future perspectives**
- **Reference:**

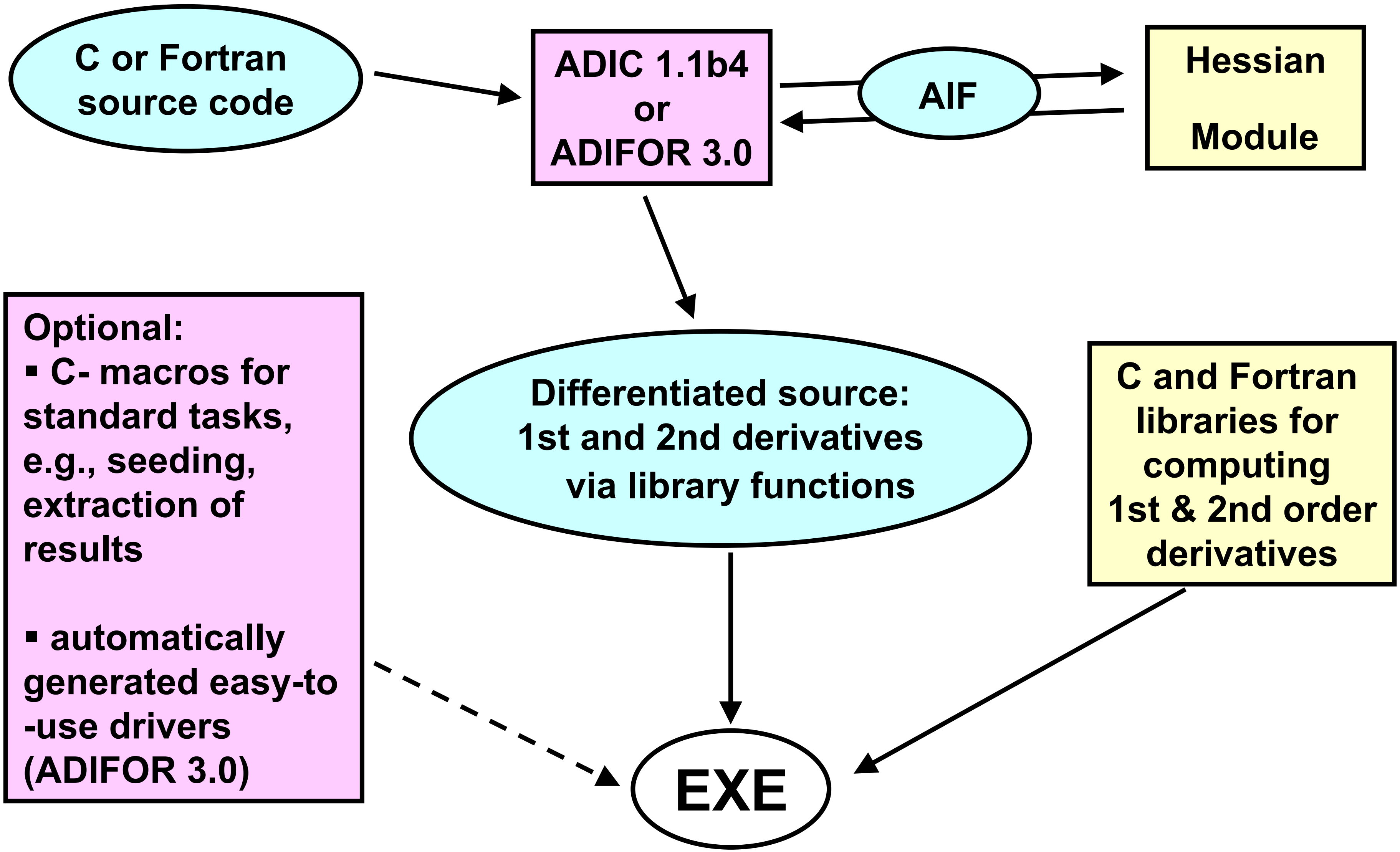
J. Abate, C. Bischof, A. Carle, L. Roh:

Algorithms and Design for a Second-Order Automatic Differentiation Module

Int. Symposium on Symbolic and Algebraic Computing (ISSAC) , 1997

- **Presentation by: Arno Rasch, RWTH Aachen University**

The Hessian Module



(1) Global forward mode

- **Break up statements into elementary unary/binary operations:**
- **compute ∇z , $\nabla^2 z$ in special subroutine:**

$$z = f(x, y)$$

n = # indep. vars

q = (n+1)·n / 2

$$\nabla z = \frac{\partial z}{\partial x} \nabla x + \frac{\partial z}{\partial y} \nabla y$$

$$\begin{aligned} \nabla^2 z &= \frac{\partial z}{\partial x} \nabla^2 x + \frac{\partial z}{\partial y} \nabla^2 y \\ &+ \frac{\partial^2 z}{\partial x^2} (\nabla x \cdot \nabla x^T) + \frac{\partial^2 z}{\partial y^2} (\nabla y \cdot \nabla y^T) \\ &+ \frac{\partial^2 z}{\partial x \partial y} (\nabla x \cdot \nabla y^T + \nabla y \cdot \nabla x^T) \end{aligned}$$

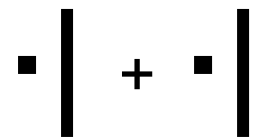
- **Hessians are stored in LAPACK packed symmetric scheme (size: q)**

(1) Global forward mode

- **Break up statements into elementary unary/binary operations:**
- **compute ∇z , $\nabla^2 z$ in special subroutine:**

$$z = f(x,y)$$

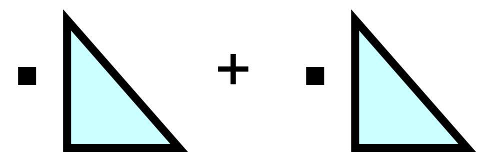
$$\nabla z = \frac{\partial z}{\partial x}\nabla x + \frac{\partial z}{\partial y}\nabla y$$

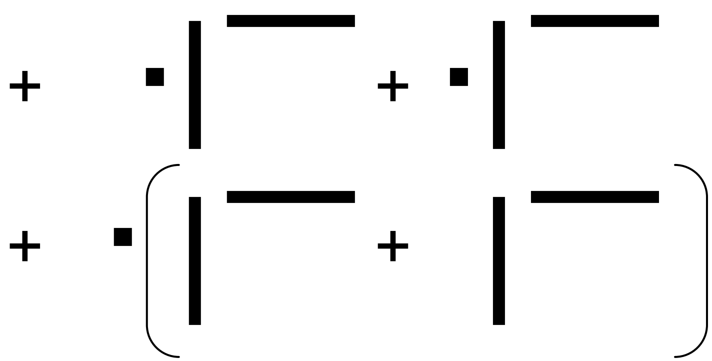


n = # indep. vars

q = (n+1)·n / 2

$$\nabla^2 z = \frac{\partial z}{\partial x}\nabla^2 x + \frac{\partial z}{\partial y}\nabla^2 y$$





$$+ \frac{\partial^2 z}{\partial x^2}(\nabla x \cdot \nabla x^T) + \frac{\partial^2 z}{\partial y^2}(\nabla y \cdot \nabla y^T)$$

$$+ \frac{\partial^2 z}{\partial x \partial y}(\nabla x \cdot \nabla y^T + \nabla y \cdot \nabla x^T)$$

- **Hessians are stored in LAPACK packed symmetric scheme (size: q)**
- **Smart implementation requires (6n+4q) FP-mult , (3n+3q) FP-add.**

(1) Global forward mode

- **Depending on elementary function f often fewer operations necessary, e.g., $f =$ „multiplication" with both variables active:**

$$\nabla z = \frac{\partial z}{\partial x}\nabla x + \frac{\partial z}{\partial y}\nabla y$$

$$\frac{\partial z^2}{\partial x^2} = 0 = \frac{\partial z^2}{\partial y^2} \quad , \quad \frac{\partial z^2}{\partial x \partial y} = 1$$

$$\nabla^2 z = \frac{\partial z}{\partial x}\nabla^2 x + \frac{\partial z}{\partial y}\nabla^2 y + \cancel{\frac{\partial^2 z}{\partial x^2}(\nabla x \cdot \nabla x^T)} + \cancel{\frac{\partial^2 z}{\partial y^2}(\nabla y \cdot \nabla y^T)} + \frac{\partial^2 z}{\partial x \partial y}(\nabla x \cdot \nabla y^T + \nabla y \cdot \nabla x^T)$$

- **With $f =$ „multiplication", and 2 active vars:**
  - **(n+4q) FP-mult**
  - **(n+3q) FP-add.**

• Depending on the activity information of arguments (i.e., 1st active / 2nd active / 1st and 2nd active) special routines for computing gradients and Hessians of binary functions $f \in \{+, -, *, /\}$ are used.

(2) Preaccumulation

- **Preaccumulation on statement level;** $z=f(x_1 \dots x_k)$
- **compute local gradients and Hessians w.r.t.** $x_1 \dots x_k$**, i.e.,**
- **in practice:** $k \leq 5$

$$\frac{\partial z}{\partial x_i}, \frac{\partial^2 z}{\partial x_i^2}, \frac{\partial^2 z}{\partial x_i \partial x_j} \qquad \begin{array}{l} i=1,\dots,k \\ j > i \end{array}$$

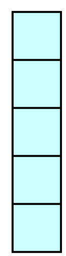
- **Computation of global gradients/Hessians („recombination“) by:**

$$\nabla z = \sum_{i=1}^{k} \frac{\partial z}{\partial x_i} \nabla x_i$$

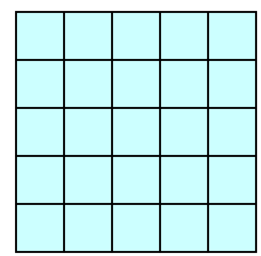
$$\begin{aligned} \nabla^2 z &= \sum_{i=1}^{k} \frac{\partial z}{\partial x_i} \nabla^2 x_i + \sum_{i=1}^{k} \frac{\partial^2 z}{\partial x_i^2} (\nabla x_i \cdot \nabla x_i^T) \\ &\quad + \sum_{i=1}^{k} \sum_{j=i+1}^{k} \frac{\partial^2 z}{\partial x_i \partial x_j} (\nabla x_i \cdot \nabla x_j^T + \nabla x_j \cdot \nabla x_i^T) \end{aligned}$$

(2) Preaccumulation

- $z = f(x_1 \ldots x_k)$
- **In practice:** $k \leq 5$ $\Rightarrow$ **shapes of local derivative objects are:**



and



- **Preaccumulation of local derivatives either in Forward mode, or by propagation of Taylor coefficients**
- **Preaccumulation and recombination by special sequence of subroutine calls, depending on the sparsity pattern of the local 5x5 Hessian**

FM preaccumulation – example 1

$$lg_1 = (1,0,0,0,0)^T$$

$$t = a \cdot a$$

$$\boxed{\begin{aligned} lh_2 &= lg_1 \cdot lg_1^T + lg_1 \cdot lg_1^T \\ lg_2 &= \frac{\partial t}{\partial a} \cdot lg_1 + \frac{\partial t}{\partial a} \cdot lg_1 \end{aligned}}$$

Original statement was: $f = a \cdot a \cdot a$

Performed by `fpinit`:
Special subroutine for initializing local gradients

Performed by `fpmula3`:
Special preaccumulation routine for multiplication where the local Hessians of both arguments are known to be zero

FM preaccumulation – example 1

Original statement was: $f = a \cdot a \cdot a$

$$lg_1 = (1,0,0,0,0)^T$$

$$t = a \cdot a$$

$$lh_2 = lg_1 \cdot lg_1^T + lg_1 \cdot lg_1^T$$

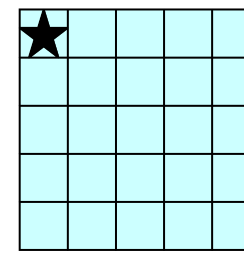
$$lg_2 = \frac{\partial t}{\partial a} \cdot lg_1 + \frac{\partial t}{\partial a} \cdot lg_1$$

$$f = t \cdot a$$

$$lh_3 = \frac{\partial f}{\partial t} \cdot lh_2 + lg_1 \cdot lg_2^T + lg_2 \cdot lg_1^T$$

$$lg_3 = \frac{\partial f}{\partial t} \cdot lg_2 + \frac{\partial f}{\partial a} \cdot lg_1$$

$lh_3 =$



Performed by `fpmula1`:
Special preaccumulation routine for multiplication where the local Hessian of the 2nd argument is known to be zero

FM preaccumulation – example 1

Original statement was: $f = a \cdot a \cdot a$

$$lg_1 = (1,0,0,0,0)^T$$

$$t = a \cdot a$$

$$lh_2 = lg_1 \cdot lg_1^T + lg_1 \cdot lg_1^T$$

$$lg_2 = \frac{\partial t}{\partial a} \cdot lg_1 + \frac{\partial t}{\partial a} \cdot lg_1$$

$$f = t \cdot a$$

$$lh_3 = \frac{\partial f}{\partial t} \cdot lh_2 + lg_1 \cdot lg_2^T + lg_2 \cdot lg_1^T$$

$$lg_3 = \frac{\partial f}{\partial t} \cdot lg_2 + \frac{\partial f}{\partial a} \cdot lg_1$$

$$\boxed{\nabla^2 f = lg_3[1] \cdot \nabla^2 a}$$

$$lh_3 = \begin{bmatrix} \star & & & & \\ & & & & \\ & & & & \\ & & & & \\ & & & & \end{bmatrix}$$

Performed by `accumhg1` :
Update of global Hessian using one local gradient.

For updating the global Hessian due to 2,3,4,5 local gradients, routines `accumhg[2,3,4,5]` are used

FM preaccumulation – example 1

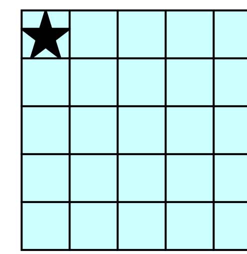
Sparsity information of the local Hessian is known. → use it in the update of the global Hessian

Two extreme possibilities:

1. **One single accumulation routine for the whole local Hessian → many checks at runtime**
2. **One subroutine call per single non-zero entry → more subroutine calls, memory accesses**

Current implementation makes a compromise by generating a sequence of routines from the set `accumh[1,2,3,4,5,6,7,8]` where at most 3 runtime checks per subroutine are performed.

ment was: $f = a \cdot a \cdot a$

$lh_3 =$ 

Performed by `accumh1`: Update of one diagonal entry in global Hessian due to the local Hessian

$$\nabla^2 f = lg_3[1] \cdot \nabla^2 a$$

$$\boxed{\nabla^2 f = \nabla^2 f + lh_3[1,1] \cdot \nabla a \cdot \nabla a^T}$$

$$\nabla f = lg_3[1] \cdot \nabla a$$

FM preaccumulation – example 1

$$lg_1 = (1,0,0,0,0)^T$$

$$t = a \cdot a$$

$$lh_2 = lg_1 \cdot lg_1^T + lg_1 \cdot lg_1^T$$

$$lg_2 = \frac{\partial t}{\partial a} \cdot lg_1 + \frac{\partial t}{\partial a} \cdot lg_1$$

$$f = t \cdot a$$

$$lh_3 = \frac{\partial f}{\partial t} \cdot lh_2 + lg_1 \cdot lg_2^T + lg_2 \cdot lg_1^T$$

$$lg_3 = \frac{\partial f}{\partial t} \cdot lg_2 + \frac{\partial f}{\partial a} \cdot lg_1$$

$$\boxed{\begin{aligned}\nabla^2 f &= lg_3[1] \cdot \nabla^2 a \\ \nabla^2 f &= \nabla^2 f + lh_3[1,1] \cdot \nabla a \cdot \nabla a^T\end{aligned}}$$

$$\nabla f = lg_3[1] \cdot \nabla a$$

Original statement was: $f = a \cdot a \cdot a$

Without pre-accumulation, computing $\nabla^2 f$ would cost:

8q Mult, 6q Add.

n = # indep. vars

q = (n+1)·n / 2

For large n, these operations are expensive

1q Mult.

2q Mult , 1q Add.

3q Mult , 1q Add.

FM preaccumulation – example 2

$$lg_1 = (1,0,0,0,0)^T,\ lg_2 = (0,0,1,0,0)^T,\ lg_3 = (0,0,1,0,0)^T$$

$$t = a \cdot b$$

Original statement was: $f = a \cdot b \cdot c$

$$lh_4 = lg_1 \cdot lg_2^T + lg_2 \cdot lg_1^T$$

$$lg_4 = \frac{\partial t}{\partial a} \cdot lg_1 + \frac{\partial t}{\partial b} \cdot lg_2$$

$$f = t \cdot c$$

$$lh_5 = \frac{\partial f}{\partial t} \cdot lh_4 + lg_3 \cdot lg_4^T + lg_4 \cdot lg_3^T$$

$$lg_5 = \frac{\partial f}{\partial t} \cdot lg_4 + \frac{\partial f}{\partial c} \cdot lg_3$$

$$lh_5 = \begin{bmatrix} & \star & \star & & \\ \star & & \star & & \\ \star & \star & & & \\ & & & & \\ & & & & \end{bmatrix}$$

FM preaccumulation – example 2

$$lg_1 = (1,0,0,0,0)^T,\ lg_2 = (0,0,1,0,0)^T,\ lg_3 = (0,0,1,0,0)^T$$

$$t = a \cdot b$$

Original statement was: $f = a \cdot b \cdot c$

$$lh_4 = lg_1 \cdot lg_2^T + lg_2 \cdot lg_1^T$$

$$lg_4 = \frac{\partial t}{\partial a} \cdot lg_1 + \frac{\partial t}{\partial b} \cdot lg_2$$

$$f = t \cdot c$$

$$lh_5 = \frac{\partial f}{\partial t} \cdot lh_4 + lg_3 \cdot lg_4^T + lg_4 \cdot lg_3^T$$

$$lh_5 = \begin{bmatrix} & \star & \star & & \\ \star & & \star & & \\ \star & \star & & & \\ & & & & \\ & & & & \end{bmatrix}$$

$$lg_5 = \frac{\partial f}{\partial t} \cdot lg_4 + \frac{\partial f}{\partial c} \cdot lg_3$$

$$\nabla^2 f = lg_5[1] \cdot \nabla^2 a + lg_5[2] \cdot \nabla^2 b + lg_5[3] \cdot \nabla^2 c$$

$$\nabla^2 f = \nabla^2 f + lh_5[1,2] \cdot (\nabla a \cdot \nabla b^T + \nabla b \cdot \nabla a^T)$$

$$\nabla^2 f = \nabla^2 f + (lh_5[1,3] \cdot \nabla a + lh_5[2,3] \cdot \nabla b) \cdot \nabla c^T$$
$$+ lh_5[1,3] \cdot \nabla c \cdot \nabla a^T + lh_5[2,3] \cdot \nabla c \cdot \nabla b^T$$

performed by:
`accumhg3`
`accumh5`
`accumh4`

$$\nabla f = lg_5[1] \cdot \nabla a + lg_5[2] \cdot \nabla b + lg_5[3] \cdot \nabla c$$

FM preaccumulation – example 2

$$lg_1 = (1,0,0,0,0)^T,\ lg_2 = (0,0,1,0,0)^T,\ lg_3 = (0,0,1,0,0)^T$$

$$t = a \cdot b$$

Original statement was: $f = a \cdot b \cdot c$

$$lh_4 = lg_1 \cdot lg_2^T + lg_2 \cdot lg_1^T$$

$$lg_4 = \frac{\partial t}{\partial a} \cdot lg_1 + \frac{\partial t}{\partial b} \cdot lg_2$$

$$f = t \cdot c$$

$$lh_5 = \frac{\partial f}{\partial t} \cdot lh_4 + lg_3 \cdot lg_4^T + lg_4 \cdot lg_3^T$$

$$lg_5 = \frac{\partial f}{\partial t} \cdot lg_4 + \frac{\partial f}{\partial c} \cdot lg_3$$

Without pre-accumulation, computing $\nabla^2 f$ would cost:

8q Mult, 6q Add.

$$\nabla^2 f = lg_5[1] \cdot \nabla^2 a + lg_5[2] \cdot \nabla^2 b + lg_5[3] \cdot \nabla^2 c$$

3q Mult , 2q Add.

$$\nabla^2 f = \nabla^2 f + lh_5[1,2] \cdot (\nabla a \cdot \nabla b^T + \nabla b \cdot \nabla a^T)$$

3q Mult , 2q Add.

$$\nabla^2 f = \nabla^2 f + (lh_5[1,3] \cdot \nabla a + lh_5[2,3] \cdot \nabla b) \cdot \nabla c^T$$
$$+ lh_5[1,3] \cdot \nabla c \cdot \nabla a^T + lh_5[2,3] \cdot \nabla c \cdot \nabla b^T$$

5q Mult , 3q Add.

11q Mult , 7q Add.

$$\nabla f = lg_5[1] \cdot \nabla a + lg_5[2] \cdot \nabla b + lg_5[3] \cdot \nabla c$$

When use preaccumulation?

- Several switching strategies for preaccumulation, globally controlled by the user
- Switch to preaccumulation, if
  - there is more than one operator [OG1]
  - the number of operators is greater than the number of active variables on the RHS [OPVAR]
  - the number of operators is greater than the number of active variables on the RHS plus 1 [OPVAR1]
  - there are three or more active variables on the RHS [VG3]
- Performance model based on flop & memory access count
- Actual machine-specific timing information (currently broken)
- Switch off preaccumulation, if
  - the number of active variables on RHS is greater than five
  - Intrinsic functions (other than + – * /) are used

Preaccumulation & ADIC 1.1b4

- Interface to ADIC 1.1b4 and ADIFOR 3.0 is AIF
- Hessian module identifies variables by name
- In ADIC 1.1b4, every occurence of a RHS variable gets a new name:

`f = a*a*a` —ADIC 1.1b4→ **`f = a1*a2*a3`**

- Hessian Module can‘t recognize that **`a1,a2,a3`** are all the same

(3) Hessian-vector product (H · v)

$z = f(x, y)$

n = # indep. vars

k = # cols. v

$$
\begin{aligned}
\nabla z &= \frac{\partial z}{\partial x}\nabla x + \frac{\partial z}{\partial y}\nabla y \\
(v^T \cdot \nabla z) &= \frac{\partial z}{\partial x}(v^T \cdot \nabla x) + \frac{\partial z}{\partial y}(v^T \cdot \nabla y) \\
(\nabla^2 z \cdot v) &= \frac{\partial z}{\partial x}(\nabla^2 x \cdot v) + \frac{\partial z}{\partial y}(\nabla^2 y \cdot v) \\
&\quad + \frac{\partial^2 z}{\partial x^2}\left[\nabla x \cdot (v^T \cdot \nabla x)^T\right] + \frac{\partial^2 z}{\partial y^2}\left[\nabla y \cdot (v^T \cdot \nabla y)^T\right] \\
&\quad + \frac{\partial^2 z}{\partial x \partial y}\left[\nabla x \cdot (v^T \cdot \nabla y)^T + \nabla y \cdot (v^T \cdot \nabla x)^T\right]
\end{aligned}
$$

For every active variable x, propagate:

- g_x = (∇x , $v^T \cdot \nabla x$) : array of length **n+k** (k= #columns of *v*)
- h_x = $\nabla^2 x \cdot v$: 2D-array of dimension **(n,k)**

(4) Projected Hessians

- Symmetric projected Hessian ($v^T \cdot H \cdot v$, $v \in R^{n \times k}$, $H \in R^{n \times n}$):
  - use the same code as in standard forward mode
  - for every active variable x, propagate $g\_x = v^T \cdot \nabla x$ and $h\_x = v^T \cdot \nabla^2 x \cdot v$
  - symmetric storage scheme can be employed for h_x
  - size for g_x is: **k** , size for h_x is: **(k+1)*k / 2**
- Unsymmetric projected Hessian ($v^T \cdot H \cdot w$, $w \in R^{n \times m}$)
  - use the same code as for the Hessian-vector product
  - for every active variable x, propagate
    - $g\_x = (v^T \cdot \nabla x, \nabla x \cdot w)$ array of length **(k+m)**
    - $h\_x = v^T \cdot H \cdot w$ 2D-array of size **(k,m)**

(5) Global Taylor mode

- Propagate 1st and 2nd order Taylor coefficients $\widetilde{\nabla} z$, $\widetilde{\nabla}^2 z$
- Rules similar to Global forward mode:

$$z = f(x, y) \qquad \widetilde{\nabla} z = \frac{\partial z}{\partial x} \cdot \widetilde{\nabla} x + \frac{\partial z}{\partial y} \cdot \widetilde{\nabla} y$$

$$\widetilde{\nabla}^2 z = \frac{\partial z}{\partial x} \cdot \widetilde{\nabla}^2 x + \frac{\partial z}{\partial y} \cdot \widetilde{\nabla}^2 y$$
$$+ \frac{1}{2} \cdot \frac{\partial^2 z}{\partial x^2} \cdot \widetilde{\nabla} x \overset{el}{*} \widetilde{\nabla} x + \frac{1}{2} \cdot \ldots$$
$$+ \frac{\partial^2 z}{\partial x \partial y} \cdot \widetilde{\nabla} x \overset{el}{*} \widetilde{\nabla} y$$

- **Very efficient for sparse Hessians**
- **Sparsity pattern needed**

- For length-**n** gradients and sparse **n x n** – Hessians with **s** off-diagonal entries, **k=n+s** univariate Taylor series are required
- $\widetilde{\nabla} z, \widetilde{\nabla}^2 z$ are **k**-vectors

Future perspectives I

- **Parallel computation of (dense) Hessians using OpenMP:**
- **Work on Hessian objects (1D-packed arrays) explicitly distributed, schedule static**
- **Redundant computation of original function and gradients**
- **OpenMP‘s *orphaning* concept allows parallel constructs outside the lexical scope of a parallel region**

$\rightarrow$ **automatically generated wrapper with OpenMP directives (done)**

$\rightarrow$ **Additional OpenMP directives in the libraries that are actually computing 1st and 2nd order derivatives (done)**

$\rightarrow$ **Experimental implementation with ADIFOR 3.0: user invokes parallel AD-code the same way like serial code**

Future perspectives II

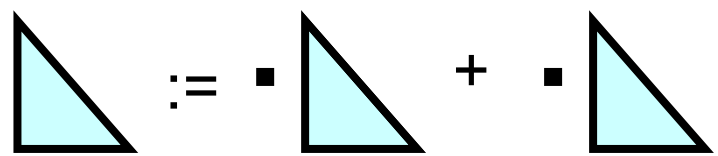
- **Partially separable function** $$f(x) = \sum_{i=1}^{m} f_i(x)$$
- **Element functions** $f_i(x)$ **have Hessians of rank < n**
- **Symmetric projection** $v^T \cdot \nabla^2 f(x) \cdot v$ **can be used to compute Hessians of element functions**
- **Parallelism**
- **Synchronization when updating global Hessian of** f
- **Sparsity patterns of** $\nabla^2 f_i(x)$ **must be available**

Future perspectives III

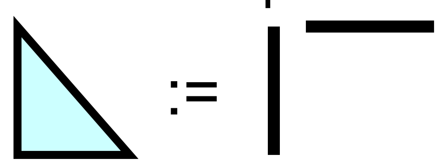
- Sparsity pattern of Hessian *not* available
- Use SparsLinC (Bischof, Khademi, Bouaricha, Carle) for computing sparse Hessians
- Since gradients and Hessians both are stored in 1D-Arrays, the standard SparsLinC operation

$$w = \sum_{i=1}^{N} a_i * v_i$$ with sparse vectors w and v_i

can be used for



- A new „SparsLinC-like“ routine for sparse symmetric outer product is needed



- Appropriate sequence of SparsLinC calls by the Hessian Module

Concluding Remarks

- Several strategies for computing Hessians:
  - Global forward mode , Global Taylor mode
    - without preaccumulation
    - with forward preaccumulation
    - with Taylor preaccumulation
  - Different switching strategies for preaccumulation
- Some ideas for future directions, e.g., parallel computation of Hessians
- Successfully computed 2nd derivatives for Aachen CFD code TFS („The Flow Solver") using ADIFOR 3.0 and the Hessian Module
- TFS consists of ~24,000 lines of (mostly) Fortran code, in 220 subroutines