CONTS

Linear Algebra in Derive 6

David Jeffrey,

Applied Mathematics, UWO
and
ORCCA - Ontario Research Centre for Computer Algebra

Slides given at TIME 2004 conference at ETS in Montreal. Please read in conjunction with the Derive worksheets whose names are given in square brackets throughout these notes.

New FACTOR commands in Derive 6

In Derive 6, there are two new options for the FACTOR command, whether it is used from the menu or from the command line.

1. TURING FACTORING:

The command `FACTOR([-59, 93, 91; a, -98, -8], Turing, a)` gives

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ -a/59 & 1 \end{pmatrix} \begin{pmatrix} -59 & 0 \\ 0 & (93a-5782)/59 \end{pmatrix} \begin{pmatrix} 1 & -93/59 \\ 0 & 1 \end{pmatrix}$$
$$\begin{pmatrix} 1 & 0 & 8174/(93a-5782) \\ 0 & 1 & (91a-472)/(93a-5782) \end{pmatrix}$$

2. QR FACTORING:

The command `FACTOR([-59, 93, 91; a, -98, -8], Schmidt, a)` gives

$$\begin{pmatrix} -59/\sqrt{a^2+3481} & a\,\mathrm{sgn}(93a-5782)/\sqrt{a^2+3481} & 0 \\ a/\sqrt{a^2+3481} & 59\,\mathrm{sgn}(93a-5782)/\sqrt{a^2+3481} & 0 \end{pmatrix}$$
$$\begin{pmatrix} \sqrt{a^2+3481} & -(98a+5487)/\sqrt{a^2+3481} & -(8a+5369)/\sqrt{a^2+3481} \\ 0 & |93a-5782|/\sqrt{a^2+3481} & (91a-472)\,\mathrm{sgn}(93a-5782)/\sqrt{a^2+3481} \\ 0 & 0 & 0 \end{pmatrix}$$

Everyone Loves Row Reduction

The time-honoured way to solve linear systems is through row reduction to "Reduced Row Echelon Form" RREF.

- I am really assuming you know about RREF, but let us see a quick review.
- Equations have zero, one or families of solutions.
- Derive has had the ROW_REDUCE command for a long time.

[The Derive worksheet `rref.dfw` reviews how RREF is used.]

So RREF does everything ??

The definition of RREF assumes *numerical* matrices. A teacher using Derive tried adding symbols to his matrix.
[The Derive worksheet `turing.dfw` describes a bug report from a teacher to Texas Instruments 'TI Cares'.]

- As soon as you have symbols you have the possibility of CASES.
- The mathematical definition of RREF forces Derive to throw away case information.
- Reporting case information is a big issue for *all* CAS.

How to handle case information

Let us think about presenting case information. It is a big topic so we shall only scratch the tip of the iceberg.

- Should the system compute all cases in advance for the user? (Perhaps the user is not interested in the mass being negative, or the distance to Quebec being complex.)
- Should the system define (or guess) a "generic" case?
- What about invisible failures?

  [The Derive worksheet `visible.dfw` introduces the ideas of visible and invisible failure in a computer algebra system.]

One solution

We can debate about the best ways to handle special cases for all problems over Blanche de Chambly, but for the special case of linear algebra there is an interesting possibility.
One way to provide a solution is to bring some modern (presently more advanced) attitudes into our linear algebra course.

- Modern Linear Algebra emphasizes *factoring* matrices.
- Factoring re-arranges information, it does not lose it.
- Traditional definitions of RREF force us to lose information.

Turing factors

- Alan Turing [photo] wrote a paper in which he pointed out that row reduction can be thought of as factoring.
- Turing's factoring became known for *square* matrices as "LU" factoring.
- We can define Turing factors for any matrix.

$$A = PLDUR$$

Advantages of Turing factors

- They provide a check for correctness: $\det D \neq 0$.
- They combine several topics into one. Usually, RREF and LU are treated in books as separate topics.
- They allow for conflicting definitions of the same thing. Textbooks are divided between Doolittle LU and Crout LU.

Anatomy of Turing factors

- The R factor is the RREF, and we saw how to use that.
- If the matrix is invertible, then $R = I$ and we recover LU factors. These are popular for the solution of multiple equations $Ax_1 = b_1, Ax_2 = b_2, \ldots$. Because L and U are triangular matrices, $LUx_2 = b_2$ is easier to solve than the un-factored equation.
- Implementation note: Derive offers only one pivoting strategy. Big numbers are better than small numbers are better than symbols.

  [The Derive worksheet `pivot.dfw` shows how Derive selects a pivot.]

CONTS

So I have to change what I teach ??

Turing factoring, like LU factoring, uses exactly the same operations as Gaussian elimination, with the additional feature that the multipliers are remembered and stored in L. Thus in principle there is only book-keeping work to convert a reduction to RREF into Turing factoring.

- For beginning students this is still "a bit rich"
- I tried teaching Turing factoring for several years and the students put up with it.
- I think now that row reduction as a white box and Turing as a black box, now that Derive can do the boring work, is better.
- Some good student will try putting a symbol in a matrix sooner or later. Adventure is why we use Derive !
- As enrichment material, and for those hard problems you like to have Derive for, now you have Turing factors "on the menu".

Gram–Schmidt and QR

Another time-honoured topic is Gram–Schmidt Orthogonalization.

- As with row reduction, Gram–Schmidt can be taught by itself or as a matrix factoring.
- The matrix factoring is "QR" and is used extensively in numerical linear algebra.

Basic Idea

We start with a set of basis vectors, for example, $\{v_1, v_2, v_3\}$, where

$$\begin{aligned} v_1 &= [1, 2, 4, -5] \ , \\ v_2 &= [1, -1, 2, 1] \ , \\ v_3 &= [2, 3, -1, 0] \ , \end{aligned}$$

and we have to compute an equivalent basis. In this case the answer is

$$\begin{aligned} e_1 &= [1, 2, 4, -5]/\sqrt{46} \ , \\ e_2 &= [22, -25, 42, 28]/\sqrt{3657} \ , \\ e_3 &= [374, 370, -81, 158]/\sqrt{308301} \ . \end{aligned}$$

The e_i are all unit length and form an *Orthonormal Basis*

Gram–Schmidt process

We normalize the first vector:

$$e_1 = v_1/\|v_1\|$$

For the second vector w_2, we remove the part of v_2 in the e_1 direction:

$$v_2 - (v_2 \cdot e_1)e_1$$

and then normalize. For the third vector, we remove both e_1 and e_2 components:

$$v_3 - (v_3 \cdot e_1)e_1 - (v_3 \cdot e_2)e_2$$

and normalize. [The Derive worksheet `Gramschmidt.dfw` reviews the process, and illustrates the connection with QR.]

From Gram–Schmidt to QR

By convention, the textbooks put vectors into the *columns* of a matrix.

$$\{[1,2,4,-5],[1,-1,2,1],[2,3,-1,0]\} \Rightarrow$$

$$\begin{bmatrix} 1 & 1 & 2 \\ 2 & -1 & 3 \\ 4 & 2 & -1 \\ -5 & 1 & 0 \end{bmatrix}$$

From this we factor

$$A = QR$$

What is the use of QR?

The most popular application is to solving over-determined systems.

$$\begin{aligned} 2x + 3y &= 2 \\ -x + y &= 1 \\ x - 2y &= 0 \end{aligned}$$

In matrix form

$$\begin{pmatrix} 2 & 3 \\ -1 & 1 \\ 1 & -2 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 2 \\ 1 \\ 0 \end{pmatrix}$$

- Gauss: the least-squares approximate solution: normal equations $A^T AX = A^T B$.
- Normal equations lose numerical accuracy.
- They are an incomprehensible recipe.

QR factors do better on each count.

Basis of graphical presentation.
Convert problem from equation solving to surface transformation.

$$AX = B \Rightarrow B = AX \Rightarrow Y = AX$$

For our over-determined system, X has dimension 2 and Y has dimension 3. Therefore we have to transform a 2-D surface in a plane into a 2-D surface in 3-D. [The Derive worksheet `LS.dfw` shows how to set up a graphical demonstration of transformations in Linear Algebra. It includes a short program that automates the tasks of transforming sets of points. Then it shows the graphical interpretation of an inconsistent system. Derive makes this much better than a textbook, because you can move the 3-D plots around and understand what they are doing. Notice in the worksheet how the original point B is projected onto the surface in a perpendicular manner. By grabbing the 3-D plot with your mouse and looking at the surface side on, you should see the point on the surface perpendicularly below the original B.
Before I could use Derive, my lectures always used a 2-D to 2-D system because that was easy to draw on a board. The problem with inconsistent systems in 2-D is that they have to be degenerate, and so the classroom demonstration was always a bit of a fudge. With Derive, however, the 3-D system is easier to present.]