LSQR and the Arnoldi Algorithm

April 29, 2020

Chapter 1

The Arnoldi Algorithm

1.1 The Inputs to the Arnoldi Algorithm

Let n be a given positive integer; $\mathbf{A}$ a given $n \times n$ matrix; and $\mathbf{q}_1$ a given $n \times 1$ column vector with unit norm in $\mathbb{R}^n$.

A tutorial example is discussed in `https://www.youtube.com/watch?v=TRktLuAktBQ`.

1.2 The First Arnoldi Iteration

The first Arnoldi iteration begins with the unit vector $\mathbf{q}_1$, and ends by constructing a second unit vector $\mathbf{q}_2$ orthonormal to $\mathbf{q}_1$. The first Arnoldi iteration goes as follows.

Form the product $\mathbf{A}\mathbf{q}_1$, and house it in $\mathbf{q}_2$:

$$\mathbf{q}_2 \leftarrow \mathbf{A}\mathbf{q}_1 \tag{1.1}$$

Resolve $\mathbf{q}_2$ into a component along $\mathbf{q}_1$ and a component perpendicular to $\mathbf{q}_1$:

$$g_{2,1} \leftarrow \mathbf{q}_2 \cdot \mathbf{q}_1 \tag{1.2a}$$

$$\mathbf{q}_2 \leftarrow \mathbf{q}_2 - g_{2,1}\mathbf{q}_1 \tag{1.2b}$$

At this point $\mathbf{q}_2$ houses the component of $\mathbf{A}\mathbf{q}_1$ that is perpendicular to $\mathbf{q}_1$. Make this $\mathbf{q}_2$ into a unit vector:

$$g_{2,2} \leftarrow \|\mathbf{q}_2\|_2 \tag{1.3a}$$

$$\mathbf{q}_2 \leftarrow \frac{\mathbf{q}_2}{g_{2,2}} \tag{1.3b}$$

It is handy to remember the computations above in this way:

$$\mathbf{A}\mathbf{q}_1 = g_{2,1}\mathbf{q}_1 + g_{2,2}\mathbf{q}_2 \tag{1.4}$$

1.3 The Second Arnoldi Iteration

The second Arnoldi iteration begins with the unit vectors $\mathbf{q}_1$ and $\mathbf{q}_2$, and ends by constructing a third unit vector $\mathbf{q}_3$ orthonormal to $\mathbf{q}_1$ and to $\mathbf{q}_1$. The second Arnoldi iteration goes as follows.

Form the product $\mathbf{A}\mathbf{q}_2$, and house it in $\mathbf{q}_3$:

$$\mathbf{q}_3 \leftarrow \mathbf{A}\mathbf{q}_2 \tag{1.5}$$

Resolve $\mathbf{q}_3$ into a component along $\mathbf{q_1}$, a component along $\mathbf{q}_2$, and a component perpendicular to both $\mathbf{q_1}$ and $\mathbf{q}_2$:

$$g_{3,1} \leftarrow \mathbf{q}_3 \cdot \mathbf{q}_1 \tag{1.6a}$$
$$\mathbf{q}_3 \leftarrow \mathbf{q}_3 - g_{3,1}\mathbf{q}_1 \tag{1.6b}$$
$$g_{3,2} \leftarrow \mathbf{q}_3 \cdot \mathbf{q}_2 \tag{1.6c}$$
$$\mathbf{q}_3 \leftarrow \mathbf{q}_3 - g_{3,2}\mathbf{q}_2 \tag{1.6d}$$

At this point $\mathbf{q}_3$ houses the component of $\mathbf{A}\mathbf{q}_2$ that is perpendicular to both $\mathbf{q}_1$ and $\mathbf{q}_2$. Make this $\mathbf{q}_3$ into a unit vector:

$$g_{3,3} \leftarrow \|\mathbf{q}_3\|_2 \tag{1.7a}$$
$$\mathbf{q}_3 \leftarrow \frac{\mathbf{q}_3}{g_{3,3}} \tag{1.7b}$$

It is handy to remember the computations above in this way:

$$\mathbf{A}\mathbf{q_2} = g_{3,1}\mathbf{q_1} + g_{3,2}\mathbf{q_2} + g_{3,3}\mathbf{q_3} \tag{1.8}$$

1.4 The Third Arnoldi Iteration

The third Arnoldi iteration begins with the unit vectors $\mathbf{q}_1$, $\mathbf{q}_2$ and $\mathbf{q}_3$, and ends by constructing a fourth unit vector $\mathbf{q}_4$ orthonormal to $\mathbf{q}_1$, to $\mathbf{q}_1$ and to $\mathbf{q}_3$. The third Arnoldi iteration goes as follows.

Form the product $\mathbf{A}\mathbf{q}_3$, and house it in $\mathbf{q}_4$:

$$\mathbf{q}_4 \leftarrow \mathbf{A}\mathbf{q}_3 \tag{1.9}$$

Resolve $\mathbf{q}_4$ into a component along $\mathbf{q_1}$, a component along $\mathbf{q}_2$, a component along $\mathbf{q}_3$ and a component perpendicular to $\mathbf{q_1}$, $\mathbf{q}_2$ and $\mathbf{q}_3$:

$$g_{4,1} \leftarrow \mathbf{q}_4 \cdot \mathbf{q}_1 \tag{1.10a}$$
$$\mathbf{q}_4 \leftarrow \mathbf{q}_4 - g_{4,1}\mathbf{q}_1 \tag{1.10b}$$
$$g_{4,2} \leftarrow \mathbf{q}_4 \cdot \mathbf{q}_2 \tag{1.10c}$$
$$\mathbf{q}_4 \leftarrow \mathbf{q}_4 - g_{4,2}\mathbf{q}_2 \tag{1.10d}$$
$$g_{4,3} \leftarrow \mathbf{q}_4 \cdot \mathbf{q}_3 \tag{1.10e}$$
$$\mathbf{q}_4 \leftarrow \mathbf{q}_4 - g_{4,3}\mathbf{q}_3 \tag{1.10f}$$

At this point $\mathbf{q}_4$ houses the component of $\mathbf{A}\mathbf{q}_3$ that is perpendicular to $\mathbf{q}_1$, $\mathbf{q}_2$ and $\mathbf{q}_3$. Make this $\mathbf{q}_4$ into a unit vector:

$$g_{4,4} \leftarrow \|\mathbf{q}_4\|_2 \tag{1.11a}$$
$$\mathbf{q}_4 \leftarrow \frac{\mathbf{q}_4}{g_{4,4}} \tag{1.11b}$$

It is handy to remember the computations above in this way:

$$\mathbf{A}\mathbf{q_3} = g_{4,1}\mathbf{q_1} + g_{4,2}\mathbf{q_2} + g_{4,3}\mathbf{q_3} + g_{4,4}\mathbf{q_4} \tag{1.12}$$

1.5 Routines

A user-supplied routine named, say, `matmul` will carry out the following steps: Eq 1.1, 1.5, 1.9, and so on.

The `BLAS` routine `ddot` will carry out the following steps efficiently: Eq 1.2a, 1.6a, 1.6c, 1.10a, 1.10c, 1.10e, and so on.

The `BLAS` routine `daxpy` will carry out the following steps efficiently: Eq 1.2b, 1.6b, 1.6d, 1.10b, 1.10d, 1.10f, and so on.

The `BLAS` routine `dnrm2` will carry out the following steps efficiently: Eq 1.3a, 1.7a, 1.11a, and so on.

The `BLAS` routine `dscal` will carry out the following steps efficiently: Eq 1.3b, 1.7b, 1.11b, and so on.

1.6 The Matrices H and Q

Eq 1.4, 1.8 and 1.12 can be written as a matrix equation in this way:

$$(\mathbf{Aq}_1, \mathbf{Aq}_2, \mathbf{Aq}_3) = (\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3) \begin{pmatrix} g_{2,1} & g_{3,1} & g_{4,1} \\ g_{2,2} & g_{3,2} & g_{4,2} \\ 0 & g_{3,3} & g_{4,3} \end{pmatrix} + (\mathbf{0}, \mathbf{0}, g_{4,4}\mathbf{q}_4) \quad (1.13)$$

The most popular notation for the matrix

$$\begin{pmatrix} g_{2,1} & g_{3,1} & g_{4,1} \\ g_{2,2} & g_{3,2} & g_{4,2} \\ 0 & g_{3,3} & g_{4,3} \end{pmatrix}$$

is **H**. Perhaps the letter H is used because the matrix is upper Hessenberg in nature, *i.e,* the upper triangle plus one subdiagonal have a lot of non-zero numbers.

For the matrix

$$(\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3)$$

two notations are popular, **Q** and **V**. Let us use **Q**. Then Eq 1.13 becomes

$$\mathbf{AQ} = \mathbf{QH} + (\mathbf{0}, \mathbf{0}, g_{4,4}\mathbf{q}_4)$$

It is also popular to denote the matrix

$$(\mathbf{0}, \mathbf{0}, g_{4,4}\mathbf{q}_4)$$

as $g_{4,4}\mathbf{q}_4\mathbf{e}_3^T$, where $\mathbf{e}_3$ is the $n \times 1$ column vector such that its 3rd entry is 1, and all the other entries are 0. With this notation, Eq 1.13 can be rewritten in this way:

$$\mathbf{AQ} = \mathbf{QH} + g_{3,3}\mathbf{q}_k\mathbf{e}_3^T$$

Those who like to use the matrix **H** also use the following symbols:

$$\begin{pmatrix} h_{1,1} & h_{1,2} & h_{1,3} \\ h_{2,1} & h_{2,2} & h_{2,3} \\ 0 & h_{3,2} & h_{3,3} \end{pmatrix} = \begin{pmatrix} g_{2,1} & g_{3,1} & g_{4,1} \\ g_{2,2} & g_{3,2} & g_{4,2} \\ 0 & g_{3,3} & g_{4,3} \end{pmatrix}$$

Of course, the foregoing equations show the matrices **H** and **Q** at the end of three Arnoldi iterations. If there are k Arnoldi iterations, the matrix **H** will be $k \times k$ and **Q** will be $n \times k$.

1.7 The Modified Gram-Schmidt Algorithm

The steps mentioned in these equations are known as the modified Gram-Schmidt algorithm: Eq 1.2a, 1.2b, 1.3a, 1.3b; Eq 1.6a, 1.6b, 1.6c, 1.6d, 1.7a, 1.7b; Eq 1.10a, 1.10b, 1.10c, 1.10d, 1.10e, 1.10f, 1.11a, 1.11b; and so on.

1.8 Breakdown

Eq 1.3b has the denominator $g_{2,2}$; Eq 1.7b the denominator $g_{3,3}$; Eq 1.11b $g_{4,4}$ Breakdown means that one of these denominators is zero.

As an example, let $g_{3,3} = 0$. It means that

$$\mathbf{A}\mathbf{q}_2 = g_{3,1}\mathbf{q}_1 + g_{3,2}\mathbf{q}_2$$

That is to say, the span of the unit vectors $\mathbf{q}_1$ and $\mathbf{q}_2$ is an invariant subspace of the matrix $\mathbf{A}$. The Arnoldi algorithm cannot proceed beyond the second iteration in this case.

The code, of course, should not test whether $g_{3,3}$ is zero; rather it should test whether $|g_{3,3}|$ is small in comparison with $g_{3,1}$ and $g_{3,2}$. Something like this would be adequate:

$$\text{if } g_{3,3}^2 \leq 0.001\left(g_{3,1}^2 + g_{3,2}^2\right) \text{ print “Breakdown”}$$

Chapter 2

The Arnoldi Algorithm on a Symmetric Matrix

2.1 The Lanczos Algorithm

When $\mathbf{A}$ is symmetric, the Arnoldi algorithm is termed the Lanczos algorithm.

2.2 Properties of the Coefficients $g_{i,j}$

Let us use the notation of Sec 1.1– Sec 1.4.

Let us consider, say, the fifth iteration of the Arnoldi algorithm. The 5th iteration computes the following quantities:

$$g_{6,1}, g_{6,2}, g_{6,3}, g_{6,4}, g_{6,5}, g_{6,6}, \mathbf{q}_6$$

The following equation holds:

$$\mathbf{A}\mathbf{q}_5 = g_{6,1}\mathbf{q}_1 + g_{6,2}\mathbf{q}_2 + g_{6,3}\mathbf{q}_3 + g_{6,4}\mathbf{q}_4 + g_{6,5}\mathbf{q}_5 + g_{6,6}\mathbf{q}_6$$

Therefore,

$$\begin{aligned} g_{6,1} &= \mathbf{A}\mathbf{q}_5 \cdot \mathbf{q}_1 \\ &= \mathbf{q}_5 \cdot \mathbf{A}^T\mathbf{q}_1 \\ &= \mathbf{q}_5 \cdot \mathbf{A}\mathbf{q}_1 \\ &= \mathbf{q}_5 \cdot (g_{2,1}\mathbf{q}_1 + g_{2,2}\mathbf{q}_2) \\ &= 0 \end{aligned}$$

In the same way, $g_{6,2}$ and $g_{6,3}$ are also 0.

Let us try $g_{6,4}$:

$$\begin{aligned} g_{6,4} &= \mathbf{A}\mathbf{q}_5 \cdot \mathbf{q}_4 \\ &= \mathbf{q}_5 \cdot \mathbf{A}^T\mathbf{q}_4 \\ &= \mathbf{q}_5 \cdot \mathbf{A}\mathbf{q}_4 \\ &= \mathbf{q}_5 \cdot (g_{5,1}\mathbf{q}_1 + g_{5,2}\mathbf{q}_2 + g_{5,3}\mathbf{q}_3 + g_{5,4}\mathbf{q}_4 + g_{5,5}\mathbf{q}_5) \\ &= g_{5,5} \end{aligned}$$

The general rules are:

1. If $i - j >= 3$, then $g_{i,j} = 0$
2. If $i - j = 2$, then $g_{i,j} = g_{i-1,i-1} = g_{j+1,j+1}$

2.3 The Tridiagonal Matrix

With the rules mentioned in Sec 2.2, the upper Hessenberg matrix $\mathbf{H}$ looks like this:

$$\mathbf{H} = \begin{pmatrix} g_{2,1} & g_{3,1} & g_{4,1} & g_{5,1} & g_{6,1} \\ g_{2,2} & g_{3,2} & g_{4,2} & g_{5,2} & g_{6,2} \\ 0 & g_{3,3} & g_{4,3} & g_{5,3} & g_{6,3} \\ 0 & 0 & g_{4,4} & g_{5,4} & g_{6,4} \\ 0 & 0 & 0 & g_{5,5} & g_{6,5} \end{pmatrix} = \begin{pmatrix} g_{2,1} & g_{2,2} & 0 & 0 & 0 \\ g_{2,2} & g_{3,2} & g_{3,3} & 0 & 0 \\ 0 & g_{3,3} & g_{4,3} & g_{4,4} & 0 \\ 0 & 0 & g_{4,4} & g_{5,4} & g_{5,5} \\ 0 & 0 & 0 & g_{5,5} & g_{6,5} \end{pmatrix}$$

That is to say, $\mathbf{H}$ is both tridiagonal and symmetric. Indeed, it is popular to change the notation from $\mathbf{H}$ to $\mathbf{T}$, T being short for "tridiagonal".

A popular notation is this:

$$\begin{aligned} \alpha_1 = g_{2,1}, \alpha_2 = g_{3,2}, \alpha_3 = g_{3,2}, \dots \\ \beta_1 = g_{2,2}, \beta_2 = g_{3,3}, \beta_3 = g_{4,4}, \dots \end{aligned}$$

With this notation, the triadiagonal and symmetric $\mathbf{H}$ looks like this:

$$\mathbf{H} = \mathbf{T} = \begin{pmatrix} \alpha_1 & \beta_1 & 0 & 0 & 0 \\ \beta_1 & \alpha_2 & \beta_2 & 0 & 0 \\ 0 & \beta_2 & \alpha_3 & \beta_3 & 0 \\ 0 & 0 & \beta_3 & \alpha_4 & \beta_4 \\ 0 & 0 & 0 & \beta_4 & \alpha_5 \end{pmatrix}$$

Chapter 3

The Arnoldi Algorithm on a Rectangular Matrix

3.1 The Augmented Matrix

Let m and n be two given positive integers. Let $\mathbf{A}$ be a given $m \times n$ matrix.

Set

$$\tilde{n} = m + n$$
$$\tilde{\mathbf{A}} = \begin{pmatrix} \mathbf{0} & \mathbf{A} \\ \mathbf{A}^T & \mathbf{0} \end{pmatrix}$$

$\tilde{\mathbf{A}}$ is a $\tilde{n} \times \tilde{n}$ matrix, and is termed the augmented matrix.

3.2 Two Styles of Picking $\mathbf{q}_1$

The Arnoldi iterations can be carried out on $\tilde{\mathbf{A}}$. Because $\tilde{\mathbf{A}}$ is symmetric, the Arnoldi iterations become Lanczos iterations, and the upper Hessenberg matrix becomes a symmetric triadiagonal matrix.

Two styles of picking the starting vector $\mathbf{q}_1$ are given in the literature. The first style is to let $\mathbf{s}_1$ be a given unit vector in $\mathbb{R}^m$, and set

$$\mathbf{q}_1 = \begin{pmatrix} \mathbf{s}_1 \\ \mathbf{0} \end{pmatrix}$$

The second style is to let $\mathbf{t}_1$ be a given unit vector in $\mathbb{R}^n$, and set

$$\mathbf{q}_1 = \begin{pmatrix} \mathbf{0} \\ \mathbf{t}_1 \end{pmatrix}$$

With either choice, $\mathbf{q}_1$ will be a unit vector in $\mathbb{R}^{\tilde{n}}$.

These two styles are termed the first and second Golub-Kahan bidiagonalization styles.

3.3 The First Arnoldi Iteration with the Augmented Matrix

Paige & Saunders, in their paper on the LSQR algorithm, prefer the first Golub-Kahan bidiagonalization style over the second. Therefore, let us pick $\mathbf{q}_1$ to be in first style. Let

$$\mathbf{q}_1 = \begin{pmatrix} \mathbf{s}_1 \\ \mathbf{0} \end{pmatrix}$$

where $\mathbf{s}_1$ is a given unit vector in $\mathbf{R}^m$. At the end of the first Arnoldi iteration, we will have a second unit vector $\mathbf{q}_2$ that will be orthonormal to $\mathbf{q}_1$.

A small point on notation. We will be speaking of these column vectors:

$$\mathbf{s}_1, \mathbf{s}_3, \mathbf{s}_5, \ldots$$
$$\mathbf{t}_2, \mathbf{t}_4, \mathbf{t}_6, \ldots$$

That is to say, the $\mathbf{s}$'s will have odd subscripts, and the $\mathbf{t}$'s will have even subscripts. (In the second Golub-Kahan style of bidiagonalization, the subscript convention will be the opposite one.)

The first Arnoldi iteration goes as follows. Form the matrix product $\mathbf{A}^T\mathbf{s}_1$ and house it in a $n \times 1$ column vector $\mathbf{w}$. This $\mathbf{w}$ serves as a work vector or temporary vector, and will be discarded at the end of this iteration.

$$\mathbf{w} \leftarrow \mathbf{A}^T\mathbf{s}_1$$

Then

$$\tilde{\mathbf{A}}\mathbf{q}_1 = \begin{pmatrix} \mathbf{0} & \mathbf{A} \\ \mathbf{A}^T & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{s}_1 \\ \mathbf{0} \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{A}^T\mathbf{s}_1 \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{w} \end{pmatrix}$$

As mentioned in connection with Eq 1.2a–1.2b, we must resolve $\tilde{\mathbf{A}}\mathbf{q}_1$ into a component along $\mathbf{q}_1$ and a component perpendicular to $\mathbf{q}_1$.

For the component along $\mathbf{q}_1$, we compute as follows:

$$g_{2,1} = \mathbf{q}_1^T \tilde{\mathbf{A}} \mathbf{q}_1 = \left(\mathbf{u}_1^T \mathbf{0}^T\right) \begin{pmatrix} \mathbf{0} \\ \mathbf{w} \end{pmatrix} = 0$$

For the component perpendicular to $\mathbf{q}_1$, we compute as follows:

$$\begin{aligned} g_{2,2}\mathbf{q}_2 &= \tilde{\mathbf{A}}\mathbf{q}_1 - g_{2,1}\mathbf{q}_1 = \begin{pmatrix} \mathbf{0} \\ \mathbf{w} \end{pmatrix} - 0\mathbf{q}_1 = \begin{pmatrix} \mathbf{0} \\ \mathbf{w} \end{pmatrix} \\ g_{2,2} &= \left\| \begin{pmatrix} \mathbf{0} \\ \mathbf{w} \end{pmatrix} \right\| = \|\mathbf{w}\| \\ \mathbf{q}_2 &= \frac{1}{g_{2,2}} \begin{pmatrix} \mathbf{0} \\ \mathbf{w} \end{pmatrix} \end{aligned}$$

The computer program does not have to house $\mathbf{q}_2$ explicitly; it only has to house $\mathbf{t}_2$:

$$\begin{aligned} \mathbf{w} &\leftarrow \mathbf{A}^T \mathbf{u}_1 \\ g_{2,2} &\leftarrow \|\mathbf{w}\| \\ \mathbf{t}_2 &\leftarrow \frac{\mathbf{w}}{g_{2,2}} \end{aligned}$$

It is handy to remember these computations in either of the following forms:

$$\begin{aligned} \tilde{\mathbf{A}}\mathbf{q}_1 &= g_{2,2}\mathbf{q}_2 \\ &\text{or} \\ \mathbf{A}^T\mathbf{s}_1 &= g_{2,2}\mathbf{t}_2 \end{aligned}$$

3.4 The Second Arnoldi Iteration with the Augmented Matrix

At the start of the second Arnoldi iteration, we have two orthonormal unit vectors $\mathbf{q}_1$ and $\mathbf{q}_2$:

$$\begin{aligned} \mathbf{q}_1 &= \begin{pmatrix} \mathbf{s}_1 \\ \mathbf{0} \end{pmatrix} \\ \mathbf{q}_2 &= \begin{pmatrix} \mathbf{0} \\ \mathbf{t}_2 \end{pmatrix} \end{aligned}$$

At the end of that iteration, we will have a third unit vector, $\mathbf{q}_3$ that is orthonormal to the first two.

The iteration goes as follows. Form the matrix product $\mathbf{A}\mathbf{t}_2$ and house it in an $m \times 1$ vector $\mathbf{w}$. This $\mathbf{w}$ serves as a work vector or temporary vector, and will be discarded at the end of this iteration.

$$\mathbf{w} \leftarrow \mathbf{A}\mathbf{v}_1$$

Then

$$\tilde{\mathbf{A}}\mathbf{q}_2 = \begin{pmatrix} \mathbf{0} & \mathbf{A} \\ \mathbf{A}^T & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{0} \\ \mathbf{t}_2 \end{pmatrix} = \begin{pmatrix} \mathbf{A}\mathbf{t}_2 \\ \mathbf{0} \end{pmatrix} = \begin{pmatrix} \mathbf{w} \\ \mathbf{0} \end{pmatrix}$$

As mentioned in connection with Eq 1.6a–1.6d, we have to resolve $\tilde{\mathbf{A}}\mathbf{q}_2$ into a component along $\mathbf{q}_1$, a component along $\mathbf{q}_2$ and a component perpendicular to both $\mathbf{q}_1$ and $\mathbf{q}_2$.

The component along $\mathbf{q}_1$ will give us the coefficient $g_{3,1}$. By the rules given in Sec 2.2, $g_{3,1} = g_{2,2}$, which got computed in the first iteration. So we are done with the component along $\mathbf{q}_1$.

For the component along $\mathbf{q}_2$, we compute as follows:

$$g_{3,2} = \mathbf{q}_2^T \tilde{\mathbf{A}}\mathbf{q}_2 = \begin{pmatrix} \mathbf{0}^T \mathbf{t}_2^T \end{pmatrix} \begin{pmatrix} \mathbf{w} \\ \mathbf{0} \end{pmatrix} = 0$$

For the component perpendicular to both $\mathbf{q}_1$ and $\mathbf{q}_2$, we compute as follows:

$$\begin{aligned} g_{3,3}\mathbf{q}_3 &= \tilde{\mathbf{A}}\mathbf{q}_2 - g_{3,1}\mathbf{q}_1 - g_{3,2}\mathbf{q}_2 \\ &= \begin{pmatrix} \mathbf{w} \\ \mathbf{0} \end{pmatrix} - g_{3,1} \begin{pmatrix} \mathbf{s}_1 \\ \mathbf{0} \end{pmatrix} - 0\mathbf{q}_2 \\ &= \begin{pmatrix} \mathbf{w} - g_{3,1}\mathbf{s}_1 \\ \mathbf{0} \end{pmatrix} \\ g_{3,3} &= \left\| \begin{pmatrix} \mathbf{w} - g_{3,1}\mathbf{s}_1 \\ \mathbf{0} \end{pmatrix} \right\| = \|(\mathbf{w} - g_{3,1}\mathbf{s}_1)\| \\ \mathbf{q}_3 &= \frac{1}{g_{3,3}} \begin{pmatrix} \mathbf{w} - g_{3,1}\mathbf{s}_1 \\ \mathbf{0} \end{pmatrix} \end{aligned}$$

The computer program does not have to house $\mathbf{q}_3$ explicitly; it only has to house $\mathbf{s}_3$:

$$\begin{aligned} \mathbf{w} &\leftarrow \mathbf{A}\mathbf{t}_2 \\ \mathbf{w} &\leftarrow \mathbf{w} - g_{3,1}\mathbf{u}_1 \\ g_{3,3} &\leftarrow \|\mathbf{w}\| \\ \mathbf{s}_3 &\leftarrow \frac{\mathbf{w}}{g_{3,3}} \end{aligned}$$

It is handy to remember these computations in either of the following forms:

$$\tilde{\mathbf{A}}\mathbf{q}_2 = g_{3,1}\mathbf{q}_1 + g_{3,3}\mathbf{q}_3$$

or

$$\mathbf{A}\mathbf{t}_2 = g_{3,1}\mathbf{s}_1 + g_{3,3}\mathbf{s}_3$$

3.5 The Third Arnoldi Iteration with the Augmented Matrix

At the start of the third Arnoldi iteration, we have three orthonormal unit vectors $\mathbf{q}_1$, $\mathbf{q}_2$ and $\mathbf{q}_3$:

$$\mathbf{q}_1 = \begin{pmatrix} \mathbf{s}_1 \\ \mathbf{0} \end{pmatrix}$$

$$\mathbf{q}_2 = \begin{pmatrix} \mathbf{0} \\ \mathbf{t}_2 \end{pmatrix}$$

$$\mathbf{q}_3 = \begin{pmatrix} \mathbf{s}_3 \\ \mathbf{0} \end{pmatrix}$$

At the end of that iteration, we will have a fourth unit vector, $\mathbf{q}_4$, that is orthonormal to the first three.

The iteration goes as follows. Form the matrix product $\mathbf{A}^T\mathbf{s}_3$ and house it in an $n \times 1$ vector $\mathbf{w}$. This $\mathbf{w}$ serves as a work vector or temporary vector, and will be discarded at the end of this iteration.

$$\mathbf{w} \leftarrow \mathbf{A}^T\mathbf{s}_3$$

Then

$$\tilde{\mathbf{A}}\mathbf{q}_3 = \begin{pmatrix} \mathbf{0} & \mathbf{A} \\ \mathbf{A}^T & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{s}_3 \\ \mathbf{0} \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{A}^T\mathbf{s}_3 \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{w} \end{pmatrix}$$

As mentioned in connection with Eq 1.10a–1.10f, we have to resolve $\tilde{\mathbf{A}}\mathbf{q}_3$ into a component along $\mathbf{q}_1$, a component along $\mathbf{q}_2$, a component along $\mathbf{q}_3$, and a component perpendicular to $\mathbf{q}_1$, $\mathbf{q}_2$ and $\mathbf{q}_3$.

The component along $\mathbf{q}_1$ will give us the coefficient $g_{4,1}$. By the rules given in Sec 2.2, $g_{4,1} = 0$. So we are done with the component along $\mathbf{q}_1$.

The component along $\mathbf{q}_2$ will give us the coefficient $g_{4,2}$. By the rules given in Sec 2.2, $g_{4,2} = g_{3,3}$, which got computed in the second iteration. So we are done with the component along $\mathbf{q}_2$.

For the component along $\mathbf{q}_3$, we compute as follows:

$$g_{4,3} = \mathbf{q}_3^T \tilde{\mathbf{A}} \mathbf{q}_3 = \left(\mathbf{s}_3^T \mathbf{0}^T\right) \begin{pmatrix} \mathbf{0} \\ \mathbf{w} \end{pmatrix} = 0$$

For the component perpendicular to $\mathbf{q}_1$, $\mathbf{q}_2$ and $\mathbf{q}_3$, we compute as follows:

$$\begin{aligned} g_{4,4}\mathbf{q}_4 &= \tilde{\mathbf{A}}\mathbf{q}_3 - g_{4,1}\mathbf{q}_1 - g_{4,2}\mathbf{q}_2 - g_{4,3}\mathbf{q}_3 \\ &= \begin{pmatrix} \mathbf{0} \\ \mathbf{w} \end{pmatrix} - 0\mathbf{q}_1 - g_{4,2} \begin{pmatrix} \mathbf{0} \\ \mathbf{t}_2 \end{pmatrix} - 0\mathbf{q}_3 \\ &= \begin{pmatrix} \mathbf{0} \\ \mathbf{w} - g_{4,2}\mathbf{t}_2 \end{pmatrix} \\ g_{4,4} &= \left\| \begin{pmatrix} \mathbf{0} \\ \mathbf{w} - g_{4,2}\mathbf{t}_2 \end{pmatrix} \right\| = \|(\mathbf{w} - g_{4,2}\mathbf{t}_2)\| \\ \mathbf{q}_4 &= \frac{1}{g_{4,4}} \begin{pmatrix} \mathbf{0} \\ \mathbf{w} - g_{4,2}\mathbf{t}_2 \end{pmatrix} \end{aligned}$$

The computer program does not have to house $\mathbf{q}_4$ explicitly; it only has to house $\mathbf{t}_4$:

$$\begin{aligned} \mathbf{w} &\leftarrow \mathbf{A}^T \mathbf{s}_3 \\ \mathbf{w} &\leftarrow \mathbf{w} - g_{4,2}\mathbf{t}_2 \\ g_{4,4} &\leftarrow \|\mathbf{w}\| \\ \mathbf{t}_4 &\leftarrow \frac{\mathbf{w}}{g_{4,4}} \end{aligned}$$

It is handy to remember these computations in either of the following forms:

$$\begin{aligned} \tilde{\mathbf{A}}\mathbf{q}_3 &= g_{4,2}\mathbf{q}_2 + g_{4,4}\mathbf{q}_4 \\ &\text{or} \\ \mathbf{A}^T\mathbf{s}_3 &= g_{4,2}\mathbf{t}_2 + g_{4,4}\mathbf{t}_4 \end{aligned}$$

3.6 The Bidiagonal Matrix

Given the computations in the preceding sections, the upper Hessenberg matrix **H** (or the tridiagonal matrix **T**) looks like this:

$$\mathbf{H} = \mathbf{T} = \begin{pmatrix} g_{2,1} & g_{2,2} & 0 & 0 & 0 \\ g_{2,2} & g_{3,2} & g_{3,3} & 0 & 0 \\ 0 & g_{3,3} & g_{4,3} & g_{4,4} & 0 \\ 0 & 0 & g_{4,4} & g_{5,4} & g_{5,5} \\ 0 & 0 & 0 & g_{5,5} & g_{6,5} \end{pmatrix}$$

$$= \begin{pmatrix} 0 & g_{2,2} & 0 & 0 & 0 \\ g_{2,2} & & g_{3,3} & 0 & 0 \\ 0 & g_{3,3} & 0 & g_{4,4} & 0 \\ 0 & 0 & g_{4,4} & 0 & g_{5,5} \\ 0 & 0 & 0 & g_{5,5} & 0 \end{pmatrix}$$

The main diagonal is made up of zeros. The superdiagonal and the subdiagonal are the only two lines that house non-zero entries; this explains the prefix "bi" in the term "Golub-Kahan bidiagonalization". (Please see Sec 4.4 as well.)

Chapter 4

The LSQR Algorithm

4.1 Aim of the LSQR Algorithm

Let m and n be two given positive integers. Let $\mathbf{A}$ be a given $m \times n$ matrix. Let $\mathbf{b}$ be a given $m \times 1$ column vector. Let $\mathbf{x}$ be a $n \times 1$ column vector; $\mathbf{x}$ is not given but is free to vary in $\mathbb{R}^n$—it is a vector-valued variable.

The LSQR algorithm by Paige & Saunders aims to minimize the objective function $\|\mathbf{b} - \mathbf{A}\mathbf{x}\|$

The paragraph above contains the sentence "Let $\mathbf{A}$ be a given $m \times n$ matrix." From the angle of writing a computer program, it is preferable to say that entity that is given is a routine named, say, `matmul` that forms matrix-vector products of the form $\mathbf{A}\mathbf{x}$; a data structure that houses $\mathbf{A}$ need not be given.

4.2 LSQR's Interest in the Arnoldi Algorithm

Let the Arnoldi algorithm be applied to the rectangular matrix $\mathbf{A}$, more precisely, to the augmented matrix associated with $\mathbf{A}$ (Section 3.1). Let an odd number of $\mathbf{q}$ vectors, say 7, be generated. (Paige & Saunders use the notation $2k + 1$ for the odd number of $\mathbf{q}$ vectors; in our example, $2k + 1 = 7$.) The 7 vectors $\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_7$ give rise to these column vectors:

$$\mathbf{s}_1, \mathbf{t}_2, \mathbf{s}_3, \mathbf{t}_4, \mathbf{s}_5, \mathbf{t}_6, \mathbf{s}_7$$

In this chapter, it will be convenient to change the notation as follows:

$$\begin{aligned}
\mathbf{u}_1 &= \mathbf{s}_1 \\
\mathbf{u}_2 &= \mathbf{s}_3 \\
\mathbf{u}_3 &= \mathbf{s}_5 \\
\mathbf{u}_4 &= \mathbf{s}_7 \\
\mathbf{v}_1 &= \mathbf{t}_2 \\
\mathbf{v}_2 &= \mathbf{t}_4 \\
\mathbf{v}_3 &= \mathbf{t}_6
\end{aligned}$$

Let us pick the starting vector $\mathbf{q}_1$ as follows:

$$\begin{aligned}
\mathbf{u}_1 &= \text{unit vector along } \mathbf{b} \\
\mathbf{s}_1 &= \mathbf{u}_1 \\
\mathbf{q}_1 &= \begin{pmatrix} \mathbf{s}_1 \\ \mathbf{0} \end{pmatrix}
\end{aligned}$$

Let us set

$$\mathbf{x} = y_1\mathbf{v}_1 + y_2\mathbf{v}_2 + y_3\mathbf{v}_3$$

where y_1, y_2, y_3 are real variables. With this setting, $\mathbf{x}$ is not free to vary over all $\mathbb{R}^n$, but merely over the subspace whose basis vectors are $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$.

Therefore

$$\begin{aligned}
\mathbf{Ax} &= y_1\mathbf{Av}_1 + y_2\mathbf{Av}_2 + y_3\mathbf{Av}_3 \\
&= y_1\mathbf{At}_2 + y_2\mathbf{At}_4 + y_3\mathbf{At}_6 \\
&= y_1\left(g_{3,1}\mathbf{s}_1 + g_{3,3}\mathbf{s}_3\right) + y_2\left(g_{5,3}\mathbf{s}_3 + g_{5,5}\mathbf{s}_5\right) + y_3\left(g_{7,5}\mathbf{s}_5 + g_{7,7}\mathbf{s}_7\right) \\
&= y_1\left(g_{3,1}\mathbf{u}_1 + g_{3,3}\mathbf{u}_2\right) + y_2\left(g_{5,3}\mathbf{u}_2 + g_{5,5}\mathbf{u}_3\right) + y_3\left(g_{7,5}\mathbf{u}_3 + g_{7,7}\mathbf{u}_4\right)
\end{aligned}$$

Also

$$\mathbf{b} = \|\mathbf{b}\|\,\mathbf{u}_1$$

Therefore

$$\begin{aligned}
\mathbf{b} - \mathbf{Ax} = \|\mathbf{b}\|\mathbf{u}_1 - y_1\left(g_{3,1}\mathbf{u}_1 + g_{3,3}\mathbf{u}_2\right) - \\
y_2\left(g_{5,3}\mathbf{u}_2 + g_{5,5}\mathbf{u}_3\right) - \\
y_3\left(g_{7,5}\mathbf{u}_3 + g_{7,7}\mathbf{u}_4\right)
\end{aligned}$$

Let us remind ourselves once more that the independent variable $\mathbf{x}$ is being restricted to vary in the three-dimensional subspace formed by the basis vectors $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$; when $\mathbf{x}$ is restricted in this way, the dependent variable $\mathbf{b} - \mathbf{Ax}$ will

vary in the four-dimensional subspace formed by the basis vectors $\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \mathbf{u}_4$. It is under these restrictions that we seek to minimize the objective function $\|\mathbf{b} - \mathbf{A}\mathbf{x}\|$

Paige & Saunders use the notation $2k+1$ for the number of $\mathbf{q}$ vectors. Therefore the subscripts of $\mathbf{u}$ vectors run from $1 \ldots k+1$; those of $\mathbf{v}$ from $1 \ldots k$; and those of y from $1 \ldots k$.

4.3 Isomorphism to Lower-Dimensional Spaces

The subspace formed by the basis vectors $\mathbf{v}_1\mathbf{v}_2, \mathbf{v}_3$ is isomorphic to $\mathbb{R}^3$. We can construct a bijective linear mapping from the subspace to $\mathbb{R}^3$ thus:

$$\begin{aligned} \mathbf{v}_1 &\text{ goes to } \mathbf{f}_1 \\ \mathbf{v}_2 &\text{ goes to } \mathbf{f}_2 \\ \mathbf{v}_3 &\text{ goes to } \mathbf{f}_3 \end{aligned}$$

where $\mathbf{f}_1, \mathbf{f}_2, \mathbf{f}_3$ denote the standard basis of $\mathbb{R}^3$.

The subspace formed by the basis vectors $\mathbf{u}_1\mathbf{u}_2, \mathbf{u}_3, \mathbf{u}_4$ is isomorphic to $\mathbb{R}^4$. We can construct a bijective linear mapping from the subspace $\mathbb{R}^4$ thus:

$$\begin{aligned} \mathbf{u}_1 &\text{ goes to } \mathbf{e}_1 \\ \mathbf{u}_2 &\text{ goes to } \mathbf{e}_2 \\ \mathbf{u}_3 &\text{ goes to } \mathbf{e}_3 \\ \mathbf{u}_4 &\text{ goes to } \mathbf{e}_4 \end{aligned}$$

where $\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, \mathbf{e}_4$ denote the standard basis of $\mathbb{R}^4$.

Under these two isomorphisms

$$\mathbf{x} \text{ goes to } \mathbf{y}, \text{ where} \tag{4.1a}$$

$$\mathbf{y} = y_1\mathbf{f}_1 + y_2\mathbf{f}_2 + y_3\mathbf{f}_3 \tag{4.1b}$$

$$\mathbf{b} - \mathbf{A}\mathbf{x} \text{ goes to } \mathbf{z}, \text{ where} \tag{4.1c}$$

$$\mathbf{z} = \|\mathbf{b}\|\mathbf{e}_1 - y_1\left(g_{3,1}\mathbf{e}_1 + g_{3,3}\mathbf{e}_2\right) - y_2\left(g_{5,3}\mathbf{e}_2 + g_{5,5}\mathbf{e}_3\right) - y_3\left(g_{7,5}\mathbf{e}_3 + g_{7,7}\mathbf{e}_4\right) \tag{4.1d}$$

As the isomorphisms preserve norms and inner products,

$$\|\mathbf{z}\| = \|\mathbf{b} - \mathbf{A}\mathbf{x}\|$$

That is to say, we can minimize $\|\mathbf{z}\|$; it would be equivalent to minimizing $\|\mathbf{b} - \mathbf{A}\mathbf{x}\|$

The minimization of $\|\mathbf{z}\|$ should be a simple task, for the variables $y_1, y_2, y_3 \in \mathbb{R}^3$, a space of much lower dimension than $\mathbb{R}^m$.

In the notation of Paige & Saunders, the two low dimensional spaces are $\mathbb{R}^k$ and $\mathbb{R}^{k+1}$. The subscript of y runs from $1 \ldots k$, so that $\|\mathbf{z}\|$ is minimized in $\mathbb{R}^k$. Assuming that $k \ll m$, minimization in $\mathbb{R}^k$ should be simpler than minimization in $\mathbb{R}^m$.

4.4 The Matrix $\mathbf{B}_k$

Eq 4.1d contains the negative of the sub-expression

$$y_1 \left(g_{3,1}\mathbf{e}_1 + g_{3,3}\mathbf{e}_2\right) + y_2 \left(g_{5,3}\mathbf{e}_2 + g_{5,5}\mathbf{e}_3\right) + y_3 \left(g_{7,5}\mathbf{e}_3 + g_{7,7}\mathbf{e}_4\right)$$

This subexpression can be expressed in terms of matrices:

$$\begin{pmatrix} g_{3,1}\mathbf{e}_1 + g_{3,3}\mathbf{e}_2 & g_{5,3}\mathbf{e}_2 + g_{5,5}\mathbf{e}_3 & g_{7,5}\mathbf{e}_3 + g_{7,7}\mathbf{e}_4 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}$$

Now the expression

$$g_{3,1}\mathbf{e}_1 + g_{3,3}\mathbf{e}_2$$

is a column vector with $g_{3,1}$ in the first row and $g_{3,3}$ in the second row; so the subexpression above is

$$\begin{pmatrix} g_{3,1} & 0 & 0 \\ g_{3,3} & g_{5,3} & 0 \\ 0 & g_{5,5} & g_{7,5} \\ 0 & 0 & g_{7,7} \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}$$

In the LSQR paper, Paige & Saunders use the notations

$$\mathbf{y}_k = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ k \text{ dimensions} \end{pmatrix}$$

$$\mathbf{B}_k = \begin{pmatrix} g_{3,1} & 0 & 0 & \ldots k \text{ columns} \\ g_{3,3} & g_{5,3} & 0 & \ldots k \text{ columns} \\ 0 & g_{5,5} & g_{7,5} & \ldots k \text{ columns} \\ 0 & 0 & g_{7,7} & \ldots k \text{ columns} \\ \vdots & & & \\ k+1 \text{ rows} & & & \ldots k \text{ columns} \end{pmatrix}$$

$\mathbf{B}_k$ is $k+1 \times k$. It has a diagonal and a subdiagonal; this is another way of understanding the presence of the prefix "bi" in the term Golub-Kahan bidiagonalization. (Please see Sec 3.6 as well.)

In terms of $\mathbf{B}_k$, the formula for $\mathbf{z}$ is

$$\mathbf{z} = \|\mathbf{b}\|\mathbf{e}_1 - \mathbf{B}_k\mathbf{y}$$

4.5 Determining the y's

It is best to consider an example. Let:

$$\begin{aligned} k &= 3 \\ \|\mathbf{b}\| &= 11 \\ \mathbf{B}_k &= \begin{pmatrix} 2 & 0 & 0 \\ 4 & 3 & 0 \\ 0 & 6 & 4 \\ 0 & 0 & 1 \end{pmatrix} \end{aligned}$$

Then

$$\mathbf{z} = \begin{pmatrix} 11 \\ 0 \\ 0 \\ 0 \end{pmatrix} - \begin{pmatrix} 2 & 0 & 0 \\ 4 & 3 & 0 \\ 0 & 6 & 4 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}$$

To determine y_1, y_2, y_3, Paige & Saunders use the QR algorithm.

4.6 The First Rotation of the QR Algorithm

In the column vector

$$\begin{pmatrix} 2 \\ 4 \\ 0 \\ 0 \end{pmatrix}$$

pick the first and second entries; we get the pair of numbers $(2, 4)$. Let us imagine the point $(2, 4) \in \mathbb{R}^2$, and convert the Cartesian coordinates to polar

coordinates; we get $\rho = 4.4721, \phi = 63.43^\circ$ Set

$$\mathbf{\Omega}_{1,2} = \begin{pmatrix} \cos(63.43^\circ) & \sin(63.43^\circ) & 0 & 0 \\ -\sin(63.43^\circ) & \cos(63.43^\circ) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$
$$= \begin{pmatrix} 0.4472 & 0.8944 & 0 & 0 \\ -0.8944 & 0.4472 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Therefore

$$\mathbf{\Omega}_{1,2}\mathbf{z} = \mathbf{\Omega}_{1,2} \begin{pmatrix} 11 \\ 0 \\ 0 \\ 0 \end{pmatrix} - \mathbf{\Omega}_{1,2} \begin{pmatrix} 2 & 0 & 0 \\ 4 & 3 & 0 \\ 0 & 6 & 4 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}$$
$$= \begin{pmatrix} 4.9192 \\ -9.8384 \\ 0 \\ 0 \end{pmatrix} - \begin{pmatrix} 4.4720 & 2.6832 & 0 \\ 0 & 1.3416 & 0 \\ 0 & 6 & 4 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}$$

Paige & Saunders use the notation $\mathbf{Q}_{1,2}$ for $\mathbf{\Omega}_{1,2}$.

4.7 The Second Rotation of the QR Algorithm

In the column vector

$$\begin{pmatrix} 2.6832 \\ 1.3416 \\ 6 \\ 0 \end{pmatrix}$$

pick the second and third entries; we get the pair of numbers $(1.3416, 6)$. Let us imagine the point $(1.3416, 6) \in \mathbb{R}^2$, and convert the Cartesian coordinates to polar coordinates; we get $\rho = 6.1482, \phi = 77.40^\circ$. Set

$$\mathbf{\Omega}_{2,3} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(77.40^\circ) & \sin(77.40^\circ) & 0 \\ 0 & -\sin(77.40^\circ) & \cos(77.40^\circ) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$
$$= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0.2182 & 0.9759 & 0 \\ 0 & -0.9759 & 0.2182 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Therefore

$$\mathbf{\Omega}_{2,3}\mathbf{\Omega}_{1,2}\mathbf{z} = \mathbf{\Omega}_{2,3}\begin{pmatrix} 4.9192 \\ -9.8384 \\ 0 \\ 0 \end{pmatrix} - \mathbf{\Omega}_{2,3}\begin{pmatrix} 4.4720 & 2.6832 & 0 \\ 0 & 1.3416 & 0 \\ 0 & 6 & 4 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}$$
$$= \begin{pmatrix} 4.9192 \\ -2.1467 \\ 9.6013 \\ 0 \end{pmatrix} - \begin{pmatrix} 4.4720 & 2.6832 & 0 \\ 0 & 6.1482 & 3.9036 \\ 0 & 6 & 0.8728 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}$$

Paige & Saunders use the notation $\mathbf{Q}_{2,3}$ for $\mathbf{\Omega}_{2,3}$.

4.8 The Third Rotation of the QR Algorithm

In the column vector

$$\begin{pmatrix} 0 \\ 3.9036 \\ 0.8728 \\ 1 \end{pmatrix}$$

pick the third and fourth entries; we get the pair of numbers $(0.8728, 1)$. Let us imagine the point $(0.8728, 1) \in \mathbb{R}^2$, and convert the Cartesian coordinates to polar coordinates; we get $\rho = 1.3273, \phi = 48.88°$. Set

$$\mathbf{\Omega}_{3,4} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \cos(48.88°) & \sin(48.88°) \\ 0 & 0 & -\sin(48.88°) & \cos(48.88°) \end{pmatrix}$$
$$= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0.6576 & 0.7534 \\ 0 & 0 & -0.7534 & 0.6576 \end{pmatrix}$$

Therefore

$$\mathbf{\Omega}_{3,4}\mathbf{\Omega}_{2,3}\mathbf{\Omega}_{1,2}\mathbf{z} = \mathbf{\Omega}_{3,4}\begin{pmatrix} 4.9192 \\ -2.1467 \\ 9.6013 \\ 0 \end{pmatrix} - \mathbf{\Omega}_{3,4}\begin{pmatrix} 4.4720 & 2.6832 & 0 \\ 0 & 6.1482 & 3.9036 \\ 0 & 0 & 0.8728 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}$$
$$= \begin{pmatrix} 4.9192 \\ -2.1467 \\ 6.3138 \\ -7.2336 \end{pmatrix} - \begin{pmatrix} 4.4720 & 2.6832 & 0 \\ 0 & 6.1482 & 3.9036 \\ 0 & 0 & 1.3274 \\ 0 & 0 & 0 \end{pmatrix}\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}$$

Paige & Saunders use the notation $\mathbf{Q}_{3,4}$ for $\mathbf{\Omega}_{3,4}$. In addition, they denote $\mathbf{\Omega}_{3,4}\mathbf{\Omega}_{2,3}\mathbf{\Omega}_{1,2}$ as $\mathbf{Q}_k$; the 3×3 submatrix

$$\begin{pmatrix} 4.4720 & 2.6832 & 0 \\ 0 & 6.1482 & 3.9036 \\ 0 & 0 & 1.3274 \end{pmatrix}$$

as $\mathbf{R}_k$; the 3×1 subvector

$$\begin{pmatrix} 4.9192 \\ -2.1467 \\ 6.3138 \end{pmatrix}$$

as $\mathbf{f}_k$; and the last entry, -7.2336, as ϕ_{k+1}.