可视化基本原则

可视化是一种将抽象的数据转化为可见的几何特征，帮助使用者观察数据的计算结果和数据分布特征。可视化提供了一种呈现数据中不可见特征的方法。

可视化研究的是人或者机器如何感知、处理、交互可视化的信息

- 基于真实数据进性可视化
- 可视化需要生成一张图像
- 生成的可视化图像是方便使用者观察的

重点是要将数据信息呈现给人(使用者)，将人放在可视化分析的回路中

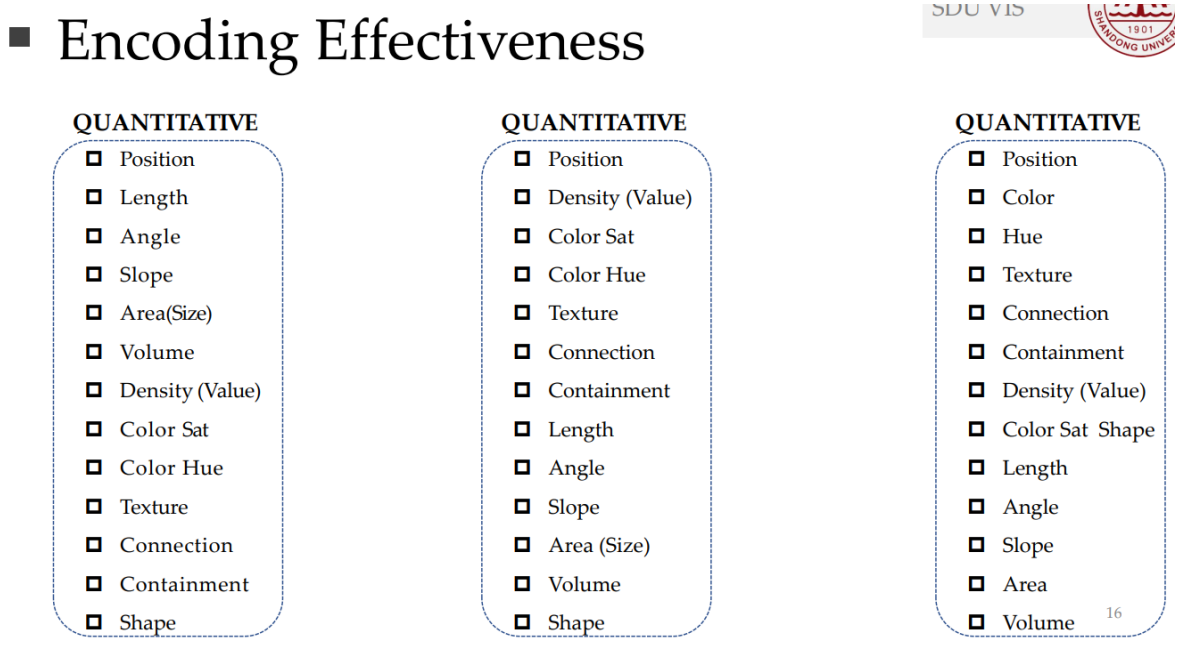
编码有效性排序

常见的可视化维度：

位置，长度，区域面积体积，亮度，色调，角度，形状

不同数据种类下，比较好的数据维度

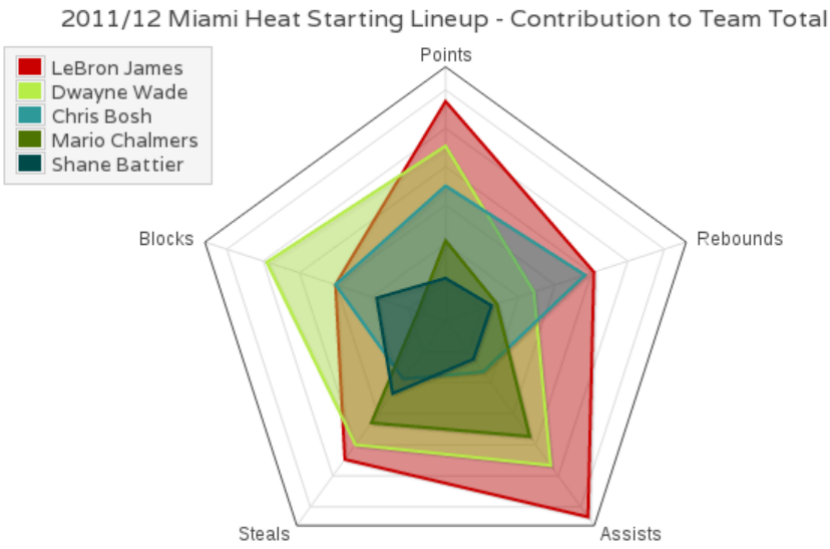
- 定量数据：位置、长度、角度
- 定序数据：位置、密度、色彩饱和度
- 名词性数据：位置、色调、质地、纹理、连接关系



多维数据可视化

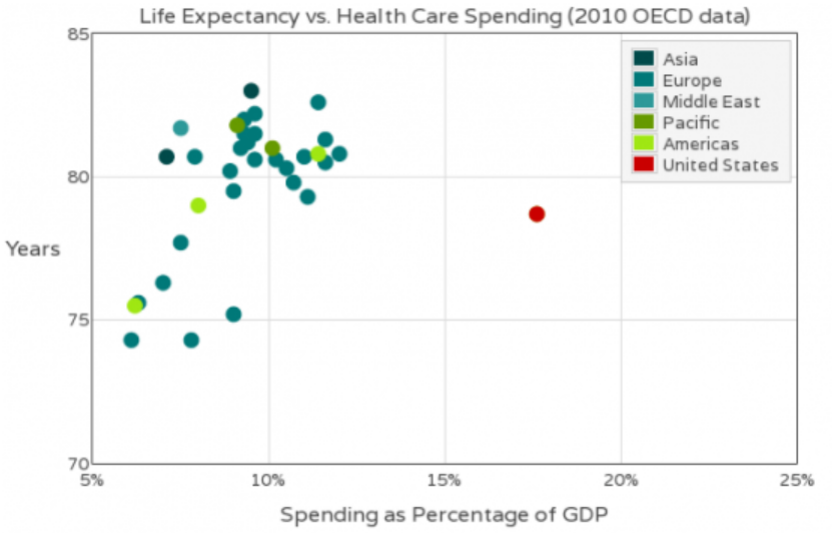
雷达图

雷达图适用于多维数据（四维以上），且每个维度必须可以排序（国籍就不可以排序）。但是，它有一个局限，就是数据点不能太多，否则无法辨别，因此适用场合有限。



散点图

散点图适用于三维数据集，但其中只有两维需要比较。



上图是各国的医疗支出与预期寿命，三个维度分别为国家、医疗支出、预期寿命，只有后两个维度需要比较。

为了识别第三维，可以为每个点加上文字标示，或者不同颜色。

平行坐标系

横轴是数据的多个属性维度，纵轴向是数据在该属性维度上的值。在平行坐标系中，一条有多个属性维度的数据，表示成一条折线，从第一个属性坐标开始到最后一个属性坐标结束。

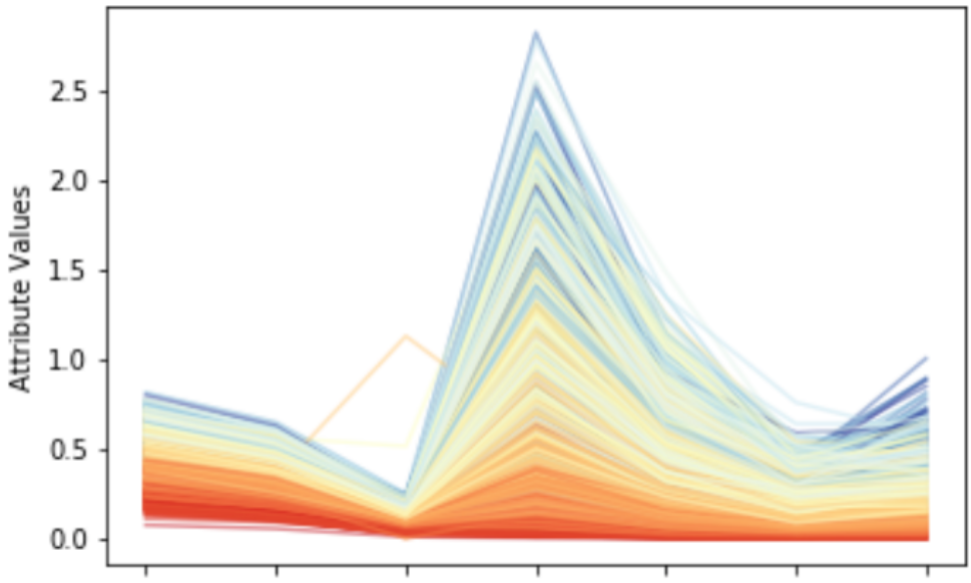
平行坐标系能够比较快捷的反映出一个数据多个属性之间的对应关系，在多维问题中有比较好的效果。

对于散点图中的一个点，可以映射到平行坐标系上的一条横跨两个维度的线。

1. 折线走势“陡峭”与“低谷”只是表示在该属性上属性值的变化范围的大小，对于标签分类不具有决定意义，但是“陡峭”的属性上属性值间距较大，视觉上更容易区分出不同的标签类别
2. 标签的分类主要看相同颜色的折线是否集中，若在某个属性上相同颜色折线较为集中，不同颜色有一定的间距，则说明该属性对于预测标签类别有较大的帮助
3. 若某个属性上线条混乱，颜色混杂，则较大可能该属性对于标签类别判定没有价值

如果数据的条数过多就会出现平行坐标系的折线过于密集导致难以分辨的问题。

数据样本数量过多的改进方法：使用聚类算法和增强视觉效果的可视化技术，例如对数据进行聚类再分别使用不同的颜色进行可视化。



平行坐标图主要帮助我们观察目标与哪些属性相关，尤其适用在属性超过三个以上的问题中

图布局

- **Force-direct layout**

  基本算法：

  1. 初始化：每个节点作为系统中的一个粒子，被初始化到一些随机区域。
  2. 力的作用过程：节点受到里的作用，被逐渐移动到某些位置上，

     两种力：

     1. 斥力：$F = Kr/d^2$
     2. 引力：仿照弹簧拉力，$F = Ks(d - L)$
  3. 迭代过程：在一个循环中不断迭代，每次计算两种力作用导致的位移并且更新节点的位置，直到收敛到一个比较好的位置。
  4. 迭代结果：两种力不断作用到节点上，节点在不断位移之后逐渐趋于平衡，达到一个稳定的状态，这种稳定状态后形成的就是力导图。

  需要调节的参数

  斥力和引力的参数：Kr、Ks、弹簧原长L、迭代过程的步长Δt

  力导图存在的问题

  1. 迭代步长不易确定，太小收敛速度慢，太大导致系统振荡，不易达到稳定状态。
  2. 原始算法复杂度高$O(n^3)$
  3. 节点和边的数量多会导致边的交叉问题，难以判断节点连接的是哪一条边。

  优化方向

  1. 优化距离的计算，改为距离平方的比较，减少平方根的计算
  2. 加入温度概念，表示图像绘制进展，允许节点在前期移动比较远的距离，之后逐渐限制节点的移动距离。
  3. 解决两个邻居节点完全相同，卡在一起的情况，系统随机生成一个斥力推开他们。
  4. 允许用户调整参数
  5. 加入弹簧等效代替斥力计算

- **Barnes-Hut 近似：**

  为了加速计算并使大规模模拟成为可能，天文学家 Josh Barnes 和 Piet Hut 设计了一 个巧妙的方案。关键思想是通过用它们的质心替换一组远距离点来近似远程力。该方案显着 加快了计算速度，复杂度为 n log n 而不是 n 2。 Barnes-Hut 算法包括三个步骤：

  1. 构建空间索引（例如，四叉树）
  2. 计算质心
  3. 估计 force strength

- **Adjacent matrix（邻接矩阵）**

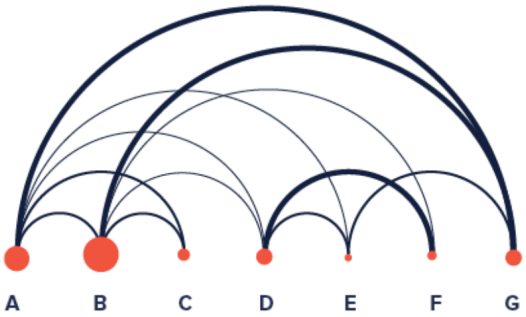
  优点

  - 消除边的交叉，因为边对应于不重叠的条目
  - 便于展示边的权重信息

  缺点

  - 行列的排列顺序对矩阵的可解释性影响大
  - 难以找到图中的路径
  - 受限于分辨率
  - 空间要求大

- **Arc-diagram**



节点表示实例，边表示实例之间的关系

优点

它可以展示不同实体之间的连接关系，研究连接的分布。具体而言

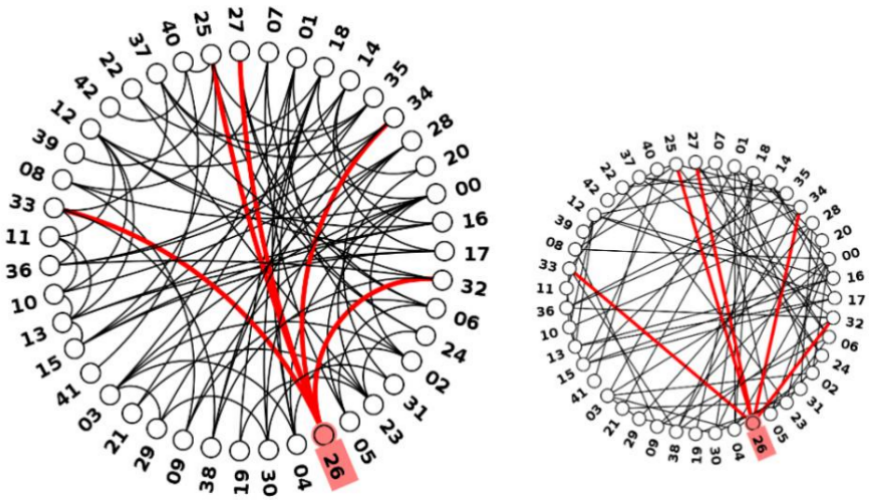
1. 在优化好节点顺序后，它可以凸显出集群。
2. 可以清晰的展示每个节点的label。

缺点

1. 不能展示拓扑结构
2. 在边数量较多的时候会导致视图比较混乱。

- **Circular layout（环形布局）**

  将节点放置在圆的圆周上，同时将边绘制为曲线而不是直线



节点的摆放顺序严重影响可视化表达是否清晰

树布局

RT算法

目的：

使用点线的结构描述出一棵树状图的结构，且更加合理的利用空间，保证节点分布的密度和图形的对称性

核心：

- 清楚标明每个节点的深度
- 保证边不出现交叉
- 子树的绘制保证有序且对称
- 使用尽可能小的空间

算法思路：

每个节点的坐标是(x, y)，y由这个节点的深度决定，x通过该深度

- 步骤一：自下而上，从子节点开始生成每个节点的位移

  从最下层的子节点开始，每次生成父节点时，合并左右两棵子树，保证两棵子树在不发生交叉的基础上尽可能的接近，保证子树的形态不发生改变，记录下每个节点的位移shift。

- 步骤二：自上而下，确定每个点的坐标

  从父节点向子节点，对每个节点位置上的位移值shift求和，计算出x坐标，根据每个节点的(x, y)的数值即可画出整棵点线树形图。

Tree Map

特点：

每棵树都以矩阵的形式呈现，矩阵被划分成多个更小的矩形对应其子节点，每次对矩形切片获得子矩阵。为了保证切片的均匀性，需要不断改变切片的方向。

算法：

```
Draw()
{
    从父级更改切片方向（水平or垂直）
    读取该目录下的所有节点信息和子目录
    为每一个节点分配矩形，并按照数据的比例进行缩放
    选择合适的颜色对矩形进行上色
    对于每一个子目录
        递归调用Draw()函数
}
```

改进——正方形树图

维护宽高比

思路：维护切片出来的矩形宽高比始终接近1:1，当宽高比变差时就改变切片方向。

优点：

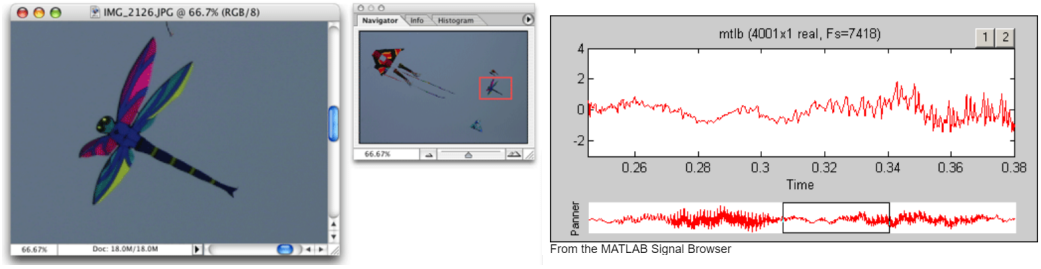
- 周长更小，减少边界墨水
- 用户易于对矩形进行选择查看
- 易于比较大小

交互

缩略-细节图(Overview-detail)模式

Overview-detail是一种将概览图和细节图相连接的交互方式，设计特点：两张图相互连接

- 缩略图和细节图一定是两种分开的图进行链接形成的缩略-细节交互模式
- 具有快捷导航的功能，保证能够从概览图向细节图进行浏览，且不会改变图上的细节
- 细节图中做出的更改可能无法立即在概览图上体现出来



焦点-上下文(Focus-Context)模式

Focus-Context焦点-上下文交互方式是将图中的基本信息和关注的信息(Focus)相结合的图一张复合图。

基本思想：

Focus-Context模式能够使得使用者能够看到呈现的主要关注对象，同时获得周围的可用信息。

使用场景，三个前提：

1. 用户需要浏览上下文信息和详细信息
2. 上下文信息和详细信息中显示的重点不同
3. 上下文信息和焦点信息在一个图中进行的组合，符合人类的视觉习惯。

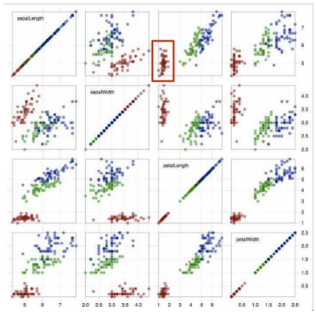
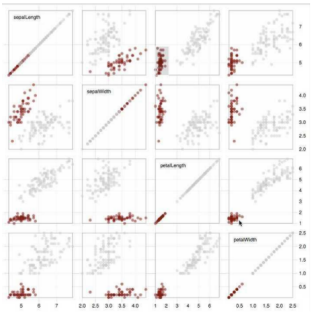
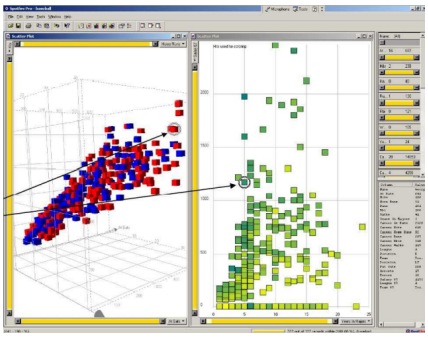
Overview-Detail和Focus- context图之间的区别性

- Overview-Detail概览-细节图中包括概览图和细节图两个图，并且在两个图之间进行链接。
- Focus-Context焦点-上下文图中只包括一个图，是将焦点信息和上下文信息在一个图中进行组合的

Brushing/Linking技术(画笔/链接)

brushing画笔/linking链接，都指代将同一数据的多个视图链接起来。基本功能：

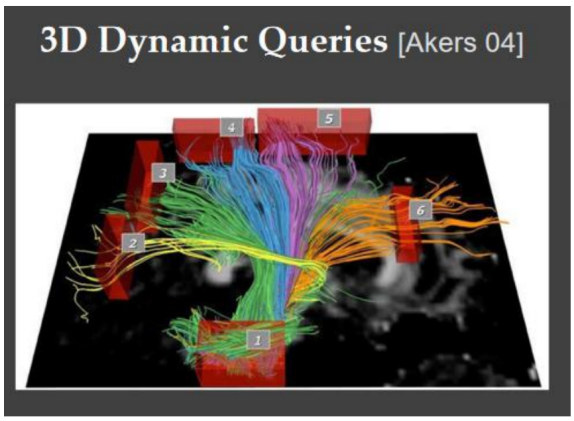
- 在某一个视图中选择突出显示的案例，在其他视图中页突出显示
- 移动鼠标到案例上，可以显示同一个数据在多个视图之间的对应关系
- 对一个视图中做的更改，在其他的视图中也会被修改

Dynamic Queries（动态查询）

Direct Manipulation（直接操控）

1. 对象和行为的可视化表示
2. 快速、渐进、可逆的动作
3. 通过点击交互（而不是打字）
4. 实时连续地展示



优点：

- 快速、简单、可逆（可恢复）
- 消除噪声
- 展现数据值得分布情况
- 方便人探索数据

缺点：

- 需要很多控制组件
- 显示的数据量受屏幕空间限制

三维体数据可视化方法

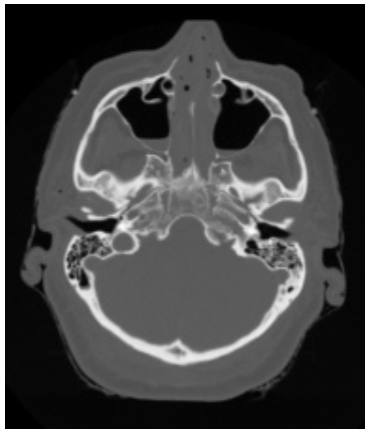
- **基于切片的方法**

最直接的解决方案，这意味着给予每个体数据切片滚动交互单独可视化机会。

优点：操作简单和复杂计算少。

缺点：可视化人员需要想象重建整个对象结构。

适合可视化已知对象的内部情况，比如，人体内部结构，不适合分析极其复杂和不明确结构。



- **其他技术仿真**

这种方法很适合于熟悉一定技术的专家可视化分析应用。比如，应用于医疗和地震行业的新技术开发，专家们可以从旧技术解决方案平稳过渡到现代化技术。此方法不常被采纳的原因如下: 首先，它需要使用非常详细的体数据集，而其它主要信息可能在通过模仿另一种技术时而丢失或损坏。 因此，在将新技术集成到专家工作流程中的过程中，可视化的普及将逐渐减少。其次，这种可视化类型的开发需要大量的时间才能接近可视化初始图像，在转换后部分图像将被丢弃使用。另外一个问题是需要有一定技术经验的人才能正确解释结果。

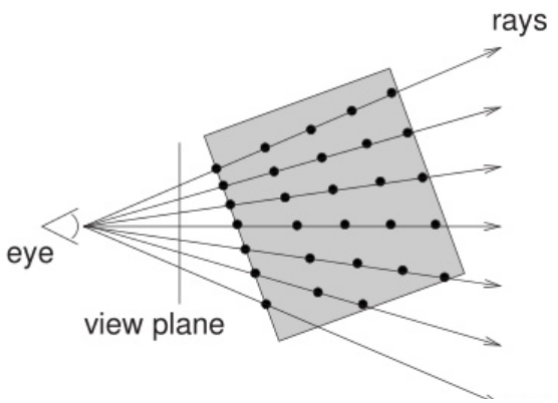
- **体渲染**

  3D渲染指用2D图像可视化3D对象。最常用的3D渲染基于多边形网格表面的逼真图像可视化。该技术被广泛应用，因为现代显卡架构加速应用操作。

  - 间接体渲染
  - 直接体渲染：直接体绘制不要求预处理。 直接从原始数据集观察数据，为算法提供了动态修改传递功能和阈值的机会。而且有些方法允许以半透明的方式可视化数据集的内部结构。

- **光线投射算法（raycasting）**

光线投射方法是基于图像序列的直接体绘制算法。从图像的每一个像素，沿固定方向（通常是视线方向）发射一条光线，光线穿越整个图像序列，并在这个过程中，对图像序列进行采样获取颜色信息，同时依据光线吸收模型将颜色值进行累加，直至光线穿越整个图像序列，最后得到的颜色值就是渲染图像的颜色。



Ray casting算法能被分为以下几个主要部分。首先，光线需要根据给定摄像机参数和各个像素位置设置好。然后，光线必须沿着射线步进，也就是需要实现一个循环。光学属性在循环内累加。最终当体被遍历过后，需要停止该过程。

$$C_{dst} = C_{dst} + (1 - \alpha_{dst})C_{src}$$

$$\alpha_{dst} = \alpha_{dst} + (1 - \alpha_{dst})\alpha_{src}$$

降维

PCA

它是一种统计方法。用于高维数据集的探索与可视化，还可用于数据的压缩和预 处理。可通过正交变换把具有相关性的高维变量转换为线性无关的低维变量，这 组低维变量称为主成分，它能保留原始数据的信息。

步骤：

1. 建立一个$N * d$的矩阵X，每一行x_n表示一条数据
2. 提取出每一个数据x_n的平均值
3. 计算X矩阵的协方差矩阵$\sum$，得到$\sum$的特征值和特征向量
4. 主成分就是最大特征值对应的特征向量

可视化结果不好原因：

1. 线性映射
2. 主要保留不同点之间距离的信息（方差尽可能大），但这并不是我们可视化降维的目的

多维缩放MDS

要求原始空间中样本间的距离在低维空间得以保持，通过利用对点（数据）做平移，旋转，翻转等操作，点的距离是不变的这一特性来对原始数据进行操作。要求高维空间（原始空间）中样本之间的距离在低维空间中得以保持。

SNE和t-SNE

特点：

- 与PCA相比，SNE侧重于保持最近的低维映射中的邻居(**主要是关注数据的局部结构**)
- SNE转换点之间的成对欧氏距离转化为概率密度
- 通过仿射(affinitie)变换将数据点映射到概率分布上

步骤：

- SNE构建一个高维对象之间的概率分布，使得相似的对象有更高的概率被选择，而不相似的对象有较低的概率被选择。
- SNE在低维空间里在构建这些点的概率分布，使得这两个概率分布之间尽可能的相似（通过原始空间和嵌入空间的联合概率的Kullback-Leibler（KL）散度来评估可视化效果的好坏，也就是说用有关KL散度的函数作为loss函数，然后通过梯度下降最小化loss函数，最终获得收敛结果。）

优点是：可以无监督的产生集群，可以处理可靠性数据

SNE使用条件概率计算两个点之间的相似度，使用高斯分布将距离转化为概率分布。

SNE的问题：1、难以优化。2、存在拥挤问题

t_SNE：

- 使用对称版的SNE，简化梯度公式
- 低维空间下，使用t分布替代高斯分布表达两点之间的相似度

参数：

perplexity，困惑度是一个全局参数，表示有效的邻居数

大规模数据可视化

两个挑战：

- 有效的视觉编码
- 实时交互

步骤：

- Bin（分箱）：

  将数据域划分为离散的“桶”

  类别：已经离散（但要注意高基数）

  数字：选择仓位间隔（均匀、分位数等）

  时间：选择时间单位：小时、日、月等。

  地理位置：地图投影后的x、y坐标

- 聚合计数（平滑）：

  总和、平均值、最小值、最大值。。。

- 绘制并可视化聚合

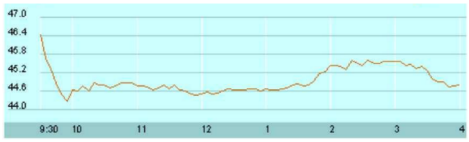
交互可扩展性策略

- 查询数据库
- 客户端索引/数据立方体
- 预取
- 近似值

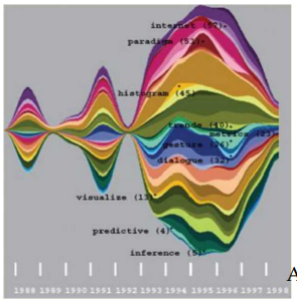
时序数据可视化化

方法：

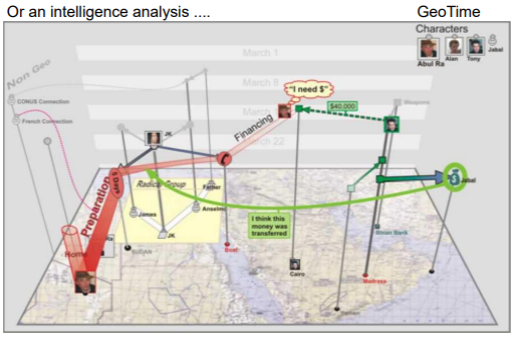
- 时间序列图（Time-Series Plot）



- 堆叠图（Stacked Graphs）



- 时间+地理空间（Time + Geo Spatial）



- 小倍数与动画（Small Multiples & Animation）