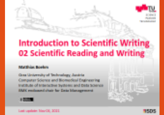
Seminar Large-scale Data Engineering (LDE)
02 Scientific Reading and Writing

Dr.-Ing. Patrick Damme
Technische Universität Berlin
Berlin Institute for the Foundations of Learning and Data
Big Data Engineering (DAMS Lab)

Last update: Oct 20, 2024

[**Credit:** Based on “Introduction to Scientific Writing”/
”02 Scientific Reading and Writing” by Matthias Boehm
(TU Graz, winter 2021/22)]



Introduction to Scientific Writing
02 Scientific Reading and Writing

Matthias Boehm

Graz University of Technology, Austria
Computer Science and Biomedical Engineering
Institute of Interactive Systems and Data Science
BMK endowed chair for Data Management

Last update: Mar 08, 2021

Announcements/Org

- **Hybrid Setting with Optional Attendance**
  - In-person in MAR 0.015
  - Virtual via zoom
    https://tu-berlin.zoom.us/j/67376691490?pwd=NmlvWTM5VUVWRjU0UGI2bXhBVkxzQT09

- **Reminder: Selection of Seminar and Project Topics Due Oct 31, 23:59**
  - **Polls in the ISIS course open now**
  - **Seminar:** 5 preferred topics/papers
  - **Project:** 5 preferred topics + preference on team/individual work + optionally team members

Agenda

- **Scientific Reading**
- **Scientific Writing**

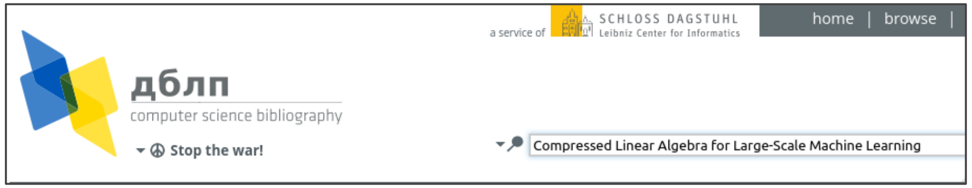
Scientific Writing skills can only be learned hands on, and incrementally improved with experience

Scientific Reading

In Computer Science (Data Management)

Obtaining the Full Text of a Paper

- **If you know the title[, author, venue, year] of a paper**
  - Use search engines like **DBLP** (https://dblp.uni-trier.de/) or **Google Scholar** (https://scholar.google.com/)
  - Make sure to select the **right version** of the paper



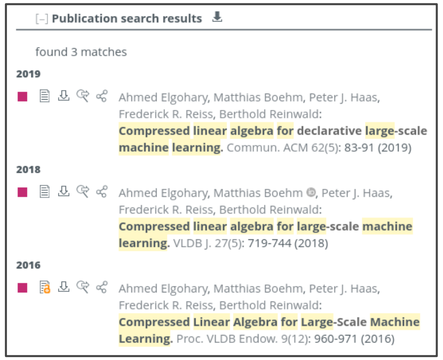
a service of SCHLOSS DAGSTUHL Leibniz Center for Informatics

home | browse |

дблп
computer science bibliography

Stop the war!

Compressed Linear Algebra for Large-Scale Machine Learning



[–] Publication search results

found 3 matches

2019

Ahmed Elgohary, Matthias Boehm, Peter J. Haas, Frederick R. Reiss, Berthold Reinwald: **Compressed linear algebra for declarative large-scale machine learning.** Commun. ACM 62(5): 83-91 (2019)

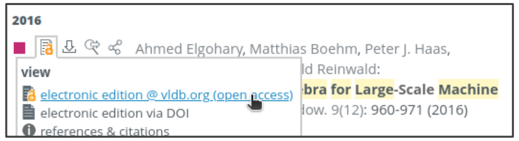
2018

Ahmed Elgohary, Matthias Boehm, Peter J. Haas, Frederick R. Reiss, Berthold Reinwald: **Compressed linear algebra for large-scale machine learning.** VLDB J. 27(5): 719-744 (2018)

2016

Ahmed Elgohary, Matthias Boehm, Peter J. Haas, Frederick R. Reiss, Berthold Reinwald: **Compressed Linear Algebra for Large-Scale Machine Learning.** Proc. VLDB Endow. 9(12): 960-971 (2016)

paper



2016

Ahmed Elgohary, Matthias Boehm, Peter J. Haas,
ld Reinwald:
bra for Large-Scale Machine
low. 9(12): 960-971 (2016)

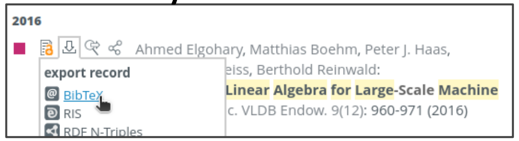
view

electronic edition @ vldb.org (open access)

electronic edition via DOI

references & citations

BibTeX entry



2016

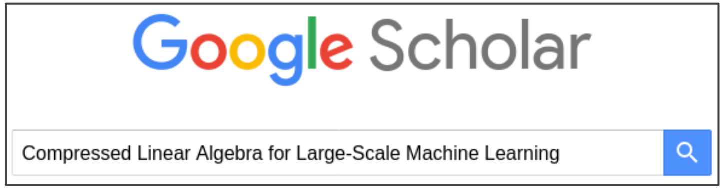
Ahmed Elgohary, Matthias Boehm, Peter J. Haas,
eiss, Berthold Reinwald:
Linear Algebra for Large-Scale Machine
c. VLDB Endow. 9(12): 960-971 (2016)

export record

BibTeX

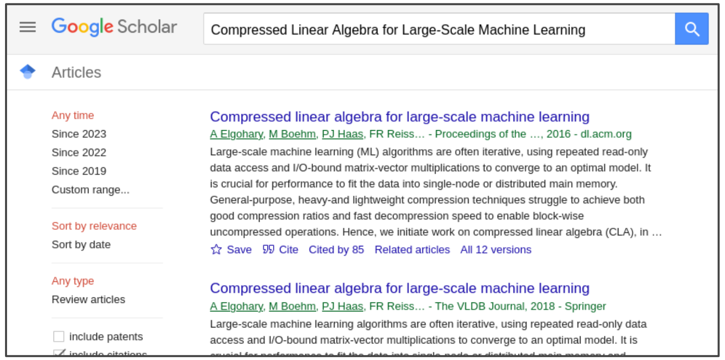
RIS

RDF N-Triples



Google Scholar

Compressed Linear Algebra for Large-Scale Machine Learning



Google Scholar

Compressed Linear Algebra for Large-Scale Machine Learning

Articles

Any time
Since 2023
Since 2022
Since 2019
Custom range...

Sort by relevance
Sort by date

Any type
Review articles

□ include patents
☑ include citations

Compressed linear algebra for large-scale machine learning
A Elgohary, M Boehm, PJ Haas, FR Reiss… - Proceedings of the …, 2016 - dl.acm.org
Large-scale machine learning (ML) algorithms are often iterative, using repeated read-only data access and I/O-bound matrix-vector multiplications to converge to an optimal model. It is crucial for performance to fit the data into single-node or distributed main memory. General-purpose, heavy-and lightweight compression techniques struggle to achieve both good compression ratios and fast decompression speed to enable block-wise uncompressed operations. Hence, we initiate work on compressed linear algebra (CLA), in …
☆ Save Cite Cited by 85 Related articles All 12 versions

Compressed linear algebra for large-scale machine learning
A Elgohary, M Boehm, PJ Haas, FR Reiss… - The VLDB Journal, 2018 - Springer
Large-scale machine learning algorithms are often iterative, using repeated read-only data access and I/O-bound matrix-vector multiplications to converge to an optimal model. It is crucial for performance to fit the data into single-node or distributed main memory and

- If the paper is **not open-access**, you typically can access the PDF when you
  - are in the university VPN (e.g., ACM Digital Library), or
  - log in with your university account (redirect to TUB login page) (e.g., IEEE Xplore)

Finding Related Work

- **Motivation**
  - Some research areas might be very large (e.g., index structures, compression)
  - How do you find relevant scientific papers/theses via **multiple channels**

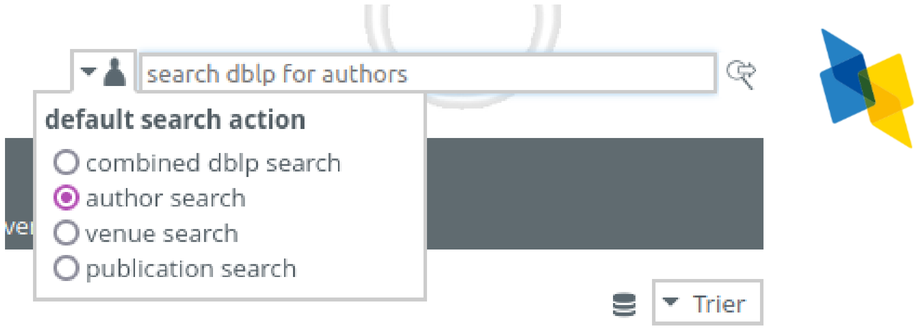
- **Prefer Trustworthy Sources**
  - Archival publications, awareness of peer-review
  - From right communities (e.g., ML systems vs ML algorithms)
  - Reputation of website, authors, etc.

- **Recap: Give Credit**
  - Cite broadly, **give credit to inspiring ideas**, create connections
  - Honestly acknowledge **limitations of your approach**

Finding Related Work, cont.

- **By Venue/Year**
  - Start off top-tier conferences/journals and find latest work
  - E.g., SIGMOD, PVLDB, CIDR, ICDE, EDBT, CIKM, …
  - These papers' related work should provide a good categorization and discussion of related work → recursive lookup

- **By Author**
  - Sometimes there are well-known experts in a certain sub-area
  - Find author publications via DBLP and other libraries



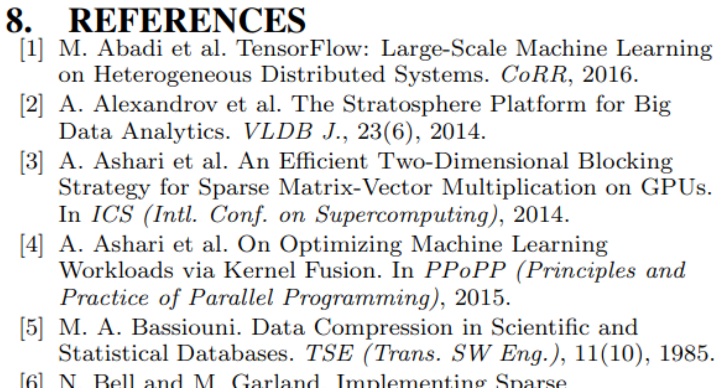
search dblp for authors

default search action

○ combined dblp search
● author search
○ venue search
○ publication search

Finding Related Work, cont.

- **By References**
  - Backwards (papers published before) & Forwards (papers published after the given paper)



8. REFERENCES

[1] M. Abadi et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *CoRR*, 2016.

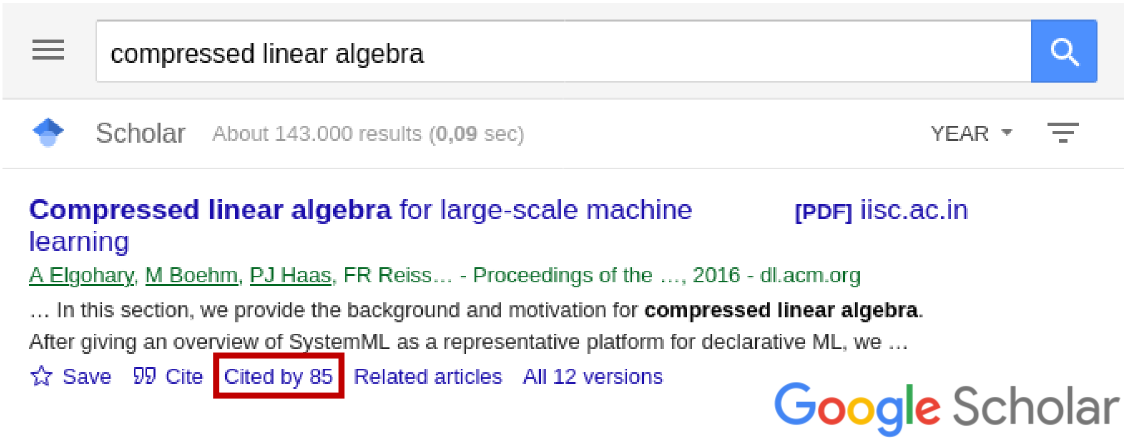
[2] A. Alexandrov et al. The Stratosphere Platform for Big Data Analytics. *VLDB J.*, 23(6), 2014.

[3] A. Ashari et al. An Efficient Two-Dimensional Blocking Strategy for Sparse Matrix-Vector Multiplication on GPUs. In *ICS (Intl. Conf. on Supercomputing)*, 2014.

[4] A. Ashari et al. On Optimizing Machine Learning Workloads via Kernel Fusion. In *PPoPP (Principles and Practice of Parallel Programming)*, 2015.

[5] M. A. Bassiouni. Data Compression in Scientific and Statistical Databases. *TSE (Trans. SW Eng.)*, 11(10), 1985.

[6] N. Bell and M. Garland. Implementing Sparse



compressed linear algebra

Scholar About 143.000 results (**0,09** sec) YEAR

Compressed linear algebra for large-scale machine learning [PDF] iisc.ac.in

A Elgohary, M Boehm, PJ Haas, FR Reiss… - Proceedings of the …, 2016 - dl.acm.org

… In this section, we provide the background and motivation for **compressed linear algebra**. After giving an overview of SystemML as a representative platform for declarative ML, we …

☆ Save 99 Cite Cited by 85 Related articles All 12 versions

Google Scholar

- **By Keywords**
  - Broad survey of other related work, to augment the bias of the year/venue/author approach
  - Think of possible synonyms (e.g., “extensible”, “extendable”, “customizable”, …)

Types of Reading

- **Skimming**
  - **Goal:** understand what the paper/thesis is about, judge relevance
  - Read abstract, and optionally introduction
  - Scan paper (sections/subsections, structure, figures)

What?

- **Understanding**
  - **Goal:** understand how the presented approach accomplishes the paper's goals
  - **#1** Skimming (see above)
  - **#2** Read the whole paper sequentially, add **notes/annotations**

How?

- **Reviewing**
  - **Goal:** evaluate potential impact, and limitations
  - **#1** Skimming (see above)
  - **#2** Understanding (see above) + strengths and weaknesses
  - **#3** Write summary, strong/weak points, detailed comments, constructive feedback, overall recommendation ([strong/weak] accept/reject)

Good enough?
How to improve?

Process of Reading – Skimming/Understanding

- **Abstract and Structure**

- **#1 Partial Reading** (mostly skimming)
  - Read into each paragraph until you get what it's about
  - 1st sentence/label: **topic sentence**

- **#2 Fast Reading**
  - Normal reading vs **reading w/o vocalization**
  - Avoid need for rereading text
    - Back/forward references,
    - Misplacement after distractions
    - Rereading due to lack of understanding

→ **Read according to your goals of reading**

Process of Reading – Understanding/Evaluation

- **Skepticism**
  - Critical reading is important for **understanding** and **evaluation**
  - **#1** Start open-minded, listen to arguments and trust provided evidence
  - **#2 Don’t accept** superficial, contradictory, or unproven claims
  - **#3** If there are problems, which **constructive feedback** could you give or how could the problems be addressed?

- **Questions to Ask Yourself?**
  - What is the problem? Is it a real or artificial problem?
  - How would you solve the problem yourself?
  - How is the paper solving the problem?
  - Is this the simplest approach that yields these results (justified complexity)
  - Are there limitations that are not covered by the paper?
  - Is there existing work that already addresses the same problem?

Proofreading Your Own Paper

- **#1 Read Slowly & Carefully**
  - **Problem:** Brain interpolates between words
  - Awareness of **common syntactic issues** (the the, missing/wrong articles, adapt/adopt)
  - Awareness of **common semantic issues** (missing reference, inconsistent / no logical consequence)

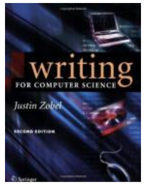
→ **Read out loud**
→ **Use PDF-to-Speech**

- **#2 Read Fully**
  - **Read and annotate issue**, don't fix immediately (destroys the flow)
  - Take annotated document and fix issues

- **#3 Ask Big Questions**
  - **Pitfall:** Being **overly focused on syntactic/local issues**
  - Is the overall idea clearly communicated and does it make sense?
  - Are there missing pieces, missing experiments, missing related work?

Scientific Writing

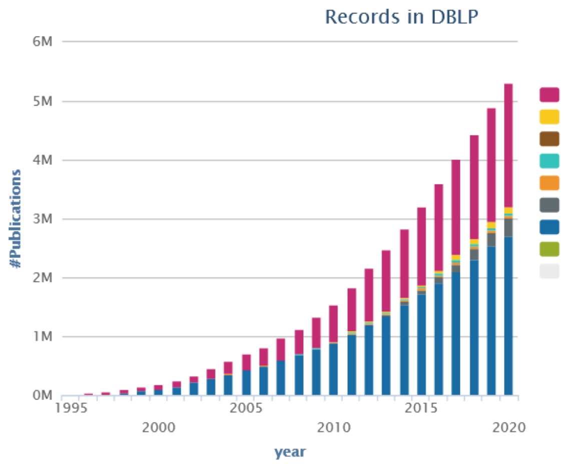
In Computer Science (Data Management)



[Justin Zobel: Writing for Computer Science,
2nd ed. Springer 2004, ISBN 978-1-85233-802-2]

Recap: Writing the Paper

- **Know Your Audience**
- **Get Your Workflow in Order / Incremental Paper Drafts**
- **Mindset: Quality over Quantity**
  - Aim for top-tier conferences/journals (act as filter)
  - Make the paper useful for others (ideas, evidence, code)



- **Make the Paper Easy to Read**
- **Present Your Work with appropriate Structure, Writing Style, and Formatting**

Recap: Prototypical Structure of a Scientific Paper

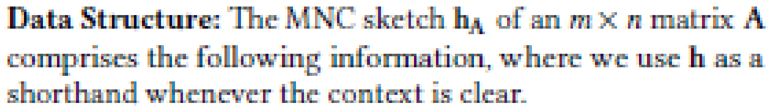
- **Sections and Subsections**
  - Abstract → short overview of problem and solution (part of meta data)
  - Introduction → context, problem, contributions
  - Background / Preliminaries → necessary background for understanding
  - Main Part → your technical core contributions
  - Main Part 2
  - Experiments → setting, micro benchmarks, end-to-end benchmarks
  - Related Work → areas of related work, differences to your own work
  - Conclusions → summary, conclusions, and future work
  - Acknowledgments → funding agencies, helpful people beyond co-authors
  - References → list of other works referenced throughout the paper
  - (Appendix) → any additional contents (e.g., proves of theorems, more results)

→ 01 Structure of Scientific Papers

- **Recommendations**
  - Avoid sections with only one subsection (e.g., 2 and 2.1)
  - Avoid more than two or at most three nesting levels
  - Clearly separate motivation/background from your own work

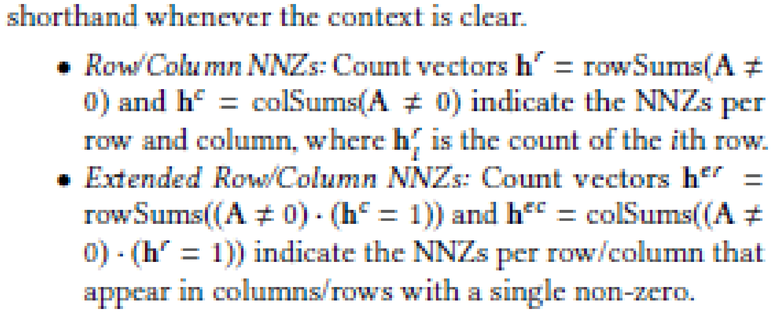
Formatting Elements Facilitating Structure

- Goal: Easy Skimming
  - Guide the readers' attention
- Paragraph Labels
  - `\paragraph{…}` …



Data Structure: The MNC sketch $\mathbf{h}_\mathbf{A}$ of an $m \times n$ matrix $\mathbf{A}$ comprises the following information, where we use $\mathbf{h}$ as a shorthand whenever the context is clear.

- Bullet Lists
  - `begin{itemize} … \item … \end{itemize}`
  - `begin{enumerate} … \item … \end{enumerate}`



shorthand whenever the context is clear.

- *Row/Column NNZs:* Count vectors $\mathbf{h}^r = \text{rowSums}(\mathbf{A} \neq 0)$ and $\mathbf{h}^c = \text{colSums}(\mathbf{A} \neq 0)$ indicate the NNZs per row and column, where $\mathbf{h}^r_i$ is the count of the ith row.
- *Extended Row/Column NNZs:* Count vectors $\mathbf{h}^{er} = \text{rowSums}((\mathbf{A} \neq 0) \cdot (\mathbf{h}^c = 1))$ and $\mathbf{h}^{ec} = \text{colSums}((\mathbf{A} \neq 0) \cdot (\mathbf{h}^r = 1))$ indicate the NNZs per row/column that appear in columns/rows with a single non-zero.

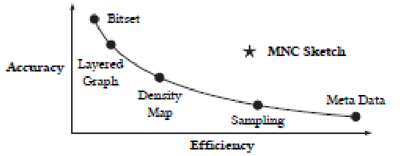


Figure 2: Accuracy/Efficiency Goal of the MNC Sketch.

- Figures and Tables
  - Should be self-explanatory
  - Captions above/below

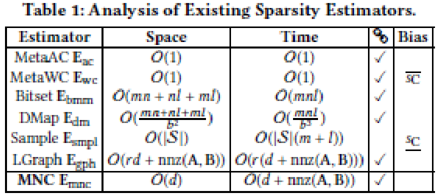
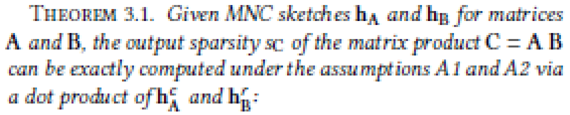


Table 1: Analysis of Existing Sparsity Estimators.

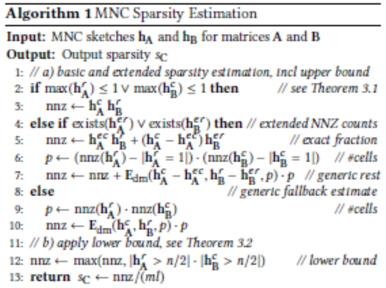
Estimator	Space	Time	%	Bias
MetaAC E_{ac}	$O(1)$	$O(1)$	✓	
MetaWC E_{wc}	$O(1)$	$O(1)$	✓	$\overline{s_C}$
Bitset $\mathrm{E}_{\mathrm{bmm}}$	$O(mn + nl + ml)$	$O(mnl)$	✓	
DMap E_{dm}	$O(\frac{mn+nl+ml}{b^2})$	$O(\frac{mnl}{b^3})$	✓	
Sample $\mathrm{E}_{\mathrm{smpl}}$	$O(\lvert S \rvert)$	$O(\lvert S \rvert (m + l))$		$\underline{s_C}$
LGraph $\mathrm{E}_{\mathrm{gph}}$	$O(rd + \text{nnz}(\mathbf{A}, \mathbf{B}))$	$O(r(d + \text{nnz}(\mathbf{A}, \mathbf{B})))$	✓	
MNC $\mathrm{E}_{\mathrm{mnc}}$	$O(d)$	$O(d + \text{nnz}(\mathbf{A}, \mathbf{B}))$	✓	

- Theorem/Definition/Example
  - Refine **`theorem`** environment as needed



Theorem 3.1. *Given MNC sketches $\mathbf{h}_\mathbf{A}$ and $\mathbf{h}_\mathbf{B}$ for matrices $\mathbf{A}$ and $\mathbf{B}$, the output sparsity s_C of the matrix product $\mathbf{C} = \mathbf{A}\,\mathbf{B}$ can be exactly computed under the assumptions A1 and A2 via a dot product of $\mathbf{h}^c_\mathbf{A}$ and $\mathbf{h}^r_\mathbf{B}$:*

- Algorithms/Pseudo-Code
  - Can be clearer than text, but not always
  - Carefully select the right level of abstraction



Algorithm 1 MNC Sparsity Estimation

```
Input: MNC sketches $\mathbf{h}_\mathbf{A}$ and $\mathbf{h}_\mathbf{B}$ for matrices $\mathbf{A}$ and $\mathbf{B}$
Output: Output sparsity $s_C$
// a) basic and extended sparsity estimation, incl upper bound
if $\max(\mathbf{h}^r_\mathbf{A}) \leq 1 \vee \max(\mathbf{h}^c_\mathbf{B}) \leq 1$ then        // see Theorem 3.1
    $nnz \leftarrow \mathbf{h}^c_\mathbf{A}\,\mathbf{h}^r_\mathbf{B}$
else if $\text{exists}(\mathbf{h}^{er}_\mathbf{A}) \vee \text{exists}(\mathbf{h}^{er}_\mathbf{B})$ then // extended NNZ counts
    $nnz \leftarrow \mathbf{h}^{ec}_\mathbf{A}\,\mathbf{h}^r_\mathbf{B} + (\mathbf{h}^c_\mathbf{A} - \mathbf{h}^{ec}_\mathbf{A})\,\mathbf{h}^{er}_\mathbf{B}$        // exact fraction
    $p \leftarrow (\text{nnz}(\mathbf{h}^r_\mathbf{A}) - \lvert \mathbf{h}^r_\mathbf{A} = 1 \rvert) \cdot (\text{nnz}(\mathbf{h}^c_\mathbf{B}) - \lvert \mathbf{h}^c_\mathbf{B} = 1 \rvert)$    // #cells
    $nnz \leftarrow nnz + \mathrm{E}_{\mathrm{dm}}(\mathbf{h}^c_\mathbf{A} - \mathbf{h}^{ec}_\mathbf{A}, \mathbf{h}^r_\mathbf{B} - \mathbf{h}^{er}_\mathbf{B}, p) \cdot p$  // generic rest
else                                        // generic fallback estimate
    $p \leftarrow \text{nnz}(\mathbf{h}^r_\mathbf{A}) \cdot \text{nnz}(\mathbf{h}^c_\mathbf{B})$                      // #cells
    $nnz \leftarrow \mathrm{E}_{\mathrm{dm}}(\mathbf{h}^c_\mathbf{A}, \mathbf{h}^r_\mathbf{B}, p) \cdot p$
// b) apply lower bound, see Theorem 3.2
$nnz \leftarrow \max(nnz, \lvert \mathbf{h}^r_\mathbf{A} > n/2 \rvert \cdot \lvert \mathbf{h}^c_\mathbf{B} > n/2 \rvert)$      // lower bound
return $s_C \leftarrow nnz/(ml)$
```

- **Refer to All Figures, Tables, Algorithms in the Text & Place Them Close to the Text**

Writing Style

- **Goal: Clear, Easy-to-Read Writing**
  - Avoid unnecessary formalism → as simple as possible

- **Formal Language**
  - Avoid contractions ("can't", "aren't", …)
  - No colloquial or slang words
- **Prefer Active Voice**
  - Easier to understand, shorter, more interesting
  - Use "we" over "I"
  - Don't directly address the reader (no "you")

✗ `In this section, the background and motivation for compressed linear algebra is introduced.`

✓ `In this section, we provide the background and motivation for compressed linear algebra.`

- **Prefer Present Tense**
  - Most content of a research paper can be described in present
  - Exceptions: user studies, (specific experimental setup), related work

Writing Style, cont.

- **Variation**
  - Diversity (structure, length of sentences/paragraphs, choice of words, sentence beginning) helps keeping the reader’s attention

✗ The system of rational numbers is incomplete. This was discovered 2000 years ago by the Greeks. The problem arises in squares with sides of unit length. The length of the diagonals of these squares is irrational. This discovery was a serious blow to the Greek mathematicians.

✓ The Greeks discovered 2000 years ago that the system of rational numbers is incomplete. The problem is that some quantities, such as the length of the diagonal of a square with unit sides, are irrational. This discovery was a serious blow to the Greek mathematicians.

- **Use of References**
  - **Don’t use references as nouns**
  - Use “et al.” for three or more authors
  - **Prefer primary sources**
  - Use \cite{key1,key2} for multiple sources

✗ Later, [40] investigated query processing on heavyweight Huffman coding schemes,

✓ Later, Raman and Swart investigated query processing on heavyweight Huffman coding schemes [40],

Writing Style, cont.

- **Singular/Plural and Articles**
  - Plural allows to drop articles

employ general-purpose compression techniques

employ a general-purpose compression technique ✓

- **Guarded Spaces**
  - Use guarded spaces for references that should not appear on a new line

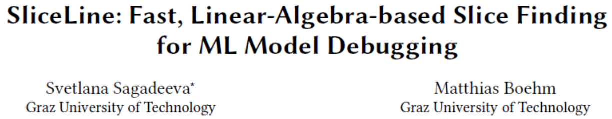
```
Figure~\ref{fig:exp1}
```

- **Clear References**
  - Make sure there are no unclear “it” or “this” references
  - Add descriptive nouns

Each entry q_i can be expressed over columns as $q_i = v^\top X_{i:}$. We rewrite **this** in […] ✗

Each entry q_i can be expressed over columns as $q_i = v^\top X_{i:}$. We rewrite **this multiplication** in […] ✓

- **Capitalize Titles and Names**
  - Titles: capitalize meaning-carrying words
  - Names: capitalize, e.g., Bayesian, Euclidean
  - References like Figure 1, Table 2, Section 3, Chapter 4, Equation 5 are names as well



SliceLine: Fast, Linear-Algebra-based Slice Finding for ML Model Debugging

Svetlana Sagadeeva*
Graz University of Technology

Matthias Boehm
Graz University of Technology

```
Figure~\ref{fig:exp1}
Equation~\eqref{eq:e1}
```

Punctuation

- **Commas**
  - Whenever a pause is appropriate, or required to avoid ambiguity

```
 When using disk[,] tree algorithms
were found to be particularly poor.
```

```
 A woman without her man is nothing.
A woman: without her, man is nothing.
```

  - **Lists:** red, blue, black, and white (Oxford/serial comma)
  - **Special sentence start:** `However,` `Hence,` `Therefore,` `In this paper,`
- **Semicolons**
  - Divide a long sentence into sub-sentences, or separation for emphasis
  - Lists with sublists

```
We use index structures like b-trees, tries,
  and hash tables; as well as compression
   techniques like run-length encoding,
 dictionary encoding, and null suppression.
```

- **Exclamations**
  - `Avoid exclamation marks! Never use more than one!!`

Diversity and Inclusion

[**Credit:** https://dbdni.github.io/]

- **Diversity, “the who”**
  - Individuals from a wide variety of backgrounds and experience, different viewpoints/reasoning/approaches
  - **Different cultures:** e.g., use names from variety of languages, cultures, nationalities (not just Alice and Bob)
  - **Differences in figures:** e.g., people-like icons: use variety of gender, skin color, ability status, …
  - **Gender diversity in pronouns:** use variety of he/she/they, use gender-neutral nouns: “chairman” ⟶ “chairperson”

- **Increasing Awareness for D&I**
  - Meanwhile part of the policies of all/most major publication venues (SIGMOD, VLDB, ICDE, EDBT, ADBIS, …)
  - **D&I issues included in the review form**

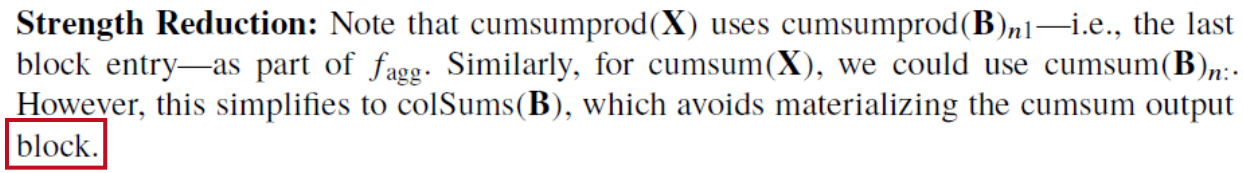
- **Inclusion, “the how”**
  - Environment welcoming and embracing diversity; **avoid** language that furthers the marginalization, stereotyping, erasure of any group of people
  - **Implicit assumptions:** "Everyone has a mother and a father."
  - **Oppressive terminology:** e.g.,
    “master-slave“ ⟶ “coordinator-worker”
    “orphaned object” ⟶ “unreferenced object”
    “blacklist/whitelist” ⟶ “blocklist/allowlist”
  - **Marginalization of under-represented groups:** e.g., “The Gender attribute is either Male or Female."
  - **Lack of accessibility:** e.g., color alone to convey info in a plot ⟶ use patterns, symbols, textures, etc.
  - **Stereotyping:** e.g., feminine names or presentations for personal secretary role

Formatting

- **Goal: Emphasize Quality of Contents with Quality of Visual Presentation**
  - A carelessly formatted paper (layout, figures, fonts, underlining) creates a bad first impression
  - Recap: **skimming** and **anchoring**

“The paper’s approach is probably equally sloppy”

- **Figures**
  - Use same font and font size as the main text / code in main paper
  - Avoid text overlap, too aggressive **colors**
- **So-called Orphans and Widows**
  - Imprecise definition
  - Avoid few words per line, single line at next page



Strength Reduction: Note that cumsumprod($\mathbf{X}$) uses cumsumprod($\mathbf{B}$)$_{n1}$—i.e., the last block entry—as part of f_{agg}. Similarly, for cumsum($\mathbf{X}$), we could use cumsum($\mathbf{B}$)$_{n:}$. However, this simplifies to colSums($\mathbf{B}$), which avoids materializing the cumsum output block.

Looks ugly and wastes lots of space

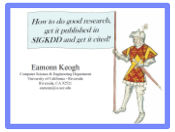
- **Text Running over Column Margin** (rephrase until it fits)
- **Highlighting**
  - `\emph{…}` (emphasize) over underlining or bold
  - `\texttt{…}` or `\verb+…+` for inline code

Page Limits

[Credit: https://twitter.com/fadeladib/status/1322646406088347649]

- **Most Conferences/Journals**
  - Given predefined template, changes not permitted
  - SIGMOD/PVDLB: **12 pages + unlimited references**
  - ICDE: 12 pages incl. references
- **Avoid Cheating**
  - Don’t change the template, fonts, or margins (at least not too excessively)
  - Condensing more text into the paper will make it harder to read
- **Carefully Trim Down Draft**
  - Write unlimited paper, then select, and revise
  - Write and revise section by section as you write
- **Never Excuse Missing Content by “lack of space”**

[Eamonn Keogh: How to do good research, get it published in SIGKDD and get it cited!, **KDD 2009**]



How to do good research, get it published in SIGKDD and get it cited!

Eamonn Keogh

Computer Science & Engineering Department

University of California - Riverside

Riverside, CA 92521

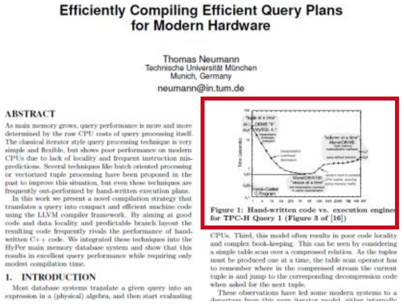
eamonn@cs.ucr.edu

```
“Due to the lack of space, we omit
[essential details] / [essential experiments]”
```

Plagiarism

- **Self-Plagiarism (Bad Idea)**
  - Avoid reusing motivation, introduction, figures, and examples
  - Start writing every thesis / paper from scratch (unless thesis summaries/extends previous papers)

- **Figure Plagiarism (Bad Idea)**
  - Never copy figures from other papers, web, etc
  - Create all figures yourself, even for surveys (can be based on ideas of existing papers)
  - Exceptions do exist w/ explicit references



Efficiently Compiling Efficient Query Plans for Modern Hardware

Thomas Neumann
Technische Universität München
Munich, Germany
neumann@in.tum.de

ABSTRACT

As main memory grows, query performance is more and more determined by the raw CPU costs of query processing itself. The classical iterator style query processing technique is very simple and flexible, but shows poor performance on modern CPUs due to lack of locality and frequent instruction mispredictions. Several techniques like batch oriented processing or vectorized tuple processing have been proposed in the past to improve this situation, but even these techniques are frequently out-performed by hand-written execution plans.

In this work we present a novel compilation strategy that translates a query into compact and efficient machine code using the LLVM compiler framework. By aiming at good code and data locality and predictable branch layout the resulting code frequently rivals the performance of hand-written C++ code. We integrated these techniques into the HyPer main memory database system and show that this results in excellent query performance while requiring only modest compilation time.

1. INTRODUCTION

Most database systems translate a given query into an expression in a (physical) algebra, and then start evaluating

Figure 1: Hand-written code vs. execution engines for TPC-H Query 1 (Figure 3 of [16])

CPUs. Third, this model often results in poor code locality and complex book-keeping. This can be seen by considering a simple table scan over a compressed relation. As the tuples must be produced one at a time, the table scan operator has to remember where in the compressed stream the current tuple is and jump to the corresponding decompression code when asked for the next tuple.

These observations have led some modern systems to a departure from this pure iterator model, either internally

- **Plagiarisms (Really Bad Idea)**
  - Never copy figures or text from other people's work and claim its yours (slight rewording does not change that)
  - For archival scientific publications, there is a high chance it will be detected

Plagiarism: Duplicate Submission

[**Credit:** https://2025.sigmod.org/calls_papers_sigmod_research.shtml]

- **Example SIGMOD 2025**
  - Submitted papers **cannot be under review for any other publishing forum** (conferences, workshops, journals)
  - Authors must **await the response** (only re-submit elsewhere if the paper is rejected/withdrawn)
  - Every research paper must present **substantial novel research** not described in any **prior publication**
    a) A **paper of 5+ pages** presented/accepted at a **refereed conference/workshop**
    b) An **article** published/accepted in a **refereed journal**
  - Requirement to **cite prior publications** in case of overlap
  - **Violation of this policy**
    - Immediate rejection of the submission
    - Notification of the chairs/committees of SIGMOD and the other involved forums

Plagiarism: Automatic CS Paper Generation

- **SCIgen**
  - Generates random CS research papers, including graphs and figures
  - Uses hand-written context-free grammar
  - Test for low-submission standards of conferences
  - **Meaningless mix of sentences and technical terms**

[**Credit:** https://pdos.csail.mit.edu/archive/scigen]

- **Generative AI** (such as ChatGPT)
  - **ACM Policy on Authorship (applies to, e.g., SIGMOD)**
    - **Generative AI tools** may not be authors of publications
    - Using generative AI to create content is **permitted**
    - **But: must be fully disclosed in the work**
    - **Basic word processing systems** (e.g., spelling/grammar corrections) generally allowed, no requirement for disclosure
    - Policy updates expected due to blurring boundaries between generative AI and basic word processing systems

[**Credit:** https://www.acm.org/publications/policies/new-acm-policy-on-authorship]

LDE seminar and project:
Use of generative AI
not allowed

Summary and Q&A

- **Scientific Reading**
- **Scientific Writing**

- **Remaining Questions?**
- **Seminar/Project Topic Selection by Oct 31, 23:59**

- **Final Introductory Lecture**
  - 03 **Experiments, Reproducibility, and Giving Presentations** [Oct 28, MAR 0.015]
    Also recommendable for participants taking only the project