Lecture 8 - *Gram-Schmidt Orthogonalization*

OBJECTIVE:
Classical Gram-Schmidt is unstable, so we present modified Gram-Schmidt, which is stable.
We also interpret it as right-multiplication by upper-triangular matrices.

◇ GRAM-SCHMIDT PROJECTIONS

We have just seen Gram-Schmidt in its classical form. Now we describe the same algorithm but in another way: using orthogonal projectors.

Let $\mathbf{A} \in \mathbb{R}^{m \times n}$ $(m \geq n)$ have full rank.
Consider now the sequence of formulas

$$\mathbf{q}_1 = \frac{\mathbf{P}_1 \mathbf{a}_1}{\|\mathbf{P}_1 \mathbf{a}_1\|}, \quad \mathbf{q}_2 = \frac{\mathbf{P}_2 \mathbf{a}_2}{\|\mathbf{P}_2 \mathbf{a}_2\|}, \quad \cdots \quad , \quad \mathbf{q}_n = \frac{\mathbf{P}_n \mathbf{a}_n}{\|\mathbf{P}_n \mathbf{a}_n\|},$$

where each $\mathbf{P}_j$ is an orthogonal projector.

Specifically, $\mathbf{P}_j \in \mathbb{R}^{m \times m}$ is the matrix of rank $m - (j-1)$ that projects $\mathbb{R}^m$ orthogonally onto the space orthogonal to $< \mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_{j-1} >$.

Note 1. *When* $j = 1$, $\mathbf{P}_1 = \mathbf{I}$.

Also note that as defined,

- each $\mathbf{q}_j$ is orthogonal to $\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_{j-1}$,
- each $\mathbf{q}_j$ lies in $< \mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_j >$,
- and each $\|\mathbf{q}_j\| = 1$.

$\therefore$ This algorithm is equivalent to the classical Gram Schmidt Algorithm 7.1

But, each $\mathbf{P}_j$ can be represented explicitly:
Let $\hat{\mathbf{Q}}_{j-1} \in \mathbb{R}^{m \times (j-1)}$ contain the first $j-1$ columns of $\hat{\mathbf{Q}}$

$$\hat{\mathbf{Q}}_{j-1} = \left[\begin{array}{c|c|c|c} \mathbf{q}_1 & \mathbf{q}_2 & \cdots & \mathbf{q}_{j-1} \end{array} \right]$$

Then, $\mathbf{P}_j$ is given by

$$\mathbf{P}_j = \mathbf{I} - \hat{\mathbf{Q}}_{j-1} \hat{\mathbf{Q}}_{j-1}^T$$

Note 2.

$$\mathbf{P}_j\mathbf{a}_j = \mathbf{a}_j - (\mathbf{q}_1^T\mathbf{a}_j)\mathbf{q}_1 - (\mathbf{q}_2^T\mathbf{a}_j)\mathbf{q}_2 - \ldots - (\mathbf{q}_{j-1}^T\mathbf{a}_j)\mathbf{q}_{j-1}$$

◇ MODIFIED GRAM-SCHMIDT

As mentioned, classical Gram-Schmidt is unstable numerically and so is never implemented in practice.

(We will treat numerical stability systematically in Lecture 14
$\longrightarrow$ for now it is sufficient to consider a stable algorithm as one which does not suffer drastically from perturbations due to roundoff errors)

Fortunately, there is a simple modification that makes Gram-Schmidt stable.

For each j, Algorithm 7.1 computes a single orthogonal projector of rank $m - (j-1)$

$$\mathbf{v}_j = \mathbf{P}_j\mathbf{a}_j$$

In contrast, the modified Gram Schmidt algorithm achieves the same result but by means of $(j-1)$ projections of rank $(m-1)$.

Recall from Lecture 6 that $\mathbf{P}_{\perp\mathbf{q}}$ denotes the rank $m-1$ orthogonal projector onto the space orthogonal to some nonzero vector $\mathbf{q} \in \mathbb{R}^m$.

From the definition of $\mathbf{P}_j$, it is easy to see that

$$\mathbf{P}_j = \mathbf{P}_{\perp\mathbf{q}_{j-1}} \quad \cdots \quad \mathbf{P}_{\perp\mathbf{q}_2}\mathbf{P}_{\perp\mathbf{q}_1}$$

with $\mathbf{P}_1 = \mathbf{I}$.

Thus, we can write

$$\mathbf{v}_j = \mathbf{P}_{\perp\mathbf{q}_{j-1}} \quad \cdots \quad \mathbf{P}_{\perp\mathbf{q}_2}\mathbf{P}_{\perp\mathbf{q}_1}\mathbf{a}_j$$

Mathematically, the two formulas for $\mathbf{v}_j$ are equivalent, but modified Gram-Schmidt is based on the second formula rather than the first.

The modified Gram-Schmidt goes as follows:

$$
\begin{aligned}
\mathbf{v}_j^{(1)} &= \mathbf{a}_j \\
\mathbf{v}_j^{(2)} &= \mathbf{P}_{\perp \mathbf{q}_1} \mathbf{v}_j^{(1)} = \mathbf{v}_j^{(1)} - \mathbf{q}_1 \mathbf{q}_1^T \mathbf{v}_j^{(1)} \\
\mathbf{v}_j^{(3)} &= \mathbf{P}_{\perp \mathbf{q}_2} \mathbf{v}_j^{(2)} = \mathbf{v}_j^{(2)} - \mathbf{q}_2 \mathbf{q}_2^T \mathbf{v}_j^{(2)} \\
&\vdots \\
\mathbf{v}_j^{(j)} &= \mathbf{v}_j = \mathbf{P}_{\perp \mathbf{q}_{j-1}} \mathbf{v}_j^{(j-1)} \\
&= \mathbf{v}_j^{(j-1)} - \mathbf{q}_{j-1} \mathbf{q}_{j-1}^T \mathbf{v}_j^{(j-1)}
\end{aligned}
$$

In finite-precision computer arithmetic, we will see the formulation introduces smaller errors than the previous one.

For the implementation, $\mathbf{P}_{\perp \mathbf{q}_i}$ can be conveniently applied to $\mathbf{v}_j^{(i)}$ for each $j > i$ as soon as $\mathbf{q}_i$ is known.

ALGORITHM 8.1: MODIFIED GRAM-SCHMIDT (STABLE)

for $i = 1$ to n **do**
  $\mathbf{v}_i = \mathbf{a}_i$
end for
for $i = 1$ to n **do**
  $r_{ii} = \|\mathbf{v}_i\|$
  $\mathbf{q}_i = \frac{\mathbf{v}_i}{r_{ii}}$
  for $j = i + 1$ to n **do**
    $r_{ij} = \mathbf{q}_i^T \mathbf{v}_j$
    $\mathbf{v}_j = \mathbf{v}_j - r_{ij}\mathbf{q}_i$
  end for
end for

Note 3. *We let* $\mathbf{v}_j$ *overwrite* $\mathbf{a}_i$ *and* $\mathbf{q}_i$ *overwrite* $\mathbf{v}_i$ *to save storage.*

◇ OPERATION COUNT

With any algorithm, it is important to assess its cost.
We now do so for Gram-Schmidt.
We assess cost in the classical sense of counting floating-point operations (*flops*)
i.e., each $+, -, *, /, \sqrt{\ }$ all count as one flop.

This is a simplification in the sense that there is much more going on during program execution that will affect performance besides simple flop counts.
e.g., how data are stored and accessed in memory, competing jobs, message passing, etc.

For both variants of the Gram-Schmidt algorithm, the result is the following:

Theorem 1. *Algorithms 7.1 and 8.1 require* $\sim 2mn^2$ *flops to compute a (reduced) QR factorization of an* $m \times n$ *matrix.*

Note 4. *We only worry about the leading-order behaviors when counting flops*
i.e., we assume m, n *are large.*

$$\sim 2mn^2 \quad \text{means} \quad \lim_{m,n\to 0} \frac{\text{number of flops}}{2mn^2} = 1$$

It is similar to $\mathcal{O}(\cdot)$ notation, except more precise. We could restate Theorem 8.1 as taking $\mathcal{O}(mn^2)$ flops, but then we would not know that the leading coefficient was 2.

We now derive the flop count specifically for modified Gram-Schmidt:

For large m, n, the work is dominated by the innermost loop:

$$r_{ij} = \mathbf{q}_i^T \mathbf{v}_j \quad \rightarrow \quad m \text{ multiplications}/m-1 \text{ additions}$$
$$\mathbf{v}_j = \mathbf{v}_j - r_{ij}\mathbf{q}_i \quad \rightarrow \quad m \text{ multiplications}/m \text{ subtractions}$$

$\therefore$ Total work $\sim 4m$ flops (~ 4 flops per vector component).

$$
\begin{aligned}
\therefore \qquad \text{total flops} \quad &\sim \quad \sum_{i=1}^{n} \sum_{j=i+1}^{n} 4m \\
&= \quad \sum_{i=1}^{n} (n-i) 4m \\
&= \quad \left(n^2 - \frac{n(n+1)}{2} \right) \cdot 4m \\
&\sim \quad 2mn^2
\end{aligned}
$$

◇ GRAM-SCHMIDT AS TRIANGULAR ORTHOGONALIZATION

Each outer step of modified Gram-Schmidt can be interpreted as right-multiplication by a square upper-triangular matrix.

e.g., beginning with $\mathbf{A}$, the first iteration multiplies $\mathbf{a}_1$ by $\frac{1}{r_{11}}$, then subtracts r_{ij} times the result from each of the remaining $\mathbf{a}_j$
$\longrightarrow$ equivalent to right-multiplication by a matrix $\mathbf{R}_1$:

$$\begin{array}{cccc} \left[\ \mathbf{a}_1 \right. & \left| \ \mathbf{a}_2 \right. & \left| \ \ldots \right. & \left| \ \mathbf{a}_n \ \right] \\ \uparrow & \uparrow & & \uparrow \\ \mathbf{v}_1^{(1)} & \mathbf{v}_2^{(1)} & & \mathbf{v}_n^{(1)} \end{array} \begin{bmatrix} \frac{1}{r_{11}} & \frac{-r_{12}}{r_{11}} & \frac{-r_{13}}{r_{11}} & \cdots \\ & 1 & & \\ & & 1 & \\ & & & \ddots \end{bmatrix}$$

$$= \left[\ \mathbf{q}_1 \ \middle|\ \mathbf{v}_2^{(2)} \ \middle|\ \mathbf{v}_3^{(2)} \ \middle|\ \ldots \ \middle|\ \mathbf{v}_n^{(2)} \ \right]$$

$\longrightarrow$ step i subtracts $\frac{r_{ij}}{r_{ii}}*$ column i of the current $\mathbf{A}$ for columns $j > i$, replacing column i by $\frac{1}{r_{11}}$ times itself.

This corresponds to multiplication by an upper triangular matrix $\mathbf{R}_i$.

e.g.,

$$\mathbf{R}_2 = \begin{bmatrix} 1 & & & \\ & \frac{1}{r_{22}} & \frac{-r_{23}}{r_{22}} & \cdots \\ & & 1 & \\ & & & \ddots \end{bmatrix}$$

$$\mathbf{R}_3 = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & \frac{1}{r_{33}} & \cdots \\ & & & \ddots \end{bmatrix}$$

etc.

At the end, we have

$$\mathbf{A}\mathbf{R}_1\mathbf{R}_2 \ldots \mathbf{R}_n = \hat{\mathbf{Q}}$$

where $\hat{\mathbf{R}}^{-1} = \mathbf{R}_1\mathbf{R}_2 \ldots \mathbf{R}_n$.

$\Rightarrow$ Gram-Schmidt is a method of *triangular orthogonalization*:
it applies triangular operations on the right of a matrix to reduce it to a matrix with orthonormal columns.

Note 5. *We never compute the* $\mathbf{R}_i$ *explicitly!*
They are meant only to give insight.
We will see a similarity to Gaussian elimination in Lecture 20.