Процедуры алгоритма Рунге-Кутта

Структура, которая содержит физические величины, необходимые в процессе вычислений переноса излучения в параметрах Стокса:

```
  TYPE
    TyETA = RECORD
      DL       : REAL;   (* DELTA_LAMBDA            *)
      X        : REAL;   (* LG(TAU)                 *)
      EP,EB,ER : REAL;   (* ETA-PI,ETA-BLUE,ETA-RED *)
      RP,RB,RR : REAL;   (*  RO-PI, RO-BLUE, RO-RED *)
   EI,EQ,EU,EV : REAL;   (* ETA_I,ETA_Q,ETA_U,ETA_V *)
      RQ,RU,RV : REAL;   (*        RO_Q, RO_U, RO_V *)
      EC       : REAL;   (* ETA_LAM / ETA_5000      *)
      EF       : REAL;   (* EF=EC+EI                *)
      ERER     : REAL;   (* EQ*RQ+EU*RU+EV*RV       *)
      EERR     : REAL;   (* EQ*EQ+EU*EU+EV*EV-RQ*RQ-RU*RU-RV*RV *)
        B      : REAL;   (* Ф-ИЯ ПЛАНКА             *)
        S      : REAL;   (* Ф-ИЯ ИСТОЧНИКА          *)
        C10    : REAL;   (* 1/CTETA*LN(10)*10^X     *)
    END;
```

От X (от τ) не зависит, глобальные переменные не используются:

```
PROCEDURE LINA(V,VH,A:REAL;VAR ZEE:TZEEMAN;VAR E:TyETA);
(*
 *  ВЫЧИСЛЯЕТ (ЭТА_P,ЭТА_B,ETA_R)/ЭТА0
 *            ( RO_P, RO_B, RO_R)/ЭТА0
 *  ВХОДНЫЕ ПАРАМЕТРЫ:
 *            A - ПАРАМЕТР ЗАТУХАНИЯ
 *            V - ОТКЛОНЕНИЕ В DLAMB/DLD ОТ ЦЕНТРА ЛИНИИ
 *            VH - МАГНИТНОЕ РАСШЕПЛЕНИЕ DLH/DLD
 *                   DLD - ДОПЛЕРОВСКАЯ ПОЛУШИРИНА
 *            ZEE.(NPI,NCOMP,RUNGE,PNORM) - СТРУКТУРА ЛИНИИ
 *)
VAR
  P,R   : REAL;
  VB,VR : REAL;
  H ,F  : REAL;
  J     : INTEGER;
BEGIN
 WITH ZEE DO BEGIN
  WITH E  DO BEGIN
   EP:=0.0;EB:=0.0;ER:=0.0;
   RP:=0.0;RB:=0.0;RR:=0.0;

   FOR J:=1 TO NCOMP DO BEGIN
     P:=PNORM[J];
     R:=RUNGE[J];
   (*IF (R=0.0) THEN P:=P/2.0; - ПРИ ВВОДЕ*)
     VB:=V-VH*R;
     VR:=V+VH*R;
     IF J<=NPI THEN BEGIN
       VOIG(A,VB,H,F);
       EP:=EP+H*P;
       RP:=RP+F*P;
       VOIG(A,VR,H,F);
       EP:=EP+H*P;
       RP:=RP+F*P
     END
     ELSE BEGIN
       VOIG(A,VB,H,F);
```

```
       IF (H<1.E-30) THEN H:=0;
       EB:=EB+H*P;
       RB:=RB+F*P;
       VOIG(A,VR,H,F);
(* ПРИ A=0 БЫВАЕТ ИСЧЕЗНОВЕНИЕ ПОРЯДКА, ЕСЛИ H<1.5E-38 *)
       IF (H<1.E-30) THEN H:=0;
       ER:=ER+H*P;
       RR:=RR+F*P
     END;
   END;
(* VR:=0.0;
   VOIG(A,VR,H,F);  IF IsBit(IO,12) THEN H:=1.0;
   RP:=-2./H*RP; (? НАДО ЛИ ДЕЛИТЬ НА H(A,V=0)? ?)
   RB:=-2./H*RB;
   RR:=-2./H*RR;
*)
   RP:=-2.*RP;
   RB:=-2.*RB;
   RR:=-2.*RR;
  END;(*WITH E*)
 END;(*WITH ZEE*)
END;(*LINA*)


PROCEDURE XETA(VAR E:TyETA;ETA0:REAL);
(*ДОМНОЖЕНИЕ КОЭФ-ТОВ ПОГЛОШЕНИЯ НА ЭТА_0*)
BEGIN
 WITH E DO BEGIN
  EP:=EP*ETA0;ER:=ER*ETA0;EB:=EB*ETA0;
  RP:=RP*ETA0;RR:=RR*ETA0;RB:=RB*ETA0
 END;(*WITH E*)
END;(*XETA*)


PROCEDURE COEF(VAR E:TyETA;SG2,CG,SX,CX:REAL);
(*НАЙТИ КОЭФ-ТЫ ПРАВОЙ ЧАСТИ УР-ИЙ ПЕРЕНОСА*)
VAR QU : REAL;
BEGIN
  (* EP,EB,ER => EI,EQ,EU,EV *)
  (* RP,RB,RR =>   ,RQ,RU,RV *)
 WITH E DO BEGIN
  EI:=EP*SG2+(EB+ER)*(2.0-SG2);
  QU:=(EP-(EB+ER))*SG2;
  EQ:=QU*CX;
  EU:=QU*SX;
  EV:=2.0*(ER-EB)*CG;
  QU:=(RP-(RB+RR))*SG2;
  RQ:=QU*CX;
  RU:=QU*SX;
  RV:=2.0*(RR-RB)*CG
  EF:=EI+EC;     (* EC УЖЕ ИЗВЕСТНО ? *)
 END;(*WITH E*)
END;(*COEF*)


(* вычисляем все фактические параметры CONT.WK.E от X и DL *)
PROCEDURE FACX;
VAR
(* глобальные переменные
  CONT.WK.X      - глубина, для которой вычисляются физ.параметры
  CONT.WK.DL     - расст.до опорной дл.волны, для к-рого вычисляем
  CONT.WK.E      - набор коэф-тов, а также X и DL для них         *)
  DLH   : REAL;         (* РАСШЕПЛ.ЛИНИИ С G=1 (В МИЛЛИАНГС.)*)
  DLD   : REAL;         (* ДЕЛЬТА ЛЯМБДА ДОПЛЕР.В МИЛЛИАНГС. *)
  DLVD  : REAL;         (* ДОПЛЕР.СМЕШ.ЦЕНТРА ЛИНИИ (В М.А.) *)
  GM    : REAL;         (* УГОЛ ВЕКТОРА М.П. С ЛУЧЕМ ЗРЕНИЯ  *)
```

```
  CG,SG2: REAL;         (* COS(GM),SIN(GM)^2                  *)
  XI    : REAL;         (* АЗИМУТ ВЕКТОРА ПОЛЯ                *)
  CX,SX : REAL;         (* COS(XI),SIN(XI)                    *)
  A     : REAL;         (* ПОСТ.ЗАТУХАНИЯ                     *)
  ETA0  : REAL;         (* ПОГЛОШ.В ЦЕНТРЕ ЛИНИИ               *)
  V     : REAL;         (* РАССТ.ОТ ЦЕНТРА ЛИН.В ЕД./ДОПЛ.УШИР.*)
  VH    : REAL;         (* DLH/DLD                            *)

  QX    : boolean;  (* изменился X - надо пересчитать всё от него зависящее *)
  iILIN : integer;      (* индекс в списке линий *)

  sEI,sEQ,sEU,sEV,sRQ,sRU,sRV : real; (* промежут.суммы для списка линий *)
BEGIN

  QX := (CONT.WK.X <>  (* X, с которым вошли в процедуру          *)
        CONT.WK.E.X); (* X, для которого актуальны физ.параметры *)

  IF QX OR (CONT.WK.DL<>CONT.WK.E.DL) THEN BEGIN  (* изменился X или DL *)
    CONT.WK.E.X :=CONT.WK.X;
    CONT.WK.E.DL:=CONT.WK.DL;

    if QX then begin
      CONT.WK.E.C10:=C_LN10*EXP10(CONT.WK.X)/CONT.CTETA; (* TAU/COS(TETA) *)

      CONT.WK.E.B  := CONT.MB.func(CONT.WK.X);
      CONT.WK.E.S  := CONT.MS.func(CONT.WK.X);
      CONT.WK.E.EC := CONT.MEC.func(CONT.WK.X);

      DLH  := CONT.MDLH.func (CONT.WK.X);
      GM   := CONT.MGM.func  (CONT.WK.X);
      XI   := CONT.MXI.func  (CONT.WK.X);
      DLVD := CONT.MDLVD.func(CONT.WK.X);

      CG   := COS(GM*C_PI180);
      SG2  := 1.0-CG*CG;
      CX   := COS(C_PI90*XI);
      SX   := SIN(C_PI90*XI);
    end;

    if CONT.NLIN > 1 then begin  (* временные суммы *)
      sEI := 0; sEQ := 0; sEU := 0; sEV := 0;
                sRQ := 0; sRU := 0; sRV := 0;
    end;
    for iILIN := 1 to CONT.NLIN do begin
      if QX then begin
        DLD  := CONT.MLI[iILIN].MDLD.func(CONT.WK.X);
        A    := CONT.MLI[iILIN].MA.func  (CONT.WK.X);
        ETA0 := CONT.MLI[iILIN].META.func(CONT.WK.X);
      end;
      VH   := DLH/DLD;
      V    := (CONT.WK.DL+DLVD)/DLD;
      LINA(V,VH,A,CONT.MLI[iILIN].ZEE,CONT.WK.E);
      (* вычисляет EP:=0.0;EB:=0.0;ER:=0.0;
                   RP:=0.0;RB:=0.0;RR:=0.0;   *)
      XETA(CONT.WK.E,ETA0); (* умножаем E/R(P/B/R) на ETA0 *)
      COEF(CONT.WK.E,SG2,CG,SX,CX); (* находим E/R(I,Q,U,V) *)
      if CONT.NLIN > 1 then begin
        sEI := sEI + CONT.WK.E.EI;
        sEQ := sEQ + CONT.WK.E.EQ;
        sEU := sEU + CONT.WK.E.EU;
        sEV := sEV + CONT.WK.E.EV;
        sRQ := sRQ + CONT.WK.E.RQ;
        sRU := sRU + CONT.WK.E.RU;
        sRV := sRV + CONT.WK.E.RV;
      end;
    end;

    if CONT.NLIN > 1 then begin  (* вернем промежуточные суммы в коэф-ты *)
```

```
      CONT.WK.E.EI := sEI;
      CONT.WK.E.EQ := sEQ;
      CONT.WK.E.EU := sEU;
      CONT.WK.E.EV := sEV;
      CONT.WK.E.RQ := sRQ;
      CONT.WK.E.RU := sRU;
      CONT.WK.E.RV := sRV;
    end;

  END;(*IF X<>X,DL<>DL*)
END;(*FACX*)
```

```
PROCEDURE TRACH.RACHI (* АЛГОРИТМ АНАЛИТИЧЕСКОГО РАСЧЕТА ПРОФИЛЕЙ СТОКСА *)
          (VAR P:TMILN;            (* ПАРАМ.АТМ-РЫ МИЛНА-ЭДДИНГТОНА *)
           VAR ZEE:TZEEMAN;        (* СТРУКТУРА РАСЩЕПЛЕНИЯ ЛИНИИ   *)
           VAR UN:TUNNO);
    VAR
  I    : INTEGER;       (* ШАГИ ПО ПРОФИЛЮ                          *)
  V    : REAL;          (* РАССТ.ОТ ЦЕНТРА ЛИНИИ В ЕД./ДОПЛ.УШИР.   *)
  VH   : REAL;          (* DLH/DLD                                  *)
  E12,EE,RRRR,ZN:REAL;  (* ПРОМЕЖУТОЧНЫЕ ВЕЛИЧИНЫ В ФОРМУЛАХ        *)
  (*EF,ER,RR : REAL*)
  E  : TyETA;           (* КОЭФ-ТЫ ПОГЛОЩ.(И ИЗЛУЧ.) В ПАРАМ.СТОКСА *)
  BC : REAL;            (* Betta*Cos(Teta)/(1+Betta*Cos(Teta))      *)
  CG,SG2: REAL;         (* COS(GM),SIN(GM)^2 *)
  CX,SX : REAL;         (* COS(XI),SIN(XI)   *)

BEGIN
  if P.DLD = 0 then App.Err('RACH DLD=0!!!');
  UN.InitIQUV;
  E.EC:=1.0;
  WITH P DO BEGIN
    CG:=COS(GM*C_PI180);
    SG2:=1.0-CG*CG;
    CX:=COS(C_PI90*XI);
    SX:=SIN(C_PI90*XI);
    BC:=BET*CTEM/(1+BET*CTEM);
    VH:=DLH/DLD;
    FOR I:=1 TO UN.NDL DO
    BEGIN
      V:=UN.UL[I]/DLD;
      LINA(V,VH,A,ZEE,E);
      XETA(E,ETA0);
      COEF(E,SG2,CG,SX,CX);

      WITH E DO BEGIN
        IF IsBit(IO,14)  (* ПРЕНЕБРЕЖЕНИЕ АНОМАЛЬНОЙ ДИСПЕРСИЕЙ *)
          THEN BEGIN RQ:=0;RU:=0;RV:=0 END;
        EF:=EI+EC;
        E12:=EF*EF;
        ERER:=EQ*RQ+EU*RU+EV*RV;
     (* IF (ABS(ERER))<1.E-15 THEN ERER:=0.0; *)
        RRRR:=RQ*RQ+RU*RU+RV*RV;
        EE  :=EQ*EQ+EU*EU+EV*EV;
        EERR:=EE-RRRR;
        ZN:=E12*(E12-EERR)-ERER*ERER;
        IF IsBit(IO,12)                (* ПРОФИЛЬ ПОГЛОЩЕНИЯ *)
        THEN BEGIN
          UN.UI[I]:=1.0-EI/ETA0;
          UN.UQ[I]:=0.0-EQ/ETA0;
```

```
          IF IsBit(IO,14)
            THEN UN.UU[I]:=ZN/E12/ETA0
            ELSE UN.UU[I]:=ZN/ETA0/ETA0;
          UN.UV[I]:=0.0-EV/ETA0;
        END
        ELSE BEGIN
          UN.UI[I]:=1.0-BC*(1.0-EF*(E12+RRRR)/ZN);
          UN.UQ[I]:=0.0-BC*(E12*EQ+EF*(EV*RU-EU*RV)+RQ*ERER)/ZN;
          UN.UU[I]:=0.0-BC*(E12*EU+EF*(EQ*RV-EV*RQ)+RU*ERER)/ZN;
          UN.UV[I]:=0.0-BC*(E12*EV+                RV*ERER)/ZN;
        END;
      END;(*WITH E*)
    END;(*FOR I *)
  END;(*WITH P*)
  UN.CONT  := 1.0;
  UN.KCONT := 1;
  UN.K4    := 4;
  UN.KSym  := 1; (* половина профиля *)
END;(*RACHI *)
```