

smex
 30.9.69
 Stellar Models
 Homogeneous
 Exercise Version, J. Otzen

begin

real pi, mod, recmo, G, k, m0, a, c, Ls, Rs, Ms, lTes, L, Te, R, lTe, lL, lR, L0, LV,
 eps, kappa, arb, arb1, U1, U3, cnsqm, epsp, epsn, Lneu, eps4, epsg,
 testg, testn, q, T, rho, P, T4, beta, age, M, X, Y, Z, qfit,
 XomH, YomHe, ZomA, eomua, eomue, tb, rb, sec,
 A0, A1, A2, A3, A4, A5, A6, A01, A02, A08, A09, Atim4, Alp5, alter,
 delta, dd, ddr, dda, gr, alder, fsum, gsum, F, dpar, step, sres, mf,
 gacc, alfa, l, Hp, U, si, t1, ltb, lps, slT, slP,
 C1, C2, C3, C4, dummy, exp30;

integer s, t, i, j, nm, t4, UNRW, UNWW, UNWP, UNWD, mode,
 g, autm, mtx4, ccore, nul, to, tre, fire, fem, num, rat, lat, ka,
 ztype, ilt, ilp, tsg, tsd;

array k1, y[1:5], XM[1:4], km[1:1,0:17,0:20], fmas[0:1],
 fp, lHJU, ddaa, lkap[0:30, 0:24], rltp, corr, res, ara[1:4], drdp[1:4, 1:4];

real procedure antlog(r); value r; real r; antlog:= exp(r/mod);

real procedure log(r); value r; real r; log:= mod*ln(r);

comment procedure LLGAUSS solves n linear equations with n unknowns;

procedure LLGAUSS(n,R,A,U,delta); value n,delta; integer n; real delta;

array R,A; label U;

begin integer i,j,k,h; real a,b;

k:= 0; Do it again: k:= k+1; a:= A[k,k]; h:= k; for i:= k step 1 until n-1 do

begin b:= A[i+1,k]; if abs(a)>abs(b) then go to S; h:= i+1; a:= b; S: end;

if abs(a)<delta then go to U; if h=k then go to eliminate;

for j:= k step 1 until n+1 do begin b:= A[k,j]; A[k,j]:= A[h,j]; A[h,j]:= b end;

eliminate: for i:= k+1 step 1 until n do

begin b:= -A[i,k]/a; for j:= k+1 step 1 until n+1 do A[i,j]:= A[i,j]+A[k,j]*b end;

if k<n then go to Do it again; next unknown: b:= 0; for j:= n step -1 until k+1 do

b:=R[j]*A[k,j]+b; R[k]:= (A[k,n+1]-b)/A[k,k]; k:= k-1;

if k>0 then go to next unknown

end LLGAUSS;

comment In the following block the tables are read to the arrays

fp, HJU, ddaa, lkap, and km;

begin integer k;

real t, p, t1;

procedure fpline(A); array A;

begin for j:= 0 step 1 until (if k<ilp then k else ilp) do A[i,j]:= read real;

for j:= ilp+1 step 1 until k do dummy:= read real end;

UNRW:= 16; UNWW:= 17; UNWD:=9; UNWP:= 33; A02:= 0.2;

nul:= 0; to:= 2; tre:= 3; fire:= 4; fem:= 5; Atim4:= 10-4;

NEWFP: dummy:= select(UNWW); writetext(⟨<

t1, ilt:= ⟩); t1:= read real; arb:= read real;

dummy:= select(UNRW);

ltb:=read real; ilt:=read integer; lps:=read real; ilp:=read integer;

```

X:= read real; Y:= read real; Z:= read real;
slT:= read real; dummy:= read real; slP:= read real; dummy:= read real;
label 1: t:= read real; p:= read real;
      if t < t1-Atim4 then begin
        ilp:= read integer; for j:=1 step 1 until 4*(ilp+1) do dummy:= read real;
        go to label 1 end;
ltb:= t; lps:= p; ilt:=arb; ilp:=24;

for i:= 0 step 1 until ilt do
begin if i≠0 then begin t:= read real; p:= read real end;

k:= read integer;
if abs(t-ltb-slTxi)+abs(p-lps-slPxi)>Atim4 then begin
dummy:= select(UNWW); writetext(⟨<
Error in fp-table⟩); dummy:= lyn; go to NEWFP end;
fpline(fp); fpline(lkap); fpline(ddaa); fpline(LHJU);
end reading of fp tables;

ny tab: dummy:= select(UNWW); dummy:= lyn; dummy:= select(UNRW);
tb:= read real; rb:= read real;
ka:= read integer; t:= read integer; nm:= read integer;

for i:= 1 step 1 until nm do
begin XM[i]:= read real;
for j := 0 step 1 until t do begin
if abs(read real-tb-jxA02)>Atim4 then begin writetext(⟨<
TABLEFEJL⟩); go to ny tab end;
dummy:=
read real; dummy:= read real; for k:= 0 step 1 until ka-1 do
km[i,j,k]:= read real end
end læs kappa;
end tablereading;

dummy:= select(UNWW);
writetext(⟨<
STELLAR MODELS WITH OUTER ZONE
HOMOGENEOUS
EXERCISE VERSION

1/Hp:= ⟩); alfa:= read real;

begin
switch SW:= choose, new model, new parameters, matrix, improve, frem;

comment procedure MERSN performs all integrations;
procedure MERSN;
begin real x0, ho3, eps, eq, gstep, tstop; boolean last;
array y0, k3, k4, k5 [1:5];

procedure CRGR;
begin integer t; real gtg, teste, qe, qg, qn;

comment procedure CRGR determines where transitions between radiative and
convective zones occur;

t:= 0 ; gtg:= -testg; qe:= q; teste:= testn; qg:= x0; writechar(29);

iter: if abs(testn)>Atim4^t<15 then
begin if t>12 v kb on then
begin if ztype<3 then WATM else SKRIV; end;
qn:=(testexqg-testgxqe)/(teste-testg);
for i:= 1 step 1 until nm do y[i]:= y0[i]; q:= x0; DIFF(k1);
RKM(qn-x0); t:= t+1; if t=14 then writechar(62); DIFF(k4);

```

```

if sign(testn)=sign(teste) then begin qe:= qn; teste:= testn end
else begin qg:= qn; testg:= testn end;
go to iter end;

```

```

if ztype<3 then WATM else SKRIV; writechar(62); DIFF(k1);
if abs(q- qfit)>210-7 then last:= false;
testg:= testn:= abs(testn)×sign(gtg);
end procedure CRGR;

```

```

procedure RKM (h); value h; real h;
begin ho3:= h/3;
for i:= 1 step 1 until nm do y[i]:= k1[i]×ho3+y0[i]; q:= x0+ho3;
DIFF(k3); for i:= 1 step 1 until nm do y[i]:= (k1[i]+k3[i])/2×ho3+y0[i];
DIFF(k3); for i:= 1 step 1 until nm do y[i]:= (k1[i]×.375+k3[i]×1.125)×ho3+y0[i];
q:=x0+h/2; DIFF(k4);
for i:= 1 step 1 until nm do y[i]:= (k1[i]×A1p5-k3[i]×4.5+k4[i]×6)×ho3+y0[i];
q:= h+x0; DIFF(k5);
for i:= 1 step 1 until nm do y[i]:=((k1[i]+k5[i])/2+k4[i]×2)×ho3+y0[i];
end RKM;

```

```

procedure RKMER(h); value h; real h;
begin x0:= q; for i:= 1 step 1 until nm do y0[i]:= y[i];
RKM(h); eps:= A0;
for i:= 1 step 1 until (if nm>4 then 4 else nm) do
begin eq:= abs(k1[i]×A02-k3[i]×A09+k4[i]×A08-k5[i]×A01);
if eq>eps then eps:= eq end; eps:= eps×abs(ho3);

```

```

if delta>eps then tsg:= tsg+1 else
begin comment step unacceptable;
if kb on then writetext(†<.†); tsd:= tsd+1;
for i:= 1 step 1 until nm do y[i]:= y0[i]; q:= x0; go to QR
end;
end RKMER;

```

comment In the first part of the procedure body the outer zone is integrated;

```

if ztype<3 then begin tstop:= fmas[1]/mod;
TOT6: RKMER(step); DIFF(k1); if sign(testn)= sign(testg)^abs(testn)>Atim4
then testg:= testn else CRGR;
if kb on v mode = 4 then WATM;

```

```

if ztype=1 then go to (if y[1]<A01 then QR else finis);
if ztype=2 then go to (if y[3]< tstop then QR else TILQ);

```

```

TILQ: delta:= delta/Atim4;
IGEN: if abs(y[3] - y0[3])>10-7 then begin
DIFF(k1); if kb on then WATM;
RKMER((x0-q)×(y[3] - tstop)/(y[3] - y0[3]));
if abs(y[3] -tstop)>510-7 then go to IGEN
end;
delta:= delta×Atim4;
go to finis;
end integration with ztype<3;

```

comment In the second part of MERSNs body the inner zone is integrated (ztype = 3);

```

nm:= if mode=4 then 5 else 4; go to R;
try last: last:= true; step:= qfit -q; go to R1;
R: if abs(step)>.05 then step:= sign(step)×.05;
if abs( qfit-q)≤abs(step) then go to try last;

```

```

last:= false;
R1: RKMER(step); DIFF(k1);
if sign(testn)= sign(testg) ^ abs(testn)>Atim4 then testg:= testn else CRGR;
if kb on v mode=4 then SKRIV;
if last then go to finis;

```

comment In the last part of MERSN the new integration step is found;

```

if eps = A0 then begin step:= A2×step; go to R end;
QR: step:= (delta/eps)∧A02×A08×step; go to (if ztype<3 then TOT6 else R);
finis: end MERSN;

```

```

procedure DIFF(K); array K;
begin integer mt,mr,tad;

```

comment In the first part of DIFF (if ztype<3) the derivatives needed in the outer zone are computed;

```

if ztype<3 then

```

```

begin integer t,p;
real tm, pn, m, n, cmm1, cm0, cml, cnml, cn0, cn1, x2, FP, PmPr;

```

```

real procedure INTP(A); array A;
INTP:= (A[t-1, p-1]×cnml+A[t-1, p]×cn0+A[t-1, p+1]×cn1)×cmm1
      +(A[t, p-1]× cnml+A[t, p]× cn0+A[t, p+1]× cn1)×cm0
      +(A[t+1, p-1]×cnml+A[t+1, p]×cn0+A[t+1,p+1]× cn1)×cml;

```

```

procedure MIXING;
begin real U8,U16,fsi,fm,hs,s2;

```

comment Here $d \ln T / d \ln P$ is computed according to the mixing - length theory with $l := \alpha \times H_p$;

```

l:= alfa×Hp; U:= C3/eomua×Hp×T4/(T×exp(arb/mod)×FP×P/PmPr×kappa×rho×l×l);
U8:= 8×U/9; U16:= 2×U8; hs:= U8×(ddr-dda);
la: s2:= sixsi; fsi:= s2×si+U8×s2+U16×U×si-hs;
fm:= 3×s2+U16×(si+U); si:= si-fsi/fm;
if abs(fsi/hs)>1.10-6 then go to la; dd:= dda+sixsi+2×U×si;
end mixing procedure;

```

```

P:= exp(q);
if ztype=1 then begin T:=Tex(0.5+0.75×y[1])∧0.25; y[3]:=ln(T) end else T:=exp(y[3]);
tm:=(y[3]×mod-ltb)/slT; t:= entier(tm);
pn:=(q×mod - (lps + tm×slP))/slP; p:= entier(pn);
if ilp > 2×p then p:= p+1;
m:= tm - t; n:= pn - p ;
if t < 1 ∨ t > ilt -1 ∨ p < 1 ∨ p > ilp - 1 then CONTROL;
cmm1:= m/2×(m-1); cm0:= 1-m×m; cml:= m/2×(m+1);
cnml:= n/2×(n-1); cn0:= 1-n×n; cn1:= n/2×(n+1);
T4:=T×T; T4:= T4×T4; PmPr:= P-a/3×T4; FP:= INTP(fp); rho:= PmPr/FP/C2/T;
kappa:= exp(INTP(lkap)/mod);
dd:= ddr:= C1×kappa×P/(y[2]×T4);

```

```

if ztype=1 then K[1]:= kappa×P/gacc else
begin x2:= y[1]×y[1];
Hp:= x2×P/(y[2]×rho×gacc);
dda:=INTP(ddaa); testn:=A1/ddr -A1/dda;
if ddr>dda then begin arb:= INTP(lHJU); MIXING; end;
K[1]:= -Hp/(R×Rs);
K[2]:= -C4×x2×rho×Hp;

```

```
K[3]:= dd
end end
```

In the following second part of DIFF the derivatives needed in the inner zone are computed

```
else begin real V,r3,T616,T626,T646,T6,t,r,f0,f1,n,p,p1,mnul,m1,Pc,ks;
```

```
comment Now rho is determined through the equation of state;
```

```
XomH:= X/1.00801; YomHe:= Y/4.0028; ZomA:= Z/16.942;
```

```
eomue:= XomH + A2 × YomHe + Z/A2;
```

```
eomua:= XomH + YomHe + ZomA;
```

```
T:= exp(y[3]); P:= exp(y[4]); V:= T3; T4:= V × T;
```

```
beta:= A1 - a/A3 × T4/P;
```

```
t:= P×beta/(T×k/m0); arb:= sqrt(V); rho:= t/(eomua+eomue×F);
```

```
ks:= X+7×Z+A3; r3:= cnsqm×ks3;
```

```
DEGL: Pc:= sqrt(r3×rho/T)×rho;
```

```
t:= (P×beta+Pc)/(T×k/m0); f1:= rho;
```

```
f0:= rho/(9.047610-9×arb)×eomue; r:= (f0 - 30)/30;
```

```
F:= (((0.109744×r-0.165376)×r+0.151420)×r-0.462698)×r+3.296236)×r+5.187420;
```

```
rho:= t/(eomua + eomue×F);
```

```
if abs((f1-rho)/rho)>10-5 then go to DEGL ;
```

```
V:= V/rho; dda:= A1/(A4-A1p5×beta×beta/(A4-A3×beta)); t:= modxy[3];
```

```
comment In the following statement the energy generation rate is
determined;
```

```
if X<10-7 then eps:= eps4:= 0 else
```

```
begin T6:= T×10-6; T616:= T61/6; T626:= T6162; T646:= T6262;
```

```
ks:= 0.1332109×sqrt(ks/V);
```

```
arb:= 2.37810-16×(1+0.00922×T626)×exp(-A3×ks+99.96/T626 -30)×exp30×(X/(1-X-Z))2;
```

```
arb1:= sqrt(1+arb); arb:= (arb1+arb/A2-1)/arb;
```

```
epsps:= 4.199106×arb×(1+0.0123×T626+0.00781×T646+0.00067×T6)/T646×
```

```
exp(ks-33.809/T626)×rho×X2;
```

```
if T6<12 then eps4:= 0 else
```

```
eps4:= 4.4581027×(1+0.00274×T626)/T646×exp(7×ks-152.31/T626)×rho×X×Z;
```

```
arb:= 1/(1+2.4281016×X/(1+X)×(1+0.0041×T626)/T616×exp(A4×ks-102.64/T626));
```

```
eps:= epsps×(0.980+(0.964×arb-1.044)/(arb1+A3))+eps4×0.936;
```

```
epsn:=epsps+eps4-eps;
```

```
if T6<7 then eps:= if T6>6.5 then eps×(T6-A6) else eps/A2; end;
```

```
epsps:= 0; eps:= eps+epsps;
```

```
comment In the following the opacity is determined by interpolation in tables;
```

```
r:= mod×ln(rho); arb:= (r-rb-A3×(t-tb))×A5; arb1:= (t-tb)×A5;
```

```
mt:= entier(arb1); mr:= entier(arb);
```

```
if mr<nul then begin mr:= nul; KTABS end;
```

```
if mr>ka-2 then begin mr:= ka-to; KTABS end;
```

```
tad:= 0; p:= 1;
```

```
f0:= km[tad+1,mt,mr]×p;
```

```
f1:= km[tad+1,mt,mr+1]×p;
```

```
n:= arb-mr; mnul:= f0+n×(f1-f0);
```

```
f0:= km[tad+1,mt+1,mr]×p;
```

```
f1:= km[tad+1,mt+1,mr+1]×p;
```

```
m1:= f0+n×(f1-f0); n:= arb1-mt;
```

```
kappa:= exp((mnul+n×(m1-mnul))/mod);
```

```
comment Now the derivatives are easily written down;
```

```

r:= exp(y[1]); r3:= r3;
K[1]:= U1×M/(r3×rho); K[2]:= Ms×M×eps/L0; K[5]:= Ms×M×epsn/L0;
K[4]:= -G×(Ms×M/r2)×q/P/(A4×pi)×(Ms×M/r2);
ddr:= U3/M×kappa×y[2]×L0×P/(q×T4); testn:= A1/ddr-A1/dda;
K[3]:= K[4]×(if ddr<dda then ddr else dda);
end diff for innerzone;
end procedure DIFF;

```

```

procedure KTABS; begin s:=select(UNWW); writetext(⟨<
ext⟩); dummy:= select(s); end;

```

```

procedure CONTROL;
begin dummy:= select(16); writetext(⟨<
fp table too small logT = ⟩); write(⟨-d.dddd⟩,y[3]×mod);
writetext(⟨< logP = ⟩); write(⟨-d.dddd⟩, q×mod); writetext(⟨<
Input new atm def⟩);
go to rep
end;

```

```

procedure WATM;
begin writecr;
if ztype=2 then
write(⟨-dd.dddd⟩, y[2], y[1], 1, log(T), log(P), log(rho), log(kappa),
log(ddr), log(dda), log(dd))
else write(⟨-dd.dddd⟩, q×mod, log(T), log(rho), log(kappa), log(ddr), y[1]);
end WATM;

```

```

procedure ATMOF(lTe, lL); value lTe, lL; real lTe, lL;
begin WTS;
lR:= A2×(lTes - lTe) + lL/A2; Te:= exp(lTe/mod); R:= exp(lR/mod); L:= exp(lL/mod);
gacc:= G×M×Ms/(R×Rs)2; y[2]:= 1; C1:= U3×Ls×L/M; C4:= 4×pi/gacc×G;
eomua:= X/1.00801+Y/4.0028+Z/16.942; C2:= k×eomua/m0;
if kb on then begin
writecr; write(⟨-dd.dddd⟩, M, lTe, lL, lR, log(gacc), 5040/Te×20.25); end;
nm:= ztype:= 1; q:=(lps + (log(Te×0.50.25) - ltb)/slT×slP)/mod +0.01; y[1]:= 0;
adlP: DIFF(k1); if kb on then WATM;
if k1[1]<0.01 then begin q:= q+1; go to adlP end;

```

```

step:= 0.1; MERSN; if kb on then begin DIFF(k1); WATM end;

```

```

ztype:= 2; nm:= 3; y[3]:= ln(Te×(0.5+0.75×y[1]))0.25; y[1]:= A1; DIFF(k1);
testg:= testn; if kb on then WATM;
step:= 0.05; MERSN;

```

```

if kb on then begin DIFF(k1); WATM end;

```

```

y[1]:= ln(y[1]×Rs×R); arb:= y[2];
y[2]:= antlog(lL)×Ls/L0;
y[4]:= q; F:= A1; q:= arb; ztype:= 3; lneu:= y[5]; y[5]:=0; DIFF(k1);
if kb on then SKRIV;

```

```

testg:= testn; step:= -0.0002; WTS
end ATMOF;

```

```

procedure WTS;
begin if kb on then begin
writecr; write(⟨ddddd⟩, tsg, tsd);
end end WTS;

```

```

procedure CKUGL(Tc, rhoc); value Tc, rhoc; real Tc, rhoc;
begin

```

```

XomH:= X/1.00801; YomHe:= (1-X-Z)/4.0028; ZomA:= Z/16.942;
eomue:= XomH + A2 × YomHe + Z/A2;

q:= fmas[0]; y[1]:= A1; y[to]:= Atim4;
ztype:= 3; y[3]:= Tc;
T:= exp(Tc); rho:= exp(rhoc);
arb:= F:= rho/(9.047610-9×exp(A1p5×Tc))×eomue;
F:= (F-30)/30; F:= (((0.109744×F-0.165376)×F+0.151420)×F-0.462698)×F+3.296236)×F
+5.187420;
if kb on then begin write(⟨-ddd.dddd⟩,arb,F); writecr end;
y[4]:= ln(k/m0×rho×T×(XomH+YomHe+ZomA+eomue×F)+a/A3×exp(Tc×A4)
-sqrt(cnsqm×(X+7×Z+A3)↑3×rho/T)×rho);

DIFF(k1); y[5]:= k1[5];
if kb on v mode=4 then SKRIV;
step:= q:= fmas[0];
y[1]:= ln(A3×M×Ms×q/(A4×pi×rho))/A3;
ddr:= U3×Ms×kappa×eps×P/T4;
arb:= -G×M×Ms×q/A2×rho/(exp(y[1])×P); y[2]:= eps×M×Ms×q/L0;
y[3]:= Tc+arb×(if ddr<dda then ddr else dda); y[4]:= y[4]+arb;
DIFF(k1); y[5]:= (y[5]+k1[5])/A2×q;
testg:= testn; if kb on v mode=4 then SKRIV;
end CKUGL;

procedure SKRIV;
begin if mode=4 then go to udskriv;
writecr; write(⟨-dd.dddd⟩,q,exp(y[1]-lR/mod)/Rs,y[2]×L0/(antlog(lL)×Ls),
modx[3], modxy[4], modxln(rho), modxln(kappa), log(ddr), eps); go to ud;
udskriv: writecr;
write(⟨-d.dddd⟩, q, exp(y[1]-lR/mod)/Rs,y[2]×L0/(antlog(lL)×Ls),
modx[3], 0.1×modx[4], modxln(rho), ddr, modxln(kappa),
X, beta, eps4×0.93610-4, eps×Atim4);
ud: end procedure SKRIV;

comment In the first part of the program values are assigned to many variables;

autm:= mode:= mt4:= 0; A0:= gsum:= 0; A1:= mf:= 1;
A2:= 2; A3:= 3; A4:= 4; A5:= 5; A6:= 6;
A01:= 0.1; A08:= 0.8; A09:= 0.9; A1p5:= 1.5;
pi:= 3.14159265; mod:= 0.434294482; recmo:= 2.30258509;
G:= 6.66810-8; m0:= 1.6602610-24; k:= 1.3804610-16; a:= 7.564110-15; c:= 2.9979291010;
Ls:= 3.901033; Rs:= 6.95981010; Ms:= 1.9891033; lTes:= log(Ls/(a×pi×c×Rs↑2))/A4;
U1:= Ms/(A4×pi); U3:= A3/(16×pi×a×c×G×Ms); cnsqm:= pi/k/8×(4.8029↑3/m0/31030)↑2/m0;
exp30:= exp(30); fmas[0]:= 0.0003; fmas[1]:= A6;
C3:= A6×sqrt(2)×a×c×m0/k; delta:= 0.0003; alter:= 0.005;

new model: writetext(⟨<
M, X, Y, Z, qfit := ⟩); M:= read real; arb:= read real; arb1:= read real;
LV:= read real; qfit:= read real;
if arb≠X v arb1≠Y v LV≠Z then writetext(⟨<
X, Y, Z from the tables for the outer zone are different; outer zone values are
used in the program⟩);

new parameters: writetext(⟨<
log Te, log L/Ls, log Tc, log rhoc := ⟩);
lTe:= rltp[1]:= read real; lL:= rltp[2]:= read real;
rltp[3]:= read real; rltp[4]:= read real; L0:= antlog(lL)×Ls;

rep: writetext (⟨<
M/Ms, X, Z = ⟩); write (⟨-ddd.ddd⟩, M, X, Z);

```

```

writetext (†<
log Te, log L/Ls, log Tc, log rhoc = †);
lR:= A2×(lTes-lTe) + lL/A2;
write(†-dd.dddd†, lTe, lL, rltp[3], rltp[4]); writecr;
if autm = 0 v kb on then go to SW[lyn];

tsg:= tsd:= 0;

CKUGL(rltp[3]/mod, rltp[4]/mod); MERSN;
for i:= 1, 2, 3, 4 do res[i]:= y[i];

ATMOF(lTe, lL); MERSN; fsum:= 0;
writetext (†<
res [i] = †); for i:= 1, 2, 3, 4 do begin res[i]:= res[i] - y[i];
write(†-dd.dddd†, res[i]); fsum:= fsum +abs(res[i]) end;

writetext (†<
fsum, tsg, tsd = †); write(†-dd.dddd†, fsum); write(†-dddd†, tsg, tsd); writecr; writecr

if autm = 0 then begin i:= lyn;
if i = 51 then go to matrix else go to SW[i] end
else begin
mtx4:= mtx4-1; ccore:= -2; if fsum<0.0005 then go to frem;
if fsum<0.0050 then begin ccore:= -3; go to improve end;
if fsum>0.2×gsum^mtx4<1 then go to matrix end;
gsum:= fsum;

comment In the following block 4 corrections to the parameters are found,
and to each parameter is added the corresponding correction multiplied
by a factor, mf;

improve:
begin array drdpa[1:4, 1:5];
for s:= 1, 2, 3, 4 do begin drdpa[s, 5]:= res[s];
for t:= 1, 2, 3, 4 do drdpa[s, t]:= drdp[s, t] end;
LLGAUSS(4, corr, drdpa, exit, 10-6);
for s:= 1, 2, 3, 4 do rltp[s]:= rltp[s] - corr[s]×mf;
lTe:= rltp[1]; lL:= rltp[2];
go to rep;

exit: writetext(†<
No solution†); go to SW[lyn]; go to rep;
end improve block;

comment In the following block the coefficients in the equations for the
corrections to the parameters are computed;

matrix:
begin
integer j;
array store res, store ys[1:4];

procedure ALTER(alterno);
integer alterno;
comment ALTER alters parameter no alterno, perform an testintegration, and
computes the corresponding 4 coefficients in the equations for the
parametercorrections;

begin
rltp[alterno]:= rltp[alterno] + alter;

```

```

OUTWARDS: if alterno<3 then
begin for s:= 1, 2, 3, 4 do res[s]:= store res[s] + store ys[s];
      go to INWARDS end;
CKUGL(rltp[3]/mod, rltp[4]/mod); MERSN;
for s:= 1, 2, 3, 4 do res[s]:= y[s];

INWARDS: if alterno>2 then
begin for s:= 1, 2, 3, 4 do res[s]:= res[s] - store ys[s];
      go to ENDINTEGRATION end;
ATMOF(rltp[1], rltp[2]); MERSN;
for s:= 1, 2, 3, 4 do res[s]:= res[s] - y[s];

ENDINTEGRATION:
for s:= 1, 2, 3, 4 do drdp[s, alterno] := (res[s] - store res[s])/alter;
rltp[alterno] := rltp[alterno] - alter;
end ALTER;

comment The coefficients in the linear equations for the 4 parameter corrections
are now computed by means of 4 test alterations of the parameters;

for j:= 1, 2, 3, 4 do
begin store res[j] := res[j]; store ys[j] := y[j] end;
for j:= 1, 2, 3, 4 do ALTER(j);
for j:= 1, 2, 3, 4 do res[j] := store res[j]; gsum := 1;
end matrix block;

if autm ≠ 0 then go to improve;
i:= lyn; if i=57 then go to improve else go to SW[i]; go to rep;

choose: t:= select(UNWW); writetext(⟨<
choose by letter: ⟩); i:= lyn; writecr;
if i = 53 then mode:= read integer;
if i = 54 then begin fmas[0]:= read real; fmas[1]:= read real end;
if i = 39 then dpar:= read real;
if i = 18 then go to stop;
if i = 36 then mf:= read real;
if i = 49 then autm:= read integer;
if i = 52 then delta:= read real;
if i = 57 then go to improve;

if i = 51 then begin
  for s:= 1, 2, 3, 4 do begin writecr;
    for t:= 1, 2, 3, 4 do write (⟨-ddd.dddd⟩, drdp[s,t]) end;
  end;
writecr; go to rep;

frem: writetext(⟨<
log gacc, Te, Mbol = ⟩); write(⟨ddd.dddd⟩, log(gacc)); write(⟨ddddddd⟩, antlog(lTe))
write(⟨-dddd.ddd⟩, 4.72 - 2.5×1L);
go to choose;
stop: end end
t<

```