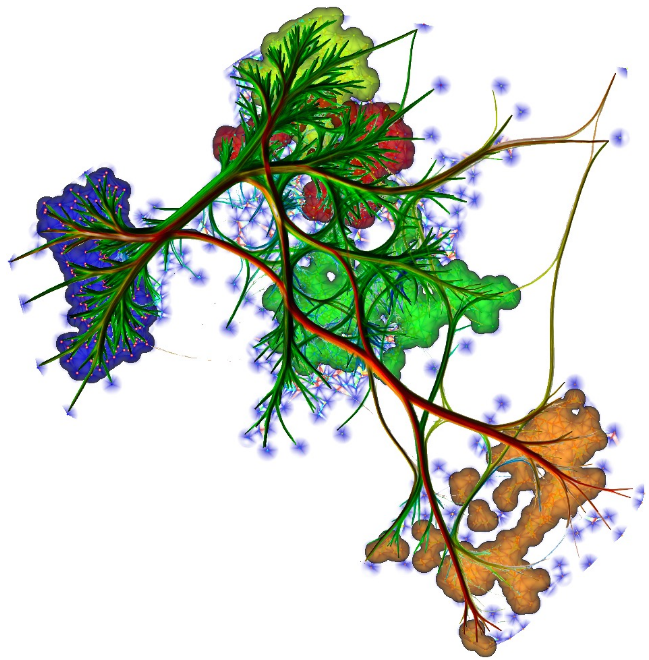
Multidimensional Data Visualization

Low-dimensional Data



prof. dr. Alexandru (Alex) Telea

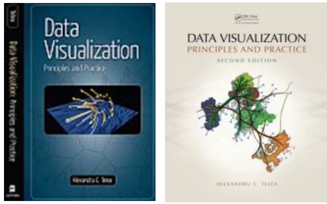
Department of Information and Computing Science
Utrecht University, the Netherlands

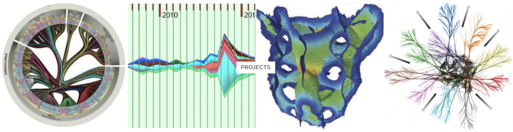
Utrecht University

Alexandru Telea

Professor Visual Data Analytics, Utrecht University, Netherlands

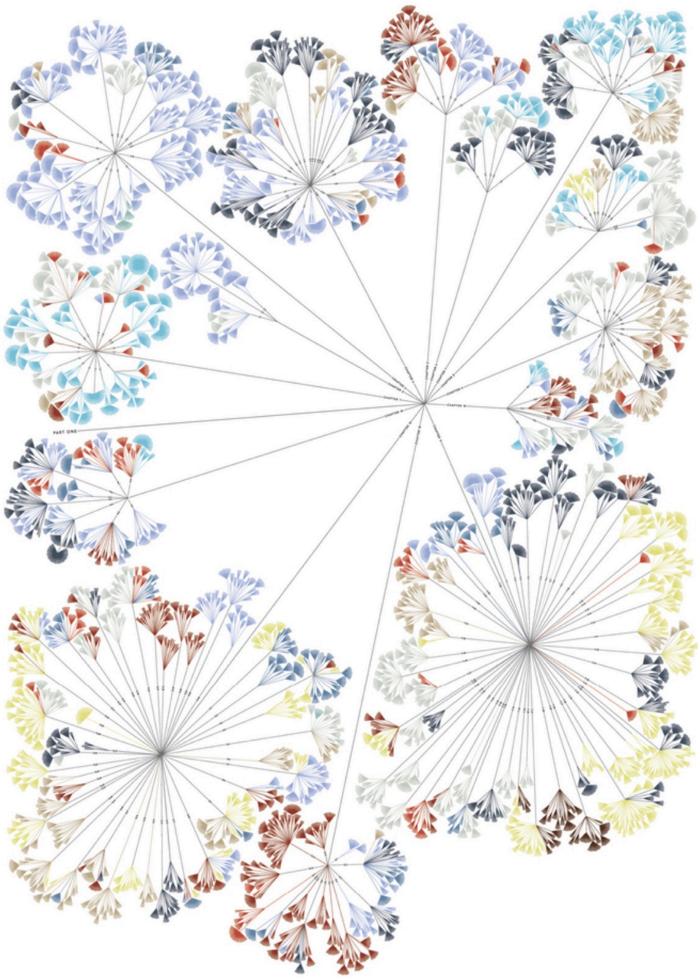
- full professor (UU, since 2019; RUG 2007-2019)
- 30 PhD students
- group leader Visualization and Graphics



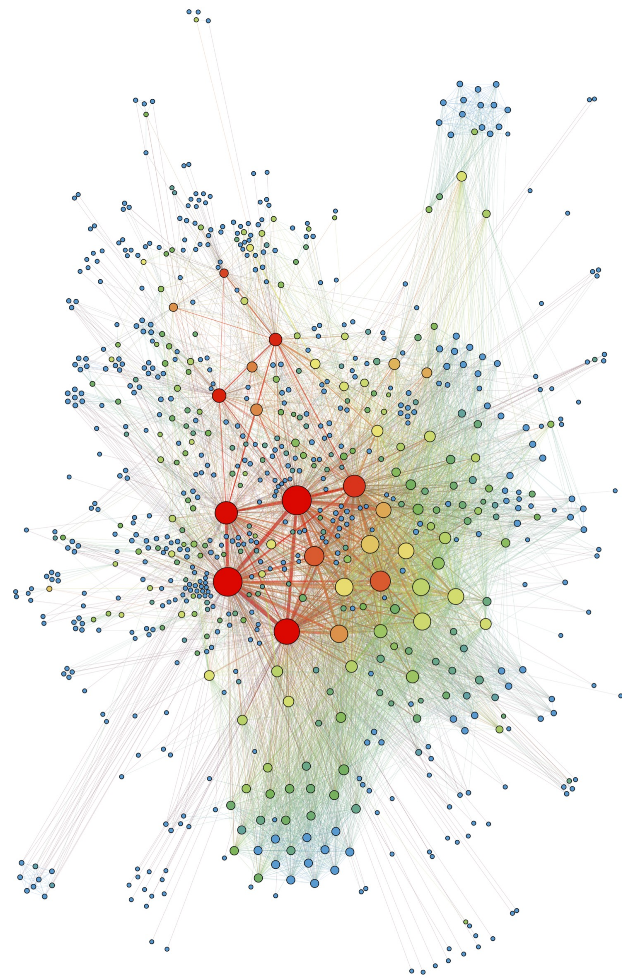


webspace.science.uu.nl/~telea001 a.c.telea@uu.nl vig.science.uu.nl

Visualizing Relational Data



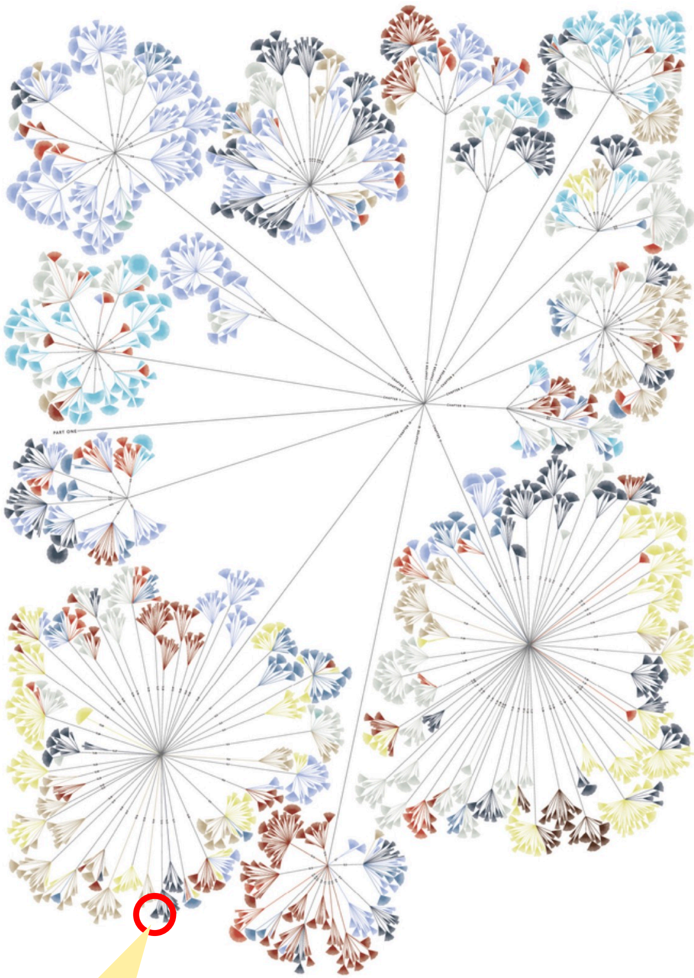
trees



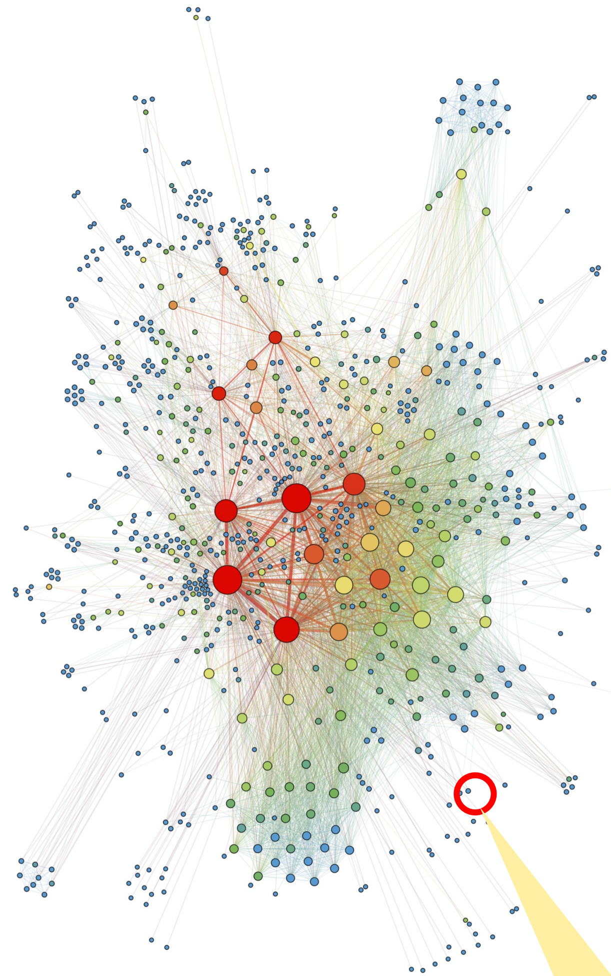
general graphs

You have learned how to do this

What about data attributes?



trees

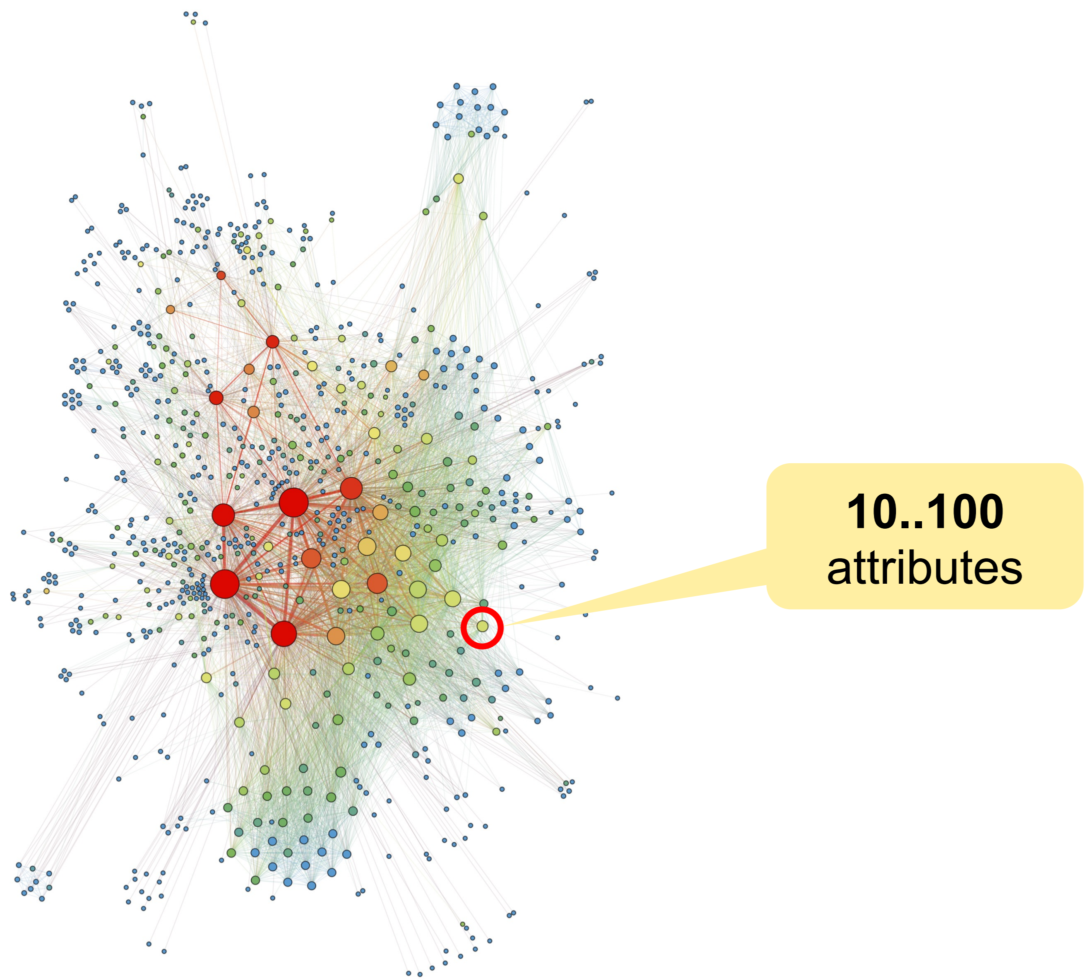


general graphs

1 attribute
(color)

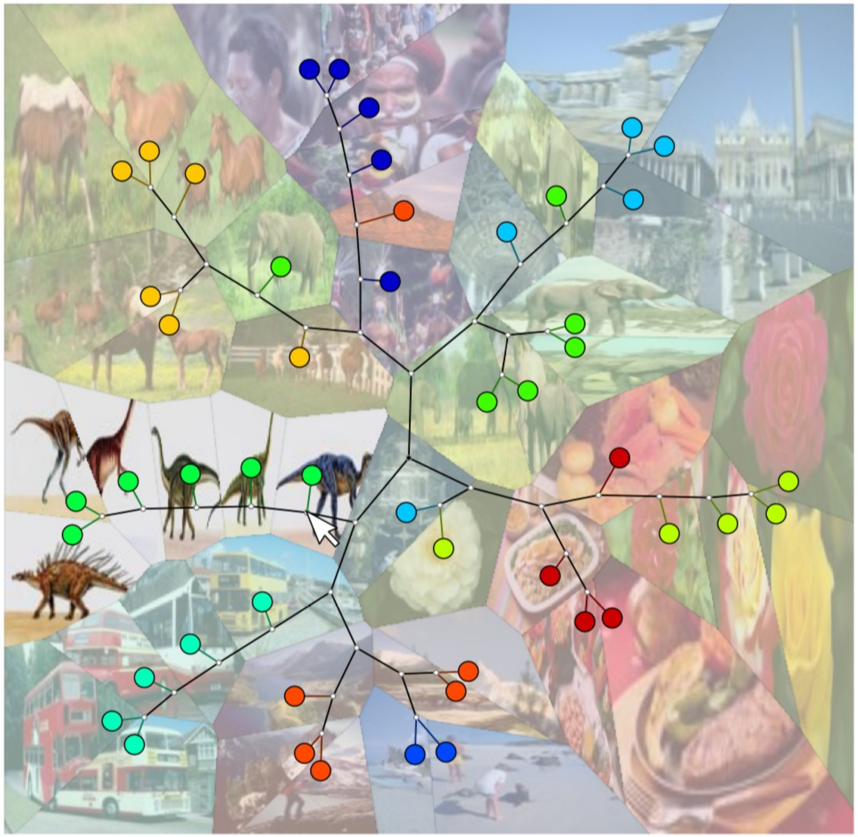
2 attributes
(color, size)

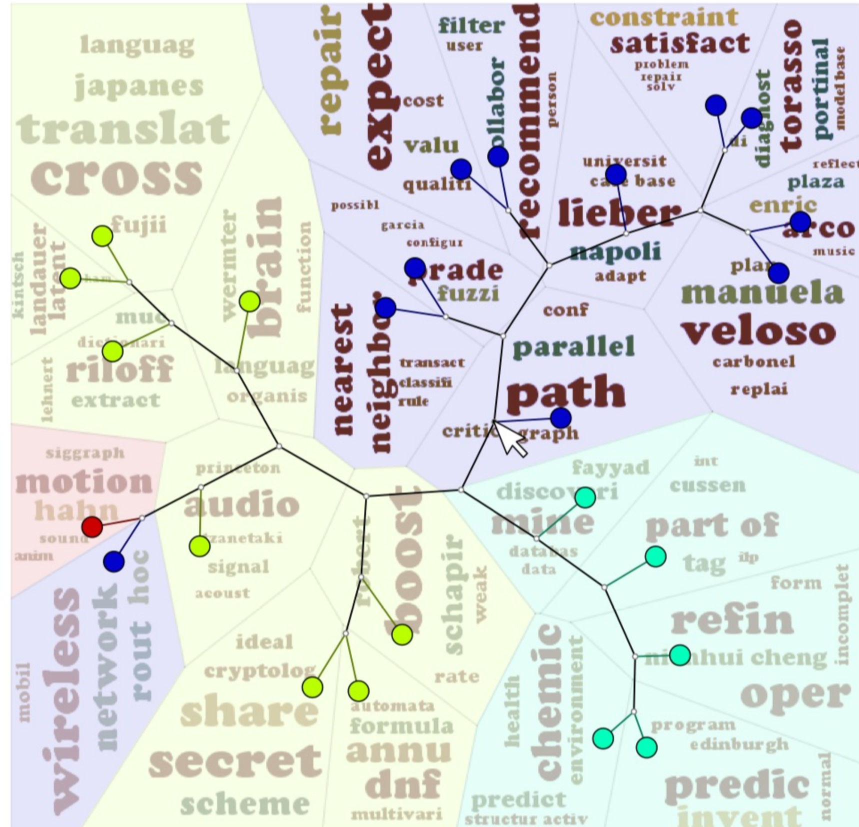
What about data attributes?



How to visualize this?

Examples of trees with data attributes





Attributes (per node)

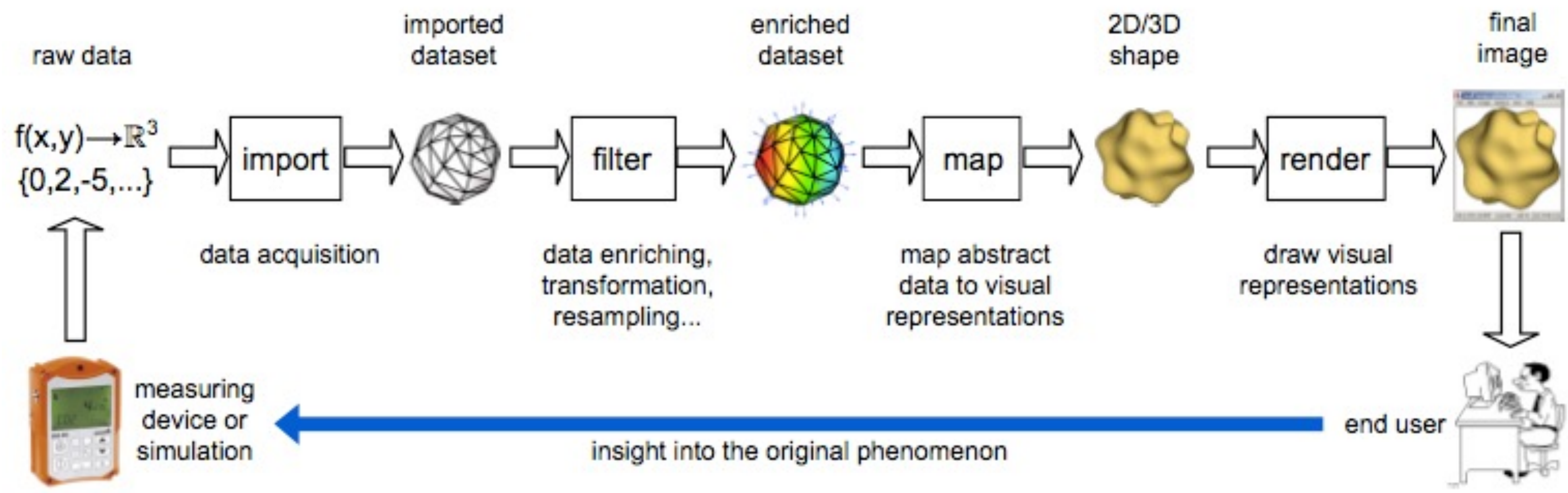
- type (color)
- image

Attributes (per node)

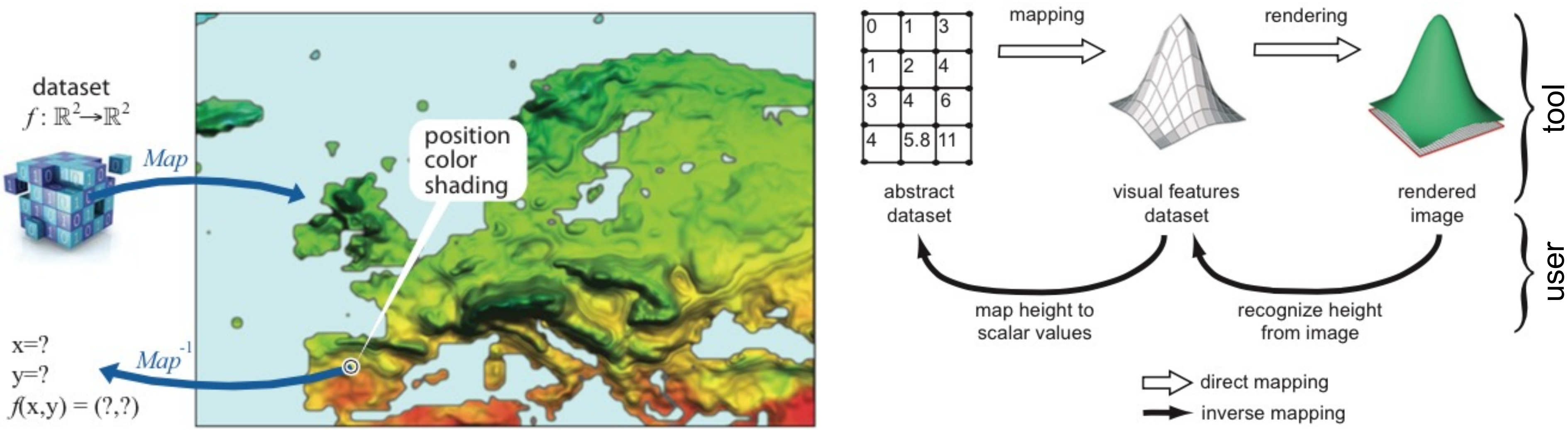
- type (color)
- multiple text tags

How to visualize *many* attributes?

The Visualization Pipeline



Direct *vs* Inverse Mapping Principles



A. Telea, Data Visualization – Principles and Practice, 2nd ed., CRC Press, 2014

What are data attributes?

attribute type	operations	examples
categorical[1]	equality	gender
ordinal	above, <,>	weekdays
discrete	above, +,-	#persons
quantitative	above, *,/	voltage

InfoVis data

SciVis data

less general

⋮

more general

[1] relations can be seen as ordered pairs of categorical attributes

Structure of a relational+attribute dataset

node ID ← *n* attributes per node → ← *m* edges (at most) per node →

Node	A_1	A_2	…	A_n	E_1	E_2	…	E_m
1					2	6		3
2					5	1		8
…								

any kinds/values of all above four attribute types

What about 'big data'?

Two independent things to measure

node ID — *n* attributes per node — *m* edges (at most) per node

Node	**A_1**	**A_2**	**…**	**A_n**	**E_1**	**E_2**	**…**	**E_m**
1					2	6		3
2					5	1		8
…								

N nodes

Number of nodes *N*

- each node (table row) has the same type and number of attributes
- also called samples, observations, or data points

Number of attributes *n*

- each attribute (table column) is of a given type (ordinal, categorical, etc)
- also called dimensions or variables

What about 'big data'?

Quiz

What is harder to visualize?

1000 data points having each **1** numerical attribute

100 data points having each **10** numerical attributes

Why?

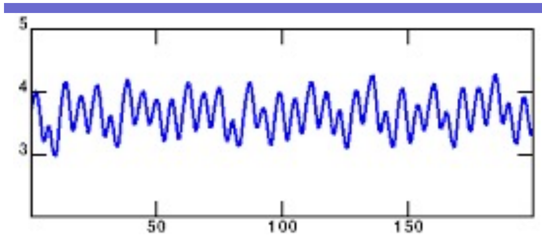
What about 'big data'?

What is harder to visualize

• **N=1000** data points having **n=1** single numerical attribute each
• **N=100** data points having **n=10** numerical attributes each

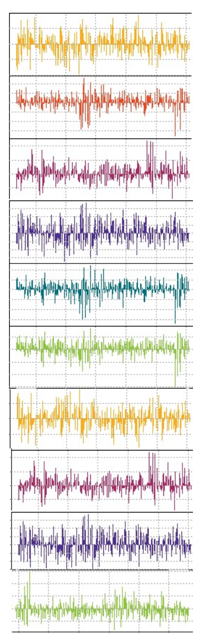
The number of values **n*N** to show is the same, but…

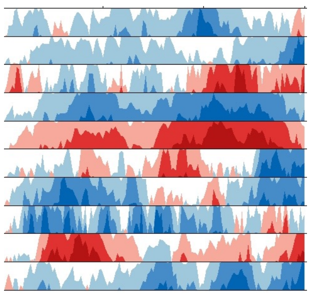
1000 samples x 1 attribute

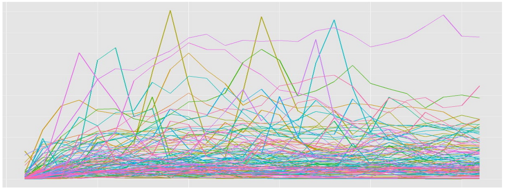


1D graphs/charts
work pretty well ☺

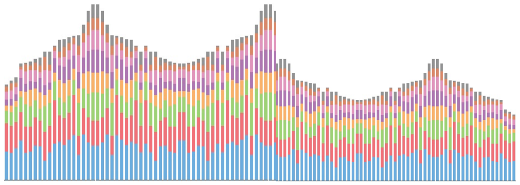
100 samples x 10 attributes

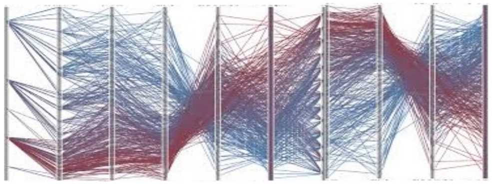






many chart kinds, many problems
(not scalable, cluttered, abstract, …)

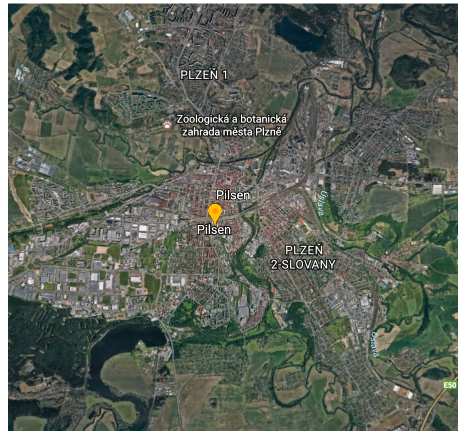


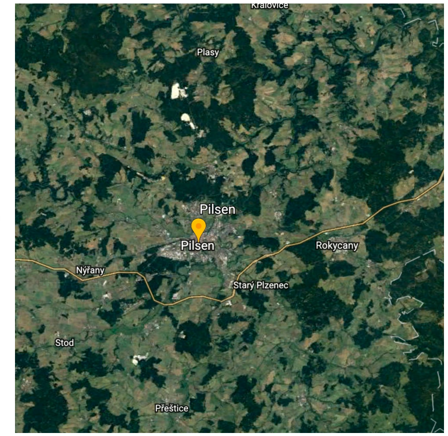


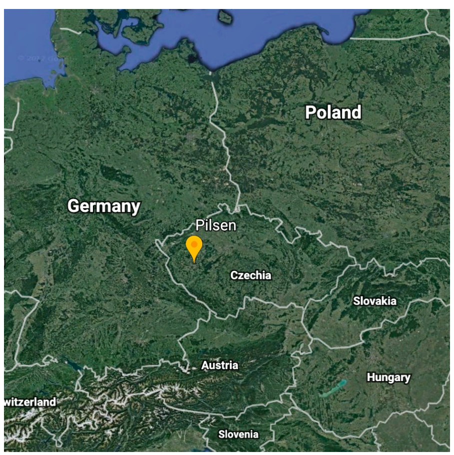
Solution: Aggregation

Aggregating the samples (N)

small N







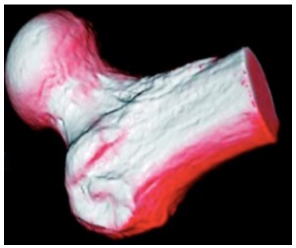
large N

Simple idea: reduce #data points (N) by **grouping** related samples (e.g. averaging)

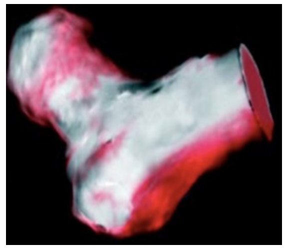
Does this always work?

Sample Aggregation Challenges

Continuous, numerical, spatial data



subsample ✓



bone dataset, 80K points

bone dataset, 20K points

subsample ✓

bone detail, 88 polygons

bone detail, 87 polygons

Discrete, non-numerical, non-spatial data

```
void ASTVisitor::traverse(ASTNode &obj)
{
  ASTNodeStack stack;
  static ASTNode sentinelNode(0); //put on the bottom of the s
  stack.push(StackItem(sentinelNode, SHOULD_IGNORE));
  stack.push(StackItem(obj, SHOULD_VISIT)); //the node that wi
  while(!stack.empty())
  {

    ASTNode &curNode(stack.top().astNode);
    if (stack.top().postVisit == SHOULD_IGNORE)
    {
      stack.pop();
    }
    else if (stack.top().postVisit == SHOULD_POSTVISIT)
    {
      const Visit visitResult(postVisitASTNode(curNode));
      if (visitResult == VISIT_STOP)
        return;
      stack.pop();
      if (visitResult == VISIT_POSTPARENT)
```

subsample ✗

? ? ? ? ? ?

C++ text, 80K lines

C++ text, 20K lines

```
#include <banking.h>

void bankCashTransfer(int amount)
{
  currentBalance += amount;
}
```

subsample ✗

```
#include <banking.h>

void bankCashTransfer(int amount)
{
  currentBalance = amount;
}
```

C++ text, 88 chars

C++ text, 87 chars

- we throw away 75% of the data
- the semantics stays the same

- interpolation: simple
- resampling: **Cauchy-continuous** ☺

- we throw away one single character
- the semantics becomes fully different!

- interpolation: often not possible
- resampling: **not Cauchy continuous** ☹

No simple solution for all datasets ☹

Solution: Aggregation

Aggregating the dimensions (n) by *selection*

table

n attributes

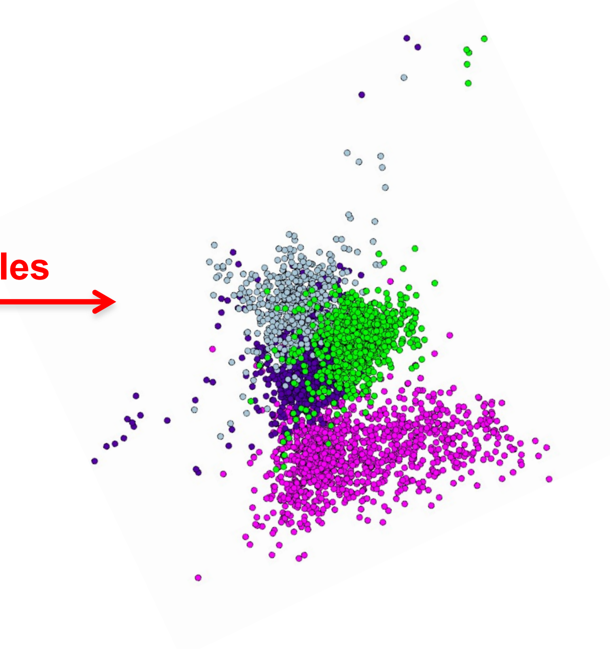
N samples

id	category	name	date	time	open	high	low	close
636	sif	SIF1	2004-11-29	13:00	0.800000	.800000	0.800000	0.800000
635	sif	SIF1	2004-11-29	14:00	0.800000	0.800000	0.800000	0.800000
633	sif	SIF1	2004-11-29	16:00	0.795000	0.795000	0.795000	0.795000
630	sif	SIF1	2004-11-30	14:00	0.795000	0.795000	0.795000	0.795000
632	sif	SIF1	2004-11-30	12:00	0.800000	0.800000	0.795000	0.795000
631	sif	SIF1	2004-11-30	13:00	0.795000	0.795000	0.795000	0.795000
628	sif	SIF1	2004-11-30	16:00	0.795000	0.795000	0.795000	0.795000
629	sif	SIF1	2004-11-30	15:00	0.795000	0.795000	0.795000	0.795000
627	sif	SIF1	2005-00-02	12:00	0.785000	0.790000	0.785000	0.790000
626	sif	SIF1	2005-00-02	13:00	0.790000	0.795000	0.790000	0.795000
625	sif	SIF1	2005-00-02	14:00	0.795000	0.795000	0.795000	0.795000
624	sif	SIF1	2005-00-02	15:00	0.800000	0.800000	0.800000	0.800000
620	sif	SIF1	2005-00-03	15:00	0.795000	0.795000	0.795000	0.795000
623	sif	SIF1	2005-00-03	12:00	0.795000	0.795000	0.795000	0.795000
622	sif	SIF1	2005-00-03	13:00	0.795000	0.795000	0.795000	0.795000
621	sif	SIF1	2005-00-03	14:00	0.795000	0.795000	0.795000	0.795000
619	sif	SIF1	2005-00-03	16:00	0.795000	0.795000	0.795000	0.795000
618	sif	SIF1	2005-00-06	11:00	0.790000	0.790000	0.790000	0.790000
614	sif	SIF1	2005-00-06	15:00	0.795000	0.795000	0.795000	0.795000
617	sif	SIF1	2005-00-06	12:00	0.795000	0.795000	0.795000	0.795000
616	sif	SIF1	2005-00-06	13:00	0.795000	0.795000	0.795000	0.795000
615	sif	SIF1	2005-00-06	14:00	0.795000	0.795000	0.795000	0.795000
613	sif	SIF1	2005-00-06	16:00	0.795000	0.795000	0.795000	0.795000
609	sif	SIF1	2005-00-07	14:00	0.790000	0.795000	0.790000	0.795000
612	sif	SIF1	2005-00-07	11:00	0.795000	0.795000	0.795000	0.795000
611	sif	SIF1	2005-00-07	12:00	0.795000	0.795000	0.795000	0.795000
610	sif	SIF1	2005-00-07	13:00	0.790000	0.790000	0.790000	0.790000
608	sif	SIF1	2005-00-07	15:00	0.790000	0.790000	0.790000	0.790000
606	sif	SIF1	2005-00-08	13:00	0.795000	0.795000	0.795000	0.795000
607	sif	SIF1	2005-00-08	12:00	0.790000	0.790000	0.790000	0.790000
605	sif	SIF1	2005-00-08	14:00	0.795000	0.795000	0.795000	0.795000

select a **few** attributes
map them to **visual variables**

A_1: x coordinate
A_2: y coordinate
A_3: color

visualization



Advantages

- very easy to do

Problems

- which (few) dimensions to select to visualize?
- what to do with the other (tens..hundreds of) dimensions?

Solution: Aggregation

Aggregating the dimensions (n) by *synthesis*

2D projection

n attributes

N samples

id	category	name	date	time	open	high	low	close
636	sif	SIF1	2004-11-29	13:00	0.800000	0.800000	0.800000	0.800000
635	sif	SIF1	2004-11-29	14:00	0.800000	0.800000	0.800000	0.800000
633	sif	SIF1	2004-11-29	16:00	0.795000	0.795000	0.795000	0.795000
630	sif	SIF1	2004-11-30	14:00	0.795000	0.795000	0.795000	0.795000
632	sif	SIF1	2004-11-30	12:00	0.800000	0.800000	0.795000	0.795000
631	sif	SIF1	2004-11-30	13:00	0.795000	0.795000	0.795000	0.795000
628	sif	SIF1	2004-11-30	16:00	0.795000	0.795000	0.795000	0.795000
629	sif	SIF1	2004-11-30	15:00	0.795000	0.795000	0.795000	0.795000
627	sif	SIF1	2005-00-02	12:00	0.785000	0.790000	0.785000	0.790000
626	sif	SIF1	2005-00-02	13:00	0.790000	0.795000	0.790000	0.795000
625	sif	SIF1	2005-00-02	14:00	0.795000	0.795000	0.795000	0.795000
624	sif	SIF1	2005-00-02	15:00	0.800000	0.800000	0.800000	0.800000
620	sif	SIF1	2005-00-03	15:00	0.795000	0.795000	0.795000	0.795000
623	sif	SIF1	2005-00-03	12:00	0.795000	0.795000	0.795000	0.795000
622	sif	SIF1	2005-00-03	13:00	0.795000	0.795000	0.795000	0.795000
621	sif	SIF1	2005-00-03	14:00	0.795000	0.795000	0.795000	0.795000
619	sif	SIF1	2005-00-03	16:00	0.795000	0.795000	0.795000	0.795000
618	sif	SIF1	2005-00-06	11:00	0.790000	0.790000	0.790000	0.790000
614	sif	SIF1	2005-00-06	15:00	0.795000	0.795000	0.795000	0.795000
617	sif	SIF1	2005-00-06	12:00	0.795000	0.795000	0.795000	0.795000
616	sif	SIF1	2005-00-06	13:00	0.795000	0.795000	0.795000	0.795000
615	sif	SIF1	2005-00-06	14:00	0.795000	0.795000	0.795000	0.795000
613	sif	SIF1	2005-00-06	16:00	0.795000	0.795000	0.795000	0.795000
609	sif	SIF1	2005-00-07	14:00	0.790000	0.795000	0.790000	0.795000
612	sif	SIF1	2005-00-07	11:00	0.795000	0.795000	0.795000	0.795000
611	sif	SIF1	2005-00-07	12:00	0.795000	0.795000	0.795000	0.795000
610	sif	SIF1	2005-00-07	13:00	0.790000	0.790000	0.790000	0.790000
608	sif	SIF1	2005-00-07	15:00	0.790000	0.790000	0.790000	0.790000
606	sif	SIF1	2005-00-08	13:00	0.795000	0.795000	0.795000	0.795000
607	sif	SIF1	2005-00-08	12:00	0.790000	0.790000	0.790000	0.790000
605	sif	SIF1	2005-00-08	14:00	0.795000	0.795000	0.795000	0.795000

synthesize **visual variables**
from **all** attributes

x coordinate: $A_1..A_n$
A_2: y coordinate
A_3: color

Advantages

- visualization encodes all the data (samples, attributes)

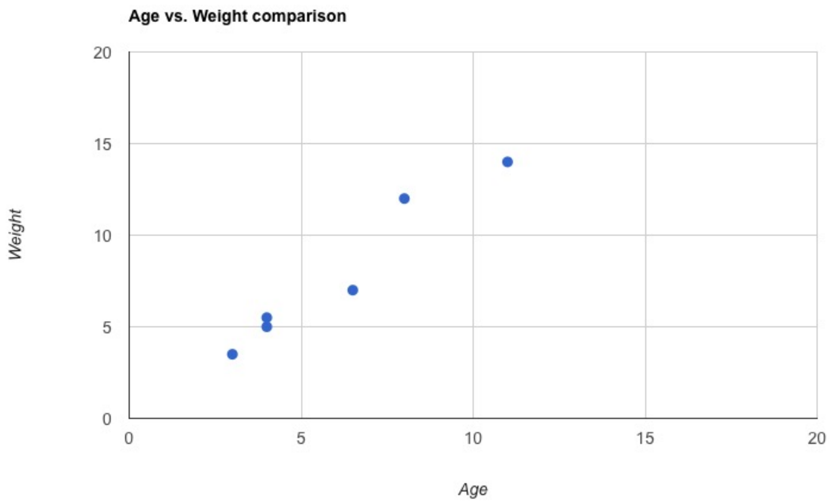
Problems

- what do the visual variables mean?
- how to decode the *n* attributes from them?

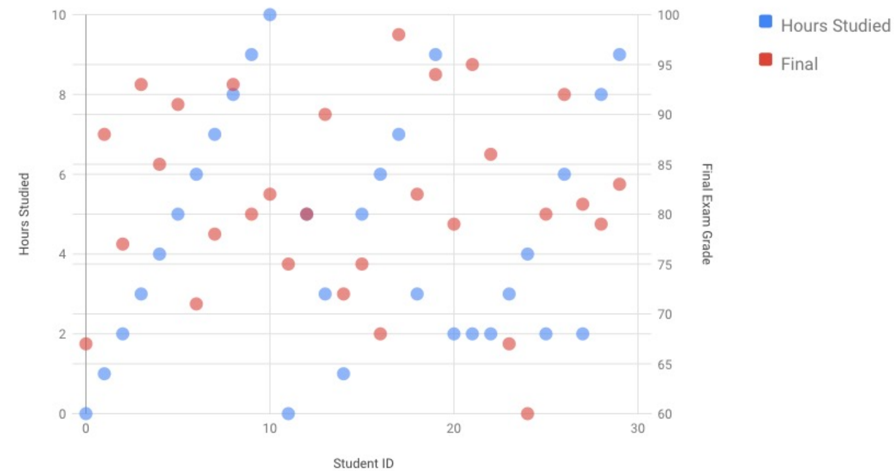
1. Scatterplots

Scatterplots

- show the **correlation** of 2..5 ordinal/quantitative variables (measured at the same points)



Two variables



Three variables (2 dots/data point)

Five variables (x/y position, size, color, label)

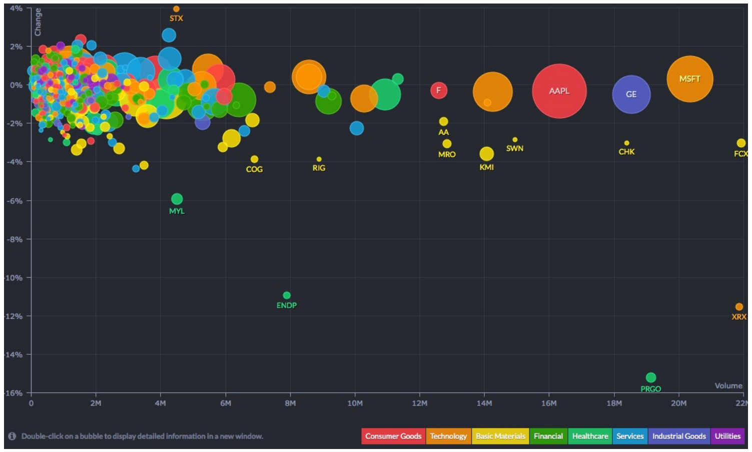
scatterplot using circle size to map data is also called a *bubble chart*

Data and tool: finviz.com

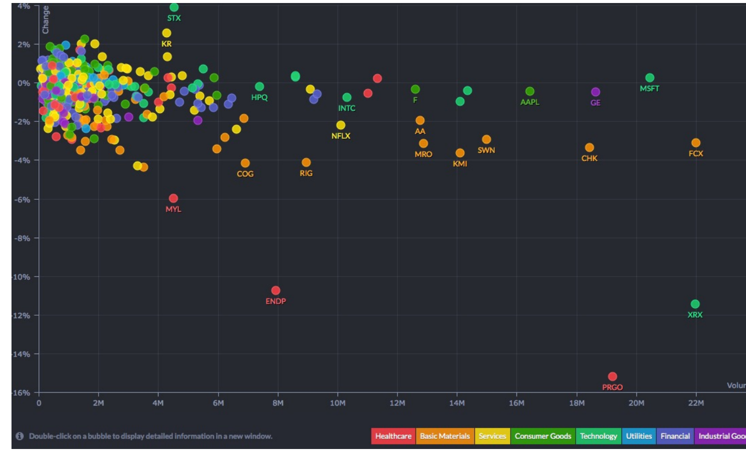
Scatterplot challenges

Size coding *vs* occlusion

- stock data (dot = stock, x = traded volume, y = percentage change, $color$ = industry sector)

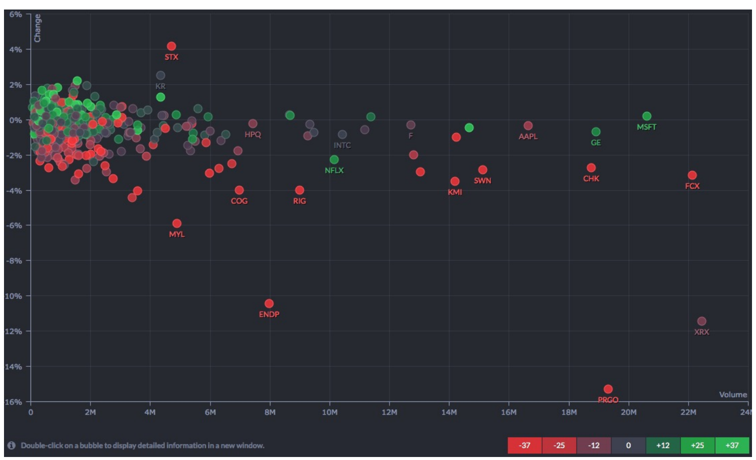


disk size = market capitalization
shows more data, but too much occlusion!

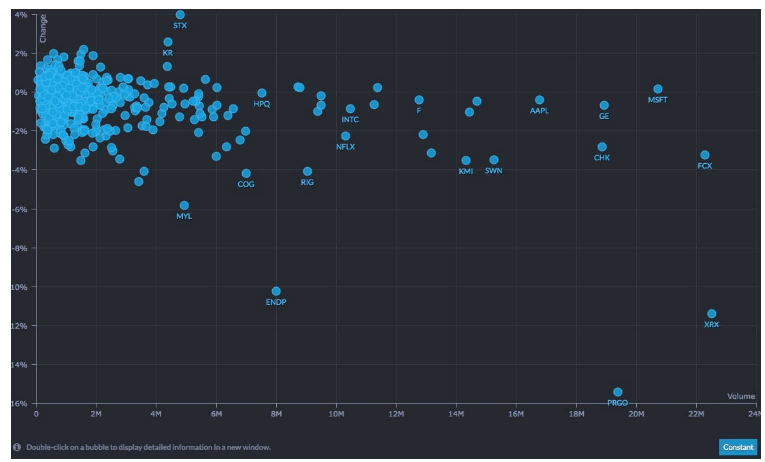


disk size = constant
shows less data (3 variables), but less occlusion

Color coding *vs* occlusion



color = 1-year performance (quantitative)
less clutter than when mapping categorical variable *industry sector* to color

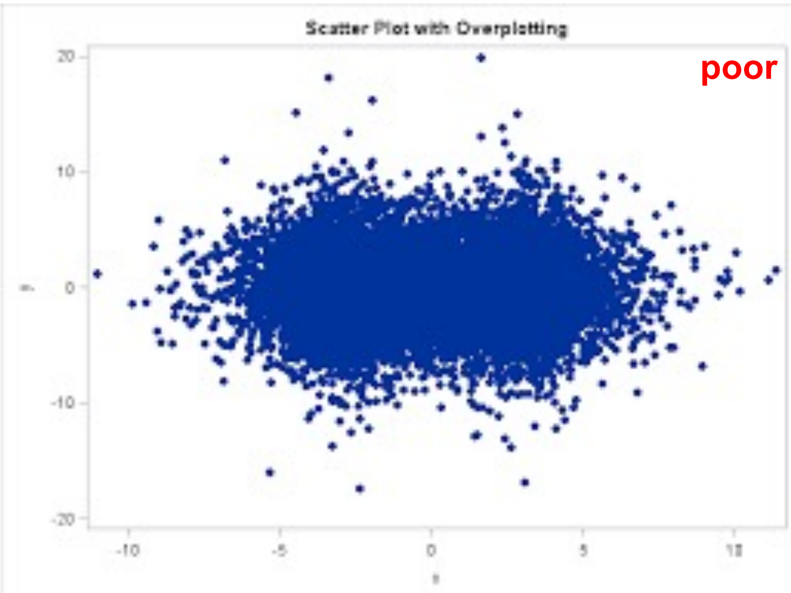


color = constant
shows less data (2 variables) – classical scatterplot
easiest plot to interpret from all

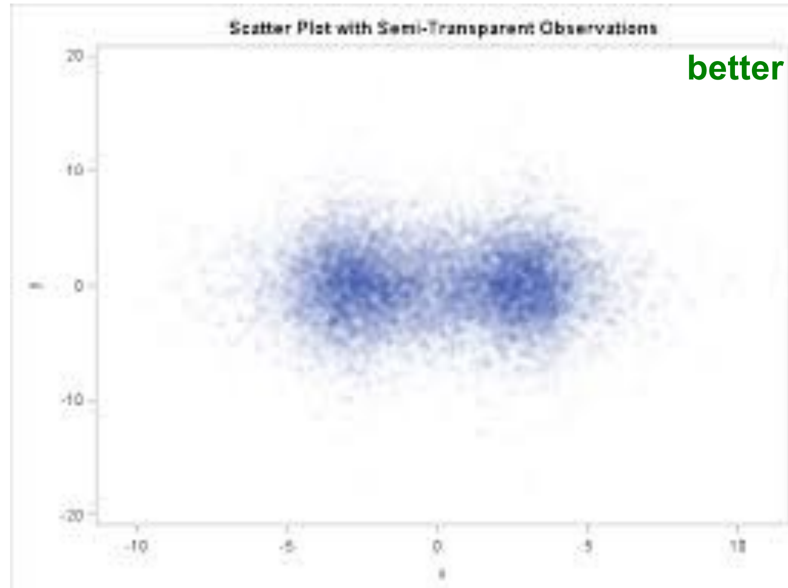
Encoding density

Take a very large scatterplot

- tens..hundreds of thousands of data items (points)
- how to handle **overplotting** (occlusion)?



standard scatterplot
we have no idea what happens inside the big blob!



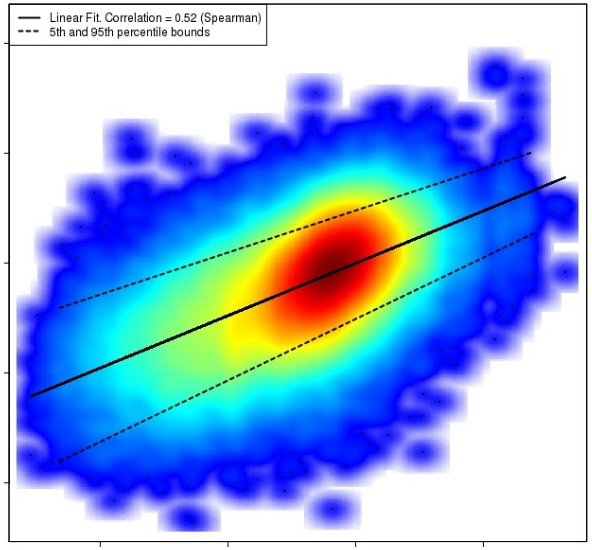
plotting half-transparent points
saturation shows local point density!

Transparency coding

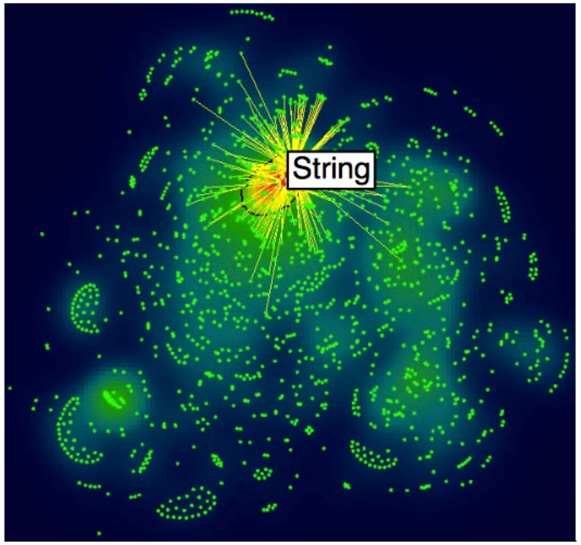
- very simple technique
- emphasizes data-rich regions, suppresses outliers
- in the limit: scatterplots become continuous **density fields** (for millions of points)

Encoding density

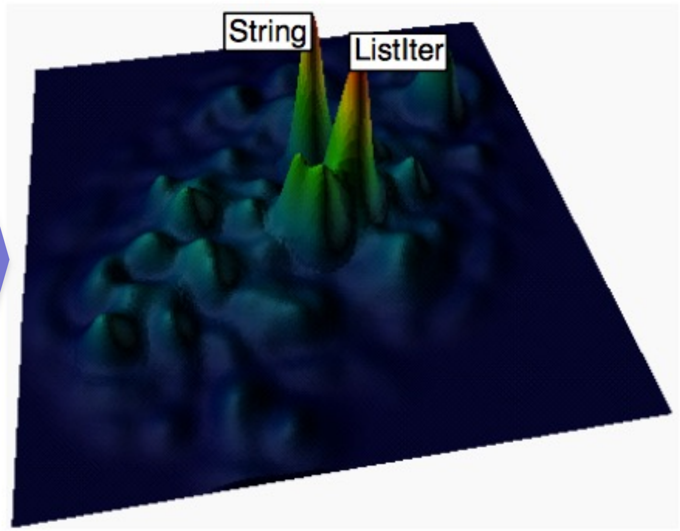
Going from discrete scatterplots to continuous fields



color-coded density field
we see local point density well



color-coded density field
we see local point density well



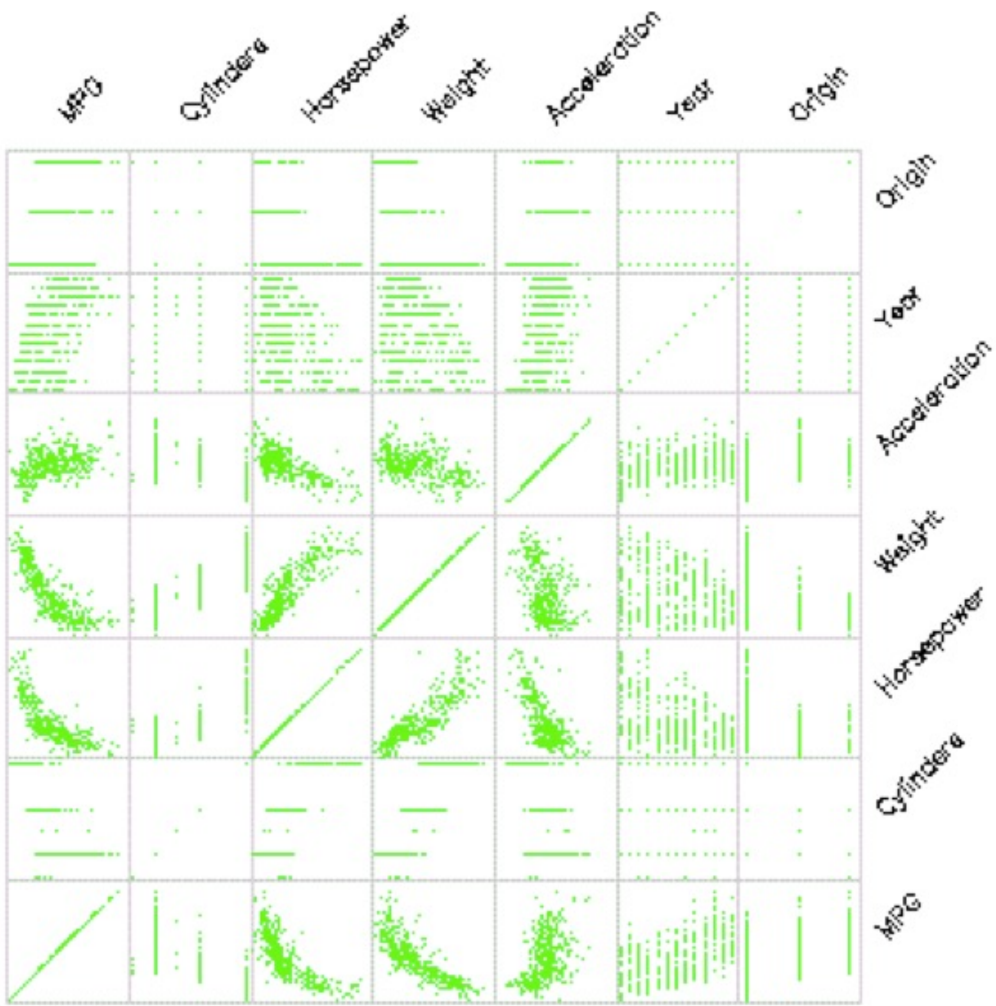
color-and-height coded density field
we see local point density even better

Main idea

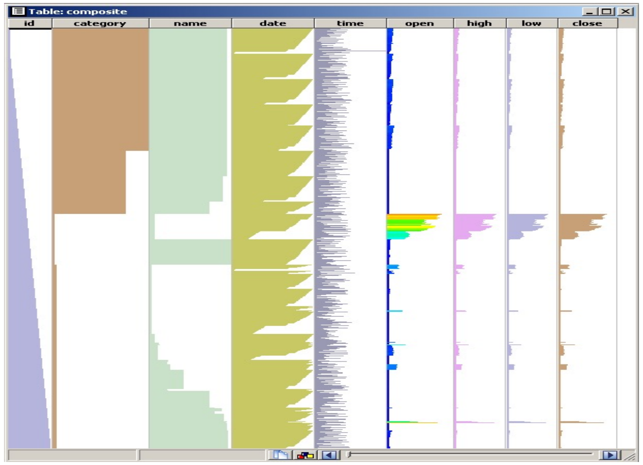
- interpret the point-density as a **continuous** 2D scalar field
- this transforms the discrete scatterplot into a continuous field
- visualize this continuous field using classical field visualization methods
    - continuous color coding (e.g. using rainbow colormap)
    - 3D height plots, contours
- abstract from details, show **overview** (good for big data)

Scatterplot matrices (SPLOMs)

- consider all pairs of **columns** c_i , c_j
  - construct scatterplots P_{ij} for all pairs c_i , c_j over all rows
  - arrange P_{ij} in a (symmetric) matrix; shows correlation of any c_i with any c_j
- nice, but does not scale to tens..hundreds of columns



2. Data Tables



Tables

- one of the most ubiquitous types of (InfoVis) data
- table: set of rows (observations) and columns (dimensions)
- columns can have different **types**
- rows and columns are not uniquely **ordered**
- drawing a large table (> 10 columns or 50 rows) becomes useless…

How to visualize big tables?

Example: stock exchange data

id	category	name	date	time	open	high	low	close
636	sif	SIF1	2004-11-29	13:00	0.800000	0.800000	0.800000	0.800000
635	sif	SIF1	2004-11-29	14:00	0.800000	0.800000	0.800000	0.800000
633	sif	SIF1	2004-11-29	16:00	0.795000	0.795000	0.795000	0.795000
630	sif	SIF1	2004-11-30	14:00	0.795000	0.795000	0.795000	0.795000
632	sif	SIF1	2004-11-30	12:00	0.800000	0.800000	0.795000	0.795000
631	sif	SIF1	2004-11-30	13:00	0.795000	0.795000	0.795000	0.795000
628	sif	SIF1	2004-11-30	16:00	0.795000	0.795000	0.795000	0.795000
629	sif	SIF1	2004-11-30	15:00	0.795000	0.795000	0.795000	0.795000
627	sif	SIF1	2005-00-02	12:00	0.785000	0.790000	0.785000	0.790000
626	sif	SIF1	2005-00-02	13:00	0.790000	0.795000	0.790000	0.795000
625	sif	SIF1	2005-00-02	14:00	0.795000	0.795000	0.795000	0.795000
624	sif	SIF1	2005-00-02	15:00	0.800000	0.800000	0.800000	0.800000
620	sif	SIF1	2005-00-03	15:00	0.795000	0.795000	0.795000	0.795000
623	sif	SIF1	2005-00-03	12:00	0.795000	0.795000	0.795000	0.795000
622	sif	SIF1	2005-00-03	13:00	0.795000	0.795000	0.795000	0.795000
621	sif	SIF1	2005-00-03	14:00	0.795000	0.795000	0.795000	0.795000
619	sif	SIF1	2005-00-03	16:00	0.795000	0.795000	0.795000	0.795000
618	sif	SIF1	2005-00-06	11:00	0.790000	0.790000	0.790000	0.790000
614	sif	SIF1	2005-00-06	15:00	0.795000	0.795000	0.795000	0.795000
617	sif	SIF1	2005-00-06	12:00	0.795000	0.795000	0.795000	0.795000
616	sif	SIF1	2005-00-06	13:00	0.795000	0.795000	0.795000	0.795000
615	sif	SIF1	2005-00-06	14:00	0.795000	0.795000	0.795000	0.795000
613	sif	SIF1	2005-00-06	16:00	0.795000	0.795000	0.795000	0.795000
609	sif	SIF1	2005-00-07	14:00	0.790000	0.795000	0.790000	0.795000
612	sif	SIF1	2005-00-07	11:00	0.795000	0.795000	0.795000	0.795000
611	sif	SIF1	2005-00-07	12:00	0.795000	0.795000	0.795000	0.795000
610	sif	SIF1	2005-00-07	13:00	0.790000	0.790000	0.790000	0.790000
608	sif	SIF1	2005-00-07	15:00	0.790000	0.790000	0.790000	0.790000
606	sif	SIF1	2005-00-08	13:00	0.795000	0.795000	0.795000	0.795000
607	sif	SIF1	2005-00-08	12:00	0.790000	0.790000	0.790000	0.790000
605	sif	SIF1	2005-00-08	14:00	0.795000	0.795000	0.795000	0.795000

m transactions

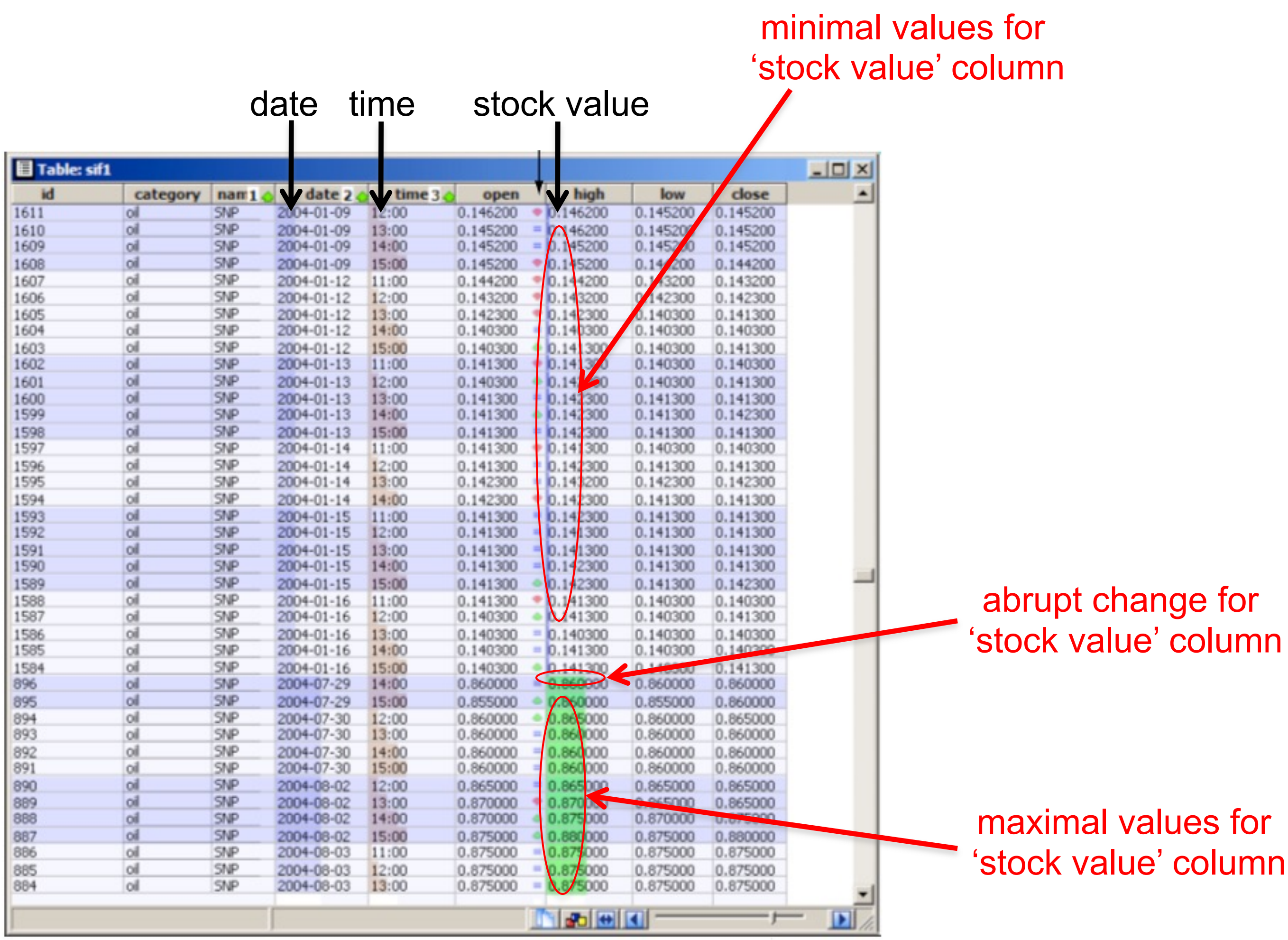
n attributes of a transaction

Tables

First enhancement: overlay bar charts on columns

Added value

- quickly scrolling through the table pre-attentively highlights minima, maxima, and large changes
- this allows us to explore large tables easily

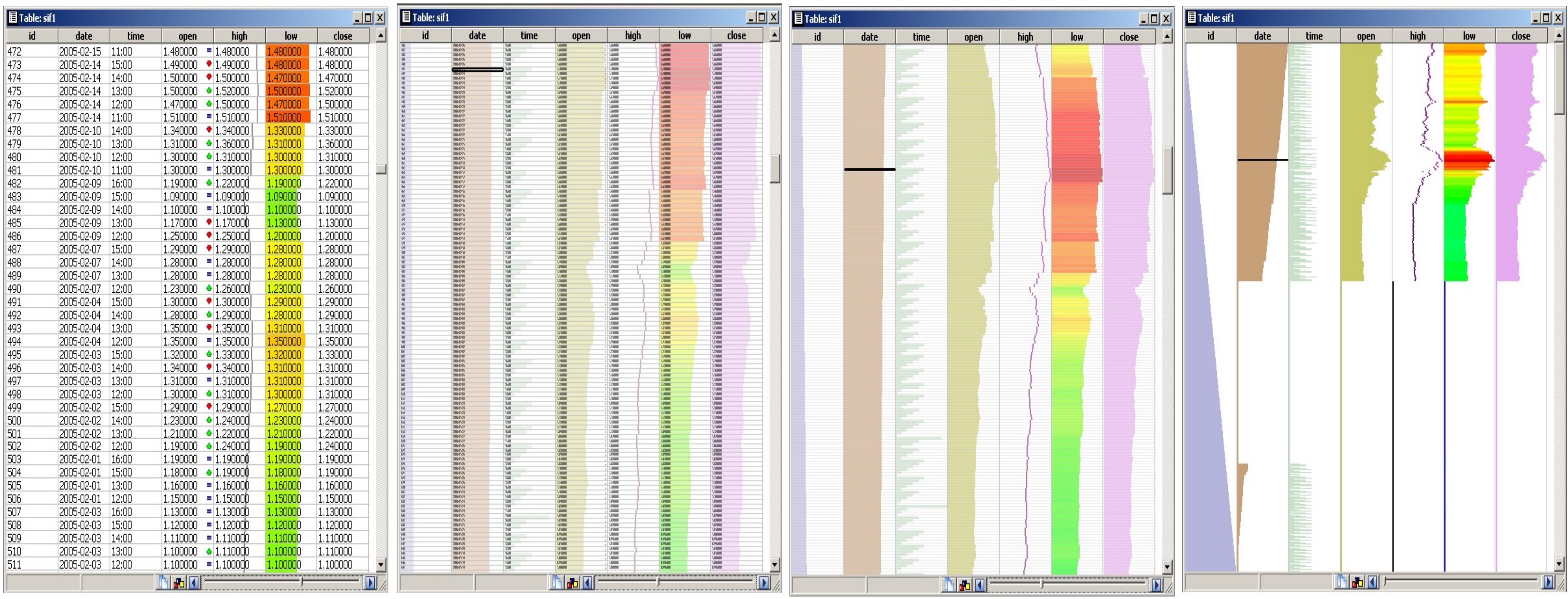


id	category	nam 1	date 2	time 3	open	high	low	close
1611	oil	SNP	2004-01-09	12:00	0.146200	0.146200	0.145200	0.145200
1610	oil	SNP	2004-01-09	13:00	0.145200	0.146200	0.145200	0.145200
1609	oil	SNP	2004-01-09	14:00	0.145200	0.145200	0.145200	0.145200
1608	oil	SNP	2004-01-09	15:00	0.145200	0.145200	0.144200	0.144200
1607	oil	SNP	2004-01-12	11:00	0.144200	0.144200	0.143200	0.143200
1606	oil	SNP	2004-01-12	12:00	0.143200	0.143200	0.142300	0.142300
1605	oil	SNP	2004-01-12	13:00	0.142300	0.142300	0.140300	0.141300
1604	oil	SNP	2004-01-12	14:00	0.140300	0.140300	0.140300	0.140300
1603	oil	SNP	2004-01-12	15:00	0.140300	0.141300	0.140300	0.141300
1602	oil	SNP	2004-01-13	11:00	0.141300	0.141300	0.140300	0.140300
1601	oil	SNP	2004-01-13	12:00	0.140300	[illegible]	0.140300	0.141300
1600	oil	SNP	2004-01-13	13:00	0.141300	0.142300	0.141300	0.141300
1599	oil	SNP	2004-01-13	14:00	0.141300	0.142300	0.141300	0.142300
1598	oil	SNP	2004-01-13	15:00	0.141300	0.142300	0.141300	0.141300
1597	oil	SNP	2004-01-14	11:00	0.141300	0.141300	0.140300	0.140300
1596	oil	SNP	2004-01-14	12:00	0.141300	0.142300	0.141300	0.141300
1595	oil	SNP	2004-01-14	13:00	0.142300	0.143200	0.142300	0.142300
1594	oil	SNP	2004-01-14	14:00	0.142300	0.142300	0.141300	0.141300
1593	oil	SNP	2004-01-15	11:00	0.141300	0.142300	0.141300	0.141300
1592	oil	SNP	2004-01-15	12:00	0.141300	0.141300	0.141300	0.141300
1591	oil	SNP	2004-01-15	13:00	0.141300	0.141300	0.141300	0.141300
1590	oil	SNP	2004-01-15	14:00	0.141300	0.142300	0.141300	0.141300
1589	oil	SNP	2004-01-15	15:00	0.141300	0.142300	0.141300	0.142300
1588	oil	SNP	2004-01-16	11:00	0.141300	0.141300	0.140300	0.140300
1587	oil	SNP	2004-01-16	12:00	0.140300	0.141300	0.140300	0.141300
1586	oil	SNP	2004-01-16	13:00	0.140300	0.140300	0.140300	0.140300
1585	oil	SNP	2004-01-16	14:00	0.140300	0.141300	0.140300	[illegible]
1584	oil	SNP	2004-01-16	15:00	0.140300	0.141300	[illegible]	0.141300
896	oil	SNP	2004-07-29	14:00	0.860000	[illegible]	0.860000	0.860000
895	oil	SNP	2004-07-29	15:00	0.855000	0.860000	0.855000	0.860000
894	oil	SNP	2004-07-30	12:00	0.860000	0.865000	0.860000	0.865000
893	oil	SNP	2004-07-30	13:00	0.860000	0.860000	0.860000	0.860000
892	oil	SNP	2004-07-30	14:00	0.860000	0.860000	0.860000	0.860000
891	oil	SNP	2004-07-30	15:00	0.860000	0.860000	0.860000	0.860000
890	oil	SNP	2004-08-02	12:00	0.865000	0.865000	0.865000	0.865000
889	oil	SNP	2004-08-02	13:00	0.870000	[illegible]	0.865000	0.865000
888	oil	SNP	2004-08-02	14:00	0.870000	0.875000	0.870000	[illegible]
887	oil	SNP	2004-08-02	15:00	0.875000	0.880000	0.875000	0.880000
886	oil	SNP	2004-08-03	11:00	0.875000	0.875000	0.875000	0.875000
885	oil	SNP	2004-08-03	12:00	0.875000	0.875000	0.875000	0.875000
884	oil	SNP	2004-08-03	13:00	0.875000	0.875000	0.875000	0.875000

Tables

Second enhancement: the ‘table lens’ technique [Rao *et al*, ’94]

- shows the table at any user-chosen level of detail



id	date	time	open	high	low	close
472	2005-02-15	11:00	1.480000	1.480000	1.480000	1.480000
473	2005-02-14	15:00	1.490000	1.490000	1.480000	1.480000
474	2005-02-14	14:00	1.500000	1.500000	1.470000	1.470000
475	2005-02-14	13:00	1.500000	1.520000	1.500000	1.520000
476	2005-02-14	12:00	1.470000	1.500000	1.470000	1.500000
477	2005-02-14	11:00	1.510000	1.510000	1.510000	1.510000
478	2005-02-10	14:00	1.340000	1.340000	1.330000	1.330000
479	2005-02-10	13:00	1.310000	1.360000	1.310000	1.360000
480	2005-02-10	12:00	1.300000	1.310000	1.300000	1.310000
481	2005-02-10	11:00	1.300000	1.300000	1.300000	1.300000
482	2005-02-09	16:00	1.190000	1.220000	1.190000	1.220000
483	2005-02-09	15:00	1.090000	1.090000	1.090000	1.090000
484	2005-02-09	14:00	1.100000	1.100000	1.100000	1.100000
485	2005-02-09	13:00	1.170000	1.170000	1.130000	1.130000
486	2005-02-09	12:00	1.250000	1.250000	1.200000	1.200000
487	2005-02-07	15:00	1.290000	1.290000	1.280000	1.280000
488	2005-02-07	14:00	1.280000	1.280000	1.280000	1.280000
489	2005-02-07	13:00	1.280000	1.280000	1.280000	1.280000
490	2005-02-07	12:00	1.230000	1.260000	1.230000	1.260000
491	2005-02-04	15:00	1.300000	1.300000	1.290000	1.290000
492	2005-02-04	14:00	1.280000	1.290000	1.280000	1.290000
493	2005-02-04	13:00	1.350000	1.350000	1.310000	1.310000
494	2005-02-04	12:00	1.350000	1.350000	1.350000	1.350000
495	2005-02-03	15:00	1.320000	1.330000	1.320000	1.330000
496	2005-02-03	14:00	1.340000	1.340000	1.310000	1.310000
497	2005-02-03	13:00	1.310000	1.310000	1.310000	1.310000
498	2005-02-03	12:00	1.300000	1.310000	1.300000	1.310000
499	2005-02-02	15:00	1.290000	1.290000	1.270000	1.270000
500	2005-02-02	14:00	1.230000	1.240000	1.230000	1.240000
501	2005-02-02	13:00	1.210000	1.220000	1.210000	1.220000
502	2005-02-02	12:00	1.190000	1.240000	1.190000	1.240000
503	2005-02-01	16:00	1.190000	1.190000	1.190000	1.190000
504	2005-02-01	15:00	1.180000	1.190000	1.180000	1.190000
505	2005-02-01	13:00	1.160000	1.160000	1.160000	1.160000
506	2005-02-01	12:00	1.150000	1.150000	1.150000	1.150000
507	2005-02-03	16:00	1.130000	1.130000	1.130000	1.130000
508	2005-02-03	15:00	1.120000	1.120000	1.120000	1.120000
509	2005-02-03	14:00	1.110000	1.110000	1.110000	1.110000
510	2005-02-03	13:00	1.100000	1.110000	1.100000	1.110000
511	2005-02-03	12:00	1.100000	1.100000	1.100000	1.100000

text opacity=1 font size=12pt	text opacity ↓ font size ↓	text not drawn bar opacity↑	bar opacity=1 simplification=on

min **zoom level** *max*

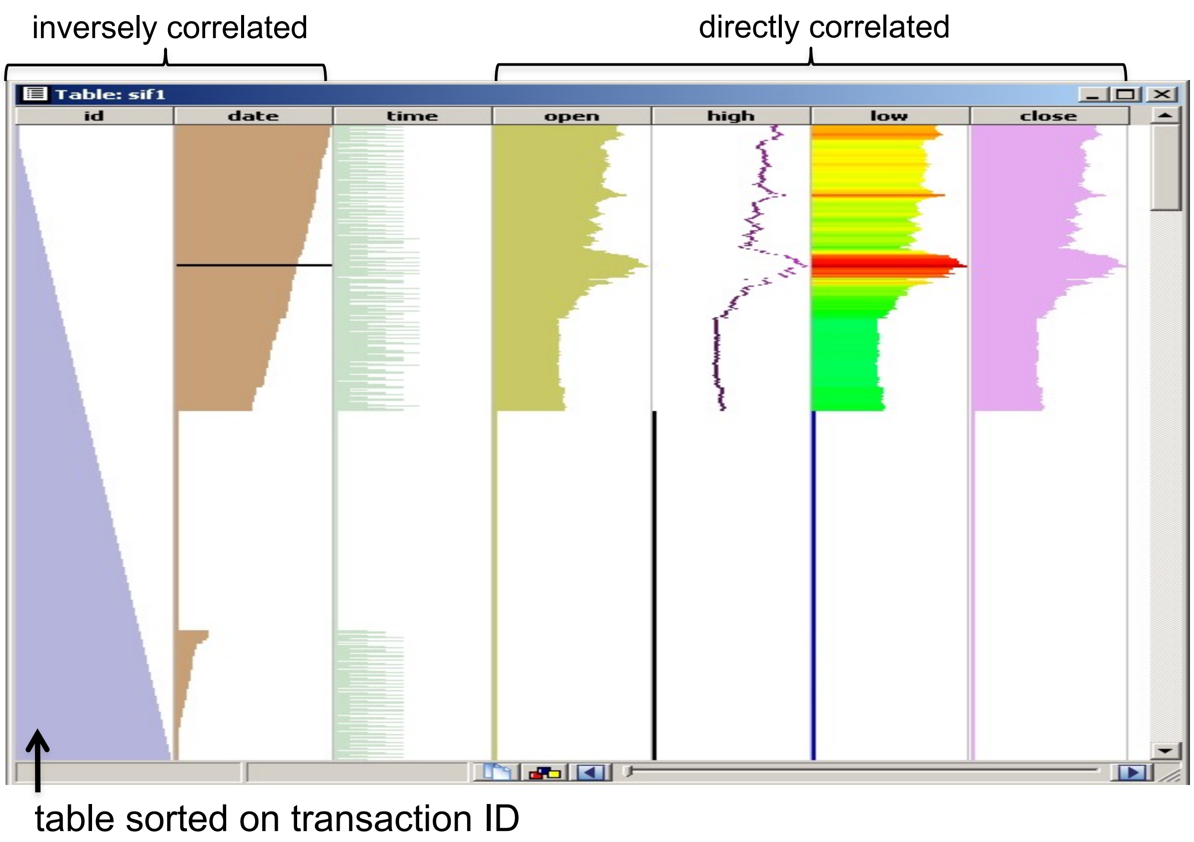
R. Rao, S. Card (1994) The table lens: merging graphical and symbolic representations in an interactive focus + context visualization for tabular information, Proc. ACM CHI, 318-322

Tables

Third enhancement: single-column sorting

- **sort** table on values of user-selected column (attribute)
- zoom-out mode shows **distribution** and **correlation** of column values

Stock data example

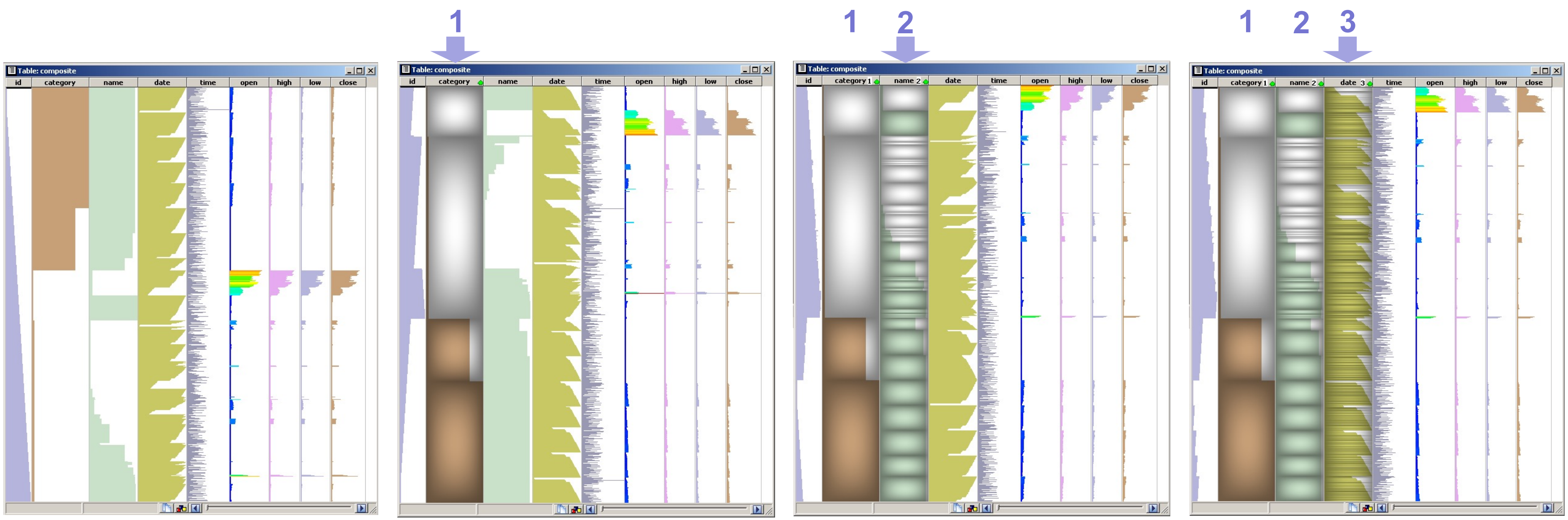


Tables

Fourth enhancement: multiple-column sorting and row grouping

- **sort** table on multiple user-selected column values
- emphasize **same-value** column ranges with *cushions*[1]

Show stock data grouped by industry, company, and date



no sort — sort on *industry* — next sort on *company* — next sort on *date*

Sorting has two roles

- group rows having same value in an attribute
- show how other attributes vary within a group

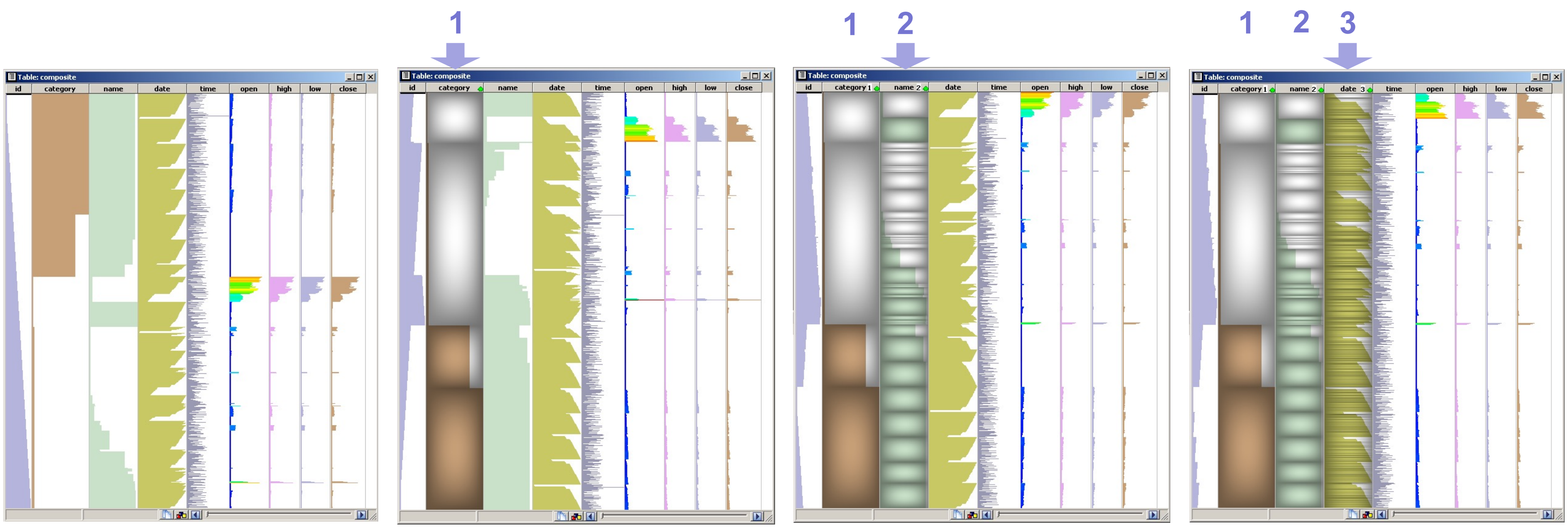
[1]J. J. van Wijk, H. van de Wetering (1999) Cushion Treemaps: Visualization of Hierarchical Information, Proc. InfoVis, 73-81

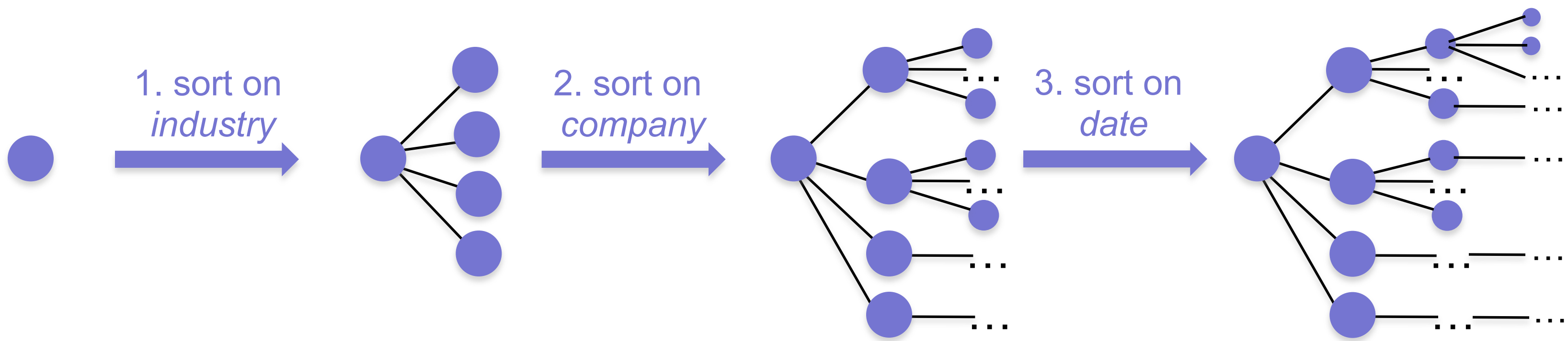
Tables

Multiple sorting: generates on-the-fly a hierarchy (tree) from the table

- one tree-level per sort
- one node per group of sorted rows having the same value

We'll see soon how to visualize this tree!



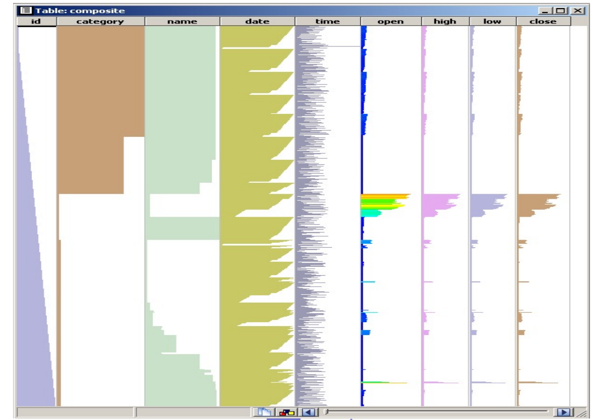


Design Question: Trees or Tables?

Consider this simple table:

Name	Age	Salary	Function
John Doe	47	65K	management
Bill Smith	35	40K	IT
Fanny Mae	37	35K	administration

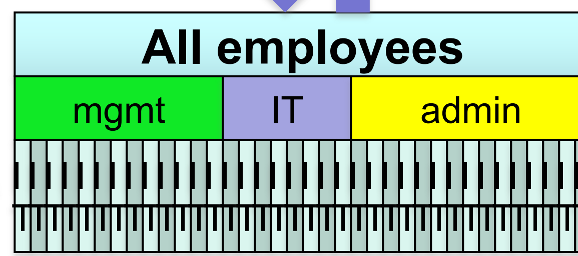
can do this…

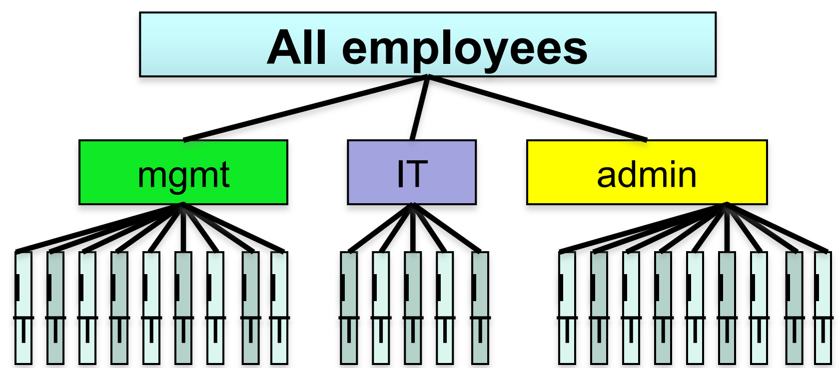


group by value of Function to create tree levels

create table from tree each tree level: add 1 table column

or this…





Conclusion

- both designs are possible
- ‘right’ one depends on type of task
  - compare items **per category:** use **tree** design
  - compare all items **across categories:** use **table** design

Icicle plots

Basic idea

- nodes: rectangles; edges: not drawn explicitly, but shown by node positions
- one level per vertical band (root at left, leaves at right)
- siblings stacked vertically within a band

- compact display, no clutter; node size = sum of subtree importance

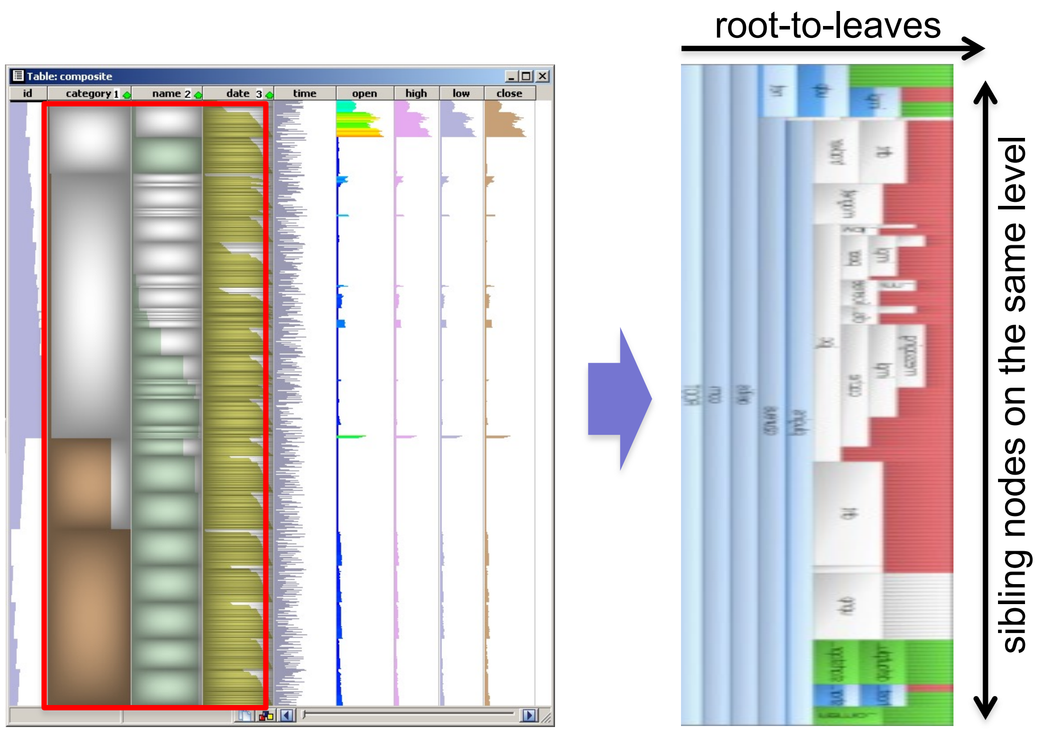


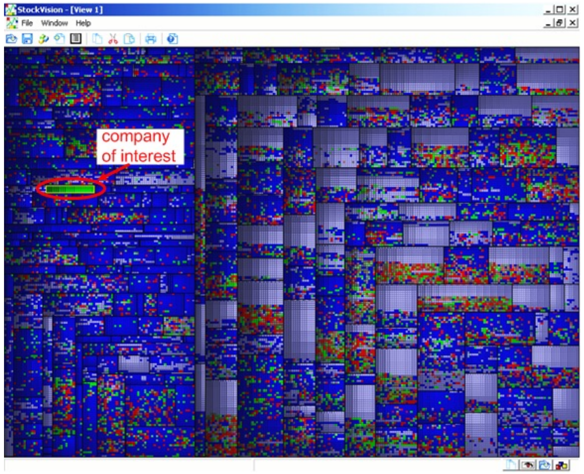
table lens, multiple sorting

icicle plot[1]

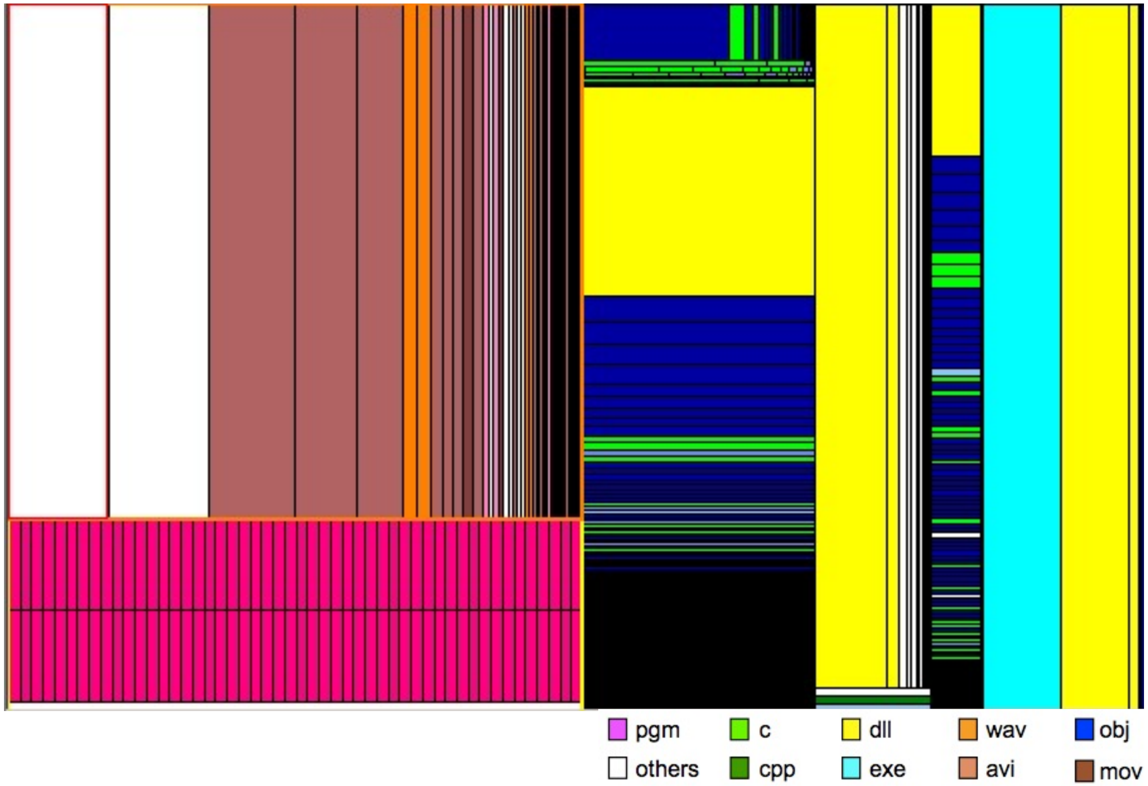
We shall see several variations of the icicle plot further!

[1]J. Kruskal, J. Landwehr, Icicle plots: Better displays for hierarchical clustering, JSTOR, 1983

3. Treemaps



Treemaps [Shneiderman '92]



ffmpeg library

Basic idea: 'slice and dice' layout

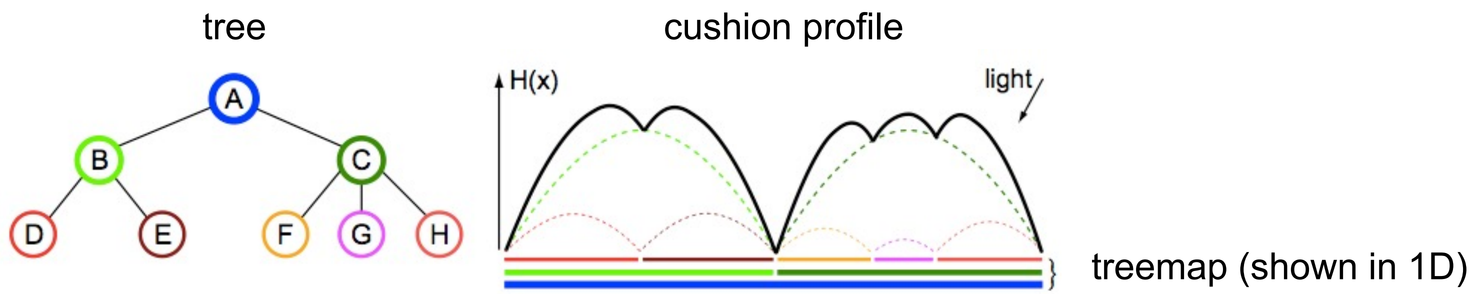
- 1 node = 1 rectangle
- child node rectangles: *nested* in the parent node rectangle (recursive subdivision)
- leaf rectangle *size* and *color* show data attributes
- edges: not drawn explicitly!
- very compact: tens of thousands of nodes on one screen! no pixel wasted
- aspect ratios are not very good; hierarchy depth unclear

How can we improve the basic idea?

Squarified Cushion Treemaps

Two extensions of basic treemaps

- enforce near-square aspect ratios during rectangle subdivision[1]
- use shading: cushion profiles to convey hierarchical structure[2]



For each rectangle r_i (except root)

- define 2D parabola $h_i(x,y)$ with height $H_i = f^{\,d}$ (d = depth of r_i in tree)

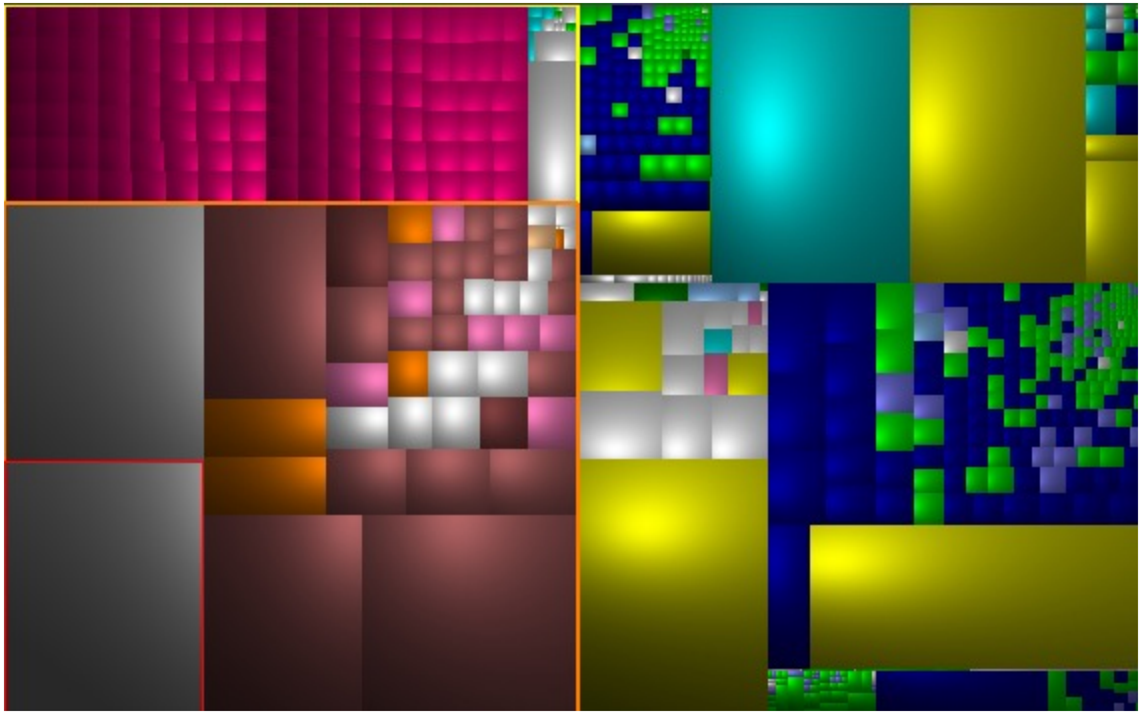
Compute global height $h = \Sigma_i\, h_i$

*F*or each image pixel $I(x,y)$

- $I(x,y) = shading\ (h(x,y))$
- use an oblique light source for better results

[1] M. Bruls, K. Huizing, J. J. van Wijk (1999) Squarified Treemaps, Proc. VisSym, 322-330
[2] J. J. van Wijk, H. van de Wetering (1999) Cushion treemaps: Visualization of hierarchical information, Proc. InfoVis, 135-142

Squarified Cushion Treemaps



ffmpeg C library

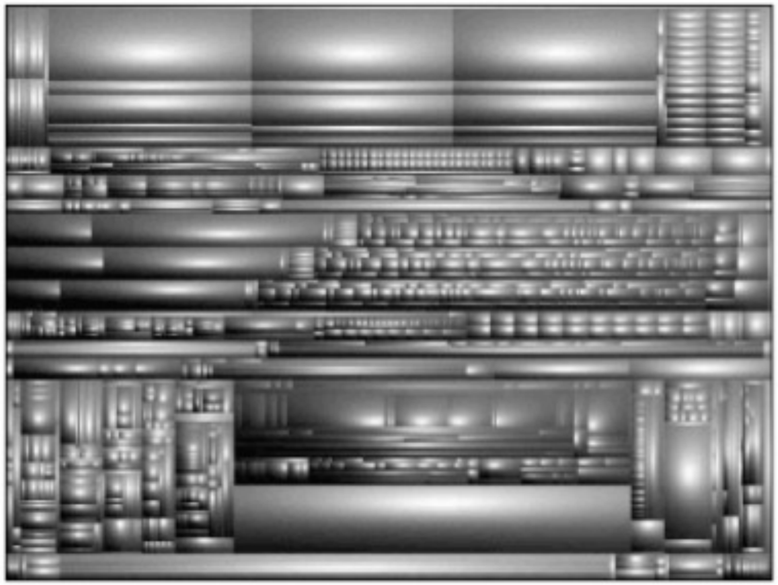
- rectangle borders are not *explicitly* drawn $\rightarrow$ gain space
- borders are *implicit* in the shading discontinuities
- discontinuity strength conveys tree depth levels
- near-square aspect ratio of cells allows easier size (area) comparison

SequoiaView tool (www.win.tue.nl/sequoiaview)

Squarified Cushion Treemaps

Comparison of methods

hard disk, ~30K files



Slice and dice layout
- unbalanced cell sizes (hard to compare)
- tree structure is not very clear

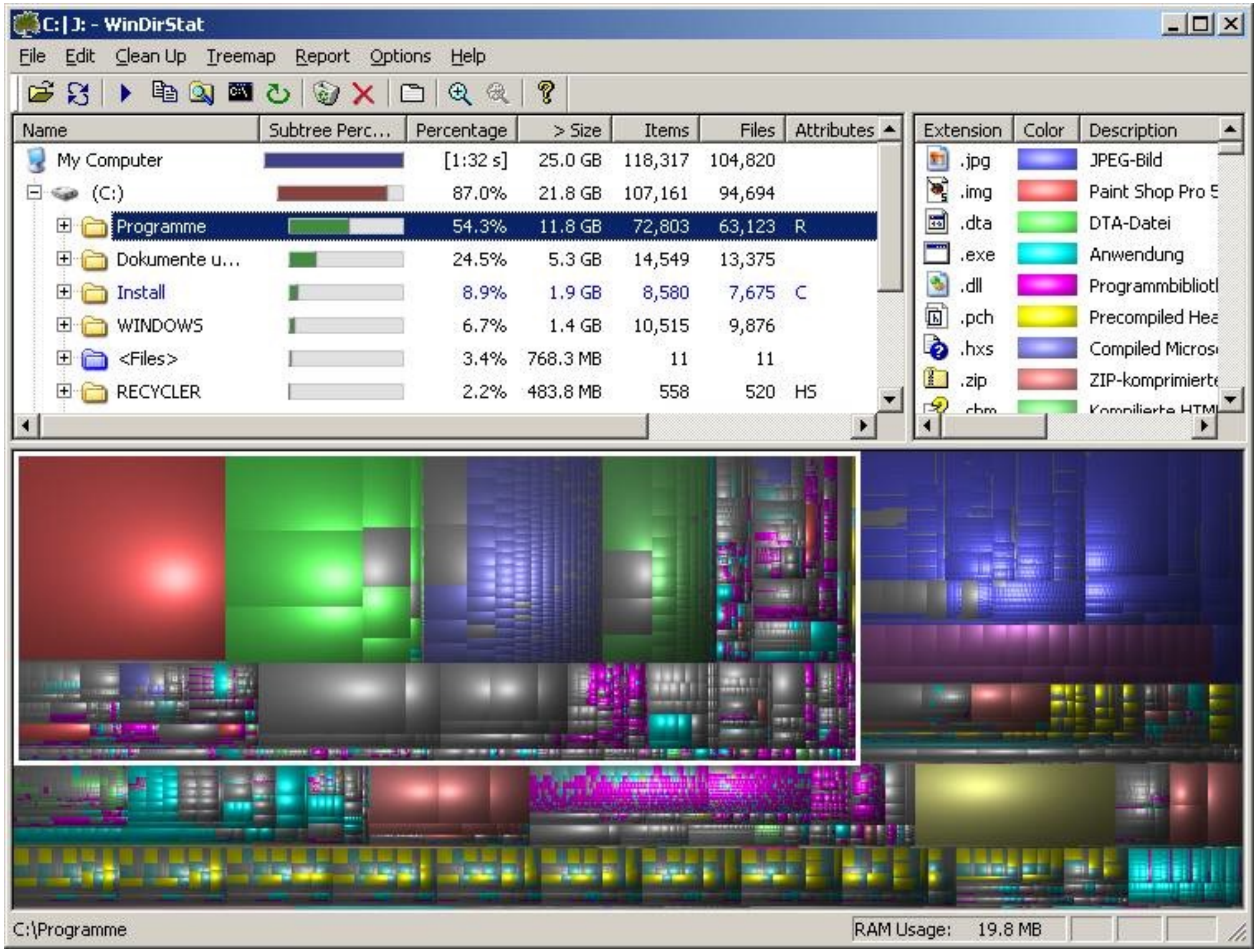
Squarified layout
- balanced cell sizes (easier to compare)
- tree structure is extremely salient

SequoiaView tool (www.win.tue.nl/sequoiaview)

Squarified Cushion Treemaps

Example 1: WinDirStat tool[1]

- visualize a file system (*e.g.* your hard disk)



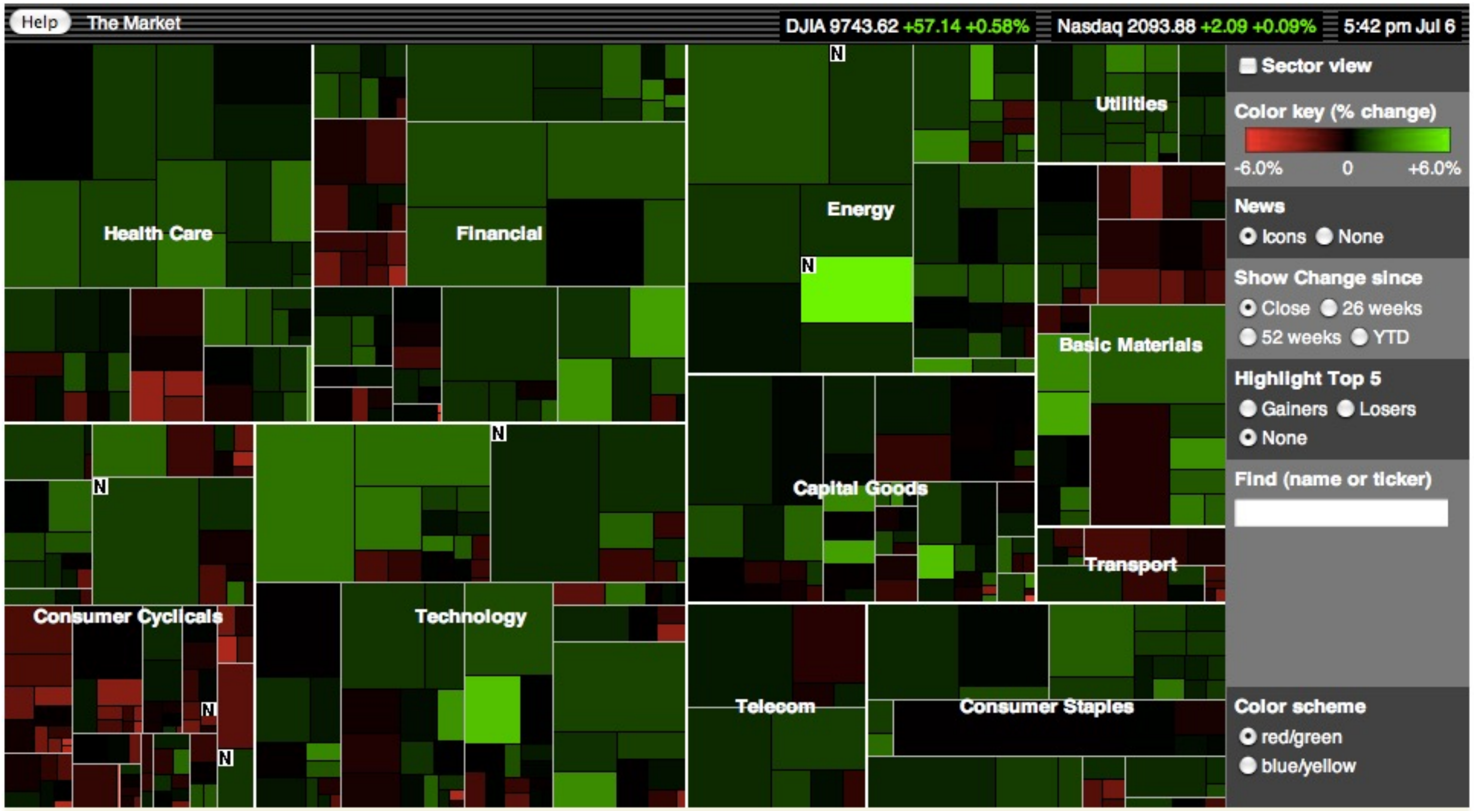
[1] http://windirstat.info

Squarified Cushion Treemaps

Example 2: Map of the Market[1]

Visualize stock exchange data online

- hierarchy: nesting of companies within sectors
- rectangle size: market capitalization
- color: gain (green) … loss (red)



[1] http://www.smartmoney.com/map-of-the-market

Squarified Cushion Treemaps

Example 3: Another Map of the Market[1]

Visualize stock exchange data online

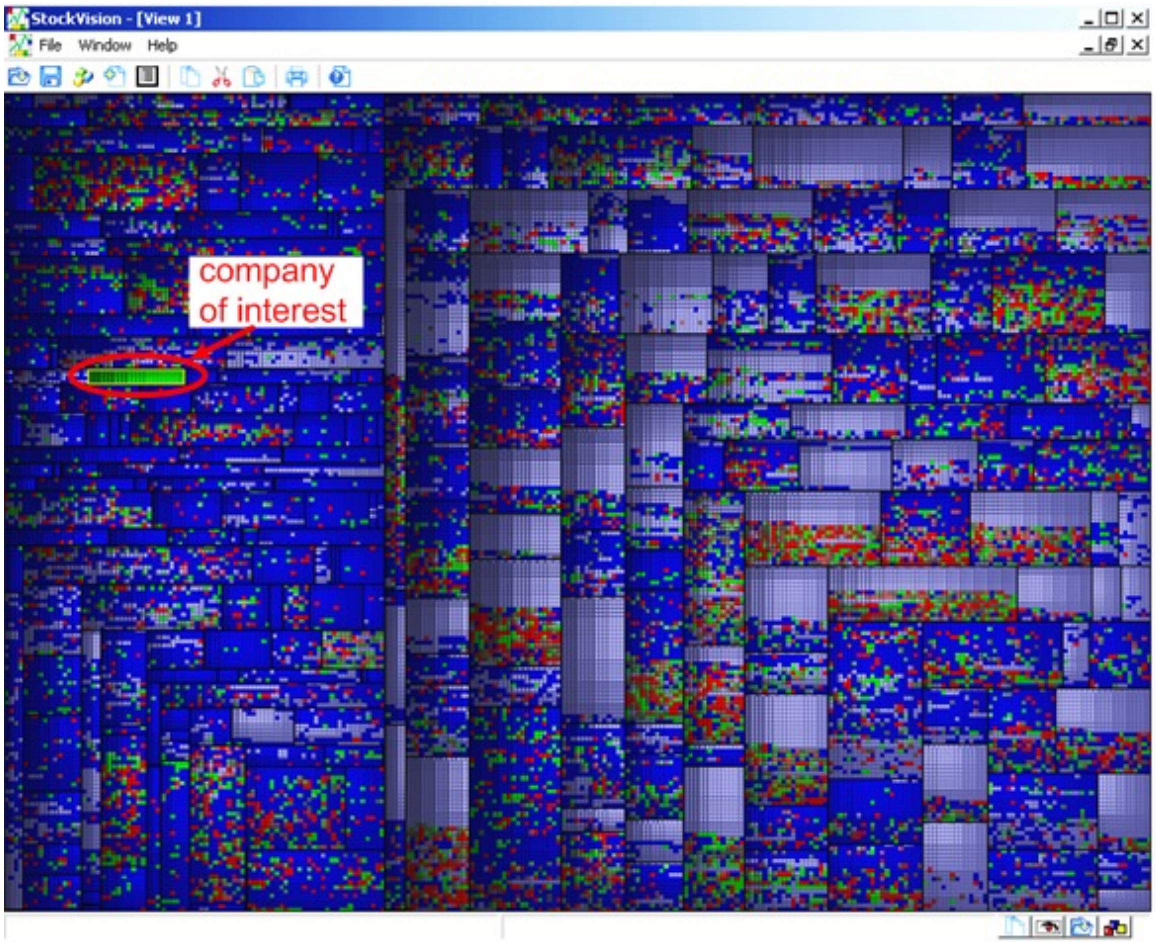
- hierarchy: nesting of companies within sectors
- rectangle size: market capitalization
- color: gain (green) … loss (red)

[1] http://www.finviz.com/map.ashx

Squarified Cushion Treemaps

Treemaps from tables

- recall the multiple-sorting idea for tables?
    - implicitly creates a *hierarchy* from a table (with just a few clicks to sort columns)
    - visualize hierarchy with a treemap

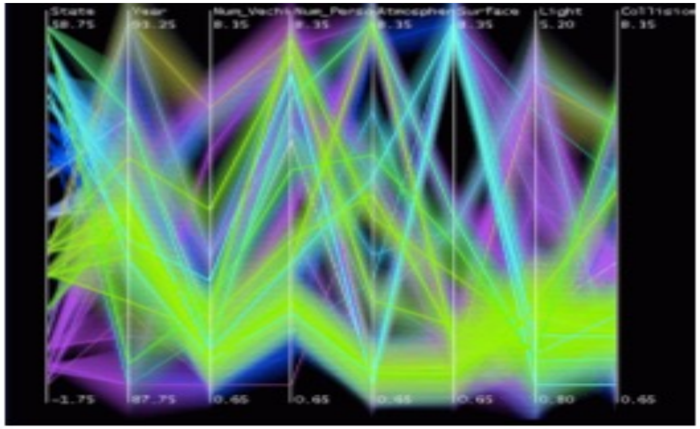


Stock exchange table
- three levels:
  - market sector
  - companies
  - prices / day over 1 month
- colors
  - red: daily loss
  - green: daily gain
  - light blue: unavailable data

Discovered a small emerging company with **steady growth**!

A. Telea (2006) Combining Extended Table Lens and Treemap Techniques for Visualizing Tabular Data, Proc. EuroVis

4. Parallel coordinates



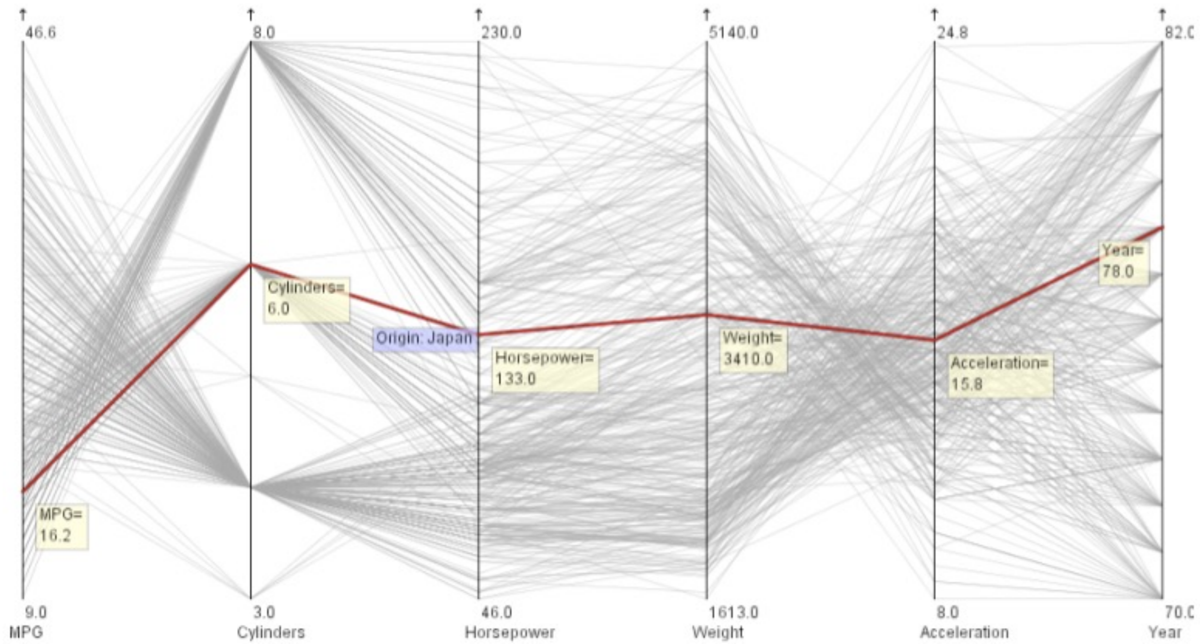
Parallel coordinates

Take again a table

- **rows:** car brands
- **columns:** car parameters (MPG, cylinders, horsepower, weight, acceleration, fabrication year)

Parallel coordinates

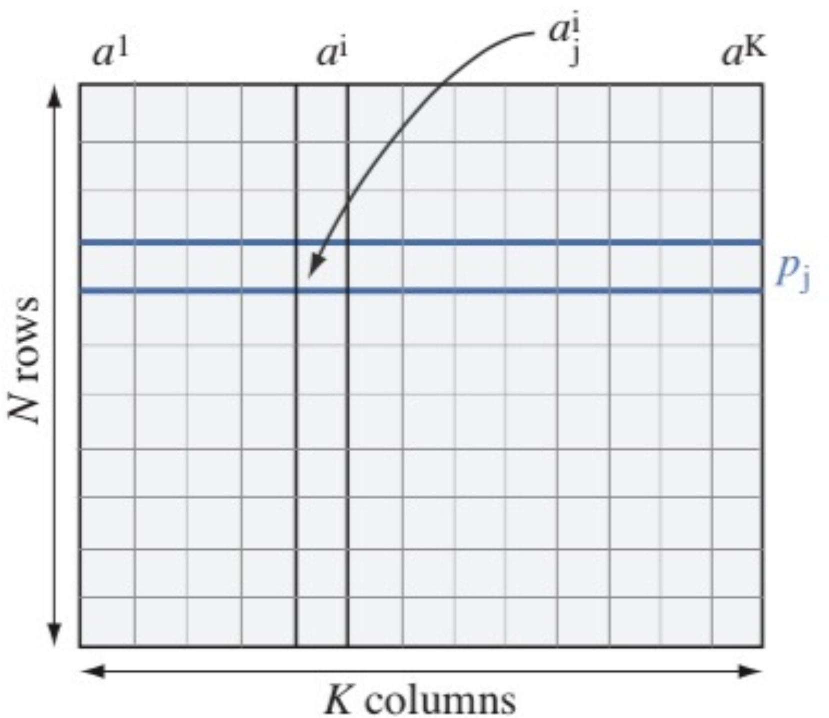
- table columns: different y axes
- table cells: points on their corresponding axes
- table rows: polylines connecting their points
- column correlations: ‘bundles’ of close lines



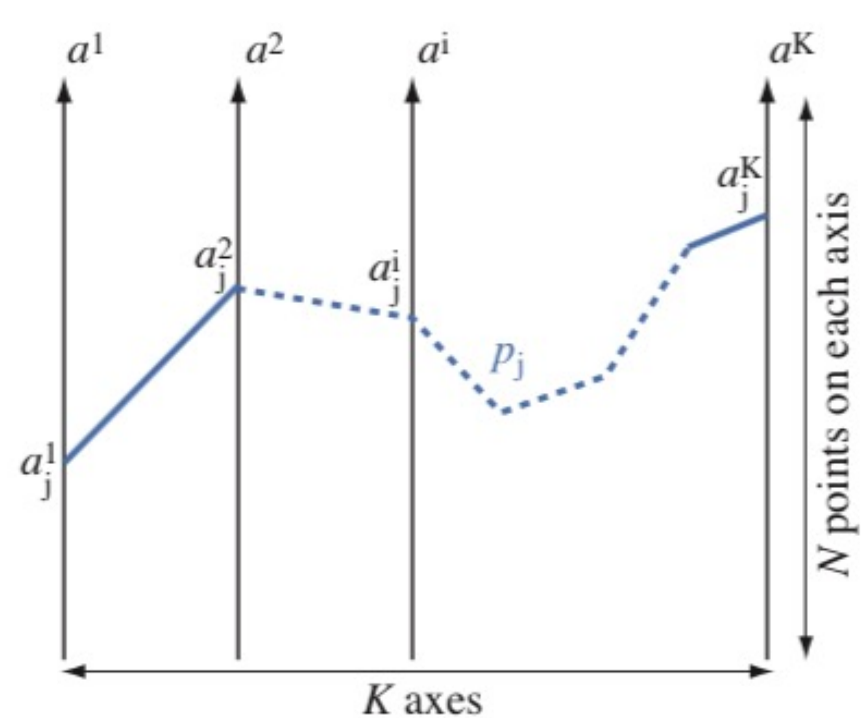
A. Inselberg (2009) Parallel Coordinates: Visual Multidimensional Geometry and its Applications. Springer
Implementations: Mondrian (theursus.de/mondrian), Prefuse (prefuse.org), Xdat (xdat.org), Xmdv (http://davis.wpi.edu/xmdv)

Tables *vs* parallel coordinates

Table plot



Parallel coordinates



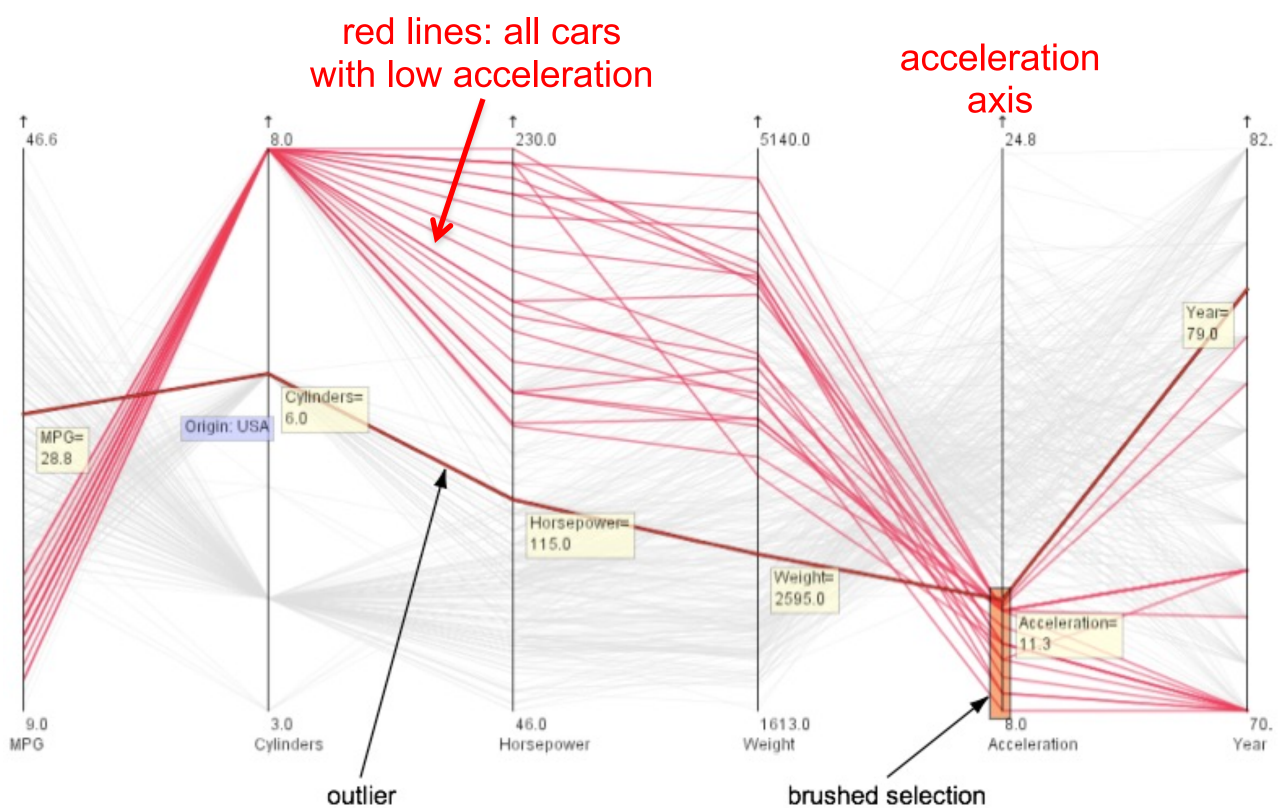
- row: horizontal line
- column: vertical line
- column-column correlation: not easy to see
- no overdraw/clutter

- row: skewed polyline
- column: vertical line (permuted values)
- column-column correlation: easy to see
- overdraw/clutter present

Parallel coordinates

Selection

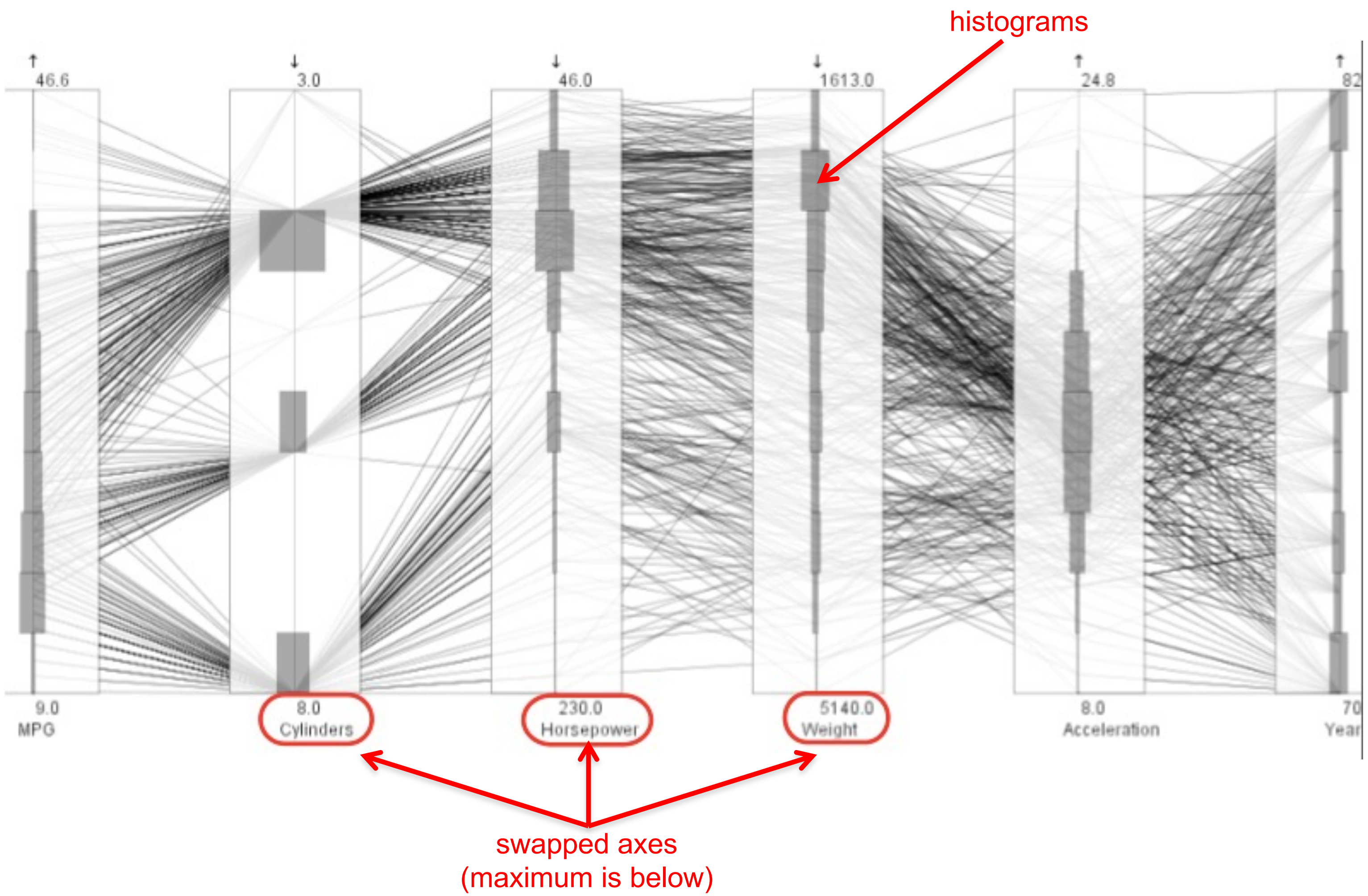
- use mouse to select attribute ranges on axes
- highlight all rows (lines) passing through selection
- supports queries such as
  - show all cars with a low acceleration
  - find what attributes (e.g. MPG, cylinders, weight, …) low-acceleration cars have



Parallel coordinates

Enhancements

- **permute** axes (horizontally) and **swap** their direction (vertically) to minimize line crossings
- add **histograms** on axes to show #rows per unit-data value

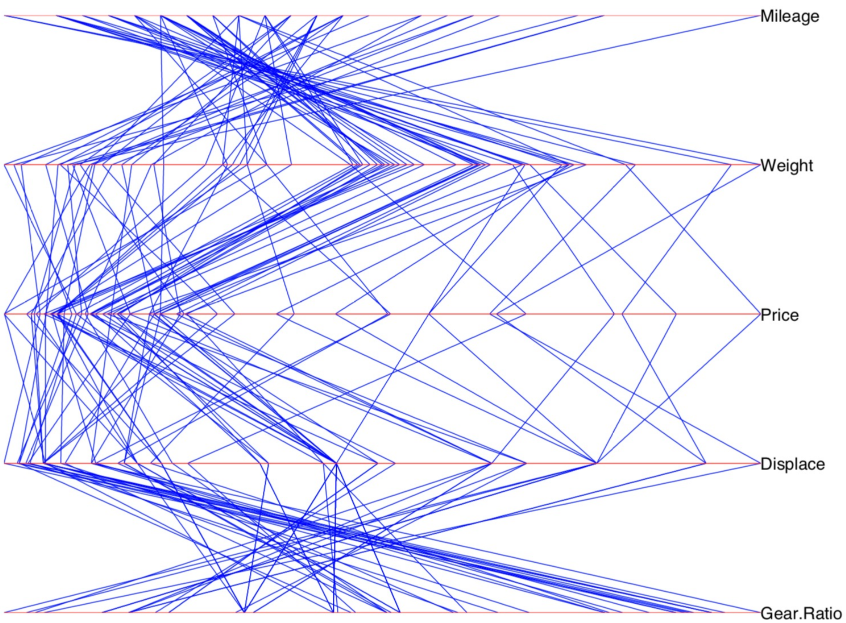


Smooth parallel coordinates

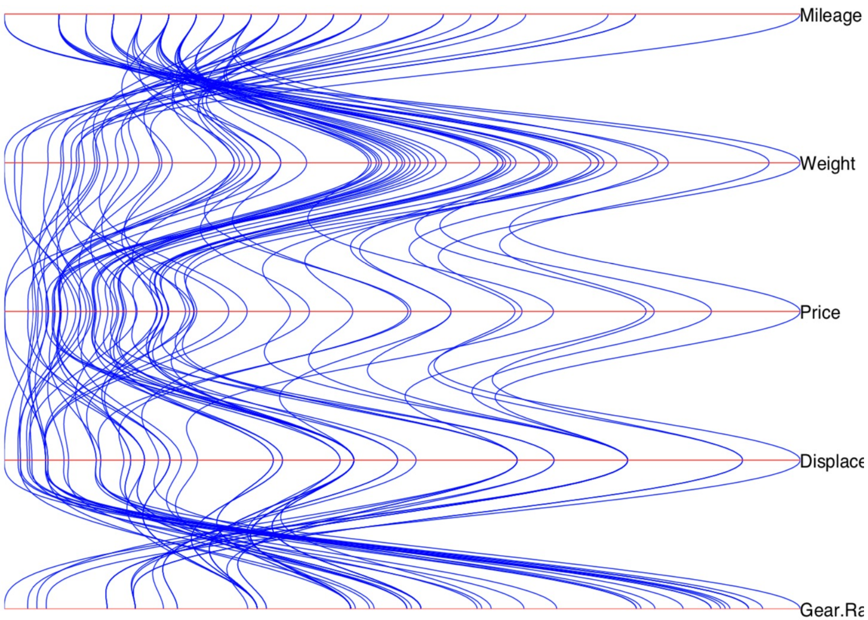
Enhancements

- use **curves** (splines) instead of polylines
- reduces visual clutter
- makes visually following a sample (curve) easier

classical parallel coordinates



smooth parallel coordinates



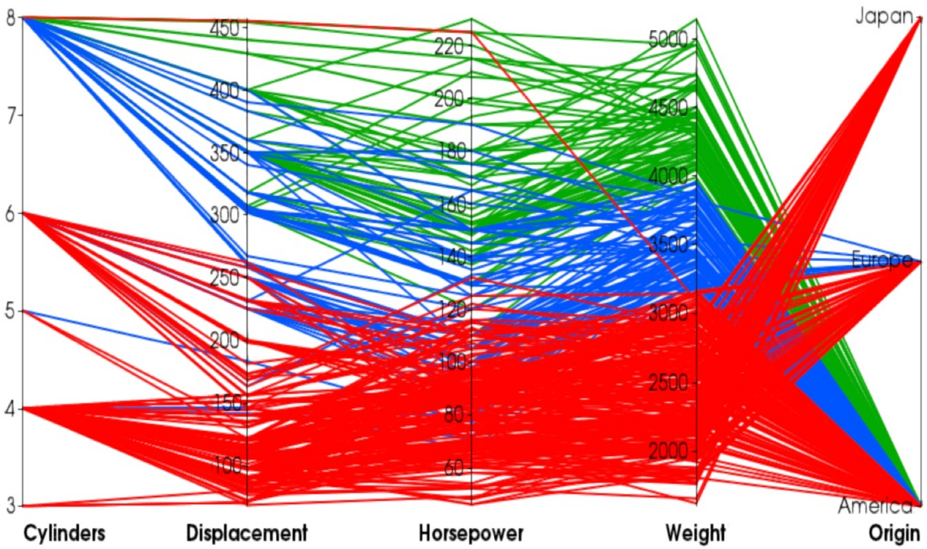
M. Rida *et al* (2006). Multivariate continuous data – Parallel Coordinates. Graphics of Large Datasets: Visualizing a Million. Springer

Bundled parallel coordinates

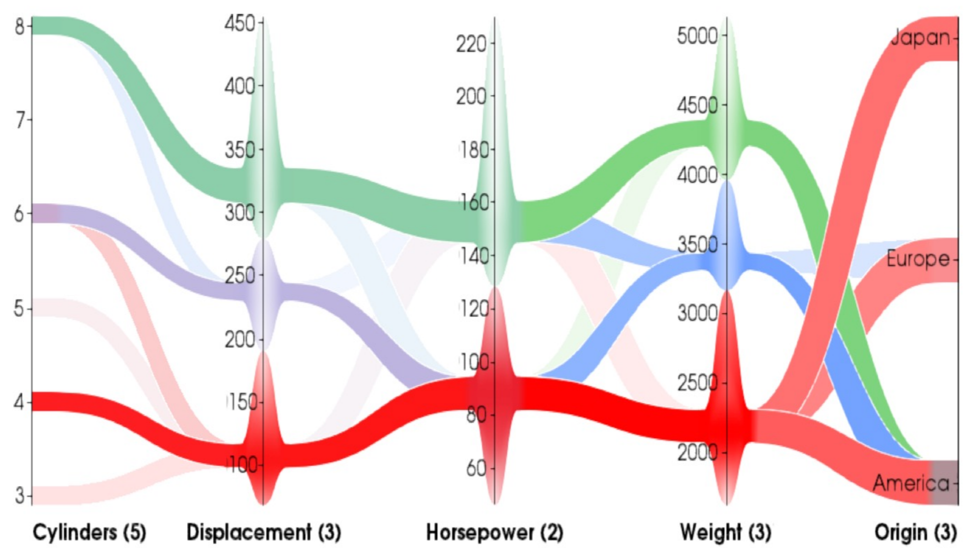
Enhancements

- use **curves** (splines) instead of polylines (as in smooth parallel coordinates)
- bundle the curves (as in graph bundling)
- massively simplifies the visualization, reduces clutter
- following groups of similar samples (close curves) is much easier

classical parallel coordinates



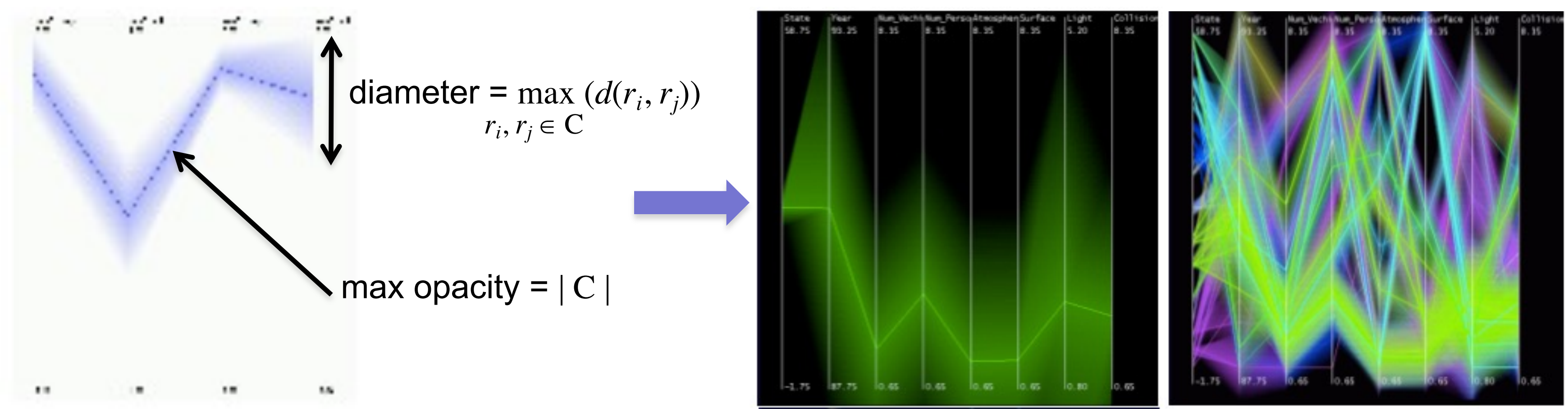
bundled parallel coordinates



G. Palmas *et al* (2014). An Edge-Bundling Layout for Interactive Parallel Coordinates. Proc. PacificVis

Hierarchical parallel coordinates

- reduce clutter for very large datasets ($10^6..10^9$ rows)
- hierarchically cluster rows r_i
  - (1) create a cluster $C_i = \{r_i\}$ for each row. Set $S = \{ C_i \}$
  - (2) find two most similar clusters C_i, C_j using an Euclidean distance metric $d(r_i, r_j) = \Sigma_k (r_{ik}-r_{jk})^2$
  - (3) build parent cluster $C = (C_i, C_j)$, $S = S \setminus (C_i \cup C_j) \cup C$
  - (4) repeat from step 2 until $S = \{$ root cluster $\}$
- select a 'cut' K in the cluster tree S at desired level-of-detail
- visualize each cluster $C \in K$ with an opacity band which encodes cluster size and diameter



root cluster

finer-level cut

* Y. Fua, M. Ward, E. Rundensteiner, Hierarchical Parallel Coordinates for Exploration of Large Datasets, IEEE InfoVis, 1999
See also the Xmdv tool, http://davis.wpi.edu/xmdv

Putting it all together

Low-dimensional data visualization

- easy-to-use tool: SPLOMs, parallel coordinates, and projections (next module)
- Java implementation (runs anywhere), simple text input format

http://www.cs.rug.nl/svcg/DataVisualizationBook/Sp11

Summary: Low-dimensional data visualization

For what

- datasets with many samples *N* but few (2..10) dimensions *n*

Main design idea

- allocate one visual variable for one..a few dimensions

Techniques

- scatterplots, scatterplot matrices
- table lenses
- table-tree duality
- icicle plots, treemaps
- parallel coordinates

Open challenge: What to do with many dimensions?