Enabling R for Big Data with PL/R and PivotalR

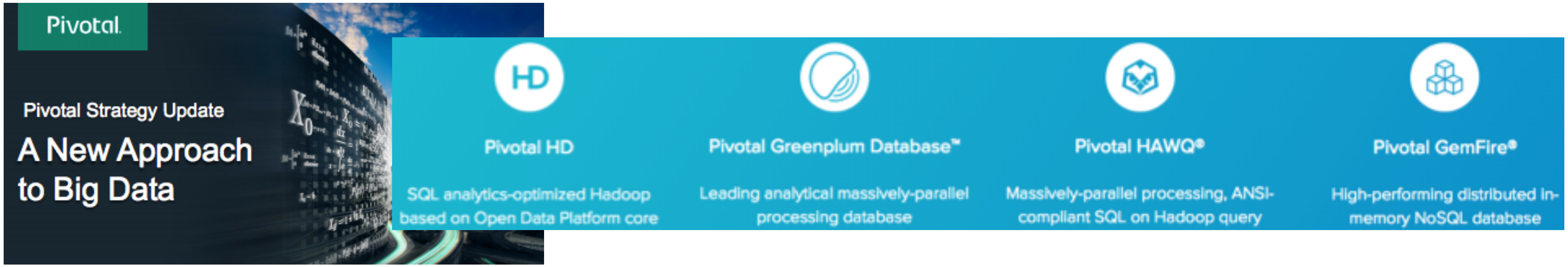
Real World Examples on Hadoop & MPP Databases

Woo J. Jung
Principal Data Scientist
Pivotal Labs

Pivotal™

All In On Open Source

Still can't believe we did this. Truly exciting.



World's First Open Sourced Big Data Portfolio

- Building on success of Cloud Foundry Foundation
- Open sourcing all Pivotal Big Data Suite components
  - **Pivotal GemFire** -premium in-memory NoSQL database
  - **Pivotal HAWQ** - world's leading SQL compliant enterprise SQL on Hadoop
  - **Pivotal Greenplum Database** - advanced enterprise MPP analytic database
- Enterprise differentiated
  - Advanced features
  - Enterprise support
  - Indemnification

Standardize Hadoop Ecosystem

- Open Data Platform
  - Focused on developing common core to enable Hadoop ecosystem
- Rapidly accelerated certifications, ecosystem development, predictability and enterprise applicability

Data Science Toolkit

KEY TOOLS

git

D3

MADlib

RStudio

IP[y]:

KEY LANGUAGES

R

python

SQL

Scala

PLATFORM

Pivotal Cloud Foundry

Pivotal Greenplum Database

Pivotal HD

Pivotal Big Data Suite

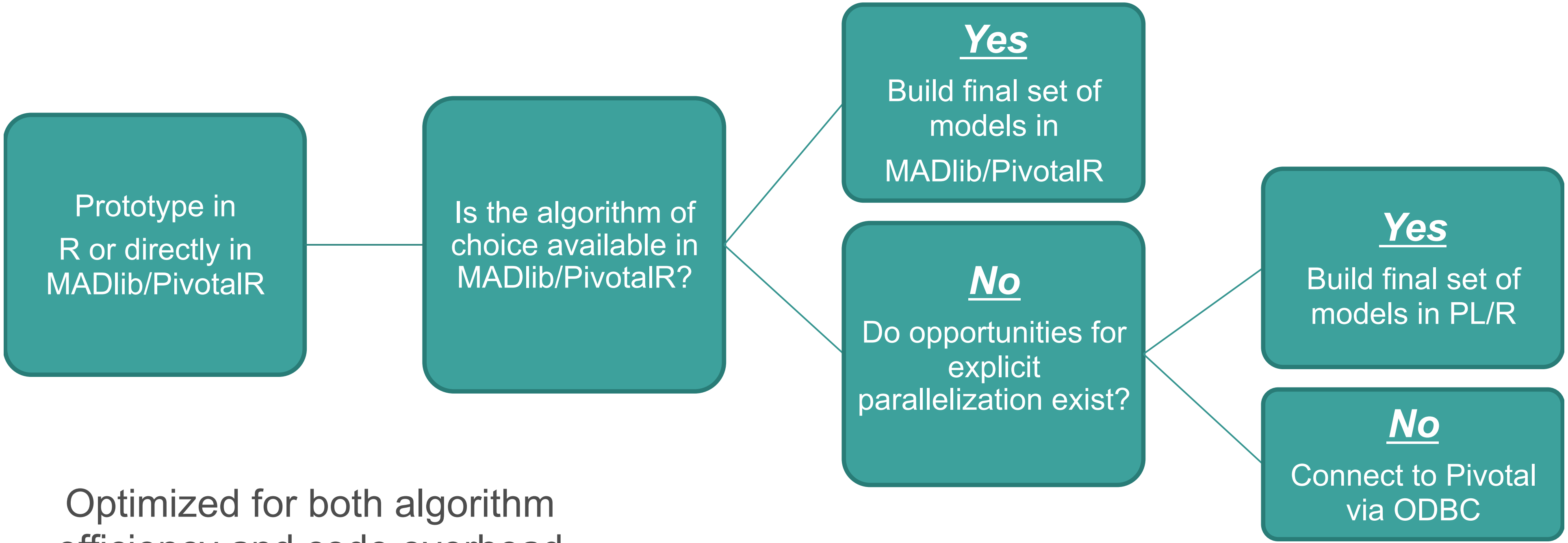
Pivotal HAWQ

Spark

Spring XD

Pivotal

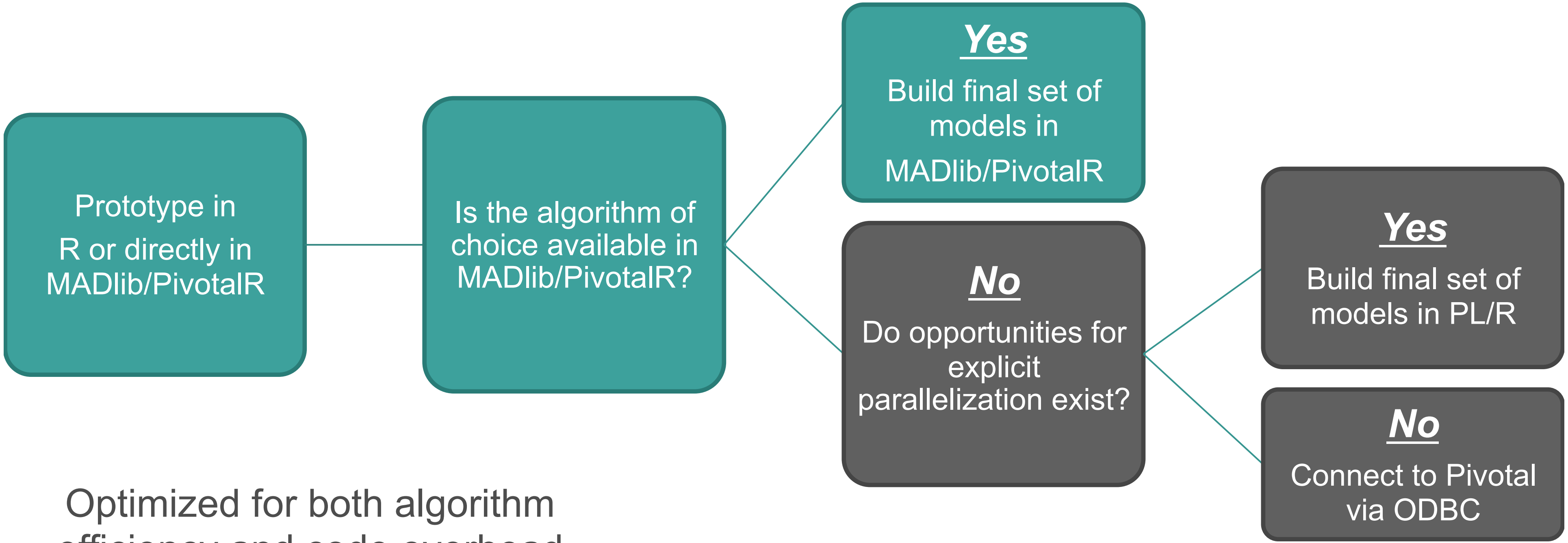
How Pivotal Data Scientists Select Which Tool to Use



Optimized for both algorithm efficiency and code overhead

Pivotal

How Pivotal Data Scientists Select Which Tool to Use



Optimized for both algorithm efficiency and code overhead

Pivotal

MADlib: Toolkit for Advanced Big Data Analytics

madlib.net

- Better Parallelism
  - Algorithms designed to leverage MPP or Hadoop architecture
- Better Scalability
  - Algorithms scale as your data set scales
  - No data movement
- Better Predictive Accuracy
  - Using all data, not a sample, may improve accuracy
- Open Source
  - Available for customization and optimization by user

Pivotal

http://doc.madlib.net/latest/

MADlib 1.7.1
User Documentation

Search

- MADlib
  - Modules
    - Regression Models
      - Linear Regression
      - Logistic Regression
      - Multinomial Regression
      - Ordinal Regression
      - Elastic Net Regularization
      - Cox-Proportional Hazards Regression
      - Robust Variance
      - Clustered Variance
      - Marginal Effects
      - Generalized Linear Models
    - Cross Validation
    - Linear Systems
      - Dense Linear Systems
      - Sparse Linear Systems
    - Matrix Factorization
      - Low-rank Matrix Factorization
      - Singular Value Decomposition
    - Tree Methods
      - Decision Tree
      - Random Forest
    - Association Rules
      - Apriori Algorithm
    - Clustering
      - k-Means Clustering
    - Topic Modelling
      - Latent Dirichlet Allocation
    - Text Analysis
      - Conditional Random Field
    - Descriptive Statistics
      - Summary
      - Pearson's Correlation
    - Inferential Statistics
      - Hypothesis Tests
    - Support Modules
      - Array Operations
      - Sparse Vectors
      - Probability Functions
      - Data Preparation
      - PMML Export
    - Dimensionality Reduction
      - Principal Component Analysis
      - Principal Component Projection
    - Time Series Analysis
      - ARIMA
    - Early Stage Development
      - Naive Bayes Classification
      - Support Vector Machines
      - Cardinality Estimators
      - Conjugate Gradient
      - Random Sampling
      - Linear Algebra Operations

Elastic Net Regularization

Regression Models

This module implements elastic net regularization for linear and logistic regression problems.

Contents

- Training Function
- Optimizer Parameters
- Prediction Functions
- Examples
- Technical Background
- Literature
- Related Topics

Training Function

The training function has the following syntax:

```
elastic_net_train( tbl_source,
                   tbl_result,
                   col_dep_var,
                   col_ind_var,
                   regress_family,
                   alpha,
                   lambda_value,
                   standardize,
                   grouping_col,
                   optimizer,
                   optimizer_params,
                   excluded,
                   max_iter,
                   tolerance
                 )
```

Arguments

tbl_source
TEXT. The name of the table containing the training data.

tbl_result
TEXT. Name of the generated table containing the output model. The output table produced by the **elastic_net_train()** function has the following columns:

regress_family	The regression type: 'gaussian' or 'binomial'.
features	An array of the features (independent variables) passed into the analysis.
features_selected	An array of the features selected by the analysis.
coef_nonzero	Fitting coefficients for the selected features.
coef_all	Coefficients for all selected and unselected features
intercept	Fitting intercept for the model.
log_likelihood	The negative value of the first equation above (up to a constant depending on the data set).
standardize	BOOLEAN. Whether the data was normalized (*standardize* argument was TRUE).
iteration_run	The number of iterations executed.

col_dep_var
TEXT. An expression for the dependent variable.

Both *col_dep_var* and *col_ind_var* can be valid Postgres expressions. For example, `col_dep_var = 'log(y+1)'`, and `col_ind_var = 'array[exp(x[1]), x[2], 1/(1+x[3])]'`. In the binomial case, you can use a Boolean expression, for example, `col_dep_var = 'y < 0'`.

col_ind_var
TEXT. An expression for the independent variables. Use `'*'` to specify all columns of *tbl_source* except those listed in the *excluded* string. If *col_dep_var* is a column name, it is automatically excluded from the independent variables. However, if *col_dep_var* is a valid Postgres expression, any column names used within the expression are only excluded if they are explicitly included in the *excluded* argument. It is a good idea to add all column names involved in the dependent variable expression to the *excluded* string.

regress_family
TEXT. The regression type, either 'gaussian' ('linear') or 'binomial' ('logistic').

alpha

PivotalR: Bringing MADlib and HAWQ to a Familiar R Interface

- Challenge
  Want to harness the familiarity of R's interface and the performance & scalability benefits of in-DB/in-Hadoop analytics
- Simple solution:
  Translate R code into SQL

SQL Code

```
SELECT madlib.linregr_train( 'houses',
                        'houses_linregr',
                              'price',
            'ARRAY[1, tax, bath, size]');
```

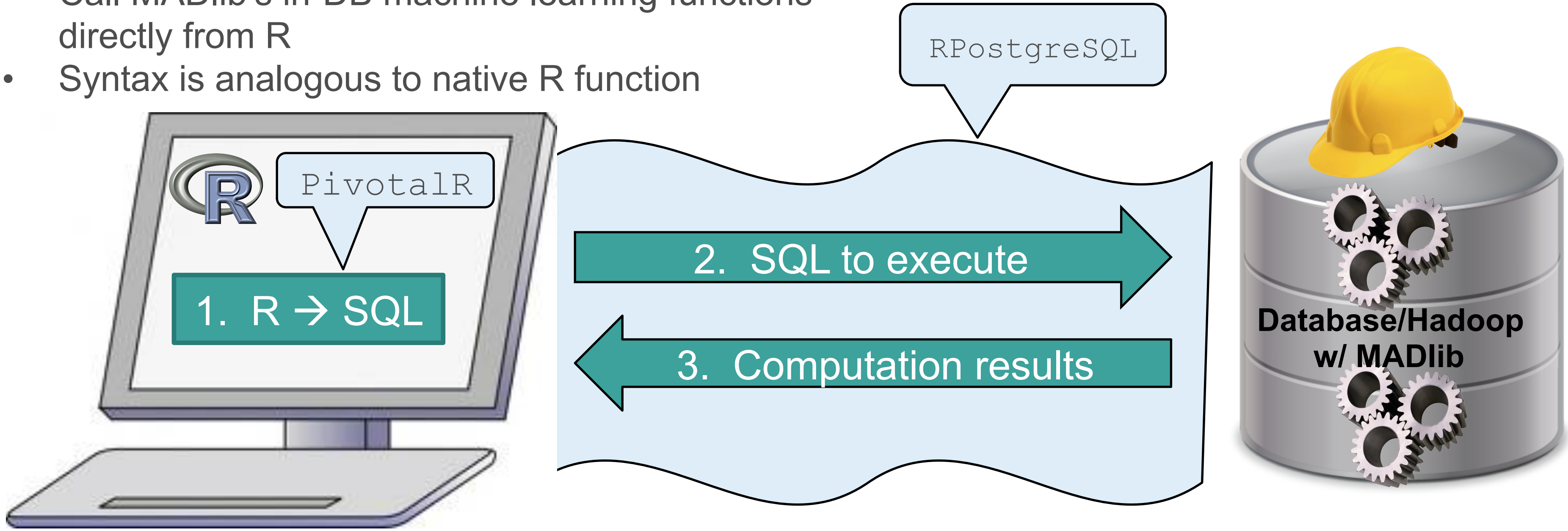
PivotalR

```
d <- db.data.frame("houses")
houses_linregr <- madlib.lm(price ~ tax
                              + bath
                              + size
                              , data=d)
```

http://cran.r-project.org/web/packages/PivotalR/index.html
https://pivotalsoftware.github.io/gp-r/
https://github.com/pivotalsoftware/PivotalR

PivotalR Design Overview

- Call MADlib's in-DB machine learning functions directly from R
- Syntax is analogous to native R function



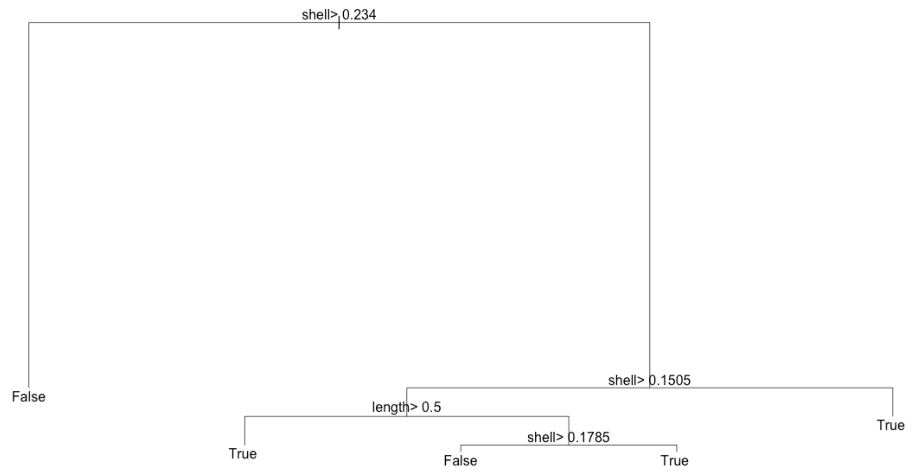
No data here

- Data doesn't need to leave the database
- All heavy lifting, including model estimation & computation, are done in the database

Data lives here

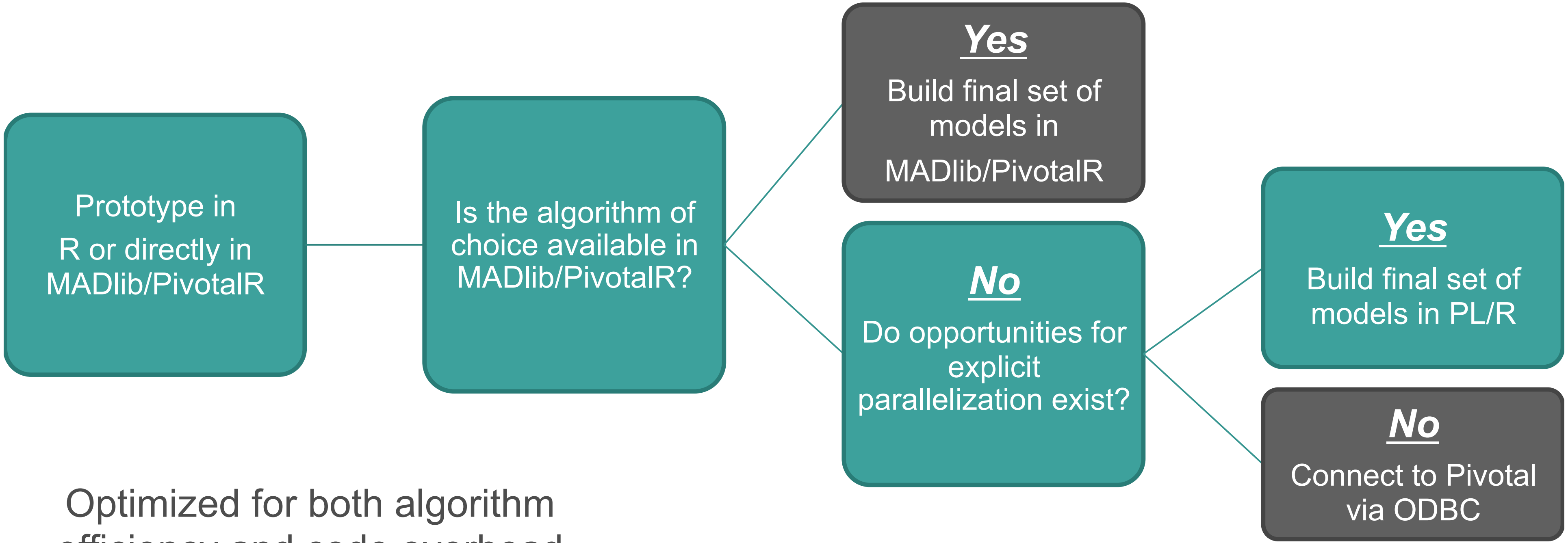
More Piggybacking

```
plot.dt.madlib
function (x, uniform = FALSE, branch = 1, compress = FALSE, nspace,
    margin = 0, minbranch = 0.3, ...)
{
    library(rpart)
    class(x) <- "rpart"
    plot(x, uniform = uniform, branch = branch, compress = compress,
        nspace = nspace, margin = margin, minbranch = minbranch,
        ...)
}
<environment: namespace:PivotalR>
```




```
fit <- madlib.rpart(rings < 10 ~ length + diameter + height + whole + shell,
                    data=x, parms = list(split='gini'), control = list(cp=0.005))
plot(fit)
text(fit)
```

How Pivotal Data Scientists Select Which Tool to Use

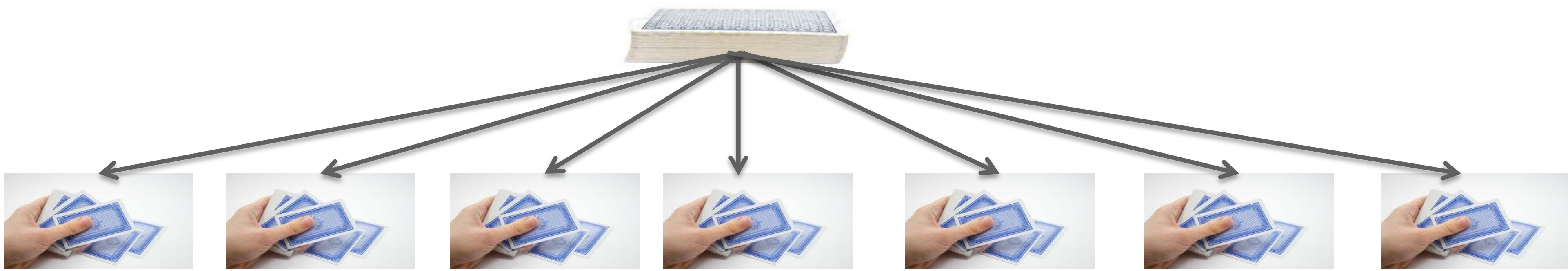


Optimized for both algorithm efficiency and code overhead

Pivotal

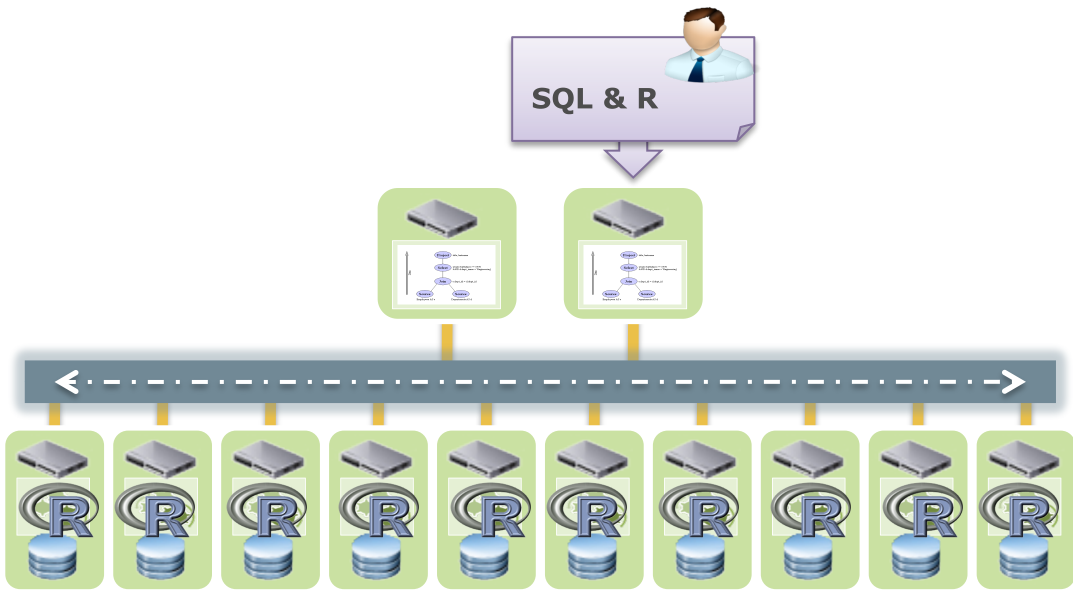
What is Data Parallelism?

- Little or no effort is required to break up the problem into a number of parallel tasks, and there exists no dependency (or communication) between those parallel tasks
- Also known as 'explicit parallelism'
- Examples:
  - Have each person in this room weigh themselves: Measure each person's weight in parallel
  - Count a deck of cards by dividing it up between people in this room: Count in parallel
  - MapReduce
  - apply() family of functions in R



Procedural Language R (PL/R)

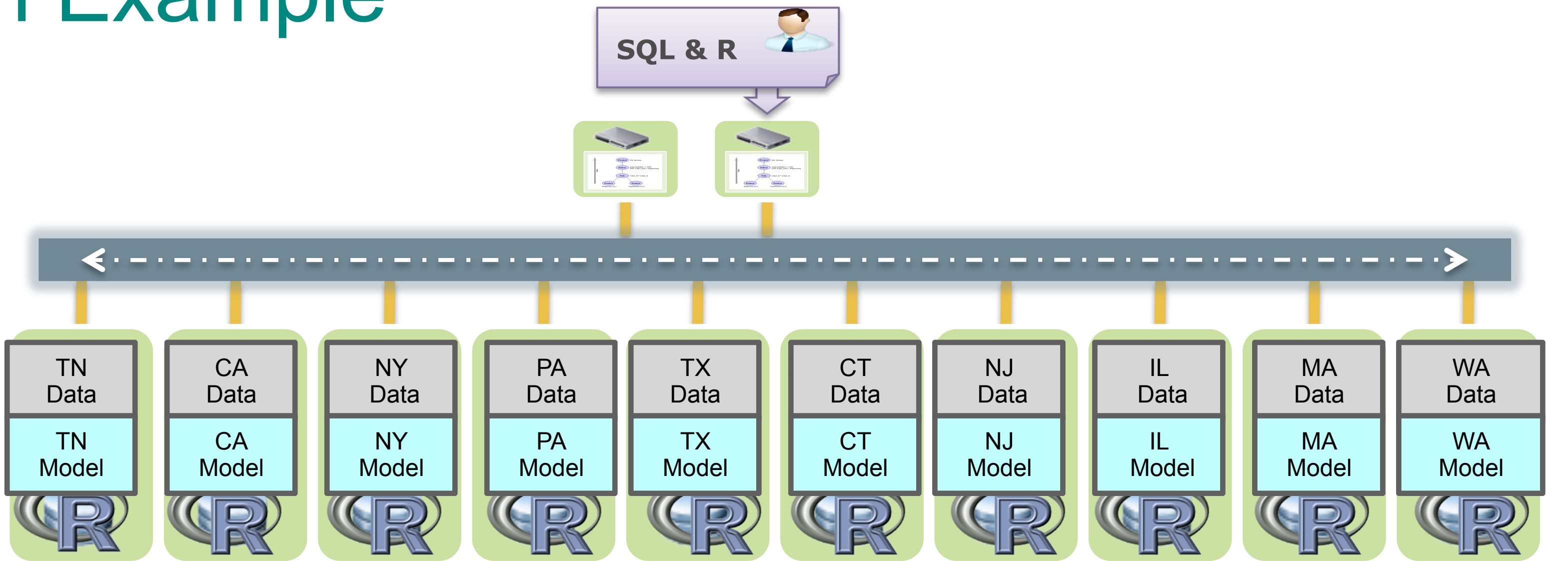
- Parallelized model building **in the R language**
- Originally developed by Joe Conway for PostgreSQL
- Parallelized by virtue of piggybacking on distributed architectures



http://pivotalsoftware.github.io/gp-r/

Pivotal

Parallelized Analytics in Pivotal via PL/R: An Example



- Parsimonious – R piggy-backs on Pivotal's parallel architecture
- Minimize data movement
- Build predictive model for each state in parallel

Pivotal

Parallelized R via PL/R: One Example of Its Use

- With placeholders in SQL, write functions in the native R language
- Accessible, powerful modeling framework

```
--Create TYPE to store model results
CREATE TYPE lm_type AS (
Variable text, Coef_Est float, Std_Error float, T_Stat float, P_Value float);

--Create PL/R function
CREATE FUNCTION lm(wage float8[], rentshouse float8[], married float8[])
RETURNS SETOF lm_type AS
$$

    m1<- lm(wage~rentshouse + married)
    m1_s<- summary(m1)$coef
    temp_m1<- data.frame(rownames(m1_s), m1_s)
    return(temp_m1)

$$
LANGUAGE 'plr';
```

Parallelized R via PL/R: One Example of Its Use

- Execute PL/R function

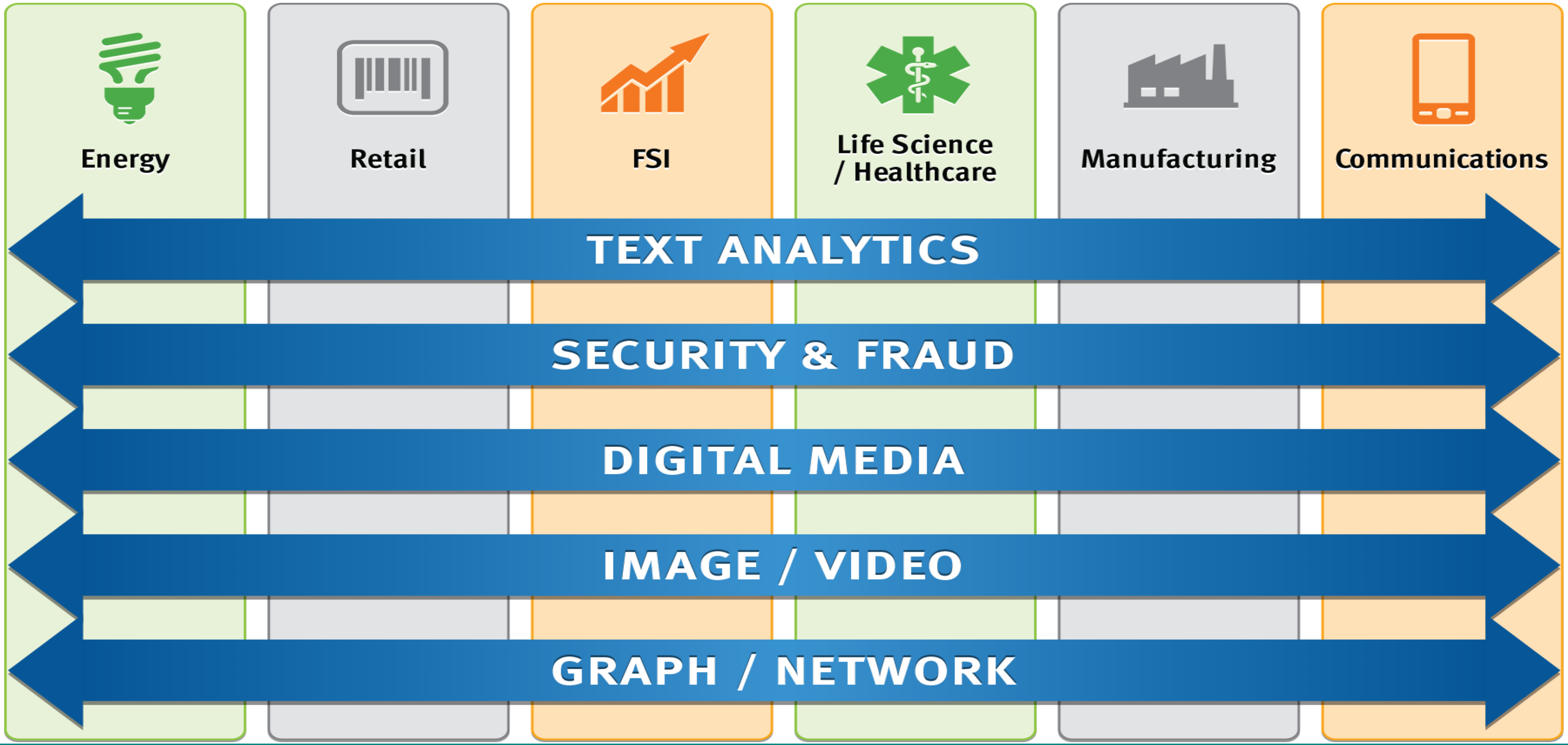
```
--Run PL/R function
SELECT h_state, (lm(wage, rentshouse, married)).* FROM use_r.census1_array_state;
```

h_state	variable	coef_est	std_error	t_stat	p_value
1	rentshouse	-153.465127622685	5.56589600570942	-27.5724029815258	3.90485302267136e-167
1	married	247.921549632485	4.58601027501907	54.0603999478511	0
1	(Intercept)	564.252659639048	3.93223032211124	143.494305627575	0
10	rentshouse	-192.916663076252	10.9051749180765	-17.6903776899967	7.67524701353641e-70
10	married	275.716955755929	9.20810244632342	29.9428636207253	1.84370671496028e-195
10	(Intercept)	724.336426473088	7.71512458444863	93.8852533805004	0
11	rentshouse	-476.714761780859	15.6795541101141	-30.4035917369202	9.4268100771193e-201
11	married	345.353553285388	16.7513113132051	20.6165085722659	5.99836808633422e-94
11	(Intercept)	1169.91840178455	13.1953649597424	88.6613144353222	0
12	rentshouse	-199.405518310734	3.19853322840866	-62.3428002997308	0
12	married	248.315061538094	2.89261212121415	85.8445761590276	0
12	(Intercept)	679.153460708787	2.46393121283416	275.63815790441	0
13	rentshouse	-204.968075908811	3.4067455844342	-60.1653604088701	0
13	married	258.41668905121	3.05488589217193	84.591273838868	0
13	(Intercept)	675.414099876061	2.63571857821484	256.254254706326	0

- Plain and simple table is returned

Examples of Usage

Pivotal Data Science: Areas of Expertise



Pivotal

Pivotal Data Science: Packaged Services

LAB PRIMER
(2-Week Roadmapping)

- Analytics Roadmap
- Prioritized Opportunities
- Architectural Recommendations

DATA JAM
(Internal DS Contest)

- Hands-on training
- Hosted data on Pivotal Data stack
- Results review & assessment

LAB 100
(Analytics Bundle)

- On-site MPP analytics training
- Analytics tool-kit
- Quick insight (2 weeks)

LAB 600
(6-Week Lab)

- Prof. services
- Data science model building
- Ready-to-deploy model(s)

LAB 1200
(12-Week Lab)

- Prof. services
- Data science model building
- Ready-to-deploy model(s)

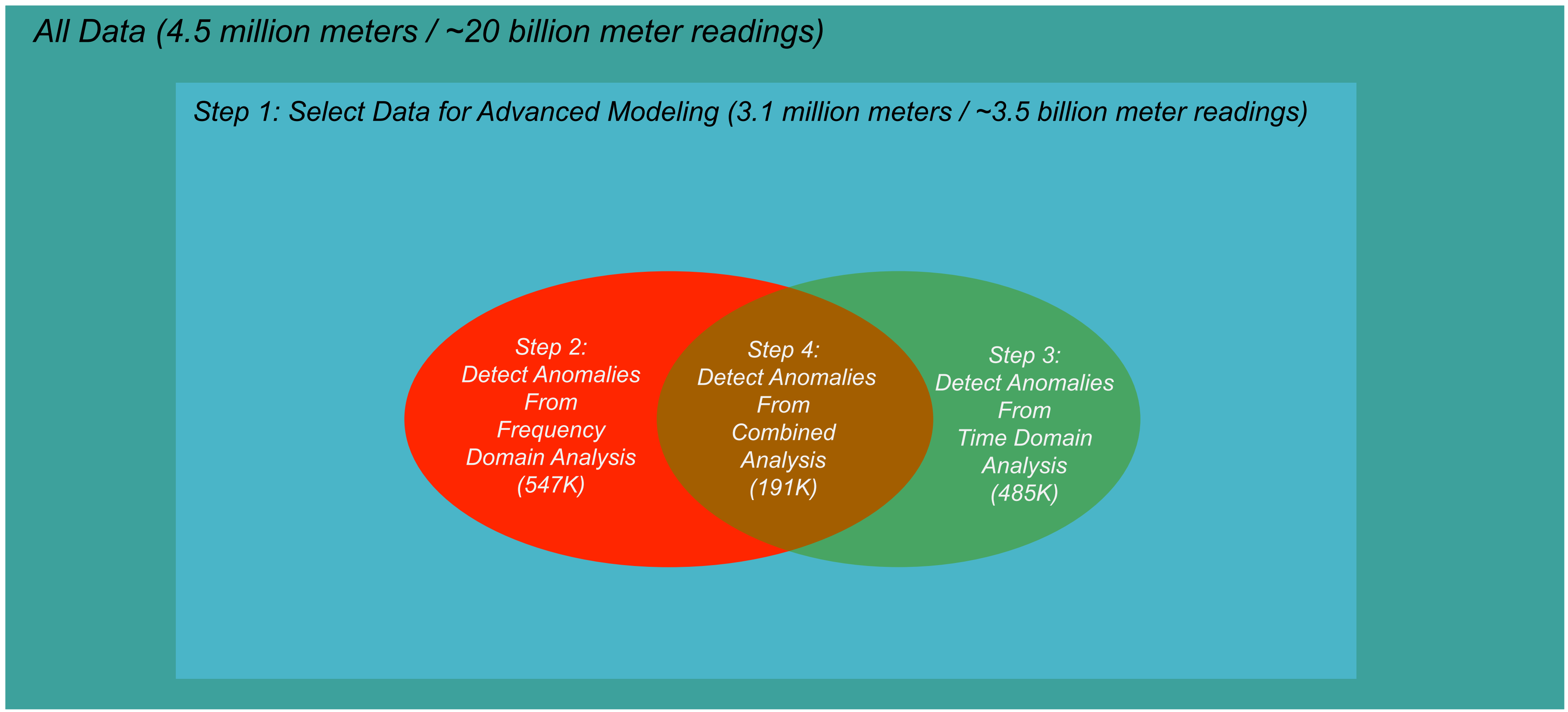
Pivotal™


The Internet of Things:
Smart Meter Analytics

Engagement Summary

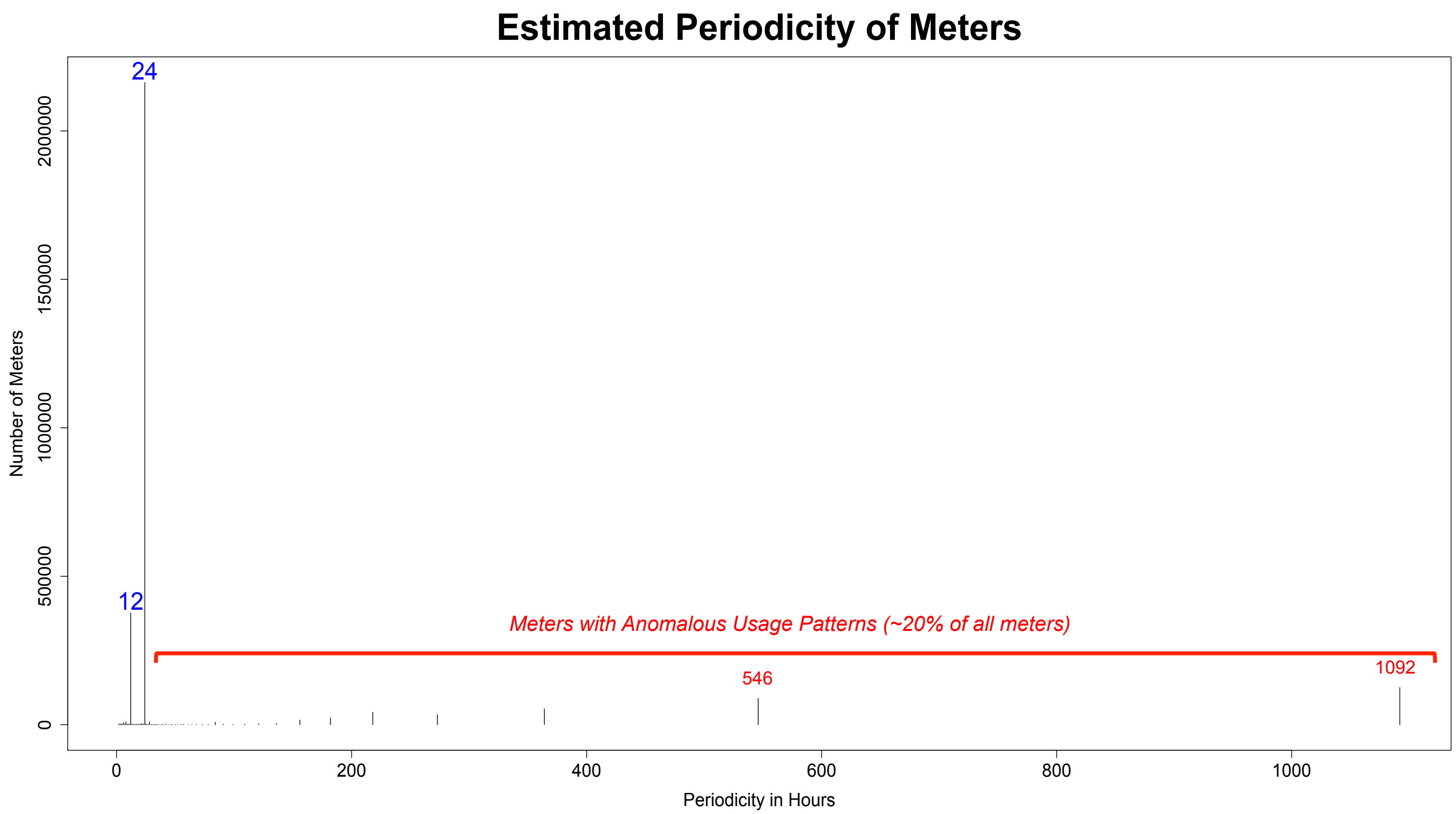
- Objective
  - Build key foundations of a data-driven framework for anomaly detection to leverage in revenue protection initiatives
- Results
  - With limited access to limited data, our models (FFT and Time Series Analysis) identified 191K potentially anomalous meters (7% of all meters).
- High Performance
  - Pivotal Big Data Suite including MADlib and PL/R
  - 90 seconds to compute FFT for over 3.1 million meters (~3.5 billion readings) → **0.0288 ms/meter**
  - ~36 minutes to compute time series models for over 3.1 million meters (~3.5 billion readings) → **0.697 ms/meter**

Anomaly Detection Methodology & Results



```
-- create type to store frequency, spec, and max freq
create type fourier_type AS (
freq text, spec text, freq_with_maxspec float8);

-- create plr function to compute periodogram and return frequency with maximum spectral density
create or replace function pgram_concise(tsval float8[])
  RETURNS float8 AS
$$
  rpgram <- spec.pgram(tsval,fast=FALSE,plot=FALSE,detrend=TRUE)
  freq_with_maxspec <- rpgram$freq[which(rpgram$spec==max(rpgram$spec))]
  return(freq_with_maxspec)
$$
LANGUAGE 'plr';

-- execute function
create table pg_gram_results
as select geo_id, meter_id, pgram_concise(load_ts) FROM
meter_data distributed by (geo_id,meter_id);
```

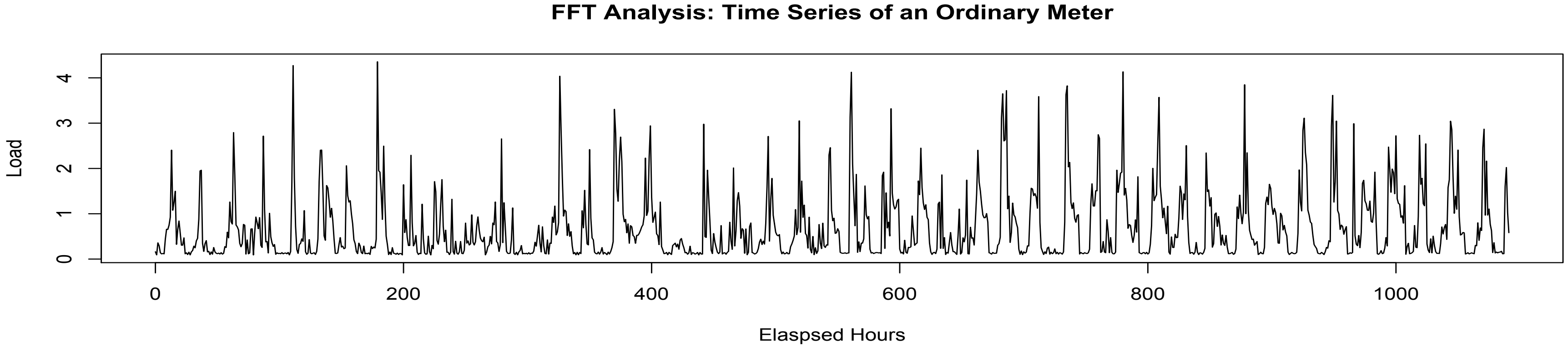
Most Households Use Energy in Daily or Half-Daily Cycles

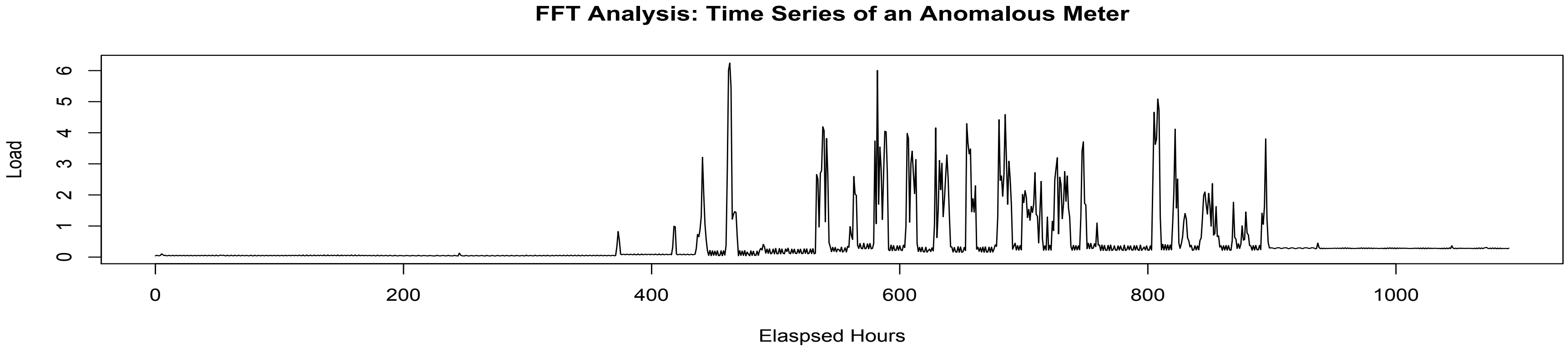


- Dominant periodicity (i.e. maximum frequency) of each meter is computed
- **~80%** of all households show **daily or half-daily patterns** of energy usage
- **~20%** of all households show **anomalous patterns** of energy usage
- Flag meters falling into the 20% as **potentially anomalous meters**
- Follow-up Items: Event type of the anomalies w.r.t. Revenue Protection to be determined with additional data & models

Pivotal

Irregular Patterns of Energy Consumption Displayed by Detected Anomalous Meters





Pivotal™

Parallelize the Generation of Visualizations

Parallelize Visualization Generation

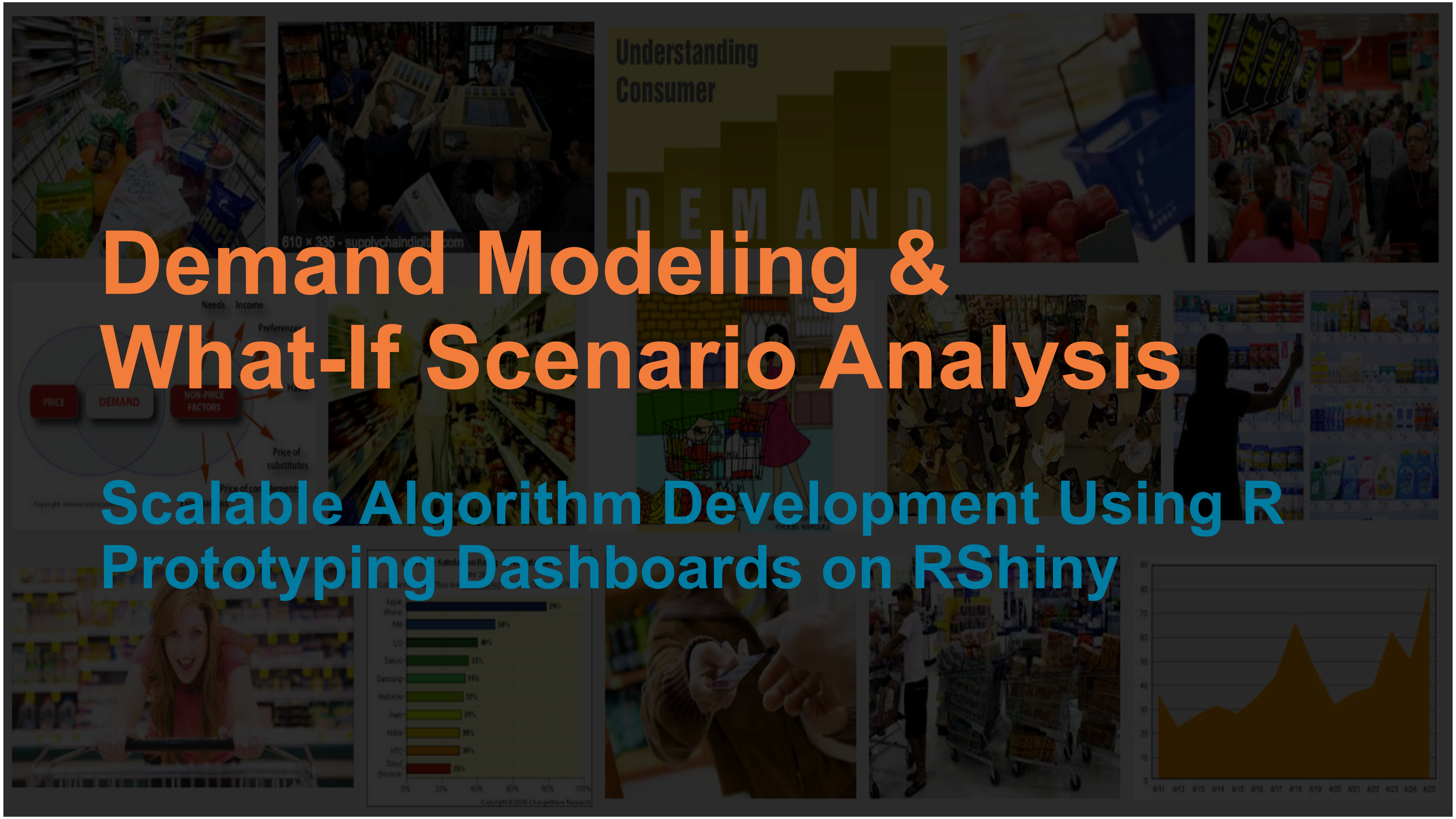
```
-- create function
create or replace function plot_pva_plr(brand text, gender text, department int4, location text, week_agg date[], actual_agg
float8[], predicted_agg float8[], wmape float8, r2 float8)
returns float8 as
$$
t2<- as.Date(week_agg)
pdf(paste("/home/gpadmin/wjung/plots_pva/", brand, "_", gender, "_", department, "_", location,
".pdf", sep=""), width=21, height=10)

# set plotting window size
par(mar=c(4, 5, 4, 5), mfrow=c(1,1))

# plot 1st series - actual units
plot(actual_agg~t2, xaxt="n", type="o", main=paste("brand_dept=",department[1], ", location=", location[1],
", weighted mape=", round(wmape[1],2)), xlab="", ylab="", cex=.7, col="red", axes=F)
axis(2, ylim=c(0,max(actual_agg)), lwd=2)
mtext(2, text="Actual Units", line=2, col="red")

# plot 2nd series - predicted units
par(new=T)
plot(predicted_agg~t2, xaxt="n", type="o", main=paste("brand_dept=",department[1], ", location=", location[1],
", weighted mape=", round(wmape[1],2)), xlab="", ylab="", cex=.7, col="blue", axes=F)
axis(4, ylim=c(0,max(actual_agg)),lwd=2)
mtext(4,text="Predicted Units",line=2, col="blue")

# plot x axis
axis(1, t2, format(t2, "%y %b %d "), cex.axis=.6, lwd=2)

dev.off()
$$
language 'plr';


--run function
select plot_pva_plr(brand, gender, department, location,
week_agg, actual_agg, predicted_agg, wmape, r2) from pva_filtered_array;
```

Parallelize Visualization Generation

Pivotal

Demand Modeling & What-If Scenario Analysis

Scalable Algorithm Development Using R
Prototyping Dashboards on RShiny



Engagement Overview

Customer's Business Goal

Make **data-driven decisions** about how to allocate resources for planning & inventory management

- Compose rich set of **reusable data assets from disparate LOBs** and make available for ongoing analysis & reporting
- Build **parallelized demand models for 100+ products & locations**
- Develop **scalable** Hierarchical/Multilevel Bayesian Modeling algorithm (Gibbs Sampling)
- Construct framework & prototype app for **what-if scenario analysis** in RShiny

Pivotal

Overview of Hierarchical Linear Model

Likelihood $\left\{ y_i \sim N(X_i B_{j[i]}, \sigma_y^2), \text{ for } i = 1, \ldots, n \right.$

Priors for parameters $\left\{ \begin{array}{l} B_j \sim N(\mu_\beta, V_\beta), \text{ for } j = 1, \ldots, J, \\ h \sim G(\underline{s}^{-2}, \underline{\nu}) \end{array} \right.$

Priors for hyperparameters $\left\{ \begin{array}{l} \mu_\beta \sim N(\underline{\mu}_\beta, \underline{\Sigma}_\beta) \\ V_\beta^{-1} \sim W(\underline{\nu}_\beta, \underline{V}_\beta^{-1}) \end{array} \right.$

Posterior $\propto$ $\underbrace{\text{Likelihood x Priors for parameters x Priors for hyperparameters}}$

This joint posterior distribution does not take the form of a known probability density, thus it is a challenge to draw samples from it directly
→ However, the full conditional posterior distributions follow known probability densities (Gibbs Sampling)

Pivotal

Game Plan

1. Figure out which components of the Gibbs Sampler can be "embarrassingly" parallelized, i.e. the key building blocks
   - Mostly matrix algebra calculations & draws from full conditional distributions, parallelized by Product-Location
2. Build functions (i.e. in PL/R) for each of the building blocks
3. Build a "meta-function" that ties together each of the functions in (2) to run a Gibbs Sampler
4. Run functions for K iterations, monitor convergence, summarize results

Pivotal™

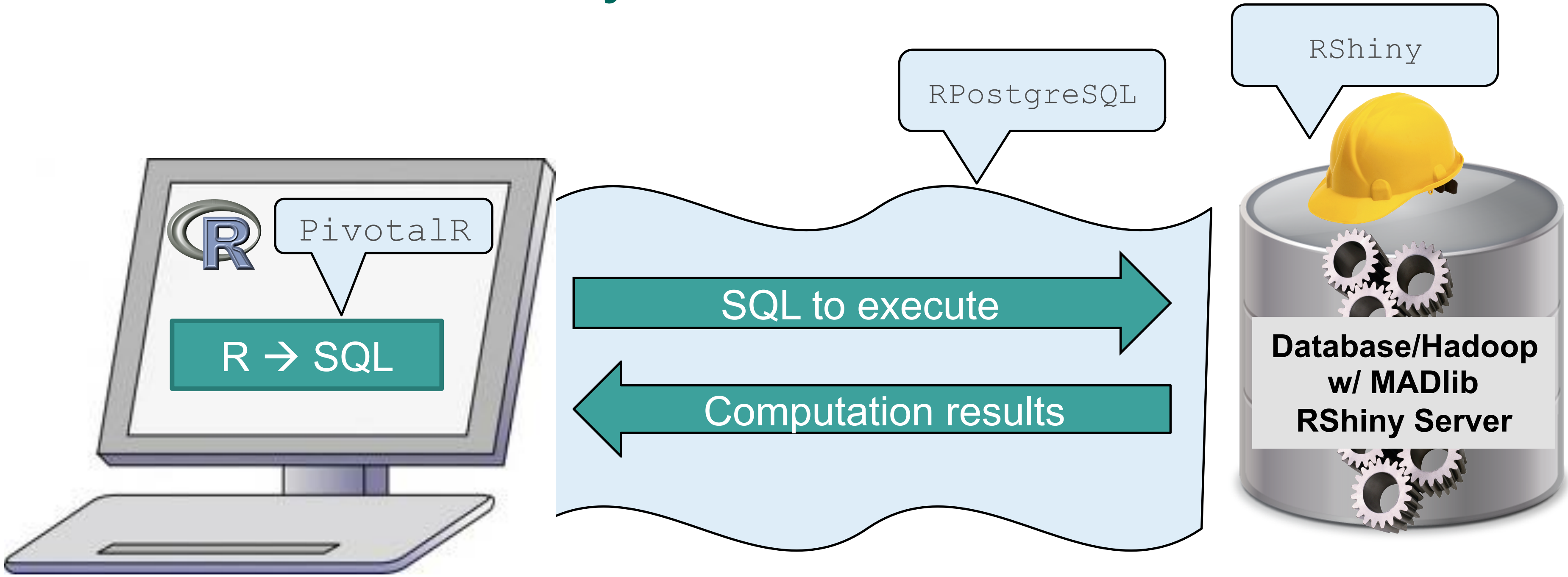
Examples of Building Block Functions

```
--Function to draw Vbeta_inv from Wishart dist'n
create or replace function Vbeta_inv_draw(float8, float8[])
returns float8[] as
$$
library(MCMCpack)
return(rwish(arg1,arg2))
$$
language 'plr';

-- Function to compute mean pooled coefficient vector to use in drawing a new pooled coefficient vector.  This function allows
for user-specified priors on the coefficients.  For use at highest level of the hierarchy.
create or replace function beta_mu_prior(float8[], float8[], float8[], float8[], float8[])
returns float8[] as
$$
beta_mu<- arg1%*%(arg2%*%arg3+as.matrix(arg4*arg5))
return(beta_mu)
$$
language 'plr';

-- Function to draw new beta. Takes mean vector of the multivariate normal distribution as its first parameter, and the
variance matrix of the multivariate normal distribution as its second parameter.  Used in cases where beta_i and beta_mu are
drawn.
create or replace function beta_draw(float8[], float8[])
returns float8[] as
$$
library(MASS)
beta_mu_draw<- mvrnorm(1, arg1, arg2)
return(beta_mu_draw)
$$
language 'plr';
```

Meta-Function & Execution

```
-- Get the first draw from the Gibbs Sampler.  Note that this drops all existing tables storing previous runs of the Gibbs
Sampler for a given model_name.  Supply a new model_name to preserve older models.
select * from
gibbs_init(
'm1'
,'d'
, 'location'
, 'department'
, 14
, 15
, 'sales'
, 'array[1, x1, x2, x3, x4, x5, x6, x7, x8, x9, x10, x11, x12, x13, x14]'
, 'array[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]'
, 'array[1,1,1,1,1,1,1,1,1,1,1,1,1,1,1]'
,1
,'random'
);

-- Update Gibbs samples.  Select beginning & ending number of iterations.
select * from
gibbs(
'm1'
, 'd'
, 'location'
, 'department'
, 14
, 15
, 'sales'
, 'array[1, x1, x2, x3, x4, x5, x6, x7, x8, x9, x10, x11, x12, x13, x14]'
, 'array[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]'
, 'array[1,1,1,1,1,1,1,1,1,1,1,1,1,1,1]'
,1
,2
,10000);
```

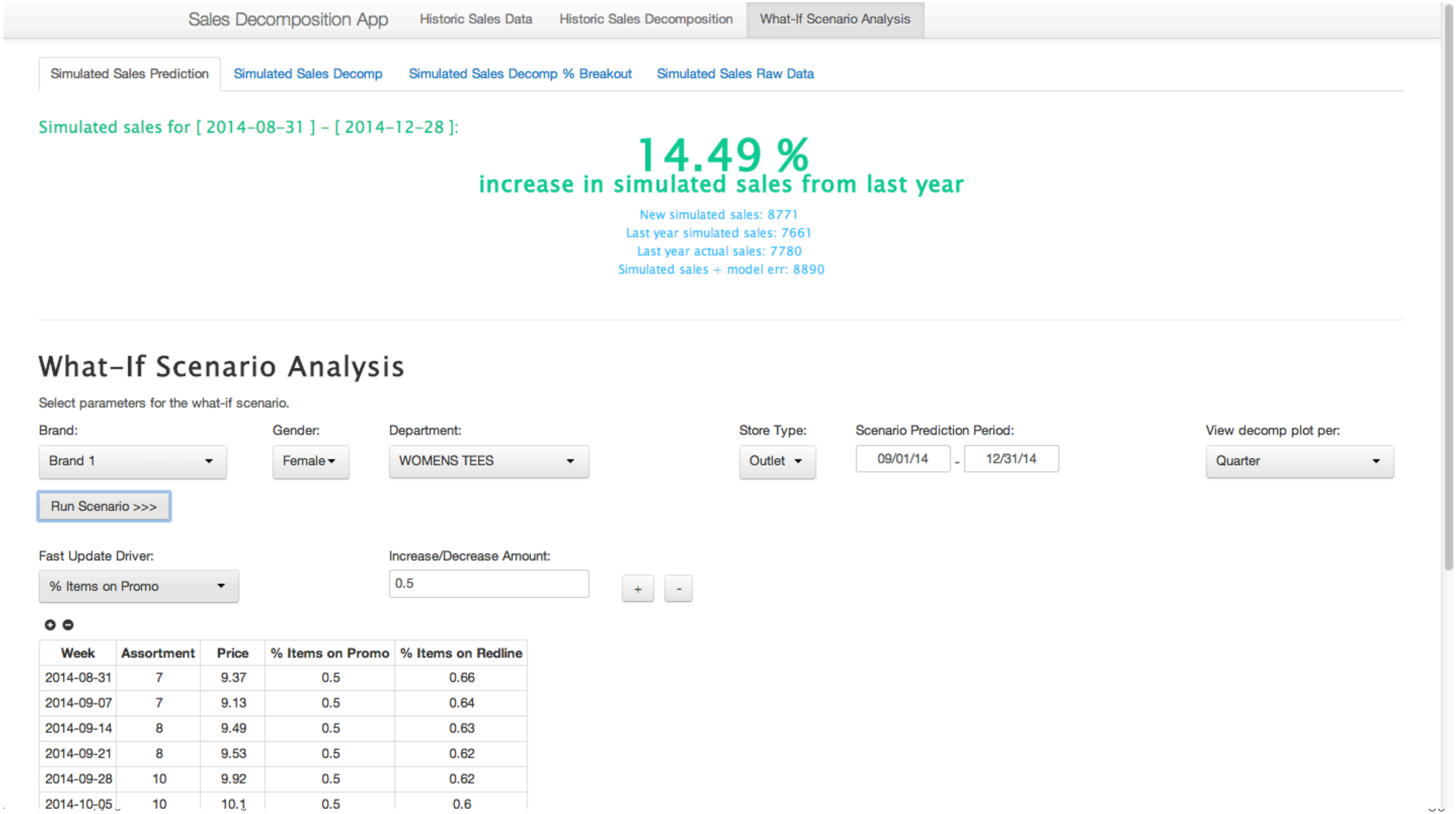
PivotalR & RShiny



No data here

- Data doesn't need to leave the database
- All heavy lifting, including model estimation & computation, are done in the database

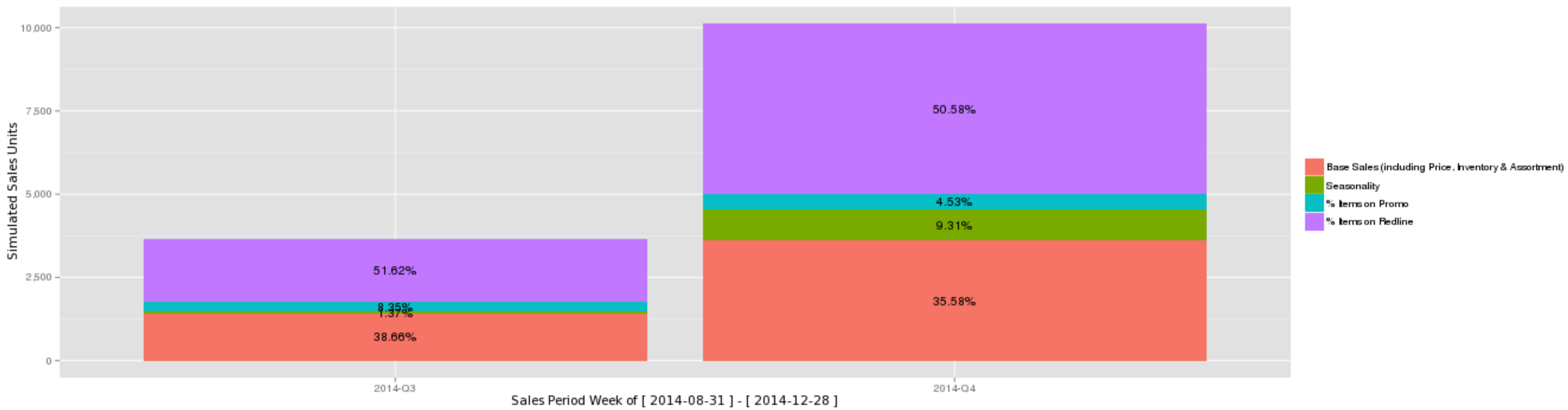
Data lives here



Sales Decomposition App
Historic Sales Data
Historic Sales Decomposition
What-If Scenario Analysis
Simulated Sales Prediction
Simulated Sales Decomp
Simulated Sales Decomp % Breakout
Simulated Sales Raw Data
Simulated sales for [2014-08-31] - [2014-12-28]:
14.49 %
increase in simulated sales from last year
New simulated sales: 8771
Last year simulated sales: 7661
Last year actual sales: 7780
Simulated sales + model err: 8890
What-If Scenario Analysis
Select parameters for the what-if scenario.
Brand:
Brand 1
Gender:
Female
Department:
WOMENS TEES
Store Type:
Outlet
Scenario Prediction Period:
09/01/14 - 12/31/14
View decomp plot per:
Quarter
Run Scenario >>>
Fast Update Driver:
% Items on Promo
Increase/Decrease Amount:
0.5
+
-
Week | Assortment | Price | % Items on Promo | % Items on Redline
2014-08-31 | 7 | 9.37 | 0.5 | 0.66
2014-09-07 | 7 | 9.13 | 0.5 | 0.64
2014-09-14 | 8 | 9.49 | 0.5 | 0.63
2014-09-21 | 8 | 9.53 | 0.5 | 0.62
2014-09-28 | 10 | 9.92 | 0.5 | 0.62
2014-10-05 | 10 | 10.1 | 0.5 | 0.6

Sales Decomposition App | Historic Sales Data | Historic Sales Decomposition | What-If Scenario Analysis

Simulated Sales Prediction | Simulated Sales Decomp | Simulated Sales Decomp % Breakout | Simulated Sales Raw Data



What-If Scenario Analysis

Select parameters for the what-if scenario.

Brand: Brand 1

Gender: Female

Department: WOMENS TEES

Store Type: Outlet

Scenario Prediction Period: 09/01/14 - 12/31/14

View decomp plot per: Quarter

Run Scenario >>>

Fast Update Driver: % Items on Promo

Increase/Decrease Amount: 0.5

\+ -

Next Steps

- Continue to build even more PivotalR wrapper functions
- Identify more areas where core R functions can be re-leveraged and made scalable via PivotalR
- Explore, learn, and share notes with other packages like PivotalR
- Explore closer integration with Spark, MLlib, H20
- PL/R wrappers directly from R

Pivotal

Thank You

Have Any Questions?

Check out the Pivotal Data Science Blog!

http://blog.pivotal.io/data-science-pivotal

Pivotal

Additional References

- PivotalR
  - http://cran.r-project.org/web/packages/PivotalR/PivotalR.pdf
  - https://github.com/pivotalsoftware/PivotalR
  - Video Demo
- PL/R & General Pivotal+R Interoperability
  - http://pivotalsoftware.github.io/gp-r/
- MADlib
  - http://madlib.net/
  - http://doc.madlib.net/latest/

BUILT FOR THE SPEED OF BUSINESS