

法律声明

□ 本课件包括：演示文稿，示例，代码，题库，视频和声音等，小象学院拥有完全知识产权的权利；只限于善意学习者在本课程使用，不得在课程范围外向任何第三方散播。任何其他人或机构不得盗版、复制、仿造其中的创意，我们将保留一切通过法律手段追究违反者的权利。

□ 课程详情请咨询

■ 微信公众号：小象学院

■ 新浪微博：小象AI学院



矩阵和线性代数



小象学院
ChinaHadoop.cn

邹博

主要内容

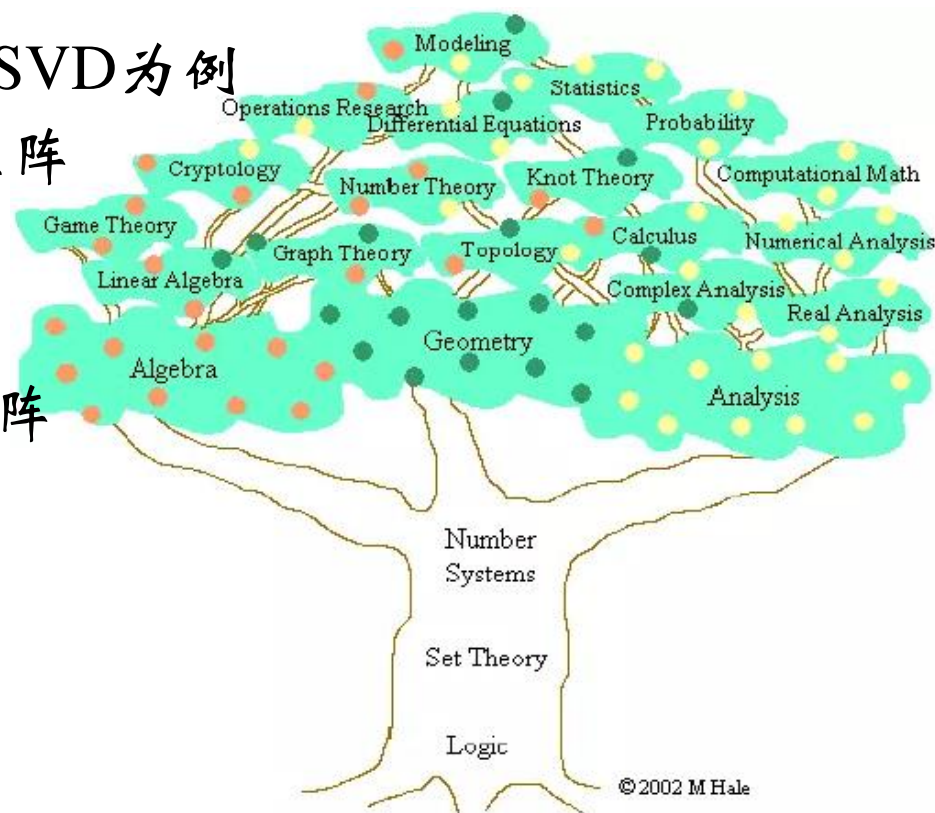
□ 矩阵

- 线性代数是有益的：以SVD为例
- 矩阵的乘法/状态转移矩阵
- 矩阵和向量组

□ 特征值和特征向量

- 对称阵、正交阵、正定阵
- 数据白化
- 正交基
- QR分解/LFM

□ 矩阵与向量间求导



SVD的提法

$$(A^T \cdot A)v_i = \lambda_i v_i \Rightarrow \begin{cases} \sigma_i = \sqrt{\lambda_i} \\ u_i = \frac{1}{\sigma_i} A \cdot v_i \end{cases} \Rightarrow A = U \Sigma V^T$$

□ 奇异值分解(Singular Value Decomposition)是一种重要的矩阵分解方法，可以看做对称方阵在任意矩阵上的推广。

■ Singular: 突出的、奇特的、非凡的

■ 似乎更应该称之为“**优值分解**”

□ 假设A是一个 $m \times n$ 阶实矩阵，则存在一个分解使得：

$$A_{m \times n} = U_{m \times m} \Sigma_{m \times n} V_{n \times n}^T$$

■ 通常将奇异值由大而小排列。这样， Σ 便能由A唯一确定了。

□ 与特征值、特征向量的概念相对应：

■ Σ 对角线上的元素称为矩阵A的奇异值；

■ U的第i列称为A的关于 σ_i 的左奇异向量；

■ V的第i列称为A的关于 σ_i 的右奇异向量。

SVD举例 $A_{m \times n} = U_{m \times m} \Sigma_{m \times n} V_{n \times n}^T$

□ 已知4×5阶实矩阵A，求A的SVD分解：

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & 2 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 \end{bmatrix}$$

$$U = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

$$\Sigma = \begin{bmatrix} 4 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & \sqrt{5} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$V^T = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \sqrt{0.2} & 0 & 0 & 0 & \sqrt{0.8} \\ 0 & 0 & 0 & 1 & 0 \\ -\sqrt{0.8} & 0 & 0 & 0 & \sqrt{0.2} \end{bmatrix}$$

□ 矩阵U和V都是单位正交方阵： $U^T U = I$ ， $V^T V = I$

SVD

```
def restore(sigma, u, v, K): # 奇异值、左特征向量、右特征向量
    print K
    m = len(u)
    n = len(v[0])
    a = np.zeros((m, n))
    for k in range(K+1):
        for i in range(m):
            a[i] += sigma[k] * u[i][k] * v[k]
    b = a.astype('uint8')
    Image.fromarray(b).save("svd_" + str(K) + ".png")
```

奇异值分解-效果



SVD



svd_0.png



svd_1.png



svd_2.png



svd_3.png



svd_4.png



svd_5.png



svd_6.png



svd_7.png



svd_8.png



svd_9.png



svd_10.png



svd_11.png



svd_12.png



svd_13.png



svd_14.png



svd_15.png



svd_16.png



svd_17.png



svd_18.png



svd_19.png



svd_20.png



svd_21.png



svd_22.png



svd_23.png



svd_24.png



svd_25.png



svd_26.png



svd_27.png



svd_28.png



svd_29.png



svd_30.png



svd_31.png



svd_32.png



svd_33.png



svd_34.png



svd_35.png



svd_36.png



svd_37.png



svd_38.png



svd_39.png



svd_40.png



svd_41.png



svd_42.png



svd_43.png



svd_44.png



svd_45.png



svd_46.png



svd_47.png



svd_48.png



svd_49.png

线性代数

□ 定义：方阵的行列式

- 1阶方阵的行列式为该元素本身
- n 阶方阵的行列式等于它的任一行(或列)的各元素与其对应的代数余子式乘积之和。

方阵的行列式

- 1×1 的方阵，其行列式等于该元素本身。

$$A = (a_{11}) \quad |A| = a_{11}$$

- 2×2 的方阵，其行列式用主对角线元素乘积减去次对角线元素的乘积。

$$A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$$

$$|A| = a_{11}a_{22} - a_{12}a_{21}$$

方阵的行列式

□ 3×3 的方阵：

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$$

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} & | & a_{11} & a_{12} & a_{13} & | & a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} & | & a_{21} & a_{22} & a_{23} & | & a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} & | & a_{31} & a_{32} & a_{33} & | & a_{31} & a_{32} & a_{33} \end{pmatrix}$$

Diagram illustrating the calculation of the determinant for a 3x3 matrix. The matrix is shown with its elements. Blue dashed lines represent the three positive terms in the expansion: $a_{11}a_{22}a_{33}$, $a_{12}a_{23}a_{31}$, and $a_{13}a_{21}a_{32}$. Red solid lines represent the three negative terms: $a_{11}a_{23}a_{32}$, $a_{12}a_{21}a_{33}$, and $a_{13}a_{22}a_{31}$.

□ 根据“主对角线元素乘积减去次对角线元素的乘积”的原则，得：

$$|A| = a_{11}a_{22}a_{33} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32} - a_{11}a_{23}a_{32} - a_{12}a_{21}a_{33} - a_{13}a_{22}a_{31}$$

代数余子式

□ 在一个n阶行列式A中,把(i,j)元素 a_{ij} 所在的第i行和第j列划去后,留下的n-1阶方阵的行列式叫做元素 a_{ij} 的余子式,记作 M_{ij} 。

□ 代数余子式: $A_{ij} = (-1)^{i+j} M_{ij}$

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \quad \forall 1 \leq j \leq n, |A| = \sum_{i=1}^n a_{ij} \cdot (-1)^{i+j} M_{ij}$$
$$\quad \quad \quad \forall 1 \leq i \leq n, |A| = \sum_{j=1}^n a_{ij} \cdot (-1)^{i+j} M_{ij}$$

$$|A| = a_{11}a_{22}a_{33} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32} - a_{11}a_{23}a_{32} - a_{12}a_{21}a_{33} - a_{13}a_{22}a_{31}$$

伴随矩阵

- 对于 $n \times n$ 方阵的任意元素 a_{ij} 都有各自的代数余子式 $A_{ij} = (-1)^{i+j} M_{ij}$ ，构造 $n \times n$ 的方阵 A^* ：

$$A^* = \begin{pmatrix} A_{11} & A_{21} & \cdots & A_{n1} \\ A_{12} & A_{22} & \cdots & A_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ A_{1n} & A_{2n} & \cdots & A_{nn} \end{pmatrix}$$

- A^* 称为 A 的伴随矩阵。

- 注意 A_{ij} 位于 A^* 的第 j 行第 i 列

方阵的逆 $A \cdot A^* = |A| \cdot I$

□ 由前述结论: $\forall 1 \leq i \leq n, |A| = \sum_{j=1}^n a_{ij} \cdot (-1)^{i+j} M_{ij}$

□ 根据:

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix} \quad A^* = \begin{pmatrix} A_{11} & A_{21} & \cdots & A_{n1} \\ A_{12} & A_{22} & \cdots & A_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ A_{1n} & A_{2n} & \cdots & A_{nn} \end{pmatrix}$$

□ 计算:

$$A \cdot A^* = \begin{pmatrix} |A| & 0 & \cdots & 0 \\ 0 & |A| & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & |A| \end{pmatrix} = |A| \cdot I \Rightarrow A^{-1} = \frac{1}{|A|} A^*$$

■ 思考: 该等式有什么用?

范德蒙行列式Vandermonde

□ 证明范德蒙行列式Vandermonde:

$$D_n = \begin{vmatrix} 1 & 1 & 1 & \cdots & 1 \\ x_1 & x_2 & x_3 & \cdots & x_n \\ x_1^2 & x_2^2 & x_3^2 & \cdots & x_n^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_1^{n-1} & x_2^{n-1} & x_3^{n-1} & \cdots & x_n^{n-1} \end{vmatrix} = \prod_{i,j(n \geq i > j \geq 1)} (x_i - x_j)$$

■ 提示：数学归纳法

■ 注：参考Lagrange/Newton插值法

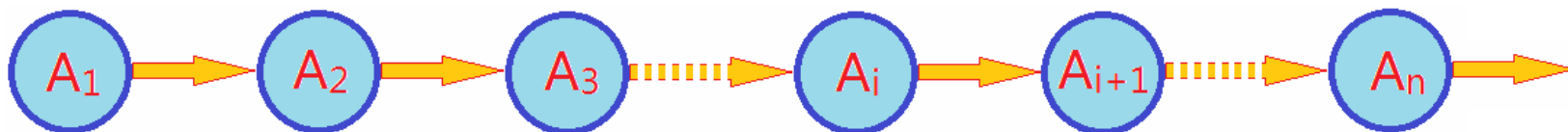
矩阵的乘法

□ A为 $m \times s$ 阶的矩阵，B为 $s \times n$ 阶的矩阵，那么， $C=A \times B$ 是 $m \times n$ 阶的矩阵，其中，

$$c_{ij} = \sum_{k=1}^s a_{ik} b_{kj}$$

矩阵模型

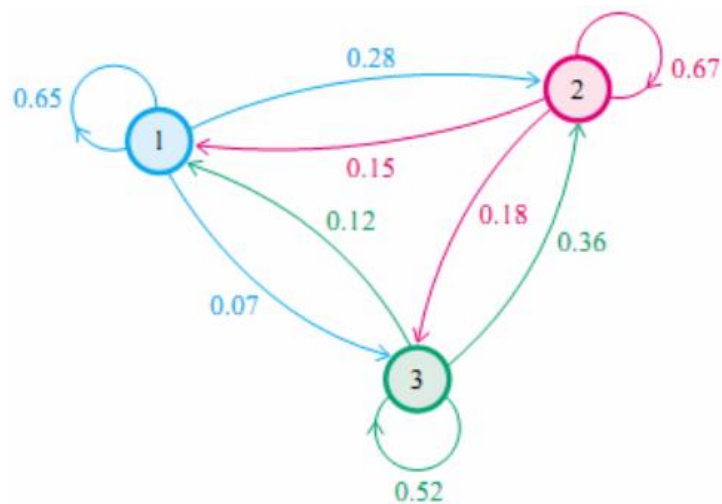
- 考虑某随机过程 π ，它的状态有 n 个，用 $1\sim n$ 表示。记在当前时刻 t 时位于 i 状态，它在 $t+1$ 时刻位于 j 状态的概率为 $P(i,j)=P(j|i)$ ：
 - 即状态转移的概率只依赖于前一个状态。



举例

□ 假定按照经济状况将人群分成上、中、下三个阶层，用1、2、3表示。假定当前处于某阶层只和上一代有关，即：考察父代为第*i*阶层，则子代为第*j*阶层的概率。假定为如下转移概率矩阵：

P	子代
父代	$\begin{bmatrix} 0.65 & 0.28 & 0.07 \\ 0.15 & 0.67 & 0.18 \\ 0.12 & 0.36 & 0.52 \end{bmatrix}$



概率转移矩阵

□ 第 $n+1$ 代中处于第 j 个阶层的概率为：

$$\pi(X_{n+1} = j) = \sum_{i=1}^K \pi(X_n = i) \cdot P(X_{n+1} = j | X_n = i)$$

$$\Rightarrow \pi^{(n+1)} = \pi^{(n)} \cdot P$$

□ 因此，矩阵 P 即为(条件)概率转移矩阵。

■ 第 i 行元素表示：在上一个状态为 i 时的分布概率，
即：每一行元素的和为1。

□ 思考：初始概率分布 π 对最终分布的影响？

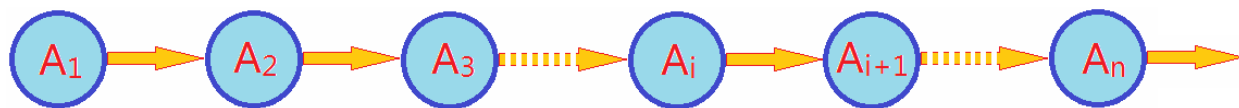
探索：初始概率 $\pi = [0.21, 0.68, 0.1]$ 迭代

第n代	第1阶层	第2阶层	第3阶层
0	0.21	0.68	0.11
1	0.252	0.554	0.194
2	0.27	0.512	0.218
3	0.278	0.497	0.225
4	0.282	0.49	0.226
5	0.285	0.489	0.225
6	0.286	0.489	0.225
7	0.286	0.489	0.225
8	0.286	0.488	0.225
9	0.286	0.489	0.225
10	0.286	0.489	0.225

初始概率 $\pi = [0.75, 0.15, 0.1]$ 的迭代结果

第n代	第1阶层	第2阶层	第3阶层
0	0.75	0.15	0.1
1	0.522	0.347	0.132
2	0.407	0.426	0.167
3	0.349	0.459	0.192
4	0.318	0.475	0.207
5	0.303	0.482	0.215
6	0.295	0.485	0.22
7	0.291	0.487	0.222
8	0.289	0.488	0.225
9	0.286	0.489	0.225
10	0.286	0.489	0.225

平稳分布



- 初始概率不同，但经过若干次迭代， π 最终稳定收敛在某个分布上。
- 从而，这是转移概率矩阵 P 的性质，而非初始分布的性质。事实上，上述矩阵 P 的 n 次幂，每行都是 $(0.286, 0.489, 0.225)$ ， $n > 20$
- 如果一个非周期马尔科夫随机过程具有转移概率矩阵 P ，且它的任意两个状态都是连通的，则 $\lim_{n \rightarrow \infty} P_{ij}^n$ 存在，记做 $\lim_{n \rightarrow \infty} P_{ij}^n = \pi(j)$ 。

平稳分布

□ 事实上，下面两种写法等价：

$$\lim_{n \rightarrow \infty} P_{ij}^n = \pi(j) \qquad \lim_{n \rightarrow \infty} P^n = \begin{bmatrix} \pi(1) & \pi(2) & \dots & \pi(n) \\ \pi(1) & \pi(2) & \dots & \pi(n) \\ \vdots & \vdots & \ddots & \vdots \\ \pi(1) & \pi(2) & \dots & \pi(n) \end{bmatrix}$$

□ 同时，若某概率分布 $\pi P = \pi$ ，说明

- 该多项分布 π 是状态转移矩阵 P 的平稳分布；
- 线性方程 $\mathbf{x}P = \mathbf{x}$ 的非负解为 π ，而 P^n 唯一，因此 π 是线性方程 $\mathbf{x}P = \mathbf{x}$ 的唯一非负解。
- 该问题将在马尔科夫模型中继续探讨。

思考

- 根据定义来计算 $C=A \times B$ ，需要 $m*n*s$ 次乘法。
 - 即：若 A 、 B 都是 n 阶方阵， C 的计算时间复杂度为 $O(n^3)$
 - 问：可否设计更快的算法？
- 三个矩阵 A 、 B 、 C 的阶分别是 $a_0 \times a_1$ ， $a_1 \times a_2$ ， $a_2 \times a_3$ ，从而 $(A \times B) \times C$ 和 $A \times (B \times C)$ 的乘法次数是 $a_0 a_1 a_2 + a_0 a_2 a_3$ 、 $a_1 a_2 a_3 + a_0 a_1 a_3$ ，二者一般情况是不相等的。
 - 问：给定 n 个矩阵的连乘积： $A_1 \times A_2 \times A_3 \dots \times A_n$ ，如何添加括号来改变计算次序，使得乘法的计算量最小？

Code

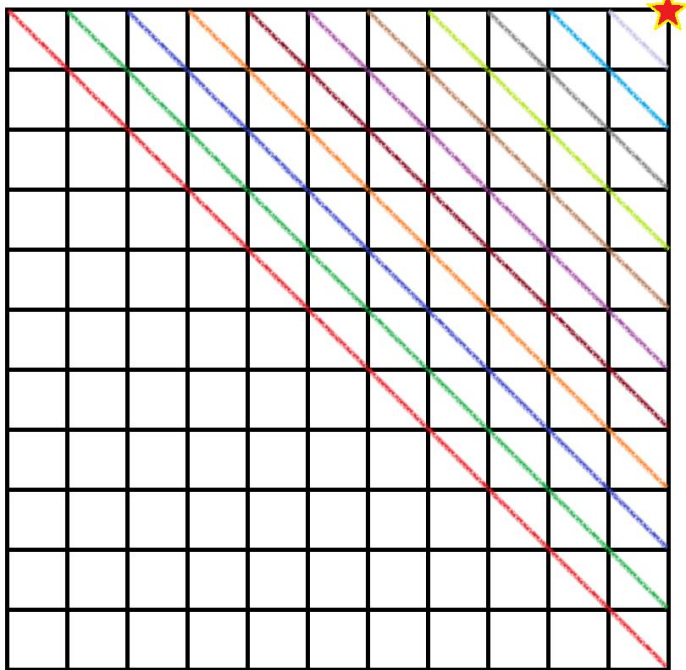
```
//p[0...n]存储了n+1个数，其中，(p[i-1],p[i])是矩阵i的阶；  
//s[i][j]记录A[i...j]从什么位置断开；m[i][j]记录数乘最小值  
void MatrixMultiply(int* p, int n, int** m, int** s)
```

```
{  
    int r, i, j, k, t;  
    for (i = 1; i <= n; i++)  
        m[i][i] = 0;
```

//r个连续矩阵的连乘：上面的初始化，相当于r=1

```
for (r = 2; r <= n; r++)
```

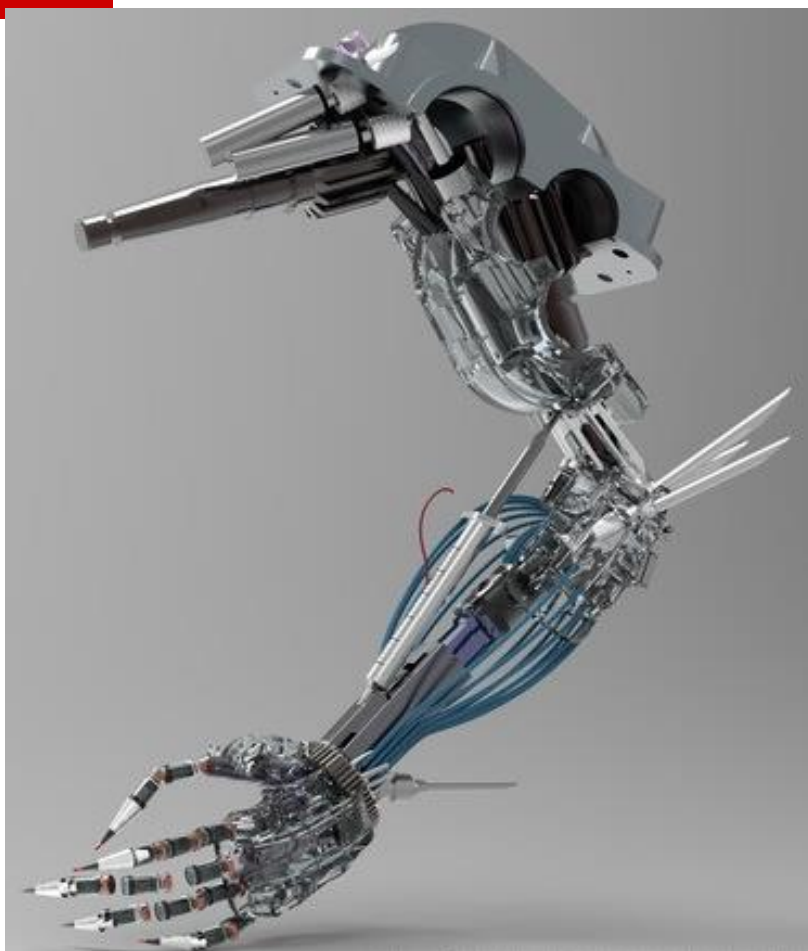
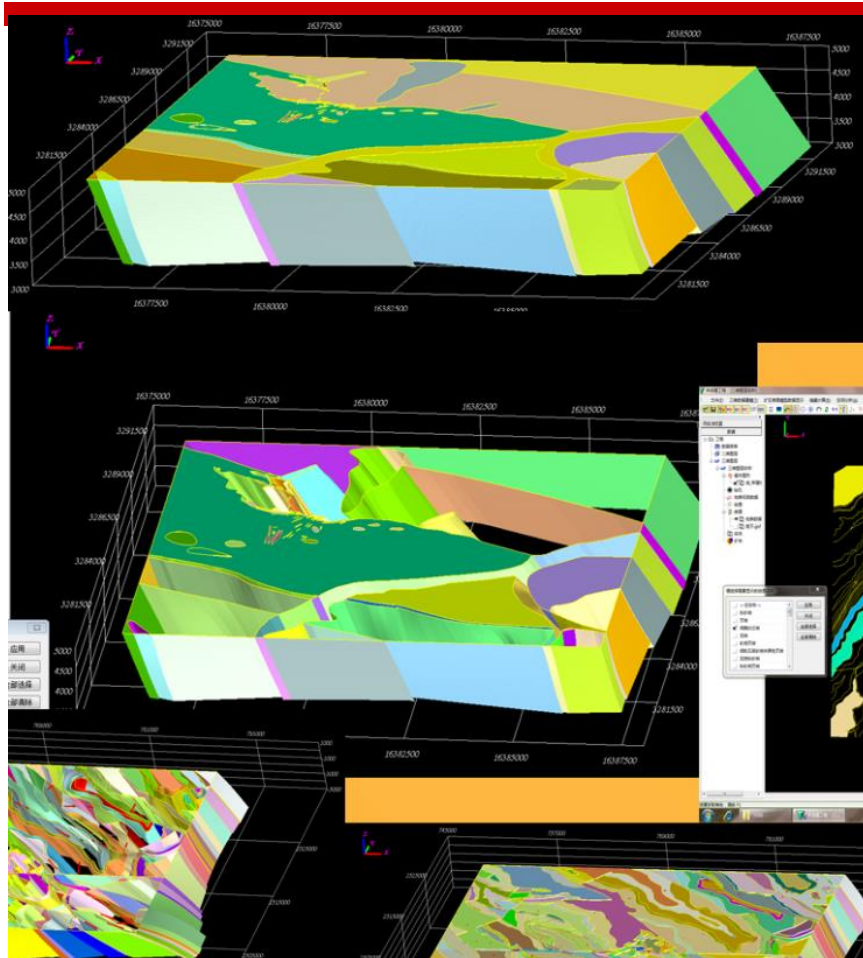
```
{  
    for (i = 1; i <= n-r+1; i++)  
    {  
        j=i+r-1;  
        m[i][j] = m[i+1][j] + p[i-1]*p[i]*p[j];  
        s[i][j] = i;  
        for (k = i+1; k < j; k++)  
        {  
            t = m[i][k] + m[k+1][j] + p[i-1]*p[k]*p[j];  
            if (t < m[i][j])  
            {  
                m[i][j] = t;  
                s[i][j] = k;  
            }  
        }  
    }  
}
```



矩阵和向量的乘法

- A 为 $m \times n$ 的矩阵， \mathbf{x} 为 $n \times 1$ 的列向量，则 $A\mathbf{x}$ 为 $m \times 1$ 的列向量，记 $\vec{y} = A \cdot \vec{x}$
- 由于 n 维列向量和 n 维空间的点一一对应，上式实际给出了从 n 维空间的点到 m 维空间点的线性变换。
 - 旋转、平移(齐次坐标下)
- 特殊的，若 $m=n$ ，且 $A\mathbf{x}$ 完成了 n 维空间内的线性变换。

矩阵和向量的乘法应用



矩阵的秩

- 在 $m \times n$ 矩阵 A 中，任取 k 行 k 列，不改变这 k^2 个元素在 A 中的次序，得到 k 阶方阵，称为矩阵 A 的 k 阶子式。
 - 显然， $m \times n$ 矩阵 A 的 k 阶子式有 $C_m^k C_n^k$ 个。
- 设在矩阵 A 中有一个不等于 0 的 r 阶子式 D ，且所有 $r+1$ 阶子式(如果存在的话)全等于 0，那么， D 称为矩阵 A 的最高阶非零子式， r 称为矩阵 A 的秩，记做 $R(A)=r$ 。
 - $n \times n$ 的可逆矩阵，秩为 n
 - 可逆矩阵又称满秩矩阵
 - 矩阵的秩等于它行(列)向量组的秩

秩与线性方程组的解的关系

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \dots\dots\dots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = b_m \end{cases} \Rightarrow A\vec{x} = \vec{b}$$

□ 对于n元线性方程组 $Ax=b$,

- 无解的充要条件是 $R(A) < R(A, b)$
- 有唯一解的充要条件是 $R(A) = R(A, b) = n$
- 有无限多解的充要条件是 $R(A) = R(A, b) < n$

推论

- $Ax=0$ 有非零解的充要条件是 $R(A) < n$
- $Ax=b$ 有解的充要条件是 $R(A) = R(A, b)$

向量组等价

- 向量 b 能由向量组 $A:a_1, a_2, \dots, a_m$ 线性表示的充要条件是矩阵 $A=(a_1, a_2, \dots, a_m)$ 的秩等于矩阵 $B=(a_1, a_2, \dots, a_m, b)$ 的秩。
- 设有两个向量组 $A:a_1, a_2, \dots, a_m$ 及 $B:b_1, b_2, \dots, b_n$ ，若 B 组的向量都能由向量组 A 线性表示，则称向量组 B 能由向量组 A 线性表示。若向量组 A 与向量组 B 能相互线性表示，则称两个向量组等价。

系数矩阵

□ 将向量组A和B所构成的矩阵依次记做
 $A=(a_1, a_2, \dots, a_m)$ 和 $B=(b_1, b_2, \dots, b_n)$, B组能由A组
线性表示, 即对每个向量 b_j , 存在 $k_{1j}, k_{2j}, \dots, k_{mj}$

□ 使得

$$b_j = k_{1j}a_1 + k_{2j}a_2 + \dots + k_{mj}a_m = (a_1 \ a_2 \ \dots \ a_m) \begin{pmatrix} k_{1j} \\ k_{2j} \\ \vdots \\ k_{mj} \end{pmatrix}$$

□ 从而得到系数矩阵K

$$(b_1 \ b_2 \ \dots \ b_n) = (a_1 \ a_2 \ \dots \ a_m) \begin{pmatrix} k_{11} & k_{12} & \dots & k_{1n} \\ k_{21} & k_{22} & \dots & k_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ k_{m1} & k_{m2} & \dots & k_{mn} \end{pmatrix}$$

对 $C=AB$ 的重认识

- 由此可知，若 $C=A \times B$ ，则矩阵 C 的列向量能由 A 的列向量线性表示， B 即为这一表示的系数矩阵。
- 对偶的，若 $C=A \times B$ ，则矩阵 C 的行向量能由 B 的行向量线性表示， A 即为这一表示的系数矩阵
- 向量组 $B: b_1, b_2, \dots, b_n$ 能由向量组 $A: a_1, a_2, \dots, a_m$ 线性表示的充要条件是矩阵 $A=(a_1, a_2, \dots, a_m)$ 的秩等于矩阵 $(A, B)=(a_1, a_2, \dots, a_m, b_1, b_2, \dots, b_n)$ 的秩，即： $R(A)=R(A, B)$ 。

正交阵

- 若 n 阶矩阵 A 满足 $A^T A = I$ ，称 A 为正交矩阵，简称正交阵。
 - A 是正交阵的充要条件： A 的列(行)向量都是单位向量，且两两正交。
- A 是正交阵， x 为向量，则 $A \cdot x$ 称作正交变换。
 - 正交变换不改变向量长度

思考

- 若A、B都是n阶正交阵，那么， $A \times B$ 是正交阵吗？
- 正交阵和对称阵，能够通过何种操作获得一定意义下的联系？

特征值和特征向量

- A 是 n 阶矩阵，若数 λ 和 n 维非 0 列向量 x 满足 $Ax = \lambda x$ ，那么，数 λ 称为 A 的特征值， x 称为 A 的对应于特征值 λ 的特征向量。
- 根据定义，立刻得到 $(A - \lambda I)x = 0$ ，令关于 λ 的多项式 $|A - \lambda I|$ 为 0，方程 $|A - \lambda I| = 0$ 的根为 A 的特征值；将根 λ_0 带入方程组 $(A - \lambda I)x = 0$ ，求得到的非零解，即 λ_0 对应的特征向量。

特征值的性质

- 设 n 阶矩阵 $A=(a_{ij})$ 的特征值为 $\lambda_1, \lambda_2, \dots, \lambda_n$, 则
- $\lambda_1 + \lambda_2 + \dots + \lambda_n = a_{11} + a_{22} + \dots + a_{nn}$
- $\lambda_1 \lambda_2 \dots \lambda_n = |A|$
 - 矩阵 A 主行列式的元素和, 称作矩阵 A 的迹。

思考

□ 已知 λ 是方阵 A 的特征值,

□ 则

■ λ^2 是 A^2 的特征值

■ A 可逆时, λ^{-1} 是 A^{-1} 的特征值。

■ 提示: 定义

不同特征值对应的特征向量

□ 设 $\lambda_1, \lambda_2, \dots, \lambda_m$ 是方阵 A 的 m 个特征值， p_1, p_2, \dots, p_m 是依次与之对应的特征向量，若 $\lambda_1, \lambda_2, \dots, \lambda_m$ 各不相同，则 p_1, p_2, \dots, p_m 线性无关。

□ 总结与思考

■ 不同特征值对应的特征向量，线性无关。

■ 若方阵 A 是对称阵呢？结论是否会加强？

□ 协方差矩阵、二次型矩阵、无向图邻接矩阵等：对称阵

引理

□ 实对称阵的特征值是实数

- 设复数 λ 为对称阵 A 的特征值，复向量 x 为对应的特征向量，即 $Ax=\lambda x(x\neq 0)$
- 用 $\bar{\lambda}$ 表示 λ 的共轭复数， \bar{x} 表示 x 的共轭复向量，而 A 是实矩阵，有 $\bar{A}=A$
- 下面给出证明过程。

证明

□ 首先 $A\bar{x} = \bar{A}\bar{x} = \overline{Ax} = \bar{\lambda}x = \bar{\lambda}\bar{x}$

□ 因为 $\bar{x}^T(Ax) = \bar{x}^T \lambda x = \lambda \bar{x}^T x$

$$\bar{x}^T(Ax) = (\bar{x}^T A^T)x = (A\bar{x})^T x = (\bar{\lambda}\bar{x})^T x = \bar{\lambda}\bar{x}^T x$$

□ 从而

$$\lambda \bar{x}^T x = \bar{\lambda} \bar{x}^T x \Rightarrow (\lambda - \bar{\lambda}) \bar{x}^T x = 0$$

□ 而

$$\bar{x}^T x = \sum_{i=1}^n \bar{x}_i x_i = \sum_{i=1}^n |x_i|^2 \neq 0$$

□ 所以

$$\lambda - \bar{\lambda} = 0 \Rightarrow \lambda = \bar{\lambda}$$

利用上述结论很快得到

- 将实数 λ 带入方程组 $(A - \lambda I)x = 0$ ，该方程组为实系数方程组，因此，**实对称阵**的特征向量可以取**实向量**。

实对称阵不同特征值的特征向量正交

□ 令实对称矩阵为A，其两个不同的特征值 λ_1, λ_2 对应的特征向量分别是 μ_1, μ_2 ；

■ $\lambda_1, \lambda_2, \mu_1, \mu_2$ 都是实数或是实向量。

$$\begin{aligned} & \begin{cases} A\mu_1 = \lambda_1\mu_1 \\ A\mu_2 = \lambda_2\mu_2 \Rightarrow \mu_1^T A\mu_2 = \mu_1^T \lambda_2\mu_2 \end{cases} \\ & \Rightarrow (A^T \mu_1)^T \mu_2 = \lambda_2 \mu_1^T \mu_2 \Rightarrow (\underline{A\mu_1})^T \mu_2 = \lambda_2 \mu_1^T \mu_2 \\ & \Rightarrow (\underline{\lambda_1 \mu_1})^T \mu_2 = \lambda_2 \mu_1^T \mu_2 \\ & \Rightarrow \lambda_1 \mu_1^T \mu_2 = \lambda_2 \mu_1^T \mu_2 \\ & \xrightarrow{\lambda_1 \neq \lambda_2} \mu_1^T \mu_2 = 0 \end{aligned}$$

最终结论

□ 设A为n阶**对称阵**，则必有**正交阵P**，使得

$$P^{-1}AP = P^T AP = \Lambda$$

- Λ 是以A的n个特征值为对角元的对角阵。
- 该变换称为“合同变换”，A和 Λ 互为合同矩阵。

□ 在谱聚类、PCA等章节中将会继续讨论

白化/漂白whitening

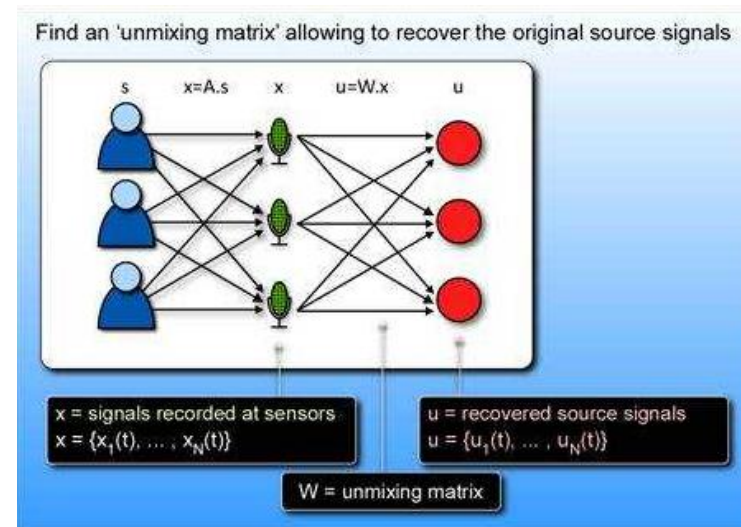
$$x = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1,m} \\ x_{21} & x_{22} & \cdots & x_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{n,m} \end{bmatrix} \stackrel{\Delta}{=} (x_1 \quad x_2 \quad \cdots \quad x_m)$$

□ 计算观测数据 x 的 $n \times n$ 的对称阵 $x \cdot x^T$ 的特征值和特征向量，用特征值形成对角阵 D ，特征向量形成正交阵 U ，则： $x \cdot x^T = U^T D U$

□ 令： $\tilde{x} = U^T D^{-0.5} U \cdot x$

□ 则：

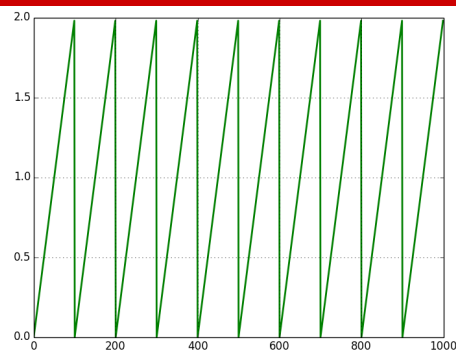
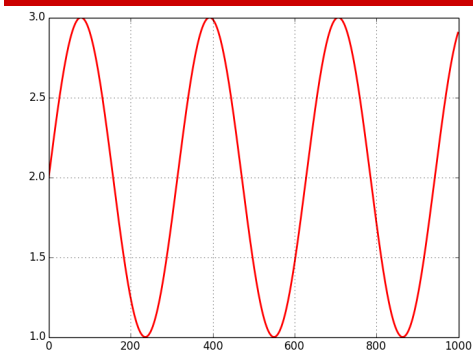
$$\begin{aligned} \tilde{x} \cdot \tilde{x}^T &= (U^T D^{-0.5} U \cdot x) (U^T D^{-0.5} U \cdot x)^T \\ &= (U^T D^{-0.5} U \cdot x) (x^T U^T D^{-0.5} U) \\ &= U^T D^{-0.5} U \cdot (x x^T) \cdot U^T D^{-0.5} U \\ &= U^T D^{-0.5} U \cdot U^T D U \cdot U^T D^{-0.5} U = I \end{aligned}$$



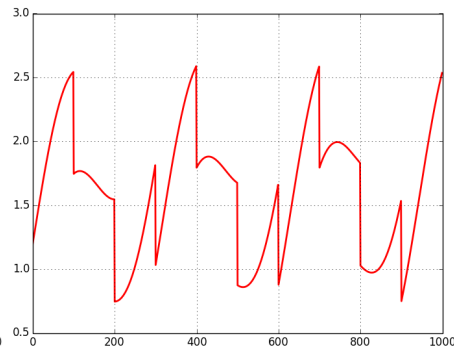
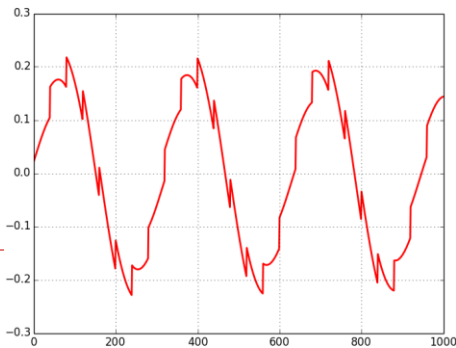
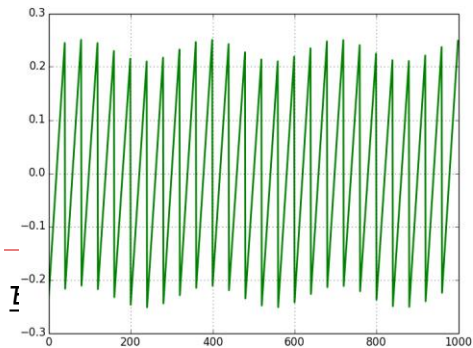
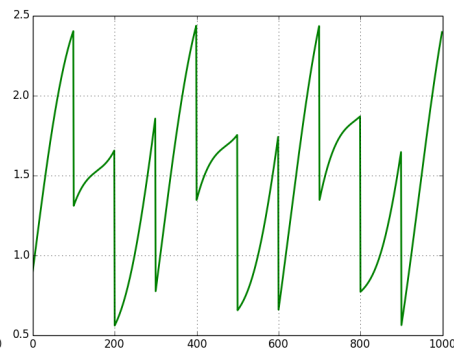
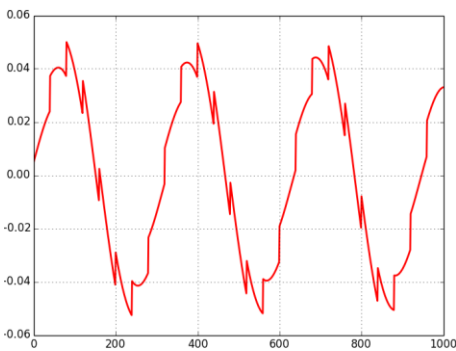
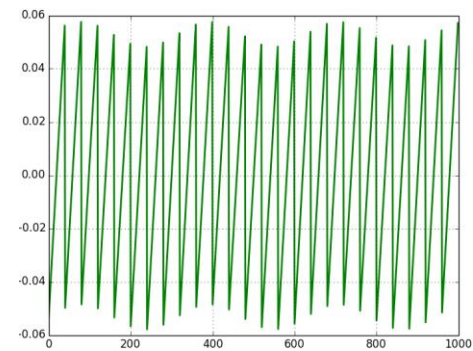
白化Code

```
def whitening(x):
    m = len(x)
    n = len(x[0])
    # 计算  $x \cdot x'$ 
    xx = [[0.0]*n for tt in range(n)]
    for i in range(n):
        for j in range(i, n):
            s = 0.0
            for k in range(m):
                s += x[k][i] * x[k][j]
            xx[i][j] = s
            xx[j][i] = s
    # 计算  $x \cdot x'$  的特征值和特征向量
    lamda, egs = np.linalg.eig(xx)
    lamda = [1/math.sqrt(d) for d in lamda]
    # 计算白化矩阵  $U'D^{(-0.5)}U$ 
    t = [[0.0]*n for tt in range(n)]
    for i in range(n):
        for j in range(n):
            t[i][j] = lamda[j] * egs[i][j]
    whiten_matrix = [[0.0]*n for tt in range(n)]
    for i in range(n):
        for j in range(n):
            s = 0.0
            for k in range(n):
                s += t[i][k] * egs[j][k]
            whiten_matrix[i][j] = s
    # 白化x
    wx = [0.0]*n
    for j in range(m):
        for i in range(n):
            s = 0.0
            for k in range(n):
                s += whiten_matrix[i][k] * x[j][k]
            wx[i] = s
    x[j] = wx[:]
```

增加白化后的ICA效果



源信号1	源信号2	
白化信号1	白化信号2	混合信号1
独立成分1	独立成分2	混合信号2



去均值ICA分离

源信号1

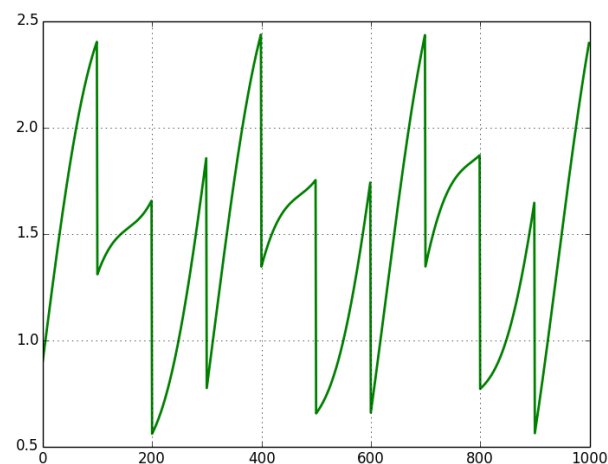
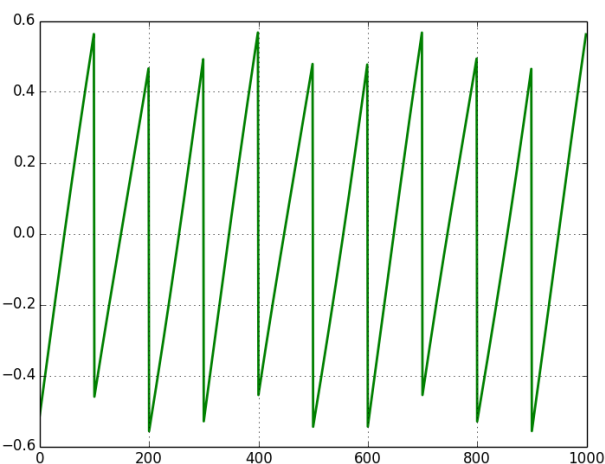
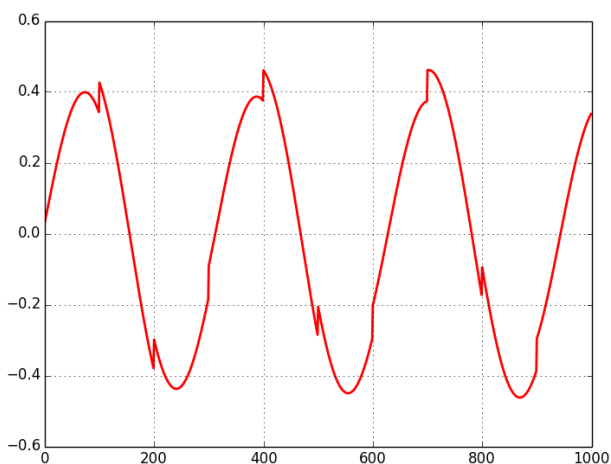
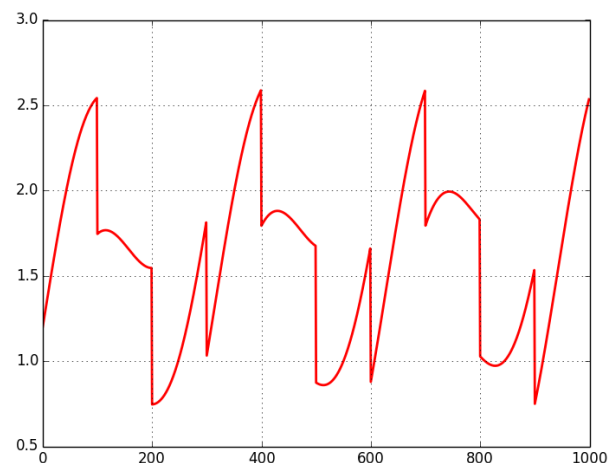
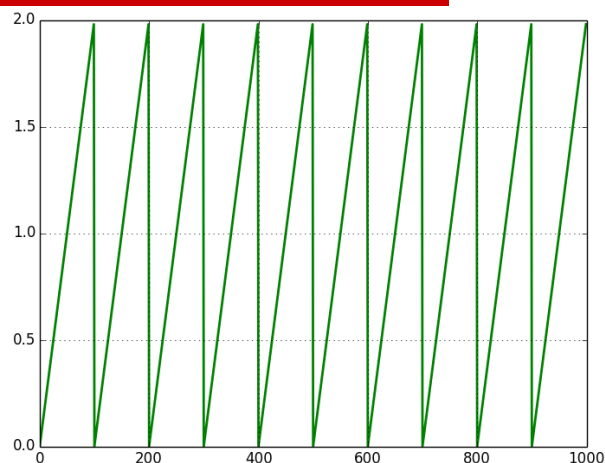
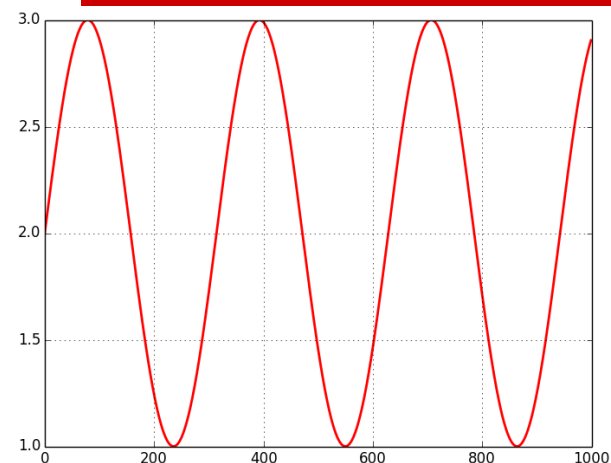
源信号2

混合信号1

独立成分1

独立成分2

混合信号2



正定阵

- 对于 n 阶方阵 A ，若任意 n 阶向量 x ，都有 $x^T A x > 0$ ，则称 A 是正定阵。
 - 若条件变成 $x^T A x \geq 0$ ，则 A 称作半正定阵
 - 类似还有负定阵，半负定阵。

思考

- 给定任意 $m \times n$ 的矩阵 A ，证明 $A^T A$ 一定是半正定方阵。
- 该结论在线性回归中将用到。

正定阵的判定

- 对称阵A为正定阵；
- A的特征值都为正；
- A的顺序主子式大于0；
- 以上三个命题等价。

$$(a_{11}) \quad \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \quad \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$$

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix}$$

例题：

□ 给定凸锥的定义如下：

C 为凸锥 $\Leftrightarrow \forall x_1, x_2 \in C, \theta_1, \theta_2 \geq 0$, 有 $\theta_1 x_1 + \theta_2 x_2 \in C$

□ 试证明：n阶半正定方阵的集合为凸锥。

利用定义证明

□ 若 A 、 B 为 n 阶半正定阵，则

$$\forall \vec{z}, \vec{z}^T A \vec{z} \geq 0, \vec{z}^T B \vec{z} \geq 0$$

□ 从而， $\forall \theta_1, \theta_2 \geq 0$,

$$\begin{aligned} \vec{z}^T \cdot (\theta_1 A + \theta_2 B) \cdot \vec{z} &= \vec{z}^T \cdot \theta_1 A \cdot \vec{z} + \vec{z}^T \cdot \theta_2 B \cdot \vec{z} \\ &= \theta_1 \vec{z}^T A \vec{z} + \theta_2 \vec{z}^T B \vec{z} \geq 0 \end{aligned}$$

□ 即： $\forall \theta_1, \theta_2 \geq 0, \theta_1 A + \theta_2 B$ 为半正定阵。从而， n 阶半正定阵的集合为凸锥。

标准正交基

```
from scipy.linalg import orth
from numpy.linalg import matrix_rank
a = (np.eye(3), np.diag((1., 2., 3.)), np.arange(9, dtype=np.float).reshape((3,3)))
for m in a:
    print m, u'的秩为: ', matrix_rank(m)
    print u'正交基为: \n', orth(m)
```

C:\Python27\python.exe D:/Python/MachineLearning/temp/temp.py

```
[[ 1.  0.  0.]
 [ 0.  1.  0.]
 [ 0.  0.  1.]] 的秩为: 3
```

正交基为:

```
[[ 1.  0.  0.]
 [ 0.  1.  0.]
 [ 0.  0.  1.]]
```

```
-----
[[ 1.  0.  0.]
 [ 0.  2.  0.]
 [ 0.  0.  3.]] 的秩为: 3
```

正交基为:

```
[[ 0.  0.  1.]
 [ 0.  1.  0.]
 [ 1.  0.  0.]]
```

```
-----
[[ 0.  1.  2.]
 [ 3.  4.  5.]
 [ 6.  7.  8.]] 的秩为: 2
```

正交基为:

```
[[ -0.13511895  0.90281571]
 [ -0.49633514  0.29493179]
 [ -0.85755134 -0.31295213]]
```

QR分解

□ 对于 $m \times n$ 的列满秩矩阵 A ，必有：

$$A_{m \times n} = Q_{m \times n} \cdot R_{n \times n}$$

□ 其中， $Q^T \cdot Q = I$ (即列正交矩阵)， R 为非奇异上三角矩阵。当要求 R 的对角线元素为正时，该分解唯一。

□ 该分解为 **QR分解**。可用于求解矩阵 A 的特征值、 A 的逆等问题。

QR分解计算特征值

□ 计算n阶方阵A的特征值：

$$A = Q \cdot R \Rightarrow A_1 = Q^T A Q = R \cdot Q$$

...

$$A_k = Q_k \cdot R_k \Rightarrow A_{k+1} = R_k \cdot Q_k$$

...

$$A_K \rightarrow \text{diag}\{\lambda_1, \lambda_2, \dots, \lambda_n\}$$

Code

```
def is_same(a, b):
    n = len(a)
    for i in range(n):
        if math.fabs(a[i]-b[i]) > 1e-6:
            return False
    return True

if __name__ == "__main__":
    a = np.array([0.65, 0.28, 0.07, 0.15, 0.67, 0.18,
                  0.12, 0.36, 0.52])
    n = math.sqrt(len(a))
    a = a.reshape((n, n))
    value, v=np.linalg.eig(a)

    times = 0
    while (times == 0) or (not is_same(np.diag(a), v)):
        v = np.diag(a)
        q, r = np.linalg.qr(a)
        a = np.dot(r, q)
        times += 1
        print "正交阵: \n", q
        print "三角阵: \n", r
        print "近似阵: \n", a
    print "次数: ", times, "近似值: ", np.diag(a)
    print "精确特征值: ", value
```

正交阵:

```
[[ -9.99999997e-01  7.96775552e-05 -1.54260931e-08]
 [ -7.96775484e-05 -9.99999962e-01 -2.62680752e-04]
 [ -3.63558527e-08 -2.62680750e-04  9.99999965e-01]]
```

三角阵:

```
[[ -9.99995561e-01 -2.70043706e-02  2.26002781e-01]
 [  0.00000000e+00 -5.18490813e-01 -2.01260695e-04]
 [  0.00000000e+00  0.00000000e+00  3.21511463e-01]]
```

近似阵:

```
[[  9.99997701e-01  2.68653258e-02  2.26009882e-01]
 [  4.13120842e-05  5.18490847e-01 -6.50631315e-05]
 [ -1.16888234e-08 -8.44548721e-05  3.21511452e-01]]
```

正交阵:

```
[[ -9.99999999e-01  4.13121805e-05  4.95968741e-09]
 [ -4.13121791e-05 -9.99999986e-01  1.62885687e-04]
 [  1.16888502e-08  1.62885687e-04  9.99999987e-01]]
```

三角阵:

```
[[ -9.99997702e-01 -2.68867458e-02 -2.26009875e-01]
 [  0.00000000e+00 -5.18489743e-01  1.26769705e-04]
 [ -0.00000000e+00 -0.00000000e+00  3.21511438e-01]]
```

近似阵:

```
[[  9.99998809e-01  2.68086196e-02 -2.26014256e-01]
 [  2.14199426e-05  5.18489757e-01  4.23151455e-05]
 [  3.75809905e-09  5.23696113e-05  3.21511434e-01]]
```

正交阵:

```
[[ -1.00000000e+00  2.14199684e-05 -1.59459985e-09]
 [ -2.14199681e-05 -9.99999995e-01 -1.01004056e-04]
 [ -3.75810352e-09 -1.01004056e-04  9.99999995e-01]]
```

三角阵:

```
[[ -9.99998810e-01 -2.68197256e-02  2.26014254e-01]
 [  0.00000000e+00 -5.18489185e-01 -7.96303223e-05]
 [  0.00000000e+00  0.00000000e+00  3.21511428e-01]]
```

近似阵:

```
[[  9.99999383e-01  2.67754771e-02  2.26016964e-01]
 [  1.11060221e-05  5.18489190e-01 -2.72608113e-05]
 [ -1.20827323e-09 -3.24739582e-05  3.21511427e-01]]
```

次数: 17 近似值: [0.99999938 0.51848919 0.32151143]

精确特征值: [1. 0.51848858 0.32151142]

LFM (Latent Factor Model)

□ 对于K个隐变量，得： $A_{m \times n} = U_{m \times k} \cdot V_{n \times k}^T$

□ 目标函数：

$$J(U, V; A) = \sum_{i=1}^m \sum_{j=1}^n \left(a_{ij} - \sum_{r=1}^k u_{ir} \cdot v_{jr} \right)^2 + \lambda \left(\sum_{i=1}^m \sum_{r=1}^k u_{ir}^2 + \sum_{j=1}^n \sum_{r=1}^k v_{jr}^2 \right)$$

□ 梯度：

$$\begin{cases} \frac{\partial J(U, V; A)}{\partial u_{ir}} = -2 \cdot \left(a_{ij} - \sum_{r=1}^k u_{ir} \cdot v_{jr} \right) \cdot v_{jr} + 2\lambda u_{ir} \\ \frac{\partial J(U, V; A)}{\partial v_{jr}} = -2 \cdot \left(a_{ij} - \sum_{r=1}^k u_{ir} \cdot v_{jr} \right) \cdot u_{ir} + 2\lambda v_{jr} \end{cases}, 1 \leq r \leq k$$

Code

```
def inverse(a):  
    b = np.zeros_like(a)  
    n = len(a)  
    c = np.eye(n)  
    alpha = 1  
    for times in range(200):  
        for i in range(n):  
            for j in range(n):  
                err = c[i][j] -  
                for k in range(n):  
                    b[j][k] += a  
    return b.T
```

```
136 :  
b:  
[[ 1.70125975 -0.72225553  0.02099578]  
 [-0.33833233  1.97720453 -0.63887219]  
 [-0.15836833 -1.20215954  2.36052786]]  
a*b:  
[[ 9.99999999e-01  3.74451374e-09 -4.12243131e-09]  
 [ 1.66690815e-09  9.99999988e-01  1.32158535e-08]  
 [-1.05973856e-09  7.63175290e-09  9.99999992e-01]]  
137 :  
b:  
[[ 1.70125975 -0.72225554  0.02099579]  
 [-0.33833233  1.97720453 -0.6388722 ]  
 [-0.15836833 -1.20215954  2.36052786]]  
a*b:  
[[ 1.00000000e+00  3.29110765e-09 -3.62326488e-09]  
 [ 1.46506981e-09  9.99999989e-01  1.16156059e-08]  
 [-9.31419677e-10  6.70765854e-09  9.99999993e-01]]  
138 :  
b:  
[[ 1.70125975 -0.72225554  0.02099579]  
 [-0.33833233  1.97720454 -0.6388722 ]  
 [-0.15836833 -1.20215954  2.36052787]]  
a*b:  
[[ 1.00000000e+00  2.89260232e-09 -3.18454021e-09]  
 [ 1.28767115e-09  9.99999991e-01  1.02091250e-08]  
 [-8.18638310e-10  5.89545857e-09  9.99999994e-01]]  
139 :  
b:  
[[ 1.70125975 -0.72225554  0.02099579]  
 [-0.33833233  1.97720454 -0.6388722 ]  
 [-0.15836833 -1.20215955  2.36052787]]  
a*b:  
[[ 1.00000000e+00  2.54235025e-09 -2.79893872e-09]  
 [ 1.13175292e-09  9.99999992e-01  8.97294861e-09]  
 [-7.19513116e-10  5.18160437e-09  9.99999994e-01]]  
[[ 1.70125975 -0.72225554  0.02099579]  
 [-0.33833233  1.97720454 -0.6388722 ]  
 [-0.15836833 -1.20215955  2.36052787]]  
真实值:  
[[ 1.70125975 -0.72225555  0.0209958 ]  
 [-0.33833233  1.97720456 -0.63887223]  
 [-0.15836833 -1.20215957  2.36052789]]
```

向量的导数

- A 为 $m \times n$ 的矩阵， \mathbf{x} 为 $n \times 1$ 的列向量，则 $A\mathbf{x}$ 为 $m \times 1$ 的列向量，记 $\vec{y} = A \cdot \vec{x}$
- 思考： $\frac{\partial \vec{y}}{\partial \vec{x}} = ?$

推导

□ 令

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \quad \vec{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \quad A \cdot \vec{x} = \begin{pmatrix} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \end{pmatrix}$$

□ 从而，

$$\frac{\partial \vec{y}}{\partial \vec{x}} = \frac{\partial A\vec{x}}{\partial \vec{x}} = \begin{bmatrix} a_{11} & a_{21} & \cdots & a_{m1} \\ a_{12} & a_{22} & \cdots & a_{m2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1n} & a_{2n} & \cdots & a_{mn} \end{bmatrix} = A^T$$

结论与直接推广

□ 向量偏导公式: $\frac{\partial A\vec{x}}{\partial \vec{x}} = A^T$

$$\frac{\partial A\vec{x}}{\partial \vec{x}^T} = A$$

$$\frac{\partial (\vec{x}^T A)}{\partial \vec{x}} = A$$

□ 在线性回归中将直接使用该公式。

注意

- 关于列向量求导，有文献给出如下方案：
- 记 \mathbf{x} 为 $n \times 1$ 的列向量， \mathbf{y} 为 $m \times 1$ 的列向量，则：

$$\frac{\partial \vec{y}}{\partial \vec{x}} = \begin{pmatrix} \frac{\partial y_1}{\partial \vec{x}} \\ \vdots \\ \frac{\partial y_m}{\partial \vec{x}} \end{pmatrix} \quad \frac{\partial y_1}{\partial \vec{x}} = \begin{pmatrix} \frac{\partial y_1}{\partial x_1} \\ \vdots \\ \frac{\partial y_1}{\partial x_n} \end{pmatrix}$$

- 以上公式将会导致向量间求导得到“超越矩阵”——矩阵的每个元素仍然是一个矩阵。在实践层面，不利于公式推导，故本课程未采纳。

标量对向量的导数

□ A 为 $n \times n$ 的矩阵， \mathbf{x} 为 $n \times 1$ 的列向量，

□ 记 $y = \vec{x}^T \cdot A \cdot \vec{x}$

□ 同理可得：
$$\frac{\partial y}{\partial \vec{x}} = \frac{\partial (\vec{x}^T \cdot A \cdot \vec{x})}{\partial \vec{x}} = (A^T + A) \cdot \vec{x}$$

□ 若 A 为对称阵，则有
$$\frac{\partial (\vec{x}^T A \vec{x})}{\partial \vec{x}} = 2A\vec{x}$$

推导

$$\frac{\partial y}{\partial \vec{x}} = \frac{\partial (\vec{x}^T \cdot A \cdot \vec{x})}{\partial \vec{x}} = (A^T + A) \cdot \vec{x}$$

□ 记：

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \quad \vec{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

□ 有：

$$\vec{x}^T A \cdot \vec{x} = (x_1 \quad x_2 \quad \cdots \quad x_n) \cdot \left(\sum_{j=1}^n a_{1j} x_j \quad \sum_{j=1}^n a_{2j} x_j \quad \cdots \quad \sum_{j=1}^n a_{n,j} x_j \right)^T$$

□ 则：

$$= \sum_{i=1}^n \left(\left(\sum_{j=1}^n a_{ij} x_j \right) x_i \right) = \sum_{i=1}^n \sum_{j=1}^n a_{ij} x_i x_j$$

$$\frac{\partial (\vec{x}^T A \cdot \vec{x})}{\partial x_i} = \left(\sum_{j=1}^n a_{ij} x_j \right) + \left(\sum_{j=1}^n a_{ji} x_j \right) = \sum_{j=1}^n (a_{ij} + a_{ji}) x_j$$

标量对方阵的导数

□ A 为 $n \times n$ 的矩阵， $|A|$ 为 A 的行列式，计算 $\frac{\partial |A|}{\partial A}$

□ 解：根据等式 $\forall 1 \leq i \leq n, |A| = \sum_{j=1}^n a_{ij} \cdot (-1)^{i+j} M_{ij}$

□ 有 $\frac{\partial |A|}{\partial a_{ij}} = \frac{\partial \left(\sum_{j=1}^n a_{ij} \cdot (-1)^{i+j} M_{ij} \right)}{\partial a_{ij}} = (-1)^{i+j} M_{ij} = A_{ji}^*$

□ 从而： $\frac{\partial |A|}{\partial A} = (A^*)^T = |A| \cdot (A^{-1})^T$

■ 依据 $A \cdot A^* = |A| \cdot I$ ，第二个等式成立；

总结与思考

- 线性代数是普适的数学工具，是进一步学习其他内容的基础。
 - 有些机器学习的推导过程使用该工具表述清晰，易于推广，如线性回归。
 - 重点思考特征值、特征向量和矩阵的关系。
- 思考：对于“非线性”问题，线性代数这一工具是否足够？是否有“非线性代数”？
 - 非线性映射、核函数
- 查阅：
 - Schmidt正交化/Givens变换/Householder变换
 - Hessenberg矩阵

参考文献

- 同济大学数学系 编，工程数学线性代数(第五版)，高等教育出版社，2007

我们在这里

□ <http://wenda.ChinaHadoop.cn>

■ 视频/课程/社区

□ 微博

■ @ChinaHadoop

■ @邹博_机器学习

□ 微信公众号

■ 小象学院

■ 大数据分析挖掘

互联网新技术在线教育领航者

小象问答 搜索标题、用户 全站内容搜索 提问 首页 动态 发现 话题 通知

全部 招聘求职 机器学习 大数据平台技术 DCon 大数据行业应用 NoSQL数据库 数据科学 江湖救急

发现 最新 推荐 热门 等待回复

graphviz has no attribute 'write' 贡献
邹博 回复了问题 • 2 人关注 • 1 个回复 • 3 次浏览 • 2017-04-09 15:48

sklearn中如何理解Pipeline机制 贡献
数据分析与数据挖掘 邹博 回复了问题 • 2 人关注 • 1 个回复 • 28 次浏览 • 2017-04-09 15:39

关于9.Logistic回归的ppt中第9页的对数线性函数 贡献
机器学习 邹博 回复了问题 • 3 人关注 • 3 个回复 • 39 次浏览 • 2017-04-09 15:35

关于“贝叶斯估计中，最大后验概率估计就是结构化风险最小化的例子：当模型是条件概率分布，损失函数为对数损失函数，模型的复杂度由模型的先验概率表示，结构化风险最小化就等价于最大后验概率估计” 贡献
机器学习 邹博 回复了问题 • 2 人关注 • 1 个回复 • 26 次浏览 • 2017-04-09 15:27

关于连续值的预测 贡献
咨询 邹博 回复了问题 • 2 人关注 • 1 个回复 • 31 次浏览 • 2017-04-09 15:24

拉格朗日对偶函数为什么一定是凸函数 贡献
数据科学 邹博 回复了问题 • 2 人关注 • 2 个回复 • 26 次浏览 • 2017-04-09 15:20

梯度下降公式中的斯堪J 是 贡献
机器学习 邹博 回复了问题 • 2 人关注 • 1 个回复 • 29 次浏览 • 2017-04-09 15:17

深度学习适合做预测吗？ 贡献
深度学习 邹博 回复了问题 • 2 人关注 • 1 个回复 • 27 次浏览 • 2017-04-09 15:15

关于6.4PCA_FeatureSelection.py中plt.legend的参数疑问 贡献
机器学习 邹博 回复了问题 • 2 人关注 • 1 个回复 • 28 次浏览 • 2017-04-09 15:04

@邹博 有哪些可以下载数据源的网站？ 贡献
数据分析与数据挖掘 邹博 回复了问题 • 4 人关注 • 1 个回复 • 31 次浏览 • 2017-04-09 14:53

LDA主题模型 贡献
机器学习 邹博 回复了问题 • 2 人关注 • 1 个回复 • 29 次浏览 • 2017-04-09 14:45

代码10.6bagging_ridged老师提到了采样率设为0.2能够使峰值部分的数据被体现出来。这是为什么呢？ 贡献
机器学习 邹博 回复了问题 • 2 人关注 • 1 个回复 • 22 次浏览 • 2017-04-09 14:26

GraphViz's executables not found 贡献
机器学习 邹博 回复了问题 • 3 人关注 • 2 个回复 • 23 次浏览 • 2017-04-09 13:47

决策树中关于feature_importances代码的问题 贡献
机器学习 邹博 回复了问题 • 2 人关注 • 1 个回复 • 6 次浏览 • 2017-04-09 13:11

专题
招聘求职
大数据行业应用
数据科学
系统与编程
云计算技术

热门话题 更多 >
机器学习 907 个问题, 230 人关注
spark 387 个问题, 172 人关注
hadoop 1059 个问题, 155 人关注
python数据分析 171 个问题, 28 人关注
数据分析与数据挖掘 54 个问题, 111 人关注

热门用户 更多 >
小心巴 14 个问题, 0 次赞同
叉叉V 45 个问题, 22 次赞同
铁甲无声 10 个问题, 0 次赞同
带刀锦衣卫 13 个问题, 0 次赞同

感谢大家！

恳请大家批评指正！

联系我们

小象学院：互联网新技术在线教育领航者

- 微信公众号：小象学院
- 新浪微博：小象AI学院

