А. А. Набебин,
кандидат физико-математических наук, доцент кафедры программного обеспечения вычислительной техники РГСУ.
Базовое образование: факультет вычислительной математики и кибернетики Московского государственного университета им. М. В. Ломоносова.
Тема кандидатской диссертации: «О некоторых аксиоматических исчислениях первого и второго порядков».
Основные публикации: «Сборник заданий по дискретной математике» (2009), «Дискретная математика» (2010).
Сфера научных интересов: математическая логика, теория алгоритмов, теория конечных автоматов, дискретная математика.

СИСТЕМЫ ЛИНЕЙНЫХ УРАВНЕНИЙ В КОНЕЧНЫХ ПОЛЯХ ДЛЯ ИНФОРМАТИКИ

Аннотация: *приведена группа алгоритмов решения общих систем линейных уравнений в конечных полях, включающая: вычисление определителя, решение квадратной системы линейных уравнений, вычисление обратной матрицы, приведение матрицы к трапециевидной диагональной форме и вычисление ранга матрицы, вычисление фундаментальной системы решений линейных уравнений и алгоритмы решения однородной и неоднородной систем линейных уравнений.*

Ключевые слова: *конечное поле, матрица, определитель, квадратная линейная система, обратная матрица, фундаментальная система решений, общие однородная и неоднородная системы линейных уравнений.*

В проблемах защиты компьютерной информации широко используется теория конечных групп, колец, полей. В книгах [1; 2] целая глава в [2] посвящена вопросам матричной алгебры и решению линейных систем в простом бинарном конечном поле.

При решении задач линейной алгебры таких как вычисление ранга матрицы и вычисление определителя (квадратной матрицы), определение совместности системы линейных уравнений, решение системы линейных уравнений (нахождение ее фундаментальной системы решений и некоторого ее частного решения) и им подобных задач широко используется алгоритм Гаусса приведения

матрицы к трапециевидной диагональной форме с помощью двух линейных операций над строками матрицы: 1) умножение строки матрицы на элемент поля; 2) прибавление к строке матрицы любой другой строки. Аналогично для столбцов.

Если речь идет о системе линейных уравнений и об ее расширенной матрице (к матрице системы приписан столбец свободных членов системы), то умножение строки матрицы системы на элемент поля соответствует умножению соответствующего уравнения системы на этот элемент, в результате которого получается новая система, эквивалентная исходной (с тем же множеством решений). Аналогично относительно второй линейной операции.

Более всего в практике приходится решать системы линейных уравнений в поле вещественных чисел. Если алгоритм, написанный для решения таких задач, использует лишь свойства поля (например, вещественных чисел P) относительно сложения и умножения (алгоритм Гаусса именно таков), то тот же алгоритм будет справедлив для решения систем линейных уравнений над любым полем. При этом в написанном алгоритме сложение и умножение надо понимать как сложение и умножение в рассматриваемом поле. В компьютерную программу для алгоритма решения систем линейных уравнений в поле вещественных чисел надо инкорпорировать программы сложения и умножения элементов рассматриваемого поля и (там, где это необходимо) адаптировать для этого поля все другие команды программы.

Ниже следующие алгоритмы написаны в виде, удобном для программирования. Массивы базируются от нуля. При базировании от другого числа алгоритм, естественно, надо подправить на принятое базирование.

1. Алгоритм вычисления определителя матрицы. Пусть *rows(d)* есть число строк матрицы *d*.

ВХОД. Матрица *d*.

ВЫХОД. Определитель det матрицы *d*.

1. $n := rows(d)-1$.
2. $det := 1$.
3. Для k от 0 до n выполнить следующее.
   3.1. $r := 0$.
   3.2. Для l от k до n выполнить следующее.
      3.2.1. Если $r-|d_{l,k}| < 0$, то $l_p := l$, $r := d_{l,k}$.
   3.3. Если $r = 0$, то вернуть $det = 0$.
   3.4. Если $r \neq 0$, то выполнить следующее.
      3.4.1. Если $k \neq l_p$, то выполнить следующее.

3.4.1.1. $det := -det$.

3.4.1.2. Для j от k до n выполнить следующее.

$r := d_{k,j},\ d_{k,j} := d_{lp,j},\ d_{lp,j} := r$.

3.4.2. Если $k+1 \leq n$, то выполнить следующее.

3.4.2.1. Для j от $k+1$ до n выполнить: $d_{k,j} := d_{k,j}/d_{k,k}$.

3.5. Если $k = 0$, то выполнить следующее.

3.5.1. Для i от $k+1$ до n, для j от $k+1$ до n выполнить следующее.

$d_{i,j} := d_{i,j} - d_{k,j} \cdot d_{i,k}$.

3.6. Если $0 < k < n$, то выполнить следующее.

3.6.1. Для i от $k-1$ до n, для j от $k+1$ до n выполнить следующее.

$d_{i,j} := d_{i,j} - d_{k,j} \cdot d_{i,k}$.

3.6.2. Для i от $k+1$ до n, для j от $k+1$ до n выполнить следующее.

$d_{i,j} := d_{i,j} - d_{k,j} \cdot d_{i,k}$.

4. $det := det \cdot \prod_{i=0}^{n} d_{i,i}$.

5. Вернуть det.

Замечание. На шаге k алгоритма:

блок 3.2: 1) сравнивает между собой элементы $d_{l,k}$ столбца k строк l от k до n и выбирает среди них наибольший по модулю элемент $r = |d_{lp,k}|$;

блок 3.3 при $r = 0$ возвращает $det = 0$;

блок 3.4 при $r \neq 0$, если $k \neq lp$, 1) меняет знак определителя, 2) меняет местами строки k и lp, 3) делит строку k на $d_{k,k}$ (кроме элемента $d_{k,k}$);

блок 3.5 при $k = 0$ организует нули в столбце 0 определителя ниже элемента $d_{0,0}$;

блок 3.6: 1) при $0 < k < n$ организует нули в столбце k определителя выше и ниже элемента $d_{k,k}$, 2) при $k = n$ организует нули в столбце n определителя выше элемента $d_{n,n}$.

2. Алгоритм решения квадратной системы линейных уравнений. Пусть $rows(A)$ есть число строк матрицы A.

ВХОД. Квадратная система линейных уравнений $Ax=b$.

ВЫХОД. Решение линейной системы $Ax=b$.

1. $n := rows(A)-1;\ n1 := n+1$.

2. Для i от 0 до n, для j от 0 до n выполнить: $d_{i,j} := A_{i,j}$.

3. Для i от 0 до n $d_{i,n1} := b_i$.

4. Для k от 0 до n выполнить следующее.
   4.1. r := 0.
   4.2. Для l от k до n выполнить следующее.
      4.2.1. Если $r-|d_{l,k}| < 0$, то $lp := l$, $r := d_{l,k}$.
   4.3. Если $r = 0$, то вернуть «Определитель равен нулю».
   4.4. Если r ≠ 0, то выполнить следующее.
      4.4.1. Если k ≠ lp, то выполнить следующее.
         4.4.1.1. Для j от k до $n+1$ выполнить следующее.
            $r := d_{k,j}$, $d_{k,j} := d_{lp,j}$, $d_{lp,j} := r$.
      4.4.2. Для j от $k+1$ до $n+1$ выполнить: $d_{k,j} := d_{k,j}/d_{k,k}$.
   4.5. Если $k = 0$, то выполнить следующее.
      4.5.1. Для i от $k+1$ до n, для j от $k+1$ до $n+1$ выполнить следующее.
         $d_{i,j} := d_{i,j} - d_{k,j} \cdot d_{i,k}$.
   4.6. Если $k = n$, то выполнить следующее.
      4.6.1. Для i от 0 до $n-1$, для j от $k+1$ до $n+1$ выполнить следующее.
         $d_{i,j} := d_{i,j} - d_{k,j} \cdot d_{i,k}$.
   4.7. Если $0 < k < n$, то выполнить следующее.
      4.7.1. Для i от $k-1$ до n, для j от $k+1$ до $n+1$ выполнить следующее.
         $d_{i,j} := d_{i,j} - d_{k,j} \cdot d_{i,k}$.
      4.7.2. Для i от $k+1$ до n, для j от $k+1$ до $n+1$ выполнить следующее.
         $d_{i,j} := d_{i,j} - d_{k,j} \cdot d_{i,k}$.
5. Для i от 0 до n выполнить: $x_i := d_{i,n+1}$.
6. Вернуть x.

Замечание. Шаги алгоритма во многом совпадают с шагами алгоритма вычисления определителя матрицы.

3. Алгоритм вычисления обратной матрицы. Пусть *rows(A)* есть число строк матрицы A.

ВХОД. Матрица A.

ВЫХОД. Обратная матрица A^{-1}.

1. $n := rows(A)-1$.
2. $n2 := n \cdot 2+1$.
3. Для i от 0 до n, для j от 0 до n выполнить: $d_{i,j} := A_{i,j}$.
4. Для i от 0 до n, для j от 0 до n выполнить:
   $d_{i,n+1+j} := 0$, $d_{i,n+1+i} := 1$.
5. Для k от 0 до n выполнить следующее.

5.1. $k1 := k+1$, $k2 := k-1$, $r := 0$.

5.2. Для l от k до n выполнить следующее.

Если $r-|d_{l,k}| < 0$, то $lp := l$, $r := d_{l,k}$.

5.3. Если $r = 0$, то вернуть: A^{-1} не существует.

5.4. Если $r \neq 0$, то выполнить следующее.

5.4.1. Для j от k до $n2$ выполнить следующее.

$r := d_{k,j}$, $d_{k,j} := d_{lp,j}$, $d_{lp,j} := r$.

5.4.2. Для j от $k1$ до $n2$ выполнить: $d_{k,j} := d_{k,j}/d_{k,k}$.

5.5. Если $k = 0$, то выполнить следующее.

5.5.1. Для i от $k1$ до n, для j от $k1$ до $n2$ выполнить следующее.

$d_{i,j} := di,j - d_{k,j} \cdot d_{i,k}$.

5.6. Если $k = n$, то выполнить следующее.

5.6.1. Для i от 0 до $n-1$, для j от $k1$ до $n2$ выполнить следующее.

$d_{i,j} := d_{i,j} - d_{k,j} \cdot d_{i,k}$.

5.7. Если $0 < k < n$, то выполнить следующее.

5.7.1. Для i от $k-1$ до n, для j от $k+1$ до $n2$ выполнить следующее.

$d_{i,j} := d_{i,j} - d_{k,j} \cdot d_{i,k}$.

5.7.2. Для i от $k+1$ до n, для j от $k+1$ до $n2$ выполнить следующее.

$d_{i,j} := d_{i,j} - d_{k,j} \cdot d_{i,k}$.

6. Для i от 0 до n, для j от 0 до n выполнить: $A1_{i,j} = d_{i,n+1+j}$.

7. Вернуть A1.

Замечание. Блок 3 выполняет присвоение $d := A$.

Блок 4 к матрице d пристраивает справа диагональную единичную матрицу того же размера, что и d.

На шаге k алгоритма:

блок 5.2 сравнивает между собой элементы $d_{l,k}$ столбца k строк l от k до n и выбирает среди них наибольший по модулю элемент $r = |d_{lp,k}|$;

блок 5.3 при $r = 0$ возвращает: «Обратная матрица не существует»

блок 5.4 при $r \neq 0$, если $k \neq lp$, то 1) меняет местами строки k и lp, 2) делит строку k на $d_{k,k}$ (кроме элемента $d_{k,k}$);

блок 5.5 при $k = 0$ организует нули в столбце 0 матрицы ниже элемента $d_{0,0}$;

блок 5.6 при $k = n$ организует нули в столбце n матрицы выше элемента $d_{n,n}$;

блок 5.7 при $0 < k < n$ организует нули в столбце k матрицы выше и ниже элемента $d_{k,k}$;

блок 6 строит матрицу *A1* из последних n столбцов матрицы *d*;

блок 7 в качестве обратной матрицы A^{-1} возвращает матрицу *A1*.

4. Алгоритм приведения матрицы к трапециевидной диагональной форме и вычисление ранга матрицы. Пусть *rows(A)* и *cols(A)* есть число строк и число столбцов матрицы *A* соответственно.

ВХОД. Ненулевая матрица *A*.

ВЫХОД. Трапециевидная диагональная форма матрицы *A*.

1. *s := rows(A), n := cols(A)*.

2. Окаймить матрицу *A* строкой снизу (нумерация столбцов) и столбцом справа (нумерация строк), получить матрицу *B* и положить

A :=B.

3. *k := 0*.

4. Пока $k \leqslant rows(A)-1$, выполнять следующее.

4.1. Если строка $k = rows(A)-1$ нумерационная, то вернуть *A*.

4.2. Если строка *k* в матрице *A* нулевая, то удалить строку *k* из *A* и перейти (с тем же *k*) к пункту 4.

4.3. В прямоугольнике $[k, rows(A)-2] \times [k, cols(A)-2]$ найти строку r и столбец c с наибольшим по модулю элементом.

4.4. Если $r \neq k$, то в *A* поменять местами строки *k* и *r*.

4.5. Если $c \neq k$, то в *A* поменять местами столбцы *k* и *c*.

4.6. Если $A_{k,k} = 0$, то из *A* удалить строку *k* и перейти к пункту 4.

4.7. Если $A_{k,k} \neq 1$, то в *A* поделить строку *k* на $A_{k,k}$.

4.8. Для *i* от 0 до *rows(A)–2* выполнить следующее.

Если $i \neq k$, то к строке *i* в *A* прибавить строку *k*, умноженную на $-A_{i,k}$.

4.9. *k := k+1*.

Замечание 1. На шаге k алгоритма:

блок 4.3 в прямоугольнике $[k, rows(A)-2] \times [k, cols(A)-2]$ ищет в матрице *A* строку *r* и столбец c с наибольшим по модулю элементом. Может оказаться, что этот наибольший по модулю элемент есть ноль. Это значит, что все элементы прямоугольника $[k, rows(A)-2] \times [k, cols(A)-2]$ есть нули. А так как в результате работы алгоритма на шаге *k–1* в прямоугольнике $[k, rows(A)-2] \times [0, k-1]$ все элементы тоже нули, то все строки матрицы *A* от строки *k* до строки *rows(A)–2* (то есть от строки *k* и больше) все нулевые; алгоритм далее удалит эти последние нулевые строки (блоки 4.6 и 4.2) и закончит работу;

блок 4.8 организует нули в столбце *k* матрицы выше и ниже элемента $d_{k,k}$.

Замечание 2. Ранг исходной матрицы A есть число строк в полученной трапециевидной диагональной матрице.

5. Алгоритм решения однородной системы линейных уравнений. Дана однородная система линейных уравнений $Ax = 0$ с $s \times n$ матрицей A и вектор-столбцом x неизвестных $x_1,\ldots,x_n$. Пусть *rows(A)* и *cols(A)* есть, соответственно, число строк и число столбцов матрицы A.

ВХОД. Матрица A.

ВЫХОД. Фундаментальная система решений для $Ax = 0$.

1. Если матрица A нулевая, то вернуть ответ о произвольности решения системы.

2. $s :=$ *rows(A)*-1, $n :=$ *cols(A)*-1.

3. Окаймить матрицу A строкой снизу (нумерация столбцов), столбцом справа (нумерация строк), получить матрицу B. $A := B$.

Далее следует приведение (окаймленной) матрицы A к трапециевидной диагональной форме.

4. $k := 0$.

5. Пока $k \leqslant$ *rows(A)*-1, выполнить следующее.

5.1. Вернуть A, если $k =$ *rows(A)*-1.

5.2. Если строка k в матрице A нулевая, то удалить строку k из A и перейти к пункту 5 (с тем же k).

5.3. В прямоугольнике *[k, rows(A)–2]* $\times$ *[k, cols(A)–2]* найти строку r и столбец c с наибольшим по модулю элементом.

5.4. Если $r \neq k$, то поменять местами строки k и r.

5.5. Если $c \neq k$, то поменять местами столбцы k и c.

5.6. Если $A_{k,k} = 0$, то перейти к пункту 5 (с тем же k).

5.6. Если $k \neq$ *rows(A)*-1 и $A_{k,k} \neq 1$, то поделить строку k на $A_{k,k}$.

5.7. Если $s=0$ или $k=$*rows(A)*-2, то поделить строку k на $A_{k,k}$ и вернуть A как трапециевидную диагональную форму матрицы.

5.8. Если $k=0$, то выполнить следующее.

5.8.1. Для i от 1 до *rows(A)*-2 выполнить следующее.

К строке i прибавить строку k, умноженную на $-Ai,k$.

5.8.2. $k := k+1$ и перейти к пункту 5.

5.9. Если $0 < k <$ *rows(A)*-2, то выполнить следующее.

5.9.1. Для i от 0 до *rows(A)*-2 выполнить следующее.

Если $i \neq k$, то к строке i прибавить строку k, умноженную на $-A_{i,k}$.

5.9.2. k := k+1 и перейти к пункту 5.

5.10. Если $k = rows(A)-2$, то выполнить следующее.

5.10.1. Для i от 0 до $rows(A)-2$ выполнить следующее.

К строке i прибавить строку k, умноженную на $-A_{i,k}$.

5.10.2. $k := k+1$ и перейти к пункту 5.

Получена окаймленная трапециевидная диагональная матрица A. Обозначим ее через D. Это матрица линейной системы, которая эквивалентна исходной линейной системе. По полученной окаймленной трапециевидной диагональной матрице D вычисляется фундаментальная система решений (ФСР) линейной системы.

ВХОД. Окаймленная трапециевидная диагональная матрица D.

ВЫХОД. ФСР линейной системы.

1. $r := rows(D)-1$, $c := cols(D)-2$.
2. Если $cols(D) = rows(D)$, то выполнить следующее.
   2.1. Для i от 0 до $r-1$ положить $X := 0$.
   2.2. Вернуть X.
3. $cr := c-r$.
4. Для i от 0 до cr, для j от 0 до с положить $X_{i,j} := 0$.
5. Для i от 0 до cr, для j от 0 до cr выполнить следующее.
   $C_{i,j} := 1$, если $i = j$, $C_{i,j} := 0$ в противном случае.
6. Для t от 0 до cr, для i от r до с выполнить следующее.
   $k := D_{r,i}$, $X_{t,k} := C_{t,i-r}$.
7. Для t от 0 до cr, для i от 0 до $r-1$ выполнить следующее.

$$k := D_{r,i},\ X_{t,k} := -\sum_{s=0}^{cr} D_{i,r+s} \cdot C_{t,s}\ .$$

8. $X := X^T$.
9. Вернуть X.

Замечание. X есть матрица, столбцы которой есть векторы фундаментальной системы решений линейной системы.

6. Алгоритм решения неоднородной системы линейных уравнений. Дана неоднородная система линейных уравнений $Ax = b$ с $s \times n$ матрицей A, вектор-столбцом b свободных членов $b1,\ldots,bs$ и вектор-столбцом x неизвестных $x1,\ldots,xn$. Пусть $rows(A)$ и $cols(A)$ есть, соответственно, число строк и число столбцов матрицы A.

ВХОД. Матрица A и вектор-столбец b свободных членов.

ВЫХОД. Фундаментальная система решений для $Ax = 0$ и некоторое частное решение для Ax = b.

1. Построить расширенную матрицу *Ab* линейной системы. *A := Ab*.

2. Если матрица *A* нулевая, то вернуть ответ о произвольности решения системы.

3. *s := rows(A)–1, n := cols(A)–1*.

4. Окаймить матрицу *A* строкой снизу (нумерация столбцов), столбцом справа (нумерация строк), получить матрицу *B*. *A := B*.

Далее следует приведение (окаймленной) матрицы *A* к трапециевидной диагональной форме.

5. *k := 0*.

6. Пока $k \leqslant rows(A)–2$, выполнить следующее.

6.1. Если строка *k* имеет вид *(0,…,0,a)* при $a \neq 0$, то вернуть ответ: Система противоречива и решений не имеет.

6.2. Если строка *k* в матрице *A* нулевая, то удалить строку *k* из *A* и перейти к пункту 6 (с тем же *k*).

6.3. В прямоугольнике $[k, rows(A)–2] \times [k, cols(A)–3]$ найти строку *r* и столбец *c* с наибольшим по модулю элементом.

6.4. Если $r \neq k$, то поменять местами строки *k* и *r*.

6.5. Если $c \neq k$, то поменять местами столбцы *k* и *c*.

6.6. Если $k \neq rows(A)–1$ и $A_{k,k} \neq 1$, то поделить строку k на $A_{k,k}$.

6.7. Если $s=0$ или $k=rows(A)–2$, то поделить строку *k* на $A_{k,k}$ и вернуть *A* как трапециевидную диагональную форму матрицы.

6.8. Если $k=0$, то выполнить следующее.

6.8.1. Для *i* от *1* до *rows(A)–2* выполнить следующее.

К строке *i* прибавить строку *k*, умноженную на $–A_{i,k}$.

6.8.2. *k := k+1* и перейти к пункту 6.

6.9. Если $0 < k < rows(A)–2$, то выполнить следующее.

6.9.1. Для *i* от *0* до *rows(A)–2* выполнить следующее.

Если $i \neq k$, то к строке *i* прибавить строку *k*, умноженную на *–Ai,k*.

6.9.2. *k := k+1* и перейти к пункту 6.

6.10. Если $k = rows(A)–2$, то выполнить следующее.

6.10.1. Для *i* от *0* до *rows(A)–2* выполнить следующее.

К строке *i* прибавить строку *k*, умноженную на *–Ai,k*.

6.10.2. *k :=k+1* и перейти к пункту 6.

Получена окаймленная трапециевидная диагональная матрица *A*. Обозначим ее через *D*. Это расширенная матрица линейной системы, которая эквивалентна исходной линейной системе. Далее следует вычисление (какого-либо) частного решения линейной системы. Пусть *rows(D)* и *cols(D)* есть, соответственно, число строк и число столбцов матрицы *D*. Пусть *r=rows(D)–1*, *c=cols(D)–2*.

ВХОД. Окаймленная трапециевидная диагональная матрица *D*.

ВЫХОД. Какое-либо частное решения линейной системы.

1. *r := rows(D)–1, c := cols(D)–2*.
2. Для *i* от *0* до *c–1* положить $x_i := 0$.
3. Для *i* от *0* до *r–1* выполнить следующее.
   $k := D_{r,i}$, $x_k := D_{i,c}$.
4. Вернуть *x*.

Замечание. За частное решение *x* линейной системы принято решение, когда все свободные переменные системы с расширенной матрицей *D* принимают значение ноль.

По полученной окаймленной трапециевидной диагональной матрице *D* вычисляется фундаментальная система решений (ФСР) линейной системы.

ВХОД. Окаймленная трапециевидная диагональная матрица *D*.

ВЫХОД. ФСР линейной системы.

1. r := rows(D)–1, c := cols(D)–3.
2. Если *cols(D) = rows(D)*+1, то выполнить следующее.
   2.1. Для *i* от *0* до *r–1* положить *X := 0*.
   2.2. Вернуть *X*.
3. *cr := c–r*.
4. Для *i* от *0* до *cr*, для *j* от *0* до *c–1* положить $X_{i,j} := 0$.
5. Для *i* от *0* до *cr*, для *j* от *0* до *c–1* выполнить следующее.
   $C_{i,j} := 1$, если $i = j$, $C_{i,j} := 0$ в противном случае.
6. Для *t* от *0* до *cr*, для *i* от *r* до с выполнить следующее.
   $k := D_{r,i}$, $X_{t,k} := C_{t,i-r}$.
7. Для *t* от *0* до *cr*, для *i* от *0* до *r–1* выполнить следующее.

   $k := Dr,i$, $X_{t,k} := -\sum_{s=0}^{cr} D_{i,r+s} \cdot C_{t,s}$.
8. *X := XT*.
9. Вернуть *X*.

Замечание 1. *X* есть матрица, столбцы которой есть векторы фундаментальной системы решений линейной системы.

Замечание 2. Для решения однородной системы линейных уравнений можно использовать алгоритм решения неоднородной линейной системы с нулевым столбцом свободных членов.

Литература:

1. Болотов А. А., Гашков С. Б., Фролов А. Б., Часовских А. А. Элементарное введение в эллиптическую криптографию. Алгебраические и алгоритмические основы. М.: КомКнига, 2006. 328 с.

2. Болотов А. А., Гашков С. Б., Фролов А. Б. Элементарное введение в эллиптическую криптографию. Протоколы криптографии на эллиптических кривых. М.: КомКнига, 2006. 280 с.

3. Глухов М. М., Елизаров В. П., Нечаев А. А. Алгебра. Т.1,2. М.: Гелиос АРВ, 2002.

4. Лидл Р., Нидеррайтер Г. Конечные поля. М.: Мир, 1988. 822 с.

5. Набебин А. А. Линейные регистры сдвига с обратной связью в шифровании информации// Ученые записки РГСУ, № 1, 2010.

6. Нечаев А. А. Элементы криптографии. М.: Высшая школа, 1999. 109 с.

7. Фаддеев Д. К. Лекции по алгебре. М.: Наука, 1984. 416 с.