An Algorithm for Multiplication in Modular Arithmetic

Walter Gross
24 Eldridge Avenue
Johnson City, NY 13790

Peter Hilton
SUNY Binghamton
Binghamton, NY 13901

Jean Pedersen
University of Santa Clara
Santa Clara, CA 95053

Kim Yew Yap
National University of Singapore
Kent Ridge, Singapore

Introduction

Modular arithmetic provides a good introduction to genuinely mathematical ideas in arithmetic and to the basic concepts of abstract algebra. In this note we offer an algorithm for multiplying in modular arithmetic. To be precise, let n and l be positive integers with $l < n$ and l prime to n. Then our algorithm will allow us to multiply by l modulo n without actually carrying out the multiplications in ordinary whole number arithmetic.

We will also discuss a refinement of the algorithm which enables us to perform multiplications modulo n by all such numbers l in a simple natural sequence.

The algorithm

Our algorithm naturally divides itself into two parts; we call these Algorithm A and Algorithm B. It is Algorithm B for which we claim originality, but we first describe Algorithm A.

Algorithm A. Find k, $1 \leqslant k < n$, such that $kl \equiv -1 (\mathrm{mod}\ n)$ (note that such an integer k is unique). We may, for instance, proceed by using the Euclidean algorithm to find integers u, v, with $|u| < n$, $|v| < l$, and $ul + vn = 1$. Then

$$k = -u \quad \text{if } u \text{ is negative,}$$
$$k = n - u \quad \text{if } u \text{ is positive.}$$

Algorithm B. Write out the residues modulo n, starting with 1 and in their natural order in a rectangular $(k \times l)$ array, which we call NA (natural array). That is, the residues increase from left to right, row by row. Then apply to this array the transformation T which consists of taking the *columns* of NA in *reverse* order and writing the entries again in a $(k \times l)$-array, row by row. Call the new array TNA. In a $(k \times l)$-array, we refer to the entry in the ith row, reading downwards, and the jth column, reading across, as the (i, j)-entry. Then *the (i, j)-entry of TNA is obtained from the (i, j)-entry of NA by multiplying by l modulo n.* Before proving this we give an example.

Example 1. Let $n = 17$, $l = 3$. Algorithm A produces $k = 11$. Our (11×3)-array, modulo 17, reads

Mathematical Association of America
is collaborating with JSTOR to digitize, preserve, and extend access to
Mathematics Magazine
www.jstor.org

$$NA = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 10 & 11 & 12 \\ 13 & 14 & 15 \\ 16 & 0 & 1 \\ 2 & 3 & 4 \\ 5 & 6 & 7 \\ 8 & 9 & 10 \\ 11 & 12 & 13 \\ 14 & 15 & 16 \end{pmatrix}; \quad \text{and} \quad TNA = \begin{pmatrix} 3 & 6 & 9 \\ 12 & 15 & 1 \\ 4 & 7 & 10 \\ 13 & 16 & 2 \\ 5 & 8 & 11 \\ 14 & 0 & 3 \\ 6 & 9 & 12 \\ 15 & 1 & 4 \\ 7 & 10 & 13 \\ 16 & 2 & 5 \\ 8 & 11 & 14 \end{pmatrix}.$$

Then each entry in TNA is, modulo 17, 3 times the corresponding entry in NA. For example, the $(5,2)$-entry of NA is 14, the $(5,2)$-entry of TNA is 8, and $8 \equiv 3 \cdot 14 \pmod{17}$. Of course, the nonzero residues modulo 17 occur *twice* in NA because $11 \times 3 = 2 \times 17 - 1$.

We now justify Algorithm B. However it will be convenient for the sequel to prove a slightly generalized version. In this version we choose any positive integer b and we suppose that we have already written the residues modulo n, starting with b and then advancing by b modulo n, in their natural order in a rectangular $(k \times l)$-array, which we call $NA(b)$, so that $NA = NA(1)$. We then claim

$$TNA(b) = NA(lb). \tag{1}$$

Proof of (1). The $(1, l)$-entry in $NA(b)$ is lb, modulo n, and the $(r+1, s)$-entry is obtained from the (r, s)-entry by adding lb modulo n. It follows that $TNA(b)$ is the array starting with lb, modulo n, and proceeding by adding lb modulo n, that is, it is the array $NA(lb)$, except perhaps at the 'seams'. Precisely, it remains to prove that, in passing from the foot of one column in $NA(b)$ to the head of the preceding column, we advance by lb modulo n.

Now since the entry at the foot of the lth column of $NA(b)$ is congruent to $-b$ modulo n, it follows that the entry at the foot of the sth column is congruent to $-b-(l-s)b$ modulo n. If $s \geqslant 2$, the entry at the head of the $(s-1)$st column is congruent to $(s-1)b$. But

$$-b-(l-s)b+lb=(s-1)b,$$

so the proof of (1) is complete.

Of course Algorithm B is justified by taking $b = 1$ in (1). Notice that the algorithm is especially nice if $l \mid n-1$. For then we simply find k by division and the array NA is of 'minimal area' $n-1$, with no repetition; we simply write out the nonzero residues mod n in a rectangular $(k \times l)$-array, with $kl = n-1$.

From (1) we immediately obtain by iteration

$$T^q NA = NA(l^q). \tag{2}$$

Thus we obtain the rule for multiplying by l^q modulo n, for any $q \geqslant 1$. This is especially satisfactory if l is a primitive residue (or root) modulo n, that is, if the powers of l run through *all* residues prime to n. *Such primitive residues exist if and only if* $n = 2, 4, p^m, 2p^m$, *where* p *is an odd prime and* $m \geqslant 1$ [**1**]. Thus in these cases, by judicious choice of l, we may quickly determine how to multiply modulo n by any number prime to n. In particular, if n is itself prime then, by suitably choosing l, we obtain the entire multiplication table of arithmetic modulo n. Let us give an example.

EXAMPLE 2. Let $n = 13$. It is then easy to see that 2 is primitive modulo 13; this amounts to showing that the smallest power of 2 which is congruent to 1 is 2^{12}. But this must be so since $2^6 = 64 \not\equiv 1$ and $2^4 = 16 \not\equiv 1$. In fact the powers of 2 run successively through the residues $2, 4, 8, 3, 6, 12, 11, 9, 5, 10, 7, 1$. Thus if we start with

$$NA = \begin{Bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \\ 7 & 8 \\ 9 & 10 \\ 11 & 12 \end{Bmatrix}$$

and successively carry out the transformations $T, T^2, T^3, \ldots, T^{11}$, we obtain the rules for multiplying by $2,4,8,3,6,12,11,9,5,10,7$, respectively, that is, the entire multiplication table. This example exhibits the favorable feature $l|n-1$ already referred to.

Some further points can be made relating to this example, but having a wider application.

First, it is not necessary to compute the powers of 2 modulo 13 (or, more generally, of l modulo n) to know what we are multiplying by—*one simply looks at the leading entry in any* T^qNA, *and that entry indicates the multiplier.*

Second, suppose that $n = 4$, p^m, or $2p^m$, and that l is a primitive residue modulo n. Then, since the multiplicative group of residues prime to n is a cyclic group of even order $\varphi(n)$ generated by l, and since such a group has exactly one element of order 2, it follows that

$$l^{\frac{1}{2}\varphi(n)} \equiv -1 \pmod{n}, \tag{3}$$

where φ is Euler's totient function. Formula (3) means that, if T is executed $\frac{1}{2}\varphi(n)$ times, the result is our original NA multiplied by -1 modulo n. But this is equivalent to *reversing* NA, that is, writing the entries of NA in reverse order. One may verify that, with $n = 13$, $l = 2$, then $\varphi(n) = 12$ and

$$T^6NA = \begin{Bmatrix} 12 & 11 \\ 10 & 9 \\ 8 & 7 \\ 6 & 5 \\ 4 & 3 \\ 2 & 1 \end{Bmatrix}.$$

Another good example is furnished by $n = 18$, (so that $\varphi(n) = 6$), $l = 5$, $k = 7$. Then

$$NA = \begin{Bmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \\ 11 & 12 & 13 & 14 & 15 \\ 16 & 17 & 0 & 1 & 2 \\ 3 & 4 & 5 & 6 & 7 \\ 8 & 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 & 17 \end{Bmatrix}, \qquad T^3NA = \begin{Bmatrix} 17 & 16 & 15 & 14 & 13 \\ 12 & 11 & 10 & 9 & 8 \\ 7 & 6 & 5 & 4 & 3 \\ 2 & 1 & 0 & 17 & 16 \\ 15 & 14 & 13 & 12 & 11 \\ 10 & 9 & 8 & 7 & 6 \\ 5 & 4 & 3 & 2 & 1 \end{Bmatrix}.$$

Third, since the transformation T enables us (in the general case) to multiply by l, it follows that the inverse transformation T^{-1} enables us to divide by l. Notice that division by l is defined since l is prime to the modulus n.

In a sense, *division by* l *is easier than multiplication by* l, since we may write out the array $T^{-1}NA$ without having to write out NA! For example, with $n = 7$ and $l = 2$ (so that $k = 3$), we simply write the numbers from 1 to 6 in two columns, starting on the right:

$$T^{-1}NA = \begin{Bmatrix} 4 & 1 \\ 5 & 2 \\ 6 & 3 \end{Bmatrix}.$$

Then we know that we get

$$NA = \begin{Bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{Bmatrix}$$

by doubling each entry in $T^{-1}NA$. We need to write out NA in order to know which residue has been halved to obtain the corresponding residue in $T^{-1}NA$, but not in order to be able to *produce* $T^{-1}NA$.

References

[1] Ivan Niven and H. S. Zuckerman, An Introduction to the Theory of Numbers, Fourth Edition, John Wiley and Sons, 1980, p. 62.

The Largest Unit Ball in Any Euclidean Space

Jeffrey Nunemacher
Oberlin College
Oberlin, OH 44074

In what dimensional Euclidean space does the unit ball have greatest volume? greatest surface area? The usual approach to this problem is to find explicit formulas for the volume and surface area and then to analyze their behavior. The standard derivations of these formulas are based on recurrence relations and typically involve some advanced calculus. For various approaches see, for example, [**1**, p. 411]; [**2**, p. 302]; [**3**, p. 220]; [**4**, p. 502]; [**5**, p. 324]. This note solves the problem by working directly from the recurrence relations. This approach is pleasingly simple and makes the argument accessible to a multivariable calculus class.

Let $B_n(r)$ denote the open ball of radius r in R^n, i.e.,

$$B_n(r) = \left\{ (x_1, x_2, \ldots, x_n) \,\middle|\, \sum_{i=1}^{n} x_i^2 < r^2 \right\},$$

and let $V_n(r)$ denote its volume. Since balls are similar n-dimensional objects, it is not surprising that there are constants a_n so that $V_n(r) = a_n r^n$. This statement can be proved using the change of variables formula (see, e.g., [**4**, p. 500]). A more elementary argument can be carried out based on approximation by Riemann sums, using the basic observation that if all sides of an n-box are magnified by r, then the volume is magnified by r^n.

By definition we have

$$V_n(r) = \iint\limits_{B^n(r)} \cdots \int 1 \, dx_1 \, dx_2 \cdots dx_n = \int_{-r}^{r} \left(\iint\limits_{\sum_{i=1}^{n-1} x_i^2 < r^2 - x_n^2} \cdots \int 1 \, dx_1 \, dx_2 \cdots dx_{n-1} \right) dx_n.$$

Since the value of the inner integral is $V_{n-1}\left(\sqrt{r^2 - x_n^2}\right)$, we find that

$$V_n(r) = \int_{-r}^{r} a_{n-1} \left(\sqrt{r^2 - x_n^2}\right)^{n-1} dx_n.$$