In fact, the matrix will have an inverse if m is the degree of the recurrence used to generate the keystream (see the Exercises for a proof).

Let's illustrate with an example.

Example 1.14 Suppose Oscar obtains the ciphertext string

$$101101011110010$$

corresponding to the plaintext string

$$011001111111000.$$

Then he can compute the keystream bits:

$$110100100001010.$$

Suppose also that Oscar knows that the keystream was generated using a 5-stage LFSR. Then he would solve the following matrix equation, which is obtained from the first 10 keystream bits:

$$(0,1,0,0,0) = (c_0, c_1, c_2, c_3, c_4) \begin{pmatrix} 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}.$$

It can be verified that

$$\begin{pmatrix} 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}^{-1} = \begin{pmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 \end{pmatrix},$$

by checking that the product of the two matrices, computed modulo 2, is the identity matrix. This yields

$$\begin{aligned}(c_0, c_1, c_2, c_3, c_4) &= (0,1,0,0,0) \begin{pmatrix} 0\,1\,0\,0\,1 \\ 1\,0\,0\,1\,0 \\ 0\,0\,0\,0\,1 \\ 0\,1\,0\,1\,1 \\ 1\,0\,1\,1\,0 \end{pmatrix} \\ &= (1,0,0,1,0).\end{aligned}$$

Thus the recurrence used to generate the keystream is

$$z_{i+5} = (z_i + z_{i+3}) \bmod 2.$$

□

1.3 Notes

Material on classical cryptography is covered in various textbooks and monographs, such as "Decrypted Secrets, Methods and Maxims of Cryptology" by Bauer [9]; "Cipher Systems, The Protection of Communications" by Beker and Piper [13]; "Cryptology" by Beutelspacher [32]; "Cryptography and Data Security" by Denning [109]; "Code Breaking, A History and Exploration" by Kippenhahn [192]; "Cryptography, A Primer" by Konheim [203]; and "Basic Methods of Cryptography" by van der Lubbe [222].

We have used the statistical data on frequency of English letters that is reported in Beker and Piper [13].

A good reference for elementary number theory is "Elementary Number Theory and its Applications" by Rosen [284]. Background in linear algebra can be found in "Elementary Linear Algebra" by Anton [4].

Two very enjoyable and readable books that provide interesting histories of cryptography are "The Codebreakers" by Kahn [185] and "The Code Book" by Singh [307].

Exercises

1.1 Evaluate the following:
  (a) 7503 mod 81
  (b) (−7503) mod 81
  (c) 81 mod 7503
  (d) (−81) mod 7503.

1.2 Suppose that $a, m > 0$, and $a \not\equiv 0 \pmod{m}$. Prove that

$$(-a) \bmod m = m - (a \bmod m).$$

1.3 Prove that $a \bmod m = b \bmod m$ if and only if $a \equiv b \pmod{m}$.

1.4 Prove that $a \bmod m = a - \lfloor \frac{a}{m} \rfloor m$, where $\lfloor x \rfloor = \max\{y \in \mathbb{Z} : y \leq x\}$.

1.5 Use exhaustive key search to decrypt the following ciphertext, which was encrypted using a *Shift Cipher*:

```
BEEAKFYDJXUQYHYJIQRYHTYJIQFBQDUYJIIKFUHCQD.
```

1.6 If an encryption function e_K is identical to the decryption function d_K, then the key K is said to be an *involutory key*. Find all the involutory keys in the *Shift Cipher* over $\mathbb{Z}_{26}$.

1.7 Determine the number of keys in an *Affine Cipher* over $\mathbb{Z}_m$ for $m = 30, 100$ and 1225.

1.8 List all the invertible elements in $\mathbb{Z}_m$ for $m = 28, 33$ and 35.

1.9 For $1 \leq a \leq 28$, determine $a^{-1} \bmod 29$ by trial and error.

1.10 Suppose that $K = (5, 21)$ is a key in an *Affine Cipher* over $\mathbb{Z}_{29}$.
  (a) Express the decryption function $d_K(y)$ in the form $d_K(y) = a'y + b'$, where $a', b' \in \mathbb{Z}_{29}$.

(b) Prove that $d_K(e_K(x)) = x$ for all $x \in \mathbb{Z}_{29}$.

1.11 (a) Suppose that $K = (a, b)$ is a key in an *Affine Cipher* over $\mathbb{Z}_n$. Prove that K is an involutory key if and only if $a^{-1} \bmod n = a$ and $b(a+1) \equiv 0 \pmod{n}$.

(b) Determine all the involutory keys in the *Affine Cipher* over $\mathbb{Z}_{15}$.

(c) Suppose that $n = pq$, where p and q are distinct odd primes. Prove that the number of involutory keys in the *Affine Cipher* over $\mathbb{Z}_n$ is $n + p + q + 1$.

1.12 (a) Let p be prime. Prove that the number of 2×2 matrices that are invertible over $\mathbb{Z}_p$ is $(p^2 - 1)(p^2 - p)$.

HINT Since p is prime, $\mathbb{Z}_p$ is a field. Use the fact that a matrix over a field is invertible if and only if its rows are linearly independent vectors (i.e., there does not exist a non-zero linear combination of the rows whose sum is the vector of all 0's).

(b) For p prime and $m \geq 2$ an integer, find a formula for the number of $m \times m$ matrices that are invertible over $\mathbb{Z}_p$.

1.13 For $n = 6, 9$ and 26, how many 2×2 matrices are there that are invertible over $\mathbb{Z}_n$?

1.14 (a) Prove that $\det A \equiv \pm 1 \pmod{26}$ if A is a matrix over $\mathbb{Z}_{26}$ such that $A = A^{-1}$.

(b) Use the formula given in Corollary 1.4 to determine the number of involutory keys in the *Hill Cipher* (over $\mathbb{Z}_{26}$) in the case $m = 2$.

1.15 Determine the inverses of the following matrices over $\mathbb{Z}_{26}$:

(a) $\begin{pmatrix} 2 & 5 \\ 9 & 5 \end{pmatrix}$

(b) $\begin{pmatrix} 1 & 11 & 12 \\ 4 & 23 & 2 \\ 17 & 15 & 9 \end{pmatrix}$

1.16 (a) Suppose that π is the following permutation of $\{1, \ldots, 8\}$:

x	1	2	3	4	5	6	7	8
$\pi(x)$	4	1	6	2	7	3	8	5

Compute the permutation π^{-1}.

(b) Decrypt the following ciphertext, for a *Permutation Cipher* with $m = 8$, which was encrypted using the key π:

```
TGEEMNELNNTDROEOAAHDOETCSHAEIRLM.
```

1.17 (a) Prove that a permutation π in the *Permutation Cipher* is an involutory key if and only if $\pi(i) = j$ implies $\pi(j) = i$, for all $i, j \in \{1, \ldots, m\}$.

(b) Determine the number of involutory keys in the *Permutation Cipher* for $m = 2, 3, 4, 5$ and 6.

1.18 Consider the following linear recurrence over $\mathbb{Z}_2$ of degree four:

$$z_{i+4} = (z_i + z_{i+1} + z_{i+2} + z_{i+3}) \bmod 2,$$

$i \geq 0$. For each of the 16 possible initialization vectors $(z_0, z_1, z_2, z_3) \in (\mathbb{Z}_2)^4$, determine the period of the resulting keystream.

1.19 Redo the preceding question, using the recurrence

$$z_{i+4} = (z_i + z_{i+3}) \bmod 2,$$

$i \geq 0$.

1.20 Suppose we construct a keystream in a synchronous stream cipher using the following method. Let $K \in \mathcal{K}$ be the key, let $\mathcal{L}$ be the keystream alphabet, and let Σ be a finite set of *states*. First, an *initial state* $\sigma_0 \in \Sigma$ is determined from K by some method. For all $i \geq 1$, the state σ_i is computed from the previous state σ_{i-1} according to the following rule:

$$\sigma_i = f(\sigma_{i-1}, K),$$

where $f : \Sigma \times \mathcal{K} \to \Sigma$. Also, for all $i \geq 1$, the keystream element z_i is computed using the following rule:

$$z_i = g(\sigma_i, K),$$

where $g : \Sigma \times \mathcal{K} \to \mathcal{L}$. Prove that any keystream produced by this method has period at most $|\Sigma|$.

1.21 Below are given four examples of ciphertext, one obtained from a *Substitution Cipher*, one from a *Vigenère Cipher*, one from an *Affine Cipher*, and one unspecified. In each case, the task is to determine the plaintext.

Give a clearly written description of the steps you followed to decrypt each ciphertext. This should include all statistical analysis and computations you performed.

The first two plaintexts were taken from "The Diary of Samuel Marchbanks," by Robertson Davies, Clarke Irwin, 1947; the fourth was taken from "Lake Wobegon Days," by Garrison Keillor, Viking Penguin, Inc., 1985.

(a) *Substitution Cipher*:

```
EMGLOSUDCGDNCUSWYSFHNSFCYKDPUMLWGYICOXYSIPJCK
QPKUGKMGOLICGINCGACKSNISACYKZSCKXECJCKSHYSXCG
OIDPKZCNKSHICGIWYGKKGKGOLDSILKGOIUSIGLEDSPWZU
GFZCCNDGYYSFUSZCNXEOJNCGYEOWEUPXEZGACGNFGLKNS
ACIGOIYCKXCJUCIUZCFZCCNDGYYSFEUEKUZCSOCFZCCNC
IACZEJNCSHFZEJZEGMXCYHCJUMGKUCY
```

HINT F decrypts to w.

(b) *Vigenère Cipher*:

```
KCCPKBGUFDPHQTYAVINRRTMVGRKDNBVFDETDGILTXRGUD
DKOTFMBPVGEGLTGCKQRACQCWDNAWCRXIZAKFTLEWRPTYC
QKYVXCHKFTPONCQQRHJVAJUWETMCMSPKQDYHJVDAHCTRL
SVSKCGCZQQDZXGSFRLSWCWSJTBHAFSIASPRJAHKJRJUMV
GKMITZHFPDISPZLVLGWTFPLKKEBDPGCEBSHCTJRWXBAFS
PEZQNRWXCVYCGAONWDDKACKAWBBIKFTIOVKCGGHJVLNHI
FFSQESVYCLACNVRWBBIREPBBVFEXOSCDYGZWPFDTKFQIY
CWHJVLNHIQIBTKHJVNPIST
```

(c) *Affine Cipher*:

```
KQEREJEBCPPCJCRKIEACUZBKRVPKRBCIBQCARBJCVFCUP
KRIOFKPACUZQEPBKRXPEIIEABDKPBCPFCDCCAFIEABDKP
BCPFEQPKAZBKRHAIBKAPCCIBURCCDKDCCJCIDFUIXPAFF
ERBICZDFKABICBBENEFCUPJCVKABPCYDCCDPKBCOCPERK
IVKSCPICBRKIJPKABI
```

(d) unspecified cipher:

works discussing the security of *Optimal Asymmetric Encryption Padding* and related cryptosystems against chosen-ciphertext attacks: Shoup [302], Fujisaki, Okamoto, Pointcheval and Stern [149] and Boneh [58].

Exercises 5.15–5.17 give some examples of protocol failures. For a nice article on this subject, see Moore [246].

Exercises

5.1 In Algorithm 5.1, prove that

$$\gcd(r_0, r_1) = \gcd(r_1, r_2) = \cdots = \gcd(r_{m-1}, r_m) = r_m$$

and, hence, $r_m = \gcd(a, b)$.

5.2 Suppose that $a > b$ in Algorithm 5.1.

(a) Prove that $r_i \geq 2r_{i+2}$ for all i such that $0 \leq i \leq m-2$.

(b) Prove that m is $O(\log a)$.

(c) Prove that m is $O(\log b)$.

5.3 Use the EXTENDED EUCLIDEAN ALGORITHM to compute the following multiplicative inverses:

(a) $17^{-1} \bmod 101$

(b) $357^{-1} \bmod 1234$

(c) $3125^{-1} \bmod 9987$.

5.4 Compute $\gcd(57, 93)$, and find integers s and t such that $57s + 93t = \gcd(57, 93)$.

5.5 Suppose $\chi : \mathbb{Z}_{105} \rightarrow \mathbb{Z}_3 \times \mathbb{Z}_5 \times \mathbb{Z}_7$ is defined as

$$\chi(x) = (x \bmod 3, x \bmod 5, x \bmod 7).$$

Give an explicit formula for the function χ^{-1}, and use it to compute $\chi^{-1}(2, 2, 3)$.

5.6 Solve the following system of congruences:

$$x \equiv 12 \pmod{25}$$
$$x \equiv 9 \pmod{26}$$
$$x \equiv 23 \pmod{27}.$$

5.7 Solve the following system of congruences:

$$13x \equiv 4 \pmod{99}$$
$$15x \equiv 56 \pmod{101}.$$

HINT First use the EXTENDED EUCLIDEAN ALGORITHM, and then apply the Chinese remainder theorem.

5.8 Use Theorem 5.8 to find the smallest primitive element modulo 97.

5.9 Suppose that $p = 2q + 1$, where p and q are odd primes. Suppose further that $\alpha \in \mathbb{Z}_p{}^*$, $\alpha \not\equiv \pm 1 \pmod{p}$. Prove that α is a primitive element modulo p if and only if $\alpha^q \equiv -1 \pmod{p}$.

5.10 Suppose that $n = pq$, where p and q are distinct odd primes and $ab \equiv 1 \pmod{(p-1)(q-1)}$. The RSA encryption operation is $e(x) = x^b \bmod n$ and the decryption operation is $d(y) = y^a \bmod n$. We proved that $d(e(x)) = x$ if $x \in \mathbb{Z}_n{}^*$. Prove that the same statement is true for any $x \in \mathbb{Z}_n$.

HINT Use the fact that $x_1 \equiv x_2 \pmod{pq}$ if and only if $x_1 \equiv x_2 \pmod{p}$ and $x_1 \equiv x_2 \pmod{q}$. This follows from the Chinese remainder theorem.

5.11 For $n = pq$, where p and q are distinct odd primes, define

$$\lambda(n) = \frac{(p-1)(q-1)}{\gcd(p-1, q-1)}.$$

Suppose that we modify the *RSA Cryptosystem* by requiring that $ab \equiv 1 \pmod{\lambda(n)}$.

(a) Prove that encryption and decryption are still inverse operations in this modified cryptosystem.

(b) If $p = 37$, $q = 79$, and $b = 7$, compute a in this modified cryptosystem, as well as in the original *RSA Cryptosystem*.

5.12 Two samples of RSA ciphertext are presented in Tables 5.1 and 5.2. Your task is to decrypt them. The public parameters of the system are $n = 18923$ and $b = 1261$ (for Table 5.1) and $n = 31313$ and $b = 4913$ (for Table 5.2). This can be accomplished as follows. First, factor n (which is easy because it is so small). Then compute the exponent a from $\phi(n)$, and, finally, decrypt the ciphertext. Use the SQUARE-AND-MULTIPLY ALGORITHM to exponentiate modulo n.

In order to translate the plaintext back into ordinary English text, you need to know how alphabetic characters are "encoded" as elements in $\mathbb{Z}_n$. Each element of $\mathbb{Z}_n$ represents three alphabetic characters as in the following examples:

$$\begin{array}{lcrcr} DOG & \rightarrow & 3 \times 26^2 + 14 \times 26 + 6 & = & 2398 \\ CAT & \rightarrow & 2 \times 26^2 + 0 \times 26 + 19 & = & 1371 \\ ZZZ & \rightarrow & 25 \times 26^2 + 25 \times 26 + 25 & = & 17575. \end{array}$$

You will have to invert this process as the final step in your program.

The first plaintext was taken from "The Diary of Samuel Marchbanks," by Robertson Davies, 1947, and the second was taken from "Lake Wobegon Days," by Garrison Keillor, 1985.

5.13 A common way to speed up RSA decryption incorporates the Chinese remainder theorem, as follows. Suppose that $d_K(y) = y^d \bmod n$ and $n = pq$. Define $d_p = d \bmod (p-1)$ and $d_q = d \bmod (q-1)$; and let $M_p = q^{-1} \bmod p$ and $M_q = p^{-1} \bmod q$. Then consider the following algorithm:

Algorithm 5.15: CRT-OPTIMIZED RSA DECRYPTION$(n, d_p, d_q, M_p, M_q, y)$

```
$x_p \leftarrow y^{d_p} \bmod p$
$x_q \leftarrow y^{d_q} \bmod q$
$x \leftarrow M_p q x_p + M_q p x_q \bmod n$
return $(x)$
```

Algorithm 5.15 replaces an exponentiation modulo n by modular exponentiations modulo p and q. If p and q are ℓ-bit integers and exponentiation modulo an ℓ-bit integer takes time $c\ell^3$, then the time to perform the required exponentiation(s) is reduced from $c(2\ell)^3$ to $2c\ell^3$, a savings of 75%. The final step, involving the Chinese remainder theorem, requires time $O(\ell^2)$ if d_p, d_q, M_p and M_q have been pre-computed.

Algorithm 5.1: EUCLIDEAN ALGORITHM(a, b)

```
$r_0 \leftarrow a$
$r_1 \leftarrow b$
$m \leftarrow 1$
while $r_m \neq 0$
    do {
        $q_m \leftarrow \lfloor \frac{r_{m-1}}{r_m} \rfloor$
        $r_{m+1} \leftarrow r_{m-1} - q_m r_m$
        $m \leftarrow m + 1$
    }
$m \leftarrow m - 1$
return $(q_1, \dots, q_m; r_m)$
comment: $r_m = \gcd(a, b)$
```

At this point, we know that any $b \in \mathbb{Z}_n{}^*$ has a multiplicative inverse, b^{-1}, but we do not yet have an efficient algorithm to compute b^{-1}. Such an algorithm exists; it is called the EXTENDED EUCLIDEAN ALGORITHM. However, we first describe the EUCLIDEAN ALGORITHM, in its basic form, which can be used to compute the greatest common divisor of two positive integers, say a and b. The EUCLIDEAN ALGORITHM sets r_0 to be a and r_1 to be b, and performs the following sequence of divisions:

$$\begin{array}{rcll}
r_0 & = & q_1 r_1 + r_2, & 0 < r_2 < r_1 \\
r_1 & = & q_2 r_2 + r_3, & 0 < r_3 < r_2 \\
\vdots & \vdots & \vdots & \vdots \\
r_{m-2} & = & q_{m-1} r_{m-1} + r_m, & 0 < r_m < r_{m-1} \\
r_{m-1} & = & q_m r_m. &
\end{array}$$

A pseudocode description of the EUCLIDEAN ALGORITHM is presented as Algorithm 5.1.

REMARK We will make use of the list $(q_1, \dots, q_m)$ that is computed during the execution of Algorithm 5.1 in a later section of this chapter. ∎

In Algorithm 5.1, it is not hard to show that

$$\gcd(r_0, r_1) = \gcd(r_1, r_2) = \cdots = \gcd(r_{m-1}, r_m) = r_m.$$

Hence, it follows that $\gcd(r_0, r_1) = r_m$.

Since the EUCLIDEAN ALGORITHM computes greatest common divisors, it can be used to determine if a positive integer $b < n$ has a multiplicative inverse modulo n, by calling EUCLIDEAN ALGORITHM(n, b) and checking to see if $r_m = 1$. However, it does not compute the value of $b^{-1} \bmod n$ (if it exists).

Now, suppose we define two sequences of numbers,

$$t_0, t_1, \dots, t_m \quad \text{and} \quad s_0, s_1, \dots, s_m,$$

according to the following recurrences (where the q_j's are defined as in Algorithm 5.1):

$$t_j = \begin{cases} 0 & \text{if } j = 0 \\ 1 & \text{if } j = 1 \\ t_{j-2} - q_{j-1} t_{j-1} & \text{if } j \geq 2 \end{cases}$$

and

$$s_j = \begin{cases} 1 & \text{if } j = 0 \\ 0 & \text{if } j = 1 \\ s_{j-2} - q_{j-1} s_{j-1} & \text{if } j \geq 2. \end{cases}$$

Then we have the following useful result.

THEOREM 5.1 *For $0 \leq j \leq m$, we have that $r_j = s_j r_0 + t_j r_1$, where the r_j's are defined as in Algorithm 5.1, and the s_j's and t_j's are defined in the above recurrence.*

PROOF The proof is by induction on j. The assertion is trivially true for $j = 0$ and $j = 1$. Assume the assertion is true for $j = i - 1$ and $i - 2$, where $i \geq 2$; we will prove the assertion is true for $j = i$. By induction, we have that

$$r_{i-2} = s_{i-2} r_0 + t_{i-2} r_1$$

and

$$r_{i-1} = s_{i-1} r_0 + t_{i-1} r_1.$$

Now, we compute:

$$\begin{aligned}
r_i &= r_{i-2} - q_{i-1} r_{i-1} \\
&= s_{i-2} r_0 + t_{i-2} r_1 - q_{i-1}(s_{i-1} r_0 + t_{i-1} r_1) \\
&= (s_{i-2} - q_{i-1} s_{i-1}) r_0 + (t_{i-2} - q_{i-1} t_{i-1}) r_1 \\
&= s_i r_0 + t_i r_1.
\end{aligned}$$

Hence, the result is true, for all integers $j \geq 0$, by induction. ∎

In Algorithm 5.2, we present the EXTENDED EUCLIDEAN ALGORITHM, which takes two integers a and b as input and computes integers r, s and t such that $r = \gcd(a, b)$ and $sa + tb = r$. In this version of the algorithm, we do not keep track of all the q_j's, r_j's, s_j's and t_j's; it suffices to record only the "last" two terms in each of these sequences at any point in the algorithm.

The next corollary is an immediate consequence of Theorem 5.1.