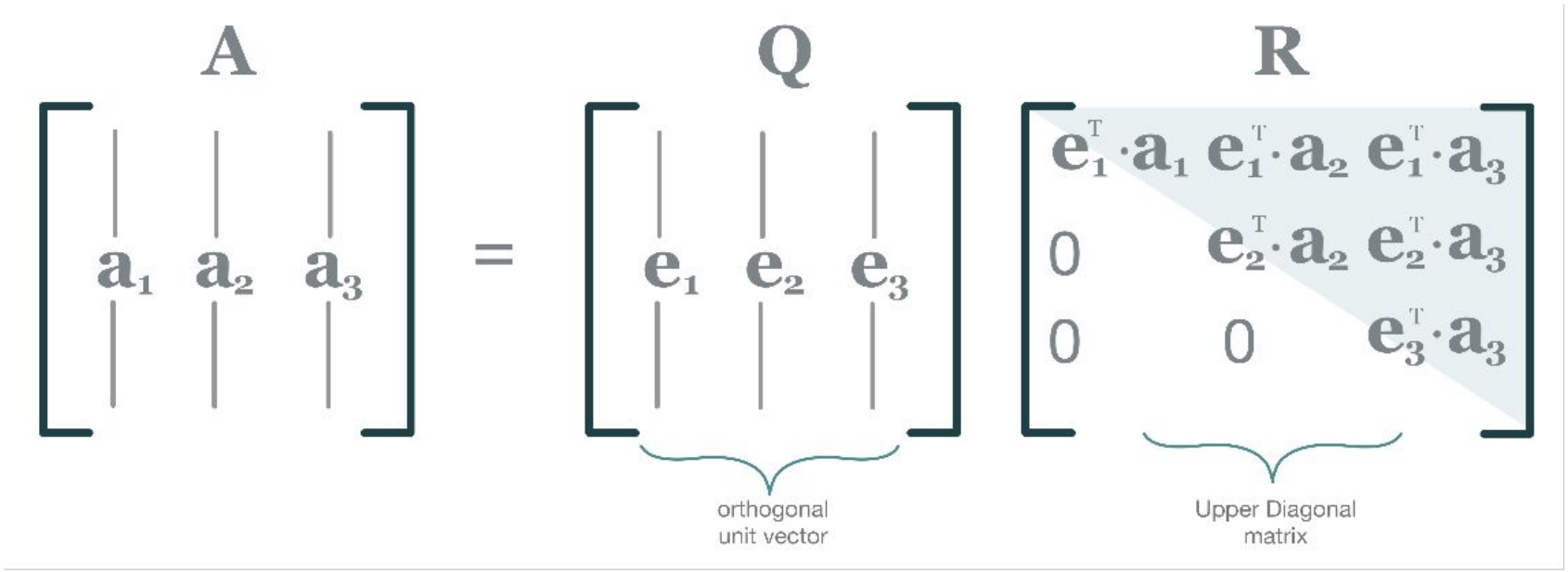
计算物理 第一部分

第6讲　本征值和本征矢量数值求解



李强　北京大学物理学院中楼411

qliphy0@pku.edu.cn，15210033542

引言

这一章中，我们在一开始的时候将继续讨论和线性代数有关的内容，特别是本征值问题。我们也将对大型稀疏矩阵的线性代数问题进行简单的介绍。

让我们首先回忆一下关于本征值和本征矢量的一些数学定义和一些重要的结果。线性代数的知识告诉我们，要求出一个矩阵的本征值，我们只需要令其特征多项式为零即可：

$$\chi_A(\lambda) = (-)^n \det(A - \lambda I) = (\lambda - \lambda_1)^{\sigma_1} \cdots (\lambda - \lambda_k)^{\sigma_k} = 0,$$

其中$\chi_A(\lambda)$为A的特征多项式，其中各个本征值$\lambda_1, \cdots, \lambda_k$被认为是不同的，它们相应的重数$\sigma_1, \cdots, \sigma_k$**被称为相应本征值的代数多重度(algebraic multiplicity)**。我们把矩阵A的所有本征值的集合称为矩阵A的谱，记为$\sigma(A)$。矩阵A的**谱半径$\rho(A)$被定义为A的本征值(可能是复的)中的最大的模**。

本征值和本征矢量

对某个$\lambda \in \sigma(A)$，由所有相应的本征矢以及零矢量一起构成了一个矢量空间，称为**本征值λ的本征空间**。这个空间可以认为是线性变换矩阵$(A-\lambda I)$的核(kernel)：**ker(A−λI)**。如果我们用rank(A)来表示一个矩阵A的秩(rank)的话，与一个本征值λ相应的本征空间的维数：

$$\dim[\ker(A-\lambda I)] = n - \operatorname{rank}(A-\lambda I)$$

被称为该本征值的几何多重度(geometric multiplicity)。可证明的是，对任何本征值而言，其几何多重度一定不大于代数多重度。**几何多重度严格小于代数多重度的本征值被称为亏损本征值**，如果矩阵A的本征值中至少有一个是亏损的，我们就称矩阵A是有亏损矩阵

$$A = \begin{pmatrix} \lambda & 1 \\ & \lambda \end{pmatrix}$$

这个矩阵的代数多重度是2。但这个本征值λ所对应的非零本征矢量却只有1个：$\begin{pmatrix} 1 & 0 \end{pmatrix}^T$

所以几何多重度小于代数多重度，这个矩阵是亏损的。

相似变换

为了求出矩阵的本征值，我们往往需要对矩阵进行相似变换。对于A∈C^{n×n}和另一个和它同样大小的、非奇异的矩阵C可以构建一个新的变换后的矩阵

$$\tilde{A} = C^{-1}AC$$

这称为原矩阵A的一个相似变换(similar transformation)。这**两个矩阵被称为是相似的，因为两者具有完全一致的谱**。事实上如果(λ,x)是A的一个本征对，那么容易验证(λ,C^{-1}x)必定是 ~A 的本征对。反之亦然。**如果我们进行变换的矩阵C是幺正的**，这时我们就称A与 ~A是**幺正相似**的。幺正相似的矩阵在物理上往往是完全等价的，不同的只是选取了不同的表象而已。

矩阵的相似性是一个非常重要的概念，因为**相似矩阵有许多相似不变量，比如说：特征多项式、特征值(包括代数多重度和几何多重度)、行列式、迹和秩等**，并且特征向量也可以借助于相似变换矩阵求出。这自然引出了寻找相似矩阵中的具有代表性矩阵的问题，代表矩阵当然越简单越好。我们把这类具有代表性的矩阵称为**标准型**。

Schur 标准型

我们讨论的第一个标准型是所谓的Schur标准型。有一个关于任意矩阵的重要的结构性定理是所谓的Schur分解定理。这里不加证明的引用。

定理：任给A∈C^{n×n}，一定存在一个**幺正矩阵U**使得

$$U^{-1}AU \equiv U^{\dagger}AU = \begin{pmatrix} \lambda_1 & \tilde{a}_{12} & \cdots & \tilde{a}_{1n} \\ 0 & \lambda_2 & & \tilde{a}_{2n} \\ \cdots & & \ddots & \cdots \\ 0 & \cdots & 0 & \lambda_n \end{pmatrix}$$

其中的对角元$\lambda_i, i=1,2,\cdots,n$为A的本征值（不一定不同）。

Schur分解定理告诉我们，一个任意的复矩阵一定可利用某个（不一定是唯一的）幺正变换U化为一个上三角矩阵，**变换后的矩阵之对角元为原矩阵的本征值。**这个形式的上三角矩阵称为原矩阵A的舒尔形式。

Schur 标准型

需要着重指出的是，**虽然任何复矩阵都可以利用幺正矩阵化为上三角矩阵，但这并不意味着任何的矩阵都是可对角化的**，后者要求矩阵A∈C^{n×n}必须与一个对角矩阵(而不仅仅是上三角矩阵)相似。一个一般的A并不一定是可对角化的。由于本征值λ可以由相应矩阵的特征多项式为零给出，按照代数学基本定理，**本征值在复数域内总是存在n个根的(可能重复)**。因此，**Schur定理中的对角元总是存在的(并且原则上可以数值求出)**；但是对不可对角化的矩阵来说，我们并不能找到一个非奇异的相似变换C使得C^-1AC是完全对角的。**这种情形出现在矩阵的某个本征值是亏损的时候，此时我们无法找到n个线性独立的本征矢来张成全空间C^n**。这时的矩阵A一定是不可对角化的。

Frobenius的学生

Schur定理推论

原始的Schur分解定理中并没有对**复矩阵A**做任何的限制。如果我们要求它再具有一些额外的性质，比如对于**厄米性**，或者其他在幺正变换下保持不变的性质，我们很容易发现Schur定理的下列重要推论：

推论：给定任意一个厄米矩阵 $A \in C^{n\times n}$，$A^\dagger = A$，一定存在一个幺正变换 U(满足 $U^\dagger = U^{-1}$)将其对角化：$U^{-1}AU = \mathrm{Diag}(\lambda_1, \cdots, \lambda_n)$，其中所有的本征值 λ_i 都是实数，并且相应的幺正矩阵 U 的列向量就是相应的本征矢量，即 $U = [x_1, x_2, \cdots, x_n]$ 满足 $Ax_i = \lambda_i x_i$，而且我们可以选择 x_i 使得它们两两正交并且张成整个 C^n 空间。

显然，厄米性在幺正变换下保持不变。因此，对一个厄米矩阵来说，我们可以将其变换为上三角矩阵的同时也将其变换为了下三角矩阵。一个同时为上和下三角的矩阵一定就是对角矩阵，即矩阵已经被对角化了。在数学上，厄米矩阵的一个推广是所谓的正规矩阵(normal matrices)。**一个矩阵A∈^C{n×n}如果满足** $AA^\dagger = A^\dagger A$ **就称为正规矩阵。**正规矩阵就是和自身的厄米共轭对易的矩阵。厄米矩阵当然是正规矩阵，因为任何矩阵都和自己对易。另一个在物理上经常用到的正规矩阵就是幺正矩阵，

因为这时候 $AA^\dagger = A^\dagger A = I$

Jordan 标准型

正规矩阵几乎满足推论中关于厄米矩阵完全一样的性质，即它也属于可对角化的矩阵，唯一的区别是它的本征值λ_i不再保证一定是实数。另一点值得提及的是，对正规矩阵而言（当然也包括厄米矩阵），Schur定理中到的幺正变换U实际上是唯一的（如果我们把不同的本征矢量重新排列一下的U矩阵看成是同一个矩阵）。

如果我们能通过相似变换把一个矩阵变成对角矩阵，那当然是最好的。由于一般矩阵与对角矩阵不相似，因此我们退而求其次，寻找几乎对角的矩阵，**对于各种标准型而言，最接近对角矩阵的是所谓的Jordan标准型。一个$\nu\times\nu$阶的Jordan标准矩阵的典型形式（假定$\nu\geq1$）为：**

$$J_\nu(\lambda)\equiv\begin{pmatrix}\lambda & 1 & \cdots & & 0\\ 0 & \lambda & \ddots & & \cdots\\ \cdots & & \ddots & \ddots & \\ & & & \lambda & 1\\ 0 & \cdots & & 0 & \lambda\end{pmatrix}_{\nu\times\nu}$$

可验证，这个矩阵的本征值为λ，这一般是一个多重根（假定$\nu\geq2$），其代数多重度为ν。但是如果你试图去寻找它相应的本征矢的话，你会发现它仅有一个线性独立的本征矢，即：x= (1,0,···,0)^T。因此，这是一个亏损矩阵，是不可对角化的

Jordan 标准型

上面的Jordan标准矩阵虽然看起来十分特殊，但是线性代数中的一个重要定理告诉我们，任何一个n×n的复矩阵都可以通过一个非奇异的相似变换化为由若干个形如上述Jordan块矩阵所构成的块对角矩阵：

定理：对任意的 $A \in C^{n\times n}$，总可以找到一个非奇异的矩阵 $X \in C^{n\times n}$ 使得

$$X^{-1}AX = J = \mathrm{Diag}(J_{\nu_1}(\lambda_1), \cdots, J_{\nu_k}(\lambda_k)),$$

其中 $J_\nu(\lambda)$ 是 Jordan 块矩阵 (若 $\nu \geq 2$)，$\lambda_1, \cdots, \lambda_k$ 是矩阵的本征值；而如果 $\nu = 1$，我们约定 $J_1(\lambda) = \lambda$。这里需要说明的是，不同约当块里面的本征值是完全可能相等的，也就是说 $\begin{pmatrix} \lambda & 1 & & \\ 0 & \lambda & 0 & \\ & 0 & \lambda & 1 \\ & & 0 & \lambda \end{pmatrix}$ 和 $\begin{pmatrix} \lambda & 1 & & \\ 0 & \lambda & 1 & \\ & 0 & \lambda & 1 \\ & & 0 & \lambda \end{pmatrix}$ 是完全不等价的约当标准型。前者由两个 2 阶约当块构成，后者由一个 4 阶约当块构成。

Jordan 标准型

这个定理告诉我们，每个矩阵都可以通过相似变换，变换为由Jordan块矩阵构成的标准形式。它称为该矩阵的Jordan标准型。其中的每一个块矩阵$J_\nu(\lambda)$称为原来矩阵的一个Jordan块。我们一般并不会真正去计算一个矩阵的Jordan标准型。**一个矩阵的Jordan标准型的数值计算是十分不稳定的。**

$$A=\begin{pmatrix}1 & 1\\ \epsilon & 1\end{pmatrix}$$

只要ε稍微偏离0，那么对应Jordan标准型为

$$\begin{pmatrix}1+\sqrt{\epsilon} & 0\\ 0 & 1-\sqrt{\epsilon}\end{pmatrix}$$

而如果ε=0，那么上式中的

$$A=\begin{pmatrix}1 & 1\\ 0 & 1\end{pmatrix}$$

已经是 Jordan 标准型了。

正因为如此，并没有非常稳定的数值方法来完成这个任务。**数值上更稳定的是计算矩阵的Schur标准型，就是那种上三角的形式。**不过研究**Jordan标准型对我们后面的本征值以及本征矢的计算有着重要的理论指导意义。**因为一个矩阵是否可以对角化，是与该矩阵的Jordan标准型的形式直接挂钩的。

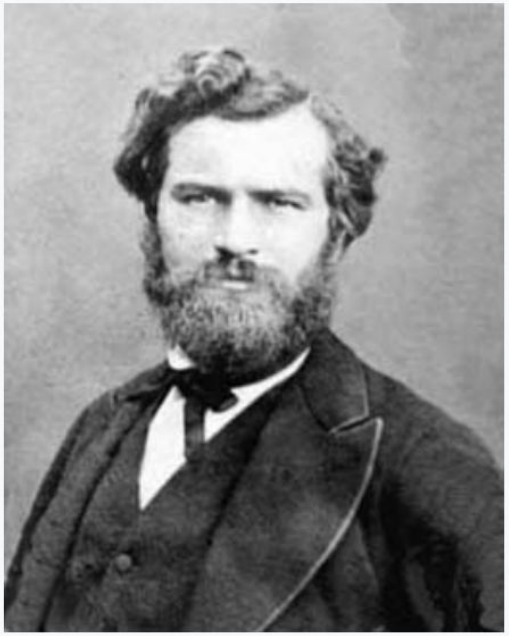
Jordan 标准型

- 对每一个给定的本征值λ，可以有不止一个Jordan块。也即对一个给定的本征值，可以出现多个与之相应的Jordan块，并且每个块的大小还可不同。但正如我们前面刚刚提到的，对每个块$J_\nu(\lambda)$来说，对应于其相应本征值(即λ)，有且只有一个线性独立的本征矢。对于不同的约当块而言，不管本征值是否相同，本征矢量肯定是线性独立的。因此，**一个矩阵的Jordan块的总数就是能够找到的线性独立的本征矢的总数。**
- 一个矩阵要是可对角化的充要条件就是它的Jordan标准型中的所有的Jordan块都是1×1的Jordan块矩阵。由此我们看到，**只要矩阵的标准型中有某个Jordan块的尺寸是超过1的，它一定是不可对角化的**，而这恰恰与亏损矩阵的定义是一致的。因此，**矩阵的非亏损性与它的可对角化性是完全等价的概念。**
- 我们前面看到，一个正规矩阵(重要的例子包括厄米矩阵、幺正矩阵等等)是可对角化的。另一个可对角矩阵的例子是所有本征值（无论是实的还是复的）都不相同的矩阵一定也是可以对角化的，因为它的每个 Jordan块都是1×1的。事实上，**如果所有本征值都不同，其相应的本征矢必定是线性无关的从而一定张成全空间。**

Jordan 标准型

上面已经提及，矩阵的亏损性直接与矩阵的Jordan标准型的形式有关。另一个相关的概念是矩阵的**减次性**。如果矩阵A的任何两个不同的约当块中包含了同一个本征值λ_i，我们就称相应的本征值是减次的。（更多细节参看刘川老师讲义）

Camille Jordan



Born	5 January 1838 Lyon
Died	22 January 1922 (aged 84) Paris

$$A = \begin{bmatrix} 5 & 4 & 2 & 1 \\ 0 & 1 & -1 & -1 \\ -1 & -1 & 3 & 0 \\ 1 & 1 & -1 & 2 \end{bmatrix}$$

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

$$P^{-1}AP = J = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 4 & 1 \\ 0 & 0 & 0 & 4 \end{bmatrix}.$$

QR 算法

QR算法是20世纪十大算法之一，也是目前各界应用的最为广泛的数值求解本征值和本征矢量的算法之一。一般来说，如果待求的矩阵不是特别巨大(n≤3000)，那么QR算法实际上是最为合适的算法。

在讲QR算法之前，我们不妨来考虑这样两个问题：

- **首先，什么样的矩阵易于求出全部本征值 ⇒ 这个问题等价于，我们需要先对矩阵做哪些有用的约化，使得它容易求解本征值**
- **第二个问题是，如果我们要对矩阵进行操作，那么什么样的操作变换能保持本征值不变**

对于第一个问题，我们知道对角型、上（下）三角型的矩阵很容易求本征值，本征值就是他们的对角线元素。对于分块对角型，或者分块上（下）三角型的矩阵，如果对角块的维数较小，本征值也是容易求的。

对于第二个问题，我们知道相似变换C^-1AC能保持矩阵特征值不变，但在实际计算中，**C为正交矩阵是一个更好的选择**，因为正交矩阵很容易求逆。而且正交矩阵因为要满足正交性，通常矩阵元素的数量级差别不大，有关的计算是数值稳定的。

QR 算法

所以QR算法的出发点就是基于以上两个考虑，**它的想法是要寻找任意实矩阵A∈R^{n×n}的正交变换T=Q^{T}AQ，使得T是个上三角矩阵，其中**

Q∈R^{n×n}是实正交矩阵。正交变换保证了T的本征值肯定与A的本征值一致。在**实际做的过程当中，我们是先对矩阵A做QR分解，把A写成正交矩阵和上三角矩阵的乘积，然后做迭代得到T=Q^{T}AQ。**

它的步骤为：取$T_0=A$，$T_k = Q_k R_k$ 1,2,···)做QR分解。

得到Q_k和R_k以后，将它顺序相反乘在一起，得到 $T_{k+1} = R_k Q_k$

所以 $T_{k+1} = Q_k^T T_k Q_k,$ 每一步都在做正交变换，我们希望迭代到最后

$$\lim_{k\to\infty} T_k = T$$

由于迭代之后新的矩阵都与原先的矩阵只差一个正交变换。**因此新的矩阵的条件数一定不坏于原先的矩阵。**这一点对于算法的稳定性是非常重要的。

QR 算法:实舒尔形式

QR算法的初衷是利用正交变换将A∈R^{n×n}约化为一个上三角矩阵。不幸的是，这一点并不是总能做到的。**因为一个任意的实矩阵有可能有成对的复本征值，但是实的正交矩阵变换后的矩阵元一定仍然保持是实的。**如果上述变换总能成立就意味着变换后的三角矩阵的对角元–也就是它的本征值–一定都是实数，与一般实矩阵可以有复本征值矛盾。但我们可以把它约化到一个**近似的上三角矩阵**：

$$Q^T AQ = \begin{pmatrix} R_{11} & R_{12} & \cdots & R_{1m} \\ 0 & R_{22} & \cdots & R_{2m} \\ \cdots & \cdots & \ddots & \cdots \\ 0 & 0 & \cdots & R_{mm} \end{pmatrix}$$

其中位于对角线上的块矩阵R_{ii}要么就是一个实数要么就是一个2×2的、具有一对复共轭根的实矩阵。我们称这样形式的矩阵为**实矩阵A的实舒尔分解或者实舒尔形式**。

原则上只要我们不断进行QR迭代，最后矩阵就会被约化为实舒尔形式，但我们还需要知道这个迭代的收敛速度。

QR 算法:收敛速度

QR迭代的收敛速度，有下面这个定理：给定A∈R^{n×n}并假设它的本征值的排序为

$$|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n|$$

那么我们有

$$\lim_{k\to\infty} T_k = \begin{pmatrix} \lambda_1 & t_{12} & \cdots & t_{1n} \\ 0 & \lambda_2 & \cdots & t_{2n} \\ \cdots & \cdots & \ddots & \cdots \\ 0 & 0 & \cdots & \lambda_n \end{pmatrix}$$

其中非对角元的收敛速度为

$$|t_{i,j}^{(k)}| \sim O\left(\left|\frac{\lambda_i}{\lambda_j}\right|\right)^k, \quad i > j, \quad k \to \infty$$

此外，如果加上A是对称矩阵的假设，那么T_k可直接收敛到对角矩阵。这个定理告诉我们，**如果矩阵的所有本征值的模都不相等，那么经过QR迭代，我们不仅仅会获得矩阵的实舒尔形式，我们甚至可以直接获得其舒尔形式–即所有的本征值。**但同时我们也看到，如果矩阵具有两个模十分接近的本征值，那么相应的非对角元可以收敛得很慢。**当矩阵具有完全等模的本征值时，QR迭代甚至可能不收敛。（如：一对复共轭根）**

QR 算法

ref

Photos courtesy of Frank Uhlig

Who is John Francis?

- born near London in 1934
- employed in late 50's, Pegasus computer
- linear algebra, eigenvalue routines
- primitive computer
- no software
- experimented with a variety of methods
- invented His algorithm and programmed it
- moved on to other things

"Actually, as far as John recalled in 2008, there had been no reaction, none whatsoever in the early 1960s when his seminal papers appeared, and then he had left the field."

QR 算法:Householder 变换

我们可以看到QR算法执行的每一步迭代，都需要算一次QR分解，它的计算量大概是O(n^3)。所以总的计算量是O(n^3) 再乘上迭代次数，这个迭代次数主要与矩阵的本征值分布相关，对于收敛很慢的矩阵，也许需要迭代的次数非常大。因此，**有必要在我们进行迭代之前，先将矩阵化为简单的形式。常用的方法包括Householder变换和Givens变换。**
我们下面先讲一下Householder变换。

事实上，Householder变换是Householder形式化的矩阵计算的分解方法的一部分，也是属于20世纪十大算法之一。

ARTICLE

Unitary Triangularization of a Nonsymmetric Matrix

Author: Alston S. Householder Authors Info & Affiliations

Journal of the ACM, Volume 5, Issue 4 • Oct. 1958 • pp 339–342 • https://doi.org/10.1145/320941.320947

Published: 01 October 1958

https://dl.acm.org/doi/10.1145/320941.320947

Alston Scott Householder

Born	5 May 1904
Died	4 July 1993 (aged 89)
Alma mater	University of Chicago

Householder 变换

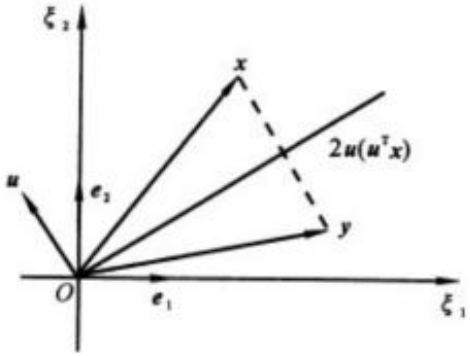
我们考虑v∈R^n中的一个矢量。构造如下的矩阵：

$$P = 1_{n\times n} - 2vv^T/||v||_2^2$$

其中||v||2=v^Tv是矢量v的欧氏模方。当我们将这个矩阵乘以x∈R^n时，即y=Px，它的作用**将原来的矢量x相对于以v为法线方向的超平面进行镜面反射**。或者说，x中与v垂直的分量不变，但是与v平行的部分反了一个符号(镜面反射的特点)。这个变换一般称为Householder变换；相应的v称为Householder 矢量，P 称为与v相对应的Householder矩阵。

注意，**Householder 矩阵并不改变矢量的长度，因此实际上是一个正交矩阵。而且它还是一个对称矩阵。**另一点值得指出的是，当我们需要计算一个Householder矩阵乘以一个矢量(例如Px)的时候，我们并不需要首先将Householder矩阵存在内存中，然后计算矩阵乘以矢量，**我们需要的只是两个矢量的内积 v^Tx，v^Tv等等。因此仅仅是一个O(n)量级的计算而不是通常的O(n^2)的计算**。

Householder 变换



如果有两个矢量x和y，他们的**欧氏模方相同**，那么我们可通过Householder变换将x变成y：Px=y。这里我们只需要**将矢量v的方向取为v=x−y即可**。

这个定理的一个推论是，我们一定可以把x，通过Householder变换，**变换成为只有第m个方向有非零分量的矢量**

$$Px = (0, \cdots, 0, \pm||x||_2, 0, \cdots, 0)^T$$

这里只需要把 y 取为 $y = \mp||x||_2 e_m$，而矢量 v 取为 $v = x \pm ||x||_2 e_m$。

我们可以直接用Householder变换对矩阵A进行QR变换

$$\begin{pmatrix} x & x & x & x \\ x & x & x & x \\ x & x & x & x \\ x & x & x & x \end{pmatrix} \rightarrow \begin{pmatrix} x & x & x & x \\ 0 & x & x & x \\ 0 & x & x & x \\ 0 & x & x & x \end{pmatrix} \rightarrow \begin{pmatrix} x & x & x & x \\ 0 & x & x & x \\ 0 & 0 & x & x \\ 0 & 0 & 0 & x \end{pmatrix}$$

但这并不是我们要讨论的重点。因为这只是对矩阵A左乘了一系列的正交矩阵，它做的并不是正交变换。我们希望做的一件事情，**是利用Householder矩阵将任意的n×n矩阵通过正交变换化为上Hessenberg形式。**

插曲：求解方程Ax = b 还是要求矩阵A 的本征谱

对前者而言，我们只需对矩阵A进行QR分解。事实上，如果知道A=QR，那么方程Ax=b即等价为QRx=b，两边乘以Q^T我们得到Rx=Q^Tb。这是一个三角矩阵的求解问题，可以通过前面提及的反代法求出解。

要对 A 进行 QR 分解我们可以对矩阵 A 不停地左乘以适当的 Householder 矩阵即可。注意到

$$P \cdot A = P \cdot [a_1^{(0)}, a_2^{(0)}, \cdots, a_n^{(0)}] = [Pa_1^{(0)}, Pa_2^{(0)}, \cdots, Pa_n^{(0)}] , \tag{6.60}$$

其中 $a_i^{(0)}$，$i = 1, \cdots, n$ 为矩阵 A 的各个列所对应的矢量。因此我们可以首先选取第一个 Householder 矩阵 P_1 使得，

$$P_1 a_1^{(0)} = k e_1 , \tag{6.61}$$

其中 $k = \pm ||a_1^{(0)}||_2$，$e_1 = (1, 0, \cdots, 0)^T$ 是 $\mathbb{R}^n$ 中的第一个单位矢量。这样一来，矩阵 $P_1 A = [a_1^{(1)}, a_2^{(1)}, \cdots, a_n^{(1)}]$ 的各个列看起来是这样的样子：$a_1^{(1)} = (\pm ||a_1^{(0)}||, 0, 0, \cdots, 0)^T$，而其余的各个列则没有什么特别的。换句话说，通过左乘上一个 Householder 矩阵，我们将所得到的新的矩阵的第一列元素从第二行一直到第 n 行都设置为零了。我们可以将矩阵写为分块的形式：

$$P_1 A = \begin{bmatrix} \pm ||a_1^{(0)}|| & \alpha^T \\ 0 & \bar{A} \end{bmatrix} . \tag{6.62}$$

其中 α 是一个一般的 $n-1$ 个分量的矢量而 $\bar{A} \in \mathbb{R}^{(n-1)\times(n-1)}$ 则是一个一般的矩阵。

插曲：求解方程Ax = b 还是要求矩阵A 的本征谱

第二步中我们寻求如下形式的矩阵：

$$P_2 = \begin{bmatrix} 1 & 0 \\ 0 & \tilde{P}_2 \end{bmatrix}, \tag{6.63}$$

其中 $\tilde{P}_2 \in \mathbb{R}^{(n-1)\times(n-1)}$ 是一个恰当的 $(n-1)\times(n-1)$ Householder 矩阵。于是我们有，

$$P_2 P_1 A = \begin{bmatrix} 1 & 0 \\ 0 & \tilde{P}_2 \end{bmatrix} \begin{bmatrix} \pm||a_1^{(0)}|| & \alpha^T \\ 0 & \bar{A} \end{bmatrix} = \begin{bmatrix} \pm||a_1^{(0)}|| & \alpha^T \\ 0 & \tilde{P}_2 \bar{A} \end{bmatrix}. \tag{6.64}$$

于是化为上三角矩阵的问题化为阶数减少一的同样问题。我们要做的是根据矩阵 $\bar{A}$ 的各个列适当地选择 $(n-1)\times(n-1)$ 的 Householder 矩阵 $\tilde{P}_2$，它可以将 $\bar{A}$ 的 (11) 元素以下的各个元素设为零。最终，通过类似的 $n-1$ 个恰当的 Householder 矩阵，我们有

$$P_{n-1} \cdots P_2 P_1 A \equiv Q^T A = R, \tag{6.65}$$

其中 R 是一个上三角矩阵。由于各个 P_i 都是 Householder 矩阵从而是正交矩阵，因此它们的乘积仍然是一个正交矩阵，记为 Q^T。于是 $A = QR$ 就完成了矩阵 A 的 QR 分解。这个方法所耗费的计算量大约是 $(2/3)n^3$。

Householder 变换

如果我们希望对矢量x的前k个分量保持不变，将从第k+2个以及以后的所有分量都设为零。由于Householder为正交矩阵不改变矢量的模，因此新的矢量的第k+1个分量一定大小为||x^(n-k)||2，其中x^(n-k)为矢量x的后n-k个分量构成的矢量。对于这个操作我们可以用如下的Householder矩阵：

$$P_{(k)} = \begin{pmatrix} 1_k & 0 \\ 0 & R_{n-k} \end{pmatrix}, \quad R_{n-k} = 1_{n-k} - 2\frac{\omega^{(k)}(\omega^{(k)})^T}{||\omega^{(k)}||_2^2}$$

其中 $\omega^{(k)} \in R^{n-k}$ 为 R^{n-k} 中的一个矢量。我们应当取

$$\omega^{(k)} = x^{(n-k)} \pm ||x^{(n-k)}||_2 e_1^{(n-k)}$$

其中 $e_1^{(n-k)}$ 是 R^{n-k} 中的第一个单位矢量。很明显 $P_{(k)}$ 满足 $P_{(k)}^T = P_{(k)}$ 和 $P_{(k)}^T P_{(k)} = 1$。

Householder 变换

有了 $P_{(k)}$ 以后，我们把矩阵 A 看成 2×2 的分块矩阵 $A=\begin{pmatrix} a_{11} & r_1^T \\ c_1 & A_{22}\end{pmatrix}$，我们对它左乘 $P_{(1)}^T$，右乘 $P_{(1)}$，得到

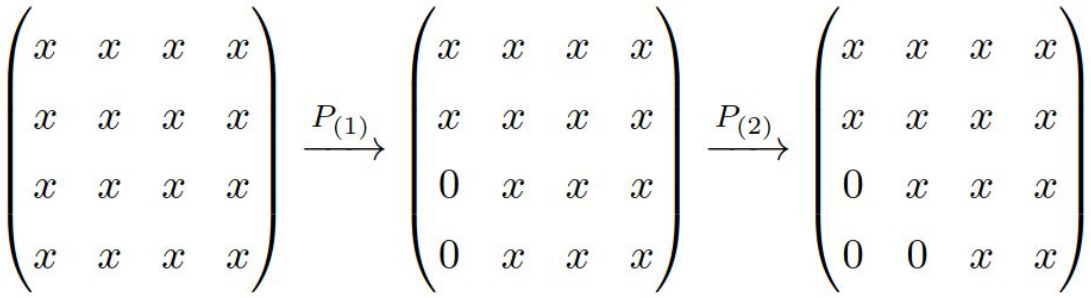
$$P_{(1)}^T A=\begin{pmatrix} a_{11} & r_1^T \\ \gamma_1 e_1^{(n-1)} & R_{n-1}A_{22}\end{pmatrix},\quad P_{(1)}^T A P_{(1)}=\begin{pmatrix} a_{11} & r_1^T R_{n-1} \\ \gamma_1 e_1^{(n-1)} & R_{n-1}A_{22}R_{n-1}\end{pmatrix}\to\begin{pmatrix} x & x & x & x \\ x & x & x & x \\ 0 & x & x & x \\ 0 & x & x & x\end{pmatrix}$$

对于任意的A∈R^{n×n}，我们可利用一连串矩阵：P(1)···P(n-2)试图将矩阵化为上Hessenberg形式。事实上，令A^(0)≡A，对任意的k≥1，我们有

$$A^{(k)}=P_{(k)}^T A^{(k-1)}P_{(k)}=(P_{(1)}\cdots P_{(k)})^T A(P_{(1)}\cdots P_{(k)})$$

或者等价地写成 $A^{(k)}=Q_{(k)}^T A^{(0)}Q_{(k)}$，其中正交矩阵 $Q_{(k)}=P_{(1)}\cdots P_{(k)}$ 为一系列 Householder 矩阵的乘积。基本上经过第一次变换:$A^{(1)}=P_{(1)}^T A^{(0)}P_{(1)}$，原先矩阵的第一列中 a_{21} 以下的数都会被变换为零; 经过第二个变换，在保持第一列的结构的同时，将矩阵的第二列的 a_{32} 以下的矩阵元都变换为零，等等。利用 Householder 矩阵将矩阵变换为 Hessenberg 形式的操作一般称为 Hessenberg-Householder 约化 (Hessenberg-Householder reduction)。下式以 4×4 矩阵为例，演示了这种约化的过程:

Householder 变换



值得提及的是，如果原先的矩阵A是一个对称矩阵，那么Householder变换后其对称性仍然保持。因此，**在经过Householder约化之后，一个对称矩阵一定能够化为一个对称的、三对角矩阵**。因为Householder变换要保证每个列矢量的模不变，所以我们没法直接得到对角矩阵，这当然也和我们之前提到过的并不是每个矩阵都可以对角化的有关。

如果我们把矩阵A化成了一个上Hessenberg矩阵，**由于Hessenberg矩阵已很接近上对角的形式，那么再对Hessenberg矩阵做QR分解，计算量就会大大下降。**事实上每次QR迭代的计算量可以**从O(n^3)减少到O(n^2)**。如果Hessenberg矩阵是三对角的形式，那么QR迭代的计算量可以进一步减小。另外，Hessenberg矩阵的大量矩阵元都已经为0。我们知道，做QR迭代时候，实际上收敛的效果是让很多非对角元直接收敛到0。**对于Hessenberg矩阵来讲，我们只需使少量的非零元收敛到0，QR迭代所需的步数也大为减小。**

Givens 变换

我们利用Hessenberg-Householder约化可以将任意的矩阵A∈R^{n×n}通过正交变换化为上Hessenberg形式。我们可以**进一步利用Givens变换矩阵（又称为Givens转动矩阵）把矩阵化成实舒尔形式**

Wallace Givens

Mathematician

James Wallace Givens, Jr. was a mathematician and a pioneer in computer science. He is the eponym of the well-known Givens rotations. Wikipedia

Born: 14 December 1910, Alberene, Virginia, United States

Died: 5 March 1993

Education: Princeton University

Karl Hessenberg

German mathematician

Karl Adolf Hessenberg was a German mathematician and engineer. The Hessenberg matrix form is named after him. From 1925 to 1930 he studied electrical engineering at the Technische Hochschule Darmstadt and graduated with a diploma. Wikipedia

Born: 8 September 1904, Frankfurt, Germany

Died: 22 February 1959, Frankfurt, Germany

Education: Technische Universität Darmstadt

$$G = \begin{pmatrix} c & s \\ -s & c \end{pmatrix}, \quad G^{-1} = \begin{pmatrix} c & -s \\ s & c \end{pmatrix},$$

其中c=cosθ，s=sinθ，这个2×2矩阵就是Givens矩阵。它的逆变换也很简单。适当选取θ的值，**我们可以用Givens旋转矩阵来消去向量的某个分量。**

Givens 变换

$$Gx = \begin{pmatrix} c & s \\ -s & c \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} \alpha \\ 0 \end{pmatrix} \qquad \alpha = \sqrt{x_1^2 + x_2^2}$$

因为**G是正交矩阵**，那么我们必有$|\alpha| = ||x||_2$，则

$$c = \frac{x_1}{\sqrt{x_1^2 + x_2^2}}, \quad s = \frac{x_2}{\sqrt{x_1^2 + x_2^2}}$$

如果x_1或者x_2是很大的值，有时候为避免上溢，会对公式进行调整，如果$|x_1| \geq |x_2|$，可以计算

$$t = \frac{x_2}{x_1}, \quad c = \frac{1}{\sqrt{1 + t^2}}, \quad s = ct.$$

一个在第一、第二轴的平面内转动的Givens矩阵的形式为：

$$G(1, 2, \theta) = \begin{pmatrix} G(\theta) & 0 \\ 0 & 1_{(n-2)\times(n-2)} \end{pmatrix}$$

Givens 变换

完成一次 Givens 变换以后 ($G_1^{(1)} = G(1,2,\theta_1)$)，我们得到

$$H^{(0)} = \begin{pmatrix} x & x & x & x \\ x & x & x & x \\ 0 & x & x & x \\ 0 & 0 & x & x \end{pmatrix} \quad G_1 H^{(0)} = \begin{pmatrix} x & x & x & x \\ 0 & x & x & x \\ 0 & x & x & x \\ 0 & 0 & x & x \end{pmatrix} \quad G_2 H^{(0)} = \begin{pmatrix} x & x & x & x \\ 0 & x & x & x \\ 0 & 0 & x & x \\ 0 & 0 & x & x \end{pmatrix}$$

我们可用一系列Givens矩阵来实现QR分解。假设T_k为上Hessenberg矩阵，我们有

$$G_{n-1} \cdots G_2 G_1 T_k = R_k \quad \Rightarrow \quad Q_k = (G_{n-1} \cdots G_2 G_1)^T$$

那么 $T_{k+1} = R_k Q_k$，因为无论是左乘 Q_k^T 还是右乘 Q_k，都是对同一个 2×2 矩阵进行操作，所以 T_{k+1} 和 T_k 一样，仍然具有上 Hessenberg 形式。然后我们可以用 Givens 矩阵对 T_{k+1} 进行 QR 分解，然后一直迭代下去。因为是在执行 QR 算法，上 Hessenberg 形式里的非对角元会随着迭代趋于 0。这里我们采用 Givens 变换的好处是我们并不需要算出 QR 分解中的矩阵 Q，实际计算中只需要记录所用到的 Givens 旋转变换的参数即可。

Givens 旋转变换是一个转动操作，对于矩阵有一对复共轭特征值的情形，就无法用 Givens 旋转变换来实现 QR 算法。这个时候，我们可以采用 Shifted QR 的方案。

Shifted QR

对于具有模非常接近的本征值的时候，QR迭代很可能收敛得很慢甚至根本不收敛于矩阵的实舒尔形式。为了克服这一点，我们可以引进具有单个shift的QR迭代。带位移的QR算法迭代公式如下

$$Q_kR_k=T_k-\mu I \quad \text{做QR分解}$$
$$T_{k+1}=R_kQ_k+\mu I$$

我们很容易验证T_{k+1}和T_k是正交相似的：

$$T_{k+1} = R_kQ_k + \mu I = Q_k^T(T_k - \mu I)Q_k + \mu I = Q_k^T T_k Q_k$$

我们选择$\mu \in R$使得 $|\lambda_1 - \mu| \geq |\lambda_2 - \mu| \geq \cdots |\lambda_n - \mu|$

这样一来变换过程中第k次迭代时的非对角元 $t_{j,j-1}^{(k)}$ 大致会按照 $|(\lambda_j - \mu)/(\lambda_{j-1} - \mu)|^k$ 的方式趋于零。因此，一个比较自然的选择是令

$$|\lambda_n - \mu| \ll |\lambda_i - \mu|, \quad i = 1, \cdots, n-1$$

那么矩阵元 $t_{n,n-1}^{(k)}$ 会快速地趋于零，所以 $t_{n,n}^{(k)}$ 最先收敛到本征值。在实际的应用中，人们一般会选择 $\mu = t_{n,n}^{(k)}$。当我们得到这个本征值以后，我们可以删除第 n 行、第 n 列，然后从剩下的子矩阵中求其他的本征值。

Shifted QR

对于实非对称矩阵，可能存在一对复共轭特征值，这个时候可以引入double shift的算法来改善QR迭代。我们考虑第n-1和第n行、第n-1和第n列的4个元素构成的一个2×2矩阵G包含一对复共轭特征值，分别为λ和 $\bar{\lambda}$。我们引入一个复参数μ，使得

$$|\lambda - \mu| \ll |\lambda_i - \mu|$$

构造QR分解

$$\begin{aligned} T - \mu I &= Q_1 R_1, \\ T_1 &= R_1 Q_1 + \mu I, \\ T_1 - \bar{\mu} I &= Q_2 R_2, \\ T_2 &= R_2 Q_2 + \bar{\mu} I \end{aligned}$$

这里Q_1和Q_2是幺正矩阵，我们有

$$T_1 = Q_1^{\dagger} T Q_1, \quad T_2 = Q_2^{\dagger} T_1 Q_2$$

从第二个和第三个式子，可以得到

$$R_1 Q_1 + (\mu - \bar{\mu}) I = Q_2 R_2$$

把这个式子左乘Q_1、右乘R_1，我们得到

$$Q_1 R_1 Q_1 R_1 + (\mu - \bar{\mu}) Q_1 R_1 = Q_1 R_1 (Q_1 R_1 + (\mu - \bar{\mu}) I) = (T - \mu I)(T - \bar{\mu} I) = Q_1 Q_2 R_2 R_1$$

这里$(T-\mu I)(T-\bar{\mu} I)$肯定是个实矩阵，因此$Q_1 Q_2 R_2 R_1$是对实矩阵的QR分解。因此，我们一定可以通过选择适当的幺正矩阵Q_1和Q_2，使得$Z=Q_1 Q_2$是实正交矩阵。

Shifted QR

$Z=Q_1Q_2$是实正交矩阵。于是，我们有

$$T_2 = (Q_1Q_2)^{\dagger}T(Q_1Q_2) = Z^TTZ$$

这里，我们可以看到，虽然单个的变换用的是幺正变换，但联合变换是正交变换。因为$|\lambda-\mu| \ll |\lambda_i-\mu|$，double shift能够帮助我们加速QR迭代的收敛。在实际操作中，我们选取

$$2\,\mathrm{Re}(\mu) = \mathrm{Tr}(G) = t^{(k)}_{n-1,n-1} + t^{(k)}_{n,n}, \quad |\mu|^2 = \det(G) = t^{(k)}_{n-1,n-1}t^{(k)}_{n,n} - t^{(k)}_{n-1,n}t^{(k)}_{n-1,n}$$

然后可以计算实矩阵的QR分解

$$(T-\mu I)(T-\bar{\mu}I) = T^2 - 2\,\mathrm{Re}(\mu)T + |\mu|^2 = ZR$$

得到Z以后，设定 $T_2 = Z^TTZ$

小结

我们讲了十大算法中的QR算法来解本征值。QR算法的基本步骤是QR分解，它是要把一个矩阵分解成实正交矩阵和上三角矩阵相乘的形式：$T_k=Q_kR_k$。但做完分解，我们并不能马上从R_k得到矩阵的本征值，我们让R_k右乘Q_k，试图构造变换$T_{k+1}=R_kQ_k=Q_k^TT_kQ_k$，那么$T_{k+1}$和$T_k$是相似变换，所以本征值完全一样，但这里有个问题，虽然Q_kR_k分解中得到的 R_k是上三角矩阵，但T_{k+1}因为右乘了Q_k，却并不是上三角矩阵。**数学上可以证明，T_{k+1}比T_k要更趋近于实舒尔型，于是我们经过$T_k\rightarrow T_{k+1}\rightarrow T_{k+2}\rightarrow\cdots$的迭代，最终$\lim_{k\rightarrow\infty}T_k$会给出我们想要的实舒尔形式。**

当然，纯做QR迭代，每次迭代都要花费O(n^3)的计算，是很耗时的。于是我们引进了Householder变换，将一般矩阵化成上Hessenberg矩阵的形式，这样有两个好处，一是对上Hessenberg矩阵做QR分解，每次迭代只需花费O(n^2)的计算量，其次，QR迭代的收敛速度也大为增加。这里Householder变换也是属于20世纪十大算法之一。有了上Hessenberg矩阵，我们可以引进Givens旋转矩阵来做QR分解。

除此之外，我们还介绍了Shifted QR的技术，包括单点shift，和double shift，其中double shift是用来处理复共轭本征对的情况，采用shift的技术，能使QR迭代的收敛性大为改善。

由矩阵的实舒尔形式计算其本征矢

设通过前面讲述的QR算法我们已得到矩阵的实舒尔形式，$Q^{T}AQ=T$，如果我们不考虑出现2×2包含复共轭根的情况（如果包含复共轭根，怎么从实舒尔形式得到本征矢量，留给同学们思考），那么T的对角元就是原先矩阵的本征值。**假定对于某个本征值λ，我们希望求其本征矢x。容易验明，若$Ax=\lambda x$，则令$y=Q^{T}x$，我们一定有$Ty=\lambda y$。**因此，为了求出x我们可以直接先求y，然后再以Q左乘之即可。我们令$\lambda=t_{kk}\in R$为A的某个单一本征值。那么矩阵T的形式一定为：

$$\begin{pmatrix} T_{11} & \nu & T_{13} \\ 0 & \lambda & \omega^{T} \\ 0 & 0 & T_{33} \end{pmatrix}$$

其中 $T_{11}\in R^{(k-1)\times(k-1)}$，$T_{33}\in R^{(n-k)\times(n-k)}$ 是两个上三角矩阵；$\nu\in R^{k-1}$，$w\in R^{n-k}$ 为两个矢量。按照假定，λ 是一个单一的本征值，因此它一定不会出现在 T_{11} 或者 T_{33} 的谱中。这意味着矩阵 $(T_{11}-\lambda I_{(k-1)\times(k-1)})$ 和 $[T_{33}-\lambda I_{(n-k)\times(n-k)}]$ 都是非奇异的上三角矩阵。我们假定 $y=(y_{k-1}^{T},y',y_{n-k}^{T})$，其中 $y_{k-1}\in C^{k-1}$，$y_{n-k}\in C^{n-k}$，于是本征方程 $(T-\lambda I)y=0$ 可以明确地写为

由矩阵的实舒尔形式计算其本征矢

$$\begin{cases} (T_{11} - \lambda I_{(k-1)\times(k-1)})y_{k-1} + y'\nu + T_{13}y_{n-k} = 0 \\ \omega^T y_{n-k} = 0 \\ [T_{33} - \lambda I_{(n-k)\times(n-k)}]y_{n-k} = 0 \end{cases}$$

由于 $(T_{11} - \lambda I_{(k-1)\times(k-1)})$ 和 $[T_{33} - \lambda I_{(n-k)\times(n-k)}]$ 都是非奇异地，因此上述方程可以“解出”(不失一般性，我们可以令 $y' = 1$) 如下的解:

$$y = \begin{pmatrix} -(T_{11} - \lambda I_{(k-1)\times(k-1)})^{-1}\nu \\ 1 \\ 0 \end{pmatrix}$$

最终，原来矩阵A的本征矢x可以由x=Qy给出。

奇异值分解

与本征值有关的，是所谓矩阵的奇异值。奇异值分解是一项应用十分广泛的计算技术。对于一个**m×n的复矩阵A∈C^{m×n}**，一定存在**两个幺正矩阵**U∈C^{m×m}和V∈C^{n×n}，它们能够将A"对角化"：

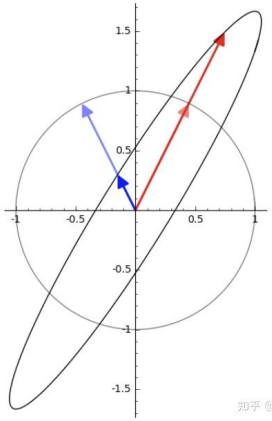
$$U^{\dagger}AV = \Sigma = \mathrm{diag}(\sigma_1, \cdots, \sigma_p) \in C^{m\times n}, \quad p = \min(m,n)$$

其中$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_p \geq 0$称为矩阵A的奇异值(singular values)而上式则称为矩阵A的奇异值分解(singular value decomposition, SVD)。**矩阵的那些非零的奇异值实际上是矩阵A†A的本征值的根号：**

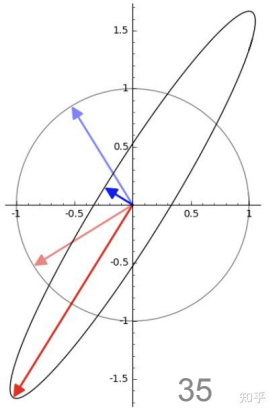
$$\sigma_i(A) = \sqrt{\lambda_i(A^{\dagger}A)}$$

A即使为方阵，也不一定有本征值，或者本征值不一定为实数。A†A则必有实数本征值

This concept was introduced by Erhard Schmidt in 1907. Schmidt called singular values "eigenvalues" at that time. The name "singular value" was first quoted by Smithies in 1937.



特征向量描述的是矩阵的方向**不变作用**(***invariant*** *action*)的向量；**奇异向量**描述的是矩阵**最大作用**(***maximum*** action)的方向向量。



奇异值分解

一个矩阵的奇异值分解的计算一般运用Golub-Kahan-Reinsch算法。不失一般性，我们假定A∈R^{m×n}且m≥n，否则我们就计算A^T的奇异值分解。奇异值分解分为以下步骤

首先，矩阵A先被变换为如下形式

$$\bar{U}^T A \bar{V} = \begin{pmatrix} B \\ 0 \end{pmatrix}$$

其中 $\bar{U}$ 和 $\bar{V}$ 是两个正交矩阵，$B \in R^{n\times n}$ 是一个上双对角 (upper bidiagonal) 矩阵。矩阵 $\bar{U}$ 和 $\bar{V}$ 由以下步骤，通过 $n+m-3$ 个 Householder 矩阵 $\bar{U}_1, \cdots, \bar{U}_{m-1}$ 和 $\bar{V}_1, \cdots, \bar{V}_{n-2}$ 依次产生。首先我们将 $(\bar{U}_1)^T$ 左乘到 A 上：$A^{(1)} = (\bar{U}_1)^T A$，使 $A^{(1)}$ 的第一列的第二个矩阵元以下都为零；然后将 $\bar{V}_1$ 右乘到 $A^{(1)}$ 上得到 $A^{(2)} = A^{(1)}\bar{V}_1$，使得 $A^{(2)}$ 的第一行的第三个矩阵元右边的全部为零，同时不破坏第一列的那些已经化为零的矩阵元

$$A^{(1)} = \bar{U}_1^T A = \begin{pmatrix} x & x & x & x \\ 0 & x & x & x \\ 0 & x & x & x \\ 0 & x & x & x \\ 0 & x & x & x \end{pmatrix} \Rightarrow A^{(2)} = A^{(1)}\bar{V}_1 = \begin{pmatrix} x & x & 0 & 0 \\ 0 & x & x & x \\ 0 & x & x & x \\ 0 & x & x & x \\ 0 & x & x & x \end{pmatrix}$$

奇异值分解

第二步，我们可以利用QR迭代将上双对角矩阵B对角化，即我们会获得两个正交矩阵W和Z使得

$$W^T B Z = \Sigma = \mathrm{diag}(\sigma_1, \cdots, \sigma_n)$$

其中σ_i，$i=1,\cdots,n$就是矩阵A的奇异值。于是矩阵A的奇异值分解可以表达为，

$$U^T A V = \begin{pmatrix} \Sigma \\ 0 \end{pmatrix}$$

如果A是复矩阵，那么在推导过程中的U，V矩阵需要替换为相应的幺正矩阵。于是我们就得到

$$U^\dagger A V = \Sigma$$

这个结论反过来写，就是

$$A = U \Sigma V^\dagger = \sum_{k=1}^{p} \sigma_k U_k (V^\dagger)_k$$

$$A_{ij} = \sum_{k=1}^{p} \sigma_k (u_i)_k (v_j)_k^*$$

其中u_i表示矩阵U的第i行矢量，而v_j则表示矩阵V的第j行矢量。

奇异值分解

事实上奇异分解具有很广泛的应用，其中一种应用就是图片的压缩存储。假如说我们有一张像素为1024×512的照片，作为数据储存的时候实际上对应于一个1024×512的矩阵。完整存下来需要保存1024×512=5×10^5的数据，如果我们仅保存按奇异值从大到小排序的前50项，那么总共需要存储的数据为(1+1024+512)×50=7.5×10^4，总存储量大概为原矩阵的15%，但由于大奇异值的信息都已包括在新的矩阵中，实际上我们能还原出和原图差别不大的图片。**事实上正因为大奇异值往往对应着矩阵中隐含的重要信息，我们不仅可以把它应用在数据压缩上，也可以对图片去噪声。我们有理由相信那些较小的奇异值是由噪声引起的，我们可以强行令它们为0。**

奇异值分解的强大之处在于，任何复矩阵都可以这样进行分解！正因为如此，它被广泛地应用在各个领域，其中除了我们熟悉的科技的领域，还包括一些非科技的领域。更多例子可以参见刘川老师讲义。

对称矩阵的算法-Jacobi 算法

如果矩阵是**实对称的方矩阵**，那么我们可以利用下面的一些算法来计算其本征值和本征矢量。最为直接的方法就是Jacobi算法，它直接利用Givens转动将矩阵的非对角元变换为零。

设 $A^{(0)} = A \in R^{n\times n}$ 为一实对称矩阵，对一对不相等的指标 p 和 q 满足 $1 \le p < q \le n$，我们将 Givens 矩阵 $G(p,q,\theta) = G_{pq}$ 作用于 $A^{(k-1)}$ 得到新的 $A^{(k)}$：

$$A^{(k)} = G_{pq}^T A^{(k-1)} G_{pq}$$

我们可以选择θ使得转动以后的A^(k)矩阵的矩阵元满足

$$a_{ij} = 0, \quad \text{如果}: (i,j) = (p,q)$$

由于Givens矩阵的结构，上述关系等价于

$$A^{(k)} = \begin{pmatrix} c & s \\ -s & c \end{pmatrix}^T \begin{pmatrix} a_{pp} & a_{pq} \\ a_{pq} & a_{qq} \end{pmatrix} \begin{pmatrix} c & s \\ -s & c \end{pmatrix}$$

我们发现要使非对角矩阵元为0，t≡s/c需满足一个二次方程：

$$t^2 + 2\eta t - 1 = 0, \quad \eta = \frac{a_{qq} - a_{pp}}{2a_{pq}}$$

对称矩阵的算法-Jacobi 算法

如果 $\eta \geq 0$，我们取 $t = 1/(\eta + \sqrt{1+\eta^2})$；如果 $\eta < 0$，我们取 $t = -1/(-\eta + \sqrt{1+\eta^2})$。这两个根写成分母的形式是为了避免两个相近的数相消，然后令

$$c = \frac{1}{\sqrt{1+\eta^2}}, \quad s = ct$$

为了考察Jacobi迭代收敛的速度我们定义矩阵的一个特征函数

$$\Psi(M) = \left(\sum_{i \neq j} m_{ij}^2 \right)^{1/2}$$

前面的Jacobi迭代给出：$\Psi(A^{(k)})^2 = \Psi(A^{(k-1)})^2 - 2\left(a_{pq}^{(k-1)}\right)^2 < \Psi(A^{(k-1)})^2$

因此，在进行Jacobi迭代时，**最有效的是从其模最大的非对角元开始**。但是实际上也可以就对每一个非对角元都做一遍。**Jacobi算法仅仅适用于实对称矩阵以及它的复推广厄米矩阵。而且它的计算代价还是比较大的，基本上是O(n^3)。但是它有一个优点就是超级稳定。**原因就在于上面讨论的严格成立的估计。因此它特别适合于体量比较小的(典型的n≤O(100))实对称矩阵或者复厄米矩阵。如果我们不仅仅要求本征值还要求本征矢，这往往会增加大约50%的计算量。

Sturm 序列

所谓的**Sturm序列适用于计算一个实的三对角对称矩阵的本征值问题**。我们假设三对角矩阵T的对角元为d_i，$i=1,2,\cdots,n$；两个副对角线上的元素为b_i，$i=1,2,\cdots,n-1$。不失一般性我们假设所有的$b_i!=0$，否则经过重新排列问题可以化为一个更小的三对角矩阵的本征值问题。一个重要的观察是，**这样一个三对角矩阵的特征多项式可以通过迭代的方法给出**。令$p_0(x)=1$和$p_1(x)=d_1-x$，

$$p_i(x) = (d_i - x)p_{i-1}(x) - b_{i-1}^2 p_{i-2}(x), \quad i = 2, \cdots, n$$

我们在讨论正交多项式的章节里面曾经讨论过相关的迭代。容易验证 $p_n(x)$恰好就是T的特征多项式。不仅如此，事实上$p_i(x)=\det(T_i-xI_{i\times i})$是$T_i$的特征多项式，其中$T_i$就是由矩阵T的前i行和i列构成的矩阵(也称为主子矩阵)。上面公式给出的多项式序列称为Sturm序列。

Sturm 序列

Sturm序列满足一系列重要的性质。例如$p_i(x)$的根(也就是T_i的本征值)与$p_{i-1}(x)$的根进行排序，可以证明，这两组根一定会交错地出现。另外一个是我们**很容易确定小于某个数值的实根的个数**，这就是下面的结论：对于上述的Sturm序列$\{p_i(x): i=0,1,\cdots,n\}$，我们选取任意的$\mu\in R$并且构造如下的序列：

$$S_\mu = \{p_0(\mu), p_1(\mu), \cdots, p_n(\mu)\},$$

那么上述实数序列中从前往后数的过程中，其数值变号的次数$s(\mu)$就是$p_n(x)=0$所具有的严格小于μ的实根的数目。这里我们约定：**如果$p_i(\mu)=0$，我们认为它是与前面的邻居$p_{i-1}(\mu)$不同号的(即算做一次变号)**。作为一个例子，考虑4×4的三对角矩阵

$$\begin{pmatrix} 2 & -1 & & \\ -1 & 2 & -1 & \\ & -1 & 2 & -1 \\ & & -1 & 2 \end{pmatrix}$$

这个矩阵的四个本征值由小到大依次是$\{0.38, 1.38, 2.62, 3.62\}$。如果我们计算$\mu=3$时各个多项式的值我们有：$p_0(3), p_1(3), p_2(3), p_3(3), p_4(3)=1,-1,0,1,-1$，从左到右数过去一共变号三次(按照定理中的约定，其中的0也算一次)，因此$p4(x)=0$的根中有3个小于3。

Sturm 序列

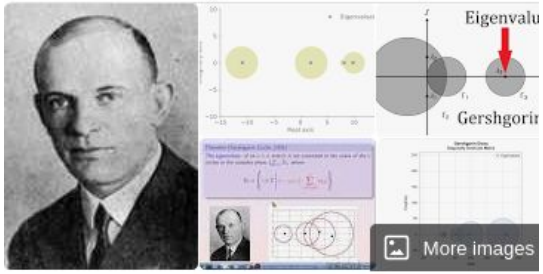
由于所有实对称矩阵(或者厄米矩阵)的根都是实根，因此如果我们仅仅希望寻找它的本征值的话，我们完全可以利用前一章中所提及的各种求根的方法，**其中最为稳定的当属对分法。当然，这需要首先选定一个寻找的区间**，这时我们可以利用**Gershgorin圆盘定理**。定理描述如下：对任意的$A \in C^{n\times n}$，它的谱记为$\sigma(A)$。我们构建复平面的n个圆(盘)：

$$R_i = \left\{ z \in C : |z - a_{ii}| \leq \sum_{j=1, j\neq i}^{n} |a_{ij}| \right\}$$

其中$i=1,2,\cdots,n$。它们称为Gershgorin圆盘。那么矩阵A的本征值一定落在这些圆盘的并集之中

$$\sigma(A) \subseteq S_R = \cup_{i=1}^{n} R_i$$

这个称为Gershgorin圆盘定理。这个定理对于实对称矩阵(或者厄米矩阵)特别有效，因为它们的本征值一定都位于实轴上。



Semyon Aranovich Gershgorin

Mathematician

Semyon Aronovich Gershgorin was a Soviet mathematician. He began as a student at the Petrograd Technological Institute in 1923, became a Professor in 1930, and was given an appointment at the Leningrad Mechanical Engineering Institute in the same year. His contributions include the Gershgorin circle theorem. Wikipedia

Born: 24 August 1901, Pruzhany, Belarus

Died: 30 May 1933, Saint Petersburg, Russia

Education: Sankt-Peterburgskiy Politekhnicheskiy Universitet Petra Velikogo (1923)

Sturm 序列

根据这个圆盘定理，我们发现，对于实对称矩阵来说，我们发现搜寻其实根区间的下限α和上限β可以分别取为，

$$\alpha = \min_{1\le i\le n}[d_i - (|b_{i-1}| + |b_i|)], \quad \beta = \max_{1\le i\le n}[d_i + (|b_{i-1}| + |b_i|)]$$

其中我们约定了$b_0=b_n=0$。

这个算法的设计如下：我们假定 $a^{(0)}=\alpha$，$b^{(0)}=\beta$，然后取 $c^{(0)}$ 为中间值 $c^{(0)}=(\alpha+\beta)/2$，然后把 $c^{(0)}$ 代入 Sturm 序列，计算符号变化的次数 $s(c^{(0)})$。如果我们希望计算第 i 个本征值 λ_i，那么我们将 $s(c^{(0)})$ 和 $n-i$ 比较，如果 $s(c^{(0)})>n-i$，就意味着 $c^{(0)}>\lambda_i$，这个时候，我们让 $b^{(1)}=c^{(0)}$，减少搜索区间的上限；反之，则让 $a^{(1)}=c^{(0)}$。经过 r 次迭代以后，$c^{(r)}=\frac{a^{(r)}+b^{(r)}}{2}$ 就给出了 λ_i 的近似值，与真实值相差不超过 $(|\alpha|+|\beta|)\cdot 2^{-(r+1)}$。

稀疏矩阵的本征值问题：Lanczos 方法

如果我们希望计算的矩阵是一个**庞大的稀疏矩阵**，那么前面所列举的方法并不能有效地进行操作。这时，如果我们仅仅需要求解矩阵的部分而不是全部本征对，那么我们可以利用Lanczos方法来进行计算。
我们假设矩阵A∈R^{n×n}是对称的实稀疏矩阵。我们给它的本征值排一下序

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$$

当n很大时，我们用Lanczos方法去近似得到它的最大和最小本征值。它的基本思路是这样的，我们可以构造Rayleigh比值

$$r(x) = \frac{x^T A x}{x^T x}, \quad \text{for } x \in R^n$$

对于矩阵A的最大和最小本征值，我们有

$$\lambda_1(A) = \max_{x \in R^n, x \neq 0} r(x), \quad \lambda_n(A) = \min_{x \in R^n, x \neq 0} r(x)$$

也就是说，我们要搜索适当的x，使得r(x)取得全局最大/小值。

稀疏矩阵的本征值问题：Lanczos 方法

我们首先想到的是搜索方向沿着梯度$\nabla r(x)$来得到最大值，和$-\nabla r(x)$来得到最小值。之前我们在共轭梯度法的章节里面曾经讲过，这样的搜索其实并不是最佳搜索，我们应该采用的是**共轭梯度法**，**在Krylov子空间里面进行搜索，是更优化的搜索方式**，随着Krylov子空间维度的增加，我们会越来越接近我们想要的全局最大和最小值。

假设我们从任意矢量v出发，考虑矢量空间：

$$K_m(A;v) = \mathrm{span}\{v, Av, \cdots, A^{m-1}v\},$$

我们称之为矢量v和矩阵A的m阶的Krylov子空间。

对于一个n×n阶的稀疏矩阵A，由于它的非零矩阵元只有O(n)个，因此对于给定的矢量v，**计算矢量Av仅仅需要O(n)的浮点数计算量**而不是稠密矩阵的O(n^2)。同时储存Av也仅仅需要O(n)的内存。这就是为什么我们对于大型稀疏矩阵需要利用迭代的方法，以及为何我们对它的Krylov子空间感兴趣。我们一般需要寻找最大和最小本征值的近似解，因此在这类计算中我们一般假设**m≪n**。

稀疏矩阵的本征值问题：Lanczos 方法

具体构造Krylov子空间的方法，就是所谓的Lanczos迭代。算法如下

$v_0 = 0, \quad v_1 = v, \quad \beta_1 = 0$

For $k = 1, 2, \cdots, m-1$, do

$\omega_k = Av_k$

$\alpha_k = \omega_k^T v_k$ 计算内积

$\omega_k = \omega_k - \alpha_k v_k - \beta_k v_{k-1}$ 更新 ω_k

$\beta_{k+1} = ||\omega_k||_2, \quad v_{k+1} = \dfrac{\omega_k}{||\omega_k||_2}$ 归一化 ω_k

End

我们不难看出，如果$v_1, \cdots, v_k$张成了k阶Krylov子空间，那么再加上v_{k+1}，就张成了k+1阶Krylov子空间，另外，如果有$v_k^T v_{k-1}=0$，我们马上能得到$v_{k+1}^T v_k=0$，因此**经过上述的Lanczos迭代，我们获得了一组$K_m(A;v)$中的正交归一的基矢，它们构成了一个长方正交矩阵**（因为一般来说m≪n）：

$$V_m = (v_1, v_2, \cdots, v_m)$$

稀疏矩阵的本征值问题：Lanczos 方法

我们还可以证明 $H_m = V_m^T A V_m$ 是个**m×m阶三对角矩阵**。这是因为

$$(H_m)_{\alpha\beta} = v_\alpha^T A v_\beta$$

如果α>β+1，那么Avβ由β+1阶的Krylov子空间中的基矢线性组合构成，根据vk矢量的构造方式，vα肯定与Avβ正交，因此当α>β+1时，我们有(Hm)αβ = 0。反之亦然。

$$\begin{bmatrix} \alpha_1 & \beta_2 & 0 & & & 0 \\ \beta_2 & \alpha_2 & \beta_3 & & & \\ 0 & \beta_3 & \alpha_3 & \ddots & & \\ & \ddots & \ddots & \beta_{m-1} & & \\ & & \beta_{m-1} & \alpha_{m-1} & \beta_m & \\ 0 & & & \beta_m & \alpha_m & \end{bmatrix}$$

由于Hm是一个比原矩阵A小很多的矩阵，因此处理它的本征值问题要简单的多。而且由于Hm的三对角形式，我们可以运用前一小节提及的Sturm序列求解。需要注意的是，由**于舍入误差的传递，一般的Lanczos算法当m变得比较大时，往往会遇到稳定性的问题**。也就是说，原先做好的正交归一化步骤有可能由于舍入误差的影响而不再成立。这时可能需要重新进行Gram-Schmidt的操作。

Lanczos 方法：另一种理解

$AV_j \in \mathrm{Span}\{V_1, V_2, \ldots V_j, V_{j+1}\}$

Then

$\mathbf{V_i^T.A.V_j=0 \text{ for } i>j+1}$

$\mathbf{V_i^T.A.V_j=(A.V_j)^TV_j=0 \text{ for } j>i+1}$

So $T=V^T.A.V$ will be Tridiagonal matrix.

$AV=VT \quad V=[v1,v2,\ldots]$

$AV_k=\beta_{k-1}V_{k-1}+\alpha_kV_k+\beta_kV_{k+1}$

$\rightarrow \quad \alpha_k=V_k^T.A.V_k$

$\beta_kV_{k+1}=AV_k-\alpha_kV_k-\beta_{k-1}V_{k-1}$

$\rightarrow |\beta_k|=||AV_k-\alpha_kV_k-\beta_{k-1}V_{k-1}||$

稀疏矩阵的本征值问题：Lanczos 方法

尽管存在一定的局限性，Lanczos方法有两个显著的优点：

- 一是它在变换的过程中保证了一直保持了稀疏矩阵的形式不变，这对于大型矩阵n很大的时候，尤为重要；
- 另外，它能够比较快地趋于矩阵A本征值的极值。

我们之前提到过可以用这个方法来计算最小的若干个本征值和本征矢量，它能够帮助我们加速CG算法解线性方程组。