PFC/JA-87-24

THE APPLICATION OF THE FINITE ELEMENT LIBRARY TWODEPEP FOR THE NUMERICAL SOLUTION OF THE RELATIVISTIC FOKKER-PLANCK EQUATION

by

M. Shoucri,* V. Fuchs,* A. Bers

June 1987

Plasma Fusion Center
Massachusetts Institute of Technology
Cambridge, Mass. 02139 U.S.A.

*IREQ, Project Tokamak, C.P. 1000, Varennes, Quebec, Canada
JOL 2PO

Submitted for publication in: Computer Physics Communications

THE APPLICATION OF THE FINITE ELEMENT LIBRARY TWODEPEP FOR THE NUMERICAL SOLUTION OF THE RELATIVISTIC FOKKER-PLANCK EQUATION

M. SHOUCRI, V. FUCHS, A. BERS*

IREQ, Projet Tokamak

C.P. 1000, Varennes, Québec, Canada JOL 2PO

Permanent address:

* Plasma Fusion Center, Massachusetts Institute of Technology

Cambridge, Mass. 02139, U.S.A.

Abstract

We report on the application of a commercially available differential equation solver library called TWODEPEP for the numerical solution of the relativistic Fokker-Planck equation. Results obtained by solving the runaway problem, and on the solution of the R.F. heating problem are presented.

The interaction of radio-frequency waves with a plasma is described by a Fokker-Planck equation with an added quasilinear diffusion term, and the response of a plasma to a homogeneous ohmic electric field is described by the same Fokker-Planck equation with an advection term. There has been an extensive literature on numerical techniques to solve this equation which is of high importance in the study of lower-hybrid RF-driven currents in Tokamaks (see, for instance, the review article in Ref. [1]). In this note, we report on the application of a commercially available differential equation solver library called TWODEPEP [2] for the numerical solution of the Fokker-Planck equation.

This differential equation solver has been successfully applied in solving a wide variety of problems of fluid dynamics, diffusion, heat transfer, etc. We have previously reported results obtained using thislibrary for the numerical solution to the non-relativistic Fokker-Planck equation [3, 5]. In this note we report on the application of this differential equation solver for the numerical solution of the Fortran code used to solve the relativistic Fokker-Planck equation. It illustrates how convenient is the library for the problem we are studying, if we notice that the preprocessor Fortran code has no more than a few dozen lines. We note, however, that a physical quantities of interest plus graphic output, are obtained from a separate code, which uses TWODEPEP results as input.

In the model Fokker-Planck equation we consider the plasma is assumed homogeneous, and azimuthally symmetric about the magnetic field. Relativistic effects are included. The model is described in sufficient detail in

Refs. [1], [6], its characteristic feature being a linear collision operator describing the interaction of fast test electrons with fixed Maxwellian field electrons and ions. The electron distribution fonction f depends on time and two momentum variables. In flux form the Fokker-Planck equation is

$$\frac{\partial f}{\partial t} + \nabla \cdot \vec{S} = 0 \qquad (1)$$

where f is the test-electron distribution function and $\vec{S}$ the test electron flux in the presence of an electric field $\vec{\varepsilon}$ and a quasi-linear diffusion coefficient $\overset{\Rightarrow}{D}$:

$$\vec{S} = \vec{\varepsilon}\, f - \overset{\Rightarrow}{D} \,.\, \text{grad } f + \vec{S}_{coll} \qquad (2)$$

We use the normalized variables

$$t \rightarrow t\nu \qquad v \rightarrow v/v_t \qquad P = P/P_t \qquad (3)$$

where the thermal velocity unit v_t is defined by the following relations [6]:

$$v_t^2 = \frac{1}{3n_e} \int v^2 f_m(P) d^3P \simeq \frac{T_e}{m_e} (1 - \frac{5}{2}\ \Theta + \frac{55}{8}\ \Theta^{\ 2}) \qquad (4)$$

where $f_m(P)$ denotes the relativistic Maxwellian distribution and $\Theta = T_e/(m_e c^2)$. We will restrict ourselves to the case where $\Theta \ll 1$ (for $T_e \simeq 5$ kV we have $\Theta = 0.01$) so that the thermal velocity $v_t = (T_e/m_e)^{1/2}$. The thermal momentum is given by

$$P_t = mv_t \tag{5}$$

and the thermal collision frequency is given by

$$\nu = 4\pi n_e e^4 \ln\Lambda / m^2 v_t{}^3 \tag{6}$$

with

$$v_t = (T_e/m_e)^{1/2} \tag{7}$$

The electric field is normalized to $E_o = P_t \nu / e$ and the diffusion coefficient is normalized to $D_o = \nu P_t{}^2$. Also, for $\Theta \ll 1$, the relativistic Maxwellian distribution is given in our normalized units by:

$$f_m(P) = \frac{1}{(2\pi)^{3/2}} e^{-P^2/(\gamma+1)} \tag{8}$$

where

$$\gamma = (1 + P^2 \beta_{th}^2)^{1/2} \tag{9}$$

and

$$\beta_{th} = v_t/c \tag{10}$$

In the spherically symmetric coordinates (v, μ), where $\mu = P_{\parallel}/P = \cos\theta$, we have:

$$\nabla.\vec{S} = \frac{1}{P^2}\ \frac{\partial}{\partial P}\,(P^2 S_P) - \frac{1}{P}\frac{\partial}{\partial \mu}\,(\sqrt{1-\mu^2}\ S\mu) \tag{11}$$

where

$$S_P = \varepsilon\mu f - D\mu\ \left(\mu\,\frac{\partial f}{\partial P} + (\frac{1-\mu^2}{P})\ \frac{\partial f}{\partial \mu}\right) - \left(A(P)\ \frac{\partial f}{\partial P} + F(P)\right)\ f \tag{12}$$

$$S_\mu = \left(1-\mu^2\right)^{\frac{1}{2}}\ \left[\ - \varepsilon f + D\ \left(\mu\,\frac{\partial f}{\partial P} + \frac{1-\mu^2}{P}\ \frac{\partial f}{\partial \mu}\right) + \frac{1}{P}\ \left(B(P) + \frac{Z}{2v}\right)\ \frac{\partial f}{\partial \mu}\ \right] \tag{13}$$

Here Z is the ion charge and the coefficients A(P), B(P) and F(P) are given in our normalized units by:

$$A(P) = \frac{4\pi}{3}\ \left(\frac{\gamma^3}{P^3}\ \int_0^P P'^4\ \frac{f_m(P')}{\gamma'^2}\ dP' + \int_P^\infty P'\gamma' f_m(P')\ dP'\right) \tag{14}$$

$$B(P) = \frac{4\pi}{3}\ \left(\frac{3}{2}\,\frac{\gamma}{P}\ \int_0^P P'^2 f_m(P')dP' - \frac{\gamma^3}{2P^3}\int_0^P\ \frac{P'^4 f_m}{\gamma'^2}\ (P')\ dP'\ + \right.$$

$$\left. \int_0^\infty\ P'\gamma' f_m\ (P')\ dP'\right) \tag{15}$$

$$F(P) = \frac{4\pi}{3} \left(\frac{\gamma^2}{P^2} \int_o^P P'f_m(P')\ (3v' - v'^3\ \beta_{th}^2)\ dP' +\right.$$

$$\left.\frac{2P}{\gamma} \int_o^\infty P'f_m(P')\ \beta_{th}^2\ dP'\right) \tag{16}$$

The finite element code TWODEPEP [2] solves the 2-D partial differential equation

$$P\frac{\partial U}{\partial t} = \frac{\partial Q_x}{\partial x} + \frac{\partial Q_y}{\partial y} + R \tag{17}$$

Comparing Eq. (17) with Eq. (11) we can identify P → x, $\mu \rightarrow y$, U → f and

$$Q_x = P^2 S_P \tag{18}$$

$$Q_y = - P \surd(1-\mu^2)\ S\mu \tag{19}$$

The momentum-space domain in which the equation is solved is $0 < P \leq P_{max}$ and $-1 \leq \mu \leq 1$. The condition $S_\mu = 0$ at $\mu = \pm 1$ follows from Eq. (13), and represents an axisymmetric situation. At $P = P_{max}$ we always set $f = 0$, although one should impose a runaway flux condition $S_P = \varepsilon\mu f$. With $f = 0$, however, the code will maintain required accuracy with considerably less computational effort in terms of required elements and memory. The $f = 0$ boundary condition at $P = P_{max}$ will distort the solution only in a narrow boundary layer close to P_{max}. To avoid the singularity at

$P = 0$, we impose the boundary condition $f = f_m$ (f_m is defined in Eq. (8)) at $P = P_{min}$ (P_{min} could be made very small, say 0.01). The imposition of f at $P = P_{min}$ will slightly violate density conservation in the bulk when ε and/or D are non-zero, but this effect is very small since for the values of ε and D considered the number of particles transferred to the tail is orders of magnitude smaller than the number of particles in the bulk. Thus when $\varepsilon \neq 0$, an imposed fixed boundary condition in the bulk represents an equilibrium situation in which electrons are being fed into the distribution at the same rate at which they run away. Hence we assume steady-state and solve for the equilibrium:

$$\frac{\partial Q_x}{\partial x} + \frac{\partial Q_y}{\partial y} = 0 \tag{20}$$

The functions Q_x and Q_y are defined in Eqs. (18) and (19) (see subroutines FXX and FXY in the Appendix, in which U → f, UX → $\frac{\partial f}{\partial x}$ and UY → $\frac{\partial f}{\partial y}$). tion (20) is a linear equation for f and TWODEPEP requires only one iteration to calculate the solution (parameter TF = 1 in the main, and NOUT = 1 denotes that the output should be given after one iteration). Also in the main NEQ = 1 indicates we are solving one equation, NTF = 9300 denotes the number of elements we are using and NDIM = 2 is a memory storage parameter. With these parameters the code would require a memory close to 5.5 million bytes on an IBM machine using double precision arithmetic. OXX and OXY are the functions Q_x and Q_y (defined in Eqs. (18) and (19)) which are calculated in the functions FXX and FXY respectively. D3EST is a function which distributes the elements in such a way as to have more elements in regions where more precision is required. UPRINT is the output solution given by U $\equiv$ f. The output solution is given on a grid defined by x = XA + i * H_x (i = 0, ... NX) and y = YA + j * HY (j = 0, ... NY). MWR = 8 is the file number on which the results are stored, and the INTEGRAL function integrates a function of the solution over the domain where the solution is calculated. In the Appendix presented, we are calculating the rf power absorbed defined by:

$$P_d = -2\pi \int_{-1}^{1} d_\mu \int_{o}^{\infty} D \frac{\partial f}{\partial P_\parallel} (P^3/\gamma) \, dP \tag{21}$$

The remaining parameters in the main define the boundary conditions. For more details see Ref. [2]. Finally, the calculations of the integrals in Eqs. (14) - (16) has been simplified by noting that for small values of P, and for the small values of H we are considering, the quantity v = P/γ $\simeq$ v and the integrals in Eqs. (14) - (16) can be calculated from well-known error functions integrals [1], while for large P, we have the asymptotic expressions:

$$A(P) = \frac{1}{v^3} \tag{22}$$

$$B(P) = \frac{1}{2v} (1 - \frac{1}{v^2}) \tag{23}$$

$$F(P) = \frac{1}{v^2} \tag{24}$$

We present in the following two sample results obtained using the program in the Appendix.

a) The runaway distribution function:

The results presented in this case are with $\varepsilon = 0.04$, $D = 0$ and $\beta_{th} = 0.0442$ (i.e. 1 kV plasmas). In Fig. (1), we plot the perpendicular velocity moment of the distribution function f, defined by

$$F = 2\pi \int_{o}^{\infty} dP_{\perp}P_{\perp}f \qquad (25)$$

as a function of $P_{\parallel}$. The function F is the parallel distribution function. It shows clearly the formation of a runaway tail. We show in Fig. (2) perpendicular cuts of f, indicating the presence of a hot tail population, a result similar to what has been recently presented for the non-relativistic case [5]. The flux Γ_R of electrons through the surface of a sphere of radius R is defined by

$$\Gamma_R = - \frac{\partial}{\partial t} \int d^3Pf = \int d^3P\mathrm{div}\vec{S} = 2\pi P^2 \int_{-1}^{1} d\mu S_P \qquad (26)$$

For the present case ($\varepsilon = 0.04$) the result for Γ_R is 0.14×10^{-5}. This is in very good agreement with the value 0.13×10^{-5} obtained from the expression [7]:

$$\Gamma_R = \Gamma_{NR} \exp \left\{ - \beta_{th}^2 \left[\frac{1}{8\varepsilon^2} + \frac{2}{3\varepsilon^3/2} (1 + Z)^{1/2} \right] \right\} \qquad (27)$$

where Γ_{NR} is the runaway production rate for the non-relativistic case (Γ_{NR} = 0.196 x 10^{-5} for ε = 0.04). Table I summarizes some of the results obtained for different values of ε for a 1 kV plasma (β_{th}= 0.0442). J_R denotes the relativistic current (in units of env_t)

$$J_R = \int \; v_{\parallel} \; fd^3P \tag{28}$$

calculated for different values of the normalized electric field ε and Z. Γ_{NR} and Γ_R are the runaway production rates calculated for the non-relativistic case and the relativistic case respectively and in the last column Γ_R is calculated from Eq. (27), showing a very good agreement with the value in the previous column. Finally, we show in Fig. (3) contour-plots for the 2-D solution f obtained for ε = 0.04.

b) Lower-Hybrid Current Drive.

We consider the case when ε = 0 and the quasilinear diffusion term is given by:

$$D = \begin{cases} 3 & \text{for } v_1 < P_{\parallel}/\gamma < v_2 \\ 0 & \text{otherwise} \end{cases} \tag{29}$$

D ≠ 0 inside a region bounded by the hyperbolae whose equations are given by

$$P_{\parallel}{}^2 \left(\frac{1}{v_{1,2}{}^2} - \beta^2_{th}\right) - P_{\perp}{}^2\beta^2_{th} = 1 \tag{30}$$

Figure (4) shows a contour plot of f obtained for the case $v_1 = 4$, $v_2 = 7$ and $\beta_{th} = 0.1$. The region where $D \neq 0$ is clearly shown to be bounded by the hyperbola whose equations is given by Eq. (30). Figure (5) shows a plot of the parallel distribution function F as defined in Eq. (25). A plateau formation is apparent in the region where $D \neq 0$. The current associated with this distribution is $J_R = 0.45 \times 10^{-2}$ and the power absorbed as defined in Eq. (21) is 1.57×10^{-4} resulting in a figure of merit $J_R/p_d = 28$.

To conclude, using TWODEPEP library and a Fortran code of no more than a few dozen lines, we have obtained results for the numerical solution of the linearized Fokker-Planck equation which compares favourably with what has been previously reported in the literature for the numerical solution of this problem [1, 6]. We are now in the process of extending these results to higher plasma temperature and to velocity spectra extending to higher relativistic speeds. In this latter case, however, we have been limited by the computer memory which for the present runs is close to 5.5 million bytes. Typical runs would take 30 min CPU time on the IBM 3084. Higher relativistic speeds will require a larger number of elements and consequently also a larger memory for the library. The use of cubic elements will require an even larger memory.

TABLE I

ε	Z	10^2 J_R	Γ_{NR}	Γ_R	Γ_R (EQ. 27)
0.005	1	1.9	0.76×10^{-30}	0.46×10^{-38}	0.3×10^{-36}
0.01	1	3.8	0.13×10^{-16}	0.18×10^{-18}	0.18×10^{-18}
0.03	1	12.7	0.83×10^{-7}	0.5×10^{-7}	0.45×10^{-7}
0.04	1	18.8	0.196×10^{-5}	0.14×10^{-5}	0.13×10^{-5}
0.04	4	8.25	0.158×10^{-6}	1.02×10^{-7}	0.94×10^{-7}

REFERENCES

1. C.F.F. Karney, "Fokker-Planck and Quasilinear Codes," Report PPPL-2290 (Princeton), 1985.

2. TWODEPEP, IMSL Problem-Solving Software System for Partial Differential Equations (IMSL, Houston 1983), IMSL No. TDP-0005, Edition 5.

3. V. Fuchs, R.A. Cairns, M.M. Shoucri, K. Hizanidis and A. Bers, Phys. Fluids 28, (1985) 3619.

4. V. Fuchs, R.A. Cairns, C.N. Lashmore-Davies and M.M. Shoucri, Phys. Fluids 29, (1986) 2931.

5. M. Shoucri, V. Fuchs and R.P. Gupta "Numerical Solution of a Two-Dimensional Fokker-Planck Equation" Report IREQ, Varennes (Nov. 1983) M. Shoucri, V. Fuchs, K. Hizanidis and A. Bers "Enhancement of Electron Runaway and Toroidal Current in the presence of Lower Hybrid Waves" Report Tokamak de Varennes, TRI 165 (December 1984); V. Fuchs, M. Shoucri, A. Bers and R.A. Cairns "Lower Hybrid Current Drive in the Presence of an Ohmic Electric Field" Report Tokamak de Varennes TVRI 187 (July 1985).

6. C.F.F. Karney and N.J. Fisch, Phys. Fluids 28, (1985) 116.

7. J.W. Connor, R.J. Hastie, Nucl. Fusion 15, (1975) 415.

Appendix

```
        1  9300 2
QXX     FXX(X,Y,U,UX,UY,FCT)                          → Function Qx in Eq.(18)
QXY     FXY(X,Y,U,UX,UY)                              → Function Qy in Eq.(19)
QBEST   WEIGHT(X,Y)
UPRINT    U
NX      200
NY      50
HX      0.30
HY      0.04
XA      0.1
YA      -1.
NOUT    1
TF      1
MWR     8
INTEGRALFNCT(X,Y,U,UX,UY,NOUT,T)
ARC=-1                                                }
FB1     FB(X,Y)                                       }
ARC=-3                                                }
FB1     0.                                            }
ARC=2                                                 }  Define boundary
GB1     0.                                            }  curve and
ARC=4                                                 }  boundary conditions
GB1     0.                                            }
XGRID   0.1,60.1                                      }
YGRID    -1.,1.                                       }
IX       -1,-3                                        }
IY       2,4
ADD.
      DOUBLE PRECISION FUNCTION FXX(X,Y,U,UX,UY,FCT)
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/D1/Z,C,PI,EPS,D0,VP1,VP2,FACT,BTH,DEN,TAU
      GAMA=DSQRT(1.+X*X*BTH*BTH)
      VEL=X/GAMA
      S=VEL/DSQRT(2.D0)
      VPAR=X*Y/GAMA
      D=D0
      IF(VPAR.LT.VP1.OR.VPAR.GT.VP2) D=0.D0
      SD=D*(Y*UX+(1.-Y*Y)*UY/X)
      SE=EPS*U
      FU=(S*DEXP(-S*S)+Z*S*DEXP(-S*S*C*C)/C)*2./DSQRT(3.1416D0)
      IF(X.LE.3.2)
     $BU=DERF(S)-FU+Z*DERF(C*S)/C**2
      IF(X.GT.3.2) BU=1.
      AA=BU/VEL**3
      FF=BU/VEL**2
      FXX=X*X*Y*(SE-SD)-(AA*UX+FF*U)*X*X      → Define Qx in Eq. (18)
```

```
 FCT=FXX*X/GAMA
 RETURN
 END
 DOUBLE PRECISION FUNCTION FXY(X,Y,U,UX,UY)
 IMPLICIT REAL*8(A-H,O-Z)
 COMMON/D1/Z,C,PI,EPS,D0,VP1,VP2,FACT,BTH,DEN,TAU
 GAMA=DSQRT(1.+X*X*BTH*BTH)
 VEL=X/GAMA
 S=VEL/DSQRT(2.D0)
 VPAR=X*Y/GAMA
 D=D0
 IF(VPAR.LT.VP1.OR.VPAR.GT.VP2) D=0.D0
 SD=D*(Y*UX+(1.-Y*Y)*UY/X)
 SE=EPS*U
 FU=(S*DEXP(-S*S)*FACT
$                   +Z*S*DEXP(-S*S*C*C)/C)*2./DSQRT(3.1416D0)
 FU2=1.-1./(2.*S*S)
 FU2C=1.-1./(2.*S*S*C*C)
 IF(X.LE.3.2)
$AU=DERF(S)*FU2*FACT+Z*DERF(C*S)*FU2C+FU/(2.*S*S)
 IF(X.GT.3.2) AU=1.-1./VEL**2+Z
 BB=AU/(2.*VEL)
 FXY=-X*(1.-Y*Y)*(BB*UY/X-SE+SD)          ————> Define Qy in Eq. (19)
 RETURN
 END
 DOUBLE PRECISION FUNCTION FNCT(X,Y,U,UX,UY,NOUT,T)
 IMPLICIT REAL*8(A-H,O-Z)
 COMMON/D1/Z,C,PI,EPS,D0,VP1,VP2,FACT,BTH,DEN,TAU
 FNCT=0.D0
 NOU=NOUT+1
 NOUM=NOU+1
 GAMA=DSQRT(1.+X*X*BTH*BTH)
 VPAR=X*Y/GAMA
 VEL=X/GAMA
 S=VEL/DSQRT(2.D0)
 FU=(S*DEXP(-S*S)+Z*S*DEXP(-S*S*C*C)/C)*2./DSQRT(3.1416D0)
 IF(X.LE.3.2) BU=DERF(S)-FU+Z*DERF(C*S)/C**2
 IF(X.GT.3.2) BU=1.
 AA=BU/VEL**3
 FF=BU/VEL**2
 D=D0
 IF(VPAR.LT.VP1.OR.VPAR.GT.VP2) D=0.D0
 FNCT=-D*X*Y*(Y*UX+(1.-Y*Y)*UY/X)*2.*3.1416*X*X/GAMA
 FETURN
```

Function to
be integrated
at the end

```
      END                                                      }
      DOUBLE PRECISION FUNCTION FB(X,Y)                        }
      IMPLICIT REAL*8(A-H,O-Z)                                 }  Define boundary
      COMMON/D1/Z,C,PI,EPS.DO,VP1,VP2,FACT,BTH,DEN,TAU         }  condition for
      GAMA=DSQRT(1. +X*X*BTH*BTH)                              }  ARC = -1
      FB=0.0634936*DEXP(-X*X/(1.+GAMA))                        }
      RETURN                                                   }
      END
      DOUBLE PRECISION FUNCTION WEIGHT(X,Y)
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/D1/Z,C,PI,EPS,DO,VP1,VP2,FACT,BTH,DEN,TAU
      V1=VP1-1.
      V2=VP2+1.
      IF(EPS.NE.0.) VCRIT=DSQRT(1./EPS)
      GAMA=DSQRT(1.+X*X*BTH*BTH)
      VPAR=X*Y/GAMA
      WDO=2.
      WEO=2.
      WD=0.
      WE=0.
      IF(EPS.NE.0..AND.VPAR.GT.VCRIT) WE=WEO
      IF(DO.NE.0..AND.(VPAR.GT.V1.AND.VPAR.LT.V2)) WD=WDO
      WEIGHT=1.+WD+WE
      RETURN
      END
      BLOCK DATA
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/D1/Z,C,PI,EPS,DO,VP1,VP2,FACT,BTH,DEN,TAU
      DATA Z/1.DO/,C/42.9DO/,PI/3.14159265DO/,FACT/1.0DO/
      DATA EPS/0.00/,DO/3.0DO/,VP1/4.00DO/,VP2/16.0DO/,BTH/0.054DO/
      DATA DEN/0.5DO/,TAU/15.0D-3/
      END
END.
```

FIGURE CAPTIONS

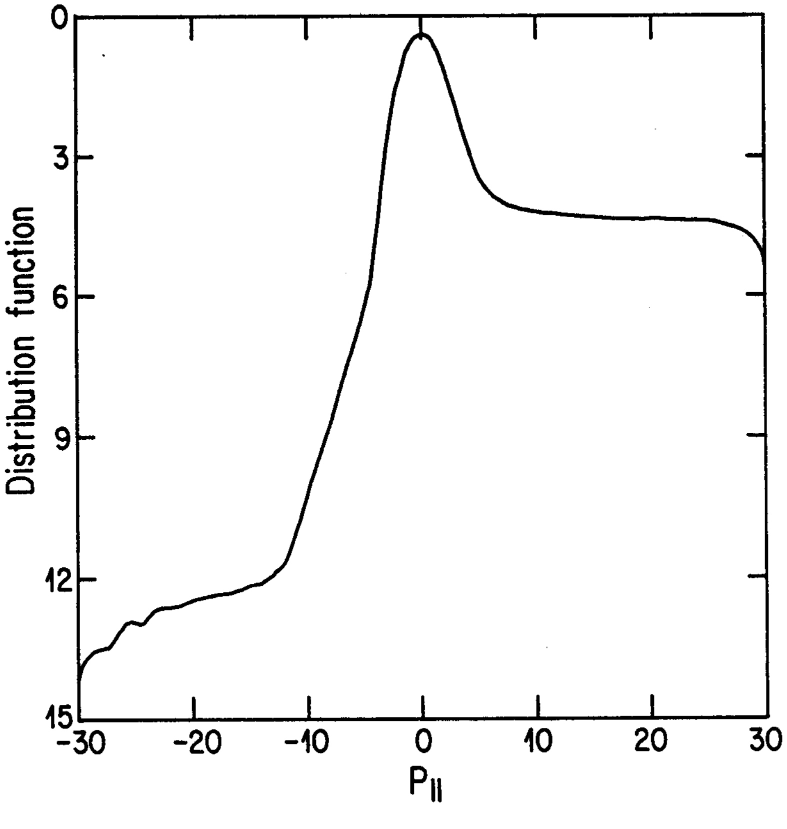
Figure 1 The perpendicular velocity moment of the distribution function f, defined by Eq. (25), for the runaway problem.

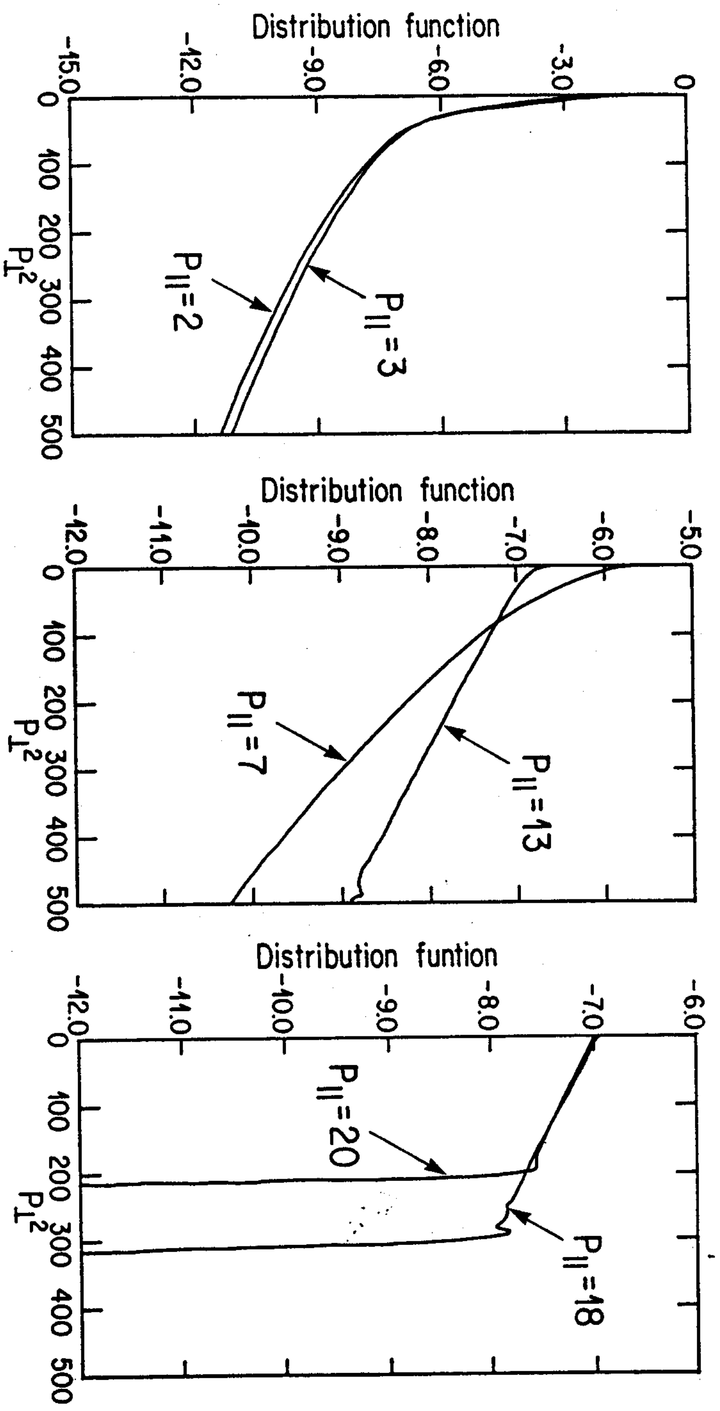
Figure 2 Perpendicular cuts of the distribution function f at different values of $P_{\parallel}$, for the runaway problem.

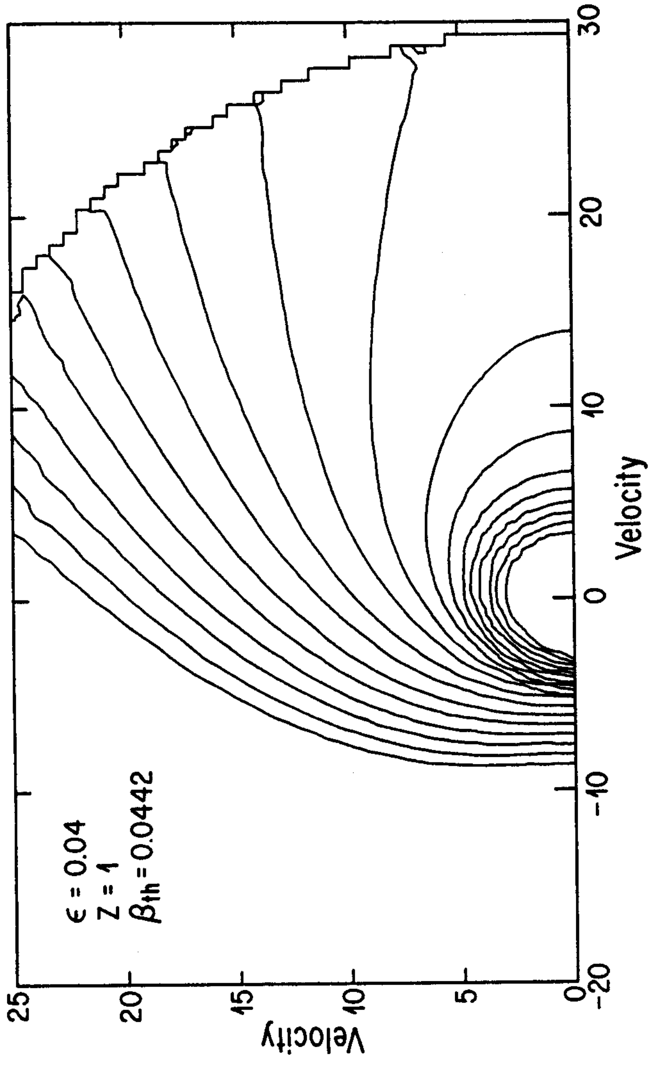
Figure 3 Contourplots of the distribution function f, for the runaway problem.

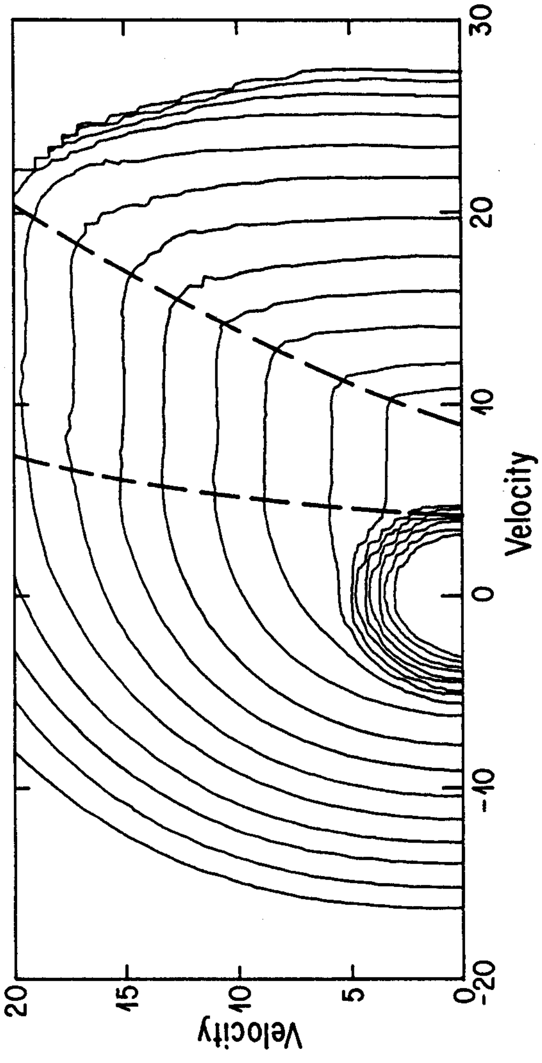
Figure 4 Contourplots of the distribution function f for the RF problem.

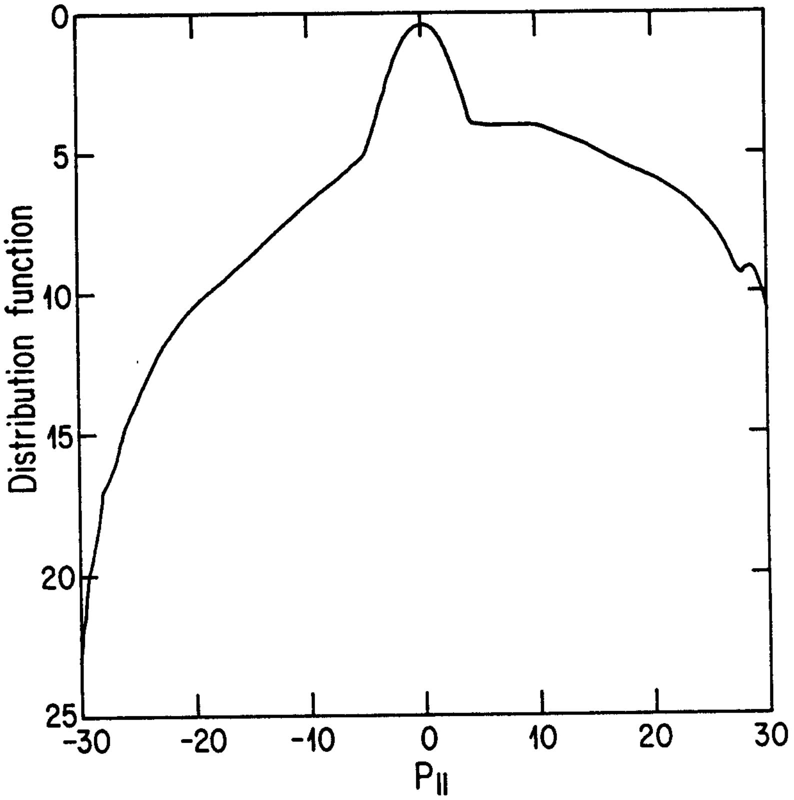
Figure 5 The perpendicular velocity moment of the distribution function f, defined by Eq. (25), for the RF problem.


0
3
6
9
12
15
Distribution function
-30
-20
-10
0
10
20
30
$P_{\parallel}$


Distribution function
-15.0
-12.0
-9.0
-6.0
-3.0
0
0
100
200
300
400
500
$P_\perp^2$
$P_{\|}=2$
$P_{\|}=3$
Distribution function
-12.0
-11.0
-10.0
-9.0
-8.0
-7.0
-6.0
-5.0
0
100
200
300
400
500
$P_\perp^2$
$P_{\|}=7$
$P_{\|}=13$
Distribution funtion
-12.0
-11.0
-10.0
-9.0
-8.0
-7.0
-6.0
0
100
200
300
400
500
$P_\perp^2$
$P_{\|}=20$
$P_{\|}=18$


$\epsilon = 0.04$
$Z = 1$
$\beta_{th} = 0.0442$
Velocity
Velocity
-20
-10
0
10
20
30
0
5
10
15
20
25


Velocity
Velocity
-20
-10
0
10
20
30
0
5
10
15
20


0
5
10
15
20
25
Distribution function
-30
-20
-10
0
10
20
30
$P_{\parallel}$