**《深度学习》花书学习**

# 线性代数

## 基本概念

标量（scalar）：一个标量就是一个单独的数，用斜体表示标量。标量通常被赋予小写的变量名称。

向量（vector）：一个向量是一列数。这些数是有序排列的。通过次序中的索引，我们可以确定每个单独的数。通常我们赋予向量粗体的小写变量名称，比如。向量中的元素可以通过带脚标的斜体表示。



矩阵（matrix）：矩阵是一个二维数组，其中的每一个元素被两个索引（而非一个）所确定。我们通常会赋予矩阵粗体的大写变量名称，比如。



张量（tensor）：在某些情况下，我们会讨论坐标超过两维的数组。一般地，一个数组中的元素分布在若干维坐标的规则网格中，我们称之为张量。我们使用字体来表示张量“A’’。张量中坐标为的元素记作。

转置（ transpose）是矩阵的重要操作之一。矩阵的转置是以对角线为轴的镜像，这条从左上角到右下角的对角线被称为主对角线（main diagonal）。我们将矩阵的转置表示为，定义如下



广播（broadcasting）：在深度学习中，我们也使用一些不那么常规的符号。我们允许矩阵和向量相加，产生另一个矩阵：，其中。换言之，向量和矩阵的每一行相加。这个简写方法使我们无需在加法操作前定义一个将向量复制到每一行而生成的矩阵。这种隐式地复制向量到很多位置的方式，被称为广播。

## 矩阵和向量相乘

两个矩阵和的矩阵乘积（matrix product）是第三个矩阵。为了使乘法定义良好，矩阵的列数必须和矩阵的行数相等。如果矩阵的形状是，矩阵的形状是，那么矩阵的形状是。



具体地，该乘法操作定义为



两个矩阵的标准乘积不是指两个矩阵中对应元素的乘积。不过，那样的矩阵操作确实是存在的，被称为**元素对应乘积**（element-wise product）或者**Hadamard乘积**（Hadamard product），记为。

矩阵乘积运算的性质：

分配律： 

结合律： 



 （一般不满足交换律）

 （向量点积为标量，转置相等）

## 单位矩阵和逆矩阵

单位矩阵（identity matrix）：任意向量和单位矩阵相乘，都不会改变。我们将保持维向量不变的单位矩阵记作。



（I3）

逆矩阵（matrix inversion）：记为，满足以下条件



对于求解，可以有以下推导过程







## 线性相关和生成子空间

如果逆矩阵存在，那么式肯定对于每一个向量恰好存在一个解。但是，对于方程组而言，对于向量的某些值，有可能不存在解，或者存在无限多个解。存在多于一个解但是少于无限多个解的情况是不可能发生的；因为如果和都是某方程组的解，则



（其中取任意实数）也是该方程组的解。

为了分析方程有多少个解，我们可以将的列向量看作从原点（origin）（元素都是零的向量）出发的不同方向，确定有多少种方法可以到达向量。在这个观点下，向量中的每个元素表示我们应该沿着这些方向走多远，即表示我们需要沿着第个向量的方向走多远：



一般而言，这种操作被称为**线性组合**（linear combination）。形式上，一组向量的线性组合，是指每个向量乘以对应标量系数之后的和，即：



一组向量的**生成子空间**（span）是原始向量线性组合后所能抵达的点的集合。

确定是否有解相当于确定向是否在列向量的生成子空间中。这个特殊的生成子空间被称为的**列空间**（column space）或者的**值域**（range）。

为了使方程对于任意向量都存在解，我们要求的列空间构成整个。如果中的某个点不在的列空间中，那么该点对应的会使得该方程没有解。矩阵的列空间是整个的要求，意味着至少有列，即。否则，列空间的维数会小于。例如，假设是一个3×2的矩阵。目标是3维的，但是只有2维。所以无论如何修改的值，也只能描绘出空间中的二维平面。当且仅当向量在该二维平面中时，该方程有解。

不等式仅是方程对每一点都有解的必要条件。这不是一个充分条件，因为有些列向量可能是冗余的。假设有一个中的矩阵，它的两个列向量是相同的。那么它的列空间和它的一个列向量作为矩阵的列空间是一样的。换言之，虽然该矩阵有2列，但是它的列空间仍然只是一条线，不能涵盖整个空间。

正式地说，这种冗余被称为**线性相关**（linear dependence）。如果一组向量中的任意一个向量都不能表示成其他向量的线性组合，那么这组向量称为**线性无关**（linearly independent）。如果某个向量是一组向量中某些向量的线性组合，那么我们将这个向量加入这组向量后不会增加这组向量的生成子空间。这意味着，如果一个矩阵的列空间涵盖整个，那么该矩阵必须包含至少一组个线性无关的向量。这是式对于每一个向量的取值都有解的充分必要条件。值得注意的是，这个条件是说该向量集恰好有个线性无关的列向量，而不是至少个。不存在一个维向量的集合具有多于个彼此线性不相关的列向量，但是一个有多于个列向量的矩阵有可能拥有不止一个大小为的线性无关向量集。

要想使矩阵可逆，我们还需要保证式对于每一个值至多有一个解。为此，我们需要确保该矩阵至多有个列向量。否则，该方程会有不止一个解。

综上所述，这意味着该矩阵必须是一个**方阵**（square），即，并且所有列向量都是线性无关的。一个列向量线性相关的方阵被称为**奇异的**（singular）。

如果矩阵不是一个方阵或者是一个奇异的方阵，该方程仍然可能有解。但是我们不能使用矩阵逆去求解。

## 范数

在机器学习中，我们经常使用被称为**范数**（norm）的函数衡量向量大小。形式上，范数定义如下



其中。

范数是将向量映射到非负值的函数。直观上来说，向量的范数衡量从原点到点的距离。更严格地说，范数是满足下列性质的任意函数：

* 
* 
* 

范数被称为**欧几里得范数**（Euclidean norm）；

范数：

范数，也被称为最大范数（max norm）：

有时候我们可能也希望衡量矩阵的大小。在深度学习中，最常见的做法是使用**Frobenius范数**（Frobenius norm），。

两个向量的**点积**（dot product）可以用范数来表示。具体地，



其中表示和的夹角。

## 特殊类型的矩阵和向量

对角矩阵（diagonal matrix）只在主对角线上含有非零元素，其他位置都是零。我们用diag()表示一个对角元素由向量中元素给定的对角方阵。

对称（symmetric）矩阵是转置和自己相等的矩阵：。

**单位向量**（unit vector）是具有**单位范数**（unit norm）的向量：。

如果=0，那么向量和向量互相正交（orthogonal）。如果这些向量不仅互相正交，并且范数都为1，那么我们称它们是**标准正交**（orthonormal）。

**正交矩阵**（orthogonal matrix）是指行向量和列向量是分别标准正交的方阵：



这意味着



## 特征分解

**特征分解**（eigendecomposition）是使用最广的矩阵分解之一，即我们将矩阵分解成一组特征向量和特征值。

方阵的特征向量（eigenvector）是指与相乘后相当于对**该向量进行缩放**的非零向量：



标量λ被称为这个特征向量对应的**特征值**（eigenvalue）。

如果是的特征向量，那么任何缩放后的向量也是的特征向量。此外，和有相同的特征值。基于这个原因，通常我们只考虑**单位特征向量**。

假设矩阵有n个线性无关的特征向量，对应着特征值。我们将特征向量连接成一个矩阵，使得每一列是一个特征向量：。类似地，我们也可以将特征值连接成一个向量。因此的特征分解（eigendecomposition）可以记作



每个实对称矩阵都可以分解成实特征向量和实特征值：



其中是的特征向量组成的正交矩阵，是对角矩阵。特征值对应的特征向量是矩阵的第列，记作。因为是正交矩阵，我们可以将看作沿方向延展倍的空间。

所有特征值都是正数的矩阵被称为**正定**（positive definite）；所有特征值都是非负数的矩阵被称为**半正定**（positive semidefinite）。同样地，所有特征值都是负数的矩阵被称为**负定**（negative definite）；所有特征值都是非正数的矩阵被称为**半负定**（negative semidefinite）。

**对称矩阵一定可以对角化（特别是对称正定矩阵）。**

## 奇异值分解

上一节我们探讨了如何将矩阵分解成特征向量和特征值。还有另一种分解矩阵的方法，被称为**奇异值分解**（singular value decomposition, SVD），将矩阵分解为**奇异向量**（ singular vector）和**奇异值**（singular value）。通过奇异值分解，我们会得到一些与特征分解相同类型的信息。然而，奇异值分解有更广泛的应用。每个实数矩阵都有一个奇异值分解，但不一定都有特征分解。例如，非方阵的矩阵没有特征分解，这时我们只能使用奇异值分解。

奇异值分解是类似的，只不过这回我们将矩阵A分解成三个矩阵的乘积：



假设是一个的矩阵，那么是一个的矩阵，是一个的矩阵，是一个矩阵。矩阵和都定义为正交矩阵，而矩阵定义为对角矩阵。注意，矩阵不一定是方阵。

对角矩阵对角线上的元素被称为矩阵的**奇异值**（singular value）。矩阵的列向量被称为**左奇异向量**（left singular vector），矩阵的列向量被称**右奇异向量**（right singular vector）。

事实上，我们可以用与相关的特征分解去解释的奇异值分解。的左奇异向量（left singular vector）是的特征向量。的右奇异向量（right singular vector）是的特征向量。**的非零奇异值是特征值的平方根，同时也是特征值的平方根**。

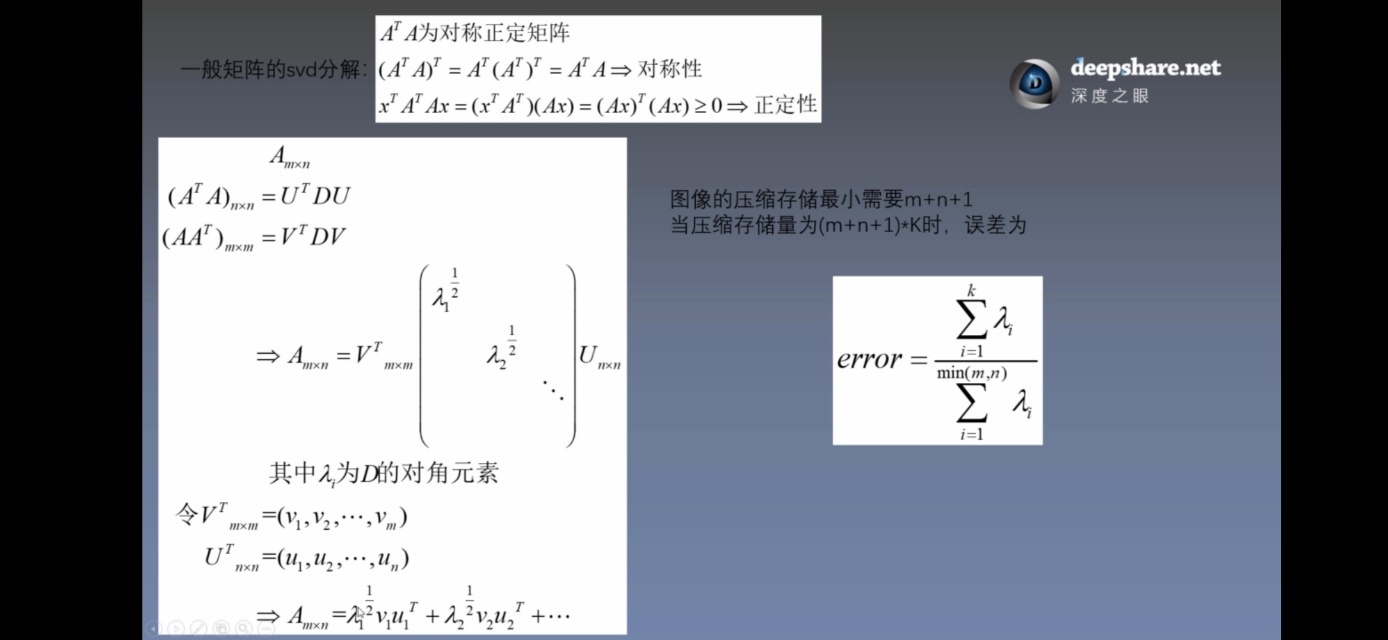


图1 对称正定证明

## Moore-Penrose（广义逆矩阵）伪逆

对于非方矩阵而言，其逆矩阵没有定义。假设在下面的问题中，我们希望通过矩阵的左逆来求解线性方程，



等式两边左乘左逆后，我们得到



如果矩阵的行数大于列数，那么上述方程可能没有解。如果矩阵的行数小于列数，那么上述矩阵可能有多个解。

**Moore-Penrose伪逆**（Moore-Penrose pseudoinverse）使我们在这类问题上取得了一定的进展。矩阵的伪逆定义为：



计算伪逆的实际算法没有基于这个定义，而是使用下面的公式：



其中，矩阵，和是矩阵奇异值分解后得到的矩阵。对角矩阵的伪逆是**其非零元素取倒数之后再转置**得到的。

当矩阵的列数多于行数时，使用伪逆求解线性方程是众多可能解法中的一种。特别地，是方程所有可行解中欧几里得范数最小的一个。

当矩阵的行数多于列数时，可能没有解。在这种情况下，通过伪逆得到的使得和的欧几里得距离最小。

## 迹运算

迹运算返回的是矩阵对角元素的和：



迹运算提供了另一种描述矩阵Frobenius范数的方式：



迹运算在转置运算下是不变的：



多个矩阵相乘得到的方阵的迹，和将这些矩阵中的最后一个挪到最前面之后相乘的迹是相同的。当然，我们需要考虑挪动之后矩阵乘积依然定义良好：



或者更一般地，



即使循环置换后矩阵乘积得到的矩阵形状变了，迹运算的结果依然不变。例如，假设矩阵，矩阵，我们可以得到



尽管和。

另一个有用的事实是标量在迹运算后仍然是它自己：。

## 行列式

行列式，记作det()，是一个将方阵映射到实数的函数。**行列式等于矩阵特征值的乘积**。行列式的绝对值可以用来衡量矩阵参与矩阵乘法后空间扩大或者缩小了多少。如果行列式是0，那么空间至少沿着某一维完全收缩了，使其失去了所有的体积。如果行列式是1，那么这个转换保持空间体积不变。

## 视频主讲内容学习

### 矩阵对角化及SVD分解

矩阵（方阵）的对角化，其中为对角矩阵，为单位正交矩阵。

一般的矩阵不一定能对角化，但是对称矩阵一定可以对角化（特别是对称正定矩阵，得到的都是正数）。



所以 

设 ，

则 

所以一个矩阵（方阵）可以写成n个方阵的求和。

当矩阵不是方阵而是一般矩阵时，如何得出和上面类似的结果呢？

可以证明为对称正定矩阵：



如果为矩阵，则



其中为的对角线元素。

令 

则有



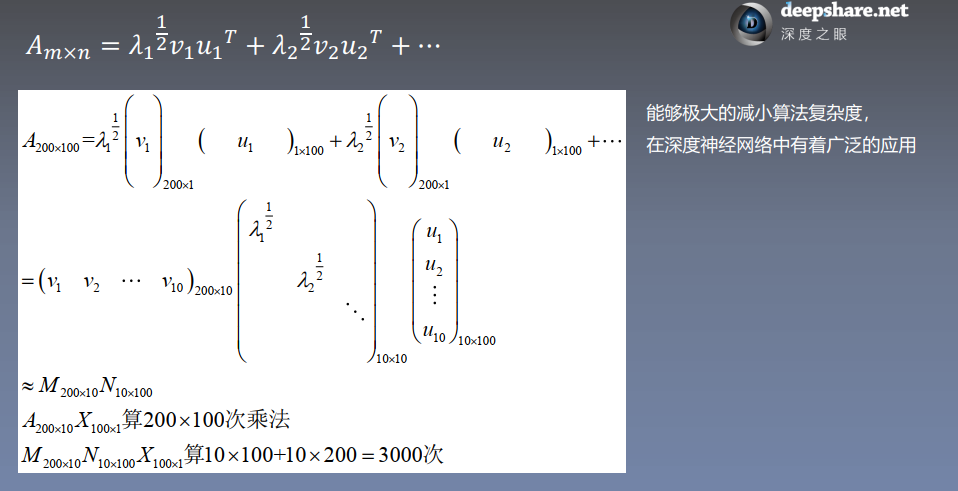


图2 SVD应用举例

### 伪逆矩阵最小二乘



写成矩阵的形式有

，即

当且可逆时，直接采用逆矩阵法求解：

但是一般情况下，因为在实际应用中一般都会要求样本数量远远大于样本维数，所以不会是方阵，也就不存在逆矩阵，此时通过最小化两者的范数：

（最小二乘解）

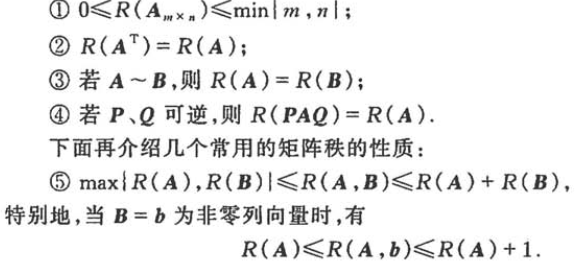
求导得（求导性质）



下面分情况判断是否可逆。

当时，如=5，=3，一般是可逆的，此时。

当时，如=3，=5，为5x5矩阵，（秩的性质⑦），（可逆性质）故不可逆。





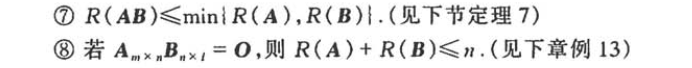
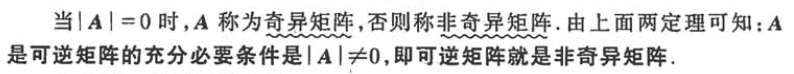


图3 秩的性质



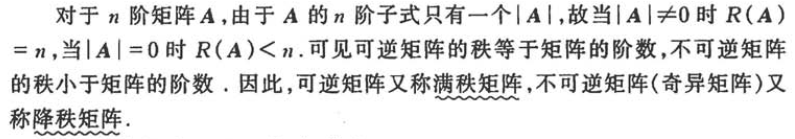


图4 矩阵可逆的充要条件

因此上面的分析仍然无法获取一个全面的估计，做如下变形：

（最小范数解）

对其进行求导



其中为对称矩阵，可以进行对角化。



分析如下：

对于判断可逆，（正定、行列式的值），所以仍可能为0，不一定可逆；

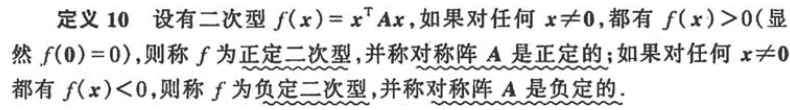




图5 正定矩阵定义

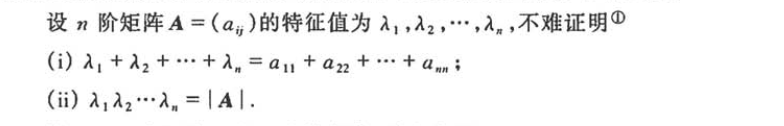


图6 特征值与矩阵的关系

对于判断可逆，，所以恒成立，一定可逆。（在机器学习中，实际上就是正则化项）

### PCA原理及推导

主成分分析是一种数据压缩或降维算法，其基本原理是确定新的方向向量（Vector direction），当投影所有的数据至该向量上时，希望投射平均均方误差尽可能地小（优化目标）。

如图3所示，A点需要x，y两个坐标来表示（二维平面空间），假设A在向量u上面的投影点为A’（垂直于向量u），则A’仅仅需要一个参数就能表示，就是OA’的长度（即A’在u上的坐标，因为此时u相当于一个新的坐标轴，A’在另一个垂直坐标轴上的坐标永远为0），我们就想着用A’来替换A，这样N个点（原来要2\*N个参数，对于M维空间就是M\*N），现在只需要（N+2）个参数（u也需要2个坐标参数，更严格地说只需要方向参数，但无所谓了，因为不管N+1还是N+2，都比原来降维了），但是此时就带来了误差，如AA’和BB’，所以我们要能够找到这样一个方向u，使得所有原始点与投影点之间的误差最小。

一般在PCA之前需要进行样本中心化，即每一维的数据都减去该维的均值。这样变换后每一维的均值都为0，这样相比于原始点，可以缩小与投影点的误差。



图7 投射原理

最小重构误差确定u的方向：

表示的投影向量，为单位向量，根据图4，要计算向量a在e方向的投影向量，首先计算a在e向上投影长度，根据余弦定理有，然后再乘以该方向上的单位向量即可。

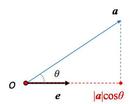


图8 投影计算

因此图6中，有



在n维空间中，x实际上就是n为列向量，点积操作可以写成矩阵相乘的形式（如图5所示），同时为表示简便，后面将省略“”，则有



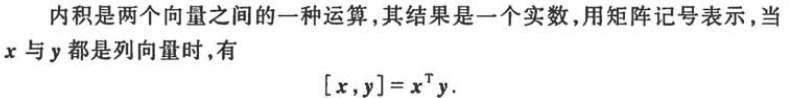
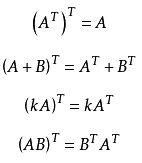


图9 点积定义

图6中的重构误差：

为实数，所以转置为本身，同时可以前提。



图10 误差计算推导

因此最小化重构误差（单样本）就等价于最大化，有



对于N个样本，则有



另，则

通过拉格朗日进行优化：





可以看出为的特征向量，为的特征值。为对称矩阵，

## 自学习内容-矩阵求导

### 标量对矩阵的求导术

首先来琢磨一下定义，标量f对矩阵X的导数，定义为，即f对X逐元素求导排成与X尺寸相同的矩阵。然而，这个定义在计算中并不好用，实用上的原因是对函数较复杂的情形难以逐元素求导；哲理上的原因是逐元素求导破坏了**整体性**。试想，为何要将f看做矩阵X而不是各元素的函数呢？答案是用矩阵运算更整洁。所以在**求导时不宜拆开矩阵**，而是要找一个从整体出发的算法。

为此，我们来回顾，一元微积分中的导数（标量对标量的导数）与微分有联系：；多元微积分中的梯度（标量对向量的导数）也与微分有联系：，这里第一个等号是全微分公式，第二个等号表达了梯度与微分的联系：全微分是梯度向量与微分向量的内积；受此启发，我们将矩阵导数与微分建立联系：。其中tr代表迹(trace)是方阵对角线元素之和，满足性质：对尺寸相同的矩阵A、B，，即是矩阵A,B的内积。与梯度相似，这里第一个等号是全微分公式，第二个等号表达了矩阵导数与微分的联系：全微分是导数(m×n)与微分矩阵(m×n)的内积。

然后来建立运算法则。回想遇到较复杂的一元函数如，我们是如何求导的呢？通常不是从定义开始求极限，而是先建立了初等函数求导和四则运算、复合等法则，再来运用这些法则。故而，我们来创立常用的矩阵微分的运算法则：

1、四则运算

加减法：

矩阵乘法：

转置：

迹：

2、逆：。此式可在两侧求微分来证明。

3、行列式：，其中表示X的伴随矩阵，在X可逆时又可以写作。

4、逐元素乘法：，表示尺寸相同的矩阵X,Y逐元素相乘。

5、逐元素函数：，是逐元素标量函数运算，是逐元素求导数。例如



我们试图利用矩阵导数与微分的联系，在求出左侧的微分后，该如何写成右侧的形式并得到导数呢？这需要一些迹技巧(trace trick)：

1、标量套上迹：

2、转置：。

3、线性：。

4、矩阵乘法交换：，其中与尺寸相同。两侧都等于。

5、矩阵乘法/逐元素乘法交换：，其中、、尺寸相同。两侧都等于。

观察一下可以断言，若标量函数f是矩阵X经加减乘法、逆、行列式、逐元素函数等运算构成，则使用相应的运算法则对f求微分，再使用迹技巧给df套上迹并将其它项交换至dX左侧，对照导数与微分的联系，即能得到导数。

特别地，若矩阵退化为向量，对照导数与微分的联系，即能得到导数。

在建立法则的最后，来谈一谈复合：假设已求得，而Y是X的函数，如何求呢？在微积分中有标量求导的链式法则，但这里我们不能随意沿用标量的链式法则，因为矩阵对矩阵的导数截至目前仍是未定义的。于是我们继续追本溯源，链式法则是从何而来？源头仍然是微分。我们直接从微分入手建立复合法则：先写出，再将dY用dX表示出来代入，并使用迹技巧将其他项交换至dX左侧，即可得到。

最常见的情形是，此时

，可得到。注意这里，由于、是常量，，以及我们使用矩阵乘法交换的迹技巧交换了与。

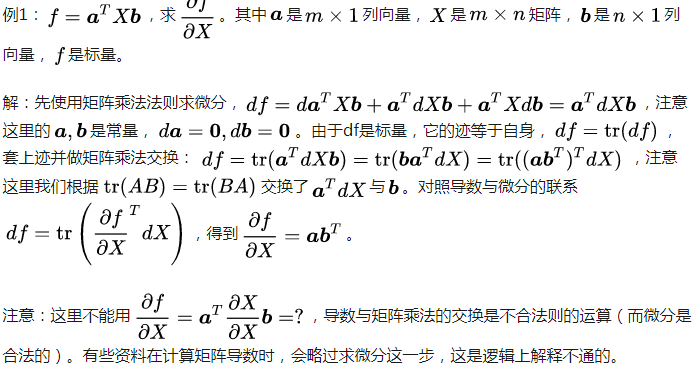


图11 例1

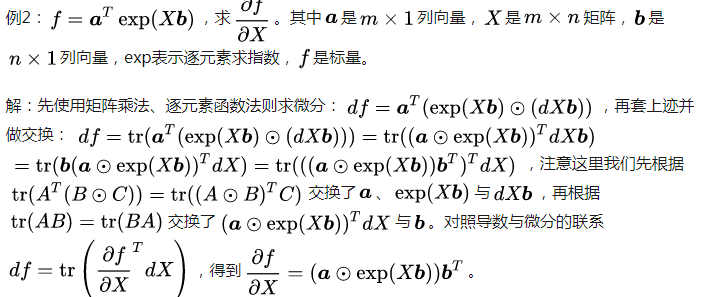


图12 例2

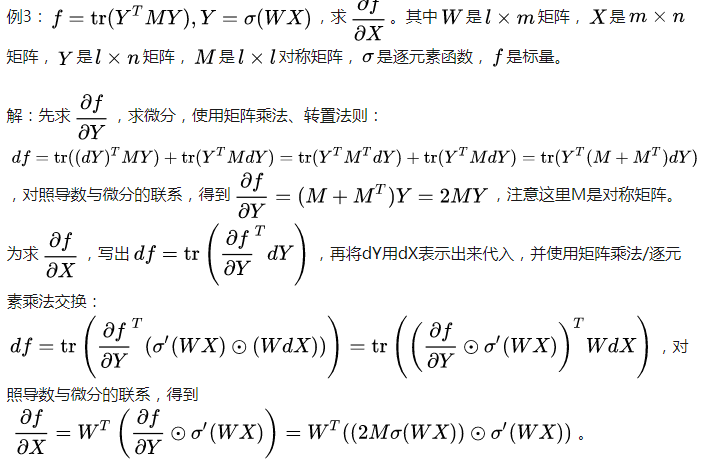


图13 例3

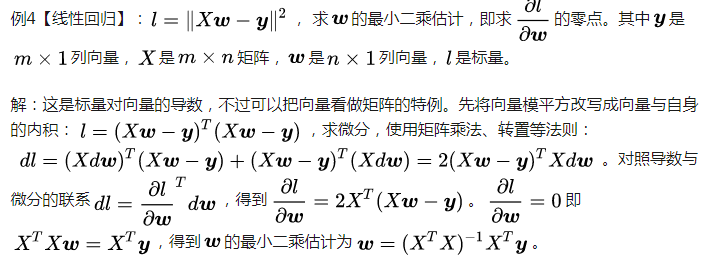


图14 例4

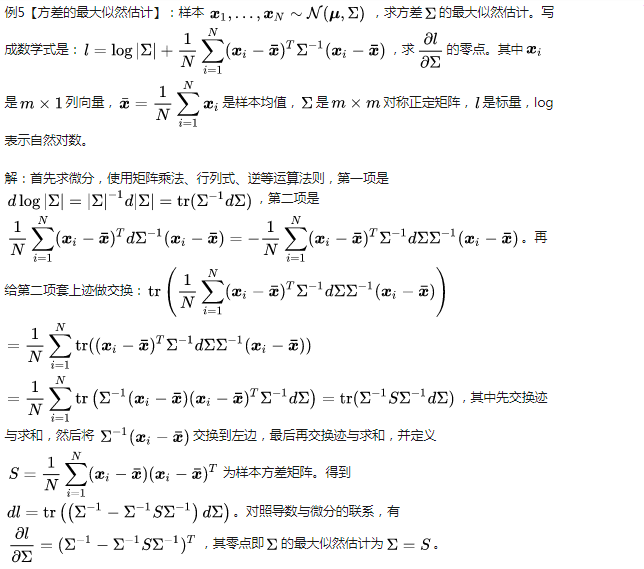


图15 例5

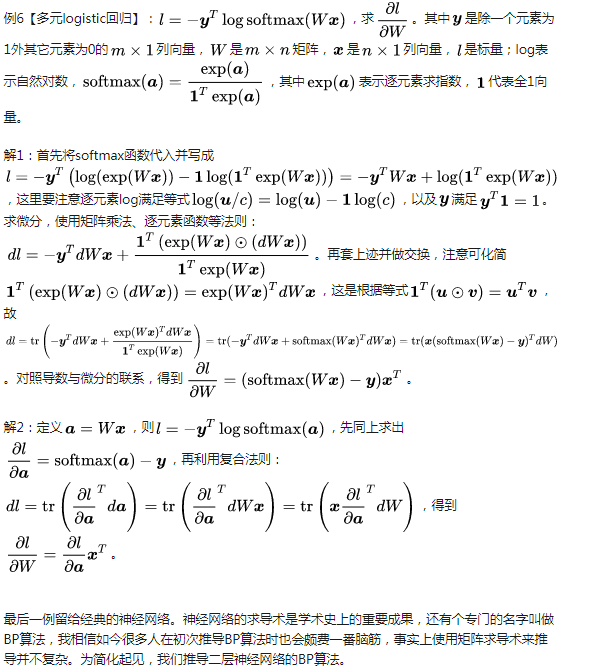


图16 例6

