Using Hadoop and Hive to Optimize Travel Search

Jonathan Seidman and Ramesh Venkataramaiah

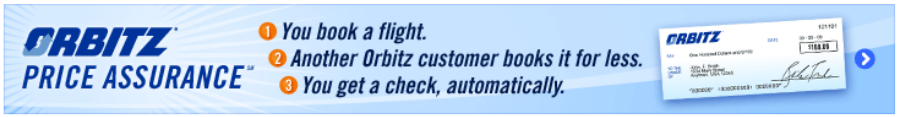
Contributors

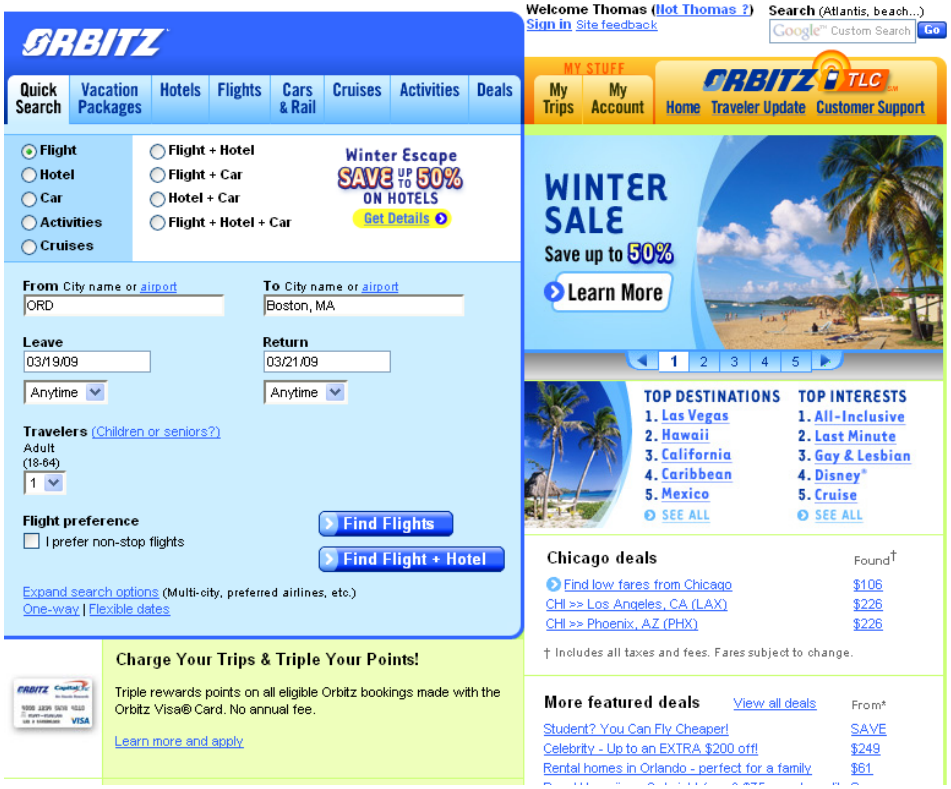
- Robert Lancaster, Orbitz Worldwide
- Wai Gen Yee, Orbitz Worldwide
- Andrew Yates, Intern - Orbitz Worldwide

Agenda

- Orbitz Worldwide
- Hadoop for Big Data
- Hive for Queries
- Web Analytics data as input
- Applications of Hadoop/Hive at Orbitz:
  - Hotel Sort
  - Data Cubes
- Sample analysis and data trends

Launched: 2001, Chicago, IL





Data Challenges

- Orbitz.com generates ~1.5 million air searches and ~1 million hotel searches every day.
- All of this activity generates massive amounts of data – over 500 GB/day of log data, and even this doesn't capture all of the data we want.
- Expensive and difficult to use existing data infrastructure for storing and processing this data.
- Need an infrastructure that provides:
  - Long term storage of very large data sets.
  - Open access to developers and analysts.
  - Allows for ad-hoc querying of data and rapid deployment of reporting applications.

Hadoop Overview

- Open source framework providing reliable and scalable storage and processing of data on inexpensive commodity hardware.
- Two primary components: The Hadoop distributed file system and MapReduce.

Hadoop Overview – Hadoop Distributed File System

- HDFS provides reliable, fault tolerant and scalable storage of very large datasets across machines in a cluster.

Hadoop Overview – MapReduce

- Programming model for efficient distributed processing. Designed to reliably perform computations on large volumes of data in parallel.
- Removes much of the burden of writing distributed computations.

The Problem with MapReduce

- Requires experienced developers to write MapReduce jobs which can be difficult to maintain and re-use.

Hive Overview

- Hive is an open-source data warehousing solution built on top of Hadoop which allows for easy data summarization, adhoc querying and analysis of large datasets stored in Hadoop.
- Developed at Facebook to provide a structured data model over Hadoop data.
- Simplifies Hadoop data analysis – users can use a familiar SQL model rather than writing low level custom code.
- Hive queries are compiled into Hadoop MapReduce jobs.
- Designed for scalability, not low latency.

Hive Overview – Comparison to Traditional DBMS Systems

- Although Hive uses a model familiar to database users, it does not support a full relational model and only supports a subset of SQL.
- What Hadoop/Hive offers is highly scalable and fault-tolerant processing of very large data sets.

Hive - Data Model

- Databases – provide namespace for Hive objects, prevent naming conflicts.
- Tables – analogous to tables in a standard RDBMS.
- Partitions and buckets – Allow Hive to prune data during query processing.

Hive – Data Types

- Supports primitive types such as int, double, and string.
- Also supports complex types such as structs, maps (key/value tuples), and arrays (indexable lists).

Hive – Hive Query Language

- HiveQL – Supports basic SQL-like operations such as select, join, aggregate, union, sub-queries, etc.
- HiveQL queries are compiled into MapReduce processes.
- Supports embedding custom MapReduce scripts.
- Built in support for standard relational, arithmetic, and boolean operators.

Hive MapReduce

- Allows analysis not possible through standard HiveQL queries.
- Can be implemented in any language.

Hive – User Defined Functions

- HiveQL is extensible through user defined functions implemented in Java.
- Also supports aggregation functions (sum, avg).
- Provides table functions when more than one value needs to be returned.

Hive – User Defined Functions

Example UDF – Find hotel's position in an impression list:

```
package com.orbitz.hive;
import org.apache.hadoop.hive.ql.exec.UDF;
import org.apache.hadoop.io.Text;

/**
 * returns hotel_id's position given a hotel_id and impression list
 */
public final class GetPos extends UDF {
    public Text evaluate(final Text hotel_id, final Text impressions) {
        if (hotel_id == null || impressions == null)
            return null;

        String[] hotels = impressions.toString().split(";");
        String position;
        String id = hotel_id.toString();
        int begin=0, end=0;

        for (int i=0; i<hotels.length; i++) {
            begin = hotels[i].indexOf(",");
            end = hotels[i].lastIndexOf(",");
            position = hotels[i].substring(begin+1,end);
            if (id.equals(hotels[i].substring(0,begin)))
                return new Text(position);
        }
        return null;
    }
}
```

Hive – User Defined Functions

```
hive> add jar path-to-jar/pos.jar;
hive> create temporary function getpos as
  'com.orbitz.hive.GetPos';
hive> select getpos('1',
  '1,3,100.00;2,1,100.00');
…
hive> 3
```

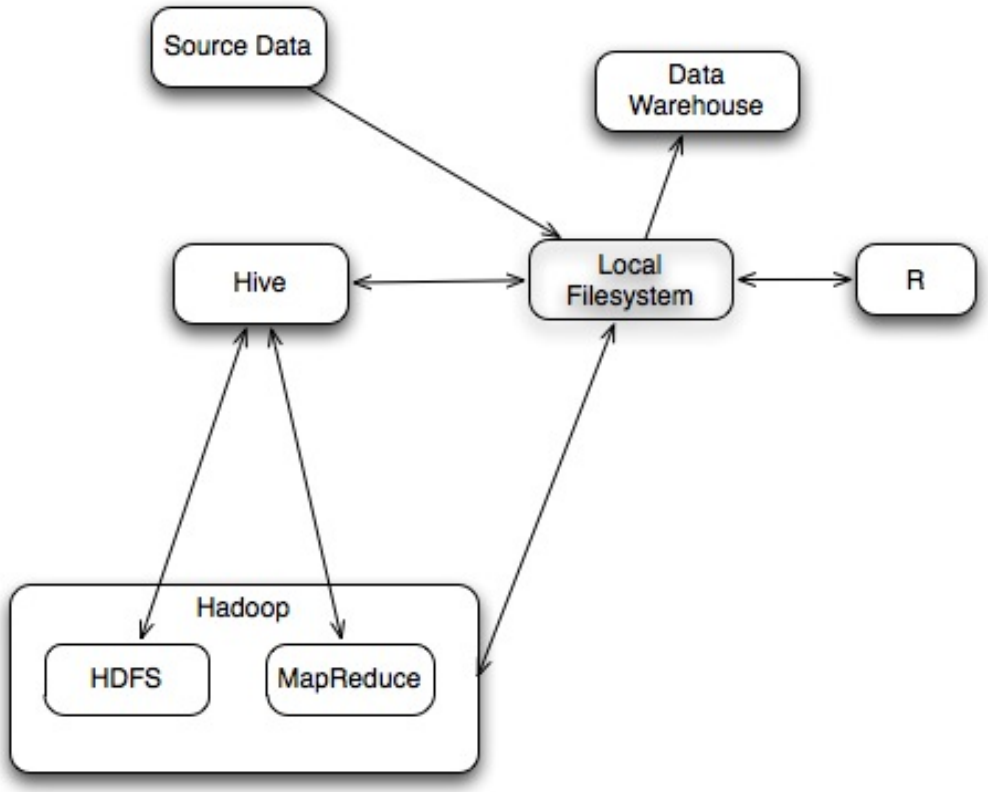
Hive – Client Access

- Hive Command Line Interface (CLI)
- Hive Web UI
- JDBC, ODBC, Thrift

Hive – Lessons Learned

- Job scheduling – Default Hadoop scheduling is FIFO. Consider using something like the fair scheduler.
- Multi-user Hive – Default install is single user. Multi-user installs require Derby network server.

Data Infrastructure



Input Data – Webtrends

- Web analytics software providing information about user behavior.
- Raw Webtrends log files are used as input to much of our processing. Logs have the following format:
  - `date time c-ip cs-username cs-host cs-method cs-uri-stem cs-uri-query sc-status sc-bytes cs-version cs(User-Agent) cs(Cookie) cs(Referer) dcs-geo dcs-dns origin-id dcs-id`

Input Data – Webtrends Cont'd

- Storing raw data in HDFS provides access to data not available elsewhere, for example "hotel impression" data:
  - `115004,1,70.00;35217,2,129.00;239756,3,99.00;83389,4,99.00`

Chicago - Hotel Search Results - ...

Welcome to Orbitz | Sign in | Register

ORBITZ

My Trips | My Account | Traveler Update | Customer Support

NO FEES

Quick Search | Vacation Packages | Hotels | Flights | Cars & Rail | Cruises | Activities | Deals

Search

Change search

Chicago, Illinois, United States | Check-in: Fri, Jun 25, 2010 | Check-out: Sat, Jun 26, 2010 | Nights: 1 | Room(s): 1 | Guest(s): 1

Saved (0)

Call us to book!
1-800-733-1297
(toll free)

Why book on Orbitz? Price Assurance + Low Price Guarantee + No Orbitz hotel change or cancel fees Learn more

Map

Large map

Map data ©2010 Google - Terms of Use

Clear Selections

Locations

Any

Magnificent Mile

The Loop, Grant Park, Millennium Park

439 total matching hotels

Previous | Next

Sort by: Best Values | Lowest Price | Distance | Star Rating | Hotel Name

Crowne Plaza Hotel CHICAGO-METRO

Select

★★★☆☆ Sponsored Listing What's this?

Luxury in the Loop At Reasonable Rates

Cutting-edge luxury awaits you in our sleek and modern guest rooms and suites offering romantic views of the Chicago Skyline More

The Whitehall Hotel

★★★★☆ Save

Nightly rates from ~~$305~~ $229

Total Price $261

Select

Price Assurance

Overview | Description | Photos | Map | Amenities

Reviewer score 3.8 out of 5 488 reviews

105 E. Delaware Place, Chicago, IL 60611
0.2 miles North from the center of Chicago

[+] Site Feedback

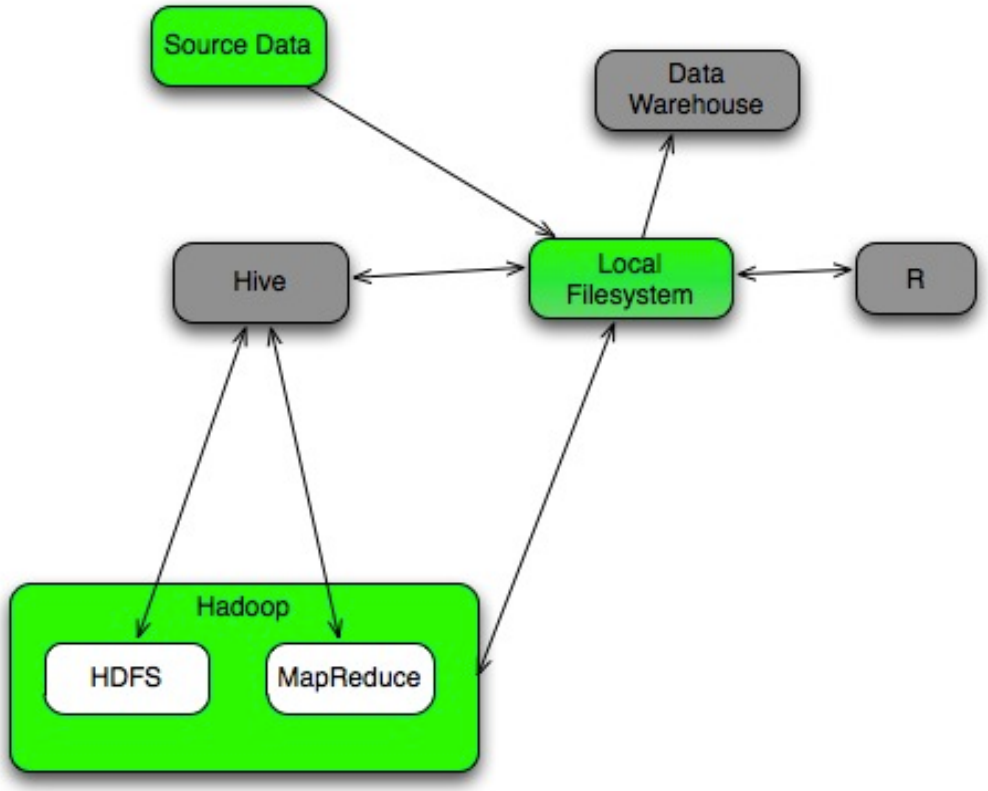
Find: Next | Previous | Highlight all | Match case

Done

Improve Hotel Sort

- Extract data from raw Webtrends logs for input to a trained classification process.
- Logs provide input to MapReduce processing which extracts required fields.
- Previous process used a series of Perl and Bash scripts to extract data serially.
- Comparison of performance
  - Months worth of data
  - Manual process took 109m14s
  - MapReduce process took 25m58s

Improve Hotel Sort – Components



Improve Hotel Sort – Processing Flow

Webtrends Analysis in Hive

- Extract data is loaded into two Hive tables:

```
DROP TABLE wt_extract;
CREATE TABLE wt_extract(
      session_id STRING,  visitor_tracking_id STRING,  host STRING,  visitors_ip STRING,  booking_date STRING,  booking_time
      STRING,   dept_date STRING,  ret_date STRING,  booked_hotel_id STRING,  sort_type STRING,  destination STRING,  location_id
      STRING,  number_of_guests INT,  number_of_rooms INT,  page_number INT,  matrix_interaction STRING,   impressions STRING,
      areacode STRING,  city STRING,  region_code STRING,  country STRING,  country_code STRING,  continent STRING,  company STRING,
      tzone STRING)
CLUSTERED BY(booked_hotel_id) INTO 256 BUCKETS
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '\t'
STORED AS TEXTFILE;

load data inpath '/extract-output/part-00000' into table wt_extract;

DROP TABLE hotel_impressions;
CREATE TABLE hotel_impressions(
    session_id STRING,   hotel_id STRING,   position INT,  rate FLOAT )
CLUSTERED BY(hotel_id) INTO 256 BUCKETS
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '\t' STORED AS TEXTFILE;

load data inpath '/impressions-output/part-00000' into table hotel_extract;
```

Webtrends Analysis in Hive Cont'd

- Allows us to easily derive metrics not previously possible.
- Example - Find the Position of Each Booked Hotel in Search Results:

```
CREATE TABLE positions(
  session_id STRING,
  booked_hotel_id STRING,
  position INT);
set mapred.reduce.tasks = 17;
INSERT OVERWRITE TABLE
   positions
SELECT
   e.session_id, e.booked_hotel_id, i.position
FROM
   hotel_impressions i JOIN wt_extract e
ON
   (e.booked_hotel_id = i.hotel_id and e.session_id = i.session_id);
```

Webtrends Analysis in Hive Cont'd

- Example - Aggregate Booking Position by Location by Day:

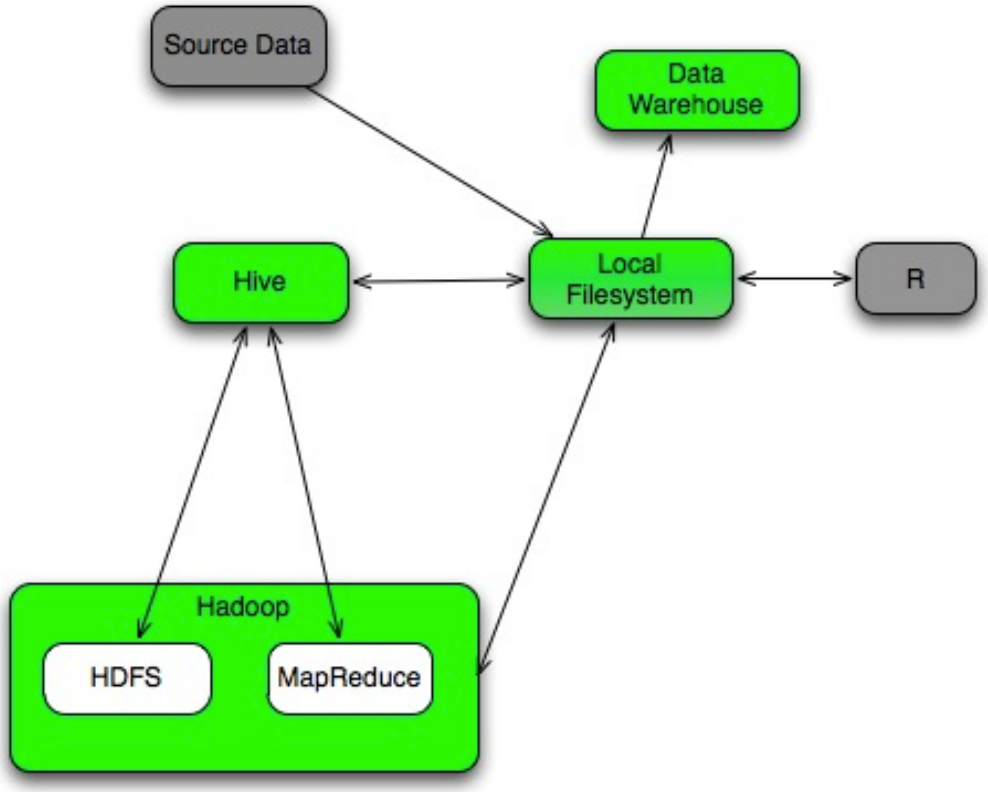
```
CREATE TABLE position_aggregate_by_day(
  location_id STRING,
  booking_date STRING,
  position INT,
  pcount INT);

INSERT OVERWRITE TABLE
  position_aggregate_by_day
SELECT
  e.location_id, e.booking_date, i.position, count(1)
FROM
  wt_extract e JOIN hotel_impressions i
ON
 (i.hotel_id = e.booked_hotel_id and i.session_id = e.session_id)
GROUP BY
  e.location_id,e.booking_date,i.position
```

Hotel Data Cube

- Goal is to provide users with data not available in existing hotel cube.
- Problem is lack of Hive support in existing visualization tools.

Hotel Data Cube – Components



Statistical Analysis of Hive Data

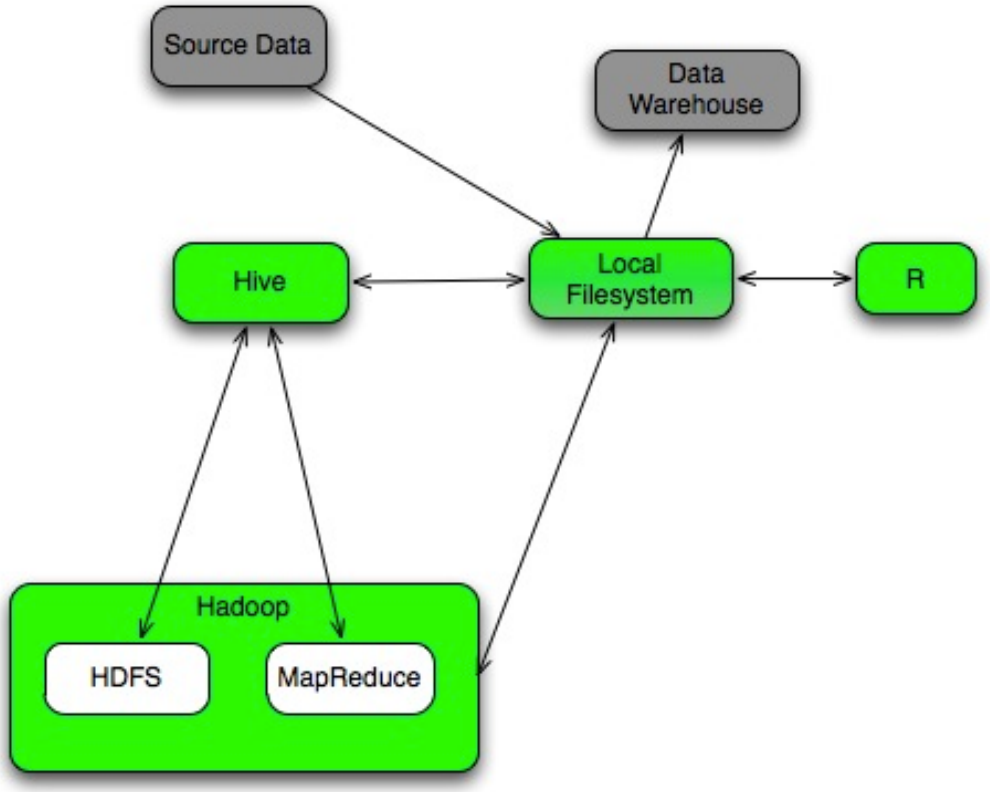
Explore statistical trends and long-tails…

to help machine learning algorithms…

by choosing well understood input datasets.

- What approximations and biases exist?
- How is the Data distributed?
- Are 25+ variables pair-wise correlated?
- Are there built-in data bias? Any Lurking variables?
- Are there outlier effects on the distribution?
- Which segment should be used for ML training?

Statistical Analysis – Components



Statistical Analysis: Infrastructure and Dataset

- Hive + R platform for query processing and statistical analysis.
- R - Open-source stat package with visualization.
- R - Steep learning curve but worth it!
- R – Vibrant community support.
- Hive Dataset used:
  - Customer hotel booking on our sites over 6 months.
  - Derived from web analytics extract files from Hive.
  - User rating of hotels.
  - Captured major holiday travel bookings but not summer peak.
- **Costs of cleaning and processing data is non-trivial.**

Some R code…

```
HIVE> Insert overwrite local directory '/home/user/…/HiveExtractTable'
Select * from . . .
```

```
R> WTData <- data.frame(scan("/user/.../HiveExtractTable", what = list(0,"",0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,"",
   0,"",""),na.strings="NA"))
R> names(CHI) <- c("locationID","hotelID","rate","distance","amenities","checkin", "cleanliness", "comfort",
   "dining", "location", "bookposition", "guests", "competitiveness", "departuredate", "returndate", "stay")
CHI <- WTData[which(WTData[,1] == '7840'|'35416'|'37961'|'38422'|'37795'| '37769'|'37730'|'33728'|'38123'),]
R> summary(CHI)
```

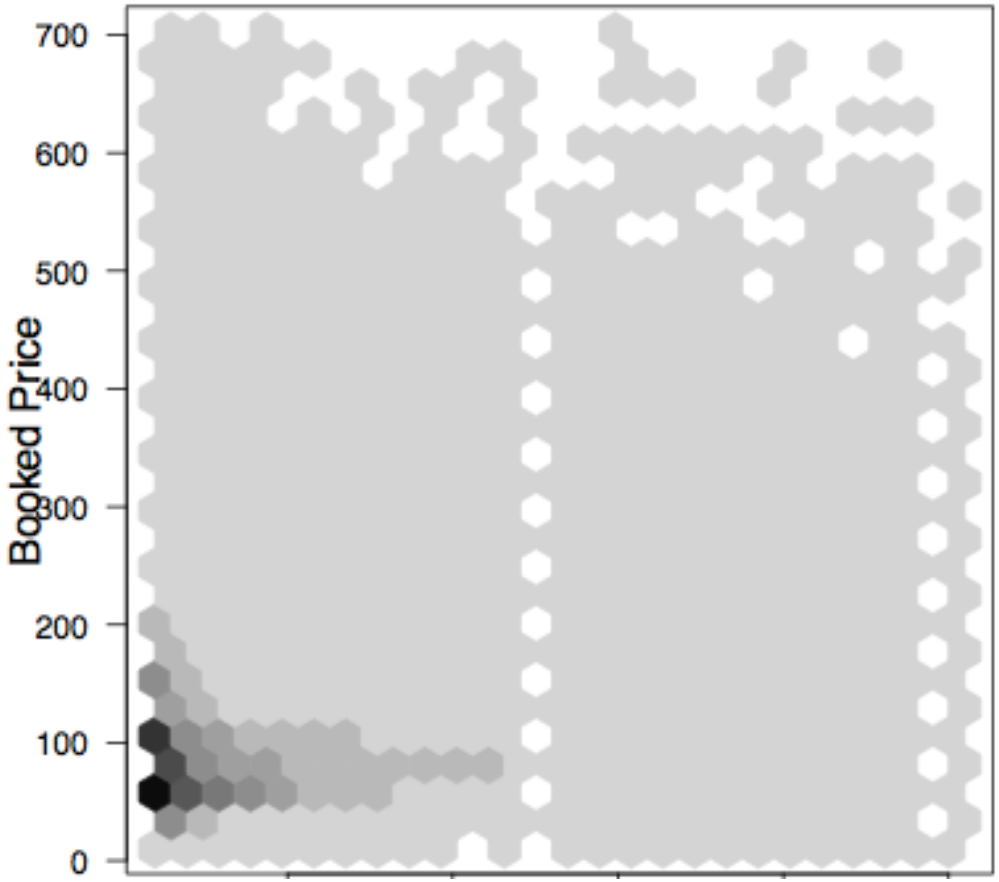
```
R> library(corrgram)
R> corrgram(CHI,order=NULL,lower.panel=panel.ellipse)
```

```
R> pdf(file="/user/.../himp.pdf")
R> par(mfrow=c(2,1))
R> bin1 <-hexbin(CHI[,15],CHI[,4], xbins=25)
R> hsmooth(bin1,c(48,24,0))
R> plot(bin1,xlab="Position", ylab="Price($)", main="Booked Hotel Impressions", colramp=function(n){plinrain
   (n,beg=35,end=225)})
R> bin2 <-hexbin(CHI[,15], CHI[,22], xbins=25)
```

```
R> png("/user/. . ./HM_agg_CHI.png",width=480,height=480,units="px",pointsize=12)
R> CHI <- aggdata[which(aggdata$position <=10),]
R> par(mfrow=c(2,2))
R> calendarHeat(CHI$Time,CHI$rate, varname="Aggregated Check-in Date vs. Avg. Daily Rate")
```

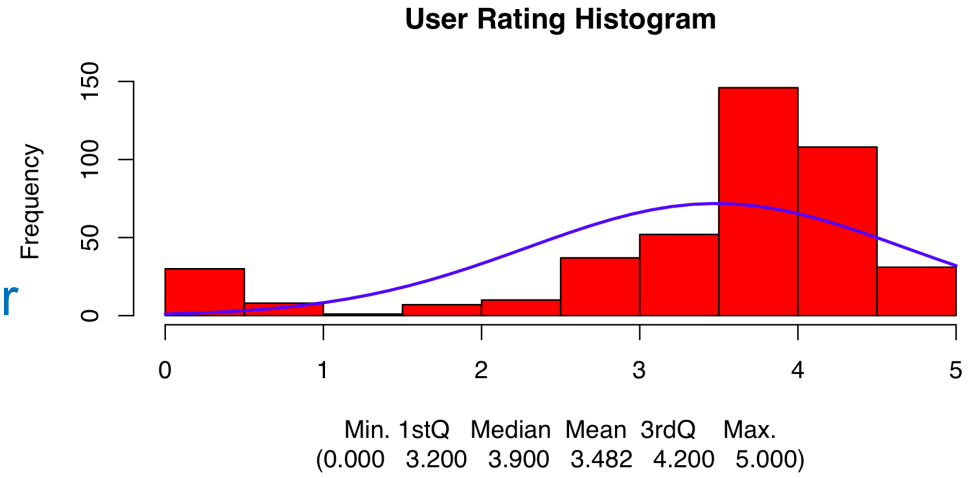
Statistical Analysis - Positional Bias

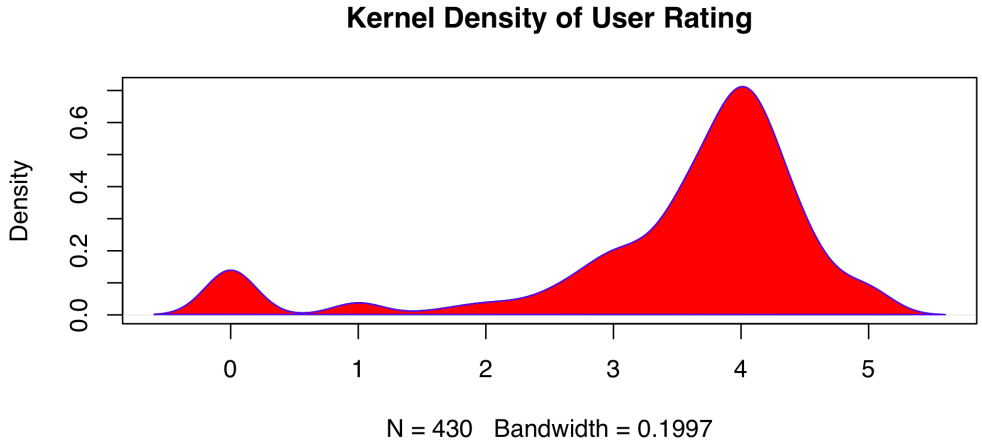
- Lurking variable is… Positional Bias.
- Top positions invariably picked the most.
- Aim to position Best Ranked Hotels at the top based on customer search criteria and user ratings.
- **If website originated data, watch for inherent hidden bias.**



Statistical Analysis - Kernel Density

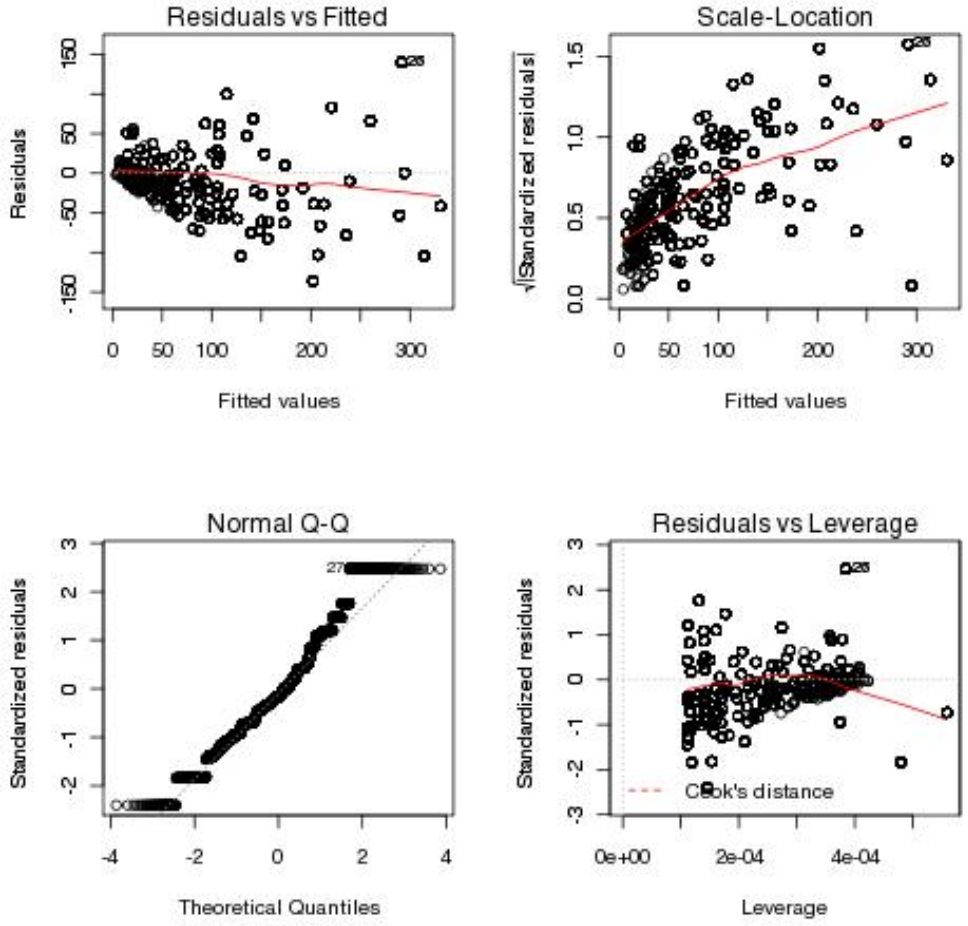
- User Ratings of Hotels
- Strongly affected by the number of bins used.
- Kernel density plots are usually a much more effective way to overcome the limitations of histograms.



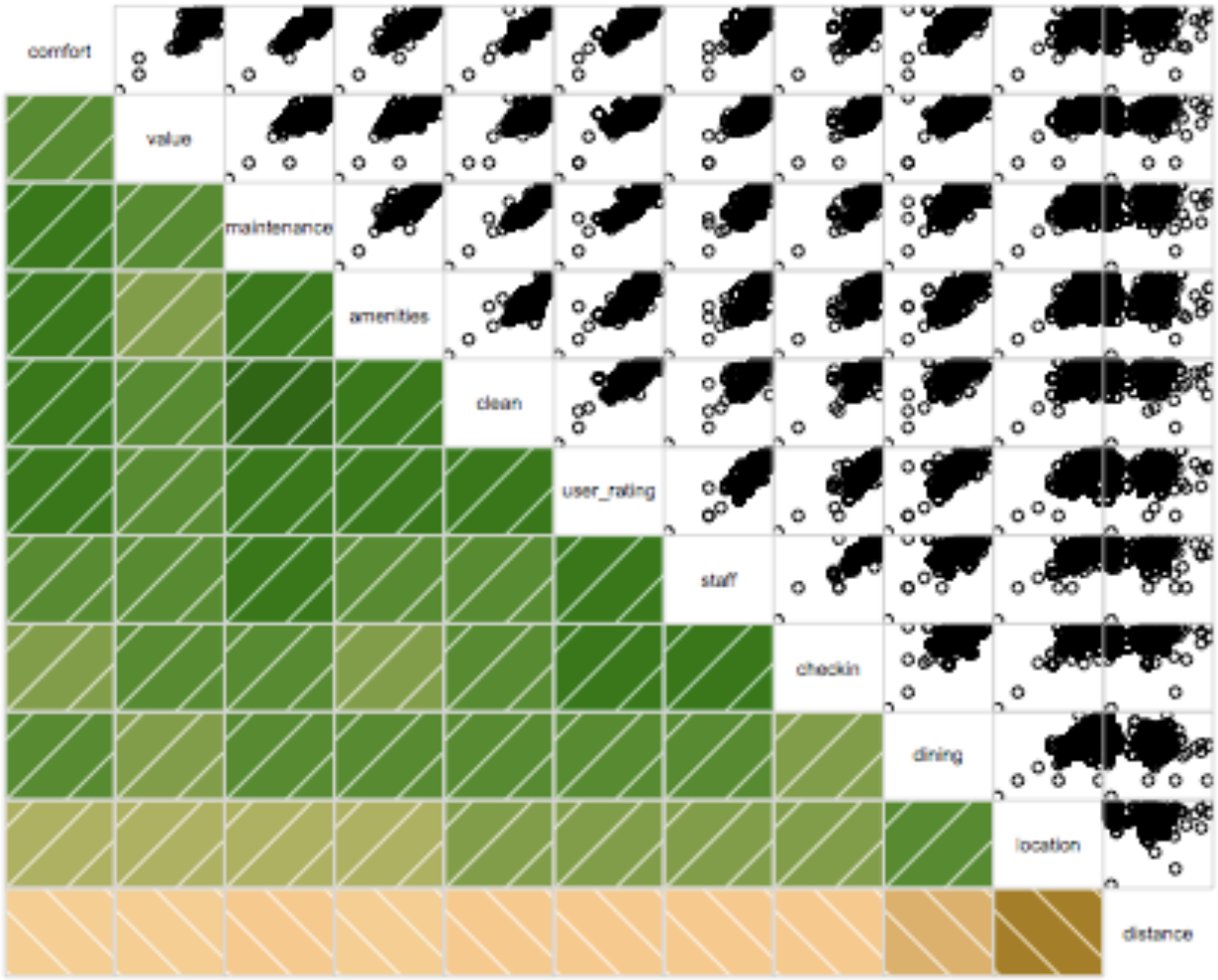


Statistical Analysis - Normality tests

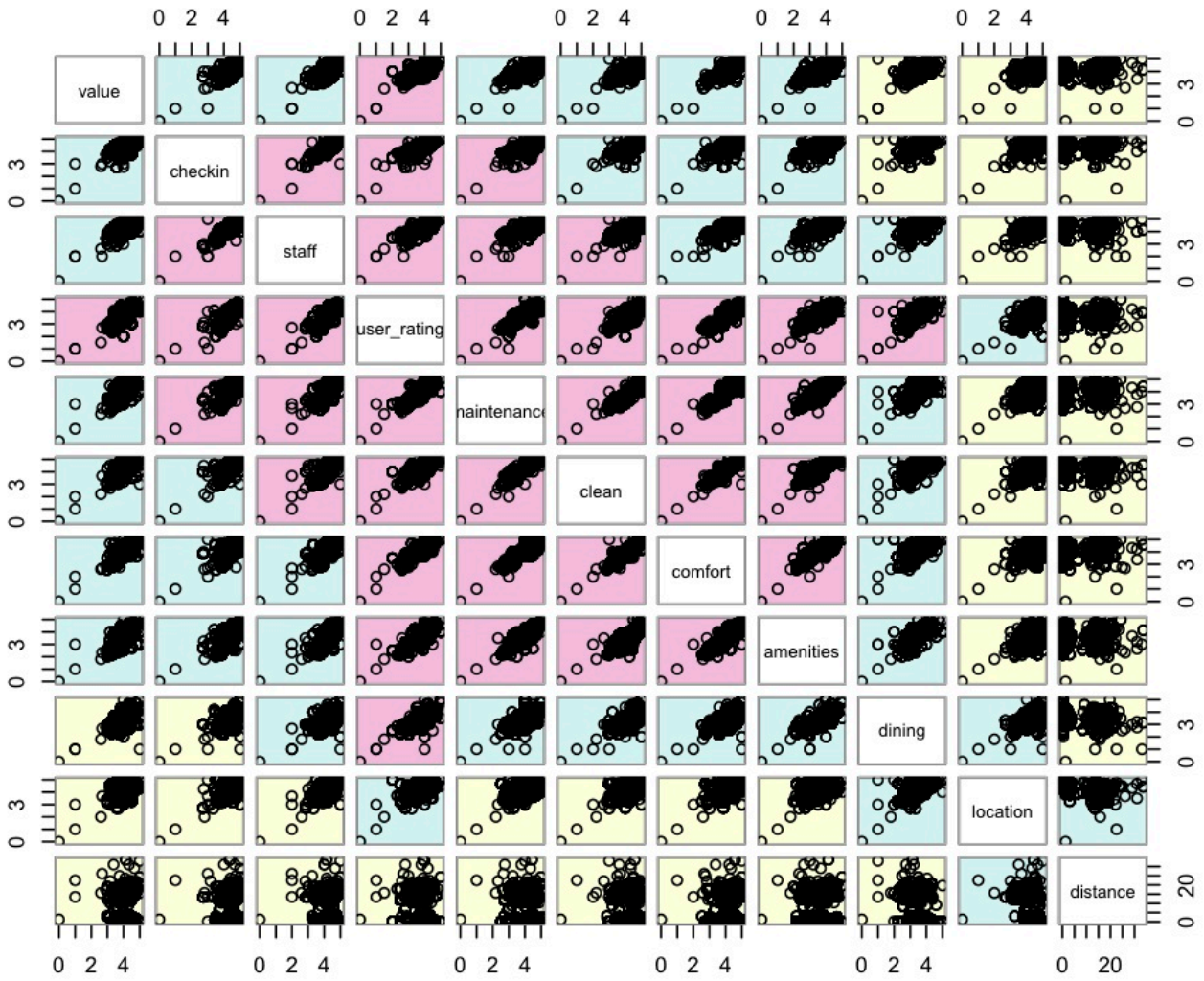
- How normal is our data?
- Plots the quantiles of the data set against the theoretical quantiles.
- If the distributions are the same, then the plot will be approximately a straight line.
- An "S" shape implies that one distribution has longer tails than the other.
- **Avoid tendency to create stories out of noise.**



Statistical Analysis - Macro level regression

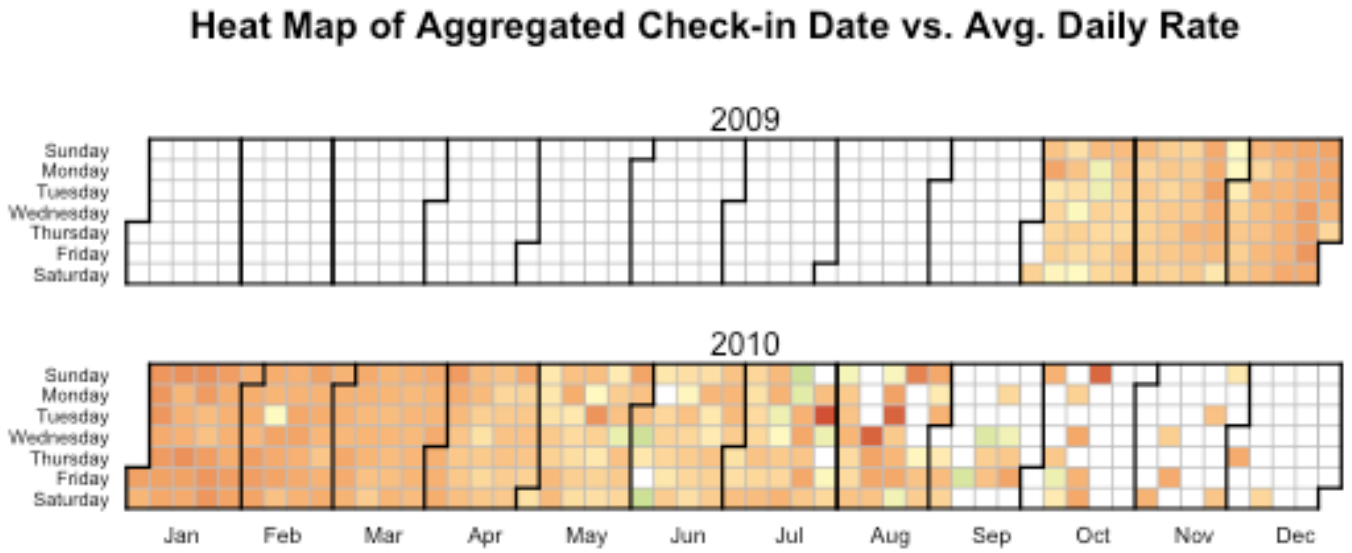


Statistical Analysis - Exploratory correlation



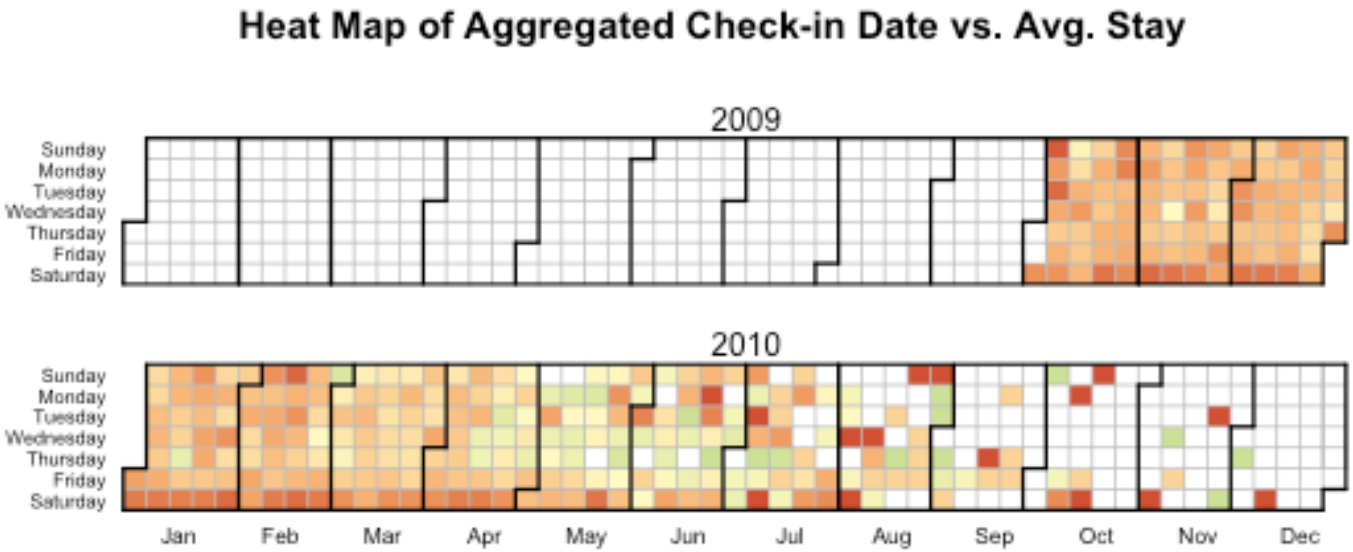
Statistical Analysis - Visual Analytics

- Show daily average rate based on booked hotels.
- Show seasonal dip in hotel rates.
- Outliers removed.
- **“Median is not the message”; Find patterns first.**



Statistical Analysis - More seasonal variations

- Customer hotel stay gets longer during summer months
- Could help in designing search based on seasons.
- Outliers removed.
- **Understand data boundaries.**

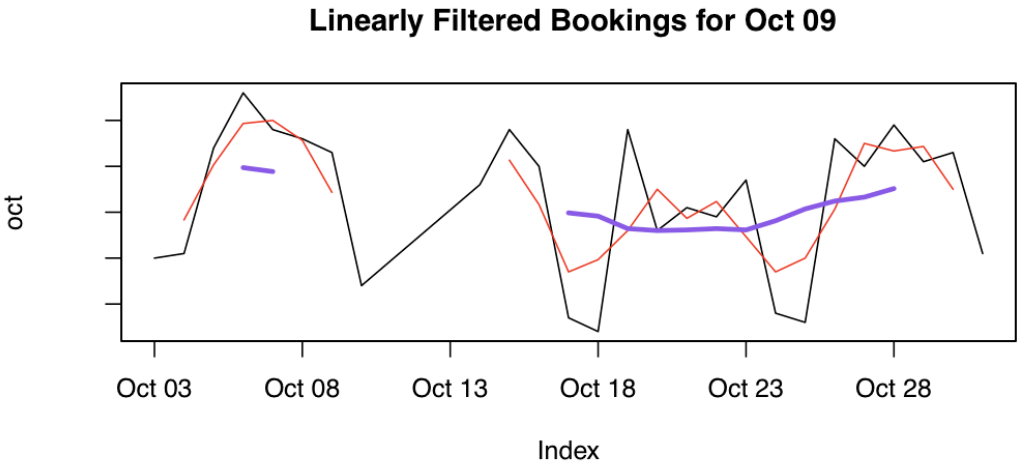


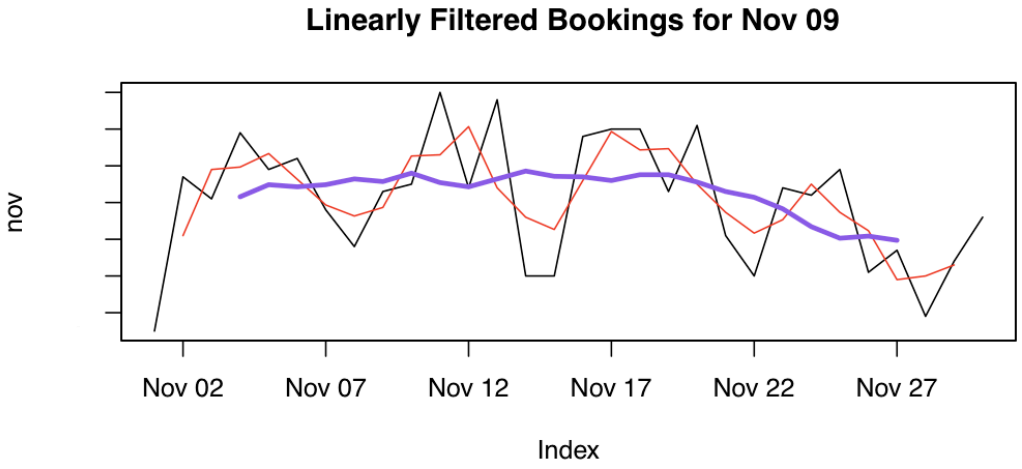
Statistical Analysis - Linear Filtering

- Decompose into a trend, a seasonal component and remainder.
- Moving average linear filters with equal weights.

$$T_t = \frac{1}{2a+1} \sum_{i=-a}^{a} X_{t+i}$$

- Filters Coefficients:
  - $a = 2 : \lambda_i = \{\frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}\}$
- **Let macro patterns guide further modeling.**





References

- Hadoop project: http://hadoop.apache.org/
- Hive project: http://hadoop.apache.org/hive/
- Hive – A Petabyte Scale Data Warehouse Using Hadoop: http://i.stanford.edu/~ragho/hive-icde2010.pdf
- Hadoop The Definitive Guide, Tom White, O'Reilly Press, 2009
- Why Model, J. Epstein, 2008
- Beautiful Data, T. Segaran & J. Hammerbacher, 2009

Contact

- Jonathan Seidman:
  - jseidman@orbitz.com
  - @jseidman
- Ramesh Venkataramaiah:
  - rvenkataramaiah@orbitz.com

Questions?