Scalable Visualization

Jeffrey Heer  University of Washington
Varieties of “big data”...
<table>
<thead>
<tr>
<th>Tall Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lots of records</td>
</tr>
<tr>
<td>Large DBs have petabytes or more</td>
</tr>
<tr>
<td>(but median DB still fits in RAM!)</td>
</tr>
</tbody>
</table>

How to manage?
- Parallel data processing
- Reduction: Filter, aggregate
- Sample or approximate

Not just about systems. Consider perceptual / cognitive scalability.
<table>
<thead>
<tr>
<th>Tall Data</th>
<th>Wide data</th>
</tr>
</thead>
</table>
|          | Lots of variables (100s-1000s…)
|          | Select relevant subset |
|          | Dimensionality reduction |
|          | Statistical methods can suggest and order related variables |
|          | Requires human judgment |
Tall Data

Wide data

Diverse data
How can we visualize and interact with billion+ record databases in real-time?
Two Challenges:

1. Effective **visual encoding**
2. Real-time **interaction**
Perceptual and interactive scalability should be limited by the chosen resolution of the visualized data, not the number of records.
1. Visualizing Large Datasets
Data

Sampling
Data

Sampling

Modeling
How to **Visualize** a Billion+ Records

Decouple the visual complexity from the raw data through aggregation.
Bin > Aggregate (> Smooth) > Plot

1. Bin  Divide data domain into discrete “buckets”

Categories: Already discrete (but watch out for high cardinality)
Numbers: Choose bin intervals (uniform, quantile, ...)
Time: Choose time unit: Hour, Day, Month, etc.
Geo: Bin x, y coordinates after cartographic projection
Bin > Aggregate (> Smooth) > Plot

1. Bin  Divide data domain into discrete “buckets”

*Categories*: Already discrete (but watch out for high cardinality)

*Numbers*: Choose bin intervals (uniform, quantile, ...)

*Time*: Choose time unit: Hour, Day, Month, etc.

*Geo*: Bin x, y coordinates after cartographic projection

2. Aggregate  Count, Sum, Average, Min, Max, ...
Bin > Aggregate (> Smooth) > Plot

1. Bin  Divide data domain into discrete “buckets”

Categories: Already discrete (but watch out for high cardinality)
Numbers: Choose bin intervals (uniform, quantile, ...)
Time: Choose time unit: Hour, Day, Month, etc.
Geo: Bin x, y coordinates after cartographic projection

2. Aggregate  Count, Sum, Average, Min, Max, ...

3. Smooth  Optional: smooth aggregates [Wickham ’13]
Bin > Aggregate (> Smooth) > Plot

1. Bin  Divide data domain into discrete “buckets”

Categories: Already discrete (but watch out for high cardinality)
Numbers: Choose bin intervals (uniform, quantile, ...)
Time: Choose time unit: Hour, Day, Month, etc.
Geo: Bin x, y coordinates after cartographic projection

2. Aggregate  Count, Sum, Average, Min, Max, ...

3. Smooth  Optional: smooth aggregates [Wickham ’13]

4. Plot  Visualize the aggregate values
# Binned Plots by Data Type

<table>
<thead>
<tr>
<th></th>
<th>Numeric</th>
<th>Ordinal</th>
<th>Temporal</th>
<th>Geographic</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1D</strong></td>
<td><strong>Histogram</strong></td>
<td><strong>Bar Chart</strong></td>
<td><strong>Line Graph/ Area Chart</strong></td>
<td><strong>Choropleth Map</strong></td>
</tr>
<tr>
<td><strong>2D</strong></td>
<td><strong>Binned Scatter Plot</strong></td>
<td><strong>Heatmap</strong></td>
<td><strong>Temporal Heatmap</strong></td>
<td><strong>Geographic Heatmap</strong></td>
</tr>
</tbody>
</table>
Design Subtleties...
Hexagonal or Rectangular Bins?

100,000 Data Points  Hexagonal Bins  Rectangular Bins

Hex bins better estimate density for 2D plots, but the *improvement is marginal* [Scott 92]. Rectangles support *reuse* and *visual queries*.
Color Scale: Discontinuity after Zero

Standard Color Ramp
Counts near zero are white.

Add Discontinuity after Zero
Counts near zero remain visible.
Examples
Example: Binned Scatter Plots

Scatterplot Matrix
Techniques for Large N
[Carr et al. ’87]
Example: Basketball Shot Chart

NBA Shooting 2011-12
[Goldsberry]
The non-normalized heatmap suffers from artifacts, seen as vertical stripes. Binned charts convey high points across the top, a collective dip in stocks during the crash of 2008, and two distinct bands of $25 and $15 stocks.
Example: Density Line Chart

[Time Series]

Repeat for each series

[Non-Normalized]
Example: Density Line Chart [Moritz & Fisher]
Example: Density Line Chart [Moritz & Fisher]

Time Series

Repeat for each series

Non-Normalized

Approx. Arc-Length Normalized
Example: Density Line Chart

Time Series

Repeat for each series

Non-Normalized

Approx. Arc-Length Normalized

Aggregate

Color

[Moritz & Fisher]
2. Enabling Real-Time Interaction
Interactive Scalability Strategies

1. Query Database
2. Client-Side Indexing / Data Cubes
3. Prefetching
4. Approximation
Interactive Scalability Strategies

1. Query Database  Offload to a scalable backend
   Tableau, for example, issues aggregation queries.
   Analytical databases are designed for fast, parallel execution.
   But round-trip queries to the DB may still be too slow…

2. Client-Side Indexing / Data Cubes

3. Prefetching

4. Approximation
Interactive Scalability Strategies

1. Query Database

2. Client-Side Indexing / Data Cubes  Query data summaries
Build sorted indices or data cubes to quickly re-calculate aggregations as needed on the client.

3. Prefetching

4. Approximation
Interactive Scalability Strategies

1. Query Database

2. Client-Side Indexing / Data Cubes

3. Prefetching  Request data *before* it is needed
Reduce latency by speculatively querying for data before it is needed. Requires prediction models to guess what is needed.

4. Approximation
Interactive Scalability Strategies

1. Query Database
2. Client-Side Indexing / Data Cubes
3. Prefetching
4. **Approximation** Give fast, approximate answers
   
   Reduce latency by computing aggregates on a sample, ideally with approximation bounds characterizing the error.
Interactive Scalability Strategies

1. Query Database
2. Client-Side Indexing / Data Cubes
3. Prefetching
4. Approximation

These strategies are not mutually exclusive! Systems can apply them in tandem.
imMens

[Liu, Jiang & Heer ‘13]

Strategies: Client-Side Data Cubes
Sampling
Google Fusion Tables
Sampling
imMens

Google Fusion Tables

Binned Aggregation
5-D Data Cube
Month, Day, Hour, X, Y
~2.3B bins
5-D Data Cube

Month, Day, Hour, X, Y

~2.3B bins
Multivariate Data Tiles

1. Send data, not pixels
2. Embed multi-dim data
Full 5-D Cube
For any pair of 1D or 2D binned plots, the maximum number of dimensions needed to support brushing & linking is **four**.
Full 5-D Cube

Σ

Σ

Σ

Σ

13 3-D Data Tiles
Full 5-D Cube $\rightarrow$ ~2.3B bins

$\sum \sum \sum \sum$

3-D cubes

3-D data tiles

13 3-D Data Tiles $\rightarrow$ ~17.6M bins (in 352KB!)
In-Memory Data Cube

~50fps querying of visual summaries of 1B data points.

5 dimensions x 50 bins/dim x 25 plots
Limitations and Questions

But where do the multivariate data tiles come from?
They must be provided by a backend server. This can be time-consuming, particularly if supporting deep levels of zooming. imMens assumes that tiles have either been pre-computed or that a backing database can suitably generate them on demand.

Does super-low-latency interaction really matter?
Is it worth it to go to all of this trouble? (Short answer: yes!)
High latency leads to reduced analytic output [Liu & Heer, InfoVis 2014]
ForeCache

[Battle, Chang, & Stonebraker ’16]

Strategies: Query Database, Prefetching
ForeCache is also a Data Tile-Based System

Manage a Cache of Tiles from DB

Example Tile-Based Views

(a) Satellite Imagery
(b) Multidimensional
(c) Time Series (Heart rate Monitoring)
1. Classify Analysis Phase

*Foraging*: Searching for patterns of interest
*Sensemaking*: Closely examine a region-of-interest (ROI)
*Navigation*: Transition between levels of detail

Train a machine learning classifier (SVM) to predict phase. The input data is the activity trace of user interactions.
Key Idea: Model & Predict User Behavior

1. Classify Analysis Phase

2. Apply Prediction Models

*Actions-Based*: Use recent interactions to predict next ones.
   You pan left twice; what is the probability you will do it again?

*Signature-Based*: Match to data characteristics of interest.
   What data tiles are visually similar to current focus tiles?

These models are weighted based on the analysis phase.
   Actions-Based for *navigation*. Signature-Based for *sensemaking*. Both applied equally for *foraging*. 
Application: MODIS Satellite Data

Analyzing snow cover in a scientific database. ROI = Region of Interest

ForeCache improves latency:

- 430% better than current non-prefetching systems
- 88% better than existing prediction methods
Falcon

[Moritz, Howe, & Heer ’19]

Strategies: Query Database, Client-Side Data Cubes, Prefetching
How does Falcon support fine-grained real-time interaction?
Falcon Interaction Log

> Brushing is more common and people are sensitive to latencies.

💡 Prioritize **brushing** latency over **view switching** latency.

5x speedup

Brushing interactions
Key Idea:

User-centered prefetching and indexing to support all brushing interactions with one view. Re-compute if the user switches the view.
brushes in the precomputed view

serves requests from a data cube

brushes in the precomputed view

serves requests from a data cube


interacts with a new view

query for new data cubes
Constant data & time. 
Client only.

💡 Aggregation decouples interactions from queries over the raw data.

Requires one pass over the data.

💡 View switches are **rare** and users are **not as latency sensitive** with them.
1.7 B stars.
1.2 TB of data.
Visualizations running in my browser.
Data stored in OmniSci database.
"With Falcon it feels like I'm really interacting with my data."

Data Platform Engineer at Stitch Fix
In Conclusion...
Two Challenges:
1. Effective **visual encoding**
2. Real-time **interaction**
Perceptual and interactive scalability should be limited by the chosen resolution of the visualized data, not the number of records.
Bin > Aggregate (> Smooth) > Plot

1. **Bin**  Divide data domain into discrete “buckets”

2. **Aggregate**  Count, Sum, Average, Min, Max, ...

3. **Smooth**  *Optional*: smooth aggregates [Wickham ’13]

4. **Plot**  Visualize the aggregate values
Interactive Scalability Strategies

1. Query Database
2. Client-Side Indexing / Data Cubes
3. Prefetching
4. Approximation

These strategies are not mutually exclusive! Systems can apply them in tandem.