Optimizing Coordinate Descent over Normalized Data in Column Store

Zhiwei Fan
University of Wisconsin-Madison
zhiwei@cs.wisc.edu

Arun Kumar
University of Wisconsin-Madison
arun@cs.wisc.edu

Jeffrey Naughton
University of Wisconsin-Madison
naughton@cs.wisc.edu

Steve Wright
University of Wisconsin-Madison
swright@cs.wisc.edu

ABSTRACT

1. INTRODUCTION

ML Technique	F_e(a, b) (For Loss)	G(a, b) (For Gradient)
Logistic regression (**LR**)	$\log(1 + e^{-ab})$	$\frac{-a}{1 + e^{ab}}$
Least-Squares Regression (**LSR**), Lasso, and Ridge	$(a - b)^2$	$2(b - a)$
Linear Support Vector Machine (**LSVM**)	$\max\{0, 1\text{-}ab\}$	$-a\delta_{ab<1}$

Table 1: GLMs and their functions.

2. BACKGROUND AND PRELIMINARIES

We provide a brief introduction to GLMs, CD and MonetDB. For a deeper understanding and interests, we refer the readers to [**?**][**?**][**?**][**?**], we use the same notation used in [**?**].

2.1 Generalized Linear Models (GLMs)

Consider a matrix $\mathbf{X}$ with dimension $n \times d$ representing a dataset including n samples and each sample has d features. Then the i_{th} row of matrix $\mathbf{X}$(namely $\mathbf{X}_{(i,)}$) represents the i_{th} sample in the dataset with its corresponding d features; the j_{th} column of matrix $\mathbf{X}$(namely $\mathbf{X}_{(,j)}$) represents the j_{th} feature of all examples in the dataset. Y is a n-dimensional vector representing numerical targets for all samples in the datasets. For instance, Y_i is a numerical target for $\mathbf{X}_{(i,)}$. $Y_i \in \mathbb{R}$ considering regression (continuous target values). For discrete target values, however, Y_i represents the numerical value representing the defined class, such as classification in binary class, $Y_i \in \{-1, 1\}$. GLMs make the assumption that the data in dataset could be distributed in different classes (discrete target values for classification) or assigned approximated values (continuous target values for regression) by a hyperplane. Such hyperplane is called "model", and the ultimate goal is to use the given dataset $\mathbf{X}$ to compute the model $\mathbf{W} \in \mathbb{R}^d$. To evaluate how accurate the model is, a *linearly-separable* objective function is given to compute the *loss* of the model $\mathbf{W} \in \mathbb{R}^d$ on the data: $F(\mathbf{W}) = \sum_{i=1}^{n} F_e(Y_i, \mathbf{W}^T\mathbf{X}_{(i,)}^T)$. Applying proper ML algorithm to find the optimal solution for model $\mathbf{W} \in \mathbb{R}^d$ to minimize the loss function is the ultimate goal. The corresponding mathematical statement is: find a vector $\mathbf{W}^* \in \mathbb{R}^d$ s.t., $\mathbf{W}^* = argmin_{\mathbf{W}^*} F(\mathbf{W})$.

2.2 Coordinate Descent(CD)

Coordinate Descent(CD) is a column-friendly algorithm (fetch one column each time for a single coordinate update), thus we consider CD in column stores taking advantage of the fact that the data is stored in columns in the column-oriented DBMS (specifically we use MonetDB as paradigm in our work).

Here, we first just present the most basic version of CD for simplicity. The algorithm just updates one coordinate each time regarding other coordinates as fixed in each iteration. However, as noted in [**?**], most applications in real life use *block coordinate descent* (BCD), in which several coordinates regarded as one block are updated synchronously each time in each iteration regarding other "blocks" are fixed instead of updating just a single coordinate each time. We will briefly discuss BCD in the extensional work part and it shows that our new-designed techniques will also have better performance improvement (in terms of speed-up) in BCD compared to the *single coordinate descent* method.

Algorithm 1 Coordinate Descent (CD)

Inputs: $\{\mathbf{X}, Y\}$ (Data)

$k \leftarrow 0$, $r_{prev} \leftarrow$ null, $r_{curr} \leftarrow$ null, $H \leftarrow \mathbf{0}$, $\mathbf{W} \leftarrow \mathbf{0}$
while (Stop (k, r_{prev}, r_{curr}) = False) **do**
  $r_{prev} \leftarrow r_{curr}$
  for j = 1 to d **do** ▷ 1 pass over data
    $\nabla F_j^k(\mathbf{W}) \leftarrow \sum_{i=1}^{n} G(Y_i, H_i) X_{(i,j)}$
    $\mathbf{W}_j^{(k)} = \mathbf{W}_j^{(k-1)} - \alpha \nabla F_j^k(\mathbf{W})$
    $H \leftarrow H + (\mathbf{W}_j^k - \mathbf{W}_j^{(k-1)}) \times \mathbf{X}_{(,j)}$
  end for
  $r_{curr} \leftarrow F_k$
  $k \leftarrow k + 1$
end while

CD is a simple algorithm to solve GLMs using iterative numerical optimization. CD initializes the model $\mathbf{W}$ to some $\mathbf{W}^{(0)}$, considering a single coordinate every time in each iteration, the algorithm assumes all other coordinates are fixed except for $\mathbf{W}_j$, the j_{th} entry in the model to be updated. Compute the partial gradient $\nabla F_j(\mathbf{W})$ on the given dataset on coordinate W_j (corresponding to j_{th} feature) , where $\nabla F_j(\mathbf{W}) = \sum_{i=1}^{n} G(Y_i, \mathbf{W}^T\mathbf{X}_{(i,)}^T)\mathbf{X}_{(i,j)}$. And then updates the j_{th} entry in the model as $W_j \leftarrow W_j - \alpha \nabla F_j(\mathbf{W})$, where $\alpha > 0$ is the *learning rate* (stepsize) parameter. Once after finishing the update of current coordinate, the algorithm goes into the updating process for the next coordinate in the model $\mathbf{W}$. The coordinate descent that updates one single coordinate each time is also called *stochastic coordinate descent* (SCD). SCD is outlined in Algorithm 1.

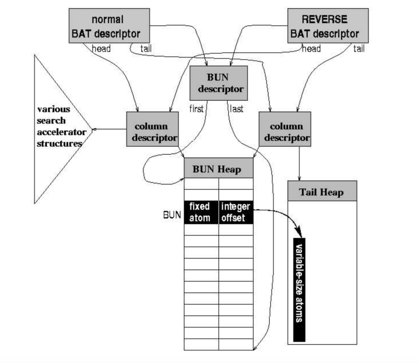
SCD updates the model repeatedly, i.e., over many *iterations* (or *epoches*), each of which requires at least one pass of data. The loss value typically decreases over iterations. The algorithm stops typically with some pre-defined conditions (i.e., specific number of iterations, the decrease of loss value across iterations). The learning rate parameter (α) is typically selected using a line search method that potentially computes the loss many times. Conventionally, *re-ordering* (shuffling) of data in the dataset is done in every iteration to reduce the *influence* that might be caused by data ordering on learning process. However, many practical experiments show that one shuffling is good enough to mitigate the influence of data ordering, thus it is not necessary to shuffle the data in every iteration. We apply this idea in our implementations.
Here, we present the general version of CD algorithm which can be applied for all GLMs listed on Table 1. H is $n \times 1$ *residual vector*, which stores the $\mathbf{W}^T\mathbf{X}_{(i,)}^T$ for every $X_{(i,)}$ with current $\mathbf{W}$. In some cases, it is possible to find the optimal $\mathbf{W}_j$ mathematical expression. For reader's interests, we refer [**?**], which presents CD applied in LSR.

2.3 MonetDB

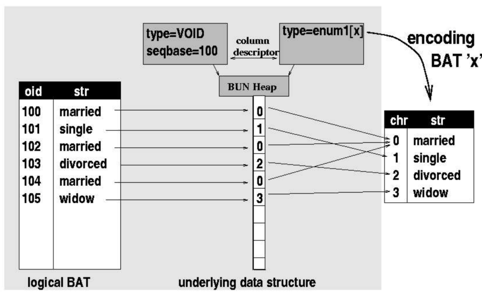
We use MonetDB, an open-source column store, as the paradigmatic column-oriented DBMS in this paper. In order to give readers a better understanding of our work, we provide a brief introduction to MonetDB. Readers can also refer to [**?**].

2.3.1 Data Structure

The data structure used in MonetDB is called *binary association table* (BAT). Every column in the table is stored in



(a) BAT Data Structure



(b) The logical mapping of BAT

Figure 1: MonetDB Data Structure

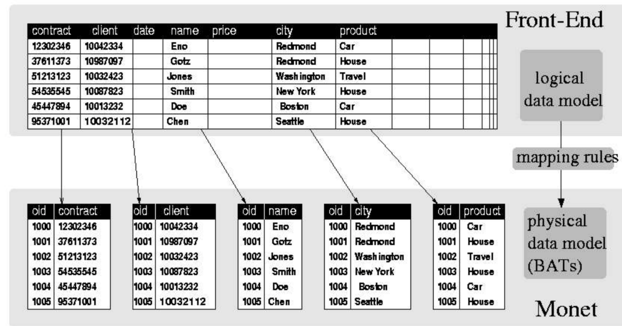


Figure 2: Examples of storage of relations in MonetDB

a BAT. The single unit in BAT is called *binary unit* (BUN). The two columns of BAT is called *BUN Heap* and the first column in BUN Heap is called *head*, which stores *objective identifier* (OID, used to refer the logical location of each tuple in the table); the second column in BUN heap is called *tail*, which stores the corresponding attribute (or feature). All BATs for the same table that store all different features have the same OID in the head columns in BUN Heaps. Figure 2 and Figure 3 shows the graphic depiction of BAT.

2.3.2 *Storing Relations in MonetDB*

In Figure 3, it shows how every attribute in the "contract" table is stored in the BAT data structure in MonetDB. We can observe that different attributes on the same row in the "contract" table have the same oid in different BATs, and this observation conforms to the description of BAT data structure given before. It should not be hard to conjecture that MonetDB uses oid to locate and fetch the data needed: once the oid for certain tuple in database is known, all attributes in this tuple could be fetched in corresponding BATs using the oid as reference. What should be noticed here is that for different tables, the oid generated by the database system will be different (length, sequence). In other words, oid is unique for different tables and **No** same oid can be or should be found in BATs for two different tables. Otherwise there will be conflicts and problems when fetching data in different tables but share the same oid.

2.3.3 *Examples of relational mapping*

Figure 4 gives a general sense of relational mapping between two tables: let's say we want to join the "Order" table and "Item" table on order "id". then "id" would be the primary key of "Order" table and "order" in "Item" table is the foreign key refers to the order "id" in "Order" table. Looking at the order_id and item_order BATs in Figure 5(a), since the oids for all attributes are the same in the same tuple of the same table, once the oid-oid mapping of "Order" and "Item" is got, the join of "Order" table and "item" table can be completed by copying all the attributes to the corresponding position referred by oid following oid-oid mapping. The oid-oid mapping is shown in Figure 5(b). To get the oid-oid mapping, simple hash join is applied since only two BATs (primary key and foreign key) are needed (thus it is reasonable to assume that they can fit well in memory). We call this oid-oid mapping as "key-foreign key mapping reference"(KKMR) and will use this notation in later sections.

2.4 Type Changes and *Special* Characters

We have already seen several typeface changes in this sample. You can indicate italicized words or phrases in your text with the command `\textit`; emboldening with the command `\textbf` and typewriter-style (for instance, for computer code) with `\texttt`. But remember, you do not have to indicate typestyle changes when such changes are part of the *structural* elements of your article; for instance, the heading of this subsection will be in a sans serif[1] typeface, but that is handled by the document class file. Take care with the use of[2] the curly braces in typeface changes; they mark the beginning and end of the text that is to be in the different typeface.

You can use whatever symbols, accented characters, or non-English characters you need anywhere in your document; you can find a complete list of what is available in the *LaTeX User's Guide*[5].

2.5 Math Equations

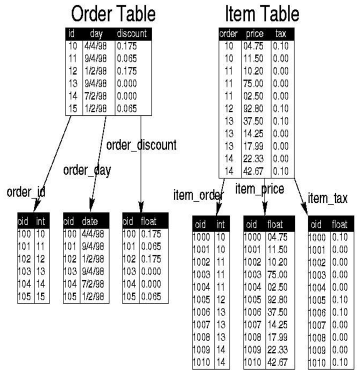
You may want to display math equations in three distinct styles: inline, numbered or non-numbered display. Each of the three are discussed in the next sections.

2.5.1 *Inline (In-text) Equations*

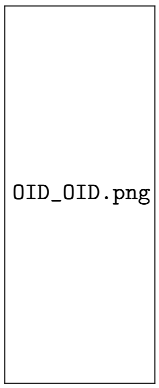
A formula that appears in the running text is called an inline or in-text formula. It is produced by the **math** environment, which can be invoked with the usual `\begin. . .\end` construction or with the short form `$. . .$`.

[1]A third footnote, here. Let's make this a rather short one to see how it looks.

[2]A fourth, and last, footnote.



(a) BAT mapping of "Order" and "Item"



(b) oid-oid mapping

Figure 3: Relational mapping of "Order" table and "Item" Table

You can use any of the symbols and structures, from α to ω, available in LaTeX[5]; this section will simply show a few examples of in-text equations in context. Notice how this equation: $\lim_{n\rightarrow \infty}x = 0$, set here in in-line math style, looks slightly different when set in display style. (See next section).

2.5.2 *Display Equations*

A numbered display equation – one set off by vertical space from the text and centered horizontally – is produced by the **equation** environment. An unnumbered display equation is produced by the **displaymath** environment.

Again, in either environment, you can use any of the symbols and structures available in LaTeX; this section will just give a couple of examples of display equations in context. First, consider the equation, shown as an inline equation above:

$$\lim_{n\rightarrow \infty}x = 0 \tag{1}$$

Notice how it is formatted somewhat differently in the **displaymath** environment. Now, we'll enter an unnumbered equation:

$$\sum_{i=0}^{\infty} x + 1$$

and follow it with another numbered equation:

$$\sum_{i=0}^{\infty}x_i=\int_{0}^{\pi+2} f \tag{2}$$

Table 2: Frequency of Special Characters

Non-English or Math	Frequency	Comments
Ø	1 in 1,000	For Swedish names
π	1 in 5	Common in math
\$	4 in 5	Used in business
Ψ^2_1	1 in 40,000	Unexplained usage

just to demonstrate LaTeX's able handling of numbering.

2.6 Citations

Citations to articles [1, 2, 3, 4], conference proceedings [3] or books [5, 6] listed in the Bibliography section of your article will occur throughout the text of your article. You should use BibTeX to automatically produce this bibliography; you simply need to insert one of several citation commands with a key of the item cited in the proper location in the `.tex` file [5]. The key is a short reference you invent to uniquely identify each work; in this sample document, the key is the first author's surname and a word from the title. This identifying key is included with each item in the `.bib` file for your article.

The details of the construction of the `.bib` file are beyond the scope of this sample document, but more information can be found in the *Author's Guide*, and exhaustive details in the *LaTeX User's Guide*[5].

This article shows only the plainest form of the citation command, using `\cite`. This is what is stipulated in the SIGS style specifications. No other citation format is endorsed.

2.7 Tables

Because tables cannot be split across pages, the best placement for them is typically the top of the page nearest their initial cite. To ensure this proper "floating" placement of tables, use the environment **table** to enclose the table's contents and the table caption. The contents of the table itself must go in the **tabular** environment, to be aligned properly in rows and columns, with the desired horizontal and vertical rules. Again, detailed instructions on **tabular** material is found in the *LaTeX User's Guide*.

Immediately following this sentence is the point at which Table 1 is included in the input file; compare the placement of the table here with the table in the printed dvi output of this document.

To set a wider table, which takes up the whole width of the page's live area, use the environment **table*** to enclose the table's contents and the table caption. As with a single-column table, this wide table will "float" to a location deemed more desirable. Immediately following this sentence is the point at which Table 2 is included in the input file; again, it is instructive to compare the placement of the table here with the table in the printed dvi output of this document.

2.8 Figures

Like tables, figures cannot be split across pages; the best placement for them is typically the top or the bottom of the page nearest their initial cite. To ensure this proper "floating" placement of figures, use the environment **figure** to enclose the figure and its caption.

Table 3: Some Typical Commands

Command	A Number	Comments
`\alignauthor`	100	Author alignment
`\numberofauthors`	200	Author enumeration
`\table`	300	For tables
`\table*`	400	For wider tables

Figure 4: A sample black and white graphic (.pdf format).

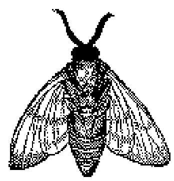


Figure 5: A sample black and white graphic (.pdf format) that has been resized with the `includegraphics` command.

This sample document contains examples of **.pdf** files to be displayable with LaTeX(See Figures 4 and 5). More details on each of these is found in the *Author's Guide*.

As was the case with tables, you may want a figure that spans two columns. To do this, and still to ensure proper "floating" placement of tables, use the environment **figure*** to enclose the figure and its caption (See Figure 6). And don't forget to end the environment with figure*, not figure!

Note that only **.pdf** files were used; if you want to include **.ps** or **.eps** formats, you can use the `\epsfig` or `\psfig` commands as appropriate for the different file types.

2.9 Theorem-like Constructs

Other common constructs that may occur in your article are the forms for logical constructs like theorems, axioms, corollaries and proofs. There are two forms, one produced by the command `\newtheorem` and the other by the command `\newdef`; perhaps the clearest and easiest way to distinguish them is to compare the two in the output of this sample document:

This uses the **theorem** environment, created by the `\newtheorem` command:

THEOREM 1. *Let f be continuous on $[a,b]$. If G is an antiderivative for f on $[a,b]$, then*

$$\int_a^b f(t)dt = G(b) - G(a).$$

The other uses the **definition** environment, created by the `\newdef` command:

Definition 1. If z is irrational, then by e^z we mean the unique number which has logarithm z:

$$\log e^z = z$$

Two lists of constructs that use one of these forms is given in the *Author's Guidelines*.

There is one other similar construct environment, which is already set up for you; i.e. you must *not* use a `\newdef` command to create it: the **proof** environment. Here is a example of its use:

PROOF. Suppose on the contrary there exists a real number L such that

$$\lim_{x\to\infty} \frac{f(x)}{g(x)} = L.$$

Then

$$l = \lim_{x\to c} f(x) = \lim_{x\to c} \left[gx \cdot \frac{f(x)}{g(x)} \right]$$
$$= \lim_{x\to c} g(x) \cdot \lim_{x\to c} \frac{f(x)}{g(x)} = 0 \cdot L = 0,$$

which contradicts our assumption that $l \neq 0$. $\square$

Complete rules about using these environments and using the two different creation commands are in the *Author's Guide*; please consult it for more detailed instructions. If you need to use another construct, not listed therein, which you want to have the same formatting as the Theorem or the Definition[6] shown above, use the `\newtheorem` or the `\newdef` command, respectively, to create it.

A *Caveat* for the TeX Expert

Because you have just been given permission to use the `\newdef` command to create a new form, you might think you can use TeX's `\def` to create a new command: *Please refrain from doing this!* Remember that your LaTeX source code is primarily intended to create camera-ready copy, but may be converted to other forms – e.g. HTML. If you inadvertently omit some or all of the `\def`s recompilation will be, to say the least, problematic.

3. CONCLUSIONS

This paragraph will end the body of this sample document. Remember that you might still have Acknowledgments or Appendices; brief samples of these follow. There is still the Bibliography to deal with; and we will make a disclaimer about that here: with the exception of the reference to the LaTeX book, the citations in this paper are to articles which have nothing to do with the present subject and are used as examples only.

4. ACKNOWLEDGMENTS

This section is optional; it is a location for you to acknowledge grants, funding, editing assistance and what have you. In the present case, for example, the authors would like to thank Gerald Murray of ACM for his help in codifying this *Author's Guide* and the **.cls** and **.tex** files that it describes.

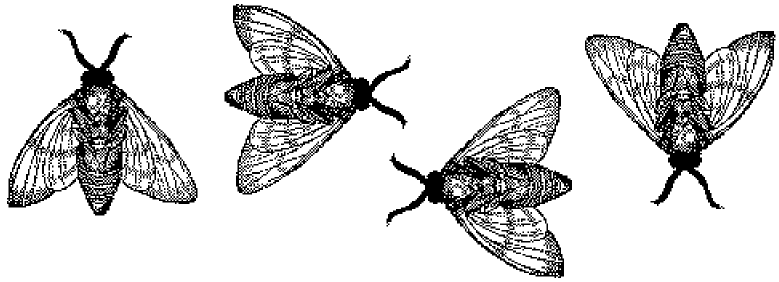


Figure 6: A sample black and white graphic (.pdf format) that needs to span two columns of text.

References

M. Bowman, S. K. Debray, and L. L. Peterson. Reasoning about naming systems. *ACM Trans. Program. Lang. Syst.*, 15(5):795–825, November 1993.

J. Braams. Babel, a multilingual style-option system for use with latex's standard document styles. *TUGboat*, 12(2):291–301, June 1991.

M. Clark. Post congress tristesse. In *TeX90 Conference Proceedings*, pages 84–89. TeX Users Group, March 1991.

M. Herlihy. A methodology for implementing highly concurrent data objects. *ACM Trans. Program. Lang. Syst.*, 15(5):745–770, November 1993.

L. Lamport. *LaTeX User's Guide and Document Reference Manual*. Addison-Wesley Publishing Company, Reading, Massachusetts, 1986.

S. Salas and E. Hille. *Calculus: One and Several Variable*. John Wiley and Sons, New York, 1978.

4.1 References

Generated by bibtex from your .bib file. Run latex, then bibtex, then latex twice (to resolve references).

APPENDIX

You can use an appendix for optional proofs or details of your evaluation which are not absolutely necessary to the core understanding of your paper.

A. FINAL THOUGHTS ON GOOD LAYOUT

Please use readable font sizes in the figures and graphs. Avoid tempering with the correct border values, and the spacing (and format) of both text and captions of the PVLDB format (e.g. captions are bold).

At the end, please check for an overall pleasant layout, e.g. by ensuring a readable and logical positioning of any floating figures and tables. Please also check for any line overflows, which are only allowed in extraordinary circumstances (such as wide formulas or URLs where a line wrap would be counterintuitive).

Use the `balance` package together with a `\balance` command at the end of your document to ensure that the last page has balanced (i.e. same length) columns.