# Magdalena Wiechczyńska 132337 06.2019r.

# Rozwiązywanie układu równań liniowych metodą Gaussa-Jordana z pełnym wyborem elementu podstawowego

# Zastosowanie

Procedura rozwiązuje układ równań liniowych postaci

. (1)

# Opis metody

Układ równań jest rozwiązywany metodą eliminacji Gaussa-Jordana z pełnym wyborem elementu podstawowego. Metoda ta polega na przekształceniu układu (1) – przez odpowiednie przestawienie i kombinacje liniowe równań – do układu równań postaci

który ma takie samo rozwiązanie, jak układ (1). Ponieważ macierz R jest macierzą trójkątną górną, do rozwiązania układu (2) (jeżeli ) można zastosować wzór

Pierwszy krok algorytmu Gaussa-Jordana z pełnym wyborem elementu podstawowego można opisać w trzech następujących punktach.

i jeżeli , to przechodzimy do punktu 2. W przeciwnym przypadku macierz A jest osobliwa i procedura zostaje zakończona.

1. Przestawiamy wiersze r-ty i pierwszy, a następnie kolumny s-tą i pierwszą macierzy A, otrzymując w wyniku macierz .
2. Od j-tego wiersza (j = 2, 3, … , n) macierzy odejmujemy -krotność wiersza pierwszego, przy czym

W wyniku tej operacji otrzymujemy macierz postaci

Drugi krok metody polega na zastosowaniu powyższego algorytmu do macierzy . Po n-1 krokach otrzymujemy układ postaci (2).

# Wywołanie procedury

# Dane

liczba równań układu (1),

procedura języka Turbo Pascal, która dla danego i oblicza elementy i-tego wiersza macierzy układu równań (1).

# Wyniki

tablica zawierająca rozwiązanie (element zawiera wartość ), Inne parametry

# Inne parametry

zmienna, której w procedurze przypisuje się jedną z następujących wartości:

* 1, jeżeli ,
* 2, gdy macierz układu (1) jest osobliwa,
* 0, w przeciwnym wypadku.

Jeżeli , to elementy tablicy x nie są obliczane.

# Typy parametrów

# Identyfikatory nielokalne

nazwa typu tablicowego o elementach typu

nazwa typu tablicowego o elementach typu

identyfikator typu proceduralnego zdefiniowany następująco:

(na wyjściu element a[j] powinien zawierać wartość

# Kod źródłowy

1. procedure GaussJordanInterval;
2. var i,j,jh,k,kh,l,lh,n1,p,q,rh : Integer;
3. max,s : Interval;
4. a,b : vectorInt;
5. r : vector2;
6. begin
7. SetLength(a,n+1);
8. SetLength(b,n+1);
9. SetLength(r,n+1);
10. st:=0;
11. if n<1
12. then st:=1;
13. if st=0
14. then begin
15. n1:=n+1;
16. p:=n1;
17. for i:=1 to n1 do
18. r[i]:=0;
19. k:=0;
20. repeat
21. k:=k+1;
22. oneeqn (k,a);
23. for i:=1 to n do
24. begin
25. rh:=r[i];
26. if rh<>0
27. then b[rh]:=a[i]
28. end;
29. kh:=k-1;
30. l:=0;
31. max:=0;
32. for j:=1 to n1 do
33. if r[j]=0
34. then begin
35. s:=a[j];
36. l:=l+1;
37. q:=l;
38. for i:=1 to kh do
39. begin
40. s:=s-b[i]\*x[q];
41. q:=q+p
42. end;
43. a[l]:=s;
44. s:=iabs(s);
45. if (j<n1) and (s>max)
46. then begin
47. max:=s;
48. jh:=j;
49. lh:=l
50. end
51. end;
52. if max=0
53. then st:=2
54. else begin
55. max:=1/a[lh];
56. r[jh]:=k;
57. for i:=1 to p do
58. a[i]:=max\*a[i];
59. jh:=0;
60. q:=0;
61. for j:=1 to kh do
62. begin
63. s:=x[q+lh];
64. for i:=1 to p do
65. if i<>lh
66. then begin
67. jh:=jh+1;
68. x[jh]:=x[q+i]-s\*a[i]
69. end;
70. q:=q+p
71. end;
72. for i:=1 to p do
73. if i<>lh
74. then begin
75. jh:=jh+1;
76. x[jh]:=a[i]
77. end;
78. p:=p-1
79. end
80. until (k=n) or (st=2);
81. if st=0
82. then for k:=1 to n do
83. begin
84. rh:=r[k];
85. if rh<>k
86. then begin
87. s:=x[k];
88. x[k]:=x[rh];
89. i:=r[rh];
90. while i<>k do
91. begin
92. x[rh]:=x[i];
93. r[rh]:=rh;
94. rh:=i;
95. i:=r[rh]
96. end;
97. x[rh]:=s;
98. r[rh]:=rh
99. end
100. end
101. end
102. end;

# Przykłady

## Przykład I

***Dane*:**

3,9859265; -374,8678824; -8,5662110918; 4,8650891; 0,99864123; 1;

42,7856242; -4,5346826; 3,86425767; -0,7643424; 6,754368; 2;

0,54576547138; 46,586424689; -0,632539975; 4,234342458; 75,535008858; 0;

0,583563489427; 0,9583924; -9,9543726; 0,83546556; -857,834678; -1;

0,34809870124; -7,7769750323; 97,64870949; 8,56801507; 80; -1;

n=5

***Wyniki*:**

Arytmetyka zmiennopozycyjna

x[1] = 4,71471013654245E-0002

x[2] = -2,02861966576158E-0003

x[3] = -1,08734993925384E-0002

x[4] = -8,80667900711923E-0003

x[5] = 1,31313215660162E-0003

st = 0

## Przykład II

***Dane*:**

1,0; 4,5; 1,2; 2,1; 2,52; 2,1;

5,5; 3,3; 9,9; 1,848; 2,31; 1,98;

2,2; 1,485; 4,752; 9,24; 1,188; 1,0395;

7,15; 5,148; 1,716; 3,432; 4,5045; 4,004;

2,002; 1,5015; 5,148; 1,05105; 1,4014; 1,26126;

n=5

***Wyniki*:**

Arytmetyka przedzialowa

x[1] = [-7.7105461283123540E-0003; -7.7105461283123446E-0003] szer=9,26992857475106E-18

x[2] = [-1.0075588680675230E-0001; -1.0075588680675228E-0001] szer=1,51788304147971E-17

x[3] = [ 8.1505200045498097E-0004; 8.1505200045498497E-0004] szer=3,9979955110403E-18

x[4] = [-5.7063465615298490E-0004; -5.7063465615298350E-0004] szer=1,39591029707509E-18

x[5] = [ 1.0164017091809813E+0000; 1.0164017091809814E+0000] szer=2,74303149638833E-17

st = 0

## Przykład III

***Dane*:**

0; 0; 1; 4;

2; 1; 5; 0;

0; 0; 0; 0;

n=3

***Wyniki*:**

Arytmetyka przedzialowa

x[1] = [ 0.0000000000000000E+0000; 0.0000000000000000E+0000] szer=0

x[2] = [ 4.0000000000000000E+0000; 4.0000000000000000E+0000] szer=0

x[3] = [ 5.0000000000000000E-0001; 5.0000000000000000E-0001] szer=0

st = 2