

The **MADlib** Analytics Library

or MAD Skills, the SQL

Caleb Welton

Florian Schoppmann

Christopher Ré



MADlib

Scalable Machine Learning for BigData

MAD Skills: New Analysis Practices for Big Data

Jeffrey Cohen
Georgetown

Brian Dolan
Fox Interactive Media

Mark Dunlap
Evergreen Technologies

Joseph M. Hellerstein
U. of Wisconsin

Caleb Welton
Cognifinity

Towards a Unified Architecture for in-RDBMS Analytics

Xixuan Feng
Department of Computer Sciences
University of Wisconsin-Madison
(xfeng, arun, brecht, chrisre)@cs.wisc.edu

Arun Kumar
Benjamin Recht
Christopher Ré

The MADlib Analytics Library or MAD Skills, the SQL

Joseph M. Hellerstein
U. of Wisconsin

Christopher Ré
U. of Wisconsin

Florian Schoppmann
Cognifinity

Daisy Zhe Wang
U. of Wisconsin

Eugene Fratkan
U. of Wisconsin

Aleks Gorajek
Georgetown

Kee Siong Ng
Cognifinity

Xixuan Feng
U. of Wisconsin

Kun Li
U. of Florida

Arun Kumar
U. of Wisconsin

ABSTRACT

As massive data is increasingly affordable, and statisticians to our paper we highlight the Deep (MAD) traditional Enterprise license. We prove experience provide larges, advertising the Greenplum database design in ing style of analy parallel algorithms with a focus on database system's ble algorithm dev interfaces over a v

1. INTRODUCTION

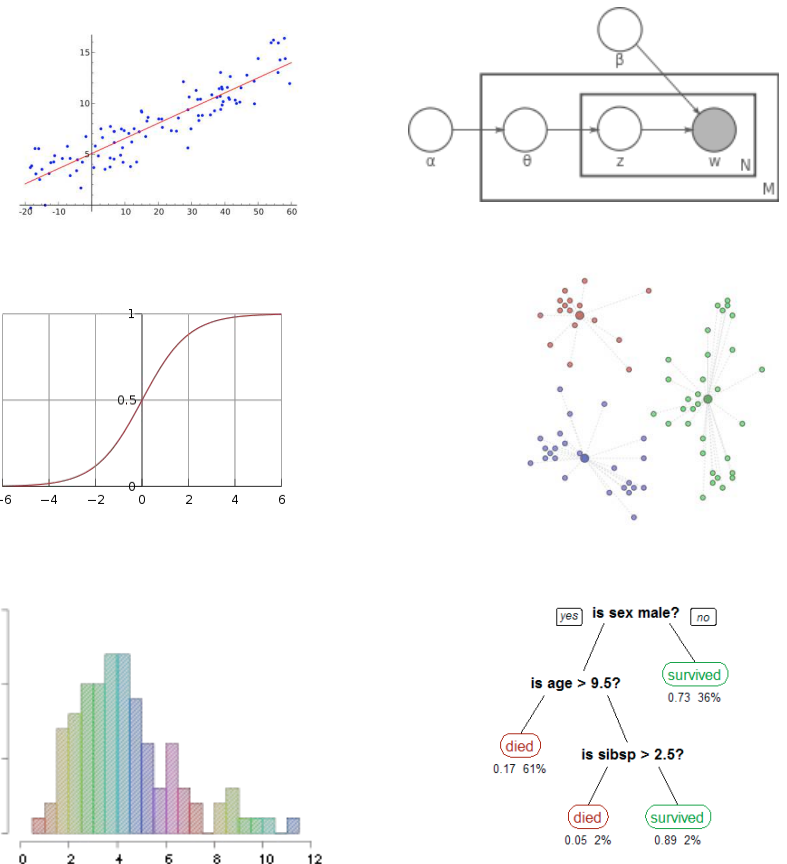
If you are looking in high demand, y a scarce, complex ubiquitous and che Data. And what is ~ Prof. Hal Varian

mad (adj.): an ad (-) - dude, you got a 6-shed, you got a ~ UrbanDictionary

Standard business ter on the notion o that is queried by tools produce rep ries data via back averages) over vit

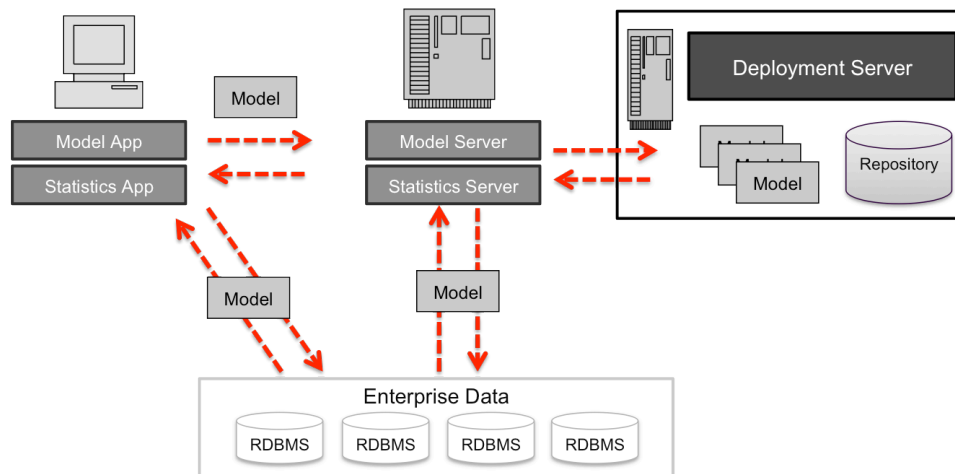
Under revision: This v

Permission to make personal or classmate not make or distribe hear this notice and republish, to post on permission under a SIGMOD '12, May, Copyright 2012, AC

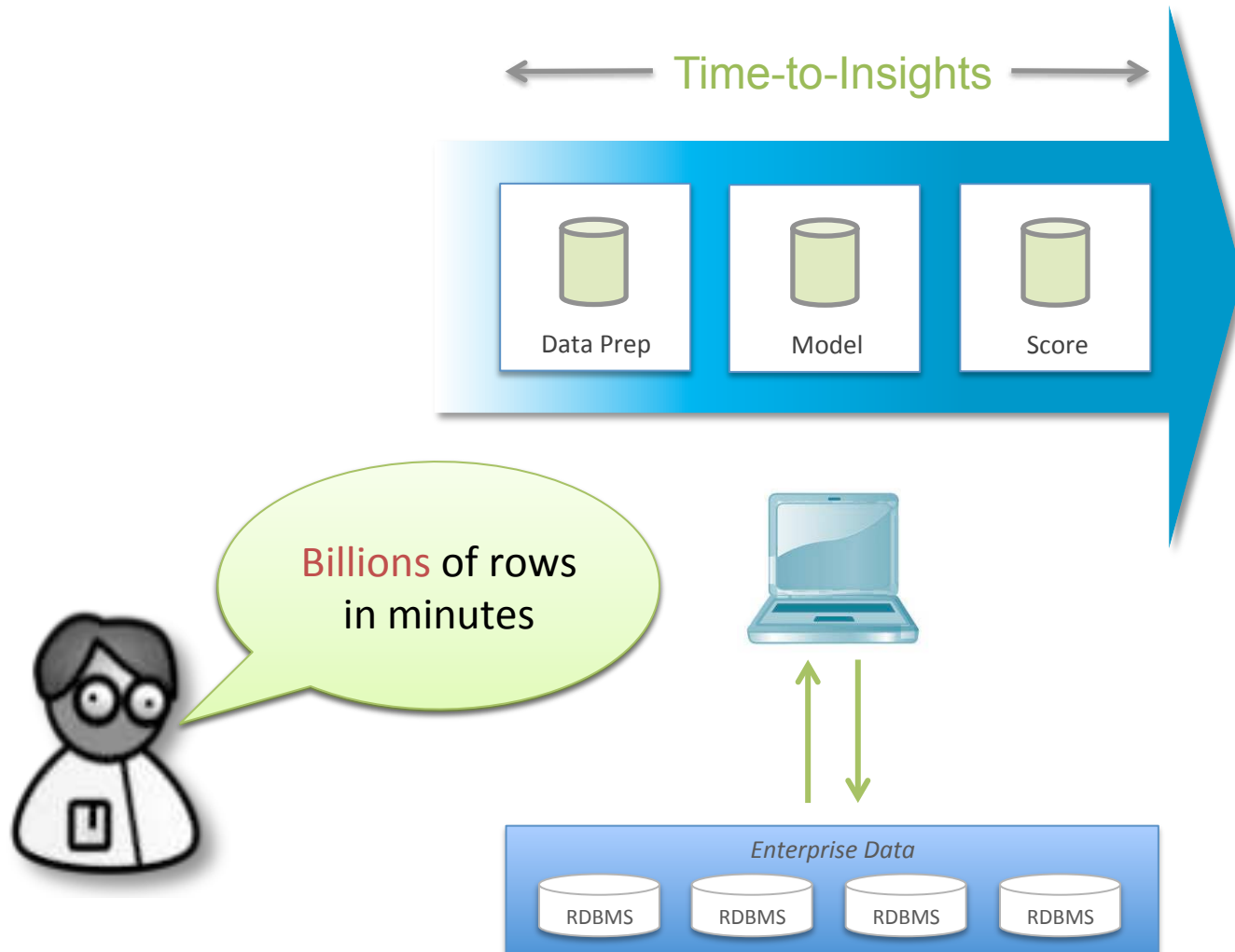


Traditional analytics pipeline

← Time-to-Insights →



The *MAD* approach



MADlib in *Action*

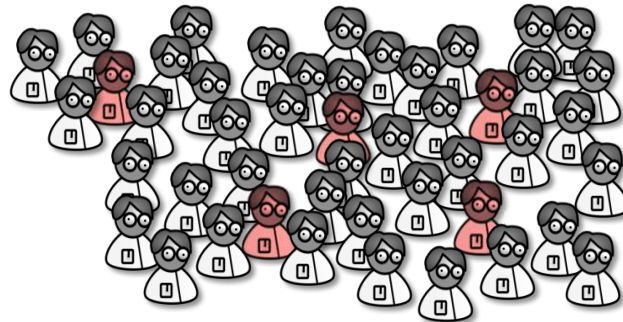
Hospital Admittance Case Study



MADlib in *Action*

Step 1:

- Identify high risk patients



Goal:

- High risk patients will be eligible for early admittance and be administered preemptive antibiotics

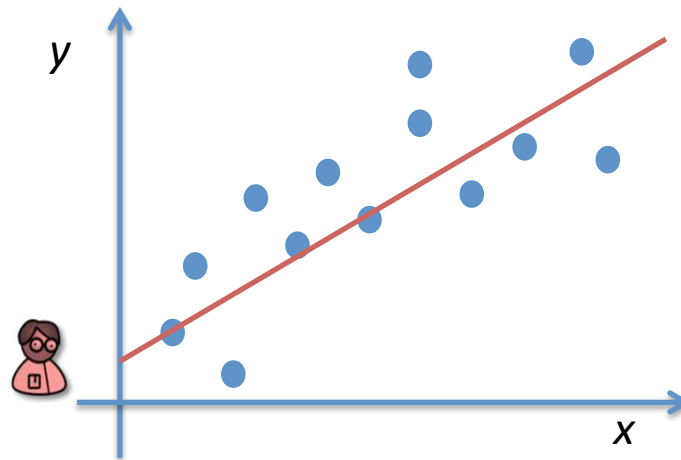


MADlib in *Action*



Step 2:

- Build cost model for treatment



Goal:

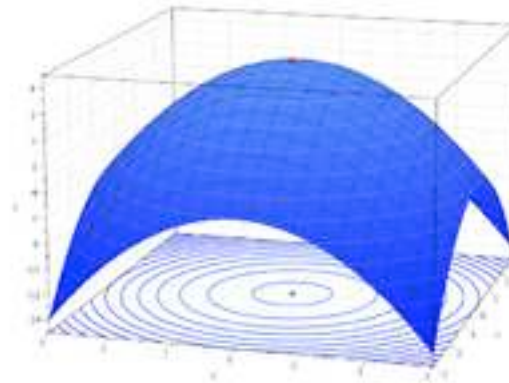
- Predict expected cost of treatment
- With and without early admittance.

MADlib in *Action*



Step 3:

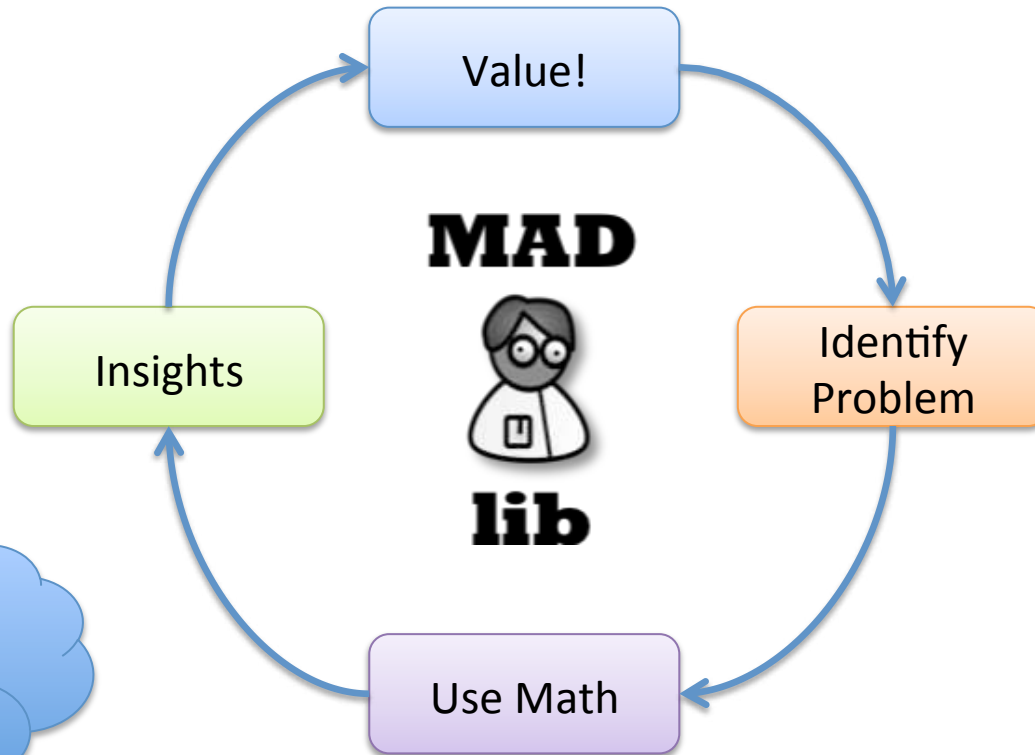
- Optimize early admittance based on risk and cost model



Goal:

- Overall hospital costs will be minimized and patients will receive better care.


MADlib cycle of success



Why didn't I
think of that
before?

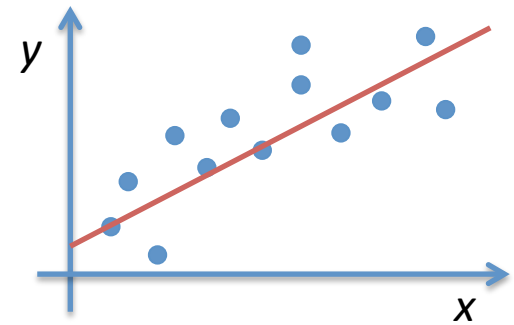


The MADlib Vision

- Academic and industry contributions
- Think of “CRAN for databases” 
 - Repository of open-source ML algorithms
 - This time with data parallelism in mind
- Open-Source Framework



Simple Example: Ordinary Least Squares



```
# SELECT y, x[1] AS x1, x[2] AS x2 FROM data
```

y	x1	x2
10.14	0	0.3
11.93	0.69	0.6
13.57	1.1	0.9
14.17	1.39	1.2
15.25	1.61	1.5
16.15	1.79	1.8

\times

```
# SELECT (linregr(y, x)).* FROM data;
-[ RECORD 1 ]+-----
coef          | {1.7307,2.2428}
r2            | 0.9475
std_err       | {0.3258,0.0533}
t_stats       | {5.3127,42.0640}
p_values      | {6.7681e-07,4.4409e-16}
condition_no  | 169.5093
```

Linear Algebra in the Database

$$\hat{\beta} = (X^T X)^{-1} X^T y$$

Diagram illustrating the matrix operations in the linear regression formula:

The matrix $(X^T X)^{-1}$ is shown as a product of X^T and X .

The vector $X^T y$ is shown as a product of X^T and y .

Below the matrix $X^T X$, the expression $\sum_{i=1}^n \mathbf{x}_i \mathbf{x}_i^T$ is shown, indicating the sum of outer products of the feature vectors.

Below the vector $X^T y$, the expression $\sum_{i=1}^n \mathbf{x}_i y_i$ is shown, indicating the sum of the products of the feature vectors and the target values.

Basic Building Block: User-Defined Aggregates

\mathbf{x}	y
$(1, 0, 3, \dots, 5)$	3
$(-2, 4, 5, \dots, 2)$	2
...	...

(A, \mathbf{b})
...

Aggregation phase 1 on each node:

1. Initialize: $(A, \mathbf{b}) = (0, 0)$

2. **Transition** for all rows:

$$(A, \mathbf{b}) = (A, \mathbf{b}) + \underbrace{(\mathbf{x} \cdot \mathbf{x}^T, \mathbf{x} \cdot y)}_{\text{map}}$$

3. Send (A, \mathbf{b})

reduce

Aggregation phase 2 on master node:

1. **Merge**: $(\bar{A}, \bar{\mathbf{b}}) = (\bar{A}, \bar{\mathbf{b}}) + (A, \mathbf{b})$

2. **Finalize**: $\hat{\beta} = \text{solve}(\bar{A}, \bar{\mathbf{b}}) = \bar{A}^{-1} \cdot \bar{\mathbf{b}}$

Problem solved?

No – not yet.

ML Algorithms Based on SQL?

- Four Representative Challenges
 1. Lack of portable multi-pass iterations
 2. Roots in first-order logic
 3. Lack of language support for linear algebra
 4. Extensible SQL limited to small working sets

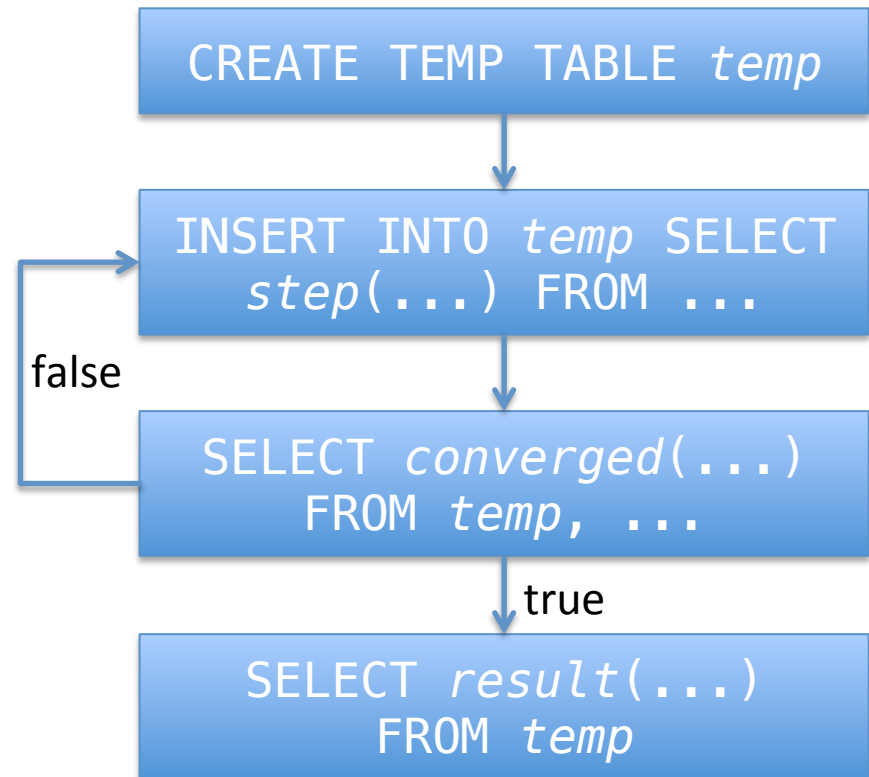
Need:

- Abstraction Layers
- A few compromises for user interface

• A few compromises for user interface

1. Lack of portable multi-pass iterations

- **WITH RECURSIVE** not reliable basis for portability
- User-defined **driver** functions in Python
 - Outer loops not performance-critical
- Compromise:
Different user interface



2. Roots in first-order logic

- Queries need be cognizant of database objects
- Emulate higher-order logic by:
 - dynamic execution of templated SQL
 - abstraction-layer support

```
FunctionHandle dist
    = args[0].getAs<FunctionHandle>();
return dist(x, y);
```

- Example: Distance or kernel functions
- On PostgreSQL, use of type REGPROC

3. Lack of language support for linear algebra

- C++ Abstraction Layer uses Eigen
- (Dense) Vectors and matrices:
DOUBLE PRECISION[]
- Example:



open source

```
AnyType  
solve::run(AnyType& args) {  
    MappedMatrix A = args[0].getAs<MappedMatrix>();  
    MappedColumnVector b = args[1].getAs<MappedColumnVector>();  
  
    MutableMappedColumnVector x = allocateArray<double>(A.cols());  
    x = A.colPivHouseholderQr().solve(b);  
    return x;  
}
```

Performance:

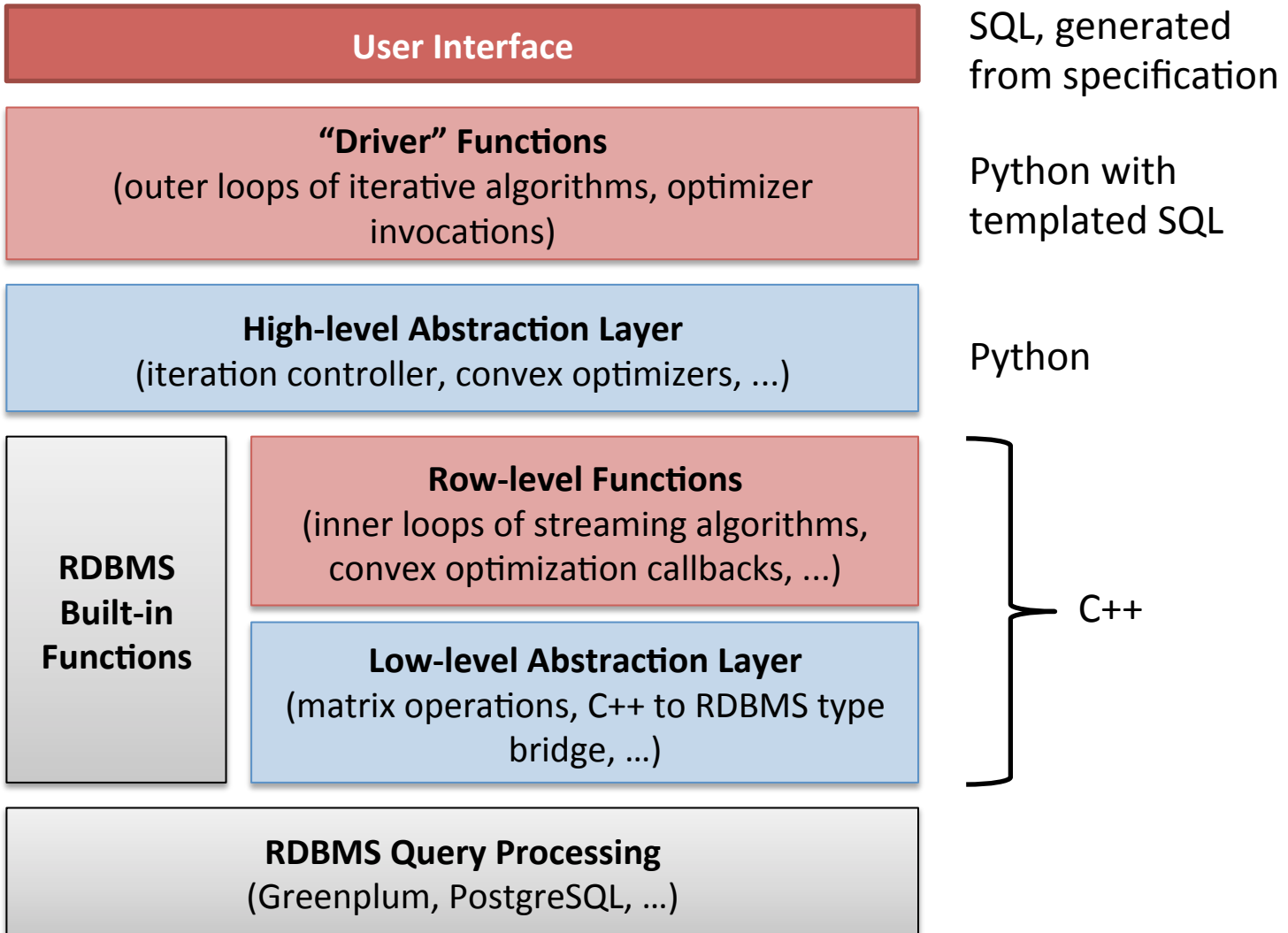
- No unnecessary copying
- No internal type conversion

• No internal type conversion

4. Extensible SQL limited to small working sets

- Tables only portable option for large states
- Access from UDAs slow or impossible
- Example: *k*-means benefits from explicit point-to-centroid assignments
 - Problematic:
`UPDATE points SET centroid_id =
closest(state, coords)`
 - Requires own pass
 - Not allowed in subqueries
 - PostgreSQL legacy

MADlib Architecture



Anatomy of an iterative MADlib module

interState = **Start**(args)

User-defined Function

Repeat

Python Driver Function

In parallel for each segment:

intraState = **Initialize**(interState)

For each row

intraState = **Transit**(intraState, row)

For each intraState:

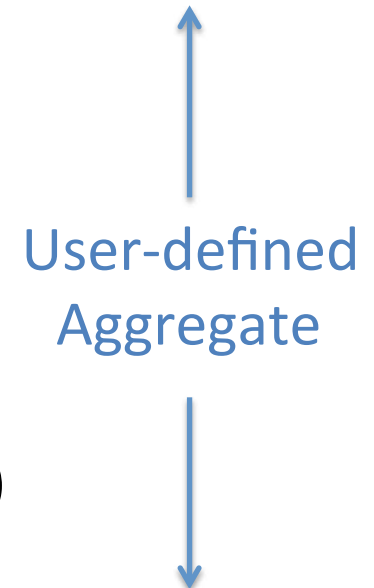
intraState = **Merge**(oldIntraState, intraState)

interState = **Finalize**(intraState)

Until **Converged**(interState)

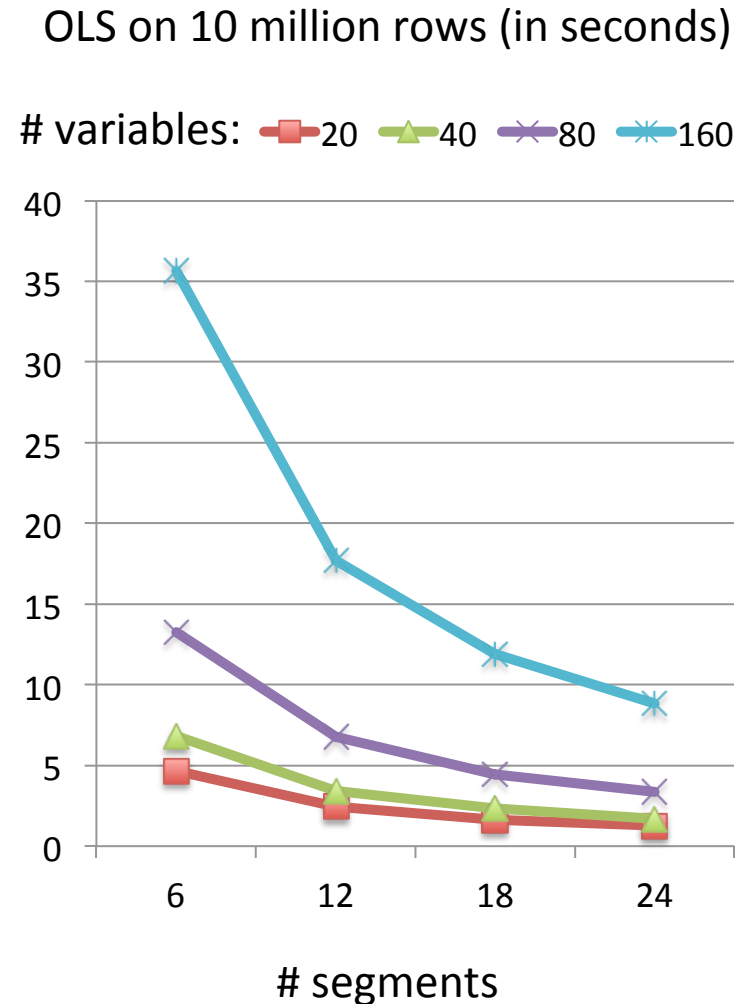
Return **End**(interState)

User-defined Function

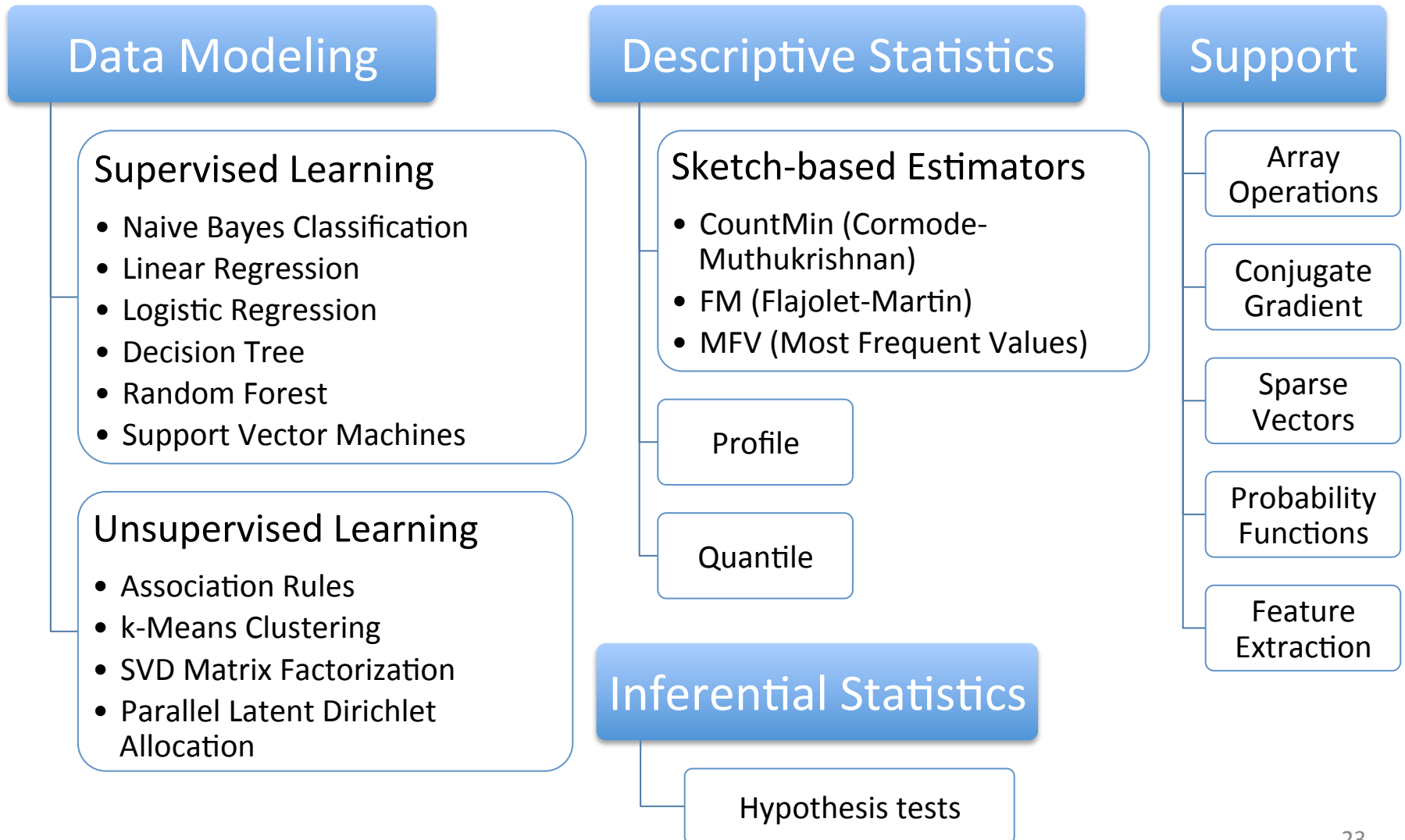


Performance Trends

- Disk I/O is not always the bottleneck
 - Performance tuning is essential
- Overhead for single query very low (fraction of a second)
- Greenplum achieves nearly perfect speedup



Current Modules



My MADlib Experience: A Testimonial.



Christopher Ré, Wisconsin

Refining Ideas and Code

Conversations with GP (and Oracle) lead us to better position our SIGMOD12 paper

Towards a Unified Architecture for in-RDBMS Analytics

Xixuan Feng

Arun Kumar

Benjamin Recht

Christopher Ré

Department of Computer Sciences
University of Wisconsin-Madison
{xfeng, arun, brecht, chrisre}@cs.wisc.edu

ABSTRACT

The increasing use of statistical data analysis in enterprise applications has created an arms race among database vendors to offer ever more sophisticated in-database analytics.

In the late 1990s and early 2000s, this brought a wave of data mining toolkits into the RDBMS. Several major vendors again made an effort toward sophisticated in-database analytics with both open source efforts, e.g., the MADlib platform.

QA from GP help to transition from *paper* to *deployed code*.

MADlib



MADlib is Open Source

DeepDive

Barack Obama

From [Hazy@Wisconsin](#), powered by [Felix](#) (learn more about DeepDive)

The following content is derived from a recent snapshot of the [Wikipedia page](#) and is licensed under [CC-BY-SA](#).
From Wikipedia, the free encyclopedia

"Barack" and "Obama" redirect here. For other uses, see [Barak](#) (disambiguation) and [Obama](#) (disambiguation). For President Obama's father, the Kenyan economist, see [Barack Obama, Sr.](#)

Barack Hussein Obama II (/bəˈrɑːk huːˈseɪn oʊˈbɑːmə/; born August 4, 1961) is the 44th and current President of the United States. He is the first African American to hold the office. Obama previously served as a United States Senator from Illinois, from January 2005 until he resigned following his victory in the 2008 presidential election.

Born in Honolulu, Hawaii, Obama is a graduate of Columbia University and Harvard Law School, where he was the president of the *Harvard Law Review*. He was a community organizer in Chicago before earning his law degree. He worked as a civil rights attorney in Chicago and taught constitutional law at the University of Chicago Law School from 1992 to 2004. He served three terms representing the 13th District in the Illinois Senate from 1997 to 2004.

Following an unsuccessful bid against the Democratic incumbent for a seat in the United States House of Representatives in 2000, Obama ran for the United States Senate in 2004. Several events brought him to national attention during the campaign, including his victory

Barack Obama

	256	Entities
	138	Relations
	12K	Books
	3	Courses
	10	Lectures
	2M	Sentences
	285	Videos (Trend)

*Enhance Wikipedia
with extracted facts
from the Web
(50+TB of data)*

hazy.cs.wisc.edu & www.youtube.com/HazyResearch

Learning & Inference run on (GP or Postgres) + MADLib

Critical: it's free, open, and we can modify it

Critical: it's free, open, and we can modify it

Testimonial Summary

MADlib



Berkeley
University of California



WISCONSIN
UNIVERSITY OF WISCONSIN-MADISON

UF | UNIVERSITY of
FLORIDA



MADlib is open to contributions and open source

Questions?



<http://madlib.net>

Caleb Welton

Caleb.Welton@emc.com

Florian Schoppmann

Florian.Schoppmann@emc.com

Christopher Ré

chrisre@cs.wisc.edu