Rozwiązywanie układu równań liniowych z macierzą symetryczną dodatnio określoną metodą Choleskiego

Adam Szczepański

1. **Zastosowanie**

Niech będzie macierzą przedziałów spełniającą . Niech będzie dane. Chcemy ograniczyć zestaw rozwiązań

przez wektor przedziałów .

1. **Opis metody**

Metoda polega na znalezieniu macierzy dolnotrójkątnej

takiej, że , a następnie obliczeniu rozwiązania na podstawie wzorów

1. **Wywołanie procedury**

*symposmatrix\_ia(n, a, b, x, st)*

1. **Dane**

*n –* liczba równań układu  
*a –* tablica z wartościami elementów macierzy układu (zawiera przedział ); macierz powinna być symetryczna i dodatnio określona  
*b –* tablica z wartościami elementów macierzy układu (zawiera przedział *)*

1. **Wynik**

*x –* tablica z rozwiązaniem układu ( zawiera przedział *)*

1. **Inne parametry**

*st –* zmienna wyjściowa, której przypisywany jest status obliczeń, tzn:  
1, jeżeli *n < 1*  
2, jeżeli macierz układu nie jest symetryczna  
3, jeżeli macierz układu nie jest dodatnio określona  
0, w przeciwnym wypadku

1. **Typy parametrów**

*Integer: n, st  
matrix\_ia: a  
vector\_ia: b, x*

1. **Identyfikatory nielokalne**

*vector\_ia –* nazwa typu tablicowego o elementach typu *interval*  
*matrix\_ia –* nazwa typu tablicowego o elemtach typu *interval*

1. **Treść procedury**
2. **procedure** symposmatrix\_ia (n      : **Integer**;
3. **var** a  : matrix\_ia;
4. b      : vector\_ia;
5. **var** x  : vector\_ia;
6. **var** st : **Integer**);
7. **var** i,j,k : **Integer**;
8. z,tmp,a1 : interval;
9. **begin**
10. a1:=int\_read('1');
11. st:=0;
12. **if** n<1
13. **then** st:=1;
14. **for** i:=2 **to** n **do**
15. **for** j:=1 **to** i-1 **do**
16. **if** ((a[i,j].a<>a[j,i].a) **or** (a[i,j].b<>a[j,i].b))
17. **then** st:=2;
18. **if** st=0
19. **then** **begin**
20. i:=0;
21. **repeat**
22. i:=i+1;
23. **for** j:=i **to** n **do**
24. **begin**
25. z:=a[i,j];
26. **for** k:=i-1 **downto** 1 **do**
27. z:=(isub(z,imul(a[j,k],a[i,k])));
28. **if** i=j
29. **then** **if** ((z.a>0) **and** (z.b>0))
30. **then** **begin**
31. tmp.a:=sqrt(z.a);
32. tmp.b:=sqrt(z.b);
33. a[i,i]:=idiv(a1,tmp)
34. **end**
35. **else** st:=3
36. **else** a[j,i]:=imul(z,a[i,i])
37. **end**
38. **until** (i=n) **or** (st=3);
39. **if** st=0
40. **then** **begin**
41. **for** i:=1 **to** n **do**
42. **begin**
43. z:=b[i];
44. **for** j:=i-1 **downto** 1 **do**
45. z:=isub(z,imul(a[i,j],x[j]));
46. x[i]:=imul(z,a[i,i])
47. **end**;
48. **for** i:=n **downto** 1 **do**
49. **begin**
50. z:=x[i];
51. **for** j:=i+1 **to** n **do**
52. z:=isub(z,imul(a[j,i],x[j]));
53. x[i]:=imul(z,a[i,i])
54. **end**
55. **end**
56. **end**
57. **end**;
58. **Przykłady**
    1. Dane:  
       Wyniki:
    2. Dane:  
         
       Wyniki:
    3. Dane:  
       Wyniki:
    4. Dane:  
         
       Wyniki: