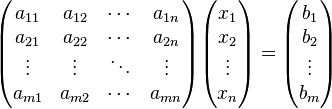
Лабораторная работа №2

«Метод квадратного корня»

выполнил Пажитных Иван, 2-й курс, 1-я группа

1. **Постановка задачи**

Необходимо найти решение системы линейных алгебраических уравнений вида Ах=b, где A- квадратная матрица n-ого порядка, х и b – столбцы размеров n×1.



Предполагается, что det A=|A|≠0. Тогда решение системы существует и оно единственно.

Нам необходимо найти:

a) Вектор решений (с точностью 6 значащих цифр)

b) Найти вектор невязки (r=Ax-b)

c) Найти определитель матрицы

**2) Алгоритм решения**

Метод квадратного корня состоит в нахождении матриц S и D таких, что: , где S – верхнетреугольная матрица, S\* – нижняетреугольная, а D – диагональная (на главной диагонали 1 и -1) Найдем значения Sij матрицы S. Для этого приравняем элементы матрицы A к элементам матрицы S\*SD и получим следующие значения:

sii – sij\*dii + ∑ski\*dii\*ski = aij, i=1..j, j=1…n.

Решая данную систему относительно элементов sii, sij и dii. получим следующее:

di i = sign( aii – ∑k=1i-1|ski|^2\*dkk) , i=1..n

sii = √( |aii – ∑k=1i-1|ski|^2\*dkk| ) , i=1..n

sij = ( aii – ∑k=1i-1ski\*dkk\*skj ) / (sii\*dii), i<j, j=1..n

Если искомое разложение найдено, то решение исходной системы сводится к решению системы из двух уравнений с треугольными матрицами:

Алгоритм решения этой системы:

а) прямой ход: последовательное нахождение элементов yi по указанным выше формулам и вычисление значений следующим образом:

y1 = b1/s11 ; yi = (bi – ∑k=1i-1ski\*yk) / sii , i>1

б) обратный ход: нахождение решения системы:

xn = yn/snn ; xi = (yi – ∑k=i+1nski\*xk) / sii , i<n

**3) Листинг программы**

for line in inp: *# читаем А и b*

temp = [float(x) for x in line.split()]

B.append(temp.pop())

A.append(temp)

a = np.array(A)

At = a.transpose() *# находим Аt*

b = np.array(B).transpose()

a = np.dot(At, a) *# перемножаем А на Аt, теперь А - симметрическая*

b = np.dot(At, b) *# то же самое с b*

s = zeros((n, m))

d = zeros(n)

s[0][0] = sqrt(a[0][0])

d[0] = copysign(1, a[0][0])

for i in range(n): *# находим S и D по известным формулам*

val = a[i][i]-fsum([(s[k][i]\*\*2)\*d[k] for k in range(i)])

d[i] = copysign(1, val)

s[i][i] = sqrt(fabs(val))

for j in range(i, n):

s[i][j] = (a[i][j] - fsum([s[k][i]\*d[k]\*s[k][j] for k in range(i)]))/(s[i][i]\*d[i])

det = np.prod([(s[i][i]\*\*2)\*d[i] for i in range(n)])  *# вычисляем определитель*

y = zeros(n)

y[0] = b[0]/s[0][0] *# находим y*

for i in range(1, n):

y[i] = (b[i] - fsum([s[k][i]\*y[k] for k in range(i)]))/s[i][i]

x = zeros(n)

x[n-1] = y[n-1]/s[n-1][n-1] *# находим x*

for i in range(n-2, -1, -1):

x[i] = (y[i] - fsum([s[i][k]\*x[k] for k in range(i+1, n)]))/s[i][i]

r1 = np.dot(A, x)-np.array(B) *# вычисляем невязку исходной*

r2 = np.dot(a, x)-b.transpose() *# вычисляем невязку симметричной*

**4) Результат и его анализ**

Расширенная матрица коэффициентов *A|b*:

[ 0.6897 -0.0908 0.0182 0.0363 0.1271 | 4.2108]

[ 0.0944 1.0799 0.0000 -0.0726 0.0726 | 4.6174]

[ 0.0545 0.0000 0.8676 -0.2541 0.1452 | -5.8770]

[ -0.1089 0.2287 0.0000 0.8531 -0.0363 | 2.7842]

[ 0.4538 0.0000 0.1634 0.0182 1.0164 | 0.2178]

Симметрическая *А*:

[ 0.70536135 0.01441237 0.13398766 -0.08030921 0.5676231]

[ 0.01441237 1.22673234 -0.00165256 0.11340719 0.05855825]

[ 0.13398766 -0.00165256 0.77976056 -0.21682262 0.2943685]

[-0.08030921 0.11340719 -0.21682262 0.79926611 -0.0500214]

[ 0.5676231 0.05855825 0.2943685 -0.0500214 1.07689486]

Новый столбец свободных членов *b*:

[ 2.81541308 5.24073616 -4.98666012 3.69013948 0.13738098]

Матрица *S*:

[ 0.83985793 0.01716049 0.1595361 -0.09562237 0.67585609]

[ 0. 1.10744655 -0.00396432 0.10388593 0.04240406]

[ 0. 0. 0.86850048 -0.23161249 0.21498337]

[ 0. 0. 0. 0.85187196 0.07042515]

[ 0. 0. 0. 0. 0.75308549]

Матрица *D*:

[ 1. 1. 1. 1. 1.]

Вектор решений *x*:

[ 7.00098534 3.99994603 -6.00022737 2.99988036 -2.00059801]

Определитель матрицы *det (A\*A) (detA)*:

0.268555796909 (0.51822369389)

Вектор невязки исходной (A) *r1*:

[ 0.00000000e+00 -2.66453526e-15 -8.88178420e-16 4.44089210e-16 1.05471187e-15]

Вектор симметрической (A\*A) *r1*:

[0.00000000e+00 -3.55271368e-15 -8.88178420e-16 4.44089210e-16 -3.05311332e-16]

Норма *||r1||*:

5.05151476204e-15

Норма *||r2||*:

5.19029264012e-15

Вывод: вектор невязки получился порядка 10^(-15) из этого следует, что решение найдено з достаточной точностью.