Higham says QR factorizations are unique if full rank.

1 Floating Point Numbers and Error Analysis Tools

We will be using floating-point operation error analysis tools developed in Higham [1]. Let $\mathbb{F} \subset \mathbb{R}$ denote the space of some floating point number system with base β, precision t, significand/mantissa m, and exponent range $e_{\text{ran}} := \{e_{\min}, e_{\min}+1, \cdots, e_{\max}\}$. Then every element y in$\mathbb{F}$ can be written as

$$y = \pm m \times \beta^{e-t}, \tag{1}$$

where m is any integer in $[0, \beta^t - 1]$, and $e \in e_{\text{ran}}$. Table 1 shows IEEE precision types described by the same paramters as in Equation 1. While operations we use on $\mathbb{R}$ cannot be replicated exactly due to the finite cardinality of $\mathbb{F}$, we can still approximate the accuracy of analogous floating point operations using these error analysis tools.

Name	β	t	# of exponent bits	$e_{\min}$	$e_{\max}$
IEEE754 half	2	11	5	-15	16
IEEE754 single	2	24	8	-127	128
IEEE754 double	2	53	11	-1023	1024

Table 1: IEEE754 formats with j exponent bits range from $1 - 2^{j-1}$ to 2^{j-1}.

Suppose that a single basic floating-point operation yields a relative error, δ.

$$\text{fl}(x \text{ op } y) = (1+\delta)(x \text{ op } y), \quad |\delta| \le u, \quad \text{op} \in \{+, -, \times, \div\} \tag{2}$$

The true value $(x \text{ op } y)$ lies in $\mathbb{R}$ and it is rounded to the nearest floating point number, $\text{fl}(x \text{ op } y)$, admitting a rounding error. A short analysis (cf. Theorem 2.2 [1]) shows that the relative error $|\delta|$ is bounded by the unit round-off, $u := \frac{1}{2}\beta^{1-t}$.

We use Equation 2 as a building block in accumulating errors from successive floating point operations in product form. Lemma 1.1 introduces new notations that replace the relative error δ with θ_n and the upper bound u with $\gamma^{(n)}$, and simplifies many error analyses.

Lemma 1.1 (Lemma 3.1 [1]). *Let $|\delta_i| < u$ and $\rho_i \in \{-1, +1\}$ for $i = 1, \cdots, n$, and $nu < 1$. Then,*

$$\prod_{i=1}^{n} (1+\delta_i)^{\rho_i} = 1 + \theta^{(n)} \tag{3}$$

where

$$|\theta^{(n)}| \le \frac{nu}{1-nu} =: \gamma^{(n)}. \tag{4}$$

While the assumption $nu < \frac{1}{2}$ (which implies $\gamma^{(n)} < 1$) is easily satisfied by single and double precision types, it becomes a problem for lower precision types. Table 2 shows the maximum value of n that still guarantees a relative error below 100% ($\gamma^{(n)} < 1$). This reflects on two sources of difficulty: 1) Numerous operations in lower precision types grow less stable quickly, and 2) the upper bound given by $\gamma^{(n)}$ becomes suboptimal faster in low precision. However, error analysis within the framework given by Lemma 1.1 is still the best method and we will be using it to study variable-precision block QR factorization methods.

precision	$\text{argmax}_n(\gamma_{nu} \leq 1)$
half	512
single	$4194304 \approx 4.19 \times 10^6$
double	$2251799813685248 \approx 2.25 \times 10^{15}$

Table 2: Upper limits of validity in the γ notation.

In addition to Lemma 1.1, Lemma 3.3 in [1] summarizes other useful relations for error analysis. We present modified versions of these relations that supports multiple precision types. We distinguish between the different precision types using subscripts— these types include products (p), sums (s), and storage formats (st).

Lemma 1.2 (Modified version of Lemma 3.3 from [1])**.** *For any nonnegative integer k, and some precision q, let $\theta_k^{(q)}$ denote a quantity bounded according to $|\theta_k^{(q)}| \leq \frac{ku^{(q)}}{1-ku^{(q)}} =: \gamma_k^{(q)}$. The following relations hold for 2 precisions s and p, positive integers, j_s,j_p, non-negative integers k_s and k_p, and $c > 0$. Most of these result from commutativity.*

$$(1+\theta_p^{(k_p)})(1+\theta_p^{(j_p)})(1+\theta_s^{(k_s)})(1+\theta_s^{(j_s)}) = (1+\theta_p^{(k_p+j_p)})(1+\theta_s^{(k_s+j_s)})$$

$$\frac{(1+\theta_p^{(k_p)})(1+\theta_s^{(k_s)})}{(1+\theta_p^{(j_p)})(1+\theta_s^{(j_s)})} = \begin{cases} (1+\theta_s^{(k_s+j_s)})(1+\theta_p^{(k_p+j_p)}), & j_s \leq k_s \text{ and } j_p \leq k_p \\ (1+\theta_s^{(k_s+2j_s)})(1+\theta_p^{(k_p+j_p)}), & j_s \leq k_s \text{ and } j_p > k_p \\ (1+\theta_s^{(k_s+j_s)})(1+\theta_p^{(k_p+2j_p)}), & j_s > k_s \text{ and } j_p \leq k_p \\ (1+\theta_s^{(k_s+2j_s)})(1+\theta_p^{(k_p+2j_p)}), & j_s > k_s \text{ and } j_p > k_p \end{cases}$$

Without loss of generality, let $u_s \ll u_p$, and define $N_{s,p} := \lfloor \frac{u_p}{u_s} \rfloor$. For all integers $k_s < N_{s,p}$ (Check: $<$ or $\leq$), we result in the following set of inequalities.

$$\begin{aligned} \gamma_s^{(k_s)}\gamma_p^{(k_p)} &\leq \gamma_p^{(k_p)}, \quad \text{for } k_p u_p \leq \frac{1}{2} \\ \gamma_s^{(k_s)} + u_p &\leq \gamma_p^{(2)} \\ \gamma_p^{(k_p)} + u_s &\leq \gamma_p^{(k_p+1)} \\ \gamma_p^{(k_p)} + \gamma_s^{(k_s)} + \gamma_p^{(k_p)}\gamma_s^{(k_s)} &\leq \gamma_p^{(k_p+1)} \end{aligned}$$

2 Householder QR Backward Error Analysis

We present up the error analysis for the Householder QR factorization where all inner products are done with precision p for products, and precision s for the summation, and stored in st precision.

2.1 Revised (hopefully tighter) 2-Norm error

Consider performing an inner product with different floating point precision assigned to multiplication and addition. That is, for some $\mathbf{x}, \mathbf{y} \in \mathbb{F}_{st}^n \subset \mathbb{R}^n$, the element-wise product is done in some precision, p with eps u_p, and summation is done in precision s with eps u_s. Let u_p be the unit round-off for products, and u_s be the unit round-off for summation. We are ignoring errors that

could(?) arise from conversion between floating types. Following the notation in [1], we let δ_p and δ_s be defined as quantities that are bounded by: $|\delta_p| < u_p$ and $|\delta_s| < u_s$. Probably too much detail:

$$\begin{aligned}
\hat{s_1} &= \text{fl}(x_1 y_1) = x_1 y_1 (1 + \delta_p^{(1)}) \\
\hat{s_2} &= \text{fl}(\hat{s_1} + x_2 y_2) \\
&= \left[x_1 y_1 (1 + \delta_p^{(1)}) + x_2 y_2 (1 + \delta_p^{(2)}) \right] (1 + \delta_s^{(1)}) \\
\hat{s_3} &= \text{fl}(\hat{s_2} + x_3 y_3) \\
&= \left(\left[x_1 y_1 (1 + \delta_p^{(1)}) + x_2 y_2 (1 + \delta_p^{(2)}) \right] (1 + \delta_s^{(1)}) + x_3 y_3 (1 + \delta_p^{(3)}) \right) (1 + \delta_s^{(2)})
\end{aligned}$$

We can see a pattern emerging. We drop the superscripts as we only need to distinguish between δ_p's and δ_s's. The error for a general length n vector dot product is then:

$$\hat{s_n} = (x_1 y_1 + x_2 y_2)(1 \pm \delta_p)(1 \pm \delta_s)^{n-1} + (1 \pm \delta_p) \sum_{i=3}^{n} x_i y_i (1 \pm \delta_s)^{n-(j-1)} \tag{5}$$

Why $\pm$'s?

Using Lemma 1.1 and that $\gamma_{u,n}$ is a monotonically increasing function with respect to n (for $nu < 1$), we further simplify.

$$\begin{aligned}
\text{fl}(\mathbf{x}^\top \mathbf{y}) &= \hat{s_n} \\
&= (1 + \theta_{1,p}) \left[|x_1 y_1|(1 + \theta_{n-1,s}) + |x_2 y_2|(1 + \tilde{\theta}_{n-1,s}) + \sum_{i=3}^{n} |x_i y_i|(1 + \theta_{n-(i-1),s}) \right] \\
&= (\mathbf{x} + \Delta\mathbf{x})^\top \mathbf{y} = \mathbf{x}^\top (\mathbf{y} + \Delta\mathbf{y}), \text{where } \{\Delta\mathbf{x}, \Delta\mathbf{y}\} \leq (1 + \theta_{1,p})(1 + \gamma_{m-1,s})\{|\mathbf{x}|, |\mathbf{y}|\} \text{componetwise.} \\
&\leq (1 + \theta_{1,p}) \left[|x_1 y_1| + |x_2 y_2|(1 + \gamma_{n-1,s}) + \sum_{i=3}^{n} |x_i y_i|(1 + \gamma_{n-(i-1),s}) \right] \\
&\leq (1 + \theta_{1,p})(1 + \gamma_{n-1,s}) \sum_{i=1}^{n} |x_i y_i| = (1 + \theta_{1,p})(1 + \gamma_{n-1,s}) |\mathbf{x}|^\top |\mathbf{y}|
\end{aligned}$$

While this result does not give us a relative backward error, it can be used as a rough estimate while keeping in mind that it loses high relative accuracy as $|\mathbf{x}^\top \mathbf{y}| \ll |\mathbf{x}|^\top |\mathbf{y}|$. If $\mathbf{y} = \mathbf{x}$, we have exactly $|\mathbf{x}^\top \mathbf{x}| = |\mathbf{x}|^\top |\mathbf{x}| = \|\mathbf{x}\|_2^2$. So we get ~~(CHECK: first inequality)~~

$$(1 - u_{1,p})(1 - \gamma_{n-1,s}) \|\mathbf{x}\|_2^2 \leq \text{fl}(\|\mathbf{x}\|_2^2) \leq (1 + u_{1,p})(1 + \gamma_{n-1,s}) \|\mathbf{x}\|_2^2,$$

which gives us:

$$\text{fl}(\|\mathbf{x}\|_2^2) = (1 + \theta_{1,p})(1 + \theta_{n-1,s}) \|\mathbf{x}\|_2^2$$

Then,

$$\begin{aligned}
\text{fl}(\|\mathbf{x}\|_2) &= \text{fl}(\sqrt{\|\mathbf{x}\|_2^2}) \\
&= (1 + \theta_{st}) \sqrt{(1 + \theta_{1,p})(1 + \theta_{n-1,s})} \|\mathbf{x}\|_2,
\end{aligned}$$

where θ_{st} represents the rounding error that results from the square root operation. Taylor expansion analysis shows that $\sqrt{1+\theta^{(n)}} = 1 + \frac{5}{8}\theta_n$.

For small $|\epsilon| \ll 1$,

$$\sqrt{1+\epsilon} = 1 + \frac{1}{2}\epsilon + \sum_{i=2}^{\infty}\left[\frac{1}{i!}\epsilon^i \prod_{j=1}^{i}\left(\frac{1}{2} - (j-1)\right)\right]$$

The series can also be written in recursive form: $a_1 = (1 + \frac{1}{2}\epsilon)$ and $a_n = a_{n-1}\frac{\epsilon}{n}\left(\frac{1}{2} - (n-1)\right)$ for $n \geq 2$. As this is an alternating series for any $\epsilon < 1$, we can bound the partial sum.

$$\begin{aligned}
\left|\sqrt{1+\epsilon} - (1 + \frac{1}{2}\epsilon)\right| &\leq \frac{1}{8}\epsilon^2 \\
\frac{1}{2}\epsilon - \frac{1}{8}\epsilon^2 \leq \sqrt{1+\epsilon} - 1 &\leq \frac{1}{2}\epsilon + \frac{1}{8}\epsilon^2 \\
-\frac{1}{2}|\epsilon| - \frac{1}{8}\epsilon^2 \leq \sqrt{1+\epsilon} - 1 &\leq \frac{1}{2}|\epsilon| + \frac{1}{8}\epsilon^2 \\
\left|\sqrt{1+\epsilon} - 1\right| &\leq \frac{1}{2}|\epsilon| + \frac{1}{8}\epsilon^2
\end{aligned}$$

Applying this result to $\theta^{(n)}$, we get:

$$\begin{aligned}
|\sqrt{1+\theta^{(n)}} - 1| &\leq \frac{1}{2}\gamma^{(n)} + \frac{1}{8}\gamma^{(n)}\gamma^{(n)} \leq \frac{5}{8}\gamma^{(n)} \\
\sqrt{1+\theta^{(n)}} &= 1 + \frac{5}{8}\theta^{(n)}
\end{aligned}$$

This is a slight improvement compared to the analysis in [1], which used a looser bound of: $\sqrt{1+\theta^{(n)}} = 1 + \theta^{(n)}$. Now, the largest possible n such that the relative error $\frac{5}{8}\gamma^{(n)} < 1$ for an arbitrary precision with unit round-off u is $\lfloor \frac{8}{13}\frac{1}{u} \rfloor$.

Thus, the 2-norm error derived from using multiple precision dot products is:

$$\text{fl}(\|\mathbf{x}\|_2) = (1+\theta_{st})(1+\frac{5}{8}\theta_p)(1+\frac{5}{8}\theta_{n-1,s})\|\mathbf{x}\|_2. \tag{6}$$

2.2 Calculation and normalization of Householder Vector

The Householder QR factorization utilizes Householder transformations to zero out elements below the diagonal of a matrix. We consider the simpler task of zeroing out all but the first element of a vector, $\mathbf{x} \in \mathbb{R}^n$. Then, the appropriate Householder vector is defined by: $\mathbf{v} := \mathbf{x} + \text{sign}(\text{x}_1)\|\mathbf{x}\|_2\hat{\text{e}}_1$. An efficient algorithm for calculating $\mathbf{v}$ is shown in Algorithm 1.

This algorithm leaves $\mathbf{v}$ unnormalized, but it is often normalized via the various methods and reasons listed below:

1. Set v_1 to 1 for efficient storage of many Householder vectors.
2. Set the 2-norm of $\mathbf{v}$ to $\sqrt{2}$ to always have $\beta = 1$.
3. Set the 2-norm of $\mathbf{v}$ to 1 to prevent extremely large values, and to always have $\beta = 2$.

Algorithm 1: Given a vector $\mathbf{x} \in \mathbb{R}^n$, return a Householder vector v and a Householder constant β such that $(I - \beta \mathbf{v}\mathbf{v}^\top)\mathbf{x} \in \text{span}(\hat{e_1})$.

Input: $\mathbf{x} \in \mathbb{R}^n$

Output: $\mathbf{v} \in \mathbb{R}^n$, and $\sigma, \beta \in \mathbb{R}$ such that $(I - \beta \mathbf{v}\mathbf{v}^\top)\mathbf{x} = \pm\|\mathbf{x}\|_2 \hat{e_1} = \sigma \hat{e_1}$

/* We choose the sign of sigma to avoid cancellation of x_1 (As is the standard in LAPACK, LINPACK packages [1]). This makes $\beta > 0$. */

$\mathbf{v} \leftarrow \mathbf{x}$

$\sigma \leftarrow -\text{sign}(\text{x}_1)\|\mathbf{x}\|_2$

$v_1 \leftarrow v_1 - \sigma$

$\beta \leftarrow -\frac{1}{\sigma v_1}$

return β, $\mathbf{v}$

The first normalizing method adds an extra rounding error to β and $\mathbf{v}$ each, whereas the remaining methods only add an extra rounding error to $\mathbf{v}$ since we set β to what it should be exactly. The LINPACK implementation of the Householder QR factorization uses CHECK! the first method of normalizing via setting v_1 to 1. Algorithm 2 shows how this convention could be carried out. The error analysis in the subsequent section assumes that there may exist errors in both β and $\mathbf{v}$ to get the worse-case scenario and to be consistent with the LINPACK implementation.

Algorithm 2: Given a vector $\mathbf{x} \in \mathbb{R}^n$, return the portion of the Householder vector $\mathbf{v}$ without the first component, and a Householder constant β such that $(I - \beta \mathbf{v}\mathbf{v}^\top)\mathbf{x} \in \text{span}(\hat{e_1})$, and $v_1 = 1$.

Input: $\mathbf{x} \in \mathbb{R}^n$

Output: $\mathbf{v} \in \mathbb{R}^n$, and $\sigma, \beta \in \mathbb{R}$ such that $(I - \beta \mathbf{v}\mathbf{v}^\top)\mathbf{x} = \pm\|\mathbf{x}\|_2 \hat{e_1} = \sigma \hat{e_1}$

/* We choose the sign of sigma to avoid cancellation of x_1 (As is the standard in LAPACK, LINPACK packages [1]). This makes $\beta > 0$. */

$\mathbf{v} \leftarrow \mathbf{x}$[2:end]

$\sigma \leftarrow -\text{sign}(\text{x}_1)\|\mathbf{x}\|_2$

$v_1 \leftarrow x_1 - \sigma$ // This v_1 is still not normalized.

$\beta \leftarrow -\frac{1}{\sigma v_1^2}$

$\mathbf{v} \leftarrow \frac{1}{v_1}\mathbf{v}$

return β, $\mathbf{v}$

2.2.1 Error analysis for v_1

This algorithm only introduces error in v_1.

$$\begin{aligned}
\text{fl}(\mathbf{x}^\top \mathbf{x}) &\leq (1+u_p)(1+\gamma_{n-1,s})\mathbf{x}^\top \mathbf{x} \\
\text{fl}(\sigma) &= \text{fl}(-\text{sign}(\text{x}_1)\|\mathbf{x}\|_2) \\
&= -\text{sign}(\text{x}_1)\sqrt{\text{fl}(\mathbf{x}^\top \mathbf{x})} \\
&= (1+\theta_{st})\sqrt{(1+\theta_{1,p})(1+\theta_{n-1,s})}\sigma \\
&= (1+\theta_{st})(1+\frac{5}{8}\theta_p)(1+\frac{5}{8}\theta_{n-1,s})\sigma \\
\text{fl}(v_1) &= \text{fl}(x_1 - \sigma) = (1+\delta_{st})(x_1 - \hat{\sigma}) \\
&= (1+\theta_{2,st})\sqrt{(1+\theta_{1,p})(1+\theta_{n-1,s})}v_1 \\
&= (1+\theta_{2,st})(1+\frac{5}{8}\theta_p)(1+\frac{5}{8}\theta_{n-1,s})v_1
\end{aligned}$$

The above equalities are permitted since we are using arbitrary θ values.

2.2.2 Error analysis for β

$$\begin{aligned}
\hat{\beta} = \text{fl}(\frac{1}{\text{fl}(\hat{\sigma}\hat{v_1})}) &= -\frac{1+\delta_{st}}{(1+\delta_{st})\left[(1+\theta_{3,st})\sqrt{(1+\theta_{1,p})(1+\theta_{n-1,s})}^2\sigma v_1\right]} \\
&= \frac{1+\theta_{9,st}}{(1+\theta_{1,p})(1+\theta_{n-1,s})}\beta \\
&= (1+\theta_{9,st})(1+\theta_{2,p})(1+\theta_{2n-2,s})\beta
\end{aligned}$$

2.3 Error analysis for a single Householder transformation to a vector

We never form the Householder matrix. Instead we apply a series of inner and outer products, since the matrix is a rank-1 update of the identity. For some $P_\mathbf{v} = I - \beta\mathbf{v}\mathbf{v}^\top$, we result in the following computation.

$$P_\mathbf{v}\mathbf{x} = (I - \beta\mathbf{v}\mathbf{v}^\top)\mathbf{x} = \mathbf{x} - (\beta\mathbf{v}^\top\mathbf{x})\mathbf{v} \tag{7}$$

Case 1: Applying $P_\mathbf{v}$ to zero out vector
Recall that we chose a specific $\mathbf{v}$ such that $P_\mathbf{v}\mathbf{x} = \sigma\hat{e}_1$. Therefore, the only error lies in the first element, σ.

Case 2: Applying $P_\mathbf{v}$ to the rest of the matrix (x has no relationship to v.)
LOOK AT LEMMA 3.9

$$\begin{aligned}
\text{fl}(P_\mathbf{v}\mathbf{x}) &= \text{fl}(\mathbf{x} - \text{fl}(\text{fl}(\mathbf{v}^\top\mathbf{x})\mathbf{v})) \\
&\text{For each component:}(i = 1:n) \\
\text{fl}(P_\mathbf{v}\mathbf{x})_i &= (1+\delta_{st})(1+\delta_{st})(x_i - \text{fl}(\mathbf{v}\top\mathbf{x})v_i)
\end{aligned}$$

Recall

2.4 Fused Multiply-Add Operation

Refer to section 2.6 of [1] and https://developer.download.nvidia.com/assets/cuda/files/NVIDIA-CUDA-Floating-Point.pdf

References

[1] Nicholas J. Higham. Accuracy and Stability of Numerical Algorithms. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2nd edition, 2002. ISBN 0898715210.