

1. BLAS-3 Implementation of HQR for TensorCore Technology Assumptions. I think it would be very suitable if we add another block Householder QR algorithm for this paper. We already have TSQR, which partitions the rows. Partitioning the columns of a matrix is actually the better known “block” HQR algorithm that can employ BLAS-3 operations for the majority of FLOPs required. I picked out the main details from section Chapter 5 of [1].

1.1. Algorithms.

1.1.1. The WY Representation. A convenient matrix representation that accumulates r Householder reflectors is known as the WY representation.

LEMMA 1.1. Suppose $\mathbf{Q} = \mathbf{I}_m - \mathbf{W}\mathbf{Y}^\top \in \mathbb{R}^{m \times m}$ is an orthogonal matrix with $\mathbf{W}, \mathbf{Y} \in \mathbb{R}^{m \times j}$. If $\mathbf{P} = \mathbf{I}_m - \beta \mathbf{v}\mathbf{v}^\top$ with $\mathbf{v} \in \mathbb{R}^m$ and $\mathbf{z} = \beta \mathbf{Q}\mathbf{v}$, then

$$\mathbf{Q}_+ = \mathbf{Q}\mathbf{P} = \mathbf{I} - \mathbf{W}_+\mathbf{Y}_+^\top,$$

where $\mathbf{W}_+ = [\mathbf{W} | \mathbf{z}]$ and $\mathbf{Y}_+ = [\mathbf{Y} | \mathbf{v}]$ are each m -by- $(j+1)$.

If \mathbf{Q} was already the accumulation of j Householder transformations, then Lemma 1.1 shows us a clever way to build the WY representation of successive Householder transformations. Let us now show the proof for Lemma 1.1.

Proof. A direct right multiplication of $\mathbf{P} := \mathbf{I}_m - \beta \mathbf{v}\mathbf{v}^\top$ onto \mathbf{Q} can be written as

$$\mathbf{Q}\mathbf{P} = \mathbf{Q} - \beta \mathbf{Q}\mathbf{v}\mathbf{v}^\top.$$

Let us use the WY representation of \mathbf{Q} .

$$\mathbf{Q}\mathbf{P} = \mathbf{I}_m - \mathbf{W}\mathbf{Y}^\top - \beta \mathbf{Q}\mathbf{v}\mathbf{v}^\top = \mathbf{I}_m - \mathbf{W}\mathbf{Y}^\top - \mathbf{z}\mathbf{v}^\top$$

Now note that the two subtracted terms are exactly the updated WY factors:

$$\mathbf{W}_+\mathbf{Y}_+^\top = [\mathbf{W} \quad \mathbf{z}] \begin{bmatrix} \mathbf{Y}^\top \\ \mathbf{v}^\top \end{bmatrix} = \mathbf{W}\mathbf{Y}^\top + \mathbf{z}\mathbf{v}^\top. \quad \square$$

With the correct initialization of \mathbf{W} and \mathbf{Y} , we can build the WY representation of successive Householder transformations as shown in Algorithm 1.

In the traditional HQR, \mathbf{A} is transformed into an upper triangular matrix \mathbf{R} by first computing the Householder transformation to zero out a column below the diagonal, then applying that Householder transformation to all of the remaining columns to the right. For example, the k^{th} Householder transformation finds an $m - k + 1$ length Householder vector, \mathbf{v}_k , and applies it to an $(m - k + 1)$ -by- $(n - k)$ matrix. The bulk FLOPs of the latter step requires two Level-2 BLAS operations when computed efficiently. The first is $\mathbf{C} := \mathbf{v}_k^\top \mathbf{A}_{k:m, k+1:n} \in \mathbb{R}^{1 \times (n-k)}$ and the second is $\mathbf{v}\mathbf{C}$, an outer product.

In Block HQR, the columns of \mathbf{A} are partitioned by groups of r with $\mathbf{A} = [\mathbf{C}_1 \cdots \mathbf{C}_N]$ except for the last block which is $\mathbf{C}_N = \mathbf{A}[:, (N-1)r+1 : n]$ and $N = \lceil \frac{n}{r} \rceil$. The first block is triangularized using HQR and the WY representation of $\mathbf{P}_1 \cdots \mathbf{P}_r = \mathbf{I}_m - \mathbf{W}_1\mathbf{Y}_1^\top$ is built at the end, with Level-2 BLAS operations. Then, $\mathbf{I}_m - \mathbf{Y}_1\mathbf{W}_1^\top = \mathbf{P}_r \cdots \mathbf{P}_1$ is applied to $[\mathbf{C}_2 \cdots \mathbf{C}_N]$ with two Level-3 BLAS operations. First, $\mathbf{W}_1^\top [\mathbf{C}_2 \cdots \mathbf{C}_N]$ is a matrix-matrix multiply with m -length inner products, and second, the result of left multiplying that by \mathbf{Y}_1 (r -length inner products) is subtracted from $[\mathbf{C}_2 \cdots \mathbf{C}_N]$. We are now ready to triangularize the second block and update rows $r+1 : m$ of $[\mathbf{C}_3 \cdots \mathbf{C}_N]$, and so on. Algorithm 2 shows the pseudoalgorithm of the described procedure and performs approximately $1 - \mathcal{O}(1/N)$ fraction of FLOPs in Level-3 BLAS operations (see [1]).

Algorithm 1: $\mathbf{W}, \mathbf{Y} \leftarrow \text{buildWY}(\mathbf{V}, \boldsymbol{\beta})$: Given a set of householder vectors $\{\mathbf{V}[:, i]\}_{i=1}^r$ and their corresponding constants $\{\beta_i\}_{i=1}^r$, form the final \mathbf{W} and \mathbf{Y} factors of the WY representation of $\mathbf{P}_1 \cdots \mathbf{P}_r$, where $\mathbf{P}_i := \mathbf{I}_m - \beta_i \mathbf{v}_i \mathbf{v}_i^\top$

Input: $\mathbf{V} \in \mathbb{R}^{m \times r}$, $\boldsymbol{\beta} \in \mathbb{R}^r$ where $m > r$.
Output: \mathbf{W}, \mathbf{Y}

```

1 Initialize:  $\mathbf{W} := \beta_1 \mathbf{V}[:, 1]$  and  $\mathbf{Y} := \mathbf{V}[:, 1]$ .
2 for  $j = 2 : r$  do
3    $\mathbf{z} \leftarrow \beta_j [\mathbf{V}[:, j] - \mathbf{W} (\mathbf{Y}^\top \mathbf{V}[:, j])]$ 
4    $\mathbf{W} \leftarrow [\mathbf{W} \quad \mathbf{z}]$  /* Update  $\mathbf{W}$ . */
5    $\mathbf{Y} \leftarrow [\mathbf{Y} \quad \mathbf{V}[:, j]]$  /* Update  $\mathbf{Y}$ . */
   //  $\mathbf{W}$  and  $\mathbf{Y}$  are now  $m$ -by- $j$  matrices.
6 return  $\mathbf{W}, \mathbf{Y}$ 
```

Algorithm 2: $\mathbf{Q}, \mathbf{R} \leftarrow \text{blockHQR}(\mathbf{A}, r)$: Perform Householder QR factorization of matrix \mathbf{A} with column partitions of size r .

Input: $\mathbf{A} \in \mathbb{R}^{m \times n}$, $r \in \mathbb{R}$ where $r < n$.
Output: \mathbf{Q}, \mathbf{R}

```

1  $N = \lceil \frac{n}{r} \rceil$ 
   // Let  $n_i = ri$  for  $i = 1 : N - 1$  and  $n_N = n$ .
2 for  $i = 1 : N$  do
3    $\mathbf{V}_i, \beta_i, \mathbf{A}_{n_{i-1}+1:m, n_{i-1}+1:n_i} \leftarrow \text{hhQR}(\mathbf{A}_{n_{i-1}+1:m, n_{i-1}+1:n_i})$  /* Algorithm 5 */
4    $\mathbf{W}_i, \mathbf{Y}_i \leftarrow \text{buildWY}(\mathbf{V}_i, \beta_i)$  /* Algorithm 1 */
5   if  $i < N$  then
6      $\mathbf{A}_{n_i+1:m, n_i+1:n} = \mathbf{Y}_i (\mathbf{W}_i^\top \mathbf{A}_{n_i+1:m, n_i+1:n})$  /* update the rest: BLAS-3 */
   //  $\mathbf{A}$  has been transformed into  $\mathbf{R} = \mathbf{Q}^\top \mathbf{A}$ .
   // Now build  $\mathbf{Q}$ .
7  $\mathbf{Q} \leftarrow \mathbf{I}$  /*  $\mathbf{I}_m$  if full QR, and  $\mathbf{I}_{m \times n}$  if thin QR. */
8 for  $i = N : -1 : 1$  do
9    $\mathbf{Q}_{n_{i-1}+1:m, n_{i-1}+1:n} = \mathbf{W}_i (\mathbf{Y}_i^\top \mathbf{Q}_{n_{i-1}+1:m, n_{i-1}+1:n})$  /* BLAS-3 */
10 return  $\mathbf{Q}, \mathbf{A}$ 
```

40 **1.2. Analysis.** Things (I think) I need to work on:

- 41 • Simplify analysis by using $\tilde{\gamma}_n$ notation and only keep track of leading order stuff.
- 42 • asdf
- 43 • asdf

44 Now that we have discussed the block HQR, let's now set the assumptions for our mixed-
 45 precision analysis. I will consider two cases, the first is where \mathbf{A} is cast down to the lower precision
 46 after updating each block. (i.e. After line 5 within the forloop in [alg. 2](#), and the [second is where](#)
 47 [casting down only happens at the end of the factorization](#). Everything will be done in the higher
 48 precision except for the cast down operations.

49 Recall that an m -length inner product results in a relative error bounded by γ_m . Therefore, if
 50 $\mathbf{C} = \mathbf{A}\mathbf{B}$ where $\mathbf{A} \in \mathbb{R}^{m \times p}$, $\mathbf{B} \in \mathbb{R}^{p \times n}$, each element of \mathbf{C} has accumulated rounding errors bounded

51 by γ_p .

52 **1.2.1. Round to lower precision at the end of factorization.** Since we're not considering
 53 casting down until the very end, this is a uniform precision analysis.

- 54 • The i^{th} block (for $i = 1 : N - 1$), goes through $(i - 1)$ WY updates (line 5 in [alg. 2](#)), and
 55 then is triangularized via [Algorithm 5](#).
- 56 • Let $m_i = m - r(i - 1)$. The WY factors at block i are m_i -by- r sized.
- 57 • Therefore, the update itself should accumulate errors bounded by $\tilde{\gamma}_{m_i+r}$.
- 58 • $m_i + r = m - r(i - 1) + r = m - ri + 2r = m - r(i - 2) = m_{i-1}$
- 59 • How much rounding errors are accumulated for forming the WY factor?
 60 – The largest error bound (componentwise) from forming the WY factors from the
 61 Householder constant and vectors is $\tilde{\gamma}_{m_k+r}$.
- 62 • The Householder constant and vectors calculated during the triangularization (line 3 in
 63 [alg. 2](#)) accumulated errors bounded by $\tilde{\gamma}_{m_k}$. This bound could be tighter, but I think
 64 it's best to keep the analysis block-wise. So, $\hat{\beta}_k = \beta_k + \Delta\beta_k$, $\hat{\mathbf{V}}_k = \mathbf{V}_k + \Delta\mathbf{V}_k$, where
 65 $|\Delta\beta_k| \leq \tilde{\gamma}_{m_k}|\beta_k|$ and $|\Delta\mathbf{V}_k| \leq \tilde{\gamma}_{m_k}|\mathbf{V}_k|$. Here the subscripts are boldfaced because they
 66 refer to all of the Householder constants and vectors formed from the k^{th} block.
- 67 • The triangularization itself also accumulated rounding errors bounded by $r\tilde{\gamma}_{m_k}$. This is
 68 from applying $\mathbf{P}_k = \mathbf{P}_{k,r} \cdots \mathbf{P}_{k,1}$ to the r^{th} column of block k , assuming that that column
 69 was exact.
- 70 • Let's use $\mathbf{R}_i = \mathbf{A}_{n_i+1:m, n_{i-1}+1:n_i}$ to simplify notation. Recall $m_i = m - r(i - 1) + 1$, $n_i = ri$
 71 for $i = 1 : N - 1$, and $n_N = n$.

72 **The first block:** This block only goes through r Householder transformations. Note that
 73 $m_1 = m$.

$$74 \quad \boxed{\hat{\mathbf{R}}_1 = \mathbf{R}_1 + \Delta\mathbf{R}_1, \text{ where } |\Delta\mathbf{R}_1| \leq r\tilde{\gamma}_{m_1}|\mathbf{R}_1|}$$

75 **The second block:** This block was transformed via $\mathbf{I}_m - \hat{\mathbf{Y}}_1 \hat{\mathbf{W}}_1^\top$, where

$$76 \quad \begin{aligned} \hat{\mathbf{W}}_1 &= \mathbf{W}_1 + \Delta\mathbf{W}_1, & |\Delta\mathbf{W}_1| &\leq \tilde{\gamma}_{m_1+r}|\mathbf{W}_1| \\ \hat{\mathbf{Y}}_1 &= \mathbf{Y}_1 + \Delta\mathbf{Y}_1, & |\Delta\mathbf{Y}_1| &\leq \tilde{\gamma}_{m_1+r}|\mathbf{Y}_1|. \end{aligned}$$

79 The action of applying this transformation also accumulates relative rounding error bounded by
 80 $\tilde{\gamma}_{m_1+r}$. Finally, this block goes through r Householder transformations of length $m_2 = m - r$.

$$81 \quad \boxed{\hat{\mathbf{R}}_2 = \mathbf{R}_2 + \Delta\mathbf{R}_2, \text{ where } |\Delta\mathbf{R}_2| \leq [(1 + r\tilde{\gamma}_{m_2})(1 + \tilde{\gamma}_{m_1+r}) - 1]|\mathbf{R}_2|}$$

82 **The third block:** This block was transformed via $(\mathbf{I}_m - \hat{\mathbf{Y}}_2 \hat{\mathbf{W}}_2^\top)(\mathbf{I}_m - \hat{\mathbf{Y}}_1 \hat{\mathbf{W}}_1^\top)$, where

$$83 \quad \begin{aligned} \hat{\mathbf{W}}_2 &= \mathbf{W}_2 + \Delta\mathbf{W}_2, & |\Delta\mathbf{W}_2| &\leq \tilde{\gamma}_{m_1+r+m_2+r}|\mathbf{W}_2| \\ \hat{\mathbf{Y}}_2 &= \mathbf{Y}_2 + \Delta\mathbf{Y}_2, & |\Delta\mathbf{Y}_2| &\leq \tilde{\gamma}_{m_1+r+m_2+r}|\mathbf{Y}_2|. \end{aligned}$$

86 The action of applying this transformation also accumulates relative rounding error bounded by
 87 $\tilde{\gamma}_{m_2+r}$. Finally, this block goes through r Householder transformations of length $m_3 = m - 2r$.

$$88 \quad \boxed{\hat{\mathbf{R}}_3 = \mathbf{R}_3 + \Delta\mathbf{R}_3, \text{ where } |\Delta\mathbf{R}_3| \leq [(1 + r\tilde{\gamma}_{m_3})(1 + \tilde{\gamma}_{m_1+r+m_2+r}) - 1]|\mathbf{R}_3|}$$

(Can we see a pattern yet?) **The i^{th} block:** This block was transformed via

$$(\mathbf{I}_m - \mathbf{Y}_{i-1}^{\hat{}} \mathbf{W}_{i-1}^{\hat{}}{}^{\top}) \cdots (\mathbf{I}_m - \mathbf{Y}_1^{\hat{}} \mathbf{W}_1^{\hat{}}{}^{\top}),$$

89 where

$$90 \quad \mathbf{W}_{i-1}^{\hat{}} = \mathbf{W}_{i-1} + \Delta \mathbf{W}_{i-1}, \quad |\Delta \mathbf{W}_{i-1}| \leq \tilde{\gamma}_{m_1 + \cdots + m_{i-1} + r(i-1)} |\mathbf{W}_{i-1}|$$

$$91 \quad \mathbf{Y}_{i-1}^{\hat{}} = \mathbf{Y}_{i-1} + \Delta \mathbf{Y}_{i-1}, \quad |\Delta \mathbf{Y}_{i-1}| \leq \tilde{\gamma}_{m_1 + \cdots + m_{i-1} + r(i-1)} |\mathbf{Y}_{i-1}|.$$

93 What is this crazy sum : $m_1 + \cdots + m_{i-1} + r(i-1)$??

$$94 \quad m_1 + \cdots + m_{i-1} + r(i-1) = r(i-1) + \sum_{k=1}^{i-1} m_k = r(i-1) + \sum_{k=1}^{i-1} (m - (k-1)r)$$

$$95 \quad = (m+r)(i-1) + r \sum_{k=1}^{i-1} (k-1) = (m+r-r)(i-1)r \sum_{k=1}^{i-1} k$$

$$96 \quad = m(i-1) + r \frac{i(i-1)}{2} = (m + ri/2)(i-1) \cong mi + ri^2/2$$

98 The action of applying this transformation also accumulates relative rounding error bounded
99 by $\tilde{\gamma}_{m_{i-1}+r}$. We can kind of drop this one since $m_{i-1} < m_{i-2}, \dots, m_1 = m$, and should be swept
100 under $\tilde{\gamma}_{mi+ri^2/2}$. Finally, this block goes through r Householder transformations of length m_i .

$$101 \quad \boxed{\hat{\mathbf{R}}_i = \mathbf{R}_i + \Delta \mathbf{R}_i, \text{ where } |\Delta \mathbf{R}_i| \leq [(1 + r\tilde{\gamma}_{m_i})(1 + \tilde{\gamma}_{mi+ri^2/2}) - 1] |\mathbf{R}_i|}$$

102 Let's now consider the last block.

103 **The $(N)^{\text{th}}$ block:**

$$104 \quad m_1 + \cdots + m_{N-1} + r(N-1) = (m + r(N)/2)(N-1) \cong mN + rN^2/2 \cong (m + n/2)N$$

105 Note that since $N = \lceil n/r \rceil$, $rN \approx n$.

This block was transformed via

$$(\mathbf{I}_m - \mathbf{Y}_{N-1}^{\hat{}} \mathbf{W}_{N-1}^{\hat{}}{}^{\top}) \cdots (\mathbf{I}_m - \mathbf{Y}_1^{\hat{}} \mathbf{W}_1^{\hat{}}{}^{\top}),$$

106 where

$$107 \quad \mathbf{W}_{N-1}^{\hat{}} = \mathbf{W}_{N-1} + \Delta \mathbf{W}_{N-1}, \quad |\Delta \mathbf{W}_{N-1}| \leq \tilde{\gamma}_{(m+n/2)N} |\mathbf{W}_{N-1}|$$

$$108 \quad \mathbf{Y}_{N-1}^{\hat{}} = \mathbf{Y}_{N-1} + \Delta \mathbf{Y}_{N-1}, \quad |\Delta \mathbf{Y}_{N-1}| \leq \tilde{\gamma}_{(m+n/2)N} |\mathbf{Y}_{N-1}|.$$

110 The action of applying this transformation also accumulates relative rounding error bounded by
111 $\tilde{\gamma}_{m_{N-2}+r}$. Finally, this block goes through $n - (N-1)r$ Householder transformations of length m_N .
112 Since $N = \lceil n/r \rceil$, $n - (N-1)r < r$.

113 If $n - (N-1)r \geq r$, then $n - (N-1)r - r = n - Nr \geq 0$ which implies $n \geq Nr$. $\Rightarrow \Leftarrow !!!$ So I'll just
114 use r .

$$115 \quad \boxed{\hat{\mathbf{R}}_N = \mathbf{R}_N + \Delta \mathbf{R}_N, \text{ where } |\Delta \mathbf{R}_N| \leq [(1 + r\tilde{\gamma}_{m_N})(1 + \tilde{\gamma}_{(m+n/2)N}) - 1] |\mathbf{R}_N|}$$

Building Q:

$$m_1 + \dots + m_N + r(N) = (m + r(N+1)/2)(N) \cong mN + rN^2/2 \cong (m + n/2)N$$

Is this a slight underestimate?

The identity matrix (thin or full) is transformed via

$$(\mathbf{I}_m - \hat{\mathbf{W}}_1 \hat{\mathbf{Y}}_1^\top) \cdots (\mathbf{I}_m - \hat{\mathbf{W}}_N \hat{\mathbf{Y}}_N^\top),$$

where

$$\begin{aligned} \hat{\mathbf{W}}_N &= \mathbf{W}_N + \Delta \mathbf{W}_N, \quad |\Delta \mathbf{W}_N| \leq \tilde{\gamma}_{(m+n/2)N} |\mathbf{W}_{N-1}| \\ \hat{\mathbf{Y}}_N &= \mathbf{Y}_N + \Delta \mathbf{Y}_N, \quad |\Delta \mathbf{Y}_N| \leq \tilde{\gamma}_{(m+n/2)N} |\mathbf{Y}_{N-1}|. \end{aligned}$$

$$\hat{\mathbf{Q}} = \mathbf{Q} + \Delta \mathbf{Q}, \text{ where } |\Delta \mathbf{Q}| \leq \tilde{\gamma}_{(m+n/2)N} |\mathbf{Q}|$$

Overall, this accumulates to

$$(m + n/2)N \cong (m + n/2) \frac{n}{r} \cong mn/r + mn^2/2r.$$

TODO:

- check algebra/analysis.
- How does this compare to traditional HQR?
- [1] said it requires MORE FLOPs, so it probably should be slightly worse.

1.2.2. Round to lower precision at the end of each block.. The only change in the analysis that I foresee is going from

$$\sum_{k=1}^{i-1} m_k + r \text{ in the higher precision}$$

to

$$\sum_{k=1}^{i-1} \lceil (m_k + r) \frac{u_{high}}{u_{low}} \rceil \text{ in the lower precision.}$$

I think we can make some statement saying that if $r \ll n$, the two should be comparable. If $r = \mathcal{O}(n)$, then N is small so there are some trade-offs.

REFERENCES

[1] G. H. GOLUB AND C. F. VAN LOAN, *Matrix computations*, JHU press, 4 ed., 2013.

2. Algorithms I may need to reference in above sections:.

Algorithm 3: $\mathbf{z}_{\text{half}} = \text{simHalf}(f, \mathbf{x}_{\text{half}}, \mathbf{y}_{\text{half}})$ Simulate function $f \in \text{OPU}\{\text{dot_product}\}$ in half precision arithmetic given input variables \mathbf{x}, \mathbf{y} . Function **castup** converts half precision floats to single precision floats, and **castdown** converts single precision floats to half precision floats by rounding to the nearest half precision float.

Input: $\mathbf{x}_{\text{half}}, \mathbf{y}_{\text{half}} \in \mathbb{F}_{\text{half}}^m$, $f : \mathbb{R}^m \times \mathbb{R}^m \rightarrow \mathbb{R}^n$
Output: $\text{fl}(f(\mathbf{x}_{\text{half}}, \mathbf{y}_{\text{half}})) \in \mathbb{F}_{\text{half}}^n$
1 $\mathbf{x}_{\text{single}}, \mathbf{y}_{\text{single}} \leftarrow \text{castup}([\mathbf{x}_{\text{half}}, \mathbf{y}_{\text{half}}])$
2 $\mathbf{z}_{\text{single}} \leftarrow \text{fl}(f(\mathbf{x}_{\text{single}}, \mathbf{y}_{\text{single}}))$
3 $\mathbf{z}_{\text{half}} \leftarrow \text{castdown}(\mathbf{z}_{\text{single}})$
4 **return** \mathbf{z}_{half}

Algorithm 4: $\beta, \mathbf{v}, \sigma = \text{hh_vec}(\mathbf{x})$. Given a vector $\mathbf{x} \in \mathbb{R}^n$, return the Householder vector, \mathbf{v} ; a Householder constant, β ; and σ such that $(I - \beta \mathbf{v} \mathbf{v}^\top) \mathbf{x} = \sigma \hat{\mathbf{e}}_1$ and $\mathbf{v}_1 = 1$, (see LAPACK, Higham2002).

Input: $\mathbf{x} \in \mathbb{R}^m$
Output: $\mathbf{v} \in \mathbb{R}^m$, and $\sigma, \beta \in \mathbb{R}$ such that $(I - \beta \mathbf{v} \mathbf{v}^\top) \mathbf{x} = \pm \|\mathbf{x}\|_2 \hat{\mathbf{e}}_1 = \sigma \hat{\mathbf{e}}_1$
/* We choose the sign of sigma to avoid cancellation of \mathbf{x}_1 (As is the standard in LAPACK, LINPACK packages Higham2002). This makes $\beta > 0$. */
1 $\mathbf{v} \leftarrow \mathbf{x}$
2 $\sigma \leftarrow -\text{sign}(\mathbf{x}_1) \|\mathbf{x}\|_2$
3 $\mathbf{v}_1 \leftarrow \mathbf{x}_1 - \sigma$ // This is referred to as $\tilde{\mathbf{v}}_1$ later on.
4 $\beta \leftarrow -\frac{\mathbf{v}_1}{\sigma}$
5 $\mathbf{v} \leftarrow \frac{1}{\mathbf{v}_1} \mathbf{v}$
6 **return** $\beta, \mathbf{v}, \sigma$

Algorithm 5: $\mathbf{V}, \beta, \mathbf{R} = \text{qr}(A)$. Given a matrix $A \in \mathbb{R}^{m \times n}$ where $m \geq n$, return matrix $\mathbf{V} \in \mathbb{R}^{m \times n}$, vector $\beta \in \mathbb{R}^n$, and upper triangular matrix \mathbf{R} . An orthogonal matrix \mathbf{Q} can be generated from \mathbf{V} and β , and $\mathbf{QR} = \mathbf{A}$.

Input: $A \in \mathbb{R}^{m \times n}$ where $m \geq n$.
Output: $\mathbf{V}, \beta, \mathbf{R}$
1 $\mathbf{V}, \beta \leftarrow \mathbf{0}_{m \times n}, \mathbf{0}_m$
2 **for** $i = 1 : n$ **do**
3 $\mathbf{v}, \beta, \sigma \leftarrow \text{hh_vec}(\mathbf{A}[i : \text{end}, i])$
4 $\mathbf{V}[i : \text{end}, i], \beta_i, \mathbf{A}[i, i] \leftarrow \mathbf{v}, \beta, \sigma$ // Stores the Householder vectors and constants.
5 /* The next two steps update \mathbf{A} . */
6 $\mathbf{A}[i + 1 : \text{end}, i] \leftarrow \text{zeros}(m - i)$
7 $\mathbf{A}[i : \text{end}, i + 1 : \text{end}] \leftarrow \mathbf{A}[i : \text{end}, i + 1 : \text{end}] - \beta \mathbf{v} \mathbf{v}^\top \mathbf{A}[i : \text{end}, i + 1 : \text{end}]$
7 **return** $\mathbf{V}, \beta, \mathbf{A}[1 : n, 1 : n]$

Algorithm 6: QB \leftarrow `hh_mult(V, B)`: Given a set of householder vectors $\{\mathbf{v}_i\}_{i=1}^n$ and their corresponding constants $\{\beta_i\}_{i=1}^n$, compute $\mathbf{P}_1 \cdots \mathbf{P}_n \mathbf{B}$, where $\mathbf{P}_i := \mathbf{I} - \beta_i \mathbf{v}_i \mathbf{v}_i^\top$

Input: $\mathbf{V} \in \mathbb{R}^{m \times n}$, $\boldsymbol{\beta} \in \mathbb{R}^n$ where $m \geq n$. $\mathbf{B} \in \mathbb{R}^{m \times d}$.
Output: **QB**

/* $\mathbf{v}_i = \mathbf{V}[i : m, i] \in \mathbb{R}^{m-(i-1)}$ and $\mathbf{B}_i = \mathbf{B}[i : \text{end}, i : \text{end}] \in \mathbb{R}^{(m-(i-1)) \times (d-(i-1))}$. */

```

1 for  $i = 1 : n$  do
2    $\mathbf{B}_i \leftarrow \mathbf{B}_i - \beta_i \mathbf{v}_i (\mathbf{v}_i^\top \mathbf{B}_i)$ 
3 return B

```

Algorithm 7: $\mathbf{Q}, \mathbf{R} = \text{tsqr}(\mathbf{A}, L)$. Finds a QR factorization of a tall, skinny matrix, \mathbf{A} .

Input: $\mathbf{A} \in \mathbb{R}^{m \times n}$ where $m \gg n$, $L \leq \lfloor \log_2 \left(\frac{m}{n} \right) \rfloor$, and 2^L is the initial number of blocks.
Output: $\mathbf{Q} \in \mathbb{R}^{m \times n}$, $\mathbf{R} \in \mathbb{R}^{n \times n}$ such that $\mathbf{QR} = \mathbf{A}$.

```

1  $h \leftarrow \lfloor \frac{m}{2^L} \rfloor$  // Number of rows for all but the last block.
2  $r \leftarrow m - (2^L - 1)h$  // Number of rows for the last block ( $h \leq r < 2h$ ).
   /* Split  $\mathbf{A}$  into  $2^L$  blocks. Note that level ( $i$ ) has  $2^{L-i}$  blocks. */
3 for  $j = 1 : 2^L - 1$  do
4    $\mathbf{A}_j^{(0)} \leftarrow \mathbf{A}[(j-1)h+1 : jh, :]$ 
5  $\mathbf{A}_{2^L}^{(0)} \leftarrow \mathbf{A}[(2^L-1)h : m, :]$  // Last block may have more rows.
   /* Store Householder vectors as columns of matrix  $\mathbf{V}_j^{(i)}$ , Householder
      constants as components of vector  $\beta_j^{(i)}$ , and set up the next level. */
6 for  $i = 0 : L - 1$  do
   /* The inner loop can be parallelized. */
7   for  $j = 1 : 2^{L-i}$  do
8      $\mathbf{V}_{2j-1}^{(i)}, \beta_{2j-1}^{(i)}, \mathbf{R}_{2j-1}^{(i)} \leftarrow \text{qr}(\mathbf{A}_{2j-1}^{(i)})$ 
9      $\mathbf{V}_{2j}^{(i)}, \beta_{2j}^{(i)}, \mathbf{R}_{2j}^{(i)} \leftarrow \text{qr}(\mathbf{A}_{2j}^{(i)})$ 
10     $\mathbf{A}_j^{(i+1)} \leftarrow \begin{bmatrix} \mathbf{R}_{2j-1}^{(i)} \\ \mathbf{R}_{2j}^{(i)} \end{bmatrix}$ 
   /* At the bottom-most level, get the final  $\mathbf{R}$  factor. */
11  $\mathbf{V}_1^{(L)}, \beta_1^{(L)}, \mathbf{R} \leftarrow \text{qr}(\mathbf{A}_1^{(L)})$ 
12  $\mathbf{Q}_1^{(L)} \leftarrow \text{hh\_mult}(\mathbf{V}_1^{(L)}, I_{2n \times n})$ 
   /* Compute  $\mathbf{Q}^{(i)}$  factors by applying  $\mathbf{V}^{(i)}$  to  $\mathbf{Q}^{(i+1)}$  factors. */
13 for  $i = L - 1 : -1 : 1$  do
14   for  $j = 1 : 2^{L-i}$  do
15      $\mathbf{Q}_j^{(i)} \leftarrow \text{hh\_mult} \left( \mathbf{V}_j^{(i)}, \begin{bmatrix} \tilde{\mathbf{Q}}_{\alpha(j), \phi(j)}^{(i+1)} \\ \mathbf{0}_{n,n} \end{bmatrix} \right)$ 
   /* At the top-most level, construct the final  $\mathbf{Q}$  factor. */
16  $\mathbf{Q} \leftarrow \mathbf{I}$ ;
17 for  $j = 1 : 2^L$  do
18    $\mathbf{Q} \leftarrow \begin{bmatrix} \mathbf{Q} \\ \text{hh\_mult} \left( \mathbf{V}_j^{(0)}, \begin{bmatrix} \tilde{\mathbf{Q}}_{\alpha(j), \phi(j)}^{(1)} \\ O_{\tilde{h}, n} \end{bmatrix} \right) \end{bmatrix}$ 
19 return  $\mathbf{Q}, \mathbf{R}$ 
```
