CSE 512 - Data Visualization

Scalable Visualization

Leilani Battle  University of Washington
Varieties of “big data”…
Many Records

<table>
<thead>
<tr>
<th>Large DBs have petabytes or more (but median DB still fits in RAM!)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Affects system and perceptual scalability</td>
</tr>
<tr>
<td>How to manage?</td>
</tr>
<tr>
<td>Parallel data processing</td>
</tr>
<tr>
<td>Reduction: Filter, aggregate, sample</td>
</tr>
</tbody>
</table>
Many Records

Lots of variables (100s-1000s…)
Select relevant subset
Dimensionality reduction
Statistical methods can suggest and order related variables

Requires human judgment
<table>
<thead>
<tr>
<th>Many Records</th>
<th>Many Columns</th>
<th>Many Sources &amp; Structures</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Many Records  Many Columns

Many Sources & Structures

Many Updates
Many Records

Many Columns

Many Sources & Structures

Many Updates
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
How to **Visualize** a Billion+ Records

Data

Sampling

Binned Aggregation

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>1D</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></td>
<td><img src="image1" alt="Histogram" /></td>
<td><img src="image2" alt="Bar Chart" /></td>
<td><img src="image3" alt="Line Graph" /></td>
<td><img src="image4" alt="Choropleth Map" /></td>
</tr>
<tr>
<td>2D</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>
<tr>
<td></td>
<td><img src="image5" alt="Binned Scatter Plot" /></td>
<td><img src="image6" alt="Heatmap" /></td>
<td><img src="image7" alt="Temporal Heatmap" /></td>
<td><img src="image8" alt="Geographic Heatmap" /></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.
Linear interpolation in RGBA is not perceptually linear.

Perceptual color spaces approximate perceptual linearity.
Examples
Scatterplot Matrix Techniques for Large N

[Carr et al. ’87]
Example: Basketball Shot Chart

NBA Shooting 2011-12
[Goldsberry]
Example: Density Line Chart

[Image of Line Chart, Non-Normalized Heatmap, and Normalized "DenseLines"]

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

[Moritz & Fisher]
Example: Density Line Chart

[Non-Normalized Time Series]

A

Time Series

Repeat for each series

B.1

Non-Normalized

B.2

Sum:

2 2 2 1 3 2 2 2
Example: Density Line Chart

**Time Series**

- **Value** vs **Time**

**Repeat for each series**

- **Non-Normalized**

  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 0 |
  | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |
  | 1 | 1 |
  | 1 |

- **Approx. Arc-Length Normalized**

  | 0 | 0 | 0 | 0 | 0.3 | 0 | 0 | 0 | 0 |
  | 0 | 0 | 0 | 0.5 | 0.5 | 0.3 | 0.5 | 0.5 | 0.5 |
  | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 |
  | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 |
  | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 |
  | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 |
  | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 |
  | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 |
  | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 |

- **Sum:** 2 2 2 2 1 3 2 2 2 2
Example: Density Line Chart

Time Series

Repeat for each series

Non-Normalized

Approx. Arc-Length Normalized

Aggregate

Color

Example: Density Line Chart

[Moritz & Fisher]
Example: Density Line Chart

The density of the second group appears to increase to the right! Without normalization, the steep lines are overrepresented.
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**

[Battle & Scheidegger 2020]
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

[Battle & Scheidegger 2020]
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

[Battle & Scheidegger 2020]
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.

[Battle & Scheidegger 2020]
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.

[Battle & Scheidegger 2020]
imMens

[Liu, Jiang & Heer ’13]

Strategies: Client-Side Data Cubes
Sampling
Google Fusion Tables
Binned Aggregation

Sampling

Google Fusion Tables
Binned Aggregation
imMens
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

3-D cubes

3-D data tiles

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

3-D cubes $\rightarrow$ ~17.6M bins (in 352KB!)

13 3-D Data Tiles
Pre-compute tiles & send from server.
Bind data tiles as image textures.
Query & Render on GPU (WebGL)

Compute aggregation for each output bin. Executes in parallel on GPU.
Accumulate results in offscreen buffer. Render resulting plots in second pass.
Performance Benchmarks

Simulate interaction: brushing & linking across binned plots.
- 4x4 and 5x5 plots
- 10 to 50 bins

Measure time from selection to render.

Test setup:
2.3 GHz MacBook Pro
NVIDIA GeForce GT 650M
Google Chrome v.23.0
~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]
Administrivia
A2: Deceptive Visualization

Design two static visualizations for a dataset:
1. An earnest visualization that faithfully conveys the data
2. A deceptive visualization that tries to mislead viewers

Your two visualizations may address different questions.

Try to design a deceptive visualization that appears to be earnest: *can you trick your classmates and course staff?*

You are free to choose your own dataset, but we have also provided some preselected datasets for you.

Submit two images and a brief write-up on Gradescope.

Due by **Fri 4/22 11:59pm.**
A2 Peer Reviews

On Thursday 4/21-Monday 4/25 you will be assigned two peer A2 submissions to review. For each:

• Try to determine which is earnest and which is deceptive
• Share a rationale for how you made this determination
• Share feedback using the “I Like / I Wish / What If” rubric

Assigned reviews will be posted on the A2 Peer Review page on Canvas, along with a link to a Google Form. You should submit two forms: one for each A2 peer review.

Due by **Fri 4/29 11:59pm**.
I Like… / I Wish… / What If?

I LIKE...
Praise for design ideas and/or well-executed implementation details. Example: "I like the navigation through time via the slider; the patterns observed as one moves forward are compelling!"

I WISH...
Constructive statements on how the design might be improved or further refined. Example: "I wish moving the slider caused the visualization to update immediately, rather than the current lag."

WHAT IF?
Suggest alternative design directions, or even wacky half-baked ideas. Example: "What if we got rid of the slider and enabled direct manipulation navigation by dragging data points directly?"
Two Tutorials Next Week

Both tutorials will be led by Vishal and Philip and will be recorded.

**D3.js Deep Dive**: Thursday 4/28 during lecture

**Web Publishing**: Friday 4/29 at 1pm on Zoom
Break Time!
How does **interactive latency** affect exploratory analysis with visualizations?

[Liu & Heer ‘14]
Prior Work - Negatives to Latency

Higher latency entails higher action costs, subjects satisfice by selecting strategies that reduce short-term effort with no guarantee that the final outcome is optimized. [Gray & Boehm-Davis]

300ms latency reduces the number of Google searches; effect persists for days. [Brutlag et al]

When the cost of acquiring information is increased, subjects change strategy and rely more on working memory. [Ballard et al]
Prior Work – Positives to Latency

When confronted with increased latencies, users resort to more mental planning, at times making fewer errors and performing better on tasks with verifiable outcomes. [O’Hara & Payne]
Prior Work - Positives to Latency

When confronted with increased latencies, users resort to more mental planning, at times making fewer errors and performing better on tasks with verifiable outcomes. [O’Hara & Payne]

Experiment Design

2 (Latency) x 2 (Scenario) Design

Latency: +0ms / +500ms

Scenario: Mobile Check-ins / FAA Flight Delays

Exploratory Analysis Tasks (2 per session)

* imMens with brush, pan, zoom, adjust scales
* Users asked to explore data and share findings
* Log events, record audio and screen capture

16 subjects, all familiar with data analysis + vis
4.5m Mobile Check-Ins
140m FAA Flight Delay Records
Data Collection & Analysis

Event Log Analysis
Analyze triggered & processed user input events
Assess data set coverage (# unique tiles)

Verbal Protocol Analysis
Think-aloud protocol: verbalize thought process
Transcribe sessions; Code actions and insights
Analyze number and type of coded events
Latency Study Results

Higher latency leads to...
Reduced user activity and data set coverage
Less observation, generalization & hypothesis
Latency Study Results

Higher latency leads to...
Reduced user activity and data set coverage
Less observation, generalization & hypothesis

Different interactions exhibit varied sensitivity to latency. Brushing is highly sensitive!
Latency Study Results

Higher latency leads to...
Reduced user activity and data set coverage
Less observation, generalization & hypothesis

Different interactions exhibit varied sensitivity to latency. Brushing is highly sensitive!
**In short: milliseconds matter!** And imMens was not a waste of time… 😅
ForeCache

[Battle, Chang, & Stonebraker ‘16]

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

Manage a Cache of Tiles from DBMS

Example Tile-Based Views

(a) Satellite Imagery
(b) Multidimensional
(c) Timeseries (Heart rate Monitoring)
Key Idea: Model & Predict User Behavior

1. Classify the User’s Analysis Phase
   - Foraging: Searching for patterns of interest
   - Sensemaking: Closely examine a region-of-interest (ROI)
   - Navigation: Transition between levels of detail

2. Predict Which Data Tiles Will be Requested
   Train a machine learning classifier (SVM) to predict phase. The input data is the activity trace of user interactions.
Foraging and Sensemaking

External Data Sources → Evidence File → Hypotheses → Sensemaking → Foraging → Evidence File → Hypotheses → Sensemaking → Foraging → Evidence File

[Pirolli & Card 2005]
Foraging and Sensemaking

External Data Sources → Evidence File → Hypotheses → Sensemaking → Foraging

[Pirolli & Card 2005]
Foraging and Sensemaking

Foraging

External Data Sources

Evidence File

Hypotheses

Sensemaking

[Pirolli & Card 2005]
Adding a “Navigation” Phase
Applying the Three Phases to Exploration Scenarios

Snow

Foraging

No Snow
Applying the Three Phases to Exploration Scenarios

Navigation

User zooms in
Applying the Three Phases to Exploration Scenarios
Applying the Three Phases to Exploration Scenarios

Navigation

User zooms out
Applying the Three Phases to Exploration Scenarios

Snow

Foraging

No Snow
Using Phases to Predict Tiles

Phase Predictor

Model Manager

Model 1

Model 2
Using Phases to Predict Tiles

Phase Predictor

Model Manager

Model 1

Model 2

"Pan"
Using Phases to Predict Tiles
Using Phases to Predict Tiles
Using Phases to Predict Tiles

Phase Predictor

Model Manager

Model 1

Model 2

\( T_A, T_B \)

\( T_C, T_D, T_E, T_F \)
Using Phases to Predict Tiles

Phase Predictor

Model Manager

Model 1

Model 2

\[ T_A, T_B, T_C, T_D, T_E, T_F \]
Using Phases to Predict Tiles
Action-Based Tile Recommendations

Idea: user consistently moves in predictable directions
Signature-Based Tile Recommendations

Idea: user wants to see more of the same thing
Signature-Based Tile Recommendations

Idea: user wants to see more of the same thing
Signature-Based Tile Recommendations

Idea: user wants to see more of the same thing
Signature-Based Tile Recommendations

Idea: user wants to see more of the same thing
Evaluating ForeCache: A User Study

Participants: 18 earth science researchers
Explored NASA MODIS snow cover queries
Retrospective Performance Experiments

Compared response times and prediction accuracy to a non-prefetching baseline and two existing pre-fetching methods:

[Doshi et al. 2003]
Results: ForeCache was 20% More Accurate and 88% Faster than Existing Pre-fetching Methods
Falcon

[Moritz, Howe, & Heer ‘19]

Strategies: Query Database, Client-Side Data Cubes, Prefetching
0.008% of the data
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
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.


- brushes in the precomputed view
- serves requests from a data cube
- interacts with a new view
- query for new data cubes
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
What if data is too large to query in a reasonable time?
Trust, but Verify: Optimistic Vis

[Moritz, Fisher, Ding & Wang ’17]

Strategies: Query Database, Approximation
Latencies reduce engagement and lead to fewer observations.

Approximation: Trade Accuracy for Speed

Approximate query processing (AQP)
Uncertainty estimation in statistics
Uncertainty visualization
Probabilistic programming
Approximate hardware
Pick your poison:
1. Trust the approximation, or
2. Wait for everything to complete.
Optimistic Visualization

Trust but Verify
What if we think of the issues with approximation as user experience problems?
Optimistic Visualization

1. Analysts uses initial estimates.
2. Precise queries run in the background.

Analysts can use approximations and also trust them.

Pangloss Implements Optimistic Visualization
Pangloss Visualizes Uncertainty
Pangloss shows a History of Previous Charts
In Pangloss, Analysts can Confirm results
Evaluation

Case studies with teams at Microsoft who brought in their own data.

Approximation works
“seeing something right away at first glimpse is really great”

Need for guarantees
“[with a competitor] I was willing to wait 70-80 seconds. It wasn’t ideally interactive, but it meant I was looking at all the data.”

Optimism works
“I was thinking what to do next— and I saw that it had loaded, so I went back and checked it . . . [the passive update is] very nice for not interrupting your workflow.”
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.