Setting Linear Algebra Problems

John D. Steele
School of Mathematics
University of New South Wales
Sydney NSW 2052
Australia
j.steele@unsw.edu.au

Abstract: In this report I collect together some of the techniques I have evolved for setting linear algebra problems, with particular attention payed towards ensuring relatively easy arithmetic. Some are given as MAPLE routines.

AMS Classification: 15-00, 15-02, 15-04, 15A36.

1 Introduction

When I first began teaching Linear Algebra at the University of New South Wales, I was disappointed by the level of arithmetic competence of my students, and annoyed to find that problems were being set where this lack of competence meant one could never really see whether students understood the material and techniques or not. I began to think through these problems backwards, and ask "How can I set these problems so that the arithmetic does not obscure the mathematics?"

I am sure many people have gone through the same process and evolved their own ideas and *ad hoc* methods of conjuring up the appropriate matrix or set of vectors, but I looked in vain for anything on paper. This report is intended to set down some of the useful tricks I came up with while setting linear algebra problems. The point of this exercise is to pay careful attention to having the arithmetic as easy as can be arranged. This latter requirement means that vectors and matrices should be integer where possible, or rational numbers with small denominator, or simple, small surds if that is the best that can be done. Of course, there is a downside to this, as it is also a useful lesson for students that the numbers do not always work out, but one can also use these tricks to make sure that they do not.

I will only be concerned with $\mathbb{R}^n$, given that any other finite-dimensional vector space over $\mathbb{R}$ can be isometrically mapped to a suitable $\mathbb{R}^n$. Of course, $\mathbb{R}$ can usually be replaced with $\mathbb{C}$.

Most of these tricks are essentially routine applications of basic linear algebra, and therefore most of the references given are to textbooks. A few use slightly more sophisticated ideas to ensure the methods work. I expect no result given here is either deep or new — most are very simple and/or very old. I would expect most readers reaction to them would be along the lines of "Well, obviously!" A few are perhaps not as well known, or as fully appreciated, as they should be.

Throughout this report I will refer to MAPLE procedures that I have written to use some of these ideas. These procedures are given in the file `MakeMatrix` , available at
http://www.maths.unsw.edu.au/∼jds/Papers/MakeMatrix
and are based around the `LinearAlgebra` package introduced in Maple 7. Similar routines based on the older `linalg` package are also available at
http://www.maths.unsw.edu.au/∼jds/Papers/makematrix

In one case the method is given as a semi-MAPLE algorithm. The routines are very simple, and readers will find it easy to write their own versions in whatever language they are most comfortable.

2 Vectors of integer norm

In $\mathbb{R}^2$, a vector will have an integer norm if and only if its components are the first two numbers in a Pythagorean triple. All these have been known since antiquity, and are given by the following well-known result:

Theorem 1 (Pythagorean triples). *The three numbers x, y, z form a Pythagorean triple if and only if there are three integers u, v and k such that*

$$x = k(u^2 - v^2) \qquad y = 2kuv \qquad z = k(u^2 + v^2)$$

(or the same with x and y exchanged).

Typical examples of such triples are, of course, $(3, 4, 5)$, $(5, 12, 13)$ etc, leading to vectors like $(-3, 4)$, $(12, -5)$ etc.

In $\mathbb{R}^3$ the situation is governed by the theorem, see e.g. [3]:

Theorem 2. *The square of an integer n can be decomposed non-trivially into the sum of two or three squares if and only if n is not a power of 2.*

Typical examples of these that are useful in linear algebra are, for example

$$\begin{aligned}
3^2 &= 1^2 + 2^2 + 2^2 \\
7^2 &= 2^2 + 3^2 + 6^2 \\
9^2 &= 1^2 + 4^2 + 8^2 = 4^2 + 4^2 + 7^2 \\
11^2 &= 6^2 + 6^2 + 7^2 = 2^2 + 6^2 + 9^2 \\
12^2 &= 4^2 + 8^2 + 8^2
\end{aligned}$$

as well as “degenerate” cases arising from Pythagorean triples.

For higher dimensions, we have Lagrange’s theorem (see, e.g. [3])

Theorem 3 (Lagrange’s Theorem). *Any integer can be decomposed into the sum of at most four squares.*

One of the most useful examples of this is that $2^2 = 1^2 + 1^2 + 1^2 + 1^2$.

3 Matrices of a given determinant

One of the standard results of linear algebra is the *PLU* decomposition (see e.g. [4]):

Theorem 4. *Any matrix can be decomposed into the product of a permutation matrix P, a unimodular lower triangular matrix L and an upper triangular matrix U.*

There are two uses for this theorem. Firstly, the choice of L will govern the number (and difficulty) of the row operations for reducing the product $A = LU$ back to echelon form U: choosing any vector $\mathbf{x}$ then gives us the right hand side for a set of linear equations $A\mathbf{x} = \mathbf{b}$. Secondly, we can use this theorem in reverse to create a matrix of a given determinant. All that is required is to take the product of a unimodular lower triangular matrix and an upper triangular matrix with the determinant desired: the MAPLE routine `RandomMatrix` has options that create such matrices. A MAPLE procedure `MakeDetMat` to make the required matrices is given in the file `MakeMatrix` . The columns (or rows) of a matrix of non-zero determinant give bases of the appropriate space of course.

A most useful way of using the *PLU* decomposition is to create matrices of determinant ± 1. These are particularly good for problems involving calculating the inverse of a matrix (there will be no fractions in the solution), and for similarity transformations. The following easy theorem illustrates why this is so:

Theorem 5. *Let A be a square integer matrix. Then A^{-1} is an integer matrix if and only if* $\det A = \pm 1$.

Proof. Since $\det A^{-1} = (\det A)^{-1}$, the only if part is easy. The if part is a consequence of the well-known result that A^{-1} is the (classical) adjoint (the transpose of the matrix of minors, which will be an integer matrix) divided by $\det A$. Or, alternatively, it follows from Cramer’s Rule. □

4 Matrices with a given kernel

There are several ways of doing this. One is to use the normal form [6] directly:

Theorem 6. *Any $n \times m$ matrix A can be written as the product QNP^{-1} where $P \in GL(m, \mathbb{R})$, $Q \in GL(n, \mathbb{R})$ and the normal form N is of the form*

$$\begin{pmatrix} I_r & O \\ O & O \end{pmatrix}$$

with $\operatorname{rank}(A) = r$, *$I_r$ the $r \times r$ identity matrix and O signifying zero matrices.*

In order to find P in this theorem, one finds the kernel of A and extends it to a basis of $\mathbb{R}^m$. This basis then forms the columns of P, with the the last $m - r$ columns of P spanning the kernel of A. For Q one takes the images of the extension vectors and extends them to a basis for $\mathbb{R}^n$.

So, to get a matrix with a given kernel, extend the basis for the kernel by adjoining random vectors as desired. Then premultiplying P^{-1} by a matrix of the form of N in the theorem and of the required size will give the matrix A with given kernel. A premultiplication by any invertible Q can be applied as a disguise.

A second way is to jump the need for extending a basis by using the fact that $(W^{\perp})^{\perp} = W$ for finite dimensions. Take a matrix A whose rows are a basis for the desired kernel. Then a matrix whose row space is $\ker(A)$ has the desired kernel.

The procedure `MakeKernel` in `MakeMatrix` does this and it is designed so that the output will always be an integer matrix with the minimum number of rows: premultiplication by any matrix of full rank and more rows will not alter the kernel.

Alternatively, one can build a suitable matrix directly from row reduction. Suppose W is a linearly independent set of r vectors in $\mathbb{R}^n$, and is to be a basis for the kernel. Then create the matrix whose first r rows are the members of W and whose last row is a general vector $(v_1, v_2, \ldots, v_n)$ in $\mathbb{R}^n$. This is easily done with Matrices and Vectors in MAPLE. Then row reduce the matrix to echelon form. There will be $n - r$ entries in the echelon form that are linear equations in the entries v_i and give conditions for a vector to be in the span of W. The coefficients of these equations are the rows of a $(n - r) \times n$ matrix whose kernel is therefore W. Any matrix whose rows are linear combinations of the rows of this latter matrix will have kernel containing W, so one can multiply on the left by any matrix of full rank to get any size of matrix required.

5 Leontieff Input-Output Matrices

The problem with creating Leontieff matrices is to ensure that we have a consumption (or technology) matrix A, a demand vector $\mathbf{d}$ and an output vector $\mathbf{x}$ all of which have non-negative entries and for which $(I - A)\mathbf{x} = \mathbf{d}$. A consumption matrix for which each entry of $(I - A)^{-1}$ is non-negative is called **productive**, and such a consumption matrix will always do the job.

One simple way of making a productive consumption matrix is to rely on the theorem (e.g. [1], p. 615) that non-negative A is productive if each row (or column) sum of A is less than 1.

More generally, a non-negative matrix A is productive if and only if there is a non-negative $\mathbf{x}$ such that $\mathbf{x} - A\mathbf{x}$ is non-negative ([1] p. 615). So create an initial non-negative matrix A' and non-negative output vector $\mathbf{x}$ and calculate $\mathbf{y} = A'\mathbf{x}$; we need to make the entries of $\mathbf{y}$ less than the corresponding entries of $\mathbf{x}$. One can fiddle a little with the entries of A', if it is small and only one problem is being set to make this the case. Alternatively, scale A' by a suitable factor: if $r \geq \max\{y_i/x_i\}$, let $A = A'/r$. Then the demand vector is set to be $(I - A)\mathbf{x}$, which will have non-negative entries.

6 Orthogonal matrices

By **orthogonal matrices**, I mean (not necessarily square) matrices Q whose columns are an orthonormal set, so that $Q^T Q = I$. Orthogonal 3×3 matrices with determinant 1 are rotations, and are dealt with separately below. More general orthogonal matrices are useful for building sets of vectors for applying the Gram-Schmidt process, or matrices to be factorised as QR, i.e. orthogonal times upper triangular, by the Gram-Schmidt process.

To make a square orthogonal matrix of any size one can rely on the Cayley transform: if A is any anti-symmetric $n \times n$ matrix, then the matrix $(I + A)(I - A)^{-1}$ is square and orthogonal, and can be cut down to size by deleting columns if necessary. The problem with this method is that is is hard to control the numbers that come out.

Another way to get these matrices is to generate a set of mutually orthogonal vectors by hand, and then normalise, giving the orthonormal set that forms a basis: of course, this is what the Gram-Schmidt process is designed to do in the first place. This can easily be done by trial and error (possibly using one or more columns from the Cayley transform) for two vectors. But for more than that trial and error becomes tedious if one tries to make the numbers come out. Instead, one can expedite matters by using the Hodge dual of the outer product in $\mathbb{R}^n$, which gives a generalised cross product. This dual is given by $\Sigma \epsilon^{i \ldots k \ldots l} \mathbf{v}_k \ldots \mathbf{w}_l$, where $\epsilon^{i \ldots l}$ is the alternating symbol — $\epsilon^{i \ldots l}$ is the sign of the permutation $(i \ldots l)$ or zero if $(i \ldots l)$ is not a permutation — and the sum is over the components of the vectors $\mathbf{v}, \ldots, \mathbf{w}$. Each column in the Hodge dual is then orthogonal to each of the vectors $\mathbf{v}$,. . . ,$\mathbf{w}$.

For example, if one begins with the orthogonal vectors $\mathbf{v} = (5, 3, 1, 1)$ and $\mathbf{w} = (1, -3, 5, 1)$ found by trial and error, then my procedure `GenCross` in `MakeMatrix` applied to these vector yields the matrix $\begin{pmatrix} 0 & -6 & 0 & 18 \\ 6 & 0 & -6 & -24 \\ 0 & 6 & 0 & -18 \\ -18 & 24 & 18 & 0 \end{pmatrix}$, and taking the vectors $\mathbf{v}$, $\mathbf{w}$ and the second column of this matrix as our orthogonal set we get the orthogonal matrix $\frac{1}{6} \begin{pmatrix} 5 & 1 & -\sqrt{2} \\ 3 & -3 & 0 \\ 1 & 5 & \sqrt{2} \\ 1 & -1 & 4\sqrt{2} \end{pmatrix}$.

7 Gram-Schmidt and QR decomposition

To get a set of vectors in $\mathbb{R}^n$ suitable for applying Gram-Schmidt without the numbers becoming horrendous, one finds a Q and R for a QR factorisation and uses the columns of the resultant matrix. Start with an orthogonal matrix Q. The entries of the upper triangular matrix R will be chosen to make a matrix of integers at the end. A little modular arithmetic helps everything work out.

Using the matrix Q of the previous section, start with an R of the form $\begin{pmatrix} 6a & x & y \\ 0 & b & z \\ 0 & 0 & c/\sqrt{2} \end{pmatrix}$, so that QR is

$$\begin{pmatrix} 5a & \frac{5}{6}x + \frac{1}{6}b & \frac{5}{6}y + \frac{1}{6}z - \frac{1}{6}c \\ 3a & \frac{1}{2}x - \frac{1}{2}b & \frac{1}{2}y - \frac{1}{2}z \\ a & \frac{1}{6}x + \frac{5}{6}b & \frac{1}{6}y + \frac{5}{6}z + \frac{1}{6}c \\ a & \frac{1}{6}x - \frac{1}{6}b & \frac{1}{6}y - \frac{1}{6}z + \frac{2}{3}c \end{pmatrix}.$$

So, a is an integer. The second column is integer iff $5x + b \equiv x + 5b \equiv x - b \equiv 0 \mod 6$. This reduces to $x \equiv b \mod 6$. The third column is integer if and only if (y, z, c) is a member of the kernel of $\begin{pmatrix} 5 & 1 & -1 \\ 1 & 5 & 1 \\ 1 & -1 & 4 \end{pmatrix}$ modulo 6. As this matrix has determinant $108 \equiv 0$ modulo 6 such vectors exists. Maple's `Nullspace` command can (sometimes) find the kernel over rings like $\mathbb{Z}_6$, and will always do so if the modulus is prime. My procedure `ModKernel` in `MakeMatrix` will find the kernel in some of the cases where `Nullspace` fails. In this example `Nullspace` does fail but `ModKernel` gives as spanning set for the kernel

$$\{\ [1,1,0],\ [3,1,-2],\ [-2,0,2]\ \}.$$

So we could take $a = 1$, $b = c = 4$, $x = -2$, $y = 3$, and $z = 1$ to give the product $\begin{pmatrix} 5 & -1 & 2 \\ 3 & -3 & 1 \\ 1 & 3 & 2 \\ 1 & -1 & 3 \end{pmatrix}$.

8 Least-Squares Problems

The commonest type of least-squares problem is fitting polynomials (most often lines) to n data points (x_i, y_i) in the plane. The normal equations come down to $M\mathbf{x} = \mathbf{z}$, where

$$M = \begin{pmatrix} n & \sum x_i & \sum {x_i}^2 & \cdots \\ \sum x_i & \sum {x_i}^2 & \sum {x_i}^3 & \cdots \\ \sum {x_i}^2 & \sum {x_i}^3 & \sum {x_i}^4 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix}, \qquad \mathbf{z} = \begin{pmatrix} \sum y_i \\ \sum x_i y_i \end{pmatrix}.$$

There is little freedom here: the equations are unpleasant to solve directly except for the 2×2 case. Anything else should be designed to be tackled with a QR factorisation, with A set up to be as easy to factorise as possible. Of course, one is constrained by the fact that in fitting a polynomial of degree n, A will of necessity consist of the first $n+1$ columns of a Vandermonde matrix so there is no freedom of choice except in the second column.

In the case of lines, while with integer points the solution is always rational, it is possible to arrange for the solution to be integer, as follows: Suppose we are looking for the line of best fit to the data points (x_i, y_i), where the x_i are known and the y_i are to be chosen to make the final answer integer. Let $\mathbf{x} \in \mathbb{R}^n$ be the vector $(x_1, \ldots, x_n)$ and A the coefficient matrix of the least squares problem, with first column 1s and second column $\mathbf{x}$. Define $\Delta = \det(A^T A)$ and $M = \Delta(A^T A)^{-1}$ to be the classical adjoint of $A^T A$. Now calculate $\ker M$ in $\mathbb{Z}_\Delta$ and form the matrix B by augmenting A^T with any vector in $\ker_{\mathbb{Z}_\Delta}(M)$. Then for any integer vector $\mathbf{w}$ in $\ker B$, the first n components can be used as the y_i in the set of data points, since then $A^T\mathbf{y}$ is Δ times an integer vector. Of course, the vector $\mathbf{y}$ chosen should not be in $\ker A^T$ or the problem is trivial. The procedure `MakeLeastSquares` in `MakeMatrix` takes the vector $\mathbf{x}$ and returns a matrix M such that any vector in the image of M is a suitable $\mathbf{y}$.

If a more general least squares problem is to be set, the easiest course is to build a matrix A whose QR factorisation is easy and use that.

The interesting general question is: can one start with the normal equations $M\mathbf{x} = \mathbf{z}$ for any symmetric M with full rank (so the solution is unique) and get a matrix A and a vector $\mathbf{y}$ for which these equations are the appropriate normal equations?

Thus, given symmetric invertible M and $\mathbf{z} \in \mathbb{R}^n$, we want to set up a problem $A\mathbf{x} = \mathbf{y}$, so that $M = A^T A$, A is $m \times n$ and $\mathbf{z} = A^T\mathbf{y}$. Clearly $\mathbf{y} \in \mathbb{R}^m$.

Firstly, the rank of A: any vector in $\ker A$ is in $\ker M$ and $\ker M$ is trivial so $\ker A$ is trivial. Therefore A has rank n and $m \geq n$.

The next point to make is that M must positive definite, since

$$\langle M\mathbf{u}, \mathbf{u}\rangle = \langle A^T A\mathbf{u}, \mathbf{u}\rangle = \langle A\mathbf{u}, A\mathbf{u}\rangle \geq 0,$$

and M has no zero eigenvalues.

Finally, if we have such an A, we can solve $A^T\mathbf{y} = \mathbf{z}$ iff $\mathbf{z} \in (\ker A)^\perp$ (see e.g. [5] p. 198). As $\ker A$ is trivial, we can always find a suitable $\mathbf{y}$ given $\mathbf{z}$.

These necessary conditions are in fact sufficient. To prove this, we make use of the Singular Value decomposition, see e.g. [4] section 6.2. The matrix A can be decomposed as $Q\Lambda P^T$, where Q is $m \times m$ orthogonal, P is $n \times n$ orthogonal and Λ is $m \times n$ and of the form $\begin{pmatrix} D \\ O \end{pmatrix}$, where D is $n \times n$ diagonal and O is an appropriately sized zero matrix. It follows that $M = PD^2P^T$, and so the diagonal entries of D are square roots of the eigenvalues of M (giving a second proof of the necessity of positive definiteness). Note that if Λ^+ is the matrix $\begin{pmatrix} D^{-1} \\ O \end{pmatrix}$, so that $A^+ = P\Lambda^+Q^T$ is the Moore-Penrose generalised inverse, the solution to the least squares problem is $\mathbf{x} = A^+\mathbf{y}$ [4].

Thus, given *positive definite* M, we can find D and P, and then we need only extend D to Λ and rig a suitable Q to get A. Whether this can be done so that A is integer is another matter, as is thinking of a problem for which A arises as required.

9 Householder matrices and QR by Householder

By the **Householder matrix** for a vector $\mathbf{v} \in \mathbb{R}^n$ of norm M, I mean the matrix of the reflection that swaps $\mathbf{v}$ with $\mathbf{w} = \pm M\mathbf{e}_1$, where $\mathbf{e}_1$ is the first member of the standard basis for $\mathbb{R}^n$ and the sign is chosen to be the *opposite* to that of $\mathbf{v} \cdot \mathbf{e}_1$, see [4]. Since this matrix is given by $I - 2\mathbf{u}\mathbf{u}^T$, where $\mathbf{u}$ is the unit vector in the direction of $\mathbf{v} - \mathbf{w}$, the vector $\mathbf{v}$ must be of integer norm if the calculation is to be possible without introducing nested surds.

Creating a matrix that can be QR factorised by hand with Householder matrices (see [4]) is best done backwards. Since Q is the product of Householder matrices, it is square and R is the same dimension as the matrix to be factorised. I illustrate one way of doing this with a 4 by 3 matrix.

The last matrix to be transformed, A_3, will be 2×1, and should be chosen to be a vector of integer norm, for example $(4, 3)$. The second last matrix to be transformed, A_2, will be 3×2, and after multiplication by a Householder matrix will look like $R_2 = \begin{pmatrix} a & b \\ 0 & 4 \\ 0 & 3 \end{pmatrix}$ for our choice of A_3. It follows that the first column of A_2 will be a vector of norm a, so choose a and the column $\mathbf{c}_2$ appropriately. Then A_2 is the matrix Q_2R_2, where Q_2 is the Householder matrix of c_2. The constant b is chosen to make A_2 as simple as possible (integer if possible). For example, with $a = 3$, $\mathbf{c}_2 = \begin{pmatrix} -1 \\ 2 \\ -2 \end{pmatrix}$ and $b = -1$, A_2 is $\begin{pmatrix} -1 & 1 \\ 2 & 3 \\ -2 & 4 \end{pmatrix}$.

Finally, the matrix A will after the first transformation look like $R_1 = \begin{pmatrix} c & d & e \\ 0 & -1 & 1 \\ 0 & 2 & 3 \\ 0 & -2 & 4 \end{pmatrix}$, so as before choose c and appropriate first column $\mathbf{c}_1$ of A of norm c, then $A = Q_1R_1$ where Q_1 is the Householder matrix of $\mathbf{c}_1$, with d and e chosen to give as simple a matrix as possible. For example, with $c = 7$, $\mathbf{c}_1 = (-2, 0, 6, 3)^T$, $d = 3$ and $b = 1$ we get

$$A = \begin{pmatrix} -2 & 0 & 4 \\ 0 & -1 & 1 \\ 6 & 4 & 1 \\ 3 & -1 & 3 \end{pmatrix} = \frac{1}{105} \begin{pmatrix} -30 & 30 & -96 & 3 \\ 0 & -35 & -14 & -98 \\ 90 & 50 & -13 & -16 \\ 45 & -80 & -38 & 34 \end{pmatrix} \begin{pmatrix} 7 & 3 & 1 \\ 0 & 3 & -1 \\ 0 & 0 & -5 \\ 0 & 0 & 0 \end{pmatrix}$$

Note that the product of the norms of the columns will occur as the factor before the matrix Q.

10 Matrices with a given Jordan form

Firstly, to create a diagonalisable matrix, the simplest way is to create a matrix P with determinant ± 1 and then perform a similarity transformation PDP^{-1}, where D is a diagonal matrix with the desired eigenvalues. The columns of P then form the eigenvectors.

A simple extension of this idea gives non-diagonalisable matrices with a given Jordan canonical form [6]. Rather than performing a similarity transformation on a diagonal matrix, perform it on the appropriate Jordan form. Alternatively, use the matrix formed from blocks given by companion matrices, since the companion matrix of the polynomial $(t-\lambda)^n$ has Jordan form consisting of a single Jordan block. The routine `MakeJordan` in `MakeMatrix` takes the first of these approaches. It can also be used with a given transition matrix.

11 Orthogonal matrices in $\mathbb{R}^3$

I have three ways of calculating these matrices, which geometrically are rotations in $\mathbb{R}^3$. If one wishes to control the axis of rotation and the angle, use the following parameterisation of $SO(3)$:

Theorem 7. *If $\mathbf{u} = (u_1, u_2, u_3)^T$ is a unit vector in $\mathbb{R}^3$ and $U = \begin{pmatrix} 0 & -u_3 & u_2 \\ u_3 & 0 & -u_1 \\ -u_2 & u_1 & 0 \end{pmatrix}$ then the matrix*

$$Q = I_3 \cos\theta + \mathbf{u}\mathbf{u}^T(1-\cos\theta) + U\sin\theta$$

is the matrix of the rotation about the direction of $\mathbf{u}$ through angle θ.

Proof. That $Q^TQ = I_3$ can be checked with a direct calculation. Since $\det Q$ is a continuous function from $\mathbb{R}$ to $\{-1, 1\}$ that clearly has value 1 at 0, it has value 1 for all θ, and so $Q \in SO(3)$. Then since $U\mathbf{u} = \mathbf{0}$, and $\mathbf{u}^T\mathbf{u} = 1$, $Q\mathbf{u} = \mathbf{u}$, so $\mathbf{u}$ is the direction of the axis. Finally $\operatorname{Tr} Q = 1 + 2\cos\theta$, so that θ is the angle of rotation. □

The MAPLE procedure `MakeRot` in the file `MakeMatrix` can use this method.

In order for the final matrix in the first method to have no (or few) surds, it is preferable to match up any surds in the direction and the angle of rotation. For example, with $\mathbf{u} = (1/\sqrt{3}, 1/\sqrt{3}, 1/\sqrt{3})$ the rotation about $\mathbf{u}$ through angle $\pi/3$ is $\dfrac{1}{3}\begin{pmatrix} 2 & 2 & -1 \\ -1 & 2 & 2 \\ 2 & -1 & 2 \end{pmatrix}$.

The other two methods of creating a rational rotation matrix can be used if one is not concerned with the axis and angle. One is to use the Cayley transform, as mentioned in section 6. The other is to take, say, two orthogonal vectors of unit norm as the first two columns and their cross product as the third. For example $\frac{1}{7}(2,3,6)^T$ and $\frac{1}{7}(3,-6,2)^T$ are orthogonal, and their cross product is $\frac{1}{7}(6,2,-3)^T$.

The question then arises as to when this latter method can be used: in other words, given a vector $\mathbf{v} = (a, b, c)$ in $\mathbb{Z}^3$, is there a second vector whose entries are some permutation of a, b

and c — with sign changes — that is orthogonal to $\mathbf{v}$? There are three possibilities, excluding re-arrangements and changes of overall sign, depending on how many of the components change position (I am grateful to Peter Brown [2] for the following analysis):

1. If $a^2+b^2=c^2$, then $(a,b,-c)$ is orthogonal to $\mathbf{v}$. For example $(3,4,5)$ and $(3,4,-5)$.

2. If $c^2=2ab$ then $(b,a,-c)$ is orthogonal to $\mathbf{v}$. For example $(1,2,2)$ and $(2,1,-2)$.

3. If $a^{-1}=b^{-1}+c^{-1}$ then $(c,a,-b)$ is orthogonal to $\mathbf{v}$. For example $(2,3,6)$ and $(6,2,-3)$.

The Diophantine equation in (3) has been studied since antiquity. Given any a it is possible to find a b and c to satisfy the equation: factorise $a^2=pq$ where $p \geq q$, and then $b=(q+a)$ and $c=(p+a)$ solve the equation.

The first of these cases will leave a $\sqrt{2}$ in the matrix, and is therefore not suitable. The other two methods both lead to vectors of rational norm, and thence to rational orthogonal matrices. In case 2, the vectors (a,b,c) etc. will all have norm $a+b$, and in case 3 the vectors will have norm $|c-bc^2-bc+b^2|$, as can easily be checked. In both cases we get the nice form “rational times integer matrix”.

Similar methods could be used in higher dimensions. For example, in $\mathbb{R}^4$, the vector (a,b,c,d) is orthogonal to $(a,b,c,-d)$ if $a^2+b^2+c^2=d^2$ etc.

12 Symmetric Matrices with given eigenvalues

What we wish to do here is ensure that the matrix we end up with has small integer entries. At first sight this is relatively simple: take a rational orthogonal matrix and use it in a similarity transformation of a diagonal matrix with the desired eigenvalues. Then multiply the final matrix by the appropriate integer to clear all fractions. Unfortunately, I have found this simple idea of somewhat limited usefulness. The eigenvalues end up being n^2 times the ones you began with (assuming they were integers to start with). In order to try to “clear up” some (at least) of the fractions, it is necessary to be careful in the choice of eigenvalues.

The 2×2 case is very easy: if one eigenvector is $\Delta^{-1}\begin{pmatrix}a\\b\end{pmatrix}$, $\Delta=1/(a^2+b^2)$, then the other is $\Delta^{-1}\begin{pmatrix}-b\\a\end{pmatrix}$. We take the eigenvalues to be 0 and $k\Delta$ (which can be shifted by adding λI), and then the matrix is

$$\begin{pmatrix}b^2k & bka\\ bka & a^2k\end{pmatrix}.$$

For the 3×3 case, proceed as follows:

Let $Q=\frac{1}{n}P$, where P is an integer matrix, be orthogonal. Define $A=Q\Lambda Q^T$, where Λ is the diagonal matrix with entries λ_i and define $M=n^2A$. We want to solve a linear system in integers, so use the MAPLE command

```
    syss:=seq(M[1,i]-n^2*d[i]=0,i=1..3),seq(M[2,j]-n^2*d[2+j]=0,j=2..3),
M[3,3]-n^2*d[6]=0;
```

where the `d[i]` are dummy variables, to set up a system of 6 equations in 9 unknowns. The command

```
G:=GenerateMatrix([syss],[seq(lambda[i],i=1..3),seq(d[k],k=1..6)])[1];
```

extracts the matrix of this system, and we want the kernel of G: set `kk:=NullSpace(G);`

Then if one takes an arbitrary linear combination of the vectors in this kernel, the first three components will give you three eigenvalues that will ensure that A is an integer matrix.

For example, with the matrix of a rotation through $\pi/3$ around $(1,1,1)$ (see previous section), the kernel of the matrix G is found to be spanned by

$$\{(3,-3,0,0,2,-2,-1,0,1),(-5/2,2,2,0,-1,2,3/2,1,0),(-2,4,1,1,-2,2,2,0,0)\}$$

and taking $x\mathbf{v}_1+2y\mathbf{v}_2+z\mathbf{v}_3$, say, the eigenvalues are $3x-5y-2z$, $-3x+4y+4z$ and $4y+z$ for any choice of x, y and z, so for example, with eigenvalues $a=-5$, $b=-2$ and $c=1$, A is an integer matrix.

Naturally, all the eigenvalues can be shifted by λ by the addition of λI. And of course this all works for higher dimensions, suitably modified. The procedure `MakeSymmetric` in `MakeMatrix` will provide the matrix A with given eigenvectors.

It is also possible to cook up symmetric matrices more directly, with less control over the eigenvalues and eigenvectors. For example the matrix

$$\begin{pmatrix} 1 & b & c \\ b & b^2 & bc \\ c & bc & c^2 \end{pmatrix}$$

has eigenvalues 0, 0 and $1+b^2+c^2$.

13 Matrices with easily calculated exponential

The $n\times n$ matrices A whose exponential is easiest to calculate are either those with only one eigenvalue, diagonalisable matrices whose eigenvalues are ±1, or matrices with eigenvalues 0 and any one other non-zero, non-deficient eigenvalue μ.

In the first case, since $(A-\lambda I)^m=O$ for some $m\leq n$, the exponential can be expressed as $e^{\lambda t}$ times a polynomial in $t(A-\lambda I)$ of degree m.

In the second case, $A^2=I$, since the minimal polynomial is λ^2-1, and one can calculate the exponential directly from the series definition.

In the third case, $A^{m+1}=\mu A^m$, where m is the size of the largest Jordan block for eigenvalue 0, since the minimal polynomial is $\lambda^m(\lambda-\mu)$, and again one can calculate the exponential directly from the series.

All these types can easily be created using the above methods.

Failing that, ensuring that there is a basis of generalised eigenvectors that is as easy to invert as possible will simplify any calculations.

14 Systems of linear differential equations with constant coefficients

The basic philosophy here is to ensure that the vectors that occur are (generalised) eigenvectors. So to solve, for example, $\mathbf{y}'(t) = A\mathbf{y}(t) + \mathbf{b}(t)$, by the formula $\mathbf{y} = \exp(tA)\mathbf{z}(t)$ with $\mathbf{z}' = \exp(-tA)\mathbf{b}$, one would want $\mathbf{b}(t)$ to be a (generalised) eigenvector, so that $\exp(-tA)\mathbf{b}$ is an easy calculation with the following theorem, which follows from basic properties of the exponential

Theorem 8. *Let A be an $n \times n$ matrix; let λ be an eigenvalue of A and $\mathbf{v} \in \ker(A - \lambda I)^{k+1}$ be a generalised eigenvector for eigenvalue λ. Then*

$$e^{tA}\mathbf{v} = e^{\lambda t}\left(\mathbf{v} + t(A - \lambda I)\mathbf{v} + \frac{1}{2!}t^2(A - \lambda I)^2\mathbf{v} + \cdots + \frac{1}{k!}t^k(A - \lambda I)^k\mathbf{v}\right) \ .$$

The point here is that one calculates a finite sum, all terms of which are of the form matrix×vector. One would also need to rig the initial condition so that the arbitrary constant from the integration is a (generalised) eigenvector. If $\mathbf{b}$ is of the form $e^{\lambda t}\mathbf{v}$ where $\mathbf{v}$ is a (generalised) eigenvector for eigenvalue λ, this latter part is easy, as the constant vector *is* the initial condition, otherwise one needs to be more careful.

To make these problems slightly more involved, rather than use (generalised) eigenvectors one can use a simple combination of two or more (generalised) eigenvectors.

15 Acknowledgements

I wish to record my thanks to Peter Brown, Peter Donovan, Dennis Trenerry and Rod James of the University of New South Wales for several useful discussions over the past several years.

MAPLE is a registered trademark of Waterloo Maple Inc.

References

[1] Howard Anton and Chris Rorres, *Elementary Linear Algebra Applications Version*, Eighth edition, Wiley (2000)

[2] Peter Brown, Private communication

[3] H. Davenport *The Higher Arithmetic*, fourth edition Hutchinson University Library (1970)

[4] Richard O. Hill, Jr *Elementary Linear Algebra with Applications*, second edition, Harcourt Brace Jovanovich (1991)

[5] Terry Lawson, *Linear Algebra*, Wiley (1996)

[6] Michael O'Nan, *Linear Algebra*, Harcourt Brace Jovanovich (1976)