> zhuanlan.zhihu.com

矩阵的奇异值分解（singular value decomposition，简称 SVD）是线性代数中很重要的内容，我们可以很轻松地对矩阵进行奇异值分解。事实上，高阶张量也可以进行奇异值分解，并且习惯上称高阶张量的奇异值分解为高阶奇异值分解（higher-order singular value decomposition，简称 HOSVD），理解矩阵的奇异值分解有助于认识高阶奇异值分解。

然而，奇异值分解作为矩阵分解中最基本的方法，通常都是作用于完整的矩阵（即矩阵中不存在元素缺失的现象），即要求矩阵必须是稠密矩阵，但在实际应用中，我们能够得到的很多矩阵都是稀疏的或者说是部分位置上的元素出现缺失，那么，如何对稀疏矩阵进行奇异值分解呢？因此，接下来将讨论这个有趣的话题。

1 矩阵的奇异值分解

假设任意一个大小为$m \times n$的矩阵A（不存在元素缺失的现象），矩阵A都可以分解为如下三组件（triplet）的形式：

$$A = Q\Sigma P^T$$

其中，大小为$m \times m$的矩阵Q列对应着矩阵AA^T的m个特征向量；大小为$n \times n$的矩阵P列对应着矩阵$A^T A$的n个特征向量；在大小为$m \times n$的矩阵Σ上，对角线上的元素对应着矩阵AA^T（或等价的$A^T A$）的非负特征值的平方根，通常，矩阵Σ的对角线上的元素被称为奇异值（都是非负的）。

但遗憾的是，这个分解过程实际上并不常用，为什么呢?

从上述定义可以看出，奇异值分解得到的三组件并没有体现出降维过程（dimensionality reduction），如果给定一个大小为20000×10000的矩阵（已经很大了），我们会发现通过奇异值分解得到三个矩阵Q，Σ，P大小依次是：20000×20000，20000×10000，10000×10000，此时，分解出来的三组件更大了，并且加重了计算机的存储负担，我们不禁会问：奇异值分解的降维处理体现在哪里呢?

事实上，奇异值分解的降维处理主要体现在其低秩逼近问题（low-rank approximation problem）上，在这里，奇异值分解的低秩逼近也被称为截断奇异值分解（truncated SVD），只选取前$k \leq \min(m, n)$个最大的奇异值和其对应的特征向量，低秩逼近问题将奇异值分解表达为

$$A \approx Q_k \Sigma_k P_k^T$$

其中，矩阵Q_k，Σ_k和P_k的大小分别为$m \times k$，$k \times k$和$n \times k$，矩阵Q_k和Σ_k分别由矩阵AA^T和$A^T A$前k个最大的特征向量组成，同时，矩阵Σ_k对角线上的元素是前k个最大特征值的平方根（即前k个最大的奇异值）。

这样，假设取

$$k = 100$$

，大小为

$$20000 \times 10000$$

的矩阵可以分解为大小依次为

20000×100，100×100和10000×100

的三组件，这样便可大大节约了计算机的存储开销。其中，MATLAB 提供了奇异值分解和其低秩逼近的函数，分别是 svd 和 svds，有兴趣的读者可以准备一个矩阵去体验一下奇异值分解。

2 稀疏矩阵的奇异值分解

上面讨论了奇异值分解的定义，但稀疏矩阵的奇异值分解过程如何进行呢？当然，对稀疏矩阵进行奇异值分解有很多方法，Charu C. Aggarwal 在其著作《Recommender systems》第 115 页提到了一种简单的迭代式分解方法，接下来将对这一有趣的方法进行详述，而其他方法将不做介绍。

给定一个大小为5×4的稀疏矩阵$A = \begin{bmatrix} ? & 99 & 449 & 517 \\ ? & ? & 412 & ? \\ 192 & ? & 697 & 687 \\ 185 & ? & 699 & 657 \\ 164 & 58 & ? & ? \end{bmatrix}$，“？”表示该位置上出现了元素的缺失，其中，真实的完整矩阵为$A_{real} = \begin{bmatrix} 208 & 99 & 449 & 517 \\ 104 & 43 & 412 & 411 \\ 192 & 77 & 697 & 687 \\ 185 & 115 & 699 & 657 \\ 164 & 58 & 696 & 599 \end{bmatrix}$，需要说明的是，矩阵的 5 行分别对应着 5 个道路断面检测器，4 列分别对应着 4 个时间窗（time window，例如 15min），矩阵中的元素表示交通流量（辆 / 15min），如何对矩阵A进行奇异值分解呢？分解出来的效果与A_{real}接近吗？能否采用迭代式的奇异值分解对缺失元素进行填补同时提取特征呢？

同时，值得一提的是，随着感知技术的发展，智能交通系统能够通过大量固定感知器（如地感线圈、高清卡口等）和移动感知器（如浮动车）采集交通数据，但在数据采集过程中，往往会发生由于检测器故障或通讯中断而导致的数据缺失，因此，上述“？”位置上缺失数据的填补也就成了相应的核心问题。另外，交通流量与时间和空间高度相关，利用稀疏矩阵的奇异值分解可以用于提取交通特征。

为了便于理解，稀疏矩阵的奇异值分解将分为缺失元素的初始化填补和迭代的奇异值分解两部分进行讲解。

3 缺失元素的初始化填补

令所有观测到的位置索引记作集合S，我们知道，如果将矩阵A中缺失的元素视为“0”，虽然也可以直接对矩阵$\begin{bmatrix} 0 & 99 & 449 & 517 \\ 0 & 0 & 412 & 0 \\ 192 & 0 & 697 & 687 \\ 185 & 0 & 699 & 657 \\ 164 & 58 & 0 & 0 \end{bmatrix}$进行奇异值分解，但各行或者各列缺失程度不同会导致分解出来的结果并不是我们所期望的，当然这只是一个方面。

因此，这就往往需要先对缺失元素做一个“粗糙的”填补，最简单地，我们可以将缺失元素用所在行或者列的均值进行填补，即所有元素缺失的位置都用均值（即$\mu = 401$）填补，填补后的 RMSE（the root mean squared error，计算公式为：$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x}_i)^2}$，$x_i$为真实值，$\hat{x}_i$为估计值。）为 **266.16**.

但是，更为精确的做法是采用如下形式：

$$\hat{a}_{ij} = \mu + b_i^{(1)} + b_j^{(2)}, i = 1, \ldots, 5, j = 1, \ldots, 4$$

其中，$\hat{a}_{ij}$是矩阵A在(i,j)位置上的估计值，$\mu=\frac{1}{|S|}\sum_{(i,j)\in S}a_{ij}$为所有观测到数据的均值，$b_i^{(1)}$表示第$i$行相对于均值$\mu$产生的偏置（bias），$b_j^{(2)}$表示第$j$列相对于均值$\mu$产生的偏置，两个偏置共同作用于均值$\mu$，可用于校正位置$(i,j)$上的值。

构造相应的优化问题

$$\min L_1=\frac{1}{2}\sum_{(i,j)\in S}(a_{ij}-\hat{a}_{ij})^2$$

，若将偏置项$b_i^{(1)}$和$b_j^{(2)}$

视为无约束优化的优化问题的决策变量，则

$$\frac{\partial L_1}{b_i^{(1)}}=-\sum_{j:(i,j)\in S}e_{ij},i=1,\ldots,5,\ \frac{\partial L_1}{b_j^{(2)}}=-\sum_{i:(i,j)\in S}e_{ij},j=1,\ldots,4$$

，

其中，$e_{ij}=a_{ij}-\hat{a}_{ij}$，另外，更新公式中的求和项下标$j:(i,j)\in S$和$i:(i,j)\in S$分别表示向量$A(i,:)$和$A(:,j)$上所有非零元素的位置索引构成的集合。采用梯度下降，偏置项$b_i^{(1)}$和$b_j^{(2)}$的更新公式分别为

$$b_i^{(1)}=b_i^{(1)}+\alpha_1\sum_{j:(i,j)\in S}e_{ij},i=1,\ldots,5,$$

$$b_j^{(2)}=b_j^{(2)}+\alpha_1\sum_{i:(i,j)\in S}e_{ij},j=1,\ldots,4.$$

根据$b_i^{(1)}$和$b_j^{(2)}$的更新公式，可以用 MATLAB 很轻松地编写出程序，具体如下。

```
A_sparse=[0,90,449,517;0,0,412,0;192,0,697,687;185,0,699,657;164,58,0,0];
A_real=[208,90,449,517;104,43,412,411;192,77,697,687;185,115,699,657;...
    164,58,696,599];
pos1=find(A_sparse==0);pos2=find(A_sparse~=0);[m,n]=size(A_real);
mu=mean(A_sparse(pos2));b1=0.1*rand(1,m);b2=0.1*rand(1,n);error=zeros(m,n);
alpha1=0.01;
for iter=1:1000
    for i=1:m
        for j=1:n
            A_hat(i,j)=round(mu+b1(1,i)+b2(1,j));
            if A_sparse(i,j)~=0
                error(i,j)=A_sparse(i,j)-A_hat(i,j);
            end
        end
    end
    b1=b1+alpha1*sum(error');b2=b2+alpha1*sum(error);
    L1(1,iter)=0.5*sum(sum(error.^2));
end
RMSE=sqrt(sum((A_real(pos1)-A_hat(pos1)).^2)/length(pos1))
```

这样得到$\hat{A}$为

$$\hat{A}=\begin{bmatrix}48 & 41 & 505 & 510\\ -44 & -51 & 412 & 418\\ 219 & 212 & 676 & 681\\ 207 & 200 & 664 & 670\\ 115 & 107 & 571 & 577\end{bmatrix}$$

相应的 RMSE 为 **110.65**，从原来“？”位置的填补情况来看，这个结果并不理想（因为交通流量都是非负的），试想一下：如果采用$\hat{a}_{ij}=\sum_{q=1}^{r}u_{iq}v_{jq},i=1,\ldots,5,j=1,\ldots,4$填补的效果如何呢？

由于在浅谈张量分解（一）：如何简单地进行张量分解？一文中已经推导了u_{iq}和v_{jq}的更新公式，即$u_{iq}\Leftarrow u_{iq}+\alpha_2\sum_{j:(i,j)\in S}e_{ij}v_{jq}$，$v_{jq}\Leftarrow v_{jq}+\alpha_2\sum_{i:(i,j)\in S}e_{ij}u_{iq}$，其中，$q=1,\ldots,r$。利用 MATLAB 编写程序，具体如下。

```
A_sparse=[0,90,449,517;0,0,412,0;192,0,697,687;185,0,699,657;164,58,0,0];
A_real=[208,90,449,517;104,43,412,411;192,77,697,687;185,115,699,657;...
    164,58,696,599];
pos=find(A_sparse==0);[m,n]=size(A_real);r=3;
U=0.1*rand(m,r);V=0.1*rand(n,r);error=zeros(m,n);
alpha2=0.0001;
for iter=1:1500
    A_hat=round(U*V');error=A_sparse-A_hat;error(pos)=0;
    U_plus=U+alpha2*error*V;
    V_plus=V+alpha2*error'*U;
    U=U_plus;V=V_plus;
    L2(1,iter)=0.5*sum(sum(error.^2));
end
RMSE=sqrt(sum((A_real(pos)-A_hat(pos)).^2)/length(pos))
```

这样得到$\hat{A}$为

$$\hat{A}=\begin{bmatrix}139 & 89 & 449 & 517\\ 114 & 46 & 412 & 408\\ 192 & 75 & 697 & 687\\ 186 & 59 & 699 & 657\\ 163 & 59 & 599 & 579\end{bmatrix}$$

相应的 RMSE 为 **47.21**，其中，程序中的更新公式是$U\Leftarrow U+\alpha_2EV$和$V\Leftarrow V+\alpha_2E^TU$，$E$是由$e_{ij},i=1,\ldots,5,j=1,\ldots,4$构成的残差矩阵。接下来考虑使用迭代的奇异值分解来进一步增强填补的效果，同时，迭代的奇异值分解最终返回的三组件也可以作为分析特征的工具。

4 迭代的奇异值分解

Charu C. Aggarwal 在其著作《Recommender systems》第 115 页给出的算法非常简单，将缺失元素初始化填补阶段得到的$\hat{A}$记为A_f，按照如下操作：

- 第一步：对A_f进行奇异值分解，只选取前$k \leq \min(m,n)$个最大的奇异值和其对应的特征向量，得到$Q_k \Sigma_k P_k^T$；
- 第二步：用$Q_k \Sigma_k P_k^T$更新矩阵A（最初的，不是A_f）缺失位置的元素，记为$\hat{A}$，若未收敛，令$A_f = \hat{A}$，返回第一步。

这个方法对初始化的A_f要求较高，另外，由于$Q_k \Sigma_k P_k^T$在缺失位置的元素上会不同于A_f，最终收敛时会输出最终的$\hat{A}$.

加载所有元素缺失的位置都用均值（即$\mu = 401$）填补的矩阵$\hat{A}$

，利用 MATLAB 编写程序，具体如下。

```
A_sparse=[0,90,449,517;0,0,412,0;192,0,697,687;185,0,699,657;164,58,0,0];
A_real=[208,90,449,517;104,43,412,411;192,77,697,687;185,115,699,657;...
    164,58,696,599];
A_hat=[401,90,449,517;401,401,412,401;192,401,697,687;185,401,699,657;...
    164,58,401,401];
pos=find(A_sparse==0);[m,n]=size(A_real);k=2;
A_f=A_hat;
for iter=1:8000
    [Q,Sigma,P]=svds(A_f,k);
    A_hat=Q*Sigma*P';
    A_f(pos)=A_hat(pos);
    convergence(1,iter)=sum(A_hat(pos).^2);
end
RMSE=sqrt(sum((A_real(pos)-A_f(pos)).^2)/length(pos))
```

相应的 RMSE 为

161.85

；加载

$$\hat{a}_{ij} = \mu + b_i^{(1)} + b_j^{(2)}, i = 1, \ldots, 5, j = 1, \ldots, 4$$

形式得到的

$$\hat{A} = \begin{bmatrix} 48 & 41 & 505 & 510 \\ -44 & -51 & 412 & 418 \\ 219 & 212 & 676 & 681 \\ 207 & 200 & 664 & 670 \\ 115 & 107 & 571 & 577 \end{bmatrix}$$

，利用 MATLAB 编写程序，具体如下。

```
A_sparse=[0,90,449,517;0,0,412,0;192,0,697,687;185,0,699,657;164,58,0,0];
A_real=[208,90,449,517;104,43,412,411;192,77,697,687;185,115,699,657;...
    164,58,696,599];
A_hat=[48,41,505,510;-44,-51,412,418;219,212,676,681;207,200,664,670;...
    115,107,571,577];
```

```
pos=find(A_sparse==0);[m,n]=size(A_real);k=2;
A_f=A_hat;
for iter=1:100
    [Q,Sigma,P]=svds(A_f,k);
    A_hat=Q*Sigma*P';
    A_f(pos)=A_hat(pos);
    convergence(1,iter)=sum(A_hat(pos).^2);
end
RMSE=sqrt(sum((A_real(pos)-A_f(pos)).^2)/length(pos))
```

相应的 RMSE 为

110.15

；加载

$$\hat{a}_{ij}=\sum_{q=1}^{r}u_{iq}v_{jq},i=1,\ldots,5,j=1,\ldots,4$$

形式得到的

$$\hat{A}=\begin{bmatrix}139 & 89 & 449 & 517\\ 114 & 46 & 412 & 408\\ 192 & 75 & 697 & 687\\ 186 & 59 & 699 & 657\\ 163 & 59 & 599 & 579\end{bmatrix}$$

，即

```
A_sparse=[0,90,449,517;0,0,412,0;192,0,697,687;185,0,699,657;164,58,0,0];
A_real=[208,90,449,517;104,43,412,411;192,77,697,687;185,115,699,657;...
    164,58,696,599];
A_hat=[139,89,449,517;114,46,412,408;192,75,697,687;186,59,699,657;...
    163,59,599,579];
pos=find(A_sparse==0);[m,n]=size(A_real);k=2;
A_f=A_hat;
for iter=1:500
    [Q,Sigma,P]=svds(A_f,k);
    A_hat=Q*Sigma*P';
    A_f(pos)=A_hat(pos);
    convergence(1,iter)=sum(A_hat(pos).^2);
end
RMSE=sqrt(sum((A_real(pos)-A_f(pos)).^2)/length(pos))
```

相应的 RMSE 为 **47.03**。通过迭代奇异值分解，缺失位置的元素会随着迭代过程而趋于收敛，另外，迭代奇异值分解过程有助于我们提取矩阵行和列的特征，并且左奇异向量间的正交特性和右奇异向量的正交特性能够帮助我们找到主要的模式。

5 相关阅读

本文主要参考了 Charu C. Aggarwal 于 2016 年出版的著作《Recommender systems》（链接：Recommender Systems）。