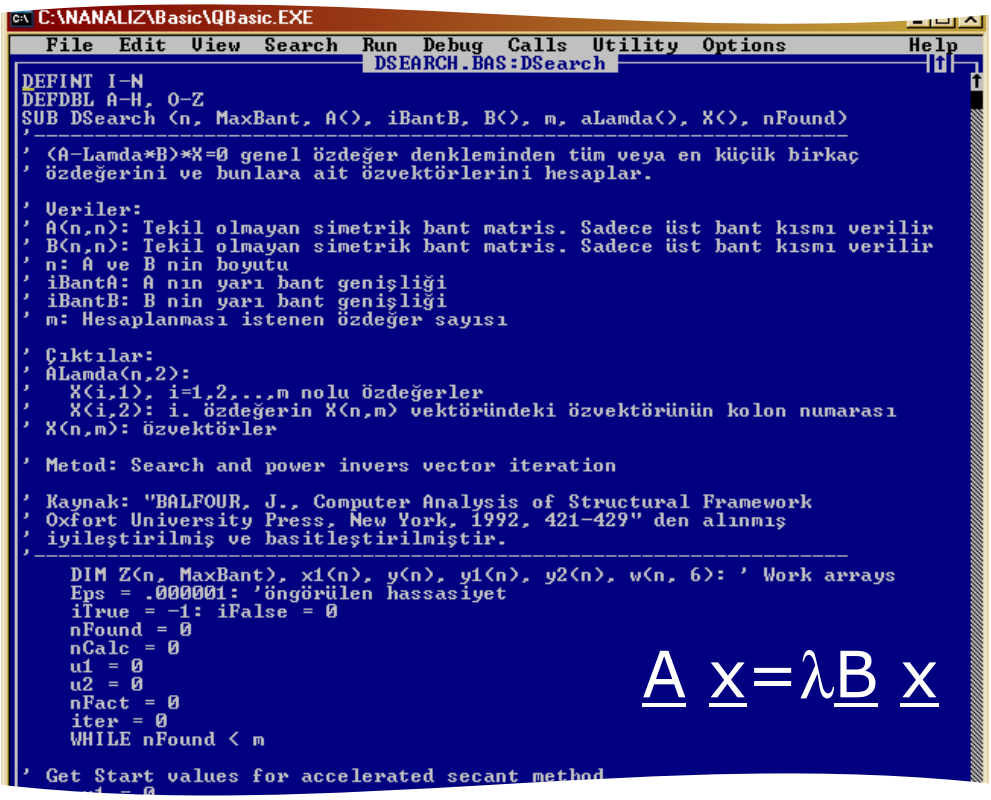
ESKİŞEHİR OSMANGAZİ ÜNİVERSİTESİ
Mühendislik Mimarlık Fakültesi
İnşaat Mühendisliği Bölümü
E-Posta: ogu.ahmet.topcu@gmail.com
Web: http://mmf2.ogu.edu.tr/atopcu

Bilgisayar Destekli
Nümerik Analiz

Ders notları 2014

Ahmet TOPÇU



35

PROGRAMLAR: Genel özdeğer ve özvektör hesabı

- En küçük birkaç özdeğerin hesabı - Dsearch

35. PROGRAMLAR: Genel özdeğer ve özvektör hesabı - DSearch

$$\underline{A}\underline{x} = \lambda \underline{B}\underline{x}$$

Genel özdeğer probleminin en küçük m tane veya tüm özdeğerlerini ve bunlara ait özvektörlerini hesaplar. A simetrik bant matris, B simetrik bant matris, det A ≠0 ve det B≠0 olmalıdır. A nın ve B nin sadece üst yarı bantı programa verilir.

Çağrılan alt programlar: DSearch, Determinant, Factor, Powerinvers

Örnek: Bir kirişin serbest titreşim problemi[1]

Yarı bant=4

$$\underline{A} = \begin{bmatrix} 242386 & 0 & -121193 & 181790 & & & \\ & 727160 & -181790 & 181790 & 0 & & \\ & & 242386 & 0 & -121193 & 181790 & \\ & & & 727160 & -181790 & 181790 & 0 \\ & \text{Simetrik} & & & 242386 & 0 & 181790 \\ & & & & & 727160 & 181790 \\ & & & & & & 363580 \end{bmatrix}$$

Kirişin rijitlik matrisi

İlk üç özdeğer ve bunlara karşılık gelen açısal frekans, periyot, frakans ve özvektörlerin (modların) hesabı?

Yarı bant=4

$$\underline{B} = \begin{bmatrix} 1.170 & 0 & 0.2025 & -0.14625 & 0 & 0 & 0 \\ & 0.270 & 0.14625 & -0.10125 & 0 & 0 & 0 \\ & & 1.170 & 0 & 0.2025 & -0.14625 & 0 \\ & & & 0.270 & 0.14625 & -0.10125 & 0 \\ & \text{Simetrik} & & & 1.170 & 0 & -0.14625 \\ & & & & & 0.270 & -0.10125 \\ & & & & & & 0.135 \end{bmatrix}$$

Kirişin kütle matrisi

A ve B nin depolanma şekli:

Yarı bant=4 Yarı bant=4

$$\underline{A} \rightarrow \begin{bmatrix} 242386 & 0 & -121193 & 181790 \\ 727160 & -181790 & 181790 & 0 \\ 242386 & 0 & -121193 & 181790 \\ 727160 & -181790 & 181790 & 0 \\ 242386 & 0 & 811790 & 0 \\ 727160 & 181790 & 0 & 0 \\ 363580 & 0 & 0 & 0 \end{bmatrix}, \quad \underline{B} \rightarrow \begin{bmatrix} 1.170 & 0 & 0.2025 & -0.14625 \\ 0.270 & 0.14625 & -0.10125 & 0 \\ 1.170 & 0 & 0.2025 & -0.14625 \\ 0.270 & 0.14625 & -0.10125 & 0 \\ 1.170 & 0 & -0.14625 & 0 \\ 0.270 & -0.10125 & 0 & 0 \\ 0.135 & 0 & 0 & 0 \end{bmatrix}$$

Sabit bant genişliği için eklenen sıfırlar

```
C:\NANALIZ\Basic\QBasic.EXE
DSearch Sonuçları:
özdeğer Lamda 1 = 5961.87971815759
Açısal frekans Omega 1 = 77.2132094797101 rad/s
Periyot T 1 = 8.13744869500687D-02 s
Frekans f 1 = 12.2888639606859 Hz
Mod 1 :
  0.4500  0.2291  1.0000  0.0901  0.8454 -0.1916 -0.3293

özdeğer Lamda 2 = 63314.7496166195
Açısal frekans Omega 2 = 251.624223032322 rad/s
Periyot T 2 = .024970510515486 s
Frekans f 2 = 40.0472388972515 Hz
Mod 2 :
 -0.9748 -0.2620 -0.4131  0.5690  1.0000  0.1198 -0.6016

özdeğer Lamda 3 = 284619.814085994
Açısal frekans Omega 3 = 533.497717039158 rad/s
Periyot T 3 = 1.17773424449695D-02 s
Frekans f 3 = 84.9087987950233 Hz
Mod 3 :
  1.0000 -0.2065 -0.8903 -0.3067  0.5390  0.6645 -0.8008
```

DSearch sonuçları

[1] Teori ve sayısal el hesabı için bak: bölüm 26.

DSearch ana programı

```
'-----------------Ana program DSearch----------------------------------------
' Ahmet TOPÇU, Eskişehir Osmangazi Üniversitesi, 1998
' (A-Lamda*B)x=0 genel özdeğer problemi çözülür
' En küçük m özdeğer ve bunlara ait  özvektörler hesaplanır

' Veri:
' A(n,n): Tekil olmayan simetrik bant matris, sadece üst bant kısmı verilir
' B(n,n): Tekil olmayan simetrik bant matris, sadece üst bant kısmı verilir
' n: A ve B nin boyutu
' iBantA: A nın yarı bant genişliği
' iBantB: B nin yarı bant genişliği
' m: Hesaplanması istenen özdeğer sayısı

' Çıktı:
' ALamda(n,2): özdeğerler
' X(n,m): Özvektörler

' Çağrılan alt programlar: Dsearch, Determinant, Factor, Powerinvers
'----------------------------------------------------------------------------

' Örnek: Betonarme kirişin ilk üç modunun hesabı
DATA 7: ' A=Rijitlik matrisinin denklem sayısı
DATA 4: ' Rijitlik matrisinin yarı bant genişliği
DATA 4: ' B= M Kütle matrisinin bant genişliği
DATA 3: ' hesaplanması istenen mod sayısı

' A=K rijitlik matrisinin üst yarı bantındaki sayılar
DATA 242386,0,-121193,181790
DATA 727160,-181790,181790,0
DATA 242386,0,-121193,181790
DATA 727160,-181790,181790,0
DATA 242386,0,181790,0
DATA 727160,181790,0,0
DATA 363580,0,0,0
```

A nın üstt bantı içindeki elemanları

```
' B=M kütle matrisinin üst yarı bantındaki sayılar
DATA 1.170,0,0.2025,-0.14625
DATA 0.270,0.14625,-0.10125,0
DATA 1.170,0,0.2025,-0.14625
DATA 0.270,0.14625,-0.10125,0
DATA 1.170,0,-0.14625,0
DATA 0.270,-0.10125,0,0
DATA 0.135,0,0,0
```

B nin üst bantı içindeki elemanları

```


    DEFINT I-N
    DEFDBL A-H, O-Z

DECLARE SUB DSearch (n, MaxBant, A(), iBantB, B(), m, aLamda(), X(), nFound)
DECLARE SUB Determinant (n, Z(), aLamda(), u2, nLess, nFound, det, Power)
DECLARE SUB Factor (n, MaxBant, Z(), iTekil, Eps, nFact)
DECLARE SUB Powerinvers (n, Z(), x1(), y(), y1(), y2(), w(), B(), MaxBant, iBantB, u1, u2, u3, iter)

    CLS : ' Ekranı temizle

    READ n:  ' denklem sayısını oku
    READ iBantA: ' K nın yarı bant genişliğini oku
    READ iBantB: ' b nin yarı bant genişliğini oku
    READ m: ' hesaplanması istenen özdeğer sayısını oku
    IF m < 1 THEN m = 1
    IF m > n THEN m = n

' Max bant genişliği
    MaxBant = iBantA
    IF iBantB > MaxBant THEN MaxBant = iBantB

' Setup dimentions of arrays
    DIM A(n, MaxBant) AS DOUBLE
    DIM B(n, MaxBant)
    DIM aLamda(m, 2): ' Özdeğerler
    DIM X(n, m): ' Özvektörler

' A=K nın üst yarı bantındaki sayıları oku:
    FOR i = 1 TO n
     FOR j = 1 TO MaxBant
      A(i, j) = 0
     NEXT j
     FOR j = 1 TO iBantA
      READ A(i, j)
     NEXT j
    NEXT i
```

Devamı sonraki sayfada

```
' B=M nın üst yarı bantındaki sayıları oku:
    FOR i = 1 TO n
     FOR j = 1 TO MaxBant
      B(i, j) = 0
     NEXT j
     FOR j = 1 TO iBantB
      READ B(i, j)
     NEXT j
     NEXT i

    CALL DSearch(n, MaxBant, A(), iBantB, B(), m, aLamda(), X(), nFound)

' Çıktılar
    PRINT "DSearch Sonuçları:"
    IF nFound = 0 THEN
     PRINT "Hiçbir özdeğer bulunamadı!"
    ELSE
     FOR i = 1 TO nFound
      aLamda = aLamda(i, 1)
      Omega = SQR(ABS(aLamda))
      T = 2 * (4 * ATN(1)) / Omega
      F = 1 / T
      PRINT "Özdeğer Lamda"; i; "="; aLamda
      PRINT "Açısal frekans Omega"; i; "="; Omega; "rad/s"
      PRINT "Periyot T"; i; "="; T; "s"
      PRINT "Frekans f"; i; "="; F; "Hz"
      PRINT "Mod"; i; ":"
      FOR j = 1 TO n
       PRINT USING " ##.####"; X(j, aLamda(i, 2));
      NEXT j
      PRINT : PRINT
     NEXT i
     PRINT
    END IF
END ' DSearch sonu
```

DSearch ana devamı

```
SUB Determinant (n, Z(), aLamda(), u2, nLess, nFound, det, Power)
'----------Determinant of the matrix Z----------------------------
   nLess = 0
   det = 1
   Power = 0
   FOR i = 1 TO n
    IF Z(i, 1) < 0 THEN nLess = nLess + 1
    det = det * Z(i, 1)
    IF i <= nFound THEN det = det / (u2 - aLamda(i, 1))
    WHILE ABS(det) > 1
     det = det * .0625
     Power = Power + 4
    WEND
    WHILE (ABS(det) < .0625)
     det = det * 16
     Power = Power - 4
    WEND
   NEXT i

END SUB ' Determinant sonu
```

```
SUB DSearch (n, MaxBant, A(), iBantB, B(), m, aLamda(), X(), nFound)
'-------------------------------------------------------------------------------
' (A-Lamda*B)*X=0 genel özdeğer denkleminden tüm veya en küçük birkaç
' özdeğerini ve bunlara ait özvektörlerini hesaplar.

' Veriler:
' A(n,n): Tekil olmayan simetrik bant matris. Sadece üst bant kısmı verilir
' B(n,n): Tekil olmayan simetrik bant matris. Sadece üst bant kısmı verilir
' n: A ve B nin boyutu
' iBantA: A nın yarı bant genişliği
' iBantB: B nin yarı bant genişliği
' m: Hesaplanması istenen özdeğer sayısı

' Çıktılar:
' ALamda(n,2):
'   X(i,1), i=1,2,..,m nolu Özdeğerler
'   X(i,2): i. özdeğerin X(n,m) vektöründeki özvektörünün kolon numarası
' X(n,m): Özvektörler

' Metod: Search and power invers vector iteration

' Kaynak: "BALFOUR, J., Computer Analysis of Structural Framework
' Oxfort University Press, New York, 1992, 421-429" den alınmış
' iyileştirilmiş ve basitleştirilmiştir.
'-------------------------------------------------------------------------------
    DIM Z(n, MaxBant), x1(n), y(n), y1(n), y2(n), w(n, 6): ' Work arrays
    Eps = .000001: 'öngörülen hassasiyet
    iTrue = -1: iFalse = 0
    nFound = 0
    nCalc = 0
    u1 = 0
    u2 = 0
    nFact = 0
    iter = 0
    WHILE nFound < m

' Get Start values for accelerated secant method
     u1 = 0
     u2 = 0
     FOR i = 1 TO n
      FOR j = 1 TO MaxBant
       Z(i, j) = A(i, j)
      NEXT j
     NEXT i

     CALL Factor(n, MaxBant, Z(), iTekil, Eps, nFact): ' Factor
     CALL Determinant(n, Z(), aLamda(), u2, nLess, nFound, det1, Power1): ' Determinant

     FOR i = 1 TO n
      y1(i) = RND     ' Generate random numbers
      WHILE y1(i) = 0
       y1(i) = RND
      WEND
     NEXT i

' Use power invers iteration to calculate u2
     WHILE iter < 5
      CALL Powerinvers(n, Z(), x1(), y(), y1(), y2(), w(), B(), MaxBant, iBantB, u1, u2, u3, iter)
      u2 = u3
     WEND

' Check that u2 has not jumped any eigenvaues. If so then divide
' u2 by no. nLess+1 and redo until u2 is to left of next eigenvalue
     iDone = iFalse
     WHILE iDone = iFalse
      FOR i = 1 TO n
       FOR j = 1 TO MaxBant
        Z(i, j) = A(i, j) - u2 * B(i, j)
       NEXT j
      NEXT i

      CALL Factor(n, MaxBant, Z(), iTekil, Eps, nFact):  ' Factorise

      IF iTekil = iTrue THEN
       u2 = .9999 * u2
      ELSE

       CALL Determinant(n, Z(), aLamda(), u2, nLess, nFound, det, Power): ' Calc determinant and cont values

       IF nLess = 0 THEN iDone = iTrue ELSE u2 = u2 / (nLess + 1)
      END IF
     WEND
     det2 = det
     Power2 = Power
```

Dsearch devam ediyor

Dsearch devamı

```
' to jump over one or more eigenvalues
    acc = 2: ' acceleration number
    WHILE (nLess <= nFound)
4690  FOR i = 1 TO n: ' Calculate the coefficient matrix
      FOR j = 1 TO MaxBant
       Z(i, j) = A(i, j) - u2 * B(i, j)
      NEXT j
     NEXT i

     CALL Factor(n, MaxBant, Z(), iTekil, Eps, nFact):  ' Factor
     IF iTekil = iTrue THEN u2 = .9999 * u2: GOTO 4690

     CALL Determinant(n, Z(), aLamda(), u2, nLess, nFound, det2, Power2): ' Calc determinant count eigenvalues

' if method has coverged then avoid the risk of division by use
' simple scaling
     IF (ABS((u2 - u1) / u2) > Eps) OR (nLess > nFound) GOTO 4850
     u3 = 1.0001 * u2
     GOTO 4900
4850  T = det2 - det1 * 2 ^ (Power1 - Power2)
     IF T = 0 THEN
      PRINT "Zero division occured, possibly A or B singular!"
      END
     END IF
     u3 = u2 - acc * det2 * (u2 - u1) / T

' if change less then %1 then accelerate the method
     IF ABS((u3 - u2) / u2) < .01 THEN acc = 2 * acc
4900  u1 = u2
     det1 = det2
     Power1 = Power2
     u2 = u3
    WEND
' end of Secant method

    nCalc = nFound
    WHILE (nCalc < nLess)
     FOR i = 1 TO n
      y1(i) = RND
      WHILE y1(i) = 0
       y1(i) = RND: ' generate random numbers
      WEND
     NEXT i
     iDone = iFalse
     WHILE iDone = iFalse

' invers iteration
      CALL Powerinvers(n, Z(), x1(), y(), y1(), y2(), w(), B(), MaxBant, iBantB, u1, u2, u3, iter)
      IF (ABS((u3 - u2) / u2)) < Eps THEN iDone = iTrue
      u2 = u3
     WEND

' This part calculates the eigenvector once eigenvalue
' has been found by inverse iteration
     nFound = nFound + 1
     aLamda(nFound, 1) = u2
     aLamda(nFound, 2) = nFound

' Calculate denominator
     Den = 0
     FOR i = 1 TO n
      Den = Den + x1(i) * y(i)
     NEXT i
     Den = SQR(ABS(Den))
     aMax = 0

' Calc the vector and aMax element
     FOR i = 1 TO n
      X(i, nFound) = x1(i) / Den
      IF ABS(X(i, nFound)) > ABS(aMax) THEN aMax = X(i, nFound)
     NEXT i


' Schmidt orthogonalisation
     K = nFound - 6 * ((nFound - 1) \ 6)
     FOR i = 1 TO n
      w(i, K) = B(i, 1) * X(i, nFound)
      FOR j = 1 TO iBantB - 1
       IF i - j > 0 THEN w(i, K) = w(i, K) + B(i - j, j + 1) * X(i - j, nFound)
       IF i + j <= n THEN w(i, K) = w(i, K) + B(i, j + 1) * X(i + j, nFound)
      NEXT j
     NEXT i
```

Dsearch devam ediyor

Dsearch devamı

```
' Normalise
      FOR i = 1 TO n
       X(i, nFound) = X(i, nFound) / aMax
      NEXT i

' end of find eigenwector
      IF u2 <= u1 THEN nCalc = nCalc + 1
      nFact = 0
      iter = 0
     WEND
    WEND

' Sorting eigenvalues
    iDone = iFalse
    WHILE iDone = iFalse
     iDone = iTrue
     FOR K = 1 TO nFound - 1
      IF aLamda(K, 1) > aLamda(K + 1, 1) THEN
       iDone = iFalse
       T = aLamda(K, 1)
       aLamda(K, 1) = aLamda(K + 1, 1)
       aLamda(K + 1, 1) = T
       T = aLamda(K, 2)
       aLamda(K, 2) = aLamda(K + 1, 2)
       aLamda(K + 1, 2) = T
      END IF
     NEXT K
    WEND

END SUB ' Dsearch sonu
```

```
SUB Factor (n, MaxBant, Z(), iTekil, Eps, nFact)
'This subroutine decomposes symmetrik, positive definit, banded
'matrix Z in the form U(Transpoz) D U
'-------------------------------------------------------------------
    iTekil = 0: ' Nonsingular
    FOR i = 1 TO n
     IF i < MaxBant THEN iFin = i - 1 ELSE iFin = MaxBant - 1
     FOR K = 1 TO iFin
      Z(i, 1) = Z(i, 1) - Z(i - K, K + 1) * Z(i - K, K + 1) * Z(i - K, 1)
      IF ABS(Z(i, 1)) < Eps THEN
       iTekil = 1: ' Matrix Singular
       EXIT SUB
      END IF
     NEXT K

     FOR j = 2 TO MaxBant
      IF i < MaxBant THEN iFin = i - 1 ELSE iFin = MaxBant - j
      FOR K = 1 TO iFin
       IF j + K > MaxBant THEN EXIT FOR
        Z(i, j) = Z(i, j) - Z(i - K, K + 1) * Z(i - K, j + K) * Z(i - K, 1)
      NEXT K
      Z(i, j) = Z(i, j) / Z(i, 1)
     NEXT j
    NEXT i
    nFact = nFact + 1

END SUB ' Factor sonu
```

```
SUB Powerinvers (n, Z(), x1(), y(), y1(), y2(), w(), B(), MaxBant, iBantB, u1, u2, u3, iter)
' This subroutine performs one cycle of vector inverse iteration and vector orthogonalises
' the trial vector to the last six eigenvalues to allow for multiple roots
'-------------------------------------------------------------------------------------------
    FOR i = 1 TO n
     y(i) = y1(i)
    NEXT i

' This part solves a system of linear equation where
' Coefficient matrix has been decomposed using subrotine Factor
' Forward subtitution
    FOR i = 1 TO n
     IF i > MaxBant - 1 THEN iFin = MaxBant - 1 ELSE iFin = i - 1
     S = 0
     IF i <> 1 THEN
      FOR j = 1 TO iFin
       S = S + Z(i - j, j + 1) * Z(i - j, 1) * x1(i - j)
      NEXT j
     END IF
     x1(i) = (y(i) - S) / Z(i, 1)
    NEXT i

' Backward subtitution
    FOR i = 1 TO n - 1
     IF i > MaxBant - 1 THEN iFin = MaxBant - 1 ELSE iFin = i
     FOR j = 1 TO iFin
      x1(n - i) = x1(n - i) - Z(n - i, j + 1) * x1(n - i + j)
     NEXT j
    NEXT i

' Calculate the weighted vector
    FOR i = 1 TO n
     y(i) = B(i, 1) * x1(i)
     FOR j = 1 TO iBantB - 1
      IF i - j > 0 THEN y(i) = y(i) + B(i - j, j + 1) * x1(i - j)
      IF i + j <= n THEN y(i) = y(i) + B(i, j + 1) * x1(i + j)
     NEXT j
    NEXT i

' Calculate new trial vector based upon the weighted vector
    FOR i = 1 TO n
     y2(i) = y(i)
    NEXT i
    IF (nFound < 6) THEN iFin = nFound ELSE iFin = 6
```

```
' Orthogonalise for last six vector,
'-----------------------------------

    FOR j = 1 TO iFin
     Alfa = 0
     FOR i = 1 TO n
      Alfa = Alfa + x1(i) * w(i, j)
     NEXT i
     FOR i = 1 TO n
      y2(i) = y2(i) - Alfa * w(i, j)
     NEXT i
    NEXT j

' Denominater
    Den = 0
    FOR i = 1 TO n
     Den = Den + x1(i) * y(i)
    NEXT i

' New trial vector
    Den = SQR(ABS(Den))
    FOR i = 1 TO n
     y2(i) = y2(i) / Den
    NEXT i
    Anum = 0

' New estimated aLamda (u3)
    Den = Den * Den
    FOR i = 1 TO n
     Anum = Anum + x1(i) * y1(i)
     y1(i) = y2(i)
    NEXT i
    u3 = u1 + Anum / Den
    iter = iter + 1

END SUB ' Powerinvers sonu
```