

Вычисление дополнительных определителей Di вначале требует формирования дополнительных матриц Abi , которые получаются путем замены i - столбца матрицы A на матрицу B :

$$ABi = \begin{matrix} & a_{00} & \dots & b_0 & \dots & a_{0n} \\ & a_{10} & \dots & b_1 & \dots & a_{1n} \\ & a_{20} & \dots & b_2 & \dots & a_{2n} \\ & \dots & \dots & \dots & \dots & \dots \\ & a_{n0} & \dots & b_i & \dots & a_{nn} \end{matrix}$$

Соответственно значения дополнительных определителей можно с помощью **MATLAB** получить следующим образом:

$$Di = \det(ABi);$$

Если главный определитель не равен нулю, то корни системы линейных уравнений вычисляются по формуле:

$$Xi = \frac{Di}{D}$$

В случае равенства нулю главного определителя, решение отсутствует, а соответствующую систему линейных уравнений называют **плохо определенной**.

Система называется **совместной**, если она имеет хотя бы одно решение, и **несовместной**, если у неё нет ни одного решения. Решения считаются различными, если хотя бы одно из значений переменных не совпадает. Совместная система с единственным решением называется **определённой**, при наличии более одного решения — **недоопределённой**.

Теорема Кронекера-Капелли

Матрица, получающаяся конкатенацией матрицы **A** и столбца правых частей **B** путем конкатенации по столбцам называется расширенной матрицей **AB** системы линейных уравнений.

$$AB = \begin{matrix} & a_{00} & a_{01} & a_{02} & \dots & a_{0n} & b_0 \\ & a_{10} & a_{11} & a_{12} & \dots & a_{1n} & b_1 \\ & a_{20} & a_{21} & a_{22} & \dots & a_{2n} & b_2 \\ & \dots & \dots & \dots & \dots & \dots & \dots \\ & a_{n0} & a_{n1} & a_{n2} & \dots & a_{nn} & b_n \end{matrix}$$

Теорема [Кронекер, Капелли]. Система совместна тогда и только тогда, когда ранг матрицы этой системы совпадает с рангом ее расширенной матрицы:

$$\text{rank}(A) = \text{rank}(AB)$$

Пример :

```
% Теорема Кронекера-Капелли. Система совместна
% тогда и только тогда,
% когда ранг матрицы этой системы совпадает с
% рангом ее расширенной матрицы.
A = rand(8)
B = rand(8,1)
AB = cat(2,A,B)    % Расширенная матрица
rA  = rank(A)
rAB = rank(AB)
if rA==rAB
    disp('Система линейных уравнений совместна');
else
    disp('Система линейных уравнений НЕ СОВМЕСТНА');
end;
```

Решение системы линейных уравнений методом Гаусса.

Основная идея метода

Пусть задана система линейных уравнений в виде:

[illegible]

Или в компактной форме:

$$\{ \sum_{k=0}^n a_k X_k = b \}, \quad \text{where } a_k, b \text{ are given constants, } X_k \text{ are independent random variables with given distributions.}$$

Основой решения системы линейных уравнений методом Гаусса, является последовательное исключение независимых переменных из строк системы. Такое исключение позволяет привести исходную систему к следующему эквивалентному виду:

$$\left\{ \begin{array}{cccccc} \mathbf{a}_{00}\mathbf{X}_0 & + \mathbf{a}_{01}\mathbf{X}_1 & + \mathbf{a}_{02}\mathbf{X}_2 & + \dots & + \mathbf{a}_{0n}\mathbf{X}_n & = \mathbf{b}_0 \\ 0 & + \mathbf{a}_{11}^*\mathbf{X}_1 & + \mathbf{a}_{12}^*\mathbf{X}_2 & + \dots & + \mathbf{a}_{1n}^*\mathbf{X}_n & = \mathbf{b}_1^* \\ 0 & + 0 & + \mathbf{a}_{22}^*\mathbf{X}_2 & + \dots & + \mathbf{a}_{2n}^*\mathbf{X}_n & = \mathbf{b}_2^* \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & + 0 & + 0 & + 0 & + \mathbf{a}_{nn}^*\mathbf{X}_n & = \mathbf{b}_n^* \end{array} \right. \quad (2)$$

Если оказывается возможным такое преобразование, то такую эквивалентную систему будем называть системой уравнений в форме Гаусса или просто Гауссовой системой. Разрешить систему уравнений в форме Гаусса достаточно просто. Для этого вначале следует найти корень X_n :

$$X_n = \frac{b_n^*}{a_{nn}^*}$$

После чего, последовательно от $k = n-1$ до $k = 0$ применяется формула, которая использует уже вычисленные значения корней:

$$X_k = \frac{b_k^* - \sum_{i=k+1}^n a_{ki}^* X_i}{a_{kk}^*}, \quad \forall k: 0 \leq k < n \quad (3)$$

Таким образом, основной задачей решения системы уравнений становится задача по последовательному исключению независимых переменных из строк системы уравнений, причем таким образом, чтобы после необходимых исключений новая система (2) оставалась эквивалентной исходной (1). Метод Гаусса предполагает решать названную задачу исключения последовательно, исключая в каждой последующей строке системы уравнений очередную независимую переменную. Такой метод становится возможным, если вспомнить, что любое равенство сохраняется если:

- обе части равенства умножить или поделить на одну и ту же величину;
- к одному равенству добавить или вычесть другое равенство.

Если такие операции применить к некоторой строке системы линейных уравнений и полученным результатом заменить исходную строку, то образованная система уравнений будет эквивалентна исходной, то есть, будет иметь такие же корни, как и исходная. Рассмотрим сформулированную идею метода Гаусса по отдельным шагам.

Метод исключения произвольной переменной из указанного уравнения в системе линейных уравнений.

Для определенности обозначим смысл используемых индексов:

- n – Число строк в системе линейных уравнений;
- m – Индекс независимой переменной X , которая подлежит исключению;
- k – Индекс строки системы уравнений, в которой исключается переменная X_m .

Для исключения переменной X_m из уравнения с k – тым строчным индексом последовательно выполним следующее:

Шаг 1.

Умножим уравнение с m – тым строчным индексом на величину:

$$\frac{a_{km}}{a_{mm}}$$

В результате такого умножения получим:

$$a_{m0} \frac{a_{km}}{a_{mm}} X_0 + a_{m1} \frac{a_{km}}{a_{mm}} X_1 + \dots + a_{mm} \frac{a_{km}}{a_{mm}} X_k + \dots + a_{mn} \frac{a_{km}}{a_{mm}} X_n = b_m \frac{a_{km}}{a_{mm}}$$

или в компактном виде:

$$\sum_{i=0}^n a_{mi} \frac{a_{km}}{a_{mm}} X_i = b_k \frac{a_{km}}{a_{mm}}, \quad \text{где } 0 < k \leq n \quad (4)$$

Шаг 2.

Вычтем уравнение, которое мы получили в результате выполнения шага 1 из исходного уравнения с k – тым строчным индексом. В результате вычитания одного равенства из другого, получим следующее:

$$(a_{k0} - a_{m0} \frac{a_{km}}{a_{mm}}) X_0 + \dots + (a_{km} - a_{mm} \frac{a_{km}}{a_{mm}}) X_k + \dots + (a_{kn} - a_{mn} \frac{a_{km}}{a_{mm}}) X_n = b_k - b_m \frac{a_{km}}{a_{mm}}$$

или в компактном виде:

$$\sum_{i=0}^n (a_{ki} - a_{mi} \frac{a_{km}}{a_{mm}}) X_i = b_k - b_m \frac{a_{km}}{a_{mm}} \quad (5)$$

Если, для наглядности, ввести обозначения:

$$a_{ki}^* = a_{ki} - a_{mi} \frac{a_{km}}{a_{mm}}$$

$$b_k^* = b_k - b_m \frac{a_{km}}{a_{mm}}$$

То выражение (5) можно записать в еще более компактной форме:

(5*)

Обратим внимание, что коэффициент при переменной X_m в этом выражении всегда будет равен нулю:

$$(a_{\text{km}} - a_{\text{mm}} \frac{a_{\text{km}}}{a_{\text{mm}}}) \equiv 0$$

Поскольку этот нулевой коэффициент является сомножителем для переменной X_m , то переменная X_m фактически исключается из выражения (5).

Шаг 3.

Кроме того, выражение (5), полученное на шаге 2, является линейно – зависимым вариантом исходного уравнения с k – тым строчным индексом, а следовательно это выражение может заменить собою исходное уравнение и при этом новая система уравнений останется эквивалентной исходной системе уравнений.

Метод построения треугольной матрицы путем последовательного исключения переменных.

Итак, теперь мы располагаем методом, который позволяет исключить любую переменную из любого уравнения системы линейных уравнений. Таким образом, преобразование системы линейных уравнений к виду (2) можно осуществить путем последовательного применения этого метода. Рассмотрим такое последовательное исключение:

Для начала, в исходной системе уравнений (см. 1) исключим переменную X_0 из всех уравнений кроме уравнения с нулевым строчным индексом. Это приведет нас к новой эквивалентной системе уравнений вида:

$$\left\{ \begin{array}{cccccc} \mathbf{a}_{00}\mathbf{X}_0 & + \mathbf{a}_{01}\mathbf{X}_1 & + \mathbf{a}_{02}\mathbf{X}_2 & + \dots\dots\dots & + \mathbf{a}_{0n}\mathbf{X}_n & = \mathbf{b}_0 \\ 0 & + \mathbf{a}_{11}^*\mathbf{X}_1 & + \mathbf{a}_{12}^*\mathbf{X}_2 & + \dots\dots\dots & + \mathbf{a}_{1n}^*\mathbf{X}_n & = \mathbf{b}_1^* \\ 0 & + \mathbf{a}_{21}^*\mathbf{X}_1 & + \mathbf{a}_{22}^*\mathbf{X}_2 & + \dots\dots\dots & + \mathbf{a}_{2n}^*\mathbf{X}_n & = \mathbf{b}_2^* \\ \dots\dots\dots & \dots\dots\dots & \dots\dots\dots & \dots\dots\dots & \dots\dots\dots & \dots\dots\dots \\ 0 & + \mathbf{a}_{n1}^*\mathbf{X}_1 & + \mathbf{a}_{n2}^*\mathbf{X}_2 & + \dots\dots\dots & + \mathbf{a}_{nn}^*\mathbf{X}_n & = \mathbf{b}_n^* \end{array} \right.$$

Обратим внимание, что некоторая часть в этой системе уравнений, а именно:

$$\left\{ \begin{array}{cccccc} a_{11}^* X_1 & + a_{12}^* X_2 & + \dots & + a_{1n}^* X_n & = b_1^* \\ a_{21}^* X_1 & + a_{22}^* X_2 & + \dots & + a_{2n}^* X_n & = b_2^* \\ \dots & \dots & \dots & \dots & \dots \\ a_{n1}^* X_1 & + a_{n2}^* X_2 & + \dots & + a_{nn}^* X_n & = b_n^* \end{array} \right.$$

стала независимой от переменной X_0 , а, следовательно, может быть разрешена как самостоятельная система линейных уравнений. Назовем такую систему уравнений подсистемой. Особо следует отметить, что размерность такой подсистемы стала на единицу меньше по отношению к исходной системе. Для нас, такое понижение размерности, может означать, что последовательное повторение процедуры исключения очередной переменной в каждой последующей подсистеме, обязательно приведет к финишной подсистеме с размерностью равной единице, а именно:

$$a_{nn}^* X_n = b_n^*$$

Очевидно, что финишная подсистема разрешима как обычное уравнение, а итоговая система уравнений, включающая финишную подсистему, будет иметь вид:

$$\left\{ \begin{array}{cccccc} a_{00} X_0 & + a_{01} X_1 & + a_{02} X_2 & + \dots & + a_{0n} X_n & = b_0 \\ & a_{11}^* X_1 & + a_{12}^* X_2 & + \dots & + a_{1n}^* X_n & = b_1^* \\ & & a_{22}^* X_2 & + \dots & + a_{2n}^* X_n & = b_2^* \\ & & & \dots & \dots & \dots \\ & & & & a_{nn}^* X_n & = b_n^* \end{array} \right.$$

Таким образом, в результате последовательного исключения одной переменной в каждой последующей подсистеме мы получили ответ на задачу, которая была сформулирована как выражение (2) и с помощью формулы (3) теперь можем вычислить все корни системы линейных уравнений (1).

(1) Пошаговая реализация метода Крамера

```
% Исходные данные
a=[-1 2 7; 4 3 1; 7 5 -8]
x1=[1; 2; 3]
b=a*x1
% Метод Крамера
% Пошаговая реализация
d=det(a)
% Вычисление x1
```

```

aw=a;
aw(:,1)=b;
d1=det(aw);
x1=d1/d
% Вычисление x2
aw=a;
aw(:,2)=b;
d2=det(aw);
x2=d2/d
aw=a;
% Вычисление x3
aw(:,3)=b;
d3=det(aw);
x3=d3/d

```

(2) Универсальная реализация метода Крамера

```

% Solving Linear Systems of Equations (KRAMER)
% РЕШЕНИЕ СИСТЕМЫ ЛИНЕЙНЫХ УРАВНЕНИЙ МЕТОДОМ КРАМЕРА
% Синтаксис:
%           [x]=fkramer01(a,b)
%
% a - Исходная матрица коэффициентов системы линейных
% уравнений вида:
%
%   a11 a12 a13 a14 ..... a1N
%   a21 a22 a23 a24 ..... a2N
%   .....
%   aN1 aN2 aN3 aN4 ..... aNN
%
% b - Исходная матрица свободных членов системы
% линейных уравнений вида:
%
%   b1
%   b2
%   .....
%   bN
%
% x - Вычисляемый вектор корней
%
function [x]=fkramer01(a,b)
if isempty(a) || isempty(b)
    disp('Одна из матриц не задана');
    x=NaN;
    return
end;
if (size(a,1) ~= size(a,2)) || (size(a,1) ~= size(b,1))
    disp('Размеры матриц системы линейных уравнений заданы
неправильно');
    x=NaN;
    return
end;

```



```

d=det(a);
if d ~= 0
    for n=1:size(a,2)
        buf=a(:,n);
        a(:,n)=b;
        x(n,1)=det(a)/d;
        a(:,n)=buf;
    end;
else
    x=NaN;
    if (sum(abs(b))==0) && (d == 0)
        x=zeros(size(a,2),1);
        disp('Система имеет тривиальное решение');
    else
        disp('Матрица плохо обусловлена');
    end;
end;
end;

```

(2-1) Тестирование реализации метода Крамера

```

% Тест точности метода КРАМЕРА
function maxerr=fctest01kramer(n)
if n < 2
    n=2
end;
if n > 500
    n=500
end;
% Создать случайную матрицу и случайные корни
A = rand(n,n);
x1=rand(n,1);
% Вычислить соответствующие свободные члены
b=A*x1;
% ВЫПОЛНИТЬ Тест
x2 = fkramer01(A,b);
err=x2-x1;
maxerr=max(abs(err));
grid on;
plot(err);

```

(2-2) Тестирование ($x=a \backslash b$) метода MatLab

```

% Тест точности метода ( $x=a \backslash b$ )
function maxerr=fctest01LSE01(n)
if n < 2
    n=2
end;
if n > 500
    n=500
end;
% Создать случайную матрицу и случайные корни
a = rand(n,n);

```

```

x1=rand(n,1);
% Вычислить соответствующие свободные члены
b=a*x1;
% ВЫПОЛНИТЬ Тест
x2 = a\b;
err=x2-x1;
maxerr=max(abs(err));
grid on;
plot(err);

```

(2-3) Тестирование ($x=\text{inv}(a)*b$) метода линейной алгебры

```

% Тест точности метода (x=inv(a)*b)
function maxerr=fetest01LSE02(n)
if n < 2
    n=2
end;
if n > 500
    n=500
end;
% Создать случайную матрицу и случайные корни
a = rand(n,n);
x1=rand(n,1);
% Вычислить соответствующие свободные члены
b=a*x1;
% ВЫПОЛНИТЬ Тест
x2 = inv(a)*b;
err=x2-x1;
maxerr=max(abs(err));
grid on;
plot(err);

```

(3) РЕШЕНИЕ СИСТЕМЫ ЛИНЕЙНЫХ УРАВНЕНИЙ МЕТОДОМ ГАУССА

```

% Solving Linear Systems of Equations (GAUSS)
% РЕШЕНИЕ СИСТЕМЫ ЛИНЕЙНЫХ УРАВНЕНИЙ МЕТОДОМ ГАУССА
% Синтаксис:
%           [x]=fgauss01(ab)
%
% ab - Исходная матрица системы линейных уравнений вида:
%
%   a11 a12 a13 a14 .... a1N b1
%   a21 a22 a23 a24 .... a2N b2
%   .....
%   aN1 aN2 aN3 aN4 .... aNN bN
%
% x - Вычисляемый вектор корней
%
function [x]=fgauss01(ab)
if isempty(ab)
    disp('Матрица системы линейных уравнений не задана');

```

```

        x=NaN;
        return
    end;
    nb = size(ab,2);
    nx = size(ab,1);
    if (nb < 2) || (nb ~= (nx+1))
        disp('Размеры матрицы системы линейных уравнений заданы
неправильно');
        x=NaN;
        return
    end;
    % Построение Гауссовой матрицы
    for nM=1:(nx-1)
        % Установка на главную диагональ строки с максимальным
элементом ab(nM,nM)
        [v,n]=max(abs(ab(:,nM))));
        if (n > nM)
            s=ab(n,:);
            ab(n,:) = ab(nM,:);
            ab(nM,:)=s;
        end;
        % Ловушка неразрешимых ситуаций
        if ab(nM,nM) == 0
            disp('Матрица плохо обусловлена');
            x=NaN;
            return;
        end;
        % Построение треугольной Гауссовой матрицы
        for row =(nM + 1):nx
            koef =ab(row,nM)/ab(nM,nM);
            for n=1:nb
                if n == nM
                    ab(row,n)=0;
                else
                    ab(row,n)=ab(row,n)-ab(nM,n)*koef;;
                end;
            end;
        end;
    end;
    % Вычисление корней
    % Ловушка неразрешимых ситуаций
    if ab(nx,nx) == 0
        disp('Матрица плохо обусловлена');
        x=NaN;
        return;
    end;
    x=zeros(nx,1);
    row = nx;
    x(nx)=ab(row,nb)/ab(row,nx); % Последний корень
    while row > 1
        row=row-1;
        s=0;
        for n=1:nx

```

```

        if ab(row,n) ~= 0
            s=s+ab(row,n)*x(n);
        end;
    end;
    x(row)=(ab(row,nb)-s)/ab(row,row); % Очередной корень
end;

```

(3-1) Тестирование реализации метода ГАУССА

```

% Тест точности метода ГАУССА
function maxerr=ftest01gauss(n)
if n < 2
    n=2
end;
if n > 500
    n=500
end;
% Создать случайную матрицу и случайные корни
a = rand(n,n);
x1=rand(n,1);
% Вычислить соответствующие свободные члены
b=a*x1;
% Создать матрицу линейных уравнений
ab=cat(2,a,b);
% ВЫПОЛНИТЬ Тест
x2 = fgauss01(ab);
err=x2-x1;
maxerr=max(abs(err));
grid on;
plot(err);

```

(4) Сравнительный тест точности методов решения SLE

```

% Сравнительный тест точности методов SLE
function [maxerr]=ftest01LSE_All(n)
if n < 2
    n=2
end;
if n > 100
    n=100
end;
% Создать случайную матрицу и случайные корни
a = rand(n,n);
x1=rand(n,1);
b=a*x1;
ab=cat(2,a,b);
% Тест 1
x2 = a\b;
maxerr(1,1)=max(abs(x2-x1));

```

```

% Тест 2
x2 = inv(a)*b;
maxerr(2,1)=max(abs(x2-x1));
% Тест 3 (Kramer)
x2 = fkramer01(a,b);
maxerr(3,1)=max(abs(x2-x1));
% Тест 4 (Gauss)
x2 = fgauss01(ab);
maxerr(4,1)=max(abs(x2-x1));
% Отчет
[v,ind]=max(abs(maxerr));
switch ind
    case 1
        disp(strcat('Max. ошибка при x=A\b. Ошибка : ',
num2str(v)));
    case 2
        disp(strcat('Max. ошибка при x=inv(A)*b. Ошибка : ',
num2str(v)));
    case 3
        disp(strcat('Max. ошибка при x=fkramer01(A,b). Ошибка
: ', num2str(v)));
    case 4
        disp(strcat('Max. ошибка при x=fgauss01(ab). Ошибка :
', num2str(v)));
end;

```

(5) Сравнительный тест СКОРОСТИ методов GAUSS и KRAMER

```

% Сравнительный тест СКОРОСТИ методов GAUSS и KRAMER
function ftest02_Gauss_Kramer(nm)
if (nm < 100) || (nm > 200)
    nm=150;
    disp(strcat('Параметр вызова заменен на :', num2str(nm)));
end;
disp('Сравнительный тест СКОРОСТИ методов GAUSS (blue) и
KRAMER (red)');
disp(strcat('Сечас будет выполнено вычисление : ',
num2str(2*nm), ' систем уравнений'));
disp('ОЖИДАЙТЕ ЗАВЕРШЕНИЯ длительной операции');
pproc=0;
nproc=0;
for n = 1:nm
    A = rand(n,n);
    b = rand(n,1);
    AB = cat(2,A,b);
    x(n)=n;
    tic
        y2 = fgauss01(AB);
    t1(n) = toc;
    tic
        y2 = fkramer01(A,b);

```

```

t2(n) = toc;
% Процент выполнения
nproc=100*(n-rem(n,10))/nm;
if nproc > pproc
    disp(strcat('Выполнено :', num2str(nproc), '%'));
    pproc=nproc;
end;
end;
clf; % Очистка графиков
disp('ГОТОВО! Смотрите графики ...');
hold on;
plot(x,t1); % GAUSS
plot(x,t2,'-r'); % KRAMER

```

ПРАКТИЧЕСКАЯ РАБОТА.

1. Выполните все примеры из теоретической части лабораторной работы.
2. Восстановите блок – алгоритмическую схему решения системы линейных уравнений методом Гаусса.
3. Напишите функцию MATLAB для интерполяции некоторой таблично заданной функции степенным рядом.

Примечание :

Идея интерполяции степенным рядом заключается в нахождении коэффициентов некоторого полинома $P(X)$, который на конкретном участке табличной функции гарантированно проходит через заданные ее значения $F(X_i)$. Коэффициенты A_i такого полинома можно определить путем составления системы линейных уравнений. Подчеркнем, что неизвестными в такой системе линейных уравнений будут искомые нами коэффициенты A_i .

Рассмотрим пример составления такой системы линейных уравнений для полинома $P(X)$ второй степени:

$$P(X) = A_0 + A_1 \cdot X + A_2 \cdot X^2$$

Если значения полинома $P(X)$ при X_i будут совпадать со значениями $F(X_i)$ в точках X_i , то это представимо системой линейных уравнений, которая должна иметь вид:

$$\begin{cases} Y_i &= A_0 + A_1 \cdot X_i + A_2 \cdot X_i^2 \\ Y_{i+1} &= A_0 + A_1 \cdot X_{i+1} + A_2 \cdot X_{i+1}^2 \\ Y_{i+2} &= A_0 + A_1 \cdot X_{i+2} + A_2 \cdot X_{i+2}^2 \end{cases}$$

Осталось только разрешить эту систему относительно ее неизвестных A_i . Теперь у нас появилась возможность вычислять $F(X)$ для произвольных X внутри области определения $(X_i \dots X_{i+2})$, что и является основной целью интерполяции.

Финальная редакция материала 23.08.2018г.
Воронов С.И.