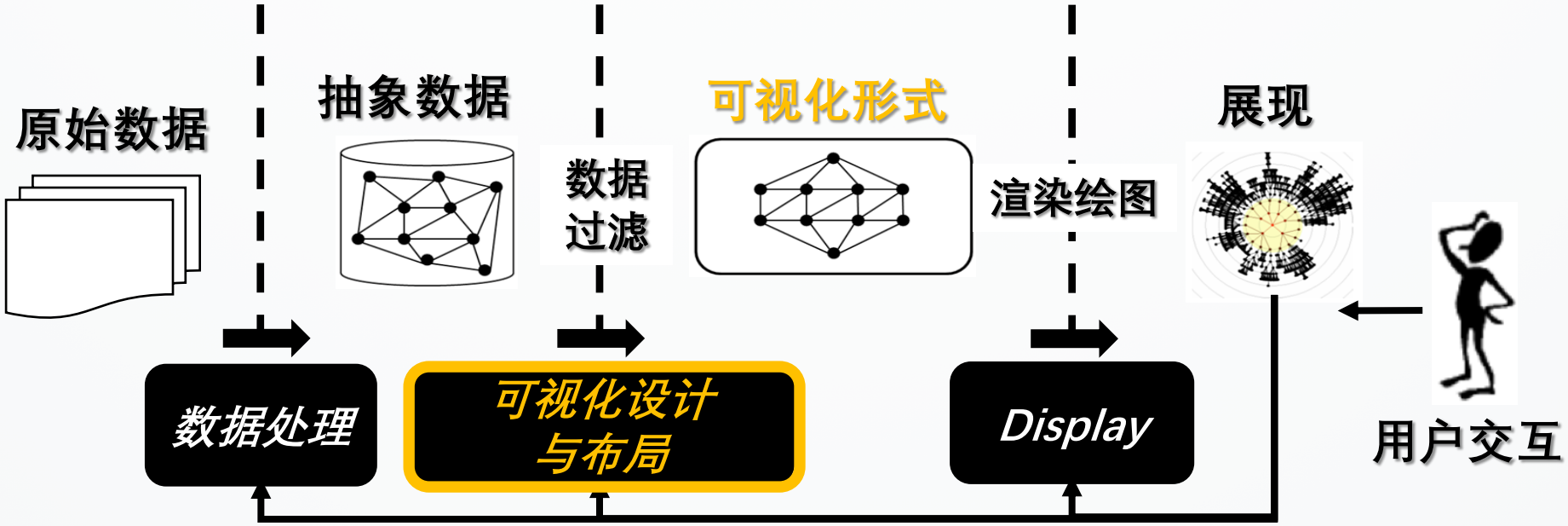
第四讲：针对不同数据类型的可视化技术

信息可视化

曹楠（教授），石洋（研究员）
https://idvxlab.com
同济大学

智能大数据可视化实验室

怎样对数据进行可视化？



信息可视化参考模型

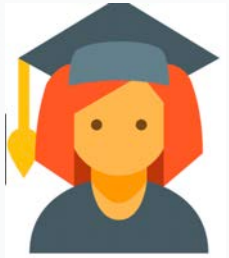
课程大纲

- 数据基础
- 多维度数据的可视化
- 树的可视化
- 图的可视化

课程大纲

- 数据基础
- 多维度数据的可视化
- 树的可视化
- 图的可视化

数据的维度



Student

Machine Learning

Data Mining Visualization

[22, Male, 3000, 20, 30, 5]

Age Sex scholarship Skills

- 数据的维度，是数据中用于描述数据元素的各种属性。例如，在学生数据集中，一个学生可以通过她的年龄、性别、助学金的额度、以及学生在相关课程上的技能，通过花费在课程上的时间来衡量，等
- 真实世界中的数据一般都是多维度的

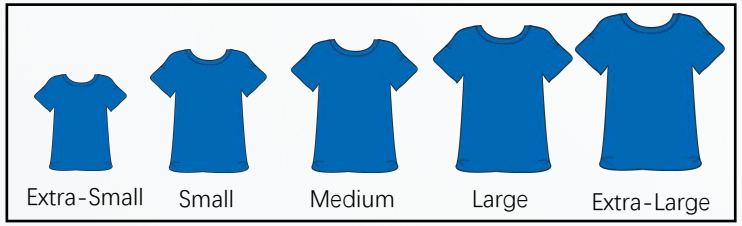
数据维度的类型

Nominal Data

Point	airport	town	mine	capital
Line	river	road	boundary	pipeline
Area	orchard	desert	forest	water

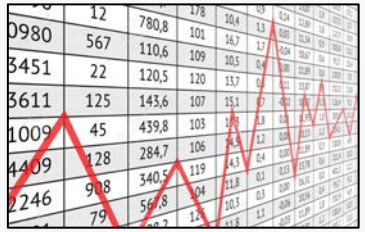
- 分类属性（Nominal ，Categorical)：该类型属性的取值代表了数据的类别，且相互间是无法排序的

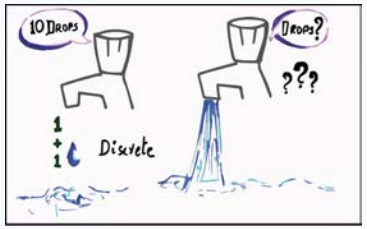
Ordinal



- 有序属性（Ordinal）：该类型属性的取值是有顺序的

Numerical





- 数值属性（Ordinal）：该类型属性的取值是数字。根据其取值的特点可以进一步分为 离散数值属性及连续的数值属性

一维数值属性的统计特征

- 均值（Mean）
- 中位数（Median）
- 方差（Variance）
- 标准差（Standard Deviation）

$$\bar{x} = \frac{\sum_{i=1}^{N} x_i}{N} = \frac{x_1 + x_2 + \cdots + x_N}{N}.$$

$$Q_{\frac{1}{2}}(x) = \begin{cases} x'_{\frac{n+1}{2}}, & \text{if } n \text{ is odd.} \\ \frac{1}{2}\left(x'_{\frac{n}{2}} + x'_{\frac{n}{2}+1}\right), & \text{if } n \text{ is even.} \end{cases}$$

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2 = \left(\frac{1}{N}\sum_{i=1}^{N} x_i^2\right) - \bar{x}^2,$$

$$SD = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}$$

数据元素之间的差异性

- 数据元素的差异性是通过数据元素在各个维度上的取值的差异性来加以度量的

- 整个数据集所有元素两两之间的差异性可以用差异矩阵来表示：
  - 差异矩阵是一个对称矩阵
  - 差异矩阵的每一行及每一列代表数据集中的一个数据元素
  - 差异矩阵中的每一个取值，代表对应元素之间的数据距离

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & & \\ d(n,1) & d(n,2) & \cdots & \cdots & 0 \end{bmatrix}$$

Dissimilarity Matrix

- 不同类型的数据维度之间有不同的距离计算方法

数据元素之间的差异性

- **分类属性** 之间的距离 通过“不匹配率”进行衡量
  - P 在某一分类属性上所有可能的取值的总数
  - m 代表数据元素在该属性上的取值相同情况的数目

$$d(i,j)=\frac{p-m}{p}$$

- **数值属性** 之间的距离 通过 欧式距离及其推广 进行计算
  - 曼哈顿距离（L1） - 数据在 **一维尺度** 上的差异
  - 欧几里得距离（L2） - 数据为 **二维平面** 上的差异
  - 明科夫斯基距离（Lp) - 数据为 **多维空间** 中的差异

$$d_{CB}=\sum_{i=1}^{d}|P_i-Q_i|$$

$$d_{Euc}=\sqrt{\sum_{i=1}^{d}|P_i-Q_i|^2}$$

$$d_{Mk}=\sqrt[p]{\sum_{i=1}^{d}|P_i-Q_i|^p}$$

课程大纲

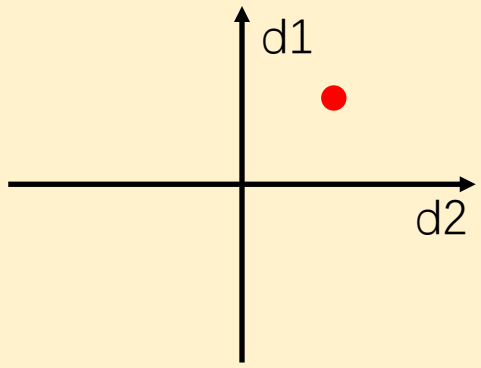
- 数据基础
- 多维度数据的可视化
  - 基于不同坐标系的可视化方法(Coordinate Systems)
  - 基于像素的可视化方法 (Pixel Oriented)
  - 基于图标的可视化方法（Icon Based)
  - 基于网格的分视图展示方法（Small Multiple）
- 树的可视化
- 图的可视化

课程大纲

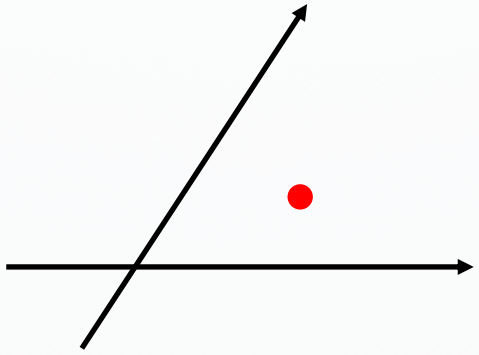
- 数据基础
- 多维度数据的可视化
  - 基于不同坐标系的可视化方法(Coordinate Systems)
  - 基于像素的可视化方法 (Pixel Oriented)
  - 基于图标的可视化方法（ Icon Based)
  - 基于网格的分视图展示方法（ Small Multiple ）
- 树的可视化
- 图的可视化

基于不同坐标系的可视化方法

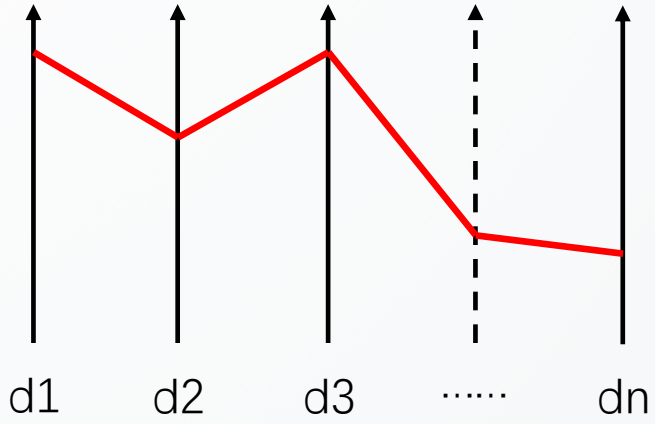
- 常用的三种不同可视空间



正交投影空间



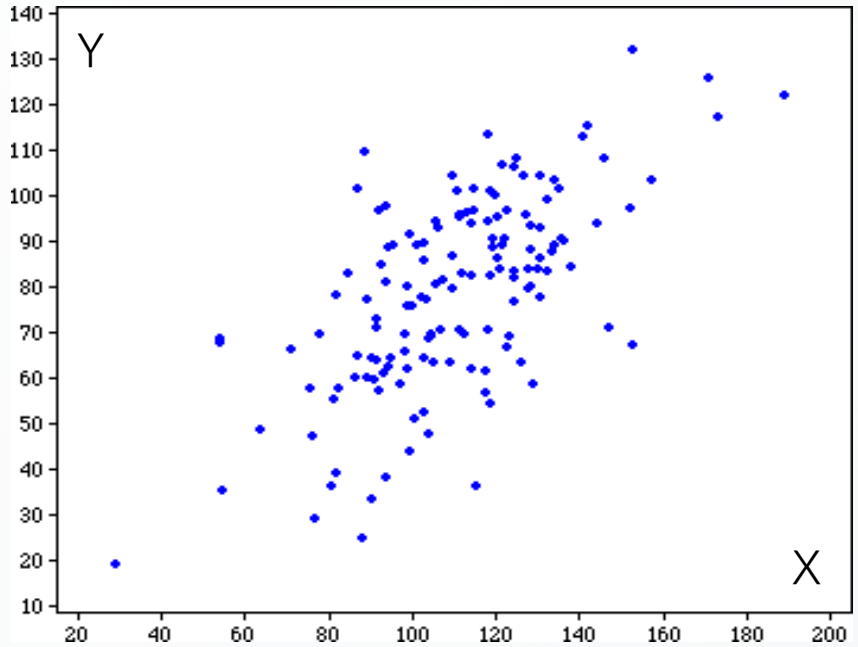
非正交投影空间

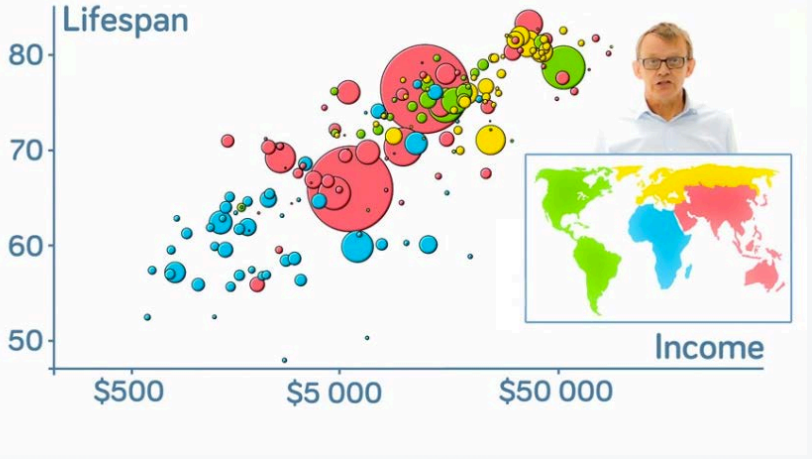


平行坐标轴系统

基于不同坐标系的可视化方法

- **正交投影空间 – 散点图（Scatter Plot）**
  - 直接利用笛卡尔坐标系显示多维度数据，可以同时表示数据元素在坐标轴所确定的两个维度上的分布
  - 每一个点代表数据集中的一个数据元素
  - 点的位置，由其在对应维度上的取值所决定
  - 可以通过点的大小、颜色等属性展示其他维度的数据信息，所得到的可视化也被称为“气泡图”，气泡图最多能够显示四个维度的信息
  - 更多维度怎么办？

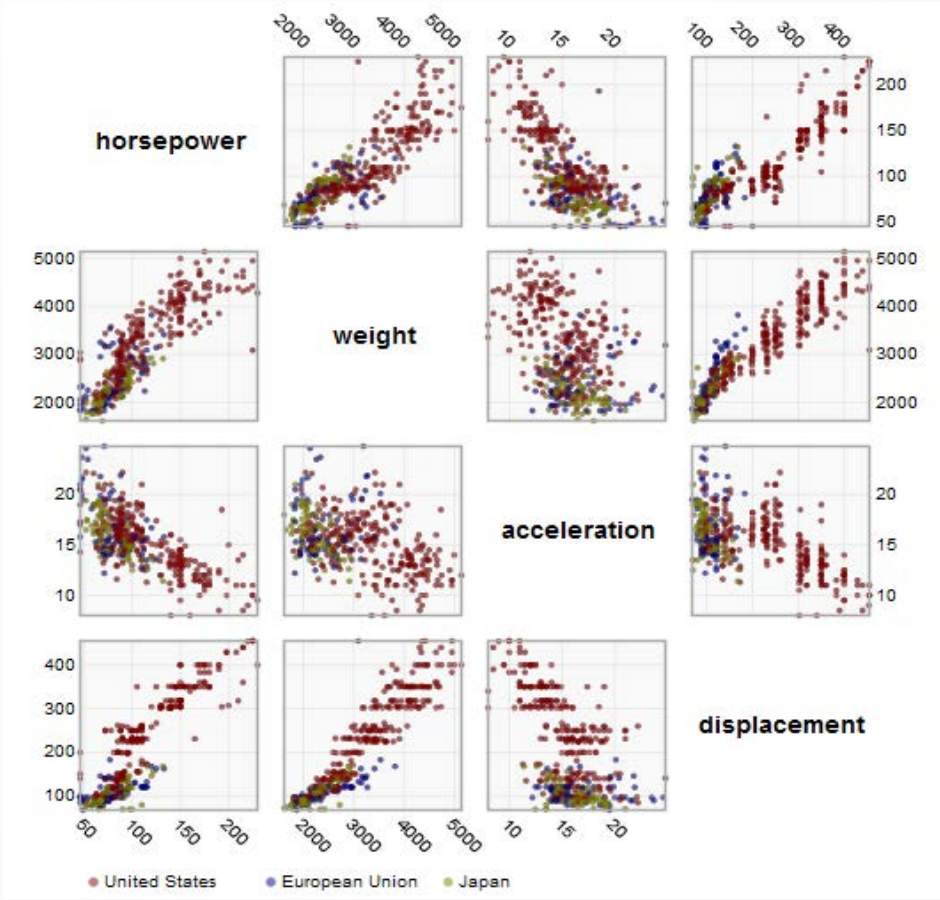




基于不同坐标系的可视化方法

- **散点图矩阵（Scatter Plot Matrix）**

  - 矩阵的每一行、每一列均为数据的一个维度
  - 矩阵的中的元素为 一个 散点图，散点图的横轴与纵轴所对应的维度分别由其所对应的矩阵的行与列所决定
  - 矩阵是对称的
  - N个维度 对应着 N (N - 1)/ 2 个散点图，因此无法显示高纬度数据



基于不同坐标系的可视化方法

- **正交投影空间 – PCA 降维**

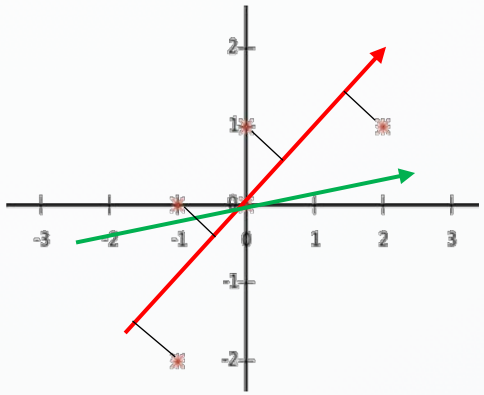
  - 投影的方法是将高维度数据 投影到低维度空间， 能够在低维度空间展示数据在所有维度上的整体分布情况，但是无法明确的展现有 实际意义的 坐标信息

  - PCA （Principle Component Analysis）即主成分分析，是一种常见的正交投影方法

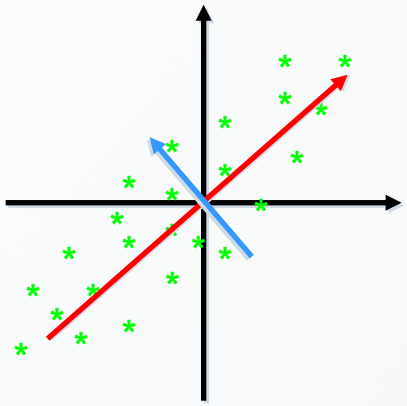
  - 该算法旨在将高维度数据投影到低纬度空间，同时数据在低维空间的分布能够最大程度的保持数据在原有高维空间中的差异

  - 当被投影在一维空间是，数据元素的差异可以用 **方差** 来计算，当被投影在二维空间时，可以用 **协方差** 来计算

  - 用 PCA 进行降维就是在空间中选择两个相互正交的投影方向，使得当数据投影到 由着两个方形所构成的空间中时，他们的协方差是最大的



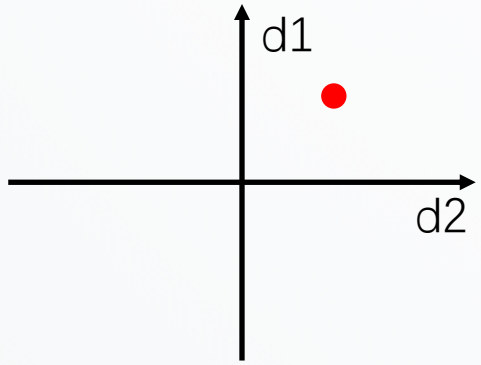
映射到1维空间就是选择一个方向进行投影，使得数据在该方向上的分布的方法最大（例如 红色的方向）



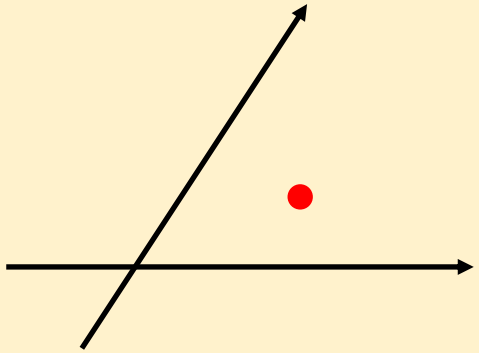
映射到2维空间就是选择连个相互正交的向进行投影，使得数据在该方向上的分布的协方差最大（例如 红色 与 蓝色的方向）

基于不同坐标系的可视化方法

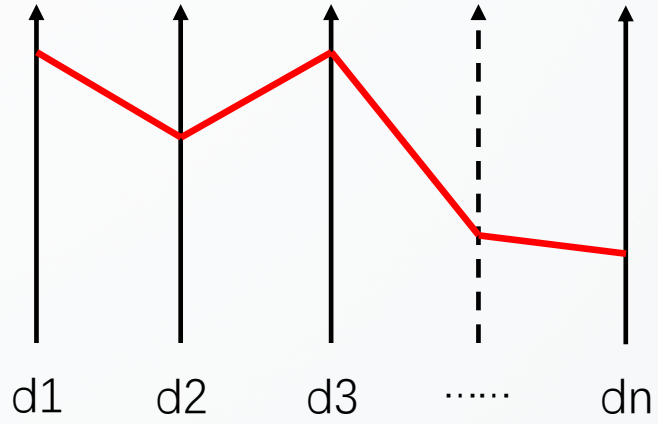
- 常用的三种不同可视空间



正交坐标系



非正交投影



平行坐标轴系统

基于不同坐标系的可视化方法

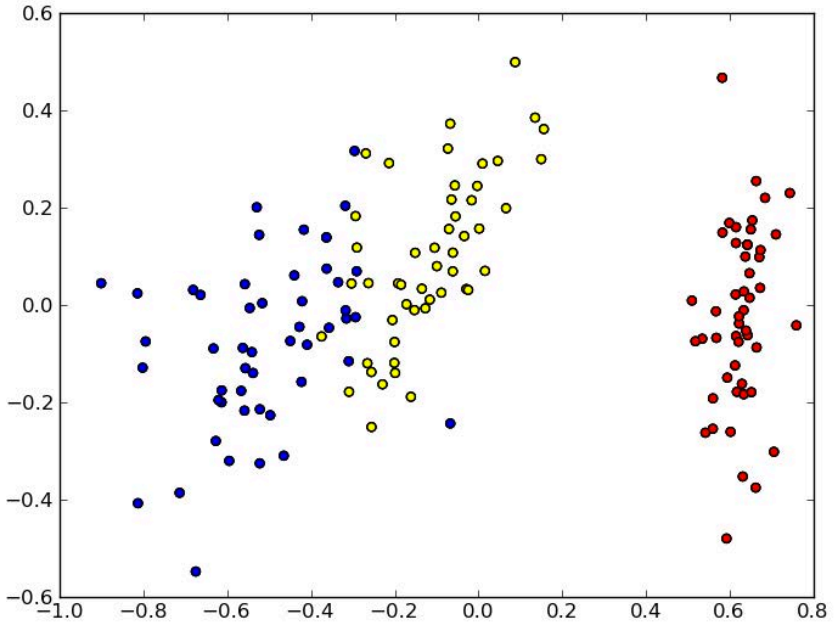
- **非正交投影空间 – MDS**

  - 将高纬度数据根据特定目标及限制条件投影到低纬度空间之上，投影过程并不保证投影方向相互垂直，而是有限确保能够达到目标约束及条件

  - MDS（Multidimensional Scaling）即相似度结构分析，便是可视化领域经常使用的针对多维度数据的投影方法

  - MDS 的目标是 尽可能在低维度空间中保持数据元素在高维度空间中的两两距离，其优化目标为右边的目标函数

  - 投影过程不涉及具体的投影方向，是对数据整体在不同空间中尺度上的调整



$$min \sum_{i<j} \mu_{ij}(d_p(x_i, x_j) - d_m(f(x_i), f(x_j)))^2$$

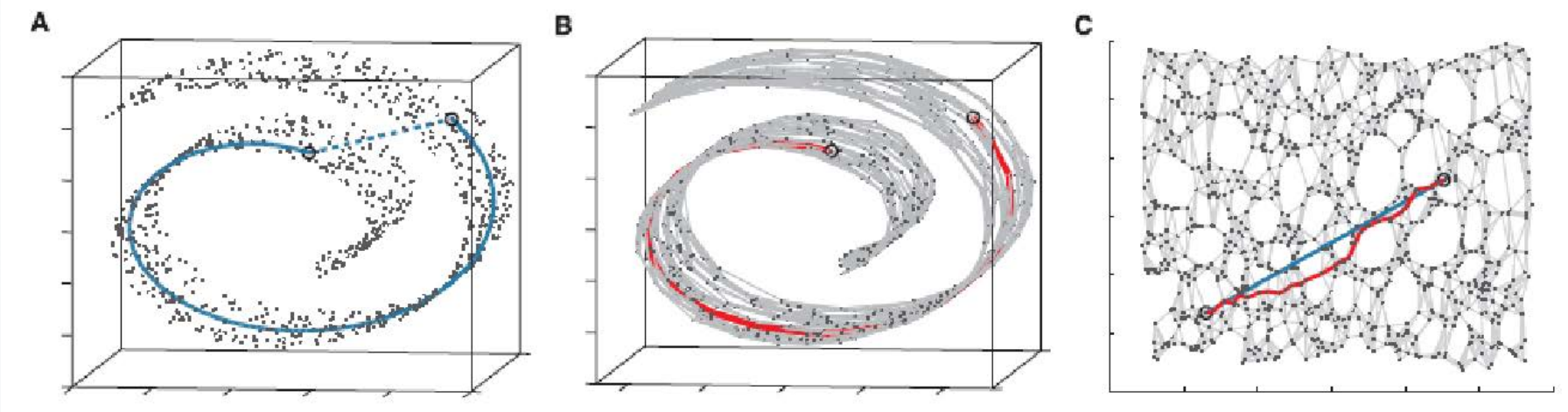
数据点X在高维度空间中的距离

数据点X 映射低维度空间中的距离

基于不同坐标系的可视化方法

- **非正交投影空间 – ISO Map**

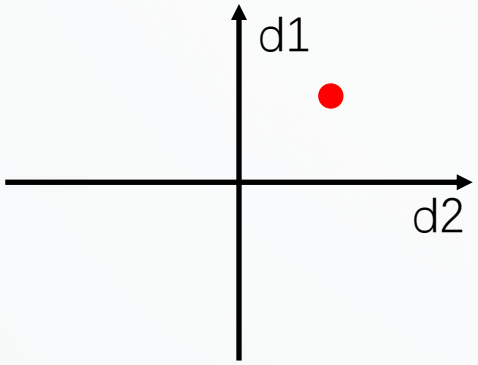
  - ISO Map 是对 MDS 在距离测度上的扩展
  - 用测地线距离（图 A 中的蓝色实曲线）取代了欧式距离（图 A 中的蓝色虚直线），从而确保了当数据在高维度空间不规则分布时 ，仍然能够正确的描绘数据元素之间的相关性 （图C）



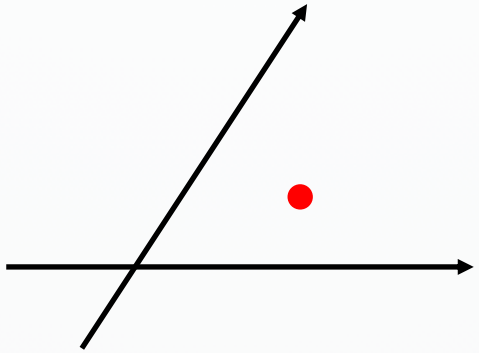
测地线距离　　近似计算方法　　投影展开后的结果

基于不同坐标系的可视化方法

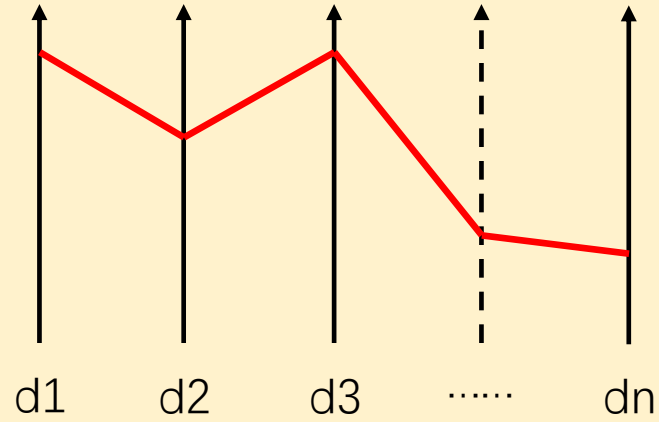
- 常用的三种不同可视空间



正交坐标系



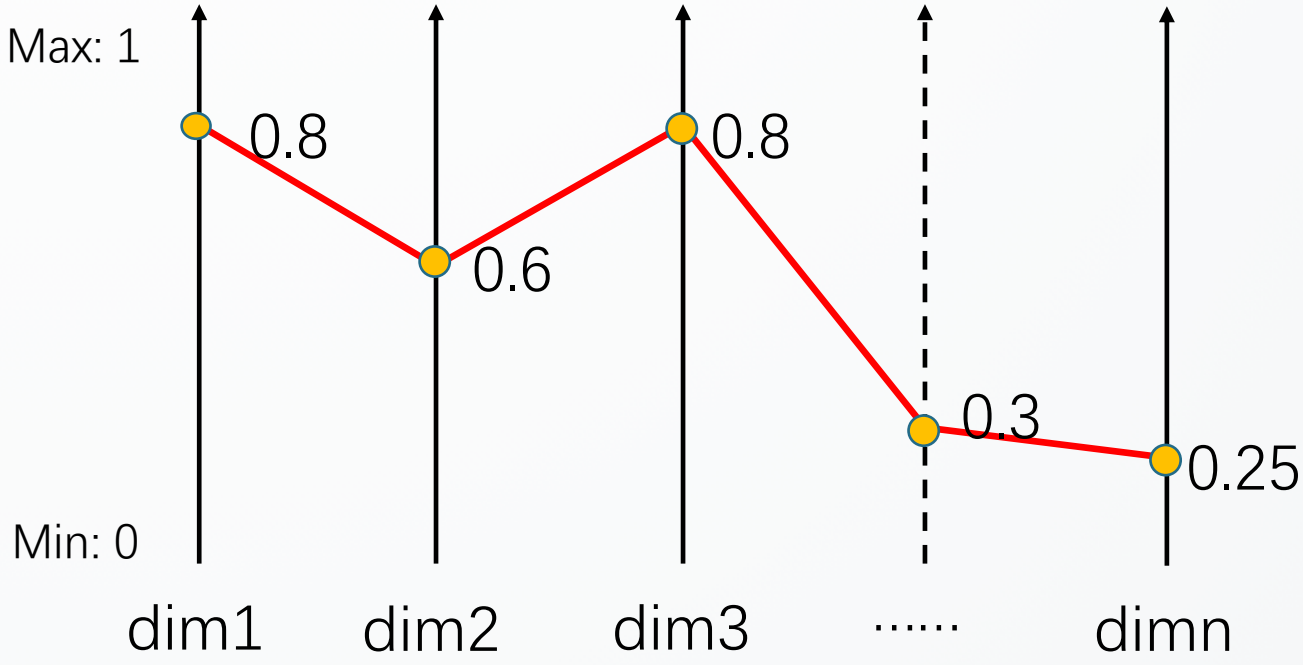
非正交投影



平行坐标轴系统

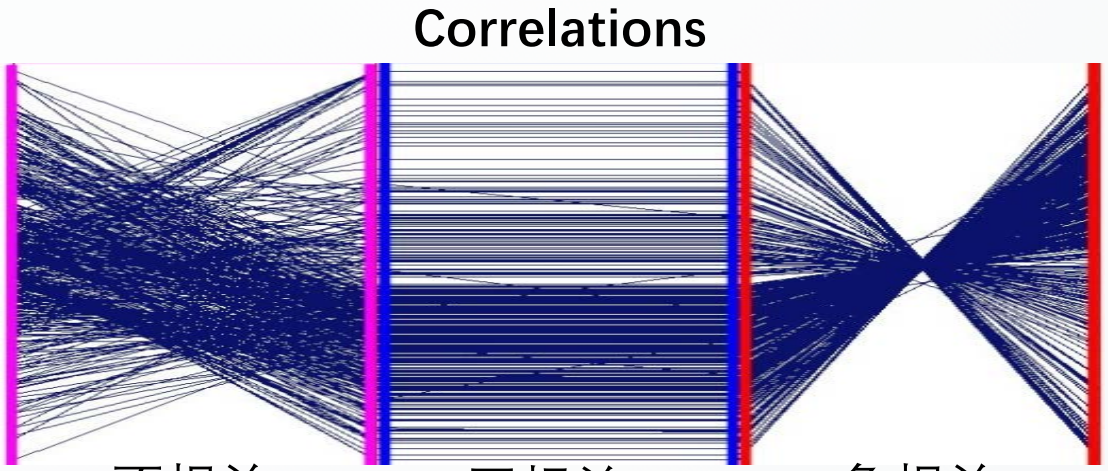
基于不同坐标系的可视化方法

- 平行坐标系
  - 数据的不同维度被显示为平行的坐标轴
  - 数据元素被表示为一根折线
  - 折线与数据轴的交点为数据元素在该维度的取值

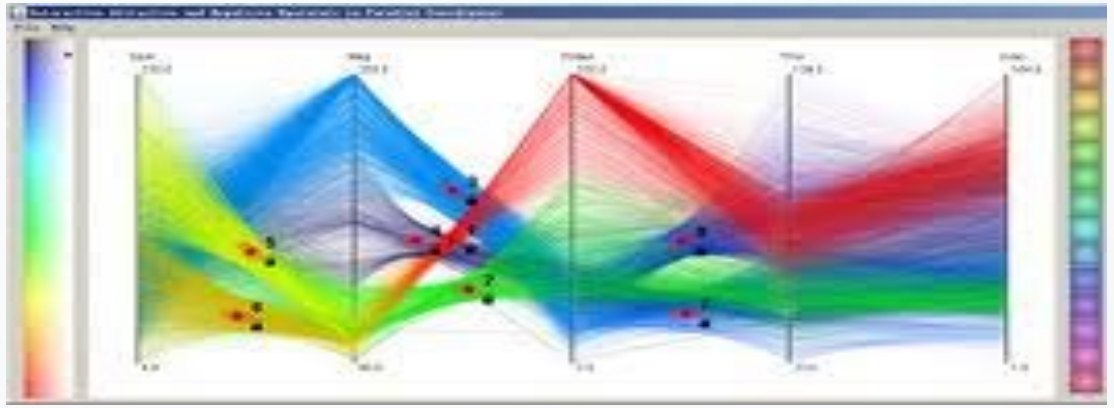


基于不同坐标系的可视化方法

- 平行坐标系中的数据模式
  - 数据的相关性
    - 正相关 – 平行线
    - 负相关 – 交叉线
    - 不相关 – 杂乱无章的交线
  - 数据中的聚类关系

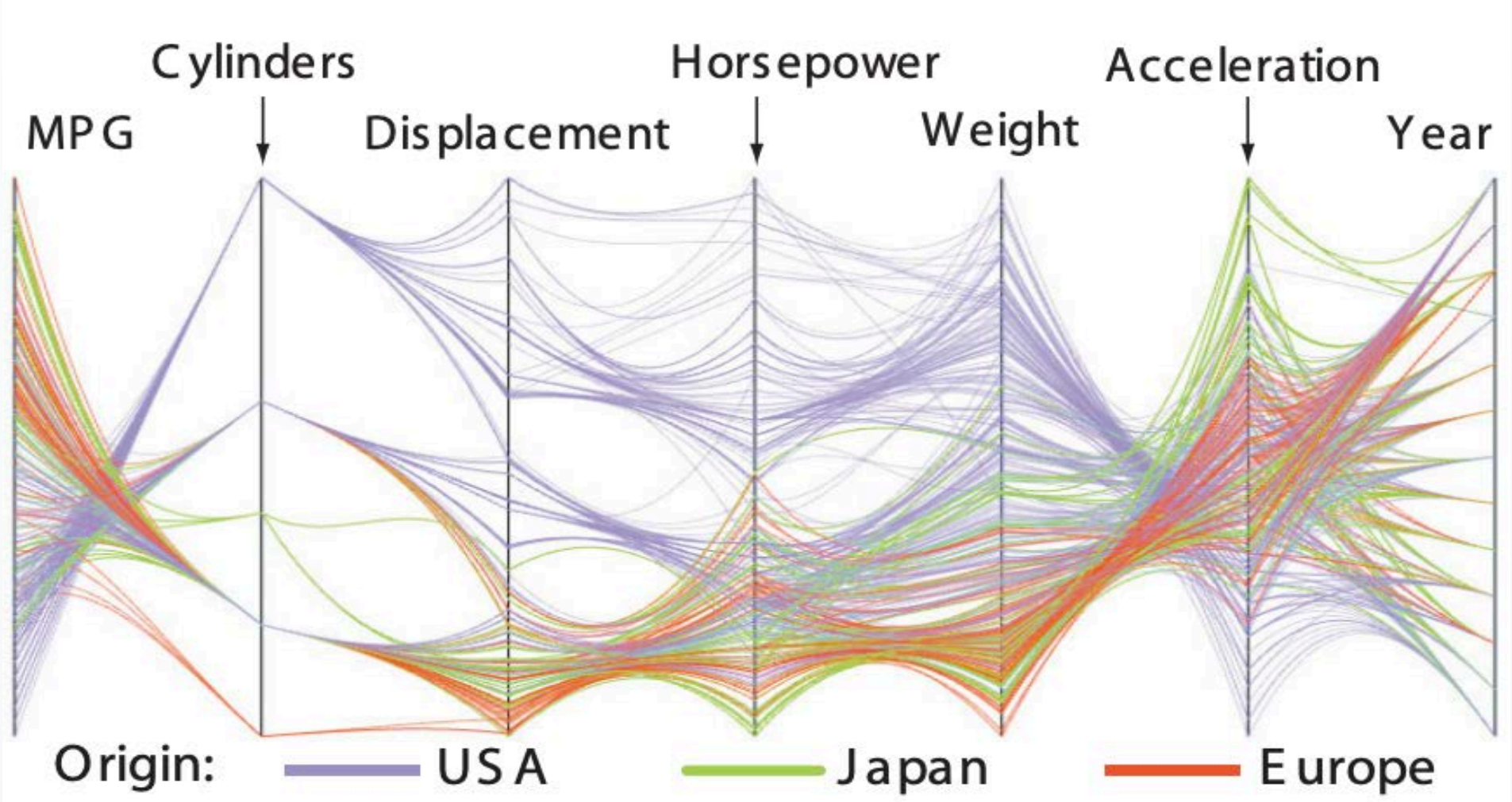


不相关　　正相关　　负相关



聚类信息

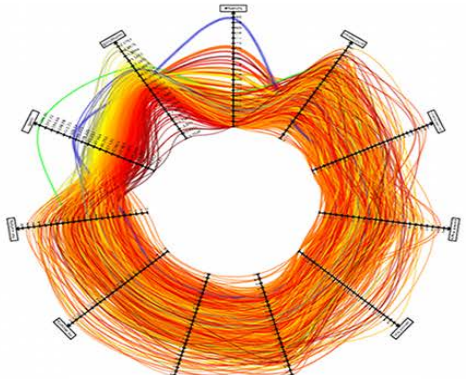
基于不同坐标系的可视化方法



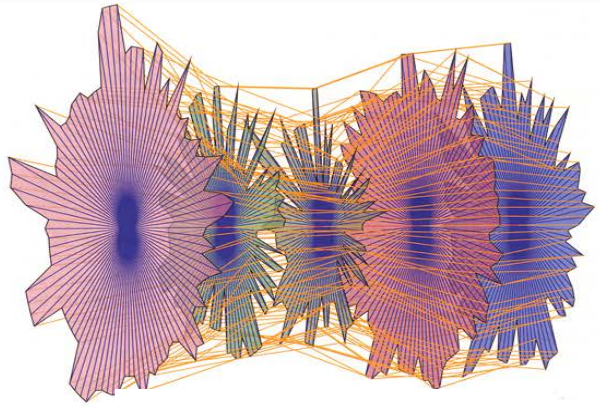
Cars dataset: 392 data items, 7 dimensions

基于不同坐标系的可视化方法

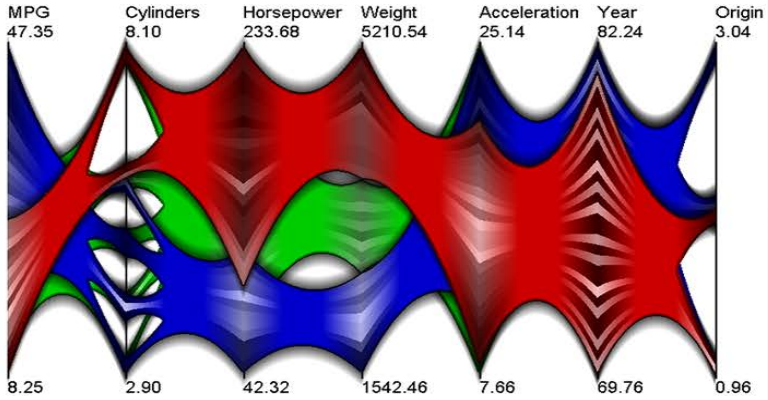
- 平行坐标轴有很多设计上的变化



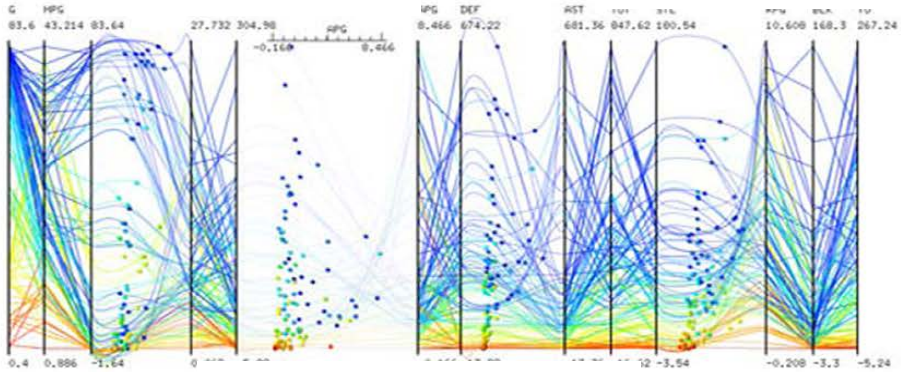
(Homan,1977)



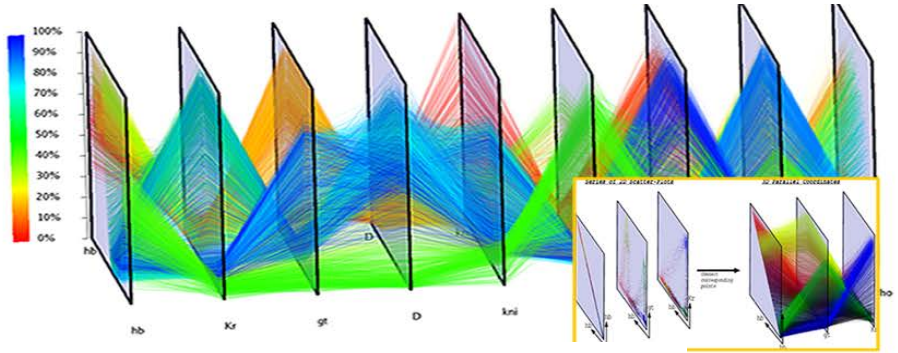
(Fanea et al., 2005)



(McDonnell & Mueller, 2008)



(Yuan et al., 2009)



(Rubel et al., 2006)

基于不同坐标系的可视化方法

- 平行坐标系
  - 看上去只要屏幕可以无限扩张，看似平行坐标轴便能够显示无穷多的数据维度
  - 但是事实并非如此，数据维度过多会导致视觉混乱（如图2）
  - 消除平行坐标系视觉混乱的基本方法
    - 基于 数据过滤 及 聚类的方法
    - 基于 坐标轴排序优化 的方法
    - 基于 视觉增强 的方法

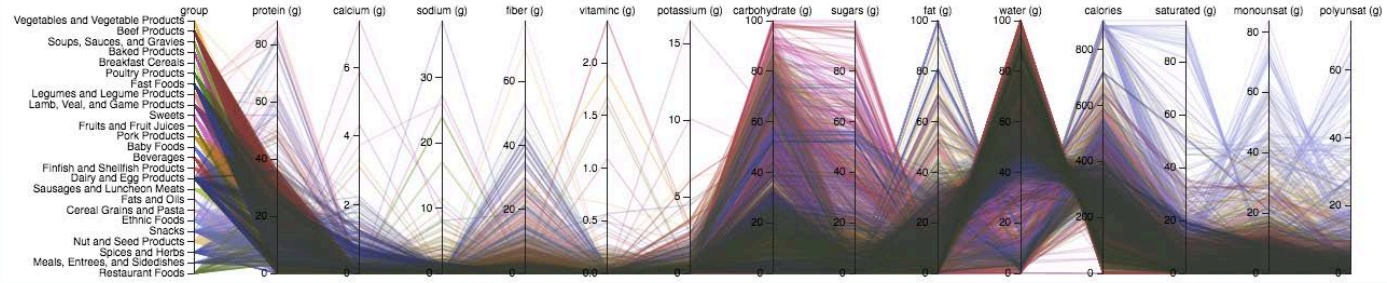


图1：平行坐标系可以显示较多的数据维度

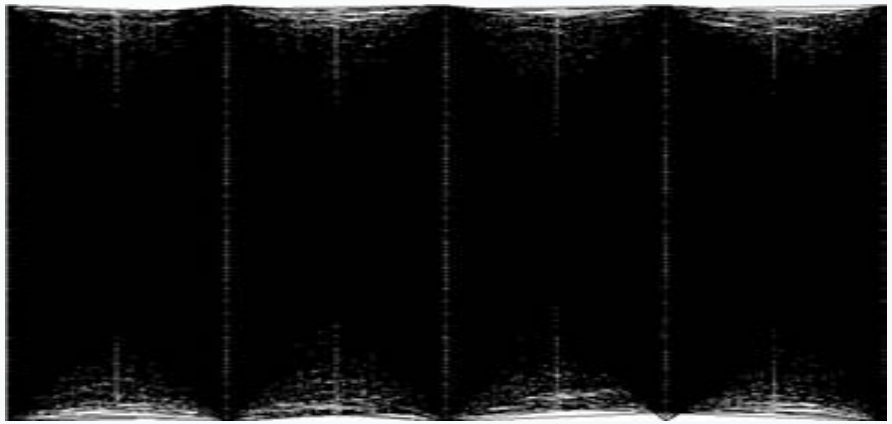
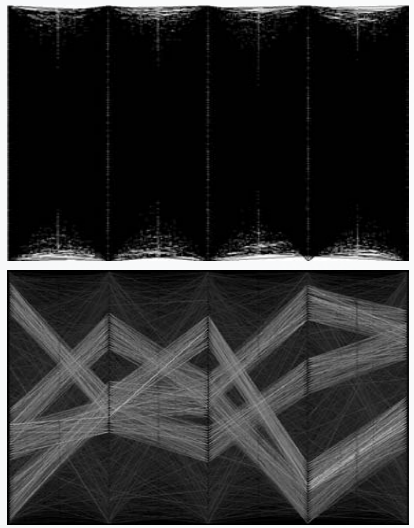


图2： 平行坐标系中的视觉混乱

基于不同坐标系的可视化方法

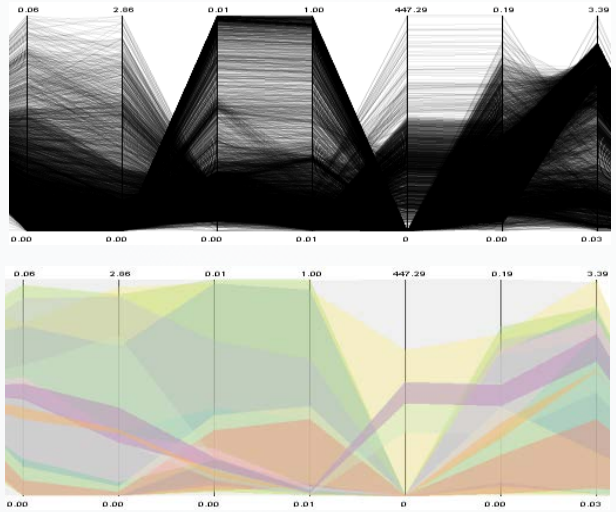
- 消除视觉混乱的基本方法 - “数据过滤”及 “聚类”

基于数据密度的过滤



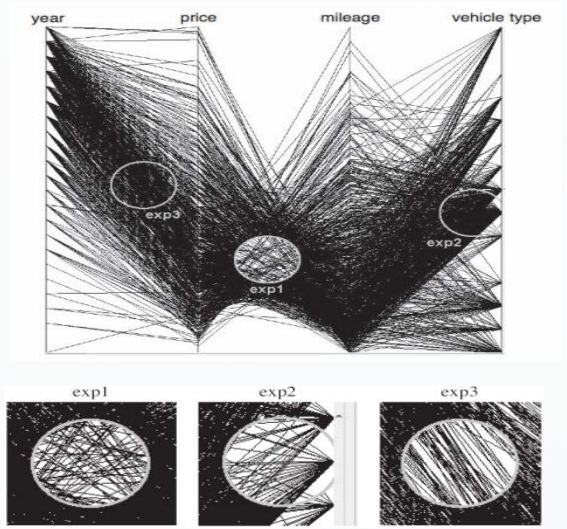
(Artero et al., 2004)

数据聚类



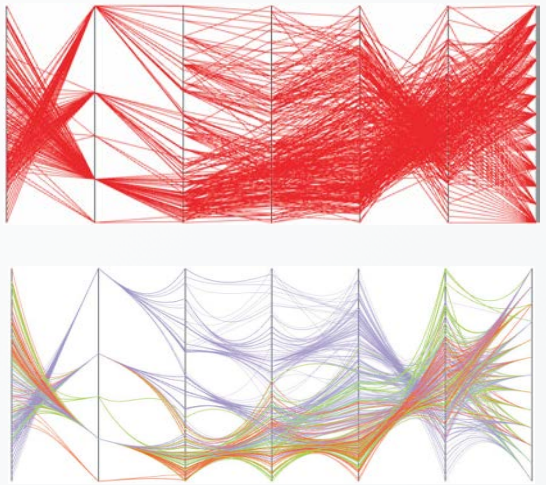
(Novotny et al., 2004)

数据采样



(Ellis & Dix, 2006)

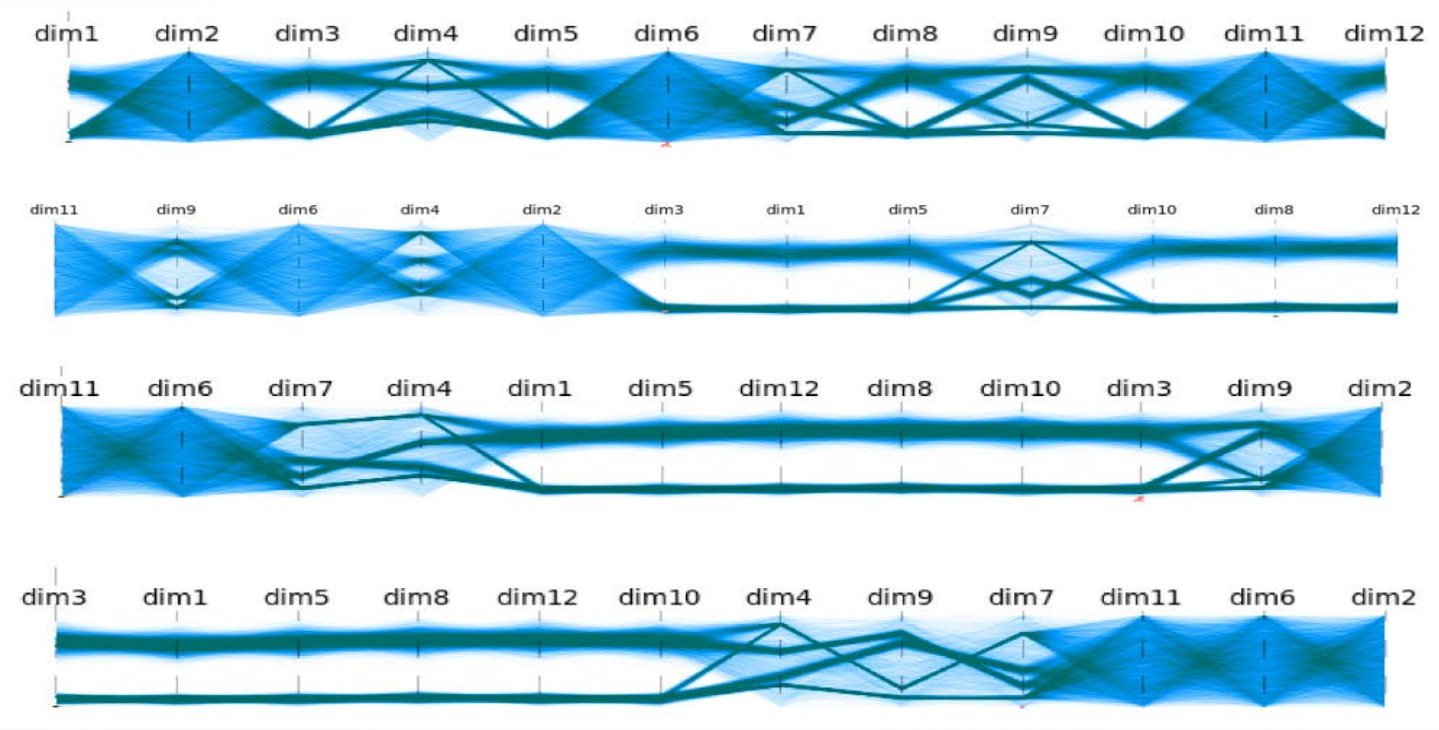
可视化元素聚类



(Zhou et al., 2008)

基于不同坐标系的可视化方法

- 消除视觉混乱的基本方法 - 坐标轴排序优化



数据集的原始排序

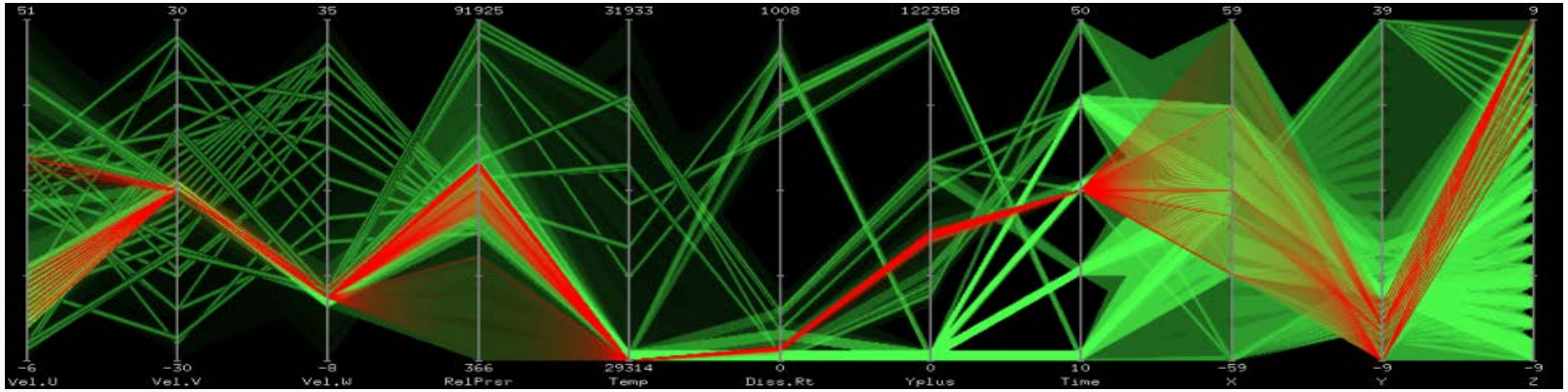
算法1 (Wang et al, 2004)

算法2 (Ankerst et al, 2004)

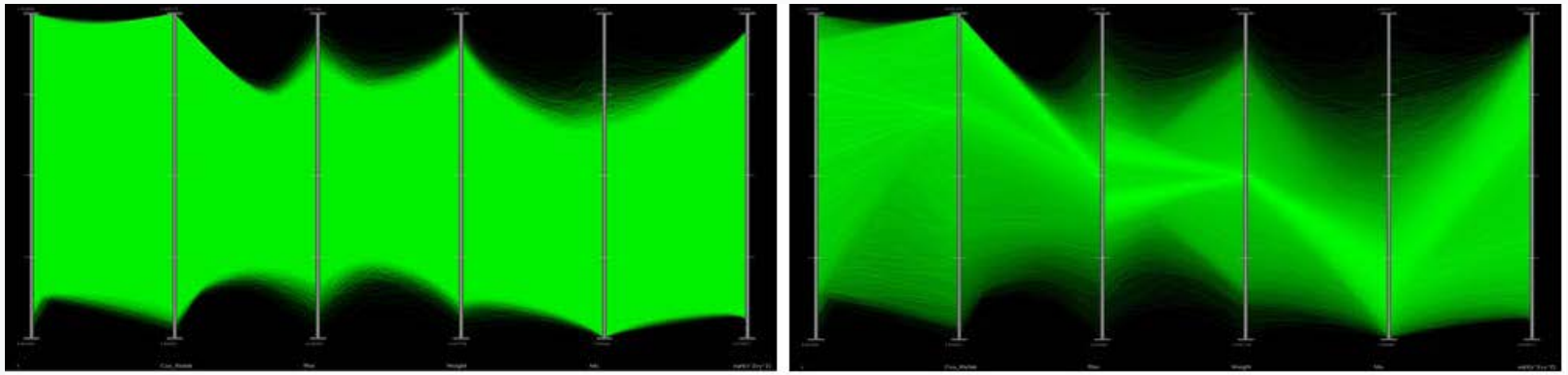
算法3 (Ferdosi & Roerdink, 2010)

基于不同坐标系的可视化方法

- 消除视觉混乱的基本方法 - 视觉增强



(Novotny & Hauser, 2006)



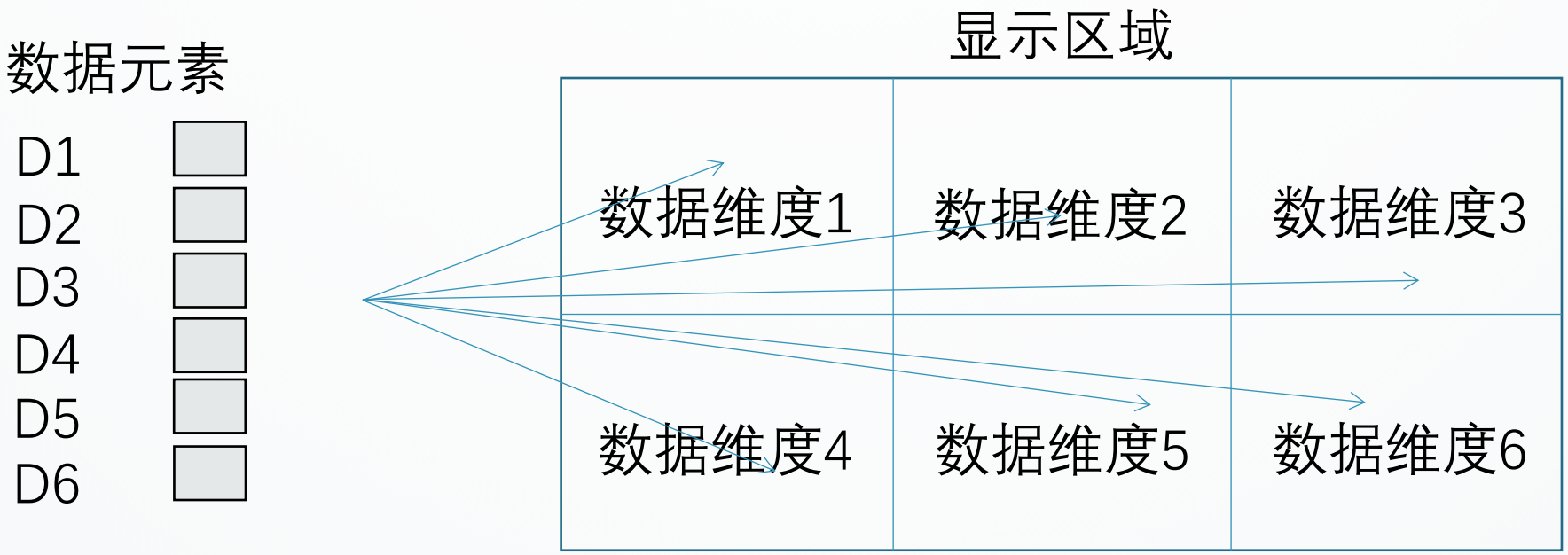
(Chad Jones et al. 2008)

根据数据密布调整颜色的透明度，从而达到增强视觉效果的目的

课程大纲

- 数据基础
- 多维度数据的可视化
  - 基于不同坐标系的可视化方法(Coordinate Systems)
  - 基于像素的可视化方法 (Pixel Oriented)
  - 基于图标的可视化方法（ Icon Based)
  - 基于网格的分视图展示方法（ Small Multiple ）
- 树的可视化
- 图的可视化

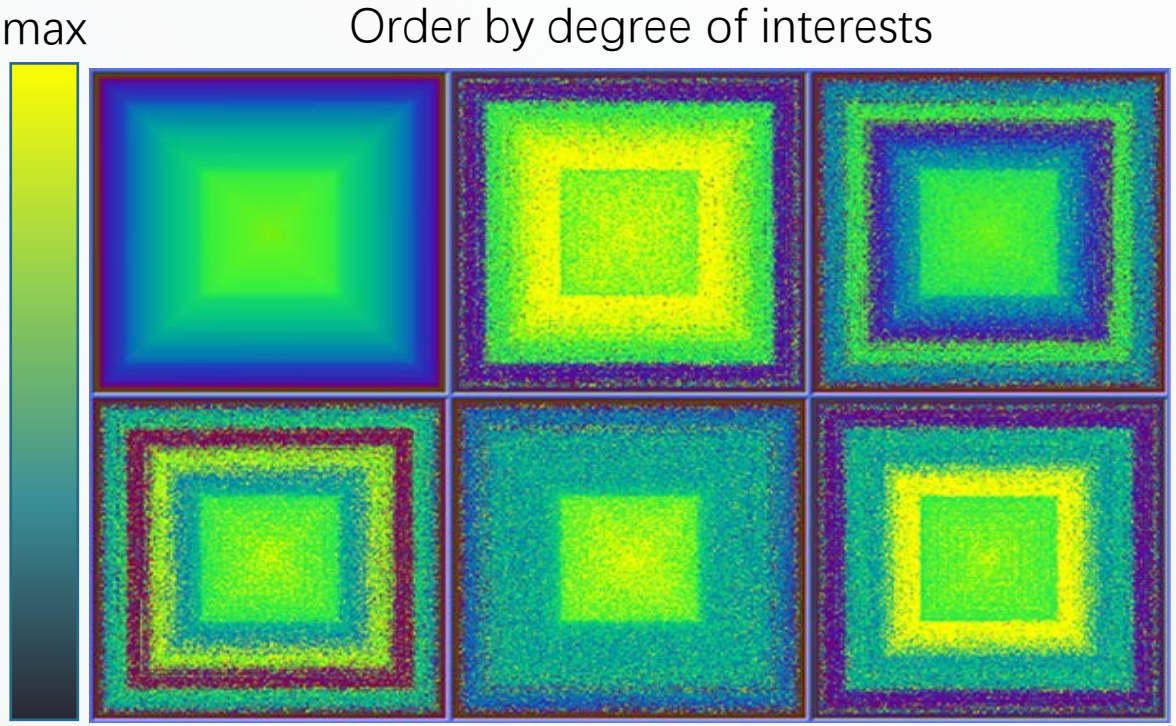
基于像素的可视化方法



- 一个包含有6个维度的多维度数据集合
- 将数据的显示区域划分为6个部分，每个部分分别对应着数据一个维度
- 数据元素中在每一个维度上的属性被表示为 对应区域中的一个像素，像素颜色映射了该属性的取值，不同的像素排列方法，对应了不同的可视化设计

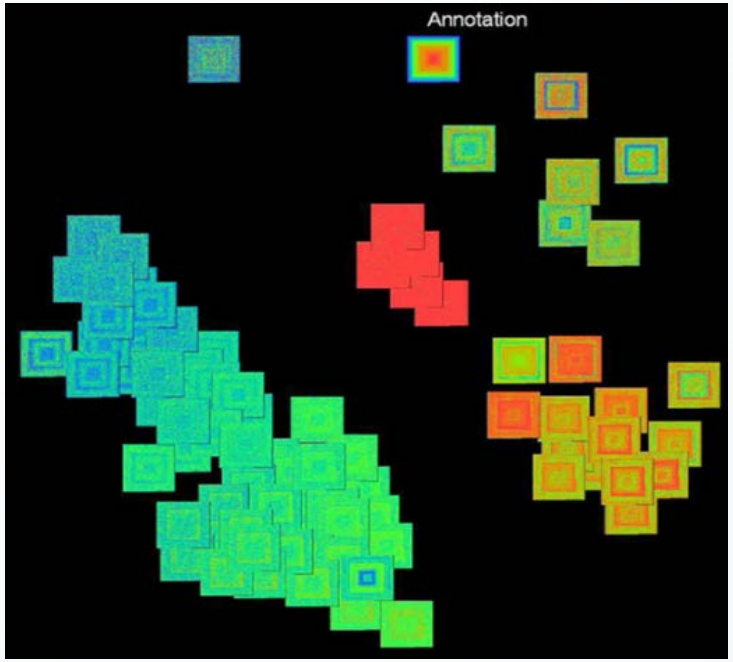
基于像素的可视化方法

- 不同的像素排列方法，对应了不同的可视化设计， 与数据关联



(Keim & Kriegel, 1994; 1996)

对数据库中的一张拥有6个维度表格进行可视化



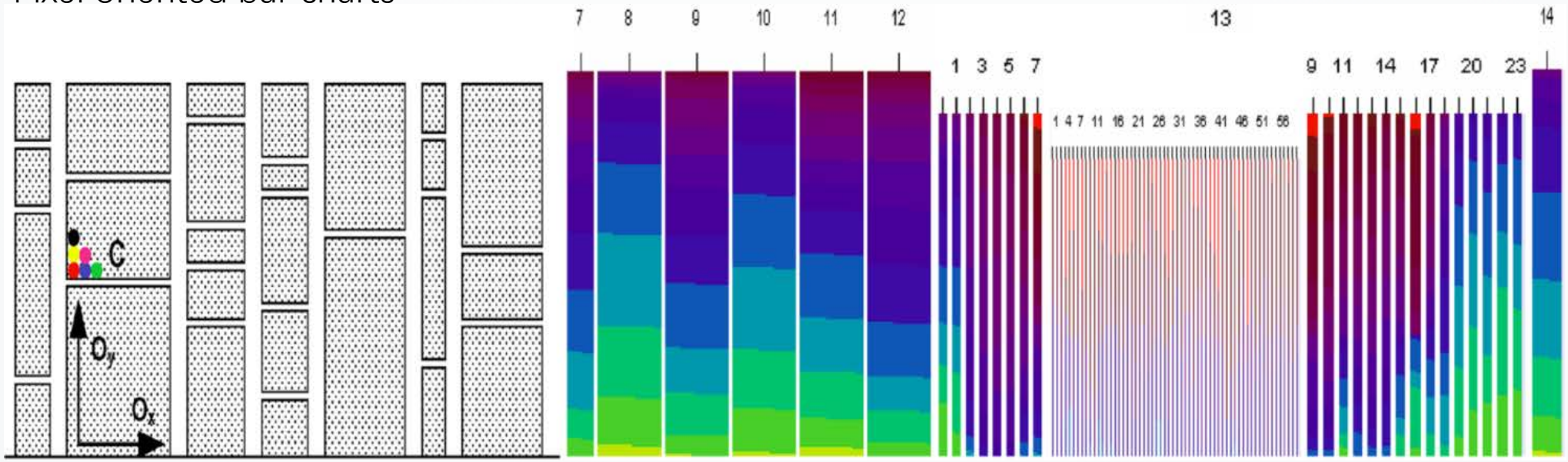
(Yang et al., 2006)

将多维度数据中的每一个维度可视化成一个由像素构成的方块，其中每一个像素对应了一个数据元素，通过MDS 映射，这些方块的位置反应出了数据维度之间的相关性

基于像素的可视化方法

- 不同的像素排列方法，对应了不同的可视化设计， 与数据关联

Pixel oriented bar charts



(Keim et al., 2001; 2002)

也可以将显示空间根据数据分类构成 柱状图的形式，从而形成一个基于像素的柱状图可视化

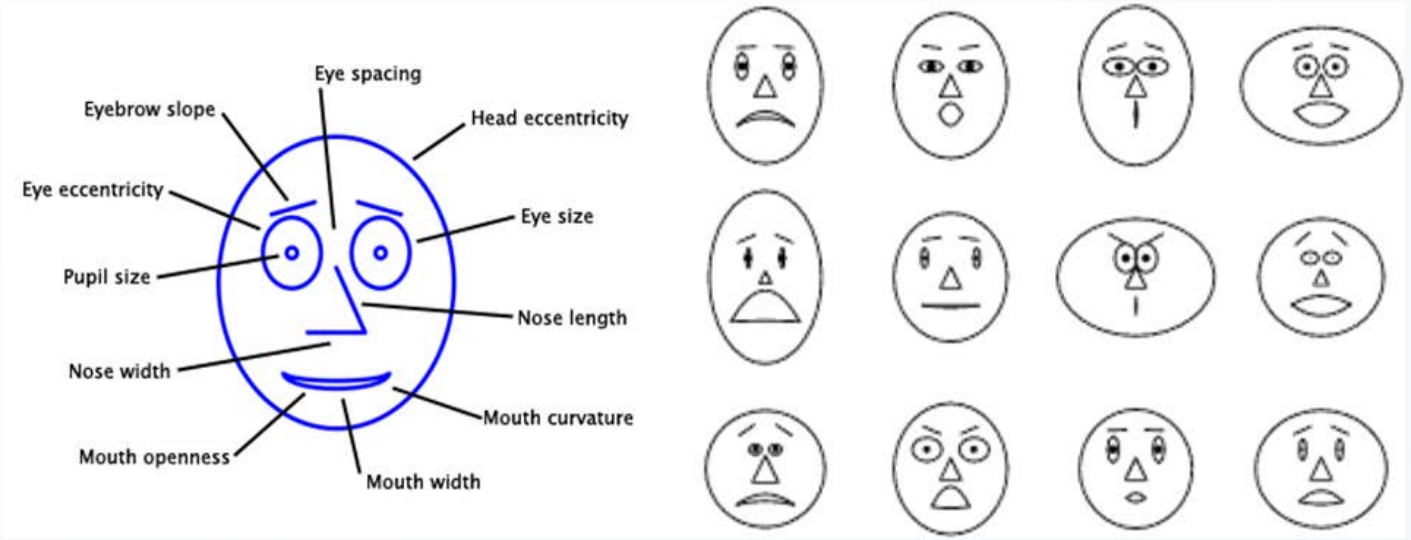
课程大纲

- 数据基础
- 多维度数据的可视化
  - 基于不同坐标系的可视化方法(Coordinate Systems)
  - 基于像素的可视化方法 (Pixel Oriented)
  - 基于图标的可视化方法（Icon Based)
  - 基于网格的分视图展示方法（Small Multiple）
- 树的可视化
- 图的可视化

基于图标的可视化方法

- **切尔诺夫面孔**（Chernoff Faces）

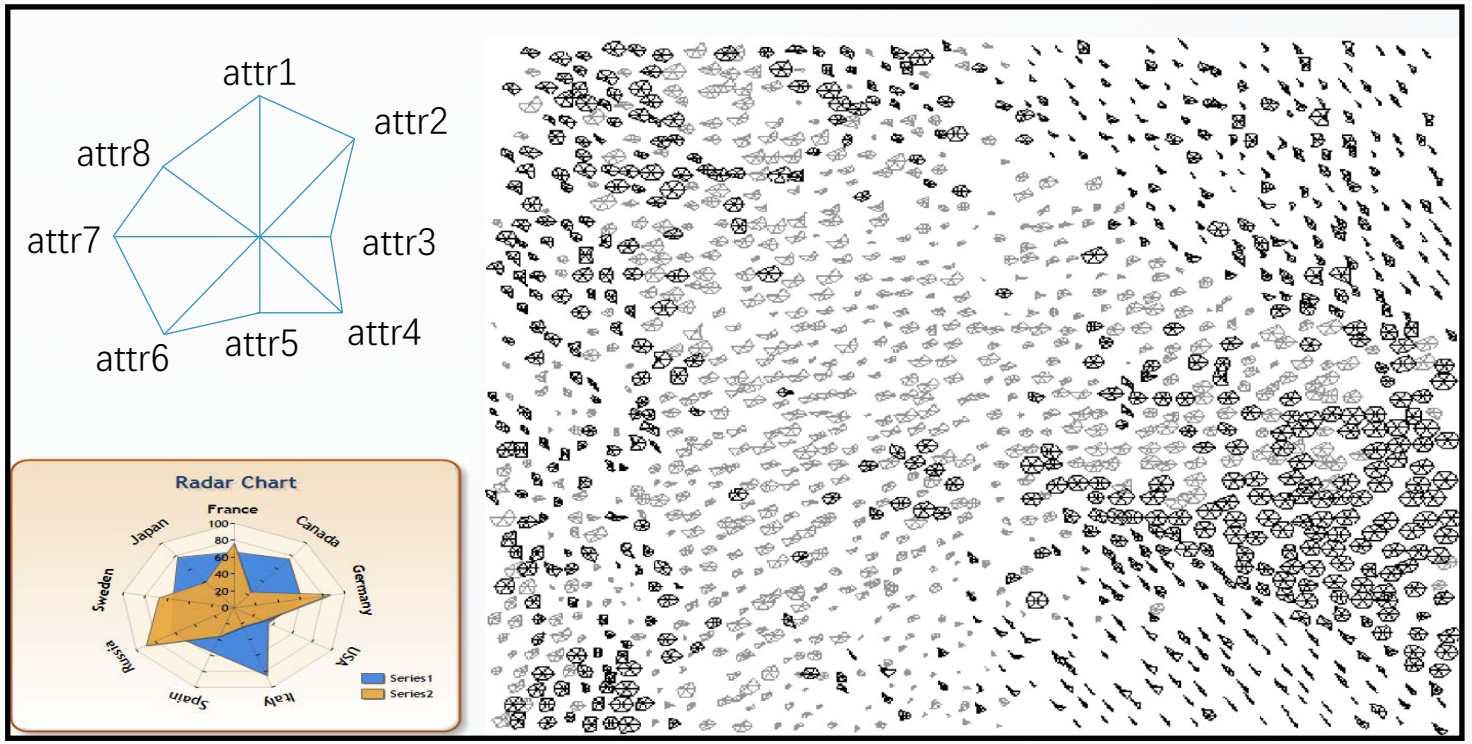
  - 一种利用人的五官特征来展示数据多维度属性的可视化方式
  - 每一个数据元素对应着一张面孔,面孔中眼睛的大小，嘴巴的宽窄、鼻子的高低等五官特征都对应着该数据元素在特定维度上的属性值的大小
  - 利用了人们能够快速辨识不同面孔特征的能力
  - 但是所形成的面部表情 往往与数据无关，导致语义上的误导



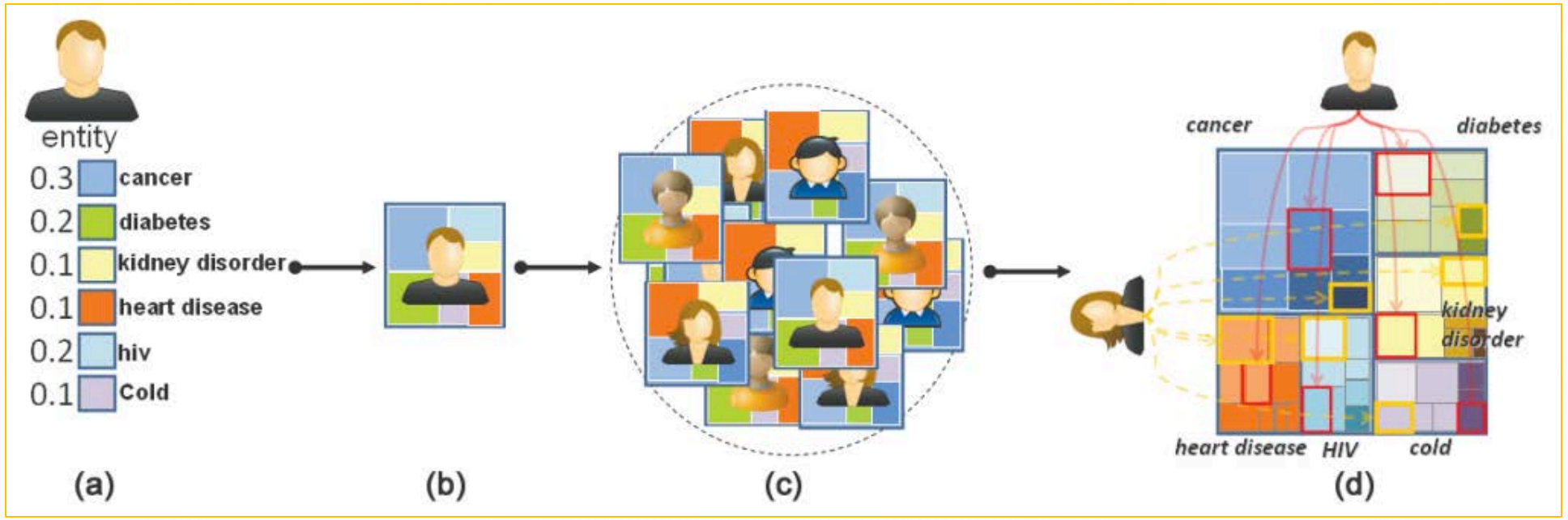
(Chernoff, 1973)

基于图标的可视化方法

- **星形图标**（Star Glyph）
  - 每一个数据元素对应着一个图标
  - 数据的维度被显示为围绕着中心原点的数据轴
  - 数据元素在不同维度上的取值构成了图标的外轮廓形状
  - 当独立重叠使用时，便构成了雷达图；当以图标形式集体使用时，便构成了能够凸显数据分布模式的可视化



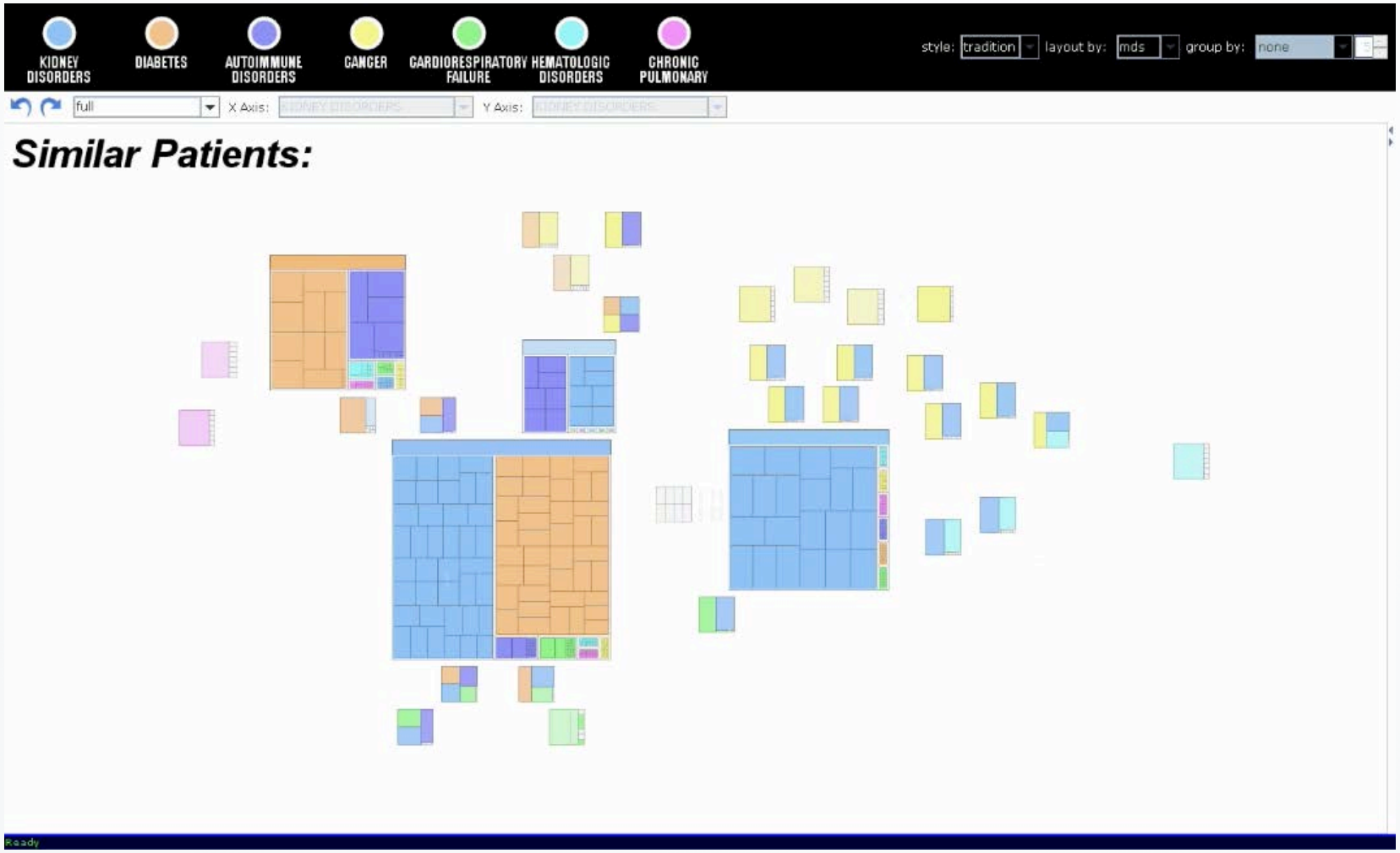
基于图标的可视化方法



- 动态图标技术（DICON）
  - 数据元素用方形图标表示
  - 图标中不同颜色的区域代表了不同的数据维度，区域的大小代表了所对应属性的取值
  - 展示多个数据元素时，可拆分单个数据元素的图标，并重组构成一个更大的代表数据集合图标
  - 重组的方式可以根据数据分组方式的不同进行动态调整

基于图标的可视化方法

- 动态图标技术（DICON）



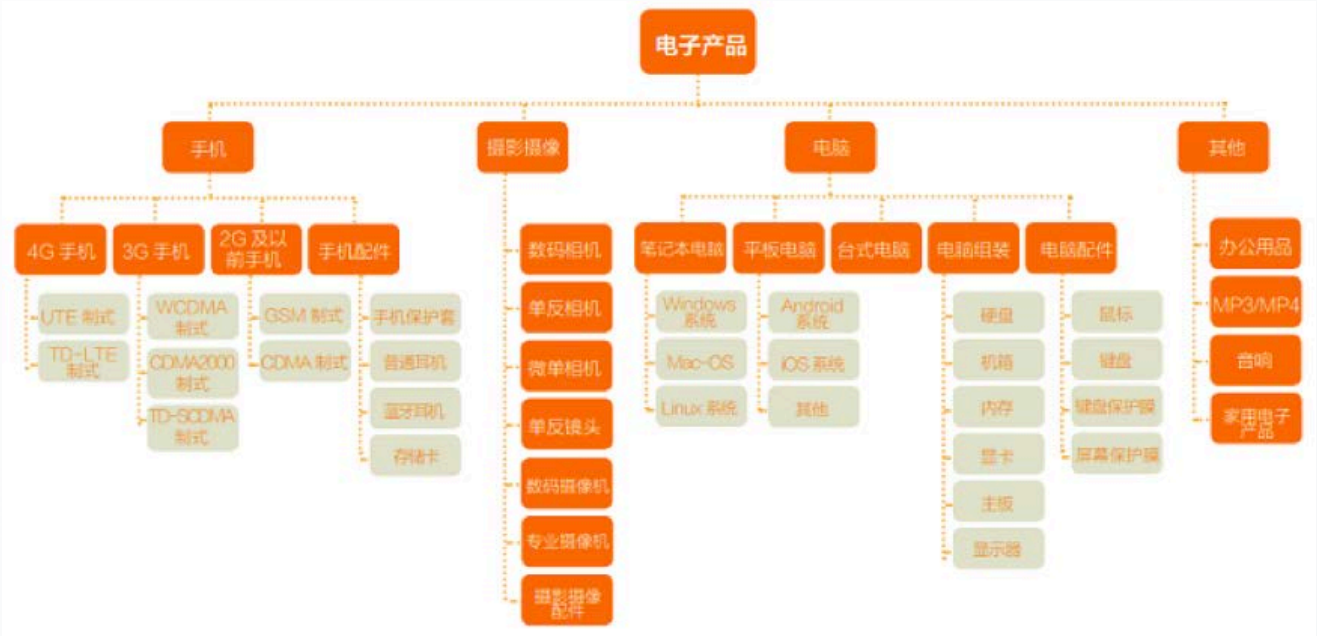
课程大纲

- 数据基础
- 多维度数据的可视化
- 树的可视化
  - 点线图（Node-Link Diagram）
  - 邻接图（Adjacency Diagram）
  - 包含图（Enclosure Diagram）
- 图的可视化

树的可视化

- ”树“是一种基本的数据结构，很多数据中存在层次结构，例如
  - 公司的组织结构图，电脑中的文件目录管理结构，商品的层次化分类等

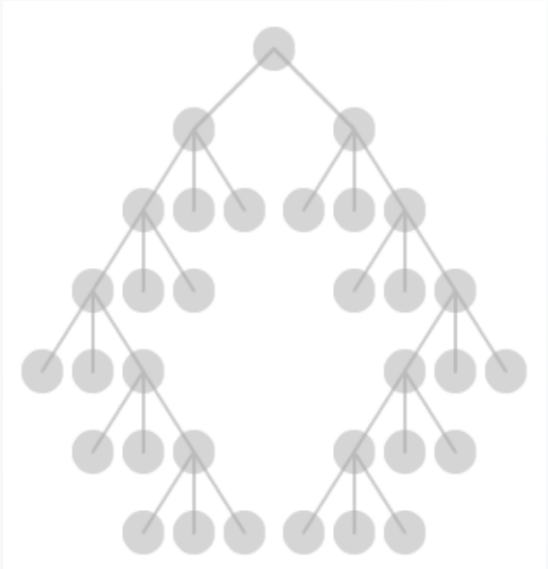
- ”树“的可视化要求必须能够清晰展示层次结构，有三种基本的可视化方式
  - 点线图（Node-Link Diagram）
  - 邻接图（Adjacency Diagram）
  - 包含图（Enclosure Diagram）



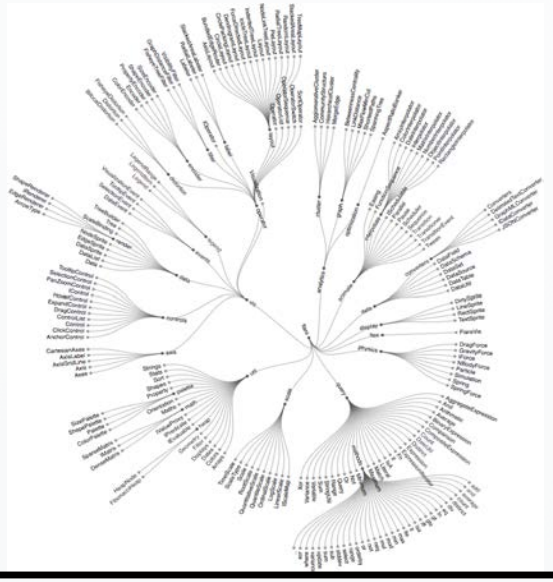
树的可视化

- 点线图（ Node-Link Diagram ）
  - 一个 整洁的 树的布局（Tidy Tree）需要做到：（1）构图紧凑；（2）相同层次的节点被放置在相同的可视化层级之上；（3）父亲节点在子节点的中心

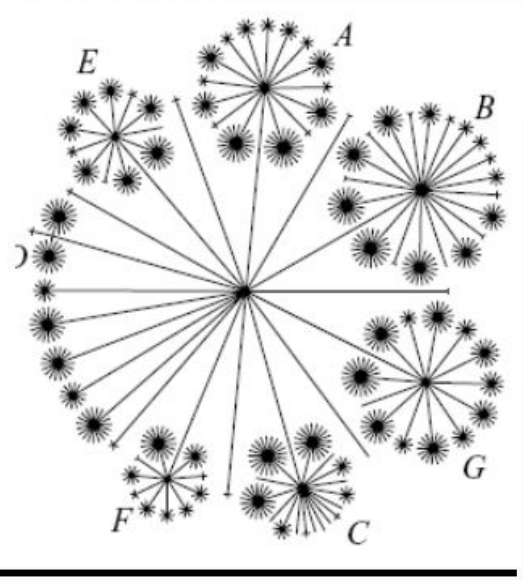
Hierarchy Layout



Radial Layout

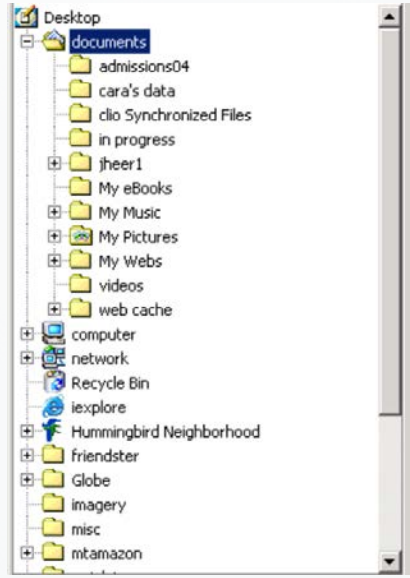


Circle Layout



Tidy Tree

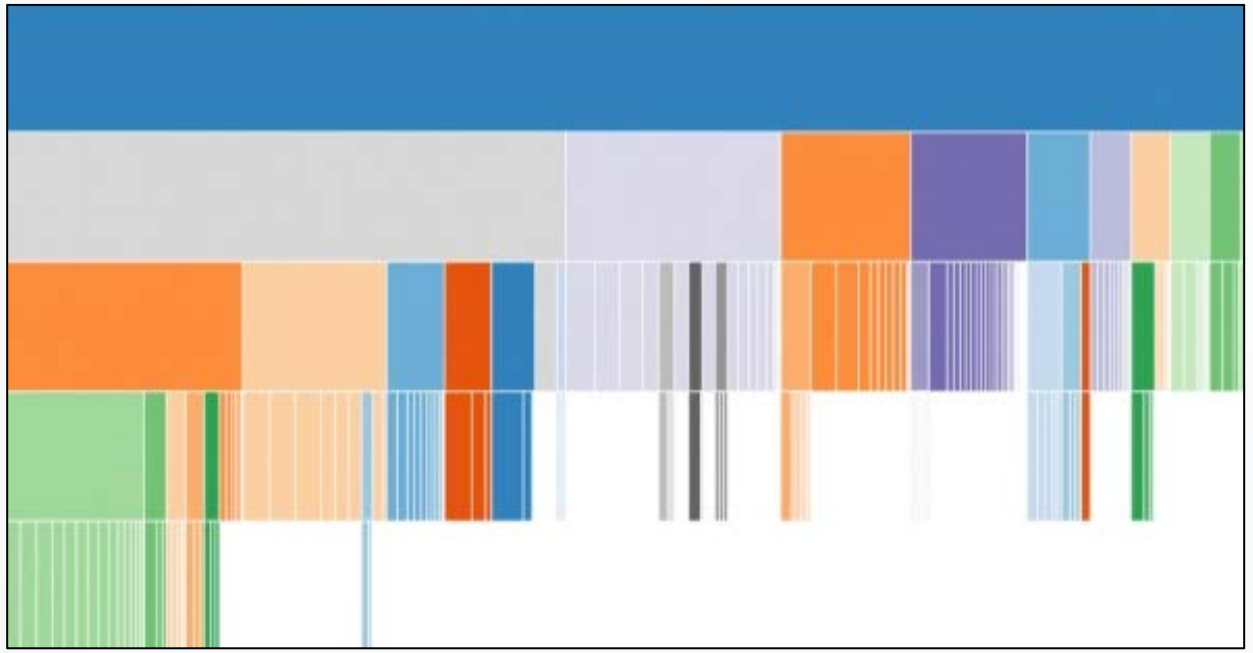
Intended Layout



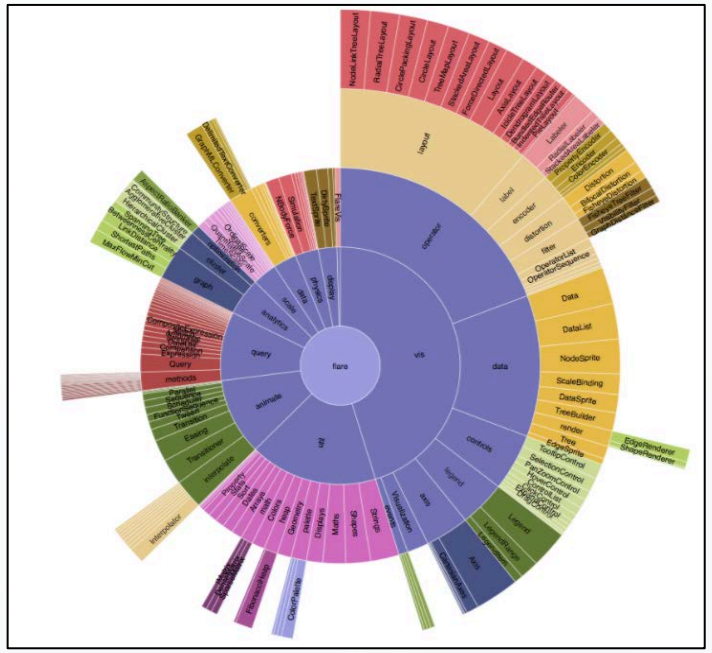
非 Tidy Tree

树的可视化

- 邻接图（ Adjacency Diagram ）
  - 通过可视化节点之间的相邻位置关系以展现树的拓扑结构
  - 节点的大小，宽度，可以用来展现另外一个数据维度



Icicle tree

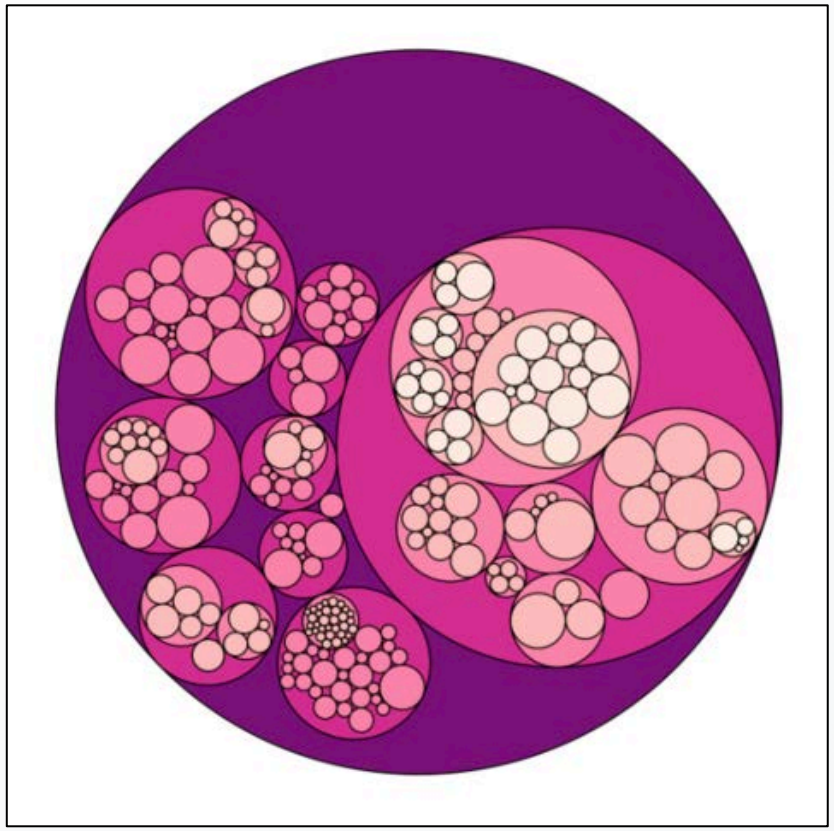


Sunburst Tree

树的可视化

- 包含图（ Enclosure Diagram ）
  - 通过包含关系来展现树中的层次结构

Treemap (树图)



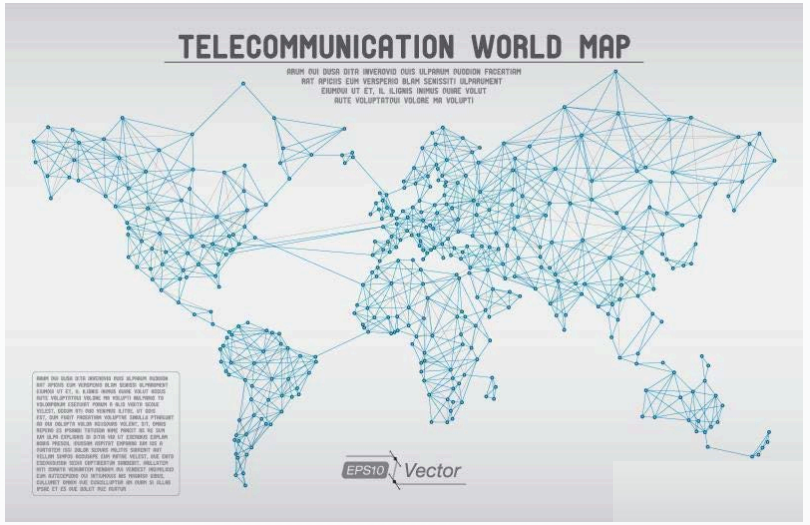
Circle Packing

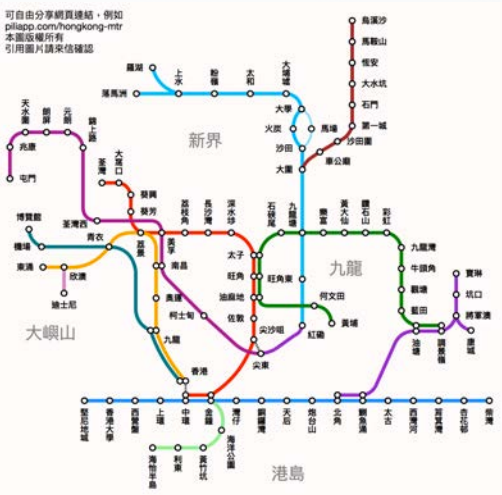
课程大纲

- 数据基础
- 多维度数据的可视化
- 树的可视化
- 图的可视化
  - 简介及分类
  - 力导向图及布局
  - 邻接矩阵
  - 混合可视化方法

图的可视化

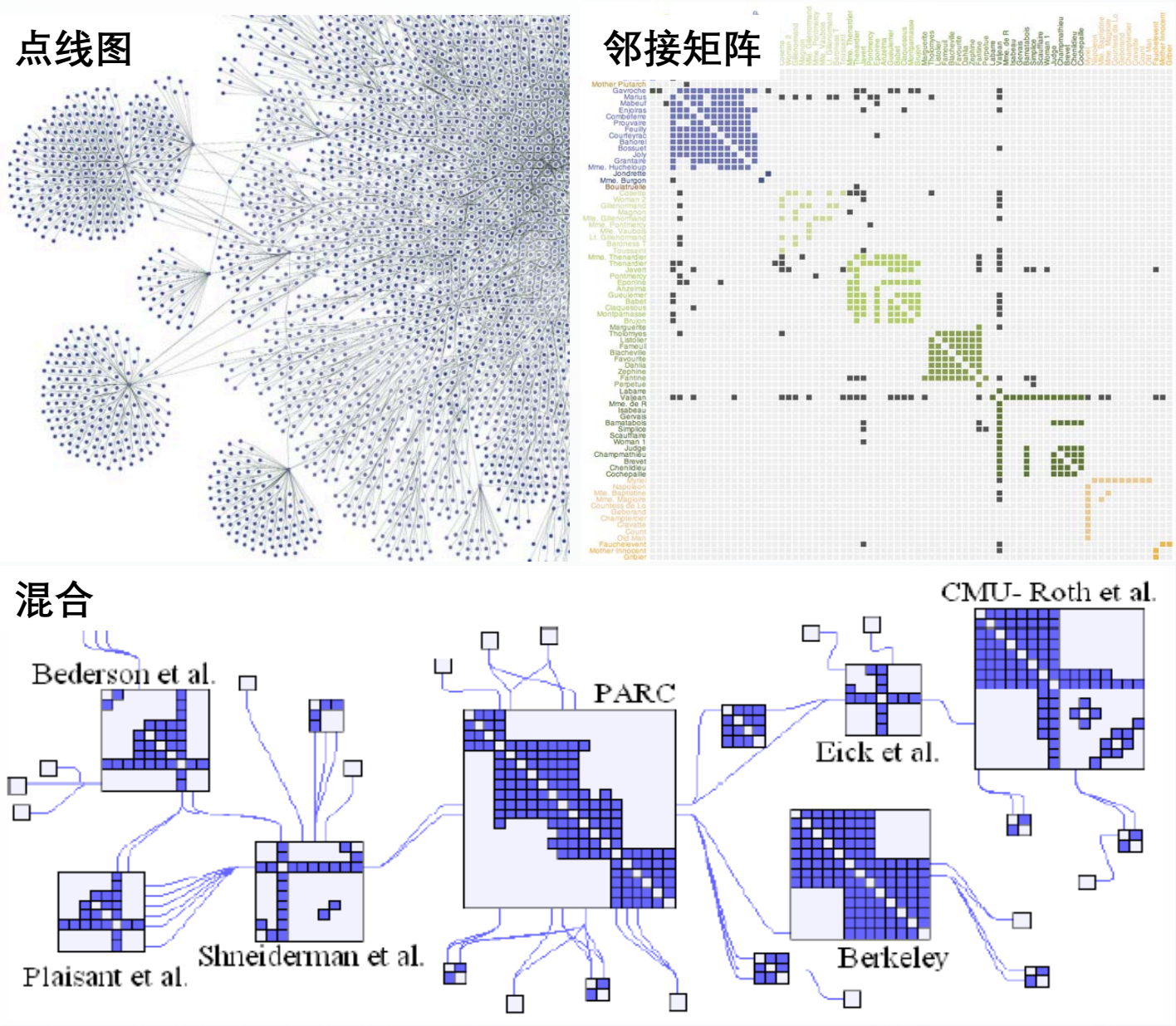
- 图是最普遍的数据结构，几乎无处不在，通信网络，互联网，地铁网
- 对于图结构的可视化可以追溯到100多年前，而现代有关图可视化的研究开始于上世纪70年代
- 今天围绕着图结构，在不同的应用领域，诞生了各种各样丰富多彩的可视化设计





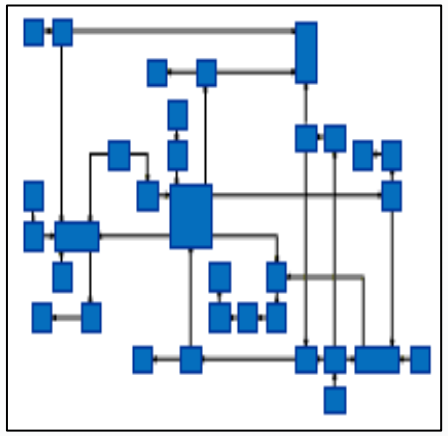
图的可视化

- 图可视化的基本形式
  - 点线图（Node-Link Diagram）
  - 邻接矩阵（Adjacency Matrix）
  - 混合可视化方法（NodeTrix）

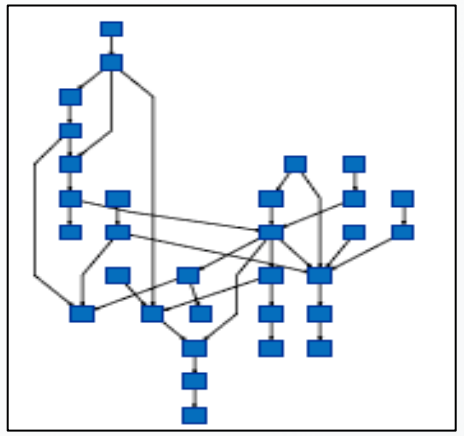


图的可视化

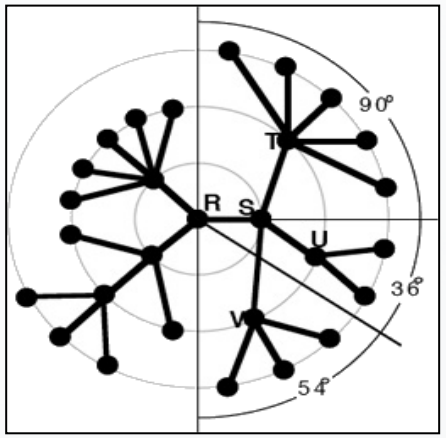
- 点线图（Node-Link Diagram）
  - 最直观的“图”表达方式
  - 布局问题是点线图可视化的关键
  - 布局问题：计算点在屏幕上的位置，以及边的绘制方式，从而使得尽可能的减少点的覆盖与线的交叉
  - 点线图的布局可以大致分为四类：
    直角折线布局（ Orthogonal ）、放射布局（Radial）、
    层次布局（Hierarchy）、力导向布局（Force-Directed）
  - 接下来我们将重点讲解 **力导向布局** 的计算方法



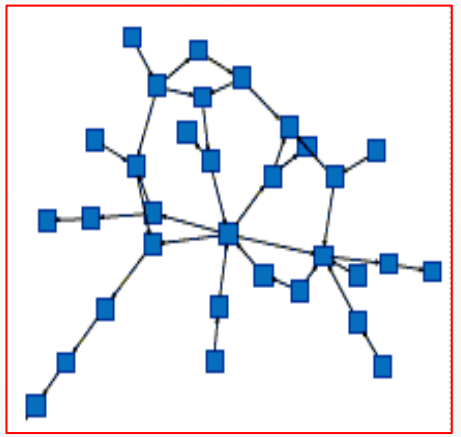
直角折线布局



层次布局图



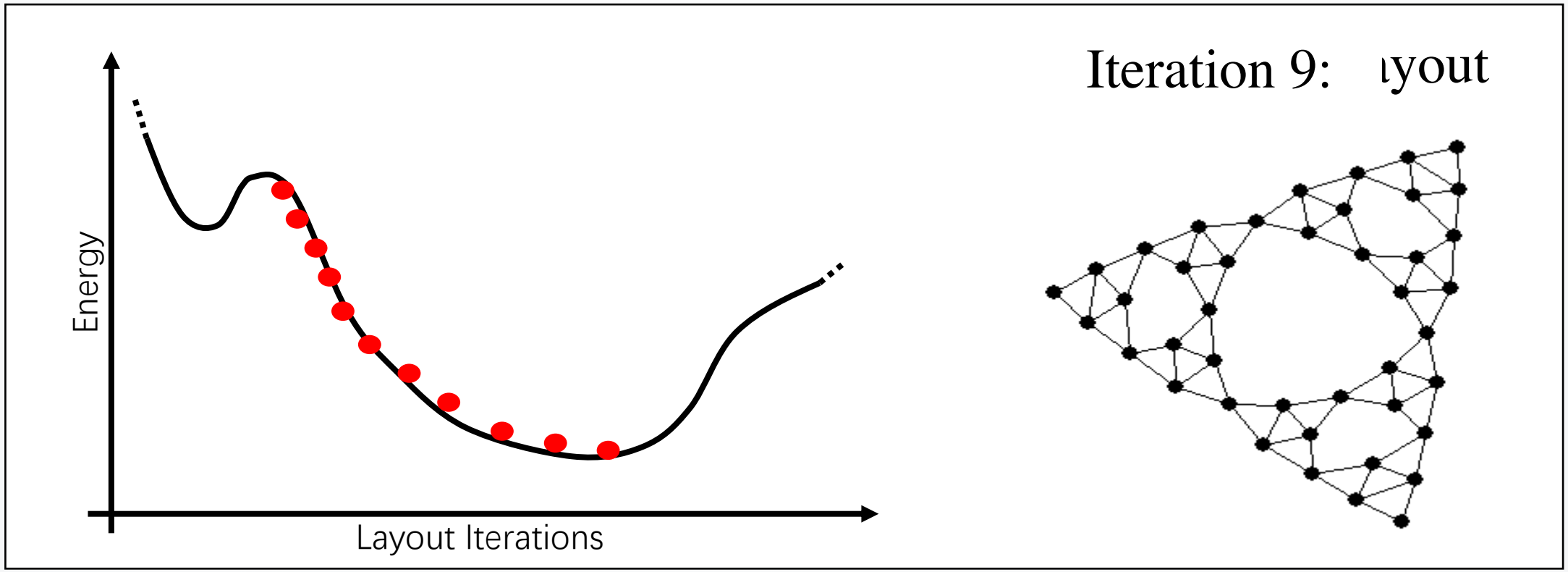
放射布局



力导向布局

图的可视化

- 点线图的布局 - 力导向布局
  - 力导向布局是一个迭代求解能量方程的过程
  - 布局的目的是让整个图中因为点与点之间相互位置而构成的“势能”降到最低
  - 力导向布局的能量方程有多种实现方式



图的可视化

- 点线图的布局 - 力导向布局

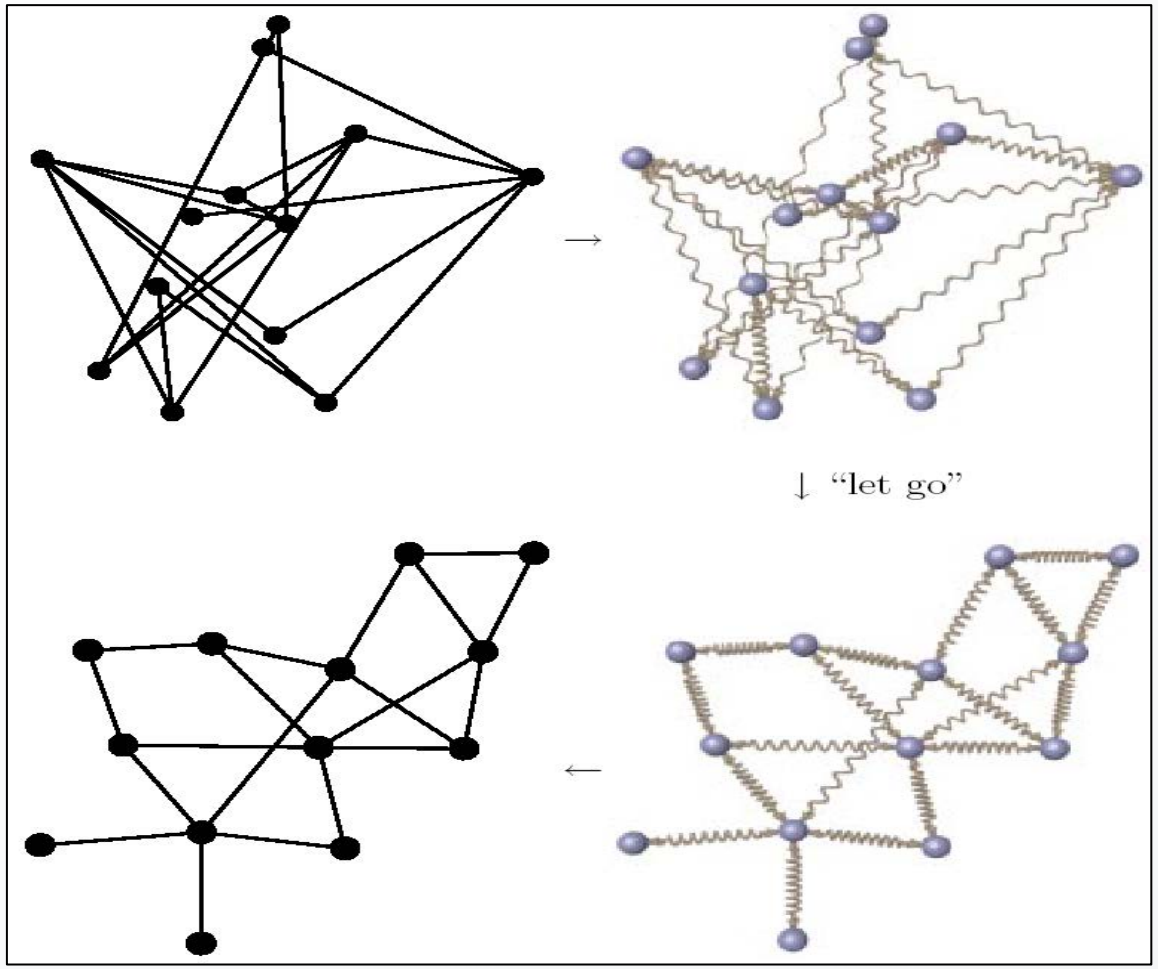
  - 方法I：物理模型
    点看做电荷，而边看做弹簧，无需人为干预，达到力平衡

  - 相连的点之间拥有相互引力，任意两点之间存在相互斥力

    引力 $f_a(d) = d^2/k$　　斥力 $f_r(d) = -k^2/d$

  - 当每个点上的力达到平衡时，势能最小，图的布局得以完成

$$F(v) = \sum_{\{u,v\} \in E} f_{uv} + \sum_{u \in V} g_{uv}$$



图的可视化

- 点线图的布局 - 力导向布局

  - 方法3：Node-Repose LinLog

$$Cost_{NodeLinLog} = \sum_{\{u,v\} \in E} \|y_u - y_v\| - \sum_{u \in V, v \in V} \ln \|y_u - y_v\|$$

当两点相连时，加在边上的引力

图中任意两点之间的斥力

图的可视化

- 点线图的布局 - 力导向布局

  - 方法4：Edge-Repose LinLog

$$Cost_{EdgeLinLog} = \sum_{\{u,v\} \in E} \|y_u - y_v\| - \sum_{u \in V, v \in V} deg(u) deg(v) \ln \|y_u - y_v\|$$

当点的度数比较大的时候，即点连接了较多边的时候，斥力也比较大

图的可视化

- 点线图的布局 - 力导向布局

  - 方法2：KK-Layout

两点之间的屏幕距离　　图中的理论距离

$$Cost=\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\frac{1}{2}k_{ij}(\|\boldsymbol{y}_i-\boldsymbol{y}_j\|-l_{ij})^2$$

$$l_{ij}=L\cdot d_{ij}$$

- l_{ij}: desired on-screen length of edge (i,j)
- d_{ij}: length of shortest path (n. of steps) from i to j
- L: desired on-screen length of an edge

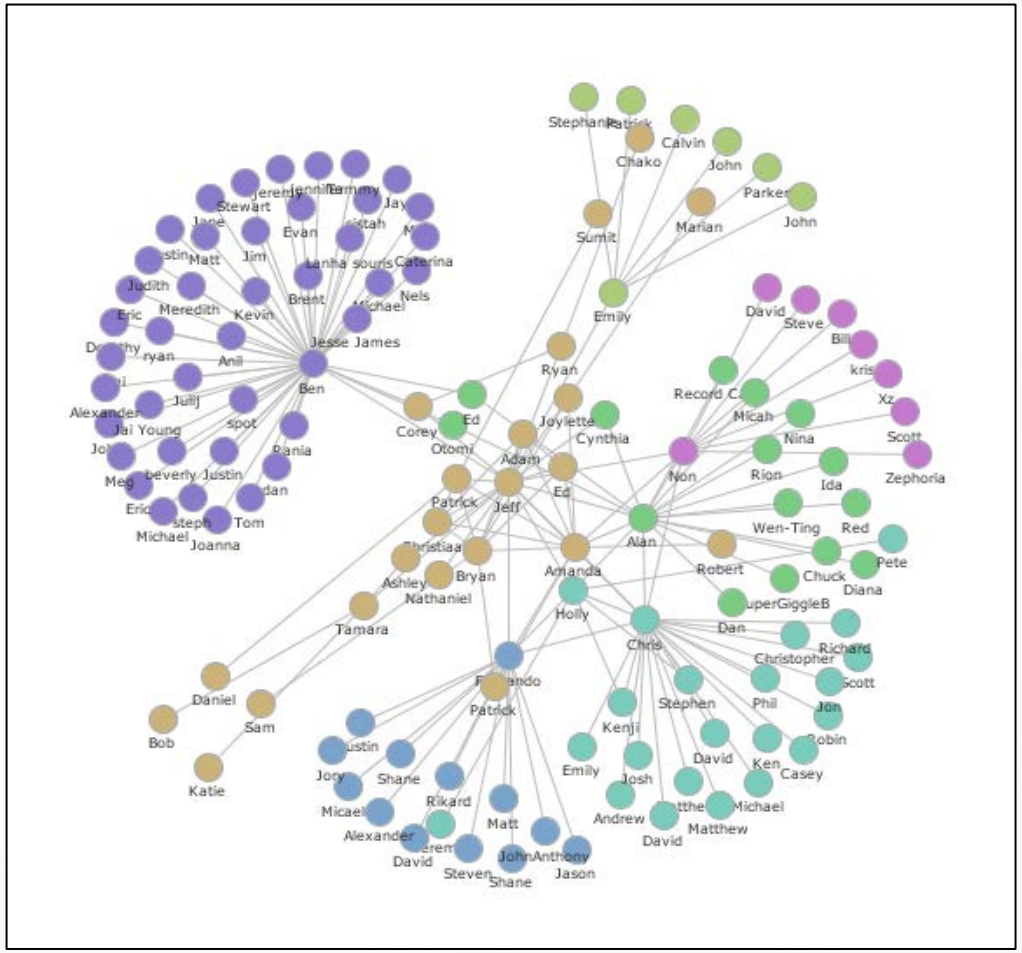
$$k_{ij}=K/d_{ij}^2$$

- K: constant
- k_{ij}: strength of spring (i,j)

当两点间距离越大的时候，改系数较小，表示精确保留两点之间的相对距离，并不是那么重要；相反，当两点间距离较小时，该系数较大，表示精确保留两点之间的相对距离比较关键

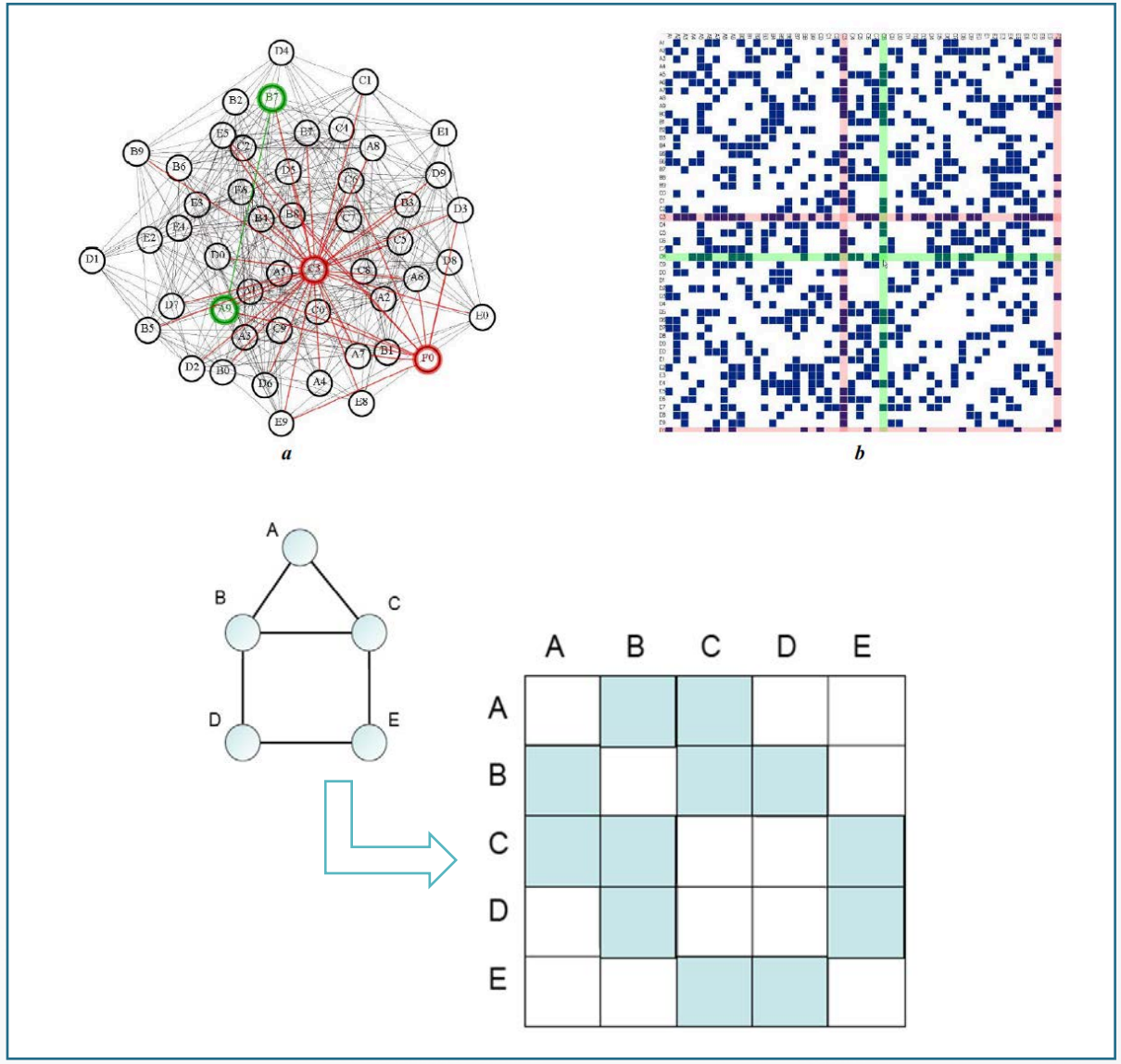
图的可视化

- 点线图的布局 - 力导向布局
  - 能够清晰的显示 图中由点组成的“簇”，符合了格式塔法则中的 相似性原则（Proximity Rule）
  - 布局结果一般具有对称性，满足了美观度的要求
  - 能够最小化边的平均长度，降低了边与其他边相交的可能
  - 没有人为的设计干预，自动根据图的拓扑结构完成布局，抗扰动性较好



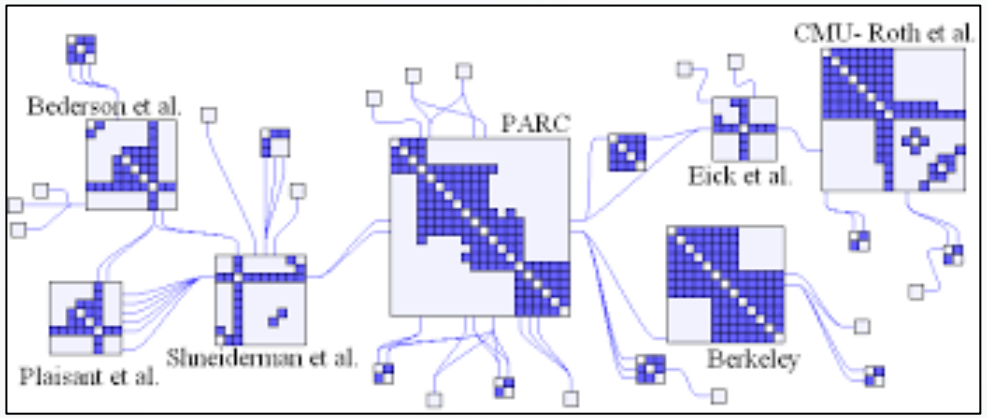
图的可视化

- 点线图 – 邻接矩阵
  - 点线图虽然直观，但是当图比较密集时（Dense），因为线的交叉及点的遮挡，难以清晰的展现图中的信息
  - 使用图的邻接矩阵展现图结构，能够完全避免上述问题
  - 邻接矩阵中的每一行，每一列代表图中的一个点，矩阵中的元素代表图中的边，表示了对应行、列所代表的点之间相互连接
  - 用户无法在邻接矩阵中追踪图中的一条路径



图的可视化

- 点线图 – 混合方法
  - 结合点线图与连接矩阵的优点
  - 当图中结构较为密集时，用邻接矩阵展现，以避免点的遮挡与线的交叉
  - 当图中结构较为疏松时，用点线图展现，以方便路径追踪



NodeTrix: a Hybrid Visualization of Social Networks,
Nathalie Henry, Jean-Daniel Fekete, and Michael J. McGuffin
IEEE InfoVis 2007

课程总结

- 本节课
  - 数据基础
  - 多维度数据的可视化
  - 树的可视化
  - 图的可视化

- 下节课
  - 可视化的评估