СООБЩЕНИЯ
ОБЪЕДИНЕННОГО
ИНСТИТУТА
ЯДЕРНЫХ
ИССЛЕДОВАНИЙ
ДУБНА

Р11 - 9677

В.Ц.Банчев

ПРОГРАММА ЧИСЛЕННОГО ИНТЕГРИРОВАНИЯ ЗАДАЧИ КОШИ ДЛЯ ЖЕСТКИХ СИСТЕМ ОБЫКНОВЕННЫХ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ

1976

Р11 - 9677

В.Ц.Банчев

ПРОГРАММА ЧИСЛЕННОГО ИНТЕГРИРОВАНИЯ ЗАДАЧИ КОШИ ДЛЯ ЖЕСТКИХ СИСТЕМ ОБЫКНОВЕННЫХ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ

Банчев В.Ц.

Р11 - 9677

Программа численного интегрирования задачи Коши для жестких систем обыкновенных дифференциальных уравнений

Приводится написанная на языке ФОРТРАН программа численного интегрирования задачи Коши для жестких систем обыкновенных дифференциальных уравнений. В ней реализован обобщенный метод Рунге-Кутта, предложенный Лоусоном, в котором матричная экспонента аппроксимируется с помощью разложения показательной функции в непрерывную дробь. Программа оценивает локальную ошибку усечения и использует полученную оценку для автоматического выбора величины шага и порядка аппроксимации матричной экспоненты.

Работа выполнена в Лаборатории вычислительной техники и автоматизации ОИЯИ.

Сообщение Объединенного института ядерных исследований

Дубна 1976

Banchev V.Ts.

P11 - 9677

A Subroutine for the Numerical Integration of the Cauchy Problem for Stiff Systems of Ordinary Differential Equations

A Fortran subroutine for the numerical integration of the Cauchy problem for stiff systems of ordinary differential equations is given. It implements the generalized Runge-Kutta method, suggested by Lawson, with a continued fraction approximation for the matrix exponential. The subroutine estimates the local truncation error and uses it for the automatic choice of step size and order of matrix exponential approximant.

The investigation has been performed at the Laboratory of Computing Techniques and Automation, JINR.

Communication of the Joint Institute for Nuclear Research

Dubna 1976

В программе GERUN реализован А-устойчивый алгоритм решения задачи Коши для жестких нормальных систем обыкновенных дифференциальных уравнений первого порядка в общем случае, когда большие собственные значения матрицы Якоби могут иметь не только большие отрицательные вещественные части [1]. Программа написана на языке ФОРТРАН, и ее можно непосредственно применять на ЭВМ СДС серии 6000. Она использует программу REGN для решения систем уравнений на основе регуляризованных итерационных процессов Гаусса-Ньютона [2].

В основу реализованного в GERUN алгоритма положен обобщенный метод Рунге-Кутта, предложенный Лоусоном [3], в котором матричная экспонента аппроксимируется с помощью разложения показательной функции в непрерывную дробь [4]. Алгоритм позволяет получить оценку локальной ошибки усечения, которая используется в программе для автоматического выбора величины шага и порядка аппроксимации матричной экспоненты. Для ознакомления с алгоритмом программы GERUN мы отсылаем читателя к [1], где даны и результаты некоторых численных экспериментов. Здесь приводится только описание программы.

Описание программы

Программа GERUN приводится в приложении. Она находит численное решение задачи Коши для нормальной системы обыкновенных дифференциальных уравнений

$$y' = f(t,y), \quad y(t_0) = y_0, \qquad (I)$$

где y и f N-мерные векторы, по формулам (4), (5), (6) в [1].

GERUN использует 19 подпрограмм. Подпрограмма EXCON вычисляет аппроксимацию матричной экспоненты, используя для этой цели подпрограмму INVR, которая с помощью REGN обращает матрицы F_{2m+1} в (5) из [1]. Заметим, что в случае жестких систем матрицы F_{2m+1} обычно являются плохо обусловленными [4] и поэтому использование программы REGN является целесообразным [2]. Выдача информации осуществляется в OUTP. Подпрограммы FMMX и FMVX с помощью VIPD вычисляют произведение двух матриц и произведение вектора на матрицу.

Формальные параметры GERUN и элементы ее общих блоков GOV1, GOV2, GOV3, GOV4 будем называть в дальнейшем параметрами управления. С их помощью можно реализовать различные модификации алгоритма GERUN. Для упрощения при задании модификаций программы GERUN предусмотрено самоуправление некоторых параметров управления, т.е. автоматическое присваивание стандартных значений (см. приложение). Индикацией для включения режима самоуправления служит нулевое значение этих параметров (см. также [2, с. 6]).

Различные модификации алгоритма GERUN позволяют: а) находить численное решение задачи Коши (I) обобщенным методом Рунге-Кутта (4), (5), (6) из [1] либо без вычисления оценки локальной ошибки усечения, либо с оценкой локальной ошибки усечения методом вложения Сарафаяна (без учета или с учетом ошибки аппроксимации матричной экспоненты) и с автоматическим выбором шага и порядка аппроксимации матричной экспоненты [1]; б) решать (I) в случае нежестких систем, используя только явный метод Рунге-Кутта пятого порядка (6) в [1], без оценки или с оценкой локальной ошибки усечения методом вложения Сарафаяна и с автоматическим выбором шага.

Ниже приводится описание параметров управления GERUN.

N Число уравнений в (I).

H Величина шага численного интегрирования.

X Переменная, принимающая значения t_n в (4) из [1].

Y Одномерный массив размерности N, содержащий значения искомых функций y_n^i $(i=1,2,\ldots,N)$ в точке t_n.

P, S1, S2, S3, S4, S5, S6

Одномерные рабочие массивы размерности N.

NC NC=N для тех модификаций алгоритма, которые используют обобщенный метод Рунге-Кутта. В остальных случаях для уменьшения необходимой памяти можно положить NC=1.

CC, PC, F

Одномерные рабочие массивы размерности NC.

A, A2, F1, F2, G1, G2, FINV, EXC, R

Двумерные рабочие массивы размерности (NC, NC).

FUN Подпрограмма, составляемая пользователем, для вычисления правых частей $f^i(t,y)$ $(i=1,2,\ldots,N)$ в (I). Способ составления следующий:

```
SUBROUTINE FUN (X, Y, F, N)
DIMENSION Y(N), F(N)
```

где параметры X, Y задают значения t, y, а F содержит вычисленные значения $f^i(t,y)$. FUN должна быть объявлена EXTERNAL в программе, вызывающей GERUN.

AMAT Подпрограмма, составляемая пользователем, с помощью которой задается матрица A в обобщенном методе Рунге-Кутта.

```
SUBROUTINE AMAT (X, Y, F, A, N)
DIMENSION Y(N), F(N), A(N,N)
```

В качестве A рекомендуется брать матрицу Якоби $\partial f/\partial y$ системы (I), вычисляемую для t и y, задаваемых с помощью X и Y. Рабочий массив F может быть использован для промежуточных вычислений в AMAT, например, если $\partial f/\partial y$ находится численным дифференцированием. AMAT должна быть объявлена EXTERNAL в программе, вызывающей GERUN.

MST Максимальное число шагов.

XEND Абсолютная величина интервала, в котором необходимо найти решение задачи Коши (I).

KPR Число шагов, через которое будет печататься решение (I).

KPD При KPD≠0 печатаются оценки локальных ошибок усечения для каждой компоненты решения.

EMAX Задает верхнюю границу для локальной ошибки усечения. Если EMAX≤0, оценка локальной ошибки усечения не вычисляется, что приводит к уменьшению времени счета.

EMIN Нижняя граница для локальной ошибки усечения.

HMAX Максимальная абсолютная величина шага.

HMIN Минимальная абсолютная величина шага.

DB Если найденная оценка локальной ошибки усечения больше EMAX, шаг уменьшается с помощью соотношения H=H*DB(DB<1). В программе DB=0,5.

DF Если найденная оценка локальной ошибки усечения меньше EMIN, шаг увеличивается с помощью соотношения H=H * DF(DF>1). В программе DF=2.

KMAX Максимальный порядок аппроксимации матричной экспоненты.

KIN Содержит начальное значение порядка аппроксимации матричной экспоненты (0≤KIN≤KMAX). При KIN=0 для решения (I) используется явный метод Рунге-Кутта пятого порядка (6) в[I]. Если EMAX≤0, порядок аппроксимации матричной экспоненты во время счета остается равным своему начальному значению. При KIN=KMAX ошибка аппроксимации матричной экспоненты (второй член в правой части неравенства (9) в[I]) не учитывается при оценке локальной ошибки усечения.

KEX Число шагов, через которое вычисляется матрица A в обобщенном методе Рунге-Кутта.

ES Масштабный множитель для оценки локальной ошибки усечения, обозначенной EPS в программе (EPS=ES * EPS) . Стандартное значение ES=1.

MRED В случае плохой обусловленности матриц F_{2m+1} в (5) из[I] рекомендуется увеличить значение MRED (MRED = = 0, I, 2, 3, ...). Это позволяет также использовать меньшее значение параметра KMAX[3,4].

Приведем и один из параметров управления программы REGN, который может понадобиться в случае, если подпрограмма INVR не обращает матрицы F_{2m+1} в (5) из[I] (при заданном MRED) с достаточной точностью из-за их плохой обусловленности. Это параметр C(3) в общем блоке CINT подпрограмм REGN и INVR , где положено C(3)=0,0000I. Рекомендуется в таком случае увеличить значение C(3) (подробнее см.[2]). Заметим, что точность обращения матриц F_{2m+1} оценивается следующим образом в INVR . Пусть

$$p_{ik} = \sum_{l=1}^{N} \left(F^{-1}_{2m+1}\right)_{il} \left(F_{2m+1}\right)_{lk} - \delta_{ik},$$

где δ_{ik} - символ Кроннекера. Если $\max_{i,k} |p_{ik}| \leq 10^{-5}$, считается, что точность обращения удовлетворительная.

ПРИЛОЖЕНИЕ

```
      SUBROUTINE GERUN(N,H,X,Y,P,S1,S2,S3,S4,S5,S6,NC,CC,PC,F,A,A2,
     *F1,F2,G1,G2,FINV,EXC,R,FUN,AMAT)
      DIMENSION Y(N),P(N),S1(N),S2(N),S3(N),S4(N),S5(N),S6(N)
      DIMENSION CC(NC),PC(NC),F(NC),A(NC,NC),A2(NC,NC)
      DIMENSION F1(NC,NC),F2(NC,NC),G1(NC,NC),G2(NC,NC)
      DIMENSION FINV(NC,NC),EXC(NC,NC),R(NC,NC)
      COMMON/GOV1/MST,XEND,KPR,KPO
      COMMON/GOV2/EMAX,EMIN,HMAX,HMIN,DB,DF
      COMMON/GOV3/KMAX,KIN,KEX
      COMMON/GOV4/ES,MRED
      IF(MST.LE.0) MST=1
      IF(XEND.LE.0.) XEND=ABS(MST*H)
      IF(HMAX.LE.0.) HMAX=ABS(H)
      IF(KPR.LE.0) KPR=1
      IF(EMAX.LE.0.) EMAX=0.
      IF(EMAX.EQ.0.) KPO=0
      IF(EMAX.EQ.0.) EPS=0.
      IF(EMAX.EQ.0.) EMIN=0.
      IF(KIN.LE.0) KIN=0
      IF(KIN.GT.KMAX) KMAX=KIN
      IF(KMAX.GT.KIN.AND.EMAX.GT.0.) KMAX=KMAX+1
      IF(KEX.LE.0) KEX=1
      IF(DB.LE.0.) DB=0.5
      IF(DF.LE.0.) DF=2.
      IF(ES.LE.0.) ES=1.
      IF(MRED.LT.0) MRED=0
      IF(ABS(H).LT.HMIN) GO TO 13
      XIN=X
      KNC=KIN
C
      DO 15 IS=1,MST
      IF(KIN.GT.0) GO TO 2
C
    1 CALL FUN(X,Y,S1,N)
C
      CALL VLINE(Y,1.,S1,H/4,P,N)
      CALL FUN(X+H/4,P,S2,N)
C
      CALL VLINE(S1,H/8,S2,H/8,P,N)
      CALL VADD(Y,P,P,N)
      CALL FUN(X+H/4,P,S3,N)
C
      CALL VLINE(S2,-H/2,S3,H,P,N)
      CALL VADD(Y,P,P,N)
      CALL FUN(X+H/2,P,S4,N)
C
      CALL VLINE(S1,3*H/16,S4,9*H/16,P,N)
      CALL VADD(Y,P,P,N)
      CALL FUN(X+3*H/4,P,S5,N)
C
      CALL VLINE(S1,-3*H/7,S2,2*H/7,P,N)
      CALL VLINE(P,1.,S3,12*H/7,P,N)
      CALL VLINE(P,1.,S4,-12*H/7,P,N)
      CALL VLINE(P,1.,S5,8*H/7,P,N)
```

```
      CALL VADD(Y,P,P,N)
      CALL FUN(X+H,P,S6,N)
C
      CALL VLINE(S1,7*H/90,S3,32*H/90,P,N)
      CALL VLINE(P,1.,S4,12*H/90,P,N)
      CALL VLINE(P,1.,S5,32*H/90,P,N)
      CALL VLINE(P,1.,S6,7*H/90,P,N)
      CALL VADD(Y,P,S6,N)
C
      IF(EMAX.EQ.0.) GO TO 12
C
      CALL VADD(S1,S4,P,N)
      CALL VLINE(P,H/12,S3,H/3,P,N)
      CALL VADD(Y,P,S5,N)
C
      X1=X+H/2
C
      CALL FUN(X1,S5,S1,N)
C
      CALL VLINE(S5,1.,S1,H/4,P,N)
      CALL FUN(X1+H/4,P,S2,N)
C
      CALL VLINE(S1,H/8,S2,H/8,P,N)
      CALL VADD(S5,P,P,N)
      CALL FUN(X1+H/4,P,S3,N)
C
      CALL VLINE(S2,-H/2,S3,H,P,N)
      CALL VADD(S5,P,P,N)
      CALL FUN(X1+H/2,P,S4,N)
C
      CALL VADD(S1,S4,P,N)
      CALL VLINE(P,H/12,S3,H/3,P,N)
      CALL VADD(S5,P,S5,N)
C
      GO TO 7
C
    2 KW1=MOD(IS,KEX)
      IF(IS.EQ.1.OR.KW1.EQ.0) GO TO 3
      IF(KIS.EQ.1) GO TO 201
      GO TO 6
    3 CALL AMAT(X,Y,F,A,NC)
  201 HRED=H/(4*2**MRED)
      CALL FMC(A2,A,HRED,NC)
      MC=0
    4 CALL EXCON(NC,A2,EXC,F1,F2,G1,G2,FINV,R,P,F,MC)
      IF(MC.LT.0) GO TO 203
      IF(MRED.EQ.0) GO TO 6
      CALL FMEQ(FINV,EXC,NC)
      DO 202 IR=1,MRED
      CALL FMMX(EXC,FINV,R,NC,NC,NC,NC,NC,NC)
      CALL FMEQ(EXC,R,NC)
      CALL FMEQ(FINV,R,NC)
  202 CONTINUE
      GO TO 6
C
  203 PRINT 5
```



```
    5 FORMAT(//,2X,35HPADE APPROXIMANT CANNOT BE OBTAINED,
     *1X,24HWITH SUFFICIENT ACCURACY)
      RETURN
C
    6 CALL FUN(X,Y,S1,NC)
      CALL FMC(R,A,-1.,NC)
      CALL FMVX(R,Y,P,NC,NC,NC)
      CALL VADD(S1,P,S1,NC)
C
      CALL VLINE(Y,1.,S1,H/4,F,NC)
      CALL FMVX(EXC,F,P,NC,NC,NC)
      CALL FMVX(R,P,F,NC,NC,NC)
      CALL FUN(X+H/4,P,S2,NC)
      CALL VADD(S2,F,S2,NC)
C
      CALL VLINE(Y,1.,S1,H/8,P,NC)
      CALL FMVX(EXC,P,F,NC,NC,NC)
      CALL VLINE(F,1.,S2,H/8,P,NC)
      CALL FMVX(R,P,F,NC,NC,NC)
      CALL FUN(X+H/4,P,S3,NC)
      CALL VADD(S3,F,S3,NC)
C
      CALL FMVX(EXC,Y,P,NC,NC,NC)
      CALL VLINE(S2,-H/2,S3,H,F,NC)
      CALL VADD(F,P,F,NC)
      CALL FMVX(EXC,F,P,NC,NC,NC)
      CALL FMVX(R,P,F,NC,NC,NC)
      CALL FUN(X+H/2,P,S4,NC)
      CALL VADD(S4,F,S4,NC)
C
      CALL VLINE(Y,1.,S1,3*H/16,P,NC)
      CALL FMVX(EXC,P,F,NC,NC,NC)
      CALL FMVX(EXC,F,P,NC,NC,NC)
      CALL VLINE(P,1.,S4,9*H/16,F,NC)
      CALL FMVX(EXC,F,P,NC,NC,NC)
      CALL FMVX(R,P,F,NC,NC,NC)
      CALL FUN(X+3*H/4,P,S5,NC)
      CALL VADD(S5,F,S5,NC)
C
      CALL VLINE(Y,1.,S1,-3*H/7,P,NC)
      CALL FMVX(EXC,P,F,NC,NC,NC)
      CALL VLINE(S2,2*H/7,S3,12*H/7,P,NC)
      CALL VADD(F,P,F,NC)
      CALL FMVX(EXC,F,P,NC,NC,NC)
      CALL FMVX(EXC,P,F,NC,NC,NC)
      CALL VLINE(F,1.,S5,8*H/7,S6,NC)
      CALL FMVX(EXC,S4,P,NC,NC,NC)
      CALL VLINE(P,-12*H/7,S6,1.,F,NC)
      CALL FMVX(EXC,F,P,NC,NC,NC)
      CALL FMVX(R,P,F,NC,NC,NC)
      CALL FUN(X+H,P,S6,NC)
      CALL VADD(S6,F,S6,NC)
C
      CALL FMVX(EXC,S4,P,NC,NC,NC)
      CALL VLINE(P,12*H/90,S5,32*H/90,S5,NC)
      CALL VLINE(Y,1.,S1,7*H/90,P,NC)
      CALL FMVX(EXC,P,F,NC,NC,NC)
      CALL VLINE(F,1.,S3,32*H/90,P,NC)
```

```
      CALL FMVX(EXC,P,F,NC,NC,NC)
      CALL FMVX(EXC,F,P,NC,NC,NC)
      CALL VADD(S5,P,F,NC)
      CALL FMVX(EXC,F,P,NC,NC,NC)
      CALL VLINE(P,1.,S6,7*H/90,S6,NC)
C
      IF(EMAX.EQ.0.) GO TO 12
C
      CALL VLINE(Y,1.,S1,H/12,P,NC)
      CALL FMVX(EXC,P,F,NC,NC,NC)
      CALL VLINE(F,1.,S3,H/3,F,NC)
      CALL FMVX(EXC,F,P,NC,NC,NC)
      CALL VLINE(S4,H/12,P,1.,S5,NC)
C
      X1=X+H/2
C
      CALL FUN(X1,S5,S1,NC)
      CALL FMVX(R,S5,F,NC,NC,NC)
      CALL VADD(S1,F,S1,NC)
C
      CALL VLINE(S5,1.,S1,H/4,F,NC)
      CALL FMVX(EXC,F,P,NC,NC,NC)
      CALL FMVX(R,P,F,NC,NC,NC)
      CALL FUN(X1+H/4,P,S2,NC)
      CALL VADD(S2,F,S2,NC)
C
      CALL VLINE(S5,1.,S1,H/8,F,NC)
      CALL FMVX(EXC,P,F,NC,NC,NC)
      CALL VLINE(F,1.,S2,H/8,P,NC)
      CALL FMVX(R,P,F,NC,NC,NC)
      CALL FUN(X1+H/4,P,S3,NC)
      CALL VADD(S3,F,S3,NC)
C
      CALL FMVX(EXC,S5,F,NC,NC,NC)
      CALL VLINE(S2,-H/2,S3,H,P,NC)
      CALL VADD(F,P,F,NC)
      CALL FMVX(EXC,F,P,NC,NC,NC)
      CALL FMVX(R,P,F,NC,NC,NC)
      CALL FUN(X1+H/2,P,S4,NC)
      CALL VADD(S4,F,S4,NC)
C
      CALL VLINE(S5,1.,S1,H/12,P,NC)
      CALL FMVX(EXC,P,F,NC,NC,NC)
      CALL VLINE(F,1.,S3,H/3,F,NC)
      CALL FMVX(EXC,F,P,NC,NC,NC)
      CALL VLINE(P,1.,S4,H/12,S5,NC)
C
    7 CALL VSUB(S6,S5,P,N)
      EPS=VMAXA(P,N)
      CALL VMOD(P,P,N)
      IF(KMAX.EQ.KNC) GO TO 101
      IF(KIN.EQ.KNC) GO TO 10
      CALL VSUB(S6,CC,F,NC)
      CALL VMOD(F,F,NC)
      CALL VADD(F,PC,F,NC)
      EPS=VMAXA(F,NC)
  101 EPS=ES*EPS
      IF(EPS.GT.EMAX) GO TO 8
```



```
      GO TO 12
    8 IF(KMAX.GT.0) GO TO 10
    9 H=H*DB
      KIN=KNC
      IF(ABS(H).LE.HMIN) GO TO 13
      IF(KIN.GT.0) GO TO 201
      GO TO 1
   10 KS=KMAX-KIN
      IF(KS.LE.0) GO TO 9
      CALL VEQ(S6,CC,NC)
      CALL VEQ(P,PC,NC)
      KIN=KIN+1
      IF(KIN.EQ.1) GO TO 2
      MC=1
      GO TO 4
   12 X=X+H
      CALL VEQ(S6,Y,N)
      IF(KMAX.EQ.KNC.OR.EMAX.EQ.0.) GO TO 102
      CALL VEQ(F,P,NC)
      CALL VEQ(CC,Y,NC)
      KIN=KIN-1
  102 KWP=MOD(IS,KPR)
      IF(KWP.EQ.0) CALL OUTP(N,X,Y,EPS,KIN,0)
      IF(KWP.EQ.0.AND.KPD.NE.0) CALL OUTP(N,X,P,EPS,KIN,1)
      KIN=KNC
      IF(ABS(X-XIN).GE.XEND) GO TO 104
      IF(ABS(H).GE.HMAX) GO TO 103
      IF(EPS.GE.EMIN) GO TO 103
      H=H*DF
      KIS=1
      GO TO 15
  103 KIS=0
      IF(KMAX.GT.KIN.AND.EMAX.GT.0.) KIS=1
      GO TO 15
C
   13 PRINT 14
   14 FORMAT(//,2X,19HSTEP SIZE TOO SMALL.//)
      GO TO 104
C
   15 CONTINUE
  104 IF(KMAX.GT.KIN.AND.EMAX.GT.0.) KMAX=KMAX-1
      RETURN
      END


      SUBROUTINE EXCON(N,A,EXC,F1,F2,G1,G2,FINV,R,P,F,MC)
      DIMENSION P(N),F(N)
      DIMENSION A(N,N),EXC(N,N),FINV(N,N),R(N,N)
      DIMENSION F1(N,N),F2(N,N),G1(N,N),G2(N,N)
      COMMON/GOV3/KMAX,KIN,KEX
      IF(MC.GT.0) GO TO 8
      DO 2 I=1,N
      DO 1 J=1,N
      F1(I,J)=0.
    1 G1(I,J)=0.
      F1(I,I)=1.
      G1(I,I)=1.
    2 CONTINUE
```

```
      CALL FMEQ(G2,A,N)
      CALL FMC(F2,A,-1.,N)
      DO 3 I=1,N
      F2(I,I)=F2(I,I)+2.
    3 G2(I,I)=G2(I,I)+2.
      IF(KIN.GT.1) GO TO 5
      CALL INVR(F2,N,N,P,F,R,FINV,D1)
      IF(D1.NE.0.) GO TO 4
      MC=-1
      RETURN
    4 CALL FMMX(FINV,G2,EXC,N,N,N,N,N,N)
      RETURN
    5 CALL FMMX(A,A,EXC,N,N,N,N,N,N)
      CALL FMEQ(A,EXC,N)
      DO 6 J=2,KIN
      CJ=2*(2*J-1)
      CALL FMMX(A,F1,R,N,N,N,N,N,N)
      CALL FMC(EXC,F2,CJ,N)
      CALL FMEQ(F1,F2,N)
      CALL FMMA(F2,EXC,R,N)
      CALL FMMX(A,G1,R,N,N,N,N,N,N)
      CALL FMC(EXC,G2,CJ,N)
      CALL FMEQ(G1,G2,N)
      CALL FMMA(G2,EXC,R,N)
      SCA=SMAXA(F2,P,N,N)
      SCA=1./SCA
      CALL FMC(F1,F1,SCA,N)
      CALL FMC(F2,F2,SCA,N)
      CALL FMC(G1,G1,SCA,N)
      CALL FMC(G2,G2,SCA,N)
    6 CONTINUE
      CALL INVR(F2,N,N,P,F,R,FINV,D1)
      IF(D1.NE.0.) GO TO 7
      MC=-1
      RETURN
    7 CALL FMMX(FINV,G2,EXC,N,N,N,N,N,N)
      RETURN
    8 IF(KIN.GT.2) GO TO 9
      CALL FMMX(A,A,EXC,N,N,N,N,N,N)
      CALL FMEQ(A,EXC,N)
    9 CJ=2*(2*KIN-1)
      CALL FMMX(A,F1,R,N,N,N,N,N,N)
      CALL FMC(EXC,F2,CJ,N)
      CALL FMEQ(F1,F2,N)
      CALL FMMA(F2,EXC,R,N)
      CALL FMMX(A,G1,R,N,N,N,N,N,N)
      CALL FMC(EXC,G2,CJ,N)
      CALL FMEQ(G1,G2,N)
      CALL FMMA(G2,EXC,R,N)
      SCA=SMAXA(F2,P,N,N)
      SCA=1./SCA
      CALL FMC(F1,F1,SCA,N)
      CALL FMC(F2,F2,SCA,N)
      CALL FMC(G1,G1,SCA,N)
      CALL FMC(G2,G2,SCA,N)
      CALL INVR(F2,N,N,P,F,R,FINV,D1)
      IF(D1.NE.0.) GO TO 10
      MC=-1
```



```
      RETURN
   10 CALL FMMX(FINV,G2,EXC,N,N,N,N,N,N)
      RETURN
      END


      SUBROUTINE INVR(ZL,M,N,XX,YR,Z,FINV,D1)
      DIMENSION XX(N),YR(M)
      DIMENSION Z(N,N),ZL(M,N),FINV(N,M)
      COMMON/CRITW/ROMW,RMW,REVW,CONDW,EPSW,TAUW
      COMMON/KSEL/KPI,KPF,KGS,L
      COMMON/CINT/C(15)
      COMMON/ITKP/ITL,KPL
      DATA EPS/1.E-05/
      IF(KPI.LE.0) KPI=0
      IF(KPF.LE.0) KPF=0
      IF(KGS.LE.0) KGS=0
      IF(KPL.GT.0) KP=KPL
      IF(L.LE.0) L=2
      IF(C(3).LE.0.) C(3)=0.00001
      IT=ITL
      IF(ITL.LE.0) IT=100
      A1=1.
      A2=1.
      K=4
      KK=1
      NP=-1
      NN=M
      AD=0.
      S=0.
      T=-1.E-18
      E=-1.
      D1=1.
      DO 5 I=1,M
      DO 1 J=1,N
    1 XX(J)=0.
      DO 2 J=1,M
    2 YR(J)=0.
      YR(I)=1.
      CALL REGN(NP,M,N,NN,K,KK,IT,KP,T,AD,S,TT,D,E,Z,ZL,YR,XX,A1,A2)
      IF((EPS-RMW).GT.0.) GO TO 3
      D1=0.
      RETURN
    3 DO 4 J=1,N
    4 FINV(J,I)=XX(J)
    5 CONTINUE
      RETURN
      END


      SUBROUTINE OUTP(N,X,Y,EPS,KIN,KDP)
      DIMENSION Y(N)
      IF(KDP.NE.0) GO TO 11
      PRINT 1,X
    1 FORMAT(/,5X,2HX=,E24.14)
      PRINT 2,EPS
    2 FORMAT(5X,21HESTIMATED LOCAL ERROR,E12.3)
      PRINT 3,KIN
    3 FORMAT(5X,31HORDER OF EXPONENT APPROXIMATION,I5)
```

```
      I1=N/3
      K=MOD(N,3)+1
      IF(I1-1)7,4,4
    4 DO 5 I=1,I1
      J1=(I-1)*3+1
      J2=J1+1
      J3=J2+1
    5 PRINT 6,J1,Y(J1),J2,Y(J2),J3,Y(J3)
    6 FORMAT(3(1X,2HY(,I4,2H)=,E22.14))
    7 J1=N-K+2
      J2=J1+1
      GO TO (10,8,9)K
    8 PRINT 20,J1,Y(J1)
      GO TO 10
    9 PRINT 21,J1,Y(J1),J2,Y(J2)
   10 RETURN
   11 PRINT 12
   12 FORMAT(5X,17HESTIMATED DEFECTS)
      I1=N/3
      K=MOD(N,3)+1
      IF(I1-1)16,13,13
   13 DO 14 I=1,I1
      J1=(I-1)*3+1
      J2=J1+1
      J3=J2+1
   14 PRINT 15,J1,Y(J1),J2,Y(J2),J3,Y(J3)
   15 FORMAT(3(1X,2HD(,I4,2H)=,E22.14))
   16 J1=N-K+2
      J2=J1+1
      GO TO (19,17,18)K
   17 PRINT 22,J1,Y(J1)
      GO TO 19
   18 PRINT 23,J1,Y(J1),J2,Y(J2)
   19 RETURN
   20 FORMAT(1X,2HY(,I4,2H)=,E22.14)
   21 FORMAT(2(1X,2HY(,I4,2H)=,E22.14))
   22 FORMAT(1X,2HD(,I4,2H)=,E22.14)
   23 FORMAT(2(1X,2HD(,I4,2H)=,E22.14))
      END
```

```
      SUBROUTINE VIPD(A,NA,B,NB,N,C)
      DIMENSION A(1),B(1)
      DOUBLE PRECISION SUM,DA,DB
      SUM=0.
      NN=1-NB
      JL=NA*N
      DO 1 JJ=1,JL,NA
      NN=NN+NB
      DA=A(JJ)
      DB=B(NN)
    1 SUM=SUM+DA*DB
      C=SUM
      RETURN
      END
```



```
      SUBROUTINE FMMX(A,B,C,M,N,MA,MB,MC,K)
      DIMENSION A(MA,1),B(MB,1),C(MC,1)
      DO 1 I=1,M
      DO 1 J=1,K
    1 CALL VIPD(A(I,1),MA,B(1,J),1,N,C(I,J))
      RETURN
      END
```

```
      SUBROUTINE FMVX(A,X,Y,M,N,MA)
      DIMENSION A(1),X(1),Y(1)
      DO 1 I=1,M
    1 CALL VIPD(A(I),MA,X,1,N,Y(I))
      RETURN
      END
```

```
      SUBROUTINE FMC(A,B,C,N)
      DIMENSION A(N,N),B(N,N)
      DO 1 I=1,N
      DO 1 J=1,N
    1 A(I,J)=B(I,J)*C
      RETURN
      END
```

```
      SUBROUTINE FMMA(A,B,C,N)
      DIMENSION A(N,N),B(N,N),C(N,N)
      DO 1 I=1,N
      DO 1 J=1,N
    1 A(I,J)=B(I,J)+C(I,J)
      RETURN
      END
```

```
      SUBROUTINE FMEQ(A,B,N)
      DIMENSION A(N,N),B(N,N)
      DO 1 I=1,N
      DO 1 J=1,N
    1 A(I,J)=B(I,J)
      RETURN
      END
```

```
      FUNCTION SMAXA(A,P,M,N)
      DIMENSION A(M,N),P(N)
      AMAX=0.
      DO 2 I=1,M
      DO 1 J=1,N
    1 P(J)=A(I,J)
      AX=VMAXA(P,N)
      AMAX=AMAX1(AX,AMAX)
    2 CONTINUE
      SMAXA=AMAX
      RETURN
      END
```

```
      SUBROUTINE VADD(B,C,A,N)
      DIMENSION A(N),B(N),C(N)
      DO 1 I=1,N
    1 A(I)=B(I)+C(I)
      RETURN
      END


      SUBROUTINE VLINE(B,ALPHA,C,BETA,A,N)
      DIMENSION A(N),B(N),C(N)
      DO 1 I=1,N
    1 A(I)=B(I)*ALPHA+C(I)*BETA
      RETURN
      END


      SUBROUTINE VSUB(A,B,X,N)
      DIMENSION A(N),B(N),X(N)
      DO 1 I=1,N
    1 X(I)=A(I)-B(I)
      RETURN
      END


      SUBROUTINE VEQ(B,A,N)
      DIMENSION A(N),B(N)
      DO 1 I=1,N
    1 A(I)=B(I)
      RETURN
      END


      SUBROUTINE VMOD(D,E,N)
      DIMENSION D(N),E(N)
      DO 1 I=1,N
    1 E(I)=ABS(D(I))
      RETURN
      END


      FUNCTION VMAXA(A,N)
      DIMENSION A(N)
      AMAX=0.
      IF(N.LE.0) GO TO 2
      DO 1 I=1,N
      IF(AMAX.LT.ABS(A(I))) AMAX=ABS(A(I))
    1 CONTINUE
    2 VMAXA=AMAX
      RETURN
      END
```


Литература

I. В.Ц.Банчев. Сообщение ОИЯИ, Р11-9678, Дубна, 1976.
2. Л.Александров. Сообщение ОИЯИ, Р5-7259, Дубна, 1973.
3. J.D.Lawson. SIAM J. Numer. Anal., v. 4, No 3, 1967.
4. M.Mori. Publ. RIMS, Kyoto Univ., v. 10, No 1, 1974.

Рукопись поступила в издательский отдел
2 апреля 1976 года.