EECS 16B Designing Information Devices and Systems II
Fall 2021 Note 12: Orthonormalization

1 Repeated System ID

So far when performing system ID, we assumed that we already knew what type of model our system was, e.g. whether it was a first order, second order, or nth order difference equation. However, this assumption may not be valid all the time, and so we might not have a good idea of the order of our system. One way to deal with this is to try fit a first order model, second order model, and so on till an nth order model, and compare how well the models perform on a test set of data. Let's consider what must happen for us to do this.

For this example, we will just consider the case of a scalar state with a scalar input. Then we will have the following candidate models:

$$y_1[i] = a_1 y[i-1] + bu[i-1] \tag{1}$$
$$y_2[i] = a_2 y[i-2] + a_1 y[i-1] + bu[i-1] \tag{2}$$
$$\vdots$$
$$y_n[i] = a_n y[i-n] + a_{n-1} y[i-(n-1)] + \cdots + a_1 y[i-1] + bu[i-1] \tag{3}$$

If we try to do system ID for the first order system (1) with r rows of data, we end up with the following least square problem:

$$D_1 \vec{x}_1 \approx \vec{y} \tag{4}$$
$$\begin{bmatrix} y[0] & u[0] \\ y[1] & u[1] \\ \vdots & \vdots \\ y[r-1] & u[r-1] \end{bmatrix} \begin{bmatrix} a_1 \\ b \end{bmatrix} \approx \begin{bmatrix} y[1] \\ y[2] \\ \vdots \\ y[r] \end{bmatrix} \tag{5}$$

Now note that to construct the system ID matrix for the second order system (2) just requires adding another column to the left of our data matrix. Here, for all negative time $i < 0$, we assume $y[i] = 0$.

$$D_2 \vec{x}_2 \approx \vec{y} \tag{6}$$
$$\begin{bmatrix} y[-1] & y[0] & u[0] \\ y[0] & y[1] & u[1] \\ \vdots & \vdots & \vdots \\ y[r-2] & y[r-1] & u[r-1] \end{bmatrix} \begin{bmatrix} a_2 \\ a_1 \\ b \end{bmatrix} \approx \begin{bmatrix} y[1] \\ y[2] \\ \vdots \\ y[r] \end{bmatrix} \tag{7}$$

This can be repeated, so each next higher order will require 1 more column on the left to the data matrix. Then, we will need to calculate the least squares solution for each problem, getting

$$\vec{\hat{x}}_1 = (D_1^\top D_1)^{-1} D_1^\top \vec{y} \tag{8}$$

$$\vec{\hat{x}}_2 = (D_2^\top D_2)^{-1} D_2^\top \vec{y} \tag{9}$$

$$\vdots$$

$$\vec{\hat{x}}_n = (D_n^\top D_n)^{-1} D_n^\top \vec{y} \tag{10}$$

This could take a lot of time if r and n are very large, especially the matrix inverse operation as it is $O(k^3)$ runtime where k is the number of rows of the matrix. Is there any way we can take advantage of the structure of our D_i matrices (where D_{i+1} is just 1 more column added to D_i) to make this process faster?

It turns out there is, and it relies on a process called **Gram-Schmidt orthonormalization**, which will be the focus of this note.

2 Orthogonal Vectors and Projection

Recall from 16A that two vectors $\vec{v}, \vec{w}$ are orthogonal if they are 90° apart. Remember that an equivalent definition is that they are orthogonal if and only if

$$\langle \vec{v}, \vec{w} \rangle = \vec{v}^\top \vec{w} = \vec{w}^\top \vec{v} = 0 \tag{11}$$

Recall that the orthogonal projection of a vector $\vec{y}$ on to any other nonzero vector $\vec{b}$ is

$$\text{Proj}_{\vec{b}} \vec{y} = \vec{y}_{\vec{b}} = \frac{\vec{y}^\top \vec{b}}{\left\|\vec{b}\right\|^2} \vec{b} = \frac{\langle \vec{y}, \vec{b} \rangle}{\|\vec{b}\|^2} \vec{b} \tag{12}$$

Also recall that least squares is just an orthogonal projection of a vector $\vec{y}$ onto an entire subspace of vectors spanned by the columns of A, so

$A\vec{x} = \vec{y} \Rightarrow \hat{x} = (A^\top A)^{-1} A^\top \vec{y}$

$$\vec{y}_A = A\hat{x} = A(A^\top A)^{-1} A^\top \vec{y} \tag{13}$$

In this section, we will show that if the columns of A are mutually orthogonal to each other, the projection of $\vec{y}$ onto $\text{Span}(A)$ is the sum of the projection of $\vec{y}$ onto each column of A individually. Let's take a look at the case where $j = 2$ and the vectors are mutually orthogonal. Suppose the vectors found so far are $\vec{v}_1$ and $\vec{v}_2$, i.e., $A_2 = \begin{bmatrix} | & | \\ \vec{v}_1 & \vec{v}_2 \\ | & | \end{bmatrix}$. Then, the projection of $\vec{y}$ onto A_2 is

$$\vec{y}_{A_2} = A_2 \left(A_2^\top A_2\right)^{-1} A_2^\top \vec{y}. \tag{14}$$

Let's first compute the term $\left(A_2^\top A_2\right)^{-1}$:

$$A_2^\top A_2 = \begin{bmatrix} — & \vec{v}_1^\top & — \\ — & \vec{v}_2^\top & — \end{bmatrix} \begin{bmatrix} | & | \\ \vec{v}_1 & \vec{v}_2 \\ | & | \end{bmatrix} \tag{15}$$

$$= \begin{bmatrix} \vec{v}_1^\top \vec{v}_1 & \vec{v}_1^\top \vec{v}_2 \\ \vec{v}_2^\top \vec{v}_1 & \vec{v}_2^\top \vec{v}_2 \end{bmatrix} \tag{16}$$

$$= \begin{bmatrix} \|\vec{v}_1\|^2 & 0 \\ 0 & \|\vec{v}_2\|^2 \end{bmatrix}. \tag{17}$$

Thus, we have a diagonal matrix and so

$$\left(A_2^\top A_2\right)^{-1} = \begin{bmatrix} \frac{1}{\|\vec{v}_1\|^2} & 0 \\ 0 & \frac{1}{\|\vec{v}_2\|^2} \end{bmatrix}. \tag{18}$$

Then, substituting this matrix into the original expression, the projection of $\vec{y}$ onto $\mathrm{Span}(A_2)$ is

$$\vec{y}_{A_2} = A_2 \left(A_2^\top A_2\right)^{-1} A_2^\top \vec{y} \tag{19}$$

$$= \begin{bmatrix} | & | \\ \vec{v}_1 & \vec{v}_2 \\ | & | \end{bmatrix} \begin{bmatrix} \frac{1}{\|\vec{v}_1\|^2} & 0 \\ 0 & \frac{1}{\|\vec{v}_2\|^2} \end{bmatrix} \begin{bmatrix} — & \vec{v}_1^\top & — \\ — & \vec{v}_2^\top & — \end{bmatrix} \vec{y} \tag{20}$$

$$= \begin{bmatrix} | & | \\ \vec{v}_1 & \vec{v}_2 \\ | & | \end{bmatrix} \begin{bmatrix} \frac{1}{\|\vec{v}_1\|^2} & 0 \\ 0 & \frac{1}{\|\vec{v}_2\|^2} \end{bmatrix} \begin{bmatrix} \vec{v}_1^\top \vec{y} \\ \vec{v}_2^\top \vec{y} \end{bmatrix} \tag{21}$$

$$= \begin{bmatrix} | & | \\ \vec{v}_1 & \vec{v}_2 \\ | & | \end{bmatrix} \begin{bmatrix} \frac{\vec{v}_1^\top \vec{y}}{\|\vec{v}_1\|^2} \\ \frac{\vec{v}_2^\top \vec{y}}{\|\vec{v}_2\|^2} \end{bmatrix} \tag{22}$$

$$= \left(\frac{\vec{v}_1^\top \vec{y}}{\|\vec{v}_1\|^2}\right) \vec{v}_1 + \left(\frac{\vec{v}_2^\top \vec{y}}{\|\vec{v}_2\|^2}\right) \vec{v}_2. \tag{23}$$

Observe that the first term in the sum above is the projection of $\vec{y}$ onto $\vec{v}_1$ and the second term is the projection of $\vec{y}$ onto $\vec{v}_2$. Generalizing this pattern, we can guess that the projection of $\vec{y}$ onto $\mathrm{Span}(A_n)$ where A_n has mutually orthogonal columns is

$$\vec{y}_{A_n} = \left(\frac{\vec{v}_1^\top \vec{y}}{\|\vec{v}_1\|^2}\right) \vec{v}_1 + \left(\frac{\vec{v}_2^\top \vec{y}}{\|\vec{v}_2\|^2}\right) \vec{v}_2 + \cdots + \left(\frac{\vec{v}_n^\top \vec{y}}{\|\vec{v}_n\|^2}\right) \vec{v}_n. \tag{24}$$

Furthermore, observe that if $\vec{v}_1, \ldots, \vec{v}_n$ are unit vectors (i.e., they all have length 1), then the above would further reduce to

$$\vec{y}_{A_n} = \left(\vec{v}_1^\top \vec{y}\right) \vec{v}_1 + \left(\vec{v}_2^\top \vec{y}\right) \vec{v}_2 + \cdots + \left(\vec{v}_n^\top \vec{y}\right) \vec{v}_n \tag{25}$$

$$= \begin{bmatrix} | & & | \\ \vec{v}_1 & \ldots & \vec{v}_n \\ | & & | \end{bmatrix} \begin{bmatrix} — & \vec{v}_1^\top & — \\ & \vdots & \\ — & \vec{v}_n^\top & — \end{bmatrix} \vec{y} = A_n A_n^\top \vec{y} \tag{26}$$

Definition: A set of vectors $\{\vec{v}_1, \ldots, \vec{v}_n\}$ is **orthonormal** if all the vectors are mutually orthogonal to each other (i.e. $\vec{v}_i^\top \vec{v}_j = 0$ if $i \neq j$) and all are of unit length (i.e. $\|\vec{v}_i\| = 1 = \vec{v}_i^\top \vec{v}_i$). Thus above, A_n has orthonormal columns. We now will generalize what we did earlier by showing that for any matrix Q with n

orthonormal columns, $Q^\top Q = I_n$.

$$Q^\top Q = \begin{bmatrix} — & \vec{q}_1^\top & — \\ & \vdots & \\ — & \vec{q}_n^{\ \top} & — \end{bmatrix} \begin{bmatrix} | & & | \\ \vec{q}_1 & \dots & \vec{q}_n \\ | & & | \end{bmatrix} \tag{27}$$

$$= \begin{bmatrix} \vec{q}_1^\top \vec{q}_1 & \vec{q}_1^\top \vec{q}_2 & \dots & \vec{q}_1^\top \vec{q}_n \\ \vec{q}_2^\top \vec{q}_1 & \ddots & & \vec{q}_2^\top \vec{q}_n \\ \vdots & & \ddots & \vdots \\ \vec{q}_n^\top q_1 & \vec{q}_n^\top \vec{q}_2 & \dots & \vec{q}_n^\top \vec{q}_n \end{bmatrix} \tag{28}$$

$$= \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix} = I_n \tag{29}$$

Remember that we are using the property of orthonormal vectors that $\vec{q}_i^\top \vec{q}_j = 0$ when $i \neq j$ and $\vec{q}_i^\top \vec{q}_i = ||\vec{q}_i||^2 = 1$. Using this, notice that the least-squares estimate with orthonormal vectors simplifies to $\vec{y}_{A_n} = A_n(A_n^\top A_n)^{-1} A_n^\top \vec{y} = A_n(I)^{-1} A_n^\top \vec{y} = A_n A_n^\top \vec{y}$. By direct algebraic manipulation, we formally validated our generalization in equation (26).

Note that the above proof is general and applies to non-square matrices (so $\vec{q}_i$ doesn't need n elements). However, we will specially refer to square matrices whose columns are orthonormal as orthonormal [1] matrices. If a square matrix Q is orthonormal, we can additionally say that $Q^\top Q = QQ^\top = I \implies Q^\top = Q^{-1}$.

Thus, we can see that having orthonormal vectors $\vec{v}_i$ will make least squares must faster and only consist of 1 matrix multiplication. But now you may ask how can we even ensure that $\vec{v}_i$ are orthonormal?

3 Orthonormalization

We want to take a sequence of vectors $\vec{v}_1, \vec{v}_2, \dots \vec{v}_n$ and construct a new sequence of vectors $\vec{q}_1, \vec{q}_2, \dots, \vec{q}_n$ that are orthonormal, i.e. $\vec{q}_i^\top \vec{q}_j = \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{if } i = j \end{cases}$. Additionally, they must satisfy the property that the subspaces spanned by the first k of the original vectors, $\text{Span}(\vec{v}_1, \vec{v}_2, \dots \vec{v}_k)$, are always the same as the subspaces spanned by the first k of the new vectors, $\text{Span}(\vec{q}_1, \vec{q}_2, \dots, \vec{q}_k)$.

This might seem hard but we will start at the beginning and proceed systematically. We first let $\vec{q}_1 = \frac{\vec{v}_1}{||\vec{v}_1||}$ to make it unit norm, and it will have the same span as $\vec{v}_1$. We will then leverage what we know about projections and least-squares from 16A. We know that the residual vector after a projection is always orthogonal [2] to the subspace being projected upon. So to ensure that the new vector $\vec{q}_k$ is orthogonal, we can just remove all parts of $\vec{v}_k$ that lie in the span of our previous vectors. From the previous section and equation (25), since the $\vec{q}_i$ are orthonormal, this is equivalent to subtracting each individual projection onto

[1] In a bit of confusing notation, in math literature you will often see such matrices called orthogonal even when they want to explicitly require that each column is normalized to have unit norm. We will try to use "orthonormal" to avoid this confusion.

[2] Recall that this is how we actually derived the least-squares formula!

all of our previous $\vec{q}_i$ vectors. Consequently, we can recursively define

$$\vec{q}_k = \frac{\vec{v}_k - \sum_{\ell=1}^{k-1} \overset{\text{Proj}_{\vec{q}_\ell} \vec{v}_k}{\vec{q}_\ell(\vec{q}_\ell^\top \vec{v}_k)}}{\|\vec{v}_k - \sum_{\ell=1}^{k-1} \vec{q}_\ell(\vec{q}_\ell^\top \vec{v}_k)\|}. \tag{30}$$

This has unit norm by construction and it is orthogonal to all the previous $\vec{q}_\ell$ because it removes all the projections of the new vector $\vec{v}_k$ onto the subspace spanned by the $\vec{q}_\ell$. This collection also preserves the same span because every new vector $\vec{q}_k$ is just a linear combination of the original vectors. It turns out that this very natural iterative process that we "discovered for ourselves" has a name: **Gram-Schmidt orthonormalization**.

Using generic language, Gram-Schmidt is a procedure that takes a list[3] of linearly independent vectors $\{\vec{v}_1, \dots, \vec{v}_n\}$ and generates an orthonormal list of vectors $\{\vec{q}_1, \dots, \vec{q}_n\}$ that span the same subspaces as the original list. Concretely, we will prove that $\{\vec{q}_1, \dots, \vec{q}_n\}$ from Gram-Schmidt satisfy the following:

$$\{\vec{q}_1, \dots, \vec{q}_n\} \text{ is an orthonormal set of vectors} \tag{31}$$

$$\text{Span}(\{\vec{v}_1, \dots, \vec{v}_k\}) = \text{Span}(\{\vec{q}_1, \dots, \vec{q}_k\}) \quad \forall 1 \le k \le n \tag{32}$$

Proof of Orthonormality (31)**:**

We first start with showing each vector is normal, or unit length. This is true by construction since we are dividing a vector by the norm of that vector, so the result must have norm 1. In other words, for all vectors $\vec{v}$,

$$\left|\left|\frac{\vec{v}}{\|\vec{v}\|}\right|\right| = \frac{1}{\|\vec{v}\|}\|\vec{v}\| = 1 \tag{33}$$

Now onto orthogonality. We will use an inductive approach by first assuming that the first $k-1$ vectors are already orthonormal. We will then show that our new constructed vector $\vec{q}_k$ will be orthogonal to all previous ones, so $\vec{q}_p^\top \vec{q}_k = 0$ for all $p = 1, \dots, k-1$. Note that constant factors don't affect orthogonality, so for simplicity we will call the 1/norm factor in $\vec{q}_k$ some constant A. Then,

$$\vec{q}_p^\top \vec{q}_k = A\vec{q}_p^\top(\vec{v}_k - \sum_{\ell=1}^{k-1} \vec{q}_\ell(\vec{q}_\ell^\top \vec{v}_k)) \tag{34}$$

$$= A(\vec{q}_p^\top \vec{v}_k - \sum_{\ell=1}^{k-1} \vec{q}_p^\top \vec{q}_\ell(\vec{q}_\ell^\top \vec{v}_k)) \tag{35}$$

Since we assumed the first $k-1$ vectors are all orthonormal, the $\vec{q}_p^\top \vec{q}_\ell$ will cause the only nonzero in the summation to occur when $\ell = p$. Then,

$$\vec{q}_p^\top \vec{q}_k = A(\vec{q}_p^\top \vec{v}_k - \vec{q}_p^\top \vec{q}_p(\vec{q}_p^\top \vec{v}_k)) \tag{36}$$

$$= A(\vec{q}_p^\top \vec{v}_k - \vec{q}_p^\top \vec{v}_k) = 0 \tag{37}$$

where we use the fact that $\vec{q}_p^\top \vec{q}_p = 1$ since we assumed it is orthonormal. Thus, for any k we know that the next vector we construct will be orthogonal to all of our previous vectors. If this idea of induction is confusing, don't worry as CS 70 will cover it in much more detail.

[3]The fact that these are lists and not sets matters. The vectors are ordered. We don't just want the overall spans to be the same, we want the spans to be the same as we walk down the lists together.

Proof of Equivalent Span (32):

Again, we will use an inductive approach by assuming that the first $k-1$ vectors of V and Q both span the same space, and wanting to show the same holds for the first k vectors. This is true for our base case since $\vec{q}_1 = \vec{v}_1/\|\vec{v}_1\|$. Now all we need to show is that $\vec{v}_k$ can be written as a linear combination of $\{\vec{q}_1, \dots, \vec{q}_k\}$. By construction of $\vec{q}_k$, that is exactly true with

$$\vec{v}_k = \left\| \vec{v}_k - \sum_{\ell=1}^{k-1} (\vec{q}_\ell^\top \vec{v}_k)\vec{q}_\ell \right\| \vec{q}_k + \sum_{\ell=1}^{k-1} (\vec{q}_\ell^\top \vec{v}_k)\vec{q}_\ell \tag{38}$$

Thus, for all k, the spans will be equivalent.

3.1 Example for three vectors

The above might have been a bit fast, so let's walk through the reasoning for why (30) works for the case of three vectors to make sure it is clear.

Consider three vectors $\{\vec{v}_1, \vec{v}_2, \vec{v}_3\}$ that are linearly independent of each other.

- **Step 1:** Find unit vector $\vec{q}_1$ such that $\text{Span}\big(\{\vec{q}_1\}\big) = \text{Span}\big(\{\vec{v}_1\}\big)$.
  Since span($\{\vec{v}_1\}$) is a one dimensional vector space, we can simply scale $\{\vec{v}_1\}$ so that it is unit norm:

$$\vec{q}_1 = \frac{\vec{v}_1}{\|\vec{v}_1\|}. \tag{39}$$

- **Step 2:** Given $\vec{q}_1$ from the previous step, find $\vec{q}_2$ such that $\text{Span}\big(\{\vec{q}_1, \vec{q}_2\}\big) = \text{Span}\big(\{\vec{v}_1, \vec{v}_2\}\big)$ and orthogonal to $\vec{q}_1$. We know that $\vec{v}_2 -$ (the projection of $\vec{v}_2$ on $\vec{q}_1$) would be orthogonal to $\vec{q}_1$ from 16A. So first, we can find the error or residual

$$\vec{e}_2 = \vec{v}_2 - \left(\vec{q}_1^\top \vec{v}_2\right)\vec{q}_1, \tag{40}$$

  which is orthogonal to $\vec{q}_1$. Then, we can normalize to get $\vec{q}_2 = \frac{\vec{e}_2}{\|\vec{e}_2\|}$. Note that these operations preserve the span because $\vec{q}_1$ and $\vec{q}_2$ are just linear combinations of $\vec{v}_1$ and $\vec{v}_2$ and vice-versa.

- **Step 3:** Now given $\vec{q}_1$ and $\vec{q}_2$ in the previous steps, we would like to find $\vec{q}_3$ such that $\text{Span}\big(\{\vec{q}_1, \vec{q}_2, \vec{q}_3\}\big) = \text{Span}\big(\{\vec{v}_1, \vec{v}_2, \vec{v}_3\}\big)$. We know that the projection of $\vec{v}_3$ onto the subspace spanned by $\vec{q}_1, \vec{q}_2$ is

$$\left(\vec{q}_2^\top \vec{v}_3\right)\vec{q}_2 + \left(\vec{q}_1^\top \vec{v}_3\right)\vec{q}_1. \tag{41}$$

  Consequently, we know that the error/residual

$$\vec{e}_3 = \vec{v}_3 - \left[\left(\vec{q}_2^\top \vec{v}_3\right)\vec{q}_2 + \left(\vec{q}_1^\top \vec{v}_3\right)\vec{q}_1\right]. \tag{42}$$

  is orthogonal to both $\vec{q}_1$ and $\vec{q}_2$. Normalizing, we have $\vec{q}_3 = \frac{\vec{e}_3}{\|\vec{e}_3\|}$.

3.2 Gram-Schmidt Algorithm

We can reformulate the mathematical definition in (30) as an iterative algorithm as follows:

Inputs

- A list of linearly independent vectors $\{\vec{v}_1, \dots, \vec{v}_n\}$.

Outputs

- An orthonormal list of vectors $\{\vec{q}_1, \dots, \vec{q}_n\}$, where $\text{Span}\big(\{\vec{v}_1, \dots, \vec{v}_k\}\big) = \text{Span}\big(\{\vec{q}_1, \dots, \vec{q}_k\}\big)$ for all $1 \le k \le n$.

Gram Schmidt Procedure

- compute $\vec{q}_1 : \vec{q}_1 = \frac{\vec{v}_1}{\|\vec{v}_1\|}$
- for $(i = 2 \dots n)$:
  (a) Compute the vector $\vec{e}_i$, such that $\text{Span}\big(\{\vec{q}_1, \dots, \vec{q}_{i-1}, \vec{e}_i\}\big) = \text{Span}\big(\{\vec{v}_1, \dots, \vec{v}_i\}\big)$:

$$\vec{e}_i = \vec{v}_i - \sum_{\ell=1}^{i-1} \left(\vec{q}_\ell^\top \vec{v}_i\right) \vec{q}_\ell \tag{43}$$

  (b) Normalize to compute $\vec{q}_i$ itself: $\vec{q}_i = \frac{\vec{e}_i}{\|\vec{e}_i\|}$.

4 QR Decomposition

If we take a look at equation (38) we can see that we can write $\vec{v}_k$ as linear combination of only $\vec{q}_1, \dots, \vec{q}_k$, and that this holds for all $1 \le k \le n$. This should remind us of a triangular structure similar to what you've seen in Gaussian elimination, and this will actually lead us to a new discovery.

Simplifying equation (38) with the notation from section 3.2, we get that the norm term $\left\|\vec{v}_k - \sum_{\ell=1}^{k-1}(\vec{q}_\ell^\top \vec{v}_k)\vec{q}_\ell\right\| = \|\vec{e}_k\|$. Then equation (38) for $k = 1, 2, \dots, n$ becomes

$$\vec{v}_1 = \|\vec{e}_1\|\, \vec{q}_1 \tag{44}$$

$$\vec{v}_2 = \|\vec{e}_2\|\, \vec{q}_2 + \overbrace{(\vec{q}_1^\top \vec{v}_2)\vec{q}_1}^{\text{Proj}_{\vec{q}_1} \vec{V}_2} = \|\vec{e}_2\|\, \vec{v}_2 - \|\vec{e}_2\| \left(\vec{q}_1^\top \vec{v}_2\right) \tag{45}$$

$$\vdots$$

$$\vec{v}_n = \|\vec{e}_n\|\, \vec{q}_n + \sum_{\ell=1}^{n-1}(\vec{q}_\ell^\top \vec{v}_n)\vec{q}_\ell \tag{46}$$

Converting the above equations to matrix multiplication form, we get

$$\begin{bmatrix} \vec{v}_1 & \vec{v}_2 & \dots & \vec{v}_n \end{bmatrix} = \begin{bmatrix} \vec{q}_1 & \dots & \vec{q}_n \end{bmatrix} \begin{bmatrix} \|\vec{e}_1\| & \vec{q}_1^\top \vec{v}_2 & \vec{q}_1^\top \vec{v}_3 & \dots & \vec{q}_1^\top \vec{v}_n \\ 0 & \|\vec{e}_2\| & \vec{q}_2^\top \vec{v}_3 & \dots & \vec{q}_2^\top \vec{v}_n \\ 0 & 0 & \|\vec{e}_3\| & \dots & \vec{q}_3^\top \vec{v}_n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \|\vec{e}_n\| \end{bmatrix} \tag{47}$$

$$V = QR \tag{48}$$

This gives us a way to transform between the original vectors $\vec{v}_i$ and the Gram-Schmidt orthonormalized vectors $\vec{q}_i$ through an upper-triangular matrix R. Additionally, note that every entry of R will already be computed by the Gram-Schmidt algorithm, so we can directly construct this matrix as we perform the algorithm.

An alternate view is since we can apply Gram-Schmidt to any set of vectors, it shows us that *any* matrix V can be decomposed into a matrix with orthonormal columns Q and an upper-triangular matrix R, and this is called the **QR Decomposition**. When V is square, we get the additional property that Q will be square and thus an orthonormal matrix.

5 Speeding Up Least Squares

Now with these tools under our belt, we can go back to our initial issue of speeding up our system ID problems from Section 1. What we can now do is apply Gram-Schmidt orthonormalization to the columns of D_n, starting from the rightmost column to the leftmost, which will return a set of orthonormal vectors $\vec{q}_0, \vec{q}_1, \ldots, \vec{q}_n$ that span the same subspaces as the columns in D_n.

$$Q_1 = \begin{bmatrix} \vec{q}_1 & \vec{q}_0 \end{bmatrix} \tag{49}$$

$$Q_2 = \begin{bmatrix} \vec{q}_2 & \vec{q}_1 & \vec{q}_0 \end{bmatrix} \tag{50}$$

$$\vdots$$

$$Q_n = \begin{bmatrix} \vec{q}_n & \ldots & \vec{q}_0 \end{bmatrix} \tag{51}$$

Since all the Q_i have orthonormal columns, we know $Q_i^\top Q_i = I$ so the least squares solution $(Q_i^\top Q_i)^{-1} Q_i^\top \vec{y}$ simplifies drastically to just $Q_i^\top \vec{y}$. We can then solve the least squares problems $Q_i \vec{w}_i \approx \vec{y}$ with

$$\vec{\hat{w}}_1 = Q_1^\top \vec{y} \tag{52}$$

$$\vdots$$

$$\vec{\hat{w}}_n = Q_n^\top \vec{y} \tag{53}$$

which is very easy computationally. Importantly, each successive problem only requires 1 more dot product with $\vec{q}_i$ and $\vec{y}$ when compared to the last problem, as opposed to recalculating the entire least squares solution.

However, note that the parameters we get are some $\vec{\hat{w}}_i$ which are different than the original $\vec{\hat{x}}_i$ parameters due to changing the data from A_i to Q_i. Thus, we must have some post-processing step to convert each $\vec{\hat{w}}_i$ to $\vec{\hat{x}}_i$. What is the relation between them? Well we know that since A_i and Q_i have the same column space, then the projection of $\vec{y}$ onto their column space is the same so

$$A_i \vec{\hat{x}}_i = Q_i \vec{\hat{w}}_i \tag{54}$$

We now use the QR decomposition from (48) to say that $A_i = Q_i R_i$ so

$$A_i \vec{\hat{x}}_i = Q_i R_i \vec{\hat{x}}_i = Q_i \vec{\hat{w}}_i \tag{55}$$

$$R_i \vec{\hat{x}}_i = \vec{\hat{w}}_i \tag{56}$$

where we left multiplied the first equation by $Q_i^\top$ to get the second equation. Now we just need to solve this $n \times n$ system of equations to get $\vec{\widehat{x}}_i$. But a key property is that R is upper-triangular, and so from our knowledge of Gaussian elimination, we just need to back-substitute all the equations starting from the bottom row. This will just take time proportional to the number of entries, so it will take $O(n^2)$ time which is faster than the time for a generic inverse calculation $O(n^3)$.

Overall, it's now much faster to try fit a one higher $n+1$ dimension model if we want — instead of doing a whole $(D_{n+1}^\top D_{n+1})^{-1} D_{n+1}^\top \vec{y}$ calculation again in $O(n^3)$ time, it becomes one extra iteration of Gram-Schmidt to orthonormalize our new column to $\vec{q}_{n+1}$ in $O(nm)$ time, then 1 extra dot product to get the new entry of $\vec{\widehat{w}}_{n+1}$ in $O(m)$ time, and then applying the post-processing step by back-substituting R_{n+1} in $O(n^2)$ time. This gives an overall runtime of $O(n^2)$ which is much more efficient than the default $O(n^3)$.

Contributors:

- Ashwin Vangipuram.
- Anant Sahai.
- Jennifer Shih.
- Rachel Hochman.
- Vasuki Narasimha Swamy.
- Steven Cao.

where we left multiplied the first equation by $Q_i^\top$ to get the second equation. Now we just need to solve this $n \times n$ system of equations to get $\vec{\hat{x}}_i$. But a key property is that R is upper-triangular, and so from our knowledge of Gaussian elimination, we just need to back-substitute all the equations starting from the bottom row. This will just take time proportional to the number of entries, so it will take $O(n^2)$ time which is faster than the time for a generic inverse calculation $O(n^3)$.

Overall, it's now much faster to try fit a one higher $n+1$ dimension model if we want — instead of doing a whole $(D_{n+1}^\top D_{n+1})^{-1} D_{n+1}^\top \vec{y}$ calculation again in $O(n^3)$ time, it becomes one extra iteration of Gram-Schmidt to orthonormalize our new column to $\vec{q}_{n+1}$ in $O(nm)$ time, then 1 extra dot product to get the new entry of $\vec{\hat{w}}_{n+1}$ in $O(m)$ time, and then applying the post-processing step by back-substituting R_{n+1} in $O(n^2)$ time. This gives an overall runtime of $O(n^2)$ which is much more efficient than the default $O(n^3)$.

Contributors:

- Ashwin Vangipuram.
- Anant Sahai.
- Jennifer Shih.
- Rachel Hochman.
- Vasuki Narasimha Swamy.
- Steven Cao.