Décomposition QR

En algèbre linéaire, la **décomposition QR** (appelée aussi, décomposition QU) d'une matrice A est une décomposition de la forme

$$A = QR$$

où Q est une matrice orthogonale ($Q^T Q = I$), et R une matrice triangulaire supérieure.

Ce type de décomposition est souvent utilisée pour le calcul de solutions de systèmes linéaires non carrés, notamment pour déterminer la pseudo-inverse d'une matrice.

Sommaire

Extensions

Il est possible de calculer une décomposition RQ d'une matrice, ou même des décompositions QL et LQ, où la matrice L est triangulaire inférieure.

Méthodes

Il existe plusieurs méthodes pour réaliser cette décomposition :

- la méthode de Householder où Q est obtenue par produits successifs de matrices orthogonales élémentaires
- la méthode de Givens **(en)** où Q est obtenue par produits successifs de matrices de rotation plane
- la méthode de Gram-Schmidt

Chacune d'entre elles a ses avantages et ses inconvénients. (La décomposition QR n'étant pas unique, les différentes méthodes produiront des résultats différents).

Méthode de Householder

Soient **x** un vecteur colonne arbitraire de dimension *m* et α= ± ||*x*||, où || || désigne la norme euclidienne. Pour des raisons de stabilité du calcul, α doit de plus être du signe du premier élément de **x**.

Soit $\mathbf{e}_1$ le vecteur $(1,0,..., 0)^T$, et définissons, si *x* n'est pas colinéaire à $\mathbf{e}_1$:

$$\mathbf{u} = \mathbf{x} - \alpha\mathbf{e}_1,$$
$$\mathbf{v} = \frac{\mathbf{u}}{||\mathbf{u}||},$$
$$Q = I - 2\mathbf{v}\mathbf{v}^T.$$

Q est la matrice de Householder ou matrice orthogonale élémentaire et

$$Qx = (\alpha\ , 0, \cdots, 0)^T.$$

(Si **x** est colinéaire à $\mathbf{e}_1$, on a le même résultat en prenant pour *Q* la matrice identité.)

Nous pouvons utiliser ces propriétés pour transformer une matrice *A* de dimension *m*n* en une matrice triangulaire supérieure. Tout d'abord, on multiplie *A* par la matrice de Householder Q_1 en ayant pris le soin de choisir pour **x** la première colonne de *A* . Le résultat est une matrice *QA* avec des zéros dans la première colonne excepté du premier élément qui vaudra α.

$$Q_1A = \begin{pmatrix} \alpha_1 & \star & \dots & \star \\ 0 & & & \\ \vdots & & A' & \\ 0 & & & \end{pmatrix}$$

Ceci doit être réitéré pour *A'* qui va être multiplié par Q'_2 (Q'_2 est plus petite que Q_1). Si toutefois, vous souhaitiez utiliser Q_1A plutôt que *A'*, vous deviez remplir la matrice de Householder avec des 1 dans le coin supérieur gauche :

$$Q_k = \begin{pmatrix} I_{k-1} & 0 \\ 0 & Q'_k \end{pmatrix}$$

Après t itérations, $t = \min(m-1, n)$,

$$R = Q_t \cdots Q_2 Q_1 A$$

est une matrice triangulaire supérieure. Si $Q = Q_1^T Q_2^T \cdots Q_t^T$ alors $A = QR$ est la décomposition QR de A. De plus, par construction les matrices Q_k sont non seulement orthogonales mais aussi symétriques, donc $Q = Q_1 Q_2 \cdots Q_t$.

Exemple

Calculons la décomposition QR de

$$A = \begin{pmatrix} 12 & -51 & 4 \\ 6 & 167 & -68 \\ -4 & 24 & -41 \end{pmatrix}$$

On choisit donc le vecteur $a_1 = (12{,}6, -4)^T$. On a donc $\|a_1\| = \sqrt{12^2 + 6^2 + (-4)^2} = 14$. Ce qui nous conduit à écrire $\|a_1\| e_1 = (14, 0, 0)^T$.

Le calcul nous amène à $u = 2(-1, 3, -2)^T$ et $v = \frac{1}{14^{\frac{1}{2}}}(-1, 3, -2)^T$. La première matrice de Householder vaut

$$\begin{aligned} Q_1 &= I - \frac{2}{14} \begin{pmatrix} -1 \\ 3 \\ -2 \end{pmatrix} \begin{pmatrix} -1 & 3 & -2 \end{pmatrix} \\ &= I - \frac{1}{7} \begin{pmatrix} 1 & -3 & 2 \\ -3 & 9 & -6 \\ 2 & -6 & 4 \end{pmatrix} = \begin{pmatrix} 6/7 & 3/7 & -2/7 \\ 3/7 & -2/7 & 6/7 \\ -2/7 & 6/7 & 3/7 \end{pmatrix}. \end{aligned}$$

Observons que

$$Q_1 A = \begin{pmatrix} 14 & 21 & -14 \\ 0 & -49 & -14 \\ 0 & 168 & -77 \end{pmatrix}.$$

Nous avons maintenant sous la diagonale uniquement des zéros dans la 1re colonne.

Pour réitérer le processus, on prend la sous matrice principale

$$A' = M_{11} = \begin{pmatrix} -49 & -14 \\ 168 & -77 \end{pmatrix}$$

Par la même méthode, on obtient

$$\alpha_2 = -\sqrt{49^2 + 168^2} = -175, \quad u_2 = (126, 168)^T, \quad v_2 = (3/5, 4/5)^T, \quad Q_2' = \Big($$

La 2e matrice de Householder est donc

$$Q_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 7/25 & -24/25 \\ 0 & -24/25 & -7/25 \end{pmatrix}.$$

Finalement, on obtient

$$Q = Q_1 Q_2 = \begin{pmatrix} 6/7 & 69/175 & -58/175 \\ 3/7 & -158/175 & 6/175 \\ -2/7 & -6/35 & -33/35 \end{pmatrix}$$

$$R = Q_2 Q_1 A = Q^T A = \begin{pmatrix} 14 & 21 & -14 \\ 0 & -175 & 70 \\ 0 & 0 & 35 \end{pmatrix}.$$

La matrice Q est orthogonale et R est triangulaire supérieure, par conséquent, on obtient la décomposition $A = QR$.

Coût et avantages

Le coût de cette méthode pour une matrice $n*n$ est en : $\frac{4}{3} \times n^3$ Ce coût est relativement élevé (la méthode de Cholesky, pour les matrices symétriques définies positives est en $\frac{1}{3} \times n^3$). Cependant, la méthode de Householder présente l'avantage considérable d'être beaucoup plus stable numériquement, en limitant les divisions par des nombres petits. La méthode de Givens, malgré un coût encore supérieur à celui-ci, offrira encore davantage de stabilité.

Méthode de Schmidt

On considère le procédé de Gram-Schmidt appliqué aux colonnes de la matrice $A = [\mathbf{a}_1, \cdots, \mathbf{a}_n]$, muni du produit scalaire $\langle \mathbf{v}, \mathbf{w} \rangle = \mathbf{v}^\top \mathbf{w}$ (ou $\langle \mathbf{v}, \mathbf{w} \rangle = \mathbf{v}^* \mathbf{w}$ pour le cas complexe).

On définit la projection :

$$\Pi_{\mathbf{e}} \mathbf{a} = \frac{\langle \mathbf{e}, \mathbf{a} \rangle}{\langle \mathbf{e}, \mathbf{e} \rangle} \mathbf{e}$$

puis les vecteurs :

$$\begin{aligned}
\mathbf{u}_1 &= \mathbf{a}_1, & \mathbf{e}_1 &= \frac{\mathbf{u}_1}{\|\mathbf{u}_1\|} \\
\mathbf{u}_2 &= \mathbf{a}_2 - \Pi_{\mathbf{e}_1} \mathbf{a}_2, & \mathbf{e}_2 &= \frac{\mathbf{u}_2}{\|\mathbf{u}_2\|} \\
\mathbf{u}_3 &= \mathbf{a}_3 - \Pi_{\mathbf{e}_1} \mathbf{a}_3 - \Pi_{\mathbf{e}_2} \mathbf{a}_3, & \mathbf{e}_3 &= \frac{\mathbf{u}_3}{\|\mathbf{u}_3\|} \\
&\vdots & &\vdots \\
\mathbf{u}_k &= \mathbf{a}_k - \sum_{j=1}^{k-1} \Pi_{\mathbf{e}_j} \mathbf{a}_k, & \mathbf{e}_k &= \frac{\mathbf{u}_k}{\|\mathbf{u}_k\|}.
\end{aligned}$$

On réarrange ensuite les équations de sorte que les $\mathbf{a}_i$ soient à gauche, en utilisant le fait que les $\mathbf{e}_i$ sont des vecteurs unitaires :

$$\begin{aligned}
\mathbf{a}_1 &= \langle \mathbf{e}_1, \mathbf{a}_1 \rangle \mathbf{e}_1 \\
\mathbf{a}_2 &= \langle \mathbf{e}_1, \mathbf{a}_2 \rangle \mathbf{e}_1 + \langle \mathbf{e}_2, \mathbf{a}_2 \rangle \mathbf{e}_2 \\
\mathbf{a}_3 &= \langle \mathbf{e}_1, \mathbf{a}_3 \rangle \mathbf{e}_1 + \langle \mathbf{e}_2, \mathbf{a}_3 \rangle \mathbf{e}_2 + \langle \mathbf{e}_3, \mathbf{a}_3 \rangle \mathbf{e}_3 \\
&\vdots \\
\mathbf{a}_k &= \sum_{j=1}^{k} \langle \mathbf{e}_j, \mathbf{a}_k \rangle \mathbf{e}_j
\end{aligned}$$

où $\langle \mathbf{e}_i, \mathbf{a}_i \rangle = \|\mathbf{u}_i\|$. Ceci s'écrit matriciellement :

$$A = QR$$

avec

$$Q = [\mathbf{e}_1, \cdots, \mathbf{e}_n] \qquad \text{et} \qquad R = \begin{pmatrix} \langle \mathbf{e}_1, \mathbf{a}_1 \rangle & \langle \mathbf{e}_1, \mathbf{a}_2 \rangle & \langle \mathbf{e}_1, \mathbf{a}_3 \rangle & \ldots \\ 0 & \langle \mathbf{e}_2, \mathbf{a}_2 \rangle & \langle \mathbf{e}_2, \mathbf{a}_3 \rangle & \ldots \\ 0 & 0 & \langle \mathbf{e}_3, \mathbf{a}_3 \rangle & \ldots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix}.$$

Exemple

On reprend la matrice de l'exemple

$$A = \begin{pmatrix} 12 & -51 & 4 \\ 6 & 167 & -68 \\ -4 & 24 & -41 \end{pmatrix}.$$

Rappelons qu'une matrice orthogonale Q vérifie

$$Q^T\, Q = I.$$

On peut alors calculer Q par les moyens de Gram-Schmidt comme suit :

$$U = \begin{pmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \mathbf{u}_3 \end{pmatrix} = \begin{pmatrix} 12 & -69 & -58/5 \\ 6 & 158 & 6/5 \\ -4 & 30 & -33 \end{pmatrix};$$

$$Q = \begin{pmatrix} \frac{\mathbf{u}_1}{\|\mathbf{u}_1\|} & \frac{\mathbf{u}_2}{\|\mathbf{u}_2\|} & \frac{\mathbf{u}_3}{\|\mathbf{u}_3\|} \end{pmatrix} = \begin{pmatrix} 6/7 & -69/175 & -58/175 \\ 3/7 & 158/175 & 6/175 \\ -2/7 & 6/35 & -33/35 \end{pmatrix}.$$

Dans ce cas, on a :

$$Q^T A = Q^T Q\, R = R;$$

$$R = Q^T A = \begin{pmatrix} 14 & 21 & -14 \\ 0 & 175 & -70 \\ 0 & 0 & 35 \end{pmatrix}.$$

Relation avec la décomposition RQ

La décomposition RQ transforme une matrice *A* en produit d'une matrice triangulaire supérieure *R* et une matrice orthogonale *Q*. La seule différence avec la décomposition QR est l'ordre de ces matrices.

La décomposition QR est l'application du procédé de Gram-Schmidt sur les colonnes de A, en partant de la première colonne ; la décomposition RQ est l'application du procédé de Gram-Schmidt sur les lignes de A, en partant de la dernière ligne.

Méthode de Givens

Dans cette méthode, la matrice Q utilise des rotations de Givens **(en)**. Chaque rotation annule un élément de la partie triangulaire inférieure stricte de la matrice, construisant la matrice R, tandis que la concaténation des rotations engendre la matrice Q.

Dans la pratique, les rotations de Givens ne sont pas effectivement assurées par la construction d'une matrice pleine et une multiplication matricielle. Une procédure de rotation de Givens est utilisé à la place qui est l'équivalent de la multiplication par une matrice de Givens creuse, sans efforts supplémentaires de la manipulation des éléments non nuls. La procédure de rotation de Givens est utile dans des situations où seul un nombre relativement restreint hors éléments diagonaux doivent être remis à zéro, et est plus facilement parallélisée que les transformations de Householder.

Exemple

Reprenons le même exemple

$$A = \begin{pmatrix} 12 & -51 & 4 \\ 6 & 167 & -68 \\ -4 & 24 & -41 \end{pmatrix}.$$

On doit d'abord construire une matrice de rotation qui annulera l'élément le plus bas de la colonne de gauche, $\mathbf{a}_{31} = -4$, qu'on construit par une méthode de rotation de Givens. On appelle cette matrice G_1. On va d'abord faire une rotation du vecteur $(6, -4)$, pour le ramener sur l'axe *X*. Ce vecteur forme un angle $\theta = \arctan\left(\frac{-4}{6}\right)$. La matrice G_1 est donc donnée par :

$$G_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & \sin(\theta) \\ 0 & -\sin(\theta) & \cos(\theta) \end{pmatrix} \approx \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0,83205 & -0,55470 \\ 0 & 0,55470 & 0,83205 \end{pmatrix}.$$

Le produit $G_1 A$ annule le coefficient $\mathbf{a}_{31}$:

$$G_1 A \approx \begin{pmatrix} 12 & -51 & 4 \\ 7,21110 & 125,6396 & -33,83671 \\ 0 & 112,6041 & -71,83368 \end{pmatrix}.$$

Par suite, on construit des matrices de Givens G_2 et G_3, qui vont respectivement annuler a_{21} et a_{32}, engendrant la matrice R. La matrice orthogonale Q^T est formée de la concaténation de toutes les matrices de Givens créées $Q^T = G_3 G_2 G_1$.

Liens externes

- **(en)** Cet article est partiellement ou en totalité issu de l'article de Wikipédia en anglais intitulé « QR decomposition (//en.wikipedia.org/wiki/QR_decomposition?oldid=378686082) » (voir la liste des auteurs (//en.wikipedia.org/wiki/QR_decomposition?action=history))
- Patrick Lascaux et Raymond Théodor, *Analyse numérique matricielle appliquée à l'art de l'ingénieur*, tome 1 : *Méthodes directes* [détail des éditions]

Articles connexes

- Décomposition LU
- Factorisation de Cholesky

QR decomposition

From Wikipedia, the free encyclopedia

In linear algebra, a **QR decomposition** (also called a **QR factorization**) of a matrix is a decomposition of a matrix A into a product $A = QR$ of an orthogonal matrix Q and an upper triangular matrix R. QR decomposition is often used to solve the linear least squares problem, and is the basis for a particular eigenvalue algorithm, the QR algorithm.

If A has n linearly independent columns, then the first n columns of Q form an orthonormal basis for the column space of A. More specifically, the first k columns of Q form an orthonormal basis for the span of the first k columns of A for any $1 \le k \le n$.[1] The fact that any column k of A only depends on the first k columns of Q is responsible for the triangular form of R.[1]

Contents

Definitions

Square matrix

Any real square matrix A may be decomposed as

$$A = QR,$$

where Q is an orthogonal matrix (its columns are orthogonal unit vectors meaning $Q^{\mathrm{T}}Q = I$) and R is an upper triangular matrix (also called right triangular matrix). This generalizes to a complex square matrix A and a unitary matrix Q (where $Q^*Q = I$). If A is invertible, then the factorization is unique if we require that the diagonal elements of R are positive.

Rectangular matrix

More generally, we can factor a complex $m \times n$ matrix A, with $m \geq n$, as the product of an $m \times m$ unitary matrix Q and an $m \times n$ upper triangular matrix R. As the bottom $(m-n)$ rows of an $m \times n$ upper triangular matrix consist entirely of zeroes, it is often useful to partition R, or both R and Q:

$$A = QR = Q \begin{bmatrix} R_1 \\ 0 \end{bmatrix} = [Q_1, Q_2] \begin{bmatrix} R_1 \\ 0 \end{bmatrix} = Q_1 R_1,$$

where R_1 is an $n \times n$ upper triangular matrix, Q_1 is $m \times n$, Q_2 is $m \times (m - n)$, and Q_1 and Q_2 both have orthogonal columns.

Golub & Van Loan (1996, §5.2) call $Q_1 R_1$ the *thin QR factorization* of A; Trefethen and Bau call this the *reduced QR factorization*.[1] If A is of full rank n and we require that the diagonal elements of R_1 are positive then R_1 and Q_1 are unique, but in general Q_2 is not. R_1 is then equal to the upper triangular factor of the Cholesky decomposition of $A^* A$ (= $A^{\mathrm{T}} A$ if A is real).

QL, RQ and LQ decompositions

Analogously, we can define QL, RQ, and LQ decompositions, with L being a *lower* triangular matrix.

Computing the QR decomposition

There are several methods for actually computing the QR decomposition, such as by means of the Gram–Schmidt process, Householder transformations, or Givens rotations. Each has a number of advantages and disadvantages.

Using the Gram–Schmidt process

For more details on this topic, see Gram–Schmidt#Numerical stability.

Consider the Gram–Schmidt process applied to the columns of the full column rank matrix $A = [\mathbf{a}_1, \cdots, \mathbf{a}_n]$, with inner product $\langle \mathbf{v}, \mathbf{w} \rangle = \mathbf{v}^\top \mathbf{w}$ (or $\langle \mathbf{v}, \mathbf{w} \rangle = \mathbf{v}^* \mathbf{w}$ for the complex case).

Define the projection:

$$\mathrm{proj}_{\mathbf{e}} \mathbf{a} = \frac{\langle \mathbf{e}, \mathbf{a} \rangle}{\langle \mathbf{e}, \mathbf{e} \rangle} \mathbf{e}$$

then:

$$\begin{aligned}
\mathbf{u}_1 &= \mathbf{a}_1, & \mathbf{e}_1 &= \frac{\mathbf{u}_1}{\|\mathbf{u}_1\|} \\
\mathbf{u}_2 &= \mathbf{a}_2 - \mathrm{proj}_{\mathbf{e}_1}\,\mathbf{a}_2, & \mathbf{e}_2 &= \frac{\mathbf{u}_2}{\|\mathbf{u}_2\|} \\
\mathbf{u}_3 &= \mathbf{a}_3 - \mathrm{proj}_{\mathbf{e}_1}\,\mathbf{a}_3 - \mathrm{proj}_{\mathbf{e}_2}\,\mathbf{a}_3, & \mathbf{e}_3 &= \frac{\mathbf{u}_3}{\|\mathbf{u}_3\|} \\
&\vdots & &\vdots \\
\mathbf{u}_k &= \mathbf{a}_k - \sum_{j=1}^{k-1}\mathrm{proj}_{\mathbf{e}_j}\,\mathbf{a}_k, & \mathbf{e}_k &= \frac{\mathbf{u}_k}{\|\mathbf{u}_k\|}
\end{aligned}$$

We then rearrange the equations above so that the $\mathbf{a}_i$s are on the left, using the fact that the $\mathbf{e}_i$ are unit vectors:

$$\begin{aligned}
\mathbf{a}_1 &= \langle\mathbf{e}_1,\mathbf{a}_1\rangle\mathbf{e}_1 \\
\mathbf{a}_2 &= \langle\mathbf{e}_1,\mathbf{a}_2\rangle\mathbf{e}_1 + \langle\mathbf{e}_2,\mathbf{a}_2\rangle\mathbf{e}_2 \\
\mathbf{a}_3 &= \langle\mathbf{e}_1,\mathbf{a}_3\rangle\mathbf{e}_1 + \langle\mathbf{e}_2,\mathbf{a}_3\rangle\mathbf{e}_2 + \langle\mathbf{e}_3,\mathbf{a}_3\rangle\mathbf{e}_3 \\
&\vdots \\
\mathbf{a}_k &= \sum_{j=1}^{k}\langle\mathbf{e}_j,\mathbf{a}_k\rangle\mathbf{e}_j
\end{aligned}$$

where $\langle\mathbf{e}_i,\mathbf{a}_i\rangle = \|\mathbf{u}_i\|$. This can be written in matrix form:

$$A = QR$$

where:

$$Q = [\mathbf{e}_1, \cdots, \mathbf{e}_n] \qquad \text{and} \qquad R = \begin{pmatrix} \langle\mathbf{e}_1,\mathbf{a}_1\rangle & \langle\mathbf{e}_1,\mathbf{a}_2\rangle & \langle\mathbf{e}_1,\mathbf{a}_3\rangle & \ldots \\ 0 & \langle\mathbf{e}_2,\mathbf{a}_2\rangle & \langle\mathbf{e}_2,\mathbf{a}_3\rangle & \ldots \\ 0 & 0 & \langle\mathbf{e}_3,\mathbf{a}_3\rangle & \ldots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix}.$$

Example

Consider the decomposition of

$$A = \begin{pmatrix} 12 & -51 & 4 \\ 6 & 167 & -68 \\ -4 & 24 & -41 \end{pmatrix}.$$

Recall that an orthogonal matrix Q has the property

$$Q^T\,Q = I.$$

Then, we can calculate Q by means of Gram–Schmidt as follows:

$$U = \begin{pmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \mathbf{u}_3 \end{pmatrix} = \begin{pmatrix} 12 & -69 & -58/5 \\ 6 & 158 & 6/5 \\ -4 & 30 & -33 \end{pmatrix};$$
$$Q = \begin{pmatrix} \frac{\mathbf{u}_1}{\|\mathbf{u}_1\|} & \frac{\mathbf{u}_2}{\|\mathbf{u}_2\|} & \frac{\mathbf{u}_3}{\|\mathbf{u}_3\|} \end{pmatrix} = \begin{pmatrix} 6/7 & -69/175 & -58/175 \\ 3/7 & 158/175 & 6/175 \\ -2/7 & 6/35 & -33/35 \end{pmatrix}.$$

Thus, we have

$$Q^T A = Q^T Q\, R = R;$$
$$R = Q^T A = \begin{pmatrix} 14 & 21 & -14 \\ 0 & 175 & -70 \\ 0 & 0 & 35 \end{pmatrix}.$$

Relation to RQ decomposition

The RQ decomposition transforms a matrix *A* into the product of an upper triangular matrix *R* (also known as right-triangular) and an orthogonal matrix *Q*. The only difference from QR decomposition is the order of these matrices.

QR decomposition is Gram–Schmidt orthogonalization of columns of *A*, started from the first column.

RQ decomposition is Gram–Schmidt orthogonalization of rows of *A*, started from the last row.

Using Householder reflections

A Householder reflection (or *Householder transformation*) is a transformation that takes a vector and reflects it about some plane or hyperplane. We can use this operation to calculate the *QR* factorization of an *m*-by-*n* matrix A with $m \geq n$.

Q can be used to reflect a vector in such a way that all coordinates but one disappear.

Let $\mathbf{x}$ be an arbitrary real *m*-dimensional column vector of A such that $\|\mathbf{x}\| = |\alpha|$ for a scalar *α*. If the algorithm is implemented using floating-point arithmetic, then *α* should get the opposite sign as the *k*-th coordinate of $\mathbf{x}$, where x_k is to be the pivot coordinate after which all entries are 0 in matrix *A'*s final upper triangular form, to avoid loss of significance. In the complex case, set

$$\alpha = -e^{i \arg x_k} \|\mathbf{x}\|$$

(Stoer & Bulirsch 2002, p. 225) and substitute transposition by conjugate transposition in the construction of *Q* below.

Then, where $\mathbf{e}_1$ is the vector $(1,0,...,0)^T$, ||·|| is the Euclidean norm and I is an *m*-by-*m* identity matrix, set

$$\mathbf{u} = \mathbf{x} - \alpha\mathbf{e}_1,$$
$$\mathbf{v} = \frac{\mathbf{u}}{\|\mathbf{u}\|},$$
$$Q = I - 2\mathbf{v}\mathbf{v}^T.$$

Or, if A is complex

$Q = I - (1+w)\mathbf{v}\mathbf{v}^H$, where $w = \mathbf{x}^H\mathbf{v}/\mathbf{v}^H\mathbf{x}$
where $\mathbf{x}^H$ is the conjugate transpose (transjugate) of $\mathbf{x}$

Q is an *m*-by-*m* Householder matrix and

$$Q\mathbf{x} = (\alpha, 0, \cdots, 0)^T.$$

This can be used to gradually transform an *m*-by-*n* matrix *A* to upper triangular form. First, we multiply *A* with the Householder matrix Q_1 we obtain when we choose the first matrix column for **x**. This results in a matrix Q_1A with zeros in the left column (except for the first row).

$$Q_1A = \begin{bmatrix} \alpha_1 & \star & \dots & \star \\ 0 & & & \\ \vdots & & A' & \\ 0 & & & \end{bmatrix}$$

This can be repeated for *A′* (obtained from Q_1A by deleting the first row and first column), resulting in a Householder matrix Q'_2. Note that Q'_2 is smaller than Q_1. Since we want it really to operate on Q_1A instead of *A′* we need to expand it to the upper left, filling in a 1, or in general:

$$Q_k = \begin{pmatrix} I_{k-1} & 0 \\ 0 & Q'_k \end{pmatrix}.$$

After t iterations of this process, $t = \min(m-1, n)$,

$$R = Q_t \cdots Q_2 Q_1 A$$

is an upper triangular matrix. So, with

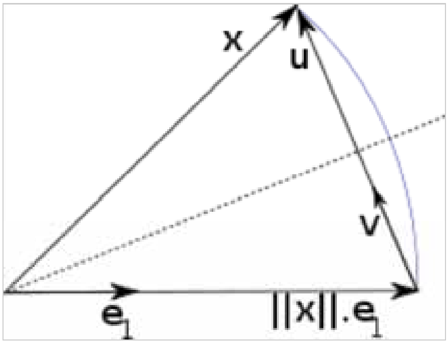
$$Q = Q_1^T Q_2^T \cdots Q_t^T,$$

$A = QR$ is a QR decomposition of A.

This method has greater numerical stability than the Gram–Schmidt method above.

The following table gives the number of operations in the *k*-th step of the QR-decomposition by the Householder transformation, assuming a square matrix with size *n*.

Operation	Number of operations in the *k*-th step
multiplications	$2(n-k+1)^2$
additions	$(n-k+1)^2 + (n-k+1)(n-k) + 2$
division	1
square root	1



Householder reflection for QR-decomposition: The goal is to find a linear transformation that changes the vector x into a vector of same length which is collinear to e_1. We could use an orthogonal projection (Gram-Schmidt) but this will be numerically unstable if the vectors x and e_1 are close to orthogonal. Instead, the Householder reflection reflects through the dotted line (chosen to bisect the angle between x and e_1). The maximum angle with this transform is at most 45 degrees.

Summing these numbers over the $n - 1$ steps (for a square matrix of size n), the complexity of the algorithm (in terms of floating point multiplications) is given by

$$\frac{2}{3}n^3 + n^2 + \frac{1}{3}n - 2 = O(n^3).$$

Example

Let us calculate the decomposition of

$$A = \begin{pmatrix} 12 & -51 & 4 \\ 6 & 167 & -68 \\ -4 & 24 & -41 \end{pmatrix}.$$

First, we need to find a reflection that transforms the first column of matrix A, vector $\mathbf{a}_1 = (12, 6, -4)^T$, to $\|\mathbf{a}_1\|\ \mathrm{e}_1 = (14, 0, 0)^T$.

Now,

$$\mathbf{u} = \mathbf{x} + \alpha\mathbf{e}_1,$$

and

$$\mathbf{v} = \frac{\mathbf{u}}{\|\mathbf{u}\|}.$$

Here,

$\alpha = -14$ and $\mathbf{x} = \mathbf{a}_1 = (12, 6, -4)^T$

Therefore

$\mathbf{u} = (-2, 6, -4)^T = (2)(-1, 3, -2)^T$ and $\mathbf{v} = \frac{1}{\sqrt{14}}(-1, 3, -2)^T$, and then

$$Q_1 = I - \frac{2}{\sqrt{14}\sqrt{14}} \begin{pmatrix} -1 \\ 3 \\ -2 \end{pmatrix} \begin{pmatrix} -1 & 3 & -2 \end{pmatrix}$$

$$= I - \frac{1}{7} \begin{pmatrix} 1 & -3 & 2 \\ -3 & 9 & -6 \\ 2 & -6 & 4 \end{pmatrix}$$

$$= \begin{pmatrix} 6/7 & 3/7 & -2/7 \\ 3/7 & -2/7 & 6/7 \\ -2/7 & 6/7 & 3/7 \end{pmatrix}.$$

Now observe:

$$Q_1 A = \begin{pmatrix} 14 & 21 & -14 \\ 0 & -49 & -14 \\ 0 & 168 & -77 \end{pmatrix},$$

so we already have almost a triangular matrix. We only need to zero the (3, 2) entry.

Take the (1, 1) minor, and then apply the process again to

$$A' = M_{11} = \begin{pmatrix} -49 & -14 \\ 168 & -77 \end{pmatrix}.$$

By the same method as above, we obtain the matrix of the Householder transformation

$$Q_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 7/25 & -24/25 \\ 0 & -24/25 & -7/25 \end{pmatrix}$$

after performing a direct sum with 1 to make sure the next step in the process works properly.

Now, we find

$$Q = Q_1^T Q_2^T = \begin{pmatrix} 6/7 & 69/175 & -58/175 \\ 3/7 & -158/175 & 6/175 \\ -2/7 & -6/35 & -33/35 \end{pmatrix}$$

Then

$$Q = Q_1^T Q_2^T = \begin{pmatrix} 0.8571 & 0.3943 & -0.3314 \\ 0.4286 & -0.9029 & 0.0343 \\ -0.2857 & -0.1714 & -0.9429 \end{pmatrix}$$

$$R = Q_2 Q_1 A = Q^T A = \begin{pmatrix} 14 & 21 & -14 \\ 0 & -175 & 70 \\ 0 & 0 & 35 \end{pmatrix}.$$

The matrix Q is orthogonal and R is upper triangular, so $A = QR$ is the required QR-decomposition.

Using Givens rotations

QR decompositions can also be computed with a series of Givens rotations. Each rotation zeros an element in the subdiagonal of the matrix, forming the *R* matrix. The concatenation of all the Givens rotations forms the orthogonal *Q* matrix.

In practice, Givens rotations are not actually performed by building a whole matrix and doing a matrix multiplication. A Givens rotation procedure is used instead which does the equivalent of the sparse Givens matrix multiplication, without the extra work of handling the sparse elements. The Givens rotation procedure is useful in situations where only a relatively few off diagonal elements need to be zeroed, and is more easily parallelized than Householder transformations.

Example

Let us calculate the decomposition of

$$A = \begin{pmatrix} 12 & -51 & 4 \\ 6 & 167 & -68 \\ -4 & 24 & -41 \end{pmatrix}.$$

First, we need to form a rotation matrix that will zero the lowermost left element, $\mathbf{a}_{31} = -4$. We form this matrix using the Givens rotation method, and call the matrix G_1. We will first rotate the vector $(6, -4)$, to point along the *X* axis. This vector has an angle $\theta = \arctan\left(\frac{-(-4)}{6}\right)$. We create the orthogonal Givens rotation matrix, G_1:

$$G_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{pmatrix}$$
$$\approx \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0.83205 & -0.55470 \\ 0 & 0.55470 & 0.83205 \end{pmatrix}$$

And the result of $G_1 A$ now has a zero in the $\mathbf{a}_{31}$ element.

$$G_1 A \approx \begin{pmatrix} 12 & -51 & 4 \\ 7.21110 & 125.6396 & -33.83671 \\ 0 & 112.6041 & -71.83368 \end{pmatrix}$$

We can similarly form Givens matrices G_2 and G_3, which will zero the sub-diagonal elements a_{21} and a_{32}, forming a triangular matrix R. The orthogonal matrix Q^T is formed from the concatenation of all the Givens matrices $Q^T = G_3 G_2 G_1$. Thus, we have $G_3 G_2 G_1 A = Q^T A = R$, and the *QR* decomposition is $A = QR$.

Connection to a determinant or a product of eigenvalues

We can use QR decomposition to find the absolute value of the determinant of a square matrix. Suppose a matrix is decomposed as $A = QR$. Then we have

$$\det(A) = \det(Q) \cdot \det(R).$$

Since *Q* is unitary, $|\det(Q)| = 1$. Thus,

$$|\det(A)| = |\det(R)| = \Big|\prod_i r_{ii}\Big|,$$

where r_{ii} are the entries on the diagonal of *R*.

Furthermore, because the determinant equals the product of the eigenvalues, we have

$$\Big|\prod_i r_{ii}\Big| = \Big|\prod_i \lambda_i\Big|,$$

where λ_i are eigenvalues of A.

We can extend the above properties to non-square complex matrix A by introducing the definition of QR-decomposition for non-square complex matrix and replacing eigenvalues with singular values.

Suppose a QR decomposition for a non-square matrix *A*:

$$A = Q\begin{pmatrix} R \\ O \end{pmatrix}, \qquad Q^*Q = I,$$

where O is a zero matrix and Q is a unitary matrix.

From the properties of SVD and determinant of matrix, we have

$$\Big|\prod_i r_{ii}\Big| = \prod_i \sigma_i,$$

where σ_i are singular values of A.

Note that the singular values of A and R are identical, although their complex eigenvalues may be different. However, if *A* is square, the following is true:

$$\prod_i \sigma_i = \Big|\prod_i \lambda_i\Big|.$$

In conclusion, QR decomposition can be used efficiently to calculate the product of the eigenvalues or singular values of a matrix.

Column pivoting

QR decomposition with column pivoting introduces a permutation matrix *P*:

$$AP = QR \quad \iff \quad A = QRP^T$$

Column pivoting is useful when *A* is (nearly) rank deficient, or is suspected of being so. It can also improve numerical accuracy. *P* is usually chosen so that the diagonal elements of *R* are non-increasing: $|r_{11}| \geq |r_{22}| \geq \ldots \geq |r_{nn}|$. This can be used to find the (numerical) rank of *A* at lower computational cost than a singular value decomposition, forming the basis of so-called rank-revealing QR algorithms.

Using for solution to linear inverse problems

Compared to the direct matrix inverse, inverse solutions using QR decomposition are more numerically stable as evidenced by their reduced condition numbers [Parker, Geophysical Inverse Theory, Ch1.13].

To solve the underdetermined($m < n$) linear problem $Ax = b$ where the matrix A has dimensions $m \times n$ and rank m, first find the QR factorization of the transpose of A: $A^T = QR$, where Q is an orthogonal matrix (i.e. $Q^T = Q^{-1}$), and R has a special form: $R = \begin{bmatrix} R_1 \\ 0 \end{bmatrix}$. Here R_1 is a square $m \times m$ right triangular matrix, and the zero matrix has dimension $(n-m) \times m$. After some algebra, it can be shown that a solution to the inverse problem can be expressed as: $x = Q\begin{bmatrix} (R_1^T)^{-1}b \\ 0 \end{bmatrix}$ where one may either find R_1^{-1} by Gaussian elimination or compute $(R_1^T)^{-1}b$ directly by forward substitution. The latter technique enjoys greater numerical accuracy and lower computations.

To find a solution, $\hat{x}$, to the overdetermined ($m \geq n$) problem $Ax = b$ which minimizes the norm $\|A\hat{x} - b\|$, first find the QR factorization of A: $A = QR$. The solution can then be expressed as $\hat{x} = R_1^{-1}(Q_1^T b)$, where Q_1 is an $m \times n$ matrix containing the first n columns of the full orthonormal basis Q and where R_1 is as before. Equivalent to the underdetermined case, back substitution can be used to quickly and accurately find this $\hat{x}$ without explicitly inverting R_1. (Q_1 and R_1 are often provided by numerical libraries as an "economic" QR decomposition.)

See also

- Polar decomposition
- Eigenvalue decomposition
- Spectral decomposition
- Matrix decomposition
- Zappa–Szép product

Historical notes

The QR decomposition, considered to be one of the 10 most important algorithms of the 20th century, was first published in a pair of papers by John GF Francis published in 1961 and 1962.

References

1. ^ *a b c* L. N. Trefethen and D. Bau, *Numerical Linear Algebra* (SIAM, 1997).

- Golub, Gene H.; Van Loan, Charles F. (1996), *Matrix Computations* (3rd ed.), Johns Hopkins, ISBN 978-0-8018-5414-9.
- Horn, Roger A.; Johnson, Charles R. (1985), *Matrix Analysis*, Cambridge University Press, ISBN 0-521-38632-2. Section 2.8.
- Press, WH; Teukolsky, SA; Vetterling, WT; Flannery, BP (2007), "Section 2.10. QR Decomposition" (http://apps.nrbook.com/empanel/index.html?pg=102), *Numerical Recipes: The Art of Scientific Computing* (3rd ed.), New York: Cambridge University Press, ISBN 978-0-521-88068-8
- Stoer, Josef; Bulirsch, Roland (2002), *Introduction to Numerical Analysis* (3rd ed.), Springer, ISBN 0-387-95452-X.

External links

- Online Matrix Calculator (http://www.bluebit.gr/matrix-calculator/) Performs QR decomposition of matrices.
- LAPACK users manual (http://netlib.org/lapack/lug/node39.html) gives details of subroutines to calculate the QR decomposition
- Mathematica users manual (http://documents.wolfram.com/mathematica/functions/QRDecomposition) gives details and examples of routines to calculate QR decomposition
- ALGLIB (http://www.alglib.net/) includes a partial port of the LAPACK to C++, C#, Delphi, etc.
- Eigen::QR (http://eigen.tuxfamily.org/dox-devel/group__QR__Module.html) Includes C++ implementation of QR decomposition.
- Into (http://intopii.com/into/) contains an open source implementation of QR decomposition in C++.