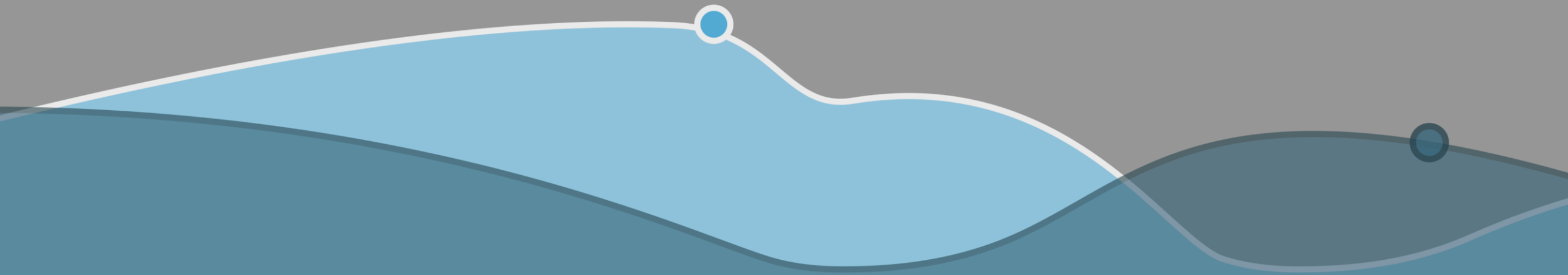
Microsoft

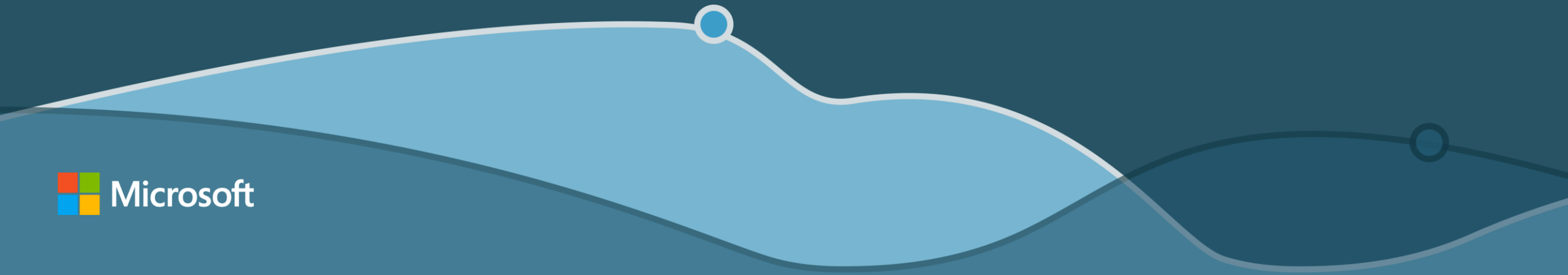
Cortana Analytics Workshop

Sept 10 – 11, 2015 • MSCC



Fundamentals of Revolution R Enterprise

Matt Parker
Associate Trainer and Data Scientist



Microsoft

Agenda

Open-Source R and Revolution R Enterprise

Open-Source R and Its Limitations
Revolution R Enterprise and Its Solutions
The XDF File Format
RevoScaleR functions and Parallel Memory Algorithms

Demo

Predicting mortgage defaults

Your Turn

Predicting flight speed

Open-Source R

Open-source
Free
Expressive
Cornucopia of packages
A bit slow
Data lives in memory

Open-Source R and Revolution R Enterprise

	**Open-source** **Free** **Expressive** **Cornucopia of packages** **A bit slow** **Data lives in memory**
REVOLUTION ANALYTICS	**Faster linear algebra libraries** **Data lives on disk** **Built for distributed computing (XDF, PEMAs)**

Revolution R and Cortana Analytics Suite

In Azure ML

RRE will soon be available as a module in Azure ML Studio, just like R and Python…

In SQL Server

… and in SQL Server, for in-database analytics

From Memory to Disk: The XDF File

The XDF File Format

Binary format for distributed computation

The XDF File Format

Binary format for distributed computation

Chunk-oriented

Append new data with minimal rewriting
Easy to distribute data among nodes

Column-oriented (within chunks)

Fast reading and writing of vectors

Pre-computed Descriptive Stats

Basic variable information, queried instantly

Managing Data with Revolution R Enterprise

Importing Data: rxImport

File Conversion

Querying Databases

Appending New Data

File Conversion

```
rxImport(inData = “somedata.csv”,
         outFile = “somedata.xdf”)
```

Querying Databases

```
dbConn <- RxOdbcData(

    connectionString =
        "Driver={SQLite3 ODBC Driver};
         Database=somedb.sqlite",

    sqlQuery = "SELECT * FROM claims"

)
```

Querying Databases

```
rxImport(inData = dbConn,
         outFile = “somedata.xdf”,
         varsToKeep = c(“somevariable”,
                        “anothervariable”),
         rowSelection = year == 2007
)
```

Appending New Data

```
rxImport(inData = “records_2015.xdf”,
         outFile = “records_all.xdf”,
         append = TRUE)
```

Appending New Data

```
# A list of paths to the CSVs
csvList <- list.files(“*.csv”, full.names = TRUE)

# The desired XDF path
xdfPath <- “appended.xdf”

# Iterate over each file
lapply(csvList, FUN = function(x) {
    rxImport(inData = x,
             outFile = xdfPath,
             append = file.exists(xdfPath))
     }
)
```

Managing Data

Subsetting

Creating and Transforming Variables

Subsetting

```
rxDataStep(inData = “fulldata.xdf”,
           outFile = “subset.xdf”,
           varsToKeep = c(“balance”,
                          “date”,
                          “accountTenure”),
           rowSelection = year < 2011
)
```

Creating and Transforming Variables

```
rxDataStep(inData = “fulldata.xdf”,
           outFile = “fulldata.xdf”,
           transforms = list(
               ageSquared = age^2,
               ageLog = log(age),
               unifRandom = runif(.rxNumRows)),
           overwrite = TRUE
)
```

Performance Note

Data manipulation functionality is available in many RevoScaleR functions

rxImport, rxDataStep, rxSummary, etc.

For well-defined processes, pack as much into one function as you can

Fewer function calls = fewer reads of the data = faster

Data Mgmt Args in Most rxFunctions

```
varsToKeep, varsToDrop

rowSelection

transforms (etc.)

numRows
```

Managing Data Across Chunks

Centering

Lagging

Factors

Centering

```
# First, calculate the mean
ageMean <- rxSummary( ~ age, data = xdfPath)$sData$Mean

# Next, make a mean-centering function
centerVar <- function(dataList) {
     dataList[[newName]] <-
         dataList[[varToCenter]] - dataList[[varMean]]
    return(dataList)
}
```

Centering

```
rxDataStep(inData = xdfPath,
           outFile = xdfPath,
           transformObjects = list(
               varMean = ageMean,
               varToCenter = age,
               newName = “ageCentered”),
           transformFunc = centerVar,
           append = “cols”,
           overwrite = TRUE
)
```

Lagging

```
# Sort first
rxSort(inData = xdfPath,
       outFile = xdfPath,
       sortByVars = “date”)

rxDataStep(inData = xdfPath,
           outFile = xdfPath,
           transformObjects = list(
               varToLag = "balance",
               newName = "priorBalance"),
           transformFunc = lagFunc,
           append = "cols",
           overwrite = TRUE)
```

Lagging in Open-Source R

```
someData$priorBalance <- c(
   NA,
   someData$priorBalance[-length(someData$priorBalance)]
)
```

Month	origVar	laggedVar
Jan	2	NA
Feb	4	2
Mar	6	4

Lagging in Revolution R Enterprise

```
lagFunc <- function(dataList) {
  if(.rxStartRow == 1) {
    dataList[[newName]] <- c(NA,
                             dataList[[varToLag]][-.rxNumRows])
  } else {
    dataList[[newName]] <- c(.rxGet("lastValue"),
                             dataList[[varToLag]][-.rxNumRows])
  }
  .rxSet("lastValue", dataList[[varToLag]][.rxNumRows])
  return(dataList)
}
```

Factors

```
rxFactors(inData = xdfPath,
          outFile = xdfPath,
          factorInfo = list(
              region = list(
    newLevels = c(Midwest = c("Ohio", "Illinois"),
                  Mountains = c("Utah", "Idaho")),
    otherLevel = "Somewhere Else"))
)
```

Analyzing Data with Parallel External Memory Algorithms

Summarizing Data

rxGetInfo

rxSummary, rxQuantile

rxCrossTabs, rxCube

Visualizing Data

rxHistogram

rxLinePlot

Analyzing Data

rxCor

rxLinMod

rxLogit

(and many others)

Next Steps

Hands-On Tutorial

On the next slide

Session SES34: Integrating SQL Server and Revo R

Here at 1pm

Attend SES39: On-Prem Hadoop and Revo R Solutions

In Baker at 2pm

Watch for RRE in Azure ML and SQL Server

Your Turn

The Dataset

US domestic flights from 2007

7.5M records in total; start with 100,000

`file.path(Sys.getenv("ACADEMYR_BIG_DATA_PATH"), “2007.csv”))`

Functions You’ll Need

`rxImport, rxHistogram, rxDataStep, rxLinMod`

Get documentation by prefixing a question mark: `?rxDataStep`

Matt’s code from a moment ago: http://aka.ms/rre_fun

If You Finish Early

Try an advanced model: `rxGlm, rxDTree, rxDForest`

Set `numRows = -1` to try working with all 7.5M flights

Microsoft