03. Digression on Row Operations and RREF

Row operations

- Scaling a row
- Swapping two rows
- Adding a scaling of one row to another *(Workhorse)*

Note that different writers will group row operations differently. So take only the *amalgam* of the discussion below, without literal fixing of particular details. The first row operation (for us) is *scaling* by a non-zero factor, say α. For example:

$$\begin{pmatrix} \frac{1}{2} & \frac{1}{2} & \frac{5}{2} \end{pmatrix} \longrightarrow 2 \cdot \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & \frac{5}{2} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 5 \end{pmatrix} .$$

This is the only row operation that involves just a single row. The operation is used largely for *normalization*, in analogy with setting a standard length, e.g., the meter or a standard temperature scale, e.g., Celsius. Typically, we'll take a non-zero row of a matrix (that is, a row not all of whose entries are zero), and scale the row so as to make the leftmost nonzero entry equal to 1. (We'll deal with zero rows, i.e., rows all of whose entries are zero, below.) A "leading" leftmost 1 entry in a row will be called a *pivot* in some contexts and will be used to eliminate variables.

Making The Swap

The second row operation (for us) consists of swapping, or equivalently, *transposing, permuting* two rows, for example,

$$\begin{pmatrix} 0 & 2 & 1 \\ 1 & -4 & 5 \end{pmatrix} \longrightarrow \begin{pmatrix} 1 & -4 & 5 \\ 0 & 2 & 1 \end{pmatrix} .$$

This helps us realize conventions, e.g., placing key '1' entries in the upmost and lefmost positions possible, or placing all identically-zero rows at the bottom. In particular, this helps us order *pivots*. (See about pivots below or in the book.) We generally do this to place a non-zero entry in a strategic location where we had a zero entry before. For instance, take a matrix of the form $\left(\begin{smallmatrix} 0 & 4 & \pi \\ 2 & 1 & 5 \end{smallmatrix}\right)$. We wish to operate on the matrix to produce a '1′ in the top of the first column. Thus we swap rows: $I \leftrightarrows II$. We obtain $\left(\begin{smallmatrix} 2 & 1 & 5 \\ 0 & 4 & \pi \end{smallmatrix}\right)$. We now scale the top row by 1/2, obtaining the matrix $\left(\begin{smallmatrix} 1 & (1/2) & (5/2) \\ 0 & 4 & \pi \end{smallmatrix}\right)$, which boasts a pivot column (the first column), and is ready for reduction work on the second column. Let's scale the second row by a factor of 1/4, obtaining $\left(\begin{smallmatrix} 1 & (1/2) & (5/2) \\ 0 & 1 & (\pi/4) \end{smallmatrix}\right)$.

The WorkHorse

The third row operation is the workhorse–it does the heavy lifting of the matrix reduction process. It involves taking one row and subtracting from it a scaling of another row. (Note that *subtracting* could be *adding* if we choose a negative scale factor.) Here is an example, where we take row II and subtract π times row I from it:

$$\begin{pmatrix} 1 & 2 & 1 \\ \pi & -4 & 5 \end{pmatrix} \longrightarrow \begin{pmatrix} 1 & 2 & 1 \\ 0 & -4-2\pi & 1-5\pi \end{pmatrix} .$$

Generally, this operation is used to zero out an entry in a given row, taking care not to alter hard-earned 0 and 1 entries already established in other key locations. Introducing a zero entry is tantamount to eliminating a variable from one equation.

For instance, take the aforementioned matrix $\left(\begin{smallmatrix} 1 & (1/2) & (5/2) \\ 0 & 1 & (\pi/4) \end{smallmatrix}\right)$ and operate on it to zero out the middle entry of the top row. We do this by subtracting (1/2) of row II from row I: $I \to I - (1/2)II$, obtaining $\left(\begin{smallmatrix} 1 & 0 & (\frac{5}{2} - \frac{\pi}{8}) \\ 0 & 1 & (\pi/4) \end{smallmatrix}\right)$. (Sorry for the inelegant typesetting.) This last matrix is in as good a form as we can obtain using row operations and the conventions we set. It is in Row Reduced Echelon Form, or *RREF*. The reader is invited to try operating on the matrix to improve it presentation. In each case, however, one can find a counterargument to any purported improvement.

There is a natural objection for the reader to make: although we called the third row operation type our *workhorse* and stated that it does most of the heavy lifting, we only employed it once. But that's just because of the special form of the example we invoked. For a more typical example. try row reducing the 3×3 walking-around matrix, $\left(\begin{smallmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{smallmatrix}\right)$.

Cautions

We must caution the reader to resist any temptations to generalize the third row operation type to forms such as $I \to 7I - 5II$, or even more general forms involving more than two rows, e.g., $I \to 7I - 5II + 9III$. Such operations may be realized through amalgamation of our existing operations. For humans this one-small-step-at-a-time approach is likely to reduce errors. For robots (read: algorithms), this approach is more systematic and is likely to result in fewer software bugs.

Getting to RREF: Gauss-Jordan

The *Gauss-Jordan* algorithm, dating back hundreds of years before Carl Friedrich Gauss (1777-1855) and Wilhelm Jordan (1842-1899), is a systematic recipe for row operating on matrices to put them in the best possible reduced form (*best* relative to certain conventions). We will describe this procedure, which takes a matrix M, viewed as the coefficient matrix of a linear system, and systematically row-operates on it to yield the best (in general), most simplified structure from which the key features of the linear system may be ascertained.

Prospecting For Pivots

We examine M and look for a non-zero row with the leftmost non-zero entry among our options. If we find none, the matrix is identically zero and our process is done. (The zero matrix needs no introduction nor elaboration; do note though that there is a zero matrix for each $p \times q$ size and shape.) If there are several rows with equally leftist non-zero entries we choose the highest one. We switch this row with the top row of M, if need be. Then we scale the row to normalize this entry, i.e., transform it into a 1. We will then have a matrix of the following form:

$$\begin{pmatrix} 0 & \dots & 0 & 1 & * & \cdots & * \\ 0 & \dots & 0 & * & * & \cdots & * \\ \cdot & \cdots & \cdot & \cdot & \cdot & \cdots & \cdot \\ 0 & \cdots & 0 & * & * & \cdots & * \end{pmatrix}.$$

In case our leftmost non-zero entry is the first one in its row the matrix at hand will not require the display of arbitrarily many zero columns, and hence will have this form: $\begin{pmatrix} 1 & * & \cdots & * \\ * & * & \cdots & * \\ \cdot & \cdot & \cdot & \cdot \\ * & * & \cdots & * \end{pmatrix}$. We call the displayed 1 a *candidate pivot*, soon to be a full fledged pivot, and use the work-horse row operation to zero out all the entries below it in its column. The result looks like this:

$$\begin{pmatrix} 0 & \dots & 0 & 1 & * & \cdots & * \\ 0 & \dots & 0 & 0 & * & \cdots & * \\ \cdot & \cdots & \cdot & \cdot & \cdot & \cdots & \cdot \\ 0 & \dots & 0 & 0 & * & \cdots & * \end{pmatrix}.$$

For illustration, see the one use of the workhorse operation in an earlier matrix above. Once again, we may not need the arbitrary number of zero columns to the left in this last display, as that number could be zero; from now on we'll take this point as understood tacitly.

Reaching For The Stars

We now concentrate on the lower right box full of stars. (One might say *reach for the stars*.)

$$\left(\begin{array}{cccc|ccc} 0 & \dots & 0 & 1 & * & \cdots & * \\ \hline 0 & \dots & 0 & 0 & * & \cdots & * \\ \cdot & \cdots & \cdot & \cdot & \cdot & \cdots & \cdot \\ 0 & \dots & 0 & 0 & * & \cdots & * \end{array}\right).$$

We simply proceed with the same ideas as above, but using the entries in the lower right box for motivation. We start by looking within this box for a *local row* (or *relative row*), that is, a row within the lower right box, with the leftmost nonzero entry. If we find none, we have a matrix of the following form:

$$\left(\begin{array}{cccc|ccc} 0 & \dots & 0 & 1 & * & \cdots & * \\ \hline 0 & \dots & 0 & 0 & 0 & \cdots & 0 \\ \cdot & \cdots & \cdot & \cdot & \cdot & \cdots & \cdot \\ 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \end{array}\right).$$

This is already "unimproveable" (read: RREF), and we are done with it for now. If there is a nonzero entry, we proceed as before, choosing the highest among the leftmost non-zero entries and moving it to the *local top*, that is, the top of the lower right box. The row operations we use are not local or relative. They are row operations on the full, original-sized matrix. But the de-facto action is entirely within the lower right box; the entries outside this box will not be altered by these row operations. (By the way, what's *de facto*?) As we proceed, we will introduce local pivots within the lower right box, and then we will be left with a small lower-right box within the lower right box, and we'll process that using the same ideas. This is an example of *recursion*, a concept that appears in many parts of mathematics and is particularly useful in computation. It is also "green", manifesting an efficient recycling of ideas.

Cleaning Up

When there are no local lower-right boxes left, we have one last task, a "cleaning-up" opportunity. Each local pivot introduced will be the only non-zero entry within its local column, but there may be additional non-zero entries higher up. For instance, in our early 2×3 example above we reached the matrix $\begin{pmatrix} 1 & (1/2) & (5/2) \\ 0 & 1 & (\pi/4) \end{pmatrix}$. Notice that this matrix has no lower-right box. (It does have a right box, but not a *lower* right box.) The middle entry of the second row is a relative pivot and becomes a full-fledged (global) pivot once we do our "clean-up" work, zeroing the entry above it via $I \to I - (1/2)II$, obtaining $\begin{pmatrix} 1 & 0 & (\frac{5}{2} - \frac{\pi}{8}) \\ 0 & 1 & (\frac{\pi}{4}) \end{pmatrix}$. This last matrix is in RREF. The reader may wish to locate a row reduction narrative for a larger matrix, say 5×5, and follow the concrete steps of the Gauss-Jordan process while reading the prose above. Many books offer concrete examples of Gauss-Jordan reduction of explicit, and not so small-sized matrices.

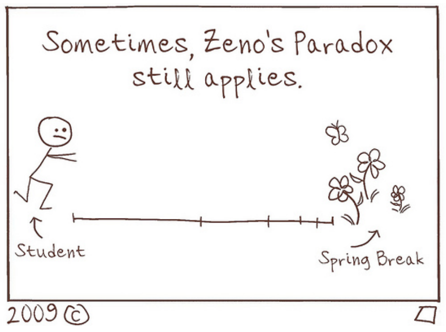
Not Walking Around In Circles

A big risk for hikers lost in the woods is wondering and returning to the same spot. This can also happen with row reduction. A simple and silly instance of this is in swapping two rows, swapping again and again. This is redundant. Indeed, redundancy can occur not only in systems of linear equations, but also in row reduction strategies. Linear systems can have obvious, silly redundancies, e,g.

$$x = 4 \quad ; \quad 2x = 8;$$

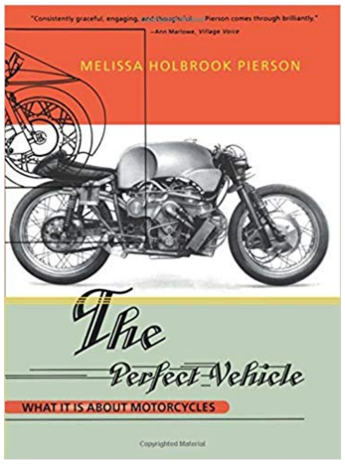
there can also be more subtle, "hidden" redundancies, as in the coefficient walking-around matrix (see the 3×3 version of this matrix, above). Similarly, row operation procedures can contain subtle, hidden redundancies that bring one back to the same matrix, or one no better than the original. The Gauss-Jordan procedure is designed to avoid this by making definitive progress at each step. (Future hikers may wish to consult the hiking literature for hints on how to avoid walking around in circles in the woods.) But even making definite (definitive?) progress at each step is not enough. The reader is invited to look up *Zeno's Paradox*, where definite progress at each step does not result in a desired end. In contrast, the Gauss-Jordan procedure results in a "best", "unimprovable" matrix in a finite number of steps. (The number is related to the size of the matrix and is easily estimated.)

(Continued on next page)



The Unimprovable:
Row Reduced Echelon form

One rarely runs across unimprovables in everyday life. Can one think of an example of unimprovable software, of an unimprovable automobile, a bicycle? See, for instance, the book [1]. (Some may argue that a *Stradivarius* violin is unimprovable, perhaps with justification.) In any case, Linear Algebra, and other parts of mathematics do have unimprovable objects, within an appropriate and reasonable context.


"Consistently graceful, engaging, and thoughtful... Pierson comes through brilliantly." —Ann Marlowe, Village Voice
MELISSA HOLBROOK PIERSON
The Perfect Vehicle
WHAT IT IS ABOUT MOTORCYCLES
Copyrighted Material

Starting with the coefficient matrix of a linear system we can perform row operations, e.g., following the Gauss-Jordan procedure above, to obtain a coefficient matrix with the following properties:

- If the matrix has any identically zero rows, these rows are as far down as possible.
- In a non-zero row the leftmost non-zero entry is 1. *We'll call this a **pivot*** entry.
- A pivot entry is the only $\neq 0$ entry in its column.

  (One may model this as an *insecurity* condition–pivots don't like competition from other non-zero entries in their column, though they are less fussy about rows; pivots are *column-insecure*.)
- Given two pivots, the rightmost of the two is also the lowest.

The conditions manifest the *Row Reduced Echelon Form*, or RREF. Note that other presenters may group the conditions in different ways, and may avoid silly and inelegant annotations of the sort found above. One may wish to give the conditions descriptive names, but to be sure, these will not be standard. The first, or *zeros on the bottom* condition is a convention, helping ensure uniqueness of the RREF form of a given matrix, and helping standardize row reduction algorithms. The same may be said of the last condition:

> *Pivots go from left to right as they go from top to bottom.*

The second condition is a kind of *normalization* condition, similar in motivation to setting a standard measure of length (meter), temperature, etc. This too helps ensure uniqueness of the RREF of a matrix and guides algorithm development.

Once a linear system is in RREF many of its properties become transparent and, in particular, it is easy to describe its solution set. This is detailed elsewhere. The question is: what makes RREF unimprovable? What else could one ask from a row-equivalent form of a matrix beyond what RREF provides? The reader is invited to ponder this.

References

[1] *The Perfect Vehicle: What It Is About Motorcycles* by Melissa, Holbrook Pierson, W. W. Norton & Company (1998).