平成17年度
卒業研究報告書

粒子コードによる荷電粒子と電磁場の相互作用の基礎的研究

秋田工業高等専門学校　電気工学科

研究者名　　相場 亮人

指導教員名　　山本 昌志

目 次

第 1 章　緒言 **1**

第 2 章　PIC 法の理論 **2**
2.1 基礎方程式 2
2.1.1 マクスウェルの方程式 2
2.1.2 運動方程式 2
2.1.3 電磁場と荷電粒子の相互作用 3
2.2 離散化 3
2.2.1 空間の離散化 3
2.2.2 時間の離散化 4
2.2.3 マクスウェルの方程式の離散化 5
2.2.4 運動方程式の離散化 8
2.3 必要となる計算方法 9
2.3.1 粒子に作用する電磁場の計算 9
2.3.2 電流密度の計算 11

第 3 章　数値計算 **13**
3.1 プログラムの概要 13

第 4 章　計算結果 **14**

第 5 章　考察と今後の課題 **16**

第 6 章　結言 **17**

付 録 A 計算プログラム **20**

要旨

本研究では，粒子コード（Particle-in-Cell Code，PIC 法）を用いて 2 次元軸対称モデルにおける荷電粒子と電磁場の相互作用を解析するための基礎的研究とプログラムの作成を行った．粒子コードとは，主にマクスウェルの方程式と運動方程式を連立させて解いていくことで相互作用の計算を行う方法である．

粒子コードによって相互作用の計算を行うプログラムを作成したところ，おおよそ正しい解析結果を得ることが出来た．

第1章　緒言

加速器を設計する際に，荷電粒子の振る舞いとそれによって作られる電磁場を解析することは重要である．様々な制約条件のもとで要求される性能を引き出すために，通常は計算機シミュレーションをして最適化を行う．

本研究では，粒子コード（Particle-in-Cell Code，PIC 法）を用いて 2 次元軸対称モデルにおける荷電粒子と電磁場の相互作用を解析するための基礎的研究とプログラムの作成を行った．粒子コードとは，電磁場の作用による粒子の運動の変化と，粒子の運動による電磁場の変化とを交互に繰り返し計算していくことで相互作用を計算する方法である．

第2章　PIC法の理論

2.1　基礎方程式

2.1.1　マクスウェルの方程式

電磁場の関係は，以下のマクスウェルの方程式によって述べられる．

$$\nabla \cdot \boldsymbol{E} = \frac{\rho}{\varepsilon} \tag{2.1}$$

$$\nabla \cdot \boldsymbol{H} = 0 \tag{2.2}$$

$$\nabla \times \boldsymbol{E} = -\mu \frac{\partial \boldsymbol{H}}{\partial t} \tag{2.3}$$

$$\nabla \times \boldsymbol{H} = \boldsymbol{j} + \varepsilon \frac{\partial \boldsymbol{E}}{\partial t} \tag{2.4}$$

ここで，$\boldsymbol{E}$ は電場の強さ，$\boldsymbol{H}$ は磁場の強さ，$\boldsymbol{j}$ は電流密度を表す．ρ は電荷密度，ε は誘電率，μ は透磁率である．

これらのうち PIC 法では (2.3) 式と (2.4) 式を用いて電磁場の変化を求める．ただし，この微分方程式を直接計算するのではなく，ストークスの定理を用いて変形した以下の積分形の方程式を計算することになる．

$$\oint \boldsymbol{E} \cdot dl = -\mu \frac{d}{dt} \int_S \boldsymbol{H} \cdot \boldsymbol{n} dS \tag{2.5}$$

$$\oint \boldsymbol{H} \cdot dl = \int_S \boldsymbol{j} \cdot \boldsymbol{n} dS + \varepsilon \frac{d}{dt} \int_S \boldsymbol{E} \cdot \boldsymbol{n} dS \tag{2.6}$$

また，他の (2.1)，(2.2) 式については初期状態において満たされている必要がある．そうすれば，その後は自動的にこれらの式は満たされることになる．

2.1.2　運動方程式

PIC 法では粒子の運動を解析するために運動方程式を用いるが，高エネルギーの粒子の運動を扱うので相対論的力学を用いる必要がある．運動方程式は

$$\frac{d\boldsymbol{p}}{dt} = \boldsymbol{F} \tag{2.7}$$

であり，ここで $\boldsymbol{F}$ は力，$\boldsymbol{p}$ は運動量であるが，運動量 $\boldsymbol{p}$ は

$$\boldsymbol{p} = m\boldsymbol{v} = \gamma m_0 \boldsymbol{\beta} c \tag{2.8}$$

となる．m は粒子の質量，m_0 は粒子の静止質量，$\boldsymbol{v}$ は粒子の速度，c は光速度である．$\boldsymbol{\beta}$ は光速度と粒子の質量の比，γ は相対論的因子で，それぞれ

$$\boldsymbol{\beta} = \frac{\boldsymbol{v}}{c} \tag{2.9}$$

$$\gamma = \frac{1}{\sqrt{1 - |\boldsymbol{\beta}|^2}} \tag{2.10}$$

となる．

一方，力 $\boldsymbol{F}$ はローレンツ力のみが解析対象となる．電磁場中で電荷量 q をもつ質点が受ける力は

$$\boldsymbol{F} = q(\boldsymbol{E} + \boldsymbol{v} \times \boldsymbol{B}) \tag{2.11}$$

である．ここで $\boldsymbol{B}$ は磁束密度を表す.

以上のことから，解くべき運動方程式は

$$\frac{d\boldsymbol{p}}{dt} = q(\boldsymbol{E} + \boldsymbol{v} \times \boldsymbol{B}) \tag{2.12}$$

となる．ここで速度 $\boldsymbol{v}$ は運動量より，

$$\boldsymbol{v} = c\boldsymbol{\beta} = \frac{c\boldsymbol{p}}{\sqrt{m_0^2 c^2 + \boldsymbol{p} \cdot \boldsymbol{p}}} \tag{2.13}$$

と求められる．

2.1.3 電磁場と荷電粒子の相互作用

(2.12) 式は，電磁場が荷電粒子の運動に与える作用を表していると言える．一方，粒子が電磁場に与える作用は電流密度の計算式

$$\boldsymbol{j} = nq\boldsymbol{v} \tag{2.14}$$

を用いて求めた電流を (2.6) 式に適用することで考慮される．ここで n は粒子の密度を表す.

以上より電磁場と粒子の運動を連立させることができ，これらの式が PIC 法において相互作用を計算するための基礎方程式となる．

2.2 離散化

2.2.1 空間の離散化

複雑な境界条件のもとで，その対象全体に対するマクスウェルの方程式を計算することは不可能に近い．そこで，PIC 法では図 2.1 のように解析対象を細かく領域（メッシュ）に区切って空間の離散化を行い，各領域に電磁場を配置する．そして，各電磁場に対して (2.5) 式や (2.6) 式を適応させ，それらを連立させて解くことで電磁場を計算していく．

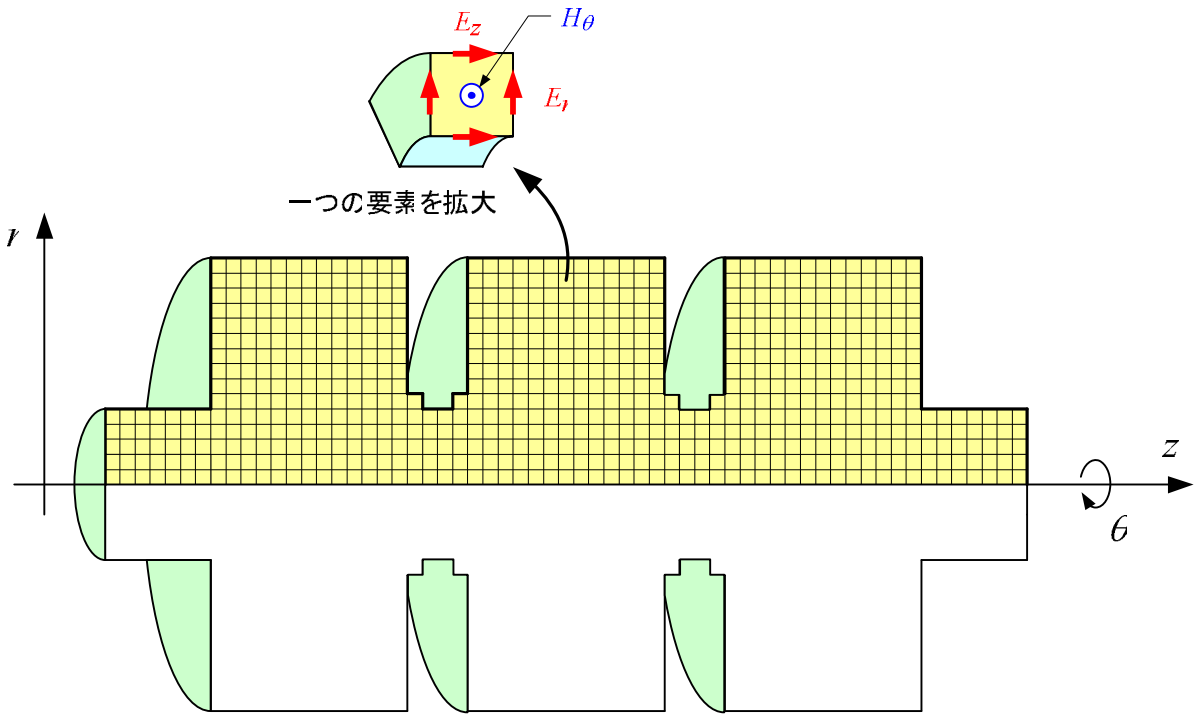

図 2.1: 空間の離散化

図 2.1 では rz 面において領域は一様な正方形で区切られているが，必ずしもこのように区切らなければならないということはない．ここでは計算を容易にするためにこのようにしている．

2.2.2 時間の離散化

PIC 法では空間だけでなく，時間も離散化する．離散化された時間軸上では，図 2.2 のように電場および粒子の位置と，磁場，電流および粒子の運動量は交互にずらして定義する．こうすることで，電磁場や粒子の運動の時間的変化が求められる．

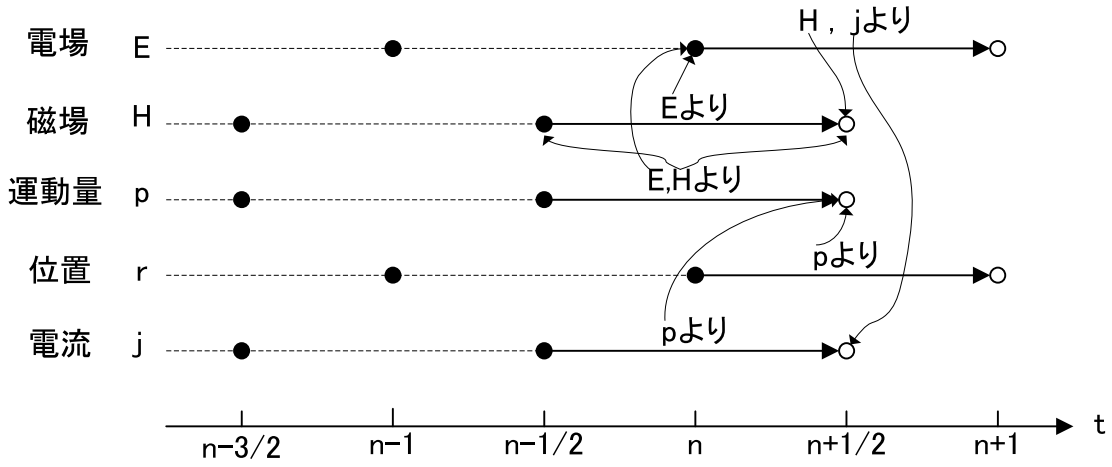

図 2.2: 時間の離散化

2.2.3 マクスウェルの方程式の離散化

(2.5) 式と (2.6) 式を離散化された空間，時間に適応させる．
本研究において解析対象の電磁場は TM_0 モードとするので，

$$E_\theta = 0 \qquad H_r = 0 \qquad H_z = 0 \tag{2.15}$$

となる．つまり，磁場は θ 方向，電場は z 方向と r 方向のみについて考えればよい．また，これらの電磁場は全て位置 (r, z) の関数となり，θ には依存しない．

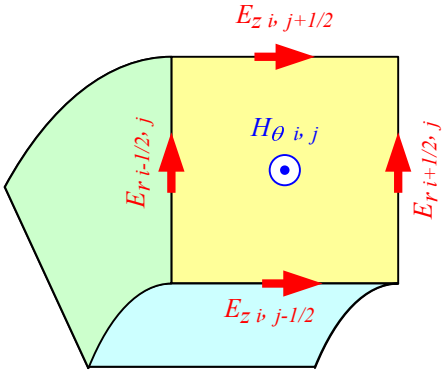

図 2.3: H_θ の計算

まず，θ 方向の磁場について考える．これは，(2.5) 式を図 2.3 の領域の積分路に対して適応することで求められる．(2.5) の左辺は，

$$\oint \boldsymbol{E} \cdot dl \simeq \left[E_r{}^{n}_{i+1/2,j} - E_r{}^{n}_{i-1/2,j}\right](r_{j+1/2} - r_{j-1/2}) + \left[E_z{}^{n}_{i,j-1/2} - E_z{}^{n}_{i,j+1/2}\right](z_{i+1/2} - z_{i-1/2}) \tag{2.16}$$

と近似できる．右辺の近似は，

$$-\mu_0 \frac{d}{dt}\int_S \boldsymbol{H} \cdot \boldsymbol{n} dS \simeq -\frac{\mu_0}{t_{n+1/2} - t_{n-1/2}}\left[H_\theta{}^{n+1/2}_{i,j} - H_\theta{}^{n-1/2}_{i,j}\right](z_{i+1/2} - z_{i-1/2})(r_{j+1/2} - r_{j-1/2}) \tag{2.17}$$

となり，この両辺は等しいので，

$$\left[E_r{}^{n}_{i+1/2,j} - E_r{}^{n}_{i-1/2,j}\right](r_{j+1/2} - r_{j-1/2}) + \left[E_z{}^{n}_{i,j-1/2} - E_z{}^{n}_{i,j+1/2}\right](z_{i+1/2} - z_{i-1/2}) \simeq -\frac{\mu_0}{t_{n+1/2} - t_{n-1/2}}\left[H_\theta{}^{n+1/2}_{i,j} - H_\theta{}^{n-1/2}_{i,j}\right](z_{i+1/2} - z_{i-1/2})(r_{j+1/2} - r_{j-1/2}) \tag{2.18}$$

となる．これを整理すると，

$$H_\theta{}^{n+1/2}_{i,j} \simeq H_\theta{}^{n-1/2}_{i,j} + \frac{t_{n+1/2} - t_{n-1/2}}{\mu_0}\left[\frac{E_r{}^{n}_{i-1/2,j} - E_r{}^{n}_{i+1/2,j}}{z_{i+1/2} - z_{i-1/2}} + \frac{E_z{}^{n}_{i,j+1/2} - E_z{}^{n}_{i,j-1/2}}{r_{j+1/2} - r_{j-1/2}}\right] \tag{2.19}$$

が得られる．この式は，$t_{n+1/2}$ での磁場はそれ以前の情報があれば計算できることを示す．

ここで，領域の分割間隔を Δl，時間間隔を Δt で一定とすると (2.19) 式は

$$H_{\theta\ i,j}^{\ n+1/2} \simeq H_{\theta\ i,j}^{\ n-1/2} + \frac{1}{\mu_0}\frac{\Delta t}{\Delta l}\left[E_{r\ i-1/2,j}^{\ n} - E_{r\ i+1/2,j}^{\ n} + E_{z\ i,j+1/2}^{\ n} - E_{z\ i,j-1/2}^{\ n}\right] \tag{2.20}$$

となる．

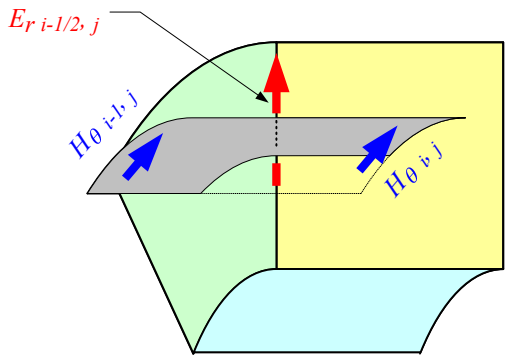

図 2.4: E_r の計算

次に，r 方向の電場について考える．これは，(2.6) 式を図 2.4 の積分路に対して適応することで求められる．(2.6) 式の左辺は，

$$\oint \boldsymbol{H}\cdot dl \simeq \left[H_{\theta\ i-1,j}^{\ n+1/2} - H_{\theta\ i,j}^{\ n+1/2}\right] r_j \Delta\theta \tag{2.21}$$

と近似できる．右辺の近似は，

$$-\varepsilon_0 \frac{d}{dt}\int_S \boldsymbol{E}\cdot\boldsymbol{n}dS \simeq \frac{\varepsilon_0}{t_{n+1}-t_n}\left[E_{r\ i-1/2,j}^{\ n+1} - E_{r\ i-1/2,j}^{\ n}\right](z_i - z_{i-1})\, r_j \Delta\theta \tag{2.22}$$

となり，両辺は等しいので，

$$H_{\theta\ i-1,j}^{\ n+1/2} - H_{\theta\ i,j}^{\ n+1/2} \simeq \frac{\varepsilon_0}{t_{n+1}-t_n}\left[E_{r\ i-1/2,j}^{\ n+1} - E_{r\ i-1/2,j}^{\ n}\right](z_i - z_{i-1}) \tag{2.23}$$

となる．これを整理すると，

$$E_{r\ i-1/2,j}^{\ n+1} \simeq E_{r\ i-1/2,j}^{\ n} + \frac{t_{n+1}-t_n}{\varepsilon_0\,(z_i - z_{i-1})}\left[H_{\theta\ i-1,j}^{\ n+1/2} - H_{\theta\ i,j}^{\ n+1/2}\right] \tag{2.24}$$

となる．この式も先程と同様，t_{n+1} での電場はそれ以前の情報があれば計算できることを示している．

ここで，領域の分割間隔を Δl，時間間隔を Δt で一定とすると (2.24) 式は

$$E_{r\ i-1/2,j}^{\ n+1} \simeq E_{r\ i-1/2,j}^{\ n} + \frac{1}{\varepsilon_0}\frac{\Delta t}{\Delta l}\left[H_{\theta\ i-1,j}^{\ n+1/2} - H_{\theta\ i,j}^{\ n+1/2}\right] \tag{2.25}$$

となる．

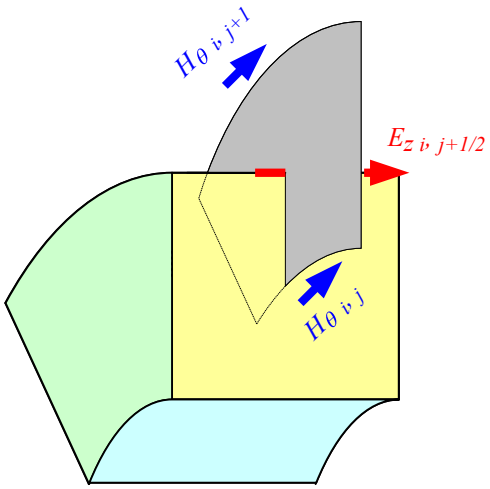

図 2.5: E_z の計算

最後に，z 方向の電場について考える．これは，r 方向と同様に (2.6) 式を図 2.5 の積分路に対して適応することで求められる．(2.6) 式の左辺は，

$$\oint \boldsymbol{H} \cdot dl \simeq \left[H_{\theta\ i,j+1}^{\ n+1/2} r_{j+1} - H_{\theta\ i,j}^{\ n+1/2} r_j \right] \Delta\theta \tag{2.26}$$

と近似できる．右辺の近似は，

$$-\varepsilon_0 \frac{d}{dt} \int_S \boldsymbol{E} \cdot \boldsymbol{n} dS \simeq \frac{\varepsilon_0}{t_{n+1} - t_n} \left[E_{z\ i,j+1/2}^{\ n+1} - E_{z\ i,j+1/2}^{\ n} \right] \frac{1}{2} \left(r_{j+1}^2 - r_j^2 \right) \Delta\theta \tag{2.27}$$

となり，両辺は等しいので，

$$H_{\theta\ i,j+1}^{\ n+1/2} r_{j+1} - H_{\theta\ i,j}^{\ n+1/2} r_j \simeq \frac{\varepsilon_0}{2\left(t_{n+1} - t_n\right)} \left[E_{z\ i,j+1/2}^{\ n+1} - E_{z\ i,j+1/2}^{\ n} \right] \left(r_{j+1}^2 - r_j^2 \right) \tag{2.28}$$

となる．これを整理すると，

$$\begin{aligned} E_{z\ i,j+1/2}^{\ n+1} &\simeq E_{z\ i,j+1/2}^{\ n} + \frac{2(t_{n+1} - t_n)}{\varepsilon_0 \left(r_{j+1}^2 - r_j^2 \right)} \left[H_{\theta\ i,j+1}^{\ n+1/2} r_{j+1} - H_{\theta\ i,j}^{\ n+1/2} r_j \right] \\ &\simeq E_{z\ i,j+1/2}^{\ n} + \frac{2(t_{n+1} - t_n)}{\varepsilon_0 \left(r_{j+1} + r_j \right) \left(r_{j+1} - r_j \right)} \left[H_{\theta\ i,j+1}^{\ n+1/2} r_{j+1} - H_{\theta\ i,j}^{\ n+1/2} r_j \right] \end{aligned} \tag{2.29}$$

となる．この式も先程と同様，t_{n+1} での電場はそれ以前の情報があれば計算できることを示している．

ここで，領域の分割間隔を Δl，時間間隔を Δt で一定とすると (2.29) 式は

$$E_{z\ i,j+1/2}^{\ n+1} \simeq E_{z\ i,j+1/2}^{\ n} + \frac{2\Delta t}{\varepsilon_0 \left(2r_j + \Delta l \right) \Delta l} \left[H_{\theta\ i,j+1}^{\ n+1/2} r_{j+1} - H_{\theta\ i,j}^{\ n+1/2} r_j \right] \tag{2.30}$$

となる．

2.2.4 運動方程式の離散化

運動量方程式 (2.12) の離散化を行う．この式に (2.13) 式を代入し，$\boldsymbol{B} = \mu_0 \boldsymbol{H}$ と変換すると

$$\frac{d\boldsymbol{p}}{dt} = q\left(\boldsymbol{E} + \frac{c\boldsymbol{p}}{\sqrt{m_0^2 c^2 + \boldsymbol{p}\cdot\boldsymbol{p}}} \times \mu_0 \boldsymbol{H}\right) \tag{2.31}$$

となるが，この式には右辺にも運動量 $\boldsymbol{p}$ が存在しているため，運動量の変化に現在の運動量が関わってくることになる．そこでこれを一度に計算することはせず，以下のような順序で計算する．

1. $\Delta t/2$ の間，時刻 n の電場で粒子は加速

$$\boldsymbol{p}_1 = \boldsymbol{p}^{n-1/2} + q\boldsymbol{E}^n \frac{\Delta t}{2} \tag{2.32}$$

2. Δt の間，時刻 $n-1/2$ と $n+1/2$ の平均磁場で粒子は加速

$$\boldsymbol{p}_2 = \boldsymbol{p}_1 + \mu_0 q \frac{c\boldsymbol{p}_1}{\sqrt{m_0^2 c^2 + \boldsymbol{p}_1\cdot\boldsymbol{p}_1}} \times \left[\frac{\boldsymbol{H}^{n-1/2} + \boldsymbol{H}^{n+1/2}}{2}\right] \Delta t \tag{2.33}$$

3. $\Delta t/2$ の間，時刻 n の電場で粒子は加速

$$\boldsymbol{p}^{n+1/2} = \boldsymbol{p}_2 + q\boldsymbol{E}^n \frac{\Delta t}{2} \tag{2.34}$$

こうすることで，運動量の変化を求められる．

求められた運動量から粒子の位置の変化を計算するわけであるが，これは

$$\frac{d\boldsymbol{r}}{dt} = \boldsymbol{v} \tag{2.35}$$

を用いればよい．(2.13) 式を用いて，

$$\boldsymbol{r}^{n+1} = \boldsymbol{r}^n + \frac{c\boldsymbol{p}^{n+1/2}}{\sqrt{m_0^2 c^2 + \boldsymbol{p}^{n+1/2}\cdot\boldsymbol{p}^{n+1/2}}} \Delta t \tag{2.36}$$

が得られる．

2.3 必要となる計算方法

2.3.1 粒子に作用する電磁場の計算

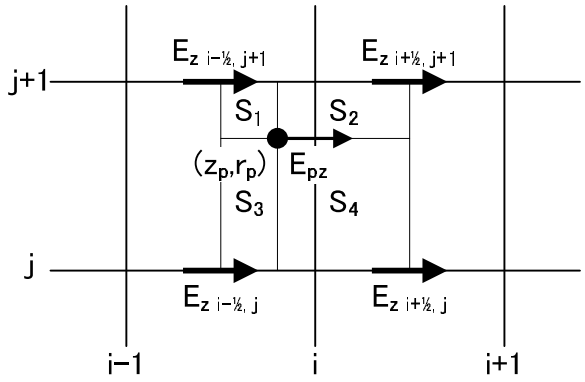

図 2.6: 粒子に作用する電場 E_{pz} の計算

図 2.6 のような位置に粒子が存在する場合の，粒子に作用する z 方向の電場 E_{pz} について考える．これを求めるには，周辺に配置された電場 $E_{z\ i-1/2,j}$，$E_{z\ i+1/2,j}$，$E_{z\ i-1/2,j+1}$，$E_{z\ i+1/2,j+1}$ の影響を粒子からの距離に応じた重み関数を用いて考える必要がある [1]．重み関数には，図 2.6 に示す長方形の面積 S_1〜S_4 を用いる．つまり，

$$E_{pz} = \frac{S_1}{\Delta l^2} E_{z\ i+1/2,j} + \frac{S_2}{\Delta l^2} E_{z\ i-1/2,j} + \frac{S_3}{\Delta l^2} E_{z\ i+1/2,j+1} + \frac{S_4}{\Delta l^2} E_{z\ i-1/2,j+1} \tag{2.37}$$

とする．ここで S_1〜S_4 は

$$\begin{aligned} S_1 &= \left(z_p - z_{i-1/2}\right)\left(r_{j+1} - r_p\right) \\ S_2 &= \left(z_{i+1/2} - z_p\right)\left(r_{j+1} - r_p\right) \\ S_3 &= \left(z_p - z_{i-1/2}\right)\left(r_p - r_j\right) \\ S_4 &= \left(z_{i+1/2} - z_p\right)\left(r_p - r_j\right) \end{aligned}$$

である．

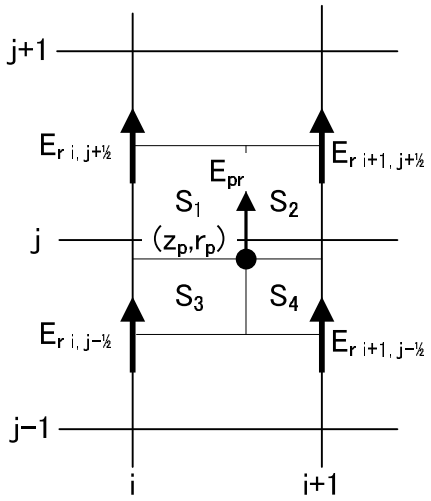

図 2.7: 粒子に作用する電場 E_{pr} の計算

r 方向の電界 E_{pr} を求める場合も同様である．図 2.7 において，

$$E_{pr} = \frac{S_1}{\Delta l^2} E_{r\ i+1,j-1/2} + \frac{S_2}{\Delta l^2} E_{r\ i,j-1/2} + \frac{S_3}{\Delta l^2} E_{r\ i+1,j+1/2} + \frac{S_4}{\Delta l^2} E_{r\ i,j+1/2} \tag{2.38}$$

と求められる．

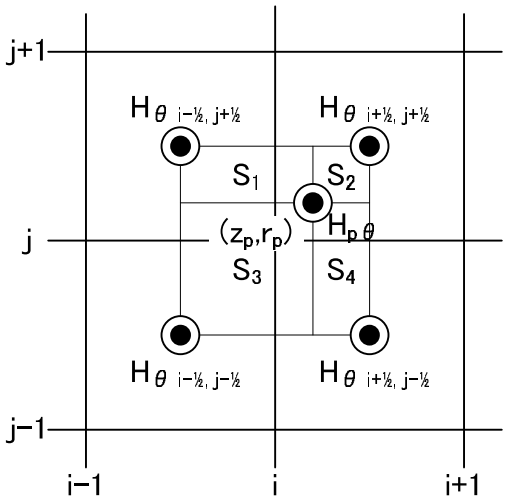

図 2.8: 粒子に作用する磁場 $H_{p\theta}$ の計算

磁場についてもやはり同様で，図 2.8 において粒子に作用する磁場 $H_{p\theta}$ は

$$H_{p\theta} = \frac{S_1}{\Delta l^2} H_{\theta\ i+1/2,j-1/2} + \frac{S_2}{\Delta l^2} H_{\theta\ i-1/2,j-1/2} + \frac{S_3}{\Delta l^2} H_{\theta\ i+1/2,j+1/2} + \frac{S_4}{\Delta l^2} H_{\theta\ i-1/2,j+1/2} \tag{2.39}$$

と求められる．

2.3.2 電流密度の計算

電流密度の計算には (2.14) 式が用いられるが，このとき電荷保存則

$$\nabla \cdot \boldsymbol{j} + \frac{d\rho}{dt} = 0 \tag{2.40}$$

が満たされている必要がある．この式は，電荷が突如発生，あるいは消滅することは無いということを示し，(2.1) 式と (2.4) 式から得られる．つまりこの式が満たされていれば，(2.1) 式が自動的に満たされることになる．

ここでは，(2.14) 式で求めた電流を重み関数を用いて周辺の各点に配分するという方法をとる．粒子が Δt の間に (z_1, r_1) から (z_2, r_2) へと移動したとして，計算手順を以下に示す [2]．ここで $|z_2 - z_1| < \Delta l$ かつ $|r_2 - r_1| < \Delta l$ とする．

1. $\frac{z_1}{\Delta l}$，$\frac{z_2}{\Delta l}$，$\frac{r_1}{\Delta l}$，$\frac{r_2}{\Delta l}$ の各値の小数点以下を切り捨てた整数値をそれぞれ i_1，i_2，j_1，j_2 とする．
2. relay point (z_r, r_r) を設定する．これは異なる領域間を粒子が移動する場合について考える際に役立つ．

$$z_r = \begin{cases} \frac{z_1+z_2}{2} & (i_1 = i_2) \\ \max(i_1\Delta l, i_2\Delta l) & (i_1 \neq i_2) \end{cases} \tag{2.41}$$

$$r_r = \begin{cases} \frac{r_1+r_2}{2} & (j_1 = j_2) \\ \max(j_1\Delta l, j_2\Delta l) & (j_1 \neq j_2) \end{cases} \tag{2.42}$$

とする．

3. $\frac{z_1}{\Delta l}$，$\frac{r_1}{\Delta l}$ を四捨五入した整数値をそれぞれ i，j とする．
4. 粒子の速度として，

$$v_z = \frac{z_r - z_1}{\Delta t} \qquad v_r = \frac{r_r - r_1}{\Delta t} \tag{2.43}$$

を求める．

5. 一次形状関数を計算する．

$$W_i^{(1)} = \frac{z_1 + z_r}{2\Delta l} - i \qquad W_j^{(1)} = \frac{r_1 + r_r}{2\Delta l} - j \tag{2.44}$$

6. モーメントを計算する．ここで，q は θ 方向一周分の電荷量である．

$$\begin{aligned} F_{z1} &= qv_z\left(\frac{1}{2} - W_i^{(1)}\right),\ F_{z2} = qv_z\left(\frac{1}{2} + W_i^{(1)}\right) \\ F_{r1} &= qv_r\left(\frac{1}{2} - W_j^{(1)}\right),\ F_{r2} = qv_r\left(\frac{1}{2} + W_j^{(1)}\right) \end{aligned} \tag{2.45}$$

7. 二次形状関数を計算する．

$$\begin{aligned} W_{i-1}^{(2)} &= \frac{1}{2}\left(\frac{1}{2} - W_i^{(1)}\right)^2,\ W_i^{(2)} = \frac{3}{4} - \left(W_i^{(1)}\right)^2,\ W_{i+1}^{(2)} = \frac{1}{2}\left(\frac{1}{2} + W_i^{(1)}\right)^2 \\ W_{j-1}^{(2)} &= \frac{1}{2}\left(\frac{1}{2} - W_j^{(1)}\right)^2,\ W_j^{(2)} = \frac{3}{4} - \left(W_j^{(1)}\right)^2,\ W_{j+1}^{(2)} = \frac{1}{2}\left(\frac{1}{2} + W_j^{(1)}\right)^2 \end{aligned} \tag{2.46}$$

8. 各電流値を計算する．

$$
\begin{aligned}
j_{z\ i-1/2,j-1} &= \frac{1}{2(j-1)\pi\Delta l^3} F_{z1} W_{j-1}^{(2)} \\
j_{z\ i+1/2,j-1} &= \frac{1}{2(j-1)\pi\Delta l^3} F_{z2} W_{j-1}^{(2)} \\
j_{z\ i-1/2,j} &= \frac{1}{2j\pi\Delta l^3} F_{z1} W_{j}^{(2)} \\
j_{z\ i+1/2,j} &= \frac{1}{2j\pi\Delta l^3} F_{z2} W_{j}^{(2)} \\
j_{z\ i-1/2,j+1} &= \frac{1}{2(j+1)\pi\Delta l^3} F_{z1} W_{j+1}^{(2)} \\
j_{z\ i+1/2,j+1} &= \frac{1}{2(j+1)\pi\Delta l^3} F_{z2} W_{j+1}^{(2)} \\
j_{r\ i-1,j-1/2} &= \frac{1}{2(j-1)\pi\Delta l^3} F_{r1} W_{i-1}^{(2)} \\
j_{r\ i-1,j+1/2} &= \frac{1}{2(j-1)\pi\Delta l^3} F_{r2} W_{i-1}^{(2)} \\
j_{r\ i,j-1/2} &= \frac{1}{2j\pi\Delta l^3} F_{r1} W_{i}^{(2)} \\
j_{r\ i,j+1/2} &= \frac{1}{2j\pi\Delta l^3} F_{r2} W_{i}^{(2)} \\
j_{r\ i+1,j-1/2} &= \frac{1}{2(j+1)\pi\Delta l^3} F_{r1} W_{i+1}^{(2)} \\
j_{r\ i+1,j+1/2} &= \frac{1}{2(j+1)\pi\Delta l^3} F_{r2} W_{i+1}^{(2)}
\end{aligned}
\tag{2.47}
$$

9. z_1，z_r，r_1，r_r をそれぞれ z_r，z_2，r_r，r_2 に変えて 3.〜8. を同様に計算する．

この電流密度の計算方法は基本的に電荷保存則を満たすが，導体境界周辺の電流密度を計算する場合には影像電荷の考え方が必要となる．つまり，図 2.9 のように大きさが同じで符号が逆の荷電粒子が境界の向こう側で境界線に対して線対称の運動をしていると考え，その影響を足し合わせるのである．

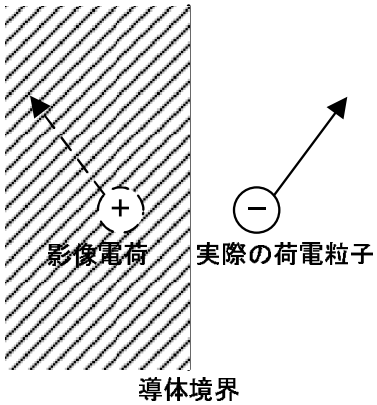

図 2.9: 影像電荷

第3章　数値計算

3.1　プログラムの概要

粒子コードを用いて 2 次元軸対称モデルの加速管内部の電磁場および粒子の運動を計算するプログラムを作成した．使用した言語は C++である．

プログラムは，以下のような順番で計算を行う．

1. 粒子の初期状態（位置，運動量）を設定する．
2. ポアソン方程式を解いてメッシュの各頂点での電位を計算し，そこから各電場の強さ E を求める．
3. 電場 E_z^n，E_r^n から，磁場 $H_\theta^{n+1/2}$ を (2.20) 式を用いて計算する．
4. 電場 E_z^n，E_r^n と磁場 $H_\theta^{n-1/2}$，$H_\theta^{n+1/2}$ から，粒子の運動量 $p_z^{n+1/2}$，$p_r^{n+1/2}$ を (2.32)〜(2.34) 式を用いて計算する．
5. 運動量 $p_z^{n+1/2}$，$p_r^{n+1/2}$ から，粒子の位置 z^{n+1}，r^{n+1} を (2.36) 式を用いて計算する．
6. 運動量 $p_z^{n+1/2}$，$p_r^{n+1/2}$ から，電流密度 $j_z^{n+1/2}$，$j_r^{n+1/2}$ を 2.3.2 節で述べた方法を用いて計算する．
7. 磁場 $H_\theta^{n+1/2}$ と電流密度 $j_z^{n+1/2}$，$j_r^{n+1/2}$ から，電場 E_z^{n+1}，E_r^{n+1} を (2.25)，(2.30) 式を用いて計算する．
8. n を増やし，3.〜7. を繰り返し計算していく．

詳しい内容については，付録 A に示す．

第4章　計算結果

今回は，図 4.1 に示す 2 次元軸対称モデルの加速管中を電子ビームが通過したときの電磁場について解析を行った．

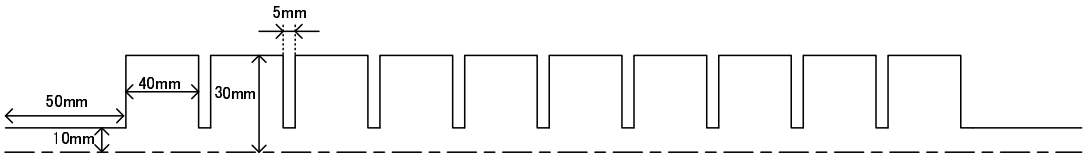

図 4.1: 解析するモデル

電子ビームはバンチ・ビーム（電子の塊）で，そのビーム長は 10.0$[mm]$，ビーム半径は 2.5$[mm]$，エネルギーは 100$[MeV]$ である．

左右端には吸収境界条件，それ以外には完全導体境界条件を適用した．

図 4.2 に磁場の変化の様子を，図 4.3 に電場の変化の様子を示す．

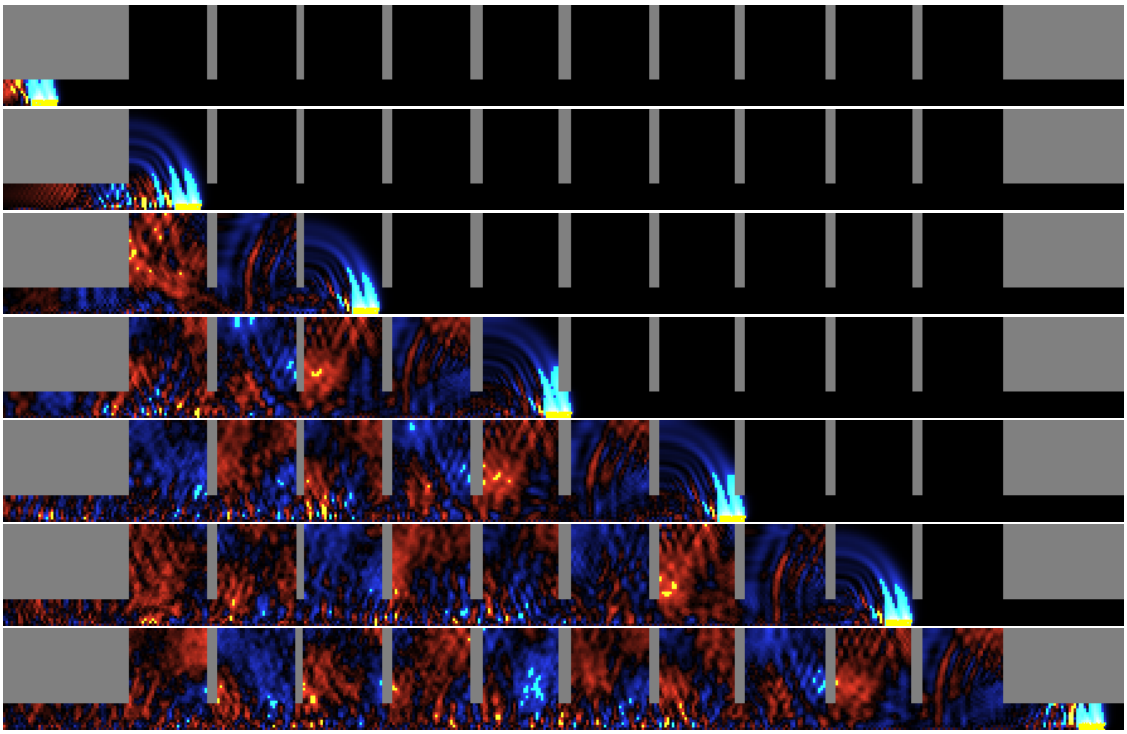

図 4.2: 磁場解析結果

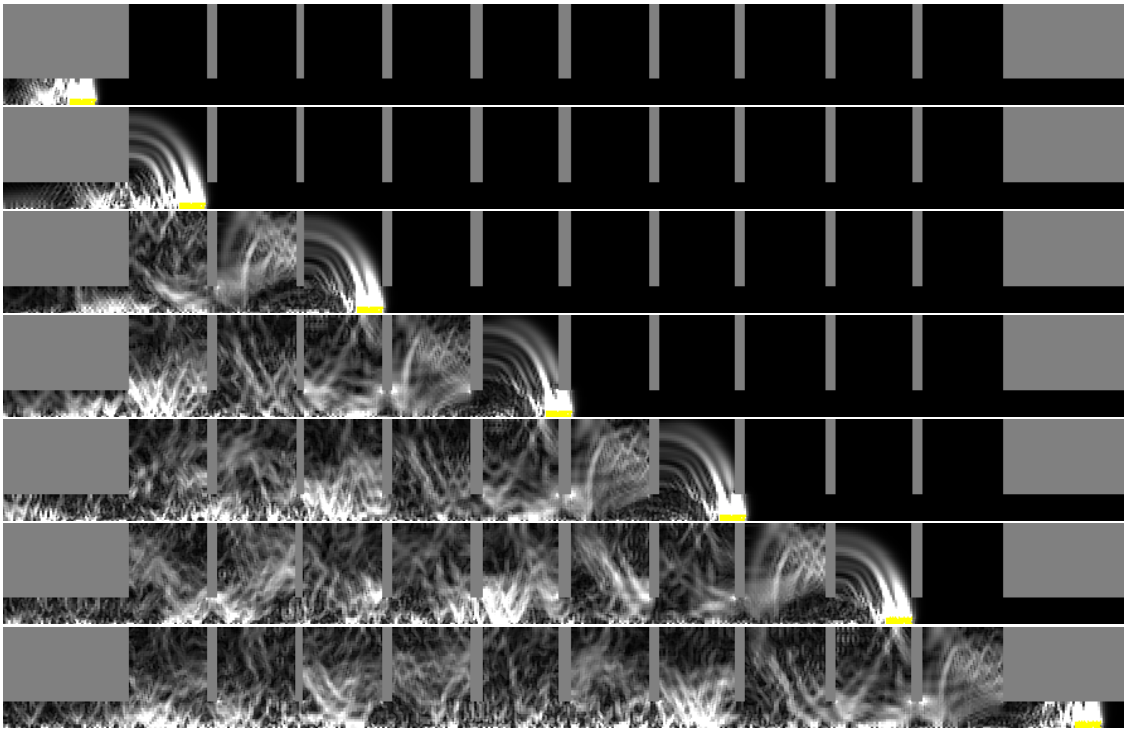

図 4.3: 電場解析結果

第5章　考察と今後の課題

結果からは，電子ビームによって加速管内に電磁場が誘起されている様子が良く見て取れる．電子の進行方向に電磁場が現れないのは，電子がほぼ光速で運動しているためである．また，導体面では電磁場の波が反射していることが見て取れた．

この電磁場の推移の様子は，加速器に関する文献 [3] の図と良く似ていた．このことから，このプログラムによるシミュレーション結果に大きな間違いは無いと考えている．詳しい計算精度の検証は今後の課題である．

今回の計算では一次近似の式を用いたが，精度を上げるためにはより高次の計算式を用いる必要がある．また，曲線を含むような複雑な形状について解析を行う場合には，今回のような正方形で領域を分割する方法では誤差が大きくなってしまう．これを改善するためには，より形状精度の高い三角形による分割を行う必要がある．

作成したプログラムは各種パラメータを変える際にプログラムコードを直接書き換えなければならない仕様になっていて不便なので，GUI による入出力インターフェイスを設けることも今後の課題である．

第6章　結言

- 本研究では，粒子コードを用いて2次元軸対称モデルの電磁場と荷電粒子の相互作用を解析するために式の変形などを行った．
- 実際に作成したプログラムでは，おおよそ正しい結果を得ることに成功した．詳しい計算精度の検証は今後の課題である．
- プログラムの改善点として，三角形分割，より高次の計算式の使用，GUIへの対応，などが挙げられる．

謝辞

ご指導して下さった担当教官の山本昌志先生，様々な面でご協力していただいた研究室の皆さんに感謝の意を表します.

関連図書

[1] 川田 重夫・松本 正己 共著『電磁気学-電磁気現象のコンピュータシミュレーション入門-』シミュレーション物理学 (1)(近代科学社,1993)

[2] Takayuki Umeda,Yoshiharu Omura,and Hiroshi Matsumoto. *Charge conservation methods for computing current densities in electromagnetic particle-in-cell simulatioons.* 2005.

[3] 竹田 誠之 他 著『OHO'90 高エネルギー加速器セミナー』(1990)

付 録A　計算プログラム

リスト A.1: 作成した電場解析のプログラム

```
1  #include<iostream>
2  #include<math.h>
3  #include<stdlib.h>
4  #include<time.h>
5  #include<GL/glut.h>
6
7  #define I 1000
8  #define P 1000000
9
10 using namespace std;
11
12 void display (void); //表示
13 void data_in (void); //初期値設定
14 void init_efield (void); //初期電界計算
15 void sor (void); //SOR法
16 void mfield (void); //磁場計算
17 void velocity (void); //速度計算
18 void position (void); //座標計算
19 void current (void); //電流計算
20 void efield (void); //電場計算
21 void emission (void); //電子放出
22 double color (double data);//色の指定
23
24 double ez[I][I]; //z方向の電場
25 double er[I][I]; //r方向の電場
26 double ht[I][I]; //t(シータ)方向の磁場
27
28 double pzp[P]; //粒子のz方向運動量
29 double prp[P]; //粒子のr方向運動量
30 double zp[P]; //粒子のz座標
31 double rp[P]; //粒子のr座標
32 double qp[P]; //粒子一周分の電荷
33
34 double jz[I][I]; //z方向の電流
35 double jr[I][I]; //r方向の電流
36 double phi[I][I]; //静電ポテンシャル
37 double hta[I][I]; //磁場平均
38
39 double zpb[P]; //前に粒子がいた位置(z座標)
40 double rpb[P]; //前に粒子がいた位置(r座標)
41
42 int f[I][I]; //=1なら管外および境界
43
44 double dl; //メッシュ間隔
45 double dt; //時間間隔
46
47 int npt; //総粒子数
48 int nz; //z方向の格子点数
```

```
49  int nr; //r方向の格子点数
50  int it_end; //最大計算回数
51  double z_max; //z最大値
52  double r_max; //r最大値
53  double q; //粒子1個の電荷量
54  double Q; //粒子一周分の電荷量（最も外側での）
55  double m0; //粒子の静止質量
56  double c; //光速
57  double myu; //透磁率
58  double eps; //誘電率
59  double v0; //管壁の電位
60  double vb; //ビームの加速電圧
61  int p_emission; //dt秒間に放出する粒子数
62  int flag; //=1ならば、これ以上総粒子数は増えない
63  double lb;//ビーム長さ
64  double rb;//ビーム半径
65  int n_over; //通過した粒子の数
66  int nd;//減衰領域数
67  double ld;//減衰領域長さ
68
69  int main(int argc,char* argv[])
70  {
71    data_in();
72
73    init_efield();
74
75    glutInit(&argc,argv);
76    glutInitDisplayMode(GLUT_RGBA | GLUT_DOUBLE);
77    glutInitWindowSize(890,80);
78    glutCreateWindow(argv[0]);
79    glutDisplayFunc(display);
80    glutMainLoop();
81
82    return 0;
83  }
84
85  /*表示*/
86  void display ()
87  {
88    int it; //計算回数
89    int i,j;
90    double z,r;
91
92    for(it=1;it<=it_end;it++){
93
94      mfield();
95
96      velocity();
97
98      position();
99
100     current();
101
102     efield();
103
104     if(((it-1)%10)==0){
105       glClear(GL_COLOR_BUFFER_BIT);
106
107       for(i=nd+1;i<=nz+nd-1;i++){
108         for(j=1;j<=nr-1;j++){
109           /*電界の強さ表示*/
```

```
110             if(f[i][j]==1&&f[i+1][j]==1&&f[i][j+1]==1&&f[i+1][j+1]==1){
111               glColor3d(0.5,0.5,0.5);
112               glBegin(GL_QUADS);
113               glVertex2d((i-nd-1)*dl*2.0/z_max-1.0,(j-1)*dl*2.0/r_max-1.0);
114               glVertex2d((i-nd)*dl*2.0/z_max-1.0,(j-1)*dl*2.0/r_max-1.0);
115               glVertex2d((i-nd)*dl*2.0/z_max-1.0,j*dl*2.0/r_max-1.0);
116               glVertex2d((i-nd-1)*dl*2.0/z_max-1.0,j*dl*2.0/r_max-1.0);
117               glEnd();
118             }else{
119               glColor3d(color(sqrt(pow(ez[i][j],2)+pow(er[i][j],2)))
120                         ,color(sqrt(pow(ez[i][j],2)+pow(er[i][j],2)))
121                         ,color(sqrt(pow(ez[i][j],2)+pow(er[i][j],2))));
122               glBegin(GL_QUADS);
123               glVertex2d((i-nd-1)*dl*2.0/z_max-1.0,(j-1)*dl*2.0/r_max-1.0);
124               glVertex2d((i-nd-0.5)*dl*2.0/z_max-1.0,(j-1)*dl*2.0/r_max-1.0);
125               glVertex2d((i-nd-0.5)*dl*2.0/z_max-1.0,(j-0.5)*dl*2.0/r_max-1.0);
126               glVertex2d((i-nd-1)*dl*2.0/z_max-1.0,(j-0.5)*dl*2.0/r_max-1.0);
127               glEnd();
128
129               glColor3d(color(sqrt(pow(ez[i][j],2)+pow(er[i+1][j],2)))
130                         ,color(sqrt(pow(ez[i][j],2)+pow(er[i+1][j],2)))
131                         ,color(sqrt(pow(ez[i][j],2)+pow(er[i+1][j],2))));
132               glBegin(GL_QUADS);
133               glVertex2d((i-nd-0.5)*dl*2.0/z_max-1.0,(j-1)*dl*2.0/r_max-1.0);
134               glVertex2d((i-nd)*dl*2.0/z_max-1.0,(j-1)*dl*2.0/r_max-1.0);
135               glVertex2d((i-nd)*dl*2.0/z_max-1.0,(j-0.5)*dl*2.0/r_max-1.0);
136               glVertex2d((i-nd-0.5)*dl*2.0/z_max-1.0,(j-0.5)*dl*2.0/r_max-1.0);
137               glEnd();
138
139               glColor3d(color(sqrt(pow(ez[i][j+1],2)+pow(er[i][j],2)))
140                         ,color(sqrt(pow(ez[i][j+1],2)+pow(er[i][j],2)))
141                         ,color(sqrt(pow(ez[i][j+1],2)+pow(er[i][j],2))));
142               glBegin(GL_QUADS);
143               glVertex2d((i-nd-1)*dl*2.0/z_max-1.0,(j-0.5)*dl*2.0/r_max-1.0);
144               glVertex2d((i-nd-0.5)*dl*2.0/z_max-1.0,(j-0.5)*dl*2.0/r_max-1.0);
145               glVertex2d((i-nd-0.5)*dl*2.0/z_max-1.0,j*dl*2.0/r_max-1.0);
146               glVertex2d((i-nd-1)*dl*2.0/z_max-1.0,j*dl*2.0/r_max-1.0);
147               glEnd();
148
149               glColor3d(color(sqrt(pow(ez[i][j+1],2)+pow(er[i+1][j],2)))
150                         ,color(sqrt(pow(ez[i][j+1],2)+pow(er[i+1][j],2)))
151                         ,color(sqrt(pow(ez[i][j+1],2)+pow(er[i+1][j],2))));
152               glBegin(GL_QUADS);
153               glVertex2d((i-nd-0.5)*dl*2.0/z_max-1.0,(j-0.5)*dl*2.0/r_max-1.0);
154               glVertex2d((i-nd)*dl*2.0/z_max-1.0,(j-0.5)*dl*2.0/r_max-1.0);
155               glVertex2d((i-nd)*dl*2.0/z_max-1.0,j*dl*2.0/r_max-1.0);
156               glVertex2d((i-nd-0.5)*dl*2.0/z_max-1.0,j*dl*2.0/r_max-1.0);
157               glEnd();
158             }
159           }
160         }
161         /*粒子の位置表示*/
162         glColor3d(1.0,1.0,0.0);
163         for(i=1;i<=npt;i++){
164           glBegin(GL_POINTS);
165           z=(zp[i]-ld)*2.0/z_max-1.0;
166           r=rp[i]*2.0/r_max-1.0;
167           glVertex2d(z,r);
168           glEnd();
169         }
170         glutSwapBuffers();
```

```
171       }
172       if(flag==0){
173         emission();
174       }
175     }
176
177     return;
178   }
179
180   /*色指定*/
181   double color(double data)
182   {
183     data=data/2000;
184     if(data>1.0){
185       data=1.0;
186     }else if(data<0.0){
187       data=0.0;
188     }
189     return data;
190   }
191
192   /*初期値設定*/
193   void data_in (void)
194   {
195     /*定数*/
196     q=-1.602e-19; //粒子1個の電荷量
197     Q=-1.602e-12; //粒子一周分の電荷量(一番外側での)
198     m0=9.1093897e-31; //粒子の静止質量
199     c=299792458; //光速
200     myu=1.25663706e-6; //透磁率
201     eps=8.854187816e-12; //誘電率
202
203     dl=1.0e-3; //メッシュ間隔
204     dt=5.0e-13; //時間間隔
205
206     npt=0; //総粒子数(最初は0に設定)
207     nz=445; //z方向のメッシュ数
208     nr=40; //r方向のメッシュ数
209
210     it_end=10000001; //最大計算回数
211
212     z_max=dl*(nz-1.0); //z最大値
213     r_max=dl*(nr-1.0); //r最大値
214
215     v0=0.0; //管壁の電位
216
217     vb=100.0e6;
218     lb=10.0e-3;
219     rb=2.5e-3;
220
221     p_emission=5; //dt秒間に放出する粒子数
222     flag=0;
223     n_over=0;
224
225     nd=50;
226     ld=nd*dl;
227
228     /*各値を0に初期化*/
229     int i,j;
230     for(i=0;i<=nz+2*nd+1;i++){
231       for(j=0;j<=nr+1;j++){
```

```
232         ez[i][j]=0.0;
233         er[i][j]=0.0;
234         ht[i][j]=0.0;
235         jz[i][j]=0.0;
236         jr[i][j]=0.0;
237         phi[i][j]=0.0;
238         hta[i][j]=0.0;
239         f[i][j]=0;
240       }
241     }
242
243     /*粒子の初期設定*/
244     double rr,zz;
245     for(i=1;i<=(P-1);i++){
246       zp[i]=ld-nd*dl/5;
247       rr=rand()%1001;
248       rp[i]=rr/1000.0*rb;
249
250       pzp[i]=sqrt(fabs(2*m0*q*vb));
251       prp[i]=0.0;
252
253       qp[i]=Q*rp[i]/r_max+q;
254     }
255
256     return;
257 }
258
259 /*初期電界計算*/
260 void init_efield (void)
261 {
262     int i,j;
263
264     /*境界条件*/
265     for(i=1;i<=nz+2*nd;i++){
266       phi[i][nr]=v0;
267       f[i][nr]=1;
268     }
269     for(i=1;i<=nz+2*nd;i++){
270       for(j=1;j<=nr;j++){
271         if(((j-1)*dl>=10.0e-3)&&
272             ((i-1)*dl<=50.0e-3+ld||
273               (80.0e-3+ld<=(i-1)*dl&&(i-1)*dl<=85.0e-3+ld)||
274               (115.0e-3+ld<=(i-1)*dl&&(i-1)*dl<=120.0e-3+ld)||
275               (150.0e-3+ld<=(i-1)*dl&&(i-1)*dl<=155.0e-3+ld)||
276               (185.0e-3+ld<=(i-1)*dl&&(i-1)*dl<=190.0e-3+ld)||
277               (220.0e-3+ld<=(i-1)*dl&&(i-1)*dl<=225.0e-3+ld)||
278               (255.0e-3+ld<=(i-1)*dl&&(i-1)*dl<=260.0e-3+ld)||
279               (290.0e-3+ld<=(i-1)*dl&&(i-1)*dl<=295.0e-3+ld)||
280               (325.0e-3+ld<=(i-1)*dl&&(i-1)*dl<=330.0e-3+ld)||
281               (360.0e-3+ld<=(i-1)*dl&&(i-1)*dl<=365.0e-3+ld)||
282               (395.0e-3+ld<=(i-1)*dl))){
283           phi[i][j]=v0;
284           f[i][j]=1;
285         }
286       }
287     }
288
289     sor();//SOR法
290
291     /*電界=電位差/距離*/
292     for(i=1;i<=nz+2*nd-1;i++){
```

```
293       for(j=1;j<=nr;j++){
294         ez[i][j]=(phi[i][j]-phi[i+1][j])/dl;
295       }
296     }
297     for(i=1;i<=nz+2*nd;i++){
298       for(j=1;j<=nr-1;j++){
299         er[i][j]=(phi[i][j]-phi[i][j+1])/dl;
300       }
301     }
302
303     return;
304   }
305
306   /*SOR法*/
307   void sor (void)
308   {
309     double phi1,phi2,omega;
310     double sa,total;
311     double e;
312     int i,j,order;
313
314     omega=1.9;
315     e=1.0e-15; //誤差
316     order=0;//0:正順  1:逆順
317
318     while(1){
319       sa=0.0;
320       total=0.0;
321       if(order==0){
322         for(i=1;i<=nz+2*nd;i++){
323           for(j=1;j<=nr;j++){
324             if(f[i][j]==0){
325               if(j==1){
326                 phi1=0.25*(2*phi[i][2]+phi[i-1][1]+phi[i+1][1]);
327               }else{
328                 phi1=0.25*(phi[i][j-1]+phi[i][j+1]+phi[i-1][j]+phi[i+1][j]);
329               }
330               phi2=phi[i][j]+omega*(phi1-phi[i][j]);
331               sa+=fabs(phi2-phi[i][j]);
332               total+=fabs(phi2);
333               phi[i][j]=phi2;
334             }
335           }
336         }
337         if(total==0){
338           total=e;
339         }
340         if(sa/total<e){
341           break;
342         }
343         order=1;
344       }else{
345         for(i=nz+2*nd;i>=1;i--){
346           for(j=nr;j>=1;j--){
347             if(f[i][j]==0){
348               if(j==1){
349                 phi1=0.25*(2*phi[i][2]+phi[i-1][1]+phi[i+1][1]);
350               }else{
351                 phi1=0.25*(phi[i][j-1]+phi[i][j+1]+phi[i-1][j]+phi[i+1][j]);
352               }
353               phi2=phi[i][j]+omega*(phi1-phi[i][j]);
```

```
354                 sa+=fabs(phi2-phi[i][j]);
355                 total+=fabs(phi2);
356                 phi[i][j]=phi2;
357               }
358             }
359           }
360           if(total==0){
361             total=e;
362           }
363           if(sa/total<e){
364             break;
365           }
366           order=0;
367         }
368       }
369
370       return;
371     }
372
373     /*磁場計算*/
374     void mfield (void)
375     {
376       int i,j;
377       for(i=1;i<=nz+2*nd-1;i++){
378         for(j=1;j<=nr-1;j++){
379           hta[i][j]=ht[i][j]
380             +(er[i][j]-er[i+1][j]+ez[i][j+1]-ez[i][j])/(2.0*dl*myu)*dt;
381           ht[i][j]+=(er[i][j]-er[i+1][j]+ez[i][j+1]-ez[i][j])/(dl*myu)*dt;
382         }
383       }
384
385       return;
386     }
387
388     /*速度計算*/
389     void velocity (void)
390     {
391       int n,z,r,z_,r_;
392       double z1,z2,r1,r2,z_1,z_2,r_1,r_2;
393       double s1,s2,s3,s4;
394       double epz,epr,hpt,p1z,p1r,p2z,p2r;
395
396       for(n=1;n<=npt;n++){
397
398         /*z方向の電界計算*/
399         z_=(int)(zp[n]/dl+0.5);
400         z_1=(double)z_*dl;
401         z_2=z_1+dl; //z_1とz_2の間に粒子が存在
402         r=(int)(rp[n]/dl);
403         r1=(double)r*dl;
404         r2=r1+dl; //y1とy2の間に粒子が存在
405         s1=(zp[n]-(z_1-0.5*dl))*(rp[n]-r1);
406         s2=((z_2-0.5*dl)-zp[n])*(rp[n]-r1);
407         s3=(zp[n]-(z_1-0.5*dl))*(r2-rp[n]);
408         s4=((z_2-0.5*dl)-zp[n])*(r2-rp[n]);
409         if(z_==0){
410           epz=((s4+s3)*ez[1][r+1]+(s2+s1)*ez[1][r+2])/(dl*dl);
411         }else{
412           epz=(s4*ez[z_][r+1]+s3*ez[z_+1][r+1]
413                +s2*ez[z_][r+2]+s1*ez[z_+1][r+2])/(dl*dl);
414         }
```

```
415
416       /* r方向の電界計算 */
417       z=(int)(zp[n]/dl);
418       z1=(double)z*dl;
419       z2=z1+dl;
420       r_=(int)(rp[n]/dl+0.5);
421       r_1=(double)r_*dl;
422       r_2=r_1+dl;
423       s1=(zp[n]-z1)*(rp[n]-(r_1-0.5*dl));
424       s2=(z2-zp[n])*(rp[n]-(r_1-0.5*dl));
425       s3=(zp[n]-z1)*((r_2-0.5*dl)-rp[n]);
426       s4=(z2-zp[n])*((r_2-0.5*dl)-rp[n]);
427       if(r_==0){
428         epr=((s2-s4)*er[z+1][1]+(s1-s3)*er[z+2][1])/(dl*dl);
429       }else{
430         epr=(s4*er[z+1][r_]+s3*er[z+2][r_]
431               +s2*er[z+1][r_+1]+s1*er[z+2][r_+1])/(dl*dl);
432       }
433
434       /*磁界計算*/
435       s1=(zp[n]-(z_1-0.5*dl))*(rp[n]-(r_1-0.5*dl));
436       s2=((z_2-0.5*dl)-zp[n])*(rp[n]-(r_1-0.5*dl));
437       s3=(zp[n]-(z_1-0.5*dl))*((r_2-0.5*dl)-rp[n]);
438       s4=((z_2-0.5*dl)-zp[n])*((r_2-0.5*dl)-rp[n]);
439       if((r_==0)&&(z_==0)){
440         hpt=((s1+s2-s3-s4)*hta[1][1])/(dl*dl);
441       }else if(r_==0){
442         hpt=((s2-s4)*hta[z_][1]+(s1-s3)*hta[z_+1][1])/(dl*dl);
443       }else if(z_==0){
444         hpt=((s4+s3)*hta[1][r_]+(s2+s1)*hta[1][r_+1])/(dl*dl);
445       }else{
446         hpt=(s4*hta[z_][r_]+s3*hta[z_+1][r_]
447               +s2*hta[z_][r_+1]+s1*hta[z_+1][r_+1])/(dl*dl);
448       }
449
450       /*運動量計算*/
451       p1z=pzp[n]+q*epz*dt/2.0;
452       p1r=prp[n]+q*epr*dt/2.0;
453
454       p2z=p1z+myu*q*c*p1r*hpt*dt/sqrt(m0*m0*c*c+p1z*p1z+p1r*p1r);
455       p2r=p1r-myu*q*c*p1z*hpt*dt/sqrt(m0*m0*c*c+p1z*p1z+p1r*p1r);
456
457       pzp[n]=p2z+q*epz*dt/2.0;
458       prp[n]=p2r+q*epr*dt/2.0;
459     }
460
461     return;
462   }
463
464   /*座標計算*/
465   void position (void)
466   {
467
468     int n;
469
470     for(n=1;n<=npt;n++){
471       zpb[n]=zp[n];
472       rpb[n]=rp[n];
473
474       zp[n]+=c*pzp[n]*dt/sqrt(m0*m0*c*c+pzp[n]*pzp[n]+prp[n]*prp[n]);
475       rp[n]+=c*prp[n]*dt/sqrt(m0*m0*c*c+pzp[n]*pzp[n]+prp[n]*prp[n]);
```

```
476
477         if(zp[n]>=lb+ld-nd*dl/5.0){
478           flag=1;
479         }
480       }
481
482       return;
483
484   }
485
486   /*電流計算*/
487   void current (void)
488   {
489
490       int n,i,j;
491       int i1,j1,i2,j2,i11,j11,i22,j22;
492       double zrp,rrp;
493       double wi,wj,wi1,wi2,wi3,wj1,wj2,wj3;
494       double fz1,fz2,fr1,fr2;
495       double vz1,vz2,vr1,vr2;
496
497       /*0に初期化*/
498       for(i=0;i<=nz+2*nd+2;i++){
499         for(j=0;j<=nr+3;j++){
500           jz[i][j]=0.0;
501         }
502       }
503       for(i=0;i<=nz+2*nd+3;i++){
504         for(j=0;j<=nr+2;j++){
505           jr[i][j]=0.0;
506         }
507       }
508
509       for(n=1;n<=npt;n++){
510
511         i1=(int)(zpb[n]/dl);
512         i2=(int)(zp[n]/dl);
513         j1=(int)(rpb[n]/dl);
514         j2=(int)(rp[n]/dl);
515
516         /*relay pointの設定*/
517         if(i1==i2){
518           zrp=(zpb[n]+zp[n])/2.0;
519         }else{
520           if(i1>i2){
521             zrp=i1*dl;
522           }else{
523             zrp=i2*dl;
524           }
525         }
526         if(j1==j2){
527           rrp=(rpb[n]+rp[n])/2.0;
528         }else{
529           if(j1>j2){
530             rrp=j1*dl;
531           }else{
532             rrp=j2*dl;
533           }
534         }
535
536         /*from(z1,r1)to(zr,rr)*/
```

```
537
538         i11=(int)(zpb[n]/dl+0.5);
539         j11=(int)(rpb[n]/dl+0.5);
540
541         /*velocity*/
542         vz1=(zrp-zpb[n])/dt;
543         vr1=(rrp-rpb[n])/dt;
544
545         /*first-order shape factor*/
546         wi=(zpb[n]+zrp)/(2*dl)-i11;
547         wj=(rpb[n]+rrp)/(2*dl)-j11;
548
549         /*charge flux*/
550         fz1=qp[n]*vz1*(0.5-wi);
551         fz2=qp[n]*vz1*(0.5+wi);
552         fr1=qp[n]*vr1*(0.5-wj);
553         fr2=qp[n]*vr1*(0.5+wj);
554
555         /*second-order shape factor*/
556         wi1=0.5*pow((0.5-wi),2);
557         wi2=0.75-pow(wi,2);
558         wi3=0.5*pow((0.5+wi),2);
559         wj1=0.5*pow((0.5-wj),2);
560         wj2=0.75-pow(wj,2);
561         wj3=0.5*pow((0.5+wj),2);
562
563         /*current density*/
564         if(j11==0){
565           if(i11==0){
566             jz[1][2]+=(fz1+fz2)*(wj1+wj3)/(M_PI*dl*dl*2.0*dl);
567             jz[1][1]+=(fz1+fz2)*wj2/(M_PI*0.25*dl*dl*dl);
568             jr[2][1]-=(fr2-fr1)*wi1/(M_PI*dl*dl*dl);
569             jr[1][1]+=(fr1-fr2)*wi2/(M_PI*dl*dl*dl);
570           }else if(i11==nz+2*nd-1){
571             jz[nz+2*nd-1][2]+=(fz1+fz2)*(wj1+wj3)/(M_PI*dl*dl*2.0*dl);
572             jz[nz+2*nd-1][1]+=(fz1+fz2)*wj2/(M_PI*0.25*dl*dl*dl);
573             jr[nz+2*nd-1][1]-=(fr2-fr1)*wi3/(M_PI*dl*dl*dl);
574             jr[nz+2*nd][1]+=(fr1-fr2)*wi2/(M_PI*dl*dl*dl);
575           }else{
576             jz[i11][2]+=fz1*(wj1+wj3)/(M_PI*dl*dl*2.0*dl);
577             jz[i11][1]+=fz1*wj2/(M_PI*0.25*dl*dl*dl);
578             jz[i11+1][2]+=fz2*(wj1+wj3)/(M_PI*dl*dl*2.0*dl);
579             jz[i11+1][1]+=fz2*wj2/(M_PI*0.25*dl*dl*dl);
580           }
581           jr[i11][1]+=(fr2-fr1)*wi1/(M_PI*dl*dl*dl);
582           jr[i11+1][1]+=(fr2-fr1)*wi2/(M_PI*dl*dl*dl);
583           jr[i11+2][1]+=(fr2-fr1)*wi3/(M_PI*dl*dl*dl);
584         }else if(j11==1){
585           if(i11==0){
586             jz[1][1]+=(fz1+fz2)*wj1/(M_PI*0.25*dl*dl*dl);
587             jz[1][2]+=(fz1+fz2)*wj2/(M_PI*dl*dl*2.0*dl);
588             jz[1][3]+=(fz1+fz2)*wj3/(M_PI*dl*dl*4.0*dl);
589             jr[2][1]-=fr1*wi1/(M_PI*dl*dl*dl);
590             jr[1][1]-=fr1*wi2/(M_PI*dl*dl*dl);
591             jr[2][2]-=fr2*wi1/(M_PI*dl*dl*3.0*dl);
592             jr[1][2]-=fr2*wi2/(M_PI*dl*dl*3.0*dl);
593           }else if(i11==nz+2*nd-1){
594             jz[nz+2*nd-1][1]+=(fz1+fz2)*wj1/(M_PI*0.25*dl*dl*dl);
595             jz[nz+2*nd-1][2]+=(fz1+fz2)*wj2/(M_PI*dl*dl*2.0*dl);
596             jz[nz+2*nd-1][3]+=(fz1+fz2)*wj3/(M_PI*dl*dl*4.0*dl);
597             jr[nz+2*nd-1][1]-=fr1*wi3/(M_PI*dl*dl*dl);
```

```
598             jr[nz+2*nd][1]-=fr1*wi2/(M_PI*dl*dl*dl);
599             jr[nz+2*nd-1][2]-=fr2*wi3/(M_PI*dl*dl*3.0*dl);
600             jr[nz+2*nd][2]-=fr2*wi2/(M_PI*dl*dl*3.0*dl);
601           }else{
602             jz[i11][1]+=fz1*wj1/(M_PI*0.25*dl*dl*dl);
603             jz[i11][2]+=fz1*wj2/(M_PI*dl*dl*2.0*dl);
604             jz[i11][3]+=fz1*wj3/(M_PI*dl*dl*4.0*dl);
605             jz[i11+1][1]+=fz2*wj1/(M_PI*0.25*dl*dl*dl);
606             jz[i11+1][2]+=fz2*wj2/(M_PI*dl*dl*2.0*dl);
607             jz[i11+1][3]+=fz2*wj3/(M_PI*dl*dl*4.0*dl);
608           }
609           jr[i11][1]+=fr1*wi1/(M_PI*dl*dl*dl);
610           jr[i11+1][1]+=fr1*wi2/(M_PI*dl*dl*dl);
611           jr[i11+2][1]+=fr1*wi3/(M_PI*dl*dl*dl);
612
613           jr[i11][2]+=fr2*wi1/(M_PI*dl*dl*3.0*dl);
614           jr[i11+1][2]+=fr2*wi2/(M_PI*dl*dl*3.0*dl);
615           jr[i11+2][2]+=fr2*wi3/(M_PI*dl*dl*3.0*dl);
616         }else{
617           if(i11==0){
618             jz[1][j11]+=(fz1+fz2)*wj1/(M_PI*dl*dl*2.0*(j11-1)*dl);
619             jz[1][j11+1]+=(fz1+fz2)*wj2/(M_PI*dl*dl*2.0*j11*dl);
620             jz[1][j11+2]+=(fz1+fz2)*wj3/(M_PI*dl*dl*2.0*(j11+1)*dl);
621             jr[2][j11]-=fr1*wi1/(M_PI*dl*dl*2.0*(j11-0.5)*dl);
622             jr[1][j11]-=fr1*wi2/(M_PI*dl*dl*2.0*(j11-0.5)*dl);
623             jr[2][j11+1]-=fr2*wi1/(M_PI*dl*dl*2.0*(j11+0.5)*dl);
624             jr[1][j11+1]-=fr2*wi2/(M_PI*dl*dl*2.0*(j11+0.5)*dl);
625           }else if(i11==nz+2*nd-1){
626             jz[nz+2*nd-1][j11]+=(fz1+fz2)*wj1/(M_PI*dl*dl*2.0*(j11-1)*dl);
627             jz[nz+2*nd-1][j11+1]+=(fz1+fz2)*wj2/(M_PI*dl*dl*2.0*j11*dl);
628             jz[nz+2*nd-1][j11+2]+=(fz1+fz2)*wj3/(M_PI*dl*dl*2.0*(j11+1)*dl);
629             jr[nz+2*nd-1][j11]-=fr1*wi3/(M_PI*dl*dl*2.0*(j11-0.5)*dl);
630             jr[nz+2*nd][j11]-=fr1*wi2/(M_PI*dl*dl*2.0*(j11-0.5)*dl);
631             jr[nz+2*nd-1][j11+1]-=fr2*wi3/(M_PI*dl*dl*2.0*(j11+0.5)*dl);
632             jr[nz+2*nd][j11+1]-=fr2*wi2/(M_PI*dl*dl*2.0*(j11+0.5)*dl);
633           }else{
634             jz[i11][j11]+=fz1*wj1/(M_PI*dl*dl*2.0*(j11-1)*dl);
635             jz[i11][j11+1]+=fz1*wj2/(M_PI*dl*dl*2.0*j11*dl);
636             jz[i11][j11+2]+=fz1*wj3/(M_PI*dl*dl*2.0*(j11+1)*dl);
637             jz[i11+1][j11]+=fz2*wj1/(M_PI*dl*dl*2.0*(j11-1)*dl);
638             jz[i11+1][j11+1]+=fz2*wj2/(M_PI*dl*dl*2.0*j11*dl);
639             jz[i11+1][j11+2]+=fz2*wj3/(M_PI*dl*dl*2.0*(j11+1)*dl);
640             jr[i11+1][j11]+=fr1*wi2/(M_PI*dl*dl*2.0*(j11-0.5)*dl);
641             jr[i11+1][j11+1]+=fr2*wi2/(M_PI*dl*dl*2.0*(j11+0.5)*dl);
642           }
643           jr[i11][j11]+=fr1*wi1/(M_PI*dl*dl*2.0*(j11-0.5)*dl);
644           jr[i11+1][j11]+=fr1*wi2/(M_PI*dl*dl*2.0*(j11-0.5)*dl);
645           jr[i11+2][j11]+=fr1*wi3/(M_PI*dl*dl*2.0*(j11-0.5)*dl);
646
647           jr[i11][j11+1]+=fr2*wi1/(M_PI*dl*dl*2.0*(j11+0.5)*dl);
648           jr[i11+1][j11+1]+=fr2*wi2/(M_PI*dl*dl*2.0*(j11+0.5)*dl);
649           jr[i11+2][j11+1]+=fr2*wi3/(M_PI*dl*dl*2.0*(j11+0.5)*dl);
650         }
651
652         /*from(zr,rr)to(z2,r2)*/
653
654         i22=(int)(zrp/dl+0.5);
655         j22=(int)(rrp/dl+0.5);
656
657         /*velocity*/
658         vz2=(zp[n]-zrp)/dt;
```

```
659         vr2=(rp[n]-rrp)/dt;
660
661         /*first-order shape factor*/
662         wi=(zrp+zp[n])/(2*dl)-i22;
663         wj=(rrp+rp[n])/(2*dl)-j22;
664
665         /*charge flux*/
666         fz1=qp[n]*vz2*(0.5-wi);
667         fz2=qp[n]*vz2*(0.5+wi);
668         fr1=qp[n]*vr2*(0.5-wj);
669         fr2=qp[n]*vr2*(0.5+wj);
670
671         /*second-order shape factor*/
672         wi1=0.5*pow((0.5-wi),2);
673         wi2=0.75-pow(wi,2);
674         wi3=0.5*pow((0.5+wi),2);
675         wj1=0.5*pow((0.5-wj),2);
676         wj2=0.75-pow(wj,2);
677         wj3=0.5*pow((0.5+wj),2);
678
679         /*current density*/
680         if(j22==0){
681           if(i22==0){
682             jz[1][2]+=(fz1+fz2)*(wj1+wj3)/(M_PI*dl*dl*2.0*dl);
683             jz[1][1]+=(fz1+fz2)*wj2/(M_PI*0.25*dl*dl*dl);
684             jr[1][1]+=(fr1-fr2)*wi2/(M_PI*dl*dl*dl);
685             jr[2][1]-=(fr2-fr1)*wi1/(M_PI*dl*dl*dl);
686           }else if(i22==nz+2*nd-1){
687             jz[nz+2*nd-1][2]+=(fz1+fz2)*(wj1+wj3)/(M_PI*dl*dl*2.0*dl);
688             jz[nz+2*nd-1][1]+=(fz1+fz2)*wj2/(M_PI*0.25*dl*dl*dl);
689             jr[nz+2*nd][1]+=(fr1-fr2)*wi2/(M_PI*dl*dl*dl);
690             jr[nz+2*nd-1][1]-=(fr2-fr1)*wi3/(M_PI*dl*dl*dl);
691           }else{
692             jz[i22][2]+=fz1*(wj1+wj3)/(M_PI*dl*dl*2.0*dl);
693             jz[i22][1]+=fz1*wj2/(M_PI*0.25*dl*dl*dl);
694             jz[i22+1][2]+=fz2*(wj1+wj3)/(M_PI*dl*dl*2.0*dl);
695             jz[i22+1][1]+=fz2*wj2/(M_PI*0.25*dl*dl*dl);
696           }
697           jr[i22][1]+=(fr2-fr1)*wi1/(M_PI*dl*dl*dl);
698           jr[i22+1][1]+=(fr2-fr1)*wi2/(M_PI*dl*dl*dl);
699           jr[i22+2][1]+=(fr2-fr1)*wi3/(M_PI*dl*dl*dl);
700         }else if(j22==1){
701           if(i22==0){
702             jz[1][1]+=(fz1+fz2)*wj1/(M_PI*0.25*dl*dl*dl);
703             jz[1][2]+=(fz1+fz2)*wj2/(M_PI*dl*dl*2.0*dl);
704             jz[1][3]+=(fz1+fz2)*wj3/(M_PI*dl*dl*4.0*dl);
705             jr[2][1]-=fr1*wi1/(M_PI*dl*dl*dl);
706             jr[1][1]-=fr1*wi2/(M_PI*dl*dl*dl);
707             jr[2][2]-=fr2*wi1/(M_PI*dl*dl*3.0*dl);
708             jr[1][2]-=fr2*wi2/(M_PI*dl*dl*3.0*dl);
709           }else if(i22==nz+2*nd-1){
710             jz[nz+2*nd-1][1]+=(fz1+fz2)*wj1/(M_PI*0.25*dl*dl*dl);
711             jz[nz+2*nd-1][2]+=(fz1+fz2)*wj2/(M_PI*dl*dl*2.0*dl);
712             jz[nz+2*nd-1][3]+=(fz1+fz2)*wj3/(M_PI*dl*dl*4.0*dl);
713             jr[nz+2*nd-1][1]-=fr1*wi1/(M_PI*dl*dl*dl);
714             jr[nz+2*nd][1]-=fr1*wi2/(M_PI*dl*dl*dl);
715             jr[nz+2*nd-1][2]-=fr2*wi1/(M_PI*dl*dl*3.0*dl);
716             jr[nz+2*nd][2]-=fr2*wi2/(M_PI*dl*dl*3.0*dl);
717           }else{
718             jz[i22][1]+=fz1*wj1/(M_PI*0.25*dl*dl*dl);
719             jz[i22][2]+=fz1*wj2/(M_PI*dl*dl*2.0*dl);
```

```
720             jz[i22][3]+=fz1*wj3/(M_PI*dl*dl*4.0*dl);
721             jz[i22+1][1]+=fz2*wj1/(M_PI*0.25*dl*dl*dl);
722             jz[i22+1][2]+=fz2*wj2/(M_PI*dl*dl*2.0*dl);
723             jz[i22+1][3]+=fz2*wj3/(M_PI*dl*dl*4.0*dl);
724           }
725           jr[i22][1]+=fr1*wi1/(M_PI*dl*dl*dl);
726           jr[i22+1][1]+=fr1*wi2/(M_PI*dl*dl*dl);
727           jr[i22+2][1]+=fr1*wi3/(M_PI*dl*dl*dl);
728
729           jr[i22][2]+=fr2*wi1/(M_PI*dl*dl*3.0*dl);
730           jr[i22+1][2]+=fr2*wi2/(M_PI*dl*dl*3.0*dl);
731           jr[i22+2][2]+=fr2*wi3/(M_PI*dl*dl*3.0*dl);
732         }else{
733           if(i22==0){
734             jz[1][j22]+=(fz1+fz2)*wj1/(M_PI*dl*dl*2.0*(j22-1)*dl);
735             jz[1][j22+1]+=(fz1+fz2)*wj2/(M_PI*dl*dl*2.0*j22*dl);
736             jz[1][j22+2]+=(fz1+fz2)*wj3/(M_PI*dl*dl*2.0*(j22+1)*dl);
737             jr[2][j22]-=fr1*wi1/(M_PI*dl*dl*2.0*(j22-0.5)*dl);
738             jr[1][j22]-=fr1*wi2/(M_PI*dl*dl*2.0*(j22-0.5)*dl);
739             jr[2][j22+1]-=fr2*wi1/(M_PI*dl*dl*2.0*(j22+0.5)*dl);
740             jr[1][j22+1]-=fr2*wi2/(M_PI*dl*dl*2.0*(j22+0.5)*dl);
741           }else if(i22==nz+2*nd-1){
742             jz[nz+2*nd-1][j22]+=(fz1+fz2)*wj1/(M_PI*dl*dl*2.0*(j22-1)*dl);
743             jz[nz+2*nd-1][j22+1]+=(fz1+fz2)*wj2/(M_PI*dl*dl*2.0*j22*dl);
744             jz[nz+2*nd-1][j22+2]+=(fz1+fz2)*wj3/(M_PI*dl*dl*2.0*(j22+1)*dl);
745             jr[nz+2*nd-1][j22]-=fr1*wi3/(M_PI*dl*dl*2.0*(j22-0.5)*dl);
746             jr[nz+2*nd][j22]-=fr1*wi2/(M_PI*dl*dl*2.0*(j22-0.5)*dl);
747             jr[nz+2*nd-1][j22+1]-=fr2*wi3/(M_PI*dl*dl*2.0*(j22+0.5)*dl);
748             jr[nz+2*nd][j22+1]-=fr2*wi2/(M_PI*dl*dl*2.0*(j22+0.5)*dl);
749           }else{
750             jz[i22][j22]+=fz1*wj1/(M_PI*dl*dl*2.0*(j22-1)*dl);
751             jz[i22][j22+1]+=fz1*wj2/(M_PI*dl*dl*2.0*j22*dl);
752             jz[i22][j22+2]+=fz1*wj3/(M_PI*dl*dl*2.0*(j22+1)*dl);
753             jz[i22+1][j22]+=fz2*wj1/(M_PI*dl*dl*2.0*(j22-1)*dl);
754             jz[i22+1][j22+1]+=fz2*wj2/(M_PI*dl*dl*2.0*j22*dl);
755             jz[i22+1][j22+2]+=fz2*wj3/(M_PI*dl*dl*2.0*(j22+1)*dl);
756             jr[i22+1][j22]+=fr1*wi2/(M_PI*dl*dl*2.0*(j22-0.5)*dl);
757             jr[i22+1][j22+1]+=fr2*wi2/(M_PI*dl*dl*2.0*(j22+0.5)*dl);
758           }
759           jr[i22][j22]+=fr1*wi1/(M_PI*dl*dl*2.0*(j22-0.5)*dl);
760           jr[i22+1][j22]+=fr1*wi2/(M_PI*dl*dl*2.0*(j22-0.5)*dl);
761           jr[i22+2][j22]+=fr1*wi3/(M_PI*dl*dl*2.0*(j22-0.5)*dl);
762
763           jr[i22][j22+1]+=fr2*wi1/(M_PI*dl*dl*2.0*(j22+0.5)*dl);
764           jr[i22+1][j22+1]+=fr2*wi2/(M_PI*dl*dl*2.0*(j22+0.5)*dl);
765           jr[i22+2][j22+1]+=fr2*wi3/(M_PI*dl*dl*2.0*(j22+0.5)*dl);
766         }
767
768         /*座標修正*/
769         double rr;
770         if(zp[n]>z_max+ld+dl*nz*0.05){
771           n_over++;
772           if(n_over>=npt){
773             npt=0;
774           }
775         }
776         if(rp[n]>r_max){
777           rp[n]-=r_max;
778         }else if(rp[n]<0.0){
779           rp[n]=fabs(rp[n]);
780           prp[n]=-prp[n];
```

```
781       }
782     }
783
784     return;
785
786   }
787
788   /*電場計算*/
789   void efield (void)
790   {
791     int i,j;
792
793     for(i=1;i<=nz+2*nd-1;i++){
794       ez[i][1]+=(4.0*ht[i][1]/dl-jz[i][1])*dt/eps;
795       if(i<=nd){
796         ez[i][1]*=(1-pow((ld-(i-0.5)*dl)/ld,2));
797       }else if(i>=nz+nd){
798         ez[i][1]*=(1-pow(((i-0.5)*dl-ld-z_max)/ld,2));
799       }
800     }
801     for(i=1;i<=nz+2*nd-1;i++){
802       for(j=2;j<=nr-1;j++){
803         if(f[i][j]==0||f[i+1][j]==0){
804           ez[i][j]+=((ht[i][j]*(j-0.5)-ht[i][j-1]*(j-1.5))
805                      /(dl*(j-1.0))-jz[i][j])*dt/eps;
806           if(i<=nd){
807             ez[i][j]*=(1-pow((ld-(i-0.5)*dl)/ld,2));
808           }else if(i>=nz+nd){
809             ez[i][j]*=(1-pow(((i-0.5)*dl-ld-z_max)/ld,2));
810           }
811         }
812       }
813     }
814
815     for(i=2;i<=nz+2*nd-1;i++){
816       for(j=1;j<=nr-1;j++){
817         if(f[i][j]==0||f[i][j+1]==0){
818           er[i][j]+=((ht[i-1][j]-ht[i][j])/dl-jr[i][j])*dt/eps;
819           if(i<=nd){
820             er[i][j]*=(1-pow((ld-(i-1)*dl)/ld,2));
821           }else if(i>nz+nd){
822             er[i][j]*=(1-pow(((i-1)*dl-ld-z_max)/ld,2));
823           }
824         }
825       }
826     }
827
828     return;
829   }
830
831   /*電子放出*/
832   void emission (void)
833   {
834     npt+=p_emission;
835
836     return;
837   }
```