Data and Image Models

Jeffrey Heer  University of Washington
The Big Picture

task
questions, goals
assumptions

data
physical data type
conceptual data type

domain
metadata
semantics
conventions

processing
algorithms

mapping
visual encoding

image
visual channel
graphical marks
Topics

Properties of Data
Properties of Images
Mapping Data to Images
Data Models
Data Models / Conceptual Models

Data models are formal descriptions
Math: sets with operations on them
Example: integers with + and x operators

Conceptual models are mental constructions
Include semantics and support reasoning

Examples (data vs. conceptual)
1D floats vs. temperatures
3D vector of floats vs. spatial location
Taxonomy of Data Types (?)

1D (sets and sequences)
Temporal
2D (maps)
3D (shapes)
nD (relational)
Trees (hierarchies)
Networks (graphs)

Are there others?

The eyes have it: A task by data type taxonomy for information visualization
[Shneiderman 96]
Nominal, Ordinal & Quantitative
Nominal, Ordinal & Quantitative

N - Nominal (labels or categories)
  - Fruits: apples, oranges, ...
Nominal, Ordinal & Quantitative

N - Nominal (labels or categories)
  - Fruits: apples, oranges, ...

O - Ordered
  - Quality of meat: Grade A, AA, AAA
Nominal, Ordinal & Quantitative

N - Nominal (labels or categories)
  - Fruits: apples, oranges, ...

O - Ordered
  - Quality of meat: Grade A, AA, AAA

Q - Interval (location of zero arbitrary)
  - Dates: Jan, 19, 2006; Location: (LAT 33.98, LONG -118.45)
  - Only differences (i.e., intervals) may be compared
Nominal, Ordinal & Quantitative

N - Nominal (labels or categories)
  - Fruits: apples, oranges, ...

O - Ordered
  - Quality of meat: Grade A, AA, AAA

Q - Interval (location of zero arbitrary)
  - Dates: Jan, 19, 2006; Location: (LAT 33.98, LONG -118.45)
  - Only differences (i.e., intervals) may be compared

Q - Ratio (zero fixed)
  - Physical measurement: Length, Mass, Time duration, ...
  - Counts and amounts
Nominal, Ordinal & Quantitative

N - Nominal (labels or categories)
  • Operations: =, ≠

O - Ordered
  • Operations: =, ≠, <, >

Q - Interval (location of zero arbitrary)
  • Operations: =, ≠, <, >, -
  • Can measure distances or spans

Q - Ratio (zero fixed)
  • Operations: =, ≠, <, >, -, %
  • Can measure ratios or proportions
From Data Model to N, O, Q

Data Model
32.5, 54.0, -17.3, ...
Floating point numbers

Conceptual Model
Temperature (°C)

Data Type
Burned vs. Not-Burned (N)
Hot, Warm, Cold (O)
Temperature Value (Q-interval)
Dimensions & Measures

**Dimensions** (~ independent variables)  
Often discrete variables describing data (N, O)  
Categories, dates, binned quantities

**Measures** (~ dependent variables)  
Data values that can be aggregated (Q)  
Numbers to be analyzed  
Aggregate as sum, count, avg, std. dev…

Not a strict distinction. The same variable may be treated either way depending on the task.
Example: U.S. Census Data
Example: U.S. Census Data

People Count: # of people in group
Year: 1850 - 2000 (every decade)
Age: 0 – 90+
Sex: Male, Female
Marital Status: Single, Married, Divorced, ...
### Example: U.S. Census

<table>
<thead>
<tr>
<th>People Count</th>
<th>Year</th>
<th>Age</th>
<th>Sex</th>
<th>Marital Status</th>
<th>People</th>
</tr>
</thead>
<tbody>
<tr>
<td>2,348</td>
<td>1850</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1,483,789</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>1,411,087</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>1,359,669</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>10</td>
<td>0</td>
<td>1</td>
<td>1,260,096</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>15</td>
<td>0</td>
<td>0</td>
<td>1,216,112</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>15</td>
<td>0</td>
<td>1</td>
<td>1,177,123</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>20</td>
<td>0</td>
<td>0</td>
<td>1,110,619</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>20</td>
<td>0</td>
<td>1</td>
<td>1,017,281</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>25</td>
<td>0</td>
<td>0</td>
<td>1,003,841</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>25</td>
<td>0</td>
<td>0</td>
<td>862,547</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>30</td>
<td>0</td>
<td>0</td>
<td>799,482</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>30</td>
<td>0</td>
<td>1</td>
<td>730,088</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>35</td>
<td>0</td>
<td>0</td>
<td>639,963</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>35</td>
<td>0</td>
<td>1</td>
<td>588,487</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>40</td>
<td>0</td>
<td>0</td>
<td>50,5012</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>40</td>
<td>0</td>
<td>1</td>
<td>475,911</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>45</td>
<td>0</td>
<td>0</td>
<td>428,185</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>45</td>
<td>0</td>
<td>1</td>
<td>384,211</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>50</td>
<td>0</td>
<td>0</td>
<td>341,254</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>50</td>
<td>0</td>
<td>1</td>
<td>321,343</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>55</td>
<td>0</td>
<td>0</td>
<td>2,865,800</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>55</td>
<td>0</td>
<td>1</td>
<td>1,940,800</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>60</td>
<td>0</td>
<td>0</td>
<td>1,872,080</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>60</td>
<td>0</td>
<td>1</td>
<td>1,749,760</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>65</td>
<td>0</td>
<td>0</td>
<td>1,622,263</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>65</td>
<td>0</td>
<td>1</td>
<td>1,068,276</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>70</td>
<td>0</td>
<td>0</td>
<td>1,055,544</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>70</td>
<td>0</td>
<td>1</td>
<td>736,777</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>75</td>
<td>0</td>
<td>0</td>
<td>717,621</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>75</td>
<td>0</td>
<td>1</td>
<td>40,0834</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>80</td>
<td>0</td>
<td>0</td>
<td>40,2299</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>80</td>
<td>0</td>
<td>1</td>
<td>23,4499</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>85</td>
<td>0</td>
<td>0</td>
<td>22,9499</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>85</td>
<td>0</td>
<td>1</td>
<td>8,186</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>90</td>
<td>0</td>
<td>0</td>
<td>8,186</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>90</td>
<td>0</td>
<td>1</td>
<td>5,259</td>
</tr>
<tr>
<td></td>
<td>1850</td>
<td>90</td>
<td>0</td>
<td>2</td>
<td>65,69</td>
</tr>
<tr>
<td></td>
<td>1860</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>21,208,45</td>
</tr>
<tr>
<td></td>
<td>1860</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>20,921,63</td>
</tr>
</tbody>
</table>

2,348 data points
<table>
<thead>
<tr>
<th>People Count</th>
<th>Q-Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>Year</td>
<td>Q-Interval (O)</td>
</tr>
<tr>
<td>Age</td>
<td>Q-Ratio (O)</td>
</tr>
<tr>
<td>Sex</td>
<td>N</td>
</tr>
<tr>
<td>Marital Status</td>
<td>N</td>
</tr>
</tbody>
</table>
Census: Dimension or Measure?

<table>
<thead>
<tr>
<th>People Count</th>
<th>Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Year</td>
<td>Dimension</td>
</tr>
<tr>
<td>Age</td>
<td>Depends!</td>
</tr>
<tr>
<td>Sex</td>
<td>Dimension</td>
</tr>
<tr>
<td>Marital Status</td>
<td>Dimension</td>
</tr>
</tbody>
</table>
Census Data Demo
Data Tables & Transformations
Relational Data Model

Represent data as a **table** (or relation)

Each **row** (or **tuple**) represents a record
   Each record is a fixed-length tuple

Each **column** (or **field**) represents a variable
   Each field has a **name** and a **data type**

A table’s **schema** is the set of names and types

A **database** is a collection of tables (relations)
Relational Algebra \[\text{[Codd '70]} / \text{SQL}\]

Operations on Data Tables: table(s) in, table out
Relational Algebra [Codd ’70] / SQL

Operations on Data Tables: table(s) in, table out
Project (select): select a set of columns
Filter (where): remove unwanted rows
Sort (order by): order records
Aggregate (group by, sum, min, max, …):
  partition rows into groups + summarize
Combine (join, union, …):
  integrate data from multiple tables
Relational Algebra \[\text{[Codd '70]} \ / \ SQL\]

**Project (select)**: select a set of columns

*select day, stock*

<table>
<thead>
<tr>
<th>day</th>
<th>stock</th>
<th>price</th>
</tr>
</thead>
<tbody>
<tr>
<td>10/3</td>
<td>AMZN</td>
<td>957.10</td>
</tr>
<tr>
<td>10/3</td>
<td>MSFT</td>
<td>74.26</td>
</tr>
<tr>
<td>10/4</td>
<td>AMZN</td>
<td>965.45</td>
</tr>
<tr>
<td>10/4</td>
<td>MSFT</td>
<td>74.69</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>day</th>
<th>stock</th>
</tr>
</thead>
<tbody>
<tr>
<td>10/3</td>
<td>AMZN</td>
</tr>
<tr>
<td>10/3</td>
<td>MSFT</td>
</tr>
<tr>
<td>10/4</td>
<td>AMZN</td>
</tr>
<tr>
<td>10/4</td>
<td>MSFT</td>
</tr>
</tbody>
</table>
Relational Algebra [Codd ’70] / SQL

Filter (where): remove unwanted rows

select * where price > 100

<table>
<thead>
<tr>
<th>day</th>
<th>stock</th>
<th>price</th>
</tr>
</thead>
<tbody>
<tr>
<td>10/3</td>
<td>AMZN</td>
<td>957.10</td>
</tr>
<tr>
<td>10/3</td>
<td>MSFT</td>
<td>74.26</td>
</tr>
<tr>
<td>10/4</td>
<td>AMZN</td>
<td>965.45</td>
</tr>
<tr>
<td>10/4</td>
<td>MSFT</td>
<td>74.69</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>day</th>
<th>stock</th>
<th>price</th>
</tr>
</thead>
<tbody>
<tr>
<td>10/3</td>
<td>AMZN</td>
<td>957.10</td>
</tr>
<tr>
<td>10/4</td>
<td>AMZN</td>
<td>965.45</td>
</tr>
</tbody>
</table>
Relational Algebra [Codd ’70] / SQL

Sort (order by): order records
select * order by stock, day

<table>
<thead>
<tr>
<th>day</th>
<th>stock</th>
<th>price</th>
</tr>
</thead>
<tbody>
<tr>
<td>10/3</td>
<td>AMZN</td>
<td>957.10</td>
</tr>
<tr>
<td>10/3</td>
<td>MSFT</td>
<td>74.26</td>
</tr>
<tr>
<td>10/4</td>
<td>AMZN</td>
<td>965.45</td>
</tr>
<tr>
<td>10/4</td>
<td>MSFT</td>
<td>74.69</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>day</th>
<th>stock</th>
<th>price</th>
</tr>
</thead>
<tbody>
<tr>
<td>10/3</td>
<td>AMZN</td>
<td>957.10</td>
</tr>
<tr>
<td>10/4</td>
<td>AMZN</td>
<td>965.45</td>
</tr>
<tr>
<td>10/3</td>
<td>MSFT</td>
<td>74.26</td>
</tr>
<tr>
<td>10/4</td>
<td>MSFT</td>
<td>74.69</td>
</tr>
</tbody>
</table>
Relational Algebra [Codd ’70] / SQL

Aggregate (group by, sum, min, max, ...):
select stock, min(price) group by stock

<table>
<thead>
<tr>
<th>day</th>
<th>stock</th>
<th>price</th>
</tr>
</thead>
<tbody>
<tr>
<td>10/3</td>
<td>AMZN</td>
<td>957.10</td>
</tr>
<tr>
<td>10/3</td>
<td>MSFT</td>
<td>74.26</td>
</tr>
<tr>
<td>10/4</td>
<td>AMZN</td>
<td>965.45</td>
</tr>
<tr>
<td>10/4</td>
<td>MSFT</td>
<td>74.69</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>stock</th>
<th>min(price)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMZN</td>
<td>957.10</td>
</tr>
<tr>
<td>MSFT</td>
<td>74.26</td>
</tr>
</tbody>
</table>
Roll-Up and Drill-Down

Want to examine population by year and age? **Roll-up** the data along the desired dimensions

```
SELECT year, age, sum(people)
FROM census
GROUP BY year, age
```
Roll-Up and Drill-Down

Want to see the breakdown by marital status? **Drill-down** into additional dimensions

```
SELECT year, age, marst, sum(people)
FROM census
GROUP BY year, age, marst
```
A two dimensional projection.

Marital Status

Year

1970
1980
1990
2000

Age

0-19
20-39
40-59
60+

Sum along Marital Status

Sum along Age

All Marital Status

All Ages

All Years

Sum along Year
<table>
<thead>
<tr>
<th>YEAR</th>
<th>AGE</th>
<th>MARST</th>
<th>SEX</th>
<th>PEOPLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1850</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1,483,789</td>
</tr>
<tr>
<td>1850</td>
<td>5</td>
<td>0</td>
<td>1</td>
<td>1,411,067</td>
</tr>
<tr>
<td>1860</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2,120,846</td>
</tr>
<tr>
<td>1860</td>
<td>5</td>
<td>0</td>
<td>1</td>
<td>1,804,467</td>
</tr>
</tbody>
</table>

Which format might we prefer? Why?
How do rows, columns, and tables match up with observations, variables, and types? In “tidy” data:
1. Each variable forms a column.
2. Each observation forms a row.
3. Each type of observational unit forms a table.

The advantage is that this provides a flexible starting point for analysis, transformation, and visualization.

Our pivoted table variant was not “tidy”!

(This is a variant of normalized forms in DB theory)
Common Data Formats

CSV: Comma-Separated Values (d3.csv)

year, age, marst, sex, people
1850, 0, 0, 1, 1483789
1850, 5, 0, 1, 1411067
...

...
Common Data Formats

CSV: Comma-Separated Values (d3.csv)

```
year,age,marst,sex,people
1850,0,0,1,1483789
1850,5,0,1,1411067
...
```

JSON: JavaScript Object Notation (d3.json)

```
[
  {
    "year":1850,"age":0,"marst":0,"sex":1,"people":1483789},
  {
    "year":1850,"age":5,"marst":0,"sex":1,"people":1411067},
  ...
]
```
Administrivia
A1: Visualization Design

Design a static visualization for a data set.

The climate of a place can have a tremendous impact on people's lived experience. You will examine average monthly climate measurements for six major U.S. cities, roughly covering the edges of the continental United States.

You must choose the message you want to convey. What question(s) do you want to answer? What insight do you want to communicate?
A1: Visualization Design

Pick a **guiding question**, use it to title your vis.
Design a **static visualization** for that question.
You are free to **use any tools** (inc. pen & paper).

**Deliverables** (upload via Canvas; see A1 page)
Image of your visualization (PNG or JPG format)
Short description + design rationale (≤ 4 paragraphs)

Due by **11:59 pm, Wednesday April 7**.
Course Participation

Quiz & discussion comments on class forum. Both are due by Friday, 11:59pm.

Quizzes will start next week.

One comment per week, ending week 8.

You have 1 “pass” (quiz + comment) for the quarter.
Image Models
Visual Language is a Sign System

Images perceived as a set of signs
Sender encodes information in signs
Receiver decodes information from signs

Jacques Bertin
Sémiologie Graphique, 1967
Bertin’s Semiology of Graphics

1. A, B, C are distinguishable
2. B is between A and C.
3. BC is twice as long as AB.

∴ Encode quantitative variables

"Resemblance, order and proportional are the three signfields in graphics.“ - Bertin
### LES VARIABLES DE L'IMAGE

<table>
<thead>
<tr>
<th>XY</th>
<th>POINTS</th>
<th>LIGNES</th>
<th>ZONES</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 DIMENSIONS DU PLAN</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Z

<table>
<thead>
<tr>
<th>TAILLE</th>
<th>VALEUR</th>
</tr>
</thead>
</table>

#### LES VARIABLES DE SÉPARATION DES IMAGES

<table>
<thead>
<tr>
<th>GRAIN</th>
<th>COULEUR</th>
<th>Orientation</th>
<th>FORME</th>
</tr>
</thead>
</table>

![Diagram with examples of individual cell configurations for each category (points, lines, zones, tail, value, grain, color, orientation, and form).]
**Visual Encoding Variables**

- Position (x 2)
- Size
- Value
- Texture
- Color
- Orientation
- Shape

---

**LES VARIABLES DE L'IMAGE**

<table>
<thead>
<tr>
<th>XY 2 Dimensions Du Plan</th>
<th>Points</th>
<th>Lignes</th>
<th>Zones</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><img src="image" alt="Points" /></td>
<td><img src="image" alt="Lignes" /></td>
<td><img src="image" alt="Zones" /></td>
</tr>
</tbody>
</table>

**LES VARIABLES DE SÉPARATION DES IMAGES**

<table>
<thead>
<tr>
<th>Grain</th>
<th>Couleur</th>
<th>Orientation</th>
<th>Forme</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Grain" /></td>
<td><img src="image" alt="Couleur" /></td>
<td><img src="image" alt="Orientation" /></td>
<td><img src="image" alt="Forme" /></td>
</tr>
</tbody>
</table>
Visual Encoding Variables

<table>
<thead>
<tr>
<th>Position</th>
<th>Length</th>
<th>Area</th>
<th>Volume</th>
<th>Value</th>
<th>Texture</th>
<th>Color</th>
<th>Orientation</th>
<th>Shape</th>
<th>Transparency</th>
<th>Blur / Focus</th>
</tr>
</thead>
</table>

![Diagram of visual encoding variables](image-url)
Information in Hue and Value

Value is perceived as ordered

- Encode ordinal variables (O)

- Encode continuous variables (Q) [not as well]

Hue is normally perceived as unordered

- Encode nominal variables (N) using color
Bertin’s Levels of Organization

<table>
<thead>
<tr>
<th>Position</th>
<th>N</th>
<th>O</th>
<th>Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>N</td>
<td>O</td>
<td>Q</td>
</tr>
<tr>
<td>Value</td>
<td>N</td>
<td>O</td>
<td>Q</td>
</tr>
<tr>
<td>Texture</td>
<td>N</td>
<td>O</td>
<td></td>
</tr>
<tr>
<td>Color</td>
<td>N</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Orientation</td>
<td>N</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Shape</td>
<td>N</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note: $Q \subset O \subset N$
Deconstructions
Exports and Imports to and from DENMARK & NORWAY from 1700 to 1780.

BALANCE in FAVOUR of ENGLAND.
William Playfair, 1786

**X-axis:** year (Q)  
**Y-axis:** currency (Q)  
**Color:** imports/exports (N, O)
Wattenberg’s Map of the Market

Rectangle Area: market cap (Q)
Rectangle Position: market sector (N), market cap (Q)
Color Hue: loss vs. gain (N, O)
Color Value: magnitude of loss or gain (Q)
Single-Axis Composition
Mark Composition

Y-axis: temperature (Q)

X-axis: longitude (Q) / time (O)

Temp over space/time (Q x Q)
Mark Composition

Y-axis: latitude (Q)

X-axis: longitude (Q)

Width: army size (Q)

Army position (Q x Q) and army size (Q)
Depicts at least 5 quantitative variables. Any others?
Formalizing Design
Choosing Visual Encodings

Assume $k$ visual encodings and $n$ data attributes. We would like to pick the “best” encoding among a combinatorial set of possibilities of size $(n+1)^k$.

**Principle of Consistency**
The properties of the image (visual variables) should match the properties of the data.

**Principle of Importance Ordering**
Encode the most important information in the most effective way.
Design Criteria [Mackinlay 86]

**Expressiveness**
A set of facts is expressible in a visual language if the sentences (i.e. the visualizations) in the language express all the facts in the set of data, and only the facts in the data.

**Effectiveness**
A visualization is more effective than another visualization if the information conveyed by one visualization is more readily perceived than the information in the other visualization.
Design Criteria [Mackinlay 86]

Expressiveness
A set of facts is *expressible* in a visual language if the sentences (i.e. the visualizations) in the language express all the facts in the set of data, and only the facts in the data.

Effectiveness
A visualization is more *effective* than another visualization if the information conveyed by one visualization is more readily perceived than the information in the other visualization.
A multivariate relation may be *inexpressive* in a single horizontal dot plot because multiple records are mapped to the same position.

**Can not express the facts**
Expresses facts not in the data

A length is interpreted as a quantitative value.
Design Criteria [Mackinlay 86]

Expressiveness
A set of facts is expressible in a visual language if the sentences (i.e. the visualizations) in the language express all the facts in the set of data, and only the facts in the data.

Effectiveness
A visualization is more effective than another visualization if the information conveyed by one visualization is more readily perceived than the information in the other visualization.
Design Criteria [Mackinlay 86]

Expressiveness
A set of facts is expressible in a visual language if the sentences (i.e. the visualizations) in the language express all the facts in the set of data, and only the facts in the data.

Effectiveness
A visualization is more effective than another visualization if the information conveyed by one visualization is more readily perceived than the information in the other visualization.
Tell the truth and nothing but the truth (don’t lie, and don’t lie by omission)

Use encodings that people decode better (where better \(\approx\) faster and/or more accurate)
Mackinlay’s Ranking

Conjectured effectiveness of encodings by data type
Mackinlay’s Design Algorithm

APT - “A Presentation Tool”, 1986

User formally specifies data model and type
Input: ordered list of data variables to show

APT searches over design space
Test expressiveness of each visual encoding
Generate encodings that pass test
Rank by perceptual effectiveness criteria

Output the “most effective” visualization
APT

Automatically generate chart for car data

Input variables:
1. Price
2. Mileage
3. Repair
4. Weight
Limitations of APT?
Limitations of APT

Does not cover many visualization techniques
Networks, hierarchies, maps, diagrams
Also: 3D structure, animation, illustration, …

Does not consider interaction

Does not consider semantics / conventions

Assumes single visualization as output

Still an active area of research, e.g., the Draco visualization design knowledge base
Summary: Data & Image Models

Formal specification
Data model: relational data; N,O,Q types
Image model: visual encoding channels
Encodings map data to visual variables

Choose expressive and effective encodings
Rule-based tests of expressiveness
Perceptual effectiveness rankings

Question: how do we establish effectiveness criteria? Subject of perception lectures…