

# A particle simulation code for analysis of nonlinear electron oscillations in a magnetized plasma waveguide

**Turikov, V.A.**

*Publication date:*
1978

*Document Version*
Publisher's PDF, also known as Version of record

RISØ

Risø-M-2116

Title and author(s)

# A Particle Simulation Code for Analysis of Nonlinear Electron Oscillations in a Plasma Waveguide

by

V.A. Turikov

Date August 1978

Department or group

Physics

Group's own registration number(s)

21 pages + tables + 5 illustrations

Abstract

A description is given of a computer code for simulation of electron oscillations in a magnetized plasma in a cylindrical waveguide. The one-dimensional particle-in-cell method with the reverse linear interpolation of charge density is used. The program has options for treating nonlinear processes in a plasma with periodical and reflecting boundary conditions. For periodical conditions, Poisson's equation is solved by means of the Fourier method. For reflecting conditions, the double recursive procedure is used. The values of the potential derivatives at the space grid points are calculated by means of the parabolic interpolation. The main purpose of the program is to investigate nonlinear phenomena in a plasma column after applying a short localized impulse of an external electric field.

Copies to

Available on request from Risø Library, Risø National Laboratory (Risø Bibliotek, Forsøgsanlæg Risø), DK-4000 Roskilde, Denmark
Telephone: (03) 35 51 01, ext. 334, telex: 43116

ISBN 87-550-0594-1

ISBN 87-550-0538-1

ISSN 0418-6435

CONTENTS

## 1. Introduction

Strongly nonlinear processes in a collisionless plasma are being intensively investigated at the present time both experimentally and theoretically. The Q-machine is a very suitable device for treating such general plasma properties. In recent experiments[1),2)] in a single-ended Q-machine with a magnetized plasma in a cylindrical waveguide, formation of various kinds of stable pulses due to localized external excitations was observed. These pulses have been identified as solitons, rarefactive waves and phase-space holes similar to the BGK-modes[3)]. It is very difficult to investigate such kinds of problem theoretically because one has to solve the nonlinear system of Vlasov and Poisson equations with a strong external perturbation.

The simulation code SIMULA has developed to investigate nonlinear electron oscillations in a magnetized cylindrical plasma column to analyze the experimental results from the Q-machine plasma. Computer calculations performed with this code[2),4)] have confirmed the fact that stable positive pulses of the potential propagating in a plasma after an external impulse in axial direction appear as vortexes or "holes" in phase space, like the vortexes forming in the two-stream instability[5),6)]. Also some important properties of solitons and rarefactive pulses have been studied by means of these computer simulations.

The particle-in-cell simulation scheme[7),8)] with the reverse linear interpolation method[5)] of charge density calculation is used in the simulation code. The program is written in FORTRAN and runs have been made on a Burroughs 6700 at the Risø Computer Installation. The purpose of this paper is to describe this simulation code.

## 2. Physical Assumptions and Equations

The computer program described here simulates the properties of a magnetized plasma in a cylindrical waveguide. We assume infinite axial magnetic field, therefore electrons can move only in the axial direction (x-direction), and we can consider the problems to be one-dimensional. The ions are considered as a stationary positive background.

Space-charge waves (azimuthally symmetric modes) in a cylindrical waveguide have the maximum phase velocity for small wavenumbers[9)]

$$v_{ph} = \frac{\omega_{pe}}{k_\perp}, \tag{1}$$

where $k_\perp = 2.404/r_o$, $r_o$ is the radius of the plasma column, and $\omega_{pe}$ is the electron plasma frequency.

For the azimuthally symmetric modes, the potential can be calculated from Poisson's equation

$$\frac{d^2\phi}{dx^2} - k_\perp^2\phi = -4\pi\rho. \tag{2}$$

We assume boundary conditions with a zero electric field at the ends of the plasma column

$$\left.\frac{d\phi}{dx}\right|_{x=0} = \left.\frac{d\phi}{dx}\right|_{x=L} = 0, \tag{3}$$

where L is the column length.

The electric field energy can be calculated using Poisson's equation (2)

$$W_E = \frac{\pi r_0^2}{8\pi}\int (E_x^2 + k_\perp^2\phi^2)dx, \tag{4}$$

where $E_x$ is the electric field in the plasma.

We assume that the localized impulse of the external potential can be presented in the form:

$$e\phi_{ext}(x,t) = W_{ph}\cdot\eta(x)\sigma(t), \tag{5}$$

where the functions $\eta(x)$, $\sigma(t)$ describe the space and time variations of the external potential, and $W_{ph}$ is the kinetic energy of the electron with the phase velocity (1). The concrete profiles of $\eta(x)$ and $\sigma(t)$ used in the simulation are presented in section 5.9.

## 3. General Algorithm of the Simulation Scheme

The particle-in-cell method (PIC method) is employed in the simulation code. In this method it is convenient to use dimensionless variables defined by the following relations[8)]

$$X = \frac{x}{\Delta}, \qquad V = v\frac{2\Delta t}{\Delta},$$

$$E = -E_x \frac{4(\Delta t)^2}{\Delta} \frac{e}{m_e}, \qquad F = \phi \cdot \frac{2(\Delta t)^2}{\Delta^2} \frac{e}{m_e}, \tag{6}$$

$$Q = -G \frac{\rho}{e n_o}, \qquad G = 2\omega_{pe}^2 (\Delta t)^2,$$

where $\Delta$ is the space step, $\Delta t$ is the time step, and $n_o$ is the initial plasma density.

The leap-frog scheme[8] is used for moving all simulation particles the time step $2\Delta t$:

$$V_m^{k+1} = V_m^{k-1} + E^k(x_m^k) + E_{ext}^k(x_m^k),$$

$$x_m^{k+2} = x_m^k + V_m^{k+1}, \qquad m = 1,2,\ldots N, \tag{7}$$

where $k$ is the time step number, $N$ is the total number of simulation particles, and $E_{ext}(x)$ is the external electric field.

The first time step is realized by the Euler method[8]

$$x_m^1 = x_m^o + V_m^o/2. \tag{8}$$

After the calculation of the coordinates of all particles at moment $t^k = k\Delta t$ we can obtain the values of the charge density $Q_i^k$ in the nodes of the space grid. The charge density is calculated by distributing the charge of each particle between the two nearest grid points according to a reverse linear interpolation[5]

$$Q_i^k = G\,[N_i(t^k)/N_c - 1],$$

$$N_i(t^k) = \sum_{m=1}^{N} \delta(\mathrm{Int}|x_m^k - i|) \cdot (1 - |x_m^k - i|), \tag{9}$$

where $N_i(t^k)$ is the number of charges in e units in grid point i at time moment $t^k$, $N_c$ is the initial number of particles in each cell, Int(z) is a truncation from real value to integer and

$$\delta(m) = \begin{cases} 1, & m = 0 \\ 0 & m \neq 0. \end{cases}$$

Having the charge density distribution $Q_i^k$, we can calculate the new values of the self-consistent electric field in eqs. (7) used to move all particles the next time step. For the electric field calculation, Poisson's equation is solved with the boundary conditions (3) (or with periodical conditions) and charge density values $Q_i^k$ using the three-point difference scheme

$$F_{i+1}^k - (2-k_1^2\Delta^2)\, F_i^k + F_{i-1}^k = Q_i^k. \tag{10}$$

Equations (7)-(10) define the general time loop of the PIC method used in the simulation code.

If a simulation particle is outside the interval $0 \le X \le J$ ($J = L/\Delta$) in some time step and reflecting boundary conditions (3) are used, the coordinate and the velocity of this particle are altered according to:

$$X_{m,\mathrm{refl}}^k = \begin{cases} -X_m^k, & X_m^k < 0, \\ 2J - X_m^k, & X_m^k > J, \end{cases}$$

$$V_{m,\mathrm{refl}}^k = -V_m^k.$$

When the periodical conditions are used, the coordinates and velocities of such particles are changed in the following way:

$$X_{m,\mathrm{per}}^k = \begin{cases} J + X_m^k, & X_m^k < 0, \\ X_m^k - J, & X_m^k > J, \end{cases}$$

$$V_{m,\mathrm{per}}^k = V_m^k.$$

We assume the initial distribution of the simulation particles is homogeneous in x-space and random in v-space with Gaussian statistics (see sec. 5.7)

The electric field energy (4) and the total plasma energy calculated per one particle are computed during the simulated process

$$W_E = \frac{1}{2GJ} \int_0^J \left[ (E + E_{ext})^2 + 4k_1^2\Delta^2 F^2 \right] dX,$$

$$W_{\mathrm{total}} = W_E + \frac{1}{N} \sum_{m=1}^{N} V_m^2. \tag{11}$$

We calculate the electric field in an arbitrary point X by means of the expression:

$$E(X) = -\frac{dF}{dX} = -(2A_jX + B_j),$$

where $A_j$ and $B_j$ are coefficients of the quadratic interpolation of the potential between the three nearest space grid points

$$F(X) = A_jX^2 + B_jX + C_j, \qquad j = \mathrm{Int}(X)+1.$$

For the calculation of the integral $\int_0^J F^2 dX$ in formula (11), we use parabolic interpolation for the values $F_i^2$.

It appeared during the simulations that the total energy conserves after the action of the external field with an error of about 1-2%, which means that the model can be considered quite accurate.

The general flow of the program is presented in Figs. 1a, 1b, and concrete features of the main program SIMULA are discussed in sec. 5.

## 4. Algorithm for Solving Poisson's Equation

There are two options for solving Poisson's equation in the simulation code - for periodical an nonperiodical conditions.

General nonperiodical boundary conditions can be written in the form:

$$\left(A_l \frac{dF}{dX} + B_lF\right)\Big|_{X=0} = C_l, \tag{12}$$

$$\left(A_r \frac{dF}{dX} + B_rF\right)\Big|_{X=J} = C_r, \tag{13}$$

where $A_l$, $B_l$, $C_l$, $A_r$, $B_r$, and $C_r$ are prescribed constants.

For solving Poisson's equation with boundary conditions (12) and (13), the double recursive procedure is employed[8] with the parabolic interpolation of the potential for calculating $\frac{dF}{dX}$ at the ends of the plasma column:

$$F_j = c_j F_{j-1} + s_j, \quad j = 1, 2, \ldots, J,$$

where the coefficients $c_j$ and $s_j$ are also defined by the recurrence relations:

$$c_{j-1} = \frac{1}{c_j - k_1^2\Delta^2 - 2},$$

$$s_{j-1} = \frac{Q_{j-1} - s_j}{c_j - k_1^2\Delta^2 - 2}, \qquad j = J, J-1, \ldots 2.$$

The value $F_o$ can be found using the boundary condition (12) for the left boundary of the interval

$$F_o = \frac{C_1 + A_1(c_2 s_1 - s_2 - 4s_1)/2}{A_1(2c_1 - c_1 c_2/2 - 3/2) + B_1}.$$

For the periodical boundary conditions, we use the Fourier method for solving the Poisson equation[8),10)]:

$$F_j = \sum_{i=1}^{J_2} \left(\hat{F}_i^c \cos\frac{2\pi ij}{J} + \hat{F}_i^s \sin\frac{2\pi ij}{J}\right), \tag{14}$$

where $J_2 = (J-1)/2$ for odd J and $J_2 = J/2$ for even J, and $\hat{F}_i^c$, $\hat{F}_i^s$ are the cosine and sine Fourier coefficients. These coefficients can be obtained by means of a Fourier analysis of the Poisson equation (10):

$$\hat{F}_i^c = \frac{\hat{Q}_i^c}{\lambda_i}, \qquad \hat{F}_i^s = \frac{\hat{Q}_i^s}{\lambda_i},$$

$$\lambda_i = 2\left(\cos\frac{2\pi i}{J} - 1 - \frac{k_1^2\Delta^2}{2}\right), \qquad i = 1,2,\ldots,J_2, \tag{15}$$

where $\hat{Q}_i^c$, $\hat{Q}_i^s$ are the cosine and sine Fourier coefficients of the charge density

$$\hat{Q}_i^c = \frac{2}{J} \sum_{m=1}^{J} Q_m \cos\frac{2\pi mi}{J},$$

$$\hat{Q}_i^s = \frac{2}{J} \sum_{m=1}^{J} Q_m \sin\frac{2\pi mi}{J}, \qquad i = 1,2,\ldots,J_2 \text{ (odd J)}$$

For even J, the coefficient $\hat{Q}_{J/2}^c$ can be written in the form:

$$\hat{Q}_{J/2}^c = \frac{1}{J} \sum_{m=1}^{J} Q_m \cos\pi m = \frac{1}{J} \sum_{m=1}^{J} Q_m (-1)^m$$

The coefficient $\hat{Q}_{J/2}^s$ is equal to zero as $\sin\pi m = 0$.

Hence the Fourier procedure for solving Poisson's equation consists of two steps. At first, we perform the Fourier analysis

of the charge density distribution to calculate the coefficients $\hat{F}_i^c$, $\hat{F}_i^s$ by means of (15), and then we perform the Fourier synthesis to find the solution of Poisson's equation using the expansion (14). In the main program SIMULA the values $\cos\frac{2\pi i}{J}$, $\sin\frac{2\pi i}{J}$ for $i = 1,2,\ldots,J$ are prepared in the case of periodical conditions to realize the fast Fourier transform.

The FORTRAN listing of the subroutine PMSOLV for solving Poisson's equation is presented in the Program Listings section.

## 5. Program Description

### 5.1. General Structure of the Program

The program performs a particle simulation of nonlinear plasma oscillations in a conducting waveguide with reflective boundary conditions and also simulates the two-stream instability as a test problem for periodical boundary conditions.

The program is divided into the main program SIMULA, the subroutines PMSOLV, INGEN, PCTRKV and the function EXTPOT. The parameters for solving the Poisson equation are transmitted to the subroutine PMSOLV by means of the COMMON-block |PSM|, and external potential parameters are transmitted to the function EXTPOT by the COMMON-block |EXTP|.

The general block diagram of the main program SIMULA is given in Figs. 1a and 1b. Detailed comments are given in the FORTRAN listing.

### 5.2. Program Input

The format of the input cards is noted in brackets.

Card 1 (I5): NC = initial number of the simulation particles in each space cell.

Card 2 (I5): J = number of cells.

Card 3 (I5): KF = number of time steps in the leap-frog scheme.

Card 4 (I5): KD = time step interval between outputs of the program

Card 5 (I5): NV = number of cases with different external potential parameters.

Card 6 (E10.3) : DL = $\frac{\Delta}{\lambda_D}$, where $\Delta$ is the space step and $\lambda_D$ is the Debye length.

Card 7 (E10.3) : DT = $\Delta t \cdot \omega_{pe}$, where $\Delta t$ is the time step and $\omega_{pe}$ is the electron plasma frequency.

Card 8 (E10.3) : DEN = initial plasma density in $cm^{-3}$.

Card 9 (E10.3) : E0 = initial plasma temperature in eV.

Card 10 (E10.3) : R0 = radius of the plasma column in cm.

Card 11 (E10.3) : UD = $v_d/v_{th}$, drift velocity of the initial v-distribution in thermal velocity units (for the case of two-stream instability).

Card 12 (5E10.3): XMIN, XMAX = boundary values of X for the phase space plot.

HX = value of the step in X units between two lines of the phase space plot.

UMIN, UMAX = boundary values of $v/v_{th}$ for the phase space plot and for the velocity distribution plot.

Card 13 (6E10.3): AL, BL, CL, AR, BR, CR = boundary conditions parameters $A_l$, $B_l$, $C_l$, $A_r$, $B_r$, $C_r$ (see eqs. (12), (13)) for the Poisson equation. If all these parameters are equal to zero the Poisson equation will be solved with periodical boundary conditions.

Card 14 (6E10.3): PE(1) - - - PE(6) = amplitudes and space parameters of the external potential (see sec. 5.9).

Card 15 (5E10.3): PE(7) - - - PE(11) = time parameters of the external potential (see sec. 5.9).

If the parameter NV > 1, pairs of cards similar to cards 14 and 15 have to be added with parameters for other cases of external potential. The total number of such pairs is equal to NV.

## 5.3 Program Output

After every KD time step, the total energy and the electrostatic energy (in dimensionless units, see sec. 3) are printed. If XMAX > XMIN the phase-space plot and the velocity distribution function are drawn by the line-printer. When XMAX $\leq$ XMIN these output data will be omitted.

After the final time step, KF, for every case of the external potential, a plot of the temporary development of the potential space variation is drawn by the plotter. The potential is plotted in $A_p W_{ph}/e$ units, where $A_p = |\eta(x)|_{max}$ (see eq. (5)). The test run output for nonperiodical boundary conditions is shown in Figs. 2a, 2b.

## 5.4 Work Constants

N = total number of simulation particles.

G = $2\omega_{pe}^2(\Delta t)^2 = 2DT^2$ = normalization factor for the charge density calculation (see eqs. (6)).

LD = the Debye length $\lambda_D$ in cm.

CFØ = $W_{ph}\cdot\frac{2(\Delta t)^2}{m_e\Delta^2} = 4\cdot\left(\frac{DT}{DL}\frac{RØ}{LD}\right)^2\frac{1}{23.116}$ = normalization factor for the external potential calculation (see eqs. (5), (6)).

VN = $V\cdot v_{th}/v = 2\sqrt{2}\,\frac{DT}{DI}$ = velocity normalization factor for the initial distribution generation, where $v_{th}$ is the electron thermal velocity.

UTN = 2GJ )
DKE = $4k_\perp^2\Delta^2$ ) = factor for the electric field energy (8) calculation.

SF = max (|AP1|, |AP2|)·CFØ = scale factor for plotting the potential space variations, where AP1, AP2 are the external potential amplitudes (see sec. 5.9).

KBOUND = a sign for the type of boundary condition. KBOUND = 1 for periodical boundary conditions and KBOUND = 0 for reflecting boundary conditions.

JEV = 1 for even J and Ø for odd J.

J2 = $J_2$ = maximum index for the potential values Fourier synthesis (14).

C(1ØØØ) = $\cos\frac{2\pi j}{J}$ )
) = constants for the fast Fourier transform in a case of periodical conditions.
S(1ØØØ) = $\sin\frac{2\pi j}{J}$ )

## 5.5 Principle Variables

X(5ØØØØ) )
V(5ØØØØ) ) = arrays of the normalized coordinates and velocities of all particles in each time step.

Q(1ØØØ) )
F(1ØØØ) ) = arrays of the normalized charge density and potential in the space grid points X = 1,2,...,J.

FØ = value of the potential in the point X = Ø.

A(5ØØ) )
B(5ØØ) ) = arrays of the cosine and sine Fourier coefficients (15) for the Fourier method of solving the Poisson equation.

K = number of the time step in the leap-frog scheme (7).

NEF = number of cases with the concrete external potential parameters.

KTP = output interval counter (the program output is realized when KTP = KD).

T = DT•(2K-1)/2π = time moment in plasma periods, which corresponds to the time step K.

UT = normalized electric field energy per one particle (7).

ET = normalized thermal plasma energy per one particle.

## 5.6 Subroutine PNSOLV

Purpose: This subroutine solves the finite difference Poisson equation (10) for the dimensionless charge density values Q(X) in the points X = 1,2,...,J. The solution is presented by the values F(X) in the same points and the value FØ in the point X = Ø. If the parameter KBOUND = 1 (all parameters in card 13 are equal to zero), the Poisson equation is solved for periodical boundary conditions, and when KBOUND = Ø,

it is solved for the general nonperiodical conditions (12), (13) (see sec. 4). The subroutine obtains all parameters by means of the COMMON-block |PSN|.

Calling Sequence: CALL PNSOLV.

COMMON-block |PSN|: KBOUND, JEV (see sec. 5.4), AL, BL, CL, AR, BR, CR, J (see eqs. (12), (13) and sec. 5.2), J2, DKT, F0, F(1000), Q(1000), C(1000), S(1000), A(500), B(500) (see secs. 5.4, 5.5).

## 5.7 Subroutine INGEN

Purpose: This subroutine generates the initial values of the dimensionless coordinates X and velocities V for all simulation particles. The initial distribution is homogeneous in X-space and random in V-space, according to the Maxwellian distribution. For generation of the V-distribution, the Risø Computer Library subroutine NORMAR is used. When the parameter UD > 0, the subroutine creates a two-Maxwellian distribution in V-space with relative drift velocity UD·$v_{th}$.

Calling Sequence: CALL INGEN (X, V, N, J, VN, UD).

Parameters:

- X(N), V(N) = arrays of the dimensionless coordinates and velocity values generated by the subroutine.
- N = number of points in phase space generated by the subroutine.
- J = length of X-space interval.
- VN = velocity normalization factor (see sec. 5.4).
- UD = relative drift velocity in $v_{th}$ units of a two-Maxwellian distribution for the two-stream instability case.

## 5.8 Subroutine PCTRXV

Purpose: By means of the line-printer the subroutine plots a phase space distribution of the simulation particles and also a velocity distribution function by a simple histogram summing.

Calling Sequence: CALL PCTRXV (X, V, XMIN, XMAX, HX, UMIN, UMAX, VN).

Parameters:

| | | |
|---|---|---|
| X(N) ) | = | coordinates of the simulation particles |
| V(N) ) | | in phase space. |
| N | = | total number of particles. |
| XMIN ) | | |
| XMAX ) | | parameters of plots printed |
| HX ) | = | by the subroutine (see the |
| UMIN ) | | description of card 13 in sec. 5.2). |
| UMAX ) | | |
| VN | = | velocity normalization factor (see sec. 5.4). |

## 5.9 Function EXTPOT

Purpose: This function calculates the value of the external potential in a point X at a time moment T for studying the nonlinear plasma response to a short localized excitation. The external potential is normalized by the kinetic energy of an electron with velocity $v_{ph}$:

$$\text{EXTPOT}\ (X,T) = e\phi_{ext}/W_{ph} = \phi_{ext} \cdot \frac{2e}{m_e}\left(\frac{k_\perp}{\omega_{pe}}\right)^2 .$$

The function EXTPOT is constructed as a product of two functions:

$$\text{EXTPOT}\ (X,T) = \eta(X)\cdot\sigma(T) ,$$

where the time-dependent function $\sigma(T)$ is assumed to satisfy the requirements:

$$\sigma(T) \geq 0 , \qquad \max \sigma(T) = 1.$$

According to the conditions of the experiments with the Q-machine plasma[1),2)], the followint approximation for the impulse profile is used (see Fig. 3):

$$\eta(X) = \begin{cases} -AP1\ , & 0 \le X \le DX1\ , \\ 0\ , & 0 \le X - DX1 - DP1 \le DX2\ , \\ -AP2\ , & X - DX1 - DP1 \ge DX2 + DP2, \end{cases}$$

$$\sigma(T) = \begin{cases} 0\ , & T \le DT1\ , \\ 1\ , & 0 \le T - DT1 - DT2 \le DT3\ , \\ STF\ , & T - DT1 - DT2 \ge DT3 + DT4\ , \end{cases} \qquad (16)$$

where AP1, AP2 are the amplitudes of the external potential jumps in $W_{ph}/e$ units (see the expression (5)).

For the approximation of the space and time shapes of the external potential in the intervals between the regions mentioned in the definitions of $\eta(X)$ and $\sigma(T)$, half a cosine period is used with corresponding parameters to satisfy the relations (16).

External potential parameters are taken from the COMMON-block |EXTP|.

Calling Sequence: EXTPOT (X,T).

Arguments: X = space point coordinate in Δ units,
T = time moment in plasma periods.

COMMON-block |EXTP|:

| | | |
|---|---|---|
| PE(1) = AP1 | ) | |
| PE(2) = DX1/J | ) | parameters of the |
| PE(3) = DP1·Δ/RØ | ) | = space profile func- |
| PE(4) = AP2 | ) | tion $\eta(X)$ (see Fig.3). |
| PE(5) = DX2/J | ) | |
| PE(6) = DP2·Δ/RØ | ) | |
| PE(7) = DT1 | ) | |
| PE(8) = DT2 | ) | parameters oi the |
| PE(9) = DT3 | ) | = time profile func- |
| PE(1Ø) = DT4 | ) | tion $\sigma(T)$ (see Fig.3). |
| PE(11) = STF | ) | |

## 6. Acknowledgements

This work was performed while the author was a visitor at the Risø National Laboratory. He is very grateful to V.O. Jensen for hospitality and interest in this work. Likewise, he is much indebted to J.P. Lynov, P. Michelsen, H.L. Pécseli, J. Juul Rasmussen and K. Saeki for helpful discussions.

## 7. References


1) K. Saeki, P. Michelsen, H.L. Pécseli, J. Juul Rasmussen (to be published).
2) J.P. Lynov, P. Michelsen, H.L. Pécseli, J. Juul Rasmussen, K. Saeki, V.A. Turikov, Physica Scripta (to be published).
3) I.B. Bernstein, J.M. Green, and M.D. Kruskal (1957), Phys. Rev. 108, 546-550.
4) V.A. Turikov (1978), Risø Report Nr. 380 (to be published).
5) R.L. Morse and C.W. Nielson (1969), Phys. Fluids, 12, 2418-2425.
6) H.L. Berk, C.E. Nielsen and K.V. Roberts (1970), Phys. Fluids, 13, 980-995.
7) R.L. Morse (1970), Methods in Computational Physics, 9 (Academic Press), p. 213-239.
8) D. Potter (1973), Computational Physics, (Wiley, London), 304 pp.
9) A.W. Trivelpiece and R.W. Gould (1959), J.Appl.Phys., 30, 1784-1793.
10) R.W. Hockney (1970), Methods in Computational Physics, 9 (Academic Press), p. 135.

## 8. Figures

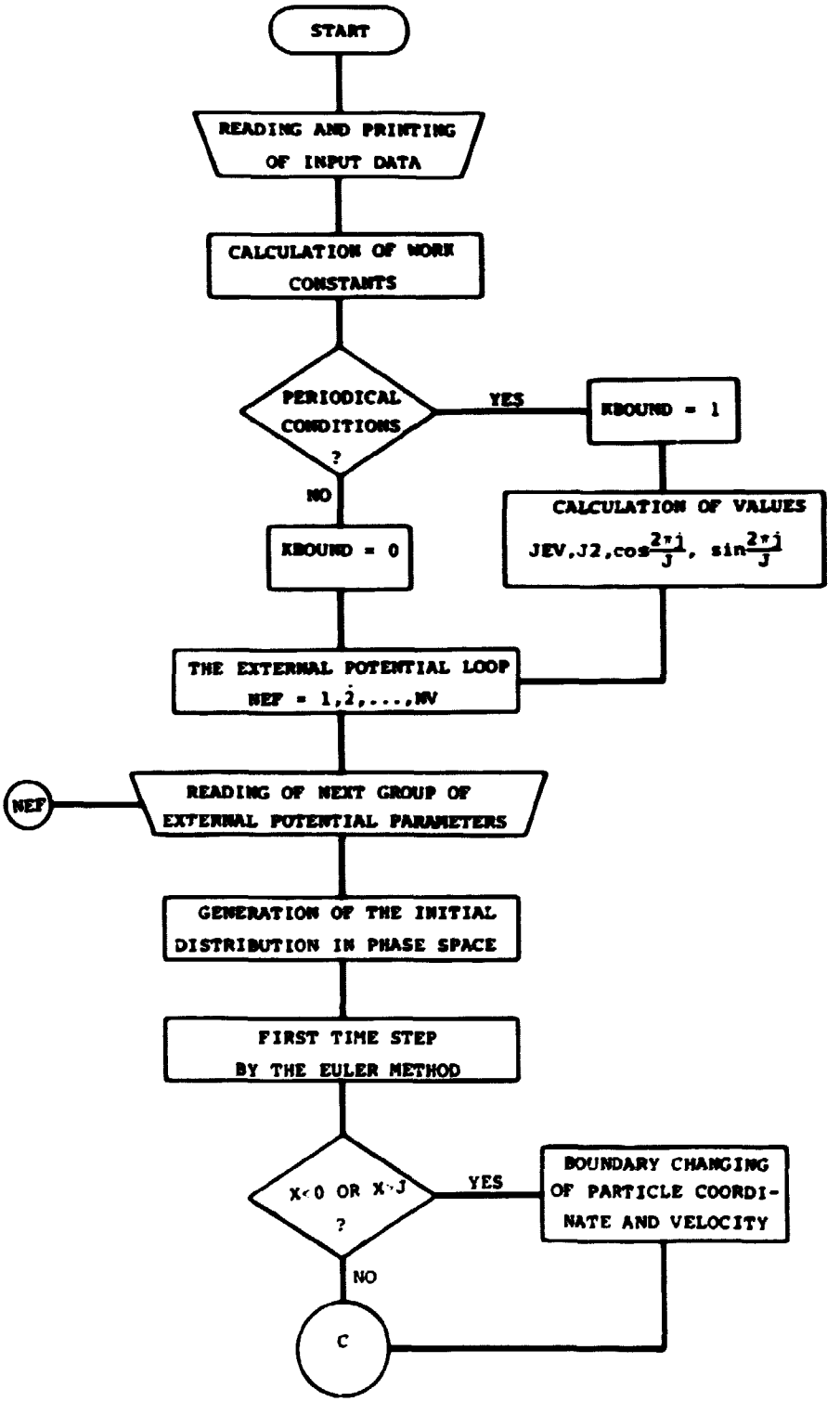


Fig. 1a. General Block Diagram of the Code.

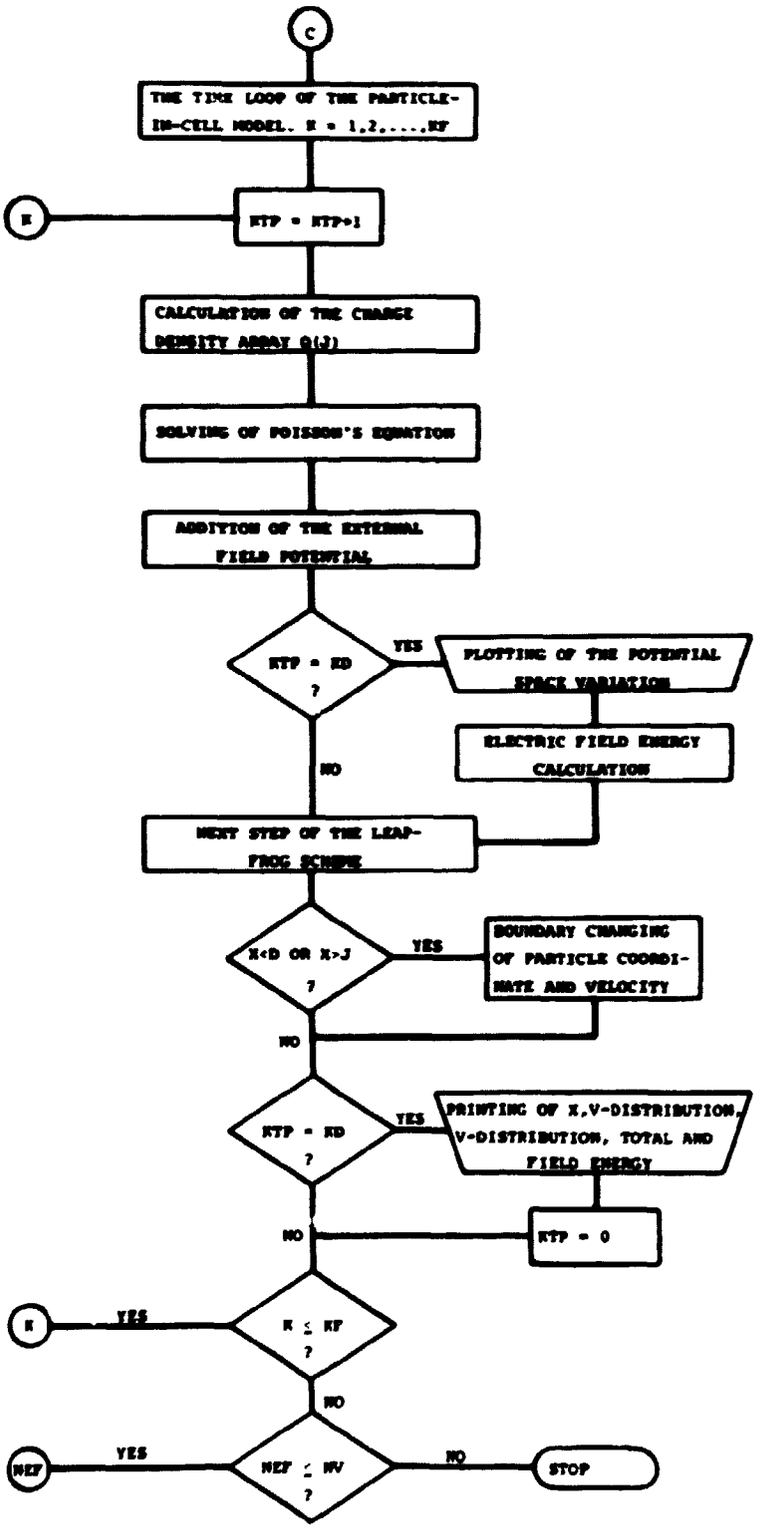


Fig. 1b. General Block Diagram of the Code (cont.)

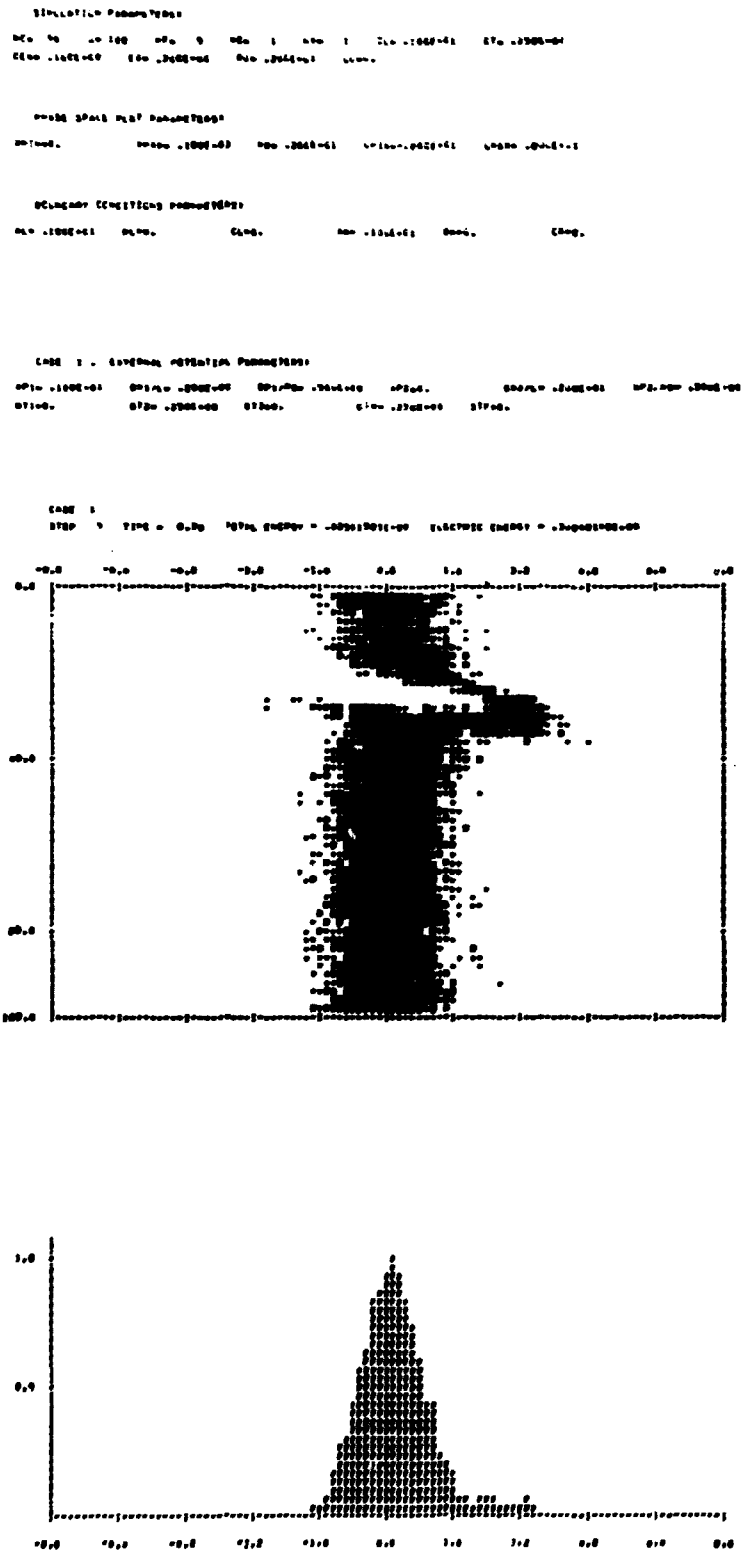

Fig. 2a. Test Run Output by the Line-printer.

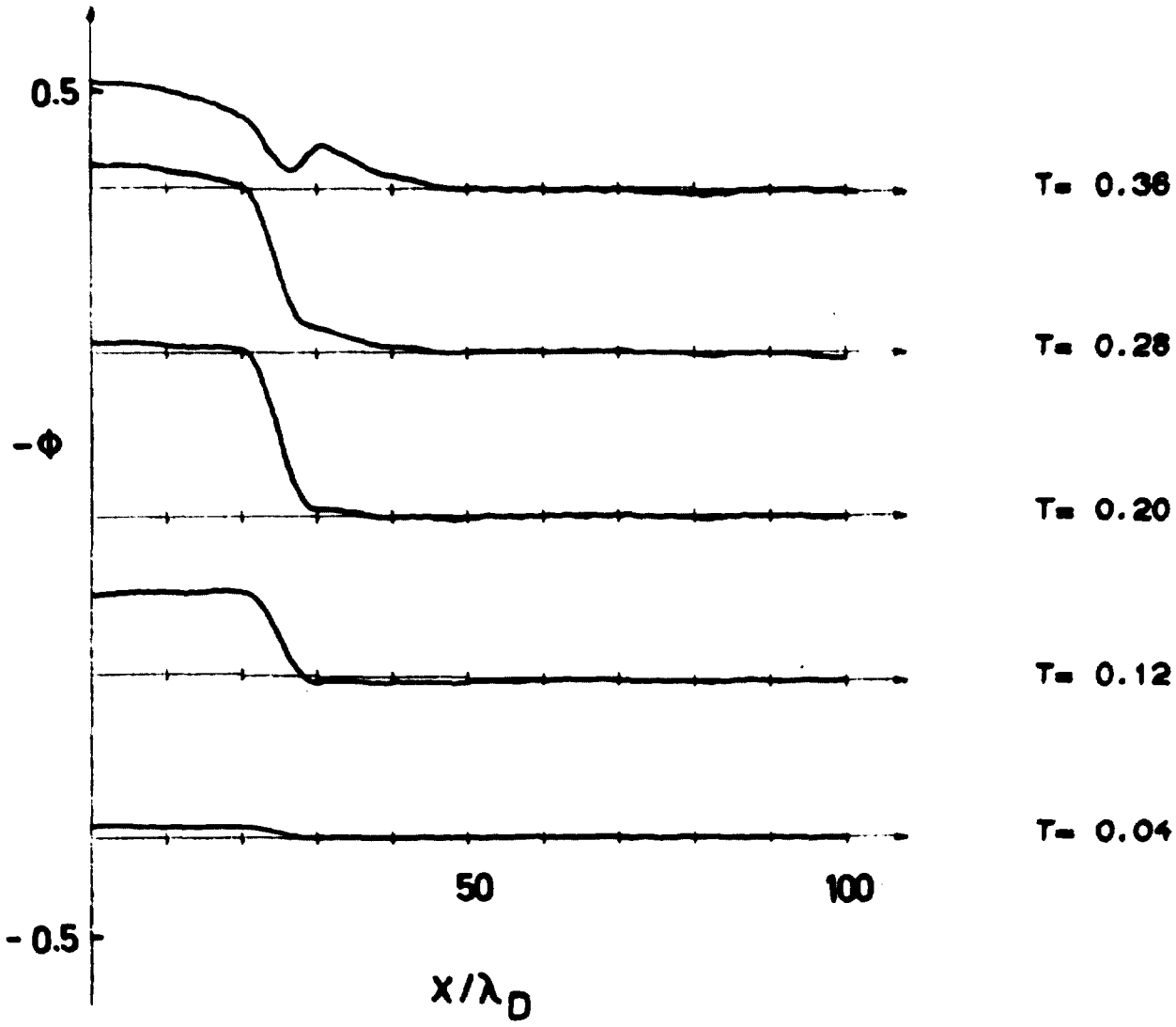


Fig. 2b. Test Run Output by the Plotter. Evolution of the Potential Space Variation.

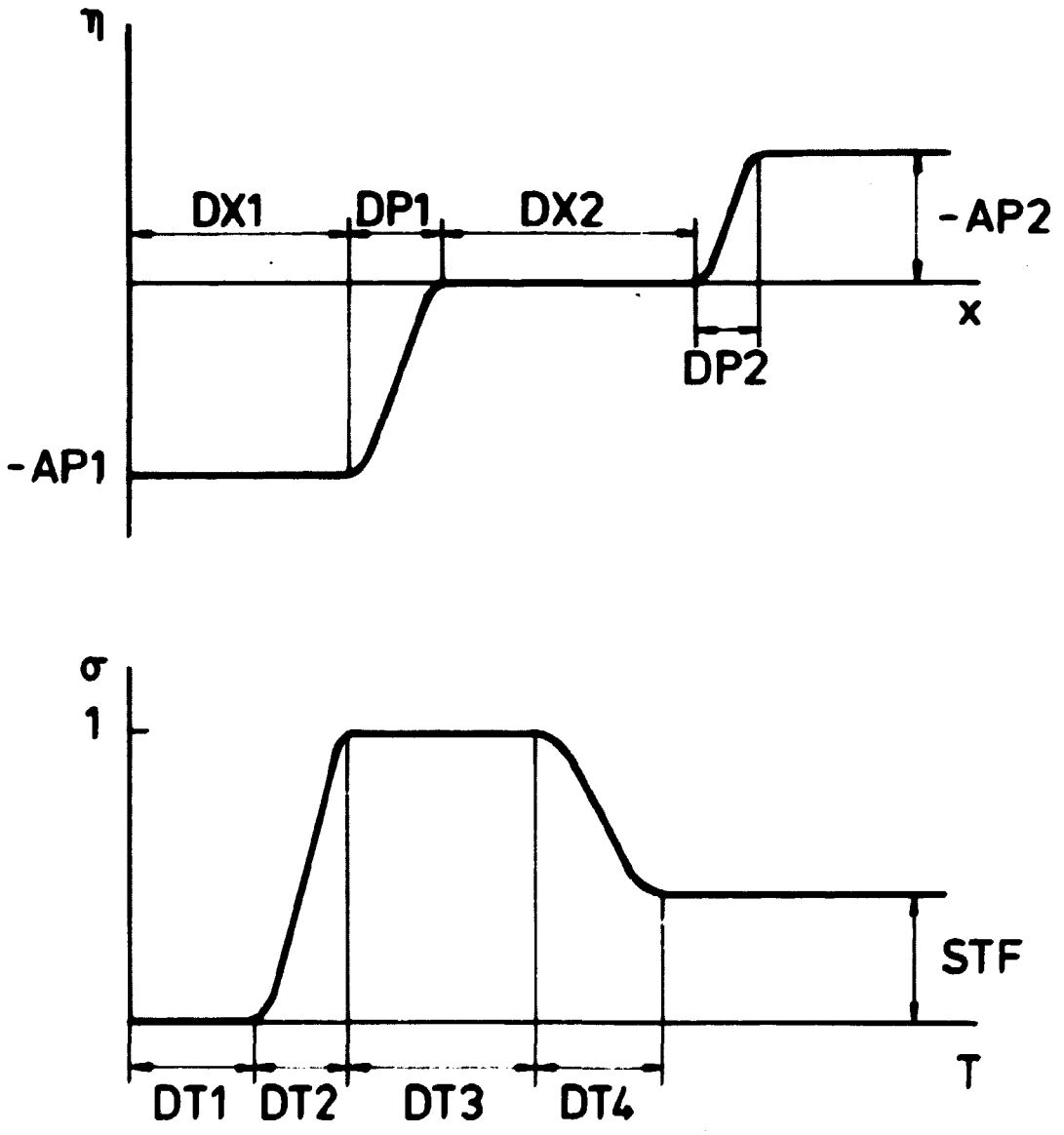


Fig. 3. Space and Time Profiles of the External Potential Function EXTPOT(X,T) = $\eta(X) \cdot \sigma(T)$.

9. Program Listings

# 1. MAIN PROGRAM SIMULA

```
C MAIN PROGRAM SIMULA

      REAL PLTEXT(1),LC
      DATA PLTEXT/'I='/
      DIMENSION X(50000),V(50000)
      COMMON /EXTP/PE(11)/PSA/XPLMAX,EV,AL,BL,CL,AR,BR,CR,J,J2,CAL,
     *FC,F(1000),C(1000),C(1000),S(1000),A(500),B(500)
C
C READING AND PRINTING OF THE INPUT DATA
C
      READ(5,1) NC,J,NF,ND,NV
    1 FORMAT(I5)
      READ(5,2) DL,DT,DEN,EO,RO,LD,XPIN,XPAX,MX,LPIN,LPAX,
     *AL,BL,CL,AR,PR,CM
    2 FORMAT(8(E10.3/),5E10.3/6E10.3)
      WRITE(6,3) NC,J,NF,ND,NV,DL,DT,DEN,EO,RO,UO
    3 FORMAT(////10X,'SIMULATION PARAMETERS:'///
     *11X,'NC=',I4,4X,'J=',I4,4X,'NF=',I4,4X,'ND=',I4,4X,'NV=',I4,4X,
     *'DL=',E9.3,4X,'DT=',E9.3//11X,'DEN=',E9.3,4X,'EO=',E9.3,4X,
     *'RO=',E9.3,4X,'UO=',E9.3///)
      WRITE(6,4) XPIN,XPAX,MX,LPIN,LPAX
    4 FORMAT(///10X,'PHASE SPACE PLOT PARAMETERS:'///
     *11X,'XMIN=',E9.3,4X,'XPAX=',E9.3,4X,'MX=',E9.3,4X,
     *'LPIN=',E9.3,4X,'LPAX=',E9.3///)
      WRITE(6,7) AL,BL,CL,AR,BR,CR
    7 FORMAT(///10X,'BOUNDARY CONDITIONS PARAMETERS:'///
     *11X,'AL=',E9.3,4X,'BL=',E9.3,4X,'CL=',E9.3,4X,
     *'AR=',E9.3,4X,'BR=',E9.3,4X,'CR=',E9.3/////)
C
C CALCULATION OF THE WORK CONSTANTS
C
      N=NC*J
      G=2.*DT*UT
      LD=740.*SQRT(EO/DEN)
      CFO=4.0*(DT*RO/DL/LD)**2/23.116
      VN=2.*SQRT(2.)*DT/DL
      UTH=J*4.*DT*DT
      DKT=1.+(2.400/RO*LD*DL)**2/2.
      DKE=8.*(DKT-1.)
      PI=3.141592654
      PI2=6.283185307
      CALL PSCALE(1200,320)
C
C BOUNDARY CONDITIONS ANALYSIS
C
      IF(AL.EQ.0.AND.BL.EQ.0.AND.AR.EQ.0.AND.BR.EQ.0)KBOUND=1
      IF(KBOUND.EQ.0) GO TO 8
C
C PREPARATION OF THE WORK CONSTANTS AND ARRAYS
C FOR PERIODICAL BOUNDARY CONDITIONS
C
      C(1)=J/2.
      J2=J/2
      IF(C(1)-J2-0.1) 51,51,52
   51 JEV=1
      GO TO 53
   52 JEV=0
      J2=(J-1)/2


   53 C(1)=PI2/J
      S(1)=SIN(C(1))
      C(1)=COS(C(1))
      DO 5 K=2,J
      C(K)=C(K-1)*C(1)-S(K-1)*S(1)
    5 S(K)=S(K-1)*C(1)+C(K-1)*S(1)
```

```
C
C  THE EXTERNAL POTENTIAL LOOP
C
    4 DO 30 NEF=1,NN
      READ(5,P) PL
    P FL=FLOAT(NL)*3/PL)+3)
      WRITE(6,23) NEF,PL
   23 FORMAT(////////////10X,'CASE',I3,'  EXTERNAL POTENTIAL PARAMETERS:'
     *///11X,'AM1=',E9.3,4X,'OM1/L=',E9.3,4X,'PH1/PI=',E9.3,4X,
     *'AM2=',E9.3,4X,'OM2/L=',E9.3,4X,'PH2/PI=',E9.3//11X,'T1=',E9.3,
     *4X,'T2=',E9.3,4X,'T3=',E9.3,4X,'T4=',E9.3,4X,'STP=',E9.3////)
      PE(2)=PE(2)*L
      PE(3)=PE(3)*PI/L2
      PE(5)=PE(5)*L
      PE(6)=PE(6)*PI/L2
      SF=AMAX1(ABS(PE(1)),ABS(PE(6)))*CFC
      IF(SF.LT.1.E-10) SF=1.
C
C GENERATION OF THE INITIAL DISTRIBUTION
C
      CALL INGEN(X,V,AMM,VM,LL)
C
      CALL PSTART
      CALL PWRITG(0.1,3.0)
      CALL PAXES(0.0,0.0,1.0,0.,'V',-1,0.,1.,-1.0)
C
C FIRST TIME STEP BY THE LEAP-FROG METHOD
C
      DO 25 M=1,N
      X(M)=X(M)+V(M)/2.
      IF(X(M).GT.0.0.AND.X(M).LT.J) GO TO 25
      X(M)=X(M)-SIGN(J,X(M))
      IF(NBLOWU) 12,14,12
   14 X(M)=J-X(M)
      V(M)=-V(M)
   12 IF(X(M).EQ.0.0.OR.X(M).EQ.J) X(M)=ABS(V(M))*1.E-10)
   25 CONTINUE
C
C TIME LOOP OF THE PARTICLE-IN-CELL MODEL
C
      DO 30 NT=1,NF
      NTP=NTP+1
C
C CALCULATION OF THE CHARGE DENSITY ANNAT O(J)
C
      DO 26 I=1,J
   26 Q(I)=0.
      DO 27 M=1,N
      I=X(M)+1
      QQ=1-X(M)
      II=I-1
      IF(II.EQ.0) II=J
      Q(II)=Q(II)+QQ
   27 Q(I)=Q(I)+1-QQ



      DO 28 I=1,J
   28 Q(I)=Q*(W(I)/AC-1.)
C
C POISSON EQUATION SOLVING
C
      CALL PASOLV
C
C ADDITION OF THE EXTERNAL FIELD POTENTIAL
C
      T=DT/PI2*(2*K-1)
      DO 32 I=1,J
   32 F(I)=F(I)+CFO*EXTPOT(I,T)
      FO=FO+CFO*EXTPOT(0,T)
      IF(NTP.LT.NC) GO TO 99
C
C PLOT OF THE POTENTIAL SPACE VARIATION
C
      CALL PSTRNG(PLTEXT,2,1.25,-0.02,0.3)
      CALL PREALF(I,6,(1.28-0.02)*0.3)
      CALL PLOT(1.0,-F(J)/SF,1)
      DO 10 M=1,J-1
      XP=1+1.0/J*M
   10 CALL PLOT(XP,-F(J-M)/SF,0)
      IF(N.EQ.NF) GO TO 93
      CALL PWRITG(0,0,0.082)
      CALL PAXES(0.0,0.0,1.0,1.0,-1.0,0.1.0,5)
```

```
C
C  RADIAL ELECTRIC FIELD ENERGY CALCULATION
C
   47 UT=0.
      DO 24 I=1,M
      II=I
      IF(I.EQ.1) II=M-1
      IF(II-1) 46,45,46
   45 F1=F(M)*F(M)
      GO TO 47
   46 F1=F(II-1)*F(II-1)
   47 F2=F(II)*F(II)
      F3=F(II+1)*F(II+1)
      AF=0.5*(F1-2.*F2+F3)
      BF=F3-F2-AF*(2*II+1)
      CF=F2-BF*II-AF*II*II
   24 UT=UT+AF/3.*(I**3-(I-1)**3)+BF/2.*(I**2-(I-1)**2)+CF
      UT=CKE*UT
C
C  CALCULATION OF THE COEFFICIENTS FOR THE ELECTRIC FIELD INTERPOLATION
C
   49 EL=2.*(F0-2.*F(1)+F(2))
      G(1)=2.*(F(2)-F(1))-EL*1.5
      DO 29 I=2,M-1
      EN=2.*(F(I-1)-2.*F(I)+F(I+1))
      F(I-1)=EL
      G(I)=2.*(F(I+1)-F(I))-EN*(2*I+1)/2.
   29 EL=EN
      F(M-1)=EN
      F(M)=F(M-1)
      G(M)=G(M-1)
      IF(NTP-NG) 22,21,22
C
C  TOTAL ELECTRIC FIELD ENERGY CALCULATION


C
   21 DO 18 I=1,M
      EN=1
      EL=I-1
      UT=UT+2(I)*G(I)*(EN-EL)
      EN=EN*EN
      EL=EL*EL
      UT=UT+F(I)*G(I)*(EN-EL)
      EN=EN*I
      EL=EL*(I-1)
   18 UT=UT+F(I)*F(I)/3.*(EN*EL)
      UT=UT/UTN
      ET=0.
C
C  NEXT STEP OF THE LEAP-FROG SCHEME
C
   22 DO 20 N=1,N
      I=X(N)+1
      IF(NTP.EQ.NC) VNC=V(N)
      V(N)=V(N)+F(I)*X(N)+G(I)
      IF(NTP.EQ.NC) ET=ET+0.25*(VNC+V(N))**2
      X(N)=X(N)+V(N)
      IF(X(N).GT.0.0.AND.X(N).LT.J) GO TO 20
      X(N)=X(N)-SIGN(J,X(N))
      IF(NBOUND) 42,40,42
   40 X(N)=J-X(N)
      V(N)=-V(N)
   42 IF(X(N).EQ.0.0.OR.X(N).EQ.J) X(N)=ABS(X(N)-1.E-10)
   20 CONTINUE
C
C  OUTPUT OF INFORMATION ABOUT PLASMA STATE
C
      IF(NTP.LT.NC) GO TO 30
      WRITE(6,17) NEF,NT,ET/N,UT,UT
   17 FORMAT(//////1X,'CASE ',I2//10X,'STEP ',I3,3X,'TIME =',F6.2,3X,
     *'TOTAL ENERGY =',E14.8,3X,'ELECTRIC ENERGY =',E14.8//)
      IF(XMIN.LT.XMAX)
     *CALL PCTRNV(V,V,N,XMIN,XMAX,PX,UMIN,UMAX,X,VA)
      NTP=0
   30 CONTINUE
      CALL PTEMP
      STOP
      END
```

# 2. SUBROUTINE PNSOLV

```
      SUBROUTINE PNSOLV
      COMMON /PSN/ KBOUND,JEV,AL,BL,CL,AR,BR,CR,J,J2,DKT,FO,F(1000),
     *A(1000),C(1000),S(1000),A(500),B(500)
      IF(KBOUND.GT.0) GO TO 3
C
C NONPERIODICAL BOUNDARY CONDITIONS (SCANNING METHOD).
C
C                 DOWNWARDS SCANNING
C
      CM=AR+BR
      C(J)=AR*(2.-DKT)/CM
      S(J)=(CR-0.5*AR*S(J-1))/CM
      DO 1 I=1,J-1
      M=J-I+1
      CM=C(M)-2.+DKT
      C(M-1)=-1./CM
    1 S(M-1)=(S(M-1)-S(M))/CM
C
C                 UPWARDS SCANNING
C
      FO=0.
      IF(DKT.GT.1.) FO=(CL+AL*(0.5*(C(2)*S(1)+S(2))-2.*S(1)))/
     *(AL*(2.*C(1)-0.5*C(1)*C(2)-1.5)+BL)
      F(1)=C(1)*FO+S(1)
      DO 2 I=1,J-1
    2 F(I+1)=F(I)*C(I+1)+S(I+1)
      GO TO 8
C
C PERIODICAL BOUNDARY CONDITIONS (FOURIER METHOD).
C
C                 FOURIER ANALYSIS
C
    3 DO 5 I=1,J2
      A(I)=0.
      B(I)=0.
      C1=1.
      S1=0.
      DO 4 M=1,J
      CM=C1*C(I)-S1*S(I)
      S1=S1*C(I)+C1*S(I)
      C1=CM
      A(I)=A(I)+S(M)*C1
    4 B(I)=B(I)+S(M)*S1
      CM=J*(C(I)-DKT)
      A(I)=A(I)/CM
    5 B(I)=B(I)/CM
      IF(JEV.GT.0) A(J2)=A(J2)/2.
C
C                 FOURIER SYNTHESIS
C
      DO 6 I=1,J
      F(I)=0.
      C1=1.
      S1=0.
      DO 6 M=1,J2
      CM=C1*C(I)-S1*S(I)
      S1=S1*C(I)+C1*S(I)
      C1=CM
    6 F(I)=F(I)+A(M)*C1+B(M)*S1



      DO 7 I=1,J
    7 F(I)=F(I)-F(J)
      FO=0.
    8 RETURN
      END
```

## 3. SUBROUTINE INGEN

```
      SUBROUTINE INGEN(X,V,N,U,VN,UD)
      DIMENSION X(N),V(N)
      DX=(1.*U)/N
      X(1)=DX/2
      DO 1 I=2,N
    1 X(I)=X(I-1)+DX
      Y=174
      IV=8479
      DO 2 I=1,N
      CALL NORMAN(IV,U)
    2 V(I)=VN*(U/1.414214+SIGN(UD,RANDOM(Y)-0.5))
      RETURN
      END
```

## 4. SUBROUTINE PCTRXV

```
      SUBROUTINE PCTRXV(X,V,N,XMIN,XMAX,MX,UMIN,UMAX,VN)
      DIMENSION X(N),V(N),STRING(101)
      DIMENSION MESH(250,100)
      REAL BL/' '/,SN1/'*'/,SN2/'X'/,SN3/'B'/,SN4/'I'/,SN5/'+'/
      REAL SN6/'='/,SN7/'6'/,SN8/'M'/
      N1=1
      N2=3
      IXMAX=(XMAX-XMIN)/MX+0.1
      DO 25 I=1,IXMAX
      DO 25 K=1,100
   25 MESH(I,K)=0
      MV=(UMAX-UMIN)*VN/100.
      VMIN=UMIN*VN
      DO 1 M=1,N
      IX=(X(M)-XMIN)/MX+0.5
      IV=(V(M)-VMIN)/MV+0.5
      IF((IX.LE.0.OR.IX.GE.IXMAX).OR.(IV.LE.0.OR.IV.GE.100))
     *GO TO 1
      MESH(IX,IV)=MESH(IX,IV)+1
    1 CONTINUE
      WRITE(6,2)(UMIN+(I-1)*10*MV/VN,I=1,11)
    2 FORMAT(//14X,F4.1,10(6X,F4.1)/)
      WRITE(6,3)
    3 FORMAT(16X,10('I---------'),'I')
      WRITE(6,14) XMIN
      MARK=20
      DO 4 IX=1,IXMAX-1
      DO 5 K=2,100
      IF(MESH(IX,K-1)) 7,6,7
    6 STRING(K)=BL
      GO TO 5
    7 IF(MESH(IX,K-1)-N1) 8,8,9
    8 STRING(K)=SN1
      GO TO 5
```

```
    9 IF(MESH(IX,K-1)-NN) 10,10,11
   10 STRING(K)=SN2
      GO TO 5
   11 STRING(K)=SN3
    5 CONTINUE
      BS=SN8
      IF(IX.EQ.MARK) BS=SN5
      STRING(1)=BS
      STRING(101)=BS
   12 FORMAT(16X,101A1)
      WRITE(6,12) STRING
      DO 22 K=1,101
      BS=BL
      IF(STRING(K).EQ.SN2) BS=SN4
      IF(STRING(K).EQ.SN3) BS=SN7
   22 STRING(K)=BS
      WRITE(6,23) STRING
   23 FORMAT('+',15X,101A1)
      DO 24 K=1,101
      BS=BL
      IF(STRING(K).EQ.SN7) BS=SN9
   24 STRING(K)=BS
      WRITE(6,23) STRING
      IF(MARK-IX) 13,13,4



   13 WRITE(6,14) XMIN+IX*HX
   14 FORMAT('+',7X,F6.1)
      MARK=MARK+20
    4 CONTINUE
      WRITE(6,3)
      WRITE(6,14) XMAX
      DO 16 IV=1,100
      DO 16 IX=2,IXMAX-1
   16 MESH(1,IV)=MESH(1,IV)+MESH(IX,IV)
      NMAX=0
      DO 17 IV=1,100
      IF(MESH(1,IV).GT.NMAX)NMAX=MESH(1,IV)
   17 CONTINUE
      BS=SN4
      WRITE(6,19) BS,BS
   19 FORMAT('1',7(/),16X,A1/16X,A1)
      DO 21 IX=1,30
      STRING(1)=SN4
      IF(IX.EQ.1.OR.IX.EQ.16) STRING(1)=SN5
      DO 20 K=2,100
      STRING(K)=BL
      IF(MESH(1,K-1).GE.(31-IX)*NMAX*0.03333) STRING(K)=SNJ
   20 CONTINUE
      WRITE(6,12) STRING
      IF(IX.EQ.1.OR.IX.EQ.16) WRITE(6,14)(31-IX)/30.
   21 CONTINUE
      WRITE(6,3)
      WRITE(6,2)(LMIN+(I-1)*10+MV/VN,I=1,11)
      RETURN
      END
```

# 5. FUNCTION EXTPOT

```
      FUNCTION EXTPOT(X,T)
      COMMON /EXTP/ PE(11)
      PI=3.141592654
      T1=T-PE(7)
      IF(T1.LE.0.0) GO TO 18
      IF(T1-PE(8)) 3,3,4
    3 AT=0.5*(1-COS(PI*(T1/PE(9))))
      GO TO 9
    4 T1=T1-PE(8)
      IF(T1-PE(9)) 5,5,6
    5 AT=1.0
      GO TO 9
    6 T1=T1-PE(9)
      IF(T1-PE(10)) 7,7,8
    7 AT=0.5*((1-PE(11))*COS(PI*T1/PE(10))+1.0+PE(11))
      GO TO 9
    8 AT=PE(11)
    9 IF(ABS(AT).LT.1.E-7) GO TO 18
      X1=X-PE(2)
      IF(X1) 10,10,11
   10 EXTPUT=-PE(1)*AT
      GO TO 14
   11 IF(X1-PE(3)) 12,12,13
   12 IF(PE(1).EQ.0) GO TO 18
      EXTPUT=-PE(1)*0.5*AT*(COS(PI*X1/PE(3))+1.0)
      GO TO 14
   13 X1=X1-PE(3)
      IF(X1-PE(5)) 18,18,15
   18 EXTPUT=0.
      GO TO 14
   15 X1=X1-PE(5)
      IF(X1-PE(6)) 16,16,17
   16 IF(PE(4).EQ.0) GO TO 18
      EXTPUT=PE(4)*0.5*AT*(COS(PI*X1/PE(6))-1.0)
      GO TO 14
   17 EXTPUT=-PE(4)*AT
   14 RETURN
      END
```