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. 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:

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

∎

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 of FLOPs of this step (line 6 in alg. 5) requires two Level-2 BLAS operations when computed efficiently, which are $\mathbf{C} := \mathbf{v}_k^\top \mathbf{A}_{k:m,k+1:n}\mathbb{R}^{1\times(n-k)}$ and $\mathbf{v}\mathbf{C}$, an outer product.

In BQR, 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. Both of these operations are rich in 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:

1. $A := \mathbf{W}_1^\top[\mathbf{C}_2 \cdots \mathbf{C}_N]$ is a matrix-matrix multiply with m-length inner products.

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

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

Initialize: $\mathbf{W} := \boldsymbol{\beta}_1 \mathbf{V}[:,1]$ and $\mathbf{Y} := \mathbf{V}[:,1]$.
for $j = 2 : r$ **do**
  $\mathbf{z} \leftarrow \boldsymbol{\beta}_j \left[\mathbf{V}[:,j] - \mathbf{W}\left(\mathbf{Y}^\top \mathbf{V}[:,j]\right)\right]$
  $\mathbf{W} \leftarrow [\mathbf{W} \quad \mathbf{z}]$ /* Update $\mathbf{W}$. */
  $\mathbf{Y} \leftarrow [\mathbf{Y} \quad \mathbf{V}[:,j]]$ /* Update $\mathbf{Y}$. */
  // $\mathbf{W}$ and $\mathbf{Y}$ are now m-by-j matrices.
return $\mathbf{W}, \mathbf{Y}$

2. $[\mathbf{C}_2 \cdots \mathbf{C}_N] - \mathbf{Y}_1 A$ is a matrix-matrix multiply with subtraction where the product $\mathbf{Y}_1 A$ computes r-length inner products.

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 [**?**]).

Algorithm 2: $\mathbf{Q}, \mathbf{R} \leftarrow$ `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: $\mathbf{Q}, \mathbf{R}$

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

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

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

Now that we have discussed the block HQR, let's now set the assumptions for our mixed-

precision analysis. I will consider two cases, the first is where $\mathbf{A}$ is cast down to the lower precision after updating each block. (i.e. After line 5 within the forloop in alg. 2, and the second is where casting down only happens at the end of the factorization. Everything will be done in the higher precision except for the cast down operations.

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

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

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

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

$$\boxed{\hat{\mathbf{R}}_{\mathbf{1}} = \mathbf{R_1} + \Delta\mathbf{R_1}, \text{ where } \quad |\Delta\mathbf{R_1}| \le r\tilde{\gamma}_{m_1}|\mathbf{R_1}|}$$

The second block: This block was transformed via $\mathbf{I}_m - \hat{\mathbf{Y}}_{\mathbf{1}}\hat{\mathbf{W}}_{\mathbf{1}}{}^{\top}$, where

$$\hat{\mathbf{W}}_{\mathbf{1}} = \mathbf{W_1} + \Delta\mathbf{W_1}, \quad |\Delta\mathbf{W_1}| \le \tilde{\gamma}_{m_1+r}|\mathbf{W_1}|$$
$$\hat{\mathbf{Y}}_{\mathbf{1}} = \mathbf{Y_1} + \Delta\mathbf{Y_1}, \quad |\Delta\mathbf{Y_1}| \le \tilde{\gamma}_{m_1+r}|\mathbf{Y_1}|.$$

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

$$\boxed{\hat{\mathbf{R}}_{\mathbf{2}} = \mathbf{R_2} + \Delta\mathbf{R_2}, \text{ where } \quad |\Delta\mathbf{R_2}| \le \left[(1 + r\tilde{\gamma}_{m_2})(1 + \tilde{\gamma}_{m_1+r}) - 1\right]|\mathbf{R_2}|}$$

The third block: This block was transformed via $(\mathbf{I}_m - \hat{\mathbf{Y}}_{\mathbf{2}}\hat{\mathbf{W}}_{\mathbf{2}}{}^{\top})(\mathbf{I}_m - \hat{\mathbf{Y}}_{\mathbf{1}}\hat{\mathbf{W}}_{\mathbf{1}}{}^{\top})$, where

$$\hat{\mathbf{W}}_{\mathbf{2}} = \mathbf{W_2} + \Delta\mathbf{W_2}, \quad |\Delta\mathbf{W_2}| \le \tilde{\gamma}_{m_1+r+m_2+r}|\mathbf{W_2}|$$
$$\hat{\mathbf{Y}}_{\mathbf{2}} = \mathbf{Y_2} + \Delta\mathbf{Y_2}, \quad |\Delta\mathbf{Y_2}| \le \tilde{\gamma}_{m_1+r+m_2+r}|\mathbf{Y_2}|.$$

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

$$\boxed{\hat{\mathbf{R_3}} = \mathbf{R_3} + \Delta\mathbf{R_3}, \text{ where } \quad |\Delta\mathbf{R_3}| \leq \left[(1 + r\tilde{\gamma}_{m_3})\,(1 + \tilde{\gamma}_{m_1+r+m_2+r}) - 1\right]|\mathbf{R_3}|}$$

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

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

where

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

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

$$\begin{aligned} m_1 + \cdots + m_{i-1} + r(i-1) &= r(i-1) + \sum_{k=1}^{i-1} m_k = r(i-1) + \sum_{k=1}^{i-1}(m - (k-1)r) \\ &= (m+r)(i-1) + r\sum_{k=1}^{i-1}(k-1) = (m+r-r)(i-1)r\sum_{k=1}^{i-1}k \\ &= m(i-1) + r\frac{i(i-1)}{2} = (m + ri/2)(i-1) \approxeq mi + ri^2/2 \end{aligned}$$

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

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

Let's now consider the last block.

The $(\mathbf{N})^{\mathbf{th}}$ block:

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

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

This block was transformed via

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

where

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

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

Since $N = \lceil n/r \rceil$, $n - (N-1)r < r$.
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 use r.

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

Building Q:

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

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

$$\hat{\mathbf{W_N}} = \mathbf{W_N} + \Delta\mathbf{W_N}, \quad |\Delta\mathbf{W_N}| \leq \tilde{\gamma}_{(m+n/2)N}|\mathbf{W_{N-1}}|$$
$$\hat{\mathbf{Y_N}} = \mathbf{Y_N} + \Delta\mathbf{Y_N}, \quad |\Delta\mathbf{Y_N}| \leq \tilde{\gamma}_{(m+n/2)N}|\mathbf{Y_{N-1}}|.$$

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

Overall, this accumulates to

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

TODO:

- check algebra/analysis.
- How does this compare to traditional HQR?
- [**?**] 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

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

Algorithm 3: $\mathbf{z}_{\text{half}} = \texttt{simHalf}(f, \mathbf{x}_{\text{half}}, \mathbf{y}_{\text{half}})$ Simulate function $f \in \text{OP} \cup \{\texttt{dot\_product}\}$ 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: $\mathbf{x}_{\text{half}}, \mathbf{y}_{\text{half}} \in \mathbb{F}_{\text{half}}^m$, $f : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}^n$
Output: $\text{fl}(f(\mathbf{x}_{\text{half}}, \mathbf{y}_{\text{half}})) \in \mathbb{F}_{\text{half}}^n$

$\mathbf{x}_{\text{single}}, \mathbf{y}_{\text{single}} \leftarrow \texttt{castup}([\mathbf{x}_{\text{half}}, \mathbf{y}_{\text{half}}])$
$\mathbf{z}_{\text{single}} \leftarrow \text{fl}(f(\mathbf{x}_{\text{single}}, \mathbf{y}_{\text{single}}))$
$\mathbf{z}_{\text{half}} \leftarrow \texttt{castdown}(\mathbf{z}_{\text{single}})$
return $\mathbf{z}_{\text{half}}$

Algorithm 4: β, $\mathbf{v}$, $\sigma = \texttt{hh\_vec}(\mathbf{x})$. Given a vector $\mathbf{x} \in \mathbb{R}^n$, return the Householder vector, $\mathbf{v}$; a Householder constant, β; and σ such that $(I - \beta \mathbf{v}\mathbf{v}^\top)\mathbf{x} = \sigma(\hat{e_1})$ 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 $(I - \beta \mathbf{v}\mathbf{v}^\top)\mathbf{x} = \pm\|\mathbf{x}\|_2 \hat{e_1} = \sigma \hat{e_1}$
/* We choose the sign of sigma to avoid cancellation of $\mathbf{x}_1$ (As is the standard in LAPACK, LINPACK packages Higham2002). This makes $\beta > 0$. */
$\mathbf{v} \leftarrow \mathbf{x}$
$\sigma \leftarrow -\text{sign}(\mathbf{x}_1)\|\mathbf{x}\|_2$
$\mathbf{v}_1 \leftarrow \mathbf{x}_1 - \sigma$ // This is referred to as $\tilde{\mathbf{v}}_1$ later on.
$\beta \leftarrow -\frac{\mathbf{v}_1}{\sigma}$
$\mathbf{v} \leftarrow \frac{1}{\mathbf{v}_1}\mathbf{v}$
return β, $\mathbf{v}$, σ

Algorithm 5: $\mathbf{V}, \boldsymbol{\beta}, \mathbf{R} = \texttt{qr}(A)$. 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 $\boldsymbol{\beta} \in \mathbb{R}^n$, and upper triangular matrix $\mathbf{R}$. An orthogonal matrix $\mathbf{Q}$ can be generated from $\mathbf{V}$ and $\boldsymbol{\beta}$, and $\mathbf{QR} = \mathbf{A}$.

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

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

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

Algorithm 7: $\mathbf{Q}, \mathbf{R} = \texttt{tsqr}(\mathbf{A}, L)$. 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 $\mathbf{QR} = \mathbf{A}$.

$h \leftarrow \lfloor \frac{m}{2^L} \rfloor$ // Number of rows for all but the last block.
$r \leftarrow m - (2^L - 1)h$ // 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. */
for $j = 1 : 2^L - 1$ **do**
  $\mathbf{A}_j^{(0)} \leftarrow \mathbf{A}[(j-1)h + 1 : jh, :]$
$\mathbf{A}_{2^L}^{(0)} \leftarrow \mathbf{A}[(2^L - 1)h : m, :]$ // Last block may have more rows.
/* Store Householder vectors as columns of matrix $\mathbf{V}_j^{(i)}$, Householder constants as components of vector $\boldsymbol{\beta}_j^{(i)}$, and set up the next level. */
for $i = 0 : L - 1$ **do**
  /* The inner loop can be parallelized. */
  for $j = 1 : 2^{L-i}$ **do**
    $\mathbf{V}_{2j-1}^{(i)}, \boldsymbol{\beta}_{2j-1}^{(i)}, \mathbf{R}_{2j-1}^{(i)} \leftarrow \texttt{qr}(\mathbf{A}_{2j-1}^{(i)})$
    $\mathbf{V}_{2j}^{(i)}, \boldsymbol{\beta}_{2j}^{(i)}, \mathbf{R}_{2j}^{(i)} \leftarrow \texttt{qr}(\mathbf{A}_{2j}^{(i)})$
    $\mathbf{A}_j^{(i+1)} \leftarrow \begin{bmatrix} \mathbf{R}_{2j-1}^{(i)} \\ \mathbf{R}_{2j}^{(i)} \end{bmatrix}$
/* At the bottom-most level, get the final $\mathbf{R}$ factor. */
$\mathbf{V}_1^{(L)}, \boldsymbol{\beta}_1^{(L)}, \mathbf{R} \leftarrow \texttt{qr}(\mathbf{A}_1^{(L)})$
$\mathbf{Q}_1^{(L)} \leftarrow \texttt{hh\_mult}(\mathbf{V}_1^{(L)}, I_{2n \times n})$
/* Compute $\mathbf{Q}^{(i)}$ factors by applying $\mathbf{V}^{(i)}$ to $\mathbf{Q}^{(i+1)}$ factors. */
for $i = L - 1 : -1 : 1$ **do**
  for $j = 1 : 2^{L-i}$ **do**
    $\mathbf{Q}_j^{(i)} \leftarrow \texttt{hh\_mult}\left(\mathbf{V}_j^{(i)}, \begin{bmatrix} \tilde{\mathbf{Q}}_{\alpha(j),\phi(j)}^{(i+1)} \\ \mathbf{0}_{n,n} \end{bmatrix}\right)$
/* At the top-most level, construct the final $\mathbf{Q}$ factor. */
$\mathbf{Q} \leftarrow []$;
for $j = 1 : 2^L$ **do**
  $\mathbf{Q} \leftarrow \begin{bmatrix} \mathbf{Q} \\ \texttt{hh\_mult}\left(\mathbf{V}_j^{(0)}, \begin{bmatrix} \tilde{\mathbf{Q}}_{\alpha(j),\phi(j)}^{(1)} \\ O_{\tilde{h},n} \end{bmatrix}\right) \end{bmatrix}$
return $\mathbf{Q}$, $\mathbf{R}$