Optimization of Sparse Generalized Jacobian Chain Products

Tamme Claus[†] and Uwe Naumann[‡]

[†] (tamme.claus@rwth-aachen.de)

[‡] supervisor, STCE (naumann@stce.rwth-aachen.de)

July 16, 2020

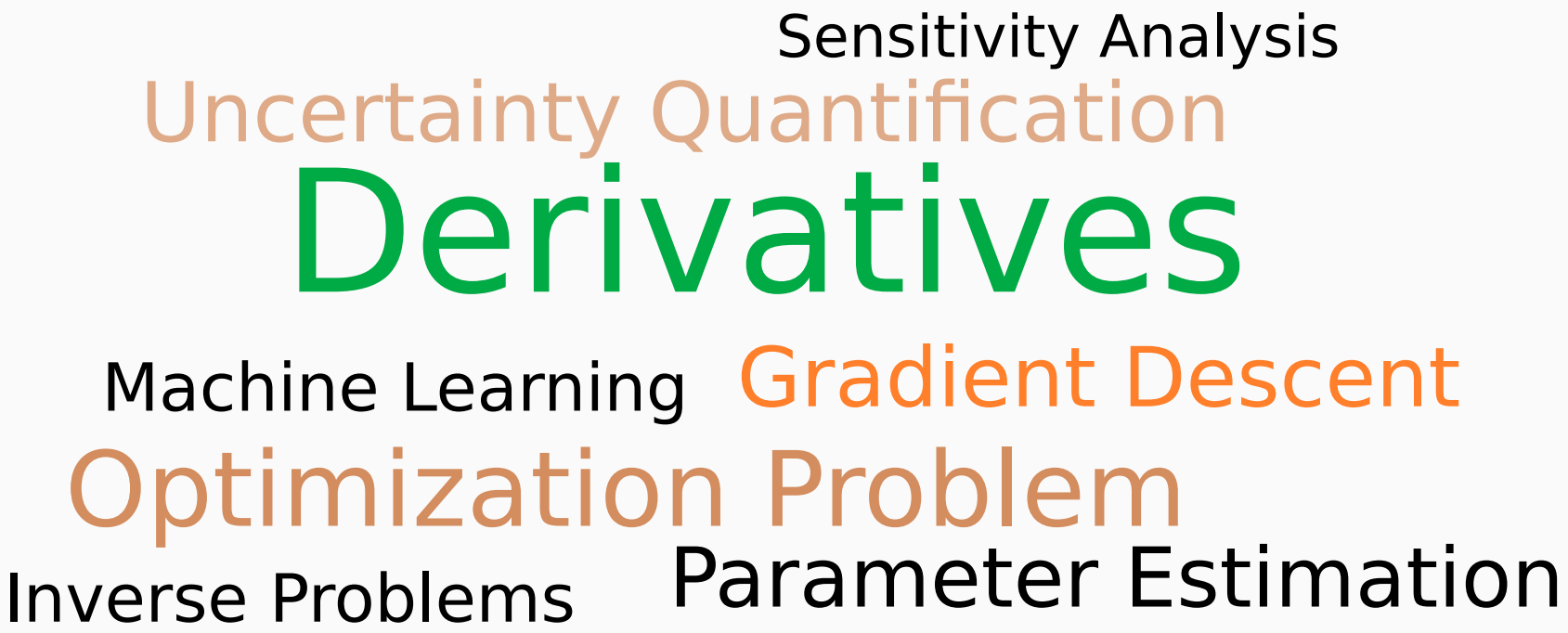
CES-Seminar, Master Computational Engineering Science,
RWTH Aachen University

What is Algorithmic Differentiation?

Sparse Generalized Jacobian Chain Product

Some Results

Motivation: Possible Applications of Algorithmic Differentiation



What is Algorithmic Differentiation?

Ways to calculate Derivatives

$$\mathbf{y} = F(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^m$$

- Symbolic differentiation / Differentiation by hand

$$F'_{i,j}(\mathbf{x}) = \lim_{h \to 0} \frac{F_i(\mathbf{x} + h\mathbf{e}_j) - F_i(\mathbf{x})}{h}$$

  $\to$ exact, but tedious for arbitrary (complex) F

- Finite differences

$$F'_{i,j}(\mathbf{x}; h) \approx \frac{F_i(\mathbf{x} + h\mathbf{e}_j) - F(\mathbf{x})}{h}$$

  $\to$ approximation, choice of h

- Algorithmic/Computational Differentiation

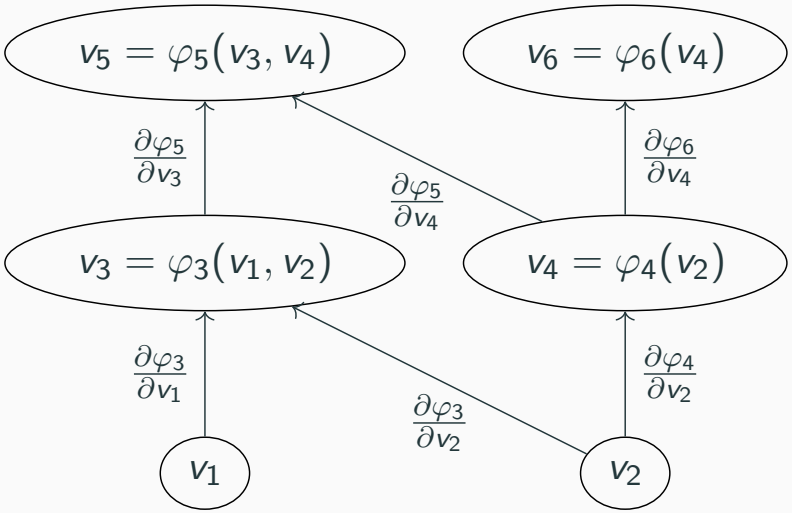
$$\dot{v}_j = \sum_{k \prec j} \frac{\partial \varphi_j}{\partial v_k} \dot{v}_k \qquad\qquad \bar{v}_k = \sum_{j \succ k} \frac{\partial \varphi_j}{\partial v_k} \bar{v}_j$$

  $\to$ what? but exact, and automatic

Algorithmic/Computational Differentiation

- subdivide F in **elemental functions**

$$v_j = \varphi_j(\{v_k\}_{k \prec j}) \quad v_i \in \mathbb{R}$$



- define **tangents** $\dot{v}$ and **adjoints** $\bar{v}$
- use the **chain rule**

$$\dot{v}_j = \sum_{k \prec j} \frac{\partial \varphi_j}{\partial v_k} \dot{v}_k \qquad \bar{v}_k = \sum_{j \succ k} \frac{\partial \varphi_j}{\partial v_k} \bar{v}_j$$

$k \prec j$: all v_k which v_j depends on (predecessors)
$j \succ k$: all v_j which depend on v_k (successors)

$\rightarrow$ still the goal: **Jacobian accumulation** but efficient!?

AD Mission Planning

Reducing `fmas` (fused multiply add-operations)

- occur frequently in AD
- native support by modern processors

---

the time spent on

finding the optimum + *optimal accumulation* < *naive accumulation*

---

graph reduction methods: vertex/edge/face elimination
fine level of granularity, but intractable for large functions/computer code

here: function compositions
coarser level of granularity, restriction of the underlying problem,

focus on solvability, still enormous fma reductions (see later)

Sparse Generalized Jacobian Chain Product

Problem Definition

Sparse Generalized Jacobian Chain Product

$$F = F_q \circ F_{q-1} \circ \ldots \circ F_1$$

$$F' = F'_q \cdot F'_{q-1} \cdot \ldots \cdot F'_1$$

Assume that for each $z_i = F_i(z_{i-1}) : \mathbb{R}^{n_i} \to \mathbb{R}^{m_i}$

- tangent $\dot{F}_i \cdot \dot{Z}_{i-1}$ implementations,
- adjoint $\bar{Z}_i \cdot \bar{F}_i$ implementations, and
- sparsity pattern $S_i \in \{0,1\}^{m_i \times n_i}$

are available.

What is the optimal way (minimum number of `fmas`) to accumulate F'? (without cost reductions "inside" F_i)

Individual Costs: Naive Accumulation

Costs to accumulate F_i' (naive)

- use tangent/adjoint implementation, "seed" unit vectors
- cost tangent: $n_i|E_i|$ ($|E_i|$ number of edges in the DAG of F_i)
- cost adjoint: $m_i|E_i|$

Individual Costs: Jacobian Compression (unidirectional)

Consider the sparsity pattern S_i of a factor F_i

$$S_i = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

Tangent Compression

$$S_i \underbrace{\begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}}_{T_i^{(s)}(n_i \times t_i^{(s)})} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$$

Adjoint Compression

$$\underbrace{\begin{pmatrix} 1 & 1 \end{pmatrix}}_{A_i^{(s)}(a_i^{(s)} \times m_i)} S_i = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix}$$

$\rightarrow$ graph coloring (heuristics)

Costs

$$\texttt{fma}_{i,i} = |E_i| \cdot \min\{\min_{s \in \mathcal{S}_i^a}\{a_i^{(s)}\}, \min_{s \in \mathcal{S}_i^t}\{t_i^{(s)}\}\}$$

Individual Costs: Sparse Matrix Mutliplication

Matrix Multiplication
Consider the sparsity pattern $A \in \{0,1\}^{m_2 \times n_2}$ and $B \in \{0,1\}^{m_1 \times n_1}$ of two matrices ($n_2 = m_1$)

$$\texttt{fma} = \sum_{i=1}^{m_2} \sum_{j=1}^{n_1} \sum_{k=1}^{n_2} A_{ik} B_{kj}$$

$\rightarrow$ we only count multiplication of two non-zeros (assuming a sparse implementation)

Matrix Recompression
Multiplication of a matrix with a seed matrix: no multiplications, only additions:

$\rightarrow$ count one *fma* per addition

Accumulation Options

Consider some subchain $F'_{j,i} \in \mathbb{R}^{m_j \times n_i}$

$$F'_{j,i} = \underbrace{F'_j \cdot \ldots \cdot F'_{k+1}}_{F'_{j,k+1}} \cdot \underbrace{F'_k \cdot \ldots \cdot F'_i}_{F'_{k,i}}$$

- **preaccumulate** both $F'_{j,k+1} \cdot F'_{k,i}$
  (accumulation of both Jacobians + matrix multiplication)
- **tangent** mode $\dot{F}_{j,i} \cdot T^{(s)}_{j,i}$ or $\dot{F}_{j,k+1} \cdot (F'_{k,i} \cdot T^{(s)}_{j,i})$
  (accumulation of Jacobian + matrix recompression + cost of tangent)
- **adjoint** mode $A^{(s)}_{j,i} \cdot \bar{F}_{j,i}$ or $(A^{(s)}_{j,i} \cdot F'_{j,k+1}) \cdot \bar{F}_{k,i}$
  (accumulation of Jacobian + matrix recompression + cost of adjoint)

Dynamic Programming
overlapping subproblems, optimal substructure

$\rightarrow$ optimal solution of smaller Jacobian chains may be reused

Dynamic Programming Recurrence

- **preaccumulate** both $F'_{j,k+1} \cdot F'_{k,i}$
- **tangent** mode $\dot{F}_{j,i} \cdot T^{(s)}_{j,i}$ or $\dot{F}_{j,k+1} \cdot (F'_{k,i} \cdot T^{(s)}_{j,i})$
- **adjoint** mode $A^{(s)}_{j,i} \cdot \bar{F}_{j,i}$ or $(A^{(s)}_{j,i} \cdot F'_{j,k+1}) \cdot \bar{F}_{k,i}$

Recurrence

$$\mathtt{fma}_{j,i} = \begin{cases} |E_j| \cdot \min\{\min\limits_{s\in\mathcal{S}^a_{j,j}}\{a^{(s)}_j\}, \min\limits_{s\in\mathcal{S}^t_{j,j}}\{t^{(s)}_j\}\} & i = j \\ \min\left\{ \min\{\min\limits_{s\in\mathcal{S}^a_{j,i}}\{a^{(s)}_{j,i}\}, \min\limits_{s\in\mathcal{S}^t_{j,i}}\{t^{(s)}_{j,i}\}\} \sum\limits_{\nu=i}^{k} |E_\nu|, \right. \\ \qquad \left. \min\limits_{i\le k<j} \left\{ \min \begin{Bmatrix} \mathtt{fma}_{j,k+1} + \mathtt{fma}_{k,i} + \mathtt{fma}_{j,k,i}, \\ \mathtt{fma}_{j,k+1} + \min\limits_{s\in\mathcal{S}^a_{j,i}}\{\mathtt{fma}^{(s)}_{j,k+1,i} + a^{(s)}_{j,i} \sum\limits_{\nu=i}^{k} |E_\nu|\}, \\ \mathtt{fma}_{k,i} + \min\limits_{s\in\mathcal{S}^t_{j,i}}\{\mathtt{fma}^{(s)}_{j,k,i} + t^{(s)}_{j,i} \sum\limits_{\nu=k+1}^{j} |E_\nu|\} \end{Bmatrix} \right\} \right\} & j > i \end{cases}$$

Restrictions

---

the time spent on

finding the optimum + *optimal accumulation* < *naive accumulation*

Balance between

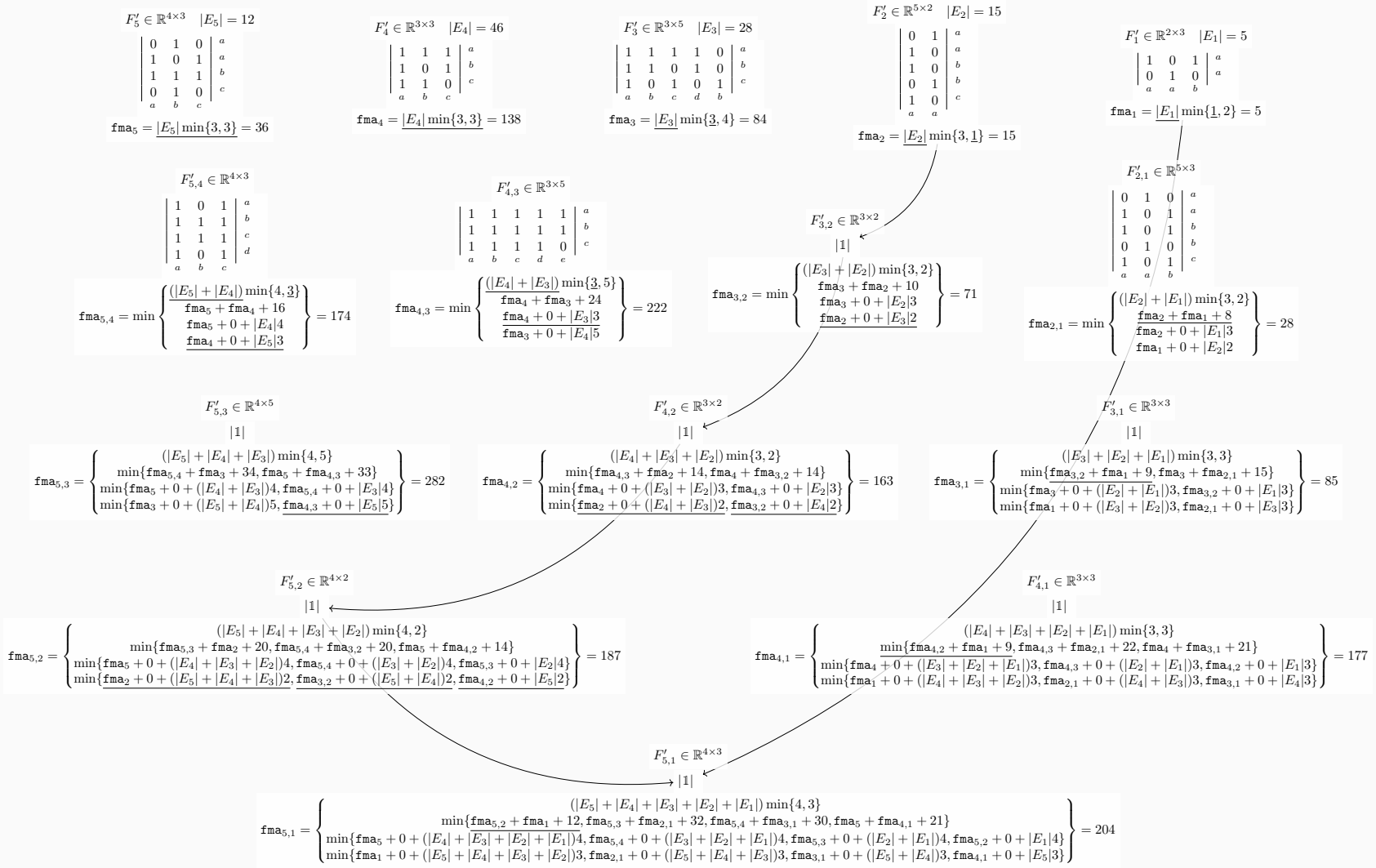
- rigorous minimal `fma`
- time spent optimizing

→ restrict to one Jacobian compression (e.g. the one found by a heuristic)

→ do not consider Jacobian compression

(see results later)

Example


$F'_5 \in \mathbb{R}^{4\times 3}$ $|E_5| = 12$
$\mathtt{fma}_5 = \underline{|E_5|}\min\{3,\underline{3}\} = 36$
$F'_4 \in \mathbb{R}^{3\times 3}$ $|E_4| = 46$
$\mathtt{fma}_4 = \underline{|E_4|}\min\{3,\underline{3}\} = 138$
$F'_3 \in \mathbb{R}^{3\times 5}$ $|E_3| = 28$
$\mathtt{fma}_3 = \underline{|E_3|}\min\{\underline{3},4\} = 84$
$F'_2 \in \mathbb{R}^{5\times 2}$ $|E_2| = 15$
$\mathtt{fma}_2 = \underline{|E_2|}\min\{3,\underline{1}\} = 15$
$F'_1 \in \mathbb{R}^{2\times 3}$ $|E_1| = 5$
$\mathtt{fma}_1 = \underline{|E_1|}\min\{\underline{1},2\} = 5$
$F'_{5,4} \in \mathbb{R}^{4\times 3}$
$\mathtt{fma}_{5,4} = \min\left\{\begin{array}{c}\underline{(|E_5|+|E_4|)\min\{4,\underline{3}\}}\\ \mathtt{fma}_5+\mathtt{fma}_4+16\\ \mathtt{fma}_5+0+|E_4|4\\ \underline{\mathtt{fma}_4+0+|E_5|3}\end{array}\right\} = 174$
$F'_{4,3} \in \mathbb{R}^{3\times 5}$
$\mathtt{fma}_{4,3} = \min\left\{\begin{array}{c}\underline{(|E_4|+|E_3|)\min\{\underline{3},5\}}\\ \mathtt{fma}_4+\mathtt{fma}_3+24\\ \underline{\mathtt{fma}_4+0+|E_3|3}\\ \mathtt{fma}_3+0+|E_4|5\end{array}\right\} = 222$
$F'_{3,2} \in \mathbb{R}^{3\times 2}$
$|1|$
$\mathtt{fma}_{3,2} = \min\left\{\begin{array}{c}\underline{(|E_3|+|E_2|)\min\{3,2\}}\\ \mathtt{fma}_3+\mathtt{fma}_2+10\\ \mathtt{fma}_3+0+|E_2|3\\ \underline{\mathtt{fma}_2+0+|E_3|2}\end{array}\right\} = 71$
$F'_{2,1} \in \mathbb{R}^{5\times 3}$
$\mathtt{fma}_{2,1} = \min\left\{\begin{array}{c}\underline{(|E_2|+|E_1|)\min\{3,2\}}\\ \underline{\mathtt{fma}_2+\mathtt{fma}_1+8}\\ \mathtt{fma}_2+0+|E_1|3\\ \mathtt{fma}_1+0+|E_2|2\end{array}\right\} = 28$
$F'_{5,3} \in \mathbb{R}^{4\times 5}$
$|1|$
$\mathtt{fma}_{5,3} = \left\{\begin{array}{c}(|E_5|+|E_4|+|E_3|)\min\{4,5\}\\ \min\{\mathtt{fma}_{5,4}+\mathtt{fma}_3+34,\mathtt{fma}_5+\mathtt{fma}_{4,3}+33\}\\ \min\{\mathtt{fma}_5+0+(|E_4|+|E_3|)4,\mathtt{fma}_{5,4}+0+|E_3|4\}\\ \min\{\mathtt{fma}_3+0+(|E_5|+|E_4|)5,\underline{\mathtt{fma}_{4,3}+0+|E_5|5}\}\end{array}\right\} = 282$
$F'_{4,2} \in \mathbb{R}^{3\times 2}$
$|1|$
$\mathtt{fma}_{4,2} = \left\{\begin{array}{c}(|E_4|+|E_3|+|E_2|)\min\{3,2\}\\ \min\{\mathtt{fma}_{4,3}+\mathtt{fma}_2+14,\mathtt{fma}_4+\mathtt{fma}_{3,2}+14\}\\ \min\{\mathtt{fma}_4+0+(|E_3|+|E_2|)3,\mathtt{fma}_{4,3}+0+|E_2|3\}\\ \min\{\underline{\mathtt{fma}_2+0+(|E_4|+|E_3|)2},\underline{\mathtt{fma}_{3,2}+0+|E_4|2}\}\end{array}\right\} = 163$
$F'_{3,1} \in \mathbb{R}^{3\times 3}$
$|1|$
$\mathtt{fma}_{3,1} = \left\{\begin{array}{c}(|E_3|+|E_2|+|E_1|)\min\{3,3\}\\ \min\{\underline{\mathtt{fma}_{3,2}+\mathtt{fma}_1+9},\mathtt{fma}_3+\mathtt{fma}_{2,1}+15\}\\ \min\{\mathtt{fma}_3+0+(|E_2|+|E_1|)3,\mathtt{fma}_{3,2}+0+|E_1|3\}\\ \min\{\mathtt{fma}_1+0+(|E_3|+|E_2|)3,\mathtt{fma}_{2,1}+0+|E_3|3\}\end{array}\right\} = 85$
$F'_{5,2} \in \mathbb{R}^{4\times 2}$
$|1|$
$\mathtt{fma}_{5,2} = \left\{\begin{array}{c}(|E_5|+|E_4|+|E_3|+|E_2|)\min\{4,2\}\\ \min\{\mathtt{fma}_{5,3}+\mathtt{fma}_2+20,\mathtt{fma}_{5,4}+\mathtt{fma}_{3,2}+20,\mathtt{fma}_5+\mathtt{fma}_{4,2}+14\}\\ \min\{\mathtt{fma}_5+0+(|E_4|+|E_3|+|E_2|)4,\mathtt{fma}_{5,4}+0+(|E_3|+|E_2|)4,\mathtt{fma}_{5,3}+0+|E_2|4\}\\ \min\{\underline{\mathtt{fma}_2+0+(|E_5|+|E_4|+|E_3|)2},\underline{\mathtt{fma}_{3,2}+0+(|E_5|+|E_4|)2},\underline{\mathtt{fma}_{4,2}+0+|E_5|2}\}\end{array}\right\} = 187$
$F'_{4,1} \in \mathbb{R}^{3\times 3}$
$|1|$
$\mathtt{fma}_{4,1} = \left\{\begin{array}{c}(|E_4|+|E_3|+|E_2|+|E_1|)\min\{3,3\}\\ \underline{\min\{\mathtt{fma}_{4,2}+\mathtt{fma}_1+9,\mathtt{fma}_{4,3}+\mathtt{fma}_{2,1}+22,\mathtt{fma}_4+\mathtt{fma}_{3,1}+21\}}\\ \min\{\mathtt{fma}_4+0+(|E_3|+|E_2|+|E_1|)3,\mathtt{fma}_{4,3}+0+(|E_2|+|E_1|)3,\mathtt{fma}_{4,2}+0+|E_1|3\}\\ \min\{\mathtt{fma}_1+0+(|E_4|+|E_3|+|E_2|)3,\mathtt{fma}_{2,1}+0+(|E_4|+|E_3|)3,\mathtt{fma}_{3,1}+0+|E_4|3\}\end{array}\right\} = 177$
$F'_{5,1} \in \mathbb{R}^{4\times 3}$
$|1|$
$\mathtt{fma}_{5,1} = \left\{\begin{array}{c}(|E_5|+|E_4|+|E_3|+|E_2|+|E_1|)\min\{4,3\}\\ \min\{\mathtt{fma}_{5,2}+\mathtt{fma}_1+12,\mathtt{fma}_{5,3}+\mathtt{fma}_{2,1}+32,\mathtt{fma}_{5,4}+\mathtt{fma}_{3,1}+30,\mathtt{fma}_5+\mathtt{fma}_{4,1}+21\}\\ \min\{\mathtt{fma}_5+0+(|E_4|+|E_3|+|E_2|+|E_1|)4,\mathtt{fma}_{5,4}+0+(|E_3|+|E_2|+|E_1|)4,\mathtt{fma}_{5,3}+0+(|E_2|+|E_1|)4,\mathtt{fma}_{5,2}+0+|E_1|4\}\\ \min\{\mathtt{fma}_1+0+(|E_5|+|E_4|+|E_3|+|E_2|)3,\mathtt{fma}_{2,1}+0+(|E_5|+|E_4|+|E_3|)3,\mathtt{fma}_{3,1}+0+(|E_5|+|E_4|)3,\mathtt{fma}_{4,1}+0+|E_5|3\}\end{array}\right\} = 204$

Some Results

Case Study

length q	max_mn	Tangent	Adjoint	Preaccumulation	Optimum
Dense					
10	10	296	2072	779	296
50	50	839925	604746	740518	135228
100	100	19160886	21611697	9034346	231143
150	150	56863838	47527984	53425505	852466
200	200	34648499	381133489	174604509	2041326
Sparse - no compression					
10	10	296	2072	681	296
50	50	839925	604746	620148	134838
100	100	19160886	21611697	8808143	231143
150	150	56863838	47527984	52552009	852466
200	200	34648499	381133489	172585171	2041326
Sparse - compression					
10	10	296	2072	345	296
50	50	839925	604746	36767	36132
100	100	19160886	21611697	237339	231143
150	150	56863838	47527984	867098	852466
200	200	34648499	381133489	2067283	2041326

Case Study

length q	max_mn	Tangent	Adjoint	Preaccumulation	Optimum
Dense					
10	10	296	2K	779	296
50	50	840K	605K	741K	135K
100	100	19M	22M	9M	231K
150	150	57M	48M	53M	852K
200	200	35M	381M	175M	2M
Sparse - no compression					
10	10	296	2K	681	296
50	50	840K	605K	620K	135K
100	100	19M	22M	9M	231K
150	150	57M	48M	53M	852K
200	200	35M	381M	173M	2M
Sparse - compression					
10	10	296	2K	345	296
50	50	840K	605K	37K	**36K**
100	100	19M	22M	237K	231K
150	150	57M	48M	867K	852K
200	200	35M	381M	2M	2M

Outlook

Conclusion

- enormous `fma` improvements possible compared to naive AD
- shows potential and necessity of *AD Mission Planning*
- exploitation of sparsity: effectiveness is problem dependent

Future work

- adaptive optimization finding balance between degrees of freedom and optimization time
- memory constraints for adjoints
- extension to different function structure (more than function compositions)

References

Gebremedhin, A., Manne, F., & Pothen, A. (2005). What color is your jacobian? graph coloring for computing derivatives. *SIAM review*, *47*, 629–705. doi:10.1137/S0036144504444711

Giering, R., & Kaminski, T. (2006). Automatic sparsity detection implemented as a source-to-source transformation. In V. N. Alexandrov, G. D. van Albada, P. M. A. Sloot, & J. Dongarra (Eds.), *Computational science – iccs 2006* (pp. 591–598). Berlin, Heidelberg: Springer Berlin Heidelberg.

Goedecker, S., & Hoisie, A. (2001). *Performance optimization of numerically intensive codes*. doi:10.1137/1.9780898718218. eprint: https://epubs.siam.org/doi/pdf/10.1137/1.9780898718218

Griewank, A. (2003). A mathematical view of automatic differentiation. *Acta Numerica*, *12*, 321–398. doi:10.1017/S0962492902000132

Griewank, A., & Mitev, C. (2002). Detecting jacobian sparsity patterns by bayesian probing. *Math. Program. 93*, 1–25. doi:10.1007/s101070100281

Griewank, A., & Naumann, U. (2002). Accumulating jacobians as chained sparse matrix products. *Math. Prog*, *3*, 2003.

Naumann, U. (2004). Optimal accumulation of jacobian matrices by elimination methods on the dual computational graph. *Math. Program. 99*, 399–421. doi:10.1007/s10107-003-0456-9

Naumann, U. (2008). Optimal jacobian accumulation is np-complete. *Mathematical Programming*, *112*, 427–441. doi:10.1007/s10107-006-0042-z

Naumann, U. (2011). *The art of differentiating computer programs*. doi:10.1137/1.9781611972078

Naumann, U. (2020a). On sparse matrix chain products. (pp. 118–127). doi:10.1137/1.9781611976229.12

Naumann, U. (2020b). Optimization of generalized jacobian chain products without memory constraints. arXiv: 2003.05755 [math.NA]

Pieterse, V., Kourie, D., Cleophas, L., & Watson, B. (2010). Performance of c++ bit-vector implementations. (pp. 242–250). doi:10.1145/1899503.1899530