Notes on Matrix Decomposition

Paul F. Roysdon, Ph.D.

I. Matrix Decomposition

Matrix factorization, or decomposition, is a factorization of a matrix into a product of matrices, e.g. QR Decomposition for $\mathbf{A} = \mathbf{QR}$ (see Section VI). These methods solve the system of linear equations $\mathbf{A}\boldsymbol{x} = \mathbf{b}$ by forward or backward substitution, resulting in fewer additions and multiplications. In practice, matrix decomposition is most commonly used to solve the matrix inverse $\mathbf{A}^{-1}$. Some useful concepts are necessary prior to the presentation of the most common decomposition methods.

II. Initial Concepts

A. Triangular matrix

A square matrix $\mathbf{A}$ is *lower triangular* if $A_{ij} = 0$ for $j > i$:

$$\mathbf{A} = \begin{bmatrix} A_{11} & 0 & \cdots & 0 & 0 \\ A_{21} & A_{22} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & 0 & 0 \\ A_{n-1,1} & A_{n-1,2} & \cdots & A_{n-1,n-1} & 0 \\ A_{n1} & \cdots & \cdots & A_{n,n-1} & A_{nn} \end{bmatrix}$$

$\mathbf{A}$ is *upper triangular* if $A_{ij} = 0$ for $j < i$, e.g. $\mathbf{A}^\intercal$ is lower triangular. A triangular matrix is unit upper/lower triangular if $A_{ii} = 1$ for all i.

B. Forward Substitution

Solve $\mathbf{A}\boldsymbol{x} = \mathbf{b}$ when $\mathbf{A}$ is lower triangular with nonzero diagonal elements:

$$\begin{aligned} x_1 &= b_1 A_{11} \\ x_2 &= (b_2 - A_{21}x_1)/A_{22} \\ x_3 &= (b_3 - A_{31}x_1 - A_{32}x_2)/A_{33} \\ &\vdots \\ x_n &= (b_n - A_{n1}x_1 - A_{n2}x_2 - \cdots - A_{n,n-1}x_{n-1})/A_{nn} \end{aligned}$$

Complexity is $1 + 3 + 5 + \cdots + (2n-1) = n^2$ flops.

C. Back Substitution

Solve $\mathbf{A}\boldsymbol{x} = \mathbf{b}$ when $\mathbf{A}$ is upper triangular with nonzero diagonal elements:

$$\begin{aligned} x_n &= b_n/A_{nn} \\ x_{n-1} &= (b_{n-1} - A_{n-1,n}x_n)/A_{n-1,n-1} \\ x_{n-2} &= (b_{n-2} - A_{n-2,n-1}x_{n-1})/A_{n-2,n-2} \\ &\vdots \\ x_1 &= (b_1 - A_{12}x_2 - A_{13}x_3 - \cdots - A_{1n}x_n)/A_{11} \end{aligned}$$

Complexity is n^2 flops.

D. Inverse of a Triangular Matrix

A triangular matrix $\mathbf{A}$ with nonzero diagonal elements is nonsingular:

$$\mathbf{A}\boldsymbol{x} = \mathbf{0} \quad \Longrightarrow \quad \boldsymbol{x} = \mathbf{0}$$

this follows from forward or back substitution applied to the equation $\mathbf{A}\boldsymbol{x} = \mathbf{0}$

- The inverse of $\mathbf{A}$ can be computed by solving $\mathbf{AX} = \mathbf{I}$ column by column: $\mathbf{A}[\boldsymbol{x}_1\ \boldsymbol{x}_2\ \cdots\ \boldsymbol{x}_n] = [\mathbf{e}_1\ \mathbf{e}_2\ \cdots\ \mathbf{e}_n]$, where $\boldsymbol{x}_i$ is the column i of $\mathbf{X}$.
- The inverse of lower triangular matrix is lower triangular.
- The inverse of upper triangular matrix is upper triangular.
- The complexity of computing inverse of $n \times n$ triangular matrix is $n^2 + (n-1)^2 + \cdots 1 \approx \frac{1}{3}n^3$ flops.

III. LU Decomposition

Assume $\mathbf{A}$ is a square matrix with non-zero leading principal minors, then

$$\mathbf{A} = \mathbf{LU}$$

where $\mathbf{L}$ is a unique unit lower triangular matrix and $\mathbf{U}$ is a unique upper triangular matrix.

IV. LUP Decomposition

Assume $\mathbf{A}$ is a square matrix with non-zero leading principal minors, then

$$\mathbf{A} = \mathbf{LUP}$$

where $\mathbf{L}$ is a unique unit lower triangular matrix, $\mathbf{U}$ is a unique upper triangular matrix, and $\mathbf{P}$ is a permutation matrix.

V. LDL Decomposition

The LDL decomposition is a special case of the LU decomposition. Assume $\mathbf{A}$ is a non-singular symmetric definite square matrix, then

$$\mathbf{A} = \mathbf{LDL}^\intercal = \mathbf{L}^\intercal\mathbf{DL}$$

where $\mathbf{L}$ is a unit lower triangular matrix and $\mathbf{D}$ is a diagonal matrix. If $\mathbf{A}$ is also positive definite, then $\mathbf{D}$ has strictly positive diagonal entries.

VI. QR Decomposition

If $\mathbf{A} \in \mathbb{R}^{m\times n}$ has linearly independent columns then it can be factored as

$$\begin{aligned}\mathbf{A} &= \mathbf{QR}\\ &= [\mathbf{q}_1\ \mathbf{q}_2\ \cdots\ \mathbf{q}_n]\begin{bmatrix} R_{11} & R_{12} & \cdots & R_{1n}\\ 0 & r_{22} & \cdots & R_{2n}\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & R_{n,n}\end{bmatrix}\end{aligned}$$

where the vectors $\mathbf{q}_1, \cdots, \mathbf{q}_n$ are orthonormal m-vectors:

$$\|\mathbf{q}_i\| = 1, \quad \mathbf{q}_i^\intercal \mathbf{q}_j = 0, \quad \text{if } i \neq j,$$

and the diagonal elements R_{ii} are nonzero. If $R_{ii} < 0$ we can switch the signs of $R_{ii}, \ldots, R_{in}$ and the vector $\mathbf{q}_i$. If we define $R_{ii} > 0$, then $\mathbf{Q}$ and $\mathbf{R}$ are unique.

In matrix notation, if $\mathbf{A} \in \mathbb{R}^{m\times n}$ and has linearly independent columns, then it can be factored as $\mathbf{A} = \mathbf{QR}$.

Define $\mathbf{Q} \in \mathbb{R}^{m\times n}$ with orthonormal columns, i.e. $\mathbf{Q}^\intercal\mathbf{Q} = \mathbf{I}$. If $\mathbf{A}$ is square ($m = n$), then $\mathbf{Q}$ is orthogonal, i.e. $\mathbf{Q}^\intercal\mathbf{Q} = \mathbf{Q}\mathbf{Q}^\intercal = \mathbf{I}$.

Define $\mathbf{R} \in \mathbb{R}^{n\times n}$, upper-triangular, with nonzero diagonal elements, i.e. $R_{ii} \neq 0$, therefore nonsingular.

A. QR Factorization and the Normal Equations

The normal equations, can be factored using QR factorization:

$$\begin{aligned}\mathbf{A}\boldsymbol{x} &= \mathbf{b}\\ \mathbf{A}^\intercal\mathbf{A}\boldsymbol{x} &= \mathbf{A}^\intercal\mathbf{b}\\ \mathbf{R}^\intercal\mathbf{Q}^\intercal\mathbf{Q}\mathbf{R}\boldsymbol{x} &= \mathbf{R}^\intercal\mathbf{Q}^\intercal\mathbf{b}\\ \mathbf{R}^\intercal\mathbf{R}\boldsymbol{x} &= \mathbf{R}^\intercal\mathbf{Q}^\intercal\mathbf{b} \quad (\mathbf{Q}^\intercal\mathbf{Q} = \mathbf{I})\\ \mathbf{R}\boldsymbol{x} &= \mathbf{Q}^\intercal\mathbf{b} \quad (\mathbf{R} \text{ non-singular}).\end{aligned}$$

The cost to solve the normal equations (e.g. a least squares problem) using QR for large (m, n), is $2mn^2$ flops.

1) QR factorization of $\mathbf{A}$: $\mathbf{A} = \mathbf{QR}$ ($2mn^2$ flops).
2) Form $\mathbf{d} = \mathbf{Q}^\intercal\mathbf{b}$ ($2mn$ flops).
3) Solve $\mathbf{R}\boldsymbol{x} = \mathbf{d}$ by back substitution (n^2 flops).

While $\mathcal{O}(2mn^2)$, QR is numerically stable, works on most matrices, and does note require square $\mathbf{A}$. However, exploiting sparsity in QR factorization can be more difficult.

B. QR Factorization and the Pseudo-inverse

The pseudo-inverse of a matrix $\mathbf{A}$ with linearly independent columns is defined as:

$$\mathbf{A}^\dagger = (\mathbf{A}^\intercal\mathbf{A})^{-1}\mathbf{A}^\intercal$$

in terms of QR:

$$\begin{aligned}\mathbf{A}^\dagger &= ((\mathbf{QR})^\intercal(\mathbf{QR}))^{-1}(\mathbf{QR})^\intercal\\ &= (\mathbf{R}^\intercal\mathbf{Q}^\intercal\mathbf{QR})^{-1}\mathbf{R}^\intercal\mathbf{Q}^\intercal\\ &= (\mathbf{R}^\intercal\mathbf{R})^{-1}\mathbf{R}^\intercal\mathbf{Q}^\intercal \quad (\mathbf{Q}^\intercal\mathbf{Q} = \mathbf{I})\\ &= \mathbf{R}^{-1}\mathbf{R}^{-\intercal}\mathbf{R}^\intercal\mathbf{Q}^\intercal \quad (\mathbf{R} \text{ is nonsingular})\\ &= \mathbf{R}^{-1}\mathbf{Q}^\intercal.\end{aligned}$$

For square nonsingular $\mathbf{A}$, the inverse is

$$\mathbf{A}^{-1} = (\mathbf{QR})^{-1} = \mathbf{R}^{-1}\mathbf{Q}^\intercal.$$

C. QR Factorization and Range

Recall definition of range of $\mathbf{A} \in \mathbb{R}^{m\times n}$ is

$$\text{range}(\mathbf{A}) = \{\mathbf{A}\boldsymbol{x} \mid \boldsymbol{x} \in \mathbb{R}^n\}.$$

Suppose $\mathbf{A}$ has linearly independent columns with QR factors $\mathbf{Q}$, $\mathbf{R}$, then $\mathbf{Q}$ has the same range as $\mathbf{A}$:

$$\begin{aligned}\mathbf{y} \in \text{range}(\mathbf{A}) &\Leftrightarrow \mathbf{y} = \mathbf{A}\boldsymbol{x} \text{ for some } \boldsymbol{x}\\ &\Leftrightarrow \mathbf{y} = \mathbf{QR}\boldsymbol{x} \text{ for some } \boldsymbol{x}\\ &\Leftrightarrow \mathbf{y} = \mathbf{Q}\mathbf{z} \text{ for some } \mathbf{z}\\ &\Leftrightarrow \mathbf{y} \in \text{range}(\mathbf{Q}),\end{aligned}$$

and the columns of $\mathbf{Q}$ are orthonormal and have the same span as columns of $\mathbf{A}$.

D. Projection on Range

Combining $\mathbf{A} = \mathbf{QR}$ and $\mathbf{A}^\dagger = \mathbf{R}^{-1}\mathbf{Q}^\intercal$ produces

$$\mathbf{A}\mathbf{A}^\dagger = \mathbf{QR}\mathbf{R}^{-1}\mathbf{Q}^\intercal = \mathbf{Q}\mathbf{Q}^\intercal.$$

Note the order of the product $\mathbf{A}\mathbf{A}^\dagger$, and the difference with $\mathbf{A}^\dagger\mathbf{A} = \mathbf{I}$.

Recall that $\mathbf{A}\mathbf{A}^\intercal\boldsymbol{x}$ is the projection of $\boldsymbol{x}$ on the range of $\mathbf{A}$. Similarly, $\mathbf{Q}\mathbf{Q}^\intercal\boldsymbol{x}$ is the projection of $\boldsymbol{x}$ on the range of $\mathbf{Q}$, and by the definition above range$(\mathbf{A})$ = range$(\mathbf{Q})$.

E. QR with Gram-Schmidt

This will not be covered here, but is mentioned for the following reasons:

- complexity is $2mn^2$ flops.
- not recommended in practice (sensitive to rounding errors).

F. QR with modified Gram-Schmidt

The Gram-Schmidt QR algorithm computes $\mathbf{Q}$ and $\mathbf{R}$ column by column. After k steps we have a partial QR factorization:

$$[\mathbf{a}_1\ \mathbf{a}_2\ \cdots\ \mathbf{a}_k] = [\mathbf{q}_1\ \mathbf{q}_2\ \cdots\ \mathbf{q}_k]\begin{bmatrix} R_{11} & R_{12} & \cdots & R_{1k}\\ 0 & R_{22} & \cdots & R_{2k}\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & R_{kk}\end{bmatrix}$$

the columns $\mathbf{q}_1, \ldots, \mathbf{q}_k$ are orthonormal, the diagonal elements $r_{11}, r_{22}, \ldots, r_{kk}$ are positive, and the columns $\mathbf{q}_1, \ldots, \mathbf{q}_k$ have the same span as $\mathbf{a}_1, \ldots, \mathbf{a}_k$.

Suppose we have completed the factorization for the first $k-1$ columns Column k of the equation $\mathbf{A} = \mathbf{QR}$ is

$$\mathbf{a}_k = R_{1k}\mathbf{q}_1 + R_{2k}\mathbf{q}_2 + \cdots + R_{k-1,k}\mathbf{q}_{k-1} + R_{kk}\mathbf{q}_k.$$

Regardless of how we choose $R_{1k}, \ldots, R_{k-1,k}$, the vector

$$\tilde{\mathbf{q}}_k = \mathbf{a}_k - R_{1k}\mathbf{q}_1 - R_{2k}\mathbf{q}_2 - \cdots - R_{k-1,k}\mathbf{q}_{k-1}$$

will be nonzero: $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_k$ are linearly independent and therefore

$$\mathbf{a}_k \notin \text{span}\{\mathbf{a}_1, \ldots, \mathbf{a}_{k-1}\} = \text{span}\{\mathbf{q}_1, \ldots, \mathbf{q}_{k-1}\}.$$

If $\mathbf{q}_k$ is normalized, e.g. $\tilde{\mathbf{q}}_k$: choose $R_{kk} = \|\tilde{\mathbf{q}}_k\|$ and $\mathbf{q}_k = (1/R_{kk})\tilde{\mathbf{q}}_k$. Note, $\tilde{\mathbf{q}}_k$ and $\mathbf{q}_k$ are orthogonal to $\mathbf{q}_1, \ldots, \mathbf{q}_{k-1}$ if we choose $r_{1k}, \ldots, r_{k-1,k}$ as

$$R_{1k} = \mathbf{q}_1^\intercal \mathbf{a}_k, \quad R_{2k} = \mathbf{q}_2^\intercal \mathbf{a}_k, \quad \ldots, \quad R_{k-1,k} = \mathbf{q}_{k-1}^\intercal \mathbf{a}_k.$$

Given $\mathbf{A} \in \mathbb{R}^{m\times n}$ with linearly independent columns $\mathbf{a}_1, \ldots, \mathbf{a}_n$, the modified Gram-Schmidt algorithm for $k = 1, \ldots, n$ is:

$$\begin{aligned}
R_{1k} &= \mathbf{q}_1^\intercal \mathbf{a}_k \\
R_{2k} &= \mathbf{q}_2^\intercal \mathbf{a}_k \\
&\vdots \\
R_{k-1,k} &= \mathbf{q}_{k-1}^\intercal \mathbf{a}_k \\
\tilde{\mathbf{q}}_k &= \mathbf{a}_k - (R_{1k}\mathbf{q}_1 + R_{2k}\mathbf{q}_2 + \cdots + R_{k-1,k}\mathbf{q}_{k-1}) \\
R_{kk} &= \|\tilde{\mathbf{q}}_k\| \\
\mathbf{q}_k &= \frac{1}{R_{kk}}\tilde{\mathbf{q}}_k
\end{aligned}$$

Note the modified Gram-Schmidt algorithm:

- Has complexity $2mn^2$ flops.
- Has better numerical properties than the standard Gram-Schmidt algorithm.

G. QR with Householder Reflection

The Householder algorithm is the most widely used algorithm for QR factorization. Householder QR is less sensitive to rounding error than GramSchmidt algorithm. Householder QR computes a 'full' QR factorization

$$\mathbf{A} = \begin{bmatrix} \mathbf{Q} & \tilde{\mathbf{Q}} \end{bmatrix} \begin{bmatrix} \mathbf{R} \\ \mathbf{0} \end{bmatrix}$$

The full Q-factor is constructed as a product of elementary orthogonal matrices

$$\begin{bmatrix} \mathbf{Q} & \tilde{\mathbf{Q}} \end{bmatrix} = \mathbf{H}_1\mathbf{H}_2 \ldots \mathbf{H}_n$$

where each $\mathbf{H}_i$ is an $m\times m$ symmetric, orthogonal 'reflector', and complexity is $2mn^2 - \frac{2}{3}n^3$ flops.

The Householder QR reflector, is defined as

$$\mathbf{H} = \mathbf{I} - 2\mathbf{v}\mathbf{v}^\intercal \text{ with } \|\mathbf{v}\| = 1$$

where $\mathbf{H}$ is symmetric, orthogonal. The product $\mathbf{H}\boldsymbol{x}$ is a reflection of $\boldsymbol{x}$ through the hyperplane $\{\mathbf{z} \mid \mathbf{v}^\intercal\mathbf{z} = 0\}$, and can be computed efficiently as

$$\mathbf{H}\boldsymbol{x} = \boldsymbol{x} - 2(\mathbf{v}^\intercal\boldsymbol{x})\mathbf{v}$$

with complexity $4p$ flops for $\mathbf{v}, \boldsymbol{x} \in \mathbb{R}^p$.

Given the nonzero p-vector $\mathbf{y} = (y_1, \ldots, y_p)$, define

$$\mathbf{w} = \begin{bmatrix} y_1 + \text{sign}(y_1)\|\mathbf{y}\| \\ y_2 \\ \vdots \\ y_p \end{bmatrix}, \quad \mathbf{v} = \frac{1}{\|\mathbf{w}\|}\mathbf{w}$$

Define $\text{sign}(0) = 1$. The vector $\mathbf{w}$ satisfies the property

$$\|\mathbf{w}\|^2 = 2(\mathbf{w}^\intercal\mathbf{y}) = 2\|\mathbf{y}\|(\|\mathbf{y}\||y_1|).$$

The reflector $\mathbf{H} = \mathbf{I} - 2\mathbf{v}\mathbf{v}^\intercal$ maps $\mathbf{y}$ to multiple $\mathbf{e}_1 = (1, 0, \ldots, 0)$:

$$\mathbf{H}\mathbf{y} = \mathbf{y} - \frac{2(\mathbf{w}^\intercal\mathbf{y})}{\|\mathbf{w}\|^2}\mathbf{w} = \mathbf{y} - \mathbf{w} = -\text{sign}(y_1)\|\mathbf{y}\|e_1.$$

Notice the vector length is retained through the reflection!

The geometry of the Householder QR reflection is defined as shown in Fig. 1, where the reflection through the

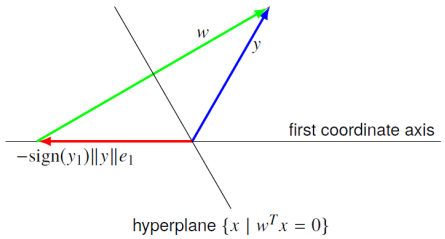


Fig. 1. Householder Reflection Geometry.

hyperplane $\{\boldsymbol{x} \mid \mathbf{w}^\intercal\boldsymbol{x} = 0\}$ with normal vector $\mathbf{w} = \mathbf{y} + \text{sign}(y_1)\|\mathbf{y}\|e_1$ maps $\mathbf{y}$ to the vector $\text{sign}(y_1)\|\mathbf{y}\|e_1$.

In matrix form, Householder QR computes reflectors $\mathbf{H}_1, \ldots, \mathbf{H}_n$ that reduce $\mathbf{A}$ to triangular form:

$$\mathbf{H}_n\mathbf{H}_{n-1}\cdots\mathbf{H}_1\mathbf{A} = \begin{bmatrix} \mathbf{R} \\ \mathbf{0} \end{bmatrix}.$$

After step k, the matrix $\mathbf{H}_n\mathbf{H}_{n-1}\cdots\mathbf{H}_1\mathbf{A}$ has the structure shown in Fig. 2, where elements in positions i, j for $i > j$

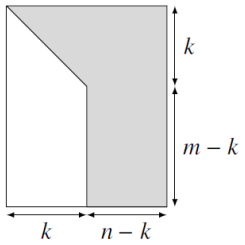


Fig. 2. Householder Structure.

and $j \le k$ are zero.

The Householder QR algorithm for $k = 1, \ldots, n$ is:

1) Define $\mathbf{y} = \mathbf{A}_{k:m,k}$ and compute $(m-k+1)$-vector $\mathbf{v}_k$:

$$\mathbf{w} = \mathbf{y} + \text{sign}(y_1)\|\mathbf{y}\|\mathbf{e}_1, \quad \mathbf{v} = \frac{1}{\|\mathbf{w}\|}\mathbf{w}.$$

2) multiply $\mathbf{A}_{k:m,k:n}$ with reflector $\mathbf{I} - 2\mathbf{v}_k\mathbf{v}_k^\intercal$:

$$\mathbf{A}_{k:m,k:n} := \mathbf{A}_{k:m,k:n} - 2\mathbf{v}_k(\mathbf{v}_k^\intercal\mathbf{A}_{k:m,k:n}).$$

Some final comments. Notice in step 2, we multiply $\mathbf{A}_{k:m,k:n}$ with the reflector $\mathbf{I} - 2\mathbf{v}_k\mathbf{v}_k^\intercal$:

$$(\mathbf{I} - 2\mathbf{v}_k\mathbf{v}_k^\intercal)\mathbf{A}_{k:m,k:n} := \mathbf{A}_{k:m,k:n} - 2\mathbf{v}_k(\mathbf{v}_k^\intercal\mathbf{A}_{k:m,k:n}).$$

this is equivalent to multiplying $\mathbf{A}$ with $m \times m$ reflector

$$\mathbf{H}_k = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} - 2\mathbf{v}_k\mathbf{v}_k^\intercal \end{bmatrix} = \mathbf{I} - 2\begin{bmatrix} \mathbf{0} \\ \mathbf{v}_k \end{bmatrix}\begin{bmatrix} \mathbf{0} \\ \mathbf{v}_k \end{bmatrix}^\intercal$$

Finally, notice that the algorithm overwrites $\mathbf{A}$ with

$$\begin{bmatrix} \mathbf{R} \\ \mathbf{0} \end{bmatrix},$$

and returns the vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$, with $\mathbf{v}_k$ of length $m-k+1$.

Complexity in cycle k, the dominant terms are

- $(2(m-k+1)-1)(n-k+1)$ flops for product $\mathbf{v}_k(\mathbf{A}_{k:m,k:n})$.
- $(m-k+1)(n-k+1)$ flops for outer product with $\mathbf{v}_k$.
- $(m-k+1)(n-k+1)$ flops for subtraction from $\mathbf{A}_{k:m,k:n}$.

the sum is roughly $4(m-k+1)(n-k+1)$ flops.

Total for computing R and vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$:

$$\sum_{k=1}^{n} 4(m-k+1)(n-k+1) \approx \int_0^n 4(m-t)(n-t)dt,$$
$$= 2mn^2 - \frac{2}{3}n^3 \text{ flops}$$

The Householder algorithm returns the vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$ that define $[\mathbf{Q}\ \tilde{\mathbf{Q}}] = \mathbf{H}_1\mathbf{H}_2\cdots\mathbf{H}_n$, thus

- usually there is no need to compute the matrix $[\mathbf{Q}\ \tilde{\mathbf{Q}}]$ explicitly.
- the vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$ are an economical representation of $[\mathbf{Q}\ \tilde{\mathbf{Q}}]$.
- products with $[\mathbf{Q}\ \tilde{\mathbf{Q}}]$ or its transpose can be computed as $[\mathbf{Q}\ \tilde{\mathbf{Q}}]\boldsymbol{x} = \mathbf{H}_1\mathbf{H}_2\cdots\mathbf{H}_n\boldsymbol{x}$ and $[\mathbf{Q}\ \tilde{\mathbf{Q}}]^\intercal\mathbf{y} = \mathbf{H}_n\mathbf{H}_{n-1}\cdots\mathbf{H}_1\mathbf{y}$.

Complexity due to multiplication with Q-factor is

- the matrix-vector product $\mathbf{H}_k\boldsymbol{x}$, defined as

$$\mathbf{H}_k\boldsymbol{x} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} - 2\mathbf{v}_k\mathbf{v}_k^\intercal \end{bmatrix}\begin{bmatrix} \boldsymbol{x}_{1:k-1} \\ \boldsymbol{x}_{k:m} \end{bmatrix}$$
$$= \begin{bmatrix} \boldsymbol{x}_{1:k-1} \\ \boldsymbol{x}_{k:m} - 2(\mathbf{v}_k^\intercal\boldsymbol{x}_{k:m})\mathbf{v}_k \end{bmatrix}.$$

- complexity of multiplication $\mathbf{H}_k\boldsymbol{x}$ is $4(m-k+1)$ flops.
- complexity of multiplication with $\mathbf{H}_1\mathbf{H}_2\cdots\mathbf{H}_n$ or its transpose is $\sum_{k=1}^{n} 4(m-k+1) \approx 4mn - 2n^2$ flops.
- roughly equal to matrix-vector product with $m \times n$ matrix ($2mn$ flops).

H. QR with Givens Rotations

Let $\mathbf{A} \in \mathbb{R}^2$, the QR decomposition computes $\mathbf{Q}^\intercal\mathbf{A} = \mathbf{R}$ by

$$\begin{bmatrix} \gamma & -\sigma \\ \sigma & \gamma \end{bmatrix}^\intercal\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix},$$

where $\gamma^2 + \sigma^2 = 1$.

From the relationship $-\sigma A_{11} + \gamma A_{21} = 0$ we obtain

$$\gamma A_{21} = \sigma A_{11}$$
$$\gamma^2 A_{21}^2 = \sigma^2 A_{11} = (1-\gamma^2)A_{11}^2$$

which yields

$$\gamma = \pm\frac{\sigma A_{11}}{\sqrt{A_{21}^2 + A_{11}^2}}.$$

It is conventional to choose the $+$ sign. We obtain

$$\sigma^2 = 1 - \gamma^2 = 1 - \frac{A_{11}^2}{A_{21}^2 + A_{11}^2} = \frac{A_{21}^2}{A_{21}^2 + A_{11}^2}$$

or

$$\sigma = \pm\frac{A_{21}}{\sqrt{A_{21}^2 + A_{11}^2}}.$$

Again, we choose the $+$ sign. As a result, we have

$$R_{11} = A_{11}\frac{A_{11}}{\sqrt{A_{21}^2 + A_{11}^2}} + A_{21}\frac{A_{21}}{\sqrt{A_{21}^2 + A_{11}^2}}$$
$$= \sqrt{A_{21}^2 + A_{11}^2}.$$

The matrix

$$\mathbf{Q} = \begin{bmatrix} \gamma & -\sigma \\ \sigma & \gamma \end{bmatrix}^\intercal,$$

is called a *Givens rotation*. It is called a rotation because it is orthogonal, and therefore length-preserving, and also because there is an angle θ such that $\sin\theta = \sigma$ and $\cos\theta = \gamma$, and its effect is to rotate a vector clockwise through the angle θ. The basic idea is that by an appropriately chosen θ we can always rotate a given vector into a vector whose second entry is zero. In particular,

$$\begin{bmatrix} \gamma & -\sigma \\ \sigma & \gamma \end{bmatrix}^\intercal\begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \rho \\ 0 \end{bmatrix},$$

where $\rho = \sqrt{\alpha^2 + \beta^2}$, $\alpha = \rho\cos\theta$ and $\beta = \rho\sin\theta$. It is easy to verify that the product of two rotations is itself a rotation. In fact it is not necessary to compute the angle θ itself, we always use only its sine and cosine.

Given a matrix $\mathbf{A}$, start with the first column. Take the first (i.e. k-th) nonzero element in the first column below the diagonal. By left multiplication of $\mathbf{R}(1,k,\theta)$, eliminate the k-th element, where θ is chosen appropriately. Proceed to the next nonzero element in the first column, etc. Once the first column has only zeros below the diagonal, proceed to the second column, eliminating below the diagonal with matrices $\mathbf{R}(2,k,\theta)$, $k > 3$. The key point is that the zeros in the first column remain zeros after these multiplications (notice that $\mathbf{R}(i,j,\theta)$ changes only the i-th and j-th columns and rows, hence $\mathbf{R}(2,k,\theta)$, $k > 3$ does not touch the first column). Then proceed to the third column etc.

The result is an upper triangular matrix $\mathbf{R}$ as a result of a product of Givens matrices $\mathbf{G}_N \ldots \mathbf{G}_1$:

$$\mathbf{G}_N\mathbf{G}_{N-1}, \ldots, \mathbf{G}_2\mathbf{G}_1\mathbf{A} = \mathbf{R}$$

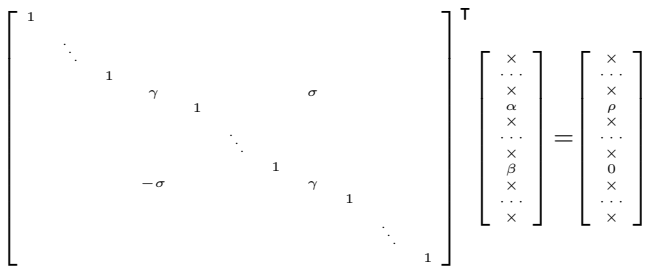
The maximal number of Givens matrices is $N = (n-1)(n-2)/2$ for an $n \times n$ square matrix $\mathbf{A}$. But then we have the QR-factorization since we can invert all Givens matrices to get $\mathbf{A} = \mathbf{QR}$ with

$$\mathbf{Q} = \mathbf{G}_1^\intercal\mathbf{G}_s^\intercal \ldots \mathbf{G}_N^\intercal.$$

Suppose we have the vector

$$[\times\ \cdots\ \times\ \alpha\ \times\ \cdots\ \times\ \beta\ \times\ \cdots\ \times]^\intercal.$$

Then



To transform $\mathbf{A}$ into an upper triangular matrix $\mathbf{R}$, we find a product of rotations $\mathbf{Q}$ such that $\mathbf{Q}^\intercal\mathbf{A} = \mathbf{R}$. It is easy to see that $\mathcal{O}(n^2)$ rotations are required. Each rotation takes $\mathcal{O}(n)$ operations, so the entire process of computing the QR factorization requires $\mathcal{O}(n^3)$ operations.

It is important to note that the straightforward approach to computing the entries γ and σ of the Givens rotation,

$$\gamma = \frac{\alpha}{\sqrt{\alpha^2+\beta^2}}, \quad \sigma = \frac{\beta}{\sqrt{\alpha+2+\beta^2}},$$

is not always advisable, because in floating-point arithmetic, the computation of $\sqrt{\alpha^2+\beta^2}$ could overflow. To get around this problem, suppose that $|\beta| \geq |\alpha|$. Then, we can instead compute

$$\tau = \frac{\alpha}{\beta}, \quad \sigma = \frac{1}{\sqrt{1+\tau^2}}, \quad \gamma = \sigma\tau,$$

which is guaranteed not to overflow since the only number that is squared is less than one in magnitude. On the other hand, if $|\beta| \leq |\alpha|$, then we compute

$$\tau = \frac{\beta}{\alpha}, \quad \gamma = \frac{1}{\sqrt{1+\tau^2}}, \quad \sigma = \gamma\tau,$$

In this section we showed how to construct Givens rotations in order to rotate two elements of a column vector so that one element would be zero, and that approximately $n^2/2$ such rotations could be used to transform $\mathbf{A}$ into an upper triangular matrix $\mathbf{R}$. Because each rotation only modifies two rows of $\mathbf{A}$, it is possible to interchange the order of rotations that affect different rows, and thus apply sets of rotations in parallel. This is the main reason why Givens rotations can be preferable to Householder reflections. Other reasons are that they are easy to use when the QR factorization needs to be updated as a result of adding a row to $\mathbf{A}$ or deleting a column of $\mathbf{A}$, and Givens rotations are more efficient than Householder when $\mathbf{A}$ is sparse.

VII. Cholesky Decomposition

Define $\mathbf{A}$ as a symmetric positive definite *square* matrix, then

$$\mathbf{A} = \mathbf{U}^\intercal\mathbf{U} = \mathbf{L}\mathbf{L}^\intercal$$

where $\mathbf{U}$ is a unique upper triangular matrix and $\mathbf{L}$ is a lower triangular matrix.

Consider the matrix $\mathbf{C} \in \mathbb{R}^{n\times n}$

$$\begin{aligned}\mathbf{C} &= \mathbf{L}\mathbf{L}^\intercal \\ &= \begin{bmatrix} L_{11} & 0 & \cdots & 0 \\ L_{21} & L_{22} & \cdots & \vdots \\ \vdots & & \ddots & 0 \\ L_{n1} & \cdots & \cdots & L_{nn} \end{bmatrix}\begin{bmatrix} L_{11} & L_{12} & \cdots & L_{1n} \\ 0 & L_{22} & \cdots & \vdots \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & 0 & L_{nn} \end{bmatrix}\end{aligned}$$

Then,

$$\begin{aligned} L_{jj} &= \sqrt{A_{jj} - \sum_{k=1}^{j-1} L_{jk}^2} \\ L_{ij} &= \frac{1}{L_{jj}}\left(A_{ij} - \sum_{k=1}^{j-1} L_{ik}L_{jk}\right), \quad i > j. \end{aligned}$$

The cost to solve the normal equations via Cholesky for large (m, n), is $mn^2 + \frac{1}{3}n^3$ flops.

1) calculate $\mathbf{C} = \mathbf{A}^\intercal\mathbf{A}$ ($\mathbf{C}$ is symmetric: $\frac{1}{2}n(n+1)(2m-1) \approx mn^2$ flops).
2) Cholesky factorization $\mathbf{C} = \mathbf{L}\mathbf{L}^\intercal$, ($\frac{1}{3}n^3$ flops).
3) calculate $\mathbf{d} = \mathbf{A}^\intercal\mathbf{b}$ ($2mn$ flops).
4) solve $\mathbf{L}\mathbf{z} = \mathbf{d}$ by forward substitution (n^2 flops).
5) solve $\mathbf{L}^\intercal\boldsymbol{x} = \mathbf{z}$ by back substitution (n^2 flops).

To exploit sparsity, we can form $\mathbf{A}^\intercal\mathbf{A}$ fast, and use a sparse Cholesky factorization (cost $\ll mn^2 + \frac{1}{3}n^3$).

VIII. Eigenvalue Decomposition (EVD)

Two forms of the Eigenvalue Decomposition exist.

Symmetric Square decomposed into squares: Assume $\mathbf{A}$ to be $n \times n$ and symmetric. Then

$$\mathbf{A} = \mathbf{V}\mathbf{D}\mathbf{V}^\intercal$$

where $\mathbf{D}$ is diagonal with the eigenvalues of $\mathbf{A}$, and $\mathbf{V}$ is orthogonal and the eigenvectors of $\mathbf{A}$.

Square decomposed into squares: Assume $\mathbf{A} \in \mathbb{R}^{n\times n}$. Then

$$\mathbf{A} = \mathbf{V}\mathbf{D}\mathbf{U}^\intercal$$

where $\mathbf{D}$ is diagonal with the square root of the eigenvalues of $\mathbf{A}\mathbf{A}^\intercal$, $\mathbf{V}$ are the eigenvectors of $\mathbf{A}\mathbf{A}^\intercal$ and $\mathbf{U}^\intercal$ are the eigenvectors of $\mathbf{A}^\intercal\mathbf{A}$.

IX. Singular Value Decomposition (SVD)

Any $n \times m$ matrix $\mathbf{A}$ can be written as

$$\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\intercal$$

where

$$\begin{aligned} \mathbf{U} &= \text{ eigenvectors of } \mathbf{A}\mathbf{A}^\intercal,\ n \times n \\ \boldsymbol{\Sigma} &= \sqrt{\text{diag}(\text{eig}(\mathbf{A}\mathbf{A}^\intercal))},\ n \times m \\ \mathbf{V} &= \text{ eigenvectors of } \mathbf{A}^\intercal\mathbf{A},\ m \times m \end{aligned}$$

Consider the matrix $\mathbf{A} \in \mathbb{R}^{p\times q}$

$$\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\intercal$$

where $\mathbf{U}\mathbf{U}^\intercal = \mathbf{I}$, and $\mathbf{V}\mathbf{V}^\intercal = \mathbf{I}$. The columns of $\mathbf{U}$ form an orthonormal basis for $\mathbf{A}$ in $\mathbb{R}^p$. The columns of $\mathbf{V}$ form

an orthonormal basis for $\mathbf{A}$ in $\mathbb{R}^q$. The diagonal matrix $\mathbf{\Sigma}$, contains the singular values of $\mathbf{A}$. This method is called the Singular Value Decomposition (SVD).

Let $\mathbf{U} \triangleq [U_1, \ldots, U_r | U_{r+1}, \ldots, U_p]$ and $\mathbf{V} \triangleq [V_1, \ldots, V_r | V_{r+1}, \ldots, V_q]$, then

- $U_{1:r}$ are an orthonormal basis for the column space of $\mathbf{A}$, i.e. $CS(\mathbf{A})$.
- $U_{(r+1):p}$ are an orthonormal basis for the left-null space of $\mathbf{A}$, i.e. $LN(\mathbf{A})$.
- $V_{1:r}$ are an orthonormal basis for the row space of $\mathbf{A}$, i.e. $RS(\mathbf{A})$.
- $V_{(r+1):q}$ are an orthonormal basis for the null space of $\mathbf{A}$, i.e. $Null(\mathbf{A})$.
- any vector in $U_{1:r} \perp U_{(r+1):p}$, i.e. $CS(\mathbf{A}) \perp LN(\mathbf{A})$
- any vector in $V_{1:r} \perp V_{(r+1):q}$, i.e. $RS(\mathbf{A}) \perp Null(\mathbf{A})$

Finally, we have that

- $CS(\mathbf{A}) \equiv$ range$(\mathbf{A}) \equiv$ image$(\mathbf{A})$.
- dim$(Null(\mathbf{A})) =$ nullity$(\mathbf{A})$.
- rank$(\mathbf{A})$ + dim$(Null(\mathbf{A})) =$ # of columns.
- dim $=$ # of linearly independent vectors in a space, which equals a basis.